Using TimeQuest Timing Analyzer

For Quartus® Prime 21.1

1 Introduction

This tutorial provides an introduction to TimeQuest Timing Analyzer. It demonstrates how to set up timing constraints and obtain timing information for a logic circuit.

The reader is expected to have a basic understanding of the VHDL hardware description language, and to be familiar with the Intel® Quartus® Prime CAD software.

Contents:

- Introduction to timing analysis
- Using TimeQuest
- Setting Up Timing Constraints

2 Background

Timing analysis is a process of analyzing delays in a logic circuit to determine the conditions under which the circuit operates reliably. One example of a timing analysis computation is to find the maximum clock frequency for a circuit, illustrated in Figure 1.

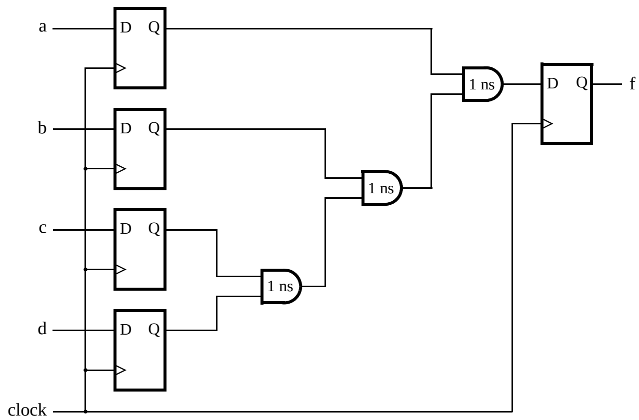

Figure 1. A example for timing analysis.

In this example, flip-flops on the left-hand side drive a combinational circuit that generates an output that is later stored in the flip-flop on the right-hand side. To operate correctly, the clock period has to be long enough to accommodate the delay on the longest path in the circuit. If we assume that the clock-to-Q and setup times for each flip-flop are 1 ns, and the delay through each gate is 1 ns, then the maximum clock frequency for this circuit is:

$$f_{max} = \frac{1}{t_{cq} + 3 \times t_{and} + t_{su}} = \frac{1}{5 \text{ ns}} = 200 \text{ MHz}$$

Computing the longest delays in a circuit and comparing these delays to the clock period is a basic function of a timing analyzer. The timing analyzer can be used to guide computer-aided design (CAD) tools in the implementation of logic circuits. For example, the circuit in Figure 1 shows an implementation of a 4-input function using 2-input AND gates. Without any timing requirements, the presented solution is acceptable. However, if a user requires the circuit to operate at a clock frequency of 250 MHz, then the above solution is inadequate. By placing timing constraints on the maximum clock frequency, it is possible to direct the CAD tools to seek an implementation that meets those constraints. As a result, the CAD tools may arrive at a solution shown in Figure 2. The new circuit has f_{max} = 250 MHz and thus meets the required timing constraints.

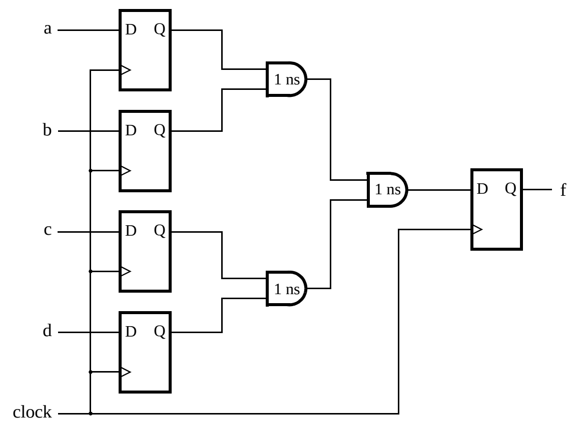

Figure 2. Functionally equivalent circuit with a different logic structure.

In this tutorial, we demonstrate how to obtain timing information and how to set timing constraints using the TimeQuest timing analyzer.

The example circuit provided with this tutorial contains only one clock signal, which is connected to all flip-flops. Performing timing analysis for circuits that have multiple clock signals is discussed in section 6.

3 Design Example

As an example we will use an adder that adds three 8-bit numbers and produces a sum output. The inputs are *A*, *B*, and *C*, which are stored in registers *reg_A*, *reg_B* and *reg_C* at the positive edge of the *clock*. The three registers provide inputs to the adder, whose result is stored in the *reg_sum* register. The output of the *reg_sum* register drives the output port *sum*. The diagram of the circuit is shown in Figure 3.

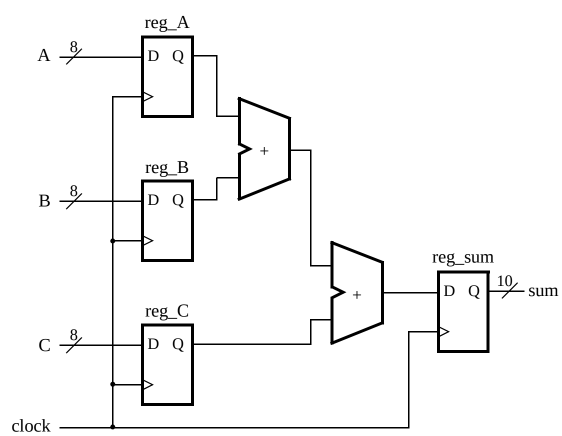

Figure 3. Diagram of the example circuit.

The VHDL source code for the design is given in Figure 4. Note that the "synthesis keep" comment is included in this code. This comment is interpreted as a directive that instructs the Quartus Prime software to retain the specified nodes in the final implementation of the circuit and keep their names as stated. This directive will allow us to refer to these nodes in the tutorial.

