ELL201 Digital Electronics
Lab Report

Atif Anwer 2020EE10479

February 23, 2022

Verilog Assignment 1

1 Objectives

To realize the function $F = \sum(0, 1, 2, 5, 6, 8, 9, 11, 13, 14, 15)$:

1. Using one 8 to 1 multiplexer, and minimum additional gates.
2. Using two 4 to 1 multiplexers, and minimum additional gates.

2 Implementation Methodology

2.1 Truth Table of the given function

A	B	C	D	F
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Table 1: Truth Table of Function F

2.2 Implementation using a 8 to 1 Multiplexer

1. Considering inputs A,B,C as selector lines, the input lines can be combinational logic of D.

Selector Lines					Input Line
A	B	C	D	F	
0	0	0	0	1	1
0	0	0	1	1	
0	0	1	0	1	D’
0	0	1	1	0	
0	1	0	0	0	D
0	1	0	1	1	
0	1	1	0	1	D’
0	1	1	1	0	
1	0	0	0	1	1
1	0	0	1	1	
1	0	1	0	0	D
1	0	1	1	1	
1	1	0	0	0	D
1	1	0	1	1	
1	1	1	0	1	1
1	1	1	1	1	

Table 2: Truth Table of Selector and Input Lines

2. This can be represented with the following circuit diagram :

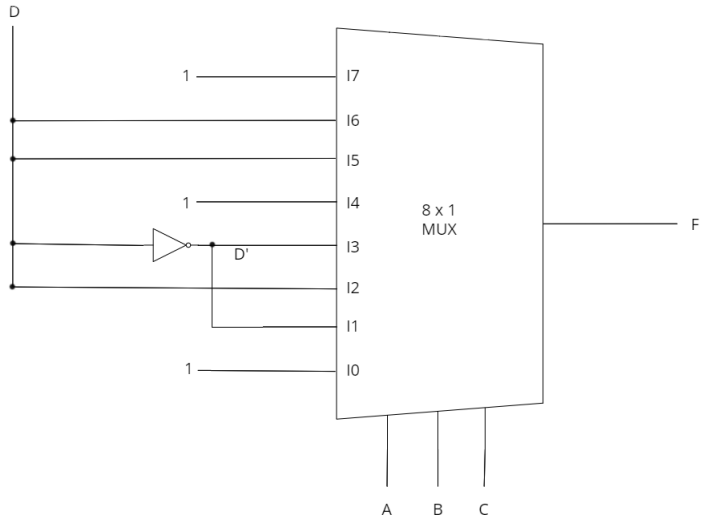

Figure 1: Implementation using a 8 to 1 Multiplexer

2.2.1 Code

```
module Combination
(
    input A,
    input B,
    input C,
    input D,
    output F
);
wire comp;
not(comp,D);
multiplexer8x1 M1(  .I0(1'b1),
                    .I1(comp),
                    .I2(D),
                    .I3(comp),
                    .I4(1'b1),
                    .I5(D),
                    .I6(D),
                    .I7(1'b1),
                    .S2(A),
                    .S1(B),
                    .S0(C),
                    .F(F));

endmodule
module multiplexer8x1
(
    input I0,
    input I1,
    input I2,
    input I3,
    input I4,
    input I5,
    input I6,
    input I7,
    input S2,
    input S1,
    input S0,
    output reg F
);
always @(I0,I1,I2,I3,I4,I5,I6,I7,S0,S1,S2)
begin
    if (S2) begin
        if (S1) begin
            if (S0) begin
                assign F = I7;
            end
            else begin
                assign F = I6;
            end
        end
        else begin
            if (S0) begin
                assign F = I5;
            end
            else begin
                assign F = I4;
```

```
            end
        end
    end
    else begin
        if (S1) begin
            if (S0) begin
                assign F = I3;
            end
            else begin
                assign F = I2;
            end
        end
        else begin
            if (S0) begin
                assign F = I1;
            end
            else begin
                assign F = I0;
            end
        end
    end
end
endmodule
```

