ECEN 429: Introduction to Digital Systems Design Laboratory

North Carolina Agricultural and Technical State University

Department of Electrical and Computer Engineering

Ian Parker (Reporter)

Tayanna Lee (Lab Partner)

January 31, 2019

Lab #2

Introduction:

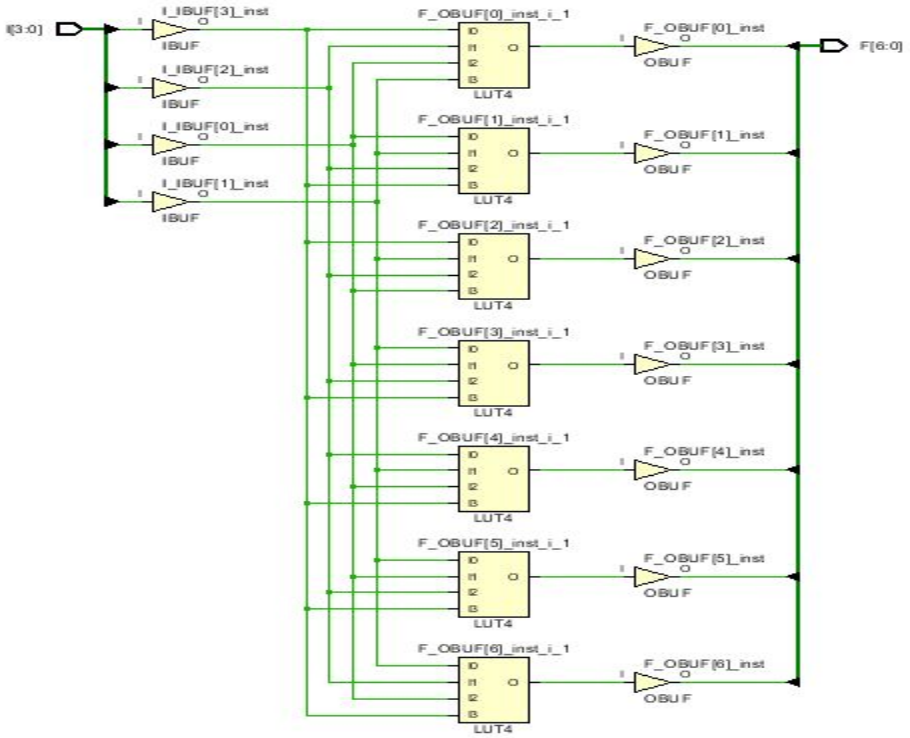
For this lab, we demonstrated becoming more knowledgeable of the Basys 3 board by learning how to operate a 7-segment display decoder, and the purpose of encoders and full adders. The BCD to 7 Segment Decoder converts 4-bit binary to a 7-bit control signal which then displays 7 LED segments to display 0 to 9 and A to F. A Binary encoder has 2n input lines and n-bit output lines. Last, but not least, we are introduced to full adders, which compute the sum of 3 inputs and a carry value.

Part 1: BCD to 7-Segment Display

Truth Table

Inputs	A-B-C-D-E-F-G
0000	1000000
0001	1001111
0010	0010010
0011	0000110
0100	0001101
0101	0100100
0110	0100000
0111	1001110
1000	0000000
1001	0000100