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_unsigned.all;

ENTITY add_three_numbers IS
    PORT ( clock : IN STD_LOGIC;
        A, B, C  : IN STD_LOGIC_VECTOR(7 DOWNTO 0);
        sum      : OUT STD_LOGIC_VECTOR(9 DOWNTO 0));
END add_three_numbers;

ARCHITECTURE Behavior OF add_three_numbers IS
    -- Registers
    SIGNAL reg_A, reg_B, reg_C : STD_LOGIC_VECTOR(7 DOWNTO 0);
    SIGNAL reg_sum             : STD_LOGIC_VECTOR(9 DOWNTO 0);
    ATTRIBUTE keep             : boolean;
    ATTRIBUTE keep OF reg_A, reg_B, reg_C, reg_sum : SIGNAL IS true;
BEGIN
    PROCESS ( clock )
    BEGIN
        IF (clock'EVENT AND clock = '1') THEN
            reg_A <= A;
            reg_B <= B;
            reg_C <= C;
            reg_sum <= ("00" & reg_A) + ("00" & reg_B) + ("00" & reg_C);
        END IF;
    END PROCESS;

    sum <= reg_sum;
END Behavior;
```

Figure 4. VHDL code for the example circuit.

To begin the tutorial create a new Quartus Prime project for the design of our example circuit. Select as the target device the EP4CE115F29C7, which is the FPGA chip on the Intel DE2-115 board. Type the VHDL code in Figure 4 into a file and add this file to the project.

You do not need to make any pin assignments for this example. Compile the project to see the results of timing analysis. These results will be available in the compilation report, once the design is compiled.

4 Using TimeQuest

As illustrated in Figure 5, open the Timequest Timing Analyzer section of the Compilation Report, and click on the Clocks item to select it. In the *Clocks* display panel that opens on the right-hand side of the Quartus Prime window, notice that the *clock* signal from the example design has been given a clock period constraint of 1 ns (frequency of 1000 MHz). This is a default constraint that the Quartus Prime CAD tool places on any clock signal in a design project that does not have any user-provided timing constraints.

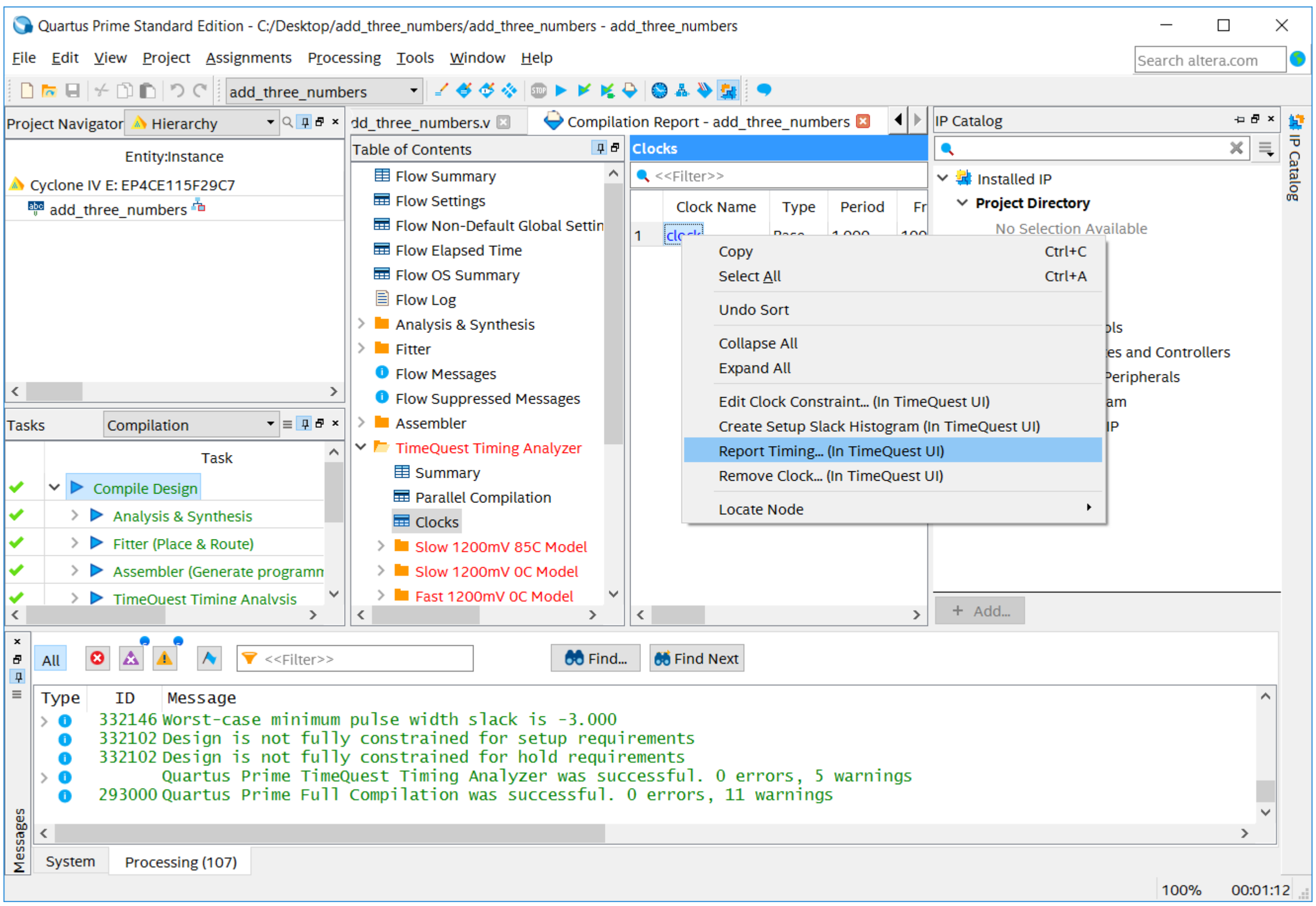

Figure 5. The Timequest section of the compilation report.

