

1) Number representation

a) For each binary vector below, what does it represent as an Octal number and a Hexadecimal number?

Binary	Octal	Hexadecimal
11010001	321	D1
00110011	063 (63 OK)	33

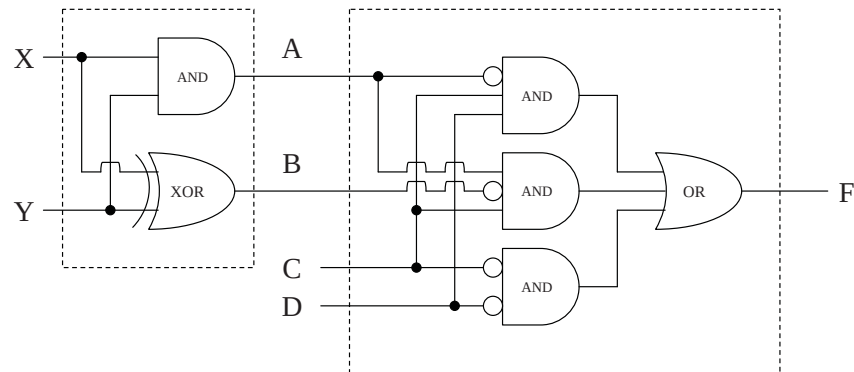
b) Given X and Y below that are 4-bit numbers in 2's complement form, fill in the table of what S should be bit-vector form, whether or not there is an *overflow*, and what S is as a number. (First row provides an example.)

X $x_3x_2x_1x_0$	Y $y_3y_2y_1y_0$	$S = X - Y$ $s_3s_2s_1s_0$	Overflow?	S as a number
0111	0011	0100	No	4
0110	1110	1000(x)	Yes	10(x)
0000	1111	0001	No	1
0111	0000	0111	No	7
1000	1100	1100	No	-4

“x” means number is not valid (because an overflow has occurred), so doesn't matter what they write down, just as long as they got the “overflow” part correct.

2) Two-level logic minimization

Consider the logic diagram shown below.



a) The second block $F(A, B, C, D)$ is a function of A , B , C , and D . Please fill out the following K-map by taking into consideration combinations of A and B that *cannot* occur as *don't cares* because A and B are themselves functions of X and Y .

		F			
		CD	00	01	11
AB	00	1		1	
	01	1		1	
	11	x	x	x	x
	10	1		1	1

b) For the K-map derived in part (a) for F , specify all the primes and essential primes.

Prime	Essential
- - 1 1 CD	X
- - 0 0 $\bar{C}\bar{D}$	X
1 1 0 0 AB	
1 - 1 - AC	
1 - - 0 $A\bar{D}$	

c) Find the minimum two-level logic implementation for F .

$$\begin{aligned}
 F &= CD + \bar{C}\bar{D} + AC \text{ OR} \\
 &= CD + \bar{C}\bar{D} + A\bar{D}
 \end{aligned}$$

3) Finite state machines

Consider the following Mealy state machine with A as the initial state.

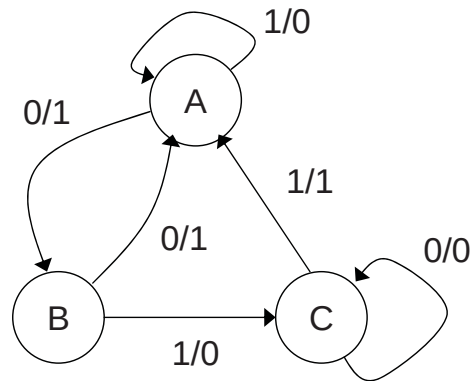


Figure 1: Mealy machine.

If the input sequence is 011, then the corresponding output sequence, $A \xrightarrow{0/1} B \xrightarrow{1/0} C \xrightarrow{1/1} A$, would be **101**.

- What would be output sequence if the input sequence is 100? **011**, $A \rightarrow A \rightarrow B \rightarrow A$ also OK.
- What would be output sequence if the input sequence is 001? **110**, $A \rightarrow B \rightarrow A \rightarrow A$ also OK.
- Consider the following Mealy state machine with R as the initial state.

