Chapter 7

Sequential Logic

7.1 Objectives

1. Understand timing diagrams, gate delays and critical path
2. Design Hazard-free two level circuits
3. Building blocks of sequential circuits
4. Analyze a sequential circuit and derive a state-table and a state-graph
5. Derive a state graph or state table from a word description of the problem
6. Understanding the structure of an FPGA

7.2 Why do we need sequential circuits?

Example 7.1. *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?*

state at time t

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

at time t+1

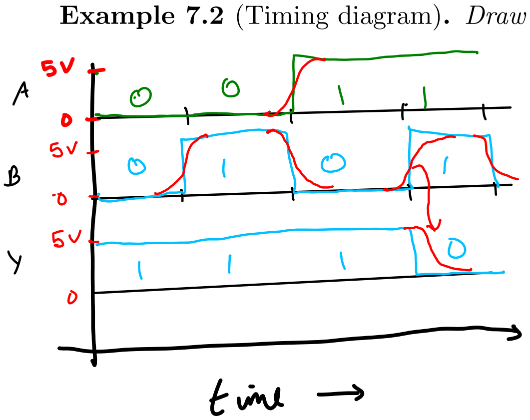
$C_1(t+1) = C_1^+$

state/memory

S	C_1	C_0	C_1^+	C_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

7.3 Timing diagrams and propagation delays

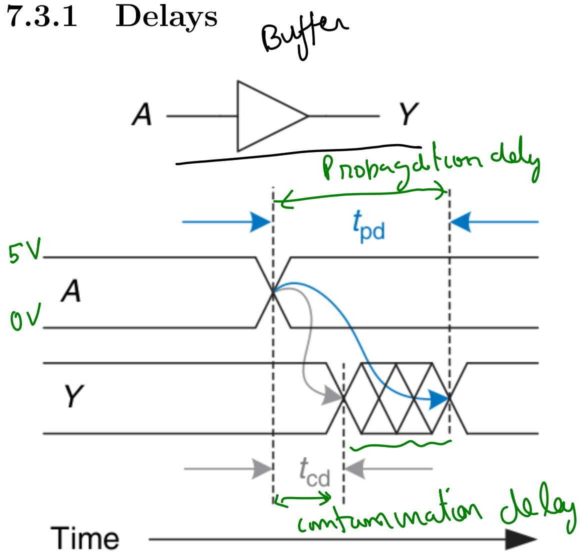
Example 7.2 (Timing diagram)**.** *Draw a timing diagram for an ideal NAND gate.*

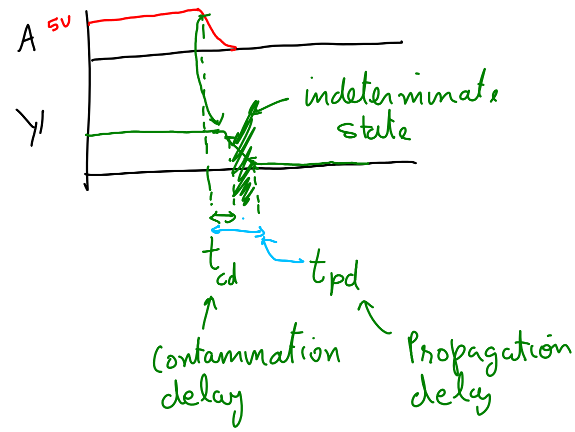


A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

A	Y
0	0
1	1

7.3.1 Delays





Definition 7.1 (Propagation delay (t_{pd})).

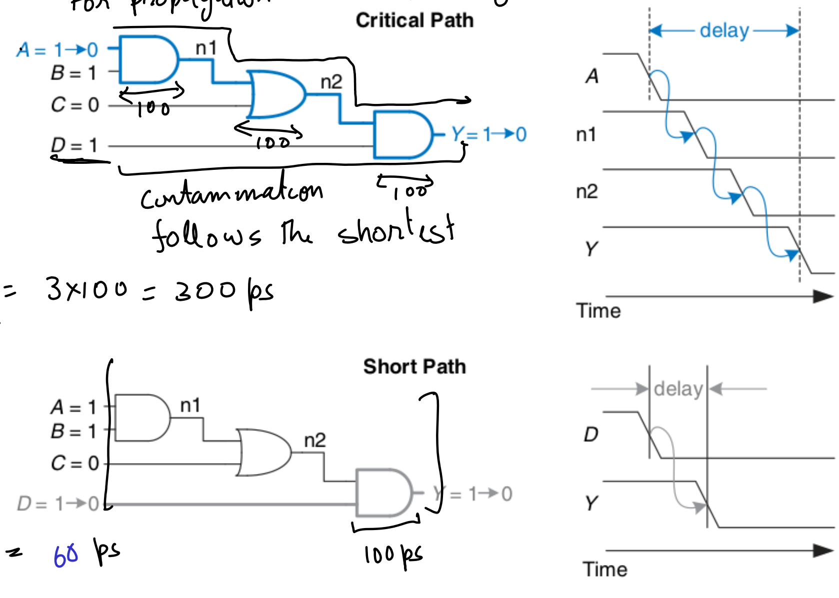
Propagation delay is the time period starting from the change in input to the settling down of the output to a valid steady state.

Definition 7.2 (Contamination delay (t_{cd})).

Contamination delay is the time period from the change in the input to any change in the output.

7.3.2 Paths

For propagation or stabalization you have to wait on the longest pat



$t_{pd} = 3 \times 100 = 300$ ps

$t_{cd} = 60$ ps

t_{cd} of a circuit = sum of t_{cd} of all elements on the shortest path

t_{pd} = sum of t_{pd} of all elements on the longest path

Example 7.3. *Find the propagation delay of the circuit above given that propagation delay of each gate is 100ps add contamination delay of 60ps.* = t_{cd}

t_{pd}

Example 7.4 (Ex 2.43 [1]). *Determine the propagation delay and contamination delay of the circuit in Figure 7.1. Use the gate delays given in Table 7.2.*

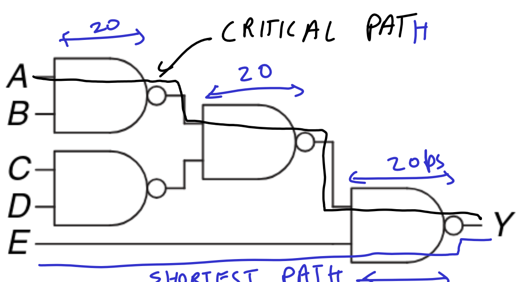


Figure 2.83 **Circuit schematic**

Table 2.8 **Gate delays for Exercises 2.43–2.47**

Gate	t_{pd} (ps)	t_{cd} (ps)
NOT	15	10
2-input NAND	20	15
3-input NAND	30	25
2-input NOR	30	25
3-input NOR	45	35
2-input AND	30	25
3-input AND	40	30
2-input OR	40	30
3-input OR	55	45
2-input XOR	60	40

$t_{pd} = 60$ ps

$t_{cd} = 15$ ps

Figure 7.1: Circuit

Table 2.8 **Gate delays for Exercises 2.43–2.47**

Gate	t_{pd} (ps)	t_{cd} (ps)
NOT	15	10
2-input NAND	20	15
3-input NAND	30	25
2-input NOR	30	25
3-input NOR	45	35
2-input AND	30	25
3-input AND	40	30
2-input OR	40	30
3-input OR	55	45
2-input XOR	60	40

Figure 7.2: Delays

Problem 7.1 (10 marks, Ex 2.44 [1]). *Determine the propagation delay and contamination delay of the circuit in Figure 7.3. Use the gate delays given in Figure 7.2.*

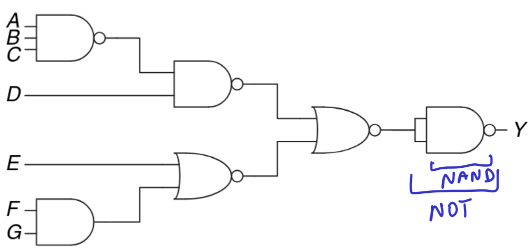
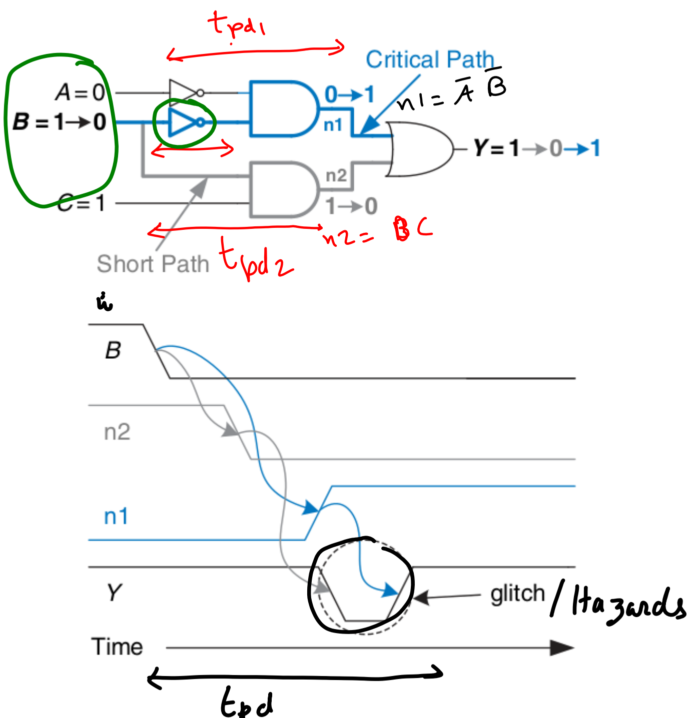


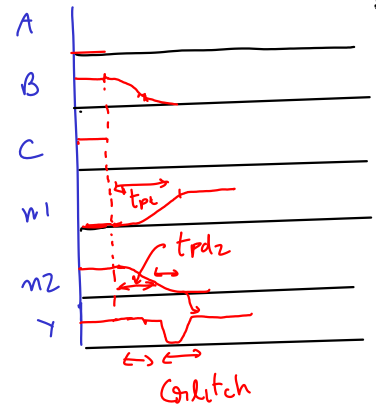
Figure 2.84 **Circuit schematic**

Figure 7.3: Circuit

7.4 Glitches or Hazards

$Y = \bar{A}\bar{B} + BC$





Definition 7.3 (Glitch or Hazard).

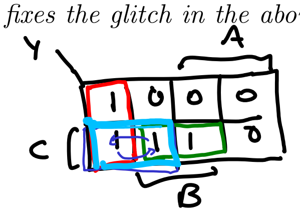
Glitch or Hazard is a temporary change in the output of a circuit that is not same as the expected steady state value.

