Design of Digital Circuits and Systems, HW5

Static Timing Analysis and Pipelining

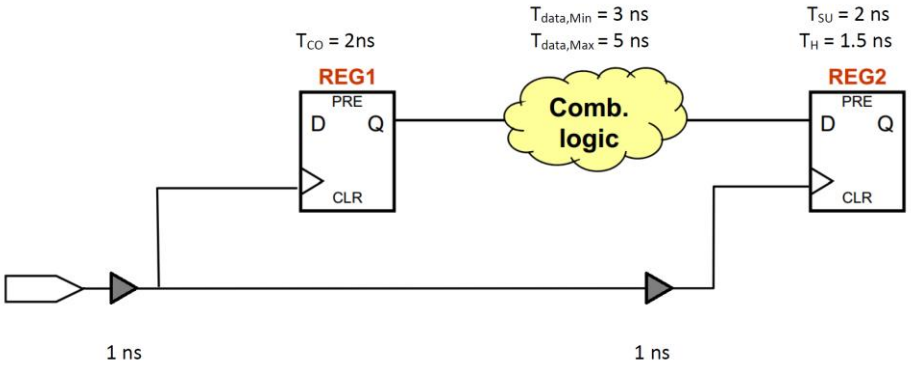
Deliverables (Gradescope)

1) Homework problem solutions (*i.e.*, text, diagrams, screenshots, work) in a single PDF file.
   a) At the end of this document, estimate how long you spent working on the homework and rate the difficulty on the following scale: Very Hard — Hard — Moderate — Easy — Very Easy
2) There is no code to submit for this homework.

Problems

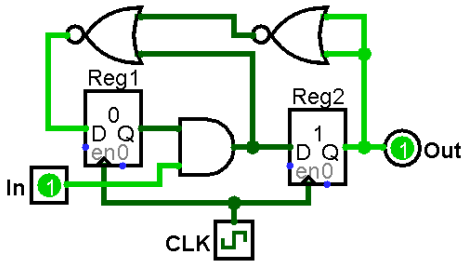
Problem 1:

Calculate the setup and hold slack if the clock runs at **150 MHz**. Does the system meet timing requirements? *Make sure your submitted PDF includes your work.*



Problem 2:

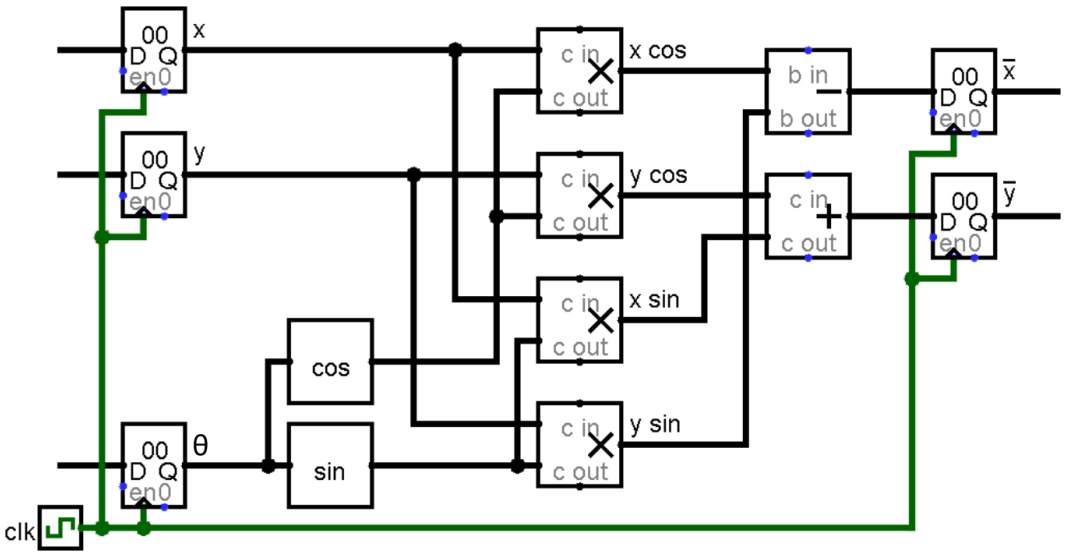
- $t_{su} = 10$ ns
- $t_{CO} \in [8, 10]$ ns
- $t_{clk1} \in [4, 5]$ ns
- $t_{AND} \in [35, 40]$ ns
- $t_{In} \in [13, 17]$ ns
- $t_h = 5$ ns
- $t_{wire} = 0$ ns (all wires)
- $t_{clk2} \in [2, 3]$ ns
- $t_{NOR} \in [25, 30]$ ns
- $t_{period} = 100$ ns



Solve for the setup slack and hold slack for this circuit.
Make sure your submitted PDF includes your work.

Problem 3:

A 2-D rotation of the point (x, y) by θ radians is a very common operation in computer graphics and results in the point $(\bar{x}, \bar{y})$, where $\bar{x} = \boldsymbol{x \cos \theta - y \sin \theta}$ and $\bar{y} = \boldsymbol{x \sin \theta + y \cos \theta}$. Glossing over the exact number representation and bit manipulation, a circuit that computes this rotation can be represented as shown below.



For simplicity, assume that $t_{su} = t_h = t_{wire} = t_{clk} = 0$ and that t_{CO} = 10 ns for all registers.
The logic delays are as follows:

- t_{cos} = 100 ns, t_{sin} = 90 ns
- t_{mult} = 60 ns
- t_{add} = 25 ns, t_{sub} = 30 ns

a) Construct a Data Flow Graph (DFG) for the circuit. Label your three starting inputs x, y, and θ for each of the respective registers, and your outputs $\bar{x}$ and $\bar{y}$.

b) Draw two cut sets on your DFG, one that makes a 2-stage pipelined and one that makes a 3-stage pipelined version of the circuit. Your pipelined circuits should have the minimum possible clock periods. You should use two copies of your DFG or make sure that the different cut sets are *clearly* distinguishable on a single DFG.

c) Assuming we use the minimum clock periods, compute the latency for both your 2-stage and 3-stage pipelined versions of the circuit.

Problem 4:

This part of the homework is an exercise create by Intel for using the Quartus Prime Timing Analyzer. You will need to download `TimingAnalyzer.qar` from the hw5 files.

> This problem will be graded on completion, not correctness. Make sure to include the responses to the questions in red in the directions that follow.

Part 1: Project Setup in Quartus Prime Lite

1) Download *TimingAnalyzer.qar* from Canvas and double-click to open it. If you see a message about Max10, you have the wrong *TimingAnalyzer.qar* file!

