Santa Clara University

Lab #2: Design Capture and Simulation

ELEN 21L 51306 Tuesday 2:15-5pm
April 18, 2017

Group 2
Story Deweese
Yutong Li

Group2
Story Deweese
Yutong Li

ELEN 21L T 2:15-5:00 PM
4/18/17

Lab #2: Design Capture and Simulation

I. Objectives:

Build a circuit that fulfill the given condition on Altera Quartus II software, and program on the board using the circuit that we designed.

II. Introduction:

For Lab 2, we learn to use Altera Quartus II software to construct a circuit that fulfills the condition given in the problem statement. During lab, we learned how to 1) draw schematic map in the software and label each pins in a specific way, referring to the board, 2) run the program and debug, 3) do the corresponding waveform for the schematic, 4) download the program to the board and test our theory, and finally 5) plug in specific components to test the circuit. In this lab, we learn about the basic information about Quartus, which prepares us for the future lab with the software.

III. Procedure:

1. Starting the new project

Create a new project wizard, specify the type of device in which the designed circuit will be implemented.

2. Design entry using the graphic editor

a. Importing logic-gate symbols

We will need 1 AND3 gate, 2 NOT gate, 1 OR2 gate, and 1 AND2 gate.

b. Importing input and output symbols

Given the specific components, where the inputs are 1) a manual switch S, 2) a motion detector M1, and 3) a disable switch D, and the outputs are 1) a light L, and 2) a alarm A, we need to design and build the controller for a light that functions both as an ordinary light and also as a motion activated light and alarm.

c. Connecting nodes with wires

When we are using wire to connect these gates, it is important not to create a node if not necessary. We have already designed the correct schematic in the prelab, so all we need to do is to build the schematic in the Quartus. Figure 2.1 is the schematic we designed in the prelab.

