combinational logic

vs

Sequential logic design

Vikas Dhiman for ECE275

October 26, 2022

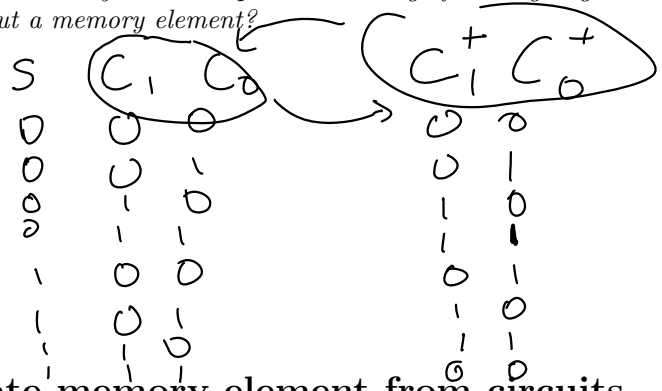
1 Objectives

1. Building blocks of sequential circuits
2. Analyze a sequential circuit and derive a state-table and a state-graph
3. Derive a state graph or state table from a word description of the problem
4. Understanding the structure of an FPGA

next state table state table

2 Why do we need sequential circuits?

Example 1. *Think about this problem: Design an occupancy counter that depends on a sensor S at the class door. The sensor is triggered every time a person passes through the door. The counter can be reset to zero with a reset button. Assume we only need up to two bit counter C_1C_0. Draw a truth table for this circuit. Do you have requisite knowledge for designing this circuit? Can this circuit be designed without a memory element?*



Memory
State
of the
system

State transition table

3 How to create memory element from circuits

Two types of memory

1. Volatile memory. For example, RAM, CPU registers.
2. Non-volatile memory. For example, SSD, Flash drives. (Not covered in this course)

Slowest

Volatile memory

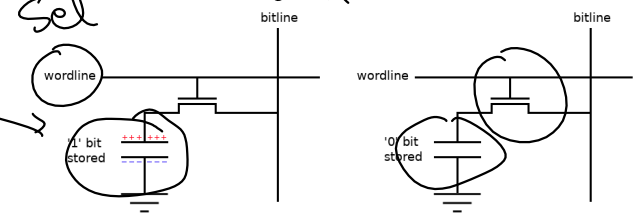
(a) Memories that require periodic refreshing. For example, DRAM: Dynamic Random Access memory (Not covered in this course)

Memory refreshing

Sel

Data

capacitor



DRAM

slightly faster

[1]

(b) Memories that are always refreshing. For example, SRAM: Static Random Access memory [2, Appendix B.64]

CPUs

L1-cache

register

fastest (power hungry)

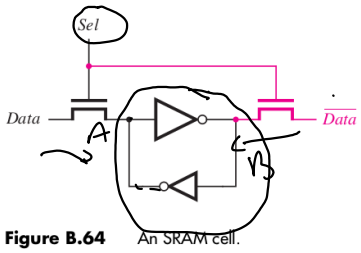
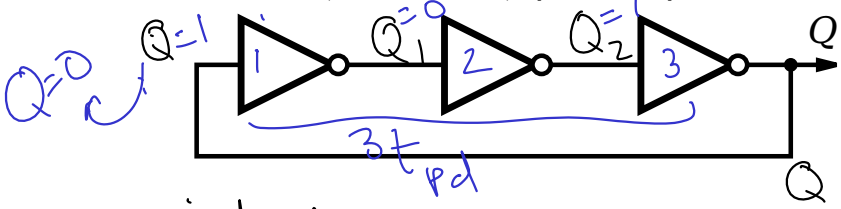
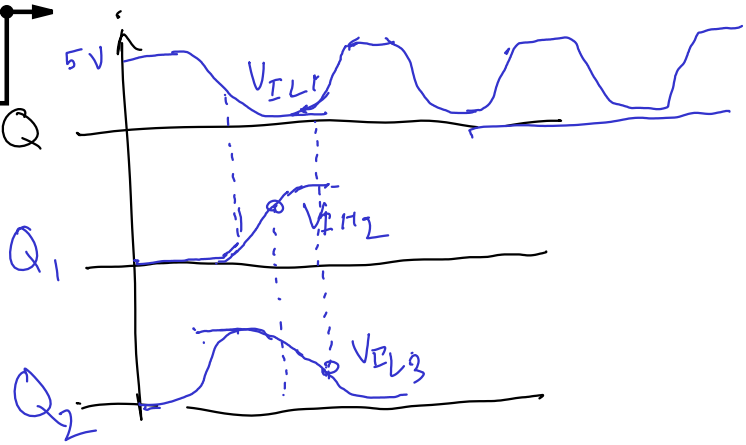


Figure B.64 An SRAM cell.

4 Latches and Flip-Flops [1, Sec 3.2]

Example 2 (Ring oscillator). *[1, Sec 3.31] How many stable states does the following circuit have?*





non-ideal

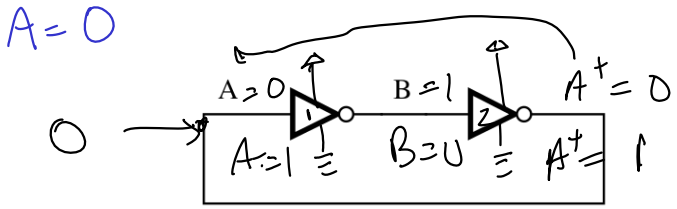
Odd number of NOT gates in a loop you get an oscillator

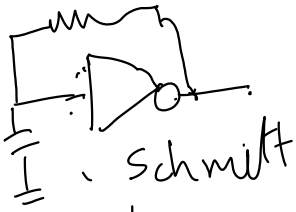
Definition 1 (Astable circuits).

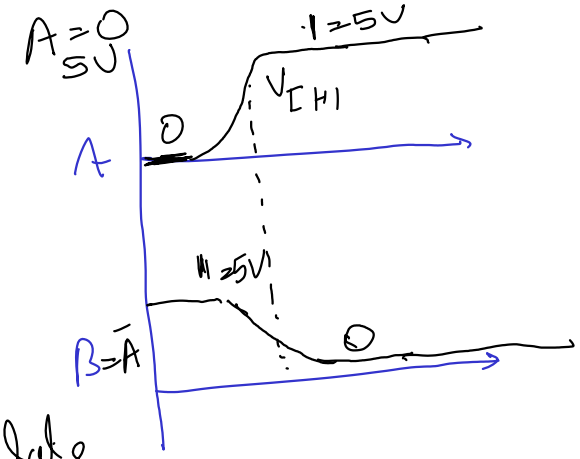
Unstable = no stable states

Example 3. *Analyze the timing diagram of the following circuit.*

[1] Image source: `allaboutcircuits.com/technical-articles/introduction-to-dram-dynamic-random-access-memory/`



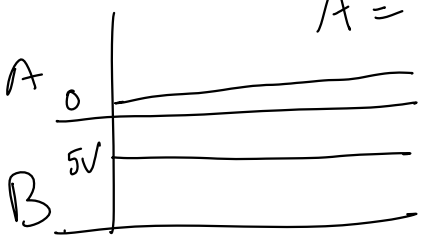




1-bit

Definition 2 (Bistable circuits).

Two stable state



Definition 3 (Characteristic or state table). *Draw the characteristic or state table of the above circuit.*

A	B	A^+
0	1	0
1	0	1

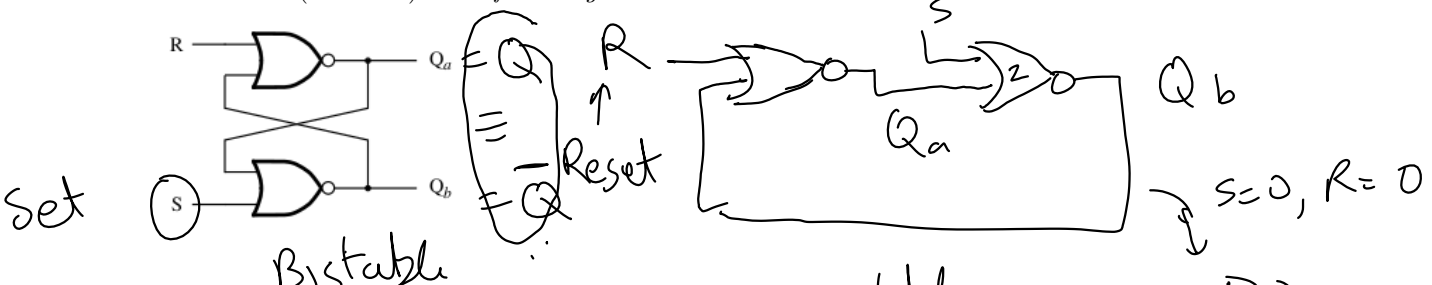
$A^+ = A(t + t_{pd})$

$A = A(t)$

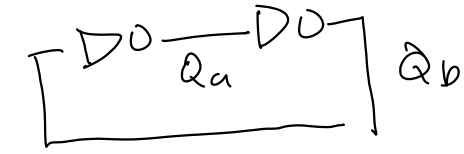
4.1 SR (Set-Reset) latch [1, Sec 3.2.1]

