Lab2: Exercises

EE332: Digital System Design

Objectives

The objectives of this laboratory are the following:

1. To understand the delay in VHDL codes.
2. To understand the different simulation results under different simulation modes in the FPGA design flow.
3. To understand the differences between the signal assignment and variable assignment.
4. To practice on the developing of test bench for both combinational and sequential circuits.
5. To become familiar with the structural coding style.
6. To investigate how the design affects the critical path of a circuit.
7. To practice on the design of sequential circuits, FSM, and FSMD.
8. To understand and practice on pipeline design and parameterized design.
9. To analyze the design performances.

This lab includes 6 exercises.

Exercise 1: Full Adder Simulation.
(Week 2, report due by noon, i.e., 12:00 noon, on the day of the week 4 class)

Do the Simulation of the VHDL code of Full adder on page 78 of the lecture notes.

Prelab exercises:
Manually derive the behavior simulation result of the full adder, including the waveforms of all intermediate signals and output signals.

During the lab:
Design a test bench with stimulus of A, B and Cin as shown on page 79 in lecture notes. Simulate the VHDL code in behavior simulation, post synthesis simulations and post place and route simulations.

To develop a test bench for a combinational circuit, please refer to Pages 156-158 in lecture notes, together with the example given in Tutorial.

A full report is required for this exercise.
In the report, you may investigate your simulation results from, but **NOT** limited to, the following aspects:

a. If the simulation result is consistent with those shown in lecture notes.
b. If the addition result of the full adder is consistent with common sense, i.e., if the simulation gives the correct addition result?
c. What cause the simulation results different from that of common sense, or in other words, how to revise the design such that the simulation results will be the same as the common sense?
d. Any difference among the 3 simulation (i.e., the behavior simulation, post synthesis simulation and post place and route simulation) results? Why there are such differences?
e. What cause the spikes if any in the output signal?

Exercise 2: Simulation of process
(Week 3)

Do the simulation of the VHDL code of process on Page 130 of the lecture notes.

Prelab exercises:
Manually derive the behavior simulation results, including all the intermediate and output signals, for the following stimulus:

Apply signal x with a period of 80 ns, signal y with a period of 40 ns, and signal z with a period of 20ns. All x, y, and z are 0 initially with duty cycle of 50%.

During the lab:
Design a test bench and simulate the VHDL code in behavior simulation, post-synthesis simulation and post-place-and-route simulation.

Any difference among the 3 simulation (i.e., the behavior simulation, post synthesis simulation and post place and route simulation) results? Why there are such differences?

No report is required for this exercise.

Exercise 3: Design a tree 16-to-4 priority encoder

(Weeks 4 and 5, report due by noon, i.e., 12:00 noon, on the day of the week 6 class)

A priority encoder is a circuit that returns the codes for the highest-priority request. By using a conditional signal assignment or if statements to naturally describe this function, the shape of the specified priority network is a single cascading chain.

Prelab exercise:

Refer to the 4-to-2 priority encoder on Page 83, develop a VHDL code to realize a 16-to-4 priority encoder using a conditional signal assignment. The inputs of the encoder are 16 bits, and the outputs of the encoder are a 4-bit code that is the binary code of the highest-priority active input bit and a 1-bit activity flag. Sketch the conceptual implementation of the design.

During the lab:

To shape the layout of the circuit to a tree form so that to reduce the critical path of the designed 16-to-4 priority encoder.

Develop the block diagram of the design.
Based on the block diagram, develop the VHDL code of the design using component instantiation.
Do the simulation of the design.
Compare the layout and performance of the tree design with that of the cascading design.

Note: The block diagram of the design is supposed to be developed before the VHDL code is developed. Therefore, do not use the synthesized result to replace the block diagram.

A simple report is required for this exercise.

In the report, you may investigate your design from, but **NOT** limited to, the following aspects:

a. block diagrams and codes showing your design
b. Theoretical analysis of the critical paths of the two designs.
c. Simulated critical delays of the two designs, at different simulation modes.
d. The comparison, and if the comparison is consistent with your expectations? If not, what's the reasons?

Exercise 4: Design and realize a three-digit decimal counter.
(Week 6, report due by noon, i.e., 12:00 noon, on the day of the week 8 class**)**

The design should have the following functions:
1) An asynchronous reset function using a push button to initialize the counter to 000;
2) A synchronous load function to set a start number of the counting; use a switch to turn on or turn off the load mode, and use 12 switches to set the start number.
3) The counter increases by 1 for every 1 second during the counting mode.
4) Display the counting results on the 12 LEDs (4 LEDs represent 1 decimal digit).

In this exercise, the counter is required to increase 1 for every 1 second. However, the FPGA board provides only a 100MHz clock source, which is 10^8 times faster than the required clock. You need design circuits to slow down the clock, such that the counter may increase at a required rate.

Prelab exercise:
Design a component of a frequency divider that slow down the output by 10^8 times for a clock input, i.e., the frequency divider produces an output pulse for every 10^8 input pulses. Sketch the conceptual diagram and develop the VHDL code of the frequency divider.