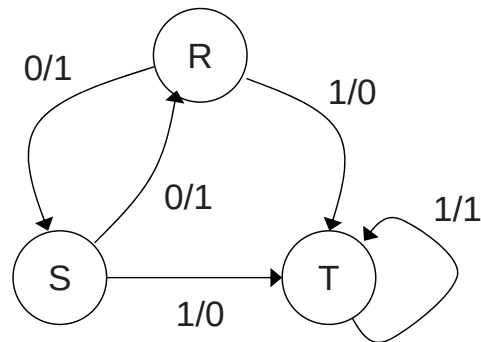


Figure 2: Mealy machine.

Part 1) If the input is **fixed** to a constant **1** for all cycles, then will the Mealy machine shown in Fig. 2 produce the **same output** as the Mealy machine in shown Fig. 1 starting from their respective initial states? **No**

Part 2) If the input is **fixed** to a constant **0** for all cycles, then will the two Mealy machines produce the **same outputs**? You have to get both parts correct to get points so that you can't just "guess" the answer. **Yes**

4) 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.

$$\begin{aligned}H &= \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + A\bar{B}\bar{C} + A\bar{B}C + AB\bar{C} + ABC \\&= \bar{A}\bar{B}(\bar{C} + C) + A\bar{B}(\bar{C} + C) + AB(\bar{C} + C) \\&= \bar{A}\bar{B} + A\bar{B} + AB \\&= \bar{B} + A = A + \bar{B} \\J &= A + \bar{B} \\&= H \\K &= \bar{A}BE(\bar{A}BE + CDE) \\&= \bar{A}BE \\&= \bar{H}E \\L &= AD + \bar{A}\bar{B}D + \bar{A}\bar{B}CD \\&= AD + \bar{A}\bar{B}D \\&= (A + \bar{A}\bar{B})D \\&= (A + \bar{B})D \\&= HD\end{aligned}$$

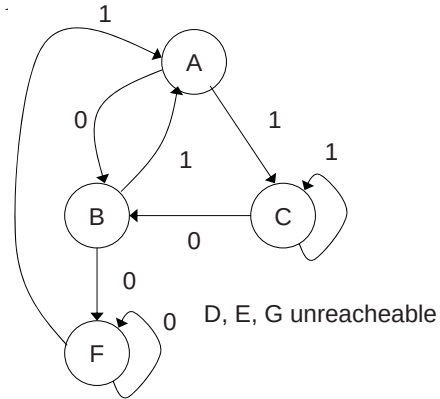
Note: J and H are the same and are interchangeable.

Final answer: 7 literals

5) Sequential logic implementation

Consider the following Moore machine with initial state .

$S_2 S_1 S_0$	$S_2^+ S_1^+ S_0^+$		Z
	$I = 0$	$I = 1$	
A	B	C	0
B	F	A	1
C	B	C	0
D	E	C	1
E	F	D	0
F	F	A	1
G	A	F	0



Assume the following state encoding.

State	$S_2 S_1 S_0$
A	000
B	001
C	100
D	101
E	011
F	111
G	010

Figure 3: State encoding.

State	$S_2 S_1 S_0$	Z	Comments
A	000	0	
B	001	1	
G	010	*	unreachable
E	011	*	unreachable
C	100	0	
D	101	*	unreachable
	110	*	unused code
F	111	1	

Figure 4: Logic table.

Derive the **minimum** two-level logic for the output Z by taking into consideration what states are reachable from the initial state A.

$S_1 S_0$		Z			
		00	01	11	10
S_2	0		1	x	x
	1		x	1	x

$$Z = S_0 \quad (\text{full 10 pts})$$

Correct K-Map with all don't cares

$S_1 S_0$		Z			
		00	01	11	10
S_2	0		1	0	0
	1		1	1	-

K-Map with don't cares missing

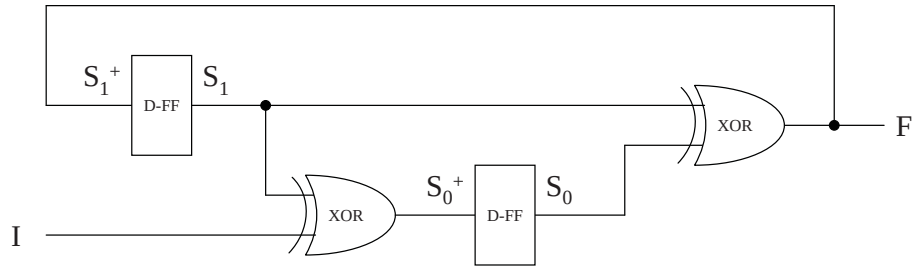
$$Z = \overline{S_2} \overline{S_1} S_0 + S_2 S_1 S_0 \quad \text{OR}$$

$$Z = \overline{S_1} S_0 + S_2 S_0 \quad \text{OR}$$

$$Z = \overline{S_1} S_0 + S_2 S_1$$

(4 pts for these solutions)

6) Reverse engineer sequential circuit to finite state machine



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

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

Note: saying $S_1+ = F$ also OK.