Definition 4 (SR latch). *The following circuit is called the SR latch.*

S=0
R=0

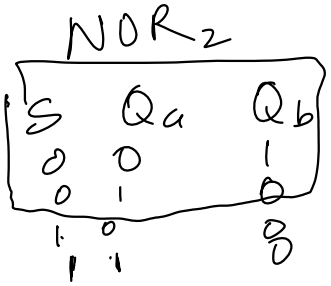


1. *How many stable states does this circuit have?* two stable state
2. *Draw its characteristic or state table.*
3. *Draw SR latch symbol*



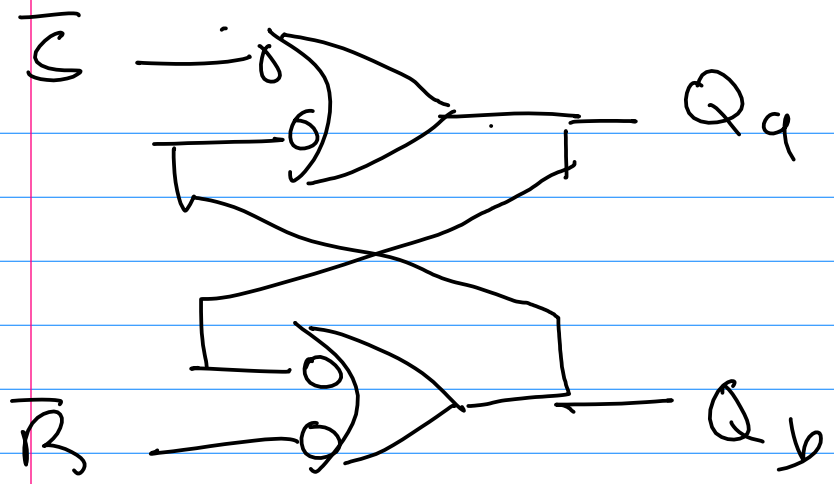
State table

S	R	Q_a	Q_b	Q_a^+	Q_b^+
Hold → 0	0	Q_a	$\bar{Q}_a$	Q_a	$\bar{Q}_a$
Reset → 0	1	d	d	0	1
Set → 1	0	d	d	1	0
Invalid → 1	1	d	d	0	0

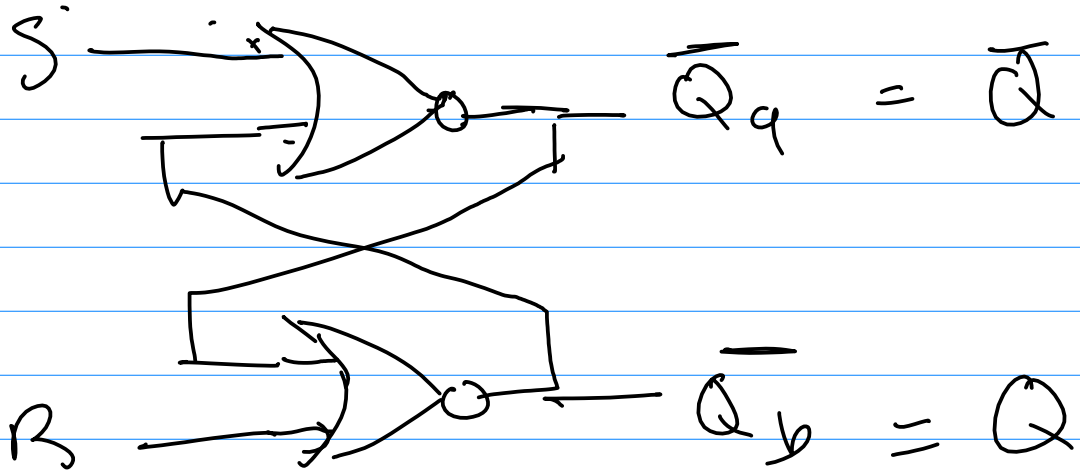


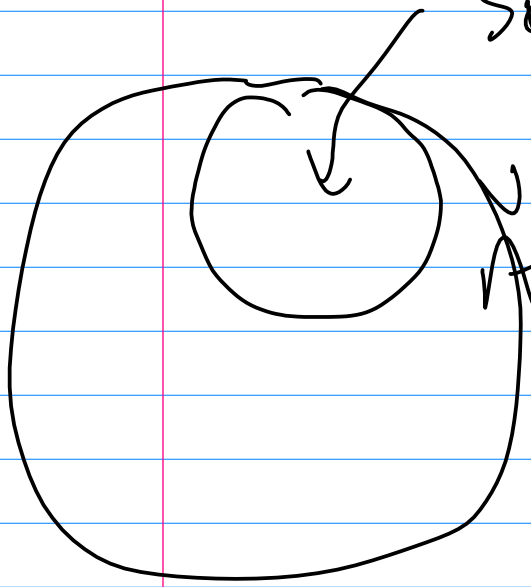
NOR1

Q_b	R	Q_a
0	0	1
0	1	0
1	0	0
1	1	0



SR Latch

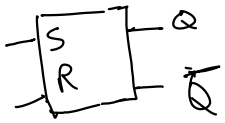




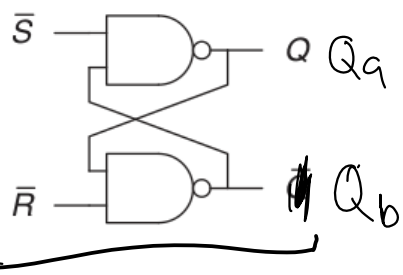
Synchronous (avoid glitches at clocked intervals)

vs

Asynchronous



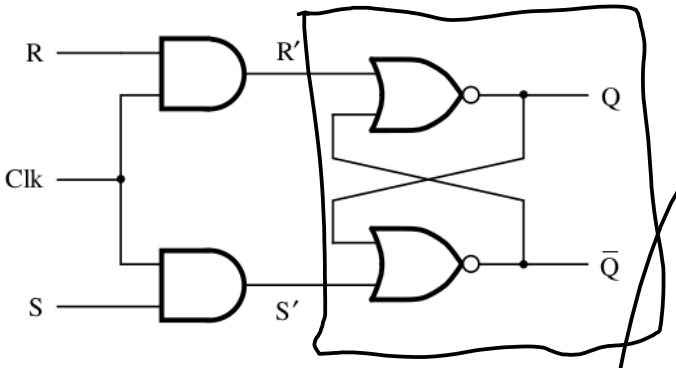
Problem 1 (SR latch using NAND gates)**.** *Draw the characteristic or state table for the following circuit*



S	R	Q_a	Q_b	Q_a^+	$\bar{Q}_b$
0	0				
0	1				
1	0				
1	1				

4.2 Gated SR latch [2, Sec 5.2]

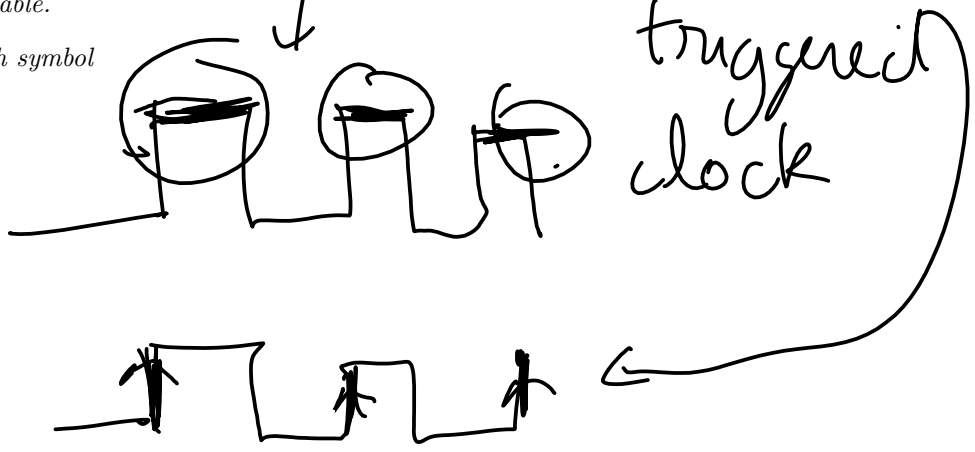
Definition 5 (Gated SR latch)**.** *The following circuit is called the Gated SR latch.*

