**Task 1:**

module task\_1(input [3:0] D0,

input [3:0] D1,

input [3:0] D2,

input [3:0] D3,

input [1:0] sel,

output reg [3:0] out);

always @(D0 or D1 or D2 or D3 or sel) begin

case(sel)

2'b00 : out <= D0;

2'b01 : out <= D1;

2'b10 : out <= D2;

2'b11 : out <= D3;

endcase

end

endmodule

**Test bench:**

module tb\_task\_1;

reg [3:0] D0;

reg [3:0] D1;

reg [3:0] D2;

reg [3:0] D3;

wire [3:0] out;

reg [1:0] sel;

integer i;

task\_1 mux0 (.D0(D0), .D1(D1), .D2(D2), .D3(D3), .sel(sel), .out(out));

initial begin

$monitor ("[%0t] sel = 0x%0h D0 = 0x%0h D1 = 0x%0h D2 = 0x%0h D3 = 0x%0h out = 0x%0h", $time, sel, D0, D1, D2, D3, out);

sel <= 0;

D0 <= $random;

D1 <= $random;

D2 <= $random;

D3 <= $random;

for (i = 1; i < 4; i = i+1) begin

#10 sel <= i;

end

#10 $finish;

end

endmodule

**Task 2:**

module full\_adder(a,b,c,sum,carry);

input a,b,c;

output sum,carry;

wire sum1,carry1,carry2;

half\_adder h1(.a(a),.b(b),.s(sum1),.c(carry1));

half\_adder h2(.a(sum1),.b(c),.s(sum),.c(carry2));

or o1(carry,carry1,carry2);

endmodule : full\_adder

module half\_adder(a,b,s,c);

input a,b;

output s,c;

xor x1(s,a,b);

and a1(c,a,b);

endmodule :half\_adder

**Test bench:**

module tb\_full\_adder;

reg a,b,c;

wire sum,carry;

full\_adder f1(.a(a),.b(b),.c(c),.sum(sum),.carry(carry));

initial begin $dumpfile("tb\_full\_adder.vcd");$dumpvars(); end

// insert all the inputs

initial begin a=1'b1; #4; a=1'b0;#10 $stop();end

initial begin b=1'b1; forever #2 b=~b;end

initial begin c=1'b1;forever #1 c=~c; #10 $stop();end

// monitor all the input and output ports at times

// when any of the input changes its state

initial begin $monitor("time=%0d A=%b B=%b

C=%b Sum=%b Carry=%b",$time,a,b,c,sum,carry); end

Endmodule

**Task 3:**

module function\_f(out, a, b, c);

output out;

input a, b, c;

assign out = (a&(!b)) | ((!b)&(!c)) | (a&c);

endmodule

module function\_g(out, a, b, c);

output out;

input a, b, c;

assign out = ((!b)|c) & (a|b|(!c));

endmodule

module function\_h(out, a, b, c);

output out;

input a, b, c;

assign out = ((!c)&(!b)) | (b&c) | (a&c);

endmodule

**Test bench:**

module tb\_task\_3\_a();

reg a,b, c;

wire out;

function\_f uut(out,a,b, c);

initial

begin

a = 1'b0;

b = 1'b0;

c = 1'b0;

#10;

a = 1'b0;

b = 1'b0;

c = 1'b1;

#10;

a = 1'b0;

b = 1'b1;

c = 1'b0;

#10;

a = 1'b0;

b = 1'b1;

c = 1'b1;

#10;

a = 1'b1;

b = 1'b0;

c = 1'b0;

#10;

a = 1'b1;

b = 1'b0;

c = 1'b1;

#10;

a = 1'b1;

b = 1'b1;

c = 1'b0;

#10;

a = 1'b1;

b = 1'b1;

c = 1'b1;

#10;

end

// set up the monitoring

initial

begin

$monitor("a=%b, b=%b, c=%b, out=%b, time=%t\n", a, b, c, out, $time);

end

endmodule

module tb\_task\_3\_b();

reg a,b, c;

wire out;

function\_g uut(out,a,b, c);

initial

begin

a = 1'b0;

b = 1'b0;

c = 1'b0;

#10;

a = 1'b0;

b = 1'b0;

c = 1'b1;

#10;

a = 1'b0;

b = 1'b1;

c = 1'b0;

#10;

a = 1'b0;

b = 1'b1;

c = 1'b1;

#10;

a = 1'b1;

b = 1'b0;

c = 1'b0;

#10;

a = 1'b1;

b = 1'b0;

c = 1'b1;

#10;

a = 1'b1;

b = 1'b1;

c = 1'b0;

#10;

a = 1'b1;

b = 1'b1;

c = 1'b1;

#10;

end

// set up the monitoring

initial

begin

$monitor("a=%b, b=%b, c=%b, out=%b, time=%t\n", a, b, c, out, $time);

end

endmodule

module tb\_task\_3\_c();

reg a,b, c;

wire out;

function\_h uut(out,a,b, c);

initial

begin

a = 1'b0;

b = 1'b0;

c = 1'b0;

#10;

a = 1'b0;

b = 1'b0;

c = 1'b1;

#10;

a = 1'b0;

b = 1'b1;

c = 1'b0;

#10;

a = 1'b0;

b = 1'b1;

c = 1'b1;

#10;

a = 1'b1;

b = 1'b0;

c = 1'b0;

#10;

a = 1'b1;

b = 1'b0;

c = 1'b1;

#10;

a = 1'b1;

b = 1'b1;

c = 1'b0;

#10;

a = 1'b1;

b = 1'b1;

c = 1'b1;

#10;

end

// set up the monitoring

initial

begin

$monitor("a=%b, b=%b, c=%b, out=%b, time=%t\n", a, b, c, out, $time);

end

endmodule