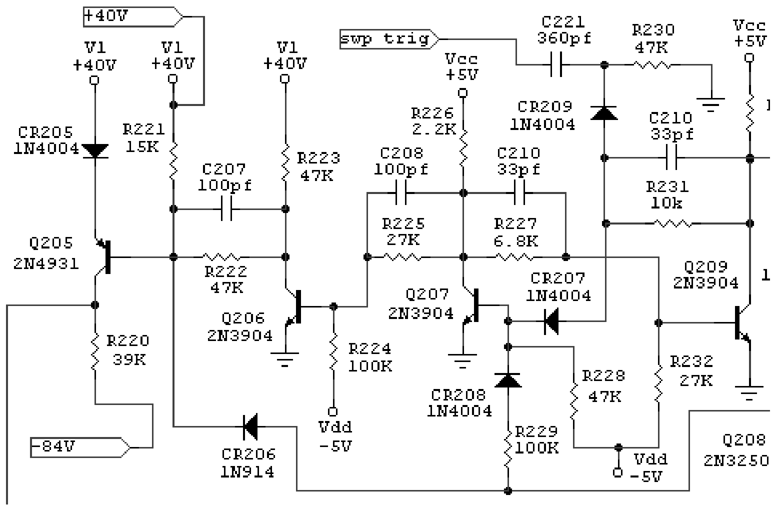
CircuitMaker® 2000

the virtual electronics lab™



CircuitMaker User Manual

advanced schematic capture
mixed analog/digital simulation

Revision A

Printed by Star Printery Pty Ltd

Table of Contents

Chapter 3: Tutorials

Chapter 4: Drawing and Editing Schematics

CHAPTER 1

Welcome to CircuitMaker

Introduction

Welcome to CircuitMaker, the most powerful, easy-to-use schematic capture and simulation tool in its class! Thank you for joining thousands of users who have discovered that CircuitMaker provides the features of "high-end" design software at a *fraction* of the cost.

Using CircuitMaker's advanced schematic capabilities, you can design electronic circuits and output netlists for TraxMaker and other PCB design tools and autorouters. You can also perform fast, accurate simulations of digital, analog and mixed analog/digital circuits using CircuitMaker's Berkeley SPICE3f5/XSpice-based simulator.

Required User Background

With just a minimum of electronics theory, you can successfully use CircuitMaker to design and simulate circuits. For beginners, CircuitMaker is perfect for learning and experimenting with electronics and circuit design. For advanced users, CircuitMaker's powerful analyses provide a sophisticated environment for testing and trying all the "what if" scenarios for your design. Best of all, you can accomplish more in less time than traditional prototyping methods.

Required Hardware/Software

- Microsoft® Windows NT4, 95, 98 or 2000
- PC with a Pentium class processor.
- 32MB of RAM
- 40MB hard disk space
- Desktop Area 800 x 600 pixels
- Color Palette 256 colors minimum
- CD ROM drive
- Mouse or compatible pointing device
- Any Windows-compatible printing device

Installing CircuitMaker

To install CircuitMaker,

1 Start your Windows operating system.

2 Insert the installation CD into your computer's drive.

3 Choose **Start** > **Run** from the Taskbar.

4 Browse to the drive containing the installation disk, select the Setup file, then choose **Open**, then choose **OK**.

5 Follow the installation instructions.

   Warning: If you are reinstalling or upgrading CircuitMaker be sure to install in a different directory to avoid writing over some of your existing work.

6 Double-click the CircuitMaker icon to launch the program.

7 If you are upgrading from an earlier version of CircuitMaker, see the next section *Updating from a Previous Version*.

Updating from a Previous Version

While upgrading from a previous version of CircuitMaker is a relatively painless process, you should take care when converting custom macro libraries and simulating existing circuits. When you load a pre-4.0 circuit file (identified by the .CIR extension), it will automatically be converted to the newer ASCII file format (which uses the .CKT extension).

If you have not added components to the library, simply follow the instructions above in *Installing CircuitMaker*. The new version will be installed in a new directory and you can delete the previous directory.

If you have added new device symbols to the macro library, see *Updating 32-bit Macro Libraries* or *Updating 16-bit Macro Libraries* below.

If you have added new SPICE models, see *Updating Model Libraries* below.

Warning: Be careful not to discard or overwrite your previous work.

Updating 32-Bit Macro Libraries

If you are upgrading from a 32-bit version of CircuitMaker and have created your own macro devices or symbols, follow these steps:

1 Install the new version of CircuitMaker as described earlier. Remember to install the new version in a different directory to avoid writing over your existing work. Run CircuitMaker.

2 Select **Macros** > **Macro Copier**.

3 Open the old USER.LIB file as the *Copy From* file. When asked if you want to list only the user defined devices, click Yes.

4 Open the USER.LIB file from your new CircuitMaker directory as the *Copy To* file.

5 Select the first device that you have created and click on the Copy button. Repeat for each additional device that you have created. Each device copied will be placed in the new USER.LIB file. You may be prompted for information regarding the simulation mode for which a device is intended. If the device can be used in digital simulations, check the Digital box; if it can be used in analog simulations, check the Analog box. If it can be used in either simulation mode, check both boxes.

Updating 16-Bit Macro Libraries

If you are upgrading from a 16-bit version of CircuitMaker and have created your own macro devices or symbols, follow these steps:

1 Install the new version of CircuitMaker as described earlier. Be sure to install the new version into a different directory to avoid writing over your existing work.

2 Run the BTOA file conversion utility.

3 Select **File** > **Convert Library**. Open the USER.LIB file from your previous CircuitMaker directory. Save the new file as USERLIB.ASC.

4 Run CircuitMaker.

5 Select **Macros** > **Convert ASCII Library**. Load the file USERLIB.ASC that you just created. Save the new file as NEWUSER.LIB.

6 Select **Macros** > **Macro Copier**.

7 Open the NEWUSER.LIB file as the *Copy From* file. When asked if you want to list only the user defined devices, click Yes.

8 Open the USER.LIB file from your new CircuitMaker directory as the *Copy To* file.

9 Select the first device that you created and click on the Copy button. Repeat for each additional device that you have created. Each device copied will be placed in the new USER.LIB file. You may be prompted for information regarding the simulation mode for which a device is intended. If the device can be used in digital simulations, check the Digital box; if it can be used in analog simulations, check the Analog box. If it can be used in either simulation mode, check both boxes.

Note: Importing of bitmaps and unconverted metafiles is no longer supported for creating devices in CircuitMaker. If you have created devices of this type, the symbol will not be available in CircuitMaker 2000 (the symbol will be replaced by a rectangle). These symbols may be redrawn with the Symbol Editor.

Updating Model Libraries

If you have added or modified any of the .MOD, .SUB or .LIB files from a previous version, you must make these same additions or modifications to the files in the new Models directory. *Since many of these .MOD, .SUB and .LIB files in CircuitMaker 2000 contain new information, it is recommended that any changes you made previously be done manually (don't just copy your old file over the top of the new one), to avoid any possible loss of data.*

If you have created any user-defined symbols to which SPICE models have been linked, you must remember that each of these user-defined symbols has a corresponding .MOD or .SUB file. Be sure to copy these files into the new Models directory.

If you have used CircuitMaker's automatic linking feature (accessed through the Model Data button in Macro Utilities dialog box) to link a symbol to a user-added SPICE model in a .LIB file, you must do one of two things:

1 Reenter the information using the Macro Utilities dialog box just like before.

OR

2 Copy the linking information that was automatically placed in the .MOD or .SUB files into the new files. This information would usually be located at the end of the .MOD or .SUB file that corresponds to the symbol.

Updating Pre-5.0 Digital Circuits for Analog Simulation

Digital circuits created in pre-5.0 versions of CircuitMaker will still run in Digital Logic Simulation mode. However, if you want to run them in Analog Simulation mode, you should be aware of the following:

- Analog is not a free-running simulation mode.
- Analog simulations require proper use of Vcc and Ground connections to digital devices.
- Analog simulations require designations of all devices.
- The digital devices saved in old circuits do not contain SPICE data for simulation. These must be replaced by new digital SimCode devices. To do this, delete the existing devices and replace them with new devices from the current library.
- The Pulser is a digital only device. It must be replaced by a Data Sequencer or a Signal Generator.
- Devices which are animated in Digital Logic Simulation mode are not animated in Analog Simulation mode.

Multi-User (Project) Installations

You can configure CircuitMaker to support “Projects” for multiple users, each user having access to separate libraries and preferences. Project installations are possible whether on a network or on a stand-alone system. Each user must be assigned a separate directory from which the CircuitMaker preferences, libraries and circuit files can be accessed. This same method may be used by a single user who needs to access multiple projects, each with separate libraries and preferences.

Note: A site license is required for network installation.

Setting Up Multiple Projects

1 Install and run CircuitMaker as described above to initialize the default file paths, then exit.

2 Create a separate directory for each user or project.

3 Place a copy of the Cirmaker.dat file in each user’s project directory. This file contains the Preferences data which allows each user to specify their own circuit and library paths and other circuit and program preferences.

4 If a user will need to make modifications or additions to the Macros library (changes that must not affect other users), place a copy of the User.lib, Devicedb.dat, Symboldb.dat and Hotkeysdb.dat files into that user’s project directory. The Devicedb.dat, Symboldb.dat and Hotkeysdb.dat files must always be placed in the same directory as User.lib.

5 If a user will need to make modifications or additions to the Spice models or subcircuits, place a copy of the entire Models directory into that user’s project directory.

6 If accidental modification of files which are common to multiple users is a concern, see your network administrator for details on how to protect these files from modification.

7 In Windows 95/98/NT4, Right-click the Start button and select **Open**. Browse to the CircuitMaker icon, right-click it, and Select **Properties**. Click the Shortcut tab.

8 Add the word **project** onto the string in the Target edit field. For example:

```
"c:\...\cirmaker.exe" project
```

then click OK. This enables the Select Project dialog box (explained later in this section).

9 Run CircuitMaker.

10 The Select Project dialog box appears, allowing you to find the individual project directories. Browse the directories to find one individual's Cirmaker.dat file and click OK.

11 When CircuitMaker has loaded, select **Options** > **Library Location**.

12 Change the Circuits Directory path to that of the project directory. This is where that individual's circuit files (*.ckt) will be stored.

13 If there is a copy of the User.lib file in the project directory, change the User Library File path to that of the individual's directory. This allows the individual to make changes/additions to the Macros library.

14 If there is a copy of the Models directory in the project directory, change the Model Directory path to that of the individual's directory. This allows the individual to make changes/additions to the Spice models and subcircuits.

15 Click OK to exit the Library Location dialog box.

16 Exit CircuitMaker.

17 CircuitMaker is now completely configured for one user. Repeat steps 9-16 to configure CircuitMaker for each individual user.

Accessing a Project

1 Run CircuitMaker.

2 The Select Project dialog box appears, allowing you to find your personal project directory. Browse the directories to find your Cirmaker.dat file and click on the OK button.

Technical Support

Protel is dedicated to producing only the finest quality software and supporting customers after the initial purchase. If you encounter problems while using CircuitMaker or just need general help, contact us via phone, FAX or Email and we'll provide prompt and courteous support.

***NOTE:** Please be prepared to provide your name and registration number (found on the back of the User Manual or on the CD jacket) when contacting us.*

Telephone:	801-224-0433
FAX:	801-224-0545
Internet:	http://www.circuitmaker.com
Email:	support@circuitmaker.com

Future versions of CircuitMaker are planned so please feel free to write and let us know what features or additions you would like to see. Our goal is to provide a product that will meet your needs and expectations, so feedback from you the end user is essential!

About the Documentation

The CircuitMaker documentation includes both a *User Manual* and a *Device Library Guide*. This *User Manual* has been designed to guide you through CircuitMaker's many features and simplify the retrieval of specific information once you have a working knowledge of the product. The *Device Library Guide* outlines the symbols and device models that are included with CircuitMaker.

The manual assumes that you are familiar with the Windows desktop and its use of icons, menus, windows and the mouse. It also assumes a basic understanding about how Windows manages applications (programs and utilities) and documents (data files) to perform routine tasks such as starting applications, opening documents and saving your work.

Manual Conventions

The following conventions are used to identify information needed to perform CircuitMaker tasks:

Professional Edition

This manual contains information for both the standard and professional editions of CircuitMaker 2000. Those features described in this manual which are only available in the professional edition are highlighted by the banner at the left.

Step-by-step instructions for performing an operation are generally numbered as in the following examples:

1 Choose **File** > **Save**.

This means "choose the **File** menu, then choose **Save**."

2 Select the **Arrow Tool** on the Toolbar.

Menu names, menu commands, and Toolbar options usually appear in bold type as are text strings to be typed:

3 Type the Label-Value: **220K**.

Notes, Hints and Tips are written in the margins for better visibility.

This manual also includes some special terminology—words that are either unique to schematic capture and circuit simulation or have some specific meaning within CircuitMaker. Such terms are italicized when first introduced.

Using On-line Help

You can access CircuitMaker's on-line Help file (Cirmaker.hlp) in several ways.

From the Help Menu

To access Help from the Help menu,

1 Choose **Help** > **Contents**.

2 Choose the **Contents** tab to see an overview of all Help topics arranged hierarchically.

OR

Choose the **Index** tab, then enter a keyword to look up a specific Help topic.

OR

Choose the **Find** tab to find Help topics that contain the word you are looking for.

Context-sensitive Help

To access context-sensitive Help,

1. Open a dialog box for which you need help.

   OR

   Click on the item with the **Arrow Tool** so it is selected.

2. Press **F1** to display Help for that dialog box or item.

From the Help File Directly

Even when CircuitMaker is not running, you can view the Help file by double-clicking its icon in the CircuitMaker program group.

Where to Go from Here

Once you have mastered a few Windows basics, you'll be ready to learn CircuitMaker. Use the following table to help you get around this User Manual.

Topic	Where to Go
CircuitMaker Basics	Chapter 2
Tutorials	Chapter 3
Drawing and Editing Schematics	Chapter 4
Simulation	Chapters 5, 6
Exporting Files	Chapter 7
Fault Simulation	Chapter 8
CircuitMaker Menus	Chapters 9–15
Advanced SPICE Tips	Chapter 16
Creating New Devices	Chapter 17
SimCode	Chapter 18

CHAPTER 2

Getting Started

CircuitMaker Basics

This chapter gives you an overview of the CircuitMaker workspace, conventions, preferences, shortcuts, and Hotkeys. It's a great place to start if you need some guidance before using CircuitMaker to draw, edit, test, and simulate electronic circuits.

Starting CircuitMaker

If you have installed CircuitMaker on your hard disk, you're ready to run the program.

1 Open the **Start** menu on the Taskbar.

2 Choose **Programs** > **CircuitMaker 2000**.

3 Choose the **CircuitMaker** program.

You can also create a shortcut for CircuitMaker and have the icon display on your desktop all the time.

CircuitMaker Workspace

When you start CircuitMaker, a blank Schematic Window appears. The Schematic Window is where you place devices that represent real life components such as resistors, transistors, power supplies, etc. The CircuitMaker workspace also includes the Toolbar, Menu Bar, Status Bar, Panel and the Analysis Window.

After placing devices exactly where you want, you simply wire them together. The wiring lines you draw form intelligent links between devices, which then allow the circuit to be simulated, tested, and analyzed using CircuitMaker's powerful simulator. Figure 2.1 shows the CircuitMaker workspace including the Menu bar, Toolbar, Status bar, Panel, Schematic Window and Analysis Window.

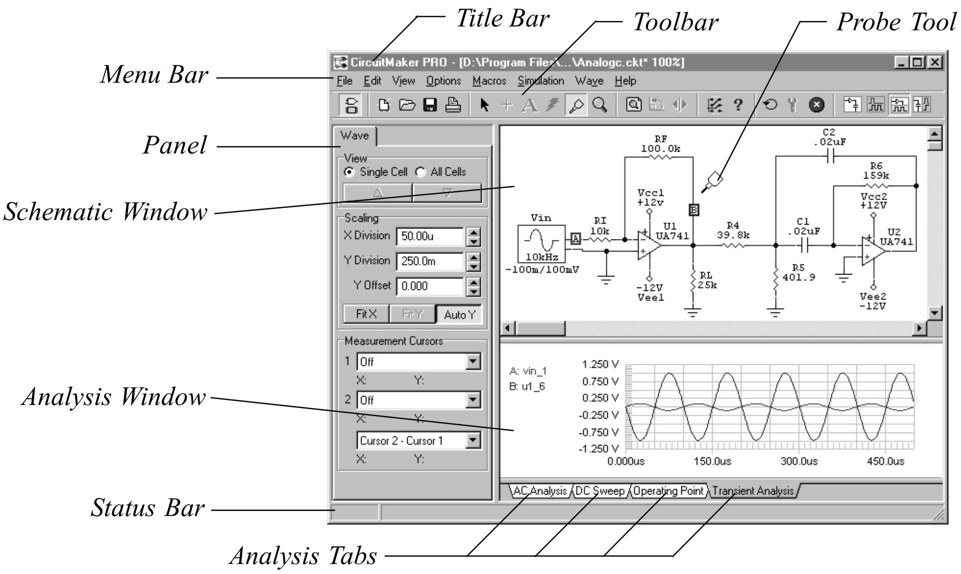


Figure 2.1. The CircuitMaker workspace.

Connectivity

An important feature of CircuitMaker is the way electrical connections between the elements in your design are recognized.

The concept of *connectivity* is the key to using CircuitMaker to draw and simulate electronic circuits. The program stores connection information for simulation, and it is also used for creating and exporting *netlists* into TraxMaker or other pcb layout programs to create a working printed circuit board (PCB).

About the CircuitMaker Windows

The CircuitMaker Window has three separate sections: the Schematic Window, the Analysis Window and the Panel. The Schematic window is where the schematics are drawn. One circuit file at a time can be opened into this window. The Analysis Window is where the simulation results are displayed. This window can consist of multiple charts which are selected by clicking on the tabs located at the bottom of the window. One tab is available for each analysis type that is enabled. The Panel has tabs across the top which are used to select controls which are relevant to the available windows. This is where you find the parts browser and search, the analog waveform controls and the digital logic mode options. The windows are separated by draggable splitter bars which allow you to resize the windows as needed.

Anatomy of a Schematic Drawing

Figure 2.2 shows a basic schematic, including device symbols, label-values and designations, wires, and pin dots.

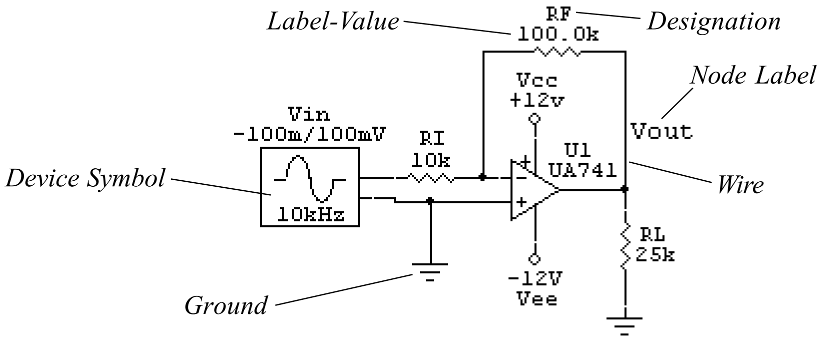


Figure 2.2. CircuitMaker's straightforward approach makes it easy to identify each part of a schematic drawing.

CircuitMaker Conventions

If you are experienced with Windows applications, you already know how to start and quit CircuitMaker, select menus using the mouse, save your work, and locate and organize your documents. On the other hand, CircuitMaker has special features that are not common to other Windows applications. These options let CircuitMaker perform some of the special tasks unique to circuit design.

CircuitMaker Files

CircuitMaker includes a number of special purpose files in addition to the CircuitMaker application. The following table lists the various types of files you will use by file extension.

.CKT	Schematic (or Circuit) files
.DAT	Data files (Hotkeys; device library classifications)
.LIB	Device library files
.MOD	Model files
.SUB	Subcircuit files
.SDF	Waveform display setup files

Accessing Tools and Features

This section shows you the fundamental tools and processes used to draw schematics.

Task Overview

Using CircuitMaker involves six basic procedures: 1) *placing* devices (such as resistors, transistors, power supplies, and grounds) in the workspace; 2) *repositioning* devices; 3) *editing* devices with precise values and parameters; 4) *deleting* devices (if necessary); 5) wiring devices together; 6) *simulating* and *testing* the circuit.

Using the Toolbar

You can perform many CircuitMaker tasks using the buttons on the Toolbar, which is conveniently located at the top of the workspace.

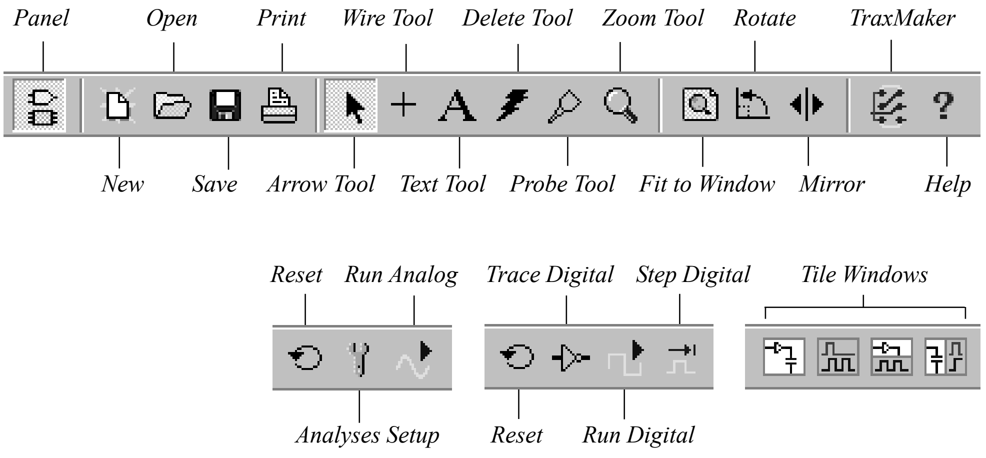


Figure 2.3. The Toolbar puts many of CircuitMaker's features just a mouse click away.

The table on the following page briefly describes each button and tool on the Toolbar. Generally, a Tool lets you apply a specific action, whereas a Button performs a general function. For more details about the drawing and editing tools, see *Chapter 4: Drawing and Editing Schematics*. For more information about the simulation tools, see *Chapters 5: Digital Logic Simulation* and *Chapter 6: Analog/Mixed Signal Simulation*.

Tool or Button	**Lets You**
Panel	Show/Hide the Panel. Frees up more space for drawing, simulation.
New	Create a new schematic.
Open	Open an existing schematic.
Save	Save the current schematic.
Print	Print the schematic.
Arrow Tool	Select, move and edit devices, wires and text. Also used to place wires (when Arrow/Wire option is checked).
Wire Tool	Place wires to connect devices in the circuit (+Shift to place bus wires).
Text Tool	Add text to the circuit.
Delete Tool	Delete devices, wires and text (Right-click to delete wire segments, Shift-click to snip wires).
Probe Tool	Observe/plot data at any point(s) in the circuit (context-sensitive).
Zoom Tool	Magnify and reduce the circuit (Shift-click to reduce).
Fit to Window	Change the Display Scale to fit the entire schematic in the Schematic Window.
Rotate	Rotate one or more selected devices.
Mirror	Mirror one or more selected devices.
TraxMaker	Automatically create a PCB netlist and launch TraxMaker.
Help	Display Help file information on selected item.
Reset	Initialize analog and digital simulations.

Analyses Setup	Open the Analyses Setup dialog box. (analog/mixed-signal simulation mode only).
Run Analog	Start and stop analog/mixed-signal simulations. Changes to a Stop icon when pressed.
Trace Digital	Interactively display the logic state of all nodes in Digital simulation mode.
Run Digital	Run and pause digital simulations. (Changes to a Pause icon when pressed).
Step Digital	Single-step digital logic simulations (Set step size in Digital Panel).
Tile Windows	Tile the Schematic and Analysis Windows to one of four views.

Using the Mouse

As in other Windows applications, CircuitMaker uses the mouse for clicking, selecting and dragging. When moving the mouse, a corresponding selection tool (or cursor) movement occurs on the screen.

The familiar "pointer" **Arrow Tool** is used for standard Windows operations, such as choosing from menus and dialog boxes.

You can return to the standard **Arrow Tool** at any time by selecting the tool from the Toolbar, or right-clicking on the schematic background and selecting **Arrow**.

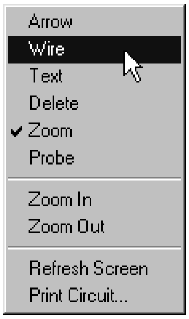


Figure 2.4. Right-click the mouse on different areas to access many CircuitMaker features.

Right-Click Pop-Up Menus

You can right-click (click with the right mouse button) in different areas of the CircuitMaker workspace to open various pop-up menus (see Figure 2.4). The items listed in the pop-up menu vary depending on where you right-click. The following locations and circumstances will each access a different pop-up menu:

- Schematic background
- Device
- Wire

- Text object created with the Text Tool.
- Group of selected items (pop-up menu depends on types of items that you select)
- Waveform label
- All Cells view cell
- Anywhere else in the Analysis window

Hotkeys

For quick and easy device placement, CircuitMaker offers up to sixty user-definable Hotkeys that let you place commonly-used devices with a single keystroke.

For example, press the **r** key to get a 1K resistor. Or press **b** for a 10V battery. These assignments can be changed and customized so that the parts you use most are right at your fingertips. See *Chapter 4: Drawing and Editing Schematics* to learn how to customize Hotkeys.

Shortcut Keys

Command key or "shortcut" keys let you select menu commands directly. The following table lists the available short cut keys in CircuitMaker.

Keystroke	**What it Does**
Ctrl+A	Select All
Ctrl+C	Copy
Ctrl+D	Duplicate
Ctrl+E	Expand Macro
Ctrl+F	Find and Select
Ctrl+H	New Macro
Ctrl+K	Device Display Data
Ctrl+L	Schematic Display Data
Ctrl+M	Mirror
Ctrl+N	New
Ctrl+O	Open
Ctrl+P	Print Schematic
Ctrl+Q	Reset Simulation
Ctrl+R	Rotate
Ctrl+S	Save
Ctrl+U	Macro Utilities
Ctrl+V	Paste

Ctrl+X	Cut
Ctrl+Z	Undo
Alt+A	Arrow Tool
Alt+D	Delete Tool
Alt+P	Probe Tool
Alt+T	Text Tool
Alt+W	Wire Tool
Alt+Z	Zoom Tool
F1	Context Sensitive Help
F2	Display Scale
F3	Normal Size/Position
F4	Zoom to Fit
F5	Schematic Options
F6	Collapse Device Tree
F7	Refresh Screen
F8	Analyses Setup
F9	Step Logic Simulation
F10	Run/Stop Simulation
F11	Trace Logic Simulation
End	Refreshes the screen.
Esc	Aborts the current operation.
Page Up	Enlarges the display (zooms in).
Page Down	Reduces the display (zooms out).
Delete	Deletes the current selection.
Home	Centers the screen around the cursor
Arrow Keys	Nudges a selected device (by pressing the Left, Right, Up, or Down Arrow keys).
Shift+Insert	Moves the currently selected group of items.

Saving Schematic Options

CircuitMaker stores many settings such as program and circuit defaults. You specify your own preferred option settings using the Schematic Options dialog box (choose **Options** > **Schematic**) as shown in Figure 2.5. See *Chapter 12: Options Menu* for more information.

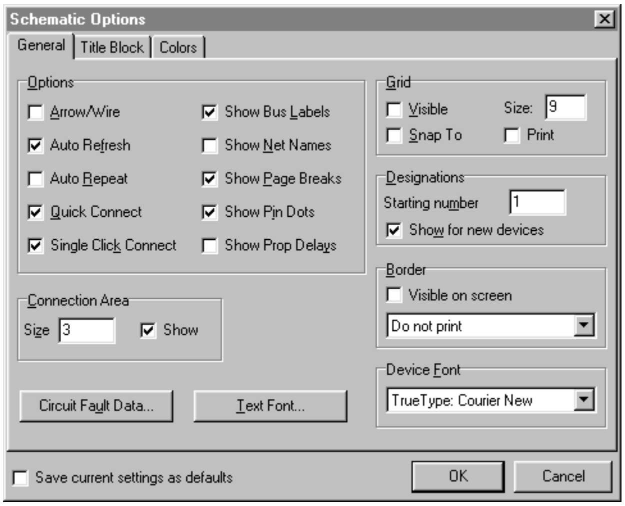
To save the option settings as defaults,

1 Choose **Options** > **Schematic**.

2 Make the desired changes.

3 Enable the **Save current settings as defaults** checkbox, then choose OK.

Note: Some of these options are saved with each schematic file. In these cases, the defaults only apply when you create a new schematic file.



Arrow/Wire ☐
Auto Refresh ☑
Auto Repeat ☐
Quick Connect ☑
Single Click Connect ☑
Show Bus Labels ☑
Show Net Names ☐
Show Page Breaks ☑
Show Pin Dots ☑
Show Prop Delays ☐
Visible ☐
Snap To ☐
Size: 9
Print ☐
Starting number 1
Show for new devices ☑
Visible on screen ☐
Do not print
TrueType: Courier New
Size 3
Show ☑
Save current settings as defaults ☐

Figure 2.5. Use the Schematic Options dialog box to manage CircuitMaker settings.

Basic .CKT File Management

This section explains the basic CircuitMaker file management procedures.

Starting, Saving & Closing a .CKT File

The features you will use most often are New, Save, and Save As.

1 Choose **File** > **New** to start a new file.

2 Choose **File** > **Save** if you've already established a filename.

Or

Choose **File** > **Save As** to give the file a filename. This is the way to copy a .CKT file.

3 Choose **File** > **Close** > **Yes** to save and exit a .CKT file without exiting CircuitMaker.

OR

Choose **File** > **Exit** > **Yes** to exit CircuitMaker and save your work.

Opening and Re-Opening a .CKT File

1 Choose **File** > **Open**.

2 Select the file with the .CKT extension that you want to open, then choose **Open**.

You can open any of the last 8 .CKT files you've had open.

1 Choose **File** > **Reopen**.

2 Select the file you want to reopen.

Reverting to Previously Saved File

If you made changes to a .CKT file that you don't wish to save, you may "revert" to the last version of the .CKT file you saved under the same filename.

To revert to the previously saved file,

1 Choose **File** > **Revert**.

CHAPTER 3
Tutorials

This chapter covers step-by-step processes in the order you would normally perform them. Working through the following examples will provide a general understanding of the way CircuitMaker works and will illustrate that there is often more than one way of doing a task.

Tutorial 1: Drawing a Schematic

This tutorial covers the following topics:

- Using the Browse tab in the Panel
- Selecting a transistor
- Selecting resistors
- Selecting a +V and ground device
- Changing resistor and transistor label values
- Wiring the circuit

Using the Browse tab in the Panel

Drawing a schematic is as easy as pointing and clicking with the mouse. Let's walk through a simple example by constructing the circuit shown in Figure 3.1.

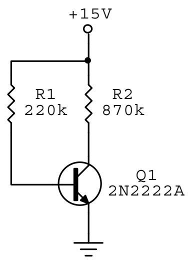
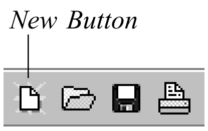


Figure 3.1. This is a very simple circuit that consists of one transistor, two resistors, a power source, and a ground.



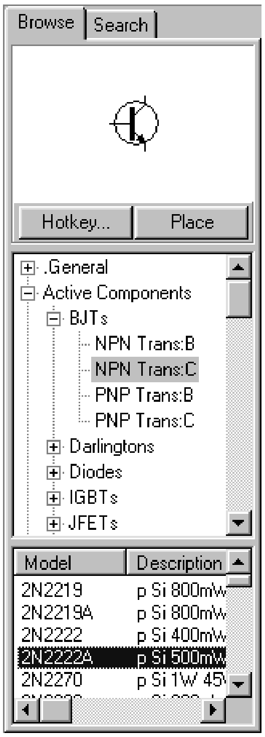
1 Begin by starting the CircuitMaker program and, if necessary, clearing the workspace by choosing **File** > **New** or clicking the **New** button on the Toolbar.

2 Choose the **Browse** tab in the Panel to display the device browse tree, pictured in Figure 3.2.

Note: Throughout this tutorial, the location of a device in the browse tree and its default Hotkey (if applicable) is indicated using this format:

[major device class/minor device class/device symbol] (default Hotkey)

For example, a battery is found at [.General/Sources/Battery] (b). By simply pressing a Hotkey (the letter “b” in this example) you can quickly select and insert a device into the workspace.

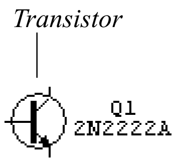


```
+ .General
- Active Components
  - BJTs
      NPN Trans:B
      NPN Trans:C
      PNP Trans:B
      PNP Trans:C
  + Darlingtons
  + Diodes
  + IGBTs
  + JFETs
```

Model	Description
2N2219	p Si 800mW
2N2219A	p Si 800mW
2N2222	p Si 400mW
2N2222A	p Si 500mW
2N2270	p Si 1W 45

Figure 3.2. Use the Browse tab in the Panel to pick a device from a large library of devices. Notice that, in this example, the 2N2222A transistor is selected.

***Tip:** You can quickly restore the device browse tree to its original “closed” condition by selecting **View** > **Collapse Device Tree (F6)**.*



Placing a Transistor

Begin the circuit by selecting the 2N2222A transistor [.General/BJTs/NPN Trans].

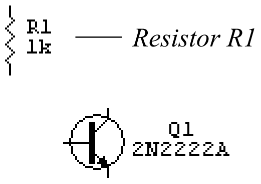
1 Select **.General** / **BJTs** / **NPN Trans:C** in the Browse tree. Select the **2N2222A** transistor in the Model list.

2 Click **Place** to select this device from the library.

You can also click the **Search** tab on the Panel, type **2n2222a**, and click **Find** to quickly find the part.

3 Position the transistor at about mid-screen and then click the left mouse button once.

Notice that the transistor is placed on the workspace and no longer follows the mouse (see picture at left).



Placing the Resistors

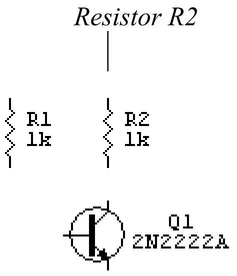
The next procedure involves placing two resistors.

1 Select a **Resistor** [Passive Components/Resistors/Resistor] (r) by pressing the **r** Hotkey on the keyboard. Notice that the resistor is oriented horizontally and moves around the screen with the mouse.

2 Press the **r** key again (or click the **Right** mouse button) to rotate the device 90°.

3 Drag the resistor above and to the left of the transistor and click the **Left** mouse button once to place it.

This is resistor R1. Don't worry about the value yet.



4 Place a second resistor directly above the transistor. This is resistor R2.

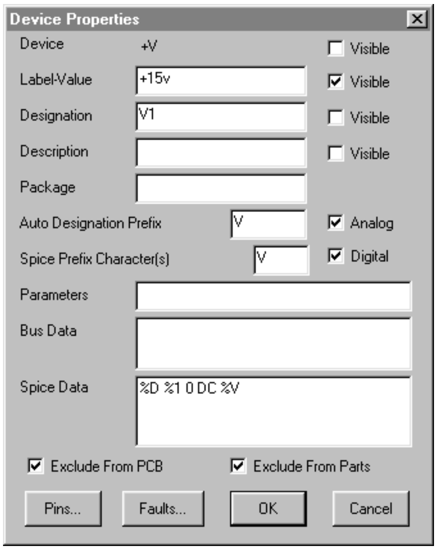
Placing +V and Ground Devices

Now you'll place a voltage source and change its settings.

1 Select a **+V** [.General/Sources/+V] (1) by pressing the **1** (number one) Hotkey. Place it above resistor R2.

2 Select a **Ground** [.General/Sources/Ground] (0) by pressing the **0** (zero) Hotkey. Place it below the transistor.

3 Double-click the **+V** device using the **Left** mouse button to open the Device Properties dialog box, pictured in Figure 3.3.



Device: +V ☐ Visible
Label-Value: +15v ☑ Visible
Designation: V1 ☐ Visible
Description: ☐ Visible
Package:
Auto Designation Prefix: V ☑ Analog
Spice Prefix Character(s): V ☑ Digital
Parameters:
Bus Data:
Spice Data: %D %1 0 DC %V
☑ Exclude From PCB ☑ Exclude From Parts

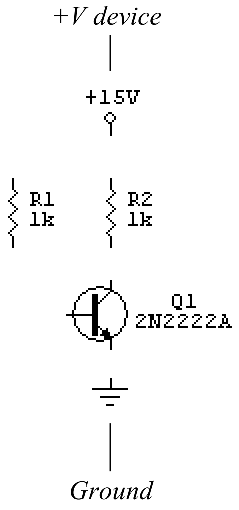


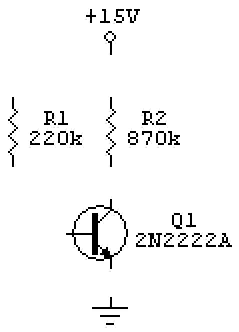
Figure 3.3. The Device Properties dialog box lets you change a wide range of device settings.

4 Change the Label-Value field to read **+15V**.

5 Click once on the topmost **Visible** check box. This causes the **+V** name to be hidden on the schematic.

6 Click once on the third **Visible** check box from the top. This causes the **V1** designation to be hidden on the schematic. Click OK.

Changing Resistor Label-Values



Now try the same editing procedure on the resistors.

1 Double-click resistor R1.

2 Change the Label-Value field to read **220k**, then click OK.

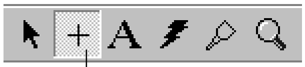
3 Double-click resistor R2.

4 Change the Label-Value field to read **870k**, then click OK.

5 If necessary, drag the devices and labels around with the mouse to place them in convenient locations.

Wiring the Circuit Together

Now it's time to hook up these devices into a working circuit by wiring them together.



Wire Tool

1 Select the **Wire Tool** from the Toolbar.

2 Place the cursor on the emitter pin of the transistor (the pin with the arrow.)

   When the cursor gets close to the pin, a small rectangle appears.

3 Click and hold the left mouse button, then drag the wire to the pin of the Ground symbol.

4 Release the mouse button to make the connection.

5 Place the cursor on the bottom pin of R2, and then click and hold the mouse button to start a new wire.

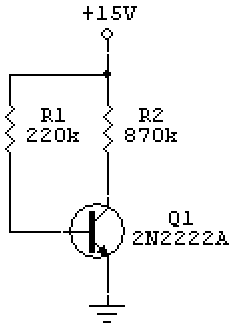
6 Drag the end of the wire to the collector pin of the transistor and release the mouse button.

7 Connect a wire from the top pin of R2 to +15V.

8 Connect another wire from the bottom pin of R1 to the base of the transistor.

9 Finally, connect a wire from the top pin of R1 to the middle of the wire which connects +15V to R2.

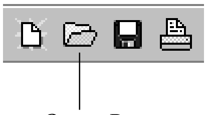
   You can move device and wire positions by dragging them with the Arrow Tool.



The completed schematic

Tutorial 2: Digital Logic Simulation

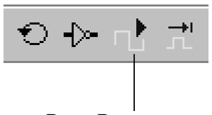
The best way to see how the digital simulation works is to load an example circuit and try some commands.


Open Button

1 Click the **Open** button in the Toolbar.

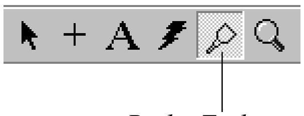
2 Select the SIM.CKT file from the list of available circuits.

The SIM.CKT circuit contains several mini-circuits and is useful for demonstrating CircuitMaker's digital simulation features.


Run Button

3 Click the **Run** button on the Toolbar to start simulation.

You know that simulation is running when you see a Hex Display showing a count sequence.


Probe Tool

4 Select the **Probe Tool** from the Toolbar and touch its tip to the wire just to the left of the label "Probe Wire to the Left." The letter **L** will be displayed in the Probe Tool.

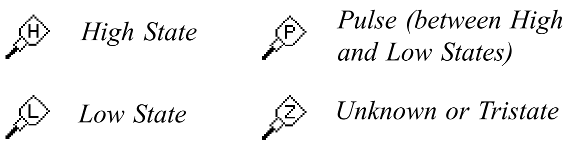
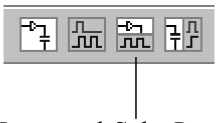


Figure 3.4. This figure describes the meaning of the letters that might appear in the Probe Tool, depending on what part of the schematic you touch with the tool.

5 Move the tip of the **Probe Tool** to the Logic Switch labeled "Toggle Switch" and click near its center. The Logic Display connected to the output of this mini-circuit should then start to toggle on and off rapidly.

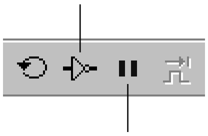

Horizontal Split Button

6 Click the **Horizontal Split** button on the Toolbar to open the digital Waveforms window. Each node in the circuit that has a SCOPE device attached is charted in this window.

7 Select **Simulation** > **Active Probe**, then run the simulation again.

A new waveform called Probe displays in the Waveforms window. Watch what happens to this waveform as you move the Probe Tool around the circuit.

Trace Button



Pause Button

8 Click the **Trace** button in the Toolbar to see the state of every wire in the circuit as the state changes. A red wire indicates a high state, a blue wire indicates a low state.

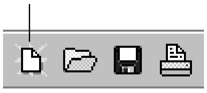
9 Click the **Pause** button in the Toolbar to stop simulation.

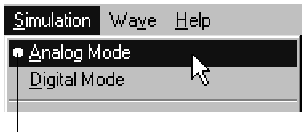
Tutorial 3: Analog Simulation

The best way to get acquainted with CircuitMaker's analog simulation is to build a few simple circuits, set up the analyses, and run the simulations. This tutorial covers:

- Simple circuit analysis
- Simulating a simple AC circuit
- More circuit simulation
- Setting up the analyses
- Running the simulation
- Mixed-signal simulation

New Button





Bullet indicating that Analog Mode is enabled.

Simple Circuit Simulation

Let's begin with a simple DC circuit:

1 Click the **New** button on the Toolbar. This creates an untitled schematic window.

2 Press **F6** if needed to collapse the device browse tree.

3 In the Simulation menu, make sure that **Analog Mode** is enabled rather than **Digital Mode**.

4 Draw the circuit as shown in Figure 3.5, using the following devices:

  - 1 Battery [.General/Sources/Battery] (b)
  - 1 Ground [.General/Sources/Ground] (0 (zero))
  - 2 Resistors [.General/Resistors/Resistor] (r)

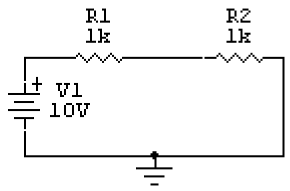
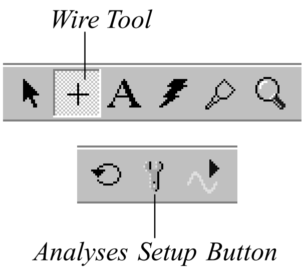


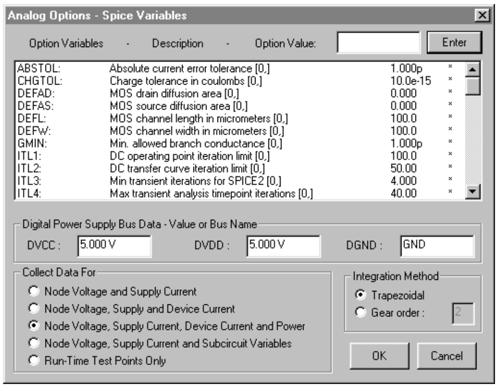
Figure 3.5. A quick way to access the devices for this circuit is to type the Hotkey shortcuts (explained in Chapter 2) for the corresponding devices.

Note: Every analog circuit must have a Ground and every node in the circuit must have a DC path to ground.



5 Use the **Wire Tool** to wire the circuit together.

6 Click on the **Analyses Setup** button in the Toolbar, then click the **Analog Options** button to display the dialog box shown in Figure 3.6.



Analog Options - Spice Variables

Option Variables - Description - Option Value: Enter

Option Variables	Description	Option Value	
ABSTOL:	Absolute current error tolerance [0,]	1.000p	*
CHGTOL:	Charge tolerance in coulombs [0,]	10.0e-15	*
DEFAD:	MOS drain diffusion area [0,]	0.000	*
DEFAS:	MOS source diffusion area [0,]	0.000	*
DEFL:	MOS channel length in micrometers [0,]	100.0	*
DEFW:	MOS channel width in micrometers [0,]	100.0	*
GMIN:	Min. allowed branch conductance [0,]	1.000p	*
ITL1:	DC operating point iteration limit [0,]	100.0	*
ITL2:	DC transfer curve iteration limit [0,]	50.00	*
ITL3:	Min transient iterations for SPICE2 [0,]	4.000	*
ITL4:	Max transient analysis timepoint iterations [0,]	40.00	*

Digital Power Supply Bus Data - Value or Bus Name

DVCC : 5.000 V DVDD : 5.000 V DGND : GND

Collect Data For
- Node Voltage and Supply Current
- Node Voltage, Supply and Device Current
- [✓ Node Voltage, Supply Current, Device Current and Power]
- Node Voltage, Supply Current and Subcircuit Variables
- Run-Time Test Points Only

Integration Method
- [✓ Trapezoidal]
- Gear order : 2

OK Cancel

Figure 3.6. The third Collect Data For option (selected) lets you measure voltage, current, and power with the Probe Tool.

7 From the **Collect Data For** group box, select the third option, **Node Voltage, Supply Current, Device Current and Power** then click OK to exit Analog Options.

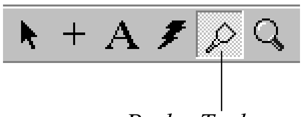
This option lets you to take current and power measurements with the Probe Tool.

8 Click the **Run Analyses** button to start the simulation.

OR

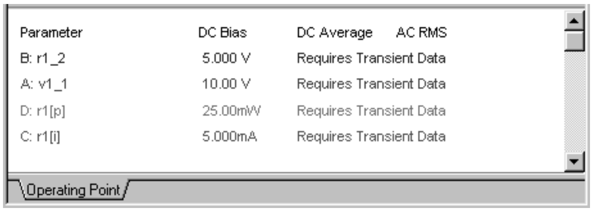
Click **Exit** and click the **Run** button on the Toolbar.

The CircuitMaker Window splits, placing the Schematic Window in the top half, the Analysis Window in the bottom half. The Operating Point chart appears in the Analysis Window. The Operating Point chart displays the DC Bias, DC Average and AC RMS values for each point that is probed on the schematic.


Probe Tool

9 Click the wire connected to the + terminal of the battery with the tip of the **Probe Tool**. Notice that the letter **V** appears on the Probe Tool when you move it over a wire indicating a voltage measurement.

The DC voltage at that node (10.00 V) appears in the Operating Point chart under DC Bias. All voltage measurements are referenced to ground. Notice that DC Average and AC RMS are not available at this time because Transient Analysis is not enabled. See Figure 3.7. DC Bias values are a result of the Operating Point Analysis. This simple analysis assumes all inductors to be shorts and all capacitors to be opens. It is ideal for simulating DC only circuits which do not need to show any type of charge/discharge transitions.



Parameter	DC Bias	DC Average	AC RMS
B: r1_2	5.000 V	Requires Transient Data	
A: v1_1	10.00 V	Requires Transient Data	
D: r1[p]	25.00mW	Requires Transient Data	
C: r1[i]	5.000mA	Requires Transient Data	

Figure 3.7. The Operating Point chart shows DC Bias, DC Average and AC RMS values for each point that is probed on the schematic. All voltage measurements are referenced to ground.

10 Hold down the Shift key and click the wire connected between the two resistors.

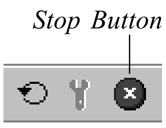
The DC voltage at that node (5.000 V) appears in the Operating Point chart.

11 Hold down the Shift key and click the pin of resistor R1. Notice that the letter **I** appears on the Probe Tool when it is over a device pin indicating a current measurement.

The current through that device (5.000mA) appears in the Operating Point chart.

12 Hold down the Shift key and click directly on resistor R1. Notice that the letter **P** appears on the Probe Tool when over a device indicating a power measurement.

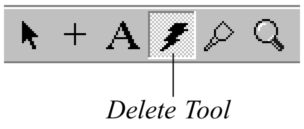
The power dissipated by that resistor (25.00mW) appears in the Operating Point chart.



13 Click the **Stop** button on the Toolbar to stop the simulation and return to editing mode.

Creating a Simple RC Circuit

Now let’s replace one of the resistors with a capacitor to create a simple RC circuit where you can see the charging of the capacitor. Transient Analysis begins its simulation in a stable DC condition where the capacitors are already charged. Since you want to see the capacitor charging from time zero, you must set the initial condition of the capacitor to 0V.



1 Using the **Delete Tool** on the Toolbar, delete R2.

2 Replace the resistor with a Capacitor [.General/Capacitors/Capacitor] (c) and wire it into place.

3 Select an .IC device [Analog/SPICE Controls] (I) and connect it to the wire between the resistor and capacitor.

This will set an initial condition of 0V on the capacitor for the analysis (the other side of the capacitor is connected to ground, so there is 0V across it at start-up). Your circuit should now look like the one pictured in Figure 3.8.

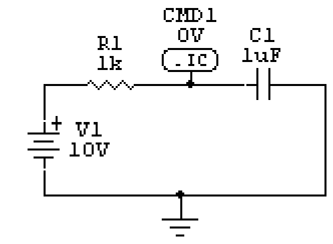


Figure 3.8. By replacing the resistor with the capacitor, you can create a simple RC circuit to see the charging of the capacitor.

4 Run the simulation again by clicking the **Run** button on the Toolbar.

This time the Transient Analysis chart (similar to an oscilloscope) appears in the Analysis Window.

5 Click the Transient Analysis window to select it, and then click with the tip of the **Probe Tool** between the resistor and capacitor.

Notice a diagonal line across the scope (see Figure 3.9). This is actually the beginning of the charge curve for the capacitor. Your view of the curve is limited by start and stop times of the Transient Analysis that were selected by default. You now have the option of changing the Transient Analysis settings to increase the size of the time segment that you can view with the scope, or you can reduce the component values so the capacitor will charge quicker. For this example, you will change the component values.

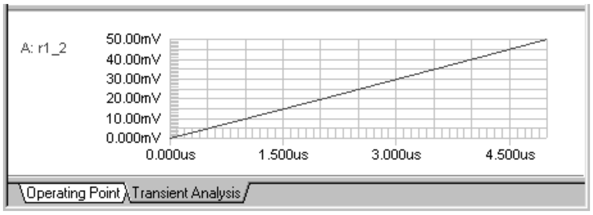


Figure 3.9. The beginning of the charge curve for the capacitor. Your view of the curve is limited by the start and stop times selected for Transient Analysis.

6 Stop the simulation by clicking the **Stop** button.

7 Double-click resistor R1 to display the Edit Device Data dialog box.

8 Change the Label-Value from 1k to 500, and then click OK.

9 Double-click the capacitor C1, change the Label-Value from 1uF to .001uF, and then click OK. Compare your schematic with Figure 3.10.

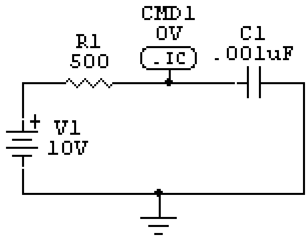


Figure 3.10. Notice that the resistor and capacitor now have different label-values.

10 Run the simulation again.

This time you will see the full charge curve of the capacitor. See Figure 3.11.

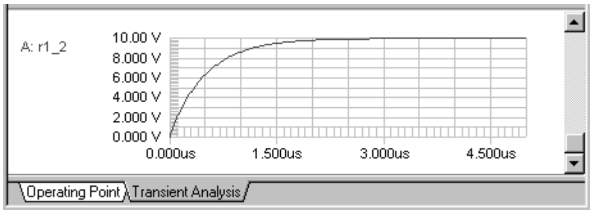


Figure 3.11. Charge curve of capacitor C1.

Simulating a Simple AC Circuit

Now let's create a simple AC circuit using a Signal Generator and two Resistors:

1 Click the **New** button on the Toolbar.

2 Draw the circuit as shown in Figure 3.12, using the following devices:

- 1 Signal Gen [.General/Instruments/Signal Gen] (g)
- 1 Ground [.General/Sources/Ground] (0 (zero))
- 2 Resistors [.General/Resistors/Resistor] (r)

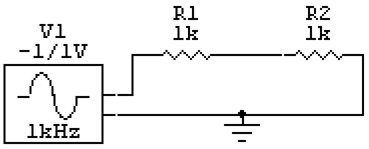


Figure 3.12. A simple AC circuit with a Signal Generator and two Resistors.

3 Use the **Wire Tool** to wire the circuit together.

4 Make sure you are in Analog simulation mode, then run the simulation.

5 Click the wire connected to the output of the Signal Generator with the **Probe Tool**.

The sine wave appears on the scope.

6 Hold down the **Shift** key and click the wire connected between the two resistors.

A second waveform appears on the scope. Compare your results to Figure 3.13.

7 Stop the simulation.

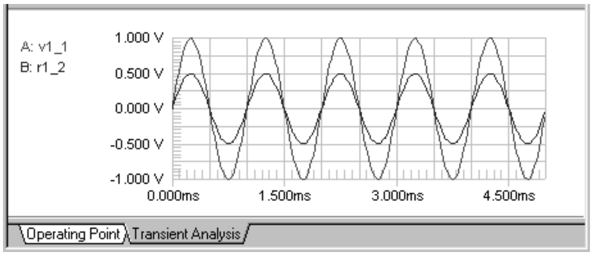


Figure 3.13. Transient Analysis waveforms.

Tutorial 4: More Circuit Simulation

The next example demonstrates how to use all of the analyses and how to take simple measurements using the cursors in the analysis windows. Let's create a basic 10X amplifier circuit using a µA741 Op Amp as shown in Figure 3.14.

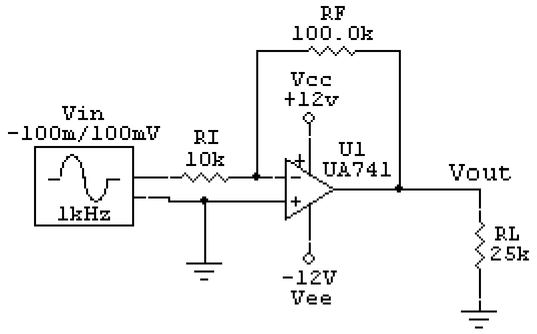
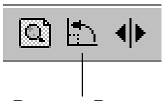


Figure 3.14. A 10X Amplifier Circuit. In this configuration, voltage gain = RF/RI.

1. Select **File** > **New**.
2. Select **Simulation** > **Analog Mode**.
3. Press **F6** if needed to collapse the device browse tree.
4. Place the following devices in the drawing area:
   - 1 Signal Gen [.General/Instruments/Signal Gen] (g) for Vin on the schematic
   - 2 +V devices [.General/Sources/+V] (1) for Vcc and Vee
   - 3 Grounds [.General/Sources/Ground] (0 (zero))
   - 3 Resistors [.General/Resistors/Resistor] (r) for RI, RF and RL
   - 1 Op-Amp5 [Linear ICs/OPAMPs/Op-Amp5] (o) for U1



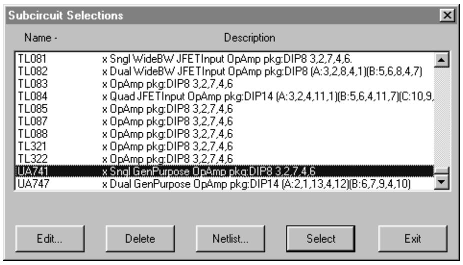
5 Select the **Rotate** button on the Toolbar, which lets you rotate devices in 90° increments. Rotate RL and the -12V supply.

6 Use the **Wire Tool** to wire the circuit together.

7 Use the **Arrow Tool** to drag the devices, wires and labels to make the circuit look clean.

8 Select the **Arrow Tool** and double-click the **Op Amp**.

9 Select **UA741** from the list of available subcircuits (located near the bottom of the list; see Figure 3.15) and click the **Select** button.



Name -	Description
TL081	x Sngl WideBW JFETInput OpAmp pkg:DIP8 3,2,7,4,6.
TL082	x Dual WideBW JFETInput OpAmp pkg:DIP8 (A:3,2,8,4,1)(B:5,6,8,4,7)
TL083	x OpAmp pkg:DIP8 3,2,7,4,6
TL084	x Quad JFETInput OpAmp pkg:DIP14 (A:3,2,4,11,1)(B:5,6,4,11,7)(C:10,9,
TL085	x OpAmp pkg:DIP8 3,2,7,4,6
TL087	x OpAmp pkg:DIP8 3,2,7,4,6
TL088	x OpAmp pkg:DIP8 3,2,7,4,6
TL321	x OpAmp pkg:DIP8 3,2,7,4,6
TL322	x OpAmp pkg:DIP8 3,2,7,4,6
UA741	x Sngl GenPurpose OpAmp pkg:DIP8 3,2,7,4,6
UA747	x Dual GenPurpose OpAmp pkg:DIP14 (A:2,1,13,4,12)(B:6,7,9,4,10)

Figure 3.15. Use the Subcircuit Selections dialog box to select a specific model, such as the UA741.

10 Click the **Exit** button.

11 Double-click the top +V device.

12 Set the **Label-Value** field to **+12V** and visible; set the **Designation** field to **Vcc** and visible; set the **Device** field to NOT visible then click OK.

13 Double-click the bottom +V device.

14 Set the **Label-Value** field to **-12V** and visible; set the **Designation** field to **Vee** and visible; set the **Device** field to NOT visible then click OK.

15 Click and drag the labels so they are positioned as shown on the schematic in Figure 3.14.

16 Double-click each resistor to change both its Label-Value and its Designation and make them visible. Set them up as follows:

Resistor	Label-Value	Designation
Input	10k	RI
Feedback	100k	RF
Load	25k	RL

17 Double-click the **Signal Generator**.

18 Set **Peak Amplitude** to 0.1V and the frequency to 10kHz.

19 Click the **Wave** button in the Signal Generator.

20 Enable the **Source** check box for **AC Analysis**; set **Magnitude** to 1V and **Phase** to 0, and then click OK.

You can now use the Signal Generator as a reference for the AC analysis.

21 Click the **Netlist** button.

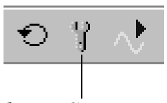
22 Set **Designation** to **Vin**, Visible, and then click OK.

Note: The Label-Value field contains **-1/1V** which represents the minimum and maximum programmed voltage swings before you double-clicked on the Signal Generator. Click OK to return to the schematic.

23 Choose **Edit** > **Place Node Label**. A small rectangle appears which follows your mouse around the schematic. When the bottom left corner of the rectangle touches a wire, a connection rectangle appears. Move the node label so that it connects to the wire which is connected to the top of resistor RL, then click the left mouse button to place it. When prompted, enter **Vout** as the node label text.

Setting Up the Analysis

Now that you have created the circuit, you will set up the analyses. When you run the simulation, the results are based on the conditions you set up.


Analyses Setup Button

1 Click on the **Analyses Setup** button on the Toolbar.

2 Uncheck the **Always set defaults for transient and OP analyses** option so it is cleared.

By unchecking this option, you can access the Transient/Fourier and Operating Point Analysis setups. When checked, defaults are used for the simulation.

3 Click the **Transient/Fourier** button.

4 In the **Default Timing** section, set **Number of Cycles** to 5 and **Points Per Cycle** to 100. Click the **Set Default Timing** button for default Transient Analysis setups and click OK.

This provides simulation for 5 cycles of the input signal with a total of 500 data points. For best reliability, Max Step should be the same size as Step Time. More data points require longer simulation time.

5 Make sure that **Operating Point** is enabled. Operating Point must be enabled in order to use the Operating Point chart.

6 Click the **DC...** button.

7 Select the **Enabled** and **Enable Secondary** options in the DC Analysis Setup dialog box. When you are finished entering the following settings into the appropriate fields, choose OK.

	Primary	**Secondary**
Source Name	Vin	Vcc
Start	-1.5V	10V
Stop	-.7V	14V
Step	0.01V	1V

This setup lets you sweep the voltage of Vin over the specified range at each of 5 different Vcc levels.

8 Click the **AC...** button.

9 Select the **Enabled** option in the AC Analysis Setup dialog box and enter the following settings into the appropriate fields:

Start Frequency	1 Hz
Stop Frequency	1MegHz
Test Points	10
Sweep	Decade

This setup lets you plot the frequency response of the circuit. Click OK to save the settings. Click Exit to return to the circuit.

10 Select **File** > **Save As** and save the circuit as **Myamp.ckt** (analyses setups are saved with the circuit).

Running the Simulation

By placing Run-Time Test Points in your circuit beforehand, you can monitor the simulation results as Spice collects the data (for more information about Run-Time Test Points, see *Chapter 6: Analog/Mixed-Signal Simulation*). The amount of time it takes to finish is based on the analyses you have enabled, the amount of data you're collecting, the complexity of the circuit, and the speed of your computer.

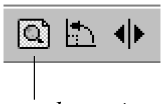
1 Select the **Probe Tool** on the Toolbar.

2 Using the left mouse button, click the Vout wire with the tip of the Probe Tool.

CircuitMaker places a Run-Time Test Point on that node.

3 Click the **Run** button on the Toolbar to start the simulation.

The Run button icon in the Toolbar is temporarily replaced during simulation with a square Stop icon. You can click on this icon to abort the simulation and view the data that has been collected up to that point. When the simulation is complete, the square stop icon is replaced by a round stop icon. When you click on this icon, you will return to editing mode. You can probe the circuit at any time during or after simulation until you return to editing mode.


Zoom schematic to fit

4 If needed, click on the **Zoom schematic to fit** button on the Toolbar to make the entire circuit visible.

5 Click the **Operating Point** tab in the Analysis Window.

6 Click on the wire connected to the output of the Signal Generator with the tip of the **Probe Tool**. Notice that the letter **V** appears on the Probe Tool when it's over a wire.

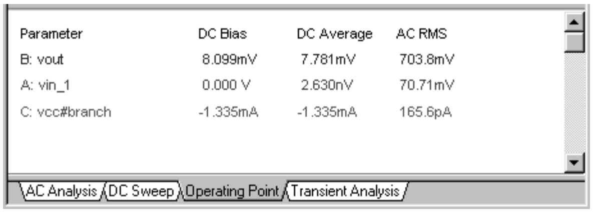
The DC Bias, DC Average and AC RMS voltages at that node with respect to ground will be displayed in the Operating Point chart. The AC RMS value should read approximately 70.7mV (100mV peak). See Figure 3.16.

7 Hold down the **Shift** key and click on the Vout wire. The voltages at that node are added to the Operating Point chart. The AC RMS value should read approximately 707mV (1V peak) which is a gain of 10 over the input voltage.

8 Hold down the **Shift** key and click on the pin of the +12V power supply. The current through that supply is added to the Operating Point chart. Notice that the letter **I** appears on the Probe Tool when it's over a pin.

You can also measure current and power on other devices, but only if you have enabled corresponding Test Points.

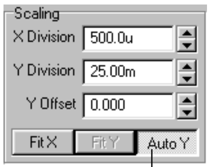
Note: Spice sees the current flowing *into* the positive terminal of a device as positive current.



Parameter	DC Bias	DC Average	AC RMS
B: vout	8.099mV	7.781mV	703.8mV
A: vin_1	0.000 V	2.630nV	70.71mV
C: vcc#branch	-1.335mA	-1.335mA	165.6pA

Figure 3.16. AC RMS values show a voltage gain of 10 from input (A) to output (B).

9 Click the **Transient Analysis** tab in the Analysis Window, then click the wire connected to the output of the Signal Generator with the tip of the **Probe Tool**.



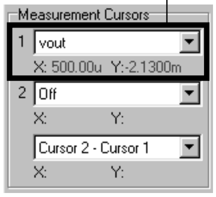
Auto Y Button

A waveform appears in the Transient Analysis chart, similar to what would be seen on an oscilloscope. Make sure the **Auto Y** button on the Panel is pressed IN.

10 Hold down the **Shift** key and click on the Vout wire.

A second waveform appears in the Transient Analysis chart. A quick comparison of the two waveforms confirms that the amplitude at the output of the amplifier is much greater than the amplitude at the input.

Cursor 1



11 In the Panel, select the vout waveform for **measurement cursor 1**. Cursor 1 is displayed on the Transient Analysis chart. Click on the cursor and drag it to *first* rising edge of the vout waveform at the point where the waveform crosses 0V. Notice as you drag the cursor that its X and Y values (displayed in the Panel) change with the position of the cursor on the waveform. To nudge the cursor, click on it once to select it, then press the left or right arrow keys on the keyboard. For a fine nudge, hold down the Shift key while nudging with the arrow keys. See Figure 3.17.

***Tip:** To nudge a cursor, click on it once to select it, then press the left or right arrow keys on the keyboard. For a fine nudge, hold down the Shift key while nudging with the arrow keys.*

12 In the Panel, select the vout waveform for **measurement cursor 2**. Cursor 2 is displayed on the Transient Analysis chart. Click on the cursor and drag it to *second* rising edge of the vout waveform at the point where the waveform crosses 0V. See Figure 3.17.

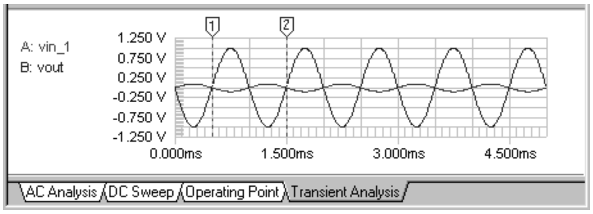
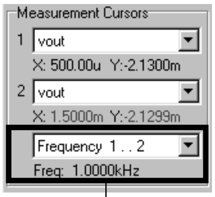


Figure 3.17. Transient Analysis waveforms with cursors 1 and 2 measuring frequency.



Frequency 1..2

13 In the Panel, set the measurement cursors to measure **Frequency 1..2**.

14 Draw a selection rectangle around a portion of the waveforms in the Transient Analysis window that is of interest to you. Do this by clicking the mouse once and holding the mouse while you draw a box. See Figure 3.18.

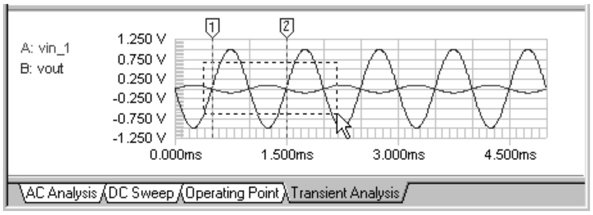


Figure 3.18. Zooming in on a section of the chart.

When you release the mouse button, the view zooms in on the portion of the waveform that you selected. See Figure 3.19.

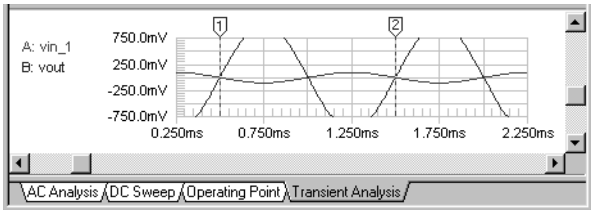


Figure 3.19. Zoomed in on a section of the chart.

15 To restore the original view, choose **Wave** > **Fit Wave-forms**.

OR

In the Panel, click the **Fit X** and **Fit Y** (or **Auto Y**) buttons.

16 Select the **DC Sweep** tab in the Analysis Window, then click on any wire in the circuit.

A DC Sweep Analysis waveform is displayed on the chart, similar to what would be seen on a curve tracer. See Figure 3.20.

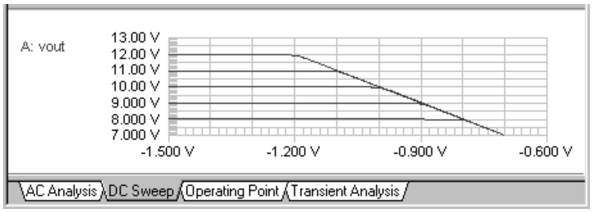


Figure 3.20. DC Sweep Analysis.

17 Select the **AC Analysis** tab in the Analysis Window, then click the Vout wire. An AC Analysis waveform is displayed on the chart. See Figure 3.21.

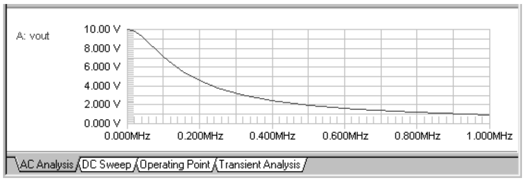


Figure 3.21. AC Analysis showing Magnitude vs. Frequency.

