**CSARCH1 Design Exercise #1**
**S14 - Group 3**

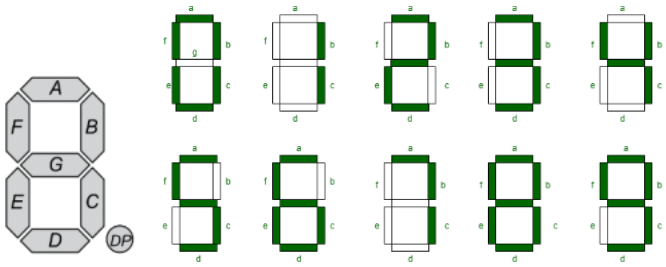
**Submitted by:**
**Homssi, Yazan**
**Ramos, Aebrahm**

**Submitted to:**
**Lovely professor Sir Carlo Adriano**

**February 12, 2024**

## 1. Truth Table

For this Design Exercise, unlike following the BCD ( 8, 4, 2, 1 ) we are familiar with, we will instead be following the ( 2, 1, 2, 6 ) binary codes inputted as ( W, X, Y, Z ) respectively. The outputs on the other hand correspond to 7 lights as ( A, B, C, D, E, F, G ). Two images for your reference down below:




The truth table below for this output displays the number from the BCD:

| W (2) | X (1) | Y (2) | Z (6) | A | B | C | D | E | F | G | Display |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 6 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 2 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 8 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 7 |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 3 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 9 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 2 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 8 |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 4 |
| *1* | *0* | *1* | *1* | *0* | *0* | *0* | *0* | *0* | *0* | *0* | *Don't Care* 😕 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 3 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 9 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 5 |
| *1* | *1* | *1* | *1* | *0* | *0* | *0* | *0* | *0* | *0* | *0* | *Don't Care* 😕 |

If you have noticed some of the results are displayed in red, this is because in the “2126” code:
2, 3, and 9 can be represented in two different ways.

- 2 = ( 0, 0, 1, 0 ) or ( 1, 0, 0, 0 )
- 3 = ( 0, 1, 1, 0 ) or ( 1, 1, 0, 0 )
- 9 = ( 0, 1, 1, 1 ) or ( 1, 1, 0, 1 )

The 2 “Don’t Care 😕” results are displayed because they go beyond 9.

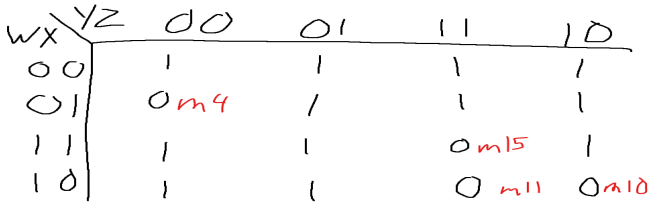
- 10 = ( 1, 0, 1, 1 )
- 11 = ( 1, 1, 1, 1 )

==========

**2. Karnaugh Maps:**

In order to get the equations needed to assist us with designing our logic gates in CircuitVerse, we made use of Karnaugh Maps ( K-maps ) in getting the Sum of Product ( SOP ), which would then lead us to getting the boolean expression needed for our next part.

Output A:




SOP Equation F( W, X, Y, Z ) = ∑m( 0, 1, 2, 3, 5, 6, 7, 8, 9, 12, 13, 14)
F = W`X` + W`Z + WY` + XYZ`

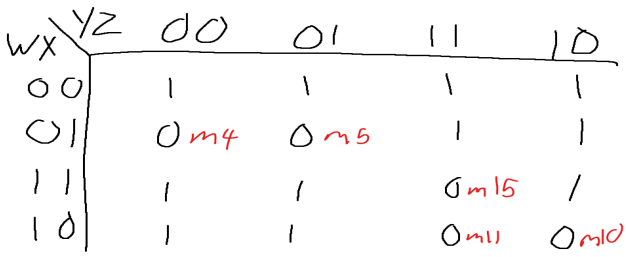
Output B:


SOP Equation F( W, X, Y, Z ) = ∑m( 0, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13)
F = X`Z` + W`Y + W`X` + WY`

Output C:

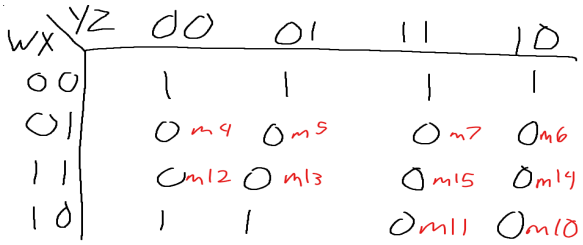

SOP Equation F( W, X, Y, Z ) = ∑m( 0, 1, 3, 4, 5, 6, 7, 9, 10, 12, 13, 14)
F = XZ` + Y`Z + W`Z + W`Y` + WYZ`

Output D:



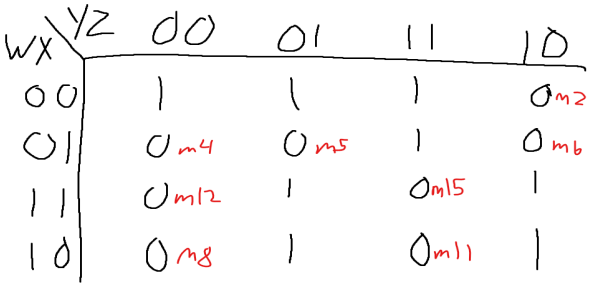

SOP Equation F( W, X, Y, Z ) = ∑m( 0, 1, 2, 3, 6, 7, 8, 9, 12, 13, 14)
F = W`X` + W`Y + WY` + XYZ`

Output E:



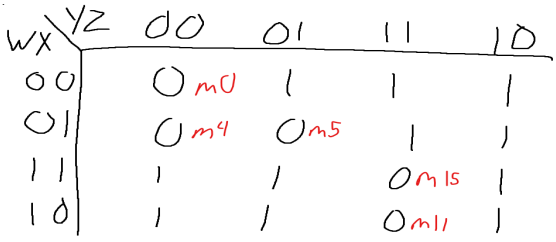

SOP Equation F( W, X, Y, Z ) = ∑m( 0, 1, 2, 3, 8, 9)
F = W`X` + X`Y`

Output F:




SOP Equation F( W, X, Y, Z ) = ∑m( 0, 1, 3, 7, 9, 10, 13, 14)
F = W`X`Y` + W`YZ + WX`Z + WYZ`

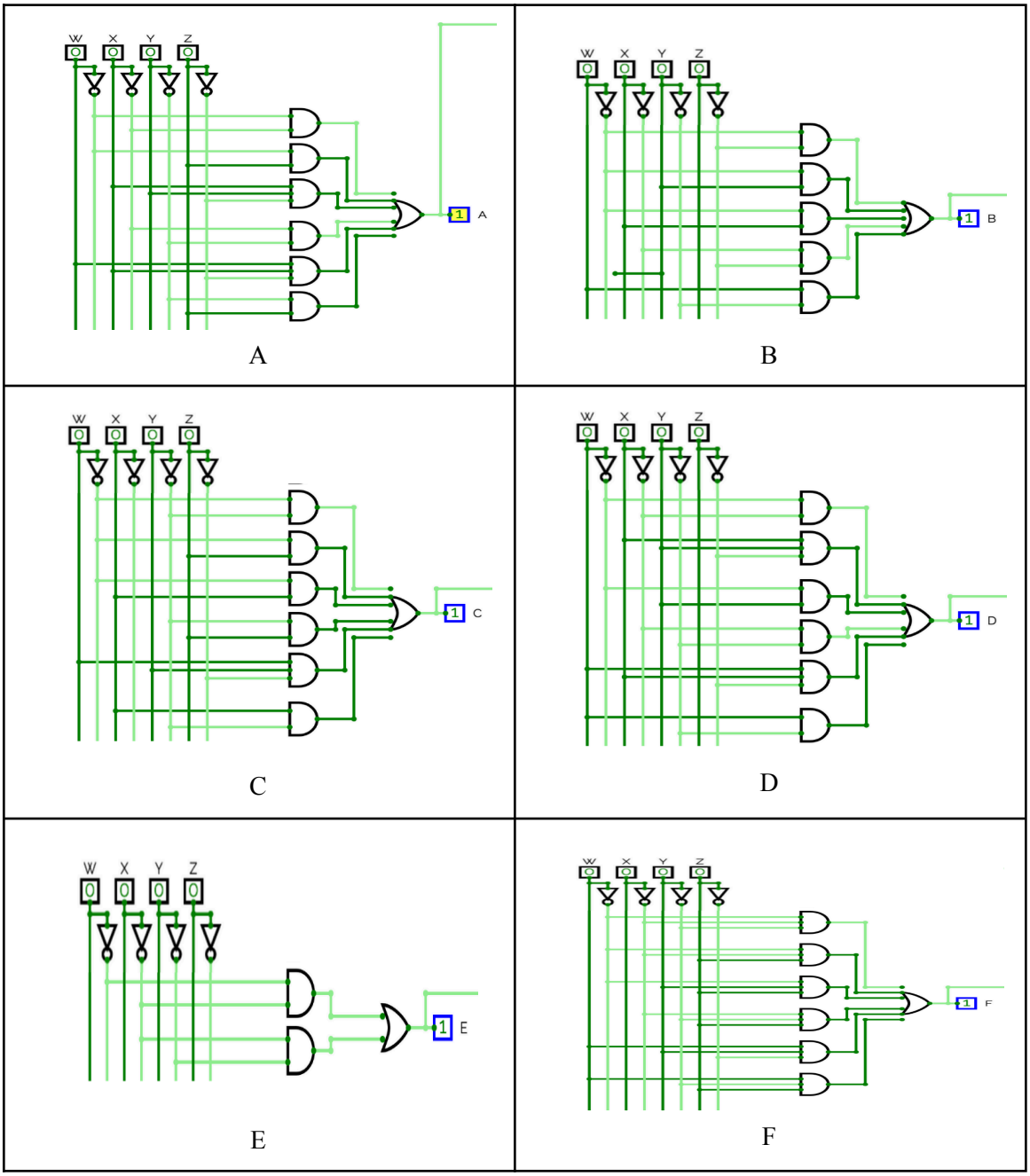
Output G:

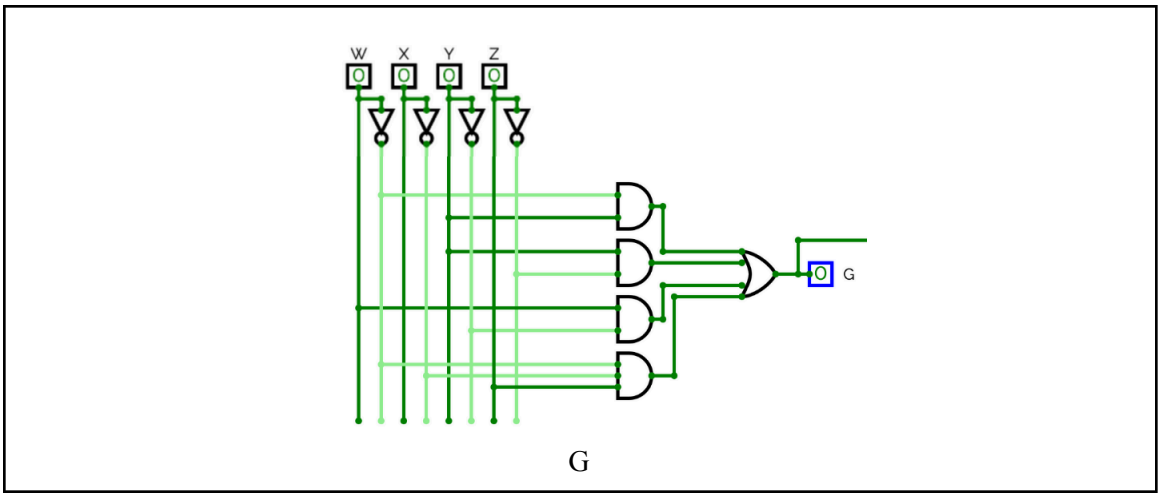



SOP Equation F( W, X, Y, Z ) = ∑m( 1, 2, 3, 6, 7, 8, 9, 10, 12, 13, 14)
F = W`Y + YZ` + WY` + W`X`Z

## 3. Logic Circuit

After getting the equations from the K-Map, we then implemented it in CircuitVerse through logic gates/circuits. The images below show the logic implementation of each output from A to G.



