ECE 272 Lab #3

Combinational Logic (Seven Segment Driver)

Hyunjae Kim

07/21/2021

Matthew Shuman

1. Introduction

In this lab, the general purpose is observing each of the seven segments display, and find the logic of each segment. To be specific, this lab requires the student to display the numbers from 0 to 15 which are inserted as 4-bit binary in hexadecimal by using DE10-Lite hardware.

To display the seven segment display from Sa to Sg, the first step is how each of the numbers 0 to 15 will be displayed in the combination of each seven segment display. After determining each display of the numbers, the next step is writing a truth table about the input numbers and the seven segment display by marking 0 when the seven segment display is ON and marking 1 when the seven segment display is OFF. The reason of marking the result opposite to the textbook is that the seven segment display in DE10-Lite is active low. This means that input value 0 turns on the segment, and input value 1 turns off the segment.

The output result of each segment display will be found by using K-map. Each of the K-maps will show the output logic of each segment display. After finding the output logic of each segment, each of the output logic will be implemented by using logic gates in Quartus software and then combined. The combined result will be programmed to DE10-Lite hardware, and then verified by the ModelSim software simulation.

2. Design

Before finding the output logic of each segment display, the first step is designing the seven segment display to demonstrate the input numbers in hexadecimals. Based on the expected design, the truth table of the input by output will be created.

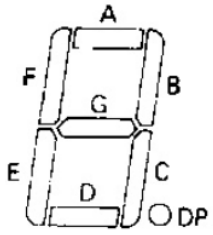
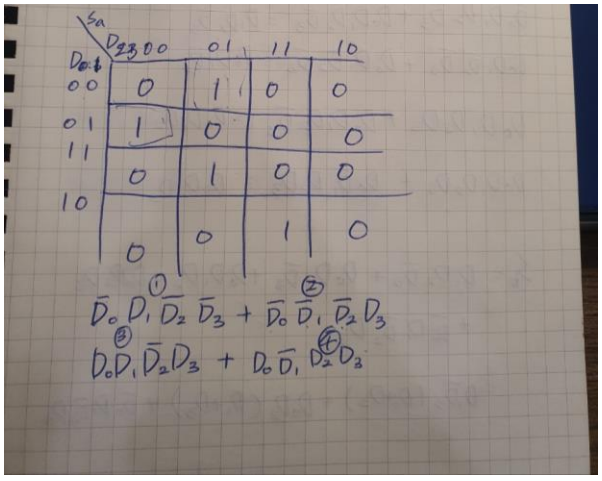


Figure 3.2: Displaying hexadecimal numbers on a seven-segment display

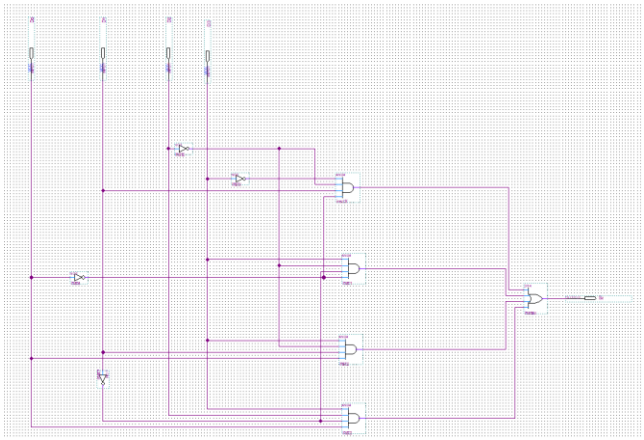
Input (Hexadecimal)	Input (4-bit Binary)	Seg_A	Seg_B	Seg_C	Seg_D	Seg_E	Seg_F	Seg_G
0	0000	0	0	0	0	0	0	1
1	0001	1	0	0	1	1	1	1
2	0010	0	0	1	0	0	1	0
3	0011	0	0	0	0	1	1	0
4	0100	1	0	0	1	1	0	0
5	0101	0	1	0	0	1	0	0
6	0110	0	1	0	0	0	0	0
7	0111	0	0	0	1	1	1	1
8	1000	0	0	0	0	0	0	0
9	1001	0	0	0	1	1	0	0
a	1010	0	0	0	1	0	0	0
b	1011	1	1	0	0	0	0	0
c	1100	0	1	1	0	0	0	1
d	1101	1	0	0	0	0	1	0
e	1110	0	1	1	0	0	0	0
f	1111	0	1	1	1	0	0	0

1) Segment a

To design Segment a, the output logic is found by using K-map.



The output logic of Sa from K-map is D0'D1D2'D3'+D0'D1'D2'D3+D0D1D2'D3+D0D1'D2D3.



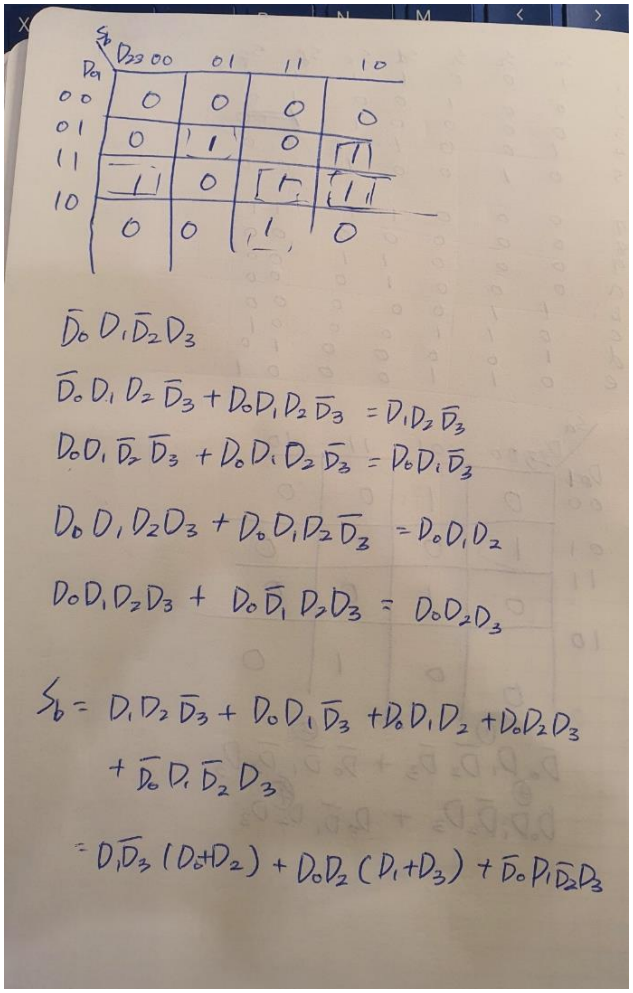
(The block design of Sa)

The figure above is logic design of the output of Segment a which is from the K-map.

