ECSE 325

Lab 4 Report

Furkan Ercan, Arash Ardakani

Group 04

Alex Hale, Anna Bieber

260672475, 260678856

April 3rd, 2018

Introduction

The goal of this lab is to learn how to specify timing constraints and perform static timing analysis of a synthesized circuit using the TimeQuest timing analyzer. The circuit used to test these functions is a finite impulse response (FIR) filter implemented in broadcasting form. The code implemented for lab 3 will be adapted to fit the current criteria and testbench as well.

Design

A conditional IF statement is used. If RESET is high, clear the values in the input array of D flip-flops to zero. If the clock is on its rising edge, enter the main logic section. First, calculate the first number of the input array by multiplying the input value by the first coefficient value. Next, use a for loop to calculate all the remaining values of the input array. Finally, after the process block, send the temporary signal to the output, truncating to the correct length of 17 bits.

The code for the FIR architecture is found in Figure 1.

```
    multiply : process(rst, clk)
    begin
        if rst = '1' then
            -- reset array values to 0
            INPUT_ARRAY <= (others=>(others=>'0'));
        elsif(rising_edge(clk)) then
            -- step 1: calculate a(0) by multiplying input value by b_(N-1)
            INPUT_ARRAY(0) <= signed(x)*COEFF_ARRAY(0);

            for t in 1 to 24 loop
                -- step 2: in a for loop, calculate a(1) to a(24) with (x*b_(N-i)+a(i-1))
                INPUT_ARRAY(t) <= (signed(x)*COEFF_ARRAY(t))+INPUT_ARRAY(t-1);
            end loop;
        end if;
    end process multiply;
  y <= std_logic_vector(INPUT_ARRAY(24)(31 downto 15));    -- assign result to output outside of the process block, truncating
end fir_implementation;
```

Figure 1 – FIR architecture

Testbench

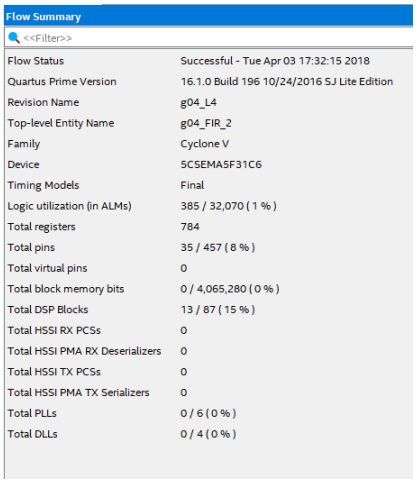
At the top of the testbench file, the empty entity is declared. Next, inside the architecture, the device under test (DUT) is declared with input and output signals matching those of the VHDL design. Internal variables are declared: three files for the inputs and output, one constant for the clock period, and a signal to match each of the signals in the DUT. After the *begin* statement, the FIR is instantiated by mapping the DUT's inputs and outputs to the internal testbench signals. Next, a process is used to generate the oscillating clock signal.

Finally, the process that feeds values to the VHDL design is reached. First, the circuit is reset by waiting for two rising edges of the clock. Next, the files to be read and written are opened in the appropriate mode. In a loop, one line at a time is read from the input files and sent to the *input* and *coefficient* input signals. Lastly, the FIR output is sent to the output file.

The testbench can be found in the folder with this report.

Resource Utilization

As seen in the compilation report (Figure 2), the design uses 784 registers. This is because we have a signal that is an array of length 25, where each index is a 32 bit signed integer. This array requires 784 registers.



Flow Summary

<<Filter>>

Flow Status	Successful - Tue Apr 03 17:32:15 2018
Quartus Prime Version	16.1.0 Build 196 10/24/2016 SJ Lite Edition
Revision Name	g04_L4
Top-level Entity Name	g04_FIR_2
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Final
Logic utilization (in ALMs)	385 / 32,070 (1 %)
Total registers	784
Total pins	35 / 457 (8 %)
Total virtual pins	0
Total block memory bits	0 / 4,065,280 (0 %)
Total DSP Blocks	13 / 87 (15 %)
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	0 / 6 (0 %)
Total DLLs	0 / 4 (0 %)

SDC File

The sdc file allows to test the circuit with various timing constraints. Whether a circuit meets timing constraints can only be known after the circuit is synthesized. The code in the sdc file can be seen in Figure 3. After the compiling and synthesizing we received no timing violations and the maximum frequency was found. This information can be found in Figure 4.

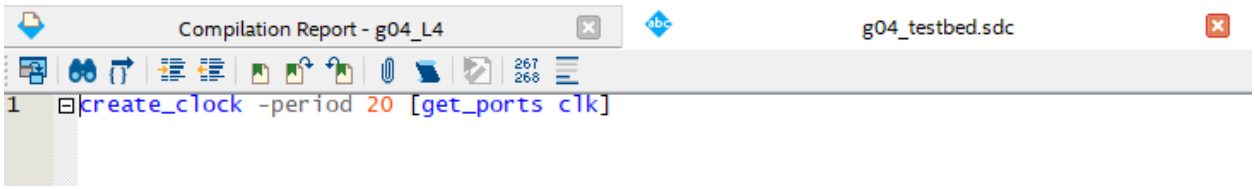
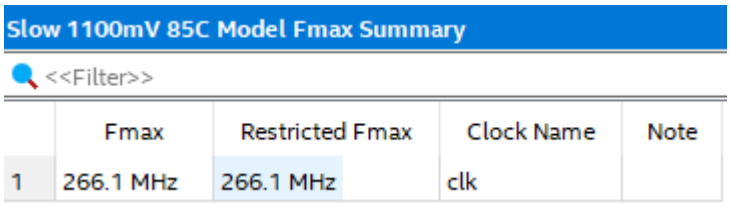


Figure 3 – SDC File



Slow 1100mV 85C Model Fmax Summary

<<Filter>>

	Fmax	Restricted Fmax	Clock Name	Note
1	266.1 MHz	266.1 MHz	clk	

Figure 4 – Max Frequency

Conclusion

This lab taught us how to specify timing constaints on a circuit and to verify that they are being satisfied. The circuit in being analyzed was a FIR filter in broadcast form which was implemented in VHDL with a similar logic to lab 3, the testbench was written to feed input values from files into the filter. To test both

the functionality of the design, its performance was simulated in ModelSim, then the timing constraints were analyzed using the TimeQuest timing analyzer.