BoardSim User's Guide

HyperLynx

December, 1999

HyperLynx has made every effort to ensure that the information in this document is accurate and complete. HyperLynx assumes no liability for errors, or for any incidental, consequential, indirect, or special damages, including, without limitation, loss of use, loss or alteration of data, delays, or lost profits or savings, arising from the use of this document or the product which it accompanies.

HyperLynx
14715 NE 95th Street, Suite 200
Redmond, WA 98052
support@hyperlynx.com

Table of Contents

Chapter 1: Installation and Licensing

Summary

This chapter describes:

- what kind of PC you need to run BoardSim
- how BoardSim's licensing works (node-locked and floating)
- how to install BoardSim
- how to install the HyperLynx floating-license server application *(for floating-license customers only)*
- additional information about running with a floating license
- how to install BoardSim on a remote network computer
- how to view manuals online

System Requirements

Operating Systems

BoardSim runs under Windows 95, Windows 98, and Windows NT 4.0 or greater. It does not run under Windows 3.1/3.11 or Windows NT 3.51.

Special Requirement for Installing under Windows NT

IMPORTANT! When you install under Windows NT, you must be logged in as an NT "administrator" (or "supervisor") in order for the hardware-protection key driver to install properly. If you receive an error message when the hardware key's driver attempts to install, it is most likely because you are not logged on with administrator privileges (only administrators are allowed to install drivers).

If this occurs, BoardSim's installation program allows you to finish installing the HyperLynx software, then gives you an easy way of installing just the key driver later, after you've logged in as an administrator. See "If the Hardware-Key Driver Fails to Install" below in this chapter for details.

Video

BoardSim works with SVGA-or-greater video resolution (i.e., 800x600 or higher). At VGA resolution (640x480), some of BoardSim's dialog boxes are too large to fit on the screen. You may still be able to run the program in VGA resolution by moving certain dialog boxes side-to-side as needed.

Memory

BoardSim runs on any PC with at least 16 Megabytes of RAM (real, physical memory). However, for analyzing large boards, more memory may be required. Generally, BoardSim requires at least as much memory to load a board as does the PCB-layout tool that created it, since BoardSim loads the same physical routing information as does the layout tool, plus creates electrical information not present in the PCB database.

Note: *It is possible that BoardSim may not run on a 16-Megabyte machine if most of the memory is used by other applications. If you get "out of memory" errors from BoardSim, try closing other open applications, or freeing more memory for Windows to use.*
Another sign that BoardSim is low on memory is excessive use of the hard disk. If the hard disk is accessed whenever you perform an operation in BoardSim, then Windows is running BoardSim out of virtual memory (i.e., off the hard disk rather than out of physical memory), which slows BoardSim considerably. Again, free more memory for BoardSim to use.

Hard-Disk Space

BoardSim itself requires about 40 Megabytes of hard-disk space for installation. Most of this space is used by IC models.

Additional space is required for translated board files (.HYP files). For large boards, .HYP files can be 5-10 Megabytes each, sometimes bigger.

Mouse

BoardSim requires a mouse. Portions of its user interface cannot be accessed from the keyboard. Any mouse that runs in Windows works with BoardSim.

Parallel Port for Hardware Key

BoardSim requires a hardware key to run. If you have a node-locked version of the software, the hardware key plugs into a parallel port on your computer. If you have floating-license version, the key plugs into the computer you use as a HyperLynx license server. (For details on node-locked versus floating licenses, see "About Licensing" below.)

If you have multiple parallel ports, you can plug the hardware key into any of the ports, and BoardSim will find it.

Note: *In rare cases under Windows NT, having multiple ports will slow the querying of the key. In these instances, you can tell the key driver to skip ports — see "If Running Windows NT with Multiple Ports and the Key Polls Slowly" below in this chapter.*

BoardSim uses the same key that all of HyperLynx's products use. If you purchase additional HyperLynx products (node-locked or floating), you will be given one or more license codes that enable your key for the additional products.

BoardSim's hardware key usually works fine in series with a printer, scanner, etc., although you may need to power up the device to get proper key operation. In rare cases, the key will not work with a peripheral device attached, and you will have to move the device to a separate parallel port.

Note: *Generally, the closer BoardSim's key is to the computer, the more reliably it operates. If you have multiple hardware keys on a parallel port and BoardSim cannot find its key, try changing the order of the keys and moving BoardSim's closer to the PC.*

About Licensing

How HyperLynx's Licensing Works

BoardSim uses a programmable hardware key which allows HyperLynx to license you on a per-product basis. You can use the key on a single computer ("node-locked"), or on a designated "server" computer that grants licenses (when available) over the network to client computers ("floating license").

When You First Receive BoardSim

In most cases, when you first receive BoardSim, it is *pre-licensed* to run the options you have purchased. In that case, you do not need to enter any license codes: just plug the hardware key into a parallel port of the appropriate computer (your computer or a designated server computer, depending on which type of licensing you purchased), and run.

However, sometimes the product cannot be pre-licensed. If you purchased a HyperLynx product and receive an error message saying you are not licensed to run it, contact HyperLynx or your local reseller to obtain a license code. Usually, the code will be sent by fax or e-mail.

Note: *There is a way of checking without running BoardSim which features you are licensed for; see "Reviewing What Features You are Licensed For" below in this chapter for details.*

Node-Locked versus Floating Licensing

HyperLynx supports two types of licensing:

- **Node-locked licensing -** Restricts the HyperLynx applications to running on a single computer (whichever computer has the hardware key attached)
- **Floating (or "networked") licensing -** Allows HyperLynx applications to run from any computer on the network, as long as the applications are not being run on too many other computers; the key remains attached to a designated "server" computer

The primary advantage of floating licensing compared to node-locked is convenience. If you have several users who, at different times, want to run a HyperLynx application, floating licensing allows them to do so without having to transport the hardware key from computer to computer. It is also very easy to add floating licenses if your user demand increases: additional licenses require no extra hardware keys.

Note: *If you have a node-locked copy of a HyperLynx application and want to convert it to a floating license, contact HyperLynx or your local representative.*

Client versus Server Computers (Floating Licenses Only)

In a floating-license scenario, the computer on the network which has the hardware key attached to it and which grants licenses to other computers is called the "server." The other computers on the network which request to run HyperLynx applications are called "clients."

You can have more than one HyperLynx license server on your network. You might want to have several servers, for example, in case one server computer crashes or is removed from the network. (If you order an additional floating license from HyperLynx or your local representative, you're given the choice of activating it on your existing hardware key or on a new key destined for a new server.)

A server can grant licenses to itself, i.e., you can run HyperLynx applications even on a server computer. This might be convenient in a small engineering group where one of the engineers who runs HyperLynx applications also wants his computer to function as the server.

HyperLynx license servers can run on any Windows 95, Windows 98, or Windows NT (version 4.0 or later) computer on the network. To be a HyperLynx server, the designated computer does not have to be a "server" in any other sense; "ordinary" PCs can act as HyperLynx servers.

Installing BoardSim (Node-Locked or Floating Client)

This section describes how to install BoardSim (and other HyperLynx application programs) for a node-locked computer or floating-license *client* computer.

Note: *A separate section below describes how to install the HyperLynx License Server software, for a floating-license* ***server*** *computer. See "Installing the Floating-License Server."*

The HyperLynx CD-ROM contains a file called "INSTALL.TXT" in the root directory of the CD; this file contains the most-recent installation instructions. The file "README.TXT" contains information about new features in the version of software you're installing.

What You're Installing

HyperLynx Applications

A complete BoardSim installation consists of one or more of the following components, depending on which PCB-layout tool you are using and what kind of licensing (node-locked or floating) you purchased:

- "BoardSim/LineSim / analysis options" — Installs the BoardSim base product (plus LineSim, the EMC option, and the Crosstalk option, too, if you purchased them)
- PCB-layout translator — Installs a translator for converting your PCB-layout tool's data into BoardSim's .HYP format; not included if your PCB-layout tool has a built-in

	HyperLynx translator
♦ HyperLynx License Server	Installs HyperLynx's license server, which serves floating licenses to "client" computers; only needed if you purchased a floating license

Depending on which PCB-layout tool you are connecting to, you may not need to install a translator. For example, Protel customers use a translator that is included in the Protel layout software, and not shipped by HyperLynx. Cadence Allegro users, on the other hand, receive a translator from HyperLynx.

You need to install the HyperLynx License Server application only if you purchase one or more floating licenses (see "Installing the HyperLynx License Server" and "Licensing a Floating-License Client Computer" below for more details).

Hardware-Key Driver (Windows NT Only)

Windows 95 and Windows 98

Under Windows 95 and Windows 98, no driver is required for the hardware key.

Windows NT V4.0 or Greater

The hardware-key driver is required under Windows NT V4.0 and greater. Normally, installation and start-up of the driver is completely automatic; it happens transparently as part of the HyperLynx-application installation.

However, if you are not logged on as an "administrator," the driver cannot install. In this case, the remainder of the software will finish installing, then you can easily return (after logging in with administrator privileges) to install only the driver — see "If the Hardware-Key Driver Fails to Install" below for details.

Steps for Installing the Software

Windows 95, Windows 98, and Windows NT 4.0 have an "AutoPlay" feature that will automatically start applications when a CD-ROM is inserted into a drive. Most Windows computers have this feature enabled.

However, it is possible that AutoPlay is not enabled on your system for security reasons. AutoPlay will also be disabled if you hold down the shift key when the CD-ROM is inserted.

If Windows AutoPlay is enabled on your computer: ***(normal installation)***

1. **Insert the HyperLynx CD-ROM into your CD-ROM drive.**

2. After a short period, the HyperLynx Install program will run automatically and open the HyperLynx Install dialog box.

3. **Click the Install HyperLynx Software button.** This will install BoardSim (and any other HyperLynx applications you have purchased). Follow the on-screen instructions until this portion of the installation is complete.

4. **Click the Install PCB Translators button.** The Install HyperLynx Translators dialog box opens, showing you a list of all the PCB-layout translators available from HyperLynx.

5. **Click the button for the appropriate translator.**
   If the translator is built-in to your PCB-layout tool, you do not need to re-install it. The installation program will notify you of this; click Exit and proceed with step 6 below.
   If the translator is not built-in to your PCB-layout tool, the installation program will request a password. **The password is recorded on your CD-ROM's packaging.** Follow the on-screen instructions until this portion of the installation is complete.

6. **If you want to view HyperLynx's manuals online** and you do not already have a copy of Adobe's Acrobat viewer (version 3.0 or greater) installed on your computer, **click the Install Adobe Acrobat button.** Follow the on-screen instructions until this portion of the installation is complete.

7. Optionally, click the View Readme and View License Agreement buttons to display this information.

8. Installation is now complete. Click Exit.

If Windows AutoPlay is disabled on your computer:

1. **Insert the HyperLynx CD-ROM into your CD-ROM drive.**

2. **Using the Windows Explorer, double-click the Setup application (SETUP.EXE) on the CD-ROM** and the HyperLynx Install program will run automatically and open the HyperLynx Install dialog box.

3. Continue with step 3 in the "If Windows AutoPlay is enabled" section just above.

If the Hardware-Key Driver Fails to Install (Windows NT Only)

Under Windows NT, only users with "administrator" privileges can install device drivers. Accordingly, if you install the HyperLynx software but are not logged on as an administrator, the portion of the installation that installs the hardware-key driver will fail ("You need System Administrator privileges…"). However, you can easily finish installing the other portions of the software (PCB-layout translators, Acrobat reader, etc.), then re-install just the key driver after you've logged back on with the required privileges.

To install just the hardware-key driver after a failed driver installation:

1. Log on to Windows NT with administrator privileges. (See your network administrator if you don't know how to log on as an administrator.)

2. From the HyperLynx CD-ROM, run the HyperLynx installation program (using AutoPlay or by running SETUP.EXE). The HyperLynx Install dialog box opens.

3. Click the Configure HyperLynx License button. The HyperLynx Licensing dialog box opens.

4. Click the Install Sentinel Device Driver button. This runs just the hardware-key portion of the installation.

If you do not have the HyperLynx CD-ROM immediately accessible, you can run the key-driver installation program directly. From the Windows Explorer, look in the SENTINEL subdirectory under your HyperLynx installation, and double-click on the file NT_KEY.EXE. This executes the key installation directly.

Testing the BoardSim Installation

General Test

To test the installation:

1. *If you are running a node-locked copy of BoardSim,* make sure the hardware key is plugged into a parallel port on your PC.

   If you are running a floating-license copy, make sure the HyperLynx License Server and hardware key are installed on a PC, and that both your PC and the server PC are running on the network. Also, be sure that the HyperLynx License Server application is running on the server PC.

2. Choose the HyperLynx Simulation Software icon from the HyperLynx group on the Start menu (Start/Programs/HyperLynx).

3. BoardSim should open, ready to load a .HYP file.

If a Node-Locked Hardware Key is Not Working *(Node-Locked Only)*

For a node-locked copy of BoardSim, the following are symptoms of BoardSim's hardware key not working correctly:

- a "hardware key" or "copy protection" error message
- the program opens, but most of its menus are not visible, and there is no way to load a board file (".HYP" file)

If either of these occur, run the steps in "Testing the Hardware-Protection Key" below.

If the Program's Menus and Buttons are Missing Features, and You Can't Load a Board *(Node-Locked or Floating Client)*

If you open BoardSim, but get only a limited set of menus and toolbar buttons (e.g., only the Notepad and Help buttons on the toolbar), BoardSim is unable to find valid licensing. Specifically, if your copy is node-locked, then this symptom means that the hardware key is unplugged or not functioning correctly. If your copy is floating, this symptom means that BoardSim cannot find any HyperLynx license servers.

Testing the Hardware-Protection Key *(Node-Locked or Floating Server)*

If BoardSim does not run properly and you suspect that the hardware-protection key may be the cause, you can test the key and its driver.

To test the hardware key:

1. From the HyperLynx CD-ROM, run the HyperLynx installation program (using AutoPlay or by running SETUP.EXE). The HyperLynx Install dialog box opens.

2. Click the Configure HyperLynx License button. The HyperLynx Licensing dialog box opens.

3. Note the status message in the Hardware Protection Key area:
   If the key is functioning properly, the message says "A hardware protection key was found."
   If the key is not functioning, the message says "No hardware protection key was found."

4. To run diagnostics on the key, click the Test Hardware Key button. A program supplied by the key vendor runs; click the Find Keys button to run the vendor tests.

If you do not have the HyperLynx CD-ROM immediately accessible, you can run the hardware-key test program directly. From the Windows Explorer, look

in the UTIL subdirectory under your HyperLynx installation, and double-click on the file FIND32.EXE. (This is the vendor-supplied test program.)

If Running Windows NT with Multiple Parallel Ports and the Key Polls Slowly *(Windows NT Only)*

If you are running under Windows NT and your computer has multiple parallel ports, it is possible that the hardware-protection key driver will run slowly, as it polls from port to port looking for the key. If you experience "sluggish" operation, you can tell the driver not to waste time looking at unused ports.

To instruct the hardware key driver not to poll unused ports:

1. From the Windows Explorer, look in the SENTINEL\WIN_NT subdirectory under your HyperLynx installation, and double-click on the file SETUPX86.EXE. The Rainbow Technologies Sentinel dialog box opens (this program is provided by the key vendor).

2. From the Functions menu, choose Configure Sentinel Driver. A dialog box opens.

3. Click the Help button to read about how to disable polling of unused ports.

Reviewing What Features You are Licensed For

If your installation is working, but certain features are not available (for example, an analysis option or a PCB-layout translator you thought you were licensed for), you can check for which features you are currently licensed.

To check what features you are licensed for:

1. From the HyperLynx CD-ROM, run the HyperLynx installation program (using AutoPlay or by running SETUP.EXE). The HyperLynx Install dialog box opens.

2. Click the Configure HyperLynx License button. The HyperLynx Licensing dialog box opens.

3. Click the Show Licenses button. The Licensing dialog box opens; it lists all of the features for which you are currently licensed, and their status.

If you do not have the HyperLynx CD-ROM immediately accessible, you can run the license-checking program directly. From the Windows Explorer, look in the UTIL subdirectory under your HyperLynx installation, and double-click on the file SPVIEW.EXE.

If You Receive a "DOS Error" Message

If you attempt to run BoardSim under Windows NT and receive the error message "unexpected DOS error: 11" (or a similar message), your version of Windows NT is out-of-date. BoardSim requires Windows NT V4.0 or later.

Laptops Must Have Parallel Port Powered Up

If you are running BoardSim on a laptop computer, be sure that the parallel port is powered up. Some laptops will default to shutting down the parallel port in order to reduce power consumption. For details on the symptoms that may occur if the port is not powered, see the sections above.

Installing the Floating-License Server

This section describes how to install the HyperLynx Floating-License Server program. This installation is required only if you purchased a floating license, and happens only on the machine which you want to act as the server of HyperLynx licenses. On client computers (the ones that will actually run the software), you only need to install the HyperLynx application programs (not the license server); see "Installing BoardSim" above for details.

Steps for Installing the License Server

Windows 95, Windows 98, and Windows NT 4.0 have an "AutoPlay" feature that will automatically start applications when a CD-ROM is inserted into a CD-ROM drive. Most Windows computers have this feature enabled.

However, it is possible that AutoPlay is not enabled on your system for security reasons. AutoPlay will also be disabled if you hold down the shift key when the CD-ROM is inserted.

If Windows AutoPlay is enabled on your computer: ***(normal installation)***

1. **Insert the HyperLynx CD-ROM into your CD-ROM drive.**
2. After a short period, the HyperLynx Install program will run automatically and open the HyperLynx Install dialog box.
3. **Click the Configure HyperLynx License button.** The HyperLynx Licensing dialog box opens.
4. **In the Floating License area, click the Install Server button.** This will install the license server, including installation (for Windows NT; not required for Windows 95 or 98) of the hardware-protection key driver. Follow the on-screen instructions until the installation is complete.

If Windows AutoPlay is disabled on your computer:

1. **Insert the HyperLynx CD-ROM into your CD-ROM drive.**
2. **Using the Windows Explorer, double-click the Setup application (SETUP.EXE) on the CD-ROM** and the HyperLynx Install program will run automatically and open the HyperLynx Install dialog box.
3. Continue with step 3 in the “If Windows AutoPlay is enabled” section just above.

Testing the License Server Installation

General Test

To test the installation:

1. Make sure the hardware key is plugged into a parallel port on the server PC.
2. Choose the HyperLynx License Server icon from the HyperLynx License group on the Start menu (Start/Programs/HyperLynx License).

3. The License Server should open in a dialog box.

Testing the Hardware-Protection Key

If the license server fails to run and the error appears to be related to the hardware-protection key, read section “Testing the Hardware-Protection Key” (and following) above to test the key.

Testing the Server with Client Computers

To test that the server (once it is running on the server PC) can serve licenses to client computers, see section “Licensing a Floating-License Client Computer” below.

Licensing a Node-Locked Computer

When you first receive your HyperLynx software, your licensing is pre-configured; you should be able to install the software and begin using it without needing to enter any licensing code. Use the license editor only when you need to enter a new license code you have been faxed from HyperLynx, or to check the status of your current licensing.

On a node-locked computer, the HyperLynx hardware key must be attached to a parallel port. Licensing is for the local computer only; other computers on the network cannot run HyperLynx applications. (If you are interested in a “floating” license which can be shared by multiple computers on the network, contact HyperLynx or your local representative.)

Using the License Editor

To open the licensing editor:

1. From the Options menu, choose Licensing.

The next sections describe information displayed in the licensing editor.

User Name

The first time you run BoardSim after entering a new license code, the program queries you for the name of the licensed user. The user name is "burned" into the hardware key; from then on, it "travels" with the key. **This is a one-time-only operation, so take care to enter the user name correctly the first time.**

Note: *The user name appears on all print-outs from BoardSim.*

Products and License Status

A list box in the licensing editor shows the current status for all HyperLynx products. Status information includes:

- product name
- kind of license
- expiration date of the option's license

There are two kinds of licenses:

- **full license:** when the expiration date is passed, the software keeps running; HyperLynx or your local representative will contact you about purchasing another year's worth of product maintenance
- **trial license:** when the expiration date is passed, the software stops running

Key Serial Number

The licensing editor shows the serial number of your hardware key. If you ever need to know your key number (for example, when you contact HyperLynx for technical support), looking in the license editor is easier than looking physically on the key.

Entering a New License Code

HyperLynx license codes are 12-digit hexadecimal numbers. The codes are case insensitive.

To enter a new license code:

1. In the Licensing dialog box, type the code number into the License Code edit box.

The box only accepts valid, HyperLynx-generated license codes; if you type a random code, BoardSim rejects it with an error message. Codes are hardware-key-specific, i.e., if you try to license a key with a code generated for a different key, the program will reject the code and report an error.

Note: *Once a license code has been successfully entered, it is not displayed in the dialog box. The license resides in the hardware key; the code is no longer needed.*

You can re-enter a license code for a key that has already been programmed with the same license; the key will simply be re-programmed.

Licensing a Floating-License Client Computer

With floating licensing, your computer acts as a "client" that requests permission to run HyperLynx applications from a "server" computer on the network. The HyperLynx hardware key is attached to the server computer. If a license is available for the HyperLynx application you request, the server grants you one and you're able to run; if no licenses are available (because other computers on the network are using them), you have to wait until a license becomes available (when another user is finished using it).

The licensing is administered by the server computer; as a client, you simply try to access a license. In order to do this, your computer must be able to "see" one or more HyperLynx server computers on the network.

Connecting Your Client Computer to HyperLynx Server(s)

To make setting up floating licenses from a client computer as simple as possible, HyperLynx applications are designed to automatically detect and connect to HyperLynx servers on the network. Therefore, you may never need to use the License Servers dialog box. The dialog box gives you a way of seeing which servers are available, and of testing your connections to them, if you ever need to.

Each time a HyperLynx application executes on a client computer, it polls the network for one or more HyperLynx license servers. If a HyperLynx server with an available license is found on the network, the application will continue to run. If no server or no available license is found, the application will run but without any useful features.

The HyperLynx License Servers dialog box lets you configure which computers on your network will be queried when you try to run HyperLynx products from your client machine. The License Servers dialog box shows which HyperLynx Servers are available on the network, and lets you test your connections to those servers.

To open the HyperLynx License Servers dialog box:

1. From the Options menu, choose Licensing.

The next sections describe information displayed in the dialog box.

Note: *If Options/Licensing opens the "Licensing" dialog box instead of the HyperLynx License Servers dialog box, then BoardSim has found a node-locked key on your local PC. The HyperLynx License Servers box opens only if no key is present on your PC.*

HyperLynx License Servers List

The HyperLynx License Servers area lists up to four HyperLynx Servers that BoardSim has found on the network. If no servers are listed, then none were successfully found. Often, there will only be one server; only if you purchase multiple floating licenses and request more than one key will multiple servers exist.

The server names are filled in automatically by BoardSim. Normally, you do not need to type in the names, although you might do so to test your connection to a server that is not being found automatically. Server computers will be queried for licenses in the order in which they are listed. If Server1 cannot be contacted or does not have a license available for checkout, then Server2 will be queried; if Server2 fails, then Server3 is queried; and so forth.

The Lock Server Name check boxes to the right of each server name are normally unchecked, so that automatic detection of servers occurs. If you have trouble with automatic detection or want to by-pass the short time delay associated with it, you can enable these check boxes; see "Manually 'Locking' a Server Connection" below for details.

Testing the Connection to a Server

To test the connection to a server:

1. Find the server name in the list of servers. If you are attempting to test the connection to a server that is not listed, type the server's name in the first available data box (see below for a description of valid server names).

2. Click the Test button next to the server's name. If the server is found, a dialog box will open with a list of the licenses available from the server. If the server cannot be found, then after a brief pause, a "could not access" error will appear.

A successful test will re-establish the connection to a server whose connection has been "broken." (Servers which "go down" are automatically marked as such on client computers and will not be queried for license requests until the connection is re-established.)

About Server Names

Each server has a unique name depending on the name by which the server computer is "known" on the network. Usually, the server is found for you and the name filled in automatically.

To determine a server's exact name:

1. On the server computer, maximize the HyperLynx License Server application. (If the Server application isn't running on the server, start it.) You may want to ask your network administrator to do this for you.

2. Look at the Server application's title bar. The server's name appears after the words "HyperLynx License Server on \\". The name begins immediately after the '\\' characters.

For example, on a server computer named "DEFIANT", in the title bar of the HyperLynx License Server you would see "HyperLynx License Server on \\DEFIANT". The proper server name is "DEFIANT" (not "\\DEFIANT" - no slashes).

If a Server Connection Fails

Test the connection to the server as described above. If the test fails, verify that:

- the server name is correct in your client machine's HyperLynx License Servers dialog box (see above for how to determine the proper name)
- the HyperLynx License Server application is running on the server computer
- your network is functioning properly; a good test is to verify that the client computer can see and mount shared drives on the server computer

If all of these items are verified but you still can't get the server connection to work, check with your network administrator that the client and server machine's networking configuration is properly set up. HyperLynx floating licenses uses "mailslot" network technology, which is a central mechanism in Windows networking.

Manually "Locking" a Server Connection

In rare cases, BoardSim may fail to automatically detect a server, even though the server connection works correctly if the server's name is manually

specified. If this occurs, you can manually enter the server's name, then "lock" it so that BoardSim always uses that server, and does not rely on automatic detection to find it.

To manually lock a server connection:

1. Type the server's name into one of the License Server boxes.
2. Verify that the manual connection works by clicking the Test button (see "Testing the Connection to a Server" above for details).
3. Click on the Lock Server Name check box next to the Test button.
4. Exit BoardSim and restart for the change to take effect.

Now, BoardSim will automatically use the locked server, and not rely on automatic polling to find it. Automatic detection will still occur for any other License Server boxes whose Lock check box is not enabled.

You can also use the locking feature to speed up BoardSim's licensing. E.g., you may see a brief delay when the program is first started while it polls for servers. To eliminate this, enter at least one server's name and lock it, and also lock the other, empty boxes. This will prevent BoardSim from polling at all.

Note that servers are tried in the order listed in the dialog box, from 1 to 4. If there is a server with more features or faster access that you want checked first for available licenses, manually put it first in the list.

Important! *Licenses for features on servers that are manually added and locked will not be available for use until BoardSim is exited and restarted.*

If A Server Stops Running

If a HyperLynx server ceases operation (e.g., the server computer goes down or the HyperLynx License Server application is stopped), the client computer automatically disables its connection to that server. (The client remains connected to other HyperLynx servers, if others are present on the network.) A client can re-enable a server connection by either re-starting the HyperLynx

application (i.e., closing and re-running BoardSim), or by executing a successful “test” operation as described above. If a server's features change (e.g., new licenses codes are entered on it), the new features will only be recognized on a client computer after restarting the HyperLynx application on the client machine.

User Name

When you first run BoardSim with a floating license, a dialog box appears asking for your User Name. This name appears on hardcopy printouts. It can be up to 20 characters long.

Thereafter, the User Name appears in the HyperLynx License Servers dialog box. You can change it on a particular client computer at any time.

To change your User Name:

1. From the Options menu, choose Licensing. The HyperLynx License Servers dialog box opens.

2. In the User Name area, type the new name.

Installing to Multiple Client Computers from a Network Hard Disk

If you have a floating-license copy of BoardSim and want to install it easily on multiple client computers, you can copy the installation CD-ROM to a shared network hard drive and run installation from the network drive, or “share” the CD-ROM drive and access the CD-ROM directly from remote computers.

Installing on a Remote Network Computer

BoardSim is normally installed on the computer that will be running the applications, but it is possible to install it on a remote host system. The following paragraphs describe how.

Summary of steps for installing HyperLynx applications to run on a remote computer:

1. Install the HyperLynx application software on the remote computer
2. Create a local directory for storing a local HyperLynx .INI file
3. Create a shortcut to the remote HyperLynx application
4. Add a command-line option to the shortcut that specifies the local .INI file

Detailed steps:

1. Install BoardSim on the remote host computer. Share the installation directory so that remote computers can access it. Remote users will not need "write access" to this directory; in fact, it is wise to share the directory as "read only."
2. On the local client computer, create a HyperLynx working directory, for example "C:\HYPERLYNX". Create a subdirectory beneath this directory for user data, typically "HYPFILES".
3. Create a "shortcut" on the local client computer that points to the executable file "BSW.EXE" on the remote host computer. A simple way to do this is to use the Windows Explorer: right-click on "BSW.EXE" in the shared host directory, then select "Create Shortcut". Since the shared host directory is read-only, Windows will create the shortcut on the local, client desktop.
4. Specify where the client's local version of BSW.INI will be located; this is done with a command line option as follows: edit the shortcut's properties (right-click on the shortcut and select Properties) and add "-INIF:DRIVE:\<DIRECTORY>\BSW.INI" to the Target line.

   For example:
   If the host's shared directory is " \\SERVER1\HYPERLYNX466"
   and the client's working directory is "C:\HYPERLYNX"
   then the target line is:
   \\SERVER1\HYPERLYNX466\BSE.EXE -INIF:C:\HYPERLYNX\BSW.INI

5. **Install PCB-layout translators on the local client computer, not the remote host computer.** This is required for adequate performance; some translators run slowly over the network and unlike the "main" HyperLynx applications should be run locally, not remotely.

To complete the installation, double-click the client-computer HyperLynx shortcut to run BoardSim. The program should open, ready to load a .HYP file. When you exit the program, a BSW.INI file should be written in the client computer's C:\HYPERLYNX directory.

Viewing Manuals Online This and all HyperLynx manuals are available for viewing online, in the form of .PDF (Acrobat) files. For details on installing the Adobe Acrobat viewer from the HyperLynx CD-ROM, see section "Steps for Installing the Software" above. For convenience, any of the manuals can be opened for viewing from inside BoardSim.

To open a manual .PDF file for online viewing:

1. From the Help menu, choose Manuals. The Adobe Acrobat viewer (if installed on your computer) is launched on a HyperLynx table-of-contents page.

2. In the table of contents, click on the name of the manual which you want to view. The manual file is opened.

3. View the manual using the features in the Acrobat viewer. For help with Acrobat features, refer to its online Help system.

You can also open a manual for viewing by double-clicking on it in the Windows Explorer. The manual files are located in the MANUALS sub-directory under the main BoardSim directory.

Chapter 2: Quick Start!

Summary

This chapter is a quick summary of the steps required to simulate a board in BoardSim. It is intended for:

- new users who refuse to read manuals
- experienced users who need a quick reminder

This chapter covers only the main points and the most-common operations. It is *not* a substitute for the detailed chapters in this manual or the online Help.

Steps for Simulating your Board

Some important aspects of BoardSim — for example, mapping reference designators to IC models to take advantage of BoardSim's IC AutoMapping feature, or running batch-mode, whole-board simulations — are not discussed here. See the remainder of this manual for details.

Basic steps for simulating your board in BoardSim:

1. Create a .HYP file representing your PCB layout.
2. Load the .HYP file into BoardSim.
3. Verify the board's stackup; edit if required.
4. Verify the power-supply nets; edit if required.
5. Choose a net to simulate.

6. Choose driver and receiver ICs for the net.
7. Verify passive-component values and packages; edit if necessary.
8. Attach oscilloscope probes to the net.
9. Open the oscilloscope and start simulation.

The following sections give more details.

Creating a .HYP File

BoardSim reads .HYP (HyperLynx-format) ASCII files. The .HYP file contains the information about your board's layout that is relevant to signal-integrity analysis.

There are several ways to create a .HYP file for your board, depending with which PCB-layout package you created your board. For details, refer to the PCB Translators Users' Guide.

Loading the .HYP File

To load the .HYP file into BoardSim:

1. From the File menu, choose Open BoardSim File.
   OR
   Click the Open BoardSim .HYP File button on the toolbar.

Editing a Stackup

When BoardSim loads your board, it examines the stackup in the .HYP file to determine if it is electrically valid. If not, BoardSim runs the Stackup Wizard to make corrections.

If the Stackup Wizard runs, it opens and shows a list of the changes it made to the stackup. You may need to make further corrections yourself, manually, with the stackup editor. Even if the Stackup Wizard does not run, you should verify that the stackup is correct.

Note: *A correct stackup is important because it affects the impedances of the traces on your boards. The impedances in turn affect BoardSim's simulation results.*

To edit a stackup:

1. From the Edit menu, choose Stackup.
   OR
   Click the Edit PCB Stackup button on the toolbar.

2. Edit the stackup by using the editor's buttons to add, move, delete, and edit layers. You can move layers by dragging them with the mouse. There must be at least one plane layer in your stackup, and all layers must have non-zero thickness.

Editing Power-Supply Voltages

When BoardSim loads your board, it attempts to identify power-supply nets by their names (BoardSim has certain built-in name-matching rules) and by looking at how many capacitors are connected to each net. In some cases, BoardSim cannot find all the power-supply nets on your board, so you may need to identify some of them yourself, manually, with the power-supply editor.

Note: *Identifying power-supply nets is important because BoardSim treats a power supply as a DC voltage. If, for example, the power-supply side of a pull-up resistor is mistaken for a non-power-supply net, BoardSim will simulate the resistor as a series terminator instead of a parallel terminator. Also, the Vcc and Vss pins on an IC can only be attached in BoardSim to nets identified as power supplies.*

To edit power-supply nets:

1. From the Edit menu, choose Power Supplies.

2. Determine if any power-supply nets are missing. Add any missing supplies to the list. Also, check the voltage values for each power-supply net, and change any if needed.

Choosing a Net to Simulate

To choose a net for simulation:

1. From the Select menu, choose Net by Name or Net by Reference Designator.
   OR
   Click the Select a Net by Name button on the toolbar.

2. Choose a net.

Note: *Named nets are easiest to choose By Name; unnamed nets are easiest to choose By Reference Designator.*

Choosing Driver and Receiver ICs

To simulate, BoardSim requires at least a driver IC on the net. You can also add receiver ICs. Unspecified IC models are treated as electrically "open."

To choose driver and receiver ICs:

1. From the Select menu, choose Component Models/Values.
   OR
   Click the Select Component Models and Values button on the toolbar.

2. Choose a driver IC model: double-click on the appropriate reference-designator/pin pair in the list box, and choose a library and device (and a pin/signal, if an IBIS or .PML model); click OK; set the buffer state to Output; check the driver's Vcc/Vss Pin.

3. Repeat, as needed, for other IC pins on the net, except leave the buffer state as Input.

Note: *BoardSim supports three model formats, .MOD and .PML (HyperLynx formats) and IBIS (an industry standard). .MOD and .PML models are in libraries with file extensions .MOD and .PML, and IBIS models are in libraries with extension .IBS. There is a large collection of standard-logic models in library GENERIC.MOD; there are generic technology models (e.g., 3.3-V fast CMOS) in library EASY.MOD*

Signal-integrity simulations require only models for device families, not specific devices, since only output-buffer and input-stage characteristics need be modeled.

Note: *There is another way to specify IC models, based on an ASCII reference-designator mapping file (the ".REF" file), which automatically loads models component-by-component. See Chapter 9 for details.*

Editing Passive-Component Values

When BoardSim loads your board, it attempts to identify resistor and capacitor values from data provided by your PCB-layout tool. In some cases, BoardSim is not given the correct values, so you may need to modify some of them yourself, manually.

To edit passive-component values:

1. From the Select menu, choose Component Models/Values.
   OR
   Click the Select Component Models and Values button on the toolbar.

2. Check the value of passive components by highlighting their reference designators and looking at the Value box. Correct any wrong values.

Note: *For ferrite beads, you load a model rather than setting a simple value. See Chapter 8, section "Ferrite-Bead Models" for details.*

Editing Resistor and Capacitor Packages

When BoardSim loads your board, it attempts to identify resistors and capacitors that reside in packages together with other Rs or Cs, rather than being discrete. In some cases, BoardSim cannot properly identify the package style or internal connectivity of the packaged Rs or Cs, so you may need to identify some of them yourself, manually, with the package editor.

To edit resistor and capacitor packages:

1. From the Select menu, choose Component Models/Values.
   OR
   Click the Select Component Models and Values button on the toolbar.

2. Double-click on the R's or C's reference designator in the list box, and choose the correct package and connectivity.

Attaching Oscilloscope Probes

BoardSim lets you place oscilloscope probes at any of the device pins on a net.

To attach oscilloscope probes:

1. From the Scope/Sim menu, choose Attach Probes.

2. Choose probe connections for the pins of interest.

Opening the Oscilloscope and Simulating

To run a simulation:

1. From the Scope/Sim menu, choose Run Scope.
   OR
   Click the Open Oscilloscope/Simulator button on the toolbar.

2. Select the Driver Waveform edge and change the scope's scale settings, if necessary. Begin simulating by clicking the Start Simulation button.

Congratulations — you have run your first signal-integrity simulation with BoardSim!

Chapter 3: Preparing your Board for BoardSim

Summary

This chapter describes:

- general information about translating a board into a .HYP file
- tips for how to design boards that are “friendly” to BoardSim

Translating Your Board into a .HYP File

This section describes general issues about translating your PCB-layout data into a BoardSim .HYP file. Specific information about the translator for your PCB-layout tool is contained in the PCB Translator's User's Guide

What is a “HYP File?”

A .HYP file is an ASCII file in a HyperLynx-proprietary format. It contains all of the information about a PCB layout needed for signal-integrity simulation. For each board you simulate, you run a translator on your PCB-layout data to produce a .HYP file. Then, you load the .HYP file into BoardSim and simulate.

.HYP-File Translators

The details of the .HYP-file translator depend entirely on which PCB-layout software you are using. For example...

...the translators differ in source:

- some come from the PCB-layout vendor
  (e.g., the Protel or PADS PowerPCB translator)
- some come from HyperLynx
  (e.g., the Cadence Allegro or Zuken-Redac Visula translator)
- some come from a non-HyperLynx third-party supplier (e.g., the Mentor Graphics BoardStation translator)

...and in how they run:

- some run from inside the PCB-layout tool
  (e.g., Protel and PADS PowerPCB)
- some run outside any tool, in “batch” mode
  (e.g., Cadence or Zuken-Redac Visula)

Details regarding the particular translator for which you are licensed can be found in the PCB Translator's User's Guide. This manual provides only general information that applies to all translators.

Major Elements in the .HYP File

This section describes the major elements in a .HYP file. The full .HYP-file specification is found in Appendix D.

Board Outline

The board outline data defines the shape of your board. An outline can include both linear and curved segments.

The board-outline data is optional; not all PCB-layout tools provide it. If the data is missing, BoardSim will create a rectangular outline big enough to encompass all of the components on the board.

Stackup

The stackup data defines your board's layer stackup. A stackup includes information about signal, power-plane, and dielectric layers.

The stackup data is optional; not all PCB-layout tools provide it. If the data is missing, BoardSim will attempt to create an electrically valid stackup, but will warn you to edit it.

Devices

The device data defines the components on your board. Device information includes reference designators, component names (for ICs), and component values (for passive components).

The device data is required. BoardSim must have at least some information about the devices on a net to perform a simulation.

Pad Stacks

The pad-stack data defines the various pad stacks used on your board. Pad-stack definitions are optional. Some older .HYP-file translators do not use explicit pad-stack definitions; newer ones do.

Nets

The net data defines the nets on your board. Net information includes definitions for each metal segment, via, pad, and device pin on the board.

The net information is required. BoardSim must have detailed information about trace metal to model and simulate the net.

Comment Lines

Comment lines in the .HYP file must have an asterisk (*) in the first column. On rare occasions, you may wish to remove an element from a .HYP file by commenting out the element's line. For example, if you wished to remove a resistor's pin from a certain net, you could precede the pin's record with an asterisk:

```
*(PIN  X=2.100  Y=2.350 R=Udrv1.1 P=PS4)   This is now a comment line
```

Note: *Rarely, if ever, will you need to look inside a .HYP file. But it is helpful to have a basic understanding of what the .HYP file contains. For details, see Appendix D.*

BoardSim Hint: How to Design a BoardSim-Friendly Board

BoardSim can handle your boards almost regardless of how you design them. However, this section describes several steps you can take to make your boards particularly "friendly" to BoardSim.

Using Consistent Reference Designators

BoardSim identifies what kind of component a device is — an IC, a resistor, a capacitor — by looking at the device's reference-designator prefix. For example, BoardSim might map the prefix "U" to component-type IC, and "R" to type resistor. (The "mapping rules" are user-definable; see Chapter 4, section "Setting Reference-Designator Mappings.")

Therefore, it is important to use reference-designator prefixes *consistently*. BoardSim will get confused if, for example, you call an IC "U1" and a resistor "U2". Each prefix should be unique to one component type.

Examples:

OK:	IC "U1", IC "U2", resistor "R1", capacitor "C1"
OK:	IC "X1", IC "XYZ", resistor "A001", resistor "B001"
Bad:	IC "U1", resistor "U2", resistor "A1", capacitor "A100"

Naming All Nets

BoardSim does not require nets to be named. But it is easiest to choose nets for simulation by name. If you leave a net unnamed in your board schematic, it ends up with a computer-generated name that you probably will not recognize when trying to find it in BoardSim.

Note: *For unnamed nets, BoardSim lets you choose nets by component reference designator and pin name, rather than by net name.*

Characters to Avoid in Names

When creating boards, try to avoid using names (for nets, stackup layers, and other items) that contain any of the following characters:

( ) { }

These characters are used as delimiters in BoardSim's ASCII files (like .HYP and .BUD session files; see Chapter 14 for details on session files). Names containing the delimiter characters may cause the files to be read incorrectly.

Note: *BoardSim DOES allow parentheses in IC device names, since some published IBIS libraries contain such names.*

Chapter 4: Preparing BoardSim for Your Board

Summary

This chapter describes:

- how to set reference-designator mappings
- how to set default directories
- how to set measurement units
- how to help BoardSim recognize power-supply nets
- how to load your board into BoardSim

Setting Reference-Designator Mappings

This section describes how to define the reference-designator mappings that BoardSim uses to identify component types (IC, R, C, L, and ferrite bead).

What is a Reference-Designator Mapping?

Component Types

When BoardSim loads your board, it examines the list of devices in the .HYP file and tries to determine the component type of each device. BoardSim must know component types in order to simulate correctly.

BoardSim supports these component types:

- IC (any driver or receiver device)
- resistor
- capacitor
- inductor
- ferrite bead
- test point

Although BoardSim does not have direct support for other component types (like connectors or transistors), this does *not* mean that you cannot simulate nets that include other types. See “BoardSim Hint: How to Simulate Unsupported Component Types” below in this chapter for details.

The component type is unrelated to how a component is packaged. A discrete resistor and the resistors in an R network are both type “resistor.” Package types for R and C components are handled separately from component types; see Chapter 11 for details.

Reference-Designator Prefixes

BoardSim determines each device’s type by looking at the device’s reference-designator prefix. “Prefix” means the first part of the reference designator (the part that stays the same for components of the same type).

For example, if you give all of the ICs on your board a reference designator of the form “Uxx” (U1, U2, U3A, U3B, etc.), then “U” would be the reference-designator prefix for ICs. Resistors would commonly have a prefix of “R”. You

might also have some resistor networks that you call "RPxx" (RP1, RP2, etc.), so "RP" might also be a valid prefix for resistors.

You are free to assign whatever reference designators you want to the devices on your board; BoardSim's reference-designator mappings are user-definable (see "Editing Reference-Designator Mappings" below in this chapter for details).

However, you should not assign the same reference-designator prefix to more than one component type. BoardSim will get confused if, for example, you call an IC "U1" and a resistor "U2". Each prefix can map to only one component type. See Chapter 3, section "How to Design a BoardSim-Friendly Board" for more details.

Note: *There is a workaround for situations in which mapping the same reference-designator prefix to more than one component type cannot be avoided; see "BoardSim Hint: How to Map a Reference-Designator Prefix to Multiple Component Types" below in this chapter for details.*

How Components are Specified in the .HYP File

All of the components on a board are listed in a DEVICES record in the .HYP file. Regarding component types, there are two ways that devices can appear in the DEVICES list.

? Record (Preferred)

Usually, a .HYP-file translator lists each device in a record of type "?". This means that the translator does not know the device's component type, and that the type will be determined by BoardSim's reference-designator mapping rules. This is the preferred way of handling devices in the .HYP file; it leaves you the flexibility to map reference designators however you wish.

"Hardcoded" Records (Not Recommended)

The .HYP-file specification also allows for device component types to be "hardcoded" in the .HYP file. There is support for records of type "IC", "R", "C", "L", and "BD" (ferrite bead) in BoardSim; these records force the corresponding device to be typed as IC, R, C, L, or BD regardless of its reference designator.

See “BoardSim Hint: How to Map a Reference-Designator Prefix to Multiple Component Types” below in this chapter for more details.

Default Reference-Designator Mappings

BoardSim has a set of default mappings for reference-designator prefixes that it uses to identify the component types of devices in the .HYP file. If your (or your company's) rules for reference designators match BoardSim's defaults, you do not need to change the mappings at all.

To see the default mappings:

1. From the Options menu, choose Reference Designator Mappings.
2. The default mappings are shown in the Mappings list box.

Note that reference-designator prefixes for components that are not directly supported by BoardSim's simulator are usually mapped to type “IC”. This allows you to use an IC model of some kind to substitute for the unsupported component. See “BoardSim Hint: How to Simulate Unsupported Component Types” in this chapter for details.

Editing Reference-Designator Mappings

Changing a Mapping

To change a reference-designator mapping:

1. From the Options menu, choose Reference Designator Mappings.
2. In the Mappings list box, highlight the mapping you want to change.
3. In the Edit/Add Selected Mapping area, change the mapping by clicking a new radio button, then clicking the Add/Apply button. The entry in the Mappings list box changes.
4. Make other desired changes, then click OK.

Adding a Mapping

To add a new reference-designator mapping:

1. From the Options menu, choose Reference Designator Mappings.
2. In the Ref Prefix box, type the new reference-designator prefix that you want to map.
3. In the Edit/Add Selected Mapping area, specify the mapping by clicking the radio button for the desired component type; then click the Add/Apply button. The new entry appears in the Mappings list box.
4. Make any other changes, and then click OK.

The location of the new entry as it automatically appears in the mappings list is significant. See "Search Order for Mappings" later in this chapter for details.

Deleting a Mapping

To delete a mapping:

1. From the Options menu, choose Reference Designator Mappings.
2. In the Mappings list box, highlight the mapping you want to delete.
3. Click the Delete button to cause the entry in the Mappings list box to disappear.
4. Make other changes, and then click OK.

Mappings for prefixes that have a default value cannot be deleted. However, default mappings can be changed; see "Changing a Mapping" above in this chapter for details.

Restoring Default Mappings

To restore BoardSim's default mappings:

1. From the Options menu, choose Reference Designator Mappings.
2. Click the Defaults button.

3. When a warning dialog box opens, click OK to overwrite the current mappings.

How Mappings are Saved

BoardSim's reference-designator mappings are saved globally, so they apply to any board you load.

When Changes Take Effect

BoardSim examines the devices in a .HYP file as it loads the file. Therefore, if you make changes to reference-designator mappings after a board is loaded, you must re-load the board before the changes take effect. (A better strategy is to change the mappings before you load the board.)

Search Order for Mappings

The order in which the reference-designator mappings appear in the Mappings list box is significant. When BoardSim examines a device's reference designator, it searches for a prefix match starting with the mappings at the top of the list, and moving toward the bottom, until the first match is found.

Notice that when you add mappings, two-character mappings appear above single-character, three-character above two-character, etc. This ensures that a mapping like "RP" will be found even if there is also a mapping for "R". (For example, if BoardSim searched on reference designator "RP31" and reached mapping "R" first, an inferior match might occur.)

Within each group of equal character size, the search ordering is alphabetical.

When to Set Mappings

BoardSim examines the devices in a .HYP file as it loads the file. Therefore, if you plan to change the reference-designator mappings for a board, you must change them *before* you load the board's .HYP file. (The Options menu is available whether or not a board is loaded, so that you can access the Reference Designator Mappings selection even without a .HYP file loaded.)

If you load the board first, then change the mappings, you must re-load the board for the new mappings to take effect.

Test Points

In BoardSim, a “test point” is a component pin that is ignored for board viewing and for simulation. The need for a test point component type arises because many board-layout systems treat PCB test points as if they were real component pins, which confuses BoardSim into thinking that the test points need device models. If you identify your test points with appropriate reference-designator mappings, then BoardSim will correctly ignore the test points when it simulates.

To add a reference-designator mapping for test points:

1. From the Options menu, choose Reference Designator Mappings.
2. In the Ref Prefix box, type the new reference-designator prefix that you want to map as meaning “test point.”
3. In the Edit/Add Selected Mapping area, specify the mapping by clicking the Test Point radio button; then click the Add/Apply button. The new entry appears in the Mappings list box.
4. Make other desired changes, then click OK.

Default Mapping for Test Points

BoardSim contains one default mapping for test points: the prefix “TP” is assumed to indicate a test point (i.e., components named TP1, TP2, etc. will automatically map as test points.)

If you want “TP” to indicate something other than “test point,” you can change the mapping to another component type. (See “Changing a Mapping” above in this chapter for details.)

One-Pin Components Automatically Treated as Test Points

BoardSim contains one other method for automatically identifying test points: all one-pin components are automatically assumed to be test points. (For

example, if “F5” is a component that has only one pin, F5 will automatically be treated as a test point, and ignored for simulation. The same would be true of a one-pin component called U1.) This feature is not controlled by a reference-designator mapping and cannot be overridden; it overrides all reference-designator mappings.

How Test Points Behave

Test points:

- cannot be seen in the board viewer
- cannot have oscilloscope probes attached to them
- cannot have device models selected for them
- are completely ignored during simulation in that they are treated as electrical “opens”

Setting Directories

What the Directories Settings Do

BoardSim allows you to specify default directories for:

- the location of your .HYP files
- the location of BoardSim's and your device-model libraries

By default, BoardSim creates two subdirectories under the directory it is installed in:

- HYPFILES for storing your .HYP files
- LIBS for storing device-model libraries

The HYPFILES directory setting is just a convenience: it specifies the default directory that the Open BoardSim File dialog box opens on. You can change in

the dialog box to any other directory, which means that you can store your .HYP files anywhere you wish, and in multiple directories if you want. You can also instruct BoardSim to always open the file dialog box on the directory from which you *last* loaded a .HYP file (see section "Setting the Default .HYP Files Path" below).

The LIBS directory setting specifies the location of BoardSim's IC-model libraries (i.e., .MOD, .PML and .IBS files; see Chapter 8, section "Interactively Choosing IC Models" for details on device-modeling libraries and formats). Unlike with the HYPFILES setting, the LIBS directory is the *only* directory that BoardSim looks for IC models in. *Note that all of your IC libraries must be in the directory you designate as the models path. Libraries not in this directory are "invisible" to BoardSim.*

If you own LineSim as well as BoardSim, your LineSim schematic (i.e., .TLN) files are also stored in the HYPFILES directory. For this reason, the Set Directories dialog box described below refers to both .HYP and .TLN files.

Setting the Default .HYP Files Path

To set the HYPFILES path to a particular directory:

1. From the Options menu, choose Directories.
2. Verify that the Use Directory of Last-Opened File check box is disabled.
3. In the .HYP and .TLN File Path edit box, type the desired directory. (You can type it with or without a trailing '\'.)
   OR
   Click the Browse button; a dialog box opens. Find the directory you want to use and double-click on a .HYP or .TLN file in the directory to set the path. (You can also highlight a .HYP or .TLN file and click Open.)
4. Click OK.

The change takes effect immediately.

To tell BoardSim to always use the directory from which you *last* loaded a .HYP file (rather than a particular "fixed" directory):

1. From the Options menu, choose Directories.
2. Click on the Use Directory of Last-Opened File check box, to enable it.
3. Click OK.
4. To restore the .HYP files directory to its default value. From the Options menu, choose Directories.
5. Verify that the Use Directory of Last-Opened File check box is disabled.
6. In the .HYP and .TLN File Path edit area, click Default. The path is reset to its default value.
7. Click OK.

Setting the Model Library File Path

To set the LIBS directory:

1. From the Options menu, choose Directories.
2. In the Model Library File Path edit box, type the desired directory. (You can type it with or without a trailing '\'.)
   OR
   Click the Browse button; a dialog box opens. Find the directory where your IC models are stored and double-click on a model file to set the path. (You can also highlight a file and click Open.)
3. Click OK.

The change takes effect immediately.

To restore the model-libraries directory to its default value:

1. From the Options menu, choose Directories.

2. In the Model Library File Path area, click Default. The path is reset to its default value.

3. Click OK.

Choosing Measurement Units

What the Measurement Units Do

BoardSim allows you to choose in which units you want to view your board's lengths and metal thicknesses.

For lengths, you choose a measurement system:

- English
- metric

For metal thicknesses (base copper thickness and plating thickness), you choose between:

- length
- weight

The default settings are "English" and "weight," e.g., X,Y positions in inches and base copper in ounces. International users might prefer "metric" and "length," e.g., X,Y positions in centimeters and base copper in microns.

Choosing the Measurement System and Metal-Thickness Units

To choose the measurement system:

1. From the Options menu, choose Units.

2. In the Measurement Units area, click the appropriate radio button.

3. Click OK.

To choose the metal-thickness units:

1. From the Options menu, choose Units.
2. In the Metal Thickness Units area, click the appropriate radio button.
3. Click OK.

The changes takes effect immediately.

Changing Measurement Units from Inside the Stackup Editor

As a convenience, you can change measurement units from directly inside the stackup editor. (For details on the stackup editor, see Chapter 5, section "Editing a Stackup.")

To change measurements units from inside the stackup editor:

1. In the Edit Stackup dialog box, click on the Measurement Units button, then proceed as described above.

Helping BoardSim Recognize Power-Supply Nets

BoardSim uses several methods to recognize which nets on your board are power-supply nets. (See Chapter 6, "Why Power-Supply Nets Matter" for details on the importance of power-supply nets being properly identified.)

The first method involves BoardSim recognizing certain names commonly used for power-supply nets (e.g., "GND" or "VCC"; see Chapter 6, section "How BoardSim Identifies Power-Supply Nets" for details.) Another method involves BoardSim counting the number of capacitors connected to each of the board's nets, and whenever the number of capacitors exceeds a threshold value (which you can change), considering that net to be a power supply. A third method

assumes that nets with more metal segments than some very large number (which you can change) must be power supplies.

Note: *The capacitor-counting method was added after the initial release of BoardSim to make finding power-supply nets more likely to succeed. The capacitor-based algorithm has proven successful across a broad spectrum of customer designs. Note that it has the advantage of finding not only power-supply nets (which typically have large numbers of decoupling capacitors connected), but also analog nets, which, like power-supplies, should not be simulated as digital nets in BoardSim.*
The segment-counting method was added later in response to several extremely large customer boards on which some power supplies slipped past the other two identification methods (names and numbers of capacitors). Because very large nets tend to slow the loading of a .HYP file (it takes longer to build database information for large nets), this improvement had the added benefit of improving the time required to load large .HYP files.

Changing the Number-of-Capacitors Threshold

By default, BoardSim considers any net with three or more capacitors connected to be a power-supply net. (If this identification is ever wrong, the misidentified net can be removed from the power-supplies list using the power-supply editor; see Chapter 6, section “Editing Power-Supply Nets” for details.) However, you can modify the identification threshold to any number of capacitors you want.

To modify the number of capacitors used as a threshold to identify power-supply nets:

1. From the Options menu, choose Preferences.
2. Click the BoardSim tab.
3. In the Net Handling area, in the Net is a Power Supply edit box, type the desired number of capacitors to use as a threshold.
4. Click OK.

The change takes effect next time you load a board.

Note: *HyperLynx has experimented with a large number of boards and determined that "3" is usually the optimal threshold value. Do not change the threshold unless you're certain that a change is needed for one of your PCBs.*

Changing the Number-of-Segments Threshold

By default, BoardSim considers any net with 20,000 or more individual metal segments to be a power-supply net. (If this identification is ever wrong, the misidentified net can be removed from the power-supplies list using the power-supply editor; see Chapter 6, section "Editing Power-Supply Nets" for details.) However, you can modify the identification threshold to any number of segments you want.

To modify the number of metal segments used as a threshold to identify power-supply nets:

1. From the Options menu, choose Preferences.
2. Click the Advanced tab; click "Yes" when queried about making changes.
3. In the BoardSim area, in the Segment Threshold for Auto Power-Supply ID edit box, type the desired number of segments to use as a threshold.
4. Click OK.

The change takes effect next time you load a board.

Note: *HyperLynx has experimented with a large number of boards and determined that "20,000" is usually an optimal threshold value. Do not change the threshold unless you're certain that a change is needed for one of your PCBs.*

Loading Your Board into BoardSim

Once BoardSim is prepared for your board, you can load the board's .HYP file and begin analysis. (Before you load, be certain that reference-designator

mappings are set; see "Setting Reference-Designator Mappings" above in this chapter for details.)

Note: *First, of course, you must have a .HYP file for your board. For details on creating the .HYP file, see Chapter 3, section "Translating Your Board into a .HYP File."*

To load the .HYP file into BoardSim:

1. From the File menu, choose Open BoardSim File.
   OR
   Click the Open BoardSim .HYP File button on the toolbar.

2. Choose the .HYP file for your board.

3. Click Open.

As the Board Loads

As your board loads, the Loading .HYP File dialog box gives percent-done status. For large boards, it may take a few minutes for the .HYP file to load.

First, BoardSim counts the number of nets in the file. (Messages about the current activity appear in the dialog box.) Then the file's details are read into BoardSim's database. Most of the loading time is spent reading net data; the name of the net currently being read is shown in the status message.

After all of the nets are loaded (when the percent-done indication = 100%), BoardSim makes a second pass to find which nets are connected to which other nets. Then, after several more seconds to sort the net names, characterize the stackup, and draw the board, the PCB appears, as an outline filled with components, in the board viewer. New menu choices appear above the viewer. (See Chapter 7, section "The Board Viewer" for details.)

Net Cleaning

Many PCB-layout programs make little or no attempt to "clean up" redundant or overlapping trace segments on a board. (Such redundancy is particularly common in designs that have been routed at least partially by hand.)

Redundancy makes no difference when a Gerber file is output, but is not acceptable to simulation tools like BoardSim that assign electrical characteristics to all metal structures on a net.

Accordingly, BoardSim “cleans” all nets before you analyze them, eliminating redundant metal and combining overlapping structures when possible into fewer, large structures. This guarantees accurate simulation results. The cleaning process actually occurs when a net is first selected.

The advantage to cleaning nets only when they are selected is that the task is distributed: the other option is to clean all nets at board-loading time, but this effort increases (usually by about a factor of two) the time required to load a board.

However, there is one disadvantage to cleaning nets only when selected. The net lengths displayed in the Select Net by Name dialog box are calculated at board-loading time. If net cleaning occurs only later when nets are selected, and if some nets contain large amounts of redundant metal, then the lengths reported in the dialog box may be too long. To avoid this problem, you can instruct BoardSim clean all nets at board-loading time.

To enable cleanup of all nets at .HYP-file-loading time:

1. From the Options menu, choose Preferences. The Options dialog box opens.
2. Click on the BoardSim tab.
3. In the Net Handling area, click on the Remove Redundant Metal... check box to enable it.

Now, when you load a board, BoardSim will make a “cleanup” pass that runs on all of the nets on your PCB. The Loading .HYP File dialog box shows you the progress of this pass, including the name of the net currently being cleaned and the total number of metal redundancies found and fixed.

Recommendation about Net Cleaning

Since your boards will load into BoardSim faster if you do NOT enable the “net cleaning during loading” option, HyperLynx generally recommends that you

not enable this option. The only exception would be if the net lengths reported in the Select Net by Name dialog box seems too long, indicating that your nets contain a significant amount of redundant metal; then you might prefer to have more-accurate lengths, at the expense of longer board-loading times.

About "Field Solver" Messages

In preparation for performing crosstalk analysis, BoardSim briefly calls its field solver near the end of the board-loading process to characterize certain aspects of the PCB's stackup. Often, you will see a progress dialog box labeled "HyperLynx" and "Running field solver" while this analysis is running.

The field solver is called regardless of whether or not you are licensed for BoardSim's Crosstalk option. If you are not licensed for Crosstalk analysis, then this (and when certain other changes, like stackup editing, occur) is the only time the field solver runs; it is not available during simulation or any other kind of analysis unless you own the Crosstalk option.

Out-of-Memory Errors

If you get an "out-of-memory" error while the .HYP file is loading, you do not have enough free memory for BoardSim to store your board in its database. This can occur if your board is very large and some of your PC's memory is used by other applications.

If you get "out of memory" errors from BoardSim, try closing other open applications, or freeing more memory for Windows to use. Note that BoardSim requires at least as much memory to load your board as does your PCB-layout tool, since BoardSim reads most of the data in the PCB-layout database, and then adds electrical information (like net connectivity) to it.

BoardSim Hint: How to Simulate Unsupported Component Types

BoardSim's simulator does not have direct support for some component types. (The supported components are IC, R, C, L, and ferrite bead; see "What is a

Reference-Designator Mapping?" above in this chapter for details.) However, you can simulate nets that include non-supported components by using components that *are* supported as substitutes.

Connectors

The current version of BoardSim does not allow you to simulate multiple boards simultaneously. Boards that are connected through a connector must be simulated individually.

To simulate a net for which the driving signal arrives through a connector, map the connector's reference-designator prefix to component type "IC". (The mapping for prefix "J" defaults to "IC".) Then choose an IC model for the connector pin and simulate the signal integrity from the pin forward onto the current board.

For example, for a net that starts at pin 2 on connector J1, choose an IC model for J1.2 and add receiver models as appropriate for other IC pins on the net. Then simulate.

Diodes

The current version of BoardSim does not explicitly support diodes. However, either of BoardSim's IC-modeling formats supports clamp diodes, so you can use an IC model to describe a discrete clamp diode or diode terminating network. (The mappings for prefixes "CR" and "D" default to "IC".)

For example, for a net that is clamped by pin A on a clamp diode CR3, choose a receiver-IC model for CR3.A.

You can construct your own diode model by modifying a .MOD model or an IBIS file. BoardSim ships with a library called DIODES.MOD that shows some sample clamp diodes implemented in the .MOD format. Note that these models use only the "input" side of the .MOD description; it makes no sense to run them as outputs. For more details on how to create and edit .MOD models, see Chapter 10, section "Editing .MOD IC Models."

Other Component Types

Here are some suggestions for how to model other non-supported component types in BoardSim:

- transistor — model as "IC"
- relay — "IC"
- crystal — "IC"

BoardSim Hint: How to Map a Reference-Designator Prefix to Multiple Component Types

BoardSim maps each reference-designator prefix to one component type. If you have a board on which a single prefix must map to *more* than one type, you can use the workaround described below.

Suppose you have a board on which the prefix "U" is used mostly for ICs, but also for two terminating networks (U1 and U20) that are actually resistors (component type "R"). (See "What is a Reference-Designator Mapping?" above in this chapter for details on component types.)

Follow these steps to map most of the U's as ICs but U1 and U20 as "R":

1. From the Options menu, choose Reference Designator Mappings to verify that prefix "U" is mapped to component type "IC". (This mapping is one of BoardSim's default mappings.)
2. Open the .HYP file for your board in a text editor (e.g., the HyperLynx File Editor).
3. Search from the top of the .HYP file for the keyword DEVICES.
4. Then search for "U1". You should find a line that starts:

   (? REF=U1 ...
5. Change the line so that it starts:

(R REF=U1 ...

6. Go back to the DEVICES keyword.

7. Repeat 4 and 5 for reference designator “U20”.

8. Save the edited .HYP file and exit.

Note: *You may have trouble editing the .HYP file because of its size; be sure you use a text editor that can handle large files, like the HyperLynx File Editor. Also, it is a good idea to make a copy of the .HYP file before editing it, in case you introduce an error into the file and need to restore it. Finally, be sure you use a text editor, not an editor that introduces non-ASCII formatting characters into the file.*

Now, when you load the .HYP file, all of the reference designators prefixed with “U” but marked in the DEVICES list as type “?” will be mapped according to the “U = IC” mapping rule. But U1 and U20, since they are now “hardcoded” as type “R”, will be forced to map as resistors.

Chapter 5: Editing Stackups and Trace Widths

Summary

This chapter describes:

- what elements make up a stackup
- why stackups are important to BoardSim
- how BoardSim reads a stackup
- how BoardSim's Stackup Wizard works
- how to edit a stackup
- how to calculate characteristic impedances
- how to document a stackup
- some restrictions BoardSim places on stackups
- how to change a board's trace widths in BoardSim

Elements of a Stackup

"Stackup" refers to the how the metal and dielectric layers in a board are ordered and constructed.

There are three types of layers in a board stackup:

- signal — a layer that carries signal traces
- plane — a layer of solid metal, tied to a DC voltage
- dielectric — a non-conducting layer separating two metal layers

The following sections describe in greater detail the geometric and material parameters of signal, plane, and dielectric layers, and of a complete stackup.

Plane Layers

A plane layer is a solid metal layer that is tied to a DC voltage, e.g., VCC or ground. Plane layers function electrically as AC grounds.

BoardSim Assumptions about Plane Layers

The current version of BoardSim assumes that a board has at least one plane layer that allows for effective ground-return currents for the traces on the board. “Effective” means that the plane layer provides a low-inductance, close-to-the-trace return path.

From a practical standpoint, this means BoardSim assumes that:

- every board has at least one plane layer
- plane layers are solid, not hatched or otherwise seriously “broken”
- plane layers are complete, not partial or mixed significantly with signal traces

If a plane layer in a stackup *is* seriously “compromised” in one of these ways, some of BoardSim’s impedance calculations may be inaccurate. BoardSim also assumes that:

- planes layers are reported as nets, so they can be connected to by terminating components and assigned to power-supply voltages

The first assumption, that every board has at least one plane layer, restricts the current version of BoardSim from simulating boards that are single-sided or double-sided with no ground plane. The remaining assumptions do not prevent simulation, but may lessen its accuracy in some cases.

Note that the above restrictions do NOT mean that BoardSim cannot simulate boards with planes that are not completely perfect. For example, consider a board with a split 5-V/3.3-V power plane. BoardSim can handle perfectly well any traces which are isolated to one side or the other of the split (i.e., the 5-V side or 3.3-V side), because the return currents for such traces are never interrupted by the power-plane gap. Traces which cross the split, on the other hand, may be problematic, although even they may simulate with sufficient accuracy if enough bypass capacitors are available in the vicinity of the trace's crossing of the gap to keep the trace's return current from deviating too wildly from the signal current.

Signal Layers

A signal layer is a metal layer that contains signal traces. Traces may move between signal layers through vias.

Signal layers are classified into categories depending on how they are positioned relative to the plane layers in a stackup. The individual segments on a trace can be in differing categories, since various segments on a single trace can be on different layers.

Microstrip

A "microstrip" is a trace segment on a layer with the following characteristics:

- has a dielectric + a plane layer on one side
- has only air on the other side

This describes an outer-layer trace on a board. Microstrip traces usually have higher impedances than traces of other types.

Buried Microstrip

A “buried microstrip” is a trace segment on a layer with the following characteristics:

- has a dielectric + a plane layer on one side
- has a non-air dielectric + air on the other side

This describes an inner-layer trace with a plane layer on only one side.

Stripline

A “stripline” is a trace segment on a layer with the following characteristics:

- has a dielectric + a plane layer on both sides

This describes an inner-layer trace between two plane layers. Stripline traces usually have lower impedances than traces of other types.

Dielectric Layers

A dielectric layer is a non-conducting layer that separates two metal layers. Dielectric layers can be made from a variety of materials, though fiberglass is the most common in PCBs.

Dielectric Constants

Associated with every dielectric material is a property called “relative permittivity,” or “dielectric constant.” Dielectric constant measures how effective a material is in establishing a capacitance.

For a table of dielectric constants for common board dielectrics, see Appendix E.

Why Stackups Matter

Characteristic Impedance and Propagation Velocity

Nearly every detail of a board's stackup affects two key characteristics of the trace segments on a board:

- characteristic impedance (Z0)
- propagation velocity

Together, these parameters determine how signals interact with and propagate along the traces on a board.

Characteristic impedance (or "Z0") is a property unique to the distributed nature of transmission lines. Because transmission lines consist of a continuous mixture of capacitance and inductance, they "look" instantaneously to a transmitted signal like a resistance.

Transmission-line impedance affects such behavior as signal reflection and step size (the percentage of a switching signal's swing that enters a transmission line).

Propagation velocity specifies how quickly a signal travels along a transmission line.

Propagation velocity determines whether or not a signal trace is likely to exhibit transmission-line effects. If the total delay time down a trace is short compared to how fast the driving IC switches, the trace will not behave much like transmission line. If the delay time is long, the transmission-line effects become significant.

Stackup Parameters that Affect Impedance and Velocity

The following stackup parameters all affect the characteristic impedances and propagation velocities of the trace segments on a board:

- layer order

- trace thickness
- trace width
- dielectric thickness
- dielectric constant

BoardSim lets you edit all of these parameters in its stackup editor, except for trace width.

Note: *Trace width also affects characteristic impedance and propagation velocity, but is not considered by BoardSim to be a "stackup parameter." Rather, it varies trace-by-trace in your PCB layout.*

How BoardSim Reads Stackups

Stackup in the .HYP File

When BoardSim loads your board, it looks for stackup data in the .HYP file. If there is a STACKUP record, BoardSim reads and constructs a stackup.

BoardSim also looks to see if there is a previous session (.BUD, or BoardSim User Data) file for the board. (See Chapter 14 for details on session files.) If so, and if there are stackup edits recorded in the session file, BoardSim incorporates them into the stackup.

After the .HYP and session files are read, BoardSim examines the stackup to determine if it is electrically valid. If not, BoardSim runs the Stackup Wizard to make corrections (see "The Stackup Wizard: How BoardSim Corrects an 'Illegal' Stackup" below in this chapter for details).

Note: *Actually, BoardSim gives the option, when you load your board, of whether or not to read the session file. You can even choose to read the session file but selectively ignore portions of it, e.g., the stackup. See Chapter 14 for details.*

How a Stackup can be Incomplete

No Stackup at All

The most obvious deficiency in a .HYP file's stackup is having no stackup at all. This is not uncommon; some PCB-layout tools do not carry or use stackup information.

When the stackup is completely missing, the Stackup Wizard attempts to synthesize one. Since it is unlikely that the synthesized stackup matches your real stackup exactly, you should run the stackup editor and make any required changes (see "Editing a Stackup" below in this chapter for details).

Other Problems

Even if there is a stackup provided in the .HYP file, there can be other errors that make the stackup electrically invalid. The Stackup Wizard checks for the following conditions and remedies them, if possible:

- missing layers
- zero or missing thicknesses
- zero or missing dielectric constants

The Stackup Wizard: How BoardSim Corrects an "Illegal" Stackup

The Stackup Wizard runs after your board is loaded to check the electrical validity of the board's stackup. If the stackup is completely missing or invalid in some respect, the Stackup Wizard attempts to synthesize a correct stackup.

If the stackup is electrically valid, the Wizard runs invisibly and does not report to the screen. If the stackup has errors, the Wizard opens a dialog box that reports the errors and how the Wizard attempted to correct them.

The first time you load a new board into BoardSim, it is a good idea to check the stackup, even if the Stackup Wizard does not report any

errors. The stackup in the .HYP file can be electrically valid but still not match your real stackup.

How the Stackup Wizard Corrects Errors

Missing Layers

Signal Layers

If there is no stackup in the .HYP or session (.BUD) files, BoardSim automatically records the layers of all the trace segments on the board, and adds the layers to the stackup. Therefore, there should never be any signal layers missing from the stackup.

Synthesized (i.e., automatically created) signal layers are given a BoardSim-created layer name.

Note: *BoardSim automatically detects signal layers that are used by at least one trace segment somewhere in the .HYP file. If you think a signal layer is missing, consider whether it is actually used by a routed trace on your board. If not, it does not need to be included in your BoardSim stackup.*

Plane Layers

The Stackup Wizard detects a board that has *no* plane layers, and reports an error. The current version of BoardSim requires every board to have at least one plane layer.

Since the Wizard has no idea where the missing plane goes in the stackup, and since the positioning of plane layers is so critical to trace impedance, the Wizard does not automatically insert a plane. Use the stackup editor to add plane layers.

If there is at least one plane layer in the stackup, the Stackup Wizard does not report any plane-layer-related errors. If there are several plane layers on your board and some (but not all) are missing, the Wizard will not detect it.

Dielectric Layers

The Stackup Wizard detects metal layers that are shorted together because there is no dielectric between them, and reports the condition.

The Wizard automatically adds a dielectric layer between shorted metal layers. Synthesized dielectric layers are given a default thickness and dielectric constant.

It is possible for a dielectric layer to be missing because it is out-of-order in the .HYP-file stackup. The synthesized layer will then be redundant; use the stackup editor to delete the synthesized layer and move the real one.

Zero Thicknesses

The Stackup Wizard detects layers (signal, plane, and dielectric) that have zero or missing thicknesses. BoardSim requires every layer to have a non-zero thickness.

The Wizard automatically changes zero thicknesses to a default thickness. (The default thickness differs for metal and dielectric layers.)

Zero Dielectric Constants

The Stackup Wizard detects dielectric layers that have zero or missing dielectric constants. BoardSim requires every dielectric to have a non-zero constant.

The Wizard automatically changes zero dielectric constants to a default constant.

No Stackup at All

If there is no stackup at all in the .HYP or session (.BUD) files, the Stackup Wizard takes the following steps:

1. Creates signal layers for all the layers on which there are trace segments.
2. Separates the signal layers with dielectric layers.
3. Sets signal-layer thicknesses to a default thickness.

4. Sets dielectric thicknesses and constants to default values.

5. Warns you that there is still no plane layer.

Editing a Stackup

There are several reasons you might edit your board's stackup in BoardSim:

1. You just loaded a new board into BoardSim, and the Stackup Wizard reports there were errors.

2. You just loaded a new board into BoardSim; the Stackup Wizard did *not* report any errors, but you want to verify that the stackup matches exactly what you expected.

3. You want to experiment with a different stackup, since stackup affects trace-segment impedance and therefore signal integrity.

4. You want a reminder of what stackup you are currently using.

5. You want to print or document your stackup.

For any of these needs, run the stackup editor as described below.

Opening the Stackup Editor:

To open the stackup editor:

1. From the Edit menu, choose Stackup.
   OR
   Click the Edit PCB Stackup button on the toolbar.

Stackup-Error Reporting

Current Errors

If there are currently any errors in the stackup, or if an editing change you make to the stackup causes an error, the stackup editor reports the error

immediately. Errors appear in a status line at the top of the stackup-editor dialog box, above the graphical area that displays the stackup. Errors are reported in a red font.

If there are no errors, i.e., the stackup is electrically valid, the status line changes to a black font and reports “no errors.”

If there are multiple errors simultaneously, the status line reports them one-at-a-time. Continue fixing the indicated errors until the status line reports no errors.

Always watch the status line as you edit a stackup. The status messages tell you immediately if an editing change has made the stackup invalid.

Previous Stackup Wizard Report

You can view the results from the previous time the Stackup Wizard ran, to see what errors existed then and what changes (if any) the Wizard made to your stackup. (See “The Stackup Wizard: How BoardSim Corrects an ‘Illegal’ Stackup” above in this chapter for details on the Wizard.)

To view the previous Stackup Wizard results:

1. In the stackup editor, click the View Stackup Wizard button.

The Stackup Wizard Report dialog box opens, showing the results.

Note: *The results can be from either the automatic run of the Wizard that occurs when you load your board, or from a manual run of the Wizard that you force from the Wizards menu — whichever occurred last. Clicking the View Stackup Wizard button does not re-run the Wizard; it only opens the last report generated by the Wizard.*
The most-typical use for this button is to conveniently review what changes the Wizard made to an illegal stackup at board-load time.

Changing Layer Order

To change the order of a stackup's layers:

1. In the stackup editor, in the graphical area that displays the stackup, position the mouse cursor over the layer you want to move.

2. Click the mouse button and continue holding it down, like you are beginning a drag-and-drop operation. Notice how the cursor changes to show that you have "grabbed" the layer.

