Thapar Institute of Engineering and Technology

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Embedded Systems

(Verilog Programming)

Submitted by: COE 4
Submitted to: Ms. Gaganpreet Kaur

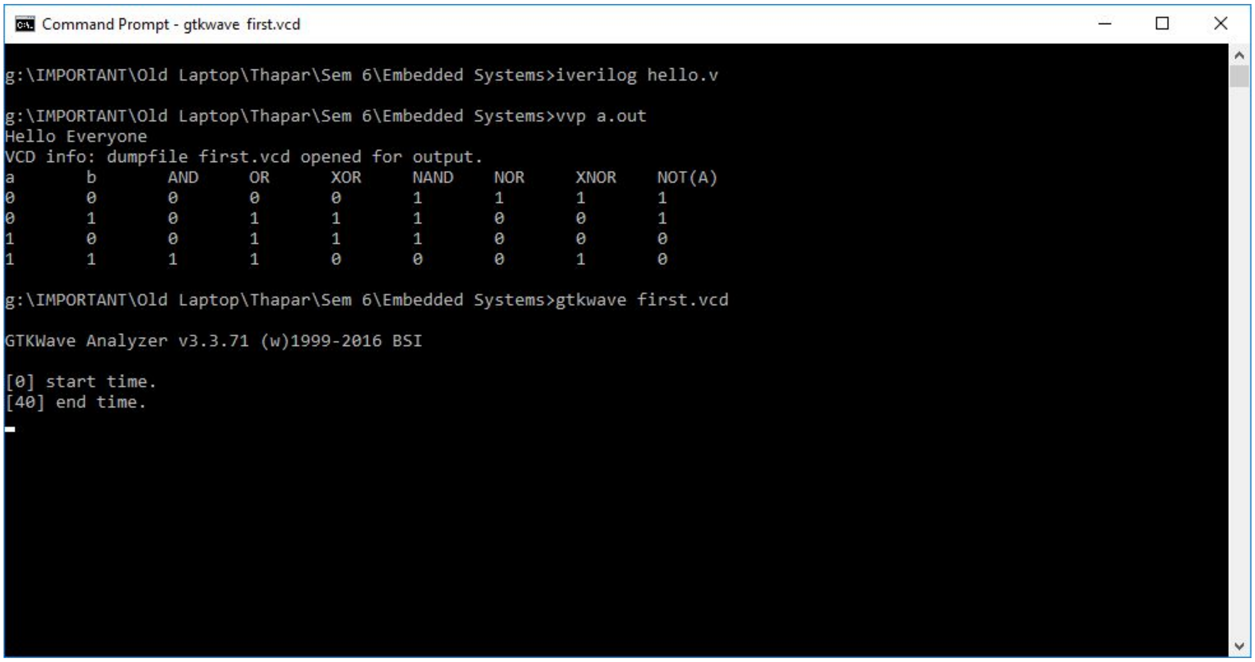
INDEX

S No.	Experiment	Page
1	Write a verilog program to show the outputs of AND, OR, XOR, NAND, NOR, XNOR and NOT gates	2
2	a). Write a verilog program to show the output of Half adder. b). Write a verilog program to show the output of full adder.	4
3	a). Write a verilog program to show the output of Half subtractor. b). Write a verilog program to show the output of full subtractor.	8
4	Write a program for 4x2 encoder using case statement. Verify the logic using truth table and time wave	12
5	Write a program for 2*1 MUX using conditional operator. Verify logic using truth table and time wave	14
6	Write a program for 1x4 DEMUX using conditional operator. Verify logic using truth table and time wave	16
7	a).Write a verilog program to show the output of T flip-flop b)..Write a verilog program to show the output of JK flip-flop	20

Q1. Write a verilog program to show the outputs of AND, OR, XOR, NAND, NOR, XNOR and NOT gates.

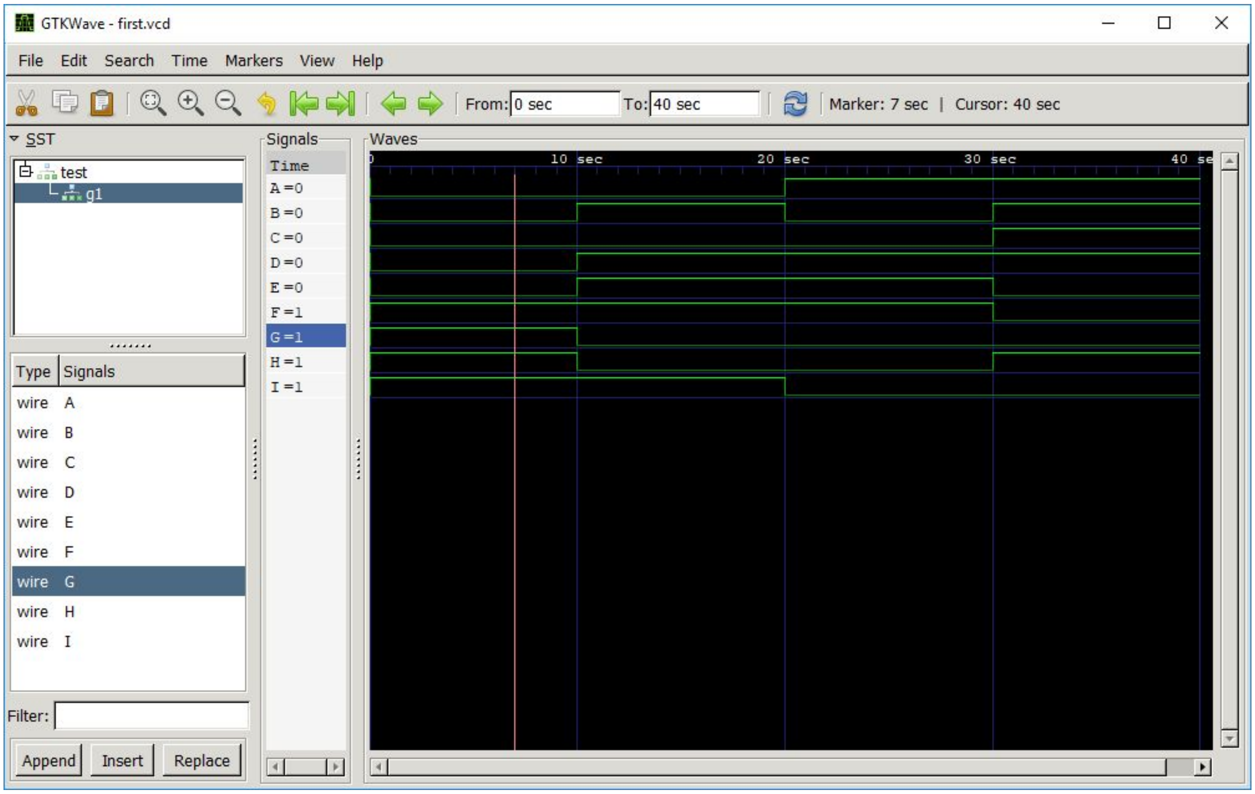
Ans.

```
module m1(A,B,C,D,E,F,G,H,I);
input A,B;
output C,D,E, F, G, H, I;
assign C= A&B;
assign D= A|B;
assign E= A^B;
assign F= ~(A&B);
assign G= ~(A|B);
assign H= ~(A^B);
assign I= ~A;
endmodule

module test;
reg a,b;
wire c,d,e,f,g,h,i;
m1 g1(a,b,c,d,e,f,g,h,i);
initial
begin
$dumpfile("first.vcd");
$dumpvars(0,test);
$display("a\tb\tAND\tOR\tXOR\tNAND\tNOR\tXNOR\tNOT(A)");
$monitor("%b\t%b\t%b\t%b\t%b\t%b\t%b\t%b\t%b",a,b,c,d,e,f,g,h,i);
a=0;b=0;
#10 a=0;b=1;
#10 a=1;b=0;
#10 a=1;b=1;
#10
$finish;
end
endmodule
```



```
g:\IMPORTANT\Old Laptop\Thapar\Sem 6\Embedded Systems>iverilog hello.v

g:\IMPORTANT\Old Laptop\Thapar\Sem 6\Embedded Systems>vvp a.out
Hello Everyone
VCD info: dumpfile first.vcd opened for output.
a       b       AND     OR      XOR     NAND    NOR     XNOR    NOT(A)
0       0       0       0       0       1       1       1       1
0       1       0       1       1       1       0       0       1
1       0       0       1       1       1       0       0       0
1       1       1       1       0       0       0       1       0

g:\IMPORTANT\Old Laptop\Thapar\Sem 6\Embedded Systems>gtkwave first.vcd

GTKWave Analyzer v3.3.71 (w)1999-2016 BSI

[0] start time.
[40] end time.
```

