FINAL EXAM REVIEW

① Simplify: algebraically:

(a) $XY + Y'Z + XZ = ?$

Answer:

$XY + Y'Z + XZ = XY + Y'Z$ by consensus theorem

(b) $(X' + Y)^D = ?$

Answer:

$(X' + Y)^D = X' \cdot Y$

(c) $\left((X' + Y)^D\right)' = \left((X' + Y)'\right)^D$ TRUE/FALSE?

Answer:

$LHS = \left((X' + Y)^D\right)' = (X' \cdot Y)' = X + Y'$

$RHS = \left((X' + Y)'\right)^D = (X \cdot Y')^D = X + Y'$ ∴ TRUE

(d) $\left((f(X,Y))^D\right)' = \left(f(X,Y)'\right)^D$ TRUE/FALSE?

Answer:

Recall: $f^D(X,Y) = f'(X',Y')$

$\therefore$ LHS $= \left((f(X,Y))^D\right)' = \left(f'(X',Y')\right)' = f(X',Y')$

($=$: De Morgan's)

RHS $= \left((f(X,Y))'\right)^D = f(X,Y)\Big|_{\substack{X\to X' \\ Y\to Y'}} = f(X',Y')$ $\therefore$ TRUE

(e) $X + YX'Z = ?$

Answer:

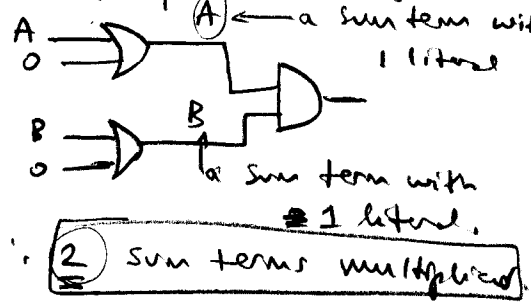
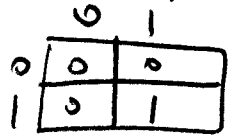
Use 2nd distributive law

$X + X'(YZ) = (X + X')(X + YZ) = X + YZ$

($X + X' = 1$)

② Basic Definitions:

$F = A \cdot B$

	Notes
(a) minterm expansion of F: $A \cdot B$	M_3
(b) maxterm expansion: $(A+B)(A+B')(A'+B)$.	$F = \overline{m_3} = M_0 \cdot M_1 \cdot M_2$ $(A+B)$ $(A+B')$ $(A'+B)$
(c) minimum sum of products: $A \cdot B$	(1) product term with 2 literals
(d) minimum product of sums: $A \cdot B$	Defn: Should have the min. # of sum terms  ∴ (2) sum terms multiplied (Impossible to have 1 sum term) $A'B'$ $A'+B$ $A+B'$ $A+B$ $A+B'$ only possible
(e) Implicants: $A \cdot B$	any product terms that make $F = 1$ 

(f) Prime implicants,

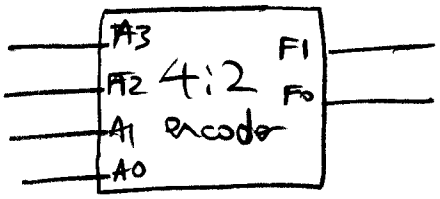
$A \cdot B$

(g) Essential prime implicants

$A \cdot B$

③ Combinational Verilog design problem :

- Write a Verilog module to implement an "encoder"; which implements the inverse fn of the 2:4 decoder.



A3	A2	A1	A0	F1	F0
1	0	0	0	1	1
0	1	0	0	1	0
0	0	1	0	0	1
0	0	0	1	0	0
(everything else)				x	x

```
module Encoder4_2(A, F);
    input[3:0] A;
    output[1:0] F;
    :
    :
endmodule
```

(a) Using only continuous assignments (with "assign" statemen

```
assign F1 = A[3] | A[2];   // assume no invalid inputs to encoder.

assign F0 = A[3] | A[1];
```

