Submitted in partial fulfillment of the course requirements

For CSARCH1

Daniel Gavrie Y. Clemente

Jan Robee E. Feliciano

Derrick M. Valdellon

Laguna

February 6, 2024

# Table of Contents

**Table of Contents** 1

**Chapter 1: The Project and Its Background** 2

Introduction 2

**Chapter 2: Circuit Design** 3

Truth Table 3

Karnaugh Maps 4

Gate Naming Conventions 6

Logic Circuit 7

Waveforms 8

**Chapter 3: Verilog Observations** 9

**Reference List** 12

# Chapter 1: The Project and Its Background

## Introduction

Circuit design is a fascinating discipline that involves the creation of electronic circuits to enable the flow of electricity and perform specific functions. It is a combination of art and science, requiring a deep understanding of electrical principles and the ability to translate conceptual ideas into tangible circuit designs (Lathi & Ding, 2009). From simple analog circuits to complex digital systems, circuit design plays a crucial role in various industries, including telecommunications, automotive, aerospace, and consumer electronics (Sedra & Smith, 2019). Designers use specialized software tools to simulate and test circuit behavior before physical implementation, ensuring optimal performance and reliability (Horie, 2010).

In this paper, we will demonstrate how our circuit design lights up a seven-segment coded display and provide the methods in helping construct our circuit design. Then we will test the circuit design on Verilog and determine whether the circuit design passes all test cases in Verilog.

# Chapter 2: Circuit Design

## Truth Table

| W (2) | X (1) | Y (2) | Z (6) | No. | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1<br>0 | 0<br>0 | 0<br>1 | 0<br>0 | 2 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1<br>0 | 1<br>1 | 0<br>1 | 0<br>0 | 3 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 4 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 5 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 6 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 7 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0<br>1 | 0<br>0 | 1<br>0 | 1<br>1 | 8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0<br>1 | 1<br>1 | 1<br>0 | 1<br>1 | 9 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Karnaugh Maps**

f(A)(W, X, Y, Z) = W'X' + W'Z + XYZ' + WY'

| f(A) | Y,Z | | | | |
|---|---|---|---|---|---|
| W,X | | 00 | 01 | 11 | 10 |
| | 00 | 1 | 1 | 1 | 1 |
| | 01 | 0 | 1 | 1 | 1 |
| | 11 | 1 | 1 | 0 | 1 |
| | 10 | 1 | 1 | 0 | 0 |

f(B)(W, X, Y, Z) = W'Y + WY' + X'Z' + W'X

| f() | Y,Z | | | | |
|---|---|---|---|---|---|
| W,X | | 00 | 01 | 11 | 10 |
| | 00 | 1 | 0 | 1 | 1 |
| | 01 | 1 | 1 | 1 | 1 |
| | 11 | 1 | 1 | 0 | 0 |
| | 10 | 1 | 1 | 0 | 1 |

f(C)(W, X, Y, Z) = W'Y' + W'Z + Y'Z + WYZ' + XZ'

| f(C) | Y,Z | | | | |
|---|---|---|---|---|---|
| W,X | | 00 | 01 | 11 | 10 |
| | 00 | 1 | 1 | 1 | 0 |
| | 01 | 1 | 1 | 1 | 1 |
| | 11 | 1 | 1 | 0 | 1 |
| | 10 | 0 | 1 | 0 | 1 |

f(D)(W, X, Y, Z) = W'Y + WY' + W'X' + XYZ'

| f(D) | Y,Z | | | | |
|---|---|---|---|---|---|
| W,X | | 00 | 01 | 11 | 10 |
| | 00 | 1 | 1 | 1 | 1 |
| | 01 | 0 | 0 | 1 | 1 |
| | 11 | 1 | 1 | 0 | 1 |
| | 10 | 1 | 1 | 0 | 0 |

f(E)(W, X, Y, Z) = W'X' + X'Y'

| f(E) | Y,Z | | | | |
|---|---|---|---|---|---|
| W,X | | 00 | 01 | 11 | 10 |
| | 00 | 1 | 1 | 1 | 1 |
| | 01 | 0 | 0 | 0 | 0 |
| | 11 | 0 | 0 | 0 | 0 |
| | 10 | 1 | 1 | 0 | 0 |

f(F)(W, X, Y, Z) = W'X'Y' + W'YZ + WYZ' + WY'Z

| f(F) | Y,Z | | | | |
|---|---|---|---|---|---|
| W,X | | 00 | 01 | 11 | 10 |
| | 00 | 1 | 1 | 1 | 0 |
| | 01 | 0 | 0 | 1 | 0 |
| | 11 | 0 | 1 | 0 | 1 |
| | 10 | 0 | 1 | 0 | 1 |