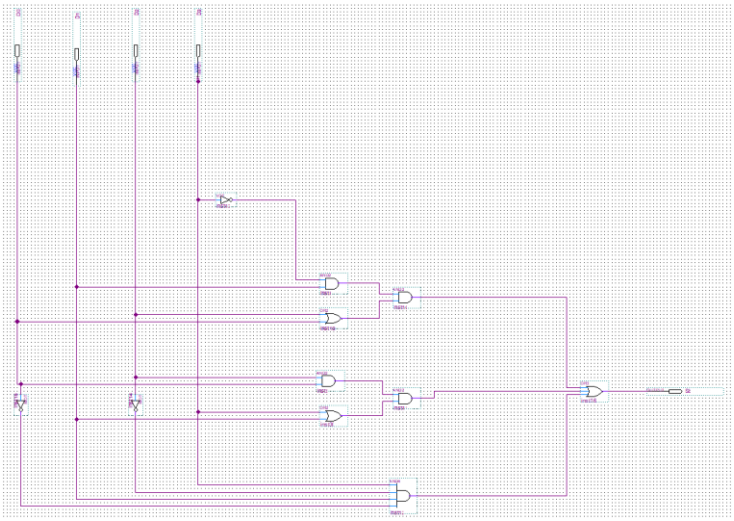
Interface Name	Interface Purpose	Input or Output	Number of Pins	Board Label	FPGA Pin
D0	Gate input	Input	1	Switch 3	C12
D1	Gate input	Input	1	Switch 2	D12
D2	Gate input	Input	1	Switch 1	C11
D3	Gate input	Input	1	Switch 0	C10
Sa	Gate output	Output	1	Hex 0	C14

2) Segment b

To design Segment b, the output logic is found by using K-map.



The output logic of Sb from K-map is D1D3'(D0+D2)+D0D2(D1+D3)+D0'D1D2'D3.



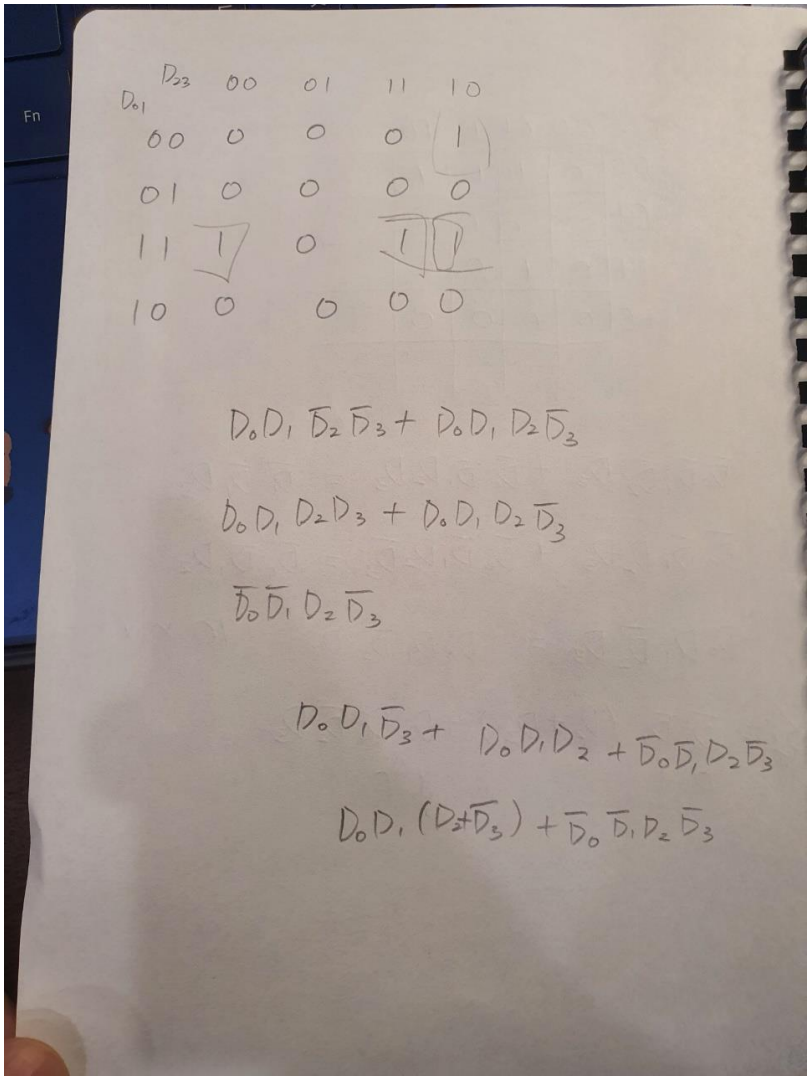
(The block design of Sb)

The figure above is logic design of the output of Segment b which is from the K-map.

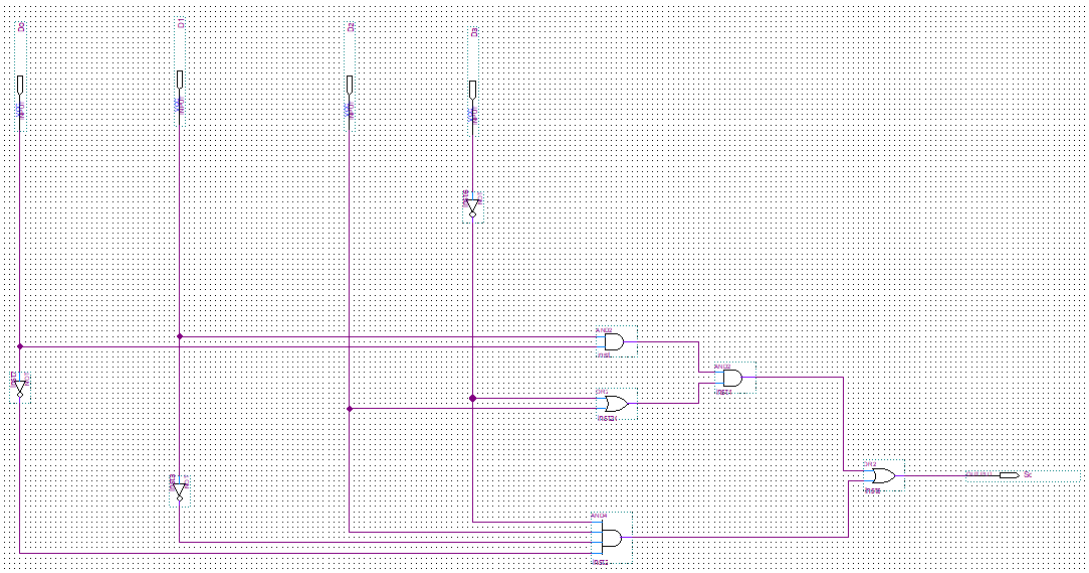
Interface Name	Interface Purpose	Input or Output	Number of Pins	Board Label	FPGA Pin
D0	Gate input	Input	1	Switch 3	C12
D1	Gate input	Input	1	Switch 2	D12
D2	Gate input	Input	1	Switch 1	C11
D3	Gate input	Input	1	Switch 0	C10
Sb	Gate output	Output	1	Hex 1	E15