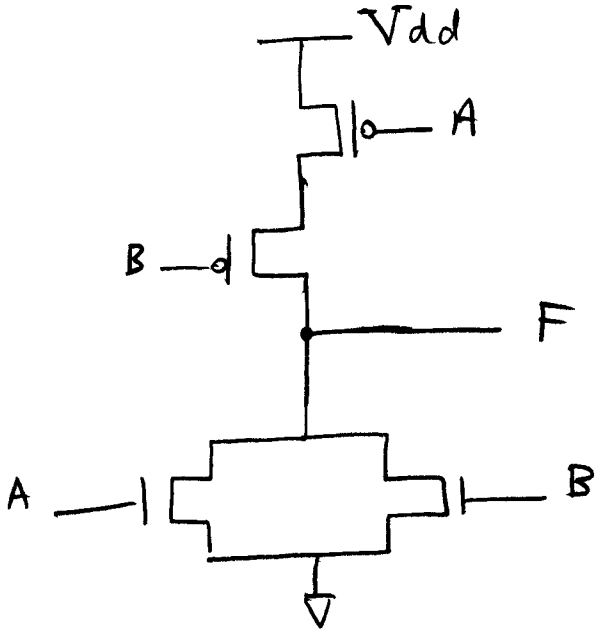
(b) using only procedural assignments.

```
reg [1:0] F;

always @(A)
    case (A)
        4'b1000 : F = 2'b11;
        4'b0100 : F = 2'b10;
        4'b0010 : F = 2'b01;
        4'b0001 : F = 2'b00;
        default : F = 2'bxx;
    endcase
```

(c

④ CMOS implementation:



(a) Is this a valid CMOS circuit?

Answer: Pull-up: $F = A' \cdot B'$

Pull-down: $\overline{F} = A + B$

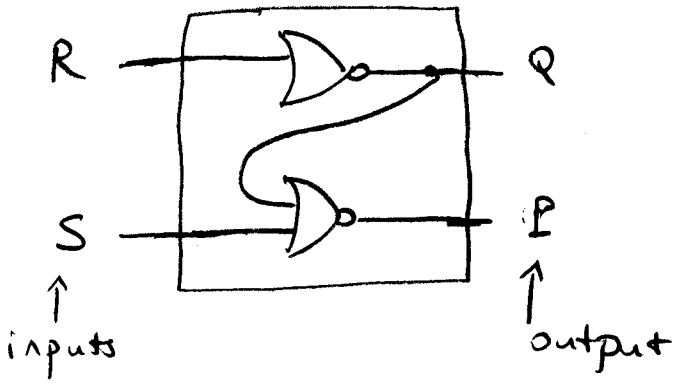
↖ ↙ valid ✓ since complements

(b) what fn does this implement?

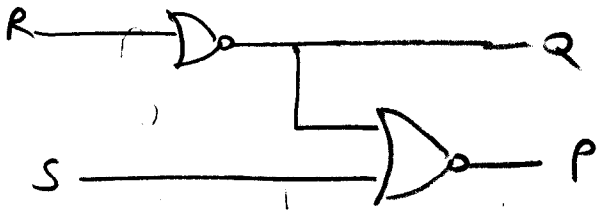
$\boxed{F = A' \cdot B'}$. (NOR)

⑤ Basic Definitions:

(a) Is the following circuit combinational?

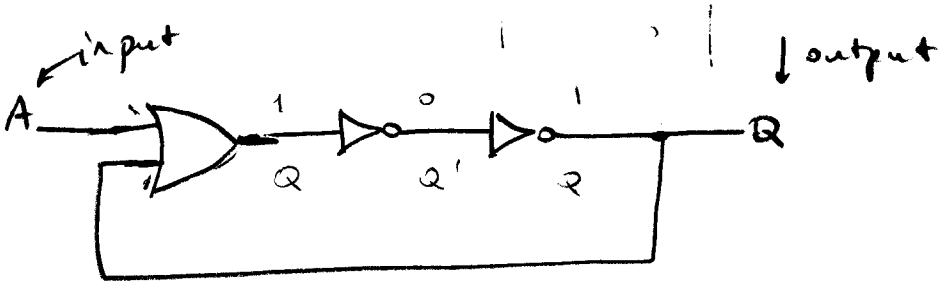


Answer: combinational iff $(Q, P)^{current}$ determined completely by $(R, S)^{current}$.



∴ **combinational**

(b) Combinational?



