

ELP201 Lab Report

Adit Malhotra

2020EE10458

1 Truth Table

Given function $\rightarrow F = \sum(0, 1, 2, 5, 6, 8, 9, 11, 13, 14, 15)$

A	B	C	D	Output (F)	Minterm
0	0	0	0	1	m_0
0	0	0	1	1	m_1
0	0	1	0	1	m_2
0	0	1	1	0	m_3
0	1	0	0	0	m_4
0	1	0	1	1	m_5
0	1	1	0	1	m_6
0	1	1	1	0	m_7
1	0	0	0	1	m_8
1	0	0	1	1	m_9
1	0	1	0	0	m_{10}
1	0	1	1	1	m_{11}
1	1	0	0	0	m_{12}
1	1	0	1	1	m_{13}
1	1	1	0	1	m_{14}
1	1	1	1	1	m_{15}

2 Implementation of the function

2.1 Using one 8x1 MUX

Since there are three selection lines available for the 8x1 MUX, thus three of the input variables (B, C, D) can be used as selection lines of the MUX. Now depending on the type of minterms in the function, $A, \bar{A}, 1, 0$ will be used as appropriate input lines to the MUX. Exact circuit diagram is given below:

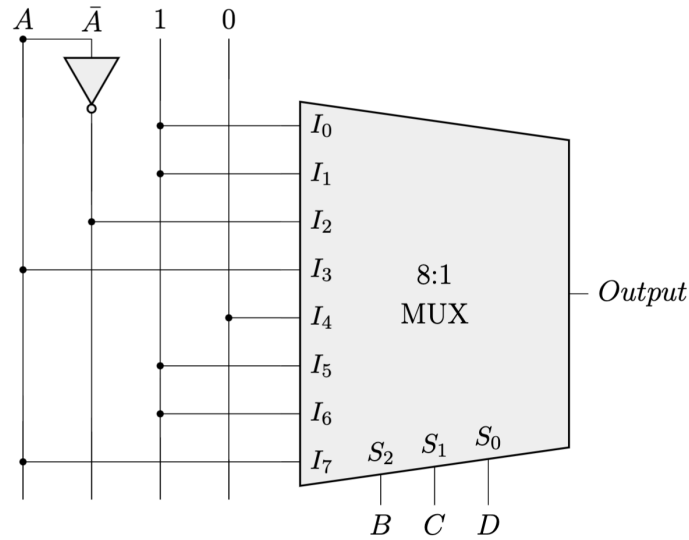


Figure 1: Realisation of the function using a 8×1 MUX

2.2 Using two 4x1 MUX

In case of two 4:1 multiplexers, two of the input variables (D and C) will be used as selection lines of the two joint multiplexers, and one variable can be used as *Enable Line* with direct line to one of the mux and with a NOT gate to other mux, so that only one of the mux is active at a moment. Finally the OR of the outputs from the muxes would result in output. In this way we have combined two 4:1 muxes into a single 8:1 mux.

