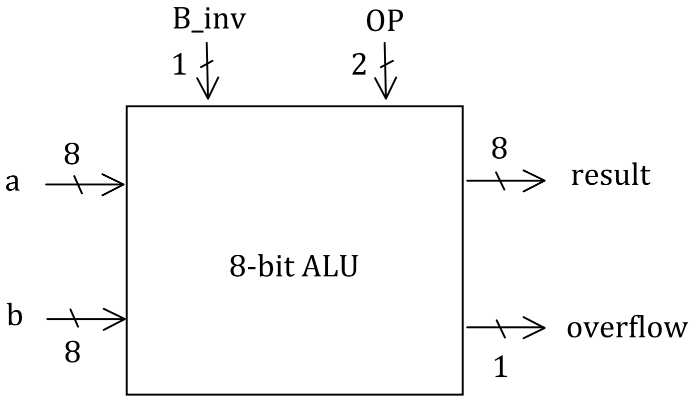
CSCI 113 - Lab 7 (25 points) due: Oct. 29 (Th)

8-bit ALU (AND, OR, ADD, SUB, overflow_checking) design and implementation

Design and implement the 8-bit ALU, which manipulates 8-bit number operations. The ALU should support AND, OR, ADD, SUB and overflow_checking operations. Inputs and outputs are as shown below:



Implement the functionalities based on the following control signal combinations:

B_inv	OP	function
0	00	AND
0	01	OR
0	10	ADD
1	10	SUB

Your final product “8bit_ALU” should include eight 1-bit ALUs, which you designed in the previous lab. In each 1-bit ALU, you should include the B-invert logic to make the b input negated. The last 1-bit ALU (ALU_7, counting from ALU_0) should include the overflow checking logic.

For each of the 8-bit input and output data, use std_logic_vector (7 downto 0) type signal instead of using eight separate 1-bit signals.

Suggested steps to complete the project:
1. Check your 1-bit full adder and 1-bit ALU; if they do not work correctly, fix them first;
2. In your 1-bit ALU, correct the OP (selection) signal if it is different from the followings: 00—AND, 01—OR , 10—ADD;
3. Include the B-invert logic in the 1-bit ALU – this includes 2x1_mux;
4. Design and implement one special 1-bit ALU (ALU_7), checking overflow condition;
5. Make a super component (8-bit ALU) and arrange subcomponents and inputs/outputs;
6. Write a testbench program and test your 8-bit ALU for the functionalities that supports; please use the following set of data to test.

Testing data – *please use the following 5 sets of data in your TB program:*

For each of the following sets of a and b values,
test your 8-bit ALU for AND, OR, ADD and SUB operations.

(1) a = 11111111 , b = 11111111

(2) a = 01111111 , b = 10000000

(3) a = 10000000 , b = 01111111

(4) a = 01010101 , b = 10101010

(5) a = 11011011 , b = 10101010

Thus, there are total 20 cases to test in your TB, i.e., 4 operations for each input set:

```
a= 11111111, b= 11111111, B_inv= 0, OP= 00; --for 11111111 AND 11111111
a= 11111111, b= 11111111, B_inv= 0, OP= 01; --for 11111111 OR   11111111
a= 11111111, b= 11111111, B_inv= 0, OP= 10; --for 11111111 ADD 11111111
a= 11111111, b= 11111111, B_inv= 1, OP= 10; --for 11111111 SUB  11111111

a= 01111111, b= 10000000, B_inv= 0, OP= 00; --for 01111111 AND 10000000
a= 01111111, b= 10000000, B_inv= 0, OP= 01; --for 01111111 OR   10000000
. . . .
```

Output signals to be checked with the wave form are result (8 bits) and overflow (1 bit).
Please do not forget using error checking codes in your test bench program.

CSCI 113 Lab7 – Report *(submit within lab session)* Name:

Draw clearly the global schematic diagram of your 8-bit ALU. Show each subcomponent as a box and show all input/output signal names (actually used in your codes) and their widths (number of bits). You may attach a separate page.

Draw clearly the detailed GATE_LEVEL schematic diagram of the last 1-bit ALU (ALU_7), which check overflow condition. You may attach a separate page.

** Make an archive file (Lab7-firstName-lastName.zip) containing all needed vhdl source codes and the testbench program, and send it to: jpark@csufresno.edu by Oct. 29 (Th). Instructor will compile and run your program.*