18 Choose **Wave** > **Scaling...**.

19 Set X Scale to **Log**. Set Y Axis Primary to **Magnitude in Decibels**, then click OK.

The waveform now shows the response of the circuit over the specified frequency. This is similar to what would be seen on a bode plotter. Use the cursors to get measurements from the waveforms. See Figure 3.22.

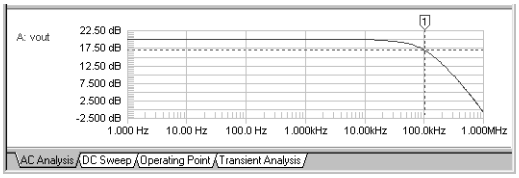


Figure 3.22. AC Analysis showing Magnitude in Decibels and Frequency on a logarithmic scale.

20 Again, choose **Wave** > **Scaling...**.

21 Set Y Axis Secondary to **Phase in Degrees**, then click OK.

The Y Axis is now split into 2 separate grids, the top grid showing magnitude in decibels and the bottom grid showing phase in degrees. The black arrowhead to the right of the grid indicates which grid is selected. Any measurements performed by the measurement cursors are with respect to the selected grid. Left-click on the other grid to select it. See Figure 3.23.

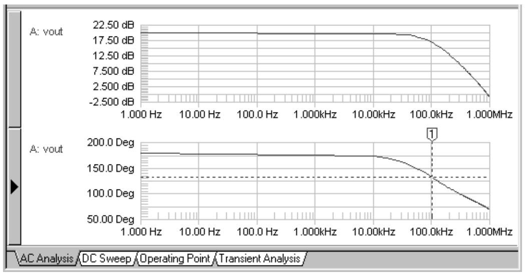


Figure 3.23. AC Analysis showing a dual grid. Magnitude in Decibels is displayed on the top grid, Phase in Degrees is displayed on the bottom. Both grids share the same X Axis.

22 Click the **Stop** button on the Toolbar to return to editing mode.

Mixed-Signal Simulation Example

The following BCD counter circuit demonstrates how digital SimCode devices can be used in analog simulation mode.

1 Choose **File** > **New**.

2 Make sure that Analog simulation mode is selected.

3 Draw the circuit shown in Figure 3.24 using the following devices:

- Data Sequencer [.General/Instruments/Data Seq]
- 74LS168A Counter

  [Digital by Number/741xx/74168/74LS168A]
- +V [.General/Sources/+V] (1)
- Ground [.General/Sources/Ground] (0 (zero))
- Logic Switch [Switches/Digital/Logic Switch] (s)
- Logic Display [Displays/Digital/Logic Display] (9)
- Hex Display [Displays/Digital/Hex Display] (h).

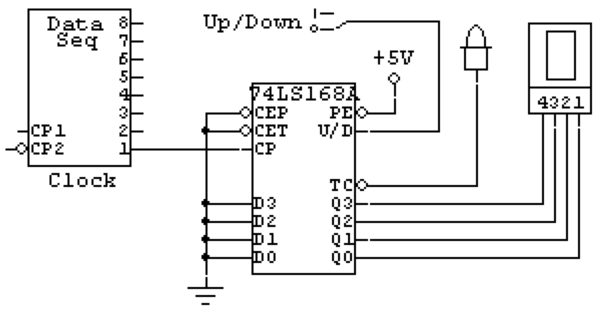


Figure 3.24. BCD Counter Circuit.

4 Double-click the +V and enter **DVCC;** (including the semicolon) in the **Bus Data** field to connect this device to the Vcc pin of the 74LS168A. Click OK.

5 Double-click the **Data Sequencer**.

6 Click the **Pattern** button.

7 Select **Count Up** and click OK.

8 Enter **20** in the **Stop Address** field and enter **10** in the **Tick Increment** field, then click OK.

9 Click on the **Analyses Setup** button on the Toolbar and make sure the **Always Set Defaults** check box is checked, then click the **Run Analyses** button to start the simulation.

10 Click the output of the Data Sequencer with the **Probe Tool** to view the clock signal on the Transient Analysis chart.

11 At the top of the Panel, click the **All Cells** radio button. The Transient Analysis chart changes to All Cells view.

12 While holding down the **Shift** key, click the **Q0** output of the **74LS168A**. This waveform appears on the same chart, but in a separate cell.

13 Repeat this procedure for each output of the 74LS168A. The resulting graph should appear similar to the one shown in Figure 3.25.

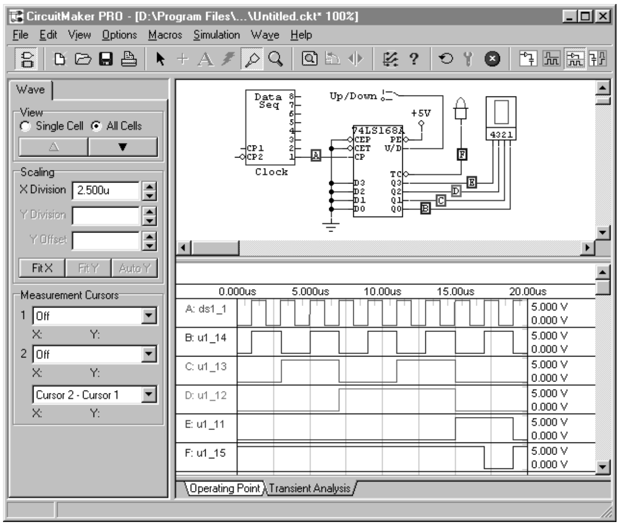


Figure 3.25. BCD Counter Circuit and waveforms.

You can run this same circuit in Digital Logic simulation mode. To try it, stop the simulation, switch to digital and run simulation. In Digital Logic simulation mode, the displays will be animated to show the correct output.

CHAPTER 4

Drawing and Editing Schematics

Using CircuitMaker's robust set of drawing and editing tools, you can create simple to complex schematics quickly and easily. This chapter covers schematic drawing and editing tools and features.

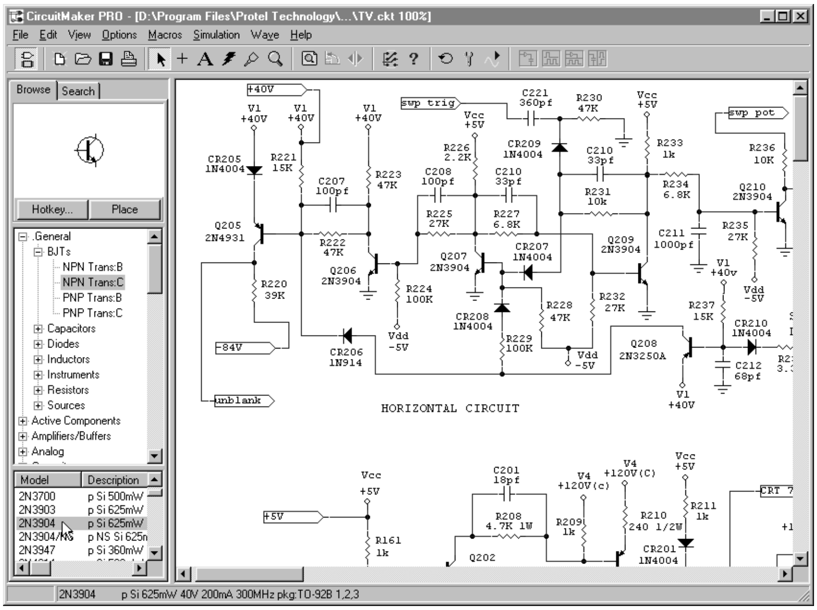
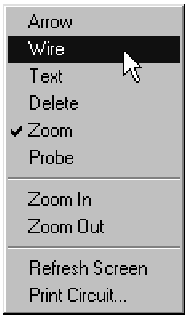


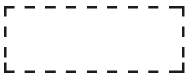
Figure 4.1. CircuitMaker lets you search a library of devices, place them, wire them together, and edit the schematic drawing exactly to your specifications.

The Drawing & Editing Tools

This section describes the buttons on the Toolbar that you will use when placing and wiring components.



Tip: *Right-click with the mouse to access helpful pop-up menus. Contents of the pop-up menus will vary depending on where you click.*



*Tip: Hold down the **Alt** key while using the Wire Tool to draw a dashed line, for example, around a section of circuitry to show that it is a logical block.*

Tip: Using the Text Tool, you can add stylized text anywhere in the schematic drawing.

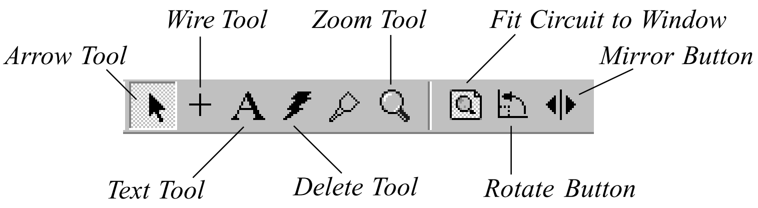


Figure 4.2. The Drawing and Editing Tools.

Arrow Tool

Use the Arrow Tool to select items, move items, flip switches, select tools from the Toolbar, etc. You can double-click items with the Arrow Tool to perform many functions, such as editing specific devices. You can also activate the Arrow Tool by right-clicking the mouse in the schematic background and choosing **Arrow** from the pop-up menu. If you enable the **Arrow/Wire** option, you can use the Arrow Tool to place a wire by clicking on a device pin.

Wire Tool

Use the **Wire Tool** to place wires in the work area. Draw bus wires by holding down the **Shift** key when starting to draw the wire. Refer to the sections *Wiring the Circuit* and *Working with Bus Wires* later in this chapter for more information. Draw a dashed line by holding down the **Alt** key while drawing a wire. Dashed lines act just like regular wires, but if they are not connected to anything, they will not be included in a netlist. You can also activate the Wire Tool by right-clicking the mouse in the schematic background and choosing **Wire** from the pop-up menu.

Text Tool

Use the **Text Tool** to place text in the circuit. Select the tool, click in the work area and type the text. Choose **Options** > **Schematic** and click on the Text Font button to stylize the text or choose the **Colors** tab in the **Options** > **Schematic** dialog box to assign a different color to text. You can alter the way multi-line text wraps by clicking it with the Text Tool and resizing its enclosing rectangle. You can also select the Text Tool by right-clicking in the schematic background and choosing **Text** from the pop-up menu.

Delete Tool

Use the **Delete Tool** to selectively delete items. Select the **Delete Tool**, click on the item you want to delete, and the item is immediately deleted, *except in the case of wires*: If you click and hold on a wire with the **Delete Tool**, the wire is highlighted but not deleted until you release the mouse button. If you hold down the mouse button and move the **Delete Tool** away from the wire, the wire is not deleted. This allows you to see the complete extent of the wire that will be deleted, before you actually delete it. To delete just a segment of wire, right-click on a wire with the **Delete Tool** and choose "Delete Wire Segment", and only the segment between the nearest corners or connections will be deleted.

Cut (or divide) a wire in two by holding down the **Shift** key and clicking the wire with the Delete Tool. The wire is separated at the contact point.

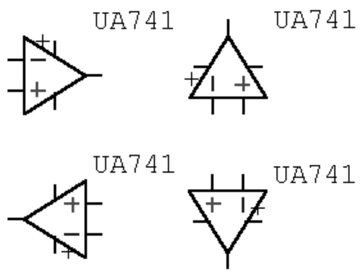
You can also select the **Delete Tool** by right-clicking in the schematic background with the mouse and choosing **Delete** from the pop-up menu. Or select an item and press the **Delete** key on the keyboard to delete that item.

Zoom Tool

Use the **Zoom Tool** to magnify (zoom in) and reduce (zoom out) the view of your circuit. To zoom in, select the Zoom Tool and position it over the area that you want to enlarge. Click the left mouse button to magnify the circuit by the selected Scale Step size. To zoom out, select the Zoom Tool and position it over the area you want to reduce. Hold down the **Shift** key and click the left mouse button to reduce the circuit by the selected Scale Step Size.

You can also activate the Zoom Tool by right-clicking the mouse in the schematic background and choosing **Zoom** from the pop-up menu.

Another zooming method is to press the **Page Up** key on the keyboard to zoom in or the **Page Down** key to zoom out at any time, without using the Zoom Tool. In this case, the zoom is centered on the position of the mouse cursor. See also *Display Scale*, *Normal Size/Position* and *Fit to Zoom* in *Chapter 11: View Menu.*



Use the Rotate Button to rotate a selected device in increments of 90°. Notice that the Label-Value remains readable as the device rotates.

Rotate Button

Use the **Rotate Button** to rotate the selected device in 90° increments.

You can also rotate a device as you select it from the library by pressing the **r** key on the keyboard or by clicking the **Right** mouse button before placing the device in the circuit. You can also select the Rotate Button by choosing **Edit** > **Rotate** or by pressing **Ctrl+R**.

Note: When you rotate a device, pin names and numbers and label values remain readable, that is, they won't be upside down.

Mirror Button

Use the **Mirror Button** to flip the device horizontally. You can also mirror a device as you select it from the library by pressing the **m** key on the keyboard before placing the device in the circuit. You can also select the Mirror Button by choosing **Edit** > **Mirror** or by pressing **Ctrl+M**.

Grid, Title Block and Borders

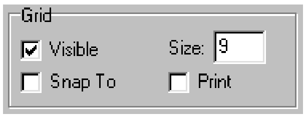
CircuitMaker gives you many advanced schematic features to enhance your schematic, and make precise placement easier.

Grid

Use the Grid option to turn the alignment grid of the drawing window on or off (see Figure 4.3). The grid is useful as an aid in precisely aligning objects. Use Snap To to place new devices (devices not already in a circuit) according to the specified grid. It also lets you move old devices (devices already in the circuit) according to the selected grid, relative to their original position.

Note: When you place a device exactly on the grid, it always remains on the grid regardless of scroll position. However, Snap To does not guarantee alignment of component pins.

Choose **Options** > **Schematic** to access the grid setup options.


Grid
[✓] Visible   Size: 9
[ ] Snap To   [ ] Print

Grid

Border

Title Block

Draggable Page Breaks can be used to visually adjust the print scale.

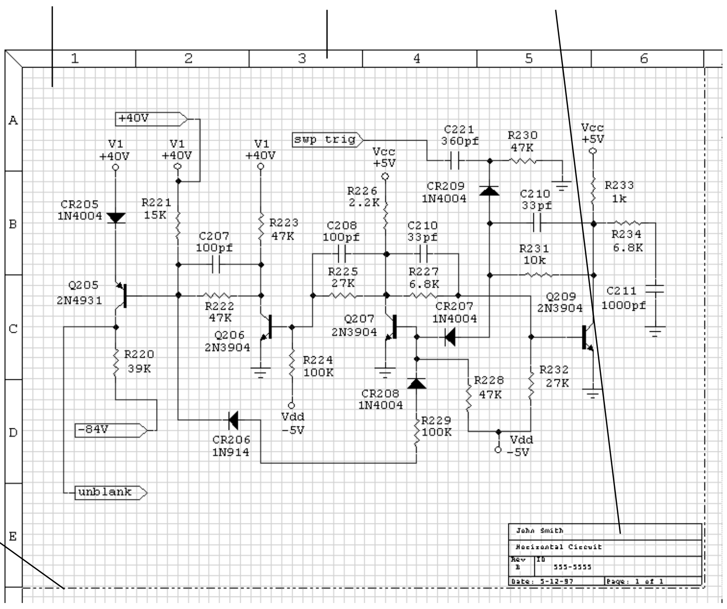


Figure 4.3. Use the Border, Grid, Title Block options to enhance the appearance of your schematic.

Title Block

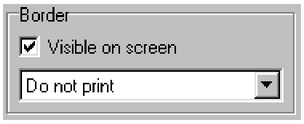
Use the Title Block option (see Figure 4.3 for an example) to add a title box to the lower right corner of the page. The title block contains the following fields: Name, Title, Revision, ID, Date, and Page. The Name and Title fields expand in height to handle multiple rows of text. If you leave the Name or Title fields blank, CircuitMaker excludes them from the title block. The title block also expands in width according to the amount of text that you enter. You can print the title block on the first page, on the last page, or on all pages. Additionally, you can print the full title block on the first page and a reduced title block (that is, one that does not include the Name and Title fields) on subsequent pages.

Choose **Options** > **Schematic** to access the Title Block setup options.

Borders

Use this option to quickly locate devices by displaying a coordinate grid system around your schematic (see Figure 4.3 for an example). For example, suppose you want to find a device that you know is located in the B-5 grid square. By drawing an imaginary line from the letter B and the number 5 on the margins of the schematic; the intersection of these lines locates the grid square containing the device. The actual size of the border is determined by the size of the paper. See *Printing and Exporting Schematics* at the end of this chapter.

To add a border to your schematic drawing,


Border
☑ Visible on screen
Do not print

1 Choose **Options** > **Schematic** to display the border setup options.

2 Enable the **Visible on screen** checkbox to display a border that outlines the total allowed schematic area.

3 Select **Do not print** if you don't want to print the border.

OR

Select **Print around entire schematic** to print the border so that it is only on the outside edges of the outside pages, making a border around the entire schematic when the pages are arranged together.

OR

Select **Print around each page** to print the complete border on each page of your schematic.

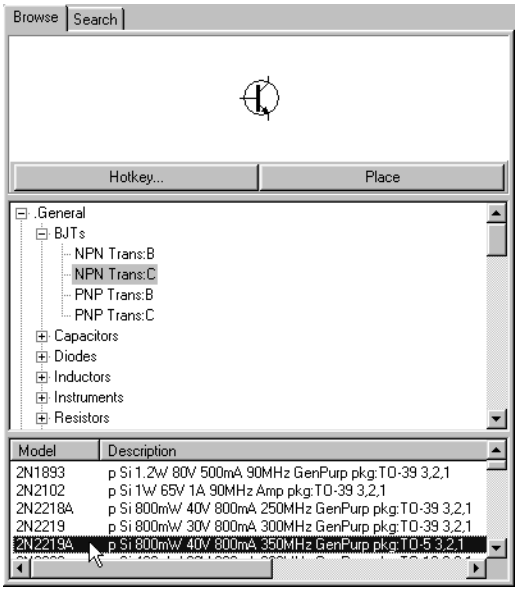
Finding and Placing Devices

CircuitMaker comes with a library of several thousand devices (see the *Device Library Guide* for a complete list of the devices and instruments). You can place devices from the library using the Browse or Search tabs in the Panel (see Figures 4.4 and 4.5) or by using Hotkey shortcuts.

The Browse Tab

You can browse through the devices in CircuitMaker using the Browse tree in the Panel (Figure 4.4). The parts are listed by Major Device Class, Minor Device Class, Device Symbol and Spice Model.

__Tip:__ You can quickly restore the device browse tree to its original "closed" condition by selecting __View__ > __Collapse Device Tree (F6)__.



Model	Description
2N1893	p Si 1.2W 80V 500mA 90MHz GenPurp pkg:TO-39 3,2,1
2N2102	p Si 1W 65V 1A 90MHz Amp pkg:TO-39 3,2,1
2N2218A	p Si 800mW 40V 800mA 250MHz GenPurp pkg:TO-39 3,2,1
2N2219	p Si 800mW 30V 800mA 300MHz GenPurp pkg:TO-39 3,2,1
2N2219A	p Si 800mW 40V 800mA 350MHz GenPurp pkg:TO-5 3,2,1

Figure 4.4. The Browse tree in the Panel lets you browse and select from a library of several thousand devices. The Panel can be widened by dragging the vertical splitter bar at the right edge of the Panel.

To find and place a device,

1 Click the **Browse** tab at the top of the Panel.

2 Locate the device you want to place by first selecting a **Major Device Class**, a **Minor Device Class**, a **Device Symbol**, and if applicable, a specific **Spice Model**.

In this manual, device location is indicated as [major class/minor class/device symbol]. For example, a 2N3904 could be found at [.General/BJTs/NPN Trans].

Notice that the schematic symbol for the part you select is shown at the top of the Panel.

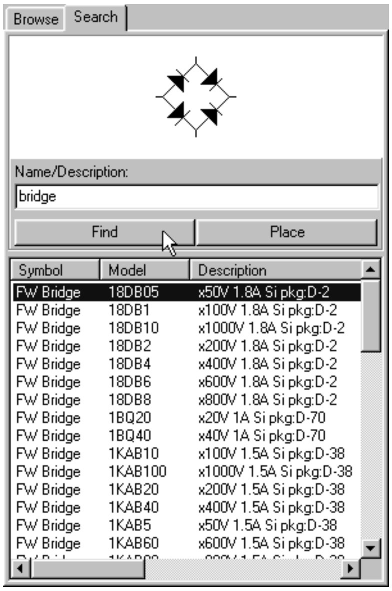
3 Double-click the device name or model or click **Place** to select the device for placement in the workspace.

Notice that the device follows the mouse around the screen until you click the **Left** mouse button. While dragging a device in this manner you can rotate (**Right** mouse button) or mirror (**M** key on the keyboard) the device.

The Search Tab

You can search for a specific device in CircuitMaker using the Search tab in the Panel (Figure 4.5), which lets you find all devices that match the part number or description that you enter. A match is found any time the search text you enter is contained in a device's major class name, minor class name, symbol name, or model description. However, only the symbol name and model description appear in the match list. If a search produces items that don't seem to match the search text you entered, it might be because the match occurred with the major or minor class names, which are not displayed.

Note: *The Device Search feature is not case sensitive.*



Name/Description: bridge

Symbol	Model	Description
FW Bridge	18DB05	x50V 1.8A Si pkg:D-2
FW Bridge	18DB1	x100V 1.8A Si pkg:D-2
FW Bridge	18DB10	x1000V 1.8A Si pkg:D-2
FW Bridge	18DB2	x200V 1.8A Si pkg:D-2
FW Bridge	18DB4	x400V 1.8A Si pkg:D-2
FW Bridge	18DB6	x600V 1.8A Si pkg:D-2
FW Bridge	18DB8	x800V 1.8A Si pkg:D-2
FW Bridge	1BQ20	x20V 1A Si pkg:D-70
FW Bridge	1BQ40	x40V 1A Si pkg:D-70
FW Bridge	1KAB10	x100V 1.5A Si pkg:D-38
FW Bridge	1KAB100	x1000V 1.5A Si pkg:D-38
FW Bridge	1KAB20	x200V 1.5A Si pkg:D-38
FW Bridge	1KAB40	x400V 1.5A Si pkg:D-38
FW Bridge	1KAB5	x50V 1.5A Si pkg:D-38
FW Bridge	1KAB60	x600V 1.5A Si pkg:D-38

Figure 4.5. This example shows the found set after searching for the word "bridge".

The Search feature accepts partial words to match devices. For example, a search text of **op** would find **Op-Amp** or **loop**. You can also use an asterisk as a "wild card." For example, entering **74*8** matches **748**, **7408**, **74LS08**, **74138**, **74285**, and so on. If you enter multiple words as the search text, a match for each word is found, though not in the order you enter them. For example, typing **741 op-amp** and **op-amp 741** would both give the same results.

To search for a device,

1 Click the **Search** tab at the top of the Panel.

2 Type a device name, number, or description in the text box, then click **Find**.

3 Use the scroll bar (if necessary) to scroll through the list. When you find the device, click it to highlight it.

4 Double-click the device or click **Place** to select the device for placement in the workspace.

Note: If you modify the User Library (USER.LIB) file, expect a slight delay the next time you search for devices in the Device Search dialog box. The delay is caused as Circuit-Maker builds a new SEARCHDB.DAT search list file. If for any reason the search list seems incorrect or out of date, you can create a new search list file by choosing **Macros** > **Update Search List**.

Hotkeys

You can also select devices by pressing a predefined Hotkey on the keyboard. You can view the Hotkey list and reassign Hotkeys to different devices as needed.

Assigning Hotkeys

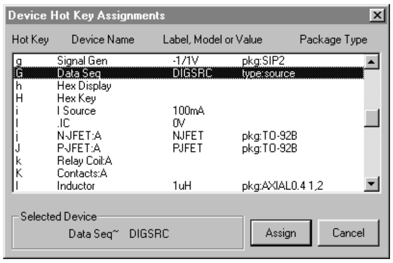
You can assign sixty-two of your most commonly used devices to Hotkeys. This lets you quickly select these devices by pressing the Hotkey on the keyboard.

To assign a Hotkey to a specific device,

1 Select the Browse tab in the Panel and find the device for which you want to assign a Hotkey.

2 Click the **Hotkey** button to display the dialog box in Figure 4.6. Hotkeys are listed alphabetically along with the devices that are currently assigned to them.

3 Scroll through the list to find the Hotkey that you want to assign to your selected device, then click the **Assign** button.



Hot Key	Device Name	Label, Model or Value	Package Type
g	Signal Gen	-1/1V	pkg:SIP2
G	Data Seq	DIGSRC	type:source
h	Hex Display		
H	Hex Key		
i	I Source	100mA	
I	.IC	0V	
j	N-JFET:A	NJFET	pkg:TO-92B
J	P-JFET:A	PJFET	pkg:TO-92B
k	Relay Coil:A		
K	Contacts:A		
l	Inductor	1uH	pkg:AXIAL0.4 1,2

Figure 4.6. Use this dialog box to assign or reassign Hotkeys to devices.

Unassigning a Hotkey

To remove a Hotkey assignment from the list,

1 Follow Steps 1 and 2 from the previous section.

2 Assign a new device in its place.

OR

Assign **none** (at the top of the list) as the Hotkey.

Placing Devices

After you have searched and found a device, you can place the device or reselect it using a number of different options illustrated in this section.

To place a device,

***Note:** If the Quick Connect feature is enabled and an unconnected device pin is placed so that it touches a wire or other device pin, CircuitMaker will automatically make the connection. See the Wiring section in this chapter for details.*

1 Select it using one of the methods discussed in the previous section.

2 Press the **r** key or **Right**-click the mouse to rotate the device into the position you want.

3 Press the **m** key to mirror the device.

4 **Left**-click the mouse to place the device in the workspace.

OR

Press any key (except r or m) to cancel placement.

Note: To repeatedly place identical devices, enable the **Auto Repeat** checkbox under **Options** > **Schematic**.

Selecting Devices

Use the **Arrow Tool** to select and move devices around the workspace. There are four different ways of selecting items in the circuit window:

Selecting a Single Item

To select a single item, click it with the **Arrow Tool**. Click anywhere else in the work area to deselect the item.

Selecting Multiple Items

To select multiple items, hold down the **Shift** key and click on one or more items with the Arrow Tool. To deselect any single item click on it a second time while still holding down the **Shift** key. To deselect all items, release the **Shift** key and click somewhere in the work area away from all items.

You can also select multiple items by holding down the left mouse button and dragging a selection rectangle around the desired items. This method works especially well for selecting switches, because clicking on a switch will not always select it (unless you click on the outer edge of the switch). To deselect any single item from a group of selected items, **Shift-click** on it. To deselect all items, click in the workspace away from all items.

Selecting All Items

To quickly select all items, choose **Edit** > **Select All** or press **Ctrl+A**.

Nudging Devices

To slightly "nudge" a device in any direction, select a single device and then press the **Left Arrow**, **Right Arrow**, **Up Arrow**, or **Down Arrow** key (as illustrated in Figure 4.7). A single nudge shifts the device one pixel, whether or not **Snap-to-Grid** is enabled.

Note: The nudge feature does not work if you have selected wires or multiple devices. It only works with a single device.

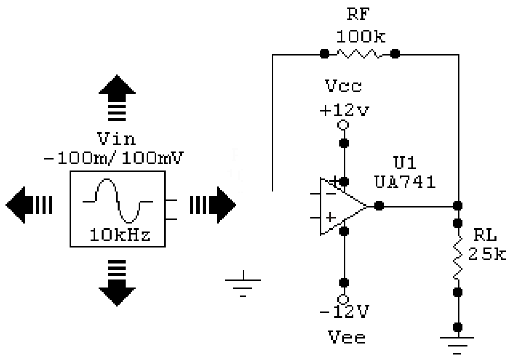


Figure 4.7. Use the Left, Right, Up, or Down Arrow key to "nudge" a selected device more precisely.

Wiring the Circuit

***Tip:** You can also draw wires with the Arrow Tool if you have enabled the Arrow/Wire option on the Options menu. See Arrow/Wire in Chapter 12: Options Menu.*

To simulate and generate PCB netlists properly, the components in your circuit must be correctly wired together. CircuitMaker's Auto Routing, Manual Routing and Quick Connect methods are fully integrated and automatic, so you don't have to choose or switch between wiring modes. A "valid connection point" for wiring is any device pin or wire. Wires can only be horizontal or vertical.

Wiring Method	**How to Use**
Auto Wire Routing	Click *and drag* with the Wire Tool from any valid connection point to another connection point, then release.

Manual Wire Routing	Click *and release* with Wire Tool to start wire, single-click to change directions, then single-click on a connection point or double-click anywhere to end wire.
Quick Connect	When enabled, place or move a device with the Arrow Tool so that unconnected pins touch a wire or other device pins.

The SmartWires™ Advantage

Regardless of the method of wiring you use, CircuitMaker’s SmartWires™ feature lets you connect a wire to a device pin or another wire without being in exactly the right place. This ensures perfect connections every time, and eliminates any guesswork. A user-definable connection area exists around each valid connection point. When you place the Wire Tool in a connection area, a rectangle appears highlighting the connection point. You can set the size of the connection area and whether or not to display the rectangle under **Options** > **Schematic**. For more information, see *Connection Area* in *Chapter 12: Options Menu*.

Auto Wire Routing

To quickly and easily Auto Route wires,

Wire Tool

1 Select the **Wire Tool** from the Toolbar.

2 Move the tool over a valid connection point.

Note: A valid connection point is any device pin or wire.

Auto Wire Routing

3 Click *and hold* the left mouse button.

4 Drag the mouse to another valid connection point and release the mouse button.

5 The wire automatically routes between the two points.

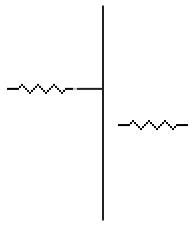
Auto routing requires two valid connection points. You cannot draw a wire with auto routing that does not connect to something on both ends. Also, you cannot draw bus wires with the auto routing method.

Manual Wire Routing

Manual routing lets you freely place wires exactly where you want them. It also lets you place “free wires” in your schematic that are not connected to anything.

Note: To draw bus wires, use the manual routing method.

To route wires manually,



Manual Wire Routing

1 Select the **Wire Tool** from the Toolbar.

2 Move the tool to the position where you want to start the wire.

3 Click *and release* the left mouse button.

The Wire Tool cursor disappears and is replaced with an extended wiring cursor. The extended cursor simplifies the task of precisely aligning wires with other objects.

4 Click once with the left mouse button to turn 90° or double-click to end the wire.

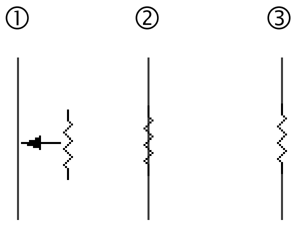
5 Single-click the mouse to terminate the wire when it is at a valid connection point (if you have enabled the **Single Click Connect** option in the Schematic Options dialog box).

6 To cancel a wire at any time while you are drawing it, press any key or right-click with the mouse.

Quick Connect Wiring

One of the easiest methods for wiring is to use the Quick Connect wiring method. This feature allows you to simply touch unconnected device pins to wires or other unconnected device pins, and the connections are made automatically.

To wire using Quick Connect,



Quick Connect Wiring

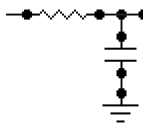
1 Choose a new device from the library

OR

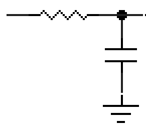
Nudge or click and drag an existing device with the Arrow Tool.

2 Move the device so that the end of the unconnected pin(s) touches a wire or other device pin(s).

Show Pin Dots:



Enabled



Disabled

Tip: Pin Dots can help to determine proper wiring. They can be enabled in Options > Schematic.

3 Once placed, CircuitMaker will connect the device to the wire or pin it is touching.

Note: Placing a device so that two pins are parallel over a wire will automatically connect both ends and "insert" that device into the wire segment (as in the illustration at left).

The same connection area size used by SmartWires (described previously) is used by the Quick Connect feature. This connection area determines how close you must be to a wire or pin before CircuitMaker will automatically make the connection, and can be altered in the Schematic Options dialog box.

By default, the Quick Connect option is active. To turn off Quick Connect, go to **Options** > **Schematic** to uncheck and deactivate the Quick Connect option.

Extending, Joining, and Cutting Wires

Wires are fully editable and can be extended, joined, and cut.

To extend a wire,

1 Select the **Wire Tool** from the Toolbar.

2 Place it over the end of the wire and start a new wire to extend the existing one.

To join two wires together,

1 Draw a wire from the end of the first wire to the end of the second wire.

Notice that they become the same wire.

To cut a single wire into two separate wires,

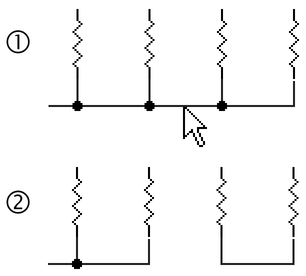
Tip: You can also cut a wire by right-clicking on it with any tool and selecting Snip Wire from the pop-up menu.

1 Select the **Delete Tool** from the Toolbar.

2 Place it over the point(s) where you want to cut the wire.

3 Hold down the Shift key.

4 Click the left mouse button to cut the wire. The wire is divided into two wires.



Deleting Wire Segments

Individual wire segments can be deleted without losing connections to adjoining wires.

To delete a wire segment,

1. Right-click on the wire segment with any tool.
2. Select Delete Wire Segment from the pop-up menu.

Highlighting an Entire Node

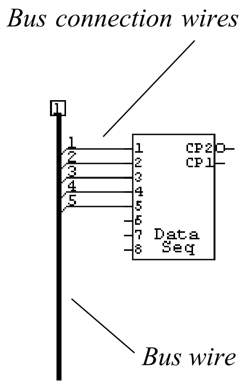
You can highlight an entire circuit node with a single mouse click. This can be very helpful when looking for wiring mistakes. Even wires which are not physically connected on the schematic (the ground node, for example) can be highlighted together if they are the same circuit node.

To highlight an entire node,

1. Select the **Arrow Tool** from the Toolbar.
2. **Alt+click** on one of the wires on the node.

Moving Devices with Connected Wires

You can move a device that has wires connected to it without disturbing the connection. When you select and drag a device with the Arrow Tool, the wires undergo a "rubberband" effect, meaning that they stretch (extend) but remain connected to the device.



Bus Wires

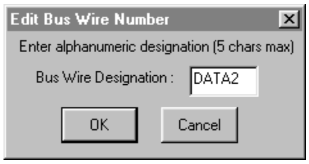
Bus wires are a special type of wire that contain multiple individual wires. Each bus wire is identified by a name and each individual wire within a bus also has a name. Bus wires can be easily identified in a circuit because they are drawn thicker than regular wires.

To draw a bus wire,

1. Hold down the **Shift** key.
2. Draw a regular wire using the manual routing method (see *Manual Wire Routing* earlier in this chapter).

Note: You must hold down the Shift key before you start a wire but you can release the key before the wire is finished.

After you draw the bus wire, a dialog box (shown in Figure 4.8) prompts you for a bus name.


Edit Bus Wire Number

Enter alphanumeric designation (5 chars max)

Bus Wire Designation : DATA2

Figure 4.8. Each bus must have a different bus name.

3 Enter a unique bus name.

Note: If two separate buses are given the same name, they are considered to be the same bus, even if they are not physically connected on the screen.

To rename a bus wire, double-click on it with the **Arrow Tool** or Right-click on it and select **Edit Bus Wire Number**.

You can extend, join, or cut bus wires just like regular wires. However, you don't need to hold down the **Shift** key when extending bus wires.

Bus Connection Wires

Regular wires that are connected to a bus wire are called "bus connection wires." These wires connect to the individual wires within the bus.

To create bus connection wires,

1 Select the **Wire Tool** from the Toolbar.

2 Move the tool over a valid connection point.

Note: A valid connection point is any device pin or wire (including a bus wire). At least one end of the bus connection wire must be connected to a bus.

3 Click and hold the left mouse button.

4 Drag the mouse to another valid connection point and release the mouse button.

When you make the connection, a dialog box appears asking you to assign a bus connection wire name.

5 Specify a bus connection wire name.

6 Choose **Angle Connection to Top/Left** if you want to change the angle of the connection.

To edit a bus wire or bus connection wire name, double-click it with the **Arrow Tool** or Right-click on it and select **Edit Bus Connection** from the pop-up menu.

CircuitMaker displays bus and bus connection labels by default, but you can disable them in the **Schematic Options** dialog box.

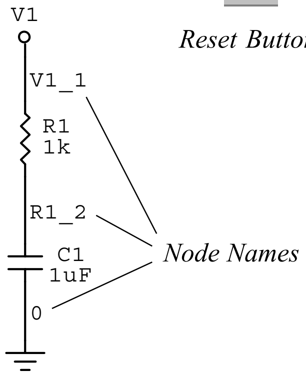
Wiring Bus Connection Wires Together

If you assign two bus connection wires the same name and you connect them to the same bus wire (or to different bus wires with the same bus wire name), they function as though they were connected together.

Node Names and Connectivity

CircuitMaker uses names for its circuit nodes. Node names are used to identify waveforms for simulation and to identify nets for PCB layout. Nodes which are not physically connected on the schematic, but which have the same node name are considered to be connected.

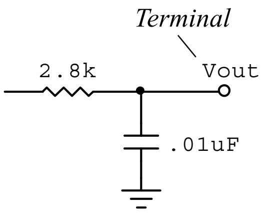
Reset Button



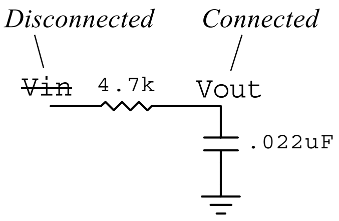
In general, the names used are derived automatically from the components to which they are connected. For example, a node named **U2_6** would get its name because it is connected to pin 6 of device U2. Of course, this node is also connected to other pins of other devices, but it derives its name from the oldest device connected to the node, that is, the device that was placed on the schematic first. The node names are updated each time you click the Reset button on the Toolbar or run a simulation.

There are a few exceptions. Some devices take priority over other devices when it comes to naming the nodes. For example, a voltage source takes node naming priority over other components. That way, if you have a +V device whose designation is Vcc, the net will be named **Vcc_1** rather than some seemingly complex name such as U7B_8. The ground node will always be either **0** or **GND**.

Other means have been provided to give a node a specific name. This has the added benefit of allowing otherwise independent nodes to be connected together.



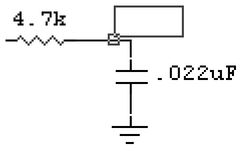
One way to do this is to use one of the special connector devices. **Input** [Connectors/Active/Input], **Output** [Connectors/Active/Output] and several **Terminal** [Connectors/Active/Terminal] (t) devices all have the ability to name a node. Each time you place one of these devices, you will be asked to provide a name which will then be used as the name of the node. These devices take node naming priority over other devices, including voltage sources.



Node Labels

Finally, a **Node Label** can be placed on a wire to create a node name. A Node Label appears as simple text on the schematic, but must be placed immediately above or to the right of a wire. A small connection rectangle is displayed at the point of contact when being placed on a wire. If, because of its position, a node label is not connected to a wire, there will be strikeout characters through the text indicating that it is not connected. The Node Label takes node naming priority over all devices except Ground.

To place a Node Label,



Placing a Node Label

1 Choose **Edit** > **Place Node Label**.

2 Drag the Node Label rectangle to where its bottom-left corner touches a wire on the desired net. Click the left mouse button.

3 Type in the desired name for the node. Click OK.

To view all the node names on the schematic,

1 Choose **Options** > **Schematic** (or press **F5**.)

2 Enable the **Show Node Names** checkbox. Click OK. Node names are displayed at the center of the longest wire on the node.

Tip*: You can verify node connectivity by holding down the Alt key and clicking on a wire with the Arrow Tool. The entire node will be highlighted.*

Power Bus Connections

We saw in the previous section that circuit nodes could be given specific names and that if two or more nodes were given the same name, they were considered to be the same node. The method by which a particular node gets its name is irrelevant. Now let's examine how this applies to power bus connections such as Vcc or Ground.

Connecting a Power Bus to a Single Source

Suppose you have several op-amps that you want to connect to Vcc. You could simply place a separate +V device on on each op-amp and it will simulate just fine. However, each +V will be an independent source on a separate node, each having a unique designation. This would be unacceptable when creating a netlist for PCB layout.

If you want all the Vcc points to be common to a single node, there can only be one source (the Spice simulator will generate an error message if you connect multiple +Vs on the same node.) In this case you can use one +V and multiple Terminals.

1 Wire the **+V** to one op-amp, and wire a **Terminal** to the each of the other op-amps. Enter **Vcc** as the Terminal Name for each of the Terminals.

2 Place a **Node Label** on the wire connected to the +V and name it **Vcc**. The +V will be connected to the Terminals by a common node name.

OR

Enter **Vcc;** (including the semicolon) into the Bus Data field of the +V. The +V will be connected to the Terminals by common Bus Data (see Figure 4.9.)

This use of Terminals instead of multiple +Vs creates a single node or net in the netlist (Spice or PCB.)

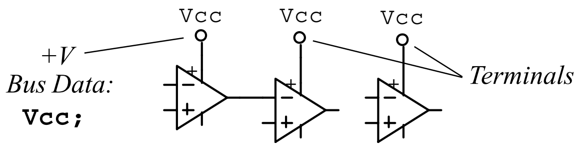


Figure 4.9. Using Terminals to connect a power bus to a single +V source via the Bus Data field.

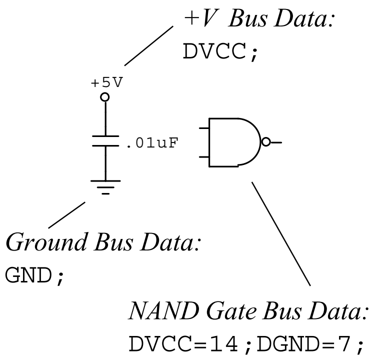


Figure 4.10. In this example, ***DVCC;*** *is entered directly into the Bus Data field of a +V device. This +V then becomes the source for the DVCC bus, overriding the 5V default specified in the Analog Options dialog box.*

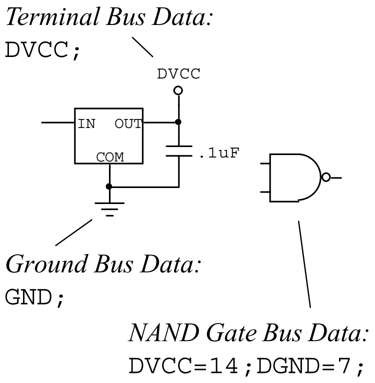


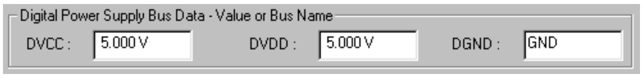
Figure 4.11. In this example, ***DVCC*** *is entered as the Terminal Name. The name is automatically copied into its Bus Data field and a semicolon is appended. This Terminal then becomes the link to the source for the DVCC bus, overriding the 5V default specified in the Analog Options dialog box.*

Digital Power Bus Connections

CircuitMaker's digital devices do not include Vcc and ground pins. However, Vcc and ground connections are required for both Analog simulation and PCB layout. These "hidden" pins are assigned to power buses using the Bus Data field for each digital device (see *Bus Data* later in this chapter.) The power buses assigned to the digital devices are DVCC, DVDD and DGND. These power bus names must not be confused with node names.

There are three devices that can be used to connect these buses to other nodes in the circuit: **+V**, **Terminal** and **Ground**. To connect one of these devices to a bus, simply enter the bus name, followed by a semicolon, into the device's Bus Data field. For example, place a +V device in your circuit, then enter **DVCC;** into its Bus Data field. The +V will now be the source for the DVCC bus (see Figure 4.10.) The Terminal is used in the same manner, but rather than being the actual source, it would be a *link* to a source to which it is connected. The Bus Data for the Terminal is filled automatically when you enter the Terminal Name (see Figure 4.11.)

You may notice that the Ground device already includes **GND;** in its Bus Data field, but that the name of the digital ground bus is DGND. These two buses are linked using the **Power Supply Bus Data** fields of the Analog Options dialog box (see Figure 4.12.) These bus linking fields are designed to allow you to enter either a value or a bus name (as in the case of GND.) If no source for the power bus is included in your schematic, the digital devices will still work using the default values entered here.



Digital Power Supply Bus Data - Value or Bus Name

DVCC : 5.000 V   DVDD : 5.000 V   DGND : GND

Figure 4.12. If no source is connected to a digital bus via the Bus Data fields, these values will be used to determine the source voltage for simulation.

Additional buses can be defined using the Bus Data field, but no default information will be available as it is for DVCC, DVDD and DGND (see *Bus Data* later in this chapter.)

Labeling the Schematic

CircuitMaker offers several methods of labeling the schematic, including the Text Tool and editing the device itself.

Using the Text Tool to Label

To label the schematic using the Text Tool,

1 Select the **Text Tool**.

2 Click in the drawing window where you want to place the text.

3 Resize the text box as necessary (see picture at left).

4 Type the text.

Text can be multi-lined and fully stylized. Always visible, text may be repositioned at any time.

Changing Device Labels

Don’t confuse a device designation with a device label-value. Figure 4.13 illustrates the difference.

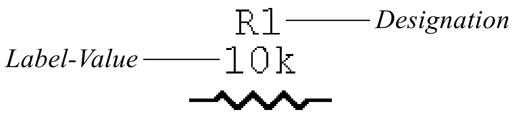


Figure 4.13. A Label-Value is information about a device whereas a Designation identifies the device in the circuit.

To label devices using the Device Properties dialog box,

1 Double-click the device.

2 If a dialog box other than the Device Properties dialog box appears, click the **Properties** button.

3 Type the appropriate text in the Label-Value, Designation, or Description text box.

4 Make the text visible on the schematic by selecting its Visible check box.

You can reposition label-values, designations and descriptions anywhere around the device by dragging them with the Arrow Tool. Even if you reposition a label around a device, it remains attached to the device when the device is moved around the workspace. For a detailed description of these labels, refer to the following section, *Editing Devices*.

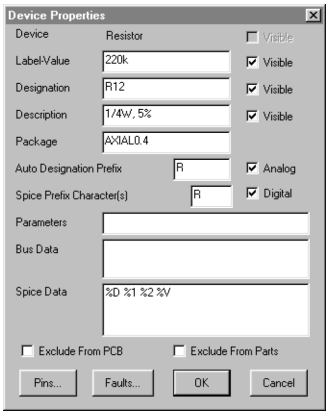
Device Properties

You can easily edit a wide range of device information related to schematic, simulation, pcb netlists and other purposes.

To edit a device,

1 Double-click on a device or Right-click on it and select **Device Properties** to display the Device Properties dialog box shown in Figure 4.14.

Note: A different dialog box will appear when you double-click on certain devices, so click the **Properties** button to get to the Device Properties dialog box.


Device Properties

Device: Resistor ☐ Visible

Label-Value: 220k ☑ Visible

Designation: R12 ☑ Visible

Description: 1/4W, 5% ☑ Visible

Package: AXIAL0.4

Auto Designation Prefix: R ☑ Analog

Spice Prefix Character(s): R ☑ Digital

Parameters:

Bus Data:

Spice Data: %D %1 %2 %V

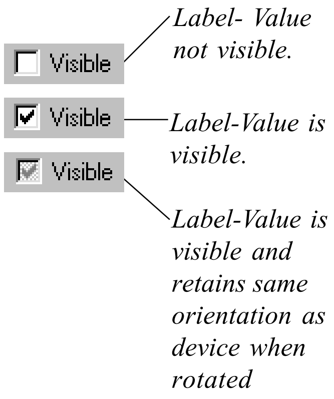
☐ Exclude From PCB ☐ Exclude From Parts

Figure 4.14. Use the Device Properties dialog box to enter or change a variety of device information.

Device

This non-editable field shows the device name as it appears in the library menus. Use the Visible check box to show or hide the name in the schematic. If visible, the device name retains the same orientation as the device when you rotate it. The label-value, designation and description, however, will remain right-side up no matter how the device is rotated. Note that some device names, such as "Resistor", cannot be made visible.

The Label-Value field has a tri-state Visible check box:


Visible
Label- Value not visible.
Visible
Label-Value is visible.
Visible
Label-Value is visible and retains same orientation as device when rotated

Label-Value

Use this field to enter information about the device such as its label (1N914, 2N3904, etc.) or its value (47K, 100U, etc.), or to replace the existing Device name (that is, make the Device not visible and the Label visible). Multi-pole switches use this field to specify which single-pole switches are linked together (see SPDT Switch in the *Device Library Guide*). The Label-Value can be dragged around the device on the schematic with the mouse and will remain attached to the device when the device is moved.

If you set the Visible check box to gray (see diagram at left), the Label replaces the Device name and retains the same orientation as the device when rotated. Otherwise, the label-value will remain right-side up no matter how the device is rotated.

Designation

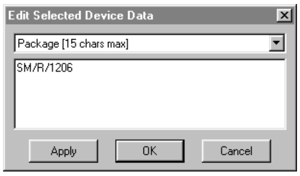
Use this field to identify the device in the circuit such as U3, CR7, RLOAD, etc. You can also make this field visible with the Visible check box. This field must contain the device designation in order for simulation and pcb netlists to work properly. The Designation label may also be dragged around on the schematic with the mouse. It will remain right-side up no matter how the device is rotated.

This field is filled in automatically when you place the device. See *Auto Designation Prefix* later in this section for more information. See also *Set Designations* in *Chapter 10: Edit Menu* and *Device Designations* in *Chapter 12: Options Menu* for other features that affect device designations.

Important: Individual parts of multipart packages must be grouped properly. CircuitMaker takes care of this automatically if you place devices directly from the library. However, if you have altered multipart packages, they must be grouped using the **Edit** > **Group Items** command. Just changing the Designation field is not sufficient. Each individual part must be identified as PART A, PART B, etc., from the Edit Device Pin Data dialog box. See *Pins* later in this chapter.

Description

Use this field for schematic reference only. You can use it to display additional information such as custom part numbers, tolerances, etc. With the exception of relays, this field does not affect simulation (relays use this field to specify coil/contact combinations--see Relays in the *Device Library Guide*). Use the Visible check box to make the field visible or not. You can also drag the Description label around on the schematic with the mouse. The Description label will remain right-side up no matter how the device is rotated.



Package [15 chars max]

SM/R/1206

Tip: *You can quickly change one of the device fields for all selected devices. Just Right-click on one of the selected devices and choose Selected Device Properties. Choose the field you want to edit, then type in the appropriate text.*

Package

Use this field to identify the type of physical package (footprint) the device is in (DIP14, TO-92B, etc.). When you are creating pcb netlists for TraxMaker or other pcb layout programs, make sure the Package name you enter *exactly* matches the name of the corresponding component footprint in your pcb layout program's library.

Auto Designation Prefix

This is the prefix used when CircuitMaker automatically assigns a device's designation (whenever you place a device or select **Edit** > **Set Designations**). The prefix may be up to 4 characters in length.

Spice Prefix Character(s)

This is the Spice prefix used in conjunction with the %D and %M flags described later under *Spice Data*. You would normally use this field when linking the macro symbols you define to the proper model selections when creating new devices (see *Chapter 17: Creating New Devices* for more information). Valid prefixes are:

Prefix	**Meaning**
A	XSpice Model
BV	Nonlinear Dependent Voltage Sources
BI	Nonlinear Dependent Current Sources
C	Capacitors
D	Junction Diodes
DZ	Zener Diodes
E	Linear Voltage-Controlled Voltage Sources
F	Linear Current-Controlled Current Sources
G	Linear Voltage-Controlled Current Sources
H	Linear Current-Controlled Voltage Sources
I	Independent Current Sources
JN	JFETs (N-channel)
JP	JFETs (P-channel)
K	Coupled (Mutual) Inductors
L	Inductors
MN	MOSFETs (N-channel)
MP	MOSFETs (P-channel)
O	Lossy Transmission Lines
QN	Bipolar Junction Transistors (NPN)
QP	Bipolar Junction Transistors (PNP)
R	Resistors
S	Voltage Controlled Switches
T	Lossless Transmission Lines
U	Uniform Distributed RC Lines (Lossy)
V	Independent Voltage Sources
W	Current Controlled Switches
X	Subcircuits
ZN	MESFETs (N-channel) (GaAs FETs)
ZP	MESFETs (P-channel) (GaAs FETs)

Analog

This check box identifies this as a device that can be used in CircuitMaker's Analog simulation mode. The analog simulator can only simulate a device if there is Spice simulation data for that device. If you attempt to run an Analog simulation using a device that does not have the Analog check box checked, CircuitMaker displays a warning message and that device will be ignored in the simulation. When creating your own device, this box should be checked only if you have supplied Spice data for the device or if the device is a macro circuit containing other analog devices (see *Chapter 17: Creating New Devices* for more information on new device creation).

Digital

This check box identifies this as a device that can be used in CircuitMaker's Digital simulation mode. The digital simulator can only simulate a device if there is digital SimCode for that device. If you attempt to run a digital simulation using a device that does not have the Digital check box checked, CircuitMaker displays a warning message and that device will be ignored in the simulation. When creating your own device, select this box only if the device is a macro circuit containing other digital devices or was created using digital SimCode (see *Chapter 18: Digital SimCode* for more on digital device creation).

Parameters

This field stores information that affects the simulation of certain devices. For digital SimCode devices, this field would contain **type:digital**. It could then be followed by a list of parameters which are generally set in the Digital Model Parameters dialog box.

Some generic device models can be redefined by passing parameters as an alias to describe a specific device. For example, the parameter field of a crystal would contain **alias:XCRYSTAL** and could be followed by a list of databook parameters that define that specific crystal. Generally, you enter these parameters in the Subcircuit Parameters dialog box. See *Chapter 17: Creating New Devices* for more information on parameter passing.

Bus Data

Use this field to specify which pins on the device are connected to the power or ground buses, since these pins are not shown on the predefined digital device packages. This field holds up to 2048 characters, which can be helpful when creating devices with many power and ground pins. The general format for this data is:

bus1=pin#,pin#,…;bus2=pin#,pin#,…;

For example, on a 74LS83 the Bus Data is:

DVCC=5;DGND=12;

This means that the Vcc bus is connected to pin 5 of this device and the Ground bus is connected to pin 12. The bus data for a 74AC11190 would look like:

DVCC=15,16; DGND=4,5,6,7;

The order in which the buses are listed is not important.

For Analog simulation or for creating a PCB netlist to export into TraxMaker, each bus must be connected to a voltage source. There are three non-digital device items that can be connected directly to the Bus Data: **+V**, **Terminal** and **Ground**.

The Bus Data format for these devices is simply:

busname;

where busname identifies the specific bus. For more information, refer to *Power Bus Connections* earlier in this chapter.

Each Ground device, by default, contains the Bus Data **GND;** which causes all of the Ground symbols to be tied together in a single node or net. For simulation purposes, the Ground device *always* equates to Spice node 0, even if the Bus Data changes.

Spice Data

This field is used to specify the Spice simulation data for the device. The analog devices provided with CircuitMaker already have default Spice data included. If you create your own devices to use with the analog simulation, you will need to fill in this field yourself. You can enter Spice data into this field directly, or you can reference it to the other fields in the dialog box. The percent sign (%) is used as a flag to tell CircuitMaker to reference the already defined fields. Their meanings are:

Name (%N)

Inserts the Device Name into the Spice data string. This is the name found in the library menus.

Label (%L)

Inserts the Label-Value into the Spice data string. This requires the first character in the Label-Value string to be an alpha character. The label may not exceed 8 characters.

Value (%V)

Inserts the Label-Value into the Spice data string. This requires the first character in the Label-Value string to be numeric. The value may be an integer (12, -44), a floating point number (3.14159), either an integer or floating point number followed by an integer exponent (1e-14, 2.65e3), or an integer or floating point number followed by one of the following multipliers:

T	= 10^{12}	u	= 10^{-6}
G	= 10^{9}	n	= 10^{-9}
Meg	= 10^{6}	p	= 10^{-12}
K	= 10^{3}	f	= 10^{-15}
m	= 10^{-3}		

If multipliers are used, they must immediately follow the number with no spaces. Letters that are not multipliers immediately following a number are ignored, and letters immediately following a multiplier are ignored. For example, 10, 10V, 10Volts, and 10Hz all represent the same number and M, MA, Msec, and MMhos all represent the same multiplier. Note that 1000, 1000.0, 1000Hz, 1e3, 1.0e3, 1KHz, and 1K all represent the same number.

Model (%M)
Inserts the Label-Value into the Spice data string. If the first character of the Spice Prefix Character(s) does not match the first character in the label, then it inserts the first character of the prefix at the beginning of the string. The %M is also required in order to guarantee that the .MODEL data for this device will be included in the Spice netlist file for simulation. The Label (including prefix) may not exceed 8 characters.

Subcircuit (%S)
Inserts the Label-Value into the Spice data string. If the first character in the label is not an **X**, then it inserts an **X** at the beginning of the string. The %S is also required to ensure that the .SUBCKT data for this device will be included in the Spice netlist file for simulation. The Label (including the **X**) may not exceed 8 characters.

Designation (%D)
Inserts the Designation label into the Spice data string. If the first character of the Spice Prefix Character(s) does not match the first character in the designation, then it inserts the first character of the prefix at the beginning of the string.

Description (%I)
Inserts the Description field into the Spice data string.

Package (%P)
Inserts the Package label into the Spice data string.

Bus Data (%B)
Inserts the entire Bus Data field into the Spice data string.

Named Subcircuit (%X)
Inserts the specified subcircuit into the Spice data string. If the subcircuit resides in a .SUB file other than the one associated with the symbol name of the current device, the .SUB file name where the subcircuit is located must also be specified. For example: **%XUA741** or **%XUA741:OPAMP5**.

Include File (%=*"path\filename.ext"*)
Inserts the ASCII text file *filename* into the Spice data string by using Spice's .INCLUDE command. Path is the same as the current circuit unless enclosed in quote marks (" ").

Node (%*number*)
Inserts the node number for the specified pin into the Spice data string. *Number* refers to the position of the pin in the Edit Device Pin Data dialog box. For example, the pin at the top of the list is represented by %1, the next pin down is %2, etc. See *Pins* later in this section.

Analysis Probe Name (%[ )
Inserts the output node number associated with an Analysis Probe. For example, %[TP1] or %[+TP1] inserts the node number associated with the + pin of the Analysis Probe that has the name TP1. %[-TP1] inserts the node number associated with the - pin.

Example of Using Spice Data

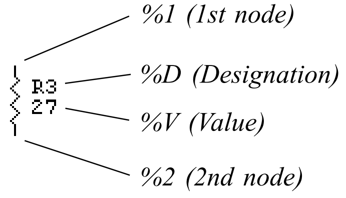
The following examples show how you can use the Spice Data field when creating custom devices. For more information on Spice, see *Chapter 16: Spice: Beyond the Basics*.

Example 1

If resistor R3 has a value of 27 ohms and is connected between node 5 and ground, the Spice data for R3 could be written as:

R3 5 0 27ohms

However, by using a generalized form of the Spice data, it can be updated automatically if items such as designations or node numbers change. For example, the items in the above example can be substituted accordingly:

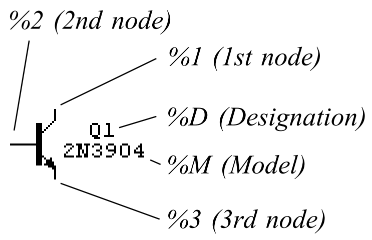


Value	Description	Substitution
R3	Designation	%D
5	First pin node number	%1
0	Second pin node number	%2
27ohms	Label-Value	%V

Using these substitutions, the generalized form of the Spice Data becomes:

```
%D %1 %2 %V
```

Example 2

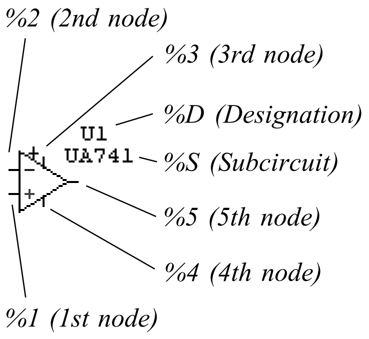


If transistor Q2 is a 2N3904 with its collector connected to node 7, its base to node 4 and its emitter to node 12, the Spice data may be written as:

Q2 7 4 12 Q2N3904

However, you may need to enter Spice model data into the netlist manually. A more universal method would be to enter the following string:

```
%D %1 %2 %3 %M
```



Example 3
If op amp U4 is an LM741 where the +input is connected to node 3, the -input is connected to node 1, the +V pin to node 5, the -V pin to node 12 and the output to node 9, the Spice data may be written as:

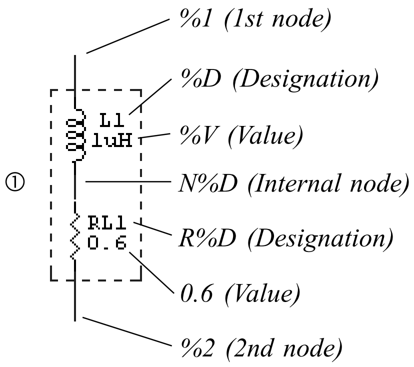
```
XU4  3  1  5  12  9  XLM741
```

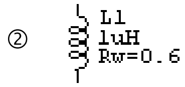
But again, a more universal method would be to enter the following string:

```
%D %1 %2 %3 %4 %5 %S
```

Example 4
You may want to add something to your circuit for simulation purposes that does not actually show up on the schematic itself. For example, an inductor may require an internal winding resistance of 0.6 ohms for proper simulation. The desired Spice netlist might look something like this:

```
L3  3  4  1uH
RL  4  5  0.6
```





These two inductor/resistor combinations are identical in Spice when given the Spice Data shown.

Enter the following *two* lines into the Spice Data field of the inductor only. *You do not need to wire the resistor into the circuit*:

```
%D %1 N%D %V
R%D N%D %2 0.6
```

CircuitMaker will produce a Spice netlist something like this:

```
L3  3  NL3  1uH
RL3  NL3  5  0.6
```

Notice the use of the %D in the node name and in the resistor designation. This allows us to use this same device in our circuit multiple times without conflict because each one will be unique.

Exclude From PCB

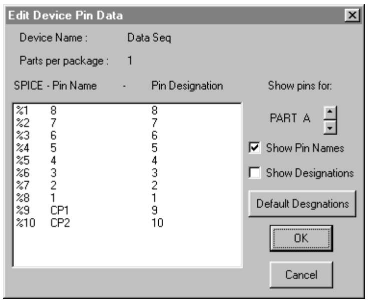
Use this check box to exclude a device from the PCB netlist. Typically, devices used for simulation purposes only (external inputs like a signal generator, for example) would be excluded from the PCB netlist.

Exclude From Parts

Use this check box to exclude a device from the Bill of Materials. See *Exporting a Bill of Materials* in *Chapter 7: Exporting Files* for more information on generating a Bill of Materials.

Pins

Click the **Pins** button on the Edit Device Data dialog box to display the dialog box shown in Figure 4.15. This dialog box lets you edit the pin designations of the package for the selected device and determine whether the pin designations will be shown on the schematic.



Edit Device Pin Data

Device Name : Data Seq

Parts per package : 1

SPICE	- Pin Name	- Pin Designation
%1	8	8
%2	7	7
%3	6	6
%4	5	5
%5	4	4
%6	3	3
%7	2	2
%8	1	1
%9	CP1	9
%10	CP2	10

Show pins for: PART A

[✓] Show Pin Names

[ ] Show Designations

Figure 4.15. Use this dialog box as a reference for pin Spice information.

You can specify pin numbers using up to five alphanumeric characters. Not only can you use standard pin numbers such as 1, 2, 3, but you can also use pin numbers such as A1, B1, C1, A2, B2, C2.

Some device packages actually contain more than one of the same device. For example, a 7400 Quad 2-Input NAND gate actually has 4 gates in the same package. The pin numbers are different for each gate. CircuitMaker groups together the individual gates to indicate which gates go in which package. As you place each gate in the circuit, the next available gate is used from the previous package. If no gates are available in the previous package, a new package is used. You can regroup the gates as needed using the **Edit** > **Group Items** option. Each gate in the package is assigned a letter (A, B, C, etc.) and the pin numbers for that gate correspond to that assignment. You can manually reassign the pins used for a particular gate by selecting the appropriate letter (PART A, PART B, etc.) for that gate with the **Up** and **Down Arrows** in the dialog box.

Default pin data has already been entered for the predefined devices. Clicking the **Default Designations** button restores the pin numbers to their original default values. You can add default pin numbers to a macro device by editing the pin numbers while the macro is expanded and while saving the macro.

To edit macro pin names and designations,

1 Double-click them in the Symbol Editor (see *Chapter 17: Creating New Devices* for more information).

The order in which the pins appear in the list is determined by the order in which you placed pins on the device.

To edit pin designations for a specific part,

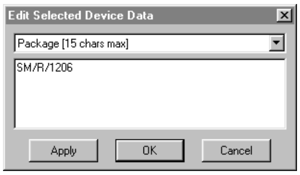
1 Double-click them in this dialog box. This does not change the pin designations in the library.

Faults

Clicking the **Faults** button on the Edit Device Data dialog box displays the Device Faults dialog box, which lets you add fault data to the device. If the fault data has been password protected, the Access Faults dialog box appears. See *Chapter 8: Fault Simulation* for details.

Editing Multiple Devices

You can quickly change one of the device fields for all selected devices. Just Right-click on one of the selected devices and choose **Selected Device Properties**. Choose the field you want to edit, then type in the appropriate text. See Figure 4.16.


Edit Selected Device Data

Package [15 chars max]

SM/R/1206

Figure 4.16. Edit all the selected devices at one time.

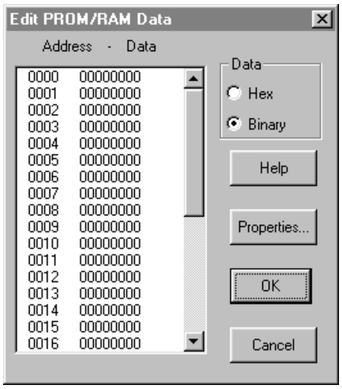
Editing PROM/RAM Data

The internal data of a PROM or RAM device can only be edited by Right-clicking on the device and selecting **Edit PROM/RAM** from the pop-up menu. Once a PROM is programmed, it retains its contents until reprogrammed. When you save a circuit containing a PROM to disk, the contents of the PROM are also saved. For debugging purposes, the contents of the RAM can be viewed and edited, but the data will not be saved with the circuit.

To edit a PROM or RAM device,

1 Right-click a single PROM or RAM, and then choose **Edit PROM/RAM** to display the dialog box pictured in Figure 4.17.

2 Program the device as desired, then choose OK.



```
0000  00000000
0001  00000000
0002  00000000
0003  00000000
0004  00000000
0005  00000000
0006  00000000
0007  00000000
0008  00000000
0009  00000000
0010  00000000
0011  00000000
0012  00000000
0013  00000000
0014  00000000
0015  00000000
0016  00000000
```

Figure 4.17. Use this dialog box to enter either hexadecimal or binary numbers for the PROM/RAM device.

Printing and Exporting Schematics

When you have finished designing your circuit, you can print the schematic on any Windows-selectable printer or plotter, or export it to a file and use it in documentation, presentations, etc.

Printing Schematics

To print a circuit, choose **File** > **Print Schematic**.

If your design is larger than a single sheet of paper it will automatically be printed on multiple sheets of paper. To see where page breaks will occur, choose **Options** > **Schematic** and enable **Show Page Breaks**.

Paper Size vs. Print Scale

Paper Size refers to the physical dimensions of the paper on which you will be printing. Paper sizes are selected solely through your printer's software or "printer driver." CircuitMaker uses the settings from the printer driver to help determine the positioning of the Page Breaks. For example, if your printer supports "B" size paper, then selecting that option in the printer's Properties dialog will also setup CircuitMaker's page breaks accordingly. Borders, title blocks and portions of the schematic that fall within the page breaks will be adjusted to fit on that specific size of paper.

Print Scale refers to the enlargement or reduction of the schematic when printed. The print scale can be adjusted from 10% to 1000%. 10% would make the printed schematic symbols look very tiny when printed. 1000% would make them appear very large. Adjusting the print scale allows you to fit the schematic, or portions of it, to a single sheet of paper. The print scale, then, along with the paper size, helps to determine the position of the page breaks.

The print scale can be visually adjusted by dragging the page breaks with the mouse to fit the portion of the schematic that you want to fit on a sheet of paper.

To adjust the paper size,

1 Choose **File** > **Print Setup** and click on the **Printer** button.

2 Select the paper size. Options vary according to the specific printer driver. Look for a Properties button.

To adjust the print scale of your schematic,

1 Choose **File** > **Print Setup** and select **Fit to Page** and the schematic will automatically be scaled to fit on a single page, or choose **Scale** and enter the percentage you would like the circuit scaled to.

