Signal Integrity Analysis with Third-Party Tools

With the ever-increasing operating speed of interfaces in traditional FPGA design, the timing and signal integrity margins between the FPGA and other devices on the board must be within specification and tolerance before a single PCB is built.

If the board trace is designed poorly or the route is too heavily loaded, noise in the signal can cause data corruption, while overshoot and undershoot can potentially damage input buffers over time.

As FPGA devices are used in high-speed applications, signal integrity and timing margin between the FPGA and other devices on the printed circuit board (PCB) are important aspects to consider to ensure proper system operation. To avoid time-consuming redesigns and expensive board respins, the topology and routing of critical signals must be simulated. The high-speed interfaces available on current FPGA devices must be modeled accurately and integrated into timing models and board-level signal integrity simulations. The tools used in the design of an FPGA and its integration into a PCB must be “board-aware”—able to take into account properties of the board routing and the connected devices on the board.

The Quartus® II software provides methodologies, resources, and tools to ensure good signal integrity and timing margin between Altera® FPGA devices and other components on the board. Three types of analysis are possible with the Quartus II software:

- I/O timing with a default or user-specified capacitive load and no signal integrity analysis (default)
- The Quartus II Enable Advanced I/O Timing option utilizing a user-defined board trace model to produce enhanced timing reports from accurate “board-aware” simulation models
- Full board routing simulation in third-party tools using Altera-provided or generated Input/Output Buffer Information Specification (IBIS) or HSPICE I/O models

I/O timing using a specified capacitive test load requires no special configuration other than setting the size of the load. I/O timing reports from the Quartus II TimeQuest or the Quartus II Classic Timing Analyzer are generated based only on point-to-point delays within the I/O buffer and assume the presence of the capacitive test load with no other details about the board specified. The default size of the load is based on the I/O standard selected for the pin. Timing is measured to the FPGA pin with no signal integrity analysis details.

The Enable Advanced I/O Timing option expands the details in I/O timing reports by taking board topology and termination components into account. A complete point-to-point board trace model is defined and
accounted for in the timing analysis. This ability to define a board trace model is an example of how the Quartus II software is “board-aware.”

In this case, timing and signal integrity metrics between the I/O buffer and the defined far end load are analyzed and reported in enhanced reports generated by the Quartus II TimeQuest Timing Analyzer.

Related Information

I/O Management
For more information about defining capacitive test loads or how to use the Enable Advanced I/O Timing option to configure a board trace model.

Signal Integrity Simulations with HSPICE and IBIS Models

The Quartus II software can export accurate HSPICE models with the built-in HSPICE Writer. You can run signal integrity simulations with these complete HSPICE models in Synopsys HSPICE. IBIS models of the FPGA I/O buffers are also created easily with the Quartus II IBIS Writer.

You can run signal integrity simulations with these complete HSPICE models in Synopsys HSPICE.

You can integrate IBIS models into any third-party simulation tool that supports them, such as the Mentor Graphics® Hyperlynx software. With the ability to create industry-standard model definition files quickly, you can build accurate simulations that can provide data to help improve board-level signal integrity.

The I/O’s IBIS and HSPICE model creation available in the Quartus II software can help prevent problems before a costly board respin is required. In general, creating and running accurate simulations is difficult and time consuming. The tools in the Quartus II software automate the I/O model setup and creation process by configuring the models specifically for your design. With these tools, you can set up and run accurate simulations quickly and acquire data that helps guide your FPGA and board design.

The information about signal integrity in this chapter refers to board-level signal integrity based on I/O buffer configuration and board parameters, not simultaneous switching noise (SSN), also known as ground bounce or VCC sag. SSN is a product of multiple output drivers switching at the same time, causing an overall drop in the voltage of the chip’s power supply. This can cause temporary glitches in the specified level of ground or VCC for the device.

This chapter is intended for FPGA and board designers and includes details about the concepts and steps involved in getting designs simulated and how to adjust designs to improve board-level timing and signal integrity. Also included is information about how to create accurate models from the Quartus II software and how to use those models in simulation software.

The information in this chapter is meant for those who are familiar with the Quartus II software and basic concepts of signal integrity and the design techniques and components in good PCB design. Finally, you should know how to set up simulations and use your selected third-party simulation tool.

Related Information

- **AN 315: Guidelines for Designing High-Speed FPGA PCBs**
  For more information about SSN and ways to prevent it.

- **Altera Signal Integrity Center**
  For information about basic signal integrity concepts and signal integrity details pertaining to Altera FPGA devices.
**I/O Model Selection: IBIS or HSPICE**

The Quartus II software can export two different types of I/O models that are useful for different simulation situations, IBIS models and HSPICE models.

IBIS models define the behavior of input or output buffers through the use of voltage-current (V-I) and voltage-time (V-t) data tables. HSPICE models, often referred to as HSPICE decks, include complete physical descriptions of the transistors and parasitic capacitances that make up an I/O buffer along with all the parameter settings required to run a simulation. The HSPICE decks generated by the Quartus II software are preconfigured with the I/O standard, voltage, and pin loading settings for each pin in your design.

The choice of I/O model type is based on many factors.

<table>
<thead>
<tr>
<th>Feature</th>
<th>IBIS Model</th>
<th>HSPICE Model</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>I/O Buffer Description</strong></td>
<td><strong>Behavioral</strong>—I/O buffers are described by voltage-current and voltage-time tables in typical, minimum, and maximum supply voltage cases.</td>
<td>Physical—I/O buffers and all components in a circuit are described by their physical properties, such as transistor characteristics and parasitic capacitances, as well as their connections to one another.</td>
</tr>
<tr>
<td><strong>Model Customization</strong></td>
<td><strong>Simple and limited</strong>—The model completely describes the I/O buffer and does not usually have to be customized.</td>
<td><strong>Fully customizable</strong>—Unless connected to an arbitrary board description, the description of the board trace model must be customized in the model file. All parameters of the simulation are also adjustable.</td>
</tr>
<tr>
<td><strong>Simulation Set Up and Run Time</strong></td>
<td><strong>Fast</strong>—Simulations run quickly after set up correctly.</td>
<td><strong>Slow</strong>—Simulations take time to set up and take longer to run and complete.</td>
</tr>
<tr>
<td><strong>Simulation Accuracy</strong></td>
<td><strong>Good</strong>—For most simulations, accuracy is sufficient to make useful adjustments to the FPGA and/or board design to improve signal integrity.</td>
<td><strong>Excellent</strong>—Simulations are highly accurate, making HSPICE simulation almost a requirement for any high-speed design where signal integrity and timing margins are tight.</td>
</tr>
<tr>
<td><strong>Third-Party Tool Support</strong></td>
<td><strong>Excellent</strong>—Almost all third-party board simulation tools support IBIS.</td>
<td><strong>Good</strong>—Most third-party tools that support SPICE support HSPICE. However, Synopsys HSPICE is required for simulations of Altera’s encrypted HSPICE models.</td>
</tr>
</tbody>
</table>

**Related Information**

**AN 283: Simulating Altera Devices with IBIS Models**

For more information about IBIS files created by the Quartus II IBIS Writer and IBIS files in general, as well as links to websites with detailed information.

**FPGA to Board Signal Integrity Analysis Flow**

Board signal integrity analysis can take place at any point in the FPGA design process and is often performed before and after board layout. If it is performed early in the process as part of a pre-PCB layout analysis, the models used for simulations can be more generic.
These models can be changed as much as required to see how adjustments improve timing or signal integrity and help with the design and routing of the PCB. Simulations and the resulting changes made at this stage allow you to analyze “what if” scenarios to plan and implement your design better. To assist with early board signal integrity analysis, you can download generic IBIS model files for each device family and obtain HSPICE buffer simulation kits from the “Board Level Tools” section of the EDA Tool Support Resource Center.

