DSD LAB

Week 5 Assignment Submission

Swamiraju Satya Praveen Varma

200905044

Batch B1

10

1. Design and simulate a combinational circuit with external gates and a 4 to 16 decoder built using a decoder tree of 2 to 4 decoders to implement the functions below.
   F= ab'c + a'cd + bcd' , G=acd' + a'b'c and H=a'b'c' + abc + a'cd

   Verilog code:

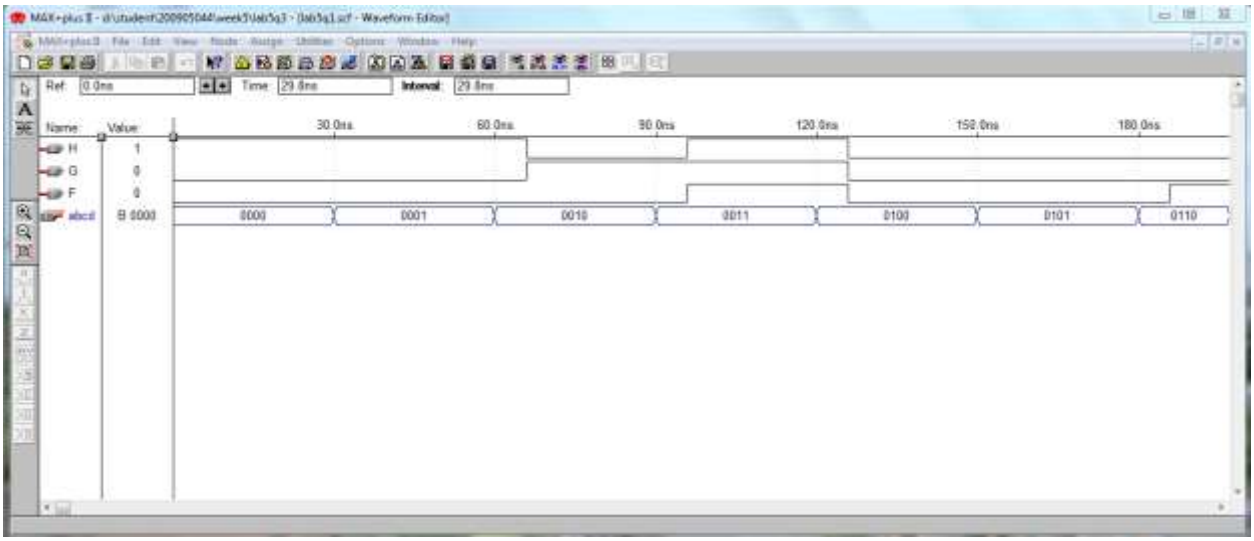
```
module lab5q1(abcd,F,G,H);
input [3:0]abcd;
output F,G,H;
wire [0:15]k;
dec4to16 stage0(1'b1,abcd[3:0],k[0:15]);
assign F=k[3] | k[6] | k[7] | k[10] | k[11] | k[14];
assign G=k[2] | k[3] | k[10] | k[14];
assign H=k[0] | k[1] | k[3] | k[7] | k[14] | k[15];
endmodule

module dec4to16(En,W,Y);
input En;
input [3:0]W;
output [0:15]Y;
wire [0:3]K;
dec2to4 stage0(1'b1,W[3:2],K[0:3]);
dec2to4 stage1(K[0],W[1:0],Y[0:3]);
dec2to4 stage2(K[1],W[1:0],Y[4:7]);
dec2to4 stage3(K[2],W[1:0],Y[8:11]);
dec2to4 stage4(K[3],W[1:0],Y[12:15]);
endmodule

module dec2to4(En,W,Y);
```

```
input En;
input [1:0]W;
output [0:3]Y;
reg [0:3]Y;
always@(En or W)
begin
case({En,W})
3'b100: Y=4'b1000;
3'b101: Y=4'b0100;
3'b110: Y=4'b0010;
3'b111: Y=4'b0001;
default: Y=4'b0000;
endcase
end
endmodule
```

Output Waveform:



2. Design and implement a full adder using 2 to 4 decoder(s) and other gates.

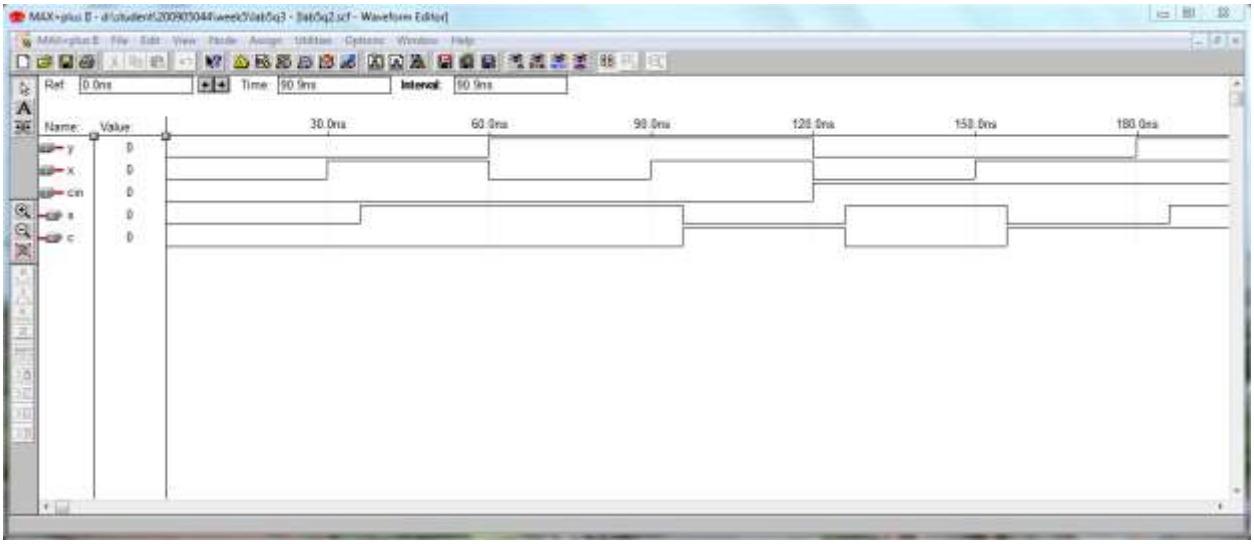
Verilog code:

```
module lab5q2(x,y,cin,s,c);
```

```
input x,y,cin;
output s,c;
wire [0:7]k;
dec2to4 stage1(~cin,{x,y},k[0:3]);
dec2to4 stage2(cin,{x,y},k[4:7]);
assign s = k[1] | k[2] | k[4] | k[7];
assign c = k[3] | k[5] | k[6] | k[7];
endmodule

module dec2to4(En,W,Y);
input En;
input [1:0]W;
output [0:3]Y;
reg [0:3]Y;
always@(En or W)
begin
case({En,W})
3'b100: Y=4'b1000;
3'b101: Y=4'b0100;
3'b110: Y=4'b0010;
3'b111: Y=4'b0001;
default: Y=4'b0000;
endcase
end
endmodule
```

Output waveform:



3. Design and simulate the circuit with 3 to 8 decoder(s) and external gates to implement the functions below.
F(a, b, c, d)= Σm(2,4,7,9) G (a, b, c, d)= Σm (0,3,15) H(a, b, c, d)= Σm(0,2,10,12)

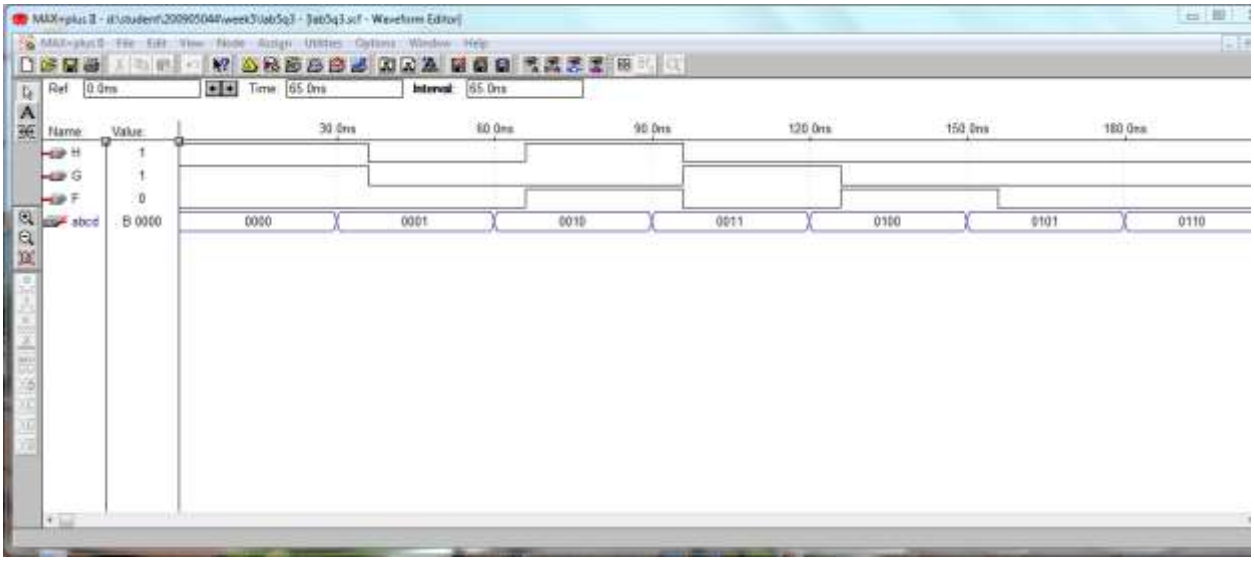
Verilog code:

```
module lab5q3(abcd,F,G,H);
input [3:0]abcd;
output F,G,H;
wire [0:15]k;
dec3to8 stage1(~abcd[3],abcd[2:0],k[0:7]);
dec3to8 stage2(abcd[3],abcd[2:0],k[8:15]);
assign F = k[2] | k[4] | k[7] | k[9];
assign G = k[0] | k[3] | k[15];
assign H = k[0] | k[2] | k[10] | k[12];
endmodule

module dec3to8(En,W,Y);
input En;
```

```
input [2:0]W;
output [0:7]Y;
reg [0:7]Y;
always@(En or W)
begin
case({En,W})
4'b1000: Y=8'b10000000;
4'b1001: Y=8'b01000000;
4'b1010: Y=8'b00100000;
4'b1011: Y=8'b00010000;
4'b1100: Y=8'b00001000;
4'b1101: Y=8'b00000100;
4'b1110: Y=8'b00000010;
4'b1111: Y=8'b00000001;
default: Y=8'b00000000;
endcase
end
endmodule
```

Output waveform:



THANK YOU!