

Model Sim

LAB # 13



Fall 2020

CSE-304L Computer Organization and Architecture Lab

Submitted by: **Shah Raza**

Registration No. : **18PWCSE1658**

Class Section: **B**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Student Signature: _____

Submitted to:

Engr. Amaad Khalil

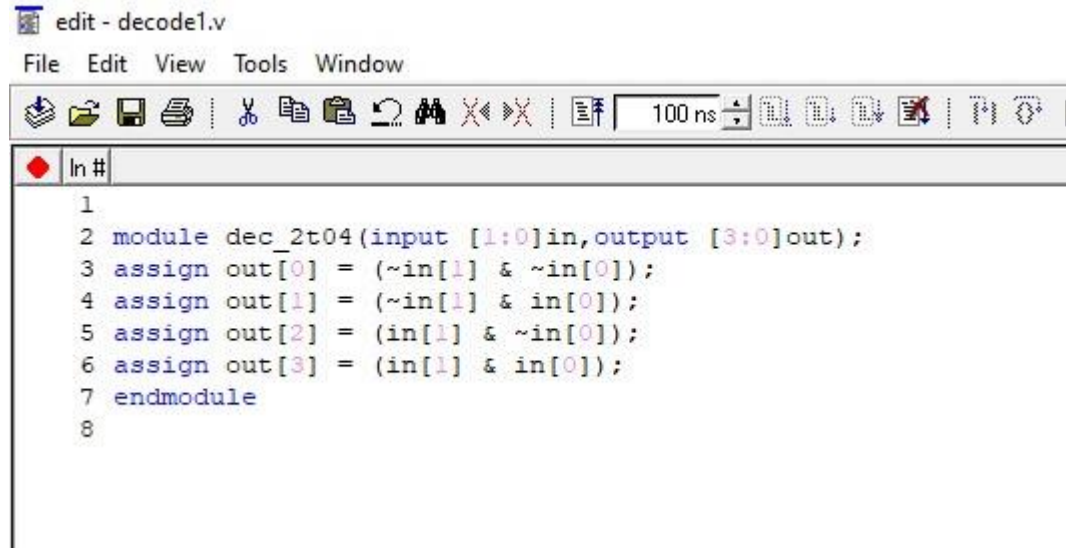
March 19, 2021

Department of Computer Systems Engineering
University of Engineering and Technology, Peshawar

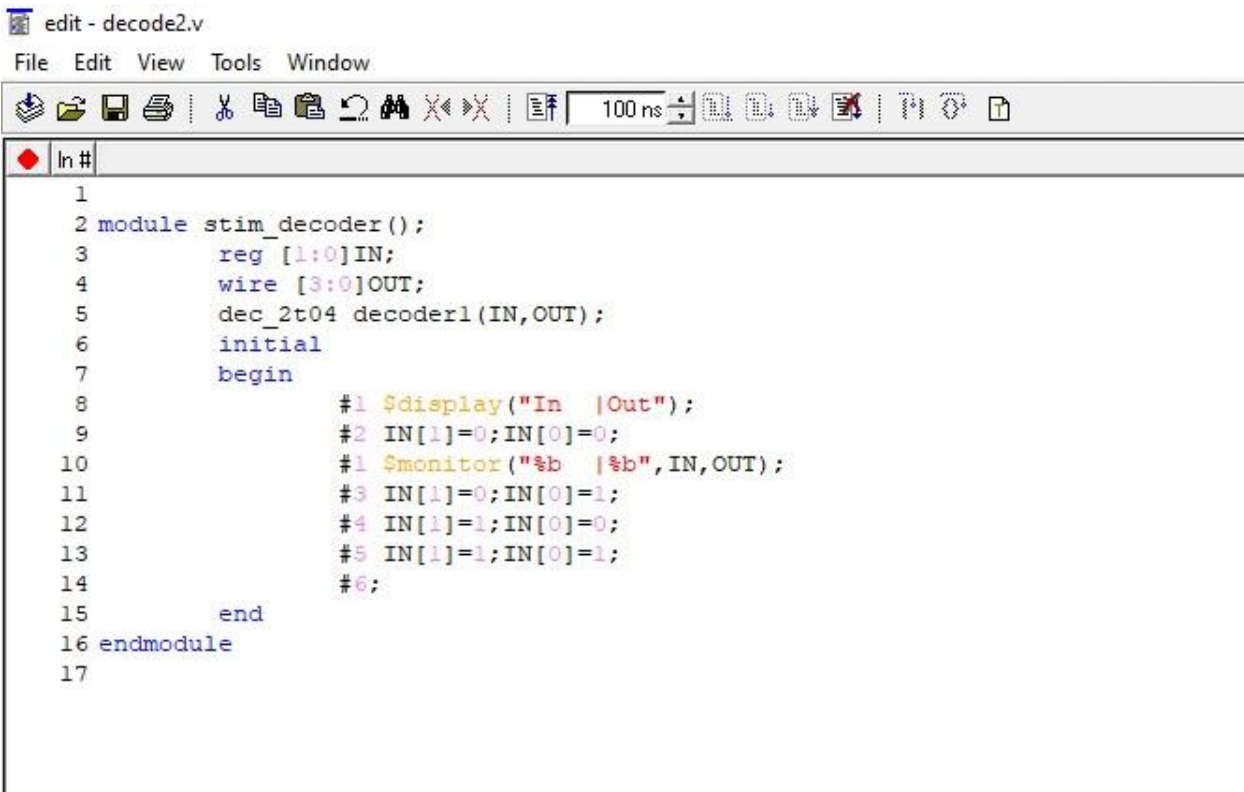
TASK01:

Write a Verilog code for 2x4 Decoder using Dataflow Level modeling.

Code:

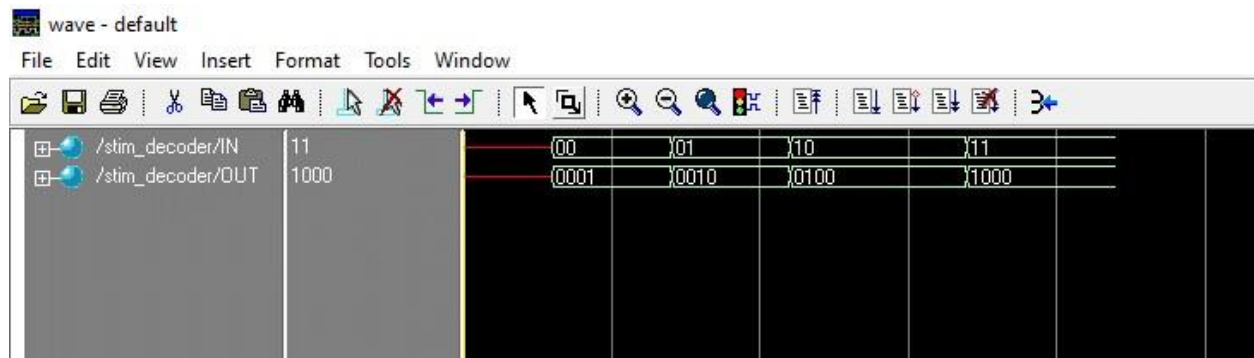


```
1
2 module dec_2t04(input [1:0]in,output [3:0]out);
3   assign out[0] = (~in[1] & ~in[0]);
4   assign out[1] = (~in[1] & in[0]);
5   assign out[2] = (in[1] & ~in[0]);
6   assign out[3] = (in[1] & in[0]);
7 endmodule
8
```



```
1
2 module stim_decoder();
3     reg [1:0]IN;
4     wire [3:0]OUT;
5     dec_2t04 decoder1(IN,OUT);
6     initial
7     begin
8         #1 $display("In |Out");
9         #2 IN[1]=0;IN[0]=0;
10        #1 $monitor("%b |%b",IN,OUT);
11        #3 IN[1]=0;IN[0]=1;
12        #4 IN[1]=1;IN[0]=0;
13        #5 IN[1]=1;IN[0]=1;
14        #6;
15    end
16 endmodule
17
```

Output Wave:



TASK02:

Write a Verilog code for 2x1 MUX using Dataflow Level modeling.

