ECE451Lab3

Huanjia Liu (ID:831993149)

Oct 09, 2020

Contents

1 Objective

During this lab, we are asked to design a design a control logic block using Cadence for subway signal controller.

2 Introduction

Design a subway signal controller for each subway station. Each section of tracks in a subway station will have a sensor to determine whether there is a train in that section and a signal with red, yellow, and green lights. You want the light in the track section to show red if there is a train in the track section in the very next station in the direction the train is traveling, yellow if there is no train in the track section in the very next station but the section after that is occupied, and green otherwise.

Tracks allow trains move both ways from left to right and vice versa. The input D indicates the direction allowed at the time. When it is 1, the left-to-right direction is allowed; when it is 0, the right-to-left direction is allowed. The outputs G, Y, and R should be 1 to light the green, yellow, and red signals, respectively. The input P is 1 if there is a train in this section (subway station). The signals Ai, Bi, Ao and Bo are signal used to communicate from/to the station to the right and from/to the station to the left. The signals Ai and Bi are received/sent from/to the section to the right of the station, and signals Ao and Bo are received/sent from/to the section to the left of the station.

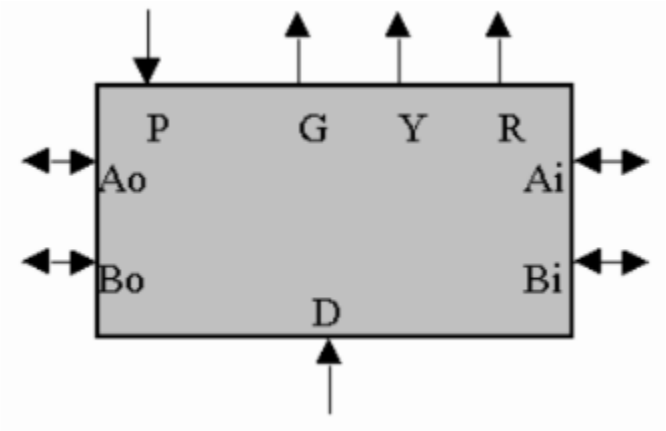

Figure 1: Control model

3 Procedure

3.1 Truth table of Control Model

At first, we need to write out the truth table to perform the control model and perform the logic minimization process to obtain the minimum Boolean equations. Although this part was asked in prelab, which has already submitted last week, I have show this part, because I invert the relationship between direction and light. Thus here is my special truth table for control model. Be careful!

D = 1

Input			Output				
A0	B0	P	Ai	Bi	G	Y	R
0	0	0	0	0	1	0	0
0	0	1	1	0	1	0	0
0	1	0	0	0	0	1	0
0	1	1	1	0	0	1	0
1	0	0	0	1	0	0	1
1	0	1	1	1	0	0	1
1	1	0	0	1	0	0	1
1	1	1	1	1	0	0	1

D = 0

Input			Output				
Ai	Bi	P	A0	B0	G	Y	R
0	0	0	0	0	1	0	0
0	0	1	1	0	1	0	0
0	1	0	0	0	0	1	0
0	1	1	1	0	0	1	0
1	0	0	0	1	0	0	1
1	0	1	1	1	0	0	1
1	1	0	0	1	0	0	1
1	1	1	1	1	0	0	1

Figure 2: Truth Table

3.2 Cadence Part

Then the truth table and equation need to be applied in Cadence in our Linux system. I separated it into several parts.

3.2.1 Tri-state buffer

Tri-state buffers is the kernel part of Bi-direction part, which can control the output through enable pin. Here is the diagram.