Command Prompt



From: 0 sec To: 40 sec Marker: 7 sec | Cursor: 40 sec

```
Time
A =0
B =0
C =0
D =0
E =0
F =1
G =1
H =1
I =1
```

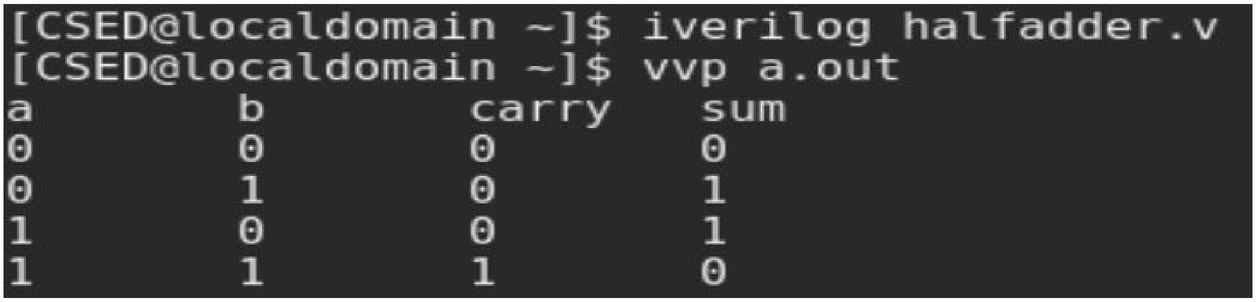
Type	Signals
wire	A
wire	B
wire	C
wire	D
wire	E
wire	F
wire	G
wire	H
wire	I

GTK Wave

Q2(a). Write a verilog program to show the output of Half adder.

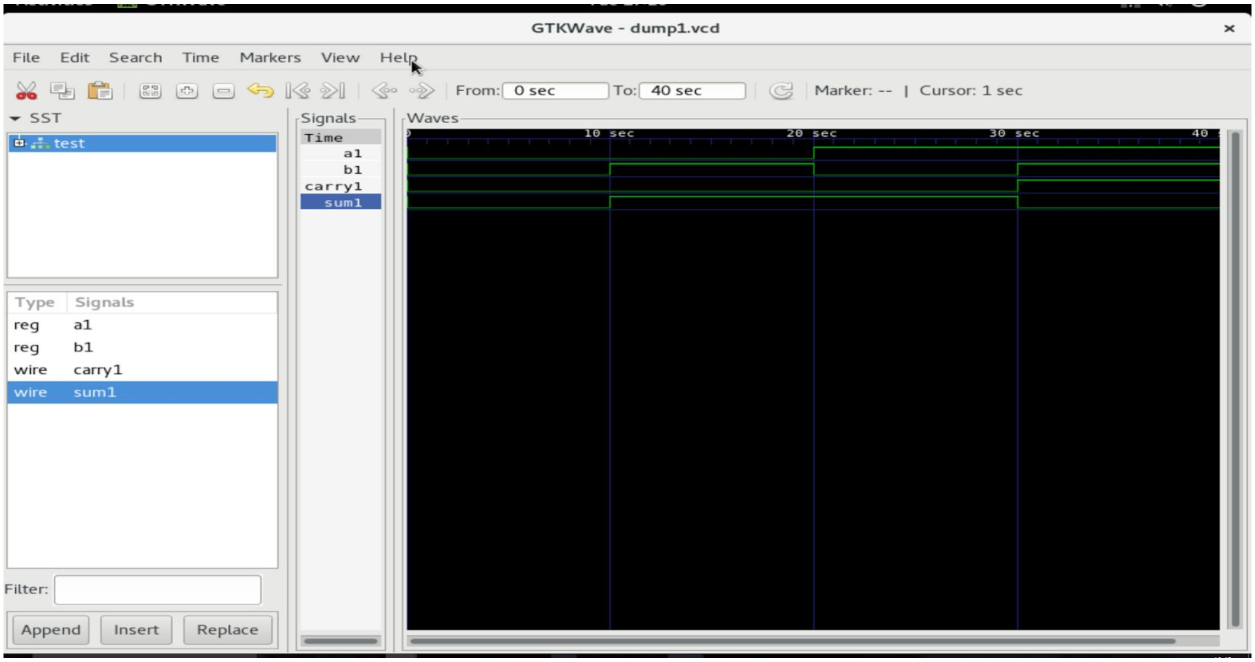
Ans.

```
module halfadder(a,b,carry,sum);
input a,b;
output carry,sum;
assign carry=a&b;
assign sum=a^b;
endmodule

module test;
reg a1,b1;
wire carry1,sum1;
halfadder h1(a1,b1,carry1,sum1);
initial
begin
$dumpfile("dump1.vcd");
$dumpvars(0,test);
$display("a\tb\tcarry\tsum");
$monitor("%b\t%b\t%b\t%b",a1,b1,carry1,sum1);
a1=0;
b1=0;
#10;
a1=0;
b1=1;
#10;
a1=1;
b1=0;
#10;
a1=1;
b1=1;
#10;
$finish;
end
endmodule
```



```
[CSED@localdomain ~]$ iverilog halfadder.v
[CSED@localdomain ~]$ vvp a.out
a        b        carry    sum
0        0        0        0
0        1        0        1
1        0        0        1
1        1        1        0
```

Command Prompt



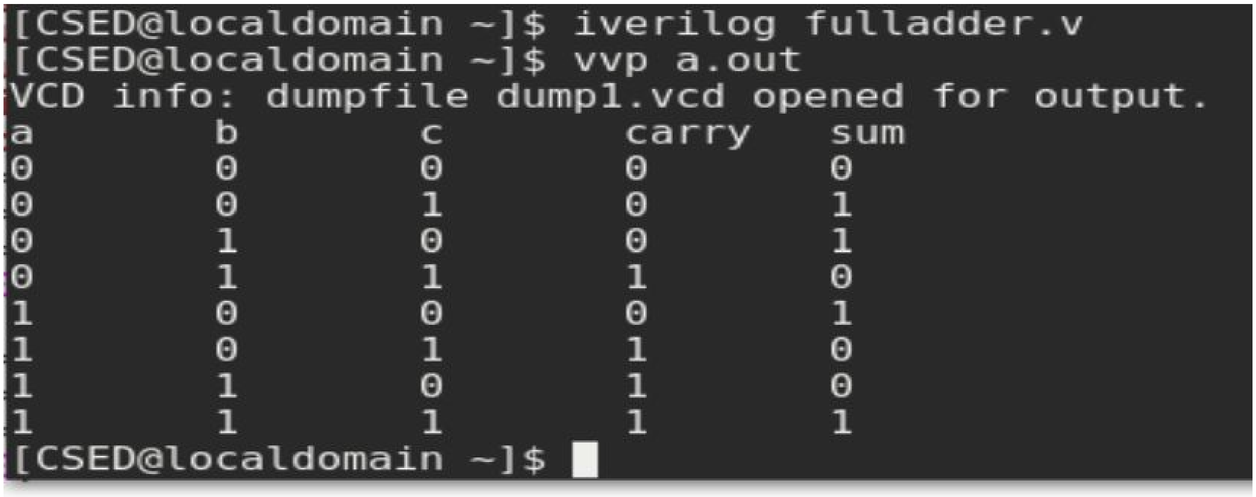
From: 0 sec To: 40 sec | Marker: -- | Cursor: 1 sec