Example 7.5. *Design a circuit that fixes the glitch in the above circuit (also known as glitch-free or hazard-free circuit).*

$Y = \bar{A}\bar{B} + BC$

Hazard free circuit

$Y = \bar{A}\bar{B} + BC + \bar{A}C$



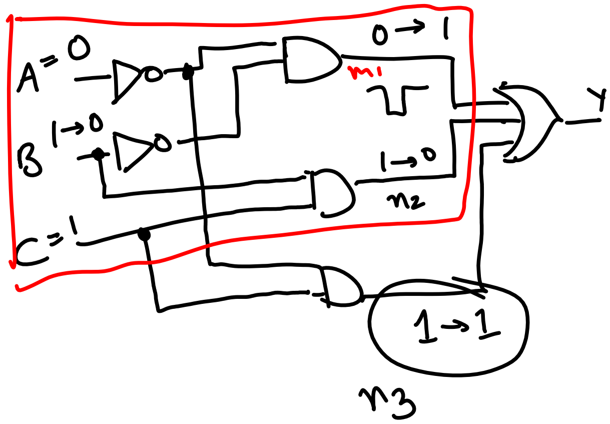
$A = 0$

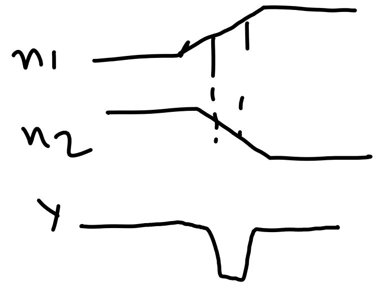
$C = 1$

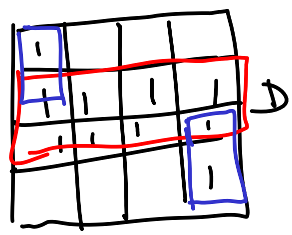
$B = 1 \rightarrow 0$

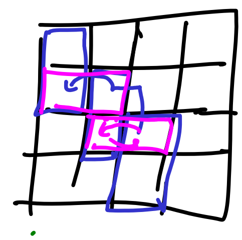
Glitches are caused due transitions betwee implicants that are adjacent and non-overlapping. Adding an overlapping implicant removes the glitch.

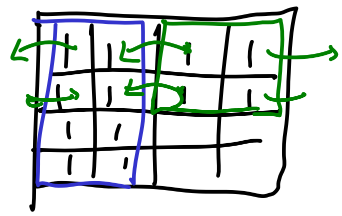
Problem 7.2. *Find a minimal Boolean equation for the function in Figure 2.85. Remember to take advantage of the don't care entries (marked X) (10 marks).*


A = 0
B 1→0
C = 1
0→1
n1
1→0
n2
1→1
n3
Y


n1
n2
Y







A	B	C	D	Y
0	0	0	0	X
0	0	0	1	X
0	0	1	0	X
0	0	1	1	0
0	1	0	0	0
0	1	0	1	X
0	1	1	0	0
0	1	1	1	X
1	0	0	0	1
1	0	0	1	0
1	0	1	0	X
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	X
1	1	1	1	1

← don't care

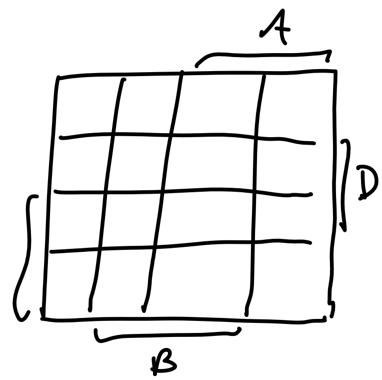


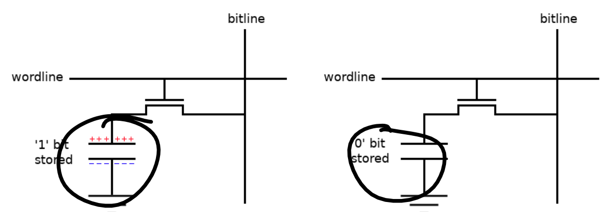
Figure 2.85 Truth table for Exercise 2.28

1. *Sketch a circuit for the function (10 marks).*
2. *Does your circuit from have any potential glitches when one of the inputs changes? If not, explain why not. If so, show how to modify the circuit to eliminate the glitches (10 marks).*

7.5 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)
   (a) Memories that require periodic refreshing. For example, DRAM: Dynamics Random Access memory (Not covered in this course)



[1]

   (b) Memories that are always refreshing. For example, SRAM: Static Random Access memory [3, Appendix B.64]

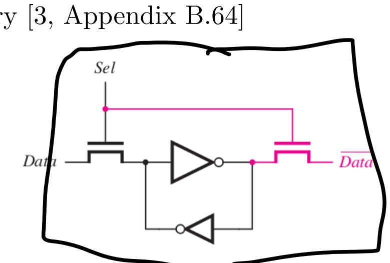
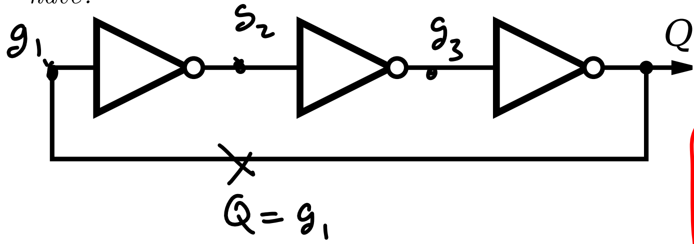


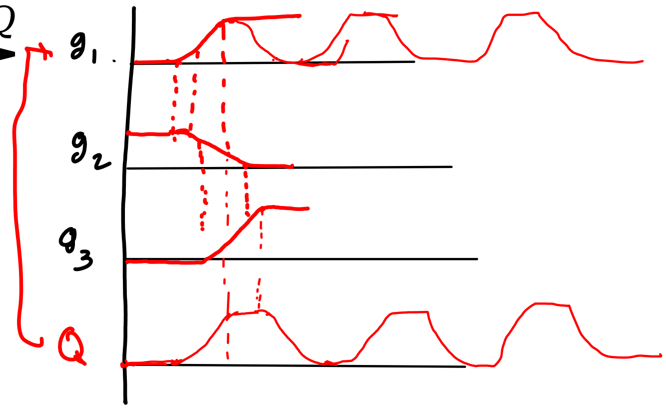
Figure B.64 An SRAM cell.

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

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

Example 7.6 (Ring oscillator). *[1, Sec 3.31] How many stable states does the following circuit have?*





This circuit does not have any stable states.

Definition 7.4 (Astable circuits).

Circuits without a stable state are called astable circu

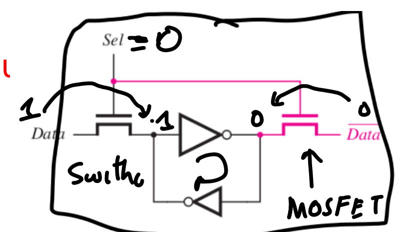
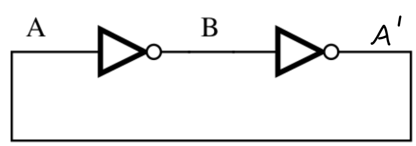
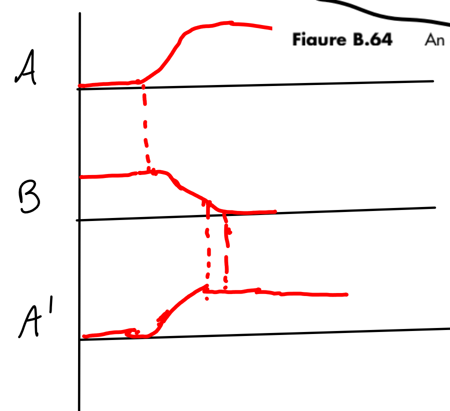


Figure B.64 An SRAM cell.

Example 7.7. *Analyze the timing diagram of the following circuit.*





$A = A'$

A(t)	B(t)	$A(t + t_{pd})$
0	1	0
1	0	1

← Two stable states

A stable state is a state of circuit, when the circuit stays in that state if initialized into that.

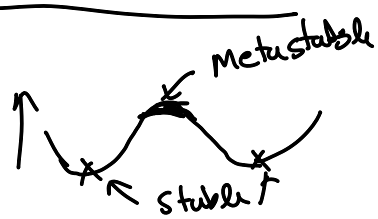
Definition 7.5 (Bistable circuits).

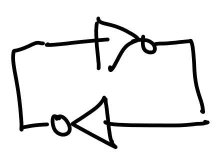
Bistable circuits have two stable states

Definition 7.6 (Characteristic or state table). *Draw the characteristic or state table of the above circuit.*

A(t) current state	$A(t+\Delta t) = A^+ = A(t+1)$ Next State
0	0
1	1

Metastable





NOR

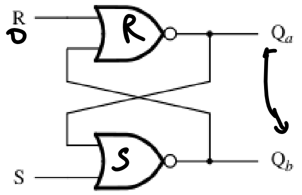
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

$Y = \overline{B}$ when $A = 0$

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

Definition 7.7 (SR latch)**.** *The following circuit is called the SR latch.*

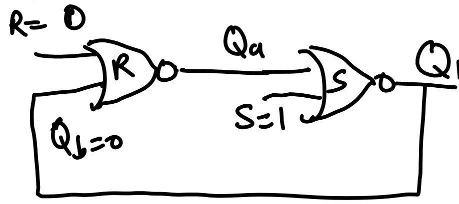
$R = 0$



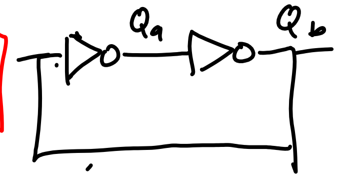
$\overline{S} = 1$ $\quad Q_a = 1$

=

$R = 0$



$S = 0$
$R = 0$
=



$Q_b = \overline{Q_a}$

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

S	R	Q_a	Q_a^+	
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	x	undefined indeterminata
1	1	1	x	

Case when $\underbrace{S = 1}_{\text{set}}$, $R = 0$

whichever is high Q_a/Q_b select that as Q

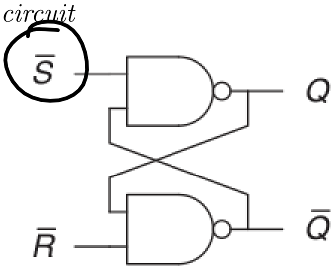
$\to Q_a = 1 \to Q_b = 0 \to Q_a = 1$

$\to Q_a = 0$

$\to Q_b = \overline{Q_a + S} = 0 \to Q_a = \overline{Q_b + R} = 1$

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

Problem 7.3 (SR latch using NAND gates)**.** *Draw the characteristic or state table for the following circuit*



when $S=0$, $\underbrace{R=1}_{\text{Reset}}$

$\rightarrow Q_a = 1$

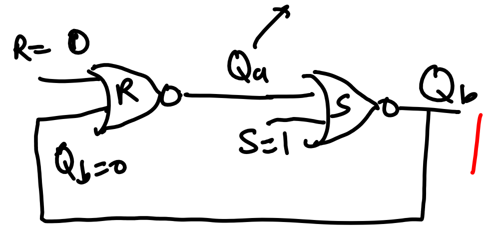
$Q_b = \overline{Q_a + S} = 0$

$Q_a = \overline{Q_b + R} = 0.$

$\rightarrow Q_a = 0$

$Q_b = \overline{Q_a + S} = 1$

$Q_a = \overline{Q_b + R} = 0$

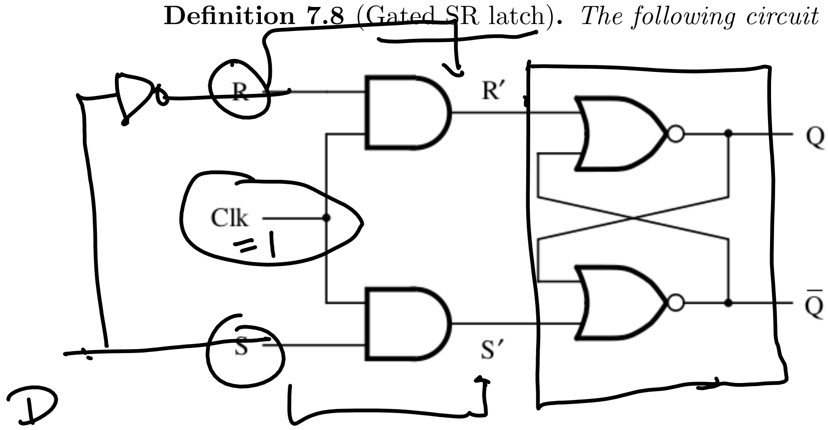


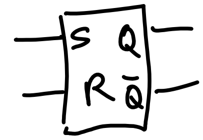
A	B	$Y = \overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