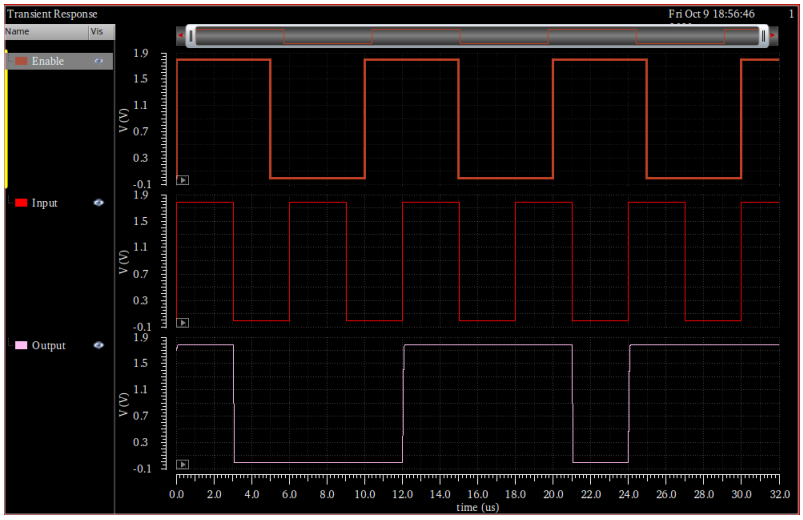

Figure 3: Tri-State Buffer

3.2.2 Bi Direction Port

In Lab3, the pin A_0, B_0, A_i, B_i are both input and output pin, thus we need to use tri-state buffer to achieve the function Bi direction port. Here is the diagram.

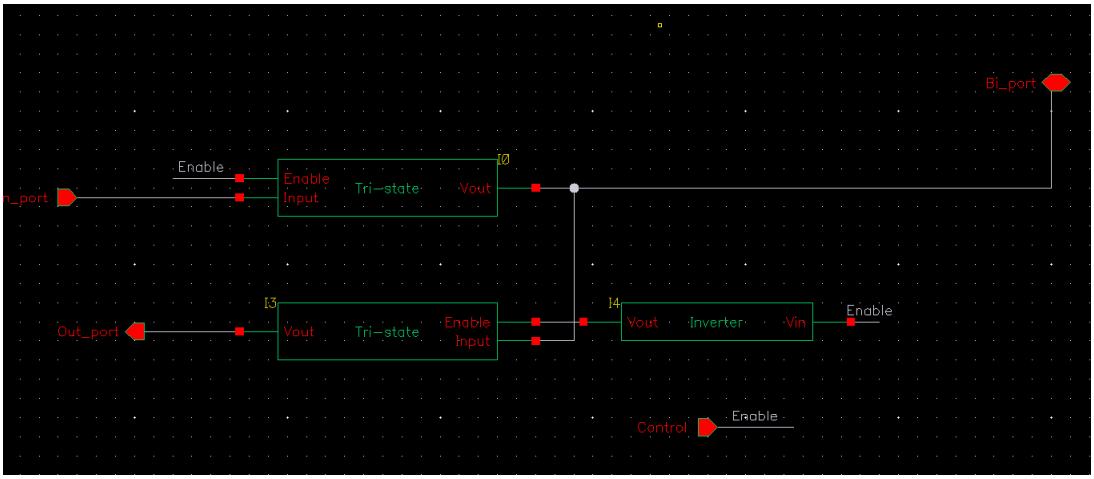

Figure 4: Bi Direction Port

3.2.3 Control Model

Above all, according to the truth table, tri-state buffer, and bi direction port, we can implement control model by combining them together. Here are Boolean equations when D = 1.

$$\mathrm{R} = A_0$$

$$\mathrm{Y} = A_0' B_0$$

$$\mathrm{Y} = A_0' B_0'$$

$$A_i = \mathrm{P}$$

$$B_i = A_0$$

Here are Boolean equations when D = 0.

$$\mathrm{R} = A_0$$

$$\mathrm{Y} = A_0' B_0$$

$$\mathrm{Y} = A_0' B_0'$$

$$A_0 = \mathrm{P}$$

$$B_0 = A_i$$

Here is my schematic of control model.

