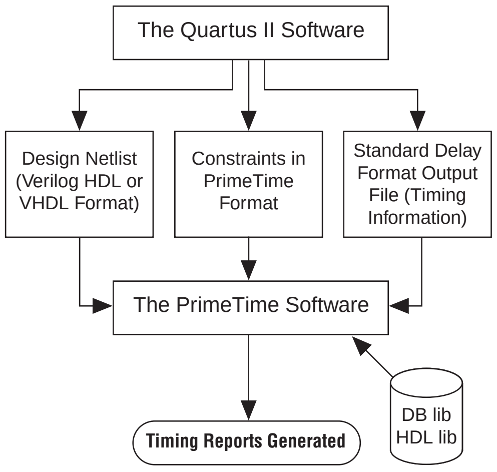
9. Synopsys PrimeTime Support

QII53005-10.0.1

PrimeTime is the Synopsys stand-alone full chip, gate-level static timing analyzer. The Quartus® II software makes it easy for designers to analyze their Quartus II projects using the PrimeTime software. The Quartus II software exports a netlist, design constraints (in the PrimeTime format), and libraries to the PrimeTime software environment. Figure 9–1 shows the PrimeTime flow diagram.

Figure 9–1. PrimeTime Software Flow Diagram


The Quartus II Software
Design Netlist (Verilog HDL or VHDL Format)
Constraints in PrimeTime Format
Standard Delay Format Output File (Timing Information)
The PrimeTime Software
DB lib HDL lib
Timing Reports Generated

This chapter contains the following sections:

- "Quartus II Settings for Generating the PrimeTime Software Files"
- "Files Generated for the PrimeTime Software Environment" on page 9–2
- "Running the PrimeTime Software" on page 9–6
- "PrimeTime Timing Reports" on page 9–7
- "Static Timing Analyzer Differences" on page 9–18

Quartus II Settings for Generating the PrimeTime Software Files

To set up the Quartus II software to generate files for the PrimeTime software, perform the following steps:

1. In the Quartus II software, on the Assignments menu, click **Settings**, and then click **EDA Tool Settings**.
2. In the **Category** list, under **EDA Tool Settings,** select **Timing Analysis**.
3. In the **Tool name** list, select **PrimeTime**, and in the **Format for output netlist** list, select either **Verilog** HDL or **VHDL**.

Subscribe

When you compile your project after making these settings, the Quartus II software runs the EDA Netlist Writer to create three files for the PrimeTime software. These files are saved in the <*revision_name*>**/timing/primetime** directory by default, where <*revision_name*> is the name of your Quartus II software revision. If it is not, you have used the wrong variable name.

Files Generated for the PrimeTime Software Environment

The Quartus II software generates a flattened netlist, a Standard Delay Output File (**.sdo**), and a Tcl script that prepares the PrimeTime software for timing analysis of the Quartus II project. These files are saved in the <*project directory*>**/timing/primetime** directory.

The Quartus II software uses the EDA Netlist Writer to generate PrimeTime files based on either the Classic Timing Analyzer or the TimeQuest Timing Analyzer static timing analysis results. When you run the EDA Netlist Writer, the PrimeTime **.sdo** files are based on delays generated by the currently selected timing analysis tool in the Quartus II software.

To specify the timing analyzer, on the Assignments menu, click **Settings**. The **Settings** dialog box appears. Under **Category**, click **Timing Analysis Settings**. Select the timing analyzer of your choice.

For more information about specifying the Quartus II timing analyzers, refer to either the *Quartus II Classic Timing Analyzer* or the *Quartus II TimeQuest Timing Analyzer* chapter in volume 3 of the *Quartus II Handbook*. Also, refer to the *Switching to the Quartus II TimeQuest Timing Analyzer* chapter in volume 3 of the *Quartus II Handbook* to help you decide which timing analyzer is most appropriate for your design.

The Netlist

Depending on whether **Verilog HDL** or **VHDL** is selected as the **Format for output netlist** option, in the **Tool name** list on the **Timing Analysis** page of the **Settings** dialog box, the netlist is written and saved as either <*project name*>**.vo** or <*project name*>**.vho**, respectively. This file contains the flattened netlist representing the entire design.

When you select the TimeQuest analyzer, only a Verilog HDL PrimeTime netlist can be generated.

The .sdo File

The Quartus II software saves the **.sdo** file as either <*revision_name*>**_v.sdo** or <*revision_name*>**_vhd.sdo**, depending on whether you select **Verilog HDL** or **VHDL** in the **Tool name** list on the **Timing Analysis** page of the **Settings** dialog box.

This file contains the timing information for each timing path between any two nodes in the design.

When you enable the Classic Timing Analyzer, the slow-corner (worst-case) timing models are used by default when generating the **.sdo** file. To generate the **.sdo** file using the fast-corner (best-case) timing models, perform the following steps:

1. In the Quartus II software, on the Processing menu, point to **Start** and click **Start Classic Timing Analyzer (Fast Timing Model)**.
2. After the fast-corner timing analysis is complete, on the Processing menu, point to **Start** and click **Start EDA Netlist Writer** to create a <*revision_name*>**_v_fast.sdo** or <*revision_name*>**_vhd_fast.sdo** file, which contains the best-case delay values for each timing path.

☞ If you are running a best-case timing analysis, the Quartus II software generates a Tcl script similar to the following: <*revision_name*>**_pt_v_fast.tcl**.

When the TimeQuest analyzer is run with the fast-corner netlist, or when the **Optimize fast-corner timing** check box is selected in the **Fitter Settings** dialog box, the fast-corner Synopsys Design Constraints File (**.sdc**) file is generated.

After the EDA Netlist Writer has finished, two **.sdc** files are created: <*revision_name*>**_v.sdo** (slow corner) and <*revision_name*>**_v_fast.sdo** (fast corner).

Generating Multiple Operating Conditions with the TimeQuest Analyzer

You can specify different operating conditions to the EDA Netlist Writer for PrimeTime analysis. The different operating conditions are reflected in the .**sdo** file generated by the EDA Netlist Writer.

☞ From the TimeQuest analyzer console pane, use the command `get_available_operating_conditions` to obtain a list of available operating conditions for the target device.

The following steps show how to generate the .**sdo** files for the three different operating conditions for a Stratix III design. Enter each command at the command prompt.

☞ The `--tq2pt` option for `quartus_sta` is required only if the project does not specify that the PrimeTime tool is be used as the timing analysis tool.

1. Generate the first slow-corner model at the operating conditions: slow, 1100 mV, and 85° C.

```
quartus_sta --model=slow --voltage=1100 --temperature=85 <project name>
```

2. Generate the fast-corner model at the operating conditions: fast, 1100 mV, and 0° C.

```
quartus_sta --model=fast --voltage=1100 --temperature=0
--tq2pt <project name>
```

3. Generate the PrimeTime output files for the corners specified above. The output files are generated in the **primetime_two_corner_files** directory.

```
quartus_eda --timing_analysis --tool=primetime
--format=verilog
--output_directory=primetime_two_corner_files
--write_settings_files=off <project name>
```

4. Generate the second slow-corner model at the operating conditions: slow, 1100 mV, and 0º C.

   ```
   quartus_sta --model=slow --voltage=1100 --temperature=0
   --tq2pt <project name>
   ```

5. Generate the PrimeTime output files for the second slow corner. The output files are generated in the **primetime_one_slow_corner_files** directory.

