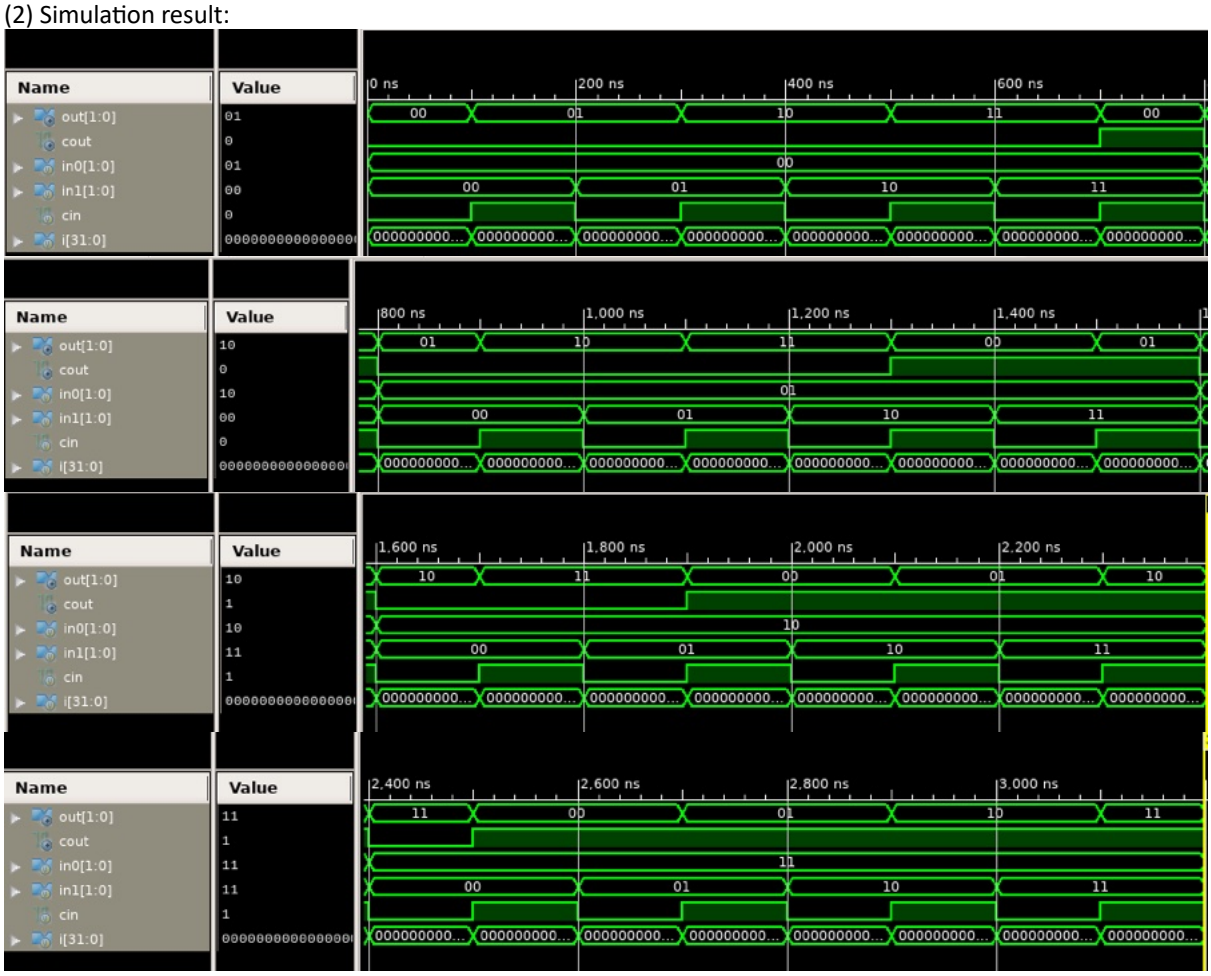
Lab05 Report

Park Chaehyun

1. Lab: Ripple-carry adder

(1) Source code: In the attached source code, the two inputs are each represented as a 2-bit array, ‘in0[1:0]’ and ‘in1[1:0]’.And the carry-in is represented as ‘cin’,the output as a 2-bit array ‘out[1:0]’and the carry-out as ‘cout’.

(2) Simulation result:

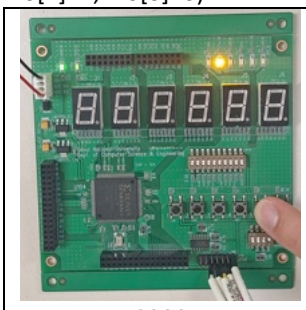


We can check that each input combination gives the expected output combination.

