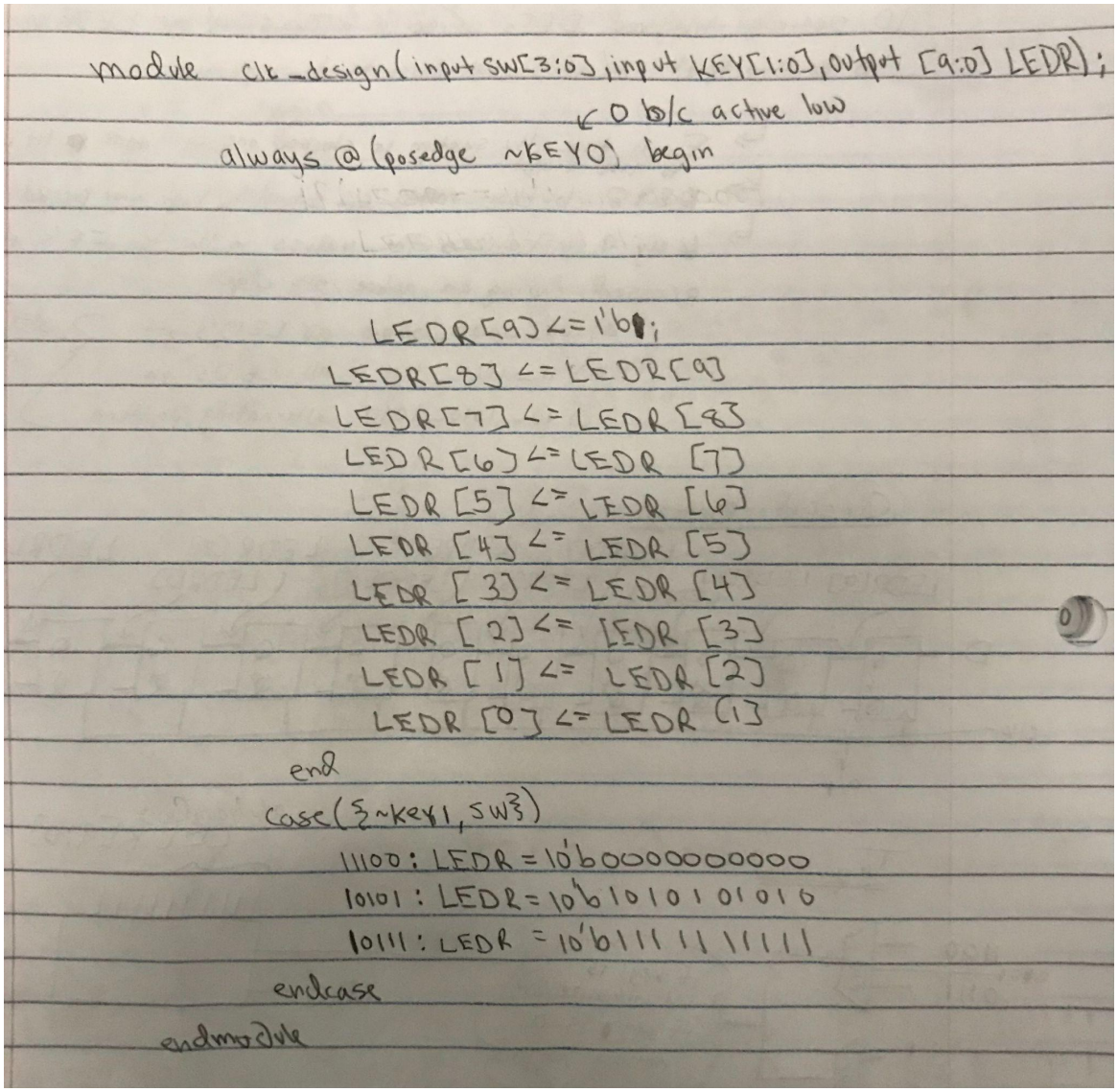
Alejandro Torres
915207383

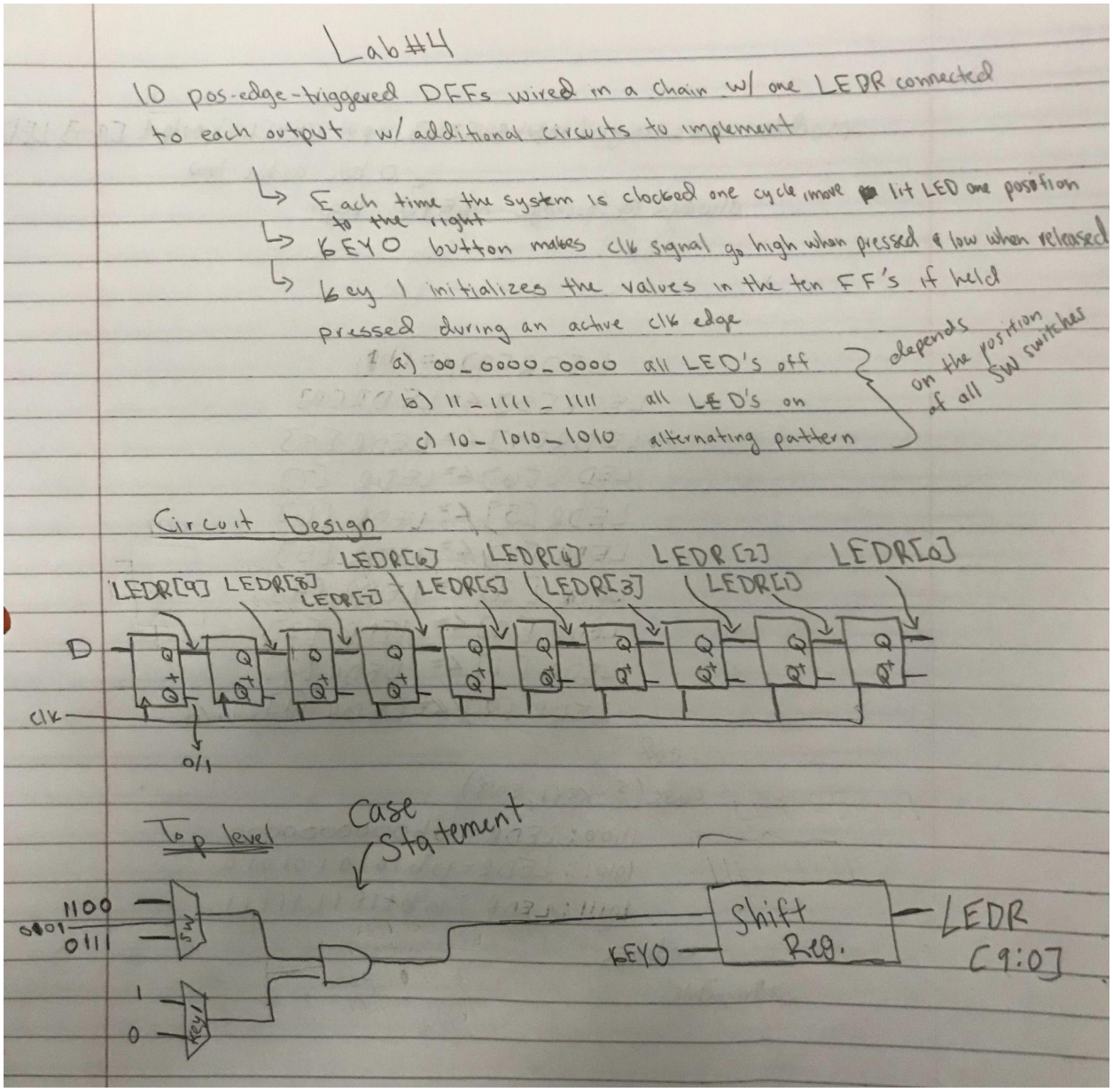
Lab 4 : A Flip-Flop-Based Shift Register

Pre-lab:

- I had an issue with my pre-lab design. I set LEDR[9] = 1, so it would always display the led on the board. I added an if condition in the verilog code to only turn on when all LEDR's were off. In the sensitivity list we add ~KEY0 because it imitates the clock signal and it's inverted because the key buttons are active low. We always make sure to include non-blocking assignments when dealing with a clock signal because we want all statements to happen in parallel not sequentially. Lastly, I used 4 switches as inputs to initialize the LEDR's.



Circuit Design:



Verilog code:

- I decided on using a 4-bit input from the switches to handle the initializations. The case will be executed when the clock signal (KEY0) is activated. If the KEY[1] value and the desired SW inputs match then the LEDR's will be initialized accordingly.

```
`timescale 1ps/1ps
module lab4(
    input [1:0] KEY,
    input [3:0] SW,
    output reg [9:0] LEDR
);

always @(posedge ~KEY[0]) begin
    if (LEDR == 10'b0000000000) begin
        LEDR[9] <= #1 1'b1;
    end
    else begin
        LEDR[9] <= #1 1'b0;
    end

    LEDR[8] <= #1 LEDR[9];
    LEDR[7] <= #1 LEDR[8];
    LEDR[6] <= #1 LEDR[7];
    LEDR[5] <= #1 LEDR[6];
    LEDR[4] <= #1 LEDR[5];
    LEDR[3] <= #1 LEDR[4];
    LEDR[2] <= #1 LEDR[3];
    LEDR[1] <= #1 LEDR[2];
    LEDR[0] <= #1 LEDR[1];

    case({~KEY[1], SW})
        // SW[3:0] = 1100 turns off LED's
        5'b11100: LEDR <= #1 10'b0000000000;
        // SW[3:0] = 0101 alternate LED's
        5'b10101: LEDR <= #1 10'b1010101010;
        // SW[3:0] = 0111 turns on LED's
        5'b10111: LEDR <= #1 10'b1111111111;
    endcase
end


endmodule
```