Type	Signals
reg	a1
reg	b1
wire	carry1
wire	sum1

GTK Wave

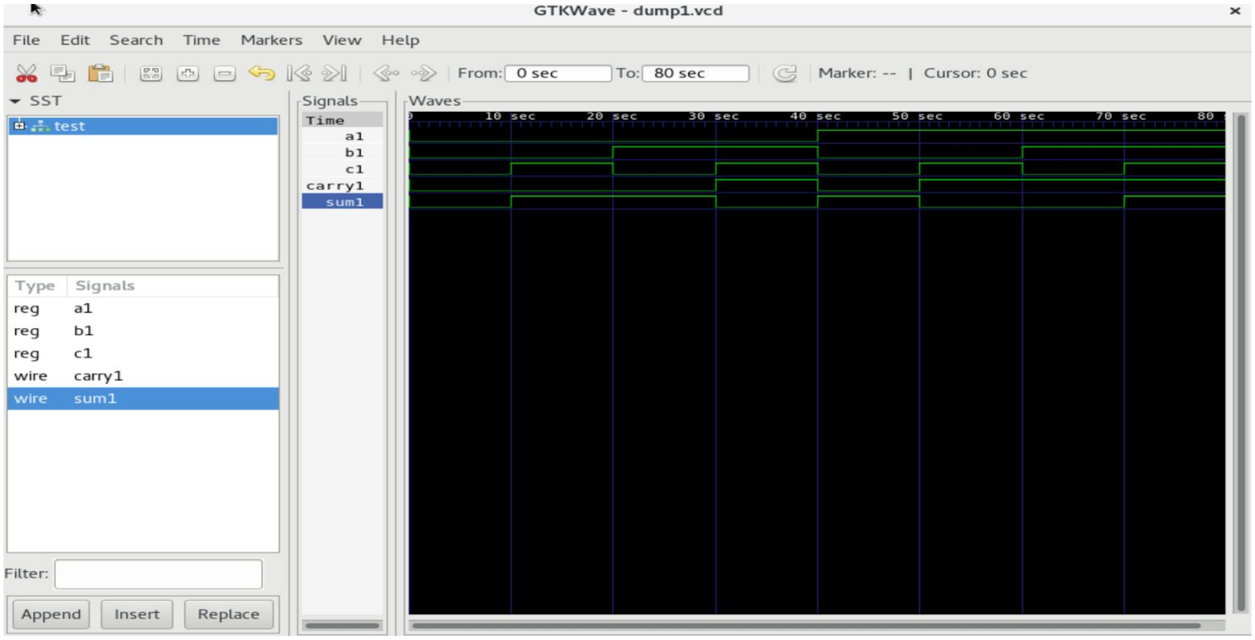
Q2(b). Write a verilog program to show the output of Full adder.
Ans.

```
module fulladder(a,b,c,carry,sum);
input a,b,c;
output carry,sum;
assign carry=a&b|a&c|b&c;
assign sum=a^b^c;
endmodule
module test;
reg a1,b1,c1;
wire carry1,sum1;
fulladder h1(a1,b1,c1,carry1,sum1);
initial
begin
$dumpfile("dump1.vcd");
$dumpvars(0,test);
$display("a\tb\tc\tcarry\tsum");
$monitor("%b\t%b\t%b\t%b\t%b",a1,b1,c1,carry1,sum1);
a1=0;b1=0;
c1=0;#10;
a1=0;b1=0;
c1=1;#10;
a1=0;b1=1;
c1=0;#10;
a1=0;b1=1;
c1=1;#10;
a1=1;b1=0;
c1=0;#10;
a1=1;b1=0;
c1=1;#10;
a1=1;b1=1;
c1=0;#10;
a1=1;b1=1;
c1=1;#10;
$finish;
end
endmodule
```



```
[CSED@localdomain ~]$ iverilog fulladder.v
[CSED@localdomain ~]$ vvp a.out
VCD info: dumpfile dump1.vcd opened for output.
a       b       c       carry   sum
0       0       0       0       0
0       0       1       0       1
0       1       0       0       1
0       1       1       1       0
1       0       0       0       1
1       0       1       1       0
1       1       0       1       0
1       1       1       1       1
[CSED@localdomain ~]$ 
```

Command Prompt



GTKWave - dump1.vcd

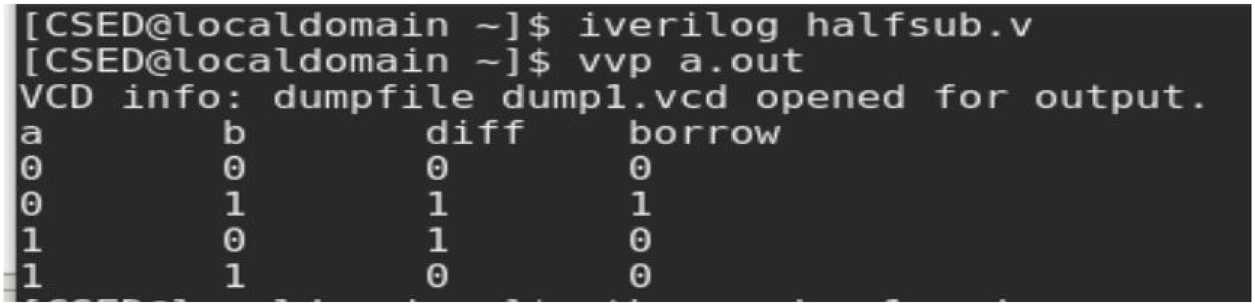
From: 0 sec To: 80 sec Marker: -- | Cursor: 0 sec

Type	Signals
reg	a1
reg	b1
reg	c1
wire	carry1
wire	sum1

GTK Wave

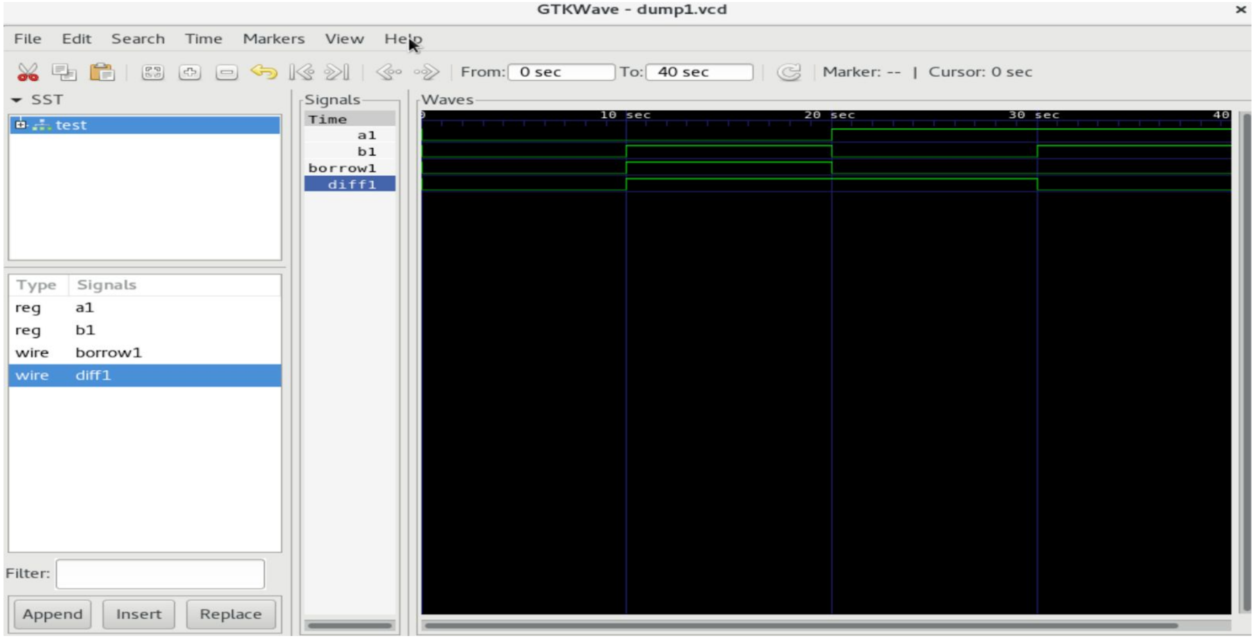
Q3(a). Write a verilog program to show the output of Half subtractor.
Ans:

```
module halfsubtractor(x,y,d,b);
input x,y;
output d,b;
assign d=x^y;
assign b=~x&y;
endmodule

module test;
reg x1,y1;
wires s1,c1;
halfsub g1(x1,y1,d1,b1);
initial
begin
$dumpfile("dump1.vcd");
$dumpvars(0,test);
$display("x y d b");
$monitor("%b\t%b\t%b\t%b",x1,y1,s1,c1);
x1=0;
y1=0;
#10     x1=0; y1=0;
#10     x1=0; y1=1;
#10     x1=1; y1=0;
#10     x1=1; y1=1;
#10
$finish;
end
endmodule
```



```
[CSED@localdomain ~]$ iverilog halfsub.v
[CSED@localdomain ~]$ vvp a.out
VCD info: dumpfile dump1.vcd opened for output.
a       b       diff    borrow
0       0       0       0
0       1       1       1
1       0       1       0
1       1       0       0
```

