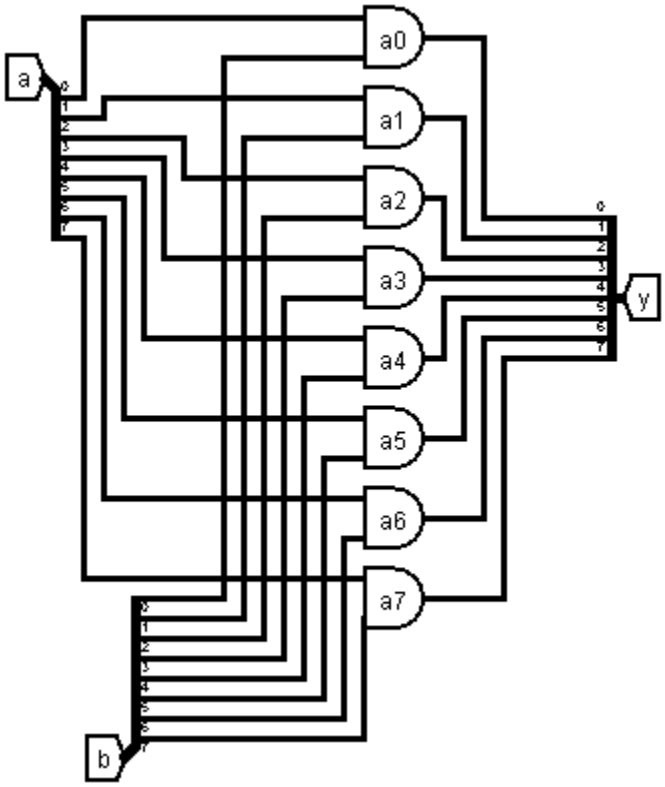
CECS 225: 8-BIT ARITHMETIC LOGIC UNIT

OBJECTIVE: Create an 8-bit ALU (Arithmetic Logic Unit).

PROCEDURE: First import the halfadder, fulladder, ripplecarry4, ripplecarry8, and ripplesubtracter8 module source files from the previous lab. *Be sure to use the exact same file names that were specified in the previous lab.*

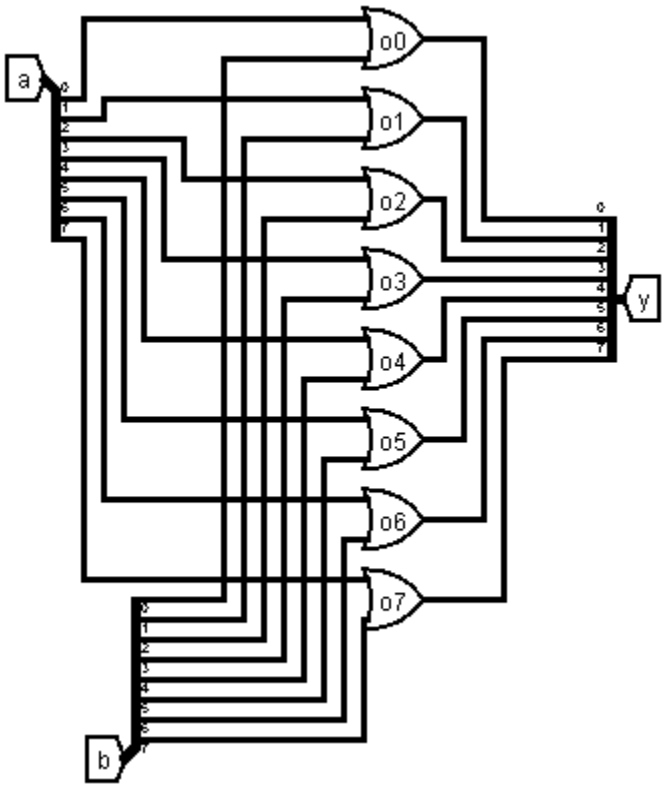
Now create an 8-bit AND gate according to the block diagram below:



Name the file **and8.v** and use the module skeleton below to get started:

```
1  module and8(input [7:0] a,b,
2              output [7:0] y);
3    and
4    a0(y[0],a[0],    ),
5    a1(y[1],a[1],    ),
6    a2(y[2],a[ ],    ),
7    a3(y[3],a[ ],    ),
8    a4(y[4],    ,b[ ]),
9    a5(y[5],    ,b[ ]),
10   a6(y[6],    ,b[6]),
11   a7(y[7],    ,b[7]);
12 endmodule
```

Then create an 8-bit OR gate according to the block diagram below:



Create a file named **or8.v** and use the skeleton code below to get started:

```
module or8(input [7:0] a,b,
           output [7:0] y);
  or
  o0(              ),
  o1(              ),
  o2(              ),
  o3(              ),
  o4(              ),
  o5(              ),
  o6(              ),
  o7(              );
endmodule
```

CECS 225: 8-BIT ARITHMETIC LOGIC UNIT

The next two modules will be given for you to use:

Create **slt8.v** given the Verilog source code below:

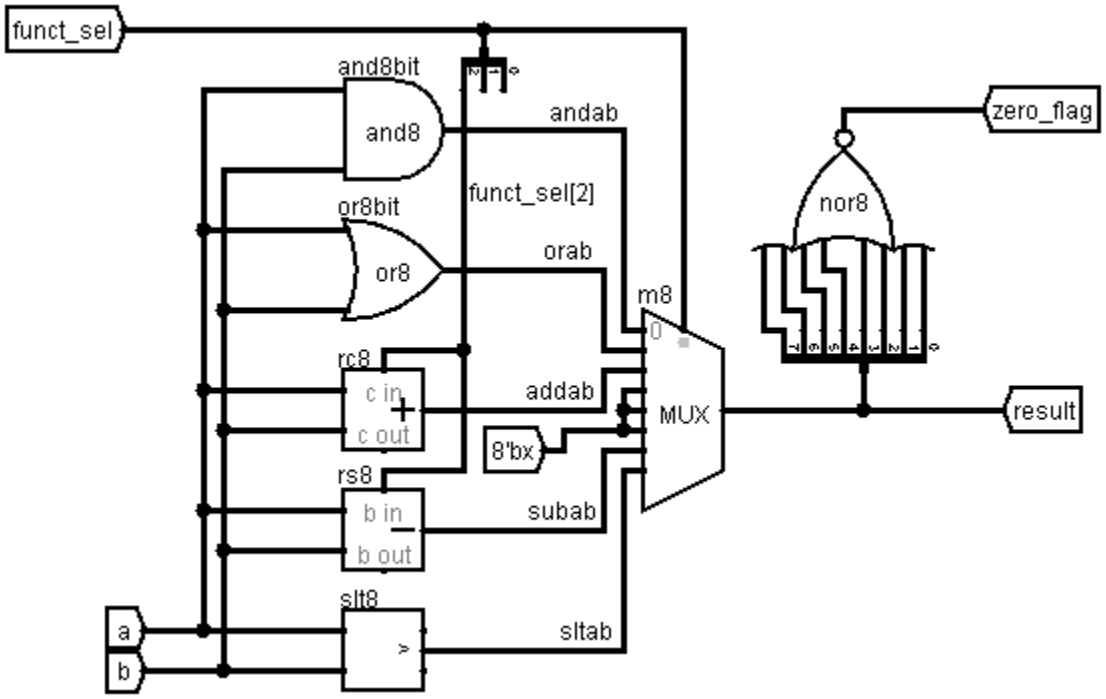
```
module slt8(input [7:0] a,b,
            output reg [7:0] y);
  always @(*)
    begin
      if(a < b) y = 8'b0000_0001;
      else y = 8'b0000_0000;
    end
endmodule
```

