Lab 5 report

Contents

Note : if you are to build the Quartus project there is a “N” macro defined at the top of the “lab5.sv” file, changing that changes the number of bits handled. E.g : “`define N 8” for 8 bit.

“Check your understanding”

A : The circuit uses the fact that taking the 2’s complement of a number gets its negative value to find the difference, then because the highest carryout indicates A is less than B and so the result is negative, another 2’s comp conversion can be done to flip the sign of the negative result.

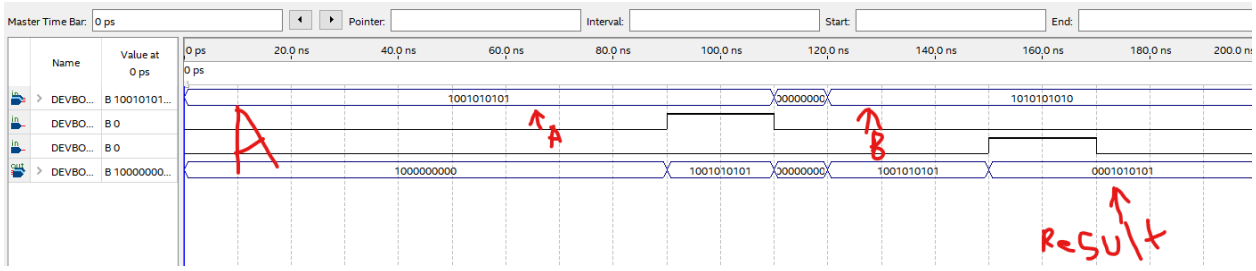
B : cout high indicates value A was less than value B. since the carries are effectively ‘borrows’ so it's just the same as having an extra ‘borrow’. cout low indicates A is greater than B since it was able to satisfy all the ‘borrows’.

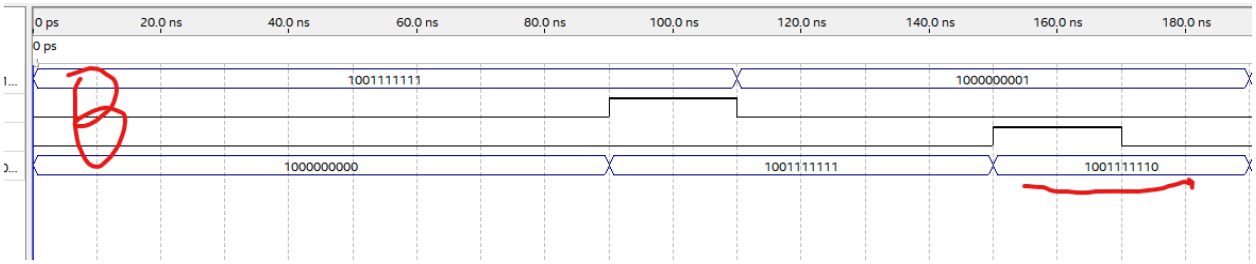
Simulation Results (8 bit)

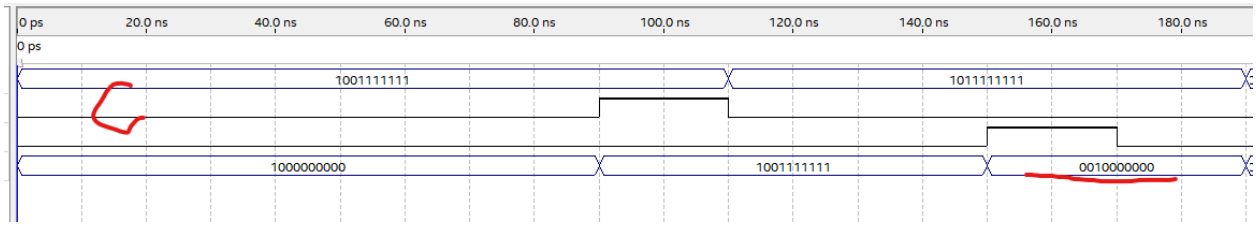
context: A & B values are input from switches, loaded on button press. Switch 9 is held high to show results, but is not part of the input value. Red LED9 indicates Cout.