During the lab:
There are two ways to make use of the output of the frequency divider in your design, one is to use its output signal as the clock, and the other is to use it as an enable signal. The first one actually is an asynchronous design, i.e., the registers in the circuits are controlled by more than 1 clock, and we should avoid such methods especially in a high-speed circuit design. Therefore, the second method serving as an enable signal should be used.

Refine the conceptual diagram of the 3-digit decimal counter on page 209, and develop the counter as a component.

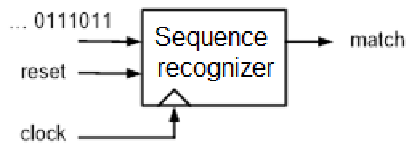
A top module, instantiating the frequency divider and the 3-digit decimal counter, realizes the required functions.

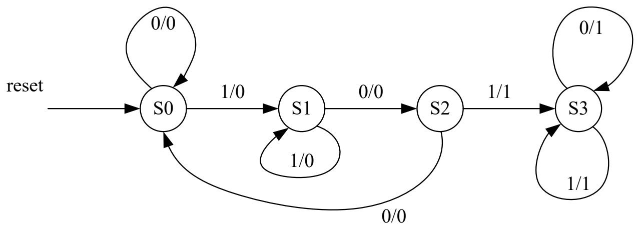
Two-segment coding style is required for sequential circuits.
For the test bench design, refer to the example in pages 216 – 221.
Implement and show your design on the Nexys 4DDR board.

A simple report is required for this exercise.
The report should include:
1. The conceptual diagram of the frequency divider, the refined conceptual diagram of the 3-digit decimal counter, and top level structural diagram of the overall design;
2. The codes of the two components in two-segment coding style, and the code of the top module. Make sure that the code consists with the diagrams.

Exercise 5. FSM and FSMD
(Week 9, report due by noon, i.e., 12:00 noon, on the day of the week 11 class)

1. Consider the state machine of a sequence recognition shown below. The state machine has one input besides the reset and clock signal. The input is a bit-serial input, feeding in a sequence of binary digits.





(a) Find out the sequence that the FSM recognizes, and describe the behavior of the FSM.
(b) Derive the VHDL code in two-segment coding style for the FSM.
(c) Simulate the VHDL model.

2. The Fibonacci function is defined as

$$fib(n) = \begin{cases} 0 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ fib(n-1) + fib(n-2) & \text{if } n > 1 \end{cases}$$

Implement this function in hardware. Assume that *n* is a 6-bit input and interpreted as an unsigned integer. Note that *fib*(63) is 6557470319842.
(a) Derive an algorithm state machine chart.
(b) Sketch the conceptual diagram of the FSMD.
(c) Develop the VHDL code in two-segment style.
(d) Simulate the VHDL model.
(e) Analyze the timing of the circuit.
(f) Implement this function on the FPGA board. You may design the input/output interfaces.

A simple report regarding the Fibonacci function is required.
The report should include, but **NOT** limited to, the followings:
(1) the whole procedure for the FSMD design of Fibonacci function (Refer to the 4 steps of the FSMD design procedure on Pages 241-251)
(2) any possible improvement. (Refer to possible improvements on Pages 252-259.)
(3) Analysis and comparison, if any.
(4) Your I/O interface.

Exercise 6. Performance and cost comparison of 16-bit multipliers.
(Week 11, report due by noon, i.e., 12:00 noon, on the day of the week 13 class)

(1) Simulate and synthesize the two designs of a 16-bit adder-based multipliers (referring to the combinational design on P185-191 and the repetitive-addition design on P241-259). Examine the synthesized RTL schematics, and compare the performance of the different designs in terms of the device utilization, delay, etc.

(2) Consider the pipelined designs of the 16-bit multiplier (referring to the one on P317-330). Develop an *n*-bit pipeline design of multiplier where *n* may be an integer from 1 to 32. Perform the post-layout simulation of the design, and find out the delay, throughput, device utilization, etc., and compare them with that of designs in (1).

(3) Or consider any other multiplier realization methods, and make the comparison.

<u>A full report is required.</u>
The report should be a full comparison of the performance and cost of different 16-bit multipliers, and their application scenarios.
As you have been trained in writing report in the past labs, no specific requirement is given for this report. You yourself should think of the contents and organization of this report.

Lab Report

The **full report** should include the following for each exercise:

1. Title
2. Introduction, including the objective of the laboratory. 10%
3. Pre-lab preparation results. 10%
4. Conceptual diagram prior to the development of VHDL codes. 10%
5. Complete VHDL code and test bench code, and remarks for your codes. 10%
5. Results, including waveform images, and data, tables if any, and remarks for the results. 20%
6. Your thinking and analysis to the results. 20%
7. Conclusions. 10%
8. Language and format. 10%

The percentages of marks are roughly. Some items may not be available in some labs. The percentages may be slightly adjusted for each lab.

If a report contains only a title, a VHDL code with resultant waveform images, the report will have not than 60 marks over 100 marks. Pre-lab preparation results, presentation of results, and the analysis to the results take heavy weight for high marks.

The **Simple report** does not require in detail analysis of the results, but the conceptual diagram prior to the development of VHDL code is a must, plus code and presentation of results.