Lab3 Seven Segment Display

1. Introduction

In this, we learned how the seven segment display works and design a system to drive the segment display. Here still, we need to design both sequential logic and combinational logic separately just as we did in the previous labs.
In board Nexy4 DDR, the seven segment display module has 8 display blocks with 7 segment strip LED on each of them. And for save of ports, we do not control all the LED in the same time, which will cost totally $7 \times 8 = 56$ ports for all the segment LEDs. Here in the board, we use a very trick method to avoid the numerical ports. We choose the each display block each time and give the signal what it should display, then shift to the next block and do the same thing again. If our refreshing speed is high enough, the effect of persistence of vision will help us to see all the blocks are displaying.
Our general idea of the design is to first generate a sweeping signal produced by the module anode driver, which will sweep the 8 display block in a constant frequency. And then we store the data we want to display in a large register by another module named display driver. It will receive the signal what in the current activated display block and gives out the correct the output for that one.
The details will be introduced in the rest of the report.

2. Pre-lab Preparation

2.1. Principle of the seven-segment

From the user guide of our board, the circuits for the seen segments is shown below in **figure 2.1.1**

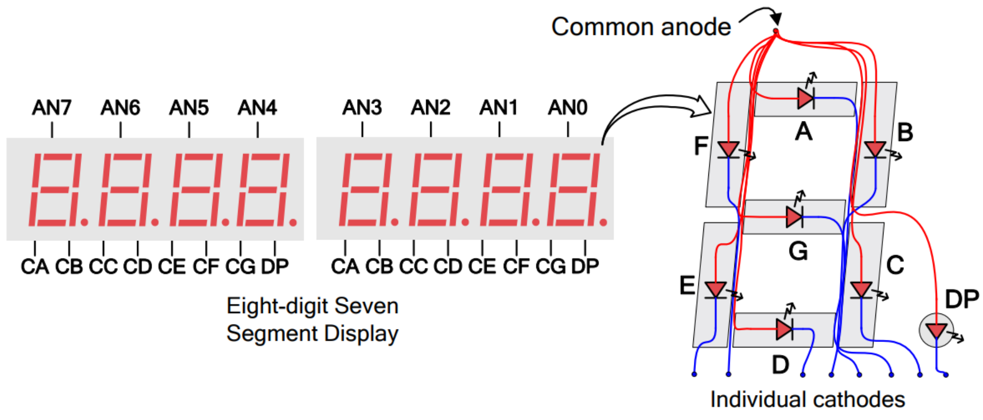


Figure 2.1.1 circuits of the seven segment display

The display in our board is in common anode mode. Thus the block choose signal is the anode choose. Only the display block with the activated anode one can be driven by the source (means can be lighted). When the display block is activated, we just need to set different values of the cathodes, and the segments will display different digital. Here we ignore the DP for our experiment is quite simple.
For a better visual effect and also protect the device, the refreshing speed should not be too quick. A range from 60Hz to 1kHz would be reasonable. In this lab we use the frequency 1kHz.

2.2. Block Diagram

The system block diagram is shown below in **figure 2.2.1**.

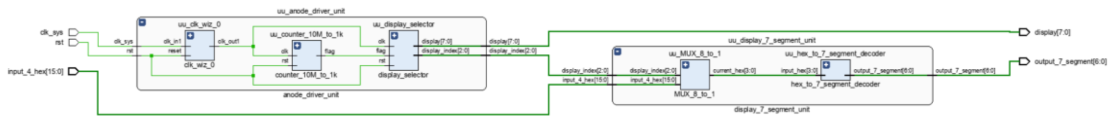


Figure 2.2.1 system block diagram

We design the following modules:

Top module 'seven_segment_display_top', which has two lower level top module:

For the 'anode_driver_unit':