Testbench code:

- I initialize each of the 3 cases. For that I initialized the SW inputs and KEY values to 1 to activate a clock cycle. I then run the next 15 cycles for each case in the repeat blocks. Inside each block the clock signal is 0 and after a #100 delay the clock signal is 1 and the LEDR get's updated. I then repeat a case with 25 clock cycles.

```
1   `timescale 1ps/1ps
2   module lab4_tb;
3   reg [1:0] KEY;
4   reg [3:0] SW;
5   wire [9:0] LEDR;
6
7   // Instanstiate shift register
8   lab4 test(KEY, SW, LEDR);
9
10  initial begin
11      // All LED's off
12      SW = 4'b1100;
13      KEY = 2'b00;
14      #100
15      KEY = 2'b11;
16      #100
17      $display("---Turn off all LED's---");
18      $display("sw = %b, output = %b", SW, LEDR);
19      $display("---Next 15 cycles---");
20      repeat(15) begin
21              KEY[0] = 0;
22              #100
23              KEY[0] = 1;
24              #100
25              $display("output = %b", LEDR);
26      end
27
28      // Alternating LED's
29      SW = 4'b0101;
30      KEY = 2'b00;
31      #100
32      KEY = 2'b11;
33      #100
34      $display("---Alternate LED's---");
35      $display("sw = %b, output = %b", SW, LEDR);
36      $display("--- Next 15 cycles---");
37      repeat(15) begin
38              KEY[0] = 0;
```

```
39          #100
40          KEY[0] = 1;
41          #100
42          $display("output = %b", LEDR);
43      end
44
45      // All LED's on
46      SW = 4'b0111;
47      KEY = 2'b00;
48      #100
49      KEY = 2'b11;
50      #100
51      $display("---Turn on all LED's---");
52      $display("sw = %b, output = %b", SW,LEDR);
53      $display("---Next 15 cycles---");
54      repeat(15) begin
55          KEY[0] = 0;
56          #100
57          KEY[0] = 1;
58          #100
59          $display("output = %b", LEDR);
60      end
61
62      // All LED's on
63      SW = 4'b0111;
64      KEY = 2'b00;
65      #100
66      KEY = 2'b11;
67      #100
68      $display("---Turn on all LED's---");
69      $display("sw = %b, output = %b", SW, LEDR);
70      $display("---Next 25 cycles---");
71      repeat(25) begin
72          KEY[0] = 0;
73          #100
74          KEY[0] = 1;
75          #100
76          $display("output = %b", LEDR);
77      end
78  end
79
80  endmodule
81
```

Transcript printout:

- The results were expected because you can see the LEDR in the printout shift by one after each clock cycle for all cases.

```
# ---Turn off all LED's---
# sw = 1100, output = 0000000000
# ---Next 15 cycles---
# output = 1000000000
# output = 0100000000
# output = 0010000000
# output = 0001000000
# output = 0000100000
# output = 0000010000
# output = 0000001000
# output = 0000000100
# output = 0000000010
# output = 0000000001
# output = 0000000000
# output = 1000000000
# output = 0100000000
# output = 0010000000
# output = 0001000000
```

```
# ---Alternate LED's---
# sw = 0101, output = 1010101010
# --- Next 15 cycles---
# output = 0101010101
# output = 0010101010
# output = 0001010101
# output = 0000101010
# output = 0000010101
# output = 0000001010
# output = 0000000101
# output = 0000000010
# output = 0000000001
# output = 0000000000
# output = 1000000000
# output = 0100000000
# output = 0010000000
# output = 0001000000
# output = 0000100000
```

```
# ---Turn on all LED's---
# sw = 0111, output = 1111111111
# ---Next 15 cycles---
# output = 0111111111
# output = 0011111111
# output = 0001111111
# output = 0000111111
# output = 0000011111
# output = 0000001111
# output = 0000000111
# output = 0000000011
# output = 0000000001
# output = 0000000000
# output = 1000000000
# output = 0100000000
# output = 0010000000
# output = 0001000000
# output = 0000100000
```

```
# ---Turn on all LED's---
# sw = 0111, output = 1111111111
# ---Next 25 cycles---
# output = 0111111111
# output = 0011111111
# output = 0001111111
# output = 0000111111
# output = 0000011111
# output = 0000001111
# output = 0000000111
# output = 0000000011
# output = 0000000001
# output = 0000000000
# output = 1000000000
# output = 0100000000
# output = 0010000000
# output = 0001000000
# output = 0000100000
# output = 0000010000
# output = 0000001000
# output = 0000000100
# output = 0000000010
# output = 0000000001
# output = 0000000000
# output = 1000000000
# output = 0100000000
# output = 0010000000
# output = 0001000000
```