| Ousmane Toure & Jianning Chen  EECE 2160 | Embedded Design: Enabling Robotics  Lab Assignment 3 |
| --- | --- |

Lab Assignment 3

Ousmane Toure & Jianning Chen

Toure.o@northeastern.edu

chen.jiann@northeastern.edu

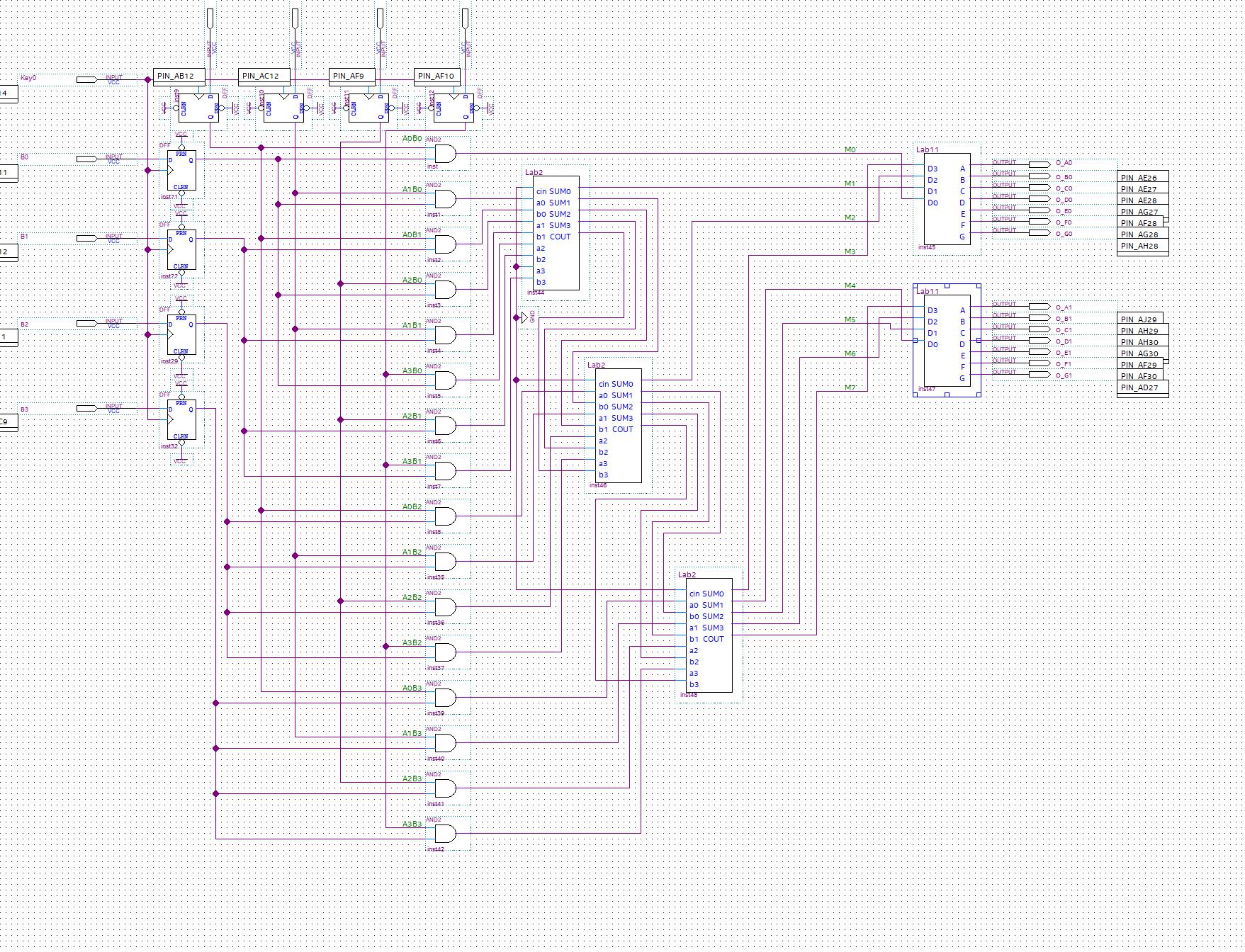
Submit date: 5/26/2022

Due Date: 5/27/2022

**3.1 Section 1 of Lab 3**

The goal of Section 3.1 of Lab 3 was to create a 4 bit multiplier utilizing logic gates to mimic a hardware challenge during a job interview. The values would be inputted by utilizing switches 7 to 0. The logic of the circuit is as follows: SW3 - SW0 represent A3 -A0 which is the input representing a one 4-bit number and SW7 - SW4 represents B3-B0 which is the input for another 4-bit number. Instead of just placing dozens of logic gates making an extremely long schematic, Full adders from Lab 2 were utilized along with 16 AND gates represented in the schematic above. Multiplying each set of numbers outputted partial products which were then placed into the full adders. If any carry bit was created it would be pushed to the sequential full adder. Next, a DFF(D flip flip) was added to each individual toggle switch. A DFF works by saving the edge input of the clock, and delays the output signal until the next clock signal occurs. The clock is controlled by a button which in this section is controlled by pressing “Key0” on the board. Two 7 segment displays were utilized, The outputs M3-M0 were assigned to one display and M7-M4 were assigned to the second display. Our results were outputted to the LED displays using **Hex Decimal.** When the DFF was connected to our 7 segment displays, we were able to keep the LED value displayed until the rising edge was activated by pressing KEY0.

