VHDL Assignment #5: Design and Simulation of Binary Coded Decimal Adders

In this VHDL assignment, you will implement an one-digit binary coded decimal (BCD) adder in VHDL and find the critical path of your design using Quartus Timing Analyzer.

1 Learning Outcomes

After completing this lab you should know how to

- Synthesize logic functions
- Use CAD tools and VHDL to implement one-digit BCD adders
- Perform functional simulation
- Find the critical path of digital circuits

2 Prerequisites

Before starting this lab you should

- Read the posted lecture slides (Lecture #16)
- Be familiar with the information in previous coding assignments
- Read Section 5.7.3.

If you need any help regarding the lab materials, you can

- Ask the TA for help during lab session and office hours.
- Refer to the text book. In case you are not aware, Appendix A "VHDL Reference" provides detailed information on VHDL.
- You can also refer to the tutorial on Quartus and ModelSim provided by Intel (click here for Quartus and here for ModelSim).

It is highly recommended that you first try to resolve any issue by yourself (refer to the textbook and/or the multitude of VHDL resources on the Internet). Syntax errors, especially, can be quickly resolved by reading the error message to see exactly where the error occurred and checking the VHDL Reference or examples in the textbook for the correct syntax.

3 VHDL Description of a One-Digit BCD Adder

In this section, you will implement a one-digit BCD adder in VHDL. A one-digit BCD adder adds two four-bit numbers represented in a BCD format. The result of the addition is a BCD-format 4-bit output, representing the decimal sum, and a carry that is generated if this sum exceeds a decimal value of 9 (see slides of Lecture #16). You are allowed to use either behavioral or structural style of coding for your implementation. You are encouraged to base your code on the VHDL code in Section 5.7.3 of the textbook, so that you learn about conditional signal assignments (these are explained in detail in the same section as well as in the Appendix in Section A.7.4).