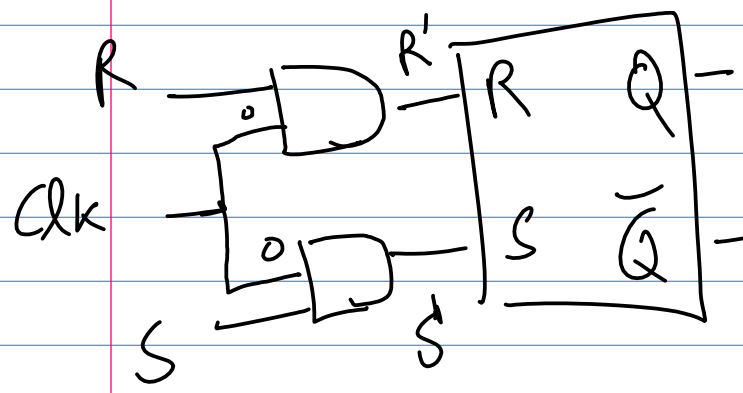


1. *Draw its characteristic table.*
2. *Draw the Gated SR latch symbol*

Level triggered clock

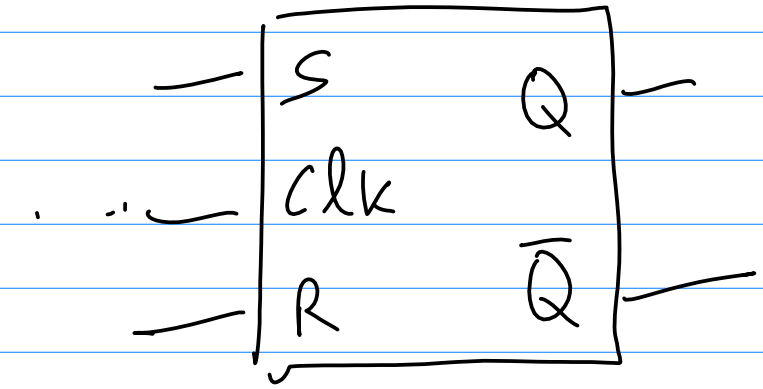
Later: edge triggered clock





	S	R	Q^+
Hold	0	0	Q
Reset	0	1	0
set	1	0	1
Invalid	1	1	$Q=\bar{Q}=0$

	S	R	Clk	S'	R'	Q^+
Hold	x	x	0	0	0	Q
Hold	0	0	1	0	0	Q
Reset	0	1	1	0	1	0
set	1	0	1	1	0	1
Invalid	1	1	1	1	1	$Q=\bar{Q}=0$

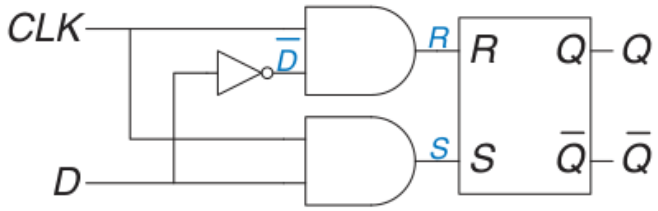


Gated SR latch

. Transparent latch → Flip flop

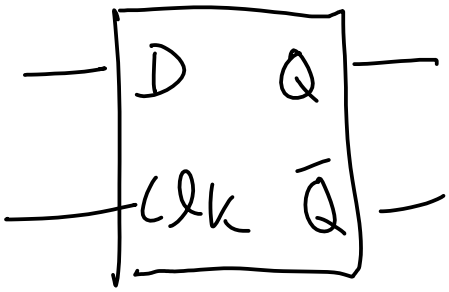
4.3 D (Data) latch [1, Sec 3.2.2]

Definition 6 (D latch)**.** *The following circuit is called the D latch.*



1. *Draw its characteristic table.*
2. *Draw the D latch symbol*

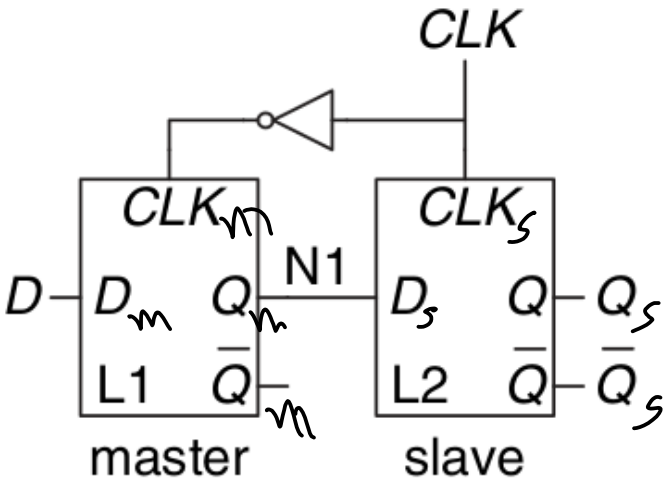
	D	$\bar{D}$	Clk	S	R	Q^+
Hold	x	x	0	0	0	Q
Reset	0	1	1	0	1	0
set	1	0	1	1	0	1



D	Clk	Q^+
x	0	Q
D	1	D

4.4 D flip-flop [1, Sec 3.2.2]

Definition 7 (D flip-flop)**.** *The following circuit is called the D flip-flop.*



1. *Draw its timing diagram*
2. *Draw its characteristic table.*
3. *Draw the D flip-flop symbol*

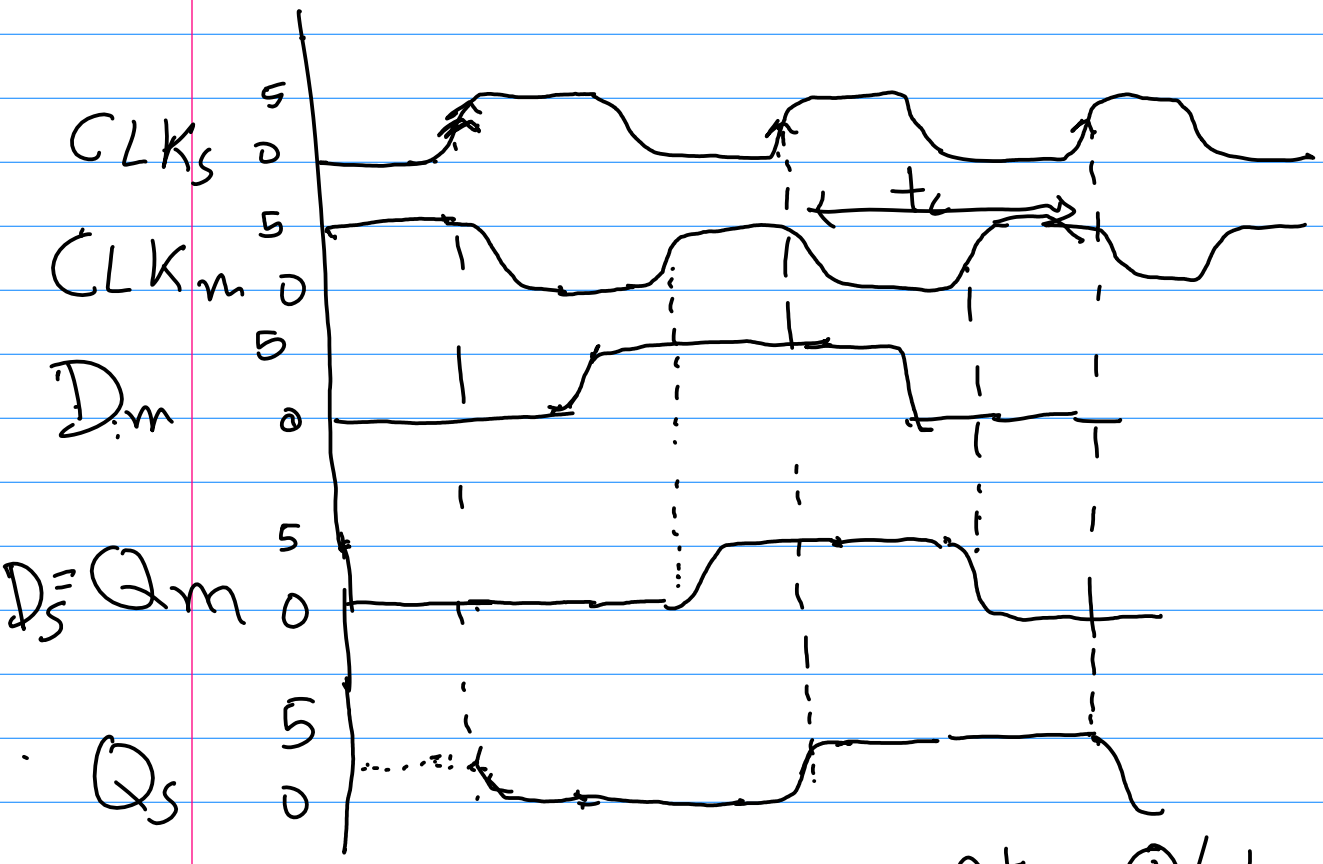
Remark 1. *What is the difference between a latch and a flip-flop?*

latches vs flip flops
level triggered on clock | edge triggered

Example 4. *Add a* RESET *signal to the D flip-flop that resets the state of flip-flop to 0.*

Example 5. *The toggle (T) flip-flop has one input, CLK, and one output, Q. On each rising edge of CLK, Q toggles to the complement of its previous value. Draw a schematic for a T flip-flop using a D flip-flop and an inverter.*

Def 7 D-flip flop

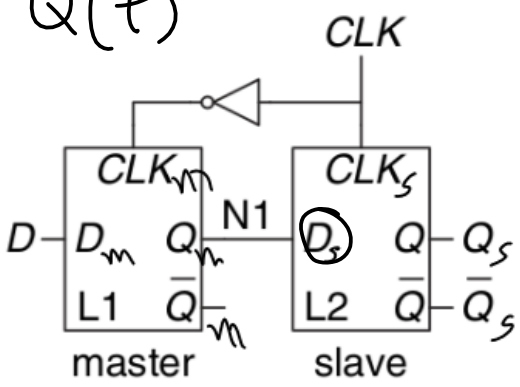


$Q^+ = Q(t + t_{pd})$.

