Programa Fuente ALU32bits

```
//////////////////////////////////////////////////////////////////////////////////
// Company: TEC
// Engineer: Eduardo Ortiz Jimenez
//
// Create Date:    21:16:34 09/17/2016
// Design Name:            ALU32bits
// Module Name:    ALU32bits
// Project Name:           I investigacion
// Description:   ALU de 3 bits para la implementacion de una arquitectura uniciclo MIPS
// Dependencies: ALU1bit.v, ALU1bit31.v
//
// Target Devices:
// ------------------------------------------------
// |                Device                 |
// ------------------------------------------------
// | Family          | Spartan3e              |
// | Part            | xc3s500e               |
// | Package         | fg320                  |
// | Temp Grade      | Commercial             |
// | Process         | Typical                |
// | Speed Grade     | -4                     |
// | Characterization | PRODUCTION,v1.2,06-23-09 |
// ------------------------------------------------
//
// Area dentro del dispositivo:
//
// Number of Slices:                  93  out of   4656     1%
// Number of 4 input LUTs:            171  out of   9312     1%
// Number of IOs:                     103
// Number of bonded IOBs:             103  out of    232    44%
//
// Tiempo de retardo:
//
// 49.450ns
//
// Entradas y Salidas:
//
// +-----------------------------------------------------------------------------------------------------+
// | 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    |
// | Cin       | Carry de entrada         | IBUF | INPUT     | LVCMOS25    |          |      | 0 / 0    |
// | Cout      | Carry de salida          | IOB  | OUTPUT    | LVCMOS25    | 12       | SLOW | 0 / 0    |
// | Func[0-3] | Señales de control       | IBUF | INPUT     | LVCMOS25    |          |      | 0 / 0    |
```

```
// | Y[0-31]   | Resultado de la operacion  | IOB  | OUTPUT    | LVCMOS25    | 12       | SLOW | 0 / 0    |
// | Zero      | Bandera de cero            | IOB  | OUTPUT    | LVCMOS25    | 12       | SLOW | 0 / 0    |
// +-------------------------------------------------------------------------------------------------------+
//
// Consumo de energía:
//
// ----------------------------------------------------
// |              Power Supply Summary                |
// ----------------------------------------------------
// |                   | Total | Dynamic | Static Power |
// ----------------------------------------------------
// | Supply Power (mW)    | 80.98 | 0.00    | 80.98        |
// ----------------------------------------------------