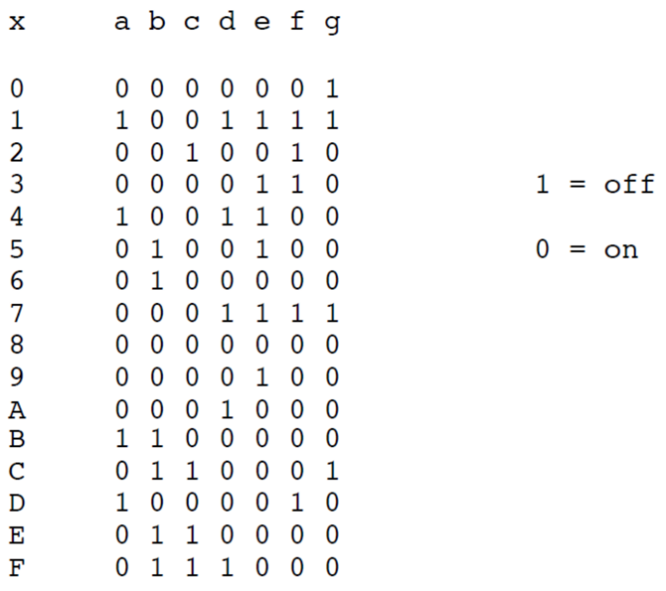
1. 'counter_10M_to_1k', this is designed because our slowest clock is 10MHz from the IP core generated module 'clk_wiz_0', we need have a slower enable signal.
2. 'display_selector', this is designed because we have to generate the signals to activate each anode in order.

For 'display_7_driver_unit':

1. 'Mux_8_to_1', this is a multiplexer, which calculates and stores all the values for all the display blocks. This receive the activate signal from the 'display_selector' and gives out the value in hex for the current activated display block.
2. 'hex_to_7_segment_decoder', this is a decoder to translate the hex from the last module to the segments display form.

2.3. Truth table

The truth table for the module 'hex_to_7_segment_decoder' is shown in **figure 2.3.1**.



x	a	b	c	d	e	f	g
0	0	0	0	0	0	0	1
1	1	0	0	1	1	1	1
2	0	0	1	0	0	1	0
3	0	0	0	0	1	1	0
4	1	0	0	1	1	0	0
5	0	1	0	0	1	0	0
6	0	1	0	0	0	0	0
7	0	0	0	1	1	1	1
8	0	0	0	0	0	0	0
9	0	0	0	0	1	0	0
A	0	0	0	1	0	0	0
B	1	1	0	0	0	0	0
C	0	1	1	0	0	0	1
D	1	0	0	0	0	1	0
E	0	1	1	0	0	0	0
F	0	1	1	1	0	0	0

1 = off

0 = on

Figure 2.3.1 Truth table for 'hex_to_7_segment_decoder'

And the truth table for module ‘display_selector’ is

Current segment	input	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0	output
AN0	000	1	1	1	1	1	1	1	0	11111110
AN1	001	1	1	1	1	1	1	0	1	11111101
AN2	010	1	1	1	1	1	0	1	1	11111011
AN3	011	1	1	1	1	0	1	1	1	11110111
AN4	100	1	1	1	0	1	1	1	1	11101111
AN5	101	1	1	0	1	1	1	1	1	11011111
AN6	110	1	0	1	1	1	1	1	1	10111111
AN7	111	0	1	1	1	1	1	1	1	01111111

Figure 2.3.2 Truth table for ‘display_selector’

