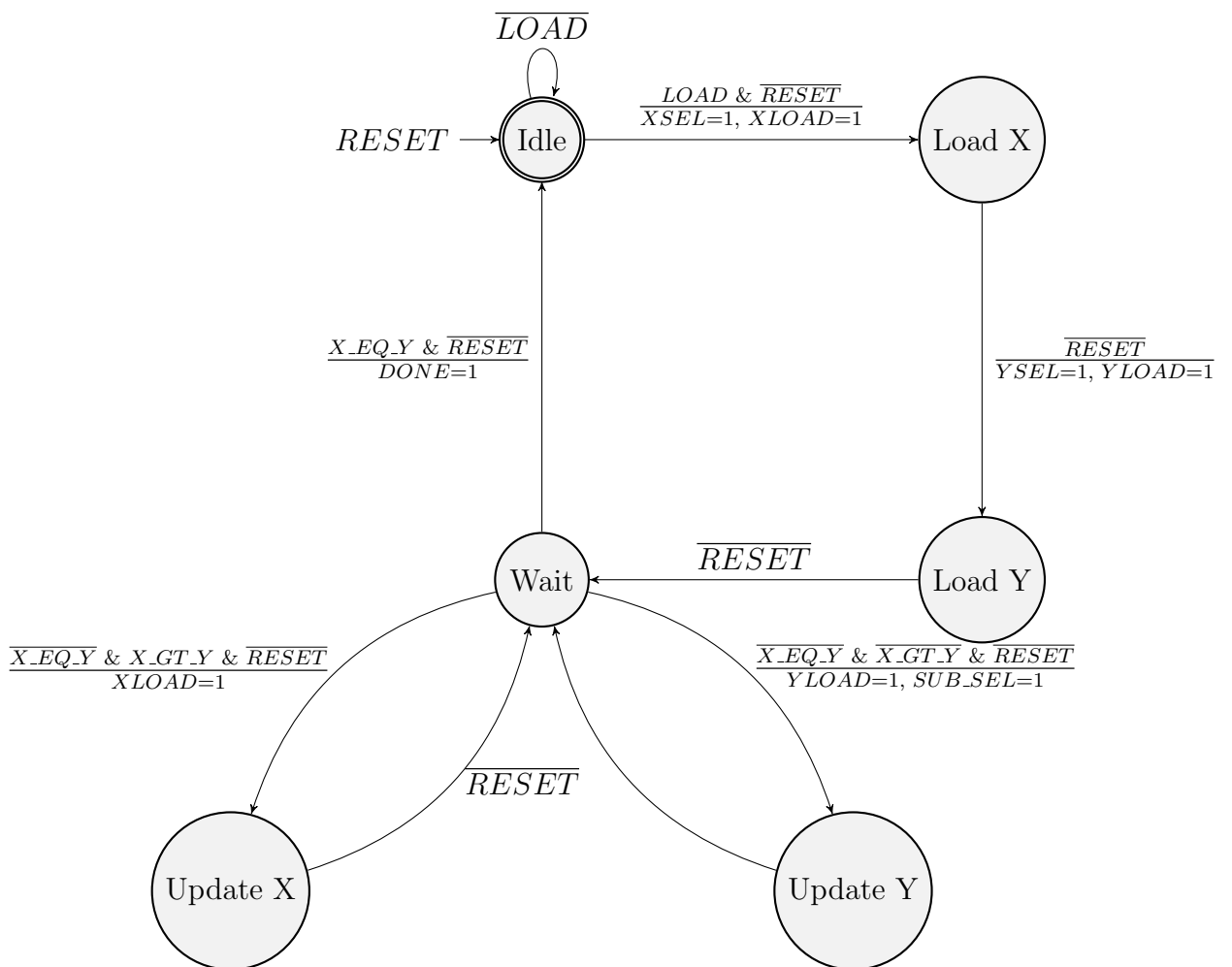


ECE 440 - Homework #2

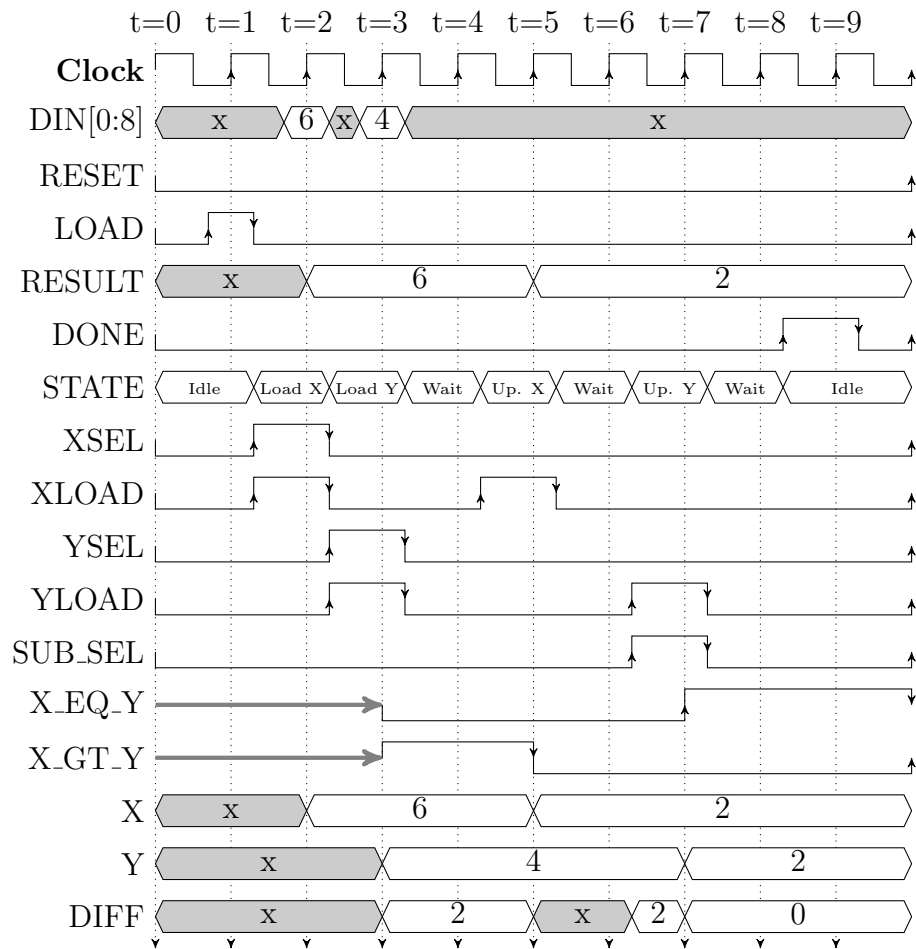
Collin Heist

7th February 2020

1 Problem 1



2 Problem 2



*Note: The temporary unknown value of the **DIFF** line is due to this being affected by the ALU being used. For that small period, $2 - 4$ is being computed - resulting in an n -bit 2's complement number.*

3 Problem 3

Listing 1: Waveform Generation

```

initial begin
    a = 0; b = 1; #3;
    a = 1; b = 0; #3;
    b = 1; #2;
    a = 0;      #3;
    a = 1; b = 0; #1;
    a = 0;      #4;
end

```

4 Problem 4

Scalar outputs N, Z, and V indicate when the alu's output is negative (2s complement), zero, or there is a twos complement overflow.

Listing 2: ALU Module

```
module alu
  #(parameter width = 8)
  (input logic carry_in ,
   input logic [1:0] fn ,
   input logic [width-1:0] a, b,
   output logic carry_out , N, Z, V,
   output logic [width-1:0] result );

  // Make enumeration of logic operations
  typedef enum logic [1:0] {sADD, sAND, sOR, sXOR} alu_ops;

  // Determine if there is a carry-out
  logic [width:0] temp = {1'b0, a} + {1'b0, b};
  assign carry_out = temp[width];

  always_comb
  begin
    assign N = 0; assign Z = 0; assign V = 0;
    unique case(fn)
      sADD:
        begin
          assign result = a + b + carry_in;
          assign N = result[width-1];
          assign Z = (result == '0);
