CS 413 Arithmetic Logic Unit

You will use Quartus II to build an 8 bit arithmetic logic unit that performs the following functions:

Control Value	**Function**
000	Copy In1 to theResult unchanged
001	Copy In2 to theResult unchanged
010	Add In1 to In2
011	Subtract In2 from In1
100	And In1 and In2
101	Or In1 and In2
110	Shift left In1 by 1 bit
111	Shift right In1 by 1 bit

You are allowed to use either gates/logic schematic, or else Verilog. We suggest that you use Verilog.

You are given a starter lab, in the directory HW Lab ALU. This contains a "starter" Verilog module, and an associated schematic block diagram of the inputs and outputs expected for the ALU:

```
module ALU1(In1,In2, Control, theResult);
  input [7:0] In1, In2;
        input[3:0] Control;
        output [7:0] theResult;

        // Insert your code here

endmodule
```

If you decide to use Verilog, you may find the following code useful:

```
reg [7:0] temp_theResult;


temp_theResult=In1; (this would need to be in an 'if' or 'case' statement)

assign theResult = temp_theResult;
```

Explanation:

temp_theResult is a variable (stored in a register), whereas theResult is net (actual wires). In certain coding situations you may not be able to write some values directly to theResult, they may need to be written to a variable first.

Useful Verilog Hints:

1. You can represent a 4 bit binary number as:
   - 4'b0000

2. A case statement in Verilog would look like this:

```
always @(Control, In1, In2)
begin
        case(Control)
        value0: code statement;
        value1: code statement;
end
```

   - The always block executes all the time (always), as opposed to initial blocks that only execute one time at the beginning of simulation
   - The @ symbol indicates that the always block will be “triggered” whenever the specified variables change

3. Various operators include:
   - + is add
   - - is subtract
   - & is and
   - | is or
   - A<<1 is shift A left by 1 bit
   - A>>1 is shift A right by 1 bit