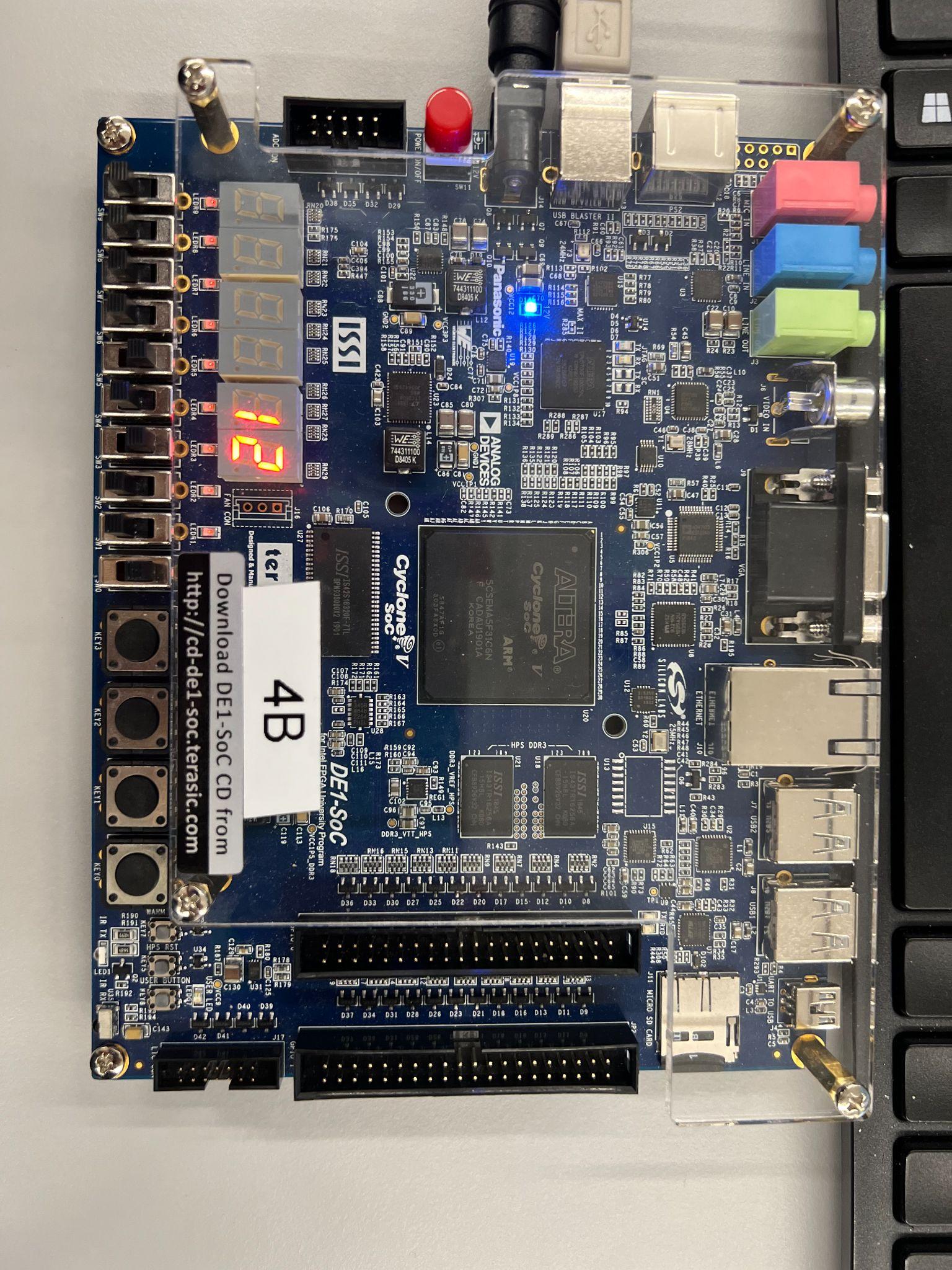
**3.2 Circuit Design**

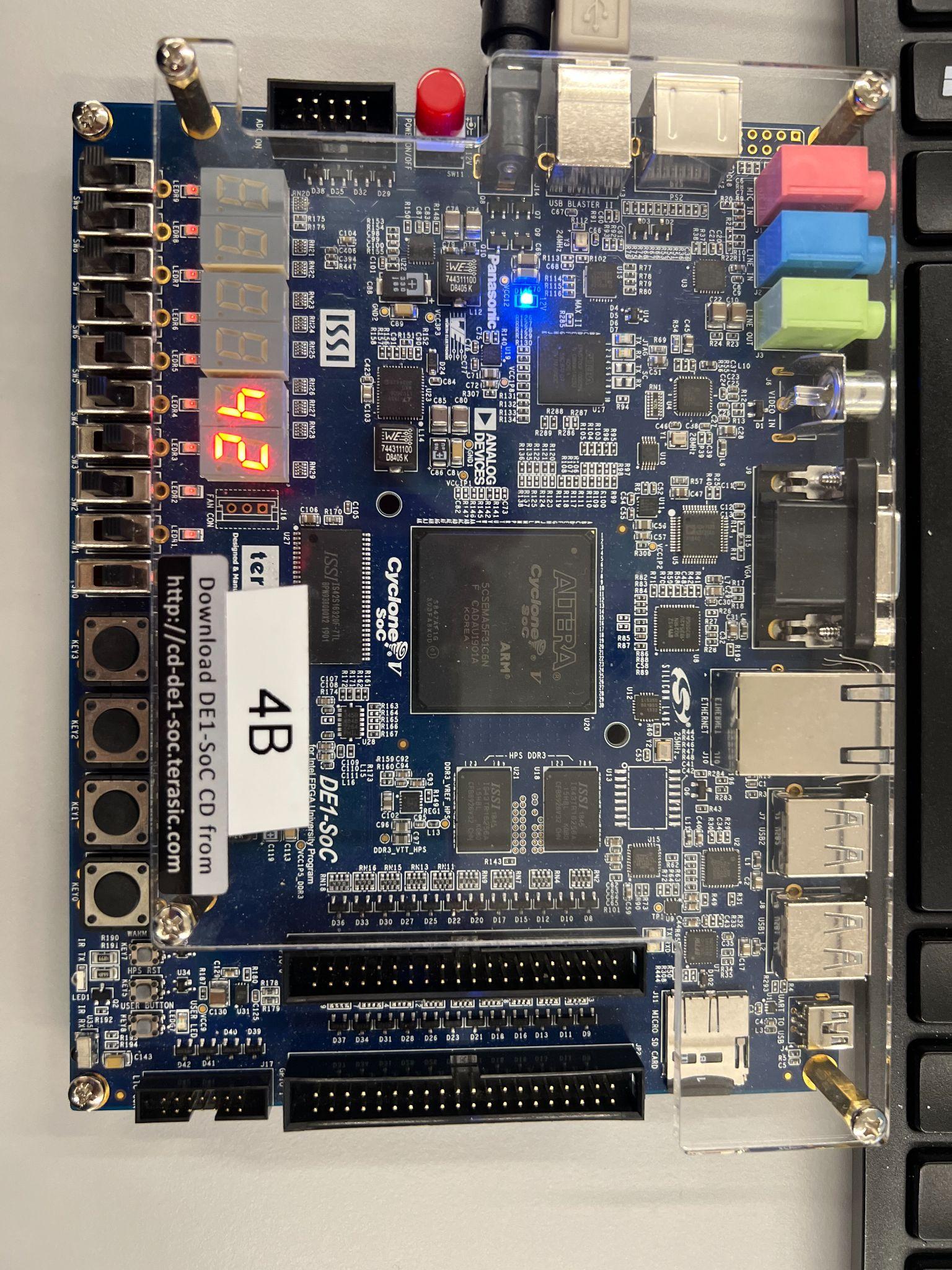
****

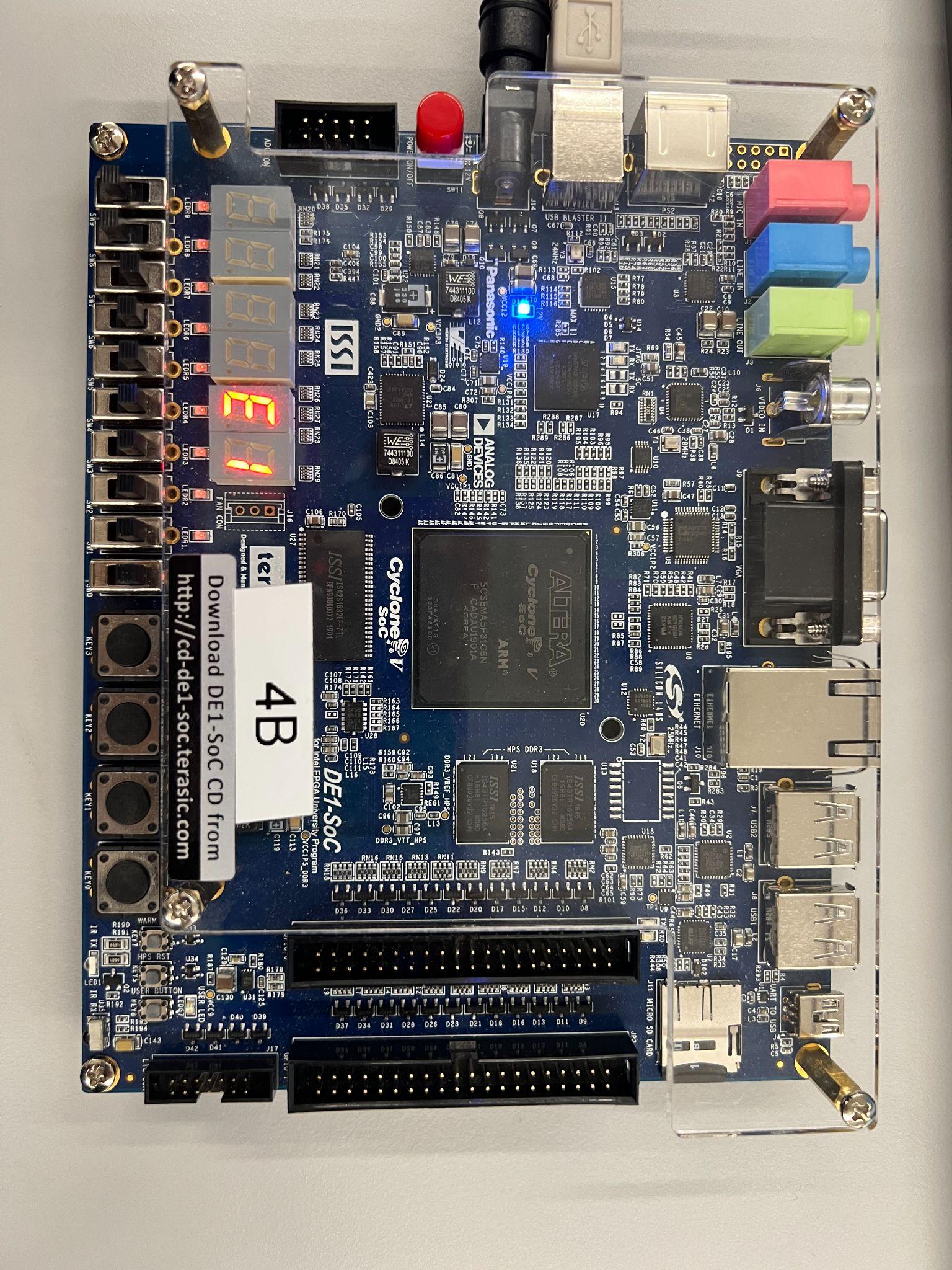
As explained in prelab questions, the circuit design will be 16 AND gates combining the results for different digits from number A and number B. The logic behind the 4-bit multiplier is using 4-bit adders and combining them in series. In this case 3 4-bit adders are needed. All the 4-bit full adders are the design from the work in lab 2.