2) A dialogue box will appear. Select a destination folder or use the default location and press **OK**.

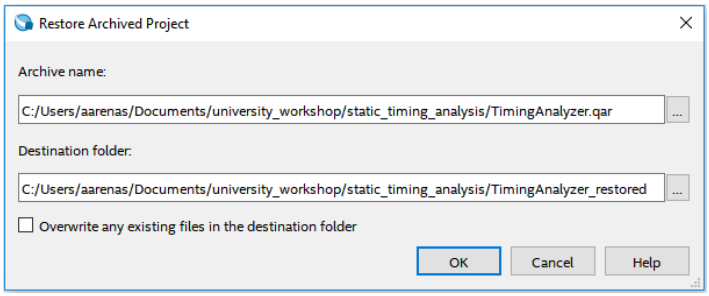


Figure 1: *Dialogue box that restores the* `.qar` *file.*

3) In the Project Navigator, click **Hierarchy** and select **Files** from the drop-down menu. Double-click on `TimingAnalyzer.v` to view the Verilog file. It adds two 128-bit numbers and then multiplies the sum by a 32-bit number.

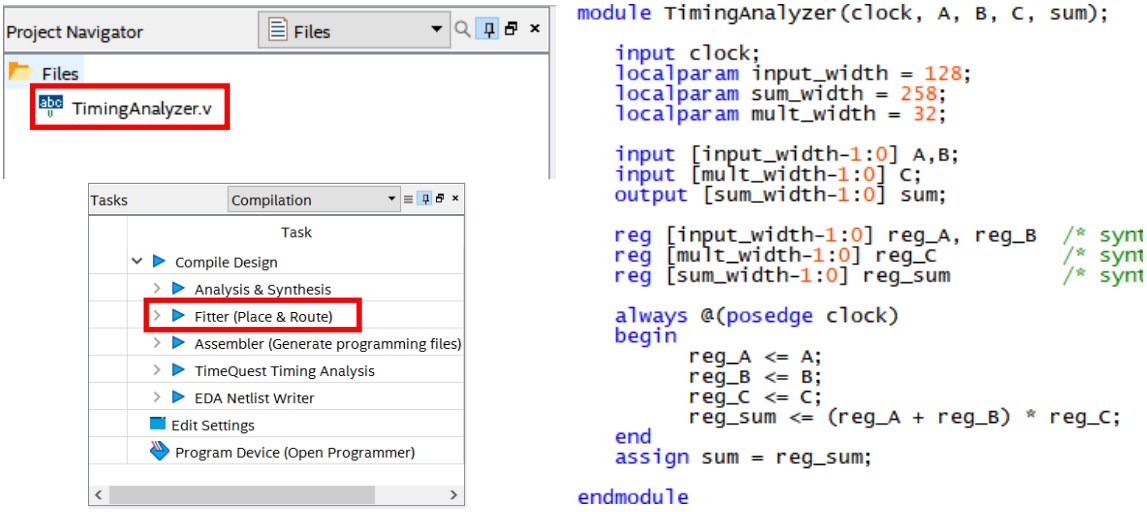


Figure 2: *In the Project Navigator (top-left), clicking the highlighted title will open the Verilog file (right) for Step 3. The Tasks pane in Quartus (bottom-left) is needed for Step 4.*

4) Perform an initial compilation by either double-clicking "**Fitter (Place and Route)**" in the Tasks pane or by going to **Processing** → **Start** → "**Start Fitter**". Before applying timing constraints, we need to create an initial database generated from the post-map results of the design. This can also be done with post-fit results, which requires a full compilation.

5) Select **Tools** → "**TimeQuest Timing Analyzer**" or click the  icon.

Part 2: Using the Quartus Prime Timing Analyzer

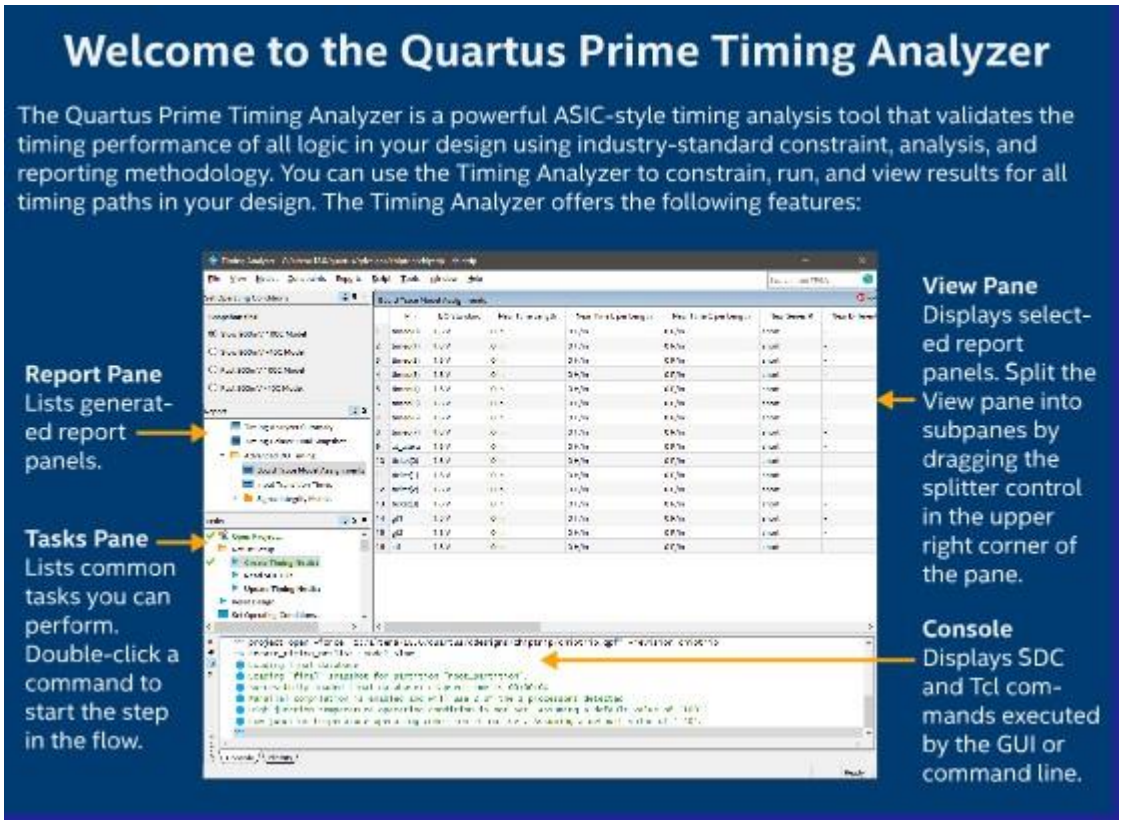


Figure 3: *The first figure that you see when you open Timing Analyzer.*

6) The first thing to do when Timing Analyzer is open is to create a timing netlist. This is done by double clicking "**Create Timing Netlist**" in the Tasks pane. This cannot be done before the initial compilation; Create Timing Netlist requires a post-fit or post-map database.