S	R	Q^+	
0	0	Q	} Hold
0	1	0	} Reset
1	0	1	} Set
1	1	x	

7.6.2 Gated SR latch [3, Sec 5.2]

Definition 7.8 (Gated SR latch). *The following circuit is called the Gated SR latch.*

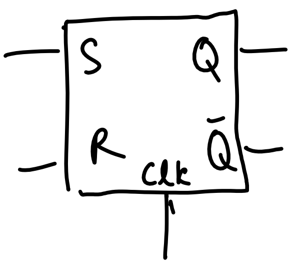




Clk	R	R′
0	0	0
0	1	0
1	0	0
1	1	1

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

Clk	R′
0	0
1	R



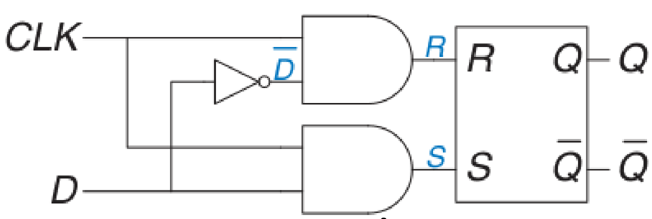
Clk	S	R	Q^+	
0	*	*	Q	Hold
1	0	0	Q	Hold
1	0	1	0	Reset
1	1	0	1	set
1	1	1	X	indeterminate state

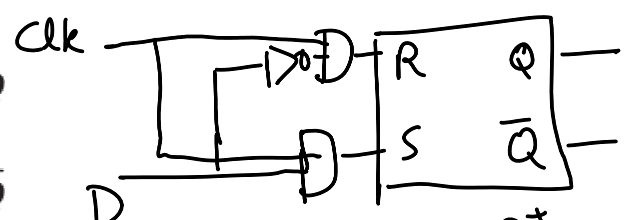
A clock input to the latch makes the state affected by input only when the clock is high.

Latches are level triggered (opposite to edge-triggered as in flip flops)

7.6.3 D (Data) latch [1, Sec 3.2.2] Transparent latch

Definition 7.9 (D latch). *The following circuit is called the D latch.*

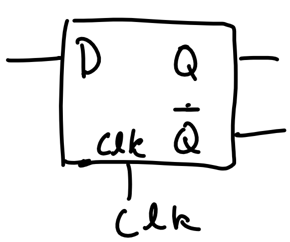




Clk	D	Q^+
1	0	0
1	1	1
0	*	Q
0	*	Q

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

Clk	Q^+
0	Q
1	D



Problem 7.4. *Consider the timing diagram in Figure 7.4. Assuming that the D and Clock inputs shown are applied to the circuit in Figure 7.5, draw waveforms for the Q_a , Q_b , and Q_c signals. (10 marks) [?, Prob 5.1]*

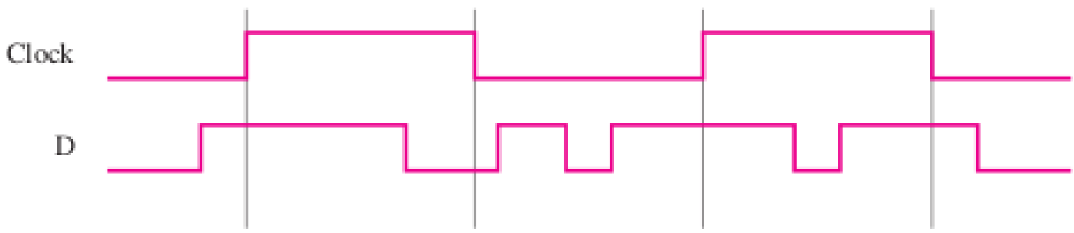
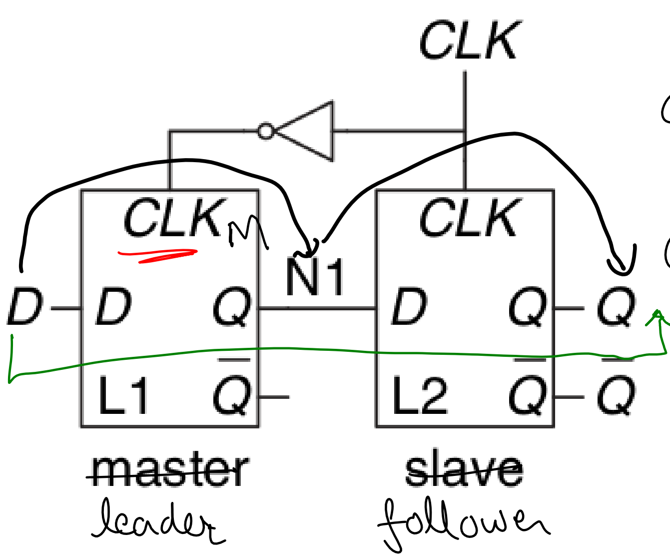
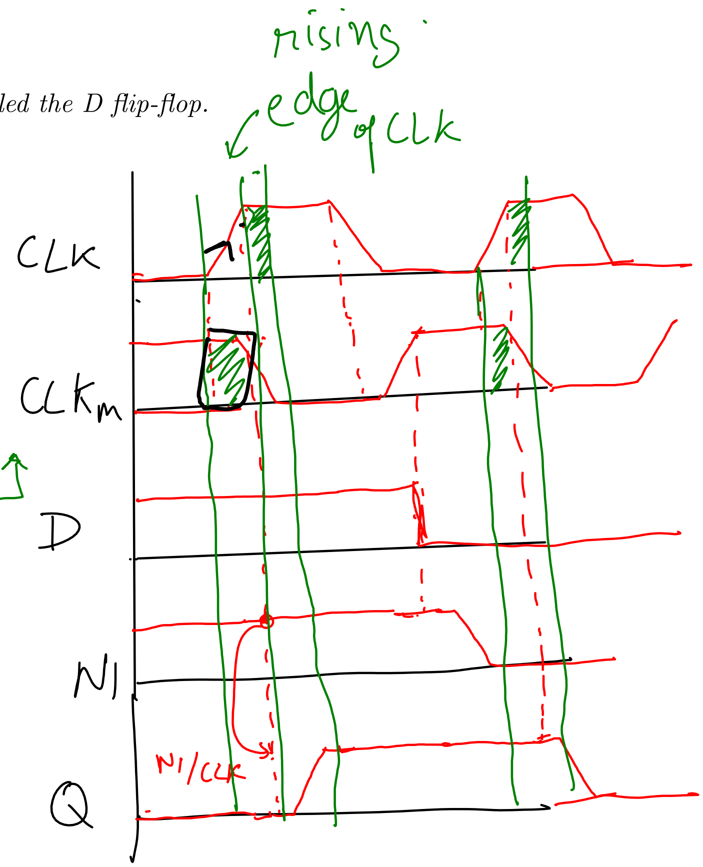


Figure 7.4: Inputs for Prob 7.4

7.6.4 D flip-flop [1, Sec 3.2.2]

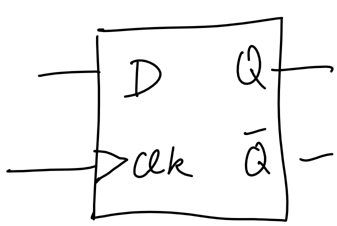
Definition 7.10 (D flip-flop)**.** *The following circuit is called the D flip-flop.*

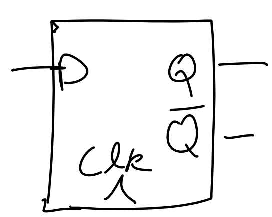




1. *Draw its timing diagram*
2. *Draw its characteristic table.*
3. *Draw the D flip-flop symbol*

D	Q^+
0 →	0
1 →	1





clk	D	Q^+
↑	0	0
↑	1	1
*	*	Q

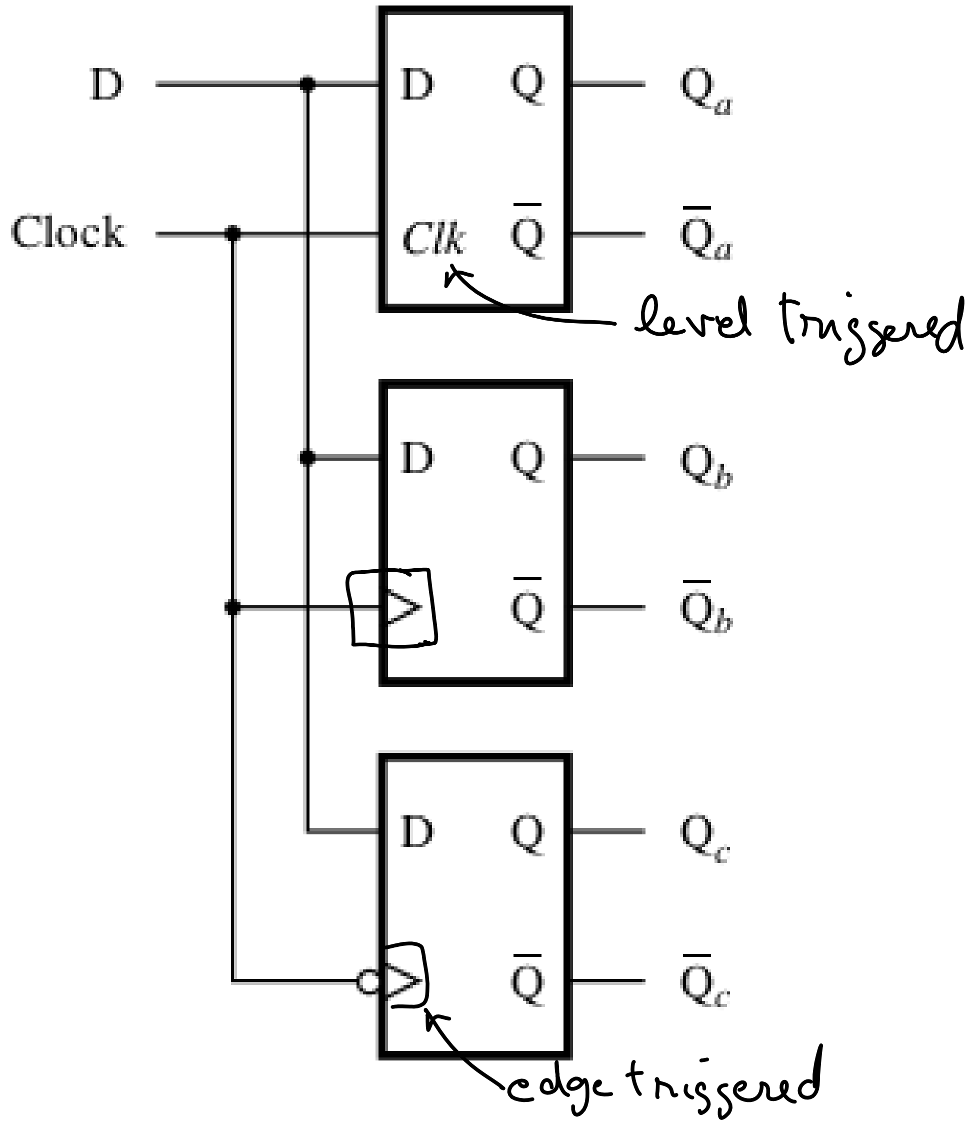
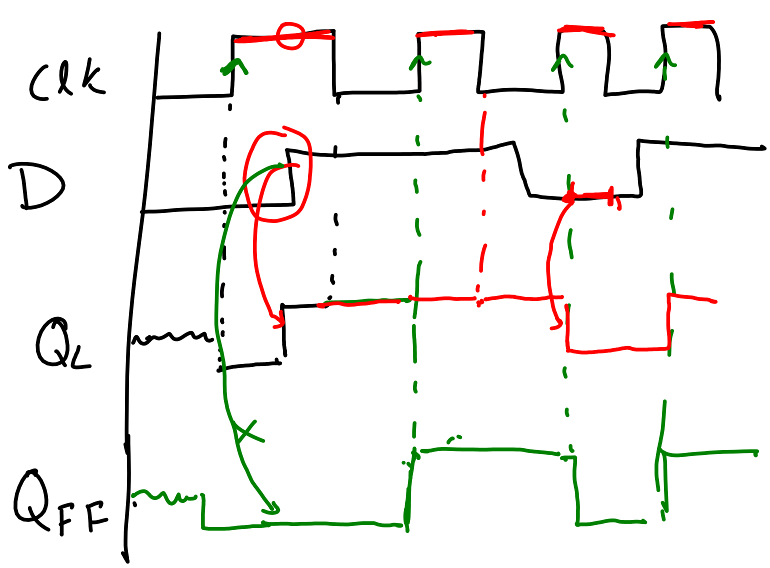