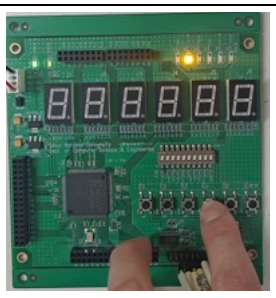
(3) Test with programmed FPGA result:

The inputs are each mapped to tactile switches, with pin numbers 47, 48, 49, 50 and 51, in the order: in0[1], in0[0], in1[1], in1[0] and cin. And the outputs are each mapped to LEDs, with pin numbers 87(red), 88(yellow) and 90(green), in the order: out[1], out[0] and cout. The pictures below show the inputs and outputs in the same order, from left to right. The switches are numbered from 1 to 5, and the LEDs are numbered from 01 to 03 in the picture.

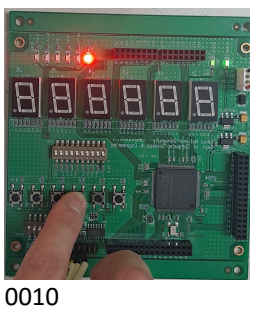
(i) cin = 0, and the rest of the inputs in the order: in1[1], in1[0], in0[1], in0[0] (0110 means in1[1]=0, in1[0]=1, in0[1]=1, in0[0]=0)



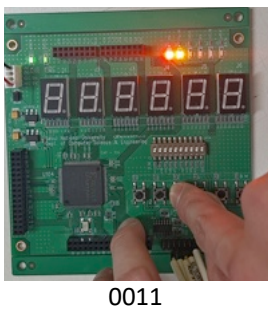
0000



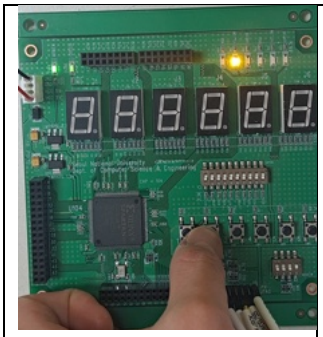
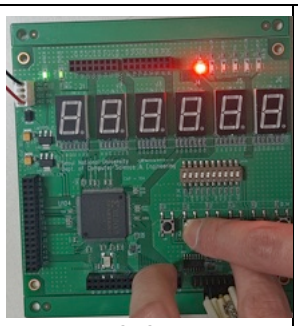
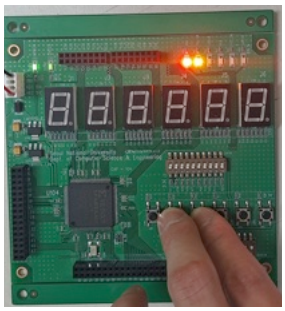
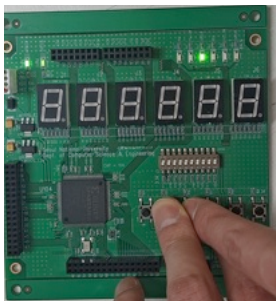
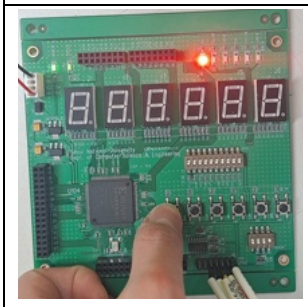
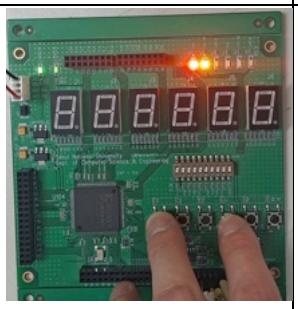
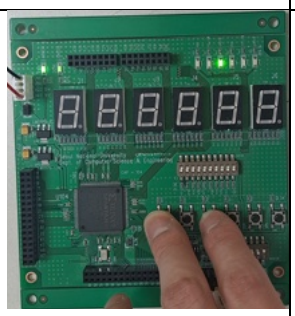
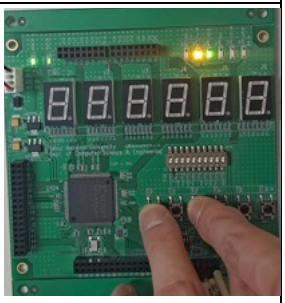
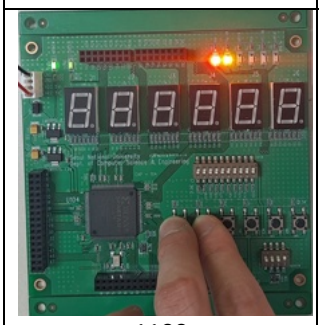
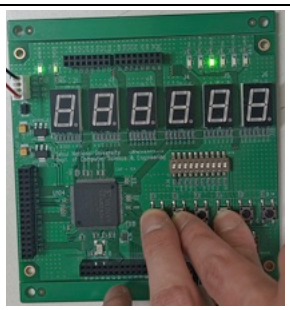
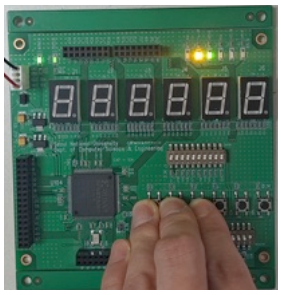
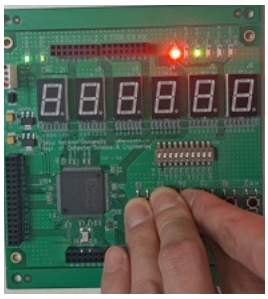
0001