//////////////////////////////////////////////////////////////////////////////////
module ALU32bits( input [31:0] A,input [31:0] B, input Cin,input [3:0] Func,output Zero,output
Cout,output [31:0] Y);

wire c1,c2,c3,c4,c5,c6,c7,c8,c9,c10,c11,c12,c13,c14,c15,c16;
wire c17,c18,c19,c20,c21,c22,c23,c24,c25,c26,c27,c28,c29,c30,c31,c32;

//Para la operacion Substract
and(c33,Func[2],Func[1]);
or(c34,Cin,c33);

//Instanciamiento de 32 ALUs de 1 bit
ALU1bit A0(.A(A[0]),.B(B[0]),.Cin(c34),.Op(Func),.Less(c32),.Cout(c1),.Y(Y[0]));
ALU1bit A1(.A(A[1]),.B(B[1]),.Cin(c1),.Op(Func),.Less(1'b0),.Cout(c2),.Y(Y[1]));
ALU1bit A2(.A(A[2]),.B(B[2]),.Cin(c2),.Op(Func),.Less(1'b0),.Cout(c3),.Y(Y[2]));
ALU1bit A3(.A(A[3]),.B(B[3]),.Cin(c3),.Op(Func),.Less(1'b0),.Cout(c4),.Y(Y[3]));
ALU1bit A4(.A(A[4]),.B(B[4]),.Cin(c4),.Op(Func),.Less(1'b0),.Cout(c5),.Y(Y[4]));
ALU1bit A5(.A(A[5]),.B(B[5]),.Cin(c5),.Op(Func),.Less(1'b0),.Cout(c6),.Y(Y[5]));
ALU1bit A6(.A(A[6]),.B(B[6]),.Cin(c6),.Op(Func),.Less(1'b0),.Cout(c7),.Y(Y[6]));
ALU1bit A7(.A(A[7]),.B(B[7]),.Cin(c7),.Op(Func),.Less(1'b0),.Cout(c8),.Y(Y[7]));
ALU1bit A8(.A(A[8]),.B(B[8]),.Cin(c8),.Op(Func),.Less(1'b0),.Cout(c9),.Y(Y[8]));
ALU1bit A9(.A(A[9]),.B(B[9]),.Cin(c9),.Op(Func),.Less(1'b0),.Cout(c10),.Y(Y[9]));
ALU1bit A10(.A(A[10]),.B(B[10]),.Cin(c10),.Op(Func),.Less(1'b0),.Cout(c11),.Y(Y[10]));
ALU1bit A11(.A(A[11]),.B(B[11]),.Cin(c11),.Op(Func),.Less(1'b0),.Cout(c12),.Y(Y[11]));
ALU1bit A12(.A(A[12]),.B(B[12]),.Cin(c12),.Op(Func),.Less(1'b0),.Cout(c13),.Y(Y[12]));
ALU1bit A13(.A(A[13]),.B(B[13]),.Cin(c13),.Op(Func),.Less(1'b0),.Cout(c14),.Y(Y[13]));
ALU1bit A14(.A(A[14]),.B(B[14]),.Cin(c14),.Op(Func),.Less(1'b0),.Cout(c15),.Y(Y[14]));
ALU1bit A15(.A(A[15]),.B(B[15]),.Cin(c15),.Op(Func),.Less(1'b0),.Cout(c16),.Y(Y[15]));
ALU1bit A16(.A(A[16]),.B(B[16]),.Cin(c16),.Op(Func),.Less(1'b0),.Cout(c17),.Y(Y[16]));
ALU1bit A17(.A(A[17]),.B(B[17]),.Cin(c17),.Op(Func),.Less(1'b0),.Cout(c18),.Y(Y[17]));
ALU1bit A18(.A(A[18]),.B(B[18]),.Cin(c18),.Op(Func),.Less(1'b0),.Cout(c19),.Y(Y[18]));
ALU1bit A19(.A(A[19]),.B(B[19]),.Cin(c19),.Op(Func),.Less(1'b0),.Cout(c20),.Y(Y[19]));
```

```
ALU1bit A20(.A(A[20]),.B(B[20]),.Cin(c20),.Op(Func),.Less(1'b0),.Cout(c21),.Y(Y[20]));
ALU1bit A21(.A(A[21]),.B(B[21]),.Cin(c21),.Op(Func),.Less(1'b0),.Cout(c22),.Y(Y[21]));
ALU1bit A22(.A(A[22]),.B(B[22]),.Cin(c22),.Op(Func),.Less(1'b0),.Cout(c23),.Y(Y[22]));
ALU1bit A23(.A(A[23]),.B(B[23]),.Cin(c23),.Op(Func),.Less(1'b0),.Cout(c24),.Y(Y[23]));
ALU1bit A24(.A(A[24]),.B(B[24]),.Cin(c24),.Op(Func),.Less(1'b0),.Cout(c25),.Y(Y[24]));
ALU1bit A25(.A(A[25]),.B(B[25]),.Cin(c25),.Op(Func),.Less(1'b0),.Cout(c26),.Y(Y[25]));
