SPRING 2024

EE330 HOMEWORK 7 SOLUTIONS

Q1. (A and B) Assume Transistor is in Saturation

$$I_D = \frac{\mu c_{OX} W}{2L}(V_{GS} - V_{TH})^2$$
$$I_D = \frac{100(10^{-6})(10)}{2(2)}(3 - 0.75)^2 = 1.266mA$$

$$V_{out} = V_{DD} - I_D R$$
$$V_{out} = 5 - 1.266m(1K)$$
$$V_{out} = 3.734V$$

(C) Assume Transistor is in saturation

Then the drain current $I_D = 1.266mA$

And

$$V_{out} = V_{DD} - I_D R$$
$$V_{out} = 5 - 1.266m(5K)$$
$$V_{out} = -1.33V$$

For the transistor to be in saturation, $V_{out} \geq V_{GS} - V_{TN}$, however this condition is not satisfied. Thus indicating the transistor is not in saturation but in the linear region.

Transistor in the Linear Region

$$I_D = \frac{\mu c_{OX} W}{2L}(V_{GS} - V_{TH} - V_{out})V_{out}$$
$$I_D = \frac{100(10^{-6})(10)}{2(2)}(3 - 0.75 - V_{out})V_{out}$$

Also,

$$I_D = \frac{5 - V_{out}}{5k}$$

$$\frac{100(10^{-6})(10)}{2(2)}(3 - 0.75 - V_{out})V_{out} = \frac{5 - V_{out}}{5k}$$
$$V_{out} = 0.225 \quad or \quad V_{out} = 2.825$$

$V_{out} = 2.825$V satisfies saturation region conditions.

$$I_D = \frac{5 - V_{out}}{5k}$$

$$I_D = \frac{5 - 0.2825}{5k} = 0.435mA$$

Q2. Assume transistor is in saturation

$$V_{out} = V_{GS} = 6V$$

$$I_D = \frac{5 - V_{out}}{5k} = \frac{3}{5k} = 0.6mA$$

$$I_D = \frac{\mu c_{OX} w}{2L}(V_{GS} - V_{TH})^2$$

$$0.6mA = \frac{100(10^{-6})(W)}{2(2)}(3 - 0.75)^2$$

$$W = 10.67\mu m$$

Q3.

$$V_{GS} = V_G - V_S = 0 - (-2) = 2V$$

For Transistor in Saturation

$$I_D = \frac{\mu c_{OX} w}{2L}(V_{GS} - V_{TH})^2$$

$$I_D = \frac{100(10^{-6})(12)}{2(2)}(2 - 0.75)^2 = 468.75\mu A$$

Also, for transistor to be in saturation, $V_{DS} \geq V_{GS} - V_{TN}$

$$V_{DS} = V_D - V_S$$

$$V_{DS} = V_{OUT} - (-2) = V_{out} + 2$$

$$V_{DS} = V_{out} + 2 = (4 - I_D R_1) + 2$$

$$V_{DS} = 6 - I_D R_1$$

But $V_{GS} - V_{TN} = 2 - 0.75 = 1.25V$

$$V_{DS} \geq V_{GS} - V_{TN}$$

$$6 - I_D R_1 \geq 1.25$$

$$R_1 \leq \frac{6 - 1.25}{I_D}$$

$$R_1 \leq \frac{4.75}{468.75\mu A}$$

$$R_1 \leq 10.13k\Omega$$

Q4.

$$V_{DD} = 3V \quad V_{SS} = -2V \quad W_1 = 8\mu \quad L_1 = 2\mu \quad W_2 = 50\mu \quad L_2 = 2\mu$$
$$\mu_n C_{ox} = 100\mu A/V^2 \quad \mu_p C_{ox} = 30\mu A/V^2$$

$$I_{D1} = I_{D2}$$

$$\frac{\mu_n c_{OX} W_1}{2L_1}(V_{GS1} - V_{TN})^2 = -\frac{\mu_p c_{OX} W_2}{2L_2}\left(V_{GS2} - V_{Tp}\right)^2$$

Lets assume Vin = 0V

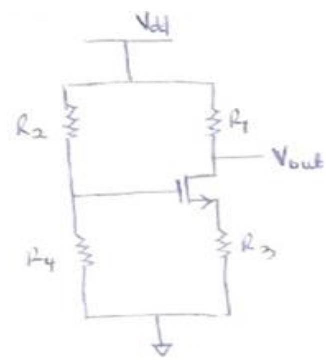
$$\frac{100(8)}{2(2)}(2 - 0.75)^2 = -\frac{30(50)}{2(2)}(V_{out} - 3 - 0.75)^2$$

$$\frac{8}{15}(1.25)^2 = -(V_{out} - 3.75)^2$$

$$0.9129 = -V_{out} + 3.75$$

$$V_{out} = 2.84V$$

Q5.