$f(G)(W, X, Y, Z) = W'Y + WY' + W'X'Z + YZ'$

| f(G) | Y,Z | | | | |
|---|---|---|---|---|---|
| W,X | | 00 | 01 | 11 | 10 |
| | 00 | 0 | 1 | 1 | 1 |
| | 01 | 0 | 0 | 1 | 1 |
| | 11 | 1 | 1 | 0 | 1 |
| | 10 | 1 | 1 | 0 | 1 |

**Gate Naming Conventions**

XYZ’ = alpha

W’X’ = bravo

W’Z = charlie

WY’ = delta

X’Z’ = echo

W’X = foxtrot

W’Y = golf

WYZ’ = hotel

W’Y’ = india

Y’Z = juliet

XZ’ = kilo

WY’ = lima

X’Y’ = mike

W’X’Y’ = november

W’YZ = oscar

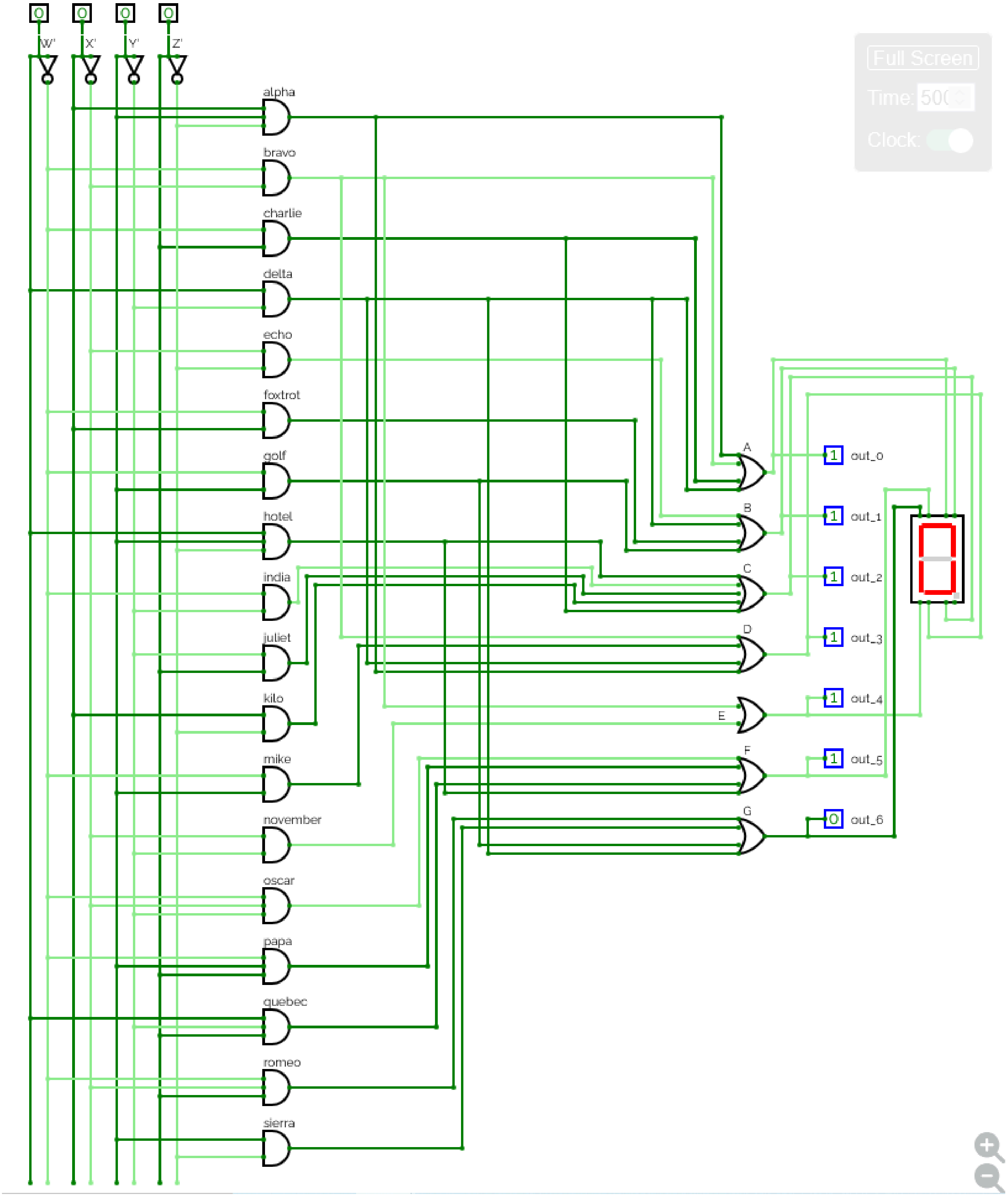
WY’Z = papa

W’X’Z = quebec

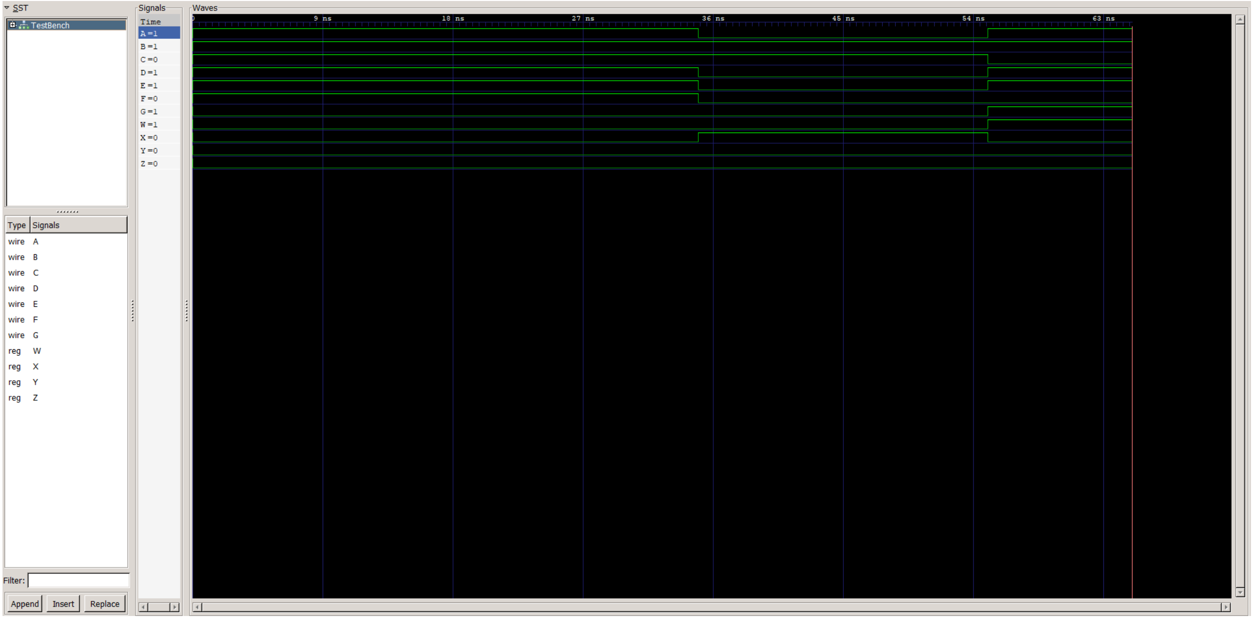
W’YZ = romeo

WY’Z = sierra

## Logic Circuit



## Waveforms



| Type | Signals |
|---|---|
| wire | A |
| wire | B |
| wire | C |
| wire | D |
| wire | E |
| wire | F |
| wire | G |
| reg | W |
| reg | X |
| reg | Y |
| reg | Z |

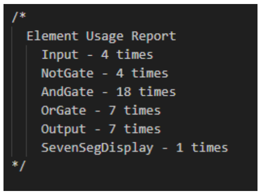
Before designing the circuit needed to light up the seven-segment display, we first needed to determine what are the truth table values needed for each number. First, we created a truth table based on the specifications with a 2126 binary value for each number. For example, in a 2126 truth table, the number 9 can either be represented as 0111 or 1101 and the number 7 is represented as 0101. After determining the truth value for each number, we then used a Karnaugh map in creating the logic for the circuit design since the output is a seven-segment display. Then after creating the Karaugh maps for each segment, we then start designing the circuit. Lastly, we then check the circuit design with a test bench to see the waveform of the circuit.