Using the following entity definition:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
entity firstname_lastname_bcd_adder is
    Port (A          : in  std_logic_vector (3 downto 0);
          B          : in  std_logic_vector (3 downto 0);
          S          : out std_logic_vector (3 downto 0);
          C_out      : out std_logic);
end firstname_lastname_bcd_adder;
```

After you have implemented the one-digit BCD adder in VHDL, you are required to test your circuit. Write a testbench code and perform an exhaustive test for your VHDL description of the one-digit BCD adder.

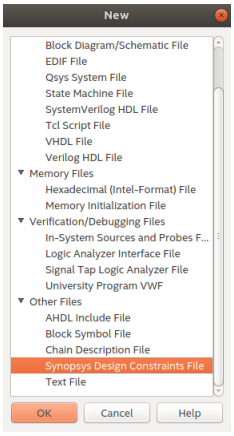
4 Critical Path of Digital Circuits

In this part, you will learn how to use the Quartus CAD tool to determine the delay of a given path in digital circuits. To this end, in this section, we use the ripple-carry adder circuit (that you designed in VHDL assignment #4) as the "circuit under examination".

Follow the instructions described in VHDL Assignment #1 to create a project. Make sure to select the **Cyclone V** family of FPGAs, with the following part number: **5CSEMA5F31C6** when creating a project. Once created, import the VHDL description of your digital circuit into the project and compile it to make sure there are no syntax errors in your design.

The critical path is the longest path in the circuit and limits the speed of the circuit speed. The speed of a digital circuit is measured in terms of latency and throughput. Latency is the time needed for the circuit to produce an output for a given input (*i.e.*, the total propagation delay (time) from the input to the output), and it is expressed units of time. Alternatively, throughput refers to the rate at which data can be processed. In this assignment, we only consider the latency as a metric to measure the speed of the circuit. In general, digital circuits are subject to timing constraints dictated by the target application. Whether a circuit meets these timing constraints can only