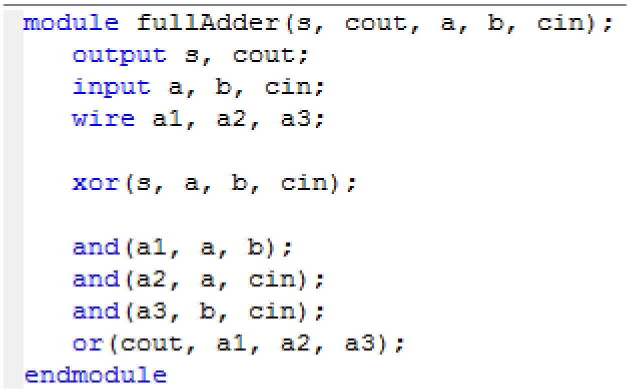
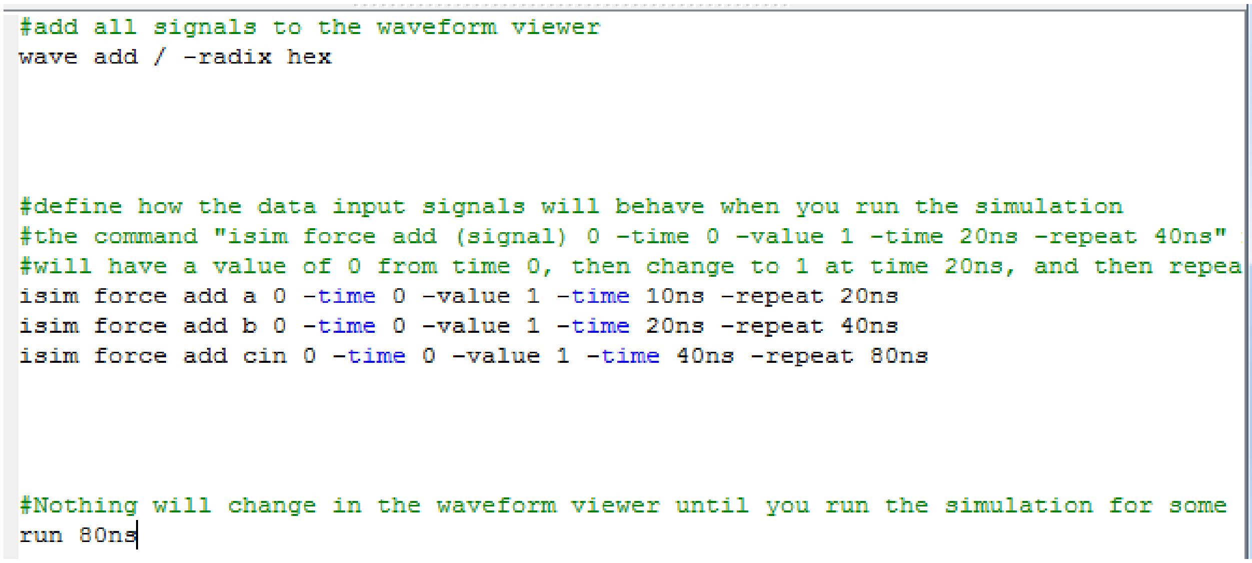
David DeGraw
Lab 5 ALU
2/12/2015

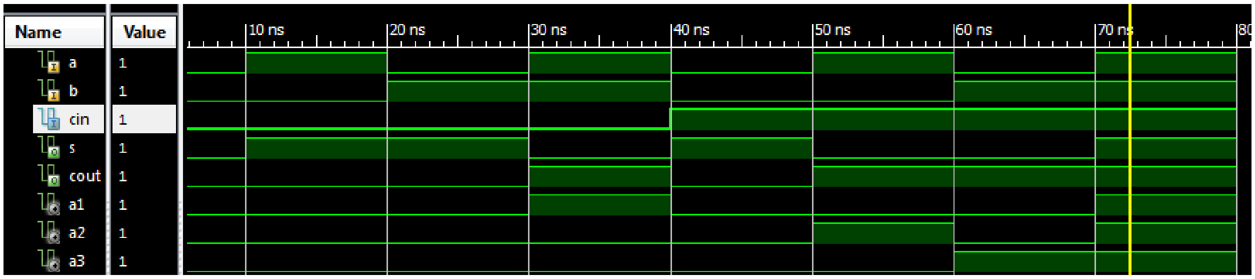
Full Adder



```
module fullAdder(s, cout, a, b, cin);
   output s, cout;
   input a, b, cin;
   wire a1, a2, a3;

   xor(s, a, b, cin);

   and(a1, a, b);
   and(a2, a, cin);
   and(a3, b, cin);
   or(cout, a1, a2, a3);
endmodule
```