As indicated in Figure 5, right-click on the name of the *clock* signal and select the command Report Timing ... (In Timequest UI). This action opens the *Report Timing* dialog shown in Figure 6. Click the drop-down arrow in the From clock item and select the *clock* signal. This selection is used to instruct TimeQuest to analyze all paths in the example circuit that start and end at flip-flops that are clocked by the *clock* signal. The various settings displayed in Figure 6 are described in Section 5.

Accept all of the other default selections in Figure 6 and click on the Report Timing button. This command opens the Timequest Graphical User Interface (GUI), as depicted in Figure 7.

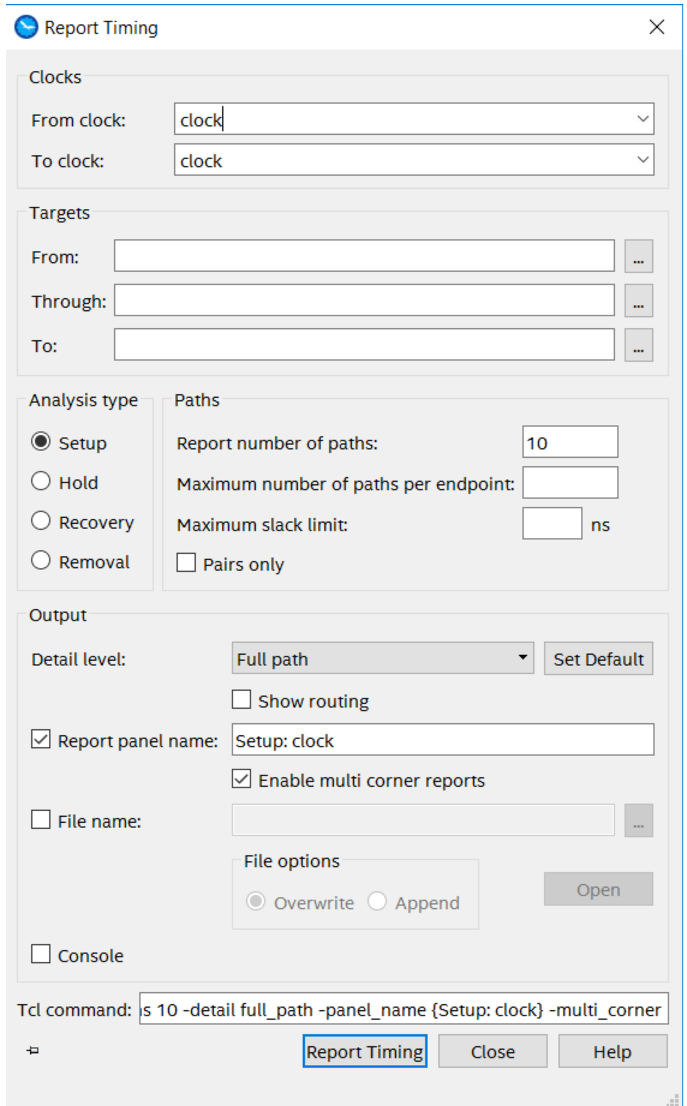

Figure 6. The Report Timing dialog.

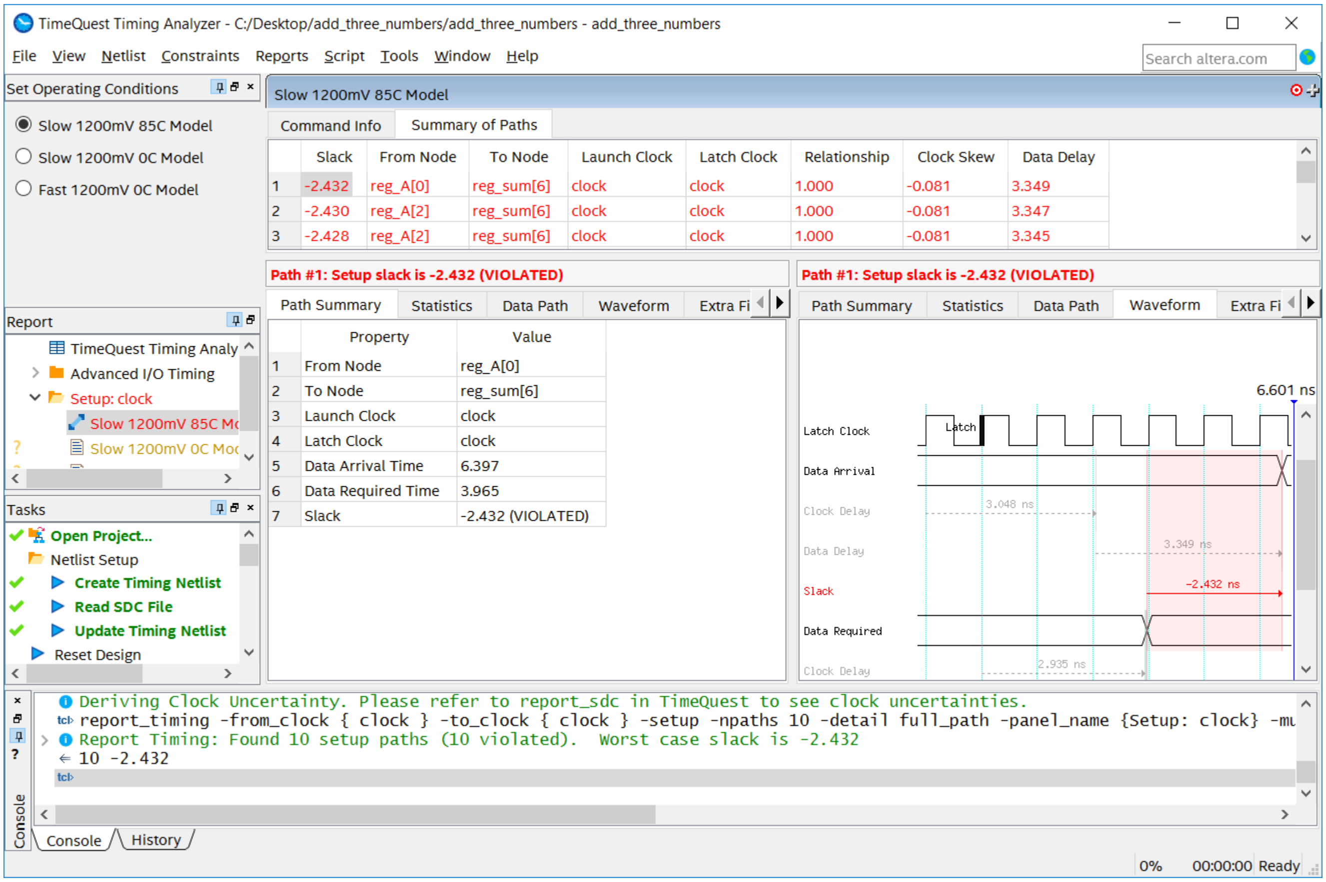

Figure 7. The TimeQuest GUI.

The TimeQuest GUI consists of several sections. They include the main menu at the top, the Report pane in the top-left corner, the Tasks pane on the left, the detailed results panes in the middle, and the Console display at the bottom of the window. The main menu is used to interact with the TimeQuest tool and issue commands. The Report pane contains any reports generated when using the tool, and the Tasks pane contains a sequence of actions that can be performed to obtain timing reports. The View pane hosts any windows that are opened, such as details about the timing information. The Console window at the bottom provides access to a command line for TimeQuest.

We will focus on the panes in the middle of the TimeQuest GUI, which show detailed results of the timing analysis. The Slow 1200mV 85C Model pane lists the analyzed paths in the circuit from source to destination flip-flops. The first column in this pane shows the *Slack* of each path with respect to the (default) clock constraint of 1 ns. For each path in the circuit the slack value represents the difference between the clock period constraint and the path delay; a positive slack means that the delay is smaller than the constraint, and a negative slack represents a delay that is larger than the constraint. The slack values in the report are negative, and shown in red, because the timing results fail to meet the required constraint. The maximum negative slack value shown is -2.432 ns, which means that the worst-case delay path is $1-(-2.432) = 3.432$ ns long. This corresponds to a maximum usable clock frequency, F_{max}, of about 291.38 MHz.

