

Introduction

As designs become more complex, the need for advanced timing analysis capability grows. Static timing analysis is a method of analyzing, debugging and validating the timing performance of a design. Timing analysis measures the delay of every design path and reports the performance of the design in terms of the maximum clock speed. Static timing analysis does not check design functionality and should be used together with simulation to verify the overall design operation.

The Quartus™ II software provides the features necessary to perform advanced timing analysis for today's system-on-a-programmable-chip designs. For example, during design compilation, the Quartus II software can automatically activate the static timing analyzer, removing the need to launch a separate timing analysis tool after each successful compilation. The results reported by the Quartus II Static Timing Analyzer are displayed as several separate tables and provide immediate and direct access to all timing analysis results.

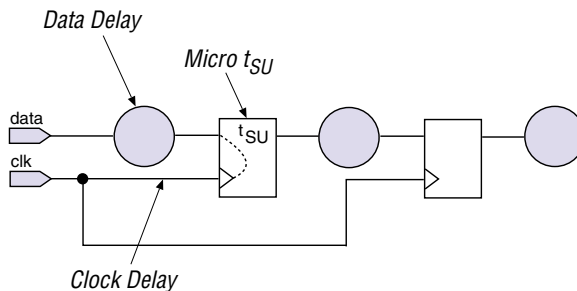
This application note explains the basic principles of static timing analysis and the advanced features supported by the Quartus II Static Timing Analyzer.

Timing Analysis Basics

A comprehensive timing analysis involves observing the setup times, hold times, clock-to-output delays, maximum clock frequencies, and slack times for the design. Obtaining and analyzing this information allows designers to validate circuit performance and to identify possible timing violations. Undetected timing violations could result in incorrect circuit operation. This section describes basic timing analysis measurements.

Clock Setup Time (t_{SU})

Data that feeds a register via its data or enable input(s) must arrive at the input pin before the register's clock signal is asserted at the clock pin. Clock setup time is the minimum length of time that this data must arrive before the active clock edge. [Figure 1](#) shows a diagram of clock setup time.

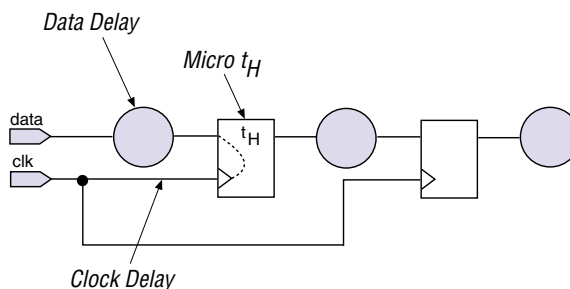
Figure 1. Clock Setup Time (t_{SU})

Micro t_{SU} is the internal setup time of the register (i.e., it is a characteristic of the register and is unaffected by the signals feeding the register). The following equation calculates the t_{SU} of the circuit shown in Figure 1.

$$t_{SU} = \text{Data Delay} - \text{Clock Delay} + \text{Micro } t_{SU}$$

Clock Hold Time (t_H)

Data that feeds a register via its data or enable input(s) must be held at an input pin after the register's clock signal is asserted at the clock pin. Clock hold time is the minimum length of time that this data must be stable after the active clock edge. Figure 2 shows a diagram of clock hold time.

Figure 2. Clock Hold Time (t_H)

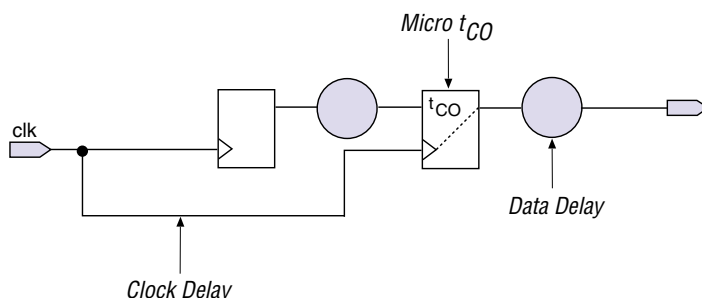
Micro t_H is the internal hold time of the register. The following equation calculates the t_H of the circuit shown in Figure 2.

$$t_H = \text{Clock Delay} - \text{Data Delay} + \text{Micro } t_H$$

Clock-to-Output Delay (t_{CO})

Clock-to-output delay is the time to obtain a valid output at an output pin fed by a register, after a clock transition on the input pin that clocks the register. Micro t_{CO} is the internal clock-to-output delay of the register. Figure 3 shows a diagram of clock-to-output delay.

Figure 3. Clock-to-Output Delay (t_{CO})



The following equation calculates the t_{CO} of the circuit shown in Figure 3.

$$t_{CO} = \text{Clock Delay} + \text{Micro } t_{CO} + \text{Data Delay}$$

Maximum Clock Frequency (f_{MAX})

Maximum clock frequency is the fastest speed that the design clock can run without violating internal setup and hold time requirements. The Quartus II software can perform timing analysis on both single and multiple clock designs, reporting a design's internal f_{MAX} and the system f_{MAX} . An internal f_{MAX} analysis calculates the register-to-register timing within the device. The system f_{MAX} models external delays to/from the device and includes on chip (t_{SU}) and off chip (t_{CO}) delays.

Pin to Pin Delay (t_{PD})

Pin to pin delay (t_{PD}) is the time required for an input pin signal to propagate through combinatorial logic and appear at an external output pin.

Slack

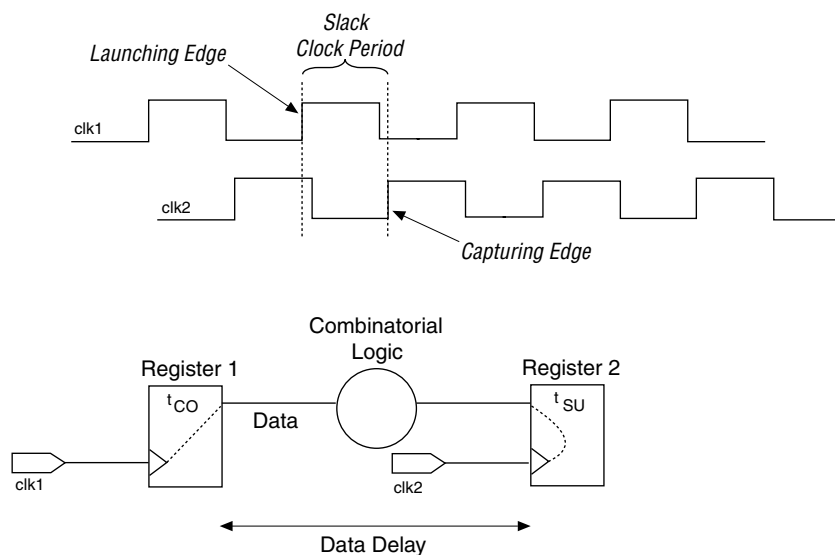
Slack is the margin by which a timing requirement (e.g., f_{MAX}) is met or not met. Positive slack indicates that the circuit meets the timing requirements; negative slack indicates that the design does not meet the timing requirements. The Quartus II software determines slack with the following equations.

$$\text{Slack} = \text{Required clock period} - \text{Actual clock period}$$

$$\text{Slack} = \text{Slack clock period} - (\text{Micro } t_{CO} + \text{Data Delay} + \text{Micro } t_{SU})$$

Figure 4 shows a slack calculation diagram.

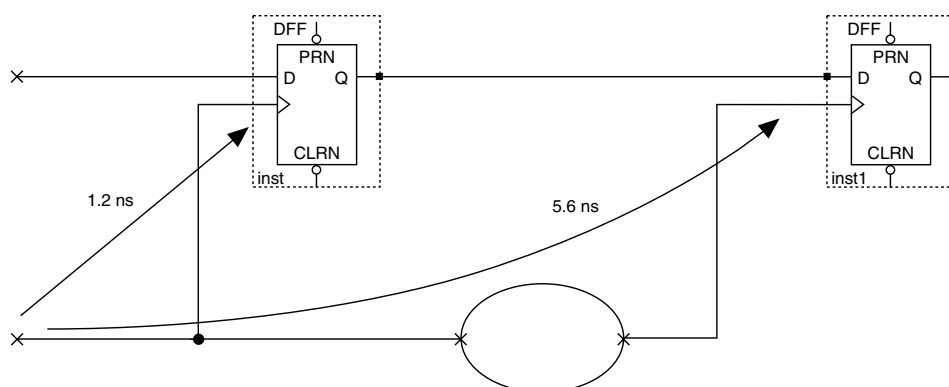
Figure 4. Slack Calculation Diagram



Clock Skew

Clock skew is the difference in arrival time of a clock signal at two different registers. Clock skew occurs when two clock signal paths have different lengths. Clock skew is common in designs that contain clock signals that are not routed globally. The Quartus II software reports clock skew for all clocks within the design.

Figure 5. Clock Skew



Setting up the Timing Analyzer

Timing analysis can be run automatically, as part of a Quartus II compilation by selecting the **Run Timing Analysis** check box from the **Verification** tab in the **Compiler Settings** (Processing menu). Timing analysis can also be run independent of a compilation by selecting **Start Timing Analysis** (Processing menu). In order for timing analysis to be performed, the design must have been previously compiled successfully. However, multiple timing analysis compilations can be performed with different timing analysis settings, without having to perform a complete design re-compile.

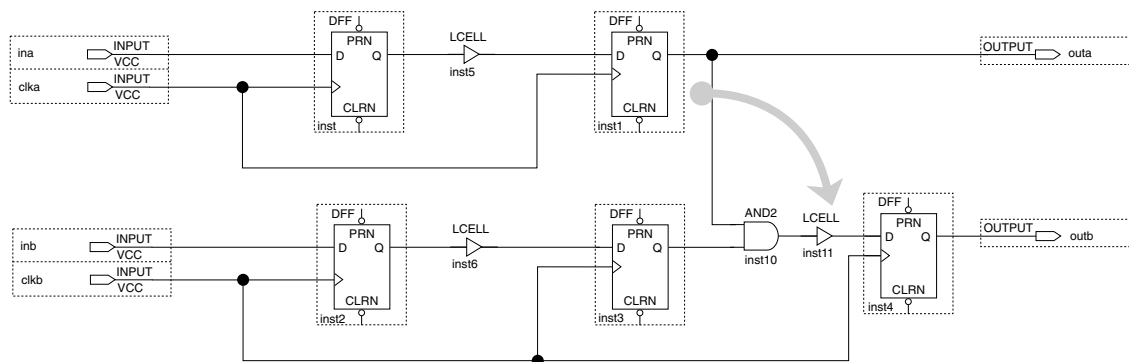
Timing assignments can be set globally for the project or they can be assigned to individual paths in the design. If a project contains a mixture of global and individual timing assignments, the individual timing assignments will take precedence over the global timing assignments.