Typically, if board signal integrity analysis is performed late in the design, it is used for a post-layout verification. The inputs and outputs of the FPGA are defined, and required board routing topologies and constraints are known. Simulations can help you find problems that might still exist in the FPGA or board design before fabrication and assembly. In either case, a simple process flow illustrates how to create accurate IBIS and HSPICE models from a design in the Quartus II software and transfer them to third-party simulation tools.

Your design depends on the type of model, IBIS or HSPICE, that you use for your simulations. When you understand the steps in the analysis flow, refer to the section of this chapter that corresponds to the model type you are using.
Figure 6-1: Third-Party Board Signal Integrity Analysis Flow

Create a Quartus II Project

- Make I/O Assignments
- Configure Board Trace Models in supported devices (Optional)
- Enable IBIS or HSPICE File Generation
- Compile and Generate Files (EDA Netlist Writer)
- Customize Files

IBIS

- Apply Models to Buffers in Board Model Simulations
- Run Simulation

Results OK?

Yes

- IBIS or HSPICE?

Run Simulations as Defined in HSPICE Deck

No

- HSPICE

Make Adjustments to Models or Simulation Parameters and Simulate Again

Changes to FPGA I/O required?

Yes

- IBIS

Changes to FPGA I/O required?

No

Related Information

EDA Tool Support Resource Center

For more information, generic IBIS model files for each device family, and to obtain HSPICE buffer simulation kits.

Create I/O and Board Trace Model Assignments

You can configure a board trace model for output signals or for bidirectional signals in output mode. You can then automatically transfer its description to HSPICE decks generated by the HSPICE Writer. This helps improve simulation accuracy.
To configure a board trace model, in the Settings dialog box, in the TimeQuest Timing Analyzer page, turn on the Enable Advanced I/O Timing option and configure the board trace model assignment settings for each I/O standard used in your design. You can add series or parallel termination, specify the transmission line length, and set the value of the far-end capacitive load. You can configure these parameters either in the Board Trace Model view of the Pin Planner, or click Settings Device Device and Pin Options.

The Quartus II software can generate IBIS models and HSPICE decks without having to configure a board trace model with the Enable Advanced I/O Timing option. In fact, IBIS models ignore any board trace model settings other than the far-end capacitive load. If any load value is set other than the default, the delay given by IBIS models generated by the IBIS Writer cannot be used to account correctly for the double counting problem. The load value mismatch between the IBIS delay and the tCO measurement of the Quartus II software prevents the delays from being safely added together. Warning messages displayed when the EDA Netlist Writer runs indicate when this mismatch occurs.

Related Information
I/O Management
For information about how to use the Enable Advanced I/O Timing option and configure board trace models for the I/O standards used in your design.

Output File Generation
IBIS and HSPICE model files are not generated by the Quartus II software by default. To generate or update the files automatically during each project compilation, select the type of file to generate and a location where to save the file in the project settings.

The IBIS and HSPICE Writers in the Quartus II software are run as part of the EDA Netlist Writer during normal project compilation. If either writer is turned on in the project settings, IBIS or HSPICE files are created and stored in the specified location. For IBIS, a single file is generated containing information about all assigned pins. HSPICE file generation creates separate files for each assigned pin. You can run the EDA Netlist Writer separately from a full compilation in the Quartus II software or at the command line. However, you must fully compile the project or perform I/O Assignment Analysis at least once for the IBIS and HSPICE Writers to have information about the I/O assignments and settings in the design.

Note: These settings can also be specified with commands in a Tcl script.

Customize the Output Files
The files generated by either the IBIS or HSPICE Writer are text files that you can edit and customize easily for design or experimentation purposes.

IBIS files downloaded from the Altera website must be customized with the correct RLC values for the specific device package you have selected for your design. IBIS files generated by the IBIS Writer do not require this customization because they are configured automatically with the RLC values for your selected device. HSPICE decks require modification to include a detailed description of your board. With Enable Advanced I/O Timing turned on and a board trace model defined in the Quartus II software, generated HSPICE decks automatically include that model’s parameters. However, Altera recommends that you replace that model with a more detailed model that describes your board design more accurately. A default simulation included in the generated HSPICE decks measures delay between the FPGA and the far-end device. You can make additions or adjustments to the default simulation in the generated files to change the parameters of the default simulation or to perform additional measurements.

Set Up and Run Simulations in Third-Party Tools
When you have generated the files, you can use them to perform simulations in your selected simulation tool.
With IBIS models, you can apply them to input, output, or bidirectional buffer entities and quickly set up and run simulations. For HSPICE decks, the simulation parameters are included in the files. Open the files in Synopsys HSPICE and run simulations for each pin as required.

With HSPICE decks generated from the HSPICE Writer, the double counting problem is accounted for, which ensures that your simulations are accurate. Simulations that involve IBIS models created with anything other than the default loading settings in the Quartus II software must take the change in the size of the load between the IBIS delay and the Quartus II t_{CO} measurement into account. Warning messages during compilation alert you to this change.

**Interpret Simulation Results**

If you encounter timing or signal integrity issues with your high-speed signals after running simulations, you can make adjustments to I/O assignment settings in the Quartus II software.

These could include such things as drive strength or I/O standard, or making changes to your board routing or topology. After regenerating models in the Quartus II software based on the changes you have made, rerun the simulations to check whether your changes corrected the problem.

**Simulation with IBIS Models**

IBIS models provide a way to run accurate signal integrity simulations quickly. IBIS models describe the behavior of I/O buffers with voltage-current and voltage-time data curves.

Because of their behavioral nature, IBIS models do not have to include any information about the internal circuit design of the I/O buffer. Most component manufacturers, including Altera, provide IBIS models for free download and use in signal integrity analysis simulation tools. You can download generic device family IBIS models from the Altera website for early design simulation or use the IBIS Writer to create custom IBIS models for your existing design.

**Elements of an IBIS Model**

An IBIS model file (.ibs) is a text file that describes the behavior of an I/O buffer across minimum, typical, and maximum temperature and voltage ranges with a specified test load.

The tables and values specified in the IBIS file describe five basic elements of the I/O buffer.

**Figure 6-2: Five Basic Elements of an I/O Buffer in IBIS Models**

The following elements correspond to each numbered block.
1. **Pulldown**—A voltage-current table describes the current when the buffer is driven low based on a pull-down voltage range of \(-V_{CC}\) to 2 \(V_{CC}\).

2. **Pullup**—A voltage-current table describes the current when the buffer is driven high based on a pull-up voltage range of \(-V_{CC}\) to \(V_{CC}\).

3. **Ground and Power Clamps**—Voltage-current tables describe the current when clamping diodes for electrostatic discharge (ESD) are present. The ground clamp voltage range is \(-V_{CC}\) to \(V_{CC}\), and the power clamp voltage range is \(-V_{CC}\) to ground.

4. **Ramp and Rising/Falling Waveform**—A voltage-time (dv/dt) ratio describes the rise and fall time of the buffer during a logic transition. Optional rising and falling waveform tables can be added to more accurately describe the characteristics of the rising and falling transitions.

5. **Total Output Capacitance and Package RLC**—The total output capacitance includes the parasitic capacitances of the output pad, clamp diodes (if present), and input transistors. The package RLC is device package-specific and defines the resistance, inductance, and capacitance of the bond wire and pin of the I/O.

**Related Information**

**AN 283: Simulating Altera Devices with IBIS Models**

For more information about IBIS models and Altera-specific features, including links to the official IBIS specification.

**Creating Accurate IBIS Models**

There are two methods to obtain Altera device IBIS files for your board-level signal integrity simulations. You can download generic IBIS models from the Altera website. You can also use the IBIS writer in the Quartus II software to create design-specific models.