OR

Choose **Options** > **Schematic** and enable **Show Page Breaks**. Return to the schematic and click and drag one of the page breaks, until the circuit fits in the page as you want it to print.

Other print options are also available, including color printing. See *Print Setup* in *Chapter 9: File Menu*. You can also print digital timing diagrams, analog waveforms, title blocks, borders, and grids. See *Chapter 12: Options Menu* for more information.

Exporting Schematics as Graphics

You can export CircuitMaker circuit schematics and use them in documentation, presentations, etc. You can either save the schematic as a graphic file, or copy and paste the schematic directly into another software program.

Use the Export Schematic Graphic File option to save the circuit to disk as a Windows Metafile, Device Independent Bitmap or Device Dependent Bitmap. Use the Export Options dialog box, described earlier, to choose the format.

To export the circuit as a graphic,

1 Choose **File > Export > Schematic Graphic File**. Select the name of the file where you want to save the schematic graphic, then choose **Save**.

OR

Choose **Edit** > **Copy to Clipboard** > **Schematic**, then open another Windows program and Paste the circuit directly into your document. See also *Chapter 7: Exporting Files.*

C H A P T E R 5

Digital Logic Simulation

One of CircuitMaker's most powerful features is circuit simulation, allowing you to try variations in a design and troubleshoot it before you invest time and money in hardware prototypes.

CircuitMaker's Simulation Modes

CircuitMaker is one of the few simulation programs that offers 2 distinct modes of simulation: Analog mode and Digital mode. This gives you greater flexibility and control over how your circuit is simulated, and each mode has advantages depending on the type of simulations you need.

Analog Mode is the accurate, "real-world" simulation mode you can use for analog, digital and mixed-signal circuits. This mode will give you results like you would get from an actual breadboard. In Analog mode, the devices function just like real-world parts, and each individual model functions like its real-world counterpart. For example, digital ICs have accurate propagation delays, setup and hold times, etc. Outputs of the devices see the effect of loading on them, and nearly all the parameters of the real world are taken into account. See *Chapter 6: Analog/Mixed-Signal Simulation* for more on this mode.

Digital Mode, on the other hand, is designed for purely digital logic simulation. This mode is only used for digital circuits, and depends solely on the logic states of the devices that make up the circuit. Digital mode simulation still takes into account propagation delays, but they are unit delays instead of actual propagation delays. No power supply is required, and the digital device output levels are constant in this mode.

Digital electronics is the world of the computer. The binary 1's and 0's of the computer are actually the high and low voltage levels of tiny electronic devices known as integrated circuits. Digital logic simulation, then, becomes a relatively simple task because of the limited number of digital states that must be represented. CircuitMaker's digital logic simulator is very fast and fully interactive, meaning you can flip switches and alter the circuit while the simulation is running and immediately see the response.

Devices and Simulation

CircuitMaker provides four types of devices, which can be used in different simulation modes.

Device Type	**Will Function In**
Digital Only device	Digital simulation mode only
Analog Only device	Analog simulation mode only
Analog/Digital device	Analog or Digital mode (most devices fall into this category)
Schematic Symbol only	No functionality

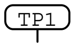
So any **Digital Only** and **Analog/Digital** devices will function properly in the Digital mode. Refer to the Device Library Guide for a description of each device and the simulation mode for which it is intended.

If you attempt to simulate a device using a simulation mode for which the device is not intended, CircuitMaker displays a warning message and that device is ignored, leaving an open circuit where that device is located.

Using the Digital Logic Simulator

Digital simulation is completely interactive, meaning that the circuit responds immediately to changes from input stimulus, and the operation of the circuit is shown in real time as it happens on the screen. You can observe the operation of the circuit in the following ways:

- Enable CircuitMaker's exclusive **Trace** feature to show the state of every node in the circuit simultaneously as the simulation runs. In this mode, wires at a logic one are shown as red, wires at a logic zero as blue, and wires at an unknown or tristate as green (these colors may be changed in the Schematic Options dialog box).



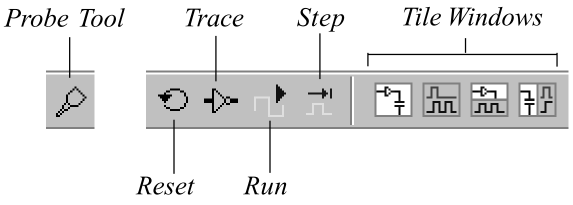
SCOPE

- Connect any number of SCOPE devices [Instruments/ Digital/SCOPE] (T) to any nodes in the circuit, so that the timing diagrams for those nodes are shown in the digital Waveforms window. The timing information is updated continuously to show changes as they happen in real time.
- Connect any of a variety of displays and note the conditions shown on them.
- Use the **Probe Tool** to probe any wire in the circuit either during simulation or after you have stopped it. The logic states seen by the Probe Tool can also be charted in the digital Waveforms window.

Digital Logic Simulation Tools

Several buttons in the Toolbar are used specifically for simulation. This section describes these tools.

Note: These buttons are somewhat different in CircuitMaker's Analog mode. See *Chapter 6: Analog/Mixed-Signal Simulation* for more information.



Probe Tool

High State

Low State

Pulse (between High and Low States)

Unknown or Tristate

Use the **Probe Tool** to monitor the state of any node in the circuit or to inject a state into a node. You can also activate the Probe Tool by right-clicking the mouse and choosing **Probe** from the pop-up menu.

To see the state of a node, either while the simulation is running or after it has stopped, touch the Probe Tool's tip on a wire or device pin. The tool displays one of four messages: H, L, P, or no letter at all. The meanings of these letters are illustrated at left. Choose **Simulation** > **Active Probe** beforehand to have CircuitMaker chart the waveforms you see with the Probe Tool in the Waveforms window.

To inject a state into a node,

1 Touch the tip of the **Probe Tool** on a wire or device pin.

2 Click the left mouse button.

The state of that node changes to be opposite of what it was (a one becomes a zero and a zero becomes a one).

To inject a tristate signal,

1 Hold down the **Shift** key and click.

Note: In both cases, if the node is driven by some other device, the state change is immediately overridden because the device drives the node back to its original state.

You can also use the Probe Tool to flip switches while the simulation is running.

Reset Button

Click the **Reset** button to restart the simulation. You can also reset by choosing **Simulation** > **Reset** or by pressing **Ctrl+Q**.

Trace Button

Click the **Trace** button (or press **F11**) to turn the trace feature on or off. Use Trace to debug your circuit or to simply provide a convenient way of observing operation of the circuit. Trace shows the state of all nodes within the circuit as it runs by drawing the wires in different colors to show the logic state of each wire. A wire at a high state is red, a wire at a low state is blue, and a tristate wire is green.

Note: Because the wires in the circuit will be redrawn each time they change states, turning this option on may slow down the simulation speed.

Run/Pause Button

Click the **Run** button to start the simulation. The icon will change to a Pause icon. Click the **Pause** button to stop the simulation (You can also choose **Simulation** > **Run** and

Simulation > **Pause** or press **F10**). When the simulation is running you can't perform edit operations such as move and delete (if you try, CircuitMaker will signal a warning beep). You can flip switches with the **Arrow** or **Probe Tool**, and view or toggle the state of a wire with the **Probe Tool**.

Step Button

Click the **Step** button to run the simulation for one step or simulation "tick". You can also choose **Simulation** > **Step** or press **F9**. Use the Digital tab in the Panel (see Figure 5.3) to control the size of a step. When you click this button, the simulation runs for one step and then stops. This command is handy for debugging a circuit, especially when used in conjunction with the **Trace** button.

Tile Windows Buttons

Tiles the drawing and digital waveform windows into one of four views. In digital mode, the simulation does not need to be running to view the waveform window. Digital waveforms are described in detail later in this chapter.

Propagation Delays

The delay of a device determines how many simulation ticks it takes for a signal to propagate from the input to the output of the device. The default delay for all devices is 1, but you can change this to any value from 1 to 14. You determine the real-time value of each tick. The concept is that if one device has a delay of one and another a delay of three, then in the real world the second device would have a propagation delay three times larger than the first device.

To change the delay of one or more devices,

1 Select the device(s).

2 Choose **Edit** > **Set Prop Delays** to display the dialog box shown in Figure 5.1.

3 Enter a new value for the delay and choose OK.

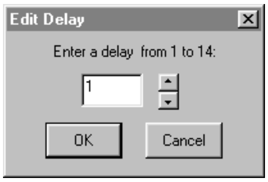


Figure 5.1. Use this dialog box to change the propagation delay of one or more selected devices.

Choose **Options** > **Schematic** and enable **Show Prop Delays** to display the propagation delay of all devices in the circuit. The delay values are shown within a rounded rectangle located near the center of each device. Some devices (Pulsers, Logic Displays, macro devices, etc.) do not have a delay, so no value will be shown. In the case of macro devices, the delay is determined by the individual delay setting of each device within the macro.

***Caution:** If your input stimulus (pulser, data sequencer) is not running slower than the longest prop delay, you will get unexpected results.*

Digital Waveforms

By attaching SCOPEs [Instruments/Digital/SCOPE] (T) to points of interest in the circuit, you can graph the states of these nodes over time as the simulation runs. Click one of the **Tile Windows** buttons in the Toolbar to display or hide the digital waveforms window. An example of the waveforms window is shown in Figure 5.2.

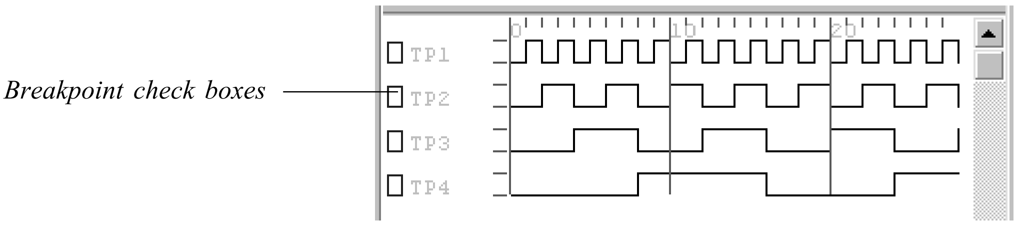


Figure 5.2. The digital waveforms window lets you graph the states of nodes over time as the simulation runs.

Before you can view timing waveforms for any node in your circuit, you must connect a SCOPE to each node you want to monitor, or choose **Simulation** > **Active Probe** to monitor the states of the Probe Tool in the digital waveforms window.

Changing Waveform Order

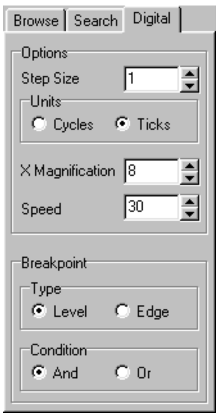
To change the order of waveforms,

1. Point the mouse at any of the scope labels in the waveform window.
2. Press and continue to hold down the left mouse button.
3. Move the rectangle to the desired position.
4. Release the mouse button.

Notice that the waveforms are automatically reordered in the window. Repeat this process as often as desired in order to position the waveforms in any order. When you save a circuit to disk, CircuitMaker also saves the order of the waveforms.

Digital Options

Use the Digital tab in the Panel to control the size of a step when running the simulation in single step mode, to set the X magnification and simulation speed, and to set the conditions for break points. See Figure 5.3.



Options
Step Size 1
Units: Cycles [✓ Ticks]
X Magnification 8
Speed 30

Breakpoint
Type: [✓ Level] Edge
Condition: [✓ And] Or

Figure 5.3. Use the Digital tab in the Panel to specify the size of a step and other digital simulation options.

You can define the **Step Size** in either ticks or cycles. A cycle always consists of 10 ticks. A tick is the smallest unit of delay for the digital simulator. It takes one tick to perform a single step of the simulation for all devices.

Adjust **X Magnification** to view a larger or smaller section of the waveforms in the digital Waveforms window. By default, the magnification is set to 8. A smaller value zooms out, a greater value zooms in.

Use **Simulation Speed** to control how fast the simulation runs. This could be useful, for example, if the simulation is running too fast to view the states of a seven-segment display. Setting this field to a lower number slows down the simulation so you can view the changes of the display. Another method of slowing the simulation would be to run it in single step mode or set breakpoints.

Use the Breakpoint **Type** and **Condition** options in conjunction with the waveforms window to set breakpoints. The following table illustrates the results of various combinations of settings.

Combination	**Result**
Level-And	All break conditions must be met before the simulation stops.
Level-Or	Any one of the break conditions stops the simulation.
Edge-And	The simulation stops when the proper edge occurs simultaneously on all of the specified waveforms.
Edge-Or	The simulation stops if a transition to any of the specified conditions occurs.

Setting Breakpoints in a Circuit

Use the breakpoint check boxes in the digital waveforms window (see Figure 5.2) to set breakpoints in a circuit.

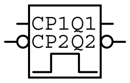
To set a breakpoint,

1. Click once in the small breakpoint check box to the left of a SCOPE's label in the Waveforms window to fill the bottom portion of the square indicating a break on zero condition.
2. Click a second time to fill the top portion of the square indicating a break on one condition.
3. Click a third time to return the square to its empty state indicating no break condition.

Digital Instruments

This section introduces two digital instruments: the Pulser and the Data Sequencer.

Pulser



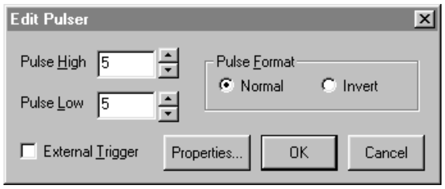
The Pulser [Instruments/Digital] (p) is a digital pulse generator which provides a continuous stream of highs and lows. In the pulse format, time high, time low, and trigger mode are individually programmable for each Pulser in the circuit.

To edit the Pulser settings,

1 Double-click the Pulser with the **Arrow Tool** to display the dialog box shown in Figure 5.4.

2 Change the number of simulation ticks for which the pulse will stay high and low, the format of the pulse (normal or inverted), and whether the pulser is in free run or external trigger mode.

Caution: *The Pulse High and Pulse Low settings must be set longer than the longest device prop delay or you will get unexpected results. See Propagation Delays earlier in this chapter.*



Edit Pulser

Pulse High: 5

Pulse Low: 5

Pulse Format: [✓ Normal] Invert

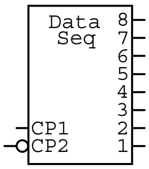
☐ External Trigger

Figure 5.4. Use the Edit Pulser dialog box to change pulse ticks and set free run or external trigger mode.

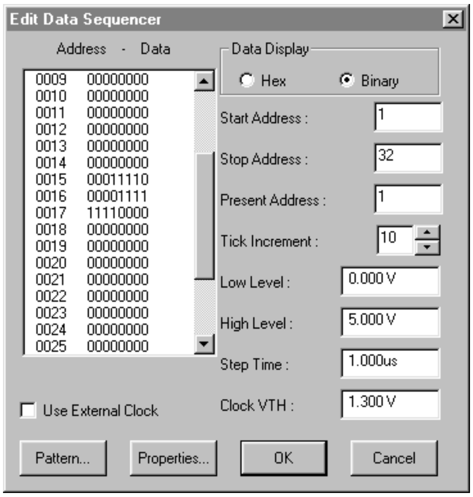
3 Select **External Trigger** to use the Pulser as a programmable one-shot.

In the External Trigger mode, the CP1 and CP2 inputs serve as rising and falling edge trigger inputs, respectively. If either pin receives a trigger pulse then the outputs of the Pulser will go active on the next simulation tick and remain active for Pulse High ticks. You can retrigger the Pulser in this mode so additional trigger pulses occurring before the completion of the cycle will cause the active time to be extended for another cycle.

Data Sequencer



You can use the Data Sequencer [.General/Instruments/Data Seq] (G) device in both digital and analog simulation modes. Also known as a Data Generator or Word Generator, it allows you to specify up to 32767 8-bit words which can be output in a defined sequence. Since there is no limit to the number of Data Sequencers that you can use in a circuit, you could place several in parallel to create a data stream of any width. Double-click the **Data Sequencer** with the Arrow Tool to display the dialog box pictured in Figure 5.5.



Address - Data

```
0009   00000000
0010   00000000
0011   00000000
0012   00000000
0013   00000000
0014   00000000
0015   00011110
0016   00001111
0017   11110000
0018   00000000
0019   00000000
0020   00000000
0021   00000000
0022   00000000
0023   00000000
0024   00000000
0025   00000000
```

Data Display: Hex [✓ Binary]

Start Address : 1

Stop Address : 32

Present Address : 1

Tick Increment : 10

Low Level : 0.000 V

High Level : 5.000 V

Step Time : 1.000us

Clock VTH : 1.300 V

☐ Use External Clock

Figure 5.5. Use the Edit Data Sequencer dialog box to edit stimulus for your circuits.

Start Address is the address of the data that outputs first when the simulation begins. **Stop Address** is the address of the data that outputs last before the sequence repeats. Use **Use External Clock** to make the CP1 and CP2 inputs rising and falling edge clock inputs, respectively. If pins are clocked, the Data Sequencer advances to the next address.

Digital Simulation Mode Only

The **Present Address** option indicates the address of the data that is output next if the circuit is not reset or modified. The **Tick Increment** specifies how many simulation ticks occur before the output is advanced to the next address when the external clock is disabled.

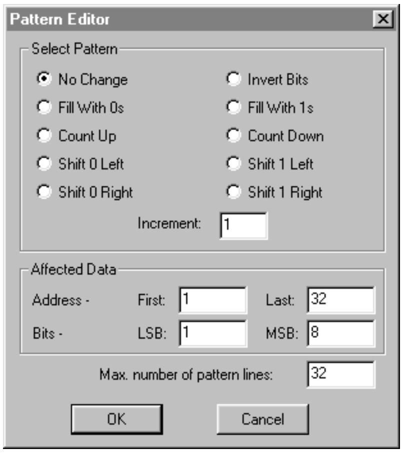
Caution: *The Tick Increment setting must be set longer than the longest device prop delay or you will get unexpected results. See Propagation Delays earlier in this chapter.*

Analog Simulation Mode Only

Low Level and **High Level** indicate the output voltage levels. **Step Time** is the length of time that the outputs remain at each address when the external clock is disabled. **Clock VTH** is the voltage threshold level at which the external clock pins cause the outputs to advance to the next address.

Pattern Editor

You can type data directly into the Data list box in the Data Sequencer dialog box. However, when creating a large pattern this method becomes time consuming. Choose the **Pattern** button on the Data Sequencer dialog box to display the dialog box shown in Figure 5.6. The Pattern Editor helps you to create large, complex patterns quickly.



Select Pattern

[✓ No Change]	Invert Bits
Fill With 0s	Fill With 1s
Count Up	Count Down
Shift 0 Left	Shift 1 Left
Shift 0 Right	Shift 1 Right

Increment: 1

Affected Data

Address -	First:	1	Last:	32
Bits -	LSB:	1	MSB:	8

Max. number of pattern lines: 32

Figure 5.6. Use the Pattern Editor to help you quickly create large, complex patterns.

The Pattern Editor not only lets you to enter predefined pattern sequences, it also lets you specify which rows (addresses) and columns (bits) are affected. For example, you could fill just one column with a count up sequence to produce a stream of ones and zeros on a single output. Or you could fill rows 23-67 with ones in just 5 columns and fill the same rows with a repeating Shift 0 Left in the other 3 columns.

The **Increment** field indicates how many rows will contain the same data before the next change in pattern. For example, with an increment of 3, a **Shift 1 Left** pattern shifts on every third pattern row.

You can set the maximum pattern size for each Data Sequencer. This lets you create small patterns for several Data Sequencers without allocating large amounts of memory or creating large circuit files. The **Max. number of pattern lines**, by default, is set to 32, but can be increased as needed to as many as 32767. It can never be smaller than the Stop Address. When you *increase* the maximum number of pattern rows, the new rows are filled with zeros. If you *decrease* the maximum number of pattern rows, any pattern data that was stored in the upper addresses is lost permanently.

CHAPTER 6

Analog/Mixed-Signal Simulation

One of CircuitMaker's most powerful features is circuit simulation, allowing you to try variations in a design and troubleshoot it before you invest time and money in hardware prototypes.

CircuitMaker's Simulation Modes

CircuitMaker is one of the few simulation programs that offers 2 distinct modes of simulation: Analog mode and Digital mode. This gives you greater flexibility and control over how your circuit is simulated, and each mode has advantages depending on the type of simulations you need.

Analog Mode is the accurate, "real-world" simulation mode you can use for analog, digital and mixed-signal circuits. This mode will give you results like you would get from an actual breadboard. In Analog mode, the devices function just like real-world parts, and each individual model functions like its real-world counterpart. For example, digital ICs have accurate propagation delays, setup and hold times, etc. Outputs of the devices see the effect of loading on them, and nearly all the parameters of the real world are taken into account.

SPICE: Simulation Program with Integrated Circuit Emphasis

Analog is the classic world of electronics. Unlike digital electronics, there are no logic state restrictions; the voltage level of any given circuit node is not limited to a high or low. Analog simulation, therefore, is much more complex. CircuitMaker's analog/mixed-mode simulation uses an enhanced version of Berkeley Spice3f5/XSpice, allowing you to accurately simulate any combination of analog and digital devices without manually inserting D/A or A/D converters. This "mixed-signal" or "mixed-mode" simulation is possible because CircuitMaker includes accurate, event-driven behavioral models for its digital devices, including TTL and CMOS digital devices.

In Analog mode, there are a also a wide variety of analyses that can be used in Analog mode to test and analyze various aspects of your design.

Digital Mode, on the other hand, is designed for purely digital logic simulation. This mode is only used for digital circuits, and depends solely on the logic states of the devices that make up the circuit. Digital mode simulation still takes into account propagation delays, but they are unit delays instead of actual propagation delays. No power supply is required, and the digital device output levels are constant in this mode. See *Chapter 5: Digital Logic Simulation* for more on this mode.

Devices and Simulation

CircuitMaker provides four types of devices, which can be used in different simulation modes.

Device Type	Will Function In
Digital Only device	Digital simulation mode only
Analog Only device	Analog simulation mode only
Analog/Digital device	Analog or Digital mode (most devices fall into this category)
Schematic Symbol only	No functionality

So any **Analog Only** and **Analog/Digital** devices will function properly in Analog mode. To find out the intended simulation mode for a device, refer to the Device Library Guide for a description of each device and the simulation mode for which it is intended.

Overview of Analog Simulation

This section explains the basic concepts for simulation in CircuitMaker's analog mode.

Before You Use the Analog Simulator

To do analog simulation, you must ensure there is Spice information for each device in the circuit. Only those devices listed as Analog or Analog/Digital in the Device Library book have Spice data associated with them. You can use other devices in the circuit as long as you provide the Spice information for those devices. See *Chapter 17: Creating New Devices* for more information.

The **Analog** check box in the Device Properties dialog box indicates whether or not that device will function in Analog simulation mode (meaning there is Spice simulation data available for the device). If the **Analog** check box is not checked and you use the device in analog simulation, a warning appears and that device is ignored, leaving an open circuit where that device is located.

Setting Up Analog Analyses

You set up analog analyses using the Analyses Setup dialog box described later in this chapter. By default, whenever you create a new circuit, the **Always Set Defaults** option is enabled for the analog analyses. This means that Operating Point Analysis (the Multimeter) is enabled for simple DC circuits. For more complex circuits, Transient Analysis (the oscilloscope) is enabled and set to its default conditions.

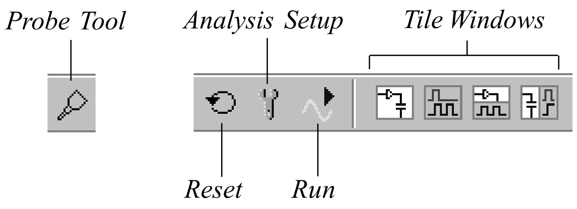
Selecting Analog Simulation Mode

In the Simulation menu, make sure that **Analog Mode** is checked rather than **Digital Mode**.

Analog Simulation Tools

Several buttons in the Toolbar are used specifically for simulation. This section describes these tools.

Note: These buttons are somewhat different in Digital Logic mode. See *Chapter 5: Digital Logic Simulation* for more information.



Probe Tool

The context-sensitive **Probe Tool** allows you to quickly probe any point(s) in the circuit during simulation and see the resulting waveform or data in the Analysis Window. Note that the Probe Tool is used differently in Analog mode than it is in Digital mode.

Before running a simulation, you can left-click the Probe tool in the circuit to add or remove Run-Time Test Points. Note, however, that you **do not** have to pre-define test points before running a simulation in CircuitMaker. See *Working with Test Points* later in this chapter for more information on using test points.

During simulation, touch the tip of the Probe Tool to a wire, device pin or device body to observe or plot data at that point. The tool displays one of six letters: V, I, P, Z, N, or R. The meanings of these letters are illustrated at left.

V	*Voltage*
I	*Current*
P	*Power*
Z	*Impedance*
N	*Noise (during Noise Analysis)*
R	*Input or output Resistance (during Transfer Function analysis)*

Click the left mouse button while the Probe Tool is at the point in the circuit you wish to probe, and a value or waveform appears instantly in the current (active) analysis window. *To see multiple waveforms*, simply Shift-click with the Probe Tool on as many points as you wish and the corresponding waveforms "stack" in the current analysis window.

Note: If CircuitMaker indicates that current or power data is not available, stop the simulation and click the **Analyses Setup** button on the Toolbar. Click the **Analog Options** button and select the **Node Voltage, Supply Current, Device Current and Power** radio button in the lower right-hand corner of the dialog box. Then rerun the simulation.

Reset Button

In Analog mode, clicking the **Reset** button resets the simulation and generates the node numbers for the circuit without running the simulation. This is important if you want to save a Spice netlist file or view the node numbers on the schematic, but not run the simulation. You can also reset the simulation by choosing **Simulation** > **Reset** or by pressing **Ctrl+Q**.

Run/Abort/Stop Button

Click the **Run** button to start the simulation. The icon will change temporarily to a square Abort button. You can click the **Abort** button to stop the simulation in progress without losing the data that has already been collected up to that point. When the simulation process is complete the icon changes to a round Stop button. Click the **Stop** button to close the Analysis Window and return to editing mode. You can probe the circuit at any time during or after simulation until you return to editing mode. The amount of time it takes to complete the simulation process is based on the analyses you have enabled, the amount of data you are collecting, the complexity of the circuit, and the speed of your computer.

Note: If no changes have been made since the last simulation was run, clicking the Run button will not rerun the simulation, but will immediately load the previous simulation data.

Tile Windows Buttons

Tiles the schematic and analysis windows into one of four views. If the simulation is stopped and has not been reset and the schematic has not been modified since the last time the simulation was run, clicking one of these buttons will immediately load the previous simulation data.

Working with Test Points

Note: *You* ***do not*** *have to set test points before running a simulation. CircuitMaker does it automatically according to the chosen analyses.*

Test Points are set in a circuit to tell CircuitMaker where to collect simulation data. Note that you **do not** have to manually set test points before running a simulation (CircuitMaker does it automatically). Test Points determine how much data is actually stored, and they determine which variables are displayed in the analysis windows when the simulation runs. As a result, more test points require a longer simulation time.

Test Point Types

There two types of Test Points used in CircuitMaker:

Test Point Type	**How Used**
Default Test Points	CircuitMaker automatically places Default Test Points in the circuit, according to the Analog Options settings. These are not displayed on the circuit.
Run-Time Test Points	These are test points you place in the circuit to graph data at a specific point. These test points can optionally be used to limit the amount of simulation data stored for the circuit.

Default Test Points

CircuitMaker automatically places Default Test Points in the circuit, allowing you to click with the Probe Tool on almost any wire, pin, or device to measure voltage, current, or power (respectively). The Default Test Points are set based on the **Collect Data For** setting in the Analog Options dialog box. The more data you choose to collect, the more test points (and memory) will be required. When you select the Run-Time Test Points Only option, the all other test points are disabled and data is only collected for the points that you define. Note that Default Test Points are not displayed on the circuit.

Run-Time Test Points

Run-Time Test Points are locations you can define in the circuit where CircuitMaker will automatically display graphical data that is collected during simulation. Run-Time Test Points can be used to measure voltage, current, or power dissipation. *Note that you can also plot waveforms by simply clicking with the Probe Tool after the simulation is complete*.

You can place Run-Time Test Points on wires to measure node voltages, on device pins to measure current, or on devices themselves to measure power dissipation. CircuitMaker displays waveforms only if it actually collects data by default or by the Test Points you define. CircuitMaker does not display waveforms for some devices, such as those containing subcircuits.

To place Run-Time Test Points in the circuit,

1 Stop the simulation.

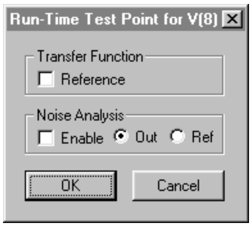
2 Select the **Probe Tool** from the Toolbar.

3 Click the appropriate locations in the circuit using the left mouse button.

To enable a Run-Time Test Point for Transfer Function or Noise Analysis,

1 Double-click it with the **Arrow Tool**.

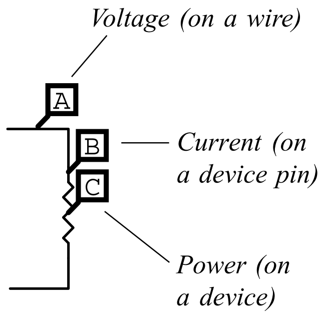
2 Place a checkmark in the appropriate checkbox. (These analysis types are discussed in more detail later in this chapter.)


Run-Time Test Point for V(8)

Transfer Function
☐ Reference

Noise Analysis
☐ Enable [✓ Out] Ref

Figure 6.2. Double-click on a Run-Time Test Point to enable it for Transfer Function or Noise Analysis.



*To add a Run-Time Test Point, **Left**-click the Probe Tool.*

Adding Multiple Run-Time Test Points

To add multiple Run-Time Test Points,

1 Hold down the **Shift** key while clicking with the Probe Tool.

Removing All Run-Time Test Points

To remove all Run-Time Test Points from the circuit,

1 **Left**-click with the **Probe Tool** in any blank area of the circuit window.

Limiting the Amount of Stored Data

If the circuit you are simulating is very large or complex, it may be necessary to limit the amount of data collected for analysis. Run-Time Test Points allow you to store specific node voltages, device currents or device power.

To limit the amount of stored data,

1 Place a Run-Time Test Point at each location where you need to see the data.

2 Click on the **Analyses Setup** button in the Toolbar, then click on the **Analog Options** button.

3 Select the last **Collect Data For** option, **Run-Time Test Points Only**.

Running the Simulation

Once you have created the circuit, you can run the simulation by simply clicking the Run button on the Toolbar. One aspect that sets CircuitMaker apart from other Spice-based simulators is the seamless integration between the schematic design and simulation process. There are no complex data fields to enter, and everything is done quickly and efficiently from the same workspace.

Using the Analysis Window

Run Button

CircuitMaker displays data and waveforms in the Analysis Window, which allows you to quickly and easily test, analyze and probe your circuits during simulation. Many attributes of the Analysis Window can be changed, to customize your view of the waveforms. There are also features within the Analysis Window that allow you to precisely measure the waveforms you are viewing.

When you run a simulation, CircuitMaker displays a separate tab in the Analysis Window for each of the following types of analyses that you enable:

- DC Sweep (curve tracer)
- AC Analysis (Bode plotter)
- Transient Analysis (oscilloscope)
- Fourier Analysis (spectrum analyzer)
- Operating Point (multimeter)
- Transfer Function
- Noise Analysis

For more information on these analyses, see *Setting Up Analog Analyses* later in this chapter. Figure 6.3 shows an example of the Analysis Window.

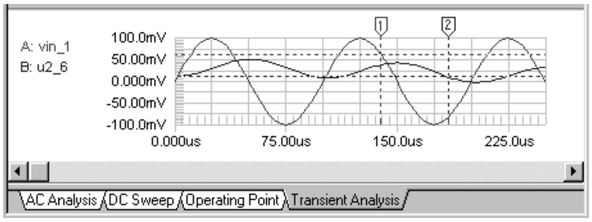


Figure 6.3. The Analysis Window.

Displaying Waveforms

Once you have run a simulation, you can quickly view data (waveforms) at any point(s) in the circuit using CircuitMaker's context-sensitive **Probe Tool**.

To plot data (waveforms) in the analysis window, first click on the appropriate tab, then:

1 Click any valid point in the circuit with the tip of the **Probe Tool**. See the *Probe Tool* section earlier in this chapter for information on some of the other types of data that can be plotted with the Probe Tool.

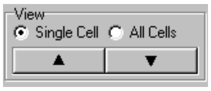
- Click on a **wire** to measure voltage
- Click on a **device pin** to measure current
- Click on a **device body** to measure power

If CircuitMaker indicates that current or power data is not available, click the **Analyses Setup** button in the Toolbar, then click the **Analog Options** button. Select the **Node Voltage, Supply Current, Device Current and Power** radio button in the lower right-hand corner of the dialog box and rerun the simulation.

Analyses Setup Button

2 Hold down the **Shift** key and click to plot multiple waveforms simultaneously in the analysis window. Note that clicking on another point in the circuit *without* holding the **Shift** key will replace any previous waveform(s) with a new one.

Single Cell vs. All Cells



The **Panel** includes a pair of radio buttons that allow you to select between Single Cell view (view one cell at a time) and All Cells view. Single Cell view displays the data in a view similar to an oscilloscope as shown in Figure 6.3. In this view, all waveforms are plotted on the same grid, but can be scaled independently. This view allows both manual and automatic scaling. If there is a waveform selected, then probing the circuit will insert the new waveform's label above the label of the selected waveform, otherwise the new waveform's label will be appended below the bottom label. If you switch to All Cells view, all the waveforms currently displayed will be placed into the same cell.

All Cells view displays multiple cells of data, each of which can contain one or more waveforms as shown in Figure 6.4. The y-axis on these cells is always scaled automatically.

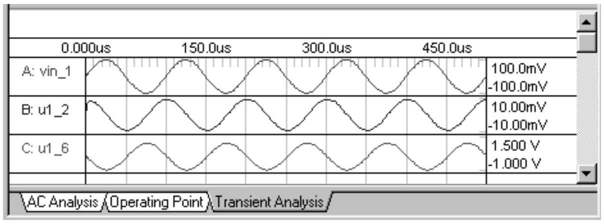


Figure 6.4. All Cells View.

In All Cells view, probing places new waveforms into the selected cell, or into the cell that has the selected waveform. Click on a wave label to select a waveform. Click on the cell body to select a cell. A selected cell will be highlighted with a bold rectangle around the left margin of that cell. If there is no waveform or cell selected, new waveforms go into a new cell below all existing cells.

You can drag and drop a wave label to move that waveform from one cell to another. If you drag and drop a wave label below the bottom cell, it will put that waveform into a new cell at the bottom. You can Right-click on a wave label to remove the waveform, change its color, and so on.

You can also Right-click on a cell body and select **Insert Cell** to insert a new cell above the selected cell. Waveforms from other cells can then be dragged and dropped into the new cell.

Cells can be resized by clicking and dragging the bottom boundary of the cell. To resize all cells, select **Wave** > **Preferences** and enter the desired cell height. If you want the specified cell height to become the new default value, check the **Save current settings as defaults** before you click OK.

If you switch to Single Cell view, only the waveforms in the selected cell will be displayed. If there are waveforms available in other cells, you can view the other cells by clicking the up or down arrows in the Panel. Probing the circuit without holding down the Shift key will delete all other waveforms, including those in other cells.

Plotting Subcircuit Internal Variables

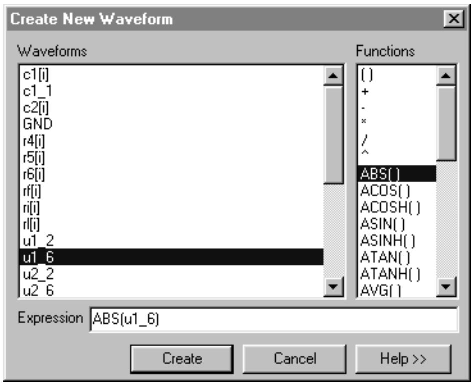
By default, CircuitMaker doesn't collect simulation data for any subcircuit's internal variables.

To plot a subcircuit's internal variables,

1. Click on the **Analyses Setup** button in the Toolbar.
2. Click **Analog Options**.
3. Click the radio button titled **Node Voltage, Supply Current and Subcircuit Variables** then choose OK.
4. Run the simulation.
5. Click the subcircuit device with the tip of the **Probe Tool** and select the variable from the list.

Waveform Mathematics

You can generate new waveforms from simulation data using a list of available math functions and operators. After running a simulation, select **Wave** > **Math**, then enter an expression in the space provided. You can either type in the expression, or build the expression by clicking on items in the available Waveforms and Functions lists. See Figure 6.5.



Expression ABS(u1_6)

Figure 6.5. Use this dialog box to create a new waveform based on a mathematical expression.

Listing of Functions and Operators

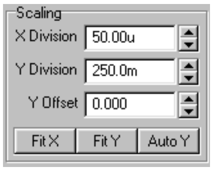
Expressions can be based on any available waveform and support the following operators and functions:

Operator/Function	**Description**
( )	Precedence Indicators. Use to set precedence of math operations. Operations contained within ( ) will be performed first.
+	Addition operator.
-	Subtraction operator.
*	Multiplication operator.
/	Division operator.
^	Power operator, y^x returns the value of "y raised to the power of x." Same as PWR(,).
ABS( )	Absolute value function. ABS(x) returns the value of |x|.
ACOS( )	Arc cosine function.
ACOSH( )	Hyperbolic arc cosine function.

ASIN( )	Arc sine function.
ASINH( )	Hyperbolic arc sine function.
ATAN( )	Arc tangent function.
ATANH( )	Hyperbolic arc tangent function.
AVG()	Average function. Returns the running average of the wave data.
BOOL(,)	Boolean function. In the expression BOOL(wave, thresh), wave would be the name of a waveform, thresh would be the switching threshold. Returns a value of one for wave arguments greater than or equal to thresh and a value of zero for wave arguments less than thresh.
COS( )	Cosine function.
COSH( )	Hyperbolic cosine function.
DER( )	Derivative function. dx/dt. Returns the slope between datapoints.
EXP( )	Exponential function. EXP(x) returns the value of "e raised to the power of x", where e is the base of the natural logarithms.
INT( )	Integral function. Returns the running total of the area under the curve.
LN( )	Natural logarithm function. Where LN(e) = 1.
LOG10( )	Log base 10 function.
LOG2( )	Log Base 2 function.
PWR(,)	Power function. Same as ^ operator. PWR(y,x) returns the value of "y raised to the power of x."
RMS( )	Root-Mean-Square function. Returns the running AC RMS value of the wave data.

SIN( )	Sine function.
SINH( )	Hyperbolic sine function.
SQRT( )	Square root function.
TAN( )	Tangent function.
TANH( )	Hyperbolic tangent function.
UNARY( )	Unary minus function. UNARY(x) returns -x.
URAMP( )	Unit ramp function. Integral of the unit step: for an input x, the value is zero if x is less than zero, or if x is greater than zero, the value is x.
USTEP( )	Unit step function. Returns a value of one for arguments greater than zero and a value of zero for arguments less than zero.

Scaling and Offsetting Waveforms



X Division: 50.00u
Y Division: 250.0m
Y Offset: 0.000

Scaling Controls

In Single Cell view mode, waveforms can be scaled either manually or automatically. The x- and y-axis scaling controls are located on the Panel. Manual scaling operations for the y-axis apply only to the selected waveform. But if there is no waveform selected, then y-axis scaling operations apply to all waveforms. You can select a waveform by clicking on its wave label that is displayed on the left side of the graph. On the graph, the displayed y-axis pertains only to the selected waveform. If there is no waveform selected, and the y-axis is common to all waveforms, then the common y-axis is displayed. If there is not a common axis, the y-axis values are not displayed, but the division size is listed for each waveform, and the zero level for each waveform is marked along the right edge of the graph. The x-axis is always common to all waveforms.

When probing, if you want to have CircuitMaker automatically scale the y-axis, then click the **Auto Y** button on the Panel. Any time you add or remove waveforms, the y-axis will be rescaled as needed so that all waveforms share a common y-scale, and all waveforms fit on the graph.

Performing manual scaling operations such as zooming will turn off the Auto Y mode.

When Auto Y scaling is off, you can click the **Fit Y** button to scale the y-axis. If there is a waveform selected, then the Fit Y operation applies only to the selected waveform. The selected waveform will be rescaled to be as large as possible while still fitting on the grid. If there is no waveform selected, then a Fit Y operation causes the y-axis to be rescaled so that all the waveforms share a common y-scale, and all waveforms are visible. In this regard, Fit Y is like Auto Y, but it is a single operation and not a mode.

To auto scale all waveforms on a common y-axis,

1 Click the **Auto Y** button in the Panel.

To manually scale or offset an individual waveform,

1 Select the waveform by clicking on its wave label to the left of the graph.

2 Adjust the **X Division**, **Y Division** and **Y Offset** using the scaling controls in the Panel.

To manually scale or offset all waveforms on a common y-axis,

1 Make sure no waveforms are selected.

2 Adjust the **X Division**, **Y Division** and **Y Offset** using the scaling controls in the Panel.

To auto scale both the x and y axes,

1 Select **Wave > Fit Waveforms**.

OR

Right-click in the analysis window and select **Fit Waveforms** from the pop-up menu.

Zooming and Panning

There are various ways to improve your view of the waveforms. The analysis window can be viewed as a full window, or it can be split with the schematic window, either horizontally or vertically. There are buttons on the Toolbar to select one of these views. You can use the splitter bars between the Panel, the Schematic Window and the Analysis Window to resize the respective areas.

To get a closer view, you can click and drag a selection rectangle around a portion of the graph, then release the mouse to zoom in on that portion of the analysis window. See Figure 6.6. You can also Right-click on the graph and select **Zoom In** or **Zoom Out** from the pop-up menu. If zoomed in on a graph, you can use the vertical and horizontal scroll bars to pan around the graph.

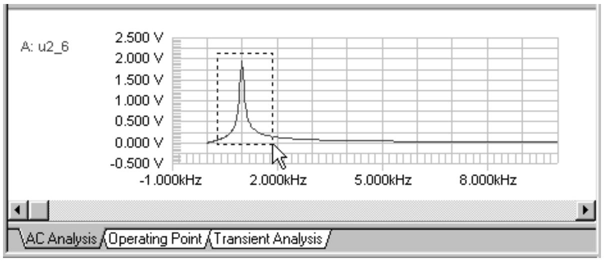


Figure 6.6. Click and drag a rectangle around any portion of the window to zoom in on that area.

AC Analysis Scaling Options

For AC analysis you can adjust the x-axis to be displayed on either a Linear or Log scale. You can also choose from a variety of y-axes and even display data on two different y-axes, such as magnitude and phase, simultaneously. See Figure 6.7.

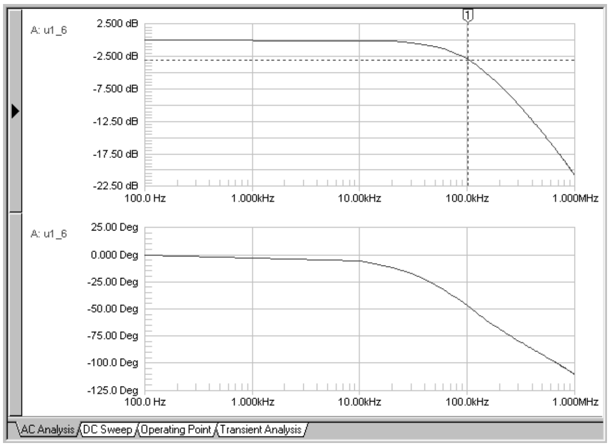


Figure 6.7. Dual axis grid in AC Analysis displays data simultaneously on two different y-axes.

Even though two y-axes are displayed simultaneously, only one at a time can be active. The black arrow to the left of the graph points to the active axis. Measurement cursor and scaling operations such as changing the y offset or y division size apply to the active axis. Click on the top or bottom graph to activate either the top or bottom y-axis. X-axis operations always apply to both graphs, since the x-axis is always common. Available y-axes include: Real, Imaginary, Magnitude, Magnitude in Decibels, Phase in Degrees, Phase in Radians and Group Delay.

To enable the dual-axis mode,

1 Select the **AC Analysis** tab in the Analysis Window.

2 Select **Single Cell** view in the Panel.

3 Select **Wave** > **Scaling**.

OR

Right-click on the graph and select **Scaling** from the pop-up menu.

4 Set the **Primary** (top grid) and **Secondary** (bottom grid) y-axes as desired (select **None** to disable the secondary grid.) Click OK.

Restoring Wave Setups

After running a simulation, probing waves, and configuring the graph exactly as you want it, you may want to edit the schematic and rerun the simulation. If you want the previous wave setup restored you must enable the **Keep last setup** option. Otherwise, enable the **Show run-time test points** option.

To choose which wave setup is used when you rerun a simulation,

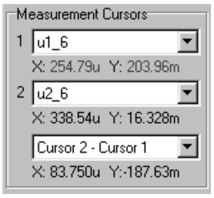
1 Click on the **Analyses Setup** button in the Toolbar.

2 Select the **Keep last setup** radio button to keep the waveform views the way you have set them up.

OR

Select the **Show run-time test points** radio button to return to the run-time test point waveforms.

Tracking Measurement Cursors


Measurement Cursors
1 u1_6
X: 254.79u Y: 203.96m
2 u2_6
X: 338.54u Y: 16.328m
Cursor 2 - Cursor 1
X: 83.750u Y:-187.63m

The cursor measurements are displayed in the Panel

Tip: *To nudge a cursor, click on it once to select it, then press the left or right arrow keys on the keyboard. For a fine nudge, hold down the Shift key while nudging with the arrow keys.*

Two measurement cursors let you precisely measure values in the waveform analysis windows such as amplitude and period. Each cursor will track its assigned waveform. To assign a cursor, select a waveform in the cursor's drop-down list or Right-click on a wave label in the graph.

Each enabled cursor has a tab at the top of the graph. Click and drag the cursor tab to move the cursor along the x-axis. A horizontal line will automatically track the waveform along the y-axis. The cursor's x and y values are displayed in the Panel.

If both cursors are enabled, **Cursor 2 - Cursor 1** values (x and y cursor deltas) can be displayed. If both the cursors are assigned to the same waveform, then the **Minimum** or **Maximum** y value between the cursors is displayed. Other options for a single waveform include Average, RMS and

Frequency. **Average** and **RMS** values are based on the average of the data between the cursors, so for an accurate measurement, it is important to measure complete cycles or a large number of cycles. Also, be sure that the waveform you are measuring has stabilized between the cursors (i.e., it is not drifting toward or away from some DC offset.) See Figures 6.8 and 6.9. **Frequency** is measured by placing the cursors exactly one cycle apart (Frequency = 1/period.)

Avg: 12.656 V

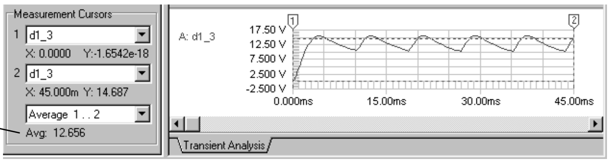


Figure 6.8. These measurement cursor positions will not give an accurate Average reading. They include an initial charge from 0V and also include about 5-1/2 cycles.

Avg: 13.016 V

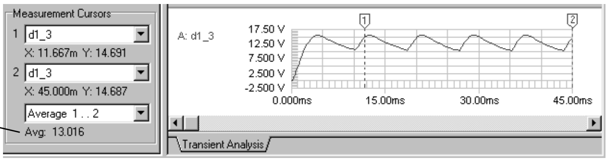


Figure 6.9. These measurement cursor positions will give a more accurate Average reading. They do not include the initial charge and they include exactly 4 cycles. Accuracy can be increased even more by increasing the resolution and the number of cycles.

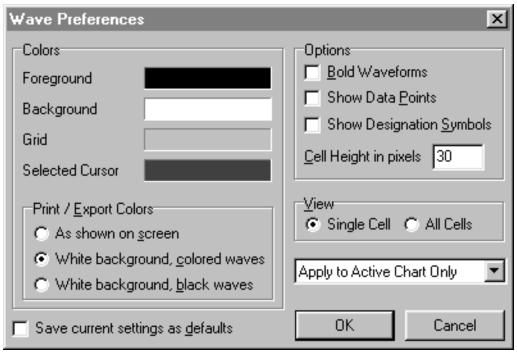
Wave Preferences

The Wave Preferences dialog box is used to control the global and default aspects of the wave charts. Selections can either apply to the Active Chart (the analysis currently showing,) or to the entire document (all analyses.) You can also choose to save the current settings as the default settings for future charts. See Figure 6.10.

To use the Wave Preferences dialog box,

1 Select **Wave** > **Preferences**.

2 Set the options as desired and click OK.


Wave Preferences

Colors

Foreground	⬛
Background	⬜
Grid	▨
Selected Cursor	⬛

Print / Export Colors
- ○ As shown on screen
- ● White background, colored waves
- ○ White background, black waves

Options
- ☐ Bold Waveforms
- ☐ Show Data Points
- ☐ Show Designation Symbols

Cell Height in pixels: 30

View: ● Single Cell ○ All Cells

Apply to Active Chart Only

☐ Save current settings as defaults

Figure 6.10. Wave Preferences

Storing and Recalling Waveforms

Waveforms can be stored for future reference or to compare with other waveforms after the circuit has been altered. The data is stored in a tab delineated ASCII text file in its raw form which may contain both real and imaginary values. Depending on the analysis type, the data might not be linearized (spacing of the datapoints along the x-axis might not be linear).

To store a waveform,

1 Click on the waveform's label to select the wave.

2 Select **Wave** > **Store**.

3 Specify the name of the file in which to save the waveform. Each waveform is stored in a separate file. Each analysis type uses a different file extension.

To recall a stored waveform and place it on the graph,

1 Select **Wave** > **Recall**.

2 Select the appropriate waveform file from the File Selection dialog box and choose **Open**.

3 To remove a stored waveform from the graph, Right-click on its wave label and select **Remove Wave** from the pop-up menu.

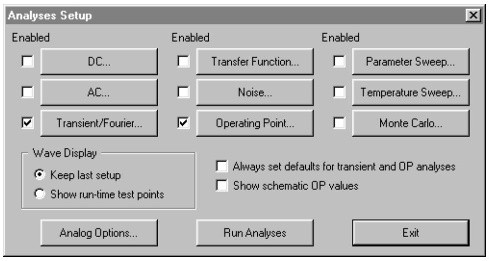
Saving a Waveform as Text

Waveforms can be also be exported for use in other applications. The Save As Text command writes the waveform, exactly as it appears on the graph, to a tab delineated ASCII text file. Depending on the analysis type, the data might not be linearized (spacing of the datapoints along the x-axis might not be linear).

Setting Up Analog Analyses

Analyses Setup Button

CircuitMaker's powerful simulation capability includes a wide variety of standard and advanced analyses. Click the **Analyses Setup** button on the Toolbar to display the Analyses Setup dialog box pictured in Figure 6.11.



[ ] DC... [ ] Transfer Function... [ ] Parameter Sweep...
[ ] AC... [ ] Noise... [ ] Temperature Sweep...
[✓] Transient/Fourier... [✓] Operating Point... [ ] Monte Carlo...

Wave Display: [● Keep last setup] ○ Show run-time test points

[ ] Always set defaults for transient and OP analyses
[ ] Show schematic OP values

Figure 6.11. Use the Analyses Setup dialog box to setup the various analyses.

CircuitMaker offers the following standard and advanced analyses:

- DC Sweep
- AC Analysis
- DC Operating Point
- Transient Analysis
- Parameter Sweep
- Fourier Analysis

- Transfer Function (CircuitMaker PRO only)
- Noise (CircuitMaker PRO only)
- Temperature Sweep (CircuitMaker PRO only)
- Monte Carlo (CircuitMaker PRO only)
- Impedance Plots (CircuitMaker PRO only)

Always Set Defaults

You can use the **Always set defaults for transient and OP analyses** checkbox to simplify the task of setting up the Transient Analyses. For simple DC circuits (those that contain no signal generators or active components,) only Operating Point Analysis is enabled. For more complex circuits, Transient Analysis is also enabled and the default start, stop and step settings are used.

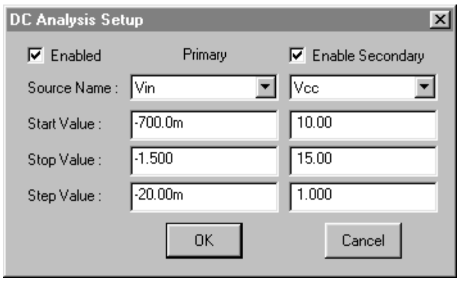
DC Sweep

The DC Analysis generates output like that of a curve tracer. It performs an Operating Point Analysis at each of a series of steps defining a DC transfer curve. Source Name is the name of an independent power source (either a fixed voltage or current supply or a Signal Generator) that is to be stepped in the circuit. The start, stop and step values define the sweep range and resolution. The primary source is required while the secondary source is optional. If a secondary source is specified, the primary source is stepped over its entire range for each value of the secondary source.

To set up and run a DC Sweep,

1 Click on the **Analyses Setup** button in the Toolbar.

2 Click the **DC...** button to display the dialog box pictured in Figure 6.12.



DC Analysis Setup

	Primary	Secondary
	[✓ Enabled]	[✓ Enable Secondary]
Source Name :	Vin	Vcc
Start Value :	-700.0m	10.00
Stop Value :	-1.500	15.00
Step Value :	-20.00m	1.000

Figure 6.12. The DC Sweep Setup dialog box.

3 Specify the settings you want, select the Enabled check box, and then choose OK.

4 Run the simulation.

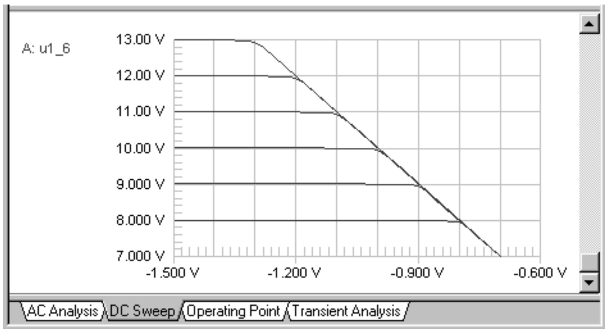


Figure 6.13. A DC Sweep waveform.

The waveform in Figure 6.13 was generated by simulating the Analog.ckt circuit using the values shown in Figure 6.12. Notice that the x axis represents the voltage of the primary source, and the y axis is the voltage at the output of the circuit. The waveforms correspond with each step of the secondary source. See *Using the Analysis Window* earlier in this chapter for more information about manipulating the waveforms.

AC Analysis (Frequency Sweep)

AC Analysis generates output like that of a Bode plotter, computing the small-signal AC output variables as a function of frequency. It performs an Operating Point Analysis to determine the DC bias of the circuit, replaces the signal source with a fixed amplitude sine wave generator, and analyzes the circuit over the frequency range you specify. The desired output of an AC small-signal analysis is usually a transfer function (voltage gain, transimpedance, etc.).

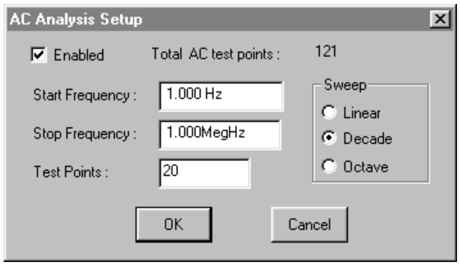
To set up and run an AC Analysis,

1 Connect at least one Signal Generator to the circuit and enable it as an AC Analysis source.

Do this by double-clicking the Signal Generator, clicking **Wave**, and setting up the AC Analysis Source options.

2 Click on the **Analyses Setup** button in the Toolbar.

3 Click the **AC** button to display the dialog box pictured in Figure 6.14.


AC Analysis Setup

[✓] Enabled Total AC test points : 121

Start Frequency : 1.000 Hz

Stop Frequency : 1.000MegHz

Test Points : 20

Sweep: Linear [● Decade] Octave

Figure 6.14. Use this dialog box to set up an AC Analysis (frequency sweep).

4 Enter the AC Analysis settings you want (see the following table), select the Enabled check box, and then choose OK.

Sweep Option	**What it Means**
Linear	Total number of data points evenly spaced on a linear scale.
Decade	Number of evenly spaced data points per decade of a $\log_{10}$ scale.
Octave	Number of evenly spaced data points per octave of a $\log_2$ scale.

5 Run the simulation.

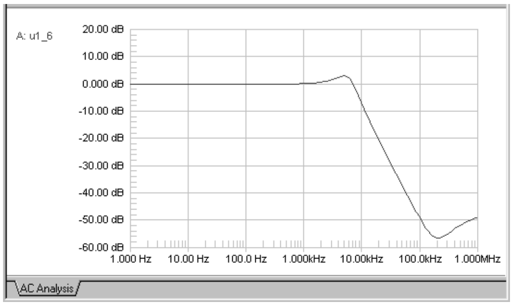


Figure 6.15. An AC Analysis waveform.

The waveform in Figure 6.15 was generated by simulating the Analog.ckt circuit using the values shown in Figure 6.10. Notice that the x axis represents frequency in a log scale. The waveform represents the magnitude of the circuit's output voltage in decibels (dB). See *Using the Analysis Window* earlier in this chapter for more information about manipulating the waveforms.

Note: You can plot voltage, source current and source impedance in AC Analysis, but you cannot plot device current or power.

Operating Point Analysis

Operating Point Analysis generates data similar to the readings of a DC multimeter. It determines the DC bias of the entire circuit with inductors shorted and capacitors opened, and determines linearized, small-signal models for all of the nonlinear devices in the circuit. It does not take into account the existence of any AC source.

Operating Point Analysis is generally performed automatically before each of the other analyses, even if it has been disabled in the Analog Analyses dialog box. However, you must enable it if you want to use the Probe Tool as a multimeter to view the DC, DC AVG or AC RMS values of current, voltage or power. (Viewing the DC AVG or AC RMS values also requires that you enable Transient Analysis). See *Probe Tool* earlier in this chapter.

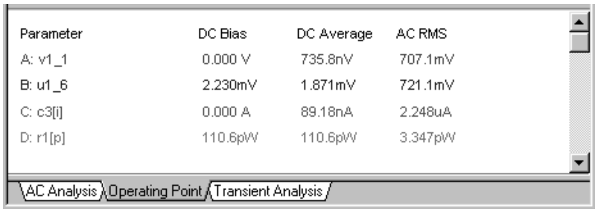
To set up and run Operating Point Analysis,

1 Click on the **Analyses Setup** button in the Toolbar.

2 Select the **Enabled** checkbox next to the Operating Point button and choose OK.

3 Run the simulation.

4 Click the **Probe Tool** on the point of the circuit where you want to analyze the DC operating point (see Figure 6.16).



Parameter	DC Bias	DC Average	AC RMS
A: v1_1	0.000 V	735.8nV	707.1mV
B: u1_6	2.230mV	1.871mV	721.1mV
C: c3[i]	0.000 A	89.18nA	2.248uA
D: r1[p]	110.6pW	110.6pW	3.347pW

Figure 6.16. The Operating Point chart showing DC Operating Point (DC Bias), DC Average and AC RMS values.

Transient Analysis

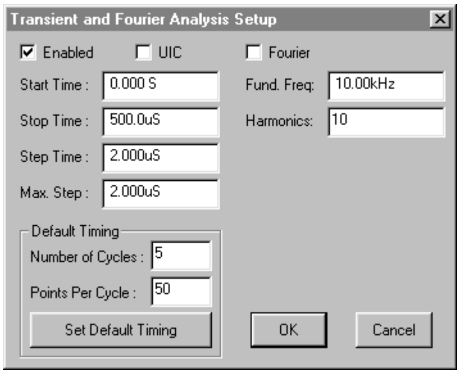
A Transient Analysis generates output like that of an oscilloscope, computing the transient output variables (voltage or current) as a function of time over the user-specified time interval.

A Transient Analysis first performs an Operating Point Analysis to determine the DC bias of the circuit, always beginning at time zero. In the time interval between zero and **Start Time**, XSpice analyzes but does not display the circuit. In the time interval between **Start Time** and **Stop Time**, XSpice both analyzes and displays the circuit. **Step Time** is the suggested computing increment, but the timestep will be varied automatically by XSpice in order to properly converge. **Max. Step** limits the varying size of the timestep that XSpice can use when calculating the transient data; by default, the program chooses either **Step Time** or (**Stop Time** - **Start Time**)/50, whichever is smaller. Typically, **Step Time** and **Max. Step** would be the same value. If you enable the **UIC** (Use Initial Conditions) option, the Transient Analysis begins from an initial condition, bypassing the Operating Point Analysis. This is useful for viewing the charging of capacitors, etc.

To set up and run a Transient Analysis,

1 Click on the **Analyses Setup** button in the Toolbar.

2 Click the **Transient/Fourier** button to display the dialog box pictured in Figure 6.17.


Transient and Fourier Analysis Setup

[✓] Enabled [ ] UIC [ ] Fourier

Start Time : 0.000 S
Stop Time : 500.0uS
Step Time : 2.000uS
Max. Step : 2.000uS
Fund. Freq: 10.00kHz
Harmonics: 10

Default Timing
Number of Cycles : 5
Points Per Cycle : 50

Figure 6.17. Use this dialog box to set up Transient and Fourier Analyses.

3 Enter the Start, Stop and Step values and choose OK.

OR

Enter the **Number of Cycles** and **Points Per Cycle**, click the **Set Default Timing** button, then choose OK. **Note:** this option can only be used if there is a Signal Generator or Data Sequencer in the circuit. If the Always Set Defaults checkbox is checked in the Analysis Setup dialog box, CircuitMaker acts as if this button is pressed before each simulation.

4 Run the simulation.

5 Assuming you have enabled the appropriate Test Points, you can view and measure voltage, current and power dissipation waveforms of the circuit in the analysis window that is displayed.

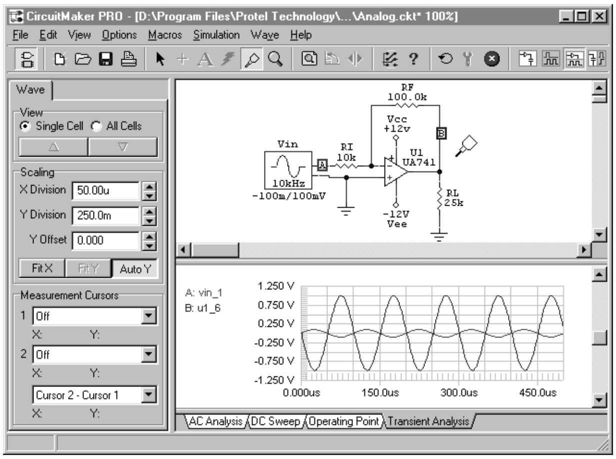


Figure 6.18. The circuit Analog.ckt, and the Transient Analysis window.

The waveform in Figure 6.18 was generated by simulating the Analog.ckt circuit using the values shown in Figure 6.17. This analysis shows the voltage at the test points during the time frame specified in the **Start** and **Stop** fields. Notice that the x axis shows the time in seconds, and the y axis is voltage. The larger waveform represents the voltage at the circuit output, while the smaller waveform is voltage at the source (signal generator) or input. See *Using the Analysis Window* earlier in this chapter for more information about manipulating the waveforms.

Parameter Sweep

Use a Parameter Sweep for two variables including transistor parameters. You can vary only basic components and models; subcircuit data is not varied during the analysis.

You can use the Parameter Sweep feature only when you have enabled one or more of the standard analyses (AC, DC Sweep, Operating Point, Transient, Transfer Function, Noise). The parameter sweep requires the following data: **Parameter**, **Start Value**, **Stop Value**, and **Step Value**. The parameter can be a single designation or a designation with a device parameter in brackets. The following are valid examples:

Example	What it Varies
RF	Resistor with designation RF
Q3[bf]	Beta forward on transistor Q3
R3[r]	Resistance of potentiometer R3
R3[position]	Position of potentiometer R3
option[temp]	Temperature (**CircuitMaker PRO only**)
U5[tp_val]	Propagation delays of digital device U5

Normally you would use a Temperature Sweep to vary the temperature for simulation; however, temperature can also be varied in the Parameter Sweep (useful if you want to vary the temperature as either the primary or secondary parameter in a two-parameter sweep).

To set up and run a Parameter Sweep Analysis,

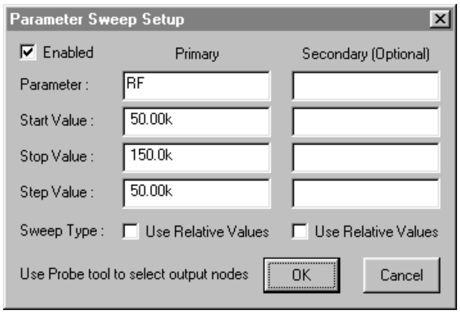
1 Click on the **Analyses Setup** button in the Toolbar.

2 Click the **Parameter Sweep** button to display the dialog box pictured in Figure 6.19.

3 Make the desired settings, select **Enabled**, and then choose OK.

4 Run the simulation.

5 View the resulting waveforms in the analysis window, such as the one in Figure 6.20.



Parameter Sweep Setup

☑ Enabled

	Primary	Secondary (Optional)
Parameter :	RF	
Start Value :	50.00k	
Stop Value :	150.0k	
Step Value :	50.00k	
Sweep Type :	☐ Use Relative Values	☐ Use Relative Values

Use Probe tool to select output nodes

Figure 6.19. Use this dialog box to set up a Parameter Sweep Analysis.

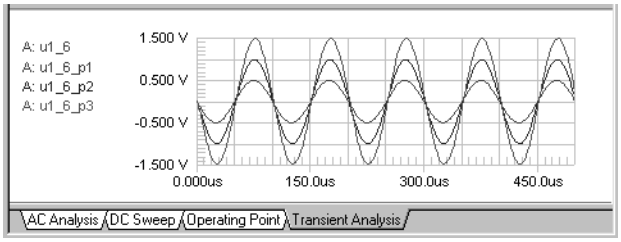


Figure 6.20. Parameter Sweep sweeps a selected component (in this example, a resistor) over a defined range in defined steps and plots the output voltage of the circuit at each of those steps.

The waveform in Figure 6.20 was generated by simulating the Analog.ckt circuit using the values shown in Figure 6.19. Notice that it also plots the input and output voltages for the nominal run (meaning all values as they are shown in the schematic).

Relative Values Option

If you enable the **Use Relative Values** option on the Parameter Sweep Setup dialog box, the values entered in the **Start Value**, **Stop Value**, and **Step Value** fields are added to the parameter's default value. For example, suppose you do a Parameter Sweep with the following conditions:

- The parameter is a 1k ohm resistor.
- The **Start**, **Stop**, and **Step** fields are –50, 50, and 20, respectively.
- You enable **Use Relative Values**.

The following resistor values would be used in the simulation runs: 950, 970, 990, 1010, 1030, and 1050.

Sweep Trace Labels

For the Monte Carlo, Temperature Sweep, and Parameter Sweep analyses, CircuitMaker gives a unique label to each trace or waveform generated. CircuitMaker displays the trace names from each sweep run in the analysis window, with a special character appended to the trace name. For example, the appended characters are **m** for Monte Carlo, **t** for Temperature Sweep, and **p** for Parameter Sweep. There is also a trace for the simulation run which was done with the nominal circuit values. This trace will not have a character appended to its trace label. The following are some example trace labels with explanations:

u1_6_p1	Voltage at node u1_6 for Parameter Sweep run 1
u1_6_p2	Voltage at node u1_6 for Parameter Sweep run 2
u1_6_p3	Voltage at node u1_6 for Parameter Sweep run 3
u1_6	Voltage at node u1_6 for nominal run
u1_6_m1	Voltage at node u1_6 for Monte Carlo run 1
u1_6_t1	Voltage at node u1_6 for Temperature Sweep run 1

You can Right-click on a trace label and select **Sweep Data** to display details about the device values used in that simulation run.

Fourier Analysis

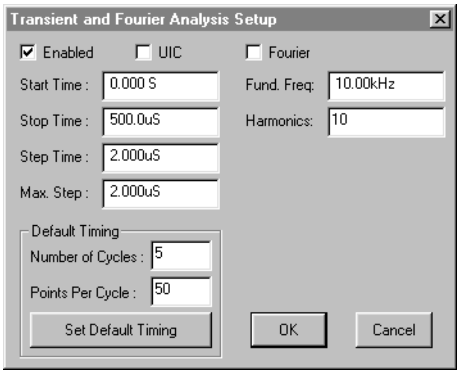
Fourier Analysis generates output like that of an spectrum analyzer, computing the frequency components of the output variables.

Fourier Analysis setup is included with the Transient Analysis setup. You must enable the Transient Analysis in order to do the Fourier Analysis. When the simulation is completed, the Fourier Analysis is displayed in a separate window. The Fourier Analysis is based on the last complete cycle of transient data. For example, if the fundamental frequency is 1.0kHz, then the transient data from the last 1ms cycle would be used for the Fourier analysis.

