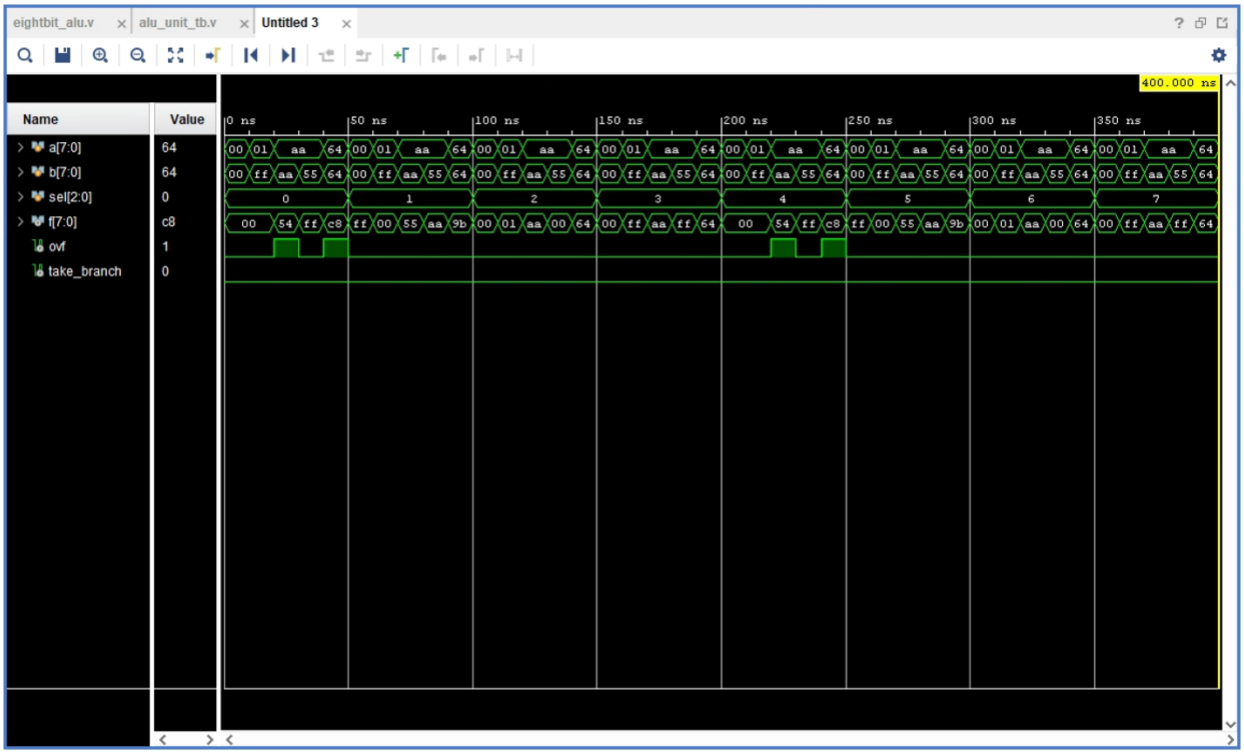
Alex Oswald
EECE 2323
Due: 22 Sep 2020

Pre-Lab Assignment 3

Simulation Screenshot:



Verilog Code for Complete ALU:

```
1.  module eightbit_alu(
2.      input signed[7:0] a,
3.      input signed[7:0] b,
4.      input [1:0] sel,
5.      output reg signed [7:0] f,
6.      output reg ovf,
7.      output reg take_branch);
8.
9.      always@(a, b, sel)
10.     begin
11.         case(sel)
12.             3'b000:
13.             begin
14.                 f = a+b;  take_branch = 0;
15.              end
16.             3'b001:
17.                 begin
18.                     f = ~b;  take_branch = 0;
19.                  end
20.             3'b010:
```

```
                begin
                    f = a&b;  take_branch = 0;
                end
            3'b011:
                begin
                    f = a|b;  take_branch = 0;
                end
            3'b100:
                begin
                    f = a>>>1;  take_branch = 0;
                end
            3'b101:
                begin
                    f = a<<1;  take_branch = 0;
                end
            3'b110:
                begin
                    f = 0;  take_branch = (a==b);
                end
            3'b111:
                begin
                    f = 0;  take_branch = (a!=b);
                end
            default:
                begin
                    f=0;  ovf=0;  take_branch=0;
                end
        endcase
        ovf = (sel==0) && ((a[7] && b[7] && ~f[7]) || (~a[7] && ~b[7] && f[7]));
    end

endmodule
```

Verilog Code for Complete ALU Testbench:

```
`timescale 1ns / 10ps

module alu_unit_tb ();
    // Inputs
    reg signed [7:0] a;
    reg signed [7:0] b;
    reg [2:0] sel;
    // Outputs
    wire signed [7:0] f;
    wire ovf;
    wire take_branch;

    // Initiate the Unit Under Test (UUT)
    eightbit_alu uut (
        .a(a),
        .b(b),
        .sel(sel),
        .f(f),
        .ovf(ovf),
        .take_branch(take_branch));

    // Initialize Inputs (stimulus)
    initial
    begin
```

```
        #0
        sel = 3'b000;
        repeat (8)
        begin
            a = 8'b00000000;  b = 8'b00000000;  #10; // 0   0
            a = 8'b00000001;  b = 8'b11111111;  #10; // 1   -1
            a = 8'b10101010;  b = 8'b10101010;  #10; // -86 -86
            a = 8'b10101010;  b = 8'b01010101;  #10; // -86 85
            a = 8'b01100100;  b = 8'b01100100;  #10; // 100 100
            sel = sel + 1;
        end
    end

endmodule
```