2.4. Module simulation

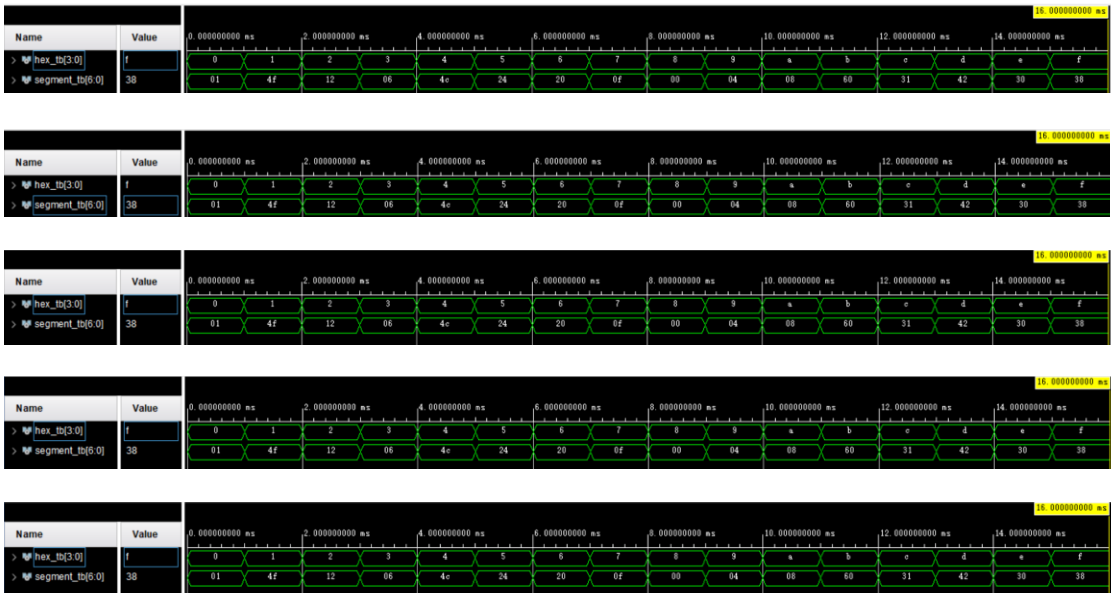
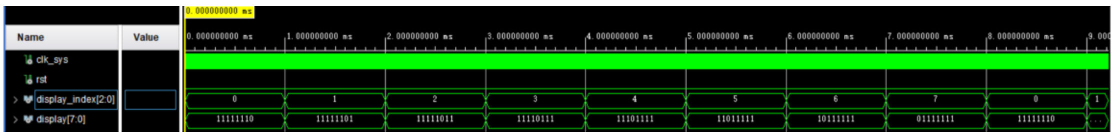

Name
Value
hex_tb[3:0]
segment_tb[6:0]
0 1 2 3 4 5 6 7 8 9 a b c d e f
01 4f 12 06 4c 24 20 0f 00 04 08 60 31 42 30 38

Figure 2.4.1 simulation of the module ‘hex_to_7_segment_decoder’

From the top to bottom are behavior, post-synthesis function, post-synthesis timing, post-implement function, post-implement timing simulation.

From the results above we can see that the module works correctly, and all the simulation produce the same results.



