Chapter 3: Sequential Devices

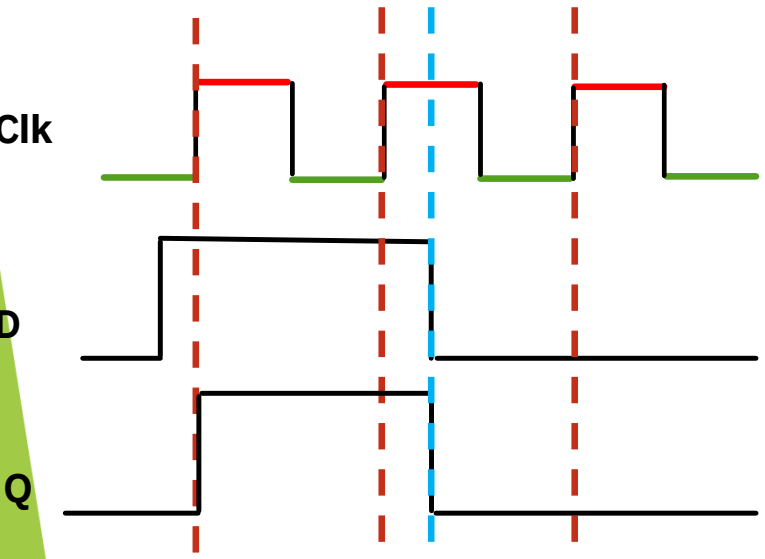
- Sequential circuits, unlike combinational ones whose outputs depend only on the present inputs, the outputs depend on present as well as past inputs. Sequential circuits would require memory while combinational circuits which are memoryless.
- The building blocks ,main devices, used in designing and implementing sequential circuits are Flop-Flops.

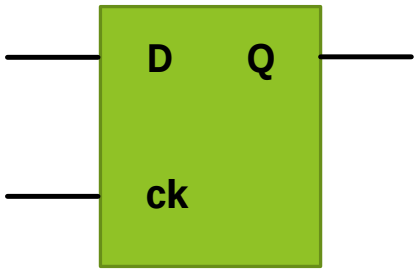
- In this chapter we will:

- Understand the design and working principles of Latches and Flip-Flops.
- Understand the importance and significance of sequential circuits in general.

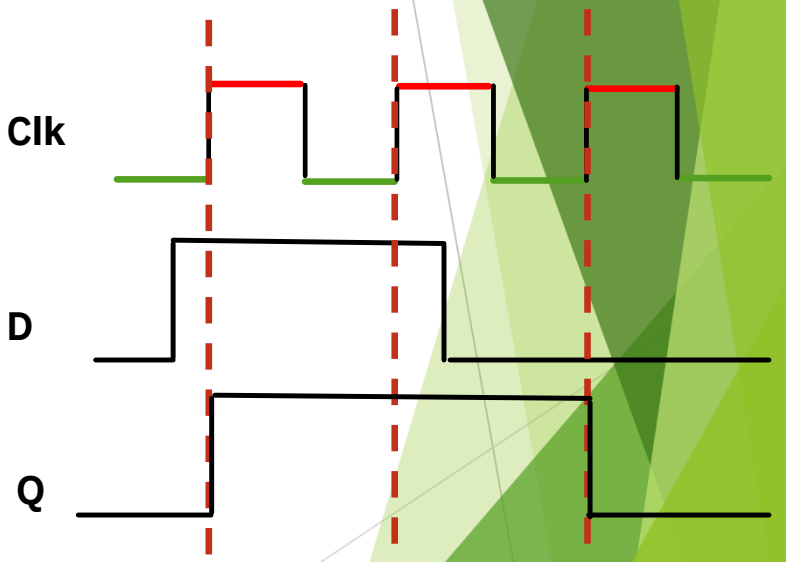
Latch vs Register (Flip-Flop)

Latch
Level Sensitive



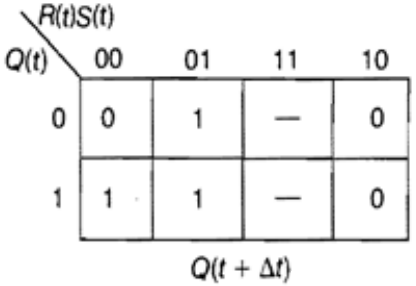


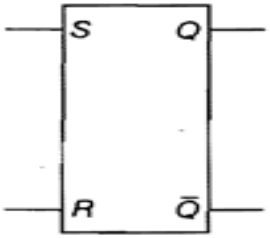
Flip-Flop
Edge Sensitive

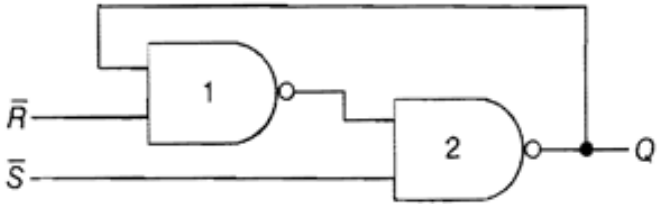


RS Latch

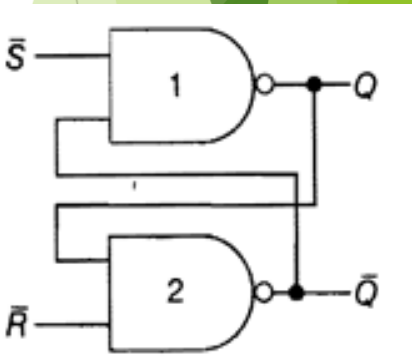
Inputs			Output
$R(t)$	$S(t)$	$Q(t)$	$Q(t + \Delta t)$
0	0	0	0
		1	1
0	1	0	1
		1	1
1	0	0	0
		1	0
1	1	0	—
		1	—







$$Q(t + \Delta t) = S(t) + \overline{R(t)}Q(t)$$
$$= \overline{\overline{S(t)} \cdot \overline{R(t)Q(t)}}$$



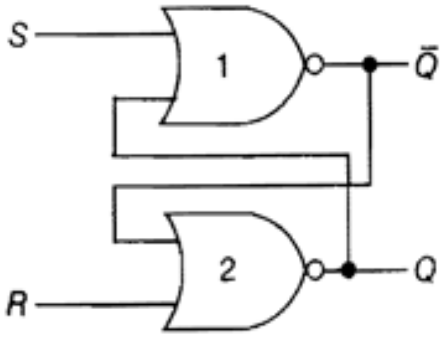
Let us assume the outputs of the forbidden state to be zeros

Q(t) \ R(t)S(t)	00	01	11	10
0	0	1	0	0
1	1	1	0	0

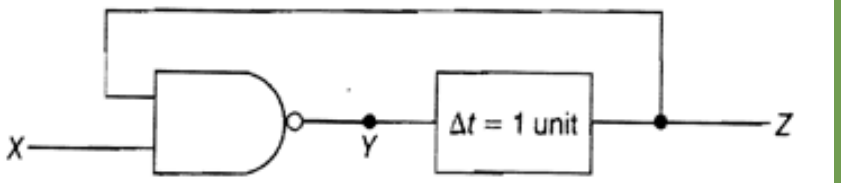
$Q(t + \Delta t)$

$$Q(t + \Delta t) = \overline{R}(t).[\,Q(t) + s(t)]$$

$$= \overline{R(t) + \overline{[Q(t) + S(t)]}}$$

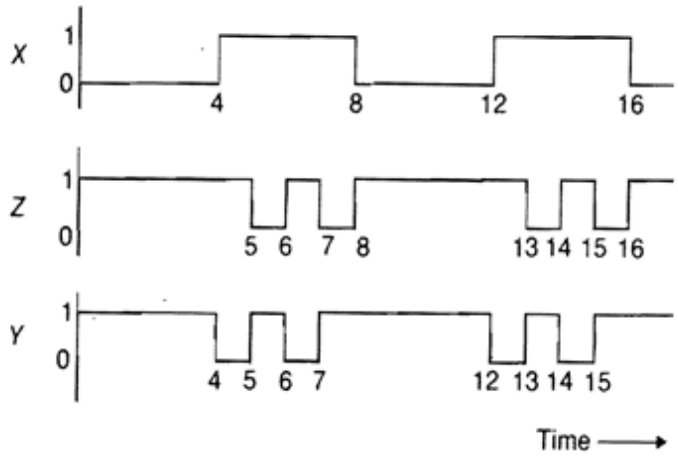


Example: Consider a NAND gate of total delay one unit, get z(t + Δt).



$$Z(t + \Delta t) = \overline{X(t)Z(t)} = \bar{X}(t) + \bar{Z}(t)$$

Lumped delay



When x changes from 0 to 1 the output oscillates with a period 2 Δt.
When x changes from 1 to 0 the output becomes 1 after a delay of Δt.

Distributed delay

$$Q(t + t_1) = \overline{\bar{S}(t) \cdot \bar{Q}(t)} = S(t) + Q(t)$$

$$\bar{Q}(t + t_2) = \overline{\bar{R}(t) \cdot Q(t)} = R(t) + \bar{Q}(t)$$

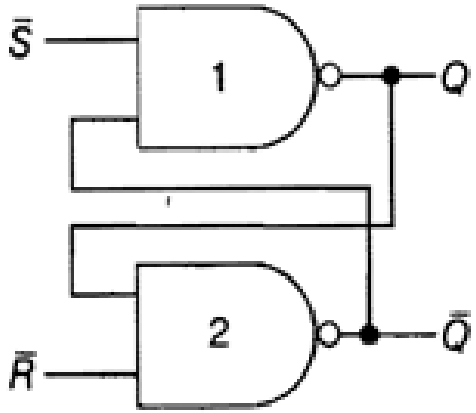
$$\bar{Q}(t + t_1 + t_2) = \overline{\bar{R}(t + t_1) \cdot Q(t + t_1)}$$

$$= \overline{\bar{R}(t + t_1)\,[S(t) + Q(t)]}$$

$$Q(t + t_1 + t_2) = \bar{R}(t + t_1)[S(t) + Q(t)]$$

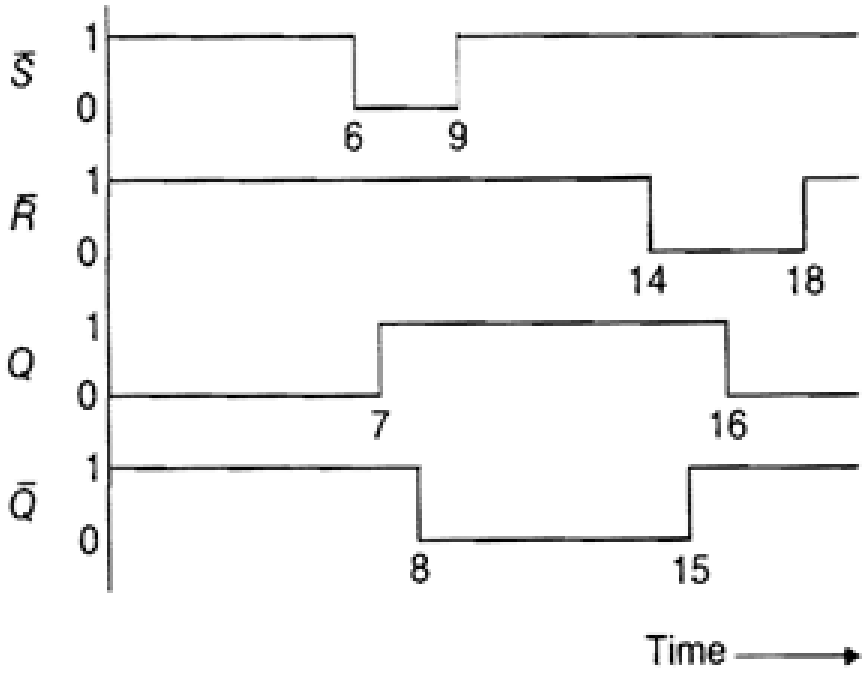
$t_1 = t_2 = \Delta t$

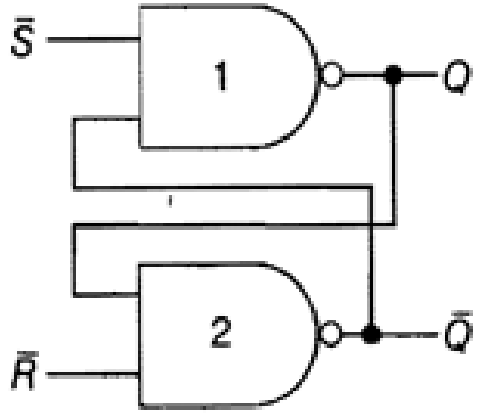
$$Q(t + \Delta t) = \bar{R}\left(t + \frac{\Delta t}{2}\right)[S(t) + Q(t)]$$



gate1 "t_1" and gate2 "t_2"

Timing diagram of the SR latch

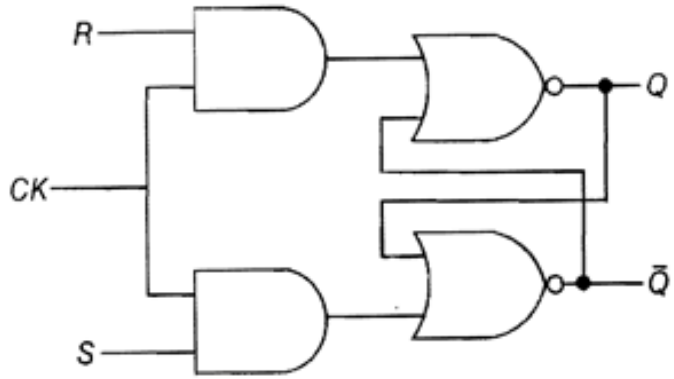


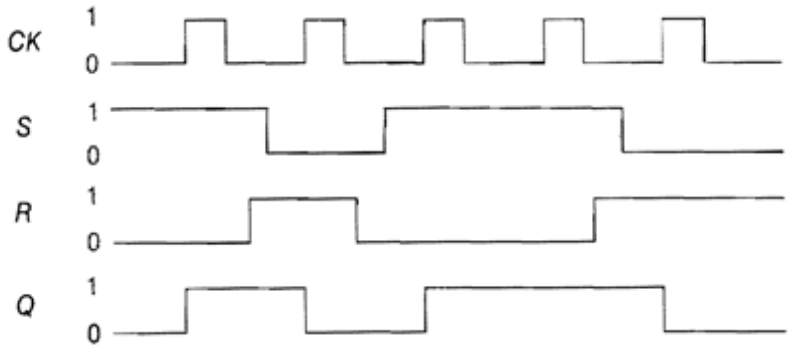


Clocked SR flip-flop

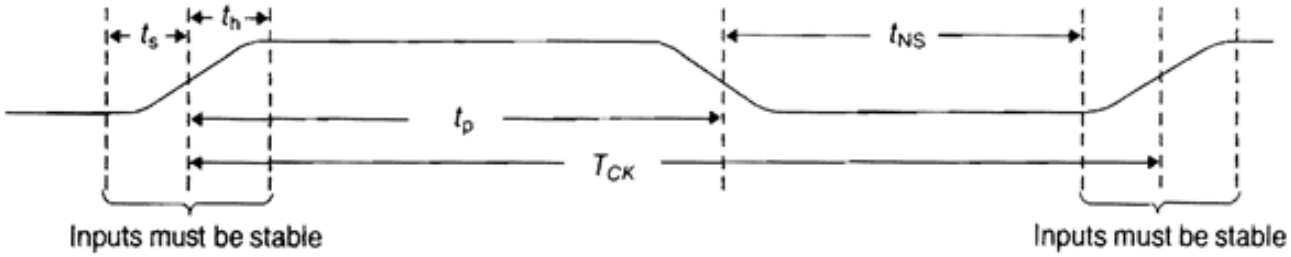
- Dependable operation of FFs must include clocks
- The Clock input forces an action to take place only when the clock is present, otherwise outputs are forced to remain unchanged

Inputs			Mode	Outputs	
$CK(t)$	$S(t)$	$R(t)$		$Q(t + \Delta t)$	$\bar{Q}(t + \Delta t)$
0	—	—	No action	$Q(t)$	$\bar{Q}(t)$
⎍	0	0	Hold	$Q(t)$	$\bar{Q}(t)$
⎍	0	1	Reset	0	1
⎍	1	0	Set	1	0
⎍	1	1	Invalid	—	—





Maximum Clock Rate



$$T_{ck} \geq t_s + t_p + t_{NS}$$

t_s : setup time, inputs have be there for some time before triggering

t_h : hold time, after triggering

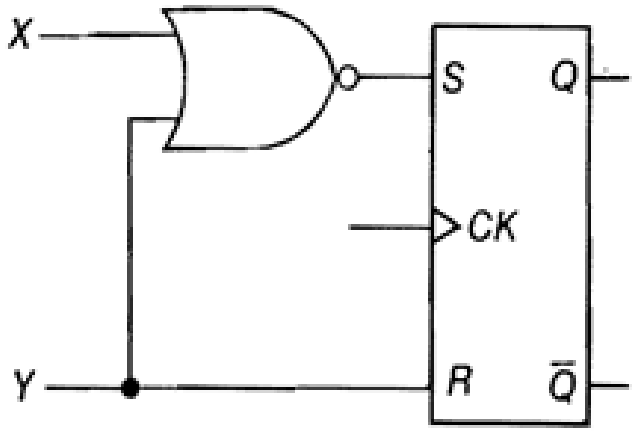
t_p : propagation delay

t_{NS} : Delay of Next-State Decoder

$$f_{CK} = \frac{1}{T_{CK}} \leq \frac{1}{t_s + t_p + t_{NS}}$$

Example: Modify the SR flip-flop to accept "11" as a RESET state.

