

# ECEN3763 - Homework 11

**Spring, 2022**

Due Monday April 4th, end of day – 15 Points

The purpose of this lab is to familiarize yourself with the Quartus Timing Analyzer tool, and to work through a simple design using the concepts of Setup Slack timing and Hold Slack timing.

**After completing this assignment, submit this document to Canvas with the questions answered (or create a separate answer sheet and submit that, either approach is fine).**

**You many not see the exact timing values displayed later in this document, just report what you see in your project.**

A timing cheat sheet is provided on the next page for your reference.

Work through this lab and answer questions where indicated. I encourage you to not speed through this Lab but to take the time to understand what you see and what is being asked. There is no more critical skill needed to be a successful FPGA designer than a competent understanding of timing and how timing information is reported.

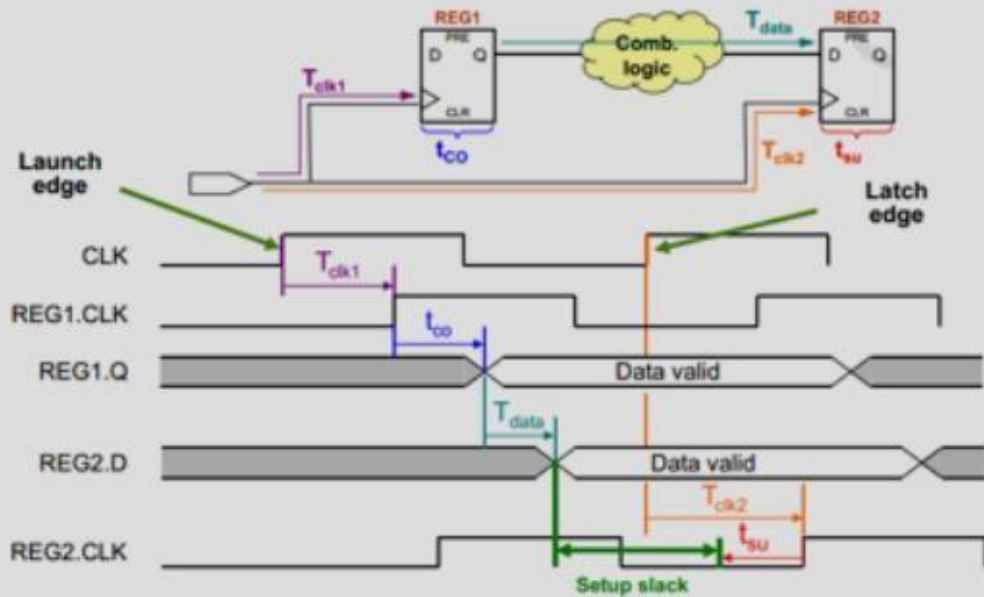
Quartus provides a tool called Timing Analyzer that can be used to create timing constraints for a design, and to allow you to analyze design timing to whatever level of detail you wish. You should use Timing Analyzer to assist you in the creation of timing constraints, and also to analyze failing timing path so that you can make the appropriate modifications to achieve timing closure.

FPGA vendors, and Intel is no exception, report timing values in a format that matches their internal representations. The reports are often difficult to decipher, especially if you are attempting to understand every detail. Areas of confusion will be pointed out as you work through this lab. We will not have time in this class to become experts in all aspects of timing analysis, or experts in Quartus Timing Analyzer reporting. However, if you have specific questions please contact the instructor or TA and we will work to get you answers.

$Setup\ Slack = Min\ Data\ Required\ Time\ (setup) - Max\ Data\ Arrival\ Time$

$Data\ Required\ Time(setup) = Latch\ Edge + \text{clock delay to destination register} - \text{setup time}$

