

Comprehensive static timing analysis involves analysis of register-to-register, I/O, and asynchronous reset paths. Timing analysis with the TimeQuest Timing Analyzer uses data required times, data arrival times, and clock arrival times to verify circuit performance and detect possible timing violations. The TimeQuest analyzer determines the timing relationships that must be met for the design to correctly function, and checks arrival times against required times to verify timing. This chapter is an overview of the concepts you need to know to analyze your designs with the TimeQuest analyzer.

 For more information about the TimeQuest analyzer flow and TimeQuest examples, refer to *The Quartus II TimeQuest Timing Analyzer* chapter of the *Quartus II Handbook*.

TimeQuest Terminology and Concepts

Table 6–1 describes TimeQuest analyzer terminology.

Table 6–1. TimeQuest Analyzer Terminology

Term	Definition
nodes	Most basic timing netlist unit. Used to represent ports, pins, and registers.
cells	Look-up tables (LUT), registers, digital signal processing (DSP) blocks, memory blocks, input/output elements, and so on. ⁽¹⁾
pins	Inputs or outputs of cells.
nets	Connections between pins.
ports	Top-level module inputs or outputs; for example, device pins.
clocks	Abstract objects representing clock domains inside or outside of your design.

Notes to Table 6–1:

(1) For Stratix® devices, the LUTs and registers are contained in logic elements (LE) and modeled as cells.

Timing Netlists and Timing Paths

The TimeQuest analyzer requires a timing netlist to perform timing analysis on any design. After you generate a timing netlist, the TimeQuest analyzer uses the data to help determine the different design elements in your design and how to analyze timing.



The Timing Netlist

Figure 6-1 shows a sample design for which the TimeQuest analyzer generates a timing netlist equivalent.

Figure 6-1. Sample Design

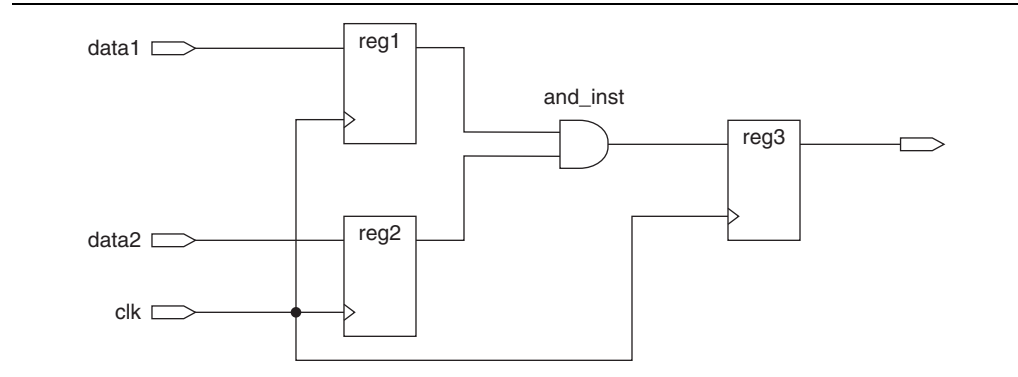
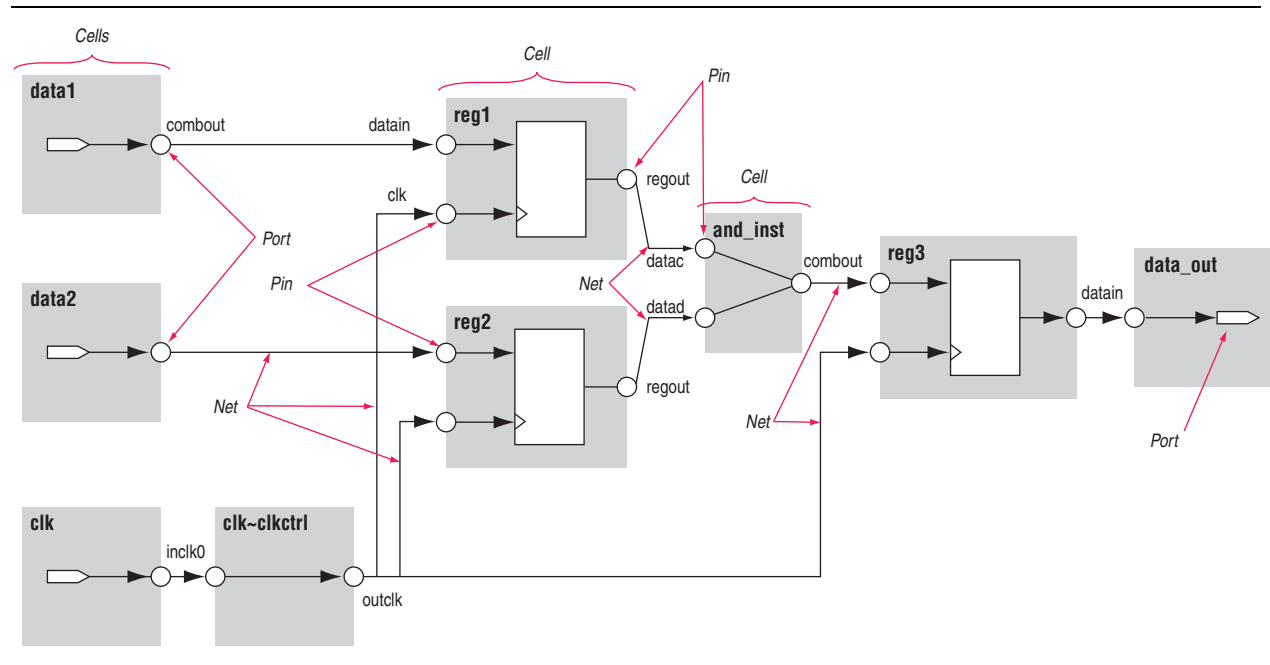