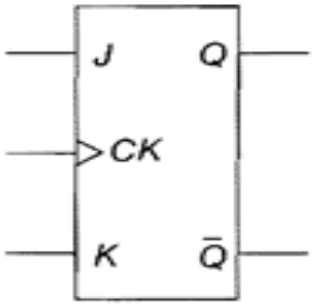
$X(t)$	$Y(t)$	$Q(t)$	$Q(t + \Delta t)$
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

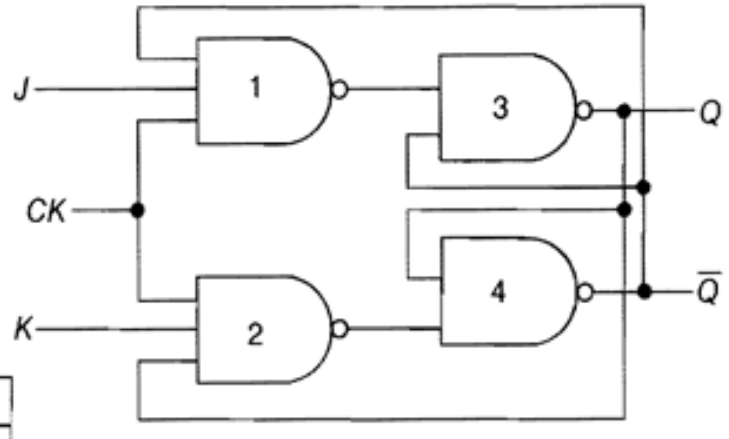


KM:

$$Q(t + \Delta t) = \bar{X}(t)\bar{Y}(t) + \bar{Y}(t)Q(t)$$

An alternative good way is to let the "11" state to make the FF toggle, JK flip-flop

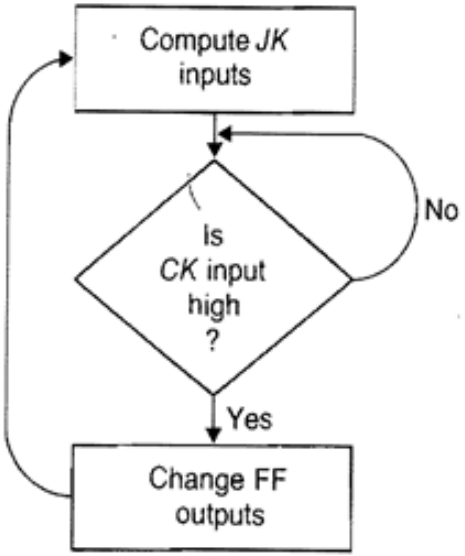


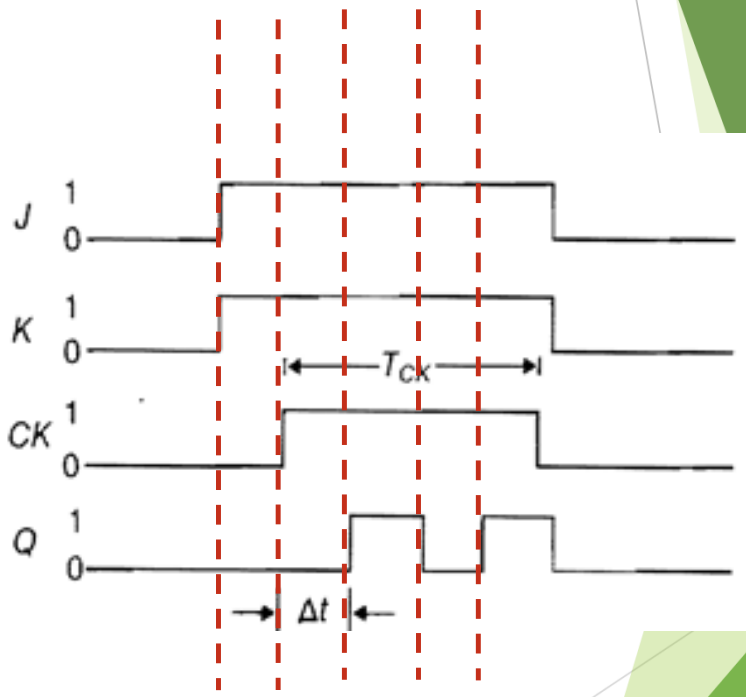


Mode	$J(t)$	$K(t)$	$Q(t)$	$Q(t + \Delta t)$
Hold	0	0	0	0
			1	1
Reset	0	1	0	0
			1	0
Set	1	0	0	1
			1	1
Toggle	1	1	0	1
			1	0

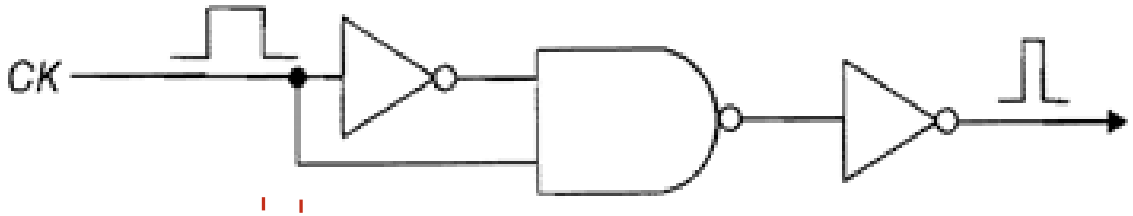
$$Q(t + \Delta t) = J(t)\overline{Q}(t) + \overline{K}(t)Q(t)$$

The clock period has to be less than the delay of the FF

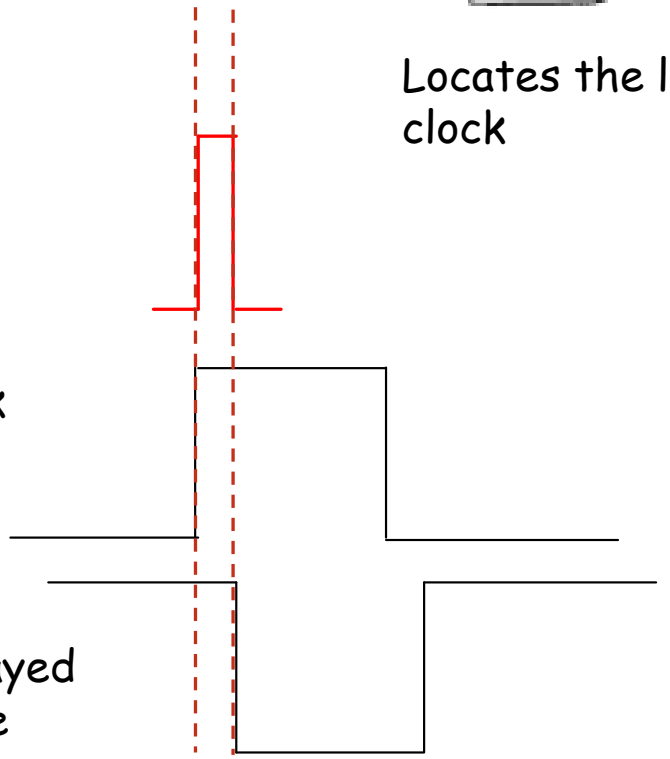




Pulse narrowing and edge-triggered FFs



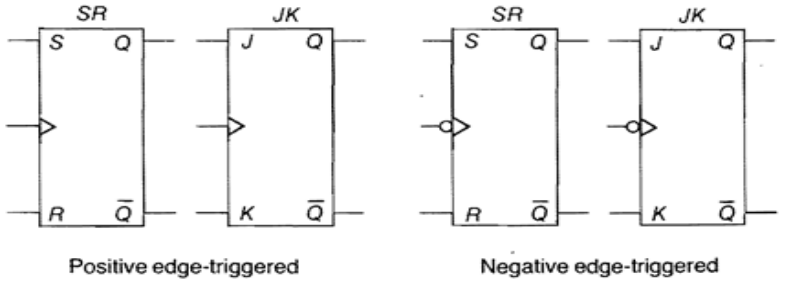
Locates the leading edge of the clock



True clock pulse

Inverted delayed clock pulse

Needed for edge-triggered FFs



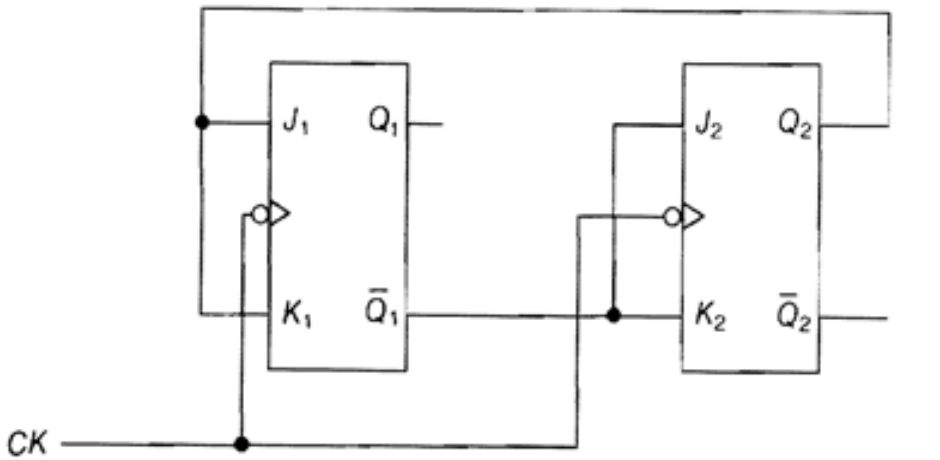
Positive edge-triggered

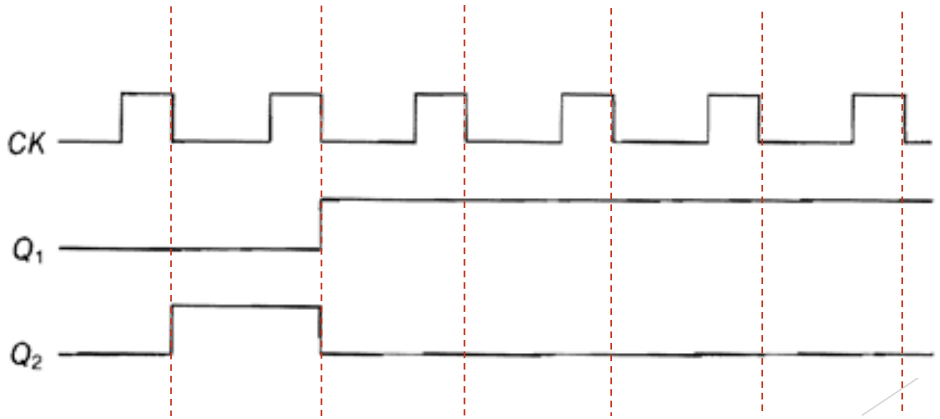
Negative edge-triggered

Example:Obtain the timing diagram for the sequential circuit shown for at least six clock cycles. Assume that $Q_1(0)\ Q_2(0) = 00$

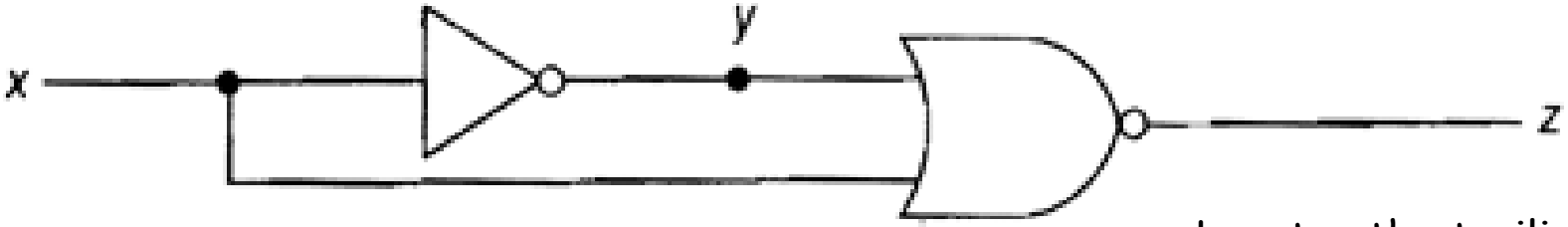
$J_1 = K_1 = Q_2$

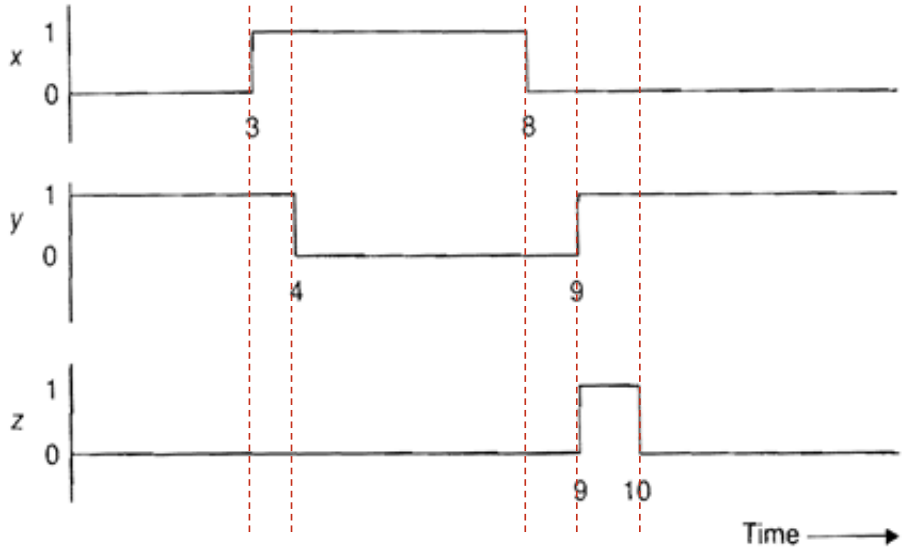
$J_2 = K_2 = \overline{Q_1}$





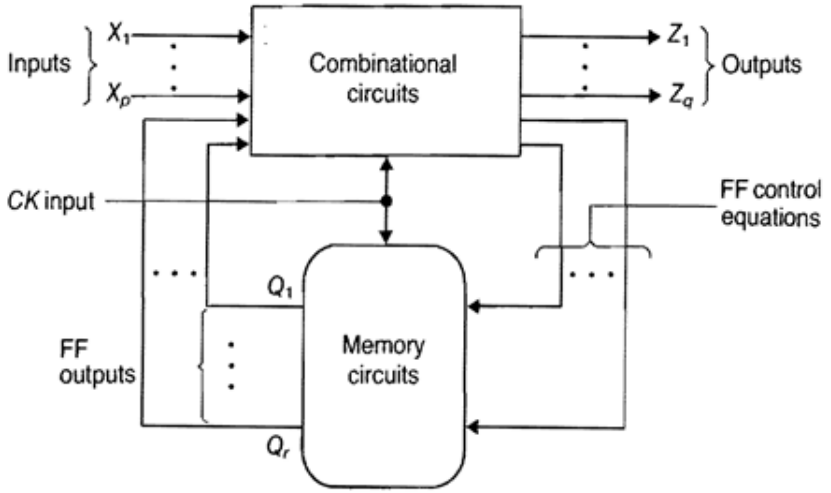
Obtain the response of the shown circuit, where each of the gates is assumed to have 1 unit of gate delay. The input x remains high for a duration longer than 5 units.



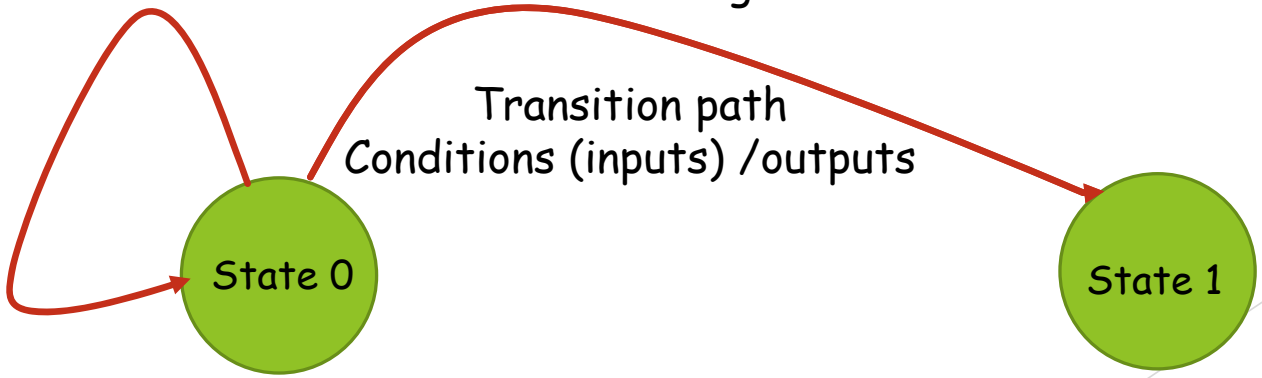


Locates the trailing edge of the clock

The memory circuit of sequential circuits is composed mainly of flip-flops.

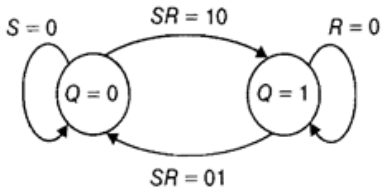


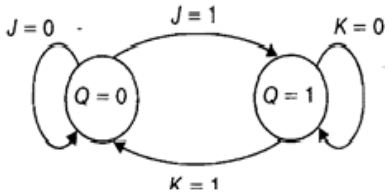
State diagram

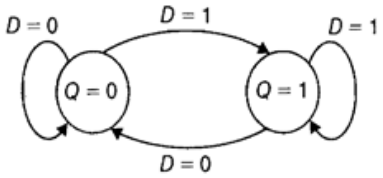


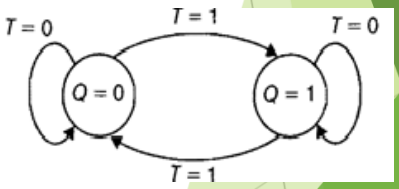
State diagrams of flip-flops

$Q(t)$	$Q(t + \Delta t)$	S	R	J	K	D	T
0	0	0	—	0	—	0	0
0	1	1	0	1	—	1	1
1	0	0	1	—	1	0	1
1	1	—	0	—	0	1	0

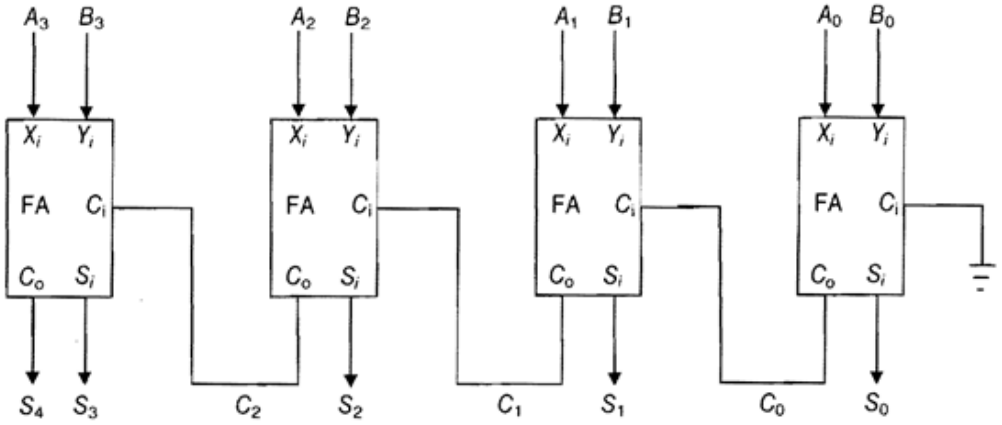




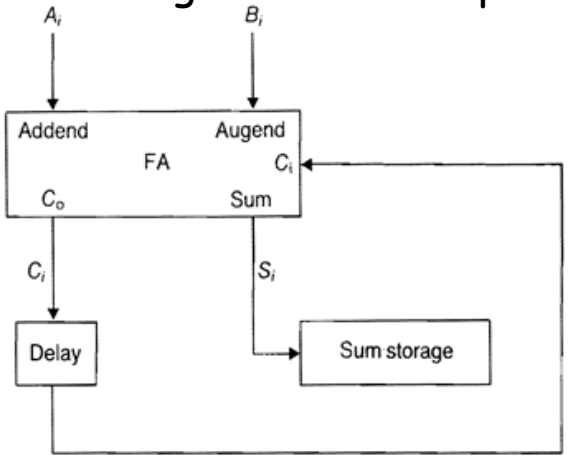




4-bit parallel (ripple) adder



n-bit serial adder implemented sequentially by one full adder, n-bit register and a flip-flop

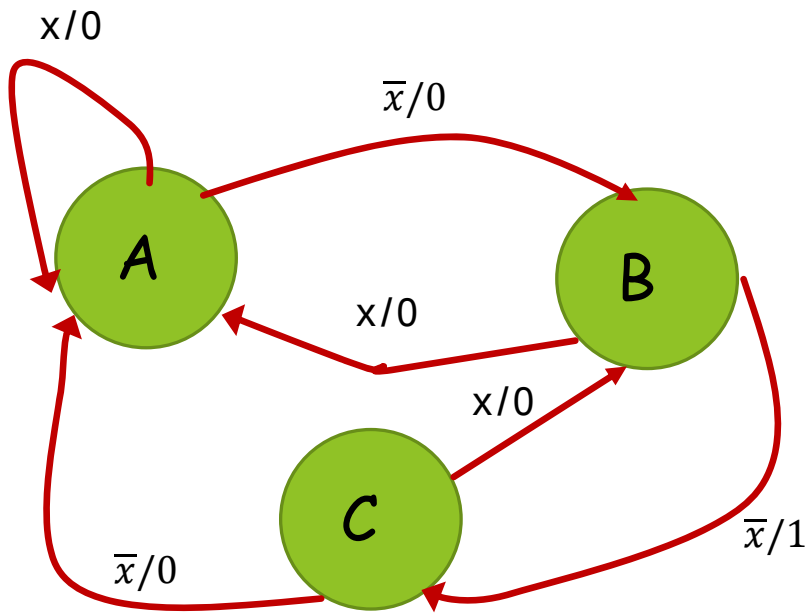


Chapter 4: Design of Synchronous Sequential Circuits

- By the end of this chapter we should be able to:

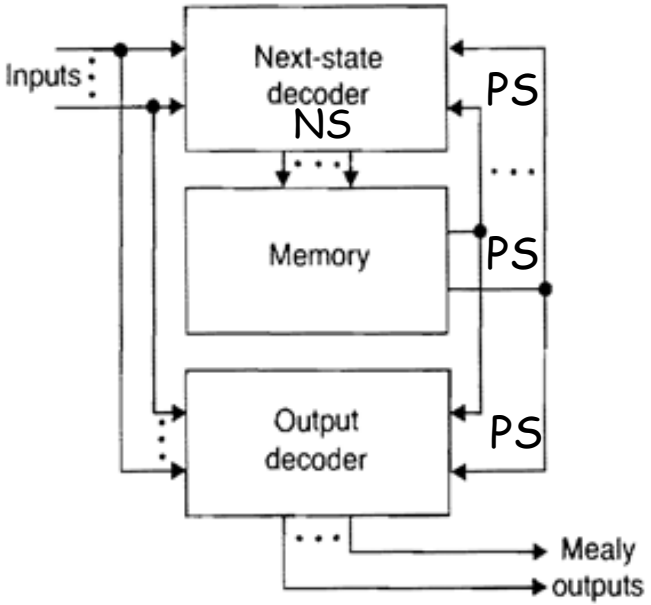
  - Differentiate between Mealy and Moore finite-state machines
  - Obtain a state diagram for a sequential circuit design
  - Follow the different design steps to realize the synchronous sequential machine