2.2.2 Test Bench

```
module tb;

reg A,B,C,D;
wire F;
integer i;

Combination Circuit(.A(A),.B(B),.C(C),.D(D),.F(F));

initial begin
$dumpfile("tb.vcd");
$dumpvars(0,tb);
    A<=0;
    B<=0;
    C<=0;
    D<=0;

    $monitor("A = %0b B = %0b C = %0b D = %0b    F = %0b",A,B,C,D,F);
    for (i =0 ;i<16 ;i = i+1 ) begin
        {A,B,C,D} = i;
        #10;
    end
end
endmodule
```

2.2.3 vvp execution output

```
VCD info: dumpfile tb.vcd opened for output.
A = 0 B = 0 C = 0 D = 0   F = 1
A = 0 B = 0 C = 0 D = 1   F = 1
A = 0 B = 0 C = 1 D = 0   F = 1
A = 0 B = 0 C = 1 D = 1   F = 0
A = 0 B = 1 C = 0 D = 0   F = 0
A = 0 B = 1 C = 0 D = 1   F = 1
A = 0 B = 1 C = 1 D = 0   F = 1
A = 0 B = 1 C = 1 D = 1   F = 0
A = 1 B = 0 C = 0 D = 0   F = 1
A = 1 B = 0 C = 0 D = 1   F = 1
A = 1 B = 0 C = 1 D = 0   F = 0
A = 1 B = 0 C = 1 D = 1   F = 1
A = 1 B = 1 C = 0 D = 0   F = 0
A = 1 B = 1 C = 0 D = 1   F = 1
A = 1 B = 1 C = 1 D = 0   F = 1
A = 1 B = 1 C = 1 D = 1   F = 1
```