Figure 6-2 shows the timing netlist for the sample design in Figure 6-1, including how different design elements are divided into cells, pins, nets, and ports.

Figure 6-2. The TimeQuest Analyzer Timing Netlist



Timing Paths

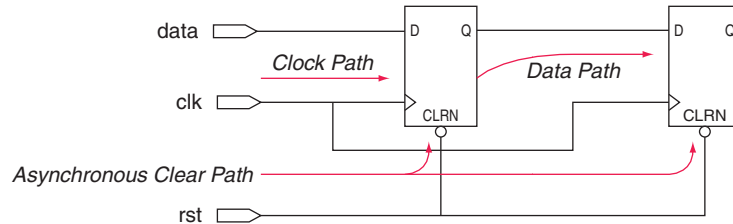
Timing paths connect two design nodes, such as the output of a register to the input of another register. Understanding the types of timing paths is important to timing closure and optimization. The TimeQuest analyzer uses the following commonly analyzed paths:

- **Edge paths**—connections from ports-to-pins, from pins-to-pins, and from pins-to-ports.

- **Clock paths**—connections from device ports or internally generated clock pins to the clock pin of a register.
- **Data paths**—connections from a port or the data output pin of a sequential element to a port or the data input pin of another sequential element.
- **Asynchronous paths**—connections from a port or asynchronous pins of another sequential element such as an asynchronous reset or asynchronous clear.

Figure 6-3 shows path types commonly analyzed by the TimeQuest analyzer.

Figure 6-3. Path Types



In addition to identifying various paths in a design, the TimeQuest analyzer analyzes clock characteristics to compute the worst-case requirement between any two registers in a single register-to-register path. You must constrain all clocks in your design before analyzing clock characteristics.

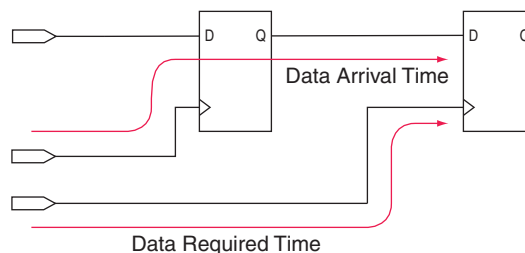
Data and Clock Arrival Times

After the TimeQuest analyzer identifies the path type, it can report data and clock arrival times at register pins.

The TimeQuest analyzer calculates data arrival time by adding the launch edge time to the delay from the clock source to the clock pin of the source register, the micro clock-to-output delay (μt_{CO}) of the source register, and the delay from the source register's data output (Q) to the destination register's data input (D).

The TimeQuest analyzer calculates data required time by adding the latch edge time to the sum of all delays between the clock port and the clock pin of the destination register, including any clock port buffer delays, and subtracts the micro setup time (μt_{SU}) of the destination register, where the μt_{SU} is the intrinsic setup time of an internal register in the FPGA. Figure 6-4 shows the flow calculated for data arrival time and data required time.

Figure 6-4. Data Arrival and Data Required Times



Equation 6-1 shows the basic calculations for data arrival and data required times including the launch and latch edges.

Equation 6-1. Data Arrival and Data Required Time Equations

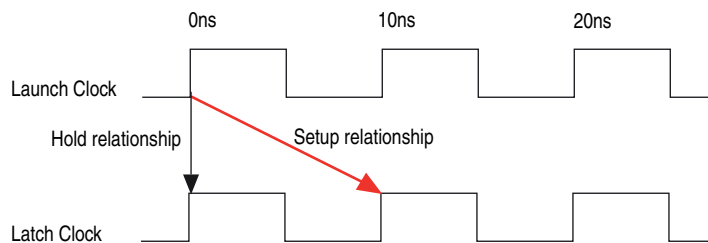
$$\begin{aligned} \text{Data Arrival Time} &= \text{Launch Edge} + \text{Source Clock Delay} + \mu t_{CO} + \text{Register-to-Register Delay} \\ \text{Data Required Time} &= \text{Latch Edge} + \text{Destination Clock Delay} - \mu t_{SU} \end{aligned}$$

Launch and Latch Edges

All timing relies on one or more clocks. In addition to analyzing paths, the TimeQuest analyzer determines clock relationships for all register-to-register transfers in your design. Figure 6-5 shows the launch edge, which is the clock edge that sends data out of a register or other sequential element, and acts as a source for the data transfer. A latch edge is the active clock edge that captures data at the data port of a register or other sequential element, acting as a destination for the data transfer. In this example, the launch edge sends the data from register reg1 at 0 ns, and the register reg2 captures the data when triggered by the latch edge at 10 ns. The data arrives at the destination register before the next latch edge.

