

1) Number representation

a) Convert the following binary numbers to unsigned decimal numbers (e.g., “1000” = 8):

100	
0000000000000001	
111111	
101010	

b) Convert the following numbers to a 5-bit 2’s complement binary representation (e.g. 3 = “00011” and $-3 = “11101”$):

9	
15	
-16	
-5	

c) Convert the following hexadecimal numbers to binary and octal (e.g. AA = “10101010” in binary or “252” in octal):

hexadecimal	binary	octal
FF		
F0A2		
0F100		
100		

2) Two-level logic minimization

$$F(A, B, C, D) = \sum m(0, 3, 6, 8, 9)$$

$$d(A, B, C, D) = \sum m(11, 12, 15)$$

		CD			
		00	01	11	10
AB	00				
	01				
	11				
	10				

a) Identify all the prime and essential prime implicants.

Prime	Essential

b) Find the minimum two-level logic implementation.

$F =$

3) Multi-level logic minimization

Given following logic equations, minimize the number of literals, e.g. by using common subexpressions, Boolean rules, etc. You can introduce new immediate equations for common sub-expressions if it helps to reduce the number of literals. Put a box around your final answer, and indicate the number of literals in your final answer.

$$M = A\bar{D}\bar{E}\bar{F} + B\bar{D}\bar{E}\bar{F} + C\bar{D}\bar{E}\bar{F} + \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}\bar{C}E + \bar{A}\bar{B}\bar{C}F$$

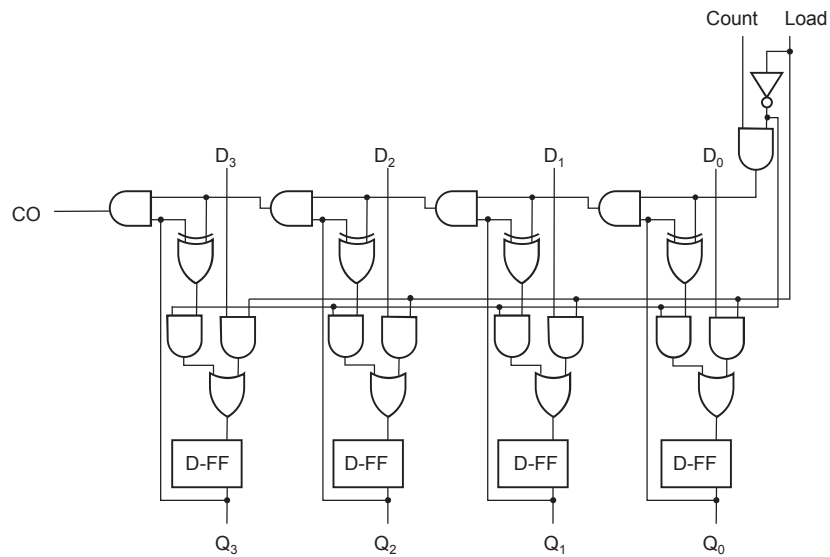
$$N = A + B + C + D + E + F + GHK + GH$$