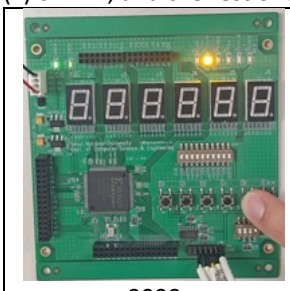
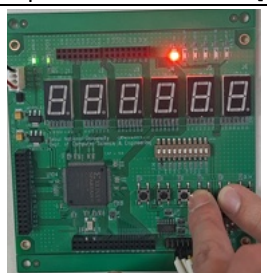
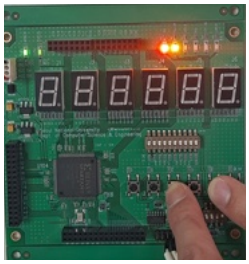
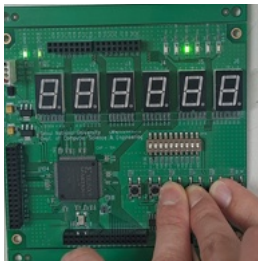
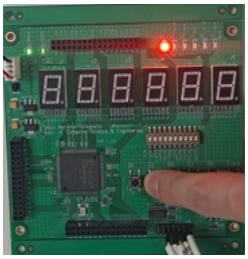
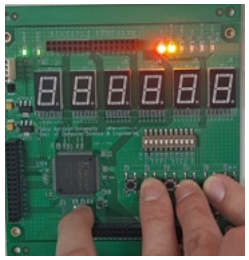
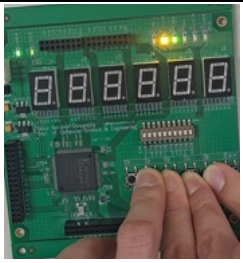
0010

