Homework 7

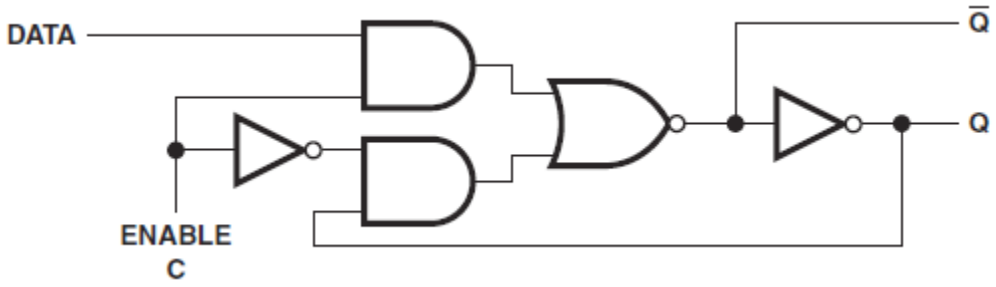
Homework 7 is due on Wednesday, April 8, at the start of the lecture. Remember to include your *Discussions section* (e.g. ED1) and follow the complete Homework submission guidelines.

Please ask all questions about this assignment during the office hours, or post them on piazza.

Sequential logic elements and serialization

1. Circuits with feedback loop

This question pertains to the following circuit.



a. Complete the next-state table for the latch circuit shown above and express next state Q^+ as a function of **C**, **DATA**, and current state **Q**.

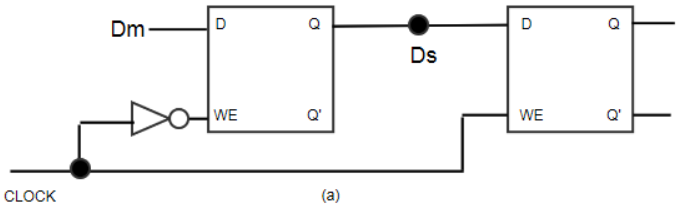
C	DATA	Q^+
0	0	
0	1	
1	0	
1	1	

b. Can this circuit store a bit? Explain your answer in terms of the above truth table.

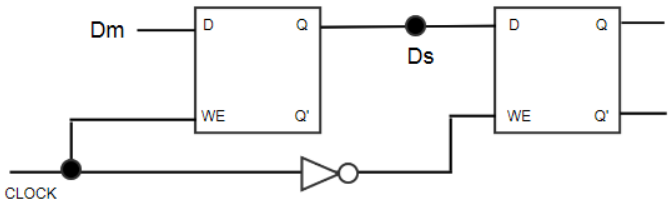
2. Master Slave Flip Flop

Review the two master-slave flip flop designs below. They both are implemented with gated D latches. (NOTE: WE stands for write enable.)

1. Express next -state outputQ^+ of the *master* gated D latch as a function of inputs **Dm**, **CLOCK**, and its current state **Q** for both circuits.
2. Express next -state outputQ^+ of the *slave* gated D latch as a function of inputs **Ds**, **CLOCK**, and its current state **Q** for both circuits.
3. Compare the behavior of the two circuits, and explain the major difference in their functionality.



(a)

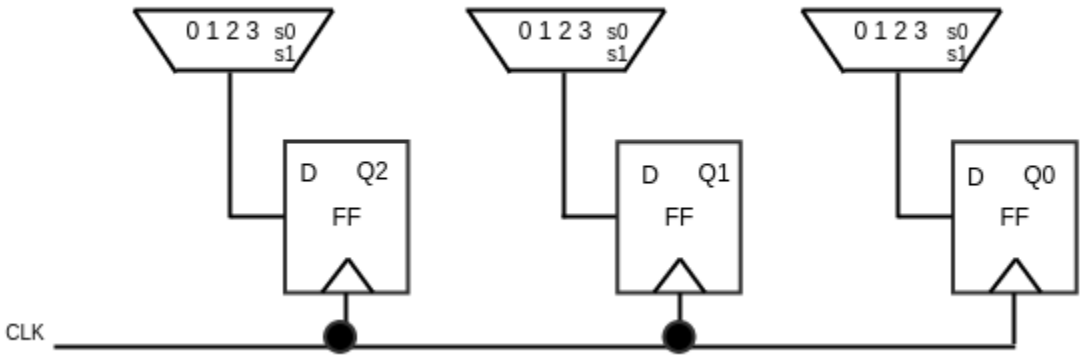


(b)

3. Registers

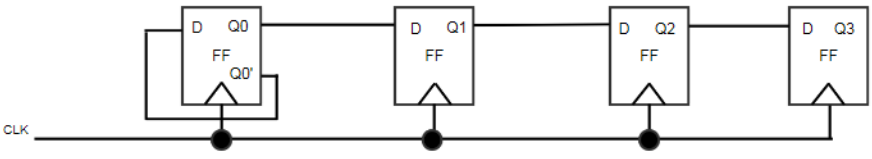
Complete the design of a 3--bit register that performs the operations listed in the table below. Parallel load inputs should be labeled and indexed as $\mathbf{P_i}$. Serial load inputs should be labeled and indexed as $\mathbf{C_i}$. Function selection should be labeled $\mathbf{F_i}$. When arithmetic operations are performed, assume that the numbers are in 2's complement representation. You may use $\mathbf{Q_i}$ inputs without drawing the wire from the outputs of the flip flops (just write the appropriate $\mathbf{Q_i}$ label).