ALU1bit A26(.A(A[26]),.B(B[26]),.Cin(c26),.Op(Func),.Less(1'b0),.Cout(c27),.Y(Y[26]));
ALU1bit A27(.A(A[27]),.B(B[27]),.Cin(c27),.Op(Func),.Less(1'b0),.Cout(c28),.Y(Y[27]));
ALU1bit A28(.A(A[28]),.B(B[28]),.Cin(c28),.Op(Func),.Less(1'b0),.Cout(c29),.Y(Y[28]));
ALU1bit A29(.A(A[29]),.B(B[29]),.Cin(c29),.Op(Func),.Less(1'b0),.Cout(c30),.Y(Y[29]));
ALU1bit A30(.A(A[30]),.B(B[30]),.Cin(c30),.Op(Func),.Less(1'b0),.Cout(c31),.Y(Y[30]));
ALU1bit31
A31(.A(A[31]),.B(B[31]),.Cin(c31),.Op(Func),.Less(1'b0),.Cout(Cout),.Y(Y[31]),.Set(c32));

//Flag de Cero
nor
Z1(Zero,Y[0],Y[1],Y[2],Y[3],Y[4],Y[5],Y[6],Y[7],Y[8],Y[9],Y[10],Y[11],Y[12],Y[13],Y[14],Y[15],Y[16],Y
[17],Y[18],Y[19],Y[20],Y[21],Y[22],Y[23],Y[24],Y[25],Y[26],Y[27],Y[28],Y[29],Y[30],Y[31]);

endmodule

module ALU1bit( input A, input B,input Cin, input [3:0] Op,input Less, output Cout,output Y );

wire A_not,B_not;
wire c1,c2;
wire Res,ASel,BSel;

//Negacion de entradas
not(A_not,A);
not(B_not,B);

//Seleccion de entradas
MUX2a1 MUXA(.A(A),.B(A_not),.Select(Op[3]),.Out(ASel));

MUX2a1 MUXB(.A(B),.B(B_not),.Select(Op[2]),.Out(BSel));

//Operacion And 0000
and A1(c1,ASel,BSel);

//Operacion Or 0001 y Nor 1100
or O1(c2,ASel,BSel);

//Operacion Add 0010 y Substract 0110
FullAdder Suma1(.Cin(Cin),.A(ASel),.B(BSel),.Cout(Cout),.Y(Res));
```

```
//MUX seleccion
MUX4a1 MUXSel(.A(c1),.B(c2),.C(Res),.D(Less),.S1(Op[1]),.S2(Op[0]),.Out(Y));

endmodule

module MUX2a1(input A,input B, input Select,output Out);

wire c1,c2,c3;

//Negacion de entradas
not(Select_not,Select);

//Out
and A1(c1,A,Select_not); // ASelect
and A2(c2,B,Select); // BSelect
and A3(c3,A,B); // AB
or O1(Out,c1,c2,c3); // ASelect' + BSelect + AB

endmodule

module FullAdder(input Cin,input A,input B,output Cout,output Y);

wire A_not, B_not, Cin_not;
wire c1,c2,c3,c4,c5,c6,c7;

//Negacion de entradas
not N1(A_not,A);
not N2(B_not,B);
not N3(Cin_not,Cin);

// Funciones Y
and A1(c1,A,B_not,Cin_not); //AB'Cin'
and A2(c2,A,B,Cin); //ABCin
and A3(c3,A_not,B_not,Cin); //A'B'Cin
and A4(c4,A_not,B,Cin_not); // A'BCin'
or O1(Y,c1,c2,c3,c4); // AB'Cin'+ ABCin + A'B'Cin + A'BCin'

//Funciones Cout
and A5(c5,A,Cin); // ACin
and A6(c6,B,Cin); // BCin
and A7(c7,A,B); //AB
or O2(Cout,c5,c6,c7); //ACin + BCin + AB
```

```
endmodule

module MUX4a1(input A, input B,input C,input D,input S1,input S2output Out);

wire c1,c2,c3,c4;

//Negacion de entradas
not(S1_not,S1);
not(S2_not,S2);

//Salida Out
and A1(c1,D,S1,S2); //DS1S2
and A2(c2,C,S1,S2_not); //CS1S2'
and A3(c3,B,S1_not,S2); // BS1'S2
and A4(c4,A,S1_not,S2_not); // AS1'S2'
or O1(Out,c1,c2,c3,c4); // DS1S2 + CS1S2' + BS1'S2 + AS1'S2'

endmodule
```