Answer:

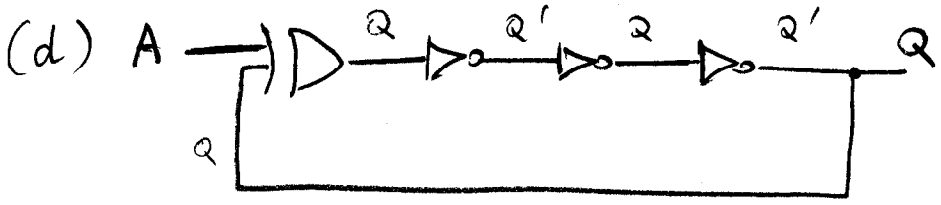
A	Q
1	1
0	hold Q

∴ **NO**

∴ **Sequential**

(c) can the circuit in (b) be used as a memory cell?

Answer: No: because no way to write in a 0.

(d) 

(d.1): combinational?

Answer:

A	behavior
1	hold Q
0	oscillatory Q.

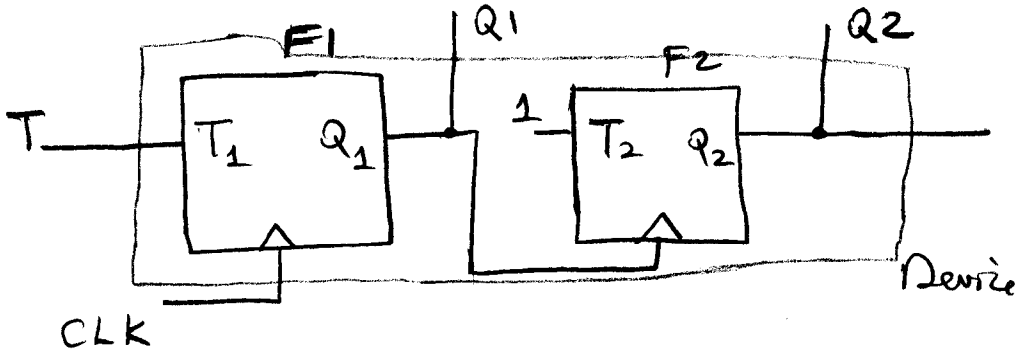
No because Q's value not completely determined by current A.

(d.2) Can this device be used as a memory ~~element~~ cell?

No: because A = 0; Q oscillates.

- no way to write anything to the device.

(e) Is the following device synchronous?



Q: output
T: input.

Defn #1: of "synchronous device"

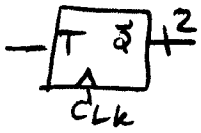
Answer:

asynchronous device ≡ at least 1 asynchronous input.

there, T only input (data). ∴, ~~does~~ can $\bar{Q}$ change without regard to the CLK input? No.

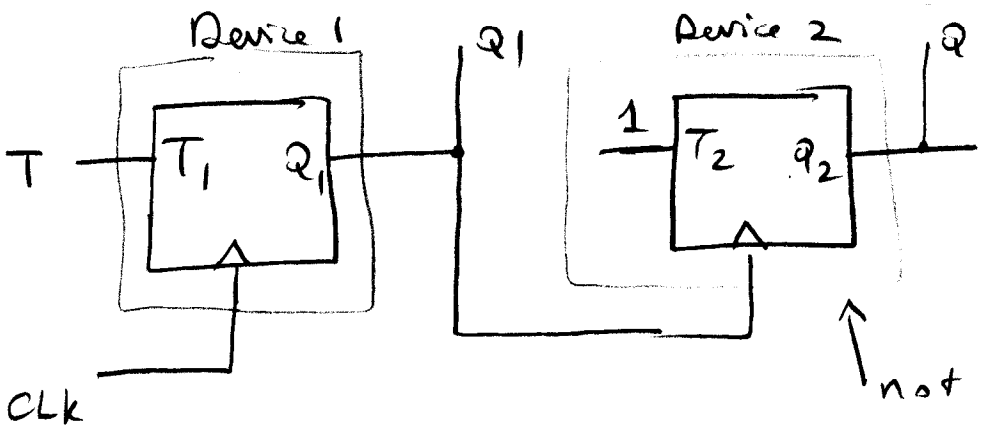
∴ As a device: the device is synchronous.

(as a black box)



Defn 2:

(f) "Synchronous design": All memory devices clocked by same global clock (or set of clocks)



not clocked by global clock

∴ asynchronous design.