$$P = AD + BD + CD + AE + BE + CE + AF + BF + CF + G + \bar{G} + H + K$$

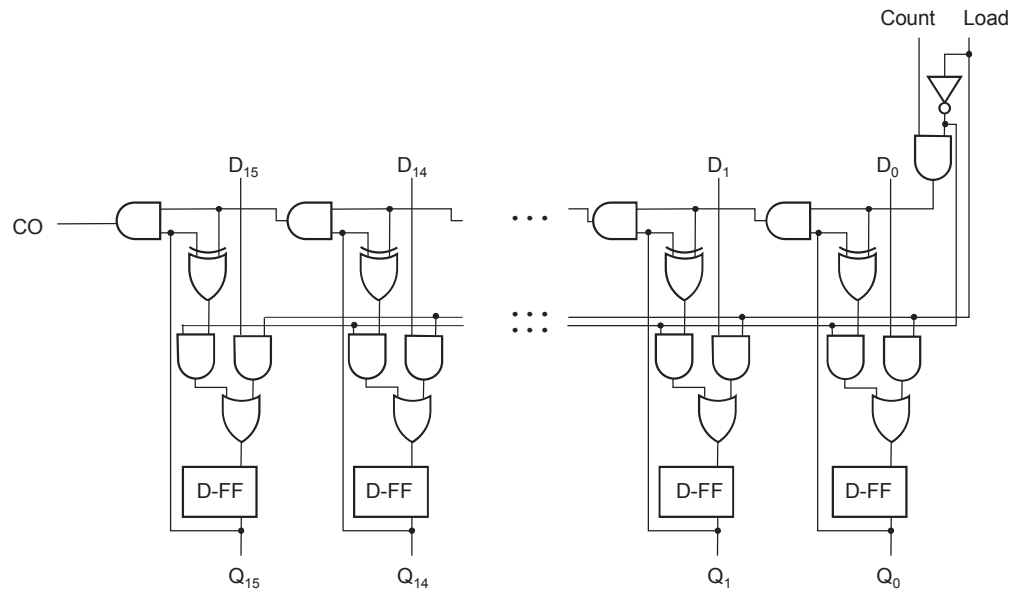
4) Clock period analysis

Consider the following 4-bit counter design, which loads the counter (Q_3, Q_2, Q_1, Q_0) with data (D_3, D_2, D_1, D_0) when “Load = 1”; otherwise, when “Count = 1”, the counter (Q_3, Q_2, Q_1, Q_0) gets incremented. Assume all gates delays are **1 ns** (e.g., AND, OR, XOR, INV). Assume the positive edge-triggered flip-flop delay is $T_{delayFF} = \mathbf{1\ ns}$, and setup time is also $T_{setupFF} = \mathbf{1\ ns}$.

a) What is the **minimum clock period** that would satisfy these delays? Clock period =



b) What is the **minimum clock period** for a 16-bit version of the same counter? Clock period =



5) Sequential logic implementation

Consider the following Moore machine with initial state A . (“*” means don’t care.)

$S_2 S_1 S_0$	$S_2^+ S_1^+ S_0^+$		F
	$J = 0$	$J = 1$	
A	D	B	1
B	B	A	*
C	C	F	0
D	C	B	0
E	B	F	1
F	A	E	1

Assume the following state encoding.

State	$S_2 S_1 S_0$
A	000
B	010
C	100
D	001
E	011
F	101

Fill-out the following K-Maps.