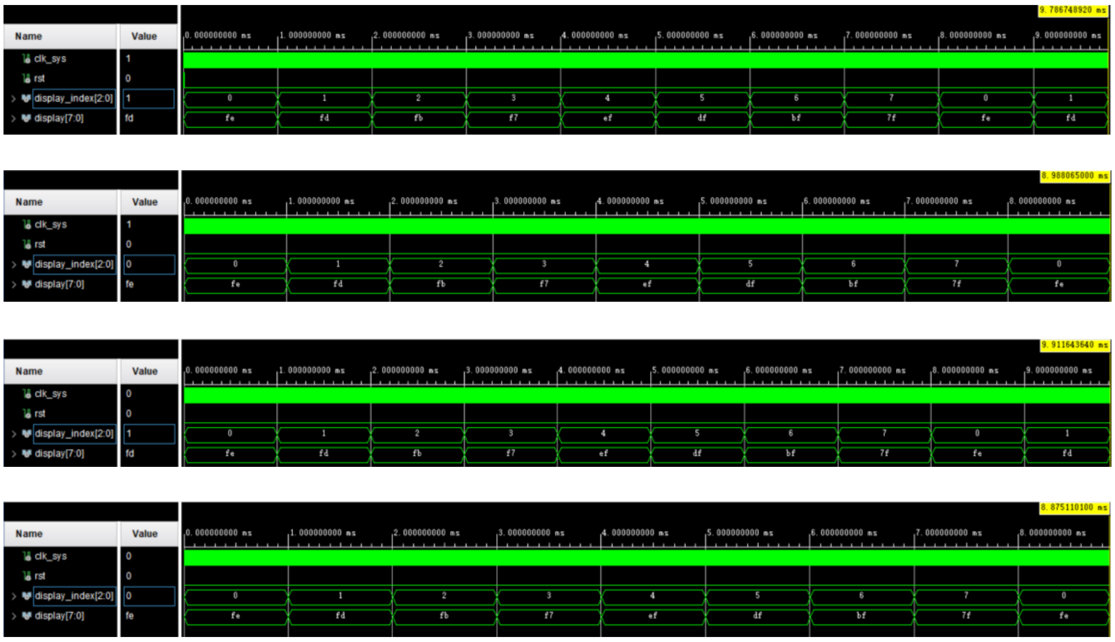


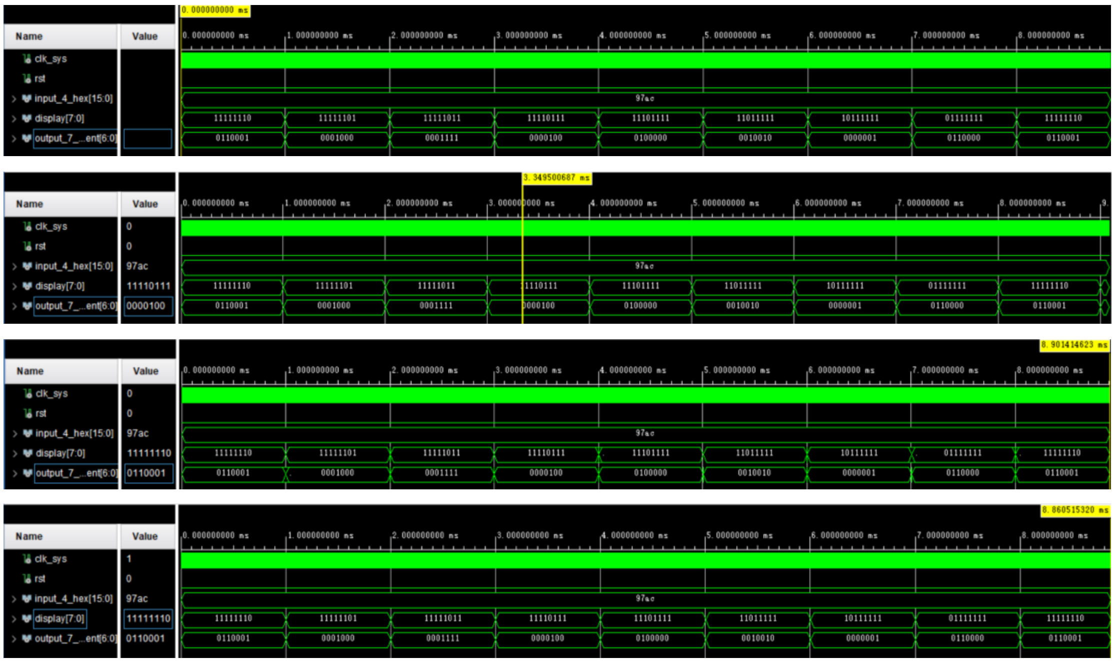
Figure 2.4.2 simulation of the module 'display_selector'

From the top to bottom are behavior, post-synthesis function, post-synthesis timing, post-implement function, post-implement timing simulation.

From the results above we can see that the module works correctly, and all the simulation produce the same results.

3. Simulation Result

We set the test input is '1001_0111_1010_1100'. Thus A is 1100 (C), B is 1010 (A), C is 0111 (7), D is 1001 (9). We do not consider the overflow and underflow. Thus A+B=0110 (A), A-B=0010 (2), C+D=0000 (0), C-D=1101 (E) .And here are the results