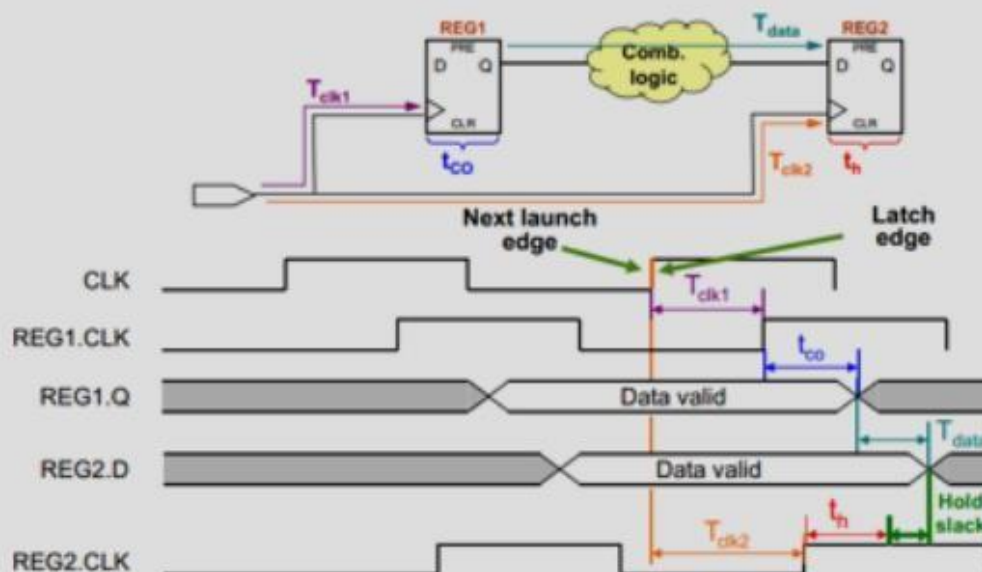
$Data\ Arrival\ Time = Launch\ Edge + \text{clock delay to source register} + \text{clock to out time} + \text{max propagation delay between registers}$



$Hold\ Slack = Min\ Data\ Arrival\ Time - Max\ Data\ Required\ Time\ (hold)$

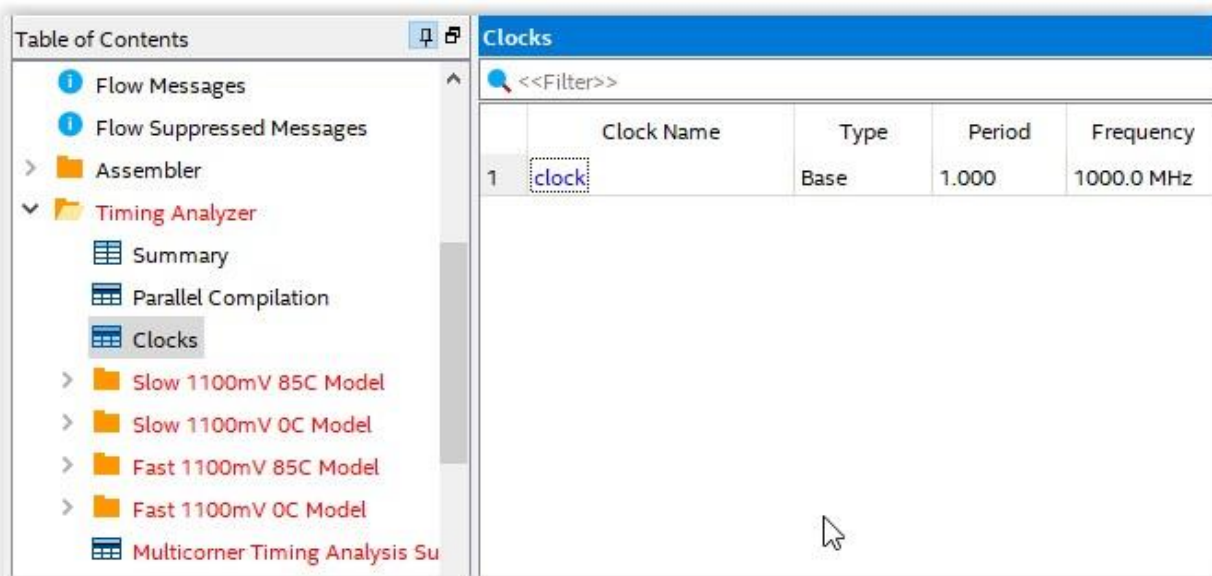
$Data\ Required\ Time(Hold) = Latch\ Edge + \text{clock delay to destination register} + \text{hold time}$

$Data\ Arrival\ Time = Latch\ Edge + \text{clock delay to source register} + \text{clock to out time} + \text{min propagation delay between registers}$