Code:

```
edit - mux_2to1.v
File Edit View Tools Window
100 ns
In #
1
2 module mux_2to1(input [2:0]IN, output O);
3     assign x=(IN[0] & ~IN[2]);
4     assign y=(IN[1] & IN[2]);
5     assign O=(x | y);
6 endmodule
```

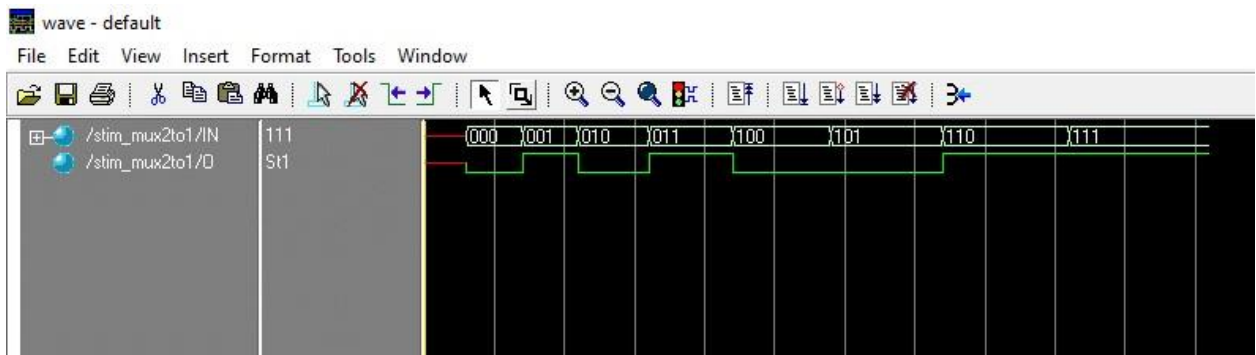
edit - mux_2to1b.v

File Edit View Tools Window

100 ns

```
1
2 module stim_mux2to1;
3     reg [2:0] IN;
4     wire O;
5     mux_2to1 mux_2l(IN, O);
6     initial
7     begin
8         #1 $display("INPUT | OUTPUT");
9         #2 IN[2]=0; IN[1]=0; IN[0]=0;
10        #1 $monitor("%b | %b", IN, O);
11        #3 IN[2]=0; IN[1]=0; IN[0]=1;
12        #4 IN[2]=0; IN[1]=1; IN[0]=0;
13        #5 IN[2]=0; IN[1]=1; IN[0]=1;
14        #6 IN[2]=1; IN[1]=0; IN[0]=0;
15        #7 IN[2]=1; IN[1]=0; IN[0]=1;
16        #8 IN[2]=1; IN[1]=1; IN[0]=0;
17        #9 IN[2]=1; IN[1]=1; IN[0]=1;
18        #10;
19    end
20 endmodule
21
22
```

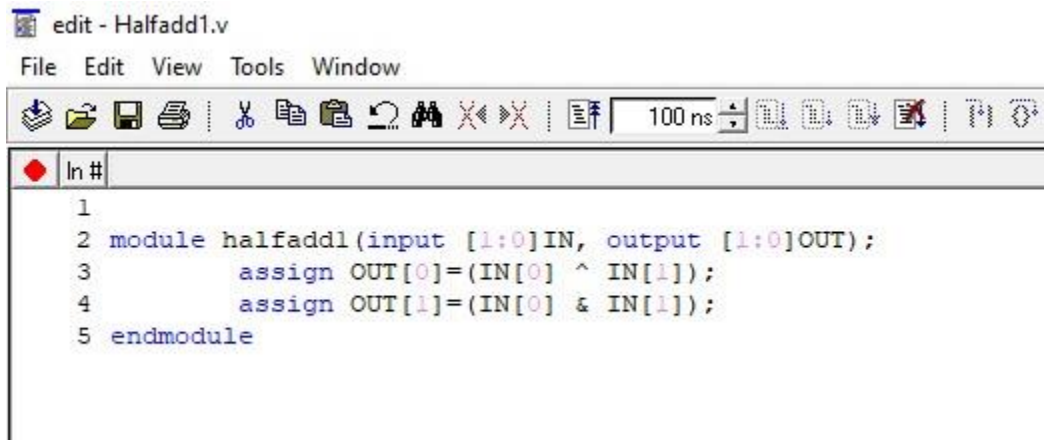
Output Wave:



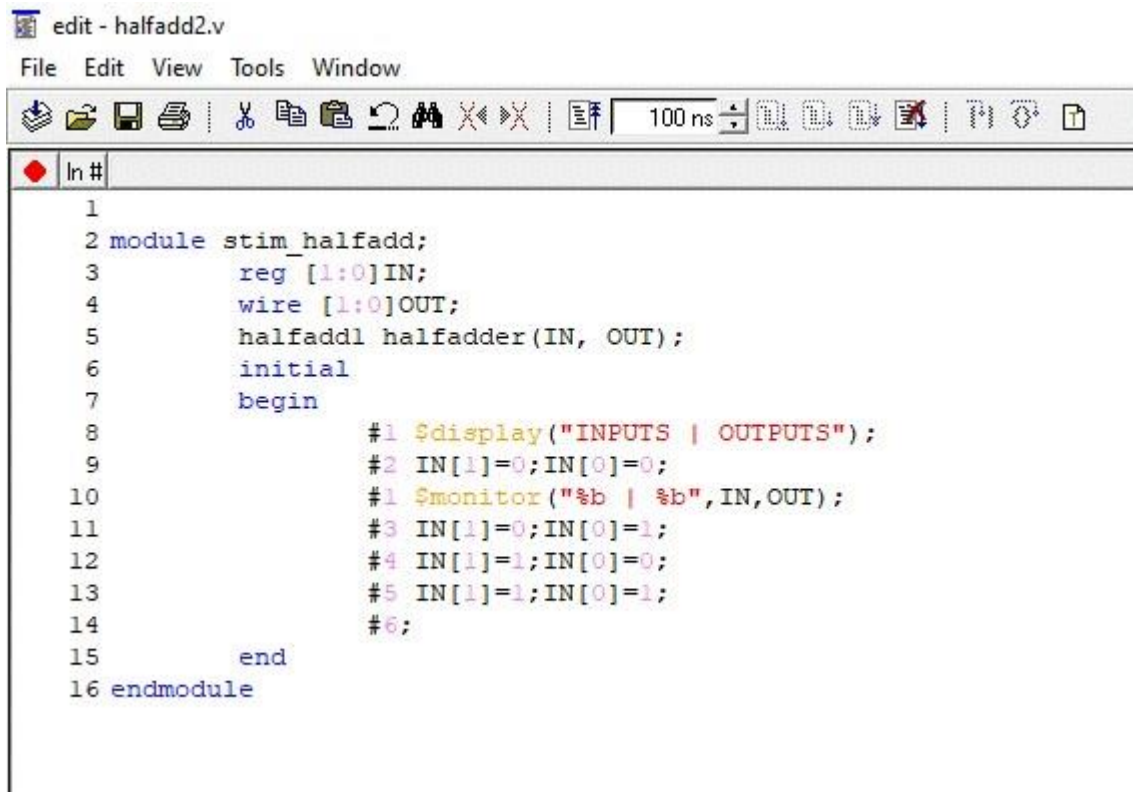
TASK03:

Write a Verilog code for Half Adder using Dataflow Level modeling.

Code:

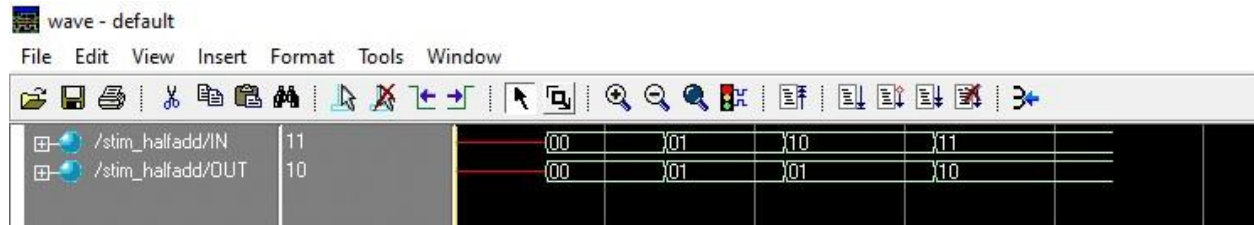


```
1
2 module halfadd1(input [1:0]IN, output [1:0]OUT);
3     assign OUT[0]=(IN[0] ^ IN[1]);
4     assign OUT[1]=(IN[0] & IN[1]);
5 endmodule
```



```
1
2 module stim_halfadd;
3     reg [1:0]IN;
4     wire [1:0]OUT;
5     halfadder halfadder(IN, OUT);
6     initial
7     begin
8         #1 $display("INPUTS | OUTPUTS");
9         #2 IN[1]=0;IN[0]=0;
10        #1 $monitor("%b | %b", IN, OUT);
11        #3 IN[1]=0;IN[0]=1;
12        #4 IN[1]=1;IN[0]=0;
13        #5 IN[1]=1;IN[0]=1;
14        #6;
15    end
16 endmodule
```