2.2.4 Waveform on GTKwave

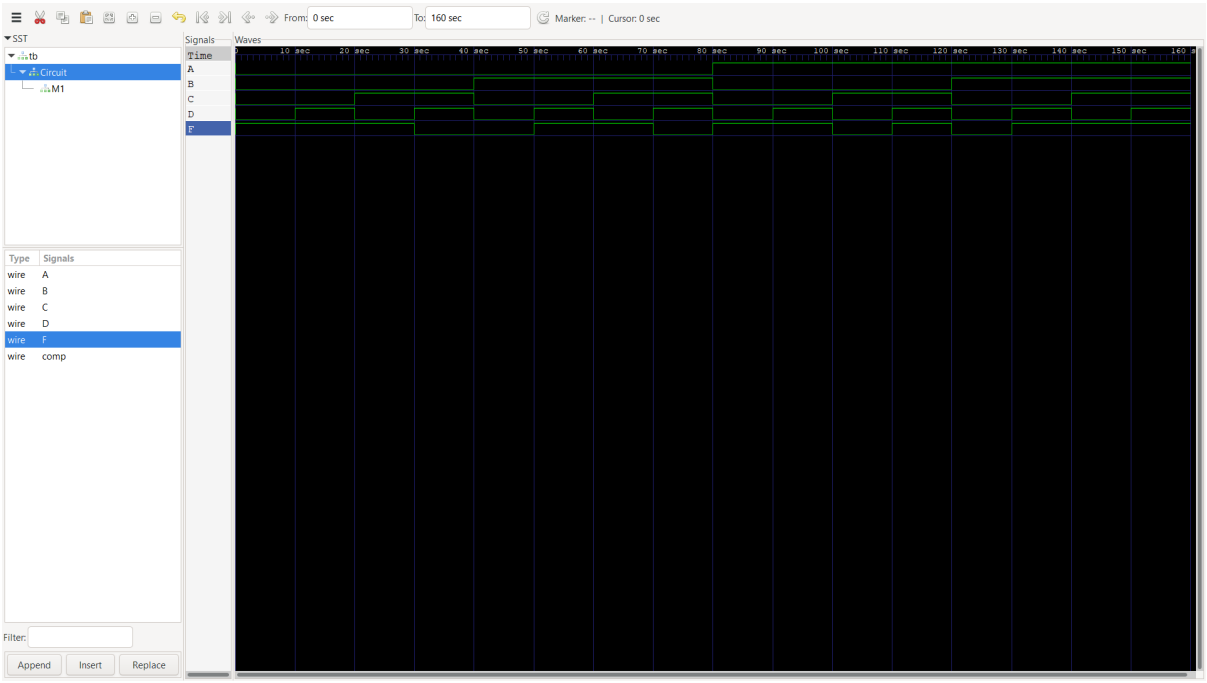

Figure 2: Waveform of simulation

2.3 Implementation using two 4 to 1 Multiplexers

1. We know that a 8 to 1 multiplexer's functionality can be implemented using two 4 to 1 multiplexers (with active high enabling) as shown below :

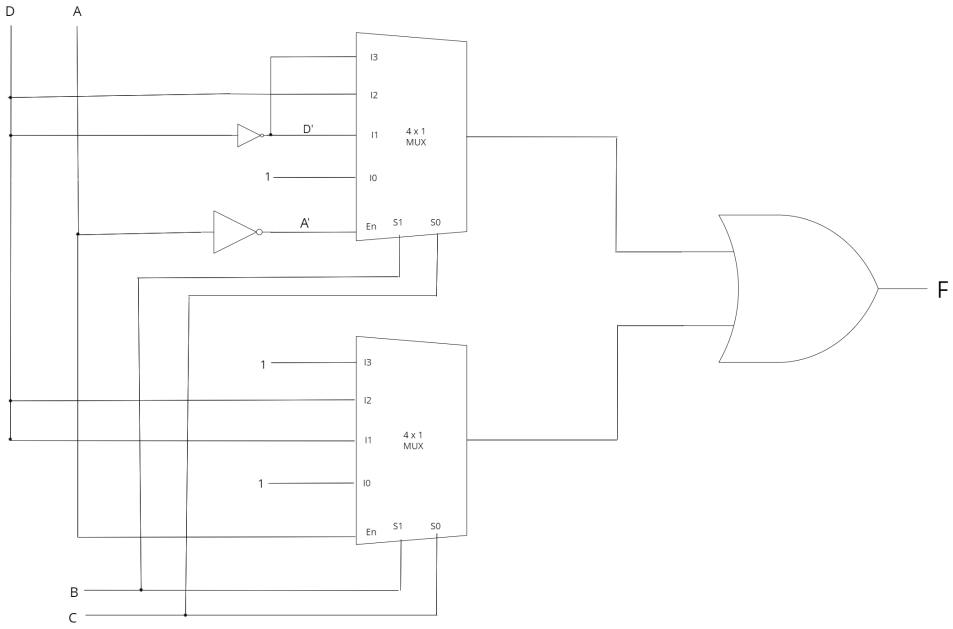

Figure 3: Implementation using two 4 to 1 Multiplexers

2. The idea behind this implementation is that if we consider the same Truth table 2 and connecting the same input lines I_0, I_1, I_2, I_3 to multiplexer 1 and input lines I_4, I_5, I_6, I_7 to multiplexer 2 as given in Figure 3 but connecting the enable signal input of multiplexer 1 to A' and multiplexer 2 to A and equating F to the OR of their outputs provides the same functionality as an 8 to 1 multiplexer with input lines $I_0, I_1 ... I_7$ and three selector lines A, B and C.

2.3.1 Code

```
module Combination
(
    input A,
    input B,
    input C,
    input D,
    output F
);
wire Dcomp,m1,m2,Acomp;
not(Acomp,A);
not(Dcomp,D);
multiplexer4x1 M1(  .I0(1'b1),
                    .I1(Dcomp),
                    .I2(D),
                    .I3(Dcomp),
                    .En(Acomp),
                    .S1(B),
                    .S0(C),
                    .F(m1));

