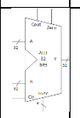
<table>
<tr>
<th>Identificación del componente</th>
<th>Nombre</th>
<th>Símbolo</th>
<th>Descripción</th>
<th>Descripción de entradas y salidas</th>
<th>Tabla de verdad</th>
<th>Simplificación</th>
<th>Funciones Lógicas</th>
<th>Código</th>
<th>Nombre del archivo</th>
</tr>
<tr>
<td>Unidad Lógica Aritmética</td>
<td>ALU32bits</td>
<td>  </td>
<td>Unidad que realiza las operaciones Add, And, Or, Nor, Substract y Set Less Than de la arquitectura MIPS</td>
<td>A y B: datos a operar, de 32 bits.
Y: Resultado de la operación en 32 bits.
Cin: Carry de entrada.
Cout: Carry de salida
Zero: Bandera de indicación de cero.
Func: 4 bits de control.</td>
<td>Dentro de la ALU se necesitan sumadores, y muxes.
Sumador:
<table>
<tr><th>A</th><th>B</th><th>Cin</th><th>Y</th><th>Cout</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table>
MUX2a1:
<table>
<tr><th>A</th><th>B</th><th>Sel</th><th>Out</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table>
MUX4a1:
<table>
<tr><th>A</th><th>B</th><th>C</th><th>D</th><th>S1</th><th>S2</th><th>Out</th></tr>
<tr><td>A</td><td>X</td><td>X</td><td>X</td><td>0</td><td>0</td><td>A</td></tr>
<tr><td>X</td><td>B</td><td>X</td><td>X</td><td>0</td><td>1</td><td>B</td></tr>
<tr><td>X</td><td>X</td><td>C</td><td>X</td><td>1</td><td>0</td><td>C</td></tr>
<tr><td>X</td><td>X</td><td>X</td><td>D</td><td>1</td><td>1</td><td>D</td></tr>
</table>
Control:
<table>
<tr><th>Op</th><th>A’</th><th>B’</th><th>S1</th><th>S2</th></tr>
<tr><td>And</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>Or</td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>Add</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>Sub</td><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>SLT</td><td>0</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>Nor</td><td>1</td><td>1</td><td>0</td><td>0</td></tr>
</table></td>
<td>Sumador:
Y
<table>
<tr><th>BCin
A</th><th>00</th><th>01</th><th>11</th><th>10</th></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td></tr>
</table>
Cout
<table>
<tr><th>BCin
A</th><th>00</th><th>01</th><th>11</th><th>10</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td><td>1</td></tr>
</table>
MUX2a1:
<table>
<tr><th>BSel
A</th><th>00</th><th>01</th><th>11</th><th>10</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td></tr>
</table>
MUX4a1:
Por la tabla simplificada se toma la ecuación directa.</td>
<td>Sumador:
Y=AB’Cin’+ABCin+A’B’Cin+A’BCin’
Cout=ACin+BCin+AB
MUX2a1:
Out=ASel’+BSel+AB
MUX4a1:
Out=DS1S2+CS1S2’+BS1’S2+AS1’S2’
Para función AND:
Y=AB
Para función OR:
Y=A+B
Para función NOR:
Y=A’B’
Para negación de entradas:
Y=A’
Y=B’
Para la función SLT se utiliza una entrada Less y una salida Set, donde la entrada Less de los 31 bits más significativos se conecta a 0 y el bit menos significativo se conecta a la salida Set del más significativo que es el signo de la operación.
Para Substract, se invierte la entrada B y se pone Carry 1 en el bit menos significativo para sumar 1 y obtener el complemento a 2 de B.
Para el Flag Zero se pasan los 31 bits del resultado por una NOR,</td>
<td><pre>
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
</pre></td>
<td>ALU32bits_Ver2_ChavarriaOrtegaOrtizVillalobos.v</td>
</tr>
</table>