3) Segment c

To design Segment c, the output logic is found by using K-map.



The output logic of Sc from K-map is D0D1(D2+D3')+D0'D1'D2D3'.



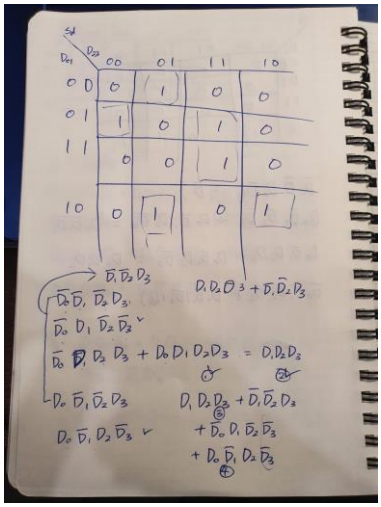
(The block design of Sc)

The figure above is logic design of the output of Segment c which is from the K-map.

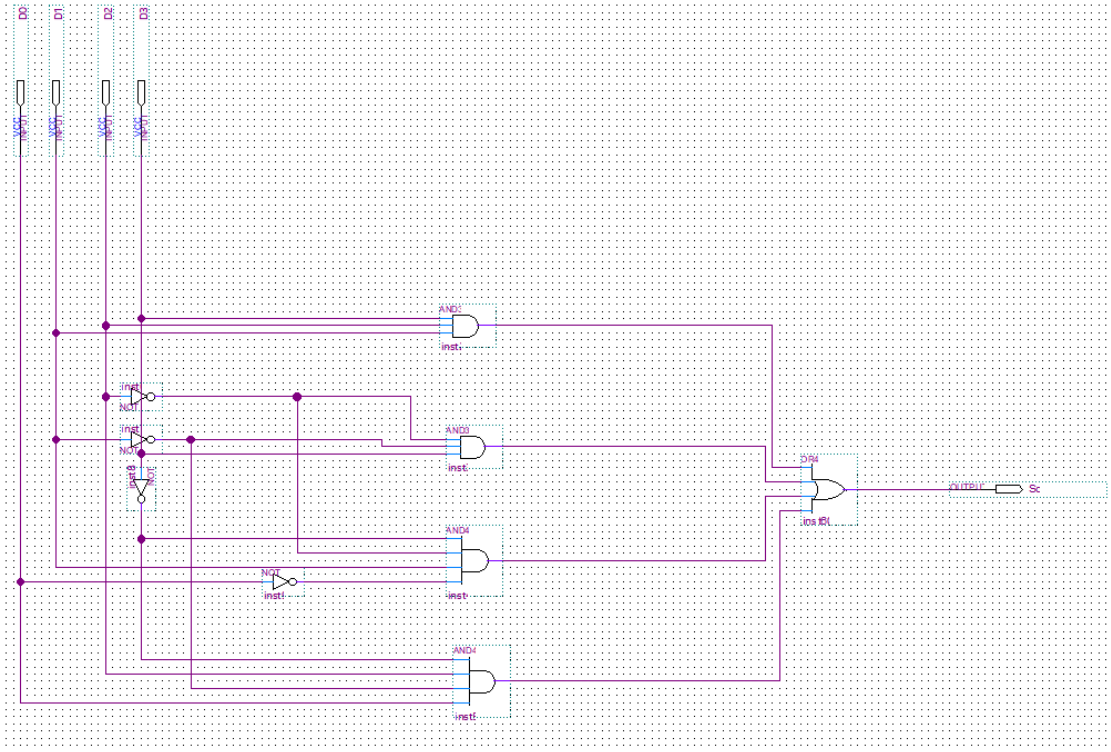
Interface Name	Interface Purpose	Input or Output	Number of Pins	Board Label	FPGA Pin
D0	Gate input	Input	1	Switch 3	C12
D1	Gate input	Input	1	Switch 2	D12
D2	Gate input	Input	1	Switch 1	C11
D3	Gate input	Input	1	Switch 0	C10
Sc	Gate output	Output	1	Hex 2	C15

4) Segment d

To design Segment d, the output logic is found by using K-map.



The output logic of Sd from K-map is D1D2D3+D1'D2'D3+D0'D1D2'D3'+D0D1'D2D3'.



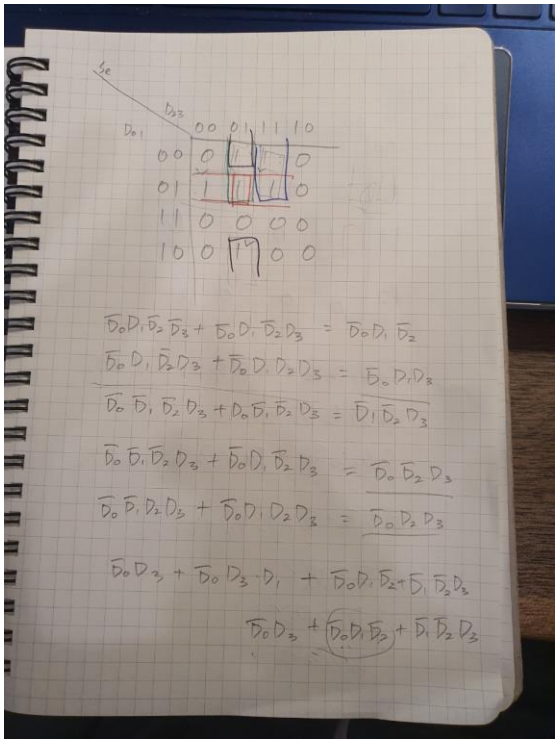
(The block design of Se)

The figure above is logic design of the output of Segment e which is from the K-map.