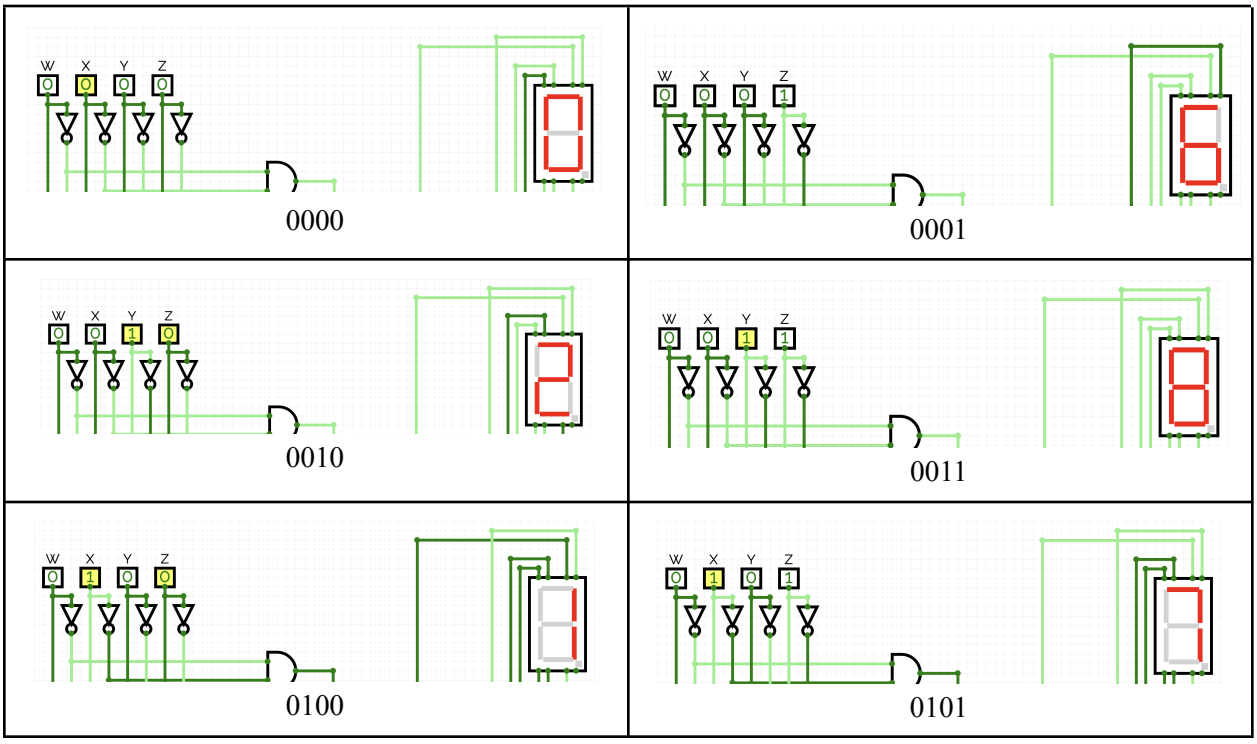


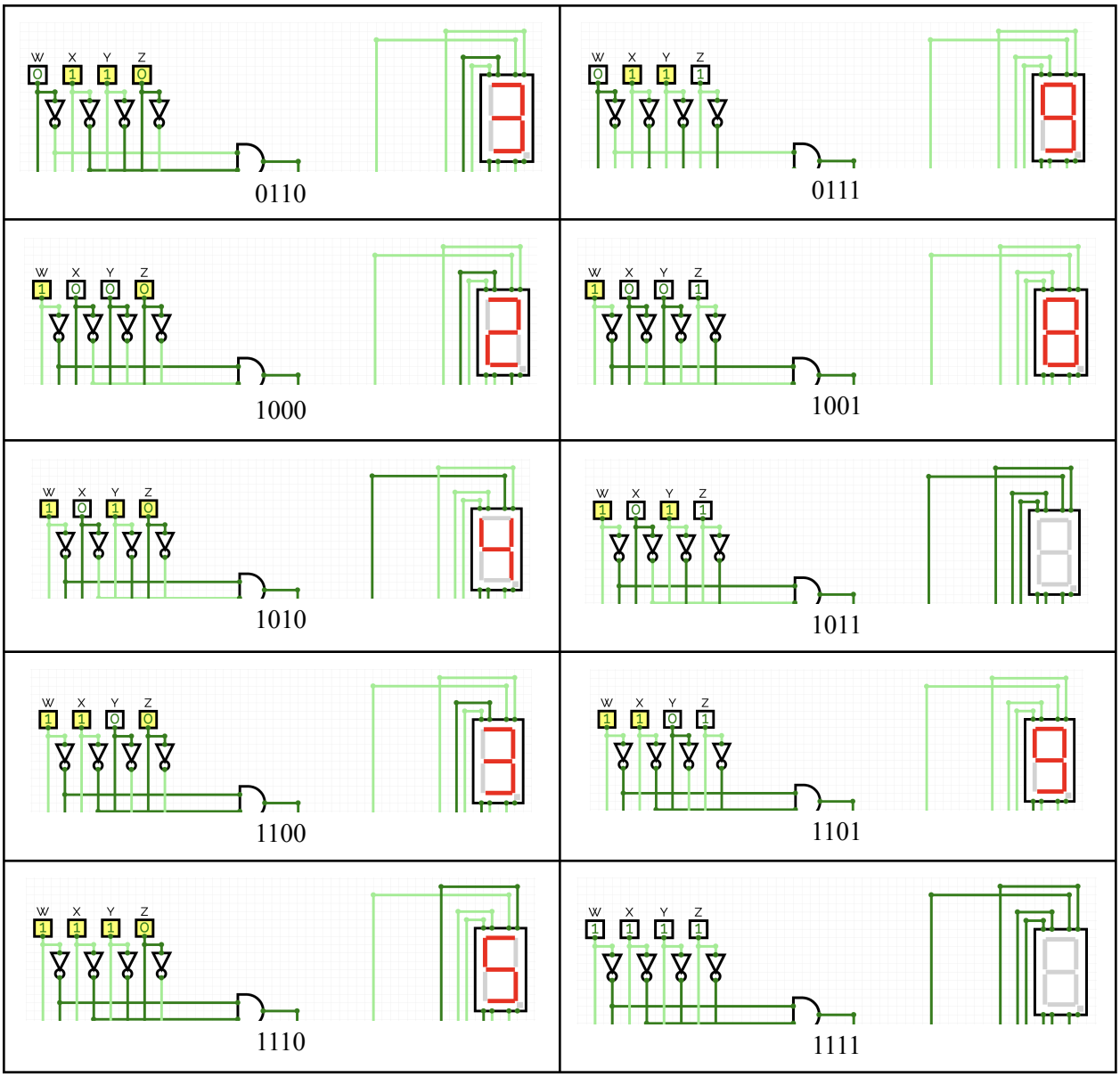

G

==========

**4. Circuit Output**

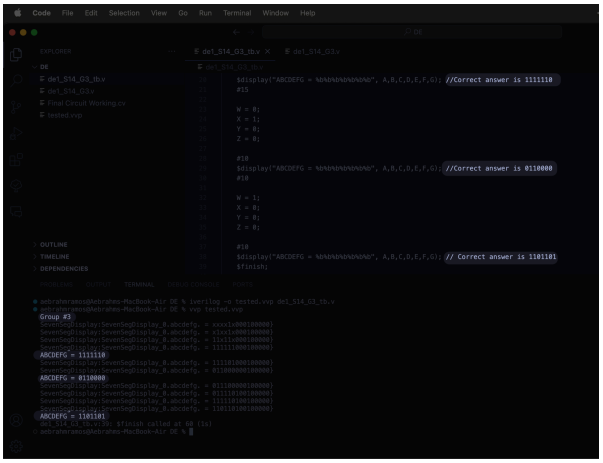
This part of the documentation will serve as proof that our circuit works while following the ( 2, 1, 2, 6 ) code digit format.