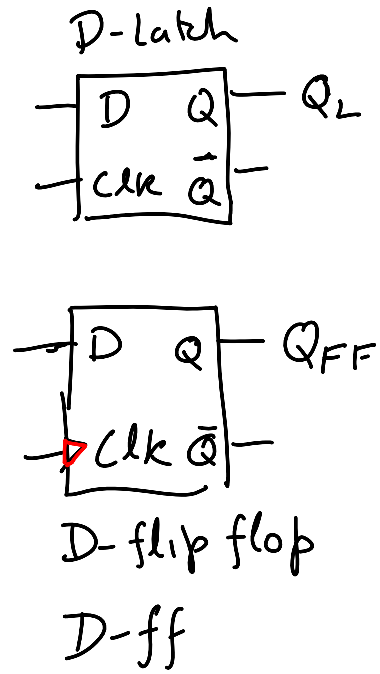
Figure 7.5: Circuit for Prob 7.4. Recall that *Clk* with a "▷" symbol indicates rising-edge (positive-edge) triggered flip-flop. *Clk* without "▷" symbol indicates a level-triggered latch. *Clk* with "○▷" symbol indicates a falling-edge (negative-edge) triggered flip-flop.

Remark 7.1. *What is the difference between a latch and a flip-flop?*

Latches are level-triggered: The states gets affected from inputs when clock is high
Flip-flops are edge-triggered: The states get affected from inputs at the rising edge of the clock

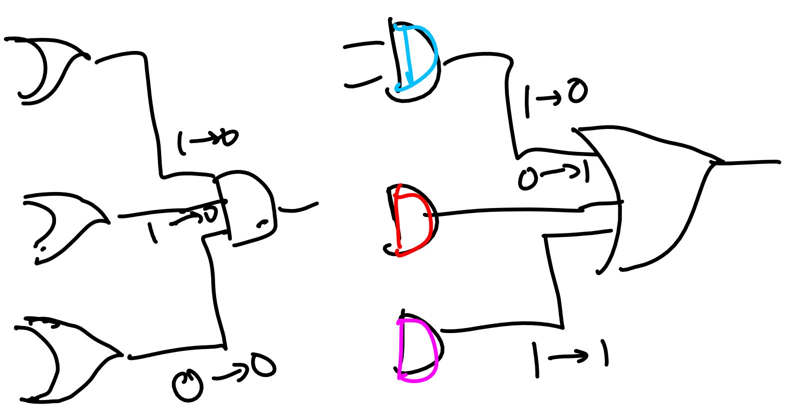
Example 7.8. *Add a* RESET *signal to the D flip-flop that resets the state of flip-flop to 0.*

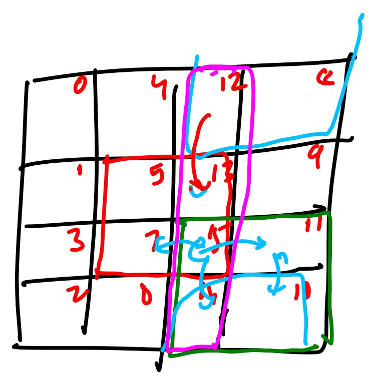




Asynchronous sequential circuits: not synchronized with the clock. E.g. Latches.
Synchronous sequential circuits: circuits synchronized with the clock. All states
and outputs change at the rising edge of the clock. E.g. Flip flops, combinations circuits

You want to stick with Synchronous sequential circuits.
1. No loops in the circuit without a flip flop. If there is a loop then there is a flip flop on that loop.
2. The propagation delay in the loop should be smaller than the clock cycle.



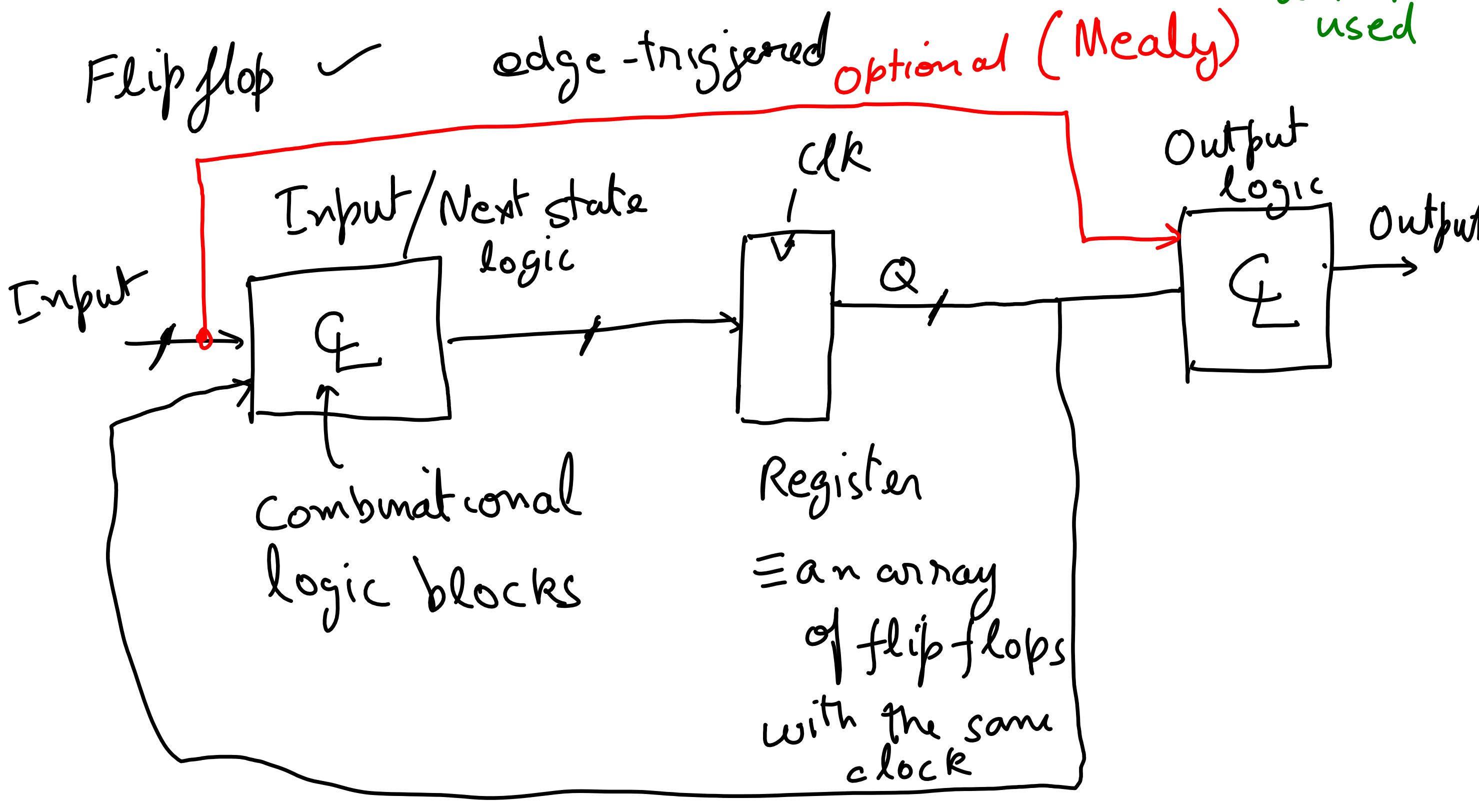


Synchronous vs Asynchronous sequential circuits

MOORE - if the optional connection is not used

Latches ✗

Flipflop ✓ edge-triggered



① D-ff ② T-ff ③ J-K ff

Data/Delay Toggle Set-Reset

Example 7.9. *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.*

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

T	Q	Q^+	
0	0	0	Hold
0	1	1	
1	0	1	Toggle
1	1	0	

J (Set)	K (Reset)	Q^+	
0	0	Q	Hold
0	1	0	Reset
1	0	1	Set
1	1	$\overline{Q}$	Toggle

D	Q^+
0	0
1	1

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

Problem 7.5. *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.*

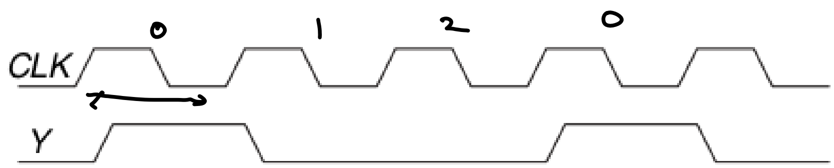
1. *Construct a JK flip-flop using a D flip-flop and some combinational logic.*
2. *Construct a D flip-flop using a JK flip-flop and some combinational logic.*
3. *Construct a T flip-flop (see Exercise 3.9) using a JK flip-flop.*

7.7 Finite State Machines [1, Sec 3.4]

[2]

Example 7.10. *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 7.6. *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:*



(a)

Sketch circuit designs for such a counter

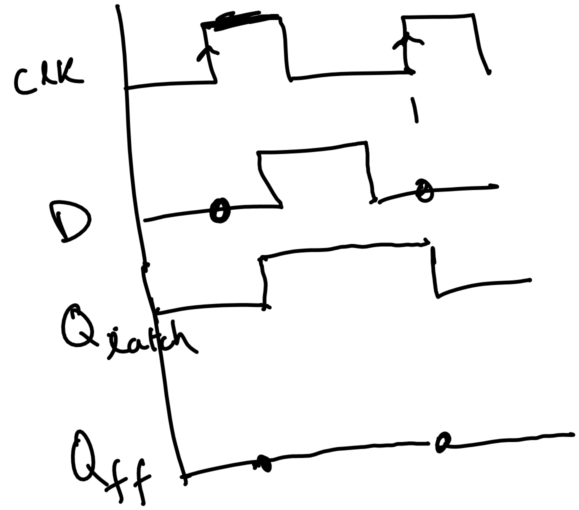
[2]These notes will not fit on your note sheet.