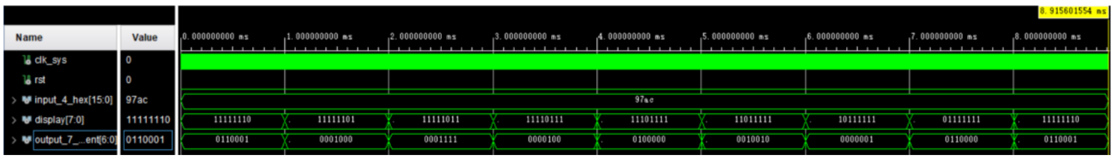
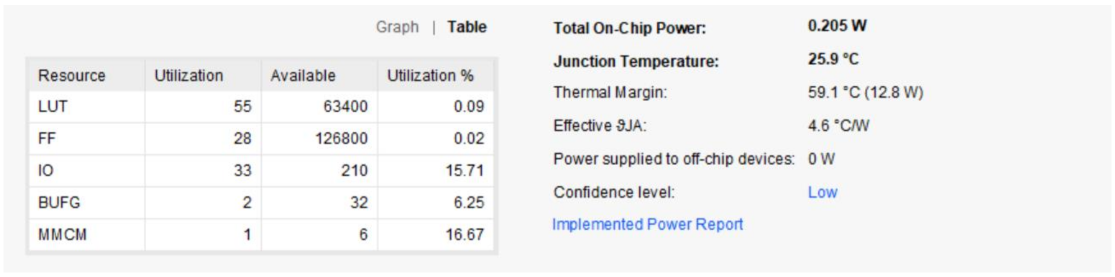


Figure 3.1 simulation of the top module 'seven_segment_display_top'

We can find that it works normally, and the timing delay is obviously negligible because our frequency of change of state is slow to 1kHz.

And the utilization of resource and power usage is shown in **figure 3.2**


Graph | Table

Resource	Utilization	Available	Utilization %
LUT	55	63400	0.09
FF	28	126800	0.02
IO	33	210	15.71
BUFG	2	32	6.25
MMCM	1	6	16.67

Total On-Chip Power:	**0.205 W**
Junction Temperature:	**25.9 °C**
Thermal Margin:	59.1 °C (12.8 W)
Effective ϑJA:	4.6 °C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Implemented Power Report

