```
`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////
// Company: TEC
// Engineer: Giovanni Villalobos Quirós,Eduardo Ortiz,Adriana Chavarría, César Ortega
//
// Create Date:    19:16:01 09/17/2016
// Design Name:         thirtyTwoBitsFullAdder
// Module Name:    thirtyTwoBitsFullAdder
// Project Name:         I investigacion
// Target Devices: Nexys 2
// Description:  Sumador de 32 bits para la implementacion de una arquitectura uniciclo MIPS
//
//
//Target Devices:
// ------------------------------------------------
// | Device |
// ------------------------------------------------
// | Family | Spartan3e |
// | Part | xc3s500e |
// | Package | fg320 |
// | Temp Grade | Commercial |
// | Process | Typical |
// | Speed Grade | -4 |
// | Characterization | PRODUCTION,v1.2,06-23-09 |
// ------------------------------------------------
//
//
// | IOB Name | Descripcion | Type | Direction | IO Standard | Drive | Slew | IOB |
// | | | | | | Strength | Rate | Delay |
// +-----------------------------------------------------------------------------------------------------+
// | a[0-31] | Dato a operar A | IBUF | INPUT | LVCMOS25 | | | 0 / 0 |
// | b[0-31] | Dato a operar B | IBUF | INPUT | LVCMOS25 | | | 0 / 0 |
// | c0 | Carry de entrada | IBUF | INPUT | LVCMOS25 | | | 0 / 0 |
// | cout | Carry de salida | IOB | OUTPUT | LVCMOS25 | 12 | SLOW | 0 / 0 |
// | s[0-31] | Resultado de la operacion | IOB | OUTPUT | LVCMOS25 | 12 | SLOW | 0 / 0 |
// +-----------------------------------------------------------------------------------------------------+
// Delay: 28.952ns
/// Area dentro del dispositivo:
//
// Number of occupied Slices Used: 41, Available: 4,656 Utilization: 1%
//
// Number of bonded IOBs Used:96, Available: 232, Utilization:41%
//
// Number of 4 input LUTs Used:70 Available:9,312 , Utilization: 1%
// Energy: 81mW
```

```
////////////////////////////////////////////////////////////////////////////////
module thirtyTwoBitsFullAdder(
    input [31:0] a,
    input [31:0] b,
    output [31:0] s,
            input c0,
            output cout
    );

            wire c1,c2,c3,c4,c5,c6,c7;
            //Instanciacion de 8 sumadores de 4 bits;
           fourbitFullAdder fbfa1 (.a(a[3:0]), .b(b[3:0]),.c0(0),.c4(c1),.s(s[3:0])
           );

           fourbitFullAdder fbfa2 (.a(a[7:4]),.b(b[7:4]),.c0(c1),.c4(c2),.s(s[7:4])
           );

           fourbitFullAdder fbfa3 (.a(a[11:8]),.b(b[11:8]),.c0(c2),.c4(c3),.s(s[11:8])
           );

           fourbitFullAdder fbfa4 (.a(a[15:12]),.b(b[15:12]),.c0(c3),.c4(c4),.s(s[15:12])
           );

           fourbitFullAdder fbfa5 (.a(a[19:16]),.b(b[19:16]),.c0(c4),.c4(c5),.s(s[19:16])
           );

           fourbitFullAdder fbfa6 (.a(a[23:20]), .b(b[23:20]),.c0(c5),.c4(c6), .s(s[23:20])
           );

           fourbitFullAdder fbfa7 (.a(a[27:24]), .b(b[27:24]),.c0(c6), .c4(c7),.s(s[27:24])
           );

           fourbitFullAdder fbfa8 (.a(a[31:28]), .b(b[31:28]), .c0(c7), .c4(cout),.s(s[31:28])
           );

endmodule

module halfAdder(
           input wire a,
           input wire b,
           output wire s,
           output wire c
    );
           //Ecuacion de la salida sumada
           xor(s,a,b);
           //Ecuacion para el carry de salida
```

```
        and(c,a,b);
endmodule


module fourbitFullAdder(
   input [3:0] a,
   input [3:0] b,
   input c0,
        output wire c4,
   output [3:0] s
        );
        wire [3:0] c;
        wire [3:0] p;
        wire [3:0] g;
        //Instanciacion de 4 half adders para hacer un solo sumador completo
        // de 4 bits

        halfAdder ha1 (
                .a(a[0]),
                .b(b[0]),
                .s(p[0]),
                .c(g[0])
        );

        halfAdder ha2 (
                .a(a[1]),
                .b(b[1]),
                .s(p[1]),
                .c(g[1])
        );

        halfAdder ha3 (
                .a(a[2]),
                .b(b[2]),
                .s(p[2]),
                .c(g[2])
        );

        halfAdder ha4 (
                .a(a[3]),
                .b(b[3]),
                .s(p[3]),
                .c(g[3])
        );

       //Se instancia la unidad del carry de 4 bits anticipado
```

```
	carryUnit4Bits uut (
		.p(p),
		.g(g),
		.c0(c0),
		.c(c)
	);
	//Cada uno de estos xor representa la salida de los 4 digitos del sumador.
	xor(s[0],c0,p[0]);
	xor(s[1],c[0],p[1]);
	xor(s[2],c[1],p[2]);
	xor(s[3],c[2],p[3]);
	or(c4,c[3],0);
endmodule

module carryUnit4Bits(
	input wire [3:0] p,
	input wire [3:0] g,
	input wire c0,
	output wire [3:0] c
);

wire [9:0] z;
	//Ecuaciones para obtener acarreo1
	//c1 = G0 + P0c0
	and(z[0],p[0],c0);
	or(c[0],g[0],z[0]);

	//Ecuaciones para obtener acarreo2
	//c2 = G1 + P1G0 + P1P0c0
	and(z[1],p[1],g[0]);
	and(z[2],z[0],p[1]);
	or(c[1],g[1],z[1],z[2]);

	//Ecuaciones para obtener acarreo3
	//c3 = G2 + P2G1+ P2P1G0 + P2P1P0c0
	and(z[3],p[2],g[1]);
	and(z[4],p[2],z[1]);
	and(z[5],p[2],z[2]);
	or(c[2],g[2],z[3],z[4],z[5]);


	//Ecuaciones para obtener acarreo4
	//c4 = G3 + P3G2+ P3P2G1+P3P2P1G0+  P3 P2P1P0c0
	and(z[6],p[3],g[2]);
	and(z[7],p[3],z[3]);
	and(z[8],p[3],z[4]);
```

```
        and(z[9],p[3],z[5]);
        or(c[3],g[3],z[6],z[7],z[8],z[9]);


endmodule
```