Output Wave:



TASK04:

Write a Verilog code for Full Adder using Dataflow Level modeling.

Code:

The screenshot shows a Verilog code editor window titled "edit - fulladd1.v". The menu bar includes File, Edit, View, Tools, and Window. The toolbar contains various icons for file operations, editing, and viewing. The main area displays the Verilog code for a Full Adder using Dataflow Level modeling.

```
1
2 module fulladder(input [2:0]IN, output [1:0]OUT);
3     assign x=(IN[0] ^ IN[1]);
4     assign y=(IN[2] & x);
5     assign z=(IN[0] & IN[1]);
6     assign OUT[0]=(IN[2] ^ x);
7     assign OUT[1]=(y | z);
8 endmodule
```

```

edit - fulladd2.v
File Edit View Tools Window
100 ns
In #
1
2 module stim_fulladd;
3     reg [2:0] IN;
4     wire [1:0] OUT;
5     fulladder addfull(IN, OUT);
6     initial
7     begin
8         #1 $display("INPUTS | OUTPUTS");
9         #2 IN[2]=0; IN[1]=0; IN[0]=0;
10        #1 $monitor("%b | %b", IN, OUT);
11        #3 IN[2]=0; IN[1]=0; IN[0]=1;
12        #4 IN[2]=0; IN[1]=1; IN[0]=0;
13        #5 IN[2]=0; IN[1]=1; IN[0]=1;
14        #6 IN[2]=1; IN[1]=0; IN[0]=0;
15        #7 IN[2]=1; IN[1]=0; IN[0]=1;
16        #8 IN[2]=1; IN[1]=1; IN[0]=0;
17        #9 IN[2]=1; IN[1]=1; IN[0]=1;
18        #10;
19    end
20 endmodule
21

```

Output Wave:

wave - default

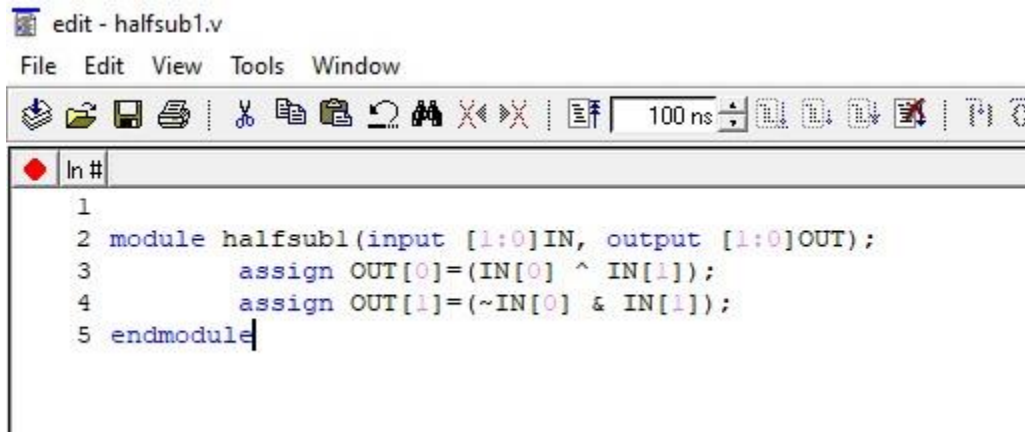
File Edit View Insert Format Tools Window

	/stim_fulladd/IN	/stim_fulladd/OUT
000	001	010
001	010	011
010	011	100
011	100	101
100	101	110
101	110	111
110	111	
111		