be known after the circuit is synthesized. After synthesis is performed, the designer can analyze the circuit to determine whether timing constraints were satisfied using the term **slack**. Slack is the margin by which a timing requirement is met or not met; it is the difference between the required arrival time and the actual arrival time. A positive slack value indicates the margin by which a requirement was met. A negative slack value indicates the margin by which a requirement was not met.

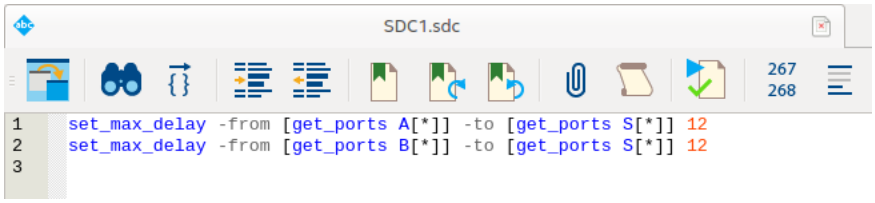
To insert timing constraints in Quartus, select "Synopsys Design Constraints File" from the "File–>New" menu.



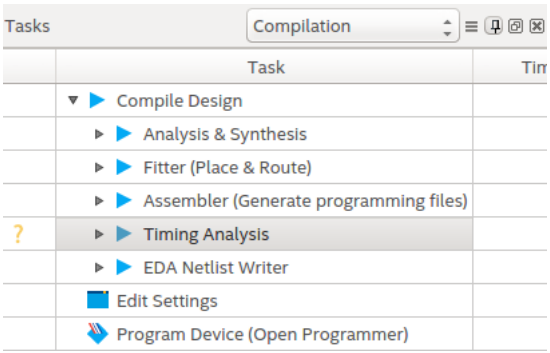
The maximum delay can be specified in the Synopsys Design Constraints File using the following command:

set_max_delay -from [get_ports <name_of_input_port>] -to [get_ports <name_of_output_port>] <time in nano seconds>

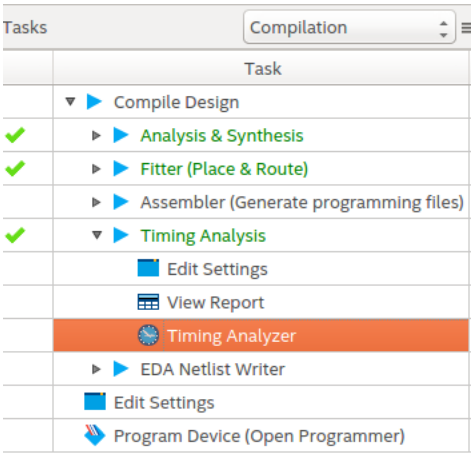
For example, we can specify a maximum delay of 12 ns for all the possible paths from the inputs of the ripple-carry adder to its outputs as shown below.



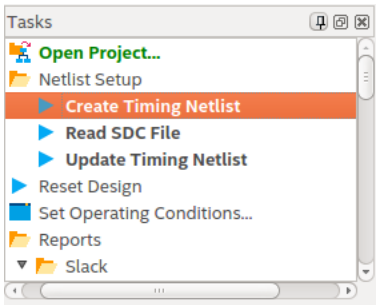
Once the timing constraints are inserted, save the file with the name "firstname_lastname_sdc.sdc". Recompile your design by double clicking on "Timing Analysis" in the Tasks window of Quartus. *Before recompilation, make sure that the .sdf file is added to the project.*



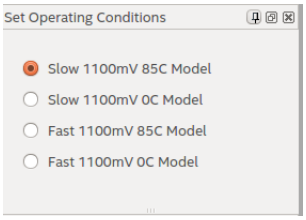
The Timing Analyzer will read the .sdc file and use the constraint information when performing timing analysis. Once a green check mark appears next to "Timing Analysis", double click on "Timing Analyzer" under "Timing Analysis" to open the Timing Analyzer tool.



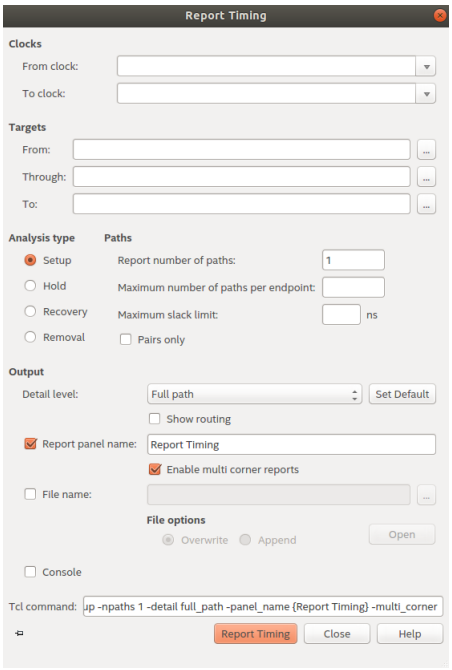
In the Tasks window of the Timing Analyzer tool, double click on "Create Timing Netlist", "Read SDC File" and "Update Timing Netlist" icons, respectively.



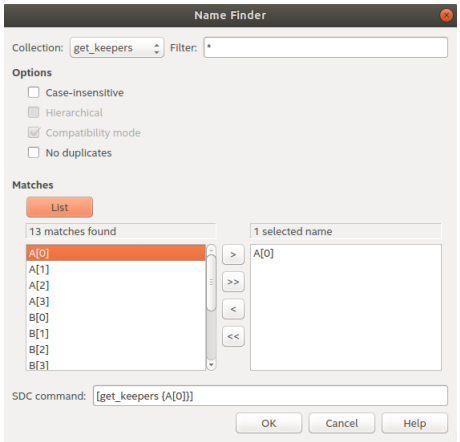