(according to Defn 2)

(g) Part (e): Compute:

(g.1): Set-up time of Device:

$$t_{su} = t_{su}^{(F1)}.$$

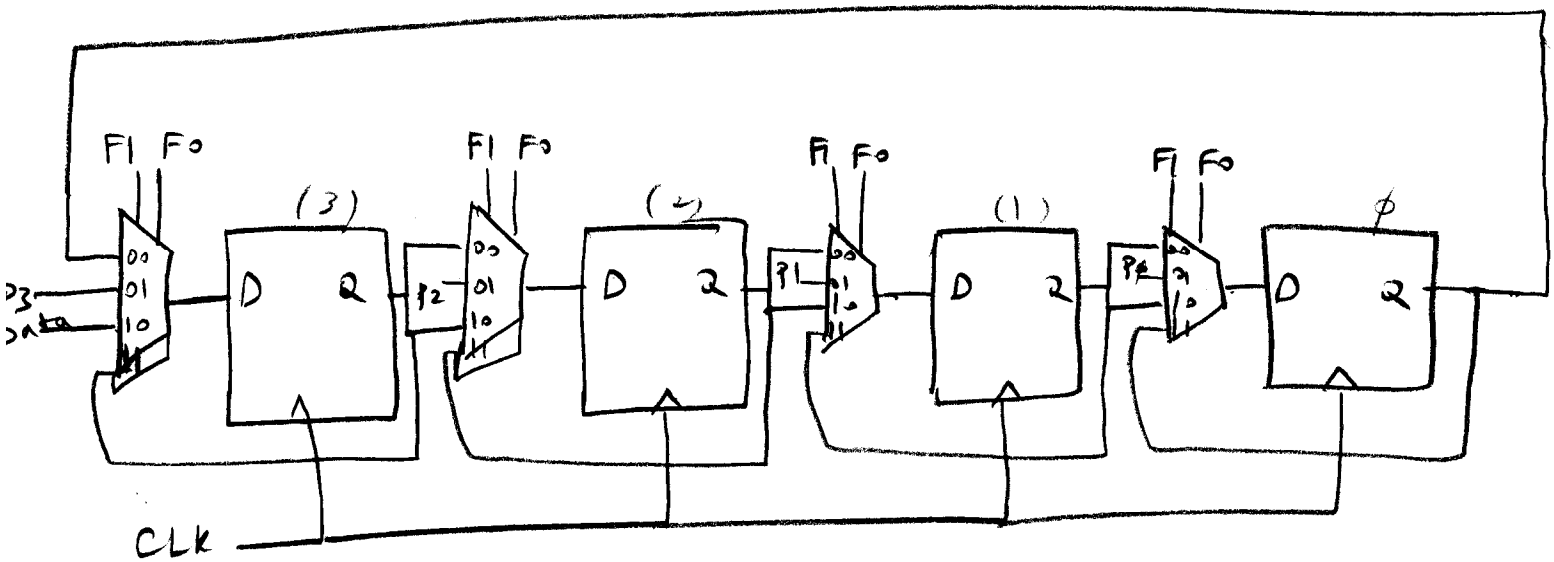
(g.2): hold time:

$$t_h = t_h^{(F1)}.$$

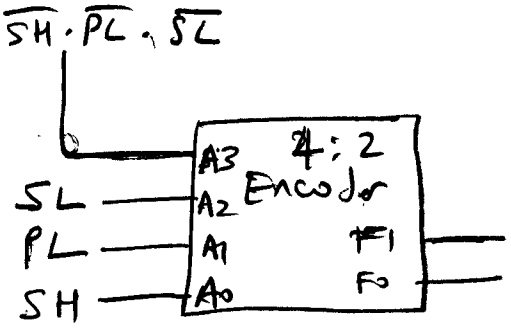
(g.3): CLK-Q delay (Q vector here)