Command prompt



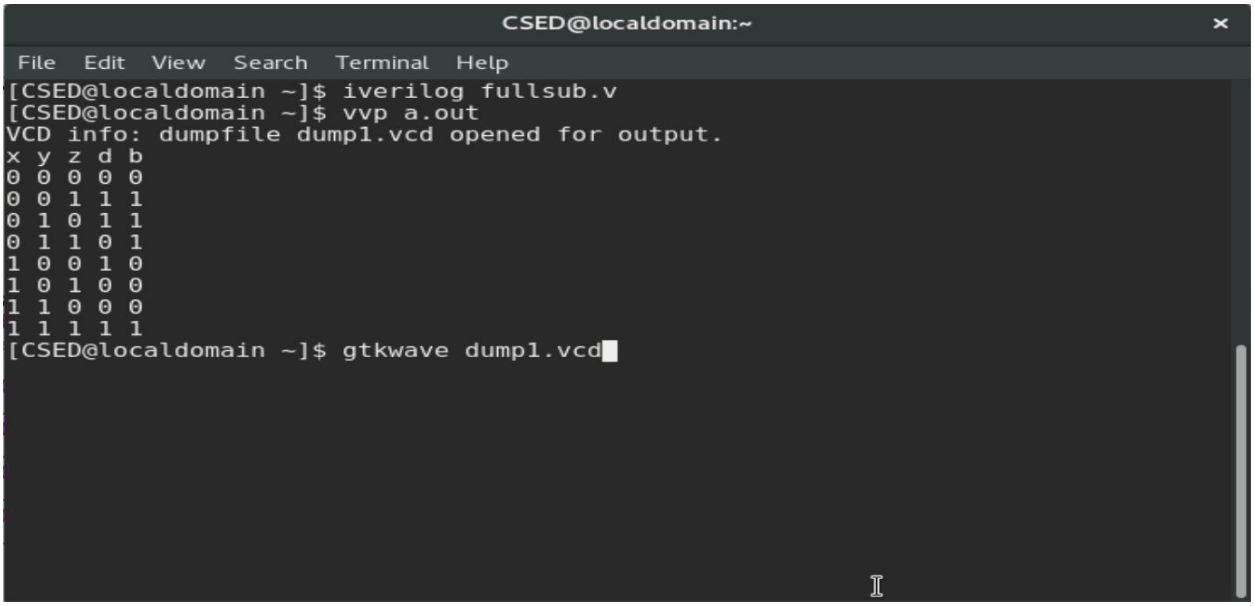
From: 0 sec To: 40 sec Marker: -- | Cursor: 0 sec

Type	Signals
reg	a1
reg	b1
wire	borrow1
wire	diff1

GTK Wave

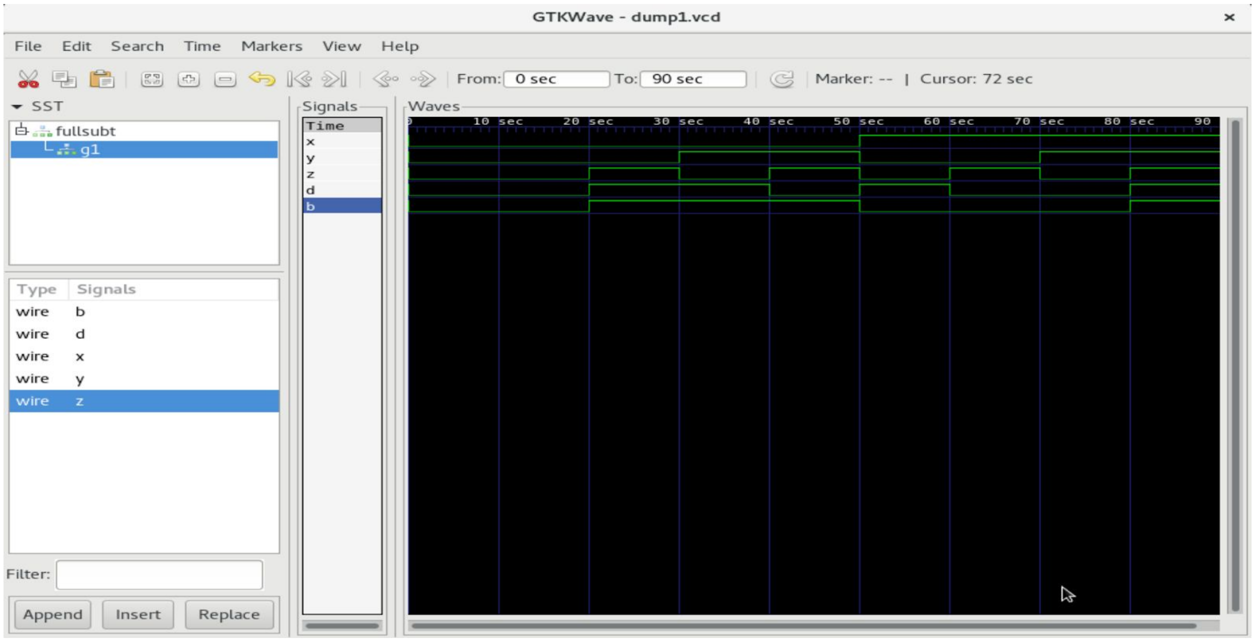
Q3(b). Write a verilog program to show the output of Full subtractor.
Ans:

```
module fullsubtractor(x,y,z,d,b);
input x,y,z;
output d,b;
assign d=x^y^z;
assign b=~x&z|~x&y|y&z;
endmodule

Module test;
reg x1,y1,z1;
wire s1,c1;
fullsub g1(x1,y1,z1,s1,c1);
initial
begin
$dumpfile("dump1.vcd");
$dumpvars(0,test);
$display("x y z d b");
$monitor("%b\t%b\t%b\t%b\t%b",x1,y1,z1,s1,c1);
x1=0;
y1=0;
z1=0;
#10     x1=0; y1=0; z1=0;
#10     x1=0; y1=0; z1=1;
#10     x1=0; y1=1; z1=0;
#10     x1=0; y1=1; z1=1;
#10     x1=1; y1=0; z1=0;
#10     x1=1; y1=0; z1=1;
#10     x1=1; y1=1; z1=0;
#10     x1=1; y1=1; z1=1;
#10
$finish;
end
endmodule
```



```
[CSED@localdomain ~]$ iverilog fullsub.v
[CSED@localdomain ~]$ vvp a.out
VCD info: dumpfile dump1.vcd opened for output.
x y z d b
0 0 0 0 0
0 0 1 1 1
0 1 0 1 1
0 1 1 0 1
1 0 0 1 0
1 0 1 0 0
1 1 0 0 0
1 1 1 1 1
[CSED@localdomain ~]$ gtkwave dump1.vcd
```

