**EVALUATION-2**

**CB.EN.U4CSE18202**

**S.AAKASH MUTHIAH**

**1.T-FLIP FLOP**

**\*CODE:**

module tff(t,clk,q,qb);

output q,qb;

input t,clk;

reg q,qb;

initial

begin

q=1'b1;

qb=1'b0;

end

always@(posedge clk)

begin

if(clk)

begin

if(t==1'b0)

begin

q=q;

qb=qb;

end

else

begin

q=~q;

qb=~qb;

end

end

end

endmodule

**\*TEST-BENCH**

module tff\_tb;

wire t\_q,t\_qb;

reg t\_clk,t\_t;

tff my\_gate(t\_t,t\_clk,t\_q,t\_qb);

initial

begin

$display("Clock\tT\t\tQ\t\tQB\t");

$monitor("%b\t%b\t\t%b\t\t%b\t",t\_clk,t\_t,t\_q,t\_qb);

t\_clk=1'b0;

t\_t=1'b0;

#50

t\_t=1'b1;

t\_clk=1'b1;

#50

t\_t=1'b1;

t\_clk=1'b0;

#50

t\_t=1'b1;

t\_clk=1'b1;

#50

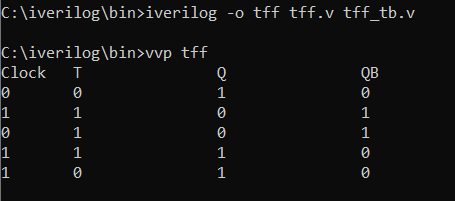
t\_t=1'b0;

t\_clk=1'b1;

end

endmodule

**OUTPUT:**



**2.JK-FLIP FLOP**

**\*CODE**

module jkff(j,k,clk,q,qb);

output q,qb;

input j,k,clk;

reg q,qb;

initial

begin

q=1'b1;

qb=1'b0;

end

always@(posedge clk)

begin

if(clk)

begin

if(j==1'b0 && k==1'b0)

begin

q=q;

qb=qb;

end

if(j==1'b0 && k==1'b1)

begin

q=1'b0;

qb=1'b1;

end

if(j==1'b1 && k==1'b0)

begin

q=1'b1;

qb=1'b0;

end

if(j==1'b1 && k==1'b1)

begin

q=~q;

qb=~qb;

end

end

end

endmodule

**\*TEST-BENCH**

module jkff\_tb;

wire t\_q,t\_qb;

reg t\_clk,t\_j,t\_k;

jkff my\_gate(.j(t\_j),.k(t\_k),.clk(t\_clk),.q(t\_q),.qb(t\_qb));

initial

begin

$display("|Clock\t|J\t|K\t|\tq\t|\tqb\t");

$monitor("|%b\t|%b\t|%b\t|\t%b\t|\t%b\t",t\_clk,t\_j,t\_k,t\_q,t\_qb);

t\_clk=1'b0;t\_j=1'b0;t\_k=1'b0;#5;

t\_clk=1'b1;t\_j=1'b1;t\_k=1'b1;#5;

t\_clk=1'b0;t\_j=1'b0;t\_k=1'b1;#5;

t\_clk=1'b1;t\_j=1'b0;t\_k=1'b1;#5;

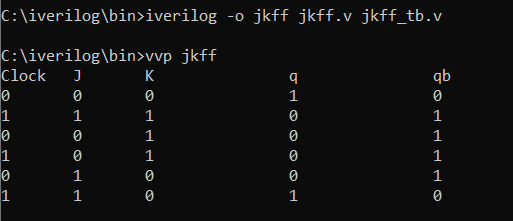
t\_clk=1'b0;t\_j=1'b1;t\_k=1'b0;#5;

t\_clk=1'b1;t\_j=1'b1;t\_k=1'b0;#5;

end

endmodule

**\*OUTPUT**



**3. 3-bit ALU with addition, subtraction, AND & OR**

**\*CODE**

module alu(a,b,opt,y);

input a,b;

input [2:0] opt;

output y;

reg y;

always @ (a or b or opt)

begin

case(opt)