State Diagram and State Table

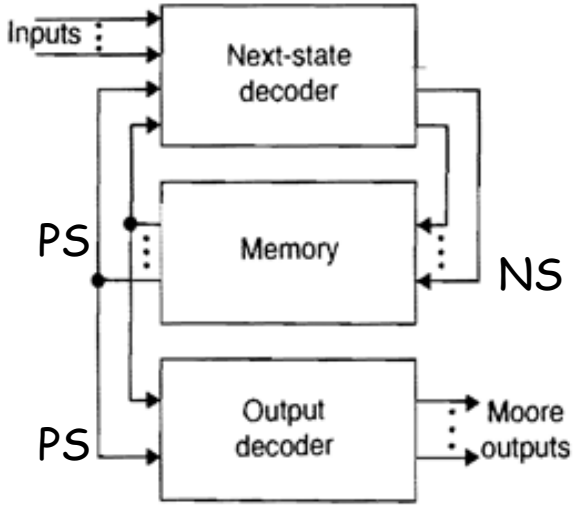


Present state PS	Next state NS, outputs	
	X = 0	X = 1
A	B, 0	A, 0
B	C,1	A,0
C	A, 0	B,0

Mealy and Moore machines

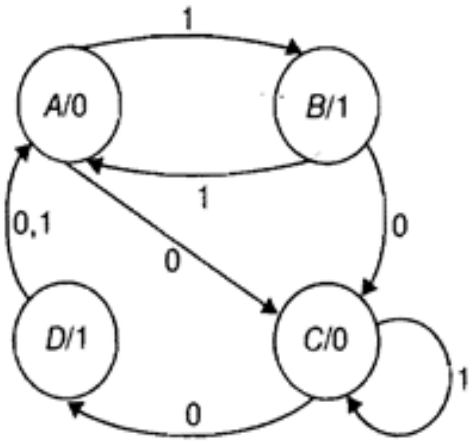


Outputs depend on both primary inputs and the present state



Outputs depend only on the present state

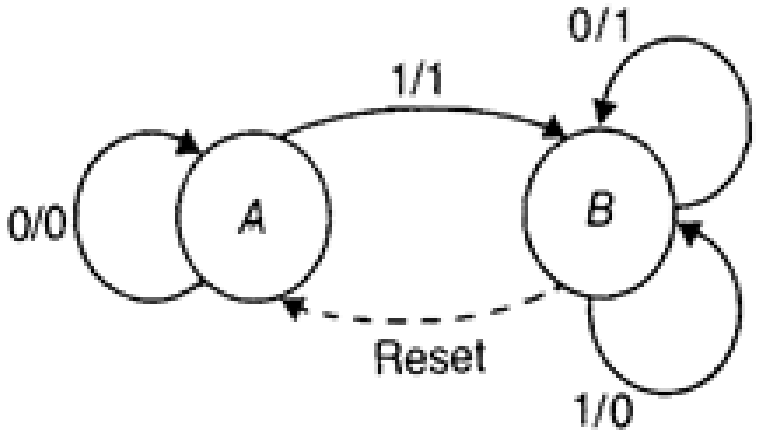
State Diagram and State Table



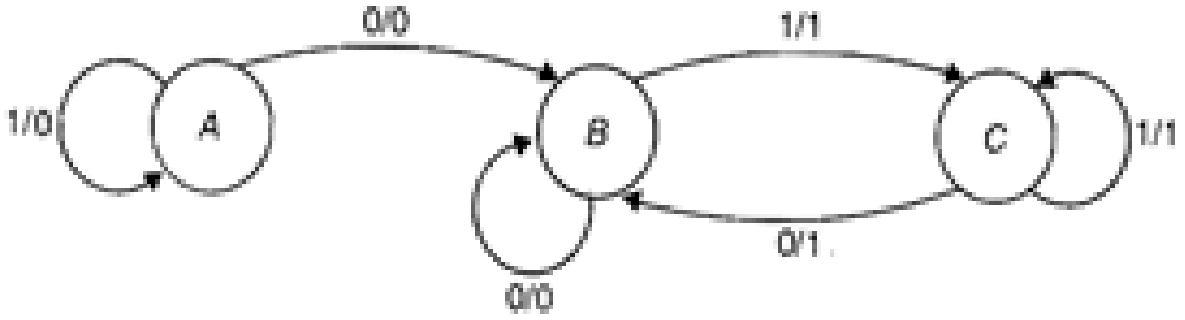
PS	NS		Z
	$x = 0$	$x = 1$	
A	C	B	0
B	C	A	1
C	D	C	0
D	A	A	1

Example: Obtain the state diagram of a controller for a serial machine that performs the 2's complement operation .

$$1100110101\overleftarrow{10000}$$



Example: Obtain the state table for synchronous sequential machine that detects a 01 sequence. The detection of sequence sets the output, Z=1, which is reset only by a 00 input sequence.



PS	NS, Z	
	$x = 0$	$x = 1$
A	B,0	A,0
B	B,0	C,1
C	B,1	C,1

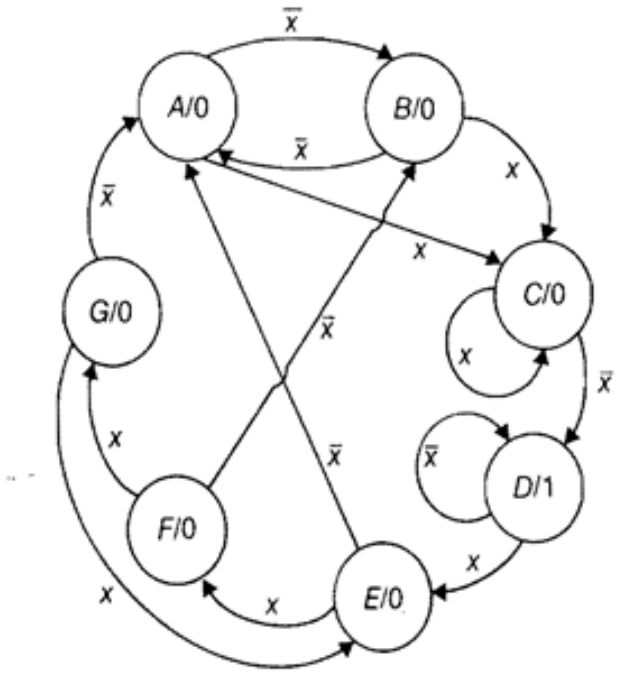
Example: Obtain the Moore equivalent state table for the Mealy machine

PS	NS, Z	
	$x = 0$	$x = 1$
A	C,0	A,0
B	B,0	A,0
C	D,1	C,1
D	D,0	B,0
E	C,1	A,0

PS	NS, Z	
	$x = 0$	$x = 1$
A	C',0	A,0
B	B,0	A,0
C'	D'',1	C'',1
C''	D'',1	C'',1
D'	D',0	B,0
D''	D',0	B,0
E	C'',1	A,0

PS	NS		Z
	$x = 0$	$x = 1$	
A	C'	A	0
B	B	A	0
C'	D''	C''	0
C''	D''	C''	1
D'	D'	B	0
D''	D'	B	1
E	C''	A	—

Eliminating Redundant States



PS	NS		Z
	$x = 0$	$x = 1$	
A	B	C	0
B	A	C	0
C	D	C	0
D	D	E	1
E	A	F	0
F	B	G	0
G	A	E	0

- Are any two states equivalent? So, do we have any redundant state?

❖ Equivalent States:

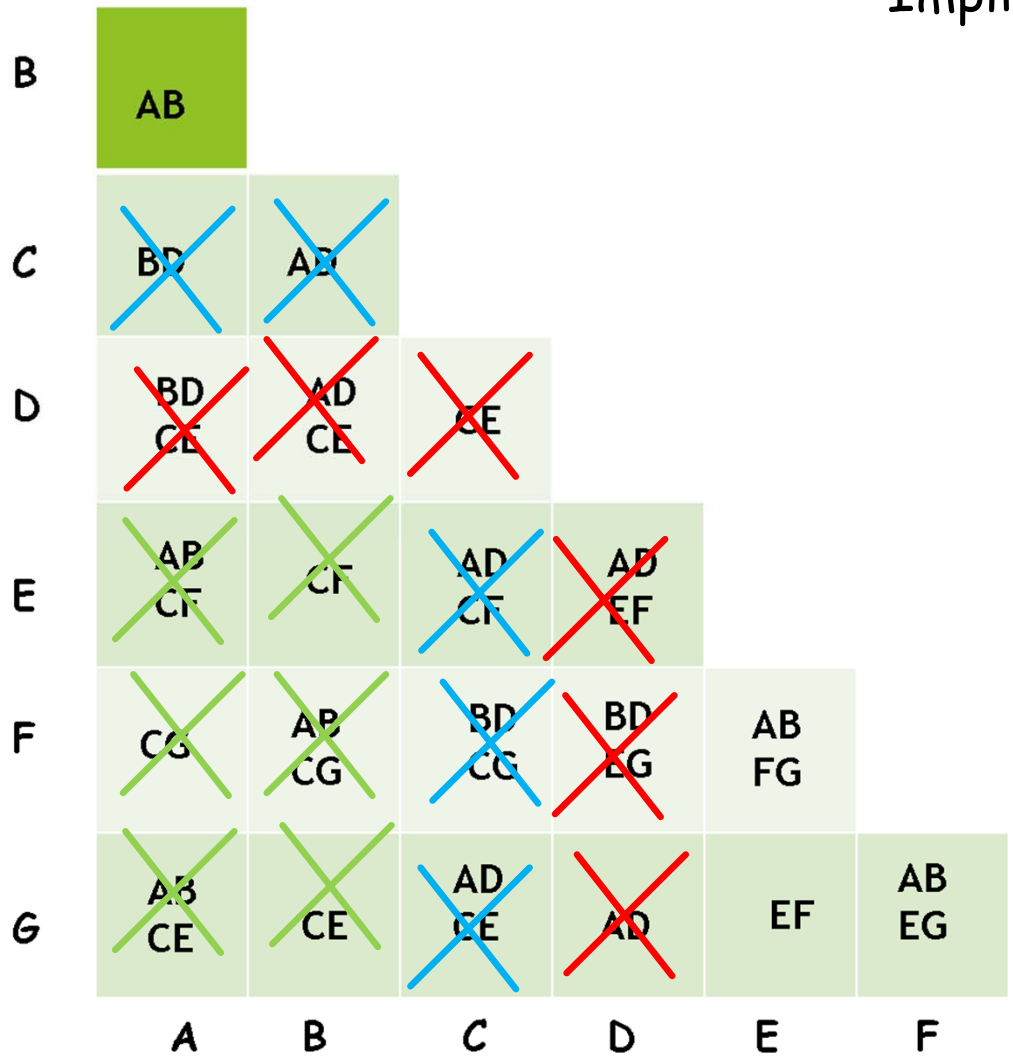
- Have the same output
- Make the same transition

Implication Table

	A	B	C	D	E	F
B	AB					
C	BD	AD				
D	BD CE	AD CE	CE			
E	AB CF	CF	AD CF	AD EF		
F	CG	AB CG	BD CG	BD EG	AB FG	
G	AB CE	CE	AD CE	AD	EF	AB EG

PS	NS		Z
	$x = 0$	$x = 1$	
A	B	C	0
B	A	C	0
C	D	C	0
D	D	E	1
E	A	F	0
F	B	G	0
G	A	E	0

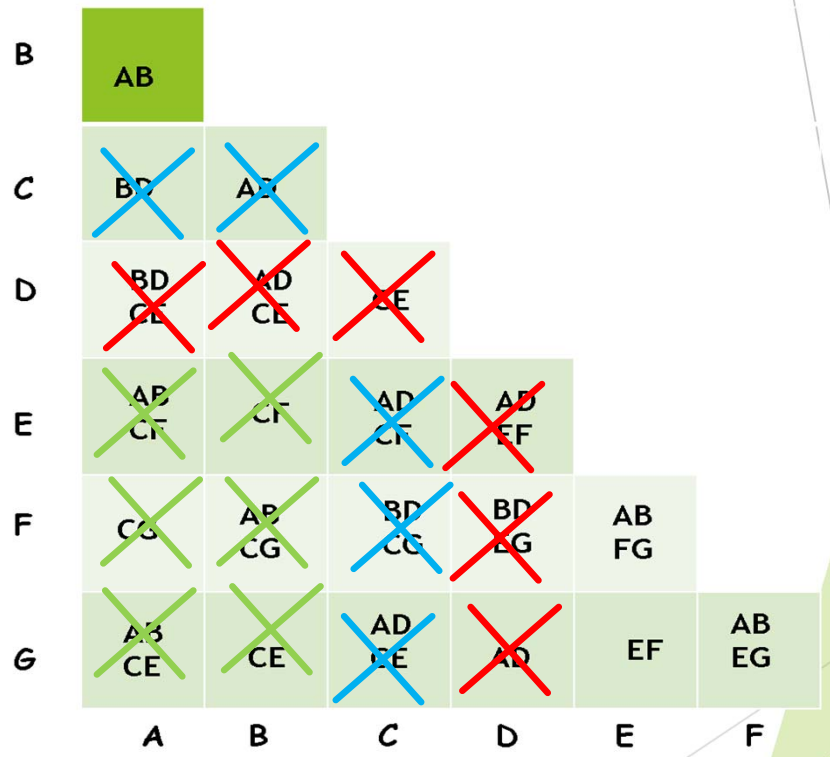
Implication Table



PS	NS		Z
	$x = 0$	$x = 1$	
A	B	C	0
B	A	C	0
C	D	C	0
D	D	E	1
E	A	F	0
F	B	G	0
G	A	E	0

Partition Table

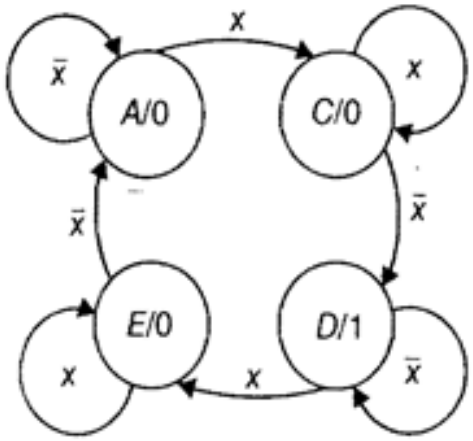
F	(FG)
E	(EFG)
D	(EFG)
C	(EFG)
B	(EFG)
A	(AB)(EFG)
	(AB)(C)(D)(EFG)



Reduced State Diagram and State Table

PS	NS		Z
	$x = 0$	$x = 1$	
A	B	C	0
B	A	C	0
C	D	C	0
D	D	E	1
E	A	F	0
F	B	G	0
G	A	E	0

(AB)(C)(D)(EFG)



PS	NS		Z
	$x = 0$	$x = 1$	
A	A	C	0
C	D	C	0
D	D	E	1
E	A	E	0

n: FFs
m: States
$m \leq 2^n$

State Transition Table

A = 00, C = 01, D = 11, and E = 10

PS	NS		
Q_1Q_2	$x = 0$	$x = 1$	Z
00	00	01	0
01	11	01	0
11	11	10	1
10	00	10	0

Excitation Maps and Design Equations

Q_1Q_2 / X	00	01	11	10
0	0	0	1	0
1	0	0	1	0

Z, output

$Z = Q_1Q_2$

Q_1Q_2 / X	00	01	11	10
0	0_	1_	_0	_1
1	0_	0_	_0	_0

J_1K_1

$J_1 = \overline{x}Q_2$

$K_1 = \overline{x}\overline{Q_2}$

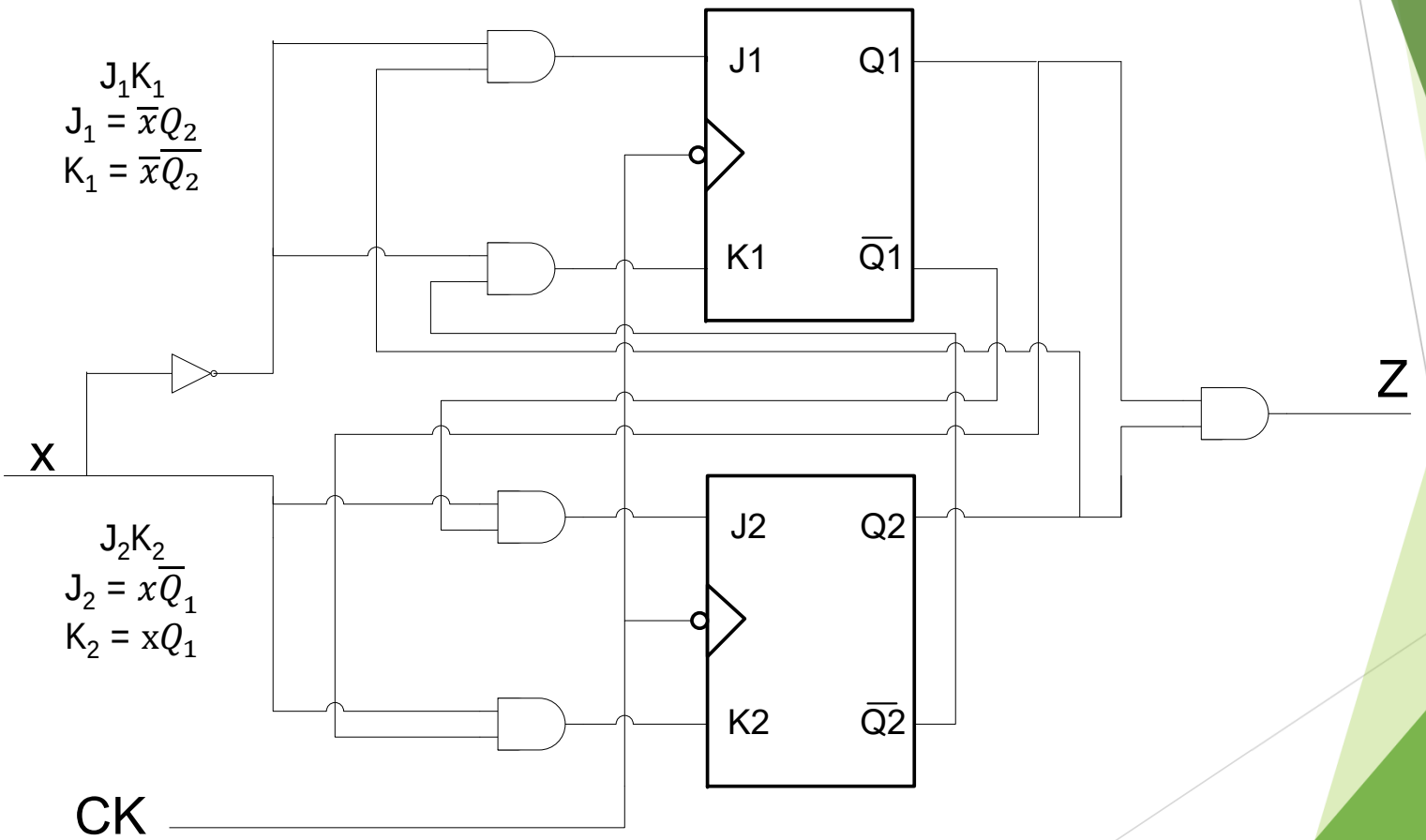
Q_1Q_2 / X	00	01	11	10
0	0_	_0	_0	0_
1	1_	_0	_1	0_

J_2K_2

$J_2 = x\overline{Q_1}$

$K_2 = xQ_1$

Hardware Implementation



Example: Find the HW implementation of the FSM whose state transition table is given below