$$t_{CLK-Q} = t_{CLK-Q}^{(F1)} + t_{CLK-Q}^{(F2)}.$$

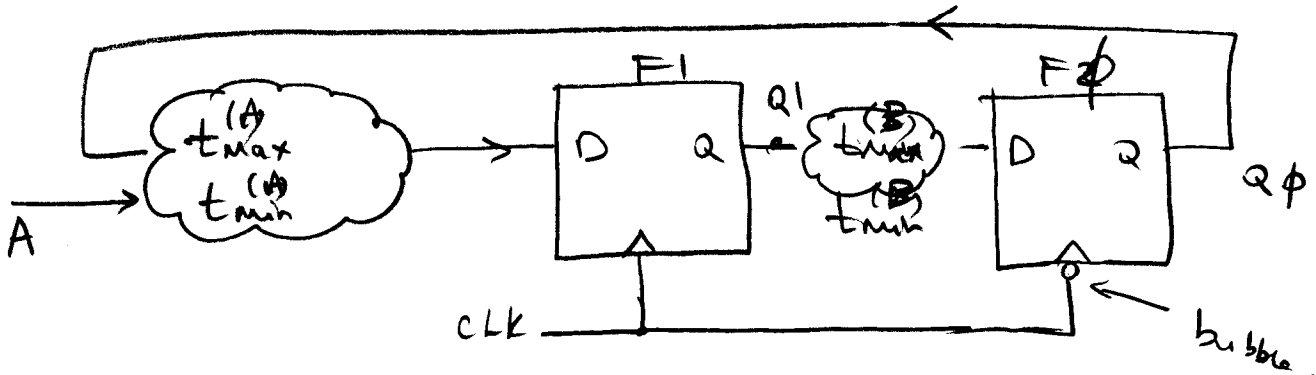
⑥ Implement a 4-bit circular shift register that shifts circularly if SH is asserted, does parallel load if PL asserted, serial load if SL asserted, otherwise holds (Shift circular when serial loading)



SH	PL	SL	F1	F0	behavior
1	0	0	0	0	circular shift ✓
0	1	0	0	1	parallel load
0	0	1	1	0	serial load
0	0	0	1	1	hold



⑦ Timing:



outputs: Q1, Q2

inputs: A.

(a) minimum clock period = ?

Answer:

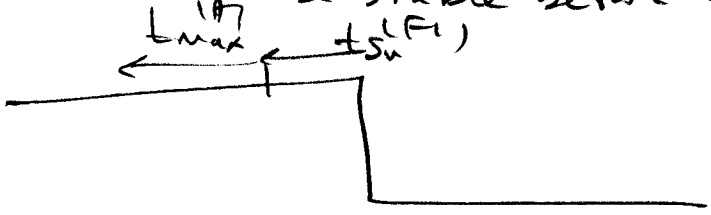
$$T \geq t_{CLK-Q}^{(F1)} + t_{max}^{(B)} + t_{su}^{(F2)} = \alpha$$

$$T \geq t_{CLK-Q}^{(F2)} + t_{max}^{(A)} + t_{su}^{(F1)} = \beta$$

$$\therefore T_{min} = \max\{\alpha, \beta\}$$

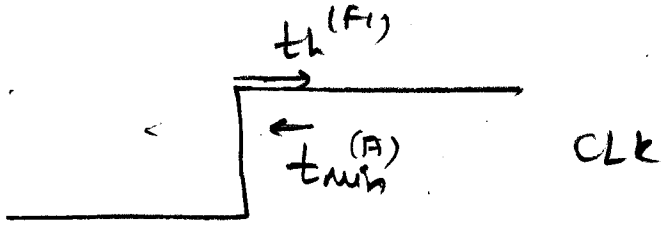
(b) Find the t_{su}^{th} of this circuit when it is considered as a device, with inputs A and CLK.

(b.1) t_{su} = min duration for which A has to be stable before the rising clock edge



