1) (10 points) A practical example of Code-Converters. Braille is a system of raised dots that can be read by a blind person. You are asked to design an encoder circuit that converts Binary Coded Decimal (BCD) numbers to Braille. The Braille patterns for the BCD numbers are shown below in Fig. 1.

![Braille patterns](image)

Derive a minimum sum-of-product form representation for each of the four Braille dot outputs \( X, Y, W, Z \) in terms of a 4-bit BCD number. Denote the 4-bit BCD number as \( ABCD \) where \( A \) is the most significant bit, and \( D \) the least significant bit.

2) (5 points) Decoders Solve problem 4.1, pp.243, from the textbook.

3) (5 points) Comparators. Consider \( A[3 : 0], B[3 : 0] \) given as unsigned 4-bit vectors. You are asked to design a \( A > B \) comparators logic circuit. The circuit takes \( A, B \) as inputs and generates a 1-bit output \( f \); where \( f = 1 \) if \( A > B \), and \( f = 0 \) otherwise. Write a Boolean logic formula for \( f \).

4) (15 points) A 4-bit ring counter counts according to the sequence 1000 → 0100 → 0010 → 0001 → 1000 · · · 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 \( 00 \rightarrow 01 \rightarrow 10 \rightarrow 11 \rightarrow 00 \rightarrow \cdots \) 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.
5) **(15 points)** Consider the circuit of Fig. 5.11 in the textbook, pp 260. This is a *positive edge-triggered* D-flip-flop. We analyzed this circuit in class in detail. Go through the write-up in Section 5.4.2 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. 5.11a, 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 cases: (i) *clk* = 1, setup *D*; (ii) Fire a negative edge-trigger on *clk*; and (iii) After a hold time, change *D*, but nothing happens.

6) **(10 points)** In class, we have studied *synchronous up-counters* using toggle-flip-flops (TFFs); also given in Fig. 5.21, Section 5.9.2, in the textbook. Design a 4-bit *synchronous down-counter* using TFFs and demonstrate the down-counting operation of the circuit.

7) **(20 points)** Solve problem 5.12 from the textbook.

8) **(20 points)** Timing issues in shift registers. 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. 2. 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. 2. Cascaded flip-flops](image)