The IBIS file generated by the Quartus II software contains models of both input and output termination, and is supported for IBIS model versions of 4.2 and later. Arria V, Cyclone V, and Stratix V device families allow the use of bidirectional I/O with dynamic on-chip termination (OCT).

Dynamic OCT is used where a signal uses a series on-chip termination during output operation and a parallel on-chip termination during input operation. Typically this is used in Altera External Memory Interface IP.

The Quartus II IBIS dynamic OCT IBIS model names end in \(g50c\_r50c\). For example: \(sstl15i\_ctnio\_g50c\_r50c\).

In the simulation tool, the IBIS model is attached to a buffer.

- When the buffer is assigned as an output, use the series termination \(r50c\).
- When the buffer is assigned as an input, use the parallel termination \(g50c\).

**Download IBIS Models**

Altera provides IBIS models for almost all FPGA and FPGA configuration devices. You can use the IBIS models from the website to perform early simulations of the I/O buffers you expect to use in your design as part of a pre-layout analysis.

Downloaded IBIS models have the RLC package values set to one particular device in each device family.

The \(.ibs\) file can be customized for your device package and can be used for any simulation. IBIS models downloaded and used for simulations in this manner are generic. They describe only a certain set of models listed for each device on the Altera IBIS Models page of the Altera website. To create customized models for your design, use the IBIS Writer as described in the next section.
To simulate your design with the model accurately, you must adjust the RLC values in the IBIS model file to match the values for your particular device package by performing the following steps:

1. Download and expand the ZIP file (.zip) of the IBIS model for the device family you are using for your design. The .zip file contains the .ibs file along with an IBIS model user guide and a model data correlation report.
2. Download the Package RLC Values spreadsheet for the same device family.
3. Open the spreadsheet and locate the row that describes the device package used in your design.
4. From the package’s I/O row, copy the minimum, maximum, and typical values of resistance, inductance, and capacitance for your device package.
5. Open the .ibs file in a text editor and locate the [Package] section of the file.
6. Overwrite the listed values copied with the values from the spreadsheet and save the file.

Related Information

Altera IBIS Models
For information about whether models for your selected device are available.

Generate Custom IBIS Models with the IBIS Writer
If you have started your FPGA design and have created custom I/O assignments, you can use the Quartus II IBIS Writer to create custom IBIS models to accurately reflect your assignments.

Examples of custom assignments include drive strength settings or the enabling of clamping diodes for ESD protection. IBIS models created with the IBIS Writer take I/O assignment settings into account.

If the Enable Advanced I/O Timing option is turned off, the generated .ibs files are based on the load value setting for each I/O standard on the Capacitive Loading page of the Device and Pin Options dialog box in the Device dialog box. With the Enable Advanced I/O Timing option turned on, IBIS models use an effective capacitive load based on settings found in the board trace model on the Board Trace Model page in the Device and Pin Options dialog box or the Board Trace Model view in the Pin Planner. The effective capacitive load is based on the sum of the Near capacitance, Transmission line distributed capacitance, and the Far capacitance settings in the board trace model. Resistance values and transmission line inductance values are ignored.

Note: If you made any changes from the default load settings, the delay in the generated IBIS model cannot safely be added to the Quartus II $t_{CO}$ measurement to account for the double counting problem. This is because the load values between the two delay measurements do not match. When this happens, the Quartus II software displays warning messages when the EDA Netlist Writer runs to remind you about the load value mismatch.

Related Information

- Generating IBIS Output Files with the Quartus II Software
  For step-by-step instructions on how to generate IBIS models with the Quartus II software, refer to Quartus II Help.

- AN 283: Simulating Altera Devices with IBIS Models
  For more information about IBIS model generation.

Design Simulation Using the Mentor Graphics HyperLynx® Software
You must integrate IBIS models downloaded from the Altera website or created with the Quartus II IBIS Writer into board design simulations to accurately model timing and signal integrity.
The HyperLynx software from Mentor Graphics is one of the most popular tools for design simulation. The HyperLynx software makes it easy to integrate IBIS models into simulations.

The HyperLynx software is a PCB analysis and simulation tool for high-speed designs, consisting of two products, LineSim and BoardSim. LineSim is an early simulation tool. Before any board routing takes place, LineSim is used to simulate “what if” scenarios to assist in creating routing rules and defining board parameters. BoardSim is a post-layout tool used to analyze existing board routing. Specific nets are selected from a board layout file and simulated in a manner similar to LineSim. With board and routing parameters, and surrounding signal routing known, highly accurate simulations of the final fabricated PCB are possible. This section focuses on LineSim. Because the process of creating and running simulations is very similar for both LineSim and BoardSim, the details of IBIS model use in LineSim applies to simulations in BoardSim.

Simulations in LineSim are configured using a schematic GUI to create connections and topologies between I/O buffers, route trace segments, and termination components. LineSim provides two methods for creating routing schematics: cell-based and free-form. Cell-based schematics are based on fixed cells consisting of typical placements of buffers, trace impedances, and components. Parts of the grid-based cells are filled with the desired objects to create the topology. A topology in a cell-based schematic is limited by the available connections within and between the cells.

A more robust and expandable way to create a circuit schematic for simulation is to use the free-form schematic format in LineSim. The free-form schematic format makes it easy to place parts into any configuration and edit them as required. This section describes the use of IBIS models with free-form schematics, but the process is nearly identical for cell-based schematics.

Figure 6-3: HyperLynx LineSim Free-Form Schematic Editor

When you use HyperLynx software to perform simulations, you typically perform the following steps:

1. Create a new LineSim free-form schematic document and set up the board stackup for your PCB using the Stackup Editor. In this editor, specify board layer properties including layer thickness, dielectric constant, and trace width.
2. Create a circuit schematic for the net you want to simulate. The schematic represents all the parts of the routed net including source and destination I/O buffers, termination components, transmission line segments, and representations of impedance discontinuities such as vias or connectors.

3. Assign IBIS models to the source and destination I/O buffers to represent their behavior during operation.

4. Attach probes from the digital oscilloscope that is built in to LineSim to points in the circuit that you want to monitor during simulation. Typically, at least one probe is attached to the pin of a destination I/O buffer. For differential signals, you can attach a differential probe to both the positive and negative pins at the destination.

5. Configure and run the simulation. You can simulate a rising or falling edge and test the circuit under different drive strength conditions.

6. Interpret the results and make adjustments. Based on the waveforms captured in the digital oscilloscope, you can adjust anything in the circuit schematic to correct any signal integrity issues, such as overshoot or ringing. If necessary, you can make I/O assignment changes in the Quartus II software, regenerate the IBIS file with the IBIS Writer, and apply the updated IBIS model to the buffers in your HyperLynx software schematic.

7. Repeat the simulations and circuit adjustments until you are satisfied with the results. When the operation of the net meets your design requirements, implement changes to your I/O assignments in the Quartus II software and/or adjust your board routing constraints, component values, and placement to match the simulation.

Related Information

www.mentor.com

For more information about HyperLynx software, including schematic creation, simulation setup, model usage, product support, licensing, and training.

Configuring LineSim to Use Altera IBIS Models

You must configure LineSim to find and use the downloaded or generated IBIS models for your design. To do this, add the location of your .ibs file or files to the LineSim Model Library search path. Then you apply a selected model to a buffer in your schematic.

To add the Quartus II software’s default IBIS model location, <project directory>/board/ibis, to the HyperLynx LineSim model library search path, perform the following steps in LineSim:

1. From the Options menu, click Directories. The Set Directories dialog box appears. The Model-library file path(s) list displays the order in which LineSim searches file directories for model files.
2. Click **Edit**. A dialog box appears where you can add directories and adjust the order in which LineSim searches them.