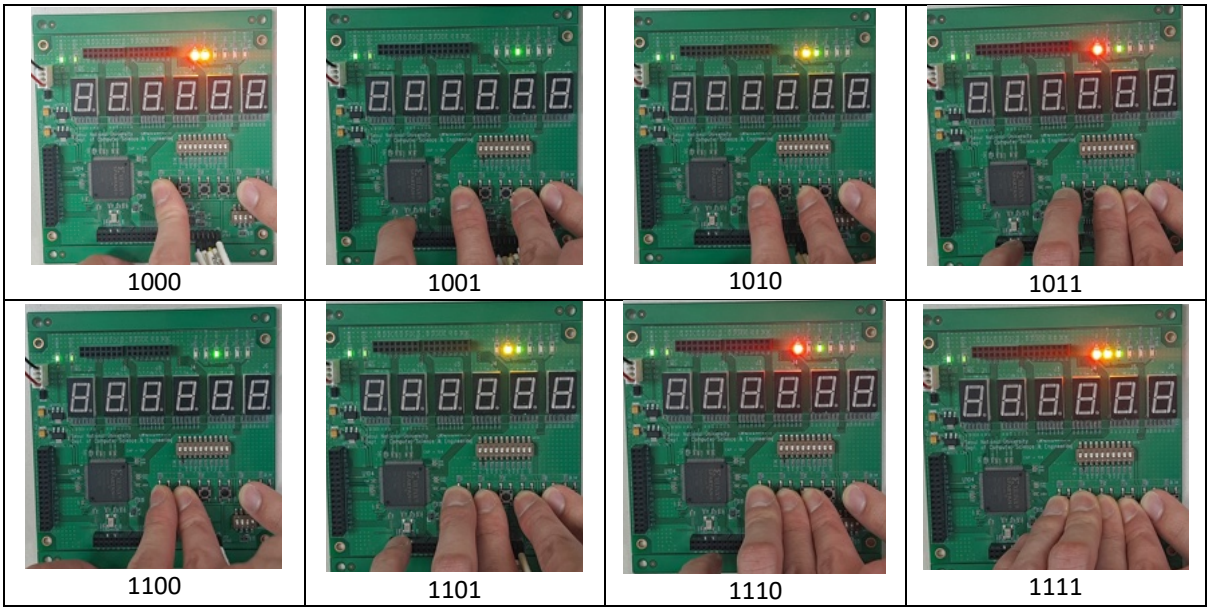


0011

 0100	 0101	 0110	 0111
 1000	 1001	 1010	 1011
 1100	 1101	 1110	 1111

(ii) cin = 1, and the rest of the inputs in the order: in1[1], in1[0], in0[1], in0[0]

 0000	 0001	 0010	 0011
 0100	 0101	 0110	 0111



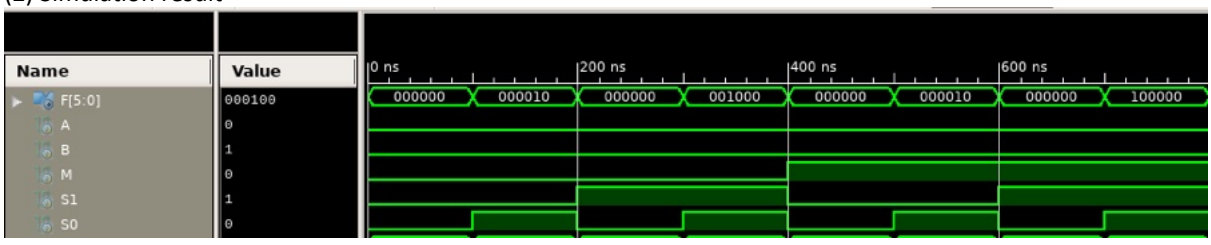
We can check that the output for each input matches that of the simulation result.

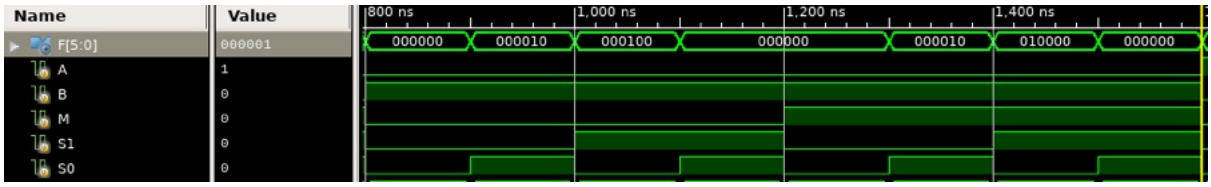
2. Homework: 1-bit arithmetic logic unit (ALU)

