

Fundamentals of Digital System Design

ECE/CS 3700

Spring 2009, Homework # 4

Due Date: Tue, April 7, by 5pm in the HW locker.

1. **(15 points)** A 4-bit ring counter counts according to the sequence $1000 \rightarrow 0100 \rightarrow 0010 \rightarrow 0001 \rightarrow 1000 \dots$ and repeats. There is a *reset* input, which resets the counter to state 1000. Design this circuit using the following pre-designed components:

- a 2-to-4 decoder;
- a pre-designed, positive edge-triggered, 2-bit up-counter that counts as $00 \rightarrow 01 \rightarrow 10 \rightarrow 11 \rightarrow 00 \rightarrow \dots$ and repeats. In addition to the clock trigger input, this counter has a reset input which can be used to reset it to 00. The counter has outputs q_1, q_0 . Use this counter as a given “black-box”.
- a minimum number of AND/OR/NOT gates, *if needed*.

2. **(15 points)** Consider the circuit of Fig. 7.11 in the textbook, pp 395. This is the *positive edge-triggered* D-flip-flop. We analyzed this circuit in class in detail. Go through the write-up in Section 7.4.2 and understand the operation of this circuit correctly. Recall, in class, we had seen the operation of this circuit as follows:

- When $clock = 0$, set-up the D-value and figure out the assignments on $P1, P2, P3, P4, Q, Q'$, as given in the figure.
- Then, fire the positive edge-trigger and see how Q/Q' update their values.
- After a hold time, change D , but no change in state occurs.

Now, you are asked to solve the following: In Fig. 7.11a, change every NAND gate to a NOR gate, while keeping the same circuit topology, wiring and labels. The circuit will now behave like a *negative edge-triggered D-flip-flop*. You are asked to demonstrate that the circuit indeed behaves like a *negative edge-triggered* DFF. The operation of this flip-flop can be demonstrated using the 3 conditions analogous to the above case: (i) $clk = 1$ setup D; (ii) Fire a negative edge-trigger on clk ; and (iii) After a hold time, change D , but nothing happens.

3. **(10 points)** In class, we have studied *synchronous up-counters* using toggle-flip-flops (TFFs); also given in Fig. 7.22, Section 7.9.2, in the textbook. Design a 4-bit *synchronous down-counter* using TFFs and demonstrate the down-counting operation of the circuit.

4. **(20 points)** Solve problem 7.13 from the textbook.

5. (20 points) Answer the following:

- Define *Setup Time* (T_{su}) and *Hold Time* (T_h) for a *negative* edge-triggered D-type flip-flop.
- The *propagation delay* of a positive edge triggered D-type flip-flop is defined as the time delay between the positive edge of the clock and the corresponding change in the output value (Q). Consider the circuit shown in Fig. 1. It shows two cascaded D flip-flops implementing a shift register. The propagation delay of flip-flop 1 is 5ns, and the hold time of flip-flop 2 is 13ns. Assume that flip-flop 1 is preset to initial value **1** and flip-flop 2 is reset to initial value **0**. The value at the D-input of flip-flop 1 is **0**. Given the hold time and propagation delays, would the shift register work correctly? Why or why not? Feel free to demonstrate your answer using a timing diagram.

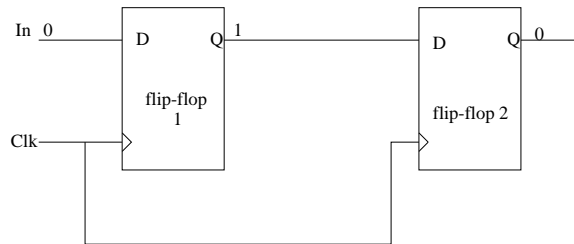


Fig. 1. Cascaded flip-flops

6. (20 points) In class, we examined how to design synchronous up counters using TFFs; and Q2 above asks you to design a down counter using TFFs. Since TFFs are composed of DFFs, it should be easy to design synchronous counters using DFFs! Of course, you've done this in the lab using Verilog. However, note that TFFs are related to DFFs via an XOR relationship. In Fig. 7.16a, $D = TQ' + T'Q = T \oplus Q$. This relationship is also depicted in the discussion on page 411-413 and in Fig. 7.24. These XOR relationships of D_i 's and Q_i 's can be easily related to Table 7.1 (pp. 409) for the purpose of designing counters. That's how the up-counter in Fig. 7.24 was actually designed. Now, analogous to the design of Fig. 7.24, design a *synchronous 3-bit down counter using DFFs*.