		S_2^+			
		00	01	11	10
S_2	S_1				
00					
01					
11					
10					

		S_1^+			
		00	01	11	10
S_2	S_1				
00					
01					
11					
10					

		S_0^+			
		00	01	11	10
S_2	S_1				
00					
01					
11					
10					

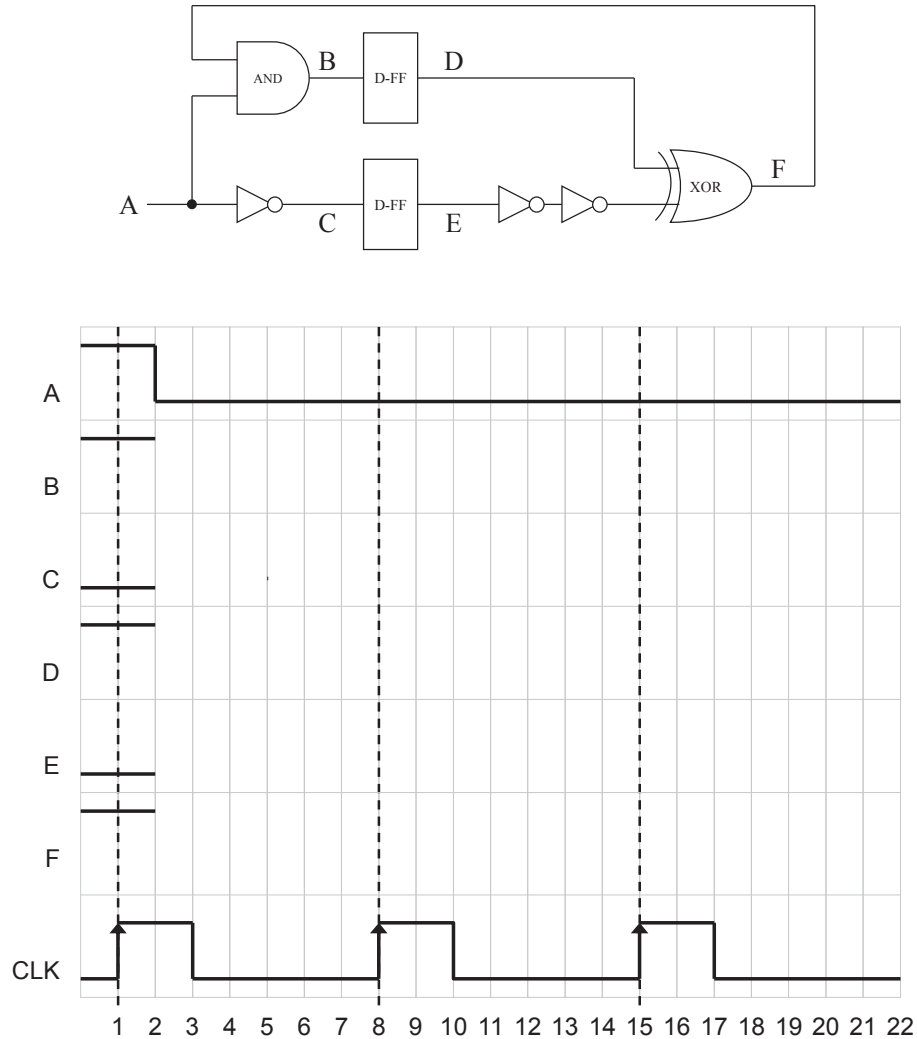
		F	
		0	1
S_2	S_1		
00			
01			
11			
10			

6) Timing analysis

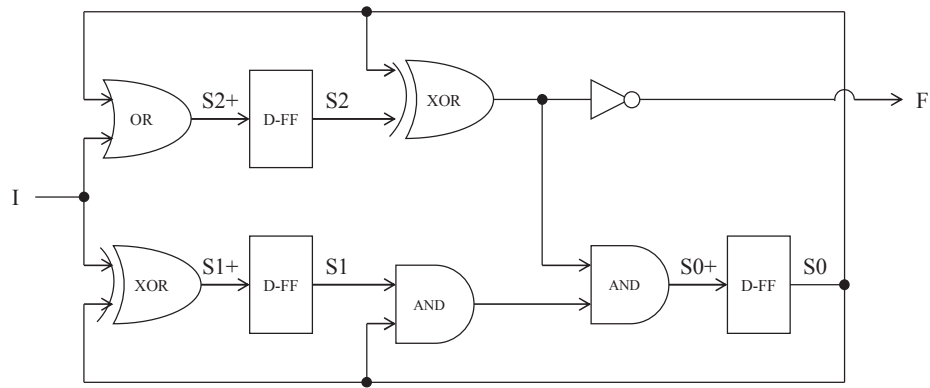
Consider the following sequential logic diagram.

Assume all gate delays are **1 ns** (AND, XOR, and INV). Assume the positive edge-triggered flip-flop delay is $T_{delay_{FF}} = \mathbf{1\ ns}$, and setup time is also $T_{setup_{FF}} = \mathbf{1\ ns}$.

Fill out the rest of the timing diagram for A , B , C , D , E and F .



7) Reverse engineer sequential circuit to finite state machine



a) Fill out the following functions.