In timing analysis, and with the TimeQuest analyzer specifically, you create clock constraints and assign those constraints to nodes in your design. These clock constraints provide the structure required for repeatable data relationships. The primary relationships between clocks, in the same or different domains, are the setup relationship and the hold relationship. Figure 6-5 also shows the setup and hold relationships between a launch edge and a latch edge which are 10ns apart.

Figure 6-5. Launch and Latch Edges

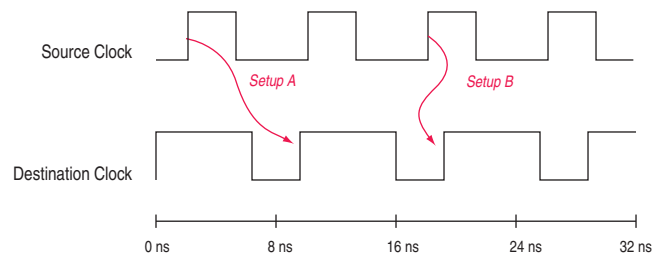


If you do not constrain the clocks in your design, the Quartus II software analyzes in terms of a 1 GHz clock to maximize timing based Fitter effort. To ensure realistic slack values, you must constrain all clocks in your design with real values.

Clock Setup Check

To perform a clock setup check, the TimeQuest analyzer determines a setup relationship by analyzing each launch and latch edge for each register-to-register path. For each latch edge at the destination register, the TimeQuest analyzer uses the closest previous clock edge at the source register as the launch edge. Figure 6-6 shows two setup relationships, setup A and setup B. For the latch edge at 10 ns, the closest clock that acts as a launch edge is at 3 ns and is labeled setup A. For the latch edge at 20 ns, the closest clock that acts as a launch edge is 19 ns and is labeled setup B. TimeQuest analyzes the most restrictive setup relationship, in this case setup B; if that relationship meets the design requirement, then setup A meets it by default.

Figure 6-6. Setup Check



The TimeQuest analyzer reports the result of clock setup checks as slack values. 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. Equation 6-2 shows the TimeQuest analyzer clock setup slack time calculation for internal register-to-register paths.

Equation 6-2. Clock Setup Slack for Internal Register-to-Register paths

$$\begin{aligned} \text{Clock Setup Slack} &= \text{Data Required Time} - \text{Data Arrival Time} \\ \text{Data Arrival Time} &= \text{Launch Edge} + \text{Clock Network Delay to Source Register} + \\ &\quad \mu t_{CO} + \text{Register-to-Register Delay} \\ \text{Data Required Time} &= \text{Latch Edge} + \text{Clock Network Delay to Destination Register} - \\ &\quad \mu t_{SU} - \text{Setup Uncertainty} \end{aligned}$$

The TimeQuest analyzer performs setup checks using the maximum delay when calculating data arrival time, and minimum delay when calculating data required time.

Equation 6-3 shows the TimeQuest analyzer clock setup slack time calculation if the data path is from an input port to an internal register.

Equation 6-3. Clock Setup Slack from Input Port to Internal Register

$$\begin{aligned} \text{Clock Setup Slack} &= \text{Data Required Time} - \text{Data Arrival Time} \\ \text{Data Arrival Time} &= \text{Launch Edge} + \text{Clock Network Delay} + \\ &\quad \text{Input Maximum Delay} + \text{Port-to-Register Delay} \\ \text{Data Required Time} &= \text{Latch Edge} + \text{Clock Network Delay to Destination Register} - \\ &\quad \mu t_{SU} - \text{Setup Uncertainty} \end{aligned}$$

Equation 6-4 shows the TimeQuest analyzer clock setup slack time calculation if the data path is an internal register to an output port.