Setting Global Timing Assignments

Global timing settings can be set via the **Timing Settings** dialog box (Project Menu). All clocks can be set in the design via the **Clock Settings** tab or for all I/O pins using the **Other Requirements & Options** tab.

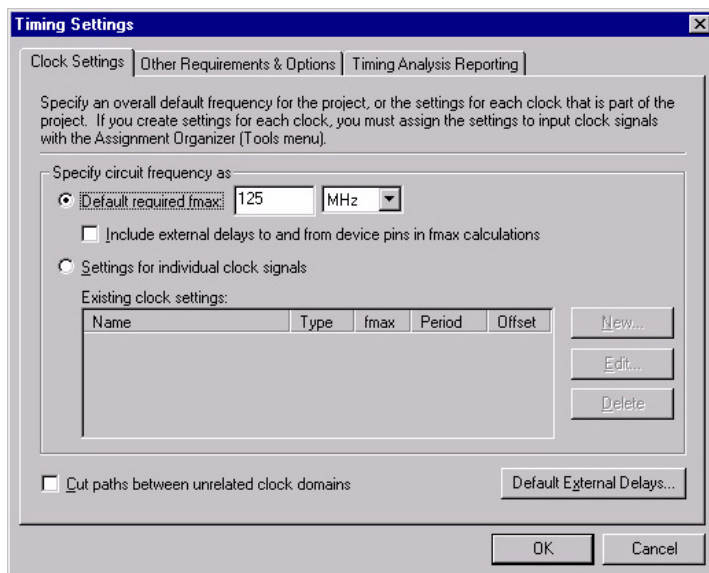
Cut Paths Between Unrelated Clock Domains

By default, the Quartus II software will cut paths between unrelated clock domains when there are no timing requirements set or only the default required f_{MAX} clock setting is used, i.e. the **Cut Timing Paths Between Unrelated Clock Domains** option is enabled by default. This option will also cut paths between unrelated clock domains if individual clock assignments are set but there is no defined relationship between the clock assignments. See [Figure 6](#).

Figure 6. Cut Paths Between Unrelated Clock Domains

For the circuit shown in Figure 6, the path between `inst1` and `inst4` will not be measured or reported by the timing analyzer. This operation can be disabled by unselecting the **Cut Timing Paths Between Unrelated Clock Domains** check box. Disabling this check box allows the Quartus II timing analyzer to include these paths as part of a timing analysis operation. When complex timing assignments exist, the Quartus II timing analyzer will always measure paths between related clock domains.

The global settings shown in Figure 7 allow the user to specify the default f_{MAX} requirement for the design. This value will be applied to all clock inputs in the design. By default, the setting value is applied to the internal f_{MAX} in the design.

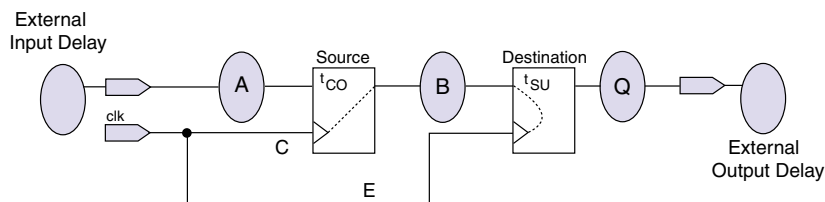
Figure 7. Calculating Internal f_{MAX} 

To determine the internal f_{MAX} of a design, it is necessary to calculate the circuit's clock period. The clock period is dependent on the data path delay, clock skew between registers, the source register's internal clock-to-output time, and the destination register's setup time. The Quartus II software uses the following equations to calculate clock period and internal f_{MAX} .

Register-to-register delay (t_{RD}) in the clock period equation represents the data path delay between two registers. Figure 8 shows a sample internal f_{MAX} diagram.

$$\text{Clock period} = t_{RD} - \text{Clock Skew} + \text{Micro } t_{CO} + \text{Micro } t_{SU}$$

$$\text{Internal } f_{MAX} = 1 / \text{Clock Period}$$

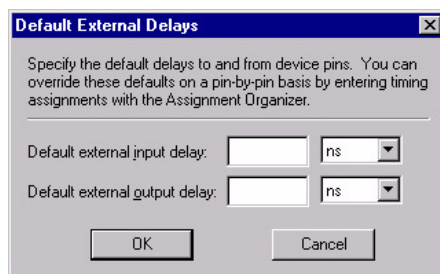
Figure 8. Quartus II Internal f_{MAX} Diagram (Not Including External Delays)

The following equation calculates the internal f_{MAX} for the circuit shown in Figure 8.

$$\text{Internal } f_{MAX} = 1 / [B - (E - C) + \text{Source Micro } t_{CO} + \text{Destination Micro } t_{SU}]$$

The default external delays can be used to calculate the system f_{MAX} . The system f_{MAX} includes external delays to/from the device and includes on chip (t_{SU}) and off chip (t_{CO}) delays. Figure 9 shows the default external delays.

Figure 9. Default External Delays

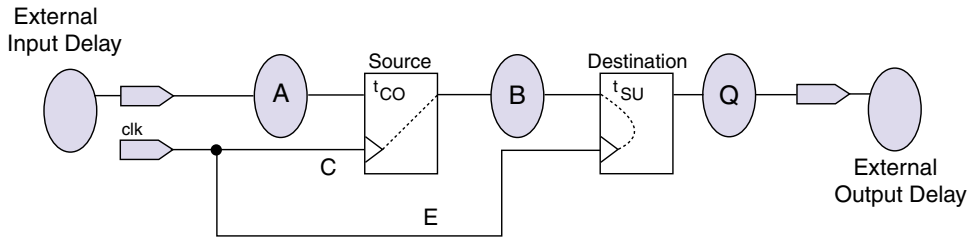


Calculating System f_{MAX}

The Quartus II software calculates the system f_{MAX} by including external delays to the device. This assumes that all input pins are registered just before entering the device, and all output pins are registered just after leaving the device. The designer models the external delays to and from the chip using the External_Input_Delay and External_output_Delay variables. The maximum f_{MAX} is restricted to the slowest clock cycle. Users can configure the external board delays as the t_{CO} of an imaginary input register and the t_{SU} of an imaginary output register. In this scenario, the t_{SU} and t_{CO} values render the external conditions for an accurate system f_{MAX} analysis. The Quartus II software uses the following equation to calculate system f_{MAX} .

$$\text{System } f_{MAX} = 1 / [\text{MAX (Input Clock Period, Clock Period, Output Clock Period)}]$$

Figure 10 shows a sample system f_{MAX} diagram.

Figure 10. Quartus II System f_{MAX} (Including External Delays)

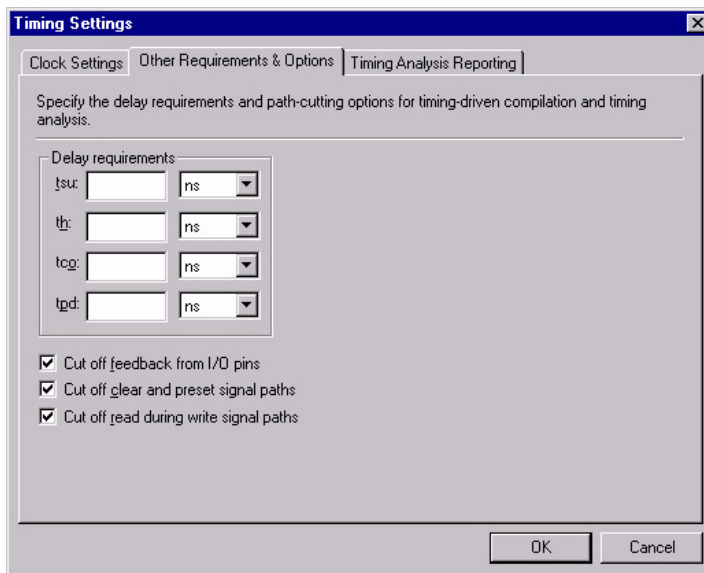
The following equations calculate the system f_{MAX} for the circuit shown in Figure 10.

$$t_{SU} \text{ (source)} = \text{Input Clock Period} = \text{External Input Delay} + A - C + \text{Micro } t_{SU}$$

$$t_{CO} \text{ (destination)} = \text{Output Clock Period} = E + \text{Micro } t_{CO} + Q + \text{External Output Delay}$$

Other Requirements & Options

It is possible to set the default t_{SU} for all input pins, t_H for all input pins, t_{CO} for all output pins and t_{PD} for the project. Figure 11 shows the **Timing Settings** dialog box.

Figure 11. Timing Settings

These assignments are applied globally to the project.

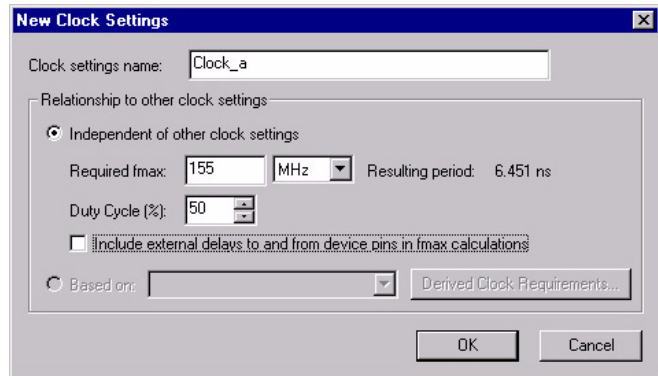
Setting Individual Timing Assignments

Timing requirements can be assigned to individual paths or individual signals.