**Figure 6-5: LineSim Select Directories Dialog Box**

3. Click **Add**
4. Browse to the default IBIS model location, `<project directory>/board/ibis`. Click **OK**.
5. Click **Up** to move the IBIS model directory to the top of the list. Click **Generate Model Index** to update LineSim’s model database with the models found in the added directory.
6. Click **OK**. The IBIS model directory for your project is added to the top of the Model-library file path(s) list.
7. To close the **Set Directories** dialog box, click **OK**.
Integrating Altera IBIS Models into LineSim Simulations

When the location for IBIS files has been set, you can assign the downloaded or generated IBIS models to the buffers in your schematic. To do this, perform the following steps:

1. Double-click a buffer symbol in your schematic to open the Assign Models dialog box. You can also click Assign Models from the buffer symbol’s right-click menu.

   **Figure 6-6: LineSim Assign Model Dialog Box**

2. The pin of the buffer symbol you selected should be highlighted in the Pins list. If you want to assign a model to a different symbol or pin, select it from the list.

3. Click Select. The Select IC Model dialog box appears.
4. To filter the list of available libraries to display only IBIS models, select .IBS. Scroll through the Libraries list, and click the name of the library for your design. By default, this is <project name>.ibs.

5. The device for your design should be selected as the only item in the Devices list. If not, select your device from the list.

6. From the Signal list, select the name of the signal you want to simulate. You can also choose to select by device pin number.

7. Click OK. The Assign Models dialog box displays the selected .ibs file and signal.

8. If applicable to the signal you chose, adjust the buffer settings as required for the simulation.

9. Select and configure other buffer pins from the Pins list in the same manner.

10. Click OK when all I/O models are assigned.

Running and Interpreting LineSim Simulations

You can now run any desired simulations and make adjustments to the I/O assignments or simulation parameters as required.

For example, if you see too much overshoot in the simulated signal at the destination buffer after running a simulation, you could adjust the drive strength I/O assignment setting to a lower value. Regenerate the .ibs file, and run the simulation again to verify whether the change fixed the problem.
If you see a discontinuity or other anomalies at the destination, such as slow rise and fall times, adjust the termination scheme or termination component values. After making these changes, rerun the simulation to check whether your adjustments solved the problem. In this case, it is not necessary to regenerate the .ibs file.

Related Information

Altera Signal Integrity Center

For more information about board-level signal integrity and to learn about ways to improve it with simple changes to your design.
Simulation with HSPICE Models

HSPICE decks are used to perform highly accurate simulations by describing the physical properties of all aspects of a circuit precisely. HSPICE decks describe I/O buffers, board components, and all of the connections between them, as well as defining the parameters of the simulation to be run.

By their nature, HSPICE decks are highly customizable and require a detailed description of the circuit under simulation. For devices that support advanced I/O timing, when Enable Advanced I/O Timing is turned on, the HSPICE decks generated by the Quartus II HSPICE Writer automatically include board components and topology defined in the Board Trace Model. Configure the board components and topology in the Pin Planner or in the Board Trace Model tab of the Device and Pin Options dialog box. All HSPICE decks generated by the Quartus II software include compensation for the double count problem. You can simulate with the default simulation parameters built in to the generated HSPICE decks or make adjustments to customize your simulation.

Related Information
The Double Counting Problem in HSPICE Simulations on page 6-17

Supported Devices and Signaling

The HSPICE Writer in the Quartus II software supports Arria, Cylcone, and Stratix devices for the creation of a board trace model in the Quartus II software for automatic inclusion in an HSPICE deck.

The HSPICE files include the board trace description you create in the Board Trace Model view in the Pin Planner or the Board Trace Model tab in the Device and Pin Options dialog box.

Note: Note that for Arria 10 devices, you may need to download the Encrypted HSPICE model from the Altera website.

Related Information

- **I/O Management**
  For more information about the Enable Advanced I/O Timing option and configuring board trace models for the I/O standards in your design.

- **SPICE Models for Altera Devices**
  For more information about the Encrypted HSPICE model.

Accessing HSPICE Simulation Kits

You can access the available HSPICE models with the Quartus II software’s HSPICE Writer tool and also at the Spice Models for Altera Devices web page.

The Quartus II software HSPICE Writer tool removes many common sources of user error from the I/O simulation process. The HSPICE Writer tool automatically creates preconfigured I/O simulation spice decks that only require the addition of a user board model. All the difficult tasks required to configure the I/O modes and interpret the timing results are handled automatically by the HSPICE Writer tool.

Related Information

**Spice Models for Altera Devices**
For more information about downloadable HSPICE models.
The Double Counting Problem in HSPICE Simulations

Simulating I/Os using accurate models is extremely helpful for finding and fixing FPGA I/O timing and board signal integrity issues before any boards are built. However, the usefulness of such simulations is directly related to the accuracy of the models used and whether the simulations are set up and performed correctly. To ensure accuracy in models and simulations created for FPGA output signals, the timing hand-off between \( t_{CO} \) timing in the Quartus II software and simulation-based board delay must be taken into account. If this hand-off is not handled correctly, the calculated delay could either count some of the delay twice or even miss counting some of the delay entirely.

Defining the Double Counting Problem

The double counting problem is inherent to the method output timing is analyzed versus the method used for HSPICE models. The timing analyzer tools in the Quartus II software measure delay timing for an output signal from the core logic of the FPGA design through the output buffer ending at the FPGA pin with a default capacitive load or a specified value for the selected I/O standard. This measurement is the \( t_{CO} \) timing variable.

Figure 6-10: Double Counting Problem

HSPICE models for board simulation measure \( t_{PD} \) (propagation delay) from an arbitrary reference point in the output buffer, through the device pin, out along the board routing, and ending at the signal destination.

It is apparent immediately that if these two delays were simply added together, the delay between the output buffer and the device pin would be counted twice in the calculation. A model or simulation that does not account for this double count would create overly pessimistic simulation results, because the double-counted delay can limit I/O performance artificially. To fix the problem, it might seem that simply subtracting the overlap between \( t_{CO} \) and \( t_{PD} \) would account for the double count. However, this adjustment would not be accurate because each measurement is based on a different load.

Note: Input signals do not exhibit this problem because the HSPICE models for inputs stop at the FPGA pin instead of at the input buffer. In this case, simply adding the delays together produces an accurate measurement of delay timing.
The Solution to Double Counting

To adjust the measurements to account for the double-counting, the delay between the arbitrary point in the output buffer selected by the HSPICE model and the FPGA pin must be subtracted from either $t_{CO}$ or $t_{PD}$ before adding the results together. The subtracted delay must also be based on a common load between the two measurements. This is done by repeating the HSPICE model measurement, but with the same load used by the Quartus II software for the $t_{CO}$ measurement.

**Figure 6-11: Common Test Loads Used for Output Timing**

With $t_{TESTLOAD}$ known, the total delay is calculated for the output signal from the FPGA logic to the signal destination on the board, accounting for the double count.

$$t_{delay} = t_{CO} + (t_{PD} - t_{TESTLOAD})$$

The preconfigured simulation files generated by the HSPICE Writer in the Quartus II software are designed to account for the double-counting problem based on this calculation automatically. Performing accurate timing simulations is easy without having to make adjustments for double counting manually.

**HSPICE Writer Tool Flow**

This section includes information to help you get started using the Quartus II software HSPICE Writer tool. The information in this section assumes you have a basic knowledge of the standard Quartus II software design flow, such as project and assignment creation, compilation, and timing analysis.