PS	NS	
Q_1Q_2	$x = 0$	$x = 1$
00	01,0	00,0
01	11,1	00,0
11	00,0	01,0
10	--,-	--,-

x \ Q_1Q_2	00	01	11	10
0	0	1	0	--
1	0	0	0	--

Z

$$Z = \bar{Q}_1 Q_2 \bar{x} \cdot CK$$

x \ Q_1Q_2	00	01	11	10
0	0—	1—	—1	--
1	0—	0—	—1	--

$J_1 K_1$

$$J_1 = \bar{x} Q_2$$

$$K_1 = 1$$

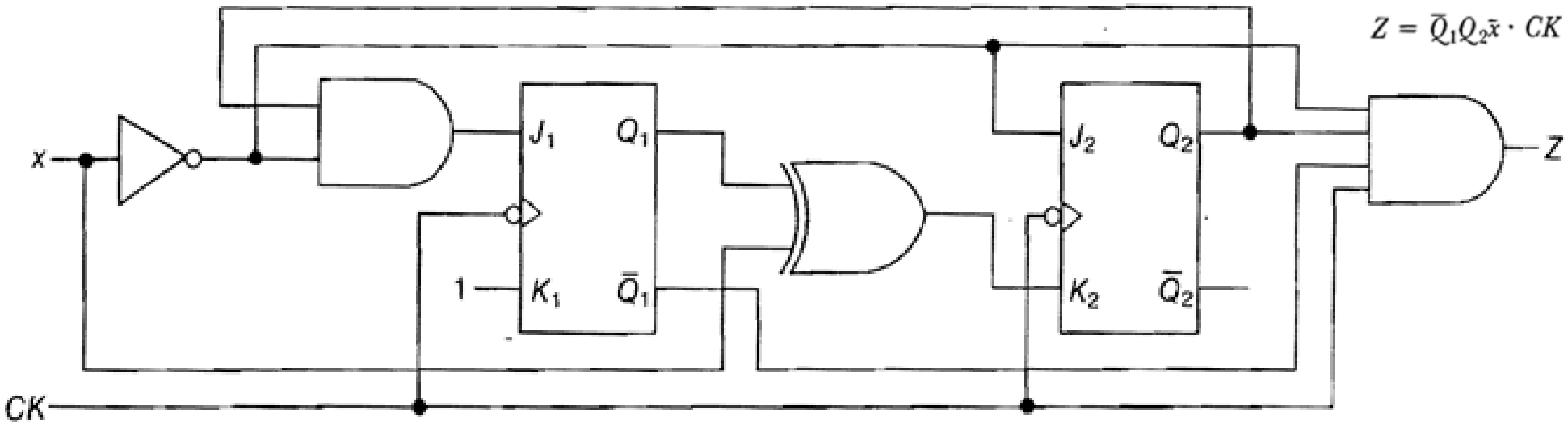
x \ Q_1Q_2	00	01	11	10
0	1—	—0	—1	--
1	0—	—1	—0	--

$J_2 K_2$

$$J_2 = \bar{x}$$

$$K_2 = \bar{Q}_1 x + Q_1 \bar{x} = Q_1 \oplus x$$

Hardware Implementation



$$J_1 = \bar{x}Q_2$$
$$K_1 = 1$$

$$J_2 = \bar{x}$$
$$K_2 = \bar{Q}_1 x + Q_1 \bar{x} = Q_1 \oplus x$$

Design Algorithm

Step 1. Obtain the state diagram from the word statement of the problem.

Step 2. Obtain the state table from the state diagram.

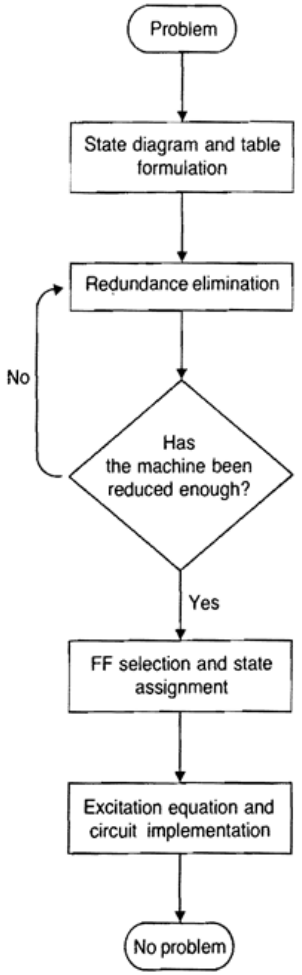
Step 3. Eliminate the redundant states.

Step 4. Make state assignments.

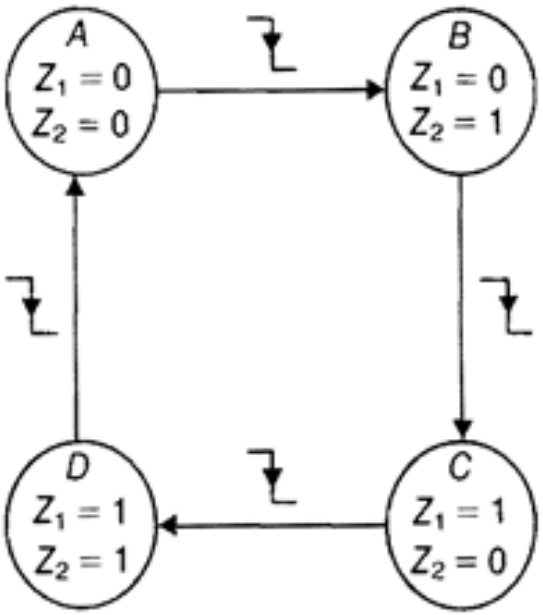
Step 5. Determine the type of FFs to use and obtain the corresponding excitation maps.

Step 6. Determine the output and FF equations.

Step 7. Construct the logic circuit.

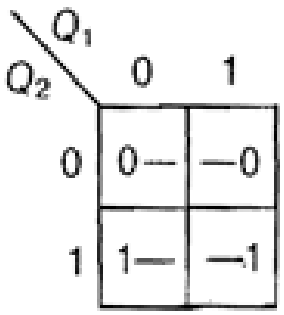


Design a two-bit clocked sequential counter circuit that counts clock pulses.



PS	NS	Z_1Z_2
A	B	00
B	C	01
C	D	10
D	A	11

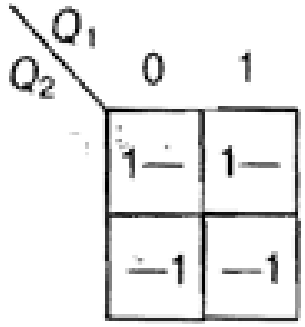
PS		
Q_1Q_2	NS	Z_1Z_2
00	01	00
01	10	01
10	11	10
11	00	11



J_1K_1

$J_1 = Q_2$

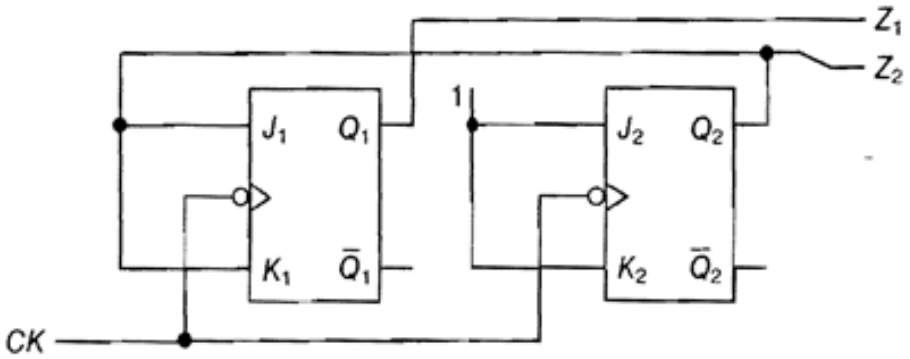
$K_1 = Q_2$



J_2K_2

$J_2 = 1$

$K_2 = 1$



Repeat the design of Example 7.5 by assigning $A = 00$, $B = 01$, $C = 11$, and $D = 10$. Construct the corresponding timing diagram as well.

PS Q_1Q_2	NS	Z_1Z_2
00	01	00
01	11	01
11	10	10
10	00	11

Q_2 \ Q_1	0	1
0	0—	—1
1	1—	—0

J_1K_1

$J_1 = Q_2$
$K_1 = \bar{Q}_2$

Q_2 \ Q_1	0	1
0	1—	0—
1	—0	—1

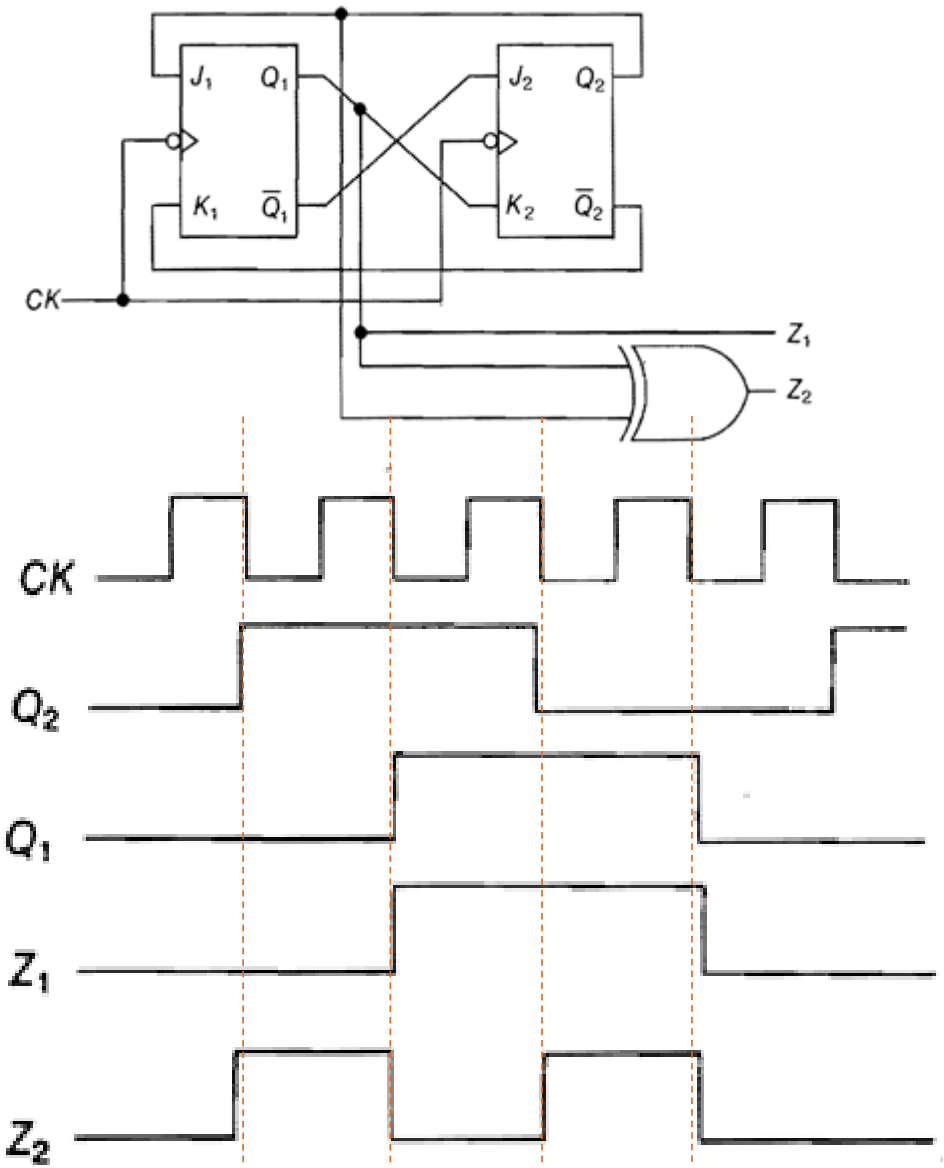
J_2K_2

$J_2 = \bar{Q}_1$
$K_2 = Q_1$

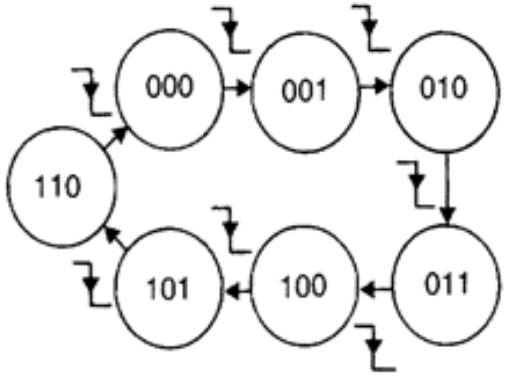
Q_2 \ Q_1	0	1
0	00	11
1	01	10

Z_1Z_2

$Z_1 = Q_1$
$Z_2 = Q_1\bar{Q}_2 + \bar{Q}_1Q_2 = Q_1 \oplus Q_2$


J1
Q1
K1
Q̄1
J2
Q2
K2
Q̄2
CK
Z1
Z2
CK
Q2
Q1
Z1
Z2

Obtain a scale-of-seven up-counter, as shown in the state diagram of Figure 7.31, using *D* FFs and PLA. Assume that the counter is tied to a seven-segment display device.



PS $Q_3Q_2Q_1$	NS
000	001
001	010
010	011
011	100
100	101
101	110
110	000

Q_1 \ Q_3Q_2	00	01	11	10
0	0	0	0	1
1	0	1	—	1

D_3

$$D_3 = \bar{Q}_2Q_3 + Q_1Q_2$$

Q_1 \ Q_3Q_2	00	01	11	10
0	0	1	0	0
1	1	0	—	1

D_2

$$D_2 = Q_1\bar{Q}_2 + \bar{Q}_1Q_2\bar{Q}_3$$

Q_1 \ Q_3Q_2	00	01	11	10
0	1	1	0	1
1	0	0	—	0

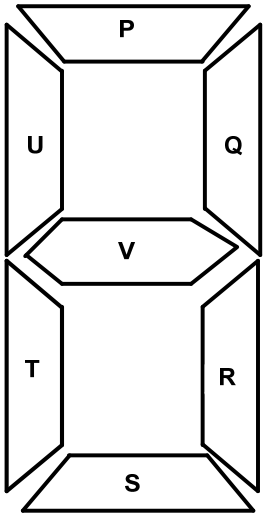
D_1

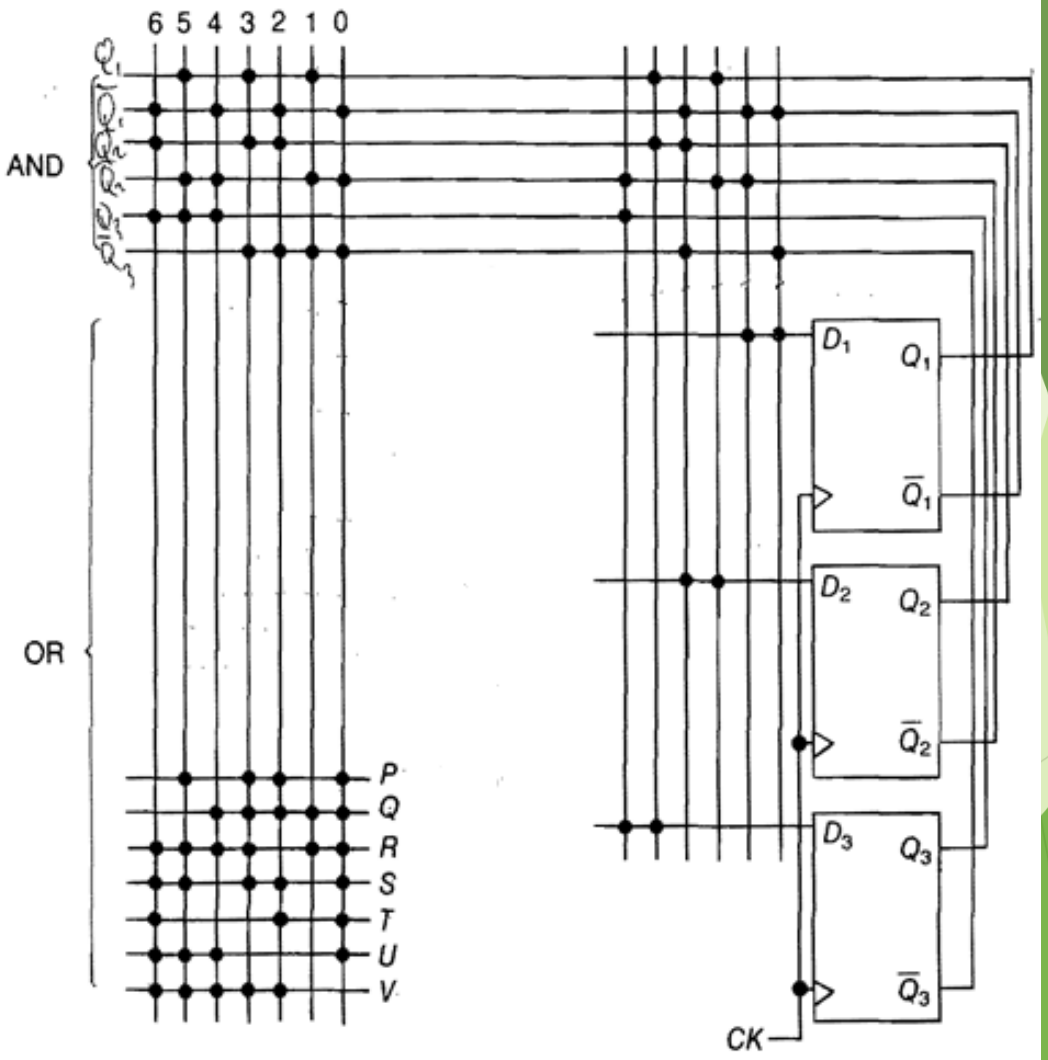
$$D_1 = \bar{Q}_1\bar{Q}_2 + \bar{Q}_1\bar{Q}_3$$

$$D_1 = \bar{Q}_1\bar{Q}_2 + \bar{Q}_1\bar{Q}_3$$

$$D_2 = Q_1\bar{Q}_2 + \bar{Q}_1Q_2\bar{Q}_3$$

$$D_3 = \bar{Q}_2Q_3 + Q_1Q_2$$





Chapter 5: Introduction to Counters and Registers

- Digital Systems usually have two main units:
  - A unit for data manipulation ⇒ REGISTERS and logic circuits
  - A second unit for regulating events of the first unit ⇒ COUNTERS

- Counters:
  - Synchronous
  - Asynchronous
- Registers:
  - Serial
  - Parallel

Synchronous Binary Counters

Single-bit counter (0, 1, 0, 1,….) $J = K = 1$

Two-bit counter (00, 01, 10, 11, 00,……)

$J_1 = K_1 = 1$
$J_2 = K_2 = Q_1$

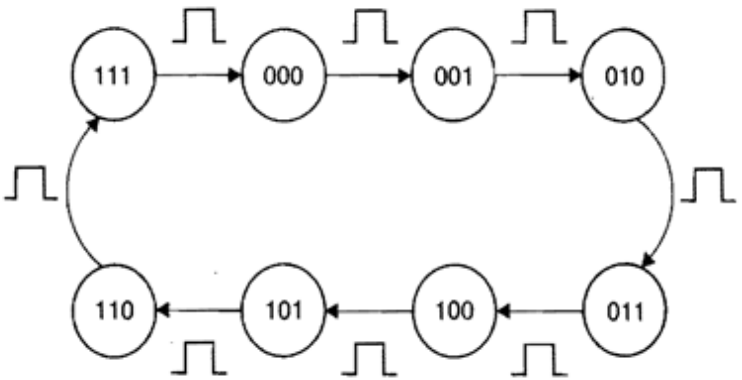
Likewise for three-bit counter

$J_1 = K_1 = 1$
$J_2 = K_2 = Q_1$
$J_3 = K_3 = Q_2Q_1$

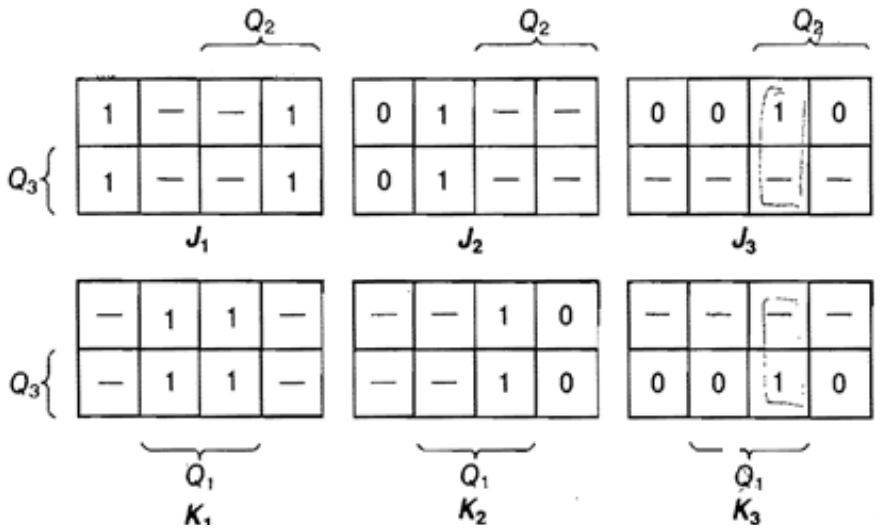
For n-bit counter

$$J_n = Q_{n-1}Q_{n-2}\cdots Q_3Q_2Q_1 = Q_{n-1}J_{n-1}$$
$$K_n = Q_{n-1}Q_{n-2}\cdots Q_3Q_2Q_1 = Q_{n-1}K_{n-1}$$