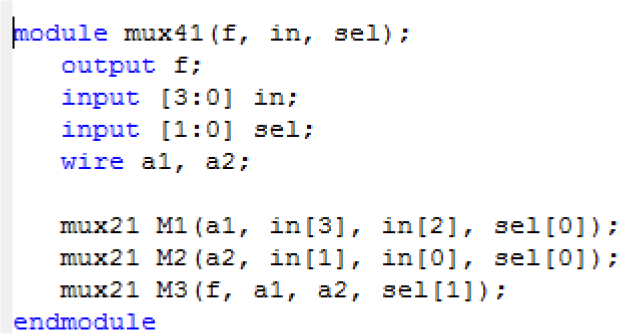


```
#add all signals to the waveform viewer
wave add / -radix hex



#define how the data input signals will behave when you run the simulation
#the command "isim force add (signal) 0 -time 0 -value 1 -time 20ns -repeat 40ns"
#will have a value of 0 from time 0, then change to 1 at time 20ns, and then repea
isim force add a 0 -time 0 -value 1 -time 10ns -repeat 20ns
isim force add b 0 -time 0 -value 1 -time 20ns -repeat 40ns
isim force add cin 0 -time 0 -value 1 -time 40ns -repeat 80ns



#Nothing will change in the waveform viewer until you run the simulation for some
run 80ns
```