   ```
   quartus_eda --timing_analysis --tool=primetime
   --format=verilog
   --output_directory=primetime_one_slow_corner_files
   --write_settings_files=off $revision
   ```

To summarize, the previous steps generate the following files for the three operating conditions:

- First slow corner (slow, 1100 mV, 85º C):
  .vo file—primetime_two_corner_files/<*project name*>**.vo**
  .sdo file—primetime_two_corner_files/<*project name*>**_v.sdo**
- Fast corner (fast, 1100 mV, 0º C):
  .vo file—primetime_two_corner_files/<*project name*>**.vo**
  .sdo file—primetime_two_corner_files/<*project name*>**_v_fast.sdo**
- Second slow corner (slow, 1100 mV, 0º C):
  .vo file—primetime_one_slow_corner_files/<*project name*>**.vo**
  .sdo file—primetime_one_slow_corner_files/<*project name*>**_v.sdo**

The **primetime_one_slow_corner_files** directory may also have files for fast corner. These files can be ignored because they were already generated in the **primetime_two_corner_files** directory.

The Tcl Script

The Tcl script generated by the Quartus II software contains information required by the PrimeTime software to analyze the timing and set up your post-fit design. This script specifies the search path and the names of the PrimeTime database library files provided with the Quartus II software. The `search_path` and `link_path` variables are defined at the beginning of the Tcl file. The `link_path` variable is a space-delimited list that contains the names of all database files used by the PrimeTime software.

Depending on whether you select **Verilog** HDL or **VHDL** in the **Format for output netlist** list on the **Timing Analysis** page of the **Settings** dialog box, when the Classic Timing Analyzer is enabled, the EDA Netlist Writer generates and saves the script as either <*revision_name*>**_pt_v.tcl** or <*revision_name*>**_pt_vhd.tcl**.

To access the **EDA Settings** dialog box, perform the following:

1. On the Assignments menu, click **Settings**, and then click **EDA Tool Settings**
2. Expand **EDA Tool Settings** under the **Category** list.

In the dialog box, you can specify VHDL or Verilog HDL for the format of the output netlist.

The script also directs the PrimeTime software to use the <*device family*>**_all_pt.v** or <*device family*>**_all_pt.vhd** file, which contains the Verilog HDL or VHDL description of library cells for the targeted device family.

Example 9–1 shows the `search_path` and `link_path` variables defined in the Tcl script:

Example 9–1. Sample PrimeTime Setup Script

```
set quartus_root "altera/quartus/"
set search_path [list . [format "%s%s" $quartus_root "eda/synopsys/primetime/lib"]  ]

set link_path [list *  stratixii_lcell_comb_lib.db  stratixii_lcell_ff_lib.db
stratixii_asynch_io_lib.db stratixii_io_register_lib.db stratixii_termination_lib.db
bb2_lib.db  stratixii_ram_internal_lib.db  stratixii_memory_register_lib.db
stratixii_memory_addr_register_lib.db  stratixii_mac_out_internal_lib.db
stratixii_mac_mult_internal_lib.db  stratixii_mac_register_lib.db
stratixii_lvds_receiver_lib.db stratixii_lvds_transmitter_lib.db
stratixii_asmiblock_lib.db stratixii_crcblock_lib.db  stratixii_jtag_lib.db
stratixii_rublock_lib.db  stratixii_pll_lib.db  stratixii_dll_lib.db alt_vtl.db]

read_vhdl  -vhdl_compiler  stratixii_all_pt.vhd
```

The EDA Netlist Writer converts any Classic Timing Analyzer timing assignments to the PrimeTime software constraints and exceptions when it generates the PrimeTime files. The converted constraints are saved to the Tcl script. The Tcl script also includes a PrimeTime software command that reads the **.sdo** file generated by the Quartus II software. You can place additional commands in the Tcl script to analyze or report on timing paths.

Table 9–1 shows some examples of timing assignments converted by the Quartus II software for the PrimeTime software. For example, the `set_input_delay -max` command sets the input delay on an input pin.

Table 9–1. Equivalent Quartus II and PrimeTime Software Constraints

Quartus II Equivalent	PrimeTime Constraint
Clock defined on input pin, clock of 10 ns period and 50% duty cycle	`create_clock -period 10.000 -waveform {0 5.000} \ [get_ports clk] -name clk`
Input maximum delay of 1 ns on input pin, din	`set_input_delay -max -add_delay 1.000 -clock \ [get_clocks clk] [get_ports din]`
Input minimum delay of 1 ns on input pin, din	`set_input_delay -min -add_delay 1.000 -clock \ [get_clocks clk] [get_ports din]`
Output maximum delay of 3 ns on output pin, out	`set_output_delay -max -add_delay 3.000 -clock \ [get_clocks clk] [get_ports out]`

When the TimeQuest analyzer is turned on, the EDA Netlist Writer generates and saves the script as <*revision_name*>**.pt.tcl**.

The EDA Netlist Writer converts all TimeQuest analyzer **.sdc** constraints and exceptions into compatible PrimeTime software constraints and exceptions when it generates the PrimeTime files. The constraints and exceptions are saved to the <*revision_name*>**.constraints.sdc** file.

Generated File Summary

The files that are generated by the EDA Netlist Writer for the PrimeTime software depend on the Quartus II timing analysis tool you select.

Table 9–2 shows the files that are generated for the PrimeTime software when the Classic Timing Analyzer is selected.

Table 9–2. Classic Timing Analyzer-Generated PrimeTime Files

File	Description
<*revision_name*>**.vho** | <*revision_name*>**.vo**	The PrimeTime software output netlist. Either a VHDL Output File (**.vho**) or a Verilog Output File (**.vo**) is generated, depending on the output netlist language set.
<*revision_name*>**_vhd.sdo** | <*revision_name*>**_v.sdo**	The PrimeTime software standard delay file. Either a VHDL Standard Delay Output File (**vhd.sdo**) or a Verilog Standard Delay Output File (**v.sdo**) is generated, depending on the output netlist language set.
<*revision_name*>**_pt_vhd.tcl** | <*revision_name*>**_pt_v.tcl**	PrimeTime setup and constraint script. Either a VHDL Tcl Script File (**vhd.tcl**) or a Verilog Tcl Script File (**v.tcl**) is generated, depending on the output netlist language set.

Table 9–3 shows the files that are generated for the PrimeTime software when the TimeQuest analyzer is selected. The EDA Netlist Writer supports the output netlist format only when the TimeQuest analyzer is enabled.