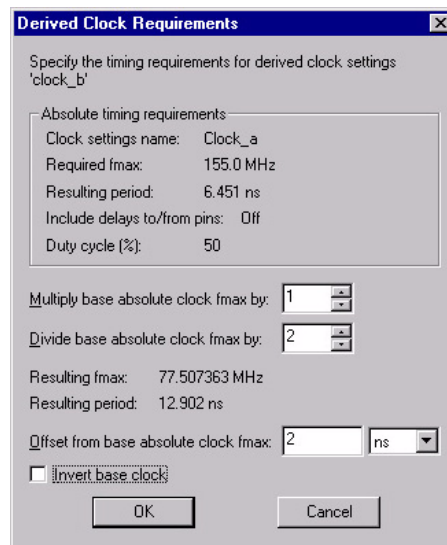
Setting Individual Clock Settings

Clock requirements can be applied to each clock within the design. These assignments can be made for independent clocks or for related clocks.

Independent or absolute clocks have no defined relationship. In order to create an independent clock assignment, the user must enter the desired f_{MAX} or period requirement and the duty cycle, as shown in [Figure 12](#). This clock assignment must then be applied to a signal in the design, usually a pin.

Figure 12. New Clock Settings

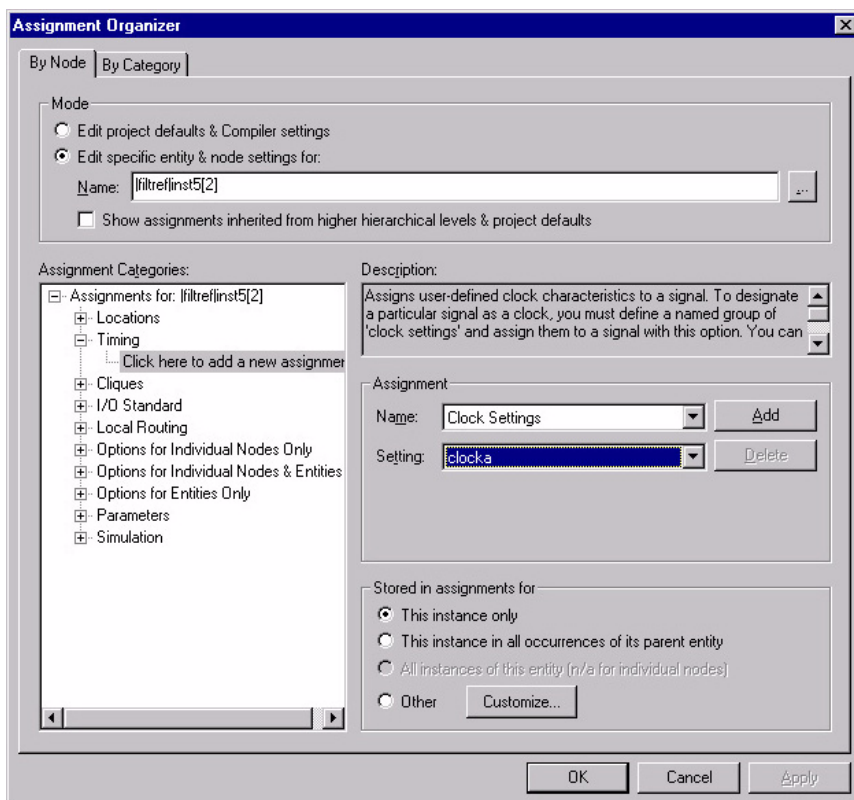
Relative clocks are based upon a previously defined clock. In order to create a relative clock assignment, the user must define the clock requirements based upon a clock previously defined. This involves defining any phase shift using the offset relationship and by defining the frequency requirements using multiply and divide factors. This is shown in Figure 13. This clock assignment must then be applied to a signal in the design, usually a pin. The offset relationship must not be specified for internally generated clocks. The Quartus II software will automatically calculate this offset after the design has been fitted in the selected device.

Figure 13. Derived Clock Requirements Dialog Box

Other Individual Timing Assignments

Individual timing assignments can be set using the Assignment Organizer. The user can select a pin(s) or a node(s) with the node finder and then make individual or point-to-point timing assignments from the timing section. This is shown in [Figure 14](#).

Figure 14. Assignment Organizer



Clock Settings

The clock settings assignment, which can be selected from the timing assignments, is used to assign a previously created individual clock setting to a pin or node in the design.

External Input Delay

This timing assignment is used to specify the delay of a signal from a register in an external device to the selected input pin(s). This value is used to calculate the system f_{MAX} as described in the section on Global Timing Requirements. Individual external input delay assignments take precedence over global external input delay assignments. Figure 15 shows a block diagram of the external input delay.

Figure 15. External Input Delay

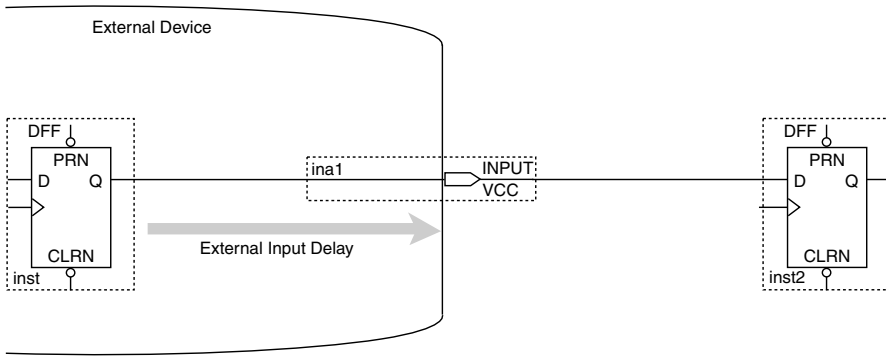
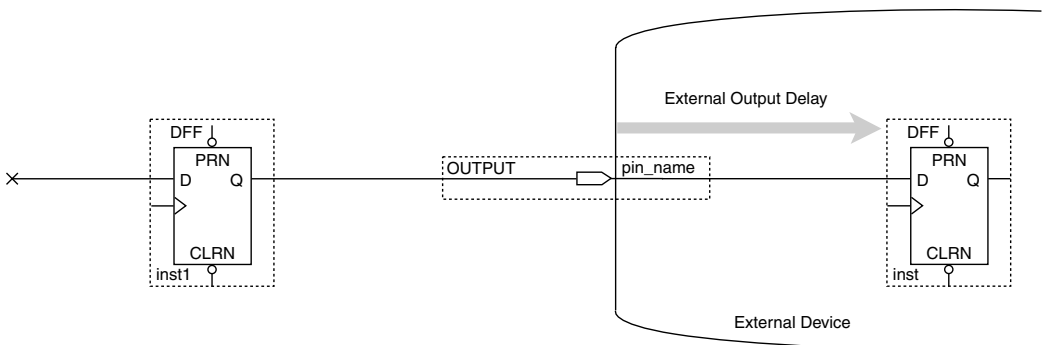


Figure 16 shows a block diagram of the external output delay.

Figure 16. External Output Delay

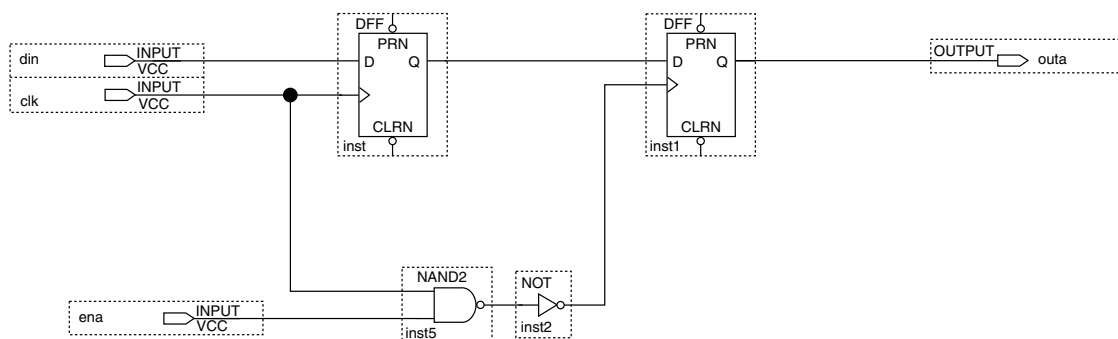


The external output delay timing assignment is used to specify the delay of a signal from an output pin(s) to a register(s) in an external device. This value is used to calculate the system f_{MAX} as described in the section on global timing requirements. Individual external output delay assignments take precedence over global external output delay assignments.

Inverted Clock

The Quartus II timing analyzer, by default, detects registers with inverted clocks and uses the inversion in the timing analysis report. This functionality applies to both clocks that use globals and clocks that do not use globals. However, the timing analyzer may fail to automatically detect inverted clocks when the inversion is part of a complex logic structure. An example of a complex logic structure is shown in [Figure 17](#).

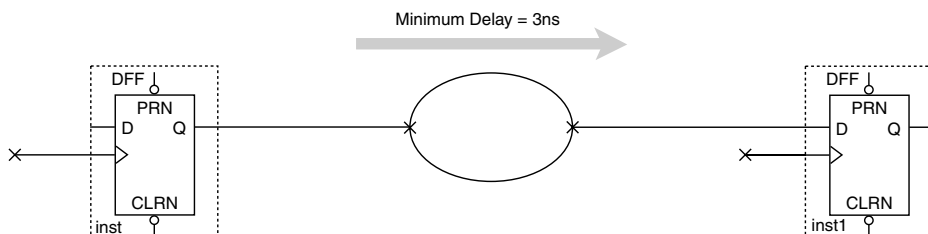
Figure 17. Complex Logic Structure



In the example shown in [Figure 17](#), when the enable is active, the clock is inverted. Under these circumstances, the user can make an inverted clock assignment to the register, `inst1`, to ensure that the timing analyzer recognizes the inverted clock.

Minimum Delay Requirement

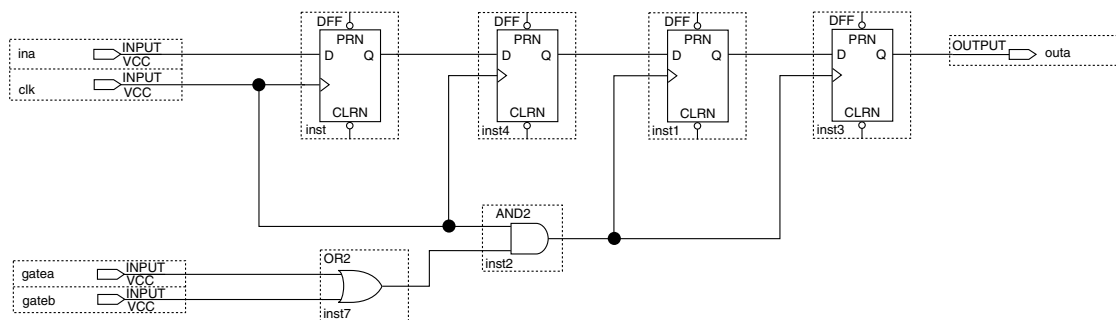
The minimum delay requirement specifies the minimum delay allowed between two points on a path, as shown in [Figure 18](#).