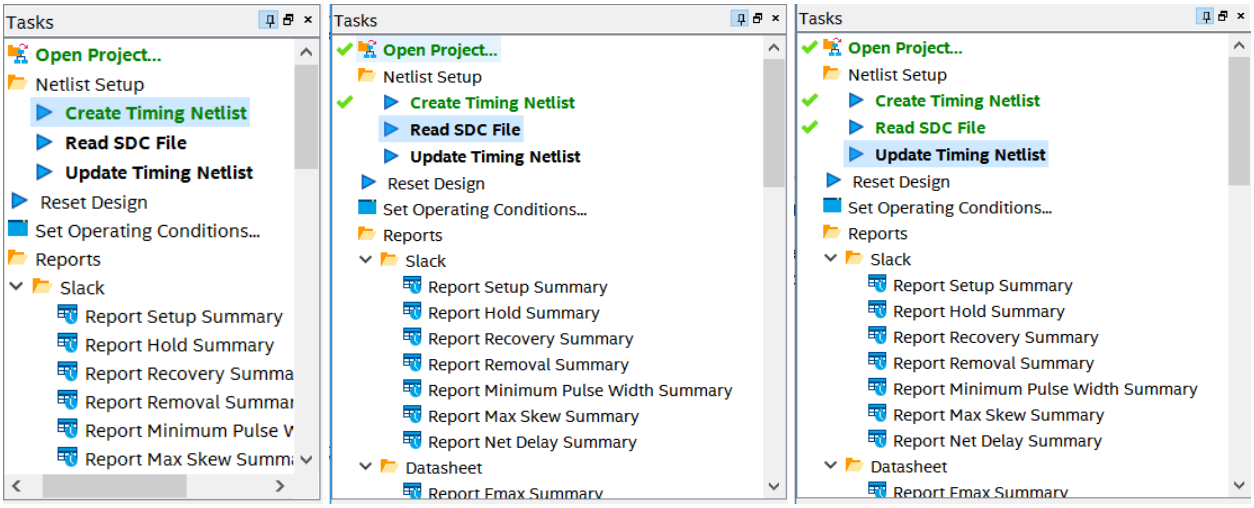


Figure 4: *(Left – Step 6) The Tasks pane in Timing Analyzer. Double-clicking any of the blue play buttons will execute the action. (Middle – Step 7) Double-clicking "**Read SDC File**" will read in the created constraints. (Right – Step 8) Double-click "**Update Timing Netlist**" to access all the reports available.*

7) For now, there is no Synopsys Design Constraints (SDC) file in our project. Timing Analyzer will create a default SDC file with one clock when none is found. Double-click "**Read SDC File**" to generate and read this SDC. In the rest of this exercise, you will overwrite this SDC file with your own constraints created in Timing Analyzer.

8) Update the timing netlist by double-clicking "**Update Timing Netlist**". This creates summaries and reports with useful information.

To save time, you can double-click "**Update Timing Netlist**" without doing Steps 6 and 7 individually. Doing this automatically runs the "**Create Timing Netlist**" and "**Read SDC File**" commands.

9) First, create a clock for the design. Do this by going to the main menu and clicking **Constraints** → "**Create Clock…**"

   a) Name it "`clock`" and give it a period of 12.5 nanoseconds.

   b) Set rising time to 0 and falling time to 6.25 nanoseconds (50% duty cycle).

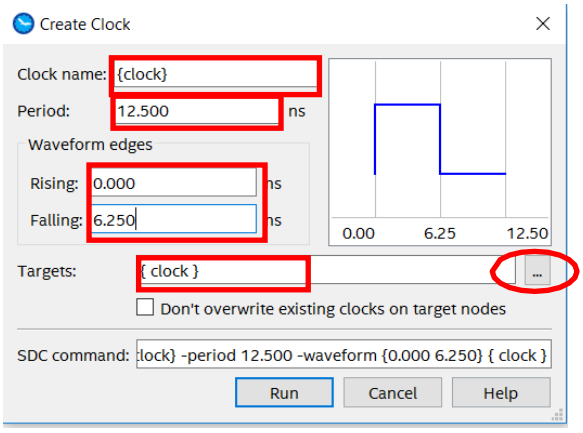
   c) For the targets section, click "**…**" to search for the clock or just type in exactly what you see in the figure below.

      i. If you clicked "**…**" then the Name Finder from Figure 5b will open. Press **List** in the dialogue box that appears. Scroll down to select "clock". Either double-click it or highlight it and press **>** to add your selection. Finally, press **OK**. Your Create Clock window should be identical to the one below. You may click "list" and don't find the "clock" in the list but if you write what you see exactly in the "create clock" pane it should be fine.

   When the Create Clock window is identical to the one below, press **Run**.

When you change a design constraint, the background page will be yellow and show "OUT OF DATE" until you update the page again.

(a)



(b)

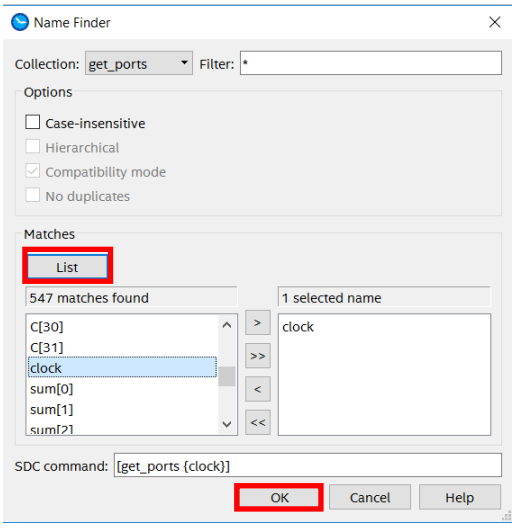


Figure 5: *(a) The Create Clock dialogue window. Click on the "…" at the end of the targets section to open the Name Finder window to specify the clock. (b) The Name Finder window where you select your targets for the Create Clock command.*

10) On the Tasks pane, scroll down to **Diagnostics** and double-click "**Report Clocks**" to see what clocks are driving this system. Doing this updates Timing Analyzer with the most recent constraints, so the "OUT OF DATE" yellow page will go away. Here we find the clock that we just created:

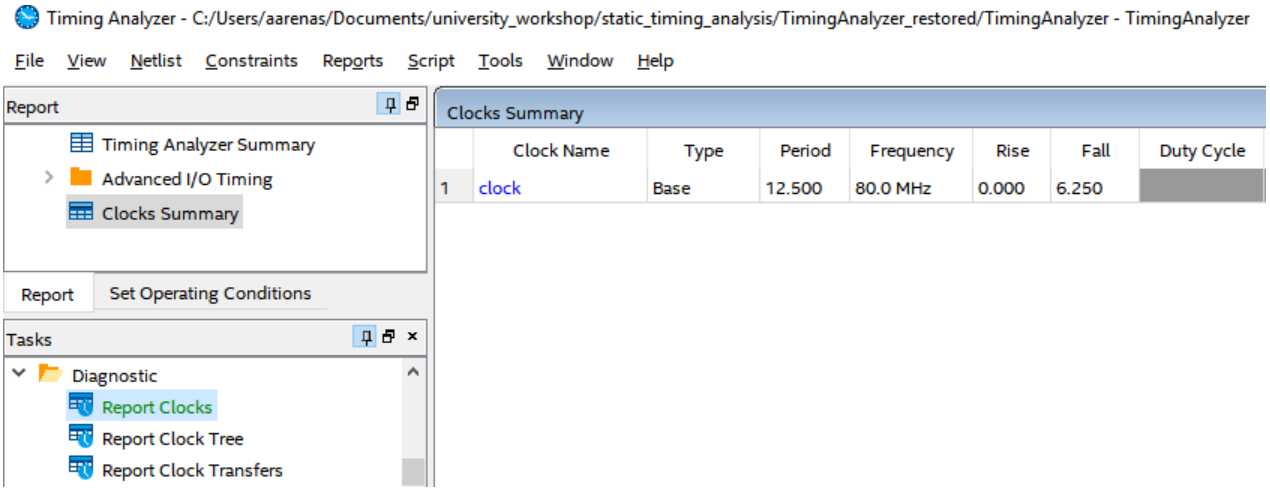


Figure 6: *The clocks in the system followed by all the attributes of the clock.*

11) To access any of the reports, find those of interest in the Tasks pane and double-click them. The next steps will look into "**Report Setup Summary**" and "**Report Hold Summary**" to check if our synchronous adder meets setup and hold requirements.

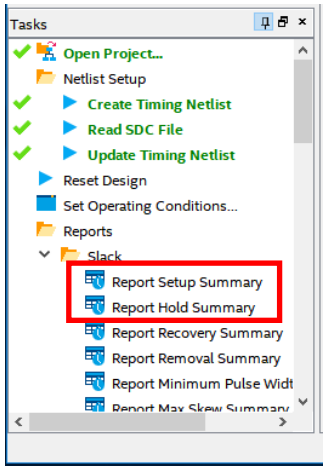


Figure 7: *Double-click a report in the Tasks pane to open it.*

12) Double-clicking "**Report Setup Summary**" shows us the setup slack of our design. If it is red and negative, then it fails the timing requirements. If it is black and positive, then it passes the timing requirements.

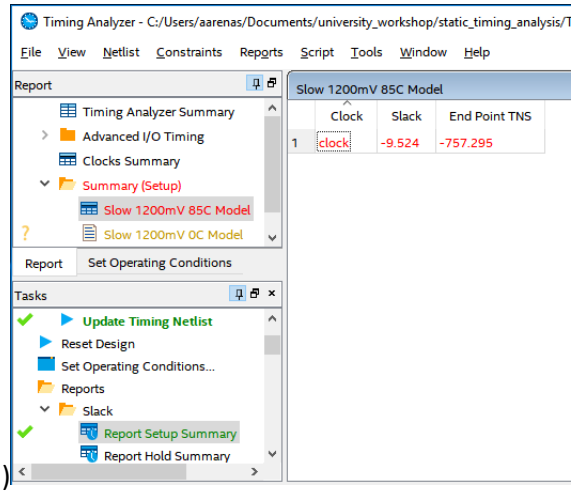
   a) The "End Point TNS" column in the report reports the *total negative slack*, the negative slack of all possible paths summed together.

   b) The "Slack" column reports the slack of the data path that fails the worst.

   c) The "Clock" column reports which clock drives the design.

   Your results may vary from the following screenshots:

(a)



(b)

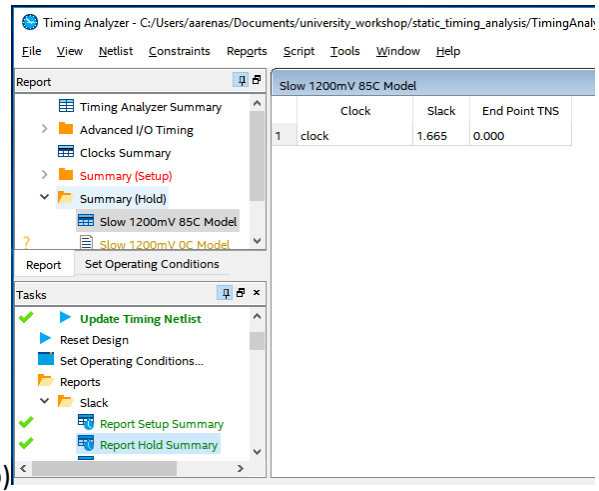


Figure 8: *(a) The setup slack of our design. It is red and negative, meaning that the setup slack requirements were not met. (b) The hold slack of our design. It is black and positive, meaning that the hold slack requirements were met.*

13) Double-clicking "**Report Hold Summary**" reveals that hold timing passes. Here, the path that comes closest to failing timing has 1.665 ns of slack. Your results may vary from the above screenshot.

14) Generate a timing report on our clock to investigate the setup violation. Do this from the Report Clocks summary window.

    a) Open the dialogue box shown to the right by either:

        i. Right-clicking "clock" and then clicking "**Report Timing…**" OR

        ii. Scroll down in the Tasks pane to "Custom Reports" and double-clicking "**Report Timing…**"

    b) The **Clocks** section allows you to decide which clock in the design you want the report on. There is only one, so select **clock** as shown to the right.

    c) The "**Analysis Type**" section determines which report you will see. Select **Setup** first.

    d) The **Paths** section allows us to select how many paths will be shown. It will show the paths that have the least slack first, so entering 1 will show us the single worst path.

    e) Finally, click "**Report Timing**" to see this custom report.

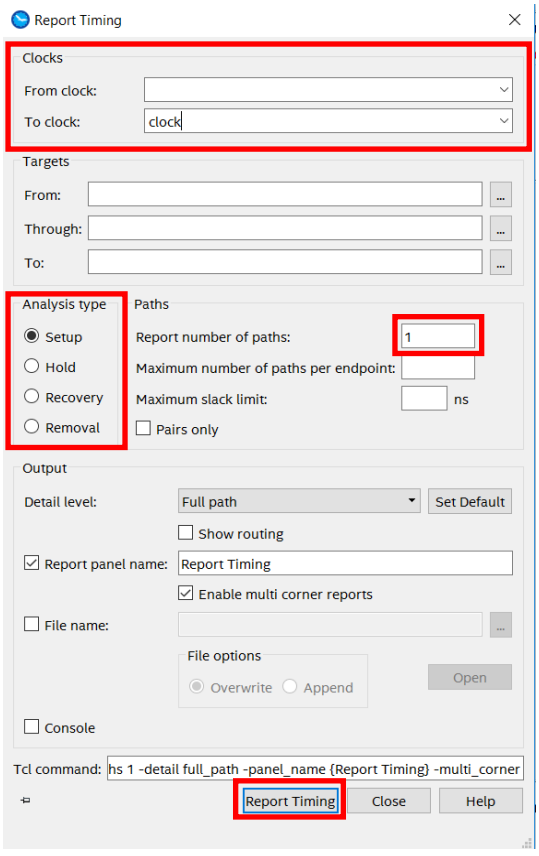


Figure 9: *The "**Report Timing…**" dialogue box.*

15) The resulting report has several parts of interest:

a) Under "**Summary of Paths**" (tab in the top window), the first column shows the setup slack of the paths that fail timing the worst in descending order. The next two columns define where this path starts and ends, followed by other clock details.

b) The "**Data Arrival Path**" (in the "Data Path" tab in the lower-left window) is equivalent to the Data Arrival Time from the slack equation, but shown in much more detail. It is possible to trace the entire path with the information shown below.

c) The "**Data Required Path**" is similarly equivalent to the Data Required Time from the slack equation. These values can be used to manually confirm Timing Analyzer's result for setup slack.

d) The **Waveform** viewer on the bottom-right gives a graphical view of each of the components in the slack equations.

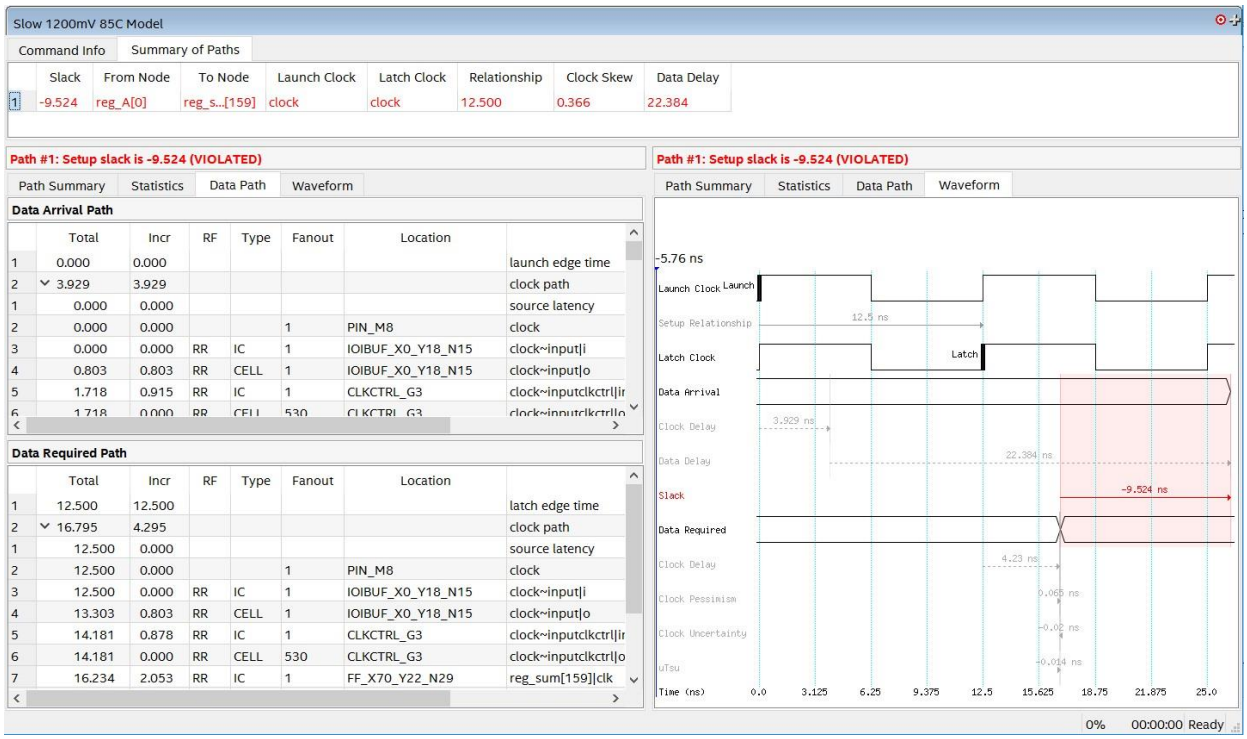


Figure 10: *The result of the custom timing report.*

16) *Setup Slack = Data Required Time – Data Arrival Time*. First, trace the "**Data Required Path**" to find all the delays along the clock path to the destination register.

a) The **Total** column counts all of the delay up to that point.

b) The **Incr** column shows the increment by which each row adds to the delay.

c) The **Type** column describes where the delay originates from.

d) The **Fanout** column shows how many outputs leave that unit.

e) The **Location** column tells you where in the FPGA it can be found.

f) The **Element** column describes what part of the design we are describing.

Data Required Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	12.000	12.000					latch edge time
2	⌄ 16.295	4.295					clock path
1	12.000	0.000					source latency
2	12.000	0.000			1	PIN_M8	clock
3	12.000	0.000	RR	IC	1	IOIBUF_X0_Y18_N15	clock~input|i
4	12.803	0.803	RR	CELL	1	IOIBUF_X0_Y18_N15	clock~input|o
5	13.681	0.878	RR	IC	1	CLKCTRL_G3	clock~inputclkctrl|inclk[0]
6	13.681	0.000	RR	CELL	530	CLKCTRL_G3	clock~inputclkctrl|outclk
7	15.734	2.053	RR	IC	1	FF_X70_Y22_N29	reg_sum[159]|clk
8	16.230	0.496	RR	CELL	1	FF_X70_Y22_N29	reg_sum[159]
9	16.295	0.065					clock pessimism removed
3	16.275	-0.020					clock uncertainty
4	16.289	0.014		uTsu	1	FF_X70_Y22_N29	reg_sum[159]

Figure 11: *The "**Data Required Path**," whose total delays add up to equal the data required time.*

17) Trace the "**Data Arrival Path**" to view all the delays involved in the data's transfer from the source to the destination register. Use the **Location** column to see where in the FPGA the data is.