Table 9–3. TimeQuest Timing Analyzer-Generated PrimeTime Files

File	Description
<*revision_name*>**.vo**	The PrimeTime software output netlist. When the TimeQuest analyzer is enabled, only PrimeTime (Verilog HDL) is supported.
<*revision_name*>**_v.sdo** | <*revision_name*>**_v_fast.sdo**	The PrimeTime software standard delay file. When the TimeQuest analyzer is enabled, only PrimeTime (Verilog HDL) is supported.
<*revision_name*>**.pt.tcl**	PrimeTime setup and constraint script. When the TimeQuest analyzer is enabled, only PrimeTime (Verilog HDL) is supported.
<*revision_name*>**.collections.sdc**	Contains the mapping from the TimeQuest analyzer netlist to the PrimeTime netlist.
<*revision_name*>**.constraints.sdc**	Contains the converted TimeQuest analyzer constraints for the PrimeTime software.

Running the PrimeTime Software

The PrimeTime software runs only on UNIX operating systems. If the Quartus II output files for the PrimeTime software were generated by running the Quartus II software on a PC/Windows-based system, follow these steps to run the PrimeTime software using Quartus II output files:

1. Install the PrimeTime libraries on a UNIX system by installing the Quartus II software on UNIX.

   The PrimeTime libraries are located in the <*Quartus II installation directory*>**/eda/synopsys/primetime/lib** directory.

2. Copy the Quartus II output files to the appropriate UNIX directory. You may need to run a PC to UNIX program, such as `dos2unix`, to remove any control characters.

3. Modify the Quartus II path in Tcl scripts to point to the PrimeTime libraries using the first line of Example 9–1:

```
set quartus_root "altera/quartus/" set search_path [list . [format
"%s%s" $quartus_root "eda/synopsys/primetime/lib"]  ]
```

Analyzing Quartus II Projects

The PrimeTime software is controlled with Tcl scripts and can be run through `pt_shell`. You can run the <*revision_name*>**_pt_v.tcl** script file. For example, type the following at a UNIX system command prompt:

```
pt_shell -f <revision_name>_pt_v.tcl ↵
```

When the TimeQuest analyzer is selected, type the following at a UNIX system command prompt:

```
pt_shell -f <revision_name>.pt.tcl ↵
```

After all Tcl commands in the script are interpreted, the PrimeTime software returns control to the `pt_shell` prompt, which allows you to use other commands.

Other pt_shell Commands

You can run additional `pt_shell` commands at the `pt_shell` prompt, including the `man` program. For example, to read documentation about the `report_timing` command, type the following at the `pt_shell` prompt:

```
man report_timing ↵
```

You can list all commands available in `pt_shell` by typing the following at the `pt_shell` prompt:

```
help ↵
```

Type `quit ↵` at the `pt_shell` prompt to close `pt_shell`.

☞ You can also run `pt_shell` without a script file by typing `pt_shell↵` at the UNIX command line prompt.

PrimeTime Timing Reports

This section describes PrimeTime timing reports.

Sample PrimeTime Software Timing Report

After running the script, the PrimeTime software generates a timing report. If the timing constraints are not met, `Violated` is displayed at the end of the timing report. The timing report also gives the negative slack.

The PrimeTime software timing report is similar to the sample shown in Example 9–2. The starting point in this report is a register clocked by clock signal, `clock`, and the endpoint is another register, `inst3-I.lereg`.

Example 9–2. Hold Path Report in PrimeTime

```
Startpoint: inst2~I.lereg
   (rising edge-triggered flip-flop clocked by clock)
Endpoint: inst3~I.lereg
   (rising edge-triggered flip-flop clocked by clock)
Path Group: clock
Path Type: min
Point                                                Incr     Path
-----------------------------------------------------------------
clock clock (rise edge)                              0.000    0.000
clock network delay (propagated)                     3.166    3.166
inst2~I.lereg.clk (stratix_lcell_register)           0.000    3.166r
inst2~I.lereg.regout (stratix_lcell_register) <- 0.176*    3.342r
inst2~I.regout (stratix_lcell)                       0.000*   3.342r
inst3~I.datac (stratix_lcell)                        0.000*   3.342r
inst3~I.lereg.datac (stratix_lcell_register)         3.413*   6.755r
data arrival time                                             6.755
clock clock (rise edge)                              0.000    0.000
clock network delay (propagated)                     3.002    3.002
inst3~I.lereg.clk (stratix_lcell_register)                    3.002r
library hold time                                    0.100*   3.102
data required time                                            3.102
----------------------------------------------------------------
data required time                                            3.102
data arrival time                                            -6.755
----------------------------------------------------------------
slack (MET)                                                   3.653
```

Comparing Timing Reports from the Classic Timing Analyzer and the PrimeTime Software

Both the Classic Timing Analyzer and the TimeQuest analyzer generate a static timing analysis report for every successful design compilation. The timing report lists all of the timing paths in your design that were analyzed, and indicates whether these paths have met or violated their timing requirements. Violations are reported only if timing constraints were specified.

The TimeQuest analyzer and PrimeTime use an equivalent set of equations when reporting the static timing analysis results for a design. However, the Classic Timing Analyzer uses slightly different reporting equations when reporting the static timing analysis results for a design. This section describes the differences between the Classic Timing Analyzer and the PrimeTime software.

The timing report generated by the Classic Timing Analyzer differs from the report generated by the PrimeTime software. Both tools provide the same data, but the data is presented in different formats. The following sections show how the PrimeTime software reports the following slack values differently from the Classic Timing Analyzer report:

- “Clock Setup Relationship and Slack” on page 9–9
- “Clock Hold Relationship and Slack” on page 9–12
- “Input Delay and Output Delay Relationships and Slack” on page 9–16

Clock Setup Relationship and Slack

The Classic Timing Analyzer performs a setup check that ensures that the data launched by source registers is latched correctly at the destination registers. The Classic Timing Analyzer does this by determining the data arrival time and clock arrival time at the destination registers, and compares this data with the setup time delay of the destination register. Equation 9–1 expresses the inequality that is used for a setup check. The data arrival time includes the longest path from the clock to the source register, the clock-to-out micro delay of the source register, and the longest path from the source register to the destination register. The clock arrival time is the shortest delay from the clock to the destination register.

Equation 9–1.

$$\text{Clock Arrival} - \text{Data Arrival} \geq t_{su}$$

Slack is the margin by which a timing requirement is met or not met. Positive slack indicates the margin by which a requirement is met. Negative slack indicates the margin by which a requirement is not met. The Classic Timing Analyzer determines the clock setup slack, as shown in Equation 9–2:

Equation 9–2.

$$\text{Clock Setup Slack} = \text{Largest Register-to-Register Requirement} - \text{Longest Register-to-Register Delay}$$

The longest register-to-register delay in the previous equation is equal to the register-to-register data delay.

Equation 9–3.