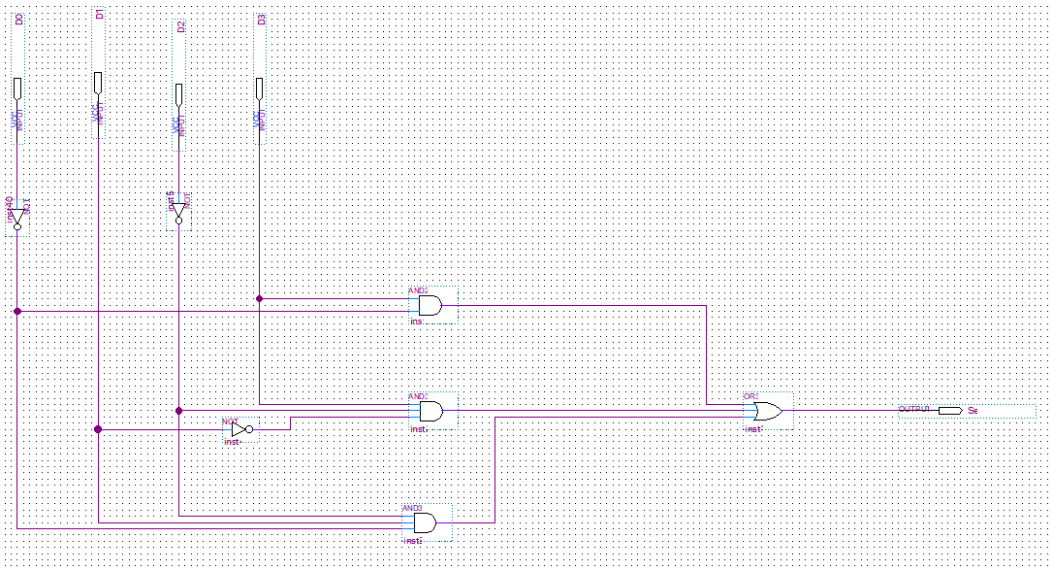
Interface Name	Interface Purpose	Input or Output	Number of Pins	Board Label	FPGA Pin
D0	Gate input	Input	1	Switch 3	C12
D1	Gate input	Input	1	Switch 2	D12
D2	Gate input	Input	1	Switch 1	C11
D3	Gate input	Input	1	Switch 0	C10
Sd	Gate output	Output	1	Hex 3	C16

5) Segment e

To design Segment e, the output logic is found by using K-map.



The output logic of Se from K-map is D0'D3+D0'D1D2'+D1'D2'D3.



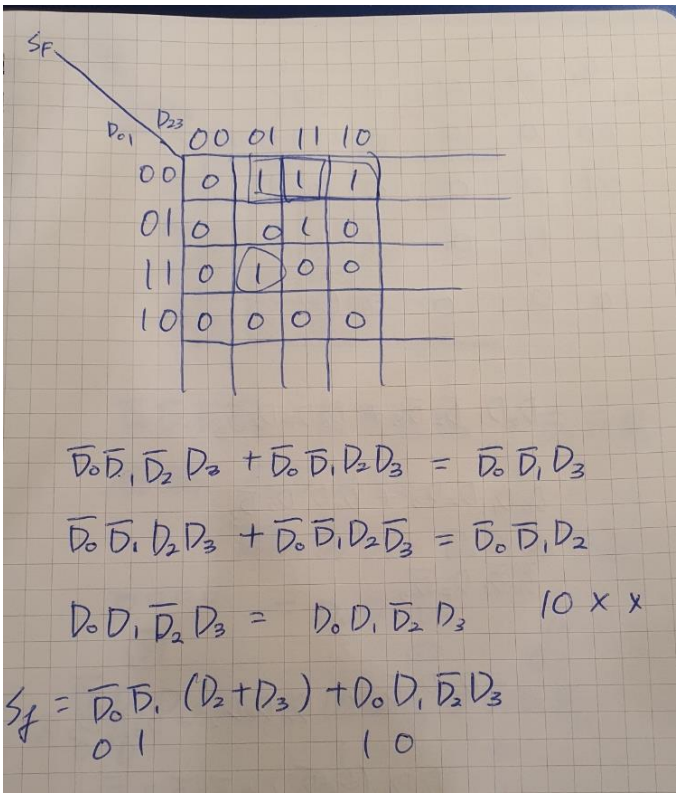
(The block design of Se)

The figure above is logic design of the output of Segment e which is from the K-map.

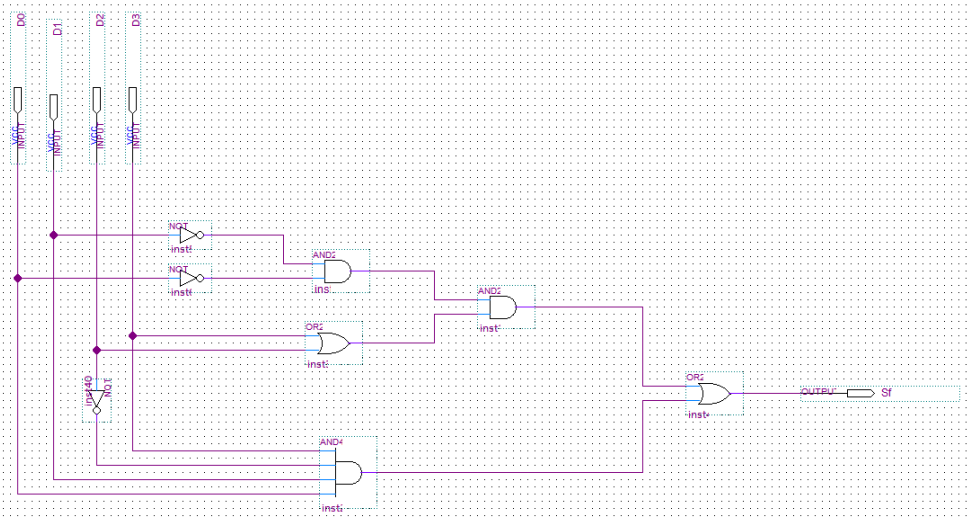
Interface Name	Interface Purpose	Input or Output	Number of Pins	Board Label	FPGA Pin
D0	Gate input	Input	1	Switch 3	C12
D1	Gate input	Input	1	Switch 2	D12
D2	Gate input	Input	1	Switch 1	C11
D3	Gate input	Input	1	Switch 0	C10
Se	Gate output	Output	1	Hex 4	E16

6) Segment f

To design Segment f, the output logic is found by using K-map.