From the process parameters

$V_{th} = 0.76$, $\frac{W}{L} = \frac{3}{0.6}$

$\frac{\mu_n C_{ox}}{2} = 57.8 \times 10^{-6} A/V^2$

$\rightarrow$ $V_G = V_{dd} * \frac{90k}{(90+10)k} = \frac{90}{100} \times 10 = 9v$

=> Assuming operation in saturation,

$I_D = \frac{\mu_n C_{ox}}{2} \frac{W}{L} (V_{GS} - V_T)^2 = 57.8 \times 10^{-6} \times \frac{3}{0.6} (9 - V_s - 0.76)^2$

$I_D = 2.89 \times 10^{-4} (8.24 - V_s)^2$

Since $I_D = I_s = \frac{V_s}{R_3} = \frac{V_s}{2 \times 10^3} = 2.89 \times 10^{-4} (8.24 - V_s)^2$

$$\Rightarrow V_S = 0.578(8.24 - V_S)^2 = 0.578\left(67.898 - 16.48 V_S + V_S^2\right)$$

$$V_S = 39.25 - 9.53 V_S + 0.578 V_S^2$$

$$0 = 39.25 - 10.53 V_S + 0.578 V_S^2$$

Using quadratic formula, $\dfrac{-b \pm \sqrt{b^2 - 4ac}}{2a} = V_S$

$$V_S = \begin{cases} 12.94 \\ 5.23 \end{cases}$$

→ The ~~correct~~ answer cannot be 12.94, since V_{dd} is only 10_V

$\Rightarrow$ For $V_S = 5.23V$, $I_D = 2.59 \times 10^{-4}(8.24 - 5.23)^2 = \underline{2.62 mA}$

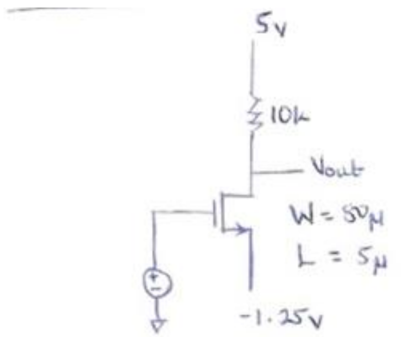
$\Rightarrow V_{out} = R_4 \times I_D = 5000 \times 2.62 \times 10^{-3} = \underline{13.1V}$

$V_{DS} = 13.1 - 5.23 = 7.87_V$

$V_{GS} = V_G - V_S = 9 - 5.23 = 3.77_V$

Therefore, $V_{DS} > V_{GS} - V_T$ and the circuit is in saturation

Q6.



(a)

$$V_{gs} = V_{in} - V_s = 0 - (-1.25) = 1.25_V$$

$$V_{DS} = V_D - V_S = V_D - (-1.25) = V_D + 1.25$$

Since $V_{DS} > V_{gs} - V_T$, the circuit is operating in saturation

$$\Rightarrow I_D = \frac{\mu_n C_{ox}}{2} \frac{W}{L} (V_{gs} - V_T)^2 = \frac{100 \times 10^{-6}}{2} \frac{80}{5} (1.25 - 0.75)^2 = 200 \times 10^{-6} A$$

• $V_{out} = V_{DD} - (I_D \times 10k) = 5 - (200 \times 10^{-6} \times 10 \times 10^{3})$

$$V_{out} = 5 - 2 = 3_V$$

(b) If $V_{IN} = 0.1V$, $V_{gs} = 0.1 + 1.25 = 1.35$

$\rightarrow I_D = \frac{100 \times 10^{-6}}{2} \frac{80}{5} (1.35 - 0.75)^2 = 288 \times 10^{-6} A$

$\rightarrow V_{out} = 288 \times 10^{-6} \times 10k = \underline{2.88V}$

$\Rightarrow$ Thus, the output waveform will be going between 2V and 2.88V

(c) $\text{Gain} = \frac{2.88}{0.1} = \underline{28.8}$

Q7.