**Related Information**

**Quartus II Handbook**

For additional information about standard design flows.
Applying I/O Assignments

The first step in the HSPICE Writer tool flow is to configure the I/O standards and modes for each of the pins in your design properly. In the Quartus II software, these settings are represented by assignments that map I/O settings, such as pin selection, and I/O standard and drive strength, to corresponding signals in your design.

The Quartus II software provides multiple methods for creating these assignments:

- Using the Pin Planner
- Using the assignment editor
- Manually editing the .qsf file
- By making assignments in a scripted Quartus II flow using Tcl

Enabling HSPICE Writer

You must enable the HSPICE Writer in the **Settings** dialog box of the Quartus II software to generate the HSPICE decks from the Quartus II software.

**Figure 6-12: EDA Tool Settings: Board Level Options Dialog Box**

Enabling HSPICE Writer Using Assignments

You can also use HSPICE Writer in conjunction with a scripted Tcl flow. To enable HSPICE Writer during a full compile, include the following lines in your Tcl script.

**Enable HSPICE Writer**

```
set_global_assignment -name EDA_BOARD_DESIGN_SIGNAL_INTEGRITY_TOOL "HSPICE (Signal Integrity)"
set_global_assignment -name EDA_OUTPUT_DATA_FORMAT HSPICE
-set_section_id eda_board_design_signal_integrity
set_global_assignment -name EDA_NETLIST_WRITER_OUTPUT_DIR <output_directory>
-set_section_id eda_board_design_signal_integrity
```
As with command-line invocation, specifying the output directory is optional. If not specified, the output directory defaults to **board/hspice**.

### Naming Conventions for HSPICE Files

HSPICE Writer automatically generates simulation files and names them using the following naming convention: `<device>_<pin #>_<pin_name>_<in/out>.sp`.

For bidirectional pins, two spice decks are produced; one with the I/O buffer configured as an input, and the other with the I/O buffer configured as an output.

The Quartus II software supports alphanumeric pin names that contain the underscore (\_) and dash (-) characters. Any illegal characters used in file names are converted automatically to underscores.

### Related Information

- Sample Output for I/O HSPICE Simulation Deck on page 6-30
- Sample Input for I/O HSPICE Simulation Deck on page 6-26

### Invoking HSPICE Writer

After HSPICE Writer is enabled, the HSPICE simulation files are generated automatically each time the project is completely compiled. The Quartus II software also provides an option to generate a new set of simulation files without having to recompile manually. In the Processing menu, click **Start EDA Netlist Writer** to generate new simulation files automatically.

**Note:** You must perform both Analysis & Synthesis and Fitting on a design before invoking the HSPICE Writer tool.

### Invoking HSPICE Writer from the Command Line

If you use a script-based flow to compile your project, you can create HSPICE model files by including the following commands in your Tcl script (.tcl file).

#### Create HSPICE Model Files

```tcl
set_global_assignment -name EDA_BOARD_DESIGN_SIGNAL_INTEGRITY_TOOL "HSPICE (Signal Integrity)"
set_global_assignment -name EDA_OUTPUT_DATA_FORMAT HSPICE
set_global_assignment -name EDA_NETLIST_WRITER_OUTPUT_DIR <output_directory>
```

The `<output_directory>` option specifies the location where HSPICE model files are saved. By default, the `<project directory>/board/hspice` directory is used.

#### Invoke HSPICE Writer

To invoke the HSPICE Writer tool through the command line, type:

```bash
quartus_eda.exe <project_name> --board_signal_integrity=on --format=HSPICE --output_directory=<output_directory>
```

`<output_directory>` specifies the location where the generated spice decks will be written (relative to the design directory). This is an optional parameter and defaults to **board/hspice**.
Customizing Automatically Generated HSPICE Decks

HSPICE models generated by the HSPICE Writer can be used for simulation as generated.

A default board description is included, and a default simulation is set up to measure rise and fall delays for both input and output simulations, which compensates for the double counting problem. However, Altera recommends that you customize the board description to more accurately represent your routing and termination scheme.

The sample board trace loading in the generated HSPICE model files must be replaced by your actual trace model before you can run a correct simulation. To do this, open the generated HSPICE model files for all pins you want to simulate and locate the following section.

Sample Board Trace Section

* I/O Board Trace and Termination Description
  * Replace this with your board trace and termination description

You must replace the example load with a load that matches the design of your PCB board. This includes a trace model, termination resistors, and, for output simulations, a receiver model. The spice circuit node that represents the pin of the FPGA package is called $\text{pin}$. The node that represents the far pin of the external device is called $\text{load-in}$ (for output SPICE decks) and $\text{source-in}$ (for input SPICE decks).

For an input simulation, you must also modify the stimulus portion of the spice file. The section of the file that must be modified is indicated in the following comment block.

Sample Source Stimulus Section

* Sample source stimulus placeholder
  * Replace this with your I/O driver model

Replace the sample stimulus model with a model for the device that will drive the FPGA.

Running an HSPICE Simulation

Because simulation parameters are configured directly in the HSPICE model files, running a simulation requires only that you open an HSPICE file in the HSPICE user interface and start the simulation.

Figure 6-13: HSPICE User Interface Window

Click Open and browse to the location of the HSPICE model files generated by the Quartus II HSPICE Writer. The default location for HSPICE model files is $\langle$project directory$\rangle$/board/hspice. Select the .sp file generated by the HSPICE Writer for the signal you want to simulate. Click OK.
To run the simulation, click **Simulate**. The status of the simulation is displayed in the window and saved in an .lis file with the same name as the .sp file when the simulation is complete. Check the .lis file if an error occurs during the simulation requiring a change in the .sp file to fix.

### Interpreting the Results of an Output Simulation

By default, the automatically generated output simulation spice decks are set up to measure three delays for both rising and falling transitions. Two of the measurements, \( t_{pd\_rise} \) and \( t_{pd\_fall} \), measure the double-counting corrected delay from the FPGA pin to the load pin. To determine the complete clock-edge to load-pin delay, add these numbers to the Quartus II software reported default loading \( t_{CO} \) delay.

The remaining four measurements, \( t_{pd\_uncomp\_rise} \), \( t_{pd\_uncomp\_fall} \), \( t_{dblcnt\_rise} \), and \( t_{dblcnt\_fall} \), are required for the double-counting compensation process and are not required for further timing usage.

**Related Information**

- **Simulation Analysis** on page 6-30

### Interpreting the Results of an Input Simulation

By default, the automatically generated input simulation SPICE decks are set up to measure delays from the source’s driver pin to the FPGA’s input pin for both rising and falling transitions.

The propagation delay is reported by HSPICE measure statements as \( t_{pd\_rise} \) and \( t_{pd\_fall} \). To determine the complete source driver pin-to-FPGA register delay, add these numbers to the Quartus II software reported \( T_H \) and \( T_{SU} \) input timing numbers.

### Viewing and Interpreting Tabular Simulation Results

The .lis file stores the collected simulation data in tabular form. The default simulation configured by the HSPICE Writer produces delay measurements for rising and falling transitions on both input and output simulations.

These measurements are found in the .lis file and named \( t_{pd\_rise} \) and \( t_{pd\_fall} \). For output simulations, these values are already adjusted for the double count. To determine the complete delay from the FPGA logic to the load pin, add either of these measurements to the Quartus II \( t_{CO} \) delay. For input simulations, add either of these measurements to the Quartus II \( T_H \) and \( T_{SU} \) delay values to calculate the complete delay from the far end stimulus to the FPGA logic. Other values found in the .lis file, such as \( t_{pd\_uncomp\_rise} \), \( t_{pd\_uncomp\_fall} \), \( t_{dblcnt\_rise} \), and \( t_{dblcnt\_fall} \), are parts of the double count compensation calculation. These values are not necessary for further analysis.

