ECE272 LAB#1

Section#1

Basic Combinational Logic and DE10-Lite

Hyunjae Kim

06/30/2021

Matthew Shuman

1. Introduction

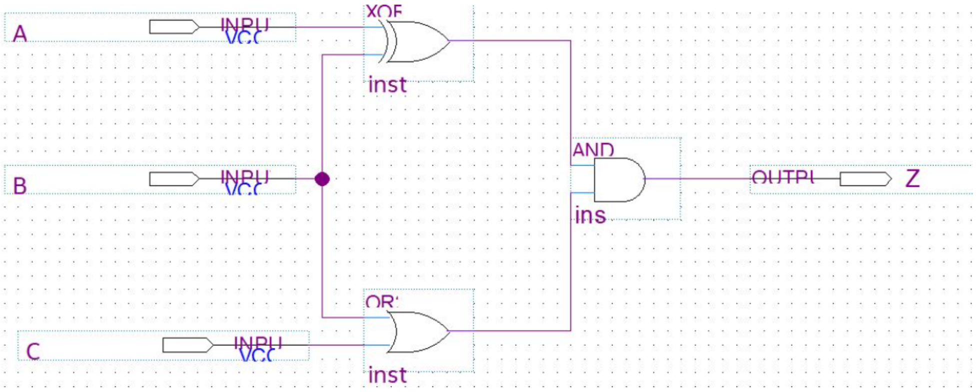
The general purpose of this lab is to design the circuit with logic gates, program the designed schematic to the hardware, and verify the programed hardware work same as expected result of the design. The lab activity intends student learn how to design digital logic and insert the designed digital logic to Max10 FPGA on DE10-Lite board by using intel Quartus software. Then the student has to verify the actual output from the programmed hardware by comparing actual to expected output which the student expected before the student activates the hardware.

To accomplish the purpose of the labs, student have to learn about background information before starting the lab activity. First of all, the student has to learn how to set up the Quartus software, and installing USB-blaster to connect the DE10-Lite board to the software. Secondly, the student has to understand about Programmable Logic Devices (PLDs) to know what device the student is using. Finally, the student had known about logic gates and Boolean equation (or algebra) to verify the design and the actual result of the lab activity.

In lab 1, the student was required to design the suggested schematic design from the lab document and program the schematic to the FPGAB by using Quartus software. In the lab document, input A and input B were controlled by XOR gate, and input B and input C were controlled by OR gate. Then they were passing the AND gate before the output Z. The student had to expect the results by using Boolean algebra and logic gates. Finally, the student had to verify the actual result that the design and the programming were correct.

2. Design

1. The suggested schematic Design to put into Quartus Software. (Block Design)



(From Figure1.5: Section 1 Combinational Logic)

2. Explanation of the planned design.
   - Mathematical process of the design:

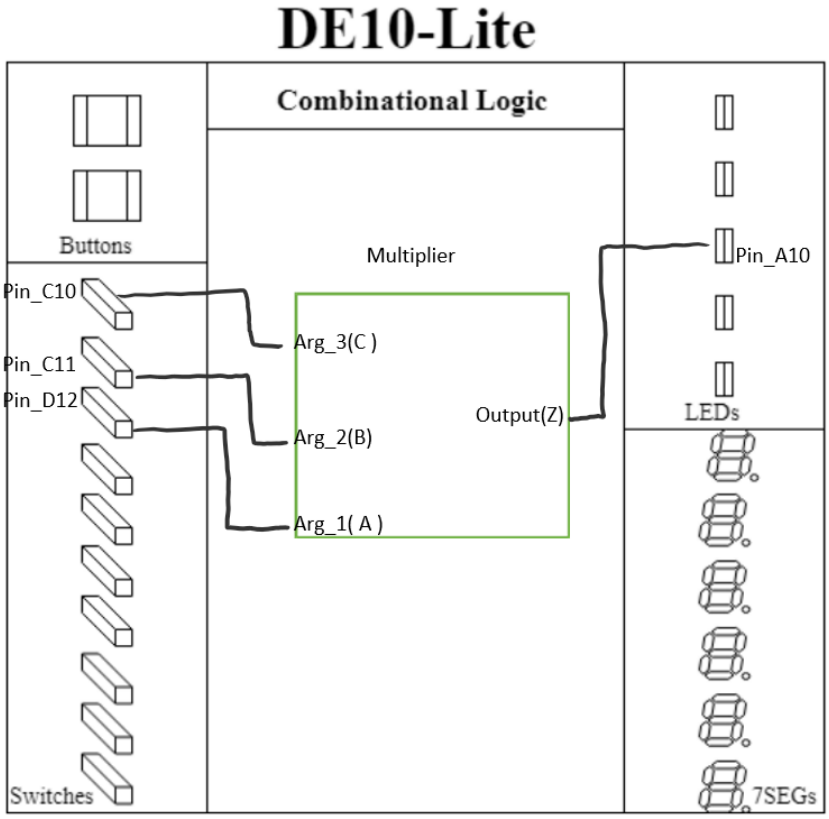
$$Z = (A \oplus B)(B + C)$$

   - The input A and input B pass the XOR gate (=$A \oplus B$), and the input B and C pass the OR gate (=B+C). The results of both XOR gate and OR gate pass AND gate (= $(A \oplus B)(B + C)$. Therefore, the result of AND gate is the output Z.