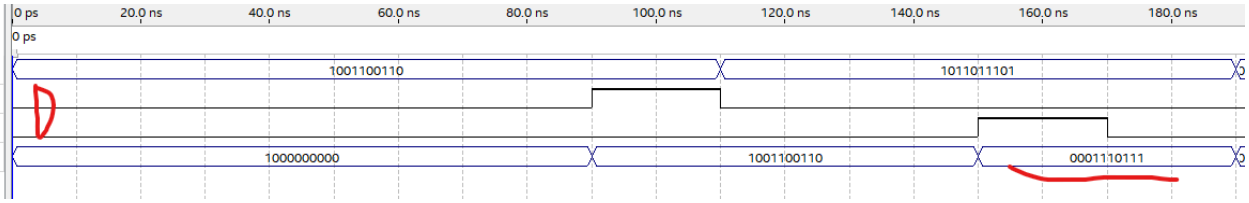
	A	B	R = | A - B |	Cout
a	01010101	10101010	01010101	0
b	01111111	00000001	01111110	1
c	01111111	11111111	10000000	0
d	01100110	11011101	01110111	0

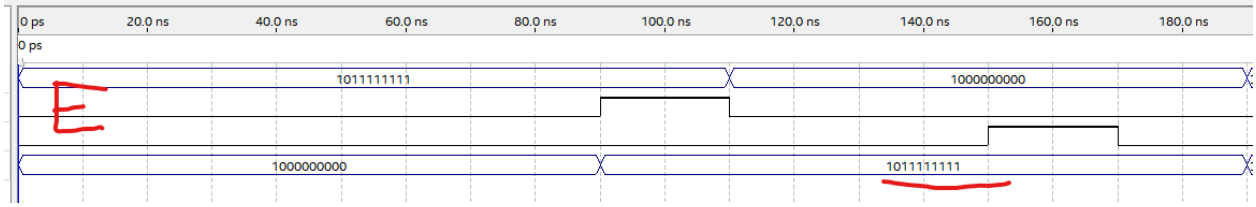
e	11111111	00000000	11111111	1
f	00000000	11111111	11111111	0
g	10101010	01010101	01010101	1
h	11111111	11111111	00000000	1

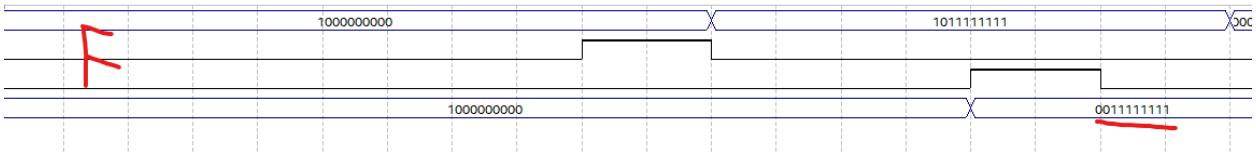

Master Time Bar: 0 ps
Pointer:
Interval:
Start:
End:
Name
Value at 0 ps
DEVBO... B 10010101...
DEVBO... B 0
DEVBO... B 0
DEVBO... B 10000000...
0 ps 20.0 ns 40.0 ns 60.0 ns 80.0 ns 100.0 ns 120.0 ns 140.0 ns 160.0 ns 180.0 ns 200.0 ns
0 ps
1001010101
0000000
1010101010
A
B
1000000000
1001010101
0000000
1001010101
0001010101
Result

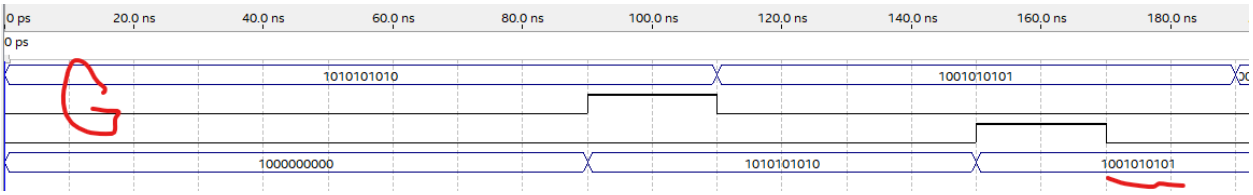

0 ps 20.0 ns 40.0 ns 60.0 ns 80.0 ns 100.0 ns 120.0 ns 140.0 ns 160.0 ns 180.0 ns
0 ps
B
1001111111
1000000001
1000000000
1001111111
1001111110