Create **mux8to1_8bit.v** given the Verilog source code below:

```
module mux8to1_8bit(input [7:0] d0,d1,d2,d3,d4,d5,d6,d7,
                    input [2:0] s,
                    output reg [7:0] y);
  always @(*) begin
    case(s) //begin
      3'b000:   y=d0;
      3'b001:   y=d1;
      3'b010:   y=d2;
      3'b011:   y=d3;
      3'b100:   y=d4;
      3'b101:   y=d5;
      3'b110:   y=d6;
      3'b111:   y=d7;
      default:  y=8'bx; //invalid output by default
    endcase
  end
endmodule
```

CECS 225: 8-BIT ARITHMETIC LOGIC UNIT

Create the Arithmetic Logic Unit! Block diagram below:



Name your Verilog source file as **alu8.v**, a skeleton is shown below:

```
`include "mux8to1_8bit.v"
`include "slt8.v"
`include "or8.v"
`include "and8.v"
`include "ripplesubtracter8.v"
`include "ripplecarry8.v"

module alu8(input [2:0] funct_sel,
            input [7:0] a,b,
            output zero_flag,
            output [7:0] result);
  wire [7:0] andab, orab, addab, subab, sltab;

  and8 and8bit(a,b,     );
  or8 or8bit(a, ,orab);
  RippleCarry8 rc8(            ,a, b, addab,);
  RippleSubtracter8 rs8(funct_sel[2], ,  , subab,);
  slt8 slt8bit( , ,sltab);
  mux8to1_8bit m8(andab,    ,     ,8'bx,8'bx,8'bx,     ,sltab,
                  funct_sel,result);

  nor nor8(zero_flag, result[0], result[ ],          , result[3],
           result[4], result[ ], result[6],          );
endmodule
```

CECS 225: 8-BIT ARITHMETIC LOGIC UNIT

Leave the **design.sv** file empty as we will not be using it in this project.

Source code for testbench.sv is provided below:

SV/Verilog Testbe

```
1  `include "alu8.v"
2  module alutester;
3    reg [7:0] a,b;  reg [2:0] funct_sel;
4    wire [7:0] result;  wire zero;
5    integer i;
6
7    alu8 uut(funct_sel,a,b,zero,result);
8
9    initial begin
10     b=8'b00001111; a=8'b01011010;
11     for(i = 0; i < 8; i = i + 1) begin
12       funct_sel=i; test_case;
13     end
14     a=8'b00001111; b=8'b01011010; i = i - 1;
15     test_case;  //one more test case!
16     $finish;
17   end
18
19   task test_case;
20     begin
21     #1;
22     case(i)
23       3'b000:   $display("\nTesting the AND function");
24       3'b001:   $display("\nTesting the OR function");
25       3'b010:   $display("\nTesting Addition");
26       3'b110:   $display("\nTesting Subtraction");
27       3'b111:   $display("\nTesting Set Less Than Function");
28       default:  $display("\nInvalid Function Select Code!");
29     endcase
30     if(i == 2) i = 5;
31       $display("a = %b, b = %b, result = %b, zero=%b",a,b,result,zero);
32     end
33   endtask
34 endmodule
```

CECS 225: 8-BIT ARITHMETIC LOGIC UNIT

If everything works correctly then the following console output will be produced:

```
Testing the AND function
a = 01011010, b = 00001111, result = 00001010, zero=0

Testing the OR function
a = 01011010, b = 00001111, result = 01011111, zero=0

Testing Addition
a = 01011010, b = 00001111, result = 01101001, zero=0

Testing Subtraction
a = 01011010, b = 00001111, result = 01001011, zero=0

Testing Set Less Than Function
a = 01011010, b = 00001111, result = 00000000, zero=1

Testing Set Less Than Function
a = 00001111, b = 01011010, result = 00000001, zero=0
```

CECS 225: 8-BIT ARITHMETIC LOGIC UNIT

If your module did not work correctly get assistance from the instructor or a reputable student to troubleshoot your design.

WHAT TO TURN IN: Once the modules in this lab are working correctly:

- Copy the contents of your **alu8.v** module to a file named **alu8.txt**
- upload **alu8.txt** to the beachboard dropbox for **alu8**

NOTE:

Keep the source files from this lab as they will be used in future Labs!