D	Q^+ ← next state
0	0
1	1

D-ff

T	Q^+ ← next state
0	Q ← prev state
1	$\overline{Q}$ ←

Toggle-ff



Set Reset

J	K	Q^+	
0	0	Q	Hold
0	1	0	Reset
1	0	1	Set
1	1	$\overline{Q}$	Toggle

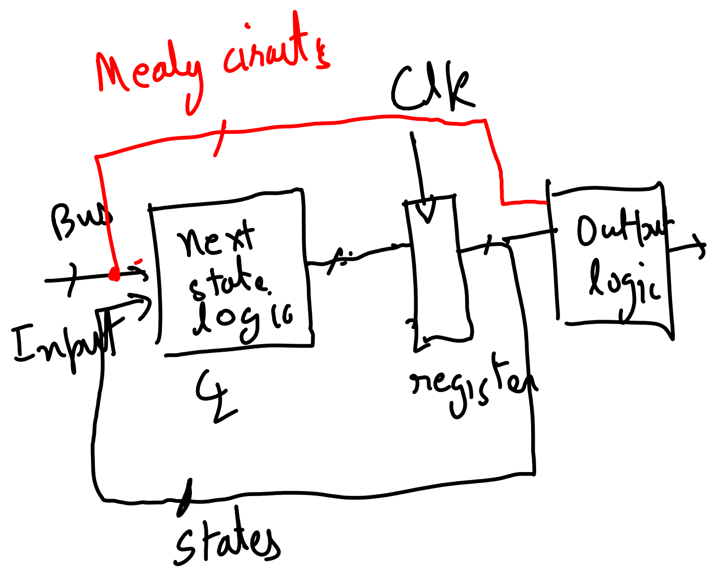
Sequential logic circuit

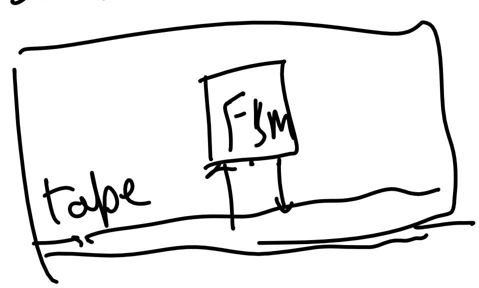
Synchronous

Finite state machines

k-ff ⟺ 2^k states

Turing machines





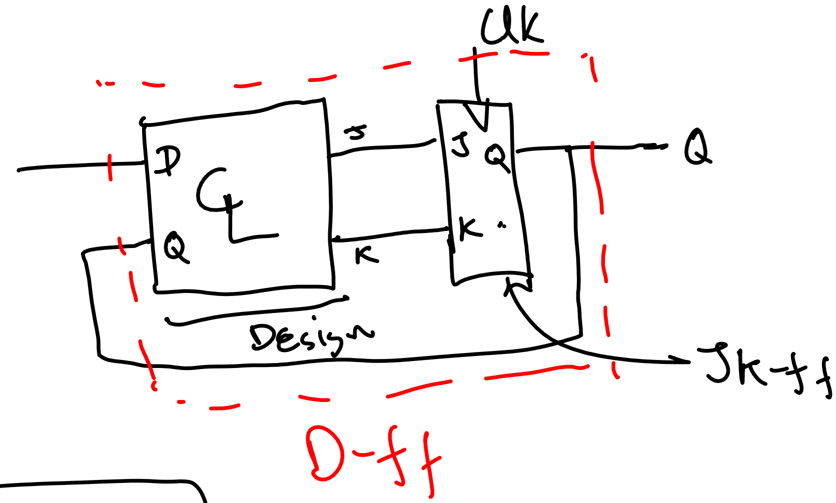
↳ Moore → Output only depends on current state

↳ Mealy → Output depends on current state and current input

Design a D-ff from JK-ff

$J = D$

$K = \overline{D}$



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

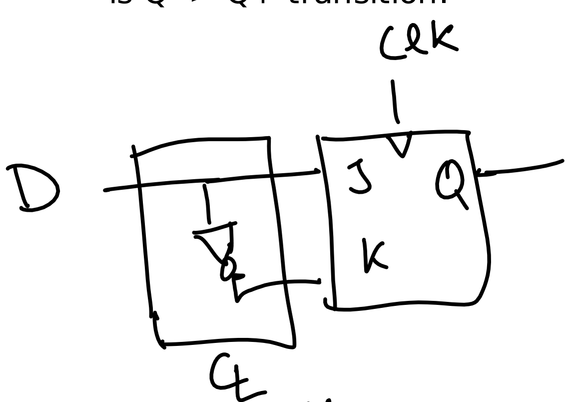
D	Q	Q^+
0	0	0
0	1	0
1	0	1
1	1	1

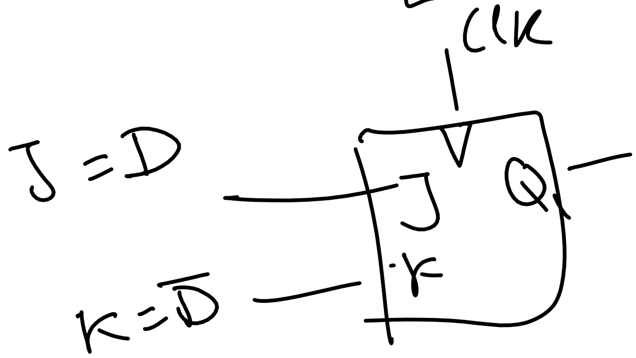
	$Q \to Q^+$		J	K
Reset/Hold	0	0	0	0/1 → d
Set/Toggle	0	1	1	d
reset/Toggle	1	0	d	1
set/Hold	1	1	d	0

Excitation table

J	K	Q^+
0	0	Q
0	1	0
1	0	1
1	1	$\overline{Q}$

What ff inputs will give you
a desired "excitation" that
is Q -> Q+ transition.





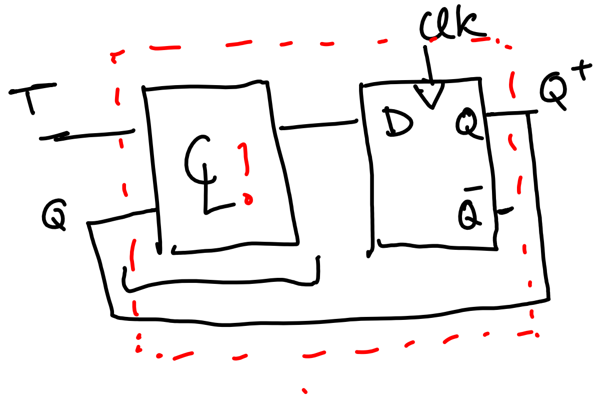
Design a T-ff using a D-ff

want

T	Q	D
0	0	0
0	1	1
1	0	1
1	1	0

$D = T \oplus Q$

$= \bar{T}Q + T\bar{Q}$



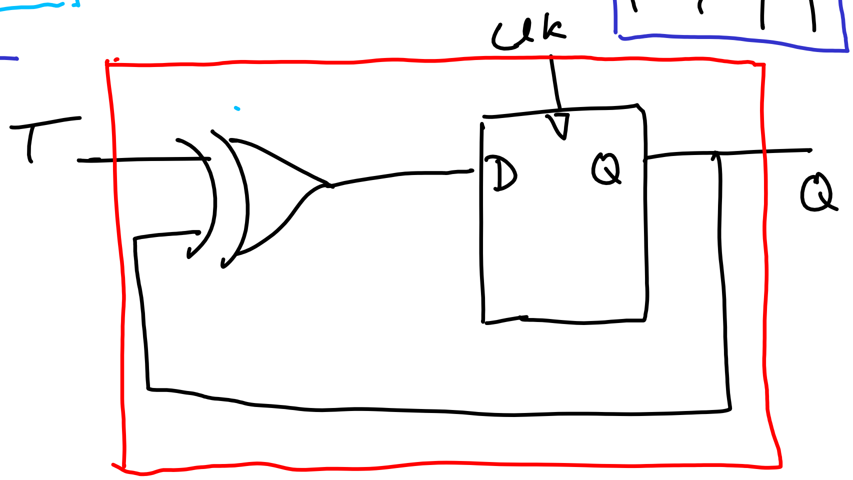
I have characteristic table of T-ff and Excitation table of D-ff

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

Desired state transition table

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

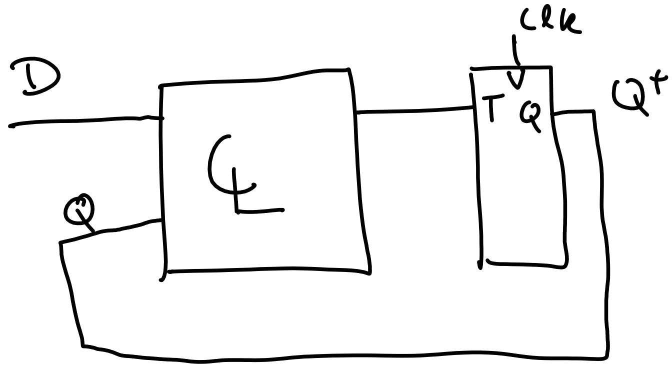
D	Q^x	Q^+
0	0	0
0	1	0
1	0	1
1	1	1



1. Decided on a ff, draw a block diagram to convert sequential circuit design to combinational logic design
2. We want to find the truth table for the combinational logic design
3. Write down the desired state transition table and the excitation table of the flip flop used.
4. Combine both to get the the truth table for combinational logic.

Design a D-ff using a T-ff

Design a D-ff using a T-ff



Want

D	Q	T
0	0	0
0	1	1
1	0	1
1	1	0

$T = D \oplus Q$

State transition table

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

Excitation table

For each possible state transition,
what should the input to my flip flop be?

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

Excitation table for J-K ff

Q	Q^+	J	K	
0	0	0	d	← Hold/Reset
0	1	1	d	Toggle/set
1	0	d	1	Toggl/Reset
1	1	d	0	Hold/set

J	K	Q^+
0	0	Q
0	1	0
1	0	1
1	1	$\overline{Q}$

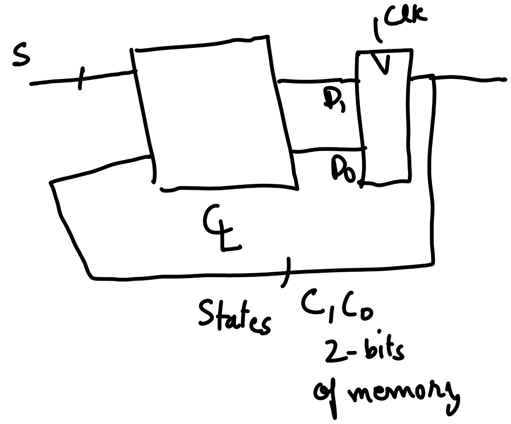
if $S=0$

$C_1C_0 = C_1C_0$

if $S=1$

$C_1C_0 = C_1C_0 + 1$ ← numerical addition

	Prev states		next states	
S	C_1	C_0	$C_1^+ = D_1$	$C_0^+ = D_0$
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	Q^+
0	0
1	1