Command prompt

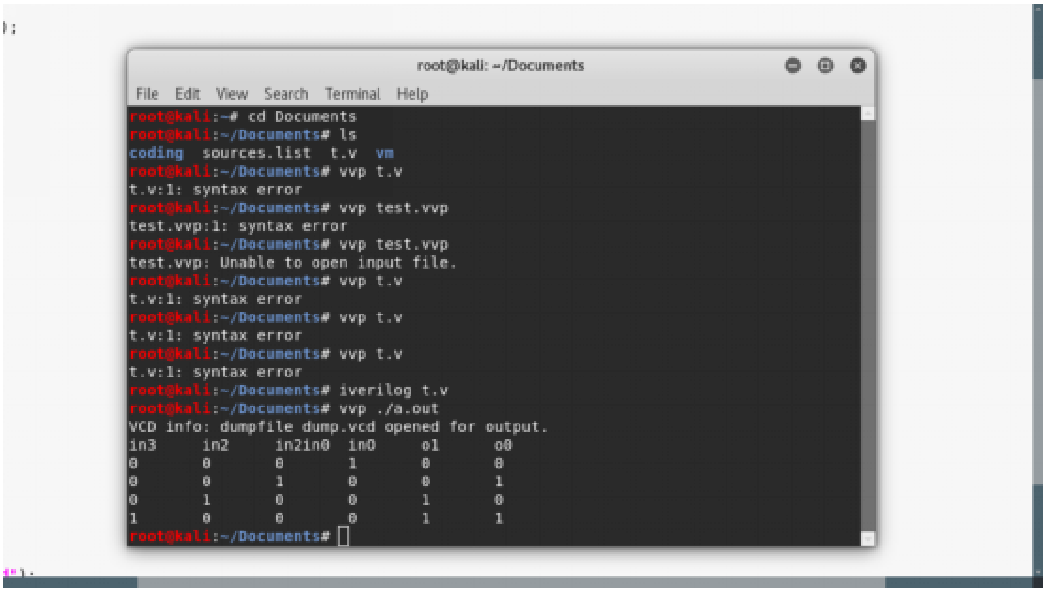


GTK Wave

Q.4 Write a program for 4x2 encoder using case statement. Verify the logic using truth table and time wave.

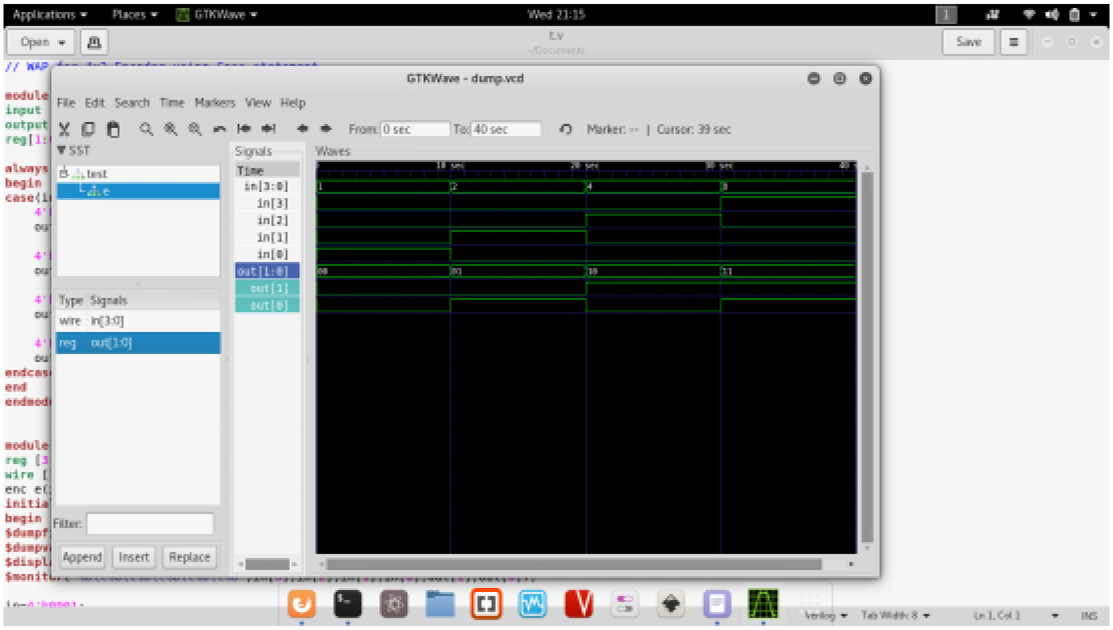
Ans.

```
module enc(in, out);
input [3:0] in;
output [1:0] out;
reg[1:0]out;
always @(*)
begin
case(in)
   4'b0001:
   out=2'b00;
   4'b0010:
   out=2'b01;
   4'b0100:
   out=2'b10;
   4'b1000:
   out=2'b11;
endcase
end
endmodule

module test;
reg [3:0] in;
wire [1:0] out;
enc e(in,out);
initial
begin
$dumpfile("dump.vcd");
$dumpvars(0,test);
$display("in3\tin2\tin2\in0\tinO\to1\to0");
$monitor("%b\t%b\t%b\t%b\t%b\t%b",in[3],in[2],in[1],in[0],out[1],out[0]);

in=4'b0001;
#10
in=4'b0010;
#10
in=4'b0100;
#10
in=4'b1000;
#10 $finish;
end
endmodule
```



```
root@kali:~# cd Documents
root@kali:~/Documents# ls
coding  sources.list  t.v  vm
root@kali:~/Documents# vvp t.v
t.v:1: syntax error
root@kali:~/Documents# vvp test.vvp
test.vvp:1: syntax error
root@kali:~/Documents# vvp test.vvp
test.vvp: Unable to open input file.
root@kali:~/Documents# vvp t.v
t.v:1: syntax error
root@kali:~/Documents# vvp t.v
t.v:1: syntax error
root@kali:~/Documents# vvp t.v
t.v:1: syntax error
root@kali:~/Documents# iverilog t.v
root@kali:~/Documents# vvp ./a.out
VCD info: dumpfile dump.vcd opened for output.
in3     in2     in2in0  in0     o1      o0
0       0       0       1       0       0
0       0       1       0       0       1
0       1       0       0       1       0
1       0       0       0       1       1
root@kali:~/Documents# 
```

Command Prompt

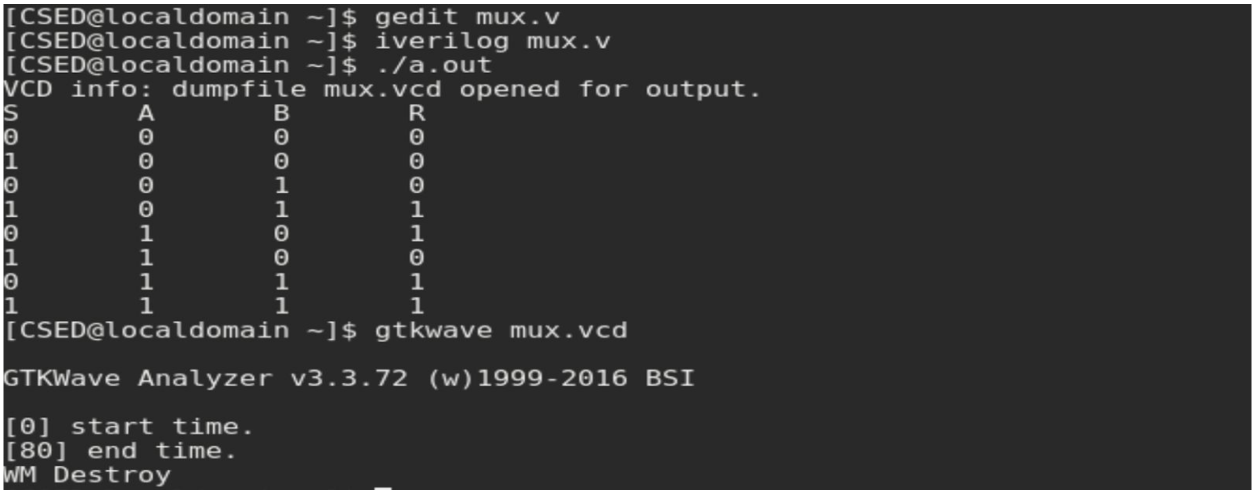


GTK Wave

Q5. Write a program for 2*1 MUX using conditional operator. Verify logic using truth table and time wave.