The output logic of Se from K-map is D0'D1'(D2+D3)+D0D1D2'D3.



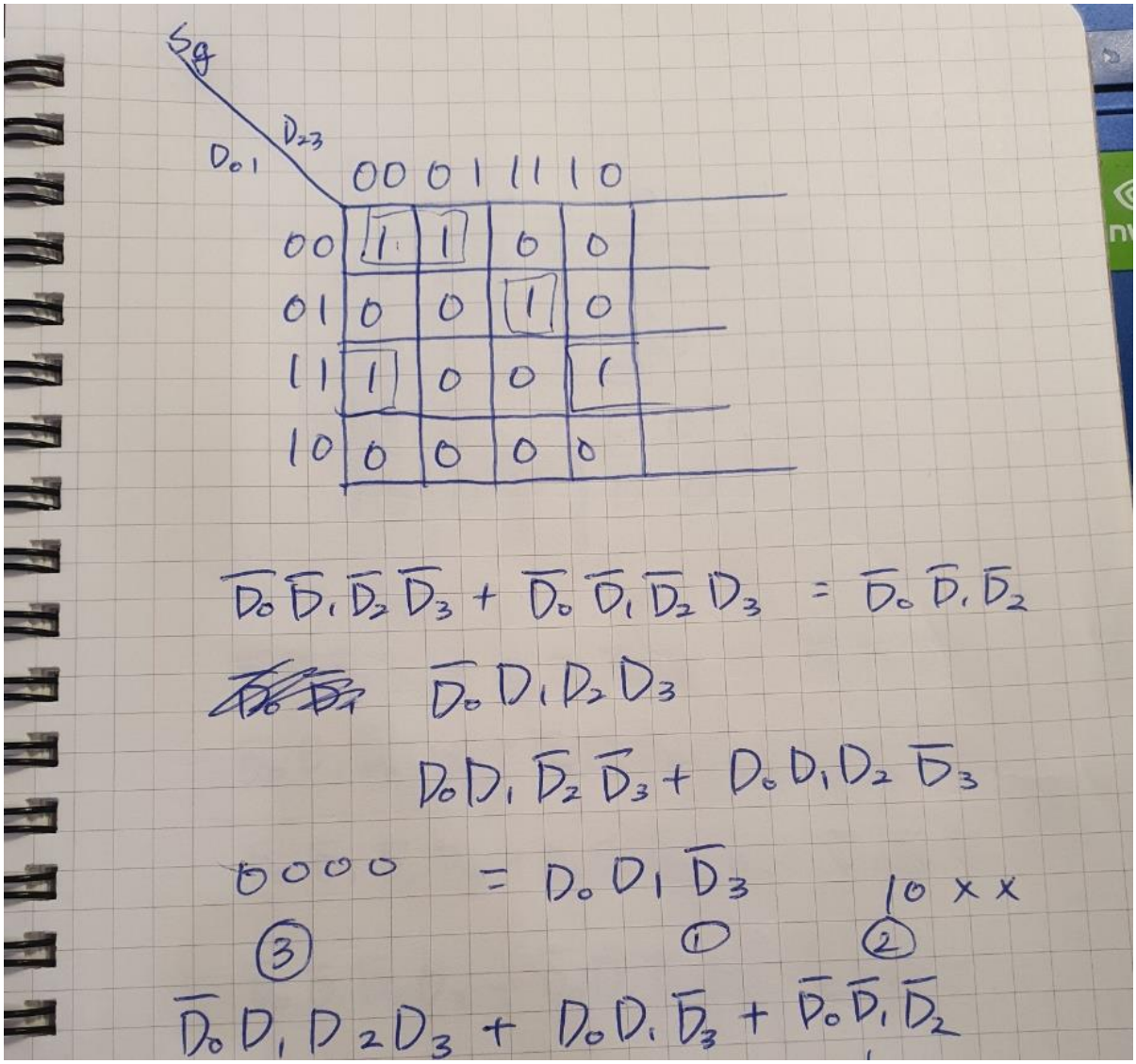
(The block design of Sf)

The figure above is logic design of the output of Segment f which is from the K-map.

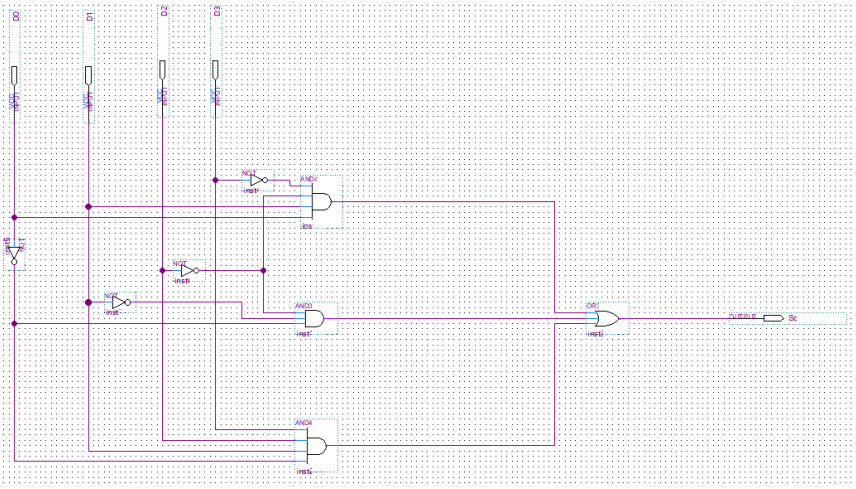
Interface Name	Interface Purpose	Input or Output	Number of Pins	Board Label	FPGA Pin
D0	Gate input	Input	1	Switch 3	C12
D1	Gate input	Input	1	Switch 2	D12
D2	Gate input	Input	1	Switch 1	C11
D3	Gate input	Input	1	Switch 0	C10
Sf	Gate output	Output	1	Hex 5	D17

7) Segment g

To design Segment g, the output logic is found by using K-map.



The output logic of Sg from K-map is D0'D1'D2'+D0'D1D2D3+D0D1D2'D3'.