Data Arrival Path

	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					launch edge time
2	⌄ 3.929	3.929					clock path
1	0.000	0.000					source latency
2	0.000	0.000			1	PIN_M8	clock
3	0.000	0.000	RR	IC	1	IOIBUF_X0_Y18_N15	clock~input|i
4	0.803	0.803	RR	CELL	1	IOIBUF_X0_Y18_N15	clock~input|o
5	1.718	0.915	RR	IC	1	CLKCTRL_G3	clock~inputclkctrl|inclk[0]
6	1.718	0.000	RR	CELL	530	CLKCTRL_G3	clock~inputclkctrl|outclk
7	3.396	1.678	RR	IC	1	FF_X67_Y29_N17	reg_A[0]|clk
8	3.929	0.533	RR	CELL	1	FF_X67_Y29_N17	reg_A[0]
3	⌄ 26.313	22.384					data path
1	4.148	0.219		uTco	1	FF_X67_Y29_N17	reg_A[0]
2	4.148	0.000	FF	CELL	1	FF_X67_Y29_N17	reg_A[0]|q
3	4.690	0.542	FF	IC	2	LCCOMB_X67_Y29_N16	Add0~0|dataa
4	5.244	0.554	FF	CELL	1	LCCOMB_X67_Y29_N16	Add0~0|cout
5	5.244	0.000	FF	IC	2	LCCOMB_X67_Y29_N18	Add0~2|cin
6	5.305	0.061	FR	CELL	1	LCCOMB_X67_Y29_N18	Add0~2|cout
7	5.305	0.000	RR	IC	2	LCCOMB_X67_Y29_N20	Add0~4|cin

Figure 12: *The "**Data Arrival Path**," whose total delays add up to equal the data arrival time.*

18) What element adds the most delay? How can we change the clock parameters such that setup slack does meet timing requirements?

19) Repeat the "**Report Timing…**" steps to investigate the *hold slack* results. Does the same element add the most delay for hold slack? Why does this make sense?

20) Edit the clock constraint such that setup slack no longer fails timing. This can be done directly in the SDC file or by using the Timing Analyzer GUI:

   a) Double-click on "**Report Clocks**" in the Tasks pane.

   b) Right-click "clock" and select "**Edit Clock Constraint…**".

   c) Enter the clock period that would make setup slack pass timing requirements. Make sure the rising time is at 0 and the falling time is at half of the clock period.

   d) Once finished, hit **Run**.

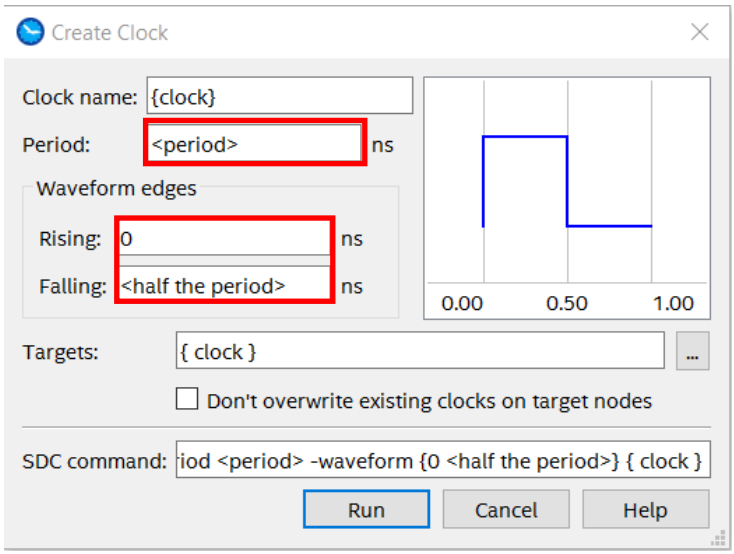


Figure 13: *The "**Create Clock**" GUI in Timing Analyzer allows you to specify clock frequency, as well as rising and falling times. It presents the equivalent SDC command in the bottom text box. Edit the highlighted boxes to make setup slack meet timing requirements!*

21) The "**Clocks Summary**" window should now be yellow and show that it is out of date. Write the recent clock creation to an SDC file by selecting **Constraints** → "**Write SDC File…**" and the newly created clock will overwrite the default one in the SDC file. Your dialogue box should look exactly the same as Figure 14. Press **Ok**.

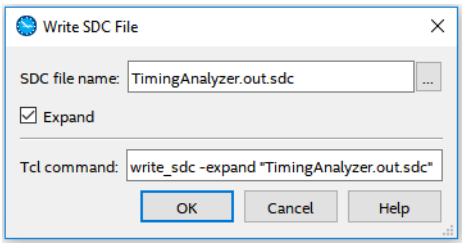


Figure 14: *The "**Write SDC File…**" dialogue box for saving an SDC constraint created in the GUI.*

22) Double-click "**Update Timing Netlist**" to view the new reports from your design with the updated clock. You can see the updated clock in the "**Report Clock**" summary page now. Double-click on "**Report Setup Summary**" to see if timing is met with these new conditions. Your results will different from the following screenshots depending on your chosen clock period. What is the fastest frequency you can set the clock to for this to pass setup timing?

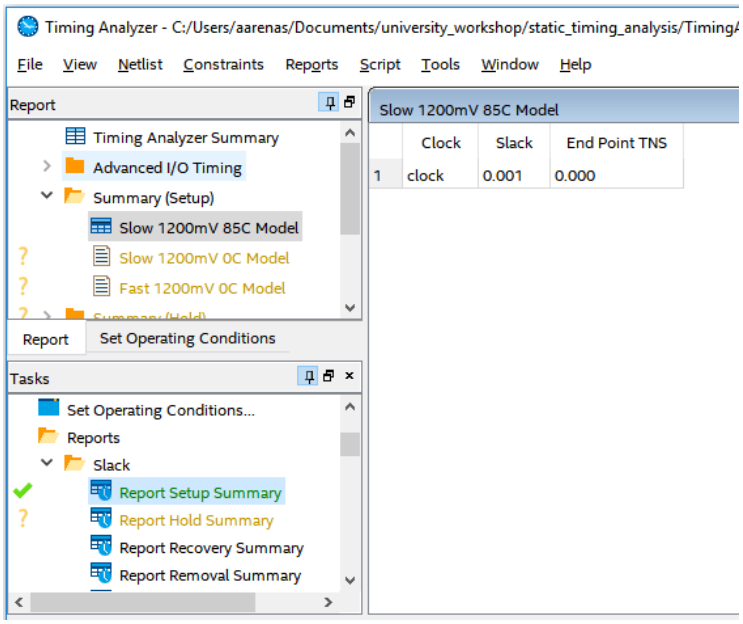


Figure 15: *The new setup summary reveals that timing does pass if the clock is adjusted to the proper frequency.*

23) Generate another timing report exactly as you did in Step 14 by selecting "**Custom Reports**" → "**Report Timing…**" in the Tasks pane. The design now passes timing. Investigate how it's different.