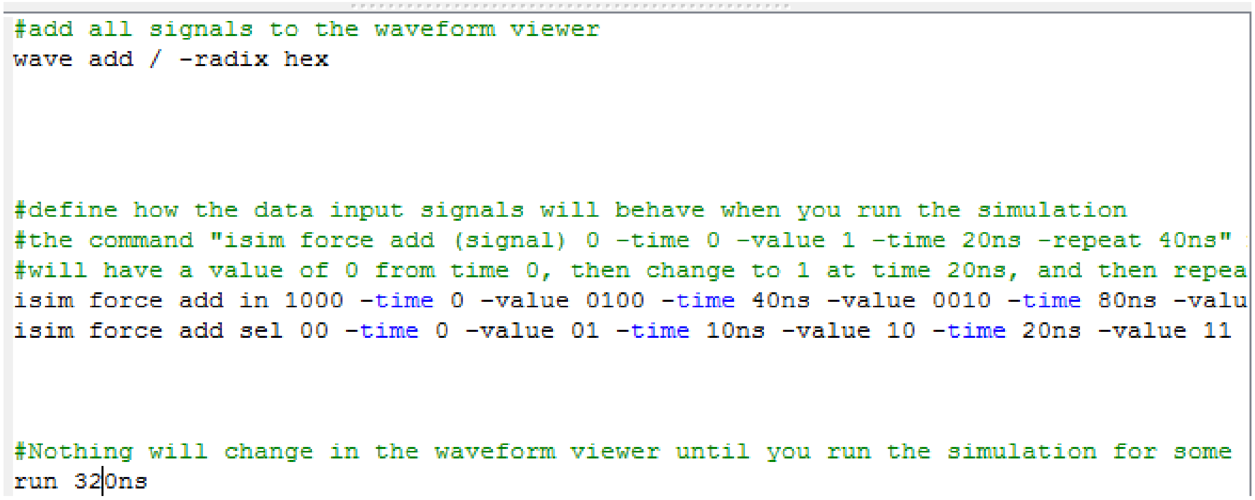


David DeGraw
Lab 5 ALU
2/12/2015

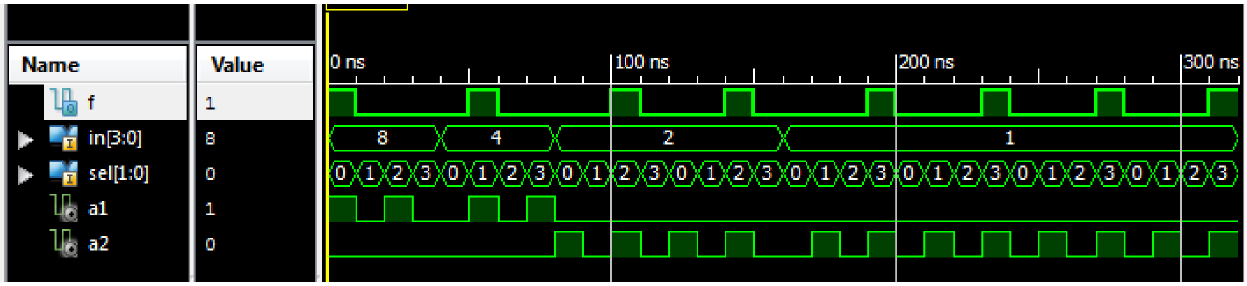
4:1 MUX



```
module mux41(f, in, sel);
   output f;
   input [3:0] in;
   input [1:0] sel;
   wire a1, a2;

   mux21 M1(a1, in[3], in[2], sel[0]);
   mux21 M2(a2, in[1], in[0], sel[0]);
   mux21 M3(f, a1, a2, sel[1]);
endmodule
```

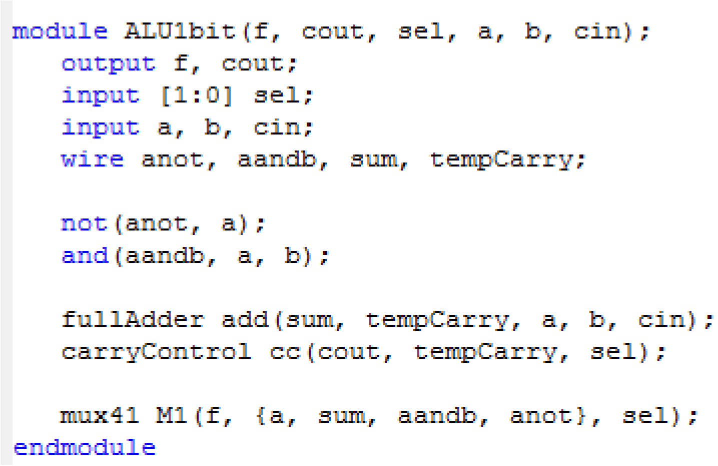


```
#add all signals to the waveform viewer
wave add / -radix hex




#define how the data input signals will behave when you run the simulation
#the command "isim force add (signal) 0 -time 0 -value 1 -time 20ns -repeat 40ns"
#will have a value of 0 from time 0, then change to 1 at time 20ns, and then repea
isim force add in 1000 -time 0 -value 0100 -time 40ns -value 0010 -time 80ns -valu
isim force add sel 00 -time 0 -value 01 -time 10ns -value 10 -time 20ns -value 11



#Nothing will change in the waveform viewer until you run the simulation for some
run 320ns
```