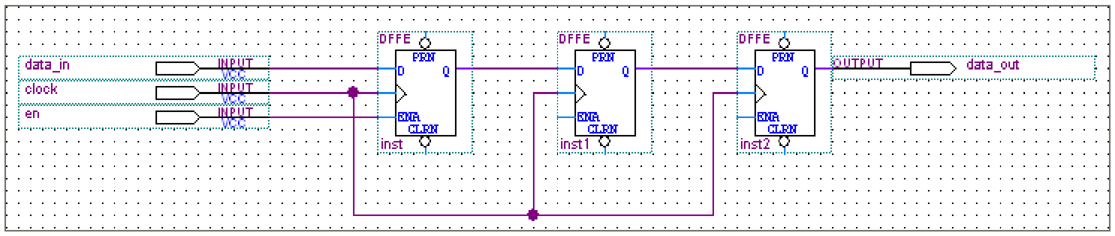
$$\begin{aligned}&\text{Largest Register-to-Register Requirement} = \\ &\text{Setup Relationship between Source and Destination} + \text{Largest Clock Skew} - \\ &\text{Micro } t_{co} \text{ of Destination Register} - \text{Micro } t_{su} \text{ of Destination Register}\end{aligned}$$

$$\text{Setup Relationship between Source and Destination} = \text{Latch Edge} - \text{Launch Edge}$$

$$\text{Clock Skew} = \text{Shortest Clock Path to Destination} - \text{Longest Clock Path to Source}$$

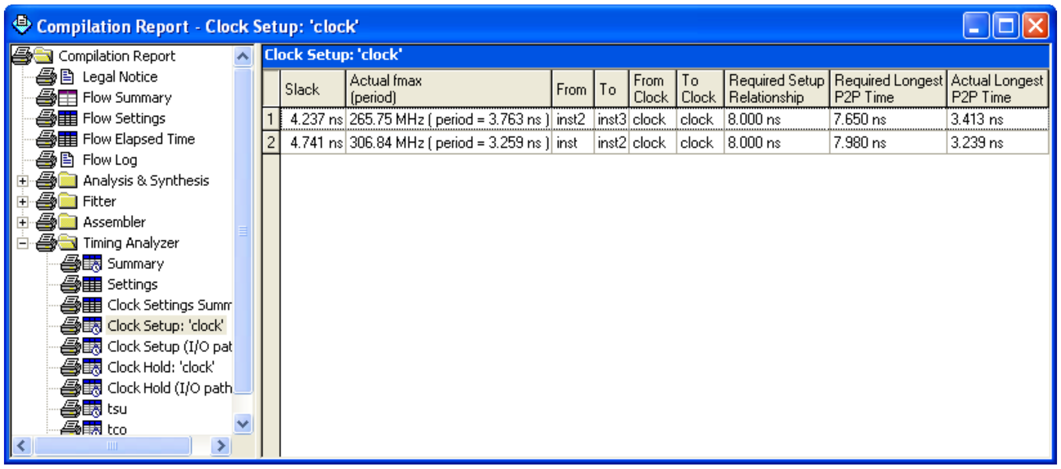
Figure 9–2 shows a simple three-register design.

Figure 9–2. Simple Three-Register Design



The Classic Timing Analyzer generates a report for the design, as shown in Figure 9–3.

Figure 9–3. Timing Analyzer Report from Figure 9–2



Clock Setup: 'clock'

	Slack	Actual fmax (period)	From	To	From Clock	To Clock	Required Setup Relationship	Required Longest P2P Time	Actual Longest P2P Time
1	4.237 ns	265.75 MHz (period = 3.763 ns)	inst2	inst3	clock	clock	8.000 ns	7.650 ns	3.413 ns
2	4.741 ns	306.84 MHz (period = 3.259 ns)	inst	inst2	clock	clock	8.000 ns	7.980 ns	3.239 ns

Equation 9–1, Equation 9–2, and Equation 9–3 are similar to those found in other static timing analysis tools, such as the PrimeTime software. Equation 9–4 through Equation 9–7, used by the PrimeTime software, are essentially the same as those used by the Classic Timing Analyzer, but they are rearranged.

Equation 9–4.

$$\text{Slack} = \text{Data Required} - \text{Data Arrival}$$

Equation 9–5.

$$\text{Clock Arrival} = \text{Latch Edge} + \text{Shortest Clock Path to Destination}$$

Equation 9–6.

$$\text{Data Required} = \text{Clock Arrival} - \text{Micro } t_{su}$$

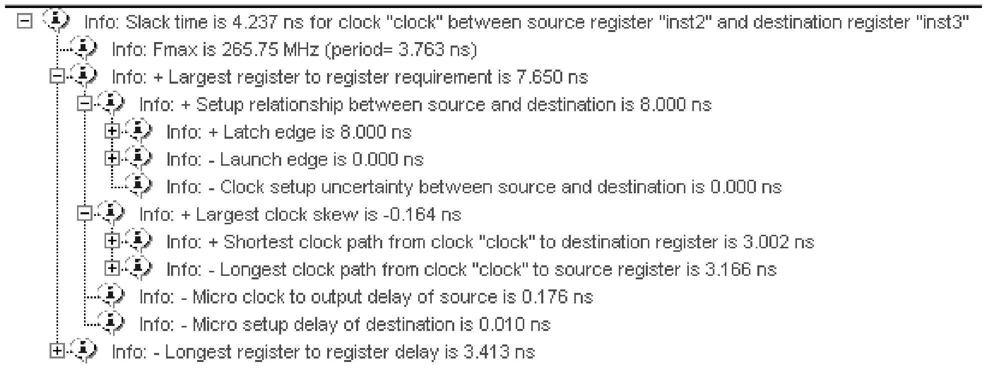
Equation 9–7.

$$\text{Data Arrival} = \text{Launch Edge} + \text{Longest Clock Path to Source} + \text{Micro } t_{co} + \text{Longest Data Delay}$$

☞ The longest data delay in the previous equation is equal to register-to-register data delay.

Figure 9–4 shows a clock setup check in the Quartus II software.

Figure 9–4. Clock Setup Check Reporting with the Classic Timing Analyzer



```
Info: Slack time is 4.237 ns for clock "clock" between source register "inst2" and destination register "inst3"
  Info: Fmax is 265.75 MHz (period= 3.763 ns)
  Info: + Largest register to register requirement is 7.650 ns
    Info: + Setup relationship between source and destination is 8.000 ns
      Info: + Latch edge is 8.000 ns
      Info: - Launch edge is 0.000 ns
      Info: - Clock setup uncertainty between source and destination is 0.000 ns
    Info: + Largest clock skew is -0.164 ns
      Info: + Shortest clock path from clock "clock" to destination register is 3.002 ns
      Info: - Longest clock path from clock "clock" to source register is 3.166 ns
    Info: - Micro clock to output delay of source is 0.176 ns
    Info: - Micro setup delay of destination is 0.010 ns
  Info: - Longest register to register delay is 3.413 ns
```

The results in Equation 9–8 are obtained by extracting the numbers from the Classic Timing Analyzer report and applying them to the clock setup slack equations from the Classic Timing Analyzer:

Equation 9–8.

$$\text{Setup Relationship between Source and Destination} = \text{Latch Edge} - \text{Launch Edge} - \text{Clock Setup Uncertainty}$$

$$8.0 - 0.0 - 0.0 = 8.0\text{ns}$$

$$\text{Clock Skew} = \text{Shortest Clock Path to Destination} - \text{Longest Clock Path to Source}$$

$$3.002 - 3.166 = -0.164\text{ns}$$

$$\text{Largest Register-to-Register Requirement} = \text{Setup Relationship between Source \& Destination} + \text{Largest Clock Skew} - \text{Micro } t_{co} \text{ of Source Register} - \text{Micro } t_{su} \text{ of Destination Register}$$

$$8 + (-0.164) - 0.176 - 0.010 = 7.650\text{ns}$$

$$\text{Clock Setup Slack} = \text{Largest Register-to-Register Requirement} - \text{Longest Register-to-Register Delay}$$

$$7.650 - 3.413 = 4.237\text{ns}$$

For the same register-to-register path, the PrimeTime software generates a clock setup report as shown in Example 9–3:

Example 9–3. Setup Path Report in PrimeTime

```
Startpoint: inst2~I.lereg
   (rising edge-triggered flip-flop clocked by clock)
Endpoint: inst3~I.lereg
   (rising edge-triggered flip-flop clocked by clock)
Path Group: clock
Path Type: max
Point                                                Incr     Path
----------------------------------------------------------------
clock clock (rise edge)                              0.000    0.000
clock network delay (propagated)                     3.166    3.166
inst2~I.lereg.clk (stratix_lcell_register)           0.000    3.166r
inst2~I.lereg.regout (stratix_lcell_register) <-  0.176*   3.342r
inst2~I.regout (stratix_lcell) <-                    0.000*   3.342r
inst3~I.datac (stratix_lcell) <-                     0.000*   3.342r
inst3~I.lereg.datac (stratix_lcell_register)         3.413*   6.755r
data arrival time                                             6.755
clock clock (rise edge)                              8.000    8.000
clock network delay (propagated)                     3.002   11.002
inst3~I.lereg.clk (stratix_lcell_register                    11.002r
library setup time                                  -0.010*  10.992
data required time                                           10.992
----------------------------------------------------------------
data required time                                           10.992
data arrival time                                            -6.755
----------------------------------------------------------------
slack (MET)                                                   4.237
```

Clock Hold Relationship and Slack

The Classic Timing Analyzer performs a hold time check along every register-to-register path in the design to ensure that no hold time violations have occurred. The hold time check verifies that data from the source register does not reach the destination until after the hold time of the destination register. The condition used for a hold check is shown in Equation 9–9:

Equation 9–9.

$$\text{Data Arrival} - \text{Clock Arrival} \geq t_H$$

The Classic Timing Analyzer determines the clock hold slack with Equation 9–10, Equation 9–11, Equation 9–12, and Equation 9–13:

Equation 9–10.

$$\text{Clock Hold Slack} = \text{Shortest Register-to-Register Delay} - \text{Smallest Register-to-Register Requirement}$$

Equation 9–11.

$$\text{Smallest Register-to-Register Requirement} = \text{Hold Relationship between Source \& Destination} + \text{Smallest Clock Skew} - \text{Micro } t_{su} \text{ of Source} + \text{Micro } t_H \text{ of Destination}$$

Equation 9–12.

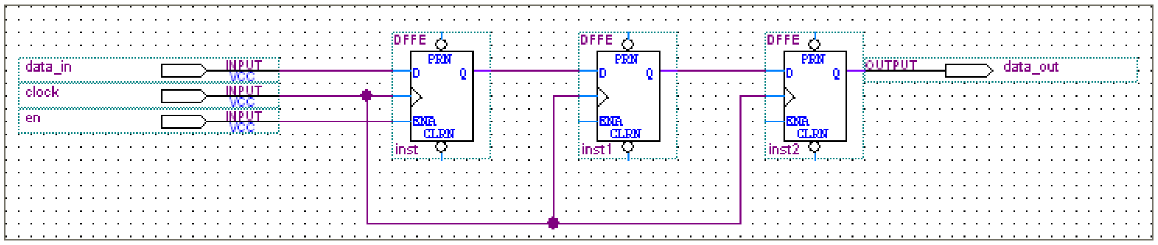
$$\text{Hold Relationship between Source \& Destination} = \text{Latch Edge} - \text{Launch Edge}$$

Equation 9–13.

$$\text{Smallest Clock Skew} = \text{Longest Clock Path from Clock to Destination Register} - \text{Shortest Clock Path from Clock to Source Register}$$

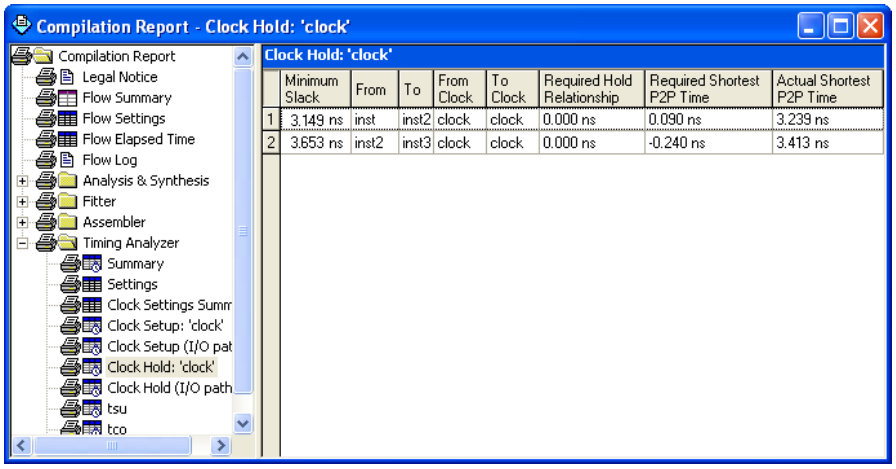
Figure 9–5 shows a simple three-register design.

Figure 9–5. Simple Three-Register Design



The Classic Timing Analyzer generates a report as shown in Figure 9–6.

Figure 9–6. Timing Analyzer Report Generated from the Three-Register Design



Clock Hold: 'clock'

	Minimum Slack	From	To	From Clock	To Clock	Required Hold Relationship	Required Shortest P2P Time	Actual Shortest P2P Time
1	3.149 ns	inst	inst2	clock	clock	0.000 ns	0.090 ns	3.239 ns
2	3.653 ns	inst2	inst3	clock	clock	0.000 ns	-0.240 ns	3.413 ns

The previous equations are similar to those found in the Quartus II software. Equation 9–14 through Equation 9–17 are the same equations that are used by the PrimeTime software, but they are rearranged.

Equation 9–14.

$$\text{Slack} = \text{Data Required} - \text{Data Arrival}$$

Equation 9–15.

$$\text{Clock Arrival} = \text{Latch Edge} + \text{Longest Clock Path to Destination}$$

Equation 9–16.

$$\text{Data Required} = \text{Clock Arrival} - \text{Micro } t_H$$

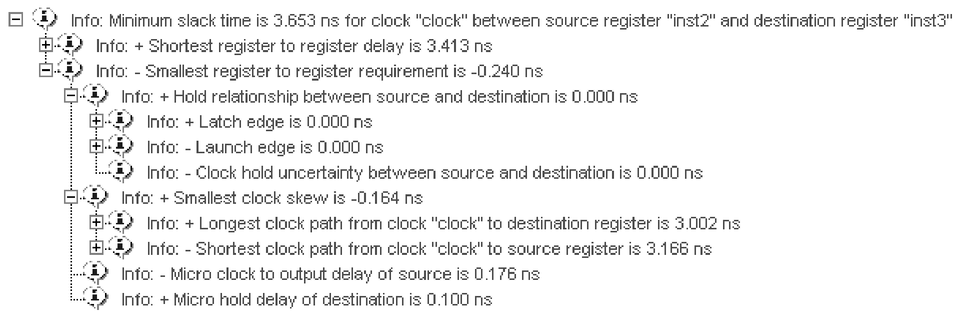
Equation 9–17.

$$\text{Data Arrival} = \text{Launch Edge} + \text{Longest Clock Path to Source} + \text{Micro } t_{co} + \text{Shortest Data Delay}$$

The shortest register-to-register delay in the previous equation is equal to register-to-register data delay.