3. Drag the layer up or down until the layer cursor's arrow is positioned at the place where you want to move the layer.

4. Release the mouse button. The layer moves to its new position.

Adding Layers

Adding Dielectric Layers

To add a dielectric layer:

1. In the stackup editor, in the graphical area that displays the stackup, click once on the layer that you want to add a dielectric above or below. Notice that a highlight box appears around the clicked-on layer.

2. Click the Dielectric radio button in the area with the Add Layer buttons.

3. To add a dielectric above the highlighted layer, click the Add Layer ^ button. To add a dielectric below the highlighted layer, click the Add Layer v button.

The new dielectric layer is given a default thickness and dielectric constant; you can change these default values (see section "Setting Default Layer Characteristics" below for details). To edit the new layer, see "Editing Dielectric Layers" below in this chapter.

Adding Plane Layers

To add a plane layer:

1. In the stackup editor, in the graphical area that displays the stackup, click once on the layer that you want to add a plane above or below. Notice that a highlight box appears around the clicked-on layer.

2. Click the Plane radio button in the area with the Add Layer buttons.

3. To add a plane above the highlighted layer, click the Add Layer ^ button. To add a plane below the highlighted layer, click the Add Layer v button.

If you highlighted a metal layer, so that the new plane would have been shorted to it, a new dielectric layer is automatically added between the metal layer and the new plane. (The new dielectric layer is given a default thickness and dielectric constant.)

The new plane layer is given a default thickness; you can change this default value (see section “Setting Default Layer Characteristics” below for details). To edit the new layer, see “Editing Plane or Signal Layers” below in this chapter.

Adding Signal Layers

Signal layers are automatically recorded when a .HYP file is loaded: BoardSim records the layers of all the trace segments on the board. Therefore, there should never be any signal layers missing from the stackup that involve routed traces on your board. See “How the Stackup Wizard Corrects Errors” above in this chapter for more details.

However, you can use the stackup editor to add signal layers (e.g., to document a layer you plan to add in a future board revision, or to see how a new signal layer would affect trace impedances).

To add a signal layer:

1. In the stackup editor, in the graphical area that displays the stackup, click once on the layer that you want to add a signal layer above or below. Notice that a highlight box appears around the clicked-on layer.

2. Click the Signal radio button in the area with the Add Layer buttons.

3. To add a signal layer above the highlighted layer, click the Add Layer ^ button. To add a signal layer below the highlighted layer, click the Add Layer v button.

If you highlighted a metal layer, so that the new signal layer would have been shorted to it, a new dielectric layer is automatically added between the metal layer and the new signal layer. (The new dielectric layer is given a default thickness and dielectric constant.)

The new signal layer is given a default thickness; you can change this default value (see section “Setting Default Layer Characteristics” below for details). To edit the new layer, see “Editing Plane or Signal Layers” below in this chapter.

Setting Default Layer Characteristics

As you edit a stackup and add layers to it (see the preceding sections for details), BoardSim applies default values to the new layers (e.g., for thickness or dielectric constant). You can edit these values after the layer is added, but sometimes it is more convenient to define what default values you want BoardSim to use for new layers — doing so may save your having to edit multiple, individual layers.

To set default stackup-layer values:

1. From the Options menu, choose Preferences.

2. Click the Default Stackup tab.

3. In the edit boxes, type the desired default values.

4. Click OK.

Now, when you create a new layer in the stackup editor, it will have the characteristics you just specified.

Deleting Layers

Deleting Layers

To delete a layer:

1. In the stackup editor, in the graphical area that displays the stackup, click once on the layer that you want to delete. Notice that a highlight box appears around the clicked-on layer.

2. Click the Delete Selected Layer button.

Cannot Delete Signal Layers with Routing

The stackup editor does not allow you to delete signal layers that have routed traces on them. Deleting a routed signal layer would invalidate simulation for many nets.

You can delete completely *unrouted* signal layers in your stackup; these could be either unrouted layers that came in from your PCB-layout tool or new signal layers that you added in the stackup editor.

Editing Layers

Editing Dielectric Layers

Dielectric-Layer Parameters

A dielectric layer has the following parameters, all of which can be edited:

thickness	required
dielectric constant	required
material name	optional

Thickness and Dielectric Constant

For BoardSim to simulate, a non-zero thickness and a non-zero dielectric constant are required. The thickness can be displayed in either English or

metric units; see Chapter 4, section “Setting Measurement Units” for details. For a table of dielectric constants for common board dielectrics, see Appendix E.

The thickness both displays in the graphical stackup area and is printed with the stackup. The dielectric constant is not displayed, but is printed.

Material Name

For a dielectric layer, you can enter a name for the dielectric's material. The material name is only for reference by a user running the stackup editor; it is not required and has no effect on simulation.

For a synthesized dielectric layer (one added by the Stackup Wizard), the layer name is BoardSim-created and the material name is set to a default.

To edit a dielectric layer:

1. In the stackup editor, in the graphical area that displays the stackup, click once on the layer that you want to edit; then click the Edit Selected Layer button.
   OR
   Double-click on the layer that you want to edit.

2. To change thickness or dielectric constant, type the new data in the appropriate edit boxes. To change material name, click the Advanced button, then type the data.

3. Click OK (twice if you edited the material name).

Editing Plane or Signal Layers

Plane- and Signal-Layer Parameters

A plane or signal layer has the following parameters:

base thickness	required
plating thickness	required (but can be 0.0)

layer name	required
bulk resistivity	required; *advanced parameter that you normally do not need to change*
temperature coefficient	required; *advanced parameter that you normally do not need to change*
passivation type	required; outer layers only; *advanced parameter that you normally do not need to change*
material name	optional

Base and Plating Thickness

For BoardSim to simulate, a non-zero base thickness is required for every signal or plane layer; the plating value can be any value, including zero. Base thickness refers to the thickness of the base metal; plating adds to the base thickness. Plating thickness is normally 0.0 for inner layers. Both parameters can be displayed in either English or metric units, and in length or weight units; see Chapter 4, section “Setting Measurement Units” for details.

The combination of thickness + plating displays in the graphical stackup area, and is printed with the stackup.

Layer Name

For a signal or plane layer, you can enter a layer name only if the layer does not contain any signal routing. If the layer *does* contain routing, you cannot edit the layer name. (This occurs because the layer name is recorded in the session (.BUD) file and must be the same in both the session and .HYP files. See Chapter 3 for details on .HYP files and Chapter 14 for details on session files.)

Note: *When creating stackups, avoid using layer names that contain any of the following characters:*

( ) { }

These characters are used as delimiters in the stackup section of BoardSim's .HYP and .BUD files. Names containing the delimiter characters may cause the files to be read incorrectly.

Bulk Resistivity

Every signal and plane layer is required to have a bulk resistivity for the layer's metal material. The resistivity is used when BoardSim calculates DC resistances for trace segments on the layer.

Bulk resistivity is considered by BoardSim to be an "advanced" parameter, i.e., one which you normally do not need to change. Signal and plane layers automatically default to the bulk resistivity of copper.

For more details on bulk resistivity and how it is used, see "Calculating DC Resistance" below.

Temperature Coefficient

Every signal and plane layer is required to have a resistivity temperature coefficient for the layer's metal material. The temperature coefficient is used in conjunction with the layer's resistivity when BoardSim calculates DC resistances for trace segments on the layer.

Temperature coefficient is considered by BoardSim to be an "advanced" parameter, i.e., one which you normally do not need to change. Signal and plane layers automatically default to the temperature coefficient of copper.

For more details on the temperature coefficient and how it is used, see "Calculating DC Resistance" below.

Passivation Type

Every *outer* signal and plane layer is required to have a passivation type. The passivation type tells BoardSim how trace widths vary depending on the method used to passivate your board's outer layers. Inner layers are not passivated and therefore have no passivation-type selection. Also, the passivation type is only used by BoardSim when fabrication compensation is enabled for impedance calculations; for more details, see "Fabrication Compensation" below in this chapter.

Passivation type is considered by BoardSim to be an “advanced” parameter, i.e., one which you normally do not need to change.

For more details on passivation type and how it is used, see “Fabrication Compensation” below.

Material Name

For a signal or plane layer, you can enter a name for the metal material. The material name is only for reference by a user running the stackup editor; it is not required and has no effect on simulation.

To edit a signal or plane layer:

1. In the stackup editor, in the graphical area that displays the stackup, click once on the layer that you want to edit; then click the Edit Selected Layer button.
   OR
   Double-click on the layer that you want to edit.

2. To change base thickness, plating thickness, or layer name, type the new data in the appropriate edit boxes. To change bulk resistivity, temperature coefficient, passivation type, or material name, click the Advanced button, then type the data (or choose the passivation type).

3. Click OK (twice if you edited an Advanced property).

Changing a Layer from One Type to Another

Planes with Signal Routing

If your board has a plane layer that contains some trace routing, BoardSim will incorrectly identify the layer as a signal layer, rather than a plane. This will result in the wrong impedances being calculated for the surrounding signal layers. For this reason, the stackup editor allows you to change a signal layer’s type to “plane.”

More generally, BoardSim supports changing almost any layer type into another. However, this feature exists mostly to support plane layers with

routing; many of the other combinations of change (e.g., dielectric to signal) do not make much sense and would rarely be used.

To change a layer from one type to another:

1. In the stackup editor, in the graphical area that displays the stackup, click once on the layer whose type you want to change; then click the Edit Selected Layer button.
   OR
   Double-click on the layer.

2. In the Set Layer Type area, click the radio button for the new type you want the layer to be.

3. Click OK.

Note: *The only layer-type change that BoardSim does not allow is from signal to dielectric, for signal layers that have routed traces. This change would effectively delete many routed traces from the board.*

About "Field Solver" Messages

When you finish editing a stackup (i.e., make changes and then close the stackup-editor dialog box), BoardSim briefly calls its field solver to characterize certain aspects of the new stackup. Often, you will see a progress dialog box labeled "HyperLynx" and "Running field solver" while this analysis is running.

The field solver is called regardless of whether or not you are licensed for BoardSim's Crosstalk option. If you are not licensed for Crosstalk analysis, then this (and at the end of the board-loading process) is the only time the field solver runs; it is not available during simulation or any other kind of analysis unless you own the Crosstalk option.

Stackup Colors

The colors in the stackup editor match the colors in the board viewer (see Chapter 7, section "The Board Viewer" for details on the viewer). A maroon

trace segment, for example, is on the layer which is shown in the stackup editor as maroon.

Note: *The only exception to the color matching between stackup editor and board viewer is if you have deliberately highlighted nets in the board with non-stackup-layer colors. See Chapter 7, section "Net Highlighting" for details on highlighting nets.*

Changing Layer Colors

You can change the color of any signal or plane layer.

To change a layer's color:

1. In the stackup editor, in the graphical area that displays the stackup, click once on the layer whose color you want to change; then click the Edit Selected Layer button.
   OR
   Double-click on the layer.

2. Click on the Layer Color button. (The area just above the button shows the layer's current color.)

3. Use the standard Windows Color dialog box to select a new color.

4. Click OK twice.

Dielectric layers have a fixed color which cannot be changed.

Measurement Units

The units used in the stackup editor to measure thicknesses and copper weights can be changed between English and metric (for the measurement system) and length and weight (for copper thickness).

Normally, you use the Preferences choice from the Options menu to change units. However, if you already have the stackup editor open, there is a quicker method than exiting the editor and using the menu.

To change measurement units from inside the stackup editor:

1. With the stackup editor open, click the Measurement Units button. The Units dialog box opens.

2. Make units selections as described in Chapter 4, section "Setting Measurement Units."

Total Board Thickness

The stackup editor displays the total thickness of your stackup (look below the graphical stackup area). As you edit the stackup, the total thickness changes.

Note: *Total thickness may affect the manufacturability of your board. For example, 62 mils is a standard thickness for many fabricators. Thick boards often increase drilling cost because fewer panels at-a-time can be drilled. On the other hand, an extra-thick board is often used for backplane applications because of the improved rigidity.*

Calculating Characteristic Impedances

When you simulate a net, BoardSim automatically calculates the characteristic impedance of every segment on the trace. To calculate the impedances, BoardSim uses your current stackup and the width of the trace segments.

Note: *A PCB trace normally consists of many individual "segments" which, taken together, make up the complete trace. BoardSim treats each of these segments individually as a separate transmission line. This means that if you have a trace which consists of a mixture segment widths, e.g., some of the segments on the trace are 8 mils wide and some are 6 mils wide, BoardSim will correctly account for the resulting impedance discontinuities and delay changes.*

Even though BoardSim calculates impedances automatically, you may want to explicitly know the characteristic impedances of various trace segments. For example, knowing the impedances might be helpful when you are designing a new stackup.

You can view trace impedances in the stackup editor. From the Edit menu, choose Stackup; then follow the steps in “Test Trace Width” below.

Test Trace Width

If you have an electrically valid stackup (no errors reported), the stackup editor shows you the characteristic impedance of the trace segments on each signal layer. The impedances appear next to each signal layer in the graphical area that displays the stackup. This assists you in creating or editing a stackup.

The stackup editor bases the impedance it displays on a “test” trace width that you can set for each stackup layer. As a convenience, when the .HYP file loads, BoardSim automatically sets each layer’s “test” width to the most-commonly used trace width on that layer.

Thus, the impedances in the editor should almost always correspond to the impedances you would naturally think of for each layer. For example, if you have a layer on which almost all traces are 8 mils wide, but there are a few 12-mil traces, by default the stackup editor will display the 8-mil impedance — the most logical choice, given that only one impedance at a time can be displayed.

However, you may want to change the test trace width, for example to see what the impedance of the less-common-width traces on a layer are, or to experiment to see how different impedances could be achieved with different widths.

To edit a layer’s test trace width:

1. In the stackup editor, in the graphical area that displays the stackup, click once on the layer whose width you want to set; then click the Edit Selected Layer button.
   OR
   Double-click on the layer that you want to edit.

2. Type a new value in the Test Trace Width edit box. Click OK.

The layer’s characteristic-impedance value in the graphical area updates immediately. If you print the stackup or copy it to the Windows Clipboard,

each layer's characteristic impedance and test-trace width display next to the layer.

If you have multiple trace widths on a single layer, you must enter each trace width separately to see all of the characteristic impedances.

It is important to realize that the Test Trace Width parameter does NOT affect the impedances calculated for any of the traces on your board during simulation or any other kind of analysis. The impedances used for simulation are based on the actual widths of the traces in your PCB layout. *The Test Trace Width only affects the impedance values displayed in the stackup editor's graphical area.* You can change the Test Trace Width to see what impedances various-width traces have on each layer in your stackup.

Fabrication Compensation

An advanced feature supported by BoardSim's impedance calculator is the ability to explicitly consider the impedance effects of how your board is fabricated. Specifically, BoardSim can compensate for:

- the effects of etched trace widths versus ideal trace widths
- the effects of metal added to outer layers to passivate the traces

*HyperLynx recommends that you **disable** the fabrication compensation features unless you know for certain that your PCB fabricator is **not** taking these factors into consideration.*

Hint: *Most good board houses automatically compensate for trace-etching effects by changing trace widths and dielectric thicknesses. If yours doesn't, a better solution than enabling this compensation in BoardSim may be to change board vendors!*

The default setting for fabrication compensation is disabled.

The following sections describe etch and passivation compensation, to help you decide whether you need these features enabled.

What is Trace-Etch Compensation?

When a board layer is etched, the resulting traces actually end up with a trapezoidal cross section. The top of the trace spends the most time in the etching solution and the bottom the least, so the top of the cross section is narrower than the bottom.

Thus, if you tell a board fabricator, "I want an 8-mil-wide trace," the fabricator will secretly say, "OK, but I'll need to start with a somewhat wider trace, knowing that it will etch to an effective 8-mils width." This compensation is normally supplied transparently by the fabricator so that you don't have to worry about the details of how trace etching occurs.

Only if your fabricator is not supplying this compensation for you should you enable BoardSim's compensation. If you tell BoardSim to compensate for you, the program will reduce the trace widths on your board by an appropriate amount to account for the effects of etching, i.e., BoardSim will assume that your trace widths are pre-etch, not post-etch.

What is Passivation Compensation?

Another fabrication process that can affect trace impedance is "passivation." If copper is exposed for prolonged periods to air, it will oxidize, which is not acceptable on PCBs. To prevent oxidation, the traces on the outer layers of a board are covered with another material, which shields the copper from exposure to air. If this passivation material is conductive, it will increase the traces' effective widths, affecting their impedances.

Passivation is another effect that a good board house will compensate automatically for, particularly if they are building controlled-impedance boards.

Enabling Fabrication Compensation

Enabling fabrication compensation gives you access to both trace-etch and passivation compensation. For details on controlling each type of compensation individually, see the sections below.

To enable fabrication compensation:

1. From the Options menu, choose Preferences. The Options dialog box opens.

2. In the Circuit Board Fabrication Compensation area (on the "General" tab), click on the Enable Compensation check box. Click OK.

Alternate method for enabling fabrication compensation, when the stackup editor is open:

1. From inside the stackup editor, in the PCB Fabrication Compensation area, click on the Enable Compensation check box.

Details of Trace-Etch Compensation

BoardSim's trace-etch compensation works by reducing the widths of the traces on your board. The amount of the reduction is based on a percentage of a trace's "clad," or base, thickness. BoardSim uses a different default percentage for outer versus inner layers, and allows you to change the percentages, if needed, to match a particular fabricator's manufacturing process.

To change the trace-etch-reduction percentages:

1. From the Options menu, choose Preferences. The Options dialog box opens.

2. In the Circuit Board Fabrication Compensation area (on the "General" tab), type new values in the Decrease Width By edit boxes. Click OK.

Details of Passivation Compensation

BoardSim's passivation compensation works on a layer-by-layer basis, and is available only for the outer layers in the current stackup. If you edit a stackup and, for example, move an outer layer to an inner position, passivation compensation no longer applies to that layer.

Note: *Passivation is only performed for a board's outer layers. The inner layers are automatically passivated by being "buried" in the board.*

Different passivation techniques affect trace impedances in different ways. The following sections describe how to specify a passivation type and briefly describe what each type means.

To specify a passivation type for an outer layer:

1. From the Edit menu, choose Stackup.
   OR
   Click the Edit PCB Stackup button on the toolbar. The stackup editor opens.

2. In the graphical area that displays the stackup, click once on the outer layer for which you want to specify passivation; then click the Edit Selected Layer button.
   OR
   Double-click on the outer layer that you want to edit.

3. Click the Advanced button.

4. Pull down the Copper Passivation combo box, and choose the passivation type that you want. (Remember, the passivation can only be set of fabrication compensation is enabled.)

5. Click OK twice.

SMOBC (Solder Mask Over Bare Copper)

In SMOBC passivation, the board is passivated with solder mask. SMOBC passivation does not affect trace thickness, because the passivation material is non-conductive. Therefore, choosing this style is the same as choosing “None.”

Hot-Air-Leveled Solder

In hot-air-leveled-solder passivation, the board is passivated by being passed through a solder bath. The solder is then leveled (flattened) with a stream of hot air.

Gold/Nickel and Flash Gold/Nickel

Both of these techniques passivate by plating outer-layer traces with a layer of nickel, then a layer of gold. The flash technique differs in that the gold layer is thinner, to save cost.

Tin-Nickel

The board is passivated with tin-nickel plating.

None

This selection says explicitly not to apply any passivation compensation.

Calculating DC Resistance

When you simulate a net, BoardSim automatically calculates the DC resistance of every segment on the trace. To calculate DC resistance, BoardSim uses the following parameters:

- trace width, thickness, and length
- bulk resistivity of trace's metal
- resistivity temperature coefficient of trace's metal
- simulation temperature

Trace width, thickness, and length are based on the actual segment-by-segment parameters in your board's layout. However, the remaining parameters (resistivity, temperature coefficient, and temperature) you can adjust, as described below.

Bulk Resistivity and Temperature Coefficient

In BoardSim, each metal layer's bulk resistivity and temperature coefficient default to the values for copper (resistivity = 1.724e-8 ohms/meter, temperature coefficient = 3.93e-3). This means that unless you are using a

metal other than copper (e.g., aluminum, for an MCM application), you do not need to change the resistivity or temperature-coefficient parameters.

To change a metal layer's bulk resistivity and temperature coefficient:

1. In the stackup editor, in the graphical area that displays the stackup, click once on the metal layer whose parameters you want to change; then click the Edit Selected Layer button.
   OR
   Double-click on the metal layer that you want to edit.

2. Click the Advanced button.

3. Type the new values in the Bulk Resistivity and Temperature Coefficient edit boxes.

4. Click OK twice.

How Resistivity, Temperature Coefficient, and Temperature are Used

The DC resistance of a piece of metal conductor varies with temperature. When BoardSim calculates the DC resistance of a trace, it uses the following equation to include the effects of temperature:

$$R = R_b \bullet (1 + T_c \bullet T)$$

where R_b is the bulk resistivity of the trace's metal; T_c the temperature coefficient; and T the temperature at which the simulation is being run.

Simulation Temperature

The temperature for a BoardSim simulation defaults to 20 degrees C.

To change the simulation temperature:

1. From the Options menu, choose Preferences.

2. In the Analysis Options area (on the "General" tab), in the Board Temperature edit box, type the new temperature. Click OK.

The board temperature affects only DC resistance calculations. It does not, for example, also affect IC-model parameters. Since DC resistance does not play a large role in most BoardSim simulations, you can generally leave the temperature at its default value without sacrificing any significant simulation accuracy. You should only change the temperature if you know for some reason that it will have a significant effect on your simulation.

Note: *Board temperature affects not only the simulator (oscilloscope), but also the Terminator Wizard and Board Wizard. For this reason, BoardSim refers to this feature as an "analysis" (not just "simulation") option. The same applies to inclusion of vias.*

Printing Stackups

You can print your board's stackup from BoardSim in order to document it. Some possible reasons to use BoardSim's stackup print-out are:

- if you change your stackup in BoardSim in order to achieve certain impedances, improve signal integrity, etc. and need to communicate the change to others in your engineering, CAD, or manufacturing departments
- if you need to document your desired stackup to your PCB fabricator

To print a stackup:

1. From the Edit menu, choose Stackup.
   OR
   Click the Edit PCB Stackup button on the toolbar.
2. In the stackup editor, click the Print button.
3. Check your printer setup. Portrait orientation is best for large stackups. Click OK to begin printing.

BoardSim supports color printers; stackups sent to a color printer are output in color.

Setting Up for Printing

You can set up printing-related defaults — e.g., printer choice, paper size, page orientation, etc. — once in BoardSim, then have them apply for the remainder of your work session, and for all types of printing (schematics, stackups, oscilloscope results, etc.).

To set up "persistent" printing defaults:

1. From the File menu, choose Print Setup. The Print Setup dialog box opens.

2. Change any parameters you wish, then click OK.

BoardSim now remembers the choices you've made, and will continue to use them.

Copying a Stackup to the Clipboard

You can copy your stackup to the Windows Clipboard in order to paste it into other Windows applications. The image sent to the Clipboard is formatted, and includes information such as the name of your board, the total board thickness, and so forth.

To copy stackups to the Windows Clipboard:

1. From the Edit menu, choose Stackup.
   OR
   Click the Edit PCB Stackup button on the toolbar.

2. In the stackup editor, click the Copy to Clip button.

BoardSim writes to the Clipboard in Windows Enhanced Metafile format. The size of the image may vary depending on which application you paste it into; resize as needed (metafiles are vectored and can be sized without damaging image quality).

Stackup Limitations

Multiple Dielectrics between Two Layers

BoardSim's stackup editor allows you to add multiple dielectric layers between metal layers. There is no problem if you set both layers to the same dielectric constant.

You can also set the dielectric layers to different dielectric constants. However, if you do, the accuracy with which BoardSim calculates characteristic impedances for trace segments will be diminished. ***Note:*** *The air-to-other-dielectric boundary for microstrip lines* ***is*** *handled with normal accuracy. The preceding caveat applies only to internal dielectric-to-dielectric boundaries. It also applies only to single, uncoupled traces segments; for coupled segments, if you own the Crosstalk option, BoardSim uses its field solver to calculate impedances, and the field solver handles mismatched dielectric constants without any loss of accuracy.*

Must Have at Least One Power Plane

BoardSim places several restrictions on plane layers in a stackup, among them that there must be at least one plane layer per board. This means that you cannot model double-sided boards (boards with only two signals layers, and no planes).

For details, see "BoardSim Assumptions about Plane Layers" above in this chapter.

Changing Trace Widths

The widths of the traces on your board play a major role (along with your board's stackup) in determining the impedance of your board's traces — and impedance, in turn, greatly affects signal quality and even radiated-emissions behavior. For this reason, you may sometimes want to experimentally change the widths of certain traces on your board to see how signal integrity is affected.

For example, suppose you have a 10-mil-wide clock net which is not incident-wave switched by the driver IC (i.e., the driver IC can't generate a large-enough initial voltage step into the trace to make all of the receivers on the net switch until several transmission-line reflections occur). If the clock trace's segments were higher impedance, the initial switching step would be larger, and you might achieve incident-wave switching. You wonder if a 6-mil trace width on the clock net would increase the impedance enough to solve the problem.

Or, suppose you have a net with marginal signal quality, and you notice that it transitions several times between a layer with 8-mil-wide trace segments and one with 6-mil segments. You wonder whether the impedance discontinuities caused by the differing widths are generating the signal problems — what if both layers had 6-mil-wide segments?

Note: *A PCB trace normally consists of many individual "segments" which, taken together, make up the complete trace. When simulating, BoardSim treats each of these segments individually as a separate transmission line. This means that if you have a trace which consists of a mixture segment widths, e.g., some of the segments on the trace are 8 mils wide and some are 6 mils wide, BoardSim will correctly account for the resulting impedance discontinuities and delay changes.*

How BoardSim Changes Trace Widths

BoardSim allows you to change trace widths directly, without having to go back to your PCB-layout tool. These changes are made to your current layout, and are experimental and temporary: when you exit BoardSim or close your board, the changes are discarded; the next time you load your board into BoardSim, the original layout, with its original trace widths, is restored.

To change the widths of trace segments on your board:

1. With your board loaded into BoardSim, from the Edit menu, choose Trace Widths. The Change Trace Widths dialog box opens.

2. In the Select Trace Segments To Change area, choose the net(s), stackup layer(s), and width(s) for which you want to change the widths. (See below for more details on making these selections.)

3. In the Width To Change To area, type the width to which you want to change.

4. Click the Change Widths button. The widths are altered immediately, and are shown in the board viewer.

5. Click Close, and resume analysis.

The changed widths are in effect until you make additional changes that override them, or until you close and re-load the board. You cannot restore your original widths except by re-loading your board.

Choosing Which Segments to Change

The Select Trace Segments To Change area in the Change Trace Widths dialog box gives you considerable flexibility in choosing which trace segments to alter. Each of the sub-selections below can be set independently of the other two selections:

Choosing Which Nets

In the Traces On These NETS area, you can choose to make width changes on only a selected net, or on all of the board's nets. If you are choosing a particular net, you can change the order in which the nets are listed in the Selected Net combo box by selecting the desired radio button in the Sort Nets By area.

Choosing Which Layers

In the AND On These LAYERS area, you can choose whether to change widths on only a selected stackup layer, or on all layers. The Selected Layer combo box lists all of the layers in the current board stackup.

Choosing a Range of Widths

In the AND With WIDTHS In This RANGE area, you can choose to limit the changes to only trace segments in a selected range of original widths, or to all segments regardless of width. If you choose to enter a range, you enter the minimum and maximum widths in the range.

Conditions are ANDed

Notice that the three selection criteria (nets, layers, and width range) are ANDed together. To eliminate one of the criteria, click the All radio button in that criterion's area. For example, to eliminate the width range, click the All Widths radio button.

Examples

Changing an Entire Single Trace, on All Its layers

To change the width of a single trace on all of the layers on which it's routed:

1. In the Traces On These NETS area, click on the Selected Net Only radio button, then choose the desired net using the Selected Net combo box.
2. In the AND On These LAYERS area, click on the All Layers radio button.
3. In the AND With WIDTHS In This RANGE area, click on the All Widths radio button.

Changing All Traces on a Single Stackup Layer

To change the widths of all the traces on a single stackup layer:

1. In the Traces On These NETS area, click on the All Nets radio button.
2. In the AND On These LAYERS area, click on the Selected Layer Only radio button, then choose the desired layer using the Selected Layer combo box.
3. In the AND With WIDTHS In This RANGE area, click on the All Widths radio button.

Changing All 10-Mil and 8-Mil Traces to 6 Mils Wide

To change all of the 10-mil and 8-mil traces on your board to 6 mils wide:

1. In the Traces On These NETS area, click on the All Nets radio button.

2. In the AND On These LAYERS area, click on the All Layers radio button.

3. In the AND With WIDTHS In This RANGE area, click on the Selected Range radio button, then choose the range by typing "8" into the Min Width box and "10" into the Max Width box.

4. In the Width To Change To area, type "6".

Changing the Top-Layer, 6-Mil Segments on Net CLK to 8 Mils Wide

To change the top-layer, 6-mil-wide segments on a net called "CLK" to 8 mils wide:

1. In the Traces On These NETS area, click on the Selected Net Only radio button, then choose CLK from the Selected Net combo box.

2. In the AND On These LAYERS area, click on the Selected Layer Only radio button, then choose layer "Top" from the Selected Layer combo box.

3. In the AND With WIDTHS In This Range area, click on the Selected Range radio button, then choose the range by typing "6" into the Min Width box and "6" again into the Max Width box.

4. In the Width To Change To area, type "8".

Restoring Original Widths

To restore the original trace widths used in your PCB layout, you must re-load your board's .HYP file. When you re-load, the width changes made in the previous session are discarded and the original widths from your .HYP file are restored.

Possible "Bad" Effects from Width Changes

Changing the widths of a board's traces usually does not affect the electrical validity of the traces, but this cannot always be guaranteed. Generally, narrowing traces (i.e., making them narrower) is usually safe; widening traces (if the board is densely routed) may cause electrical problems.

For example, if you widen a trace too much such that it touches another trace, BoardSim may connect the two traces together (because you've "shorted" the widened trace to the other trace).

***Note**: In a given simulation, BoardSim only "looks" at the net you've chosen for simulation, plus any associated nets (see Chapter 7, section "What are Associated Nets?" for an explanation of "associated nets"). Therefore, it's not as dangerous to widen traces as it might seem. Problems only arise if the widened trace touches another segment on the same net or a segment on an associated net.*

In rare cases, narrowing a trace may cause electrical problems. This could occur, for example, if a trace connects to a pad marginally, at the edge of the trace only. Narrowing the trace could cause the trace-to-pad connection to be opened.

Width Changes Not Reported in Design Change Summary

Trace-width changes you make are not reported in BoardSim's Design Change Summary. Also, although changed widths are used when the Board Wizard analyzes your PCB, the changes are not summarized in the design-change sections of the board report.

Therefore, if you want to keep a record of the traces and layers on which you've changed widths, you must do so manually.

Chapter 6: Setting Power-Supply Nets

Summary

This chapter describes:

- why identifying power-supply nets is important to BoardSim
- how BoardSim identifies power-supply nets
- how to edit the power-supplies list

Why Power-Supply Nets Matter

In order to simulate properly, BoardSim must know the difference between signal nets and power-supply nets. “Signal nets” are ordinary digital nets that carry switching signals. “Power-supply nets” are nets that are tied to a non-switching, DC voltage.

The difference is important because BoardSim propagates signals along signal nets, but not along power-supply nets. Power supplies are treated as fixed DC sources. Also, ICs can be only be run off of nets identified as power supplies.

How BoardSim Identifies Power-Supply Nets

BoardSim attempts to automatically identify as many power-supply nets as possible. On some boards, the automatic identification will work perfectly; on others, you may need to make corrections to the list of power-supply nets

yourself, manually, with the power-supply editor. (See “Editing Power-Supply Nets” below in this chapter for details.)

Methods for Identifying Power-Supply Nets

When BoardSim loads your board, it attempts to identify power-supply nets and to infer to what voltage each supply net is tied. This identification occurs in several ways:

- by looking for nets with common power-supply names (e.g., GND or VCC)
- by looking for nets that are connected to multiple capacitors
- by looking for nets with very large numbers of metal segments

The following sections describe in greater detail the methods used by BoardSim to identify power-supply nets.

Name Matching

The name-matching method attempts to identify power-supply nets by their names. The identification process includes guessing from the name to what voltage each net is likely attached.

“Automatic” Net Names

Some power-supply net names are recognized and assigned an “automatic” voltage.

VCC

The following net names are automatically interpreted as being a 5-V power supply:

PWR	VCC
POWER	VDD

Note: *The list above includes all of the automatically assigned net names as of this manual's printing. Check the online help for possible additions.*

BoardSim automatically adds any of these to its list of power-supply nets, and assigns a voltage of 5V. The name-matching is case-insensitive, e.g., "VCC", "vcc", and "Vcc" all match.

If the voltage is wrong (e.g., should be 3.3V), it can easily be changed in the power-supply editor. Also, if any of the nets is actually not a power-supply net, it can be removed from the list in the editor. (See "Editing Power-Supply Nets" below in this chapter for details.)

GND

Similarly, BoardSim interprets the following as being a 0.0-V power supply:

GND	GROUND
GRND	VSS

Note: *The lists above include the most-important of the automatically assigned names. BoardSim actually recognizes a larger, growing list of power-supply-net names that HyperLynx has seen frequently on customer boards.*

"Inferred" Net Names

Some power-supply net names are recognized and assigned a voltage that is based on part of the name.

Net names with the following forms are interpreted as being power-supply nets. The assigned voltage is the number (i.e., <number>) found in the net name:

+<number>V	or	+<number>	or	V+<number>
-<number>V	or	-<number>	or	V-<number>
<number>V	or	<number>		

The “V” can also be lower-case.

For example, each of the following is considered a power-supply net:

+12V, v+12, -12, 12V, 12

If the voltage is wrong or if any of these is not a power-supply net, the error can be fixed in the power-supply editor. (See “Editing Power-Supply Nets” below in this chapter for details.)

BoardSim places one additional requirement on nets whose names are matched by “inference”: that the net must also have at least one capacitor on it. This occasionally prevents an entire digital bus with supply-like net names from being mistaken as a collection of power supplies, since digital nets rarely have capacitors connected directly to them.

Counting Capacitors

The capacitor-counting method takes advantage of the fact that most power-supply nets are connected to a large number of decoupling capacitors. By default, BoardSim considers any net with three or more capacitors connected to be a power-supply net. If this identification is ever wrong, the misidentified net can be removed from the power-supplies list using the power-supply editor; see “Editing Power-Supply Nets” below in this chapter for details.

You can modify the power-supply identification threshold to any number of capacitors you want. For details on changing the threshold, see Chapter 4, section “Helping BoardSim Recognize Power-Supply Nets.”

However, the capacitor-based algorithm has proven successful across a broad spectrum of customer designs, and HyperLynx recommends not changing the threshold unless you know specifically that it is causing a problem with a particular board.

Note: *The capacitor-counting method has the advantage of finding not only power-supply nets (which typically have large numbers of decoupling capacitors connected), but also analog nets, which, like power supplies, should not be simulated as digital nets in BoardSim.*

Counting Metal Segments

The metal-segment-counting method relies on the fact that nets with a very large number of metal segments (> 20,000) are almost always power-supply nets. By default, BoardSim considers any net with 20,000 or more segments connected to be a power-supply net. If this identification is ever wrong, the misidentified net can be removed from the power-supplies list using the power-supply editor; see "Editing Power-Supply Nets" below in this chapter for details.

You can modify the power-supply identification threshold to any number of metal segments you want. For details on changing the threshold, see Chapter 4, "Helping BoardSim Recognize Power-Supply Nets."

However, HyperLynx recommends not changing the threshold unless you know specifically that it is causing a problem with a particular board.

Undetected Power-Supply Nets

BoardSim may fail to identify some of the power-supply nets on your board. When this happens, use the power-supply editor to add the undetected nets to the power-supplies list. (See "Editing Power-Supply Nets" below for details.)

Undetected power-supply nets can lead to some nets looking complicated and huge in the BoardSim's board viewer. This occurs because BoardSim displays not only the chosen net, but also all non-power-supply nets connected to the chosen net through passive components (e.g., resistors and capacitors). (The

connected nets are called “associated nets”; see Chapter 7, section “What are Associated Nets?” for details.)

Also, when you choose IC models for simulation, you can “run” the ICs only off of voltages in the power-supplies list. (See Chapter 8, section “Setting the Vcc or Vss Pin” for details.)

If a major power-supply net is undetected, nets in the board viewer will be particularly “strange” looking. Before proceeding (especially before simulating or running the Board Wizard), use the power-supply editor to add the undetected supply nets to the power-supplies list. (See the next section for details.) In fact, it’s a good idea to always review the power-supplies list for any board the first time you load it into BoardSim, to ensure that no supplies are missing from the list. Failure to do may result in very slow analysis results, as BoardSim attempts to simultaneously analyze huge sets of nets that are erroneously tied together.

Editing Power-Supply Nets

The power-supply editor allows you to:

- change which nets are listed as power-supply nets
- change the voltages on power-supply nets

Opening the Power-Supply Editor:

To open the power-supply editor:

1. From the Edit menu, choose Power Supplies.

Changing a Power-Supply Voltage

To change a power-supply voltage:

1. In the power-supply editor, in the Power Supply Nets list box, highlight the net whose voltage you want to change.

2. Type the new voltage into the New Voltage edit box.

3. Click OK.

Power-supply voltages can be positive or negative. Changes to supply voltages take effect immediately (i.e., as soon as you close the editing box).

Adding Power-Supply Nets

To add a power-supply net:

1. In the power-supply editor, in the All Signal and Supply Nets list box, highlight the net you want to add to the power-supplies list.

2. Click the Add Net button.

3. In the Power Supply Nets list box, highlight the new net.

4. Type the net's supply voltage into the New Voltage box.

5. Click OK.

The new power-supply net appears in the Power Supply Nets list box. If you attempt to add a net that already appears in the power-supplies list, BoardSim gives an error.

Removing Power-Supply Nets

To remove a power-supply net:

1. In the power-supply editor, in the Power Supply Nets list box, highlight the net you want to remove from the power-supplies list.

2. Click the Remove Net button.

The removed net disappears from the Power Supply Nets list box and will no longer be considered a DC voltage when BoardSim simulates.

Note: *Removing a net as power supply does not remove it from BoardSim's database. It only marks the net as "no longer a DC voltage."*

Sorting Power-Supply Nets

For convenience, BoardSim gives you several ways to sort the nets in the All Signal and Supply Nets list box.

The default sorting is By Width. Since power-supply nets are often the widest nets on the board, a width sort usually brings them to the top of the list. With By Width selected, BoardSim presents nets from widest (at the top) to narrowest (at the bottom).

To change the sorting method:

1. In the power-supply editor, click the appropriate radio button in the Sort Nets By area.

Other choices are By Name and By Length. Net lengths are calculated by summing the lengths of all the segments on the net, regardless of how they are connected. **However, net lengths do NOT include the lengths of other "associated" nets to which the named net is connected by series components, e.g., a series resistor.**

Effects on the Board Viewer

After you edit power-supply nets, you may see changes in how the board viewer displays the current net. This occurs because the net's list of associated nets changes based on the new power-supplies list. (See Chapter 7 for details on the board viewer and associated nets.) On very large boards, when you close the Set Power Supply Voltages and Nets dialog box, you may see a pause while BoardSim re-finds associated nets.

Chapter 7: Choosing and Viewing Nets

Summary

This chapter describes:

- the concept of “associated nets”
- how to choose a net for simulation and viewing
- how to view associated nets
- how to highlight a net (for viewing only; not for simulation)
- how BoardSim’s board viewer works
- limitations on simulating associated nets

What are Associated Nets?

When BoardSim simulates a net, it must consider not only the net itself, but also any other nets connected to it through passive components or differential-IC pin pairs. (“Passive components” are resistors, capacitors, inductors, or ferrite beads. See Chapter 4, section “What is a Reference-Designator Mapping?” for details on component types.)

For example, if you ask BoardSim to simulate Net1, and BoardSim finds that Net1 is connected through a series resistor to Net2, then BoardSim needs to simulate both nets together. (If there is a driver IC on Net1, then probably all of the receiver ICs are on Net2.)

On the other hand, if Net2 is a power-supply net, then BoardSim does not need to simulate Net2 along with Net1. In this case, Net2 is a DC voltage, not a continuation of Net1.

Definitions

An **associated net** is a non-power-supply net connected through a passive component to the net being simulated.

A **passive component** is a resistor, capacitor, inductor, or ferrite bead. An IC is not a passive component.

The Board Viewer

When BoardSim first opens, it presents a main window which is mostly blank except for a small set of menus. After you load the .HYP file for your board, BoardSim draws a picture of your board's outline, filled with component outlines. (See Chapter 4, section "Loading Your Board into BoardSim" for details on loading boards.)

The area in which your board appears is called the board viewer. It allows you to view a number of aspects of your board:

- board outline
- component outlines
- component reference designators

...plus, once you have loaded a net to simulate:

- complete nets (and their associated nets), one-at-a-time, including:
  - trace segments
  - vias
  - pads

For details on loading a net to simulate, see “Choosing Nets” below. For details on viewing boards, see “Viewing a Board.”

Reviewing your Layout with BoardSim’s Viewer

BoardSim’s board viewer (even though it was designed primarily to use in conjunction with the product’s simulation features) is a fairly powerful tool for reviewing your PCB layout. It has several advantages over Gerber viewers and your PCB layout tool’s viewer:

- it knows about nets electrically, and displays not only selected nets but also their associated nets (see “What are Associated Nets?” for details); thus, for example, if you select a series-terminated clock net, BoardSim will show you the nets on both sides of the series resistor
- it is simple to use (probably simpler than your PCB-layout tool’s viewer, if you’re not intimately familiar with the layout tool)
- if you are licensed for BoardSim’s Crosstalk option, the viewer is a powerful way to see graphically which nets are coupled to other nets (see the Crosstalk User’s Guide for more details)

Why review your PCB layout? There are many reasons, even when thinking strictly from a high-speed-design standpoint. For example, if you added terminators to your design, where were they actually placed? — very close to the component they terminate, or an inch away? Or if you specified a clock net as “critical,” how was it actually routed? In a nice, clean, short daisy chain, or in an unnecessarily long chain with half-inch stubs to every receiver IC?

Some BoardSim users actually review every net on their board each time they get a layout back from their CAD group or PCB service bureau.

To perform a net-by-net review of your PCB layout:

1. Load the PCB’s .HYP file into BoardSim.
2. Review (and modify, if needed) the power-supplies list (see Chapter 6, section “Editing Power-Supply Nets” for details).
3. From the Select menu, choose Net by Name.

4. Position the Select Net by Name dialog box in a corner of the screen. If desired, use the Sort Nets By radio buttons to change the sorting criterion for the nets in the dialog-box list.

5. Click once to highlight the first net in the list.

6. Review the net's (and its associated nets') routing. To move to the next net, click the down arrow key on the keyboard.

7. Repeat step 6 until the review is complete.

Choosing Nets

Before you can run a simulation in BoardSim, you must choose a net to simulate. Choosing a net also allows you to view the net in the board viewer.

Note: *You must have a board loaded before you can choose a net. See Chapter 4, section "Loading Your Board into BoardSim" for details.*

To choose a net to simulate and view:

1. From the Select menu, choose either Net by Name or Net by Reference Designator.
   OR
   Click the Select a Net by Name button on the toolbar.

Which choice you make depends on how you want to choose the net. See the following sections for details.

Choosing nets By Name makes sense if you know the names of the nets you want to simulate. Choosing By Reference Designator makes sense if you have nets on the board whose names you are not sure of; you can choose these nets by the components and pins to which they're attached.

Note: *Unknown names typically occur when you do not name a net in your schematic, and the net then gets a computer-generated name during netlisting.*

Choosing Nets by Name

To choose a net By Name:

1. From the Select menu, choose Net by Name.
   OR
   Click the Select a Net by Name button on the toolbar.

2. In the Net Name list box, scroll until you see the name of the net you want to simulate.

3. Highlight the net name, then click OK.
   OR
   Double-click on the net name.

The dialog box closes, and the chosen net appears in the board viewer.

If you're not sure of the net you want to select, you can also click once to highlight a net in the list. The net is selected and displayed immediately, with the dialog box still open. If you then decide it's a different net that you really want, you can select that net instead, immediately — no need to re-open the dialog box.

Sorting Nets

For convenience, BoardSim gives you several ways to sort the nets in the Select Net by Name dialog box.

The default sorting is By Name. By Name makes sense if you know the names of the nets you want to simulate and view.

To change the sorting method:

1. Click the appropriate radio button in the Sort Nets By area.

Other choices are By Length and By Width.

Note: *By Length makes sense if you are not sure which nets on your board are most likely to have transmission-line problems: the longest nets are the likeliest culprits. Note, however, that the lengths are for the named net only, so the "short side" of net pairs that are series-terminated may not appear high in the*

list.
But whether it is worth simulating a net also depends on whether or not the net is timing-critical or edge-sensitive, and whether the driver IC on the net has a fast slew rate or not. Nets that are not timing critical, or do not drive edge-sensitive inputs, or are driven by slow drivers, often do not have transmission-line problems.

Net Lengths Do NOT Include Associated Nets

The net lengths shown in the Select Net by Name dialog box are calculated by summing the lengths of all the segments on the net, regardless of how they are connected. **The lengths do NOT include the lengths of associated nets.** For example, if you have a clock net which consists of a short net named "CLK", a series terminating resistor, and a "continuation" net named "CLK_T," the length shown for net CLK will be only CLK's short length; the length of net CLK_T will not be added.

Net Lengths Reported are "Pre-Cleanup" Unless You Enable "Net Cleaning During Loading" Option

Many PCB-layout programs make little or no attempt to "clean up" redundant or overlapping trace segments on a board. (Such redundancy is particularly common in designs that have been routed at least partially by hand.) Redundancy makes no difference when a Gerber file is output, but is not acceptable to simulation tools like BoardSim that assign electrical characteristics to all metal structures on a net.

Accordingly, BoardSim "cleans" all nets before you analyze them, eliminating redundant metal and combining overlapping structures when possible into fewer, large structures. This guarantees accurate simulation results. The cleaning process actually occurs when a net is first selected.

The advantage to cleaning nets only when they are selected is that the task is distributed: the other option is to clean all nets at board-loading time, but this effort increases (usually by about a factor of two) the time required to load a board.

However, there is one disadvantage to cleaning nets only when selected. The net lengths displayed in the Select Net by Name dialog box are calculated at

board-loading time. If net cleaning occurs only later when nets are selected, and if some nets contain large amounts of redundant metal, then the lengths reported in the dialog box may be too long.

To avoid this problem, you can instruct BoardSim clean all nets at board-loading time. See Chapter 4, section “Net Cleaning” for details.

Viewing Net Lengths Including Associated Nets and “Cleanup”

If you do NOT enable the “net cleaning during loading option” (see above for details), but you want to view “post-cleanup” net lengths…

OR

If you want to view net lengths *including* the lengths of associated nets…

…use BoardSim’s Net Statistics feature. The Net Statistics dialog box cleans up redundant segments on a net (if they exist) and sums into the reported net length the lengths of all associated nets.

To view “cleaned-up” net lengths, including the lengths of associated nets:

1. Select the net whose length you want to view.
2. From the Reports menu, choose Net Statistics. The Statistics for Selected Net dialog box opens.
3. Look at the Total Length of All Segments value.

Choosing Nets by Reference Designator

To choose a net By Reference Designator:

1. From the Select menu, choose Net by Reference Designator.
2. In the Reference Designator list box, scroll until you see the reference designator of the component to which the net you want to simulate is connected. Highlight the reference designator.
3. In the Pin Name list box, scroll until you see the name of the pin to which the net is connected.

4. Highlight the pin name, then click OK.
   OR
   Double-click on the pin name.

The dialog box closes, and the chosen net appears in the board viewer.

Net and Pin Information

The Select by Reference Designator dialog box displays information about the net connected to the currently highlighted component and pin.

Net Information

In the Net Information area, BoardSim displays data about the corresponding net, including:

- net name
- net length (excluding associated nets)
- maximum width of any segment on the net

Component Information

In the Device Info field, BoardSim displays:

- the name of the chosen component, if the component is an IC or a ferrite bead
- the value of the component, if the component is a passive component (resistor, capacitor, inductor)

Current-Net Status

The name of the currently chosen net is displayed in the status bar at the bottom of BoardSim.

Cannot Choose Power-Supply Nets

Whether you are choosing nets By Name or By Reference Designator, you cannot choose a power-supply net for simulation. (See Chapter 6 for details on power-supply nets.)

Power-supply nets are marked in the net-selection dialog boxes with a green icon that looks like the symbol for an ideal voltage source. In the By Name dialog box, the icons appear in the main list box. In the By Reference Designator dialog box, they appear in the Pin Name list box.

If a net is erroneously marked as a power-supply net, you can remove it from the power-supplies list. See Chapter 6, section “Removing a Power-Supply Net” for details.

Displaying Associated Nets

When BoardSim displays a net in the board viewer, by default it displays not only the chosen net, but also all of the net’s associated nets. (See “What are Associated Nets?” above in this chapter for details on what an associated net is.)

Displaying a net *and* its associated nets allows you to view the complete simulation problem, i.e., all of the trace segments, vias, pads, and components involved in the simulation. You can place oscilloscope probes on any of the displayed nets, not just the chosen net. (See Chapter 12 for details on running simulations and the oscilloscope.)

BoardSim displays the same nets regardless of which net in a group of associated nets you choose to simulate. For example, if Net1 and Net2 are tied together through a series resistor, the board viewer will show both Net1 and Net2 regardless of whether you choose to simulate Net1 or Net2.

Turning Off Associated-Net Viewing

If for some reason you do not want to view associated nets, you can instruct the board viewer not to display them.

To turn off associated-nets viewing:

1. From the View menu, choose Options.
2. Click on the Associated Nets check box to disable it.
3. Click OK.

If the net currently displayed in the board viewer has associated nets, they disappear.

Note: *If nets look "too complicated" or seem to be connected to other nets they should not be, you probably have undetected power-supply nets. See Chapter 6, section "Undetected Power-Supply Nets" for details on correcting the problem. Turning off associated-net viewing only removes the extraneous nets from the board viewer, not from the simulation database.*

Net Highlighting

When you select a net in BoardSim (see "Choosing Nets" above for details), the program assumes you want to both view the net and perform analysis on it (i.e., see the net AND run interactive simulation, Terminator Wizard, Net Statistics, etc.). But what if you simply want to see a net in the board viewer, without selecting it for analysis; or want to view multiple nets simultaneously? For these uses, BoardSim has a "net highlight" feature.

When you highlight a net, it becomes visible (in a color you choose) in the board viewer, but it is NOT the selected net, i.e., if you run a simulation (or other analysis), the simulation will be for the selected net, not the highlighted net. Also, you can highlight more than one net at a time, in case, for example, you want to see several nets simultaneously and how they are laid out relative to each other.

Highlighting a Net

Each net can be highlighted either in a color you choose (each net can have its own color, if desired), or in the standard layer colors (same as when a net is selected for analysis).

To highlight a net in the board viewer:

1. From the View menu, choose Highlight Net.

2. In the dialog box, find the net you want to highlight.

3. Click on the net, then click Apply.
   OR
   Double-click on the net.
   Another dialog box opens.

4. Choose a highlighting style (user color or layer colors), as follows:
   If you want to highlight the entire net in a color of your choice (independent of the stackup layer colors), in the Colorize Net By area, click the User Color radio button.
   OR
   If you want to highlight the net using the standard method of displaying each segment in its stackup layer color, in the Colorize Net By area, click the Layer Colors radio button.

   If you do not want to also highlight the net's associated nets, click on the Include Associated Nets check box to disable it (normally, you would leave associated nets enabled).

5. *If you selected the User Color radio button in step #4,* then choose a highlighting color by clicking once on one of the color "chips" in the dialog box, then clicking OK.
   OR
   If you selected the Layer Colors radio button in step #4, click OK. (If you're using layer colors, there is no need to select a highlighting color.)

   The color-selection dialog box closes and the net is highlighted in the board viewer. An icon (see picture in section "Highlighting Icon in Net Dialog

Boxes" below) appears to the left of the net name in the Highlight Net dialog box, to indicate that the net is now highlighted.

You can repeat the net-selection process for as many additional nets as you want.

Note that when other nets are highlighted, the selected net (the one you choose for analysis) also remains visible.

If you highlight the selected net, it displays with its highlighting color. If you remove highlighting from the selected net (see "Removing Highlighting" below for details), it returns to displaying as normal, in its layer colors.

Custom Highlight Colors

You are not limited to the colors presented as "chips" in the color-selection dialog box; you can also construct your own custom color, if desired. To do so, in step #5 above, click the Custom Color button instead of selecting one of the chips.

How Highlighted Nets Display in the Board Viewer

When a net is highlighted in the board viewer, it displays in the color scheme you have chosen (a selected color or layer colors), ***and*** has long white dashes overlaid on it. The dashes allow you to distinguish between the selected net and highlighted nets when both are displayed with layer colors.

If you are licensed for BoardSim Crosstalk and have crosstalk analysis enabled, you will also see nets displayed in layer colors, with short white dashes along their lengths. These are aggressor nets, not highlighted nets. Remember: long white dashes for highlighted nets, short dashes for aggressor nets (see the Crosstalk User's Guide for details on aggressor nets and coupling).

If you highlight a net that is also an aggressor net (applies to BoardSim Crosstalk only), the net appears with long white dashes, i.e., the highlighting takes precedence.

Highlighting Icon in Net Dialog Boxes

When a net is highlighted, it is marked by a special icon in both the Select by Name and the Highlight Net dialog boxes, to help you remember which nets you've highlighted. The icon appears next to the net name, just to the left of the numerical data that shows net length or width; it looks like a "rainbow" of colors:

Removing Highlighting

Usually, after you've highlighted one or more nets and looked at them in the board viewer, you'll want to turn off all of the highlighting and resume using BoardSim in a mode where it displays only the selected net (and its associated nets). (Having a large number of nets permanently highlighted can get confusing after a while.) You can remove highlighting from nets one-at-a-time, or for each net individually.

To remove highlighting from all currently highlighted nets:

1. From the View menu, choose Highlight Net.
2. In the Highlight Net dialog box, click the Remove All button.

— OR —

1. From the View menu, choose Options.
2. Click the Remove Highlights button.

All currently highlighted nets disappear.

To remove highlighting from a particular net:

1. From the View menu, choose Highlight Net.
2. In the dialog box, find the net whose highlight you want to remove.

3. Click on the net, then click Apply.
   OR
   Double-click on the net.
   Another dialog box opens.

4. Click the Remove Net Highlight button. The dialog box closes and highlighting is removed from the specified net.

When highlighting is removed from a net, its “highlighted” icon disappears from the Select by Name and the Highlight Net dialog boxes.

Using Highlighting to See Power-Supply Nets

In BoardSim, power-supply nets cannot be selected for simulation (because analysis of power-supply nets is not supported). Therefore, the only way to see power-supply nets in the board viewer is to highlight them. For details on how to highlight a net, see the preceding sections.

If you highlight a power-supply net which is implemented with a plane layer in the stackup (or a copper pour) — e.g., “VCC” or “Ground” — the board viewer will show you only vias, pads, and any residual segments attached to the plane. The plane itself (or poured copper) does not display.

Viewing a Board

How BoardSim Displays a Board

As soon as you load a board’s .HYP file, the board appears in the board viewer. The viewer’s zoom level (i.e., “magnification”) is automatically adjusted so that the entire board appears in the viewer.

X-Y Position Readout

As you move the mouse in the board viewer, a status-bar area in the lower-right corner of BoardSim shows you a real-time readout of the mouse’s X-Y position. The origin (i.e., position X=0,Y=0) of the board’s coordinate system matches whatever was set for your board in the PCB-layout tool.

Which units are used for the readout depends on how you have set BoardSim's measurement units (English or metric). See Chapter 4, section "Setting Measurement Units" for details.

Board Outline

The board viewer attempts to display the outline of your board. The .HYP file includes an optional keyword BOARD which is followed by a detailed description of the line segments making up the board's outline. An outline can include both linear and curved segments.

If No Board Outline is Supplied

Not all .HYP-file translators provide a board outline. If the data is missing from the .HYP file, BoardSim will create a rectangular outline big enough to encompass all of the components on your board.

Component Outlines (Silk Screen)

BoardSim displays the outlines of each of the components on your board. For simplicity, BoardSim does not read silk-screen information from your PCB-layout tool; rather, component outlines are constructed for each component in the following manner:

1. As the .HYP file is loaded, build a list of the pins on each component that are connected to something on the board.

2. When loading is complete, draw a perimeter around the pins that were found for each component.

3. Use the perimeter as each component's outline.

Note: *BoardSim uses a "smart" algorithm to construct the component outlines. This algorithm increases the odds that the drawn outline matches the real one in situations where some component pins are unconnected and so not reported in the .HYP file. For example, if for a DIP package several pins are missing on one side only, BoardSim will still draw a rectangular outline.*

How to Tell which Side Components are On

Component outlines are drawn with a dashed line. Components on one side of the board are drawn in black; on the other side, in gray. Which side (top or bottom) is which color depends on the .HYP-file translator. If you flip the board (see "Orienting a Board" below in this chapter for details), the component outlines change sides on the board, and therefore change colors.

Reference Designators

By default, the board viewer labels each component outline with the component's reference designator. This helps you to identify components.

The reference-designator labels remain the same size on the screen regardless of how far in or out you zoom the board viewer.

(See "Zooming" below in this chapter for details on zooming in the viewer.)

Turning Reference-Designator Viewing On/Off

On very dense boards, or when a board is viewed "at a distance" (zoomed far out), the reference-designator labels may crowd together and overlap each other. In this case, it is convenient to instruct the board viewer not to display the reference designators. BoardSim allows you to enable/disable reference-designator viewing on a per-component-type basis (see Chapter 4, section "What is a Reference-Designator Mapping?" for details on component types.)

To turn off reference-designator viewing:

1. From the View menu, choose Options.
2. In the Show Reference Designators For area, click off the check box(es) for the component types whose reference designators you do not want to view.
3. Click OK.

The disabled reference designators disappear from the board viewer immediately.

Nets

Viewable Net Elements

The board viewer displays the following elements of a net:

- trace segments
- vias
- pads (through-hole and SMD)
- drill holes
- pins
- pin numbers (if the viewer is zoomed in sufficiently far)

Several of these are described in more detail below.

How to Correlate Net Elements and Stackup Layer

For the selected net and its associated nets, all of the constituent metal (e.g., trace segments or pads) on a net is shown in the board viewer *color-coded*. The colors correspond to the layer colors shown in the stackup editor (see Chapter 5 for details on the stackup editor.) (For highlighted nets — see section “Net Highlighting” above for details — you have a choice of displaying in layer colors or a user-selected color.)

To check which layer a trace segment or pad is on (unless the net is highlighted in a non-layer, user-selected color):

1. In the board viewer, note the color of the segment or pad.
2. From the Edit menu, choose Stackup.
   OR
   Click the Edit PCB Stackup button on the toolbar.
3. In the graphical area that displays the stackup, look for the noted color; the segment or pad is on the layer that has the matching color.

Pads

The .HYP file supports four pad shapes:

- round
- rectangular
- oval
- oblong

Pads are drawn with their proper shapes and actual sizes. An “oblong” pad is rectangular with rounded corners.

If you “flip” the board (see “Orienting a Board” below in this chapter for details), vias will change color because you are viewing the via pads on the opposite side of the board when the board is flipped.

Turning Pad Viewing On/Off

Occasionally, it may be convenient to instruct the board viewer not to display pads. BoardSim allows you to enable/disable pad viewing.

To turn off pad viewing:

1. From the View menu, choose Options.
2. Click on the Show Pads check box to disable it.
3. Click OK.

All pads disappear from the board viewer immediately.

Drill Holes

The .HYP file allows each via to have an independently sized drill hole. Drill holes are displayed as filled-in black circles.

Pins

The .HYP file supports a construct called “pin.” A pin links a component to a physical location. For example, consider this PIN record in a .HYP file:

(PIN X=2.5500 Y=1.9000 R=C2.2)

This record indicates that component C2 has a pin connected to the board at location 2.55,1.90.

The board viewer shows pins as small black dots. Pins remain the same size on the screen regardless of how far in or out you zoom the board viewer.

To be connected to a net, a component pin also requires a pad on the board. In rare cases, a .HYP-file translator may fail to provide a pad; BoardSim may still be able to establish connectivity with “pad synthesis.” (See Chapter 13, section “Pad Synthesis” for details.)

Pin Numbers

Pin numbers for components are displayed *if* the board viewer is zoomed in sufficiently far. (See “Zooming” below in this chapter for details on zooming in.)

To view a component’s pin numbers, zoom in toward the component’s outline until the component appears fairly large. Pin numbers are displayed only for components that are connected to the currently selected net and its associated nets, and aggressor nets and their associated nets (aggressor nets available only in BoardSim Crosstalk). All of the pin numbers on each connected component are displayed.

Pin-number viewing is disabled in view-all-nets mode.

Associated Nets

When BoardSim displays a net in the board viewer, by default it displays not only the chosen net, but also all of the net’s associated nets. See “What are Associated Nets?” above in this chapter for details.

Net-Name Readout

As you move the mouse in the board viewer, a status-bar area in the bottom middle of the BoardSim window identifies the name(s) of net(s) that the mouse is currently positioned over. This gives you an easy way to identify nets in the board viewer.

To identify a net in the board viewer:

1. Point to the net with the mouse. The net's name is displayed in the status bar (in the bottom middle of the BoardSim window).

If Multiple Nets per Pointed-To Location

On a complex board with many stackup layers, it is likely when you point the mouse at a given place in the board viewer that there are several nets at or below that position in the stackup. In cases where the mouse is pointing to multiple nets (on different layers) simultaneously, the net-name readout lists *all* of the pointed-to nets, in order from the top to bottom stackup layer.

Note: *The net-readout feature can only display a limited number of net names at a time. If the number of pointed-to nets down through the stackup exceeds this limit, then only the first N names are displayed, starting from the top of the stackup and continuing down until N names are found. As of this writing, N was set to 5.*

Orienting a Board

The board viewer allows you to "flip" your board around two axes.

Default Orientation

When your board first appears in the board viewer, BoardSim guesses at the orientation from which you want to view the board. You can quickly tell from looking at the board outline and the component positions whether or not you want a different orientation.

Flipping a Board

To flip a board's orientation:

1. From the View menu, choose Flip Board.

2. To flip around a vertical axis, choose Right;
   to flip around a horizontal axis, choose Down.

The board changes appearance immediately:

- the board outline changes orientation
- the component outlines appear in new positions
- component outlines change sides on the board: black outlines become gray, and gray outlines become black

Zooming

By default, the board viewer displays your board at a zoom level (i.e., "magnification") that fits the entire board ("full fit"). However, you can "zoom in" on the board as much as you need to see more detail.

Zooming In

To "zoom in" on a board:

1. From the View menu, choose Zoom In.

The board viewer zooms in closer to the board.

The center point of the new view is the same as the center point of the old view. To set the center point before zooming in, use the scroll bars, one of the "Zoom to Area" commands, or the "Zoom Pan" command (see "Zooming to an Area Defined by a Box," "Zooming to an Area Around a Point," or "Scrolling a Zoomed-In Board" below for details).

Zooming to an Area Defined by a Box

The Zoom Area command allows you to draw a box around the zoomed-in view you want.

To "zoom in" on an area defined by a box:

1. From the View menu, choose Zoom Area.
   OR
   Click the Zoom into Area button on the toolbar.
   The cursor changes to a magnifying glass.

2. Position the mouse cursor where you want one of the corners of the zoom box to be.

3. Click the mouse button and while continuing to hold the button down, "drag out" a box defining the zoom area. The zoom box appears in gray.

4. Release the mouse button.

The board viewer zooms in to the area defined by the zoom box.

Zooming to an Area Around a Point

The Zoom Point command allows you to draw a "zooming box" centered on an exact point.

To "zoom in" around an exact point:

1. From the View menu, choose Zoom Point. The cursor changes to a magnifying glass.

2. Position the mouse cursor where you want the *center* of the zooming to be.

3. Click the mouse button and while continuing to hold the button down, "drag out" a box defining the zoom area. The zoom box appears in gray; it expands symmetrically around the center point.

4. Release the mouse button.

The board viewer zooms in to the area defined by the zoom box, centered on the starting point.

Zooming to a Scale Value

The board viewer allows you to zoom to a numerical “zoom percentage” or “scale value.”

To zoom to a scale value:

1. From the View menu, choose Options.
2. Use the Zoom Scale slider bar to choose a new zoom percentage.
3. Click OK.

The board viewer zooms in to the new scale value.

The center point of the new view is the same as the center point of the old view. To set the center point before zooming in, use the scroll bars, one of the “Zoom to Area” commands, or the “Zoom Pan” command (see “Zooming to an Area Defined by a Box,” “Zooming to an Area Around a Point,” or “Scrolling a Zoomed-In Board” for details).

Scrolling a Zoomed-In Board

Once your board is zoomed in closer than full view, you may want to scroll the view of the board. There are two ways to scroll or “pan” in the board viewer.

To scroll a zoomed-in board:

1. Use the scroll bars to the right and at the bottom of the board viewer.
   OR
   From the View menu, choose Zoom Pan. The cursor changes shape. Move the cursor to the position which you want to be the center of the new view; then click the mouse.

Zooming Out

To “zoom out” on a board:

1. From the View menu, choose Zoom Out.

The board viewer zooms farther out from the board.

The center point of the new view is the same as the center point of the old view. To set the center point before zooming out, use the scroll bars.

Zooming to a Full View

The board viewer allows you to use a single command to move from a zoomed in view of your board to a "full fit" view, i.e., a zoom level that fits the entire board.

To zoom out to a "full fit":

1. From the View menu, choose Zoom Full.

The board viewer zooms to a full fit.

Zooming to the Previous Zoom Level

The board viewer lets you return from any current zoom level to the previous level. This feature is implemented as a "stack"; as you execute it repeatedly, the board viewer "walks" back through the previous zoom levels you have used.

To zoom to the previous zoom level:

1. From the View menu, choose Zoom Previous.
   OR
   Click the Return to Previous Zoom button on the toolbar.

The board viewer zooms to the level it was previously at.

Viewing Limitations

Unattached Reference Designators

The board viewer displays a reference-designator label for every component listed in the .HYP file's DEVICES list. (See Chapter 4, section "Setting Reference-Designator Mappings" or Appendix D for more details.)

If a component has no outline to which to attach the label, the board viewer displays the label at board position X=0,Y=0. If there are multiple unattached labels, they overwrite each other at position X=0,Y=0.

Viewing All Nets Simultaneously

In addition to viewing nets individually when you choose them for simulation or highlight them, the board viewer can display all of the nets on a board simultaneously.

To enable viewing of all nets simultaneously:

1. From the View menu, choose Options.
2. In the View Nets area, click the All Nets radio button.
3. Click OK.

After a pause (which can be long for large boards), the board is re-displayed with all nets visible.

As soon as you select a specific net for simulation, the viewer returns to displaying only the selected net, its associated nets, and any highlighted nets (see section “Highlighting Nets” above for details on highlighting).

Note: *The view-all-nets mode has NOT been optimized for performance (unlike in a PCB-layout package). If it runs too slowly on your larger boards, don't use it! It is provided for users with fast PC hardware or smaller boards who may occasionally need to view their entire boards at one time.*

Returning to Single-Net Viewing

To return the board viewer from displaying all nets back to displaying only the selected net:

1. Select another net (or re-select the currently selected net) for simulation. The viewer automatically changes back to single-net mode.

   — OR —
   From the View menu, choose Options.

2. In the Show Nets area, click on the Current Net radio button.

3. Click OK.

Viewing Power-Supply Nets

In BoardSim, power-supply nets cannot be selected for simulation (because analysis of power-supply nets is not supported). Therefore, the only way to see power-supply nets in the board viewer is to *highlight* them. For details on how to highlight a net, see section “Net Highlighting” above.

If you highlight a power-supply net which is implemented with a plane layer in the stackup (or a copper pour) — e.g., “VCC” or “Ground” — the board viewer will show you only vias, pads, and any residual segments attached to the plane. The plane itself (or poured copper) does not display.

Viewing Traces on Inner Layers

Sometimes, on dense boards with many layers in the PCB stackup, viewing traces on inner layers can be difficult because traces on other layers obscure the inner-layer details. In these situations, you can use BoardSim’s stackup editor to temporarily re-arrange your board’s layer ordering and thereby move inner layers into more-viewable positions in the stackup.

For example, in a dense 10-layer stackup, you might have trouble seeing the details on inner layer 5 and 6. To work around this, you could use the stackup editor to move layer 5 temporarily to the top of the stackup, and layer 6 to them bottom. Then, flipping the board up and down as needed, all the details of layers 5 and 6 could be seen with obscuration.

Of course, if you use this technique, you must be careful to re-instate the actual stackup before performing any analysis, e.g., interactive or batch-mode simulation, Terminator Wizard, etc.

For details on using the stackup editor to change layer order, see Chapter 5, section “Editing a Stackup.” For details on flipping a board from side-to-side, see section “Orienting a Board” above in this chapter.

Changing the Viewer's Background Color

You can change the background color in the board viewer from its default to anything you prefer.

To change the background color in the board viewer:

1. From the Options menu, choose Preferences. The Options dialog box opens.
2. Click on the Color tab.
3. Click on the Edit Color button. (The color "chip" next to the button shows the viewer's current background color.)
4. Use the standard Windows Color dialog box to select a new color.
5. Click OK twice.

Chapter 8: Interactively Choosing and Editing ICs and Other Components

Summary

This chapter describes:

- what “interactively choosing and editing” components means
- what is in the Assign Models dialog box
- how to choose IC and ferrite-bead models
- how to edit passive-component values
- how to update IC models over the Internet

Interactive versus Automatic Selection of IC Models

BoardSim supports two methods of selecting IC models:

- interactively choosing models, as you simulate each net on your board
- automatically choosing models — “IC AutoMapping” — using an ASCII “.REF” mapping file, created before you load your board

This chapter describes the *interactive* method of selecting IC models. The interactive method allows you to select models only as you need them, net-by-net and pin-by-pin as you simulate. You can also use this method to select models before batch-mode simulation, although if the number of nets you plan to simulate in a batch run is large, it's probably worth constructing a .REF file (more efficient). (For details on batch mode, see Chapter 16.)

Chapter 9 describes the .REF-file method of specifying IC models. The .REF-file method allows you to create an ASCII file that maps IC reference designators to models. For ICs that are mapped in the .REF file, each net's models are automatically selected when you choose the net for simulation.

Note: *The .REF file allows you to specify models for ICs. You cannot specify models for ferrite beads or values for passive components (resistors, capacitors, or inductors).*

Interactive or Automatic Method?

For every board you simulate, you have a choice between the interactive and automatic .REF-file methods of selecting IC models. Which you choose is entirely up to you. (You can also mix the two methods; see "Mixing Interactive and .REF-File Methods" below for details.)

The following table lists some of the advantages and disadvantages of each method:

	Advantages	**Disadvantages**
Interactive method	Only need to select models for the IC pins being simulated	Must choose IC models on a net-by-net, pin-by-pin basis
	No setup required before loading board and beginning simulation	Must select models as each net is simulated for the first time
	Easy if only a small set of nets being simulated	Tedious if a large set of nets being simulated

	Advantages	Disadvantages
Automatic (.REF file) method	Can select IC models component-by-component, rather than pin-by-pin	ASCII file must be created before board is loaded; "smart" editor available for doing it with, though
	Easier if a large set of nets is being simulated	Might be too much work if only a few nets are being simulated

The biggest difference is this: with the interactive method, you specify IC models *pin-by-pin* as you simulate. In the .REF file method, you specify models *component-by-component*. Therefore, if you plan to simulate a large number of nets (for example, in batch mode), it may be worth the time to set up a .REF file, since once the file is created, *every* pin on the ICs you map in the file will load a model automatically when you simulate.

On the other hand, if you plan to simulate only a small set of nets, the overhead of interactively specifying models pin-by-pin may be acceptable compared to the time required to create and debug a .REF file.

Mixing Interactive and .REF-File Methods

You can mix the two methods of selecting models and values. For example, you could create a .REF file from which to automatically load IC models for some of the most important ICs on your board (the ones connected to nets you think are most important to simulate); then rely on interactive model loading for ICs connected to other nets you later decide to also simulate.

If you mix the .REF-file and interactive model-selection methods, and certain IC pins end up using *both* methods, the interactive model choice takes precedence.

See Chapter 9 for details on loading IC models from a .REF file.

Kinds of Interactive Component Choosing and Editing

With the interactive method of selecting models and values, before you can simulate a net you have chosen, you must choose models and edit values for the components on the net. What steps are required depends on each component's type:

- for an IC... ...choose model
- for an R... ...check value; modify if needed
- for a C... ...check value; modify if needed
- for an L... ...check value; modify if needed
- for a ferrite bead... ...choose model

(See Chapter 4, section "What is a Reference-Designator Mapping?" for details on component types and how BoardSim determines them.)

If you have any resistors or capacitors that are packaged as networks rather than discretely, read "Packages for Resistors or Capacitors" below.

Packages for Resistors or Capacitors

Resistors or capacitors that are packaged as networks rather than discretely have package choices in addition to a choice of values. *Resistor and capacitor packaging is described in Chapter 11; this chapter describes only how to set values for the Rs and Cs in the package.*

For example, a discrete resistor R1 has only one parameter that can be edited: its resistance. Everything about R1 that can be changed is described in this chapter.

But resistor network RP1 (a network of four resistors to be used as pull-ups or pull-downs) has two parameters that can be edited: the resistance of the resistors in the network, and the type of package in which the resistors are

housed. This chapter describes how to change the resistor values; Chapter 11 describes how to choose the package.

IC Models

With interactive IC modeling, the first time you choose a net, you must manually choose models for the IC components on the net before simulating. Once you have chosen an IC's model, BoardSim remembers your choice; if you come back to re-simulate the net (in the same BoardSim session or in another), BoardSim will automatically re-load the model for you. (See Chapter 14 for details on how models are remembered between sessions.)

"Choosing a model" for an IC means selecting a model from one of the libraries supplied with BoardSim; or modifying a supplied model, saving it to one of your own libraries, and selecting it.

For details on choosing IC models, see "Interactively Choosing IC Models" below in this chapter.

Resistors, Capacitors, and Inductors

For resistors, capacitors, and inductors, BoardSim has a model built-in, but it must know the *value* of the component.

Note: *For resistors or capacitors that are packaged as networks rather than discretely, BoardSim must also know how the components are packaged. See Chapter 11 for details on choosing packages.*

BoardSim attempts to automatically determine a component's value when it loads the .HYP file for your board. For Rs, Cs, and Ls, each record in the .HYP-file DEVICES list contains a field called "VAL"; BoardSim examines the VAL record and attempts to convert it to a component value. (See Appendix D for details on the .HYP-file format.)

Default R, C, and L Values

Unfortunately, there is no guarantee that the VAL record in the .HYP file will be set by the .HYP-file translator to the component value. In fact, some

translators do not support VAL records at all. In other cases, VAL may be set properly for some components on your board and not for others. If the VAL record is missing or invalid, the corresponding component values are set as follows:

R	1000 ohms
C	0.0 Farads
L	1.0 nanoHenry

You should check the values of all the Rs, Cs, and Ls on a net before you simulate the net for the first time. For details on checking and modifying component values, see "Editing Rs, Cs, and Ls" below in this chapter.

Once you have chosen a component's value, BoardSim remembers your choice; if you come back to re-simulate the net (in the same BoardSim session or in another), BoardSim will automatically re-load the value for you. (See Chapter 14 for details on how values are remembered between sessions.)

Ferrite-Bead Models

Ferrite beads require a model more complex than that for other, simple passive components (e.g., a resistor). You cannot, for example, adequately model a bead by specifying a single numeric value. BoardSim models a bead with a parallel combination of inductance, resistance, and capacitance. The L-R-C model is automatically synthesized from three of the bead's impedance-vs.-frequency points.