$Q = Q(t)$

$CLK = 0 \quad Q^+ = Q$

D	CLK ↑	Q^+
D	↑	D
*	*	Q



clock period

Synchronous circuit

$Q^+ = Q(t + t_c)$

State table

D flip flop

D	Q^+
0	0
1	1

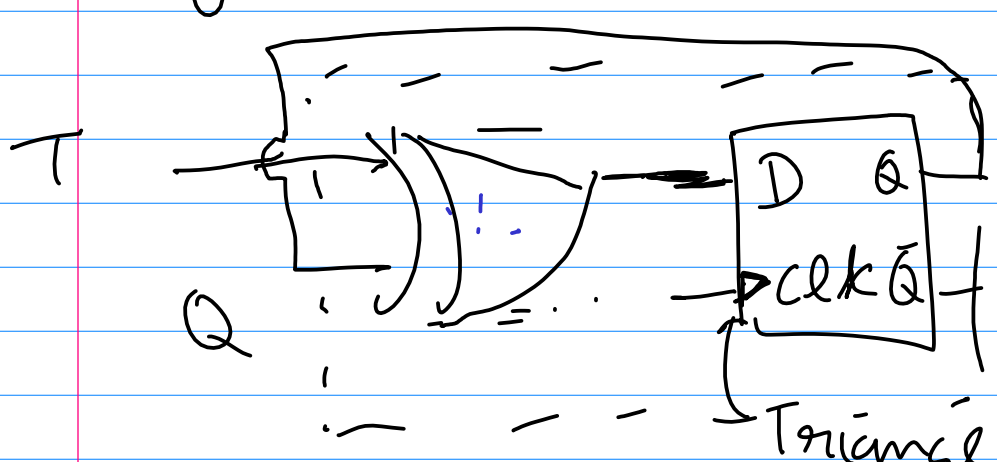
T-flip flop

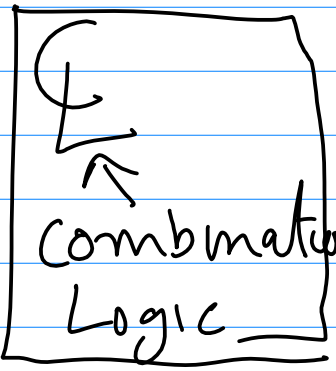
↑

Toggle

T	Q^+
0	Q
1	$\bar{Q}$

How can we construct T-flip flop from D-flip flop?





Triangle denotes edge triggered

T	Q	D	Q^+
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

XOR

(2) Excitation table
(3) Truth table

Q	Q^+	D
0	0	0
1	1	1

J	K	Q	D
			?

Problem 2. *A JK flip-flop receives a clock and two inputs, J and K. On the rising edge of the clock, it updates the output, Q. If J and K are both 0, Q retains its old value. If only J is 1, Q becomes 1. If only K is 1, Q becomes 0. If both J and K are 1, Q becomes the opposite of its present state.*

1. *Construct a JK flip-flop using a D flip-flop and some combinational logic.*
2. *Construct a D flip-flop using a JK flip-flop and some combinational logic.*
3. *Construct a T flip-flop (see Exercise 3.9) using a JK flip-flop.*

(1) input → present states, next states

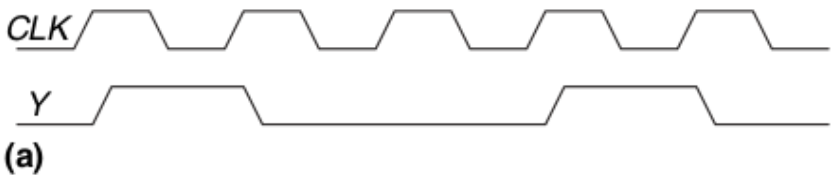
J	K	Q	Q^+

5 Finite State Machines [1, Sec 3.4]

[2]

Example 6. *Design an occupancy counter that depends on a sensor S at the class door. The sensor is triggered every time a person passes through the door. Assume that the counter starts at zero. Assume we only need up to two bit counter C_1C_0. Draw a state table for this circuit.*

Problem 3. *A divide-by-N counter has one output and no inputs. The output Y is HIGH for one clock cycle out of every N. In other words, the output divides the frequency of the clock by N. The waveform for a divide-by-3 counter is shown here:*



Sketch circuit designs for such a counter

Problem 4. *Design a 3-bit counter which counts in the sequence: 001, 011, 010, 110, 111, 100, (repeat) 001, ...*

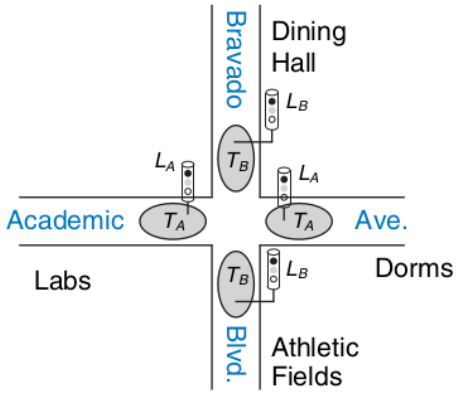
Example 7. *Design an odd-even counter for an single bit input. The output of this circuit should be 1 if the number of 1s to the input have been odd so far and 0 otherwise.*

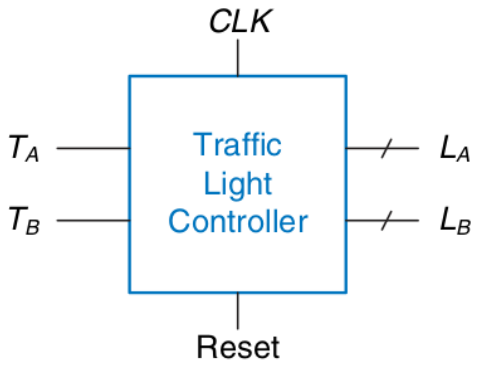
Example 8 (Sequence detectors)**.** *A sequential circuit has one input and one output. The output becomes 1 and remain 1 thereafter when at least two 0's and at least two 1's have occurred as inputs regardless of the order of*

Example 9. *Consider the problem of inventing a controller for a traffic light at a busy intersection on campus. There are two traffic sensors, T_A and T_B , on Academic Ave. and Bravado Blvd., respectively. Each sensor indicates TRUE if students are present and FALSE if the street is empty. There are two traffic lights, L_A and L_B, to control traffic. Each light receives digital inputs specifying whether it should be green, yellow, or red. When the system is reset, the lights are green on Academic Ave. and red on Bravado Blvd. As long as traffic is present on Academic Ave., the lights do not change. When there is no longer traffic on Academic Ave., the light on Academic Ave. becomes yellow for 5 seconds before it turns red and Bravado Blvd.'s light turns green. Similarly, the Bravado Blvd. light remains green as long as traffic is present on the boulevard, then turns*

[2] These notes will not fit on your note sheet.

yellow and eventually red.





1. *Draw a state transition diagram*
2. *Draw a state table*
3. *Assign binary encodings to each of the states*
4. *Redraw the state table with binary encodings. Design a minimal SOP boolean expression.*
5. *Assign binary encodings to each of the output and redraw the output table. Design a minimal SOP boolean expression for the outputs.*

Problem 5. *Design a circuit for a 2x2 pixel resolution pong game, where the ball can only occupy 4 possible pixels and a single paddle occupies another 2 pixels. The ball bounces of the paddle when the paddle is in the correct row. To keep it interesting, the ball takes a different path from the source path. Track the score with a single bit counter.*

References

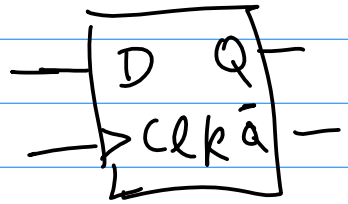
[1] Sarah L Harris and David Harris. *Digital design and computer architecture*. Morgan Kaufmann, 2022.

[2] Brown Stephen and Vranesic Zvonko. *Fundamentals of digital Logic with Verilog design*. McGraw Hill, 2022.

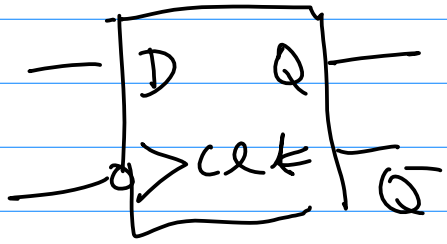
Example 6 : occupancy counter

D - flip flop

+ve edge triggered



-ve edge triggered



Characteristic table of D flip flop?