Figure 9–7 shows a clock setup check with the Classic Timing Analyzer.

Figure 9–7. Clock Hold Check Reporting with the Classic Timing Analyzer



```
Info: Minimum slack time is 3.653 ns for clock "clock" between source register "inst2" and destination register "inst3"
  Info: + Shortest register to register delay is 3.413 ns
  Info: - Smallest register to register requirement is -0.240 ns
    Info: + Hold relationship between source and destination is 0.000 ns
      Info: + Latch edge is 0.000 ns
      Info: - Launch edge is 0.000 ns
      Info: - Clock hold uncertainty between source and destination is 0.000 ns
    Info: + Smallest clock skew is -0.164 ns
      Info: + Longest clock path from clock "clock" to destination register is 3.002 ns
      Info: - Shortest clock path from clock "clock" to source register is 3.166 ns
    Info: - Micro clock to output delay of source is 0.176 ns
    Info: + Micro hold delay of destination is 0.100 ns
```

The results in Equation 9–18 are obtained by extracting the numbers from the Timing Analysis report and applying the clock setup slack equations from the Classic Timing Analyzer.

Equation 9–18.

$$\text{Clock Hold Slack} = \text{Shortest Register-to-Register Delay} - \text{Smallest Register-to-Register Requirement}$$

$$3.413 - (-0.240) = 3.653\text{ns}$$

$$\text{Smallest Register-to-Register Requirement} = \text{Hold Relationship between Source \& Destination} + \text{Smallest Clock Skew} - \text{Micro } t_{co} \text{ of Source} + \text{Micro } t_H \text{ of Destination}$$

$$0 + (-0.164) - 0.176 + 0.100 = -0.240\text{ns}$$

$$\text{Hold Relationship between Source \& Destination} = \text{Latch} - \text{Launch}$$

$$0.0 - 0.0\text{ns}$$

$$\text{Smallest Clock Skew} = \text{Longest Clock Path from Clock to Destination Register} - \text{Shortest Clock Path from Clock to Source Register}$$

$$3.002 - 3.166 = -0.164\text{ns}$$

For the same register-to-register path, the PrimeTime software generates the report shown in Example 9–4:

Example 9–4. Hold Path Report in PrimeTime

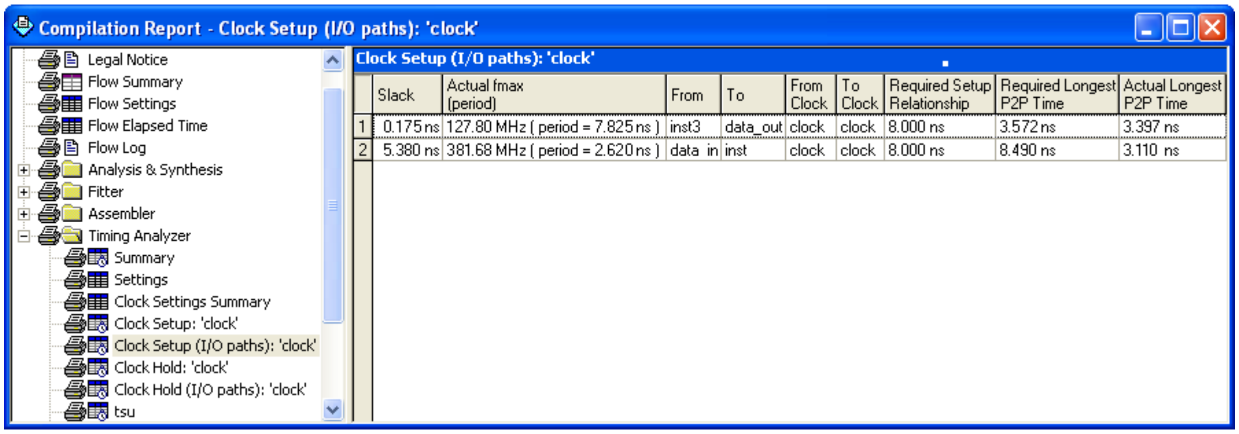
```
Startpoint: inst2~I.lereg
   (rising edge-triggered flip-flop clocked by clock)
Endpoint: inst3~I.lereg
   (rising edge-triggered flip-flop clocked by clock)
Path Group: clock
Path Type: min
Point                                                Incr     Path
----------------------------------------------------------------
clock clock (rise edge)                              0.000    0.000
clock network delay (propagated)                     3.166    3.166
inst2~I.lereg.clk (stratix_lcell_register)           0.000    3.166r
inst2~I.lereg.regout (stratix_lcell_register)<-       0.176*   3.342r
inst2~I.regout (stratix_lcell)                       0.000*   3.342r
inst3~I.datac (stratix_lcell)                        0.000*   3.342r
inst3~I.lereg.datac (stratix_lcell_register)         3.413*   6.755r
data arrival time                                             6.755

clock clock (rise edge)                              0.000    0.000
clock network delay (propagated)                     3.002    3.002
inst3~I.lereg.clk (stratix_lcell_register)                    3.002r
library hold time                                    0.100*   3.102
data required time                                            3.102
--------------------------------------------------------------
data required time                                            3.102
data arrival time                                            -6.755
--------------------------------------------------------------
slack (MET)                                                   3.653
```

Both sets of hold slack equations can be used to determine the hold slack value of any path.

Input Delay and Output Delay Relationships and Slack

Input delay and output delay reports generated by the Classic Timing Analyzer are similar to the clock setup and clock hold relationship reports. Figure 9–8 shows the input delay and output delay report for the design shown in Figure 9–5 on page 9–13.