Schematic



Pin Assignment

Output F	Pin
F[6]	U7
F[5]	W6
F[4]	U8
F[3]	V8
F[2]	U5
F[1]	V5
F[0]	W7

Result

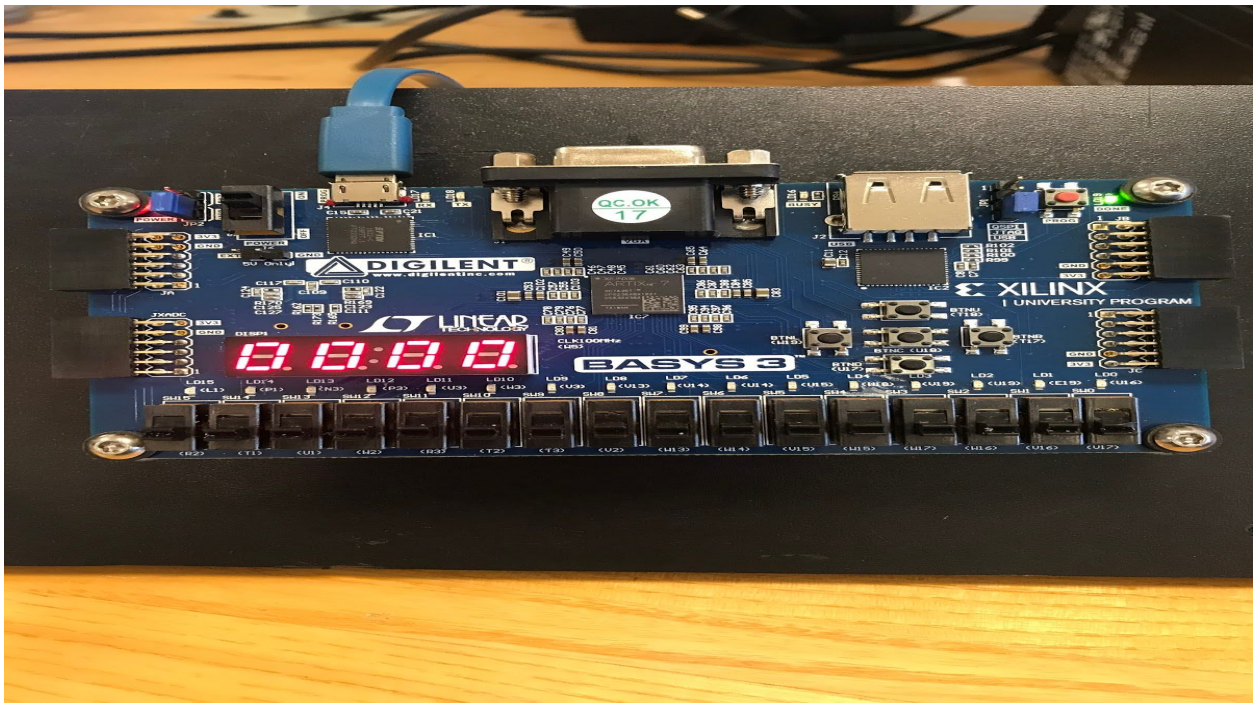


Figure 1.1) 0

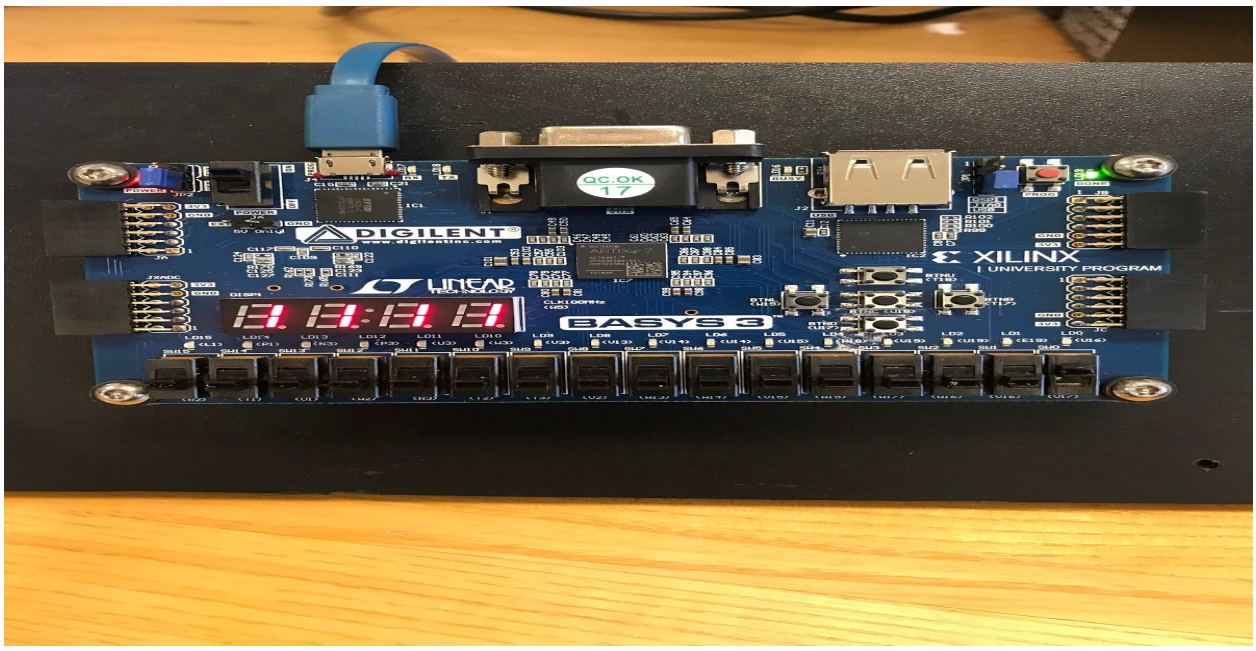


Figure 1.2)1

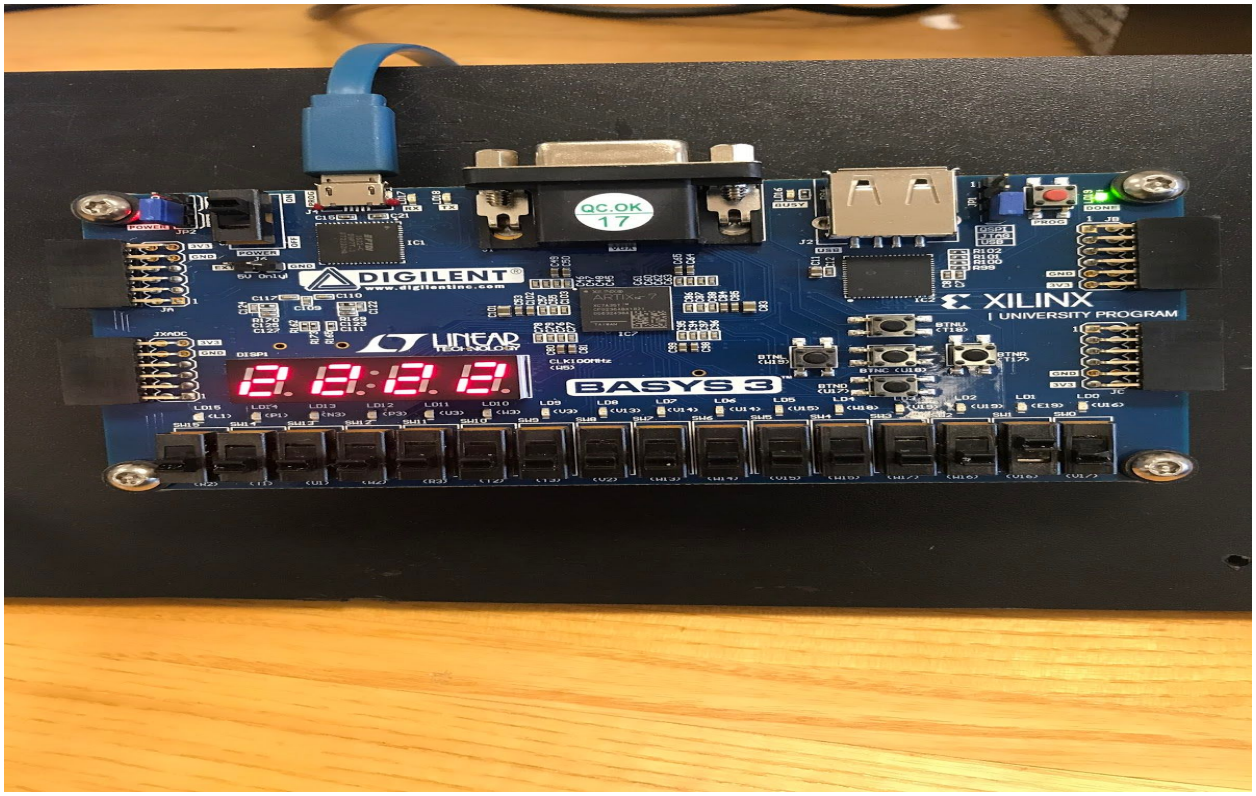


Figure 1.3)2

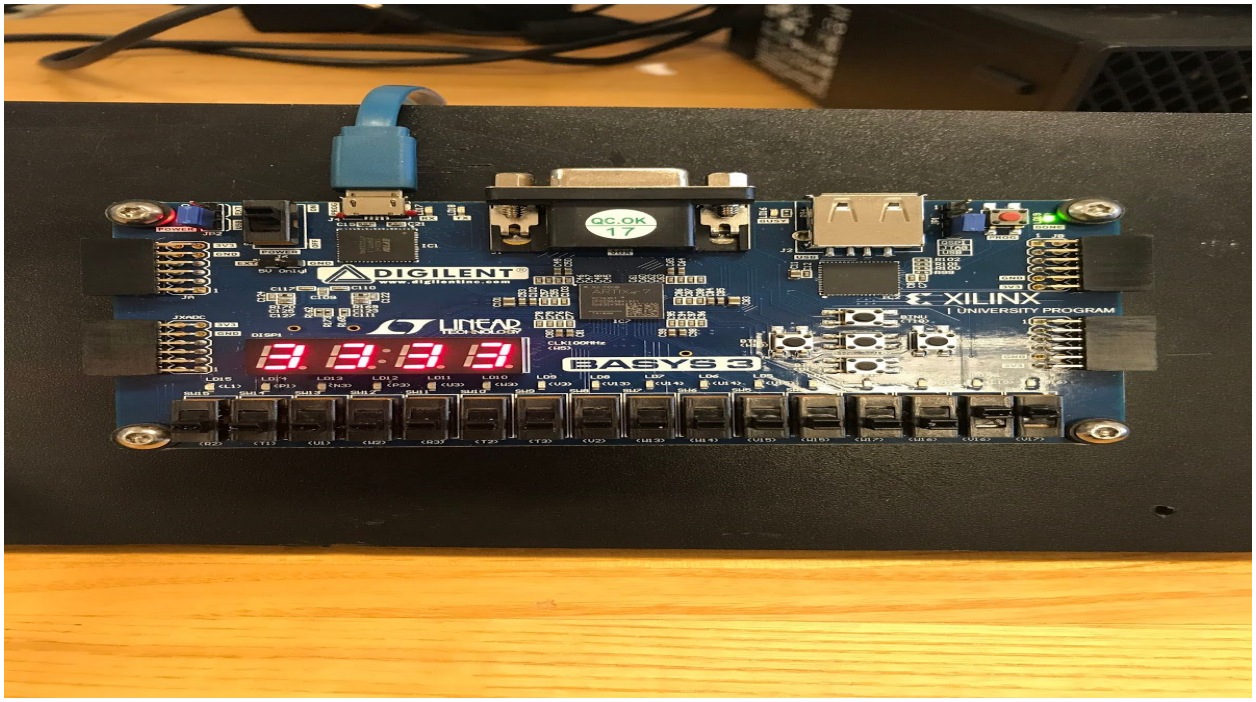


Figure 1.4)3

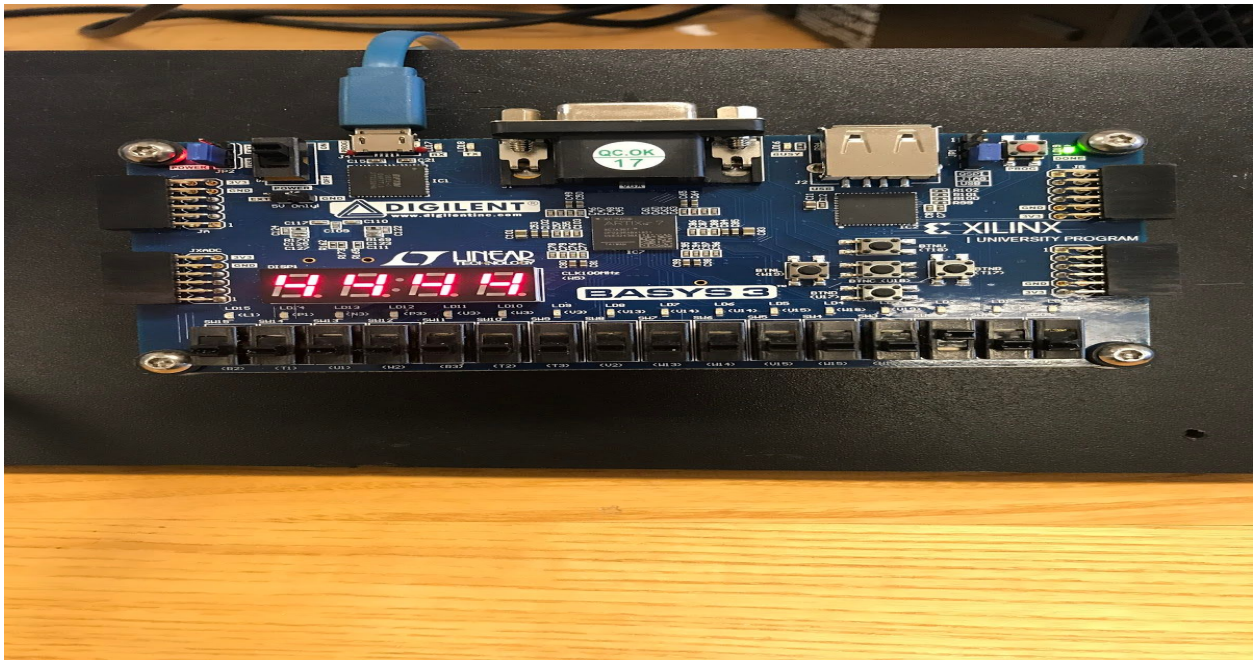


Figure 1.5)4

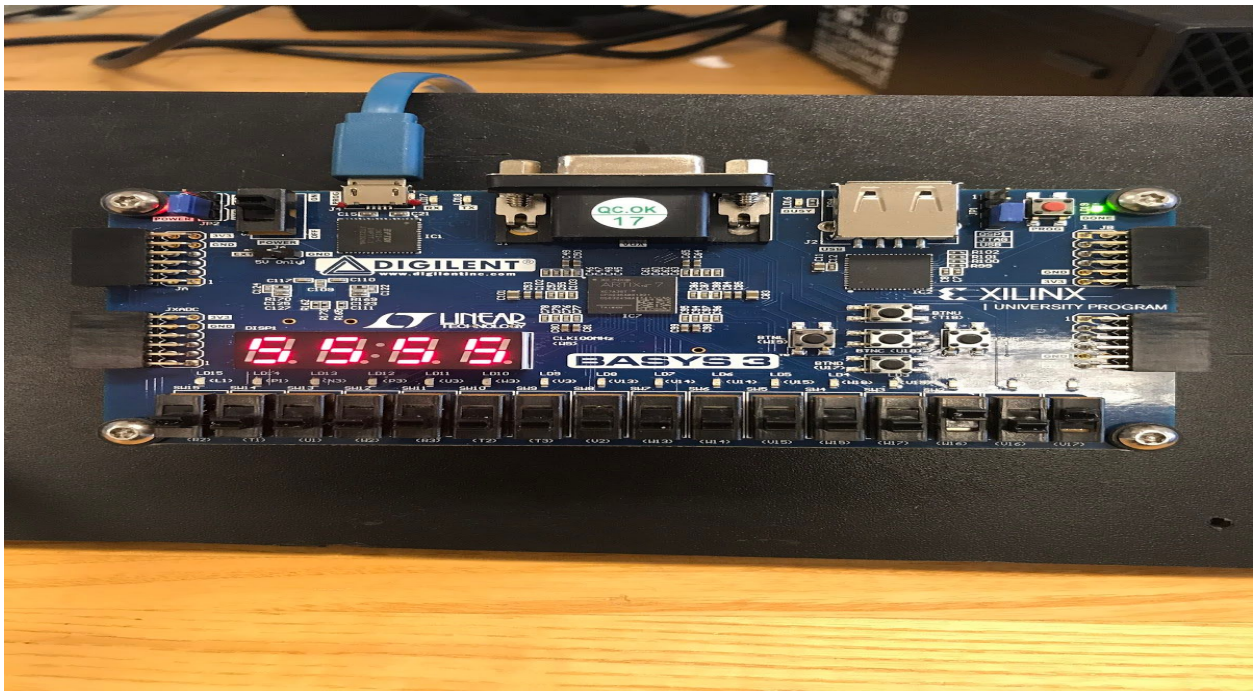


Figure 1.6)5