Figure 2.1 Schematic we designed in the prelab

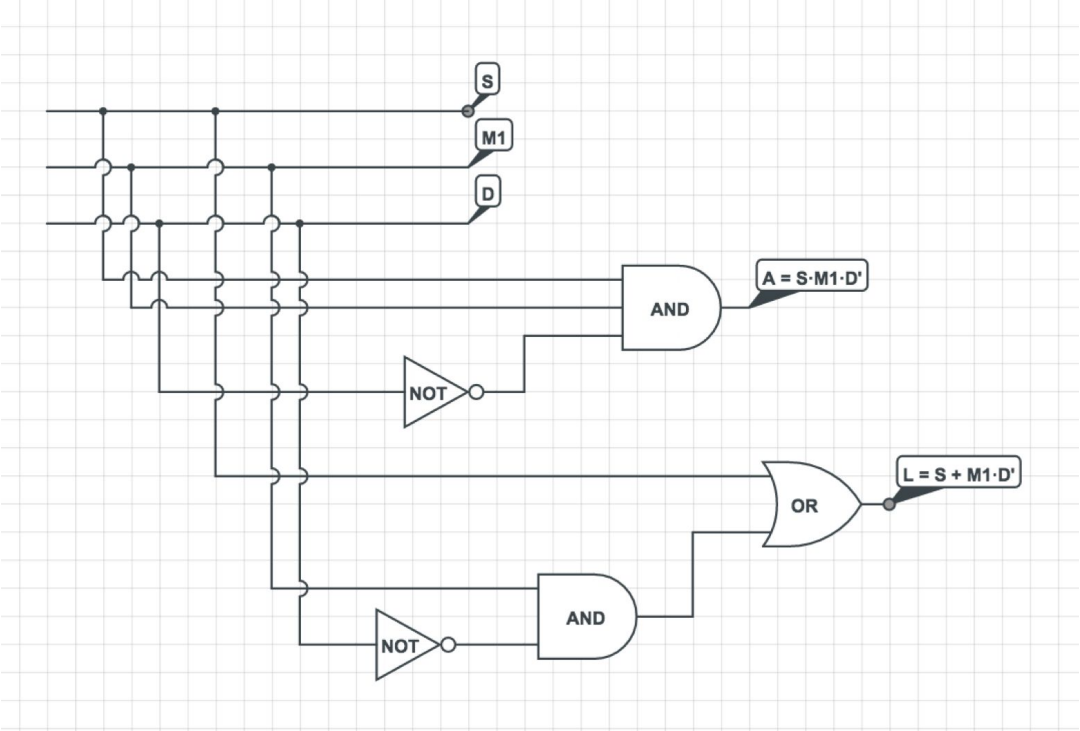
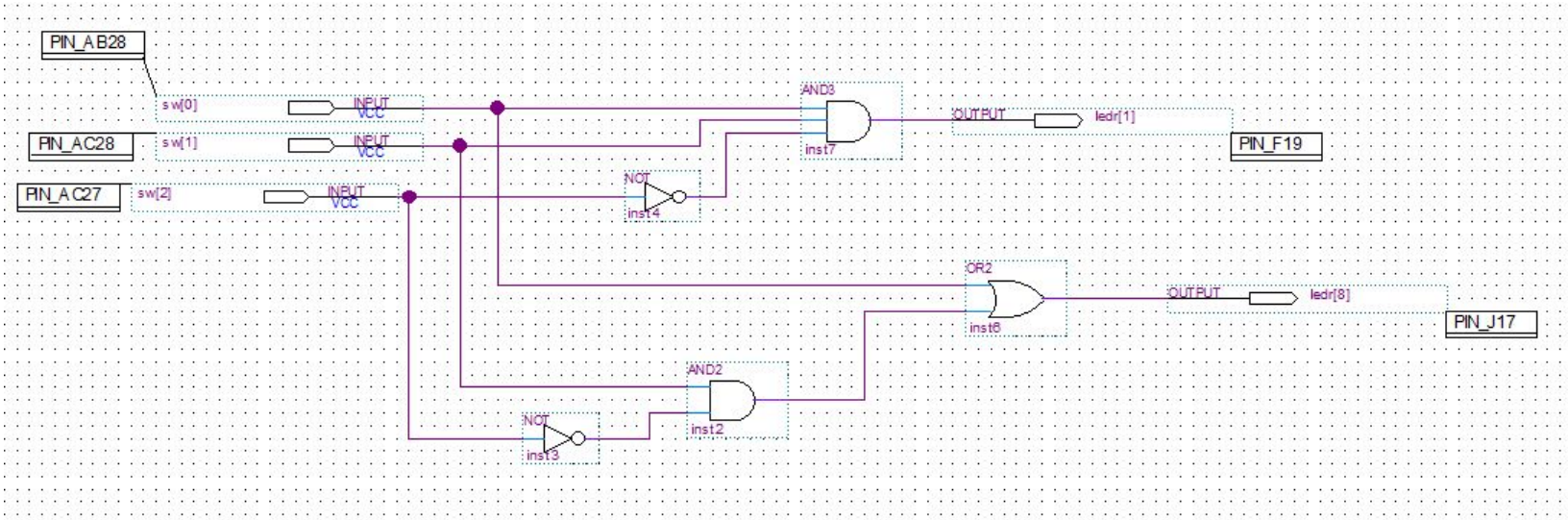


Figure 2.2 is what we build in the software, where sw[0] refers to manual switch, sw[1] refers to motion detector, sw[2] refers to disable switch, ledr[1] refers to the alarm, and ledr[8] refers to the light. Always remember that number should go into the bracket.

Figure 2.2 Schematic we designed in class in Quartus



3. *Compiling the designed circuit*

The default device in our computer is different from the device that we have, so we need to assign the correct device to the computer every time before we compile. When we didn't do that, there will be error.

4. *Pin assignment*

After we insert the assignment “DE2_115.qsf” as set it as global, we will have the reference for each pin to the its value on the board. By adding this assignment, we don’t need to manually add the value for the pins one by one.

5. Simulating the designed circuit

Before implementing the designed circuit on the board, we need to simulate it to ascertain its correctness. In order to do that, we use the waveform editor. In the waveform editor, we need to insert all of our inputs and outputs and specify their type, whether they are input or output. We find out that input is represented by a straight line, and output is represented as a bunch of Xs. After inserting the nodes, we need to set the inputs to different values according to the truth table that we make in the prelab, which is shown in Figure 2.3.