### Viewing Graphical Simulation Results

You can view the results of the simulation quickly as a graphical waveform display using the AvanWaves viewer included with HSPICE. With the default simulation configured by the HSPICE Writer, you can view the simulated waveforms at both the source and destination in input and output simulations.

To see the waveforms for the simulation, in the HSPICE user interface window, click **AvanWaves**. The AvanWaves viewer opens and displays the **Results Browser**.
The **Results Browser** lets you select which waveform to view quickly in the main viewing window. If multiple simulations are run on the same signal, the list at the top of the **Results Browser** displays the results of each simulation. Click the simulation description to select which simulation to view. By default, the descriptions are derived from the first line of the HSPICE file, so the description might appear as a line of asterisks.

Select the type of waveform to view, by performing the following steps:

1. To see the source and destination waveforms with the default simulation, from the **Types** list, select **Voltages**.
2. On the **Curves** list, double-click the waveform you want to view. The waveform appears in the main viewing window.

You can zoom in and out and adjust the view as desired.
Making Design Adjustments Based on HSPICE Simulations

Based on the results of your simulations, you can make adjustments to the I/O assignments or simulation parameters if required. For example, after you run a simulation and see overshoot or ringing in the simulated signal at the destination buffer, you can adjust the drive strength I/O assignment setting to a lower value. Regenerate the HSPICE deck, and run the simulation again to verify that the change fixed the problem.
If there is a discontinuity or any other anomalies at the destination, adjust the board description in the Quartus II Board Trace Model, or in the generated HSPICE model files to change the termination scheme or adjust termination component values. After making these changes, regenerate the HSPICE files if necessary, and rerun the simulation to verify whether your adjustments solved the problem.
Figure 6-17: Example of Signal Integrity Anomaly in the AvanWaves Waveform Viewer

Related Information

Altera Signal Integrity Center
For more information about board-level signal integrity and to learn about ways to improve it with simple changes to your FPGA design.

Sample Input for I/O HSPICE Simulation Deck
The following sections examine a typical HSPICE simulation spice deck for an I/O of type input. Each section presents the simulation file one block at a time.

Header Comment
The first block of an input simulation spice deck is the header comment. The purpose of this block is to provide an easily readable summary of how the simulation file has been automatically configured by the Quartus II software.

This block has two main components: The first component summarizes the I/O configuration relevant information such as device, speed grade, and so on. The second component specifies the exact test condition that the Quartus II software assumes for the given I/O standard.

Sample Header Comment Block

* Quartus II HSPICE Writer I/O Simulation Deck*

* This spice simulation deck was automatically generated by Quartus for the following IO settings:
Simulation Conditions

The simulation conditions block loads the appropriate process corner models for the transistors. This condition is automatically set up for the slow timing corner and is modified only if other simulation corners are desired.

Simulation Conditions Block

* Process Settings
  .options brief
  .inc 'sii_tt.inc' * TT process corner

Simulation Options

The simulation options block configures the simulation temperature and configures HSPICE with typical simulation options.

Simulation Options Block

* Simulation Options
  .options brief=0
  .options badchr co=132 scale=1e-6 acct ingold=2 nomod dv=1.0
  +  dcstep=1 absv=1e-3 absi=1e-8 probe csdf=2 accurate=1
+  converge=1
  .temp 85
Note: For a detailed description of these options, consult your HSPICE manual.

Constant Definition

The constant definition block of the simulation file instantiates the voltage sources that controls the configuration modes of the I/O buffer.

Constant Definition Block

* Constant Definition

voeb       oeb       0     vc   * Set to 0 to enable buffer output
vopdrain   opdrain   0     0    * Set to vc to enable open drain
vrambh     rambh     0     0    * Set to vc to enable bus hold
vrvpullup   rpullup   0     0    * Set to vc to enable weak pullup
vpcdp5      rpcdp5    0     rp5  * Set the IO standard
vpcdp4      rpcdp4    0     rp4
vpcdp3      rpcdp3    0     rp3
vpcdp2      rpcdp2    0     rp2
vpcdp1      rpcdp1    0     rp1
vpcdp0      rpcdp0    0     rp0
vpcdn4      rpcdn4    0     rn4
vpcdn3      rpcdn3    0     rn3
vpcdn2      rpcdn2    0     rn2
vpcdn1      rpcdn1    0     rn1
vpcdn0      rpcdn0    0     rn0
vdin din    0     0

Where:

• Voltage source voeb controls the output enable of the buffer and is set to disabled for inputs.
• vopdrain controls the open drain mode for the I/O.
• vrambh controls the bus hold circuitry in the I/O.
• vrvpullup controls the weak pullup.
• The next 11 voltages sources control the I/O standard of the buffer and are configured through a later library call.
• vdin is not used on input pins because it is the data pin for the output buffer.

Buffer Netlist

The buffer netlist block of the simulation spice deck loads all the load models required for the corresponding input pin.

Buffer Netlist Block

* IO Buffer Netlist
.include 'vio_buffer.inc'

Drive Strength

The drive strength block of the simulation SPICE deck loads the configuration bits necessary to configure the I/O into the proper I/O standard and drive strengths.

Although these settings are not relevant to an input buffer, they are provided to allow the SPICE deck to be modifiable to support bidirectional simulations.
Drive Strength Block

* Drive Strength Settings

.lib 'drive_select_hio.lib' 3p3ttl_12ma

I/O Buffer Instantiation

The I/O buffer instantiation block of the simulation SPICE deck instantiates the necessary power supplies and I/O model components that are necessary to simulate the given I/O.

I/O Buffer Instantiation

I/O Buffer Instantiation

* Supply Voltages Settings

.param vcn=3.135
.param vpd=2.97
.param vc=1.15

* Instantiate Power Supplies

vvcc       vcc       0     vc       * FPGA core voltage
vvss       vss       0     0        * FPGA core ground
vvccn      vccn      0     vcn      * IO supply voltage
vvssn      vssn      0     0        * IO ground
vvccpd     vccpd     0     vpd      * Pre-drive supply voltage

* Instantiate I/O Buffer

xvio_buf din oeb opdrain die rambh
+ rpcdn4 rpcdn3 rpcdn2 rpcdn1 rpcdn0
+ rpcdp5 rpcdp4 rpcdp3 rpcdp2 rpcdp1 rpcdp0
+ rpullup vccn vccpd vcpad0 vio_buf

* Internal Loading on Pad
* - No loading on this pad due to differential buffer/support
* circuitry

* I/O Buffer Package Model
* - Single-ended I/O standard on a Row I/O

.lib 'lib/package.lib' hio
xpkg die pin hio_pkg

Board Trace and Termination

The board trace and termination block of the simulation SPICE deck is provided only as an example. Replace this block with your own board trace and termination models.

Board Trace and Termination Block

* I/O Board Trace and Termination Description
* - Replace this with your board trace and termination description

wtline pin vssn load vssn N=1 L=1 RLGCMODEL=tlinemodel
.MODEL tlinemodel W MODELTYPE=RLGC N=1 Lo=7.13n Co=2.85p
Rterm2 load vssn 1x

Stimulus Model

The stimulus model block of the simulation spice deck is provided only as a place holder example. Replace this block with your own stimulus model. Options for this include an IBIS or HSPICE model, among others.
**Stimulus Model Block**

* Sample source stimulus placeholder
  * Replace this with your I/O driver model

Vsource source 0 pulse(0 vcn 0s 0.4ns 0.4ns 8.5ns 17.4ns)

**Simulation Analysis**

The simulation analysis block of the simulation file is configured to measure the propagation delay from the source to the FPGA pin. Both the source and end point of the delay are referenced against the 50% $V_{CCN}$ crossing point of the waveform.