"Choosing a model" for a ferrite bead means selecting a model from one of the libraries supplied with BoardSim. BoardSim ships with a library of representative ferrite beads from several leading manufacturers. To choose one of these models, follow the steps in "Choosing Ferrite-Bead Models" below in this chapter.

The first time you choose a net, you must interactively choose models for the ferrite beads (if any) on the net before simulating. Once you have chosen a bead's model, BoardSim remembers your choice; if you come back to re-

simulate the net (in the same BoardSim session or in another), BoardSim will automatically re-load the model for you. (See Chapter 14 for details on how models are remembered between sessions.)

Assign Models Dialog Box

To interactively choose IC and ferrite-bead models and edit passive-component values for a net:

1. From the Select menu, choose Component Models/Values.
   OR
   Click the Select Component Models and Values button on the toolbar.

The Assign Models dialog box opens. It allows you to control modeling for all the components (ICs *and* passive components) on a net. The following sections describe the elements in the dialog box, and how to use them to choose models and edit values.

The Pins List

The Pins list box displays the component pins on the currently selected net and its associated nets. If you are licensed for BoardSim Crosstalk and crosstalk analysis is enabled, the list also shows pins on aggressor nets and their associated nets. (See Chapter 7, section “Choosing Nets” for details on choosing a net; Chapter 7, section “What are Associated Nets?” for details on associated nets; and the Crosstalk User’s Guide for information on running BoardSim Crosstalk.)

Pins are listed in the form:

```
<reference_designator>.<pin_name>
```

For example, U1.2 means pin 2 on component U1; R100.A means pin A on R100.

For passive components (Rs, Cs, Ls, and ferrite beads), the Pins list displays only one pin per component, since the second pin is redundant from the

modeling standpoint. (Both pins share a component value, e.g., for a 100-ohm discrete resistor, both pin 1 and pin 2 have the value “100 ohms.”)

For resistors or capacitors that are packaged as a network (rather than discretely), the Pins list box shows one pin on every sub-component in the package. For more details about network-packaged Rs and Cs, see Chapter 11.

Pin Icons

Each component pin is marked in the Pins list box with an icon that shows the status of its model or value. The icons allow you to easily see which components have models or valid values. You can also easily determine which IC pins on the net are the drivers (“outputs”) and which are receivers (“inputs”).

Note: *If you are licensed for BoardSim Crosstalk, you will see an additional set of icons that distinguish between pins on the victim net versus those on aggressor nets. The icons described in this section appear at the far left in the list box. The aggressor/coupled-net icons appear to the right of the “pin type” icons described here.*

Most icons apply only to certain component types, as described in the following sections.

Red Question Mark

The red question-mark icon applies to all component types. Generically, it means that the component is missing modeling information. More specifically (depending on component type) the meanings are:

for an IC...	...no model selected; treated as an electrical “open”
for an R...	...invalid value
for a C...	...invalid value
for an L...	...invalid value

for a ferrite bead...	...no model selected

Before simulating, you must remove at least some of the red question marks by choosing IC and ferrite-bead models and editing resistor, capacitor, and inductor values. See “Interactively Choosing IC Models,” “Interactively Editing Rs, Cs, and Ls,” and “Choosing Ferrite-Bead Models” below in this chapter for details on models and values.

You are not required to remove all red question marks before you can simulate, though typically you would. BoardSim requires only one driver-IC model before simulating.

The following matrix summarizes some possibilities:

for a complete simulation	remove all red question marks (i.e., supply all IC models and check all component values)
for a quick simulation	supply a driver-IC model; check all resistor and capacitor values; ignore receivers
minimum requirement	supply a driver-IC model

Green Driver Icon

When you select an IC model that has an output-only buffer direction; or when you select an IC model and set its buffer state to Output, Output Inverted, Stuck High, or Stuck Low, the IC’s icon changes from a red question mark to a green driver shape; the driver shape points to the right (as opposed to a receiver shape, which points to the left). (See “Setting IC Buffer Direction/State” below in this chapter for details on buffer direction and state.)

Green Receiver Icon

When you select a new IC model; or when you change a bi-directional IC model’s buffer state to Input, the IC’s icon shape changes to a green receiver shape; the receiver shape points to the left (as opposed to a driver shape,

which points to the right). (See "Setting IC Buffer Direction/State" below in this chapter for details on buffer direction and state.)

Green Check Mark

The green check-mark icon applies only to non-ferrite-bead passive components (Rs, Cs, and Ls). BoardSim marks a component with a green check-mark if it finds a valid value for the component in the .HYP file or the session (.BUD) file. (See Chapter 14 for details on session files.)

If BoardSim does not find a valid value, it sets the component value to a "safe" default number, but marks the component with a red question mark to remind you to check and (if necessary) modify the value. See "Default R, L, and C Values" above in this chapter for details on default values.

The green check mark always applies to all pins on a passive component. E.g., if one pin on a resistor or resistor network has a check-mark icon, all of the other pins on the component will have check marks, too.

The Models Area

The area to the right of the Pins list box is called the "models area." When you highlight a pin in the Pins list box, the models area changes to show you the current status of the pin's model or component value.

Component-Type Icons

The models area displays an icon that allows you to easily see what kind of component you have highlighted in the Pins list box. There are different icons for each of the component types (IC, R, C, L, and ferrite bead).

Note: *Remember that the component type for every pin is determined by the reference-designator mappings that were in effect when you loaded your board. You cannot change component type in the Assign Models dialog box. See Chapter 4, section "Setting Reference-Designator Mappings" for details on reference-designator mappings.*

Component-Data Information

In addition to the component-type icons, the models area displays information about each model or passive-component value. What information is displayed differs depending on which component type is currently highlighted.

For example, if you have a resistor highlighted, you see an editable Value. If you have an IC highlighted, you see a variety of model-related information. See "Interactively Choosing IC Models," "Interactively Editing Rs, Cs, and Ls," and "Choosing Ferrite-Bead Models" below in this chapter for details.

Additional Icon for Networked Resistors and Capacitors

If you highlight a pin on a resistor or capacitor which is packaged as a network (rather than discretely), an additional "connectivity" icon may appear in the models area next to the component-type icon.

The connectivity icon shows you how BoardSim believes the component's package connects the network internally. For details on changing a networked component's package (and internal connectivity), see Chapter 11.

Component-Type Tabs

At the top of the Assign Models dialog box is a series of tabs, labeled with the names of the component types supported by BoardSim. (The tab names also include "Quick Terminator," a special type of "virtual" component; see Chapter 15, section "Quick Terminators" for details.)

When you click on a tab, the highlight in the Pins list box jumps to the first component of that type (if any) in the list.

Other Sections of the Assign Models Dialog Box

The Assign Models dialog box contains several other areas (e.g., the Buffer Settings area, Vcc/Vss Pin combo boxes, and Model to Paste area), but these are discussed in detail in other sections below in this chapter.

Interactively Choosing IC Models

You can interactively choose an IC model for any component pin in the Assign Models pins list that is an IC-type component. (See "Assign Models Dialog Box" above in this chapter for details on the Assign Models dialog box; see Chapter 4, section "What is a Reference-Designator Mapping?" for details on component types.)

Note: *The converse is true, too: for an IC-type component, you can only choose an IC model. If a resistor, for example, is somehow mapped to component type IC, it can only be modeled as an IC, not as a resistor. Mappings are user-definable; see Chapter 4, section "Setting Reference-Designator Mappings" for details on how BoardSim determines component types.*

Important: *You can also choose IC models using the .REF-file method, in which you map IC reference designators to model names in an ASCII file, before loading your board. For details on the .REF-file method, see Chapter 9.*

Relationship between Pins and Models

When you use the interactive method of choosing IC models, BoardSim allows you to choose any model you wish for each pin on an IC. There is no requirement that all the pins on an IC have matching models.

For example, you could set pin 1's IC model to a standard-logic 74AC driver and pin 2's model to a complex-PLD driver, and BoardSim would not complain even though on your real board, the component would have to be one or the other.

This feature lets you experiment with different IC types more easily: even though you would usually set all of the pins on an IC to models from the same family, you have the freedom to make exceptions in order to experiment with different driver/receiver types.

For example, after running some simulations, you might say, "I wonder what would happen to those control lines if I combined the decoder and buffer into a PLD and used the PLD's outputs as drivers?" — and try it on one net by

changing a single output pin's model to a PLD, without bothering to change the other pins on the IC.

With the interactive method of choosing IC models, you are only required to choose models for the pins on the net you are currently simulating. For example, if an IC connects to only one net that you are interested in simulating, you never need to choose a model for more than the one IC pin that drives the interesting net.

Note: *If there is no exact model for a pin you are trying to simulate, you can easily create one. See Chapter 10 for details, especially section "BoardSim Hint: How to Create a Custom IC Model."*

How ICs are Modeled for Signal-Integrity Simulation

A major difference between modeling for digital simulation and modeling for signal-integrity simulation is that for signal-integrity modeling, the simulator does not need to model the logical function of the IC. Only the characteristics of the driver's output stage or the receiver's input stage matter. This means that BoardSim only needs models for every unique buffer type, not for every unique IC. For standard-logic devices especially, this simplifies a signal-integrity tool's modeling libraries.

For example, a 74AC04, a 74AC74, and a 74AC161 all have the same output stage, so all three behave the same in a signal-integrity simulation and can be described by a single model. But a 74AC240 has a different, higher-current output stage, so it requires a different model. Likewise, many of the I/Os on Intel's Pentium microprocessor share the same device model, regardless of their logical functions.

Note: *On the other hand, the models in a signal-integrity library must be detailed analog characterizations, not merely logical models.*

The key parameters for a signal-integrity driver model are rise/fall time, "on" impedance or V-I characteristic, the nature of the impedance change from "off" to "on" (and vice versa), and the output capacitance.

For a receiver model, the key characteristics are the nature of the clamp diodes, input resistance, and input capacitance.

IC-Model Formats

BoardSim supports three kinds of device models:

.MOD format	a HyperLynx proprietary format; many of BoardSim's standard-logic models are in .MOD format
.PML format	an extension to the .MOD databook format that adds to .MOD component pin-outs and package parasitics
IBIS format	an industry-standard format, supported by a variety of simulation and IC vendors; IC vendors can create detailed, accurate IBIS models without giving away proprietary information; BoardSim customers can run IBIS models obtained directly from IC vendors

The following table compares .MOD and IBIS models. For details on each format, see the following sections.

Characteristic	**.MOD Format**	**.PML Format**	**IBIS Format**
ASCII?	✓	✓	✓
keyword-based?		✓	✓
edit in BoardSim?	✓	✓	✓
edit with text editor?		✓	✓

packages modeled?		✓	✓
includes min/max?	✓	✓	✓
signals/pins listed in model?		✓	✓
model includes both driver and receiver?	✓		
easy for users to create?	✓		
supported by other simulators?			✓
contains detailed vendor specs?			✓
available on HyperLynx Web site?	✓	✓	✓

The .MOD Format

MOD models are described in ASCII libraries with extension .MOD. The .MOD format is HyperLynx-proprietary and has been supported since 1989. Many of BoardSim's standard-logic models are in a library called GENERIC.MOD. HyperLynx adds additional .MOD libraries as necessary to support newer ICs.

BoardSim has a dialog-box editor for modifying .MOD models. (See Chapter 10 for details on editing models.) You might edit a .MOD model, for example, as a starting point for a new model. Modified .MOD models can be saved to a user library.

Although .MOD files are ASCII, they are not keyword-based and the .MOD ASCII format is not published. As a result, HyperLynx recommends against modifying them with a text editor; use the .MOD model editor in BoardSim instead.

.MOD files model the silicon portion of an IC plus the device-package capacitance; package inductance is not modeled. If you need package

inductance, use an IBIS or .PML model. Also, .MOD files can model min/max device characteristics as well as typical, but do so through a pair "global" scaling factors that modify up/down the parameters in all .MOD models in a simulation, to give a best-case/worst-case effect. (This contrasts with IBIS models, which contain detailed min/max data on a per-device basis.)

Note: *.MOD files themselves model the silicon portion of an IC; you can use the parasitic modeling capability in LineSim (if you own it) to separately model an IC's package. You can also convert a .MOD model to .PML and include package characteristics.*

.MOD models usually represent an entire family of ICs. For example, the 74ACXX:GATE model in GENERIC.MOD represents the output of any non-line-driver 74AC IC. .MOD models do not contain lists of the specific devices or signals they represent.

.MOD models combine drivers and receivers into a single model. The 74ACXX:GATE model has a driver and receiver; the receiver represents input pins on any 74AC non-line-driver device. For certain devices which are input- or output-only (e.g., a clamp diode), the other "side" of the model can be set to all "off" characteristics and ignored (e.g., for a clamp diode, the output side could be set to all "open" and never used).

The .PML "Package Model Library" Format

The .PML format is an extension to the .MOD databook format that adds component pin-out and package-parasitic information to .MOD models. The .PML — "Package Model Library" — format was added to give .MOD models a more-equal footing with IBIS models.

The .PML format works by adding a new library-file type (with extension .PML) that defines components in an IBIS-like syntax. .PML component definitions attach specific .MOD models to each pin on an IC, and add directionality (input, output, bi-directional, etc.) to each pin. .PML also defines a component's package, and supplies parasitic R, L, and C values for each pin.

.PML files do not themselves contain .MOD models. Rather, .PML files define component pin-outs and point to the models in a .MOD file that actually define the pin's analog behavior. Thus, the .PML file is really just an extension of the

.MOD format. A .PML model requires two files: the .PML file and the .MOD file to which it points.

The IBIS Format

IBIS — "I/O Buffer Information Specification" — is an industry standard for signal-integrity IC modeling. Created originally by Intel, IBIS has been rapidly endorsed by all of the major CAE vendors and IC manufacturers.

Note: *HyperLynx is proud that its LineSim Pro program was the first program in the industry to be declared "IBIS Certified." HyperLynx was a founding member of the IBIS Open Forum, the industry committee that defines and maintains the IBIS standard.*

IBIS is significant because it allows semiconductor vendors to provide accurate models without giving away the proprietary details of their silicon. This removes a key barrier that previously made models difficult to get, or completely unavailable.

IBIS models are described in ASCII libraries with extension ".IBS." BoardSim ships with a collection of IBIS libraries; new libraries are available to customers on HyperLynx's Web site (see "Updating Models over the Internet" below in this chapter for details.)

IBIS models can optionally include detailed package models (pin R,L,C) and min/max data. BoardSim automatically simulates packages if the data is present; you can specify whether to use min, typ, or max data during simulation. (See Chapter 12, section "Setting IC Operating Parameters" for details.)

IBIS models are arranged into components. Each IBIS library may contain multiple components; each component has a list of signals; each signal has an associated model. Each signal may be input, output, or I/O.

IBIS Specification

The IBIS format is public; the IBIS specification is available in Appendix A of this manual and in BoardSim's Help.

Obtaining Models

Semiconductor Vendors

The IBIS format is specifically intended to allow IC vendors to supply signal-integrity models directly to their customers. You are strongly encouraged to request IBIS models from your IC vendors. As paying customers of the IC vendors, you have greater leverage in demanding models than do EDA suppliers.

If you speak with an IC vendor who needs information about or software tools for IBIS development, please have them contact HyperLynx.

From HyperLynx's Web Site

As they become available, HyperLynx posts new models on the HyperLynx Web site. In fact, if your computer is connected to the Internet, you can update your IC models at any time from inside BoardSim. See "Updating Models over the Internet" below in this chapter for details.

The HyperLynx World Wide Web site also provides extensive links to other IBIS modeling sites. See Chapter 19 for details on accessing the Web site.

How to Interactively Choose an IC Model

To interactively choose an IC model:

1. In the Assign Models dialog box, in the Pins list box, highlight the pin for which you want to choose a model. Be sure that the models area displays an IC icon for the pin; otherwise, the pin cannot be assigned an IC model.

2. Click the Select button.
   OR
   Double-click on the pin in the Pins list box.

   The Select IC Model dialog box opens.

3. In the Library list box, highlight the library from which you want choose a model. If desired, use the radio buttons to the left of the Library list box to limit the libraries displayed to one type (IBIS, .PML, or .MOD); you can

also click on the EASY.MOD or GENERIC.MOD buttons to jump to those specific libraries. When you highlight a library, the Device and Pin/Signal list boxes update to display the contents of the selected library.

Note: *The Pin/Signal list box is grayed out for .MOD libraries. Only IBIS and .PML libraries contain lists of component signals/pins.*

---

4. Highlight the device for which you want a model. If the library is .IBS or .PML, the Pin/Signal list box updates to display the contents of the highlighted device.

5. *If the library is .IBS or .PML,* highlight the pin or signal name for which you are choosing a model. If you want to change from choosing by pin name to choosing by signal name (or vice versa), first click the appropriate radio button in the Select By area. The directionality/state of the highlighted pin/signal is shown graphically in the I/O Type area. If the model is differential (applies to .IBS only), the picture will clearly show two drivers or receivers. The I/O Type area also displays any threshold information present in the model.

6. Click OK.
   OR
   If the library is .MOD, double-click on the device name;
   if the library is .IBS or .PML, double-click on the pin/signal name.

   The Select Models dialog box closes, and the model is chosen.

7. Back in the Assign Models dialog box, in the Buffer Settings area, click the appropriate radio button to set the model's driving direction and state. The Settings area displays radio buttons for only those direction/state combinations that are valid for the selected model (e.g., for an output-only model, there is no "Input" button displayed).

8. *For driver (i.e., output) models only,* look at the Vcc Pin and Vss Pin combo boxes. These allow you to power the driver from any two power-supply nets, or from the driver model's "typical" value (i.e., the voltage recommended internally in the model). The "typical" choice appears at the top of each list.

9. To finish choosing IC models, click Close. Or, to choose a model for another IC pin, repeat steps 1 - 8.

For more details about the operations described in these steps, see the following sections.

Choosing a Library

The first step in choosing an IC model is choosing a library. You can always use any of .MOD, .PML, or IBIS models; you can mix the model formats freely in a simulation.

BoardSim ships with several groups of libraries:

GENERIC.MOD	Contains HyperLynx's models for a large number of standard-logic families (all 74XX, ECL, and certain other, miscellaneous models)
other .MOD libraries	Contain vendor- or family-specific models, usually created by HyperLynx. An example is IDT.MOD (IDT FCT devices).
.IBS libraries	Contain family- or component-specific models created by the IC vendors or in a few cases, by HyperLynx. Examples are 82430.IBS (created by Intel), and ALTERA.IBS (created by HyperLynx).
EASY.MOD	Contains completely generic models that are based on technology types (CMOS or bipolar) and approximate switching speeds (e.g., fast, medium, slow). Use this library when you can't find an exact model for a device and don't have time to create or search for one.

The GENERIC.MOD Library

Most of BoardSim's standard-logic models are contained in the library GENERIC.MOD. There are exceptions; some families have their own, separate .MOD libraries.

Model Names in GENERIC.MOD

In GENERIC.MOD, the format for most models' names is

family : type

Examples of device families are:

74ACxx

74FCTxx

74BCTxx

MACH

74Fxx

DRAM

The device types and their meanings are

GATE	gates, with "normal" current drive and capacitance; e.g., 74AC00, 74F74
LIN-DRV or BUS-DRV	line drivers, with enhanced current drive and extra capacitance; e.g., 74F244, 74AC245
OPEN-COL	open-collector; e.g., 7406
OPEN-DRN	open-drain
PLD	any PLD or FPGA
FST/SLW	fast or slow edge of a device with programmable slew rate

The naming format is used especially to distinguish between the normal-output devices and the line drivers in standard-logic families. It also identifies models with special driver output stages, like open collector or fast/slow versions of devices with programmable output slew rate.

Other models — especially those for more-specialized devices — use only the common device name, without specifically identifying the model "type." This is true, for example, of bus-driver families which come only with line-driver outputs.

Modeling Bi-directional Devices

This section describes how to choose models for bi-directional devices when you are using a .MOD library, especially GENERIC.MOD.

When you are modeling a driver with a high-current output, e.g., an 74xx244 or 74xx245 (where xx is any technology type, like "HC" or "F"), use the line-driver version of the model.

When you are modeling a receiver, if the device is bi-directional (i.e., a transceiver) and you are modeling the case where the driver is tristated and the device is receiving, use the line-driver version. (It correctly models the extra capacitance of the tristated driver.)

But if the device is *not* bi-directional and you are modeling one of the input pins, use the "gate" model, *not* the line-driver. The input in this case is no different than a standard gate's input (no extra capacitance).

The following table summarizes the correct choices for the 74xx24x series:

Device	Correct Driver Model	Correct Receiver Model
xx240	LIN-DRV	GATE
xx241	LIN-DRV	GATE
xx242	LIN-DRV	LIN-DRV
xx243	LIN-DRV	LIN-DRV

Device	Correct Driver Model	Correct Receiver Model
xx244	LIN-DRV	GATE
xx245	LIN-DRV	LIN-DRV

Hint: *GENERIC.MOD contains two models — 74HCXX:GATE and 74HCTXX:GATE — that each have two versions. The "-1" version has low-resistance clamp diodes, and the "-2" version has high-resistance. These families come in two versions, depending on the manufacturer. The low-resistance version clamps overshoot and undershoot more effectively. If you're not sure which version you're using, check with the manufacturer.*

The EASY.MOD Library

Sometimes you are in a hurry to simulate and don't have an exact model for a device. In these cases, when there's no time to find the appropriate model (e.g., from the vendor) or create one yourself, use the EASY.MOD library to get an approximate model that will get you simulating, and — in most cases — be sufficiently accurate to give good analysis results.

EASY.MOD is based on the fact that the two most-important parameters in a driver-IC model are the basic technology type (CMOS or bipolar?) and the approximate switching time of the output buffer (fast? slow? medium?). If you know those two basic facts about an IC (usually easy to glean from a data sheet), you can choose a model from EASY.MOD. While the model may not be exactly perfect, it will usually be sufficient to proceed with simulation.

To choose a model from EASY.MOD:

1. In the Select IC Model dialog box, in the Library list box, highlight EASY.MOD.
   OR
   Click the EASY.MOD button.

2. Determine the basic technology type of the IC you're modeling. If it is a CMOS (or similar, e.g., NMOS) device, look in the Device list box at the

model names beginning with “CMOS”. If it is bipolar, look at the models beginning with “TTL”.

3. If the device is CMOS, determine if it runs from a 5.0-V or 3.3-V power supply. Narrow the models to those with the appropriate power supply.

4. Determine approximately how fast the IC switches. The data sheet for the device should give at least some feeling for this. If you don’t know the answer to this question, make a guess based on the IC’s “age.” If it is a brand-new device, for example, it likely switches fairly fast. If an older device, it may switch at a medium or even slow rate. If a specialty device intended for very-high-speed applications (e.g., a clock driver or a specialized bus technology), likely it switches “ultra-fast.” If you have absolutely no idea about switching speed, choose the “FAST” model.

5. Based on these decisions, narrow the model to a single choice. In the Device list box, double-click on the appropriate choice.

Cannot Save into EASY.MOD

You can edit a model in EASY.MOD, but you cannot save the edited model back into EASY.MOD; you must save to a different library. The model editor will not write into EASY.MOD. (See Chapter 10, section “Editing .MOD IC Models” for details on editing .MOD models.)

Note: If you modify a HyperLynx-supplied library, you should rename it first. If you do not change the library’s name, your version of the library will be overwritten next time you receive updated HyperLynx software.

Other Libraries

One other useful library in BoardSim is DIODES.MOD, which contains models for several generic types of clamp diode. If you are using external clamp diodes for termination, for example, you might try one of the models in DIODES.MOD. (Or start with a model in DIODES.MOD and modify it to match the diode you’re using.)

When you use a model in DIODES.MOD, be sure to set the model’s buffer state to Input rather than Output. (Diodes are passive devices; they don’t

"drive." The output sides of the models in DIODES.MOD are completely "open.")

.MOD Libraries

Choosing a Device

If you have chosen a .MOD library in the Select IC Model dialog box, the next step is to choose a device. In some cases (particularly if the library is GENERIC.MOD), you will actually choose a device family rather than a specific device. (See "Model Names in GENERIC.MOD" above in this chapter for details.)

For a .MOD library, the Pin/Signal list box is grayed out, because .MOD libraries do not contain lists of component pins/signals. For the same reason, the Select By radio buttons are grayed out, too. Also, because .MOD models always model both driver (i.e., output) and receiver (input) functionalities, the picture in the I/O Type area shows both directions. If the model contains threshold information, the I/O Type area shows it, too.

Since the library and device are the only choices you can make for a .MOD model, you can double-click on the device name to close the dialog box and make your choice.

Model Information

The Information on Selected Device area displays the current library and device choices. The Signal and Pin choices are grayed out, since they do not apply for .MOD libraries. For .MOD libraries, there is no "source" information; the Notes field does not carry model-specific information.

Searching for .MOD Models

For details on searching for an appropriate .MOD model, see "How to Search for an IC Model" below.

IBIS Libraries

Choosing a Device

If you have chosen a .IBS library in the Select IC Model dialog box, the next step is to choose a device. IBIS libraries show explicit IC component names in the Device list box. (There can be one or multiple ICs in a single IBIS library.)

When you highlight a device, the Pin/Signal list box updates to display the contents of the highlighted device. The list-box title changes depending on whether you choose to sort pin/signal names by pin name or by signal name.

Choosing a Pin or Signal

The last step is to choose a device pin/signal.

To change how pin/signal names are sorted:

1. In the Select By area, click the appropriate radio button.

You can double-click on the pin/signal name to close the dialog box and make your choice. There is sometimes a short pause after you double-click (or click OK), while the model data is loaded from the IBIS file into memory.

The Model Information Area

The Information on Selected Device area (near the IC icon) displays the following information for an IBIS library:

- library name
- device name
- signal name
- pin name
- informational notes

All information is for the currently highlighted pin/signal.

If you are sorting By Pin, the information area is a way to see what signal name corresponds to the pin name you have highlighted (and vice versa).

The Notes Field

The Notes field contains information supplied by the creator of the model, e.g., who created the model and when, how the model is copyrighted, revision history, limitations or recommendations on the model's use, etc. To view all of the information, pull the combo box down and scroll if necessary.

Searching for IBIS Models

For details on searching for an appropriate IBIS model, see "How to Search for an IC Model" below.

Previewing Model Directionality/Type

The I/O Type area shows graphically the directionality/type of any pin's or signal's model (e.g., output-only, input-only, bi-directional, 3-state output, etc.). The picture changes as you highlight various pins/signals in the Pin/Signal dialog box.

This feature allows you to see directionality/type without having to actually select the pin/signal and return to the Assign Models dialog box.

Note: *The I/O Type area also displays input-receiver and output-driver switching thresholds for the highlighted pin's or signal's model. For more information on these threshold values and how they are used in BoardSim, see Chapter 17, section "Delay Rules."*

.PML Libraries

Choosing a Device

If you have chosen a .PML library in the Select IC Models dialog box, the next step is to choose a device. .PML libraries show explicit IC names in the Device list box. (There are typically many ICs in a single .PML library, although there may be only one.)

When you highlight a device, the Pin/Signal list box updates to display the contents of the highlighted device. The list-box title changes depending on whether you choose to sort pin/signal names by pin name or by signal name.

Choosing a Pin or Signal

The last step is to choose a device pin/signal.

To change how pin/signal names are sorted:

1. In the Select By area, click the appropriate radio button.

You can double-click on the pin/signal name to close the dialog box and make your choice. There is sometimes a short pause after you double-click (or click OK), while the model data is loaded from the .PML file into memory.

The Model Information Area

The Information on Selected Device area displays the following information for a .PML library:

- library name
- device name
- signal name
- pin name
- informational notes

All information is for the currently highlighted choice.

If you are sorting By Pin, the information area is a way to see what signal name corresponds to the pin name you have highlighted (and vice versa).

The Notes Field

The Notes field contains information supplied by the creator of the model, e.g., who created the model and when, how the model is copyrighted, revision history, limitations or recommendations on the model's use, etc. To view all of the information, pull the combo box down and scroll if necessary.

Previewing Model Directionality/Type

The I/O Type area shows graphically the directionality/type of any pin's or signal's model (e.g., output-only, input-only, bi-directional, 3-state output, etc.). The picture changes as you highlight various pins/signals in the Pin/Signal dialog box.

This feature allows you to see directionality/type without having to actually select the pin/signal and return to the Assign Models dialog box.

If a model contains threshold information, the I/O Type area shows it, too.

Check Mark Status

After you have chosen the model, you return to the Assign Models dialog box. The model's default direction and state depend on several factors:

- If the model is output-only (can't be an input), the default state is "Output"
- If the model is input-only (can't be an output), the default state is "Input"
- If the model can be an input or an output, and there was previously a model specified for the pin, then the default state matches the direction/state set for the previous model
- If the model can be an input or an output, and there was NOT previously a model specified for the pin (i.e., red question mark), then the default state is "Input"

For more details on setting buffer direction/state, see "Setting IC Buffer Direction/State" below.

Setting IC Buffer Direction/State

To change an IC model's buffer direction/state:

1. In the Buffer Settings area, click the appropriate radio button.

Only the buttons for the buffer states that are valid for the model are displayed. See "Possible Buffer States" below for more details on possible states.

If you change a model from state Input to some type of Output, or from some type of Output to Input, the icon in the Pins list box changes direction. The icon for an Input points to the left; the icon for an Output points to the right.

There is also a picture in the Buffer Settings area that shows graphically the direction and type of the currently chosen model.

Note: *.MOD models always include both driver and receiver data, so if you choose a .MOD model, both the Input and various Output radio buttons are available. IBIS and .PML models can be unidirectional or bi-directional, so sometimes only a few buffer states are available (e.g., if a receiver pin, only Input).*

Possible Buffer States

IC models can have nearly any mixture of the following buffer states:

- **Output —** Indicates a driving or output state; the sense of the output is "true," meaning that in the oscilloscope, the model will follow the sense of the Driver Waveform setting (i.e., will fall on a falling edge and rise on a rising edge). If the oscilloscope is set to waveform type "Oscillator," the model will start with a rising edge.

- **Output Inverted —** Indicates a driving or output state, but with inverted sense, meaning that in the oscilloscope, the model will invert the sense of the Driver Waveform setting (i.e., will rise on a falling edge and fall on a rising edge). If the oscilloscope is set to waveform type "Oscillator," the model will start with a falling edge. This setting is most useful for differential-driver pairs, where one pin's model must be inverted relative to the other.

- **Output Hi-Z —** Indicates a passive, high-impedance state in which a driver or output model has "turned off." This selection is available exclusively for output-only models that have "tristate" capability; the high-impedance state of an I/O or bi-directional model is selected with state "Input."

- **Input —** Indicates a receiving or input state. In this state, an I/O or bi-directional model's driver is turned off, and the pin is receiving. For input-only pins, this is the only available state.

- **Stuck High —** Indicates a driving or output state, but one in which the model stays high and never switches. In the oscilloscope, the model will stay high regardless of the Driver Waveform setting. This setting is useful for wired-OR or wired-AND buses for which you want to investigate the effect of one driver switching while other drivers remain static. It is also extremely important for crosstalk simulations, in which the driver on "victim" net is usually simulated in a static, "stuck" state (see the Crosstalk User's Guide for details).

- **Stuck Low —** Indicates a driving or output state, but one in which the model stays low and never switches. In the oscilloscope, the model will stay low regardless of the Driver Waveform setting. This setting is useful for wired-OR or wired-AND buses for which you want to investigate the effect of one driver switching while other drivers remain static. It is also extremely important for crosstalk simulations, in which the driver on "victim" net is usually simulated in a static, "stuck" state (see the Crosstalk User's Guide for details).

Which buffer states are available for any particular model is determined by the model's internal "type." In practice, no model can simultaneously have all of the buffer types described above. (For example, types "Input" and "Output Hi-Z" are mutually exclusive.)

Input versus Output Hi-Z

Two buffer states — Input and Output Hi-Z — are similar to each other, but not quite the same. Both generally refer to a state in which an IC pin is high-impedance and not actively driving the net to which it is connected. The difference is that Input is available on either "pure" input pins (pins that can never drive) or on I/O or bi-directional pins that can act as inputs when the output or driving circuitry is shut off; whereas Output Hi-Z is available on pins that can only drive or be turned "off," but have no receiver-input stage.

If you have an IC pin for which the data sheet refers to the "high-impedance" state, but when you model it in BoardSim there is no Output Hi-Z selection

available, choose buffer state Input instead. This means that the pin is actually I/O; when the data sheet refers to "high-Z" it means that the driving circuitry is disabled and the pin is in a high-impedance receiving (i.e., "input") state.

Threshold Voltages

The Buffer Settings area also displays the input-receiver and/or output-driver switching thresholds present in the IC model for the currently highlighted pin. Only the thresholds relevant to current buffer-state choice are displayed, e.g., if for a bi-directional pin you set the buffer state to "Input," the input thresholds Vih and Vil are shown; if you change the state to "Output," the output-switching threshold "Vmeasure" is shown.

Thresholds are used by BoardSim's Board Wizard when it calculates timing delays. For more information on thresholds and how they are used, see Chapter 17, section "Delay Rules."

Setting the Vcc or Vss Pin

For every IC pin with its buffer direction/state set to some type of Output, BoardSim allows you to choose a Vcc and a Vss pin, which in turn allows you to power the driver IC from any two power-supply nets. (You can also power the driver from the "typical" power-supply voltage contained in the model itself; see '"Typical' Power Supplies" below for details.) The models for the driver and all the receivers on the net then run off the voltages to which the Vcc and Vss pins are connected.

Typical" Power Supplies

"When convenient, you can run either or both of a driver IC's supply pins from the "typical" power-supply voltages specified in the driver's model. This is useful, for example, when a power-supply pin is missing or unconnected in your board layout. See "If Both Vcc and Vss are Set to 'Typical'" below in this chapter for more details on typical power supplies.

Note: *The typical values are built-in to every IC model (whether .MOD, .PML, or .IBS) and specify the voltages at which the IC most commonly runs. Any time you have trouble connecting a driver-IC model to a "real" power-supply net in a BoardSim simulation, you should defer to using the model's typical*

value instead. The typical value is a "virtual" supply voltage to which you can always safely attach.

How BoardSim Defaults Vcc and Vss Pins

BoardSim attempts to automatically determine a Vcc and Vss pin for every IC pin on your board. (The Vcc/Vss pins only apply if the IC pin is made an Output in a simulation; see "Setting IC Buffer Direction/State" above in this chapter for details on setting buffer direction/state.)

To identify an IC's Vcc and Vss pins, BoardSim looks at the power-supply nets attached to the IC. If an IC is not connected to any power-supply net, the driver's Vcc and Vss voltages are defaulted as follows:

- if a .MOD or .PML model, to the Default Power Supply in the .MOD model (see Chapter 10, section "Default Power Supply" for details)
- if an IBIS model, to the typical supply voltage in the .IBS model

If an IC is connected to only one non-ground power-supply net (the most-common case), BoardSim will always correctly find a Vcc and Vss pin for all of the driver pins on the IC. In fact, using these rules, BoardSim will correctly find Vcc and Vss pins for most of the ICs on your entire board, provided that BoardSim has a complete list of the power-supply nets for the board. (For this reason, you should always check the power-supplies list when you first load a board, before choosing IC models. See Chapter 6, section "Editing Power-Supply Nets" for details on editing the power-supplies list.) When the rules fail to determine the correct pins, you can manually edit the choices as described below.

Limitations of Automatic Vcc/Vss-Pin Detection

There are situations in which BoardSim cannot correctly identify a Vcc or Vss pin for an IC pin:

1. *If the pin's IC runs off multiple power-supply rails (e.g., some of its pins off one rail and other pins off another).* BoardSim will associate each set of power-supply pins to one power-supply net, then for a particular pin's driver model, arbitrarily choose one power-supply pin. Because this

selection is arbitrary, you may need to change the pin's model manually to run off the correct power-supply pin. See "Choosing a Vcc or Vss Pin" below for details.

2. *If the IC is connected to no power-supply nets.* In this case, BoardSim chooses a default voltage for the IC to run off; see "How BoardSim Defaults Vcc and Vss Pins" above. But if this occurs, you are very likely missing nets in the power-supplies list; see Chapter 6, section "Undetected Power-Supply Nets" for details on how to add nets to the power-supplies list.

With Multiple Drivers

If multiple driver ICs with different default power-supply values are present in the simulation, BoardSim will attempt to connect each to a unique and correct power-supply value. However, the automatic connections may not always be correct, so it is wise to check which supplies each driver IC is running from before simulating. If you run a simulation and see unexpected initial or final voltages, check the driver-IC supply-pin settings.

Changing a Vcc or Vss Pin

To change a Vcc or Vss pin:

1. In the Assign Models dialog box, in the Pins list box, highlight a driver IC's pin.

2. In the models area, pull down the Vcc Pin or Vss Pin combo box.

3. Choose a pin that connects to the desired power-supply net. Note that the model's "typical" supply value is available at the top of the combo box's list.

For example, suppose you highlight pin U7.2, and pin 2 runs off a special non-VCC supply voltage called V_OUT. Pull down the Vcc Pin combo box, and from the following list of pins:

7	GND
8	V_OUT
14	VCC

choose pin 8. Now the driver model for U7.2 will run off whatever voltage V_OUT is set to in the power-supply editor.

The Vcc Pin and Vss Pin combo boxes show all of the pins on the IC that are connected to power-supply nets. Each entry in the list displays a pin name and the name of the associated power-supply net. There is also always an entry *at the top of the list* called "use model's 'typical' value". Choosing this connects the Vcc or Vss pin to the typical Vcc or Vss value specified in the IC's model.

If the IC has several pins connected to the desired power-supply net (e.g., pins 18 and 52 both connect to net VCC), you can choose any of the pins to get the desired connection.

Changing a Vcc or Vss Voltage

To change a driver model's Vcc or Vss voltage:

1. First, check that the IC pin has a correct Vcc or Vss pin identified. (See "Choosing a Vcc or Vss Pin" above in this chapter for details.)

2. If so, close the Assign Models dialog box and use the power-supply editor to change the voltage on the power-supply net to which the Vcc or Vss pin connects. See Chapter 6, section "Editing Power-Supply Nets" for details on changing power-supply voltages.

If Both Vcc and Vss are Set to 'Typical'

If both the Vcc Pin and Vss Pin values are set to "use model's 'typical' value", one of the values automatically becomes 0.0V. If the typical supply voltage is positive, then Vss=0.0V; if the typical supply voltage is negative, Vcc=0.0V.

Vcc and Vss Errors

If you set the Vcc and Vss Pin values — or the voltages for the corresponding power-supply nets — such that Vcc is more negative than Vss, or the difference between Vcc and Vss is less than 0.5 V, BoardSim will give an error when you attempt to simulate. Correct the values before proceeding.

Other Parameters and Power Supply

If you change a Vcc or Vss voltage to a value other than the typical value specified in a driver's model, BoardSim does not automatically correct the other parameters in the model to reflect the new supply voltage.

For example, any of BoardSim's .MOD-format 5-V CMOS models can be made to run at VCC=3.3V, but the model's slew rates, "on" impedances, etc. may no longer be correct; compensating them is up to the user. (On the other hand, the models created specifically for 3.3V are correct at 3.3V, but may need compensation to run at, say, 2.5V.)

Reminder About Power Supplies for Receiver ICs

For every IC pin with its buffer direction set to some type of Output, BoardSim allows you to choose a Vcc and Vss pin. The simulation models for the driver and all the receivers on the net then run off the voltages to which the Vcc and Vss pins are connected. In the current version of BoardSim, you cannot set Vcc or Vss pins for receiver ICs; receivers must run off the same supply voltage as the drivers on a net.

For example, if Net1 has a driver output model at pin U1.1 and two receivers at U2.2 and U2.3, you can choose a Vcc pin for the IC model at pin U1.1. If the Vcc pin is pin 16, and pin 16 is connected to net VCC which is a power supply with voltage 5.0V, then ICs U1 *and* U2 will run off a 5.0-V power supply. If you change net VCC in the power-supply editor to have a voltage of 3.3V, U1 and U2 will run off a 3.3-V supply.

Note: *You can save yourself work by making certain that ALL of the power-supply nets on your board are in BoardSim's power-supplies list BEFORE you begin selecting IC models. If key power-supply nets (like Vcc, masquerading under a less-common name) are unknown when you begin adding models, you will be forced to manually change the Vcc/Vss Pin values at driver ICs. Check the power-supply nets list (choose Edit/Power Supplies) immediately after you load your board, and before you begin adding IC models.*

Warning about Power-Supply Pins in BoardSim EMC

The position of power-supply pins affects the amount of radiation generated by an IC package. BoardSim EMC searches for power-supply pins by looking at

the nets to which the pins on each IC are connected. Even though you can run simulations with an IC that is not connected to power-supply nets (by choosing the “typical” Vcc and Vss values; see “'Typical' Power Supplies” above for details), in this case BoardSim will not know the locations of the power-supply pins on the IC, and will use the positions of the pins farthest from the pin being simulated. In some cases, this may lead to prediction of too much radiation.

If you have such a situation (e.g., on a partially routed PCB where power-supply connections have not yet been made), turn off prediction of package radiation and focus entirely on the radiation being generated by nets' trace segments. For details on disabling package-radiation prediction, see the EMC Analysis User's Guide.

How to Search for an IC Model

If you are in the Select IC Model dialog box, but don't know in which library to find an appropriate device model, BoardSim includes a model-finding feature that may help. The model finder accesses in a spreadsheet a database listing of all models available in HyperLynx's shipping software; you can use this database to search for component names, look for all of the IC models from a particular vendor, and so forth.

To open the model-finding database:

1. From inside the Select IC Model dialog box, click the Find Model button. After a brief pause, the IC Model Finder dialog box opens. (The pause occurs because the database of available IC models is very large, and takes some time to load.)

2. If desired, adjust the size of the finder's spreadsheet window to be larger, by grabbing it with the mouse on an edge or in a corner and dragging it to the desired size.

The model-finder spreadsheet lists all of the component models available when HyperLynx last compiled its libraries of device models (thousands of entries). You can search this list for particular components or device names, and sort on various criteria to group together, e.g., all ICs from a particular vendor, all recent models, etc.

To search for a text string in the model-finder spreadsheet:

1. In the Search String box, type the text for which you want to search.
2. Click the Search for String button. After a pause, all of the items matching the string (in any data column) are check marked in the far left-hand column and brought to the top of the spreadsheet.

To sort the spreadsheet based on a column's data:

1. Click the header button (at the top of the column). After a pause, all of the items in the spreadsheet are sorted in ascending order of the column's data.
2. To sort in the opposite order, click the column's header button again.

For example, to bring to the top of the spreadsheet the most-recent models in the database, click once on the File Date column's header button; after a pause, the list is sorted from oldest to newest model (ascending order). Click again, and the list changes to newest-to-oldest order (descending order).

Selecting a Model Directly from the Model Finder

If you find in the model-finder spreadsheet a model that you want to use in the Select IC Model dialog box, you can select it directly from the spreadsheet.

To choose a model directly from the spreadsheet:

1. Click once on the model's line in the spreadsheet, to highlight it.
2. Click OK. The model-finder dialog box closes and the model is automatically selected in the Select IC Model dialog box.

Updating the Model Finder's Database

The model finder stores its database in a comma-separated-values file called HyperLynxIcModels.CSV; this file resides in the LIBS subdirectory along with all of your IC models (.IBS, .MOD, and .PML).

Occasionally, you may want to update the .CSV database file, for example to add to it models that you have obtained or created on your own. BoardSim includes a built-in utility for generating an updated database.

To generate an updated model-finder database, which includes all of the models currently in your LIBS subdirectory:

1. From the Files menu, choose Generate Model Finder Index.

2. A DOS or command-prompt window opens, and the HyperLynx utility "MAKE_CSV" runs. When it finishes, the DOS box closes and the file HyperLynxIcModels.CSV has been updated.

The database file now includes information about every model currently in your LIBS subdirectory. Any models that you have deleted from the directory are no longer in the database. However, your previous database file has been saved as HyperLynxIcModels.BAK, in case you need to restore it.

Copying IC Models

Once you have interactively chosen a model for one IC pin, it may be convenient to quickly "copy" the same model to other pins. This is particularly true of receiver models, and of .MOD models generally since they are not pin-specific.

For example, suppose a net has one driver (i.e., "output") and ten receivers, and all the receivers are 74ACxxx inputs. Once you have chosen a model for one of the receivers, it is convenient to paste it immediately to the other receivers on the net.

Copying an Existing Model

To make a copy of a model:

1. In the Assign Models dialog box, in the Pins list box, highlight an IC pin that has been assigned the model you want to copy.

2. In the Model to Paste area, click the Copy button.

The information in the Model to Paste area updates to display the copied model. Also, the Paste and Paste All buttons become available (ungrayed).

The Model to Paste Information Area

The Model to Paste area shows the following information for the copied model:

- library name
- device name
- signal name
- pin name

All information is for the last-copied IC model.

If you highlight a different component pin in the Pins list box, the information in the Model to Paste area remains the same. The Model to Paste area is a storage buffer: it always remembers the model you last copied.

Pasting to Another IC Pin

To paste the last-copied model to another IC pin:

1. Be sure that the model you want to paste is displayed in the Model to Paste area. (See "Copying an Existing Model" above for details on copying a model.)
2. In the Assign Models dialog box, in the Pins list box, highlight the IC pin to which you want to copy the model.
3. Click the Paste button.

After a brief pause (while the model data is loaded into memory):

- the Pins list box updates with the new model assignment
- the model-information area updates; the fields display the new model's name and other data

A pasted model defaults to the buffer direction/state of the model that was copied. See “Setting IC Buffer Direction/State” above in this chapter for details on changing the direction.

Pasting to All Other IC Pins

To paste the last-copied model to all other IC pins:

1. Be sure that the model you want to paste is displayed in the Model to Paste area. (See “Copying an Existing Model” above for details on copying a model.)

2. Click the Paste All button.

BoardSim pauses while the model is loaded into memory for every IC pin on the net. For nets with a large number of component pins, this may take a while. When the operation is complete, the Pins list box updates with the new model assignments.

A pasted model defaults to the buffer direction/state of the model that was copied. See “Setting IC Buffer Direction/State” above in this chapter for details on changing the direction.

Note: *If you are licensed for BoardSim Crosstalk and are running with crosstalk analysis enabled, pasting a model may result in the contents of the Pins list box changing. This is due to the fact that aggressor nets are chosen based in part on the characteristics of the driver ICs on nearby nets; a change in a driver IC may change how strongly a given net couples to the selected net, and cause, e.g., that net to be dropped as an aggressor. If a net is dropped, then its pins will disappear from the Pins list. See the Crosstalk User's Guide for details.*

Quickly Creating Multiple Receivers and One Driver of the Same Type

Sometimes it is convenient to make all of the ICs on one net use the same model, with one of the pins set as a driver (i.e., an “output”), and the rest as receivers (“inputs”). The Paste All command can be used to do this efficiently, as follows:

To set all pins on a net to the same model, with one pin as a driver:

1. Select the desired model for one pin on the net. In the Buffer Settings area, set the pin's direction/state to Input.

2. With the same pin still highlighted in the Pins list box, click the Copy button.

3. Then click the Paste All button. All of the other ICs on the net are set to the same model, with every pin's direction/state set to Input.

4. Now highlight the pin which you wish to be the driver. In the Settings area, click the Output radio button.

Note that if you set the initial pin to state Output before copying and pasting, all of the other pins will also be set during the Paste operation to state Output. Since you actually want them to be Inputs, be sure to perform the copy operation with the pin set to Input, and change it later (after the Paste) to Output.

Changing IC Models

To change a pin's IC model from one choice to another, simply re-choose the pin's model. Follow the steps in "How to Interactively Choose an IC Model" above in this chapter.

How to Choose Models for a Differential Driver or Receiver

The steps for choosing models for a differential-IC driver or receiver are the same as those for choosing a model for "regular" non-differential IC, except that in the differential case you must choose two models (one for each half of the differential pair) and invert one of the pin's buffer states. For details on choosing models generally, see "How to Interactively Choose an IC Model" above in this chapter.

To choose IC models for a pair of differential pins:

1. In the Assign Models dialog box, in the Pins list, highlight a pin on the differential pair.

2. Choose a model for the pin (as described above in "How to Interactively Choose an IC Model"). If you are selecting a pin from an IBIS differential model, be sure that the pin name in the model matches the pin name on the PCB (see "With IBIS Differential Model, Pins Names Must Match PCB" below for details).

3. Then highlight the second pin in the differential pair, and choose its model. If the pair's second pin is not visible in the list, read section "If Second Pin is Not Visible" below for possible reasons why.

4. Of the two pins in the differential pair, in the Pins list box, highlight the one that is the '+' or "positive" pin, and set its buffer direction/state to Output.

5. Highlight the '-' or "negative" pin, and set its direction/state to Output Inverted.

You can use any type of IC model — IBIS, .MOD, or .PML — in a differential simulation. However, in BoardSim, .MOD and .PML models can only be used on nets pairs with line-to-line termination. (No such restriction exists for IBIS models, unless you're using a non-differential model for differential pins.) For more details on this restriction, see the following section.

If Second Pin is Not Visible

The second (or "opposite") pin in a differential-net pair actually resides on a different net than the first pin. BoardSim must automatically detect that a second net is involved, and makes its pin(s) available for model choosing and simulation (i.e., BoardSim must detect an "associated net"; see Chapter 7, section "What Are Associated Nets?" for more details). This association occurs in two possible ways:

- because there is a terminating component (e.g., a resistor) on the board connecting the two nets

- because the model you choose for the first pin in the differential pair has internal information stating that a second pin is involved

If neither of the above conditions is met by a net you are simulating and the model you choose for its pin, then the net association will not occur and you cannot simulate differentially. The following table summarizes the possibilities:

Terminator Across Lines?	Model Type	Can Simulate Differentially?
yes	any	yes
no	IBIS differential	yes
no	IBIS non-differential	no
no	.MOD or .PML	no

Note that if the conditions for net association and therefore differential simulation are not met, you can still simulate the two nets in the pair individually, in a single-sided fashion.

With IBIS Differential Model, Pins Names Must Match PCB

The net association described in the preceding section occurs with an IBIS model only if the model contains differential-pin information internally (i.e., tells which pins pair with which other pins), AND if you choose the correct pin in the model for the PCB pin you're trying to simulate. For example, if you're using an IBIS differential model that pairs pins 1-2, 3-4, and 7-8, and you choose the model for pin 1 and apply it to pin 7 on your board, BoardSim will not associate the other net in the differential pair. If you correct the model choice to match pin 7 in the model with pin 7 on your board, then proper association will occur and pin 6 will appear, as expected, in the Assign Models dialog box.

How to Tell if an IBIS Model is Differential

If an IBIS model contains differential-pair information (i.e., tells which pins pair with which other pins), the pictures in the Buffer Settings area in the Assign Models dialog box and in the I/O Type area in the Select IC Model dialog box clearly show a pair of drivers/receivers. If the pictures for a particular model show only single drivers/receivers, then the model is not differential.

Note: *You can also look in the IBIS file itself, using the Visual IBIS Editor. Differential-pin pairings (if they exist in a model) are described in a table that begins with a [Diff Pin] keyword. If there's no such keyword and table in a particular IBIS model, then it's not a differential model.*

Removing IC Models

Occasionally, you may want to interactively remove a previously selected IC model from a component pin, so that the pin has no model. This is equivalent to "lifting" an IC pin from your board.

In the Assign Models dialog box, when an IC pin is highlighted in the Pins list box, a Remove button appears near the Select button (in the models area).

To remove an IC model from a pin:

1. In the Assign Models dialog box, highlight in the Pins list box the pin whose model you want to remove.

2. Click the Remove button.

The model previously assigned to the pin is removed, and the green driver or receiver icon changes back into a red question mark.

The Remove button is only available when an IC pin is highlighted; you cannot remove a passive component (e.g., resistor or capacitor) from a pin. To remove the *effects* of a passive component, set its value to 0.0 or a large number (depending on whether it is in series or parallel with the selected net).

Cannot Remove a Model Specified in a .REF File

If a model is loaded from the .REF file (see Chapter 9, section "Creating a .REF File" for details on using a .REF file), you cannot remove it by using the Remove button in the Assign Models dialog box. If you highlight in the Pins list box a pin with a .REF-specified model, clicking the Remove button will do nothing.

Instead, to remove a model specified in a .REF file, you must remove the model assignment from the .REF file itself. (Again, see Chapter 9.)

Interactively Editing Rs, Cs, and Ls

For resistors, capacitors, and inductors, there is no model to choose, but you can edit the component's value. In fact, you may be required to edit values; see "Resistors, Capacitors, and Inductors" above in this chapter for a description of how BoardSim finds component values.

When you highlight an R, C, or L pin in the Pins list box in the Assign Models dialog box, the models area displays:

- an icon identifying the component type
- a box that displays the component's value, and allows you to enter a new value

Changing a Resistor Value

To change a resistor value:

1. In the Assign Models dialog box, in the Pins list box, highlight a pin on the resistor. Be sure that the models area displays a resistor icon for the pin (otherwise, the pin is not on a resistor).

2. Type a new value into the Value edit box. Values can be entered as a "simple" number (e.g., "1000" or ".01") or in scientific notation (e.g., "1e3" or "1e-2").

The value is always measured in ohms.

New Value Applies to Whole Component

When you change the value for one pin on a resistor, the value changes for all pins on the resistor. For example, if a discrete resistor has pins 1 and 2, and you change the value for pin 1, it is automatically changed for pin 2. (However, since the Pins list box normally shows only one of the resistor's pins, you would have to select another net to see the second pin.) The same is true of resistor networks (multiple resistors in a network, inside a single package). If you change the resistor value for any pin on the package, the value of all resistors in the network changes. (See Chapter 11 for details on passive-component packages.)

Changing a Capacitor Value

Changing a capacitor value is exactly the same as changing a resistor value (see the steps in "Changing a Resistor Value" above), *except:*

- be sure that the models area displays a capacitor icon
- the value is always measured in picoFarads (pF)

Changing an Inductor Value

Changing an inductor value is exactly the same as changing a resistor value (see the steps in "Changing a Resistor Value" above), *except:*

- be sure that the models area displays an inductor icon
- the value is always measured in nanoHenries (nH)

Copying a Value

You can copy one R's, C's, or L's value and paste it to another component of the same type. For example, you can copy one resistor's value, then paste it to another resistor. You cannot paste a resistor's value to a component of a different type, e.g., to a capacitor or inductor.

To copy an R's, C's, or L's value:

1. In the Assign Models dialog box, in the Pins list box, highlight a pin on the component whose value you want to copy.

2. Click the Copy button.

The information in the Model to Paste area updates to display the copied value. Also, the Paste and Paste All buttons become available (ungrayed).

The Model to Paste Information Area

The Model to Paste area shows the value copied from the component.

If you highlight a different pin on a component of the same type in the Pins list box, the information in the Model to Paste area remains the same. The Model to Paste area is a storage buffer: it always remembers the model you last copied.

If you highlight a pin on a component of a different type (e.g., you copied a value from a resistor, and then highlight a pin on a capacitor), the Model to Paste area shows the last value (if any) copied from a component of the type now highlighted. For example, if a capacitor pin is highlighted, the Model to paste area shows the last *capacitor* value that was copied. Separate buffers are maintained for each component type.

Pasting to Another Component

To paste the last-copied value to another component:

1. Be sure that the value you want to paste is displayed in the Model to Paste area. (See "Copying a Value" above for details on copying a value.)

2. In the Assign Models dialog box, in the Pins list box, highlight a pin on the component to which you want to copy the value. The component must be of the same type (e.g., resistor or capacitor) as the one from which the value was copied.

3. Click the Paste button.

The component's value is updated.

Pasting to All Other Components of the Same Type

To paste the last-copied value to all other components of the same type:

1. Be sure that the value you want to paste is displayed in the Model to Paste area. (See “Copying a Value” above for details on copying a value.)

2. Click the Paste All button.

The values are updated for every component of the type whose value is in the Model to Paste buffer. For example, if a resistor value is currently in the Model to Paste area, Paste All updates all of the resistors connected to the current net and its associated nets.

Adding Resistors and Capacitors

You can add to your board terminating resistors and capacitors that are not in the actual layout, and edit and experiment with their values just like with “real” components. This feature is known as “Quick Terminators,” and is described in Chapter 15, section “Quick Terminators.”

You can only add resistors or capacitors that are not in the actual layout, not other component types, i.e., ICs, inductors, or ferrite beads. See Chapter 15.

Choosing Ferrite-Bead Models

Ferrite beads require a model more complex than that for other, simple passive components (e.g., a resistor). Other passive components are modeled with a single value (e.g., “100 ohms” or “33 pF”); ferrite beads are modeled more like an IC, with a detailed model contained in a library. “Choosing a model” for a ferrite bead means selecting a model from one of the libraries supplied with BoardSim.

BoardSim ships with a library of representative ferrite beads from several leading manufacturers. You can also create your own ferrite-bead models. For details on how BoardSim models beads, and how to create your own bead models, see Chapter 10, section “Creating Your Own Ferrite-Bead Models.”

You can choose a ferrite-bead model for any component pin in the Assign Models pins list that is a ferrite-bead-type component. (See "Assign Models Dialog Box" above in this chapter for details on the Assign Models dialog box; see Chapter 4, section "What is a Reference-Designator Mapping?" for details on component types.)

Note: *Currently in BoardSim, you can only model ferrite beads using the interactive method. The .REF file, from which you can load IC models by mapping IC reference designators to model names, does not currently support ferrite beads. For details on the .REF-file method, see Chapter 9.*

Relationship between Pins and Models

When you choose models for ferrite beads in BoardSim, a single model applies to the entire component. This differs from choosing models for an IC; each pin on an IC can have a different model. (See "Interactively Choosing IC Models" above in this chapter for details.)

Note: *If there is no model for the exact ferrite bead you are trying to simulate, you can create one of your own. See Chapter 10, section "Creating Your Own Ferrite-Bead Models" for details.*

How to Choose a Ferrite-Bead Model

To choose a ferrite-bead model:

1. In the Assign Models dialog box, in the Pins list box, highlight a pin on the ferrite bead for which you want to choose a model. Be sure that the models area displays a bead icon for the pin; otherwise, the pin cannot be assigned a bead model.

2. Click the Select button.
   OR
   Double-click on the pin in the Pins list box.

   The Select Ferrite Bead Model dialog box opens.

3. In the Vendor list box, highlight the vendor for whom you want choose a model. The Part Number list box updates to display a list of beads from the highlighted vendor.

4. Highlight the part number matching the bead for which you want a model. The data in the Model Values area and the impedance-vs-frequency graph update to display the modeling parameters for the highlighted bead.

5. Click OK.
   OR
   Double-click on the part number.

   The Select Ferrite Bead Models dialog box closes, and the model is chosen.

6. To finish choosing bead models, click Close. Or, to choose a model for another bead, repeat steps 1 - 5.

For more details about the operations described in these steps, see the following sections.

About the Ferrite-Bead Models Supplied by HyperLynx

All of the models listed in the Select Ferrite Bead Model dialog box (unless you have created additional models of your own) are contained in a HyperLynx-supplied library called “BSW.FBD.” This library contains a representative sampling of beads from several of the leading manufacturers. Usually, even if the bead you’re using is not contained in the library, you can find a close match to it in BSW.FBD.

If you want to create a bead model of your own, you can do so by creating a file called “USER.FBD.” For details on how to create your own bead models, and how BoardSim models ferrite beads generally, see Chapter 10, section “Creating Your Own Ferrite-Bead Models.”

The Model Values Area

For the currently highlighted vendor and part number, the Model Values area shows an equivalent L-R-C model for the bead. These values are automatically synthesized from three of the bead’s impedance-vs-frequency points.

The Impedance-vs-Frequency Graph

Above the Model Values area and Vendor and Part Number list boxes is a graph showing the impedance of the currently highlighted bead versus frequency. It is this curve that determines how a given bead responds on your board when terminating a signal.

If you are trying to find a model for a bead which is not specifically listed in BSW.FBD (see "About the Ferrite-Bead Models Supplied by HyperLynx" above for details on the BSW.FBD library), look through the HyperLynx-supplied models for the one whose impedance curve best matches the curve for the bead you're using. Key parameters are the peak impedance and the overall shape of the curve (is it fairly flat, or sharply peaked?).

Ferrite-bead data sheets normally show impedance-vs-frequency curves. If your data sheets don't include curves, contact your bead vendor and demand more information.

Ferrite beads are often described in terms of their impedance at a nominal frequency, usually 100 MHz. However, simply because two vendors' beads are both called "120-ohm" (both have approximately 120 ohms' impedance at 100 MHz) does not mean they behave the same in-circuit. Look at their complete impedance curves to determine how similar they actually are.

Simulating Before a Ferrite-Bead Model is Chosen

If you run a simulation before a model is chosen for a ferrite bead (when it still displays with a red question mark in the Pins list box), the ferrite bead is modeled as a 0.0-ohm resistor. This is equivalent to there being no bead present, or the bead being shorted out.

Copying Bead Models

Once you have chosen a model for one ferrite bead, it may be convenient to quickly "copy" the same model to other bead components.

Copying an Existing Model

To make a copy of a model:

1. In the Assign Models dialog box, in the Pins list box, highlight a ferrite-bead pin that has been assigned the model you want to copy.

2. Click the Copy button.

The information in the Model to Paste area updates to display the copied model. Also, the Paste and Paste All buttons become available (ungrayed).

The Model to Paste Information Area

The Model to Paste area shows the following information for the copied model:

- vendor
- part number

All information is for the last-copied model.

If you highlight a different ferrite-bead pin in the Pins list box, the information in the Model to Paste area remains the same. The Model to Paste area is a storage buffer: it always remembers the model you last copied.

Pasting to Another Ferrite-Bead Pin

To paste the last-copied model to another ferrite-bead pin:

1. Be sure that the model you want to paste is displayed in the Model to Paste area. (See “Copying an Existing Model” above in this section for details on copying a model.)

2. In the Assign Models dialog box, in the Pins list box, highlight the bead pin to which you want to copy the model.

3. Click the Paste button.

After a brief pause (while the model data is loaded into memory):

- the Pins list box updates; a green check mark appears next to the pin
- the model-information area updates; the fields display the new model's vendor and part number

Pasting to All Other Ferrite-Bead Pins

To paste the last-copied model to all other ferrite-bead pins:

1. Be sure that the model you want to paste is displayed in the Model to Paste area. (See "Copying an Existing Model" above for details on copying a model.)
2. Click the Paste All button.

BoardSim pauses while the model is loaded into memory for every ferrite bead on the net. When the operation is complete:

- the Pins list box updates
- a green check mark appears next to every bead pin

Changing Ferrite-Bead Models

To change a ferrite bead's model from one choice to another, simply re-choose the bead's model. Follow the steps in "How to Choose a Ferrite-Bead Model" above in this chapter.

Updating Models over the Internet

On computers that are connected to the Internet, BoardSim has the ability to automatically connect to HyperLynx's Internet site and update the IC models in your HyperLynx LIBS directory. This feature — called "Instant Online Models" — gives you an easy way to ensure that you always have HyperLynx's latest IC models.

Requirements for Accessing Online Models

In order to access HyperLynx's online models, your PC must have access to the Internet. This means that your machine must either have:

- a permanent Internet connection,

  OR

- dial-up Internet access, and you must dial in before attempting to access the models

Retrieving Models Online

To access the latest IC models from HyperLynx's Internet site:

1. Verify that your computer is presently connected to the Internet (dial first, if your access requires dial-up).

2. Verify that BoardSim is pointing to your model subdirectory: from the Options menu, choose Directories. Verify that the Model Library File Path is set correctly.

3. Then, from the File menu, choose Download Latest Models from HyperLynx.

4. The Accessing HyperLynx WWW Site dialog box opens; there may be a pause while a connection to HyperLynx's site is opened and verified. If the site cannot be accessed (e.g., your computer's Internet connection is "broken"), then after a brief time-out period, an error message will appear.

5. Once the connection to HyperLynx's site is established, the Download an Update dialog box opens. In the Select a Compressed File to Download list box are shown several compressed files that each contain the latest models of a particular kind. For example, the "ibis" file contains all of the latest IBIS models available from HyperLynx. For a description of what each file contains, see the File Descriptions area in the dialog box.

6. Choose a file to download by clicking once on the file in the list box to highlight it.

7. Click the Download button. After a brief pause, the Downloading File dialog box opens to show you the status of the download. When downloading is complete, the status dialog closes and a check mark appears beside the downloaded file; the compressed model file has been placed in your LIBS subdirectory (see step 2 above), uncompressed into multiple models, and deleted. The names of the model files appear near the top of the dialog box as they are uncompressed.

8. Repeat for additional model files.

9. When you are finished updating, click the Close button. Your IC models have been automatically updated.

If You Can't Download Models

If you try to download models but the operation never successfully completes (you get time-out or error messages), the feature is probably failing due to permission problems with a firewall. Unfortunately, there is little you can do to fix this, except trying from another site that is outside the firewall.

Retrieving Models Directly from Manufacturer Web and FTP Sites

Many IC manufacturers now make IBIS IC models available directly from their World Wide Web or FTP sites. While HyperLynx attempts to gather up and make available to customers as many of these models as possible, some manufacturers do not allow their models to be shipped with third-party products. Also, new models are appearing constantly.

Accordingly, you are strongly encouraged to browse manufacturer sites yourself to see what additional IBIS are available to you. For details on an easy way to do this, see Chapter 19, section "Using the HyperLynx IBIS-Model Web Page."

Chapter 9: Choosing ICs with a .REF File (IC AutoMapping)

Summary

This chapter describes:

- what choosing IC models with a ".REF file" means
- the format of the .REF file
- how to create a .REF file
- how to debug a .REF file
- how a .REF file relates to a session (.BUD) file

Automatic versus Interactive Selection of Models

BoardSim supports two methods of selecting IC models:

- interactively choosing models, as you simulate each net on your board or prepare for a batch-mode run
- automatically choosing models — "IC AutoMapping" — using an ASCII ".REF" mapping file, created before you load your board

This chapter describes the *automatic .REF-file* method of selecting IC models. The .REF-file method allows you to specify IC models in an ASCII file that you create before you load your board. Then each net's model choices are automatically loaded, based on the .REF-file choices, when the net is chosen for simulation (by you or the batch-mode analysis engine).

Chapter 8 describes the *interactive* method of selecting models. The interactive method allows you to select IC models as you need them, net-by-net and pin-by-pin as you simulate, or as you set up for batch mode.

For details on the trade-offs between the automatic and interactive methods, see Chapter 8, section "Interactive versus Automatic Selection of IC Models" for details. In summary, the interactive method requires no setup and loads models, but may be tedious if you plan to simulate a large number of nets because modeling occurs *pin-by-pin*. The automatic .REF-file method requires setup, but eliminates the need to manually choose models as you simulate, and loads models *component-by-component*.

You can also mix the two methods — see Chapter 8 and "Mixing .REF-File and Interactive Methods" below in this chapter for more details.

What Is the AutoMapping ".REF File?"

The .REF file is an ASCII file that you create before you load your board into BoardSim. BoardSim has a "smart" editor that makes creating a .REF file easy. The .REF file specifies a set of mappings between IC reference designators and the models that should be used to represent the ICs. These mappings are used by BoardSim to automatically load models when a net is chosen for simulation.

Note: *If you are unsure as to what is meant by "models" in BoardSim, see Chapter 8, section "Kinds of Interactive Component Choosing and Editing" for details.*

The .REF file does not support mappings for ferrite beads or other passive components. Bead models and passive-component values must be chosen

interactively; see Chapter 8, sections “Interactively Editing Rs, Cs, and Ls” and “Choosing Ferrite-Bead Models” for details.

How the .REF File Works

When BoardSim loads a .HYP file, it looks in the directory in which the .HYP file is located to see if there is also a file called <*HYP_file_name*>.REF (where <*HYP_file_name*> is the name of your board’s .HYP file). If there is, BoardSim reads the information in the .REF file while the .HYP file is loading.

Then, when a net is chosen for simulation (interactively or in batch mode), BoardSim looks in the .REF file’s list to see if any of the ICs on the net are mentioned in the list; if so, those ICs’ models are loaded automatically. If all of the net’s ICs are mapped in the .REF file, you do not need to make any manual/interactive model choices before running simulations (with one possible exception: see “Buffer Direction of Pins Loaded from a .REF File” below).

Note: *If an IC’s model for a particular pin is specified in both the .REF file and in a session (.BUD) file, the session file’s information takes precedence. See “How .REF-File Models are Overridden by Session-File (.BUD) and Interactively Chosen Models” below in this chapter for details.*

Model loading occurs only the first time a net is chosen for simulation (unless a previous attempt at loading failed for some reason; see “.REF-File Errors” below for details). If you subsequently return to a net, its models are already in place.

IC Models and the .REF File

In BoardSim, IC models must always be specified by the user. You can specify models for the ICs on the nets you want to simulate either interactively (see Chapter 8) or using a .REF file, but one way or the other, you must choose IC models.

Note: *More exactly, models must be chosen for the driver IC on each net before the net is simulated in detail in BoardSim’s interactive oscilloscope or*

spectrum analyzer, or by the batch-mode analysis engine. On the other hand, you can run the Board Wizard in "Quick Analysis" mode to get a fast signal-integrity scan of your entire board without specifying models; the Board Wizard uses a default driver switching time when no models are present. Detailed simulations, though, require at least a driver-IC model. See Chapter 16 for details on the Board Wizard.

A big difference between how IC models are chosen interactively versus in the .REF file is summarized in the following table:

Interactively, IC models...	...are chosen pin-by-pin
In the .REF file, IC models are chosen...	...component-by-component

Interactively, you choose IC models pin-by-pin, for each net you simulate or set up for batch-mode analysis. For example, if you simulate net FOO, and in the process choose IC model 74AC11XX:GATE for pin 1 of component U1; then simulate net FOO2, which is connected to pin 2 of U1, you must still choose a model for pin 2: the choice of model 74AC11XX:GATE for pin 1 applies only to that pin, not to the whole component.

On the other hand, when you use the automapping .REF file, you choose models component-by-component. For example, when you map U1 to model 74AC11XX:GATE; then simulate net FOO which is connected to pin 1 of U1, the 74AC11XX:GATE model is automatically loaded for pin 1; if you simulate FOO2 connected to pin 2 of U1, the model is also loaded for pin 2: the choice of 74AC11XX:GATE applies to the entire U1 component.

For this reason, the .REF file is more-powerful and efficient method of loading IC models. However, if you plan to simulate only a small number of nets on your board, the overhead of creating and debugging a .REF file may be too large, and interactive loading of IC models perfectly fine.

Note: *For details on IC models, e.g., what "74AC11XX:GATE" means, see Chapter 8, section "IC-Model Formats" and following.*

Buffer Direction of Pins Loaded from a .REF File

When a pin's IC model is loaded from a .REF file, as much information as possible about the model is set automatically. The only parameter which you may still need to edit manually is the model's buffer direction. (For details on how to set a model's buffer direction, see Chapter 8, section "Setting IC Buffer Direction.")

Specifically, IC pins modeled from an IBIS or .PML library will be loaded with the correct buffer direction unless the pin's model is bi-directional; then, BoardSim will default the model to buffer direction "receiver," and let you change it if the pin is actually driving.

IC pins loaded from a .MOD library will always default to buffer direction "receiver," since .MOD models do not contain information about pin directionality. Therefore, for .MOD models, you must always change the driving pin's buffer direction to "driver" manually.

Note: *For details on the IBIS, .PML, and .MOD IC-model formats, see Chapter 8, section "IC-Model Formats."*

The following table summarizes the relationship between model type, model direction, and buffer direction:

IBIS/PML model, type "output"	Loads as a driver; no need to change manually
IBIS/PML model, type "input"	Loads as a receiver; no need to change manually
IBIS/PML model, type "bi-directional"	Loads as a receiver; may need to change manually to driver
.MOD model (any)	Loads as a receiver; may need to change manually to driver

Format of the AutoMapping .REF File

The automapping .REF file is an ASCII file that you create with a text editor. The easiest editor to use is HyperLynx's .REF File Editor, a "smart" editor that is built-in to BoardSim. However, you can use any text editor.

The following sections describe the format of the .REF file. If you use the .REF File Editor, the lines in the file are created for you, so you may not need to know the exact syntax.

Note: *The .REF file cannot contain binary characters, and should be in DOS carriage return / line feed format, not UNIX.*

Example File

The .REF format is very simple, as the following brief example shows:

```
* DEMO.REF - this is a comment line
U1, 701V.IBS, CGS701V
U7, 74AC.PML, 74AC161_SOIC
U9, GENERIC.MOD, 74HCTXX:GATE-2
```

This example .REF file does the following:

- specifies that IC U1 is modeled as a CGS701V component from the IBIS library file 701V.IBS;
- specifies that IC U7 is modeled as a 74AC161_SOIC component from the PML library file 74AC.PML;
- specifies that IC U9 is modeled as 74HCTXX:GATE-2 components from the .MOD library file GENERIC.MOD

Format Rules

The following rules describe the .REF format:

- The .REF file is an ASCII file; it must contain no binary characters
- .REF files are case-insensitive
- Comment lines begin with an asterisk (*) in the first column
- Each model record must be on a single line, with fields separated by commas
- Any kind of white space between fields is allowed
- To specify a .MOD IC model, use a line of the form:

  ```
  <reference_designator>, <library.MOD>, <model_name>
  ```

- To specify a .PML IC model, use a line of the form:

  ```
  <reference_designator>, <library.PML>, <component_name>
  ```

- To specify a .IBS IC model, use a line of the form:

  ```
  <reference_designator>, <library.IBS>, <component_name>
  ```

The .REF file format is deliberately simple, to make it easy for you to create.

The format file is “comma delimited” (meaning that the parameters on each line are separated from one another by comma characters), since many other software packages can write data out in this format. This means that if you have bill-of-material data about the components on your board in another program (e.g., a database program), you may be able to create a template for your .REF file by writing that data out in comma-delimited form.

Note: *Examples of other programs in which you might have your component data, and which could be capable of writing it out in comma-delimited form: a spreadsheet application (e.g., Excel), a database program (e.g., Access or Oracle), or your PCB-layout tool (if it has a flexible report-generating feature).*

Library files are specified with a name *excluding* file path. The path is assumed to be whatever is specified in the Model Library File Path box in Set

Directories dialog box (see Chapter 4, section “Setting Directories” for details on specifying the library directory).

For .MOD IC models, you specify a model name that applies to the entire IC component; the model name must be present in the specified library file. For IBIS and .PML models, you specify a component name; the component name must match a component described in the specified library file.

Creating a .REF File

Requirements for Creating a .REF File

You can create a .REF file with any editor you want, as long as it:

- conforms to the proper format description (see “Format of the AutoMapping .REF File” above for details), ***AND***
- has the file name <HYP_file_name>.REF, where <HYP_file_name> is the name of your board’s .HYP file, ***AND***
- is located in the same directory as the .HYP file it accompanies

Opening The .REF File Editor

The easiest way to create a .REF file is to use HyperLynx’s .REF File Editor. This is a text editor (based on the general-purpose HyperLynx File Editor) that adds “smart” features that make creating and maintaining a .REF file particularly simple.

To open the .REF File Editor:

1. First, load the board for which you’re creating a .REF file into BoardSim (from the File menu, choose Open BoardSim File).
2. Then, from the Edit menu, choose IC AutoMapping (.REF). The .REF File Editor opens automatically on the file <board_name.REF>, where <board_name> is the name of your board’s .HYP file.

The reason you must first load your board's .HYP file into BoardSim is that the .REF File Editor needs to know detailed information about your board, e.g., what reference designators are present on the board. The Editor can only be run with a board loaded.

Elements of the .REF File Editor

In the .REF File Editor are the following elements:

- Reference Designator/Part Name list box — lists all of the reference designators on the currently loaded board, and the part name (if available) for each reference designator
- Text area at the bottom of the editor — an editor window, in which you can type; this displays the current contents of the .REF file
- Model to Insert area — allows you to choose a component/model to associate with the currently highlighted reference designator(s); the Library and Component/Model list boxes access all the models available in the LIBS subdirectory
- Information on Selected Device area — gives vendor-supplied information on IBIS and .PML models; unused for .MOD models
- Menus and toolbar — provide standard text-editor features; same functionality as in the HyperLynx File Editor

About Part Names

The Part Name data shown in the Reference Designator/Part Name list box is taken from the NAME= fields in the .HYP file's DEVICE section. The NAME field is intended to allow PCB-layout translators to record whatever information exists in the layout database about a part's name; the information is shown in the .REF file only as a "comment" or display field that might help you identify (or remember) which device a particular reference designator refers to. BoardSim does NOT read or use the Part Name data.