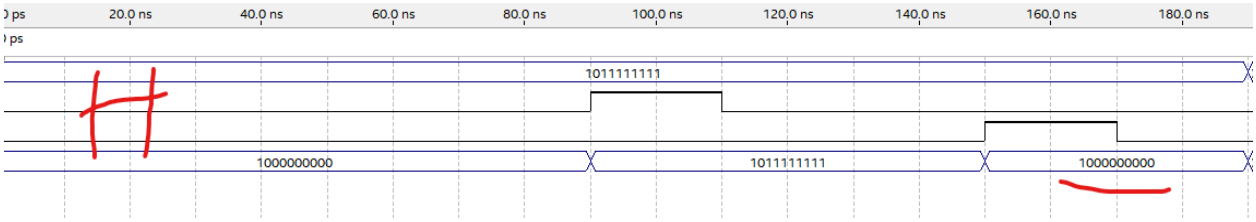

0 ps 20.0 ns 40.0 ns 60.0 ns 80.0 ns 100.0 ns 120.0 ns 140.0 ns 160.0 ns 180.0 ns
0 ps
C
1001111111
1011111111
1000000000
1001111111
0010000000


0 ps 20.0 ns 40.0 ns 60.0 ns 80.0 ns 100.0 ns 120.0 ns 140.0 ns 160.0 ns 180.0 ns
0 ps
D
1001100110
1011011101
1000000000
1001100110
0001110111


0 ps 20.0 ns 40.0 ns 60.0 ns 80.0 ns 100.0 ns 120.0 ns 140.0 ns 160.0 ns 180.0 ns
0 ps
E
1011111111
1000000000
1000000000
1011111111


F
1000000000
1011111111
1000000000
0011111111

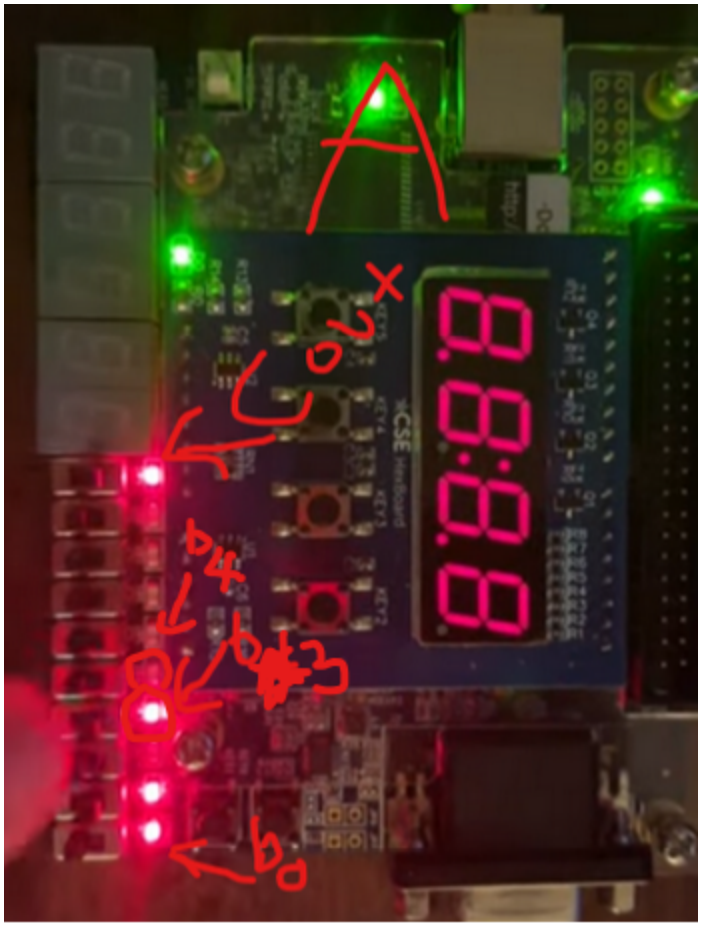

0 ps 20.0 ns 40.0 ns 60.0 ns 80.0 ns 100.0 ns 120.0 ns 140.0 ns 160.0 ns 180.0 ns
0 ps
G
1010101010
1001010101
1000000000
1010101010
1001010101