          assign V = (!a[width-1] & !b[width-1] &
            result[width-1]) | (a[width-1] & b[width-1] &
            !result[width-1]);
        end
      sAND: assign result = a & b;
      sOR:  assign result = a | b;
      sXOR: assign result = a ^ b;
    endcase
  end
endmodule;
```

Listing 3: ALU Testbench

```
'timescale 1ns / 1ps
module testbench;
  parameter width = 4;
```

```

logic carry_in , carry_out , N, Z, V;
logic [1:0] fn;
logic [width-1:0] a, b, result;

alu dut(.*);

initial begin
    carry_in = 0;
    fn = 2'b0;
    a = width'b0; b = width'b0;
    #100; // Wait for global reset

    // Add (7) and (-7)
    fn = 2'b00;
    a = width'b0111; b = width'b1001; #3;
    if ((result == 'b0000) && (N == '0) &&
        (Z == '1) && (V == '0) && (carry_out == '0))
        $display("ADD_test_passed.");

    fn = 2'b01;
    a = width'b1110; b = width'b1100;
    if ((result == 'b1100) && (N == '0) &&
        (Z == '0) && (V == '0) && (carry_out == '0))
        $display("AND_test_passed.");

    fn = 2'b10;
    a = width'b0101; b = width'b1000;
    if ((result == 'b1101) && (N == '0) &&
        (Z == '0) && (V == '0) && (carry_out == '0))
        $display("OR_test_passed.");

    fn = 2'b11;
    a = width'b1101; b = width'b1011;
    if ((result == 'b0110) && (N == '0) &&
        (Z == '0) && (V == '0) && (carry_out == '0))
        $display("XOR_test_passed.");
end
endmodule

```