Automatically Adding Lines to the .REF File

A .REF file basically consists of multiple lines each one of which maps a reference designator to a component or model in a library. While these lines can be typed manually into the file (see "Format of the AutoMapping .REF File" above for a description of the .REF-file syntax), it is easier to have the .REF File Editor create the lines for you. This has two advantages:

- you are guaranteed not to create syntax errors
- you can browse in the .REF File Editor to find library and component/model names (rather than having to remember them)

To create a new mapping line in the .REF File:

1. In the Reference Designator/Part Name list box, click to highlight the reference designator to which you wish to attach an IC model. If you want to attach the same model to multiple reference designators, you can highlight more than one by using the standard Windows multiple-selection features (Shift-click and Ctrl-click).

2. In the Model to Insert area, pull down the Library combo box and choose the desired model's library.

3. Pull down the Component/Model combo box and choose the component or model you want. For IBIS and .PML libraries, you choose a specific component; for .MOD libraries, you choose a model. When you choose a component, the Notes combo box above in the Information on Selected Device area shows any model-specific information supplied by the vendor (not available for .MOD models).

4. In the text area at the bottom of the Editor, click to position the cursor where you want the new line of information to appear in the .REF file. If you click at the beginning of an existing line, the new line will be inserted before the existing line. If you highlight an existing line, the new line will overwrite the old.

5. Click the Paste Model(s) button. The new line appears, with correct syntax and the chosen reference-designator/component mapping.

How to Search for an IC Model

If you don't know in which library to find an appropriate device model for a particular reference designator, BoardSim includes a model-finding feature that may help. The model finder accesses in a spreadsheet a database listing of all models available in HyperLynx's shipping software; you can use this database to search for component names, look for all of the IC models from a particular vendor, and so forth.

To open the model-finding database:

1. From inside the .REF File Editor, click the Find Model button. After a brief pause, the IC Model Finder dialog box opens. (The pause occurs because the database of available IC models is very large, and takes some time to load.)

2. If desired, adjust the size of the finder's spreadsheet window to be larger, by grabbing it with the mouse on an edge or in a corner and dragging it to the desired size.

The model-finder spreadsheet lists all of the component models available when HyperLynx last compiled its libraries of device models (thousands of entries). You can search this list for particular components or device names, and sort on various criteria to group together, e.g., all ICs from a particular vendor, all recent models, etc.

To search for a text string in the model-finder spreadsheet:

1. In the Search String box, type the text for which you want to search.

2. Click the Search for String button. After a pause, all of the items matching the string (in any data column) are check marked in the far left-hand column and brought to the top of the spreadsheet.

To sort the spreadsheet based on a column's data:

1. Click the header button (at the top of the column). After a pause, all of the items in the spreadsheet are sorted in ascending order of the column's data.

2. To sort in the opposite order, click the column's header button again.

For example, to bring to the top of the spreadsheet the most-recent models in the database, click once on the File Date column's header button; after a pause, the list is sorted from oldest to newest model (ascending order). Click again, and the list changes to newest-to-oldest order (descending order).

Selecting a Model Directly from the Model Finder

If you find in the model-finder spreadsheet a model that you want to use in the .REF File Editor, you can select it directly from the spreadsheet.

To choose a model directly from the spreadsheet:

1. Click once on the model's line in the spreadsheet, to highlight it.

2. Click OK. The model-finder dialog box closes and the model is automatically selected in the Library and Component/Model combo boxes.

Manually Editing the .REF File

Although the .REF File Editor has features to automatically add reference-designator/component mappings, you can also type mappings into the file manually, just like you would with any other text editor.

To manually edit the .REF file:

1. In the text area at the bottom of the Editor, click to position the cursor where you want to type.

2. Begin typing.

The advantages to using the Editor's automatic line-adding features rather than manual entry (see "Automatically Adding Lines to the .REF File" above for details) are that you are guaranteed no syntax errors, and that it's easy to choose library-component/model combinations that are valid and available.

Cutting, Copying, Pasting, and Deleting Text

The .REF File Editor supports cutting, copying, pasting, and deleting of text, from the menus, from the toolbar, or using the standard Windows accelerator keys.

To cut text:

1. In the text area at the bottom of the Editor, highlight the text you want to cut.
2. Choose Cut from the Edit menu, or click the Cut button on the toolbar, or type Ctrl-X to cause the text to disappear.

To copy text:

1. Highlight the text you want copy.
2. Choose Copy from the Edit menu, or click the Copy button on the toolbar, or type Ctrl-C to copy text to the Clipboard.

To paste text:

1. Position the cursor where you want the text to be pasted.
2. Choose Paste from the Edit menu, or click the Paste button on the toolbar, or type Ctrl-V.

The pasted text is inserted at the cursor location.

To delete text:

1. Highlight the text you want to delete.
2. Choose Delete from the Edit menu, or press the delete key.

The text disappears.

You can also delete text by clicking in the text to position the cursor, then typing with the Backspace key or Delete key.

Undoing an Action

The .REF File Editor provides a single-level “undo” feature.

To undo the previous editing action:

1. From the Edit menu, choose Undo.
   OR
   Type Ctrl-Z.

The last editing action you performed is undone.

Going to a Line Number

To go to a line number:

1. From the Search menu, choose Go To Line.
   OR
   Click the Go To button on the toolbar.
   A dialog box opens.

2. Type the line number to which you want to go.

3. Click OK.

The editor jumps to the specified line number, with the matching line appearing at the top of the window.

Finding Text

To find text:

1. From the Search menu, choose Find.
   OR
   Click the Find button on the toolbar.
   OR
   Type Ctrl-F.A dialog box opens.

2. Type text you want to find.

3. Click OK.

The Editor jumps to the first occurrence of the specified text (or gives an error that no matching text could be found). The line with the matching text appears at the top of the window.

To find the next occurrence of the same text:

1. From the Search menu, choose Find Next.
   OR
   Click the Next button on the toolbar.
   OR
   Press F3.

The editor jumps to the next occurrence of the text (or gives an error that no matching text could be found). The line with the matching text appears at the top of the window.

Printing a File

To print the .REF file that is open in the editor:

1. From the File menu, choose Print.
   OR
   Click the Print button on the toolbar.
   A dialog box opens.

2. Change options, if needed, in the dialog box.

3. Click OK.

Saving a File

The .REF File Editor opens automatically on the file called <board_name.REF>, which is the file name required for working with board file <board_name.HYP>. When you have created or modified the .REF file and are ready to save it, the Editor is prepared automatically to save to the proper file name.

To save the .REF file:

1. From the File menu, choose Save.
   OR
   Click the Save button on the toolbar.

You can also save to a different file name, although this is not recommended, since <board_name.REF> is the only file name that will work with a board called <board_name.HYP>.

To save the file under a different name (not recommended):

1. From the File menu, choose Save As.
2. Type the new file name, then click Save.

Closing a File

To close the .REF file that is open in the editor:

1. From the File menu, choose Close.
   OR
   Click the Close button on the toolbar.

If you have edited the file without saving it, you are prompted to save before the file closes.

Exiting the Editor

To exit the Editor:

1. From the File menu, choose Exit.
   OR
   Click the Exit button on the toolbar.

If you have edited the file without saving it, you are prompted to save before the Editor closes.

Debugging a .REF File

Particularly if you create a large .REF file, you may accidentally introduce errors into the file which need to be corrected before the .REF file works perfectly. In order to make this process easier, BoardSim warns you when it finds errors in a .REF file, and allows you to change the file without having to re-load your .HYP file.

.REF-File Errors

There are two types of errors in a .REF file:

- errors that are reported when the .REF file is read (immediately after the .HYP file is loaded)
- errors that are reported only when a related net is selected

Errors Reported when the .REF File is Read

When the .REF file is read (while the .HYP file is being loaded), any lines in the file that are syntactically incorrect (e.g., that are missing parameters) are flagged immediately: when the .HYP file is finished loading, a Warning dialog box opens, listing the first such error found in the file.

Note: *You will rarely, if ever, have syntax errors in your .REF files if you create them with the "smart" .REF File Editor, because the Editor creates each mapping line in the file for you — and always uses the proper syntax. For details on using the Editor, see "Creating a .REF File" above.*

These kinds of errors are not "fatal," i.e., you do not need to re-load your board, but they generally indicate that the lines below the erroneous one in the .REF file have been discarded.

For example, in the following .REF file...:

```
*Line 2 is bad
U1, generic.mod, 74ac11xx:gate
x
U7, GENERIC.MOD, 74HCTXX:GATE-2
U9, GENERIC.MOD, 74HCTXX:GATE-2
```

...line 1 is read and will take effect when a net connected to U1 is selected for simulation, but because line 2 has a syntax error, the remaining lines will not be read. When the .HYP file corresponding to this .REF file is finished loading, a dialog box will open, warning that line 2 has an error.

See “Fixing a .REF File” below in this chapter for details on how to rectify these kinds of errors.

Errors Reported when a Net is Selected

Lines in a .REF file can be syntactically correct, but still erroneous because the information they contain is not valid. For example, an IC-modeling line could have the correct syntax, but refer to a library or model which cannot be found.

During interactive simulation, BoardSim does not report an error until the information in the line is actually used, i.e., until a net that causes the line to be “exercised” is simulated. Then, BoardSim opens the Warnings Found Loading .REF File dialog box, and reports a list of the errors found while trying to load models for the just-selected net.

For example, in the following .REF file...:

```
*Two errors in this file
U1, generic.mod, foo
U7, GENERIC.MOD, 74HCTXX:GATE-2
U9, GENERIC.FOO, 74HCTXX:GATE-2
```

...assuming that there is no model “foo” in library GENERIC.MOD and that net CLK connects to components U1 and U9, when net CLK is selected and interactively simulated, the dialog box opens and reports two errors (on lines 2 and 4). Line 2 is erroneous because the model “foo” cannot be found; line 4 because GENERIC.FOO is not the name of a .MOD, .PML, or .IBS library.

The behavior in batch mode is different. There, it is not permissible to interactively flag .REF-file errors (because the batch run must continue without your needing to manually intervene), so the analysis engine suppresses .REF-file errors. Instead, the results in the batch-mode report file will indicate errors, like nets for which driver-IC models were not found, etc.

Again, these errors are not “fatal,” but they do indicate that some of the models that were specified in the .REF file have not been loaded for the selected net. These errors occur line-by-line and are independent of other such errors in the .REF file. For example, in the preceding .REF file, if net CLK connects to components U1, U7, and U9, no models would be loaded for U1 or

U9, but U7's model would load correctly: each line is independent of the others.

This kind of when-the-net-is-simulated error can have numerous causes:

- invalid or non-existent IC-model library specified
- valid IC-model library specified, but BoardSim's Model Library File Path is set incorrectly (see Chapter 4, section "Setting Directories" for details on setting the library path)
- invalid or non-existent model name specified (.MOD models only)
- invalid or non-existent component name specified (IBIS or .PML models only)
- valid IBIS or .PML library and component specified, but component's pin names do not match pin names of actual component on your board

An IBIS or .PML model that is invalid because the component's pin names do not match the pin names on your board tends to generate many errors as various nets are selected. .MOD models generate potentially fewer errors, since no pin-name matching occurs.

See "Fixing a .REF File" below for details on how to rectify these kinds of errors.

Fixing a .REF File

You have several options when errors are reported from a .REF file:

- ignore the errors, and interactively load the models that failed to load automatically from the .REF file
- fix the .REF file to eliminate the errors; re-save the file; and continue working in BoardSim

To make the second option (fixing the .REF file) easier, *BoardSim does not require you to re-load your board after editing and re-saving the .REF file.*

Instead, BoardSim detects when the .REF file has changed; automatically re-loads it; and continues, using the information from the new version of the file.

Specifically, BoardSim re-loads models from an edited .REF file when you:

- re-select a net
- run a simulation
- run the Terminator Wizard, Board Wizard, or other type of analysis
- remove an IC model (in the Assign Models dialog box, using the Remove button)

For example, suppose you specified an IC model incorrectly by mistyping the model's name. When you select a net which connects to the IC, a dialog box reports the error and no model is loaded for the IC's pin. If you then edit the .REF file and fix the error; re-save the file; and re-select the net or simulate, the new .REF file's information is used, and the IC's model is loaded.

How .REF-File Models are Overridden by Session-File (.BUD) and Interactively Chosen Models

IMPORTANT! *This topic — how the models in the .REF file are overridden on a pin-by-pin basis by models in a session (.BUD) file or models you've chosen interactively — is the source of potential confusion when debugging a .REF file.* ***If you plan to create .REF files large enough to require any serious amount of debugging, read this section!***

Models specified in a .REF file take lower priority than models specified manually in BoardSim's user interface, or models recorded in a session (.BUD) file. This allows you to override, pin-by-pin, any models specified in a .REF file by interactively re-choosing them with BoardSim's user interface; any such overrides are stored in the subsequent session file and still take precedence.

Effectively this means that pins whose IC models are interactively chosen are "disconnected" from the .REF file; they no longer respond to the .REF file, even if the .REF file is edited or changed.

Exception: Removed Models

If an IC model is interactively chosen for a pin (therefore disconnecting the pin from the .REF file), but is subsequently removed (with the Remove button), the pin is re-connected back to the .REF file. This gives you a way to "undo" interactive edits on particular pins, so that the pins once again respond to the .REF file. (See Chapter 8, section "Removing IC Models" for details on removing an IC model.)

When you remove an IC model, if there is a .REF file present, the .REF file's model takes effect immediately. Therefore, when you remove a model, you may see it change to another model (the .REF file's) almost instantly.

Session-File Example

If you specify in a .REF file that U1 is model 74AC11XX:GATE; simulate with it on net FOO and decide to change the model to 74AC11X:LINE-DRV; close your board or BoardSim and save your edits into a session file; then re-load your board, when you re-simulate net FOO, the model 74AC11X:LINE-DRV will be loaded, since the model in the session file takes precedence over that in the .REF file.

However, this override occurs only for net FOO, since the session file applies models pin-by-pin, unlike the .REF file which works component-by-component. Other nets connected to U1 will use the model specified in the .REF file.

You can "undo" the session file's override by removing the 74AC11X:LINE-DRV model. This will re-connect the pin on net FOO to the .REF file.

Interactive Model Example

If you specify in a .REF file that U1 is model 74AC11XX:GATE; simulate with it on net FOO and decide to change the model to 74AC11X:LINE-DRV; simulate; edit your .REF file and change U1 to yet another model (e.g., 74ALSXX:GATE); then re-select net FOO, when you re-simulate, the model 74AC11X:LINE-DRV will still be loaded, since models that are chosen

interactively take precedence over models specified in the .REF file. In fact, after you've manually edited a particular IC pin, it will no longer load from the .REF file, *unless* you remove it with the Remove button.

Note that this override occurs only for net FOO, since the session file applies models pin-by-pin, unlike the .REF file which works component-by-component. Other nets connected to U1 will use the model specified in the .REF file.

Mixing .REF-File and Interactive Methods

You can mix the .REF-file method of selecting IC models with the interactive method. For example, you could create a .REF file from which to automatically load IC models for some of the most important ICs on your board (the ones connected to nets you think are most important to simulate); then rely on interactive model loading for ICs connected to other nets you later decide to also simulate, interactively. You can also mix the .REF-file and interactive model-choosing methods in preparation for batch-mode simulation.

If you interactively select models that are also specified in the .REF file, the interactive selections take precedence. See "How .REF-File Models are Overridden by Session-File (.BUD) and Interactively Chosen Models" above for details.

Disabling a .REF File

If you have a .REF file which you want to temporarily "disable" (i.e., temporarily have its model selections ignored), rename it or move it out of the .HYP-file directory, then re-load your board. Later, when you want to re-apply the .REF file, rename it or move it back.

Chapter 10: Editing IC and Ferrite-Bead Models

Summary

This chapter describes:

- how to edit a .MOD model
- how to edit an IBIS model
- where IC model libraries are stored, and how to point to them
- how to create your own ferrite-bead models

Editing .MOD IC Models

IC models created in the .MOD format can be edited directly inside BoardSim, with the .MOD model editor. If you are unfamiliar with the .MOD format or how it compares to the IBIS modeling format (another type of IC model supported by BoardSim), see Chapter 8, section “IC-Model Formats.” (A third supported format, .PML, is an extension of .MOD that adds component pin-out and package characteristics.)

The .MOD Format and Editor

The .MOD format is an ASCII format first created by HyperLynx in 1989 for its LineSim Pro product, and supported now in BoardSim. Though .MOD models are saved into ASCII libraries, the format is not keyword-based and therefore difficult to edit directly with a text editor.

Instead, BoardSim includes a .MOD model editor that allows you to edit models in a Windows dialog box. The editor allows you to modify existing models and create new ones; new models can be saved into user-defined libraries. .MOD models are based on parameters commonly found in IC databooks, to make the models as easy as possible to create and support.

.MOD Model Parameters

This section describes the parameters that together make up a .MOD model. All of the parameters can be edited by the user.

Output Vs. Input

Every .MOD model contains *both* output driver and input receiver information. This allows a single .MOD model to describe a typical output and input for an IC or family of ICs.

Note: *For ICs with multiple output buffer or input buffer types, multiple .MOD models can be saved into one IC- or family-specific library. If you need to model only an output or only an input, the other half of the model (e.g., the input if you are modeling the output) can be ignored.*

Output Driver Parameters

This section describes the parameters that make up the output-driver half of a .MOD model. See “Editing a .MOD Model” below in this chapter for details on how to change these parameters in the .MOD model editor.

Transistor Type

Transistor type describes the basic technology type of an output-stage transistor (e.g., Schottky-clamped bipolar or CMOS FET). You can customize a transistor's model further with other parameters like “on” resistance and slew time.

The following table lists the valid transistor types:

Type	Description
CMOS	a CMOS FET
silicon	a fully saturating bipolar transistor
Schottky	a Schottky-clamped bipolar transistor
ECL	an ECL emitter-follower; set both the high and low stages to this for an ECL device, even though there is really only one stage (the emitter)
open	nothing, i.e., no transistor at all
ramp	a special construct for simplifying a driver model; a resistance only, which switches between stages infinitely fast

Transistor "On" Resistance

Transistor "on" resistance describes the effective, fully-on impedance of the upper- or lower-stage transistor.

Hint: *This is the slope of the DC output-buffer V-I curve in the databook. It is NOT the resistance implied by the guaranteed worst-case DC currents, i.e., Ioh and Iol; these values are usually unrelated to the driver's dynamic switching characteristics and yield much too large a resistance.*

Slew Time

Slew time specifies the **10%-90%** switching time of the upper- or lower-stage transistor.

Offset Voltage

Offset voltage describes the effective offset for the upper- or lower-stage transistors from the rail voltage. It models the internal biasing of the driver.

Special Note about ECL

Even though ECL output buffers are actually referenced only to Vcc, the Low Offset Voltage is still interpreted in the .MOD editor as being from the low rail, in this case Vee. So, for example, to make an ECL driver switch to –1.55V when its Vee = -4.5, set the low-side offset voltage to +2.95 V.

Clamp-Diode Type

Clamp-diode type specifies the technology type of the upper- or lower-stage output clamp diode. You can customize a clamp diode's model further with the "on" resistance parameter.

The following table lists the valid diode types:

Type	Description
silicon	a silicon clamp diode
Schottky	a Schottky clamp diode

Clamp-Diode "On" Resistance

Clamp-diode "on" resistance describes any resistance effectively in series with the upper- or lower-stage clamp diode.

Hint: *This is the slope of the clamp-diode DC V-I curve in the databook. Sometimes, this data is found in a special section of the databook that covers ESD issues.*

Capacitance

The capacitance specifies the total output capacitance of the driver, including transistors and clamp diodes.

Default Power Supply

Default power supply specifies the default non-ground supply voltage off of which the driver is run. The default value applies only if BoardSim cannot find any power-supply nets connected to the driver's IC or if you explicitly choose to run the model off typical power-supply values; otherwise, a driver's supply

voltages are determined by the voltages to which its Vcc and Vss pins are connected. See Chapter 8, section “Setting the Vcc or Vss Pin” for details.

Measurement Thresholds and Loads

These parameters are used by the Board Wizard to calculate delay and related data. They do not affect a driver model's waveform, only how those waveforms are *measured* by the Board Wizard. They are unused in LineSim. The following table gives more detail.

Parameter	**Explanation**
Vmeasure	Voltage at which the driver is considered “switched,” i.e., when it transitions past this voltage, it has switched
Rload	The pull-up/down resistor used in the IC manufacturer's standard driver-output test load
Vload	The pull-up/down voltage used in the IC manufacturer's standard driver-output test load
Cload	The capacitance used in the IC manufacturer's standard driver-output test load

A model's Vmeasure value is displayed in both the Select IC Model dialog box (in the I/O Type area) and the Assign IC Models dialog box (in the Buffer Settings) area.

Note: *Most devices use standard values for these parameters (Vmeasure=1.5V, Rload=1000ohms, Vload=0V, Cload=50pF). They do not normally need to be changed from these values unless you are modeling ECL or a newer, low-voltage driver family like LVDS, GTL, etc.*
Vmeasure can also sometimes be calculated as:
(high input threshold + low input threshold) / 2.

Input Receiver Parameters

This section describes the parameters that make up the input-receiver half of a .MOD model. See "Editing a .MOD Model" below in this chapter for details on how to change these parameters in the .MOD model editor.

Input Resistance

Input resistance describes the effective resistance of the receiver's biased input stage.

Hint: *Generally, you can neglect input resistance for signal-integrity simulation, since it is normally a large value. The combination of input resistance and offset voltage should result in the input current specified in the databook.*
For CMOS, the input resistance is typically 1 Mohm or more; for signal-integrity simulation, 1 Mohm is sufficient.

Offset Voltage

Offset voltage describes the equivalent voltage of the receiver's input-stage biasing.

Hint: *The offset voltage is the open-circuit voltage of the input pin. For CMOS, this is typically Vcc/2.*

Clamp-Diode Type

Clamp-diode type specifies the technology type of the upper- or lower-stage input clamp diode. You can customize a clamp diode's model further with the "on" resistance parameter.

The following table lists the valid diode types:

silicon	a silicon clamp diode
Schottky	a Schottky clamp diode

Clamp-Diode "On" Resistance

Clamp-diode "on" resistance describes any resistance effectively in series with the upper- or lower-stage clamp diode.

Hint: *This is the slope of the clamp-diode DC V-I curve in the databook. Sometimes, this data is found in a special section of the databook that covers ESD issues.*

Capacitance

The capacitance specifies the total input capacitance of the driver, including transistors and clamp diodes.

Measurement Thresholds and Loads

These parameters are used by the Board Wizard to calculate delay and related data. They affect only how receiver-input signals are *measured* by the Board Wizard. They are unused in LineSim. The following table gives more detail.

Parameter	**Explanation**
Vih or Vih+	Receiver's primary high-going threshold; receiver is guaranteed to have recognized as a '1' any rising-edge signal that crosses this value
Vih-	Receiver's secondary high-going threshold; for devices with hysteresis, the high-going threshold backs down to this value after Vih+ is crossed; if no hysteresis, disable Schmitt Trigger check box, or set this value = Vih+
Vil+	Receiver's secondary low-going threshold; for devices with hysteresis, the low-going threshold backs up to this value after Vil- is crossed; if no hysteresis, disable Schmitt Trigger check box, or set this value = Vil-
Vil or Vil-	Receiver's primary low-going threshold; receiver is guaranteed to have recognized as a '0' any falling-edge signal that crosses this value

As indicated in the table, only two of the threshold values (Vih+ and Vil-) are needed for most devices; the other two (Vih- and Vil+) are important only for receiver inputs that exhibit hysteresis. If a device has no hysteresis, you can hide the "secondary" thresholds from view, or set them equal to the "primary" values.

To hide the "secondary" thresholds for devices that do not have hysteresis (i.e., whose inputs are not "Schmitts"):

1. In the Edit .MOD Model dialog box, in the Measurement Thresholds and Loads area, click off the Schmitt Trigger check box.

A model's input-threshold values are displayed in both the Select IC Model dialog box (in the I/O Type area) and the Assign IC Models dialog box (in the Buffer Settings) area.

Note: *Most devices use standard values for these parameters (Vih+ = Vih- = 2.0V, Vil+ = Vil- = 0.8V). They do not normally need to be changed from these values unless you are modeling ECL or a newer, low-voltage driver family like LVDS, GTL, etc.*

Editing a .MOD Model

To edit a .MOD model:

1. From the Edit menu, choose Databook IC Models.

The .MOD model editor opens.

Choosing a Model to Edit

The first step in editing a .MOD model is to choose the library and model.

To choose the model to be edited:

1. In the Library and Model area, pull down the Model Library combo box, and choose the library that the model you want to edit is in.

2. Pull down the Device Model combo box, and choose the model.

The libraries displayed in the combo box are the .MOD files in the directory pointed to by the Model Library File Path directory setting. Only libraries in this directory can be accessed. See Chapter 4, section “Setting the Model Library File Path” for details on setting the directory.

Choosing Driver or Receiver

Once the model is chosen, you must choose whether to edit its output-driver or input-receiver half.

To choose the output or input half of the model:

1. In the Library and Model area, click the Output or Input radio button.

If you change the radio-button setting, the contents of the .MOD-model-editor dialog box change to show different parameters.

Editing the Model

To edit the model:

1. Change the transistor types, diode types, and default power-supply voltage, if needed, by pulling down the combo boxes and choosing new values.

2. Change other parameters by typing new values into the boxes.

Editing both Driver and Receiver

You can edit both halves of a model (output and input) by editing one half; clicking the opposite radio button in the Library and Model area; then editing the other half.

Creating a New Model

To create a new model, start with an existing one. If you can find a model in GENERIC.MOD (or one of the other libraries that ships with BoardSim) that is similar to the model you want to create, choose that model and then edit it.

If you cannot think of a similar model in the supplied libraries, choose any model as a starting point and completely change its parameters, if necessary.

Once you have created the new model, save it; see "Saving a .MOD Model" below in this chapter for details.

Hint: *GENERIC.MOD or EASY.MOD almost always contain a model which is a good starting point for another, new model. Choose the closest match to the IC you're modeling, and change some parameters.*
As examples of starting with GENERIC.MOD, if the IC is CMOS and has an output slew time faster than 2 ns, start with a 74AC line driver. If it's CMOS with a slower slew rate, start with 74HC. If it's a bipolar IC, start with a 74AS line driver. If it's an ECL IC, start with 100K ECL.

Setting the Default Vcc or Vee Voltage

The default Vcc or Vee voltage applies only if BoardSim cannot find any power-supply nets connected to the driver's IC; see Chapter 8, section "Setting the Vcc or Vss Pin" for details.

BoardSim gives you a number of choices for default Vcc/Vee voltage. (You must use one of the pre-supplied choices; you cannot type your own value.) The second voltage in the pair is always 0.0V. For example, if you choose 3.0V, VCC=3.0V and VSS=0.0V; if you choose -5.2V, VCC=0.0V and VSS=-5.2V.

5.0V/0.0V are the default values for all non-ECL models; 0.0V/-4.5V and 0.0V/-5.2V are the defaults for the appropriate ECL models.

Saving a .MOD Model

After you have edited a model, you must save it into a library.

If you are changing an existing model, save it back into the same library and under the same model name as when you chose it. If you are creating a new model, save it into a different library and/or under a different model name.

Saving to the Same Library and Model Name

To save a model to the same library and model name:

1. In the .MOD-model-editor dialog box, click the Save button.

After a brief pause, the library file is updated.

Saving to a Different Library or Model Name

To save a model to a different library and/or model name:

1. In the .MOD-model-editor dialog box, click the Save As button. The Save .MOD Model As dialog box opens.
2. *If you are saving to a different library,* in the Library Name list box, choose or type the library.
3. *If you are saving to a different model name,* in the Model Name box, type the model name.
4. Click OK.

After a brief pause, the library file is updated.

Cannot Save into GENERIC.MOD or EASY.MOD

You can edit a model in the BoardSim-supplied libraries GENERIC.MOD and EASY.MOD, but you cannot save the edited model back into either library; you must save to a different library. The model editor will not write into GENERIC.MOD or EASY.MOD.

Note: *It often makes sense to edit a model in GENERIC.MOD or EASY.MOD as a starting point for creating a new model. Once you have loaded and modified the model, save it into an existing library of your own, or into a completely new library.*
Note: *If you modify a HyperLynx-supplied library, you should rename it first. If you do not change the library's name, your version of the library will be overwritten next time you receive updated HyperLynx software.*

Creating a New Library

To create a new library:

1. Follow the steps in "Saving to a Different Library or Model Name," except for the library name, type a completely new name.

When you type the library's name, the extension .MOD is optional; if you omit it, it is automatically provided.

Deleting a .MOD Model

You can delete models out of .MOD libraries. Before deleting a model, be certain that you never want to use it again: it will be removed from the .MOD file and lost.

Note: *If a deleted model is recorded in a session (.BUD) file, the pin that calls out the deleted model will not have a model when the board is re-loaded. BoardSim intentionally does not give warnings about models in the session file it cannot find, so that if a session file records a large number of models that do not any longer exist, you can still load the board. For pins that call out the missing models, you must re-choose models. See Chapter 14 for details on session files.*

To delete a .MOD model:

1. From the Edit menu, choose Databook IC Models.
2. In the .MOD-model-editor dialog box, pull down the Model Library combo box and choose the library that the model you want to delete is in.
3. Pull down the Device Model combo box, and choose the model.
4. Click the Delete button.
5. BoardSim asks if you are sure you want to delete the model; click OK if so.

Cannot Delete from GENERIC.MOD or EASY.MOD

You cannot delete a model in the BoardSim-supplied libraries GENERIC.MOD or EASY.MOD. If you try to, BoardSim gives an error message.

Note: *You can delete models from other .MOD libraries (libraries other than GENERIC.MOD and EASY.MOD) supplied with BoardSim. If you accidentally delete a model from a HyperLynx-supplied library, you can download the library from the HyperLynx Internet site — or re-install*

BoardSim. See "Updating Models over the Internet" below in this chapter for details.

Editing IBIS IC Models

IC models created in the IBIS format can be edited directly inside BoardSim, using the Visual IBIS Editor. If you are unfamiliar with the IBIS format or how it compares to the .MOD modeling format (another type of IC model supported by BoardSim), see Chapter 8, section "IC-Model Formats." (A third supported format, .PML, is an extension of .MOD that adds component pin-out and package characteristics.)

For details on using the Visual IBIS Editor, see "The Visual IBIS Editor" below. A shareware copy of the editor can be downloaded free of charge from HyperLynx's World Wide Web site; see Chapter 19 for details on the Web site.

Note: *Because most IBIS models come directly from the semiconductor manufacturer with guaranteed data, they are not typically edited by users. You may want to create your own IBIS libraries, however, to model custom devices, ASICs, etc. But before creating an IBIS model, consider whether the .MOD modeling format would meet your needs: it is simpler and easier to create models with. See Chapter 8, section "IC-Model Formats" for a comparison of the .MOD and IBIS formats.*

The Visual IBIS Editor

The Visual IBIS Editor is a program — available separately but also shipped with BoardSim and launchable from it — for creating, editing, verifying, and maintaining IBIS (I/O Buffer Information Specification) device models. The Editor includes a number of useful features for developers of IBIS models, including:

- a large-file Windows text editor
- an integrated IBIS-syntax-check utility

- a graphical viewer for looking at IBIS V-I tables and waveform tables
- an IBIS-file template generator
- a test generator that automatically creates a test schematic for any IBIS-model pin
- a complete, self-contained online Help system
- an online IBIS specification

HyperLynx developed the Editor to encourage the development of IBIS device models. The Editor's "smart" features make creating and verifying IBIS models much simpler than with IBIS-ignorant, standalone tools.

Opening the Editor

In BoardSim, you run the Editor by launching it from BoardSim's menu bar, then operating it from its own, standalone user interface.

To open the Visual IBIS Editor:

1. From the Edit menu, choose IBIS IC Models.

The Editor opens in its own window, ready to load or create an IBIS file.

In all of the sections below, the menus and toolbar buttons referred to are Visual IBIS Editor menus and buttons, not BoardSim's.

Editing an IBIS File

To open an existing IBIS file in the Editor:

1. From the File menu, choose Open.
   OR
   Click the Open button on the toolbar.
2. Choose the file you want to open.
3. Click OK.

The IBIS file appears in the Editor.

To open a file in read-only mode:

1. After opening the file, choose Read Only from the Options menu.

In read-only mode, any changes you attempt to make to the file in the Editor will be ignored.

There are two windows in the Editor:

- a main, upper window for viewing and editing IBIS files
- a secondary, lower window for seeing the results of running the IBIS syntax checker

The Editor works like most other Windows text editors. It includes support for standard Windows features like cut, copy, paste, and so forth; see the following sections for details.

The Editor handles files of arbitrary size. There are no artificial limits (like 32K or 64K) as with some Windows text editors. The Editor is ASCII only; it will not insert binary characters into a file.

Cutting, Copying, Pasting, and Deleting Text

The Visual IBIS Editor supports cutting, copying, pasting, and deleting of text, from the menus, from the toolbar, or using the standard Windows accelerator keys.

To cut text:

1. Highlight the text you want to cut.
2. From the Edit menu, choose Cut.
   OR
   Type Ctrl-X.
   OR
   Click the Cut button on the toolbar.

The text disappears.

To copy text:

1. Highlight the text you want copy.
2. From the Edit menu, choose Copy.
   OR
   Type Ctrl-C.
   OR
   Click the Copy button on the toolbar.

The text is stored on the Clipboard.

To paste text:

1. Position the cursor where you want the text to be pasted.
2. From the Edit menu, choose Paste.
   OR
   Type Ctrl-V.
   OR
   Click the Paste button on the toolbar.

The pasted text is inserted at the cursor location.

To delete text:

1. Highlight the text you want to delete.
2. Choose Delete from the Edit menu, or press the delete key.
   The text disappears.

You can also delete text by clicking in the text to position the cursor, then typing with the Backspace key or Delete key.

Undoing an Action

The Visual IBIS Editor provides a single-level “undo” feature.

To undo the previous editing action:

1. From the Edit menu, choose Undo.
   OR
   Type Ctrl-Z.

The last editing action you performed is undone.

Converting Tabs to Spaces

The Visual IBIS Editor has a feature which automatically converts tab characters to space characters. The IBIS specification allows tabs, but recommends against using them because different tools expand them in different ways.

To convert tab characters to space characters:

1. From the Edit menu, choose Convert Tabs to Spaces.

All of the tabs in the file are replaced with spaces.

Going to a Line Number

To go to a line number:

1. From the Search menu, choose Go To Line.
   OR
   Click the Go To button on the toolbar.
   A dialog box opens.

2. Type the line number to which you want to go.

3. Click OK.

The editor jumps to the specified line number, with the matching line appearing at the *top* of the window.

Finding Text

To find text:

1. From the Search menu, choose Find.
   OR
   Click the Find button on the toolbar.
   A dialog box opens.

2. Type text you want to find.

3. Click OK.

The editor jumps to the first occurrence of the specified text (or gives an error that no matching text could be found). The line with the matching text appears at the *top* of the window.

Searching always occurs from the top of the file, regardless of where the cursor is positioned when the search is requested.

To find the next occurrence of the same text:

1. From the Search menu, choose Find Next.
   OR
   Click the Next button on the toolbar.

The editor jumps to the next occurrence of the text.

Viewing an IBIS V-I or Waveform Table

One of the most-valuable features in the Visual IBIS Editor is the ability to graphically view V-I or waveform table data. Viewing table data graphically makes it much easier to find errors in the data, e.g., a mistyped number or bad sign.

Viewing a table is a two-step process:

- first, you choose a signal or pin in the IBIS file whose model's tables you want to view
- then, you choose exactly which table you want to see, and you view it

To choose a signal or pin:

1. From the IBIS menu, choose Select Signal or Pin.
   OR
   Click the Select button on the toolbar.
   The Select Model dialog box opens.

2. In the Devices list box, click once to highlight the device that the signal or pin is on.

3. In the Signal list box, highlight the signal you want. If you prefer to choose by pin name, first click the Pin radio button in the Select By area, then highlight the pin.

4. Click OK.

To choose and view a table:

1. From the IBIS menu, choose View Data for Selected Pin.
   OR
   Click the Views button on the toolbar.
   The viewing dialog box opens.

2. Click on the tab for the table you want to view.

The graphical display shows the table's minimum, typical, and maximum curves, if available, each in a different color. The display scales itself automatically to best fit the table's data.

Viewing Rising/Falling Waveform Tables

When you click the Rising Waveform or Falling Waveform tab, to view a model's V-t table(s), an extra combo box labeled "Conditions" appears. This allows you to choose amongst multiple waveform tables for viewing, if there is more than one table in the model.

Validating an IBIS File's Syntax

Another valuable feature in the Visual IBIS Editor is the ability to check an IBIS file's syntax without ever leaving the Editor. This is accomplished by

running the official EIA-656 (IBIS) validation-checking program on the file that is currently being edited. You can run the check periodically as you create or edit a model to ensure that you haven't introduced errors.

To run the syntax validation check:

1. From the IBIS menu, choose Run IBIS Validation Check.
   OR
   Click the "IBIS" button on the toolbar (looks like a check mark).
   The validation-checking program is automatically launched.

The lower window in the Editor displays messages from the validation program. If a large number of errors appears, you can move through them with the scroll bar on the right edge of the window.

The validation checker automatically determines (from the [IBIS Ver] record in the file) whether to run a V1.x, V2.x, or V3.x syntax check.

If the file is error-free, the number of warnings and errors is reported as "0."

Creating a New IBIS Model

The Visual IBIS Editor gives you two kinds of assistance with creation of a new IBIS model. One is a template generator that creates a skeleton — complete with all of the required keywords — for an IBIS V1.1 or V2.1 model. The second is a more-powerful feature called the "Easy IBIS File Creation Wizard" that actually "interviews" you about the model's data, then automatically generates the corresponding model file for you.

For details about the template generator, see "Creating a Template for a New IBIS File" below. For details about the Easy IBIS Wizard, see "Running the Easy IBIS File Creation Wizard."

Creating a Template for a New IBIS File

The IBIS template generator automatically creates a template file that contains a "skeleton" IBIS model, in either V1.1 or V2.1 IBIS format. The template gives you a head start on creating model, because it reminds you of

required IBIS keywords, syntax, and so forth. The data in tables, etc. is "dummy" and should be replaced manually with the correct information.

To run the IBIS File Creation Wizard:

1. From the IBIS menu, choose Run IBIS Template Creation Wizard. A dialog box opens.
2. Use the radio buttons to specify the IBIS file version you want (V1.1 or V2.1).
3. Click the Next button, then Finish.
4. In the dialog box, type the name of IBIS file for which you want to create a template. Remember that IBIS files names are limited to DOS "8.3" format.

The template generator creates the template file and opens it in the Editor.

Running the Easy IBIS File Creation Wizard

The Easy IBIS Wizard is a powerful model-generation utility that interviews you about the characteristics of the component you want to model, then, based on the information you input, automatically generates a complete IBIS model, syntactically correct and ready-to-simulate.

To begin running the Easy IBIS Wizard:

1. With the Visual IBIS Editor open, from the IBIS menu, choose Run Easy IBIS File Creation Wizard.
   OR
   Click the "Easy IBIS" button on the toolbar.

The Easy IBIS Wizard opens, ready for you begin specifying the IBIS file you want created.

Entering Data in the Easy IBIS Wizard

About Data Entry

Entering Text Strings

The IBIS syntax specifies definite limits to the length of various user-created text strings. In the Easy IBIS Wizard, these length limits are enforced; if you try to type in a string that is too long, the Wizard will refuse to enter the extra characters and will "beep." This cues you to shorten the text string.

Entering Numerical Data

The IBIS syntax allows most numerical values without limits, but IBIS simulators will sometimes have trouble with absurdly large or small values. In the Easy IBIS Wizard, if you enter any such values on a page and click "Next," the Wizard will warn you that one or more values are unreasonable, and force you to re-enter them. This prevents you from generating a model that may not simulate properly in some simulators.

Entering the IC Component Name

The first page of the Easy IBIS Wizard prompts you for a component name for the model you're about to create. Every IBIS file can contain multiple buffer models that are tied together through a pin out to make up a "component"; a component models a complete IC. On its first page, the Wizard asks you for the name of the component, i.e., the IC name. This should generally be a fairly specific name; typically it might include manufacturer, device, and package information.

Entering Header Information

To advance to the Header Info page in the Wizard:

1. From the Wizard's first page, click Next.

At the top of an IBIS file, model developers are encouraged to include a file-creation date, file-revision number, and copyright. These help both the developer and users of a model track multiple revisions of the file. The copyright notice protects the model legally. The data in these fields is free form, although the revision number is normally a number.

Entering the Data Source

To advance to the Data Source page in the Wizard:

1. From the Wizard's Header Info page, click Next.

Near the top of an IBIS file, model developers are encouraged to specify — in some detail, if needed — the source of the model's data. In particular, users are interested in knowing whether the model was developed from measured or simulated data, and any other information pertinent to how the model was created. The data in this field is free-form, and is often multi-line. Use as much detail as is needed.

The Wizard gives a suggested starting phraseology, but its use is not required.

When the Easy IBIS Wizard writes out the data-source section, it will wrap it as needed to prevent from exceeding the 80-character line-length restriction imposed by the IBIS specification. This means that the text may not appear exactly as you enter it on this page of the Wizard.

Entering Notes

To advance to the User Notes page in the Wizard:

1. From the Wizard's Data Source page, click Next.

The [NOTES] section of an IBIS file can be used to add for the user any amount of clarifying detail about the model that is not already captured in the preceding fields. Typical information included here might be caveats about data missing from the model; descriptions of when the model is most accurate, and when it is not; and so forth. The data in this field is free-form, and is often multi-line. Use as much detail as is needed.

When the Easy IBIS Wizard creates the notes section, it will wrap text as needed to prevent it from exceeding the 80-character line-length restriction imposed by the IBIS specification. This means that the text may not appear exactly as you enter it on this page of the Wizard.

Entering a Disclaimer

To advance to the Disclaimer page in the Wizard:

1. From the Wizard's User Notes page, click Next.

The [DISCLAIMER] section of an IBIS file is primarily for use by semiconductor manufacturers. It is typically used to disclaim legal responsibility for the model's accuracy, suitability, and so forth. The data in this field is free-form, and is often multi-line. Use as much detail as is needed.

The Wizard suggests typical disclaimer phraseology, but its use is not required.

When the Easy IBIS Wizard writes out the disclaimer section, it will wrap it as needed to prevent from exceeding the 80-character line-length restriction imposed by the IBIS specification. This means that the text may not appear exactly as you enter it on this page of the Wizard.

Entering Additional Header Information

To advance to the Additional Header Info page in the Wizard:

1. From the Wizard's Disclaimer page, click Next.

The additional header information allows you to enter the name of the IC's manufacturer, and information about whom to contact regarding the model. The manufacturer name is required by the IBIS specification; the remaining fields are optional (although encouraged, since they give the user of the model someone to query if technical questions arise). Any reasonable entry can be made in the manufacturer field; if you are creating a model but do not work for the company that actually manufactures the silicon, enter your own company name.

When the IBIS file is generated, the contact information, if present, is preceded by the phrase "If you have comments concerning this file, please address them to:".

Entering Pin Count and Package Parasitics

To advance to the Part Pin Count page in the Wizard:

1. From the Wizard's Additional Header Info page, click Next.

Every IBIS file must contain a pin out table that lists the pins on the IC component being modeled, and connects each pin to a buffer model in the file (or specifies that a pin is a power pin or not connected). Every file must also contain a table defining the default package parasitics — Rpkg, Lpkg, and Cpkg — to be assumed for pins for which no pin-specific parasitic data is later specified. The pin out table must contain at least one pin (although in a complete model, it would contain an entry for every pin on the IC). The package parasitic data can be all 0.0, although doing so will omit package effects from consideration when the model is simulated.

Predefined versus User-Defined Packages

This page of the Easy IBIS Wizard allows you to specify either your own user-defined package, or start with a predefined one. You specify the number of pins on the package, or use a predefined package, whose definition includes not only a fixed number of pins, but also parasitic R/L/C data for the package.

The parasitics included in a predefined package are typical values for a package of that style (e.g., larger values for DIPs, smaller for SMD packages, etc.). The advantage to using a predefined package is that it comes "for free" with a complete list of pins and parasitics, meaning you have less data to enter than with a user-defined package. However, there may not be a predefined package that matches the IC you're trying to model, so you may need to create your own.

Using a Predefined Package

To select a predefined package:

1. Click the Predefined radio button. The list of predefined packages becomes available.

2. Scroll through the list of predefined packages and highlight the one you want to use.

Creating a User-Defined Package

To create your own user-defined package:

1. Click the User-Defined Package radio button. Several data boxes in the lower part of the Wizard page become available.

2. In the Number of Pins box, type the number of pins on the package you're defining.

3. In the Default Pin Parasitics boxes, enter the default values of R, L, and C to be used for modeling the bond wires and pins of the package. These values will be used during simulation for any pins that do not have pin-specific R/L/C values (specified later in the Wizard). The values can be 0.0 if you do not wish to simulate package parasitics (provides a less-accurate simulation, but is acceptable).

Entering Pin Data

To advance to the Pin Data page in the Wizard:

1. From the Wizard's Part Pin Count page, click Next.

The table in the IBIS file that lists the IC component's pins also includes, for each pin, an associated signal name and , optionally, a set of pin-specific package parasitics (Rpkg, Lpkg, and Cpkg). The package data is optional on a per-pin basis. If no pin-specific parasitics are supplied, then all pins will be simulated using the default package parasitics specified on an earlier Wizard page (see “Entering Pin Count and Package Parasitics” above). If pin-specific parasitics are supplied for some pins and not others, then the pins with parasitics will be simulated using the pin-specific values, and the pins without will use the default values.

Adding Signal Names

Every pin in an IBIS file is required to have an associated signal name. The Easy IBIS Wizard creates default signal names for every pin, but you should modify the names to match the signals on the IC you're modeling.

To add signal names to pins (by modifying the default name):

1. In the Pin list box, click once to highlight the pin whose signal name you want to modify.
2. In the Signal Name box, type the new pin name.
3. Repeat as needed for other pins.

Modifying Pin Names

If the IC you're modeling has a package with alphanumeric pin names (or if the default pin numbers are wrong in some way), you can modify the default names as needed.

To modify the default pin names:

1. In the Pin list box, click once to highlight the pin whose pin name you want to modify.
2. In the Pin name list box, type the new name of the pin. The pin's name changes in the Pin list box.
3. Repeat as needed for other pins.

Specifying Pin-Specific Package Parasitics

For any or all pins in the Pin list box, you can specify pin-specific package characteristics (R/L/C). If you are satisfied with the default package values entered on an earlier Wizard page, there is no need to supply pin-specific values. However, especially for some packages that have significantly different R/L/C values depending on pin position, you may want pin-specific data to increase simulation accuracy.

To specify pin-specific package parasitics for a pin:

1. In the Pin list box, click once to highlight the pin for which you want to specify parasitics.
2. In the Pin Parasitic Values area, click on the Customize check box. The R, L, and C data boxes become available.

3. Type the values of R, L, and C you want for the pin.

4. Repeat the steps above as needed for other pins.

You can mix pins that have and do not have pin-specific parasitics. As you highlight various pins in the Pin list box, the ones that have pin-specific values are evident because the Customize check box below enables and the R, L, and C data boxes become active.

Entering Min/Max Scaling

To advance to the Min and Max Scale Factors page in the Wizard:

1. From the Wizard's Pin Data page, click Next.

Throughout an IBIS file, data can be supplied either with a typical value only, or with typical, minimum, and maximum values. In the Easy IBIS Wizard, to keep entry of a model simple and yet still provide for minimum and maximum data, the values you enter are used as "typical," and then two scaling factors are used to generate min/max data automatically from the typical values. This is a reasonable approach, since ICs are subject to process variations that cause various circuit parameters to scale up and down.

Using this approach, if you have more exact min/max data and want to include it in your model, you can generate the IBIS file, then manually edit it to replace the min/max values created by the Wizard.

Most minimum and maximum values written by the Wizard are created with the scaling factors. An exception is the power-supply min/max values, which are automatically set to +/-10% of the typical value you enter.

To specify the minimum and maximum data scaling factors:

1. In the Min Scale Factor box, type the value to be used to create minimum data.

2. In the Max Scale Factor box, type the value to be used to create maximum data.

Creating Buffer Models

To advance to the Buffer Models page in the Wizard:

1. From the Wizard's Min and Max Scale Factors page, click Next.

The "core" element in an IBIS file is one or more buffer models. It is these models that actually produce the analog waveforms that signal-integrity simulators use to analyze transmission-line effects and other high-speed phenomena. The pin out table in an IBIS file connects pins/signals with underlying buffer models. In the Easy IBIS Wizard, once you have defined a pin out, you must next create models for the various buffer types present on your IC.

Almost any real IC has at least two buffer types, a basic output or I/O buffer, and an input buffer. Large, complex devices may have many different buffer types, for example, a very strong output buffer on clock-signal outputs, a medium-strength buffer used for address and data lines, and a relatively weak buffer used for non-critical signals. There might also be multiple input-buffer types, with different input capacitances or clamp-diode strengths, for example.

How the Wizard Models Buffers

In the IBIS specification, buffers are described with V-I tables. In the Easy IBIS Wizard, to simplify the data-entry process, these tables are generated from single-valued impedances plus saturation currents. The Wizard uses this data along with internal knowledge about transistor V-I curves for various technologies (like CMOS or TTL) to generate detailed V-I curves from the simplified impedance and saturation data.

In addition, output buffers have slew-rate data, which in the IBIS specification can either be input as a simple slew rate into a given resistive load, or in a more-complex V-t table. The Wizard uses the slew-rate/load method.

Pre-defined versus User-Created Buffer Models

The Easy IBIS Wizard allows you to either create your own custom buffers, or if you're in a hurry or don't know the exact characteristics of the buffer you're trying to create, base your model on a generic, technology-based buffer. Creating a custom buffer requires you to know the approximate driving impedance of the buffer, slew rate, and so forth. Basing a model on a pre-

defined buffer requires only that you know the technology type of the model (CMOS or TTL?) and the approximate slew rate (fast, medium, slow?). When you base a model on a pre-defined model, you can either use the pre-defined model "as is," or use it as a starting point but modify some of its characteristics.

Creating a New Buffer Model

To create a new buffer model:

1. Click the New Buffer button.
2. Follow the steps in the section below, "Specifying a New Buffer's Characteristics."

Creating a Buffer Model Based on a Pre-defined Model

To base a buffer model on a pre-defined model:

1. In the list box that shows the currently defined buffer models, click once to highlight the model you want to use as the base for your new model.
2. Click the Copy Buffer button. A copy of the model is created, at the bottom of the list box.
3. With the copied model still highlighted, click the Edit Buffer button.
4. Follow the steps in the section below, "Specifying a New Buffer's Characteristics."

Specifying a New Buffer's Characteristics

Setting Buffer Name, Technology, Type, and Operating Voltage

To advance to the Buffer Model – Operating Voltage page in the Wizard:

1. From the Wizard's Buffer Models page, click new Buffer or Edit Buffer (see "Creating Buffer Models" above for details).

To specify the buffer name, technology, type, Vcc voltage, and capacitance:

1. In the Buffer Name box, type a name for your new buffer.

2. In the Technology area, click on the radio button for either CMOS or TTL (i.e., bipolar), depending on the technology type of the IC you're modeling.

3. In the Buffer Type area, click on the radio button for the buffer type — input/ output/3-state — that you're modeling. "Open sink" is equivalent to "open drain" or "open collector."

4. In the Operating Voltage edit box, type the typical Vcc value for the IC.

5. In the Die Capacitance box, type the typical die capacitance for this buffer. "Die capacitance" means I/O capacitance (almost always given in the IC data sheet) minus the package capacitance. (2-8 pF is typical.)

Specifying Clamp Diodes

To advance to the Buffer Model – Clamp Diodes page in the Wizard:

1. From the Wizard's Operating Voltage page, click Next.

To specify the data for clamp diodes:

1. In the High Rail Clamp Diode area, pull down the combo box and choose a type for the high-side clamp diode:

   If the buffer has no high-side clamp diode, choose "None".

   If there is a clamp diode, choose between silicon and Schottky diodes (check the IC data sheet for which is correct), and choose an approximate "clamping strength" (strong, typical, weak).

   If you know little about the diodes, choose type "Silicon Typical."

   If you know in detail about the diode's characteristics, choose type "User Defined" (silicon or Schottky, as appropriate), then go to step 2.

   If you did not choose "User Defined", skip to step 3.

2. *If you chose a "User Defined" diode type in step 1,* in the On Impedance edit box, type the value of the clamp diode's effective on resistance.

3. Repeat step 1, but for the Low Rail Clamp Diode.

Specifying Pull-Up and Pull-Down Buffers

Important: *Pull-up data applies only if buffer is type Output, I/O, or 3-State; does not apply if Input or Open Sink.*

To advance to the Buffer Model – Output High Parameters page in the Wizard:

1. From the Wizard's Clamp Diodes page, click Next.

To specify the high-side driver-transistor parameters:

1. In the Rload data box, type the value of the resistance into which the driver's rising-edge switching characteristics are specified (check the data sheet). If you do not know the value, type "1000".

2. In the Slew Time data box, type the amount of time it takes the driver to slew from 20% to 80% of the final DC values (in ns, rising edge).

3. In the Slew Voltage data box, type the voltage difference between the 20% and 80% final DC values (in ns, rising edge).

4. In the Open Circuit Voltage box, type the voltage at which the driver "sits" when it is unloaded (no external load, when switched high). For standard CMOS outputs, this generally equals Vcc; for bipolar, NMOS, and "specialty" outputs, it is different than Vcc.

5. In the Saturation Current box, type the approximate maximum or saturation current of the transistor stage (in Amps). Every output design differs, but all technologies limit at some reasonable value.

6. In the Output Impedance box, type the approximate driving impedance of the buffer. If you do not know this value, read section "Determining Output Impedance" below.

To advance to the Buffer Model – Output Low Parameters page in the Wizard:

1. From the Wizard's Output High Parameters page, click Next.

Important: *Pull-down data does not apply if buffer type is Input or Open Source.*

To specify the low-side driver-transistor parameters:

1. Repeat steps 1-6 in the preceding section, except enter the data for the low-side transistor stage.

Specifying Input Thresholds

To advance to the Buffer Model – Input Levels page in the Wizard:

1. From the Wizard's Output Low Parameters page, click Next.

Important: *Applies only if buffer is type Input or I/O; does not apply if Output, 3-State, or Open Sink.*

To specify the input thresholds:

1. In the Logic High Threshold data box, type the value of the worst-case high-going threshold. For most devices, this is 2.0V; it may be higher for certain older CMOS families, like HC. (It may also be different for newer, very-low-Vcc devices.)

2. In the Logic Low Threshold data box, type the value of the worst-case low-going threshold. For most devices, this is 0.8V; it may be different for certain older CMOS families, like HC. (It may also be different for newer, very-low-Vcc devices.)

Specifying Output Polarity and Load Circuit

To advance to the Buffer Model – Input Levels page in the Wizard:

1. From the Wizard's Input Levels page, click Next.

Important: *Applies only if buffer is type Output, I/O, 3-State; does not apply if Input.*

To specify output polarity and manufacturer load circuit:

1. In the Output Polarity area, click on the radio button for the buffer's output polarity (true or inverted). For most devices, this value is "true."

2. In the Vmeasure box, type the voltage at which, for timing measurements (like propagation delay), the manufacturer of the IC considers the output buffer "switched." For most devices, this value is 1.5V. (It may be different for newer, very-low-Vcc devices.)

3. In the Rref, Vref, and Cref boxes, type the circuit values that describe the manufacturer's standard test loads for timing measurements. This is almost always specified in the data sheet; if not, use the default values of 1000 ohms, 0V, and 50 pF.

Completing a Buffer Model

When you have finished entering all the parameters described in the preceding sections, then finish specifying the buffer model:

1. Click the Finish button. The buffer model is now complete, and ready to be assigned to specific pins on the component IC.

2. Assign the buffer to specific pins on the IC component.

Mapping Buffer Models to Pins

After creating all desired buffer models, then to advance to the Map Buffer Definitions to Pins page in the Wizard:

1. Click Next.

Once you've created the buffer models for all of the pins on the IC component you're modeling (or decided which pre-defined models you can use for various pins), you can map the models to pins. There is no requirement to map every pin to a model, but since pins by default are "no connects," any pins that you do not map will not be available for analysis in an IBIS simulator.

Mapping Buffer Models to IC Pins

To map buffer models to IC pins:

1. In the Buffer Models list box in the upper right corner of the page, click once to highlight the model which you wish to attach to one or more pins. User-created models are listed at the bottom of the list box, below all of the pre-defined models. If you plan to use a pre-defined model for one or more pins, choose it based on the criteria described in section "Choosing a Pre-defined Buffer Model" below.

2. In the list box on left, highlight one or more pins to which you want to attach the highlighted buffer model. You can use standard Windows multiple-selection mouse actions to choose pins (e.g., shift-click to select a group pins, ctrl-click to select isolated multiple pins).

3. Map the highlighted pins to the highlighted buffer model by clicking the right-arrow button. The mapped pins move from the list box on the left to the Pins Attached list box on the right.

4. Repeat steps 1-3 for additional buffer models, as needed.

5. Continue mapping buffers to pins until the entire list of IC pins on the left is exhausted (or until you've mapped every pin you care about).

Removing Mapping on a Pin

To remove a pin that has been mapped (so that it is eligible for a new mapping):

1. In the Pins Attached list box on the right, highlight one or more pins that from which you want to remove mapping(s).

2. Click the left-arrow button.

3. The unmapped pins move from the list box on the right to the list box on the left.

Notice that you can filter the list box on the right in two different ways: to show all of the pins not attached to the currently highlighted model, or to show all of the pins that are not currently attached to ANY model.

Choosing a Pre-defined Buffer Model

The pre-defined buffer models, any of which you are free to use to model pins on ICs for which you're creating IBIS files, are classified by several categories:

- CMOS or TTL?
- 5-V power supply or 3.3-V power supply?
- switching speed: slow, medium, fast, or ultra-fast?
- directionality: I/O (i.e., bi-directional), output-only, or input-only?

If you can categorize the IC buffer you're trying to model by these criteria, then you can choose the proper pre-defined model for the buffer.

The switching speeds in the pre-defined buffer models (fast, slow, etc.) equate to approximately the switching times shown below:

CMOS, 3.3V, ULTRA-FAST = 0.3 ns	CMOS, 5V, MEDIUM = 6 ns
CMOS, 3.3V, FAST = 1 ns	CMOS, 5V, SLOW = 15 ns
CMOS, 3.3V, MEDIUM = 3 ns	TTL, 5V, FAST = 3 ns rising / 2 ns falling
CMOS, 5V, ULTRA-FAST = 0.3 ns	TTL, 5V, MEDIUM = 6 ns rising / 4 ns falling
CMOS, 5V, FAST = 2.5 ns	

Mapping Power and Ground Pins

For power and ground pins, the IBIS specification uses special, reserved keywords. When you map the pins on your IBIS component, map power pins to pre-defined buffer model "POWER", and ground pins to model "GND".

Single-Pin versus Multi-Pin Models

Although you would normally create an IBIS model that has as many pins as the actual IC you're trying to model, there are occasions when you might

create a single-pin model. One example would be to test a single custom ASIC buffer that is of interest: you are focused on one buffer type and don't know (or care) about how it will be pinned out in the eventual silicon.

There is a slight behavior difference between single-pin and multi-pin IBIS models. When you map pins to buffer models, if the IBIS component is multi-pin, you are required to map at least one pin to a power-supply buffer. However, if the component is single-pin, then the Wizard assumes you are not modeling a real pin out and relaxes this restriction.

About Unmapped Models

When the Wizard generates the IBIS file, any models that are not mapped to any pins are not written into the resulting IBIS file.

Determining Output Impedance

Driver output impedances are not always specified in the IC datasheet. Lacking a manufacturer's specification for impedance, there are several ways to obtain it, described in the following sections.

Measure from Published V-I Curves

Many manufacturers are now publishing (or have readily available) V-I curves for their output-buffer stages (high and low). If you can obtain such curves, output impedance is easy to calculate.

To measure impedance from a V-I curve:

1. Draw a straight line along the linear portion of the V-I curve, before the current "flattens out" or saturates.

2. Take two points on the line. Measure delta(v) and delta(i) between the points.

3. Then calculate the output impedance as Zout = delta(v) / delta(i)

Extract from a SPICE Model

HyperLynx has application note describing how to extract output-impedance from a SPICE buffer model. Contact HyperLynx or your local reseller to request the application note.

Generating the IBIS File

When you have successfully defined your IC component, created buffer models for all of its pins, and mapped the buffer models to the pins, you are ready to generate the IBIS file representing the IC. The power of the Easy IBIS Editor is that it generates the file automatically for you, with guaranteed-correct syntax.

To generate the IBIS file representing the IC about which you were just interviewed:

1. On the Map Buffer Definitions to Pins page, click the Finish button.

2. The Wizard queries you to name the IBIS file. By default, the name is <component_name>.IBS, where <component_name> is the name you entered for the IC component on the very first page of the Wizard. However, IBIS file names are restricted to eight characters in length, so if needed, modify the proposed file name so that it is eight characters long or shorter.

3. Click the Save button.

The Wizard automatically generates the IBIS file, then opens it in the Visual IBIS Editor for viewing and editing.

What to Do with the New Model You've Just Created

At this point, there are several things you can do with your new IBIS model:

- you can make modifications to the model, using the text-editing features in the Visual IBIS Editor

- you can syntax-check the model (although it's extremely unlikely that the Easy IBIS Wizard would generate a syntactically flawed model); for details

on running the IBIS syntax checker, see “Validating an IBIS File's Syntax” above

- you can view its contents graphically, using the viewing features in the Visual IBIS Editor; see “Viewing an IBIS V-I or Waveform Table” above for details

- you can test the model, using the LineSim simulator; see "Testing an IBIS File” below for details

Loading Existing Models into the Easy IBIS Wizard

If you have previously created a model with the Easy IBIS Wizard, and want to edit or view it again in the Wizard, you can do so by first re-loading the model's IBIS file into the Visual IBIS Editor.

To re-load into the Wizard a model previously created by the Wizard:

1. In the Visual IBIS Editor, from the File menu, choose Open.
   OR
   Click the Open File button on the toolbar.

   Then choose the IBIS file that was generated previously by the Wizard. The Editor opens on the IBIS file.

2. Then, from the Editor's IBIS menu, choose Run Easy IBIS File Creation Wizard.
   OR
   Click the “Easy IBIS” button on the toolbar.

The Wizard opens, and the data previously entered into it is restored for modification or viewing.

The Wizard "remembers" the data that was previously entered and subsequently used to generate the IBIS file, because the Wizard saves a binary file with all of the relevant data. The file is called <IBIS_file_name>.HDS, where <IBIS_file_name> is the name of the IBIS file that the Wizard previously generated, and "HDS" means "HyperLynx Development System."

The HDS file contains data even for buffer models that were not originally assigned to pins, and therefore not written to the original IBIS file.

If you move the IBIS file created by the Wizard to another directory and might want to run the Wizard on it again, be sure to move the .HDS file, also.

Limitations to the Easy IBIS Wizard

The following paragraphs describe some limitations to the Easy IBIS Wizard:

1. There is no direct support for ECL or pseudo-ECL models. If you need to generate such a model, you can start with a standard technology type (like CMOS), then modify the resulting IBIS file as needed to make it work for your ECL buffer. (See the IBIS specification for details on the changes needed.) The CMOS model will be incorrect in some important respects (like table reference voltages and model type), but will at least give you a starting "skeleton" to work with. A second option would be to find an existing ECL IBIS model and modify it (i.e., not use the Easy IBIS Wizard at all).

2. If you own a floating-license (i.e., network-licensed) version of LineSim, and have it open and running when you request that the Easy IBIS Wizard generate and launch a test schematic, the launch will fail with a message saying "can only run one copy of LineSim." To launch the test schematic successfully, you must first close the open copy of LineSim. This limitation does not exist with a node-locked copy of LineSim, i.e., you can have LineSim running, then launch a test schematic which will successfully open a new copy of LineSim.

Testing an IBIS File *(Requires LineSim)*

To help you test an IBIS model (whether vendor-supplied or one you just created with the Easy IBIS Wizard — see above for details), the Visual IBIS Editor has a feature which creates a special LineSim schematic, and launches LineSim on that schematic, ready for testing. You must first select, in the Visual IBIS Editor, a particular pin/signal whose buffer model you want tested.

To generate a test schematic for a buffer in an IBIS file, and open it in LineSim:

1. With the IBIS file loaded in the Visual IBIS Editor, from the IBIS menu, choose Select Signal or Pin.
   OR
   Click the Select button on the toolbar.

   The Select Model dialog box opens.

2. Click to highlight the signal whose buffer model you want to test. Or if you prefer to choose by pin name, in the Select By area, first click the Pin radio button, then choose the pin whose model you want.

3. Then, from the IBIS menu, choose Check Model with Simulator.
   OR
   Click the Test button on the toolbar.

LineSim is launched and opened on an automatically generated schematic that invokes the buffer model for the signal/pin you chose. The buffer is tied to a standard test load (50-ohm resistor to ground). You can begin by testing with the schematic "as is" and then modify and enhance it as needed.

About the Schematic Generated for LineSim

The schematic generated for LineSim is placed in the current location specified by LineSim for storing .TLN files (i.e., LineSim schematic files; location is set in the LineSim user interface; from the LineSim Options menu, choose Directories). This storage location contrasts with the IBIS file generated by the Wizard, which is stored in LineSim's default library directory (where all other model files would be residing; again, location is set in the LineSim user interface).

If the Wizard finds an already-existing schematic with the name of the schematic it was about to create, the new schematic will not be created and the old copy will be loaded instead.

More-Advanced Testing *(Requires LineSim)*

BoardSim also ships with a schematic (for use in LineSim; located in the HYPFILES sub-directory) that has a set of recommended tests for an IBIS model: “IbisTest.tln”. This schematic includes four loads which any IBIS model should switch into with sensible results, provided the model is “good.” If you develop a model and it does not behave acceptably into each of these loads, then the model is probably “bad.”

“IbisTest.tln” assumes that the IBIS model represents a normal push-pull driver that does not need any special external load in order to switch properly. For buffers that do not fit this description, you may need to modify the schematic. (For example, an open-drain driver won't switch unless pulled up by an appropriate resistor. Similarly, ECL drivers need a pull-down to Vtt.) For differential drivers, an alternate schematic (“IbisDiff.tln”) is supplied as a starting point.

Printing a File

To print the file that is open in the Visual IBIS Editor:

1. From the File menu, choose Print.
   OR
   Click the Print button on the toolbar.
   A dialog box opens.

2. Change options, if needed, in the dialog box.

3. Click OK.

Printing occurs to whichever printer Windows is currently connected.

Saving/Closing Files and Exiting

The Visual IBIS Editor allows you to save a file as long as the editor is not running in read-only mode. (For details on read-only mode, see “Editing an IBIS File” above in this chapter.)

To *save* the file that is open in the editor:

1. From the File menu, choose Save.
   OR
   Click the Save button on the toolbar.

To *save* the file under a new name:

1. From the File menu, choose Save As.

2. Type the new file name, then click Save.

To *close* the file that is open in the editor:

1. From the File menu, choose Close.
   OR
   Click the Close button on the toolbar.

If you attempt to close a file before saving changes that were made to it, the Editor prompts you to save first.

To exit the Editor:

1. From the File menu, choose Exit.
   OR
   Click the Exit button on the toolbar.

Help with the IBIS Standard

A number of resources are available that describe the IBIS standard and how to create models with it:

- *in the Visual IBIS Editor,* from the Help menu, choose Help IBIS (or click the IBIS button on the toolbar); a Help window opens with the contents of the V2.1 IBIS specification

- *in this manual:*

  - Appendix A, “IBIS V2.1 Specification”; the complete IBIS V2.1 specification; V2.1 offers a rich set of modeling features, some

intended specifically to help semiconductor vendors in detailed model creation

- Application Note: “Creating IBIS Models”; a guide to creating IBIS models written by HyperLynx and intended for non-semiconductor-vendor users

BoardSim Hint: How to Create a Custom IC Model

Sooner or later, you will need an IC model which neither HyperLynx nor the silicon vendor can immediately supply you. This might be for an ASIC, an obsolete IC, a brand-new IC, etc. Fortunately, it is not difficult to create IC models in BoardSim.

The first thing to decide is whether to model the IC with the .MOD or IBIS format. Read Chapter 8, section “IC-Model Formats” for a detailed comparison of the formats.

It is generally easier to create a model with the .MOD format, because:

- BoardSim includes a dialog-box editor for .MOD models, which creates model libraries and files for you
- less device data is required to create a .MOD model than to create an IBIS model
- there is almost always an existing, similar model to use as a starting point for a .MOD model

On the other hand:

- IBIS is the new, emerging signal-integrity modeling standard; creating a model is a good way of becoming familiar with IBIS

- an IBIS model is fairly easy to create using HyperLynx's Easy IBIS Wizard (see section "Running the Easy IBIS File Creation Wizard" above for details)
- IBIS models are portable to other simulators

In order to create a good .MOD model of a driver IC, you must have the following device data:

- the transistor technology (bipolar, Schottky bipolar, CMOS FET, etc.)
- the effective "on" resistance of the upper- and lower-stage output transistors
- the slew time of the low-to-high and high-to-low switching transitions

There are other parameters in the model, too, but the remainder are less critical; you can more safely approximate them, if needed.

In order to create a good IBIS model of a driver IC, you must have:

- at least an approximation of the upper- and lower-stage V-I output curves (although the Easy IBIS Wizard will create a curve for you if you know only the driving impedance, i.e., "on" resistance)
- the slew time of the low-to-high and high-to-low switching transitions

Thus the primary difference between the data requirements for a .MOD model and an IBIS model of a driver IC is the level of detail with which you need to know the driver's V-I output characteristics. A .MOD model runs surprisingly well knowing only the transistor's basic technology (is it bipolar?, CMOS?, etc.) and the effective "on" resistance of the output stage; for a good IBIS model, you need to know at least a few points on the V-I curve, or use the Easy IBIS Wizard, which will create a curve for you from an "on" resistance.

Note: *If you want to create an IBIS model but know only one V-I data point on an output stage's curve, it is critical to know where on the curve that point is. If the point is taken near the "knee" of the curve, such that the driver current saturates beyond the point, then you can safely enter a two-point table with*

entries 0,0 and V1,I1 in your IBIS table, or better yet, use the implied resistance in the Easy IBIS Wizard. But if the point is taken at the beginning or in the middle of the curve, such that the driver's current keeps increasing beyond the point, the two-entry table is erroneous.

Why? Because if a voltage is above or below the two points, BoardSim holds a driver's current at the previous value in the table in an attempt to model the driver's saturation. If the largest current in your table is significantly lower than the driver's actual maximum current, your model will be erroneous.

.MOD Example: Modeling an ASIC Driver

Suppose you need to model an output buffer on a CMOS ASIC. As far as you can tell, the output buffer looks similar to an AC-standard-logic-family driver, but may have a different slew rate, capacitance, etc. This example shows how you can create your own buffer model using the .MOD format.

First, collect whatever data you can about the output driver; this may require going back to the silicon vendor and requesting extra information. The most-critical data is for output V-I characteristics and slew times. (Many vendors are starting to publish V-I curves in their data sheets.)

To begin creating the model:

1. With no board loaded into BoardSim, from the Edit menu choose Databook IC Models.

2. In the Library and Model area, choose Model Library GENERIC.MOD. A model from this library often makes a good starting point for a new model.

3. Choose Device Model 74ACXX:GATE, since the ASIC output is somewhat similar to the 74AC standard-logic family's.

4. Click the Save As button; in Save .MOD Model As dialog box, type Library Name “ASIC.MOD” and Model Name “OUTPUT”.

5. Click OK. This saves the 74AC model into a separate model and library (ASIC.MOD), which can now be edited to create the ASIC model.

To edit the model so it matches the ASIC output:

1. In the Output Drivers area of the Edit .MOD Model dialog box, leave the Type parameters set to CMOS.

2. Calculate an effective "on" resistance for the ASIC buffer: find the "knee" point in the upper stage's curve (the point where the current starts to "roll off" or saturate) and the zero-current point; calculate R_on =delta_V/delta_I; enter this in the high stage's Resistance box; *then repeat for the lower stage.* If you do not have this data, you can measure it using a sample IC, a resistor, and a variable voltage supply.
   If you do not have time for even a simple measurement, then guess! Your models do not need to be exact to get you a simulation that is at least "in the ballpark." Probably, a CMOS ASIC output is not much different than a 74AC output if the two are fabricated in similar geometries. But be conservative and make the ASIC run "hotter," say, 5 ohms.

3. Enter the Slew Time for the upper and lower stages. If you do not have this data, you can measure it with an oscilloscope. (But be sure that you use a high-bandwidth scope, preferably 500-MHz or above. Otherwise, you may measure only the scope's response, not the true slew time.)
   If you do not have time for a measurement, then guess! Again, the ASIC output probably is not much different than a 74AC output if the two are fabricated in similar geometries. But be conservative and make the ASIC run "hotter," say, 1.0 ns instead of 2.0.

4. Leave the Offset Voltage and Clamp Diode data the same as in the 74AC model. You could measure the effective diode resistance with a resistor and variable power supply, but *driver* clamp diodes are usually insignificant in signal-integrity simulations because the driver itself is such a low impedance.

5. If you know it, enter the driver output Capacitance. If you do not know it, leave the data the same as in the 74AC model. (The difference between 5 pF and 7pF is not terribly significant in most cases.)

To save the model:

1. In the Edit .MOD Model dialog box, click the Save button. You now have a custom model OUTPUT for your ASIC buffer, in a library called ASIC.MOD.

If You Decide to Create an IBIS Model Instead

If you decide to create your own model in IBIS format, use HyperLynx's Easy IBIS Wizard. This "smart" tool, which "interviews" you about your buffer's characteristics and then automatically generates syntactically correct, ready-to-simulate IBIS file from your data, makes creating IBIS files relatively easy, even for novices. See section "Running the Easy IBIS File Creation Wizard" above for complete details.

Creating Your Own Ferrite-Bead Models

A ferrite bead, even though a passive component, requires a complex model that cannot be summed up in simple numeric value (unlike a resistor or capacitor value). Accordingly, BoardSim includes a library of ferrite-bead models, much like it includes libraries of IC models. (For details on ferrite-bead libraries and how to choose bead models, see Chapter 8, section "Choosing Ferrite-Bead Models.")

Furthermore, just like BoardSim allows you to create your own custom IC models and save them into libraries, BoardSim also allows you to create custom ferrite-bead models, and save them into your own library.

How Ferrite Beads are Modeled

BoardSim models a ferrite bead with an L-R-C model. (This is the same method as used by several of the more-sophisticated SPICE packages.) However, BoardSim does not require the models' creator to know the values of L, R, and C; these are complex and would probably never be known even to the vendor of a particular ferrite bead.

Instead, BoardSim *synthesizes* an equivalent ferrite-bead model from four pieces of data that can be read from any basic ferrite-bead data sheet:

- the bead's DC resistance (including package resistance), and
- three points of impedance versus frequency

Even summary data sheets on a ferrite bead almost always give these values: a DC resistance and a graph of impedance versus frequency. The three Z-vs-f data points can be read from the graph.

Library File for User-Defined Bead Models: USER.FBD

BoardSim ships with a library, BSW.FBD, that contains a representative sampling of ferrite-bead models from several leading manufacturers. Usually, even if the exact bead you want to simulate is not modeled in BSW.FBD, you can find a close substitute among the shipping bead models. (For details on how to view the contents of BSW.FBD, see Chapter 8, section "Choosing Ferrite-Bead Models.")