3. Expecting the output Z by using truth table

A	B	C	Z
0	0	0	$(0\oplus 0)(0+0)=0$
0	0	1	$(0\oplus 0)(0+1)=0$
0	1	0	$(0\oplus 1)(1+0)=1$
1	0	0	$(1\oplus 0)(0+0)=0$
1	1	0	$(1\oplus 1)(1+0)=0$
1	0	1	$(1\oplus 0)(0+1)=1$
0	1	1	$(0\oplus 1)(1+1)=1$
1	1	1	$(1\oplus 1)(1+1)=0$

4. Programming the hardware (Designing Block Diagram)



(Note: Each of pin connected to the input and the output is 1 bit.)

	FPGA Pin
Input A	D12 (Switch [2])
Input B	C11 (Switch [1])
Input C	C10 (Switch [0])
Output Z	A10 (Led [2])

(From Figure 1.9: FPGA Pin Table)

Pin D12, C11, and C10 control the logic of inputs by switching logic 0 and 1. If the logic of output is 1, then the LED which is connected with the output Z by pin A10 will be turned on, and if the logic of output is zero, then the LED will be turned off.

3. Results

1. Truth table with expected output Z and actual output Z

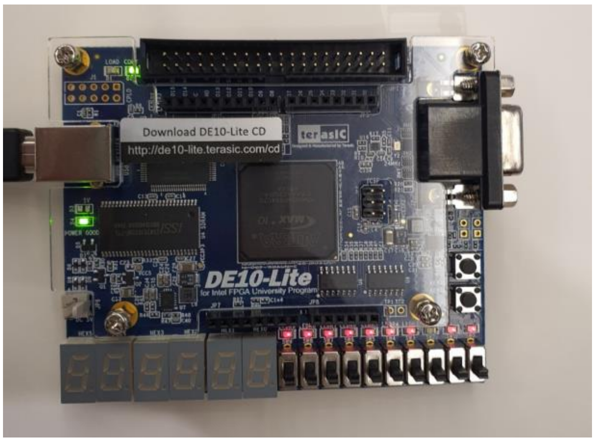
A	B	C	Z(Expected)	Z(Actual)
0	0	0	0	0
0	0	1	0	0
0	1	0	1	1
1	0	0	0	0
0	1	1	1	1
1	0	1	1	1
1	1	0	0	0
1	1	1	0	0

(From Figure 1.10: Truth table for Section 1 Testing)

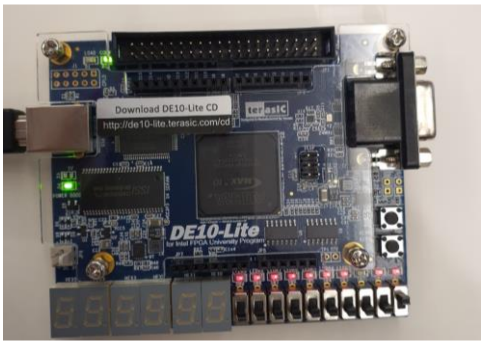
2. Images of Actual Results

   (Note: number of the order is the order of the row (i.e., row0, row1, row 2, …, row 8), and output Z is connected to LED [2])

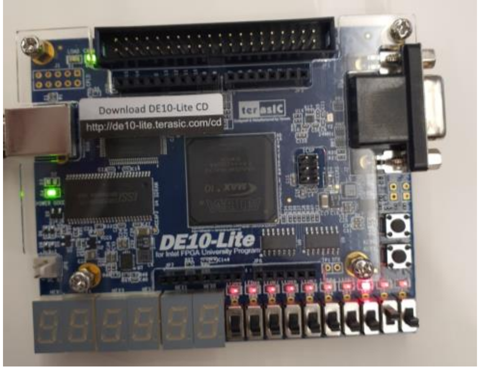
1) Z = 0


Download DE10-Lite CD
http://de10-lite.terasic.com/cd
terasIC
DE10-Lite
for Intel FPGA University Program

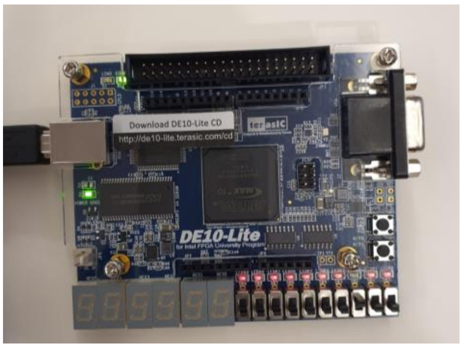
2) Z = 0


Download DE10-Lite CD
http://de10-lite.terasic.com/cd
terasIC
DE10-Lite
for Intel FPGA University Program