Figure 18. Minimum Delay Requirement

In **Figure 18**, the minimum delay requirement is used to specify that the path between `inst` and `inst1` must be 3 ns or longer. The minimum delay requirement can be specified either between two registers, between a pin and a register, between a register and a pin, or from a pin to a pin with purely combinatorial logic between the pins.

Not a Clock

By default, the timing analyzer automatically identifies any pin that feeds through to the clock input of a register as a clock. An example is shown in **Figure 19**.

Figure 19. Not A Clock Diagram

In **Figure 19**, the timing analyzer identifies three clock pins for the design, `clock`, `gatea` and `gateb`. The pins `gatea` and `gateb` are identified as clock pins because they feed through an or gate and an and gate to the clock inputs of registers `inst1` and `inst3`. If the user does not want to view these pins as clocks, they can be disabled from timing analysis by using the Not A Clock assignment.

t_{CO} Requirement

Individual t_{CO} assignments have priority over global assignments. t_{CO} assignments can be made to either the pin, the output register or from the output register to the pin.

t_H Requirement

Individual t_H assignments have priority over global assignments. t_H assignments can be made to either the pin, the input register, from the pin to the input register or from the clock pin to the input register.

t_{PD} Requirement

Individual t_{PD} assignments have priority over global assignments. t_{PD} assignments can be made from input pins to output pins, from input pins to registers, from registers to registers, from registers to output pins and as a single point assignment to an input pin.

t_{SU} Requirement

Individual t_{SU} assignments have priority over global assignments. t_{SU} assignments can be made to either the input pin, the input register, from the input pin to the input register, or from the clock pin to the input register.

Timing Analysis Reporting in the Quartus II Software

The Quartus II timing analysis report is displayed as a section in the compilation report. The format of the Quartus II timing analysis reporting varies, depending on the timing assignments that are used. If there are no timing assignments for the design or there are only global timing assignments, the timing report will report an f_{MAX} for all clock pins that are automatically detected. The report will also show t_{CO} for all output pins and both t_{SU} and t_H for all input pins. t_{PD} will be reported for any pin-to-pin combinatorial paths in the design. If the design contains complex clock timing assignments, e.g., there are related clocks in the design, the timing analysis report will not display an f_{MAX} report but will provide a slack report as shown in [Figure 20](#).

Figure 20. Slack Timing Analysis Report

Clock Requirement: 'clk' (50.0 MHz)							
Source Name	Destination Name	Source Clock Name	Destination Clock Name	Required Setup Relationship	Required Maximum P2P Time	Actual Maximum P2P Time	Slack
state_m.inst1 filter-15	acc:inst3 accum.inst_1 lpm_add_sub:lpm_add_sub_component addcore:adderja_csnsbuffer:r	clk	clk	20.000 ns	18.554 ns	13.787 ns	4.767 ns
state_m.inst1 filter-16	acc:inst3 accum.inst_1 lpm_add_sub:lpm_add_sub_component addcore:adderja_csnsbuffer:r	clk	clk	20.000 ns	18.554 ns	13.775 ns	4.779 ns
state_m.inst1 filter-17	acc:inst3 accum.inst_1 lpm_add_sub:lpm_add_sub_component addcore:adderja_csnsbuffer:r	clk	clk	20.000 ns	18.554 ns	13.762 ns	4.792 ns
state_m.inst1 filter-15	acc:inst3 accum.inst_1 lpm_add_sub:lpm_add_sub_component addcore:adderja_csnsbuffer:r	clk	clk	20.000 ns	18.554 ns	13.731 ns	4.823 ns
state_m.inst1 filter-16	acc:inst3 accum.inst_1 lpm_add_sub:lpm_add_sub_component addcore:adderja_csnsbuffer:r	clk	clk	20.000 ns	18.554 ns	13.719 ns	4.835 ns
state_m.inst1 filter-17	acc:inst3 accum.inst_1 lpm_add_sub:lpm_add_sub_component addcore:adderja_csnsbuffer:r	clk	clk	20.000 ns	18.554 ns	13.706 ns	4.848 ns
state_m.inst1 filter-15	acc:inst3 accum.inst_1 lpm_add_sub:lpm_add_sub_component addcore:adderja_csnsbuffer:r	clk	clk	20.000 ns	18.554 ns	13.228 ns	5.326 ns
state_m.inst1 filter-16	acc:inst3 accum.inst_1 lpm_add_sub:lpm_add_sub_component addcore:adderja_csnsbuffer:r	clk	clk	20.000 ns	18.554 ns	13.216 ns	5.338 ns
state_m.inst1 filter-17	acc:inst3 accum.inst_1 lpm_add_sub:lpm_add_sub_component addcore:adderja_csnsbuffer:r	clk	clk	20.000 ns	18.554 ns	13.203 ns	5.351 ns
state_m.inst1 filter-15	acc:inst3 accum.inst_1 lpm_add_sub:lpm_add_sub_component addcore:adderja_csnsbuffer:r	clk	clk	20.000 ns	18.554 ns	13.172 ns	5.382 ns

Timing analysis restricted to 10 rows. To change the limit use Timing Settings (Project menu)

A positive slack indicates the margin by which the path surpasses the clock timing requirements. A negative slack indicates the margin by which the path fails the clock timing requirements.

If a design contains individual t_{SU} , t_H or t_{CO} assignments and does not contain global t_{SU} , t_H or t_{CO} assignments, only the individual assignments will be reported in the Timing Analysis Reports. If a design contains individual t_{SU} , t_H or t_{CO} assignments and a timing report is required for t_{SU} , t_H or t_{CO} on all I/O pins, the user must set global t_{SU} , t_H or t_{CO} assignments to generate a timing report on the pins not specified by the individual timing assignments.

Advanced Timing Analysis



The Quartus II software can perform timing analysis of designs containing paths that cross multiple clock domains and designs that contain multicycle paths. This offers designers greater control over design functionality. This section describes these advanced features.

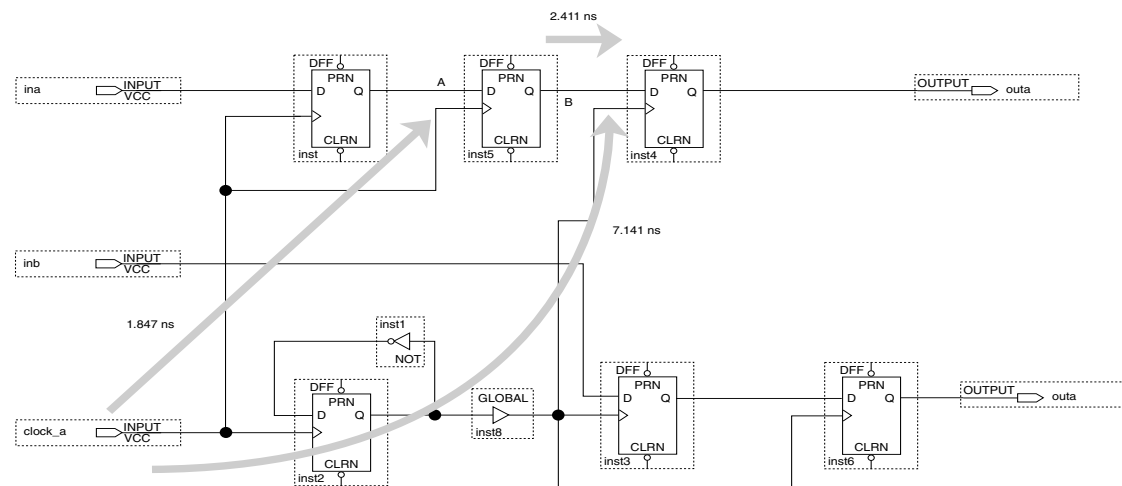
For detailed instruction on how to use these or any of the Quartus II Static Timing Analyzer features, see *Quartus II On-line Help*.

Clock Skew

This section describes some common cases where clock skew will result in incorrect circuit operation.

Derived Clocks

Clock skew error reporting is common in designs containing derived clocks and very short register-to-register data paths. An example is shown in Figure 21.

Figure 21. Derived Clocks Example

In this design, the longest clock path is from `clock_a` to destination register (`inst4`), 7.141 ns. The shortest clock path is from `clock_a` to the source register (`inst5`), 1.847 ns.