The waveforms shown for path #1 in Figure 7 illustrate the detailed timing situation. The waveforms show that the clock signal takes 3.048 ns to propagate from its input pin to the source flip-flop, and then this flip-flop produces data that takes 3.349 ns to reach the destination flip-flop. Also, the clock signal takes 2.935 ns to reach the destination flip-flop. The clock delays at the source and destination flip-flops represent a clock skew $t_{skew} = 2.935 - 3.048 = -0.113$. A value of 0.032 to account for *Clock Pessimism* is added to the clock skew, making the final clock skew value to be -0.081 ns. The difference in the required arrival time of the data on this path and the actual arrival time is shown by the negative slack value of $1 - 3.349 + t_{skew} = -2.430$ ns. TimeQuest adds a value of -0.002 ns to account for *Clock Uncertainty*, leading to the final slack value of -2.432 ns.

4.1 Setting Up Timing Constraints for a Design

In the TimeQuest GUI, select Constraints > Create Clock, which leads to the Create Clock window shown in Figure 8. Set the Clock name to *clock* and the Period to *4.000* ns. It is necessary to tell TimeQuest which signal in our design this clock constraint applies to. To do this, click the ... button to the right of the Targets field, leading to the Name Finder window shown in Figure 9. Click List to show all of the ports in the design. In the list of ports, highlight *clock*, which is the clock signal in our circuit, press >, then click OK. Finally, click the Run button in the Create Clock window to apply the constraint.

In order to use this clock constraint for all future compilations and timing analysis of this project, we must save the constraint to a file of the type *sdc* which stands for Synopsys* Design Constraint. This file uses an industry-standard format for specifying timing constraints. Select Constraints > Write SDC File... to write all of the currently set constraints (in our case just the one clock constraint) to an SDC file. This leads to the dialog shown in Figure 10. Specify the file name *add_three_numbers.sdc* and press OK. Note that Quartus will by default try to locate and use the sdc file whose file name matches the project name (except for the .sdc extension).