## Part 1

1. Create a new Quartus project, and **select the device used on the DE10-Standard board (5CSXFC6D6F31C6)**. You will not use your DE10-Standard board for this lab, so pin assignments are not necessary. The Verilog source file for this project is ECEN3763\_Homework\_Week11.sv (provided). Review the file to understand the design. Note the use of **synthesis keep** directives which prevent the compiler from optimizing away any of the logic, in order to help us better understand what Timing Analyzer reports. **Make sure you do not have a .sdc file as a project source**, as we will create a constraint file later in the assignment.
2. Compile the design in Quartus. Notice that the Timing Analyzer header in the Table of Contents is red, indicating timing errors. The reason for this is that Quartus assumes a clock period of 1ns for all clocks that are not properly constrained. Since we do not have a .sdc file in the project, our clock period is assumed to be 1 ns. You can see this by selecting the Clocks section under Timing Analyzer in the Table of Contents.

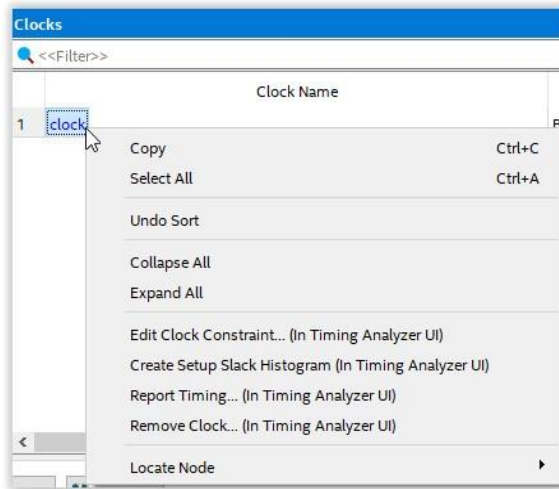


Flow Messages
Flow Suppressed Messages
Assembler
<b>Timing Analyzer</b>
Summary
Parallel Compilation
<b>Clocks</b>
Slow 1100mV 85C Model
Slow 1100mV 0C Model
Fast 1100mV 85C Model
Fast 1100mV 0C Model
Multicorner Timing Analysis Su

	Clock Name	Type	Period	Frequency
1	clock	Base	1.000	1000.0 MHz

3. Right click on clock in the Clocks section above, and select Report Timing.



## Part 2

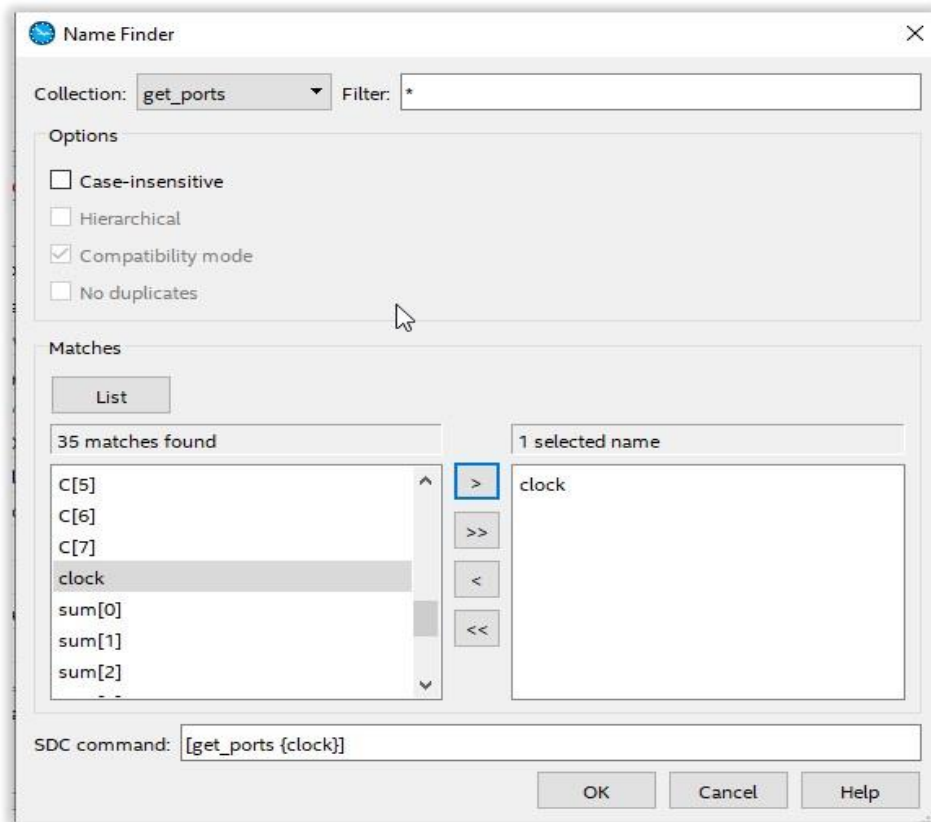
1. Timing Analyzer opens, and a Report Timing window is displayed. **In the From clock: drop down, select or enter clock. Check the Show Routing box.** At the bottom of the Report Timing window, notice that a Tcl command has been created that shows the command that will be run when the Report Timing button is pressed. As you become more familiar with Quartus and Timing Analyzer, you may want to create your own scripts to automate certain steps, so you can use the Tcl command from this window to help with creation of your scripts. For now, we will just press the Report Timing button.