3'b000: y=a+b;

3'b001: y=a-b;

3'b010: y=a&b;

3'b011: y=a|b;

3'b100: y=a+b;

3'b101: y=a-b;

3'b110: y=a&b;

3'b111: y=a|b;

endcase

end

endmodule

**\*TEST-BENCH**

module alu\_tb;

reg a,b;

reg [2:0] opt;

wire y;

alu mygate(a,b,opt,y);

initial

begin

$display("Opt: \n000,100 = ADD \n001,101 = SUB \n010,110 = AND \n011,111 = OR\n");

$monitor("Opt= %b Input = %d %d Output = %d",opt,a,b,y);

a=0; b=0; opt=3'b000;

#40;

a=0; b=1; opt=3'b000;

#40;

a=1; b=0; opt=3'b100;

#40;

a=1; b=1; opt=3'b100;

#40

a=0; b=0; opt=3'b001;

#40;

a=0; b=1; opt=3'b001;

#40;

a=1; b=0; opt=3'b101;

#40;

a=1; b=1; opt=3'b101;

#40

a=0; b=0; opt=3'b010;

#40;

a=0; b=1; opt=3'b010;

#40;

a=1; b=0; opt=3'b110;

#40;

a=1; b=1; opt=3'b110;

#40

a=0; b=0; opt=3'b011;

#40;

a=0; b=1; opt=3'b011;

#40;

a=1; b=0; opt=3'b111;

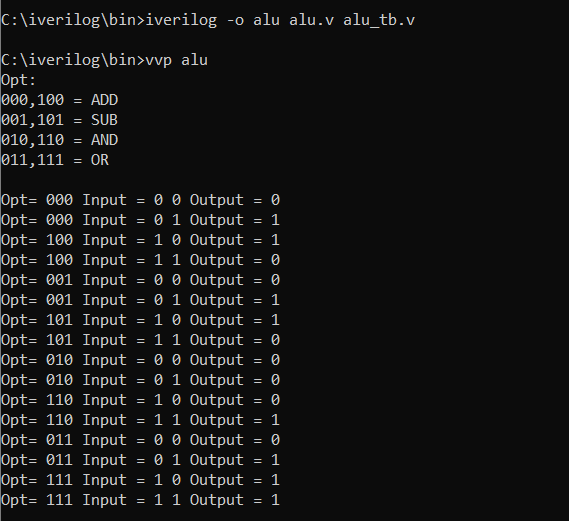
#40;

a=1; b=1; opt=3'b111;

end

endmodule

**OUTPUT:**



**4.1\*4 DEMUX**

**\*CODE**

module demux14(d,s1,s0,y1,y2,y3,y4);

input s1,s0,d;

output y1,y2,y3,y4;

assign y1=(!s0&!s1&d);

assign y2=(s0&!s1&d);

assign y3=(!s0&s1&d);

assign y4= (s0&s1&d);

endmodule

**\*TEST-BENCH**

module demux14\_tb;

reg t\_d,t\_s0,t\_s1;

wire t\_y1,t\_y2,t\_y3,t\_y4;

demux14 mygate(t\_d,t\_s1,t\_s0,t\_y1,t\_y2,t\_y3,t\_y4);

initial begin

$monitor("Input(D):%b Select Lines:%b %b Output:%b %b %b %b\n",t\_d,t\_s1,t\_s0,t\_y1,t\_y2,t\_y3,t\_y4);

t\_d=1'b1;

t\_s1=1'b0;

t\_s0=1'b0;

#5

t\_d=1'b1;

t\_s1=1'b0;

t\_s0=1'b1;

#5

t\_d=1'b1;

t\_s1=1'b1;

t\_s0=1'b0;

#5

t\_d=1'b1;

t\_s1=1'b1;

t\_s0=1'b1;

end

endmodule

**OUTPUT:**