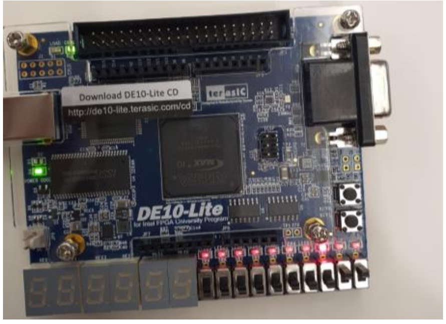
3) Z = 1


Download DE10-Lite CD
http://de10-lite.terasic.com/cd
terasIC
DE10-Lite
for Intel FPGA University Program

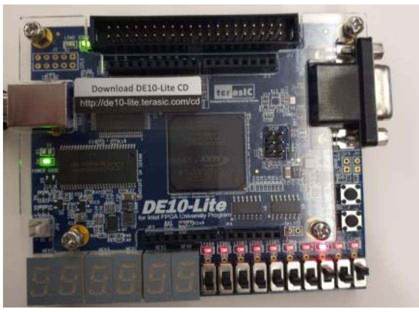
4) Z = 0


Download DE10-Lite CD
http://de10-lite.terasic.com/cd
terasIC
DE10-Lite
for Intel FPGA University Program

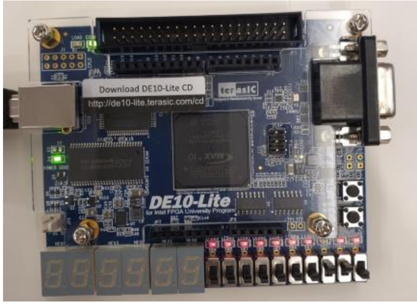
5) Z = 1



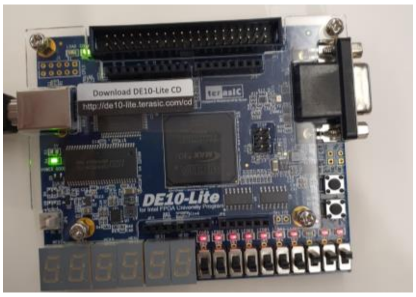
6) Z = 1



7) Z = 0



8)Z = 0



3. Explanation of the result

   The result of implementation matched the expected result which was conjectured before activating DE10-Lite board.

4. Experiment Notes

In this lab, setting the lab was the challenging part for me. The reason is that I was unfamiliar in using a software like Quartus, and installing USB-blaster was failed almost 10 times. However, except these parts, most of the parts went well. To be specific, designing schematic, compiling the design, and setting pins in the FPGA were went well. Furthermore, after I solve the problem of installing USB-blaster by downloading driver and certificate of USB-blaster from external website, programming the hardware went well.

5. Study Question

1. Describe any problems encountered in this lab and your solutions to those problems.

   I encountered problems which were setting up and activating Quartus and connecting USB-blaster. When I set up and activate the Quartus project, the software stopped activating. Thus, I visited professor’s office hour and got solution that Quartus has to be activated in administrator mode. Furthermore, I failed to connect USB-blaster to Quartus. I tried many methods such as restarting my laptop, installing USB-blaster driver from Quartus file, and using “Mprog” software. These methods were failed. Finally, I got an USB-blaster driver and certificate file from external website, and successfully install the USB-blaster driver and connected USB-blaster to Quartus.

2. Give an example of where discrete logic ICs (such as the 7400 series logic chips mentioned in section 1.1) are used in industry and why.

   Discrete ICs such as 7400 series can be used for brake system of electric cars. Electric cars have motors to move the wheel, and if the driver pushes the brake, then the brake will be act as a switch and the output of discrete IC will be 0 or 1. Then, when it is 0, the motor of the wheel will stop, and when it is 1, the motor of the wheel will activate. Thus, discrete IC can be used for brake system of electric cars.

3. Give an example of when you should use an FPGA instead of a PLA and explain why.

FPGA can be used for AI instead of a PLA. Unlike conventional software, AI can learn itself and develop its ability. In the process of learning and developing, the logic of hardware part of AI will be changed fast so that FPGA is appropriate because FPGA is reprogrammable.

4. Give an example of when you should use an PLA instead of a FPGA and explain why.

   PLA can be used for brake system of electric cars with discrete logic IC instead of FPGA. The reason is that PLA has smaller amount than FPGA, and superior to work with multiple ICs such as 7400 series to achieve the same result. [1]

5. Summarize the main differences between FPGAs and CPLDs, other than the difference described in the note in section 1.1.

   FPGA has more logic blocks than CPLD has. FPGA has strength in complex application and CPLD has strength in simple application. In the aspect of the base of digital logic chip, FPGA is RAM-based and CPLD is EEPROM-based. Moreover, in the aspect of the delay, the delays in CPLD are much more predictable than delays in FPGA. In economic aspect, FPGA is much more expensive than CPLD. [2]

6. Citation

[1] PLA vs CPLD vs FPGA

https://blog.digilentinc.com/pal-vs-cpld-vs-fpga/

[2] Difference between FPGA and CPLD

http://www.differencebetween.net/technology/difference-between-fpga-and-cpld/