Figure 9–8. Input and Output Delay Reporting with the Classic Timing Analyzer

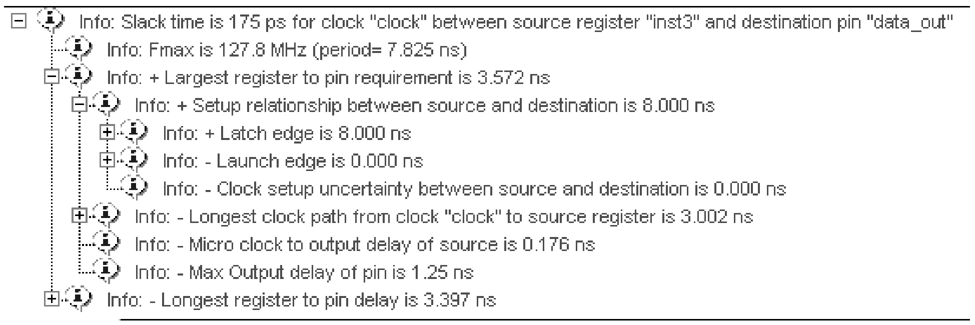


Clock Setup (I/O paths): 'clock'

	Slack	Actual fmax (period)	From	To	From Clock	To Clock	Required Setup Relationship	Required Longest P2P Time	Actual Longest P2P Time
1	0.175 ns	127.80 MHz (period = 7.825 ns)	inst3	data_out	clock	clock	8.000 ns	3.572 ns	3.397 ns
2	5.380 ns	381.68 MHz (period = 2.620 ns)	data_in	inst	clock	clock	8.000 ns	8.490 ns	3.110 ns

Figure 9–9 shows the fully expanded view for the output delay path.

Figure 9–9. Output Delay Path Reporting with the Classic Timing Analyzer



```
Info: Slack time is 175 ps for clock "clock" between source register "inst3" and destination pin "data_out"
  Info: Fmax is 127.8 MHz (period= 7.825 ns)
  Info: + Largest register to pin requirement is 3.572 ns
    Info: + Setup relationship between source and destination is 8.000 ns
      Info: + Latch edge is 8.000 ns
      Info: - Launch edge is 0.000 ns
      Info: - Clock setup uncertainty between source and destination is 0.000 ns
    Info: - Longest clock path from clock "clock" to source register is 3.002 ns
    Info: - Micro clock to output delay of source is 0.176 ns
    Info: - Max Output delay of pin is 1.25 ns
  Info: - Longest register to pin delay is 3.397 ns
```

For the same output delay path, the PrimeTime software generates a report similar to Example 9–5:

Example 9–5. Setup Path Report in PrimeTime

```
Startpoint: inst3~I.lereg
  (rising edge-triggered flip-flop clocked by clock)
Endpoint: data_out
  (output port clocked by clock)
Path Group: clock
Path Type: max
Point                                             Incr     Path
---------------------------------------------------------------
clock clock (rise edge)                           0.000    0.000
clock network delay (propagated)                  3.002    3.002
inst3~I.lereg.clk (stratix_lcell_register)        0.000    3.002r
inst3~I.lereg.regout (stratix_lcell_register)<-   0.176*   3.178r
inst3~I.regout (stratix_lcell)<-                  0.000    3.178r
data_out~I.datain (stratix_io)<-                  0.000    3.178r
data_out~I.out_mux3.A (mux21)<-                   0.000    3.178r
data_out~I.out_mux3.MO (mux21)<-                  0.000    3.178r
data_out~I.and2_22.IN1 (AND2)<-                   0.000    3.178r
data_out~I.and2_22.Y (AND2)<-                     0.000    3.178r
data_out~I.out_mux1.A (mux21)<-                   0.000    3.178r
data_out~I.out_mux1.MO (mux21)<-                  0.000    3.178r
data_out~I.inst1.datain (stratix_asynch_io)<-     0.902*   4.080r
data_out~I.inst1.padio (stratix_asynch_io)<-      2.495*   6.575r
data_out~I.padio (stratix_io)<-                   0.000    6.575r
data_out (out)                                    0.000    6.575r
data arrival time                                          6.575
clock clock (rise edge)                           8.000    8.000
clock network delay (propagated)                  0.000    8.000
output external delay                             1.250    6.750
data required time                                         6.750
--------------------------------------------------------------
data required time                                         6.750
data arrival time                                          6.575
--------------------------------------------------------------
slack (MET)                                                0.175
```

To generate a list of the 100 worst paths and place this data into a file called **file.timing**, type the following command at the `pt_shell` prompt:

```
report_timing -nworst 100 > file.timing ↵
```

Timing paths in the PrimeTime software are listed in the order of most-negative-slack to most-positive-slack. The PrimeTime software does not categorize failing paths by default. Timing setup (tsu) and timing hold (th) times are not listed separately. In the PrimeTime software, each path is shown with a start and end point; for example, if it is a register-to-register or input-to-register type of path. If you only use the `report_timing` part of the command without adding a `-delay` option, only the `setup-time-related` timing paths are reported.

The following command is used to create a minimum timing report or a list of hold-time-related violations:

```
report_timing -delay_type min ↵
```

Ensure that the correct .**sdo** file, either minimum or maximum delays, is loaded before running this command.

Static Timing Analyzer Differences

Under certain design conditions, several static timing analysis differences can exist between the Classic Timing Analyzer and the TimeQuest analyzer, and the PrimeTime software. The following sections explain the differences between the two static timing analysis engines and the PrimeTime software.

Classic Timing Analyzer and PrimeTime Software

The following section describes the differences between the Classic Timing Analyzer and the PrimeTime software.

Rise/Fall Support

The Classic Timing Analyzer does not support rise/fall analysis. However, rise/fall support is available in PrimeTime.

Minimum and Maximum Delays

The Classic Timing Analyzer calculates minimum and maximum delays for all device components with the exception of clock routing. PrimeTime does not model these delays. This can result in different slacks for a given path on average of 2 to 3%.

Recovery/Removal Analysis

The Classic Timing Analyzer performs a more pessimistic recovery/removal analysis for asynchronous paths than PrimeTime. This can result in different delays reported between the two tools.

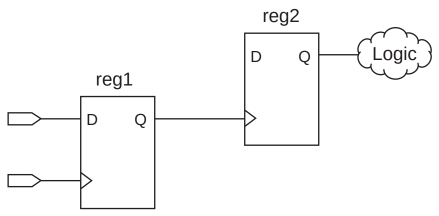
Encrypted Intellectual Property Blocks

The Quartus II software has the capability to decrypt all intellectual property (IP) blocks designed for Altera® devices that have been encrypted by their vendors. The decryption process allows the Quartus II software to perform a full compilation of the design that contains an encrypted IP block. This also allows the Classic Timing Analyzer to perform a complete static timing analysis on the design. However, licensed and encrypted IP blocks do not permit output netlists to be generated when using PrimTime as the static timing analysis tool. (The EDA Netlist Writer does not generate .**vho** or .**vo** netlist files.)

Registered Clock Signals

Registered clock signals are clock signals that pass through a register before reaching the clock port of a sequential element. Figure 9–10 shows an example of a registered clock signal.

Figure 9–10. Registered Clock Signal



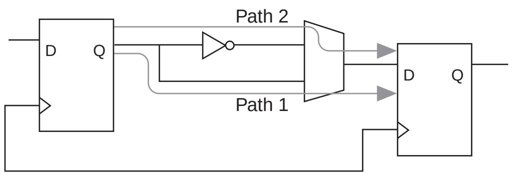
If no clock setting is applied to the register on the clock path (shown as register `reg_1` in Figure 9–10), the Classic Timing Analyzer treats the register in the clock path as a buffer. The delay of the buffer is equal to the CELL delay of the register plus the t_{CO} of the register. The PrimeTime software does not treat the register as a buffer.