(The block design of Sg)

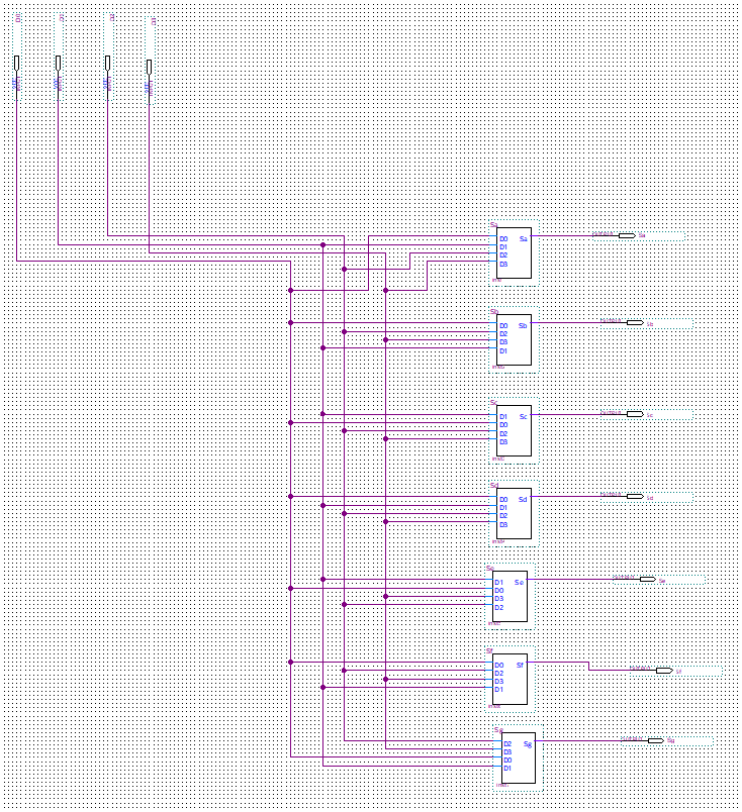
The figure above is logic design of the output of Segment g which is from the K-map.

Interface Name	Interface Purpose	Input or Output	Number of Pins	Board Label	FPGA Pin
D0	Gate input	Input	1	Switch 3	C12
D1	Gate input	Input	1	Switch 2	D12
D2	Gate input	Input	1	Switch 1	C11
D3	Gate input	Input	1	Switch 0	C10
Sg	Gate output	Output	1	Hex 6	C17

8) Seven Segment Display

This design is a design which merged the entire segment display blocks.

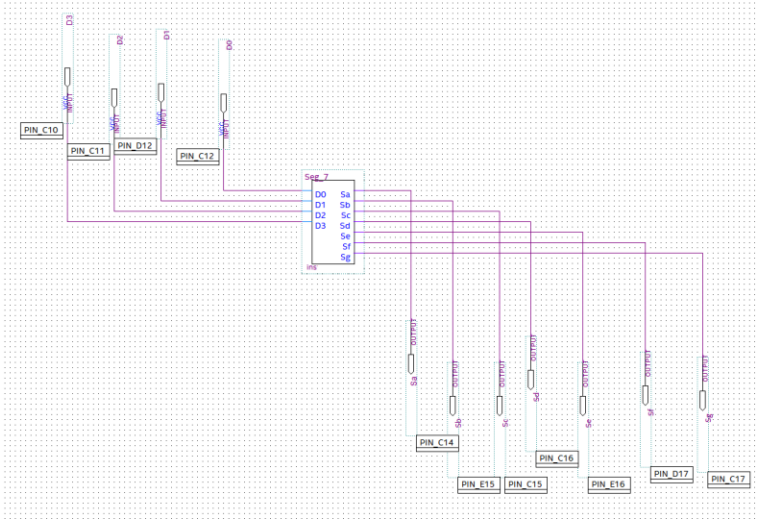
Interface Name	Interface Purpose	Input or Output	Number of Pins	Board Label	FPGA Pin
D0	Gate input	Input	1	Switch 3	C12
D1	Gate input	Input	1	Switch 2	D12
D2	Gate input	Input	1	Switch 1	C11
D3	Gate input	Input	1	Switch 0	C10
Sa	Gate output	Output	1	Hex 0	C14
Sb	Gate output	Output	1	Hex 1	E15
Sc	Gate output	Output	1	Hex 2	C15
Sd	Gate output	Output	1	Hex 3	C16
Se	Gate output	Output	1	Hex 4	E16
Sf	Gate output	Output	1	Hex 5	D17
Sg	Gate output	Output	1	Hex 6	C17



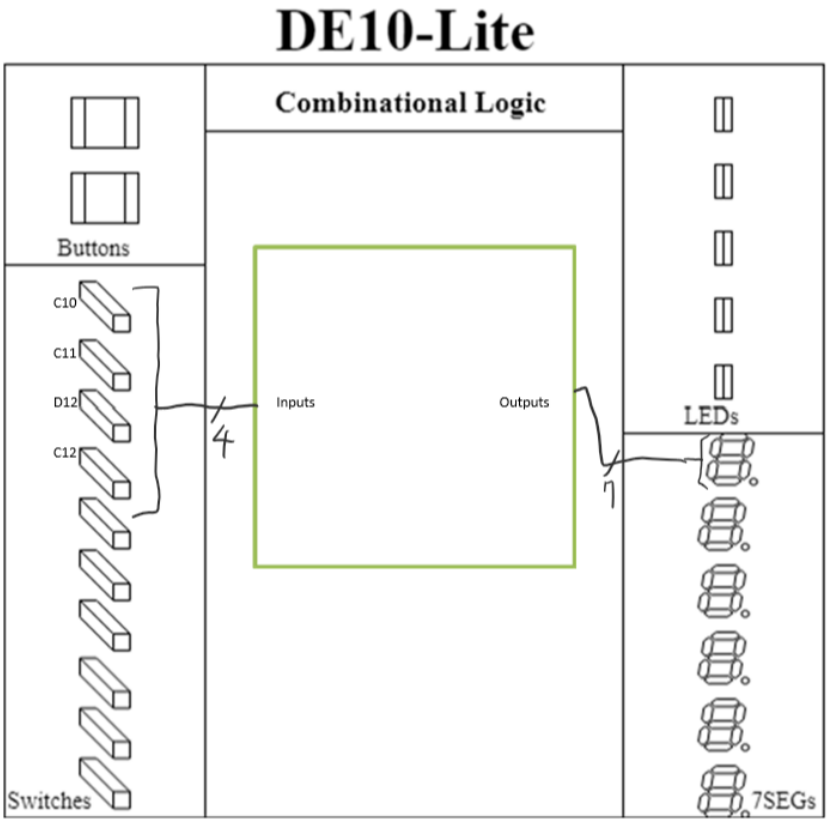
The figure above is the block design which merged from Segment a to Segment g.