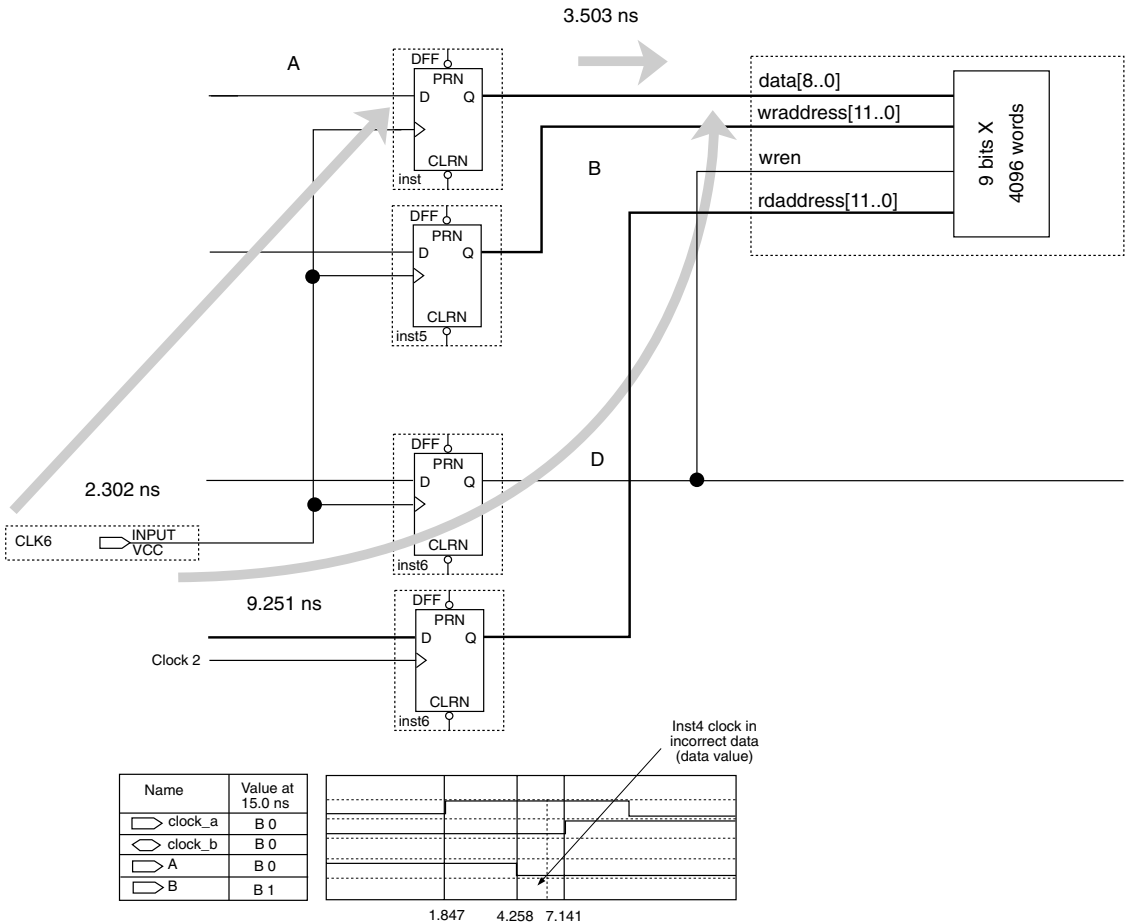
This creates a clock skew of 5.294 ns.

The shortest register-to-register data path between the source and destination register is 2.411 ns. The micro hold delay of the destination register is 0.710 ns. Thus, the clock skew is longer than the data path (5.294 ns > 2.411 ns). This will result in incorrect circuit functionality. To remove the clock skew error, Path B needs to be lengthened such that it is longer than the clock skew. This can be achieved by adding cells to the path or through the placement of the source & destination registers.

Asynchronous Memory

With asynchronous memory, the memory element acts as a latch and the designer must check the setup and hold time on the latch. An example is shown in Figure 22. The longest clock path from `clk6` to destination memory is 9.251 ns. The shortest clock path from `clk6` to source register is 2.302 ns. Thus the largest clock skew is 6.949 ns. The shortest register to memory delay is 3.503 ns and the micro hold delay of the destination register is 0.106 ns. As a result, the clock skew is longer than the data path and the circuit will not operate normally.

Figure 22. Clock Skew



Multiple Clock Domains

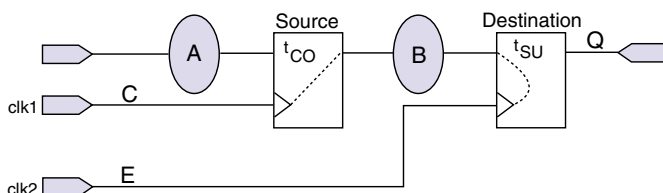
Multiple clock circuits are designs that have more than one clock driving a circuit. After the clocks are specified, the Quartus II software analyzes timing for register-to-register paths controlled by different clocks, and the results are reported as slack. If the clocks are not specified, the Quartus II software reports f_{MAX} for each clock pin that it automatically detects and by default, disregards any paths between unrelated clock domains. The Quartus II timing analyzer can be instructed to perform cross-domain analysis, i.e., include paths between unrelated clock domains. See the Quartus II on-line help for more information.

To assign multiple clocks, the user must define an absolute clock, specify a desired f_{MAX} or clock period, and then define other clocks and their relationship, if any, to the base clock. These settings must then be assigned to the clock pins that supply the design's clock signals. Upon successful compilation, the Quartus II Static Timing Analyzer automatically verifies circuit operability.

Calculation of f_{MAX} for a Design Containing Multiple Clock Domains

Many complex designs contain paths between registers controlled by different clocks. The Quartus II software can determine the f_{MAX} for these multiclock domain designs. Figure 23 shows a sample multiclock internal f_{MAX} diagram.

Figure 23. Multiclock Internal f_{MAX} Calculation Diagram

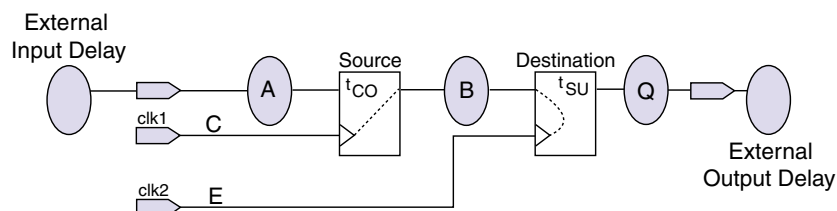


The following equation calculates the internal f_{MAX} for the circuit shown in Figure 23.

$$\text{Internal } f_{MAX} = 1 / [B - (E - C) + \text{Source Micro } t_{CO} + \text{Destination Micro } t_{SU}]$$

Figure 24 shows a sample multiclock system f_{MAX} diagram.

Figure 24. Multiclock System f_{MAX} Calculation Diagram



The following equations calculate the system f_{MAX} for the circuit shown in Figure 24.

$t_{SU}(\text{source}) = \text{Input Clock Period} = \text{External Input Delay} + A - C + \text{Micro } t_{SU}$

$t_{CO}(\text{destination}) = \text{Output Clock Period} = E + \text{Micro } t_{CO} + Q + \text{External Output Delay}$

The Quartus II software allows designers to specify the desired f_{MAX} of both `clk1` and `clk2` for the design. The Quartus II Static Timing Analyzer reports whether these clock periods meet or violate design operability under the specified f_{MAX} values.

Multicycle Paths

Multicycle paths are paths between registers that intentionally require more than one clock cycle to become stable. For example, a register may need to trigger a signal on every second or third rising clock edge.

Figure 25 shows an example of a design with a multicycle path between the multiplier's input registers and output register.

Figure 25. Example Diagram of a Multicycle Path

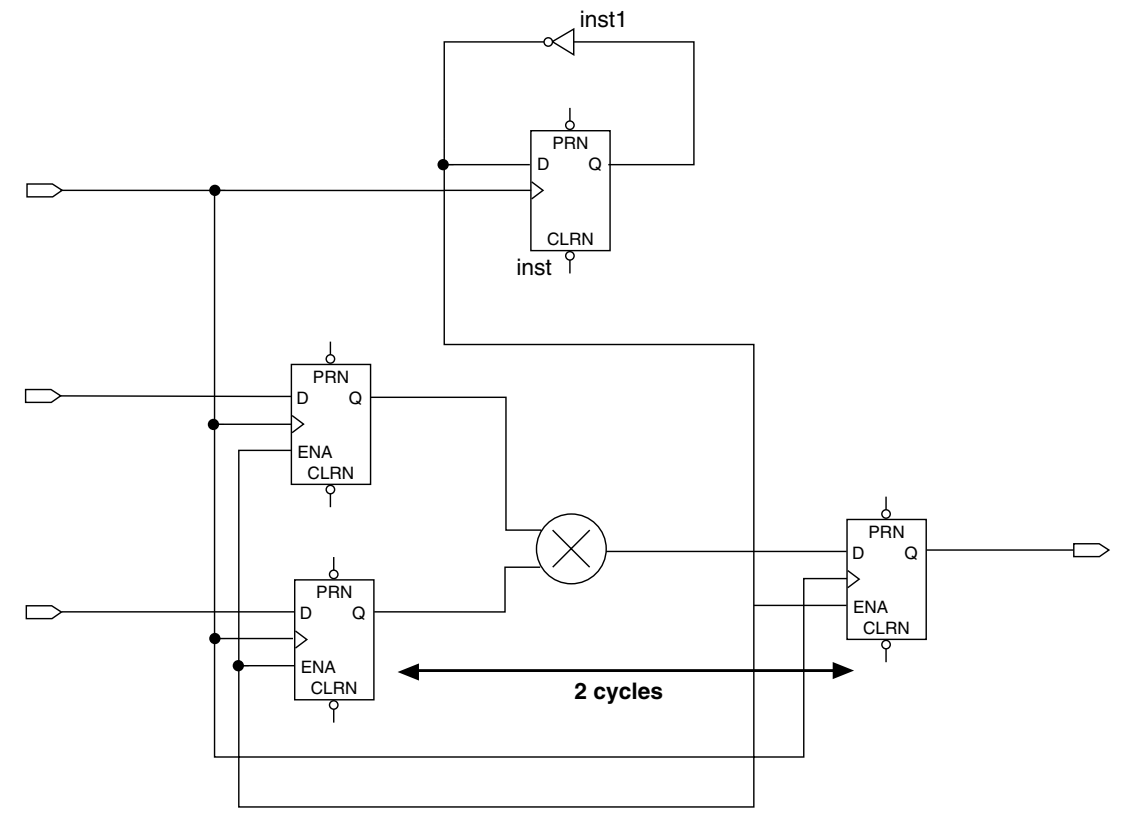
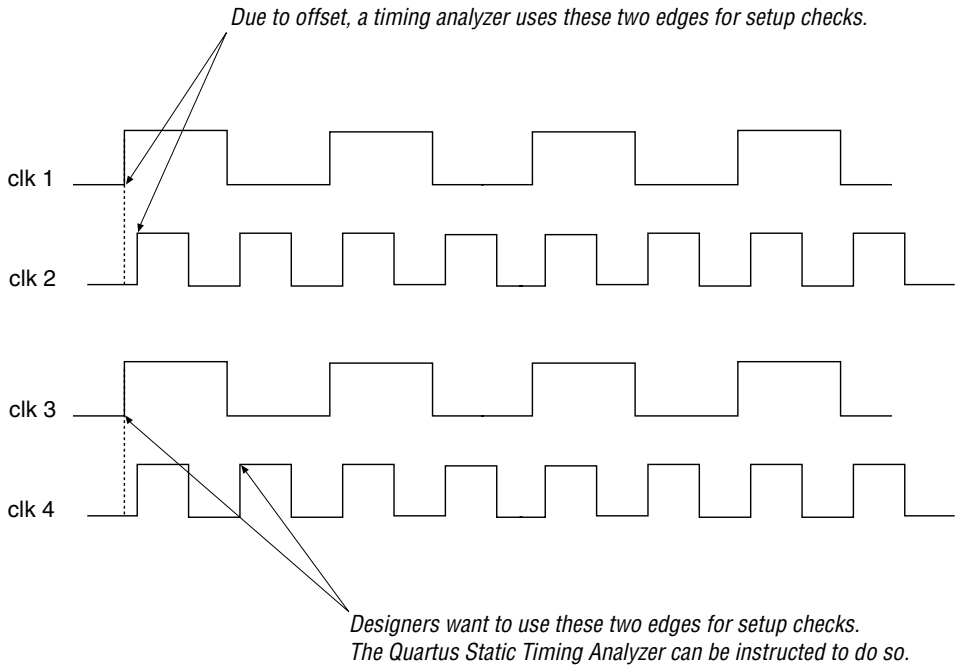


Figure 26 shows a timing diagram for a multicycle path between registers that exists in a design with multiple clocks, with a small offset between the clocks.