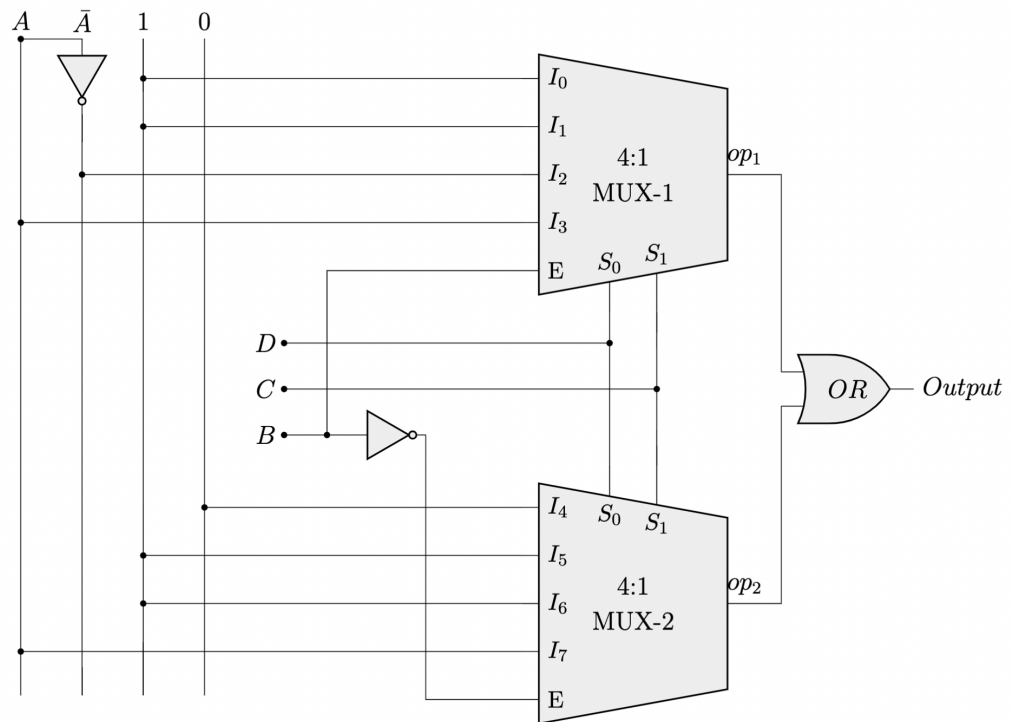


Figure 2: Realisation of the function using two 4×1 MUX

3 Verilog part

3.1 Simplification using K-Maps

By marking the minterms on the K-Map, the implicants come out to be AD , $\bar{B}\bar{C}$, $\bar{C}D$, $\bar{A}C\bar{D}$, $BC\bar{D}$.

Thus the simplified expression for F is:

$$F = AD + \bar{B}\bar{C} + \bar{C}D + \bar{A}C\bar{D} + BC\bar{D}$$

cd \ ab	00	01	11	10
00	1	1	0	1
01	0	1	0	1
11	0	1	1	1
10	1	1	1	0

Figure 3: K-Map for the function

3.2 Code

The code for file containing the implementation of 8×1 mux, 4×1 mux and realisation of the function using k-maps (design.v) :

```
1 module kmap(A,B,C,D,out);
2   input A,B,C,D;
3   output out;
4   assign out = (A & D) | (~B & ~C) | (~C & D) | (~A & C & ~D) | (B & C & ~D);
5 endmodule
6
7 module mux8x1(i0,i1,i2,i3,i4,i5,i6,i7,sel0,sel1,sel2,out);
8   input i0,i1,i2,i3,i4,i5,i6,i7,sel0,sel1,sel2;
9   output reg out;
10  always@(*)
11  begin
12    if (~sel2 & ~sel1 & ~sel0)
13      out=i0;
14    else if (~sel2 & ~sel1 & sel0)
15      out=i1;
16    else if (~sel2 & sel1 & ~sel0)
17      out=i2;
18    else if (~sel2 & sel1 & sel0)
19      out=i3;
20    else if (sel2 & ~sel1 & ~sel0)
21      out=i4;
22    else if (sel2 & ~sel1 & sel0)
23      out=i5;
24    else if (sel2 & sel1 & ~sel0)
25      out=i6;
```

```

26     else if (sel2 & sel1 & sel0)
27         out=i7;
28     end
29 endmodule

30
31 module mux4x1(i0,i1,i2,i3,sel0,sel1,E,out); //E is enable line
32 input i0,i1,i2,i3,sel0,sel1,E;
33 output reg out;
34 always@(*)
35     begin
36         if(E)
37             out=0; //if the mux is not active then giving 0 output
38         else if (~sel1 & ~sel0)
39             out=i0;
40         else if (~sel1 & sel0)
41             out=i1;
42         else if (sel1 & ~sel0)
43             out=i2;
44         else if (sel1 & sel0)
45             out=i3;
46         end
47 endmodule

48
49 module ORgate(in1,in2,out);
50 input in1,in2;
51 output out;
52 assign out=in1|in2;
53 endmodule