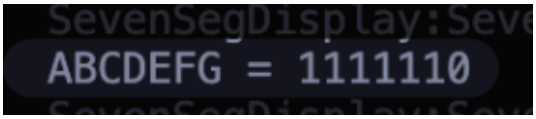
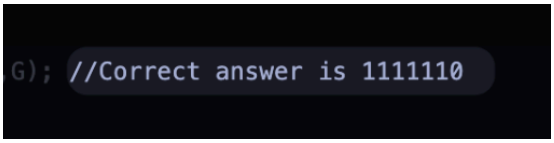
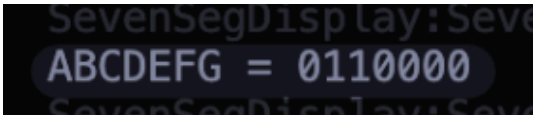
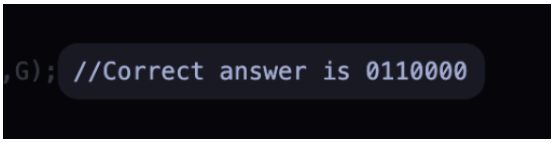
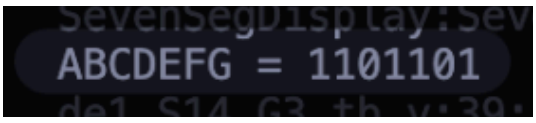
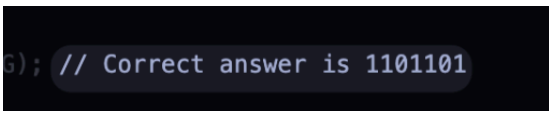

W X Y Z
0 1 1 0
0110
W X Y Z
0 1 1 1
0111
W X Y Z
1 0 0 0
1000
W X Y Z
1 0 0 1
1001
W X Y Z
1 0 1 0
1010
W X Y Z
1 0 1 1
1011
W X Y Z
1 1 0 0
1100
W X Y Z
1 1 0 1
1101
W X Y Z
1 1 1 0
1110
W X Y Z
1 1 1 1
1111

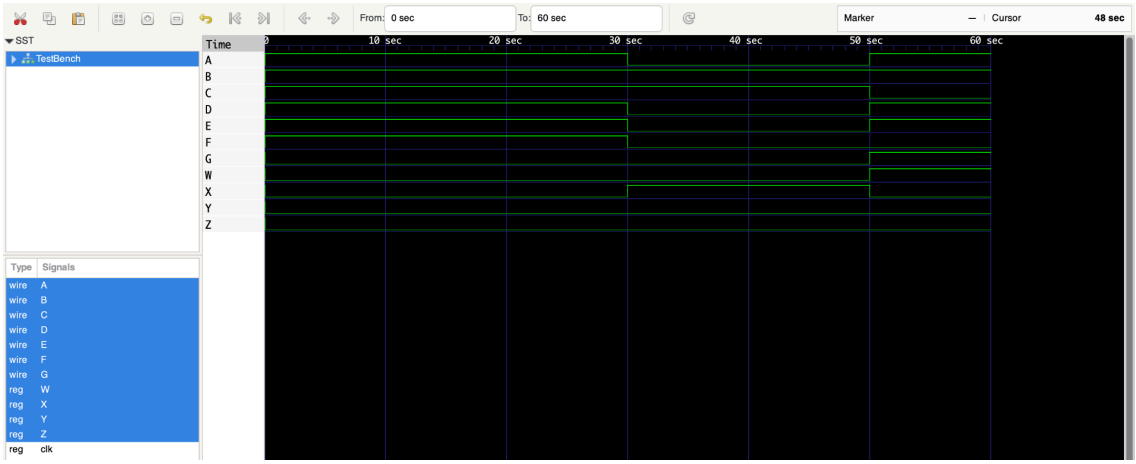
==========

5. **Verilog Test Bench**



This part of the documentation serves as proof that the Verilog file of our circuit passes the autochecker. As seen in the image below, our .vvp file has outputted the same result as the one written as a comment on the test bench file given by the instructor. In addition, the waveform of our circuit can be seen in the image below the table. This waveform was produced or visualized through the use of GTK Wave. Producing the vcd file for GTK Wave required the modification of the test bench to add “$dumpfile("de1_S14_G3.vcd");” and “$dumpvars(0, TestBench);”

| Actual Output | Expected Output |
|---|---|
|  |  |
|  |  |
|  |  |


From: 0 sec
To: 60 sec
Marker
Cursor 48 sec
SST
TestBench
Type Signals
wire A
wire B
wire C
wire D
wire E
wire F
wire G
reg W
reg X
reg Y
reg Z
reg clk
Time
A
B
C
D
E
F
G
W
X
Y
Z
0
10 sec
20 sec
30 sec
40 sec
50 sec
60 sec