You will notice that our clock timing report *Setup clock* is now out of date, as indicated by the yellow font and highlighting. Right-click the report and select Regenerate, as shown in Figure 11, to re-run the timing analysis using the new 4 ns clock period constraint. This analysis results in a positive slack of 0.568 ns. The corresponding waveforms are depicted in Figure 12.

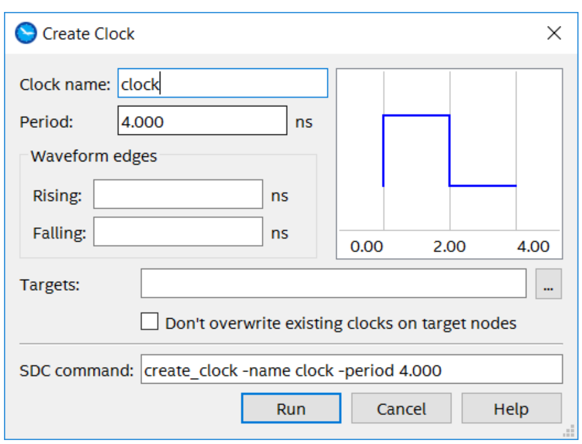

Figure 8. The Create Clock window.

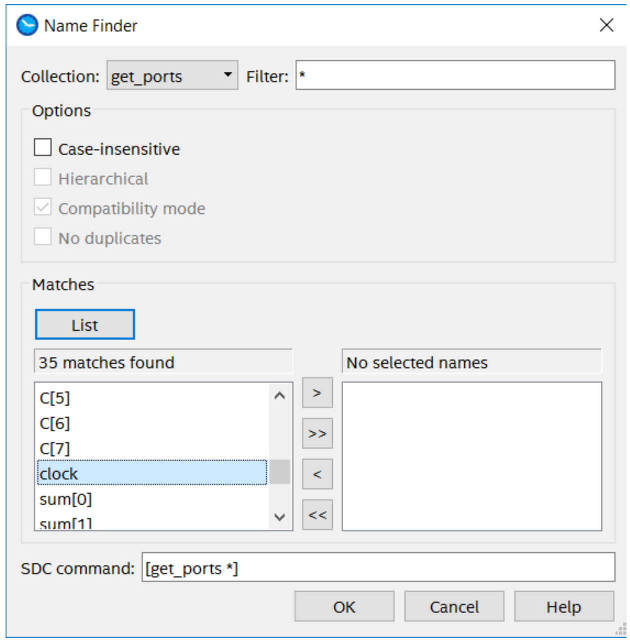

Figure 9. The Name Finder window.

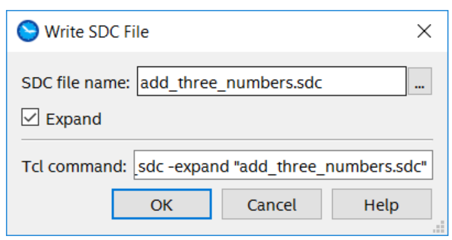

Figure 10. The Write SDC File dialog.

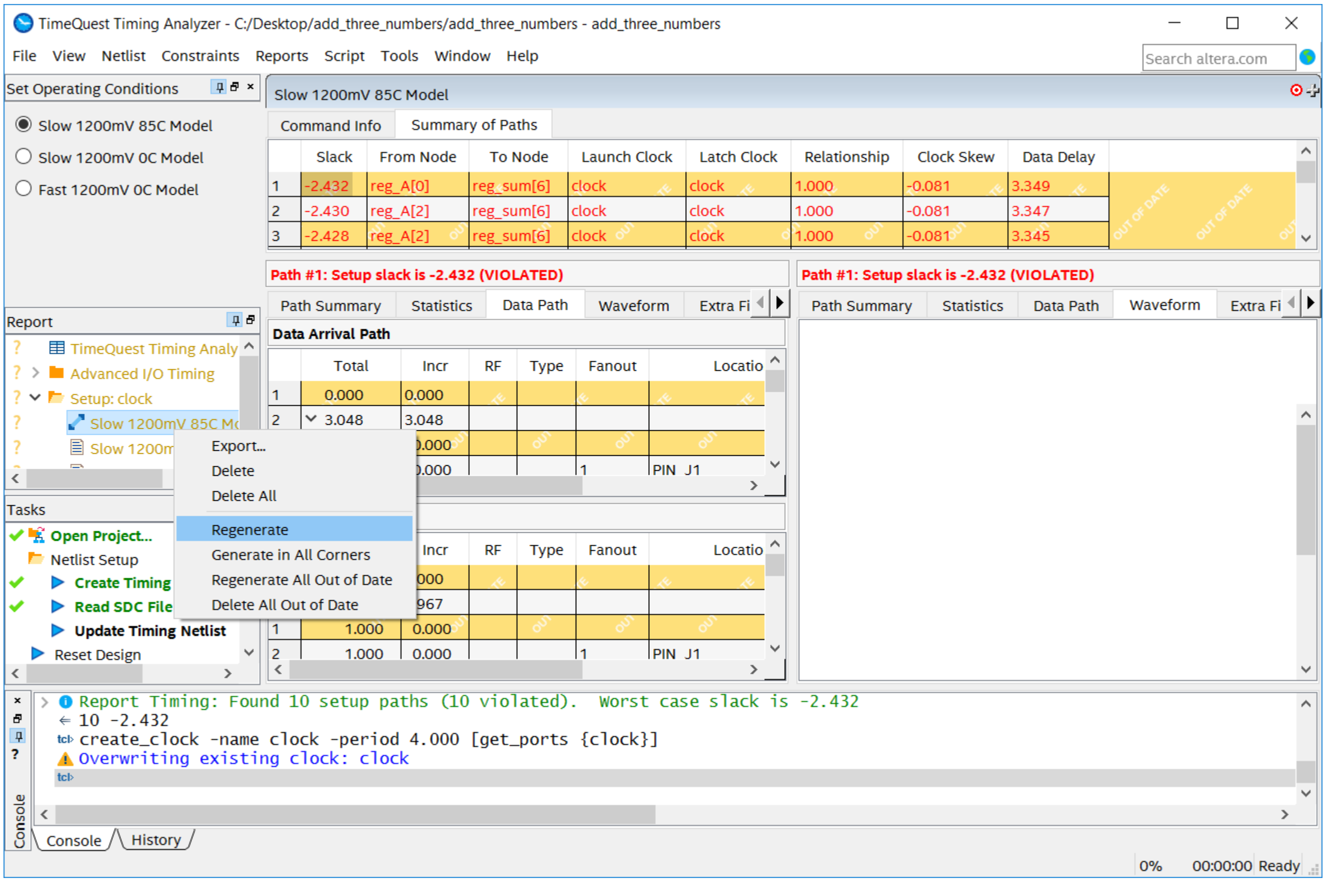

Figure 11. Regenerating the Timing Report.

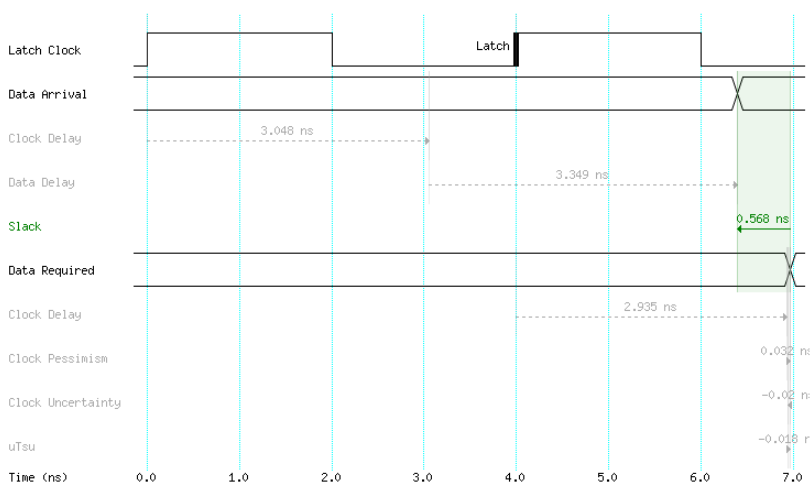

Figure 12. Timing analysis results using the 4 ns clock period constraint.

Close the TimeQuest GUI. A dialog will appear, asking if you want to save the SDC file. Select No, since we have already saved the SDC file.

Since we have not recompiled the example design after adding the 4 ns clock period constraint, the timing analysis is based of the circuit that was produced by Quartus Prime when using the (default) 1 ns clock period constraint. To see the effect of the new timing constraint on the compilation results, recompile the project. Then, follow the steps described previously to perform a new timing analysis using TimeQuest. The 4 ns timing constraint will cause the Quartus Prime optimization algorithms to make different decisions from those made when the (default) 1 ns constraint was used. In particular, the optimization algorithms will likely take less time to execute, because once the generated circuit has sufficient positive slack to meet the constraint, the algorithms can terminate. Figure 13 shows the results of timing analysis, with a positive slack of 0.718 ns.

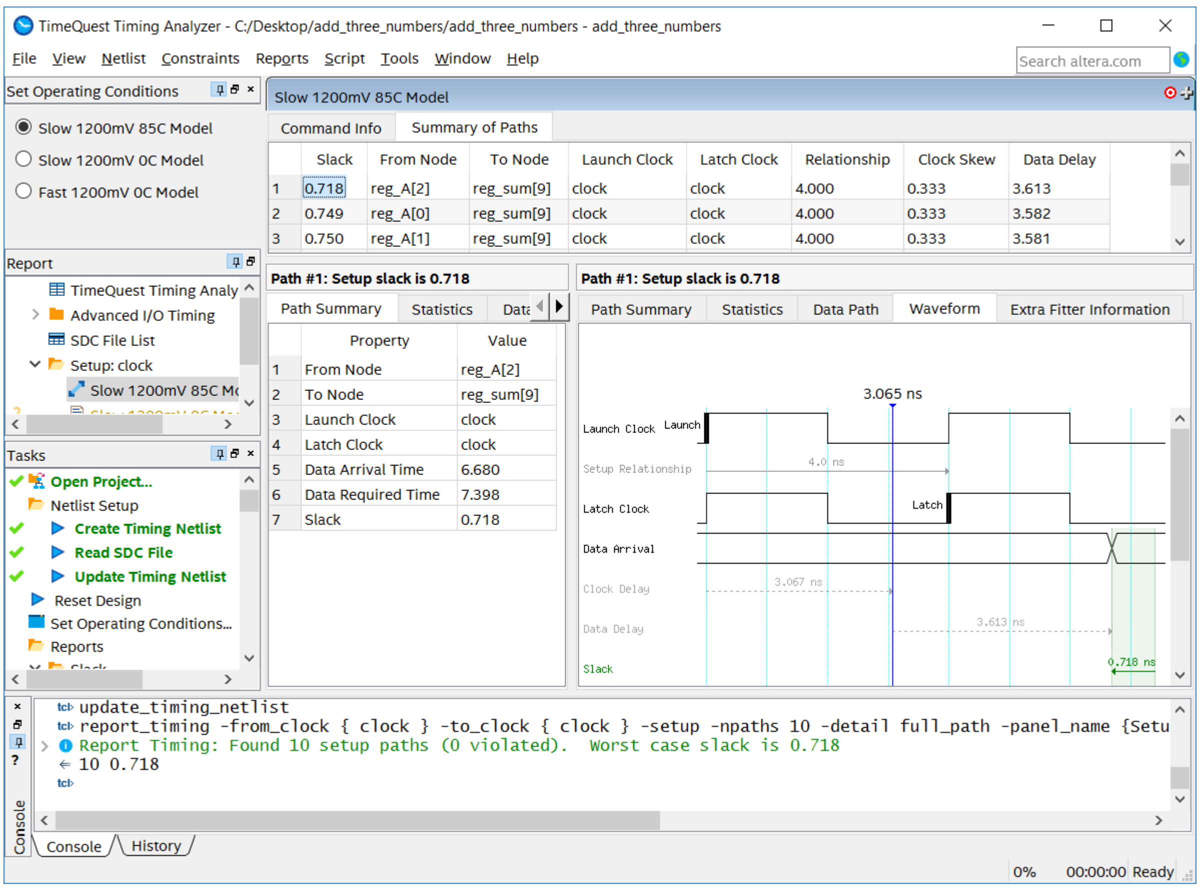

Figure 13. Updated compilation results with the 4 ns clock period constraint.

5 The TimeQuest Graphical User Interface

In the above sections we accessed TimeQuest via the Quartus Prime Compilation Report. Another way to open the TimeQuest GUI is to use the command Tools > TimeQuest Timing Analyzer from the main Quartus Prime window. The TimeQuest window shown in Figure 14, will appear.