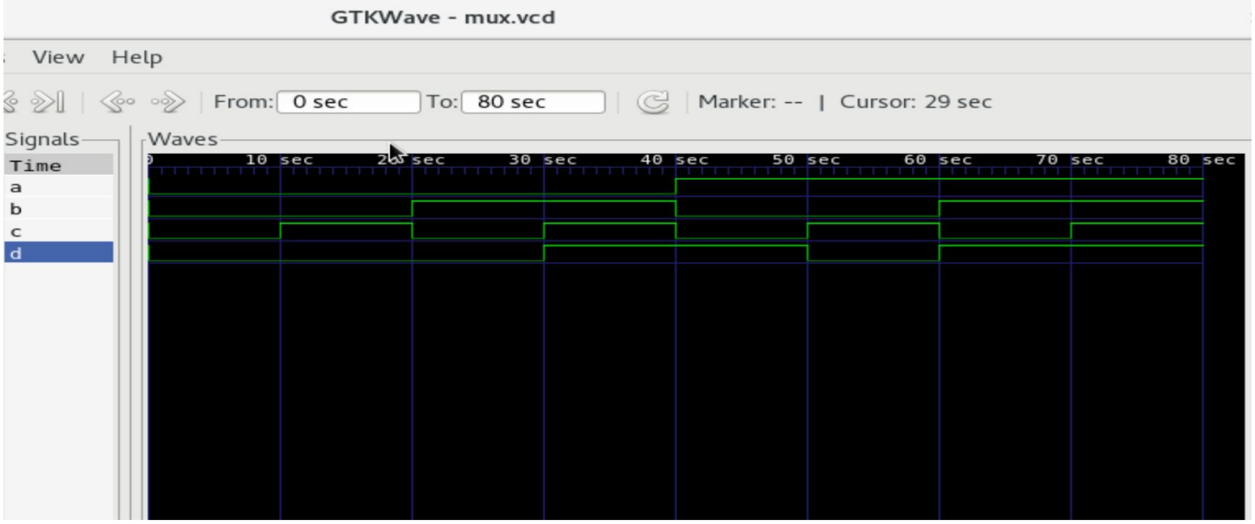
Ans.

```
module m1(C, A, B, D);
input A, B, C;
output D;
assign D=C? B: A;
endmodule

module test2;
reg a, b, c;
wire d;
m1 g1( c, a, b, d);
initial
begin
$dumpfile( "mux.vcd" );
$dumpvars( 0, test2);
$display( "S\tA\tB\tR" );
$monitor( "%b\t%b\t%b\t%b" , c, a, b, d);
a= 0; b= 0; c= 0;
#a= 0; b= 0; c= 1;
#a= 0; b= 1; c= 0;
#a= 0; b= 1; c= 1;
#a= 1; b= 0; c= 0;
#a= 1; b= 0; c= 1;
#a= 1; b= 1; c= 0;
#a= 1; b= 1; c= 1;

#10 $finish;
end
endmodule
```



```
[CSED@localdomain ~]$ gedit mux.v
[CSED@localdomain ~]$ iverilog mux.v
[CSED@localdomain ~]$ ./a.out
VCD info: dumpfile mux.vcd opened for output.
S       A       B       R
0       0       0       0
1       0       0       0
0       0       1       0
1       0       1       1
0       1       0       1
1       1       0       0
0       1       1       1
1       1       1       1
[CSED@localdomain ~]$ gtkwave mux.vcd

GTKWave Analyzer v3.3.72 (w)1999-2016 BSI

[0] start time.
[80] end time.
WM Destroy
```

Command Prompt



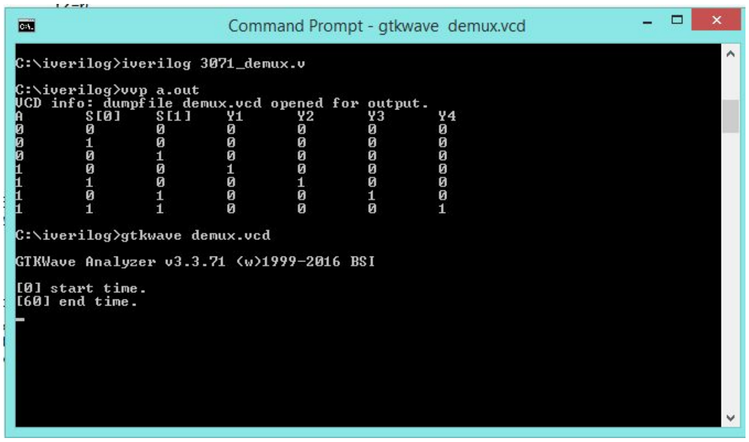
From: 0 sec To: 80 sec Marker: -- | Cursor: 29 sec

GTK Wave

Q.6) Write a program for 1x4 DEMUX using conditional operator. Verify logic using truth table and time wave

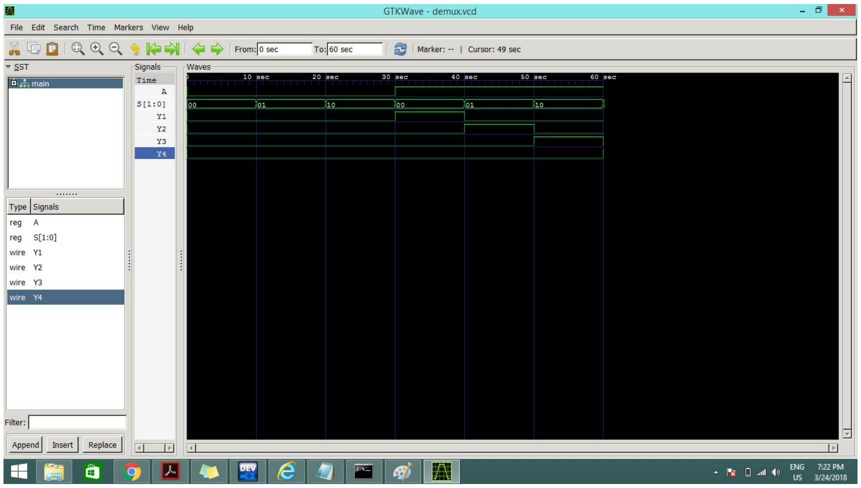
Ans.

```
module demux12(A,S,Y1,Y2,Y3,Y4);
input A;
input[1:0] S;
output Y1,Y2,Y3,Y4;
reg Y1,Y2,Y3,Y4;
always @ (*)
begin
        if(S==2'b00)
        begin
                Y1=A;
                Y2=0;
                Y3=0;
                Y4=0;
        end
        else if(S==2'b01)
        begin
                Y1=0;
                Y2=A;
                Y3=0;
                Y4=0;
        end
        else if(S==2'b10)
        begin
                Y1=0;
                Y2=0;
                Y3=A;
                Y4=0;
        end
        else if(S==2'b11)
        begin
                Y1=0;
                Y2=0;
                Y3=0;
                Y4=A;
        end
end
endmodule

module main;
reg A;
reg [1:0]S;
wire Y1,Y2,Y3,Y4;
demux12 dmx(A,S,Y1,Y2,Y3,Y4);

initial
begin
$dumpfile( "demux.vcd" );
$dumpvars( 0, main);
$display("A\tS[0]\tS[1]\tY1\tY2\tY3\tY4");
$monitor("%b\t%b\t%b\t%b\t%b\t%b\t%b",A,S[0],S[1],Y1,Y2,Y3,Y4);

A=0;
S=2'b00;
#10 S=2'b01;
#10 S=2'b10;
#10 S=2'b11;

A=1;
S=2'b00;
#10 S=2'b01;
#10 S=2'b10;
#10 S=2'b11;
$finish;
end
endmodule
```



```
C:\iverilog>iverilog 3071_demux.v

C:\iverilog>vvp a.out
VCD info: dumpfile demux.vcd opened for output.
A       S[0]    S[1]    Y1      Y2      Y3      Y4
0       0       0       0       0       0       0
0       1       0       0       0       0       0
0       0       1       0       0       0       0
1       0       0       1       0       0       0
1       1       0       0       1       0       0
1       0       1       0       0       1       0
1       1       1       0       0       0       1

C:\iverilog>gtkwave demux.vcd

GTKWave Analyzer v3.3.71 (w)1999-2016 BSI

[0] start time.
[60] end time.
_
```