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 "*" to indicate don't cares).

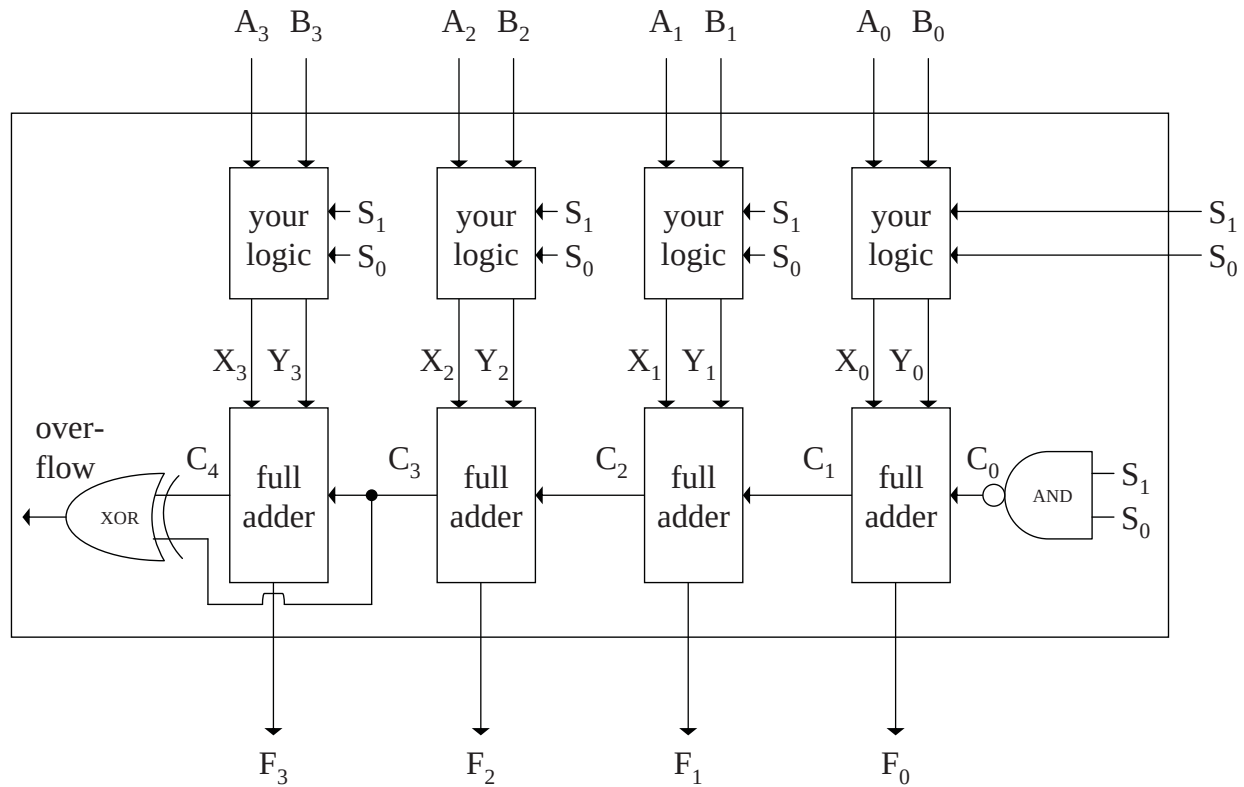
$S_1 S_0$	$S_1+ S_0+$		F
	$I = 0$	$I = 1$	
00	00	01	0
01	10	11	1
10	11	10	1
11	01	00	0

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

None

7) ALU design

Consider the following diagram with two sets of data inputs $A_3A_2A_1A_0$ and $B_3B_2B_1B_0$, two selection bits S_1 and S_0 , and one set of outputs $F_3F_2F_1F_0$.



