**LAB – 2**

1. Full Adder:

Verilog code:

module fulladder(cin,x,y,s,cout);

input cin,x,y;

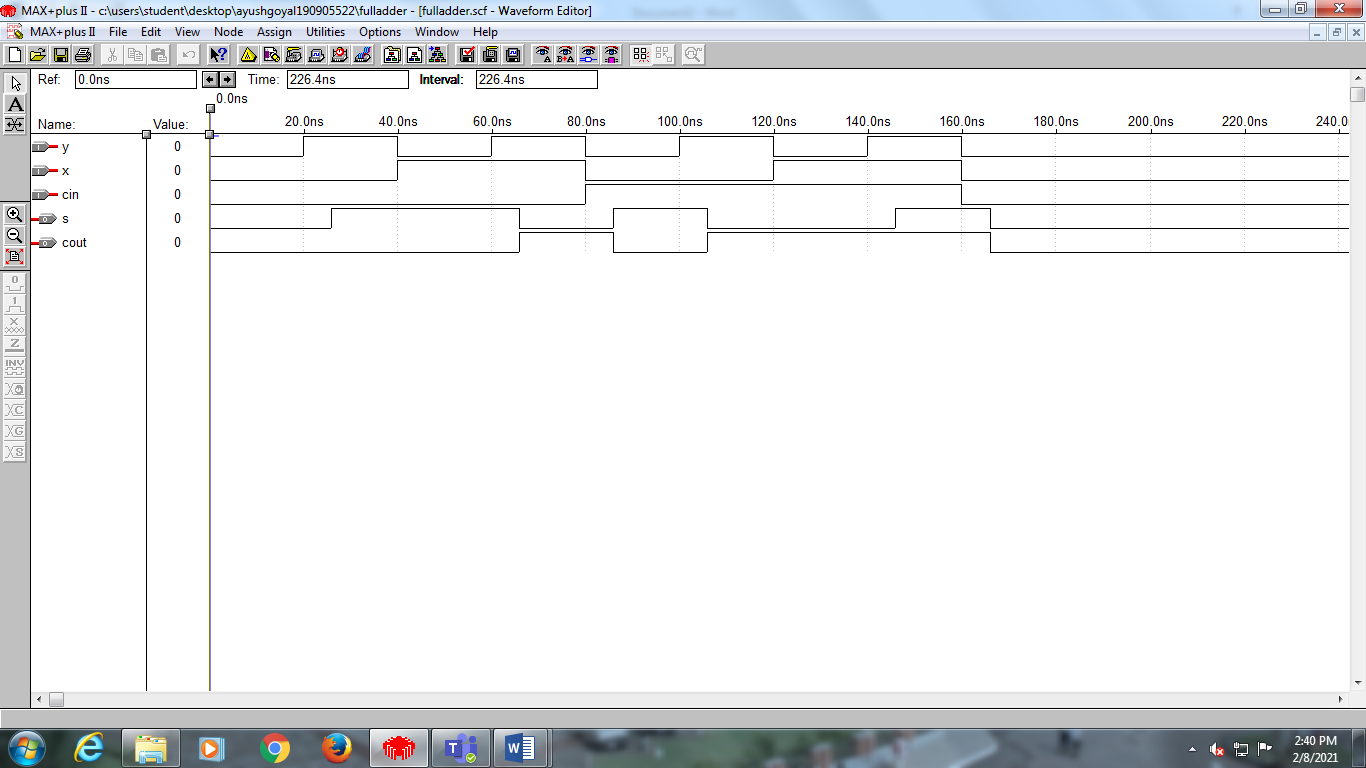
output s,cout;

assign s = cin^x^y;

assign cout = (x&y)|(x&cin)|(y&cin);

endmodule

Output:



1. Four bit Adder Subtractor:

Verilog Code:

module addsub4(cin,x,y,s,cout);

input cin;

input [3:0]x,y;

output cout;

output [3:0]s;

wire[3:1]c;

fulladder first(cin,x[0],y[0]^cin,s[0],c[1]);

