MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Department of Electrical Engineering and Computer Sciences

Introductory Digital Systems Lab (6.111)
Quiz - Spring 2006
Prof. Anantha Chandrakasan

Student Name: ____________________________

Problem 1 (20 Points): ______________________

Problem 2 (22 Points): ______________________

Problem 3 (22 Points): ______________________

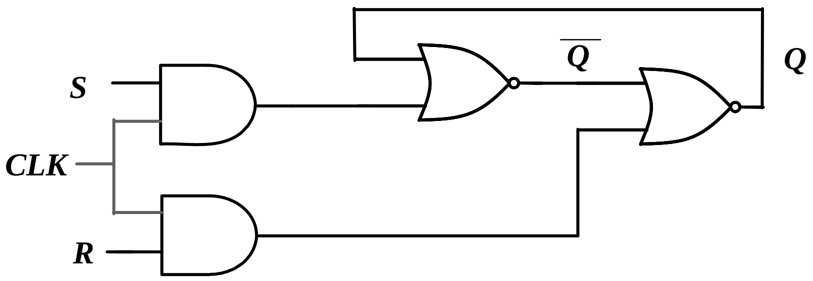
Problem 4 (18 Points): ______________________

Problem 5 (18 Points): ______________________

Total (100 Points): ______________________

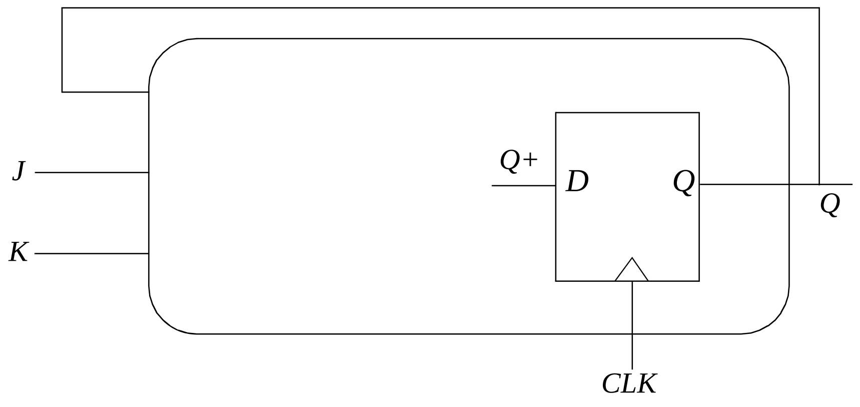
Problem 1: Sequential building blocks and counters

(a) Consider the following sequential building block.
(i) What type of sequential block is it?
(ii) Assume that S = R = 1. What happens to the outputs Q and $\overline{Q}$ when CLK transitions from a 1 to 0? Briefly explain your answer. **(6 points)**

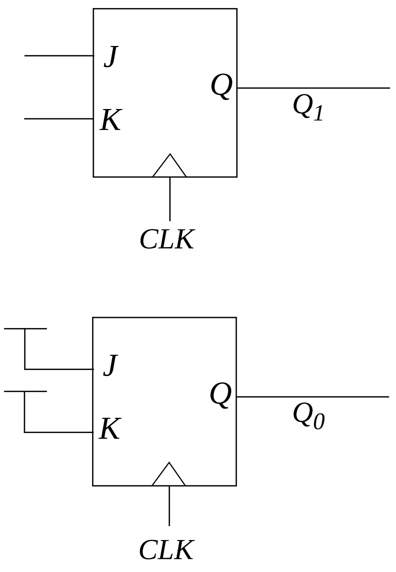


(b) Assume you have access to a ***D positive* edge-triggered register**. Construct a ***J-K positive edge triggered* register** that implements the functionality shown in the truth-table. Q^+ is the next state. You may add logic gates as necessary (show a logic diagram). **(6 points)**

J	K	$Q+$
0	0	Q
0	1	0
1	0	1
1	1	$\overline{Q}$



(c) Construct a 2-bit **synchronous** counter using the *J-K register* above (it should count 00→01→10→11→00...). Assume a black box for the *J-K register* with three inputs (*J*, *K*, *CLK*) and one output *Q*. A partial implementation is shown below. Minimize the number of additional gates required. Show the K-MAP optimization for the MSB.**(8 points)**

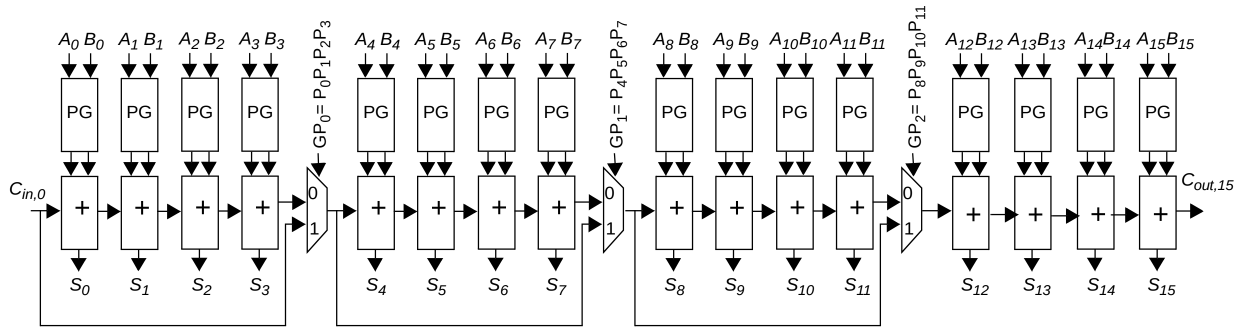


<u>Problem 2: Carry Skip Adder:</u> In this problem, we will explore the addition of two numbers based on the Carry Skip (or Carry Bypass) technique.