Consider the following four ALU operations:

S_1	S_0	ALU operation
0	0	$-A$ (two's complement of A)
0	1	$-B$ (two's complement of B)
1	0	$A - B$
1	1	$A + B$

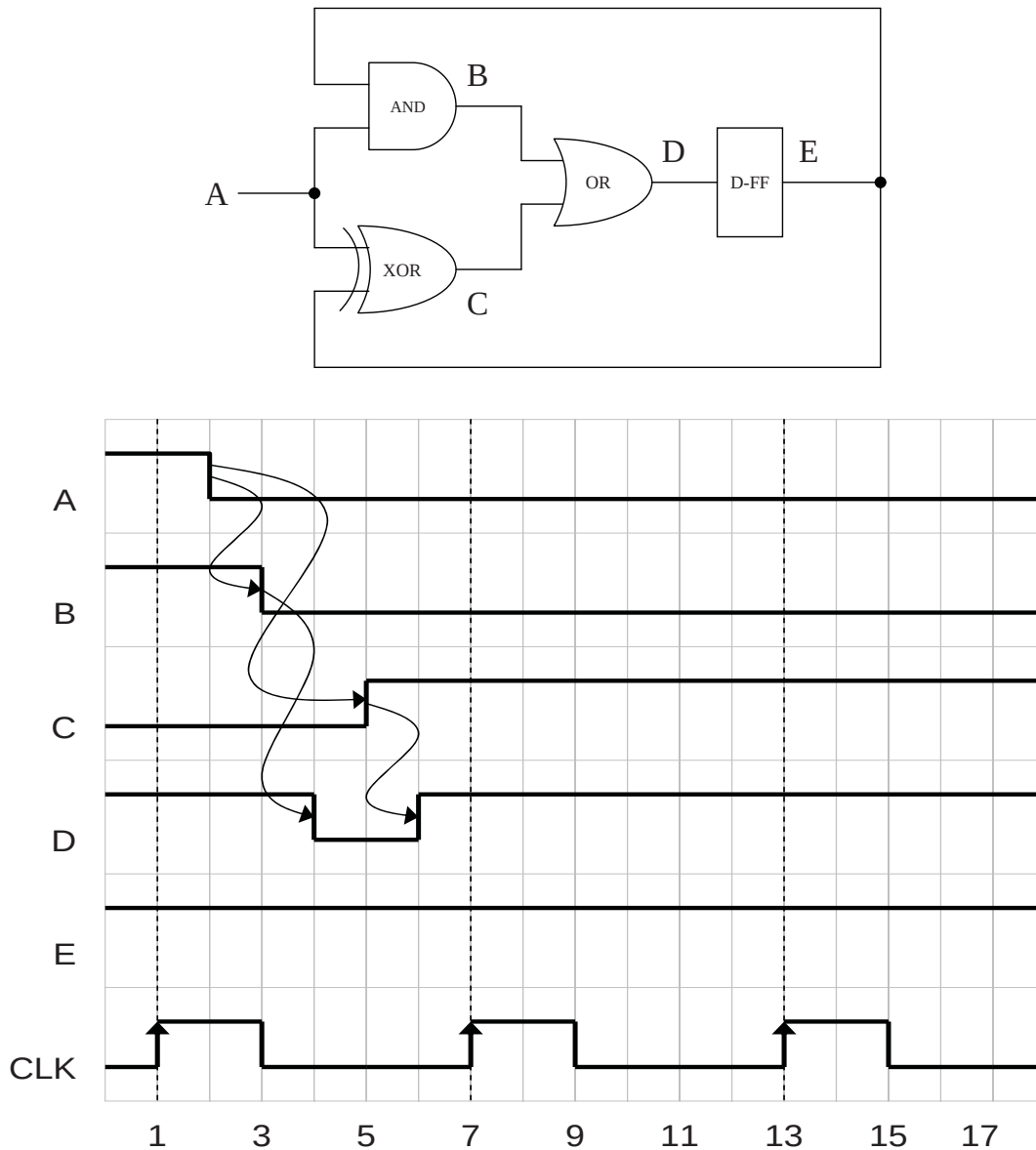
Specify the minimized logic equations for X_i and Y_i of the “your function” box shown in the diagram. (X_i and Y_i are both functions of A_i , B_i , S_1 , S_0 .)

$$\begin{aligned}
 X_i &= S_1 A_i + \bar{S}_1 \bar{S}_0 \bar{A}_i \\
 Y_i &= (S_1 \oplus S_0) \bar{B}_i + S_1 S_0 B_i \text{ OR} \\
 &= (\bar{S}_1 S_0 + S_1 \bar{S}_0) \bar{B}_i + S_1 S_0 B_i \text{ OR} \\
 &= \bar{S}_1 S_0 \bar{B}_i + S_1 \bar{S}_0 \bar{B}_i + S_1 S_0 B_i
 \end{aligned}$$

8) Timing analysis

Consider the following sequential logic diagram. Assume the gate delay is **1 ns** for both 2-AND and 2-OR. Assume the gate delay is **3 ns** for 2-XOR. 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 $T_{setup_{FF}} = 1 \text{ ns}$.