Clk	D	Q	Q^+
↑	0	d	0
	1	$\bar{d}$	1
⊓	d	0	0
		1	1

$Q^+ = Q(t+t_c)$

$Q = Q(t)$

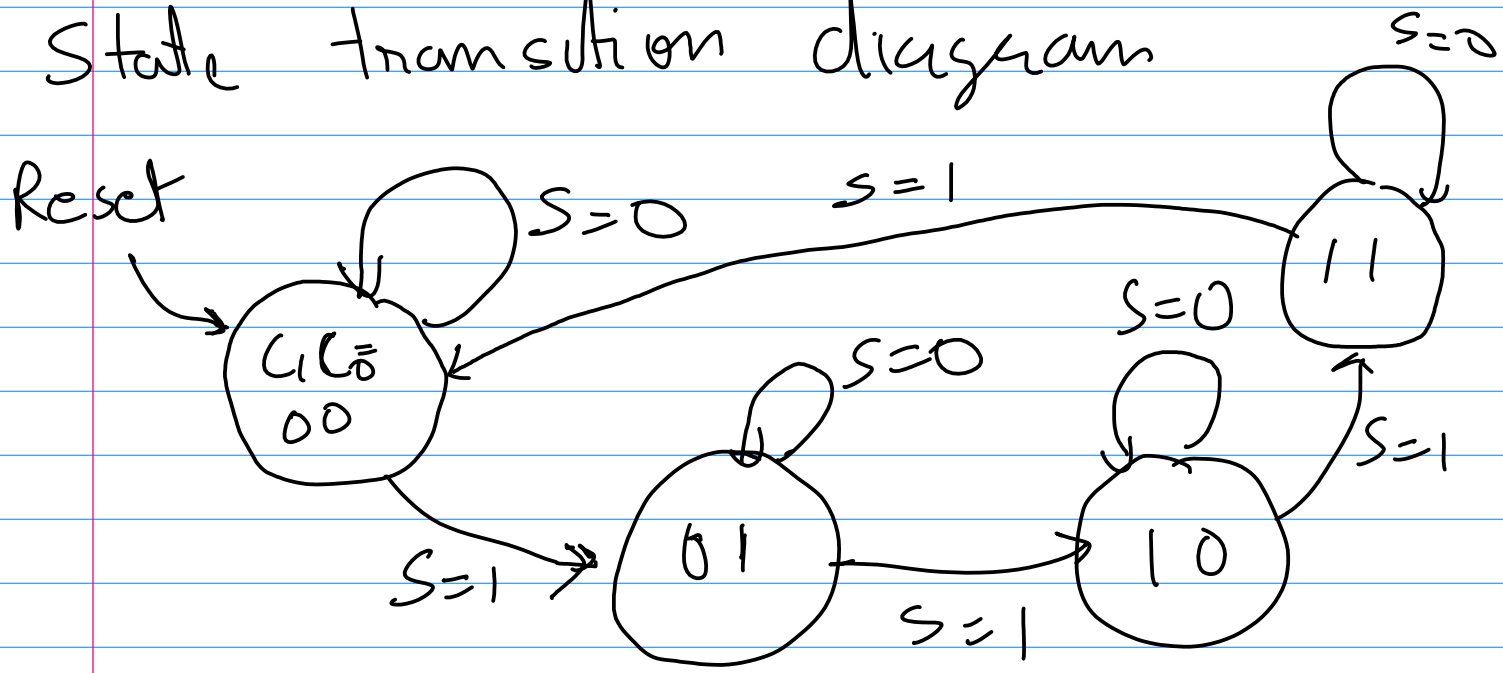
D	Q^+
0	0
1	1

characteristic table

State transition table

Sensor	State C_1	State C_0	Next State C_1^+	Next State C_0^+	
0	0	0	0	0	counter stay same
0	0	1	0	1	
0	1	0	1	0	
0	1	1	1	1	
1	0	0	0	1	Increment the counter
1	0	1	1	0	
1	1	0	1	1	
1	1	1	0	0	