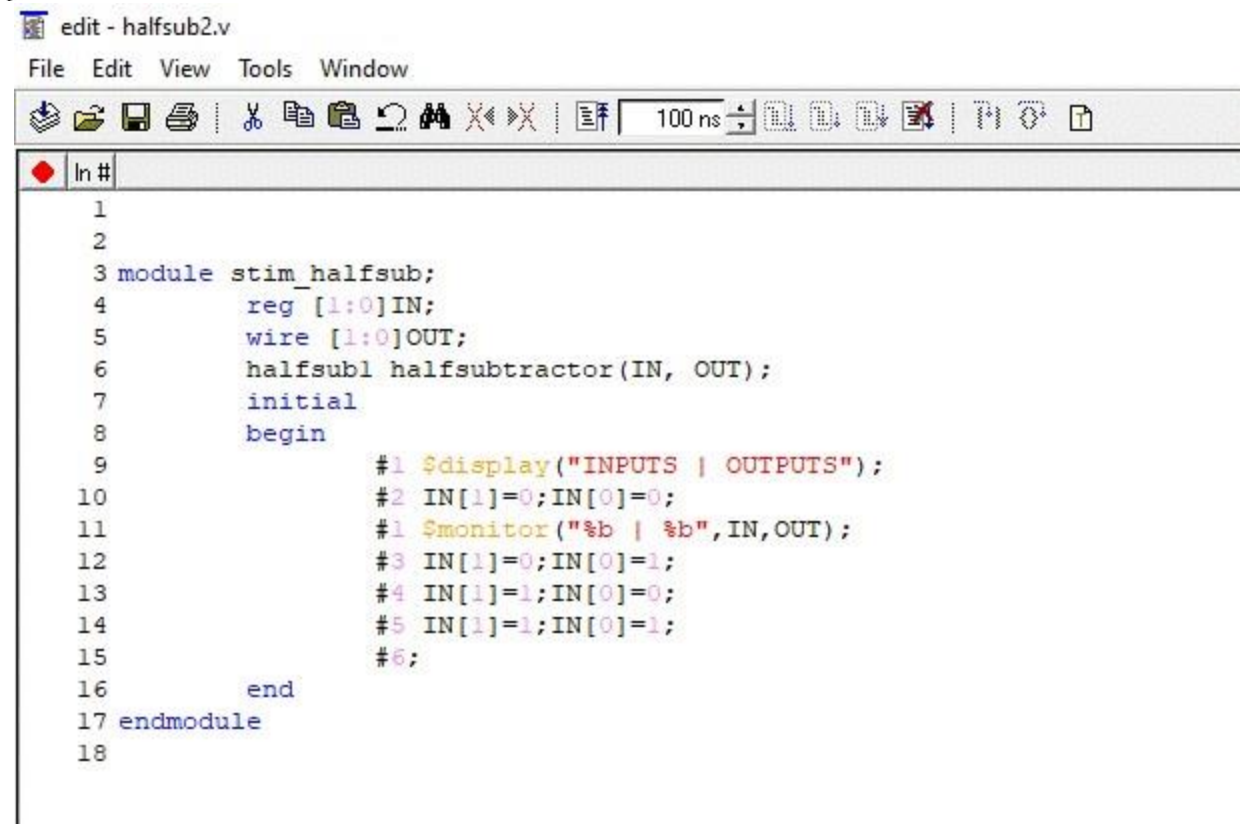
TASK05:

Write a Verilog code for Half Subtractor using Dataflow Level modeling.

Code:



```
1
2 module halfsub1(input [1:0]IN, output [1:0]OUT);
3     assign OUT[0]=(IN[0] ^ IN[1]);
4     assign OUT[1]=(~IN[0] & IN[1]);
5 endmodule
```



```
1
2
3 module stim_halfsub;
4     reg [1:0]IN;
5     wire [1:0]OUT;
6     halfsub1 halfsubtractor(IN, OUT);
7     initial
8     begin
9         #1 $display("INPUTS | OUTPUTS");
10        #2 IN[1]=0;IN[0]=0;
11        #1 $monitor("%b | %b",IN,OUT);
12        #3 IN[1]=0;IN[0]=1;
13        #4 IN[1]=1;IN[0]=0;
14        #5 IN[1]=1;IN[0]=1;
15        #6;
16    end
17 endmodule
18
```

Output Wave:

