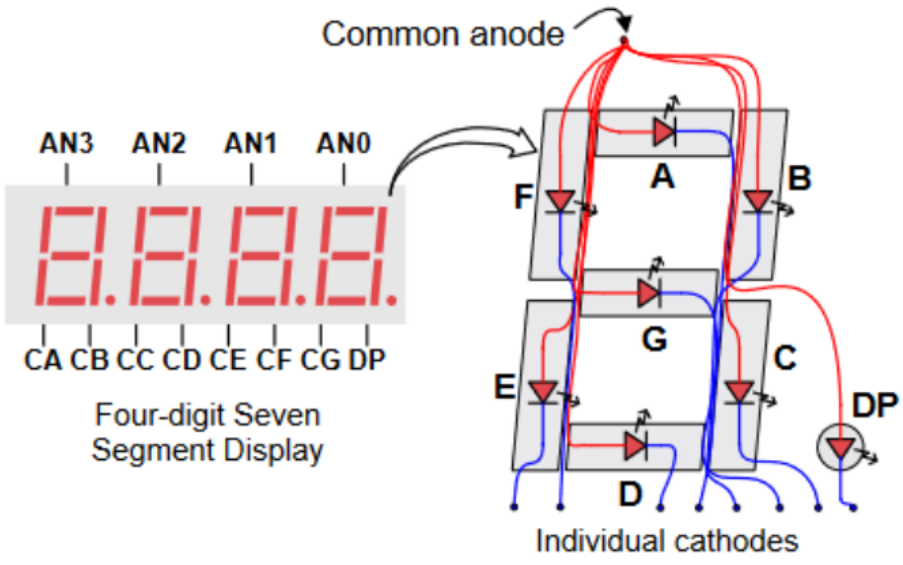
COL215 HW Assignment - 1

4-Digit 7-Segment Display

Jahnabi Roy : 2022CS11094
Anamika : 2022CS11098

Date : 1st August, 2023



Pin Details for 7-segment display on Basys3 board.

1 AIM

To design and implement a circuit that takes a 4-digit decimal/hexadecimal number as input and displays it on the seven-segment displays on the board.

2 MODULE IMPLEMENTED

2.1 Seven-Segment Decoder

We implemented this module to display a 4-bit number on a 7-segment display. The display has 7 cathode pins and an anode pin that is common for all the seven LEDs and one pin is for the decimal point. To switch on an LED we drive the anode HIGH and the cathode LOW so current flows due to the potential difference created. The inputs are taken from the switches present on the board. 4 switches were used for taking an input of a 4-bit number. Important point to note on the Basys 3 board the anode/cathode are ACTIVE LOW pins i.e. they are active when it is in logic 0 level.