**Simulation Analysis Block**

* Simulation Analysis Setup

* Print out the voltage waveform at both the source and the pin
  .print tran v(source) v(pin)

*.tran 0.020ns 17ns

* Measure the propagation delay from the source pin to the pin
  * referenced against the 50% voltage threshold crossing point

.measure TRAN tpd_rise TRIG v(source) val='vcn*0.5' rise=1
+ TARG v(pin) val = 'vcn*0.5' rise=1
.measure TRAN tpd_fall TRIG v(source) val='vcn*0.5' fall=1
+ TARG v(pin) val = 'vcn*0.5' fall=1

**Sample Output for I/O HSPICE Simulation Deck**

A typical HSPICE simulation SPICE deck for an I/O-type output has several sections. Each section presents the simulation file one block at a time.

**Header Comment**

The first block of an output simulation SPICE deck is the header comment. The purpose of this block is to provide a readable summary of how the simulation file has been automatically configured by the Quartus II software.

This block has two main components:

- The first component summarizes the I/O configuration relevant information such as device, speed grade, and so on.
- The second component specifies the exact test condition that the Quartus II software assumes when generating $t_{CO}$ delay numbers. This information is used as part of the double-counting correction circuitry contained in the simulation file.

The SPICE decks are preconfigured to calculate the slow process corner delay but can also be used to simulate the fast process corner as well. The fast corner conditions are listed in the header under the notes section.

The final section of the header comment lists any warning messages that you must consider when you use the SPICE decks.

**Header Comment Block**

* Quartus II HSPICE Writer I/O Simulation Deck
*
Simulation Conditions

The simulation conditions block loads the appropriate process corner models for the transistors. This condition is automatically set up for the slow timing corner and must be modified only if other simulation corners are desired.

Simulation Conditions Block

* Process Settings

.options brief
.inc 'sii_tt.inc' * typical-typical process corner

Note: Two separate corners cannot be simulated at the same time. Instead, simulate the base case using the Quartus corner as one simulation and then perform a second simulation using the desired customer corner. The results of the two simulations can be manually added together.

Simulation Options

The simulation options block configures the simulation temperature and configures HSPICE with typical simulation options.
**Simulation Options Block**

```plaintext
* Simulation Options
  .options brief=0
  .options badchr co=132 scale=1e-6 acct ingold=2 nomod dv=1.0
  +       dcstep=1 absv=1e-3 absi=1e-8 probe csdf=2 accurate=1
  +       converge=1
  .temp 85
```

**Note:** For a detailed description of these options, consult your *HSPICE* manual.

**Constant Definition**

The constant definition block of the output simulation SPICE deck instantiates the voltage sources that controls the configuration modes of the I/O buffer.

**Constant Definition Block**

```plaintext
* Constant Definition

voeb       oeb       0     0 * Set to 0 to enable buffer output
vopdrain   opdrain   0     0 * Set to vc to enable open drain
vrambh     rambh     0     0 * Set to vc to enable bus hold
vrpulup    rpulup    0     0 * Set to vc to enable weak pullup
vpci       rpci      0     0 * Set to vc to enable pci mode
vpcdp4     rpcdp4    0     rp4  * These control bits set the IO standard
vpcdp3     rpcdp3    0     rp3
vpcdp2     rpcdp2    0     rp2
vpcdp1     rpcdp1    0     rp1
vpcdp0     rpcdp0    0     rp0
vpcdn4     rpcdn4    0     rn4
vpcdn3     rpcdn3    0     rn3
vpcdn2     rpcdn2    0     rn2
vpcdn1     rpcdn1    0     rn1
vpcdn0     rpcdn0    0     rn0
vdin       din       0     pulse(0 vc 0s 0.2ns 0.2ns 8.5ns 17.4ns)
```

Where:
- Voltage source `voeb` controls the output enable of the buffer.
- `vopdrain` controls the open drain mode for the I/O.
- `vrambh` controls the bus hold circuitry in the I/O.
- `vpulup` controls the weak pullup.
- `vpci` controls the PCI clamp.
- The next ten voltage sources control the I/O standard of the buffer and are configured through a later library call.
- `vdin` is connected to the data input of the I/O buffer.
- The edge rate of the input stimulus is automatically set to the correct value by the Quartus II software.

**I/O Buffer Netlist**

The I/O buffer netlist block loads all of the models required for the corresponding pin. These include a model for the I/O output buffer, as well as any loads that might be present on the pin.

**I/O Buffer Netlist Block**

```plaintext
*IO Buffer Netlist
```
Drive Strength

The drive strength block of the simulation spice deck loads the configuration bits for configuring the I/O to the proper I/O standard and drive strength. These options are set by the HSPICE Writer tool and are not changed for expected use.

Drive Strength Block

* Drive Strength Settings
  .lib 'drive_select_hio.lib' 3p3ttl_12ma

Slew Rate and Delay Chain

Stratix and Cyclone devices have sections for configuring the slew rate and delay chain settings.

Slew Rate and Delay Chain Settings

* Programmable Output Delay Control Settings
  .lib 'lib/output_delay_control.lib' no_delay
* Programmable Slew Rate Control Settings
  .lib 'lib/slew_rate_control.lib' slow_slow

I/O Buffer Instantiation

The I/O buffer instantiation block of the output simulation spice deck instantiates the necessary power supplies and I/O model components that are necessary to simulate the given I/O.

I/O Buffer Instantiation Block

* I/O Buffer Instantiation
* Supply Voltages Settings
  .param vcn=3.135
  .param vpd=2.97
  .param vc=1.15
* Instantiate Power Supplies
  vvcc       vcc       0     vc     * FPGA core voltage
  vvss       vss       0     0      * FPGA core ground
  vvccn      vccn      0     vcn    * IO supply voltage
  vvssn      vssn      0     0      * IO ground
  vvccpd     vccpd     0     vpd    * Pre-drive supply voltage
* Instantiate I/O Buffer
  xhio_buf din oeb opdrain die rambh
  + rpcdn4 rpcdn3 rpcdn2 rpcdn1 rpcdn0
  + rpcdp4 rpcdp3 rpcdp2 rpcdp1 rpcdp0
  + rpullup vccn vccpd vcpad0 hio_buf
* Internal Loading on Pad
  * - This pad has an LVDS input buffer connected to it, along
* with differential OCT circuitry. Both are disabled but introduce loading on the pad that is modeled below.
xlvds_input_load die vss vccn lvds_input_load
xlvds_oct_load die vss vccpd vccn vcpad0 vccn lvds_oct_load

* I/O Buffer Package Model
  * - Single-ended I/O standard on a Row I/O
.lib 'lib/package.lib' hio
xpkg die pin hio_pkg

**Board and Trace Termination**
The board trace and termination block of the simulation SPICE deck is provided only as an example. Replace this block with your specific board loading models.

**Board Trace and Termination Block**

```
* I/O Board Trace And Termination Description
  * - Replace this with your board trace and termination description
wtline pin vssn load vssn N=1 L=1 RLGCberman tlinemodel
.MODEL tlinemodel W MODELTYPE=RLGC N=1 Lo=7.13n Co=2.85p
Rterm2 load vssn 1x
```

**Double-Counting Compensation Circuitry**
The double-counting compensation circuitry block of the simulation SPICE deck instantiates a second I/O buffer that is used to measure double-counting. The buffer is configured identically to the user I/O buffer but is connected to the Quartus II software test load. The simulated delay of this second buffer can be interpreted as the amount of double-counting between the Quartus II software and HSPICE Writer simulated results.

As the amount of double-counting is constant for a given I/O standard on a given pin, consider separating the double-counting circuitry from the simulation file. In doing so, you can perform any number of I/O simulations while referencing the delay only once.