$Q^+ = D$

D_1

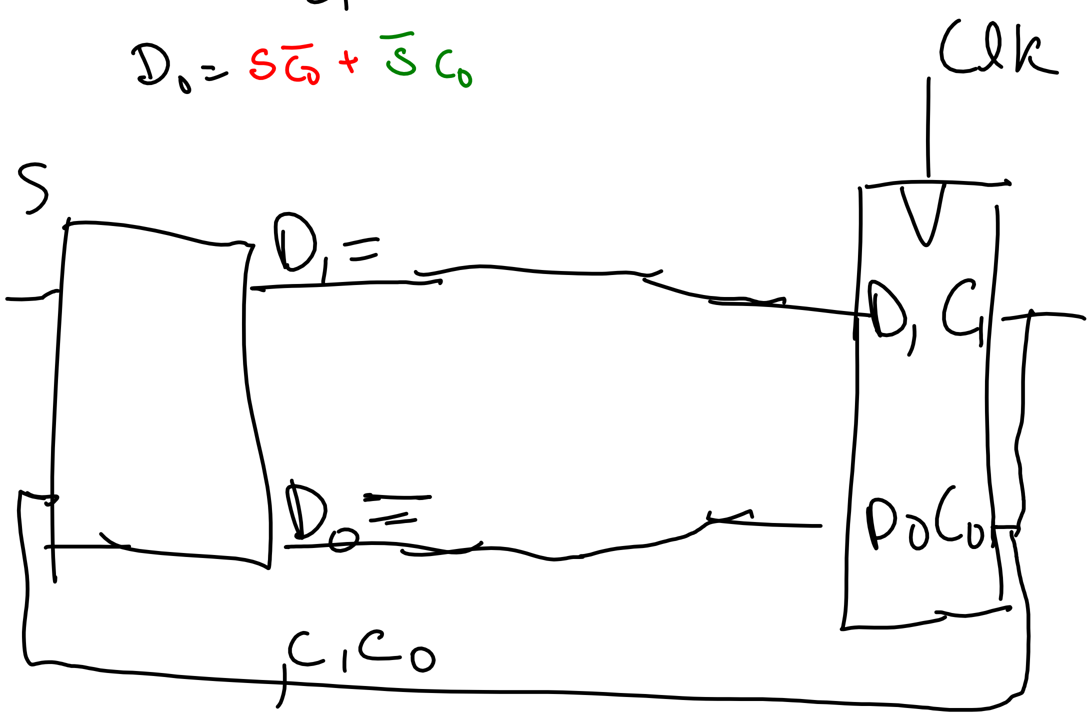
		C_1	C_1, S	S
	0	1	1	0
C_0	0	1	0	1

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

D_0

		C_1	C_1, S	S
	0	0	1	1
C_0	1	1	0	0

$D_0 = S\bar{C_0} + \bar{S}C_0$



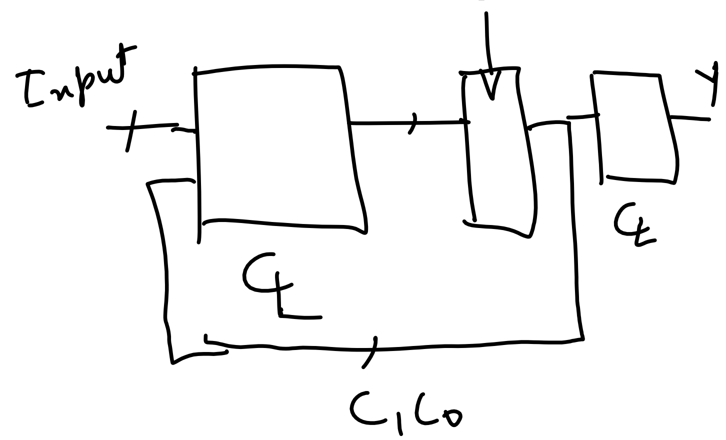
How many states? ceil)

3 - states $= \lceil \log_2(3) \rceil =$

$2^1 < 3 \leq 2^{\boxed{2}}$

C_1	C_0	$D_1 = C_1^+$	$D_0 = C_0^+$

C_1	C_0	Y

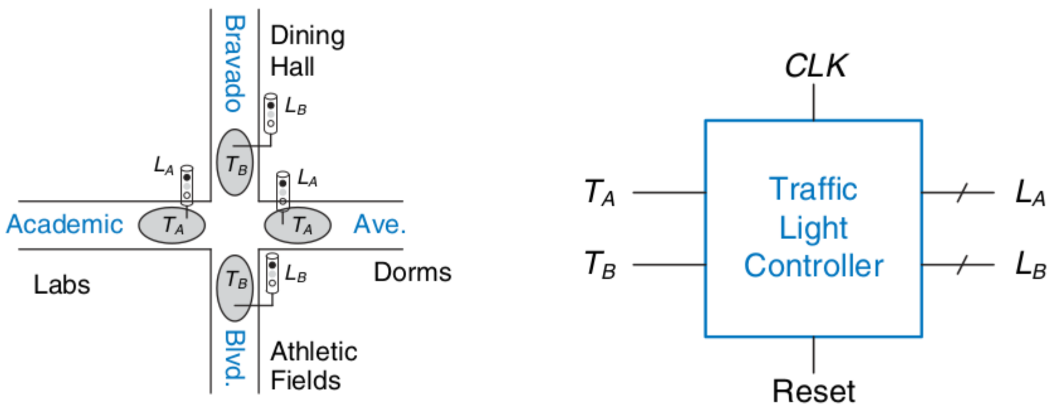


Problem 7.7. *Design a 3-bit counter which counts in the sequence: 001, 011, 010, 110, 111, 100, (repeat) 001, ...*

Example 7.11. *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 7.12 (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*

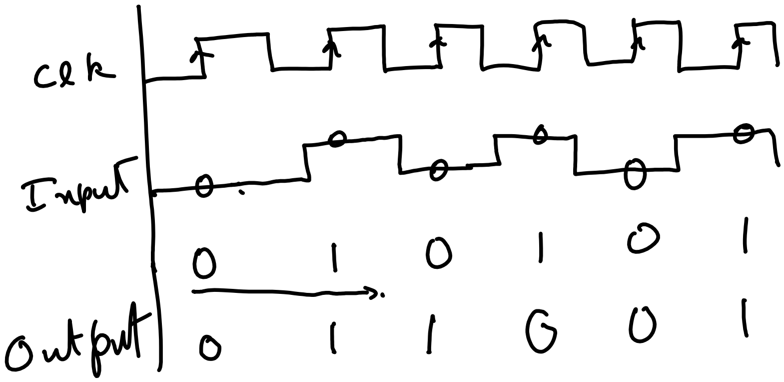
Example 7.13. *Consider the problem of inventing a controller for a traffic light at a busy intersection on campus. There are two traffic sensors, T_A and T_B , on Academic Ave. and Bravado Blvd., respectively. Each sensor indicates TRUE if students are present and FALSE if the street is empty. There 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 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 7.8. *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.*

Example 7.11. *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

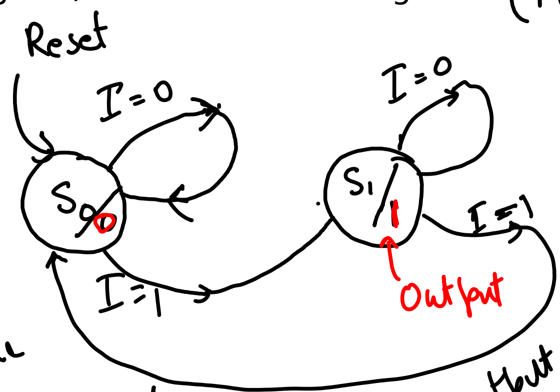
Input	0	1	1	1	0	1	0	1	0	1	0
Output	0	1	0	1	1	0	0	1	1	0	0

Is it sequential or combinational? Yes

How much memory? How many states does this circuit need? = 2 states = 1-bit memory = 1-ff

State diagram/State transition diagram (Moore)

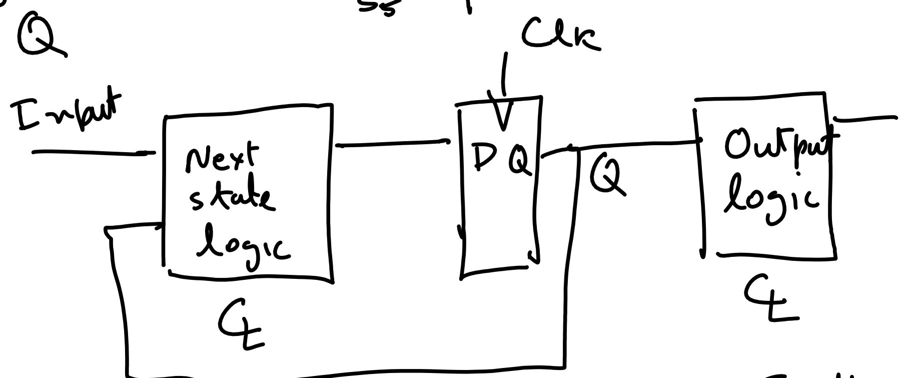
Q	
$S_0 = 0$	
$S_1 = 1$	



State assignment (guideline methods)

$S_0 \to 010$?
S_1
$S_2 \to 111$?
S_3
S_4
S_5

I	Q	Q^+ next state = D	O = Q = output
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	1

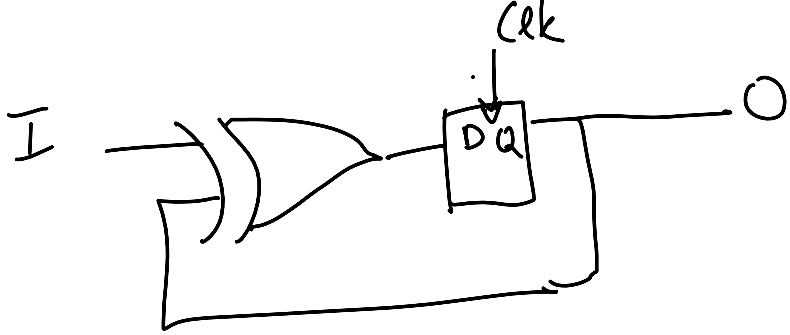


Next state table/State transition table/State table

Truth table

I	Q	D
0	0	0
0	1	1
1	0	1
1	1	0

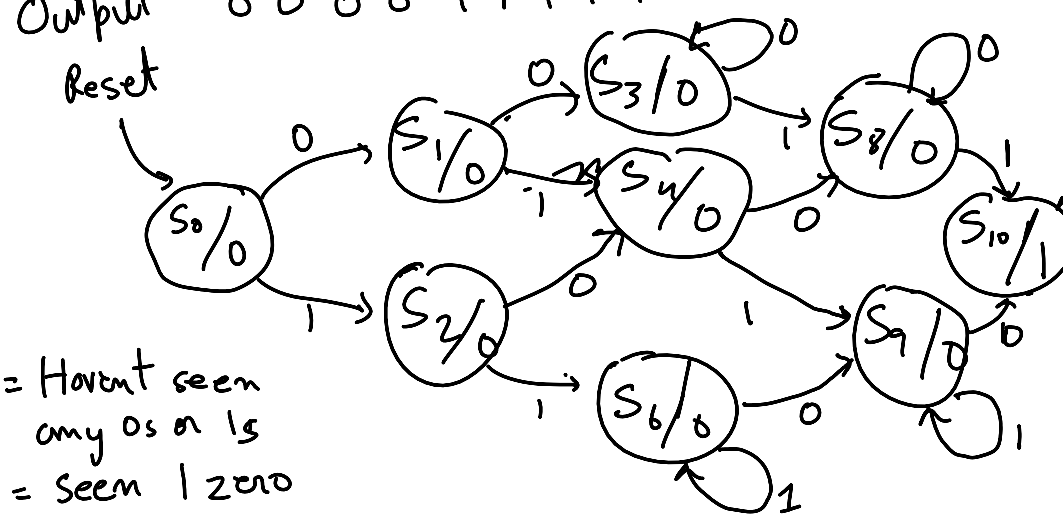
$D = I \oplus Q$



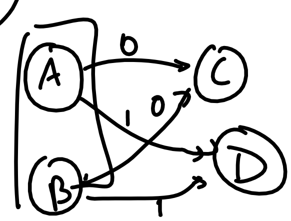
Example 7.12 (Sequence detectors). *A sequential circuit has one input and one output. The output becomes 1 and remains 1 thereafter when at least two 0's and at least two 1's have occurred as inputs regardless of the order of* in

Input 0 1 1 1 0 1 0 1 0 1 1 0

Output 0 0 0 0 1 1 1 1 1 1 1 1



Try to reuse the same states as much as possible.
If two states have the same next states and outputs for corresponding inputs, then the two states are equivalent.