```

```
multiplexer4x1 M2(   .I0(1'b1),
                     .I1(D),
                     .I2(D),
                     .I3(1'b1),
                     .En(A),
                     .S1(B),
                     .S0(C),
                     .F(m2));
or(F,m1,m2);

endmodule
module multiplexer4x1
(
    input I0,
    input I1,
    input I2,
    input I3,
    input En,
    input S1,
    input S0,
    output reg F
);
always @(I0,I1,I2,I3,S0,S1)
begin
    if(En) begin
        if (S1) begin
            if (S0) begin
                assign F = I3;
            end
            else begin
                assign F = I2;
            end
        end
        else begin
            if (S0) begin
                assign F = I1;
            end
            else begin
                assign F = I0;
            end
        end
    end
    else begin
        assign F = 1'b0;
    end
end
endmodule
```

2.3.2 Test Bench

```
module tb;

reg A,B,C,D;
wire F;
integer i;

Combination Circuit(.A(A),.B(B),.C(C),.D(D),.F(F));

initial begin
```

```
10 $dumpfile("tb.vcd");
11 $dumpvars(0,tb);
12     A<=0;
13     B<=0;
14     C<=0;
15     D<=0;
16
17     $monitor("A = %0b B = %0b C = %0b D = %0b    F = %0b",A,B,C,D,F);
18     for (i =0 ;i<16 ;i = i+1 ) begin
19         {A,B,C,D} = i;
20         #10;
21     end
22 end
23 endmodule
```

2.3.3 vvp execution output

```
VCD info: dumpfile tb.vcd opened for output.
A = 0 B = 0 C = 0 D = 0   F = 1
A = 0 B = 0 C = 0 D = 1   F = 1
A = 0 B = 0 C = 1 D = 0   F = 1
A = 0 B = 0 C = 1 D = 1   F = 0
A = 0 B = 1 C = 0 D = 0   F = 0
A = 0 B = 1 C = 0 D = 1   F = 1
A = 0 B = 1 C = 1 D = 0   F = 1
A = 0 B = 1 C = 1 D = 1   F = 0
A = 1 B = 0 C = 0 D = 0   F = 1
A = 1 B = 0 C = 0 D = 1   F = 1
A = 1 B = 0 C = 1 D = 0   F = 0
A = 1 B = 0 C = 1 D = 1   F = 1
A = 1 B = 1 C = 0 D = 0   F = 0
A = 1 B = 1 C = 0 D = 1   F = 1
A = 1 B = 1 C = 1 D = 0   F = 1
A = 1 B = 1 C = 1 D = 1   F = 1
```

2.3.4 Waveform on GTKwave

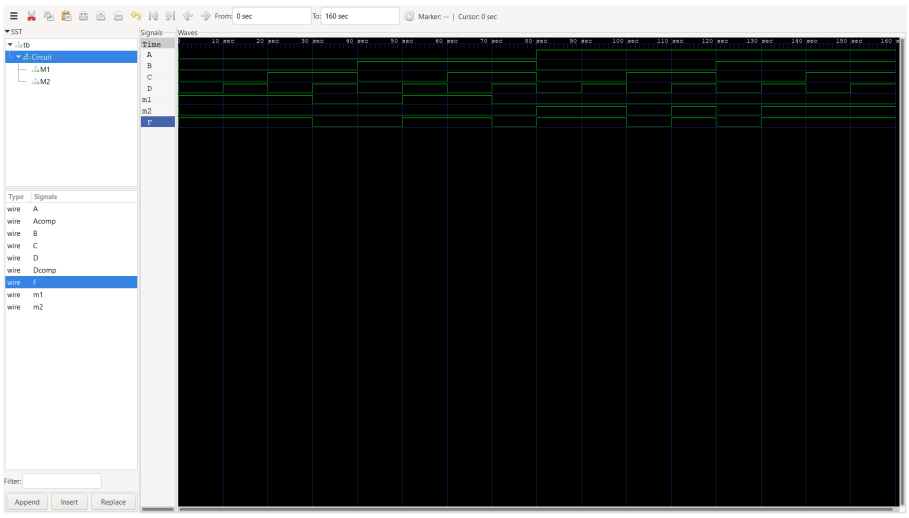

Figure 4: Waveform of simulation