// Write your modules here!

module cp(cin,p,g,c2,c3);

input cin;

input[2:0]p,g;

output c2,c3;

assign c2=g[1]|(p[0]&cin&p[1])|(p[1]&g[0]);

assign c3=(g[2])|(g[1]&p[2])|(p[2]&p[1]&g[0])|(cin &p[0]&p[1]&p[2]);

endmodule

//..........................

module fa0(a0,b0,cin,c1,sum0,p0,g0);

input cin ,a0 ,b0;

output c1,sum0,p0,g0;

assign p0=a0^b0;

assign g0=a0&b0;

assign sum0=a0^b0^cin;

assign c1=(a0&b0)|(a0&cin)|(b0&cin);

endmodule

//......................

module fa1(a,b,c,sum,p,g);

input c ,a ,b;

output sum,p,g;

assign p=a^b;

assign g=a&b;

assign sum=a^b^c;

endmodule

//.............................

module fa3(a,b,c,sum3,cout);

input c ,a ,b;

output sum3,cout;

reg sum3,cout;

initial

sum3=a^b^c;

initial

cout=(a&b)|(a&c)|(b&c);

endmodule

//.............................

module circuit(a,b,s,cout,sel,overflow);

input[3:0]a,b;

input sel;

output [3:0]s;

output cout,overflow;

reg [3:0]w;

wire [2:0] c,p,g;

//to make b' when sel==1;

always @(b or sel)

begin

if (b[0]==sel)

w[0]=1'b0;

else

w[0]=1'b1;

if (b[1]==sel)

w[1]=1'b0;

else

w[1]=1'b1;

if (b[2]==sel)

w[2]=1'b0;

else

w[2]=1'b1;

if (b[3]==sel)

w[3]=1'b0;

else

w[3]=1'b1;

end

/\*

xor u(w[0],sel,b[0]);

xor u1(w[1],sel,b[1]);

xor u2(w[2],sel,b[2]);

xor u3(w[3],sel,b[3]);\*/

//initial

fa0 FA0(a[0],w[0],sel,c[0],s[0],p[0],g[0]);

//initial

fa1 FA1(a[1],w[1],c[0],s[1],p[1],g[1]);

//initial

fa1 FA2(a[2],w[2],c[1],s[2],p[2],g[2]);

cp carry\_ahead(sel,p,g,c[1],c[2]);

//initial

fa3 FA3(a[3],w[3],c[2],s[3],cout);

xor(overflow,cout,c[2]);

endmodule