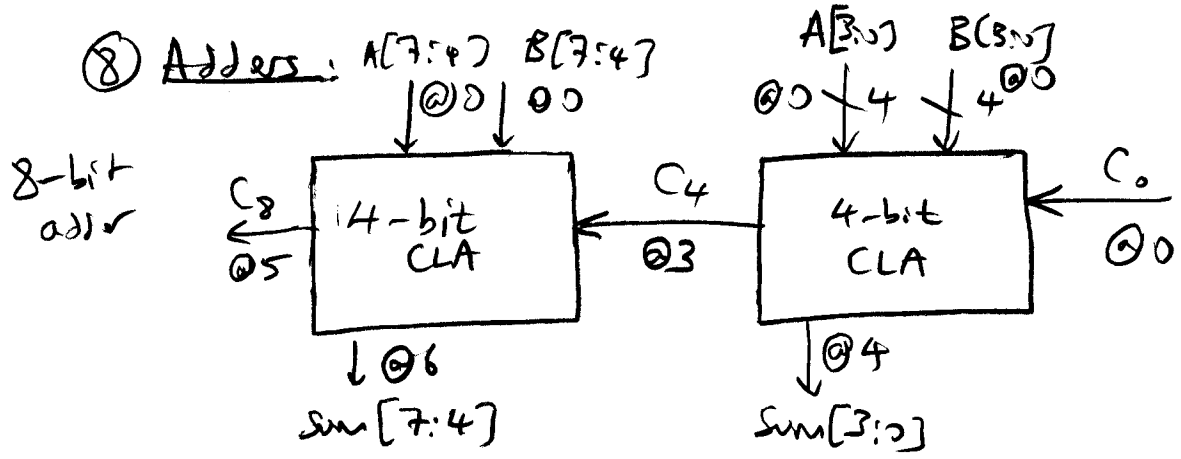
$$t_{su} = t_{max}^{(A)} + t_{su}^{(F1)}$$

(b.2) $t_h = \max\{t_h^{(FF)} - t_{min}^{(A)}, 0\}$.



(b.3) $t_{CLK-Q} = \max\{t_{CLK-Q}^{(FF)}, t_{CLK-Q}^{(F\phi)}\}$.

⑧ Adder:

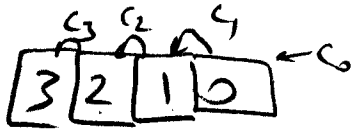


(a) Find worst-case delay:

Inside 4-bit CLA: (first)

$P_i^{@1} = A_i^{@0} \oplus B_i^{@0}$ $\quad 0 \le i \le 3$

$G_i^{@1} = A_i^{@0} \cdot B_i^{@0}$ $\quad 0 \le i \le 3$



$C_1^{@3} = G_0^{@1} + P_0^{@1} C_0^{@0}$

$C_2^{@3} = G_1^{@1} + P_1^{@1} G_0^{@1} + P_1^{@1} P_0^{@1} C_0^{@0}$

$C_3^{@3} = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$

$C_4^{@3} = G_3^{@1} + P_3^{@1} G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_0$

$S_3^{@4} = C_3^{@3} \oplus \underbrace{A[3]^{@0} \oplus B[3]^{@0}}_{@1}$

Note that: Inside second 4-bit CLA:

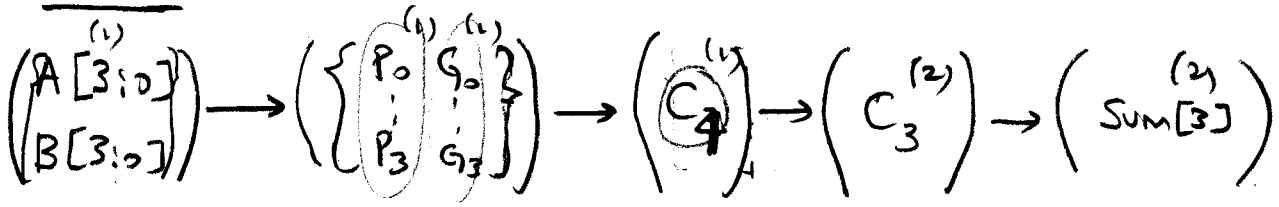
C_i @ $(x+2)$; where x: delay of Carry in. (~~4~~)

S_i @ $(x+3)$

∴ Answer: 6

(b) Find the <u>critical path</u>.