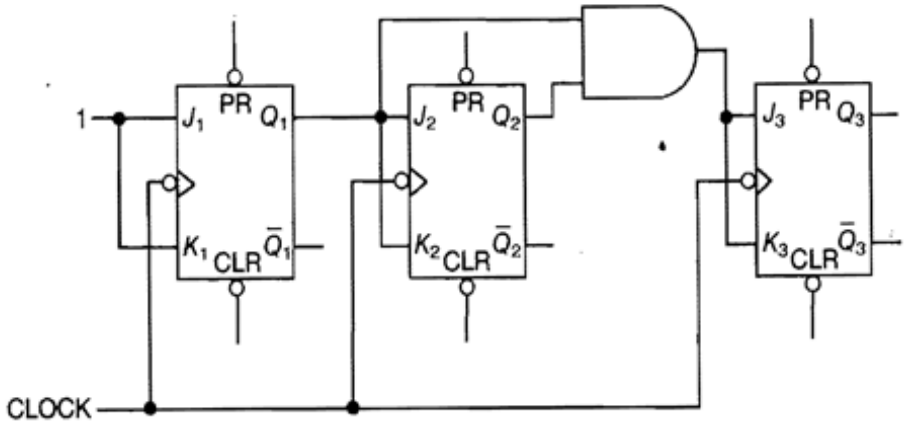
Design a three-bit counter



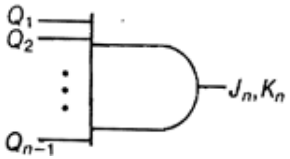
PS	
$Q_3Q_2Q_1$	NS
000	001
001	010
010	011
011	100
100	101
101	110
110	111
111	000



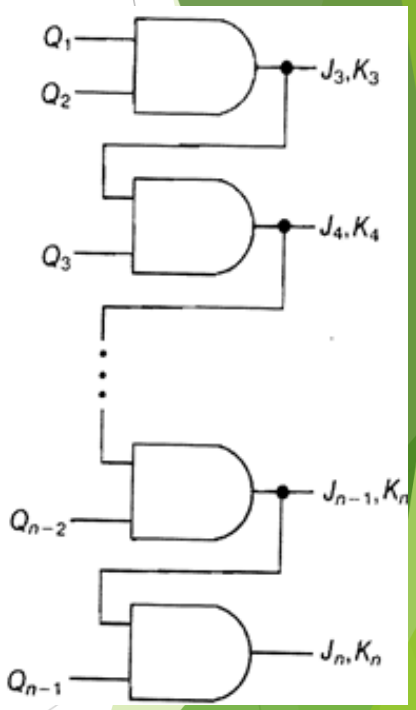
$J_1 = K_1 = 1$
$J_2 = K_2 = Q_1$
$J_3 = K_3 = Q_2Q_1$



Delay and fan-in problems

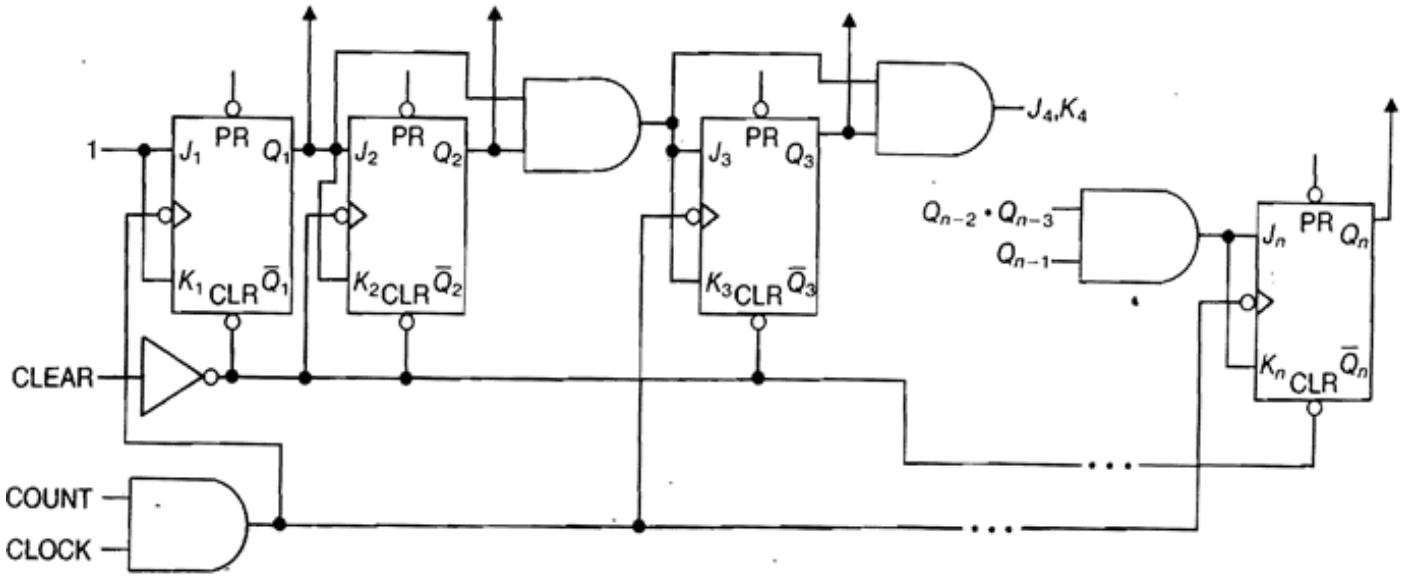


- Delay is the same for all stages
- Fan-in problem



- Delay increases with stages
- Fan-in is always two.

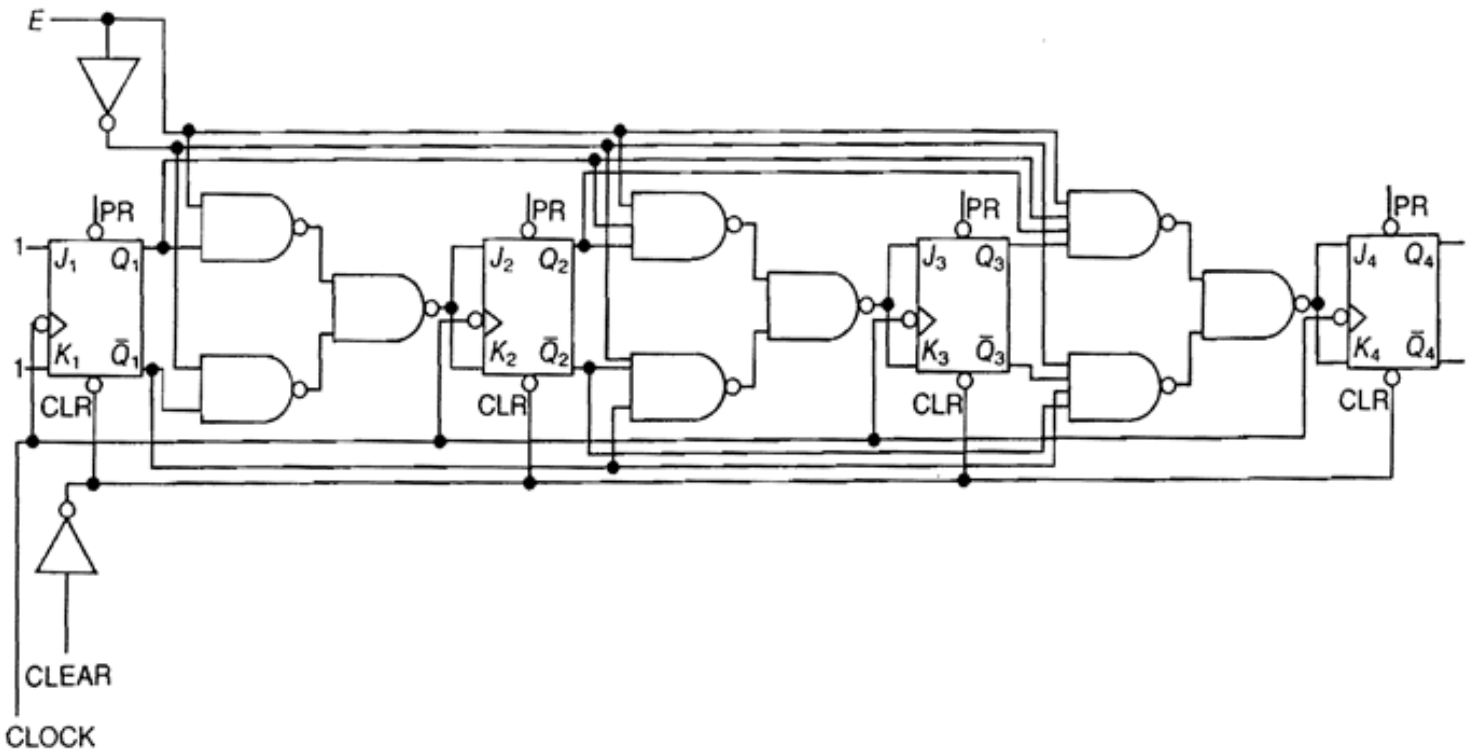
Standard n-bit up counter



Up Down-Counter

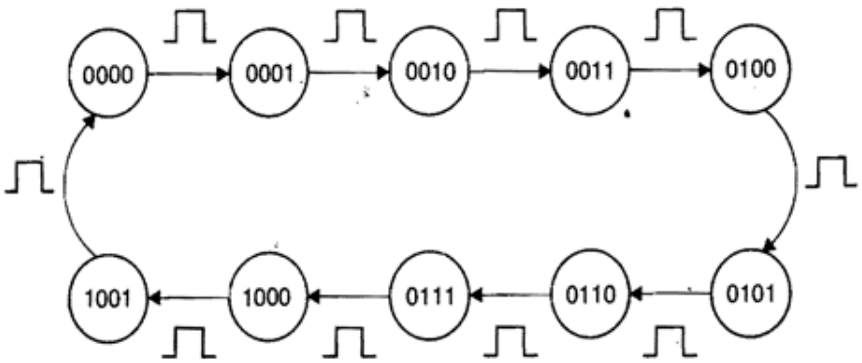
$$J_n = K_n = EQ_{n-1}Q_{n-2} \cdots Q_2Q_1 + \bar{E}\bar{Q}_{n-1}\bar{Q}_{n-2} \cdots \bar{Q}_2\bar{Q}_1$$

E = 1 for up counting, and E = 0 for down counting

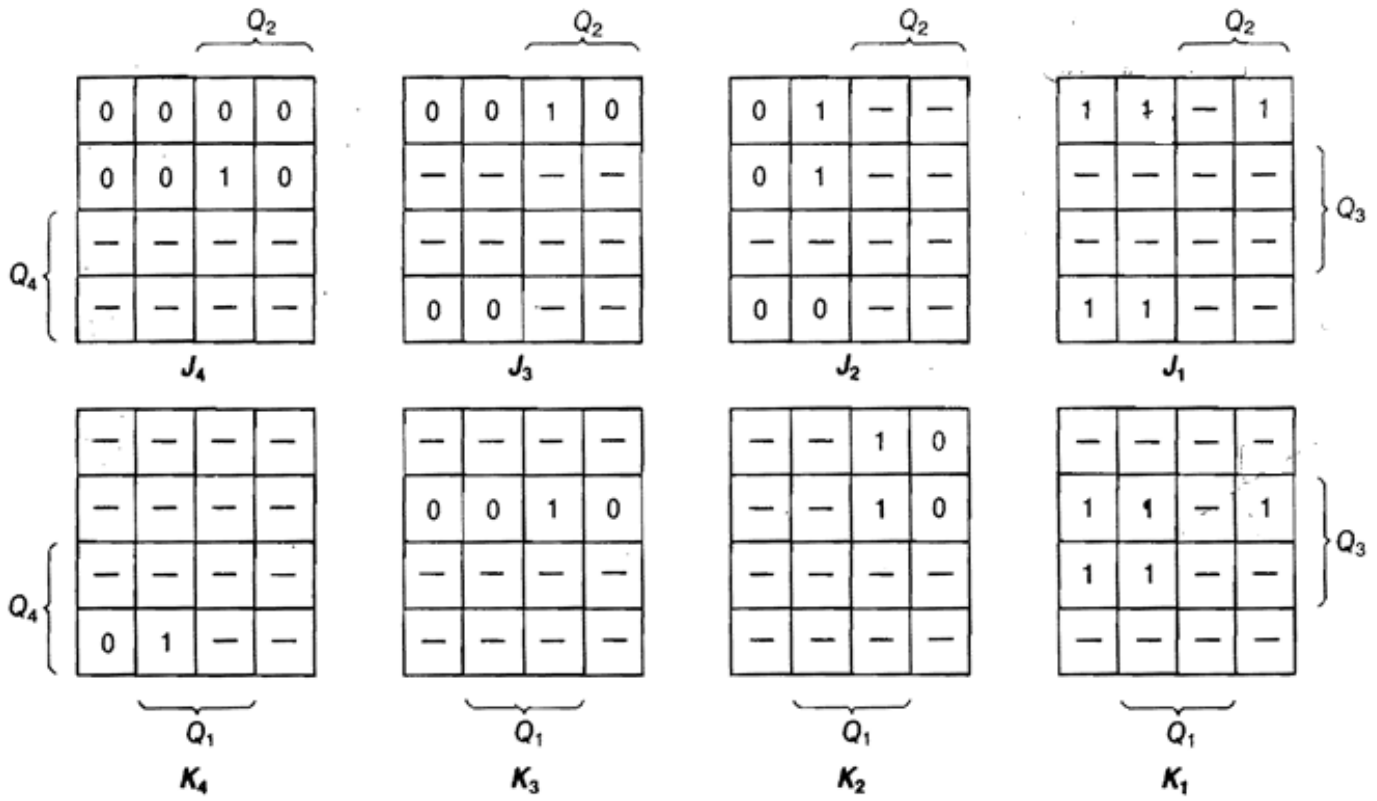


E = 1 for up counting, and E = 0 for down counting

Obtain the J and K equations for a BCD up-counter.



PS $Q_4Q_3Q_2Q_1$	NS	J_4K_4	J_3K_3	J_2K_2	J_1K_1
0000	0001	0—	0—	0—	1—
0001	0010	0—	0—	1—	—1
0010	0011	0—	0—	—0	1—
0011	0100	0—	1—	—1	—1
0100	0101	0—	—0	0—	1—
0101	0110	0—	—0	1—	—1
0110	0111	0—	—0	—0	1—
0111	1000	1—	—1	—1	—1
1000	1001	—0	0—	0—	1—
1001	0000	—1	0—	0—	—1



$J_4 = Q_1Q_2Q_3$

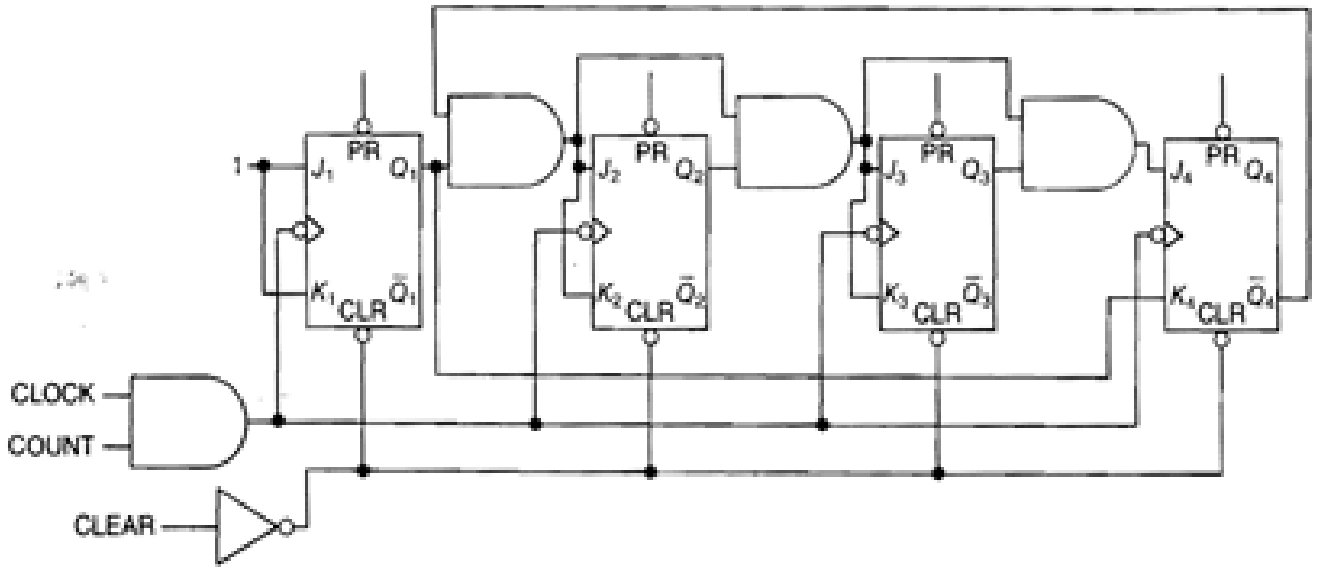
$K_4 = Q_1$

$J_3 = K_3 = Q_1Q_2$

$J_2 = Q_1\overline{Q}_4$

$K_2 = Q_1$

$J_1 = K_1 = 1$



BCD Counter

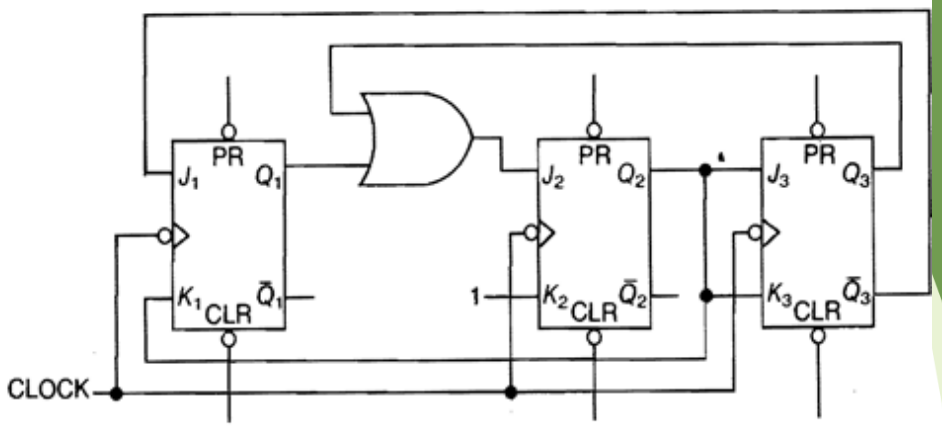
From a regular 4-bit counter the BCD one Can be thought of as such:

1001 should switch to
0000 and not to
1010

Q_4, the MSB, should become a 0,
Q_2 should be prevented from becoming a 1.

Q_1 should be fed to K_4,
K_4 and J_4 should not be connected,
$Q_1\bar{Q}_4$ should be fed to J_2 and K_2.

Obtain the state table for the counter shown in Figure 10.10, starting from the count 000. The MSB of the count is at Q_3.



$J_1 = \overline{Q_3}$, $K_1 = Q_2$

$J_2 = Q_1 + Q_3$ $K_2 = 1$

$J_3 = K_3 = Q_2$

PS	J_3K_3	J_2K_2	J_1K_1	NS
000	00	01	10	001
001	00	11	10	011
011	11	11	11	100
100	00	11	00	110
110	11	11	01	000

1
3
4
6
0

0, 1, 3, 4, 6, 0, ...