Command Prompt



From: 0 sec To: 60 sec Marker: -- | Cursor: 49 sec

Type	Signals
reg	A
reg	S[1:0]
wire	Y1
wire	Y2
wire	Y3
wire	Y4

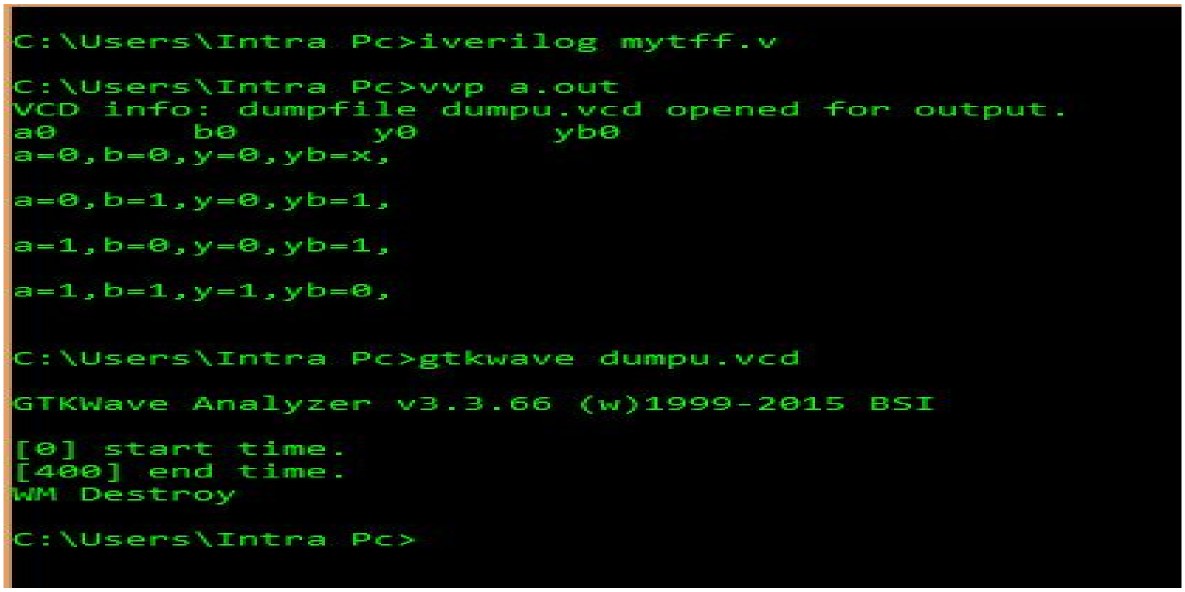
GTK Wave

Q.7(a) .Write a verilog program to show the output of T flip-flop.

Ans-

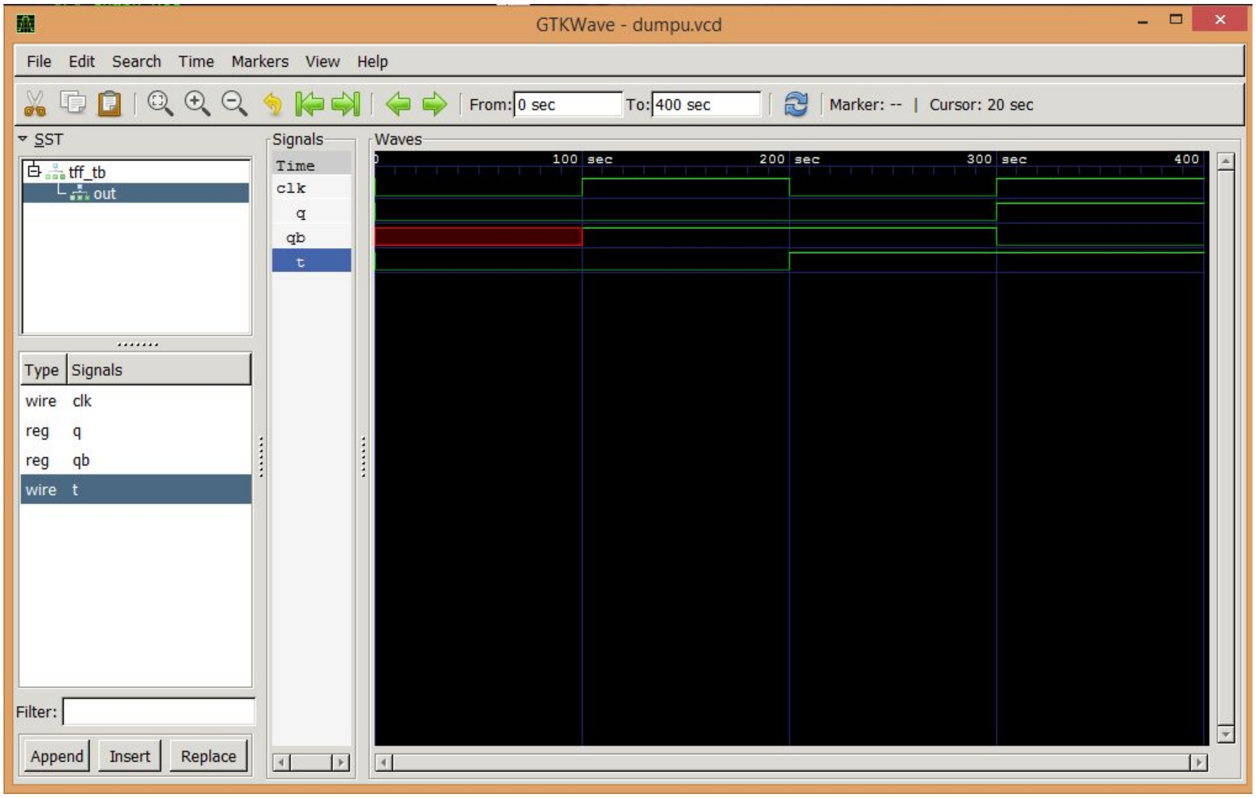
```
module mytff(t,q,qb,clk);
input   t,clk;
output q,qb;
reg q,qb;
initial q=0;
always@(posedge clk)
begin
        if (t==1)
        begin
                q=~q;
                end
        Else


        begin
            q=q;
            end
            qb=~q;
    end
endmodule

module tff_tb;
reg a,b;
wire y,yb;
mytff out(.t(a), .clk(b),.q(y),.qb(yb));
initial
begin
            $dumpfile("dumpu.vcd");
            $dumpvars(0,tff_tb);
            $display("a0\tb0\ty0\tyb0");
            $monitor("a=%d,b=%d,y=%d,yb=%d,\n",a,b,y,yb);
                        a=0;
                        b=0;
            #100;
                        b=1;
            #100;
                        a=1;
                        b=0;
            #100;
                        b=1;
            #100;
```

```
end
endmodule
```



```
C:\Users\Intra Pc>iverilog mytff.v

C:\Users\Intra Pc>vvp a.out
VCD info: dumpfile dumpu.vcd opened for output.
a0       b0       y0       yb0
a=0,b=0,y=0,yb=x,

a=0,b=1,y=0,yb=1,

a=1,b=0,y=0,yb=1,

a=1,b=1,y=1,yb=0,


C:\Users\Intra Pc>gtkwave dumpu.vcd

GTKWave Analyzer v3.3.66 (w)1999-2015 BSI

[0] start time.
[400] end time.
WM Destroy

C:\Users\Intra Pc>
```