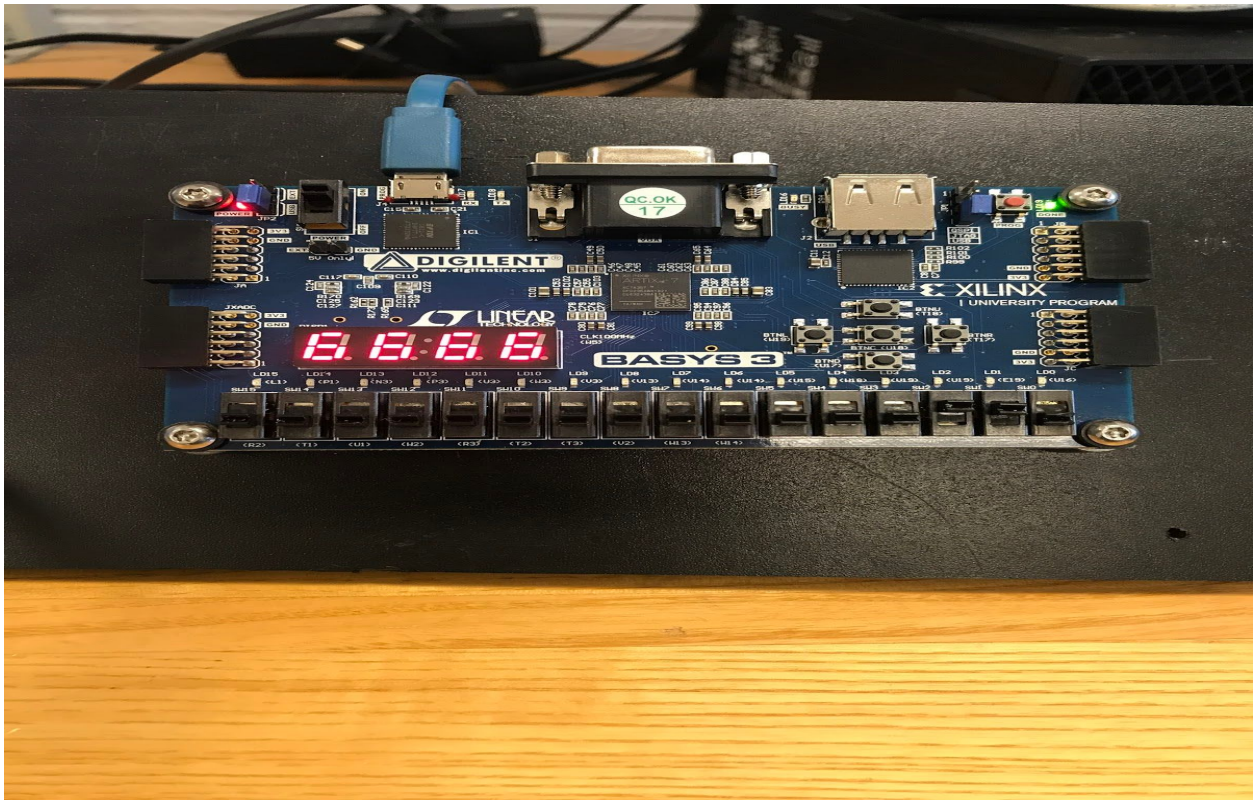


Figure 1.7)6

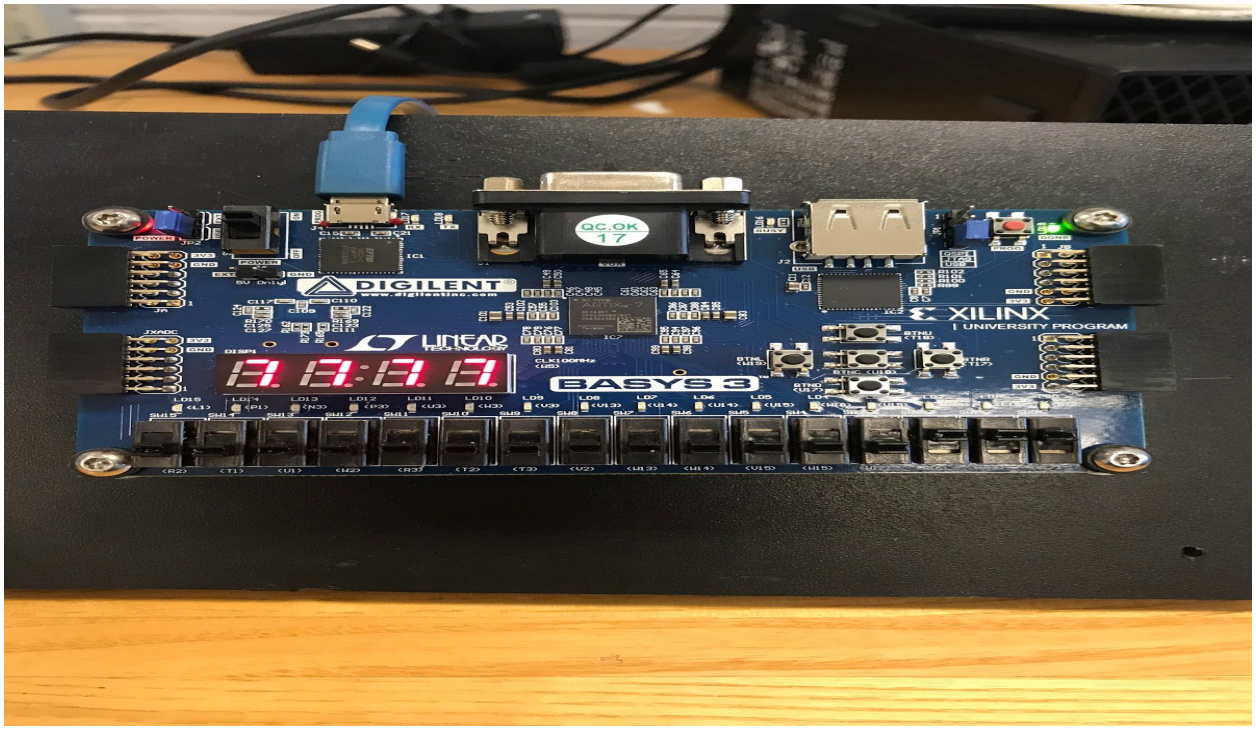


Figure 1.8)7

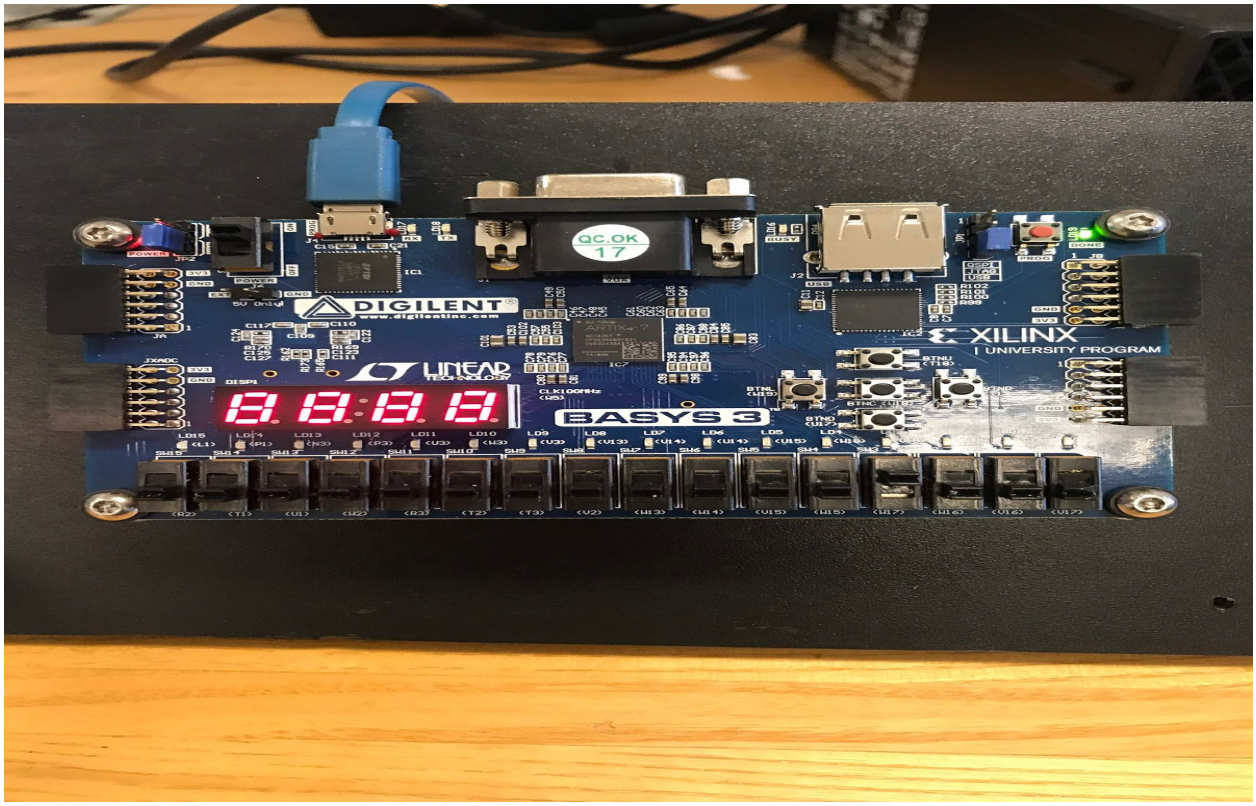


Figure 1.9)8

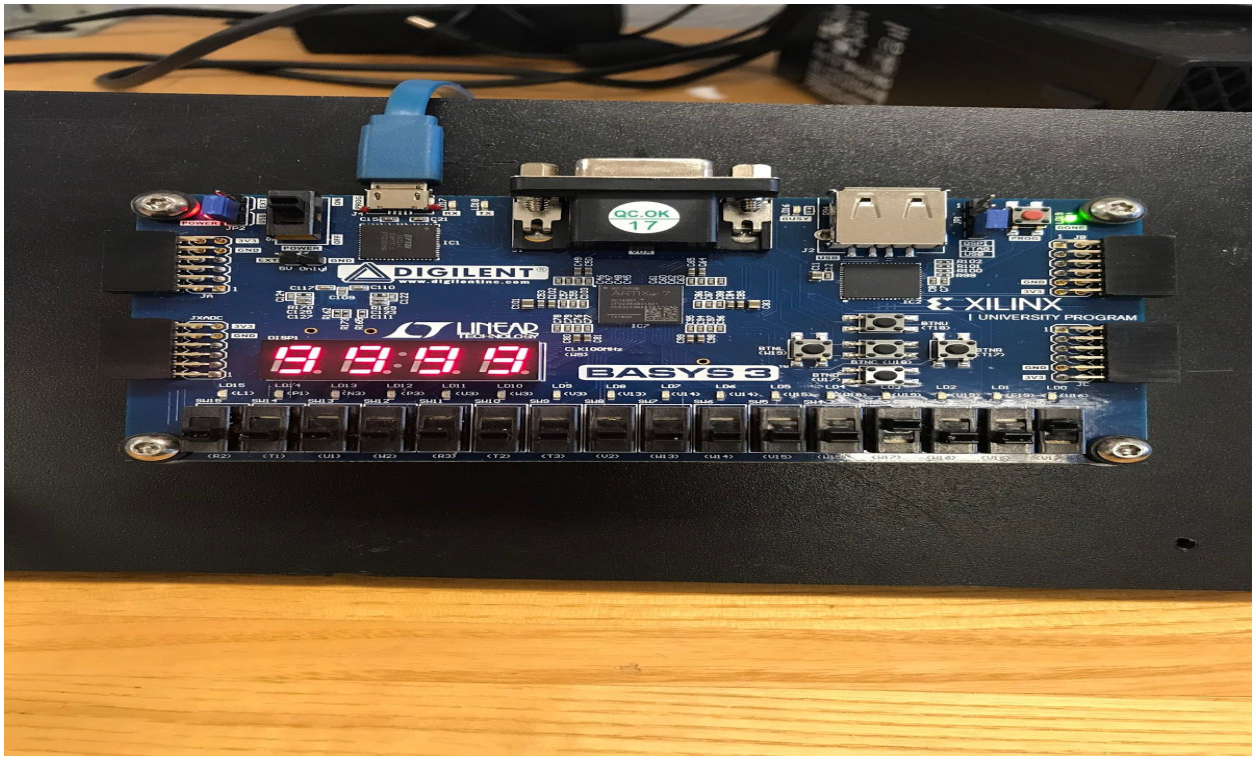


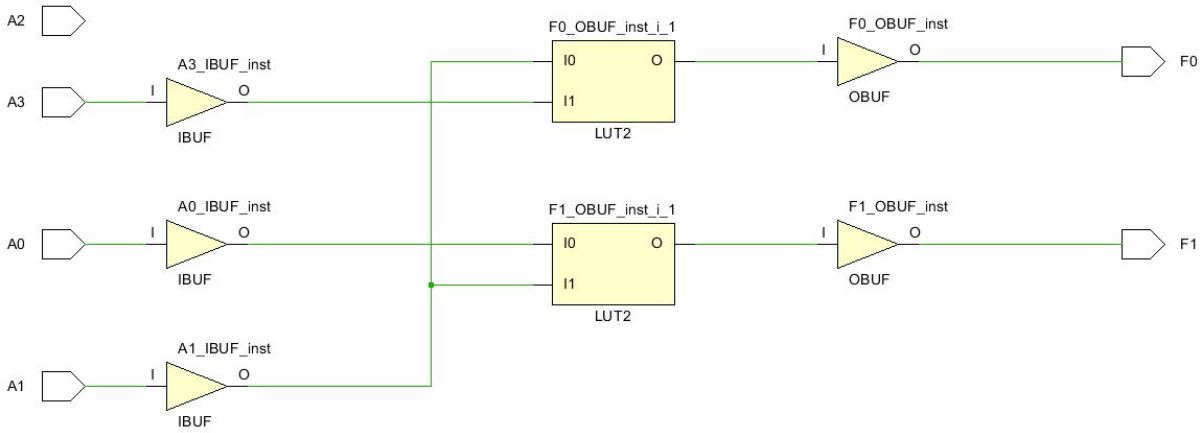
Figure 1.10)9

Part 2: 4:2 Encoder

Truth Table

Input (A3, A2, A1, A0)	Output F0	Output F1
0000	0	1
0001	0	0
0010	1	0
0011	1	0
0100	0	1
0101	0	0
0110	1	0
0111	1	0
1000	1	1
1001	1	0
1010	1	0
1011	1	0
1100	1	1
1101	1	0
1110	1	0
1111	1	0

