EECE 320 – Digital Systems Design

Project #1

8-Bit Arithmetic and Logic Unit (ALU) Design

We would like to design an 8-bit arithmetic and logic unit (ALU) that takes two inputs ***A*** and ***B*** (two's complement numbers) each of size 8 bits and that performs the following operations depending on the values of a control signal C (4 bits).

- **Arithmetic Operations**
  - Addition and subtraction are supposed to be done for ***two's complement*** numbers.
  - You should only build a Two's complement adder and use it for all operations (***Operations #1, #2, #3, #4***).
  - The result of these operations, ***Result***, is an 8-bit vector.
  - Also, add a carry out signal, ***Cout***.

OP #	Control Signals C	Operation
1	0001	A plus B
2	0010	Increment A by 1
3	0011	A minus B
4	0100	Decrement A by 1

- **8-bit Comparison Operations**
  - You should build your ***own*** comparator that check the operands bit-by-bit, and not rely on built-in operators in Verilog, or you can subtract the numbers and decide, but be careful with overflow cases.

OP#	Control Signals C	Operation
5	0110	Minimum of A and B
6	0111	Maximum of A and B

- **Shift Operations**

OP#	Control Signals C	Operation
7	1000	Circular right shift of A
8	1001	Circular left shift of A
9	1010	Right shift of A with feed in 0
10	1011	Left shift of A with feed in 0
11	1100	Right shift of A with MSB replication
12	1101	Left shift of A with LSB replication

Circular right shift of A. You perform right shift and the bit that comes off the least significant position (LSB) on the right rotates to the most significant position (MSB). Red bit rotated right, blue bits shifted right.

Before:	11001101	Before:	01010100
After:	11100110	After:	00101010

Circular left shift of A. You perform left shift and the bit that comes off the most significant position (MSB) on the left rotates to the least significant position (LSB). Red bit rotated left, blue bits shifted left.

Before:	11001101	Before:	01010101
After:	10011011	After:	10101010

Right shift of A with feed in 0. You perform right shift and the LSB bit is dropped. The empty position at the MSB is filled with 0. Red bit is lost, green bit added, blue bits shifted right.

Before:	11001101	Before:	01010100
After:	01100110	After:	00101010

Left shift of A with feed in 0. You perform left shift and the MSB bit is dropped. The empty position at the LSB is filled with 0. Red bit is lost, green bit added, blue bits shifted left.

Before:	11001101	Before:	01010101
After:	10011010	After:	10101010

Right shift of A with MSB replication. You perform right shift and the LSB bit is dropped. The empty position at the MSB is filled with the same previous MSB bit. Red bit dropped; green bit is a duplication of the blue bit.

Before:	11001101	Before:	01010100
After:	11100110	After:	00101010

Left shift of A with LSB replication. You perform left shift and the MSB bit is dropped. The empty position at the LSB is filled with the same previous LSB bit. Red bit dropped; green bit is a duplication of the blue bit.

Before:	11001101	Before:	01010110
After:	10011011	After:	10101100

- **Testing**
  - You should implement a “testbench” that tests the circuit as follows:
    - For operations #1 to #6, use the following numbers:

      A = 0 B = -2
      A = 50 B = 50
      A = 50 B = 75
      A = -50 B = -75
      A = 100 B = -100

    - For operations #7 to #12 with the following numbers:

      A = 11001101

      A = 01010101

      A = 11001101

      A = 01010101

- **Deliverables**
  - You should submit the Verilog files (*.sv) for all the modules of the circuit and the testbench of the complete circuit on Moodle. Do not submit testbenches for individual modules. Additionally, include a PDF file containing the code for all modules, including the testbench, and screenshots of the EPwave viewer showing the results of the different test cases.