State transition diagram

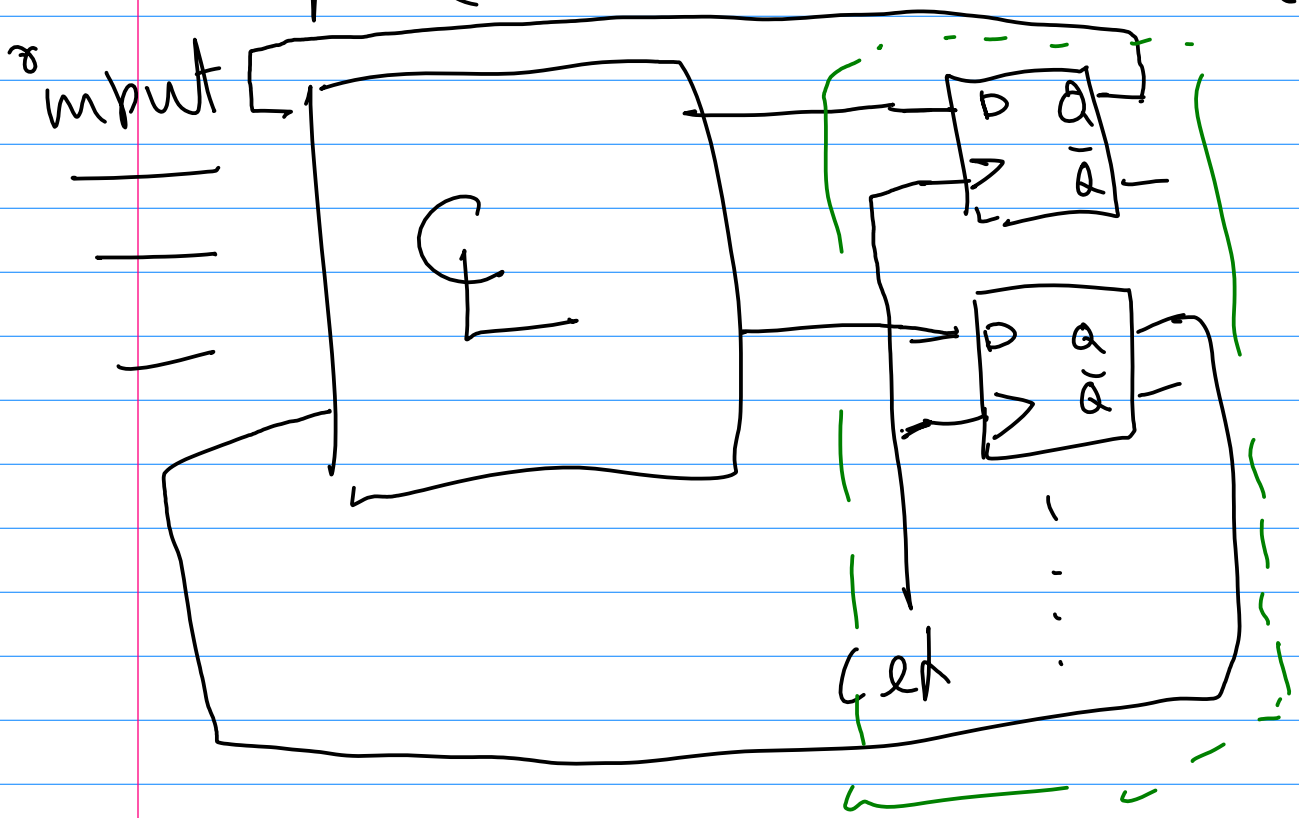


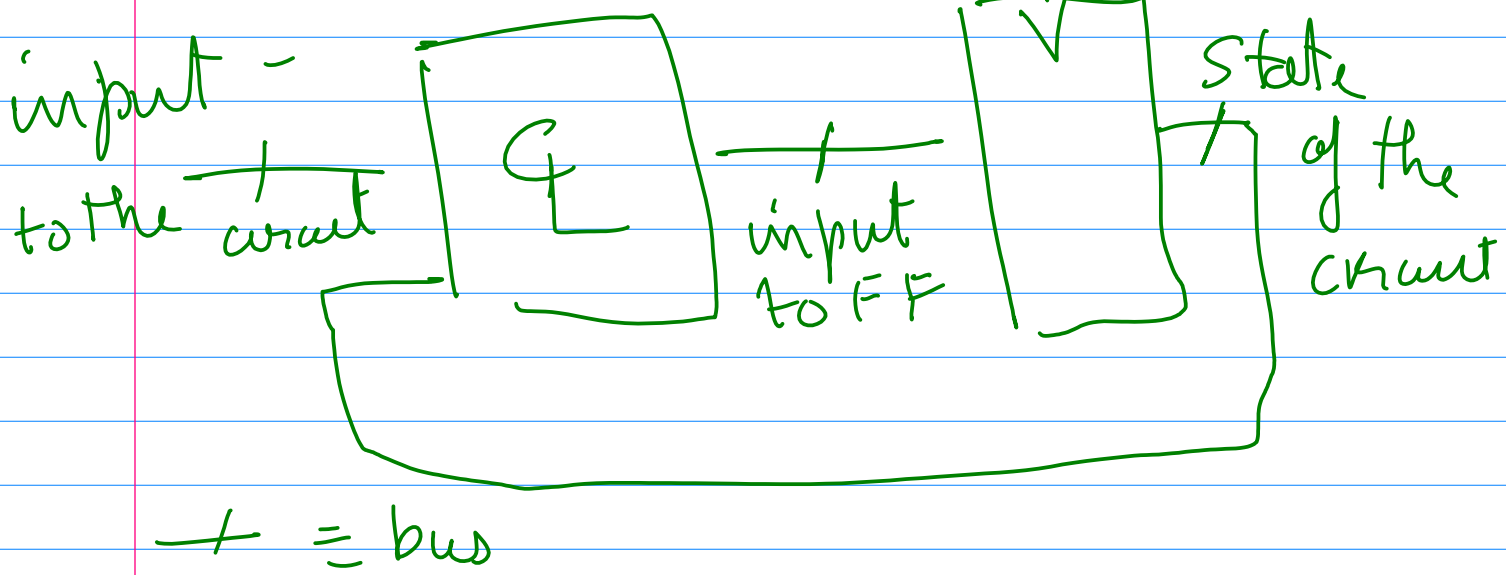
① Identify what states will you need

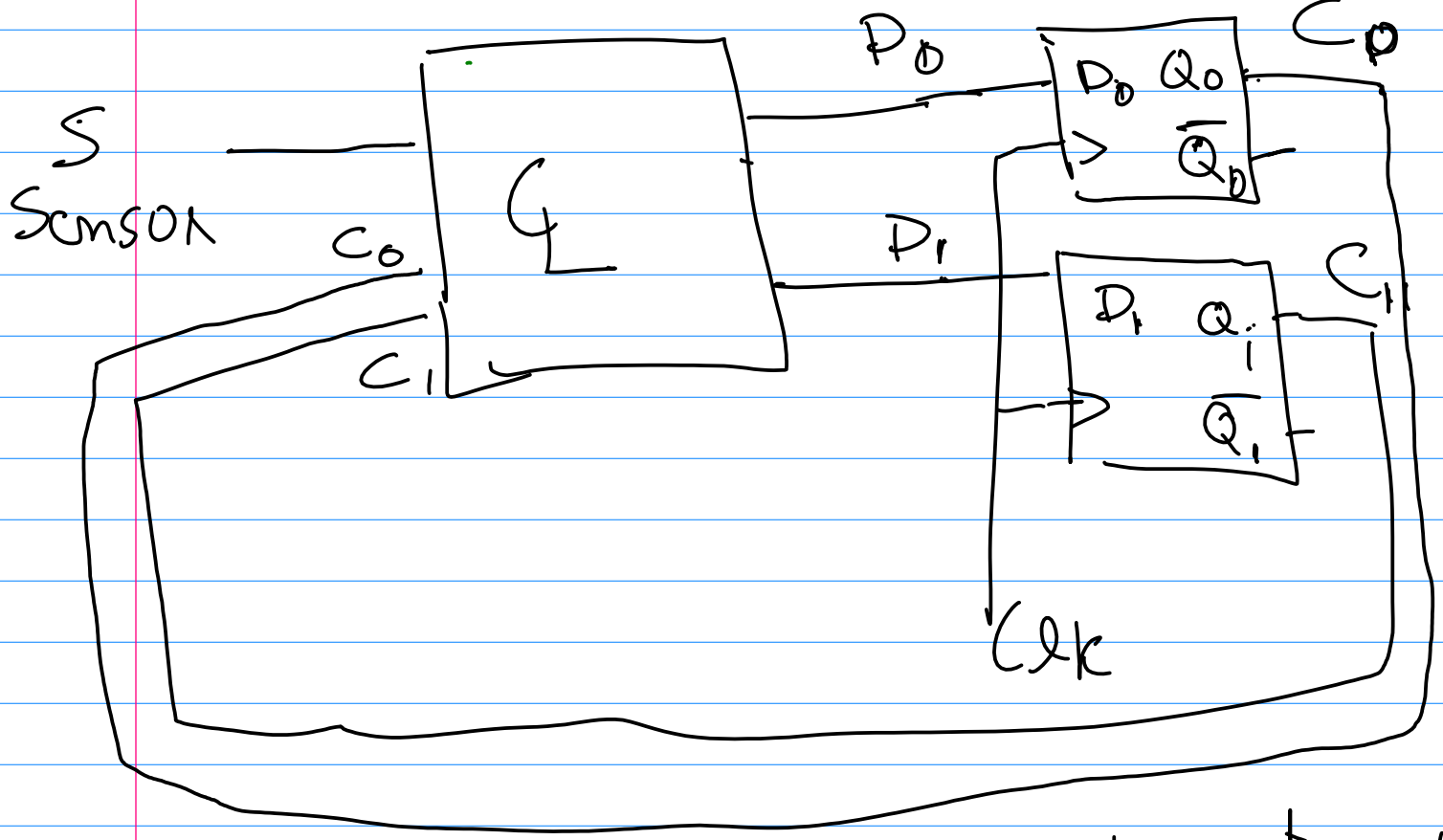
② How are you transitioning b/w the states

How to design a circuit from state transition table?

Synchronous sequential circuit template







characteristic table

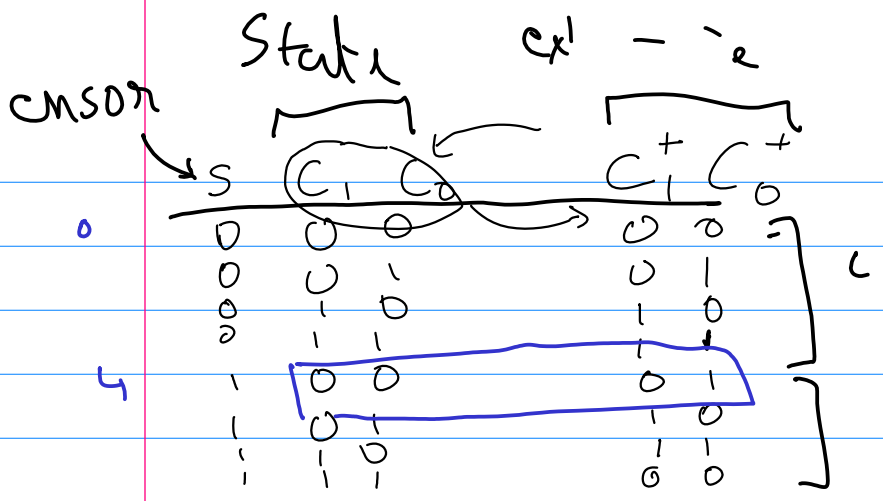
S	C_1	C_0	D_1	D_0
			?	?

C_1	C_0	C_1^+	C_0^+
0	0	0	1

D_0	Q_0	Q_0^+
0	d	0
1	d	1

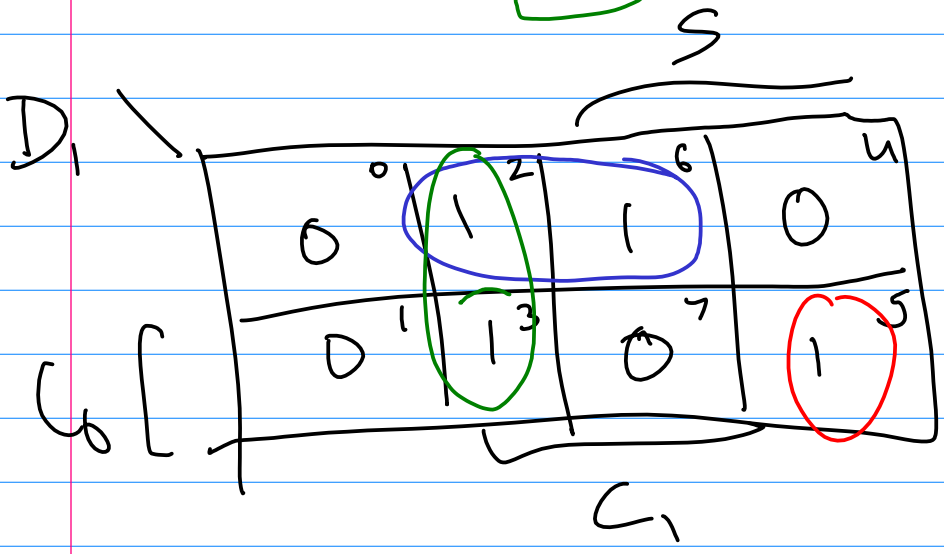
↓ Excitation table

Q	Q^+	D
0	0	0
0	1	1
1	0	0
1	1	1



Truth table

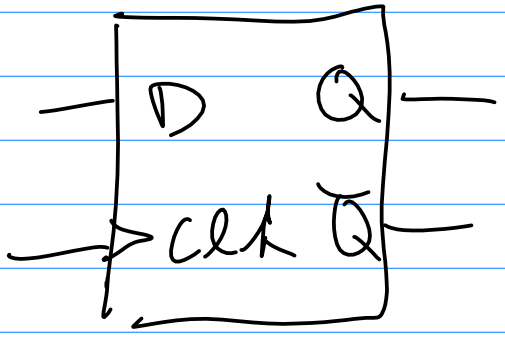
Q_1	Q_0			
S.	$=C_1$	$=C_0$	D_1	D_0
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	1	1
1	0	0	0	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