F1	F0	Operation
0	0	Parallel load register
0	1	Logical shift left
1	0	Arithmetic shift right
1	1	Circular shift left

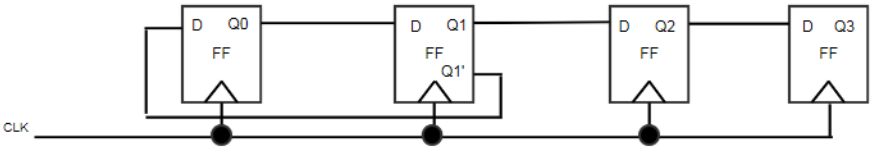


4. Shift Registers

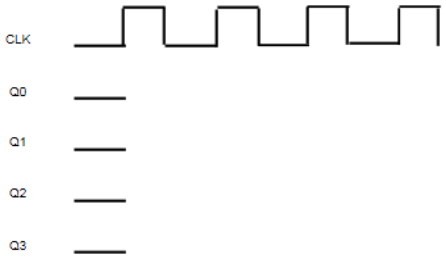
Draw the timing diagrams for $\mathbf{Q_0}$, $\mathbf{Q_1}$, $\mathbf{Q_2}$ and $\mathbf{Q_3}$ for the two circuits below (for the 4 clock cycles shown below). Assume all the $\mathbf{Q_i}$ outputs are at low state at the beginning.



(a)

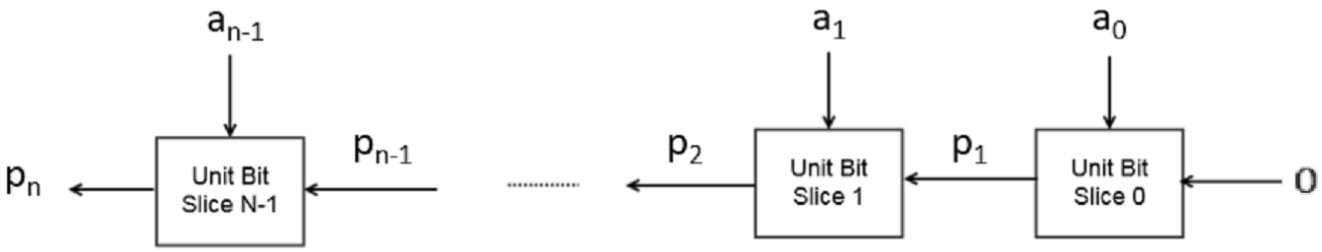


(b)



5. Odd number of 1s checking using serialization

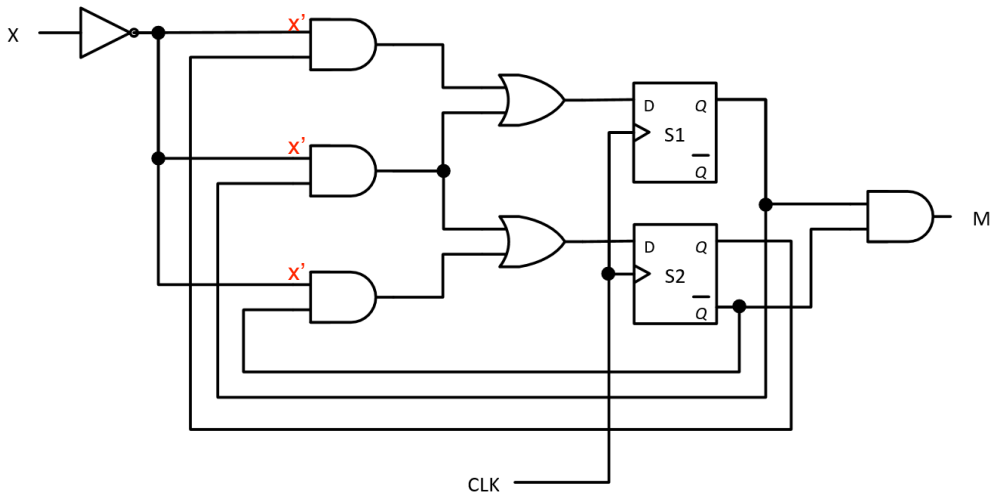
Shown below is a circuit that checks whether the n-bit input A = $a_{n-1}...a_1a_0$ contains an odd number of 1s in its binary representation. The circuit produces an output p_n of 0 if the number of 1s is even, or produces an output p_n of 1 otherwise. In homework 6, you already designed each of the Unit Bit Slices *as identical circuits*.



Re-implement this bit-sliced design using *serialization* approach presented in Prof. Lumetta's notes set 3.1. **Do NOT redraw the gate-level implementation of a Unit Bit Slice from homework 6.** Instead use a box called Unit Bit Slice as a building block and add the storage and logic necessary to turn the bit-sliced design into a serial implementation. Assume that you have an input F to make sure that the Unit Bit Slice starts with the correct data. F is 1 when a_0 enters the Unit Bit Slice and it is 0 at all other times.

6. Analyzing sequential circuits

Below is an implementation for a sequential circuit, with **X** as input, and **M** as the output.



Derive Boolean expressions for next states $\mathbf{S_1^+}$, $\mathbf{S_2^+}$, and output **M** (based on S_1, S_2 and input X).

7. Programming

Download, compile, and execute the program latch.c. The program finds and prints all stable states for an R'-S'-latch. Read the program and examine its output to make sure that you understand how it works.

1. Modify the program to compute stable states for two cross-coupled AND gates. In other words, replace the two NAND calculations with AND. Execute the program to find the stable states of such a circuit. For this part, you may turn in either a printed or a handwritten copy of the output (the list of states).
2. For which combination of inputs R' and S' does the "latch" that you simulated in **part (1)** have two stable states?
3. Since the "latch" simulated in **part (1)** has two stable states, one can use it to store a bit. Give two reasons that such a design (using two AND gates) is inferior to the R'-S'-latch (using two NAND gates) in CMOS technology. Explain your answers.