0 S_0 = Haven't seen any 0s or 1s

1 S_1 = seen 1 zero

2 S_2 = seen 1 one.

3 S_3 = seen 2 zeros = seen at least 2 zeros

$\boxed{\begin{array}{l} 4\ S_4 = \text{seen 1 zero / 1 one} \\ 5\ S_5 = \text{seen 1 zero / 1 one} \end{array}}$ Merged

6 S_6 = seen 2 ones = seen at least 2 ones

7 S_8 = at least 2 zeros + seen 1 - one

8 S_9 = at leas 2 ones + seen 1 - zero

9 S_{10} = at least 2 zeros + seen 2 ones

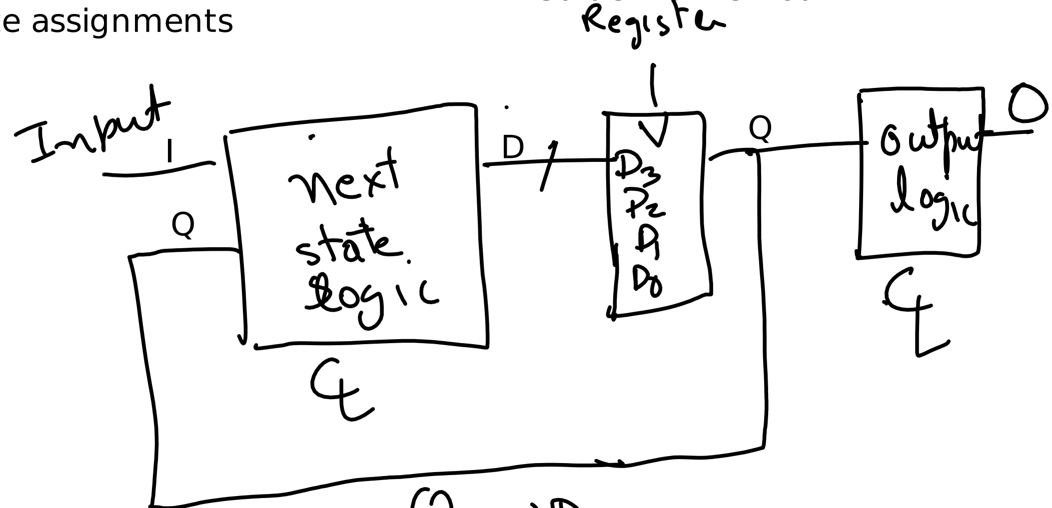
9 states = ff?

State reduction techniques and algorithms

- Row reduction
- Implication tables

State transition table/state assignments

Guideline method

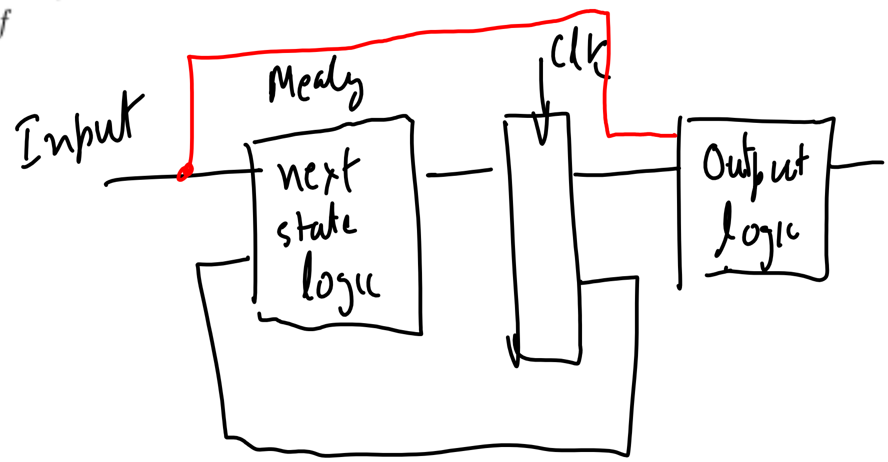


I	Q_3	Q_2	Q_1	Q_0	$D_3 = Q_3^+$	$D_2 = Q_2^+$	$D_1 = Q_1^+$	$D_0 = Q_0^+$	$O(Q_{3:0})$

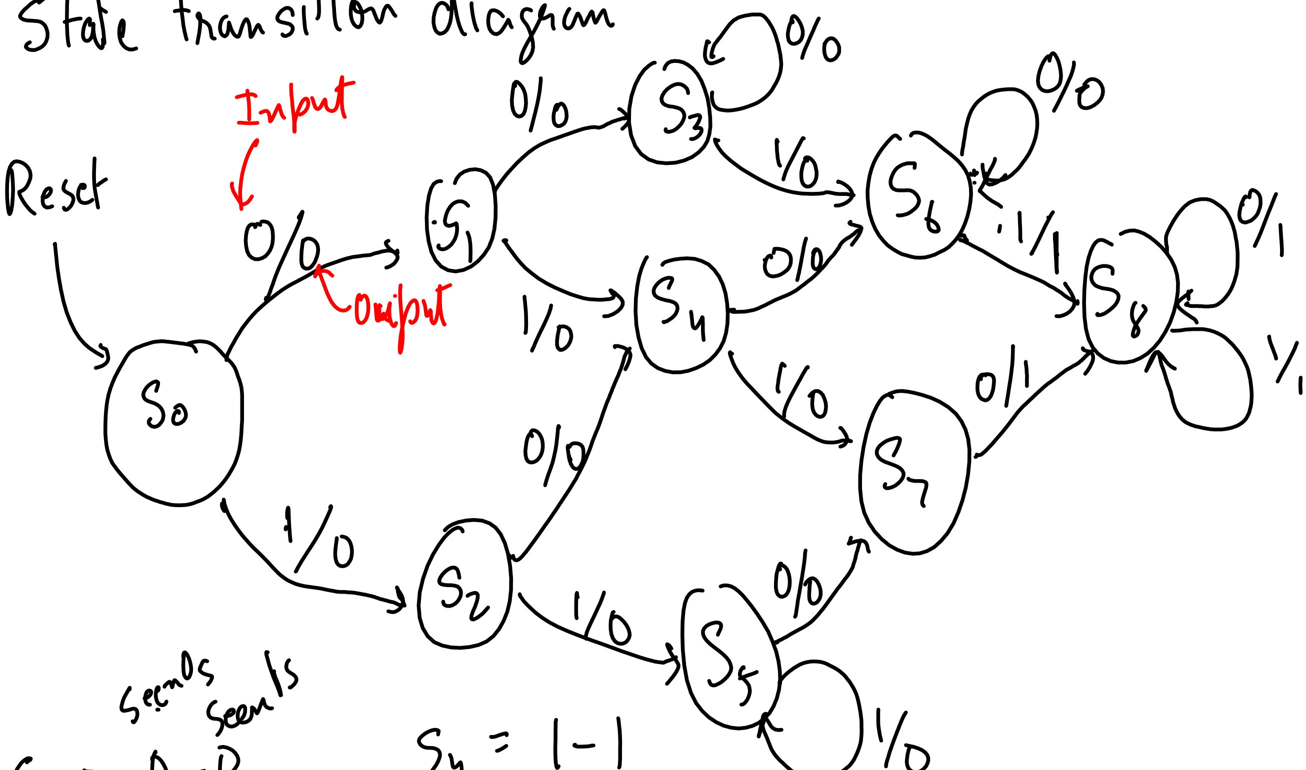
March 28,

Finite state machines
1. Moore: the output depends only on the state
2. Mealy: the output depends on both state and the input

Example 7.12 (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* in



State transition diagram



seen 0s seen 1s

$S_0 = 0-0$

$S_1 = 1-0$

$S_2 = 0-1$

$S_3 = 2-0$

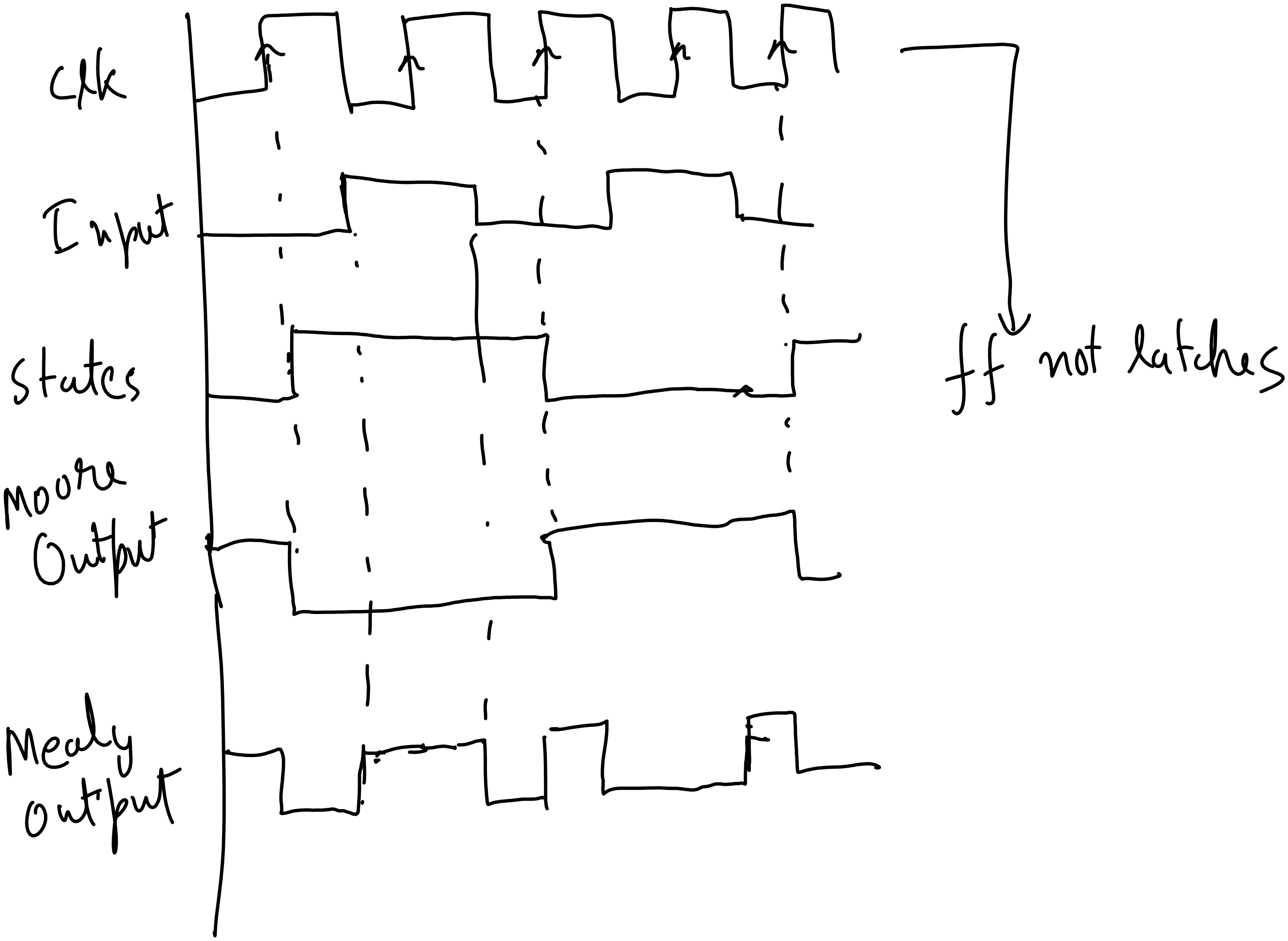
$S_4 = 1-1$

$S_5 = 0-2$

$S_6 = 2-1$

$S_7 = 1-2$

$S_8 = 2-2$



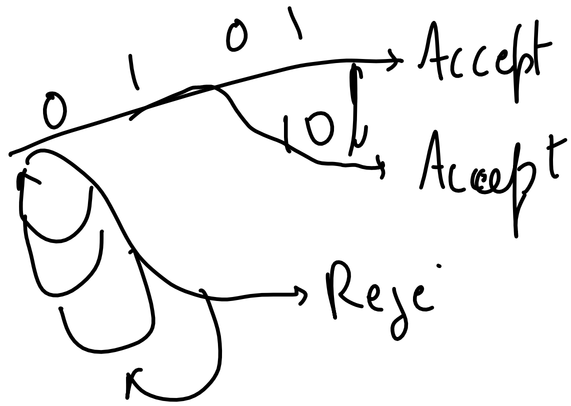
Advantages and disadvantages:
1. Moore: Makes the input synchronized with the clock/ Mealy: causes the output to vary with input
2. Moore: Needs more (or equal) states than a Mealy design.