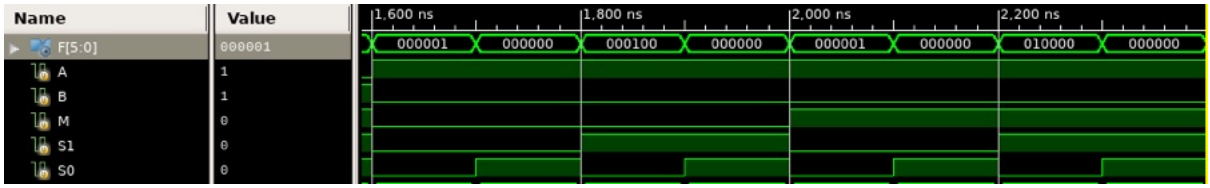
(1) Source code

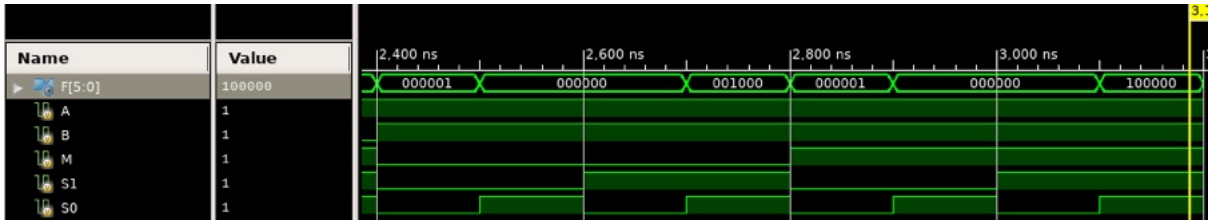
In the attached source code, the three control inputs are represented as M, S1 and S0, the two data inputs represented as A and B, and the output is represented as a 6-bit array F[5:0]. The array, in the increasing order of the index, represents LEDs with pin numbers P87, P88, P90, P91, P92 and P93.

(2) Simulation result





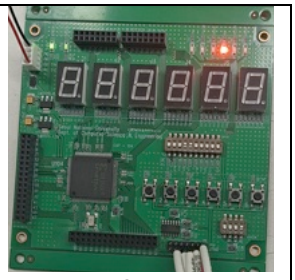
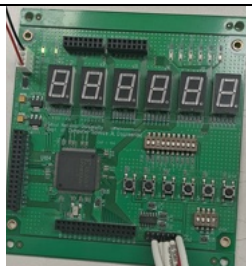
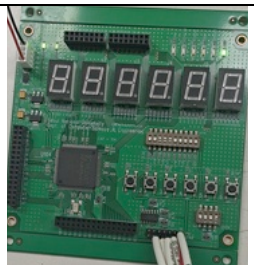




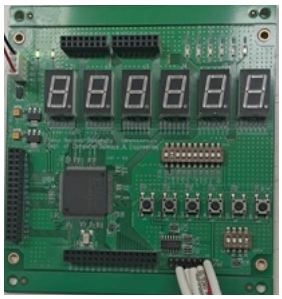
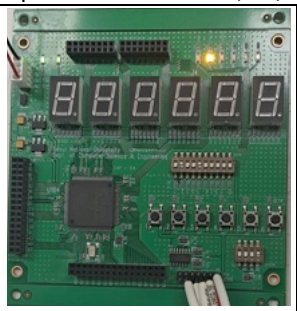
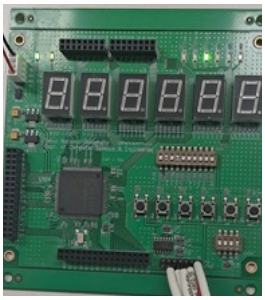
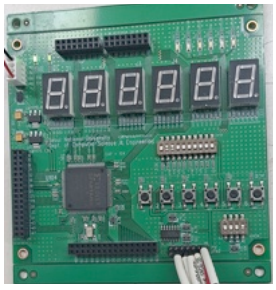
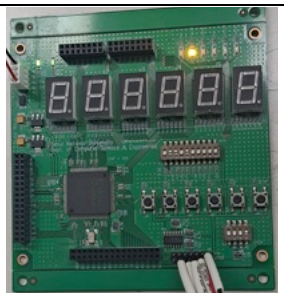
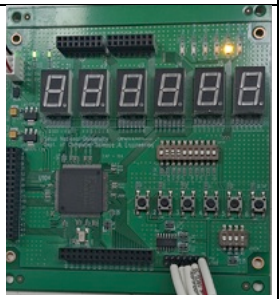
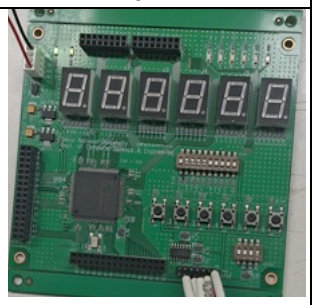
We can check that each input combination gives the expected output combination.

(3) Test with programmed FPGA result:
The inputs are each mapped to DIP switches, with pin numbers 30, 31, 32, 33 and 41, in the order: A, B, M, S1, S0. And the outputs are each mapped to LEDs, with pin numbers 87(red), 88(yellow), 90(green), 91(red), 92(yellow) and 93(green) in the order: F[0], F[1], F[2], F[3], F[4] and F[5]. The pictures below show the inputs and outputs in the same order, from left to right. The switches are numbered from 1 to 5, and the LEDs are numbered from 01 to 06 in the picture.