Consider the C_{ox} given at the top of the assignment $C_{ox} = 4\ fF/\mu m^2$

$C_P = C_{ox} * W_p * L_P = 4 * 20 * 2 = 160\ fF$

$C_n = C_{ox} * W_n * L_n = 4 * 5 * 1 = 20\ fF$

$C_{total} = 160 + 20 = 180\ fF$

$$R_{LH} = R_P = \frac{1}{\mu_{cox_p}\left(\frac{W_p}{L_p}\right)(V_{GS} - V_t)} = \frac{1}{75 * 10^{-6} * \left(\frac{20}{2}\right)(1.5 - 0.5)} = 1.33K\Omega$$

$$R_{HL} = R_N = \frac{1}{\mu_{cox_n}\left(\frac{W_n}{L_n}\right)(V_{GS} - V_t)} = \frac{1}{350 * 10^{-6} * \left(\frac{5}{1}\right)(1.5 - 0.5)} = 571.4K\Omega$$

Q8.

(a)

$$V_{out} = 10 - I_c(2k)$$

$$V_{out} = 10 - (100x20\mu A)(2k)$$

$$V_{out} = 6V$$

(b)

$$I_b = \frac{10 - V_{BE}}{300K} = \frac{10 - 0.6}{300K} = 31.33\mu A$$

$$I_C = 100\ I_b = 3.133mA$$

$$V_{out} = 10 - I_c(2k)$$

$$V_{out} = 10 - (100x31.33\mu A)(2k)$$

$$V_{out} = 3.734V$$

(c)