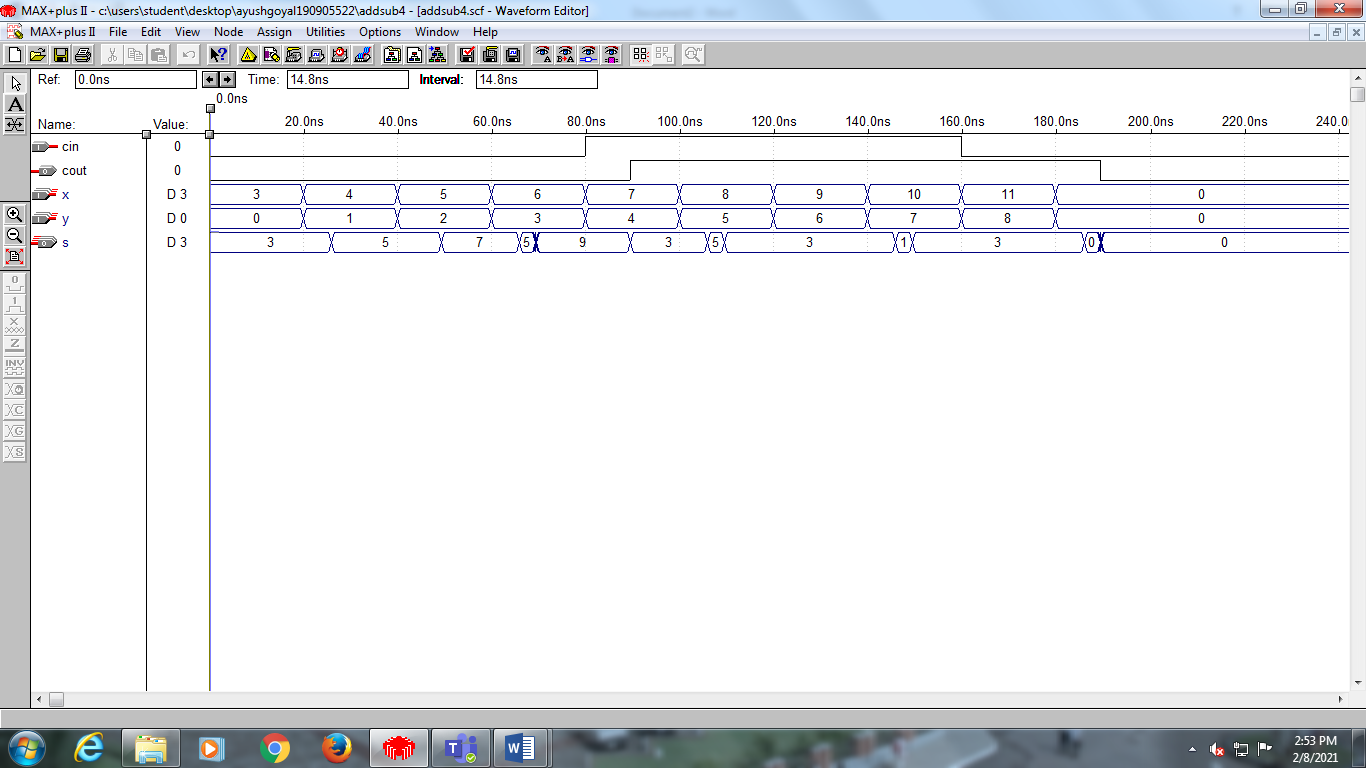
fulladder second(c[1],x[1],y[1]^cin,s[1],c[2]);

fulladder third(c[2],x[2],y[2]^cin,s[2],c[3]);

fulladder fourth(c[3],x[3],y[3]^cin,s[3],cout);

endmodule

Output:



1. Single Digit BCD Adder using a four bit adder:

Verilog code:

module bcdadder(cin, a, b, sum, cout);

input [3:0]a,b;

input cin;

output cout;

output [3:0]sum;

wire [3:0]z;

wire m;

wire [3:1]k;

addsub4 stage0(cin,a,b,z,m);

assign k[1] = z[3]&z[2];

