Digital Systems

Prof. James Clark

ECSE 325
Lab #3: Timing constraint specification and timing analysis using TimeQuest

Winter 2018

Introduction

In this lab, you will learn how to specify timing constraints and perform static timing analysis of the synthesized circuit using the TimeQuest timing analyzer. You will also learn two techniques to reach timing closure for a time-critical circuit.

Digital Filters

Digital filters are a very important part of digital signal processing (DSP). Digital filters have two uses: signal separation and signal restoration.

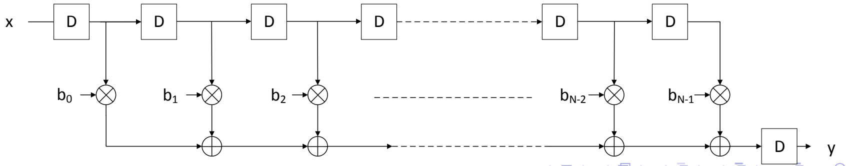
- Signal separation is needed when a signal has been contaminated with interference, noise, or other signals. For example, imagine a device for measuring the electrical activity of a baby's heart while still in the womb. The raw signal will likely be corrupted by the breathing and heartbeat of the mother. A filter might be used to separate these signals so that they can be individually analyzed.
- Signal restoration is used when a signal has been distorted in some way. For example, an audio recording made with poor equipment may be filtered to better represent the sound as it actually occurred. Another example is the deblurring of an image acquired with an improperly focused lens, or a shaky camera.

FIR Filters

The most straightforward way to implement a digital filter is by convolving the input signal with the digital filter's impulse response. Filters carried out by convolution are called Finite Impulse Response (FIR) filters. An FIR filter is a filter whose response to any finite length input is of finite period. For a causal discrete-time FIR filter of order N, each value of the output sequence is a weighted sum of the most recent input values:

$$y(n) = \sum_{i=0}^{N} b_i \times x(n-i),$$

where $x(n)$ and $y(n)$ are input and output signals, respectively. The weights are also denoted by b_i.



Root Square Mean Error

Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit. Root mean square error is commonly used in signal processing.

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=0}^{N-1}(\hat{y}_i - y_i)^2}{N}}$$

where

- $\hat{y}_i$ is the estimated value
- y_i is the actual value
- N is the number of samples

FIR Filters–Implementation

In this lab, you will implement a bandpass FIR filter with order 25 (25-tap FIR filter) to restore a sine wave corrupted by a white noise. You are provided with three text files containing the filter's input, output signals and weights in floating-point format. Given the block diagram of an N-tap filter in Fig. 1, implement the 25-tap FIR filter in VHDL while representing the filter's input, output signals and weights in the fixed-point representations (1,15), (2,15) and (1,15), respectively.

Note: Use constant parameters to represent the weights.

```vhdl
entity gNN_FIR is
port ( x      : in std_logic_vector (15 downto 0); -- input Signal
       clk    : in std_logic; -- clock
       rst    : in std_logic; -- asynchronous active-high reset
       y      : in std_logic_vector (16 downto 0)); -- output signal
end gNN_FIR;
```

FIR Filters–Simulation

Once you have implemented your design in VHDL, you need to verify its functionality by writing a testbech code. In the testbench code, you will read the given test vector in your testbench and obtain an output test vector. The obtained output signal must match with the given output signal.

When you have finished the VHDL codes, show them to the TA.

Using the TimeQuest Timing Analyzer

To ensure a properly working circuit, the designer must take into consideration various timing constraints. In class we saw that for a register to correctly store an input value, the input must be held stable for a period (called the setup time) before the clock edge, and also for a period (called the hold time) after the clock edge.

Whether a circuit meets these timing constraints can only be known after the circuit is synthesized. After synthesis is done one can analyze the circuit to see if the timing constraints (setup and hold times) are 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 time and the 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.

