**VERILOG CODE FOR FULL DADDA MULTIPLIER**

module dadda\_multiplier(s,a,b);

output [7:0]s;

input [3:0]a,b;

wire [11:0]w;

wire [5:0]x,y;

wire [15:0]p;

ppg p1(p,a,b);

assign s[0]=p[0];

HA\_CSLA\_BEC1 h1(p[8],p[7],p[6],w[0],w[1]);

HA\_CSLA\_BEC1 h2(p[11],p[12],p[10],w[2],w[3]);

HA\_CSLA\_BEC1 h3(w[3],p[14],p[13],w[10],w[11]);

half\_adder ha1(p[4],p[5],w[4],w[5]);

half\_adder ha2(p[9],w[0],w[6],w[7]);

half\_adder ha3(w[1],w[2],w[8],w[9]);

assign x={w[11],w[9],w[7],w[5],p[3],p[1]},y={p[15],w[10],w[8],w[6],w[4],p[2]};

rca r1(x,y,1'b0,s[7:1]);

endmodule

module HA\_CSLA\_BEC1(A,B,Cin,Y,Cout);

input A,B,Cin;

output Y,Cout;

wire a,b,c,d,e,f;

half\_adder h1(A,B,a,b);

assign c= ~a;

assign e = a^b;

mux\_2to1 m1(c,a,Cin,Y);

mux\_2to1 m2(e,b,Cin,Cout);

endmodule

module half\_adder(A,B,Y,Cout);

input A,B;

output Y,Cout;

assign {Cout,Y}=A+B;

endmodule

module ppg(p,a,b);

input[3:0]a,b;

output [15:0]p;

and (p[0],a[0],b[0]),(p[1],a[1],b[0]),(p[2],a[0],b[1]),(p[3],a[2],b[0]),(p[4],a[0],b[2]),(p[5],a[1],b[1]),(p[6],a[2],b[1]),(p[7],a[1],b[2]),(p[8],a[0],b[3]),(p[9],a[3],b[0]),(p[10],a[3],b[1]),(p[11],a[1],b[3]),(p[12],a[2],b[2]),(p[13],a[3],b[2]),(p[14],a[2],b[3]),(p[15],a[3],b[3]);

endmodule

module rca(

input [5:0]a,b,

input cin,

output [6:0]sum);

wire c1,c2,c3,c4,c5; //Carry out of each full adder

assign cin=1'b0;

HA\_CSLA\_BEC1 fa0(a[0],b[0],cin,sum[0],c1);

HA\_CSLA\_BEC1 fa1(a[1],b[1],c1,sum[1],c2);

HA\_CSLA\_BEC1 fa2(a[2],b[2],c2,sum[2],c3);

HA\_CSLA\_BEC1 fa3(a[3],b[3],c3,sum[3],c4);

HA\_CSLA\_BEC1 fa4(a[4],b[4],c4,sum[4],c5);

HA\_CSLA\_BEC1 fa5(a[5],b[5],c5,sum[5],sum[6]);

endmodule

module mux\_2to1 (

input wire A, // Input A

input wire B, // Input B

input wire S, // Select input

output wire Y // Output

);

assign Y = (S == 1'b1) ? A : B; // MUX logic

endmodule