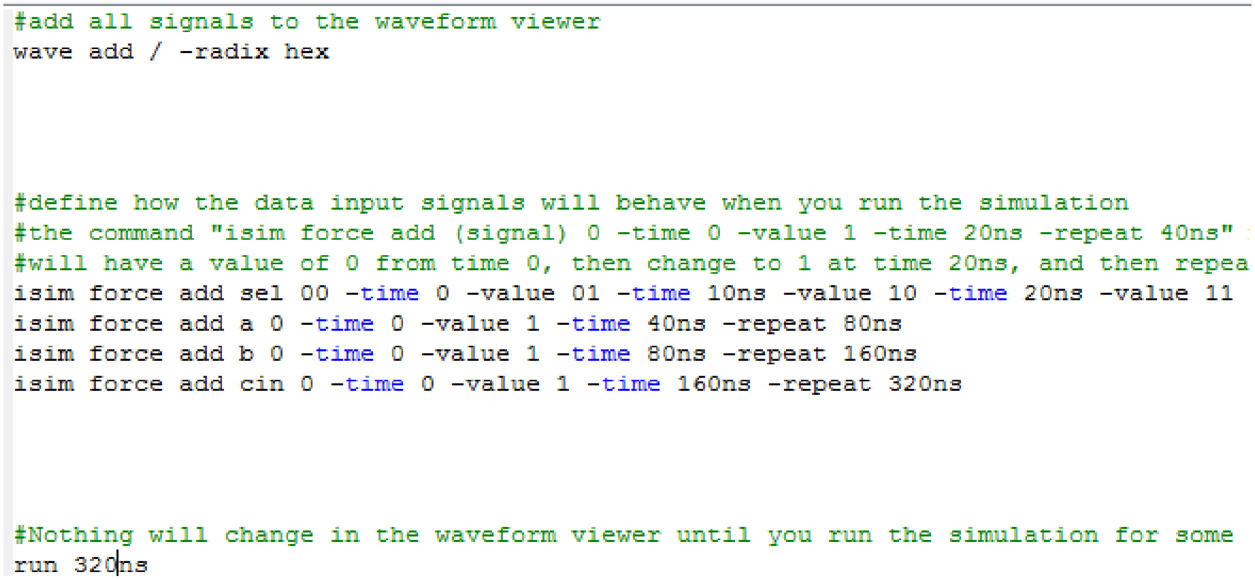


David DeGraw
Lab 5 ALU
2/12/2015

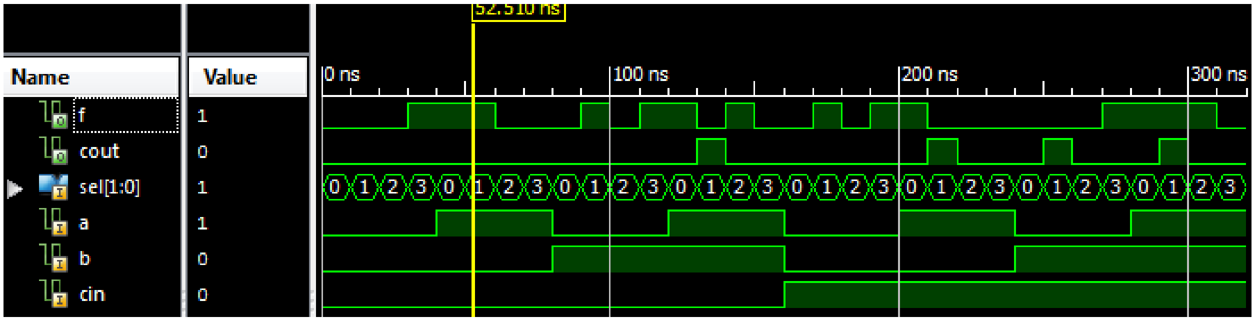
1 Bit ALU



```
module ALU1bit(f, cout, sel, a, b, cin);
   output f, cout;
   input [1:0] sel;
   input a, b, cin;
   wire anot, aandb, sum, tempCarry;

   not(anot, a);
   and(aandb, a, b);

   fullAdder add(sum, tempCarry, a, b, cin);
   carryControl cc(cout, tempCarry, sel);

   mux41 M1(f, {a, sum, aandb, anot}, sel);
endmodule
```

