

1) Sequential logic implementation

Consider the following incompletely-specified Moore machine (“X” indicates don’t cares).

S_1S_0	$S_1^+S_0^+$				F
	$XY = 00$	01	10	11	
A	B	C	X	B	0
B	A	B	X	C	X
C	C	A	X	C	1

Suppose we use the following state encoding.

State	S_1S_0
A	00
B	11
C	01

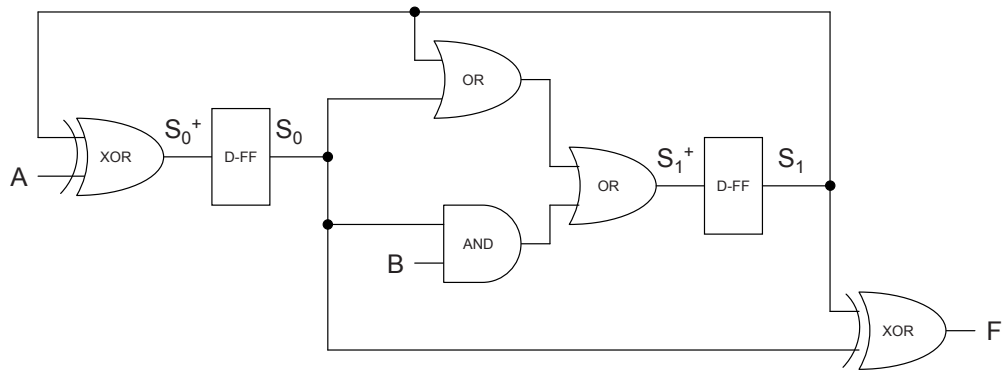
Derive the **minimum** two-level logic for each of the following (you should minimize each logic function separately, no need to worry about sharing logic between logic functions).

$$F = S_0$$

$$S_1^+ = \bar{S}_0X + \bar{S}_0\bar{Y} + S_1\bar{X}Y$$

$$\begin{aligned} S_0^+ &= X + S_1Y + \bar{S}_1\bar{Y} + \bar{S}_1\bar{S}_0 \text{ OR} \\ &= X + S_1Y + \bar{S}_1\bar{Y} + \bar{S}_0 \end{aligned}$$

2) Reverse engineer sequential circuit to finite state machine



Answer:

a) Fill out the following functions (OK to use $\oplus$ in functions).

$$\begin{aligned} F &= S_1 \oplus S_0 \\ S_1^+ &= (S_1 + S_0) + S_0 B = S_1 + S_0 \\ S_0^+ &= A \oplus S_1 \end{aligned}$$

b) Assume initial state is **00**, fill in the following state table. If a state is not reachable, indicate the appropriate **don't cares** (use "X" to indicate don't cares).

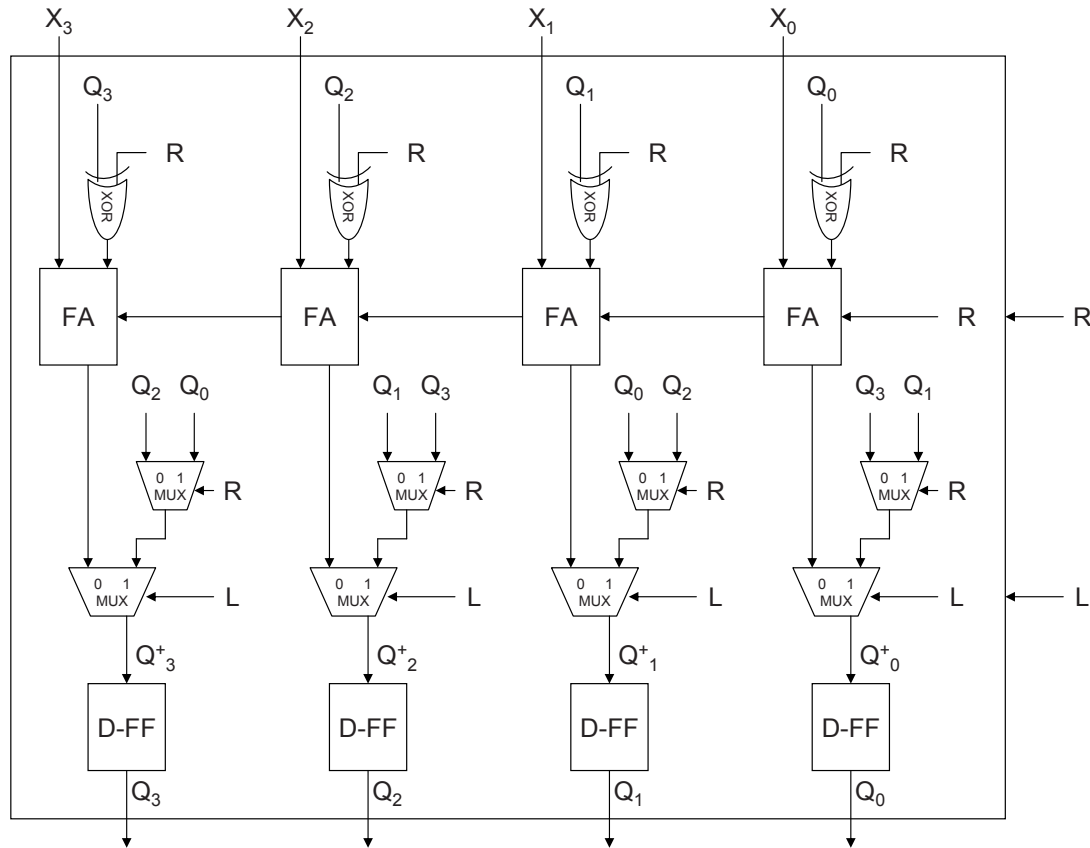
$S_1 S_0$	$S_1^+ S_0^+$				F
	$AB = 00$	01	10	11	
00	00	00	01	01	0
01	10	10	11	11	1
10	11	11	10	10	1
11	11	11	10	10	0

c) With initial state = **00**, what states, if any, are **not** reachable?