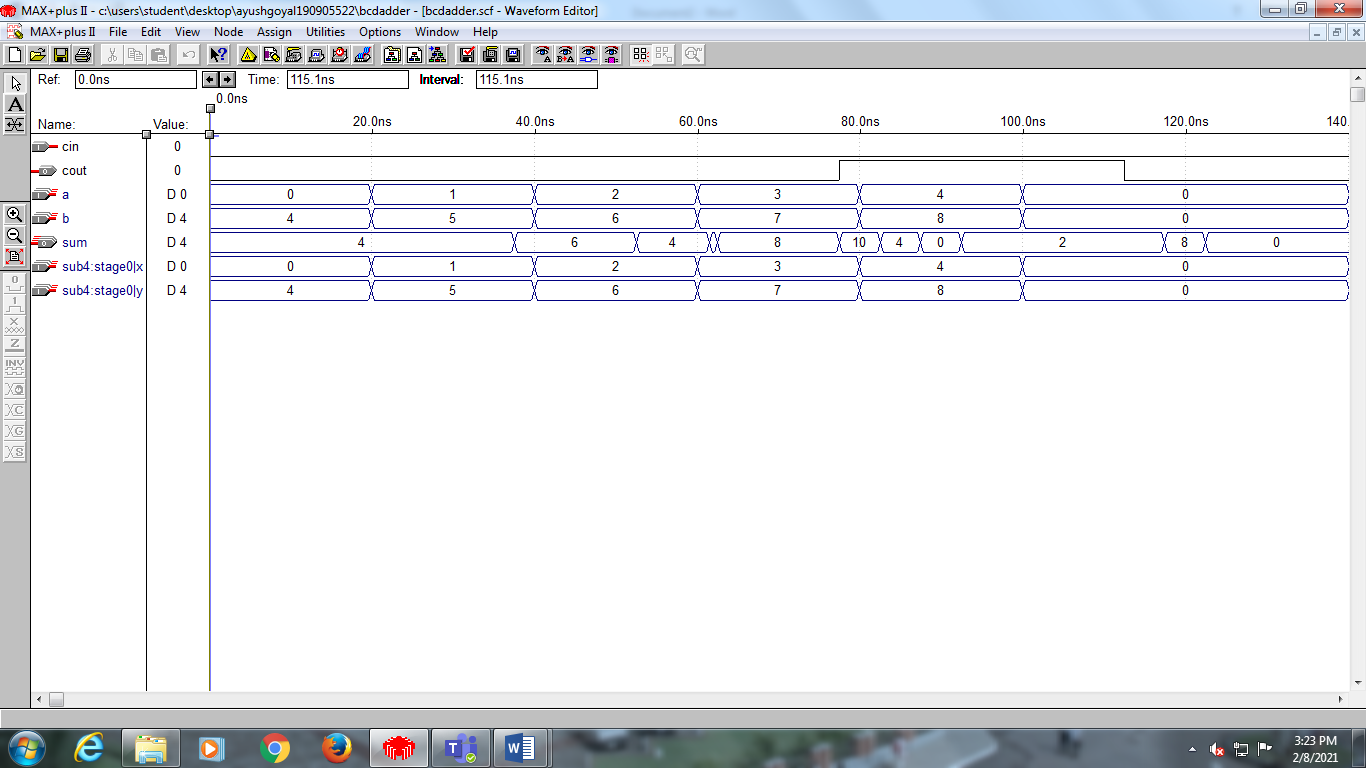
assign k[2] = z[3]&z[1];

assign k[3] = (m|k[1]|k[2]);

addsub4 stage1(cin, z, {1'b0,k[3],k[3],1'b0}, sum, cout);

endmodule

Output:



Additional Exercise Lab 2:

2 bit Multiplier:

Verilog Code:

module twobitmul(a,b,s);

input [1:0]a,b;

output [3:0]s;

wire m,n,h,c0,c1;

assign s[0] = a[0]&b[0];

assign m = a[1]&b[0];

assign n = a[0]&b[1];

assign h = a[1]&b[1];

halfadder stage0(m,n,c0,s[1]);

halfadder stage1(c0,h,s[3],s[2]);

endmodule

module halfadder(x,y,c,s);

input x,y;

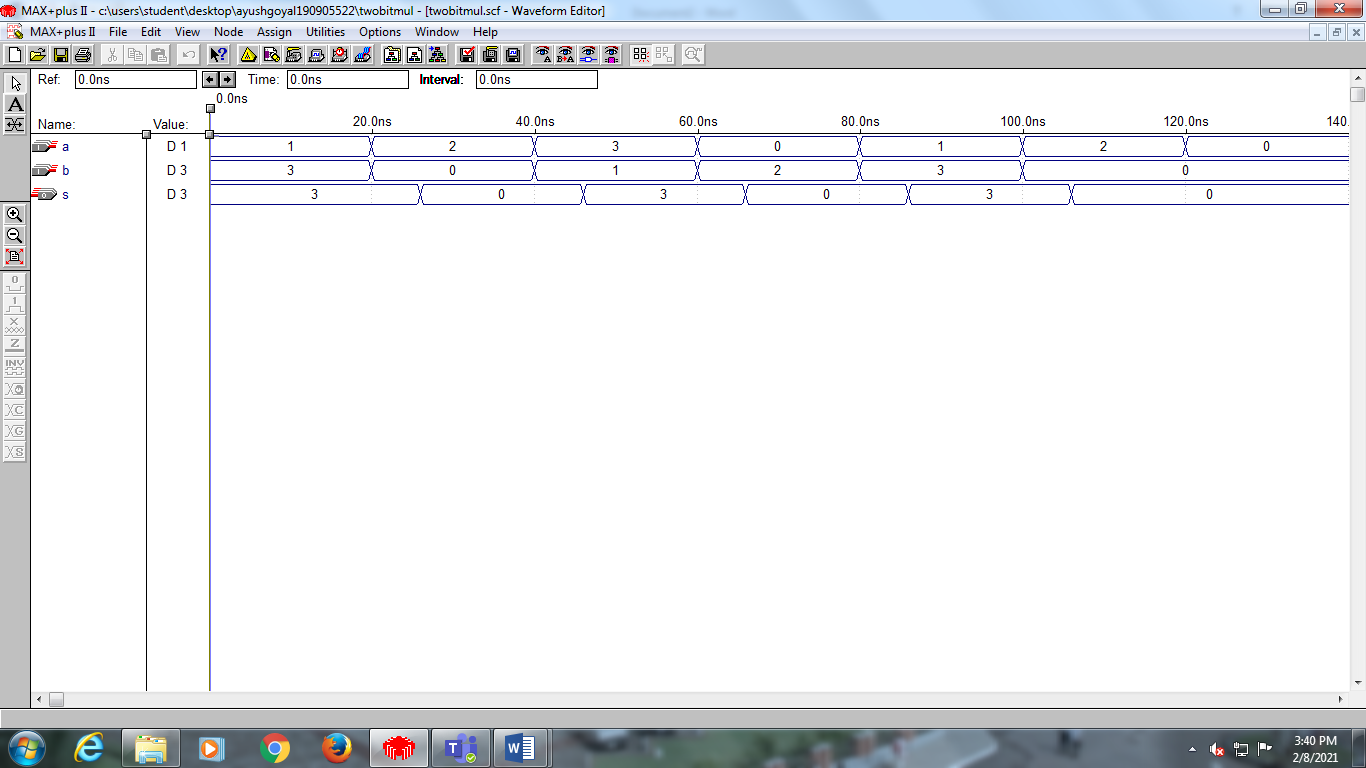
output c,s;

assign s = x^y;

assign c = x&y;

endmodule

Output:



END OF LAB 2(WEEK 2)