Still, if you are using a bead which is not in the BoardSim-supplied library and for which you want an exact model, you can create a custom model and store it in a library called "USER.FBD." When you interactively model a ferrite bead in BoardSim, the program reads the models in BSW.FBD, and then, if USER.FBD exists, reads its models, too. In the Select Ferrite Bead Model dialog box, the models from USER.FBD are promoted to the beginning of the Vendor list box, so that you see your custom models first. (For details on choosing ferrite-bead models, see Chapter 8, section "Choosing Ferrite-Bead Models.")

Ferrite-bead (.FBD) library files must be stored in the root BoardSim directory. This differs from IC-model libraries (which are stored in the LIBS sub-directory under the BoardSim root directory).

Syntax for Ferrite-Bead Models

USER.FBD must be written in BoardSim's .FBD-file format, which is described in Appendix C. The BSW.FBD file contains a header which

succinctly describes the format. If you choose to look at BSW.FBD for a format definition (or even to use it as a starting point for your own USER.FBD file), make a copy of the file first and edit the copy; be careful not to edit or otherwise damage BSW.FBD itself.

How to Create a USER.FBD Library

USER.FBD must be ASCII-only; create it in a text editor (like the HyperLynx File Editor), not an editor that introduces non-ASCII formatting characters into the file. The file must be located in BoardSim's root directory (i.e., the directory that BSW.EXE is installed in.)

You might want to copy a portion of BSW.FBD to USER.FBD to give yourself a "head start" on creating the new library. Then you can modify existing bead models to create your own. Be careful not to leave any bead-model names in USER.FBD that already exist in BSW.FBD, where "names" means combinations of vendor and part-number names.

For an example of how, in detail, to model a particular ferrite bead, see "How to Create a Custom Ferrite-Bead Model" below.

BoardSim Hint: How to Create a Custom Ferrite-Bead Model

BoardSim ships with a library of ferrite-bead models (BSW.FBD). However, eventually you may want to model a bead not contained (or with no close equivalent) in BoardSim's library. Fortunately, it is easy to add your own ferrite-bead models to the user-defined bead library, USER.FBD.

When you first install BoardSim, there is no file USER.FBD. You create it the first time you need to add your own bead model.

The format for .FBD files is described in Appendix C. Before you attempt to create your own definition, you should read the specification thoroughly.

USER.FBD Example: Defining a Bead Model

Suppose you need to model a bead called “Matic19” from a vendor named “Bead-O-Matic.” Assume there is no model for this bead in BSW.FBD.

To create USER.FBD:

1. In a text editor (like the HyperLynx File Editor), begin editing a new file. Be sure you use a text editor, not a word processor that inserts non-ASCII formatting characters into the file.

2. At the top of the file, place these two lines:

```
{FBD}
{VERSION=1.0}
```

To enter the bead’s definition:

1. Immediately following the two header lines, add the definition of the new bead:

```
********************** My Ferrite Bead Models **********************
{MANUFACTURER=Bead-O-Matic}
{BEAD=Matic19
 (R_DC=0.035)
 (PT1=6.0MHZ,   4.0)
 (PT2=100.0MHZ,19.0)
 (PT3=500.0MHZ,27.0)
}
```

   See “Where the Bead Came From” below for a description of how the bead-model data was determined.

2. End the file with the line:

```
{END}
```

To save USER.FBD:

1. Save the file as USER.FBD, into BoardSim’s root directory. (For example, if BoardSim is installed in C:\BSW, save the file as C:\BSW\USER.FBD).

The new bead model will be available in BoardSim as soon as you load (or re-load) a board. (BSW.FBD and USER.FBD are read every time a board is loaded.)

Where the Bead Data Came From

In the example model above, the DC resistance value and three impedance-versus-frequency points are all taken directly from the bead's data sheet. (It is standard practice that ferrite-bead data sheets include Z-vs-f graphs.)

The only "trick" to creating a model is to know which three points to take from the impedance graph. The .FBD-format specification in Appendix C lists detailed rules for choosing the three points. In this case, the frequency points were at:

- about 10% of the nominal frequency (100 MHz)
- the nominal frequency (since the resonant frequency was not available — off the graph)
- the highest frequency on the graph

Example USER.FBD File

Below is a complete sample USER.FBD file, for two imaginary ferrite beads:

```
{FBD}
{VERSION=1.0}

********************** My Ferrite Bead Models **********************
{MANUFACTURER=Bead-O-Matic}
{BEAD=Matic19
 (R_DC=0.035)
 (PT1=6.0MHZ,   4.0)
 (PT2=100.0MHZ,19.0)
 (PT3=500.0MHZ,27.0)
}

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

```
{MANUFACTURER=Bead-O-Rama}
{BEAD=Bead120
 (R_DC=0.42)
 (PT1=2.0MHZ,   3.0)
 (PT2=100.0MHZ,120.0)
 (PT3=300.0MHZ,200.0)
}
{END}
```

Chapter 11: Choosing Resistor and Capacitor Packages

Summary

This chapter describes:

- what a networked-component package is
- the default package library (BSW.PAK)
- how BoardSim automatically identifies packages
- how to choose a package
- how to add a user-defined package

What is a Networked-Component Package?

On your board, resistors and capacitors can be packaged either discretely (for instance, a single resistor) or as part of a component network (e.g., one of four pull-up resistors in a single package).

In order to simulate a net that connects to a networked component, BoardSim must know in what kind of package the component is housed. In particular, BoardSim must know how the package connects the networked components internally.

For example, there is a big difference to BoardSim's simulator between an 8-pin SIP with four series resistors, and an 8-pin SIP with 7 pull-up resistors.

Kinds of Packages

Component Types

BoardSim allows resistors and capacitors to be housed in network packages. BoardSim does not currently support inductors, ferrite beads, or R-C combinations in network packages. (See Chapter 4, "What is a Reference-Designator Mapping?" for details on component types.)

Connection Styles

BoardSim recognizes several styles of internal connection in network packages:

Connection Style	**Description**
series	each component in the package has two independent pins, i.e., is independent of the other components
pull-up	each component in the package has one independent pin and one pin in common with the other components
pull-up/pull-down	each component in the package has one independent pin and *two* pins in common with the other components

The names of the connection styles are descriptive of how each style is typically used, but you can connect a package to the nets on your board in any way you like.

For example, a pull-up-style package with four resistors is typically used to implement four pull-up or pull-down resistors, but BoardSim does not care if you use it some other way.

Note: *However, in the preceding example, BoardSim will not automatically identify the correct package. You will have to change the package choice manually. See "How BoardSim Automatically Identifies Packages" below in this chapter for details on automatic matching, and "Choosing a Package" for details on making choices manually.*

How Packages Affect Simulation

BoardSim uses package information to identify associated nets, when the net being simulated connects to a resistor or capacitor network. If BoardSim does not know how a network package is connected internally, it cannot properly find associated nets. (See Chapter 7, "What are Associated Nets?" for details on associated nets.)

Note: *It is not just a matter of BoardSim omitting associated nets if it has wrong connectivity information — it may actually find incorrect associations. Always be sure that any network packages on nets you are simulating are correctly identified. See "Choosing a Package" below in this chapter for details.*

Default Package Library (BSW.PAK)

BoardSim supplies a library of common network packages in the file BSW.PAK. BoardSim automatically loads BSW.PAK and makes the package definitions in the library available for assigning to network packages on your board. It also attempts to make automatic assignments for networked components on your board to packages in the library. (See "How BoardSim Automatically Identifies Packages" and "Choosing a Package" in this chapter for details.)

Note: *If your board uses a package not described in BSW.PAK, you can add a definition of your own; see "Adding a User Package Definition (USER.PAK)" below in this chapter for details.*

Elements of a Package Definition

Each package definition consists of the following information:

Package Type	Package Name
style	combination of connection style and component type (e.g., R_PULLUP, meaning a resistor network connected internally in pull-up style)
shape	SIP or DIP
number of pins	the total number of pins on the package
list of pin pairs	a list showing to which pins each component in the package is connected

BoardSim shows you each package definition graphically when you choose packages.

The complete specification of the .PAK format (in which the library BSW.PAK is written) is contained in Appendix B.

How BoardSim Automatically Identifies Packages

When you load your board (and again if you edit power-supply nets), BoardSim scans your board to find associated nets. (See Chapter 7, “What are Associated Nets?” for details.) Part of finding associated nets is attempting to automatically identify a package definition for each resistor or capacitor network on your board.

BoardSim cannot correctly determine the package for every networked component on your board. You should check the package assignments of all the networked components on a net before you simulate the net for the first time. Make changes to any incorrect definitions that BoardSim has made. (See

"Choosing a Package" below in this chapter for details on how to make corrections. In some cases, you may be required to add your own package definition; see "Adding a User Package Definition (USER.PAK)" below for details.)

Once you have chosen a package, BoardSim remembers your choice; if you come back to re-simulate the net (in the same BoardSim session or in another), BoardSim will automatically re-load the package for you. (See Chapter 14 for details on how packages are remembered.)

The following section describes what rules BoardSim uses to make its automatic package assignments.

Package-Matching Criteria

Note: *You probably will not need to know these rules, unless you are confused about why BoardSim cannot match a particular component, or need to create your own package definition.*

Determining There is a Package

BoardSim determines that a passive component is in a networked package by looking at the number of pins on the component. If the component has three or more pins, BoardSim assumes that it is in a networked package, not discrete.

For example, if R2 has only pins 1 and 2, BoardSim assumes it is a discrete resistor. But if R2 has six pins(1 - 6), BoardSim assumes it is a resistor network.

Package Shape

BoardSim supports two package shapes: DIP and SIP. To determine which shape a component is, BoardSim looks at the location of its pins. If all the pins fall on a line, the package is SIP; if not, it is assumed to be DIP.

Number of Pins

BoardSim uses two methods to count the number of pins on a networked component. The larger of the two counts is used for the number of pins.

By Counting Connections

The first method for counting pins is to simply count the number of pins on the component that are connected to nets on the board.

Note: *Though this method sounds fool-proof, it is not: some pins on a component may be unconnected. Unconnected pins are not reported in the .HYP file for your board, and the resulting pin count is too low.*

By Looking at Pin Names

The second method for counting pins is to attempt to convert each pin name on the component into an integer. (This works only for components that have numeric names, e.g., “1”, “2”, etc. All packages defined in BSW.PAK have numeric names.) The largest resulting integer is used as the pin count.

Limitations on Automatic Pin Counting

There is at least one situation in which BoardSim cannot correctly count the number of pins on a networked component: if the highest-numbered pin on the component is unconnected, and therefore not reported in the .HYP file. For this reason, BoardSim always makes available the candidate packages with the next-highest pin count (e.g., 10-pin packages even if only 8 pins are counted. See section “Next-Bigger Packages Included” below for details.)

Connection Style

BoardSim determines the connection style of a package by counting the number of power-supply nets connected to the component. The rules are:

0 power-supply nets	series style
1 power-supply net	pull-up style
2 power-supply nets	pull-up/pull-down style

If a package is connected in an unusual way (e.g., a pull-up-style package is used to implement series resistors), BoardSim may assign an incorrect package (with the wrong connection style) to the component. You may need to

manually change the assignment; see “Choosing a Package” below in this chapter for details.

Final Matching

When BoardSim has determined all of the above criteria for a networked component (package shape, number of pins, and connection style), it begins searching its package definitions for a match to the component.

When a candidate definition is found, BoardSim applies two additional criteria before declaring a match:

- if the package is pull-up style or pull-up/pull-down style, are the power-supply nets connected to the power-supply pins (i.e., “common” pins) on the candidate package?
- does the name of every pin on the component match the name of a pin on the candidate package?

If the answer to both questions is “yes,” BoardSim matches the package definition to the networked component. If either answer is “no,” BoardSim continues searching for a match.

Note: *All of the package definitions in BSW.PAK use numeric pin names (1, 2, etc.). The requirement that pin names on the component match pin names in the package definition means that if you number your networked-component pins differently (e.g., A, B, etc.), you must create your own package definition that includes your custom pin names. See “Adding a User Package Definition (USER.PAK)” below in this chapter for details.*

If No Match is Found

If no match is found, BoardSim will omit nets associated through the resistor or capacitor network with the net being simulated. This can result in serious simulation errors.

For example, if you ask BoardSim to simulate Net1 which is connected through a networked series resistor to Net2, but BoardSim cannot identify a

package style for the resistor network, BoardSim will fail to find Net2 as an associated net and will not simulate it.

Or, similarly, if Net1 is connected through a networked resistor to a power-supply voltage, but the network has no package, BoardSim will ignore the pull-up voltage, resulting in an incorrect simulation waveform.

The solution to these problems is to manually choose the correct packages for mis-identified component packages on your board, before simulating. Sometimes, this requires creating your own custom package definition; see "Adding a User Package Definition (USER.PAK)" below in this chapter for details.

If Multiple Matches are Found

It is possible for multiple package definitions in BoardSim's list to match a networked component. For example, for an 8-pin series-style resistor network in a DIP package, there may be two kinds of internal connection: resistors between adjacent pins (1 and 2, 3 and 4, etc.); and resistors between opposite pins (1 and 8, 2 and 7, etc.).

In cases where there are multiple matches, BoardSim arbitrarily uses the first match. This may not be correct. The solution to this problem is to choose the correct package before simulating; see "Choosing a Package" below in this chapter for details.

Next-Bigger Packages Included

In the Select Package Dialog Box

Because it is not uncommon for a resistor or capacitor package to have one or more of its highest-numbered pins unconnected, the Select Package dialog box displays not only package styles that match the number of connected pins on a package, but also styles with the next-largest number of pins.

For example, if a package has 8 pins connected, the dialog box lists possible 8-pin packages AND (if the next-largest packages in the database are 10-pin) possible 10-pin packages. Then, if the 8-pin package really has 10 pins but pins 9 and 10 are unconnected, you can still choose the correct, 10-pin package.

During Automatic Selection

Also, if there are no packages in the database with a matching number of pins, BoardSim tries to automatically match the package to styles with the next-highest number of pins. For example, if a package has 9 connected pins, but the package database has no 9-pin packages, BoardSim will try to match the package to a 10-pin package.

Choosing a Package

To choose a package for a networked resistor or capacitor:

1. From the Select menu, choose Component Models/Values.
   OR
   Click the Select Component Models and Values button on the toolbar.

2. In the Pins list box, highlight a pin on the network-packaged component. Be sure that the component-type icon in the models area shows a resistor or capacitor rather than an IC or other component. (You can choose network packages only for resistors or capacitors.)

3. Click the Select button.
   OR
   Double-click on the pin in the Pins list box.

   The Select Package dialog box opens.

4. If the Packages list box is empty, BoardSim cannot match any existing package descriptions to the component; you must add a description to file USER.PAK. Close the editor and refer to “Adding a User Package Definition (USER.PAK)” below.

   If the Packages list box has entries, the Connectivity area displays the current package choice. (See the following section for details on the picture.)

5. In the Packages list box, highlight the new package you want to choose. (As you highlight packages in the list, the connectivity picture changes to show you how the package is connected internally.)

6. Click OK.
   OR
   Double-click on the package name.

The Select Package dialog box closes, and the package is chosen. Its connectivity picture appears in the Connectivity area.

You can choose a new package by highlighting any pin on the affected component; changing the package for one pin changes it for the whole component.

The information in the Packages list box includes:

- package name
- package shape (SIP or DIP)
- total number of pins on the package

The package names in BSW.PAK are fairly detailed, so the shape and number of pins are usually obvious just from reading the name.

The packages listed in the Packages list box are taken from the file BSW.PAK when BoardSim loads your board. If you create any additional package definitions and put them in file USER.PAK, your packages are displayed at the end of the list. See “Adding a User Package Definition (USER.PAK)” below in this chapter for details.

Connectivity Picture

The connectivity picture attempts to show you graphically how the components in a network package are connected. The following points apply to the connectivity picture:

- internal components (resistors or capacitors) are displayed only as little boxes
- package pins are displayed in blue

- connections are displayed in the following colors:
  - black for connections between independent pins and component ends
  - maroon for connections between common pin #1 (i.e., power-supply pin #1) and component ends
  - green for connections between common pin #2 and component ends
- as much of a package as will fit in the Connectivity area is displayed; if a package is too long, its picture is truncated

To quickly see how a group of packages differ internally:

1. In the Select Package dialog box, in the Packages list box, highlight the first package in the group.
2. Move the highlight down through the list with the arrow key on the keyboard, looking at the Connectivity picture for each package.

Editing a Package's Component Values

Editing the Value for a Series-Style Package

To edit the component value for series-style package:

1. In the Assign Models dialog box, in the Pins list box, highlight a pin on the network package.
2. In the models area, type the new value in the Value edit box.
3. Click OK.

To edit the component value for pull-up-style package:

1. In the Assign Models dialog box, in the Pins list box, highlight a pin on the network package.

2. In the models area, type the new value in the Value edit box. The label around the Value box identifies the pin name of the single common pin on the package.

3. Click OK.

To edit the component value for pull-up/pull-down-style package:

1. In the Assign Models dialog box, in the Pins list box, highlight a pin on the network package.

2. In the models area, there are two Value edit boxes: one for the components connected to the first common pin on the network, and one for the components connected to the second common pin. Type the value for the first-pin components in the upper box, and for the second-pin components in the lower box. Each box is labeled with the pin name of the common pin to which it applies.

3. Click OK.

The new value(s) apply to all components in the network package. Changing the value for one component changes it for all.

Adding a User Package Definition (USER.PAK)

If none of the packages in BSW.PAK match a particular component on your board, you can add your own custom definitions to BoardSim's list. User-created packages go in the file USER.PAK; the file is created in an ASCII format defined in Appendix B.

Reasons for Creating a Custom Package

You must create your own custom package definition any time there is a passive-component network on your board that does not match any definition in BoardSim's package library (see the following section for details). There are several reasons why there might be no matches:

- *You have a component on your board for which there is truly no definition.* For example, you have a component with 28 pins, but BSW.PAK only supports components up to 24 pins; or you have a custom component with an unusual interconnection scheme not covered in BSW.PAK

- *You have a component on your board that IS described in BSW.PAK — but BoardSim does not recognize the component.* For example, you have 16-pin DIP series resistor network, but pin 16 is unconnected and so does not appear in BoardSim's database. BoardSim thinks the package has 15 pins, and cannot find a match.

For more insight into how mis-identifications can occur, see "Package-Matching Criteria" above in this chapter for a complete description of how BoardSim matches packages.

If you must create a package definition to cover a mis-identification, the package may well be "phony," i.e., something that does not really exist, but that matches BoardSim's "understanding" of what the component looks like.

How USER.PAK Supplements BSW.PAK

At the end of loading your board, BoardSim loads the file BSW.PAK. BSW.PAK is a package library supplied by HyperLynx; it contains package definitions for a wide range of common resistor and capacitor networks. The

information in BSW.PAK is displayed in the Packages list box when you open the package editor. (See “Choosing a Package” above in this chapter for details on the package editor.)

After loading BSW.PAK, BoardSim looks to see if there is a file called “USER.PAK”. If so, it loads USER.PAK and appends the information in it to the packages list. Thus, you can add package definitions of your own to USER.PAK to supplement the packages in BSW.PAK.

Syntax for Package Definitions

USER.PAK must be written in BoardSim’s .PAK-file format, which is described completely in Appendix B. “Elements of a Package Definition” above in this chapter also describes the elements found in a package definition.

How to Create a Custom Package Definition

USER.PAK must be ASCII-only; create it in a text editor (like the HyperLynx File Editor), not an editor that introduces non-ASCII formatting characters into the file. The file must be located in BoardSim’s root directory (i.e., the directory that BSW.EXE is installed in.)

You might want to copy a portion of BSW.PAK to USER.PAK to give yourself a “head start.” Then you can modify existing definitions to create your own. Be careful not to leave any PACKAGE names in USER.PAK that already exist in BSW.PAK.

BoardSim Hint: How to Create a Custom Package Definition

BoardSim ships with a library of passive-component packages (BSW.PAK). However, eventually you may use a component network that is not described in BSW.PAK. Fortunately, it is easy to add your own package definitions to the user-defined package library, USER.PAK.

When you first install BoardSim, there is no file USER.PAK. You create it the first time you need to add your own package definition.

The format for .PAK files is described in complete detail in Appendix B. Before you attempt to create your own definition, you should read the specification thoroughly.

USER.PAK Example: Defining a New Package

Suppose you need to model a resistor network that is a 9-pin pull-up style (8 resistors, each with one end tied to a common pin), in a SIP package. There is no definition for this package in BSW.PAK.

To create USER.PAK:

1. In a text editor (like the HyperLynx File Editor), begin editing a new file. Be sure you use a text editor, not a word processor that inserts non-ASCII formatting characters into the file.

2. At the top of the file, place these two lines:

   ```
   {PAK}
   {VERSION=1.10}
   ```

To enter the package's definition:

1. Immediately following the two header lines, add the definition of the new package:

   ```
   {PACK=9_PIN_SIP_PULLUP
   (STYLE=R_PULLUP)
   (SHAPE=SIP)
   (TOTAL_PINS=9)
   (PIN_PAIR=2,1)
   (PIN_PAIR=3,1)
   (PIN_PAIR=4,1)
   (PIN_PAIR=5,1)
   (PIN_PAIR=6,1)
   (PIN_PAIR=7,1)
   (PIN_PAIR=8,1)
   (PIN_PAIR=9,1)
   ```

```
(PIN_LOC=1,1)
(PIN_LOC=2,2)
(PIN_LOC=3,3)
(PIN_LOC=4,4)
(PIN_LOC=5,5)
(PIN_LOC=6,6)
(PIN_LOC=7,7)
(PIN_LOC=8,8)
(PIN_LOC=9,9)
}
```

To save USER.PAK:

1. Save the file as USER.PAK, into BoardSim's root directory. (For example, if BoardSim is installed in C:\BSW, save the file as C:\BSW\USER.PAK).

The package description defines a 9-pin resistor network in a SIP package, with PULL_UP connection style, model name 9_PIN_SIP_PULLUP, and pin 1 as the common pin shared by each resistor. It is easily created by copying the 8-pin SIP pull-up from BSW.PAK, pasting it into USER.PAK, and modifying it slightly.

The new package model will be available in BoardSim as soon as you load (or re-load) a board. (BSW.PAK and USER.PAK are read every time a board is loaded.)

If there are aspects of the package definition in this example that you do not understand, see the .PAK-file specification in Appendix B. It includes more example definitions.

Chapter 12: Running Interactive Simulations

Summary

This chapter describes:

- what steps you must complete before you can simulate interactively
- how to choose oscilloscope probes for viewing your simulation results
- how to set up the oscilloscope
- how to enter an oscilloscope “comment”
- how to run simulations
- how to make time and voltage measurements
- how to include or exclude the effects of vias in your simulation
- how to print simulation results
- how to copy simulation results to the Windows Clipboard
- how to export the oscilloscope’s data to another program, like Microsoft Excel

Requirements Before Simulating

You can open BoardSim's oscilloscope as soon as your board is loaded and you have selected a net, but before it will actually run a simulation, you must:

- have an electrically valid stackup
- choose a driver IC for the net

If you try to simulate before having a valid stackup or selecting a driver IC, BoardSim will give an error. (For details on these topics, see Chapter 5, section "Editing a Stackup" and Chapter 8, section "Interactively Choosing IC Models.")

See "Setting Up the Oscilloscope" below in this chapter for details on opening the oscilloscope.

Choosing Scope Probes

BoardSim's oscilloscope probes work like real oscilloscope probes: you place them at various points on the nets you are simulating to see the voltage waveforms at those points. Each probe can be either a "normal" single-ended probe, or a differential probe.

Where Probes Can be Placed

You can place a scope probe on any component pin on the net you are simulating. BoardSim supports a maximum of six probes.

Attaching a Probe to a Pin

Probe Types: Single-Ended or Differential

Each oscilloscope probe can be either a "normal" singled-ended probe or a differential probe. Most of the time, you'll use single-ended probes. If you are using differential-signal technology (e.g., differential PECL, LVDS, etc.), you'll

probably use differential probes. (You can also look at differential signals single-ended.)

Single-ended probes attach to and display one signal at-a-time. Differential probes attach to two signals, and display the difference between the signals attached to the '+' and '-' sides of the probe.

Attaching Single-Ended Probes

To attach a single-ended scope probe to a pin:

1. From the Scope/Sim menu, choose Attach Probes.

2. In the Pins list box, highlight the pin to which you want attach a probe. Then, in the Probe Channels area, click on the channel button (e.g., Ch1) for the probe that you want to attach.
   OR
   Double-click on the pin to which you want attach a probe; this attaches the next available probe.

3. Click OK.

Each probe channel has a distinct color; the voltage for each probe is displayed in the oscilloscope with the channel's color.

In the Attach Oscilloscope Probes dialog box, the double-clicking method of attaching probes is the fastest, provided you do not care to which color each pin is assigned. If you do care (e.g., you prefer to always have the driver IC's voltage in red), use the highlight-and-click-channel-button method. There is a way to assign all six probes with one button click; see the following section for details.

Attaching Differential Probes

To attach a differential scope probe to a pin:

1. From the Scope/Sim menu, choose Attach Probes.

2. In the Probe Channels area, next to the channel you want to be differential, click on the Differential check box. The Channel button changes into a pair of channel buttons, one labeled '+' and one labeled '-'.

3. In the Pins list box, highlight the pin to which you want attach the '+' side of the differential probe. Then, in the Probe Channels area, click on the '+' button (e.g., Ch1+) for the probe that you just converted to differential.
   OR
   Double-click on the pin which you want attach to the '+' channel (this attaches the next available probe channel).

4. Repeat step 3 for the '-' channel button (e.g., Ch1-).

5. Click OK.

Attaching All Probes Automatically

To quickly attach all six probes:

1. In the Attach Oscilloscope Probes dialog box, click the Attach All button.

2. Click OK.

This method assigns the six probes to the first six pin locations on the currently selected net. Priority is given to ICs, i.e., if there are six or more ICs on the net, plus some passive components, the probes will all attach to ICs.

Note: *Scope probes connect to component pins, not to trace segments or pads or vias. If a scope probe is "dead" (displays no voltage when it should), it means that the component pin which is being probed is not validly connected to the routing metal which is delivering the voltage. This can occur, for example, if the .HYP file contains insufficient or bad pad information.*

If No Probes are Attached

If you do not attach any oscilloscope probes, BoardSim will automatically assign single-ended probes to the ICs on the net (and associated nets) being simulated. This guarantees that you will see at least one waveform (usually more) when the simulator runs.

If one or more probes have their differential check boxes enabled, but no probes are attached to signals when you begin simulating, the auto-attach

feature will turn the differential probes back into single-ended probes, then attach to IC pins.

How Probes Display on the Board

When you exit the Attach Oscilloscope Probes dialog box, the board viewer updates to show the probe locations on the currently selected net. Probes are displayed in the viewer as colored arrows labeled with the pin to which they are attached.

For a differential probe, two arrows of the same color are displayed, one marked with a black '+' and one marked with a black '-'.

Detaching Probes

Detaching All Probes Simultaneously

To detach all of the oscilloscope probes simultaneously:

1. In the Attach Oscilloscope Probes dialog box, click the Detach All button.
2. Click OK.

This returns all of the oscilloscope probes to being unattached.

Detaching Probes One-at-a-Time

To detach a single oscilloscope probe:

1. In the Attach Oscilloscope Probes dialog box, in the Pins list box, highlight the pin whose probe you want to detach.
2. Click on the probe's channel button (ChX).

This detaches the probe and returns the channel to being unassigned. If you click again, the probe is re-attached.

Changing an Oscilloscope Probe

Changing a Probe to Another Signal

To change an assigned oscilloscope probe to another signal:

1. In the Attach Oscilloscope Probes dialog box, in the Pins list box, highlight the pin to which you want to change the probe.

2. In the Probe Channels area, click on the channel button for the probe that you want to change.

The new pin overwrites the old assignment.

Changing a Probe from Differential to Single-Ended

To change a probe from being differential back to single-ended:

1. In the Attach Oscilloscope Probes dialog box, in the Probe Channels area, click on the differential probe's Differential check box to disable it.

The '-' channel button disappears. The signal formerly attached to the differential probe's '+' channel is now attached to the newly single-ended probe.

Setting Up the Oscilloscope

To open the oscilloscope:

1. From the Scope/Sim menu, choose Run Scope.
   OR
   Click the Open Oscilloscope/Simulator button on the toolbar.

The Digital Oscilloscope dialog box opens.

Before you simulate, there are several oscilloscope parameters to set up. The following sections describe them. All settings are made in the Digital Oscilloscope dialog box.

Choosing the Driver Waveform

Edge versus Oscillator Stimulus

BoardSim allows you to simulate with either a single edge (rising or falling) or a repetitive oscillator waveform.

The single edge is often a better choice when you're trying to isolate transmission-line effects, since you can study how a transition settles out without the possibly confusing effects of additional transitions. The oscillator waveform is better for studying the standing-wave effects of repetitive stimulus.

To choose between edge or oscillator stimulus:

1. Click the Edge or Osc radio button in the Driver Waveform area.

Choosing the Edge Direction

If you choose edge stimulus, BoardSim lets you display either the rising or the falling edge of the driver's switching transition.

To choose between a rising or falling edge:

1. Click the Rising Edge or Falling Edge radio button in the Driver Waveform area.

If you want to see both edges simultaneously, set the edge direction one way; run the simulator; set the edge the opposite way; and simulate again.

Specifying the Oscillator Frequency and Duty Cycle

If you choose oscillator stimulus, BoardSim lets you specify the frequency and duty cycle of the switching waveform.

To specify the frequency:

1. Type a value (in MegaHertz) into the MHz edit box in the Driver Waveform area.

To specify the duty cycle:

1. Type a percentage value into the Duty edit box in the Driver Waveform area.

The duty cycle value defines the percentage of time that the driver spends high.

Setting the Horizontal Scale

To set the horizontal time scale:

1. In the Horizontal area, click on the right or left arrows to increase or decrease the amount of simulation time that appears on the oscilloscope's screen.

The oscilloscope's time-scale increments are logarithmic (1,2,5,10,...), just like a real oscilloscope's.

If you change the time scale after simulating (with a waveform already on the screen), the waveform disappears and you must re-simulate.

Setting the Vertical Scale

To set the vertical voltage scale:

1. In the Vertical Scale area, click on the right or left arrows to increase or decrease the voltage "magnification."

The oscilloscope's voltage-scale increments are logarithmic (1,2,5,10,...), just like a real oscilloscope's.

If you change the voltage scale after simulating (with a waveform already on the screen), the waveform re-displays with the new setting.

Ground Marker

On the oscilloscope screen, the 0.0-V position (ground) is marked in two ways:

- with a green arrow just to the left of the screen

- with a green dashed line across the screen

You can move the ground position with the Vertical Position control (see the following section).

Setting the Vertical Position

To set the vertical voltage position:

1. In the Vertical Position area, use the slider bar to move the oscilloscope's vertical position up and down.

When you change the vertical position, the green ground arrow and line move up or down.

If you change the vertical position after simulating (with a waveform already on the screen), the waveform re-displays with the new setting.

Settings Readout

For convenience, the horizontal-scale, vertical-scale, and vertical-position values are summarized and displayed in the oscilloscope display, in white text.

You can disable the settings readout to reduce clutter on the oscilloscope screen.

To disable the oscilloscope settings readout:

1. In the Display area, click on the Show Readout check box to disable it.

The readout disappears from the oscilloscope display. You can re-enable it at any later time.

Setting IC Operating Parameters

You can control from the oscilloscope whether the IC models in a simulation run with best-case, typical, or worst-case operating parameters.

What IC Operating Settings Mean

The IC operating settings are actually combinations of the min and max data in an IBIS model, or scaled versions of a .MOD model (which itself contains only typical data). The combinations are named Slow-Weak, Typical, and Fast-Strong, to be as descriptive as possible.

The following table shows for IBIS models how the combinations are defined:

Parameter	for Fast-Strong	for Slow-Weak
driver current	max	min
slew rate	max	min
clamp-diode current	max	min
component capacitance	min	max
package inductance	min	max
package capacitance	min	max
package resistance	min	max

(See Chapter 8, section “The IBIS Format” for details on the IBIS modeling format.)

For .MOD models, all of the models in a simulation are scaled up or down from their typical values by globally defined scaling factors to give Slow-Weak or Fast-Strong operation. (See “Scaling .MOD Models for Best/Worst-Case Operation” below for details on the scaling.) Not all parameters in a .MOD model are scaled; the following table shows for .MOD models how the operating combinations are created:

Parameter	for Fast-Strong	for Slow-Weak
driver slew time	scaled down	scaled up
driver output impedance	scaled down	scaled up

driver/receiver I/O capacitance	scaled down	scaled up

Setting the Operating Parameters

To set the IC operating parameters:

1. Click the appropriate radio button in the IC Modeling area.

If you want to see the results of more than one operating point simultaneously, set the parameters one way; run the simulator; set the parameters another way; and simulate again.

Does Not Affect IBIS Models with Only "Typical" Data

Changing IC operating parameters only affects the IBIS IC models in your circuit that actually contain min/max data. If you change the IC operating parameters but see no change in your simulation waveforms, it is probably because the IBIS model(s) you are using do not have min/max data. The IBIS format allows for min/typ/max data, but only requires typical.

Scaling .MOD Models for Best/Worst-Case Operation

By default, BoardSim scales the typical device parameters in all of the .MOD models in a simulation up by a factor of 1.8 to create Slow-Weak operation, and down by 0.6 to create Fast-Strong. However, you can adjust these parameters, if you wish, to increase or decrease the "pessimism" of your simulations.

To change the scaling factors used to create best/worst-case .MOD models during simulation:

1. From the Options menu, choose Preferences. Verify that the General tab is selected.

2. In the .MOD IC Model Best/Worst Case Scale Factors area, type the new scaling in the Min Scale Factor and Max Scale Factor edit boxes.

3. Click OK. The new factors apply immediately, and are saved for future BoardSim sessions.

Turning Probes On/Off

You can turn your oscilloscope probes off and on directly in the oscilloscope. Turning a probe "off" means it is still attached in the circuit, but does not display its waveform on the oscilloscope screen.

To turn an oscilloscope probe off:

1. In the oscilloscope, in the Probe Enables area, click on the check box for the channel you want to turn off, to disable it.

If a waveform for the channel you turned off was already on the oscilloscope screen, the waveform disappears.

To turn an oscilloscope probe back on:

1. In the Probe Enables area, click on the check box for the channel you want to turn back on.

Probe Enables

In addition to allowing you to turn probes on and off, the Probe Enables area displays to which pin on the current net each probe is attached. If a probe is not attached to a pin, the status says "Open."

Entering an Oscilloscope Comment

The Digital Oscilloscope dialog box includes an area above the oscilloscope screen labeled "Comment." The Comment box allows you to enter a description or comment that prints when you print your oscilloscope waveforms. (See "Printing a Simulation" below in this chapter for details on printing.)

To enter a printable comment in the oscilloscope:

1. In the Comment box above the oscilloscope screen, click once with the mouse. A cursor appears in the upper left of the box.

2. Type the description or comment.

Attaching Probes from Inside the Oscilloscope

If you want to modify your probe assignments (e.g., add a probe or change which pins are being probed) and currently have the Digital Oscilloscope dialog box open, you can open the Attach Oscilloscope Probes dialog box without closing the oscilloscope.

To attach probes from inside the oscilloscope:

1. Click on the Probes button. The Attach Oscilloscope Probes dialog box opens.

2. Follow the steps listed in "Attaching a Probe to a Pin" above in this chapter.

Running an Interactive Simulation

To run a simulation:

1. In the Digital Oscilloscope dialog box, click the Start Simulation button.

The Simulation Status dialog box opens, and simulation begins.

Pre-Transient Steps

Before anything is displayed on the oscilloscope screen, BoardSim:

1. builds a simulation model

2. optimizes the model for faster performance

3. performs a DC simulation

For many nets, the pre-transient steps occur very quickly, in a second or less. But for larger nets, they can take longer. For very large nets, you may have to wait 30 seconds or more.

Note: *To BoardSim's DC simulator, the "size" of a net is determined by how many trace segments make it up. The lengths of the segments make little*

difference. For the transient simulator, on the other hand, net lengths matter: very short nets slow a simulation.

Transient Steps

When the pre-transient steps are completed, BoardSim begins its transient simulation. The results of the transient simulation are displayed in the oscilloscope as they are calculated.

BoardSim's transient simulations are constrained by the time resolution the program must use for the shortest transmission line in the circuit. BoardSim uses intelligent algorithms (the pre-transient "optimizing passes") to minimize simulation time when possible.

Percent Done and Status Messages

The Simulation Status box displays the percentage of the transient simulation that has completed.

The dialog box also displays messages from BoardSim's simulator. The messages tell you what steps are currently being run.

Stopping a Simulation

To stop a simulation before it has completed:

1. In the Simulation Status dialog box, click the Stop button.

Timing and Voltage Measurements

Sometimes it is necessary to make detailed, accurate time, voltage, or slew-rate measurements from the oscilloscope display (e.g., to measure flight times or over/undershoot). The oscilloscope provides a pair of measurement crosshairs for this purpose.

To measure a single voltage and/or time:

1. Position the mouse in the oscilloscope display exactly where you want to make a measurement.

2. Click the left mouse button. A yellow crosshair appears in the display.

Look in the Cursors area (below the display). An accurate voltage and time readout for the crosshair appears.

To measure a second point or measure a delta voltage, delta time, or slew rate:

1. Click again at the second point in the oscilloscope display. A second yellow crosshair appears.

In the Cursors area, a second readout, "delta readout," and slew-rate readout ("slope") all appear; the delta readout shows the voltage and time difference between the crosshairs, and the slew rate gives the slope (in V/ns) between the two cursor points.

To turn the crosshairs back off:

1. Click a third time in the oscilloscope display. The crosshairs disappear.

"Live" Cursor Readout

The Cursors area in the oscilloscope also provides a "live" (i.e., constantly updated) display of the position of the mouse cursor whenever it is inside the oscilloscope screen. The position reads in V and ns. Therefore (as an alternative to the method described above), you can also make quick measurements simply by moving the mouse to the point on a waveform at which you want to measure, holding the mouse steady, and looking at the Cursor field.

Including/Excluding the Effects of Vias in Simulations

By default, BoardSim includes the effects of the vias on your nets when it builds a simulation model. This gives you the ability to see whether the vias on a particular net are having an adverse effect on your signal quality.

Note: *Vias are modeled in BoardSim with a first-order capacitive model. No attempt is made to calculate via inductance.*

If you want to get "before-and-after" simulations for a certain net, showing what the simulation looks like with and without vias included, you can simulate with via modeling enabled; then disable it; then re-simulate. (Make sure the Show Previous check box is enabled in the oscilloscope; see "Re-Simulating" below in this chapter for details.)

To enable or disable via modeling for simulation:

1. From the Options menu, choose Preferences.
2. Click on the BoardSim tab.
3. In the Analysis Options area, click on the Include Vias check box to toggle its state.

If via modeling was enabled, it is now disabled, and vice versa.

BoardSim "remembers" your last setting between sessions, so if you toggle the via-modeling status, be careful to leave it set the way you want it in future sessions before you exit BoardSim.

Note: *If you include vias, their presence affects not only the simulator (oscilloscope), but also the Terminator Wizard and Board Wizard. For this reason, BoardSim refers to this feature as an "analysis" (not just "simulation") option. The same applies to board temperature.*

Vias and Signal Integrity

In most cases, vias have a perceivable but not a significant effect on simulations (and your signal quality). A typical via "looks" like less than 1.0 pF of capacitance, which is not a large disturbance to a driving signal. In fact, the presence of a receiver IC (often a 5-pF load) midway along a net is usually much more significant than the presence of a via.

There are exceptions, of course. If a net has an unusually high number of vias (e.g., because an auto-router struggled to complete a very dense board), the

cumulative effect may be significant. Also, abnormally large vias, or vias with normal-sized pads but a very thin stackup, may exhibit disruptive amounts of capacitance. Finally, extremely fast drivers (e.g., < 300 ps slew time) tend to be more affected by vias, because of the very high frequencies present in the switching waveform.

Printing a Simulation

You can print your simulation results in order to document them.

To print simulation waveforms:

1. In the Digital Oscilloscope dialog box, click the Print button.
2. In the Print dialog box, check your printer setup. Click OK to begin printing.

BoardSim supports color printers; simulation results sent to a color printer are output with colored waveforms.

Setting Up for Printing

You can set up printing-related defaults — e.g., printer choice, paper size, page orientation, etc. — once in BoardSim, then have them apply for the remainder of your work session, and for all types of printing (schematics, stackups, oscilloscope results, etc.).

To set up “persistent” printing defaults:

1. From the File menu, choose Print Setup. The Print Setup dialog box opens.
2. Change any parameters you wish, then click OK.

BoardSim now remembers the choices you've made, and will continue to use them.

Copying a Simulation to the Clipboard

You can copy your simulation results to the Windows Clipboard in order to paste them into other Windows applications. The image sent to the Clipboard is formatted, and includes information such as the name of the .HYP file, the oscilloscope settings, and a time and date stamp.

To copy simulation waveforms to the Windows Clipboard:

1. In the Digital Oscilloscope dialog box, click the Copy to Clip button.

BoardSim writes to the Clipboard in Windows Enhanced Metafile format. The size of the image may vary depending on which application you paste it into; resize as needed (metafiles are vectored and can be sized without damaging image quality).

Exporting Simulation Data to Another Application (.CSV File)

Occasionally, you may want to export the voltage-versus-time simulation data displayed in the oscilloscope (and the associated currents-versus-time) to another application, like Excel or Mathcad. To facilitate this need, BoardSim's oscilloscope can output a comma-separated-values (.CSV) file which includes all of the data displayed in the oscilloscope window for the present simulation. .CSV files can be read directly by Excel (and most other spreadsheet programs), and are easily read by an input routine in mathematical programs like Mathcad.

To write a .CSV file representing the results of a simulation:

1. In BoardSim, open the Digital Oscilloscope dialog box and run the simulation whose results you want in a .CSV file.

2. Click the Save As CSV button. The Save Oscilloscope Output dialog box opens.

3. Choose a directory and name for the .CSV file, then click the Save button.

The .CSV file is created. BoardSim displays a dialog box reminding you of where and with what name the file was saved.

Format of the .CSV File

The top of the .CSV file records the .HYP-file name, creation date, and other similar data; and also lists to which IC pins the oscilloscope probes were attached when the file was generated. Then, below, arranged in columns, is the time and voltage data for the present oscilloscope simulation, and then in additional columns, the currents corresponding to each voltage.

Important! There is a difference in how voltages and currents are measured in the .CSV file. If an IC has package parasitics, voltages are measured *outside* the package, but currents are measured *inside*. Thus, an oscilloscope probe that appears in the schematic to be outside of an IC is really located inside the IC's package *for current measurements* (but not voltage).

The .CSV file will open directly in programs like Microsoft Excel (in Excel, use choose File/Open or double-click in the Windows Explorer on the .CSV file). If you are reading the file with a mathematics-package program and do not want the header information at the top (creation date, etc.), you can remove it using any text editor.

The time data in the file is in seconds; voltages are in volts; currents are in amps.

Re-Simulating; Comparing Results

You can simulate a given net multiple times to see the effects of opposite driver edges, different IC models, IC operating parameters, and so forth. Sometimes, you want to display the waveforms one on top of the other, to make comparisons between the results; other times, you want to display each result individually.

Plotting One Simulation At-A-Time

By default, if you run a series of simulations without closing the Digital Oscilloscope dialog box, the results of each new simulation supersede the previous simulation's results. For example, if you run with the IC operating parameters set to Slow-Weak, you get one set of waveforms. If you then change operating parameters to Fast-Strong and re-simulate, the Slow-Weak waveforms disappear and the Fast-Strong set replaces them.

Displaying the Previous Plot

For purposes of comparison, you can view the results of both the current simulation *and* the previous simulation *simultaneously*.

To display the results of the next-to-the-last simulation:

1. In the Digital Oscilloscope dialog box, in the Display area, click on the Show Previous check box.

As long as the Show Previous box is checked, the oscilloscope will always show results for both the last and the next-to-the-last simulations that were run, even if the oscilloscope is closed and then re-opened.

Saving and Restoring *Any* Plot

You can also save an oscilloscope plot for restoration at *any* time later in your BoardSim session. This allows you to make comparisons with any simulation you wish, not just the previous simulation as with the Show Previous feature (see above). The saving occurs to a buffer in memory, not to the hard disk, so you can restore the plot only during the same BoardSim session in which you saved it.

To save the results of an oscilloscope plot (to memory):

1. In the Digital Oscilloscope dialog box, click the Copy to Buffer button. The waveform is stored in memory.

To restore a saved plot:

1. In the Digital Oscilloscope dialog box, in the Display area, click on the Show Buffer check box.

You can have both the Show Previous and Show Buffer check boxes enabled, resulting in the waveforms from a saved, the previous, and the current plots all being displayed simultaneously.

Does Not Apply if Oscilloscope Time Scale is Changed

If the oscilloscope's horizontal time scale is changed, any previous simulation results are erased, because there may no longer be sufficient data to display them.

Erasing a Simulation

You can force the results of the current and previous simulations to be erased from the oscilloscope's screen.

To erase the oscilloscope screen:

1. In the Digital Oscilloscope dialog box, click the Erase button.

This causes the current and previous simulations' results to disappear. Results saved in the oscilloscope's buffer (see "Saving and Restoring Any Plot" above for details) are not erased; to make saved waveforms disappear, disable the Show Buffer check box.

Displaying Three Consecutive Simulations

When you are running with IBIS IC models that contain min/typ/max data, you often want to run and compare three consecutive simulations: with the IC operating parameters set to Slow-Weak, then Typical, then Fast-Strong.

To display the results of three consecutive simulations:

1. Click on the Show Previous and Show Buffer check boxes.

2. Run the first simulation, then click the Copy to Buffer button.

3. Run the second simulation.

4. Run the third simulation.

All three sets of waveforms are displayed together.

Simulation Options

In the Options dialog box (choose Options / Preferences), on the Advanced tab, there are several advanced options that affect BoardSim's simulation engine. *HyperLynx recommends that BoardSim users never change these options, though for completeness they are documented in Appendix F.*

Chapter 13: Viewing Board and Net Information

Summary

This chapter describes:

- how to view statistics about your board
- how to view statistics about a net
- pad synthesis

Viewing Board Information

You can view a set of statistics about your board after you have loaded it into BoardSim (see Chapter 4, section “Loading Your Board into BoardSim” for details on loading a board).

To view board statistics:

1. From the Reports menu, choose Board Statistics.

A dialog box opens, and displays a number of statistics about your board, including the total number of:

- nets
- segments
- pins

- vias

Note: *These totals are derived from the .HYP file as it is loaded; depending on how the .HYP-file translator for your PCB-layout package works, there may be small discrepancies from similar totals reported by your layout software.*

Viewing Net Information

You can view a set of statistics about your currently chosen net (see Chapter 7, section "Choosing Nets" for details on choosing a net).

To view net statistics:

1. From the Reports menu, choose Net Statistics.

A dialog box opens, and displays a number of statistics about the currently chosen net and its associated nets. *In the following paragraphs, "net" means "selected net and its associated nets."*

The total delay gives the summed propagation delay of every metal segment on the net. Total length gives the summed physical length of every segment.

The minimum and maximum characteristic impedances are per-segment on the net, and give a rough indication of how much impedance mismatch there is on the net.

The total receiver load capacitance gives the summed value of all the receiver capacitances on the selected net. Large capacitance values may indicate increased signal delays.

Total resistance gives the summed DC resistance of every segment on the net.

The effective net Z0 is a figure that attempts to show by how much the selected net's actual characteristic impedance is effectively lowered by the presence of IC capacitance along the net. This value can be used as a guide when choosing termination resistances, since for nets that are significantly loaded by IC capacitance, the proper termination value is often lower than suggested by the net's actual Z0.

Estimated peak crosstalk gives a rough estimate of the total amount of crosstalk that could occur on the net, based on the neighboring "aggressor" nets and the ICs driving them, *This value is displayed only if you are licensed for BoardSim Crosstalk, have crosstalk analysis enabled, and are using electrical (rather than geometric) thresholds. For details, see the Crosstalk Analysis User's Guide*

For convenience, the Associated Nets list box displays the nets associated with the selected net and — *if you are licensed for BoardSim Crosstalk and have crosstalk analysis enabled* — the aggressor nets coupled to it. Nets that are coupled are identified in the list with the tag "by coupling."

Chapter 14: Saving Session Edits

Summary

This chapter describes:

- how BoardSim saves session edits
- what information BoardSim saves
- what happens when session edits are re-loaded

How BoardSim Saves Session Edits

What are Session Edits?

When you run BoardSim, you typically make at least some interactive changes to its database: you edit your board's stackup, interactively choose IC models, edit passive-component values, set certain parameters, and so forth. Collectively, these changes are called "session edits."

So that you do not have to re-specify this information each time you run the program, BoardSim captures your edits during a session and saves them to a file when you exit. Then, when you re-load the same board in another session, BoardSim reads the edits from the file and re-establishes them automatically.

The BoardSim User Data (.BUD) Session File

Name and Location of the .BUD Session File

The file that BoardSim saves your interactive edits into is called the "BoardSim User Data" (".BUD") session file. There is typically a .BUD session file for each board you load into BoardSim. When BoardSim creates a session file, the file:

- has name <*HYP_file_name*.BUD>, where *HYP_file_name* is the name of your .HYP file
- is located in the same directory as the .HYP file it is associated with

For example, if you load board CONTROL.HYP from directory C:\BSW\HYPFILES; make some interactive changes; and exit BoardSim, the program will create a session file in C:\BSW\HYPFILES called CONTROL.BUD. (You can also save your session edits in the middle of a session; see "How to Save Session Edits" below for details on creating a session file.) CONTROL.BUD contains the edits you made while in BoardSim. The next time you run BoardSim and load CONTROL.HYP, BoardSim will also load CONTROL.BUD and the edits it describes (unless you instruct BoardSim otherwise; see "When Session Edits are Re-Loaded" below).

Format of the .BUD Session File

The .BUD session file is in an ASCII format similar to the .HYP-file format. (See Appendix D for the .HYP-file specification.)

However, the .BUD-file specification is not public. HyperLynx recommends against editing session files. (See "Do Not Edit Either File" below in this chapter for more details.)

The BoardSim .INI (BSW.INI) File

BoardSim also saves some session information into a Windows-style initialization file called BSW.INI. This file is used for edits that are not board-specific like the ones saved into the .BUD session file. An example is the settings in the Options dialog box.

The format of BSW.INI is not public. (See the following section for more details.)

Do Not Edit Either File

HyperLynx recommends against editing or modifying in any way a .BUD session file or the file BSW.INI. Only the BoardSim program should ever write these files. The files are described in this chapter only so that you understand how they function.

What Information is Saved in the Session File

The .BUD session file is board-specific, i.e., there is potentially a session file for each board you load into BoardSim. The session file captures any edits you make in BoardSim that are specific to the board you currently have loaded.

Edits to the following items are saved into the session file:

- stackup
- power supplies
- IC and ferrite-bead models
- passive-component values
- passive-component packages
- for driver-IC models, the Vcc-pin and Vss-pin settings
- new components (Quick Terminators)
- net-by-net batch-mode analysis settings (see Chapter 16)
- simulation temperature

How to Save Session Edits

When Edits are Saved Automatically

BoardSim automatically attempts to save your edits when:

- you exit BoardSim
- you use File/Open to load another board (or re-load the current board)
- you exit the current board with the system control in the corner of the board viewer

When you perform any of the operations listed above , BoardSim queries you, asking whether or not you want to save this session's edits.

To save the current session's edits:

1. Click Yes.

To discard the current session's edits:

1. Click No.

Generally, HyperLynx recommends saving session edits. Not saving risks losing IC models, stackup changes, etc.

Manually Saving Edits During the Middle of a Session

You can also tell BoardSim to save your current edits during the middle of a session. You might want to do this from time to time during a long session to prevent your edits from accidentally being lost if your computer has trouble.

To force session edits to be saved in the middle of a session:

1. From the File menu, choose Save BoardSim Session File. A .BUD session file is created immediately.

When You Might *Not* Want to Save Edits

Normally, you would save the edits you make in a session, so that you do not have to re-make them the next time you load your board. For example, if in a session you interactively choose IC models for a number of component pins, you do not want to have to re-choose all the models next time you run BoardSim.

However, in some sessions you might make a series of edits that are strictly experimental or "throw-away." These you might choose *not* to save.

For example, you might interactively try a series of different driver-IC models, searching for one that improves a certain simulation waveform. In the end, you might decide that none of the alternatives is any better than the IC with which you started. When you exit BoardSim, you decide not to save your edits.

Note: *If in a session you have made some edits that you* ***do*** *want to keep, and now you are planning to make some that you do* ***not,*** *you can force a session file to be saved (see "Manually Saving Edits During the Middle of a Session" above for details); then make the experimental edits; then exit BoardSim (or close the board) and say "No" to discard the additional, throw-away edits.*

When Session Edits are Re-Loaded

Session edits saved in a .BUD session file are read and loaded immediately after the board's .HYP file is loaded, but before the Loading .HYP File dialog box disappears. This allows BoardSim's database to be established with the contents of the .HYP file, then supplemented with additional and modified information from the session file.

BoardSim Queries Automatically before Loading Edits

When BoardSim has finished loading your board, and is about to load the .BUD session file, it opens the Restore Session Edits dialog box and queries you about whether or not you want to load the edits in the session file, and if so, which ones.

BoardSim loads only the sections in the session file that you specifically request. You can turn some sections "off" so that they are selectively ignored.

You can selectively enable or disable loading of:

- stackup
- all other session edits, including component data

To load *all* of the edits in the session file:

1. In the Restore Session Edits dialog box, in the Information to Restore area, verify that the check boxes for Stackup and Component Data are enabled.
2. Click OK.

To load *none* of the edits in the session file:

1. In the Restore Session Edits dialog box, in the Information to Restore area, click on the check boxes for Stackup and Component Data to disable them. Click OK.

To load *some* of the edits in the session file:

1. In the Restore Session Edits dialog box, in the Information to Restore area, click on the check box for the portion of the data you don't want to load, to disable it. Click OK.

When You Might *Not* Want to Load Certain Edits

Suppose, for example, you change your board's stackup in the PCB-layout software, then re-generate the board's .HYP file and load it into BoardSim. BoardSim will load the new .HYP file, then — unless you tell it not to — will overwrite the new stackup with the old one which is saved in the .BUD session file.

To prevent this, disable the stackup portion of the session file from being loaded. (See "BoardSim Queries Automatically before Loading Edits" above for details.) Then, your new stackup is loaded and preserved; when you exit BoardSim, the new stackup is written into the session file. The next time you load your board, you can safely re-enable stackup loading.

Relationship Between Session File and .REF File

Any edits contained in a session file take precedence over models or values for the same IC pins called out in a .REF file (see Chapter 9 for details on loading models/values from an ASCII .REF file).

For example, if the file <*HYP_file_name*>.REF specifies an IC model for pin 1 of IC U1, and the session file <*HYP_file_name*>.BUD specifies a different model for pin 1, the model specified by the session file "wins." This allows the interactive, pin-by-pin edits you make in a BoardSim session to override more-general, component-wide specifications made in a .REF file.

Missing Models and Packages

If the session file has saved some interactive IC-model edits which call out models that no longer exist, the affected edits are simply ignored and no warning messages given. This prevents the session file from becoming a barrier to loading a board.

For example, if you interactively choose a number of models from library MY_LIB.MOD; exit BoardSim so that the choices are recorded in the .BUD file; later (for some reason) delete or move MY_LIB.MOD; then re-load the board, the "bad" references to the now-missing models in MY_LIB.MOD will simply be ignored.

Note: *If a large number of IC models that are recorded in a session file do not "come in" when you re-load a board, check your Model Library File Path (select Options/Directories). Possibly, BoardSim cannot find some of the required libraries.*

Warning about Changing Reference Designators

The component information in a .BUD session file is based on reference designators. *If you renumber the reference designators on your board, you will invalidate most or all of the information in your session file.* This may force you to re-enter much of your component data.

The Backup Session File (.BBD)

When BoardSim saves a new .BUD session file, it looks to see if there is an existing session file. If there is, BoardSim renames the existing file to the name <*file_name*.BBD> (BBD for "backup .BUD"), then saves the new .BUD session file. This ensures that you always have the last two session files that were saved.

By default, BoardSim assumes you want to load session data from <*HYP_file_name*>.BUD, where <*HYP_file_name*> is the name of the .HYP file you are loading. However, you can also choose to load data from the backup session file <*HYP_file_name*>.BBD.

To load session edits from the backup session file (.BBD):

1. In the Restore Session Edits dialog box, in the Session to Restore area, click the Backup to Previous radio button.

2. Click OK.

BoardSim loads session-file data from the file <*HYP_file_name*>.BBD.

You might want to load from the backup session file instead of the current session file if, for example, you accidentally saved data from your last session that you consider "throw-away." Then the .BUD file contains data you do not want, and the .BBD (backup session) file is preferred.

If the Session Files are Deleted

When you first create a given .HYP file, there is no corresponding session file. The session file is created only after you load the .HYP file for the first time, make some edits in BoardSim, and exit; or load the .HYP file, make some edits, and force a session file to be written (see section "Manually Saving Edits During the Middle of a Session" above for details).

If you accidentally delete or lose a session file, you can still load the backup session file, saved as a .BBD file. See "The Backup Session File (.BBD)" above for details.

If you accidentally delete or lose both the .BUD *and* .BBD session files, you can still re-load the corresponding board: if BoardSim finds no session files, it proceeds assuming there are no edits to load. (You lose any edits you made, of course.)

Note: *You should never delete a session file unless you truly want to abandon the editing information it contains.*

If BSW.INI File is Deleted

If for some reason BSW.INI is deleted or lost, BoardSim will create a new file with all settings restored to "factory defaults." BoardSim will still run fine — but you must re-modify any global settings you previously changed.

Chapter 15: Terminator Wizard and Quick Terminators

Summary

This chapter describes:

- the Terminator Wizard, a “smart” tool which recommends terminating-component values
- Quick Terminators, a type of “virtual” component that allows you to add terminators that do not actually exist on your board

Working with Terminators

One of your key weapons in fighting poor signal quality and EMC problems (and sometimes even crosstalk) is *terminators*. Among BoardSim’s primary benefits is helping you to find proper termination schemes for the “problem” nets on your boards.

To this end, BoardSim offers several major features directed specifically at termination:

- the ability to *interactively change the values of the terminating components* on your board and re-simulate with them, until you are satisfied with your waveforms
- *the Terminator Wizard*, a “smart” tool which automatically recommends optimal terminating-component values

- *Quick Terminators,* a feature which allows you to add "virtual" terminating components that are not actually present in your PCB layout
- the *Design Change Summary,* a convenient report summarizing the terminating-component changes you've made

This chapter discusses in detail the Terminator Wizard and Quick Terminators. The ability to interactively change the values of terminating components is discussed in Chapter 8, section "Interactively Editing Rs, Cs, and Ls." The Design Change Summary is discussed in Chapter 16, section "The Design Change Summary."

The Terminator Wizard

BoardSim includes a "smart" Terminator Wizard to help you find optimal values for the terminating components on your boards. "Terminating components" include resistors and capacitors, both series and parallel; "optimal values" means resistances and capacitances that give the best waveforms from a signal-integrity and EMC standpoint. The Terminator Wizard can also work with Quick Terminators ("virtual terminators" that BoardSim allows you to place on a net that doesn't have any termination in its actual layout; see section "Quick Terminators" below for details), and with nets that have multiple terminators in their layout.

Running the Terminator Wizard Interactively

The Terminator Wizard is available to run interactively, any time you choose, on the currently selected net (and its associated nets). The Wizard works in one of two ways depending on whether the net it's running on is terminated or not.

Terminated versus Unterminated Nets

If the net is terminated already, the Wizard bases its analysis on the terminating components present on the net, suggesting, if possible, optimal component values. The terminating components can be actual components present in your board's layout, or Quick Terminators added in BoardSim. (For

details on Quick Terminators, see “Quick Terminators” below in this chapter.) This feature works on any net with a single termination type (e.g., AC or series) , and with a number of useful topologies involving multiple terminators (see “Running the Terminator Wizard on Nets with Multiple Terminators” below for details).

If the net is not terminated, and the Wizard thinks the net is too long to be unterminated, the Wizard will attempt to suggest a termination strategy — both a type of termination and optimal component value(s).

Running the Wizard

To run the Terminator Wizard:

1. Verify that the net you want to analyze is currently selected, and that a driver model has been chosen for the net. (See “Must Have a Driver Model Selected” below in this chapter for details.)

2. Then, from the Wizards menu, choose Terminator Wizard.
   OR
   Click the Open Terminator Wizard button on the toolbar.

   The Terminator Wizard dialog box opens.

The Wizard runs automatically, as soon as its dialog box opens, and for the selected net and any associated nets, displays a list of results in the Terminator Analysis area. The results include information about:

- what components were found on the net(s)
- the electrical characteristics of the net's driver IC (driving impedance and slew time)
- the physical and electrical characteristics of the net(s) (total length, nominal impedance, and “effective” impedance including IC-loading effects)

If the net has any terminating components, the Wizard also displays:

- what kind of termination it found (e.g., series or parallel AC)

- recommendations for optimal termination values
- information about the physical placement of the terminating components
- warnings about any component values that seem problematic
- warnings about any component placements that seem problematic

If the net does *not* have any terminating components, and the Wizard believes the net is too long to be unterminated, the Wizard displays:

- what kind of termination it recommends
- recommendations for optimal termination values

The component-value recommendations and the recommended "best" termination type (at the bottom of the Terminator Analysis area) are the Wizard's most-important benefit. See section "Terminator Wizard Results: Optimal Component Values and Recommended Terminators" below for more details. The Wizard's warnings about improper component values and placement are also very useful (see "Terminator Wizard Results: Signal-Integrity Checks/Warnings" below).

At the bottom of the dialog box, in the Messages area, the Wizard displays warning/error messages and hints about improving the net's signal quality. See the following sections for details on what kinds of messages may appear.

Recognizing Terminator Types

If there is a driver IC present on the currently selected net (see "Must Have a Driver Model Selected" below for details), the Terminator Wizard performs an analysis to determine how the net is terminated, and to find the optimal terminating-component values.

If there are terminating components present on the net, the analysis can succeed only if the Wizard is able to automatically determine from the components (either "real" or Quick Terminators) what type of termination you are using (e.g., series resistor or AC parallel). To determine the termination type, the Wizard examines:

- what resistor and capacitor components are present
- what nets are connected to each component (e.g., two signal nets or a signal net and an AC ground?)
- where the components are placed, especially relative to the driver IC
- other topological details of the net's routing

If the termination type can be identified, the result is displayed in the Terminator Analysis area, in the Termination Type field. If the type cannot be identified, the Termination Type is left as "unknown" (red question mark) and a warning appears in the Messages area; the Wizard then cannot make a recommendation on component values.

Must Have a Driver Model Selected for Complete Analysis

In order for the Terminator Wizard to run a complete analysis and recommend component values, you must have a model selected for the driver IC on the current net. The presence of a driver model is critical because many of the driver's properties have a profound effect on terminating-component values. Important driver properties include:

- slew time
- output impedance
- physical position on the net

If you run the Wizard without a driver model selected, BoardSim gives a warning in the Messages area; even if a termination is present on the net, the Wizard lists the Termination Type as "unknown" (with a red question mark). Some of the statistics about the net are displayed, but no recommendation is made for terminating-component values.

Supported Termination Types and Net Topologies

Important: *Each revision of BoardSim adds additional capabilities to the Terminator Wizard, so termination schemes that were not recognized in an*

earlier version of the program may be in the current one. The following sections describe the Wizard's features at the time of this writing.

Types of Terminators and Topologies Recognized by the Wizard

If the Terminator Wizard finds terminating components already on the net being analyzed (any mixture of "real" components and Quick Terminators), it attempts to identify the termination type and determine optimum values for the components. The following table lists the termination types currently recognized by the Terminator Wizard. See the sections following the table for additional details.

Termination/Topology Type	**Comments**
single series R single DC parallel R (pull-up *or* pull-down) single DC parallel pull-up/pull-down combination single AC parallel R+C	
multiple series Rs, each terminating one branch of a "star" route	"Star" route means a topology in which an IC drives multiple trace branches in parallel, with the branch point close to the driver
multiple parallel terminators, of any mixture of types	"Any mixture of types" means any mix of DC parallel R, DC parallel pull-up/pull-down combo, and AC parallel R+C
single series R + multiple parallel terminators of the same type	The Wizard cannot recognize series R + multiple parallel terminators of *differing* types (e.g., one pull-up R + one AC terminator)
differential trace-to-trace R, *if*	If the traces in a differential pair are

Termination/Topology Type	Comments
the two traces are driven by an IBIS differential IC model	driven by .MOD, .PML, or an IBIS non-differential model, then the Wizard cannot recognize the termination; also, values are recommended in BoardSim Crosstalk only — see "Differential Line-to-Line Termination" below

Differential Line-to-Line Termination (BoardSim Crosstalk Only)

A line-to-line differential resistor is recognized by the Terminator Wizard. Generally, the Wizard does not support nets (or groups of nets) with multiple drivers present. However, since differential pairs require two drivers for proper circuit operation, an exception is made for them. But unless the IC model used is IBIS and specifically identifies the driving pins as differentially paired, the Wizard has no way of knowing that a differential situation is present, and so won't recognize the termination.

Further, in order to predict an optimal value for such a terminator, the Wizard needs access to BoardSim Crosstalk's field solver. Accordingly, recommendations for differential-terminator values are available only if you are licensed for BoardSim Crosstalk.

Note: *.MOD and .PML models do not support the concept of differential pin pairs. IBIS models do; the Terminator Wizard requires their use in order to identify differential pairs.*

Some Combinations of Multiple Terminators Not Supported

The Wizard also does not support some complex termination schemes based on multiple terminators. See the table above; any combinations not specifically described in the table are probably not supported.

In situations where the Wizard cannot recognize a complex termination type, use interactive simulation instead of the Wizard to choose optimal component values.

Multiple Drivers Not Supported, Except for Differential IBIS Models

Except for the case of a trace pair driven by an IBIS differential IC model, the Wizard will also not analyze any net that has more than one driver actively selected. In order to analyze such a net (provided it is not differential), change all but one of the drivers into a receiver (or remove the other driver models entirely).Ferrite Beads Not Supported

The Wizard does not support ferrite-bead terminators. Use interactive simulation to find an optimal ferrite bead; see Chapter 8, section “Choosing Ferrite-Bead Models” for details.

How the Wizard Recognizes “Branched” Topologies

The Wizard supports termination schemes in which a “star-routed” net has each of its branches terminated by a separate series resistor. However, in order to recognize such a topology and make useful component-value recommendations for it, the Wizard must be able to automatically judge whether a given net is actually routed as a valid star.

To make a topological judgment about star routing, the Wizard uses a path-tracing algorithm. If a net has multiple series resistors, it is considered to be a valid star route only if one end of each resistor traces back only to the driver IC, and the other end traces only to receiver ICs.

Terminator Wizard Results: Optimal Component Values and Recommended Terminators

For a terminated net, the optimal component values are often not trivial to find, particularly since the loading effect of IC capacitances effectively alters a net’s characteristic impedance. The Terminator Wizard accounts for all of the IC models currently on the net and its associated nets, factoring the models’ capacitances into an effective characteristic-impedance calculation.

Effective Z0 Value

One of the net-statistic values that the Wizard displays is Effective Z0. The “effective Z0” is a figure that attempts to show by how much the selected net’s actual characteristic impedance is effectively lowered by the presence of IC

capacitance along the net and associated nets. This value can be used as a guide when choosing termination resistances, since for nets that are significantly loaded by IC capacitance, the proper termination value is almost always lower than suggested by the net's actual Z0.

Results for Nets With Single Terminators

For nets with a single terminator already in place (composed either of "real" component(s) or a Quick Terminator), the Wizard attempts to identify the termination type; if the identification succeeds, then the Wizard calculates an optimal value for the terminating component(s).

Recommended Terminating-Component Values

The Terminator Wizard displays its recommended resistor and/or capacitor values at the bottom of the Terminator Analysis area. If the currently selected net uses series or DC parallel termination, only one or more resistor values are recommended; if it uses AC parallel termination, resistor *and* capacitor values are suggested.

If there are multiple resistors or capacitors on a net, then the Wizard's recommended values are identified per-component in the following manner:

```
<component_type> <reference_designator.pin>  suggested:   <value>
```

where <component_type> is the type of component ("resistor" or "capacitor"); <reference_designator.pin> specifies the component's reference designator and a pin on the component; and <value> is the recommended value.

If you make changes to the net being analyzed — for example, change any of its IC models or alter the board's stackup — re-run the Wizard to see how the recommended termination values may have changed in response. The series-resistor value, for example, is strongly dependent on your current choice of driver IC.

Applying Recommended Termination Values

If the Terminator Wizard recommends resistor and/or capacitor values for a terminated net, you can easily apply the recommended values to the

components on your board (or to a Quick Terminator), then re-simulate to see the resulting waveforms.

To apply component values recommended by the Terminator Wizard to the components on your board or a Quick Terminator:

1. Run the Terminator Wizard (see "Running the Wizard" above in this chapter for details). Verify that the Wizard is able to identify the termination type, and recommends component values.

2. Click the Apply Values button.

The recommended component values are exported to the components on your board (or to the Quick Terminator components you've added).

To re-simulate using the recommended values:

1. Close the Terminator Wizard by clicking OK.

2. Open the oscilloscope, and re-simulate.

Results for Nets with Multiple Terminators

If a net has multiple terminators ("real" components or Quick Terminators, or a mixture), the Wizard behaves similarly to how it does when only a single terminator is present (see section "Results for Nets With Single Terminators" above). However, there is now an additional challenge: the Wizard must also determine which of the terminating strategies it thinks is best, and give you a way to get recommended values for whichever type *you* prefer to use.

Why Multiple Terminators?

Why use multiple terminator types on a single net? If you have enough time before layout to analyze your critical nets with HyperLynx's tools, there should be no need to do this. However, some designers, under heavy pressure to get a PCB into layout and "figure the details out later, in parallel" resort to the strategy of placing multiple terminators on critical nets, then deciding later, after layout, which terminating components to actually stuff.

Choosing Between Multiple Terminators

When you run the Wizard on a net with multiple terminators (see “Running the Wizard” for details on running), the Wizard first examines the net to see if the multiple-component configuration is one that it can identify and “understand.” (See section “Supported Termination Types and Net Topologies” above for a description of what multiple-terminator combinations are supported.)

If the terminating scheme is successfully identified, the Wizard displays the combination of components in the Termination Type field, in the Terminator Analysis area (e.g., “series, AC, pull-up”). If not, the Wizard marks the Termination Type with a red question mark, and cannot proceed with analysis.

Assuming the Type has been correctly identified, the Wizard then displays in the Preferred Choice area (on the right side of the dialog box) a set of radio buttons that normally (with single-terminator or unterminated nets) is not displayed. The buttons offer several choices for which terminator type to recommend values for: “Best” (meaning let the Wizard choose what it thinks is the most-optimal of the terminator types it found on the net), and two or more selections that specify exactly which terminator type to use.

For example, if a net has three terminators in its layout, series, AC parallel, and DC pull-up, the Wizard will:

- identify the net as having terminator type “Series, AC, pull-up”
- present radio buttons in the Preferred Choice box for:
  - Best (= let the Wizard recommend the best termination type to use)
  - Series Termination (= force analysis of the series terminator)
  - AC Termination (= force analysis of the AC terminator)
  - DC Termination (= force analysis of the DC pull-up terminator)

To tell the Terminator Wizard on which type of terminator you want analysis:

1. In the Preferred Choice area, click the appropriate radio button.

After you make your choice, the Wizard immediately shows its recommended value for that termination type. If you choose "Best," the Wizard will make a recommendation for the terminator type it thinks will best suit your net; the Wizard's choice is listed in the Terminator Analysis area as the "Suggested Termination."

When you select a new net for analysis, the Wizard does not save your choice of preferred terminator type for the previous net. If you return to the previous net to analyze it again, you must re-choose your preferred type.

Simulating with a Particular Terminator

To simulate with a particular terminator that you chose (or the Wizard recommended) during analysis:

1. In the Termination Suggestions area, click the Apply Values button.

The recommended termination values are automatically placed in your circuit. You can now close the Wizard and open the oscilloscope or spectrum analyzer to simulate and see an actual waveform.

How "Unused" Termination Components are Treated

When you set the component values of your preferred type of terminator using the Apply Values buttons, the Wizard must also set the values for the "unused" terminating components in such a way that they do not interfere with the simulation of the preferred terminator. This is accomplished as follows:

- unused series resistors are set to 0.0 ohms
- in an unused AC terminator, the resistor value is set to 1 Mohm (the capacitor is unchanged)
- in an unused DC pull-up/down terminator, the resistor value is set to 1 Mohm

Results for Nets With No Terminators

If you run the Terminator Wizard on a net that has no terminating components, the Wizard first runs an analysis to determine if the net's signal integrity is likely to be acceptable without termination.

If the Wizard believes that the net does not need termination, then in the Terminator Analysis area, the Termination Type is set to "No termination found"; no termination is suggested; and Apply Values button is grayed out.

On the other hand, if the Wizard concludes that the net is too long to be left unterminated, it will attempt to recommend a termination type, and optimal values for the terminator's components. The algorithms used to determine the optimal terminator type are complex; they take into account the positions of driver and receiver ICs along the net, the topology of the net's routing (e.g., daisy-chained versus star-routed), comparison of driver versus net impedance, etc. Part of the process of recommending a terminator is to also recommend its position on the net, since location is often just as important as component values.

For nets with complex routing schemes (e.g., complicated, "non-obvious" branching), the Wizard can sometimes not find an optimal termination scheme. (You may not be able to either.) Generally, The Terminator Wizard works best on nets that are single-receiver, or daisy-chained, or cleanly star-routed ("cleanly" meaning "with clearly identifiable branches").

Applying Recommended Terminators

If the Terminator Wizard recommends a terminator for an unterminated net, you can easily create the terminator and apply the recommended values to the components, then re-simulate to see the resulting waveforms. The new terminator will be created as a Quick Terminator (see "Quick Terminators" below in this chapter for details.)

To create a recommended terminator as a Quick Terminator, and apply the Wizard's suggested values:

1. Run the Terminator Wizard (see "Running the Wizard" above in this chapter for details). Verify that the Wizard thinks the net needs termination, and recommends a terminator type and component values.

2. Click the Apply Values button.

The recommended terminator is created automatically as a Quick Terminator, and the Wizard's suggested component values are exported to the terminator.

To re-simulate using the new terminator:

1. Close the Terminator Wizard by clicking OK.

2. Open the oscilloscope, and re-simulate.

Manually Adding Terminators to Unterminated Nets

If a net on your board has no terminator, and you want to experiment manually with adding one (rather than automatically creating one based on the Terminator Wizard's recommendation), add your own terminating components wherever desired using the Quick Terminator feature. (See "Quick Terminators" below in this chapter for details on adding terminators.) Then run the Wizard to get component-value recommendations, and the oscilloscope to see the effectiveness of the termination.

Using Standard Component Tolerances for Recommended Values

By default, when the Terminator Wizard calculates recommended terminating-component values, it displays them exactly, without regard for the values you could actually purchase and install on a board, i.e., without considering the components' standard values.

However, you can tell the Wizard to use only standard values, and specify values of which tolerance.

To tell the Wizard to recommend standard component values, for a particular standard tolerance:

1. Run the Terminator Wizard (see "Running the Wizard" above in this chapter for details).

2. In the Terminator Suggestions area, pull down the Apply Tolerance combo box and choose the desired tolerance.

The Terminator Analysis area, if it is currently recommending any component values, updates immediately to display the closest values in the specified tolerance's sequence. If you click the Apply Values button, the components are updated using the standard values.

Terminator Wizard Results: Signal-Integrity Checks/Warnings

In addition to recommending component values and types, the Terminator Wizard automatically runs various signal-integrity checks against the currently selected net. If a violation is found, it is reported in a red font in the Terminator Analysis area, usually in the messages sub-area.

The checks fall into two broad categories: searching for problematic component values (e.g., resistors that are too large or small), and searching for problematic component placement (e.g., a series resistor located too far from the driver it terminates).

The following table lists the signal-integrity checks currently run by the Terminator Wizard. The following sections provide additional details.

Type of Check	**Description**
unterminated-net length	for unterminated nets, checks if the net's length is too long to have no termination; suggests a solution, if possible
component value non-optimal	if a terminating component's value is more than 25% different from the value the Wizard thinks is optimal, the Wizard issues a warning; this makes it easy to find components that probably need "fixing"
driver-to-series-resistor length	for series-resistor terminators, checks if the distance from driver to resistor is too long for effective termination
AC-terminator resistor-	for AC parallel terminators, checks if the distance between resistor and capacitor is too long for

Type of Check	Description
to-capacitor length	effective termination
pull-up/pull-down combo resistor-to-resistor length	for DC parallel pull-up/pull-down combo terminators, checks if the distance between the two resistors is too long for effective termination
receiver-IC stub length	for each receiver IC, checks if its stub length (i.e., distance from the "main" trace routing) is too long
resistor placement relative to receiver ICs	for any non-series type of terminating resistance, checks for improper placement relative to receiver ICs on the net; e.g., will flag a DC parallel resistor that is located too far from a receiver IC
driver impedance exceeding net's impedance	for each driver, issues a warning if the driver's impedance exceeds the net's impedance, i.e., if the driver intrinsically over-terminates the net
driver impedance large enough to cause bad DC levels or excessive tolerance variation in the driver itself	for each driver, issues a warning if the driver impedance is large enough to cause any of the following problems: — likelihood of invalid DC levels (when DC termination used) — likelihood of an excessive portion of series termination residing in the driver itself and therefore subject to excessive tolerance variations

If the Terminator Wizard issues a warning based on one or more of its signal-integrity checks, it does not necessarily mean that your termination won't work. It may help explain, however, why the waveform you see in the oscilloscope is less than perfect, even if you are using the component values recommended by the Terminator Wizard. The Wizard cannot compensate, for example, for improper component placement (e.g., a series resistor located too far from the driver IC) or poor routing topology.

About Driver-IC Impedance

The Terminator Wizard runs several types of signal-integrity checks having to do with the impedance of the driver IC relative to the characteristic impedance of the net being driven. This section describes why.

One problem that can occur if a driver IC has a higher impedance than the driven net is *over-termination*. Over-termination means that the driver IC by itself has more than enough impedance to series terminate the net it's driving. This implies first of all that there's no point in adding more series resistance to the driver externally; and second, that the driver may be delivering too small a step into the net to make series termination even work. In cases such as this, it may be necessary to use a different driver with lower impedance, or to increase the impedance of the driven net.

Note: *To quickly see what a driver's impedance is, open the Terminator Wizard and look in the Terminator Analysis area. Driver impedance is one of the listed statistics.*

Another set of problems may occur when the driver impedance is lower than net's impedance, but still greater than about 20% of the net's Z_0. Now, series termination is possible; however, a significant portion (20% or more) of the terminating impedance is present in the driver IC itself, and this portion impedance has a wide range of values that depends on the IC's manufacturing tolerance. Such inexactness in the overall series-resistance value may make it difficult to reliably terminate the net.

Also, if the net is DC parallel terminated, the relatively large driver impedance will cause the DC levels on the nets to be shifted noticeably away from the normal, unloaded levels. This may cause threshold-crossing problems.

Driver Models vs. Default Slew Rate

When determining if there are signal-integrity violations for a particular net, the Terminator Wizard examines the slew time of the net's driver-IC model. If there is no driver model specified for the net, the Wizard uses a default rise/fall time for its analysis. (This same parameter is used for nets without models when the Board Wizard runs.)

Note: *In many cases, lack of a known driver–IC position will prevent detailed analysis. So even though the Wizard can "fall back" on the default slew-time value, it may not be able to provide much meaningful information if it doesn't have a specific driver-IC model to look at.*

To set the default driver rise/fall time:

1. From inside the Terminator Wizard dialog box, click the Preferences button. The Options dialog box opens.
2. Click the BoardSim tab.
3. In the Board Wizard Defaults are, in Default Rise/Fall edit box, type the desired value (in nanoseconds).
4. Click OK.

You can also set the default time from outside the Terminator Wizard dialog box: from the Options menu, choose Preferences; then follow the steps above.

No Placement Checks for Differential Terminators

The current version of the Terminator Wizard does not check for proper positioning of a differential terminator, e.g., whether a line-to-line terminator has excessive stub length or is otherwise mis-placed. Use interactive simulation to gauge the effectiveness of a differential terminator's location.

Terminator Wizard Results: Pin-to-Pin Physical Lengths

One additional feature of the Terminator Wizard is a list of the physical lengths between component pins. This data can be useful if you're trying to debug Wizard warnings about improper component placement, or generally as a way of seeing exactly how far various component pins are from each other.

The pin-to-pin length distance is displayed at the bottom of the Terminator Analysis area (scroll down to see it, if needed).

Running the Terminator Wizard on Your Entire Board (Batch Mode)

In addition to running the Terminator Wizard interactively, net-by-net, you can also use it in conjunction with BoardSim's Board Wizard feature to scan your entire PCB. This allows you, for example, to get a report of the how many nets on the entire board are too long to be unterminated, or to get component recommendations for all of the terminators on your PCB.

For details on running the Board Wizard and how to link it to the Terminator Wizard, see Chapter 16, section "Running the Board Wizard." Quick Terminators

BoardSim's "Quick Terminator" feature allows you to add to your board terminating components (resistors and capacitors) that are not actually present in the board's layout. This allows you to experiment with terminations not currently in your design.

Quick Terminators are useful whenever you find a signal-integrity or EMC problem on a net that is unterminated, and you want to see if termination fixes the problem. You can also use Quick Terminators to experiment with different termination types on nets that are already terminated with another type (e.g., a DC parallel terminator whose effects you don't like, so you want to try series termination instead).

The Terminator Wizard also uses Quick Terminators, in two ways. First, if one or more Quick Terminators are present on a net, then the Wizard treats them exactly like "real" components, recommending values for them, etc. Second, if a net is unterminated but the Wizard is recommending a termination, it uses Quick Terminators (when you click the Apply Value button) to create the necessary components. See "Results for Nets With No Terminators" above in this chapter for details.

Adding a Quick Terminator

BoardSim allows you to add terminating components to the currently selected net and its associated nets. You can add components regardless of what other components are already present on the net.

Where Quick Terminators Can be Placed

Quick Terminators can be added at any IC pin on the net. This gives you the flexibility to place a resistor at a driver IC (series termination), a resistor and capacitor at the last receiver IC (AC parallel termination), and so forth.

In addition, for series resistor terminators only, you can specify a "stub" length distance from the driver IC. See "Series Resistor 'Stub'" below for details.

You can add terminators at as many IC pins as you like. For example, if a net has two branches and you want to parallel terminate at the end of each branch, you can place terminators at the two last-IC positions.

Types of Quick Terminators

Quick Terminators supports the following termination types:

- series resistor (with stub transmission line)
- parallel AC termination (resistor + capacitor)
- single parallel DC resistor
- pull-up / pull-down parallel DC resistors
- parallel capacitor

The Quick Terminator feature does not support ferrite beads.

Note: *A parallel capacitor is an unusual choice for termination, but is occasionally used, for example at a driver IC as a way of slowing down the driver's output.*

To add a Quick Terminator:

1. From the Select menu, choose Quick Terminator.
   OR
   Click the Select a Quick Terminator button on the toolbar.

   The Assign Models dialog box opens, with the Quick Terminator tab selected.

2. In the Quick Terminator Location list box, click once to highlight the IC pin at which you want to add a terminator.

3. In the Terminator Style area, click the radio button for the terminator type you want to add. A picture of the terminator appears to the left, and edit boxes for the terminator's component values below. A small resistor icon appears next to the selected pin in the Quick Terminator Location list box.

4. In the Terminator Values area, in the edit boxes, type the desired component values. The new values are shown in the picture. For parallel DC terminators, the values include selectable pull-up and pull-down voltages.

5. Click Close to exit the Assign Models dialog box
   OR
   Highlight another IC pin in the Quick Terminator Location list box, to add another terminator.
   OR
   Click another tab at the top of the Assign Models dialog box, to edit other kinds of models.

Note that the Quick Terminator resistor icon appears in the Pins list box next to the pin with the terminator, as a reminder that a Quick Terminator has been applied. This icon is visible regardless of which Assign Models tab has been clicked (IC, Resistor, etc.).

Editing Quick Terminator Values

To edit a Quick Terminator's value(s):

1. In the Assign Models dialog box, click the Quick Terminator tab.
   OR
   From the Select menu, choose Quick Terminator.
   OR
   Click the Select a Quick Terminator button on the toolbar.

2. In the Quick Terminator Location list box, highlight the IC pin whose Quick Terminator values you want to change.

3. In the Terminator Values area, type new values in the appropriate edit boxes. Click Close.

The component values for each Quick Terminator type are listed below:

Quick Terminator Type	Component Values
series resistor	series resistance, stub layer, stub length, stub width
parallel AC (R+C)	resistance, capacitance
parallel DC resistor	resistance, pull-up or pull-down voltage
parallel split DC resistor	pull-up resistance, pull-up voltage, pull-down resistance, pull-down voltage
parallel capacitor	capacitance

For details on the series resistor "stub," see "Series Resistor 'Stub'" below.

Single DC Resistor can be Pull-up or Pull-down

The single, parallel DC resistor Quick Terminator is drawn as a pull-up resistor, but since the voltage to which it is tied can be set to any value (e.g., 5V, or 3.3V, or 0.0V, etc.), this terminator can really be

either a pull-up or pull-down. For example, if you tie the resistor to Vcc, it is a pull-up; if you tie it to 0.0V, it becomes a pull-down.

Series Resistor "Stub"

If you add a series-resistor Quick Terminator, BoardSim allows you to also specify what kind of interconnect separates the resistor from the driver IC. This interconnection is called the series-resistor "stub."

The series-R stub allows you to see what effect separating a series resistor from the driver IC has on the effectiveness of the terminator. In any real PCB layout, the series resistor cannot be *exactly* at the driver IC, as it ideally would. Varying the stub length shows you how far away the resistor can be before the terminator begins to fail.

Note: *Generally, the maximum allowable stub length depends on the switching speed of the driver IC. For slow-switching ICs, a long stub is acceptable; for faster-switching ICs, the maximum-allowable stub length shrinks.*

To specify the series-resistor stub interconnect:

1. Add a series-resistor Quick Terminator (see "Adding a Quick Terminator" above for details). Then, before closing the dialog box or selecting a different tab, in the Terminator Values area:

2. Type values in the Length and Width edit boxes; and

3. Pull down the Layer combo box and select a stackup layer for the stub.

The layers in the Layer combo box match those in your board's stackup. To view the stackup or edit it, use the stackup editor; see Chapter 5 for details.

The stub parameters default as follows:

Layer	to whatever layer the actual trace touching the driver-IC pin is on
Length	to a default short distance

Width	to the width of the actual trace that touches the driver IC pin

Because the stub layer and width default to match the layer and width of the portion of your board's actual routing that touches the driver IC, you usually do not need to change layer and width.

The units used for stub length and width default to those used for measuring lengths everywhere on your board. For details on changing units, see Chapter 4, section "Setting Measurement Units."

Stub Adds to Existing Routing

BoardSim simulates a series-resistor stub by adding a transmission line between the driver IC and the existing routing on your board. This method is used for simplicity; it avoids the complexity of forcing you to specify how to break apart your existing routing to insert the series resistor.

Quick Terminators and EMC Simulations (BoardSim EMC Only)

You can add a Quick Terminator to a net for which you are running EMC simulations, to see whether adding the terminator to your board would improve the net's radiation profile. However, be aware that unlike real passive components (components that are actually included in your layout), Quick Terminators do not contribute any package radiation to the spectrum analyzer's results. This occurs because BoardSim EMC does not have any information about a Quick Terminator's physical package or its orientation on your board.

Removing a Quick Terminator

To remove a Quick Terminator:

1. In the Assign Models dialog box, click the Quick Terminator tab.
   OR
   From the Select menu, choose Quick Terminator.
   OR
   Click the Select a Quick Terminator button on the toolbar.

2. In the Quick Terminator Location list box, highlight the IC pin whose Quick Terminator you want to remove.

3. In the Terminator Style area, click the None radio button. Then click Close.

The terminator's effect in subsequent simulations is removed. Also, the Quick Terminator resistor icon disappears from the Pins list box in the Assign Models dialog box.

Removing a "Real" Terminator to Try a Quick Terminator

Sometimes, you may want to replace a real terminator (one physically present on your PCB) with a Quick Terminator. For example, you might want to remove a DC parallel terminator on your board whose effects you don't like, and try a Quick Terminator series resistor instead.

To do this, use the Assign Models dialog box to change the real terminator's component values such that the terminator no longer has any effect in the circuit. (For details on editing component values, see Chapter 8, section "Interactively Editing Rs, Cs, and Ls.") Then, add the Quick Terminator as described above in "Adding a Quick Terminator."

Example values to use for "removing" a real terminator:

parallel DC resistor	1000 ohms or greater
parallel AC terminator	1000 ohms or greater and 0 pF
series resistor	0 ohms

Quick Terminators and The Terminator Wizard

The Terminator Wizard works with Quick Terminators just like it does with real terminators. This means you can add a Quick Terminator, then run the Terminator Wizard to get recommended component value(s) for it. The Wizard also uses Quick Terminators to create terminations it is recommending on nets that are unterminated in your board's actual layout.

For details on the Terminator Wizard, see “The Terminator Wizard” above in this chapter.

Keeping a Record of Quick Terminators

As you find unterminated nets that need termination and “fix” them by adding Quick Terminators in BoardSim, you may want to keep a record of the new terminators that should be added to your board layout. For example, a list of Quick Terminators might be useful for handing back to your PCB-layout department or service bureau, so they can incorporate the new terminators into a new board revision.

Quick Terminators and the Design Change Summary

BoardSim can create a special report called the “Design Change Summary” which records, among other items, the Quick Terminators you add to your board and their component values. The Design Change Summary is a specialized version of the Board Wizard’s batch-mode output report. (For details on the Board Wizard, see Chapter 16.)

To create a Design Change Summary, which includes a list of Quick Terminators:

1. From the Reports menu, choose Design Change Summary. The Design Changes dialog box opens.

2. Click the Finish button.

After a brief delay during which the report is created, the HyperLynx File Viewer opens to show the contents of the Design Change Summary. The record of Quick Terminators you added to your board appears in the section called “New Terminating Components.”

The report is written to a file called <*HYP_file_name*>.TXT, where <*HYP_file_name*> is the name of your board’s .HYP file. The .TXT file is a text file which you can view with any Windows editor, and copy and paste from if you want to move the information to other Windows tools.

For more details on the Design Change Summary, see Chapter 16, section “The Design Change Summary.”

Chapter 16: The Board Wizard (Batch Mode): Quick Analysis of an Entire PCB

Summary

This chapter describes:

- what the Board Wizard is, and how you might use it
- the difference between Board Wizard's "Quick Analysis" features (described in this chapter) and the "detailed simulation" features (described in Chapter 17)
- how to run Quick Analysis in the Board Wizard
- how to view the Board Wizard's results
- what the Design Change Summary is
- how to generate a Design Change Summary

What is the Board Wizard?

Much of BoardSim is oriented around quick, easy-to-use *interactive* simulation. However, BoardSim also includes a powerful feature called the Board Wizard which lets you scan, in one batch-mode operation, your entire

PCB, looking for possible signal-integrity and EMC problems. The Board Wizard includes a number of different kinds of batch-mode checks and analyses; these features are divided into two categories:

"Quick Analysis" features	fast-running features that do not require detailed simulations, so they run quickly; Board Wizard runs them automatically on *all* of the nets on your board
"detailed simulation" features	more-detailed analysis features which require oscilloscope and spectrum-analyzer waveforms to be generated; Board Wizard runs these only on nets you specifically select

The "Quick Analysis" features of the Board Wizard are described in this chapter. The "detailed simulation" features, are described in Chapter 17. To get a complete overview of BoardSim's batch-mode capabilities, you should refer to both chapters.

Relationship Between the Board Wizard and Terminator Wizard

One particularly beneficial aspect of the Board Wizard is its ability to link to the Terminator Wizard for analysis. Because the Board and Terminator Wizards work together, the Board Wizard can offer all the advantages of the Terminator Wizard, but on a quick, net-by-net basis as the Board Wizard scans your entire PCB. This includes the ability to flag signal-integrity violations and recommend optimal component values for nets with terminators.

(For details on the Terminator Wizard and its features, see Chapter 15, section "The Terminator Wizard." Some of the Terminator Wizard's features are also described below, in the context of the Board Wizard.)

For details on enabling the Terminator Wizard within the Board Wizard, see "Board Wizard Report Options" below in this chapter.

Ways of Using the Board Wizard

Because the Board Wizard is capable of analyzing your entire PCB, there are various possible ways to use it in conjunction with BoardSim's interactive simulation capability. For example, you can use the Wizard:

- as an *initial* screening tool to show you where on your board you might have signal-integrity problems; based on the Wizard's output, you can decide which nets to simulate interactively
- as a *final* "sanity check" tool to verify, after you've completed your interactive simulations, that there are no other major problems on your board that should be investigated

As an Initial Screening Tool

If you are unsure which nets on your board you should simulate, or where your board might have signal-integrity problems, consider running the Board Wizard as a screening tool to help you locate problem nets. Used in this manner, the Wizard becomes an "advisor" about where on your PCB to focus. (See "Viewing the Board Wizard's Results" below for details on how the Wizard reports its results.)

Note: *Users who are licensed for BoardSim's optional Crosstalk module have access to a particularly powerful advisor capability: the Crosstalk Strength Report, which can quickly generate a ranked list of the all the nets on a PCB likely to exhibit more than a selectable amount of crosstalk. See the Crosstalk User's Guide for more details.*

There are some caveats that apply to using the Wizard in this mode. The first is that the Wizard does not know which of your signals are truly critical, e.g., whether a given net is edge-sensitive (like a clock, for which signal quality is always extremely important) or completely uncritical (a reset line), so it flags warnings on all nets without discrimination. *You* have the designer's advantage of knowing which nets are most important, so in some cases, it may actually be more efficient for you to simulate just your critical nets interactively, guided by your knowledge of your design, than to run the Wizard.

Second, the Wizard runs best when you first choose driver models for at least the critical nets on your board. This is true because the Terminator Wizard (and certain other batch-mode quick-analysis features) can do a more in-depth job knowing for each net what the exact driver IC is. Specifying driver ICs is not required; the Terminator Wizard uses a default slew time for nets for which no driver model is provided (see “Default Slew Time” below for details). If many of your nets are critical and you don’t want to make the effort to choose IC models for them, the Board Wizard’s results will be somewhat less accurate than if models are provided; some types of analysis may not be able to run at all.

The best way to specify a large number of driver-IC models is to create a .REF “automapping” file that maps reference designators to IC components/models. For details, see Chapter 9, section “Automatic versus Interactive Selection of Models” and following sections.

As a Final “Sanity Check” or Regression Test

If you have a fairly clear understanding of what nets on your board to simulate (i.e., which ones are signal-integrity critical and which are not), and you choose to perform interactive simulations without first running the Board Wizard, you may still want to run the Wizard before you finish with your board, as a final automated check. Used in this mode, the Wizard becomes a last “sanity check” to help you ensure that you didn’t miss any major problems on your board through oversight.

If your board is large and you’re confident that your interactive simulations targeted all the important nets, you may want to skip this step to avoid the extra setup time (if you choose to specify driver models) and run time (the Wizard takes some time to run on large boards). Otherwise, this final “safety check” probably makes good sense.

Another use for the Board Wizard is as a regression test. Suppose you solve all of the critical high-speed-design problems on a certain layout, but then for other reasons (e.g., mechanical, logical, cost-reducing, etc.) you need to make revisions to the board and re-lay it out. Running the Board Wizard on the new layout as a way of seeing whether any of your critical nets have “broken” (compared to the old layout) may prove very useful.

Running the Board Wizard

The Board Wizard runs by first asking you want kind of information you want it to generate, then performing its analysis of your board based on your selections. Its results are written to a report file.

Check Power Supplies Before Running

Before running the Board Wizard, you should check to see that all of your board's power-supply nets are correctly identified to BoardSim. (For details on editing the power-supplies list, see Chapter 6.) If a power-supply net that connects to many other nets (through components like resistors and capacitors) is left unidentified, the Board Wizard will run for a very long time when it analyzes the net (the Wizard may appear to be "hung").

Running the Wizard for Quick Analysis

To run the Board Wizard:

1. From the Wizards menu, choose Board Wizard.
   OR
   Click the Run Board Wizard button on the toolbar.
   The Board Wizard dialog box opens.

2. In the Quick Analysis area, click on the check boxes for the sections you want included in the report. (See "Types of Quick Analysis" below for details on what each option means.)
   Then click Next.

   Note: *The detailed-simulations options enable detailed batch-mode simulation, as opposed to the other options discussed here, which are quick-analysis options. For information on the detailed simulations, see Chapter 17.*

3. The next several pages of the Wizard (how many pages you get depends on how many options you enabled in step 2) allow you to check on or off which detailed portions of each information section you want in the report. Set these as desired, and continue to click Next.

4. When you reach the Batch-Mode Analysis page, click Finish. The Board Wizard begins running on your board, showing its progress as it runs. For large boards, the Wizard may take some time to run, even for only quick-analysis options. For very large boards, several hours may be required; for more-average-sized PCBs, run times are usually much shorter.

5. When the report is completed, it opens automatically in the HyperLynx Report File Viewer for viewing. If the report is very large, it may take a while to load even after the Wizard itself has stopped running.

Stopping the Board Wizard

You can force the Board Wizard to stop running before it has completed analysis of your entire board (e.g., if you remember that you set something up incorrectly, or if the Wizard is taking longer to run than you expected).

To stop the Board Wizard before it has completed its analysis:

1. Click the Cancel button. A message appears briefly, saying that the Wizard will stop after it finishes analysis of the present net.

When the Wizard has finished with the present net, it stops and opens its report for viewing. The report shows the results data for the nets that *did* run, before you stopped the Wizard. For details on the report, see below.

Types of Quick Analysis

This section describes the Quick Analysis options available in the Board Wizard. These are distinguished from the Wizard's detailed-simulation options by the speed at which they run(much faster) and by the fact that they do not require as much set-up (e.g., many of the Quick Analysis features do not require IC models on nets, although some run with higher-accuracy if IC models are set up).

For details on the Board Wizard's detailed-simulation features, see Chapter 17.

Quick Analysis Options

On the Board Wizard's first page, in the Quick Analysis area, is a list of the available quick-analysis features. You can enable/disable these features to customize the Wizard's analysis and output reporting to your liking, e.g., you can streamline the report if you're interested only in a subset of its data.

The following table lists each of the Quick Analysis features. For more details on each, see the following sections.

Feature	**Description**
Signal-integrity problems	Flags possible signal-integrity violations like too-long nets, too-long "stubs," etc.; uses the Terminator Wizard to identify problems
Terminator suggestions	Suggests optimal values for terminating components; uses the Terminator Wizard to calculate values
Component changes	Lists the resistor and capacitor values that you have changed interactively
New components	Lists the Quick Terminators that you have added, and their component values
Stackup	Lists your board's most-recent stackup and some of its characteristics, like layer impedance
Metal interconnects	Lists electrical statistics about each net, like delay, impedance, etc.
Counts	Lists statistical counts for each net, like number of ICs, segments, etc.

Signal-Integrity Problems

Enabling the signal-integrity-problems check box ("Show Signal-Integrity Problems Caused by Line Lengths") will cause the Terminator Wizard to examine every non-power-supply net on your board, running the checks and

generating the warnings described in Chapter 15, section “Terminator Wizard Results: Signal-Integrity Checks/Warnings.” See Chapter 15 for complete details, but these include such items as checking net lengths against driver-IC switching speed, checking stub lengths, looking for improper component placement, warning about non-optimal termination-component values, etc.

Terminator Suggestions

Enabling the terminator-suggestions check box (“Suggest Termination Changes and Optimal Values”) will cause the Terminator Wizard to examine every non-power-supply net on your board, calculating optimal values for existing terminating components and recommending terminator types and values for too-long unterminated nets. See Chapter 15, section “Terminator Wizard Results: Optimal Component Values and Recommended Terminators” for complete details on these capabilities.

Component Changes and New Components

Enabling the component-change check box (“Show Component Changes”) causes the Board Wizard to list, for your entire board, the manual edits you’ve made to resistor and capacitor values. This can serve as a record of changes you make to terminating-component values in the process of “tuning” terminators as you simulate.

Similarly, enabling the new-components check box (“Show New Components”) causes the Board Wizard to list (again, for your entire board) the Quick Terminators you’ve added to your board, and their component values.

Either or both of these lists (i.e., sections of the Board Wizard’s report file) might be appropriate to give to your layout designer or service bureau as a record of changes you want made to your board for the sake of its signal integrity or EMC behavior. You can also use the lists yourself to drive changes in your board’s schematics.

You can generate a streamlined version of the Board Wizard’s output that contains only the elements most likely to be handed back to your layout person or service bureau (Stackup, Component Changes, and New Components) by creating the Design Change Summary. See “The Design Change Summary” below in this chapter for details.

Stackup

Enabling the stackup check box ("Show Stackup") causes the Board Wizard to list your board's current stackup, in considerable detail and in an organized format suitable for handing to your PCB manufacturer. If you've made any changes to your board's stackup for signal-integrity reasons using BoardSim's stackup editor, those changes are reflected in this section.

You can also print directly from the stackup editor or copy its contents to the Windows Clipboard, if you want a *graphical* record of your stackup. See Chapter 5, sections "Printing Stackups" and "Copying a Stackup to the Clipboard" for details.

Metal Interconnects

Enabling the metal-interconnects check box ("Show Metal Interconnects") causes the Board Wizard to list a number of detailed electrical statistics about each net on your board. Among these are total delay, minimum, maximum, and average impedance, and inductance, capacitance, and resistance.

The metal-interconnects statistics occupy a fairly large percentage of the Board Wizard's report file, if included in it. If you do not need these detailed statistics, disable the Metal Interconnects feature to avoid unnecessary clutter in the file.

Counts

Enabling the counts check box ("Show Counts") causes the Board Wizard to list a number of numeric counts for each net on your board. Among these are the number of segments making up the net, the number of driver ICs, number of receiver ICs, number of resistors, and number of capacitors.

The counts occupy a fairly large percentage of the Board Wizard's report file, if included in it. If you do not need this data (often it's of minimal interest), disable the Counts feature to avoid unnecessary clutter in the file.

Options for Each Quick Analysis Feature

This section describes the options applying to each of the Quick Analysis features listed above (see "Types of Quick Analysis"). As you page through the

Board Wizard, after enabling the Quick Analysis features you want to run subsequent pages give you a finer level of detail regarding what information for each analysis feature to include in the Wizard's report. You are only presented with pages that are relevant to the Quick Analysis features you enabled, e.g., if you disable Metal Interconnects, the page detailing which interconnect information to include is skipped.

Default IC Model Settings

The Batch-Mode Default IC Model Settings page is fairly important and is used by many of the Quick Analysis features: it guides the Board Wizard's analysis for all nets for which no driver-IC model has been selected. For example, if a board had 10 nets and you specified selected driver-IC models for two of the nets, then ran the Wizard, it would use detailed information based on the models you chose for the two nets with models, and use the default IC characteristics for the remaining eight nets.

Hints on Choosing a Default Rise/Fall Time

The most-important of the default-IC-model parameters is the rise/fall time. It should be a switching time that represents the "average worst-case driver IC" on your board. By "average worst-case" is meant a value that, on average, represents the faster-switching ICs on your board.

For example, if your board has ICs with rise/fall times ranging from 2 to 3 nanoseconds, with a few others that switch more slowly (5 - 10 nanoseconds), a good default value would be 2 nanoseconds. This will be too conservative for some nets, but ensures that no net's signal-integrity problems will be missed when the Board Wizard runs.

If the faster-switching devices on your board have non-symmetric rise/fall times, e.g., the falling edge is consistently faster than the rising edge, use a value that represents the faster edge. The faster edge will nearly always constrain your signal-integrity problems.

If your board's ICs have a wide range of switching times, e.g., you have a lot of ICs that switch in 3 nanoseconds, and an important subset that switches in 1 nanosecond, you may want to run the Board Wizard twice, once with the Default Rise/Fall Time set to 3 nanoseconds, and once with it set to 1 nanosecond.

If you do run the Wizard twice and want to save both sets of results, a good method is to run and generate the first report; save the file out of the Report File Viewer (where you're viewing it) under a different name; run and generate the second report; and save it under a different name, too. Then, to view one of the reports later, first rename it (using the Windows Explorer or File Manager) to <*HYP_file_name*>.RPT (where <*HYP_file_name*> is the name of your board's .HYP file), then open it from the Board Wizard (see "Opening an Existing .RPT File" below in this chapter for details). (Or you can view the reports as named using an external text editor.)

Hints on Choosing Output Impedance and Input Capacitance

In the current version of the Board Wizard, the output-impedance and input-capacitance parameters for the default IC model are less important than the rise/fall time (see "Hints on Choosing a Default Rise/Fall Time" above). If you know good approximate values for these, then use them. If not, then "safe" default choices are 5 ohms for output impedance and 5 pF for input capacitance.

Metal-Interconnects Options

The Batch-Mode Interconnect Reporting page gives various reporting options for the metal-interconnect analysis feature. Most are self-explanatory. Enable only the data that you want to see in the report; disable any statistics that you're not interested in to reduce clutter in the file.

The "total trace delay" is NOT a report of settling or flight time for a given net's signal. Rather, it reports the summed propagation delay of all of the segments making up the net. Remember that a real signal delay must account for the routing topology of the net, possible reflections and ringing, receiver thresholds, etc. To get detailed driver-to-receiver delay data, you must run detailed simulations in the Board Wizard; see Chapter 17 for details.

Terminator Wizard Options

If you enable the terminator suggestions feature, the Batch-Mode Termination Reporting page lets you optionally suppress additional information about each net's total IC receiver capacitance and "effective" impedance. ("Effective" impedance reduces the nominal characteristic impedance of a trace to account

for the presence of lumped IC-capacitance loading.) If you don't care about seeing these values explicitly, disable these options to shorten the Board Wizard report file (the Terminator Wizard will account for these parameters whether you choose to report them or not).

If you enable the signal-integrity problems feature, the Batch-Mode Signal-Integrity Warnings page gives you several related options. Normally, you would always leave the Termination Length Violations enabled, since reporting these violations is the feature's main benefit.

350-Ohm Default Threshold for Considering Resistors to be Terminators

By default, the Board Wizard considers a net unterminated (and so will report length violations, etc. for it) if it has no resistors attached to it with a value less than 350 ohms. The reason for this is nets with high-impedance pull-ups (e.g., a 10-Kohm pull-up). Without this threshold, the Terminator Wizard would consider a 10-K pull-up to be a DC parallel terminator with a bad value, and focus on recommending a better value for it. But it reality, it's not a terminator at all, and so you'd probably prefer to have the Wizard ignoring the pull-up completely.

However, if for some reason you do not want the 350-ohm-threshold behavior, then by checking the Do Not Report Length Violations if Any Resistors Found on Net box, you can override it and tell the Wizard to consider as terminated *any* net with any resistors attached.

Viewing the Board Wizard's Results

When you run the Board Wizard, it writes its results into an ASCII report file called <*HYP_file_name*>.RPT, where <*HYP_file_name*> is the name of your board's .HYP file. When the Wizard has finished running and the report file is complete, BoardSim automatically opens the report for viewing in the Report File Viewer.

Changing the Name of the Report File

You can change the default name of the report file, if you want.

To change the name of the report file:

1. On the Batch-Mode Analysis page in the Board Wizard, just before clicking the Finish button to begin running the Wizard, type a different name in the Report File Name edit box.

Format of the Wizard's Report

The Wizard's report file (.RPT) is formatted to be easy to read and understand. It is divided into sections as described in the table below. The table notes for each section which report option must be enabled for the section to be included in the report. See "Types of Quick Analysis" above for details on how to enable report options, and what kind of information each section contains.

Option	Information Contained...
General Information	Date and time of report, and general statistics about your board; always included
Stackup	Description of your board's current stackup; enable stackup feature to include
Changed Passive-Component Values	Interactively edited resistor and capacitor values; enable component-changes feature to include
New Terminating Components	Quick Terminators that have been added; enable new-components feature to include
Net Information	Net-by-net statistics, including: ♦ Counts — various counts; enable counts feature to include ♦ Interconnect Statistics — electrical statistics; enable metal-interconnects feature to include; warning for completely unrouted nets

Option	Information Contained...
	♦ Signal-Integrity Statistics — Terminator Wizard violations, recommendations, and statistics; enable signal-integrity-problems and/or terminator-suggestions feature to include

Nets with Multiple Terminators Not Analyzed

If the Board Wizard encounters a net with multiple terminators (i.e., more than one type of terminator, e.g., series and AC parallel), it will report that it cannot perform an analysis in batch mode, and recommend that you analyze the affected net interactively. If run interactively, the Terminator Wizard allows you to choose which specific terminator you want analyzed.

Searching for Signal-Integrity Warnings and Violations

Searching in the HyperLynx Report File Viewer

The HyperLynx Report File Viewer makes it particularly easy to search in the Board Wizard's report file for signal-integrity warnings and violations. There are searching features in the editor specifically for finding such warnings.

To search for signal-integrity warnings in the Board Wizard's report file, using the HyperLynx Report File Viewer:

1. From the Viewer's Search menu, choose Find Warning.
   OR
   Click the yellow Find Warning button on the Viewer's toolbar.

The viewer jumps to the next signal-integrity warning below the cursor.

To search for the next warning in the report file:

1. From the Viewer's Search menu, choose Find Next.
   OR
   Click the yellow Find Warning button on the Viewer's toolbar.
   OR
   Press F3.

Signal-integrity warnings are subdivided between “warnings” and “severe warnings.” See “Warnings vs. Severe Warnings” below for details.

To search for only “severe warnings”:

1. From the Viewer’s Search menu, choose Find Warning Severe.
   OR
   Click the red Find Warning Severe button on the Viewer’s toolbar.

Warnings vs. Severe Warnings

The signal-integrity warnings written into the Board Wizard’s report file by the Terminator Wizard are divided into two categories:

- normal warnings
- “severe” warnings

While warnings of either kind merit attention and, if they occur on nets whose signal integrity is important, detailed simulation, the severe warnings indicate particularly troubling problems.

In the current version of BoardSim, severe warnings are issued when stub-length violations are found, i.e., series-terminator-driver-to-resistor lengths or AC-terminator-resistor-to-capacitor lengths that are too long. These are considered severe because they indicate problems in your layout which are extremely difficult to fix unless the lengths themselves are reduced.

Searching in a Non-HyperLynx Editor

You can view the Board Wizard’s report file in an editor other than the HyperLynx Report File Viewer.

To search for signal-integrity warnings in the Board Wizard’s report file, using a non-HyperLynx editor:

1. Using your editor’s “search” or “find” capabilities, search on the string “warning”.

To search for severe signal-integrity warnings in the Board Wizard's report file:

1. Using your editor's "search" or "find" capabilities, search on the string "severe".

Opening an Existing .RPT File

To view an existing Board Wizard report using the HyperLynx Report File Viewer:

1. From the Wizards menu, choose Board Wizard.
   OR
   Click the Run Board Wizard button on the toolbar.
   The Board Wizard dialog box opens.

2. In the View Previous Board Report area, click Open. The HyperLynx Report File Viewer is launched.

The Open button automatically causes BoardSim to search for a file named <*HYP_file_name*>.RPT, where <*HYP_file_name*> is the name of the currently loaded .HYP file.

The Design Change Summary

One use for the Board Wizard is to generate a concise report of all the component changes you've made on your board due to signal-integrity or EMC concerns. This report might be appropriate to give to your layout designer or service bureau as a record of changes you want made to your board in its next revision. You could also use the list yourself to drive changes in your board's schematics.

If you generated such a report with the Board Wizard, the Quick Analysis features you would need to enable include:

- stackup (in case you've changed the stackup to affect impedances)
- component changes (in case you've modified terminating-component values to improve signal quality)

- new components (in case you've added new terminators using the Quick Terminator feature)

Rather than you having to customize the Board Wizard's output every time you want this kind of report, BoardSim can automatically run the Board Wizard with the options listed above pre-set. This customized version of the Board Wizard's report is called a "Design Change Summary."

To create a Design Change Summary, which includes changes to your stackup and terminating-component values, and a record of new (Quick) terminators:

1. From the Reports menu, choose Design Change Summary. The Design Changes dialog box opens.

2. Click the Finish button.

After a brief delay during which the report is created, the HyperLynx Report File Viewer opens to show the contents of the Design Change Summary.

The Design Change Summary is written to a file called <*HYP_file_name*>.TXT, where <*HYP_file_name*> is the name of your board's .HYP file. The .TXT file is a text file which you can view with any Windows editor, and copy and paste from if you want to move the information to other Windows tools.

Chapter 17: The Board Wizard (Batch Mode): Detailed Simulation of an Entire PCB

Summary

This chapter describes:

- the difference between the Board Wizard's "Detailed Simulation" features (described in this chapter), and its "Quick Analysis" features (described in Chapter 16)
- how to set up ands run the Board Wizard to do detailed simulation that check for signal-integrity and EMC violations
- how to view the Board Wizard's results in BoardSim
- how to view the Board Wizard's results in another application, like Microsoft Excel

Comparison of Detailed-Simulation and Quick-Analysis Features

The Board Wizard's detailed-simulation features perform complete signal-integrity and EMC simulations on every net for which you enable them. This type of batch-mode simulation is in contrast to the Quick Analysis features in

the Board Wizard (described in Chapter 16), which do not perform detailed simulations but run very fast. If you enable the Board Wizard's detailed-simulation features, the time required for the Board Wizard to run will increase substantially (to perhaps multiple hours, if you enable simulations on a large number of nets). However, the analysis will be much more thorough, since actual, complete simulations will be run for every selected net.

Compared to the Quick Analysis options in the Board Wizard, the detailed features trade execution time for thoroughness of analysis. These features run the same kind of oscilloscope and spectrum-analyzer simulations you would if you analyzed the signal-integrity- and EMC-critical nets on your board one-by-one, interactively — except that the Board Wizard runs them in true batch mode, without manual intervention.

Overview of How Detailed Simulation Works in Board Wizard

The Quick Analysis options in the Board Wizard run on every net on your board. No setup (e.g., model selection) is required for these quick-analysis features, although the quality of the analyses improves if more information is provided (IC models, driver location, etc.) By contrast, the Board Wizard's detailed-simulation features must be enabled for specific nets on your board. They do not run on nets for which they have not been enabled.

Requirements for Simulating a Net

Some setup is required for nets on which the Board Wizard is to run detailed simulation. Specifically, before a net can be simulated, the net must:

- be explicitly selected for simulation
- have at least a driver-IC model selected (model and I/O direction); see section "Setting Up IC Models before Running the Board Wizard" below for details

In addition, to run an EMC simulation on a net, the net must:

- have a repetitive clock rate and duty cycle specified

Nets for which any of these requirements are missing will not be simulated (an error will be generated in the Board Wizard's report file). For details on setting up nets for the detailed simulation in the Board Wizard, see "Setting Up for the detailed Simulations" below.

EMC Simulations Available in BoardSim EMC Only; Crosstalk Simulations in BoardSim Crosstalk Only

In the descriptions that follow, the signal-integrity and EMC (radiated emissions) aspects of the Board Wizard's detailed simulations are discussed. However, the EMC features are available only if you are licensed for BoardSim EMC.

The Board Wizard can also run detailed crosstalk simulations. For complete details on this feature, see the Crosstalk User's Guide. The crosstalk features are available only if you are licensed for BoardSim Crosstalk.

How Detailed Batch-Mode Simulations Run

After you enable the Board Wizard for detailed simulations, set up nets for it, choose several "global" simulation options, and start the Board Wizard, BoardSim begins running detailed signal-integrity and EMC simulations. (See "Setting Up for Detailed Simulations" below and following sections for how to setting up nets and options.) This proceeds as if you were manually running simulations in the oscilloscope and spectrum analyzer, except that no interactivity is required: BoardSim selects nets for you, automatically launches simulations, and stores the results in a report file. Simulations are automatically run for both a rising and falling driver transition.

Another important difference is that the oscilloscope and spectrum analyzer are not visible while Board Wizard simulations run. Everything occurs in the background, in "batch mode." Unlike with interactive simulation, analysis results are presented in a report, textually, rather than visually in the oscilloscope/spectrum analyzer. See "Viewing the Board Wizard's Results" below in this chapter for details.

Setting Up IC Models before Running the Board Wizard

For detailed simulations in the Board Wizard —signal-integrity or EMC — the presence of IC models is required. Therefore the first step in running detailed batch-mode signal-integrity or EMC simulation is setting up models.

Step #1: Loading IC Models onto Pins

The first step in setting up IC models is to load them onto the IC pins that will be involved in the simulations you plan to run. Note that if you intend to simulate only a subset of your board's nets using the Board Wizard, there's no need to load models onto all of the IC pins on your board: concentrate only on pins attached to the nets you'll be analyzing.

The easiest way to attach large numbers of models to pins is to use the .REF file (i.e., "IC AutoMapping"). This feature allows you to attach models to every pin on an IC with a single-line entry in the .REF file; BoardSim includes a smart editor that makes this mapping especially easy. For details on using .REF files, see Chapter 9.

Another way to attach models to pins is interactively, using the Assign Models dialog box (see Chapter 8, section "Interactive versus Automatic Selection of IC Models" for details). This method is especially appropriate if you are simulating only a small number of nets. You can mix the .REF-file and interactive methods; models specified manually in the Assign Models dialog box take precedence over assignments made in the .REF file.

In a Hurry?: Workaround for Quickly Specifying IC Models

If you're in a hurry to get simulation results and don't have time to find exact IC models, consider supplementing with models from library EASY.MOD. Or you can even run all of your analysis using EASY.MOD models. The results will not be as accurate as if you had taken the time to specify correct models, but the time-savings can be large and the results are usually at least fairly good.

If most or all of the ICs on your board switch at the same rate (i.e., with approximately the same switching time), you may be able to use one model for every IC. Applying the same model repeatedly to every IC reference designator on your board is easy using the .REF-file editor: choose the model in the Library and Component/Model combo boxes, highlight the first reference designator in the Reference Designator/Part Name list box, then click Paste Model(s) repeatedly.

Even if you model all ICs using one model from EASY.MOD, you must still specify buffer directions as described in the next section.

Step #2: Setting Driver-IC Buffer Directions

After IC models are loaded (see the preceding section), the second step in setting up IC models for batch-mode simulation is to specify which IC pins are drivers. Specifically, in order for the Board Wizard to know which ICs on a given net are the drivers, you must manually set each driver's buffer direction to "Output." The only exception is for IC pins which are modeled with IBIS or .PML models whose buffer direction is output-only; these will load automatically as outputs. However, all other pins must be explicitly set to direction "Output." This includes any pins with IBIS or .PML models of type "I/O" — unless you tell it explicitly, the Board Wizard does not know whether I/Os should drive or receive during simulation.

Buffer directions are set manually in the Assign Models dialog box. If you are unfamiliar with how to set directions or the operation of the dialog box, see Chapter 8, section "Interactively Choosing IC Models."

Note: *This restriction — that you must explicitly set driver ICs to buffer direction "Output" — will likely be removed in a future version of the product. In particular, two helpful features are planned: "Auto Driver Direction," which will remove the requirement for driver direction to be specified by the user (BoardSim itself will "turn on" drivers); and a "round-robin" capability, which will cycle through multiple drivers one-at-a-time, if multiple drivers are found on a given net.*
If you have ideas about how you'd like BoardSim to treat the setup of driver ICs, please contact us at support@hyperlynx.com.

When Board Wizard simulates a net (and the nets coupled to it), and no driver ICs are present, simulations will stop and record an error in the batch-mode report file. **Remember, the Board Wizard does not know which ICs on your nets you want to drive with (except for output-only models); you must tell it before starting batch-mode simulation. It is not enough to have a model loaded onto the net; the model's buffer direction must be set to "Output" (or "Output Inverted").**

Setting Up for Detailed Simulations

The Board Wizard's detailed-simulation features run only on nets which you have specifically selected batch-mode signal-integrity and/or EMC simulation. Additionally, certain information must be known about each selected net before simulation can run. Finally, there are several options that apply "globally" to the simulation of every net that must be set.

To set up detailed simulations, and then enable detailed simulation for signal-integrity and/or EMC analysis:

1. From the Wizards menu, choose Board Wizard.
   OR
   Click the Run Board Wizard button on the toolbar.
   The Board Wizard dialog box opens.

2. In the Detailed Simulations area, if you plan to have the Board Wizard run signal-integrity analysis on any nets, click on the Run Signal-Integrity and Crosstalk Simulations on Selected Nets check box to enable it.

3. If you plan to have the Board Wizard run EMC analysis on any nets, click on the Run EMC Simulations on Selected Nets check box.

4. Optionally, you can also enable any of the features in the Quick Analysis area (see Chapter 16 for details on these features).

5. Click Next. Then follow the instructions in sections "Setting Up for Signal-Integrity Simulations" and "Setting Up for EMC Simulations" below.

Setting Up for Signal-Integrity Simulations

If you enabled signal-integrity simulation in step 2 of the instructions in section "Setting Up for Detailed Simulations" above, then clicked Next, the Board Wizard advances to the Batch-Mode SI and Crosstalk Options page. On this page, you can enable specific types of signal-integrity simulation, and choose on which nets to run analysis.

To enable specific types of signal-integrity simulation:

1. Verify that the Board Wizard is on the Batch-Mode SI and Crosstalk Options page. Enable any or all of the first three boxes in the Signal-Integrity Options area: Simulate Nets Using Fast-Strong; Simulate Nets Using Typical; and/or Simulate Nets Using Slow-Weak. For details on these options, see "Signal-Integrity Options: IC Strengths" below.

   You can also set the Report Delays Relative to Driver check box as desired. Enabling the check box means that the "starting time" for each delay measurement is the time when the driver crosses Vmeasure; disabling the check box means measure all delays from t=0. (For more details, see section "Delays: Driver-Relative versus Zero-Time-Relative" below.)

   Leave the Run at High Accuracy and Crosstalk Options check boxes disabled, unless you are licensed for BoardSim Crosstalk and plan to run coupled-mode/crosstalk simulations also. For details on these features, see the Crosstalk User's Guide.

When you enable these check boxes, you are enabling signal-integrity and EMC simulations *in general*. You still must specify on exactly which nets you want to run either or both kinds of analysis. If you plan to run no signal-integrity on any nets (you're doing only EMC simulations), then you can leave the Signal Integrity check box disabled; and vice versa if you are only doing signal-integrity simulation.

Signal-Integrity Options: IC Strengths

In step 1 above, you enabled any or all of three signal-integrity simulation options. These options tell the simulator at which strength(s) to run IC models during simulation. The "strengths" correspond to the three IC "operating

parameters" supported by BoardSim (see Chapter 12, section "Setting IC Operating Parameters" for details).

For each check box you enable, the Board Wizard runs two simulations per selected net: a rising-edge simulation with the IC models set as specified, and a falling-edge simulation. For example, if you enable Simulate Nets Using Fast-Strong Driver IC, then for any net selected for detailed simulation, the Board Wizard sets all of the ICs on the net to use their "fastest" and "strongest" characteristics; then runs a rising-edge simulation; then runs a falling-edge simulation. (For details on what combinations of best- and worst-case characteristics make up Fast-Strong and Slow-Weak IC operation , see Chapter 12, section "Setting IC Operating Parameters.")

If you are interested in best- and worst-case "corner" simulations, you would probably enable both the Fast-Strong and Slow-Weak options (and skip Typical, since its results are likely not to result in any extremes). On the other hand, if you're interested in a quicker batch-mode run and need only approximate results, you might enable only Typical simulations and avoid the overhead of running at multiple IC operating points.

Performance Status Bar

As you enable more simulation options, the resulting batch-mode simulation will get slower and slower. (This occurs because each simulation type requires two complete simulation runs per net.) For convenience, a "Fast-Slow" progress bar displays below the simulation-type check boxes. As you enable each simulation option, the bar moves progressively to the right to show relatively how much simulation time will be required per net. The closer to the "Slow" side the bar moves, the longer you can expect your batch run to take.

Limiting Per-Net Simulation Times

The Board Wizard lets you specify a maximum amount of time spent on any one net's simulations. If a net's simulations exceed this limit, the Board Wizard will abandon the net and move on to the next selected net.

This feature is mostly applicable to crosstalk simulations, in which large numbers of nets (the selected net plus potentially many aggressor nets) are

simulated simultaneously. For signal-integrity and EMC simulations, any reasonable time limit is unlikely to be exceeded.

To set the per-net maximum simulation time:

1. In the SI Settings for Each Net area, in the Max Run Time edit box, type the value, in minutes, of the run-time limit you want to impose.

Since time limits rarely apply to signal-integrity and EMC simulations, you can usually leave the time limit at its default value.

Enabling Nets for Signal-Integrity Analysis

The Batch-Mode SI and Crosstalk Options page gives you access to a spreadsheet interface (called the "Nets Spreadsheet") which lists every net on your board and allows you to choose which of those nets you wish to perform signal-integrity simulation on. Also, you can choose against what "compliance rules" (i.e., electrical limits) you want to test each net.

Analyzing *Every* Net Not Recommended

One option in the Nets Spreadsheet is to enable every net on your PCB; however, there are major disadvantages to this strategy.

If you enable all nets on your board rather than only those which it is critical to analyze, then:

- batch-mode analysis will run longer
- you will need to set up more IC models
- the resulting report files will be longer and more-difficult to read

A better approach is to select for analysis only those nets about whose signal-quality you are truly concerned. It is true that in some high-speed designs, nearly every signal is critical; then it may be necessary to enable all nets for batch-mode simulation. However, many boards have only a subset of nets whose signal quality is worth analyzing in detail.

You can save yourself effort in both setup and ease of interpreting results if you spend a small amount of time up-front thinking about which nets on your board are truly critical, and choosing only them for batch-mode analysis.

Opening the Nets Spreadsheet

To open the Nets Spreadsheet for signal-integrity analysis:

1. From the Batch-Mode SI and Crosstalk Options page, in the SI Settings for Each Net area, click Nets Spreadsheet. The spreadsheet opens.

The spreadsheet is sizable. To make maximum use of space and see the most nets at a time, maximize the spreadsheet to be full-screen.

How the Spreadsheet Works

Net Names

In the left-hand column labeled "Net Name," the Nets Spreadsheet lists every net on your PCB, *except* nets which you or BoardSim have identified as power supplies (for details on editing the list of power supplies, see Chapter 6, section "Editing Power-Supply Nets").

Sorting Net Names and Other Items

You can sort the contents of any column in the Nets Spreadsheet by clicking on the column's header button, at the top of the column. Clicking once sorts in ascending order; clicking again sorts in descending order; clicking a third time returns to ascending; and so forth.

For example, to sort the list of net names from A-Z, click once on the Net Name button at the top of the left-hand column. To re-sort from Z-A, click the button again. Clicking a third time returns to A-Z sorting.

Net Statistics

Immediately adjacent to the Net Name column are two columns showing each net's width and length. These are display-only columns, i.e., you cannot enter data in them. These columns are useful for sorting nets in an electrically meaningful manner, e.g., so that the longest nets appear at the top of the list.

Note: *For nets with multiple segment widths, the Width column shows the widest segment.*

Value of Sorting Nets by Length

Sorting nets by length is often a valuable exercise, because on high-speed boards, the longest nets often have the most signal-quality problems (a basic consequence of transmission-line theory). If you are analyzing a PCB for which you do not have a good understanding of the critical nets, sorting by length will bring to the top of the rules list some likely candidates for analysis.

Enabling Nets for Signal-Integrity Simulation

To enable a net for signal-integrity simulation:

1. To enable a net for signal-integrity simulation, click the check box in the net's SI Enable column. A red check mark indicates that the net is selected and will be simulated in detail once the Board Wizard begins running.

To disable a net that has already been selected:

1. Click the net's SI Enable check box again. The red check mark disappears, and the net will not be simulated by the Board Wizard.

Note: *The abbreviation "SI" for "signal integrity" is used in several places in the Board Wizard.*

Enabling a Net Also Enables Associated Nets

If you enable a net for analysis that has one or more associated nets (see Chapter 7, section "What are Associated Nets?" for an explanation of the term), the associated nets will also be enabled at the same time. Similarly, if you disable a net, its associated nets are also disabled.

This behavior is required because associated nets are simulated together, as a group. Enabling or disabling one necessarily does the same to the others.

Viewing Which Nets are Enabled

If you enable a number of nets throughout the list for analysis, and want to see a summary of which ones you selected, click the button at the top of the SI Enable column. This will sort the nets to bring the selected ones to the top of the list.

Setting Net-by-Net Signal-Integrity Compliance Rules

What are "Compliance Rules"?

For every net it simulates, the Board Wizard gives two kinds of information:

- a tabular summary of the waveform that resulted, including information about pin-to-pin delays and overshoot
- optionally, a warning if user-defined "compliance rules" (e.g., maximum allowed delay or overshoot) are exceeded

If you were to run a batch-mode simulation without any rules set, you would get tabular data for each net simulated, but no warnings about compliance violations. The value of setting compliance rules is that it enables the Board Wizard to highlight (with a warning) every net that exceeds the electrical limits you impose. When batch simulation is completed, you can scan the report file and quickly identify the "offending" nets, which you may then want to simulate further, try applying terminators to, etc.

When you first enable a net for analysis, its compliance rules are set to reasonable default values. If you want different values, you can change them net-by-net.

Note: *Actually, there is no way to completely "turn off" a net's rules, but you can make them unlikely to ever be violated by setting them to extreme values. See below for more details.*

Types of Signal-Integrity Compliance Rules

The Board Wizard supports the following types of compliance rules:

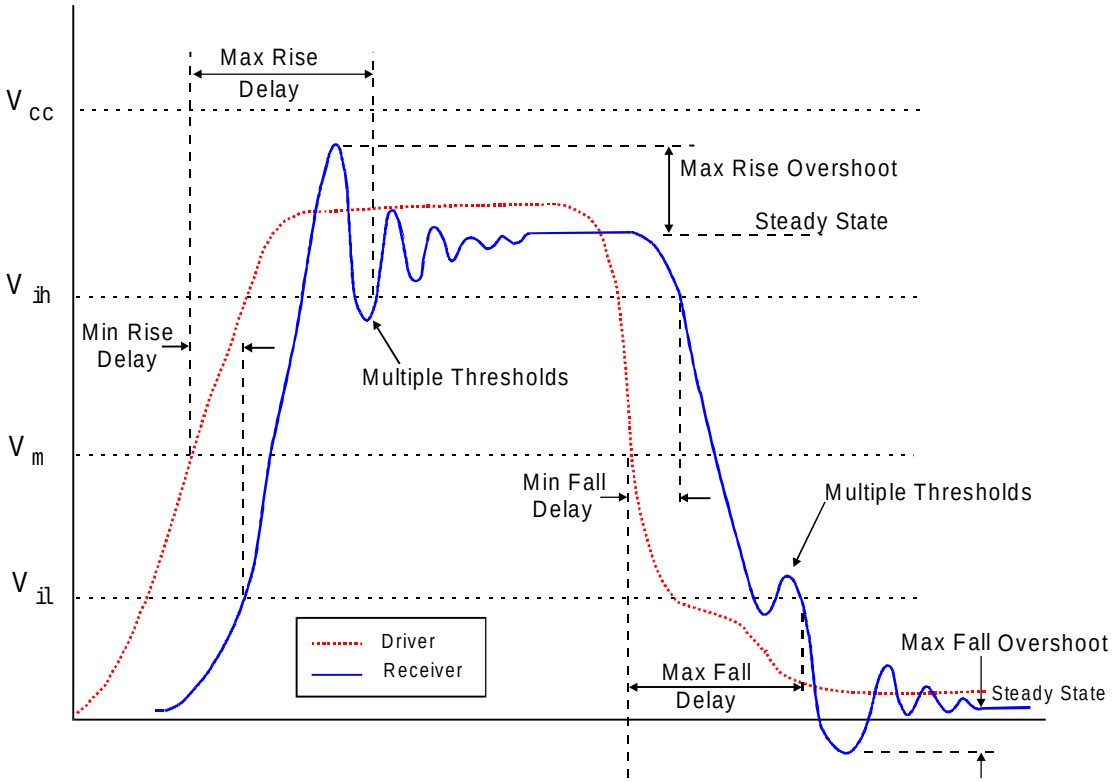
Maximum rising-edge	For all receivers on the net, specifies the maximum

overshoot	voltage by which the signal can go *above* the final DC value
Maximum falling-edge overshoot	For all receivers on the net, specifies the maximum voltage by which the signal can go *below* the final DC value
Maximum pin delay	Specifies the maximum delay to any receiver on the net; measured at each receiver from the time the driver switches until the receiver's *farther* threshold is crossed for the *final* time
Minimum pin delay	Specifies the minimum delay to any receiver on the net; measured at each receiver from the time the driver switches until the receiver's *nearer* threshold is crossed for the *first* time

Note: *There is also a maximum-crosstalk compliance rule. However, it is only available to users of BoardSim's Crosstalk option. See the Crosstalk User's Guide for more details.*

Figure 17-1 shows graphically for hypothetical driver and receiver signals how overshoots and pin delays are defined.

Figure 17-1: How Overshoots and Pin Delays are Defined for Batch-Mode Simulation



Overshoot Rules

The overshoot rules (rising and falling) set a limit for how far beyond the final DC voltage the signal at any receiver on the net can go. For a rising edge, the limit is on how far above the final DC high voltage any receiver-pin signal can go; for a falling edge, the limit is on how far below the final DC low voltage.

To enter a new overshoot compliance rule for a net:

1. In the SI Overshoot Rise or SI Overshoot Fall cell for the net, type the desired maximum value, in mV.

Note that overshoot is judged relative to the final DC voltage to which a switching edge is transitioning, not to the power-supply voltage. This method of judging overshoot keeps it meaningful for devices that switch to levels different than the supply voltages.

Delay Rules

The delay rules let you set constraints on the minimum and maximum delays to any receiver pin on the net.

Delays are measured relative to receiver-input and driver-switching thresholds contained in the IC models you specified for the net being simulated. (For details on choosing IC models, see Chapters 8 and 9.)

The IC-model thresholds involved in delay calculations are:

"Vmeasure"	for driver-output models; specifies the voltage at which the driver is considered "switched"; set to 1.5V for most ICs; can also often be calculated as (high input threshold + low input threshold) / 2
high and low input thresholds	for receiver-input models; specify the lowest and highest voltage at which the receiver recognizes a state change
high and low hysteresis thresholds	for receiver-input models; specify the alternate thresholds which apply to generate hysteresis; not present in most models

Given the above thresholds in a net's IC models, the *maximum* delay for a specific receiver pin is measured as follows:

- find the time at which the input signal crosses the receiver threshold furthest from the initial DC voltage (uses furthest threshold to get most-pessimistic delay)
- if the input signal crosses the further threshold multiple times, take the delay as the time of the *last* crossing

- subtract from the just-found delay the time required for the driver to switch (i.e., the time at which the driver crosses threshold Vmeasure)

The *minimum* delay for a specific receiver pin is measured as follows:

- find the time at which the input signal crosses the receiver threshold nearest to the initial DC voltage (uses nearest threshold to get most-optimistic delay)
- even if the input signal crosses the nearest threshold multiple times, take the delay as the time of the *first* crossing
- subtract from the just-found delay the time required for the driver to switch (i.e., the time at which the driver crosses threshold Vmeasure)

***Note**: If you run all three IC-model "strengths" in one batch run (min, typ, and max — see "Signal-Integrity Options: IC Strengths" above — then for maximum conservativeness, **all** delays are calculated from the **smallest** driver-switching time, regardless of from which IC strength it comes (usually "min").*

To enter a new delay compliance rule for a net:

1. In the SI Pin Delay Max or SI Pin Delay Min cell for the net, type the desired value, in ns.

Specifying No Overshoot or No Delay Checking

On some nets, you may not care about overshoot or delay values at all. Or, on certain nets, you may want to for example, check for maximum delay, but not care about minimum. You can specify these conditions by entering very large overshoot values, or very large or negative delay values, e.g., 10V for overshoot, or 1000 ns for maximum delay or -5 ns for minimum.

BoardSim defaults the delay-rule entries to "turned-off" values when you first enable signal-integrity analysis on a net. Therefore, *not* checking for a delay is the default state for a net.

To "turn off" maximum overshoot checking (rise or fall) on a net:

1. Set the net's SI Overshoot Rise or SI Overshoot Fall value to a very large value, e.g., 10 V.

To "turn off" maximum-delay checking on a net:

1. Set the net's SI Pin Delay Max value to a very large value, e.g., 1000 ns.

To "turn off" minimum-delay checking on a net:

1. Set the net's SI Pin Delay Min value to a negative value, e.g., -5.0 ns.

Delays: Driver-Relative versus Zero-Time-Relative

The delays calculated by the Board Wizard must be measured relative to some "starting" time. Normally, you'll want this to be the time at which the driver IC switches, so that the delay at a receiver IC represents the *extra* time, after the driver IC switches, at which the receiver switches. By default, this is how the Wizard calculates delays.

Note: *A driver IC is considered to have "switched" when its output voltage crosses a threshold value specified by the IC's manufacturer. This threshold is called "Vmeasure" and is specified in the IC's model. For more information about Vmeasure, see Chapter 10, section "Measurement Thresholds and Loads."*

However, you can optionally change delay calculations to be relative to time "0.0" in the simulation. Then, the delay at a receiver IC represents that amount of time it takes the driver IC to switch *plus* the additional time before the receiver switches.

To change between driver-relative and zero-time-relative delays:

1. In the Board Wizard, on the Batch-Mode SI and Crosstalk Options page, click on the Report Delays Relative to Driver check box to toggle it. Enabling the check box means that the "starting time" for each delay measurement is the time when the driver crosses its Vmeasure threshold; disabling the check box means that all delays are measured from t=0.

Setting Values in an Entire Spreadsheet Column

Sometimes, when working in the Nets Spreadsheet, you will want to set every net's entry in a given column to the same value. For example, you may want to set every net's maximum-overshoot compliance rule to 200 mV (a change from the default value), even if you don't intend to simulate every net.

To set the same value for every net in a spreadsheet column:

1. In the Nets Spreadsheet, click once on the heading button (the area that labels the column, at the top). This "selects" the column.

2. From the Column menu, choose Set Selected Column To. A dialog box opens.

3. In the dialog box, if the desired column value is numeric, type the new number; or if "binary" (i.e., true or false), click the appropriate radio button.

4. Click Apply. The dialog box closes and the column is filled with the new value. The value is displayed only for nets that are currently enabled for simulation.

Sometimes you may want to set an entire column back to its default value.

To reset every net in a spreadsheet column back to its default value:

1. In the Nets Spreadsheet, click once on the heading button (the area that labels the column, at the top). This "selects" the column.

2. From the Column menu, choose Set Selected Column To. A dialog box opens.

3. In the dialog box, click Apply Default. The dialog box closes and the column is filled with the default value. The value is displayed only for nets that are currently enabled for simulation.

There is also a way to set every entry in the *entire* spreadsheet (rather than just one column) back to default values.

To reset every entry in the entire spreadsheet back to default values (rather than just one column):

1. From the Sheet menu, choose Set Entire Sheet to Default.

All cells in the spreadsheet are set to their default values.

Setting Up for EMC Simulations

EMC simulations — if you enabled them in step 3 of section "Setting Up for Detailed Simulations" above — are set up in detail on the Batch-Mode EMC Settings page of the Board Wizard. To reach this page (assuming you are starting the first page of the Wizard):

- if you enabled only EMC simulation (no signal-integrity simulation), click Next
- if you enabled EMC and signal-integrity simulation, follow the steps in section "Setting Up for Signal-Integrity Simulations" above, then click Next

On the Batch-Mode EMC Settings page, you can enable the antenna and other analysis characteristics, and choose on which nets to run analysis.

To set antenna and other analysis characteristics:

1. Verify that the Board Wizard is on the Batch-Mode EMC Settings page.
2. In the Driver Stimulus area, click on the desired IC strength to be used during simulation. For details on these choices, see Chapter 12, section "Setting IC Operating Parameters". For worst-case EMC simulations, the Fast-Strong Setting is the best choice because it usually generates the largest currents (and therefore the most radiation) during simulation.
3. Pull down the Distance from Antenna to PCB combo box, and choose one of the standard antenna-distance settings. If you plan to test your board in an EMC lab, you may want to choose the same distance for the Board Wizard that you will use in your lab testing.
4. In the Regulatory Test Limits area, click on the check boxes for the regulatory limits against which you want to test (FCC, CISPR, etc.). You

can choose one or multiple tests. Also, in the Class area, click on the check box for either or both of the Class A/Class B choices. If you choose multiple regulatory limits and/or classes, the Board Wizard will check every net individually against all test choices. (For details on the regulatory limits and classes, see the EMC Analysis User's Guide. The regulatory limits are scaled appropriately depending on the antenna-distance choice you made in step 3.)

5. In the Include Radiation From area, click in the check boxes to enable/disable sensing of radiation from board traces and component packages. (On some modern boards, as much or more radiation emanates from packages, especially IC packages, as from board traces.) The Multipath from Earth Ground setting should almost always be left enabled. For details on these settings, see the EMC Analysis User's Guide.

Note: *The Board Wizard's EMC-analysis engine uses only antenna probing. The current probe — an option in interactive simulation — is not available in batch mode.*

Enabling Nets for EMC Analysis

The Batch-Mode EMC Settings page gives you access to a spreadsheet interface (called the "Nets Spreadsheet") which lists every net on your board and allows you to choose which of those nets you wish to perform EMC simulation on. Also, you can choose against what "compliance rules" (i.e., electrical limits) you want to test each net.

Analyze Only EMC-Critical Nets

One option in the Nets Spreadsheet is to enable every net on your PCB; however, for EMC simulation, this makes no sense. EMC simulations are inherently time-consuming; running detailed simulations on every net on your board would take prohibitively long. Also, it is only periodic signals (e.g., clocks and other regularly strobed signals) which generate sharply peaked amounts of radiation. Random signals generally have their radiant energy distributed in a wider, less-peaked manner. Therefore, it usually makes sense only to simulate the key, periodic signals on your PCB. For more details, see the EMC Analysis User's Guide.

You can save yourself effort in both setup and ease of interpreting results if you spend a small amount of time up-front thinking about which nets on your board are truly critical, and choosing only them for batch-mode analysis.

Opening the Nets Spreadsheet

To open the Nets Spreadsheet for EMC analysis:

1. From the Batch-Mode EMC Settings page, in the EMC Settings for Each Net area, click Nets Spreadsheet. The spreadsheet opens.

The spreadsheet is sizable. To make maximum use of space and see the most nets at a time, maximize the spreadsheet to be full-screen.

How the Spreadsheet Works

Net Names

In the left-hand column labeled "Net Name," the Nets Spreadsheet lists every net on your PCB, *except* nets which you or BoardSim have identified as power supplies (for details on editing the list of power supplies, see Chapter 6, section "Editing Power-Supply Nets").

Sorting Net Names and Other Items

You can sort the contents of any column in the Nets Spreadsheet by clicking on the column's header button, at the top of the column. Clicking once sorts in ascending order; clicking again sorts in descending order; clicking a third time returns to ascending; and so forth.

For example, to sort the list of net names from A-Z, click once on the Net Name button at the top of the left-hand column. To re-sort from Z-A, click the button again. Clicking a third time returns to A-Z sorting.

Net Statistics

Immediately adjacent to the Net Name column are two columns showing each net's width and length. These are display-only columns, i.e., you cannot enter data in them. These columns are useful for sorting nets in an electrically meaningful manner, e.g., so that the longest nets appear at the top of the list.

Note: *For nets with multiple segment widths, the Width column shows the widest segment.*

Enabling Nets for EMC Simulation

To enable a net for EMC simulation:

1. To enable a net for EMC simulation, click the check box in the net's EMC Enable column. A red check mark indicates that the net is selected and will be simulated in detail once the Board Wizard begins running.

To disable a net that has already been selected:

1. Click the net's EMC Enable check box again. The red check mark disappears, and the net will not be simulated by the Board Wizard.

Enabling a Net Also Enables Associated Nets

If you enable a net for analysis that has one or more associated nets (see Chapter 7, section "What are Associated Nets?" for an explanation of the term), the associated nets will also be enabled at the same time. Similarly, if you disable a net, its associated nets are also disabled.

This behavior is required because associated nets are simulated together, as a group. Enabling or disabling one necessarily does the same to the others.

Viewing Which Nets are Enabled

If you enable a number of nets throughout the list for analysis, and want to see a summary of which ones you selected, click the button at the top of the EMC Enable column. This will sort the nets to bring the selected ones to the top of the list.

Setting Net-by-Net EMC Stimulus

The rules against which EMC simulations are run are global (i.e., one set applies to all nets on the board; see the steps above ("To set antenna and other analysis characteristics") for details). However, for each net enabled for EMC analysis, you must specify on a net-by-net basis at what clock frequency and duty cycle the net is stimulated.

Note: *The requirement that you enter a clock frequency for any net to be EMC analyzed implies that only nets with periodic signals can be analyzed. This is appropriate, since nets with random signals rarely cause EMC problems. However, for a more-detailed discussion of this issue, see the EMC Analysis User's Guide.*

To specify a net's clock frequency:

1. In the EMC Clock Freq cell for the net, type the desired frequency, in MHz.

To specify a net's duty cycle:

1. In the EMC Clk Duty Cycle box for the net, type the desired duty cycle, in percent. The duty-cycle value represents the percentage of time that the signal will spend in the high state.

Setting Values in an Entire Spreadsheet Column

Sometimes, when working in the Nets Spreadsheet, you will want to set every net's entry in a given column to the same value. For example, you may want to set every net's duty cycle to 200 mV (a change from the default value), even if you don't intend to simulate every net.

To set the same value for every net in a spreadsheet column:

1. In the Nets Spreadsheet, click once on the heading button (the area that labels the column, at the top). This "selects" the column.

2. From the Column menu, choose Set Selected Column To. A dialog box opens.

3. In the dialog box, if the desired column value is numeric, type the new number; or if "binary" (i.e., true or false), click the appropriate radio button.

4. Click Apply. The dialog box closes and the column is filled with the new value. The value is displayed only for nets that are currently enabled for simulation.

Sometimes you may want to set an entire column back to its default value.

To reset every net in a spreadsheet column back to its default value:

1. In the Nets Spreadsheet, click once on the heading button (the area that labels the column, at the top). This "selects" the column.

2. From the Column menu, choose Set Selected Column To. A dialog box opens.

3. In the dialog box, click Apply Default. The dialog box closes and the column is filled with the default value. The value is displayed only for nets that are currently enabled for simulation.

There is also a way to set every entry in the *entire* spreadsheet (rather than just one column) back to default values.

To reset every entry in the entire spreadsheet back to default values (rather than just one column):

1. From the Sheet menu, choose Set Entire Sheet to Default.

All cells in the spreadsheet are set to their default values.

Saving Settings in the Nets Spreadsheet

The Nets Spreadsheet saves your settings when you close it, unless you specifically tell it not to save them.

To close the Nets Spreadsheet and save changes to settings:

1. From the File menu in the spreadsheet, choose Close.
   OR
   Click the close button in the upper-right corner of the spreadsheet window. A dialog box asks if you want to save your changes; click Yes.

To close the Nets Spreadsheet *without* saving changes to settings:

1. From the File menu in the spreadsheet, choose Cancel.
   OR

Click the close button in the upper-right corner of the spreadsheet window. A dialog box asks if you want to save your changes; click No.

Running the Board Wizard

Check Power Supplies Before Running

IMPORTANT! Before running the Board Wizard, you must check to see that all power-supply nets are correctly identified to BoardSim. (For details on editing the power-supplies list, see Chapter 6, section "Editing Power-Supply Nets".) If a power-supply net that connects to many other nets through components like resistors and capacitors is left unidentified, the Board Wizard will run for a very long time when it analyzes the net (the Wizard may appear to be "hung").

Running the Wizard

To run the Board Wizard:

1. Run the preliminary steps in the Wizard to choose simulation types, enable specific nets for simulation, and set other parameters. See the preceding sections for detailed instructions.

2. Click ahead in the Wizard until the final, Batch-Mode Analysis page appears. Then click Finish. The Board Wizard begins running on your board, showing its progress as it runs. For large boards, the Wizard may take some time to run.

   The Batch-Mode Analysis dialog box shows your progress. Specifically:

   — in the Analysis Status area, the Percent Done Processing Board status bar shows the overall progress on your board, i.e., what percentage of your board's net have been analyzed

   — in the Net Being Analyzed area, the Percent Done Processing Net status bar shows the progress on the net that is presently being processed; this

status bar may run multiple times per net; detailed messages appear below the status bar

3. When the batch run is completed, it automatically opens its report file in the HyperLynx Report File Viewer for viewing. If the report is very large, it may take a while to load even after the Wizard itself has stopped running.

Stopping the Board Wizard

You can force the Board Wizard to stop running before it has completed analysis of your entire board (e.g., if you remember that you set something up incorrectly, or if the Wizard is taking longer to run than you expected).

To stop the Board Wizard before it has completed its analysis:

1. Click the Cancel button. A message appears briefly, saying that the Wizard will stop after it finishes analysis of the present net.

When the Wizard has finished with the present net, it stops and opens its report for viewing. For details on the report, see below.

Viewing the Board Wizard's Results

When you run the Board Wizard, it writes its results into an ASCII report file called <*HYP_file_name*>.RPT, where <*HYP_file_name*> is the name of your board's .HYP file. When the Wizard has finished running and the report file is complete, the Board Wizard automatically opens the report for viewing in the HyperLynx Report File Viewer.

Changing the Name of the Report File

You can change the default name of the report file, if you want.

To change the name of the report file:

1. On the Batch-Mode Analysis page in the Board Wizard, just before clicking the Finish button to begin running the Wizard, type a different name in the Report File Name edit box.

Format of the Wizard's Report for Signal-Integrity Simulations

Results Table for Each Net

For signal-integrity simulations, the report file contains a concise table summarizing the simulation data for each net that was simulated. The table is titled "Signal-Integrity Simulation Results" and shows the following information:

Net Type	Results
Device.Pin	A list of all the IC pins (driver and receiver) on the selected net and its associated nets; pins are named as<reference designator>.<pin name>
Dir	The pin's direction; "out" means the pin is driving, "in" means receiving; the suffix "df" is added to designate differential pins
Delay Rise	Minimum and maximum delays to each pin on the net, for a rising-edge transition of the driver (for details on how delays are calculated, see Figure 1 above in this chapter)
Delay Fall	Same as Delay Rise, except data is for a falling-edge transition of the driver
Overshoot	The maximum overshoot, in the rising and falling directions, beyond the final DC value (for details on how overshoot is calculated, see Figure 1 above in this chapter)
Crosstalk	The maximum crosstalk that occurred on the

Net Type	Results
	selected net; available only in the BoardSim Crosstalk product; see the Crosstalk User's Guide for details
ERROR FLAGS	A section summarizing any signal-integrity violations, versus the compliance rules you entered, for the net and its associated nets; see description below and the legend at the top of the report file for details on interpreting this section
max. rising overshoot allowed	The compliance rule you entered for maximum allowed overshoot on the rising edge
max. falling overshoot allowed	The compliance rule you entered for maximum allowed overshoot on the falling edge
min. delay allowed	The compliance rule you entered for minimum allowed pin delay, for rising and falling edges
max. delay allowed	The compliance rule you entered for maximum allowed pin delay, for rising and falling edges

Searching in the Report for Signal-Integrity Violations

If a net (and its associated nets) violates any of the compliance rules you specified (see section "Setting Net-by-Net Signal-Integrity Compliance Rules" above for details on setting rules), a warning is issued in the net's report table. There are two classes of warnings in the file; signal-integrity violations are generally considered "severe" warnings.

To search for warnings in the Board Wizard's report file, using the Report File Viewer:

1. From the Report File Viewer's Search menu, choose Find Warning.
   OR
   Click the yellow Find Warning button on the Viewer's toolbar.
   The editor jumps to the next warning below the cursor.

To search for the next warning in the report file:

1. From the Report File Viewer's Search menu, choose Find Next.
   OR
   Click the yellow Find Warning button on the Viewer's toolbar.
   OR
   Press F3.

To search for only "severe warnings":

1. From the Report File Viewer's Search menu, choose Find Warning Severe.
   OR
   Click the red Find Warning Severe button on the Viewer's toolbar.

Interpreting Violations

For nets that have signal-integrity violations, the ERROR FLAGS section of the net's Signal-Integrity Simulation Results table summarizes what kind(s) of violation(s) occurred. Each IC pin on the net has a violations line of its own; rising-edge and falling-edge violations are reported separately.

Types of Violations

The Board Wizard checks for the following kinds of signal-integrity violations:

Type	Description
delay violation	receiver-IC pin's maximum delay is longer than the net's compliance rule *OR* receiver-IC pin's minimum delay is shorter than the net's compliance rule (not checked for driver-IC pins)
threshold error	IC pin's signal level never reached the switching threshold
overshoot violation	IC pin's signal level exceeded the final DC level by more than the net's overshoot threshold
multi-threshold-	receiver-IC pin's signal level crossed the Vih or Vil

Type	Description
crossing error	threshold more than once during transition (not checked for driver-IC pins)

Note that two of these violations are checked against user-entered compliance rules (delay and overshoot), while two are automatically checked against IC-model threshold values (threshold and multi-threshold crossing). Also, two (threshold and overshoot) are checked at drivers and receivers, whereas two (delay and multi-threshold-crossing) are checked only at receivers.

For 3-state drivers that are 3-stated (i.e., set to be "off" rather than driving), only overshoot violations are checked for.

How Violations are Reported

The ERROR FLAGS section for each net shows which (if any) of these violations occurred during the Board Wizard's analysis. There is one group of flags for the rising switching edge and one for falling. If a violation occurs, a "code letter" is printed to the appropriate column; otherwise, the columns are marked with a hyphen ('—').

The following table shows the violation codes and what they mean (this information is also summarized at the top of the report file):

Code	Description
D	delay violation
T	threshold error
O	overshoot violation
M	multi-threshold-crossing error
X	crosstalk violation — *used only by BoardSim Crosstalk*
—	no error

When looking at a net's report data, you can quickly scan the ERROR FLAGS table and look for any column entries other than'—'. Any letters in a column indicate a violation. Check the detailed numerical information to the right to see the details of the violation, e.g., by how much a delay was too long, etc.

Also, if any violation occurs, then the message ** Warning(Severe) ** is printed into the net's section in the report. You can search on this message to quickly find nets with violations (see "Searching in the Report for Signal-Integrity Violations" above for the easiest ways to search).

About Negative Delays

If you are calculating delays driver-relative, it is possible for some receiver delays to be negative. This can occur especially at unterminated trace ends where the high impedance causes a doubling effect at the receiver and may cause it to cross its threshold before the driver crosses its threshold.

Format of the Wizard's Report for EMC Simulations

When the Board Wizard runs a detailed EMC simulation on a net, it automatically checks the net's radiation profile against the EMC compliance rules you chose when you set up the Wizard. If a net violates any one of the specified rules, the Board Wizard writes a warning into its report.

For EMC simulations, the report file contains a concise table summarizing the simulation data for each net that was simulated. The table is titled "EMC Simulation Results" and shows information as follows:

- If you enable multiple test limits (e.g., FCC and CISPR, both Class B; or FCC Class A and Class B), and a net exceeds any or all of the limits at any frequency, the Board Wizard issues a warning.
- The warning contains information about the worst-case excess, i.e., the radiation level at the frequency that most exceeded the smallest of the limits.
- The Board Wizard generates two categories of warning. If the worst-case excess frequency exceeded no limit by more than 6 dBuV/m (a linear factor of 2), a "normal" warning is generated. If a limit is exceeded by more than

6 dBuV/m, a “severe” warning is generated. Thus, if no net receives a severe warning, none of your nets is seriously above the selected test limits.

Searching in the Report for EMC Violations

See “Searching in the Report for Signal-Integrity Violations” above.

Opening an Existing .RPT File

To view an existing Board Wizard report using the Report File Viewer:

1. From the Wizards menu, choose Board Wizard.
   OR
   Click the Run Board Wizard button on the toolbar.
   The Board Wizard dialog box opens.

2. In the View Previous Board Report area, click Open. The Report File Viewer is launched.

The Open button automatically causes BoardSim to search for a file named <*HYP_file_name*>.RPT, where <*HYP_file_name*> is the name of the currently loaded .HYP file.

Viewing Detailed Results for a Net

Sometimes, if a particular net receives a Board Wizard violation in the report file, you want to see its simulation results in detail, i.e., see it in the oscilloscope or spectrum analyzer. To view the net in detail, close the report (you may want to print it first), and analyze the net interactively with the oscilloscope or spectrum analyzer.

For details on running the oscilloscope interactively, see Chapter 12. For details on running the spectrum analyzer interactively, see the EMC Analysis User’s Guide.

Viewing Board Wizard Results in Excel and Other Applications (.CSV File)

In addition to viewing the Board Wizard's report file in the HyperLynx report viewer or another text editor, you can also view some portions of the Wizard's output in Excel or other, similar Windows-based applications. The advantage of using a program like Excel is that it allows you to sort and perform other operations (like charting/graphing) on the results data.

To facilitate viewing in other programs, when you run the Board Wizard, BoardSim writes a comma-separated-values (.CSV) file which includes all of the signal-integrity violation data that also appears in the ASCII report file (see "Viewing the Board Wizard's Results" above for details on the report file). Excel (and most other spreadsheet/database programs) can read .CSV files directly.

Note: *In the examples that follow, instructions are given for reading data into Microsoft Excel, and manipulating it in Excel. Performing the same operations in any non-Excel application should be similar; see your particular application's manual or Help file for details.*

Reading Board Wizard Results into Excel

To read the Board Wizard's signal-integrity results into Microsoft Excel:

1. In Excel, choose Open from the File menu. Open file <*HYP_file_name*>-SI.CSV, where <*HYP_file_name*> is the name of your board's .HYP file. BoardSim creates this file in the same directory in which the .HYP file is located.

.CSV files read directly into Excel, so there is no translating or re-formatting of the data required.

The .CSV file contains all of the signal-integrity data generated by the Board Wizard. In the far left column are the names of the nets in the analysis (organized by groups of associated nets; see Chapter 7, section "What are

Associated Nets?" for details on what is meant by "associated net"). Additional columns give:

- IC pins (organized by reference designator and pin name)
- minimum and maximum delay values (for rising and falling edge)
- maximum overshoot (rising and falling edge)
- signal-integrity violation flags ('1' = violation occurred; '0' = no violation)

For details on the meaning of the various violation flags, see "Interpreting Violations" above.

Sorting Signal-Integrity Data in Excel

Once the signal-integrity data is read into Excel, you can sort the data by any column.

To sort the data by a particular column in Excel:

1. Click on any cell in the column by which you want to sort, so that the cell highlights. (Do not highlight the entire column, because Excel will think you want to sort just the highlighted column, which would "detach" that column from the net names and other columns in the file.)
2. From the Data menu, choose Sort. The Sort dialog box opens.
3. In the Sort by area, choose Ascending or Descending. Change other settings, too, if needed, if you want more-sophisticated sorting (multi-column, etc.) (In the My List Has area, do not click on Header Row, i.e., leave set to No Header Row, because the header rows in the .CSV file have deliberately been separated from the data so that they do not get mixed into the sorting.)
4. Click OK. The dialog box closes and the sorting occurs.

There are a number of useful ways to sort the data in the .CSV file. A few examples are:

- by maximum delay
- by maximum overshoot
- in descending order on any of the violations columns; this will bring all of the nets with violations to the top of the list

Exporting Board Wizard Delays in SDF Format

One of the Board Wizard's most-powerful signal-integrity features is the receiver-specific delays it calculates for every net on which simulation is run. These delays fully account for reflection effects, non-incident-wave switching, receiver thresholds, etc. See Figure 1 above in this chapter for details on how delays are calculated.

There are several possible uses for these delay values. One is to simply sort and view them (see section "Viewing Board Wizard Results in Excel" above). Another is to formally incorporate the delays into another EDA tool, e.g., a Verilog or VHDL timing simulator.

In order to make it easier to annotate delays calculated by the Board Wizard into an external EDA tool, the Wizard automatically creates (at the same time it creates its ASCII report file; see section "Viewing the Board Wizard's Results" above for details) a Standard Delay Format (.SDF) file summarizing all pin-to-pin delays. This file has name <*HYP_file_name*>-SI.SDF, where <*HYP_file_name*> is the name of your board's .HYP file. The file is created in the same directory in which the .HYP file is located.

For details on how to utilize the SDF file in an external EDA tool, refer to that tool's documentation.

Chapter 18: The HyperLynx File Editor

Summary

This chapter describes:

- why BoardSim includes a file editor
- how to open the HyperLynx File Editor
- how to use the Editor

Why a HyperLynx File Editor?

BoardSim includes a Windows file editor which can be used for viewing or editing files associated with BoardSim (e.g., the Board Wizard's report file, or a .HYP file).

There is no requirement to use the HyperLynx File Editor; you can view and edit files with any Windows editor you choose. However, the HyperLynx File Editor has some advantages over generic editors that are worth considering.

First, the HyperLynx File Editor is a text editor which not limited in the file sizes it can handle. Second, the Editor includes convenient searching features that are specialized to finding signal-integrity warnings in the Board Wizard's report file. Finally, the HyperLynx File Editor is convenient to use when you're running BoardSim because you can launch it from inside BoardSim.

Opening The HyperLynx File Editor

When the editor is used for viewing a file generated by BoardSim, e.g., the Board Wizard's report file, BoardSim will open the Editor automatically for you.

However, when you decide to create or edit a file with the Editor, you must open it yourself.

To open the HyperLynx File Editor:

1. From the File menu, choose File Editor.
   OR
   Click the File Editor button on the toolbar.

The Editor opens, ready to load a file or begin creating a new one.

Editor Functions as a "Child" of BoardSim

The HyperLynx File Editor, because it is launched from inside BoardSim, is a "child" application of BoardSim, not a completely independent Windows application.

This has two implications:

- if you iconize the Editor, its icon "belongs" to BoardSim, not to Windows; this means, for example, that the Editor's icon does not appear on the Windows tool bar
- if you have the Editor open and close BoardSim, the Editor is closed, too

Opening a File

When the HyperLynx File Editor first opens, you can begin typing immediately to create a new file. Or:

To open an existing file in the Editor:

1. From the File menu, choose Open.
   OR
   Click the Open button on the toolbar.

2. Change directories, if needed, and highlight the file you want to open.

3. Click Open.

The File opens in the Editor, ready for editing.

Setting Read-Only Mode

You can set the HyperLynx File Editor to read-only mode, so that it functions as a viewer, but not an editor. This prevents you from accidentally editing a file when all you really want to do is look at it.

To enable read-only mode:

1. From the Options menu, choose Read Only. The check mark appears, to indicate that the editor is functioning in read-only mode.

Cutting, Copying, Pasting, and Deleting Text

The HyperLynx File Editor supports cutting, copying, pasting, and deleting of text, from the menus, from the toolbar, or using the standard Windows accelerator keys.

To cut text:

1. Highlight the text you want to cut.

2. Choose Cut from the Edit menu, or click the Cut button on the toolbar, or type Ctrl-X.
   The text disappears, but is saved on the Clipboard.

To copy text:

1. Highlight the text you want to copy.
2. Choose Copy from the Edit menu, or click the Copy button on the toolbar, or type Ctrl-C.

   The text is copied to the Clipboard.

To paste text:

1. Position the cursor where you want the text to be pasted.
2. Choose Paste from the Edit menu, or click the Paste button on the toolbar, or type Ctrl-V.

   The pasted text is inserted at the cursor location.

To delete text:

1. Highlight the text you want to delete.
2. Choose Delete from the Edit menu, or press the delete key.
   The text disappears.

You can also delete text by clicking in the text to position the cursor, then typing with the Backspace key or Delete key.

Undoing an Action

The HyperLynx File Editor provides a single-level “undo” feature.

To undo the previous editing action:

1. From the Edit menu, choose Undo.
   OR
   Type Ctrl-Z.

The last editing action you performed is undone.

Going to a Line Number

To go to a line number:

1. From the Search menu, choose Go To Line.
   OR
   Click the Go To button on the toolbar.
   A dialog box opens.

2. Type the line number you want to go to.

3. Click OK.

The editor jumps to the specified line number, with the matching line appearing at the *top* of the window.

Finding Text

Finding Normal Text

To find text:

1. From the Search menu, choose Find.
   OR
   Click the Find button on the toolbar.
   OR
   Type Ctrl-F.
   A dialog box opens.

2. Type text you want to find.

3. Click OK.

The editor jumps to the first occurrence of the specified text (or gives an error that no matching text could be found). The line with the matching text appears at the *top* of the window.

To find the next occurrence of the same text:

1. From the Search menu, choose Find Next.
   OR
   Click the Next button on the toolbar.
   OR
   Press F3.

The editor jumps to the next occurrence of the text (or gives an error that no matching text could be found). The line with the matching text appears at the *top* of the window.

Printing a File

To print the file that is open in the editor:

1. From the File menu, choose Print.
   OR
   Click the Print button on the toolbar.
   A dialog box opens.

2. Change options, if needed, in the dialog box.

3. Click OK.

Saving a File

To save the file that is open in the editor:

1. From the File menu, choose Save.
   OR
   Click the Save button on the toolbar.

To save the file under a new name:

1. From the File menu, choose Save As.

2. Type the new file name, then click Save.

Closing a File

To close the file that is open in the editor:

1. From the File menu, choose Close.
   OR
   Click the Close button on the toolbar.

If you have edited the file without saving it, you are prompted to save before the file closes.

Exiting the Editor

To exit the editor:

1. From the File menu, choose Exit.
   OR
   Click the Exit button on the toolbar.

If you have edited the file without saving it, you are prompted to save before the Editor closes.

Chapter 19: Getting Technical Support and Updating IC Models

Summary

This chapter describes:

- how to contact HyperLynx for technical support
- how to send BoardSim files for technical investigation
- how to connect to the HyperLynx Web site
- how to update IC models automatically over the Internet
- a HyperLynx Web information feature called “HyperLynx Web News”

How to Contact HyperLynx for Technical Support

HyperLynx automatically offers technical support to all customers for the first 30 days after they purchase a HyperLynx product. Beyond 30 days, support is available to customers who have purchased product maintenance.

U.S. customers can contact HyperLynx in any of the following ways:

- e-mail: support@hyperlynx.com

- voice: 425-869-2320
- fax: 425-881-1008

International customers should first contact the VAR or reseller from whom they purchased their HyperLynx software. If local support is not available or is inadequate, then international customers should e-mail to support@hyperlynx.com.

All written correspondence to HyperLynx should include the user name, company name, exact version of HyperLynx software being used, and key serial number. For details on determining your version and key number, see the sections below.

Determining the Version of HyperLynx Software

When requesting technical support, knowing the exact version of HyperLynx software with which you are having a problem is often important.

To determine the version of HyperLynx software you're running:

1. With the HyperLynx software running, from the Help menu, choose About.
2. At the top of the About dialog box, note the version number ("Vx.xx") and the build number (e.g., "168").

Determining the Key Serial Number

When requesting technical support, knowing serial number of your hardware key is important. (Even if you are using a floating license, the hardware key on your server PC has a serial number.)

To determine the serial number of your key:

1. With the HyperLynx software running, from the Options menu, choose Licensing.
2. *If you are running node-locked* (i.e., with the key attached to your PC), the number is displayed in the Key Serial Number field toward the bottom of

the dialog box.

If you are running with a floating license (meaning that the key is attached to a remote server PC), then in the Licensing dialog box, in the HyperLynx License Servers area, click the Test button next to a server name. A dialog box opens; the server's key serial number is listed at the top of the box.

Sending Board Files to HyperLynx

If you experience problems with a specific board design, it is often critical to send the .HYP and associated files to HyperLynx for investigation. **Because .HYP files can be fairly large, you should always zip them before sending.** Not all of a design's information is included in its .HYP file. Session edits you make, for example, are stored in the associated .BUD file (see Chapter 14 for details). And you may be using IC models that are not included in HyperLynx's standard product.

To send a complete board design to HyperLynx:

1. Create a ZIP file containing each of the following files:
   — the .HYP board file
   — the .BUD session file (in same directory as .HYP)
   — the .PJH "project" file (in same directory as .HYP)
   — the .REF AutoMapping file (in same directory as .HYP)
   — the file BSW.INI (in the main BoardSim directory)
   — any IC model files which are not shipped by HyperLynx

2. E-mail the ZIP file to support@hyperlynx.com

If the problem that needs investigation is related to the PCB translator that creates the .HYP file, you will also need to include the "native" file(s) from your PCB-layout tool.

HyperLynx World Wide Web Site

HyperLynx maintains a World Wide Web site, which all customers are encouraged to browse. Of particular interest on the Web site are links to large numbers of other sites offering IBIS models.

To browse the HyperLynx Web site:

1. Point your browser to http://www.hyperlynx.com/

To browse the Web site from inside BoardSim:

1. From inside BoardSim, from the Help menu, choose HyperLynx on the Web; then choose HyperLynx Web Site.

BoardSim attempts to automatically launch your Web browser and connect directly to HyperLynx's home page.

Links to Other Sites Offering IBIS Models

Many IC manufacturers now make IBIS IC models available directly from their World Wide Web or FTP sites. While HyperLynx attempts to ship with BoardSim as many of these models as possible, some manufacturers do not allow their models to be shipped with third-party products. Also, new models are appearing constantly.

In order to make searching for new IBIS models on manufacturer Web and FTP sites as easy as possible, HyperLynx maintains a Web page with links to all known IBIS sites. This page is updated approximately once a month.

To go directly to the HyperLynx IBIS page, from inside BoardSim:

1. From inside BoardSim, from the Help menu, choose HyperLynx on the Web; then choose Links to IBIS Models.
   OR
   From the File menu, choose Go to the IBIS Web Page.

To go to the HyperLynx IBIS Web page "manually":

1. Point your Web browser to http://www.hyperlynx.com/ibis.html

Updating IC Models over the Internet

BoardSim has the ability (if you are properly connected to the Internet) to update your IC models automatically over the Internet from HyperLynx's Web site. For complete details, see Chapter 8, section "Updating Models over the Internet."

HyperLynx Web News

As part of its Web site, HyperLynx offers a feature called "HyperLynx Web News" (or "HyperNews"). Web News gives you a way to receive messages from HyperLynx regarding product updates and patches, technical information about high-speed design, and so forth. You can access the feature from inside BoardSim, in conjunction with your Web browser.

To access HyperLynx Web News:

1. From inside BoardSim, from the File menu, choose HyperNews.
   OR
   From the Help menu, choose HyperLynx on the Web, then HyperNews.

BoardSim automatically invokes your Web browser and points it to a location at HyperLynx's site. In addition to reading the information on the News page, you can also move to other topics on the HyperLynx Web site.

Appendix A: IBIS V2.1 Specification

Summary

This appendix contains the specification for the IBIS signal-integrity modeling format, version 2.1. One section of the specification has been removed, for size and readability reasons: the rarely used [Package Model] and [Define Package Model] keywords. To view the complete specification including these keywords, open the Visual IBIS Editor and access its Help system. (See Chapter 10, section "The Visual IBIS Editor" for details.)

See Chapter 8, section "IC-Model Formats" for details on BoardSim's IC-modeling formats.

Detailed Specification

```
|=============================================================================
| I/O Buffer Information Specification (IBIS) Version 2.1 (December 13, 1995)
|
| IBIS is a standard for electronic behavioral specifications of integrated
| circuit input/output analog characteristics.
|=============================================================================
| Statement of Intent:
|
| In order to enable an industry standard method to electronically transport
| IBIS Modeling Data between silicon vendors, simulation software vendors, and
| end customers, this template is proposed.  The intention of this template is
| to specify a consistent format that can be parsed by software, allowing
| simulation vendors to derive models compatible with their own products.
|
| One goal of this template is to represent the current state of IBIS data,
| while allowing a growth path to more complex models / methods (when deemed
| appropriate).  This would be accomplished by a revision of the base
| template, and possibly the addition of new keywords or categories.
|
| Another goal of this template is to ensure that it is simple enough for
| silicon vendors and customers to use and modify, while ensuring that it is
| rigid enough for software simulation vendors to write reliable parsers.
```

```
|
| Finally, this template is meant to contain a complete description of the I/O
| elements on an entire component.  Consequently, several models will need to
| be defined in each file, as well as a table that equates the appropriate
| buffer to the correct pin and signal name.
|
| Version 2.0 of this electronic template was finalized by an industry-wide
| group of simulation experts representing various companies and interests.
| "IBIS Open Forum" meetings were held biweekly to accomplish this task.
|
| Commitment to Backward Compatibility.  Version 1.0 is the first valid IBIS
| ASCII file format.  It represents the minimum amount of I/O buffer
| information required to create an accurate IBIS model of common CMOS and
| bipolar I/O structures.  Future revisions of the ASCII file will add items
| considered to be "enhancements" to Version 1.0 to allow accurate modeling
| of new, or other, I/O buffer structures.  Consequently, all future
| revisions will be considered supersets of Version 1.0, allowing backward
| compatibility.  In addition, as modeling platforms develop support for
| revisions of the IBIS ASCII template, all previous revisions of the
| template must also be supported.
|
| Version 1.1 update.  The file "ver1_1.ibs" is conceptually the same as
| the 1.0 version of the IBIS ASCII format (ver1_0.ibs).  However, various
| comments have been added for further clarification.
|
| Version 2.0 update.  The file "ver2_0.ibs" maintains backward compatibility
| with Versions 1.0 and 1.1.  All new keywords and elements added in Version
| 2.0 are optional.  A complete list of changes to the specification is in the
| IBIS Version 2.0 Release Notes document ("ver2_0.rn").
|
| Version 2.1 update.  The file "ver2_1.ibs" contains clarification text
| changes, corrections, and two additional waveform parameters beyond
| Version 2.0.
|
|=============================================================================
|
| General syntax rules and guidelines for ASCII IBIS files:
|
| 1)  The content of the files is case sensitive, except for reserved
|     words and keywords.  File names must be all lower case.
|
| 2)  The following words are reserved words and must not be used for
|     any other purposes in the document:
|        POWER - reserved model name, used with power supply pins,
|        GND   - reserved model name, used with ground pins,
|        NC    - reserved model name, used with no-connect pins,
|        NA    - used where data not available.
|
| 3)  File names used in the file must only have lower case characters to
|     enhance UNIX compatibility and must conform to DOS rules.  (The length of
|     a file name should not exceed eight plus three characters and it must
|     not contain special characters that are illegal in DOS).
|
| 4)  The file must have no more than 80 characters per line.
|
```

```
| 5)  Anything following the comment character is ignored and considered a
|     comment on that line.  The default "|" (pipe) character can be changed
|     by the keyword [Comment Char] to any other character. The [Comment Char]
|     keyword can be used throughout the file as desired.
|
| 6)  Keywords must be enclosed in square brackets, [], and must start in
|     column 1 of the line.
|
| 7)  Underscores and spaces are equivalent in keywords.  Spaces are not
|     allowed in subparameter names.
|
| 8)  Valid scaling factors are:
|        T = tera        k = kilo        n = nano
|        G = giga        m = milli       p = pico
|        M = mega        u = micro       f = femto
|     When no scaling factors are specified, the appropriate base units are
|     assumed.  (These are volts, amperes, ohms, farads, henries, and seconds.)
|     The parser looks at only one alphabetic character after a numerical
|     entry, therefore it is enough to use only the prefixes to scale the
|     parameters.  However, for clarity, it is allowed to use full
|     abbreviations for the units, (e.g., pF, nH, mA, mOhm).  In addition,
|     scientific notation IS allowed (e.g., 1.2345e-12).
|
| 9)  The V/I data tables should use enough data points around sharply curved
|     areas of the V/I curves to describe the curvature accurately.  In linear
|     regions there is no need to define unnecessary data points.
|
| 10) The usage of TAB characters is legal, but they should be avoided as far
|     as possible.  This is to eliminate possible complications which might
|     arise in situations when TAB characters are automatically converted to
|     multiple spaces by text editing, file transfering and similar software.
|     In cases like that, lines might become longer than 80 characters, which
|     is illegal in IBIS files.
|
| 11) Currents are considered positive when their direction is into the
|     component.
|
| 12) All temperatures are represented in degrees Celsius.
|
| 13) Important supplemental information is contained in the last section,
|     "NOTES ON DATA DERIVATION METHOD", concerning how data values are
|     derived.
|
|==============================================================================
|     Keyword:  [IBIS Ver]
|    Required:  Yes
| Description:  Specifies the IBIS template version.  This keyword informs
|               electronic parsers of the kinds of data types that are
|               present in the file.
| Usage Rules:  [IBIS Ver] must be the first keyword in any IBIS file.  It is
|               normally on the first line of the file, but can be preceded
|               by comment lines that must begin with a "|".
|------------------------------------------------------------------------------
[IBIS Ver]      2.1                    | Used for template variations
|
```

```
|================================================================================
|     Keyword:  [Comment Char]
|    Required:  No
| Description:  Defines a new comment character to replace the default
|               "|" (pipe) character, if desired.
| Usage Rules:  The new comment character to be defined must be followed by
|               the underscore character and the letters "char".  For example:
|               "|_char" redundantly redefines the comment character to be
|               the pipe character.  The new comment character is in effect
|               only following the [Comment Char] keyword.  The following
|               characters MAY NOT be used:  A B C D E F G H I J K L M N O P
|               Q R S T U V W X Y Z a b c d e f g h i j k l m n o p q r s t u
|               v w x y z 0 1 2 3 4 5 6 7 8 9 [ ] . _ / = + -
| Other Notes:  The [Comment Char] keyword can be used throughout the file, as
|               desired.
|--------------------------------------------------------------------------------
[Comment Char]  |_char
|
|================================================================================
|     Keyword:  [File Name]
|    Required:  Yes
| Description:  Specifies the name of the IBIS file, "filename.ibs".
| Usage Rules:  The file name must comply with normal DOS rules (8 char. max.
|               and no characters that are illegal in DOS).  In addition, it
|               must be all lower case, and use the extension ".ibs".
|--------------------------------------------------------------------------------
[File Name]     ver2_1.ibs
|
|================================================================================
|     Keyword:  [File Rev]
|    Required:  Yes
| Description:  Tracks the revision level of a particular .ibs file.
| Usage Rules:  Revision level is set at the discretion of the engineer
|               defining the file.  The following guidelines are recommended:
|                  0.x     silicon and file in development
|                  1.x     pre-silicon file data from silicon model only
|                  2.x     file correlated to actual silicon measurements
|                  3.x     mature product, no more changes likely
|--------------------------------------------------------------------------------
[File Rev]      1.0                     | Used for .ibs file variations
|
|================================================================================
| Keywords:     [Date] [Source] [Notes] [Disclaimer] [Copyright]
| Required:     No
| Description:  Optionally clarifies the file.
| Usage Rules:  The keyword arguments can contain blanks, and be of
|               any format.  The [Date] keyword argument is limited to a
|               maximum of 40 characters, and the month should be spelled
|               out for clarity.
|
|               Because IBIS model writers may consider the information in
|               these keywords essential to users, and sometimes legally
|               required, design automation tools should make this information
|               available.  Derivative models should include this text
|               verbatim.  Any text following the [Copyright] keyword must be
```

```
|               included in any derivative models verbatim.
|-------------------------------------------------------------------------------
[Date]          December 13, 1995          | The latest file revision date
|
[Source]        Put originator and the source of information here.  For
                example:
                From silicon level SPICE model at Intel.
                From lab measurement at IEI.
                Compiled from manufacturer's data book at Quad Design, etc.
|
[Notes]         Use this section for any special notes related to the file.
|
[Disclaimer]    This information is for modeling purposes only, and is not
                guaranteed.                     | May vary by component
|
[Copyright]     Copyright 1995, XYZ Corp., All Rights Reserved
|
|===============================================================================
|     Keyword:  [Component]
|    Required:  Yes
| Description:  Marks the beginning of the IBIS description of the integrated
|               circuit named after the keyword.
| Usage Rules:  If the .ibs file contains data for more than one component,
|               each section must begin with a new [Component] keyword.  The
|               length of the Component Name must not exceed 40 characters,
|               and blank characters are allowed.
|
|               NOTE: Blank characters are not recommended due to usability
|               issues.
|-------------------------------------------------------------------------------
[Component]     7403398 MC452
|
|===============================================================================
|     Keyword:  [Manufacturer]
|    Required:  Yes
| Description:  Clarifies the component's manufacturer.
| Usage Rules:  The length of the Manufacturer's Name must not exceed 40
|               characters (blank characters are allowed, e.g., Texas
|               Instruments).  In addition, each manufacturer must use a
|               consistent name in all .ibs files.
|-------------------------------------------------------------------------------
[Manufacturer]  Intel Corp.
|
|===============================================================================
|     Keyword:  [Package]
|    Required:  Yes
| Description:  Defines a range of values for the default packaging resistance,
|               inductance, and capacitance of the component pins.
| Sub-Params:   R_pkg, L_pkg, C_pkg
| Usage Rules:  The typical (typ) column must be specified.  If data for the
|               other columns are not available, they must be noted with "NA".
| Other Notes:  If RLC parameters are available for individual pins, they can
|               be listed in columns 4-6 under keyword [Pin].  The values
|               listed in the [Pin] description section override the default
|               values defined here.  Use the [Package Model] keyword for more
```

```
|              complex package descriptions.  If defined, the [Package Model]
|              data overrides the values in the [Package] keyword.
|              Regardless, the data listed under the [Package] keyword must
|              still contain valid data.
|-----------------------------------------------------------------------------
[Package]
| variable      typ             min             max
R_pkg           250.0m          225.0m          275.0m
L_pkg           15.0nH          12.0nH          18.0nH
C_pkg           18.0pF          15.0pF          20.0pF
|
|=============================================================================
|     Keyword:  [Pin]
|    Required:  Yes
| Description:  Associates the component's I/O models to its various external
|               pins and signal names.
|  Sub-Params:  signal_name, model_name, R_pin, L_pin, C_pin
| Usage Rules:  All pins on a component must be specified.  The first column
|               must contain the pin name.  The second column, signal_name,
|               gives the data book name for the signal on that pin.  The
|               third column, model_name, associates the I/O model for that
|               pin.  Each model_name must have a [Model] keyword below,
|               unless it is a reserved model name (POWER, GND, or NC).
|
|               Each line must contain either three or six columns.  A pin
|               line with three columns only associates the pin's signal and
|               model.  Six columns can be used to override the default
|               package values (specified under [Package]) FOR THAT PIN ONLY.
|               When using six columns, the headers R_pin, L_pin, and C_pin
|               must be listed.  If "NA" is in columns 4 through 6, the
|               default packaging values must be used.
|
|               Column length limits are:
|                  [Pin]         5 characters max
|                  model_name   20 characters max
|                  signal_name  20 characters max
|                  R_pin         9 characters max
|                  L_pin         9 characters max
|                  C_pin         9 characters max
|-----------------------------------------------------------------------------
[Pin]   signal_name     model_name      R_pin   L_pin   C_pin
|
  1     RAS0#           Buffer1         200.0m  5.0nH   2.0pF
  2     RAS1#           Buffer2         209.0m  NA      2.5pF
  3     EN1#            Input1          NA      6.3nH   NA
  4     A0              3-state
  5     D0              I/O1
  6     RD#             Input2          310.0m  3.0nH   2.0pF
  7     WR#             Input2
  8     A1              I/O2
  9     D1              I/O2
 10     GND             GND             297.0m  6.7nH   3.4pF
 11     RDY#            Input2
 12     GND             GND             270.0m  5.3nH   4.0pF
|  .
```

```
|  .
|  .
 18     Vcc3            POWER
 19     NC              NC
 20     Vcc5            POWER           226.0m  NA      1.0pF
|
|================================================================================
|    Keyword:  [Pin Mapping]
|   Required:  No
|Description:  Used to indicate which power and ground buses a given driver,
|              receiver, or terminator is connected to.
| Sub-Params:  pulldown_ref, pullup_ref, gnd_clamp_ref, power_clamp_ref
|Usage Rules:  Each power and ground bus is given a unique name which must
|              not exceed 15 characters.  The first column contains a pin
|              number.  Each pin number must match one of the pin numbers
|              declared previously in the [Pin] section of the IBIS file.
|              The second column, pulldown_ref, designates the ground bus
|              connections for that pin.  Here the term ground bus can
|              also mean another power bus.  The third column pullup_ref
|              designates the power bus connection.  The fourth and fifth
|              columns gnd_clamp_ref and power_clamp_ref contain
|              entries, if needed, to specify different ground bus
|              and power bus connections than those previously specified.
|
|              If the [Pin Mapping] keyword is present, then the bus
|              connections for EVERY pin listed in the [Pin] section must
|              be given.
|
|              Each line must contain either three or five columns.  Use the
|              NC reserved word for entries that are not needed or that follow
|              the conditions below:
|
|              All entries with identical labels are assumed to be connected.
|              Each unique entry label must connect to at least one pin whose
|              model_name is POWER or GND.
|
|              If a pin has no connection, then both the pulldown_ref
|              and pullup_ref subparameters for it will be NC.
|
|              GND and POWER pin entries and buses are designated by
|              entries in either the pulldown_ref or pullup_ref columns.
|              There is no implied association to any column other than
|              through explicit designations in other pins.
|
|              For any other type of pin, the pulldown_ref column contains
|              the power connection for the [Pulldown] table for non-ECL type
|              [Models].  This is also the power connection for the [GND Clamp]
|              table and the [Rgnd] model unless overridden by a specification
|              in the gnd_clamp_ref column.
|
|              Also, the pullup_ref column contains the power connection
|              for the [Pullup] table and, for ECL type models, the [Pulldown]
|              table.  This is also the power connection for the [POWER Clamp]
|              table and the [Rpower] model unless overridden by a
|              specification in the power_clamp_ref column.
```

```
|
|              The column length limits are:
|                      [Pin Mapping]     5 characters max
|                      pulldown_ref     15 characters max
|                      pullup_ref       15 characters max
|                      gnd_clamp_ref    15 characters max
|                      power_clamp_ref  15 characters max
|
|              When 5 columns are specified, the headings gnd_clamp_ref and
|              power_clamp_ref must be used.  Otherwise, these headings can
|              be omitted.
|-------------------------------------------------------------------------
[Pin Mapping]  pulldown_ref     pullup_ref   gnd_clamp_ref  power_clamp_ref
|
1              GNDBUS1          PWRBUS1   | Signal pins and their associated
2              GNDBUS2          PWRBUS2   | ground and power connections
3              GNDBUS1          PWRBUS1      GNDCLMP        PWRCLAMP
4              GNDBUS2          PWRBUS2      GNDCLMP        PWRCLAMP
5              GNDBUS2          PWRBUS2      NC             PWRCLAMP
6              GNDBUS2          PWRBUS2      GNDCLMP        NC
                                             | Some possible clamping connections
|  .                                         | are shown above for illustration
|  .                                         | purposes
|  .
11             GNDBUS1          NC        | One set of ground connections.
12             GNDBUS1          NC        | NC indicates no connection to
13             GNDBUS1          NC        | power bus.
|  .
21             GNDBUS2          NC        | Second set of ground connections
22             GNDBUS2          NC
23             GNDBUS2          NC
|  .
31             NC               PWRBUS1   | One set of power connections.
32             NC               PWRBUS1   | NC indicates no connection to
33             NC               PWRBUS1   | ground bus.
|  .
41             NC               PWRBUS2   | Second set of power connections
42             NC               PWRBUS2
43             NC               PWRBUS2
|  .
51             GNDCLMP          NC        | Additional power connections
52             NC               PWRCLMP   | for clamps
|
|=========================================================================
|    Keyword:  [Diff Pin]
|   Required:  No
|Description:  Associates differential pins, their differential
|              threshold voltages, and differential timing delays.
| Sub-Params:  inv_pin, vdiff, tdelay_typ, tdelay_min, tdelay_max
|Usage Rules:  Enter only differential pin pairs.  The first column,
|              [Diff Pin], contains a non-inverting pin number.  The second
|              column, inv_pin, contains the corresponding inverting pin number
|              for I/O output.  Each pin number must match the pin
|              numbers declared previously in the [Pin] section of the IBIS
|              file.  The third column, vdiff, contains the specified
```

```
|              output and differential threshold voltage between pins if
|              the pins are Input or I/O model types.  For output only
|              differential pins, the vdiff entry is 0 V.  The fourth, fifth,
|              and sixth columns, tdelay_typ, tdelay_min, and tdelay_max,
|              contain launch delays of the non-inverting pins relative to
|              the inverting pins.  The values can be of either polarity.
|
|              If a pin is a differential input pin, the differential input
|              threshold (vdiff) overrides and supersedes the need for Vinh and
|              Vinl.
|
|              If vdiff is not defined for a pin that is defined as requiring a
|              Vinh by its [Model] type, vdiff is set to the default value of
|              200 mV.
|
|Other Notes:  The output pin polarity specification in the table overrides
|              the [Model] Polarity specification such that the pin in the
|              [Diff Pin] column is Non-Inverting and the pin in the
|              inv_pin column is Inverting.  This convention enables
|              one [Model] to be used for both pins.
|
|              Column length limits are:
|                 [Diff Pin]     5 characters max
|                 inv_pin        5 characters max
|                 vdiff          9 characters max
|                 tdelay_typ     9 characters max
|                 tdelay_min     9 characters max
|                 tdelay_max     9 characters max
|
|              Each line must contain either four or six columns.  If "NA" is
|              entered in the vdiff, tdelay_typ, or tdelay_min columns, its
|              entry is interpreted as 0 V or 0 ns.  If "NA" appears in
|              the tdelay_max column, its value is interpreted as the
|              tdelay_typ value.  When using six columns, the headers
|              tdelay_min and tdelay_max must be listed.  Entries for the
|              tdelay_min column are based on minimum magnitudes; and
|              tdelay_max column, maximum magnitudes.  One entry of vdiff,
|              regardless of its polarity, is used for difference magnitudes.
|-------------------------------------------------------------------------
[Diff Pin]  inv_pin  vdiff  tdelay_typ tdelay_min tdelay_max
|
 3           4       150mV    -1ns       0ns      -2ns  | Input or I/O pair
 7           8         0V      1ns        NA        NA  | Output* pin pair
 9          10         NA       NA        NA        NA  | Output* pin pair
16          15       200mV     1ns    | Input or I/O pin pair
20          19         0V       NA    | Output* pin pair, tdelay = 0
22          21         NA       NA    | Output*, tdelay = 0
                                      | * Could be Input or I/O with vdiff = 0
|
|=========================================================================
|     Keyword:  [Model]
|    Required:  Yes
| Description:  Used to define a model, and its attributes.
|  Sub-Params:  Model_type, Polarity, Enable, Vinl, Vinh, C_comp, Vmeas, Cref,
|               Rref, Vref
```

```
| Usage Rules:  Each model type must begin with the keyword [Model].
|               The model name must match the one that is listed under
|               the [Pin] keyword and must not contain more than 20 characters.
|               A .ibs file must contain enough [Model] keywords to cover all
|               of the model names specified under the [Pin] keyword, except
|               for those model names that use reserved words (POWER, GND and
|               NC).  Model names with reserved words are an exception and they
|               do not have to have a corresponding [Model] keyword.
|
|               Model_type must be one of the following:
|               Input, Output, I/O, 3-state, Open_drain, I/O_open_drain,
|               Open_sink, I/O_open_sink, Open_source, I/O_open_source,
|               Input_ECL, Output_ECL, I/O_ECL, and Terminator.
|
|               Special usage rules apply to the following.  Some definitions
|               are included for clarification:
|
|               Input              These model types must have Vinl and Vinh
|               I/O                defined.  If they are not defined, the
|               I/O_open_drain     parser issues a warning and the default
|               I/O_open_sink      values of Vinl = 0.8 V and Vinh = 2.0 V are
|               I/O_open_source    assumed.
|
|               Input_ECL          These model types must have Vinl and Vinh
|               I/O_ECL            defined.  If they are not defined, the
|                                  parser issues a warning and the default
|                                  values of Vinl = -1.475 V and Vinh =
|                                  -1.165 V are assumed.
|
|               Terminator         This model type is an input-only device that
|                                  can have analog loading effects on the
|                                  circuit being simulated but has no digital
|                                  logic thresholds.  Examples of Terminators
|                                  are: capacitors, termination diodes, and
|                                  pull-up resistors.
|
|               Output             This model type indicates that an output
|                                  always sources and/or sinks current and
|                                  cannot be disabled.
|
|               3-state            This model type indicates that an output
|                                  can be disabled, i.e. put into a high
|                                  impedance state.
|
|               Open_sink          These model types indicate that the output
|               Open_drain         has an OPEN side (do not use the [Pullup]
|                                  keyword, or if it must be used, set I = 0 mA
|                                  for all voltages specified) and the output
|                                  SINKS current.  Open_drain model type is
|                                  retained for backward compatibility.
|
|               Open_source        This model type indicates that the
|                                  output has an OPEN side (do not use the
|                                  [Pulldown] keyword, or if it must be used,
|                                  set I = 0 mA for all voltages specified) and
```

```
|                                 the output SOURCES current.
|
|                Input_ECL        These model types specify that the model
|                Output_ECL       represents an ECL type logic that follows
|                I/O_ECL          different conventions for the [Pulldown]
|                                 keyword.
|
|                The Model_type and C_comp subparameters are required.  The
|                Polarity, Enable, Vinl, Vinh, Vmeas, Cref, Rref, and Vref sub-
|                parameters are optional.  C_comp defines the silicon die
|                capacitance.  This value should not include the capacitance of
|                the package.  C_comp is allowed to use "NA" for the min and max
|                values only.  The Polarity subparameter can be defined as
|                either Non-Inverting or Inverting, and the Enable sub-
|                parameter can be defined as either Active-High or Active-Low.
|
|                The Cref and Rref subparameters correspond to the test load
|                that the manufacturer uses when specifying the propagation
|                delay and/or output switching time of the device.  The Vmeas
|                subparameter is the reference voltage level that the
|                manufacturer uses for the component.  Include Cref, Rref, and
|                Vmeas information to facilitate board-level timing simulation.
|                The assumed connections for Cref, Rref, and Vref are shown in
|                the following diagram:
|
|                                  _________
|                                 |         |
|                                 |      |\ |            Rref
|                                 |Driver| \|------o----/\/\/\----o Vref
|                                 |      | /|      |
|                                 |      |/ |     === Cref
|                                 |_________|      |
|                                                  |
|                                                 GND
|
| Other Notes:  A complete [Model] description normally contains the following
|                keywords:  [Voltage Range], [Pullup], [Pulldown], [GND Clamp],
|                [POWER Clamp], and [Ramp].  A Terminator model uses one
|                or more of the [Rgnd], [Rpower], [Rac], and [Cac].  However,
|                some models may have only a subset of these keywords.  For
|                example, an input structure normally only needs the
|                [Voltage Range], [GND Clamp], and possibly the [POWER Clamp]
|                keywords.  If one or more of [Rgnd], [Rpower], [Rac], and [Cac]
|                keywords are used, then the Model_type must be Terminator.
|-------------------------------------------------------------------------------
| Signals        CLK1, CLK2,...          | Optional signal list, if desired
[Model]          Clockbuffer
Model_type       I/O
Polarity         Non-Inverting
Enable           Active-High
Vinl = 0.8V                               | input logic "low" DC voltage, if any
Vinh = 2.0V                               | input logic "high" DC voltage, if any
Vmeas = 1.5V                 |Reference voltage for timing measurements
Cref = 50pF                  |Timing specification test load capacitance value
Rref = 500                   |Timing specification test load resistance value
```

```
Vref = 0                    |Timing specification test load voltage
| variable      typ             min             max
C_comp          12.0pF          10.0pF          15.0pF
|
|================================================================================
|     Keyword:  [Temperature Range]
|    Required:  Yes, if other than the preferred 0, 50, 100 degree Celsius
|               range
| Description:  Defines the temperature range over which the model is
|               to operate.
| Usage Rules:  List the actual die temperatures (not percentages) in the
|               typ, min, max format.  "NA" is allowed for min and max only.
| Other Notes:  The [Temperature Range] keyword also describes the temperature
|               range over which the various V/I curves and ramp rates were
|               derived.
|--------------------------------------------------------------------------------
| variable              typ             min             max
[Temperature Range]     27.0            -50             130.0
|
|================================================================================
|    Keyword:  [Voltage Range]
|   Required:  Yes, if [Pullup Reference], [Pulldown Reference],
|              [POWER Clamp Reference], and [GND Clamp Reference] are not
|              present
|Description:  Defines the power supply voltage tolerance over which the
|              model is intended to operate.  It also specifies the default
|              voltage rail to which the pull-up and [POWER Clamp] V/I data is
|              referenced.
|Usage Rules:  Provide actual voltages (not percentages) in the typ, min,
|              max format.  "NA" is allowed for the min and max values only.
|Other Notes:  If the [Voltage Range] keyword is not present, then all four
|              of these keywords described below must be present: [Pullup
|              Reference], [Pulldown Reference], [POWER Clamp Reference],
|              and [GND Clamp Reference].  If the [Voltage Range] is present,
|              the other keywords are optional and may or may not be used as
|              required.  It is legal (although redundant) for an optional
|              keyword to specify the same voltage as specified by the
|              [Voltage Range] keyword.
|-------------------------------------------------------------------------
| variable              typ             min             max
[Voltage Range]         5.0V            4.5V            5.5V
|
|================================================================================
|    Keyword:  [Pullup Reference]
|   Required:  Yes, if the [Voltage Range] keyword is not present.
|Description:  Defines a voltage rail other than that defined by the
|              [Voltage Range] keyword as the reference voltage for the
|              pull-up V/I data.
|Usage Rules:  Provide actual voltages (not percentages) in the typ, min,
|              max format.  "NA" is allowed for the min and max values only.
|Other Notes:  This keyword, if present, also defines the voltage range over
|              which the min and max dV/dt_r values are derived.
|-------------------------------------------------------------------------
| variable              typ             min             max
[Pullup Reference]      5.0V            4.5V            5.5V
```

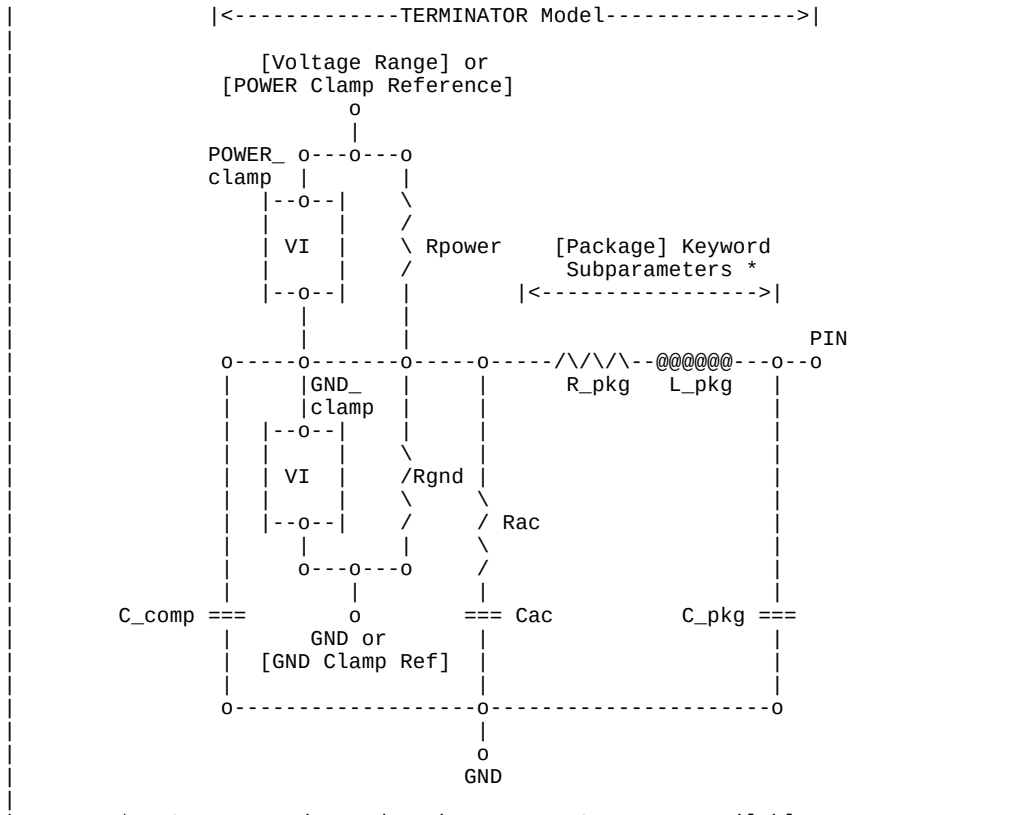
```
|
|================================================================================
|    Keyword:  [Pulldown Reference]
|   Required:  Yes, if the [Voltage Range] keyword is not present.
|Description:  Defines a power supply rail other than 0 V as the reference
|              voltage for the pull-down V/I data.  If this keyword is not
|              present, the voltage data points in the pull-down V/I table
|              are referenced to 0 V.
|Usage Rules:  Provide actual voltages (not percentages) in the typ, min,
|              max format.  "NA" is allowed for the min and max values only.
|Other Notes:  This keyword, if present, also defines the voltage range over
|              which the typ, min, and max dV/dt_f values are derived.
|--------------------------------------------------------------------------
| variable              typ             min             max
[Pulldown Reference]    0V              0V              0V
|
|================================================================================
|    Keyword:  [POWER Clamp Reference]
|   Required:  Yes, if the [Voltage Range] keyword is not present.
|Description:  Defines a voltage rail other than that defined by the
|              [Voltage Range] keyword as the reference voltage for the
|              [POWER Clamp] V/I data.
|Usage Rules:  Provide actual voltages (not percentages) in the typ, min,
|              max format.  "NA" is allowed for the min and max values only.
|Other Notes:  Refer the "Other Notes" section of the [GND Clamp Reference]
|              keyword.
|--------------------------------------------------------------------------
| variable              typ             min             max
[POWER Clamp Reference] 5.0V            4.5V            5.5V
|
|================================================================================
|    Keyword:  [GND Clamp Reference]
|   Required:  Yes, if the [Voltage Range] keyword is not present.
|Description:  Defines a power supply rail other than 0 V as the reference
|              voltage for the [GND Clamp] V/I data.  If this keyword is not
|              present, the voltage data points in the [GND Clamp] V/I table
|              are referenced to 0 V.
|Usage Rules:  Provide actual voltages (not percentages) in the typ, min,
|              max format.  "NA" is allowed for the min and max values only.
|Other Notes:  Power Supplies: It is intended that standard TTL and CMOS
|              devices be specified using only the [Voltage Range] keyword.
|              However, in cases where the output characteristics of a device
|              depend on more than a single supply and ground, or a pull-up,
|              pull-down, or clamp structure is referenced to something other
|              than the default supplies, use the additional 'reference'
|              keywords.
|--------------------------------------------------------------------------
| variable              typ             min             max
[GND Clamp Reference]   0V              0V              0V
|
|================================================================================
|    Keywords:  [Pulldown], [Pullup], [GND Clamp], [POWER Clamp]
|    Required:  Yes, if they exist in the device
| Description:  The data points under these keywords define the V/I curves of
|               the pull-down and pull-up structures of an output buffer and
```

```
|               the V/I curves of the clamping diodes connected to the GND and
|               the POWER pins, respectively.  Currents are considered positive
|               when their direction is into the component.
|
| Usage Rules:  In each of these sections, the first column contains the
|               voltage value, and the three remaining columns hold the
|               typical, minimum, and maximum current values.  The four
|               entries, Voltage, I(typ), I(min), and I(max) must be placed on
|               a single line and must be separated by at least one white
|               space or tab character.
|
|               All four columns are required under these keywords.  However,
|               data is only required in the typical column.  If minimum
|               and/or maximum current values are not available, the reserved
|               word "NA" must be used.  "NA" can be used for currents in the
|               typical column, but numeric values MUST be specified for the
|               first and last voltage points on any V/I curve.  Each V/I
|               curve must have at least 2, but not more than 100, voltage
|               points.
|
| Other Notes:  The V/I curve of the [Pullup] and the [POWER Clamp] structures
|               are 'Vcc relative', meaning that the voltage values are
|               referenced to the Vcc pin.  (Note: Under these keywords, all
|               references to 'Vcc' refer to the voltage rail defined by the
|               [Voltage range], [Pullup Reference], or [POWER Clamp Reference]
|               keywords, as appropriate.)  The voltages in the data tables are
|               derived from the equation:  Vtable = Vcc - Voutput.
|
|               Therefore, for a 5 V component, -5 V in the table actually
|               means 5 V above Vcc, which is +10 V with respect to ground;
|               and 10 V means 10 V below Vcc, which is -5 V with respect to
|               ground.  Vcc-relative data is necessary to model a pull-up
|               structure properly, since the output current of a pull-up
|               structure depends on the voltage between the output and Vcc
|               pins and not the voltage between the output and ground pins.
|               Note that the [GND Clamp] V/I curve can include quiescent
|               input currents, or the currents of a 3-stated output, if so
|               desired.
|
|               When tabulating data for ECL devices, the data in the pull-down
|               table is measured with the output in the 'logic low' state.
|               In other words, the data in the table represents the V/I
|               characteristics of the output when the output is at the most
|               negative of its two logic levels.  Likewise, the data in the
|               pull-up table is measured with the output in the 'logic one'
|               state and represents the V/I characteristics when the output
|               is at the most positive logic level.  Note that in BOTH of
|               these cases, the data is referenced to the Vcc supply voltage,
|               using the equation  Vtable = Vcc - Voutput.
|
|               Monotonicity Requirements:
|               To be monotonic, the V/I table data must meet any one of the
|               following 8 criteria:
|                 1- The CURRENT axis either increases or remains constant as
|                    the voltage axis is increased.
```

```
|                2- The CURRENT axis either increases or remains constant as
|                   the voltage axis is decreased.
|                3- The CURRENT axis either decreases or remains constant as
|                   the voltage axis is increased.
|                4- The CURRENT axis either decreases or remains constant as
|                   the voltage axis is decreased.
|
|                5- The VOLTAGE axis either increases or remains constant as
|                   the current axis is increased.
|                6- The VOLTAGE axis either increases or remains constant as
|                   the current axis is decreased.
|                7- The VOLTAGE axis either decreases or remains constant as
|                   the current axis is increased.
|                8- The VOLTAGE axis either decreases or remains constant as
|                   the current axis is decreased.
|
|              An IBIS syntax checking program shall test for non-monotonic
|              data and provide a maximum of one note per V/I table if
|              non-montonic data is found.  For example:
|                "NOTE: Line 300, Pulldown V/I table for model DC040403 is
|                 non-monotonic!  Most simulators will filter this data
|                 to remove the non-monotonic data."
|
|              It is also recognized that the data may be monotonic if
|              currents from both the output stage and the clamp diode are
|              added together as most simulators do.  To limit the complexity
|              of the IBIS Version 2.x syntax checking programs, such
|              programs will conduct monotonicity testing only on one
|              V/I table at a time.
|
|              It is assumed that the simulator sums the clamp curves
|              together with the appropriate pull-up or pull-down curve when
|              a buffer is driving high or low, respectively.  From this
|              assumption and the nature of 3-statable buffers, it follows
|              that the data in the clamping curve sections are handled as
|              constantly present curves and the pull-up and pull-down curves
|              are used only when needed in the simulation.
|
|              The clamp curves of an input or I/O buffer can be measured
|              directly with a curve tracer, with the I/O buffer 3-stated.
|              However, sweeping enabled buffers results in curves that are
|              the sum of the clamping curves and the output structures.
|              Based on the assumption outlined above, the pull-up and
|              pull-down curves of an IBIS model must represent the
|              difference of the 3-stated and the enabled buffer's curves.
|              (Note that the resulting difference curve can demonstrate a
|              non-monotonic shape.)  This requirement enables the simulator
|              to sum the curves, without the danger of double counting, and
|              arrive at an accurate model in both the 3-stated and enabled
|              conditions.
|
|              Since in the case of a non 3-statable buffer, this difference
|              curve cannot be generated through lab measurements (because
|              the clamping curves cannot be measured alone), the pull-up and
|              pull-down curves of an IBIS model can contain the sum of the
```

```
|               clamping characteristics and the output structure.  In this
|               case, the clamping curves must contain all zeroes, or the
|               keywords must be omitted.
|------------------------------------------------------------------------------
[Pulldown]
|  Voltage   I(typ)    I(min)    I(max)
|
   -5.0V    -40.0m    -34.0m    -45.0m
   -4.0V    -39.0m    -33.0m    -43.0m
|    .
|    .
    0.0V      0.0m      0.0m      0.0m
|    .
|    .
    5.0V     40.0m     34.0m     45.0m
   10.0V     45.0m     40.0m     49.0m
|
[Pullup]                              | Note: Vtable = Vcc - Voutput
|
|  Voltage   I(typ)    I(min)    I(max)
|
   -5.0V     32.0m     30.0m     35.0m
   -4.0V     31.0m     29.0m     33.0m
|    .
|    .
    0.0V      0.0m      0.0m      0.0m
|    .
|    .
    5.0V    -32.0m    -30.0m    -35.0m
   10.0V    -38.0m    -35.0m    -40.0m
|
[GND Clamp]
|
|  Voltage   I(typ)    I(min)    I(max)
|
   -5.0V  -3900.0m  -3800.0m  -4000.0m
   -0.7V    -80.0m    -75.0m    -85.0m
   -0.6V    -22.0m    -20.0m    -25.0m
   -0.5V     -2.4m     -2.0m     -2.9m
   -0.4V      0.0m      0.0m      0.0m
    5.0V      0.0m      0.0m      0.0m
|
[POWER Clamp]                         | Note: Vtable = Vcc - Voutput
|
|  Voltage   I(typ)    I(min)    I(max)
|
   -5.0V   4450.0m       NA        NA
   -0.7V     95.0m       NA        NA
   -0.6V     23.0m       NA        NA
   -0.5V      2.4m       NA        NA
   -0.4V      0.0m       NA        NA
    0.0V      0.0m       NA        NA
|
```

```
|================================================================================
|    Keywords:  [Rgnd], [Rpower], [Rac], [Cac]
|    Required:  Yes, if they exist in the device
| Description:  The data for these keywords define the resistance values of
|               Rgnd and Rpower connected to GND and the POWER pins,
|               respectively.
| Usage Rules:  For each of these keywords, the three columns hold the
|               typical, minimum, and maximum resistance values.  The three
|               entries for R(typ), R(min), and R(max), or the three entries
|               for C(typ), C(min), and C(max) must be placed on a single line
|               and must be separated by at least one white space or tab
|               character.  All three columns are required under these
|               keywords.  However, data is only required in the typical
|               column.  If minimum and/or maximum values are not available,
|               the reserved word "NA" must be used indicating the R(typ) or
|               C(typ) value by default.
| Other Notes:  It should be noted that [Rpower] is connected to 'Vcc' and
|               [Rgnd] is connected to 'GND'.  However, [GND Clamp Reference]
|               voltages, if defined, apply to [Rgnd].  [POWER Clamp Reference]
|               voltages, if defined, apply to [Rpower].  Either or both [Rgnd]
|               and [Rpower] may be defined and may coexist with [GND Clamp]
|               and [POWER Clamp] structures.  If the terminator consists
|               of a series R and C (often referred to as either an AC or RC
|               terminator), then both [Rac] and [Cac] are required.  When
|               [Rgnd], [Rpower], or [Rac] and [Cac] are specified, the
|               Model_type must be Terminator.
|
```



variable	R(typ)	R(min)	R(max)	
[Rgnd]	330ohm	300ohm	360ohm	Parallel Terminator
[Rpower]	220ohm	200ohm	NA	
[Rac]	30ohm	NA	NA	
variable	C(typ)	C(min)	C(max)	AC terminator
[Cac]	50pF	NA	NA	

Keyword:	[Ramp]
Required:	Yes, except for inputs and terminators
Description:	Defines the rise and fall times of a buffer. The ramp rate does not include packaging but does include the effects of the C_comp parameter.
Sub-Params:	dV/dt_r, dV/dt_f, R_load

```
| Usage Rules:  The rise and fall time is defined as the time it takes the
|               output to go from 20% to 80% of its final value.
|               The ramp rate is defined as:
|
|               dV          20% to 80% voltage swing
|               -- = ---------------------------------------
|               dt   Time it takes to swing the above voltage
|
|               The ramp rate must be specified as an explicit fraction
|               and must not be reduced.  The [Ramp] values can use "NA" for
|               the min and max values only.  The R_load subparameter is
|               optional if the preferred 50 ohm load is used.  The R_load sub-
|               parameter is required if a non-standard load is used.
|-------------------------------------------------------------------------------
[Ramp]
| variable      typ             min             max
dV/dt_r         2.20/1.06n      1.92/1.28n      2.49/650p
dV/dt_f         2.46/1.21n      2.21/1.54n      2.70/770p
R_load = 300ohms
|
|===============================================================================
|     Keywords:     [Rising Waveform], [Falling Waveform]
|     Required:     No
|     Description:  Describes the shape of the rising and falling edge
|                   waveforms of a driver.
|     Sub-Params:   R_fixture, V_fixture, V_fixture_min, V_fixture_max,
|                   C_fixture, L_fixture, R_dut, L_dut, C_dut
|     Usage Rules:  Each [Rising Waveform] and [Falling Waveform] keyword
|                   introduces a table of time vs. voltage points that
|                   describe the shape of an output waveform.  These
|                   time/voltage points are taken under the conditions
|                   specified in the R/L/C/V_fixture and R/L/C_dut
|                   subparameters.  The table itself consists of
|                   one column of time points, then three columns of
|                   voltage points in the standard typ, min, and max format.
|                   The four entries must be placed on a single line and
|                   must be separated by at least one white space or tab
|                   character.  All four columns are required.  However, data
|                   is only required in the typical column.  If minimum
|                   or maximum data is not available, use the reserved word
|                   "NA".  The first value in the time column need not be '0'.
|                   Time values must increase as one parses down the table.
|                   The waveform table can contain a maximum of 100 data
|                   points.  A maximum of 100 waveform tables are allowed per
|                   model.  Note that for backwards compatibility, the existing
|                   [Ramp] keyword is still required.  The data in the waveform
|                   table is taken with the effects of the C_comp parameter
|                   included.
|
|                   A waveform table must include the entire waveform;
|                   i.e., the first entry (or entries) in a voltage column
|                   must be the DC voltage of the output before switching
|                   and the last entry (or entries) of the column must be
|                   the final DC value of the output after switching.  Each
|                   table must contain at least two entries.  Thus, numerical
```

```
|                  values are required for the first and last entries of any
|                  column containing numerical data.
|
|                  A [Model] specification can contain more than one rising
|                  edge or falling edge waveform table.  However, each new
|                  table must begin with the appropriate keyword and sub-
|                  parameter list as shown below.  If more than one rising or
|                  falling edge waveform table is present, then the data in
|                  each of the respective tables must be time correlated.
|                  In other words, the rising (falling) edge data in each
|                  of the rising (falling) edge waveform tables must be
|                  entered with respect to a common reference point on the
|                  input stimulus waveform.
|
|                  The 'fixture' subparameters specify the loading
|                  conditions under which the waveform is taken.  The R_dut,
|                  C_dut, and L_dut subparameters are analogous to the
|                  package parameters R_pkg, C_pkg, and L_pkg and are used
|                  if the waveform includes the effects of pin
|                  inductance/capacitance.  The diagram below shows the
|                  interconnection of these elements.
|
|                     PACKAGE            |   TEST FIXTURE
|      _________                         |
|      |  DUT    |   L_dut   R_dut        | L_fixture  R_fixture
|      |  die    |---@@@@@--/\/\/\--o-----|--@@@@---o---/\/\/\----- V_fixture
|      |_________|                  |     |         |
|                                   |     |         |
|                                   |     |         |
|                           C_dut ===     |        === C_fixture
|                                   |     |         |
|                                   |     |         |
|                                 GND     |        GND
|
|                  Only the R_fixture and V_fixture subparameters are
|                  required, the rest of the subparameters are optional.
|                  If a subparameter is not used, its value defaults to
|                  zero.  The subparameters must appear in the text after
|                  the keyword and before the first row of the waveform
|                  table.
|
|                  V_fixture defines the voltage for typ, min, and max
|                  supply conditions.  However, when the fixture voltage
|                  is related to the power supply voltages, then the
|                  subparameters V_fixture_min and V_fixture_max can
|                  be used to further specify the fixture voltage for
|                  min and max supply voltages.
|-------------------------------------------------------------------------------
[Rising Waveform]
R_fixture = 500
V_fixture = 5.0
V_fixture_min = 4.5          |Note, R_fixture is connected to Vcc
V_fixture_max = 5.5          |Specified, but not used in this example
C_fixture = 50p
L_fixture = 2n
```

```
C_dut = 7p
R_dut = 1m
L_dut = 1n
|Time     V(typ)     V(min)     V(max)
 0.0ns     0.3        0.5         NA
 0.5ns     0.3        0.5         NA
 1.0ns     0.6        0.7         NA
 1.5ns     0.9        0.9         NA
 2.0ns     1.5        1.3         NA
 2.5ns     2.1        1.7         NA
 3.0ns     3.0        2.7         NA
 3.5ns     3.2        3.0         NA
|
[Falling Waveform]
R_fixture = 50
V_fixture = 0
|Time     V(typ)     V(min)     V(max)
 10.0ns     3.2        3.0         NA
 10.5ns     3.0        2.7         NA
 11.0ns     2.1        1.7         NA
 11.5ns     1.5        1.3         NA
 12.0ns     0.9        0.9         NA
 12.5ns     0.6        0.7         NA
 13.0ns     0.3        0.5         NA
 13.5ns     0.3        0.5         NA
|
|=============================================================================
|     Keyword:  [End]
|    Required:  Yes
| Description:  Defines the end of the .ibs file.
|-----------------------------------------------------------------------------
[End]
|
|=============================================================================
|
|                        NOTES ON DATA DERIVATION METHOD
|
| This section explains how data values are derived.  It describes certain
| assumed parameter and table extraction conditions if they are not
| explicitly specified.  It also describes the allocation of data into
| the "typ", "min", and "max" columns under variations of voltage,
| temperature, and process.
|
| The required "typ" column for all data represents typical operating
| conditions.  For most [Model] keyword data, the "min" column describes
| slow, weak performance, and  the "max" column describes the fast, strong
| performance.  It is permissible to use slow, weak devices or models to
| derive the data for the "min" column, and to use fast, strong devices or
| models to derive the data in the "max" columns under the corresponding
| voltage and temperature derating conditions for these columns.  It is also
| permissible to use typical devices or models derated by voltage and
| temperature and optionally apply proprietary "X%" and "Y%" factors
| described later for further derating.  This methodology has the
| nice feature that the data can be derived either from vendor-proprietary
| silicon models, or typical device measurement over temperature/voltage.
```

```
|
| The voltage and temperature keywords and optionally the process models
| control the conditions which define the "typ", "min", and "max" column
| entries for all V/I table keywords [Pulldown], [Pullup], [Gnd Clamp],
| and [Power Clamp]; all [Ramp] subparameters dV/dt_r and dV/dt_f; and
| all waveform table keywords and subparameters [Rising Waveform],
| [Falling Waveform], V_fixture, V_fixture_min, and V_fixture_max.
|
| The voltage keywords which control the voltage conditions are [Voltage
| Range], [Pulldown Reference], [Pullup Reference], [Gnd Clamp Reference],
| and [Power Clamp Reference].  The entries in the "min" columns
| contain the smallest magnitude voltages, and the entries in the "max"
| columns contain the largest magnitude voltages.
|
| The optional [Temperature] keyword will contain the temperature
| which causes or amplifies the slow, weak conditions in the "min" column
| and the temperature which causes or amplifies the fast, strong
| conditions in the "max" column.  Therefore, the "min" column for
| [Temperature] will contain the lowest value for bipolar devices (TTL
| and ECL) and the highest value for CMOS devices.  Default values
| described later are assumed if temperature is not specified.
|
| The "min" and "max" columns for all remaining keywords and subparameters
| will contain the smallest and largest magnitude values.  This applies
| to the [Model] subparameter C_comp as well even if the correlation
| to the voltage, temperature, and process variations are known because
| information about such correlation is not available in all cases.
|
| C_comp is considered an independent variable.  This is because C_comp
| includes bonding pad capacitance, which does not necessarily track
| fabrication process variations.  The conservative approach to using IBIS
| data will associate large C_comp values with slow, weak models, and the
| small C_comp values with fast, strong models."
|
| The default temperatures under which all V/I tables are extracted are
| provided below.  The same defaults also are stated for the [Ramp]
| subparameters, but they also apply for the waveform keywords.
|
| The stated voltage ranges for V/I tables cover the most common, single
| supply cases.  When multiple supplies are specified, the voltages shall
| extend similarly to values which handle practical extremes in reflected
| wave simulations.
|
| For the [Ramp] subparameters, the default test load and voltages are
| provided.  However, the test load can be entered directly by the R_load
| subparameter.  The allowable test loads and voltages for the waveform
| keywords are stated by required and optional subparameters; no defaults
| are needed.  Even with waveform keywords, the [Ramp] keyword continues
| to be required so that the IBIS model remains functional in situations
| which do not support waveform processing.
|
| The following discussion lists test details and default conditions.
|
| 1) V/I curves for CMOS devices:
|      typ = typical voltage, typical temp deg C, typical process
```

```
|      min = minimum voltage, max temp deg C, typical process, minus "X%"
|      max = maximum voltage, min temp deg C, typical process, plus "X%"
|
|    V/I curves for bipolar devices:
|      typ = typical voltage, typical temp deg C, typical process
|      min = minimum voltage, min temp deg C, typical process, minus "X%"
|      max = maximum voltage, max temp deg C, typical process, plus "X%"
|
|    Nominal, min, and max temperature are specified by the manufacturer
|    of the part.  The default range is 50 deg C nom, 0 deg C min, and
|    100 deg C max temperatures.
|
|    X% should be statistically determined by the silicon vendor based
|    on numerous fab lots, test chips, process controls, etc..  The value
|    of X need not be published in the IBIS file, and may decrease over
|    time as data on the I/O buffers and silicon process increases.
|
|    Temperatures are junction temperatures.
|
| 2) Voltage Ranges:
|    Points for each curve must span the voltages listed below:
|
|       Curve                  Low Voltage            High Voltage
|       -----------           -----------            ------------
|       [Pulldown]             GND - POWER            POWER + POWER
|       [Pullup]               GND - POWER            POWER + POWER
|       [GND Clamp]            GND - POWER            GND + POWER
|       [POWER Clamp]          POWER                  POWER + POWER
|
|    As described in the [Pulldown Reference] keyword section, the V/I curve of
|    the [Pullup] and the [POWER Clamp] structures are 'Vcc relative', using
|    the equation:  Vtable = Vcc - Voutput.
|
|    For example, a device with a 5 V power supply voltage should be
|    characterized between (0 - 5) = -5 V and (5 + 5) = 10 V; and a
|    device with a 3.3 V power supply should be characterized between
|    (0 - 3.3) = -3.3 V and (3.3 + 3.3) = 6.6 V for the pull-down curve.
|
|    When tabulating output data for ECL type devices, the voltage points
|    must span the range of Vcc to Vcc - 2.2 V.  This range applies to both the
|    pull-up and pull-down tables.  Note that this range applies ONLY when
|    characterizing an ECL output.
|
|    These voltage ranges must be spanned by the IBIS data.  Data derived
|    from lab measurements may not be able to span these ranges as such and
|    so may need to be extrapolated to cover the full range.  This data must
|    not be left for the simulator to provide.
|
| 3) Ramp Rates:
|    The following steps assume that the default load resistance of 50 ohms is
|    used.  There may be devices that will not drive a load of only 50 ohms
|    into any useful level of dynamics.  In these cases, use the manufacturer's
|    suggested (nonreactive) load and add the load subparameter to the [Ramp]
|    specification.
|
```

```
|    The ramp rate does not include packaging but does include the effects of
|    the C_comp parameter; it is the intrinsic output stage rise and fall time
|    only.
|
|    The ramp rates (listed in AC characteristics below) should be derived
|    as follows:
|
|    a. If starting with the silicon model, remove all packaging.  If starting
|       with a packaged device, perform the measurements as outlined below then
|       use whatever techniques are appropriate to derive the actual, unloaded
|       rise and fall times.
|
|    b. If: The Model_type is one of the following: Output, I/O, or
|           3-state (not open or ECL types);
|           Then: Attach a 50 ohm resistor to GND to derive the rising edge
|                 ramp.  Attach a 50 ohm resistor to POWER to derive the
|                 falling edge ramp.
|
|       If: The Model_type is Output_ECL or I/O_ECL;
|           Then: Attach a 50 ohm resistor to the termination voltage
|                 (Vterm = VCC - 2 V).  Use this load to derive both the
|                 rising and falling edges.
|
|       If: The Model_type is either an Open_sink type or Open_drain type;
|           Then: Attach either a 50 ohm resistor or the vendor-suggested
|                 termination resistance to either POWER or the vendor-
|                 suggested termination voltage.  Use this load to derive both
|                 the rising and falling edges.
|
|       If: The Model_type is an Open_source type;
|           Then: Attach either a 50 ohm resistor or the vendor-suggested
|                 termination resistance to either GND or the vendor-suggested
|                 termination voltage.  Use this load to derive both the rising
|                 and falling edges.
|
|    c. Due to the resistor, output swings will not make a full transition as
|       expected.  However the pertinent data can still be collected as
|       follows:
|           1)  determine the 20% to 80% voltages of the 50 ohm swing
|           2)  measure this voltage change as "dV"
|           3)  measure the amount of time required to make this swing "dt"
|
|    d. Post the value as a ratio "dV/dt".  The simulation tool vendor
|       extrapolates this value to span the required voltage swing range in
|       the final model.
|
|    e. Typ, Min, and Max must all be posted, and are derived at the same
|       extremes as the V/I curves, which are:
|
|       Ramp rates for CMOS devices:
|          typ = typical voltage, typical temp deg C, typical process
|          min = minimum voltage, max temp deg C, typical process, minus "Y%"
|          max = maximum voltage,  min temp deg C, typical process, plus  "Y%"
|
|       Ramp rates for bipolar devices:
```