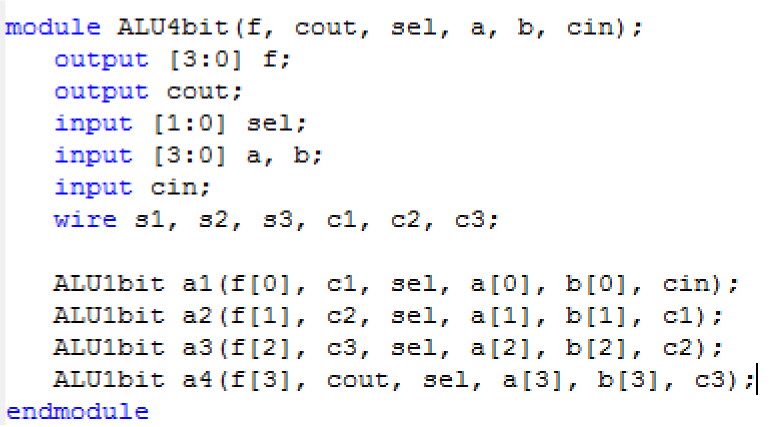


```
#add all signals to the waveform viewer
wave add / -radix hex




#define how the data input signals will behave when you run the simulation
#the command "isim force add (signal) 0 -time 0 -value 1 -time 20ns -repeat 40ns"
#will have a value of 0 from time 0, then change to 1 at time 20ns, and then repea
isim force add sel 00 -time 0 -value 01 -time 10ns -value 10 -time 20ns -value 11
isim force add a 0 -time 0 -value 1 -time 40ns -repeat 80ns
isim force add b 0 -time 0 -value 1 -time 80ns -repeat 160ns
isim force add cin 0 -time 0 -value 1 -time 160ns -repeat 320ns




#Nothing will change in the waveform viewer until you run the simulation for some
run 320ns
```