24) Timing Analyzer only works completely when the designer enters constraints for all possible paths and clocks. Check if there are unconstrained paths by scrolling down in the Tasks pane in the **Diagnostics** section and double-clicking "**Report Unconstrained Paths**." A completely constrained design has 0 unconstrained ports, paths, and clocks.

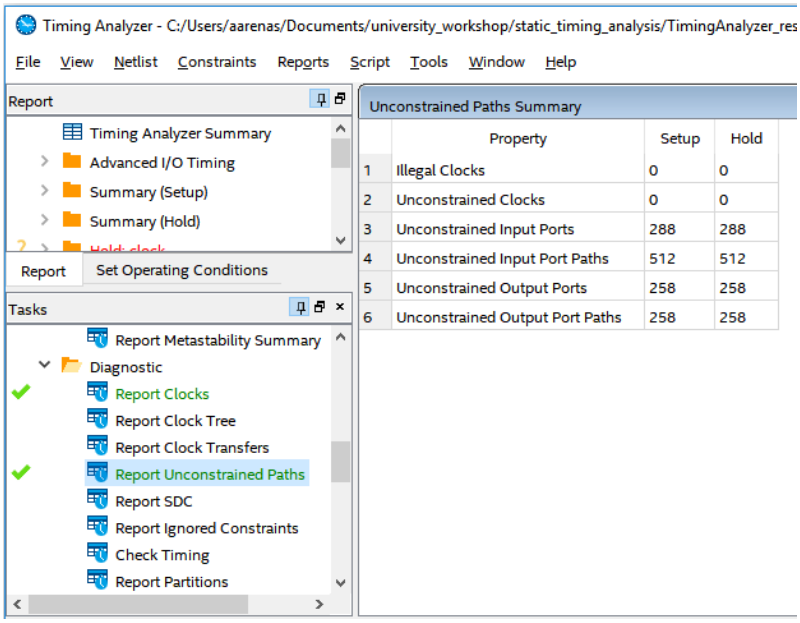


Figure 16: *The report containing the number of unconstrained paths in the design.*

25) Constrain the input ports and paths by adding a minimum and maximum `set_input_delay`.

a) To add the constraints in Timing Analyzer, click **Constraints** → "**Set Input Delay…**". Use -0.5 ns as the minimum delay and 6.5 ns as the maximum delay (*i.e.*, do steps 25 & 26 for each).

b) After setting the clock name to "clock" and the delay value to the proper number, click the "**…**" to the right of the Targets section highlighted below to add all the inputs.

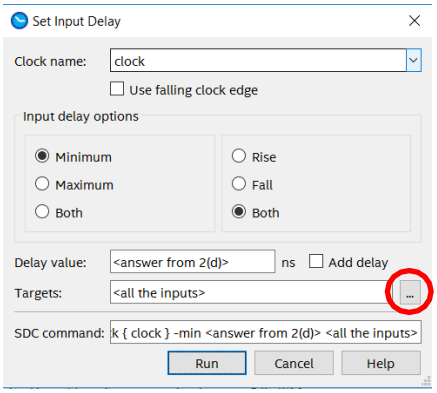


Figure 17: *The* `set_input_delay` *GUI window.*

26) Press **List** to see all the ports in our design. Select every port that begins with A, B, and C (besides clock) and press the **>** button to add them. Press OK.

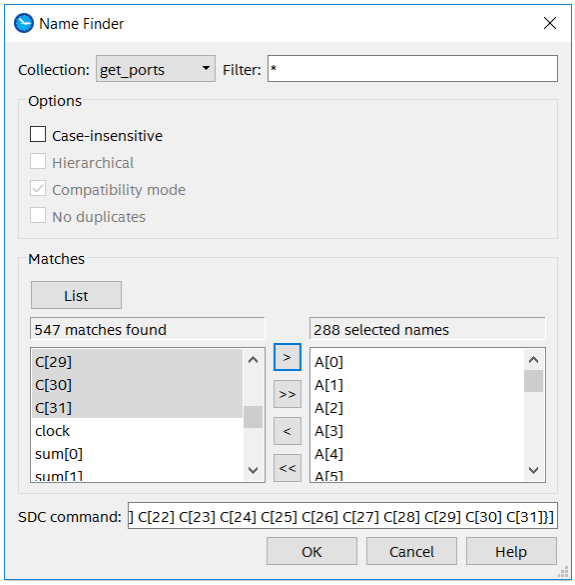


Figure 18: *The window that selects which ports to apply the* `set_input_delay` *to.*

27) The Unconstrained Paths Summary should now have a yellow background with "OUT OF DATE" watermarks across the page. Double-click "**Report Unconstrained Paths**" in the Tasks pane to update it with your recent actions. Similarly, set the min output delay to -0.5 ns and max output delay to 5.6 ns for all `sum` outputs. Double-clicking "**Report Unconstrained Paths**" from the Tasks pane should reveal that there are zero unconstrained paths remaining.

28) Once you've finished adding all the constraints, click **Constraints** → "**Write SDC File…**" to save them (same as Step 21 and Figure 14). Now open the SDC file from your working directory. Look around and see how much time Timing Analyzer saved you by making all these for you! There should be the created clock, as well as all the minimum and maximum `set_input_delay` and `set_output_delay` constraints you created.

29) All the new constraints, however, have created a hold violation within our system. Double-click "**Report Hold Summary**" to verify this.