Equation 6-4. Clock Setup Slack from Internal Register to Output Port

Clock Setup Slack = Data Required Time – Data Arrival Time

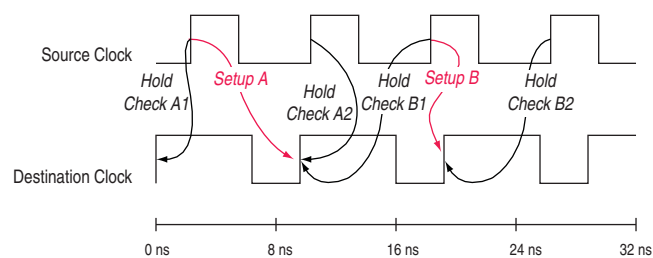
Data Required Time = Latch Edge + Clock Network Delay to Output Port –
Output Maximum Delay

Data Arrival Time = Launch Edge + Clock Network Delay to Source Register +
 μt_{CO} + Register-to-Port Delay

Clock Hold Check

To perform a clock hold check, the TimeQuest analyzer determines a hold relationship for each possible setup relationship that exists for all source and destination register pairs. The TimeQuest analyzer checks all adjacent clock edges from all setup relationships to determine the hold relationships. The TimeQuest analyzer performs two hold checks for each setup relationship. The first hold check determines that the data launched by the current launch edge is not captured by the previous latch edge. The second hold check determines that the data launched by the next launch edge is not captured by the current latch edge. From the possible hold relationships, the TimeQuest analyzer selects the hold relationship that is the most restrictive. The most restrictive hold relationship is the hold relationship with the smallest difference between the latch and launch edges and determines the minimum allowable delay for the register-to-register path. Figure 6-7 shows two setup relationships, setup A and setup B, and their respective hold checks. In this example, the TimeQuest analyzer selects hold check A2 as the most restrictive hold relationship.

Figure 6-7. Hold Checks



Equation 6-5 shows the TimeQuest analyzer clock hold slack time calculation.

Equation 6-5. Clock Hold Slack for Internal Register-to-Register Paths

Clock Hold Slack = Data Arrival Time – Data Required Time

Data Arrival Time = Launch Edge + Clock Network Delay to Source Register +
 μt_{CO} + Register-to-Register Delay

Data Required Time = Latch Edge + Clock Network Delay to Destination Register +
 μt_H + Hold Uncertainty

The TimeQuest analyzer performs hold checks using the minimum delay when calculating data arrival time, and maximum delay when calculating data required time.

Equation 6-6 shows the TimeQuest analyzer hold slack time calculation if the data path is from an input port to an internal register.

Equation 6-6. Clock Hold Slack from Input Port to Internal Register

$$\begin{aligned} \text{Clock Hold Slack} &= \text{Data Arrival Time} - \text{Data Required Time} \\ \text{Data Arrival Time} &= \text{Launch Edge} + \text{Clock Network Delay} + \\ &\quad \text{Input Minimum Delay} + \text{Pin-to-Register Delay} \\ \text{Data Required Time} &= \text{Latch Edge} + \text{Clock Network Delay to Destination Register} + \mu t_H \end{aligned}$$

Equation 6-7 shows the TimeQuest analyzer hold slack time calculation if the data path is from an internal register to an output port.

Equation 6-7. Clock Hold Slack from Internal Register to Output Port

$$\begin{aligned} \text{Clock Hold Slack} &= \text{Data Arrival Time} - \text{Data Required Time} \\ \text{Data Arrival Time} &= \text{Latch Edge} + \text{Clock Network Delay to Source Register} + \\ &\quad \mu t_{CO} + \text{Register-to-Pin Delay} \\ \text{Data Required Time} &= \text{Latch Edge} + \text{Clock Network Delay} - \text{Output Minimum Delay} \end{aligned}$$

Recovery and Removal Time

Recovery time is the minimum length of time for the deassertion of an asynchronous control signal relative to the next clock edge; for example, signals such as `clear` and `preset` must be stable before the next active clock edge. The recovery slack calculation is similar to the clock setup slack calculation, but it applies to asynchronous control signals. Equation 6-8 shows the TimeQuest analyzer recovery slack time calculation if the asynchronous control signal is registered.


Equation 6-8. Recovery Slack if Asynchronous Control Signal Registered

$$\begin{aligned} \text{Recovery Slack Time} &= \text{Data Required Time} - \text{Data Arrival Time} \\ \text{Data Required Time} &= \text{Latch Edge} + \text{Clock Network Delay to Destination Register} - \mu t_{SU} \\ \text{Data Arrival Time} &= \text{Launch Edge} + \text{Clock Network Delay to Source Register} + \\ &\quad \mu t_{CO} + \text{Register-to-Register Delay} \end{aligned}$$

Equation 6-9 shows the TimeQuest analyzer recovery slack time calculation if the asynchronous control signal is not registered.

Equation 6-9. Recovery Slack if Asynchronous Control Signal not Registered

$$\begin{aligned} \text{Recovery Slack Time} &= \text{Data Required Time} - \text{Data Arrival Time} \\ \text{Data Required Time} &= \text{Latch Edge} + \text{Clock Network Delay to Destination Register} - \mu t_{SU} \\ \text{Data Arrival Time} &= \text{Launch Edge} + \text{Clock Network Delay} + \text{Input Maximum Delay} + \\ &\quad \text{Port-to-Register Delay} \end{aligned}$$

 If the asynchronous reset signal is from a device I/O port, you must create an input delay constraint for the asynchronous reset port for the TimeQuest analyzer to perform recovery analysis on the path.