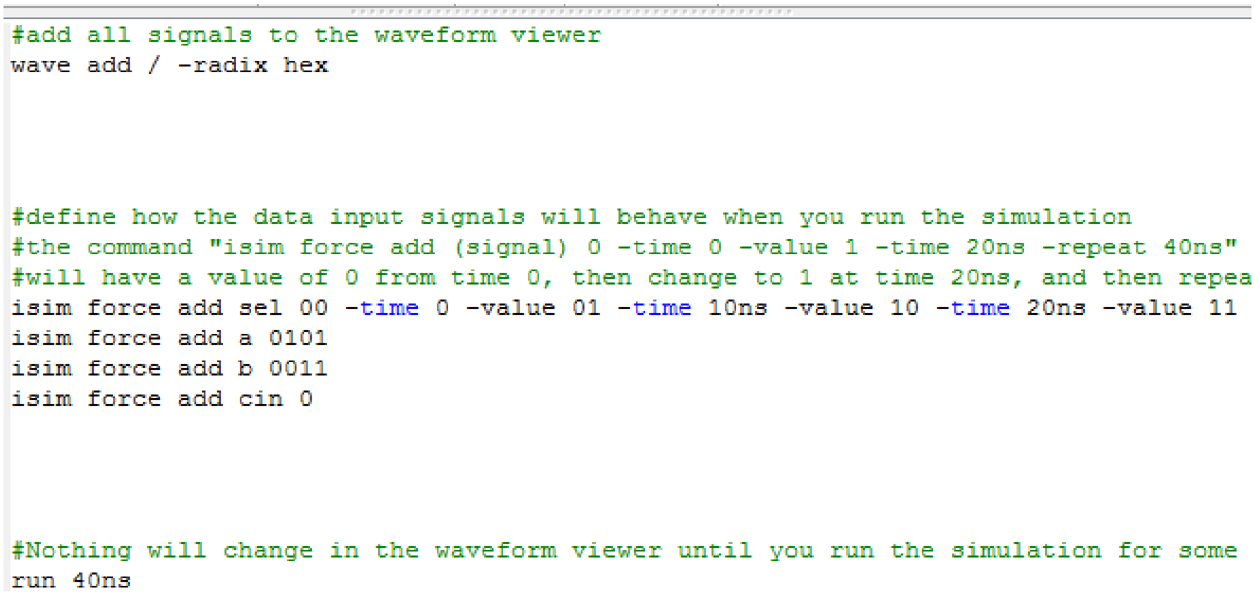


David DeGraw
Lab 5 ALU
2/12/2015

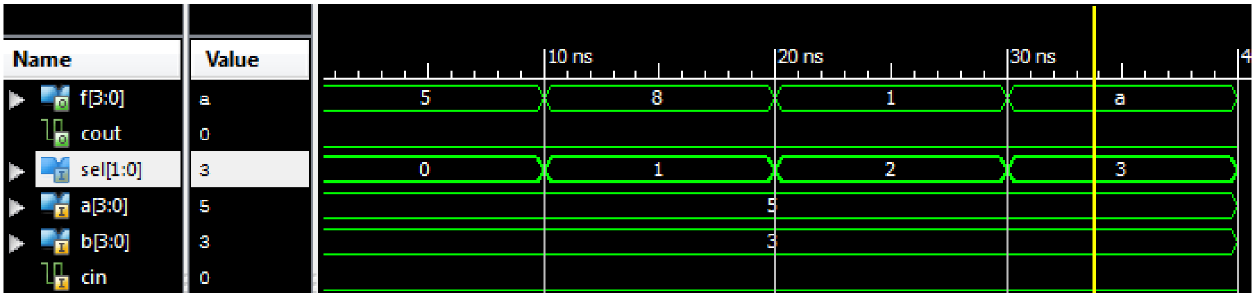
4 Bit ALU



```
module ALU4bit(f, cout, sel, a, b, cin);
   output [3:0] f;
   output cout;
   input [1:0] sel;
   input [3:0] a, b;
   input cin;
   wire s1, s2, s3, c1, c2, c3;

   ALU1bit a1(f[0], c1, sel, a[0], b[0], cin);
   ALU1bit a2(f[1], c2, sel, a[1], b[1], c1);
   ALU1bit a3(f[2], c3, sel, a[2], b[2], c2);
   ALU1bit a4(f[3], cout, sel, a[3], b[3], c3);
endmodule
```

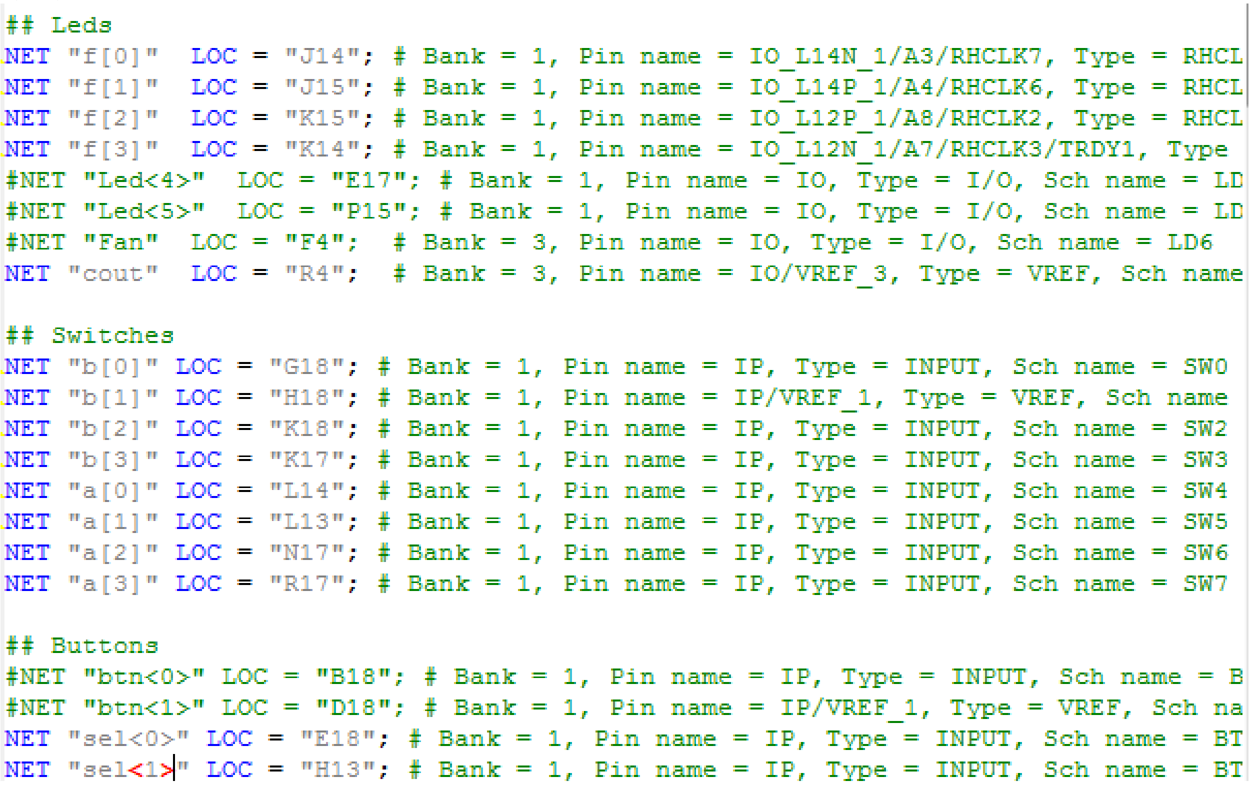


```
#add all signals to the waveform viewer
wave add / -radix hex



#define how the data input signals will behave when you run the simulation
#the command "isim force add (signal) 0 -time 0 -value 1 -time 20ns -repeat 40ns"
#will have a value of 0 from time 0, then change to 1 at time 20ns, and then repea
isim force add sel 00 -time 0 -value 01 -time 10ns -value 10 -time 20ns -value 11
isim force add a 0101
isim force add b 0011
isim force add cin 0



#Nothing will change in the waveform viewer until you run the simulation for some
run 40ns
```