Once completed, the colors of the icons will become green. Before measuring the delay of your design, you should specify the operating conditions of the FPGA device. This can be simply set by selecting one of the possible operating conditions listed in the "Set Operating Conditions" in the Timing Analyzer tool.



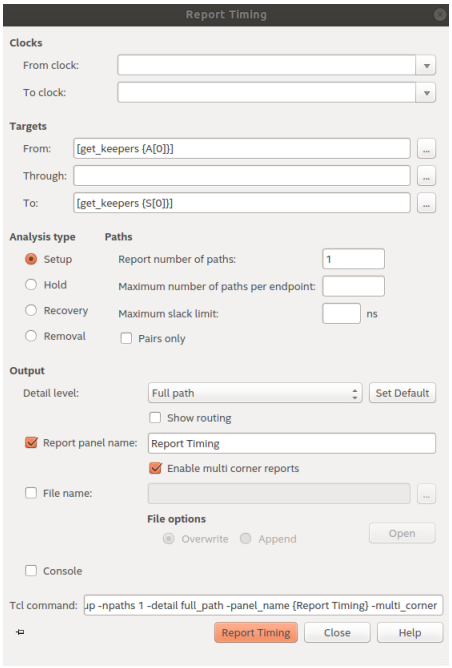
To report the delay of different paths from inputs to outputs, select "Report Timing ..." from the "Reports –> Custom Reports" menu.



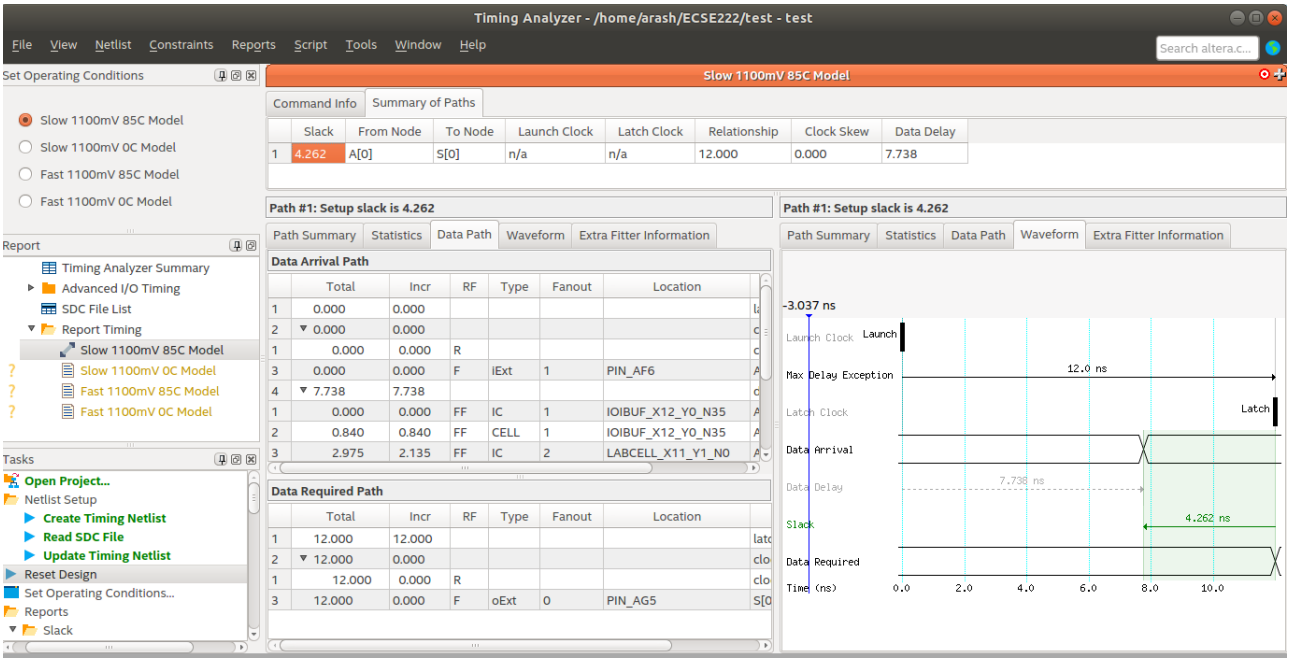
Since no clock signal is associated with your design, we only specify the beginning of the target path(s) by clicking on "From" and the end of the target path/paths by clicking on "To" under the section labeled as "Targets". In the "Name Finder" window that pops up, click on "List" to list all the I/O signals of your design. Select (double click on) a signal (or signals) to determine the beginning of the path that you want to examine and click "OK". Repeat the same procedure to determine the end of the path.



As an example, we can examine the path from the LSB of the input A (*i.e.*, A(0)) to the LSB of the output S (*i.e.*, S(0)) as shown below.

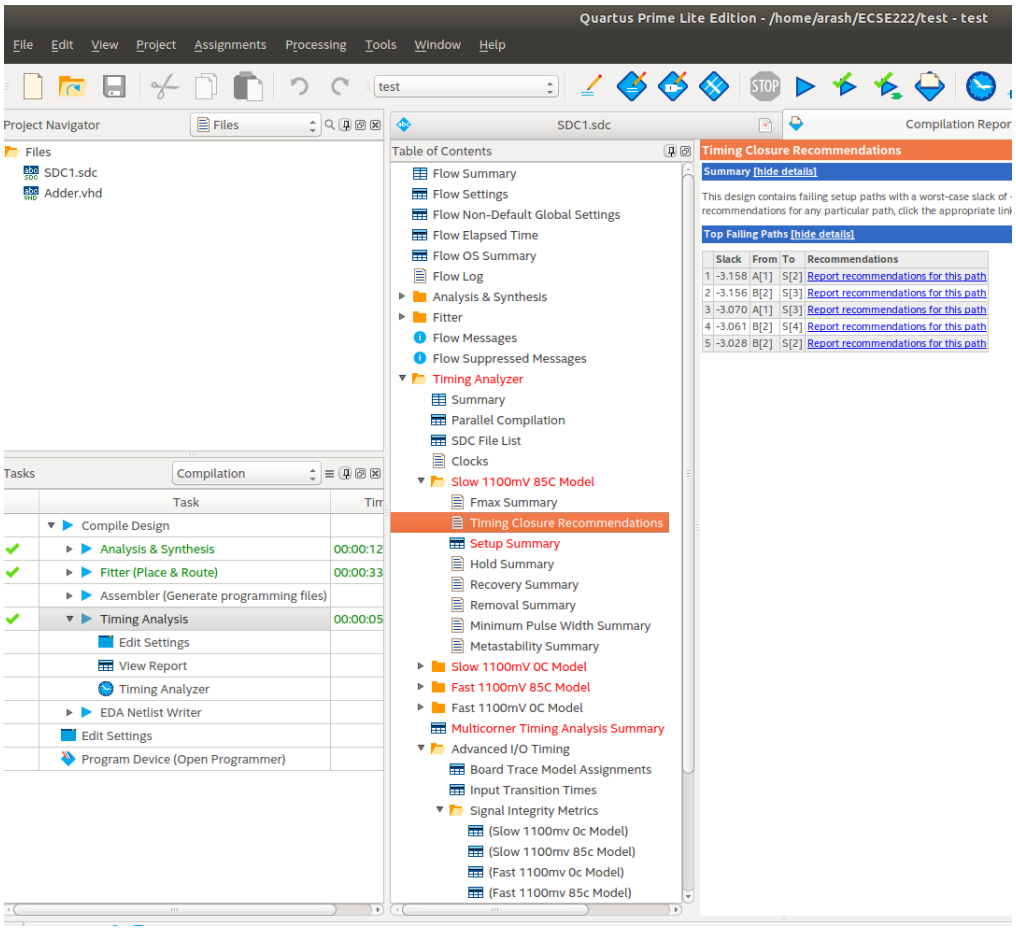


Now, click on "Report Timing" to obtain timing information for the specified path. The delay of the specified path is denoted "Data Delay" in the section entitled "Summary of Paths". The positive value of the slack denotes the difference between the delay of the path (*i.e.*, Data Delay) and the timing constraint inserted in the .sdf file (*i.e.*, 12 ns). This information is also visualized under the "waveform" tab.



To find the critical path of your design, you should examine all possible paths from all inputs to all outputs and find the one with the longest delay. However, using this "exhaustive search" method is very time consuming as the number of I/O ports increases. To limit the number of paths under examination, we reduce the target delay value in the .sdf file so that a timing violation occurs. For instance, we reduce the target delay constraint of the "circuit under examination" from 12 ns to 5 ns and recompile the design by double clicking on "Timing Analysis"