$$I_b = \frac{V_{out} - V_{BE}}{200K} = \frac{V_{out} - 0.6}{200K}$$

$$I_C = 100\ I_b = \frac{V_{out} - 0.6}{200K} x100$$

$$V_{out} = 10 - I_c(2k)$$

$$V_{out} = 10 - \frac{V_{out} - 0.6}{200K} x100(2k)$$

$$V_{out} = 5.3V$$

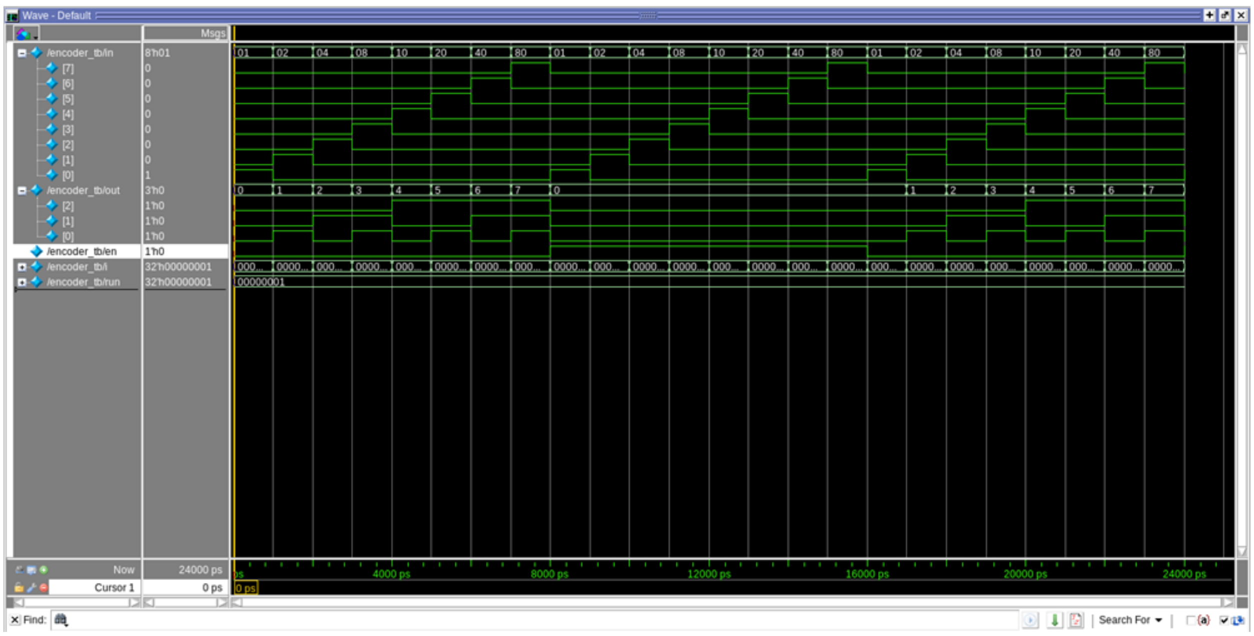
Q9 & Q10

```
//encoder
`timescale 1ns/1ps                          //nice timescale for simulation

module encoder_8to3(in, out, en);           //instantiate module
input [7:0] in;                             //instantiate input, 8 bits
input en;                                   //enable pin
output reg [2:0] out;                       //instantiate output, 3 bits
always @ (*) begin                          //any time an input changes, execute

    if (en) begin                           //active low enable, when en is high
        out = 0;                            //set output to zero
    end
    else begin                              //if enabled
        case (in)                           //case maps inputs to output
            8'b00000001 : out = 3'b000;     //only anticipates one input at a time
            8'b00000010 : out = 3'b001;
            8'b00000100 : out = 3'b010;
            8'b00001000 : out = 3'b011;
            8'b00010000 : out = 3'b100;
            8'b00100000 : out = 3'b101;
```

```
                8'b01000000 : out = 3'b110;

                8'b10000000 : out = 3'b111;

            endcase
        end
end
endmodule
```

```
//encoder testbench

`timescale 1ns/1ps                              //nice timescale for simulation

module encoder_tb();                            //instantiate testbench

reg [7:0] in;                                   //instantiate input

reg en;                                         //enable pin

wire [2:0] out;                                 //instantiate output

integer i, run;                                 //used within testbench loop

encoder_8to3 DUT(.in(in),.out(out),.en(en));    //instantiate DUT

initial begin                                   //start test

    run = 1;                                    //used to force while loop to run forever

    in = 0;                                     //input initial value

    en = 0;                                     //enable initial value

    i = 0;                                      //iteration variable

    while (run) begin                           //force to loop forever

        if (i > 7) begin                        //if iteration variable exceeds seven

            en = ~en;                           //toggle enable

            i = 0;                              //set iteration variable to zero

            in = 1;                     //set input to one to avoid negative exponent error

            #1;                                 //wait one time unit

            i = 1;                      //set iteration variable to one, enter back into loop

        end
```

```
            in = in + (2 ** i);              //add 2^n to input
            in = in - (2 ** (i-1));          //subtract 2^(n-1) from input, above if avoid 2^-1
            i = i + 1;                       //increment iteration variable
            #1;                              //wait one time unit
      end                                    //above while loop mimics 8 bit shift register which
end                                     //feeds back into itself. Has one active bit, rest are zero
endmodule
```




```
/decoder implementation
`timescale 1ns/1ps                           //nice simulation timescale
module decoder_3to8(in, out, en);            //instantiate module
input [2:0] in;                              //instantiate input, 3 bits
input en;                                    //enable pin
output reg [7:0] out;                        //instantiate output, 7 bits
always @ (*) begin                           //any time an input changes, execute
```

```
        if (en) begin                       //active low enable, when en is high
            out = 0;                        //set output to zero
        end

            else begin                      //if enabled
            case (in)                       //case statement maps input to output
                3'b000 : out = 8'b00000001;
                3'b001 : out = 8'b00000010;
                3'b010 : out = 8'b00000100;
                3'b011 : out = 8'b00001000;
                3'b100 : out = 8'b00010000;
                3'b101 : out = 8'b00100000;
                3'b110 : out = 8'b01000000;
                3'b111 : out = 8'b10000000;
            endcase
        end
end

endmodule

//decoder testbench

`timescale 1ns/1ps                          //nice simulation timescale

module decoder_tb();                        //instantiate testbench

reg [2:0] in;                               //instantiate input, 3 bits

reg en;                                     //enable pin

wire [7:0] out;                             //instantiate output, 8 bits

decoder_3to8 DUT(.in(in),.out(out),.en(en)); //instantiate DUT

        initial in = 0;                     //set initial values
```

```
initial en = 0;

always #1 in[0] = ~in[0];           //toggle at timed intervals

always #2 in[1] = ~in[1];

always #4 in[2] = ~in[2];

always #8 en = ~en;
endmodule
```