To set up and run a Fourier Analysis,

1 Click on the **Analyses Setup** button in the Toolbar.

2 Click the **Transient/Fourier** button to display the dialog box pictured in Figure 6.21.


Transient and Fourier Analysis Setup

☑ Enabled ☐ UIC ☐ Fourier

Start Time : 0.000 S
Stop Time : 500.0uS
Step Time : 2.000uS
Max. Step : 2.000uS
Fund. Freq: 10.00kHz
Harmonics: 10

Default Timing
Number of Cycles : 5
Points Per Cycle : 50

Figure 6.21. Use this dialog box to set up a Fourier Analysis.

3 Enter the analysis settings and choose OK. **Note:** The fundamental frequency is the size of the frequency step and the number of harmonics is the number of steps.

4 Run the simulation.

5 Assuming you have enabled the appropriate Test Points, view and measure voltage, current and power dissipation waveforms of the circuit in the analysis window that CircuitMaker displays.

The waveform in Figure 6.22 was generated by simulating the Bandpass.ckt circuit using the values shown in Figure 6.21. This analysis shows the frequency spectrum of the square wave from the signal generator. The first peak in the waveform is the amplitude of the component at the fundamental frequency. Amplitudes at various harmonics within the specified range are also shown. See *Using the Analysis Window* earlier in this chapter for more information about manipulating the waveforms.

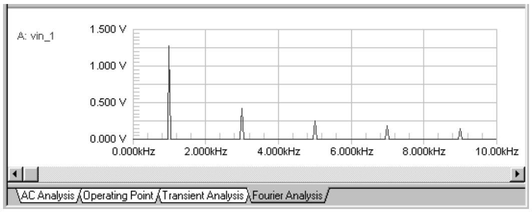


Figure 6.22. A Fourier Analysis waveform.

Professional Edition

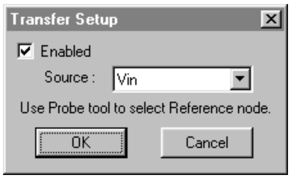
Transfer Function Analysis

The Transfer Function analysis calculates the DC input resistance, DC output resistance, and DC gain.

To set up and run a Transfer Function Analysis,

1 Click on the **Analyses Setup** button in the Toolbar.

2 Click the **Transfer** button to display the dialog box pictured in Figure 6.23.



Transfer Setup

☑ Enabled

Source : Vin

Use Probe tool to select Reference node.

Figure 6.23. Use this dialog box to set up a Transfer Function Analysis.

3 Select the input source you want to consider from the **Source** drop-down list.

4 Select the **Enabled** check box and choose OK.

Note: You do not need to specify the output node ahead of time. However, to specify a reference other than ground, place a Run-Time Test Point, double-click on it and check the Transfer Function **Reference** checkbox before you run the simulation.

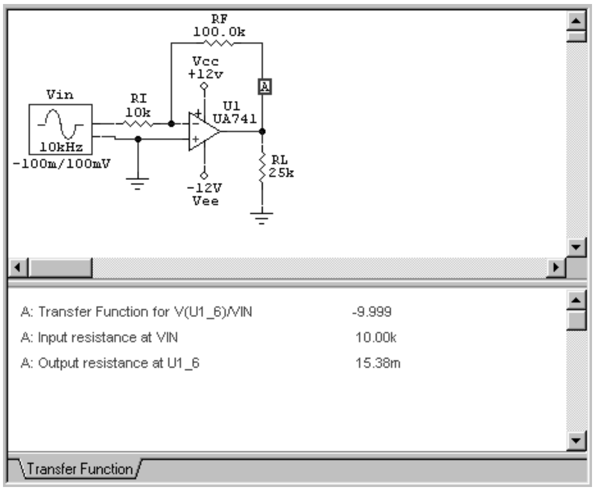
Professional Edition

5 Run the simulation.

6 Click the **Transfer Function** window, and then click on any node in the circuit to see the Transfer Function data for that node.

7 To display the Transfer Function data for multiple nodes, hold down the Shift key when you click to display.

The data that appears indicates the transfer function from the input to the specified node.



```
A: Transfer Function for V(U1_6)/VIN          -9.999
A: Input resistance at VIN                    10.00k
A: Output resistance at U1_6                  15.38m
```

Figure 6.24. Transfer Function Analysis data.

The waveform data in Figure 6.24 was generated by simulating the Analog.ckt circuit using the values shown in Figure 6.23. This analysis shows the gain from the specified source to the output (the point you click with the Probe Tool). Notice that it shows DC resistance seen by the source and the DC resistance seen by the output load.

Professional Edition

Noise Analysis

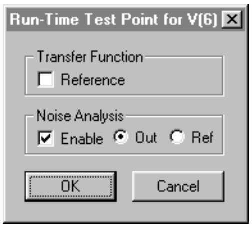
Noise Analysis lets you measure the noise in your circuit due to noise contributions of resistors and semiconductor devices. CircuitMaker can plot the Noise Spectral Density, which is the noise measured in Volts squared per Hertz (V^2/Hz). Capacitors, inductors, and controlled sources are treated as noise free. The following noise measurements can be made in CircuitMaker:

Measurement	**Description**
Output Noise	The noise measured at a specified output node.
Input Noise	The amount of noise that, if injected at the input, would cause the calculated noise at the output. For example, if the output noise is 10p, and the circuit has a gain of 10, then it would take 1p of noise at the input to measure 10p of noise at the output. Thus the equivalent input noise is 1p.
Component Noise	The output noise contribution of each component in the circuit. The total output noise is the sum of individual noise contributions of resistors and semiconductor devices. Each of these components contributes a certain amount of noise, which is multiplied by the gain from that component's position to the circuit's output. Thus the same component can contribute different amounts of noise to the output, depending on its location in the circuit.

Professional Edition

To set up and run a Noise Analysis,

1 Select the output node by clicking the output node of the circuit with the **Probe Tool** to place a Run-Time Test Point. Double-click on it to display the dialog box shown in Figure 6.25 In the dialog box that appears, check the Noise Analysis **Enable** checkbox, and make sure **Out** is selected.

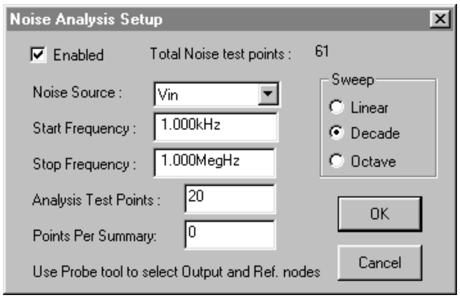


[✓ Enable] [✓ Out] Ref

Figure 6.25. Use this dialog box to set up Run-Time Test Points for a Noise Analysis.

2 Click on the **Analyses Setup** button in the Toolbar.

3 Click the **Noise** button to display the dialog box pictured in Figure 6.26.



[✓ Enabled] Total Noise test points : 61

Noise Source : Vin

Start Frequency : 1.000kHz

Stop Frequency : 1.000MegHz

Analysis Test Points : 20

Points Per Summary: 0

Sweep: Linear [✓ Decade] Octave

Use Probe tool to select Output and Ref. nodes

Figure 6.26. Use this dialog box to set up Noise Analysis.

4 Specify the analysis information in the Noise Analysis dialog box.

If you want to measure the noise contribution of each component, enter 1 in the **Points Per Summary** field. If you only want to measure input and output noise, enter 0 in this field.

Professional Edition

5 Run the simulation.

6 View the input and output noise results.

7 Click the Noise analysis window, then click on the noise output node.

The input noise waveform is labeled **NI**, and the output noise waveform is **NO**. For example, if the specified noise output node is node **u1_6**, then the output noise waveform would be labeled **NO(u1_6)**, and the input noise waveform would be labeled **NI(u1_6)**.

8 Click a component to measure the output noise contribution of that component. If the component you click on is a subcircuit, then you may select from a list of contributing noise sources within the subcircuit. Note that component noise data is *not* available if 0 (zero) was entered in the Points Per Summary field in the Noise Analysis Setup dialog box.

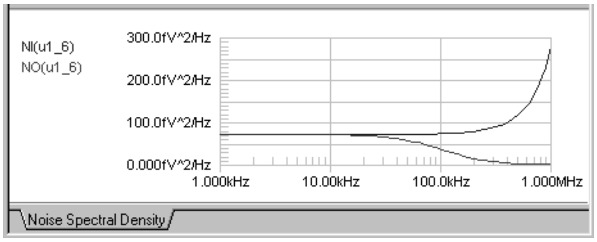


Figure 6.27. This Noise Analysis waveform shows noise at the output and the equivalent input noise at the specified source over a specified frequency range.

The waveform in Figure 6.27 was generated by simulating the Analog.ckt circuit using the values shown in Figures 6.25 and 6.26. See *Using the Analysis Window* earlier in this chapter for more information about manipulating the waveforms.

Professional Edition

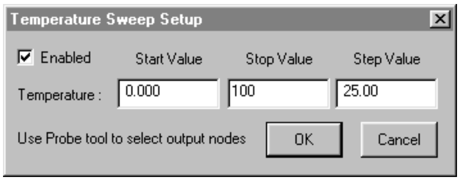
Temperature Sweep

You can use a Temperature Sweep only when you have enabled one or more of the standard analyses (AC, DC Sweep, Operating Point, Transient, Transfer Function, Noise).

To set up and run a Temperature Sweep Analysis,

1 Click on the **Analyses Setup** button in the Toolbar.

2 Click **Temperature Sweep** to display the dialog box shown in Figure 6.28.



Temperature Sweep Setup

☑ Enabled

	Start Value	Stop Value	Step Value
Temperature :	0.000	100	25.00

Use Probe tool to select output nodes

Figure 6.28. Use this dialog box to set up a Temperature Sweep Analysis.

3 Enter the temperature range you want to sweep.

4 Select the **Enabled** checkbox.

5 Set up one or more standard analysis so that each analysis is performed at the indicated temperatures.

CircuitMaker displays the results of a Temperature Sweep in the Analysis Window.

The waveform in Figure 6.29 was generated by simulating the Analog.ckt circuit using the values shown in Figure 6.28. Notice that the analysis sweeps the temperature of the circuit in specified steps, between the start and stop temperature values. The waveforms in this example represent the magnitude of the circuit output voltage (in db) for each temperature step. See *Using the Analysis Window* earlier in this chapter for more information about manipulating the waveforms.

Professional Edition

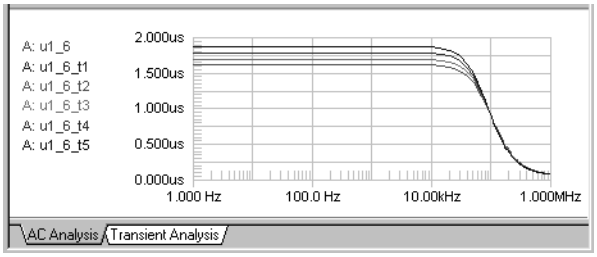


Figure 6.29. Temperature Sweep Analysis waveforms.

Monte Carlo Analysis

Monte Carlo Analysis lets you do multiple simulation runs with device values randomly varied according to specified tolerances. You can use this feature only when you have enabled one or more of the standard analyses (AC, DC Sweep, Operating Point, Transient, Transfer Function, Noise). Furthermore, CircuitMaker saves data only for nodes that have a Run-Time Test Point. Subcircuit data is not varied during the Monte Carlo Analysis. Only basic components and models can be varied.

The items in the Monte Carlo Setup dialog box are as follows:

Option	**Description**
Simulation Runs	Enter the number of simulation runs you want CircuitMaker to perform. For example, if you enter 10, then CircuitMaker will run 10 simulation runs, with different device values on each run, within the specified tolerances.
Seed	CircuitMaker uses the specified seed to generate random numbers for the Monte Carlo runs. The default seed value is –1. If you want to run a simulation with a different series of random numbers, then you must change the seed value to another number.

Professional Edition

Distribution	You can choose from the following three distributions for random number generation in the Monte Carlo Analysis:

Uniform distribution
Values are uniformly distributed over the specified tolerance range. Suppose you have a 1K resistor with a tolerance of 10 percent. There is an equal chance of the generated value being anywhere from 900 ohms to 1100 ohms. It is a flat distribution.

Gaussian distribution
Values are distributed according to a gaussian (bell-shaped) curve with the center at the nominal value and the specified tolerance at +/- 3 standard deviations. Given a 1K, 10 percent resistor, the center of the distribution would be at 1000 ohms, + 3 standard deviations at 1100 ohms, and –3 standard deviations at 990 ohms.

Worst Case distribution
This is the same as the uniform distribution, but only the end points (worst case) of the range are used. Given a 1K, 10 percent resistor, the value used would be randomly chosen from the two worst case values of 990 ohms and 1100 ohms. On any one simulation run, there is an equal chance that the high-end worst case value (1100) or low-end worst case value (990) will be used.

To set up and run a Monte Carlo Analysis,

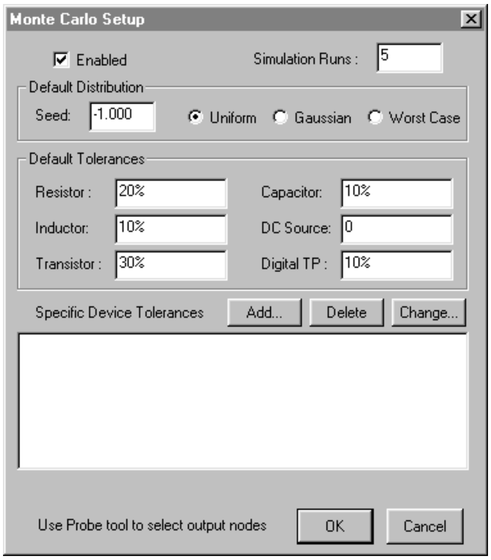
1 Click on the **Analyses Setup** button in the Toolbar.

2 Click the **Monte Carlo** button to display the dialog box pictured in Figure 6.30.

3 Enter tolerances as actual values or as percentages for the six general categories of devices then choose OK (see *Specifying Default Tolerances* later in this section).

4 Run the simulation.

5 View the data as it appears in Figure 6.34.



Monte Carlo Setup

☑ Enabled   Simulation Runs : 5

Default Distribution

Seed: -1.000   [✓ Uniform] Gaussian Worst Case

Default Tolerances

Resistor :	20%	Capacitor:	10%
Inductor:	10%	DC Source:	0
Transistor :	30%	Digital TP :	10%

Specific Device Tolerances

Figure 6.30. Use this dialog box to set up Monte Carlo Analysis.

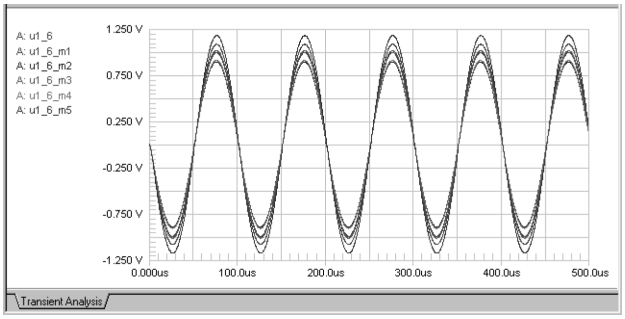


Figure 6.31. Waveforms for the Monte Carlo Analysis.

The waveform in Figure 6.31 was generated by simulating the Analog.ckt circuit using the values shown in Figure 6.30. Notice that this analysis randomly varies the component values within the specified tolerances, and then plots the output voltage of the five simulation runs, including the input and output voltages for the nominal run. See *Using the*

Professional Edition

Analysis Window earlier in this chapter for more information about manipulating the waveforms.

Specifying Default Tolerances

You can specify default tolerances for six general categories of devices: resistor, capacitor, inductor, DC source, transistor (beta forward), and digital TP (propagation delay for digital devices). You can enter tolerances as actual values or as percentages.

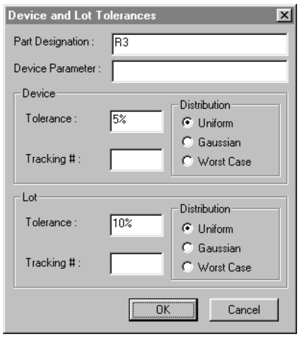
For example, you can enter a resistor tolerance as **10** or **10%**. If a 1kohm resistor has a tolerance of 10, it varies between 990 and 1010 ohms. With a tolerance of 10%, a 1kohm resistor varies between 900 and 1100 ohms. Each device is randomly varied independent of the other devices. For example, if a circuit has two 10kohm resistors, and the default tolerance is set to 10%, then during the first pass of the simulation, one resistor might have a value of 953 ohms, and the other one could be 1022 ohms. CircuitMaker uses a separate and independent random number to generate the value for each device.

Overriding with Specific Tolerances

To override the default tolerance value with specific tolerance values for specific devices,

1 Click **Add** in the Monte Carlo Setup dialog box.

2 Enter the appropriate information in the various fields (see Figure 6.32).


Device and Lot Tolerances

Part Designation : R3

Device Parameter :

Device

Tolerance : 5%

Tracking # :

Distribution: [✓ Uniform] Gaussian Worst Case

Lot

Tolerance : 10%

Tracking # :

Distribution: [✓ Uniform] Gaussian Worst Case

Figure 6.32. Use this dialog box to enter device and lot tolerances.

Professional Edition

For basic components like resistors, capacitors, and inductors, leave the **Device Parameter** field blank because there are no parameters besides the device value. Both device and lot tolerances are allowed, but only one or the other is required. CircuitMaker calculates device and lot tolerances independently (using different random numbers) and then adds them together. The example in Figure 6.29 has a device tolerance of 5% and a lot tolerance of 10%, and thus a total variation of up to 15%. In general, component values in the analysis vary independently, unless a tracking number is assigned. If you give two devices the same device tolerance tracking number and distribution, then the same random number is used for both devices when the device values for a simulation run are calculated. The same holds true for lot tolerances. However, device tracking and lot tracking are independent of each other. In other words, device tracking #1 and lot tracking #1 are unrelated.

To edit an item in the **Specific Tolerances** list in the Monte Carlo Setup dialog box,

1. Select the item you want to change.
2. Click **Edit** or click **Delete** to delete the item.

Impedance Plot Analysis

An Impedance Plot Analysis shows the impedance seen by any two-terminal source. Normally plotted in the AC Analysis window, an Impedance Plot does not have a separate setup dialog box.

To run an Impedance Plot Analysis, run a standard simulation, then click the source's negative terminal with the **Probe Tool**. An impedance plot appears in the selected analysis window. This is especially useful to measure input and output impedance versus frequency in the AC Analysis window. The impedance measurement is calculated from the voltage at the supply's positive terminal divided by the current out of that same terminal (see Figure 6.30). Clicking the positive terminal of a supply is already defined as an operation to plot current, thus the negative terminal must be clicked on for impedance.

Professional Edition

To measure input impedance,

1 Run the desired simulation (usually AC analysis).

2 Select the desired analysis window and then click on the negative terminal of the input source.

To measure output impedance,

1 Remove the source from the input.

2 Ground the circuit's inputs where the input supply was connected.

3 Remove any load connected to the circuit.

4 Connect a two-terminal source to the output, with the source's positive terminal connected to the output and the source's negative terminal connected to ground.

5 Run the desired simulation.

6 Select the desired analysis window and then click on the negative terminal of the source with the **Probe Tool**.

This causes an impedance plot to appear in the selected analysis window as in Figure 6.34.

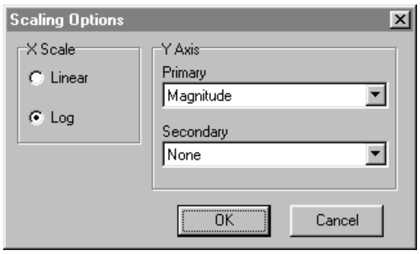


Figure 6.33. For Impedance Plots, you would normally change the Y axis to Magnitude. Click the Settings button in the analysis window to change X and Y axes for that window.

Professional Edition

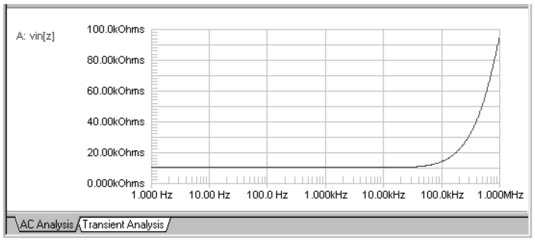


Figure 6.34. This Impedance Plot shows impedance seen by a two-terminal source over frequency.

The waveform in Figure 6.34 was generated by simulating the Analog.ckt circuit, and making an adjustment to the AC Analysis window as shown in Figure 6.33. See *Using the Analysis Window* earlier in this chapter for more information about manipulating the waveforms.

The Spice Simulator

When the simulation is run, CircuitMaker generates a Spice netlist to disk in a temporary file called ***filename*.NET** where *filename* is the name of the circuit file. Spice analyzes the netlist file and generates the simulation data. The amount of time it takes to complete the simulation is based on the analyses that are enabled, their sweep ranges, the complexity of the circuit, and the speed of your computer. The amount of data collected can be very large for a complex simulation and is discarded when changes are made to the circuit. You can greatly reduce the amount of simulation data by reducing the number of Test Points in the circuit. See the *Run-Time Test Points* section earlier in this chapter for more information.

Warning Messages vs. Error Messages

Sometimes CircuitMaker displays warning or error messages during circuit simulation. These messages are saved in a text file called ***filename*.ERR**. CircuitMaker prompts you to view these messages, displaying them in the Windows Notepad editor. The following distinguishes the two messages.

Warning Messages

Warning messages are not fatal to the simulation. They generally provide information about changes that Spice had to make to the circuit in order to complete the simulation. These include invalid or missing parameters, etc.

Note: Normally, valid simulation results are generated even if warning messages are reported.

SimCode warnings may include information such as timing violations (tsetup, thold, trec, tw, etc.) or significant drops in power supply voltage on digital components.

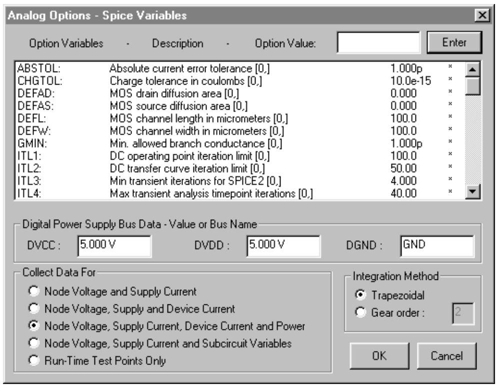
Error Messages

Error messages provide information about problems that Spice could not resolve and were fatal to the simulation. Error messages indicate that simulation results could not be generated, so they must be corrected before you will be able to analyze the circuit. If you need help troubleshooting Spice simulation errors, refer to *Chapter 16: Spice: Beyond the Basics.*

Setting Up Analog/Spice Variables

Spice allows you to control certain aspects of its simulation such as iteration limits, temperature, etc. Click on the **Analyses Setup** button in the Toolbar, then click the **Analog Options** button to display the dialog box pictured in Figure 6.35.

To learn more about the option variables listed in this dialog box, and how to change their values, turn to *Spice Option Variables* in *Chapter 16: Spice: Beyond the Basics*.


Analog Options - Spice Variables

Option Variables	Description	Option Value:	
ABSTOL:	Absolute current error tolerance [0,]	1.000p	*
CHGTOL:	Charge tolerance in coulombs [0,]	10.0e-15	*
DEFAD:	MOS drain diffusion area [0,]	0.000	*
DEFAS:	MOS source diffusion area [0,]	0.000	*
DEFL:	MOS channel length in micrometers [0,]	100.0	*
DEFW:	MOS channel width in micrometers [0,]	100.0	*
GMIN:	Min. allowed branch conductance [0,]	1.000p	*
ITL1:	DC operating point iteration limit [0,]	100.0	*
ITL2:	DC transfer curve iteration limit [0,]	50.00	*
ITL3:	Min transient iterations for SPICE2 [0,]	4.000	*
ITL4:	Max transient analysis timepoint iterations [0,]	40.00	*

Digital Power Supply Bus Data - Value or Bus Name

DVCC : 5.000 V DVDD : 5.000 V DGND : GND

Collect Data For

- Node Voltage and Supply Current
- Node Voltage, Supply and Device Current
- [✓ Node Voltage, Supply Current, Device Current and Power]
- Node Voltage, Supply Current and Subcircuit Variables
- Run-Time Test Points Only

Integration Method

- [✓ Trapezoidal]
- Gear order : 2

Figure 6.35. Use this dialog box to control certain aspects of the Spice simulation.

DVCC, DVDD and DGND

These are the digital device power buses. If no sources are specified in the circuit for these buses, these default values will be used. If a bus name is entered instead of a value, that bus will be connected to the devices. By default, DVCC and DVDD=+5V, DGND="GND". For more information, see *Digital Power Bus Connections* in *Chapter 4: Drawing and Editing Schematics*.

Integration Method

Choose which numerical integration method you want to use with XSpice. The *trapezoidal* method is relatively fast and accurate, but tends to oscillate under certain conditions. The *gear* method requires longer simulation times, but tends to be more stable. The gear order must be a value between 2 and 6. Using a higher gear order theoretically leads to more accurate results, but increases simulation time. Default=Trapezoidal.

Collect Data For

Use the five radio buttons (see Figure 6.35) to select the default level at which variables are stored. This affects how much memory is used and which variables can be plotted in the analysis windows when the simulation is run.

Node Voltage and Supply Current

Allows you to measure voltage on wires and current on voltage sources.

Node Voltage, Supply and Device Current

Allows you to measure voltage on wires and current on voltage sources and simple device pins (not on pins of subcircuit or macro devices.)

Node Voltage, Supply Current, Device Current and Power

Allows you to measure voltage on wires, current on voltage sources and simple device pins and power dissipation on simple devices.

Node Voltage, Supply Current and Subcircuit Variables

Allows you to measure voltage on wires, current on voltage sources and subcircuit internal variables.

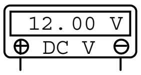
Run-Time Test Points Only

Allows you to make measurements only where Run-Time Test Points are located.

Analog/Mixed-Signal Instruments

Following is a description of instruments found in CircuitMaker's device library that are most commonly used in analog and mixed-signal circuits.

Multimeter

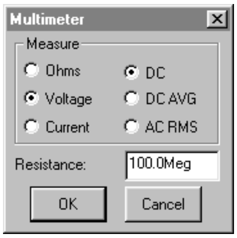


Multimeter

In addition to the Operating Point tab in the Analysis Window, CircuitMaker includes a multimeter device for measuring resistance, DC, DC AVG or AC RMS voltage or current. You may place as many multimeter devices into your circuit as you like. When you run the simulation, the measured value is then displayed on the meter. Remember, when measuring voltage, connect the meter in parallel with the circuit; when measuring current, connect it in series. When measuring resistance, be sure to remove any power sources from the circuit and beware of dangling devices that may cause XSpice errors. Also, since the multimeter forces a current through the circuit to measure ohms, make sure you have only one multimeter set to ohms in the circuit at a time. XSpice sees the current flowing into the positive terminal of a device as positive current.

When you place a Multimeter [.General/Instruments/ Multimeter] in your circuit, the dialog box shown in Figure 6.36 appears showing the settings of the multimeter.

Note: To measure DC AVG or AC RMS values, Transient Analysis must be enabled and must simulate enough cycles of transient data to make the measurements meaningful. Likewise, Operating Point Analysis (multimeter) must be enabled in order to obtain resistance and DC values.



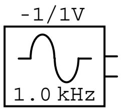
Measure: Voltage, DC

Resistance: 100.0Meg

Figure 6.36. This dialog box appears when you place a multimeter in your circuit.

The resistance of a voltmeter is high and the resistance of an ammeter is low, so it will have little effect on the circuit it is measuring. When measuring voltage on a high resistance circuit it may be desirable to increase the resistance of the voltmeter. When measuring current through a low resistance circuit it may be desirable to decrease the resistance of the ammeter. When measuring pin junction resistance, it may be desirable to increase the forcing current of the ohmmeter.

Multifunction Signal Generator



Signal Gen

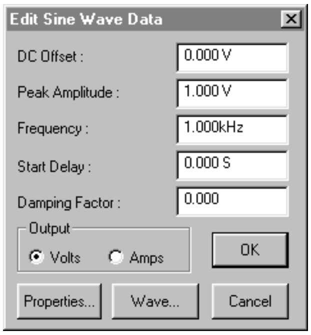
You can place as many of CircuitMaker's multifunction signal generators in your circuit as you like. Waveform functions include:

- Sinusoidal
- Single-Frequency AM
- Single-Frequency FM
- Exponential
- Pulse (including Triangle and Sawtooth)
- Piece-Wise Linear

The settings for each of these functions are edited through separate dialog boxes, which are described in the following sections.

Accessing the Signal Generator Editor

After placing a Signal Generator [.General/Instruments/ Signal Gen] (g) in your circuit, double-click it with the **Arrow Tool** to display the dialog box pictured in Figure 6.37.


Edit Sine Wave Data

DC Offset : 0.000 V
Peak Amplitude : 1.000 V
Frequency : 1.000kHz
Start Delay : 0.000 S
Damping Factor : 0.000

Output: [✓ Volts] Amps

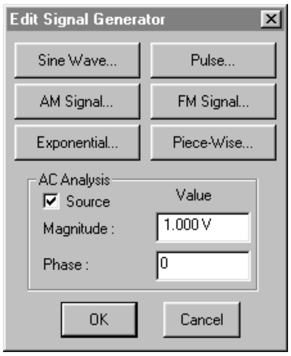
Figure 6.37. This dialog box shows the settings for the currently selected waveform function.

All of the waveform dialog boxes have the following items in common:

Volts/Amps
Allows you to select whether this is a voltage or current source.

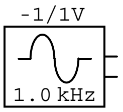
Properties Button
Displays the Device Properties dialog box described in *Editing Devices* in *Chapter 4: Drawing and Editing Schematics.*

Wave Button
Displays the dialog box pictured in Figure 6.38, allowing you to change waveform functions for this generator. To change waveform functions, click one of the function buttons and specify this generator's effect when running AC Analysis. AC Analysis temporarily replaces each AC source with a sinusoidal wave of a fixed magnitude and phase. The Source check box allows you to specify whether this generator will be used as an AC source in the AC Analysis. The Magnitude and Phase edit fields allow you to specify the fixed values that will be used.



☑ Source
Magnitude : 1.000 V
Phase : 0

Figure 6.38. Use the Edit Signal Generator dialog box to select the waveform you want to edit.



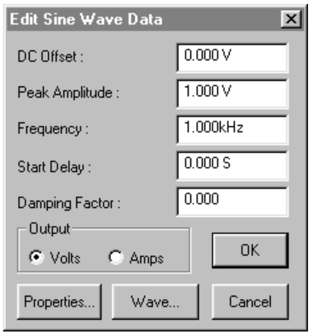
Editing Sine Wave Data

Click the **Sine Wave** button to display the dialog box pictured in Figure 6.39. Use this dialog box to set the parameters of the sinusoidal waveform.

The waveform, beginning at Start Delay, is described by the following formula where t = instance of time:

$V(t_0 \text{ to } t_{SD}) = VO$

$V(t_{SD} \text{ to } t_{STOP}) = VO + VA \sin(2\pi F\,(t\text{-}SD))\, e^{-(t-SD)THETA}$



Edit Sine Wave Data

DC Offset : 0.000 V
Peak Amplitude : 1.000 V
Frequency : 1.000kHz
Start Delay : 0.000 S
Damping Factor : 0.000
Output: [✓ Volts] Amps

Figure 6.39. Use this dialog box to edit sine wave data.

DC Offset (VO)

Used to adjust the DC bias of the signal generator with respect to the negative terminal (usually ground), measured in volts or amps.

Peak Amplitude (VA)

Maximum amplitude of the output swing, excluding the DC Offset, measured in volts or amps.

Frequency (F)

Frequency of the output in hertz.

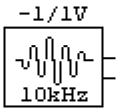
Start Delay (SD)

Provides a phase shift of the output by delaying the start of the sine wave.

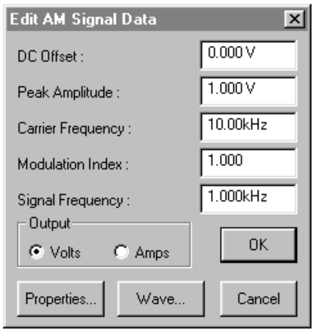
Damping Factor (THETA)

A positive value results in an exponentially decreasing amplitude; a negative value results in an exponentially increasing amplitude.

Editing AM Signal Data



Click the **AM Signal** button to display the dialog box pictured in Figure 6.40. Use this dialog box to set the parameters of the single-frequency AM waveform.


Edit AM Signal Data

DC Offset :	0.000 V
Peak Amplitude :	1.000 V
Carrier Frequency :	10.00kHz
Modulation Index :	1.000
Signal Frequency :	1.000kHz

Output: [✓ Volts] Amps

Figure 6.40. Use this dialog box to edit an AM signal.

DC Offset (VO)
Adjust the DC bias of the signal generator with respect to the negative terminal (usually ground), measured in volts or amps.

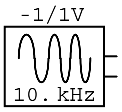
Peak Amplitude (VA)
Maximum amplitude of the output swing, excluding the DC Offset, measured in volts or amps.

Carrier Frequency (F_C)
Frequency of the unmodulated output in hertz.

Modulation Index (MDI)
Value corresponding to the percentage of amplitude modulation. 1 = 100%, 0.5 = 50%, etc.

Signal Frequency (F_S)
Frequency of the modulating signal in hertz.

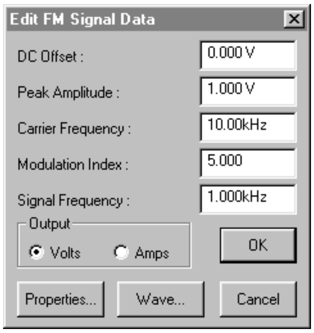
Editing FM Signal Data



Click the **FM Signal** button to display the dialog box pictured in Figure 6.41. Use this dialog box to set the parameters of the single-frequency FM waveform.

The waveform is described by the following formula where t = instance of time:

$$V(t) = VO + VA \sin(2\pi F_C t + MDI \sin(2\pi F_S t))$$



DC Offset : 0.000 V

Peak Amplitude : 1.000 V

Carrier Frequency : 10.00kHz

Modulation Index : 5.000

Signal Frequency : 1.000kHz

Output: [✓ Volts] Amps

Figure 6.41. Use this dialog box to edit an FM signal.

DC Offset (VO)

Adjust the DC bias of the signal generator with respect to the negative terminal (usually ground), measured in volts or amps.

Peak Amplitude (VA)

Maximum amplitude of the output swing, excluding the DC Offset, measured in volts or amps.

Carrier Frequency (F_C)

Frequency of the unmodulated output in hertz.

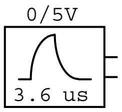
Modulation Index (MDI)

Value corresponding to a function of amplitude of the modulating signal indicating the level of modulation. MDI = (frequency deviation) / F_S

Signal Frequency (F_S)

Frequency of the modulating signal in hertz.

Editing Exponential Data

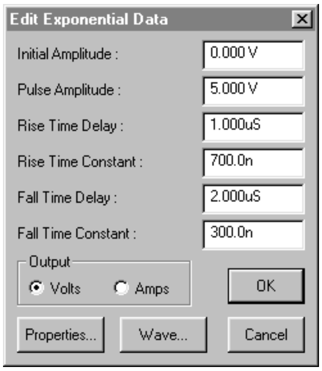


Use the dialog box in Figure 6.42 to set the parameters of the exponential waveform. The waveform is described by the following formulas where t = instance of time:

$V(t_0 \text{ to } t_{RD}) = VI$

$V(t_{RD} \text{ to } t_{FD}) = VI + (VP - VI)\,(1 - e^{-(t - tRD)/tRT})$

$V(t_{FD} \text{ to } t_{STOP}) = VI + (VP - VI)\,(-e^{-(t - tRD)/tRT}) + (VI - VP)\,(1 - e^{-(t - tFD)/tFT})$



Edit Exponential Data

Initial Amplitude :	0.000 V
Pulse Amplitude :	5.000 V
Rise Time Delay :	1.000uS
Rise Time Constant :	700.0n
Fall Time Delay :	2.000uS
Fall Time Constant :	300.0n

Output: [✓ Volts] Amps

Figure 6.42. Use this dialog box to edit exponential wave data.

Initial Amplitude (VI)
Initial amplitude of the output with respect to the negative terminal (usually ground), measured in volts or amps.

Pulse Amplitude (VP)
Max. amplitude of output swing, measured in volts or amps.

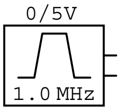
Rise Time Delay (RD)
The point in time, from t0, when the output begins to rise. This provides a phase shift of the output by delaying the start of the exponential waveform.

Rise Time Constant (RT)
Standard RC charging time constant.

Fall Time Delay (FD)
The point in time, from t0, when the output begins to fall.

Fall Time Constant (FT)
Standard RC discharging time constant.

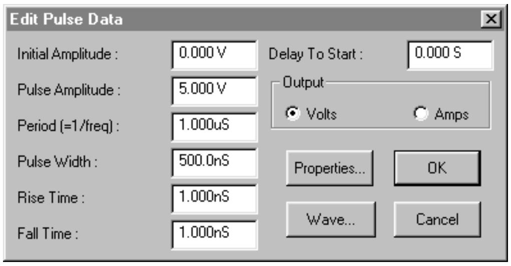
Editing Pulse Data



To set the parameters of the pulse waveform, click **Pulse/ Triangle/Sawtooth** to display the dialog box in Figure 6.43.

The waveform is described as follows where t = instance of time. Intermediate points are set by linear interpolation:

$V(t_0)$	=VI
$V(t_{SD})$	=VI
$V(t_{SD}+t_{TR})$	=VP
$V(t_{SD}+t_{TR}+t_{PW})$	=VP
$V(t_{SD}+t_{TR}+t_{PW}+t_{TF})$	=VI
$V(t_{STOP})$	=VI



Edit Pulse Data

Initial Amplitude : 0.000 V
Pulse Amplitude : 5.000 V
Period (=1/freq) : 1.000uS
Pulse Width : 500.0nS
Rise Time : 1.000nS
Fall Time : 1.000nS
Delay To Start : 0.000 S
Output: [✓ Volts] Amps

Figure 6.43. Use this dialog box to edit pulse data.

Initial Amplitude (VI)
Initial amplitude of the output with respect to the negative terminal (usually ground), measured in volts or amps.

Pulse Amplitude (VP)
Maximum amplitude of output swing, in volts or amps.

Period (=1/freq)
Duration of one complete cycle of the output.

Pulse Width (PW)
Duration output remains at VP before ramping toward VI.

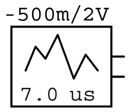
Rise Time (TR)
Duration of the ramp from VI to VP.

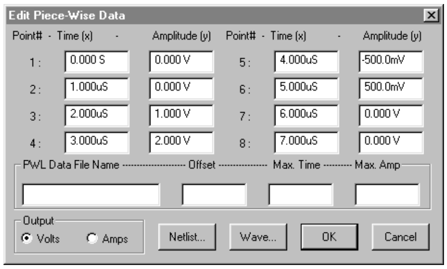
Fall Time (TF)
Duration of the ramp from VP to VI.

Delay to Start (SD)
Duration that the output remains at VI before beginning to ramp toward VP the first time.

Editing Piece-Wise Data

Click the Piece-Wise button to set the parameters of the piecewise linear waveform as pictured in Figure 6.44.




Edit Piece-Wise Data

Point#	Time (x)	Amplitude (y)	Point#	Time (x)	Amplitude (y)
1 :	0.000 S	0.000 V	5 :	4.000uS	-500.0mV
2 :	1.000uS	0.000 V	6 :	5.000uS	500.0mV
3 :	2.000uS	1.000 V	7 :	6.000uS	0.000 V
4 :	3.000uS	2.000 V	8 :	7.000uS	0.000 V

PWL Data File Name ---- Offset ---- Max. Time ---- Max. Amp

Output: [✓ Volts] Amps

Figure 6.44. Use this dialog box to set the parameters of the piecewise linear waveform.

Piecewise linear data must come from one of two sources:

- You can describe the waveform with a set of up to 8 points that you enter directly into this dialog box. The time specified for each successive point must be more positive than its predecessor. If it is not, the cycle will end, excluding that and all successive points.

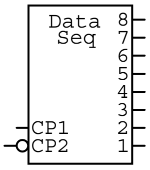
- You can define the waveform in an ASCII text file containing an indefinite number of points. Values must be entered in pairs and each pair must include a time position followed by an amplitude. The first character of each data line must be a plus sign (+) and each line may contain up to a maximum of 255 characters. Values must be separated by one or more spaces or tabs. Values may be entered in either scientific or engineering notation. Comments may be added to the file by making the first character of the line an asterisk (*). For example:

```
* Random Noise Data
+ 0.00000e-3  0.6667 0.00781e-3  0.6372 0.01563e-3 -0.1177
+ 0.02344e-3 -0.6058 0.03125e-3  0.2386 0.03906e-3 -1.1258
+ 0.04688e-3  1.6164 0.05469e-3 -0.3136 0.06250e-3 -1.0934
+ 0.07031e-3 -0.1087 0.07813e-3 -0.1990 0.08594e-3 -1.1168
+ 0.09375e-3  1.4890 0.10156e-3 -0.2169 0.10938e-3 -1.4915
+ 0.11719e-3  1.4914 0.12500e-3  0.1486
```

Intermediate points are determined by linear interpolation. If **Max. Amp** is specified in the dialog box, the data in the file will be scaled vertically so that the peak-to-peak amplitude of the waveform is equal to the specified **Max. Amp** and is centered around zero. If an Offset is specified, the waveform will be offset vertically by the specified amount. If a **Max. Time** is specified, the waveform will be scaled horizontally to fit the specified **Max. Time**. Example circuit: PWL.CKT.

Note: The .PWL file must be located in the same directory as the circuit you are simulating.

Data Sequencer



You can use this device in both digital and analog simulation modes. Also known as a Data or Word Generator, it allows you to specify up to 32767 8-bit words which can be output in a defined sequence. Since there is no limit to the number of Data Sequencers that you can use in a circuit, you could place several in parallel to create a data stream of any width. For a complete description of the data sequencer, refer to *Data Sequencer* in *Chapter 5: Digital Logic Simulation*

CHAPTER 7

Exporting Files

CircuitMaker offers several powerful export options, letting you export your circuit data to any of the following output types:

- Bill of Materials
- Waveforms as graphic files
- Waveform data as ASCII text files
- Schematics as graphic files
- Spice Netlists
- Spice Subcircuits
- PCB Netlists

This level of flexibility lets you easily integrate the work you do in CircuitMaker with a wide variety of other schematic capture, simulation, and printed circuit board layout programs.

CircuitMaker works seamlessly with TraxMaker, meaning that you can export to the TraxMaker PCB netlist format and also configure and launch TraxMaker directly from CircuitMaker.

This chapter defines the different types of output files and provides step-by-step procedures for them.

Bill of Materials

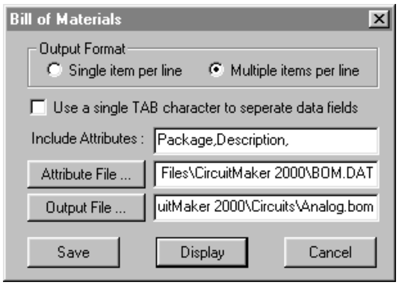
Formerly known as a Parts List, a Bill of Materials is a file that contains information about how many and which kinds of parts are used in your circuit. You can export a Bill of Materials to a text file.

To access the Bill of Materials feature,

1 Choose **File** > **Bill of Materials** to display the dialog box pictured in Figure 7.1.

2 See the following sections for more information.

You can save a Bill of Materials in one of two formats: Single Item per line or Multiple items per line.


Bill of Materials

Output Format: Single item per line [✓ Multiple items per line]

☐ Use a single TAB character to seperate data fields

Include Attributes : Package,Description,

Attribute File ... Files\CircuitMaker 2000\BOM.DAT

Output File ... uitMaker 2000\Circuits\Analog.bom

Figure 7.1. You can save, display, or print a Bill of Materials in one of two formats.

Single Item Per Line

This format puts each component on a separate line in the Bill of Materials. Items are listed in order according to the device designation. The columns are tab delimited. This format can easily be loaded into a spreadsheet and then sorted and arranged to your liking. The following is an example of the single item per line format:

```
Bill of Materials for:
C:\CM\Circuits\Diffamp.BOM

Item Label-Value  Attributes Designation
1    1N914        DIODE0.4   D1
2    1N914        DIODE0.4   D2
3    2N2222A      TO-92      Q1
4    2N2222A      TO-92      Q2
5    2N2222A      TO-92      Q3
6    3.2k         AXIAL0.4   R1
7    1.5k         AXIAL0.4   R2
8    50           AXIAL0.4   RB1
9    50           AXIAL0.4   RB2
10   7.75k        AXIAL0.4   RC1
```

Multiple Items Per Line

This Bill of Materials format lists components on the same line if the label-value, description and package fields match. Columns are tab delimited. The following is an example of the multiple items format:

```
Bill  of  Materials  for:
C:\CM\Circuits\Diffamp.BOM

Item  Count Label-Value Attributes  Designations
1     2     1N914       DIODE0.4    D1,D2
2     3     2N2222A     TO-18       Q1,Q2,Q3
3     1     3.2k        AXIAL0.4    R1
4     1     1.5k        AXIAL0.4    R2
5     2     50          AXIAL0.4    RB1,RB2
6     2     7.75k       AXIAL0.4    RC1,RC2
7     1     2.5k        AXIAL0.4    RE
```

As you can see in the examples, the three main things listed for each item are Label-Value, Attributes, and Designation. The default attributes are package and description, which can be set for each device in the Edit Device Data dialog box. Notice that none of the items in the list had a description, so only the package is listed. If multiple attributes are present for devices, then additional lines are added to list those attributes. For example, a group of items that has both a package and a description would require a second line to list the second attribute (description):

```
5     2     50          AXIAL0.4     RB1,RB2
                        5%  Resistor
```

Saving, Displaying, and Printing the Bill of Materials

To save the Bill of Materials to a file,

1 Enter the desired output file path and name in the Bill of Materials dialog box.

2 Click the **Save** button.

To display the Bill of Materials,

1 Click the **Display** button.

  The Bill of Materials appears in the Notepad editor.

2 Choose **File** > **Print** in the Notepad editor to print the Bill of Materials.

Including Attributes

Use the **Include Attributes** field in the Bill of Materials dialog box to enter items that you want to appear in the attribute column of the Bill of Materials. These items must be separated by commas, and should be listed in the order that you want them to appear in the Bill of Materials. There are two predefined attributes: package and description. These are listed as the default attributes in the **Include Attributes** field. Deleting an item from the **Include Attributes** field excludes that attribute from the Bill of Materials. You can enter other attributes in the **Include Attributes** field if they are defined an attribute file as described below.

Creating an Attribute File

You can create an attribute file to define additional attributes.

To use an attribute file,

1 Create an attribute file.

2 Select the attribute file in the **Attribute File** field of the Bill of Materials dialog box by typing the full path and naming the file, or by clicking the **Attribute File** button to find and select the file.

3 Define the attributes in the attribute file and enter the names of the attributes in the **Include Attributes** field of the Bill of Materials dialog box.

If you use an attribute file, devices that match an entry in the attribute file will have the defined attributes listed in the attribute column in the Bill of Materials. An entry in an attribute file must have one of the following two comma delimited formats:

Format 1

[Label-Value], [Package], [Description], [Attribute Name],
    [Attribute Data]

This format is used for all devices except for resistors, capacitors, and inductors.

Example:

```
ua741,DIP8,*,MFG,FairChild
```

Format 2

[Spice Prefix (R, C, or L)], [Package], [Description],
    [Attribute Name], [Attribute Data]

This format is used for resistors, capacitors, and inductors.

Example:

```
c,CAP0.2,*,MFG,Mallory
```

A device is considered to match an entry in the attribute file if the label-value (or spice prefix), package, and description of the device match those of the entry in the attribute file. The matching is case insensitive. You can use an asterisk (wild-card) to indicate that a match for a particular item is not required. In the Format 1 ua741 example, only the Label-Value and the package must match. The description does not need to match because of the asterisk.

A single line in the attribute file can only contain one attribute. Additional attributes for the same device type can be listed by using the plus (+) character. The plus character means that the Label-Value, package, and description for the new line are the same as the line above. This means that only the new attribute name and attribute data are listed. Here is the above ua741 example restated, with additional attributes:

```
ua741,DIP8,*,MFG,FairChild
+,COST,$0.25
+,MFGID,UA741N
```

The information in the **Include Attributes** field of the Bill of Materials dialog box could be as follows:

```
Package,Description,MFG,MFGID,cost
```

A corresponding entry in a Bill of Materials would then be:

```
24    3     UA741     DIP8          U1,U2,U3
                      Fairchild
                      UA741N
                      $0.25
```

Even though Description is listed in the **Include Attributes** field, a description does not appear in the Bill of Materials because the **Description** field in the Edit Device Data dialog is blank for this device. Note that the order of the attributes in the Bill of Materials depends on the order in the **Include Attributes** field, not the order in the attribute file. The following are examples of additional entries from an attribute file:

```
LM741,DIP8,*,MFG,National
+,COST,$0.26
+,MFGID,LM741N

r,AXIAL0.4,*,MFG,TRW
+,COST,$0.05

LF411,CAN8,*,MFG,National
+,COST,$0.26
+,MFGID,LM741N

c,CAP0.2,*,MFG,Mallory
```

Setting Up Export Options

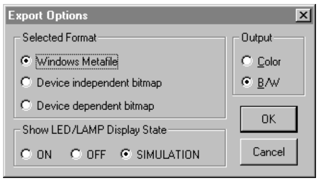
Use the Export Options dialog box to specify the format used when exporting schematic graphics to the clipboard or to a file.

To setup export options,

1 Choose **File > Export > Schematic Options** to display the dialog box shown in Figure 7.2.

2 Choose **On**, **Off**, or **Simulation** for the **Show LED/ LAMP Display State** option.

3 Choose Color or B/W (Black and White) to specify the output color.



Export Options

Selected Format: [✓ Windows Metafile] Device independent bitmap Device dependent bitmap

Output: Color [✓ B/W]

Show LED/LAMP Display State: ON OFF [✓ SIMULATION]

Figure 7.2. The Export Options dialog lets you specify output settings.

4 Choose an export format:

Format	**Characteristics**
Windows Metafile	Windows format for vector graphics.
Device Independent Bitmap	A pixel-by-pixel representation of a graphic that looks good at any resolution. Not available for analog waveforms.
Device Dependent Bitmap	A pixel-by-pixel representation of a graphic that only looks good at its original resolution. Not available for analog waveforms.

Exporting Waveforms as Graphics

You can export CircuitMaker waveforms and use them in documentation, presentations, etc. You can either save the waveform as a graphic file, or copy and paste the waveform directly into another software program.

Digital Logic waveforms can be saved in Windows Metafile, Device Independent Bitmap, and Device Dependent Bitmap formats. Use the Export Options dialog box, described earlier, to choose the format.

Analog/Mixed-signal waveforms can be saved in Windows Metafile format only.

Note: CircuitMaker saves only the information currently displayed in the active tab of the Analysis Window.

To save Digital Logic waveforms as graphics,

1 Run a simulation so that the waveforms appear in the digital waveforms window.

2 Choose **File** > **Export** > **Options** and select the desired format.

3 Choose **File > Export > Waveforms as Graphic**. Enter the name of the file where you want to save the circuit graphic, then choose **Save**.

OR

Choose **Edit** > **Copy to Clipboard** > **Waveforms**, then open another Windows program and Paste the waveform directly into your document.

To save Analog/Mixed-signal waveforms as graphics,

1 Run a simulation so that the waveforms appear in the analysis window.

2 Choose **File > Export > Waveform Metafile**. Enter the name of the file where you want to save the circuit graphic, then choose **Save**.

OR

Choose **Edit** > **Copy to Clipboard** > **Waveforms**, then open another Windows program and Paste the waveform directly into your document.

Exporting Waveform Data

Analog waveform data can be also be exported for use in other applications. The Save As Text command writes the waveform, exactly as it appears on the graph, to a tab delineated ASCII text file. Depending on the analysis type and setup, the data might not be linearized (the datapoints along the x-axis might not be evenly spaced).

To save analog waveform data in an ASCII text file,

1 Click on the waveform's label so that the waveform is selected.

2 Choose **Wave** > **Save As Text**. Enter the name of the file where you want to save the data, then choose **Save**.

Exporting Schematics as Graphics

You can export CircuitMaker schematics and use them in documentation, presentations, etc. You can either save the circuit as a graphic file, or copy and paste the circuit directly into another software program.

Use the Export Schematic as Graphic option to save the circuit to disk as a Windows Metafile, Device Independent Bitmap or Device Dependent Bitmap. Use the Export Options dialog box, described earlier, to choose the format.

To export the circuit as a graphic,

1 Choose **File** > **Export** > **Schematic Options** and select the desired format.

2 Choose **File > Export > Schematic as Graphic**. Enter the name of the file where you want to save the circuit graphic, then choose **Save**.

OR

Choose **Edit** > **Copy to Clipboard** > **Schematic**, then open another Windows program and Paste the circuit directly into your document.

Exporting a Spice Netlist

A Spice netlist is intended for simulation. For more information, see *Chapter 6: Analog/Mixed-Signal Simulation* and *Chapter 16: Spice: Beyond the Basics.*

Use the Spice netlist (.NET) option to create a Spice compatible text file describing your circuit. You can then import this text file into other Berkeley Spice3 compatible simulation programs.

1 Choose **File** > **Export** > **Spice Netlist**.

2 Type the name of the netlist file with a .NET extension, and then choose OK.

Exporting a Spice Subcircuit

Use the Spice Subcircuit option to create a .SUB text file describing your circuit. You can then copy the text file into a .SUB file which corresponds to the device symbol you intend to use. Connect SCOPE devices [Instruments/Digital/SCOPE] (T) to your circuit to indicate the connecting nodes of the subcircuit. Label them TP1 (1st node), TP2 (2nd node), etc., or anything that ends with a number (1, 2, 3, etc.) up to a maximum of 64 connecting nodes.

To export a Spice Subcircuit,

1 Choose **File** > **Export** > **Spice Subcircuit**.

2 Type the name of the subcircuit file with a .SUB extension, and then choose OK.

Exporting a PCB Netlist

Use the **PCB Netlist** export option to create a PCB netlist file for your circuit. CircuitMaker generates PCB netlists in TraxMaker, Protel, Tango, and many other formats, allowing you to use it with a wide variety of Printed Circuit Board layout programs.

What is a Net?

Each *net* represents one electrical junction point or *node* of a circuit. Numerous device pins may be connected to the same net.

What is a Netlist?

A *netlist* is an ASCII text file listing connections which describe the networks (or nets) of component connections in an electronic circuit. Widely used in electronics CAD packages, netlists let you transfer design details between applications, such as CircuitMaker and TraxMaker. Netlists generally contain two types of information:

- Descriptions of the individual components
- A list of all pin-to-pin connections

PCB Netlists come in various formats but generally carry similar data. They can often be translated into another format using a text editor. The file extension ".NET" is used for CircuitMaker netlist files.

As straightforward ASCII text files, netlists are easily viewed, created, and modified using a simple text editor or word processor (such as Notepad).

PCB Netlist Requirements

To create a PCB netlist, your circuit must meet the following requirements:

- Each device in the circuit must have a unique Designation (that is, D1, C3, etc.).
- Each device must have a value or label (that is, 10k, 2N2222A, etc.).
- Each device must have a Package type which should match one of the available Patterns in your PCB Layout program (that is, DIP14, AXIAL0.4, etc.).
- Pin numbers must be assigned for each device in the circuit.
- Designators and Package Descriptions (types) are limited to 12 alphanumeric characters. Net names can be 20 characters. Pin numbers in netlists are limited to 4 alphanumeric characters. No blank spaces may be used within these strings.

If any of these required items is missing, an error message will appear.

Exporting to Popular PCB Netlist Formats

CircuitMaker can export PCB netlists in the following formats, making your design usable in any product that supports these PCB netlist formats:

• TraxMaker	
• Protel	
• Tango	
• Orcad PCB II	(Professional Edition)
• PADS	(Professional Edition)
• CadNetix	(Professional Edition)
• Calay	(Professional Edition)
• Calay 90	(Professional Edition)

Note that package description names and pin numbers for each device in CircuitMaker must have exact matches to the library footprints of the pcb layout program you are using. For example, let's say your schematic contains a resistor, and you want that resistor to use an AXIAL0.4 footprint when the netlist is imported into TraxMaker. You would need to make sure that the name "AXIAL0.4" is in the Package field of that resistor (double-click on a part in CircuitMaker to access the Edit Device Data dialog box, to view and edit the package field).

To export a PCB netlist format other than the TraxMaker format,

1 Choose **File** > **Export** > **PCB Netlist**.

2 Select the appropriate format from the drop-down list such as Orcad PCB II.

3 Type the name of the netlist file, and then choose Save.

TraxMaker PCB Netlist Format

TraxMaker is a powerful pcb layout and autorouting program. It is fully compatible with CircuitMaker, and the products are designed to work together seamlessly, giving you a complete start-to-finish design system.

TraxMaker's netlist format is identical to the Tango and Protel netlist formats, and TraxMaker accepts either a dash

(-) or a comma (,) delimiter between the designator and pin number (U1-16 or U1,16). Some netlists, including TraxMaker netlists, provide separate formats for component descriptions and connections. Others combine the two sets of data in a single section. The following describes the TraxMaker pcb netlist format.

Character	**What it Does**
[	Marks the start of each component description.
U8	Labels the Component Designator (Designation field).
DIP16	Indicates the package description, type, or pattern (Package field). An identical description is required in the TraxMaker component library.
74LS138	Shows the component's comment (or value). Compare with Label-Value field.
Blank line	Left blank for future provision. There are usually three blank lines.
]	Marks the end of the component description.
(	Marks the start of each net.
NETU5_5	Names the net.
U5-5	Shows the first component (by designator) and pin number. Pin numbers in TraxMaker library components must be an exact match.
J21-1	Indicates the second pin in the net.
U8-3	Indicates another pin.
)	Marks the end of the net.

Note that net descriptions are distinguished from component descriptions by the use of rounded, rather than square brackets.

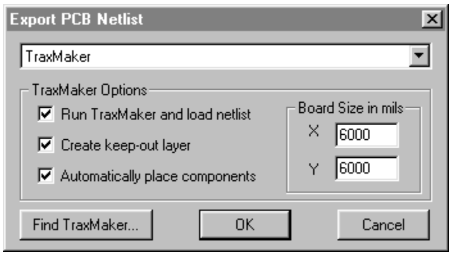
CircuitMaker to TraxMaker

Export to TraxMaker automatically using the TraxMaker Button.

TraxMaker is a powerful pcb layout and autorouting program. It is fully compatible with CircuitMaker, and the products are designed to work together seamlessly, giving you a complete start-to-finish design system. Because CircuitMaker and TraxMaker are tightly integrated, you can automatically export a TraxMaker netlists, open TraxMaker, define a keep out area, load the netlist, place and arrange the component footprints all with the **TraxMaker Button** in CircuitMaker.

To take a schematic from CircuitMaker into TraxMaker,

1 Choose **File** > **Export** > **PCB Netlist** (or click the TraxMaker button on the Toolbar) to display the dialog box pictured in Figure 7.3.



Export PCB Netlist

TraxMaker

TraxMaker Options
- [✓] Run TraxMaker and load netlist
- [✓] Create keep-out layer
- [✓] Automatically place components

Board Size in mils
X 6000
Y 6000

Figure 7.3. Not only can you export a TraxMaker netlist, but you can also launch TraxMaker, load the netlist, and automatically define a keep out area and place the components, all from within CircuitMaker.

2 Choose **TraxMaker** from the drop-down list (it is chosen for you by default).

3 From the **TraxMaker Options** group box, specify the options as described in the following sections.

Run TraxMaker and Load Netlist

Use this option to export the currently opened .CKT file to netlist (.NET) format, run TraxMaker, and load the netlist. If CircuitMaker can't find the TraxMaker executable file, use the **Find TraxMaker** button to locate it. Once you have located it, you don't have to find it again unless the location changes.

Create Keep-Out Layer

Use this option to define the board size for the TraxMaker AutoPlacement and Autorouting features, both of which attempt to stay within this area when placing and routing netlists. See the *TraxMaker User Manual* for more information.

Board Size in Mils

Use this option to define the size of the Keep Out Layer. See the *TraxMaker User Manual* for more information.

Automatically Place Components

This option activates the TraxMaker AutoPlacement feature, which uses several strategies when placing netlisted components on the Board. AutoPlacement attempts to arrange components by first grouping, then placing the groups using the placement and clearance grids you've specified. See the *TraxMaker User Manual* for more information.

CHAPTER 8

Fault Simulation

An exciting educational feature of CircuitMaker is the ability to place faulty devices in circuits. This lets you, as the instructor, to create troubleshooting exercises for students. For example, you can create a working circuit, then “zap” one or more of the devices by applying shorts, opens or other faults to selected pins on those devices. You can then password protect the fault data and limit the resources available to the student. You can add faults to both digital and analog circuits.

Create a working circuit before adding fault data. While CircuitMaker’s comprehensive fault simulator lets you create multiple, complex faults, use discretion when adding faults to a circuit. One or two damaged pins on an IC, an open resistor, or a shorted capacitor might be enough to challenge the student.

Device Faults

Each device might have multiple faults, but each device pin can have only a single fault. There are 6 basic types of device faults that can be simulated in CircuitMaker:

Pin(s) Stuck High

In Digital Simulation mode the specified pins will be connected to a logic high. In Analog Simulation mode the specified pins will be connected to an invisible, independent voltage source of an instructor-specified value.

Pin(s) Stuck Low

In Digital Simulation mode the specified pins will be connected to a logic low. In Analog Simulation mode the specified pins will be connected to an invisible, independent voltage source of an instructor-specified value.

Pin(s) Open

In Digital Simulation mode the specified pins will be disconnected from the device. In Analog Simulation mode the specified pins will be connected to the device through an invisible resistor of an instructor-specified value.

Pins Shorted Together

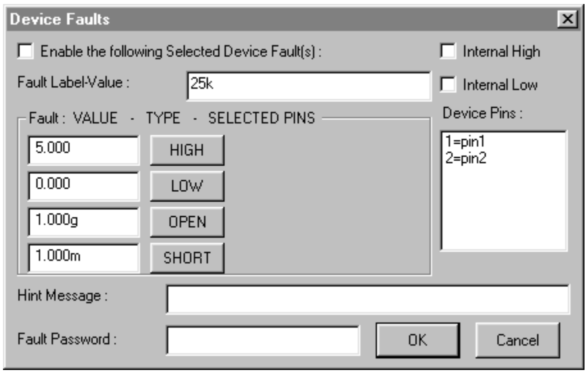
In Digital Simulation mode the specified pins will be shorted directly together. In Analog Simulation mode the specified pins will be connected together through a daisy-chain of invisible resistors of an instructor-specified value.

Wrong Value

For Analog Simulation mode only. The Fault Label-Value replaces the Label-Value for the device during simulation.

User Defined

For Analog Simulation mode only. The Spice model or subcircuit specified in the Label-Value for the device is replaced during simulation by the model or subcircuit specified in the Fault Label-Value. The faulty model or subcircuit may be one that has been modified by the instructor to operate incorrectly.



Device Faults

☐ Enable the following Selected Device Fault(s) :

Fault Label-Value : 25k

☐ Internal High

☐ Internal Low

Fault : VALUE - TYPE - SELECTED PINS

5.000	HIGH
0.000	LOW
1.000g	OPEN
1.000m	SHORT

Device Pins :

1=pin1
2=pin2

Hint Message :

Fault Password :

Figure 8.1. Use the Device Faults dialog box, which you access from the Edit Device Data dialog box, to set up faults for the device.

Adding Device Faults

To add faults to a device, follow these steps:

1 Double-click the device to display the Device Properties dialog box.

2 Click **Faults** to display the Device Faults dialog box.

Following is a description of the various items on the Device Faults dialog box (see Figure 8.1).

Enable Device Faults

When checked, the faults specified for this device will be enabled. It is enabled automatically when you edit fault data.

Fault Label-Value

Use this text box to enter the “wrong value” and “user defined” faults mentioned earlier in this chapter. If a value is entered in this field, it will replace the device’s Label-Value when the simulation is run. Faulty Spice model and subcircuit names may also be entered here. If the field is left blank, the device’s real Label-Value will be used in the simulation.

Faults and Device Pins

Select one or more pins from the **Device Pins** list, then click on one of the fault buttons. The selected pins will be removed from the selection list and placed to the right of the associated fault button. Pressing a fault button again will remove that fault from the pins listed next to it, and place the pins back in the selection list.

Button	**Used For**
HIGH	The **pin(s) stuck high** fault. In Analog/Mixed mode simulation, the value to the left of the button () indicates the voltage of an invisible voltage source to which the pins are stuck.
LOW	The **pin(s) stuck low** fault. In Analog/ Mixed mode simulation, the value to the left of the button () indicates the voltage of an invisible voltage source to which the pins are stuck.

Button	**Used For**
OPEN	The **pin(s) open** fault. In Analog/Mixed mode simulation, the value to the left of the button indicates the resistance (very high) between the pin and the device.
SHORT	Used for the **pins shorted together** fault. In Analog/Mixed mode simulation, the value to the left of the button indicates the value of the resistors (very low) that are daisy-chained between each of the selected pins.

Internal High/Low Check Boxes

Use these options for digital simulation only; they do not affect analog simulation. Stuck high and stuck low faults are assumed, by default, to be external to the device. Due to the nature of the digital simulation, it is desirable to use internal high/low faults on input pins and external high/low faults on output pins. For this reason you should not have an input pin and an output pin both stuck high or both stuck low on the same device.

Hint Message

The instructor can type a brief hint about the nature of the fault in this field that the student can access while troubleshooting the circuit.

Fault Password

The instructor can enter a password in this text box to restrict access to the fault data. If a password is entered here, all fault data will be password protected. If the password is deleted, the fault data will no longer be protected. Passwords are case sensitive and may be up to 15 characters in length. They may include any printable character, excluding Tab or Enter. Password protection:

- Restricts access to the Device Faults dialog box.
- Restricts access to the Circuit Faults dialog box.
- Eliminates fault comments from the Spice .NET file.

Using the Access Faults Dialog Box

If the fault data has been password protected, the Access Faults dialog box (pictured in Figure 8.2) appears whenever the student:

- Clicks the **Circuit Fault Data** button in the **Preferences** dialog box.
- Clicks the **Faults** button in the **Edit Device Data** dialog box.
- Double-clicks a device (if **Device Data Display** has been checked in the Circuit Faults dialog box).

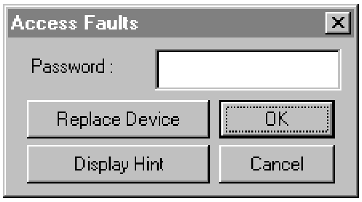


Figure 8.2. The Access Faults dialog box appears whenever a password has been set and the student tries to perform certain actions.

If **Replace Device** has not been disabled (in the Circuit Faults dialog box), the student can click on this button to replace a device. This disables the fault data for the device, but does not actually delete the fault data.

If **Replacement Status** has not been disabled (in the Circuit Faults dialog box), a message will then be displayed indicating whether the replaced device was good or faulty. Disable **Replacement Status** from the Circuit Faults dialog box.

If **Display Hint** has not been disabled (in the Circuit Faults dialog box), the student can click on the **Display Hint** button to view any hint the instructor has provided for that device. The instructor can see how many devices were replaced, and how many hints were displayed, all from the Circuit Faults dialog box. The instructor can also enter the password to gain access to the Device Faults or Circuit Faults dialog box.

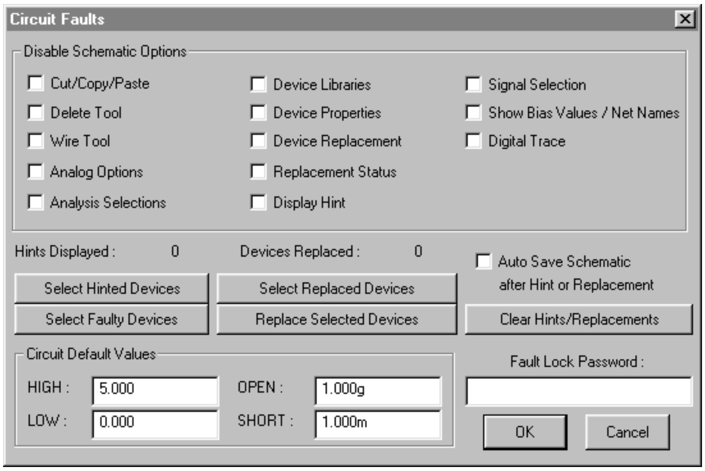
Managing Circuit Faults

Use the Circuit Faults dialog box (see Figure 8.3) to control which resources are available to students when faults are enabled. You can also define the default analog fault values. The settings specified in this dialog box are saved with the circuit.

To access the Circuit Faults dialog box,

1 Choose **Options** > **Schematic**.

2 Click the **Circuit Fault Data** button.



Hints Displayed : 0

Devices Replaced : 0

HIGH : 5.000

LOW : 0.000

OPEN : 1.000g

SHORT : 1.000m

Figure 8.3. Use this dialog box to control which resources are available to students when faults are enabled.

The following information describes the features on the Circuit Faults dialog box.

Disable Circuit Options

Use the options in this group box to limit the features available to the student.

Note: Only the items which are *not checked* are available to the student.

By default, none of the items are checked, meaning that the student has access to all of CircuitMaker's features.

Option	**Disables Student's Ability To**
Cut/Copy/Paste	Use the Cut, Copy, Paste, Duplicate and Move commands to modify the circuit. **Note:** This prevents the student from copying faulty devices into another circuit which is not password protected.
Delete Tool	Use the Delete Tool or the Delete key to modify the circuit. Note: This does not prevent the student from replacing a device (by clicking on the Replace Device button in the Access Faults dialog box).
Wire Tool	Use the Wire Tool or the Arrow/Wire Tool to modify the circuit.
Analog Options	Access the Analog Options dialog box.
Analysis Selections	Change the enabled/disabled status of the analog analyses, which limits the student to analysis types you specify. The student can, however, change settings of enabled analyses and run modified simulations.
Device Libraries	Select new devices from the device library. **Note:** This does not prevent the student from replacing a device by clicking on the **Replace Device** button in the Access Faults dialog box.
Device Properties	View or edit the device data in the Device Properties dialog box, and change the Visible status of the device labels and values.
Device Replacement	Use the Replace Device button in the Access Faults dialog box. **Note:** This does not prevent the student from deleting the device or selecting a new device from the device library.

Option	Disables Student's Ability To
Replacement Status	View a message when a device is replaced. The message indicates whether or not the replaced device was faulty.
Display Hint	View any hint messages.
Signal Selection	Change the settings of the Signal Generators.
Show Bias Values/Node Names	Display the OP analysis bias values on the schematic. Viewing the node names could help identify opens and shorts in a circuit.
Digital Trace	Use the Trace feature. The Trace feature is used in digital simulation to view the logic states of all the wires in the circuit.

Hints and Replacements

Each time the student presses the **Replace Device** or **Display Hint** button in the Access Faults dialog box, a flag is set for the selected device.

If you check the **Auto Save Circuit after Hint or Replacement** option, the circuit is saved automatically each time the buttons are pressed, preserving the student's work.

Option	Lets Instructor
Hints Displayed	View the number of hints that the student viewed.
Devices Displayed	View the number of devices the student replaced.
Select Replaced Devices	View (or highlight) all devices that were replaced.
Select Hinted Devices	View all devices for which the student viewed a hint.
Select Faulty Devices	View all devices that have fault data in them (whether the faults are enabled for that device or not).

Replace Selected Devices	Disable the fault data for all selected devices; does not delete the fault data.
Clear Hints/Replacements	Clear the flags.

Circuit Default Values

Use the Circuit Default Values group box to set the default values that correspond to the various faults in Analog Simulation mode. These defaults can be overridden for each device individually.

Value	**Default Value For the Invisible**
HIGH	Voltage source to which the specified pins of a device are connected when a HIGH fault is specified. The factory default is +5V.
LOW	Voltage source to which the specified pins of a device are connected when a LOW fault is specified. The factory default is 0V.
OPEN	Resistor placed between the specified pins and the device. The factory default is 1G ohm.
SHORT	Resistors that are daisy-chained between the specified pins of the device. The factory default is 1m ohm.

Fault Lock Password

The instructor might enter a password in this field to restrict access to the fault data. If you enter a password here, all fault data is password protected. If the password is deleted, the fault data will no longer be protected. Passwords are case sensitive and may be up to 15 characters in length. They may include any printable character, excluding Tab or Enter.

Warning: If you forget the password, you will not be able to access the fault data for the circuit.

Creating Black Box Macros

The instructor may lock a macro so that it cannot be expanded by the students, thus creating a “black box” macro.

To create a black box macro,

1 Choose **Macros** > **Expand Macro**.

2 Choose **Macros** > **Macro Lock**.

3 Enter any number (up to 4 digits), and then save the macro.

When you attempt to expand the macro again, a display prompt asks you to enter the 4-digit code. A code of 0 (zero) leaves the macro unlocked.

Warning: If you forget the code, you will not be able to expand the macro.

Fault Example

The following step-by-step example illustrates how to use the Circuit Faults dialog box.

1 Open the example circuit PS1.CKT.

2 Click the **Run** button on the Toolbar to run the simulation.

3 Select the Transient Analysis window, and then probe around on the circuit to verify that the circuit is working properly.

If you click on the positive side of the filter capacitor (C1), you should see a ripple on waveform at about 15VDC. If you click on the emitter side of the transistor (Q1), you should see a DC voltage of about 6V.

4 Stop the simulation by clicking the **Stop** button on the Toolbar.

5 Double-click the filter capacitor (C1) to display the Device Properties dialog box then click the **Faults** button.

6 Click on device pins 1 and 2 shown in the “Device Pins” list box. Click on the “short” button. This will cause a short to be placed across the capacitor.

7 Click on the “Enable the following selected Device Faults” check box to enable the fault.

8 Click OK to exit the Device Faults dialog box then click OK to exit the Edit Device Data dialog box.

9 Choose **Options** > **Schematic**.

10 Click the **Circuit Fault Data** button to display the Circuit Faults dialog box.

11 Click **Device Properties** so it is checked.

12 Click the **Clear Hints/Replacements** button to clear these flags.

13 Type **xxx** in the **Fault Lock Password** text box.

14 Click OK to exit the Circuit Faults dialog box then click OK to exit the Schematic Options dialog box.

15 If you want to save the fault settings for the circuit (not required for this example), you must now save the circuit.

16 Click the **Run** button on the Toolbar to restart the simulation.

17 Look at the waveform on the positive side of the filter capacitor. You should see a sine wave of very low amplitude (about 80mV peak-to-peak). The student might determine from this measurement that the filter capacitor is shorted and needs to be replaced.

18 Stop the simulation by clicking the **Stop** button in the Toolbar.

19 Double-click the filter capacitor (C1) to display the Access Faults dialog box then click the **Display Hint** button to view the hint.

20 Double-click the filter capacitor again to display the Access Faults dialog box then click the **Replace Device** button to disable the fault data for this device.

21 Run the simulation again to see if the problem has been fixed.

22 Stop the simulation.

23 Choose **Options** > **Schematic**, and then click the **Circuit Fault Data** button to display the Access Faults dialog box.

24 Type **xxx** into the **Fault Lock Password** text box then click OK to exit the Circuit Faults dialog box.

Notice that there was one device replaced and one hint displayed.

25 In the **Options** > **Schematic** > **Circuit Faults** dialog box, click the **Device Properties** check box to remove the check and delete the **xxx** from the **Fault Lock Password** field.

26 Click the **Select Replaced Devices** button.

27 Click OK to exit the Circuit Faults dialog box then click OK to exit the Schematic Options dialog box. The device that was replaced (the filter cap) is selected.

CHAPTER 9

File Menu

The File menu contains commands that enable you to open, save and print circuits and waveforms.

New

Select **New** to clear the current circuit from the work area and begin a new circuit. If an unsaved circuit is present in the work area when you select this command, you will be asked if you want to save the current circuit first.

Open

Choose **Open** to abandon the current circuit and load a different circuit into the workspace. If an unsaved circuit is present in the work area, you will be asked if you want to save the current circuit first. A file selector dialog will then be displayed allowing you to choose the circuit to load into the work area. The file extension used by default is .CKT. Circuit files created with older versions of CircuitMaker have the .CIR extension. To open these files, select the .CIR file type. As these files are opened, they will be converted to the new ASCII format and should be saved with the .CKT extension.

Reopen

Use the Reopen command to quickly reopen any of the last 8 circuits that you have used. If changes have been made to the current circuit you will be asked if you want to save the circuit before closing.

Merge

The **Merge** option lets you to add a circuit saved on disk to the circuit in the work area. This command is useful because it lets you save commonly used circuits or portions of circuits to disk then reuse them as often as you need, without having to rebuild them from scratch each time.

Select **Merge** and choose the circuit to be added to the work area. The circuit will be placed starting at the top left corner of the screen, so be sure to position the work area and existing circuit accordingly.

Close

Use **Close** to close the open windows. If changes have been made to the current circuit you will be asked if you want to save the circuit before closing.

Save

Choose **Save** to save the current circuit in the work area to disk using the name shown in the Title Bar. When you save a circuit whose title is UNTITLED.CKT, a file selector dialog box appears, allowing you to save it under a specific name.

Save As

Choose **Save As** to save the current circuit to disk using a file name that is different than the one shown in the Title Bar. CircuitMaker will display a file selector dialog box allowing you to choose the path and a file name of up to eight characters. The circuit is saved in an ASCII format with a default file extension of .CKT.

Revert

Choose **Revert** to abandon any changes made since last saving the circuit to disk and load the last saved version back into the work area. CircuitMaker will then display a dialog box asking you to confirm that you want to revert to the last saved version.

Import > Simulate Spice Netlist

Use this option to import and simulate a Spice netlist that you might have created in another simulation program. When you use this feature, CircuitMaker does not display the actual schematic drawing. Instead, it runs the simulation, displays the analysis window, and places a Waveforms list at the bottom of the Panel which you can use to choose different variables to plot. SVGA display is a minimum requirement to access the Waveforms list. See Figure 9.1.

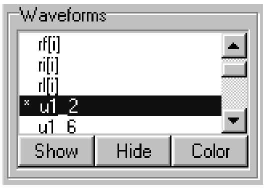


Figure 9.1. Waveforms list appears in the Panel when you import a Spice netlist.

To import and simulate a Spice netlist,

1 Choose **Import** > **Simulate Spice Netlist**.

2 Select a file with a .NET extension and choose **Open**.

4 Select a variable or variables from the Waveform list at the bottom of the Panel, then choose **Show**.

A waveform plots data for the variable you selected.

5 Use the **Show** and **Hide** buttons to view other waveforms.

Export

CircuitMaker allows you to export a variety of files and formats, including circuit and waveforms, pcb and Spice netlists. See *Chapter 7: Exporting Files* for a complete description of the exporting options.

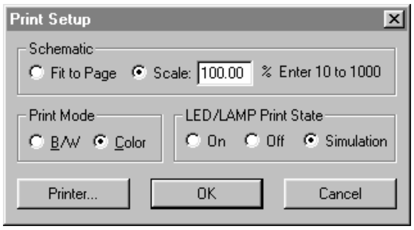
Bill of Materials

A Bill of Materials is a file that contains information about how many and which kinds of parts are used in your circuit. You can export a Bill of Materials to a text file. See *Exporting a Bill of Materials* in *Chapter 7: Exporting Files* for more information.

Schematic Print Setup

The **Schematic Print Setup** option allows you to scale the output to the printer 10% to 1000%, and choose between black and white, or color. You can also select the desired printer, paper size, page orientation and other printer properties by clicking the **Printer** button.

Note: The paper size that you select for you printer determines the size of the schematic document in CircuitMaker. For example, if your printer supports “B” size paper, then selecting that option in the printer’s Properties dialog will also setup CircuitMaker’s page breaks accordingly.



Schematic: ( ) Fit to Page [✓ Scale:] 100.00 % Enter 10 to 1000

Print Mode: ( ) B/W [✓ Color]

LED/LAMP Print State: ( ) On ( ) Off [✓ Simulation]

Figure 9.2. The Print Setup dialog box lets you scale the output and set other printing options.

Fit to Page

Use the **Fit to Page** option on the Print Setup dialog box to scale the schematic automatically so that the entire schematic fits on a single printed page.

Print Schematic

Choose **Print Schematic** to print the current schematic or to save it to a print file. A Printer dialog box appears, allowing you to select the desired output. If the schematic is too large to fit onto a single page, it will automatically be divided into page-size blocks as it is printed (see *Show Page Breaks* in the View menu). CircuitMaker supports all printers and printer options in the Printer dialog box. Printer setups will vary depending on the device you are printing to, but typically options such as page orientation and reduction factor are available.

Print Waveforms

Choose **Print Waveforms** to print the waveforms to a printer or save them to a print file. Only the information currently displayed in the active Waveforms or Analysis window is printed.

Exit

Choose **Exit** to exit CircuitMaker and return to Windows. If there are unsaved changes, you will be asked if you want to save them.

CHAPTER 10

Edit Menu

The commands in the Edit menu provide some of the tools necessary to construct and modify circuit diagrams.

Undo

Many of CircuitMaker's editing commands can be undone. Edit operations which can be undone are Cut, Paste, Delete and Move. Only the most recent edit can be undone and many commands are not able to be undone, including Run, Single Step, Reset, Define New Macro, Expand Macro and Delete Macro.

Cut

Use **Cut** to remove all selected items and place them in the paste buffer and on the system clipboard.

Copy

Use **Copy** to place all selected items in the paste buffer and on the system clipboard.

Paste

Normal cursor

Paste cursor

The **Paste** option displays the contents of the paste buffer in the CircuitMaker workspace and the displayed items follow the mouse around the screen. The cursor is replaced by the paste cursor, indicating the top left corner of the paste area. When the items are positioned at the desired location, click the mouse to finalize the paste. To cancel the paste, press any key or double-click the mouse.

Note: You cannot use Paste to paste items from the system clipboard.

Move

Use the **Move** option to perform both a Cut and a Paste. This provides a quick way to move an entire circuit or portion of a circuit to a new position on the screen. Connections to other portions of the circuit are lost.

Delete

This feature removes all selected items from the workspace.

Duplicate

Choose **Duplicate** to make a copy of all selected items. When this command is selected the duplicated items follow the mouse around the screen. The cursor is replaced by the paste cursor, indicating the top left corner of the paste area. When the items are positioned at the desired location, click the mouse to finalize the duplication. To cancel the duplication, press any key or double-click the mouse.

Copy to Clipboard

Sub-menus allow you to copy either the schematic or the waveforms to the system clipboard for placement in other applications.

Schematic

Choose **Copy to Clipboard** > **Schemetic** to copy the entire schematic to the system clipboard in one of three formats:

- Windows Metafile
- Device Independent Bitmap
- Device Dependent Bitmap

Choose **File** > **Export** > **Schematic Options** to select the desired format. This feature enables you to later paste the contents of the clipboard into a graphics program for further manipulation, printout, etc. When this command has been executed, an alert box appears informing you that the circuit has been copied to the clipboard.

Note: You cannot paste items into the CircuitMaker workspace that were copied using this command. It is designed only to copy the circuit to the system clipboard.

Waveforms

Use **Edit > Copy to Clipboard > Waveforms** to copy the contents of the Digital Waveforms window or Analysis window to the system clipboard in one of three formats:

- Windows Metafile
- Device Independent Bitmap (Digital mode only)
- Device Dependent Bitmap (Digital mode only)

Choose **File** > **Export** > **Options** to select the desired format. When copying waveforms to the clipboard, an alert box appears to inform you that the waveforms have been copied.

Before you can use this feature, the Analysis or Digital Waveform window must be open. You can copy only the waveforms currently visible to the clipboard. To ensure that meaningful waveform information is copied, set up the waveform display prior to using this feature.

***Note:** To find terminals or input or output connectors, put the connector name in the "Label or Value" field of the Find and Select dialog box.*

Select All

This option selects all items in the work area. This is useful when you want to cut, copy or move the entire circuit.

Find and Select

Use the Find and Select feature (Figure 10.1) to locate a specific device placed on the workspace by Node Name, Designation, Label or Value, or Symbol Name. If Circuit-Maker finds a device that matches your description, it is highlighted.

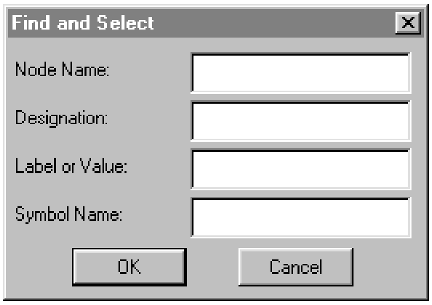


Figure 10.1. Use the Find and Select dialog box to search for devices you have placed in the workspace.

For example, if a Spice message indicates a problem at node c7_2 and your schematic is rather complex, this command can help you find the specified node. All wires connected to that node will be selected. You can also search for multiple devices. For example, to select all the transistors, enter **Q*** in the Designation field. All characters after the **Q** will be ignored in the search.

Rotate

This is the same as the Rotate button described in *Chapter 4: Drawing and Editing Schematics*.

Mirror

This is the same as the Mirror button described in *Chapter 4: Drawing and Editing Schematics.*

Straighten Wires

When you have wires with multiple bends, sometimes you may want to clean them up a little. This is a quick way of taking out some of the extra “kinks” without having to adjust the wire manually.

To straighten wires,

1 Select the wires you want to straighten.

2 Choose **Edit** > **Straighten Wires**.

Place Node Label

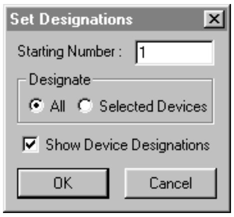
This command lets you place Node Labels on specific nodes in your circuit. See *Node Names* in *Chapter 4: Drawing and Editing Schematics*.

Set Label Positions

This command automatically places the device labels of all selected devices in their standard positions.

Set Designations

Choose **Edit** > **Set Designations** to display the Set Designations dialog box (pictured in Figure 10.3). Use this feature to renumber devices designations of devices you have already placed in the schematic. You can renumber all devices or a group of devices you have selected.


Set Designations

Starting Number : 1

Designate

[✓ All] Selected Devices

[✓ Show Device Designations]

Figure 10.3. Use this dialog box to renumber device designations for all devices or a group of devices.

The Set Designations feature is useful if you want the devices in a schematic, or a particular group of devices, to have designations numbered with a certain range of numbers. For example, you might want to renumber all the resistors in your schematic R40, R41, R42, R43, and so on.

To renumber device designations,

1 Unselect all devices (that is, don't select any) devices to renumber all devices on the board.

   OR

   Select a group of devices by holding the left mouse button while you draw a box around the devices.

   OR

   Hold down the **Shift** key while clicking the devices you want to select.

Tip: *If you select multiple devices using a Shift-click, the designation sequence will match the order in which the devices were clicked.*

2 Choose **Edit** > **Set Designations**.

3 Type a number in the **Starting Number** text box.

4 Select **All** if you want all devices to be renumbered.

   OR

   Click **Selected Devices** to renumber only those devices you have selected.

5 Select **Show Device Designations** if you want the designations to appear in the schematic.

Designation assignments will start at the stated starting number, and proceed from there, skipping any designations that are already in use.

Note: Set Designations is not the same thing as the Device Designations feature on the Options menu, which numbers newly placed devices with a specified number, or the next available number after the starting number you specify. For more information, see *Device Designations* in *Chapter 12: Options Menu*.

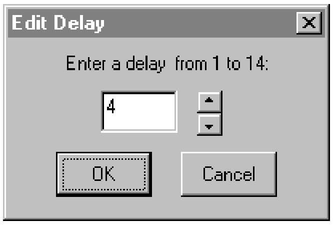
You can set the designation prefix for individual devices using the Device Properties dialog box described in *Editing Devices* of *Chapter 4: Drawing and Editing Schematics.* Each device must have a unique designation in order for analog simulation to work.

Set Prop Delays

Use **Edit > Set Prop Delays** (for Digital Logic Simulation only) to alter the propagation delay of all selected devices. The delay of a device determines how many simulation ticks it takes for a signal to propagate from the input to the output of the device.

1 Select a device then choose **Edit** > **Set Prop Delays** to display the dialog box pictured in Figure 10.2.

2 Enter a new value for the delay then click OK.


Edit Delay

Enter a delay from 1 to 14:

4

Figure 10.2. Use the Edit Delay dialog box to alter the propagation delay of all selected devices.

The default delay for all devices is one (1), but by using Set Prop Delays, you can change this value to any integer from 1 to 14. The units assigned to delay are arbitrary, and it is left to you to determine what they are. The concept is that if one device has a delay of one and another a delay of three, then in the real world the second device would have a propagation delay three times larger than the first device.

Group Items

Use this option to specify the devices that are contained within the same package. For example, a 7400 Quad 2-Input NAND gate actually has 4 gates in the same package. The pin numbers are different for each gate. The individual gates are grouped together automatically as they are placed in the circuit. You could regroup the gates using this feature.

To group items,

1 Select the items that you want to group in the same package.

2 Choose **Edit** > **Group Items**.

CHAPTER 11

View Menu

The View contains features that give you control of various display options. If a check mark or bullet is shown beside an item, then the item is currently enabled.

Panel

Use the Panel option to display or hide the Panel. The Panel is the area to the left of the drawing window which contains the Browse, Search, Digital and Wave tabs. Hiding the Panel leaves more room in the CircuitMaker workspace for the Drawing and Analysis windows.

Toolbar

Use the Toolbar option to display or hide the Toolbar. For an overview of all Toolbar options, see *Chapter 2: Getting Started* and *Chapter 4: Drawing and Editing Schematics.*

Status Bar

Use the Status Bar option to display or hide the Status Bar. The Status Bar is the area to the bottom of the CircuitMaker workspace. The Status Bar is used to display information such as simulation progress and model descriptions.

Collapse Device Tree

The Collapse Device Tree command allows you to quickly restore the device browse tree structure to its original, "unopened" appearance.

Schematic

One of four possible views during simulation. This view show the schematic, but not the waveforms.

Waveforms

One of four possible views during simulation. This view show the waveforms, but not the schematic.

Split Horizontal

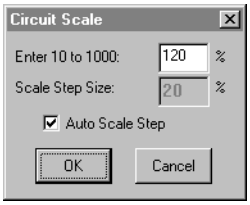
One of four possible views during simulation. This view places a horizontal splitter bar between the schematic and the waveforms.

Split Vertical

One of four possible views during simulation. This view places a vertical splitter bar between the schematic and the waveforms.

Display Scale

This feature lets you select the scale at which a circuit is displayed. It also lets you select the Scale Step of the Zoom Tool. See Figure 11.1.



Circuit Scale

Enter 10 to 1000: 120 %

Scale Step Size: 20 %

☑ Auto Scale Step

Figure 11.1. This dialog box controls the scale of the displayed circuit and the scale step of the Zoom Tool.

Normal Size/Position

This feature changes the circuit display scale to 100% and moves the horizontal and vertical scroll bars to their home positions.

Zoom To Fit

This feature will reduce or enlarge the circuit (by changing the Display Scale and scroll bar positions) so that the entire circuit can be displayed within the circuit window.

Refresh Screen

Choose **Refresh Screen** to redraw the entire circuit. Redrawing may be desirable following a command or operation which causes parts of the screen to become “messy.” Also see *Auto Refresh* in *Chapter 12: Options Menu.*

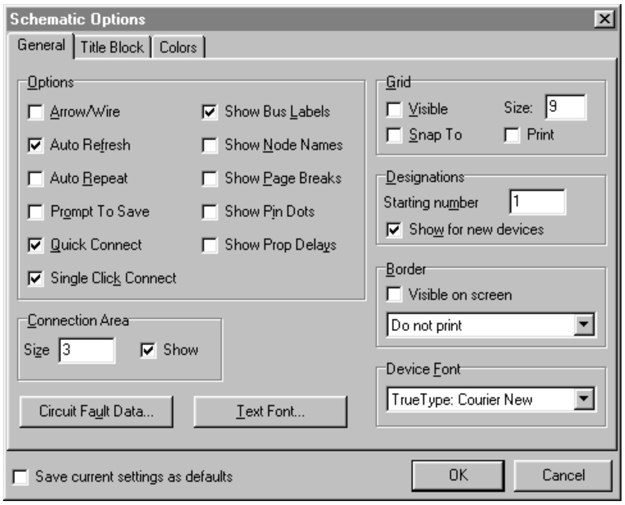
C H A P T E R   12

Options Menu

The Options menu contains commands which enable you to control various display, editing and simulation options.

Schematic

Choose **Schematic** to change the schematic options of CircuitMaker. Default settings are stored in the Cirmaker.dat file. See Figures 12.1, 12.4 and 12.5.



Options
☐ Arrow/Wire
☑ Auto Refresh
☐ Auto Repeat
☐ Prompt To Save
☑ Quick Connect
☑ Single Click Connect
☑ Show Bus Labels
☐ Show Node Names
☐ Show Page Breaks
☐ Show Pin Dots
☐ Show Prop Delays

Connection Area
Size 3 ☑ Show

Grid
☐ Visible Size: 9
☐ Snap To ☐ Print

Designations
Starting number 1
☑ Show for new devices

Border
☐ Visible on screen
Do not print

Device Font
TrueType: Courier New

☐ Save current settings as defaults

Figure 12.1. The Schematic Options dialog box lets you customize some of the schematic features.

Arrow/Wire

This option, when checked, allows you to initiate a wire by clicking once on a device pin with the Arrow Tool. You can only use the Arrow Tool to initiate a wire on a device pin or to extend an existing wire. You cannot initiate a wire from the middle of another wire using the Arrow Tool as you can with the Wire Tool. This option can be used for both auto and manual routing.

Auto Refresh

This option lets you control the refresh mode. When Auto Refresh is on, the screen refreshes automatically as the circuit is edited. When it is disabled, the screen must be refreshed manually while editing (see the Refresh Screen option in the View menu). Disabling this option lets you work more quickly on a slower computer system.

Auto Repeat

This option lets you control how devices are selected from the library. If **Auto Repeat** is checked, as soon as a device is selected from the library and placed, another identical device is automatically selected and made ready for placement. This repeat placement process continues until you cancel it by pressing any key on the keyboard or by double-clicking the mouse. If **Auto Repeat** is not checked, you must select and place each device separately.

Prompt To Save

If modifications have been made to the circuit, you may want to save the circuit before you run the simulation. If this box is checked, then each time you run the simulation Circuit-Maker will warn you if the circuit has not been saved.

Quick Connect

This option allows you to simply place a device pin on a wire or other device pin, and CircuitMaker will automatically wire the parts together for you. See *Wiring* in *Chapter 4: Drawing and Editing Schematics* for more details.

Single Click Connect

Use this option to manually route wires. When Single Click is enabled, you can terminate a wire by a single mouse click on any valid connection point. You can still terminate a wire at any location with a double-click, even if the end of the wire is not at a valid connection point. When the Single Click option is disabled, you must double-click to end the wire. This double-click mode lets you turn a wire very close to other wires or pins without connecting to them.

Show Bus Labels

This option displays bus labels (up to 5 alphanumeric characters) by putting a small text box at the end of each bus wire, and also shows the name of each bus connection wire next to the corresponding bus connection.

Show Node Names

This option displays the node names for each node in your circuit. The number is shown at the center of the longest wire segment of each node. The node names, which are determined by CircuitMaker, are used for simulation purposes and for PCB layout. They also correspond to the waveform names used in the Analysis Window.

Show Page Breaks

This option displays the page divisions of a multi-page circuit when it is sent to the printer. The Show Page Breaks option is also useful when positioning a circuit on a single page. The page breaks you see on your screen are based on the printer selection, paper size and scaling factor. The print scale can be visually adjusted by dragging the page breaks with the mouse.

Show Pin Dots

This option allows you to control how connections between wires and devices are drawn. If **Show Pin Dots** is checked, every device pin which has a wire connected to it will have a small dot where the connection occurs. If **Show Pin Dots** is not checked, the connecting dots will not be drawn. Dots showing where wires connect to other wires are always displayed.

Show Prop Delays

Use **Show Prop Delays** to display the propagation delay of all devices in the circuit. If **Show Prop Delays** is checked, the delay values for each device will be shown within a rounded rectangle located at the center of each device. Some devices (pulsers, LED's, macro devices, etc.) do not have a delay, so no value will be shown on them. In the case of macro devices, the delay is determined by the individual delay setting of each device contained within the macro.

If **Show Prop Delays** is not checked all devices will be drawn in normal form without the delay value being visible. To change the propagation delay of one or more devices select **Edit > Set Prop Delays**.

Connection Area

The connection area defines a rectangle around valid connection points (a valid connection point is any device pin or wire). The SmartWires™ feature lets you connect a wire to a valid connection point, even if your cursor is not exactly on that point. By adjusting the **Size**, you specify how close (within how many pixels) you must be to the connection point before the wire will snap to that point. By default, the cursor must be within 3 pixels of the connection point. When the **Show** checkbox is enabled, a rectangle will appear around the connection area each time the cursor enters that area.

Grid

Use the **Visible** option to turn the alignment grid of the circuit window on or off. The grid is useful as an aid in precisely aligning objects. Enable the **Snap To** checkbox to place new devices (devices not already in a circuit) according to the specified grid. It also lets you move old devices (devices already in the circuit) according to the selected grid, relative to their original position.

Note: When you place a device exactly on the grid, it always remains on the grid regardless of scroll position. However, the **Snap To** option does not guarantee alignment of component pins. Most of CircuitMaker's devices are designed for a 9-point grid size.

Designations

CircuitMaker automatically assigns a designation to each new device *when it is placed*. Use this option to specify the starting designation numbers and whether they are displayed on new devices as they are placed.

Compare this option with the **Edit** > **Set Designations** feature, which renumbers the designations of *already placed* devices.

To change device designation options,

1 Choose **Options** > **Schematic**.

2 In the Designations section, select **Show for new devices** to show the designation for all new devices that you place.

3 Type a number in the **Starting number** text box.

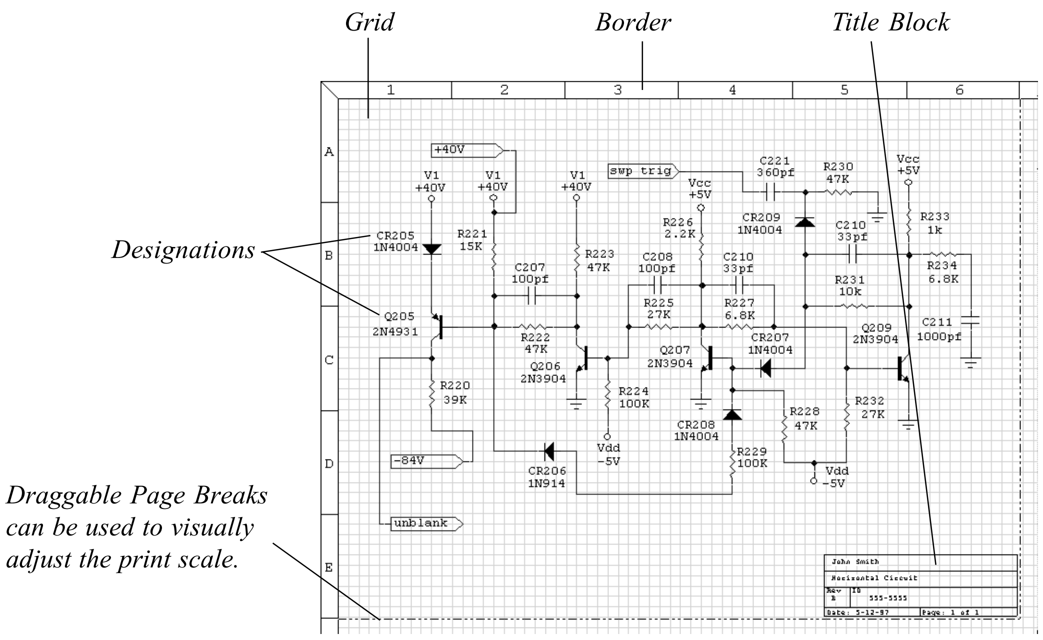
The devices that you place hereafter use the number you specified as the starting number, or the next available number after the specified starting number. For example, if the starting number is 10, and the schematic already contains resistors R10, R11, and R12, then the next resistor placed would be assigned the designation R13.

Border

Use this option to place a map coordinate border around your schematic printout. The border has numbers across the top and bottom, and has letters down both sides. This border can be useful in helping to quickly locate a specific area on the schematic.

Enable the **View on screen** checkbox to view the border in the schematic window. Only the top and left side borders will be visible on screen.

Choose one of three print options from the drop-down list. Select **Do not print** if you don't want to print the border. Select **Print around entire schematic** to print the border so that it is only on the outside edges of the outside pages, making a border around the entire schematic when the pages are arranged together. Select **Print around each page** to print the complete border on each page of your schematic.



Grid
Border
Title Block
Designations
Draggable Page Breaks can be used to visually adjust the print scale.

Figure 12.2. Use the Grid, Border and Title Block options to enhance the appearance of your schematic.

Device Font

This option lets you specify the device font used in Circuit-Maker schematics. This font is used on all device labels, wire labels, pin names, key caps and ASCII displays. It does not affect free text which is placed using the Text Tool. CircuitMaker's default device font is Courier New. This is a TrueType (scalable/rotatable), monospaced font that comes with the Windows operating system.

Circuit Fault Data

Click on the Circuit Fault Data button to access and modify the circuit fault options CircuitMaker will use. See *Circuit Faults* in *Chapter 8: Fault Simulation* for more information.

Text Font

Click on the Text Font button to change the font attributes of any selected, free-floating text field, or the default font for new text. See Figure 12.3.

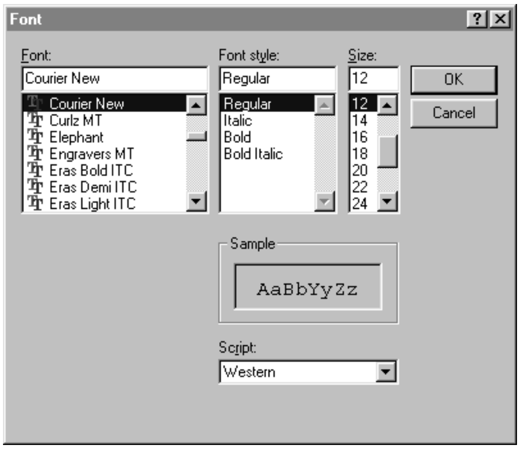


Figure 12.3. Use the Font dialog box to change the font for free-floating text fields or new text.

To change the font,

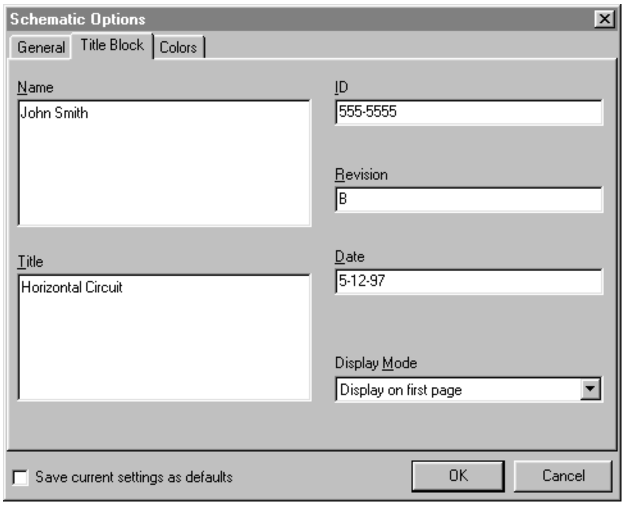
1 Select the free text by clicking on it.

2 Choose Options > Schematic, then click on the Text Font button.

3 Make the desired changes and click OK.

Title Block

Use the Title Block option to add a title box to the lower right corner of the page (see Figure 12.2 for an example.) The title block contains the following fields: Name, Title, Revision, ID, Date, and Page. The Name and Title fields expand in height to handle multiple rows of text. If you leave the Name or Title fields blank, CircuitMaker excludes them from the title block. The title block also expands in width according to the amount of text that you enter. You can print the title block on the first page, on the last page, or on all pages. Additionally, you can print the full title block on the first page and a reduced title block (that is, one that does not include the Name and Title fields) on subsequent pages.



Name: John Smith

ID: 555-5555

Revision: B

Title: Horizontal Circuit

Date: 5-12-97

Display Mode: Display on first page

☐ Save current settings as defaults

Figure 12.4. Title Block information is entered via the Title Block tab in the Schematic Options dialog box.

Colors

The Colors tab (Figure 12.5) lets you select the color associated with several functions (low level and high level, etc.) and items (Lamp color, Hex, and ASCII Key Cap color, etc.) A few devices can be assigned colors on an individual basis, others color assignments are global. The following example shows how to change the default LED color.

To change the default color for all new LEDs,

1 Choose **Options** > **Schematic** and click on the **Colors** tab.

2 Click on the colored rectangle next to LED.

3 Choose a new color and click OK. Click OK again.

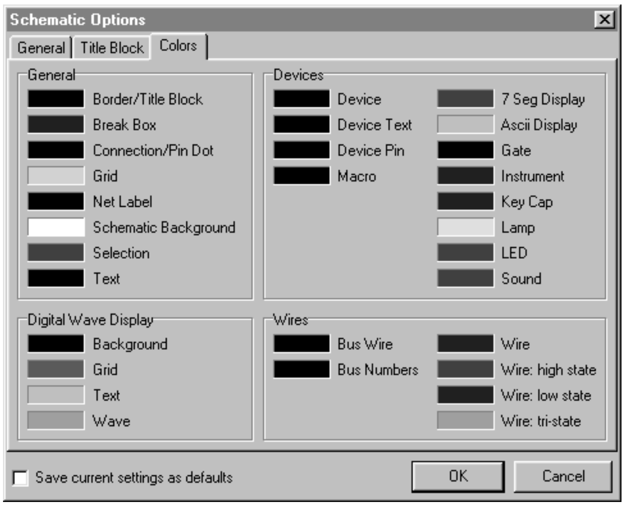


Figure 12.5. Use the Select Colors dialog box to change the color of different types of items.

To change the color of a single LED,

1 Place an LED on your schematic.

2 Right-click on the LED and choose **Device Color** from the pop-up menu. Choose the desired color and click OK.

Save current settings as defaults

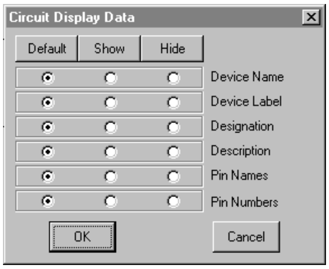
When this checkbox is enabled, all of the current settings in the Schematic Options dialog box will be saved as default settings. These default settings will be applied whenever you start CircuitMaker or create a new schematic.

Cursor Tools

This submenu provides an alternate method of selecting the Arrow, Wire, Text, Delete, Probe and Zoom tools.

Schematic Display Data

This feature (as seen in Figure 12.6) allows you to temporarily override the Visible settings of every device in the entire schematic. The buttons at the top of each column will quickly change the settings for the entire column, or you can change individual items by clicking on the radio buttons. The Default setting tells CircuitMaker to use the Visible settings of each individual device. Refer to *Editing Devices* in *Chapter 4: Drawing and Editing Schematics.*

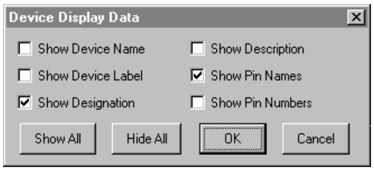

Circuit Display Data

Default	Show	Hide	
●	○	○	Device Name
●	○	○	Device Label
●	○	○	Designation
●	○	○	Description
●	○	○	Pin Names
●	○	○	Pin Numbers

Figure 12.6. Use the Schematic Display Data dialog box to temporarily override the Visible settings of every device in the schematic.

Device Display Data

This feature (pictured in Figure 12.7) allows you to quickly change the Visible settings of all selected devices. If the display item is currently visible on some of the selected devices, but not on others, the checkbox will be gray. Refer to *Editing Devices* in *Chapter 4: Drawing and Editing Schematics* for more information.



Device Display Data

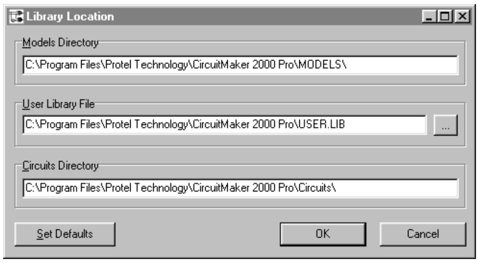
- [ ] Show Device Name
- [ ] Show Description
- [ ] Show Device Label
- [x] Show Pin Names
- [x] Show Designation
- [ ] Show Pin Numbers

Figure 12.7. Use the Device Display Data dialog box to quickly change the Visible settings of all selected devices.

Library Location

The directory paths used by CircuitMaker are user definable. Model Directory is the path to where your Spice model libraries are located. User Library File is the path and file name of your User.lib (Macros) file. Circuit Directory is the path to where CircuitMaker stores your circuit (.ckt) files.

Note: The Devicedb.dat, Hotkeydb.dat and Symboldb.dat files must be in the same directory as User.lib.



Library Location

Models Directory: C:\Program Files\Protel Technology\CircuitMaker 2000 Pro\MODELS\

User Library File: C:\Program Files\Protel Technology\CircuitMaker 2000 Pro\USER.LIB

Circuits Directory: C:\Program Files\Protel Technology\CircuitMaker 2000 Pro\Circuits\

Figure 12.8. The Library Location dialog allows you to select the path to your user library, models and circuits.

C H A P T E R   13

Macros Menu

Using commands in the Macros menu, you can expand CircuitMaker to meet your exact needs. This chapter describes the options used to create, edit and manipulate macros. Refer to *Chapter 17: Creating New Devices* for a step-by-step tutorial about creating macro devices.

New Macro

Select **New Macro** to create a new schematic symbol to add to the library. If there is a circuit displayed, CircuitMaker asks if you want that circuit included inside the macro for simulation. Macros are saved in a disk file called USER.LIB and once created, are available for use in any circuit or in another macro device.

Edit Macro

To edit a macro, follow these steps:

1 Expand the macro by highlighting it and choosing **Macros** > **Expand Macro**.

  Note: When a macro is expanded, the workspace will be cleared. If necessary, save your work beforehand.

2 Click the expanded macro's symbol to select it.

3 Choose **Macros** > **Edit Macro** to edit its name, device data or symbol.

  You can also double-click the expanded macro's symbol with the Arrow Tool.

If an existing macro is expanded and saved with a new name, it is saved as a new macro and does not overwrite the existing macro. The dialog box shown in Figure 13.1 appears when you use Edit Macro feature.

If no macro (either a newly defined or an expanded one) is present in the work area, the Edit Macro menu item is grayed indicating that it is not available.

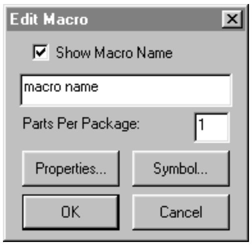


Figure 13.1. Use this dialog box to change an existing macro.

You can define a macro as having multiple parts per package; that is, you may specify that there are more than one of these macro circuits, all identical, in a single chip.

Save Macro

Use the **Macros > Save Macro** option to save or resave a macro device. An existing macro saved in this manner will be placed in its original position in the device library. All devices require a Major Device Class designation, so if you use the Save Macro option to save a new macro, CircuitMaker displays the Macro Utilities dialog box allowing you to specify both a major and minor class. To save an existing macro in a new location, refer to the *Macro Utilities* later in this chapter. To save under a different name, refer to *Edit Macro*. Macro devices are stored in the USER.LIB file.

Expand Macro

Use the **Macros > Expand Macro** option to edit a previously saved macro device. Expanding a macro device means that the black box containing the macro's circuitry is opened and all internal circuitry is visible.

Note: Most of the devices included with CircuitMaker are not macro devices and cannot be expanded.

To expand a macro, follow these steps:

1 Click a single macro device to select it.

2 Choose **Macros** > **Expand Macro**. An alert box appears warning that expansion of the macro will clear the work area.

Note: If you haven't saved your work, click the Cancel button to abort the expansion, save the current workspace, and then reselect **Expand Macro**. If you don't need to save the work area, click OK to complete the expansion of the macro.

3 Make changes to an expanded macro using any of the available editing commands.

4 When you have finished making changes to the macro, choose **Macros** > **Save Macro**.

Macro Lock

An instructor may want to lock a macro so that it cannot be expanded by the students, thus creating a "black box."

To lock a macro, follow these steps:

1 Expand the macro.

2 Click the expanded macro's symbol to select it.

3 Choose **Macros** > **Macro Lock** to display the dialog box pictured in Figure 13.2.

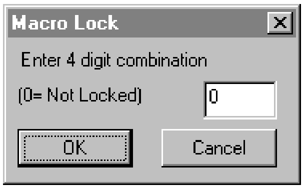


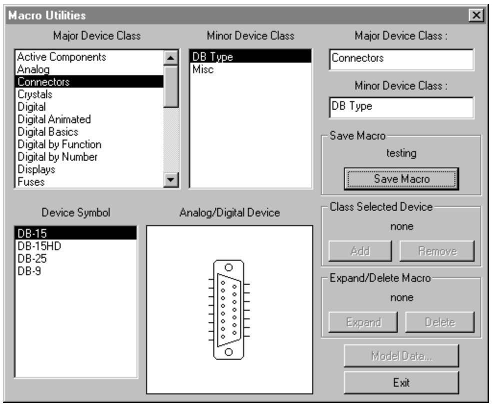
Figure 13.2. Use the Macro Lock feature to lock a macro as a "black box."

Enter any number (up to 4 digits), then save the macro. When you attempt to expand the macro again, you will be prompted to enter the 4-digit code. A code of 0 (zero) leaves the macro unlocked.

Warning: If you forget the code, you will be unable to expand the macro.

Macro Utilities

The Macro Utilities option on the Macros menu displays the dialog box pictured in Figure 13.3, which enables you to save, expand or delete a macro. The following sections describe the use of these features.



Major Device Class : Connectors
Minor Device Class : DB Type

Figure 13.3. The Macro Utilities dialog box lets you save, expand, or delete a macro.

Save Macro

The **Save Macro** button on the Macro Utilities dialog box lets you to save or resave a macro device. If there is no macro (expanded or newly defined) in the work area, this button is grayed. Refer also to the section *Save Macro* described earlier in this chapter.

To save a macro, follow these steps:

1 Select both a major and minor device class to indicate where the device is saved in the device library.

Note: You can add new major and minor class names by typing in a new name in the Major Device Class and Minor Device Class text edit fields.

2 Click **Save Macro**.

CircuitMaker saves the macro in the USER.LIB file and clears the workspace.

Class Selected Device

Use the Add and Remove buttons in the Class Selected Drive group box to move devices into different major and minor device classes.

To use this group box, follow these steps:

1 Select the device in the circuit window.

2 Choose **Macros** > **Macro Utilities**.

3 Select the major and minor device classes from the list boxes.

4 Click **Add** to add the device to the selected classes.

OR

Click **Remove** to remove the device from the selected classes.

If a device is removed from all classes, it will be inserted in the User Defined major class. To remove a device completely, see *Delete Macro* below.

Expand Macro

Use the Expand button in Expand/Delete Macro group box to perform the same function explained earlier in this chapter. To expand a macro device, select the macro then click the **Expand** button.

Delete Macro

Use the Delete button in the Expand/Delete Macro group box to delete a macro device from the USER.LIB file.

Warning: You cannot open a circuit which uses a macro that has been deleted from the library unless you first create a new macro with the same name.

To delete a macro device, follow these steps:

1 Save the workspace if you need to save.

When the macro is deleted the workspace will be cleared.

2 Make a backup of the USER.LIB file before you create or delete macros in case something goes wrong and you want to restore the original library.

Note: Most of the devices listed in the Macro Utilities dialog box are actually in the DEVICE.LIB file and cannot be deleted.

3 Select the macro.

4 Click **Delete Macro**.

Model Data

Use the Model Data button on the Macro Utilities dialog box to add new Spice models to CircuitMaker's library. This button displays a dialog box that lets you place new references into the linking file for selected symbol. Refer to *Model and Subcircuit Linking Files* in *Chapter 17: Creating New Devices.*

Macro Copier

The following steps illustrate how you can use the Macro Copier to copy your macros into a new version of USER.LIB.

1 Choose **Macros** > **Macro Copier** to display the dialog box pictured in Figure 13.4.

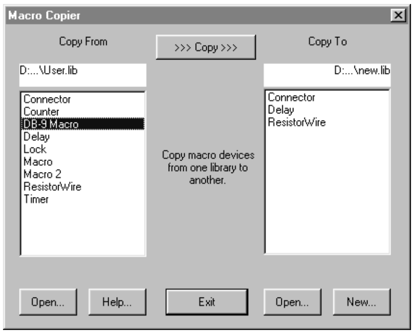


Figure 13.4. Use this dialog box to copy macros from one macro library to another.

2 Click lower left **Open** button to display the Open dialog box.

CircuitMaker asks you if you want to list only the user defined devices.

3 If you are copying devices that you have created, click **Yes**.

4 Select the file *from which* macros will be copied.

This is the library file which has your macros in it (probably USER.LIB in your old CircuitMaker directory).

5 Click the lower right **Open** or **New** button to display either the Open or the Save As dialog box.

6 Select the file *to which* macros will be copied.

This is usually the USER.LIB file in your new CircuitMaker directory.

7 Select a macro to copy from the left hand file list. If a macro already exists with the same name, CircuitMaker will delete the existing macro.

8 Click the **>>>Copy>>>** button.

CircuitMaker might prompt you for information regarding the simulation mode for which the device is intended. If the device can be used in digital simulations, check the **Digital** box; if it can be used in analog simulations, check the **Analog** box. If it can be used in either simulation mode, check both boxes.

Save ASCII Library

Use **Save ASCII Library** to write the currently loaded USER.LIB file to an ASCII file. CircuitMaker displays a file selector dialog box asking for the name of the ASCII file. In this format, user defined symbols which have been saved in Windows metafile or bitmap format will be lost and replaced by a simple rectangle. The pins, however, remain intact. The ASCII format is used for conversion between 16- and 32-bit systems. This library may be converted back to binary format by choosing **Macros** > **Convert ASCII Library**.

Convert ASCII Library

Use **Convert ASCII Library** to convert an ASCII user library file (created with the **Save ASCII Library** option) to binary format. CircuitMaker displays a file selector dialog box asking for the name of the ASCII user library file to be converted. Then a second file selector dialog appears, asking for the name of the new binary file. This option is provided for compatibility between 16- and 32-bit systems.

Update Search List

Use this feature to “refresh” the SEARCHDB.DAT file. Normally, the SEARCHDB.DAT file is updated automatically when necessary. If for any reason the Device Search list seems incorrect or out of date, you can create a new search list file by choosing **Macros** > **Update Search List**.

CHAPTER 14

Simulation Menu

The Simulation menu contains options that let you choose a simulation type and different kinds of analyses.

Analog Mode

This item is used to select the Analog/Mixed-signal simulation mode. This mode is described in *Chapter 6: Analog/ Mixed-Signal Simulation.*

Digital Mode

This item is used to select the Digital Logic simulation mode. This mode is described in *Chapter 5: Digital Logic Simulation.*

Analyses Setup

Use the Analyses Setup dialog box to setup the Spice analyses you want to perform, as well as simulation options such as temperature, tolerances, etc. For more information see *Analyses Setup* and *Analog Options in Chapter 6: Analog/Mixed-Signal Simulation*.

Check Pin Connections

Use Check Pin Connections to check the entire schematic and be informed of any devices that have unconnected pins.

Reset

This works just like the **Reset** button in the Toolbar, which is described in *Chapter 5: Digital Logic Simulation* and *Chapter 6: Analog/Mixed-Signal Simulation*.

Step

This works just like the **Step** button in the Toolbar, which is described in *Chapter 5: Digital Logic Simulation*.

Run/Stop

This works just like the **Run/Stop** button in the Toolbar, which is described in *Chapter 5: Digital Logic Simulation* and *Chapter 6: Analog/Mixed-Signal Simulation*.

Trace

This works just like the **Trace** button in the Toolbar, which is described in *Chapter 5: Digital Logic Simulation*.

Active Probe

This option, when checked, causes the logic levels detected by the **Probe** Tool to be displayed in the digital waveforms window while running digital logic simulation.

CHAPTER 15

Wave Menu

The Wave menu is used to access commands specific to the Analog/Mixed-Signal waveforms.

Fit Waveforms

This command will automatically scale the x and y axes of all the waveforms so that they are placed on a common scale and which fits the waveform with the greatest amplitude to fill the grid. This command can also be selected from the pop-up menu when you Right-click in the Analysis Window.

Zoom In

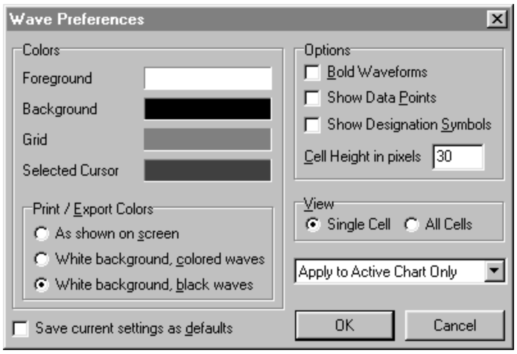
This command will magnify your view of the waveforms by adjusting both the x and y scales, maintaining the approximate center of the point of view. This command can also be selected from the pop-up menu when you Right-click in the Analysis Window.

Zoom Out

This command will reduce your view of the waveforms by adjusting both the x and y scales, maintaining the approximate center of the point of view. This command can also be selected from the pop-up menu when you Right-click in the Analysis Window.

Preferences

This command allows you to customize your view and print options for the Analysis Window. When you choose this command, you will see the dialog box shown in Figure 15.1. This command can also be selected from the pop-up menu when you Right-click in the Analysis Window.



Wave Preferences

Colors

Foreground	⬜
Background	⬛
Grid	▨
Selected Cursor	▨

Print / Export Colors

- ○ As shown on screen
- ○ White background, colored waves
- ● White background, black waves

☐ Save current settings as defaults

Options

- ☐ Bold Waveforms
- ☐ Show Data Points
- ☐ Show Designation Symbols

Cell Height in pixels: 30

View: ● Single Cell ○ All Cells

Apply to Active Chart Only

Figure 15.1. The Wave Preferences dialog box allows you to customize your view of the Analysis Window.

Colors

The Foreground (text), Background, Grid and Selected Cursor colors for the Analysis Window can all be set based on the user's preference. There are also two color schemes used for the waveforms themselves, but the scheme is chosen automatically by CircuitMaker, based on the selected background color. One scheme is provided for light colored backgrounds and the other for dark colored backgrounds.

Print / Export Colors

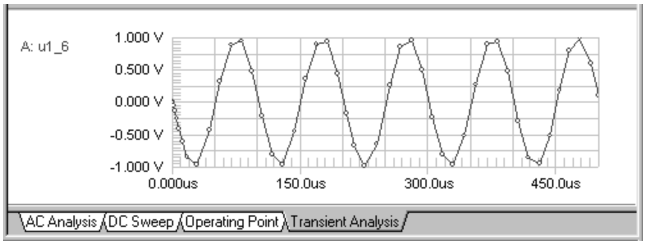
You have the option of printing the waveforms or exporting them to another application using the colors exactly as they appear on the screen. However, if your Analysis Window has a colored background you could end up using a lot of extra ink or toner. The second option is to print using a white background, but with colored waveforms. The third option is to print using a white background and black waveforms. If you choose the latter, you may wish to enable the **Show Designation Symbols** option (see below.)

Bold Waveforms

When enabled, CircuitMaker will draw all the waveforms using a double-width pen. This can sometimes be helpful when printing or exporting waveforms on a dark background.

Show Data Points

Enabling this option causes CircuitMaker to draw a tiny circle to mark each point on the waveform where an actual Spice-derived data point exists. CircuitMaker draws lines between these data points in a connect-the-dot fashion to complete the waveform. The more resolution (smaller step size) you allow in the analysis setup, the closer the spacing of the data points. Enabling this option can sometimes be very helpful when debugging a simulation. See Figure 15.2.



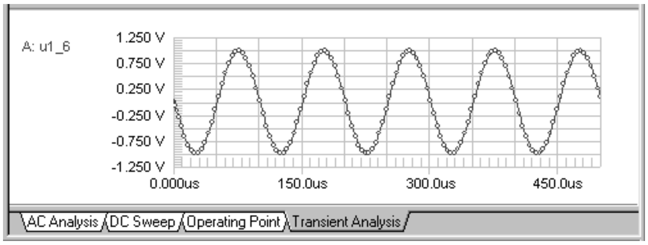


Figure 15.2. Notice the difference in the shape of the waveforms between these two simulation runs. The top waveform is set at a minimum of 5 steps/cycle. The bottom one is set at a minimum of 25 steps/cycle.

Show Designation Symbols

Designation symbols can be helpful in identifying waveforms on a single-color printout. If the waveforms are shown in individual cells, then all waveforms use a square symbol. If 2 or more waveforms share a cell, then a different shape is used for each waveform in that cell. See Figure 15.3.

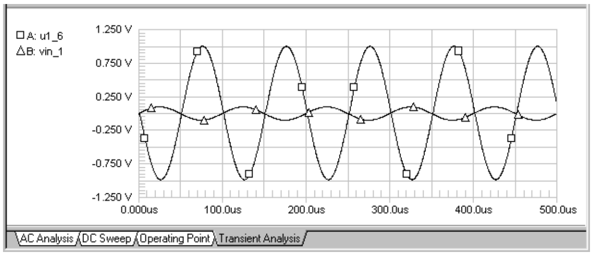


Figure 15.3. Use the designation symbols to clearly identify each waveform.

Cell Height in pixels

This option sets the height of all of the cells at once in All Cells view.

View

Selects either Single Cell or All Cells view. For more information, see *Using the Analysis Window* in *Chapter 6: Analog/ Mixed-Signal Simulation*.

Apply to...

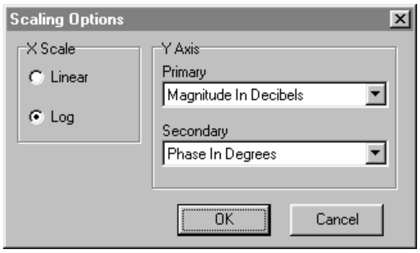
A drop-down list is provided which allows you to select whether the changed made in the Wave Preferences dialog box will apply to the Active Chart Only or to the Entire Document (all charts.)

Save current settings as defaults

This checkbox is provided to allow you to set the Wave Preferences as defaults. When set as defaults, you don’t have to set them up each time you start CircuitMaker.

Scaling

This command allows you to select the grid type to be used in the Analysis Window. When you choose this command, you will see the dialog box shown in Figure 15.4. This command can also be selected from the pop-up menu when you Right-click in the Analysis Window.



X Scale: Linear [✓ Log]

Y Axis
Primary: Magnitude In Decibels
Secondary: Phase In Degrees

Figure 15.4. Scaling Options dialog box.

X Scale

AC and Noise Analysis waveforms can be displayed on either a linear or logarithmic scale. Other waveforms are displayed on a linear scale only.

Y Axis

AC Analysis waveforms can be displayed on different grid types including: Real, Imaginary, Magnitude, Magnitude in Decibels, Phase in Degrees, Phase in Radians, and Group Delay. Two different grid types can be displayed simultaneously. Other waveforms are displayed on a Real grid only. See *AC Analysis Scaling Options* in *Chapter 6: Analog/ Mixed-Signal Simulation*.

Math

This feature allows you to create a new waveform based on a mathematical expression. This command can also be selected from the pop-up menu when you Right-click in the Analysis Window. For more information, see *Waveform Mathematics* in *Chapter 6: Analog/Mixed-Signal Simulation*.

Store

This command allows you to store a waveform in a file for future reference. This command can also be selected from the pop-up menu when you Right-click in the Analysis Window. For more information, see *Storing and Recalling Waveforms* in *Chapter 6: Analog/Mixed-Signal Simulation*.

Recall

This command allows you to recall a stored waveform. This command can also be selected from the pop-up menu when you Right-click in the Analysis Window. For more information, see *Storing and Recalling Waveforms* in *Chapter 6: Analog/Mixed-Signal Simulation*.

Save As Text

This command allows you to export a waveform for use in other applications. This command can also be selected from the pop-up menu when you Right-click in the Analysis Window. For more information, see *Exporting Waveform Data* in *Chapter 7: Exporting Files*.

CHAPTER 16

Spice: Beyond the Basics

This chapter discusses strategies for troubleshooting Spice convergence, Spice option variables, and Spice 's elementary devices. Use this information as a companion and reference as you complete the tasks in *Chapter 6: Analog/Mixed-Signal Simulation*.

Troubleshooting Spice Convergence

SPICE: Simulation Program with Integrated Circuit Emphasis

Berkeley Spice3 uses simultaneous linear equations, expressed in matrix form, to determine the operating point (DC voltages and currents) of a circuit at each step of the simulation. The circuit is reduced to an array of conductances which are placed in the matrix to form the equations (G * V = I). When a circuit includes nonlinear elements, Spice uses multiple iterations of the linear equations to account for the nonlinearities. Spice makes an initial guess at the node voltages then calculates the branch currents based on the conductances in the circuit. Spice then uses the branch currents to recalculate the node voltages and the cycle is repeated. This cycle continues until all of the node voltages and branch currents fall within specified tolerances (converge).

However, if the voltages or currents do not converge within a specified number of iterations, Spice produces error messages (such as "singular matrix", "Gmin stepping failed", "source stepping failed" or "iteration limit reached") and aborts the simulation. Spice uses the results of each simulation step as the initial guesses for the next step. If you are performing a Transient Analysis (that is, time is being stepped) and Spice cannot converge on a solution using the specified timestep, the timestep is automatically reduced, and the cycle is repeated. If the timestep is reduced too far, Spice displays a "Timestep too small" message and aborts the simulation. Use the Analog Options dialog box to specify the tolerances and iteration limits for the various analyses.

The Operating Point may fail to converge for various reasons. For example, the initial guesses for the node voltages may be too far off, the circuit may be unstable or bistable (there may be more than one solution to the equations), there may be discontinuities in the models, or the circuit may contain unrealistic impedances.

Use the following techniques to solve most convergence problems. When you have a convergence problem, first identify which analysis is causing the problem. Keep in mind that the Operating Point analysis is generally performed automatically before each of the other analyses, even if you have disabled Operating Point Analysis in the Analyses Setup dialog box. Begin with step 1, then consider the recommendations, as needed, to solve the error.

Solving Operating Point Analysis Failures

1 Check the circuit topology and connectivity; specifically,

- Make sure the circuit is correctly wired. Dangling nodes and stray parts are not allowed. The RSHUNT option can be used to overcome these problems.
- Don't confuse zeros with the letter O.
- Use proper Spice multipliers (MEG instead of M for 1E+6).
- Don't put a space between values and multipliers (1.0uF, not 1.0 uF).
- Every circuit must have a ground node and every node in the circuit must have a DC path to ground. Make sure no sections of your circuit are isolated from ground by transformers, capacitors, etc.
- Do not use series capacitors or current sources.
- Do not use parallel inductors or voltage sources.
- Make sure all devices and sources are set to their proper values.
- Make sure all dependent source gains are correct.
- Make sure your models/subcircuits have been correctly entered.

2 Increase ITL1 to 300 in the Analog Options dialog box. This will allow the OP Analysis to go through more iterations before giving up.

3 Add .NODESET devices. If the initial guess of a node voltage is way off, the .NODESET device can be used to set it to a more realistic value (see *.NODESET* later in this chapter).

4 Add resistors and use the OFF keyword. Specify the series resistance parameters of your models and increase the GMIN option by a factor of 10. Specify the initial condition of semiconductor devices, especially diodes, as OFF.

5 Use initial conditions. Enable the UIC check box for Transient Analysis in the Analyses Setup dialog box. Place .IC devices in your circuit or set the applicable initial conditions to assist in the initial stages of the Transient Analysis.

Solving DC Analysis Failures

1 Check the circuit topology and connectivity. See the common mistakes listed under Step 1 of *Solving Operating Point Analysis Failures* earlier in this chapter.

2 Increase ITL2 to 200 in the Analog Options dialog box. This will allow the DC Analysis to go through more iterations for each step before giving up.

3 Make the steps in the DC sweep larger or smaller. If discontinuities exist in a device model (perhaps between the linear and saturation regions of the model), increasing the step size may allow the simulation to step over the discontinuity. Making the steps smaller will allow the simulation to resolve rapid voltage-transition discontinuities.

4 Do not use DC Analysis. Some problems (such as hysteresis) cannot be resolved by DC Analysis. In such cases, it is more effective to use Transient Analysis, by ramping the appropriate power sources.

Solving Transient Analysis Failures

1. Check the circuit topology and connectivity. See the common mistakes listed under Step 1 of *Solving Operating Point Analysis Failures* earlier in this chapter.

2. Set RELTOL to 0.01 in the Analog Options dialog box. By increasing the tolerance from 0.001 (0.1% accuracy), fewer iterations will be required to converge on a solution and the simulation will complete much more quickly.

3. Increase ITL4 to 100 in the Analog Options dialog box. This will allow the Transient Analysis to go through more iterations for each timestep before giving up.

4. Reduce the accuracy of ABSTOL/VNTOL if current/voltage levels allow. Your particular circuit may not require resolutions down to 1uV or 1pA. You should allow at least an order of magnitude below the lowest expected voltage or current levels of your circuit.

5. Realistically model your circuit. Add realistic parasitics, especially stray/junction capacitance. Use RC snubbers around diodes. Replace device models with subcircuits, especially for RF and power devices.

6. Increase the rise/fall times of the Pulse Generators. Even the best pulse generators cannot switch instantaneously.

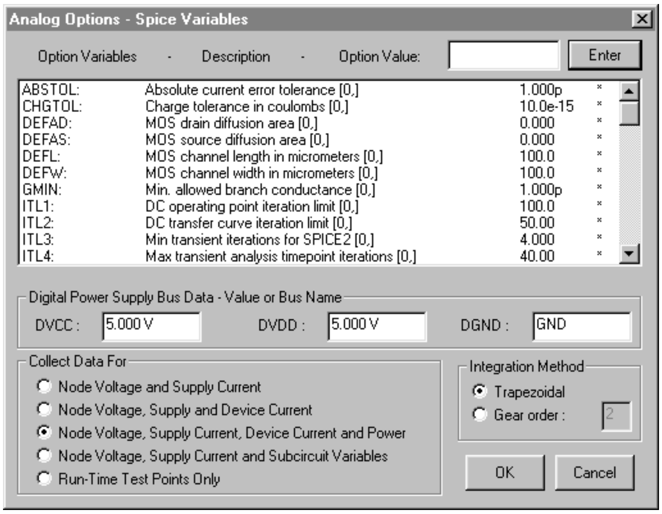
7. Change the integration method to Gear. Gear integration requires longer simulation time, but is generally more stable than the trapezoidal method. Gear integration may be particularly useful with circuits that oscillate or have feedback paths.

Spice Option Variables

Use Spice option variables to control certain aspects of a simulation such as iteration limits, temperature, etc.

To change the value of a Spice option variable,

1. Choose **Simulation** > **Analyses Setup > Analog Options** to display the Analog Options - Spice Variables dialog box pictured in Figure 16.1.
2. Select a variable and type the new value into the Option Value edit field.
3. If you want to set a specific option to its default value, type an asterisk in the Value edit field.


Analog Options - Spice Variables

Option Variables - Description - Option Value: [ ] Enter

Option Variables	Description	Option Value	
ABSTOL:	Absolute current error tolerance [0,]	1.000p	*
CHGTOL:	Charge tolerance in coulombs [0,]	10.0e-15	*
DEFAD:	MOS drain diffusion area [0,]	0.000	*
DEFAS:	MOS source diffusion area [0,]	0.000	*
DEFL:	MOS channel length in micrometers [0,]	100.0	*
DEFW:	MOS channel width in micrometers [0,]	100.0	*
GMIN:	Min. allowed branch conductance [0,]	1.000p	*
ITL1:	DC operating point iteration limit [0,]	100.0	*
ITL2:	DC transfer curve iteration limit [0,]	50.00	*
ITL3:	Min transient iterations for SPICE2 [0,]	4.000	*
ITL4:	Max transient analysis timepoint iterations [0,]	40.00	*

Digital Power Supply Bus Data - Value or Bus Name

DVCC : 5.000 V   DVDD : 5.000 V   DGND : GND

Collect Data For

- Node Voltage and Supply Current
- Node Voltage, Supply and Device Current
- [✓ Node Voltage, Supply Current, Device Current and Power]
- Node Voltage, Supply Current and Subcircuit Variables
- Run-Time Test Points Only

Integration Method

- [✓ Trapezoidal]
- Gear order : 2

Figure 16.1. This dialog box lists the Spice option variables, which you can change. An asterisk (*) in the dialog box denotes default values for each option.

Following is a list of the options and their effect on the simulation. See *Setting Up Analog/Spice Variables* in *Chapter 6: Analog/Mixed-Signal Simulation* for information about the other option in the Analog Options - Spice Variables dialog box.