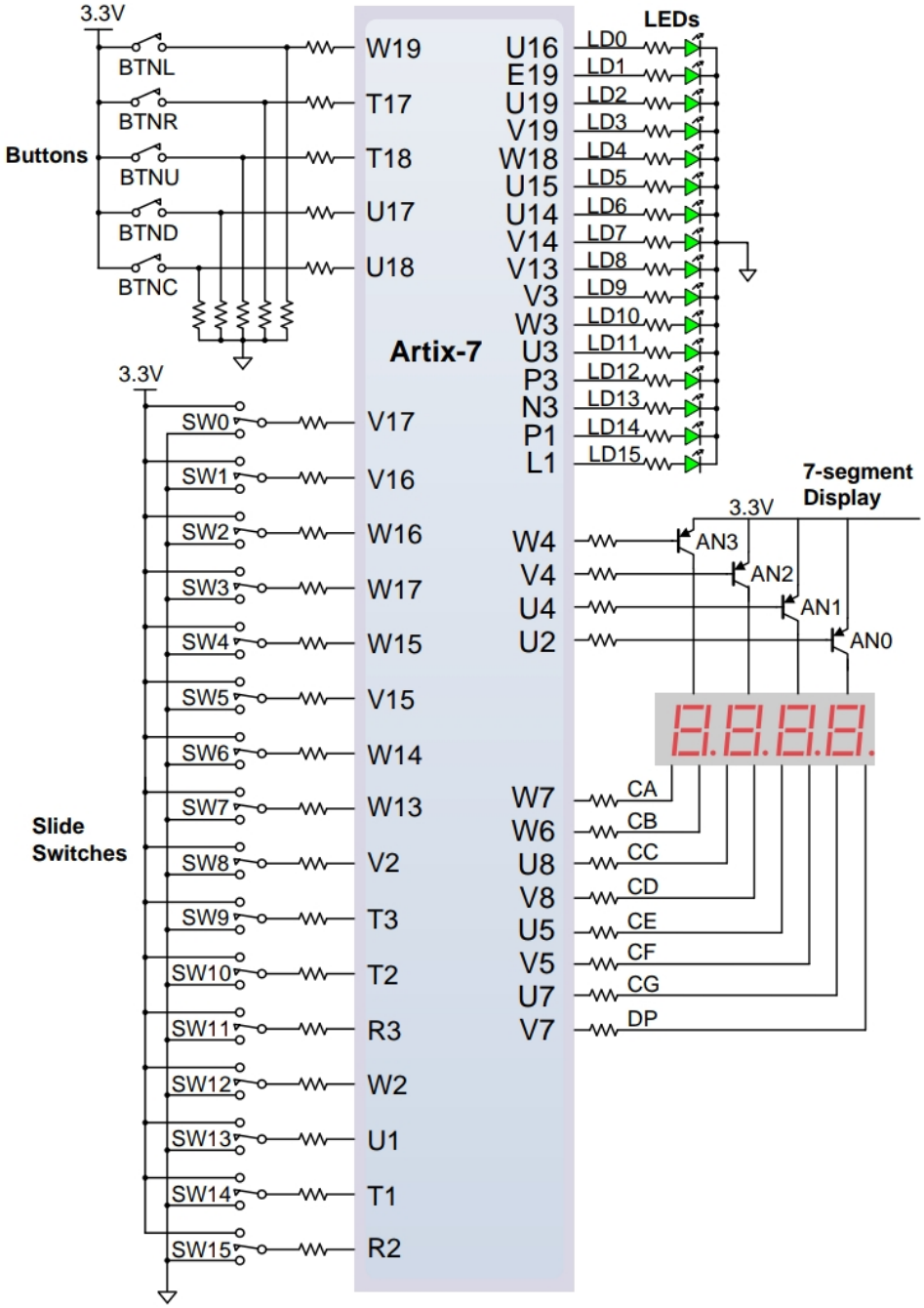
2.1.1 OUR APPROACH

- Created a truth table with 4 inputs (4-bit binary representation of the number to be displayed in the hexadecimal on the seven segment display) and seven outputs (each output corresponding to a LED). In the truth table 1 for the output means that the LED corresponding to that is to ON. Ultimately, we will take the negation ('not') of the entire relation obtained and thus, have the correct corresponding boolean value to the active state of the anode and cathode pins.
- Made K-Maps to get the logical expression for the output of each LED i.e for the seven segments of the display. Our logical expression includes only the elementary operations

OR/AND.

- Then created a VHDL source file for the SEVEN-SEGMENT-DECODER.
- Used the constraint file provided to us and modified it according to the requirement and included it as constraint file in our project.
- We mapped V17, V16, W16, W17 to inputs and W7, W6, U8, V8, U5, V5 and U7 to outputs.
- We created three variables b1, b2 and b3 for the three anodes to be turned off and set their boolean to '1'.
- We then ran synthesis and then the output of the synthesis was used to run implementation.
- The output of the implementation was used to generate the bitstream.
  We then demonstrated the functioning to our TA.

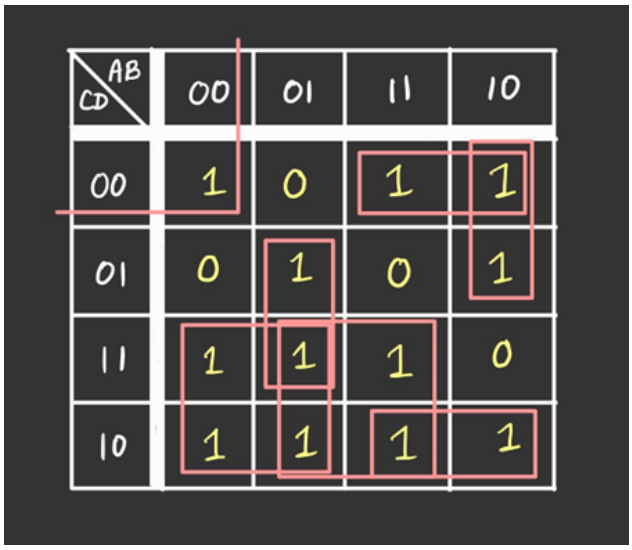
2.1.2 KARNAUGH MAPS



The internal cathode and anode connection details of the Basys3 board.

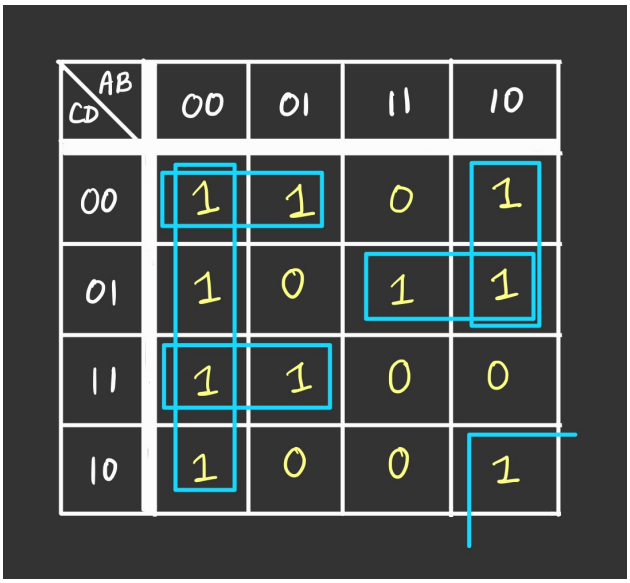
Truth Table for different inputs (A, B, C, D) and corresponding outputs.