$$\begin{aligned}
 F &= \\
 S_2+ &= \\
 S_1+ &= \\
 S_0+ &=
 \end{aligned}$$

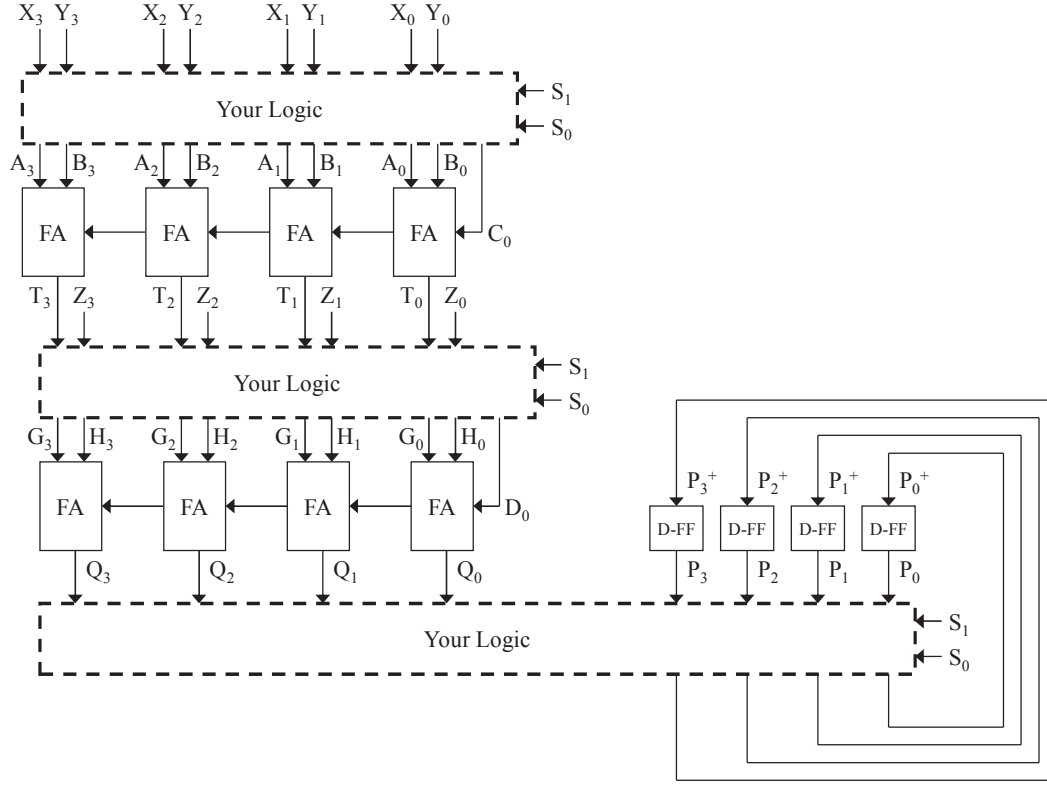
b) Assume initial state is **000**, fill out the following state table. **Important: Only include reachable states in the state table.**

$S_2S_1S_0$	$S_2^+S_1^+S_0^+$		F
	$I = 0$	$I = 1$	

c) With initial state = **000**, what states, if any, are **not** reachable?

8) Datapath/registers

Consider the following diagram.



The external data inputs are $X_3X_2X_1X_0$, $Y_3Y_2Y_1Y_0$, and $Z_3Z_2Z_1Z_0$, and the external control signals are S_1S_0 . We would like to implement the following operations:

S_1S_0	Operations
00	$P^+ = X + Y - (Z_2, Z_1, Z_0, Z_3)$
01	$P^+ = X - Y + (Z_0, Z_3, Z_2, Z_1)$
10	$P^+ = (P_2, P_1, P_0, P_3)$
11	$P^+ = (P_0, P_3, P_2, P_1)$

where “+” means “addition”, “−” means “subtraction”, X means $X_3X_2X_1X_0$, Y means $Y_3Y_2Y_1Y_0$, and P^+ means $P_3^+P_2^+P_1^+P_0^+$.

To implement the above computations, please describe:

a) the logic required to implement A_2 , B_2 , and C_0 as functions of S_1S_0 , $X_3X_2X_1X_0$, and $Y_3Y_2Y_1Y_0$,

$$A_2 =$$

$$B_2 =$$

$$C_0 =$$

b) the logic required to implement G_2 , H_2 , and D_0 as functions of S_1S_0 , $T_3T_2T_1T_0$, and $Z_3Z_2Z_1Z_0$,

$$G_2 =$$

$$H_2 =$$

$$D_0 =$$

c) and the logic required to implement P_2^+ as a function of S_1S_0 , $Q_3Q_2Q_1Q_0$, and $P_3P_2P_1P_0$.

$$P_2^+ =$$

You should minimize each logic equation. No need to worry about “overflow” problems or “overflow” logic.