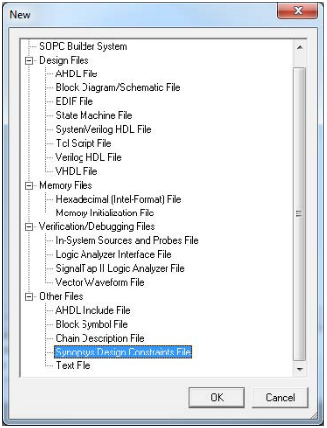
In Quartus II, timing analysis can be done using the TimeQuest Timing Analyzer. Please read pages 7-6 through 7-12 of the document "Altera TimeQuest Timing Analyzer.pdf" before proceeding to the next part of the lab.

Using the TimeQuest Timing Analyzer

From the "File/New" menu item, select "Synopsys Design Constraints File".

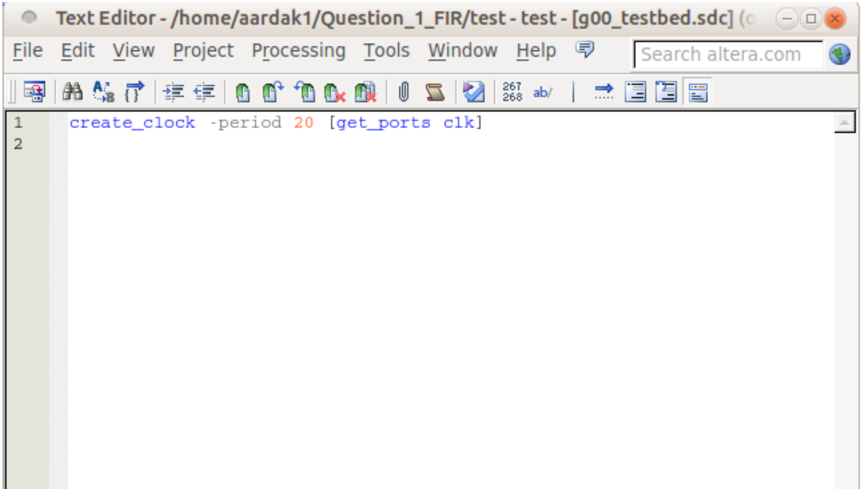
This ".sdc" file is where we can specify various timing constraints for our design.

We will add a single constraint specifying the clock period.



Using the TimeQuest Timing Analyzer

Type the following text into the window, and save the file with the name "gNN_testbed.sdc" where NN is your group number. This will tell the timing analyzer that your clock period is 20 ns.


Text Editor - /home/aardak1/Question_1_FIR/test - test - [g00_testbed.sdc]
File
Edit
View
Project
Processing
Tools
Window
Help
Search altera.com

```
create_clock -period 20 [get_ports clk]
```

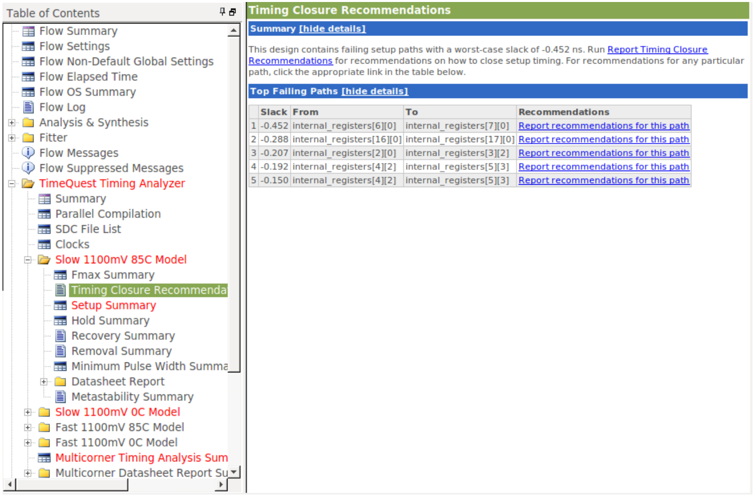
Using the TimeQuest Timing Analyzer

Recompile your design. The Timing Analyzer will read the .sdc file and use the constraint information when doing its analysis.