ABCD	***a1***	***a2***	***a3***	***a4***	***a5***	***a6***	***a7***
0000	1	1	1	1	1	1	0
0001	0	1	1	0	0	0	0
0010	1	1	0	1	1	0	1
0011	1	1	1	1	0	0	1
0100	0	1	1	0	0	1	1
0101	1	0	1	1	0	1	1
0110	1	0	1	1	1	1	1
0111	1	1	1	0	0	0	0
1000	1	1	1	1	1	1	1
1001	1	1	1	1	0	1	1
1010	1	1	1	0	1	1	1
1011	0	0	1	1	1	1	1
1100	1	0	0	1	1	1	0
1101	0	1	1	1	1	0	1
1110	1	0	0	1	1	1	1
1111	0	1	1	1	0	0	0



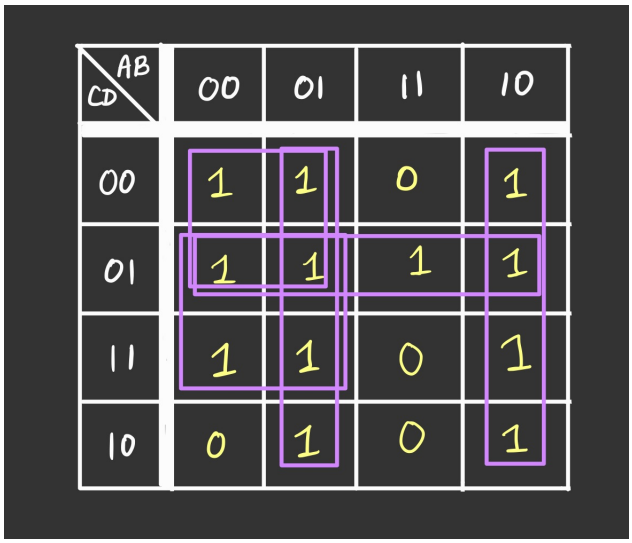
Karnaugh Map for Output bit **a1**

a1 = not(A'B'C'D' + AC'D' + AB'C' + A'B'D + A'C + BC + ACD')



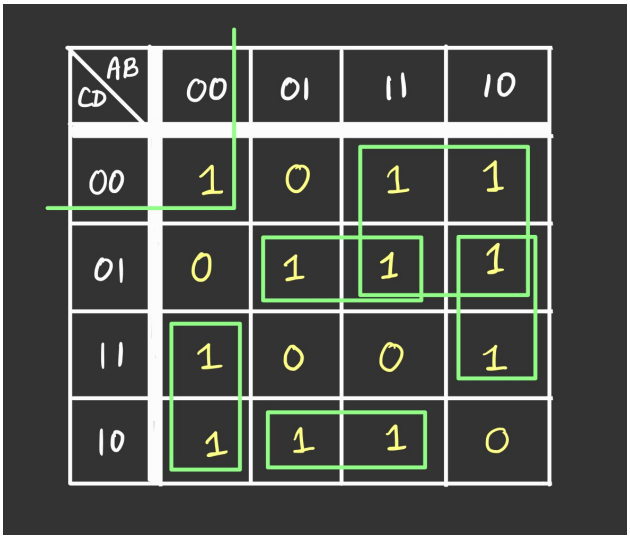
Karnaugh Map for Output bit **a2**

a2 = not(AB'CD' + AB'C' + AC'D + A'B' + A'CD + A'C'D')



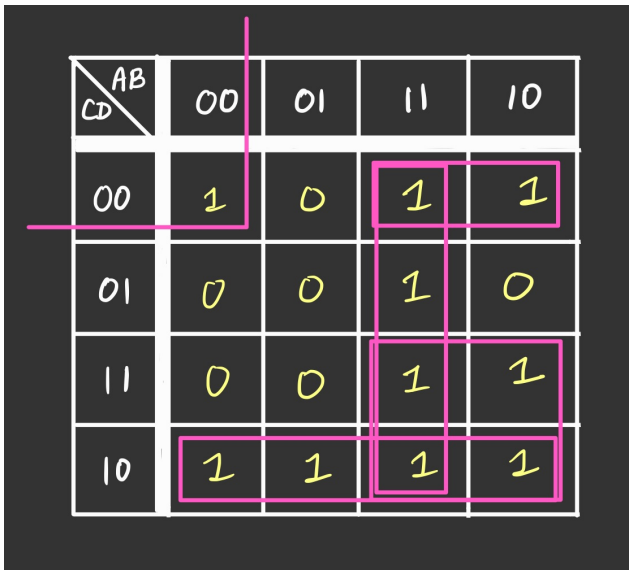
Karnaugh Map for Output bit **a3**

a3 =not(AB' + C'D + A'B + A'D + A'C')