### Chapter 3: Verilog Observations

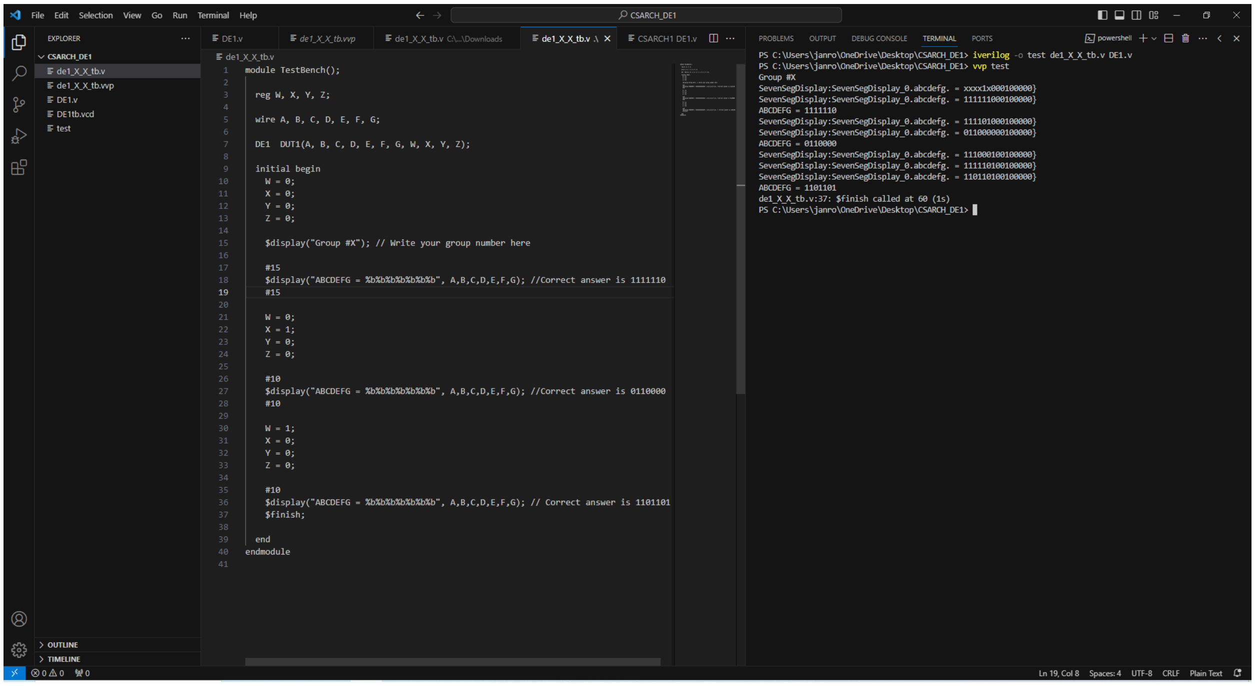
```
module DE1 (out_0, out_1, out_2, out_3, out_4, out_5, out_6, W, X, Y, Z);
  output out_0,  out_1,  out_2,  out_3,  out_4,  out_5,  out_6;
  input W, X, Y, Z;
  wire \Z'_out , and_17_out, G_out, and_10_out, C_out, and_7_out, F_out,
and_4_out, B_out, and_3_out, D_out, A_out, and_16_out, and_15_out,
and_14_out, and_9_out, and_1_out, \Y'_out , and_13_out, and_12_out, E_out,
and_8_out, and_2_out, and_11_out, and_6_out, \X'_out , and_0_out,
and_5_out, \W'_out ;
  assign \Z'_out  = ~Z;
  assign and_17_out = Y & \Z'_out ;
  assign G_out = and_16_out | and_17_out | and_6_out | and_2_out;
  assign out_6 = G_out;

      always @ (*)
      $display("SevenSegDisplay:SevenSegDisplay_0.abcdefg. =
%b%b%b%b%b%b%b}",
                  A_out, B_out, C_out, D_out, E_out, F_out, G_out, );
  assign and_10_out = X & \Z'_out ;
  assign C_out = and_7_out | and_8_out | and_9_out | and_10_out |
and_1_out;
  assign out_2 = C_out;
  assign and_7_out = W & Y & \Z'_out ;
  assign F_out = and_13_out | and_14_out | and_15_out | and_7_out;
  assign out_5 = F_out;
  assign and_4_out = \X'_out  & \Z'_out ;
  assign B_out = and_4_out | and_2_out | and_5_out | and_6_out;
  assign out_1 = B_out;
  assign and_3_out = X & Y & \Z'_out ;
  assign D_out = and_0_out | and_11_out | and_2_out | and_3_out;
  assign out_3 = D_out;
  assign A_out = and_3_out | and_0_out | and_1_out | and_2_out;
  assign out_0 = A_out;
  assign and_16_out = \W'_out  & \X'_out  & Z;
  assign and_15_out = W & \Y'_out  & Z;
  assign and_14_out = \W'_out  & Y & Z;
  assign and_9_out = \Y'_out  & Z;
  assign and_1_out = \W'_out  & Z;
  assign \Y'_out  = ~Y;
  assign and_13_out = \W'_out  & \X'_out  & \Y'_out ;
  assign and_12_out = \X'_out  & \Y'_out ;
  assign E_out = and_0_out | and_12_out;
  assign out_4 = E_out;
  assign and_8_out = \W'_out  & \Y'_out ;
  assign and_2_out = W & \Y'_out ;
  assign and_11_out = \W'_out  & Y;
  assign and_6_out = \W'_out  & Y;
  assign \X'_out  = ~X;
  assign and_0_out = \W'_out  & \X'_out ;
  assign and_5_out = \W'_out  & X;
  assign \W'_out  = ~W;
endmodule
```