in Quartus. In case of a timing violation, Quartus determines the violating path(s) in the compilation summary of the "Timing Analyzer" tool.



In this approach, our search for the critical path will now be limited to the violating paths. Examining the violating paths in the "Timing Analyzer" tool will, therefore, determine the critical path.

5 Deliverables and Grading

Read this section extra carefully!

You are required to submit the following files through myCourses:

- A report that includes the answers to the questions in Section **??**. Use the report template (provided in .docx format) that is attached to the assignment. Make sure that all figures are clearly visible and contain relevant information (*e.g.*, axes and relevant signal names). Save your report in a .pdf format (include your name in the filename).
- All design files (.vhd)
- All schematic files (.bdf)
- All testbench files (.vht)
- All design constraints files (.sdc)

Per the VHDL assignments submission policy, please note

- For partially submitted assignments, *i.e.*, some of the files (design/simulation/report) are missing, the penalty is 25% of the full mark of the assignment.
- For assignments where all the design/simulation files are missing, *i.e.*, only a report was submitted, the penalty is 50% of the full mark of the assignment.
- For not submitted assignments, *i.e.*, all files are missing, the penalty is 100% of the full mark of the assignment.

6 Questions

1. Show representative simulation plots of the one-digit BCD adder circuit for all the possible input values. (2 marks)
2. Perform timing analysis and find the critical path(s) of the one-digit BCD adder circuit for the Fast 1100mV 85C Model. Show the obtained timing waveform(s) of the critical path(s) that you found. (2 marks)