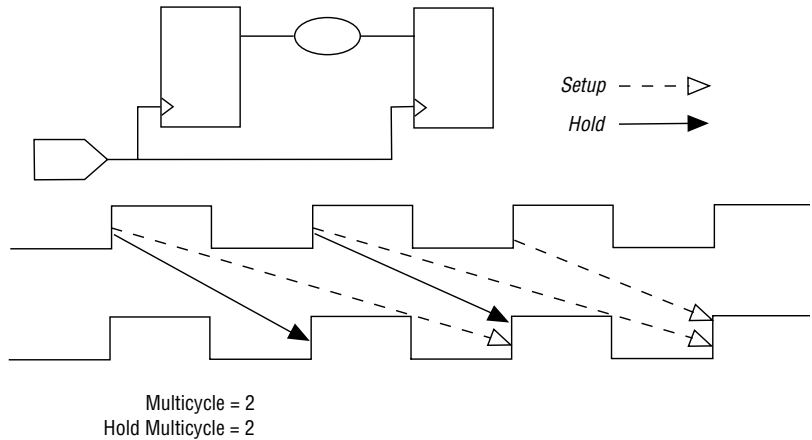
Figure 26. Multicycle Paths with Offset Between Clocks

Designers can set multicycle paths in their designs to instruct the Quartus II Static Timing Analyzer to adjust its measurements, thus avoiding incorrect set-up or hold time violation reports. These assignments can be made in the Quartus II Assignment Organizer.

Hold Multicycle Assignment

A hold multicycle assignment is used to specify the minimum number of clock cycles required before a register should latch a value. If no hold multicycle value is specified, the hold multicycle defaults to the value of the multicycle assignment.

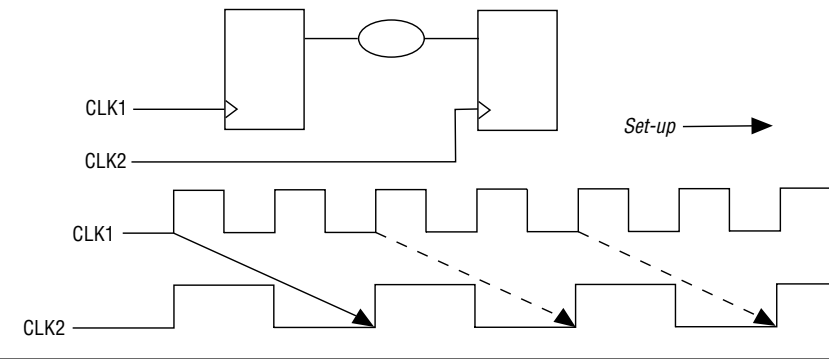
Figure 27. Hold Multicycle Assignment



Source Multicycle Assignment

The source multicycle assignment is useful when the source and destination registers are clocked by separate clocks at different frequencies. It is used to extend the required delay by adding periods of the source clock rather than the destination clock.

Figure 28. Source Multicycle Assignment



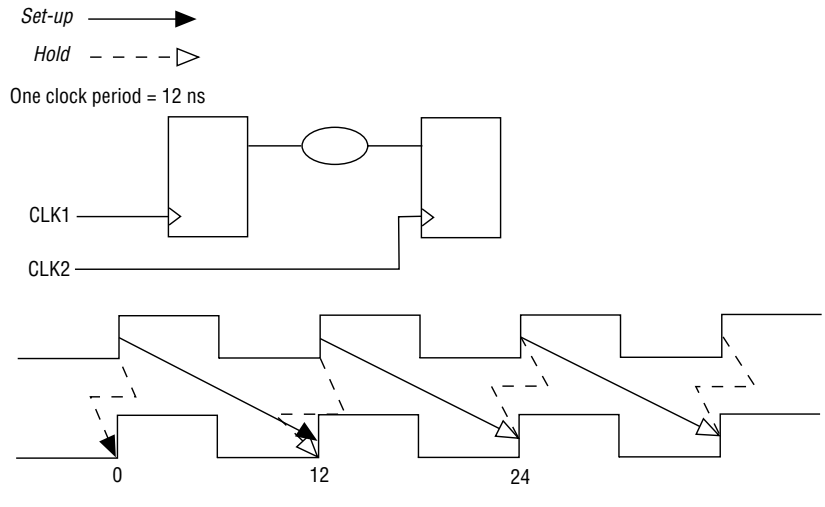
Source Hold Multicycle Assignment

The source hold multicycle assignment is useful when the source and destination registers are clocked by separate clocks at different frequencies. This assignment allows you to increase the required hold delay by adding source clock cycles.

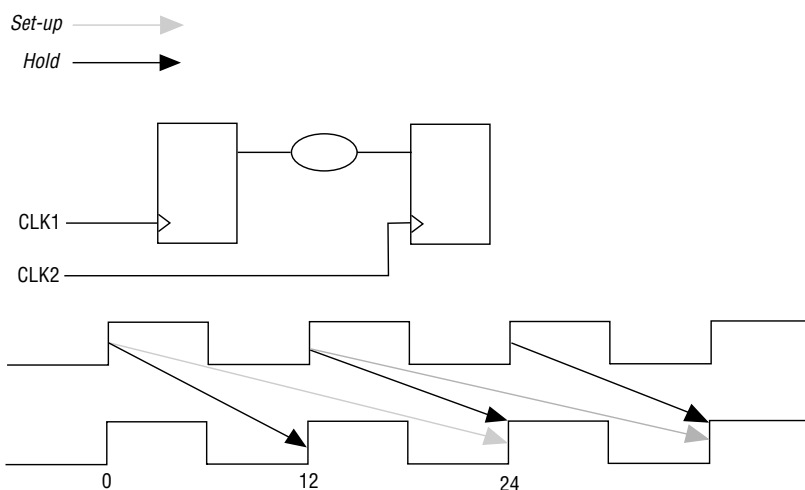
Typical Applications of Multicycle Paths

Figure 29 shows the measurement of t_{SU} and t_H for a standard path.

Figure 29. t_{SU} and t_H Standard Measurement Paths



In the example shown in Figure 29, both `clk1` and `clk2` have the same period and zero offset. In Figure 29, where the clocks have a period of 12 ns, the data delay between the source and destination registers must be between 0 and 12 ns, in order for the circuit to operate. If the data delay is longer than one clock period and the circuit is intended to operate as a multicycle circuit, it is necessary to set a `multicycle = 2` assignment. If `multicycle = 2`, `hold multicycle = 2`, by default and does not need to be set. Figure 30 shows how timing analysis is measured with these assignments.

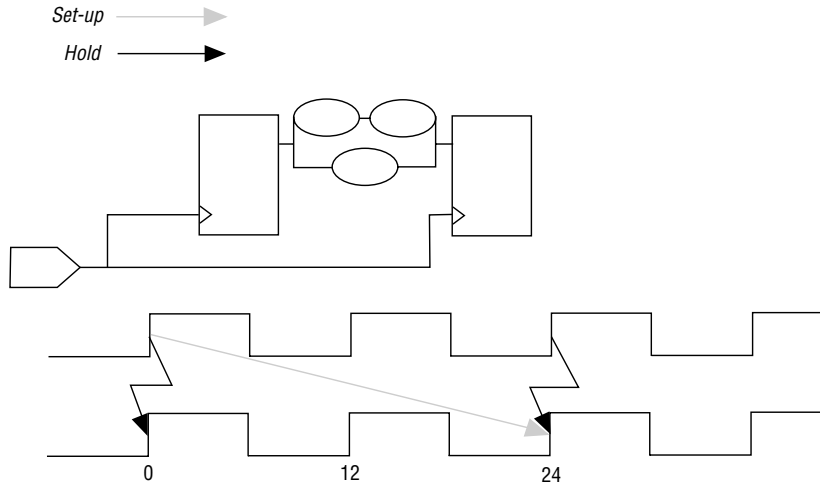
Figure 30. Timing Analysis

One clock period = 12 ns, MULTICYCLE = 2, HOLD_MULTICYCLE = 2

In **Figure 30**, the data delay between the two registers is longer than one clock cycle, but is less than two clock cycles. This circuit requires two clock cycles for a change at the input of the source register to appear at the destination register. The t_{SU} check on `clk2` is performed every second clock period (at 24 ns) and t_H is performed every second period (at 12 ns). This analysis ensures that the data delay is between 12 ns and 24 ns. The minimum data delay is 12 ns and the maximum delay is 24 ns.

Figure 31 illustrates a design that has two data paths between the registers. One data delay is shorter than one clock period and the other data delay is longer than one clock period but shorter than two clock periods. The circuit is intended to operate as a multicycle path.

One clock, period = 12 ns, MULTICYCLE = 2, HOLD_MULTICYCLE = 1

Figure 31. Data Path Delay Example

In [Figure 31](#), the circuit is intended to operate as a multicycle path of two, however one of the data paths between the registers is less than one clock cycle. The designer can ignore the invalid output that occurs on the second clock edge, at 12 ns, and the designer will not change the input on the source register at that input. The source input will only change every two clock cycles. This functionality is usually achieved by using enables on the registers.