After the compilation is finished, look at the timing summaries for the **Slow Model** and the **Fast Model**. Pay attention to the **Fmax Summary** (which gives the maximum clock speed) and the **Setup** and **Hold** summaries (which give the slack amounts for the setup and hold constraints). Find the maximum achievable frequency of your design and show it to the TA. Does your design meet the specified timing constraint (i.e., the clock period of 20 ns)?

Using the TimeQuest Timing Analyzer-Timing Violation

In case of any timing violation, Quartus recommends solutions for each violated path in your design. Denote violated paths in your design on Fig. 1 and show them to the TA.



Timing Closure Recommendations

Summary [hide details]

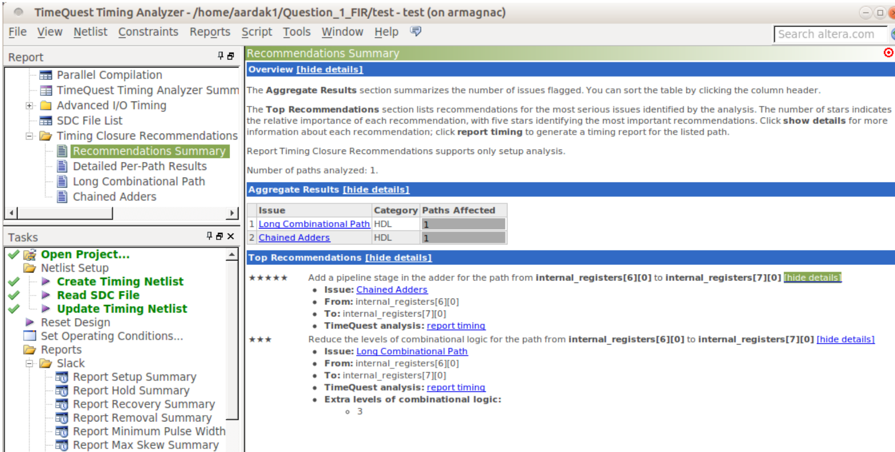
This design contains failing setup paths with a worst-case slack of -0.452 ns. Run Report Timing Closure Recommendations for recommendations on how to close setup timing. For recommendations for any particular path, click the appropriate link in the table below.

Top Failing Paths [hide details]

	Slack	From	To	Recommendations
1	-0.452	internal_registers[6][0]	internal_registers[7][0]	Report recommendations for this path
2	-0.288	internal_registers[16][0]	internal_registers[17][0]	Report recommendations for this path
3	-0.207	internal_registers[2][0]	internal_registers[3][2]	Report recommendations for this path
4	-0.192	internal_registers[4][2]	internal_registers[5][3]	Report recommendations for this path
5	-0.150	internal_registers[4][2]	internal_registers[5][3]	Report recommendations for this path

Using the TimeQuest Timing Analyzer-Timing Violation

Click on the "Report recommendations for this path" to open up TimeQuest Timing Analyzer and get more information on the violated paths. Under the "Recommendation Summary", you can find possible solutions to fix the timing violations.



Overview [hide details]

The **Aggregate Results** section summarizes the number of issues flagged. You can sort the table by clicking the column header.

The **Top Recommendations** section lists recommendations for the most serious issues identified by the analysis. The number of stars indicates the relative importance of each recommendation, with five stars identifying the most important recommendations. Click **show details** for more information about each recommendation; click **report timing** to generate a timing report for the listed path.

Report Timing Closure Recommendations supports only setup analysis.

Number of paths analyzed: 1.