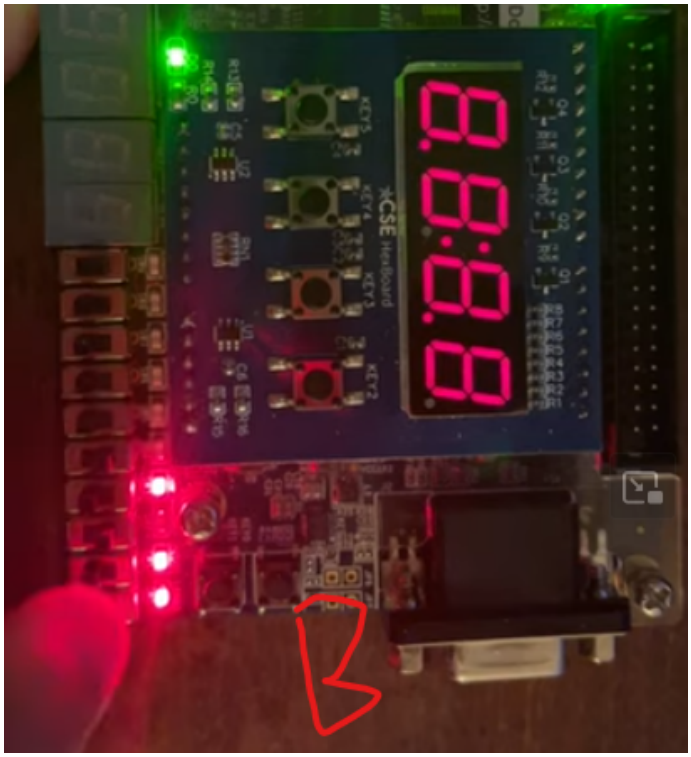


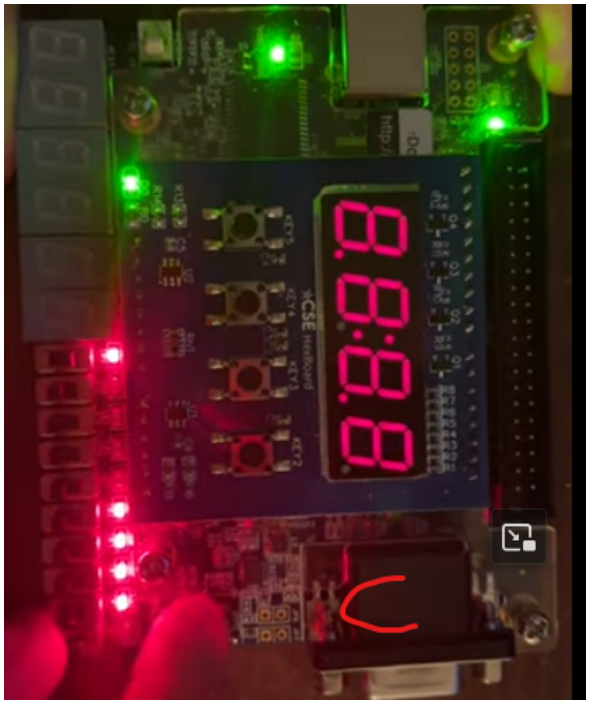
Test Results (5 bit)