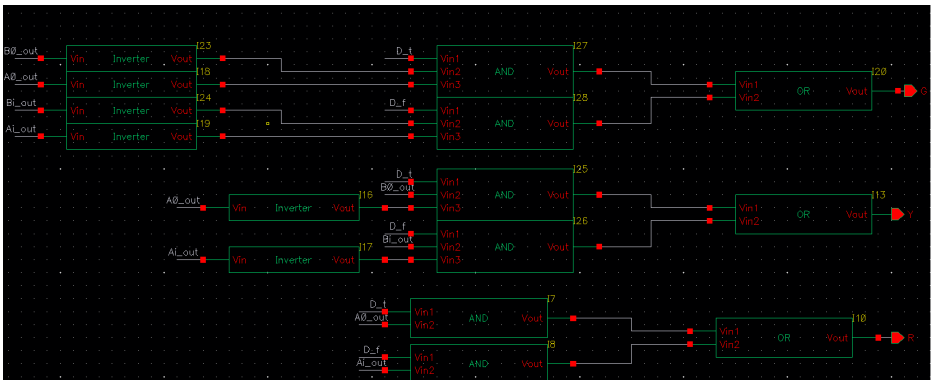

Figure 5: Control Model - Light part

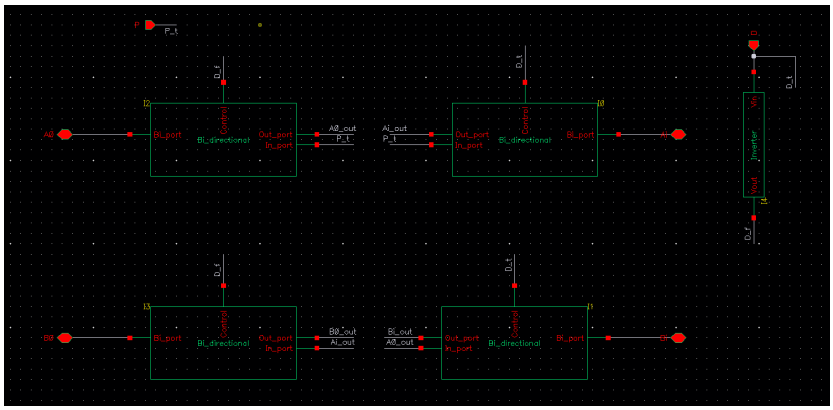

Figure 6: Control Model - Pin part

Here is the total view.

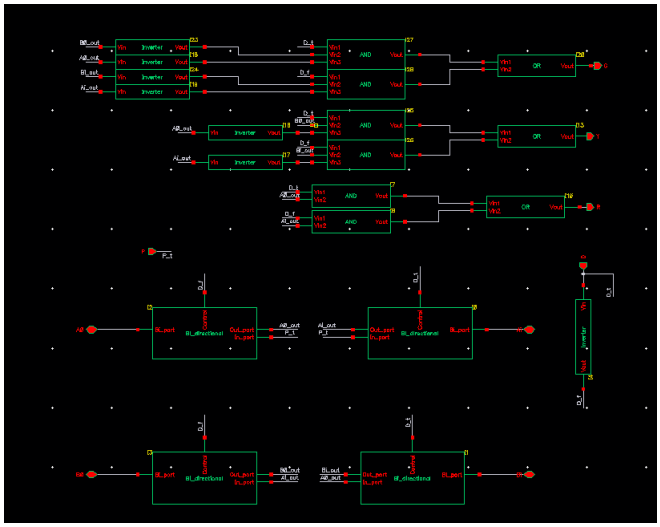

Figure 7: Control Model - Total view

3.2.4 3 Station Control Module

Combining three of control module together, we can get our 3 station control module, here is the diagram.