```
|          typ = typical voltage, typical temp deg C, typical process
|          min = minimum voltage, min temp deg C, typical process, minus "Y%"
|          max = maximum voltage,  max temp deg C, typical process, plus  "Y%"
|
|       where nominal, min, and max temp are specified by the manufacturer of
|       the part.  The preferred range is 50 deg C nom, 0 deg C min, and
|       100 deg C max temperatures.
|
|       Note that the derate factor, "Y%", may be different than that used
|       for the V/I curve data.  This factor is similar to the X% factor
|       described above.  As in the case of V/I curves, temperatures are
|       junction temperatures.
|
|    f. During the IV measurements, the driving waveform should have a
|       rise/fall time fast enough to avoid thermal feedback.  The specific
|       choice of sweep time is left to the modeling engineer.
|
| It is expected that this data will be created from silicon vendor
| proprietary silicon-level models, and later correlated with actual device
| measurement.
```

Appendix B: .PAK-File Specification

Summary

This appendix contains the complete and official specification for HyperLynx's package-modeling (.PAK) format. The .PAK format is used to create the files BSW.PAK and USER.PAK, which together define the networked-component packages that BoardSim understands.

See Chapter 11 for details on packages in BoardSim.

Detailed Specification

HyperLynx Resistor/Capacitor Pack (PAK) File Format, Version 1.02B
December 29, 1994

Introduction

This document describes a file format (called ".PAK") for describing the electrical connections in a resistor or capacitor network package. The .PAK format can be read by HyperLynx's BoardSim signal-integrity software, and allows users to add new definitions to BoardSim's default R/C-package library.

General Format of a .PAK File:

{PAK}

{VERSION=*number* [*comment*]}

{PACK=*name* [*comment*]

(*package information*) [*comment*]
(*package information*)
...
(*package information*)
}

{PACK=*name*
(*package information*)
(*package information*)
...
(*package information*)
}

...more packages...

{PACK=*name*
(*package information*)
(*package information*)
...
(*package information*)
}

{END}

Detailed Syntax:

General Notes:

Italicized fields are to be filled in with appropriate values.

Square brackets [ ] denote optional parameters.

All subrecords [lines beginning with '('] must be on a single line.

Curly braces { } can be separated from keywords and record ends by white space; the right brace } can be on the same line as the last subrecord or on the next line.

Parentheses ( ) can be separated from keywords and record ends by white space; must be on the same line as the subrecord.

If on the same line as other text, comments must be separated by at least one white space from the preceding text. If an entire line is a comment, it can begin in column 1, but must contain not contain the character '}'.

The maximum allowed line length is 180 characters.

Lines can be terminated by CR, LF, CR-LF, or LF-CR.

White space is defined as space, horizontal tab, vertical tab, linefeed, form feed, or carriage return.

Any characters are allowed in a name, except white-space characters.

Numeric values can be followed by an exponent of the form

exxx or Exxx
where xxx is any integer value, positive or negative.

All numeric values can be followed by alphabetic scaling factors:

M=mega (1,000,000x)
K or k =kilo (1,000x)
m=milli (0.001x)
u or U=micro (1e-6x)
n or N=nano (1e-9x)
p or P=pico (1e-12x)

Suffixes may be separated from their numeric values by white space.

Scaling suffixes may be followed by other alphanumeric characters, e.g., uH or pF; the additional characters are terminated by white space

PAK

Format:
{PAK}

- PAK identifies the file format as .PAK to BoardSim
- required record; must be the first non-blank line in file
- only one PAK record allowed per file

VERSION

Format:
{VERSION=*number* [*comment*]}

- VERSION specifies the version of the file format
- *number* specifies the .PAK-file version number
- required record; must be the second non-blank line in file
- only one VERSION record allowed per file

Example Records:

example 1:
{VERSION=1.02}

example 2:
{VERSION=1.02 new definition added by Dave S.}

PACK

Format:
{PACK=*name* [*comment*]
(STYLE=*package_style*) [*comment*]
(SHAPE=*package_shape*) [*comment*]

(TOTAL_PINS=*number*) [*comment*]
(PIN_PAIR=*pin_name*,*pin_name*[,*pin_name*]) [*comment*]
(PIN_PAIR=*pin_name*,*pin_name*[,*pin_name*]) [*comment*]
...more pin pairs...
(PIN_PAIR=*pin_name*,*pin_name*[,*pin_name*]) [*comment*]
(PIN_LOC=*pin_name,location_number*)
(PIN_LOC=*pin_name,location_number*)
...more pin locations...
(PIN_LOC=*pin_name,location_number*)
}

- PACK identifies a resistor- or capacitor-package record
- must be at least one PACK record per PAK file
- *name* is the package's name; if it exceeds 20 characters, it will be truncated
- three different subrecords must follow: STYLE, TOTAL_PINS, and PIN_PAIR
- subrecords must be in the specified order, i.e., first STYLE, then TOTAL_PINS, the PIN_PAIR

STYLE subrecord

- STYLE record defines the package's component and connection style
- *package_style* specifies the package's style; valid values are:

  R_SERIES
  R_PULLUP
  R_PULLUP_PULLDOWN
  C_SERIES
  C_PULLUP

- R_xxx specifies a resistor package; C_xxx specifies a capacitor package
- x_SERIES specifies that each element in the package has two independent pins, i.e., is independent of the other elements;
- x_PULLUP specifies that each element in the package has one independent pin and one pin in common with the other elements;
- x_PULLUP_PULLDOWN specifies that each element in the package has one independent pin and two pins in common with the other elements

SHAPE subrecord

- SHAPE record defines the package's physical shape

- *package_shape* specifies the package's shape; valid values are:
  DIP
  SIP

- DIP specifies a dual-in-line package; SIP specifies a single-in-line

TOTAL_PINS subrecord

- TOTAL_PINS record specifies the total number of pins on the package
- *number* must be a positive integer

PIN_PAIR subrecord

- PIN_PAIR record specifies the pairing of two or three pins on a package
- *pin_name* is the pin's name; can be any valid name string (typically an integer number, but can be alphabetic; if it exceeds 5 characters, it will be truncated)
- all styles except R_PULLUP_PULLDOWN allow two *pin_name* fields; R_PULLUP_PULLDOWN requires three *pin_name* fields
- for x_SERIES styles, both *pin_name* fields are for independent pins;
  for x_PULLUP styles, the first *pin_name* field is for the independent pin and the second is for the common pin;
  for the R_PULLUP_PULLDOWN style, the first *pin_name* field is for the independent pin, the second is for the common pull-up pin, and the third is for the common pull-down pin

PIN_LOC subrecord

- PIN_LOC record specifies the physical position of each pin on the package
- *pin_name* is the pin's name; must match a name specified in a PIN_PAIR record
- *location_number* is an integer number that specifies the named pin's position on the package, according to the following rules:
  if the package shape is SIP and the package has *n* pins, the pin names must be numbered from 1 to *n* with 1 defined as the top-most pin, *n* as the bottom-most pin;
  if the package shape is DIP and the package has *n* pins, the pin names must be numbered from 1 to *n* with 1 in standard DIP style: pin 1 in the upper left-hand corner, pin *n* as the pin in upper right-hand corner
- the PIN_LOC information is used for drawing the package's internal connections; if missing, the internal connections are not drawn when the package is displayed in BoardSim's user interface
- the PIN_LOC records must come AFTER the PIN_PAIR records; if a PIN_LOC record for a pin comes before the PIN_PAIR record for that pin, the .PAK file is considered to have a syntax error

- each STYLE, TOTAL_PINS, or PIN_PAIR record must be on a single line
- every pin on a package must be mentioned at least once in a PIN_PAIR *pin_name* field; independent pins can be mentioned only once; common pins can be mentioned multiple times

Example Records:

example 1:

```
{PACK=R_16_SER_SIP          16-pin, series-style DIP resistor
(STYLE=R_SERIES)            package
(SHAPE=DIP)
(TOTAL_PINS=16)
(PIN_PAIR=1,16)
(PIN_PAIR=2,15)
(PIN_PAIR=3,14)
(PIN_PAIR=4,13)
(PIN_PAIR=5,12)
(PIN_PAIR=6,11)
(PIN_PAIR=7,10)
(PIN_PAIR=8,9)
(PIN_LOC=1,1)
(PIN_LOC=2,2)
(PIN_LOC=3,3)
(PIN_LOC=4,4)
(PIN_LOC=5,5)
(PIN_LOC=6,6)
(PIN_LOC=7,7)
(PIN_LOC=8,8)
(PIN_LOC=9,9)
(PIN_LOC=10,10)
(PIN_LOC=11,11)
(PIN_LOC=12,12)
(PIN_LOC=13,13)
(PIN_LOC=14,14)
(PIN_LOC=15,15)
(PIN_LOC=16,16)
}
```

example 2:

```
{PACK=cap_9_comm_sip      9-pin, pull-up-style SIP capacitor
(STYLE=C_PULLUP)          package, with alpha pin names
(SHAPE=SIP)
(TOTAL_PINS=9)
(PIN_PAIR=B,A)       pin A is to the common pull-up voltage
(PIN_PAIR=C,A)
(PIN_PAIR=D,A)
(PIN_PAIR=E,A)
(PIN_PAIR=F,A)
(PIN_PAIR=G,A)
(PIN_PAIR=H,A)
(PIN_PAIR=I,A)
(PIN_LOC=A,1)
(PIN_LOC=B,2)
(PIN_LOC=C,3)
(PIN_LOC=D,4)
(PIN_LOC=E,5)
(PIN_LOC=F,6)
(PIN_LOC=G,7)
(PIN_LOC=H,8)
(PIN_LOC=I,9)
}
```

example 3: a 10-pin, SIP Pullup/pull-down package

```
{PACK=R_PACK_9SIP
(STYLE=R_PULLUP_PULLDOWN)
(SHAPE=SIP)
(TOTAL_PINS=10)
(PIN_PAIR=2,1,10)    pin 1 is to the common pull-up voltage;
(PIN_PAIR=3,1,10)    pin 10 is to the common pull-down voltage
(PIN_PAIR=4,1,10)
(PIN_PAIR=5,1,10)
(PIN_PAIR=6,1,10)
(PIN_PAIR=7,1,10)
(PIN_PAIR=8,1,10)
(PIN_PAIR=9,1,10)
```

(PIN_LOC=1,1)
(PIN_LOC=2,2)
(PIN_LOC=3,3)
(PIN_LOC=4,4)
(PIN_LOC=5,5)
(PIN_LOC=6,6)
(PIN_LOC=7,7)
(PIN_LOC=8,8)
(PIN_LOC=9,9)
(PIN_LOC=10,10)
}

END

Format: {END}

o END identifies the end of the file

Appendix C: .FBD-File Specification

Summary

This appendix contains a brief specification for HyperLynx's ferrite-bead-modeling (.FBD) format. The .FBD format is used to create the files BSW.FBD and USER.FBD, which together define the ferrite-bead models available to BoardSim.

See Chapter 8 for details on ferrite beads in BoardSim.

Specification

```
********************** Ferrite Bead Models **********************
* Copyright 1996, HyperLynx, Inc.
*
* Description of format:
* ----------------------
* {MANUFACTURER=<name>}
* - Sets the manufacturer name for all ensuing beads
* - <name> is limited to 12 characters; truncated if longer
* - Applies until superseded by another MANUFACTURER record
* - Must be a MANUFACTURER record before the first BEAD
*
* Example:
* {MANUFACTURER=DALE}
* ----------------------
* {BEAD=<name>
*  (R_DC=<resistance>)
*  (PT1=<freq>, <impedance>)
*  (PT2=<freq>, <impedance>)
```

```
*  (PT3=<freq>, <impedance>)
* }
* - BEAD=<name>  names the bead model; <name> is limited to 30
*   characters
* - R_DC=<resistance>  gives the bead's DC resistance
* - PTx=<freq>, <impedance>  gives a frequency/impedance pair
*   for the bead
* - <freq> numbers can be scaled by a trailing 'M' or 'MHZ' to
*   give values in MegaHertz
* - Exactly 3 freq/Z pairs are required, as follows:
*   - The first point should be at (10-20)% of the nominal
*     impedance (nominal impedance is usually at 100 MHz)
*   - The last point should be the highest frequency shown in
*     the impedance vs. frequency graph for the bead
*   - If the resonant point is available, it should be PT2.
*     If the resonant point is not available, the 2nd point
*     should generally be the impedance at the nominal bead
*     impedance indicated by the vendor (usually 100 MHz).
*     If the nominal impedance and frequency are not specified,
*     then the 2nd point should be taken from the mid-point
*     of the Z-vs-freq graph for the bead
*
* WARNING: To provide accurate modeling, the data must following
*          the above rules.
*
* For example:
* {BEAD=ILB-1206-19
*  (R_DC=0.035)
*  (PT1=6.0MHZ,   4.0)
*  (PT2=100.0MHZ,19.0)
*  (PT3=500.0MHZ,27.0)
* }
**********************************************************************
```

Appendix D: .HYP-File Specification

Summary

This appendix contains the complete and official specification for HyperLynx's signal-integrity transfer (.HYP) format. The .HYP format describes a file format which PCB-layout tools can use to interface to HyperLynx's signal-integrity tools; a .HYP file contains all of the information about a PCB layout that is relevant to signal-integrity simulation. BoardSim reads .HYP files directly.

See Chapters 3 and 4 for details on creating .HYP files and loading them into BoardSim.

Detailed Specification

Introduction

This document describes a file format which PCB-layout tools can use to interface to HyperLynx's signal-integrity tools. Specifically, the format (".HYP") allows a complete-enough description of a printed circuit board for HyperLynx's products to accurately simulate the PCB's signal integrity. The .HYP file is easy to output; it is ASCII and parts of it are optional.

Version 2.0x of this specification represents a major revision of the .HYP-file format, most notably through the addition of the PADSTACK record, which allows full support for padstacks. Some of the constructs in the 1.xx version of the .HYP-file specification have been dropped from this version for clarity (most notably the PAD subrecord and the portions of the VIA subrecord referring to pads and layers).

However, BoardSim continues to support all V1.xx constructs, and is guaranteed to read all V1.xx-compliant .HYP files. This version specifies the now-preferred constructs; see the V1.10 specification for details on the older constructs. *If you are writing a new translator or making major revisions to an older one, you should use the V2.0x constructs described in this document.* Contact HyperLynx directly (support@hyperlynx.com) if you need technical support.

Version 2.1x adds support for *unrouted* segments, which can be used to implement signal-integrity checking for an unrouted or partially routed PCB (e.g., after placement but before routing).

Overview of a .HYP File's Format:

{VERSION=*number* [*comment*]}

{UNITS=*unit unit* [*comment*]}

{BOARD
  (*outline information*) [*comment*]
    ...
  (*outline information*)
}

{STACKUP
  (*layer information*) [*comment*]
    ...
  (*layer information*)
}

{DEVICES
  (*device information*) [*comment*]
  (*device information*)
    ...
  (*device information*)
}

{SUPPLIES [optional]
  (*power-supply information*)
  (*power-supply information*)
    ...
  (*power-supply information*)
}

{PADSTACK=*name*
  (*pad information*) [*comment*]
  (*pad information*)
    ...
  (*pad information*)
}

...more padstacks...

{PADSTACK=*name*
(*pad information*)
(*pad information*)
...
(*pad information*)
}

{NET=*name*
(*segment/arc/via/pin information*) [*comment*]
(*segment/arc/via/pin information*)
...
(*segment/arc/via/pin information*)
}

{NET=*name*
(*segment/arc/via/pin information*)
(*segment/arc/via/pin information*)
...
(*segment/arc/via/pin information*)
}

...more nets...

{NET=*name*
(*segment/arc/via/pin information*)
(*segment/arc/via/pin information*)
...
(*segment/arc/via/pin information*)
}

{END}
{KEY=*value*}

Notes on Detailed Syntax of .HYP File

Italicized fields are to be filled in with appropriate values.

Square brackets [ ] denote optional parameters.

Subrecord fields [lines beginning with ‘)’] can come in any order, but the specified order is recommended for readability.

Comment lines have an asterisk (*) in the first column.

All subrecords must be on a *single* line. *This is true in spite of the fact that, for readability, some subrecords in this document are shown on multiple lines.*

Curly braces { } can be separated from keywords and record ends by white space; the right brace can be on the same line as the last subrecord or on the next line.

Parentheses ( ) can be separated from keywords and subrecord ends by white space; must be on the same line as the subrecord.

Comments must be separated by at least one white space from the preceding text.

The maximum allowed line length is 180 characters.

Lines can be terminated by CR, LF, CR-LF, or LF-CR.

White space is defined as space, horizontal tab, vertical tab, linefeed, form feed, or carriage return.

Any characters are allowed in a name, except the following: white-space characters, curly braces ‘{‘ or ‘}’, parentheses ‘(‘ or ‘)’, or equal sign ‘=‘.

Numeric values can be followed by an exponent of the form
exxx or Exxx
where xxx is any integer value, positive or negative.

All numeric values can be followed by alphabetic scaling factors:
M=mega (1,000,000x)
K or k =kilo (1,000x)
m=milli (0.001x)
u or U=micro (1e-6x)
n or N=nano (1e-9x)
p or P=pico (1e-12x)

Scaling-factor suffixes may be separated from their numeric values by white space.

Scaling-factor suffixes may be followed by other alphanumeric characters, e.g., uH or pF; the additional characters are terminated by white space.

***Keyword* VERSION**

Format:
{VERSION=*number* [*comment*]}

o VERSION specifies the version of the file format
o *number* specifies the .HYP-file version number
o required record; must be the first non-blank line in file
o only one VERSION record allowed per file

Example Records for VERSION:

example 1:
{VERSION=2.01}

example 2:
{VERSION=2.01 written by program xxx}

***Keyword* UNITS**

Format:
{UNITS=*unit_system metal_thickness_unit* [*comment*]}

o UNITS identifies the units-specification record
o required record; if present, must be the second non-blank line in file
o *unit_system* specifies the measurement system for the remainder of the file; valid values are:
    ENGLISH
    METRIC
o if ENGLISH, lengths are in inches and weights in ounces; if METRIC, lengths are in centimeters and weights in grams

o *metal_thickness_unit* specifies the unit of measure for metal thickness; valid values are:
    WEIGHT
    LENGTH
o if WEIGHT, thicknesses are in ounces or grams; if LENGTH, thicknesses are in inches or centimeters
o must be on a single line
o only one UNITS record allowed per file

Example Records for UNITS:

example 1:
{UNITS=ENGLISH WEIGHT}

example 2:
{UNITS=metric length default for Euro version}

***Keyword* BOARD**

Format:
{BOARD [*comment*]
(PERIMETER_SEGMENT X1=*position* Y1=*position* X2=*position* Y2=*position*)
    [*comment*]
                ...more perimeter segments...
(PERIMETER_ARC X1=*position* Y1=*position* X2=*position* Y2=*position*
XC=*position* YC=*position* R=*radius*) [*comment*]
                ...more perimeter arcs...
}

o BOARD identifies the board-perimeter-specification record
o optional record; if missing, the board outline is assumed to be rectangular, and is determined by the position of the most-extreme components or traces
o the BOARD record can also be used to specify holes and other cut-outs in the interior of the board

o PERIMETER_SEGMENT record reports information for straight segments of the board's perimeter
o units throughout subrecord are as specified in the UNITS record
o X1, Y1, X2, or Y2 *position* is the x or y position of end 1 or end 2 of the segment

o PERIMETER_ARC record reports information for arcs on the board's perimeter
o units throughout subrecord are as specified in the UNITS record
o X1, Y1, X2, or Y2 *position* is the x or y position of end 1 or end 2 of the arc
o XC or YC *position* is the position of the center of the arc
o R *radius* is the radius of the arc
o BoardSim draws the arc *counterclockwise* from X1,Y1 to X2,Y2; note that this is the opposite of how BoardSim draws trace-segment arcs (clockwise)

o each PERIMETER_SEGMENT record must be on a single line
o only one BOARD record allowed per file

Example Records for BOARD:

example 1:
{BOARD rectangular board with edge connector
(PERIMETER_SEGMENT X1=0.0 Y1=0.0 X2=12.0 Y2=0.0)
(PERIMETER_SEGMENT X1=12.0 Y1=0.0 X2=12.0 Y2=-5.0)
(PERIMETER_SEGMENT X1=12.0 Y1=-5.0 X2=3.0 Y2=-5.0)
(PERIMETER_SEGMENT X1=3.0 Y1=-5.0 X2=3.0 Y2=-5.5) connector segment
(PERIMETER_SEGMENT X1=3.0 Y1=-5.5 X2=0.5 Y2=-5.5) connector segment
(PERIMETER_SEGMENT X1=0.5 Y1=-5.5 X2=0.5 Y2=-5.0) connector segment
(PERIMETER_SEGMENT X1=0.5 Y1=-5.0 X2=0.0 Y2=-5.0)
(PERIMETER_SEGMENT X1=0.0 Y1=-5.0 X2=0.0 Y2=0.0)
}

***Keyword* STACKUP**

Format:
{STACKUP [*comment*]
(SIGNAL T=*thickness* [P=*plating_thickness*] [C=*constant*]
[TC=*temperature_coefficient*]
L=*layer_name* [M=*material_name*])
[*comment*]
...more signal layers...
(DIELECTRIC T=*thickness* [C=*constant*] [L=*layer_name*] [M=*material_name*])
[*comment*]
...more dielectric layers...

(PLANE T=*thickness* [C=*constant*] [TC=*temperature_coefficient*] [L=*layer_name*]
[M=*material_name*])
[*comment*]
...more plane layers...
}

o STACKUP identifies the stackup-specification record
o optional record; if missing, stackup is built as layer names encountered in NET and PADSTACK subrecords; first layer encountered becomes top layer, next becomes second-from-top, etc.
o also, if missing, the user is warned about stackup deficiencies and allowed to fix them
o subrecords can be mixed in any order that correctly describes the board's top-to-bottom construction; the first SIGNAL or PLANE layer will be numbered 1 (top), the next 2 (second from the top), and so forth

o SIGNAL record reports information for signal layers
o units throughout subrecord are as specified in the UNITS record
o *thickness* specifies how thick the metal is
o optional *plating_thickness* specifies how thick the metal plating is; if missing, assumed to be 0.0
o optional *constant* specifies resistivity; if missing, assumed to be 1.724e-8 (copper)
o optional *temperature_coefficient* specifies temperature coefficient of resistivity; if missing, assumed to be 0.00393 (copper)
o *layer_name* specifies a name for the layer; maximum of 20 characters; layers reported in NET subrecords are matched to these names
o optional *material_name* specifies a name for the trace material; maximum of 20 characters

o DIELECTRIC record reports information for dielectric layers
o units throughout subrecord are as specified in the UNITS record
o *thickness* specifies how thick the dielectric is
o optional *constant* specifies the dielectric constant; if missing, assumed to be 4.8 (FR-4)
o optional *layer_name* specifies a name for the layer; maximum of 20 characters
o optional *material_name* specifies a name for the dielectric material; maximum of 20 characters

o PLANE record reports information for power-plane layers
o units throughout subrecord are as specified in the UNITS record

o *thickness* specifies how thick the metal is
o optional *constant* specifies resistivity; if missing, assumed to be 1.724e-8 (copper)
o optional *temperature_coefficient* specifies temperature coefficient of resistivity; if missing, assumed to be 0.00393 (copper)
o optional *layer_name* specifies a name for the layer; maximum of 20 characters; layers reported in NET subrecords are matched to these names
o optional *material_name* specifies a name for the plane material; maximum of 20 characters

o each SIGNAL, PLANE, or DIELECTRIC record must be on a single line
o only one STACKUP record allowed per file

Example Records for STACKUP:

example 1:

```
{STACKUP    4-layer board; units for this example are English, weight
(SIGNAL  T=0.0014  P=0.0014  L=TOP)   1-oz base and plating; resistivity = 0.0
(DIELECTRIC  T=0.02  C=4.8)
(PLANE  T=0.0014  L=2)                plating and resistivity = 0.0
(DIELECTRIC  T=0.01  C=4.8)
(PLANE  T=0.0014  L=3)                plating and resistivity = 0.0
(DIELECTRIC  T=0.02  C=4.8)
(SIGNAL  T=0.0014  P=0.0014  L=BOTTOM)
}
```

example 2:

```
{STACKUP    same example, except this one includes resistivities and some layer and material names
(SIGNAL  T=14e-3  P=14e-3  C=0.017u  L=TOP)  1-oz base and plating; resistivity = copper's
(DIELECTRIC  T=0.02  C=4.8  M=FR4)
(PLANE  T=14e-3  C=0.017u  L=Vcc)
(DIELECTRIC  T=0.01  C=4.8  M=FR4)
(PLANE  T=14e-3  C=0.017u  L=GND)
(DIELECTRIC  T=0.02  C=4.8  M=FR4)
(SIGNAL  T=14e-3  P=14e-3  C=0.017u  L=BOTTOM)
}
```

***Keyword* DEVICES**

Format:

{DEVICES [*comment*]
(? REF=*reference_designator* [NAME=*value*] L=*layer_name*
[PKG=*package_type*])
[comment]
...more components of unknown type...

(IC REF=*reference_designator* [NAME=*value*] L=*layer_name*
[PKG=*package_type*])
[comment]
...more ICs...

(R REF=*reference_designator* [VAL=*value*] L=*layer_name*) [comment]
...more resistors...

(C REF=*reference_designator* [VAL=*value*] L=*layer_name*) [comment]
...more capacitors...

(J REF=*reference_designator* [NAME=*value*] L=*layer_name*) [comment]
...more connectors...

(L REF=*reference_designator* [VAL=*value*] L=*layer_name*) [comment]
...more inductors...

(CR REF=*reference_designator* [NAME=*value*] L=*layer_name*) [comment]
...more diodes...

(BD REF=*reference_designator* [NAME=*value*] L=*layer_name*) [comment]
...more beads...
}

o DEVICES identifies the device-specification record
o required record
o any non-zero number of these subrecords may follow: ?, IC, R, C, J, L, CR, BD
o *value* is optional for all subrecords, but highly recommended; for ?, R, C, and L subrecords, if not supplied, the user must specify the component value
o for IC subrecords, *package_type* is optional, but recommended

o *NOTE:* In V2.00 of the .HYP specification, HyperLynx recommends using the ? record for *all* components, since in BoardSim the user can flexibly map reference designator prefixes into component types. If it is desired to provide the user a mechanism for specifying or "locking down" a specific component's type regardless of reference designator (from the schematic editor, for example), then the non-? records are useful.

? subrecord

o ? record reports information for components of "unknown" type; the user can specify the type through BoardSim's user interface based on reference-designator-prefix mappings (e.g., R=resistor, U=IC, etc.)
o *reference_designator* specifies the component's reference designator
o optional *value* specifies the component's name
o *layer_name* is the name of the layer which the component is on
o *package_type* specifies the type of component packaging; valid values are:

DIP
SO
PLCC
LCC
PGA
QFP
BGA
SSOP
TSSOP
TQFP

IC subrecord

o IC record reports information for ICs (drivers and receivers)
o *reference_designator* specifies the IC's reference designator
o optional *value* specifies the IC's name
o *layer_name* is the name of the layer which the device is on
o *package_type* specifies the type of IC packaging; valid values are:

DIP
SO
PLCC
LCC
PGA
QFP
BGA
SSOP
TSSOP
TQFP

R subrecord
- R record reports information for resistors
- *reference_designator* specifies the resistor's reference designator
- optional *value* specifies the resistor's value, in ohms
- *layer_name* is the name of the layer which the device is on

C subrecord
- C record reports information for capacitors
- *reference_designator* specifies the capacitor's reference designator
- optional *value* specifies the capacitor's value, in Farads
- *layer_name* is the name of the layer which the device is on

J subrecord
- J record reports information for connectors
- *reference_designator* specifies the connector's reference designator
- optional *value* specifies the connector's name
- *layer_name* is the name of the layer which the device is on

L subrecord
- L record reports information for inductors
- *reference_designator* specifies the inductor's reference designator
- optional *value* specifies the inductor's value, in Henries
- *layer_name* is the name of the layer which the device is on

CR subrecord
- CR record reports information for diodes
- *reference_designator* specifies the diode's reference designator
- optional *value* specifies the diode's name
- *layer_name* is the name of the layer which the device is on

BD subrecord
- BD record reports information for ferrite beads
- *reference_designator* specifies the bead's reference designator
- optional *value* specifies the bead's name
- *layer_name* is the name of the layer which the device is on

- each device-type record must be on a single line
- only one DEVICES record is allowed per file

Example Records for DEVICES:

example 1:

```
{DEVICES
(?  REF=U13  NAME=XC_4010  L=TOP)        Xilinx LCA; on top layer
(?  REF=U20  NAME=74AC244  L=8  P=SO)  Advanced CMOS 244; on layer 8;
SOIC
(?  REF=R15  VAL=1K  L=TOP)                 resistance = 1000 ohms
(?  REF=C1  VAL=22p  L=8)                   capacitance = 22 pF
(?  REF=L100  VAL=5n)                       inductance = 0.005 uH
(?  REF=J2  NAME=Amp_HDI  L=TOP)         Amp HDI connector
(?  REF=capacitor_tweak  L=8)               error: missing 'VAL=' record
(C  REF=FOO VAL=10u  L=8)              locked to component type "capacitor"
}
```

***Keyword* SUPPLIES**

Format:

```
{SUPPLIES        [comment]
(S NAME=net_name  VAL=voltage V?=yes/no C?= yes/no)     [comment]
                    ...more power-supply definitions...
}
```

o optional SUPPLIES identifies the power-supply-specification record
o this record is normally not used; BoardSim has built-in provisions for automatically identifying power-supply nets and allows users to edit the power-supply list in a dialog box; this record should only be included if the PCB-layout tool or translator independently offers the user a dialog box in which to identify and set values for power-supply nets

o S record reports information for individual power-supply nets
o *net_name* specifies the name of the power-supply net
o *voltage* specifies the power-supply voltage associated with the net
o *yes/no* is either 'yes' or 'no'; for V?, specifies whether the net's voltage value is known or was defaulted; set to 'yes' if known, 'no' if not (e.g., for net 'VCC', V?=no); for C?, specifies whether the net's identification (i.e., conversion) was questionable or not; set to 'yes' if questionable, 'no' if not (e.g., for net 'V5', C?=yes)

Example Records:

example 1:
{SUPPLIES power-supply values redefined by user
(S NAME=Vcc VAL=3.3 V?=NO C?=NO)
(S NAME=RAIL VAL=12 V?=YES C?=NO) C?=NO because net added manually by user
(S NAME=V3 VAL=3 V?=NO C?=YES)
}

***Keyword* PADSTACK**

Format:
{PADSTACK=*padstack_name, [drill_size]*
(*layer_name, pad_shape, pad_sx, pad_sy, pad_angle, [thermal_clear_shape], [thermal_clear_sx], [thermal_clear_sy], [thermal_clear_angle], [pad_type]*)
[comment]
...more padstack elements...
(*layer_name, pad_shape, pad_sx, pad_sy, pad_angle, [thermal_clear_shape], [thermal_clear_sx], [thermal_clear_sy], [thermal_clear_angle], [pad_type]*)
[comment]
}

o PADSTACK identifies the padstack-specification record
o optional record, but only if .HYP-file V1.xx constructs are used; for new translators, using PADSTACK is highly recommended
o at least one padstack-element subrecord must follow
o a comment is *not* allowed on the PADSTACK line, owing to the fixed-position nature of this record; comments *are* allowed on ensuing lines with padstack-element subrecords
o *padstack_name* is a unique a name for this padstack; maximum of 32 characters and cannot contain white space
o optional *drill_size* is the diameter of the padstack's drill hole, if any; if there is no drill hole, MUST omit this parameter; *drill_size* is the diameter of the drill hole *before* plating (hole plating is specified by the STACKUP record and is the same as the outside-layer plating thickness)
o units are as specified in the UNITS record

o *NOTE:* The format of the PADSTACK record differs somewhat from the other records in the .HYP file, in that PADSTACK is fixed-position rather than keyword-based. This means that all parameters must come in the specified order (optional parameters are always at the end of a line). This change (to a fixed-position format) has been introduced to shorten the length of the .HYP file.

Padstack-element subrecord

o padstack-element record reports information for individual pads in a padstack
o units throughout subrecord are as specified in the UNITS record
o *layer_name* is name of the layer which the pad is on; valid values are:

user_name
MDEF
ADEF

where:

user_name is any user-created name that does not violate .HYP-file syntax;

MDEF is a special, reserved layer name indicating that all layers have this default metal pad unless otherwise specified with an explicit padstack-element subrecord;

ADEF is a special, reserved layer name indicating that all plane layers defined in the STACKUP record will have this default anti-pad unless otherwise specified with an explicit padstack-element subrecord (an 'anti-pad' defines the size and shape of a non-conducting hole in a plane layer through which a via passes)

o if *layer_name* does not match a name already in the board's stackup, a new layer with name *layer_name will* be created
o *pad_shape* is the shape of the pad; valid values are: 0, 1, 2
where '0' means oval or round (*pad_sx* = *pad_sy* if round)
'1' means rectangular or square (*pad_sx* = *pad_sy* if square)
'2' means oblong (oblong shape is a rectangle with rounded corners)
o *pad_sx* or *pad_sy* is the x or y dimension of the pad
o *pad_angle* is the counter-clockwise rotation angle of the pad in degrees; rotation angle can range from 0.0 to +/-359.999; valid angular resolution is 0.001 degree; 0 (without a decimal point) is the recommended way of specifying 'no rotation'
o *thermal_clear_shape (required only if pad_type is T, thermal relief)* is the shape of the clear area around a thermal-relief pad; valid values are: 0, 1 , 2
where:
'0' means oval or round (*thermal_clear_sx* = *thermal_clear_sy* if round)
'1' means rectangular or square (*thermal_clear_sx* = *thermal_clear_sy* if square)

‘2’ means oblong (oblong shape is a rectangle with rounded corners)
o *thermal_clear_sx* or *thermal_clear_sy (required only if pad_type is T, thermal relief)* is the x or y dimension of the clear area around a thermal-relief pad
o *thermal_clear_angle (required only if pad_type is T, thermal relief)* is the counter-clockwise rotation angle of the clear area around a thermal-relief pad, in degrees; rotation angle can range from 0.0 to +/-359.999; valid angular resolution is 0.001 degree; 0 (without a decimal point) is the recommended way of specifying ‘no rotation’
o if any one of *thermal_clear_sx*, *thermal_clear_sy*, or *thermal_clear_angle* is specified, all three must be specified and the pad_type must be specified as T
o optional *pad_type* specifies whether the pad is a normal metal pad, an anti-pad, or a thermal relief; if missing and three x/y/angle parameters specified, the pad is assumed to be a normal metal pad; if missing and six x/y/angle parameters specified, the pad is assumed to be a thermal-relief pad; valid values are:

M
A
T

where:
‘M’ means normal metal pad (used for signal vias and component pads)
‘A’ means anti-pad (used to define non-conducting hole through a plane layer)
‘T’ means a thermal-relief pad (used to connect a pad to a plane layer)

o *NOTE:* BoardSim does not require the use of anti-pads; a padstack not using anti-pads can simply omit information about plane layers; if anti-pads are not used in the .HYP file, the simulation accuracy is only very slightly reduced.

Example Records for PADSTACK:

example 1: rectangular surface-mount pad on the top layer; spaces after commas are not required

```
{PADSTACK=SMT001
(TOP, 1, 0.04, 0.08, 0)
}
```

example 2: round surface-mount pad on bottom layer; board has 16 layers and layer names are numeric

```
{PADSTACK=PSTK-054
(16,0,0.04,0.04,0)
}
```

example 3: oblong surface-mount pad on bottom layer, at 45-degree angle

{PADSTACK=SMT003
(BOTTOM, 2, 0.04, 0.06, 45.0)
}

example 4: through-hole, 60-mil-diameter round pads on all layers with a 30-mil-diameter drill hole
{PADSTACK=STD1, 0.03
(MDEF, 0, 0.06, 0.06, 0)
}

example 5: through-hole, 60-mil-diameter round pads on all layers *except* layer 1; layer 1 has a 60-mil-wide square pad; drill hole is 30 mils in diameter
{PADSTACK=STD1, 0.03
(1, 1, 0.06, 0.06, 0)
(MDEF, 0, 0.06, 0.06, 0)
}

example 6: through-hole padstack; 60-mil-wide square pad on layers 1 and 8; 60-mil-diameter round pads on layers 2 and 7; no pads on layers 3 and 6; 60-mil-diameter round thermal-relief pad with 100-mil-diameter round clear area on layer 4;100-mil-diameter round anti-pad on layer 5
{PADSTACK=STD1, 0.03
(1, 1, 0.06, 0.06, 0)
(2, 0, 0.06, 0.06, 0, M) pad type 'M' not required
(4, 0, 0.06, 0.06, 0, 0, 0.10, 0.10, 0, T) thermal pad
(5, 0, 0.10, 0.10, 0, A) anti-pad
(7, 0, 0.06, 0.06, 0)
(8, 1, 0.06, 0.06, 0)
}

***Keyword* NET**

Format:
{NET=*name* [*comment*]
(SEG X1=*position* Y1=*position* X2=*position* Y2=*position* W=*width* L=l*ayer_name*) [*comment*]
...more segments...

(ARC X1=*position* Y1=*position* X2=*position* Y2=*position* XC=*position* YC=*position* R=*radius* W=*width* L=l*ayer_name*) [*comment*]
...more arcs...

(VIA X=*position* Y=*position* P=*padstack_name*) [*comment*]
...more vias...

(PIN X=*position* Y=*position* R=*reference_designator.pin_name* P=*padstack_name* [F=*function*]) [*comment*]
...more pins...
(USEG X1=*position* Y1=*position* L1=*layer_name* X2=*position* Y2=*position* L2=*layer_name* ZL=*layer_name* ZW=*width* ZLEN=*length*) [*comment*]
...more unrouted segments...
}

o NET identifies a net-specification record
o *name* is the net's name
o any non-zero number of these subrecords may follow: SEG, ARC, VIA, PIN, and USEG

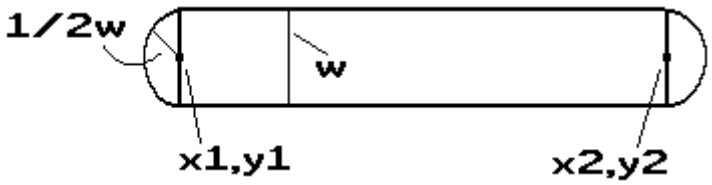
SEG subrecord
o SEG record reports information for straight metal trace segments
o units throughout subrecord are as specified in the UNITS record
o X1, Y1, X2, or Y2 *position* is the x or y position of end 1 or end 2 of the segment
o *width* is the segment's width
o *layer_name* is name of the layer which the segment is on; if *layer_name* does not match a name already in the board's stackup, a new layer with name *layer_name* will be created

Example Records:

example 1: a simple trace segment; 10 mils wide; on layer "top"
(SEG X1=1.0 Y1=0.5 X2=1.0 Y2=1.8 W=0.01 L=top)

example 2: a 45-degree trace segment; 15 mils wide; on layer 3; note lack of case-sensitivity
(seg x1=1 y1=1 x2=2 y2=2 w=15e-3 l=3)

How BoardSim draws a segment:



ARC subrecord
o ARC record reports information for curved metal trace segments
o units throughout subrecord are as specified in the UNITS record
o X1, Y1, X2, or Y2 *position* is the x or y position of end 1 or end 2 of the arc
o XC or YC *position* is the x or y position of the center of the arc
o R *radius* is the radius of the arc
o *width* is the arc's metal width
o *layer_name* is name of the layer which the arc is on; if *layer_name* does not match a name already in the board's stackup, a new layer with name *layer_name* will be created
o BoardSim draws the arc *clockwise* from X1,Y1 to X2,Y2; note that this is the opposite of how BoardSim draws board-outline arcs (counterclockwise)

Example Records:

example 1: a curved trace segment; radius of a half inch; 10 mils wide; on layer "top"
(ARC X1=1.0 Y1=1.0 X2=1.0 Y2=2.0 XC=1.0 YC=1.5 R=0.5 W=0.01 L=top)

VIA subrecord
o VIA record reports information for vias; use for vias where no component pin is present; if there is a component pin, use a PIN record instead
o units throughout subrecord are as specified in the UNITS record
o X or Y *position* is the x or y position of the via

o *padstack_name* is the name of the padstack associated with the via; must match the name of a padstack reported in a PADSTACK record

Example Records:

example 1: a via connecting two trace segments; pads and layers defined in padstack STD1
(VIA X=2.35 Y=4.05 P=STD1)

PIN subrecord

o PIN record reports information for a component pin
o units throughout subrecord are as specified in the UNITS record
o X or Y *position* is the x or y position of the component pin
o *reference_designator.pin_name* is the reference designator and pin name of the component pin
o *padstack_name* is the name of the padstack associated with the pin; must match the name of a padstack reported in a PADSTACK record
o optional *function* specifies the functionality *during simulation* of the pin, if on an IC; valid values are:

SIM_OUT
SIM_BOTH

o do not report this information unless it is known to be correct
o correct usage of SIM_OUT and SIM_BOTH is described in the table below:

If an IC pin is known to be an output only (not 3-state or bi-directional)...	...set F=SIM_OUT
If an IC pin is known to be bi-directional or 3-state AND the user has specified that he wants the pin to be a driver during simulation...	...set F=SIM_OUT
If an IC pin is known to be bi-directional or 3-state (not output only) AND the user has specified that he wants the pin to be both a driver and receiver during simulation...	...set F=SIM_BOTH
If an IC pin is known to be bi-directional or 3-state (not output only) AND the user has not provided any	

information about its function during simulation...	...set F=SIM_BOTH
If an IC pin is known to be an input only...	...do not use the F= subrecord
If an IC pin's functionality is not known...	...do not use the F= subrecord
If the component is not known for certain to be an IC...	...do not use the F= subrecord

Example Records:

example 1: pin 4 on U3, with surface-mount pad defined by padstack SMD1
(PIN X=7.55 Y=9.15 R=U3.4 P=SMD1)

example 2: emitter of Q2
(PIN X=0.05 Y=1 R=Q2.E P=FOO)

example 3: pin 57 U29; IC pin is bi-directional but user has specified that he wants it to drive during simulation
(PIN X=2 Y=2.95 R=U29.57 P=SMD2 F=SIM_OUT)

example 4: pin 57 U29; IC pin is bi-directional but user has specified that he wants it to sometimes drive and sometimes receive during simulation
(PIN X=2 Y=2.95 R=U29.57 P=SMD2 F=SIM_BOTH)

o as many of each net subrecord (SEG, ARC, VIA, and PIN) per NET record as needed are allowed; subrecords can come in any order
o as many NET records per file as needed are allowed

USEG subrecord
o USEG record reports information for *unrouted* straight metal trace segments; these can be output for a placed-but-unrouted board, or for the unrouted portions of a partially routed board
o units throughout subrecord are as specified in the UNITS record
o USEG record is divided into a physical and an electrical portion
Physical portion (used for establishing connectivity):

o X1, Y1, X2, or Y2 *position* is the x or y position of either of the points to be connected by the unrouted segment
o L1 or L2 *layer_name* is name of the layer that the end 1 or end 2 connection point is on (unrouted segments can begin on one layer and end on another, for example to connect two surface-mount pins on opposite sides of a board); if *layer_name* does not match a name already in the board's stackup, a new layer with name *layer_name* will be created
Electrical portion (used for calculating impedance and delay):
o width is the width to use for calculating the unrouted segment's impedance and delay
o ZL *layer_name* is the name of the layer to use for calculating impedance and delay; can be different than L1 and L2
o *length* is the length to use for calculating delay; can be different than the straight length between points (X1,Y1) and (X2,Y2)

Example Records:

example 1: a single-layer unrouted segment on layer "top"; 10 mils wide; use layer "top" for impedance calculation, and a length longer than the point1-to-point2 distance
(USEG X1=1.0 Y1=0.5 L1=top X2=1.0 Y2=1.8 L2=top ZL=top ZW=0.01 ZLEN=1.5)

example 2: a two-layer unrouted segment that connects SMD components on layers "top" and "bottom"; use an inner layer for impedance calculation
(useg X1=2.0 Y1=2.5 L1=top X2=3.0 Y2=4.8 L2=bottom ZL=inner2 ZW=0.01 ZLEN=3.65)

o as many of each net subrecord (SEG, ARC, VIA, PIN, and USEG) per NET record as needed are allowed; subrecords can come in any order
o as many NET records per file as needed are allowed

Example Records for NET:

example 1: a complete trace, with 4 segments; 3 vias connecting the segments; an IC pin at both ends
{NET=clock
(PIN X=0.0 Y=1.0 R=U1.10 P=PS005) surface-mount IC pin

```
(SEG  X1=0.0  Y1=1.0  X2=1.0  Y2=1.0  W=.01  L=TOP)      segment 1
(SEG  X1=1.0  Y1=1.0  X2=1.95  Y2=1.0  W=.01  L=TOP)    segment 2
(VIA  X=1.95  Y=1.0  P=PS002)                           connecting via
(SEG  X1=1.95  Y1=1.0  X2=1.95  Y2=1.2  W=.008  L=2)    segment 3
(VIA  X=1.95  Y=1.2  P=PS002)                           connecting via
(SEG  X1=1.95  Y1=1.2  X2=2.65  Y2=1.2  W=.008  L=6)    segment 4
(VIA  X=2.65  Y=1.24  P=PS002)                          connecting via
(SEG  X1=2.65  Y1=1.2  X2=4  Y2=2.2  W=.01  L=BOT)      segment 5
(PIN  X=4  Y=2.2  R=U17.4  P=PS005)                     surface-mount IC pin
}
```

example 2: the same trace as example 1, but built from a single unrouted segment; use layer “inner1” for impedance calculations, and use longer-than-Manhattan distances for delay calculations

```
{NET=clock
(PIN  X=0.0  Y=1.0  R=U1.10  P=PS005)                   surface-mount IC pin
(USEG  X1=0.0  Y1=1.0  L1=TOP  X2=4  Y2=2.2  L2=BOT  ZL=inner1  ZW=.008
ZLEN=6.0)
(PIN  X=4  Y=2.2  R=U17.4  P=PS005)                     surface-mount IC pin
}
```

example 3: the same trace as example 1, but built with a mixture of two routed segments and one unrouted

```
{NET=clock
(PIN  X=0.0  Y=1.0  R=U1.10  P=PS005)                   surface-mount IC pin
(SEG  X1=0.0  Y1=1.0  X2=1.0  Y2=1.0  W=.01  L=TOP)     routed segment
(SEG  X1=1.0  Y1=1.0  X2=1.95  Y2=1.0  W=.01  L=TOP)    routed segment
(USEG  X1=1.95  Y1=1.0  L1=TOP  X2=4  Y2=2.2  L2=BOT  ZL=inner1 ZW=.008
ZLEN=6.0)
(PIN  X=4  Y=2.2  R=U17.4  P=PS005)                     surface-mount IC pin
}
```

***Keyword* END**

Format:

```
{END}
```

- END identifies the end of the PCB-layout data

Keyword KEY

Format:
{KEY=*value*}

- KEY identifies the source of the .HYP-file translation
- *value* is an identification number unique to each .HYP file; it should never be modified

A Complete .HYP File with Two Nets:

```
{VERSION=2.10}
{UNITS=ENGLISH  WEIGHT}

{BOARD
(PERIMETER_SEGMENT  X1=0.0  Y1=0.0  X2=8.0  Y2=0.0)
(PERIMETER_SEGMENT  X1=8.0  Y1=0.0  X2=8.0  Y2=8.0)
(PERIMETER_SEGMENT  X1=8.0  Y1=8.0  X2=0.0  Y2=8.0)
(PERIMETER_SEGMENT  X1=0.0  Y1=8.0  X2=0.0  Y2=0.0)
}

{STACKUP
(SIGNAL  T=0.0014  L=TOP)
(DIELECTRIC  T=0.010)
(SIGNAL  T=0.0014  L=2)
(DIELECTRIC  T=0.020)
(PLANE  T=0.0014  L=VCC)
(DIELECTRIC  T=0.020)
(PLANE  T=0.0014  L=GND)
(DIELECTRIC  T=0.020)
(SIGNAL  T=0.0014  L=3)
(DIELECTRIC  T=0.010)
(SIGNAL  T=0.0014  P=0.0014  L=BOTTOM)
}

{DEVICES
(?  REF=U15  NAME=74ACT240  L=BOTTOM)
(?  REF=U222  NAME=LCA3000  L=TOP)
(?  REF=U17  NAME=F161  L=TOP)
(?  REF=R15  NAME=68  L=BOTTOM)
(?  REF=R29  NAME=100  L=BOTTOM)
}

{PADSTACK=THR001, 0.03
(MDEF, 0, 0.06, 0.06, 0)
}

{PADSTACK=SMD001
```

```
(TOP, 1, 0.04, 0.06, 0)
}

{NET=SLOW_CARRY
(PIN  X=1.250  Y=0.950  R=U15.18  P=SMD001)
(SEG  X1=1.250  Y1=0.950  X2=1.350  Y2=1.050  W=0.015  L=TOP)
(SEG  X1=1.350  Y1=1.050  X2=1.600  Y2=1.050  W=0.015  L=TOP)
(VIA  X=1.600  Y=1.050  P=THR001)
(SEG  X1=1.600  Y1=1.050  X2=1.650  Y2=1.100  W=0.015  L=BOTTOM)
(VIA  X=1.650  Y=1.100  P=THR001)
(SEG  X1=1.650  Y1=1.100  X2=1.650  Y2=1.150  W=0.015  L=TOP)
(PIN  X=1.650  Y=1.150  R=R15.2  P=SMD001)
(SEG  X1=1.650  Y1=1.150  X2=2.050  Y2=1.150  W=0.015  L=TOP)
(SEG  X1=2.050  Y1=1.150  X2=2.050  Y2=1.050  W=0.015  L=TOP)
(VIA  X=2.050  Y=1.050  P=THR001)
(SEG  X1=2.050  Y1=1.050  X2=2.050  Y2=2.500  W=0.015  L=2)
(VIA  X=2.050  Y=2.500  P=THR001)
(SEG  X1=2.050  Y1=2.500  X2=2.150  Y2=2.600  W=0.015  L=TOP)
(PIN  X=2.150  Y=2.600  R=U222.85  P=SMD001)
}

{NET=FAST_CARRY
(PIN  X=4.600  Y=0.600  R=U17.2  P=THR001)
(SEG  X1=4.600  Y1=0.600  X2=4.550  Y2=0.550  W=0.015  L=2)
(SEG  X1=4.550  Y1=0.550  X2=4.200  Y2=0.550  W=0.015  L=2)
(SEG  X1=4.200  Y1=0.550  X2=4.150  Y2=0.600  W=0.015  L=2)
(SEG  X1=4.150  Y1=0.600  X2=4.150  Y2=1.350  W=0.015  L=2)
(VIA  X=4.150  Y=1.350  P=THR001)
(SEG  X1=4.150  Y1=1.350  X2=4.150  Y2=1.500  W=0.015  L=TOP)
(SEG  X1=4.150  Y1=1.500  X2=4.100  Y2=1.550  W=0.015  L=TOP)
(SEG  X1=4.100  Y1=1.550  X2=2.350  Y2=1.550  W=0.015  L=TOP)
(SEG  X1=2.350  Y1=1.550  X2=2.300  Y2=1.500  W=0.015 L=TOP)
(PIN  X=2.300  Y=1.500  R=R29.1  P=THR001)
}

{END}
```

Appendix E: Table of Dielectric Constants

Summary

This appendix lists dielectric constants (i.e., relative permittivities) for a number of common printed-circuit-board materials.

Dielectric Table

Material Name	Fiber Material	Bulk Material	Dielectric Constant
Rogers RO2800			2.9
Rogers RO4003			3.38
Rogers RO4350			3.48
FR-2	paper	phenolic	4.3
FR-4	fiberglass	epoxy	4.8
FR-5	fiberglass	epoxy	4.8
G-2	staple-glass	phenolic	5.1

Material Name	Fiber Material	Bulk Material	Dielectric Constant
G-5	fiberglass	melamine	7.3
G-7	fiberglass	silicone	3.9
G-10	fiberglass	epoxy	4.8
G-11	fiberglass	epoxy	4.8
XXPC	paper	phenolic	4.1
N-1	Nylon	phenolic	3.6
GORE-PTFE	expanded PTFE/glass	epoxy	2.8
alumina			10.0

Appendix F: Setting Simulation Options

Summary

This appendix describes several advanced simulation options, available in BoardSim's user interface, that affect which algorithms BoardSim runs during simulation.

HyperLynx recommends against changing these options except under special circumstances. You should contact HyperLynx for technical support before changing the default settings

Advanced Simulation Options

To see the advanced simulation options:

1. From the Options menu, choose Preferences. The Options dialog box opens.
2. Click on the Advanced tab. A dialog box opens, warning that you should not change the option settings.
3. Click Yes to proceed.

Inform Users of Zero Length…

When this option is enabled, if BoardSim finds a zero-length routing segment in the .HYP file (might result from a PCB-layout tool which does not "clean up" its database), it will report a warning when it tries to simulate the net. Since

zero-length segments rarely cause a problem, it is almost always advisable to leave this option disabled.

Find and Repair Invalid...

When this option is enabled, BoardSim searches for trace segments that are partially overlapped or that intersect. (Such conditions are an artifact of your PCB-layout editor not maintaining a "clean database.") This optimization results in correct electrical connectivity and improved delay modeling.

Simulate Frequency-Independent Line Loss

When this option is enabled, BoardSim invokes a special algorithm that more-accurately finds final DC simulation values in the case of long/high-resistance transmission lines. This simulation is not frequency-dependent (as in a true lossy-line simulator), but it gives an approximate loss effect that helps the simulator better converge on final DC values.

Note: *Future versions of BoardSim will offer improved lossy simulation.*

Segment Threshold for Auto…

See Chapter 4, section "Changing the Number-of-Segments Threshold."

For EMC, Ignore Traces… *(BoardSim Crosstalk Only)*

See the EMC Analysis User's Guide.

For Crosstalk, Ignore Coupling… *(BoardSim Crosstalk Only)*

See the Crosstalk Analysis User's Guide.

Combine Line Segments…

When this option is enabled, BoardSim combines adjoining trace segments that have the same impedance into a single, longer segment. This optimization results in a faster transient simulation.

High-Accuracy Field Solver *(BoardSim Crosstalk Only)*

When this option is enabled, the field solver is forced to run with higher-than-normal spatial resolution. The default (non-high-accuracy) mode has been set to give excellent accuracy for nearly any problem on which BoardSim would normally run. However, in rare cases (e.g., extremely narrow trace separations), the higher-accuracy mode may be preferable. However, high-accuracy mode runs substantially slower than normal mode, so should be enabled only when absolutely necessary.

Use Field-Solver Cache *(BoardSim Crosstalk Only)*

BoardSim Crosstalk's field solver normally uses a smart "cache" to prevent having to re-solve cross sections which it has already encountered and analyzed. (Such repeated cross sections occur frequently during crosstalk analysis on a given PCB.) Disabling this feature turns off the cache, which may seriously degrade the product's performance. Since there is rarely (if ever) any reason to disable the cache, it is advisable to never disable this option.

Maximum Line-Length Tolerance

This percentage tells BoardSim how large an error (as a percentage of the segment's total delay) it can make in modeling a segment's propagation delay. This tolerance applies only to very short segments, and even for short segments, is rarely invoked by BoardSim.

Max DC Convergence… *and* Min DC Convergence…

The Max DC Convergence Iterations threshold tells BoardSim the maximum number of times it can re-simulate to find a driver's initial DC voltage. Such iterative simulation is only required when there are multiple IBIS (any model)

or .MOD/.PML bipolar IC models present on a net. The default value is almost always sufficient to stabilize a multi-driver DC simulation; in very rare cases the number of iterations should be increased to give more-accurate results.

The Min DC Convergence Threshold tells BoardSim how tightly each driver's voltage in a multi-driver DC simulation must converge before the DC simulation can safely be considered "successful." The default value should never be changed unless a circuit and combination of drivers has difficulty converging, and might benefit from a "relaxed" convergence criterion.

Note: *The DC convergence parameters are advanced parameters. If you find a circuit which you believe would benefit from a change in the parameters, contact HyperLynx for advice.*

Restoring Default Settings

To restore the default settings for all of the advanced simulation options:

1. From the Options menu, choose Preferences. The Options dialog box opens.
2. Click on the Advanced tab. A dialog box opens, warning that you should not change the option settings. Click Yes to proceed.
3. Click the Restore Defaults button.

App Note: Creating IBIS Models

Summary

This application note discusses how to create an IBIS model. Most or all of the information here can be gleaned from the IBIS specification, contained in Appendix A. However, you may find this discussion easier to read and follow than the specification alone.

Detailed Note

.IBS models are not difficult to create. They are based on data which can be collected from real devices, or derived from proprietary silicon models. Whatever the data-collection method, the resulting .IBS model will accurately describe the device's behavior without giving away proprietary information about the silicon's design.

Before attempting to create a .IBS model, you should read the IBIS V2.1 specification, contained in Appendix A. The specification is also available in a text file: "IBIS21.TXT" in the main BoardSim directory.

Elements of a .IBS Model

The following elements are required in a .IBS model:

- "default" package R, L, and C
- pin/signal list
- model for each unique I/O type

Models include:

- model type (I/O type)
- component capacitance
- power-supply voltage range
- V-I table for pull-up stage, pull-down stage, GND clamp diode, and Vcc clamp diode (whichever apply)
- rising/falling slew rate

The following elements are optional in a model:

- package R, L, and C for individual pins
- model polarity; enable polarity
- input thresholds

The next few sections describe the model elements in detail.

“Default” Package R, L, and C

The “default” package resistance, inductance, and capacitance specify a range of values for modeling the device package. Typical data is required; min and max data are optional. If only typical data is available, the package R, L, and C can be thought of as the “average” data for the device package.

BoardSim converts the R, L, and C values into an equivalent transmission line -- a good model for the IC's bond-out structure and package pins.

BoardSim provides the additional flexibility to set the package parameters to 0.0. This indicates that the data was unavailable; BoardSim will omit the package-model transmission line and simulate with the silicon data only.

Semiconductor vendors can determine R, L, and C with high-accuracy test equipment or electromagnetic modeling. End users can get package data from

the vendor, use data for a similar package, or set R, L, and C to 0.0 and omit package modeling entirely.

Note: *In most situations, package R, L, and C have only a minor effect on simulation. Don't be afraid to omit them if you have no data.*

Example of package R, L, and C:

```
[Package]
    | variable      typ          min          max
    R_pkg           250.0m       225.0m       275.0m
    L_pkg           15.0nH       12.0nH       18.0nH
    C_pkg           18.0pF       15.0pF       20.0pF
```

Note: *IBIS keywords (delimited by square brackets []) must begin in the first column of the actual .IBS file.*

Pin/Signal List

The pin/signal list is a list that associates device signal names and pin numbers with simulation models. There are simulation models for each unique kind of driver or receiver on a device. Typically, though, many signals will share a single model, e.g., all the address lines on a device might have the same driver structure.

See the IBIS specification (Appendix A) for details on length limits for names, etc.

Example of pin/signal list:

```
[Pin]          signal_name     model_name     R_pin    L_pin    C_pin
|
1       RAS0#           Buffer1
2       RAS1#           Buffer2
3       EN1#            Input1
4       A0              3-state
5       D0              I/O1
6       RD#             Input2
```

7	WR#	Input2
8	A1	I/O2
9	D1	I/O2
10	GND	GND

Note: *See "Package R, L, C for Individual Pins" below for details on R_pin, L_pin, and C_pin.*

Model Type

Within a simulation model, the model type specifies whether the signal is a driver, a receiver, or bi-directional.

If bi-directional, BoardSim allows the model's Buffer Direction to be specified as either a driver or a receiver.

Example of model type:

```
Model_type      Output
```

Note: *The IBIS specification defines two model types, 3-state and Open_drain, which are not supported by the modeling constructs in V1.1.*

Component Capacitance

The component capacitance specifies a model's silicon die capacitance. It should not include the capacitance of the device package. Typical data is required; min and max data are optional.

BoardSim allows the component capacitance to be 0.0, if no data is available.

Example of component capacitance:

variable	typ	min	max
C_comp	12.0pF	10.0pF	15.0pF

Note: *Values of 4-5 pF are common for component capacitance and would be a reasonable approximation in the absence of better data.*

Power-Supply Voltage Range

The power-supply voltage range defines the power-supply tolerance over which a model's data is valid. This should be the same range over which the model's V-I curves and slew rates were measured. All data is required, though the min and max values can be the same as typical if only a nominal voltage is supported.

BoardSim allows you to set a driver's supply voltage to a variety of common increments, but only inside the range specified.

Example of power-supply voltage range:

```
| variable          typ     min     max
[Voltage range]             5.0V    4.5V    5.5V
```

V-I Tables

The V-I tables define the V-I curves of the pull-up and pull-down structures in an output driver and the clamp diodes (if any) to Vcc and GND. A table can be omitted if the corresponding structure doesn't exist (for example, the pull-up table for an open-drain output, or the Vcc-side clamp diode for a 74F receiver.)

Typical data is required; min and max data are optional. The IBIS specification disallows more than 100 points in a table, although BoardSim doesn't bother enforcing this. Linear interpolation is used to find values that lie between points in the table. Currents for voltages outside the table are assumed equal to the last point in the table.

Note: *The last point — that currents for voltages outside the table are assumed to be equal to the last stated current precludes creating a purely resistive driver by using data points 0,0 and some other V1,I1. For voltages greater than V1, BoardSim will still use current I1. Be sure you specify V-I points all the way out to where the current begins to saturate.*

BoardSim allows changing between best-case, typical, and worst-case signal specs if max, typical, and min currents are all specified. See Chapter 12, section "What IC Operating Settings Mean" for details. BoardSim requires at least two points in a V-I table.

Voltages in pull-up and Vcc-clamp-diode tables are relative to Vcc, not ground. See the IBIS specification (Appendix A) for more details.

End users can collect V-I data in the lab by a variety of means. (Don't be afraid to use an apparatus as simple as a multimeter, power supply, and current-limiting resistor.) Since driver pull-up and pull-down structures cannot be completely isolated on a real device, some approximations must be made for the turning-off portion of each stage's curve. Don't bother with many data points in regions where a curve is fairly linear; BoardSim will automatically interpolate.

There is some risk of damaging a device while taking clamp-diode data. If you have any information regarding the diode's fully-on "resistance," construct a table that has zero current until the correct turn-on voltage, then is linear with the slope dictated by the "on" resistance.

Example of V-I table (for a driver pull-up):

```
[Pullup]
|
|       Voltage           I(typ)    I(min)    I(max)
|
        -5.0V             32.0m     30.0m     35.0m
        -4.0V             31.0m     29.0m     33.0m
|               .
|               .
         0.0V             0.0m       0.0m     0.0m
|               .
```

```
|                .
      5.0V          -32.0m  -30.0m  -35.0m
      10.0V         -38.0m  -35.0m  -40.0m
```

Slew Rates

The slew rates (or “ramp” rates) define the rise and fall times of a driver. Typical data is required; min and max data are optional. BoardSim allows changing between best-case, typical, and worst-case signal specs if max, typical, and min slew rates are all specified.

End users can collect slew-rate data in the lab with a high-bandwidth oscilloscope. To remove package-related effects, slew the driver output into an open load. The slew rates are specified in the IBIS file as a convenient ratio of the delta-V and delta-T values measured on the scope. The IBIS specification recommends measuring between the 20% and 80% points of the driver's swing.

Example of slew rates:

```
[Ramp]
| variable     typ            min            max
dV/dt_r              4.2/1.8n       3.5/2.5n       5.0/1.1n
dV/dt_f              2.5/1.5n       2.0/2.3n       3.0/0.8n
```

Package R, L, C for Individual Pins

Individual signals can have their own, unique pin R, L, and C values. These appear in the signal/pin list, and can be specified for any or all of the device's signals. If a signal has no individual data, the default package R, L, and C is used.

BoardSim allows the flexibility to specify any or all of the values for each signal. E.g., you can specify a signal-specific C, but leave R and L unspecified (as “NA”); the default R and L will be used.

See “‘Default’ Package R, L, and C” above for details on package modeling.

Example of individual package R, L, and C:

[Pin]	signal_name	model_name	R_pin	L_pin	C_pin
1	RAS0#	Buffer1	200.0m	5.0nH	2.0pF
2	RAS1#	Buffer2	209.0m	NA	2.5pF
3	EN1#	Input1	NA	6.3nH	NA
4	A0	3-state			
5	D0	I/O1			
6	RD#	Input2	310.0m	3.0nH	2.0pF

Model and Enable Polarity

The model polarity specifies whether an output signal is inverting or non-inverting. The enable polarity defines whether an enabling signal is active-high or active-low.

Both of these constructs are read by BoardSim but ignored. They are present in the IBIS specification for timing-analysis tools.

Input Thresholds

The input thresholds specify the voltages at which an input signal is recognized as a valid 1 or 0. They are used by BoardSim's Board Wizard to calculate timing delays.

App Note: Converting a SPICE Model to HyperLynx Databook Format

Introduction

Occasionally, you may have no model for a driver IC other than a SPICE model. This application note describes a relatively simple way in which you can convert a SPICE model — as long as you can run it in SPICE to extract some basic behavioral information — into a HyperLynx databook-format (i.e., ".MOD") model.

Note: *If you cannot run the steps described below (because, for example, you do not have a SPICE package capable of compiling and exercising the model), another option is to get the semiconductor vendor who created the model to run the steps for you.*

Another means of converting from SPICE models exists — a university-written SPICE-to-IBIS translator — but HyperLynx believes that for customers, the method described in this application note is considerably easier and more likely to succeed than using the converter. The converter is not a commercial product; is poorly documented; has bugs; and is not trivial to understand or run. The method described here is relatively straightforward.

SPICE Writer Option

If the conversion process described in this application note is not possible or appealing, a different way of interfacing BoardSim and SPICE models is by using HyperLynx's SPICE Writer option. The SPICE Writer converts nets in

BoardSim into detailed SPICE netlists. This enables you to use BoardSim to automatically model all the details of your PCB routing — including impedance calculations — and end up with a netlist in SPICE to which you can attach SPICE IC models. In BoardSim Crosstalk, the Writer can even generate netlists that include trace-to-trace coupling (currently supported for Hspice only).

For information on the SPICE Writer option, contact HyperLynx or your local reseller.

Requirements for Converting

In order to implement the method described in this application note, the following must be true:

- you have access to a version of SPICE that compiles and runs the model you want to convert
- you can embed the SPICE model in a simple SPICE test circuit that allows the model's output(s) to be switched high and low
- you can plot the waveform results of such a simulation so you can measure certain features of the waveforms

Access to a SPICE Simulator

Having a SPICE package and having one that compiles a particular SPICE model are sometimes different things. Some semiconductor-vendor-supplied SPICE models are developed to run in a proprietary in-house SPICE package, and may not compile in a commercial package. Others may require particular versions of Avanti's Hspice simulator. If you lack a suitable SPICE package, fax the semiconductor vendor who created the SPICE model this application note and have them generate the data for you.

SPICE Test Circuit

The SPICE test circuit required by this application note is simple: it entails adding only a few extra nodes to the model's SPICE deck. However, in order to understand where in the SPICE deck to add the new nodes, you need to know which existing nodes represent the input and output of the buffer you're modeling. In some cases, you may also need to tie certain other input nodes (e.g., an enable pin) high or low to make the buffer output respond to input stimuli.

If the SPICE model you're trying to convert is complex or poorly documented, request a schematic diagram or list of key nodes from the semiconductor vendor.

How This Method Works

The conversion method described below has a simple goal: find out enough about the behavior of the SPICE model to fill in the parameters in BoardSim's databook (.MOD) model editor. If you can fill in the required parameters, then the editor will generate a .MOD model for you and you're ready to simulate.

To open the editor in BoardSim, choose Databook IC Models from the Edit menu. (For more information on the editor and creating databook models generally, see Chapter 10.)

Many of the parameters in the editor are relatively easy to fill in: you can find them in a data sheet, or (in a few non-critical cases) even make reasonable guesses. But two important ones are harder to determine and must usually be found experimentally from the SPICE model:

- driver resistance (high and low)
- driver slew time (high and low)

Note: *If you happen to find resistance or slew time — or both — in a data sheet or by some other non-SPICE means, use them! This application note assumes that you lack both and have to resort to the SPICE model.*
If the data sheet contains I-V curves for the output buffer, you can convert the

curves to equivalent resistances by drawing a straight line through the linear part of the curve (after it "turns on" but before it shows any saturation) and measuring the slope ΔV/ΔI.

Finding Driver Resistance from a SPICE Model

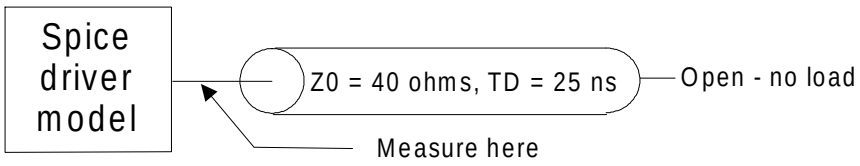
To make a SPICE model "reveal" a driving resistance, you can make use of a simple fact of electromagnetics: **when a device output drives a transmission line, the transmission line *initially* looks exactly like a resistance to ground or Vcc** (depending on whether the driver rises or falls). As soon as a reflection from the end of the transmission line travels back to the device, the line's behavior changes and becomes more complex, of course; ultimately, the line looks like an open circuit (i.e., presents no load to the driver).

But the transmission line's behavior *before any reflections occur* gives a simple and powerful method of finding a driver's resistance via simple voltage division. A transmission line's effective resistance during initial switching is equal to its characteristic impedance (Z_0). Therefore, if you load a driver with a transmission line of known impedance and measure — again, before any reflections occur — how much of the driver's voltage step actually appears in the transmission line, you can trivially solve for the driver's resistance (see below for formulas).

In practice, to isolate a driver's initial switching behavior into a transmission line from the behavior after reflections occur, you can simply use a very long transmission line. In the detailed steps below, for example, a 25-ns line is recommended. Any delay value that is longer than the entire amount of time for which you simulate in SPICE guarantees that your measurements will not be "polluted" by line reflections. Figure 1 shows the recommended transmission line and how it connects to the driver model's output.

Note that driving resistance (and slew time) must be found separately for the high and low stages of a driver, because the stages are often not balanced. (Usually, the low side has lower impedance.)

Figure 1: Connecting the Test Transmission Line to the Driver Model



Finding Driver Slew Time from a SPICE Model

Unlike driver resistance, driver slew time is not difficult to find — no "tricks" are required. Slew times can be measured directly from plots of the driver switching high and low into a transmission-line load. See below for details on making the measurements.

Finding Driver Resistance and Slew Time

To find a driver's high and low resistances and slew times:

First, add a test circuit to the SPICE model:

1. Open the driver's SPICE model in a text editor. Find the SPICE node representing the driver's output. Add a new line to the model that ties the driver output to a simple, lossless transmission line. Set the line's parameters to delay = 25 ns and characteristic impedance = 40 ohms. Leave the far end of the transmission line open.

   For example, if the driver's output node number is 10, add the following line to the SPICE model:

   T1 10 0 200 0 Z0=40 TD=25ns

   This ties one end of a transmission line to the driver output node (node 10); creates a new, open node at the other end (node 200; use any unused node number); and sets delay and impedance as described above. The "0's" reference the line to SPICE's global ground node.

2. Open the driver's SPICE model in a text editor. Find the SPICE node representing the driver's input. Add a new line to the model that ties the driver input to a ramping voltage generator. Set the generator's ramp time to 1 ns; make it switch from the driver's low power supply to the high supply.

   For example, if the driver's input node number is 20, add the following line to the SPICE model:

   ```
   V1  20  0  PWL  0ns 0V,  1ns  5V
   ```

   This ties a 0-ohm (ideal) voltage source to the driver input node (node 20), and switches the source in a linear ramp from 0V to 5V in 1 ns.

   If the driver is inverting, then compensate by reversing the polarity of the input voltage source.

3. If the driver model requires certain other input nodes to be tied high or low in order for the output to switch, tie them to Vcc or ground as needed.

Then, run a simulation with the driver switching high, and plot the driver's output node:

4. Run a transient SPICE simulation. To ensure that your measurements are not "polluted" by reflections, set the total simulation time to be greater than the driver's switching time, but less than the delay time of the transmission line you added above in step 1.

   For example, if the driver model switches in about 3 ns and you used a 25-ns transmission line as recommended above, run the transient simulation for 10 ns. This is long enough to see the driver switch completely, but not long enough for reflections to return from the line's far end.

5. Plot the simulation voltage versus time at the driver's output node. (This is the voltage at the near end of the transmission line, not the far, open end.) Set up the plot so that the switching fills most of the

horizontal scale, so you can make an accurate measurement of the switching time.

Then, from the plot, calculate the driver's high-stage resistance and slew time:

6. Calculate the driving resistance with the following formula:

   $R_{drv}(\text{rise}) = Z_0 (V_{final} / V_{step} - 1)$,

   where Z_0 = characteristic impedance of the transmission line (40 ohms),
   V_{final} = final DC voltage swing of the driver,
   V_{step} = voltage swing of the initial step into the transmission line
   (ignore signs of the voltage values)

   The "final DC swing" means how many volts the driver switches edge-to-edge after any transmission line reflections die out. For example, for a 5-V CMOS driver, this would be 5V; for a TTL device, about 3.6V.

   The "initial step" swing is the lesser number of volts that the driver swings in the simulation plot. The value is less than the final DC swing because, initially, before reflections, the voltage divides between the resistance of the driver and the resistance (characteristic impedance) of the transmission line. See Figure 2.

   (There's nothing mysterious about this formula — it's a simple voltage division, solved for the unknown driving resistance.)

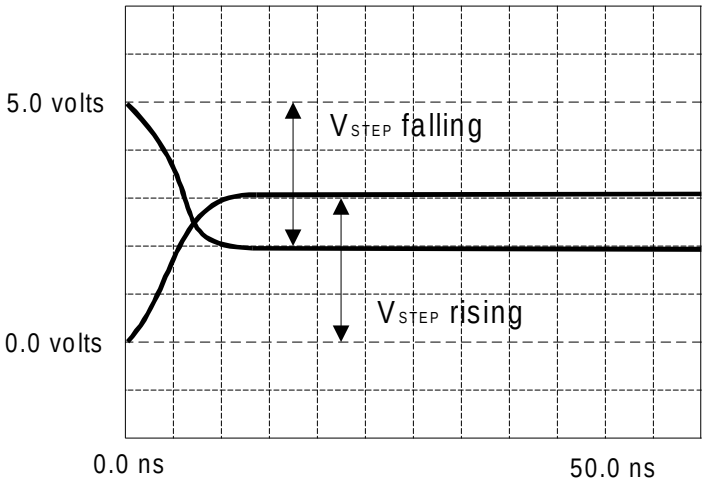
   Then, calculate the slew time by measuring the amount of time to go from 10% to 90% of the plotted waveform voltage.

Finally, re-run the simulation with the driver switching low, and use the same methods to find the low-side resistance and slew time:

7. Alter the input voltage source to switch in the opposite direction. Run another simulation and plot the driver-output node (falling edge this time). Use the same formula as above to calculate the driving

resistance; measure the 90% to 10% time as the slew time. Use absolute values of all quantities, i.e., ignore negative signs.

Figure 2: How to Measure Vstep



Finding Other Required Model Parameters

Driver resistance and slew time (high and low) are usually the only parameters that must be derived from the SPICE model. The remaining databook model values can generally be found in a data sheet, or in a few non-critical cases, can be guessed at. The following table summarizes how to find the other parameters in the .MOD model editor.

Parameter	**How to Find**
Type (high and low)	Pull down the combo box and choose the appropriate technology type. For a bipolar device, use the data sheet's circuit diagram to determine whether a given stage uses a silicon or Schottky-clamped transistor (unless ECL). Ignore "ramp"; use "open" for the high side of an open collector/drain.
Offset Voltage (high and low)	Set to 0.0V for any device that switches from supply rail to supply rail. For stages that do not switch to a rail, set equal to the amount by which the final voltage is offset

Parameter	How to Find
	from the rail, *minus one diode drop*. E.g., for 5-V TTL, high side offset is 1.4V – 0.4V (0.4 for the Schottky-diode drop). For most CMOS devices, use 0.0V, since switching is rail-to-rail.
Clamp Diodes, Type and Resistance (high and low)	Relatively unimportant for driver models. Can do a SPICE DC sweep above/below power-supply rails to find diode "resistance," but not usually worth the effort. Instead, use standard "good guess" values: Type = Silicon, Resistance = 15 ohms.
Default Power Supply	Set to Vcc for all non-ECL devices; to Vee for ECL devices.
Capacitance	Get from data sheet (C_{out} or "output capacitance"). If not provided (rare), use 5 pF.

Creating the Model

Once you've collected all the required parameters, you can actually create the databook (.MOD) model by running BoardSim's model editor; entering the parameters; and saving the model under an appropriate name in user library. For details on this process, see the appropriate section in the User's Guide or online Help system.

Once the model is saved, you're ready to simulate.

Index