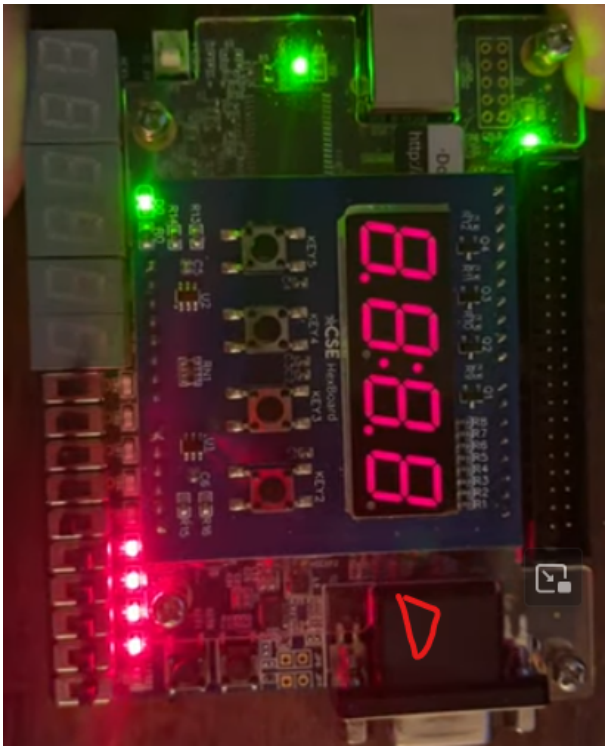
5-bit results recording : https://youtu.be/OHLeVnUSI3M

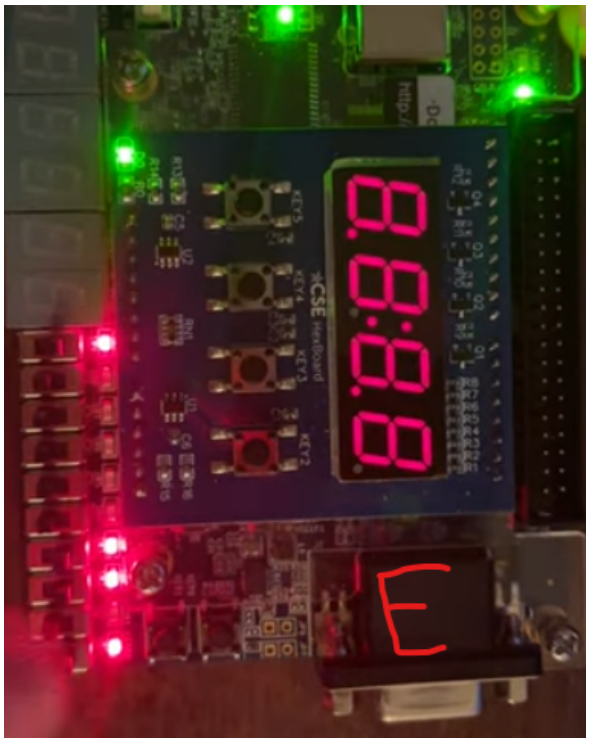
	A	B	R = | A - B |	cout
a	10101	01010	01011	1
b	01010	10101	01011	0
c	11111	00001	11110	1
d	00001	11111	11110	0
e	10110	00000	01101	1
f	10110	10110	00000	1