9) The Top-Level Design

As the result of merging the entire segment display, the top-level design is simplicated.



Interface Name	Interface Purpose	Input or Output	Number of Pins	Board Label	FPGA Pin
D0	Gate input	Input	1	Switch 3	C12
D1	Gate input	Input	1	Switch 2	D12
D2	Gate input	Input	1	Switch 1	C11
D3	Gate input	Input	1	Switch 0	C10
Sa	Gate output	Output	1	Hex 0	C14
Sb	Gate output	Output	1	Hex 1	E15
Sc	Gate output	Output	1	Hex 2	C15
Sd	Gate output	Output	1	Hex 3	C16
Se	Gate output	Output	1	Hex 4	E16
Sf	Gate output	Output	1	Hex 5	D17
Sg	Gate output	Output	1	Hex 6	C17



(The block design of top-level design programmed in the hardware, DE10-Lite)

The block design is a seven segment display. The block gets 4-bit inputs and then display the input value in hexadecimal to the seven segment display. Each of the segment displays have combinational logic to 4-bit inputs. To be specific, the combinational logic controls each segment display, Sa ~ Sg, when to turn on or off depending on the input values from 0000 to 1111 in binary. Thus, the combination of each segment display's output logic shows the hexadecimal numbers depending on the input values.

3. Results

1) Actual Result

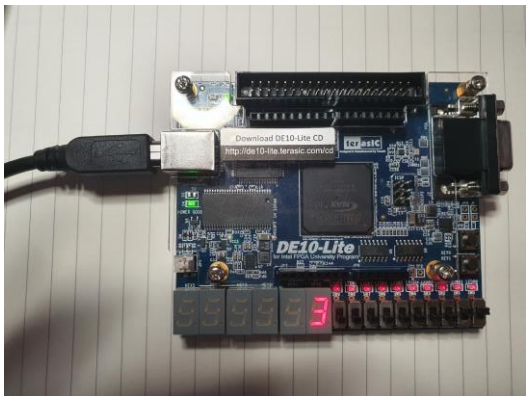
Hexadecimal	4-bit binary	Sa	Sb	Sc	Sd	Se	Sf	Sg
0	0000	0	0	0	0	0	0	1
1	0001	1	0	0	1	1	1	1
2	0010	0	0	1	0	0	1	0
3	0011	0	0	0	0	1	1	0
4	0100	1	0	0	1	1	0	0
5	0101	0	1	0	0	1	0	0
6	0110	0	1	0	0	0	0	0
7	0111	0	0	0	1	1	1	1
8	1000	0	0	0	0	0	0	0
9	1001	0	0	0	1	1	0	0
A	1010	0	0	0	1	0	0	0
B	1011	1	1	0	0	0	0	0
C	1100	0	1	1	0	0	0	1
D	1101	1	0	0	0	0	1	0
E	1110	0	1	1	0	0	0	0
F	1111	0	1	1	1	0	0	0