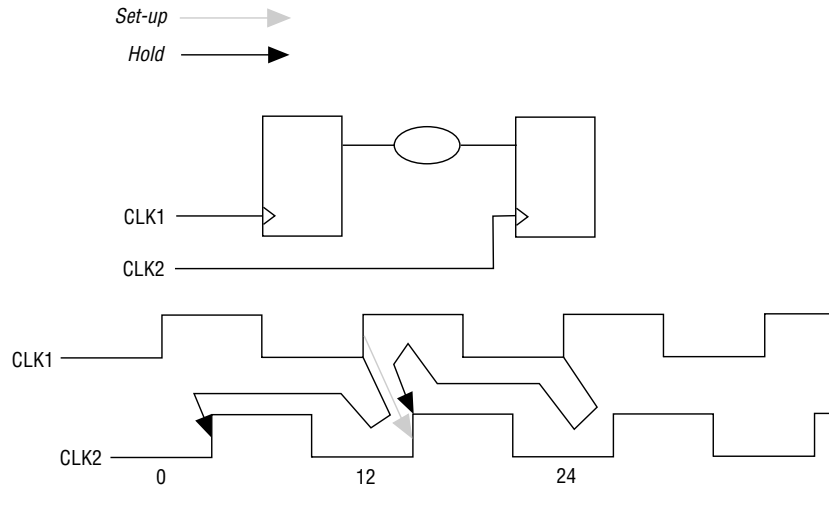
t_{SU} is measured on every second clock edge and t_H is measured on the latching edge. The data delay must be between 0 ns and 24 ns for circuit operation.

Multicycle Paths with Offsets

In the example shown in [Figure 32](#), `clk2` is offset from `clk1` by 2 ns.

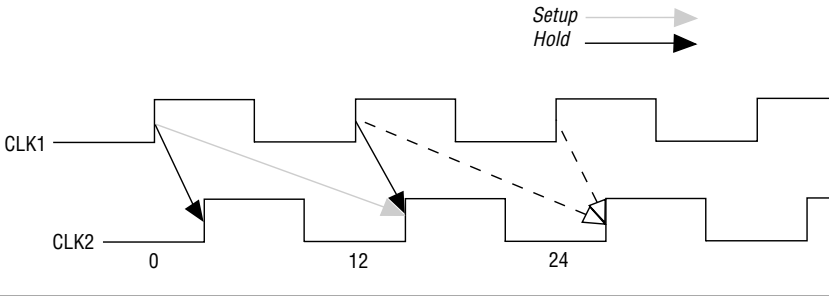
Period = 12 ns, offset = 2 ns

Figure 32. Multicycle Paths with Offsets



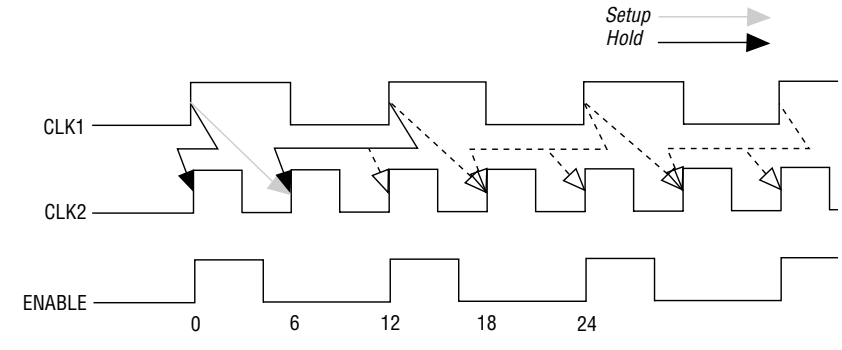
The setup time for `clk2` is 2 ns and the hold time is -10 ns. Therefore the data delay must be between -10 ns and 2 ns. It is unlikely that the design is intended to latch the delay within 2 ns but is intended to latch the data on the second `clk2` edge, i.e., operate as a multicycle path of 2. If the designer sets a `multicycle=2` and `multicycle hold=2`, the set-up requirements are 14 ns and the hold requirements are 2 ns, as shown in Figure 33. The circuit will operate as a multicycle path of two, providing the data delay between the registers is between 2 ns and 14 ns.

Figure 33. Hold Requirements

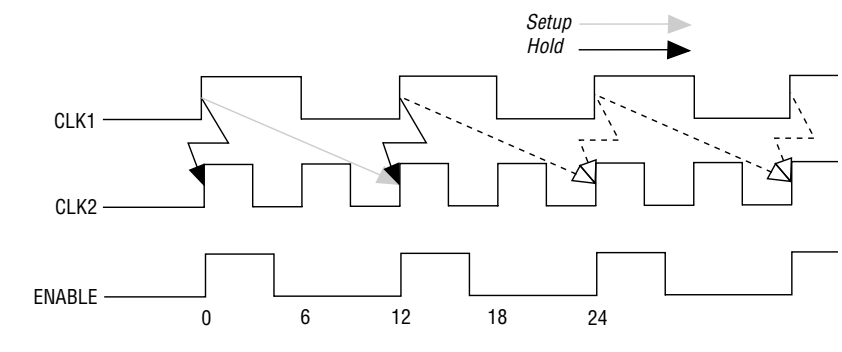


Destination Clock is Faster than Source Clock

The waveform in Figure 34 is the same design as shown in Figure 31. The only difference is that `clk2` operates at 2X the frequency of `clk1`.

Figure 34. Destination Clock Example

By default, the timing analyzer shows a set up time of 6 ns and hold time of 0 ns. However, the intended operation is that the clock enable enables the destination register on every second clock cycle which means multicycle is set at two. The circuit has a data path that has a delay of less than one destination clock cycle but the data available after one clock cycle at 6 ns, is not clocked into the destination register because of the enable operation and should be ignored for hold time measurement. This is done by setting multicycle hold to one. The results in the correct operation are shown in [Figure 35](#).

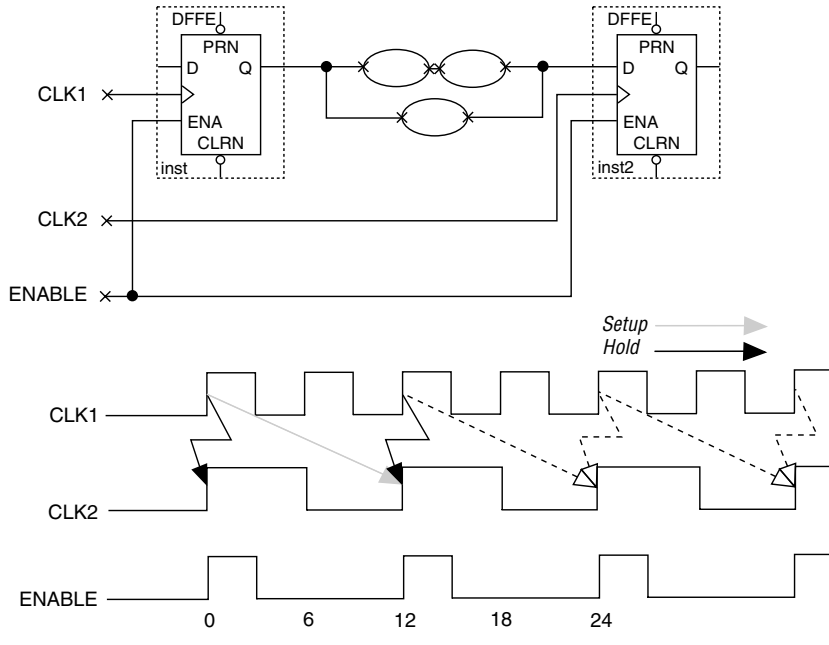
Figure 35. Results

Setup is 12 ns and hold is 0 ns, so providing the data path delays are between 0 ns and 12 ns, the circuit will operate correctly.

Destination Clock is Slower than the Source Clock

In the example shown in [Figure 36](#), the destination clock operates at 2x the frequency of the source clock.

Figure 36. Destination Clock Example



The operation of the clock enable is setup so that the data on the source is not clocked into the source register on its second clock cycle, but the data on the destination register is clocked in on its first clock cycle. This setup requires a source multicycle of 2, which gives a setup requirement of 12 ns. The hold multicycle must be set to one in order to measure the hold on the valid data change at the source, i.e. a source hold multicycle of one.

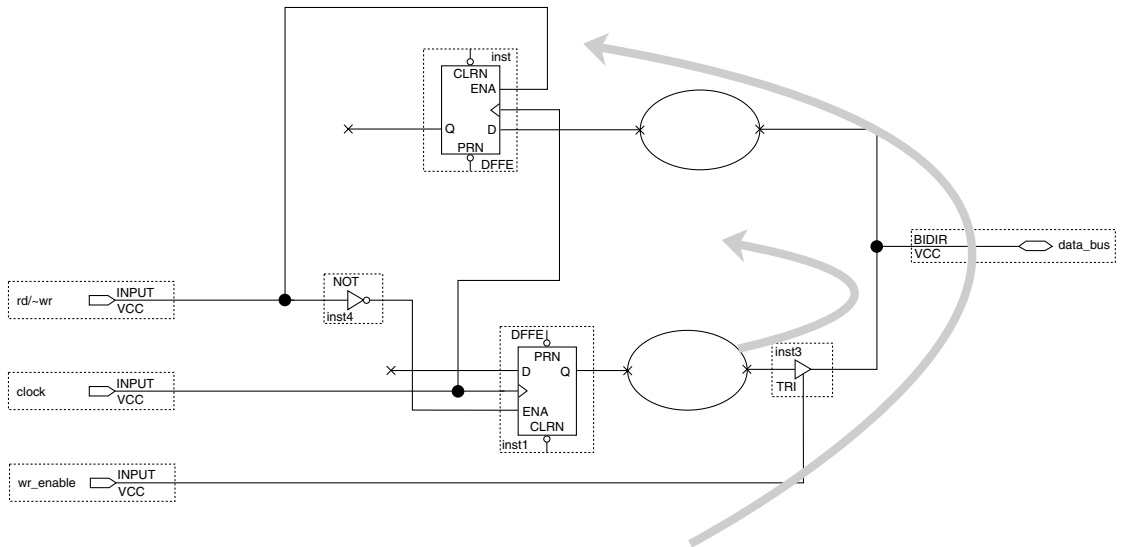
False Paths

A false path is any path that is not relevant to a circuit's operation.

Cut off Feedback from I/O Pins

This option, which is on by default, cuts off feedback paths from the I/O pins as shown in [Figure 37](#).