Obtain a synchronous counter that produces the count sequence 0, 2, 4, 3, 6, 7, 0, ...

$J_1 = Q_3$

$K_1 = 1$

$J_2 = 1$

$K_2 = \overline{Q_1 \oplus Q_3}$

$J_3 = Q_2$

$K_3 = Q_1 + \overline{Q}_2$

PS	NS			
$Q_3Q_2Q_1$	$Q_3Q_2Q_1$	J_3K_3	J_2K_2	J_1K_1
000	010	0—	1—	0—
001	— —	— —	— —	— —
010	100	1—	—1	0—
011	110	1—	—0	—1
100	011	—1	1—	1—
101	— — —	— —	— —	— —
110	111	—0	—0	1—
111	000	—1	—1	—1

Q_2Q_1

Q_3	00	01	11	10
0	0	-	-	0
1	1	-	-	1

J_1

	00	01	11	10
0	1	-	-	-
1	1	-	-	-

J_2

	00	01	11	10
0	0	-	1	1
1	-	-	-	-

J_3

Q_3	00	01	11	10
0	-	-	1	-
1	-	-	1	-

K_1

	00	01	11	10
0	-	-	0	1
1	-	-	1	0

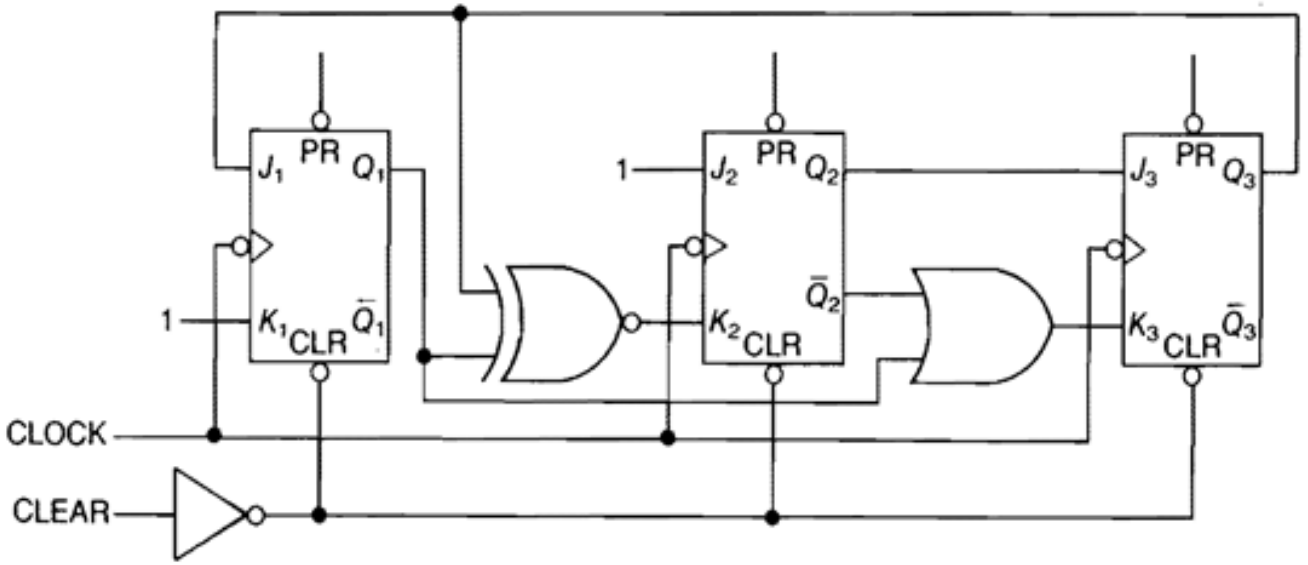
K_2

	00	01	11	10
0	-	-	-	-
1	1	-	1	0

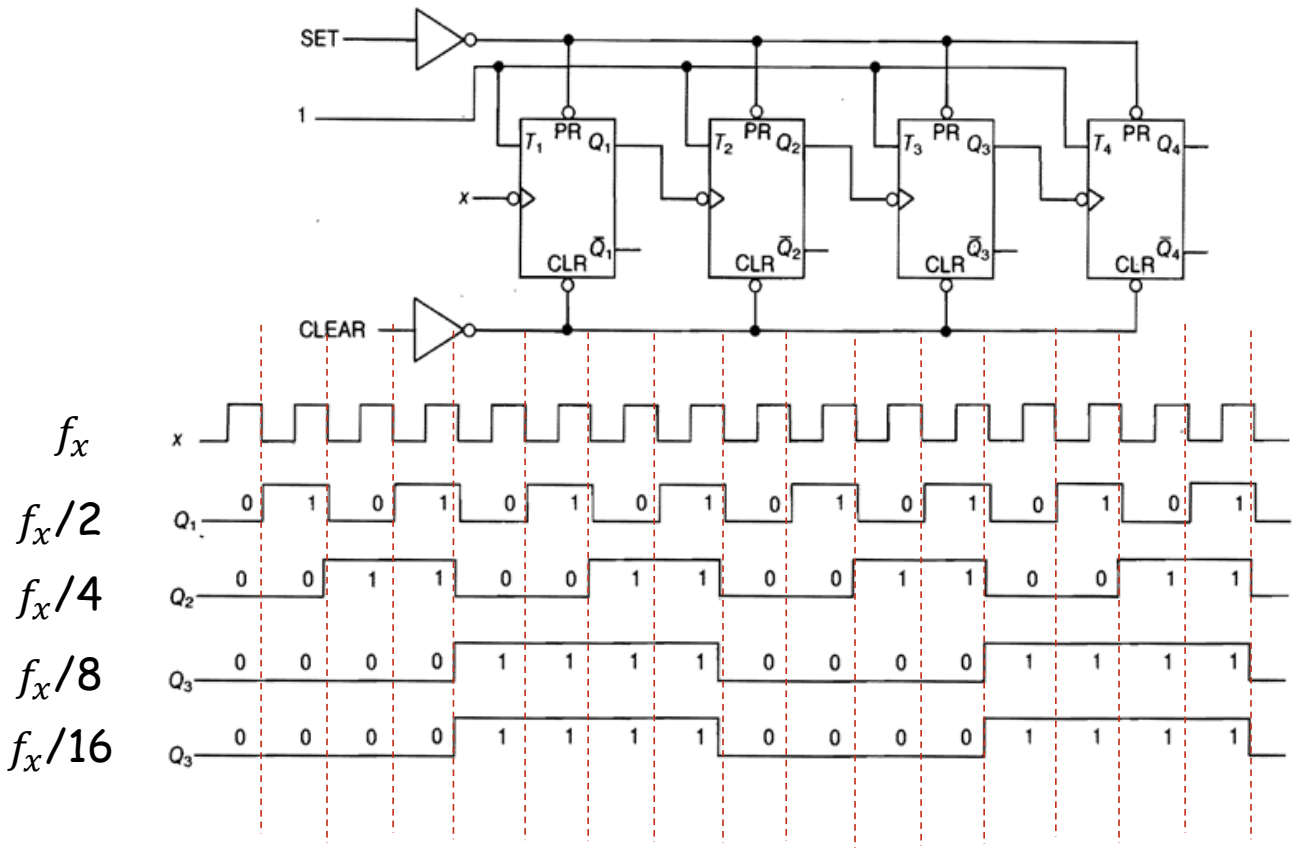
K_3

Hardware Implementation

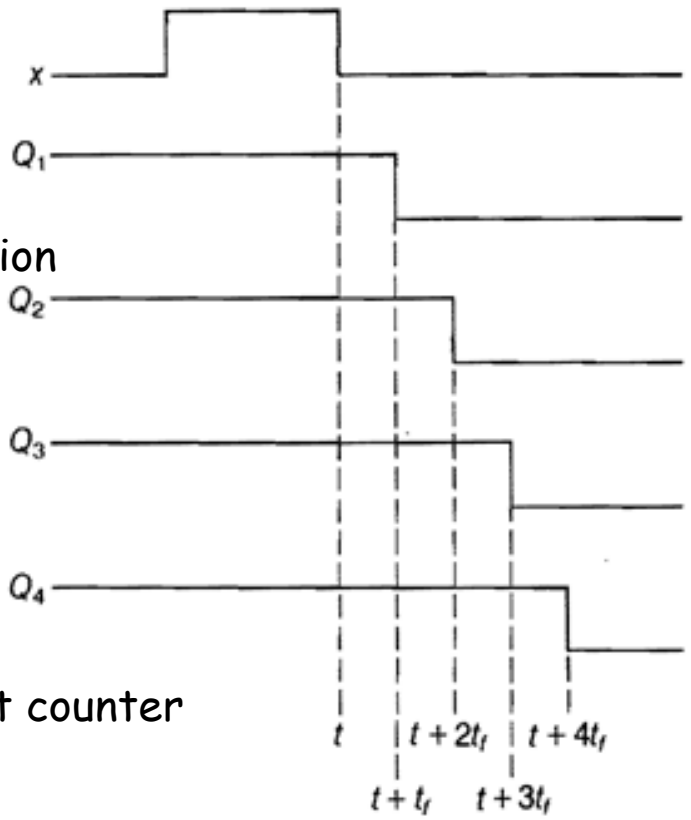
$$J_1 = Q_3$$
$$K_1 = 1$$
$$J_2 = 1$$
$$K_2 = \overline{Q_1 \oplus Q_3}$$
$$J_3 = Q_2$$
$$K_3 = Q_1 + \overline{Q}_2$$



Asynchronous Binary Counters



Problem : Timing Consequences of a Four-Bit Asynchronous Counter During 1111 → 0000 Transition.



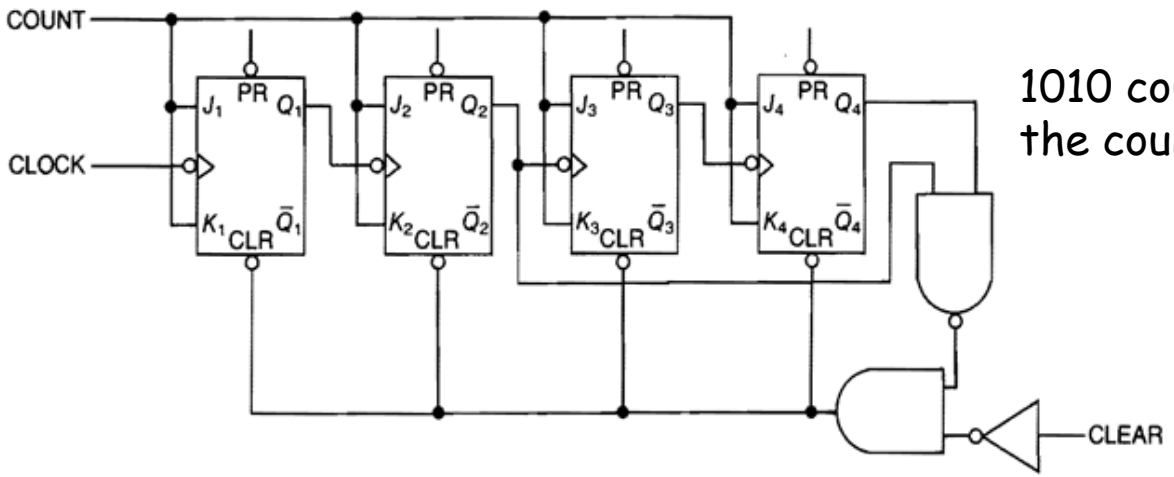
The counter does no go through the transition 1111→ **0000**.
Instead: 1111→ 1110 →1100 → 1000 → **0000**

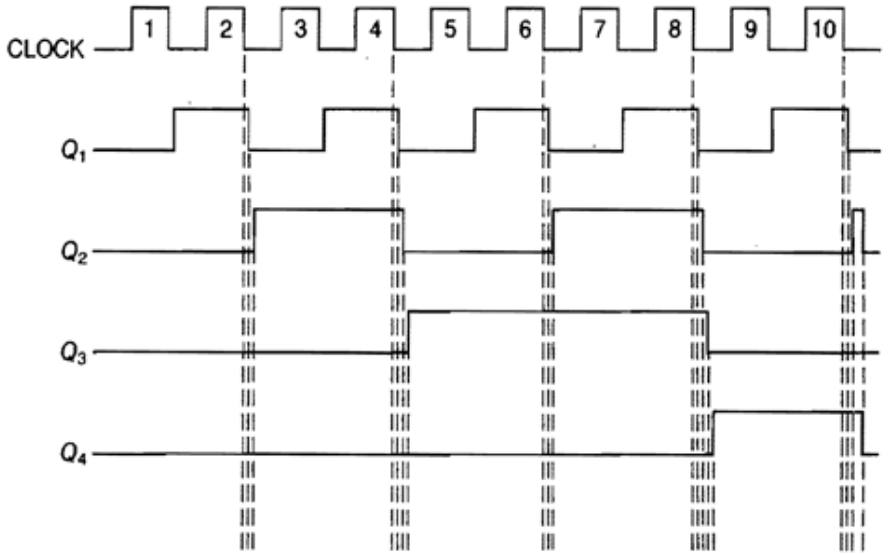
Total delay for n-bit counter is nt_f

Problems:

- Forced regular binary count
- Speed

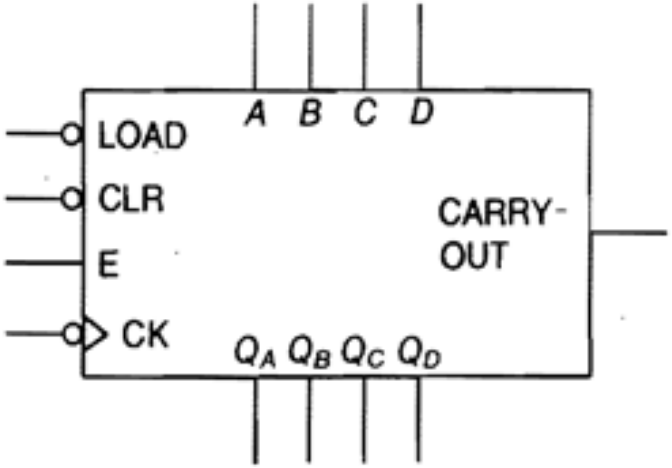
BCD Asynchronous Counter

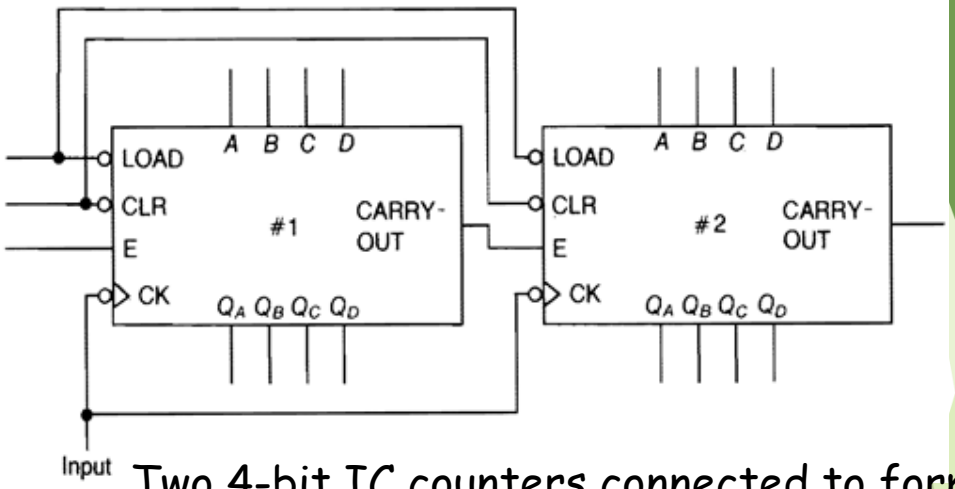




1010 count resets the counter

Integrated Circuit Counter



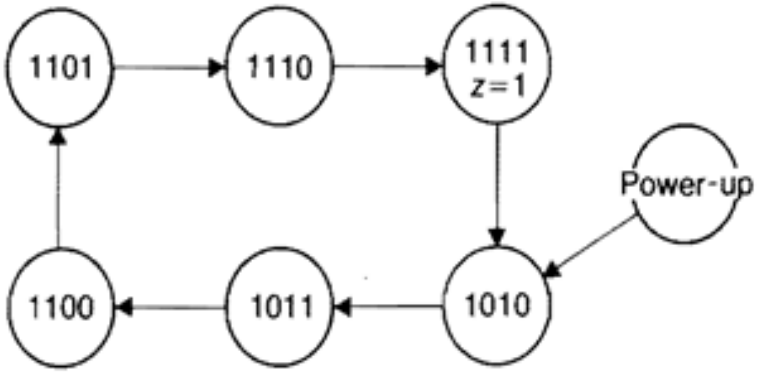


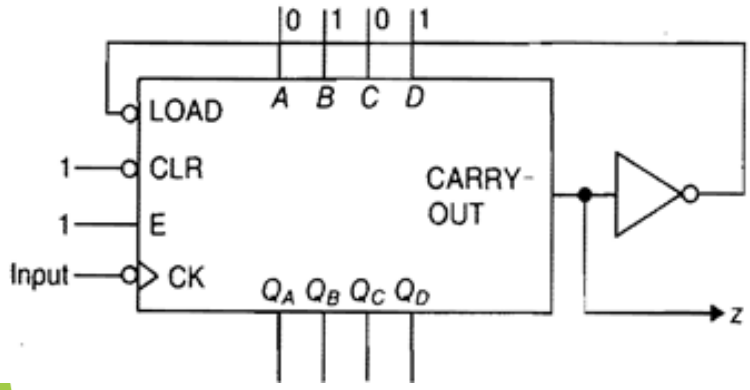
Two 4-bit IC counters connected to form an 8-bit counter

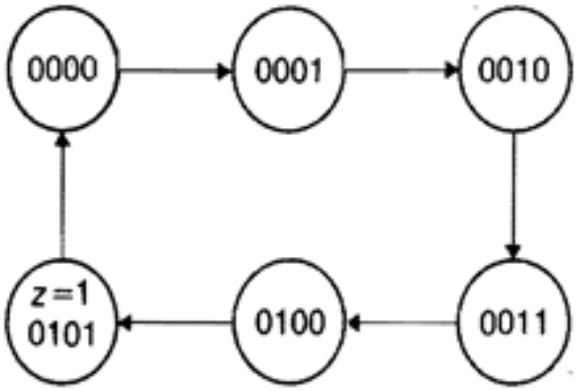
Q_{D2}	Q_{C2}	Q_{B2}	Q_{A2}	Q_{D1}	Q_{C1}	Q_{B1}	Q_{A1}
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1
0	0	0	0	0	0	1	0
.							
.							
0	0	0	0	1	1	1	1
0	0	0	1	0	0	0	0

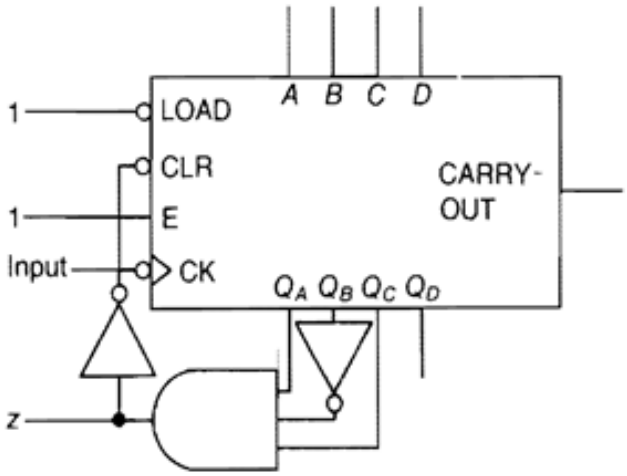
Carry out = 1
E = 1

Design a counter using the module of Figure 10.18 that outputs a 1 each time six counts have been received.