Removal time is the minimum length of time the deassertion of an asynchronous control signal must be stable after the active clock edge. The TimeQuest analyzer removal slack calculation is similar to the clock hold slack calculation, but it applies asynchronous control signals. Equation 6-10 shows the TimeQuest analyzer removal slack time calculation if the asynchronous control signal is registered.


Equation 6-10. Removal Slack if Asynchronous Control Signal Registered

$$\begin{aligned} \text{Removal Slack Time} &= \text{Data Arrival Time} - \text{Data Required Time} \\ \text{Data Arrival Time} &= \text{Launch Edge} + \text{Clock Network Delay to Source Register} + \\ &\quad \mu t_{CO} \text{ of Source Register} + \text{Register-to-Register Delay} \\ \text{Data Required Time} &= \text{Latch Edge} + \text{Clock Network Delay to Destination Register} + \mu t_H \end{aligned}$$

Equation 6-11 shows the TimeQuest analyzer removal slack time calculation if the asynchronous control signal is not registered.

Equation 6-11. Removal Slack if Asynchronous Control Signal not Registered

$$\begin{aligned} \text{Removal Slack Time} &= \text{Data Arrival Time} - \text{Data Required Time} \\ \text{Data Arrival Time} &= \text{Launch Edge} + \text{Clock Network Delay} + \text{Input Minimum Delay of Pin} + \\ &\quad \text{Minimum Pin-to-Register Delay} \\ \text{Data Required Time} &= \text{Latch Edge} + \text{Clock Network Delay to Destination Register} + \mu t_H \end{aligned}$$

 If the asynchronous reset signal is from a device pin, you must assign the **Input Minimum Delay** timing assignment to the asynchronous reset pin for the TimeQuest analyzer to perform removal analysis on the path.

Multicycle Paths

Multicycle paths are data paths that require a non-default setup and/or hold relationship for proper analysis. For example, a register may be required to capture data on every second or third rising clock edge. Figure 6-8 shows an example of a multicycle path between the input registers of a multiplier and an output register where the destination latches data on every other clock edge.

Figure 6-8. Multicycle Path

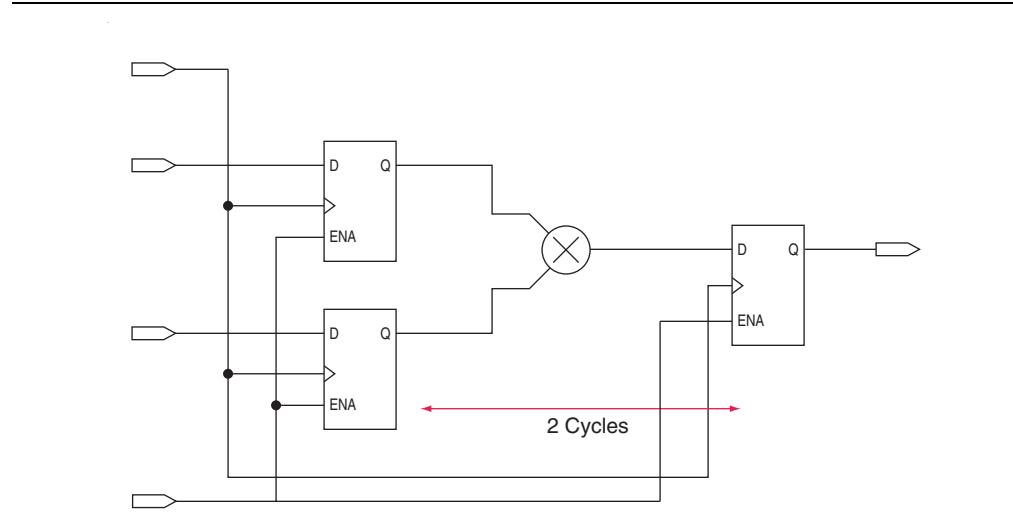
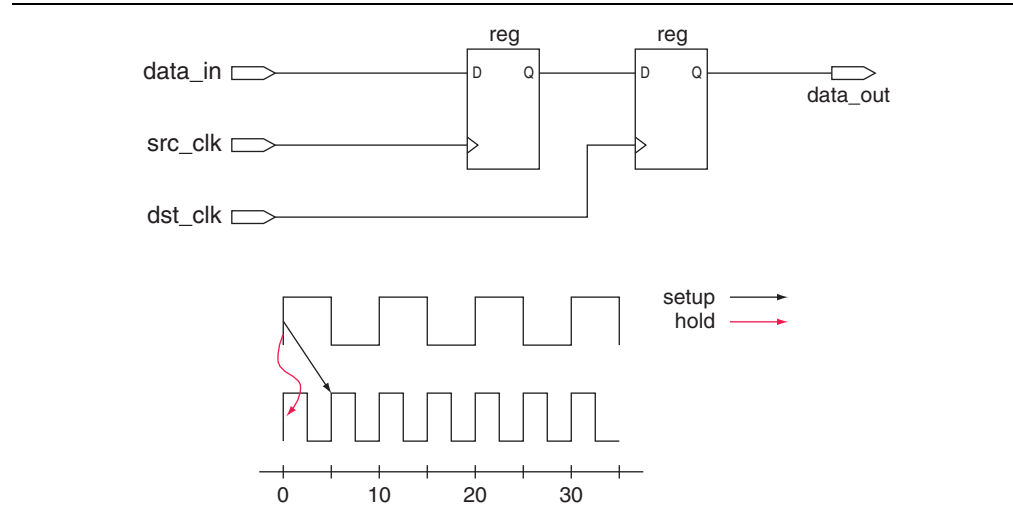