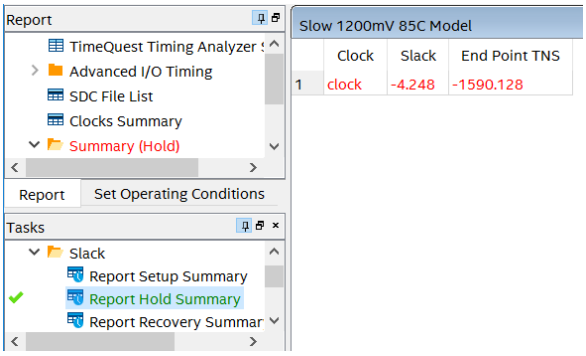


Figure 19: *The hold slack report which reveals the hold violation.*

30) To fix the hold violation, we will first add our SDC file to the Quartus project and then re-run the fitter. It will take into account every constraint we have made when making decisions for how to route the design. To add the SDC file, go back to the Quartus window and click **Project** → "**Add/Remove Files in Project…**".

a) Click "**…**" and navigate to your working directory.

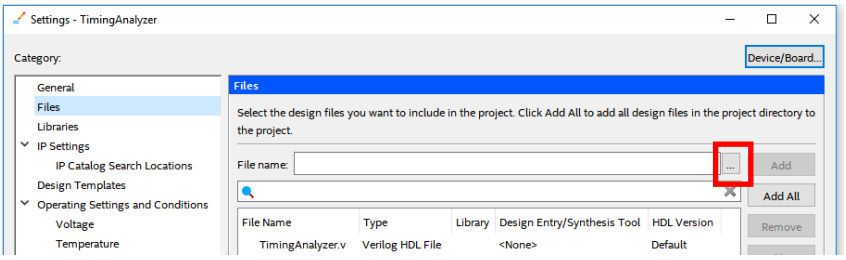


Figure 20: *The "**Add/Remove Files in Project…**" dialogue window.*

b) Change the file type to "**All Files (*.*)**". Find the **.** *sdc* file and select it. Press **Open**.

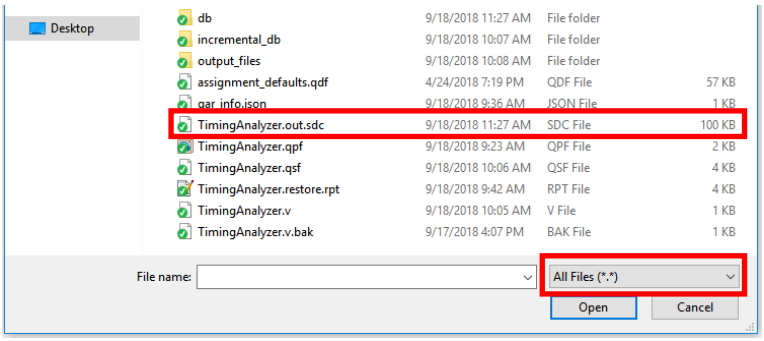


Figure 21: *To select the .sdc file, change the file type to "All Files" and select the only SDC file you see.*

c) Press **Ok**. You have now added the SDC file to your design. Verify it is there by going to your Project Navigator and selecting **Files** from the drop-down tab.

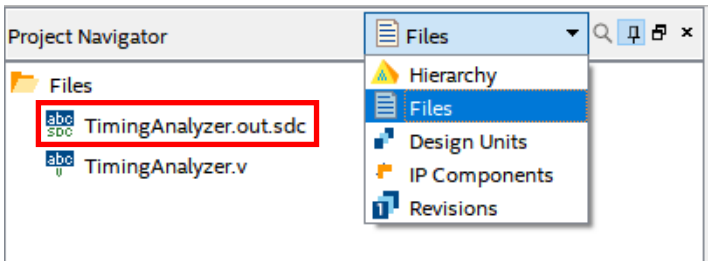


Figure 22: *The Project Navigator showing the list of files in the project.*

d) Finally, run the Fitter again. This time the Fitter will make decisions while considering all the user entered timing constraints.

31) Open Timing Analyzer again. In the Tasks pane, double-click "**Update Timing Netlist**" and then double-click "**Report Hold Summary**." The design should now pass both setup and hold timing requirements!

32) Temperature and voltage have known effects on the speed of a circuit. Quartus by default will assume "Slow 1100mV 85C Model" (the 1200 mV shown in the image below is for another board). We can change this using the "**Set Operating Conditions**" tab. You will see additional options (Slow/Fast, 85C/0C). Select each new model and then double-click "**Report Setup Summary**" in the Tasks pane to compute the setup slack for that model. The corresponding model name under the "Summary (Setup)" folder in the Report pane will change from yellow to black or red.

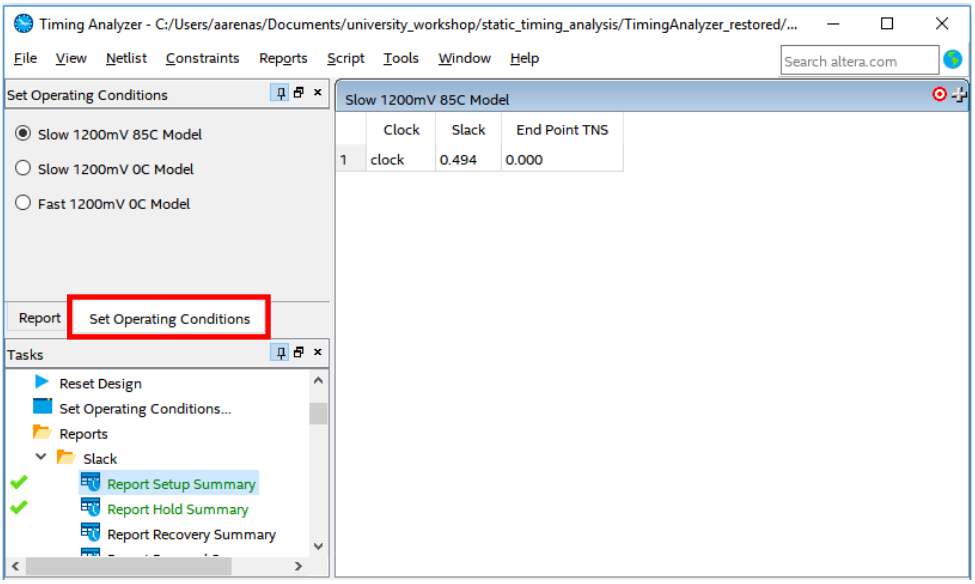


Figure 23: *Clicking Set Operating Conditions will allow the designer to choose from one of three conditions Quartus makes timing calculations based off of. The numbers will vary depending on the frequency/period you selected for your clock.*

33) Now use the “Summary (Setup)” reports to answer the following questions:

a) Which of the options has the most setup slack?

b) How does increasing voltage speed change the speed of a circuit?

c) How does increasing temperature change the speed of a circuit?

34) Repeat Steps 32-33 for hold slack.