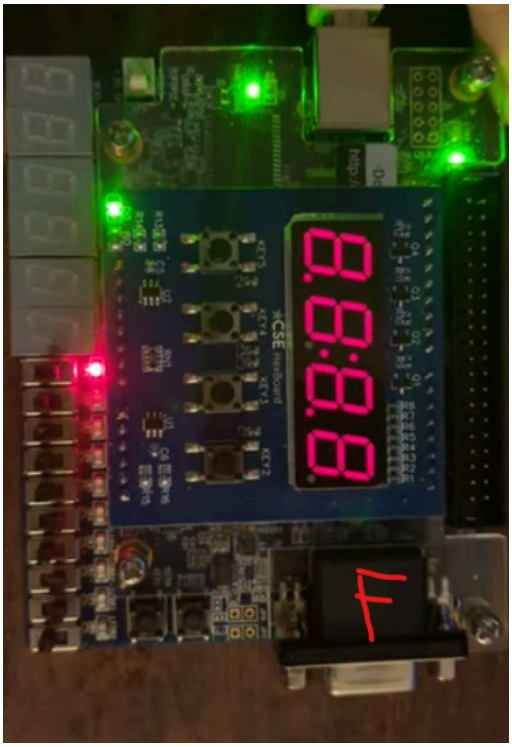

A
x
b4
b3
b0


B


C


D


E


F

Pin Assignments

effective assignments (from top level of lab module)

```
    assign val_in     = DEVBOARD_SWS[0+:`N];
    assign ina        = DEVBOARD_BTN[0];
    assign inb        = DEVBOARD_BTN[1];
    assign showAelseB = DEVBOARD_SWS[8];
    assign showResult = DEVBOARD_SWS[9];

    assign DEVBOARD_RLEDS[9]   = cout;

    //blank the seg displays
    assign DEVBOARD_SEGS = ~0;
```

Actual

	tatu	From	To	Assignment Name	Value	Enabled	Entity	Co
1	✔		in DEVB...N[1]	Location	PIN_A7	Yes		
2	✔		in DEVB...N[0]	Location	PIN_B8	Yes		
3	✔		out DEVB...S[1]	Location	PIN_A9	Yes		
4	✔		out DEVB...S[2]	Location	PIN_A10	Yes		
5	✔		out DEVB...S[3]	Location	PIN_B10	Yes		
6	✔		out DEVB...S[4]	Location	PIN_D13	Yes		
7	✔		out DEVB...S[5]	Location	PIN_C13	Yes		
8	✔		out DEVB...S[6]	Location	PIN_E14	Yes		
9	✔		out DEVB...S[7]	Location	PIN_D14	Yes		
10	✔		out DEVB...S[8]	Location	PIN_A11	Yes		
11	✔		out DEVB...S[9]	Location	PIN_B11	Yes		
12	✔		out DEVB...S[0]	Location	PIN_A8	Yes		
13	✔		in DEVB...S[1]	Location	PIN_C11	Yes		
14	✔		in DEVB...S[2]	Location	PIN_D12	Yes		
15	✔		in DEVB...S[3]	Location	PIN_C12	Yes		
16	✔		in DEVB...S[4]	Location	PIN_A12	Yes		
17	✔		in DEVB...S[5]	Location	PIN_B12	Yes		
18	✔		in DEVB...S[6]	Location	PIN_A13	Yes		
19	✔		in DEVB...S[7]	Location	PIN_A14	Yes		
20	✔		in DEVB...S[8]	Location	PIN_B14	Yes		
21	✔		in DEVB...S[9]	Location	PIN_F15	Yes		
22	✔		in DEVB...S[0]	Location	PIN_C10	Yes		
23	✔		out DEVB...S[1]	Location	PIN_E15	Yes		
24	✔		out DEVB...S[2]	Location	PIN_C15	Yes		
25	✔		out DEVB...S[3]	Location	PIN_C16	Yes		
26	✔		out DEVB...S[4]	Location	PIN_E16	Yes		
27	✔		out DEVB...S[5]	Location	PIN_D17	Yes		
28	✔		out DEVB...S[6]	Location	PIN_C17	Yes		
29	✔		out DEVB...S[7]	Location	PIN_D15	Yes		
30	✔		out DEVB...S[8]	Location	PIN_C18	Yes		
31	✔		out DEVB...S[9]	Location	PIN_D18	Yes		
32	✔		out DEVB...[10]	Location	PIN_E18	Yes		
33	✔		out DEVB...[11]	Location	PIN_B16	Yes		
34	✔		out DEVB...[12]	Location	PIN_A17	Yes		
35	✔		out DEVB...[13]	Location	PIN_A18	Yes		
36	✔		out DEVB...[14]	Location	PIN_B17	Yes		