Figure 6-9 shows a register-to-register path used for the default setup and hold relationship, the respective timing diagrams for the source and destination clocks, and the default setup and hold relationships, when the source clock, `src_clk`, has a period of 10 ns and the destination clock, `dst_clk`, has a period of 5 ns. The default setup relationship is 5 ns; the default hold relationship is 0 ns.

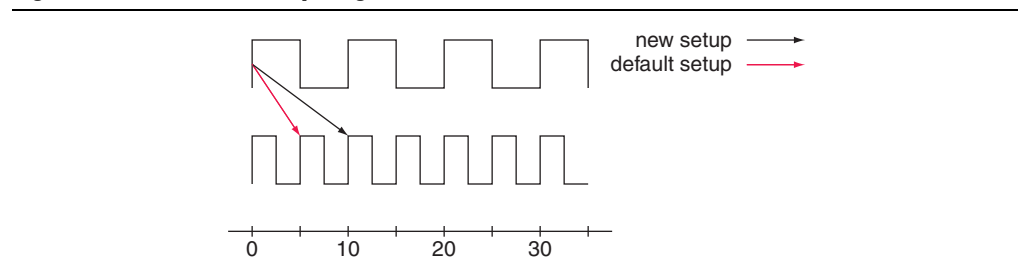
Figure 6-9. Register-to-Register Path and Default Setup and Hold Timing Diagram



To accommodate the system requirements you can modify the default setup and hold relationships with a multicycle timing exception.

Figure 6-10 shows the actual setup relationship after you apply a multicycle timing exception. The exception has a multicycle setup assignment of two to use the second occurring latch edge; in this example, to 10 ns from the default value of 5 ns.

Figure 6-10. Modified Setup Diagram



For more information about creating exceptions with multicycle paths, refer to *The Quartus II TimeQuest Timing Analyzer* chapter of the *Quartus II Handbook*.

Metastability

Metastability problems can occur when a signal is transferred between circuitry in unrelated or asynchronous clock domains because the designer cannot guarantee that the signal will meet setup and hold time requirements. To minimize the failures due to metastability, circuit designers typically use a sequence of registers, also known as a synchronization register chain, or synchronizer, in the destination clock domain to resynchronize the data signals to the new clock domain.

The mean time between failures (MTBF) is an estimate of the average time between instances of failure due to metastability.

The TimeQuest analyzer analyzes the potential for metastability in your design and can calculate the MTBF for synchronization register chains. The MTBF of the entire design is then estimated based on the synchronization chains it contains.

In addition to reporting synchronization register chains found in the design, the Quartus II software also protects these registers from optimizations that might negatively impact MTBF, such as register duplication and logic retiming. The Quartus II software can also optimize the MTBF of your design if the MTBF is too low.

For more information about metastability, its effects in FPGAs, and how MTBF is calculated, refer to the *Understanding Metastability in FPGAs* white paper. For more information about metastability analysis, reporting, and optimization features in the Quartus II software, refer to the *Managing Metastability with the Quartus II Software* chapter in volume 1 of the *Quartus II Handbook*.

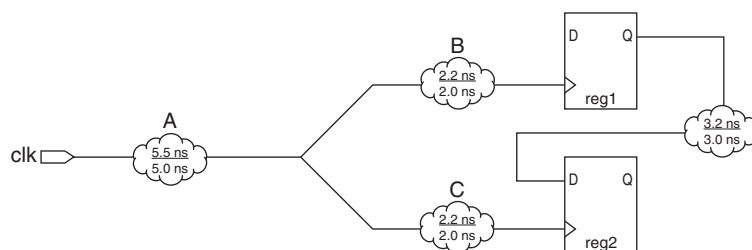
Common Clock Path Pessimism Removal

Common clock path pessimism removal accounts for the minimum and maximum delay variation associated with common clock paths during static timing analysis by adding the difference between the maximum and minimum delay value of the common clock path to the appropriate slack equation.

Minimum and maximum delay variation can occur when two different delay values are used for the same clock path. For example, in a simple setup analysis, the maximum clock path delay to the source register is used to determine the data arrival time. The minimum clock path delay to the destination register is used to determine the data required time. However, if the clock path to the source register and to the destination register share a common clock path, both the maximum delay and the minimum delay are used to model the common clock path during timing analysis. The use of both the minimum delay and maximum delay results in an overly pessimistic analysis since two different delay values, the maximum and minimum delays, cannot be used to model the same clock path.