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



b) The clock period shown in the diagram is **6ns**. Can the clock safely run faster? If yes, how fast can we run the clock (what is the shortest feasible clock period)? **No, 6ns is fastest.**

9) Delay analysis

Consider the following 32-bit “priority encoder” with inputs $R_{31}R_{30}\cdots R_1R_0$ (“requests”) and outputs $G_{31}G_{30}\cdots G_1G_0$ (“grants”). When **both** $R_i = 1$, **and** there is no $R_j = 1$ such that $j > i$, then $G_i = 1$; otherwise, $G_i = 0$. The logic in Fig. 5 is designed so that at most one G_i is equal to 1, corresponding to the highest $R_i = 1$ request. The output “status = 1” indicates that there was a grant. Assume **1ns** gate delay for all gates (i.e., 2-AND, 2-OR, and INV).

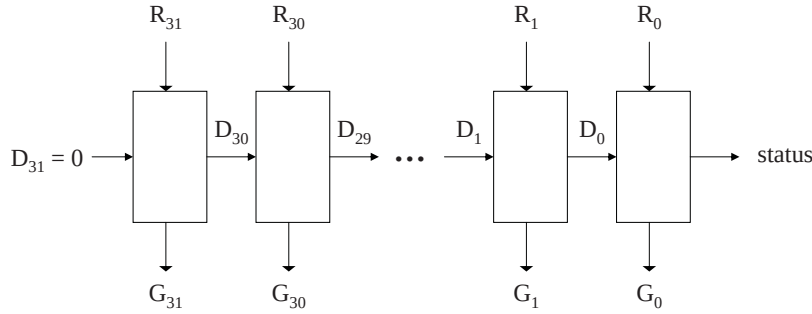


Figure 5: Priority encoder.

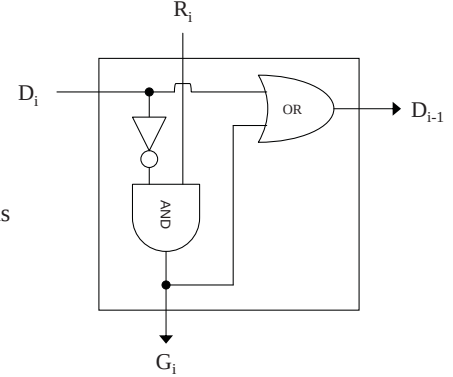


Figure 6: Logic for rectangle box.

a) What is the longest delay in Fig. 5 (in ns)? **3 * 32 = 96ns**

b) Which output in Fig. 5 determines the longest delay? **status**

Now consider the following modified diagram (same gate delays and same Fig. 6 rectangle box).

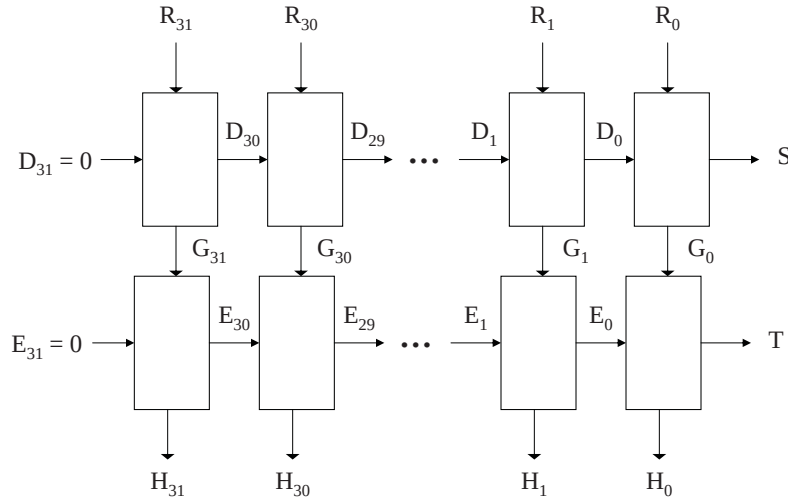


Figure 7: Modified design.

c) What is the longest delay in Fig. 7 (in ns)? **97 ns**

d) Which output in Fig. 7 determines the longest delay? **T**