Schematic



FPGA Pins

Name	I/O	Pin
A3	IN	V17
A2	IN	V16
A1	IN	W16
A0	IN	W17
F0	OUT	E19
F1	OUT	U16

Results

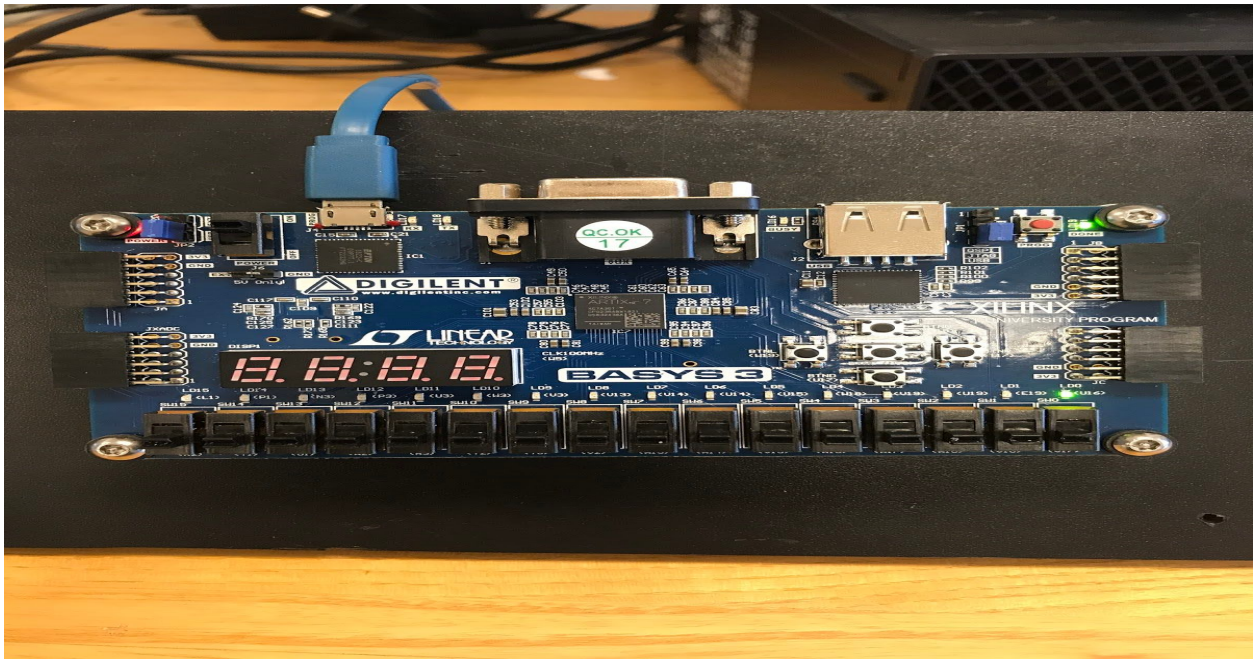


Figure 1.1) [0000] F0 =0 F1 =1

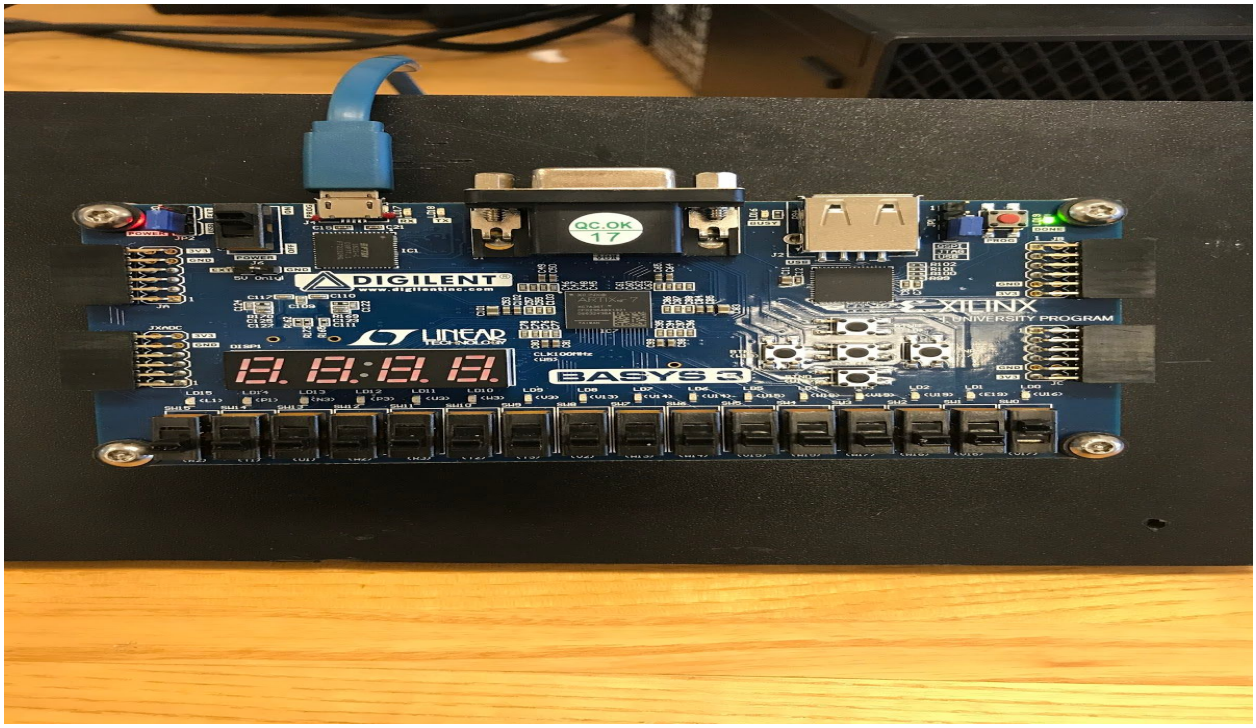


Figure 1.2) [0001] F0 =0 F1 =0

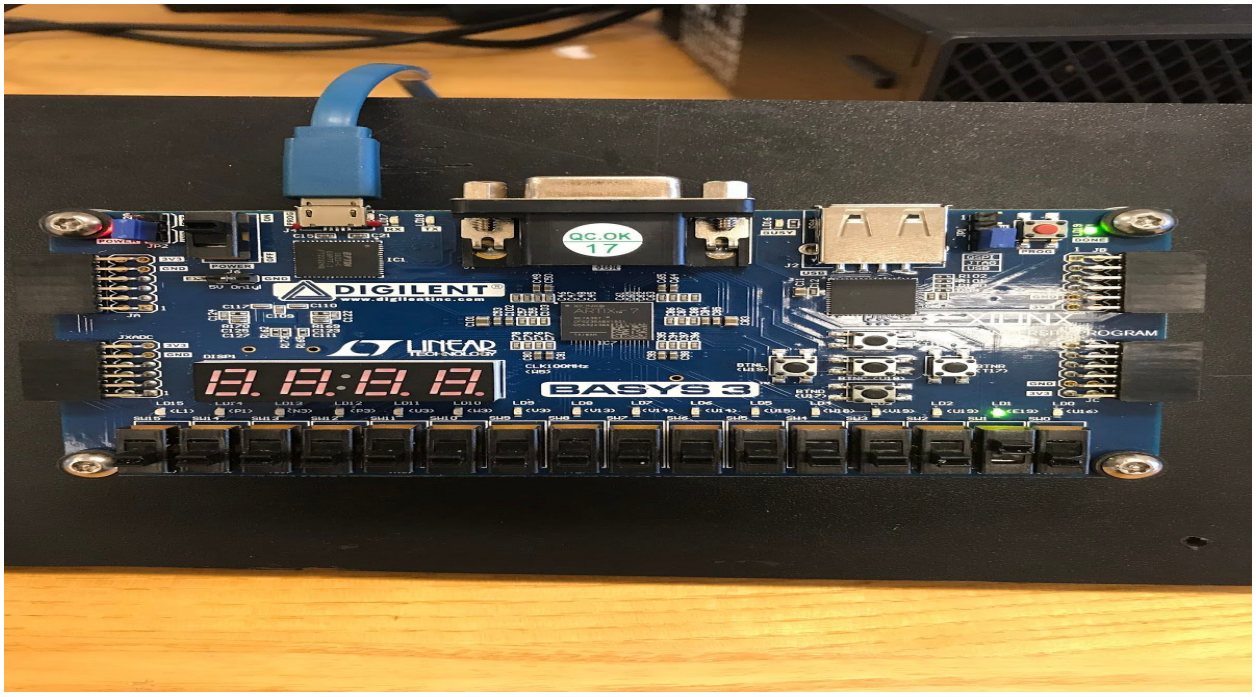


Figure 1.3) [0010] F0 =1 F1 =0

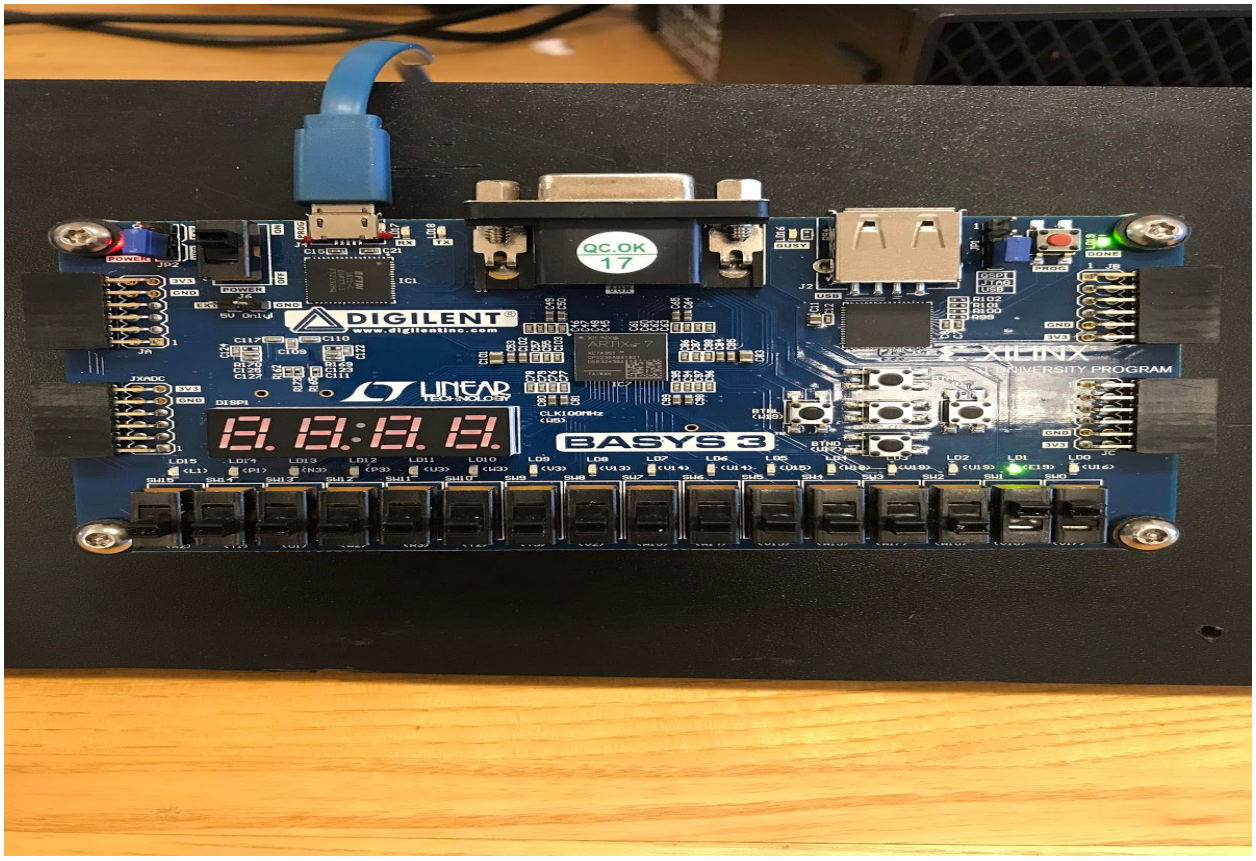


Figure 1.4) [0011] F0 =1 F1 =0

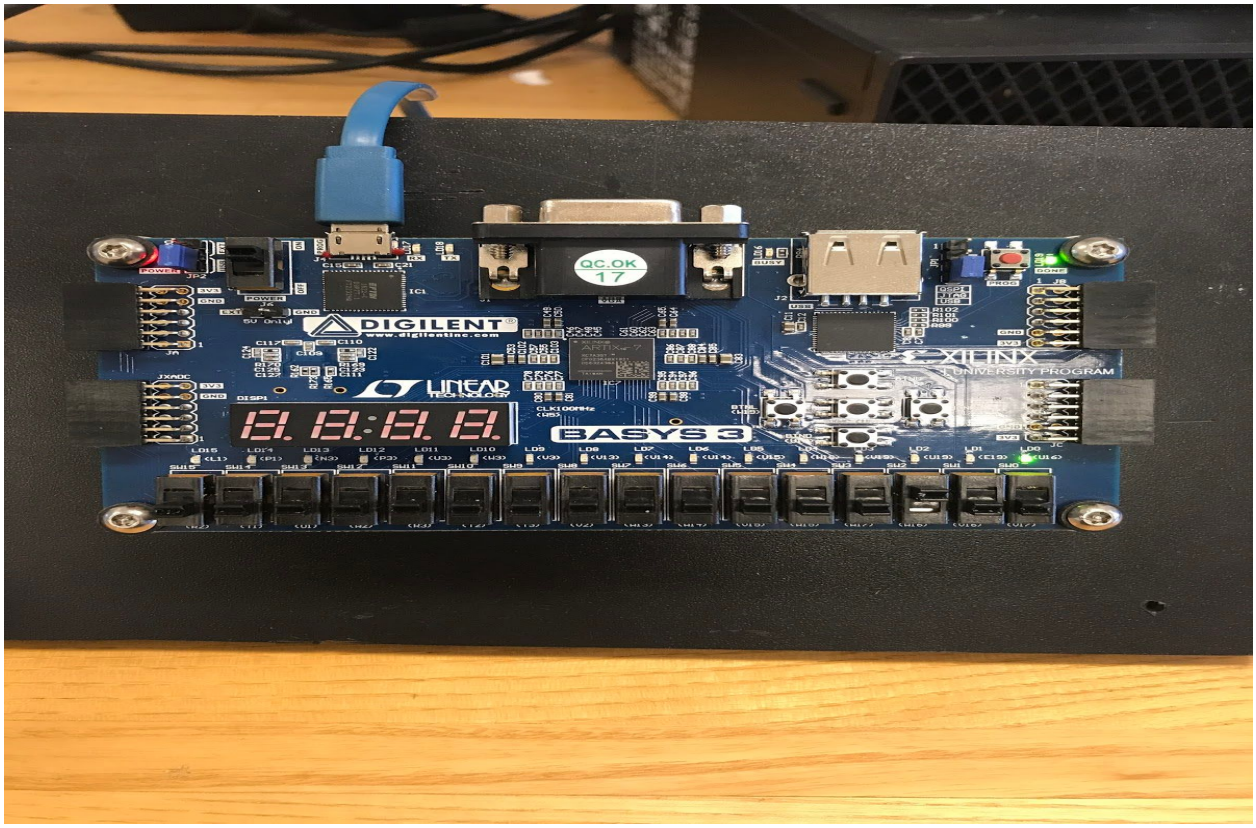


Figure 1.5) [0100] F0 =0 F1 =1

