Digital Design and Computer Organisation Laboratory

UE22CS251A

3rd Semester, Academic Year 2023

Name: **ASHLESHA.T**	SRN: **PES1UG22CS118**	Section **B**

Week : 3 program: 1

DESIGN OF 2x1 MUX:

Source code:

mux2.v

```
module mux2(input wire d0,d1,s,output wire y);
 assign y=(s==0)?d0:d1;
endmodule
```

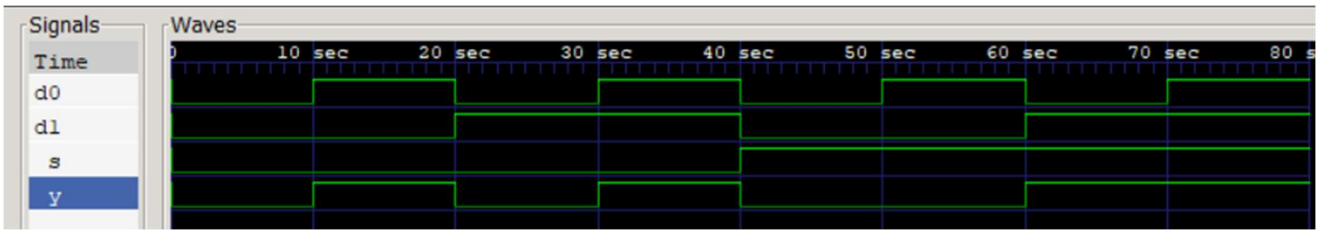
mux2_1_tb.v

```
module mux2_1_tb;
 reg d1,d0,s;
 wire y;
 mux2 mux2_1(d0,d1,s,y);
 initial begin
  d1=0;d0=0;s=0; #10;
  d1=0;d0=1;s=0; #10;
  d1=1;d0=0;s=0; #10;
  d1=1;d0=1;s=0; #10;
  d1=0;d0=0;s=1; #10;
  d1=0;d0=1;s=1; #10;
  d1=1;d0=0;s=1; #10;
  d1=1;d0=1;s=1; #10;
 end
initial begin
    $monitor($time," d1=%b d0=%b  s=%b   y=%b",d1,d0,s,y);
end
initial begin
 $dumpfile("mux12_test.vcd");
 $dumpvars(0,mux2_1_tb);
end
endmodule
```

VVP OUTPUT:

```
PS C:\iverilog\bin> iverilog -o test mux2.v mux2_1_tb.v
PS C:\iverilog\bin> vvp test
VCD info: dumpfile mux12_test.vcd opened for output.
                   0 d1=0 d0=0  s=0   y=0
                  10 d1=0 d0=1  s=0   y=1
                  20 d1=1 d0=0  s=0   y=0
                  30 d1=1 d0=1  s=0   y=1
                  40 d1=0 d0=0  s=1   y=0
                  50 d1=0 d0=1  s=1   y=0
                  60 d1=1 d0=0  s=1   y=1
                  70 d1=1 d0=1  s=1   y=1
```

GTK WAVE:



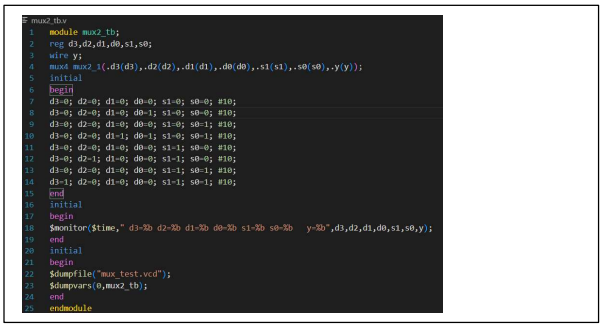
Name: **ASHLESHA.T**	SRN: **PES1UG22CS118**	Section **B**

Program: 2

4x1 MUX using 2x1:

mux2.v

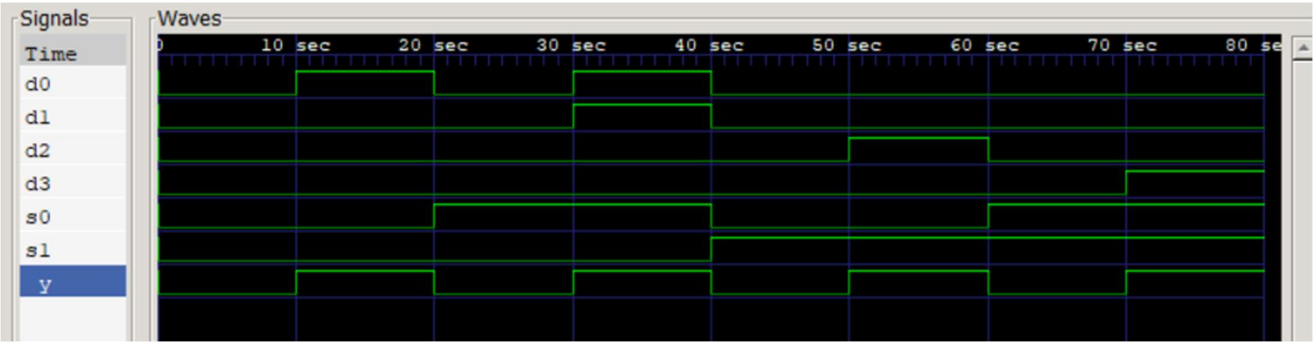
```
1   module mux2(input wire d0,d1,s,output wire y);
2    assign y=(s==0)?d0:d1;
3   endmodule
4   module mux4(input wire d3,d2,d1,d0,s1,s0,output wire y);
5   wire [1:0]x;
6   mux2 mux2_1(.d0(d0),.d1(d1),.s(s0),.y(x[0]));
7   mux2 mux2_2(.d0(d2),.d1(d3),.s(s0),.y(x[1]));
8   mux2 mux2_3(.d0(x[0]),.d1(x[1]),.s(s1),.y(y));
9   endmodule
```



VVP OUTPUT:

```
PS C:\iverilog\bin> iverilog -o test mux2.v mux2_tb.v
PS C:\iverilog\bin> vvp test
VCD info: dumpfile mux_test.vcd opened for output.
                   0 d3=0 d2=0 d1=0 d0=0 s1=0 s0=0   y=0
                  10 d3=0 d2=0 d1=0 d0=1 s1=0 s0=0   y=1
                  20 d3=0 d2=0 d1=0 d0=0 s1=0 s0=1   y=0
                  30 d3=0 d2=0 d1=1 d0=1 s1=0 s0=1   y=1
                  40 d3=0 d2=0 d1=0 d0=0 s1=1 s0=0   y=0
                  50 d3=0 d2=1 d1=0 d0=0 s1=1 s0=0   y=1
                  60 d3=0 d2=0 d1=0 d0=0 s1=1 s0=1   y=0
                  70 d3=1 d2=0 d1=0 d0=0 s1=1 s0=1   y=1
```

GTK WAVE:



Name: **ASHLESHA.T**	SRN: **PES1UG22CS118**	Section **B**

PROGRAM : 3

RIPPLE CARRY ADDER:

Source code:

```
module invert(input wire i, output wire o1);
    assign o1 = !i;
endmodule

module and2(input wire i0, i1, output wire o2);
    assign o2 = i0 & i1;
endmodule

module or2(input wire i0, i1, output wire o3);
    assign o3 = i0 | i1;
endmodule

module xor2(input wire i0, i1, output wire o4);
    assign o4 = i0 ^ i1;
endmodule

module nand2(input wire i0, i1, output wire o5);
    wire t;
    and2 and2_0 (i0, i1, t);
    invert invert_0 (t, o5);
endmodule

module fulladd(input wire a, b, cin, output wire sum, cout);
wire [4:0] t;
    xor2 x0(a, b, t[0]);
    xor2 x1(t[0], cin, sum);

    and2 a0(a, b, t[1]);
    and2 a1(a, cin, t[2]);
    and2 a2(b, cin, t[3]);

    or2 o0(t[1], t[2], t[4]);
    or2 o1(t[3], t[4], cout);
endmodule
```

