Connor Cousineau
2/26/20

Lab 3 Report

In this lab, the objective was to build a Comparator such that based on the input C, it will use either unsigned or signed respectively. One specification was the mandatory use of the always@ block. Inside the always block are multiple strings of if then else statements that determine if A is greater or less than B, else it is equal. This works very well for when C is unsigned. When C is signed there is one extra step that must be taken. We must first determine if the most significant bit is either a one or a zero. Based on the values the numbers are strictly greater or less than one another. If they are equal, then we can use the logic in unsigned as all numbers less than fifteen up until 8 follow the same logic as unsigned.

While using the Always blocks we had to ensure that all cases were covered to ensure no latches were created. Latches are formed when the compiler has to assign values for the user. To avoid this, all outputs were assigned as 1 or 0 based on the logic.

An observation of the RTL schem is that the unsigned logic only needed two mux's to compare the numbers. The signed comparator needs 9. The conclusion is that signed math takes significantly more resources than unsigned.

The over all cost was about 13 LE which is more expensive than the RCA and LACA being 8 and 12 respectively. This makes me come to the conclusion that Comparison is more expensive than adding.

Code ----------

Comparator

```
module Lab3(A,B,C,F1,F2,F3);
input [3:0]A,B;
input C;
output reg F1,F2,F3;

always@(*)
begin
	if(C==0)
	begin
		if(A[3:0] > B[3:0])
			begin
			F1= 1'b1;
			F2= 1'b0;
			F3= 1'b0;
			end
			else if(A[3:0] < B[3:0])
				begin
				F1 = 1'b0;
				F2 = 1'b0;
				F3 = 1'b1;
				end
				else
				begin
```

```
                                        F1=1'b0;
                                        F2=1'b1;
                                        F3=1'b0;
                                end


        end
//C=1
else
        begin
        if(A[3]>B[3])
        begin
        F1 = 0;
        F2=0;
        F3=1;
        end
        else if(A[3]<B[3])
        begin
        F1=1;
        F2=0;
        F3=0;
        end

        else
        begin
        if(A[3:0] > B[3:0])
                        begin
                        F1= 1'b1;
                        F2= 1'b0;
                        F3= 1'b0;
                        end
                        else if(A[3:0] < B[3:0])
                                begin
                                F1 = 1'b0;
                                F2 = 1'b0;
                                F3 = 1'b1;
                                end
                                else
                                begin
                                        F1=1'b0;
                                        F2=1'b1;
                                        F3=1'b0;
                                end

        end
        end
end
endmodule
```

Test Bench

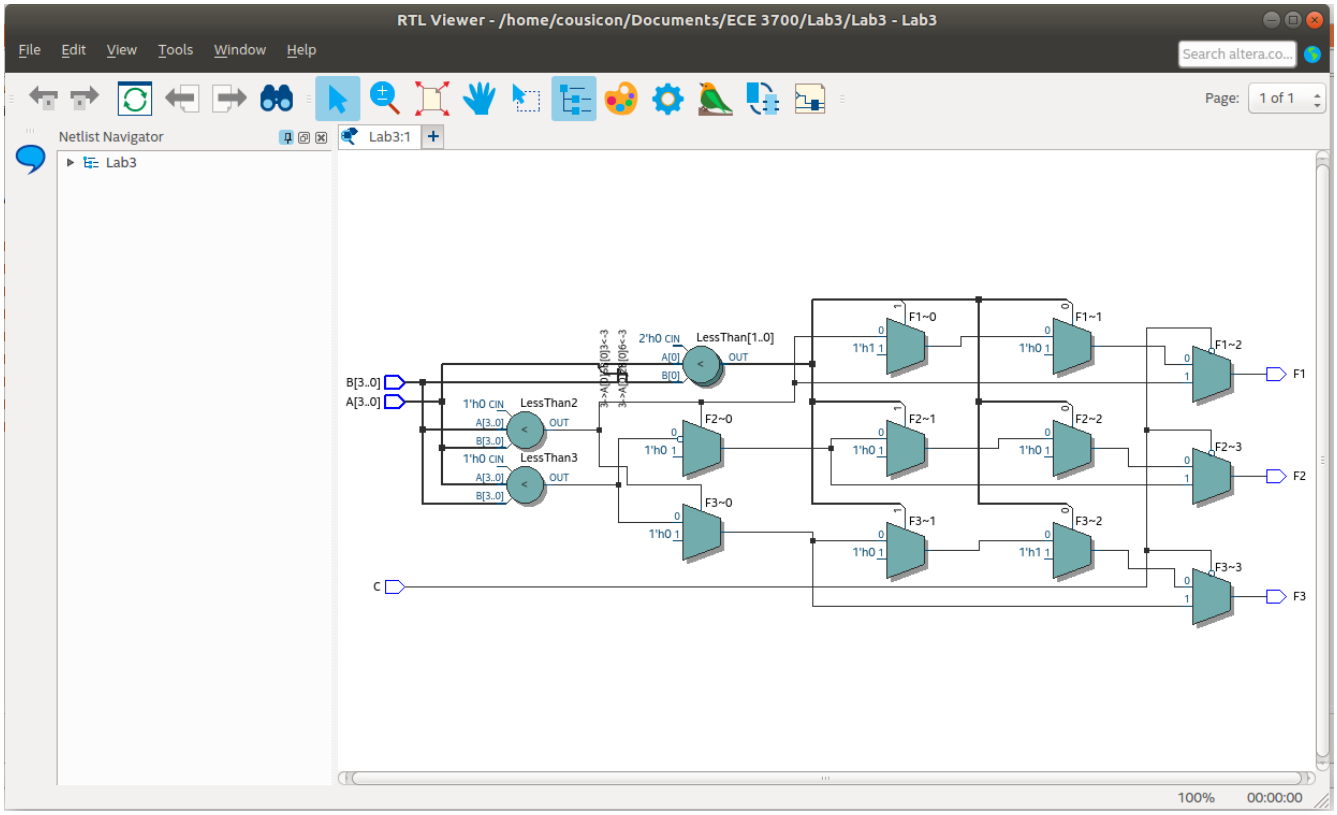
```
`timescale 1ps / 1ps
module Lab3TB;
reg [3:0]A,B;
reg C;
wire F1,F2,F3;
integer i;

Lab3 uut (
.A(A),
.B(B),
.C(C),
.F1(F1),
.F2(F2),
.F3(F3)
);

initial begin
	for(i = 0; i < 2**9; i = i+1)
	begin
	{C, A , B} = i;
	#5;
	$display("%d %d %d => %d %d %d ",A, B, C, F1,F2, F3);
	end
end
endmodule
```

Images

RTL Schem
And LE Report



	Resource	Usage
1	Estimated Total logic elements	13
2		
3	Total combinational functions	13
4	▼ Logic element usage by number of LUT inputs	
1	-- 4 input functions	9
2	-- 3 input functions	4
3	-- <=2 input functions	0
5		
6	▼ Logic elements by mode	
1	-- normal mode	13
2	-- arithmetic mode	0
7		
8	▼ Total registers	0
1	-- Dedicated logic registers	0
2	-- I/O registers	0
9		
10	I/O pins	12
11		
12	Embedded Multiplier 9-bit elements	0
13		
14	Maximum fan-out node	A[3]~input
15	Maximum fan-out	6
16	Total fan-out	63
17	Average fan-out	1.70