Above is the DE1 module of the exported verilog file from CircuitVerse. The generated verilog file also provides the number of gates used in the logic circuit. As can be seen below, there are 29 gates in total.



```
/*
  Element Usage Report
    Input - 4 times
    NotGate - 4 times
    AndGate - 18 times
    OrGate - 7 times
    Output - 7 times
    SevenSegDisplay - 1 times
*/
```



```verilog
module TestBench();

  reg W, X, Y, Z;

  wire A, B, C, D, E, F, G;

  DE1  DUT1(A, B, C, D, E, F, G, W, X, Y, Z);

  initial begin
    W = 0;
    X = 0;
    Y = 0;
    Z = 0;

    $display("Group #X"); // Write your group number here

    #15
    $display("ABCDEFG = %b%b%b%b%b%b%b", A,B,C,D,E,F,G); //Correct answer is 1111110
    #15

    W = 0;
    X = 1;
    Y = 0;
    Z = 0;

    #10
    $display("ABCDEFG = %b%b%b%b%b%b%b", A,B,C,D,E,F,G); //Correct answer is 0110000
    #10

    W = 1;
    X = 0;
    Y = 0;
    Z = 0;

    #10
    $display("ABCDEFG = %b%b%b%b%b%b%b", A,B,C,D,E,F,G); // Correct answer is 1101101
    $finish;

  end
endmodule
```

```
PS C:\Users\janro\OneDrive\Desktop\CSARCH_DE1> iverilog -o test de1_X_X_tb.v DE1.v
PS C:\Users\janro\OneDrive\Desktop\CSARCH_DE1> vvp test
Group #X
SevenSegDisplay:SevenSegDisplay_0.abcdefg. = xxxx1x000100000}
SevenSegDisplay:SevenSegDisplay_0.abcdefg. = 111111000100000}
ABCDEFG = 1111110
SevenSegDisplay:SevenSegDisplay_0.abcdefg. = 111101000100000}
SevenSegDisplay:SevenSegDisplay_0.abcdefg. = 011000000100000}
ABCDEFG = 0110000
SevenSegDisplay:SevenSegDisplay_0.abcdefg. = 111000100100000}
SevenSegDisplay:SevenSegDisplay_0.abcdefg. = 111110100100000}
SevenSegDisplay:SevenSegDisplay_0.abcdefg. = 110110100100000}
ABCDEFG = 1101101
de1_X_X_tb.v:37: $finish called at 60 (1s)
PS C:\Users\janro\OneDrive\Desktop\CSARCH_DE1>
```

Above are the test results of the exported verilog file from CircuitVerse. As can be seen, the test came out successful.

## Reference List

Horie, M. (2010). Digital design and computer architecture. Morgan Kaufmann.

Lathi, B. P., & Ding, Z. (2009). Modern digital and analog communication systems. Oxford University Press.

Sedra, A. S., & Smith, K. C. (2019). Microelectronic circuits. Oxford University Press.