☞ For more information about creating clock settings, refer to the *Quartus II Classic Timing Analyzer* chapter in volume 3 of the *Quartus II Handbook*.

Multiple Source and Destination Register Pairs

In any design, multiple paths may exist from a source register to a destination register. Each path from the source register to the destination register may have a different delay value due to the different routes taken. For example, Figure 9–11 shows a sample design that contains multiple path pairs between the source register and destination register.

Figure 9–11. Multiple Source and Destination Pairs



The Classic Timing Analyzer analyzes all source and destination pairs, but reports only the source and destination register pair with the worst slack. For example, if the Path 2 pair delay is greater than the Path 1 pair delay in Figure 9–11, the Classic Timing Analyzer reports the slack value of the Path 2 pair and not the Path 1 pair. The PrimeTime software reports all possible source and destination register pairs.

Latches

By default, the Quartus II software implements all latches as combinational loops. The Classic Timing Analyzer can analyze such latches by treating them as registers with inverted clocks or analyze latches as a combinational loop modeled as a combinational delay.

For more information about latch analysis, refer to the *Quartus II Classic Timing Analyzer* chapter in volume 3 of the *Quartus II Handbook.*

The PrimeTime software always analyzes these latches as combinational loops, as defined in the netlist file.

LVDS I/O

When analyzing the dedicated LVDS transceivers in your design, the Classic Timing Analyzer generates the Receiver Skew Margin (RSKM) report and a Channel-to-Channel Skew (TCCS) report. The PrimeTime software does not generate these reports.

Clock Latency

When a single clock signal feeds both the source and destination registers of a register-to-register path, and either an Early Clock Latency or a Late Clock Latency assignment has been applied to the clock signal, the Classic Timing Analyzer does not factor in the clock latency values when it calculates the clock skew between the two registers. The Classic Timing Analyzer factors in the clock latency values when the clock signal to the source and destination registers of a register-to-register path are different. The PrimeTime software applies the clock latency values when a single clock signal or different clock signals feeds the source and destination registers of a register-to-register path.

Input and Output Delay Assignments

When a purely combinational (non-registered) path exists between an input pin and output pin of the Altera FPGA and both pins have been constrained with an input delay and an output delay assignment applied, respectively, the Classic Timing Analyzer does not perform a clock setup or clock hold analysis. The PrimeTime software analyzes these paths.

Generated Clocks Derived from Generated Clocks

The Classic Timing Analyzer does not support a generated clock derived from a generated clock. This situation might occur if a generated clock feeds the input clock pin of a PLL. The output clock of the PLL is a generated clock.

TimeQuest Timing Analyzer and PrimeTime Software

The following sections describe the static timing analysis differences between the TimeQuest analyzer and the PrimeTime software.

Encrypted Intellectual Property Blocks

The Quartus II software has the capability to decrypt all IP blocks, designed for Altera devices that have been encrypted by their vendors. The decryption process allows the Quartus II software to perform a full compilation on the design containing an encrypted IP block. This also allows the TimeQuest analyzer to perform a complete static timing analysis on the design. However, licensed and encrypted IP blocks do not permit output netlists to be generated when using PrimTime as the static timing analysis tool. (The EDA Netlist Writer does not generate **.vho** or **.vo** netlist files.)

Latches

By default, the Quartus II software implements all latches as combinational loops. The TimeQuest analyzer can analyze such latches by treating them as registers with inverted clocks. The TimeQuest analyzer analyzes latches as a combinational loop modeled as a combinational delay.

For more information about latch analysis, refer to the *Quartus II TimeQuest Timing Analyzer* chapter in volume 3 of the *Quartus II Handbook.*

The PrimeTime software always analyzes these latches as combinational loops, as defined in the netlist file.

LVDS I/O

When analyzing the dedicated LVDS transceivers in your design, the TimeQuest analyzer generates a Receiver Skew Margin (RSKM) report and a Channel-to-Channel Skew (TCCS) report. The PrimeTime software does not generate these reports.

The TimeQuest Timing Analyzer .sdc File and PrimeTime Compatibility

Because of differences between node naming conventions with the netlist generated by the EDA Netlist Writer and the internal netlist used by the Quartus II software, **.sdc** files generated for the Quartus II software or the TimeQuest analyzer are not compatible with the PrimeTime software.

Run the EDA Netlist Writer to generate a compatible **.sdc** file from the TimeQuest **.sdc** file for the PrimeTime software. After the files <*revision_name*>**.collections.sdc** and <*revision_name*>**.constraints.sdc** have been generated, both files can be read by the PrimeTime software for compatibility of constraints between the TimeQuest analyzer and the PrimeTime software.

Clock and Data Paths

If a timing path acts both as a clock path (a path that connects to a clock pin with a clock associated with it), and a data path (a path that feeds into the data-in port of a register), the TimeQuest analyzer reports the data paths, whereas PrimeTime does not.

Inverting and Non-Inverting Propagation

The TimeQuest analyzer always propagates non-inverting sense for clocks through non-unate paths in the clock network.

PrimeTime's default behavior is to propagate both inverting and non-inverting senses through a non-unate path in the clock network.

Multiple Rise/Fall Numbers For a Timing Arc

For a given timing path with a corresponding set of pins/ports that make up the path (including source and destination pair), if the individual components of that path have different rise/fall delays, there can potentially be many timing paths with different delays using the same set of pins. If this occurs, the TimeQuest analyzer reports only one timing path for the set of pins that make up the path.

Virtual Generated Clocks

PrimeTime does not support virtual generated clocks. To maintain compatibility between the TimeQuest analyzer and PrimeTime, all generated clocks should have an explicit target specified.

Generated Clocks Derived from Generated Clocks

The Classic Timing Analyzer does not support the creation of a generated clock derived from a generated clock. This situation might occur if a generated clock feeds the input clock pin of another generated clock. The output clock of the PLL is a generated clock.

Conclusion

The Quartus II software can export a netlist, constraints, and timing information for use with the PrimeTime software. The PrimeTime software can use data from either best-case or worst-case Quartus II timing models to measure timing. The PrimeTime software is controlled using a Tcl script generated by the Quartus II software that you can customize to direct the PrimeTime software to produce violation and slack reports.

Document Revision History

Table 9–4 shows the revision history for this chapter.

Table 9–4. Document Revision History

Date	Version	Changes Made
December 2010	10.0.1	Changed to new document template.
July 2010	10.0.0	■ Minor corrections throughout, and Quartus II interface changes.
November 2009	9.1.0	■ Updated "Setting the Quartus II Software to Generate the PrimeTime Software Files" figure for changes in the Quartus II software version 9.1
March 2009	9.0.0	■ This was chapter 10 in version 8.1. ■ Updated for the Quartus II software version 9.0 release.
November 2008	8.1.0	■ Changed to 8-1/2 x 11 page size. No change to content.
May 2008	8.0.0	■ Updated to Quartus II software version 8.0 and date. ■ Added hyperlinks to referenced Altera documentation throughout the chapter.

For previous versions of the *Quartus II Handbook*, refer to the Quartus II Handbook Archive.

Take an online survey to provide feedback about this handbook chapter.