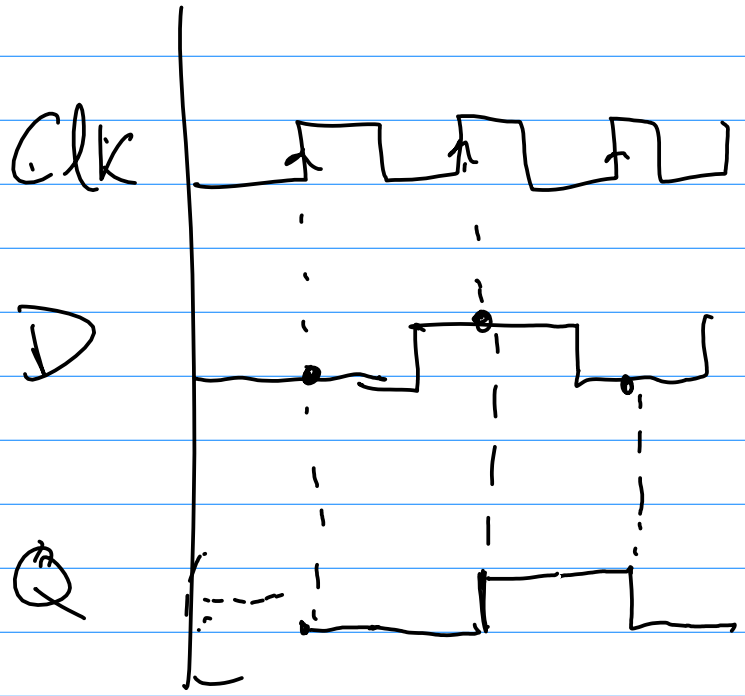


$D_1 = C_1\bar{S} + C_1\bar{C_0} + S\bar{C_0}C_1$

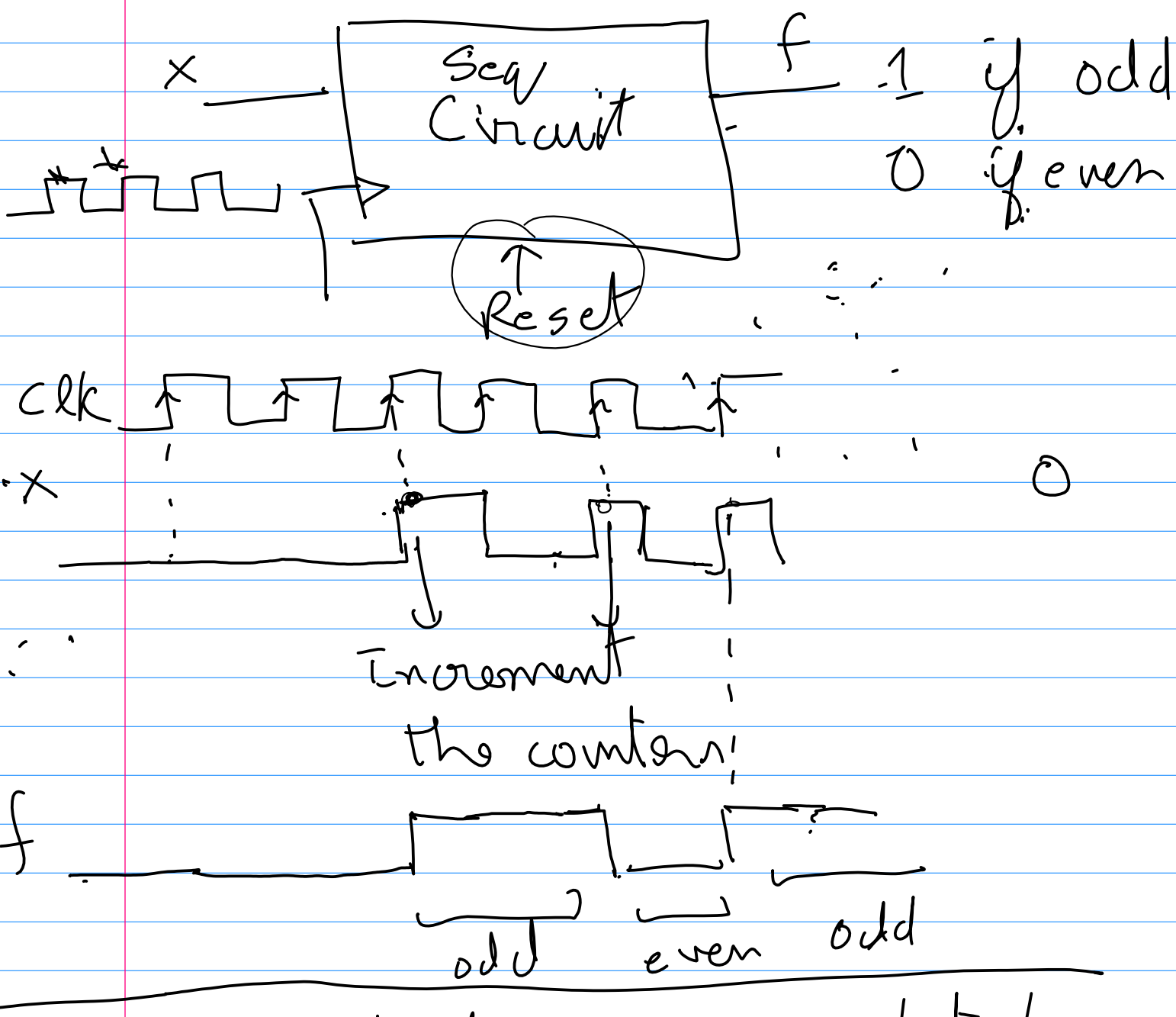
D- flip flop

(Data)





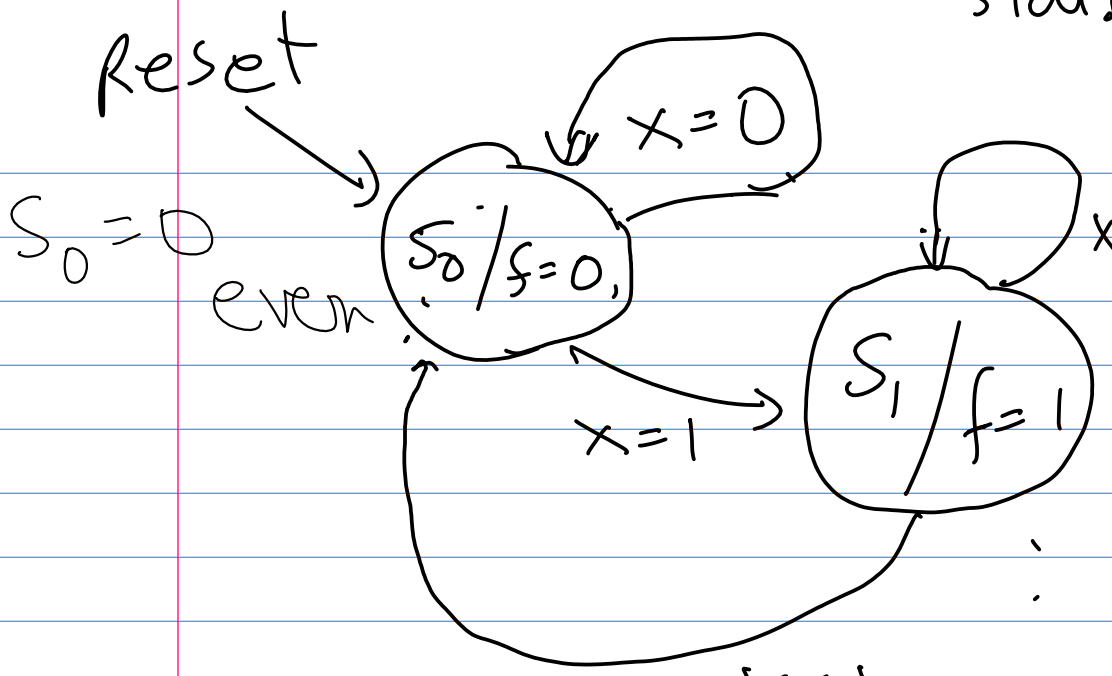
Ex 7 Odd-even counter



① Does it even need state/memory?