Option	**What it Does**
ABSTOL	Sets the absolute current error tolerance of the program. Set ABSTOL=RELTOL* (lowest current magnitude in the circuit). Default=1 picoamp.
CHGTOL	Provides a lower limit on capacitor charge or inductor flux; used in the LTE timestep control algorithm. Default=1.0e-14 coulombs.
DEFAD	Sets the MOS drain diffusion area. Default=0.0 meters^2.
DEFAS	Sets the MOS source diffusion area. Default=0.0 meters^2.
DEFL	Sets the MOS channel length. Default=100.0 micrometer.
DEFW	Sets the MOS channel width. Default=100.0 micrometer.
GMIN	Sets the minimum conductance (maximum resistance) of any device in the circuit. It also sets the value of the conductance that is placed in parallel with each pn junction in the circuit. Default=1.0e-12 mhos. **Note:** Raising this value may help with simulation convergence in many circuits, but might decrease accuracy.
ITL1	Sets the Operating Point Analysis iteration limit. Default=100 iterations. **Note:** This may need to be raised as high as 500 for many circuits.
ITL2	Sets the DC Analysis iteration limit. Default=50 iterations. **Note:** This may need to be raised as high as 200 for some circuits.
ITL3	Sets the lower Transient Analysis iteration limit. Default=4 iterations. **Note:** This is not implemented in

	Spice3. It is provided in CircuitMaker for compatibility in creating Spice2 netlists.
ITL4	Sets the Transient Analysis timepoint iteration limit. Default=10 iterations. **Note:** Raising this value to 100 or more may help to eliminate "timestep too small" errors improving both convergence and simulation speed.
ITL5	Sets the Transient Analysis total iteration limit. Default=5000 iterations. **Note:** This is not implemented in Spice3. It is provided in CircuitMaker for compatibility in creating Spice2 netlists.
PIVREL	Sets the relative ratio between the largest column entry in the matrix and an acceptable pivot value. The value must be between 0 and 1. Default=1.0e-3. In the numerical pivoting algorithm the allowed minimum pivot is determined by EPSREL=AMAX1(PIVREL*MAXVAL, PIVTOL) where MAXVAL is the maximum element in the column where a pivot is sought (partial pivoting).
PIVTOL	Sets the absolute minimum value for a matrix entry to be accepted as a pivot. Default=1.0e-13.
RELTOL	Sets the relative error tolerance of the program. The value must be between 0 and 1. Default is 0.001 (0.1%). *Larger values mean faster simulation time, but less accuracy.*
TEMP	Sets the actual operating temperature of the circuit. Any deviation from TNOM will produce a change in the simulation results. Default=27°C. **Note:** TEMP can be overridden by a temperature specification on any temperature dependent instance.

TNOM	Sets the nominal temperature for which device models are created. Default=27°C. **Note:** TNOM can be overridden by a specification on any temperature dependent device model.
TRTOL	Used in the LTE timestep control algorithm. This is an estimate of the factor by which Spice overestimates the actual truncation error. Default=7.0.
VNTOL	Sets the absolute voltage tolerance of the program. Set VNTOL= RELTOL* (lowest voltage magnitude in the circuit). Default=1 microvolt.
BOOLL	Sets the low output level of a boolean expression. Default=0.0V.
BOOLH	Sets the high output level of a boolean expression. Default=4.5V.
BOOLT	Sets the input threshold level of a boolean expression. Default=1.5V.
BADMOS3	Uses the older version of the MOS3 model with the "kappa" discontinuity. Default=NO (don't use the older version).
KEEPOPINFO	Retains the operating point information when an AC Analysis is run. **Note:** This is particularly useful if the circuit is large and you do not want to run a redundant Operating Point Analysis. Default=NO (run OP each time).
TRYTOCOMPACT	Applicable to the LTRA model. When specified, the simulator tries to condense LTRA transmission line's past history of input voltages and currents. Default=NO (don't compact).
NOOPITER	Skip directly to GMIN stepping algorithm. Default=NO (don't skip).

GMINSTEP	Sets the number of steps in the GMIN stepping algorithm. When set to 0, GMIN stepping is disabled, making source stepping the simulator’s default DC (operating point) convergence algorithm. Default=10 steps.
SRCSTEP	Sets the number of steps in the source stepping algorithm for DC (operating point) convergence. Default=10 steps.
ACCT	Causes accounting and run-time statistics to be displayed. Default=NO (no display).
LIST	Displays a comprehensive list of all elements in the circuit with connectivity and values. Default=NO (no list).
OPTS	Displays a list of all standard Spice3 Option parameter settings. Default=NO (no list).
BYPASS	Enables the device bypass scheme for nonlinear model evaluation. Default=1 (on).
MINBREAK	Sets the minimum time between breakpoints. Default=0 seconds (sets the time automatically).
MAXOPALTER	Sets the maximum number of analog/event alternations for DC (operating point) convergence. Default=0.
MAXEVTITER	Sets the maximum number of event iterations for DC (operating point) convergence. Default=0.
NOOPALTER	Enables DC (operating point) alternations. Default=NO.
RAMPTIME	Controls turn-on time of independent sources and capacitor and inductor initial conditions from zero to their final value during the time period specified. Default=0.0 seconds.

CONVLIMIT	Disables convergence algorithm used in some built-in component models. Default=NO.
CONVSTEP	Sets the limit of the relative step size in solving for the DC operating point convergence for code model inputs. Default=0.25.
CONVABSSTEP	Sets the limit of the absolute step size in solving for the DC operating point convergence for code model inputs. Default=0.1.
AUTOPARTIAL	Enables the automatic computation of partial derivatives for XSpice code modules. Default=NO.
PROPMNS	Sets scale factor used to determine minimum propagation delay when actual value is not specified in SimCode model. Default=0.5 (50% of typical propagation delay).
PROPMXS	Sets scale factor used to determine maximum propagation delay when actual value is not specified in SimCode model. Default=1.5 (150% of typical propagation delay).
TRANMNS	Sets scale factor used to determine minimum transition time when actual value is not specified in SimCode model. Default=0.5 (50% of typical transition time).
TRANMXS	Sets scale factor used to determine maximum transition time when actual value is not specified in SimCode model. Default=1.5 (150% of typical transition time).
LOADMNS	Sets scale factor used to determine minimum input loading (maximum input resistance) when actual value is not specified in SimCode model.

	Default=1.5 (150% of typical input resistance).
LOADMXS	Sets scale factor used to determine maximum input loading (minimum input resistance) when actual value is not specified in SimCode model. Default=0.5 (50% of typical input resistance).
DRIVEMNS	Sets scale factor used to determine minimum output drive capacity (maximum output resistance) when actual value is not specified in SimCode model. Default=1.5 (150% of typical output resistance).
DRIVEMXS	Sets scale factor used to determine maximum output drive capacity (minimum output resistance) when actual value is not specified in SimCode model. Default=0.5 (50% of typical output resistance).
CURRENTMNS	Sets scale factor used to determine minimum supply current (maximum internal resistance) when actual value is not specified in SimCode model. Default=1.5 (150% of typical internal resistance).
CURRENTMXS	Scale factor used to determine maximum supply current (minimum internal resistance) when actual value is not specified in SimCode model. Default=0.5 (50% of typical internal resistance).
TPMNTYMX	Temporary global override for propagation delay index on SimCode devices (0=default, 1=min, 2=typ, 3=max). Default=0.
TTMNTYMX	Temporary global override for transition time index on SimCode devices (0=default, 1=min, 2=typ, 3=max). Default=0.

LDMNTYMX	Temporary global override for input loading index on SimCode devices (0=default, 1=min, 2=typ, 3=max). Default=0.
DRVMNTYMX	Temporary global override for output drive capacity index on SimCode devices (0=default, 1=min, 2=typ, 3=max). Default=0.
IMNTYMX	Temporary global override for supply current index on SimCode devices (0=default, 1=min, 2=typ, 3=max). Default=0.
SIMWARN	A nonzero value indicates that SimCode warning messages can be displayed during run time. SimCode warnings may include information concerning timing violations (tsetup, thold, trec, tw, etc.) or indicate supply voltage dropping below device specifications. Default=0.
RSHUNT	Value in ohms of resistors added between each circuit node and ground, helping to eliminate problems such as “singular matrix” errors. In general, the value of RSHUNT should be set to a very high resistance (1e+12). Default=0 (no shunt resistors.)
ADCSTEP	The minimum step size required to register an event on the input of the internal A/D converters. Default=0.01 volts.

Device Models

General Form

```
.MODEL MNAME TYPE(PNAME1=PVAL1 PNAME2=PVAL2 ... )
```

Netlist Example

```
.MODEL  MOD1  NPN(BF=50  IS=1E-13  VBF=50)
```

Most simple circuit elements typically require on a few parameter values. However, some devices (semiconductor devices in particular) that are included in Spice require many parameter values. Often, many devices in a circuit are defined by the same device model parameters. For there reasons, a set of device model parameters is defined on a separate .MODEL line and assigned a unique model name. The device element lines in Spice then refer to the model name.

For these more complex device types, each device element line contains the device name, the nodes to which the device is connected, and the device model name. In addition, other optional parameters may be specified for some devices: geometric factors and an initial condition (see the following section on Transistors and Diodes for more details.)

MNAME in the above general form is the model name, and type is one of the following fifteen types:

R	Semiconductor resistor model
C	Semiconductor capacitor model
SW	Voltage-controlled switch
CSW	Current-controlled switch
URC	Uniform distributed RC model
LTRA	Lossy transmission line model
D	Diode model
NPN	NPN BJT model
PNP	PNP BJT model
NJF	N-channel JFET model
PJF	P-channel JFET model
NMOS	N-channel MOSFET model
PMOS	P-channel MOSFET model
NMF	N-channel MESFET model
PMF	P-channel MESFET model

Parameter values are defined by appending the parameter name followed by an equal sign and the parameter value. Model parameters that are not given a value are assigned default values. Models, model parameters, and default values are listed in the next section along with the description of the device element lines.

Elementary Devices

Each device used in a circuit requires certain information for the Spice simulation to run. This section describes each type of device and information that each requires. CircuitMaker provides this information to the Spice netlist provided you properly draw and label the circuit. See *Editing Devices* in *Chapter 4: Drawing and Editing Schematics* for more information.

The information provided here will help you better understand how each of the components affects the simulation of the circuit. Parameters that are enclosed in "< >" symbols are optional. Refer to *Chapter 16:Creating New Devices* for more information about the models mentioned here.

Resistors

General Form

```
RXXXXXXX  N1  N2  VALUE
```

Netlist Example

```
R1  1  2  10K
```

Spice Data Example

```
%D  %1  %2  %V
```

N1 and N2 are the two element nodes. Value is the resistance (in ohms) and may be positive or negative, but not zero. If N1 is at a higher voltage than N2, the current flow through the resistor is positive. If N2 is at a higher voltage than N1, the current flow is negative.

See Also

Resistor, Var Resistor (example circuit: ANALOG.CKT)

Semiconductor Resistors

General Form

```
RXXXXXXX N1 N2 <VALUE> <MNAME> <L=LENGTH> <W=WIDTH> <TEMP=T>
```

Netlist Examples

```
R1  1  2  10K
R3  5  6  RMOD  L=12u  W=1u
```

Spice Data Example

```
%D  %1  %2  %M  TEMP=100
```

Spice Model Example

```
.MODEL  RMOD  R(TC1=15E-6)
```

This is the more general form of the resistor. It allows modeling of temperature effects, and calculation of the actual resistance value from strictly geometric information and the specifications of the process. If VALUE is specified, it overrides the geometric information and defines the resistance. If MNAME is specified, then the resistance may be calculated from the process information in the model MNAME and the given LENGTH and WIDTH. If VALUE is not specified, then MNAME and LENGTH must be specified. If WIDTH is not specified, then it is taken from the default width given in the model. The (optional) TEMP value is the temperature at which this device is to operate, and overrides the temperature specification in the Analog Options dialog box.

Model Parameters (R)

The resistor model consists of process-related device data that allow the resistance to be calculated from geometric information and to be corrected for temperature. The parameters available are:

name	parameter	units	default
TC1	first order temperature coeff.	Ω/°C	0.0
TC2	second order temperature coeff.	$\Omega/°C^2$	0.0
RSH	sheet resistance	Ω/square	-
DEFW	default width	meters	1e-6
NARROW	narrowing due to side etching	meters	0.0
TNOM	parameter measurement temperature	°C	27

The sheet resistance is used with the narrowing parameter and L and W from the resistor device to determine the nominal resistance by the formula:

$$R = RSH\,(L - NARROW) / (W - NARROW)$$

DEFW is used to supply a default value for W if one is not specified for the device. If either RSH or L is not specified, then the standard default resistance value of 1kW is used. TNOM is used to override the circuit-wide value given in Analog Options where the parameters of this model have been measured at a different temperature. After the nominal resistance is calculated, it is adjusted for temperature by the formula:

$$R(T) = R(T_0)\,[1 + TC1\,(T - T_0) + TC2\,(T - T_0)^2]$$

Capacitors

General Form

```
CXXXXXXX  N+  N-  VALUE  <IC=INCOND>
```

Netlist Example

```
C2  13  0  0.1UF
C5  7  0  10UF  IC=3V
```

Spice Data Example

```
%D  %1  %2  %V  IC=12V
```

N+ is the positive node and N- is the negative node. VALUE is the capacitance in Farads. The initial condition is the initial (time-zero) value of the capacitor voltage. Initial conditions only apply if the UIC option is enabled for the Transient Analysis. If N+ is at a higher voltage than N-, the current flow through the capacitor is positive. If N- is at a higher voltage than N+, the current flow is negative.

Spice uses perfect capacitors, that is, capacitors with no DC leakage. Since all nodes in a circuit must have a DC path to ground, you cannot simulate a circuit with capacitors in series as this would completely isolate the nodes that are between the capacitors. One solution to this problem is to connect a large-value resistor (such as 1-gigaohm) in parallel with each capacitor to account for the leakage resistance of that capacitor.

See Also

Capacitor, Polar Cap, Var Capacitor (example circuit: 555.CKT)

Semiconductor Capacitors

General Form

```
CXXXXXXX N+ N- <VALUE> <MNAME> <L=LENGTH> <W=WIDTH> <IC=INCOND>
```

Netlist Example

```
C2  13  0  0.1UF
C5  7  0  10UF  CMOD  L=12u  W=1u
```

Spice Data Example

```
%D  %1  %2  %M  L=10U
```

Spice Model Example

```
.MODEL  CMOD  C(CJ=120E-12  CJSW=200E-6)
```

This is the general form of the Capacitor and allows calculation of the capacitance value from strictly geometric information and the specifications of the process. If VALUE is specified, it defines the capacitance. If MNAME is specified, then the capacitance is calculated from the process information in the model MNAME and the given LENGTH and WIDTH. If VALUE is not specified, then MNAME and LENGTH must be specified. If WIDTH is not specified, then it is taken from the default width given in the model. Specify VALUE or MNAME, LENGTH, and WIDTH (not both).

Model Parameters (C)

The resistor model contains process information that may be used to compute the capacitance from strictly geometric information.

name	parameter	units	default
CJ	junction bottom capacitance	F/meters^2	-
CJSW	junction sidewall capacitance	F/meters	-
DEFW	default device width	meters	1e-6
NARROW	narrowing due to side etching	meters	0.0

The capacitor has a capacitance computed as:

$$\begin{aligned} \text{CAP} = \; & \text{CJ (LENGTH-NARROW) (WIDTH-NARROW)} + \\ & \text{2 CJSW (LENGTH+WIDTH - 2 NARROW)} \end{aligned}$$

Inductors

General Form

```
LYYYYYYY  N+  N-  VALUE  <IC=INCOND>
```

Netlist Example

```
L3  12  9  1UH
L4  5  0  100UH   IC=12.3MA
```

Spice Data Example

```
%D  %1  %2  %V  IC=5MA
```

N+ is the positive node and N- is the negative node. VALUE is the inductance in Henries. Initial condition (which applies only if UIC is enabled for Transient Analysis) is the initial (time-zero) value of the inductor current. If N+ is at a higher voltage than N-, the current flow through the inductor is positive. If N- is at a higher voltage than N+, the current flow is negative.

Inductors induce voltage across their coils based on the amount of magnetic flux; consider them as voltage sources (and therefore inductors) in Spice. Don't connect them directly in parallel. Solution: connect a small-value resistor, (such as 0.001 ohms) in series with each inductor to account for the winding resistance of that inductor.

See Also

Inductor, Var Inductor, Coil 3T, Coil 5T (see RESONANT.CKT)

Coupled (Mutual) Inductors

General Form

```
KXXXXXXX  LYYYYYYY  LZZZZZZZ  VALUE
```

Netlist Examples

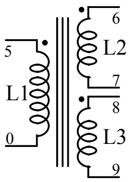
```
K12  L4  L3  0.999
KXFRMR  L1  L2  0.87
```

Spice Data Example (center-tap inductor)

`%DA  %1  %2  50UH`	(inductor A)
`%DB  %2  %3  50UH`	(inductor B)
`K%D  %DA  %DB  .85`	(inductive coupling)

LYYYYYYY and LZZZZZZZ are the names of two coupled inductors, and VALUE is the coefficient of coupling (K), which must be greater than 0 and less than or equal to 1. Using the "dot" convention, place a "dot" on the first node of each inductor, indicating that the voltages at these node are in phase (the voltages go up and down together).

If more than two inductors are being coupled, Spice data must be provided for each coupling. For example, a transformer with one primary coil (L1) and two secondary coils (L2 and L3) might be expressed as follows:



```
L1  5  0  10MH
L2  6  7  1MH
L3  8  9  1MH
K12  L1  L2  0.93
K13  L1  L3  0.93
K23  L2  L3  0.97
```

Note*: A transformer that is simulated in this manner will not reflect the impedance of the secondary winding back into the primary.*

The turns ratio for a given pair of windings can be determined by the following formula where LP and LS are the inductance of the primary and secondary windings, respectively:

```
Turns  Ratio  =  Sqrt(LS/LP)
```

See Also
Transformers (example circuit: VTPWRAMP.CKT)

Voltage/Current Controlled Switches
General Form
```
SXXXXXXX  N+  N-  NC+  NC-  MODEL  <ON><OFF>
WXXXXXXX  N+  N-  VNAM  MODEL  <ON><OFF>
```

Netlist Examples
```
S1  1  2  3  4  SVS1        (V->Switch)
S2  5  6  3  0  SVS2  ON    (V->Switch)
W1  1  2  VS1  WIS1         (I->Switch)
```

Spice Data Example
```
%D  %1  %2  %3  %4  %M  ON
```

Spice Model Example
```
.MODEL  NE2S  SW(RON=1  ROFF=1T  VT=55  VH=15)
```

Nodes 1 and 2 are the nodes between which the switch terminals are connected. ON/OFF indicates the initial condition of the switch. For the voltage-controlled switch nodes, 3 and 4 are the positive and negative controlling nodes, respectively. For the current-controlled switch, the controlling current is that through the specified voltage source. The direction of positive controlling current flow is from the positive node, through the source, to the negative node.

Model Parameters (SW/CSW)

The switch model allows an almost ideal switch to be described in Spice. The switch is not quite ideal, in that the resistance can not change from 0 to infinity, but must always have a finite positive value. By proper selection of the on and off resistances, they can be effectively zero and infinity in comparison to other circuit elements. The parameters available are:

name	**parameter**	**units**	**default**	**switch**
VT	threshold voltage	Volts	0.0	S
IT	threshold current	Amps	0.0	W
VH	hysteresis voltage	Volts	0.0	S
IH	hysteresis current	Amps	0.0	W
RON	on resistance	Ω	1.0	both
ROFF	off resistance	Ω	1/GMIN*	both

*(See GMIN under Analog Options, its default value results in an off-resistance of 1.0e+12 ohms.)

The switch turns **ON** when the controlling voltage or current rises to VT + VH or IT + IH and turns **OFF** when the it falls to VT - VH or IT - IH.

The use of an ideal element that is highly nonlinear such as a switch can cause large discontinuities to occur in the circuit node voltages. A rapid change such as that associated with a switch changing state can cause numerical roundoff or tolerance problems leading to erroneous results or timestep difficulties. The user of switches can improve the situation by taking the following steps:

First, it is wise to set ideal switch impedances just high or low enough to be negligible with respect to other circuit elements. Using switch impedances that are close to “ideal” in all cases aggravates the problem of discontinuities mentioned above. Of course, when modeling real devices such as MOSFETs, the on resistance should be adjusted to a realistic level depending on the size of the device being modeled.

If a wide range of ON to OFF resistance must be used in the switches (ROFF/RON > 1e+12), then the tolerance on errors allowed during transient analysis should be decreased by specifying TRTOL to be less than the default value of 7.0 in the Analog Options. When switches are placed around capacitors, then the option CHGTOL should also be reduced. Suggested values for these two options are 1.0 and 1e-16 respectively. These changes inform Spice3 to be more careful around the switch points so that no errors are made due to the rapid change in the circuit.

See Also

I->Switch, V->Switch (example circuit: SWITCHES.CKT)

Independent Sources

General Form

```
VXXXXXXX N+ N- <<DC> VALUE> <AC <MAG <PHASE>>>
IYYYYYYY N+ N- <<DC> VALUE> <AC <MAG <PHASE>>>
```

Netlist Examples

`VCC 10 0 DC 6`	(V Source)
`ISRC 1 2 AC .3 45 SIN(0 1 1MEG)`	(Signal Gen)
`VMEAS 12 9`	(Ammeter)

Spice Data Example

`%D %1 %2 DC 0 SIN(0 1 1k 0 0) AC 1 0` (Signal Gen)

N+ is the positive node and N- is the negative node. Voltage sources need not be grounded. Positive current is assumed to flow into the positive node, through the source, and out of the negative node. A current source of positive value forces current to flow into the N+ node, through the source, and out of the N- node. Voltage sources, in addition to being used for circuit excitation, are the 'ammeters' for Spice; that is, zero valued voltage sources may be inserted into the circuit for the purpose of measuring current. They, of course, have no effect on circuit operation since they represent short-circuits.

VALUE is the DC (operating point) value or offset of the source. If the source value is zero it may be omitted. If the source is time-invariant (e.g., a power supply), then the value can be preceded by the letters DC.

The letters AC indicate a small-signal AC source. MAG (AC magnitude) and PHASE (AC phase) are used for AC analysis only. If MAG is omitted following the keyword AC, a value of 1 is assumed. If PHASE is omitted, a value of 0 is as-sumed.

Any independent source can be assigned a time-dependent value for Transient Analysis. If a source is assigned a time-dependent value, the time-zero value is used for DC (operating point) analysis. There are five independent source functions: pulse, exponential, sinusoidal, piece-wise linear, and single-frequency FM. These are discussed in the *Multifunction Signal Generator* section of this chapter.

Note: For Spice simulation, voltage sources cannot be placed directly in parallel, and current sources cannot be placed directly in series.

See Also

+V, V Source, I Source, Battery, Signal Gen (example circuit: CEAMP.CKT)

Linear Voltage-Controlled Current Sources

General Form

```
GXXXXXXX  N+  N-  NC+  NC-  VALUE
```

Netlist Example

```
G1  2  0  5  0  0.1MMHO
```

Spice Data Example

```
%D  %1  %2  %3  %4  %V
```

N+ is the positive node and N- is the negative node. Current flow is from the positive node, through the source, to the negative node. NC+ is the positive controlling node and NC- is the negative controlling node. VALUE is the transconductance (in mhos).

Note: For Spice simulation, current sources cannot be placed directly in series.

See Also

V->I Source (example circuit: 741.CKT)

Linear Voltage-Controlled Voltage Sources

General Form

```
EXXXXXXX  N+  N-  NC+  NC-  VALUE
```

Netlist Example

```
E1  2  3  14  1  2.0
```

Spice Data Example

```
%D %1 %2 %3 %4 %V
```

N+ is the positive node and N- is the negative node. Current flow is from the positive node, through the source, to the negative node. VALUE is the voltage gain.

Note: For Spice simulation, voltage sources cannot be placed directly in parallel.

See Also

V->V Source

Linear Current-Controlled Current Sources

General Form

```
FXXXXXXX  N+  N-  VNAM  VALUE
```

Netlist Example

```
F1  5  17  VS2  0.5K
```

Spice Data Example

`V%D %3 %4 DC 0V`	(measures controlling current)
`%D %1 %2 V%D %V`	(current source)

N+ and N- are the positive and negative nodes, respectively. Current flow is from the positive node, through the source, to the negative node. VNAM is the name of a voltage source through which the controlling current flows. The direction of positive controlling current flow is from the positive node, through the source, to the negative node of VNAM. VALUE is the current gain. **Note:** You cannot place current sources directly in series for Spice simulation.

See Also

I->I Source

Linear Current-Controlled Voltage Sources

General Form

```
HXXXXXXX  N+  N-  VNAM  VALUE
```

Netlist Example

```
H1  5  17  VS2  0.5K
```

Spice Data Example

`V%D %3 %4 DC 0V`	(measures controlling current)
`%D %1 %2 V%D %V`	(voltage source)

N+ and N- are the positive and negative nodes, respectively. VNAM is the name of a voltage source through which the controlling current flows. The direction of positive controlling current flow is from the positive node, through the source, to the negative node of VNAM. VALUE is the transresistance (in ohms). **Note:** You cannot place voltage sources directly in parallel for Spice simulation.

See Also

I->V Source. Example circuit: 741.CKT.

Nonlinear Dependent Sources

General Form

```
BXXXXXXX N+ N- <I=EXPR> <V=EXPR>
```

Netlist Examples

```
BSO1 0 1 I=COS(V(1))+SIN(V(2))
BVS1 0 1 V=LN(COS(LOG(V(1,2)^2))) (3)^4+V(2)^V(1)
```

Spice Data Example

```
%D %1 %2 V=%L
```

N+ is the positive node, N- is the negative node. The values of the V and I parameters determine the voltages and currents across and through the device, respectively. If I is given then the device is a current source. If V is given the device is a voltage source. One and only one of these parameters must be specified for each source.

The expressions given for V and I may be any function of voltages and currents through voltages sources in the system (e.g., V(2) indicates the DC voltage at node 2 referenced to ground, V(3,4) indicates the voltage difference between nodes 3 and 4, and I(VS2) indicates the DC current through the voltage supply (VS2). The following functions of real variables are defined:

abs	acos	acosh	asin	asinh	atan
atanh	cos	cosh	exp	ln	log
sin	sinh	sqrt	tan	u	uramp

"u" is the unit step function, with a value of one for arguments greater than zero and a value of zero for arguments less than zero. "uramp" is the integral of the unit step: for an input *x*, the value is zero if *x* is less than zero, or if *x* is greater than zero the value is *x*. These two functions are useful in synthesizing piecewise nonlinear functions, though convergence may be adversely affected.

The following standard operators are defined:

```
+    *    /    ^    unary-
```

In addition, the following boolean operators are defined. Input threshold values (BOOLT) and output values (BOOLL and BOOLH) are universally defined in the Analog Options dialog box.

& (AND)

| (OR)

! (XOR)

~ (NOT)

Older versions of Spice used a POLY function to describe nonlinear sources. For example, the following statements are all equivalent:

```
E1  19  0  POLY(2)7  4  2  0  3  .1  .5          (Spice2)
E1  19  0  POLY(2)(7,4)(2,0)3.1.5                (Spice2)
B1  19  0  V  =  3  +  .1*V(7,4)  +  .5*V(2,0)   (Spice3)
B1  19  0  V  =  3  +  .1*V(7,4)  +  .5*V(2)     (Spice3)
```

Each statement indicates that the voltage at node 19 will equal 3 volts plus .1 times the voltage across nodes 7 and 4 plus .5 times the voltage at node 2, using ground (node 0) as a reference. Many existing Spice subcircuits contain this type of nonlinear source. CircuitMaker automatically converts them to the Spice3 format each time you run a simulation.

To get time into the expression you can integrate the current from a constant current source with a capacitor and use the resulting voltage (don't forget to set the initial voltage across the capacitor). Nonlinear resistors, capacitors, and inductors may be synthesized with the nonlinear dependent source. Nonlinear resistors are obvious. Nonlinear capacitors and inductors are implemented with their linear counterparts by a change of variables implemented with the nonlinear dependent source. The following subcircuit will implement a nonlinear capacitor:

```
.Subckt nlcap pos neg
* Bx: calculate f(input voltage)
Bx   1    0    v = f(v(pos,neg))
* Cx: linear capacitance
Cx   2    0    1
* VxL Ammeter to measure current into the capacitor
Vx   2    1    DC 0Volts
* Drive the current through Cx back into the circuit
Fx   pos  neg  Vx 1
```

Nonlinear inductors are similar.

Note: For Spice simulation, voltage sources cannot be placed directly in parallel and current sources cannot be placed directly in series.

See Also

NLV Source, NLI Source, I-Math, V-Math. Example circuit: 741.CKT

Lossless Transmission Lines

General Form

```
TXXXXXXX  N1  N2  N3  N4  Z0=VALUE
+  <TD=VALUE>  <F=FREQ  <NL=NRMLEN>>
+  <IC=V1,I1,V2,I2>
```

Netlist Example

```
T1  3  0  2  0  Z0=50  TD=20NS
```

Spice Data Example

```
%D  %1  %2  %3  %4  Z0=%V  TD=10NS
```

N1 and N2 are the nodes at port1; N3 and N4 are the nodes at port 2. Z0 is the characteristic impedance. The length of the line may be expressed in either of two forms (one form must be specified). The transmission delay (TD) may be specified directly (as TD=10NS, for example). Alternately, a frequency F may be given, together with NL (the normalized electrical length of the transmission line with respect to the wavelength in the line at the frequency F). If a frequency is specified but NL is omitted, 0.25 is assumed (that is, the frequency is assumed to be the quarter-wave frequency).

The initial condition specification consists of the voltage and current at each of the transmission line ports. Initial conditions only apply if the UIC option is enabled for the Transient Analysis.

The lossy transmission line described below with zero loss may be more accurate than the lossless transmission line due to implementation details.

See Also

LossLessLine (example circuit: LLTRAN.CKT)

Lossy Transmission Lines

General Form

```
OXXXXXXX  N1  N2  N3  N4  MNAME
```

Netlist Example

```
O2  3  0  2  0  OXLINE
```

Spice Data Example

```
%D  %1  %2  %3  %4  %M
```

Spice Model Example

```
.MODEL  OXLINE  LTRA(LEN=0.11  LININTERP=SET)
```

This is a two-port convolution model for single-conductor lossy transmission lines. N1 and N2 are the nodes at port1; N3 and N4 are the nodes at port 2. Note that a lossy transmission line with zero loss may be more accurate than the lossless transmission line due to implementation details.

Model Parameters (LTRA)

The uniform RLC/RC/LC/RG transmission line model (referred to as the LTRA model henceforth) models a uniform constant-parameter distributed transmission line. The RC and LC cases may also be modeled using the URC and TRA models; however, the newer LTRA model is usually faster and more accurate than the others. The operation of the LTRA model is based on the convolution of the transmission line's impulse responses with its inputs.

The LTRA model takes a number of parameters, some of which must be given and some of which are optional.

name	parameter	units	default
R	resistance/length	Ω/unit	0.0
L	inductance/length	henrys/unit	0.0
G	conductance/length	mhos/unit	0.0
C	capacitance/length	farads/unit	0.0
LEN	length of line	arbitrary unit	-
REL	breakpoint control	-	1
ABS	breakpoint control	-	1
NOSTEPLIMIT	don’t limit timestep to less than line delay	flag	not set
NOCONTROL	don’t do complex timestep control	flag	not set
LININTERP	use linear interpolation	flag	not set
MIXEDINTERP	use linear when quadratic seems bad	flag	not set
COMPACTREL	special reltol for history compaction	-	RELTOL
COMPACTABS	special abstol for history compation	-	ABSTOL
TRUNCNR	use Newton-Raphson method for timestep control	flag	not set
TRUNCDONTCUT	don’t limit timestep to keep impulse-response errors low	flag	not set

The following types of lines have been implemented so far: RLC (uniform transmission line with series loss only), RC (uniform RC line), LC (lossless transmission line) and RG (distributed series resistance and parallel conductance only). Any other combination will yield erroneous results and should not be tried. The length LEN of the line must be specified.

NOSTEPLIMIT is a flag that will remove the default restriction of limiting time-steps to less than the line delay in the RLC case. NOCONTROL is a flag that prevents the default limiting of the time-step based on convolution error criteria in the RLC and RC cases. This speeds up simulation but may in some cases reduce the accuracy of results. LININTERP is a flag that, when specified, will use linear interpolation instead of the default quadratic interpolation for calculating delayed signals. MIXEDINTERP is a flag that, when specified, uses a metric for judging whether quadratic interpolation is applicable and if so uses linear interpolation; otherwise is uses the default quadratic interpolation. TRUNCDONTCUT is a flag that removes the default cutting of the time-step to limit errors in the actual calculation of impulse-response related quantities. COMPACTREL and COMPACTABS are quantities that control the compaction of the past history of values stored for convolution. Larger values of these lower accuracy but usually increase simulation speed. These are to be used with the TRYTOCOMPACT Analog Option. TRUNCNR is a flag that turns on the use of Newton-Raphson iterations to determine an appropriate timestep in the timestep control routines. The default is a trial and error procedure by cutting the previous timestep in half. REL and ABS are quantities that control the setting of breakpoints.

The option most worth experimenting with for increasing the speed of the simulation is REL. The default value of 1 is usually safe from the point of view of accuracy but occationally increases computation time. A value greater than 2 eliminates all breakpoints and may be worth trying depending on the nature of the rest of the circuit, keeping in mind that it might not be safe from the viewpoint of accuracy. Breakpoints may usually be entirely eliminated if it is expected the circuit will not display sharp discontinuities. Values between 0 and 1 are usually not required but may be used for setting many breakpoints.

COMPACTREL may also be experimented with when the Analog Option TRYTOCOMPACT is specified. The legal range is between 0 and 1. Larger values usually decrease the accuracy of the simulation but in some cases improve speed. If TRYTOCOMPACT is not specified in Analog Options, history compaction is not attempted and accuracy is high. NOCONTROL, TRUNCDONTCUT and NOSTEPLIMIT also tend to increase speed at the expense of accuracy.

See Also

LossyLine

Uniform Distributed RC Lines (Lossy)

General Form

```
UXXXXXXX  N1  N2  N3  MNAME  L=LEN  <N=LUMPS>
```

Netlist Example

```
U1  1  2  0  UXLINE  L=50UM  N=6
```

Spice Data Example

```
%D  %1  %2  %3  %M  L=25u
```

Spice Model Example

```
.MODEL UXLINE URC(K=1.2 FMAX=6.5MEG)
```

N1 and N2 are the two element nodes the RC line connects, while N3 is the node to which the capacitances are connected. MNAME is the model name, LEN is the length of the RC line in meters. LUMPS, if specified, is the number of lumped segments to use in modeling the RC line (if omitted, a default value based on the model parameters will be used).

Model Parameters (URC)

The URC model is derived from a model proposed by L. Gertzberrg in 1974. The model is accomplished by a subcircuit type expansion of the URC line into a network of lumped RC segments with internally generated nodes. The RC segments are in a geometric progression, increasing toward the middle of the URC line, with K as a proportionality constant. The number of lumped segments used, if not specified for the URC line device, is determined by the following formula:

$$N = \log [F_{max} (R/L) (C/L) 2\pi L^2 \{(K-1)/K\}^2] / \log K$$

The URC line is made up strictly of resistor and capacitor segments unless the ISPERL parameter is given a non-zero value, in which case the capacitors are replaced with reverse biased diodes with a zero-bias junction capacitance equivalent to the capacitance replaced, and with a saturation current of ISPERL amps per meter of transmission line and an optional series resistance equivalent to RSPERL ohms per meter.

name	parameter	units	default
K	propagation constant	-	2.0
FMAX	maximum frequency of interest	Hz	1.0G
RPERL	resistance per unit length	Ω/meter	1000
CPERL	capacitance per unit length	F/meter	1.0e-15
ISPERL	saturation current per unit length	A/meter	0
RSPERL	diode resistance per unit length	Ω/meter	0

See Also

URC-Line

Transistors and Diodes

The area factor used on the diode, BJT, JFET, and MESFET devices determines the number of equivalent parallel devices of a specified model. The affected parameters are marked with an asterisk under the heading ‘area’ in the following model descriptions. Several geometric factors associated with the channel and the drain and source diffusions can be specified on the MOSFET device line.

Two different forms of initial conditions may be specified for some devices. The first form is included to improve the dc convergence for circuits that contain more than one stable state. If a device is specified OFF, the dc operating point is determined with the terminal voltages for that device set to zero. After convergence is obtained, the program continues to iterate to obtain the exact value for the terminal voltages. If a circuit has more than one dc stable state, the OFF option can be used to force the solution to correspond to a desired state. If a device is specified OFF when in reality the device is conducting, the program still obtains the correct solution (assuming the solutions converge) but more iterations are required since the program must independently converge to two separate solutions. The .NODSET control serves a similar purpose as the OFF option. The .NODESET option is easier to apply and is the preferred means to aid convergence.

The second form of initial conditions are specified for use with the transient analysis. These ate true ‘initial conditions’ as opposed to the convergence aids above. See the .IC Statement at the end of this section for a detailed description of initial conditions.

Junction Diodes

General Form

```
DXXXXXXX N+ N- MNAME <AREA> <OFF> <IC=VD> <TEMP=T>
```

Netlist Example

```
D3  2  10  1N914  OFF

D5  7  12  1N4001  3.0  IC=0.2
```

Spice Data Example

```
%D  %1  %2  %M  OFF  IC=.6  TEMP=70
```

Spice Model Example

```
.MODEL  DCLAMP  D(IS=1E-15  IBV=1E-13)
```

N+ is the positive node and N- is the negative node. MNAME is the model name, AREA is the area factor, and OFF indicates an optional starting condition on the device for Operating Point Analysis. The initial condition specification using IC=VD only applies if the UIC option is enabled for the Transient Analysis. The TEMP value is the temperature at which this device is to operate, and overrides the temperature specification in the Analog Options dialog.

Model Parameters (D)

The DC characteristics of the diode are determined by the parameters IS and N. An ohmic resistance, RS is included. Charge storage effects are modeled by transit time, TT, and a nonlinear depletion layer capacitance which is determined by the parameters CJO, VJ, and M. The temperature dependence of the saturation current is defined by the parameters EG, the energy and XTI, the saturation current temperature exponent. The nominal temperature at which these parameters were measured is TNOM, which defaults to the circuit-wide value specified in Analog Options. Reverse breakdown is modeled by an exponential increase in the reverse diode current and is determined by the parameters BV and IBV (both of which are positive numbers).

name	**parameter**	**units**	**default**	**area**
IS	saturation current	A	1.0e-14	*
RS	ohmic resistance	Ω	0	*
N	emission coefficient	-	1	
TT	transit-time	sec	0	
CJO	zero-bias junction capacitance	F	0	*
VJ	junction potential	V	1	
M	grading coefficient	-	0.5	
EG	activation energy	eV	1.11 (Si=1.11, Sbd=0.69, Ge=0.67)	
XTI	saturation-current temp. exp	-	3.0 (jn=3.0, Sbd=2.0)	
KF	flicker noise coefficient	-	0	
AF	flicker noise exponent	-	1	
FC	coefficient for forward-bias depletion capacitance formula	-	0.5	
BV	reverse breakdown voltage	V	infinite	

name	parameter	units	default	area
IBV	current at breakdown voltage	A	1.0e-3	
TNOM	parameter measurement temperature	°C	27	

See Also
Diode, Zener Diode (example circuit: PS1.CKT)

Bipolar Junction Transistors (BJTs)
General Form
```
QXXXXXXX  NC  NB  NE  <NS>  MNAME  <AREA>
+  <OFF>  <IC=VBE,  VCE>  <TEMP=T>
```

Netlist Example
```
Q5  11  26  4  2N3904  IC=0.6,  5.0
Q3  5  2  6  9  QNPN  .67
```

Spice Data Example
```
%D  %1  %2  %3  %M  .67  OFF
```

Spice Model Example
```
.MODEL  BF1234C  NPN(IS=1E-15  BF=150  VKF=200
+       IKF=0.01  CJE=13pF  CJC=10pF  CJS=10pF)
```

NC, NB and NE are the collector, base and emitter nodes, respectively. NS is the optional substrate node; if unspecified, the ground is used. MNAME is the model name, AREA is the area factor, and OFF indicates an optional starting condition on the device for Operating Point Analysis. The initial condition specification using IC=VBE, VCE only applies if you have enabled UIC for the Transient Analysis. TEMP is the temperature at which this device operates, and overrides the specification in the Analog Options dialog.

Model Parameters (NPN/PNP)
The bipolar junction transistor model in Spice is an adaptation of the integral charge control model of Gummel and Poon. This modified Gummel-Poon model extends the original model to include several effects at high bias levels. The model automatically simplifies to the simpler Ebers-Moll model when certain parameters are not specified. The parameter names used in the modified Gummel-Poon model have been chosen to be more easily understood by the program user, and to reflect better both physical and circuit design thinking.

The dc model is defined by parameters IS, BF, NF, ISE, IKF, and NE which determine the forward current gain characteristics, IS, BR, NR, ISC, IKR, and NC which determine the reverse current gain characteristics, and VAF and VAR which determine the output conductance for forward and reverse regions. Three ohmic resistances RB, RC, and RE are included, whre RB can be high current dependent. Base charge storage is modeled by forward and reverse transit times, TF and TR, the forward transit time TF being bias dependent if desired, and nonlinear depletion layer capacitances which are determined by CJE, VJE, and MJE for the B-E junction, CJC, VJC, and MJC for the B-C junction and CJS, VJS, and MJS for the C-S (Collector-Substrate) junction. The temperature dependence of the saturation current, IS, is determined by the energy-gap, EG, and the saturation current temperature exponent XTI. Additionally base current temperature dependence is modeled by the beta temperature exponent XTB in the new model. The values specified are assumed to have been measured at the temperature TNOM, which can be specified in Analog Options or overridden by a specification in the Spice Data field.

The BJT parameters used in the modified Gummel-Poon model are listed below. The parameter names used in earlier versions of Spice2 are still accepted.

Modified Gummel-Poon BJT Parameters

name	**parameter**	**units**	**default**	**area**
IS	transport saturation current	A	1.0e-16	*
BF	ideal maximum forward beta	-	100	
NF	forward current emission coefficient	-	1.0	
VAF	forward Early voltage	V	infinite	
IKF	corner for forward beta high current roll-off	A	infinite	*
ISE	B-E leakage saturation current	A	0	*
NE	B-E leakage emission coefficient	-	1.5	
BR	ideal maximum reverse beta	-	1	
NR	reverse current emission coefficient	-	1	
VAR	reverse Early voltage	V	infinite	
IKR	corner for reverse beta high current roll-off	A	infinite	*
ISC	B-C leakage saturation current	A	0	*
NC	B-C leakage emission coefficient	-	2	
RB	zero bias base resistance	Ω	0	*

name	parameter	units	default	area
IRB	current where base resistance falls halfway to its min value	A	infinite	*
RBM	minimum base resistance at high currents	Ω	RB	*
RE	emitter resistance	Ω	0	*
RC	collector resistance	Ω	0	*
CJE	B-E zero-bias depletion capacitance	F	0	*
VJE	B-E built-in potential	V	0.75	
MJE	B-E junction exponential factor	-	0.33	
TF	ideal forward transit time	sec	0	
XTF	coefficient for bias dependence of TF	-	0	
VTF	voltage describing VBC dependence of TF	V	infinite	
ITF	high-current parameter for effect on TF	A	0	*
PTF	excess phase at freq=1.0/(TF*2π)Hz	deg	0	
CJC	B-C zero-bias depletion capacitance	F	0	*
VJC	B-C built-in potential	V	0.75	
MJC	B-C junction exponential factor	-	0.33	
XCJC	fraction of B-C depletion capacitance connected to internal base node	-	1	
TR	ideal reverse transit time	sec	0	
CJS	zero-bias collector-substrate capacitance	F	0	*
VJS	substrate junction built-in potential	V	0.75	
MJS	substrate junction exponential factor	-	0	
XTB	forward and reverse beta temperature coefficient	-	0	
EG	energy gap for temperature effect on IS	eV	1.11	
XTI	temperature exponent for effect on IS	-	3	
KF	flicker-noise coefficient	-	0	
AF	flicker-noise exponent	-	1	
FC	coefficient for forward-bias depletion capacitance formula	-	0.5	
TNOM	parameter measurement temperature	°C	27	

See Also

NPN Trans, PNP Trans (example circuit: CEAMP.CKT)

Junction Field-Effect Transistors (JFETs)

General Form

```
JXXXXXXX ND NG NS MNAME <AREA> <OFF> <IC=VDS, VGS> <TEMP=T>
```

Netlist Example

```
J2  6  3  21  2N3819  OFF
```

Spice Data Example

```
%D  %1  %2  %3  %M  .67
```

Spice Model Example

```
.MODEL  2N1234  NJF(VTO=-1  RD=100  RS=90  CGS=5pF)
```

ND, NG and NS are the drain, gate and source nodes, respectively. MNAME is the model name, AREA is the area factor, and OFF indicates an optional initial condition on the device for Operating Point Analysis. The initial condition specification using IC=VDS, VGS only applies if you have enabled UIC for the Transient Analysis. TEMP is the temperature at which this device operates, and overrides the temperature specification in the Analog Options dialog.

Model Parameters (NJF/PJF)

The JFET model is derived from the FET model of Shichman and Hodges. The dc characteristics are defined by the parameters VTO and BETA, which determine the variation of drain current with gate voltage, LAMBDA, which determines the output conductance, and IS, the saturation current of the two gate junctions. Two ohmic resistances, RD and RS, are included. Charge storage is modeled by nonlinear depletion layer capacitances for both gate junctions which vary as the –1/2 power of junction voltage and are defined by the parameters CGS, CGD, and PB.

Note that in Spice3f and later, a fitting parameter B has been added.

name	parameter	units	default	area
VTO	threshold voltage	V	-2.0	
BETA	transconductance parameter (β)	A/V^2	1.0e-4	*
LAMBDA	channel-length modulation parameter (λ)	1/V	0	
RD	drain ohmic resistance	Ω	0	*
RS	source ohmic resistance	Ω	0	*
CGS	zero-bias G-S junction capacitance (C_{gs})	F	0	*
CGD	zero-bias G-D junction capacitance (C_{gd})	F	0	*
PB	gate junction potential	V	1	

name	parameter	units	default	area
IS	gate junction saturation current (Is)	A	1.0e-14	*
B	doping tail parameter	-	1	
KF	flicker noise coefficient	-	0	
AF	flicker noise exponent	-	1	
FC	coefficient for forward-bias depletion capacitance formula	-	0.5	
TNOM	parameter measurement temperature	°C	27	

See Also

N-JFET, P-JFET. Example circuit: CSJFAMP.CKT

MOSFETs

General Form

```
MXXXXX ND NG NS NB MNAME <L=VAL> <W=VAL> <AD=VAL>
+ <AS=VAL> <PD=VAL> <PS=VAL> <NRD=VAL> <NRS=VAL> <OFF>
+ <IC=VDS,VGS,VBS> <TEMP=T>
```

Netlist Example

```
M6  23  16  0  17  MRF150
```

Spice Data Example

```
%D  %1  %2  %3  %3  %M  TEMP=55
```

Spice Model Example

```
.MODEL  MOSMOD1  NMOS(LEVEL=1  VTO=1  RD=1  RS=1
+   IS=1E-15  KF=1.0E-26  CJ=2.0E-4  CJSW=1.0E-9
+   KP=3.1E-5  PB=0.87)
```

ND, NG, NS and NB are the drain, gate, source and bulk (substrate) nodes, respectively. MNAME is the model name. L and W are the channel length and width, in meters. AD and AS are the areas of the drain and source diffusions, in meters^2. Note that the suffix U specifies microns (1e-6 m) and P sq-microns (1e-12 m^2). If any of L, W, AD, or AS are not specified, default values are used. The use of defaults simplifies input file preparation, as well as the editing required if device geometries are to be changed. PD and PS are the perimeters of the drain and source junctions, in meters. NRD and NRS designate the equivalent number of squares of the drain and source diffusions; these values multiply the sheet resistance (RSH) specified in the model for an accurate representation of the parasitic series drain and source resistance of each transistor. PD and PS default to 0.0 while NRD and NRS default to 1.0. OFF indicates an optional starting condition on the device for DC analysis. The initial condition specification (optional) using IC=VDS, VGS, VBS only applies if the UIC option is enabled for the Transient Analysis, when a transient analysis is desired starting from other than the quiescent operating point. See the .IC device for a better and more convenient way to specify transient initial conditions.

The TEMP value (optional) is the temperature at which this device is to operate, and overrides the temperature specification in the Analog Options dialog. The temperature specification is ONLY valid for level 1, 2, 3, and 6 MOSFETs, not for level 4 or 5 (BSIM) devices.

Model Parameters (NMOS/PMOS)

Spice provides four MOSFET device models, which differ in the formulation of the I-V characteristic. The variable LEVEL specifies the model to be used:

LEVEL=1	Shichman-Hodges
LEVEL=2	MOS2
LEVEL=3	MOS3, a semi-empirical model
LEVEL=4	BSIM
LEVEL=5	new BSIM (BSIM2)
LEVEL=6	MOS6

The dc characteristics of the level 1 through level 3 MOSFETs are defined by the device parameters VTO, KP, LAMBDA, PHI and GAMMA. These parameters are computed by Spice if process parameters (NSUB, TOX, …) are given, but user-specified values always override. VTO is positive (negative) for enhancement mode and negative (positive) for depletion mode N-channel (P-channel) devices. Charge storage is modeled by three constant capacitors, CGSO, CGDO, and CGBO which represent overlap capacitances, by the nonlinear thin-oxide capacitance which is distributed among the gate, source, drain, and bulk regions, and by the nonlinear depletion-layer capacitances for both substrate junctions divided into bottom and periphery, which vary as the MJ and MJSW power of junction voltage respectively, and are determined by the parameters CBD, CBS, CJ, CJSW, MJ, MJSW and PB. Charge storage effects are modeled by the piecewise linear voltages-dependent capacitance model proposed by Meyer. The thin-oxide charge-storage effects are treated slightly different for the

LEVEL=1 model. These voltage-dependent capacitances are included only if TOX is specified in the input description and they are represented using Meyer's formulation.

There is some overlap among the parameters describing the junctions, e.g. the reverse current can be input either as IS (in A) or as JS (in A/m^2). Whereas the first is an absolute value the second is multiplied by AD and AS to give the reverse current of the drain and source junctions respectively. This methodology has been chosen since there is no sense in relating always junction characteristics with AD and AS entered on the device line; the areas can be defaulted. The same idea applies also to the zero-bias junction capacitances CBD and CBS (in F) on one hand, and CJ (in F/m^2) on the other. The parasitic drain and source resistance can be expressed as either RD and RS (in ohms) or RSH (in ohms/sq.), the latter being multiplied by the number of squares NRD and NRS input on the device line.

A discontinuity in the MOS level 3 model with respect to the KAPPA parameter has been detected. The supplied fix has been implemented in Spice3f2 and later. Since this fix may affect parameter fitting, the option "BADMOS3" may be set to use the old implementation (see Analog Options).

Spice level 1, 2, 3 and 6 parameters:

name	**parameter**	**units**	**default**
LEVEL	model index	-	1
VTO	zero-bias threshold voltage (V_{TO})	V	0.0
KP	transconductance parameter	A/V^2	2.0e-5
GAMMA	bulk threshold parameter (γ)	V$^{1/2}$	0.0
PHI	surface potential (ϕ)	V	0.6
LAMBDA	channel-length modulation (MOS1 and MOS2 only) (λ)	1/V	0.0
RD	drain ohmic resistance	Ω	0.0
RS	source ohmic resistance	Ω	0.0
CBD	zero-bias B-D junction capacitance	F	0.0
CBS	zero-bias B-S junction capacitance	F	0.0
IS	bulk junction saturation current (Is)	A	1.0e-14
PB	bulk junction potential	V	0.8
CGSO	gate-source overlap capacitance per meter channel width	F/m	0.0

name	parameter	units	default
CGDO	gate-drain overlap capacitance per meter channel width	F/m	0.0
CGBO	gate-bulk overlap capacitance per meter channel width	F/m	0.0
RSH	drain and source diffusion sheet resistance	Ω/sq.	0.0
CJ	zero-bias bulk junction bottom cap. per sq-meter of junction area	F/m^2	0.0
MJ	bulk junction bottom grading coeff.	-	0.5
CJSW	zero-bias bulk junction sidewall cap. per meter of junction perimeter	F/m	0.0
MJSW	bulk junction sidewall grading coeff.	-	0.50 (level 1) 0.33 (level 2, 3)
JS	bulk junction saturation current per sq-meter of junction area	A/m^2	
TOX	oxide thickness	meter	1.0e-7
NSUB	substrate doping	$1/cm^3$	0.0
NSS	substrate state density	$1/cm^2$	0.0
NFS	fast surface state density	$1/cm^2$	0.0
TPG	type of gate material +1 opp. to substrate -1 same as substrate 0 A1 gate	-	1.0
XJ	metallurgical junction depth	meter	0.0
LD	lateral diffusion	meter	0.0
UO	surface mobility	cm^2/Vs	600
UCRIT	critical field for mobility degradation (MOS2 only)	V/cm	1.0e4
UEXP	critical field exponent in mobility degradation (MOS2 only)	-	0.0
UTRA	transverse field coeff. (mobility) (deleted for MOS2)	-	0.0
VMAX	maximum drift velocity of carriers	m/s	0.0
NEFF	total channel-charge (fixed and mobile) coefficient (MOS2 only)	-	1.0
KF	flicker noise coefficient	-	0.0
AF	flicker noise exponent	-	1
FC	coefficient for forward-bias depletion capacitance formula	-	0.5

name	parameter	units	default
DELTA	width effect on threshold voltage (MOS2 and MOS3)	-	0.0
THETA	mobility modulation (MOS3 only)	1/V	0.0
ETA	static feedback (MOS3 only)	-	0.0
KAPPA	saturation field factor (MOS3 only)	-	0.2
TNOM	parameter measurement temperature	°C	27

The level 4 and level 5 (BSIM1 and BSIM2) parameters are all values obtained from process characterization, and can be generated automatically. Parameters marked below with an * in the l/w column also have corresponding parameters with a length and width dependency. For example, VFB is the basic parameter with units of Volts, and LVFB and WVFB also exist and have units of Volt-mmeter. The formula

$$P = P_0 + (P_L / L_{effective}) + (P_W / W_{effective})$$

is used to evaluate the parameter for the actual device specified with

$$L_{effective} = L_{input} - DL$$

and

$$W_{effective} = W_{input} - DW$$

Note that unlike the other models in Spice, the BSIM model is designed for use with a process characterization system that provides all the parameters, thus there are no defaults for the parameters, and leaving one out is considered an error.

Spice BSIM (level 4) parameters:

name	parameter	units	l/w
VFB	flat-band voltage	V	*
PHI	surface inversion potential	V	*
K1	body effect coefficient	$V^{1/2}$	*
K2	drain/source depletion charge sharing coefficient	-	*
ETA	zero-bias drain-induced charge-sharing coefficient	-	*
MUZ	zero-bias mobility	cm^2/V-s	
DL	shortening of channel	µm	

name	parameter	units	l/w
DW	narrowing of channel	µm	
U0	zero-bias transverse-field mobility degradation coefficient	V^{-1}	*
U1	zero-bias velocity saturation coefficient	µm/V	*
X2MZ	sens. of mobility to substrate bias at V_{ds}=0	cm^2/V^2-s	*
X2E	sens. of drain-induced barrier lowering effect to substrate bias	V^{-1}	*
X3E	sens. of drain-induced barrier lowering effect to drain bias at $V_{ds}=V_{dd}$	V^{-1}	*
X2U0	sens. of transverse field mobility degradation effect to substrate bias	V^{-2}	*
X2U1	sens. of velocity saturation effect to substrate bias	μmV^{-2}	*
MUS	mobility at zero substrate bias and at $V_{ds}=V_{dd}$	cm^2/V^2-s	
X2MS	sens. of mobility to substrate bias at $V_{ds}=V_{dd}$	cm^2/V^2-s	*
X3MS	sens. of mobility to drain bias at $V_{ds}=V_{dd}$	cm^2/V^2-s	*
X3U1	sens. of velocity saturation effect on drain bias at $V_{ds}=V_{dd}$	μmV^{-2}	*
TOX	gate oxide thickness	µm	
TEMP	parameter measurement temperature	°C	
VDD	measurement bias range	V	
CGDO	gate-drain overlap capacitance per meter channel length	F/m	
CGSO	gate-source overlap capacitance per meter channel width	F/m	
CGBO	gate-bulk overlap capacitance per meter channel length	F/m	
XPART	gate-oxide capacitance-charge model flag	-	
N0	zero-bias subthreshold slope coefficient	-	*
NB	sens. of subthreshold slope to substrate bias	-	*
ND	sens. of subthreshold slope to drain bias	-	*
RSH	drain and source diffusion sheet resistance	Ω/sq.	
JS	source drain junction current density	A/m^2	
PB	built-in potential of source-drain junction	V	
MJ	grading coeff. of source-drain junction	-	
PBSW	built-in potential of source-drain junction sidewall	V	
MJSW	grading coefficient of source-drain junction sidewall	-	

name	parameter	units	l/w
CJ	source-drain junction capacitance per unit area	F/m^2	
CJSW	source-drain junction sidewall capacitance per unit length	F/m	
WDF	source-drain junction default width	m	
DELL	source-drain junction length reduction	m	

XPART=0 selects a 40/60 drain/source charge partition in saturation, while XPART=1 selects a 0/100 drain/source charge partition.

ND, NG, and NS are the drain, gate, and source nodes, respectively. MNAME is the model name, AREA is the area factor, and OFF indicates an (optional) initial condition on the device for operating point analysis. If the area factor is omitted, a value of 1.0 is assumed. The (optional) initial condition specification, using IC=VDS, VGS is intended for use with the UIC option on in the Transient Analysis setup when transient analysis is desired starting from other than the quiescent operating point. See the .IC statement for a better way to set initial conditions.

See Also

N-MOSFET 3T, N-MOSFET 4T, P-MOSFET 3T, P-MOSFET 4T

MESFETs (GaAsFETs)

General Form

```
ZXXXXXXX ND NG NS MNAME <AREA> <OFF> <IC=VDS, VGS>
```

Netlist Example

```
Z1  3  5  6  ZM2  OFF
```

Spice Data Example

```
%D  %1  %2  %3  %M  OFF
```

Spice Model Example

```
.MODEL  ZM2  NMF(VTO=-1.0  RD=1.0E-4  RS=100)
```

ND, NG and NS are the drain, gate and source nodes, respectively. MNAME is the model name, AREA is the area factor, and OFF indicates an optional starting condition on the device for Operating Point Analysis. The initial condition specification using IC=VDS, VGS only applies if the UIC option is enabled for the Transient Analysis.

Model Parameters (NMF/PMF)

The MESFET model is derived from the GaAs FET model of Statz et al. The dc characteristics are defined by the parameters VTO, B, and BETA, which determine the variation of drain current with gate voltage, ALPHA, which determines saturation voltage, and LAMBDA, which determines the output conductance. The formula are given by:

$$I_d = [\beta(V_{gs}-V_T)^2 / (1 + b(V_{gs}-V_T))]\,[1 - [1-\alpha\,(V_{ds}/3)]^3]\,(1+ \lambda\, V_{ds})$$
$$\text{for } 0 < V_{ds} < 3/\alpha$$

$$I_d = [\beta(V_{gs}-V_T)^2 / (1+b(V_{gs}-V_T))]\,(1 + \lambda\, V_{ds})$$
$$\text{for } V_{ds} > 3/\alpha$$

Two ohmic resistances, RD and RS, are included. Charge storage is modeled by total gate charge as a function of gate-drain and gate source voltages and is defined by the parameters CGS, CGD, and PB.

name	parameter	units	default	area
VTO	pinch-off voltage	V	-2.0	
BETA	transconductance parameter	A/V^2	1.0e-4	*
B	doping tail extending parameter	1/V	0.3	*
ALPHA	saturation voltage parameter	1/V	2	*
LAMBDA	channel-length modulation parameter	1/V	0	
RD	drain ohmic resistance	Ω	0	*
RS	source ohmic resistance	Ω	0	*
CGS	zero-bias G-S junction capacitance	F	0	*
CGD	zero-bias G-D junction capacitance	F	0	*
PB	gate junction potential	V	1	
KF	flicker noise coefficient	-	0	
AF	flicker noise exponent	-	1	
FC	coefficient for forward bias depletion capacitance formula	-	0.5	

See Also

N-MESFET, P-MESFET

Subcircuits

General Form

```
XYYYYYYY  N1  <N2  N3  …>  SUBNAM
```

Netlist Example

```
XU1  7  5  6  FILTER
```

Spice Data Example

```
%D  %1  %2  %3  %S
```

Spice Subcircuit Example

```
.SUBCKT  FILTER  1  2  3
R1  1  2  1k
C1  2  3  1uF
.ENDS  FILTER
```

Subcircuits are used in Spice by specifying the device designation beginning with the letter X, followed by the circuit nodes to be used in expanding the subcircuit, followed by the subcircuit name.

See Also

Subcircuits (example circuit: ANALOG.CKT)

SimCode™ Devices

General Form

```
AXXXXXXX  [NPI  NGI  NI1I  <NI2I  …>][NPO
+  NI1O  <NI2O  …>  NO1O  <NO2O  …>]  MNAME
```

Netlist Example

```
A2  [6  8  4]  [7  9  5]  2404B
```

Spice Data Example

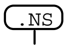
```
%D  [%4bi  %2bi  %1i]  [%4bo  %1o  %2o]  %M
```

All nodes listed in a digital SimCode device are the digital nodes of the device. NPI and NGI are the digital input nodes to the power and ground pins. NI1I, NI2I, etc. are the digital input nodes to the device's input pins. NPO is the digital output node from the power pin. NI1O, NI2O, etc. are the digital output nodes from the device's input pins. NO1O, NO2O, etc. are the digital output nodes from the device's output pins. Note the square brackets ([ ]) surrounding the input nodes and the output nodes. In the Spice Data field, the power and ground buses include the letter "b" to indicate that node numbers for these pins come from the Bus Data field.

See Also

Digital SimCode Devices. Example circuit: Logic Probe.ckt. See *Chapter 17: Digital SimCode.*

.NODESET Statement



Nodeset Device

General Form

```
.NODESET V(NODNUM)=VAL V(NODNUM)=VAL ...
```

Netlist Example

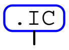
```
.NODESET V(7)=3.33 V(11)=1.5
```

The Nodeset line helps the program find the dc or initial transient solution by making a preliminary pass with the specified nodes held to the given voltages. The restriction is then released and the iteration continues to the true solution. The .NODESET line may be necessary for convergence on bistable or astable circuits. In general, this line should not be necessary.

See Also

.NODESET

.IC Statement



Initial Condition Device

General Form

```
.IC V(NODNUM)=VAL V(NODNUM)=VAL ...
```

Netlist Example

```
.IC V(3)=2.5 V(4)=-1 V(9)=1
```

The IC line is for setting transient initial conditions. It has two different interpretations, depending on whether or not you have enabled the UIC parameter in the Transient Analysis. Also, don't confuse this line with the .NODESET line. The .NODESET line is only to help DC convergence, and does not affect final bias solution (except for multi-stable circuits). The two interpretations of this line are as follows:

- When you have enabled the UIC parameter in the Transient Analysis, the node voltages specified on the .IC control line are used to compute the capacitor, diode, BJT, JFET, and MOSFET initial conditions. This is equivalent to specifying the IC=… parameter on each device line, but is much more convenient. You can still specify the IC=… parameter, which takes precedence over the .IC values. Since no DC bias (initial transient) solution is computed before the Transient Analysis, you should take care to specify all DC source voltages on the .IC control line if you are going to use them to compute device initial conditions.

- If you have *not enabled* the UIC parameter in the Transient Analysis, the DC bias (initial transient) solution is computed before the Transient Analysis. In this case, the node voltages you have specified on the .IC control line are forced to the desired values during the bias solution. The Transient Analysis removes the constraint on these node voltages. This is the preferred method since it allows Spice to compute a consistent DC solution.

See Also
.IC (example circuit: 555.CKT)

Suggested Reading

Tuinenga, P. W., *SPICE, A guide to Circuit Simulation & Analysis Using PSpice*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1988, ISBN: 0-13-834607-0, Library: TK454.T85 1988, 621.319'2-dc 19

Written as a supplement for electronic circuit design courses, this book focuses on the design and analysis of analog circuits using PSpice (a commercial variation of the industry standard Berkeley Spice). Through examples, this book demonstrates what a simulator can and cannot do. Although this book is written specifically for PSpice, much of the information it contains can be applied directly to CircuitMaker.

Vladimirescu, A., *The SPICE Book*, John Wiley & Sons, Inc., N.Y., 1994, ISBN: 0-471-60926-9, Library: TK454.V58 1994, 621.319'2'028553-dc20

Written as a tutorial and reference for electrical engineering students and professionals just starting to use the Spice program to analyze and design circuits. This book explains how to use the Spice program and describes the differences and similarities between the most popular commercial versions of it, including Spice3, the latest version from Berkeley which is used by CircuitMaker.

Kielkowski, R., *Inside SPICE*, McGraw-Hill, Inc., N.Y., 1994, ISBN: 0-070911525, Library: TK 454.K48 1994, 621.319'2'011353-dc20

Written as a tutorial and reference for electrical engineering students and professionals who are familiar with the Spice program. This book goes beyond the basics and covers the internal operation of the Spice program to give the reader a solid understanding of how Spice works. It provides step-by-step coverage of how to overcome nonconvergence, numerical integration instabilities and timestep control errors. It also shows how to make simulations run faster and more efficiently by setting the .OPTION parameters.

CHAPTER 17

Creating New Devices

The devices you can create in CircuitMaker fall into two major categories: 1) nonfunctional device symbols that you simply want to represent schematically or export to a PCB netlist and 2) fully functional devices that will simulate. Simulation functionality is achieved by attaching internal circuitry or a Spice model/subcircuit to a device symbol.

The term *macro device* is used loosely throughout this manual to refer to any device you create or modify.

What's In This Chapter?

The following summarizes the information contained in this chapter.

- **Creating Device Symbols**: Learn how to use the Symbol Editor to create or edit custom device symbols. Learn how to draw symbols freehand with the mouse, create new symbols based on an existing symbol and quickly create a DIP, LCC or QFP symbols. Learn how pins are used as connection points for wires. Also learn how to prepare the symbol for use in CircuitMaker by adding the default Device Properties to the symbol. Go through a step-by-step example to create your own device symbol.
- **Creating Macro Devices with Internal Circuitry**: Learn how to attach hidden internal circuitry to a custom device symbol. Learn how this simple process can be used to expand the selection of simulatable devices in CircuitMaker. Learn how macro devices can be nested for hierarchical construction of a circuit for simulation. Go through a step-by-step example to attach circuitry to a symbol, thus making a functional macro device.
- **Working with Spice Models**: Learn about the three basic types of components in Spice, selecting a Spice model for simulation, modifying Spice models, and how

to attach a new Spice model to a device symbol. Go through a step-by-step example to add a new Spice model.

- **Creating New Spice Models Using Parameter Passing**: Learn how to create a generic Spice model which can be used to simulate any number of like components by passing parameters specific to each device into the generic model.
- **Editing Digital Model Parameters**: Learn about Digital SimCode devices, their various parameters, and how to modify them.

Creating Device Symbols

You can create or modify device symbols in one or more of the following ways:

- Drawing a symbol with the mouse.
- Entering a description in the Element List.
- Adding existing shapes.
- Importing a Metafile.
- Adding DIP, LCC, and QFP Packages.

To create a symbol for a new macro,

1 Clear the drawing area by clicking the **New** button on the Toolbar.

2 Choose **Macros** > **New Macro**.

3 Enter a unique name for the macro of 13 characters or less.

The Macro Name is used to identify the macro device in the library.

4 Specify how many of these parts would be found in a single IC package.

Note: The same symbol is used for each device in the package.

5 Click OK to display the Symbol Editor (Figure 17.1).

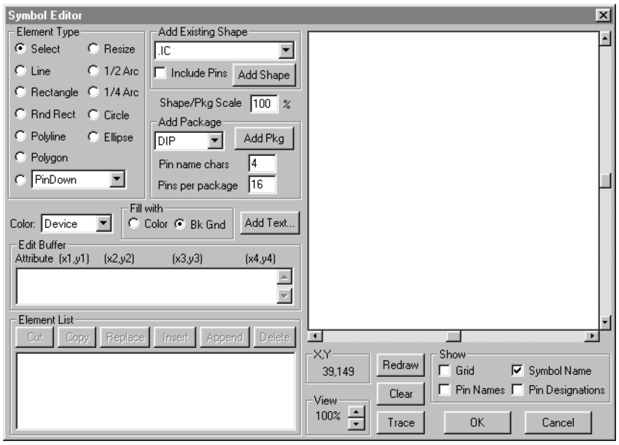


Figure 17.1. Use the Symbol Editor to create or modify schematic symbols.