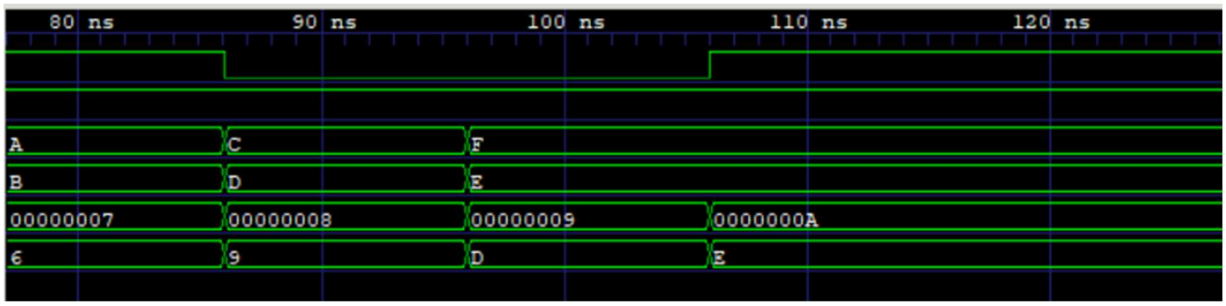
```
module fulladdR(input wire [3:0] a, b, input wire cin,
 output wire [3:0] sum, output wire cout);
wire [2:0] c;
  fulladd u0 (a[0], b[0], cin, sum[0], c[0]);
  fulladd u1 (a[1], b[1], c[0], sum[1], c[1]);
  fulladd u2 (a[2], b[2], c[1], sum[2], c[2]);
  fulladd u3 (a[3], b[3], c[2], sum[3], cout);
endmodule
```

```
`timescale 1 ns / 100 ps
`define TESTVECS 10
module tb;
reg clk, reset;
 reg [3:0] i0, i1;
 reg cin;
wire [3:0] o;
wire cout;
 reg [8:0] test_vecs [0:(`TESTVECS-1)];
integer i;
 initial
begin
$dumpfile("rca_test.vcd");
$dumpvars(0,tb);
 end
initial
begin
 reset = 1'b1; #12.5 reset = 1'b0; end
  initial clk = 1'b0; always #5 clk =~ clk;
  initial begin
 test_vecs[0] = 9'b000000000;
 test_vecs[1] = 9'b000000001;
 test_vecs[2] = 9'b000100010;
 test_vecs[3] = 9'b000100011;
 test_vecs[4] = 9'b001000100;
 test_vecs[5] = 9'b101010110;
 test_vecs[6] = 9'b101010111;
 test_vecs[7] = 9'b110011010;
 test_vecs[8] = 9'b111111100;
 test_vecs[9] = 9'b111111101;
 end
  initial {i0, i1, cin, i} = 0;
  fulladdR u0 (i0, i1, cin, o, cout);
  initial begin
    #6 for(i=0;i<`TESTVECS;i=i+1)
      begin #10 {i0, i1, cin}=test_vecs[i]; end
    #100 $finish;
  end
always@(i0 or i1 or cin)
$monitor("At time = %t, i0=%b, i1=%b,cin=%b,Sum = %b,Carry %b", $time,i0,i1,cin,o,cout);
endmodule
```

Vvp output:

```
PS C:\iverilog\bin> iverilog -o test basicfa.v rca.v nrca_tb.v
PS C:\iverilog\bin> vvp test
VCD info: dumpfile rca_test.vcd opened for output.
At time =                    0, i0=0000, i1=0000,cin=0,Sum = 0000,Carry 0
At time =                  260, i0=0000, i1=0000,cin=1,Sum = 0001,Carry 0
At time =                  360, i0=0001, i1=0001,cin=0,Sum = 0010,Carry 0
At time =                  460, i0=0001, i1=0001,cin=1,Sum = 0011,Carry 0
At time =                  560, i0=0010, i1=0010,cin=0,Sum = 0100,Carry 0
At time =                  660, i0=1010, i1=1011,cin=0,Sum = 0101,Carry 1
At time =                  760, i0=1010, i1=1011,cin=1,Sum = 0110,Carry 1
At time =                  860, i0=1100, i1=1101,cin=0,Sum = 1001,Carry 1
At time =                  960, i0=1111, i1=1110,cin=0,Sum = 1101,Carry 1
At time =                 1060, i0=1111, i1=1110,cin=1,Sum = 1110,Carry 1
```

Gtkwave:



Truth table:

4-bit Ripple Carry Adder Truth Table

	a3	a2	a1	a0	b3	b2	b1	b0	cin		
	I0[3]	I0[2]	I0[1]	I0[0]	I1[3]	I1[2]	I1[1]	I1[0]	cin	Sum[3:0]	cout
TESTVECTOR[0]	0	0	0	0	0	0	0	0	0	0+0+0=0000	0
TESTVECTOR[1]	0	0	0	0	0	0	0	0	1	0+0+1=0001	0
TESTVECTOR[2]	0	0	0	1	0	0	0	1	0	1+1+0=0010	0
TESTVECTOR[3]	0	0	0	1	0	0	0	1	1	1+1+1=0011	0
TESTVECTOR[4]	0	0	1	0	0	0	1	0	0	2+2+0=0100	0
TESTVECTOR[5]	1	0	1	0	1	0	1	1	0	A+B+0=0101	1
TESTVECTOR[6]	1	0	1	0	1	0	1	1	1	A+B+1=0110	1
TESTVECTOR[7]	1	1	0	0	1	1	0	1	0	C+D+0=1001	1
TESTVECTOR[8]	1	1	1	1	1	1	1	0	0	E+F+0=1101	1
TESTVECTOR[9]	1	1	1	1	1	1	1	0	1	E+F+1 =1110	1