**Question 1:** What is the worst case Setup Slack value? -0.999

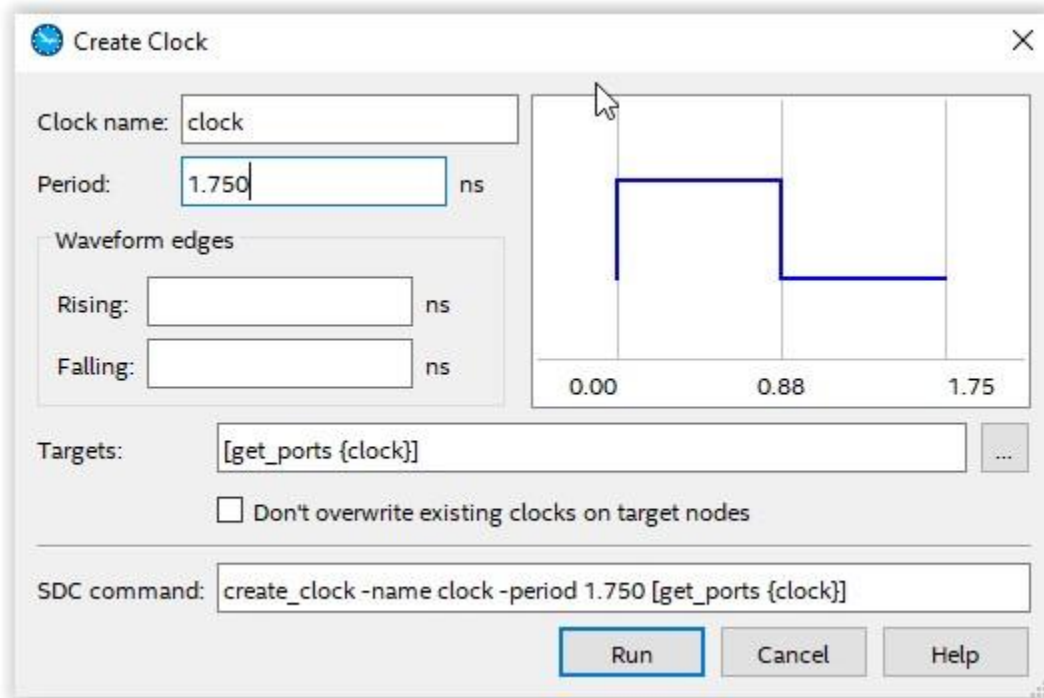
The worst case value is the first value displayed.

Note: By selecting From clock To clock, you are looking at the timing for all paths that transfer data between registers clocked by the clock signal. By default, only 10 paths are reported, but you can look at as many or as few paths as desired. By reviewing the Data Arrival Path and Data Required Path windows that are displayed, you can see all the details of both timing paths.

2. At this point we probably don't care what Timing Analyzer is reporting, because Timing Analyzer is assuming a 1ns period for clock. We need to provide a clock period constraint that matches the clock in our design.
3. Assume we want our clock period to be 1.75ns, which corresponds to approximately 570Mhz. To create this constraint, go to Constraints > Create Clock, and the Create Clock window opens. For Clock name:, enter clock. For Period, enter 1.750, and for Targets, click the 3 dots to open the Name Finder Window. Click the List button, select clock and add it to the selected signal list, and notice that the SDC command is created. Close the window by clicking OK.

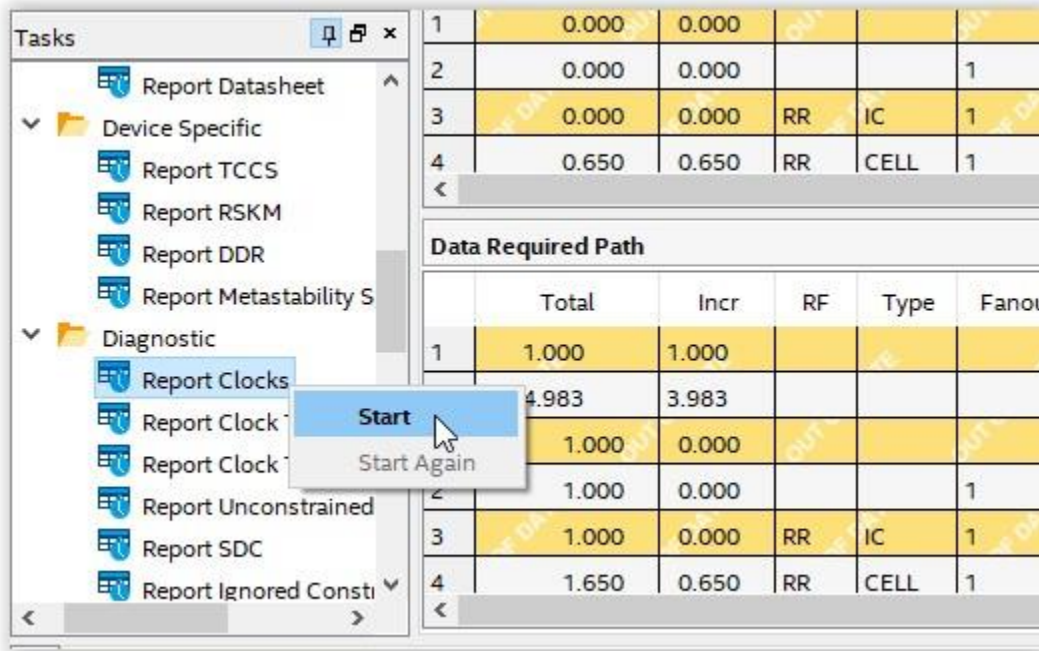


The Create Clock window will look like this.



Press the Run button. The Timing Analyzer main screen will show a lot of yellow highlights, which indicates that a constraint has been changed and that the report needs to be regenerated.

4. Locate Report Clocks in the Tasks pane, right click and select Start. To create a new timing report, select Reports > Custom Reports > Report Timing. From clock and To clock should be set to clock



You will find the setup values in the Report pane (above the Tasks pane), Setup should be red.

**Question 2:** What is the worst case Slack reported? -0.249

Note: If you checked the Show Routing box, the timing reports will contain a lot of entries. Sometimes this is useful, sometimes not. If you want a simplified report, rerun the report with the Show Routing box unchecked.

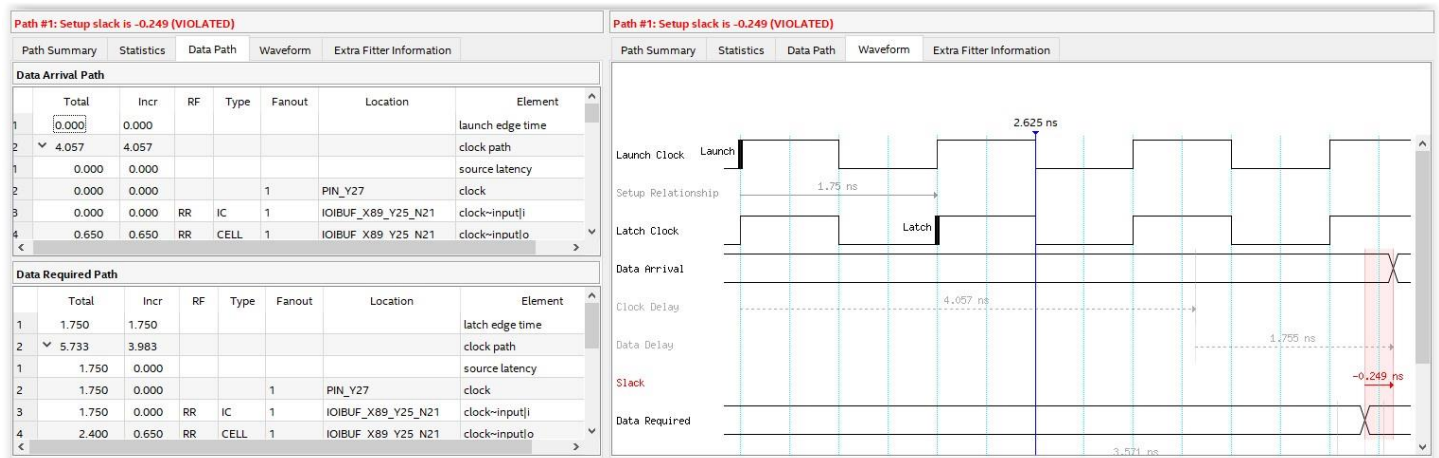
### Part 3

You will now use Timing Analyzer to see how the timing information is provided, and how the Slack calculations are done.

The information displayed in the various panes of Timing Analyzer will match the specific path that is highlighted in the Summary of Paths tab. Highlight Line 1 so we you see the timing information for the worst case path (the path with the largest negative Slack).

The center section of the Timing Analyzer display allows you to display different information. To begin, make sure that you select the Waveform tab on the right side of the screen.





1. Start by selecting the Path Summary tab on the left side of the view above. You can see information on the failing path, the launch and latch clocks, the Data Arrival and Data Required Times, and the resulting Slack (which is a negative value in this example, meaning the design has a timing violation).

From Node: `reg_B[1]`  
To Node: `reg_sum[8]`

**Question 3:** What is the worst case path (from/to)? \_\_\_\_\_

2. Look at the Statistics tab. A given data or clock path delay is the sum of logic (Cell) and interconnection (IC) delays, and this view shows the percentages in the % of total column. If you are trying to determine how to address a timing problem, a path that has a high percentage of logic delay can usually be recoded to modify the logic created (or maybe you add pipeline registers). This usually indicated a large number of logic element blocks in a path. Paths that are a large percentage of routing delay can sometimes be improved by doing manual placement of logic blocks, to bring those blocks closer together.

Path Summary		Statistics	Data Path	Waveform	Extra Fitter Information		
	Property	Value	Count	Total Delay	% of Total	Min	Max
5	▼ Physical Delays						
1	▼ Arrival Path						
1	▼ Clock						
1	IC		3	2.566	63	0.000	2.075
2	Cell		3	1.491	37	0.382	0.650
2	▼ Data						
1	IC		9	0.244	14	0.000	0.244
2	Cell		10	1.511	86	0.000	0.858
3	uTco		1	0.000	0	0.000	0.000
2	▼ Required Path						
1	▼ Clock						
1	IC		3	2.155	60	0.000	1.694
2	Cell		3	1.416	40	0.356	0.650

3. Select the Data Path tab on the left hand side of the Timing Analyzer screen.

Path Summary		Statistics	Data Path	Waveform	Extra Fitter Information		
Data Arrival Path							
	Total	Incr	RF	Type	Fanout	Location	Element
1	0.000	0.000					launch edge time
2	> 4.057	4.057					clock path
3	> 5.812	1.755					data path
<							
Data Required Path							
	Total	Incr	RF	Type	Fanout	Location	Element
1	1.750	1.750					latch edge time
2	> 5.733	3.983					clock path
3	5.563	-0.170					clock uncertainty
4	5.563	0.000		uTsu	1	FF_X74_Y4_N25	reg_sum[8]

The top half of the display shows the details of the Data Arrival Path, and the bottom half shows the details of the Data Required Path.



The Data Arrival Path is defined as the latch clock delay + tco + the data path delay. The launch clock delay is shown to be 4.057ns, and the data path delay is 1.755ns, for a total data arrival time of 5.812ns.

Data Arrival Path							
	Total	Incr	RF	Type	Fanout	Location	Element
2	4.057	4.057					clock path
1	0.000	0.000					source latency
2	0.000	0.000			1	PIN_Y27	clock
3	0.000	0.000	RR	IC	1	IOIBUF_X89_Y25_N21	clock~input i
4	0.650	0.650	RR	CELL	1	IOIBUF_X89_Y25_N21	clock~input o
5	1.141	0.491	RR	IC	1	CLKCTRL_G10	clock~inputCLKENA0 ir
6	1.523	0.382	RR	CELL	34	CLKCTRL_G10	clock~inputCLKENA0 o
7	3.598	2.075	RR	IC	1	FF_X74_Y4_N35	reg_B[1] clk
8	4.057	0.459	RR	CELL	1	FF_X74_Y4_N35	reg_B[1]

4. An analysis of the clock arrival time can be done using the expanded view above.

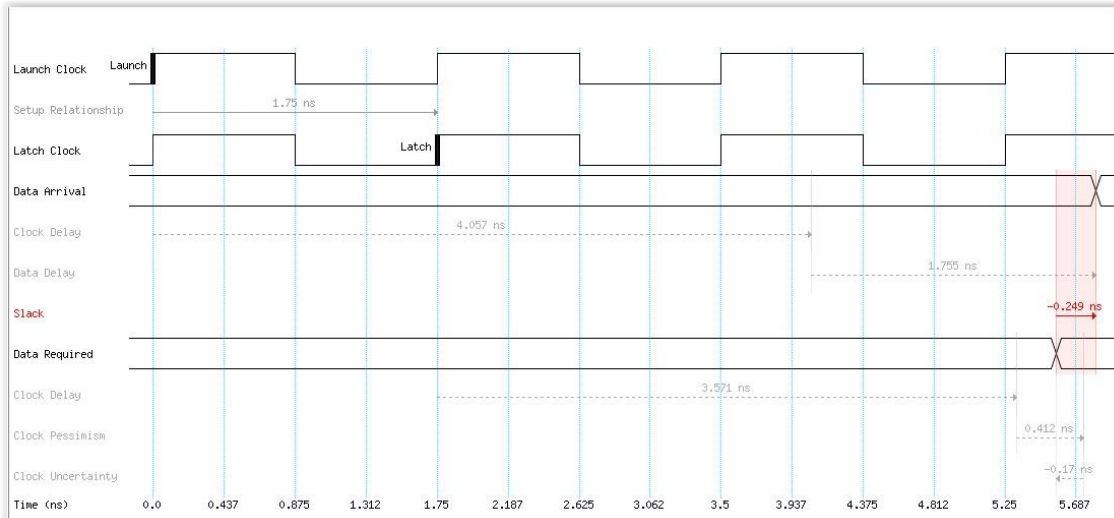
When you see IC in the Type column, the value displayed is a routing or interconnect (IC) delay. CELL in the Type column indicates logic delay, which can be buffers or logic elements.

Lines 3 and 4 show the delay through the input buffer at the I/O cell of the FPGA where the clock is connected. The delay of this input buffer is 650ps. The output of this buffer is then routed to a global clock buffer (Lines 5 and 6). The routing delay is 491ps, and the global clock buffer propagation delay is 382ps.

The delay of the clock from the global clock buffer consists of the main clock route (2.075ns) plus delay as the clock enters the LE and is routed to the destination register (459ps). The sum of all these delays results in the 4.057ns clock path delay.

5. Expanding the data path section of the Data Arrival Path window shows many rows of delays, and many of the delays are 0ns. This highlights the situation where the Intel timing database often combines look up table delays with route delays to adjacent look up tables, which makes the table difficult to decipher. You may never need to understand all the details in this table, so review the details, but don't worry about identifying everything. The important information is the overall data path delay.

6. Review the waveform view on the right side of the display and answer the following questions.



**Question 4:** What is the relationship of the Latch Edge to the Launch Edge?

The latch edge occurs exactly one clock period after the launch edge. Based on the clock period specified earlier in this lab, there is a 1.75ns delay between the launch and latch edges.

**Question 5:** Why does this design demonstrate negative slack? Try to explain what is happening in words, and not in an equation.

Based on the transmitting and receiving delays computed for the two worst case register connections for this design. The data being transmitted from one register to another needs to arrive at the receiving register 0.249ns earlier than it does. This indicates that the receiving register will not have enough time to ingest the data causing a data transmission error.

7. Two bottom two items shown on the timing diagram are Clock Pessimism and Clock Uncertainty. What are those items?

## Clock pessimism explained by Intel:

Within a timing corner, there are two timing "sub-models", basically a fast and a slow timing model. So when doing setup analysis on a path, it will use the slow sub-model for the source clock and data delay, and the fast sub-model for the destination clock path. For hold analysis on the exact same path, it will do the opposite. If you compare setup and hold analysis on the exact same path, you can visually see different numbers. Part of this is rise-fall variation, but a large component is for On-Die Variation, which models the fact that different paths may not be at the exact fastest or slower allowed at the same time. This is all a good thing.

The problem is that the same clock feeds the source and destination. It's impossible to have on-die variation and rise/fall variation on the exact same signal, so it goes up to the point until the clock tree split, figures out how much pessimism was added because of these two models, and removes it. Again, it makes timing easier and is correct.

## Got that?

There is a launch clock and a latch clock. The timing tool will make worst case assumptions on delays for both clocks, in order to model timing in a conservative manner. As mentioned above for setup time analysis, the launch clock will be modeled with the slow timing model, and the latch clock will use the faster timing model. However, there is a large section of the overall launch and latch clock paths that share common buffers and routing. For the sections of the clock paths that are identical for the launch and latch clocks, it makes no sense to model the timing as different, the timing must be identical.

Once the launch clock path is analyzed using the slow timing model, and the latch clock path is modeled using the fast timing model, the overall timing is much more pessimistic than reality, since much of the clock paths are identical. The differences on the common portion of the path are then removed, making the calculation match reality, and this removal value is called clock pessimism. Clock pessimism is your friend, and something that must be calculated by the timing tool.

## What about clock uncertainty?

Clock uncertainty is a catch all phrase for (primarily) different types of clock jitter that occur that reduce overall timing margins. We will discuss jitter (briefly) in the lectures. For now recognize that real world imperfections do have small negative impacts on timing margins, and Quartus will provide realistic clock uncertainty values when clock constraints are created by Timing Analyzer.

8. Go to Constraints > Remove Clock, and enter clock, then Run. Next, go to Constraints > Create Clock and set the clock period to 4 ns, and regenerate the timing report.

Note: The Timing Analyzer seems to have a bug, sometimes you run reports and the analyzer tells you there is nothing to report. This is why it is sometimes helpful to first remove an existing clock and then read it, when you are changing the timing.

Question 6: What is the worst case Setup Slack now? 2.001

Question 7: Does the design meeting timing? Yes

### Part 3

1. Before closing Timing Analyzer, look in your project directory. Do you see a .sdc file?

The answer should be no. Even though a clock constraint was created, you need to explicitly write the constraint into an .sdc file, otherwise the constraints may be lost when Timing Analyzer is closed. Timing Analyzer should prompt you to save any .sdc file changes when you close it.

2. Go to Constraints > Write SDC File, and save your .sdc file.

Question 8: What constraints were written into the .sdc file?

```
create_clock -name {clock} -period 4.000 -waveform { 0.000 2.000 } [get_ports {clock}]
```

```
set_clock_uncertainty -rise_from [get_clocks {clock}] -rise_to [get_clocks {clock}] -setup 0.170
set_clock_uncertainty -rise_from [get_clocks {clock}] -rise_to [get_clocks {clock}] -hold 0.060
set_clock_uncertainty -rise_from [get_clocks {clock}] -fall_to [get_clocks {clock}] -setup 0.170
set_clock_uncertainty -rise_from [get_clocks {clock}] -fall_to [get_clocks {clock}] -hold 0.060
set_clock_uncertainty -fall_from [get_clocks {clock}] -rise_to [get_clocks {clock}] -setup 0.170
set_clock_uncertainty -fall_from [get_clocks {clock}] -rise_to [get_clocks {clock}] -hold 0.060
set_clock_uncertainty -fall_from [get_clocks {clock}] -fall_to [get_clocks {clock}] -setup 0.170
set_clock_uncertainty -fall_from [get_clocks {clock}] -fall_to [get_clocks {clock}] -hold 0.060
```

Don't forget, you will need to add the .sdc file as a source file in your Quartus project for future compilations.

3. Finally, add the .sdc file to your project and rerun Quartus. You should not see any timing failures, but will see Unconstrained Paths.