Your circuit has one input and one output. Detect a sequence 0101 or 0110. Whenever the sequence is detected, the system does not reset and outputs a 1 for one clock cycle.

Example:
Input: 0 0 0 1 0 1 1 0 0 1 1 0 1
Output: 0 0 0 0 0 1 0 1 0 0 0 1 0

MOORE

Draw Moore state transition diagram for this problem



Acceptable sequences seen so far

$S_0 = ****$

$S_1 = ***0$

$S_2 = **01$

$S_3 = *010$

$S_4 = *011$

$S_5 = 0101$

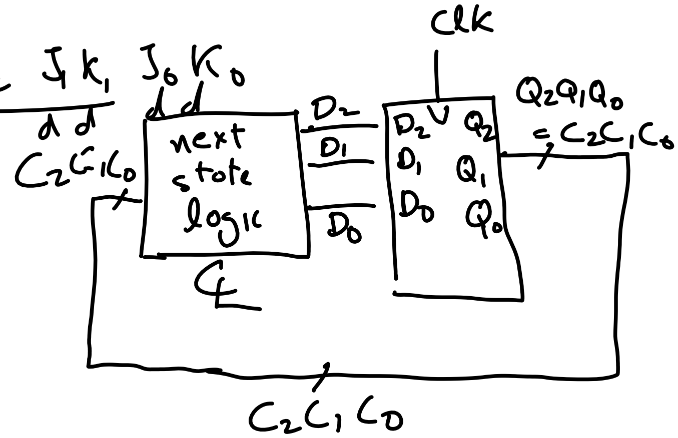
$S_6 = 0110$

$C_2 C_1 C_0$

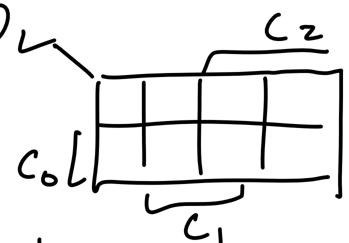
Problem 7.7. *Design a 3-bit counter which counts in the sequence: 001, 011, 010, 110, 111, 100, (repeat) 001, ...*

S_0 S_1 S_2 S_3 S_4 S_5

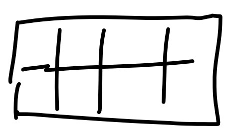
state C_2	C_1	C_0	next state $D_2 = C_2^+$	$C_1^+ = D_1$	$C_0^+ = D_0$	J_2	K_2	J_1	K_1	J_0	K_0
0	0	0	d	d	d	d	d	d	d	d	d
$\boxed{0}$	0	1	$\boxed{0}$	1	1	d	d				
$\boxed{0}$	1	0	$\boxed{1}$	1	0	0	d				
0	1	1	0	1	0	1	d				
1	0	0	0	0	1						
1	0	1	d	d	d						
1	1	0	1	1	1						
1	1	1	1	0	0						



D_2



D_1



Excitation table

$D = Q^+$

C_2^+ (C_2 over the last two columns, C_0 on the second row, C_1 under the middle two columns)

d	1	1	0
0	0	1	0

$\rightarrow J_2$, K_2

J_2

d	1	d	d
0	0	d	d

K_2

d	d	0	1
d	d	0	d

Excitation table

C_2	C_2^+	J_2	K_2
0	0	0	d
0	1	1	d
1	0	d	1
1	1	d	0

when $C_2 = 0$: $J_2 = C_2^+$ / $K = d$

else: $J_2 = d$ / $K = \overline{C_2^+}$

Problem 7.8. *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.*

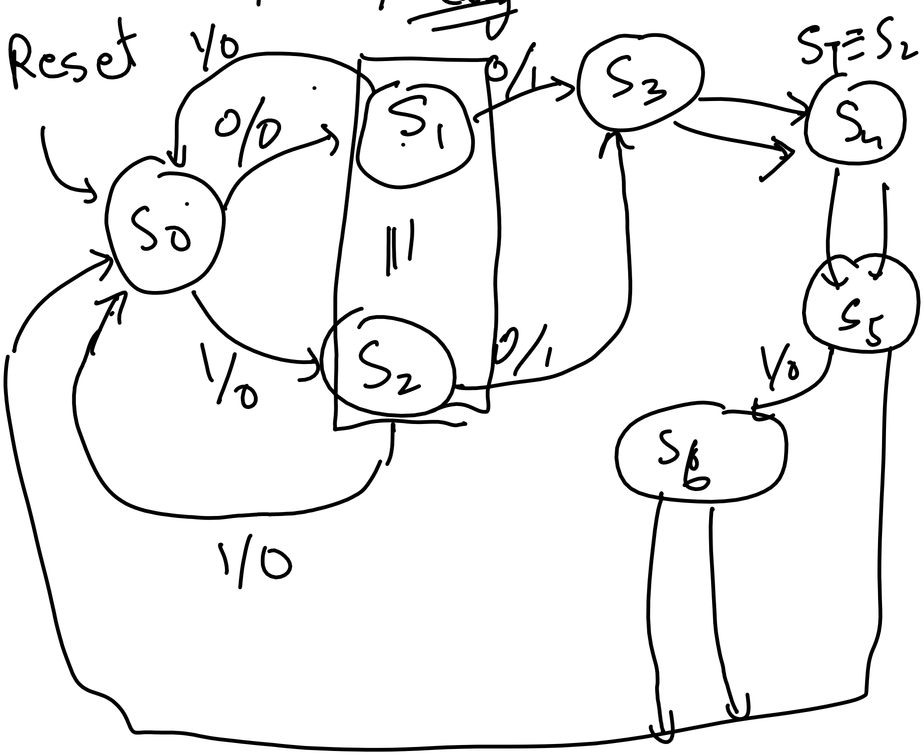
.

1. Decide on states, draw state transition diagram/state transition table
2. Assign binary values to the states
3. Get State transition table with binary values
4. Use excitation table of the ff to get truth table
5. Design the required combinational logics

Input X=0 Paddle move up

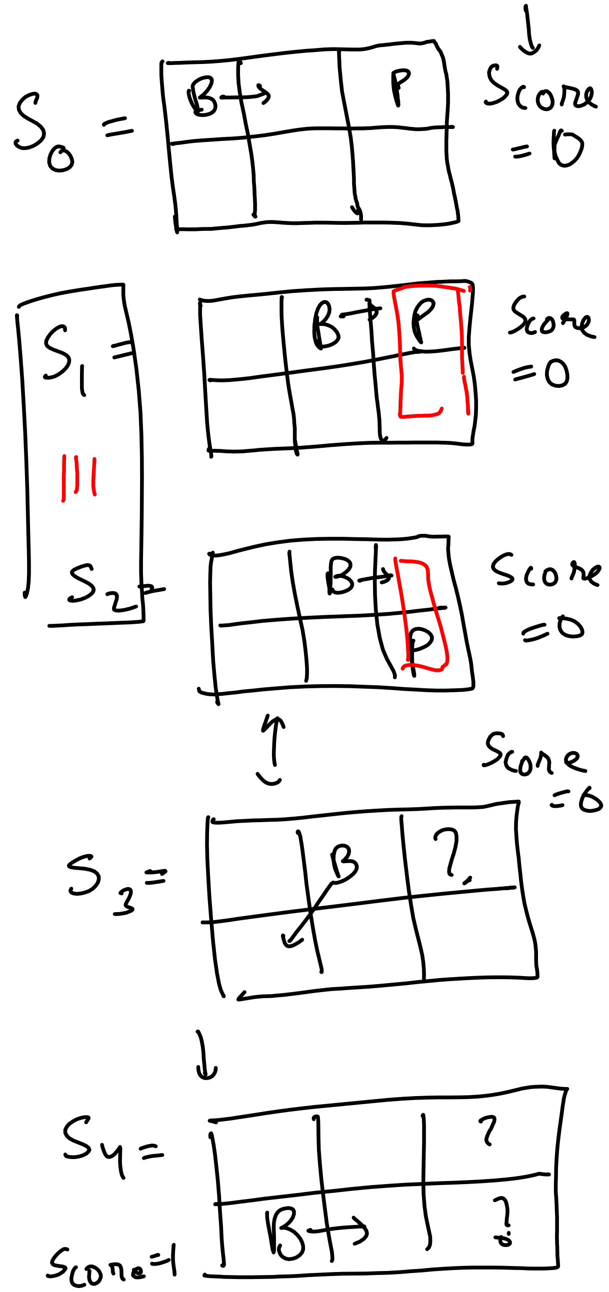
X=1 : paddle moves down

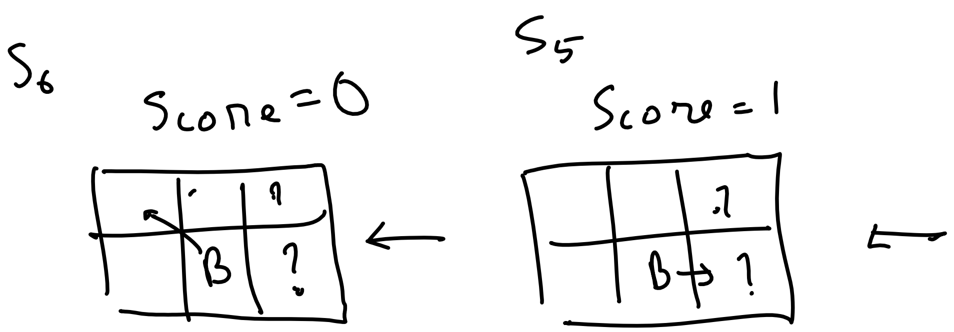
Moore / Mealy



State meanings

output





State encoding

$Q_2\ Q_1\ Q_0$

$S_0 = 0\ 0\ 0$ $\boxed{S_1 = 0\ 0\ 1}$ $S_3 = 0\ 1\ 1$ $S_4 = 1\ 0\ 0$ $S_5 = 1\ 0\ 1$ $S_6 = 1\ 1\ 0$

State encoded transition table

state	next state $X=0$	next state $X=1$	Output $X=0$	Output $X=1$
$X\ Q_2\ Q_1\ Q_0$	$D_2 = Q_2^+\ Q_1^+\ Q_0^+$	$Q_2^+\ Q_1^+\ Q_0^+$	Z	Z
$S_0 =$ 0 0 0	$S_1 =$ 0 0 1	$S_1 =$ 0 0 1	0	0
0 0 1	$S_3 =$ 0 1 1	$S_0 =$ 0 0 0		0
0 1 0	d d d	d d d	d	d
0 1 1				
1 0 0				
1 0 1				
1 1 0				
1 1 1				

0

1 $_1 =$ 0 0 1

1