Figure 6-11 shows a typical register-to-register path with the maximum and minimum delay values shown.

Figure 6-11. Common Clock Path



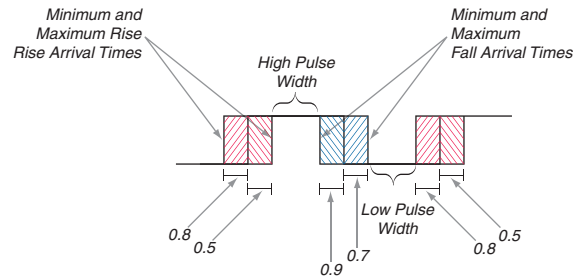
Segment A is the common clock path between reg1 and reg2. The minimum delay is 5.0 ns; the maximum delay is 5.5 ns. The difference between the maximum and minimum delay value equals the common clock path pessimism removal value; in this case, the common clock path pessimism is 0.5 ns. The TimeQuest analyzer adds the common clock path pessimism removal value to the appropriate slack equation to determine overall slack. Therefore, if the setup slack for the register-to-register path in Figure 6-11 equals 0.7 ns without common clock path pessimism removal, the slack would be 1.2 ns with common clock path pessimism removal.

You can also use common clock path pessimism removal to determine the minimum pulse width of a register. A clock signal must meet a register's minimum pulse width requirement to be recognized by the register. A minimum high time defines the minimum pulse width for a positive-edge triggered register. A minimum low time defines the minimum pulse width for a negative-edge triggered register.

Clock pulses that violate the minimum pulse width of a register prevent data from being latched at the data pin of the register. To calculate the slack of the minimum pulse width, the TimeQuest analyzer subtracts the required minimum pulse width time from the actual minimum pulse width time. The TimeQuest analyzer determines the actual minimum pulse width time by the clock requirement you specified for the clock that feeds the clock port of the register. The TimeQuest analyzer determines the

required minimum pulse width time by the maximum rise, minimum rise, maximum fall, and minimum fall times. Figure 6-12 shows a diagram of the required minimum pulse width time for both the high pulse and low pulse.

Figure 6-12. Required Minimum Pulse Width



With common clock path pessimism, the minimum pulse width slack can be increased by the smallest value of either the maximum rise time minus the minimum rise time, or the maximum fall time minus the minimum fall time. For Figure 6-12, the slack value can be increased by 0.2 ns, which is the smallest value between 0.3 ns ($0.8 \text{ ns} - 0.5 \text{ ns}$) and 0.2 ns ($0.9 \text{ ns} - 0.7 \text{ ns}$).

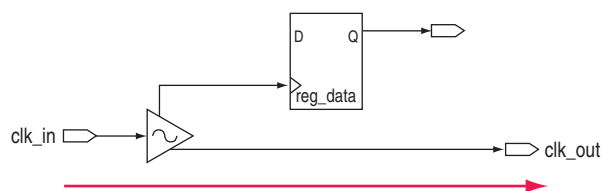
- For more information, refer to [TimeQuest Timing Analyzer Page \(Settings Dialog Box\)](#) in Quartus II Help.

Clock-As-Data Analysis

The majority of FPGA designs contain simple connections between any two nodes known as either a data path or a clock path. A data path is a connection between the output of a synchronous element to the input of another synchronous element. A clock is a connection to the clock pin of a synchronous element. However, for more complex FPGA designs, such as designs that use source-synchronous interfaces, this simplified view is no longer sufficient. Clock-as-data analysis is performed in circuits with elements such as clock dividers and DDR source-synchronous outputs.

The connection between the input clock port and output clock port can be treated either as a clock path or a data path. Figure 6-13 shows a design where the path from port `clk_in` to port `clk_out` is both a clock and a data path. The clock path is from the port `clk_in` to the register `reg_data` clock pin. The data path is from port `clk_in` to the port `clk_out`.