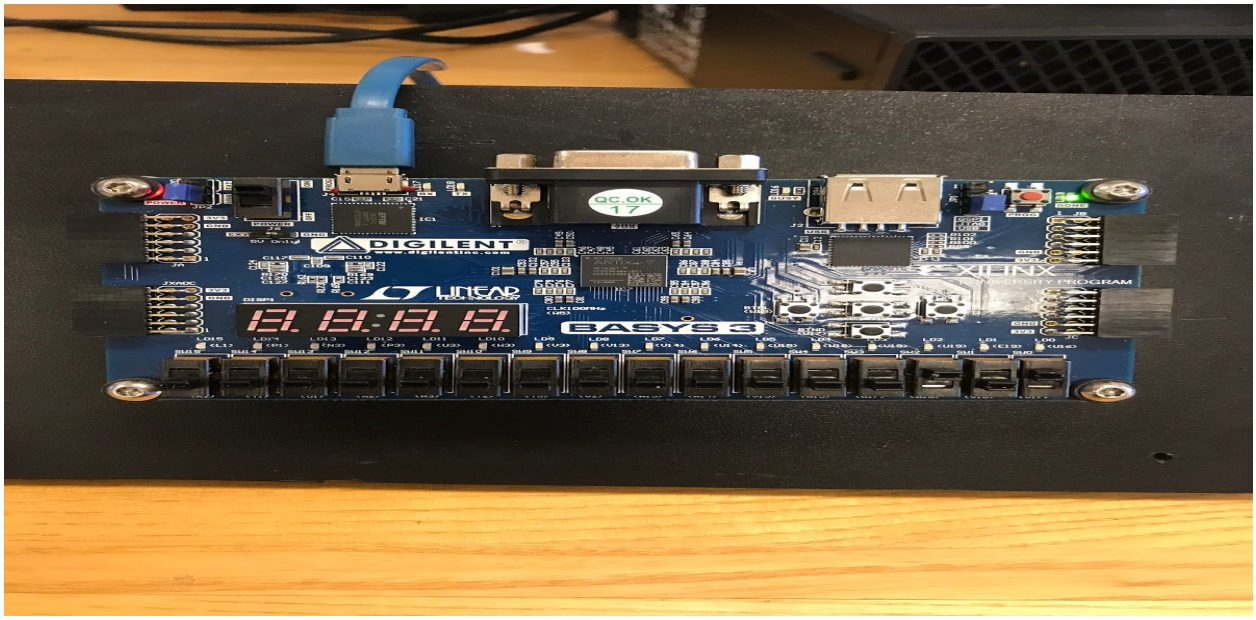


Figure 1.6) [0101] F0 =0 F1 =0

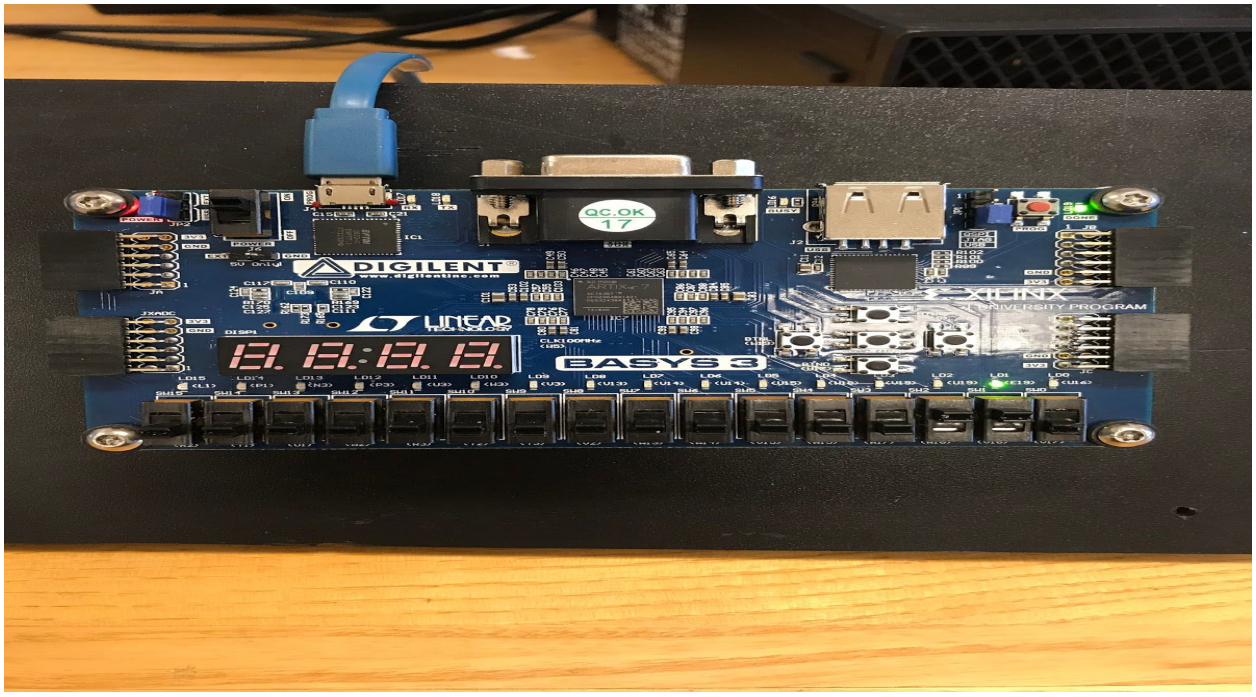


Figure 1.7) [0110] F0 =1 F1 =0

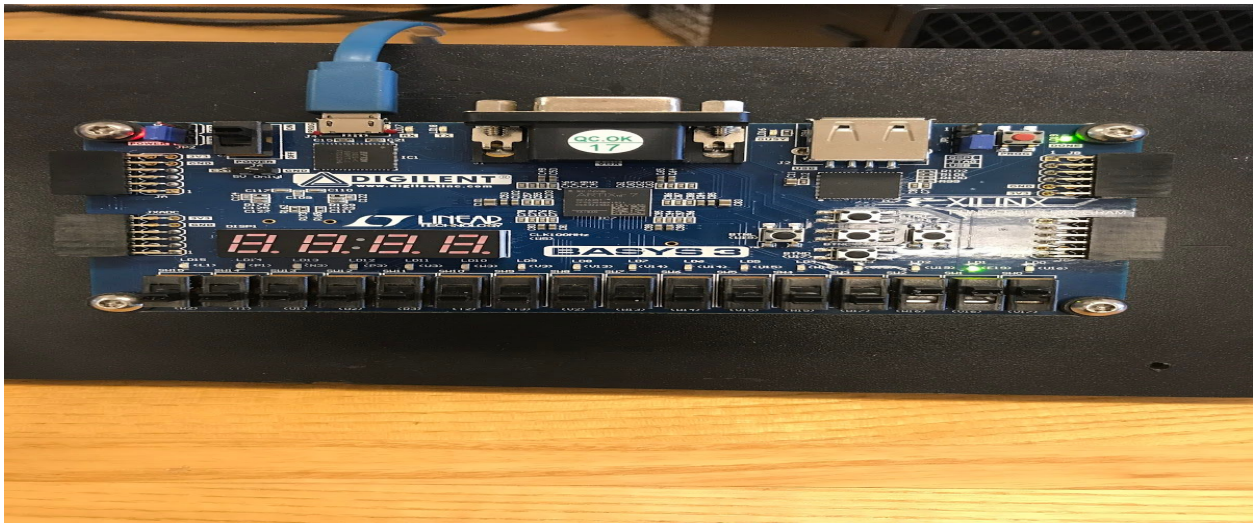


Figure 1.8) [0111] F0 =1 F1 =0

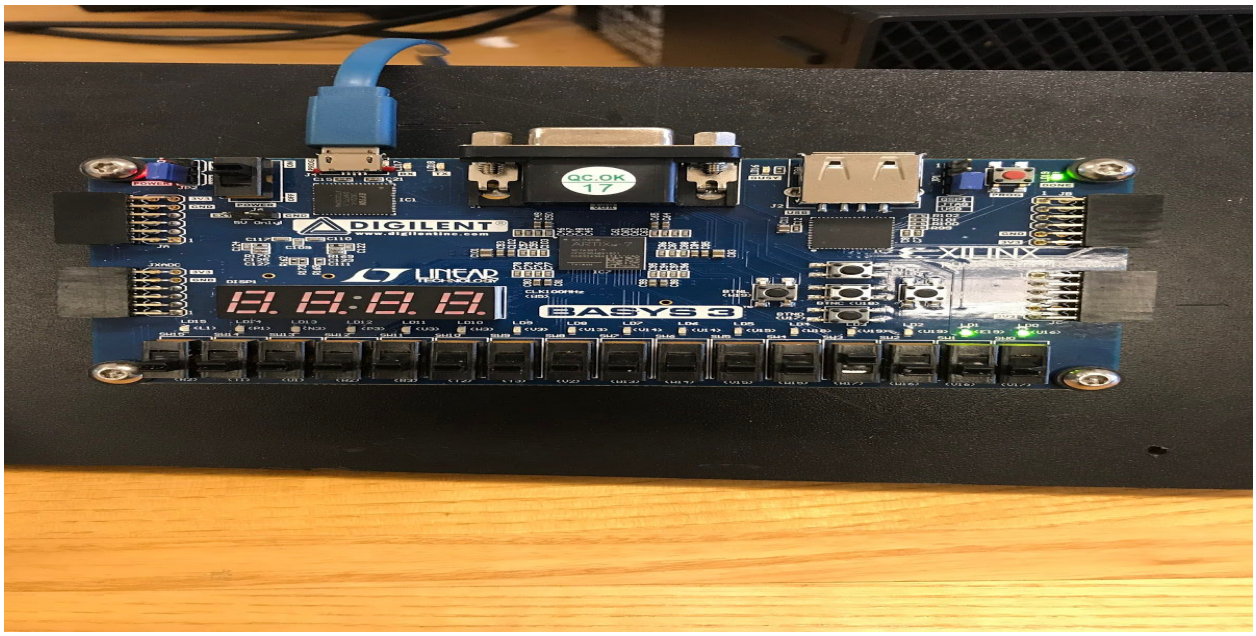


Figure 1.9) [1000] F0 =1 F1 =1

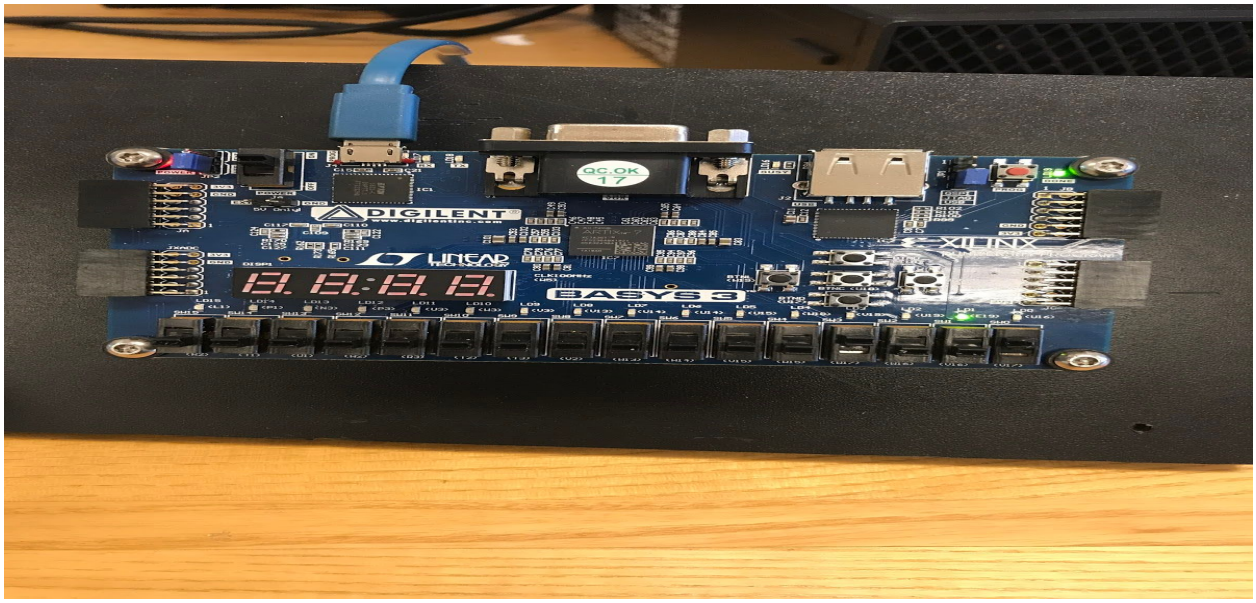


Figure 1.10) [1001] F0 =1 F1 =0

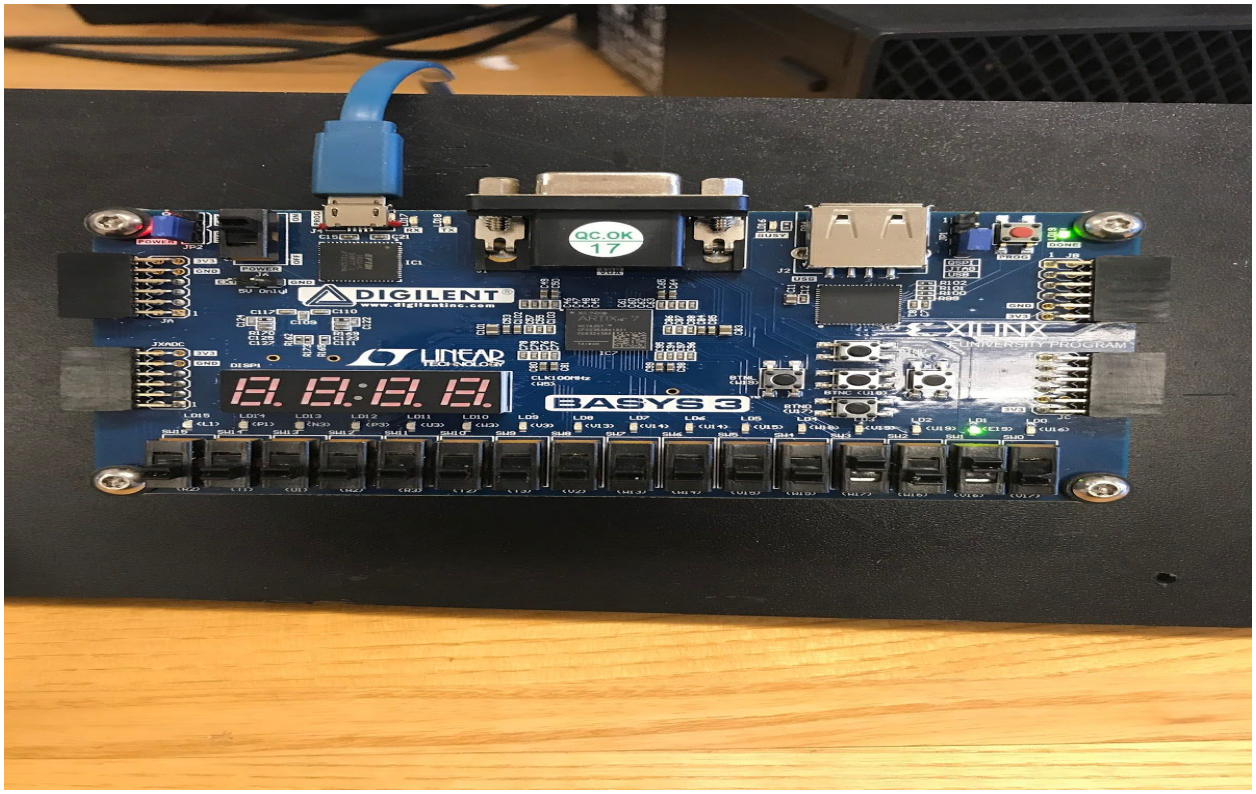


Figure 1.11) [1010] F0 =1 F1 =0

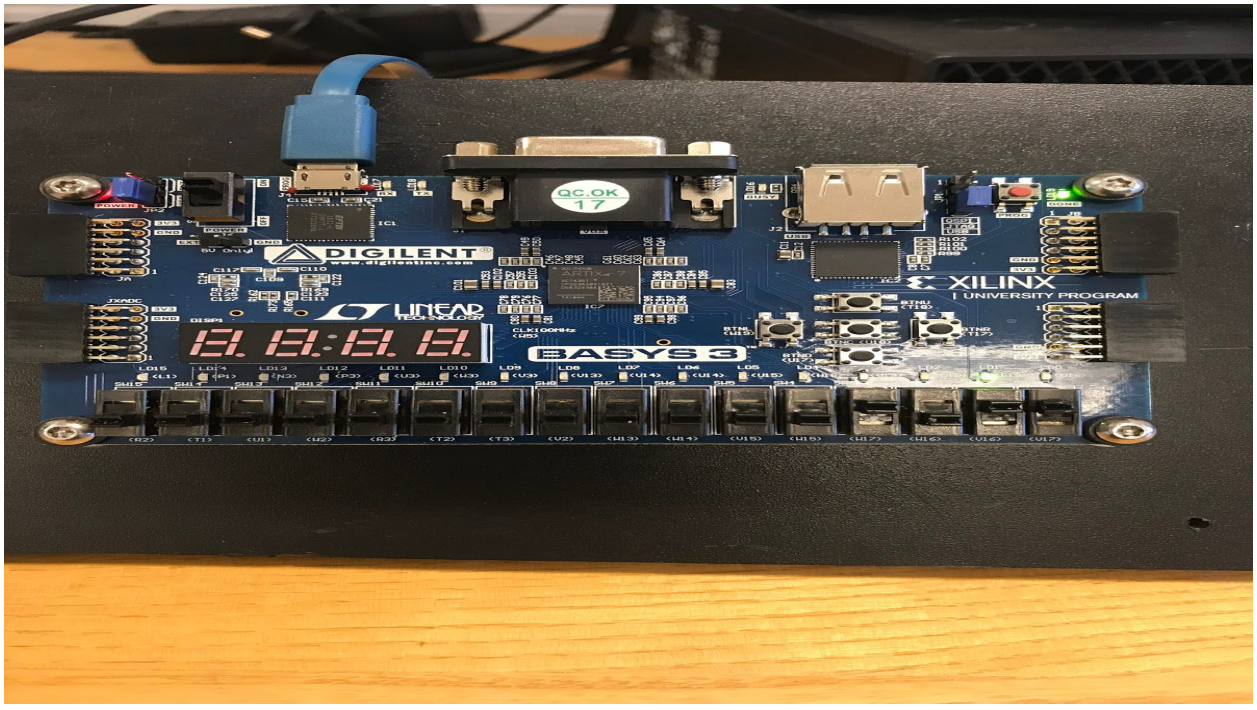