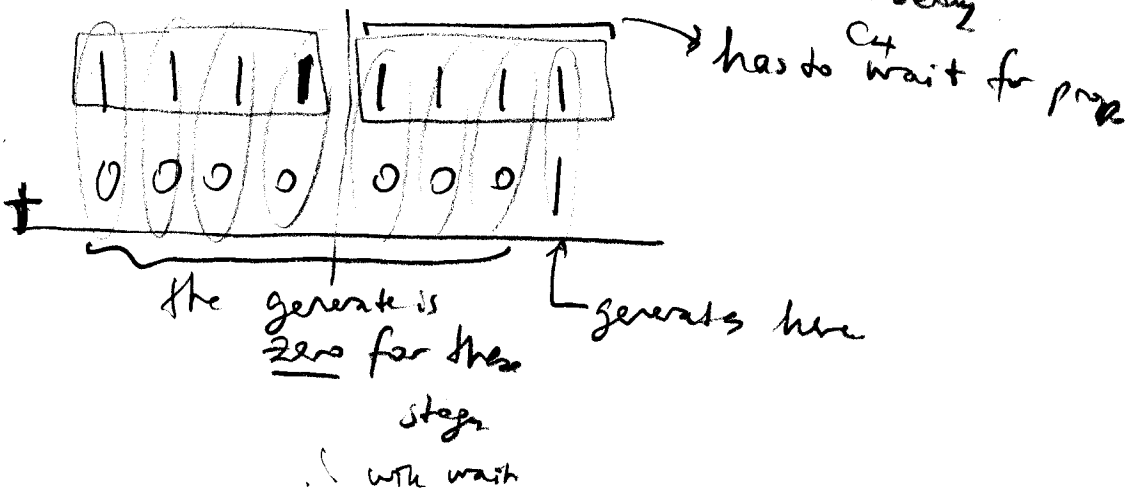
<u>Answer:</u>



(Note that it doesn't start from $C_{0,}$).

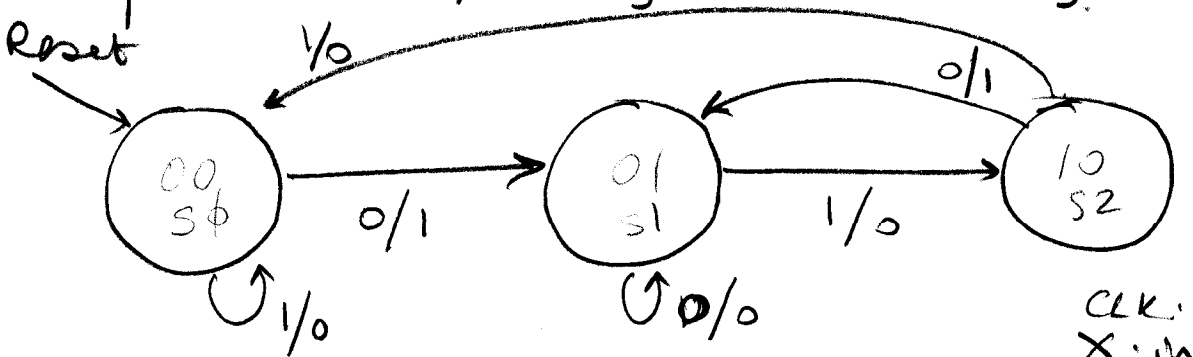
(c) <u>critical transition</u>:

- For adders, usually need to supply both A and B and then wait for the result:
- ~~state it~~ instead of critical transition, the critical inputs stated: that realize the worst-case delay



⑨ FSM & Verilog:

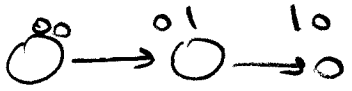
Implement the following machine in Verilog.

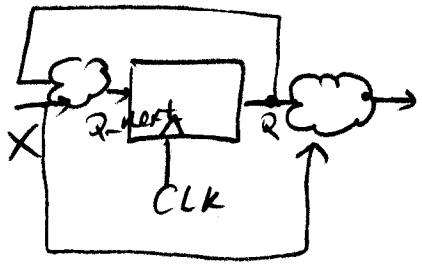


CLK: X: input Y: output

```
Module fsm(X, Y, CLK);
    input X, CLK;
    output Y;
    reg[1:0] Q, Q_next;
parameter S0 = 2'b00, S1 = 2'b01, S2 = 2'b10;
    // flip-flops
    always@(posedge CLK)
        if (Reset)
            Q <= S0;
        else
            Q <= Q_next;
    // next state logic, output logic
    always@(X or Q)
        case({Q,X})
            2'b000: Q_next = S1; Y = 1;
            2'b001: Q_next = S0; Y = 0;
            2'b010: Q_next = S1; Y = 0;
            2'b011: Q_next = S2; Y = 0;
```

State:



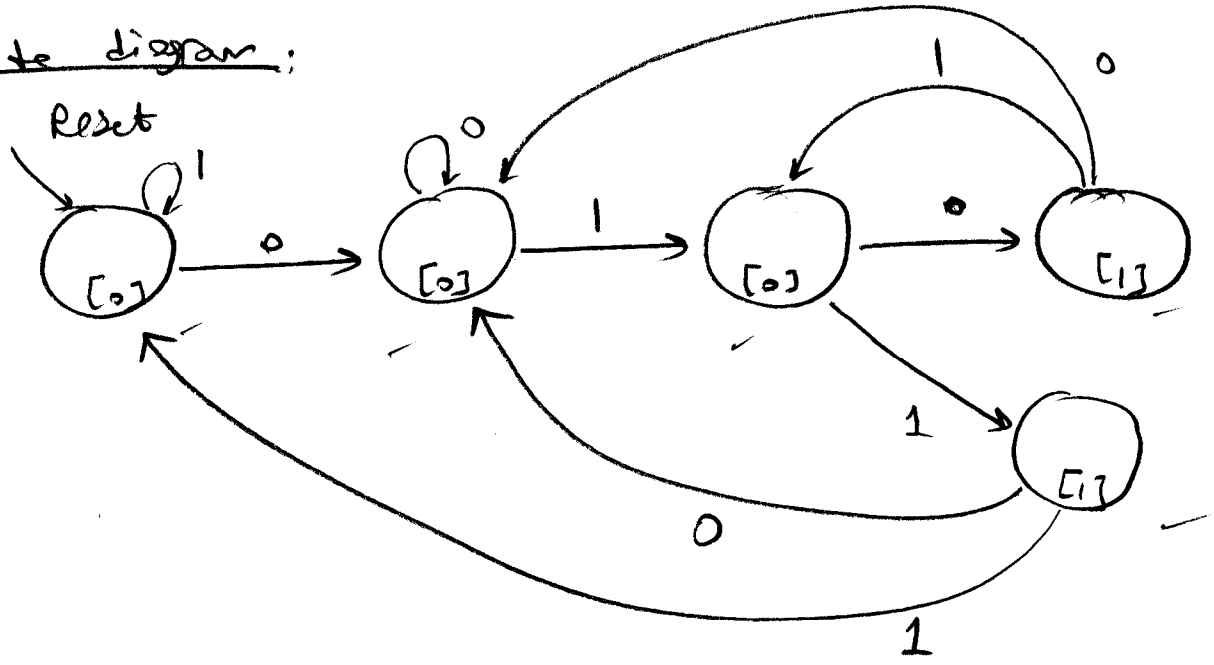


```
        2'b100:   Q_next = s1; Y = 1;
        2'b101:   Q_next = s0; Y = 0;
        default:  Q_next = s0; Y = 0;
    endcase

endModule
```

(b) Design an FSM that recognizes the patterns 010, 011, observing 1 input bit at a time. overlaps allowed

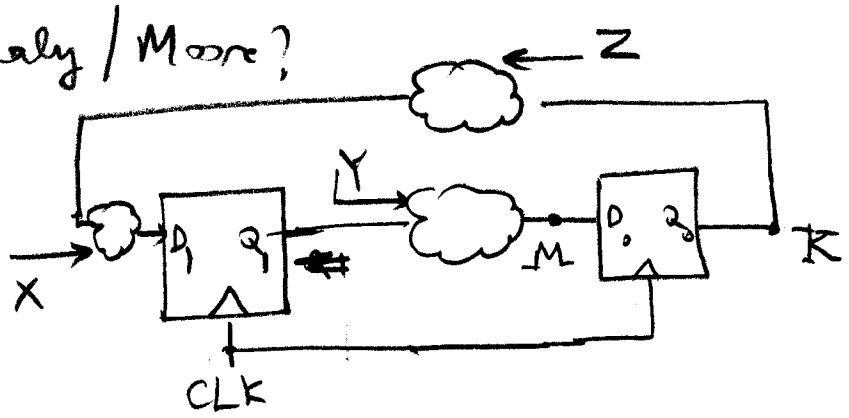
State diagram:



0 1 0

0 1 1

(11) Mealy / Moore?



inputs: X, Y, Z

outputs: M, K

Answer: Mealy because $M = f(Q_1 \text{ and } Y)$