Prelab

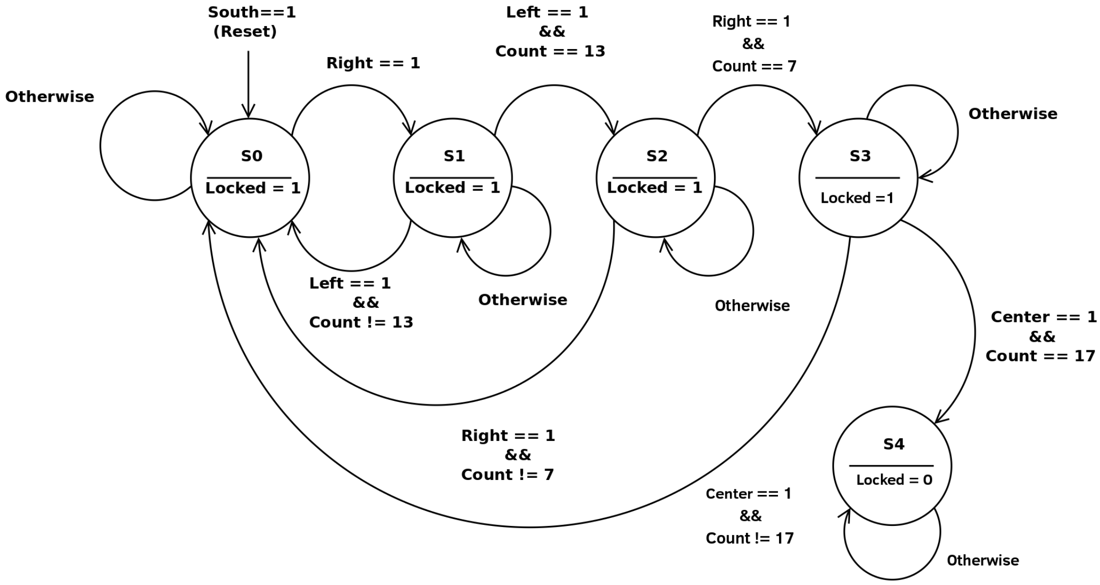

Figure 1: *State Diagram for Combination Lock*

Code Block 1: 4-Bit ALU

```
module combination_lock_fsm (
  output reg [2:0] state,
  output wire Locked, // asserted when locked
  input wire Right, Left, // indicate direction
  input wire [4:0] Count, // indicate position
  input wire Center, // the unlock button
  input wire Clk, South // clock and reset
  );

  parameter S0 = 2'b00,
            S1 = 2'b01,
            S2 = 2'b10,
            S3 = 2'b11;

  reg [1:0] state;
reg [1:0] nextState;


always @ ( * ) begin
  case (state)
    S0: begin
      if (Right)
        nextState = S1;
      else
        nextState = S0;
      end
    S1: begin
      if (Left && Count == 13)
        nextState = S2;
      else if (Left && Count != 13)
        nextState = S1;
      end
    S2: begin
      if (Right == 1 && Count == 7)
        nextState = S3;
      else if (Right == 1 && Count != 7)
        nextState = S2
      end
    S3: begin
      if (Center == 1 && Count == 17)
        nextState = S4;
      else if (Center == 1 && Count != 17)
        nextState = S3
      end
    // S4: begin
    //     if (condition)
```

```
      //
      //     else
      // end
    endcase
  end

  always @ (posedge Clk) begin
    if (Rst)
      state <= S0;
    else
      state <= nextState;
  end

  if (state == S4)
    assign Locked = 0;
  else
    assign Locked = 1;
endmodule // combination_lock_fsm
```