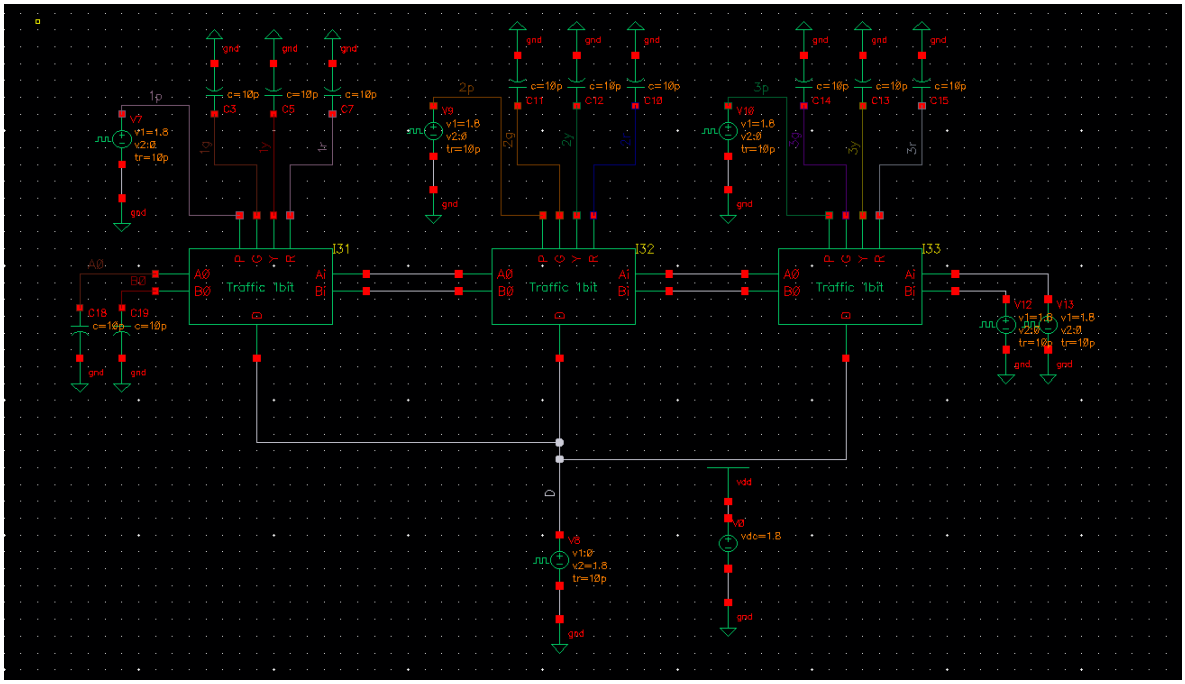

Figure 8: 3 station control module

4 Analysis

Testing your design is also your basic responsibility as an engineer.

4.1 Tri-state buffer

The figure below is tri-state buffer waveform. When enable is 1, output = input, when enable is 0, output keep the previous state.