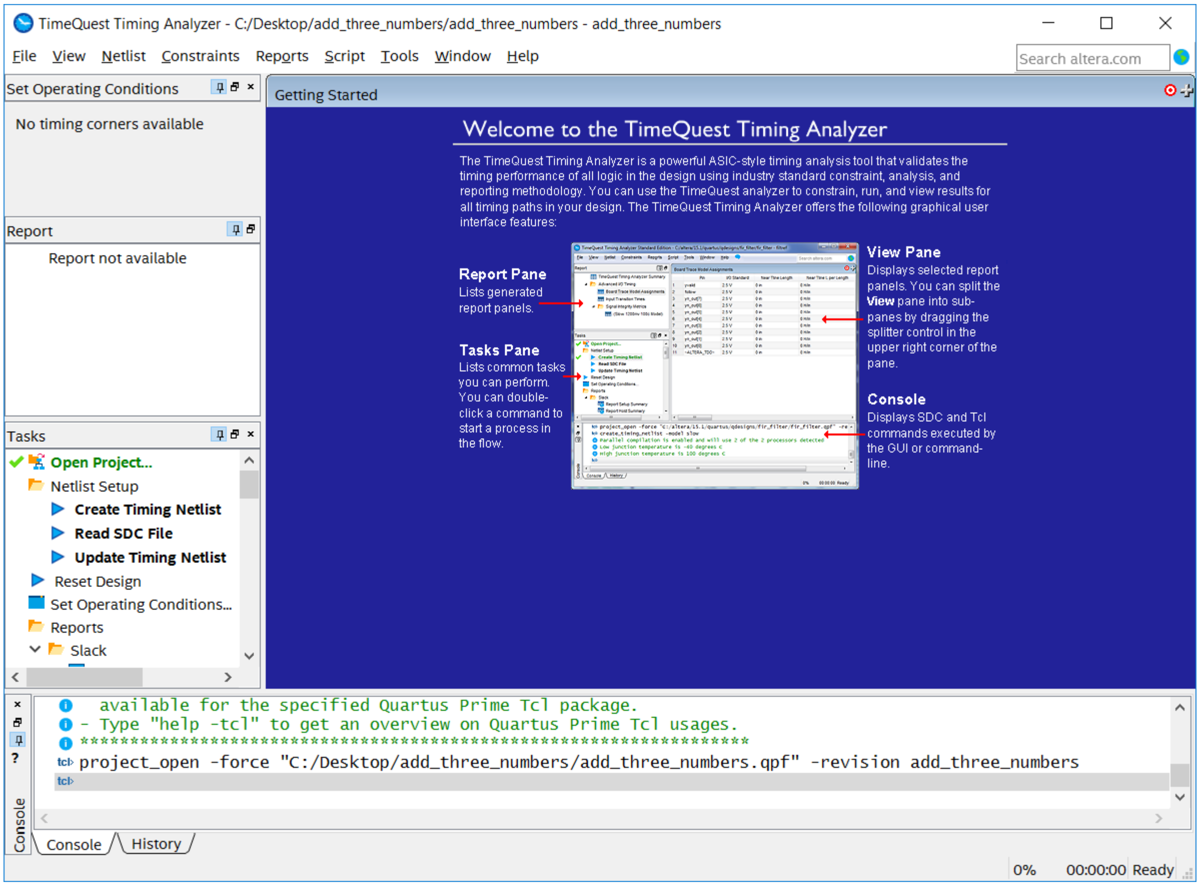

Figure 14. TimeQuest window.

To demonstrate some of the commands available in the TimeQuest GUI, we go through a set of basic steps to obtain timing data for the example design. In the Tasks pane, begin by double-clicking the Create Timing Netlist command to create a timing netlist, which will be used to perform the analysis. Note that while this netlist was generated automatically when performing a timing analysis as described in the previous sections, the netlist can also be generated manually by using the Tasks pane. Next double-click Read SDC File to instruct the analyzer to read a Synopsys Design Constraints (SDC) file and apply the constraints during analysis. The SDC file can be edited manually (using any text editor) at any time, and the timing analysis can then be re-run using the new constraints. Finally, double-click the Update Timing Netlist command to use the specified constraints to determine which parts of the circuit fail to meet them. Once the timing netlist is updated, reports can be generated.

5.1 Timing Analysis Reports

To generate a report, double-click on a report name in the Tasks pane. For example, double-click on the Report Setup Summary. This command will bring up a window in the view pane as shown in Figure 15. Right-click on the *clock* name then click on Report Timing... to open the Report Timing dialog in Figure 16. Although we previously showed this dialog in Figure 6 we now describe it in more detail.

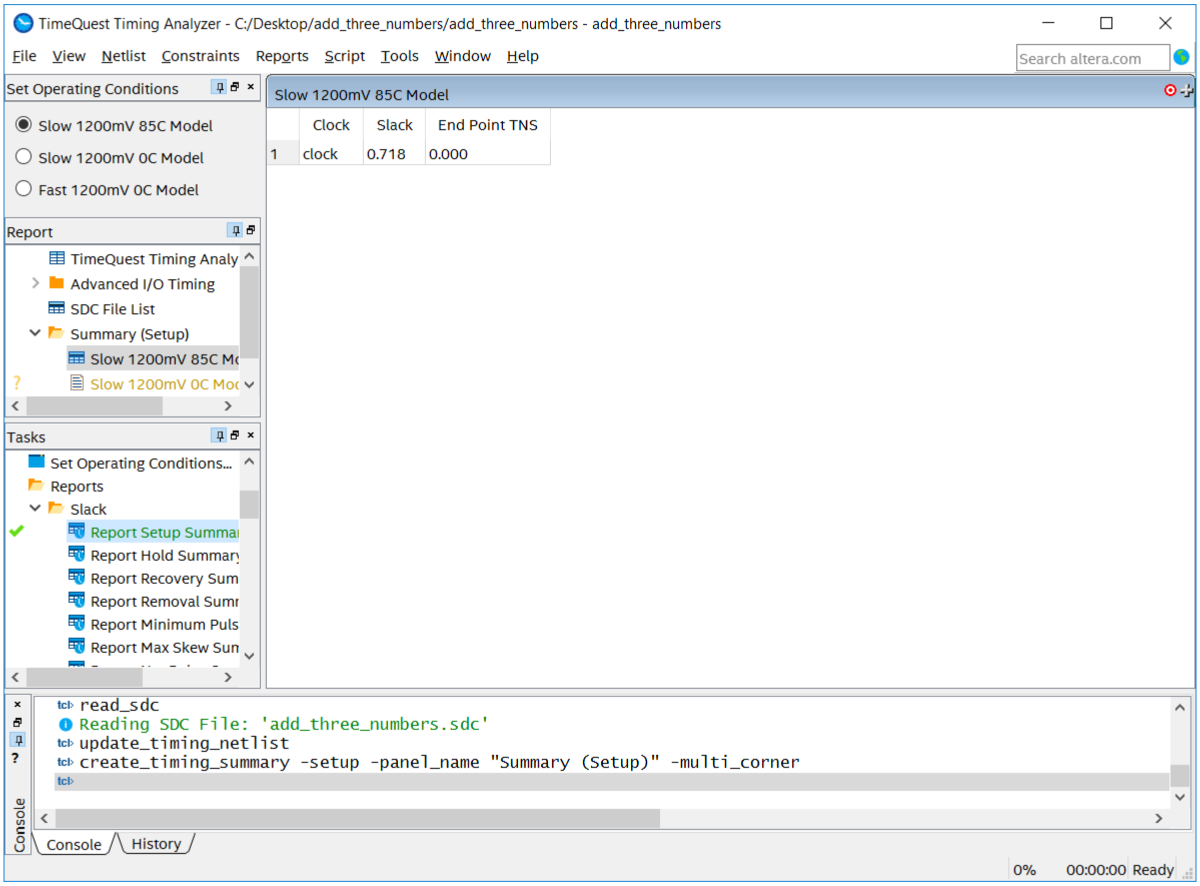

Figure 15. Setup summary.

There are several fields in Figure 16 that help specify the data to be reported. The first field is the Clocks field, which specifies the types of paths that will be reported. More precisely, it specifies the clock signal at the source flip-flops (From clock) and the clock signal at the destination flip-flops (To clock). For this example, choose the signal named *clock* for the To clock and From clock fields. This will limit the reporting to the register-to-register paths only.

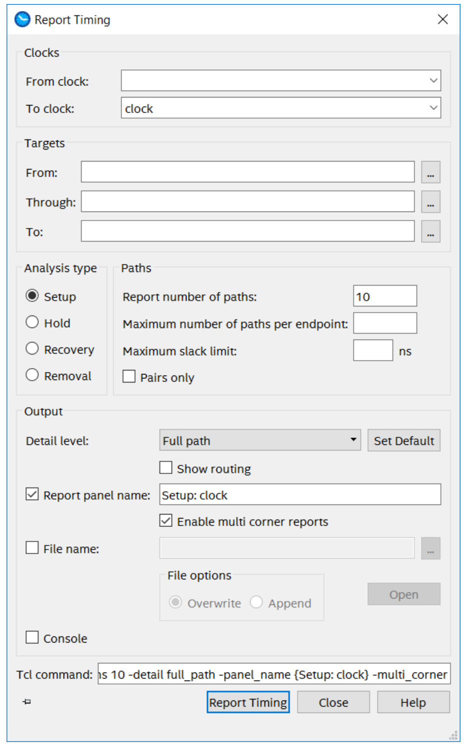

Figure 16. Timing report generation window.

