4 bit ALU IMPLEMENTATION VHDL

Name : Md. Khairul Hasib
ID: 2015-2-60-105

Cse 360 (sec 3)

module fulladd

(sum1,c_out,a1,b1,c_in1);

output sum1,c_out;

input a1,b1,c_in1;

wire s1,c1,s2;

xor(s1,a1,b1);

and(c1,a1,b1);

xor(sum1,s1,c_in1);

and(s2,s1,c_in1);

xor(c_out,s2,c1);

endmodule

module mux4to1

(out,i0,i1,i2,i3,s1,s0);

output out;

input i0,i1,i2,i3;

input s1,s0;

wire s1n,s0n;

wire y0,y1,y2,y3;

not(s1n,s1);

not(s0n,s0);

and(y0,i0,s1n,s0n);

and(y1,i1,s1n,s0);

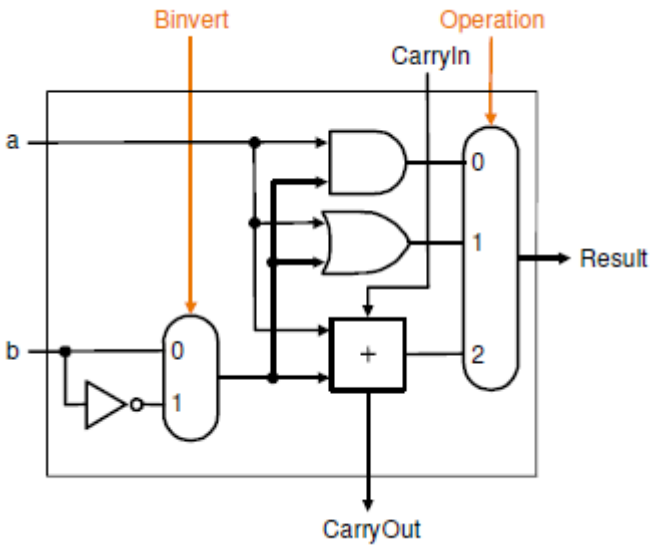
and(y2,i2,s1,s0n);

```
and(y3,i3,s1,s0);
or (out,y0,y1,y2,y3);
endmodule

module ALU
(output result, output cout, input a,b,input cin,binvart,op1,op2);

wire w1,w2,w3,bx,binx;
wire a0,a1,a2,a3;
and(a0,a,b);
or(a1,a,b);
not(bx,b);
not(binx, binvart);
and(w1,b,binx);
and(w2,bx,binvart);
or(w3,w1,w2);
fulladd stage1 ( sum1,c_out,a1,w3,cin);
mux4to1 stage2(result,a0,a1,sum1,a3,op1,op2);

endmodule
```



module ALU4bit

(output [3:0]result, output cout, input [3:0]a,b,input cin,binvart,op1,op2 ,output zero,overflow);

wire cout0,cout1,cout2,cout3,t1;

ALU alu0(result[0],cout0,a[0],b[0],cin,binvart,op1,op2),

alu1(result[1],cout1,a[1],b[1],cout0,binvart,op1,op2),

alu2(result[2],cout2,a[2],b[2],cout1,binvart,op1,op2),

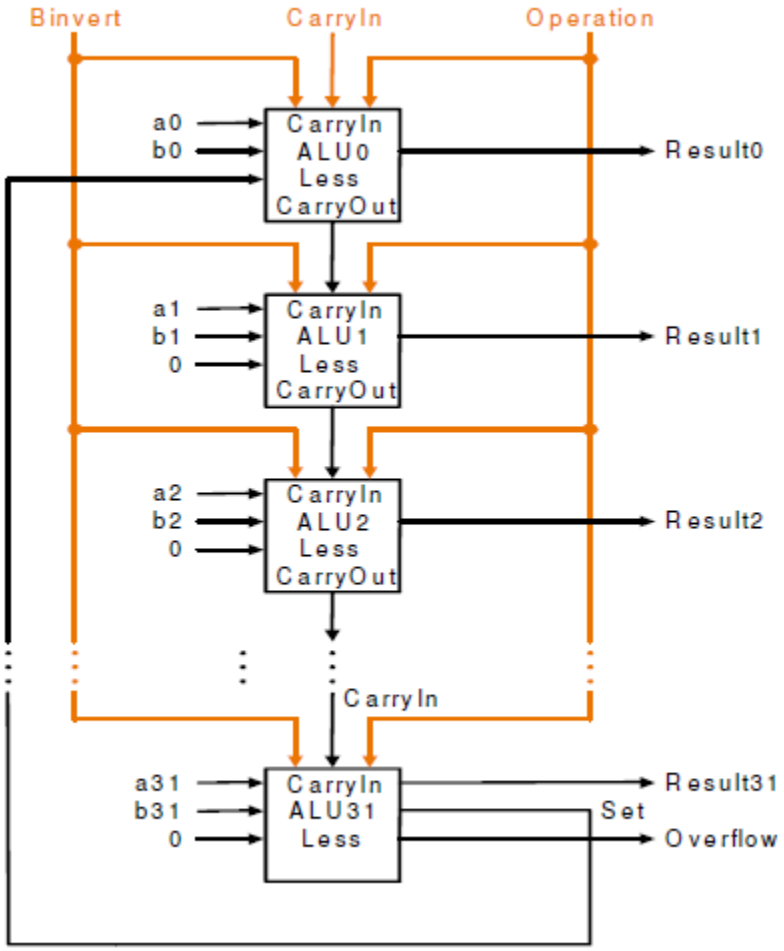
alu3(result[3],cout3,a[3],b[3],cout2,binvart,op1,op2);

or(t1,result[0],result[1],result[2],result[3]);

not(zero,t1);

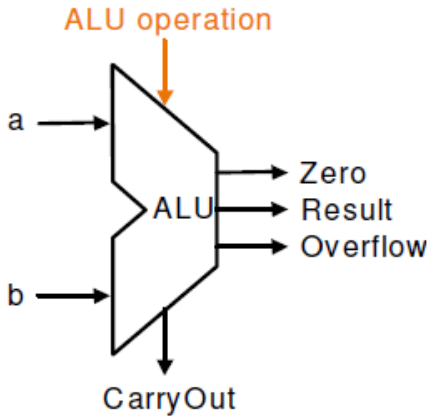
xor(overflow,cout2,cout3);

endmodule



ALU control lines

Bneg-ate	Oper-ation	Func-tion
0	00	and
0	01	or
0	10	add
1	10	sub
1	11	slt



```
module stimulus;
wire [3:0] result;
Wire c_out, zero, overflow;
reg [3:0] a,b;
reg binvart, op1, op2;
alu_4_bit (result, c_out, zero, overflow, a, b, binvart, op1, op2);
initial
begin
$monitor($time, “result[0]=%b, result [1]=%b, result [2]=%b, result [3]=%bb\n”,result[0], result [1], result [2], result [3]);
a[0]=1;a[1]=0;a[2]=0;a[3]=0;
b[0]=0;b[1]=0;b[2]=1;b[3]=1;

#5 binvart=0; op1=0; op2=0
#5 binvart=0; op1=0; op2=1;
#5 binvart=0; op1=1; op2=0;
#5 binvart=0; op1=1; op2=1;
#5 binvart=1; op1=0; op2=0;
#5 binvart=1; op1=0; op2=1;
#5 binvart=1; op1=1; op2=0;
#5 binvart=1; op1=1; op2=1;
end
endmodule
```