In addition, the output is shown in two 7-segment displays. Since there will be 8 bits total as outputs, the entire output can be broken into 2 parts, and each part consists of 4 bits that can be directly transformed into a hex digit. That corresponds to the task in lab 1, where we were asked to input 4 bits in binary form and show them on a 7-segment display using hexadecimal form. Therefore, the 2 blocks on the right side of the picture above is the work from lab 1, where 4 digits are inputs and the 7 outputs on the other side are just outputs corresponding to the different segments on the display.

**3.3 Implementation Results**

****

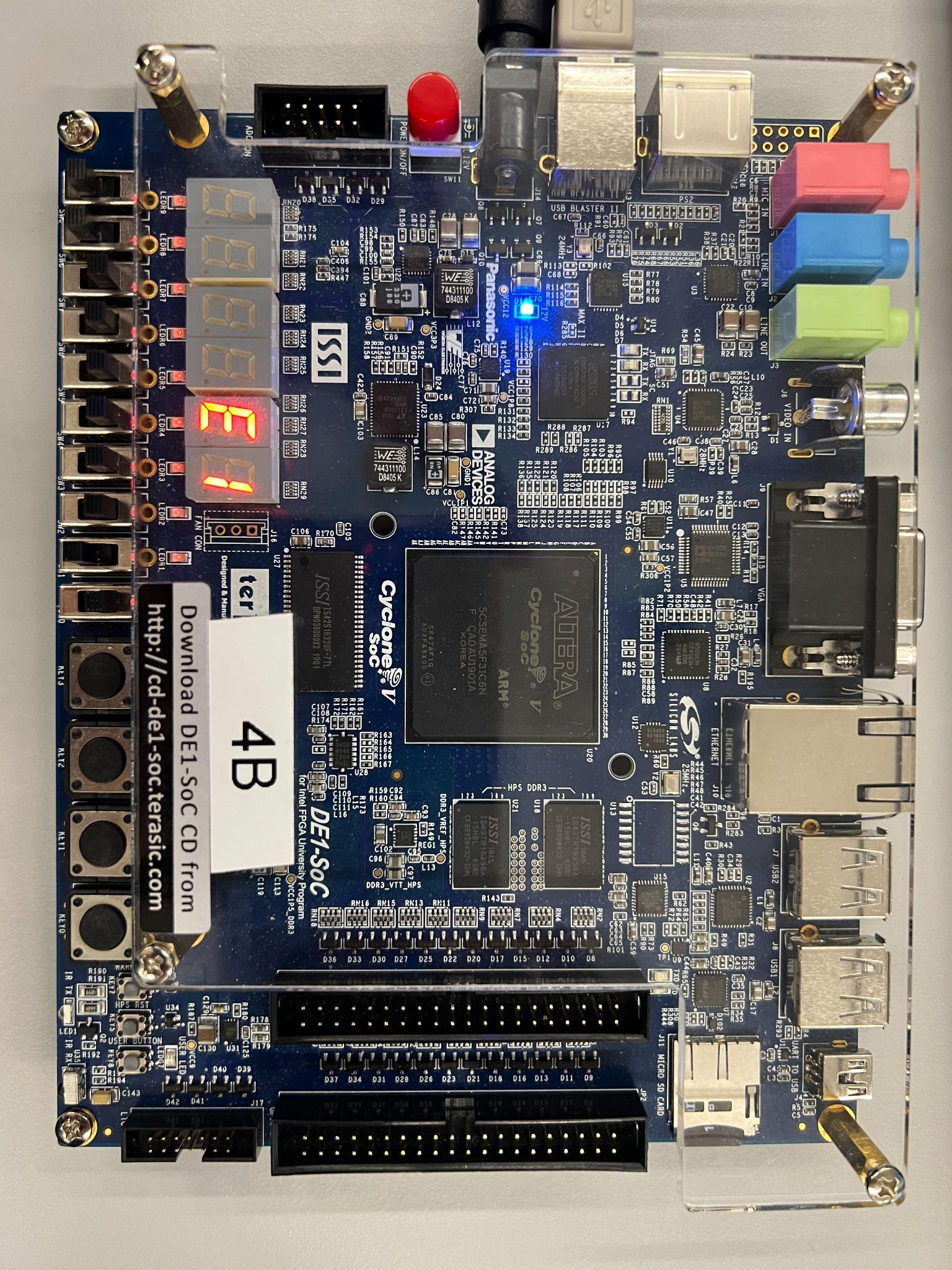
