**Introduction**

The principal purpose of this lab is to learn about programmable logic where VHDL is used to specify the functionality. You will implement a complex combinational circuit that includes about programmable logic and VHDL to drive a seven-segment display. Objective: You are to design the system shown below which is a code converter to convert a 4 bit binary code into a seven segment display output code **(0-9,A-F)**

Here are diagrams of the two chips used:

**Preliminary Work**

1) Knowing visually how numbers and letters appear and using the chip diagrams above, we were able to create a truth table. ABCD are the inputs and abcdefg are the outputs.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| # | A | B | C | D | a | b | c | d | e | f | g |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 2 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 3 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 4 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 5 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 6 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 7 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| 8 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| A | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| b | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| C | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| d | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| E | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| F | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |

Note: Some letters are capitalized and some are not. That is because were able to choose how to display numbers and letters the way we wanted to.

2. After that, we simplified the expressions for each of the seven functions using the online K-map tool.

|  |  |
| --- | --- |
| F | Expression |
| a | ¬AC + BC + ¬B¬D + ¬ABD + A¬B¬C |
| b | A¬B + AC + ¬AB¬D + ¬A¬C¬D |
| c | AB + C¬D + AC + A¬D + ¬B¬D |
| d | ¬A¬B¬D + AB¬D + ¬AC¬D + B¬CD + A¬C¬D + ¬BCD | |
| e | ¬AB + A¬B + ¬CD + ¬A¬C + ¬AD |
| f | ¬B¬C + ¬B¬D + ¬A¬C¬D + ¬ACD + AD¬C |
| g | A + C¬D + B¬C + ¬BC |

**Lab Work**

3. We then entered those expressions into VHDL and added a stimulus.

--expressions

Ya <= ((not A) and C) or (B and C ) or ((not B) AnD (not D)) or ((not A) and B AND D) or (A and (not B) and (not C));

Yb <= (A and (not B)) or (A and C) or ((not A) and B and (not D)) or ((not A) and (not C) and (not D));

Yc <= (A and B) or (C and (not D)) or (A and C) or (A and (not D)) or ((not B) and (not D));

Yd <= ((not A) and (not B) and (not D)) or (A and B and (not D)) or ((not A) and C and (not D)) or (B and (not C) and D) or (A and (not C) and (not D)) or ((not B) and C and D);

Ye <= ((not A) and B) or (A and (not B)) or ((not C) and D) or ((not A) and (not C)) or ((not A) and D);

Yf <= ((not B) and (not C)) or ((not B) and (not D)) or ((not A) and (not C) and (not D)) or ((not A) and C and D) or (A and D and (not C));

Yg <= (A) or (C and (not D)) or (B and (not C)) or ((not B) and C);

--stimulus

D <= '0', '1' after 10ns, '0' after 20ns, '1' after 30ns, '0' after 40ns,

'1' after 50ns, '0' after 60ns, '1' after 70ns, '0' after 80ns, '1' after

90ns, '0' after 100ns, '1' after 110ns, '0' after 120ns, '1' after 130ns,

'0' after 140ns, '1' after 150ns;

C <= '0', '1' after 20ns, '0' after 40ns, '1' after 60ns, '0' after 80ns,

'1' after 100ns, '0' after 120ns, '1' after 140ns;

B <= '0', '1' after 40ns, '0' after 80ns, '1' after 120ns;

A <= '0', '1' after 80ns;

4. We then generated a .jed file from the program.

5. Using the pinouts from the .jed file we then created the wire list.

U1 - PALCE22V10

U2 - HD74LS04P

U3 - HD74LS04P

U4 - display

VCC🡪U1-24,U2-14,U3-14

GND🡪U1-12,U2-3,U4-7

A🡪U1-1

B🡪U1-2

C🡪U1-3

D🡪U1-4

U1-20🡪Ya🡪U2-1

U1-16🡪Yb🡪U2-3

U1-21🡪Yc🡪U2-5

U1-15🡪Yd🡪U2-9

U1-22🡪Ye🡪U2-11

U1-14🡪Yf🡪U2-13

U1-13🡪Yg🡪U3-1

U2-2 🡪U4-1

U2-4 🡪U4-2

U2-6 🡪U4-5

U2-8 🡪U4-6

U2-10🡪U4-8

U2-12🡪U4-10

U3-2 🡪U4-9

6. Finally we wired the circuit.

**Results**

No one was able to get this to work properly. All I was able to get was a blinking light in the center led. The main problem was finding the matching chip in the software. This was our first time using this software, and there was a lot to learn in very little time. Hopefully next time we’ll know the device-type and get quicker at using the program