Figure 1.12) [1011] F0 =1 F1 =0

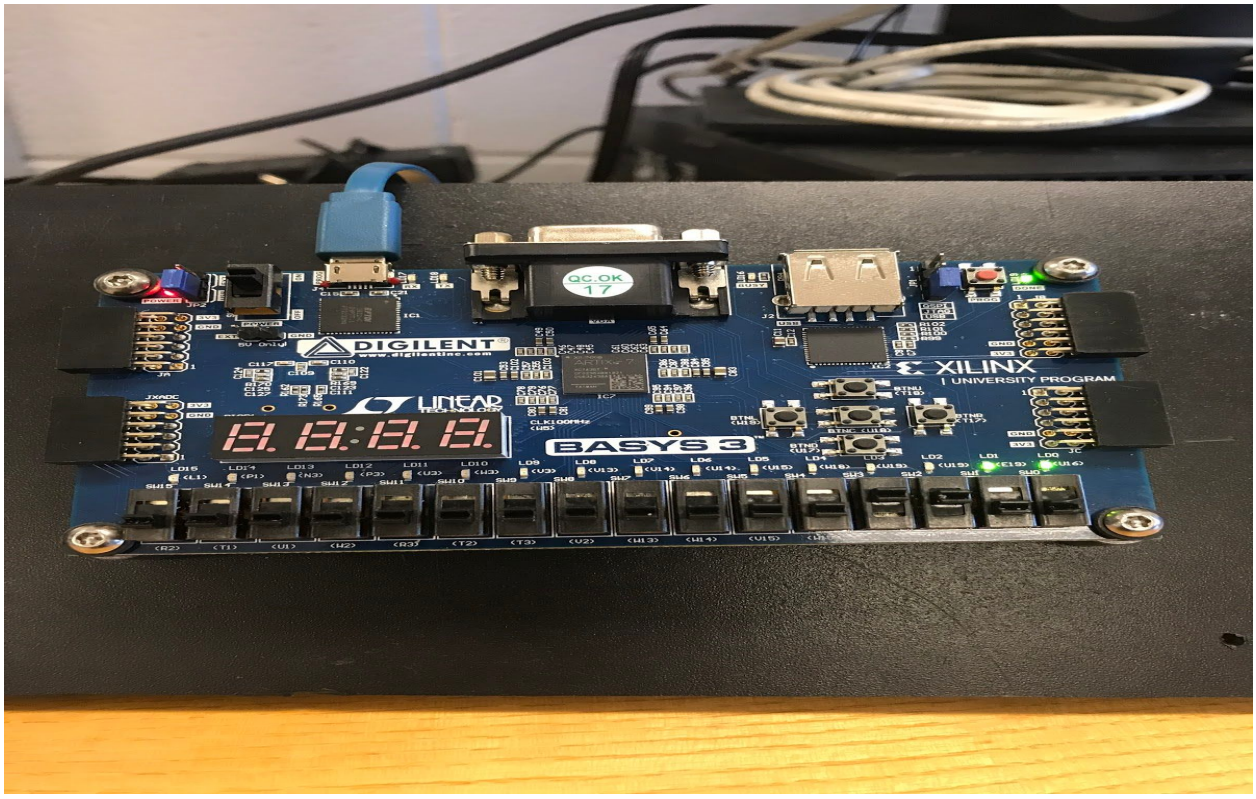


Figure 1.13) [1100] F0 =1 F1 =1

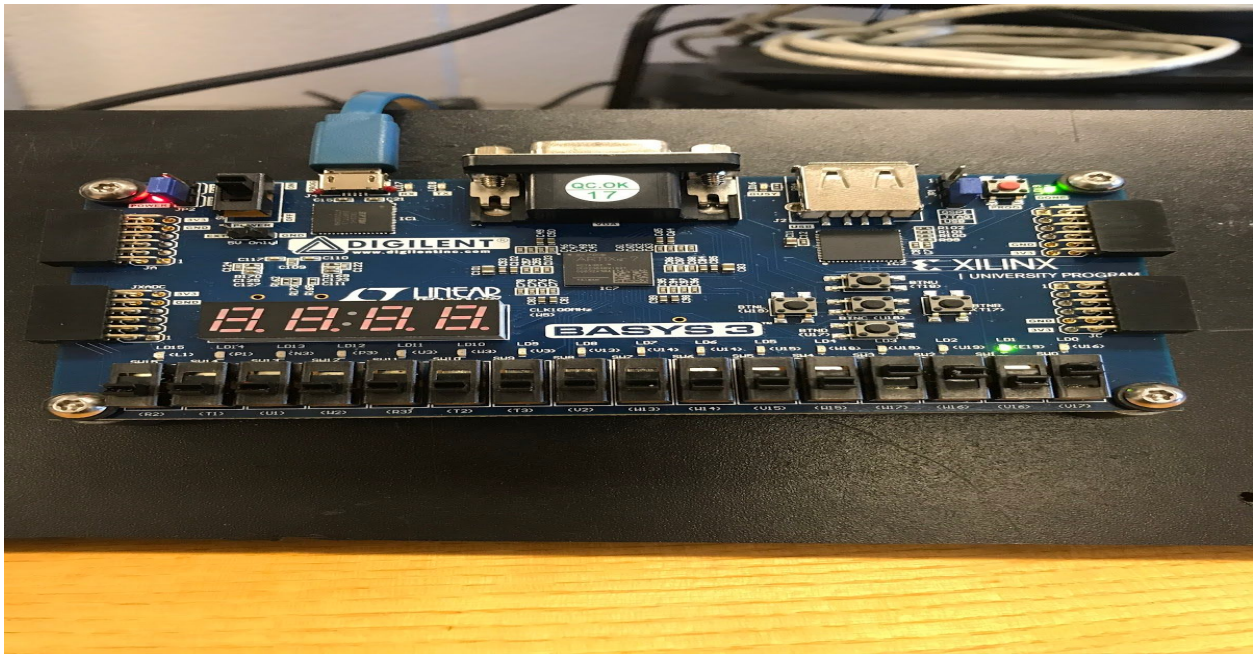


Figure 1.14) [1101] F0 =1 F1 =0

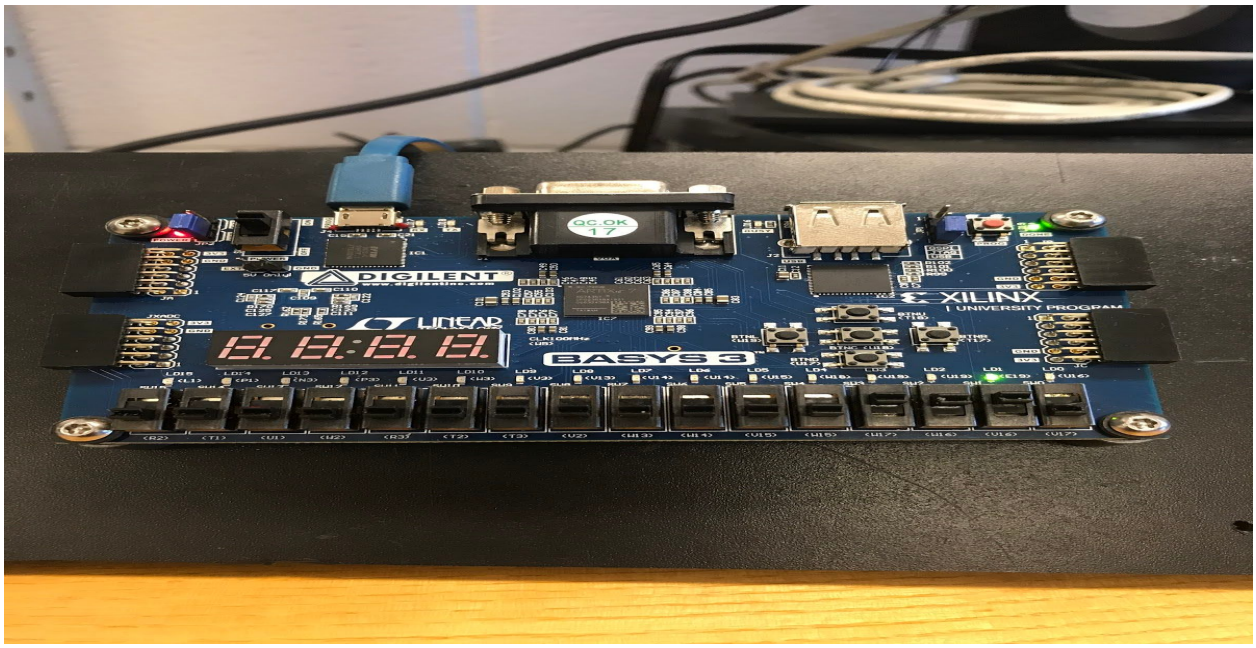


Figure 1.15) [1110] F0 =1 F1 =0

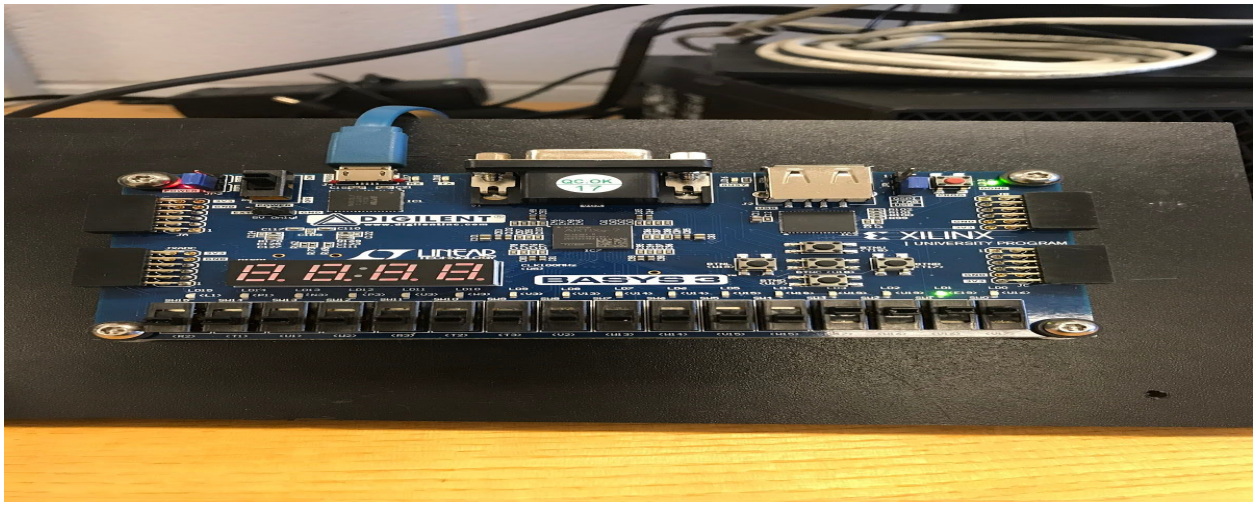


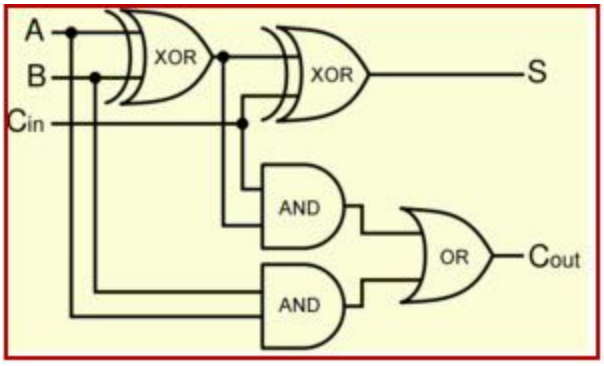
Figure 1.16) [1111] F0 = 1 F1 =0

Part 3: Full Adder

Truth Table

Inputs			Outputs	
Cin	B	A	Sum	Co
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Schematic