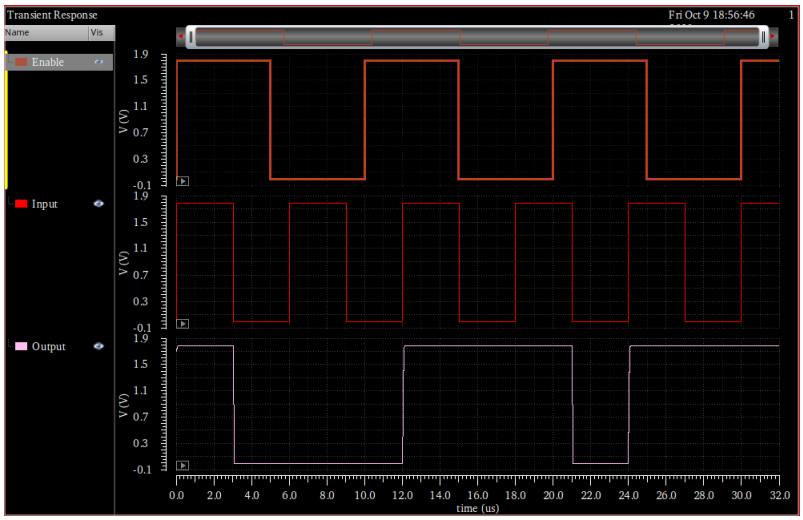

Figure 9: Tri-state Buffer Waveform

4.2 Station signal controller block for when D=0

Input: A_i, B_i, P, D

Output: A_0, B_0, R, G, Y

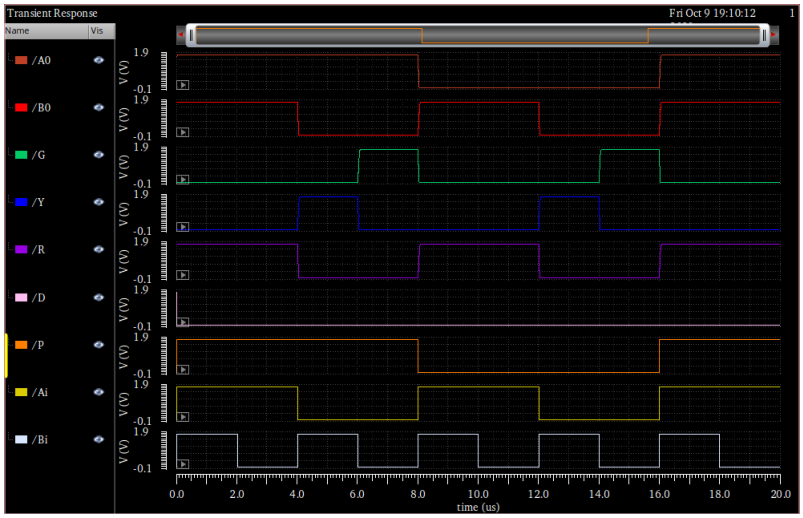

Figure 10: D = 0

4.3 Station signal controller block for when D=1

Input: A_0, B_0, P, D

Output: A_i, B_i, R, G, Y

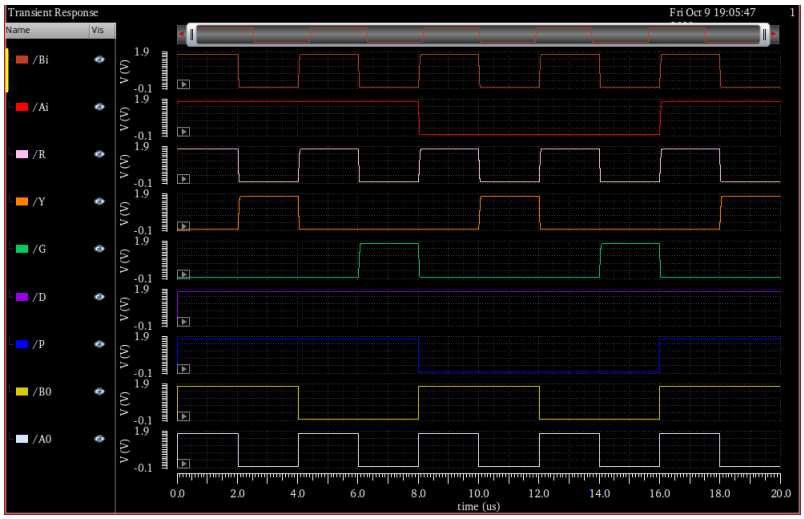

Figure 11: D = 1

4.4 Three station subway controller block for when D=0

Here, I set $A_i = 0$, $B_i = 0$. The figure below is my waveform. For example, when $p_1p_2p_3 = 001$, the light in the first station shows yellow, in the second station shows red, and in the third station shows green.