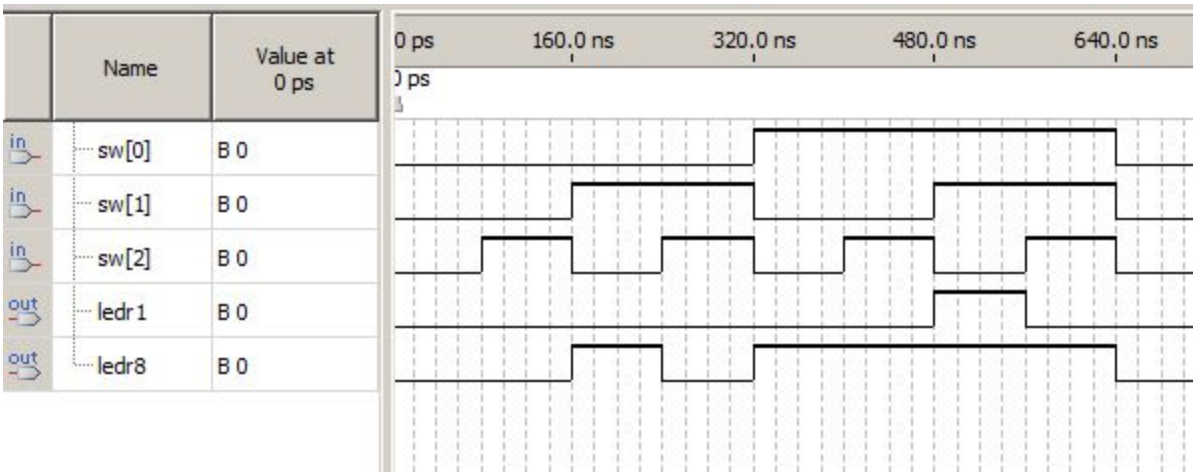
Figure 2.3 Truth table that we came out in prelab

S	M1	D	L	A
0	0	0	0	0
0	0	1	0	0
0	1	0	1	0
0	1	1	0	0
1	0	0	1	0
1	0	1	1	0
1	1	0	1	1
1	1	1	1	0

6. Performing the functional simulation

Choose FUNCTIONAL as the simulation type. Before running the functional simulation it is necessary to create the required netlist. After running, we see something like what is shown in Figure 2.4. The result of the functional simulation matches the truth table in Figure 2.3.

Figure 2.4 Result of the functional simulation



It is shown that ledr1, which is alarm in the case, only goes off when sw[0] the manual switch is 1, sw[1] the motion detector is 1, and sw[2] the disable switch is 0, and the light is on when sw[0] is 0, sw[1] is 1, and sw[2] is 0, and for all the conditions in which sw[0] is 1.

7. Programming and configuring the FPGA Device

We need to compile the design circuit every time before running the programmer. The hardware is not automatically selected in our computer, so we choose USB-Blaster [USB-0]. Then we add the file with sof type, and then we start. When it is in progress, we can see that the lights on the board are lit up brightly, and when it is successfully done, all the other lights except ledr1 and ledr8 becomes really dim, and ledr1 and ledr8 is off.