Figure 6-13. Simplified Source Synchronous Output

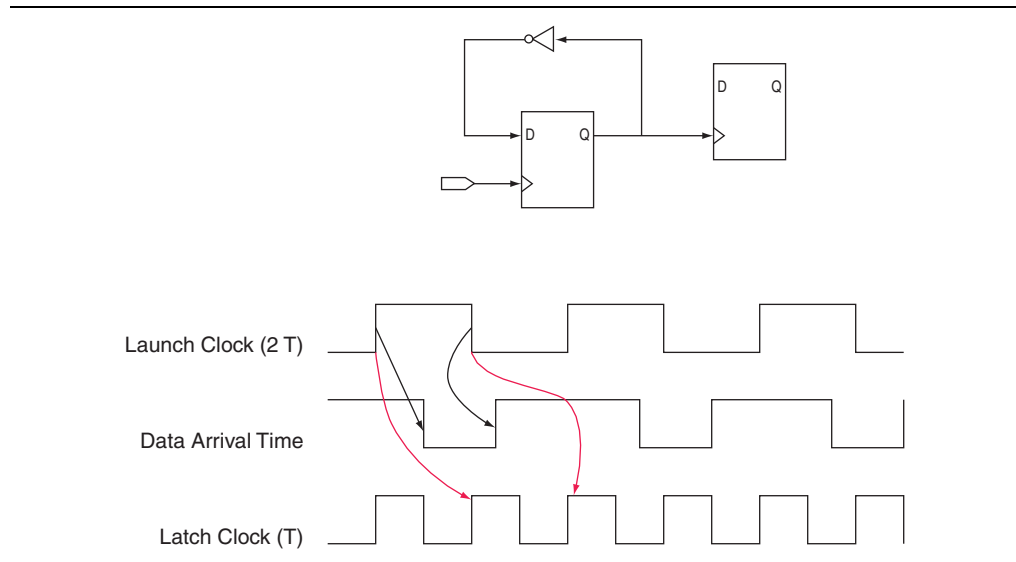


With clock-as-data analysis, the TimeQuest analyzer provides a more accurate analysis of the path based on user constraints. For the clock path analysis, any phase shift associated with the phase-locked loop (PLL) is taken into consideration. For the data path analysis, any phase shift associated with the PLL is taken into consideration rather than ignored.

The clock-as-data analysis also applies to internally generated clock dividers.

Figure 6-14 shows an internally generated clock divider. In this figure, waveforms are for the inverter feedback path, analyzed during timing analysis. The output of the divider register is used to determine the launch time and the clock port of the register is used to determine the latch time.

Figure 6-14. Clock Divider



Multicorner Analysis

The TimeQuest analyzer performs multicorner timing analysis to verify your design under a variety of operating conditions—such as voltage, process, and temperature—while performing static timing analysis.

To change the operating conditions or speed grade of the device used for timing analysis, use the `set_operating_conditions` command.

- ❓ For more information about the `set_operating_conditions` and `get_available_operating_conditions` commands—including full syntax information, options, and example usage—refer to [set_operating_conditions](#) and [get_available_operating_conditions](#) in Quartus II Help.

If you specify an operating condition Tcl object, the `-model`, `speed`, `-temperature`, and `-voltage` options are optional. If you do not specify an operating condition Tcl object, the `-model` option is required; the `-speed`, `-temperature`, and `-voltage` options are optional.



To obtain a list of available operating conditions for the target device, use the `get_available_operating_conditions -all` command.

To ensure that no violations occur under various conditions during the device operation, perform static timing analysis under all available operating conditions. [Table 6-2](#) shows the operating conditions for the slow and fast timing models for device families that support only slow and fast operating conditions.

Table 6-2. Operating Conditions for Slow and Fast Models

Model	Speed Grade	Voltage	Temperature
Slow	Slowest speed grade in device density	V_{cc} minimum supply ⁽¹⁾	Maximum T_J ⁽¹⁾
Fast	Fastest speed grade in device density	V_{cc} maximum supply ⁽¹⁾	Minimum T_J ⁽¹⁾

Note to Table 6-2:

(1) Refer to the DC & Switching Characteristics chapter of the applicable device Handbook for V_{cc} and T_J values

[Example 6-1](#) shows how to set the operating conditions in [Example 6-2](#) with a Tcl object.

Example 6-1. Setting Operating Conditions with a Tcl Object

```
set_operating_conditions 3_slow_1100mv_85c
```

[Example 6-2](#) shows how to set the operating conditions for a Stratix III design to the slow timing model, with a voltage of 1100 mV, and temperature of 85° C.

Example 6-2. Setting Operating Conditions with Individual Values

```
set_operating_conditions -model slow -temperature 85 -voltage 1100
```

Example 6-3 shows how to use the `set_operating_conditions` command to generate different reports for various operating conditions.

Example 6-3. Script Excerpt for Analysis of Various Operating Conditions

```
#Specify initial operating conditions
set_operating_conditions -model slow -speed 3 -grade c -temperature 85
-voltage 1100

#Update the timing netlist with the initial conditions
update_timing_netlist

#Perform reporting

#Change initial operating conditions. Use a temperature of 0C
set_operating_conditions -model slow -speed 3 -grade c -temperature 0
-voltage 1100

#Update the timing netlist with the new operating condition
update_timing_netlist

#Perform reporting

#Change initial operating conditions. Use a temperature of 0C and a
model of fast
set_operating_conditions -model fast -speed 3 -grade c -temperature 0
-voltage 1100

#Update the timing netlist with the new operating condition
update_timing_netlist


#Perform reporting
```

Document Revision History

Table 6-3 shows the revision history for this document.

Table 6-3. Document Revision History

Date	Version	Changes
June 2012	12.0.0	Added social networking icons, minor text updates
November 2011	11.1.0	Initial release.

 For previous versions of the *Quartus II Handbook*, refer to the [Quartus II Handbook Archive](#).