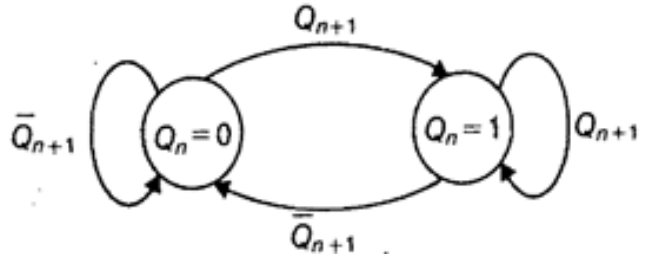


The END

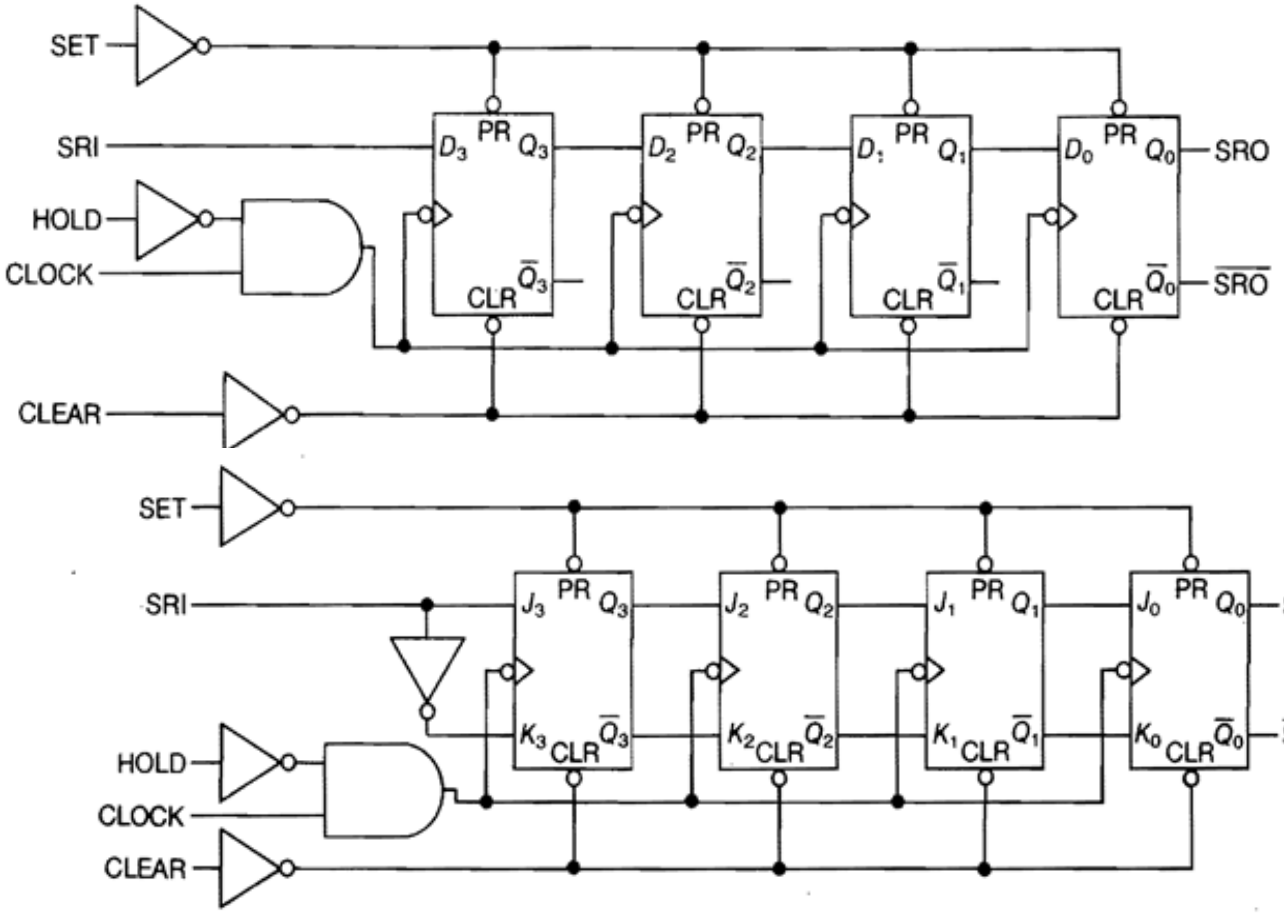
The Basic Shift Reregister

- LSBs are lost and MSBs are replaced with ones
  Shift-right register
  - LSBs replace MSBs
    Circulate-right register
  - MSBs replace LSBs
    Circulate-left register
  - Shift left one bit = multiply by 2
  - Shift right one bit = divide by 2

After Clock	Bit Pattern
0	00101100101
1	10010110010
2	11001011001
3	11100101100
4	11110010110
5	11111001011
⋮	⋮
10	11111111110
11	11111111111



Shift-right register Serial-in Serial-out



D FFs

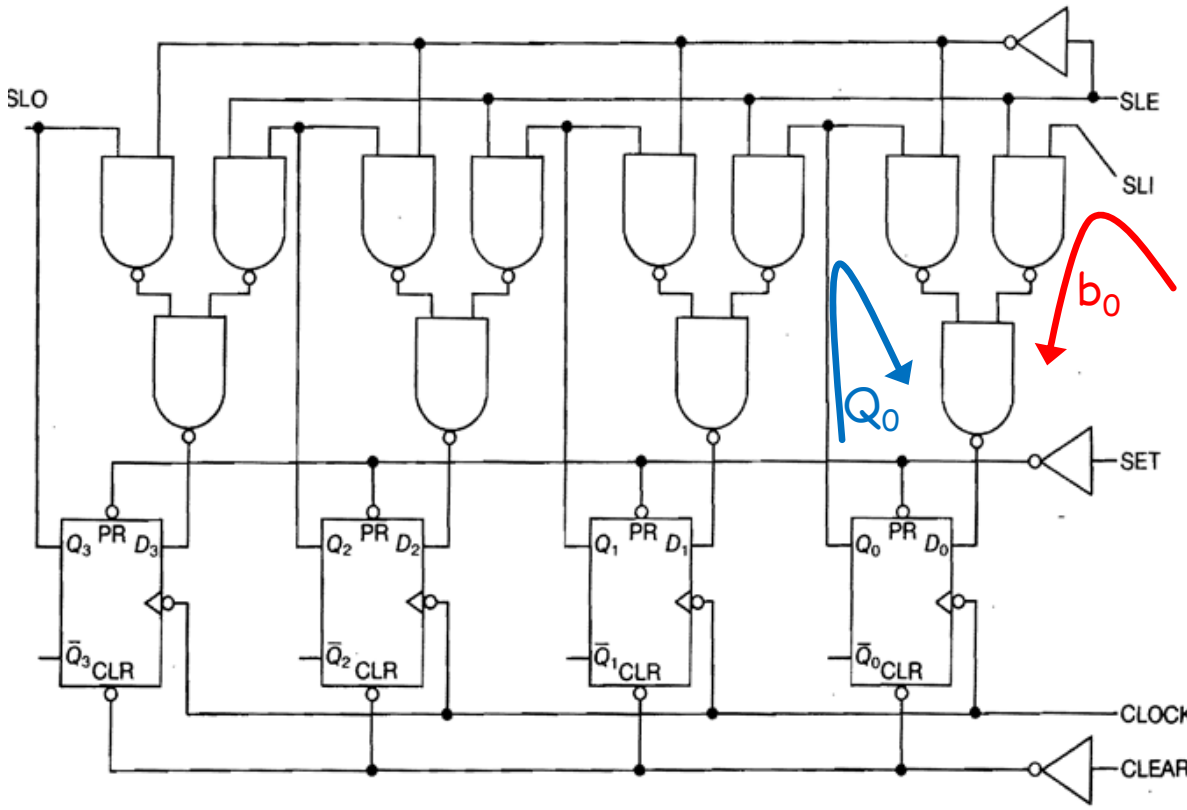
At each clock edge Ds are right-shifted to Qs

JK FFs

At each clock edge JKs are right-shifting 0 or 1 to Qs

Let it go then HOLD

Shift-left serial-in serial-out register

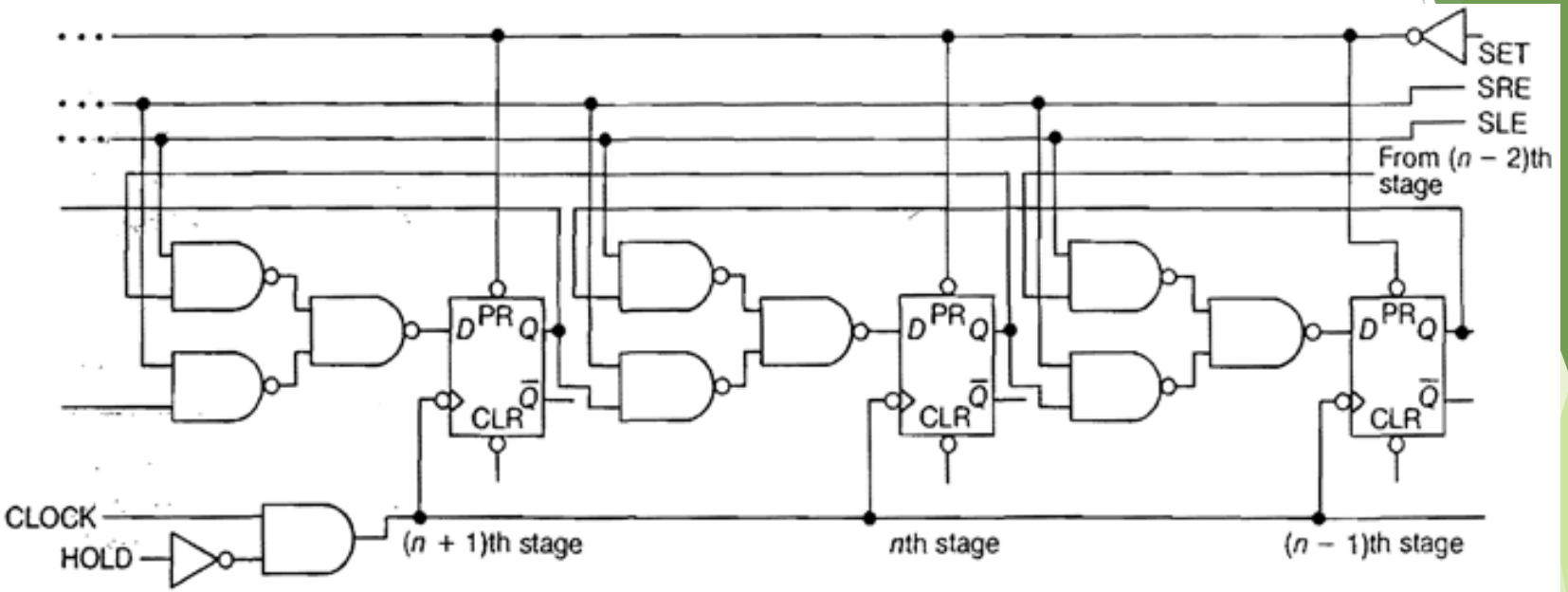


SLE = 1, shift left

SLE = 0, stores shifted bits

$b_0\, b_1 b_2 \ldots\ldots$

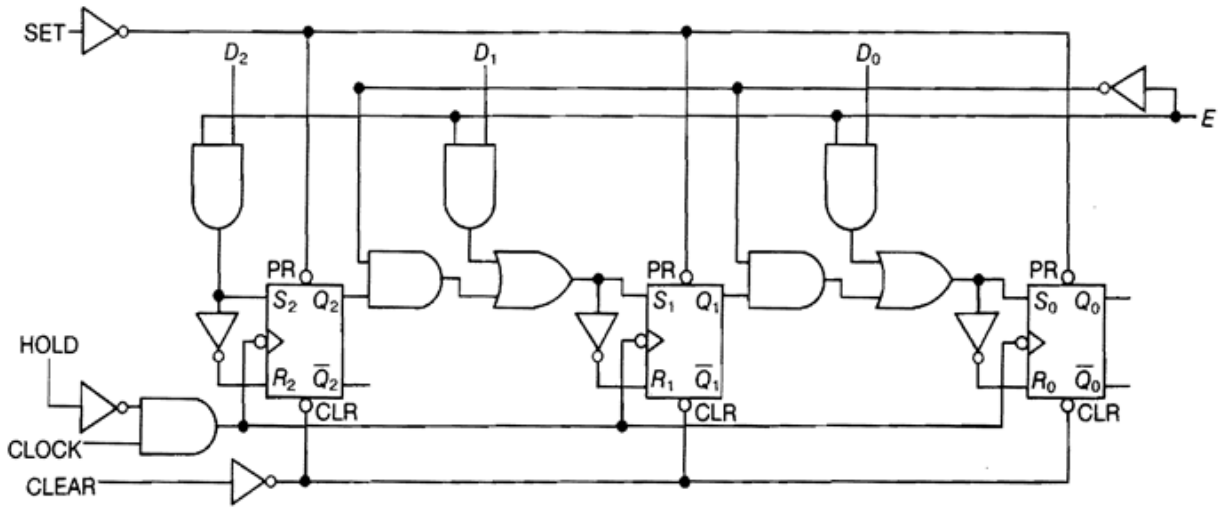
Sigle register used for both serial-right and serial-left shifts



SRE	SLE	action
0	0	FFs reset
0	1	Shift-left
1	0	Shift-right
1	1	never

Hold	Action
0	Shifts normally
1	Restores old values

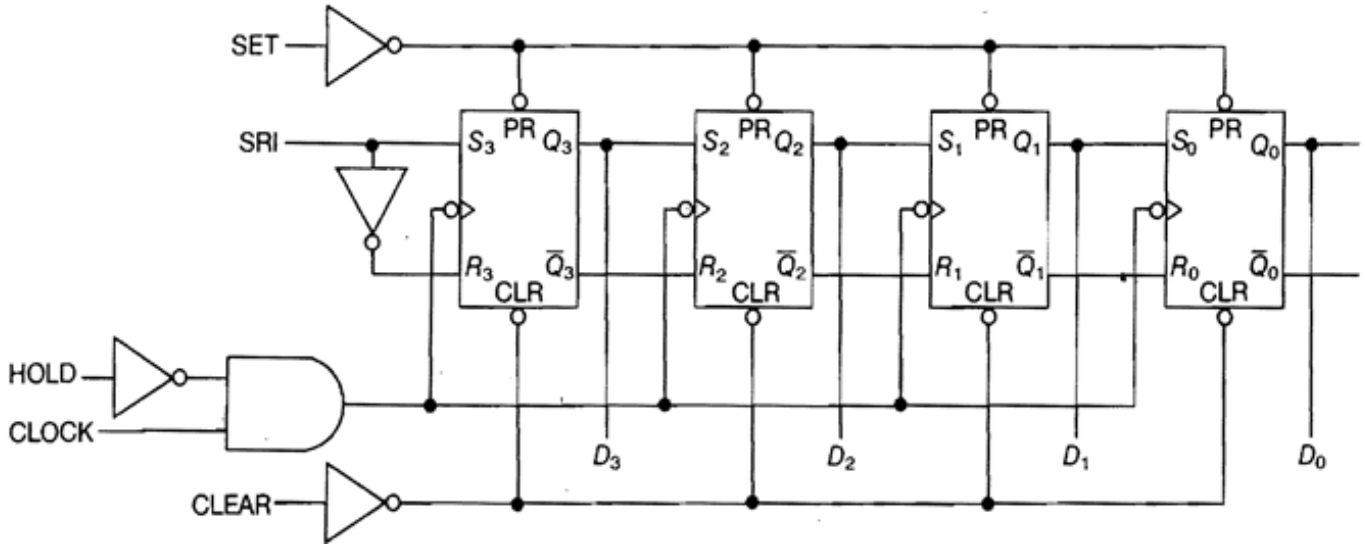
3-bit parallel in serial out shift register



E	Action
1	Data loaded
0	Right-shift

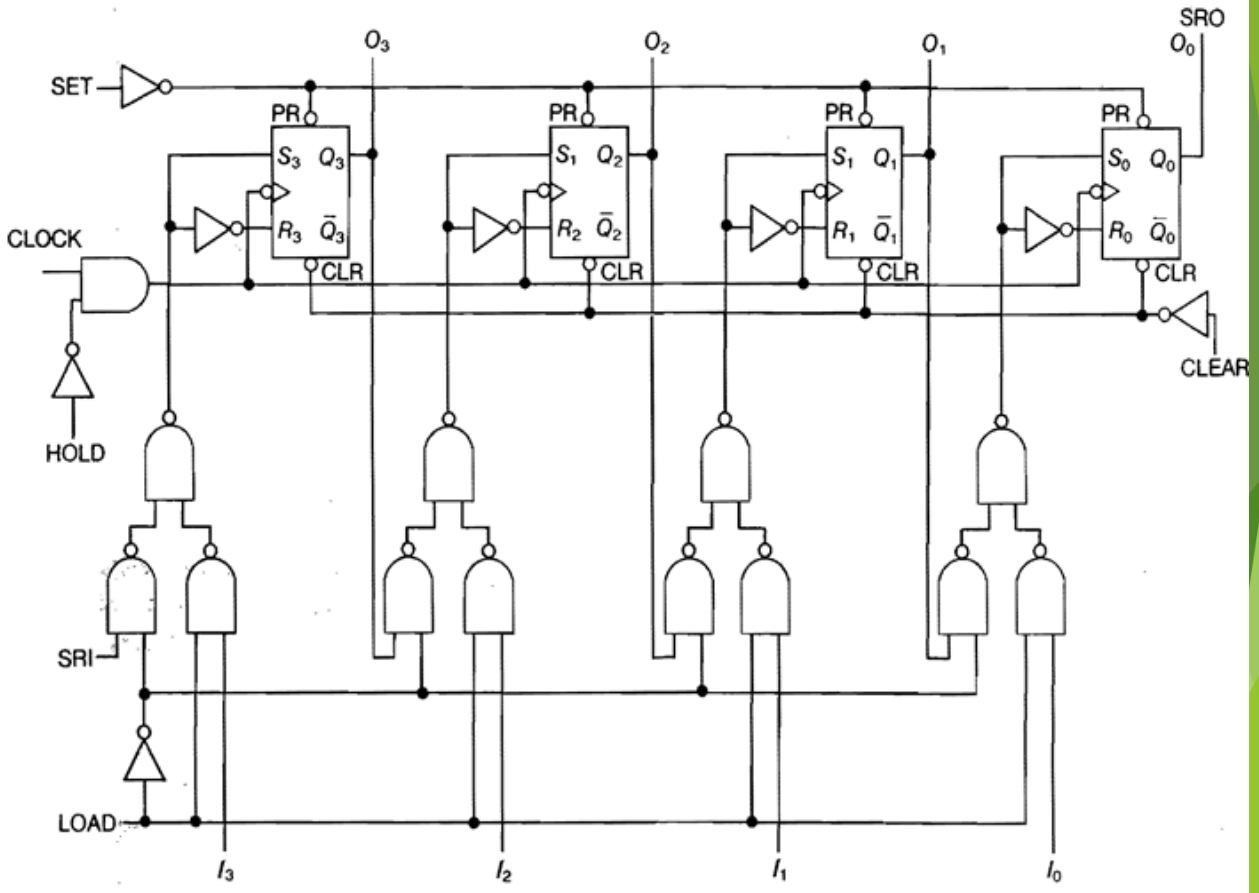
Hold	Action
0	Load or shift
1	Restores

4-bit Serial in Parallel out Shift Register

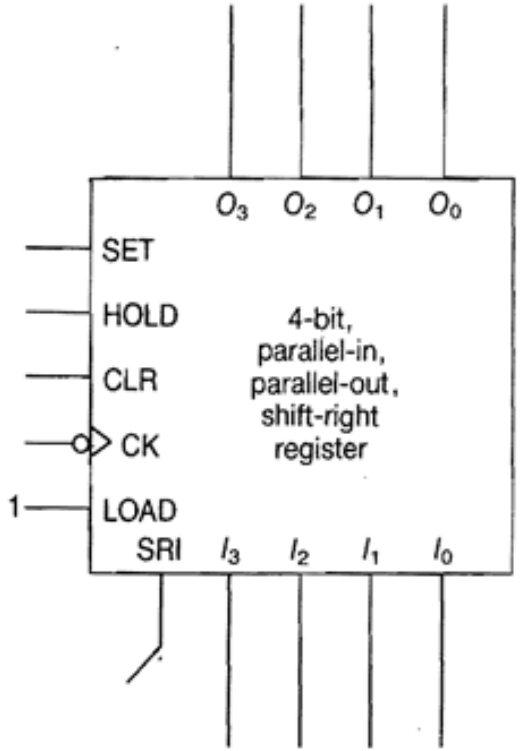


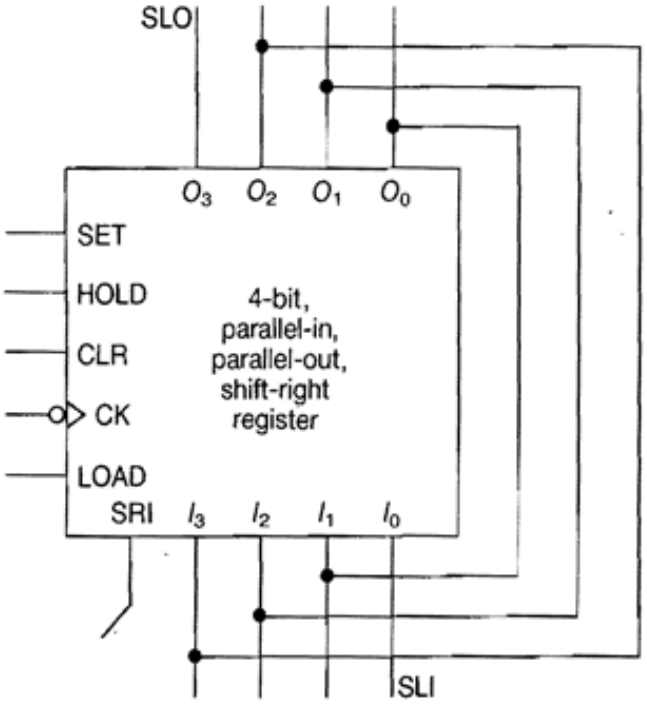
4-bit Parallel in Parallel out Shift Register

Load	Action
1	Parallel in/out
0	Serial in/out



Universal Shift Register





Load	Action
0	Shift-right serial in/out
1	Parallel

Wired shift-left register

Chapter 5: Design of Asynchronous Sequential Circuits

Pulse-Mode Circuits:

No clock for state transitions.