8. Testing the designed circuit

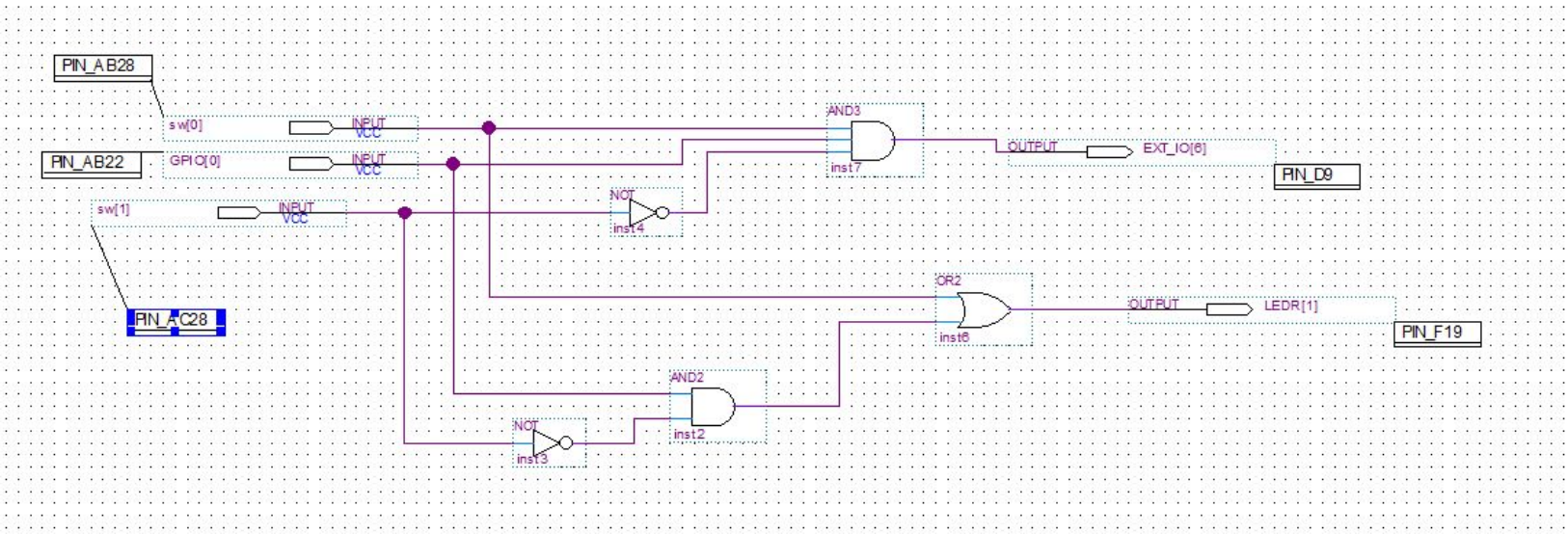
Having downloaded the configuration data into the FPGA device, we can now test the implemented circuit. First, we flip the RUN switch to RUN position. We try all 8 valuations of the input variables S, M1 and D, by setting the corresponding states of the switches sw[0], sw[1], and sw[2]. We find that the circuit implements the truth table above correctly, therefore, we know that our circuit is correctly built.

9. Plugging in the actual component into the circuit

After testing the designed circuit in the board with switches, we plug in the motion detector and alarm. We put motion detector in the GPIO connector, use 5V as input and ground it and let the output be in the spot GPIO[0]. We put alarm in the expansion header, and let both of the wires of the alarm be in EXT_IO[6]. After changing one input and one output, the schematic looks like what is shown in Figure 2.5.

(continued on next page)

Figure 2.5 Schematic after adding alarm and motion detector



We try all 8 valuations of the input variables, and find that the circuit implements the truth table correctly.

IV. Mistakes that we make:

1. Each time we wanted to download our work to the board, we had to re-import the DE2_115.qsf file or else our program would fail.
2. The device family would automatically default to a different board than we were using. Even though we setup the software to work with the board, each time we tried to upload our work to the board, it would revert back to its default board setting.
3. If we did not have the correct board listed, since it often automatically reverted to the default, some of the pin names would not show up even after we used the file from camino that converted from our inputs such as "sw[1]" to PIN_AC28 as seen in the schematic above.
4. Occasionally, we forgot to run the compilation again before downloading to the board.
5. We attempted to jump from sw[0] to sw[2] without using sw[1], which also caused errors.
6. Lastly, we mistook the inputs and outputs when we added the alarm and motion detector into the board.

V. Final question:

In lab 1, what would you have to change to have any of three different motion detectors turn on the light and turn on the buzzer if the light is already on because of s? Specifically consider components, wiring, and testing. Compare that to the changes you would need to make for the Altera FPGA implementation.

In lab 1, we would need to change the AND gate to an OR gate to allow the light to be on. We would change the inputs for the OR gate to be the three motion detectors. Also, if the light was already on due to the switch, the buzzer would also turn on.

VI. Conclusion:

In conclusion, we learned that it is important to always check that the board is correct before compiling our code and downloading it to the board. Also, we became more familiar with the Altera Quartus II software, testing to see if our design will work using waveform simulation methods, and how to solve both problems with our coded program and with the hardware of the system. This lab helped us familiarize ourselves with the equipment we will be using in lab in the future.

VII. Reference lists:

- Details on pin assignment tables:

ftp://ftp.altera.com/up/pub/Altera_Material/12.1/Boards/DE2-115/DE2_115_User_Manual.pdf

- Altera Quartus II tutorial:

ftp://ftp.altera.com/up/pub/Altera_Material/12.1/Tutorials/Schematic/Quartus_II_Introduction.pdf

- Details on waveform simulation methods:

ftp://ftp.altera.com/up/pub/Altera_Material/12.1/Boards/DE2-115/DE2_115_User_Manual.pdf

- lab2_spring2017.pdf

https://camino.instructure.com/courses/29358/files?preview=1546155