No unreachable states. All states reachable.

3) Arithmetic/logic and registers

Consider the following diagram. To avoid drawing many messy wires, the **name** of a wire is used in many cases instead. For example, the output of the far right D-FF, Q_0 , is connected to the input of the far right XOR gate.



Given the following inputs (L , R , $X_3X_2X_1X_0$) and flip-flop values ($Q_3Q_2Q_1Q_0$), what will be the new flip flop values ($Q_3^+Q_2^+Q_1^+Q_0^+$) ?

Answer:

LR	$X_3X_2X_1X_0$	$Q_3Q_2Q_1Q_0$	$Q_3^+Q_2^+Q_1^+Q_0^+$	
00	0100	0011	0111	Operation is “addition” = $X + Q$
01	0100	0011	0001	Operation is “subtraction” = $X - Q$
10	0100	0011	0110	Operation is “rotate LEFT”
11	0100	0011	1001	Operation is “rotate RIGHT”

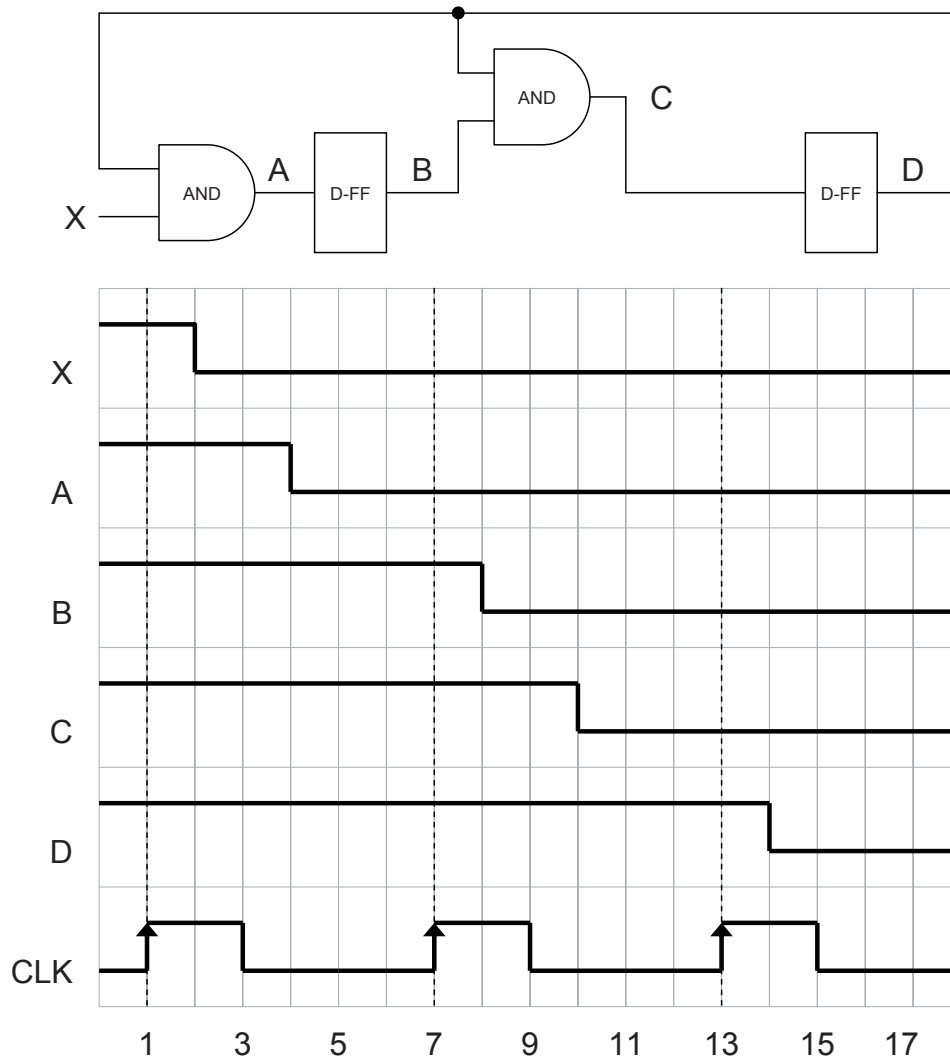
4) Timing analysis

Consider the following sequential logic diagram.

Assume the gate delay for 2-AND is **2 ns**, assume the positive edge-triggered flip-flop delay is $T_{delay_{FF}} = 1 \text{ ns}$, and assume the positive edge-triggered flip-flop setup time is also $T_{setup_{FF}} = 1 \text{ ns}$.

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

Answer:



5) Clock period analysis

Consider the following sequential logic diagram where X_3, X_2, X_1, X_0 , and Y_3, Y_2, Y_1, Y_0 are **external inputs**, and F_3, F_2, F_1, F_0 are **external outputs**.

Assume the gate delay for 2-XOR is **3 ns**, assume the gate delay for INVERTER is **1 ns**, assume the positive edge-triggered flip-flop delay is $T_{delay_{FF}} = \mathbf{1\ ns}$, and assume the positive edge-triggered flip-flop setup time is also $T_{setup_{FF}} = \mathbf{1\ ns}$.

What is the **minimum clock period** that would satisfy these delays? Please explain how you computed the clock period (e.g. give the equation).

Answer: Clock period = $T_{delay_{FF}} + T_{INV} + T_{XOR} + T_{setup_{FF}} = 6\ \text{ns}$.