The next field is the Targets field. It can be used to refine the report by focusing only on certain paths in the design. We can specify the starting and the ending point of the paths of interest by filling the From and To fields. In addition, we can look at only the paths that pass through certain nodes in the design. For this example, we leave these fields blank to indicate that every path should be taken into account for the report.

The next two fields are the Analysis type and Paths fields. The Analysis type field specifies if the report should contain setup, hold, recovery, or removal information. Each of these analyses looks for distinct timing characteristics in your design. For example, the setup analysis determines if the data arrives at a flip-flop sufficiently early for the flip-flop to store it reliably, given a clock period. On the other hand, the hold analysis determines if the data input at any given flip-flop remains stable after the positive edge of the clock long enough for the data to be stored in a flip-flop reliably. The Paths field specifies the maximum number of paths to be reported and the maximum slack required for a path to be included in the report. For this example, choose the type of analysis to be Setup and select 10 paths to be reported. This will generate a setup analysis report and show 10 paths with the least slack.

The next set of fields specify the Output format and the level of detail in the report. The output could be to a window or a file. Set the Detail level to Path Only, then set the output to a window by checking the Report panel name check box (and not the File name check box). The window should be named Setup: clock by default, and that name will identify the report in the report pane.

Finally, the last field is the Tcl command field. This field shows a command that will be executed to generate the requested report. You do not need to edit this field.

5.2 Creating Timing Constraints in the TimeQuest GUI

Timing constraints can be entered by using the Constraints menu in the TimeQuest GUI. To assign a clock constraint, select Create Clock... from the Constraint menu. A window shown in Figure 17 will appear.

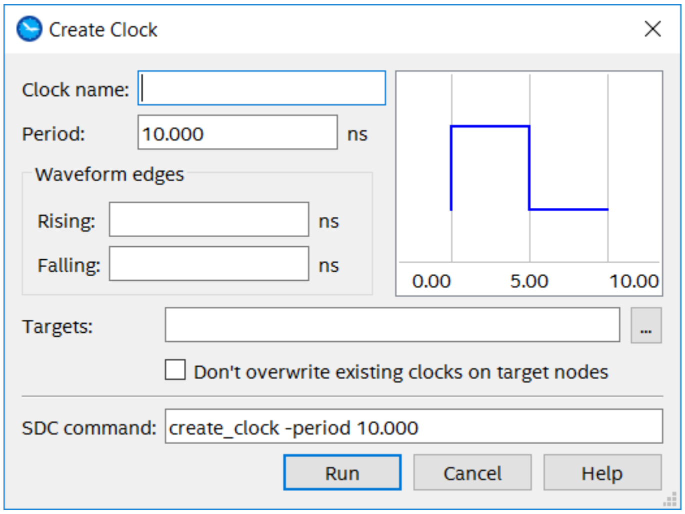

Figure 17. TimeQuest window to create a clock constraint.