Figure 3.2 utilization of resource and power usage

The overall implementation is shown in **figure 3.3.**

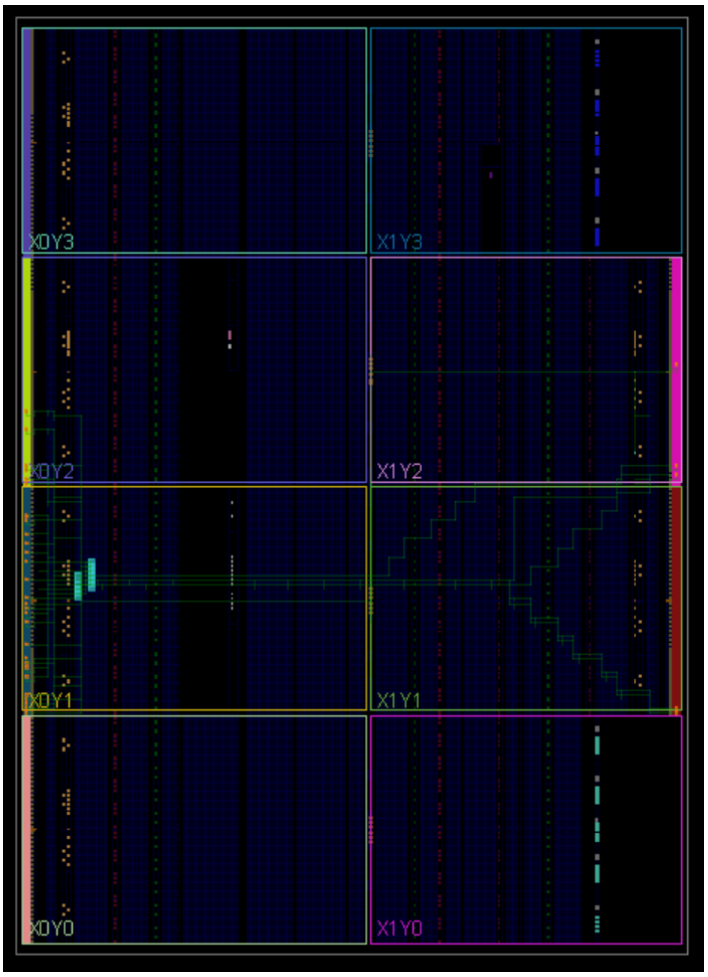
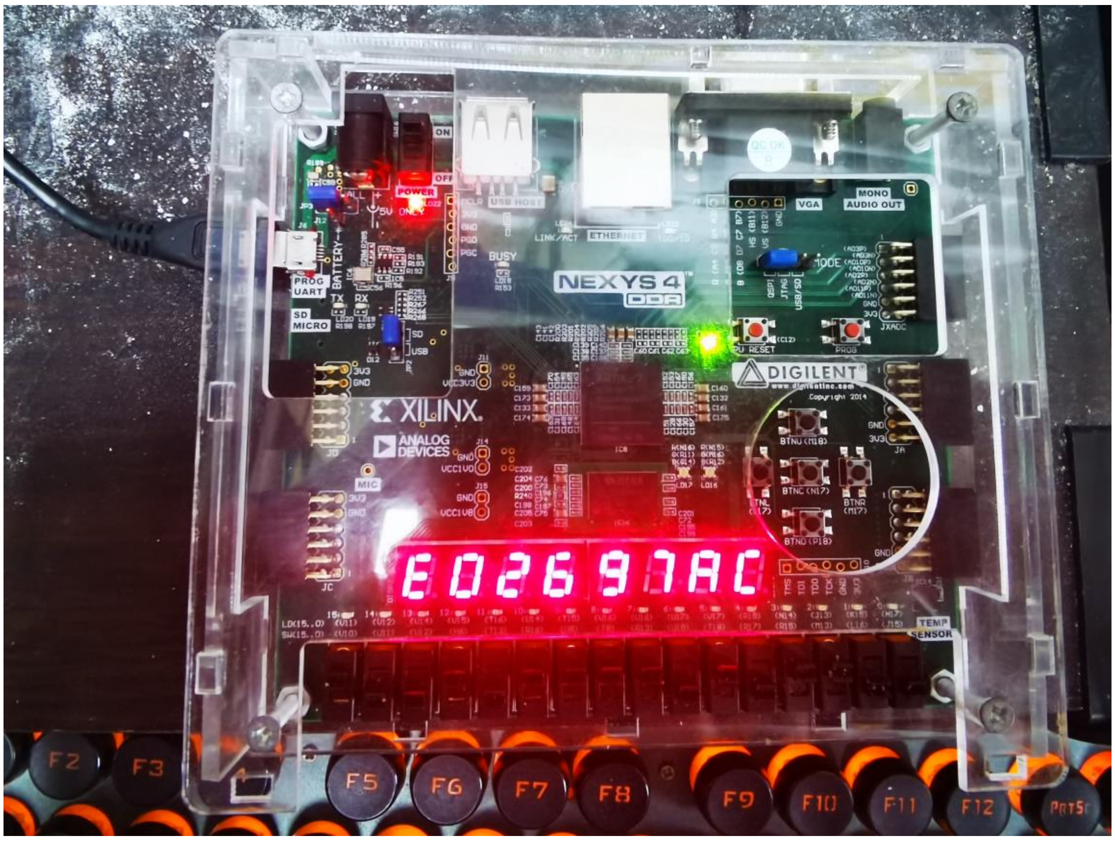


Figure 3.3 implementation

4. On Board Test

Here is the on board test. The system works well.



5. Conclusion

In this lab, we learned overall idea of how to build up a complete system from the base modules step by step, and create different level of test bench to simulate each module independently to make sure they work well. The level design, coming up with the module design is the two main design idea in the hardware design.

For the lab itself, we realize the display of the 7 segment display with 8 display block successfully.

Due to time limits, we did not introduce and explain the detailed code in the report. This is because most of the code are either easy to implement or we have already implement successfully in the previous lab. Thus there is no challenge in coding. Hence we only introduce the main module design idea here. Detailed code can refer to the attachments.

#something else, there is really not too much thing to talk about after the design abaaba