**(Part of )Double-Counting Compensation Circuitry Block**

```
* Double Counting Compensation Circuitry
  *
  * The following circuit is designed to calculate the amount of double counting between Quartus II and the HSPICE models. If you have not changed the default simulation temperature or transistor corner the double counting will be automatically compensated by this spice deck. In the event you wish to simulate an IO at a different temperature or transistor corner you will need to remove this section of code and manually account for double counting. A description of Altera’s recommended procedure for this process can be found in the Quartus II HSPICE Writer AppNote.
  *
  *
* Supply Voltages Settings
.param vcn_tl=3.135
.param vpd_tl=2.97

* Test Load Constant Definition
vopdrain_tl opdrain_tl 0 0
vrambh_tl rambh_tl 0 0
vnpullup_tl rpullup_tl 0 0
```
* Instantiate Power Supplies
  vvccn_tl  vccn_tl  0  vcn_tl
  vvssn_tl  vssn_tl  0  0
  vvccpd_tl  vccpd_tl  0  vpd_tl

* Instantiate I/O Buffer
  xhio_testload  din  oeb  opdrain_tl  die_tl  rambh_tl
  + rpcdn4  rpcdn3  rpcdn2  rpcdn1  rpcdn0
  + rpcdp4  rpcdp3  rpcdp2  rpcdp1  rpcdp0
  + rpullup_tl  vccn_tl  vccpd_tl  vcpad0_tl  hio_buf

* Internal Loading on Pad
  xlvdsl_input_testload  die_tl  vss  vccn_tl  lvds_input_load
  xlvdsl_oct_testload  die_tl  vss  vccpd_tl  vccn_tl  vcpad0_tl  vccn_tl
  lvds_oct_load

* I/O Buffer Package Model
  * - Single-ended I/O standard on a Row I/O
  .lib 'lib/package.lib' hio
  xpkg die pin hio_pkg

* Default Altera Test Load
  * - 3.3V LVTTL default test condition is an open load

Related Information
The Double Counting Problem in HSPICE Simulations on page 6-17

Simulation Analysis
The simulation analysis block is set up to measure double-counting corrected delays. This is accomplished by measuring the uncompensated delay of the I/O buffer when connected to the user load, and when subtracting the simulated amount of double-counting from the test load I/O buffer.

Simulation Analysis Block
*Simulation Analysis Setup

* Print out the voltage waveform at both the pin and far end load
  .print tran v(pin) v(load)
  .tran 0.020ns 17ns

* Measure the propagation delay to the load pin. This value will
  include some double counting with Quartus II’s Tco
  .measure TRAN tpd_uncomp_rise TRIG v(din) val='vc*0.5’ rise=1
  + TARG v(load) val='vcn*0.5’ rise=1
  .measure TRAN tpd_uncomp_fall TRIG v(din) val='vc*0.5’ fall=1
  + TARG v(load) val='vcn*0.5’ fall=1

* The test load buffer can calculate the amount of double counting
  .measure TRAN t_dblcnt_rise TRIG v(din) val='vc*0.5’ rise=1
  + TARG v(pin_tl) val='vcn_tl*0.5’ rise=1
  .measure TRAN t_dblcnt_fall TRIG v(din) val='vc*0.5’ fall=1
  + TARG v(pin_tl) val='vcn_tl*0.5’ fall=1

* Calculate the true propagation delay by subtraction
  .measure TRAN tpd_rise PARAM='tpd_uncomp_rise-t_dblcnt_rise'
  .measure TRAN tpd_fall PARAM='tpd_uncomp_fall-t_dblcnt_fall'

Advanced Topics
The information in this section describes some of the more advanced topics and methods employed when setting up and running HSPICE simulation files.
PVT Simulations

The automatically generated HSPICE simulation files are set up to simulate the slow process corner using low voltage, high temperature, and slow transistors. To ensure a fully robust link, Altera recommends that you run simulations over all process corners.

To perform process, voltage, and temperature (PVT) simulations, manually modify the spice decks in a two step process:

1. Remove the double-counting compensation circuitry from the simulation file. This is required as the amount of double-counting is dependant upon how the Quartus II software calculates delays and is not based on which PVT corner is being simulated. By default, the Quartus II software provides timing numbers using the slow process corner.
2. Select the proper corner for the PVT simulation by setting the correct HSPICE temperature, changing the supply voltage sources, and loading the correct transistor models.

A more detailed description of HSPICE process corners can be found in the family-specific HSPICE model documentation.

Related Information
Accessing HSPICE Simulation Kits on page 6-16

Hold Time Analysis

Altera recommends performing worst-case hold time analysis using the fast corner models, which use fast transistors, high voltage, and low temperature. This involves modifying the SPICE decks to select the correct temperature option, change the supply voltage sources, and load the correct fast transistor models. The values of these parameters are located in the header comment section of the corresponding simulation deck files.

For a truly worst-case analysis, combine the HSPICE Writer hold time analysis results with the Quartus II software fast timing model. This requires that you change the double-counting compensation circuitry in the simulations files to also simulate the fast process corners, as this is what the Quartus II software uses for the fast timing model.

Note: This method of hold time analysis is recommended only for globally synchronous buses. Do not apply this method of hold-time analysis to source synchronous buses. This is because the source synchronous clocking scheme is designed to cancel out some of the PVT timing effects. If this is not taken into account, the timing results will not be accurate. Proper source synchronous timing analysis is beyond the scope of this document.

I/O Voltage Variations

Use each of the FPGA family datasheets to verify the recommended operating conditions for supply voltages. For current FPGA families, the maximum recommended voltage corresponds to the fast corner, while the minimum recommended voltage corresponds to the slow corner. These voltage recommendations are specified at the power pins of the FPGA and are not necessarily the same voltage that are seen by the I/O buffers due to package IR drops.

The automatically generated HSPICE simulation files model this IR effect pessimistically by including a 50-mV IR drop on the V_{CCPD} supply when a high drive strength standard is being used.

Correlation Report

Correlation reports for the HSPICE I/O models are located in the family-specific HSPICE I/O buffer simulation kits.
## Document Revision History

### Table 6-2: Document Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>June 2014</td>
<td>14.0.0</td>
<td>Updated format.</td>
</tr>
<tr>
<td>December 2010</td>
<td>10.0.1</td>
<td>Template update.</td>
</tr>
<tr>
<td>July 2010</td>
<td>10.0.0</td>
<td>Updated device support.</td>
</tr>
<tr>
<td>November 2009</td>
<td>9.1.0</td>
<td>No change to content.</td>
</tr>
<tr>
<td>March 2009</td>
<td>9.0.0</td>
<td>- Was volume 3, chapter 12 in the 8.1.0 release.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- No change to content.</td>
</tr>
<tr>
<td>November 2008</td>
<td>8.1.0</td>
<td>- Changed to 8-1/2 x 11 page size.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Added information for Stratix III devices.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Input signals for Cyclone III devices are supported.</td>
</tr>
<tr>
<td>May 2008</td>
<td>8.0.0</td>
<td>- Updated “Introduction” on page 12–1.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Updated Figure 12–1.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Updated Figure 12–3.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Updated Figure 12–13.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Updated “Output File Generation” on page 12–6.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Updated “Simulation with HSPICE Models” on page 12–17.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Updated “Invoking HSPICE Writer from the Command Line” on page 12–22.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Added “Sample Input for I/O HSPICE Simulation Deck” on page 12–29.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Added “Sample Output for I/O HSPICE Simulation Deck” on page 12–33.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Updated “Correlation Report” on page 12–41.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Added hyperlinks to referenced documents and websites throughout the chapter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Made minor editorial updates.</td>
</tr>
</tbody>
</table>

### Related Information

**Quartus II Handbook Archive**
For previous versions of the Quartus II Handbook.