Think of the Symbol Editor as a drawing program. The following sections describe the use of the Symbol Editor options.

Using Symbol Editor Display Controls

Once you have placed a device symbol in the View window, use the Symbol Editor options to control the view of the symbol. You can click and drag device designations at any time to position them where you want.

Control	**What it Does**
Redraw	Refreshes the picture.
Clear	Erases the picture without deleting any elements.
Trace	Step through the Element List, highlighting each element one at a time beginning with the currently selected element. Select the first element in the list, click the **Clear** button to erase the drawing window, and then click the **Trace** button repeatedly. This helps you identify elements that are hidden behind other elements, etc.

Grid	Displays or hides the currently defined grid in the View window. We recommend a 9 point spacing for pin placement. **Note:** The Symbol Editor does not use the Snap To Grid feature. See *Chapter 12: Options Menu* about grid setup and changing the grid color.
Symbol Name	Displays or hides the device's symbol name.
Pin Names	Displays or hides the device's pin names.
Pin Designations	Displays or hides the device's pin designations.
View	Zoom in or out (from 25%-800%) to better view the device while you are working on it.

Drawing a Symbol with the Mouse

You can draw symbols or parts of symbols freehand with the mouse.

To draw a symbol element with the mouse,

1 Select the color and fill.

Any enclosed element will be filled with either the element color or the background color.

2 Select an element type by clicking its radio button.

You can draw the following element types: **Line**, **1/4 Arc**, **1/2 Arc**, **Circle**, **Ellipse**, **Polyline**, **Polygon**, **Rectangle**, **Round Rect**, and **Pins**.

Pins can point up, down, left or right and can have bubbles to indicate negative logic. Pins with bubbles are indicated with a tilde (~).

Note: Any device symbol you want to wire into a circuit must have pins as connection points for wires.

3. To place a line, arc, circle, ellipse, rectangle, or pin, hold down the left mouse button, drag, and release. To place a polyline or polygon, click and release the mouse to begin then single-click to turn; double-click to end.

   Note: After you place a pin, a dialog box appears that lets you enter a pin name and designation. Pin names and designations are required for circuit simulation and PCB netlist generation.

When you place an element, its description is appended to the Element List. The element's description and shape are also selected and highlighted, which lets you easily move or delete elements that you have placed. After you have placed an element, the selected Element Type does not change until you select a new Element Type or click on a text line in the Element List.

Selecting Shapes

To select single shapes,

1. Choose the **Select/Move** option in the Element Type group box.
2. Click the shape in the view window.

   OR

   Click the description in the Element List.

To select multiple shapes,

1. Select the **Select/Move** option in the Element Type group box.
2. Drag a selection rectangle around the desired element shape in the view window.

   OR

   Click and drag the cursor in the Element List.

   OR

   Hold down the **Shift** key while clicking individual element shapes in the view window or element descriptions in the Element List.

To delete elements,

1 Select the elements you want to delete.

2 Click the **Delete** button.

To move elements,

1 Select the **Select/Move** option in the Element Type group box.

2 Drag the element shape with the mouse while holding down the **Left** mouse button.

OR

If you have selected multiple elements, drag one of the elements with the mouse while holding down the **Left** mouse button.

To resize elements,

1 Select the **Resize** option in the Element Type group box.

2 Drag the end of a line, the corner of a rectangle, the side of a rounded rectangle, or the side of an ellipse with the mouse while holding down the **Left** mouse button.

Note: You must use the Element List or Element Buffer to change the size and shape of other elements. See *Element List and Edit Buffer* later in this chapter for more information.

Adding an Existing Shape

To make it easier to create a new device symbol, you can use and modify existing device shapes as needed.

To add an existing shape,

1 In the Symbol Editor dialog box, click the **Add Existing Shape** drop-down list box.

2 Select the name of the existing shape you want to add.

3 If you want to include pins for the existing device, check the **Include Pins** check box.

4 If you want to change the size of the shape you are adding, enter a value in the **Scale** edit box.

5 Click the **Add Shape** button.

6 Press the **r** key (or click the **Right** mouse button) to rotate the shape.

7 Press the **m** key to mirror the shape.

8 Move the shape to the desired location in the Symbol Editor's view window.

9 Click the **Left** mouse button to place the shape or press the **Spacebar** to cancel.

Note: Only the location (not the size) of included pins will be scaled.

Importing a Metafile Device

The Symbol Editor can import device symbols (pictures) created in another drawing program only if that drawing program can copy the desired symbol to the Windows clipboard in the Metafile format.

To import a Metafile device,

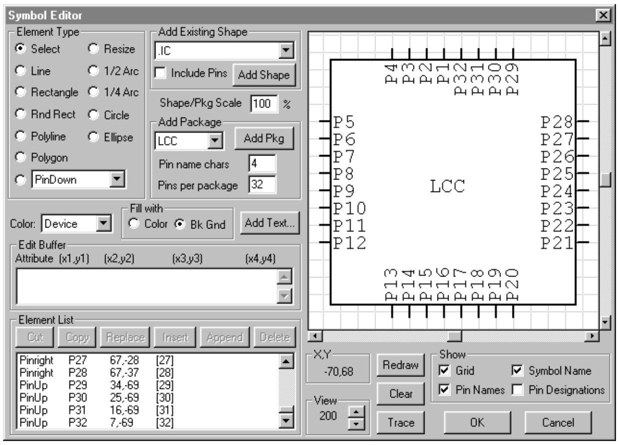
1 Copy the Metafile object onto the Windows clipboard.

2 From the Symbol Editor, select **Clipboard-WMF** as the existing shape (see *Adding an Existing Shape* earlier in this section).

Note: Only the vector graphics shapes (lines, rectangles, circles, etc.) in the Metafile object will be converted. Colors, bitmaps, and text will not be included.

Adding DIP, LCC, and QFP Packages

Use the Symbol Editor to quickly create DIP (Dual In-line Package), LCC and QFP symbols. LCC and QFP are square outlines with an equal number of pins on each side. The LCC symbol has pin 1 on the center of the top side. The QFP symbol has pin 1 on the top of the left side. You can control the size of the package you are adding using the **Scale** value and the number of **Pin name chars** you enter. You can specify the width of the DIP, LCC, and QFP symbols; however, DIPs cannot be scaled directly, they can only be rotated and mirrored. See Figure 17.2 for an example of the LCC symbol.



Add Existing Shape: .IC

Shape/Pkg Scale: 100 %

Add Package: LCC

Pin name chars: 4

Pins per package: 32

Color: Device

Fill with: Color [✓ Bk Gnd]

Element Type: [✓ Select] PinDown

X,Y: -70,68

View: 200

Show: [✓ Grid] [✓ Symbol Name] [✓ Pin Names] Pin Designations

```
Pinright  P27  67,-28  [27]
Pinright  P28  67,-37  [28]
PinUp     P29  34,-69  [29]
PinUp     P30  25,-69  [30]
PinUp     P31  16,-69  [31]
PinUp     P32  7,-69   [32]
```

Figure 17.2. Use the Symbol Editor to quickly add DIP, QFP, and LCC symbols.

To add a DIP, LCC, or QFP symbol,

1. From the Symbol Editor, specify the number of pins (in 2 pin increments) in the **Pins per Pkg** field.
2. Select the Scale (100% by default).
3. Click the **Add Pkg** button.
4. Move the package to the desired location.

   Note: You can press the **Spacebar** to cancel this operation.
5. Click the **Left** mouse button to place the package.

Editing Pin Information

Pin names and pin designations are required for circuit simulation and PCB netlist generation. For compatibility with TraxMaker, each pin designation should match the Pad Designation of the corresponding pad in the TraxMaker component.

To edit pin names and pin designations,

1. In the Symbol Editor, select the **Select/Move** option in the Element Type group box.

Tip: *You can create barred pin names by entering braces { } around all or part of a pin name. Barred pin names work correctly only with monospaced fonts, such as Courier New, CircuitMaker's default font. You can also orient pin names perpendicular or parallel to the pin.*

2 Right-click a pin to display the dialog box pictured in Figure 17.3.

3 Give each pin a name (up to 15 characters) and a designation (up to 5 alphanumeric characters), even if the names are not displayed.

Note: For multipart packages, one pin designation should be listed for each part in the package, separated by commas.

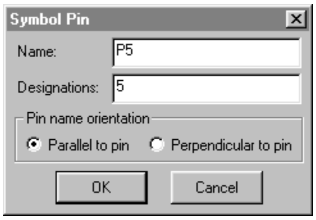

Symbol Pin
Name: P5
Designations: 5
Pin name orientation: [✓ Parallel to pin] Perpendicular to pin

Figure 17.3. Use the Symbol Pin dialog box to edit pin names and designations.

Element List and Edit Buffer

As you add elements to the drawing, they are also added to the Element List. The Element List contains a text description of each element in the device's symbol. The link between the drawing window and the Element List is interactive. When you select an element in the drawing window, it is highlighted in the Element List. Likewise, when you select an element in the Element List, it is highlighted in the drawing window.

Elements at the beginning of the list are drawn first so they are in the back of the drawing (bottom layer); those at the end of the list are drawn last so they are in front (top layer).

Use the Edit Buffer to add or edit the element text descriptions contained in the Element List box. Use the following buttons to edit and change the order of the elements.

Button	**What it Does**
Cut	Removes the selected elements from the Element List and places them in the Edit Buffer.
Copy	Places a copy of the selected elements in the Edit Buffer.
Replace	Replaces the selected elements with the contents of the Edit Buffer.
Insert	Inserts the contents of the Edit Buffer immediately *before* the selected element in the Element List.
Append	Attaches the contents of the Edit Buffer to the end of the Element List.
Delete	Removes the selected elements from the Element List.

When editing an element, you must observe strict rules of syntax. The definition for each element must state the element type, attribute (line/fill color or pin name), and a set of xy coordinates. A more complete description of the syntax required for each element is given the *Element Definitions* section which follows.

Element Definitions

When the pen color on an enclosed element is immediately followed by an asterisk (for example, LtBlue*), the fill color will be the same as the pen color, otherwise the fill color will be the background color.

General Format

```
[element type][attribute] x1,y1 x2,y2 x3,y3 x4,y4 [pin numbers]
```

Line

Attribute	pen color
x1,y1	start point of the line
x2,y2	end point of the line
x3,y3	n/a
x4,y4	n/a

Example:

```
Line  Device  -36,-7  20,-7
```

Polyline

Attribute	pen color
x1,y1	1st point of the polyline
x2,y2	2nd point of the polyline (optional if +Polyline)
x3,y3	3rd point of the polyline (optional)
x4,y4	4th point of the polyline (optional)
Note	A preceding plus sign (+Polyline) indicates that this is an extension to the preceding polyline element.

Examples:

```
Polyline  Device  -24,-46   42,-22   24,35
```

or

```
Polyline  Device  -51,-54   46,-23   34,42   -53,32
+Polyline  Device  -53,-19
```

Polygon

Attribute	pen color*
x1,y1	1st point of the polygon
x2,y2	2nd point of the polygon (optional if +Polygon)
x3,y3	3rd point of the polygon (optional if +Polygon)
x4,y4	4th point of the polygon (optional)

Note: A preceding plus sign (+Polygon) indicates that this is an extension to the preceding polygon element.

Example:

```
Polygon  Device  -39,12   0,55   -61,75
```

or

```
Polygon  Device  -30,-44   12,-44   29, -21   15,6
+Polygon  Device  -25,6   -42,16
```

Rect

Attribute	pen color*
x1,y1	left top corner of the rectangle
x2,y2	right bottom corner of the rectangle
x3,y3	n/a
x4,y4	n/a

Example:

```
Rect  Device  -25,-30   15,28
```

RRect

Attribute	pen color*
x1,y1	left top corner of the rounded rectangle
x2,y2	right bottom corner of the rounded rectangle
x3,y3	width and height of ellipse defining the corners
x4,y4	n/a

Example:

```
RRect  Device  -56,-22   11,8   10,10
```

Ellipse/Circle

Attribute	pen color*
x1,y1	left top corner of the defining rectangle
x2,y2	right bottom corner of the defining rectangle
x3,y3	n/a
x4,y4	n/a

Example:

```
Ellipse  Device  -49,-44  9,33
```

Arc 1/4, 1/2

Attribute	pen color
x1,y1	left top corner of the rectangle defining the complete ellipse
x2,y2	right bottom corner of the rectangle defining the complete ellipse
x3,y3	end point of line 1 whose start point is at the center of the ellipse
x4,y4	end point of line 2 whose start point is at the center of the ellipse

Note: The arc follows the outline of the ellipse and is drawn counterclockwise from line 1 to line 2.

Example:

```
Arc  Device  -51,-13  -13,33  -32,-13  -51,10  1/4
Arc  Device  -28,-11  5,45  5,17  -28,17  1/2
```

Text

Attribute	text color
x1,y1	bottom center location of text
x2,y2	n/a
x3,y3	n/a
x4,y4	n/a
text	string enclosed in single quote marks (15 characters max.)

Example:

```
Text  Device  0,25  'symbol  text'
```

PinUp, PinDown, PinLeft, PinRight
PinUp~, PinDown~, PinLeft~, PinRight~

Attribute	pin name (16 characters max.)
x1,y1	point where pin attaches to package
x2,y2	n/a
x3,y3	n/a
x4,y4	n/a

Examples of single part per package:

```
Pinleft  P1  -26,-27  [1]
Pinright  P9  26,36  [9]
```

Examples of two parts per package:

```
Pinleft  P1  -26,-27  [1,2]
Pinright  P9  26,36  [9,10]
```

Example of inverted pin:

```
Pinleft~  P7  -26,27  [7]
```

Note: A pin is a connection point for wires. Pins can have bubbles to indicate negative logic. Pins with bubbles are indicated with the tilde (~).

Tutorial: Creating a Device Symbol

The following step-by-step example shows how to use the Symbol Editor to create macro circuits.

1 Make a backup copy of the USER.LIB file prior to creating or deleting a macro.

In case the library is damaged or altered in any undesirable way, you will always have a copy of the library as a backup.

2 Choose **Macros** > **New Macro** (or click the **Macro** button on the Toolbar) to display the Define New Macro dialog box shown in Figure 17.4.

It's possible that a message will appear asking if you want to include the present circuit inside the new symbol. For this example you can click **No**.

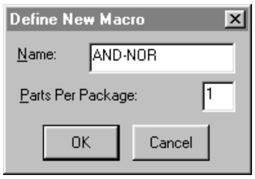

Define New Macro
Name: AND-NOR
Parts Per Package: 1

Figure 17.4. The name you enter here is used later to select the device from the library.

3 Enter a unique name up to 13 characters in length in the **Macro Name** text edit field. For this example, type **AND-NOR**.

4 Specify the number of parts per package. For this example, use the default of **1** part per package.

This refers to the number of devices found in the same chip. For example, a 7400 package contains 4 NAND gates and a 556 package contains 2 timers.

5 Click OK to display the Symbol Editor.

For this example, begin by placing an 8 Pin DIP package in the drawing window.

6 Type **8** in the Pins Per Pkg field then click the **Add Pkg** button.

7 Click the left mouse button to place the package in the center of the drawing window.

8 Change the View to 200% by clicking the **Up Arrow** in the View group box.

The macro name **AND-NOR** displays in the center of the drawing window.

9 Click and drag the name into position near the top of the DIP.

Notice that this device consists of 1 Rect, 4 PinLefts and 4 PinRights. You will remove 3 of the 4 PinRights.

10 Click the top right-hand pin in the drawing to select it, and then click **Delete**. Repeat for the 2 bottom right-hand pins.

11 Click the remaining right-hand pin to select it, and then click **Copy** to copy the pin from the Element List to the Edit Buffer.

12 Change **PinRight** to **PinRight~**.

13 Click **Replace** to replace the selected pin with the new pin.

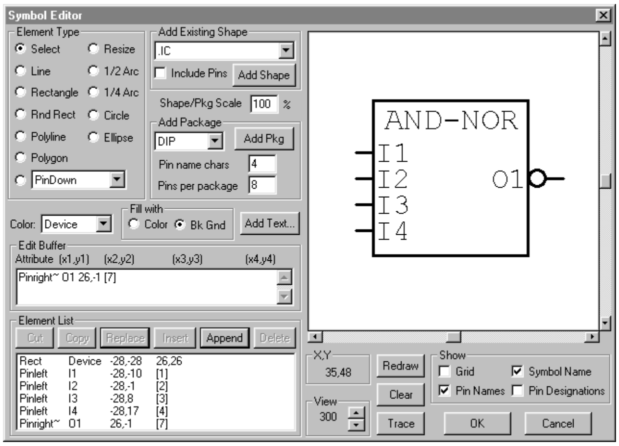
Notice that the pin in the drawing now has a bubble.

14 Left-click the top left-hand pin and choose **Edit**.

15 Change **P1** to **I1**, and then click OK. Repeat for each remaining pin, naming them **I2**, **I3**, **I4** and **O1**.

Compare your device with the one pictured in Figure 17.5.

16 Click OK in the Symbol Editor dialog



Element Type: [✓ Select]; PinDown

Add Existing Shape: .IC; Include Pins [ ]

Shape/Pkg Scale 100 %

Add Package: DIP

Pin name chars 4

Pins per package 8

Color: Device

Fill with: Color [✓ Bk Gnd]

Edit Buffer

```
Attribute (x1,y1)   (x2,y2)      (x3,y3)      (x4,y4)
Pinright~ O1 26,-1 [7]
```

Element List

```
Rect       Device  -28,-28   26,26
Pinleft    I1      -28,-10   [1]
Pinleft    I2      -28,-1    [2]
Pinleft    I3      -28,8     [3]
Pinleft    I4      -28,17    [4]
Pinright~  O1      26,-1     [7]
```

X,Y 35,48

View 300

Show: Grid [ ]; [✓ Symbol Name]; [✓ Pin Names]; Pin Designations [ ]

Figure 17.5. This is what the new symbol should look like after completing the steps so far.

After you click OK, the new symbol you created displays and follows the mouse around the work area.

17 Click the mouse to place the symbol in the workspace.

18 Double-click the macro package.

19 Click **Netlist**.

Any data you enter in the Device Properties dialog box at this time will be there every time you select this macro from the library menus. Now is a good time to add the default Package, Auto Designation Prefix, Spice Prefix Character(s) and Spice Data if required.

Note: For compatibility with TraxMaker, the Package field must match the name of the corresponding component in TraxMaker.

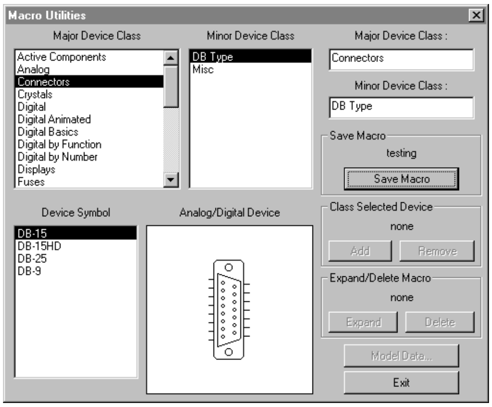
20 If you are satisfied with your macro, choose **Macros** > **Macro Utilities** to display the Macro Utilities dialog box as pictured in Figure 17.6.

21 To place the device in an existing Major or Minor Device Class, click the appropriate items in the lists.

22 To create a new Major or Minor Device Class, type the **Major** or **Minor Device Class** name in the appropriate text edit fields.

23 Finally, click **Save Macro**.

Clicking Save Macro saves the new macro in the file USER.LIB and clears the workspace. To use the new macro simply select it from the device library and use it just as you would any other device.



Major Device Class : Connectors

Minor Device Class : DB Type

Figure 17.6. You must specify the Major and Minor Device Classes under which your macro will be shown in the Device Selection dialog box.

Expanding an Existing Macro Device

Expanding a macro device lets you modify the device's symbol, the default netlist attributes, and add or edit any internal circuitry.

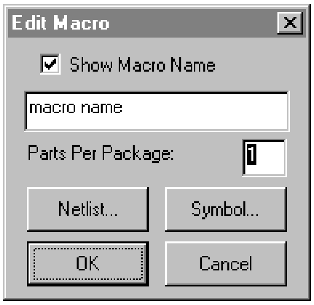
To expand an existing macro device,

1 Place the macro you want to expand in the drawing area.

2 Select the macro by clicking it once with the **Arrow Tool**.

3 Choose **Macros** > **Expand Macro**. CircuitMaker warns you that the workspace will be cleared (any circuitry that has not been saved will be lost.) Click OK.

4 Double-click on the macro device. The dialog box shown in Figure 17.7 appears.



[✓] Show Macro Name

macro name

Parts Per Package: 1

Figure 17.7. Use this dialog box to specify what part of the macro device you want to edit.

5 If you want, type a new name for the macro.

Saving a macro under a new name creates a new macro; it does not delete the old macro device.

6 Use the **Parts Per Package** option to change the number of parts that would be found in a single IC package.

Note: Use this option only when creating a new device.

7 Click the **Netlist** button to display the Device Properties dialog box.

Depending on the macro device you are editing, you might need to specify mandatory parameters for the device, including Label-Value, Package, Auto Designation Prefix, Spice Prefix, and Spice Data.

8 Click the **Symbol** button to display the Symbol Editor, where you can edit the expanded macro's schematic symbol.

9 Click OK when you have finished drawing and specifying parameters for the symbol.

Clicking OK returns you to the expanded macro where you can add or edit circuitry before the new device is saved. See *Creating Macro Circuits* later in this chapter for the step-by-step procedure.

Creating Macro Devices with Internal Circuitry

If you cannot find a device in CircuitMaker's library of devices (or elsewhere) that performs a particular function, you can create a functional macro device by attaching hidden circuitry to a custom symbol.

A macro device can contain internal circuitry that you can base on CircuitMaker's library of existing devices. You can also nest macro devices, meaning that a macro can be used within another macro device, letting you create building blocks and piece them together in order to create the final device.

When you save a macro, all of the circuitry you have added to it is hidden within its symbol. The new macro device functions according to the circuitry hidden within. You can then use the new macro in any circuit.

To create a macro device with internal circuitry,

1 Make a backup copy of the USER.LIB file.

In case the library is damaged or altered in any undesirable way, you will always have a copy of the library as a backup.

2 Create a symbol as described earlier in this chapter (see *Creating Device Symbols* earlier in this chapter).

3 Place the symbol that you created in the drawing area and expand it by selecting it and then choosing **Macros** > **Expand Macro** (see *Expanding an Existing Macro Device* earlier in this chapter for more details).

4 Construct the circuitry which performs the function your new macro device is to have.

For example, suppose you want to build a device that performs an AND-NOR function. You would construct something like the circuit shown in Figure 17.8.

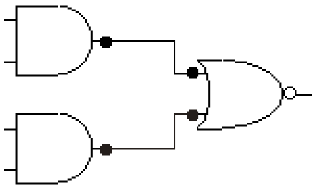


Figure 17.8. You can add functional circuitry such as this AND-NOR function.

5 Select the **Wire Tool** from the Toolbar and connect wires from the macro symbol's pins to the other points within the circuit. Figure 17.9 shows an example of a completely wired macro device.

Note: *If a macro device contains internal circuitry as well as SPICE data, the SPICE data will be ignored.*

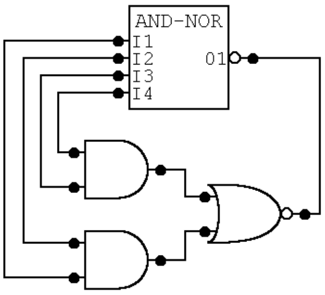


Figure 17.9. The functional circuitry is wired to the symbol to create a functional macro device.

6 If default data was not added when the symbol was created, then double-click on the macro symbol and click the Netlist button. Add any necessary data such as Package and Auto Designation Prefix. Data entered now in the Device Properties dialog box will be there every time this macro device is selected from the library menus. Note: For compatibility with TraxMaker, the Package field must match the name of the corresponding component in TraxMaker.

7 Save the Macro (choose **Macros** > **Save Macro**).

Working with Spice Models

Spice is an industry standard program for simulating circuits. In order to make devices work in analog simulation, there must be Spice data available for each device. Using the Symbol Editor, you can include Spice model and subcircuit information with a new or existing device. You can also edit Spice information using an ASCII Text Editor.

Note: If you intend to create new device symbols to export a PCB netlist or simply draw schematics, you do not need to include Spice information. However, even if you are going to simulate your circuit, adding Spice models from other sources is easy and beneficial. To learn more about Spice, see *Chapter 17: Spice: Beyond the Basics.*

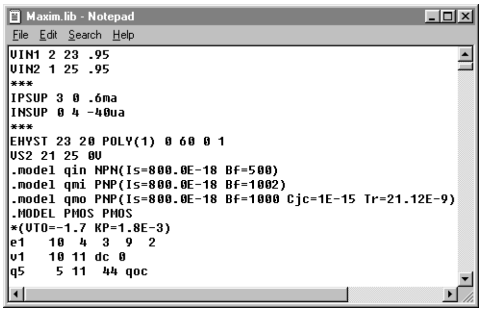
There are 3 basic types of components in Spice:

- **Elementary** components such as resistors, capacitors, power sources, etc.
- **Models** defining discrete devices such as BJTs, J-FETs, MOSFETs, etc.
- **Subcircuits** which combine multiple items (such as elementary components, models, and other subcircuits) to create a more complex device.

Spice models and subcircuits are available from many sources, including component manufacturers, engineering magazines and books. You can also download them from the internet.

Editing Spice Models with a Text Editor

Since Spice models (and subcircuits) are text files (see Figure 17.10), they can be modified with a text editor such as Notepad. If you edit this file with a word processor, be sure to save the file in a TEXT ONLY format.

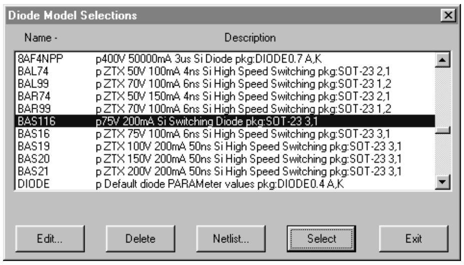


```
VIN1 2 23 .95
VIN2 1 25 .95
***
IPSUP 3 0 .6ma
INSUP 0 4 -40ua
***
EHYST 23 20 POLY(1) 0 60 0 1
VS2 21 25 0V
.model qin NPN(Is=800.0E-18 Bf=500)
.model qmi PNP(Is=800.0E-18 Bf=1002)
.model qmo PNP(Is=800.0E-18 Bf=1000 Cjc=1E-15 Tr=21.12E-9)
.MODEL PMOS PMOS
*(VTO=-1.7 KP=1.8E-3)
e1   10  4  3  9  2
v1   10 11 dc 0
q5    5 11  44 qoc
```

Figure 17.10. You can open Spice data into any text editor and edit it directly.

Editing Spice Models in CircuitMaker

When you double-click a device that has Spice models associated with it, the dialog box in Figure 17.11 appears.



Name -	Description
8AF4NPP	p400V 50000mA 3us Si Diode pkg:DIODE0.7 A,K
BAL74	p ZTX 50V 100mA 4ns Si High Speed Switching pkg:SOT-23 2,1
BAL99	p ZTX 70V 100mA 6ns Si High Speed Switching pkg:SOT-23 1,2
BAR74	p ZTX 50V 150mA 4ns Si High Speed Switching pkg:SOT-23 2,1
BAR99	p ZTX 70V 100mA 6ns Si High Speed Switching pkg:SOT-23 1,2
BAS116	p75V 200mA Si Switching Diode pkg:SOT-23 3,1
BAS16	p ZTX 75V 100mA 6ns Si High Speed Switching pkg:SOT-23 3,1
BAS19	p ZTX 100V 200mA 50ns Si High Speed Switching pkg:SOT-23 3,1
BAS20	p ZTX 150V 200mA 50ns Si High Speed Switching pkg:SOT-23 3,1
BAS21	p ZTX 200V 200mA 50ns Si High Speed Switching pkg:SOT-23 3,1
DIODE	p Default diode PARAMeter values pkg:DIODE0.4 A,K

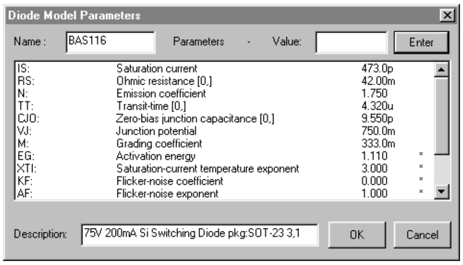
Figure 17.11. This dialog box lets you select a diode model.

The currently selected model is highlighted in the list. To select a different model, click it with the mouse, then click **Select** (or just double-click on the model). If any subcircuits are found in the .MOD file, they are indicated by an **x** as the first character in the description instead of a **p**.

To edit or view an existing model,

1 Click on the name of the model you want to edit or view.

2 Click on the **Edit** button to display the dialog box pictured in Figure 17.12.



Name : BAS116   Parameters - Value:

```
IS:     Saturation current                          473.0p
RS:     Ohmic resistance [0,]                       42.00m
N:      Emission coefficient                        1.750
TT:     Transit-time [0,]                           4.320u
CJO:    Zero-bias junction capacitance [0,]         9.550p
VJ:     Junction potential                          750.0m
M:      Grading coefficient                         333.0m
EG:     Activation energy                           1.110    *
XTI:    Saturation-current temperature exponent     3.000    *
KF:     Flicker-noise coefficient                   0.000    *
AF:     Flicker-noise exponent                      1.000    *
```

Description: 75V 200mA Si Switching Diode pkg:SOT-23 3,1

Figure 17.12. Use the Diode Model Parameters dialog box to change the settings of a particular model.

The model parameters displayed are Spice model parameters not data book parameters. Unless you are familiar with Spice modeling, it is recommended that you do not modify the existing Spice models.

The values listed for each parameter represent values defined for that specific device type. Default Spice model parameter values are indicated by an asterisk (*) after the value.

To change the value of a Spice parameter,

1 Click the name of the Spice parameter.

This selects the parameter and copies the parameter's value into the Value edit box.

2 Change the data in the **Value** box and click the **Enter** button.

If you want to set a specific parameter to equal that of the DEFAULT model, type an asterisk (*) in the Value edit field and press Enter.

3 Click OK to save the model.

OR

To save it under a new name, type the new name into the Name field, and then click OK.

Note: Models created or modified in this manner are stored directly in the .MOD linking file in place of the model reference.

The original .LIB file remains unchanged. When viewed in the Model Selections dialog box, the first character of the device description will be an asterisk (*).

To remove an existing model,

1 Select it by clicking it with the mouse.

2 Click **Delete**.

This only removes the model reference from the linking file; it does not remove the actual model from the library.

Adding New Models to an Existing Symbol

When you obtain new models from an outside source, they are generally provided in a single ASCII library file.

Note: If you edit this file with a word processor, be sure to save the file in TEXT ONLY format.

Use the **Model Data** button in the **Macro Utilities** dialog box to create a link to each model in the new library file. The following steps illustrate an example of adding a new model.

__Note:__ Normally you must copy the .lib file that contains the .MODEL or .SUBCKT data to the CircuitMaker Models directory. For this 2N5209 example, the MCEBJT.LIB file is already in the Models directory.

To add a 2N5209 transistor to CircuitMaker,

1 Choose **Macros** > **Macro Utilities**.

2 Select the symbol for an NPN transistor, then click **Model Data**.

3 Click **Open** and open the library file (MCEBJT.LIB) in which the 2N5209 model resides.

All of the model and subcircuit names found in this library will be displayed in the list box on the left.

4 Click **2N5209**.

5 Enter appropriate information about the model in the Description field (for example: **Si 625mW 50V 50mA 30MHz Amp**).

6 In the Pkg Name field enter **TO-92B** to match the name of the component pattern in TraxMaker.

7 Enter pin numbers to match the pad designations of the component in TraxMaker. On the TO-92B in TraxMaker, with the flat side facing you, pin 1 is on the right, pin 2 is in the middle and pin 3 is on the left. For the 2N5902, these pins correspond to the collector, base and emitter, respectively (C=1, B=2, E=3).

8 Click **Add** to add the new reference.

You can now select the new model alphabetically from the list of npn transistors.

Adding Existing Models to New Macro Symbol

To add an existing model to a new macro symbol,

1 Create a new nonfunctional macro device symbol as described earlier in this chapter.

Be sure to place the pins in the same order as they are listed in the syntax for the corresponding model.

For example, if you are creating a new symbol for a diode, you should place the anode pin first, then the cathode pin to correspond with the Spice definition of a diode. This is not required, but it will be easier to understand when filling in the Spice Data field for the device.

To link the new symbol to an existing model file, you must name the new symbol appropriately.

For example, you may want to name the symbol Diode:A to link to the DIODE.MOD file or **Schottky:A** to link to the SCHOTTKY.MOD file.

2 With the new macro expanded, double-click the device, and then click **Netlist** to view the Device Properties dialog box.

3 Enter the following data:

- Auto Designation Prefix for the device. For example, Q for transistors, D or CR for Diodes, etc.
- Set the Spice Prefix Character(s) for this device to be representative of the type of device models you are using. For example, an NPN Bipolar Junction Transistor must have the prefix **QN**. Refer to *Chapter 4: Drawing and Editing Schematics* for a listing of valid prefix characters.

- Enter an appropriate Spice instruction for this device into the Spice Data field. Refer to *Chapter 4: Drawing and Editing Schematics* for more information.

6 Click OK.

7 Click **Macros** > **Save Macro** to save the macro.

8 After you have entered the appropriate data, Spice models can be linked to the new symbol by following the instructions under *Adding New Subcircuits to an Existing Model* later in this chapter.

Example

To create a new device symbol for standard junction diodes,

1 Create a new macro symbol for a diode using the Symbol Editor described in *Creating a New Macro Device* earlier this chapter.

2 Name the new macro symbol DIODE:A.

When placing pins on the new device, place the *first* pin on the anode (N+), and the *second* pin on the cathode (N-).

They should be placed in this order to match the syntax for the diode model.

3 Expand the new macro symbol (if it is not already expanded) by clicking it once with the left mouse button, and then choosing **Macros** > **Expand Macro**.

4 Double-click on the device symbol, and click on the **Netlist** button.

5 Enter the following information in the Device Properties dialog box of the expanded macro (to set the defaults for this macro device):

Auto Designation Prefix:	D
Spice Prefix Character(s):	D
Spice Data field:	`%D %1 %2 %M`

6 Save the macro (see *Save Macro* in *Chapter 13: Macros Menu* for more information).

7 Follow the instructions in *Adding New Subcircuits to an Existing Symbol* later in this chapter.

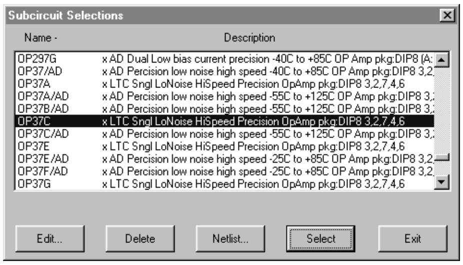
Editing Spice Subcircuits

Spice subcircuits fall into three basic categories:

- Component models
- Macromodels
- Equivalent circuits

A component model is basically a complete schematic of the chip, simulated using discrete components. This type of subcircuit is generally more accurate than a macromodel, but requires more time to simulate. The macromodel is more of a block diagram of the chip, where inputs and outputs may be simulated using discrete components, but the internal workings consist of simpler items such as gain blocks, etc. This type of subcircuit simulates rather quickly and in most cases is accurate enough that a component model is not needed. Equivalent circuits may be needed to simulate discrete devices that have no Spice model. For example, an SCR can be roughly equated to a pair of NPN and PNP transistors coupled together.

When you double-click a device that has subcircuits in the library the dialog box pictured in Figure 17.13 appears.



Name -	Description
OP297G	x AD Dual Low bias current precision -40C to +85C OP Amp pkg:DIP8 (A:
OP37/AD	x AD Percision low noise high speed -40C to +85C OP Amp pkg:DIP8 3,2
OP37A	x LTC Sngl LoNoise HiSpeed Precision OpAmp pkg:DIP8 3,2,7,4,6
OP37A/AD	x AD Percision low noise high speed -55C to +125C OP Amp pkg:DIP8 3,
OP37B/AD	x AD Percision low noise high speed -55C to +125C OP Amp pkg:DIP8 3,
OP37C	x LTC Sngl LoNoise HiSpeed Precision OpAmp pkg:DIP8 3,2,7,4,6
OP37C/AD	x AD Percision low noise high speed -55C to +125C OP Amp pkg:DIP8 3,
OP37E	x LTC Sngl LoNoise HiSpeed Precision OpAmp pkg:DIP8 3,2,7,4,6
OP37E/AD	x AD Percision low noise high speed -25C to +85C OP Amp pkg:DIP8 3,2
OP37F/AD	x AD Percision low noise high speed -25C to +85C OP Amp pkg:DIP8 3,2
OP37G	x LTC Sngl LoNoise HiSpeed Precision OpAmp pkg:DIP8 3,2,7,4,6

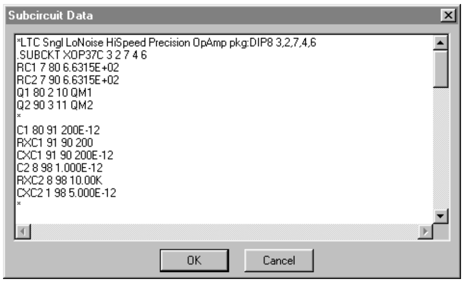
Figure 17.13. This device is made up of many subcircuits, any of which you can select and edit.

The currently selected subcircuit is highlighted in the list. To select a different subcircuit, click on it with the mouse, then click **Select** (or just double-click it with the mouse).

To edit or view an existing subcircuit,

1 Click it with the mouse.

2 Click **Edit** to display the dialog box in Figure 17.14.



```
*LTC Sngl LoNoise HiSpeed Precision OpAmp pkg:DIP8 3,2,7,4,6
.SUBCKT XOP37C 3 2 7 4 6
RC1 7 80 6.6315E+02
RC2 7 90 6.6315E+02
Q1 80 2 10 QM1
Q2 90 3 11 QM2
*
C1 80 91 200E-12
RXC1 91 90 200
CXC1 91 90 200E-12
C2 8 98 1.000E-12
RXC2 8 98 10.00K
CXC2 1 98 5.000E-12
*
```

Figure 17.14. Use this dialog box to edit or view an existing subcircuit.

This dialog box lists the subcircuit, beginning with the description line and ending with the .ENDS line. Any changes you make to the subcircuit cause the modified subcircuit to replace the reference in the subcircuit linking file. The original subcircuit remains unchanged in the library file, but the reference to it will be lost. Naming conventions for .SUB files are discussed in *Model and Subcircuit Linking Files* later in this chapter.

Adding New Subcircuits to an Existing Symbol

When you obtain new subcircuits from an outside source, they are generally provided in a single ASCII library file.

Note: If you edit this file with a word processor, be sure to save the file in TEXT ONLY format.

Use the **Model Data** button in the **Macro Utilities** dialog box to create a link to each model in the new library file. The following steps illustrate adding a new subcircuit to an existing symbol.

To add an LF412C operational amplifier (5-pin subcircuit) to CircuitMaker,

Note: *Normally you must copy the .lib file that contains the .SUBCKT data to the CircuitMaker Models directory. For this LF412C example, the MCEMODS2.LIB file is already in the Models directory.*

1 Select **Macros** > **Macro Utilities**.

2 Select the symbol for the 5-pin opamp (Op-Amp5), and then click **Model Data**.

3 Click **Open** and open the library file (MCEMODS2.LIB) in which the LF412C model resides.

All of the model and subcircuit names found in this library will be displayed in the list box on the left.

4 Click on the **LF412C**.

5 Enter appropriate information about the model in the Description field (for example: Dual LoOffset LoDrift JFET OpAmp).

6 In the Pkg Name field enter **DIP8** to match the name of the component pattern in TraxMaker.

7 Enter pin numbers to match the pad designations of the component in TraxMaker.

Since the **LF412C** is a dual op amp, you must enter pin numbers for both PART A and PART B (click the **Up** or **Down Arrow** to switch between the two).

For PART A the pin numbers should be: IN+ = 3, IN- = 2, V+ = 8, V- = 4 and Out = 1. For PART B the pin numbers should be: IN+ = 5, IN- = 6, V+ = 8, V- = 4 and Out = 7. Note that the V+ and V- power supply pins are the same for both op amps.

8 Click **Add** to include the new reference.

You can now select the new subcircuit alphabetically from the list of 5-pin op amps.

Adding Existing Subcircuits to New Macro Symbol

To add an existing subcircuit to a new macro symbol,

1 Create a new nonfunctional macro device symbol as described earlier in this chapter.

Be sure to place the pins in the same order as they are listed in the corresponding subcircuit. For example, if you are creating a new symbol for a 5-pin opamp, you should place the +Input pin first, then the -Input pin, the +Vsupply pin, the -Vsupply pin and finally the Output pin to correspond with the Spice node connections for the 5-pin opamp subcircuit. This is not required, but it will be easier to understand when filling in the Spice Data field for the device. In order to link the new symbol to an existing subcircuit file, you must name the new symbol appropriately. For example, you may want to name the symbol "Op-Amp5:A" to link to the OPAMP5.SUB file.

Note: Since CircuitMaker identifies pins by the order of placement in a device symbol, for clarity make sure that the placement order of the pins matches the order of the nodes listed in the subcircuits. The actual link between device symbol pins and Spice subcircuits is controlled by the Spice Data field of the device where %*n* refers to the *nth* pin in the element list. For example, in the Spice Data

```
%D %1 %2 %3 %4 %S
```

the node number connected to the first pin placed on the device symbol is represented by %1, the second pin by %2, etc. When used with a subcircuit which begins

```
.SUBCKT XMR9933A 21 23 6 15
```

node 21 in the subcircuit connects to the first pin on the device symbol, etc.

2 With the new macro expanded, double-click on the device, then click on the **Netlist** button to view the Device Properties dialog box. Enter the following data:

- Enter an appropriate Auto Designation Prefix for the device. For example, U or IC for integrated circuits, etc.
- Set the Spice Prefix Character(s) to "X" for subcircuit devices.
- Place an appropriate Spice instruction for this device into the Spice Data field. Refer to *Editing*

Devices in *Chapter 4: Drawing and Editing Schematics* for more information or refer to the examples provided with other devices.

3 Click OK.

4 Choose **Macros** > **Save Macro** to save the macro.

5 After you have entered the appropriate data, Spice subcircuits can be linked to the new symbol by following the instructions in *Adding New Subcircuits to an Existing Symbol* earlier in this chapter.

Example

To create a new device symbol for optoisolators,

1 Create a new macro symbol for an optoisolator using the symbol editor described earlier in this chapter.

Name the new macro device symbol Opto Isol:A.

When placing pins on the new device, place the *first* pin on the anode (N+), the *second* pin on the cathode (N-), the *third* pin on the collector, and the *fourth* pin on the emitter.

They should be placed in this order to match the syntax for the optoisolator subcircuits.

2 Expand the new macro symbol (if it is not already expanded) by clicking on it once with the left mouse button, then click on the **Macro** button in the Toolbar.

3 Double-click the device symbol, and click **Netlist**.

4 Enter the following information in the Device Properties dialog box of the expanded macro (to set the defaults for this macro device):

Auto Designation Prefix:	OP
Spice Prefix Character(s):	X
Spice Data field:	`%D %1 %2 %3 %4 %S`

5 Click OK then choose **Macros** > **Save Macro**.

6 Follow the instructions in *Adding New Subcircuits to an Existing Symbol* earlier in this chapter.

Model and Subcircuit Linking Files

Model and Subcircuit data is stored in ASCII text files, typically with the .LIB extension. You can edit these files directly with any ASCII text editor.

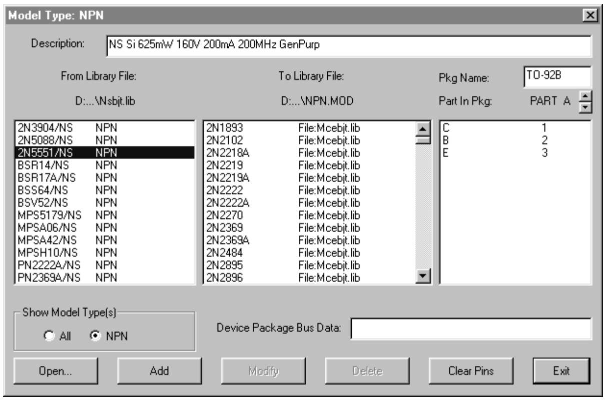
Each analog device symbol in CircuitMaker has a linking file associated with it. In order to use Spice models in CircuitMaker, you must link the models to a corresponding device symbol by placing a reference to the model in the linking file.

To create a new link or to modify an existing link,

1 Select **Macros** > **Macro Utilities**.

2 Select the symbol that you want to use for the specific model or subcircuit you are adding.

3 Click **Model Data** to display the dialog box pictured in Figure 17.15.



Model Type: NPN

Description: NS Si 625mW 160V 200mA 200MHz GenPurp

From Library File: D:...\Nsbjt.lib

To Library File: D:...\NPN.MOD

Pkg Name: TO-92B

Part In Pkg: PART A

```
2N3904/NS     NPN
2N5088/NS     NPN
2N5551/NS     NPN
BSR14/NS      NPN
BSR17A/NS     NPN
BSS64/NS      NPN
BSV52/NS      NPN
MPS5179/NS    NPN
MPSA06/NS     NPN
MPSA42/NS     NPN
MPSH10/NS     NPN
PN2222A/NS    NPN
PN2369A/NS    NPN
```

```
2N1893     File:Mcebjt.lib
2N2102     File:Mcebjt.lib
2N2218A    File:Mcebjt.lib
2N2219     File:Mcebjt.lib
2N2219A    File:Mcebjt.lib
2N2222     File:Mcebjt.lib
2N2222A    File:Mcebjt.lib
2N2270     File:Mcebjt.lib
2N2369     File:Mcebjt.lib
2N2369A    File:Mcebjt.lib
2N2484     File:Mcebjt.lib
2N2895     File:Mcebjt.lib
2N2896     File:Mcebjt.lib
```

```
C    1
B    2
E    3
```

Show Model Type(s): All [✓ NPN]

Device Package Bus Data:

Figure 17.15. Use this dialog box to add, modify, and remove linking information.

Initially, only the center list box will have anything in it. This is a list of the devices in the linking file associated with the selected symbol. Click one of the models to select it and the description, package and pinout information appears.

To modify this information,

1 Select the name of the model or subcircuit you want to delete.

2 Type in the desired changes.

3 Click **Modify**.

To remove a reference from the linking file,

1 Select the name of the model or subcircuit you want to delete.

2 Click **Delete**.

To add new references to the linking file,

1 Click **Open** and open the library file in which the actual model or subcircuit resides.

   All of the model and subcircuit names found in this library will be displayed in the list box on the left. If desired, the list can be limited to only those devices which are compatible with the selected device symbol.

2 Enter appropriate description, package and pin information for the model.

   If there are multiple parts per package, be sure to enter pin numbers for each of PART A, PART B, etc. If the model is similar to another part that is already in the linking file, it may be simpler to first select the similar part to fill in the blanks, make any necessary corrections, then select the new model by changing the **Show Model Type**.

3 Click **Add** to include the new reference.

The names of the linking files correspond to the device symbol names, with the following exceptions:

- Linking file names are limited to 8 characters, so the device symbol name is truncated to 8 characters when searching for a match.
- All spaces and punctuation characters are removed.

- If the last 1 or 2 characters of a truncated device symbol name are numbers, those numbers replace the last characters in the file name.
- If a device symbol name includes a colon (:), the colon and all characters after the colon are ignored. For example, the devices **NPN Trans**, **NPN Trans:A**, **NPN Trans:B** and **NPN Trans:C** are all associated with the linking file NPNTRANS.MOD.

If CircuitMaker cannot find a linking file using the file name formed from the device symbol name, and if that symbol can be simulated using simple Spice models, a default file name is used instead. The default file name is based on the Spice Prefix Character(s). Following is a list of the Spice prefix character(s) and their corresponding default file names:

Spice Prefix	**Default File Names**
C	CAP.MOD
D	DIODE.MOD
DZ	ZENER.MOD
JN	NJFET.MOD
JP	PJFET.MOD
MN	NMOS.MOD
MP	PMOS.MOD
O	LTRA.MOD
QN	NPN.MOD
QP	PNP.MOD
R	RESISTOR.MOD
S	SW.MOD
U	URC.MOD
W	CSW.MOD
ZN	NMESFET.MOD
ZP	PMESFET.MOD

All others, including subcircuit-only device symbols, do not have a default file name. Subcircuit-only device symbols (symbols that cannot be simulated with a simple model) must use linking files with the .SUB extension.

Transistors are an example of when a default file name is used. Since the file NPNTRANS.MOD does not exist, CircuitMaker looks in the default file NPN.MOD for the model reference information.

.MOD linking files contain references to a collection of Spice models and subcircuits specific to a particular discrete device symbol in CircuitMaker. In some cases, it is desirable to replace a model with a subcircuit which will more accurately model a particular device. For example, some RF or Power Transistors are not modeled well by a simple Spice model. In such cases, it is OK to reference the subcircuit in the .MOD file just like it was a model. For more information on using models, see *Working with Spice Models* earlier in this chapter.

.SUB linking files contain references to a collection of Spice subcircuits specific to a particular device symbol in CircuitMaker. For more information on using subcircuits, see *Editing Spice Subcircuits* earlier in this chapter.

Under normal circumstances, the linking files should not be edited directly. The file format is described below for information only.

The general format for a reference in a linking file is:

```
*Device Description pkg:PACKAGE [DVCC=14;DGND=7;] 1,2,3,…
.PARAM QXXXXXXXX File:Filenam.lib
```

where PACKAGE [DVCC=14;DGND=7;] 1,2,3,… is the component name and pad designations in TraxMaker (the bus data listed here is for digital component symbols that do not have external power and ground pins), Q is the Spice prefix character for the specific device type, XXXXXXXX is the model name and Filenam.lib is the name of the library file in which the model or subcircuit actually resides.

There should be an appropriate description of the device on the line *immediately before* each .PARAM line. This line will be displayed as the description in the Model or Subcircuit Selections dialog box. The first character of this description line must be an asterisk (*). When you double-click a device to select a model, subcircuits that are found in the .MOD file will also be displayed, but the first character in the description will appear as an “x” instead of a “p”. You should also include appropriate package and pin information at the end of the description line. This information is required when creating a PCB netlist for TraxMaker. The package name represents the component name that is to be used in TraxMaker. The pin information indicates the pinout of the device in the package. For example:

```
*500mW 40V 800mA pkg:TO-18 3,2,1
.PARAM Q2N2222A File:Mcebjts.lib
```

Note that the pin order is specified as 3,2,1. Spice defines the pin order of a BJT as collector, base, emitter (see *Bipolar Junction Transistors (BJTs)* in *Chapter 6: Analog/Mixed - Signal Simulation*). In a TO-18 package containing a 2N2222A transistor, the emitter is connected to pin 1 (next to the tab), the base is connected to pin 2 and the collector is connected to pin 3. These pin numbers correspond to the pad designations in TraxMaker. CircuitMaker identifies pins by the order of placement in a device symbol and TraxMaker identifies pads by their designation. This information can be used to create a netlist linking one program to the other.

If the same device is available in different packages, you may specify additional packages using an alias. Notice the Spice prefix character is replaced with the letter **A** and the file reference is removed:

```
*350mW 40V 800mA alias:Q2N2222A pkg:TO-92B 1,2,3
.PARAM APN2222A
```

If there are multiple devices in the same package, the pins need to be specified for each device with a letter to indicate the designation extension:

```
*Dual Op Amp pkg:DIP8 (A:3,2,8,4,1)(B:5,6,8,4,7)
.PARAM XMC1458 File:Mcemods.lib
```

These two methods may be combined if you have multiple devices that are available in different packages.

Linking Inside Subcircuits Using an Alias

Sometimes it is necessary to link two different devices to the same subcircuit. This may be required when two devices contain identical internal circuitry, but different package or pin information. The best way to do this is to add an ALIAS to the subcircuit (*.sub) file using any text editing program (like NotePad®, WordPad®, etc.). The file must be saved in standard text format. Below is the general text format:

```
*Description alias:XLINK|SUBNAME pkg:[Package & Pins]
.PARAM XNEWNAME
```

Data	**Description**
Description	Information about the nature of the device, such as voltage, current, etc.
XLINK	The name of the subcircuit to be referenced. The first letter will always be the appropriate Spice character, followed by the subcircuit name.
|	Vertical bar separator. NOTE: There cannot be a space between the vertical bar separator and the XLINK or SUBNAME.
SUBNAME	The name of the subcircuit file (*.sub) where the subcircuit to be referenced is located. This will include the first 8 characters, not including spaces. Do not include the .sub extension.
pkg	The package name and pin numbers in appropriate format for export to a PCB layout program such as TraxMaker. NOTE: See different .sub files for examples of this format.
XNEWNAME	The new name of the device as seen in the parts list.

The following is an example of a subcircuit internal alias link:

Reference

```
*Instrumentation Amp: LoPWR pkg:DIP8 3,2,7,4,6,5,1,8
.PARAM XAD620A File:AnalogD.lib
```

New Device Data

```
*Instrument Amp: LoPWR alias:XAD620A|INSTAMP pkg:SMD8A 3,2,7,4,6,5,1,8
.PARAM XAD620AR
```

Creating New Spice Models with Parameter Passing

Parameter passing simplifies the task of creating new components. It allows you to pass databook values directly into generic Spice models or subcircuits, using mathematical equations to create Spice model parameters. The generic model is placed in the linking file associated with the device symbol, then referenced by an alias.

Devices created in this manner are selected and edited just like any other device model. To edit the parameters being passed, double-click the device, then click **Edit**.

General Form (Generic Model)

```
*MNAME:Device Title
*MNAME:P1:|P1 Description [<<Min>,<Max>>]|Default
*MNAME:P2:|P2 Description [<<Min>,<Max>>]|Default
      .
      .
      .
*MNAME:Pn:|Pn Description [<<Min>,<Max>>]|Default
*{P1=Default P1=Default ... Pn=Default}
*Desc pkg:Package Pins
```

General Form (Alias)

```
*Desc alias:MNAME {P1=Val P2=Val} pkg:Package Pins
.PARAM ANAME
```

Data	**Description**
MNAME	The name of the generic model or subcircuit, including the appropriate Spice prefix character.
P1, P2, etc.	The names of the parameters being passed.
P1 Description, etc.	The description of the parameters.
Min and Max	Optional items which limit the value that can be entered for each parameter.
Default	The default value for each parameter if no value is specified.
Desc	A description of the device.

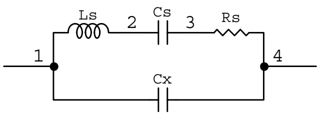
Package and Pins	Information used for export to TraxMaker for PCB layout.
ANAME	The alias name (the name of the specific device).

Valid operators in the math expressions include:

```
+ - * /
```

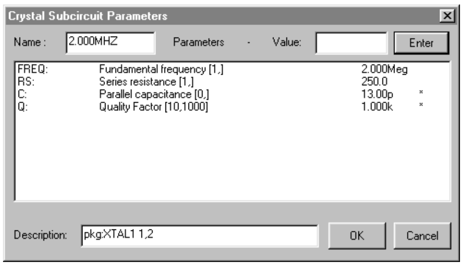
Following is an example of a generic subcircuit for a crystal which can be used as a template for creating other crystals:

```
*XCRYSTAL:Crystal Subcircuit Parameters
*XCRYSTAL:FREQ:|Fundamental frequency [1,]|1MEG
*XCRYSTAL:RS:|Series resistance [1,]|750
*XCRYSTAL:CX:|Parallel capacitance [0,]|13pf
*XCRYSTAL:Q:|Quality Factor [10,1000]|1000
*{FREQ=1Meg RS=750 C=13pf Q=1000}
*Generic 1MHz Crystal:crystal pkg:XTAL1 1,2
.SUBCKT XCRYSTAL 1 2
LS 1 2 {((Q*RS)/(6.2831852*FREQ))} IC=0.5M
CS 2 3 {(1/(Q*6.2831852*FREQ*RS))}
RS 3 4 {RS}
CX 1 4 {CX}
.ENDS XCRYSTAL
*alias:XCRYSTAL {FREQ=2E6 RS=250} pkg:XTAL1 1,2
.PARAM A2.000MHZ
```



Crystal Subcircuit

Since CX and Q are not passed in the alias parameter list, the default values of 13pF and 1000 will be used. See Figure 17.16.



Name: 2.000MHZ   Parameters -   Value:

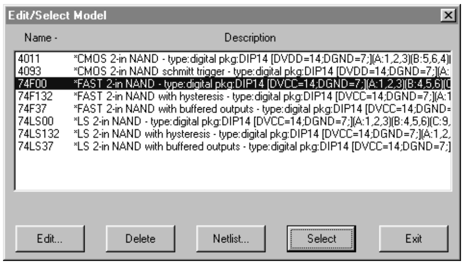
```
FREQ:   Fundamental frequency [1,]      2.000Meg
RS:     Series resistance [1,]          250.0
C:      Parallel capacitance [0,]       13.00p   *
Q:      Quality Factor [10,1000]        1.000k   *
```

Description: pkg:XTAL1 1,2

Figure 17.16. Editing the Crystal Subcircuit parameters.

Editing Digital Model Parameters

When you double-click a device that has digital simcode models associated with it, Figure 17.17 appears.



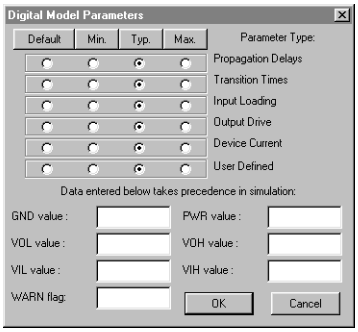
Edit/Select Model

Name -	Description
4011	*CMOS 2-in NAND - type:digital pkg:DIP14 [DVDD=14;DGND=7;](A:1,2,3)(B:5,6,4)(
4093	*CMOS 2-in NAND schmitt trigger - type:digital pkg:DIP14 [DVDD=14;DGND=7;](A:
74F00	*FAST 2-in NAND - type:digital pkg:DIP14 [DVCC=14;DGND=7;](A:1,2,3)(B:4,5,6)(
74F132	*FAST 2-in NAND with hysteresis - type:digital pkg:DIP14 [DVCC=14;DGND=7;](A:1
74F37	*FAST 2-in NAND with buffered outputs - type:digital pkg:DIP14 [DVCC=14;DGND=
74LS00	*LS 2-in NAND - type:digital pkg:DIP14 [DVCC=14;DGND=7;](A:1,2,3)(B:4,5,6)(C:9,
74LS132	*LS 2-in NAND with hysteresis - type:digital pkg:DIP14 [DVCC=14;DGND=7;](A:1,2,
74LS37	*LS 2-in NAND with buffered outputs - type:digital pkg:DIP14 [DVCC=14;DGND=7;]

Figure 17.17. Digital simcode devices are mixed-signal digital devices which you can simulate in analog mode using features of XSpice.

Digital simcode devices use event-driven behavioral models created for CircuitMaker. You cannot add new simcode models; however, simcode devices can be used in macro circuits to create new mixed-mode digital devices.

While you cannot edit the model itself, you can alter certain parameters of the model. To do so, click it with the mouse, then click **Edit** to display the dialog box in Figure 17.18.



Digital Model Parameters

Default	Min.	Typ.	Max.	Parameter Type:
○	○	●	○	Propagation Delays
○	○	●	○	Transition Times
○	○	●	○	Input Loading
○	○	●	○	Output Drive
○	○	●	○	Device Current
○	○	●	○	User Defined

Data entered below takes precedence in simulation:

GND value :
PWR value :
VOL value :
VOH value :
VIL value :
VIH value :
WARN flag:

Figure 17.18. Use the Digital Model Parameters dialog box to alter certain parameters of a model.

The radio buttons let you specify whether each of the parameters of the model will be minimum, typical or maximum values for the selected device(s). The Default setting allows the parameter to remain unchanged. All parameters can vary from pin to pin, part to part and family to family.

Parameter	**Description**
Propagation Delays	The time it takes for a signal change on an input to affect the data on the output.
Transition Times	The rise and fall times of the outputs.
Input Loading	The amount of load resistance that will be applied to the output of the driving device.
Output Drive	The amount of output current available.
Device Current	The amount of current drawn through the supply pin to ground.
User Defined	This parameter does not affect digital models provided with CircuitMaker.

The edit fields let you enter specific values for certain parameters. The values take precedence over family specific values. Under normal conditions they should be left blank.

Parameter	**Description**
GND & PWR	These two parameters must be programmed as a pair (if you set one, you must also set the other). Setting these voltages will override any other power and ground voltages specified for the selected devices.
VOL & VOH	These two parameters will override the family defaults.
VIL & VIH	These two parameters will override the family defaults.
WARN Flag:	When set to one (1), warning messages will be generated when there are timing or supply voltage violations on the device.

CHAPTER 18

Digital SimCode™

Digital SimCode™ provides the means for the electronics professional to create and/or modify digital components which operate in Analog/Mixed-Signal Simulation. Digital SimCode *cannot* be used to create components for Digital Logic Simulation. The functionality of devices for Digital Logic Simulation is built-in to CircuitMaker and you cannot add to it except by creating macro circuits.

Due to the complexity of digital devices, it is generally not practical to simulate them using standard, non-event-driven, Spice instructions. For this reason, CircuitMaker has its own special descriptive language that allows digital devices to work in Analog/Mixed-Signal Simulation using and extended version of the event-driven XSpice. The digital devices included in the CircuitMaker library are modeled using the Digital SimCode language.

The Digital SimCode language does *not* provide the means to create digital devices with analog characteristics such as analog switches, time-delayed one-shots, etc., nor can it be used to create new components for Digital Logic Simulation. Digital SimCode is a proprietary language created specifically for use with CircuitMaker and devices created with it are not compatible with other simulators, nor are digital components created for other simulators compatible with CircuitMaker.

Note: *Programming in the Digital SimCode language is not intended for the average CircuitMaker user. It requires a basic understanding of standard programming techniques as well as a complete understanding of the characteristics of the device that you are trying to create. These topics are not covered in this manual.*

Creating New SimCode Devices

Digital SimCode, used in conjunction with XSpice, is what allows digital components to be simulated in Analog/Mixed-Signal Simulation mode.

To create a new digital SimCode device,

1 Create a symbol to represent the device in your schematic.

See *Chapter 17: Creating New Devices* for more detailed information about the process.

Note: This step may not be necessary if you are creating a new device that has an equivalent function in another family. For example, if you are creating a 74F190, we recommend that you use the equivalent symbol for the 74190 for a couple of reasons. First, the function of the device is identical and the symbol has already been drawn. Second, if you want to use the component in Digital Logic Simulation mode, the existing symbol will work. New symbols that you create will not work in Digital mode unless they contain internal macro circuitry. This internal macro circuitry would disable the Digital SimCode model in Analog mode.

2 Create a .MOD file to be used with the symbol.

The .MOD file name must match the name of the symbol (see *Model and Subcircuit Linking Files* in *Chapter 17: Creating New Devices*). If you are using an existing symbol, you must use the existing .MOD file as well. Just add the new model information (see below) into the existing .MOD file.

Each Digital SimCode model declaration in a .MOD file has two lines. Line 1 is a comment line which contains information used by CircuitMaker. Line 2 contains the model information that is used by XSpice during simulation. This example is followed by a brief description of each item.

```
*Counter - type:digital pkg:DIP14 [DVCC=5;DGND=10;](6,7,2,3,14,1,11,8,9,12)
.MODEL A74LS90 xsimcode(file="{MODEL_PATH}LS.SCB" func=ls90 {mntymx})
```

Line 1	What it Does
Counter	Describes the device.
type:digital	Sets the device's Parameters field.
pkg:DIP14	Sets the device's Package field.
[DVCC=5;DGND=10;]	Sets device's Bus Data field (required if supply pins not on symbol).
(6,7,2,3,14,...)	Sets the device's Pin Data list.

Line 2	What it Does
.MODEL	Declares the model statement.
A74LS90	Names the model (digital SimCode models begin with the letter "A").
xsimcode	Model type for a digital SimCode model.
file=	Points to file containing the device's digital SimCode. {MODEL_PATH} is a shortcut to the Models directory as specified in the CircuitMaker preferences.
func=	Names the device's digital SimCode function.
data=	Contains ASCII data for the *READ_DATA* function (optional).
{mntymx}	Passes the device's Digital Model Parameters into SimCode (this must appear exactly as shown).

3 Create a digital SimCode model for the device.

You can do this in any ASCII text editor such as Notepad. If using a word processor, be sure to save the file in text only format. The file can be given any name and extension as long as it matches the file name listed in the "file=" parameter in the .MOD file. Multiple digital SimCode device models can be placed in the same file.

The simplest process is to start by making a copy of an existing device, preferably one which is similar in either function or characteristics, and make modifications to the SimCode as needed. Some examples of SimCode devices are included in a file called SIMCODE.TXT, for your reference.

4 Test the new device in CircuitMaker by creating a simple circuit to test its functionality.

Test only one new device at a time. When you run the simulation, the source code model is automatically compiled and the compiled code is placed in an ASCII text file called SIMLIST.TXT in the same directory as CIRMAKER.EXE. This file also contains a listing of the execution order of the source code model. Refine the SimCode source model as needed and continue testing until you've completely debugged the model.

5 Copy the compiled model file to your SimCode library.

Create a separate library file in which to place your compiled SimCode models. It does not matter what you name this file, but in order to be used, you must set the "file=" parameter in the .MOD file to be the same as the file name of the compiled SimCode model library.

SimCode models are stored in two types of files. The source models are stored in files with the .TXT extension (for example, LS.txt and S.txt) and the compiled models are stored in files with the .SCB extension (for example, Std.scb and Cmos.scb).

The 74LS74 Example

The following sections contain information about the 74LS74 Digital SimCode model example.

SimCode Function Identification

See ① in the example on page 17-6. **# ls74 source** identifies the beginning of the SimCode source function for the 74LS74.

Data declarations

See ② the example. This section consists of pin and variable declarations.

The *INPUTS* statement declares the names of the input pins. VCC and GND pins are included in this statement. The order of these pins must match the order of their corresponding pin declarations in the device's Spice Data field.

The *OUTPUTS* statement declares the names of the output pins. Notice that the input pins are listed here as well, but with the suffix "_LD". The input pins must also be declared as outputs so that the device can provide a load on the driving circuitry. VCC pins are included in this statement, but not GND pins. The order of these pins must match the order of their corresponding pin declarations in the device's Spice Data field.

The *PWR_GND_PINS* statement declares which pins will be used for device power and ground and samples their voltage levels for use later in the SimCode.

SimCode Function Initialization

See ③ in the example. The "*IF (init_sim) THEN*" section is executed only once, at the beginning of the simulation. In this section we set the device characteristics that are not subject to change due to outside influences such as databook specifications. The outputs states should also be initialized here to their "most likely" state. The *EXIT* command should be placed at the end of this section.

LOAD and DRIVE Statements

See ④ in the example. These statements are used to declare the load and drive capabilities of the device pins.

Device Functionality

See ⑤ in the example. This section can vary dramatically from part to part. In this example an *EXT_TABLE* command has been used. Other device models use a variety of *IF...THEN*, *STATE_BIT*, *NUMBER*, and other statements to define the logical function of the device.

Tests for Device Setup Violations

See ⑥ in the example. These tests warn of device setup violations which, in the real world, may cause a device not to function properly. In the simulation, the device will generally still function, but warnings, if enabled, will be displayed.