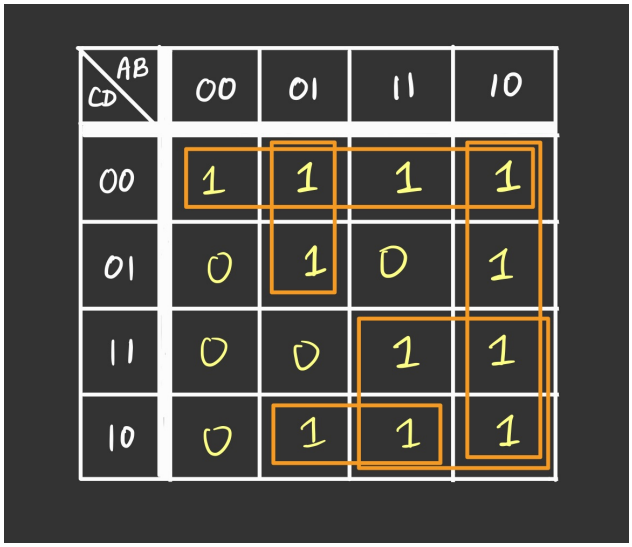
Karnaugh Map for Output bit **a4**

a4 = not (A'B'C'D' + A'B'C + BCD' + BC'D + AB'D + AC')



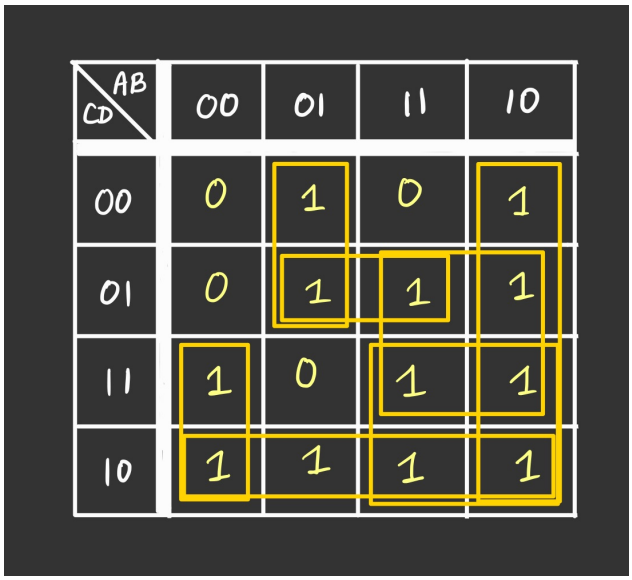
Karnaugh Map for Output bit **a5**

a5 = not(A'B'C'D' + CD' + AC + AC'D' + AB)



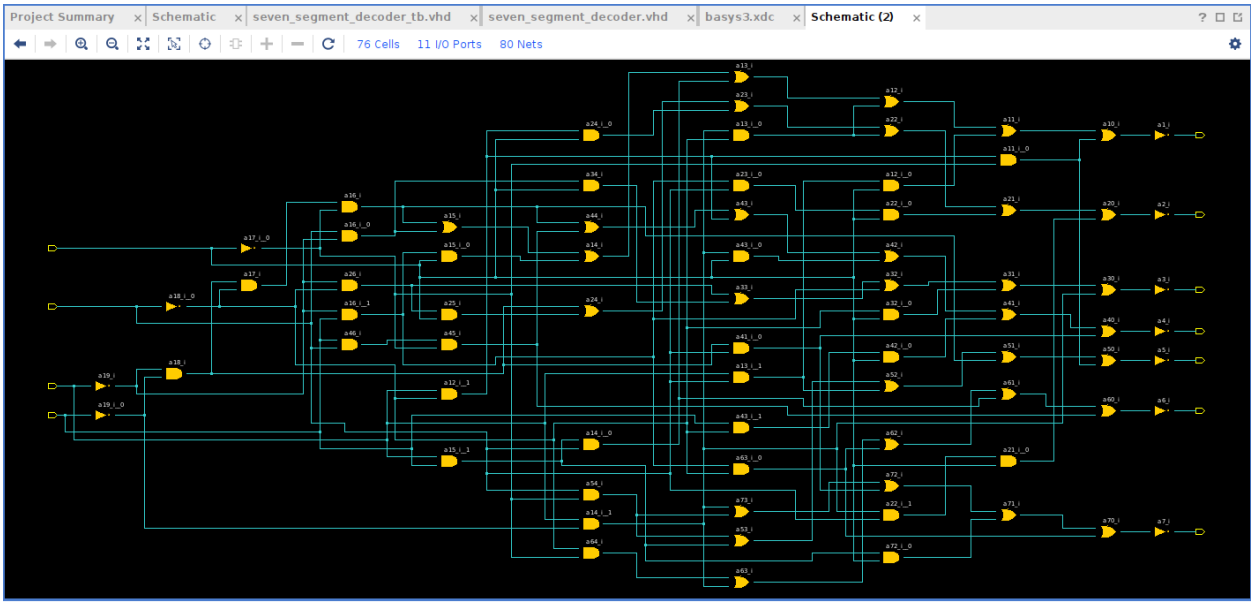
Karnaugh Map for Output bit **a6**

a6 = not (C'D' + AB' + AC + BCD' +A'BC')



Karnaugh Map for Output bit **a7**

$$\mathbf{a7 = not\ (AB' + A'B'C + CD' + ABD + A'BC')}$$

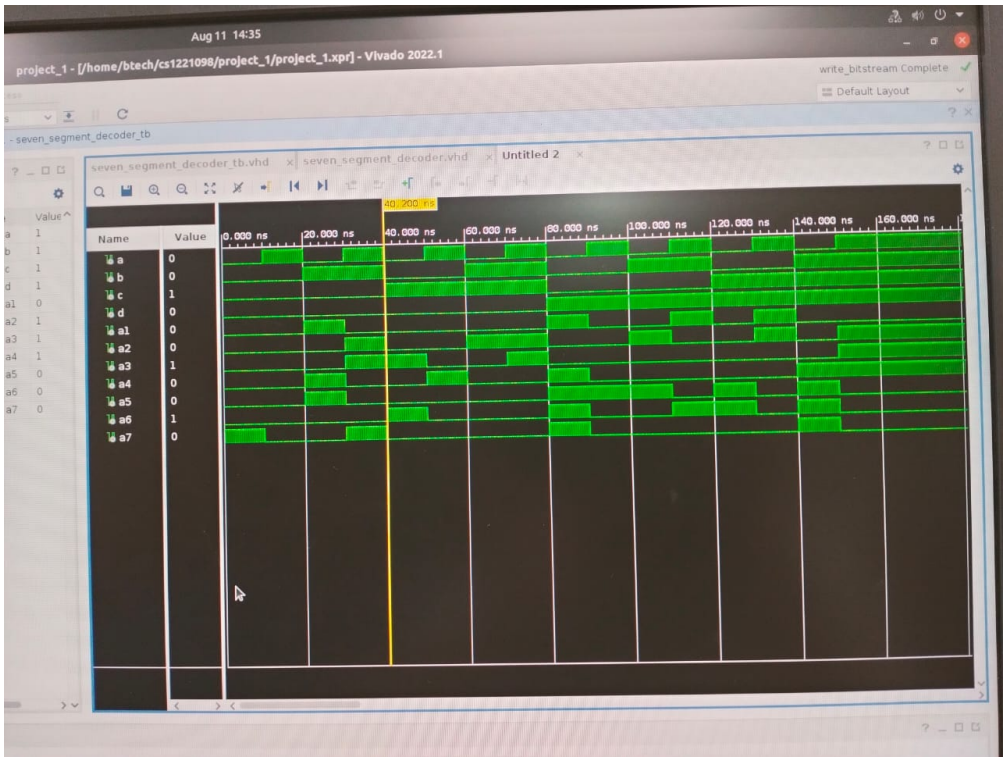


Schematic view for the above input-output relations obtained.

2.2 Seven-Segment Decoder Test Bench

For the simulation of the seven segment gate, we made a test bench which gives different inputs at different time stamps and 'run simulation' command allows us to see the wave generated in the process. This is only for simulation process and observing the resultant waveform. We connect the test bench signals with the seven segment decoder source file with the input ports mapped. The inputs are timed as (with corresponding outputs) :

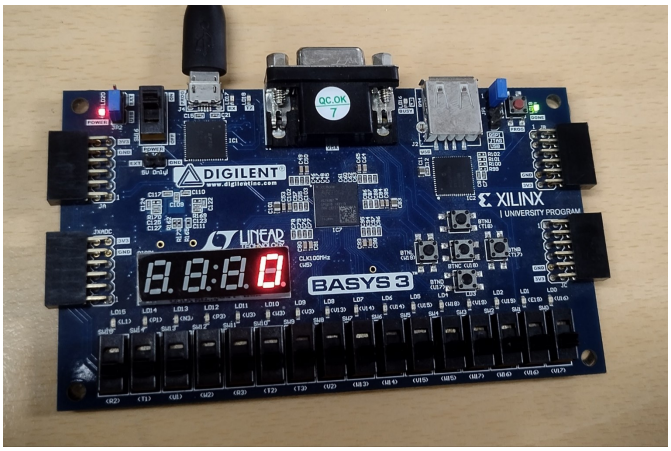
Input	Output
'0000' at 0 ns	'0000001'
'1000' at 10 ns	'0000000'
'0100' at 20 ns	'1001100'
'1100' at 30 ns	'0110001'
'0010' at 40 ns	'0010010'
'1010' at 50 ns	'0001000'
'0110' at 60 ns	'0100000'
'1110' at 70 ns	'0110000'
'0001' at 80 ns	'1001111'
'1001' at 90 ns	'0000100'
'0101' at 100 ns	'0100100'
'1101' at 110 ns	'1000010'
'0011' at 120 ns	'0000110'
'1011' at 130 ns	'1100000'
'0111' at 140 ns	'0001111'
'1111' at 150 ns	'1000111';

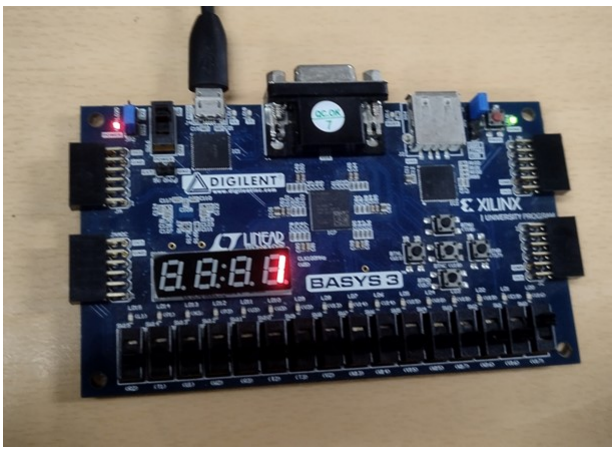


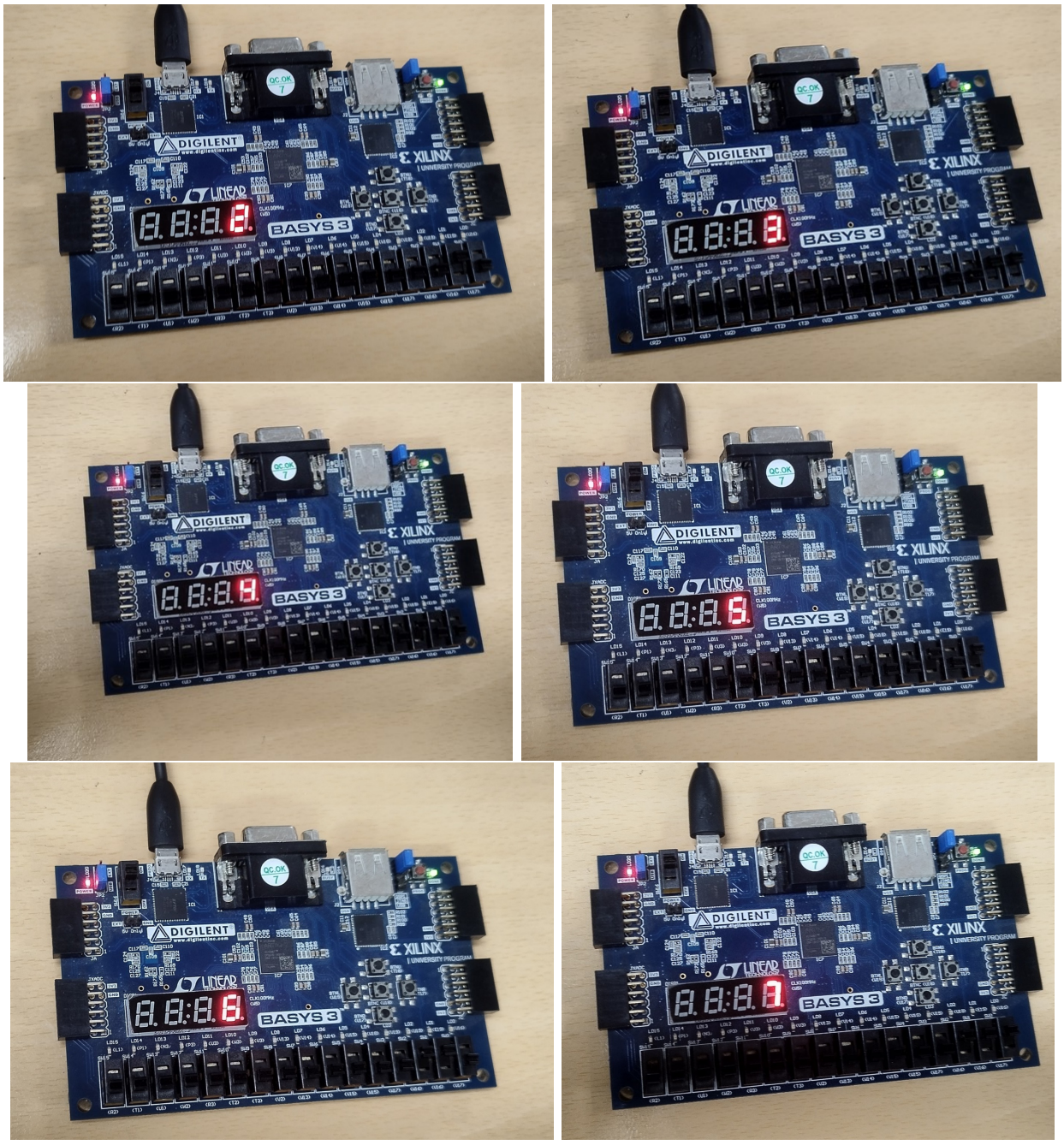
The waveform achieved for different inputs

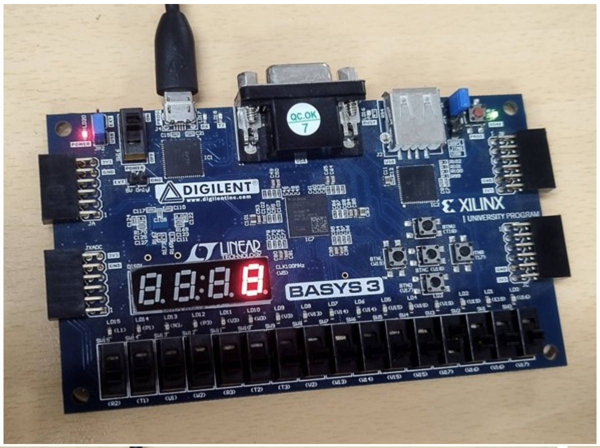
and their corresponding outputs.

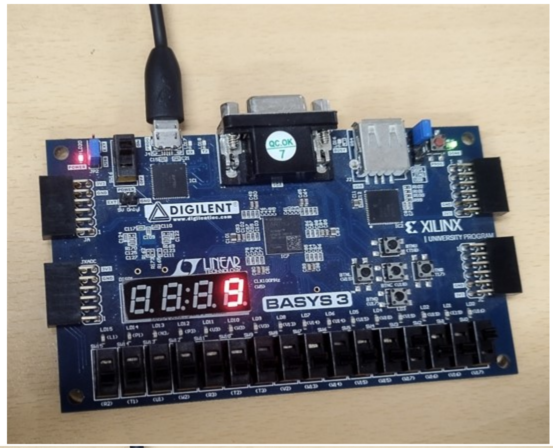
3 OUTPUT DISPLAY

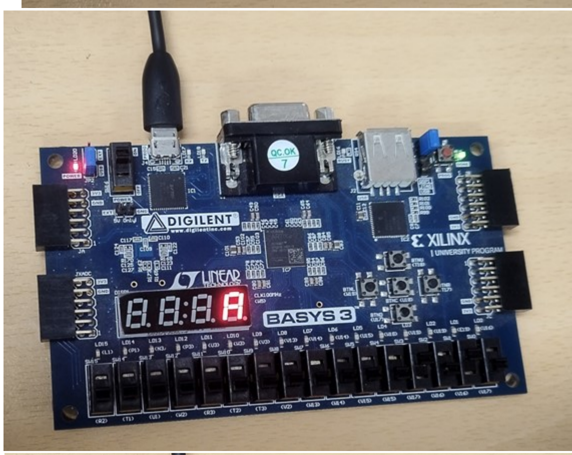


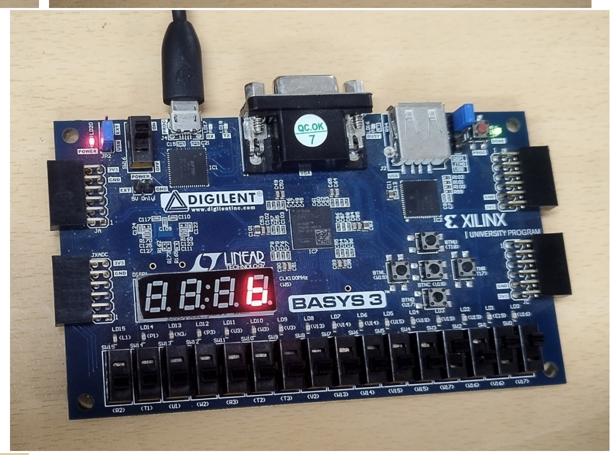


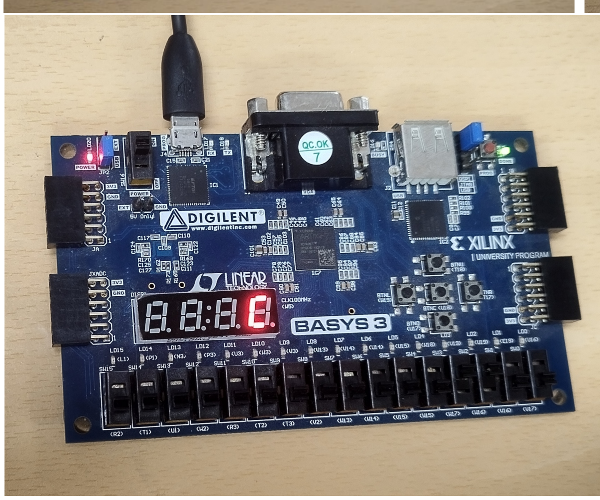

DIGILENT
XILINX
BASYS 3

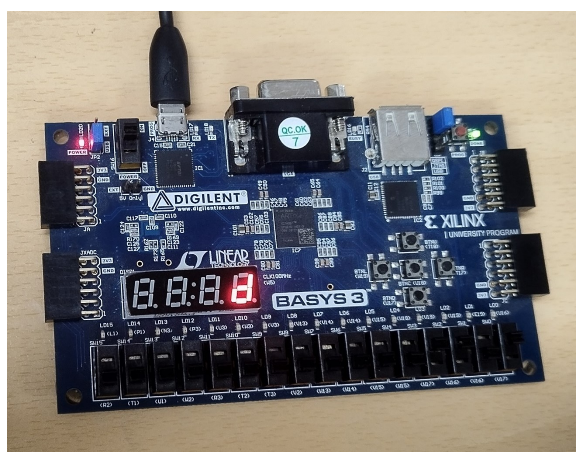

DIGILENT
XILINX
BASYS 3

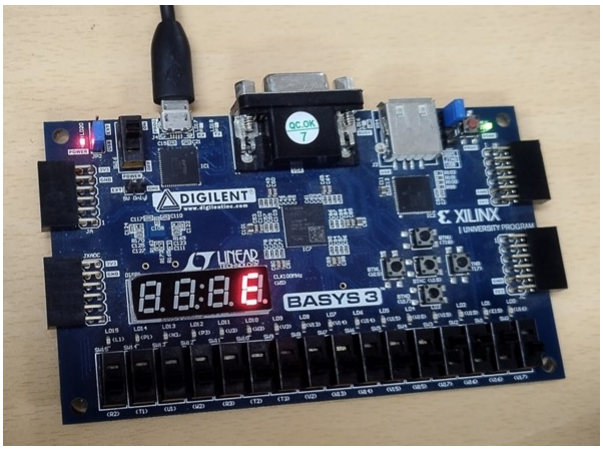

DIGILENT
XILINX
BASYS 3

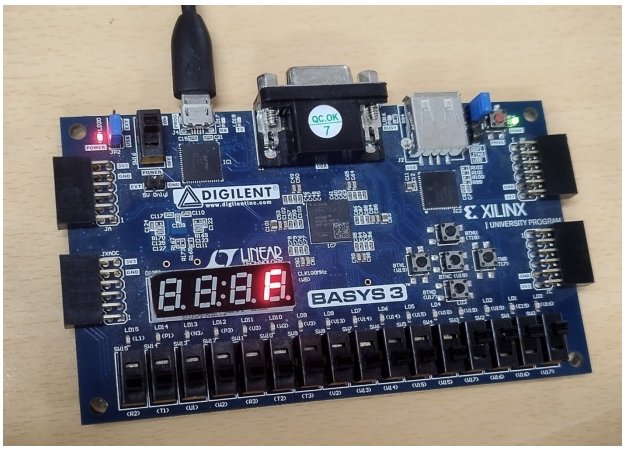

DIGILENT
XILINX
BASYS 3


DIGILENT
XILINX
BASYS 3


DIGILENT
XILINX
BASYS 3


DIGILENT
XILINX
BASYS 3





4 ATTACHMENTS AND REFERENCES

1. Slice Logic

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	4	0	0	20800	0.02
LUT as Logic	4	0	0	20800	0.02
LUT as Memory	0	0	0	9600	0.00
Slice Registers	0	0	0	41600	0.00
Register as Flip Flop	0	0	0	41600	0.00
Register as Latch	0	0	0	41600	0.00
F7 Muxes	0	0	0	16300	0.00
F8 Muxes	0	0	0	8150	0.00

The submission zip folder contains the following files :-

the constraint file `basys3.xdc`, the VHDL code file `seven_segment_decoder.vhd`, the VHDL code for testbench `seven_segment_decoder_tb.vhd`, the bit file `seven_segment_decoder.bit`, `seven_segment_decoder_utilization_synth.rpt` which contained the counts for LUTs, Flip-flops, BRAMs, and DSPs, `seven_segment_decoder_utilization_synth.pb` and `seven_segment_decoder.vds`.

For reference, we used the references given to us in the assignment (the Basys3 board reference manual), the hardware assignment 0 and the lectures on Karnaugh Maps in ELL101.