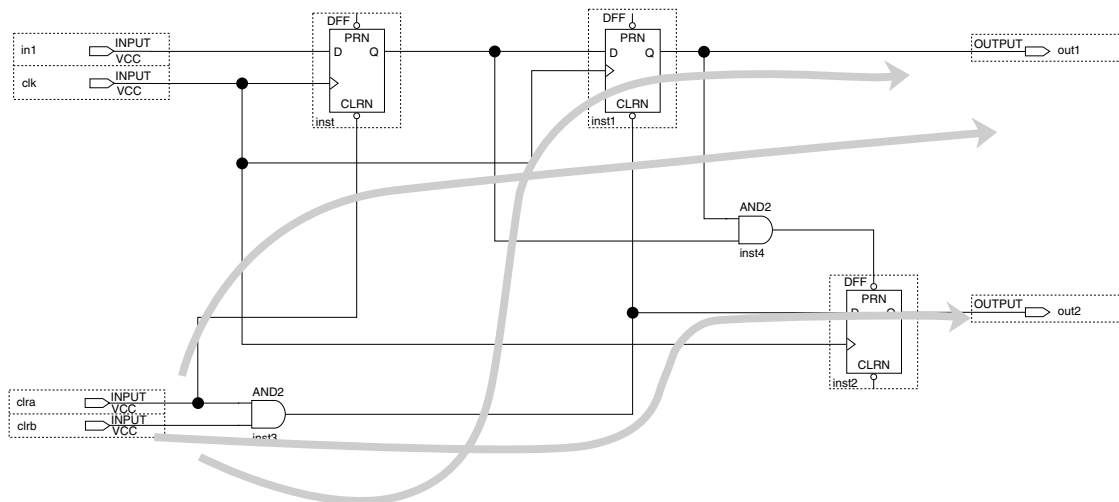
Figure 37. Cutoff Feedback



The path between the registers, marked with the blue arrow, are not measured by timing analysis when this option is selected. Unselecting this option will result in this path being measured during timing analysis.

Cut Off Clear and Preset Signal Paths

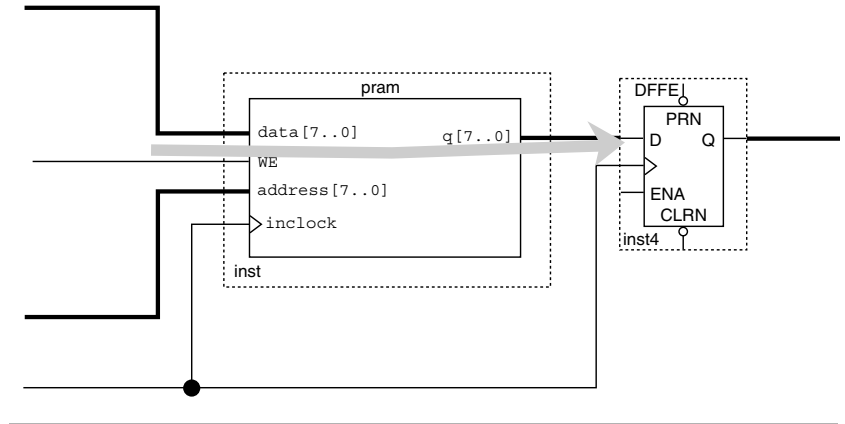
This option is enabled by default and cuts the register's clear and preset paths during timing analysis, as shown in [Figure 38](#).

Figure 38. Cutoff Clear and Preset Signal Paths

The paths, shown in blue, are cut from timing analysis when this feature is enabled. Unselecting this feature will result in these paths through the register preset and clear ports being analyzed and reported in the t_{PD} section of the timing report file.

Cut Off Read During Write Signal Paths

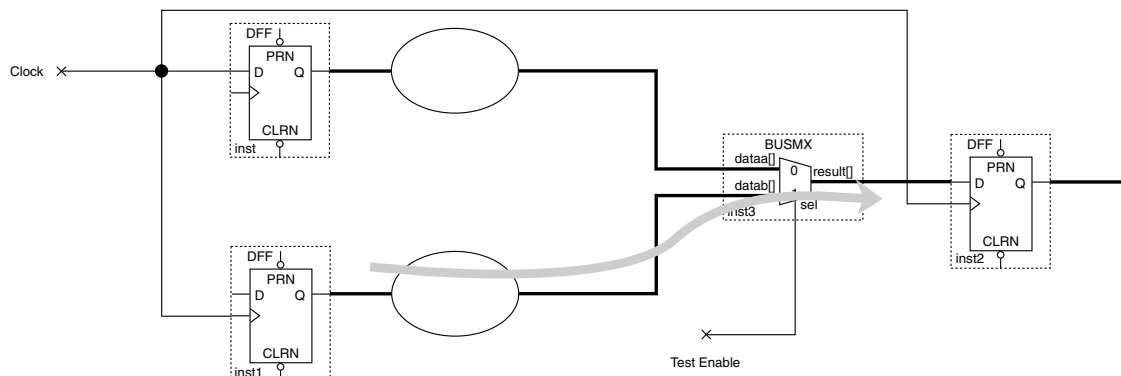
This option, which is enabled by default, cuts the path from the write enable register through the embedded system block (ESB) to a destination register, as shown below in [Figure 39](#).

Figure 39. Cutoff Read During Write Signal Paths

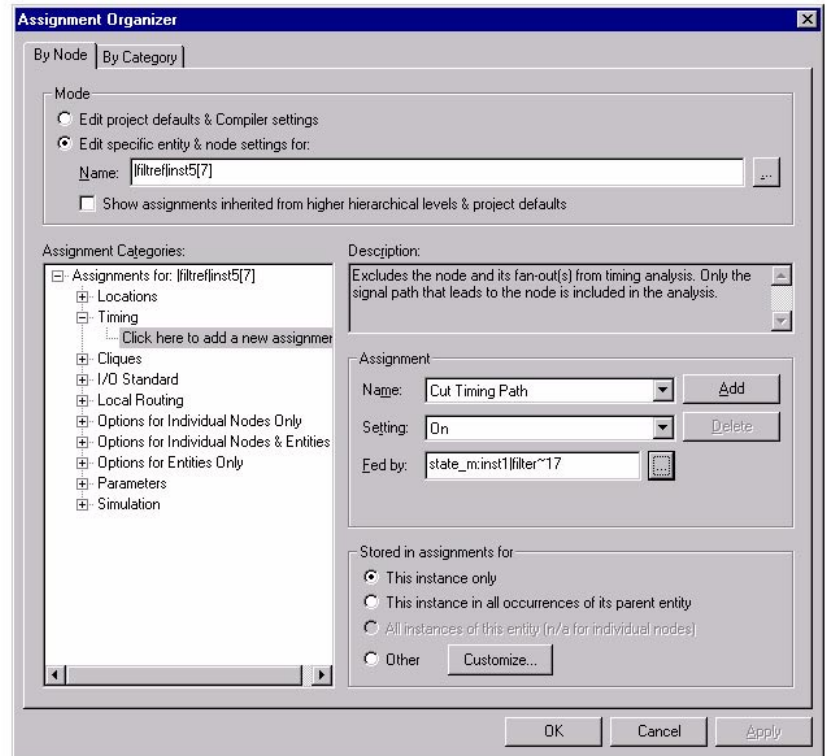
The path, shown in blue, between the we register in the memory block pram to the register inst4, will not be reported by the Quartus II timing analyzer. This path is reported if the cut off read during write signal paths option is unselected.

Cut Timing Path

A cut timing path assignment excludes the specified path or paths from timing analysis. Designs may contain paths between registers that are not relevant to the circuit's operation. These paths are referred to as false paths. To obtain accurate results, designers can configure the Quartus II Static Timing Analyzer to ignore these paths during static timing analysis. Examples of false path signals are signals that are not used under normal operation. [Figure 40](#) shows an example of a false path.

Figure 40. False Path Signal

In Figure 40, the path from `inst1` through the multiplexer to `inst2` is only used for design testing. False paths will not be used under normal operation and should not be considered during timing analysis. A false path can be removed from timing analysis by performing a cut timing path operation from register `inst1` to register `inst2`. Figure 41 shows how a cut timing path assignment can be made in the assignment organizer.

Figure 41. Cut Path Timing Assignment

The assignment can be made to pins or registers and has the functionality described in the following cases.

If the Fed By field is specified, the Fed By node(s) becomes the source and the Name node becomes the destination.

If the Fed By field is not completed, the Name node becomes the source and all paths from the node(s) will be cut. If both the Name node and the Fed By node are clocks, the paths between the two clock domains will be excluded from timing analysis.

Timing Wizard

The Quartus II Timing Wizard allows designers to create, edit, and delete timing assignments, as well as specify overall circuit performance.

The **Timing Wizard** (Project menu) helps designers assign or change one or more of the following timing performance requirements and timing analysis settings. All timing assignments in the Timing Wizard are consolidated under one menu.

- Clock signals or overall circuit frequency (f_{MAX})
- Default system t_{SU} , t_H , t_{CO} , and input-to-nonregistered-output time
- Default external delays to and from device pins
- Settings to control timing analysis and timing-driven compilation

Designers can also change timing settings with the **Timing Settings** command (Project menu), the **Compiler Settings** command (Processing menu), and the **Assignment Organizer** command (Tools menu).

Conclusion

Evolving design and aggressive process technologies call for larger and higher-performance designs to be implemented in programmable logic devices (PLDs). Increasing design complexity creates a need for enhanced timing analysis tools that aid designers in verifying design timing requirements. Without advanced timing analysis tools, designers risk circuit failure of their multiclock and multipath designs. The Quartus II Static Timing Analyzer incorporates a set of powerful, new timing analysis features that are critical in enabling system-on-a-programmable-chip designs.



101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
<http://www.altera.com>
Applications Hotline:
(800) 800-EPLD
Customer Marketing:
(408) 544-7104
Literature Services:
(888) 3-ALTERA
lit_req@altera.com

Altera, APEX, APEX 20K, APEX 20KE, Quartus, and System-on-a-Programmable-Chip are trademarks and/or service marks of Altera Corporation in the United States and other countries. Altera acknowledges the trademarks of other organizations for their respective products or services mentioned in this document. Altera products are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera Corporation. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.



Copyright © 2001 Altera Corporation. All rights reserved.

I.S. EN ISO 9001