Command prompt



From: 0 sec To: 400 sec Marker: -- | Cursor: 20 sec

tff_tb
out

Type	Signals
wire	clk
reg	q
reg	qb
wire	t

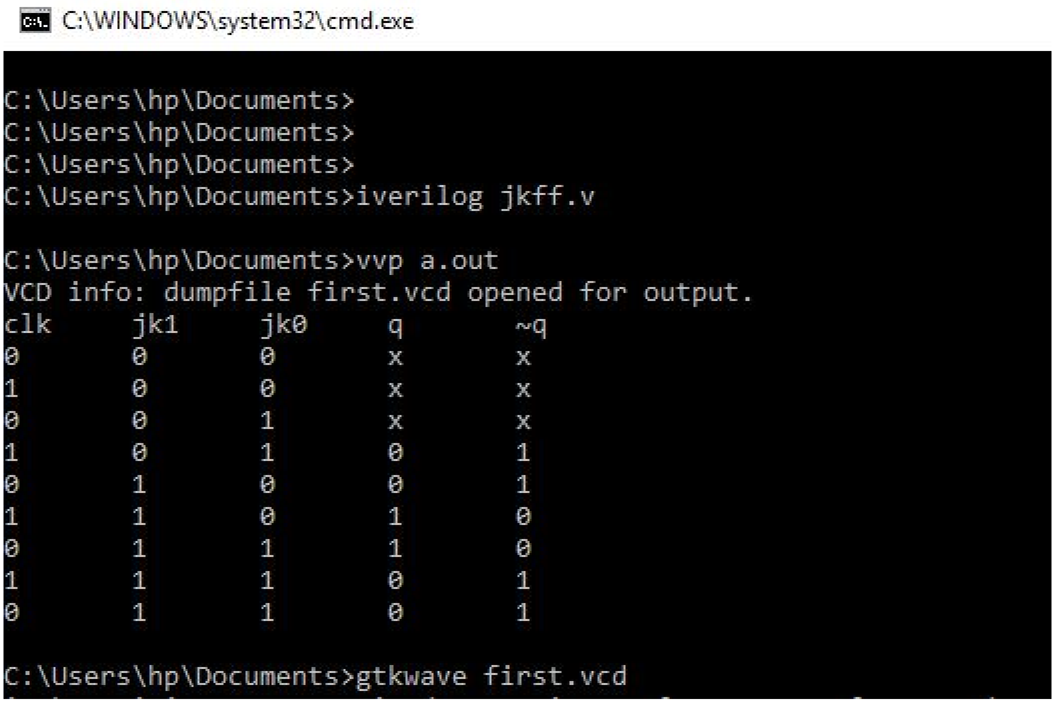
Filter:

GTK Wave

Q.7(b) Write a verilog program to show the output of JK FlipFlop

Ans.

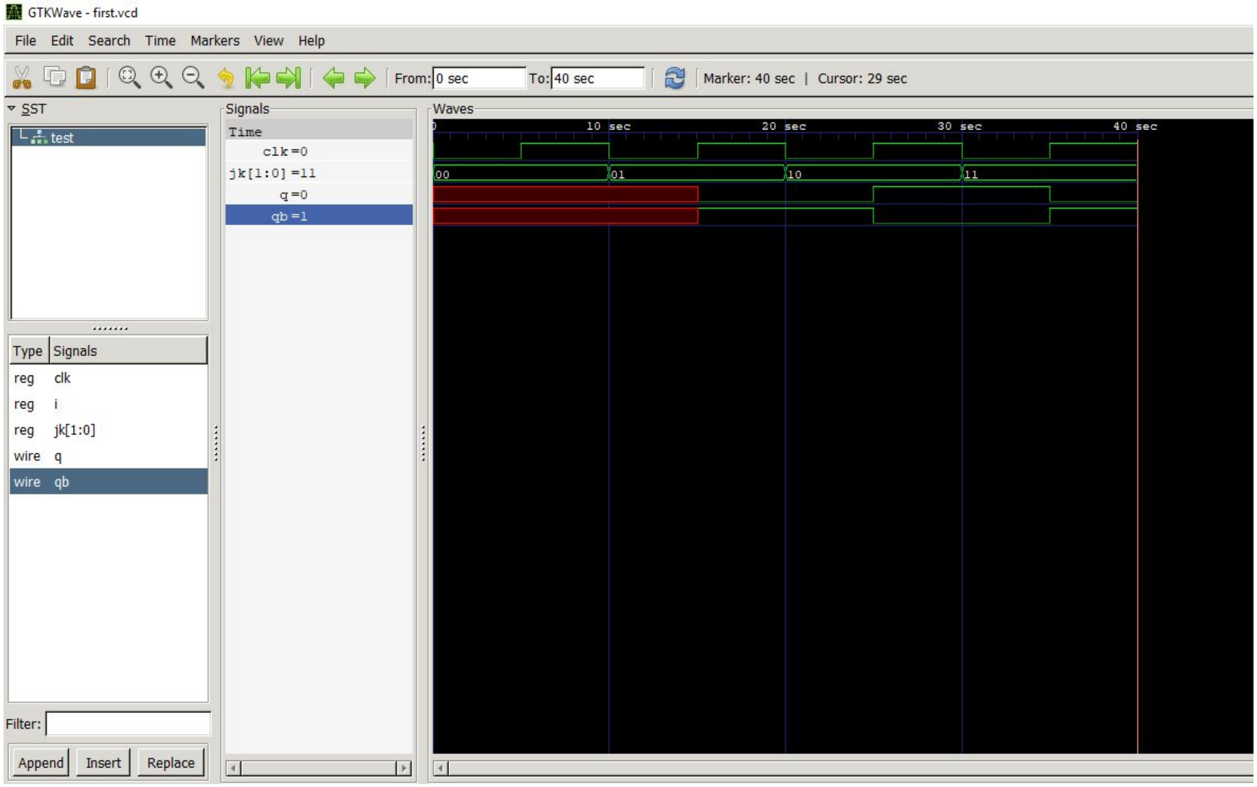
```
module jkff(input [1:0] jk,input clk,output q,output qb);
reg q,qb;
always@(posedge clk)
begin
case(jk)
2'b00:q=q;
2'b01:q=0;
2'b10:q=1;
2'b11:q=~q;
endcase
qb=~q;
end
endmodule

module test;
reg [1:0]jk;
reg clk,i;
wire q,qb;
jkff s(jk,clk,q,qb);
initial
begin
$dumpfile("first.vcd");
$dumpvars(1,test);
$display("clk\tjk1\tjk0\tq\t~q");
$monitor("%b\t%b\t%b\t%b\t%b",clk,jk[1],jk[0],q,qb);
jk=2'b00;#10
jk=2'b01;#10
jk=2'b10;#10
jk=2'b11;#10
$finish;
end
initial
begin
clk=0;
for(i=0;i<=20;i++)
#5 clk=~clk;
end
endmodule
```



C:\WINDOWS\system32\cmd.exe

```
C:\Users\hp\Documents>
C:\Users\hp\Documents>
C:\Users\hp\Documents>
C:\Users\hp\Documents>iverilog jkff.v

C:\Users\hp\Documents>vvp a.out
VCD info: dumpfile first.vcd opened for output.
clk     jk1     jk0     q       ~q
0       0       0       x       x
1       0       0       x       x
0       0       1       x       x
1       0       1       0       1
0       1       0       0       1
1       1       0       1       0
0       1       1       1       0
1       1       1       0       1
0       1       1       0       1

C:\Users\hp\Documents>gtkwave first.vcd
```

Command Prompt



GTKWave - first.vcd

From: 0 sec To: 40 sec Marker: 40 sec | Cursor: 29 sec

Type	Signals
reg	clk
reg	i
reg	jk[1:0]
wire	q
wire	qb

```
Time
clk=0
jk[1:0]=11
q=0
qb=1
```

GTK Wave