Hardware Assignment 1

Jaskaran Singh Bhalla (2021TT11139) **Saurabh Arge (2022CS11610)**

Problem

- Design a combinational circuit and implement.
- Take a 4-digit decimal/hexadecimal number (4-bit number) from switches in the Basys3 board.
- Display output on the 4-seven segment displays on the board.
- Display the output on a single board.

Approach

- **Assumption**
  - We have represented 4-bits using the hexadecimal system.
- **Analysis**
  - We understood the functioning of a single digit on one of the four seven segment displays including the connections between 7 cathodes and 1 anode.
  - We figured out how input is taken using the switch from the Basys3 board.
- **Calculations**
  - We created the k-maps for each of the 7 leds (cathodes) and derived the combinational logic to display 16 different outputs on the display using 4-bit input given through switches.
  - The **Seven K-maps** with their derived combination logic are given below.
  - We have derived the combination logics for each led and then taken negation of each logic in implementation through code because in the problem statement Active mode is taken to be 0 and inactive state is taken as 1.
  - **ACTIVE = 0, INACTIVE = 1**

Implementation

- We wrote the code using combinational logic in VHDL and generated the bitstream and ran the code on the Basys3 board.
- For a single display we created 4 variables and with 3 of the variables set to inactive (1) state indicating that the display is off and 1 variable is marked in active state (0) indicating that 1 display is on.

Results

- We manually confirmed all the outputs on the Basys3 board.
- With no flickering all the displays , correctly displayed the output
- Later using the 4 variable logic we were able to display the 4-bit digit on a single display only.
- Images of the output have been attached below.

Concluding Remarks

It was a great experience learning VHDL and implementing the design for the 7-segment decoder on our own. Understanding the display LED logic using k-maps and implementing it practically was a really nice experience.

K-Maps

Output LED a

AB \ CD	00	01	11	10
00	1	0	1	1
01	0	1	1	1
11	1	0	1	1
10	1	1	0	1

Combinational Logic :
a => A’B’C’D’ + A’BC’D + AB’C’D + AB’CD’ + AC’D’ + A’B’C + BC

Output LED b

AB \ CD	00	01	11	10
00	1	1	1	1
01	1	0	1	0
11	0	1	0	0
10	1	1	0	1

Combinational Logic :
b => A’BC’D’ + A’BCD + AB’CD’ + AB’C’D’ + AC’D + A’B’

Output LED c

AB \ CD	00	01	11	10
00	1	1	1	0
01	1	1	1	1
11	0	1	0	0
10	1	1	1	1

Output LED c

Combinational Logic :
c => ABC'D + A'B'CD + A'B'C' + AB' + A'B

Output LED d

AB \ CD	00	01	11	10
00	1	0	1	0
01	0	1	0	1
11	1	1	0	1
10	1	1	1	0

Output LED d

Combinational Logic :
d => A'B'C'D' + A'BC'D + AB'CD + A'B'C + BCD' + AC'

Output LED e

AB \ CD	00	01	11	10
00	1	0	0	1
01	0	0	0	1
11	1	1	1	1
10	1	0	1	1

Output LED e

Combinational Logic :
e => A’B’C’D’ + AB’C’D’ + A’CD’ + AB’C + AB

Output LED f

AB \ CD	00	01	11	10
00	1	0	0	0
01	1	1	0	1
11	1	0	1	1
10	1	1	1	1

Output LED f

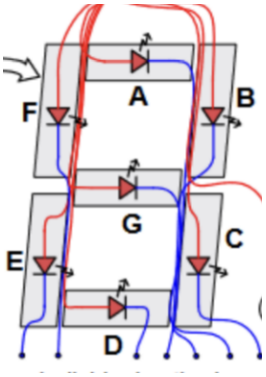
Combinational Logic :
f => A’B’C’D’ + A’BCD’ + ABC’D’ + A’BC’ + ABC + AB’

Output LED g

AB \ CD	00	01	11	10
00	0	0	1	1
01	1	1	0	1
11	0	1	1	1
10	1	1	1	1

Combinational Logic :
g => A'BCD' + AB'C'D' + A'B'C + A'BC' + AC'D + AC

Mapping of LEDs to VHDM code variables



a = Combination Logic for LED A
b = Combination Logic for LED B
c = Combination Logic for LED C
d = Combination Logic for LED D
e = Combination Logic for LED E
f = Combination Logic for LED F
g = Combination Logic for LED G

Visual Outputs

Input format : ABCD
For Example , Take a 4-bit digit with numeric value = 5
A = 0, B = 1, C = 0, D = 1
It will display 5.

Input	Visual
0000	
0001	
0010	
0011	
0100	
0101	
0110	
0111	
1000	
1001	
1010	
1011	
1100	
1101	
1110	
1111	

Basys3 Board Output