In the figure the clock constraint is given the name *clock* in the top field. The clock period is assigned to be 4 ns in the field below. The next two fields define the time at which the clock changes from 0 to 1 and 1 to 0. Leaving these fields empty indicates that the rising edge of the clock should appear at time 0, and the falling edge at one half of the clock period. Finally, the Targets field is set to the signal name *clock* as shown in the figure, to indicate that the given constraint is for the signal named *clock*. Pressing the Run button applies the constraint. The constraint can be saved into an SDC file by double-clicking on the Write SDC File... task as shown in Figure 18.

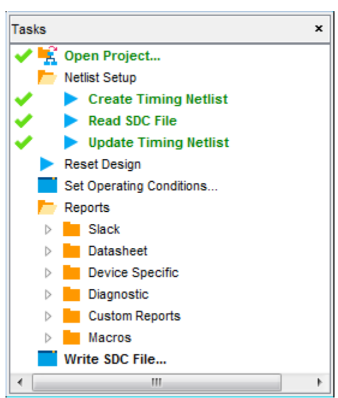

Figure 18. Saving a constraints file.

Once the constraints file is saved, it can be used by Quartus Prime when compiling a project. This is done in Quartus Prime by going into Assignments > Settings... > TimeQuest Timing Analyzer, and adding the SDC file to the TimeQuest timing analyzer settings as shown in Figure 19. The Quartus Prime project can then be recompiled to use the constraint.

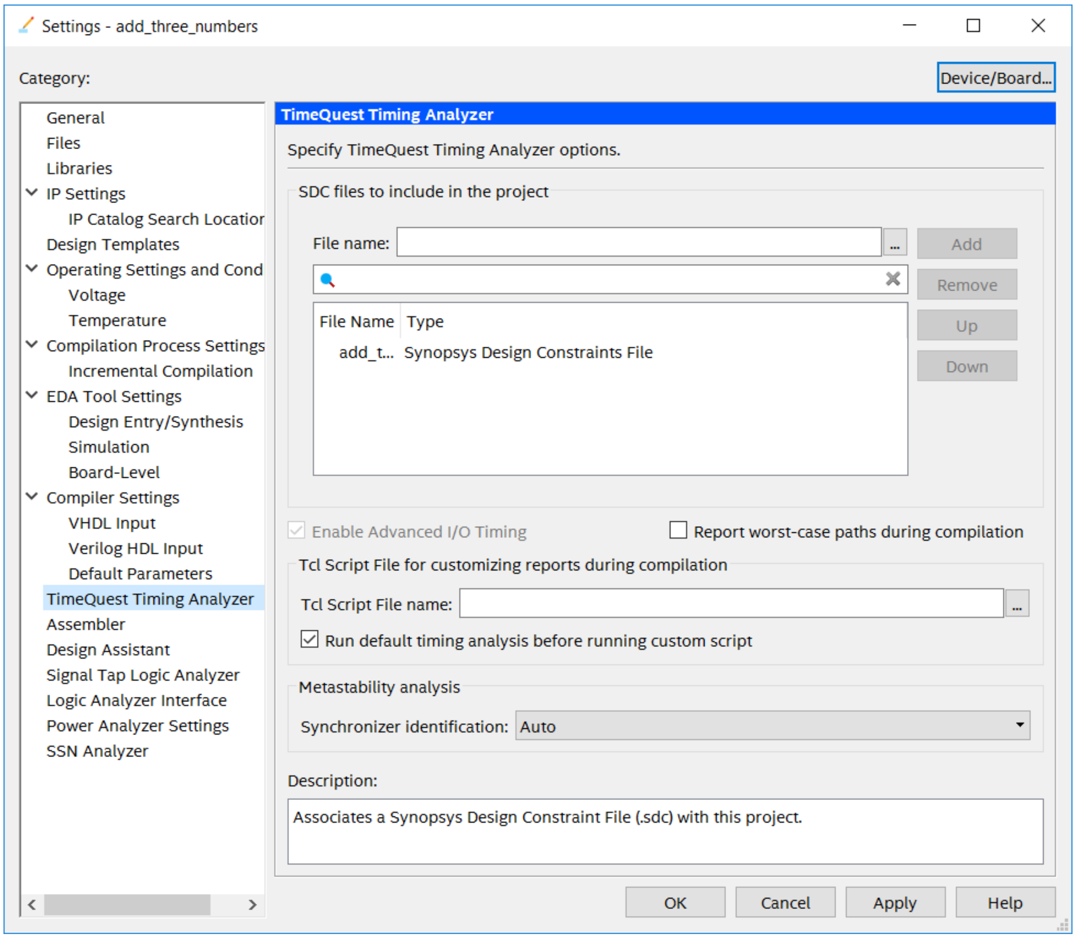

Figure 19. Including a constraints file in Quartus Prime project.

6 Circuits with Multiple Clock Signals

TimeQuest is capable of analyzing circuits that contain multiple clocks. This includes cases where the designer uses several clocks, or where clock signals are generated automatically to support features such as the SignalTap II Logic Analyzer or a JTAG* interface. Should the reader work with such designs, it is important to note that the experience with TimeQuest may differ from that described above. In designs with multiple clocks, it is important to apply constraints to each clock before performing timing analysis. Doing so will make the analyzer provide the same reports as described in previous sections.

7 Conclusion

This tutorial demonstrated the basic use of the TimeQuest timing analyzer. While the descriptions of timing analysis and setting up timing constraints were limited to clock constraints in a simple circuit, TimeQuest provides even more powerful tools to specify timing constraints for larger and more complex designs.