② How many bits of memory? = 1

State diagram



S_0 = 0 inputs

S_1 = 1 input

S_2 = 2 input

S_0 = even # input

S_1 = odd # inputs

2 → 1 bit

4 → 2 bits

8 → 3 bits

#

next table state transition

→ state $0 = S_0$, $1 = S_1$

X	S	S^+
0	0	0
0	1	1
1	0	1
1	1	0

$ff = S^+$ → Input to D-flipflop

D- flip flop

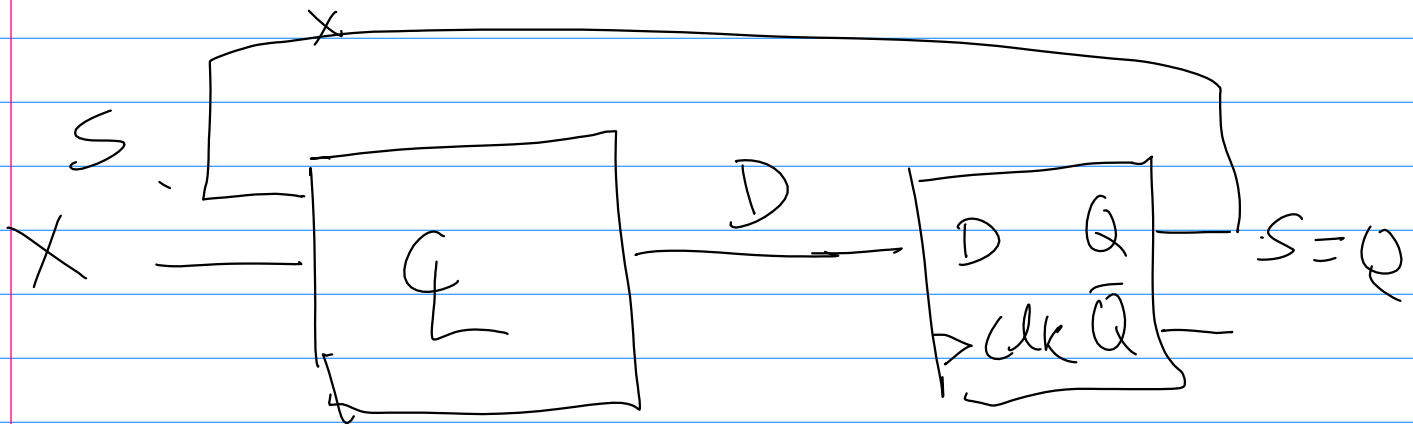
D	Q	Q^+
0	*	0
1	*	1

Excitation table

inversion

Q	Q^+	D
0	0	0
0	1	1
1	0	0
1	1	1

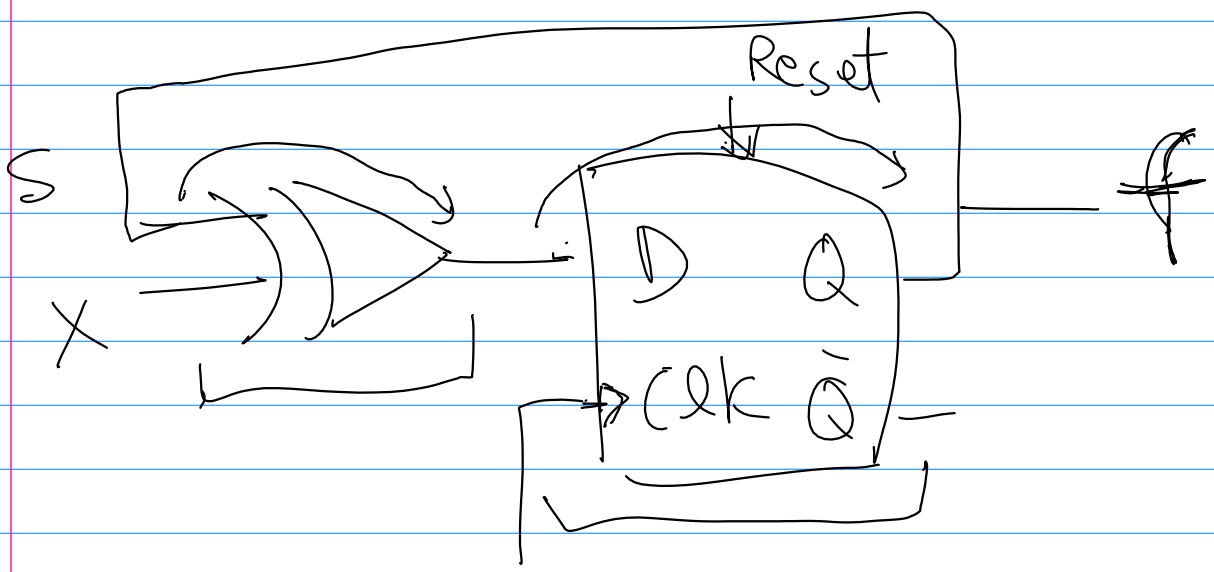
X	S	S^+	D
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

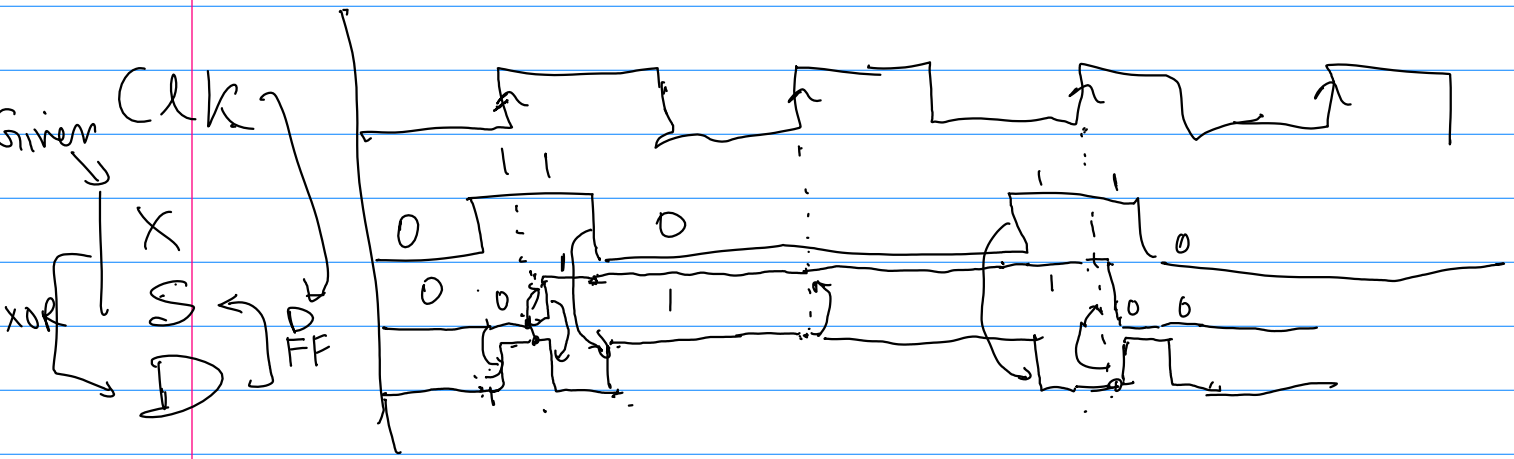


X	S	D
0	0	0
0	1	1
1	0	1
1	1	0

Named gate

XOR gate





① next state table

J	K	Q	Q^+

② Use excitation table of the desired ff

③ Find the truth table for inputs to the ff

① Next state table :-

Set & Reset

J	K	Q	Q^+	
0	0	0	0	Hold
0	0	1	1	
0	1	0	0	Reset
0	1	1	0	
1	0	0	1	Set
1	0	1	1	
1	1	0	1	Toggle
1	1	1	0	Toggle

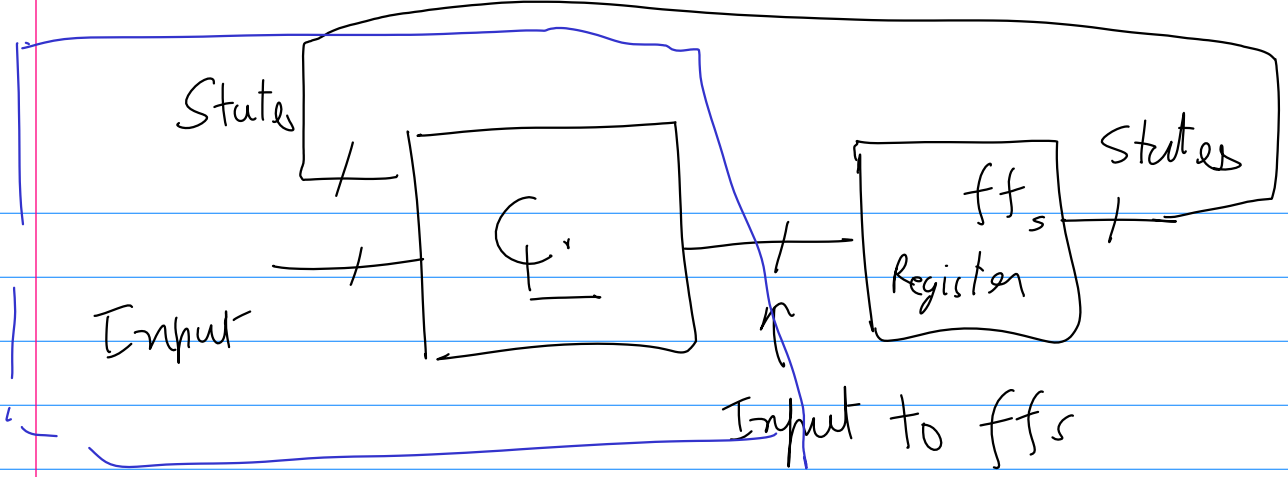
② D - ff

Q	Q^+	D
d	0	0
d	1	1

③ Truth table

J	K	Q	D
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

D



D - ff

T - ff

JK - ff