- 0000

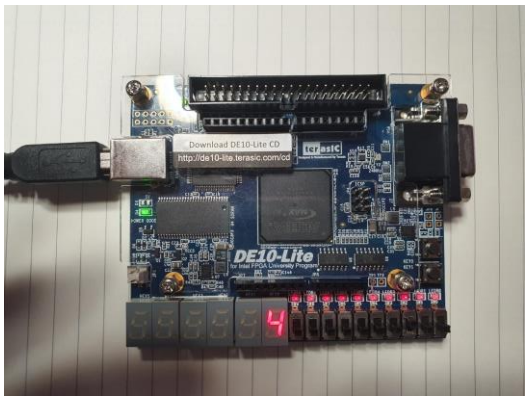
- 0001

- 0010

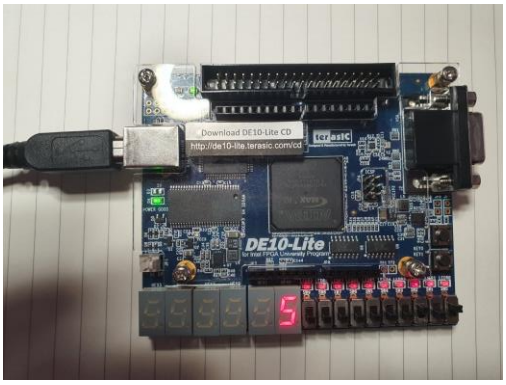
- 0011



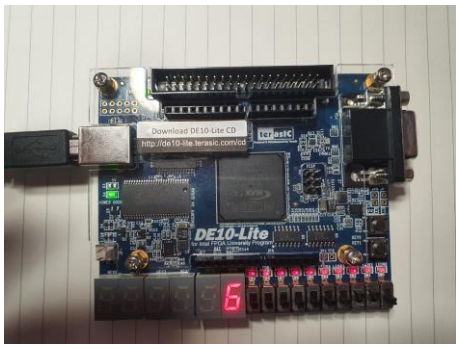
- 0100



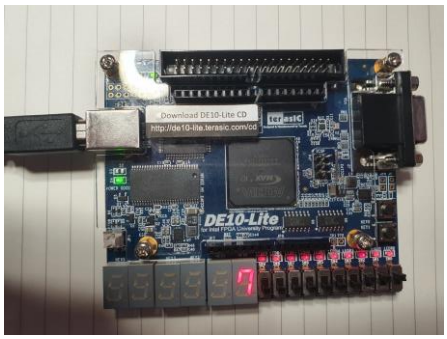
- 0101



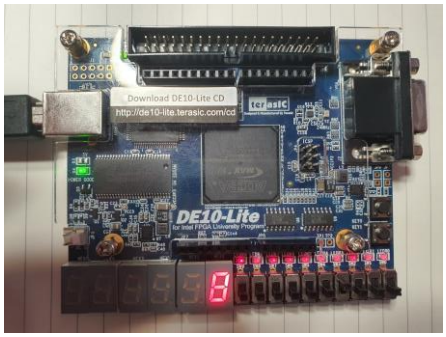
- 0110



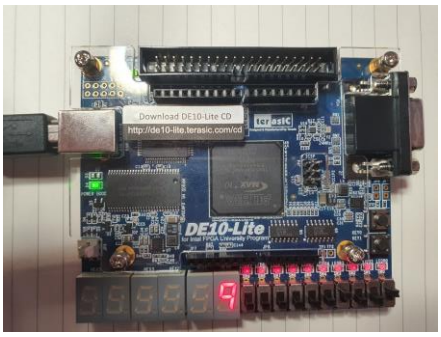
- 0111



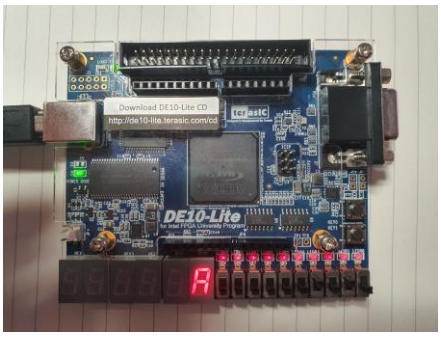
- 1000



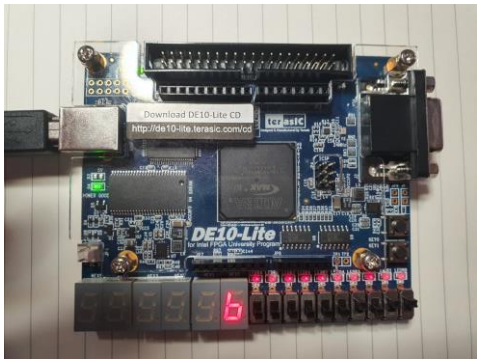
- 1001



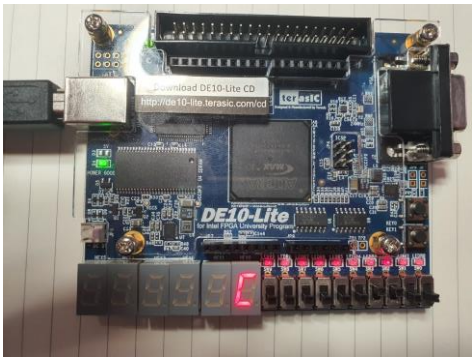
- 1010



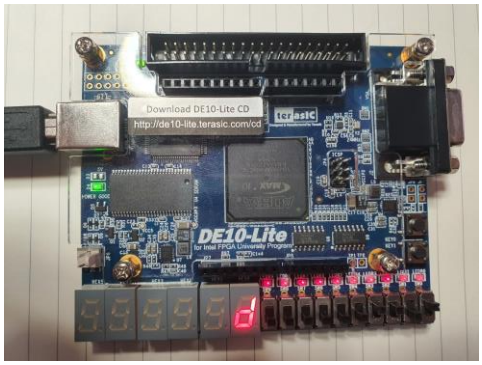
- 1011



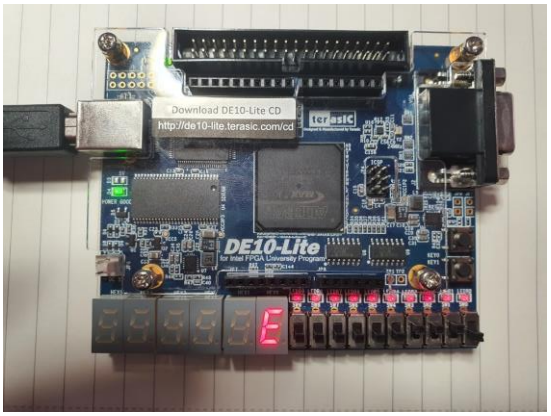
- 1100



- 1101

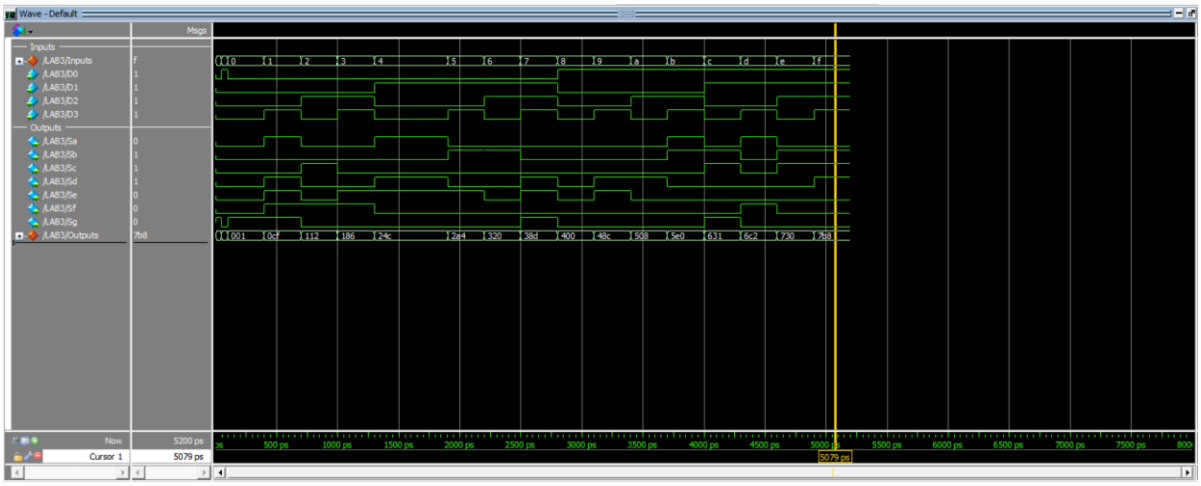


- 1101



- 1111

2) Simulation Results



3) Conclusion

The output of hardware was exactly same as the expected output in the design step. Moreover, the output of Simulation was accorded to both the actual results in the hardware and the expected results in the design step. Thus, the experiment is successful because the result of hardware matches the expected results and the verification via ModelSim simulation is accorded.

4. Experiment Notes

In this lab, there were some challenges. I was keep failing to find the output logic of each segment display. The fundamental reasons of these failure were that I was not familiar to K-map enough and tried to condense the results from the K-map. Thus, this led me huge loads of error in Boolean equation. After confronting the huge loads of errors, I decided to reduce condensing the results from the K-map. Then, the most challenging problem was solved, and the work went well. Designing a combinational logic by using logic gates and creating new symbols based on each segment display went well.

5. Study Questions

When is a simulation necessary? Was it useful for this section?

A simulation is necessary when the design is complicated and has many connections between other complicated so that it is hard to verify the design. In this section, the simulation was useful. The reason is that the design was complicated so that it was hard and had huge amounts of writings to verify by hand. By using simulation, the complicated verification by hand is solved easily.