	tatu	From	To	Assignment Name	Value	Enabled	E
36	✔		out DEVB...[14]	Location	PIN_B17	Yes	
37	✔		out DEVB...[15]	Location	PIN_A16	Yes	
38	✔		out DEVB...[16]	Location	PIN_B20	Yes	
39	✔		out DEVB...[17]	Location	PIN_A20	Yes	
40	✔		out DEVB...[18]	Location	PIN_B19	Yes	
41	✔		out DEVB...[19]	Location	PIN_A21	Yes	
42	✔		out DEVB...[20]	Location	PIN_B21	Yes	
43	✔		out DEVB...[21]	Location	PIN_C22	Yes	
44	✔		out DEVB...[22]	Location	PIN_B22	Yes	
45	✔		out DEVB...[23]	Location	PIN_A19	Yes	
46	✔		out DEVB...[24]	Location	PIN_F21	Yes	
47	✔		out DEVB...[25]	Location	PIN_E22	Yes	
48	✔		out DEVB...[26]	Location	PIN_E21	Yes	
49	✔		out DEVB...[27]	Location	PIN_C19	Yes	
50	✔		out DEVB...[28]	Location	PIN_C20	Yes	
51	✔		out DEVB...[29]	Location	PIN_D19	Yes	
52	✔		out DEVB...[30]	Location	PIN_E17	Yes	
53	✔		out DEVB...[31]	Location	PIN_D22	Yes	
54	✔		out DEVB...[32]	Location	PIN_F18	Yes	
55	✔		out DEVB...[33]	Location	PIN_E20	Yes	
56	✔		out DEVB...[34]	Location	PIN_E19	Yes	
57	✔		out DEVB...[35]	Location	PIN_J18	Yes	
58	✔		out DEVB...[36]	Location	PIN_H19	Yes	
59	✔		out DEVB...[37]	Location	PIN_F19	Yes	
60	✔		out DEVB...[38]	Location	PIN_F20	Yes	
61	✔		out DEVB...[39]	Location	PIN_F17	Yes	
62	✔		out DEVB...[40]	Location	PIN_J20	Yes	
63	✔		out DEVB...[41]	Location	PIN_K20	Yes	
64	✔		out DEVB...[42]	Location	PIN_L18	Yes	
65	✔		out DEVB...[43]	Location	PIN_N18	Yes	
66	✔		out DEVB...[44]	Location	PIN_M20	Yes	
67	✔		out DEVB...[45]	Location	PIN_N19	Yes	
68	✔		out DEVB...[46]	Location	PIN_N20	Yes	
69	✔		out DEVB...[47]	Location	PIN_L19	Yes	
70	✔		out DEVB...S[0]	Location	PIN_C14	Yes	
71		<<new>>	<<new>>	<<new>>			

Verilog Code

Top level lab

```
/**
	cse3341, digital logic 2, lab 5
	George Boone
	1002055713

	modular exponent subtractor
	this module is not part of the logic, it only serves to demo it
*/

`define N 8

module lab5(
	// standard in. i got tired of the assignment editor, see immideatly after wire decelerations for
assignments
	input [9:0] DEVBOARD_SWS,
	input [1:0] DEVBOARD_BTN,
	output[9:0] DEVBOARD_RLEDS,
	output[47:0]DEVBOARD_SEGS
	);


//-------------------------------------------------------------------------
//	INPUT ASSIGMENTS  \ EXTERNAL WIRING
//-------------------------------------------------------------------------

	wire [`N-1:0] val_in;
	wire ina, inb;
	wire cout;
	reg showAelseB, showResult;    //yeah

	assign val_in     = DEVBOARD_SWS[0+:`N];
	assign ina        = DEVBOARD_BTN[0];
	assign inb        = DEVBOARD_BTN[1];
	assign showAelseB = DEVBOARD_SWS[8];
	assign showResult = DEVBOARD_SWS[9];

	assign DEVBOARD_RLEDS[9]          = cout;

	//blank the seg displays
	assign DEVBOARD_SEGS = ~0;

//-------------------------------------------------------------------------
//	INTERNAL WIRING
//-------------------------------------------------------------------------
```

```
        reg [`N:0] valA, valB, diff;

        always @ (posedge ina)
        valA = val_in;

        always @ (posedge inb)
        valB = val_in;

        always @ (showAelseB, showResult)
        if(showResult == 1)
        DEVBOARD_RLEDS[0+:`N] = diff;
        else if(showAelseB == 1)
        DEVBOARD_RLEDS[0+:`N] = valA;
        else
        DEVBOARD_RLEDS[0+:`N] = valB;

//------------------------------------------------------------------------
//      MODULES
//------------------------------------------------------------------------

//there is a bug where valA and/or valB are updated when flicking around the switches.
// i

//works
//      RCSubtractor(
//      .A(valA),
//      .B(valB),
//
//      .D(diff),
//      .COUT(cout)
//      );

//works
//      BintTwosComp(
//      .BIN(valA),
//
//      .TWOSCOMP(diff),
//      .COUT(cout)
//      );

        ExponentSubtractor#(
        .N(`N)
        )(
        .EA(valA),
        .EB(valB),

        .Cout(cout),
        .ED(diff)
        );
```

```
endmodule
```