****

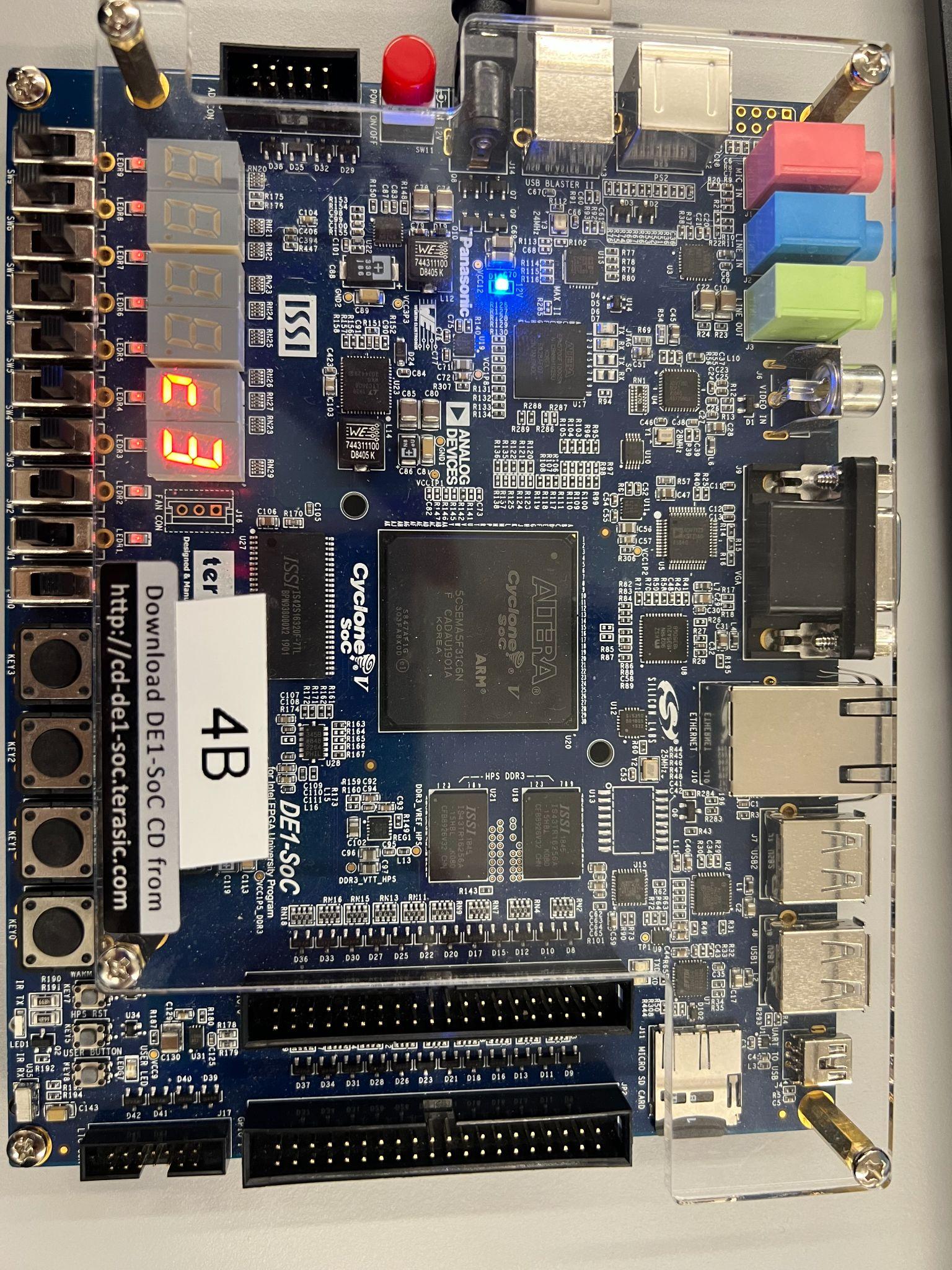
****

The pictures above is showing results for , , and respectively. As the pictures shown, the first

picture should have the result of 10010, which is 12 in hexadecimal form; the second picture should show the result 1000010, which is 42 in hexadecimal form; the third picture should show the result 11100001, which is E1 in hexadecimal form. From all pictures shown above, they are showing correct results in hexadecimal form.

The pictures below are showing the functionality of D Flip-Flops. From the schematic in section 3.3, the D Flip-Flops are placed right from the input to the AND gates, which means whenever the button is pressed, the clock will rise to the high edge and will trigger the result to be changed, otherwise the result will always remain unchanged. As shown in the two pictures below, when the input changes from , the result still remains E1, after the button is pressed, the result changes to C3, which is the correct result for .

****

****

# References

1. Prof. Julius Marpaung, “*Lab Report Guide*”, Northeastern University, January 6 2020.
2. Prof. Julius Marpaung, “*Lecture 7: Timing of Combinational Logicand Arithmetic Circuit* ”, Northeastern University, May 6 2022.
3. *“Full adder in digital logic”,* GeeksForGeeks, November 25, 2019, https://www.geeksforgeeks.org/full-adder-in-digital-logic/