(a) Consider the following 16-bit Carry Skip Adder (each + block represents a one full-bit adder). Assume the following delays:

- Delay to produce the P_i, G_i signals from the A_i, B_i inputs is 1 (i.e., the delay of the PG block)
- Delay to compute $C_{out,i}$ from P_i, G_i and $C_{in,i}$ inputs is 1
- Delay to compute S_i from $C_{in,i}$, P_i, G_i being valid is 2
- Delay for the 2:1 multiplexor is 2
- Delay to compute the group propagate, GP_i, once the propagates are available is 1

Highlight the critical path for this 16-bit adder directly on the figure below **(4 points)**



A_0 B_0 A_1 B_1 A_2 B_2 A_3 B_3
$GP_0 = P_0P_1P_2P_3$
A_4 B_4 A_5 B_5 A_6 B_6 A_7 B_7
$GP_1 = P_4P_5P_6P_7$
A_8 B_8 A_9 B_9 A_{10} B_{10} A_{11} B_{11}
$GP_2 = P_8P_9P_{10}P_{11}$
A_{12} B_{12} A_{13} B_{13} A_{14} B_{14} A_{15} B_{15}
PG
+
0
1
$C_{in,0}$
$C_{out,15}$
S_0 S_1 S_2 S_3 S_4 S_5 S_6 S_7 S_8 S_9 S_{10} S_{11} S_{12} S_{13} S_{14} S_{15}

A 16 bit Carry Skip Adder with a block size = 4.

(b) What is the delay for a 32-bit adder assuming eight equal block sizes of 4 and making the same assumptions of delay as part 2 (a)? Assume that the last block does not have a multiplexor as in part 2 (a). **(4 points)**

(c) In the computation of group propagate signal (GP_i) which involves the product of the propagate signals, the propagate for each bit should be computed as:

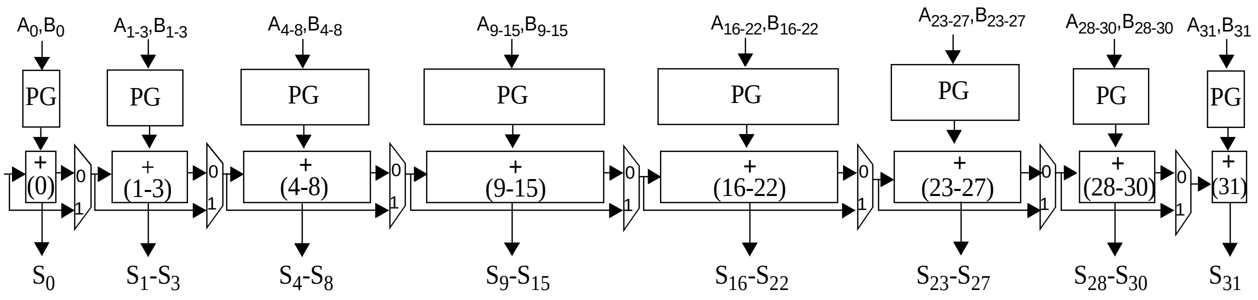
(1) $P = A + B$

(2) $P = A \oplus B$

(3) either $P = A + B$ or $P = A \oplus B$

Clearly explain your answer. **(6 points)**

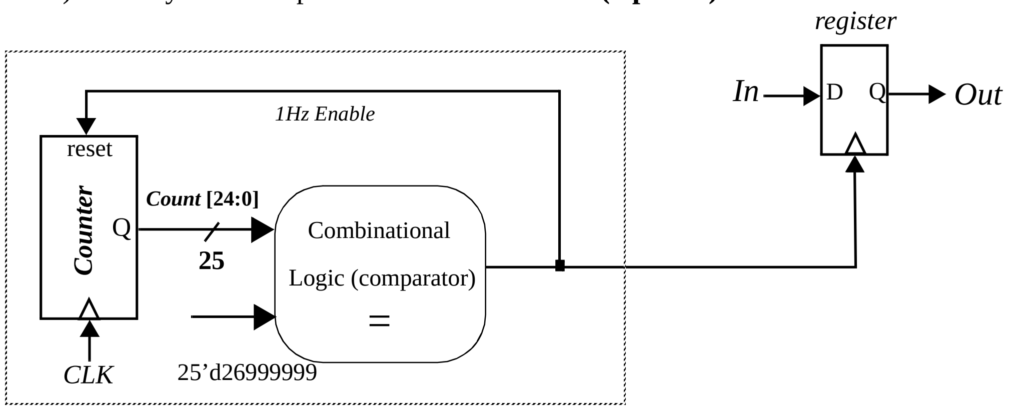
(d) Consider the following 32-bit adder variation where the block sizes of each stage are not equal. Highlight the critical path directly on the figure below. The figure shows a simplified diagram with some obvious details left out. What is the delay of the critical path? Assume the same delays as part (a). **(8 points)**



A 32-bit Carry Skip Adder with a variable block sizes.