```

The code for the testbench file (main.v), the wire $y1$ is the output obtained when the function is realised using only 8×1 *mux*, the wire $y2$ is the output obtained when the function is realised using two 4×1 *mux* and finally wire F is output of the function when realised using kmaps :

```

1  `timescale 1s/100ms
2  `include "design.v" //inlcuding the gates implementation file
3
4  module main(); //module of the testbench starts
5  reg a,b,c,d;
6  wire F; wire y1,y2; wire out1,out2;
7
8  //instantitation the design module and passing the parameters
9  // case a:
10 mux8x1 func1(1,1,~a,a,0,1,1,a,d,c,b,y1); // 8x1 MUX

```

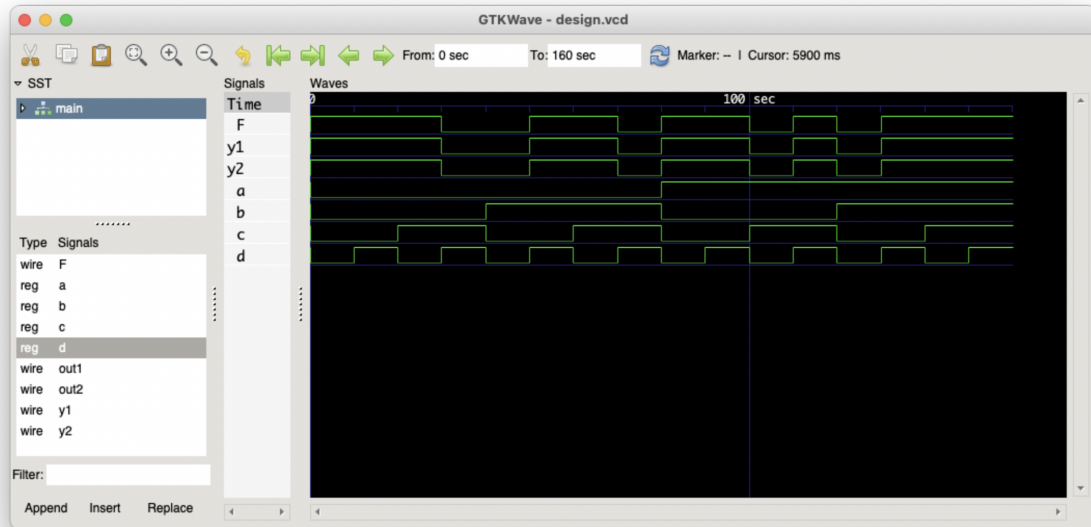
```

11
12 // case b:
13 mux4x1 func2(1,1,~a,a,d,c,b,out1); //4x1 MUX-1
14 mux4x1 func3(0,1,1,a,d,c,~b,out2); // 4x1 MUX-2
15 ORgate func4(out1,out2,y2);
16
17 //Realising function using k-maps
18 kmap func5(a,b,c,d,F);
19
20 initial
21 begin
22     $monitor("a=%b,b=%b,c=%b,d=%b,y1=%b,y2=%b,F=%b",a,b,c,d,y1,y2,F);
23     $dumpfile("design.vcd");
24     $dumpvars(0,main);
25
26     a=0;b=0;c=0;d=0;#10;
27     a=0;b=0;c=0;d=1;#10;
28     a=0;b=0;c=1;d=0;#10;
29     a=0;b=0;c=1;d=1;#10;
30
31     a=0;b=1;c=0;d=0;#10;
32     a=0;b=1;c=0;d=1;#10;
33     a=0;b=1;c=1;d=0;#10;
34     a=0;b=1;c=1;d=1;#10;
35
36     a=1;b=0;c=0;d=0;#10;
37     a=1;b=0;c=0;d=1;#10;
38     a=1;b=0;c=1;d=0;#10;
39     a=1;b=0;c=1;d=1;#10;
40
41     a=1;b=1;c=0;d=0;#10;
42     a=1;b=1;c=0;d=1;#10;
43     a=1;b=1;c=1;d=0;#10;
44     a=1;b=1;c=1;d=1;#10;
45
46     $finish;
47 end
48 endmodule

```

3.3 The waveform output

Thus it can be observed that output of $y1$ and $y2$ is exactly same as F , hence we have correctly implemented the function using the multiplexers in both the cases.



The code files can be downloaded from [here](#).