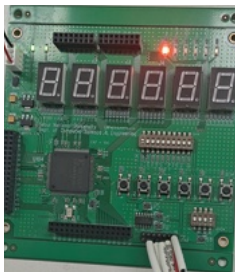
(i) A=0, B=0, the rest of the inputs in the order: M, S1, S0

 000	 001	 010	 011
 100	 101	 110	 111

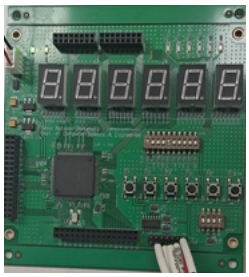
(ii) A=0, B=1, the rest of the inputs in the order: M, S1, S0

 000	 001	 010	 011
 100	 101	 110	 111

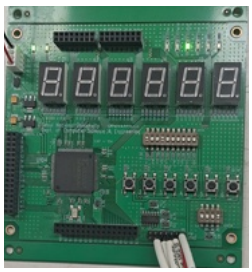
(iii) A=1, B=0, the rest of the inputs in the order: M, S1, S0



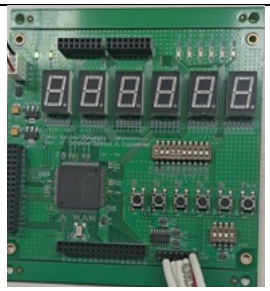
000



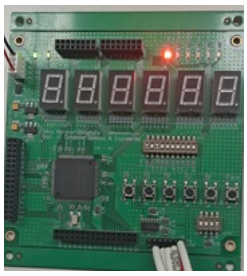
001



010

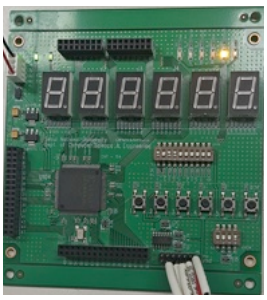


011

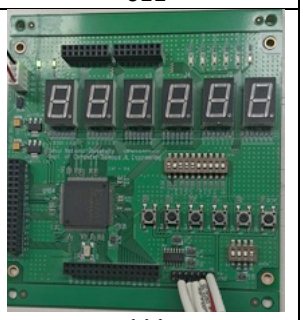


100

101

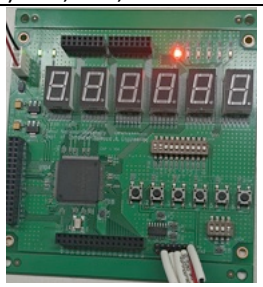


110

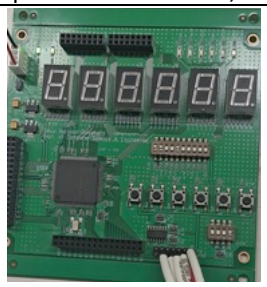


111

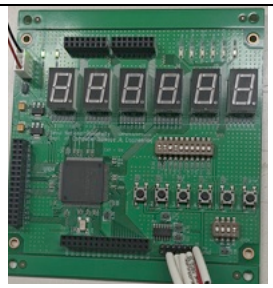
(iv) A=1, B=1, the rest of the inputs in the order: M, S1, S0



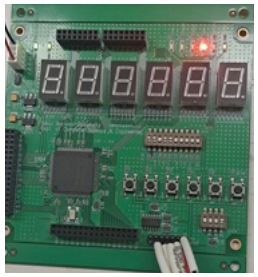
000



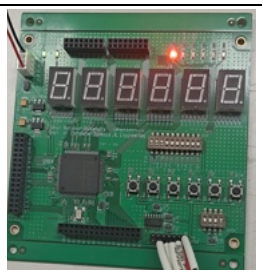
001



010

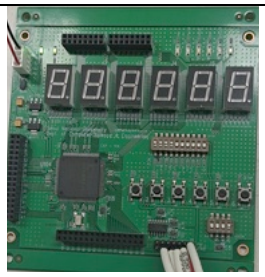


011

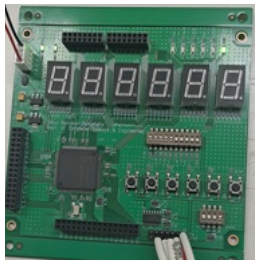


100

101



110



111

We can check that the output for each input matches that of the simulation result.

(3) Discussion

In this week's lab session, we not only wrote code but programmed the written code on the FPGA board. This enabled us to give input via switches and check the output on the LEDs, instead of checking only the simulation result. Working with actual hardware enabled us to understand the purpose of writing and using hardware description languages such as Verilog more clearly.