FPG Pin Assignments

I/O	Pin
A	W16
B	V16
Cin	V17
Sum	E19
Co	U16

Results

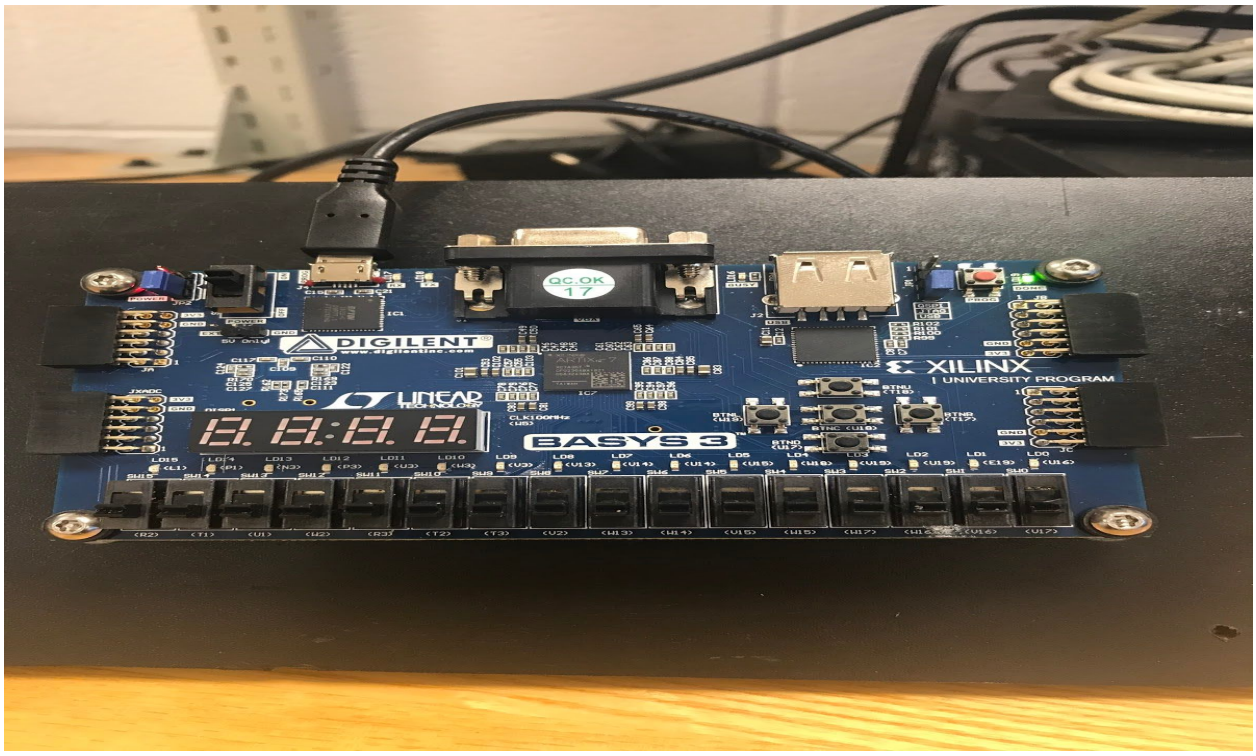


Figure 3.0) Cin, B, A = [000] ; Sum, Co = [00] (the rest of the figures follow this format)

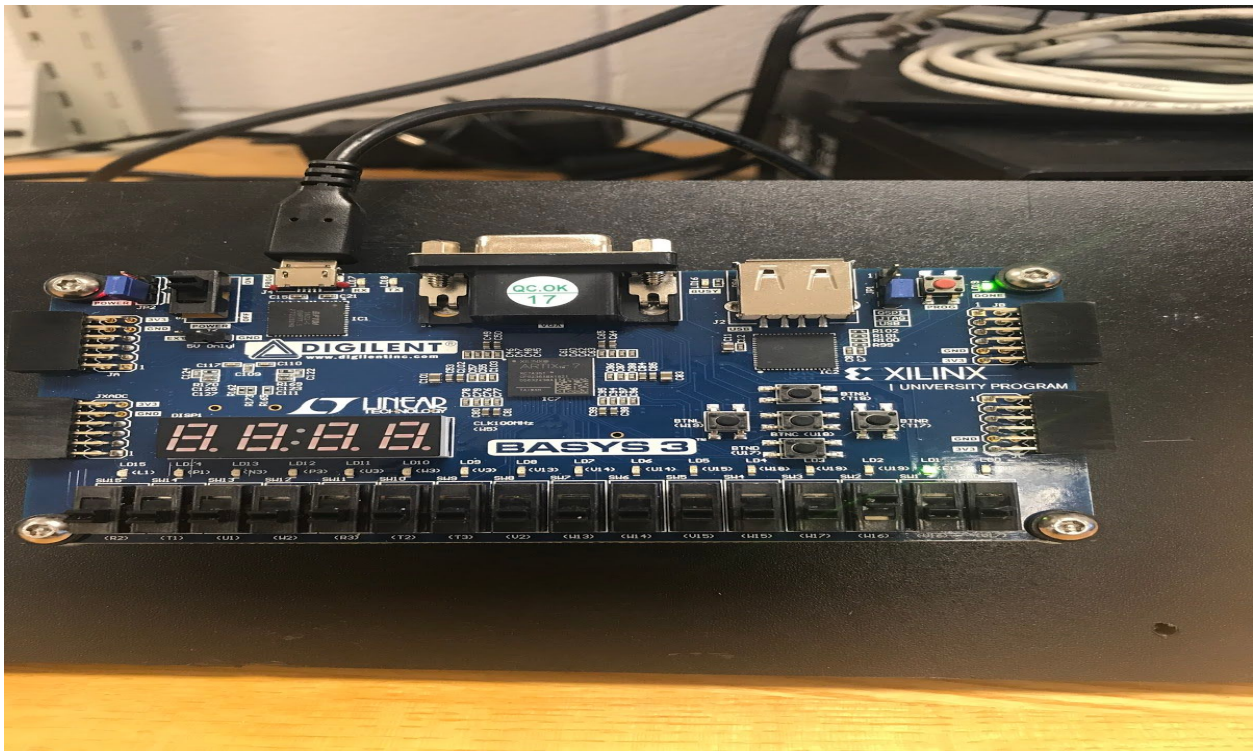


Figure 3.1)[001];[10]

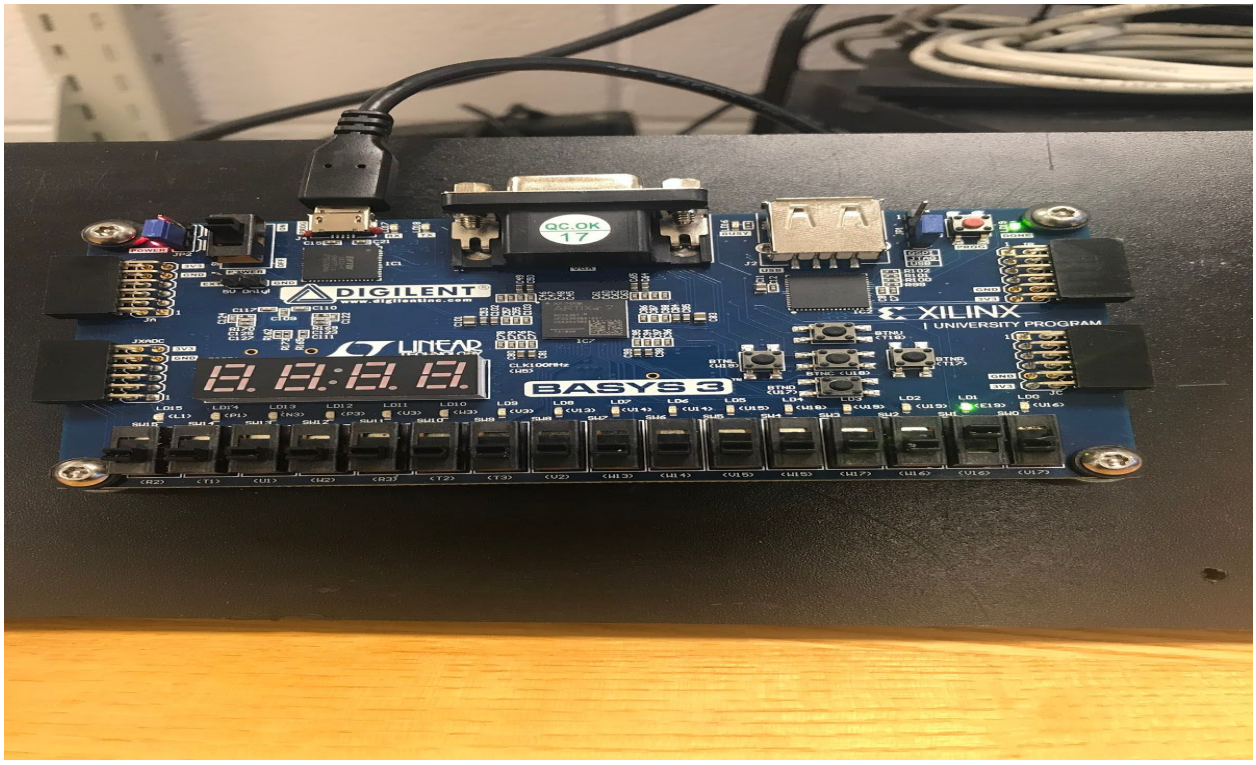


Figure 3.2)[010];[10]

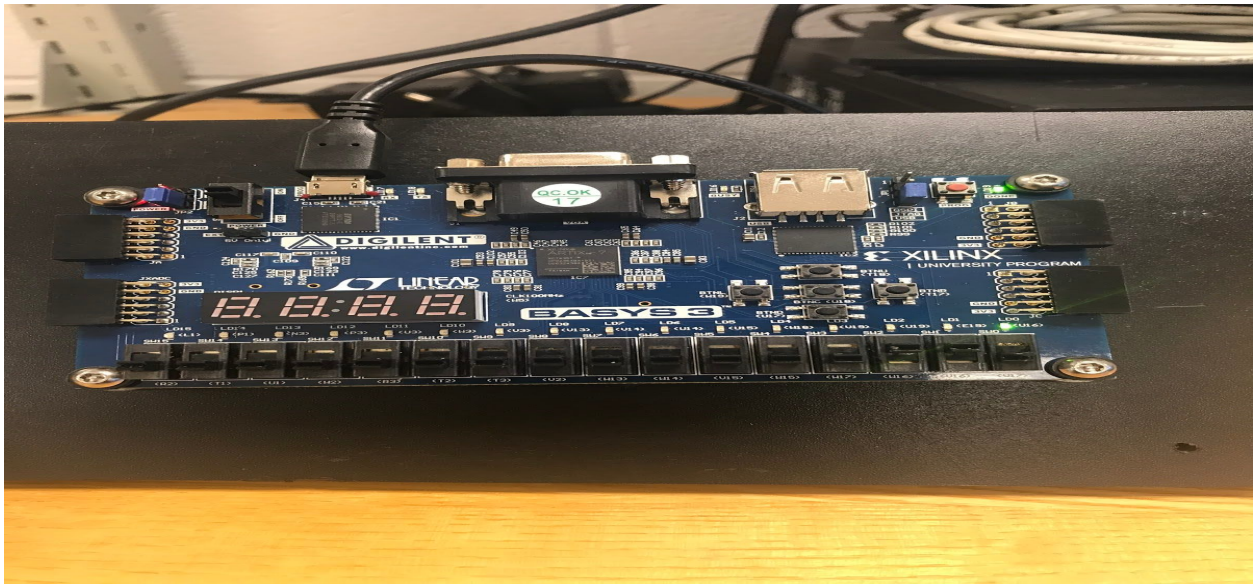


Figure 3.3)[011];[01]

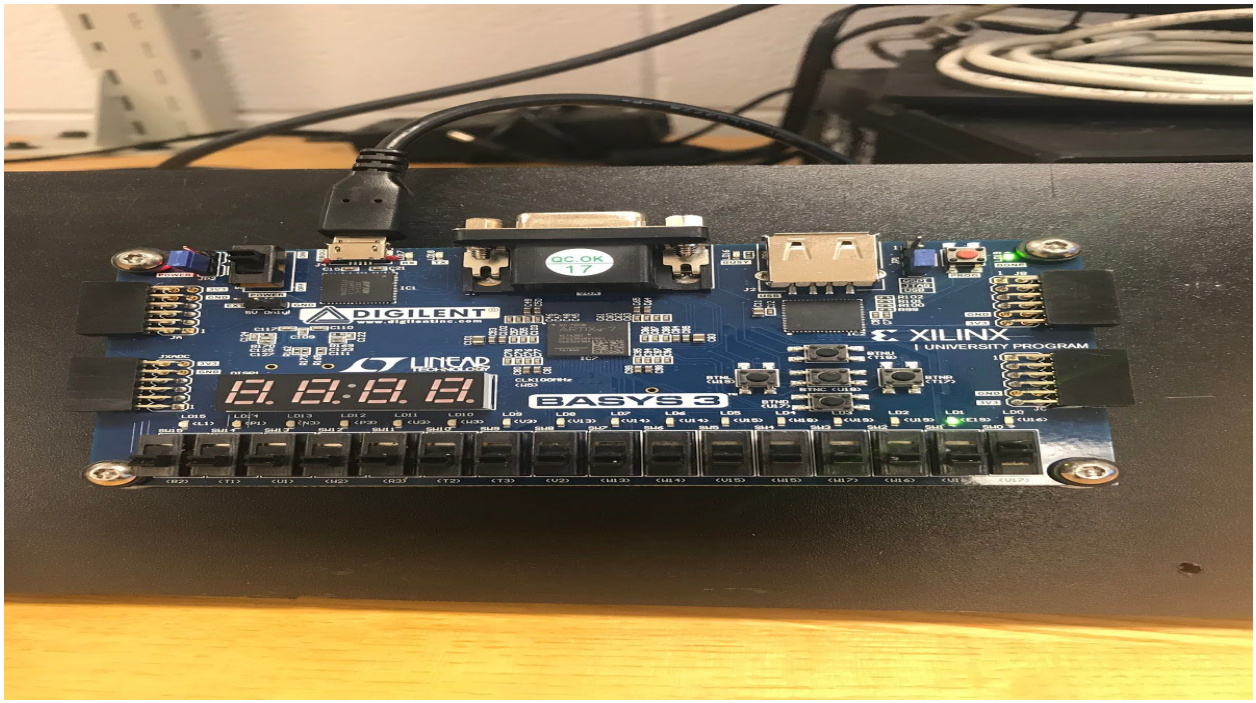


Figure 3.4)[100];[10]

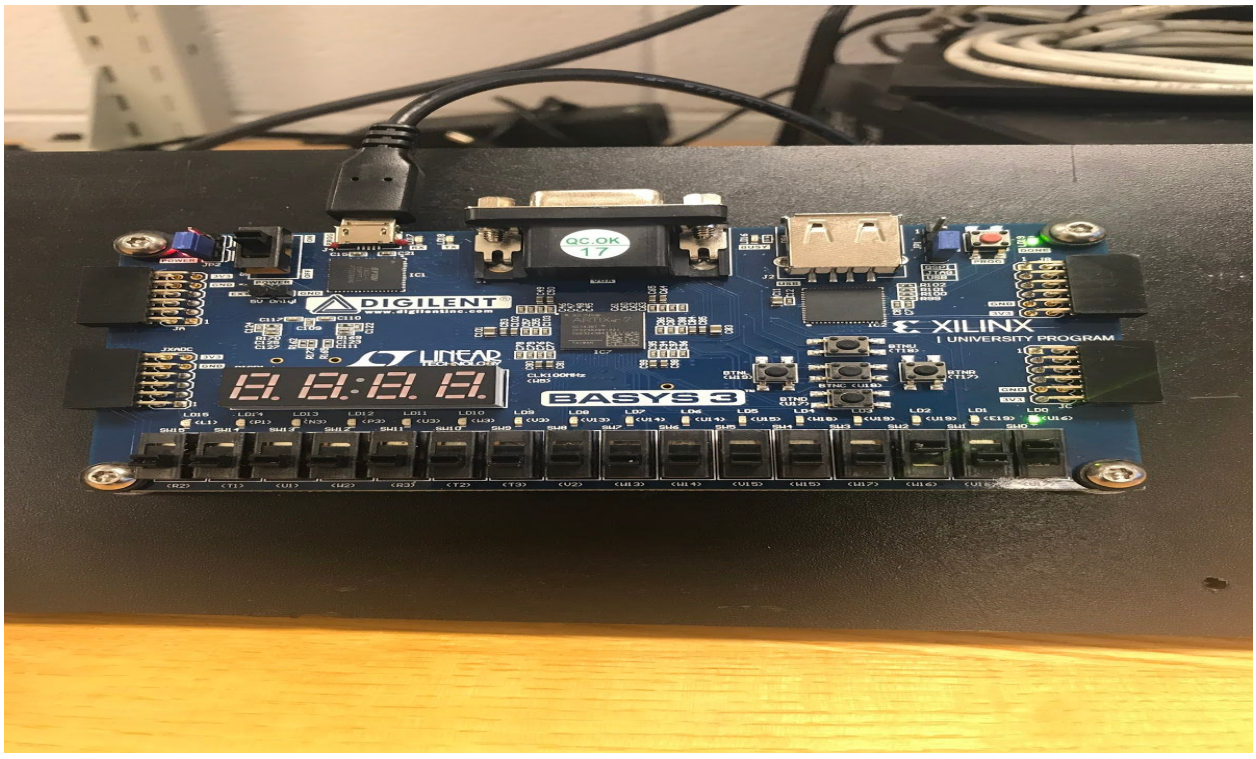


Figure 3.5)[101];[01]

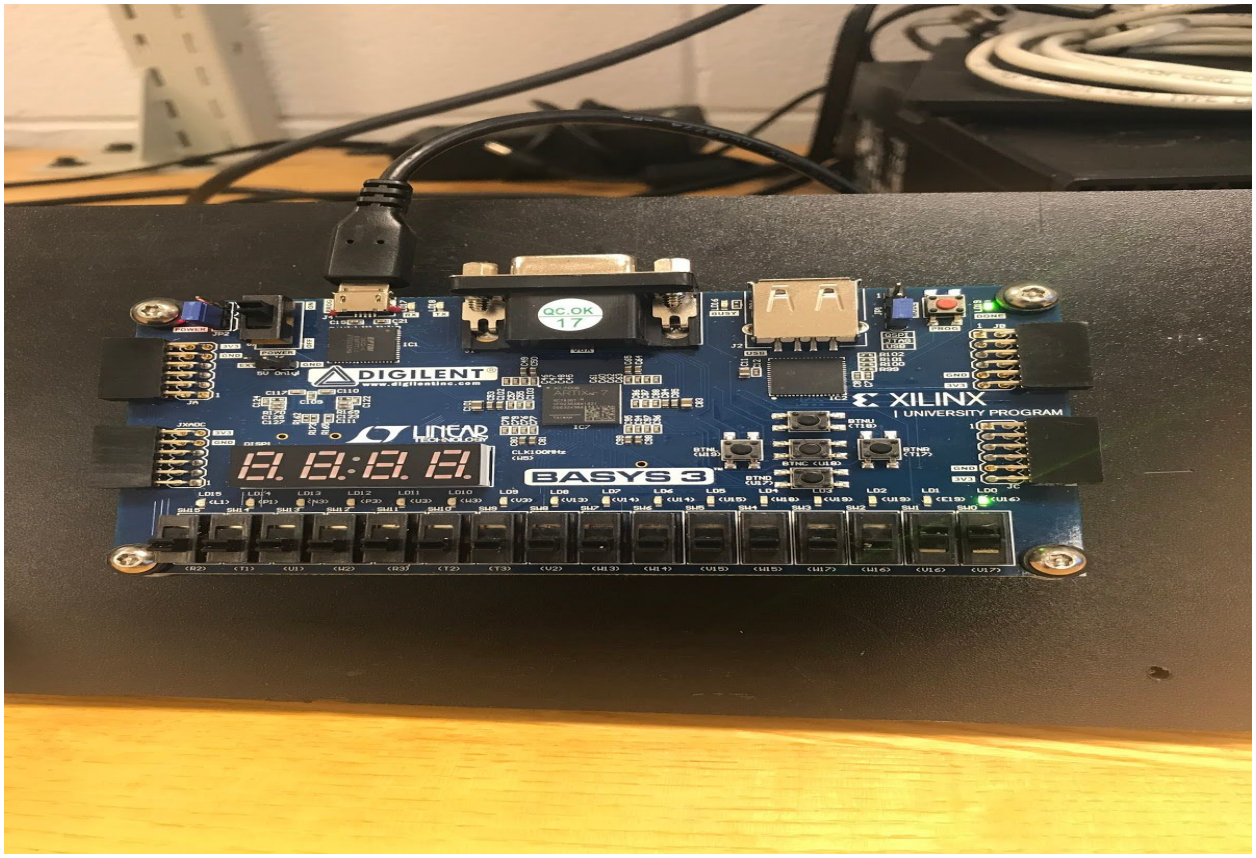


Figure 3.6)[110];[01]

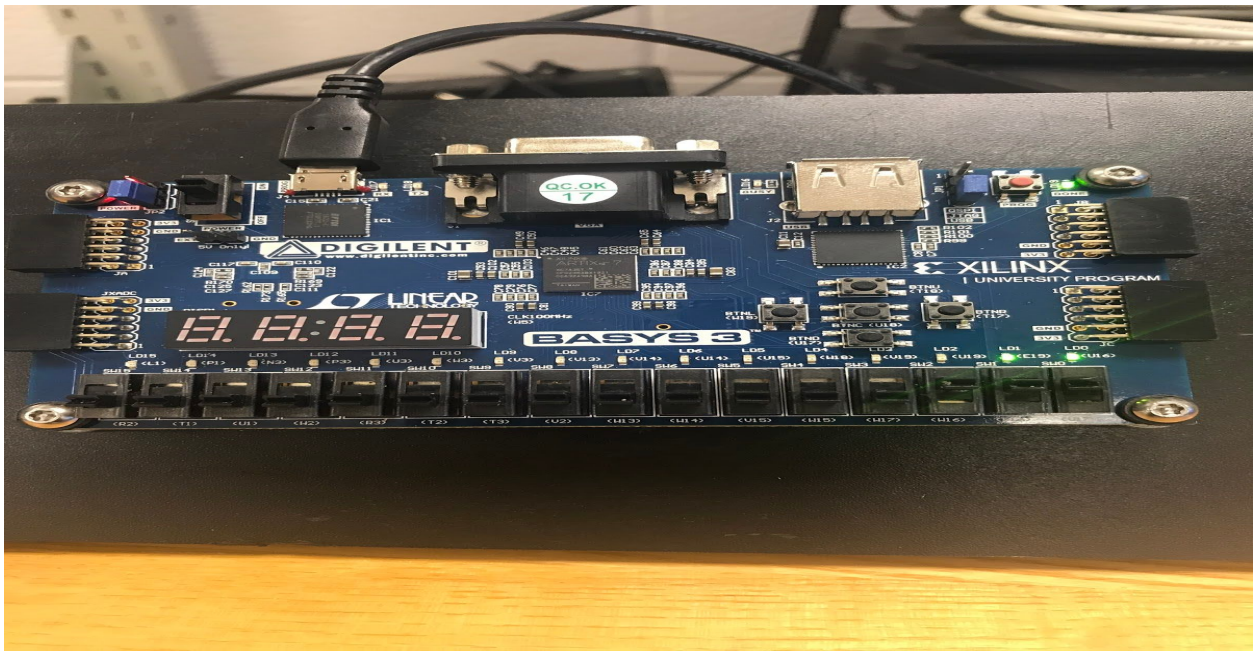


Figure 3.7)[111];[11]

Conclusion:

In conclusion, this lab was fairly achievable and allowed us to better our skills at implementing 7-segment displays as well as full adders. We were able to use truth tables to assist with creating VHDL codes to program the Basys 3 board. This lab also introduced us to XOR gates. We also learned that binary encoders can have 4-to-2, 8-to-3 and 16-to-4 line configurations.

Appendices:

Part 1: VHDL Code for BCD to 7-Segment Display Driver

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

  entity bcdto7 is
    port ( I : in STD_LOGIC_VECTOR (3 downto 0);--input
          F : out STD_LOGIC_VECTOR (6 downto 0));--output
  end bcdto7;

  architecture behavior of bcdto7 is
    begin
     process(I)
       begin
         case I is
         when "0000" => F <= "1000000"; --0
         when "0001" => F <= "1001111"; --1
         when "0010" => F <= "0010010"; --2
         when "0011" => F <= "0000110"; --3
         when "0100" => F <= "0001101"; --4
         when "0101" => F <= "0100100"; --5
         when "0110" => F <= "0100000"; --6
         when "0111" => F <= "1001110"; --7
         when "1000" => F <= "0000000"; --8
         when "1001" => F <= "0000100"; --9
         when others => F <= "0000001"; --null
         end case;
    end process;
```

```
end behavior;
```

Part 2: 4:2 Decoder VHDL Code

```
entity fourtotwoconverter is
   Port ( A3 : in STD_LOGIC;
          A2 : in STD_LOGIC;
          A1 : in STD_LOGIC;
          A0 : in STD_LOGIC;
          F0 : out STD_LOGIC;
          F1 : out STD_LOGIC);
end fourtotwoconverter;

architecture Behavioral of fourtotwoconverter is


begin
   F0 <= A3 or A1;
   F1 <= (not A1) and (not A0);

end Behavioral;
```

Part 3: Full Adder VHDL Code

```
entity Lab2Part3 is --for a full adder
   Port ( A : in STD_LOGIC;
          B : in STD_LOGIC;
          Cin : in STD_LOGIC;
          Co : out STD_LOGIC;
          Sum : out STD_LOGIC);
end Lab2Part3;

architecture Behavioral of Lab2Part3 is
          begin
          Sum <= Cin XOR (A XOR B) ;
          Co <= (Cin AND (A OR B))  OR  (A AND B);
```

```
end Behavioral;
```