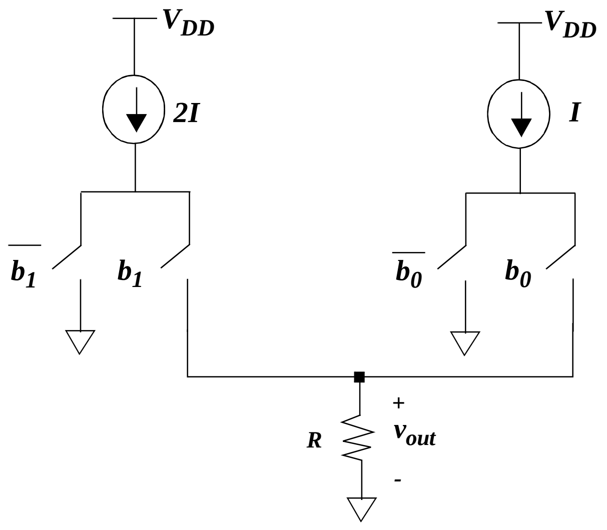
<u>PROBLEM 3: Glitches in Digital Design</u>

Assume for that for the entire problem the counter reset is synchronous.
(a) Consider the one-second clock enable from lab2. The 1Hz enable is derived from the circuit below (the clock is the 27Mhz clock). Assume that a global reset signal puts the count to 0 (the circuit is not shown here). The 1Hz enable is used to clock other circuits that need to be updated once per second (e.g., the register shown below). Identify the main problem with this circuit. **(5 points)**



(b) Propose a solution (draw the logic diagram) that fixes the above problem, but implements the exact same functionality. The solution must be fully **synchronous (i.e., a single clock to every register)**.
(6 points)

(c) As discussed in class, a common architecture used in the design of Digital-to-Analog Converters is using binary weighted current sources. Assume that the two input bits (b_1, b_0) are output from registers but can arrive at the switches at different times (due to the variation in wire delays from the register to the switch). What is the LSB (in volts) and full scale swing of this DAC (you might want to sketch out the output voltage vs. digital code word). Describe the key problem associated with this architecture. **(5 points)**



(d) Propose a solution that avoids the problem in part (c) and only uses unit sized current sources (I). Add the necessary gate-level logic required to implement the digital-to-analog conversion. **(6 points)**

PROBLEM 4: Memory

(a) The 6264 memory in lab3 is a 8192 x 8 bit memory chip. Architect this memory at the block diagram level (include decoders, amplifier, wordline, etc.). Clearly indicate how many bits are at the input and output of decoders. Assume each row stores 32, 8-bit words (i.e., 256 bits). Also include one line to describe each block. **(14 points)**

Block Diagram:

Provide a 1-line description of the functionality:

Row Decoder:

Column Decoder:

Sense Amplifier:

Wordline:

Bitline:

(b) What are two major differences between SRAM and DRAM? **(4 points)**

PROBLEM 5: Verilog

(a) Complete the behavioral description in Verilog for a synchronous 74163 counter used in lab1. The basic template is shown below -- fill in the necessary code to complete the description. **(8 points)**

```
module counter (LDbar, CLRbar,  P,  T, CLK, D, Q, RCO);
  input LDbar, CLRbar, P, T, CLK;
  input [3:0] D;
  output [3:0] Q;
  output RCO;
  reg [3:0] Q;

 always @ (posedge CLK) begin









 end


 assign RCO =


endmodule
```

(b) Complete the follow piece of code to implement a **synchronizer** (which includes two back to back registers) **(4 points)**

```
input in;
reg q1, q2;

always @ (posedge clk)
begin




end
```

(c) The code below should implement a simple four state FSM which increments a **4-bit synchronous counter** when the FSM is in state 2’b10. Identify the major problem with this code when synthesized. Propose a solution to fix the problem. You can define a new variable if necessary. **(6 points)**

```
 module simple_fsm(clk, reset, count);
  input reset, clk;
  output [3:0] count;
  reg [3:0] count;
  reg [1:0] state;
  reg [1:0] next;

  always @ (posedge clk) begin
    if (reset)       state <= 2'b00;
    else state <= next;
  end

  always @ (state or reset or count) begin
   if (reset)
      count = 4'b0000;
   case (state)
      2'b00:  next = 2'b01;
     2'b01:  next = 2'b10;
      2'b10:  begin
              next = 2'b11;
              count = count +1;
              end
      2'b11:  next = 2'b00;
      default: next = 2’b00;
    endcase
  end //always
endmodule
```