Output Delays/Post Events

See ⑦ in the example. The *DELAY* statements occur at the end of the SimCode function. These statements actually post the events to XSpice to let it know that something has changed and when these events are scheduled to occur relative to the rest of the simulation. Timing (propagation delay) is assigned to each output based on the databook specifications, input stimulus and the functionality of the device.

```
  //==============================================================
①# ls74 source
  //1/2- 74LS74 D flip-flop Digital Simcode Model
  //typical prop delay values from TI 1981 2nd edition data book
  //==============================================================
②INPUTS VCC, GND, PRE, DATA, CLK, CLR;
  OUTPUTS VCC_LD, PRE_LD, DATA_LD, CLK_LD, CLR_LD, QN, Q;
  INTEGERS tblIndex;
  REALS tplh_val, tphl_val, ts_val, th_val, trec_val, tt_val, temp_tp,
       clk_twl, clk_twh, pre_clr_twl, ril_val, rih_val, ricc_val;

  PWR_GND_PINS(VCC,GND);         //set pwr_param and gnd_param values
  SUPPLY_MIN_MAX(4.75,5.25);     //test for min supply=4.75 and max supply=5.25
  VOL_VOH_MIN(0.2,-0.4,0.1);     //vol_param=gnd_param+0.2,voh_param=pwr_param-0.4
  VIL_VIH_VALUE(1.25,1.35);      //set input threshold values: vil and vih
  IO_PAIRS(PRE:PRE_LD, DATA:DATA_LD, CLK:CLK_LD, CLR:CLR_LD);

③ IF (init_sim) THEN
    BEGIN               //select prop delay, setup, hold, and width times
  //MESSAGE("time\t\tPRE\tCLR\tCLK\tDATA\tQ\tQN"); //debug

      //NOTE: both ttlh and tthl are the same value
      tt_val= (MIN_TYP_MAX(tt_param: NULL, 5n,  NULL));

      temp_tp= (PWL_TABLE(sim_temp: -75, -5n, 125, 5n)); //tp temperature affect
      tplh_val= (MIN_TYP_MAX(tp_param: NULL, 14n, 25n)) + temp_tp;
      tphl_val= (MIN_TYP_MAX(tp_param: NULL, 20n, 40n)) + temp_tp;

      ts_val= (20n);
      th_val= (5n);
      trec_val= (5n);
      clk_twl= (25n);            //not specified - derived from fmax
      clk_twh= (25n);
      pre_clr_twl= (20n);

      //LS stdout drive IOL max=8mA @ VOL typ=0.35V:rol_param=0.35V/8mA=43.75
      //LS stdout drive IOL max=8mA @ VOL max=0.5V: rol_param=0.5V/8mA=62.5
      rol_param= (MIN_TYP_MAX(drv_param: 62.5, 43.75,  NULL));

      //LS stdout drive IOS min=20mA @ VCC max=5.25V: roh_param=5.25V/20mA=262.5
      //LS stdout drive IOS max=100mA @ VCC max=5.25V:roh_param=5.25V/100mA=52.5
      roh_param= (MIN_TYP_MAX(drv_param: 262.5, NULL, 52.5));

      //LS input load IIH max=20uA @ Vin=2.7V: ril= (2.7-vol_param)/20uA=125k
      ril_val= (MIN_TYP_MAX(ld_param: NULL, NULL, 125k));
      //LS input load IIL max=-0.4mA @ Vin=0.4V:rih= (voh_param-0.4)/0.4mA=10.5k
      rih_val= (MIN_TYP_MAX(ld_param: NULL, NULL, 10.5k));

      //Icc @ 5V: 2500= 4mA/2 typical, 1250= 8mA/2 max
      ricc_val= (MIN_TYP_MAX(i_param: NULL, 2500, 1250));

      STATE Q = ONE;            // initialize output states
      STATE QN = ZERO;
      EXIT;
    END;

④DRIVE Q QN = (v0=vol_param,v1=voh_param,ttlh=tt_val,tthl=tt_val);
  LOAD PRE_LD DATA_LD CLK_LD CLR_LD =
  (v0=vol_param,r0=ril_val,v1=voh_param,r1=rih_val,io=1e9,t=1p);

⑤EXT_TABLE tblIndex
  PRE CLR CLK DATA    Q     QN
  0   1   X   X       H     L
  1   0   X   X       L     H
  0   0   X   X       H     H
  1   1   ^   X       DATA  ~DATA
  1   1   X   X       Q     ~Q;

  //MESSAGE("%fs\t%d\t%d\t%d\t%d\t%d\t%d",present_time,PRE,CLR,CLK,DATA,Q,QN);
  LOAD VCC_LD = (v0=gnd_param,r0=ricc_val,t=1p);

⑥IF (warn_param) THEN
    BEGIN
      IF (PRE && CLR) THEN
        BEGIN
          SETUP_HOLD(CLK=LH DATA Ts=ts_val Th=th_val "CLK->DATA");
          RECOVER(CLK=LH PRE CLR Trec=trec_val "CLK->PRE or CLR");
          WIDTH(CLK Twl=clk_twl Twh=clk_twh "CLK");
          WIDTH(PRE CLR Twl= pre_clr_twl "PRE or CLR");
        END;
    END;

⑦DELAY Q QN =
    CASE (TRAN_LH) : tplh_val
    CASE (TRAN_HL) : tphl_val
  END;
  EXIT;
```

Editing Device Properties for SimCode Devices

Each device symbol includes netlist information specific to the device found in the Device Properties dialog box. For Digital SimCode devices, the information required is slightly different from other analog devices. For example, the Spice Data field must contain data not found in other devices.

The node list in the Spice Data field is divided into two sections, one for input nodes and one for output nodes. Each of these sections is delimited by square brackets ([ ]), input nodes first, followed by output nodes. The nodes must be listed in the *same order* as the pins in the *INPUTS* and *OUTPUTS* statements in the SimCode. The pin number used in the Spice Data field (for example, %1i, %2i, etc.) indicates the position of each pin in the devices Pin Data list. If that pin number is followed by the letter “b” (for example, %14bi), that indicates that the pin is found in the Bus Data field rather than on the symbol itself and represents the actual pin number on the package. The “i” and the “o” specify the pin type as input or output. For the 7474 (1/2 device symbol), the Spice Data field is set to:

```
%D [%14bi %7bi %1i %2i %3i %4i][%14o %1o %2o %3o %4o %5o %6o] %M
```

When using these SimCode pin declarations:

```
INPUTS VCC, GND, PRE, DATA, CLK, CLR;
OUTPUTS VCC_LD, PRE_LD, DATA_LD, CLK_LD, CLR_LD, QN, Q;
```

the items in the Spice Data field have the following meanings:

Item	**Meaning**
%D	Device Designation
%14bi	VCC (pin 14 on the actual package, declared in Bus Data field)
%7bi	GND (pin 7 on the actual package, declared in Bus Data field)
%1i	PRE input (1st pin in Pin Data list)
%2i	DATA input (2nd pin in Pin Data list)

%3i	CLK input (3rd pin in Pin Data list)
%4i	CLR input (4th pin in Pin Data list)
%14o	VCC_LD (still pin 14, but acts as load on VCC supply)
%1o	PRE_LD (load applied by PRE input)
%2o	DATA_LD (load applied by DATA input)
%3o	CLK_LD (load applied by CLK input)
%4o	CLR_LD (load applied by CLR input)
%5o	QN output (5th pin in Pin Data list)
%6o	Q output (6th pin in Pin Data list)
%M	A74LS74 (the name of the selected SimCode model)

Other information for the Edit Device Data dialog box includes:

Option	**Information**
Analog checkbox	Enabled
Label-Value	74LS74
Auto Designation Prefix	U
Spice Prefix Character	A
Bus Data	DVCC=14;DGND=7;
Parameters	type:digital

SimCode Language Definition

The following items make up the Digital SimCode language. The following pages described each of these items in detail.

INPUTS	Input pins (pins that monitor the circuit).
OUTPUTS	Output pins (pins that drive or load the circuit).
INTEGERS	Integer variables and arrays.
REALS	Real variables and arrays.
PWR_GND_PINS	Power and ground pins and record supply voltage.
IO_PAIRS	Input/output pin associations for input loading.

Device Setup Functions

Use these functions to set certain characteristics of the device pins.

VIL_VIH_VALUE	Sets absolute VIL and VIH values.
VIL_VIH_PERCENT	Sets VIL and VIH values to a percentage of supply voltage.
VOL_VOH_MIN	Sets VOH and VOL relative to power and ground.

Device Test Functions

Use these functions to test for any device setup violations which may occur in the circuit. These violations may not affect the simulation of the device's functionality (i.e., the device may continue to function in simulation even with setup violations). In order to know if any of these setup violations have occurred, you must enable warnings.

SUPPLY_MIN_MAX	Tests supply pins for min/max supply voltage violations.
RECOVER	Tests inputs for recovery time violations.
SETUP_HOLD	Tests inputs for setup and hold time violations.

WIDTH	Tests inputs for minimum pulse width violations.
FREQUENCY(FMAX)	Tests inputs for minimum & maximum frequency violation.

Output Pin Functions

Use these functions to program the output pins of a device.

STATE	Sets outputs to the declared logic state.
STATE_BIT	Sets outputs to binary weighted logic states.
LEVEL	Sets the level of the output state.
STRENGTH	Sets the strength of the output state.
TABLE	Sets output logic states based on truth table.
EXT_TABLE	Sets output logic states based on extended truth table.
LOAD	Declares loading characteristics of input pins.
DRIVE	Declares drive characteristics of output pins.
DELAY	Sets propagation delay to specified outputs.
NO_CHANGE	Leaves output state of I/O pins unchanged.
EVENT	Causes a digital event to be posted.

Expression Operations

Use these operators and functions in expressions to manipulate data and to make comparisons which control program flow. Expressions are always contained within parentheses ( ). Operator precedence is from left to right, starting with the inner most parentheses.

Operators	+, -, *, /, ~, !, &&, ||, ^^, &, |, >>, <<, >, <, =, !=, >=, <=
Math Functions	POW, ABS, SQRT, EXP, LOG, LOG10, SIN, COS, TAN, ASIN, ACOS, ATAN, HSIN, HCOS, HTAN

Expression Functions

Use these functions various expressions.

PARAM_SET	Determines if a predefined SimCode param has been set.
PWL_TABLE	Returns value from interpolative lookup table.
SELECT_VALUE	Returns value from simple lookup table.
MIN_TYP_MAX	Returns value from MIN_TYP_MAX lookup table.
NUMBER	Returns number based on binary weighted pin states.
VALUE	Returns state of the specified pin.
CHANGE_TIME	Returns time when the specified pin last changed state.
WIDTH_TIME	Returns last pulse width encountered on specified pin.
INSTANCE	Checks to see if this is the specified device instance.
CHANGED_xx	Checks to see if the specified pin has changed state.
READ_DATA	Reads data from an ASCII file into arrays.

Program Control

Use these statements to control the flow of the program.

# xxxx source	Identifies the beginning of the SimCode source function.
IF ... THEN	Conditionally controls flow through the SimCode.
WHILE ... DO	Conditionally controls looping in the SimCode.
GOTO	Jumps to a new location in the SimCode.
GOSUB	Jumps to a subroutine in the SimCode.
RETURN	Returns from a subroutine in the SimCode.
EXIT	Terminates SimCode execution.

Output Text

Use these commands to display messages during simulation and debugging.

PROMPT	Pause simulation and display a message.
MESSAGE	Display a message without pausing.

Debug

Use these commands to trace through the execution of the SimCode for debugging purposes.

STEP_ON	Turn on the SimCode trace mode
STEP_OFF	Turn off the SimCode trace mode

SimCode Language Syntax

This section describes each of the language items in detail. The following punctuations are used in describing the syntax:

italics	reserved words or emphasis
<>	value/variable/pin/expression
[ ]	optional parameter
{ }|{ }	selections (you must choose ONE of these parameters)

# xxxx source

Identifies the beginning of the SimCode source function.

General Form

```
# <func name> source
```

Parameters

<func name>	Name of the SimCode function.

Use

This statement identifies the SimCode function so that it can be called when it is time to simulate this device. It must be the first statement of each Digital SimCode device function.

Notes

XSpice has the ability to read either source code models or compiled code models. The keyword “source” identifies this as a source code model to be compiled by XSpice. When the simulation is run, the source code model is compiled and the compiled code is placed in an ASCII text file called SIMLIST.TXT in the same directory as CIRMAKER.EXE.

Example

```
//===================================
# MyDevice source
//===================================
INPUTS VCC, GND, IN1, IN2;
OUTPUTS VCC_LD, IN1_LD, IN2_LD, OUT;
.
.
.
EXIT;
```

CHANGE_TIME

Returns time when the specified pin last changed state.

General Form

```
CHANGE_TIME(<pin>)
```

Parameters

<pin>	Input or output pin name.

Use

This function returns a real value that indicates the last time the specified input or output pin changed states.

Example

```
T1 = (CHANGE_TIME(INA));
```

CHANGED_xx

Checks if the specified pin has changed state.

General Form

```
CHANGED_xx(<pin> [{<}|{<=}|{>}|{>=} <var/time/value>])
```

Parameters

<pin>	Input or output pin name.
<var/time/value>	Item to which <pin> is compared.

Use

The *CHANGED_xx* function is used to determine if the specified <pin> has changed state. The _xx that follows the keyword CHANGED can be eliminated (to indicate *any* type of change) or the xx can be set to:

LH, LX, HL, HX, XL, XH, LZ, ZL, ZH, ZX, HZ or XZ

to indicate a specific type of change. The optional compare operator (<, <=, >, >=) and <var/time/value> would be included to check for a more specific change. If they are not included, the function will return 1 if the pin has changed at the current simulation step.

Examples

```
IF (CHANGED_LH(CLK)) THEN ...
IF (CHANGED(DATA < 10n)) THEN ...
```

DELAY

Sets propagation delay to specified outputs.

General Form 1

```
DELAY <output> [<output> ...] = <delay>;
```

General Form 2

```
DELAY <output> [<output> ...] =
   CASE (<conditional exp>) : <delay>
   CASE (<conditional exp>) : <delay>
   [CASE (<conditional exp>) : <delay> ...]
END;
```

Parameters

<output>	Name of/variable index to the output pin.
<conditional exp>	Conditional expression that determines which delay is used.
<delay>	Propagation delay time to the output pin.

Use

The *DELAY* command is executed once for each pin listed and posts a propagation delay for each pin that has changed its level. The *CASE* option allows more than one <delay> to be specified. The <conditional exp> then determines which <delay> will be used. If a delay is set for a pin that has not changed then the pin will be flagged as NO-CHANGE and the delay will *not* be posted. The <delay> can be a real constant, a real variable or a real expression.

Notes

The *DELAY* command must be executed exactly once for each output pin, that is, for each pin declared in the *OUT-PUTS* statement which is *not* listed in the *LOAD* or *NO_CHANGE* statements. The order in which the delays are set is based on the order in which these pins are listed in the *DELAY* command (i.e. first pin listed is set first). Each <conditional expression> is evaluated in the order it is listed until one expression evaluates TRUE. When this occurs, the <delay> value associated with the TRUE expression is posted for the output being set. When using the *CASE* option, at least one <conditional exp> should evaluate as TRUE for each output pin listed. If no <conditional exp> evaluates to TRUE, the <delay> associated with the last *CASE* statement is posted.

In addition to the standard expression functions, the following terms apply *only* to the output pin being set and can be used in the <conditional exp> :

TRAN_LH	low-to-high
TRAN_LX	low-to-other
TRAN_HL	high-to-low
TRAN_HX	high-to-other
TRAN_HZ	high-to-tristate
TRAN_XL	other-to-low
TRAN_XH	other-to-high
TRAN_LZ	low-to-tristate
TRAN_ZL	tristate-to-low
TRAN_ZH	tristate-to-high
TRAN_ZX	tristate-to-other
TRAN_XZ	other-to-tristate
TRAN_XX	other-to-different

If the <delay> value is less than or equal to 0.0 a run-time error message will be displayed. Output pins can be specified by using the output pin name or by an integer variable the contains the *index* of an output pin. Pin names and variables *cannot* be mixed in the same *DELAY* statement. References to outputs must be either all pin names or all variable names.

Examples

```
DELAY Q1 Q2 Q3 Q4 = 10n;
DELAY Q QN =
   CASE (TRAN_LH) : tplh_val
   CASE (TRAN_HL) : tphl_val
END;


data = (E0_1 && (CHANGED(D0) || CHANGED(D1)));
DELAY Q1 Q0 =
   CASE (data && TRAN_LH) : tplh_D_Q
   CASE (data && TRAN_HL) : tphl_D_Q
   CASE (TRAN_LH) : tplh_E_Q
   CASE (TRAN_HL) : tphl_E_Q
END;
```

In this example, if data is nonzero and Q1 is changing from High to Low, the tphl_D_Q delay will be posted for Q1. Then, if Q0 is changing from Low to High, the tplh_D_Q delay will be posted for Q0.

DRIVE

Declares drive characteristics of output pins.

General Form

```
DRIVE <output> [<output> ...] =
   (v0=<value> v1=<value> ttlh=<value> tthl=<value>);
```

Parameters

<output>	Name of or variable index to the output pin.
<value>	Real value or variable.
v0	VOL for the output pin.
v1	VOH for the output pin.
ttlh	Low-to-high transition time for the output pin.
tthl	High-to-low transition time for the output pin.

Use

The *DRIVE* command is used to declare the output pin's DRIVE characteristics. When the output is set to a LOW state, the output pin is connected to voltage value *v0* through resistance *rol_param*. When the output is set to a HIGH state, the output pin is connected to voltage value *v1* through resistance *roh_param*. The low-to-high transition time is set by *ttlh* and the high-to-low transition time is set by *tthl*.

Notes

Pin names and variables *cannot* be mixed in the same *DRIVE* statement. References to outputs must be either all pin names or all variable names.

The values used for *rol_param* should be derived using the databook specs for VOL. This value represents the total saturation resistance of the pull-down structure of the device's output. A standard LS output in the LOW state, for example, sinking 8mA will not exceed 0.5V, typically closer to 0.35V. Therefore:

for typ LOW state drive:	*rol_param* = VOLtyp / IOLmax
	rol_param = 0.35V / 8mA
	rol_param = 43.75 ohms
for min LOW state drive:	*rol_param* = VOLmax / IOLmax
	rol_param = 0.5V / 8mA
	rol_param = 62.5 ohms

The values used for *roh_param* should be derived using the databook specs for IOS, if available. This value represents the total saturation resistance of the pull-up structure of the device's output. A standard LS output in the HIGH state with the output shorted to ground and Vcc=5.25V will source at least 20mA but not more than 100mA. Therefore:

for min HIGH state drive:	*roh_param* = VCCmax / IOSmin
	roh_param = 5.25V / 20mA
	roh_param = 262.5 ohms
for max HIGH state drive:	*roh_param* = VCCmax / IOSmax
	roh_param = 5.25V / 100mA
	roh_param = 52.5 ohms

Example

```
rol_param = (MIN_TYP_MAX(drv_param: 62.5, 43.75, NULL);
roh_param = (MIN_TYP_MAX(drv_param: 262.5, NULL, 52.5);
DRIVE Q QN = (v0=vol_param,v1=voh_param,ttlh=ttlh_val,
   tthl=tthl_val);
```

See Also

LOAD

EVENT

Causes a digital event to be posted.

General Form

```
EVENT = ({<time>}|{<expression>})
```

Parameters

<time>	Time at which event should occur.
<expression>	Expression indicating time at which event should occur.

Use

In most cases a digital event is posted when one or more INPUT pins for a simcode model changes state. When the event is processed, the simcode for the specified event is called and run. This instruction allows a simcode model to post a digital event at a specified <time>. If the specified EVENT time is greater than the simulation time (indicated by present_time), then a digital event will be posted. If more than one EVENT is posted in a single call to a simcode model, only the longest EVENT <time> will be used. This function allows the creation of one-shots and other similar device models.

Notes

If a digital event for a specific simcode model occurs before an EVENT <time> posted by that simcode, the EVENT <time> must be posted again. For example, if 1) the present simulation time is 1us, 2) a simcode model sets EVENT = 2us and 3) an INPUT pin in the simcode model changes state at 1.5us, then the 2us event must be posted again.

Example

```
EVENT = (present_time + 1e-6);   //return in
1us
```

EXIT

Terminates SimCode execution.

General Form

```
EXIT;
```

Use
The *EXIT* statement is used to terminate SimCode execution.

Notes
This is the last line of a SimCode model, but it may also be placed at other locations to abort execution of remaining SimCode.

EXT_TABLE
Sets output logic states based on extended truth table.

General Form
```
EXT_TABLE <line>
<input pin> [<input pin> ...] <output pin> [<output pin> ...]
<input state> [<input state> ...]
   <output state> [<output state> ...];
```

Parameters

<line>	Variable into which the line number used in the table is placed
<input pin>	Name of the input pin
<output pin>	Name of the output pin
<input state>	State of the individual inputs
<output state>	State of the individual outputs based on input conditions

Use
The *EXT_TABLE* statement is an extended truth table function used to set the level and strength of the specified outputs. Valid input states are:

0	low (input voltage is <= *vil_param*)
1	high (input voltage is >= *vih_param*)
^	low-to-high-transition
v	high-to-low-transition
X	don’t care what input voltage is

Valid output states are:

L	ZERO (set output level to *vol_param*).
H	ONE (set output level to *voh_param*).
Z	UNKNOWN (set output level to *v3s_param*).

It also allows INPUT and/or OUTPUT pin names with optional prefixes to specify the output states. Prefixes are:

-	State is the previous state.
~	State is the inverse of the state.
-~	State is the inverse of the previous state.

Output state letters can be followed by a colon and a letter to indicate strength:

s	STRONG (set output to *rol_param* for L and *roh_param* for H).
z	HI_IMPEDANCE (set output to *r3s_param*).

If a strength character is *not* specified after an output state then STRONG will be used for L and H states and HI_IMPEDANCE will be used for Z states.

Notes

Each row is tested sequencially from top to bottom until the input conditions are met. The outputs are set for the *first* row to meet the input conditions. <line> is set to the line number in the table that was used. If no match was made then <line> is set to 0. Pin names used to specify output states do not need to be in the table heading. Unlike the *TABLE* statement, input variables are not allowed.

Example

```
EXT_TABLE tblIndex
PRE    CLR    CLK    DATA   Q      QN
0      1      X      X      H      L
1      0      X      X      L      H
0      0      X      X      H      H
1      1      ^      X      DATA   ~DATA
1      1      X      X      Q      ~Q;
```

This example is representative of 1/2 of a 7474 D type flip-flop. If input pins PRE, CLR, and DATA are all high (>= *vih_param*) and CLK has a low-to-high transition, Q is set to high (*voh_param)* and STRONG (*roh_param*), QN is set to low (*vol_param)* and STRONG (*rol_param*) and tblIndex is set to 4.

See Also

REALS, STATE, STATE_BIT, TABLE

FREQUENCY (FMAX)

Tests inputs for minimum and maximum frequency violation.

General Form

```
FREQUENCY(<input> [<input>...] MIN=<frequency>
   MAX=<frequency> ["<message>"])
```

Parameters

<input>	Name of or variable index to the input pin under test.
MIN	Minimum frequency allowed on the pin under test.
MAX	Maximum frequency allowed on the pin under test.
<message>	Text string that will be displayed if a warning occurs.

Use

The *FREQUENCY* function compares the <input> period (the time from one low-to-high edge to the next low-to-high edge) with the reciprical of the specified <frequency> (1/ freq). If the time period for the <input> is smaller than the reciprical of the specified MAX<frequency> or the time period for the <input> is greater than the reciprocal of the specified MIN<frequency>, then a WARNING will be displayed. An optional <message> string can be included in the *FREQUENCY* statement which will be output if a WARNING is displayed.

Notes

Databook specifications should be used with this function. Pin and variable names *can* be mixed in the same *FREQUENCY* statement. Only the first FREQUENCY failure for each pin listed will be reported.

Example

```
FREQUENCY(CLK MAX=10MEG "CLK"); //check fmax
only
```

GOSUB

Jumps to a subroutine in the SimCode.

General Form

```
GOSUB  <label>;
```

Parameters

<label>	Location in SimCode where program flow resumes.

Use

The *GOSUB* instruction is used to perform non-sequencial execution of the SimCode. However, unlike the *GOTO* statement, SimCode execution will continue on the instruction following the *GOSUB* instruction when a *RETURN* instruction is encountered in the SimCode being run.

Example

```
GOSUB  Shift_Left;
.
.
.
Exit;

Shift_Left:
.
.
.
RETURN:
```

See Also

RETURN

GOTO

Jumps to a new location in the SimCode.

General Form

```
GOTO <label>;
```

Parameters

<label>	Location in SimCode where program flow resumes.

Use

The *GOTO* instruction is used to perform non-sequencial execution of the SimCode.

Notes
Program flow resumes from the location where <label>: appears in the SimCode. <label> must being with an alpha character, followed by any number of alpha-numeric characters or the underscore (_) character. Where <label> appears in the code, it must be followed *immediately* by a colon (:).

Example
```
GOTO  Shutdown;
.
.
.
Shutdown:
.
.
.
Exit;
```

See Also
GOSUB, IF ... THEN

IF ... THEN
Conditionally controls flow through the SimCode.

General Form 1
```
IF (<expression>) THEN BEGIN ... [ELSE ...] END;
```

General Form 2
```
IF  (<expression>)  THEN  GOTO  <label>;
```

Parameters
<expression>	Any expression that can be evaluated as true or false.
<label>	Location in SimCode where program flow resumes.

Use
The *IF ... THEN* statement is used to control the flow of the program, based on whether <expression> evaluates to true or false. Multiple *IF ... THEN* statements may be nested.

Notes
When the BEGIN...ELSE...END form of this statement is used and <expression> evaluates to true, program flow resumes from the BEGIN statement and skips any optional SimCode between the ELSE and END statements. If <expression>

evaluates to false, program flow resumes from the optional ELSE statement if it exists or after the END statement if it does not exist.

When the GOTO form of this statement is used and <expression> evaluates to true, program flow resumes from the location where <label>: appears in the SimCode. <label> must being with an alpha character, followed by any number of alpha-numeric characters or the underscore (_) character. Where <label> appears in the code, it must be followed *immediately* by a colon (:).

Examples

```
IF (EN) THEN
   BEGIN
      STATE Q0 = UNKNOWN;
   ELSE
      IF (IN2) THEN
        BEGIN
           STATE Y2 = ONE;
        ELSE
           STATE Y2 = ZERO;
        END;
   END;

IF (x = -2) THEN GOTO Do_Neg2;
.
.
.
Do_Neg2:
.
.
.
```

See Also

GOTO, WHILE ... DO

INPUTS

Declares input pins (pins that monitor the circuit).

General Form

```
INPUTS <input pin>[, <input pin>, ...];
```

Parameters

<input pin>	Name of the input pin.

Use

The *INPUTS* data type is used to define the pins which monitor stimulus external to the device. These generally include input, i/o, power and ground pins.

Notes

Input pin names *must* begin with a letter and be defined before they are used.

Example

```
INPUTS VCC, GND, PRE, DATA, CLK, CLR;
```

See Also

OUTPUTS, IO_PAIRS, PWR_GND_PINS

INSTANCE

Checks if this is the specified device instance.

General Form

```
INSTANCE("<instance name>")
```

Parameters

<instance name>	Text string indicating instance name.

Use

The *INSTANCE* function returns 1 if the present instance of the SimCode device matches the <instance name> specified. Otherwise it returns 0;

Notes

A circuit may contain more than one of any given device. During simulation it may be important to know if the device being simulated at this moment is the one you are interested in. This would allow you, for example, to print messages for one specific NAND gate without having to wade through messages for all the other NAND gates as well. The instance name is the device Designation preceded by its Spice Prefix Character (the letter A).

Example

```
IF (INSTANCE("AU23")) THEN
   BEGIN
      MESSAGE("U23-Q0 = %d", Q0);
   END;
```

INTEGERS

Declares integer variables and arrays.

General Form

```
INTEGERS <var>[, <var>, ...];
```

Parameters

<var>	Name of the variable.

Use

The *INTEGERS* data type is used to define integer variables and arrays.

Notes

Integer variables and arrays *must* begin with a letter and be defined before they are used. Integer arrays are defined by following the array name with a left bracket ([ ), an integer number which defines the size of the array, and a right bracket ( ]). Integer arrays can be set and/or used in expressions.

The following are reserved SimCode integer variables which do not need to be declared:

Variable	Use	Digital Model Parameter	Spice Option
tp_param	tplh/hl index	Propagation Delays	TPMNTYMX
tt_param	ttlh/hl index	Transition Times	TTMNTYMX
ld_param	LOAD index	Input Loading	LDMNTYMX
drv_param	DRIVE index	Output Drive	DRVMNTYMX
i_param	ICC index	Device Current	IMNTYMX
user_param	USER index	User Defined	USERMNTYMX
warn_param	Warning messages	WARN flag	SIMWARN
init_sim	1 during simcode init	N/A	N/A
tran_pin	TRAN_xx pin index	N/A	N/A

The first six variables in this list are expected to have a value of 1, 2 or 3. These values represent an index into the min/typ/max arrays:

Value	**Represent**
1	Index to minimum value.
2	Index to typical value.
3	Index to maximum value.

The Digital Model Parameter can be set independently for each digital device in the Digital Model Parameters dialog box. If a Spice Option parameter is set in the Analog Options dialog box, that setting will globally override the Digital Model Parameter settings for all digital devices. If the variable is set explicitly in the SimCode, that setting will override all other settings.

warn_param can be set to any positive value to condtionally display warning messages for the device. Different levels of warning could be created by the device programmer, accessed by entering different positive values. The value of *init_sim* is set 1 during SimCode initialization, otherwise it is set to 0. The value of *tran_pin* is set to the index of the pin being set during a *DELAY CASE* statement. This index is used to determine which pin the *TRAN_xx* instruction is applied to.

Example

```
INTEGERS tblIndex, count, data[64];
```

See Also

DELAY, MIN_TYP_MAX

IO_PAIRS

Declares input/output pin associations for input loading.

General Form

```
IO_PAIRS (<ipin:opin>[, <ipin:opin>, ...]);
```

Parameters

<ipin:opin>	Pin names of associated input and output pins.

Use

The *IO_PAIRS* statement defines which of the *INPUTS* pins are associated with which of the *OUTPUTS* pins. This association is used by the *LOAD* statement.

Notes

Each physical input pin on a device consists of both an ipin and an opin in SimCode. The opin is required to provide input loading characteristics. This statement can only be used once in the SimCode. Power pins are not listed in the *IO_PAIRS* statement.

Example

```
IO_PAIRS (IN1:IN1_LD, IN2:IN2_LD);
```

In this example, IN1 and IN2 are *INPUTS* and IN1_LD and IN2_LD are *OUTPUTS*. IN1 and IN1_LD both refer to the same physical pin on the device.

See Also

INPUTS, OUTPUTS, LOAD

LEVEL

Sets the level of the output state.

General Form 1

```
LEVEL <output> [<output> ...] = (<expression>);
```

General Form 2

```
LEVEL  <output>  [<output>  ...]  =
   {ZERO}|{ONE}|{UNKNOWN};
```

Parameters

<output>	Name of or variable index to the output.
<expression>	Any expression compared to VOL or VOH.

Use

The state of an output pin is determined by its level and its strength. Use the *LEVEL* command to set the level of one or more output pins.

<expression>	**State**	**Level**
<= *vol_param*	ZERO	*vol_param*
>= *voh_param*	ONE	*voh_param*
other	UNKNOWN	*v3s_param*

Notes

Output pins can be specified by using the output pin name or by an integer variable the contains the INDEX of an output pin. Pin and variable names *cannot* be mixed in the same *LEVEL* statement. References to outputs must be either all pin names or all variable names.

Examples

```
LEVEL  Q  =  ONE;
LEVEL  Q1  Q2  Q3  Q4  =  ZERO;
LEVEL  OUT  =  ((1+2)/3);
```

In the last example, OUT will be:

ZERO if *vil_param* > 1
UNKNOWN if *vil_param* < 1 and *vih_param* > 1
ONE if *vih_param* < 1

See Also

REALS, STATE, STATE_BIT, STRENGTH

LOAD

Declares loading characteristics of input pins.

General Form

```
LOAD  <output>  [<output>  ...]  =
   (v0=<value>  r0=<value>  [v1=<value>
   r1=<value>]  [io=<value>]  t=<value>);
```

Parameters

<output>	Name of or variable index to the output pin.
<value>	Real value or variable.
v0	Load voltage for HIGH state input.
r0	Load resistance for HIGH state input.
v1	Load voltage for LOW state input.
r1	Load resistance for LOW state input
io	Off-state load resistance for unused load.
t	Time delay before the load will be applied.

Use

The *LOAD* command is typically used with input or power pins to provide loading for the driving circuit. Since only output pins can provide a load, each input must have a corresponding output. These are assigned using the *IO_PAIRS* statement.

If different loads are required for different inputs, multiple *LOAD* statements may be used. Power pins should be placed in a separate *LOAD* statement which does not include the *v1/r1* load or *io*. Power pins are not included in the *IO_PAIRS* statement. The *IO_PAIRS* statement must be entered before any *LOAD* statements that contain *io*.

Notes

An input load consists of a voltage and a resistance (*v0/r0* or *v1/r1*). The voltage level of the incoming signal determines which load will be used. If the voltage level goes below VIL and remains below VIH, then the input is considered to be in the LOW state and the *v1/r1* is applied. If the voltage level goes above VIH and remains above VIL, then the input is considered to be in the HIGH state and the *v0/r0* is applied. *io* is the input state off resistance. The unused load is essentially removed from the circuit by changing its *r* value to the value specified for *io*.

The values for *v0*, *r0*, *v1* and *r1* can be either real constants or real variables. The values for *io* and *t* must be real constants. Pin names and pin variables *cannot* be mixed in the same *LOAD* statement. References to outputs must be either all pin names or all variable names.

For input pins, the values used for *r0* should be derived using the databook specs for IIH. A standard LS input, for example, will sink a maximum of 20uA at Vin=2.7V. Therefore, if *vol_param* = 0.2V, then:

for max HIGH state load:	*r0* = (Vin - *vol_param*) / IIHmax
	r0 = (2.7V - 0.2V) / 20uA
	r0 = 125k ohms

The values used for *r1* should be derived using the databook specs for IIL. A standard LS input, for example, will source a maximum of 400uA at Vin=0.4V. Therefore, if *voh_param* = 4.6V then:

for max LOW state load:	*r1* = (*voh_param* - Vin) / IILmax
	r1 = (4.6V - 0.4V) / 400uA
	r1 = 10.5k ohms

For power pins, the value used for *r0* should be derived using the databook specs for ICC. For a 74LS151, Icc typ is 6mA at Vcc=5V and Icc max is 10mA at Vcc=5.25V. Therefore:

for Icc typ:	*r0* = 5V / 6mA = 833 ohms
for Icc max:	*r0* = 5.25V / 10mA = 525 ohms

If creating a multiple-parts-per-package device, such as a 74LS00 quad NAND gate, you should adjust the Icc load for the individual parts accordingly.

Examples

```
r0_val = (MIN_TYP_MAX(ld_param: NULL, NULL, 125k);
r1_val = (MIN_TYP_MAX(ld_param: NULL, NULL, 10.5k);
ricc_val = (MIN_TYP_MAX(ld_param: NULL, 833, 525);
LOAD PRE_LD DATA_LD CLK_LD CLR_LD = (v0=vol_param, r0=r0_val,
         v1=voh_param, r1=r1_val, io=1e9, t=1p);
LOAD VCC_LD = (v0=gnd_param, r0=ricc_val, t=1p);
```

See Also

IO_PAIRS, DRIVE

MATH FUNCTIONS

Function	Description	Example
POW	power	`X= (12 POW(3));`
ABS	absolute value	`X= (ABS(-12));`
SQRT	square-root	`X= (SQRT(2));`
EXP	exponent	`X= (EXP(10));`
LOG	natural log	`X= (LOG(0.1));`
LOG10	log base 10	`X= (LOG10(0.1));`
SIN	sine	`X= (SIN(0.1));`
COS	cosine	`X= (COS(0.1));`
TAN	tangent	`X= (TAN(0.1));`
ASIN	arc sine	`X= (ASIN(0.1));`
ACOS	arc cosine	`X= (ACOS(0.1));`
ATAN	arc tangent	`X= (ATAN(0.1));`
HSIN	hyperbolic sine	`X= (HSIN(0.1));`
HCOS	hyperbolic cosine	`X= (HCOS(0.1));`
HTAN	hyperbolic tangent	`X= (HTAN(0.1));`

See Also

OPERATORS

MESSAGE

Displays a message without pausing.

General Form

```
MESSAGE("<message>"[, <value/pin>...]);
```

Parameters

<message>	Message string including formatting characters as needed.
<value>	Variable or constant value.
<pin>	Pin name or index to pin variable.

Use

The *MESSAGE* statement is used to output the information specified by the <message> string. It does *not* interrupt the simulation. The message is displayed in the XSpice window during simulation.

Notes

A format string in *MESSAGE* is similar to a format that may be used in a printf statement in C. Valid formatting characters include (but are not limited to):

\t	tab
\n	new line
\r	carraige return
%d	Deciaml display for short variable or current input output state
%D	Decimal display for short variable or old input/ output state
%x	Hex display for short variable or current input/ output state
%X	Hex display for short variable or old input/output state
%c	Character display for short variable or current input/output state
%C	Character display for short variable or old input/ output state

%e	Exponential display for real variable
%f	Floating point engineering display for real variable
%g	Short display (%e or %f) for real variable
%s	String constant display. The *only* valid string constants are:

INSTANCE	The present SimCode device instance name.
FUNC	The present SimCode device function name.
FILE	The present SimCode device file name.

Examples

```
MESSAGE("time\t\tCLK\tDATA\tQ\tQN");
MESSAGE("device  instance=  %s",INSTANCE);
MESSAGE("%.3e\t%d\t%d\t%d\t%d",present_time,CLK,DATA,Q,QN);
```

See Also
PROMPT

MIN_TYP_MAX
Returns value from MIN_TYP_MAX look-up table.

General Form
```
MIN_TYP_MAX(<index>:  <min>,  <typ>,  <max>);
```

Parameters

<index>	Input variable (index to select min, typ or max values).
<min>	Minimum databook value.
<typ>	Typical databook value.
<max>	Maximum databook value.

Use
The *MIN_TYP_MAX* function is similar to the *SELECT_VALUE* function except that three (3) values/ variables *must* be entered. The keyword “NULL” can be

substitued for one or two unknown values. If a predefined integer variable (see *INTEGERS*) is used as the <index>, unknown (NULL) values are calculated from the known values as follows:

Known Values	Formula
<min>,<max>	typical = (<max> + <min>) / 2;
<min> only	typical = (<min> / <min scale factor>)
	maximum = (<min> / <min scale factor>) * <max scale factor>
<typ> only	minimum = (<typ> * <min scale factor>)
	maximum = (<typ> * <max scale factor>)
<max> only	minimum = (<max> / <max scale factor>) * <min scale factor>
	typical = (<max> / <max scale factor>)

Notes

If <index> is not one of the predefined variables listed below, then <min scale factor> = 0.5 and <max scale factor> = 1.5. The <min scale factor> and <max scale factor> for each of these predefined variables can be changed in CircuitMaker's Analog Options dialog box. The <min scale factor> and <max scale factors> are reversed for *ld_param*, *drv_param* and *i_param* because these parameters control a resistance value rather than a current value (i.e., maximum load equates to minimum resistance.)

Variable	Spice Option	Parameter	Default
tp_param	PROPMNS	<min scale factor>	0.5
tp_param	PROPMXS	<max scale factor>	1.5
tt_param	TRANMNS	<min scale factor>	0.5
tt_param	TRANMXS	<max scale factor>	1.5
ld_param	LOADMNS	<min scale factor>	1.5
ld_param	LOADMXS	<max scale factor>	0.5
drv_param	DRIVEMNS	<min scale factor>	1.5
drv_param	DRIVEMXS	<max scale factor>	0.5
i_param	CURRENTMNS	<min scale factor>	1.5

i_param	CURRENTMXS	<max scale factor>	0.5
vth_param	VTHMNS	<min scale factor>	0.5
vth_param	VTHMXS	<max scale factor>	1.5
user_param	USERMNS	<min scale factor>	0.5
user_param	USERMXS	<max scale factor>	1.5

Examples

```
tplh_val = (MIN_TYP_MAX(tp_param: NULL, 5n, NULL));
```

In this example, if we assume that PROPMNS and PROPMXS are set to their default values, then:

if *tp_param* = 1, then tplh_val = 2.5n

if *tp_param* = 2, then tplh_val = 5n

if *tp_param* = 3, then tplh_val = 10n

```
ricch_val = (MIN_TYP_MAX(i_param: NULL, 2500, 1250));
```

In this example, if we assume that CURRENTMNS and CURRENTMXS are set to their default values, then:

if *i_param* = 1, then ricch_val = 5000

if *i_param* = 2, then ricch_val = 2500

if *i_param* = 3, then ricch_val = 1250

See Also

INTEGERS, SELECT_VALUE

NO_CHANGE

Leaves output state of I/O pins unchanged.

General Form

```
NO_CHANGE <output> [<output> ...];
```

Parameters

<output>	Name of or variable index to the output pin.

Use

Use the *NO_CHANGE* function to indicate no-change for specified output pins. Use this statement on bi-directional pins when the bi-directional pin is being treated as an input.

Notes

Pin names and variables *cannot* be mixed in the same *NO_CHANGE* statement. References to outputs must be either all pin names or all variable names.

Example

```
NO_CHANGE Q1 Q2 Q3 Q4;
```

NUMBER

Returns number based on binary weighted pin states.

General Form

```
NUMBER(<MSB pin>, [<pin>,   ...] <LSB pin> );
```

Parameters

<pin>	Name of or index to a pin.

Use

The *NUMBER* function returns a short integer that represents the decimal value of the binary number represented by the list of <pin>. Each bit (represented by a <pin>) is set to 1 if the <pin> is non-zero, otherwise it is set to 0.

Notes

The first <pin> in the list represents the most-significant-bit (MSB) and the last <pin> in the list represents the least-significant-bit (LSB).

Example

```
A = (NUMBER(D3,D2,D1,D0));
```

In this example, if D3 is HIGH, and D2, D1 and D0 are LOW (1000_2), then A = 8.

OPERATORS

Assignment Operator

=	Equals (sets a variable or output pin to a value or state).

Math Operators

+	Add
-	Subtract
*	Multiply
/	Divide

Unary Operators

~	Logical not
!	Bitwise complement

Logical Operators

&&	AND
||	OR
^^	XOR

Bitwise Operators

&	AND
|	OR
^	XOR
<<	Shift left
>>	Shift right

Relative Comparators

=	Equal
!=	Not equal
<	Less than
<=	Less than or equal to
>	greater than
>=	greater than or equal to

Use

Operators are used to set and manipulate variables and expressions.

Notes

Expressions must enclosed within parenthases ( ). Expressions are *always* evaluated from left to right within parenthases. You should use parenthases to set precedence within an expression. When using the Unary Operators on values, variables, expressions, etc. the values, variables, expressions, etc. must be in parenthases ( ).

Examples

```
clk_twl = (25n);
reg = (reg + 1);
vx = (vol_param - 10m);
C = (A * B);
val = (xval / 2);
X = (A && ~(B));
Y = (!(X));                  //if X=1 then Y=FFFFFFFE
A = (X & 1);                 //if X=1 then A=1, if X=2 then A=0
B = (X | 8);                 //if X=1 then B=9, if X=2 then B=10
C = (X >> 2);                //if X=1 then C=0, if X=2 then C=0
D = (2 >> X);                //if X=1 then D=1, if X=2 then D=0
E = (X << 2);                //if X=1 then E=4, if X=2 then E=8
F = (2 << X);                //if X=1 then F=4, if X=2 then F=8
IF (A >= B) THEN ...
IF ((A < 2) && (B > 3)) THEN ...
IF ((C < 2) || (X > 4)) THEN ...
```

See Also

MATH FUNCTIONS

OUTPUTS

Declares output pins (pins that drive or load the circuit).

General Form

```
OUTPUTS  <output  pin>[,  <output  pin>,  ...];
```

Parameters

<output pin>	Name of the output pin.

Use

The *OUTPUTS* data type is used to define the pins which affect the operation of circuitry external to the device. These generally include input, output, I/O and power pins. Input and power pins are included in this list because their presence contitutes a load on the driving circuitry.

Notes

Output pin names *must* begin with a letter and be defined before they are used.

Example

```
OUTPUTS  VCC_LD,  PRE_LD,  DATA_LD,  CLK_LD,
   CLR_LD,  QN,  Q;
```

See Also
INPUTS, IO_PAIRS, PWR_GND_PINS

PARAM_SET
Determines if a predefined SimCode param has been set.

General Form

```
PARAM_SET(<param  var>)
```

Parameters

<param var>	SimCode model definition parameter.

Use
The *PARAM_SET* function is used to determine if a parameter in the SimCode model definition has been set. It returns 1 if the specified parameter was set (e.g., vil_param=0.8) otherwise it returns 0.

Notes
See *INTEGER* and *REAL* declarations for a list of SimCode model definition parameters and their associated variable names.

Example

```
A  =  PARAM_SET(ld_param);
IF  (PARAM_SET(voh_param))  THEN  ...
```

See Also
INTEGERS, REALS

PROMPT

Pauses simulation and displays a message.

General Form

```
PROMPT("<message>"[, <value/pin>...]);
```

Parameters

<message>	Message string including formatting characters as needed.
<value>	Variable or constant value.
<pin>	Pin name or index to pin variable.

Use

The *PROMPT* statement is used to stop simulation and display the information specified by the <message> string. The message is displayed in the XSpice window during simulation. The user must click on a button to continue execution of the SimCode.

Notes

A format string in *PROMPT* is similar to a format that may be used in a printf statement in C. Valid formatting characters include (but are not limited to):

\t	tab
\n	new line
\r	carraige return
%d	Deciaml display for short variable or current input/output state.
%D	Decimal display for short variable or old input/output state.
%x	Hex display for short variable or current input/output state.
%X	Hex display for short variable or old input/output state.
%c	Character display for short variable or current input/output state.

%C	Character display for short variable or old input/ output state.
%e	Exponential display for real variable.
%f	Floating point engineering display for real variable.
%g	Short display (%e or %f) for real variable.
%s	String constant display. The *only* valid string constants are:

INSTANCE	The present SimCode device instance name.
FUNC	The present SimCode device function name.
FILE	The present SimCode device file name.

Example

```
PROMPT("input=%d  time=%f device=%s", D1, t1, INSTANCE);
```

See Also

MESSAGE

PWL_TABLE

Returns value from interpolative look-up table.

General Form

```
PWL_TABLE (<IN var>:
  <IN1>,<OUT1>,<IN2>,<OUT2>[,...<OUTn>,<OUTn>])
```

Parameters

<IN var>	input variable (integer or real)
<INx>	input compare value
<OUTx>	output value at <INx>

Use

This piece-wise-linear function is essentially a look-up table. The value of <IN var> is used to look up an entry in the table which consists of pairs of values. The first value in each pair is an input compare value and the second value is the corresponding output value. If the <IN var> value is less than the first <IN> value the first <OUT> value is returned. If the <IN var> value is greater than the last <INn> value then the last <OUTn> value is returned. Linear interpolation is done between enteries according to the formula:

value = (((OUTA-OUTB)/(INA-INB))*(<IN var>-INA)+OUTA)

where <IN var> falls between the input compare values INA and INB. The actual output value will fall between output values OUTA and OUTB.

Notes

Two or more IN/OUT data value pairs must be entered and the IN values must be entered in assending order. There is no limit to the maximum number of IN/OUT data pairs that can be entered.

Example

```
twh = (PWL_TABLE(var: 5,180n,10,120n,15,80n));
```

In this example, if var = 10 then twh = 120n and if var = 12 then twh = 104.

See Also

SELECT_VALUE

PWR_GND_PINS

Declares power and ground pins; records supply voltage.

General Form

```
PWR_GND_PINS (<pwrpin>, <gndpin>);
```

Parameters

<pwrpin>	name of the power pin
<gndpin>	name of the ground pin

Use

The *PWR_GND_PINS* statement defines which of the *INPUTS* pins are power and ground and sets the Power and Ground parameters of the device to absolute voltages as follows:

pwr_param = voltage on <pwrpin>

gnd_param = voltage on <gndpin>

Notes

This statement can only be used once in the SimCode. Only one pin can be defined for power and one for ground.

Example

```
PWR_GND_PINS(VCC, GND);
```

See Also

INPUTS, OUTPUTS, REALS, VIL_VIH_PERCENT, SUPPLY_MIN_MAX

READ_DATA

Reads data from an ASCII file into arrays.

General Form

```
READ_DATA(<array>[, <array>, ...])
```

Parameters

<array>	Name of the array into which the value is placed.

Use

The *READ_DATA* function opens the file specified by the "data=" parameter in the device's .MODEL statement and reads ASCII text data. The number and type (integer/real) of the values per line that will be read is based on the number and type of array variables that are specified in the function call. The number of data lines read is determined by the number of data lines in the specified file and/or the size of the smallest array in the function call. The *READ_DATA* function returns the number of lines read. A negative number is returned if an error is encountered:

-1	Invalid file name
-2	Can't find file
-3	Invalid array
-4	Illegal array access
-5	Data type
-6	Expected data value

Notes

Multiple values per line in the data file must be seperated by commas. The real values in the data file *must* be in scientific notation. The device's .MODEL statement which contains the "data=" parameter must be placed in the device symbol's .MOD file.

Example

MYDEVICE.MOD file:

```
.MODEL AMYDEVICE XSIMCODE(file="{MODEL_PATH}MYDEVICES.SCB"
+ func=MyDevice data="{MODEL_PATH}MYDEVICE.DAT" {mntymx})
```

MYDEVICE.DAT file:

```
8, 8E-6

9, 9E-6

10, 1E-5

11, 1.1E-5
```

MyDevice SimCode:

```
nlines  =  READ_DATA(int_array,  real_array);
```

This example opens a filed called MYDEVICE.DAT in the Models directory. It reads 2 columns of data from the file where the first column contains integer values and the second column contains real values. If the arrays are declared as int_array[3] and real_array[5] then only the first 3 data lines will be read and nlines will be set to 3.

REALS

Declares real variables and arrays.

General Form

```
REALS <var>[, <var>, ...];
```

Parameters

<var>	name of the variable

Use

The *REALS* data type is used to define real variables and arrays.

Notes

Real variables and arrays *must* begin with a letter and be defined before they are used. Real arrays are defined by following the array name with a left bracket ([ ), an integer number which defines the size of the array, and a right bracket ( ]). Real arrays can be set and/or used in expressions.

The following are reserved SimCode real variables which do not need to be declared:

Variable	Use	Digital Model Parameter
vil_param	low input state value	VIL value
vih_param	high input state value	VIH value
vol_param	low output state value	VOL value
voh_param	high output state value	VOH value
v3s_param	tri-state output state value	N/A
rol_param	low output strength value	N/A
roh_param	high output strength value	N/A
r3s_param	tri-state output strength value	N/A
pwr_param	voltage on power pin	PWR value
gnd_param	voltage on ground pin	GND value
present_time	present simulation time	N/A
previous_time	previous simulation time	N/A
sim_temp	circuit operating temperature	N/A (Spice Option: TEMP)

The Digital Model Parameter can be set independently for each digital device in the Digital Model Parameters dialog box. If the variable is set explicitly in the SimCode, that setting will override all other settings.

The values of *pwr_param* and *gnd_param* are set each time the *PWR_GND_PINS* statement is executed. The value of *present_time* and *previous_time* are set each time the time step changes. The value of *sim_temp* is the current operating temperature of the circuit which can be set from the Spice Option "TEMP".

Example

```
REALS tplh_val, tphl_val, ricc_val, vbias, values[64];
```

See Also

PWR_GND_PIN, VIL_VIH_VALUE, VIL_VIH_PERCENT, VOL_VOH_MIN

RECOVER

Tests inputs for recovery time violations.

General Form

```
RECOVER(<clk input> = {LH}|{HL} <mr input> [<mr input> ...]
   {TREC=<time>}|{TRECL=<time> TRECH=<time>} ["<message>"]);
```

Parameters

<clk input>	Name of or index to the input clock/ reference pin under test
<mr input>	Name of or index to the input set/reset pin under test.
TREC	Recovery time for both low and high going <mr pin>.
TRECL	Recovery time for low going <mr pin>.
TRECH	Recovery time for high going <mr pin>.
<message>	Text string that will be displayed if a warning occurs.

Use

The *RECOVER* function compares the time difference between a level change (LH or HL) on the <clk input> and a level change on the <mr input> to a specified test time. RECOVER test times are specified jointly using TREC=<time> (which sets TRECL and TRECH to the same value) or individually using TRECL=<time> and TRECH=<time>. If the compare time is less than the specified <time> a warning will be displayed during simulation. An optional <message> string can be included in the *RECOVER* statement which will be output if a warning is displayed.

Notes

Databook specifications should be used with this function. TRECL=<time> and TRECH=<time> can be entered in the same *RECOVER* test. The *RECOVER* test will be made only if the state of the <mr input> matches the time parameter (TRECL=LOW, TRECH=HIGH) when the <clk input> makes the specified transition (LH or HL). For example, if <clk input> = LH and TRECL is specified then the <mr input> must be LOW when the <clk input> goes from LOW to HIGH for a *RECOVER* test to be made. Pin names and variables *can* be mixed in the same *RECOVER* statement.

Example

```
RECOVER(CLK=LH  PRE  CLR  TREC=trec_val
   "CLK->PRE  or  CLR");
```

RETURN

Returns from a subroutine in the SimCode

General Form

```
RETURN;
```

Use

The *RETURN* instruction is used to return program flow to the instruction that followed the last *GOSUB* instruction.

See Also

GOSUB

SELECT_VALUE

Returns value from simple look-up table.

General Form

```
SELECT_VALUE  (<index>:  <val/pin/var>,
   <val/pin/var>[,<val/pin/var>,...]);
```

Parameters

<index>	input variable (index to <val/pin/var>)
<val/pin/var>	output value, pin or variable

Use

The *SELECT_VALUE* function returns the value of the number or variable indicated by the value of the index variable.

Notes

The number of values and/or variables used is not limited.

Example

```
A  =  (SELECT_VALUE(B:  16,  8,  4,  2,  1));
```

In this example, if B = 2 then A = 8 (the 2nd value).

See Also

PWL_TABLE, MIN_TYP_MAX

SETUP_HOLD

Tests inputs for setup and hold time violations

General Form

```
SETUP_HOLD(<clk input> = {LH}|{HL}
    <data input> [<data input> ...]
    {TS=<time>}|{TSL=<time> TSH=<time>}
    {TH=<time>}|{THL=<time> THH=<time>} ["<message>"];
```

Parameters

<clk input>	Name of or index to the input clock/ reference pin under test.
<data input>	Name of or index to the input data pin under test.
TS	Setup time for both low and high going <data input>.
TSL	Setup time for low going <data input>.
TSH	Setup time for high going <data input>.
TH	Hold time for both low and high going <data input>.
THL	Hold time for high going <data input>.
THH	Hold time for low going <data input>.
<message>	Text string that will be displayed if a warning occurs.

Use

The *SETUP_HOLD* function compares the time difference between a level change (LH or HL) on the <clk input> and a level change on the <data input> to a specified test time. SETUP test times are specified jointly using TS=<time> (which sets TSL and TSH to the same value) or individually using TSL=<time> and TSH=<time>. HOLD test times are specified jointly using TH=<time> (which sets THL and THH to the same value) or individually using THL=<time> and THH=<time>. If the compare time is less than the specified <time> a WARNING will be displayed. An optional <message> string can be included in a *SETUP_HOLD* statement which will be output if a WARNING is displayed.

Notes

Databook specifications should be used with this function. TSL=<time>, TSH=<time>, THL=<time> and THH=<time> can be entered in the same *SETUP_HOLD* statement. The SETUP and/or HOLD test will be made only if the state of the <data input> matches the time parameter (TSL or THL=LOW, TSH or THH=HIGH) when the <clk input> makes the specified transition (LH or HL). For example, if <clk input>=LH and TSL is specified, then the <data input> must be LOW when the <clk input> goes from LOW to HIGH for a SETUP test to be made. Pin names and variables *can* can be mixed in the same *SETUP_HOLD* statement.

Example

```
SETUP_HOLD(CLK=LH  DATA  Ts=ts_val  Th=th_val
   "CLK->DATA");
```

STATE

Sets outputs to the declared logic state.

General Form 1

```
STATE  <output>  [<output>...]  =  (<expression>);
```

General Form 2

```
STATE  <output>  [<output>...]  =
   {ZERO}|{ONE}|{UNKNOWN};
```

Parameters

<output>	Name of or variable index to the output pin.
<expression>	Any expression to be compared to VIL or VIH.

Use

The state of an output pin is determined by its level and its strength. The *STATE* command sets the level and strength for one or more output pins or variables. If <expression> is less than or equal to *vil_param*, the output will be set to ZERO. If <expression> is greater than or equal to *vih_param*, the output will be set to ONE. Otherwise, the output will be set to UNKNOWN. The level and strength values are set according the the state:

<expression>	State	Level	Strength
<= *vol_param*	ZERO	*vol_param*	*rol_param*
>= *voh_param*	ONE	*voh_param*	*roh_param*
other	UNKNOWN	*v3s_param*	*r3s_param*

Notes

Output pins can be specified by using the output pin name or by an integer variable that contains the *index* of an output pin. Pin and variable names *cannot* be mixed in the same *STATE* command. References to outputs must be either all pin names or all variable names.

Examples

```
STATE Q = ONE;
STATE Q1 Q2 Q3 Q4 = ZERO;
STATE OUT = ((1+2)/3);
```

In the last example, OUT will be:

ZERO if *vil_param* > 1

UNKNOWN if *vil_param* < 1 and *vih_param* > 1

ONE if *vih_param* < 1

See Also

REALS, STATE_BIT, LEVEL, STRENGTH, TABLE, EXT_TABLE

STATE_BIT

Sets outputs to binary weighted logic states.

General Form

```
STATE_BIT <output> [<output> ...] = (<expression>) ;
```

Parameters

<output>	name of or variable index to the output pin
<expression>	any expression which can be bitwise matched with the listed outputs

Use

The state of an output pin is determined by its level and its strength. The *STATE_BIT* command is used to set the level and strength for one or more output pins based on the value of the <expression>. The state of the first pin listed is set according to the first (least-significant-bit) of the expression's value, the state of the second pin listed is set according to second bit of the expression's value, and so on. The level and strength values are set by the bit's value:

Bit Value	**State**	**Level**	**Strength**
0	ZERO	*vol_param*	*rol_param*
1	ONE	*voh_param*	*roh_param*

Notes

Output pins can be specified by using the output pin name or by an integer variable that contains the *index* of an output pin. Pin and variable names *cannot* be mixed in the same *STATE_BIT* statement. References to outputs must be either all pin names or all variable names. The maximum number of output pins/vars is limited to 16.

Example

```
STATE_BIT  Q1  Q2  Q3  Q4  =  (internal_reg);
```

In this example, if internal_reg = 11 (1011 binary) then Q1 (LSB) = ONE, Q2 = ONE, Q3 = ZERO and Q4 (MSB) = ONE.

See Also

REALS, STATE, LEVEL, STRENGTH, TABLE, EXT_TABLE

STEP_OFF

Turns off the SimCode trace mode.

General Form

```
STEP_OFF
```

Use

The *STEP_OFF* statement turns off the SimCode TRACE mode.

See Also

STEP_ON

STEP_ON

Turns on the SimCode trace mode.

General Form

```
STEP_ON
```

Use

The *STEP_ON* statement turns on the SimCode TRACE mode. This causes the SimCode to display the Program Counter (PC) number and each SimCode instruction before it is executed.

See Also

STEP_OFF

STRENGTH

Sets the strength of the output state.

General Form 1

```
STRENGTH <output> [<output> ...] = (<expression>);
```

General Form 2

```
STRENGTH <output> [<output> ...] = {STRONG}|{HI_IMPEDANCE};
```

Parameters

<output>	Name of or variable index to the output pin.
<expression>	Any expression to be used directly as a strength.

Use

The state of an output pin is determined by its level and its strength. Use the *STRENGTH* command to set the strength of one or more output pins.

Value	**State**	**Strength**
STRONG	ZERO	*rol_param*
STRONG	ONE	*roh_param*
HI_IMPEDANCE	N/A	*r3s_param*
<expression>	N/A	<expression>

Notes

Output pins can be specified by using the output pin name or by an integer variable the contains the *index* of an output pin. Pin and variable names *cannot* be mixed in the same *STATE* statement. References to outputs must be either all pin names or all variable names.

See Also

REALS, STATE, STATE_BIT, LEVEL

SUPPLY_MIN_MAX

Tests supply pins for min and max supply voltage violations.

General Form

```
SUPPLY_MIN_MAX(<min value>, <max value>);
```

Parameters

<min value>	Minimum recommended power supply voltage.
<max value>	Maximum recommended power supply voltage.

Use

The *SUPPLY_MIN_MAX* function checks the voltage difference between the power and ground pins defined in *PWR_GND_PINS*. If the "WARN flag" is set in the Digital Model Parameters dialog box and the voltage difference (*pwr_param* - *gnd_param*) is less than <min value> or greater than <max value> a warning will be displayed during simulation.

Notes

Databook specifications should be used with this function. *PWR_GND_PINS* must be defined to use this function.

Example

```
SUPPLY_MIN_MAX(4.75, 5.25);
```

See Also

INTEGERS, PWR_GND_PINS

TABLE

Sets output logic states based on truth table.

General Form

```
TABLE <line>
<input> [<input> ...] <output pin> [<output pin> ...]
<input state> [<input state> ...] <output state> [<output state> ...];
```

Parameters

<line>	Variable into which the line number used in the table is placed.
<input>	Name of the input pin or variable index to the input pin.
<output pin>	Name of the output pin.
<input state>	State of the individual inputs.
<output state>	State of the individual outputs based on input conditions.

Use

The *TABLE* statement operates like a truth table to set the level and strength of the specified outputs. Valid input states are:

0	low (input voltage is <= *vil_param*).
1	high (input voltage is >= *vih_param*).
X	don't care what input voltage is.

Valid output states are:

L	ZERO (set output level to *vol_param*)
H	ONE (set output level to *voh_param*).
Z	UNKNOWN (set output level to *v3s_param*).

Output state letters can be followed by a colon and a letter to indicate strength:

s	STRONG (set output to *rol_param* for L and *roh_param* for H).
z	HI_IMPEDANCE (set output to *r3s_param*).

If a strength character is *not* specified after an output state then STRONG will be used for L and H states and HI_IMPEDANCE will be used for Z states.

Notes

Each row is tested sequencially from top to bottom until the input conditions are met. The outputs are set for the *first* row to meet the input conditions. The <line> is set to the line number in the table that was used. If no match was made then <line> is set to 0. Input pin and variable names *cannot* be mixed in the same *TABLE* statement. References to inputs must be either all pin names or all variable names.

Example

```
TABLE  tblIndex
INA    INB    OUT
0      0      H
0      1      H
1      0      H
1      1      L;
```

This example is representative of 1/4 of a 7400 2-input NAND gate. If input pins INA and INB are both high (>= *vih_param*), OUT is set to ZERO (*vol_param*) and STRONG (*rol_param*) and tblIndex is set to 4.

See Also

REALS, STATE, STATE_BIT, EXT_TABLE

VALUE

Returns the value of the specified pin.

General Form

```
VALUE(<pin>)
```

Parameters

<pin>	Name of or index to a pin.

Use

The *VALUE* function returns a real number that indicates the voltage level of the specified pin.

Example

```
v  =  (VALUE(D3));
```

VIL_VIH_PERCENT

Sets VIL and VIH values to a percentage of supply voltage.

General Form

```
VIL_VIH_PERCENT (<vil %>, <vih %>);
```

Parameters

<vil %>	Percentage of the suppy voltage which defines vil.
<vih %>	Percentage of the suppy voltage which defines vih.

Use

VIL and VIH do not use a min/typ/max array to select their values, but must be declared explicitly for each digital device. The *VIL_VIH_PERCENT* statement sets the VIL and VIH parameters of the device to a percentage of the supply voltage as follows:

vil_param = (*pwr_param* - *gnd_param*) * <vil %>

vih_param = (*pwr_param* - *gnd_param*) * <vih %>

Notes

PWR_GND_PINS must be defined to use this function. The % values must be greater than 0 and less than 100. The *vil_param* and *vih_param* values set by *VIL_VIH_PERCENT* are overridden by any values set for “VIL value” and “VIH value” in the Digital Model Parameters dialog box.

Example

```
VIL_VIH_PERCENT(33, 67);
```

See Also

REALS, PWR_GND_PINS, VIL_VIH_VALUE

VIL_VIH_VALUE

Sets absolute VIL and VIH values.

General Form

```
VIL_VIH_VALUE (<vil>, <vih>);
```

Parameters

<vil>	Absolute voltage level which defines vil.
<vih>	Absolute voltage level which defines vih.

Use

VIL and VIH do not use a min/typ/max array to select their values, but must be declared explicitly for each digital device. The *VIL_VIH_VALUE* statement sets the VIL and VIH parameters of the device to absolute voltages as follows:

vil_param =<vil>

vih_param =<vih>

Notes

In order to more accurately model the actual switching characteristics of a digital input, VIL and VIH are *not* generally set to their specified databook values. The exception is the case of devices with a specified hysteresis such as the 74LS14. Typically, the hysteresis of a digital device is small, in the order of 100mV, but never 0V.

The *vil_param* and *vih_param* values set by *VIL_VIH_VALUE* are overridden by any values set for “VIL value” and “VIH value” in the Digital Model Parameters dialog box.

Example

```
VIL_VIH_VALUE(1.25, 1.35);
```

See Also

REALS, VIL_VIH_PERCENT

VOL_VOH_MIN

Sets VOH and VOL relative to power and ground.

General Form

```
VOL_VOH_MIN (<vol offset>, <voh offset>, <min voh-vol>);
```

Parameters

<vol offset>	Voltage offset which must be applied to ground pin voltage to get vol.
<voh offset>	Voltage offset which must be applied to power pin voltage to get voh.
<min voh-vol>	Minimum allowed difference between voh and vol.

Use

VOL and VOH do not use a min/typ/max array to select their values, but must be declared explicitly for each digital device. The *VOL_VOH_MIN* statement sets the VOL and VOH parameters of the device as follows:

vol_param = *gnd_param* + <vol offset>

voh_param = *pwr_param* + <voh offset>

Notes

In order to more accurately model the actual characteristics of a digital output, VOH is *not* generally set to its specified databook value. The reason for this deviation is that databook values for VOH are specified for maximum IOH load. In digital SimCode, VOL and VOH represent an *unloaded* output voltage.

PWR_GND_PINS must be defined to use this function. The *vol_param* and *voh_param* values set by *VOL_VOH_MIN* are overridden by any values set for “VOL value” and “VOH value” in the Digital Model Parameters dialog box. These are offset values rather than absolute voltages. The <voh offset> is negative so that when added to *pwr_param*, the resulting VOH will not be greater than *pwr_param*. If the difference between the resulting *vol_param* and *voh_param* is less than <min voh-vol>, then *vol_param* will be set to the value of *gnd_param* and *voh_param* will be set to *gnd_param* + <min voh-vol>.

Example

```
VOL_VOH_MIN(0.2, -0.4, 0.1);
```

In this example:

1 If *gnd_param* = 0V and *pwr_param* = 5.0V, then

vol_param = 0.2V and *voh_param* = 4.6V

2 If *gnd_param* = 0V and *pwr_param* = 0.5V, then

vol_param = 0.0V and *voh_param* = 0.1V

See Also

REALS, PWR_GND_PINS

WHILE ... DO

Conditionally controls looping in the SimCode.

General Form

```
WHILE (<expression>) DO BEGIN ... END;
```

Parameters

<expression>   Any expression that can be evaluated as true or false

Use

The *WHILE ... DO* statement is used to loop through a section of SimCode until <expression> evaluates to false.

Notes

Program flow will remain in a loop between the BEGIN and END statements until <expression> evaluates to false, then program flow resumes after the END statement.

Examples

```
i = 1;
WHILE (i <= 5) DO
   BEGIN
      data[i] = data[i + 1];
     i = i + 1;
   END;
```

See Also

IF ... THEN

WIDTH

Tests inputs for minimum pulse width violations.

General Form

```
WIDTH(<input>  [<input>...]
   {TWL=<time>}|{TWH=<time>}  ["<message>"];
```

Parameters

<input>	Name of or variable index to the input pin under test.
TWL	Width of a low going pulse.
TWH	Width of a high going pulse.
<message>	Text string that will be displayed if a warning occurs.

Use

The *WIDTH* function compares the pulse width on each <ipin> to the specified test WIDTH times. A low level test time is specified using TWL=<time> while a high level test time is specified using TWH=<time>. If the compare time is less than the specified <time> a WARNING will be displayed. An optional <message> string can be included in the *WIDTH* statement which will be output if a WARNING is displayed.

Notes

Databook specifications should be used with this function. The input pins can be input pin names and/or integer variables that contain an index value to an input pin. Pin names and variables *can* be mixed in the same *WIDTH* statement.

Examples

```
WIDTH(CLK  TWL=clk_twl  TWH=clk_twh  "CLK");
WIDTH(PRE  CLR  TWL=  pre_clr_twl  "PRE  or  CLR");
```

WIDTH_TIME

Returns last pulse width encountered on specified pin.

General Form

```
WIDTH_TIME(<input>)
```

Parameters

<input>	Name of or index to an input pin.

Use

This function returns a real value that indicates the last pulse width encountered on the specified <input>.

Example

```
PW  =  (WIDTH_TIME(CP2));
```

Index