Name	Value
f[3:0]	a
cout	0
sel[1:0]	3
a[3:0]	5
b[3:0]	3
cin	0

David DeGraw
Lab 5 ALU
2/12/2015



```
## Leds
NET "f[0]"  LOC = "J14"; # Bank = 1, Pin name = IO_L14N_1/A3/RHCLK7, Type = RHCL
NET "f[1]"  LOC = "J15"; # Bank = 1, Pin name = IO_L14P_1/A4/RHCLK6, Type = RHCL
NET "f[2]"  LOC = "K15"; # Bank = 1, Pin name = IO_L12P_1/A8/RHCLK2, Type = RHCL
NET "f[3]"  LOC = "K14"; # Bank = 1, Pin name = IO_L12N_1/A7/RHCLK3/TRDY1, Type
#NET "Led<4>"  LOC = "E17"; # Bank = 1, Pin name = IO, Type = I/O, Sch name = LD
#NET "Led<5>"  LOC = "P15"; # Bank = 1, Pin name = IO, Type = I/O, Sch name = LD
#NET "Fan"  LOC = "F4";  # Bank = 3, Pin name = IO, Type = I/O, Sch name = LD6
NET "cout"  LOC = "R4";  # Bank = 3, Pin name = IO/VREF_3, Type = VREF, Sch name

## Switches
NET "b[0]" LOC = "G18"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = SW0
NET "b[1]" LOC = "H18"; # Bank = 1, Pin name = IP/VREF_1, Type = VREF, Sch name
NET "b[2]" LOC = "K18"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = SW2
NET "b[3]" LOC = "K17"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = SW3
NET "a[0]" LOC = "L14"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = SW4
NET "a[1]" LOC = "L13"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = SW5
NET "a[2]" LOC = "N17"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = SW6
NET "a[3]" LOC = "R17"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = SW7

## Buttons
#NET "btn<0>" LOC = "B18"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = B
#NET "btn<1>" LOC = "D18"; # Bank = 1, Pin name = IP/VREF_1, Type = VREF, Sch na
NET "sel<0>" LOC = "E18"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = BT
NET "sel<1>" LOC = "H13"; # Bank = 1, Pin name = IP, Type = INPUT, Sch name = BT
```

Anomalies:

I had a strange error in my 4 bit ALU, but trying to simulate my full adder would fail. I just bagged the whole thing and started a new project from scratch and re-wrote it, and it worked the second time.