Aggregate Results [hide details]

	Issue	Category	Paths Affected
1	Long Combinational Path	HDL	1
2	Chained Adders	HDL	1

Top Recommendations [hide details]

★★★★★ Add a pipeline stage in the adder for the path from **internal_registers[6][0]** to **internal_registers[7][0]** [hide details]
- **Issue:** Chained Adders
- **From:** internal_registers[6][0]
- **To:** internal_registers[7][0]
- **TimeQuest analysis:** report timing

★★★ Reduce the levels of combinational logic for the path from **internal_registers[6][0]** to **internal_registers[7][0]** [hide details]
- **Issue:** Long Combinational Path
- **From:** internal_registers[6][0]
- **To:** internal_registers[7][0]
- **TimeQuest analysis:** report timing
- **Extra levels of combinational logic:**
  - 3

Using the TimeQuest Timing Analyzer-Timing Violation

Now, you will learn two common solutions to resolve the timing issues of the violated paths.

- Reduce levels of combinational logic for the violated paths
- Redesign architecture and rearrange registers

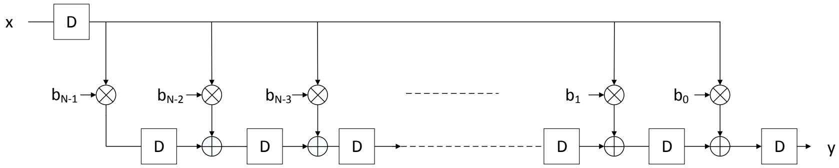
Reduce Levels of Combinational Logic

One way to reduce the levels of combinational logic of the violated paths is to reduce the bitwidth of signals. Use any programming language of your own choice and the given data to find the minimum bitwidth required to keep the quantization root mean square error (RMSE) below 0.27.

Modify your VHDL code to perform the computations with the new bitwidth. Then, recompile your design and use the Timing Analyzer to find the maximum frequency. Does the new design meet the specified timing constraint (i.e., the clock period of 20 ns)? Finally, verify your design using your testbench. Show the obtained results to the TA.

Redesign Architecture and Rearrange Registers

Another solution to increase the frequency of systems is redesigning architecture and rearranging registers. Using the broadcasting form, which is the result of register rearrangement and design modification, is a common approach for hardware implementations of FIR filters. The broadcasting form of FIR filters is shown below.



Redesign Architecture and Rearrange Registers

First, show why the broadcasting form works by writing down equations for both the regular and broadcasting forms and include them in your report. Second, implement the broadcasting form of the FIR filter in VHDL while representing the filter's input, output signals and weights in the fixed-point representations (1,15), (2,15) and (1,15), respectively. Then, recompile your design and use the Timing Analyzer to find the maximum frequency. Does the new design meet the specified timing constraint (i.e., the clock period of 20 ns)? Finally, verify your design using your testbench.

Show the obtained results to the TA.

Writeup the Lab Report

You are required to submit a written report and your code to my-Courses on the first Friday after the Lab3 period at midnight.

- The report in the electronic file should be in PDF format.
- The report should be written in the standard technical report format.
- Document every design choice clearly.
- Everything should be organized for the grader to easily reproduce your results by running your code through the tools.
- The code should be well-documented and easy to read.
- The grader should not have to struggle to understand your design.

Writeup of the Lab Report - cont'd

Please refer to the example lab report from myCourses content for an example lab report template.

Your report must include:

- An introduction in which the objective of the lab is discussed
- For both filter designs:
- The VHDL code you wrote for the 25-tap FIR filter
- Discussion on the resource utilization of your design (i.e. how many registers are used and why? What changes would you respect in the resource utilization if you increased the bit size of the counter?)
- Testbench VHDL code to verify your filter design
- Content of SDC file, screenshots of timing violations (if any), reported maximum clock frequency

Grading Sheet

Group Number:
Name 1:
Name 2:

Task	Grade	/Total	Comments
VHDL for 25-tap FIR		/20	
VHDL for broadcast		/30	
Testbench VHDL		/35	
Maximum Frequency		/15	