```
1   //Aaron Hong (ahong02)
2   //Stephen Macris (smacris)
3   //3/29/23
4   //EE469 Lab1
5
6   //This module creates a basic arithmetic logic unit capable of addition, subtraction,
7   //ANDing, and ORing.
8
9   //Inputs: Two 32-bits a, b (inputs to be processed), one 2-bit ALUControl (controls which
    operation).
10  //Outputs: One 32-bit Result (result of chosen operation), one 4-bit ALUFlags (flags thrown
    depending on the result).
11  module alu (a, b, ALUControl, Result, ALUFlags);
12
13     input  logic  [31:0] a, b;
14     input  logic [1:0] ALUControl;
15     output logic [31:0] Result;
16     output logic [3:0] ALUFlags;
17     logic cout, x, diff_sign;
18     logic [31:0] sum, b_mod;
19
20     assign x = ~ALUControl[0] & ~ALUControl[1] & (a[31] == b[31]);
21     assign diff_sign = a[31] ^ sum[31];
22
23     //Instantiates a 32-bit full adder to do addition and subtraction operations.
24     fulladder32 FA(.A(a), .B(b_mod), .cin(ALUControl[0]), .sum(sum), .cout(cout));
25
26     //Determines what operation to perform depending on ALUControl.
27     always_comb begin
28        case (ALUControl[0])
29           1'b0: b_mod = b;
30           1'b1: b_mod = ~b;
31        endcase;
32
33        case (ALUControl)
34           2'b00: Result = sum;
35           2'b01: Result = sum;
36           2'b10: Result = a & b;
37           2'b11: Result = a | b;
38        endcase
39
40        ALUFlags[3] = Result[31];
41        ALUFlags[2] = (Result == 32'b0);
42        ALUFlags[1] = ~ALUControl[1] & cout;
43        ALUFlags[0] = x & diff_sign & ~ALUControl[1];
44     end
45
46  endmodule
```