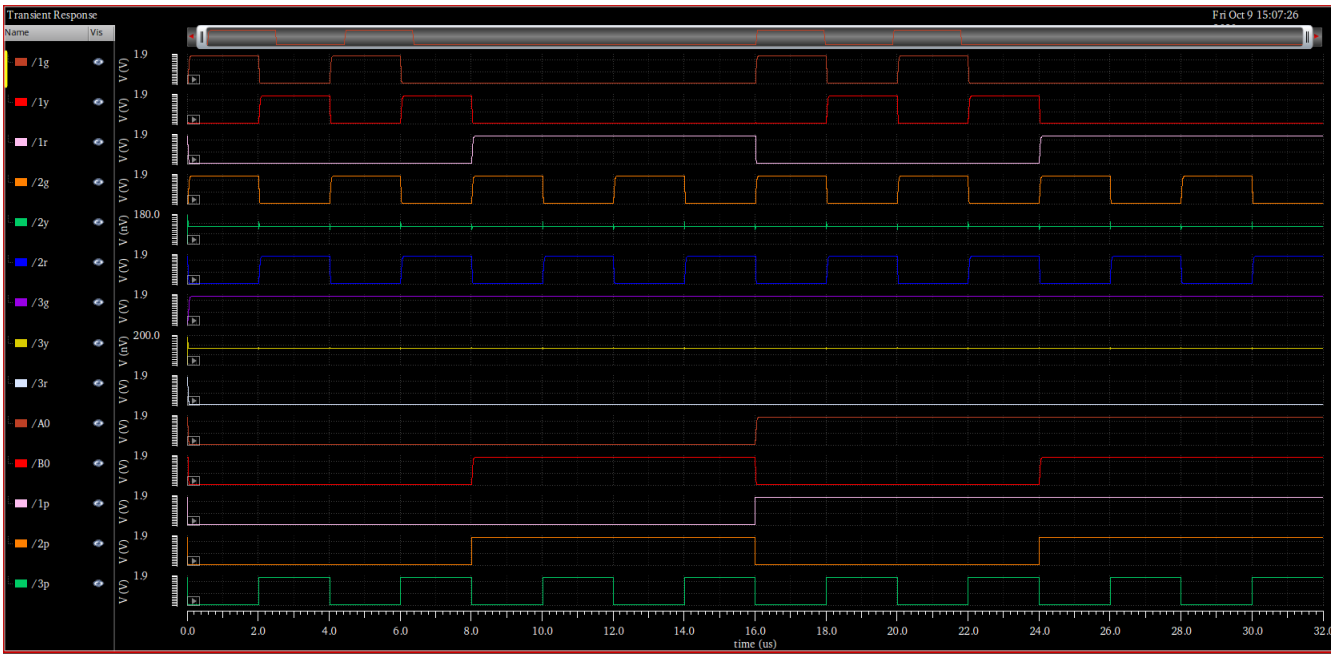

Figure 12: D = 0

4.5 Three station subway controller block for when D=1

Here, I set $A_0 = 0$, $B_0 = 0$. The figure below is my waveform. For example, when $p_1p_2p_3 = 101$, the light in the first station shows green, in the second station shows red, and in the third station shows yellow.

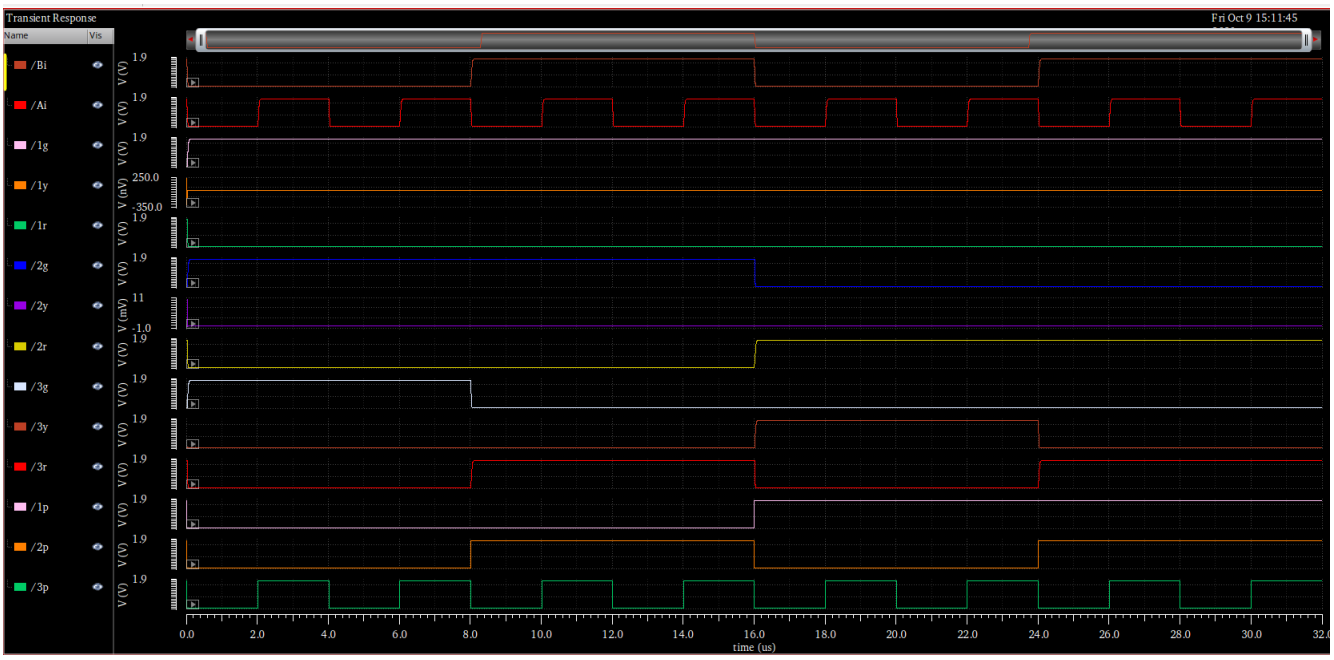

Figure 13: D = 1

5 Questions

5.0.1 What is a static 1-hazard and a static 0-hazard?

Static 1-hazard If the output is currently at logic state 1 and after the input changes its state, at the same moment the output changes to 0 before settling on 1, then it is a Static-1 hazard.

5.0.2 Discuss how you remove a static 1-hazard and a static 0-hazard.

We can use tri-state buffer. As we find in /Analysis/Tri-state buffer, when enable is equal to 0, the output will keep the previous state, which will help a lot to remove static 1-hazard or static 0-hazard.

6 Conclusion

Now we can get the conclusion, I finish this lab successfully, and achieve all the objective.