Top level exponent subtractor

```
/**
cse3341, digital logic 2, lab 5
George Boone
1002055713

hardcoded exponent subtractor according to figure 1 of lab document
*/

module ExponentSubtractor#(
        parameter N = 8
        )(
        input [N-1:0] EA, EB,
        output Cout,
        output [N-1:0] ED
        );

        wire [N-1:0] d, invd;      // difference, inverted-difference

        RCSubtractor #(
        .N(N)
        )       _rcs    (
        .A(EA),
        .B(EB),

        .D(d),
        .COUT(Cout)
        );

        Mux2t1 #(
        .N(N)
        )       _m2t1   (
        .CTRL(Cout),
        .A(invd),
        .B(d),

        .OUT(ED)
        );

        BintTwosComp #(
        .N(N)
        )       _bttc   (
```

```
        .BIN(d),

        .TWOSCOMP(invd)
        );

endmodule
```

Full adder

```
/**
cse3341, digital logic 2, lab 5
George Boone
1002055713
*/

module FullAdder(
        input A, B, CIN,
        output S, COUT
        );

        wire axb;

        assign axb = A ^ B;

        assign S = axb ^ CIN;
        assign COUT = ((axb) & CIN) | (A & B);

endmodule
```

Ripple carry subtractor

```
/**
cse3341, digital logic 2, lab 5
George Boone
1002055713

accepts 2 raw binary nums
*/

module RCSubtractor#(
        parameter N = 8
        )(
        input [N-1:0] A, B,
        output [N-1:0] D,
        output COUT
```

```
        );

        wire [N-1:0] _carries;

        assign COUT = _carries[N-1];

        // ripple carry design
        genvar i;
        generate
        for(i = N-1; i >= 0; i = i-1) begin : adder_array
        FullAdder _fa(
        .A(A[i]),
        .B(~B[i]),
        .CIN(i == 0 ? 1 : _carries[i-1]),
        .S(D[i]),
        .COUT(_carries[i])
        );
        end
        endgenerate

endmodule
```

2 to 1 multiplexer

```
/**
cse3341, digital logic 2, lab 5
George Boone
1002055713
*/

module Mux2t1#(
        parameter N = 8
        )(
        input CTRL,                 // 0:A, 1:B
        input [N-1:0] A, B,
        output wire [N-1:0] OUT
        );

        assign OUT = CTRL ? B : A;

endmodule
```

Binary to 2’s complement converter

```
/**
```

```
cse3341, digital logic 2, lab 5
George Boone
1002055713
*/

module BintTwosComp#(
        parameter N = 8
        )(
        input [N-1:0] BIN,
        output reg [N-1:0] TWOSCOMP,

        // debug
        output reg COUT
        );

        wire [N-1:0] _carries;

        assign COUT = ~_carries[N-1];

        // ripple carry design
        genvar i;
        generate
        for(i = N-1; i >= 0; i = i-1) begin : adder_array
        FullAdder _fa(
        .A(~BIN[i]),
        .B(0),
        .CIN(i == 0 ? 1 : _carries[i-1]),
        .S(TWOSCOMP[i]),
        .COUT(_carries[i])
        );
        end
        endgenerate

endmodule
```