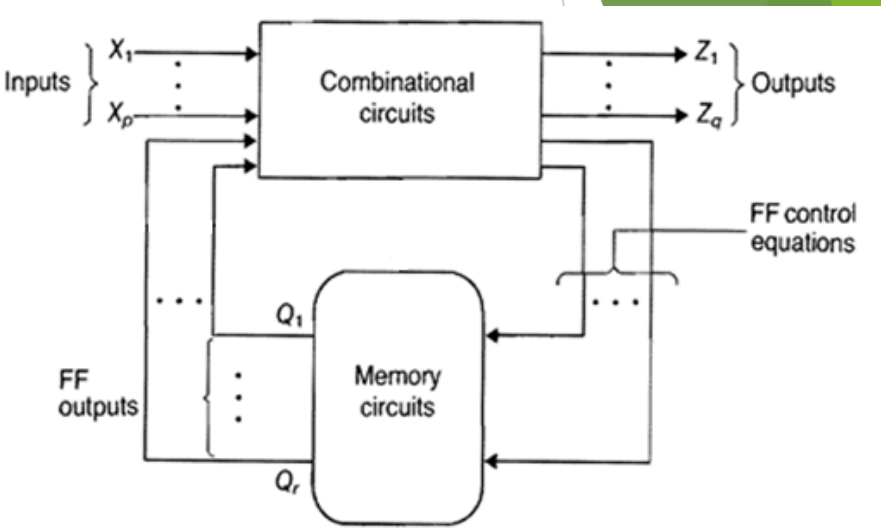
Just input pulses, Key boards and Vending machines

No overlapping pulses

Conditions for perfect circuit operation

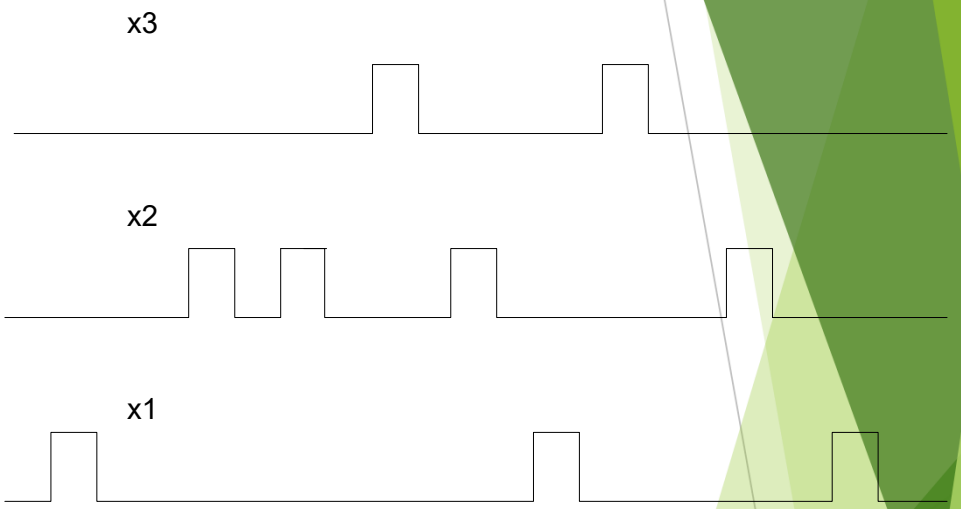
1. Simultaneous input pulses on two or more lines are forbidden. Interval between pulses is large enough for system to return to a stable state.

2. Pulse widths must be sufficient to allow the components to respond to them.



Looks like synchronous without a clock

As long as the previously stated two conditions are met the design of asynchronous circuits will be similar to the synchronous ones



An example of three nonoverlapping input pulses

Design Algorithm

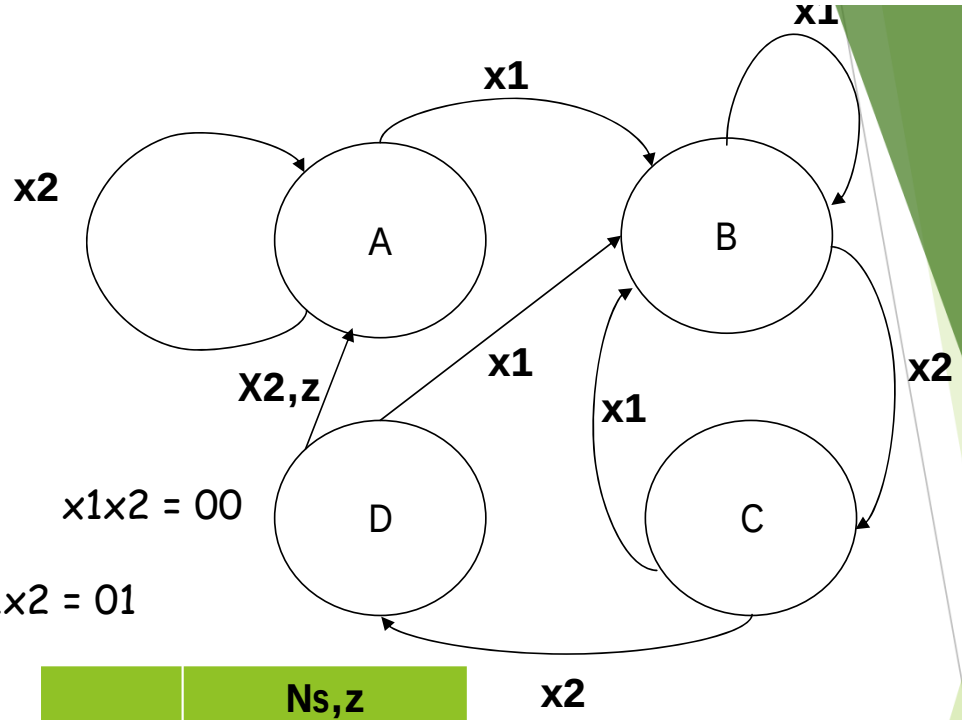
1. Get the state diagram. For n-inputs there be n transition paths leaving each of the sates.
2. Remove redundancies.
3. Assign states and generate state transition table.
4. Get the excitation maps and write down the excitation equations.
5. Draw the circuit diagram.

Example: Design a circuit that receives two inputs x1 and x2 and gives an output coincident with the third consecutive x2 pulse following at least one x1 pulse.

Forbidden pulses: x1x2 = 11

No pulses have occured x1x2 = 00

Useful pulses x1x2 = 10 and x1x2 = 01



PS	Ns,z	
	x2	x1
A	A	B
B	C	B
C	D	B
D	A,z	B

PS Q1Q2	Ns,z	
	X2 (01)	X1 (10)
00	00	01
01	11	01
11	10	01
10	00,z	01

PS Q1Q2	Ns,z	
	X2 (01)	X1 (10)
00	00	01
01	11	01
11	10	01
10	00,z	01

Excitation maps:
T FFs are assumed

Q1Q2	x1	x2
00	0	0
01	0	1
11	1	0
10	1	1

T1= $x1Q1 + x2(Q1 \oplus Q2)$

Q1Q2	x1	x2
00	1	0
01	0	0
11	0	1
10	1	0

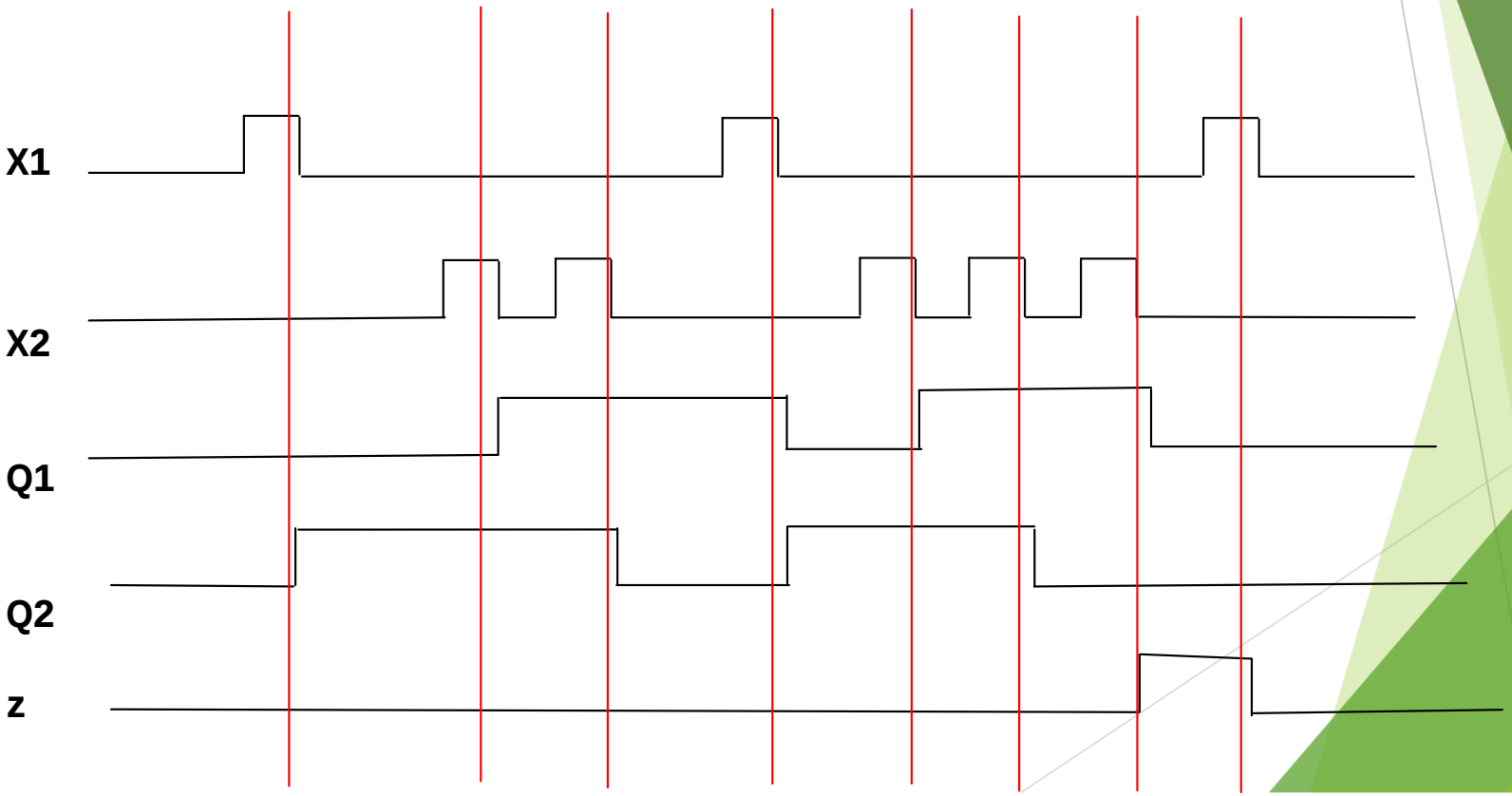
T2= $x1\overline{Q2} + x2Q1Q2$

Q1Q2	x1	x2
00	0	0
01	0	0
11	0	0
10	0	1

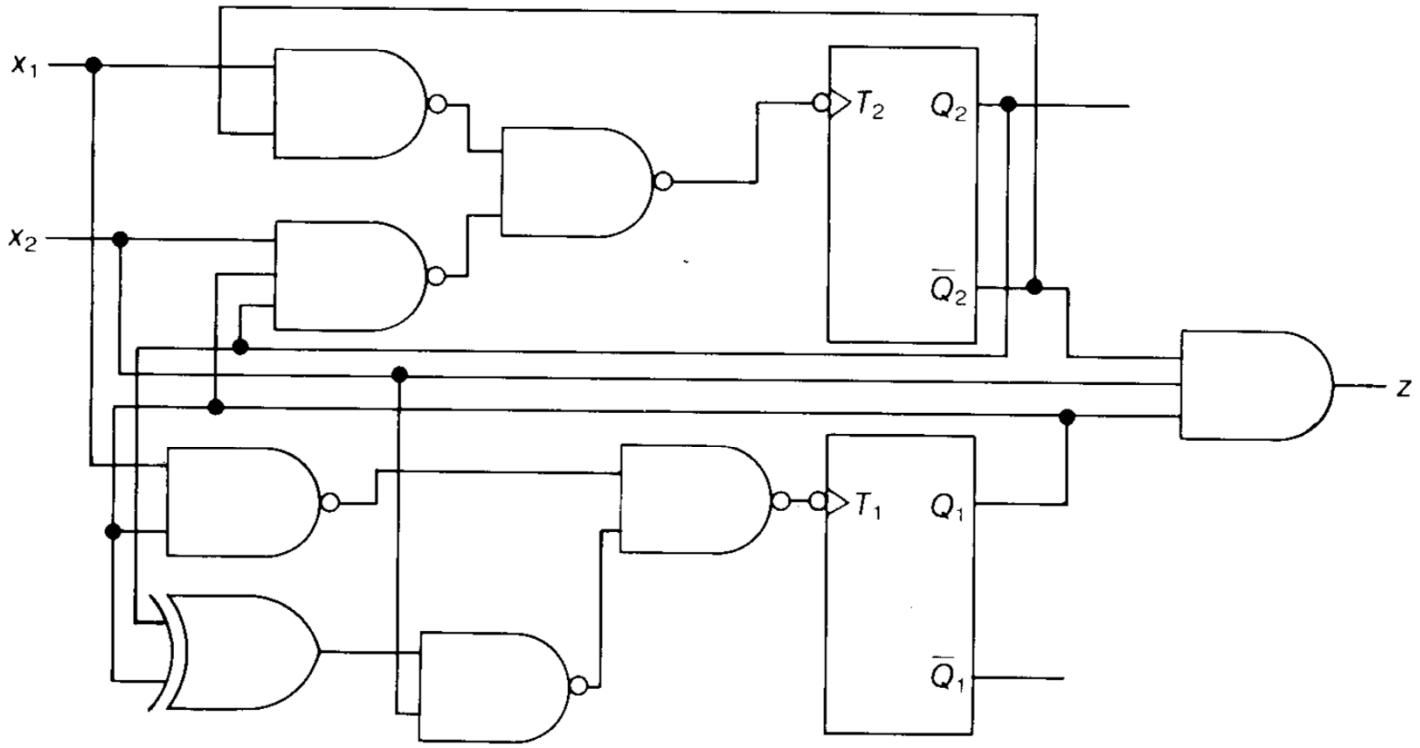
z= $x2Q1\overline{Q2}$

Timing diagram of Negative-edge triggered T FFs

T1= $x1Q1 + x2(Q1 \oplus Q2)$ T2= $x1\overline{Q2} + x2Q1Q2$ z= $x2Q1\overline{Q2}$

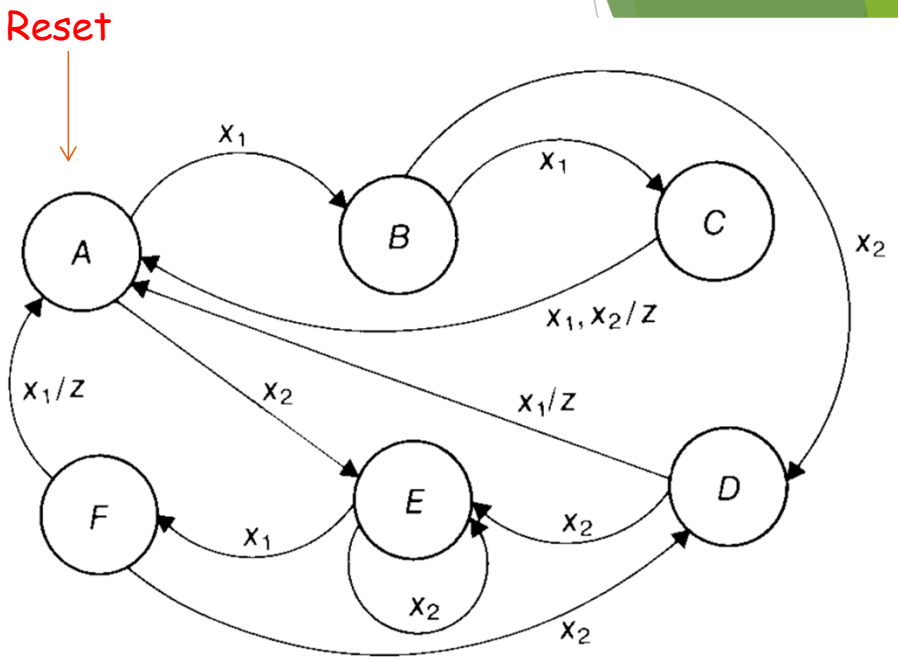


The HW implementation of the excitation equations



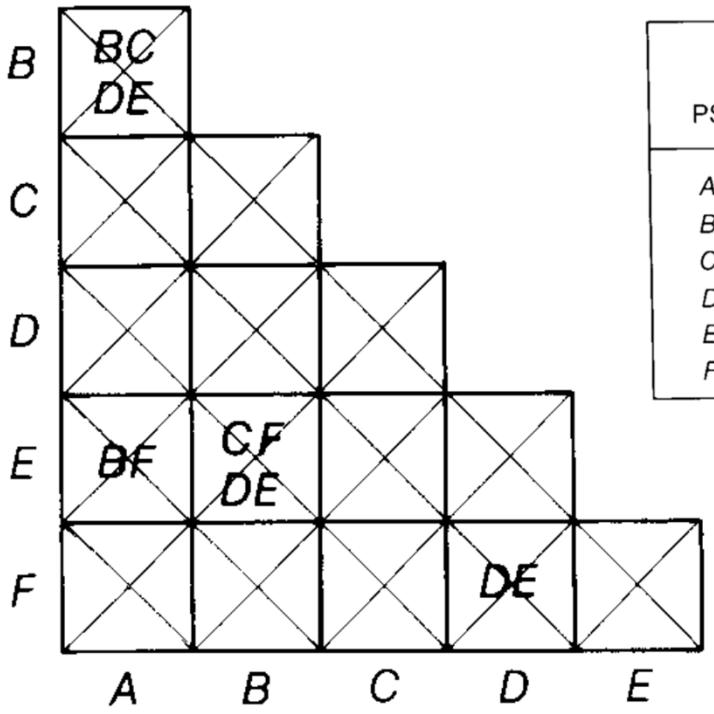
Example 2:

Design a pulse-mode circuit that satisfies the following requirements. The circuit has two inputs, x_1 and x_2, and one output, z. The output pulse is to be produced simultaneously with the last of a sequence of three input pulses if and only if the sequence contained at least two x_1 pulses. After each output a new sequence is looked for.



PS	NS, z	
	x_1	x_2
A	B	E
B	C	D
C	A,z	A,z
D	A,z	E
E	F	E
F	A,z	D

Implication table



No redundance states

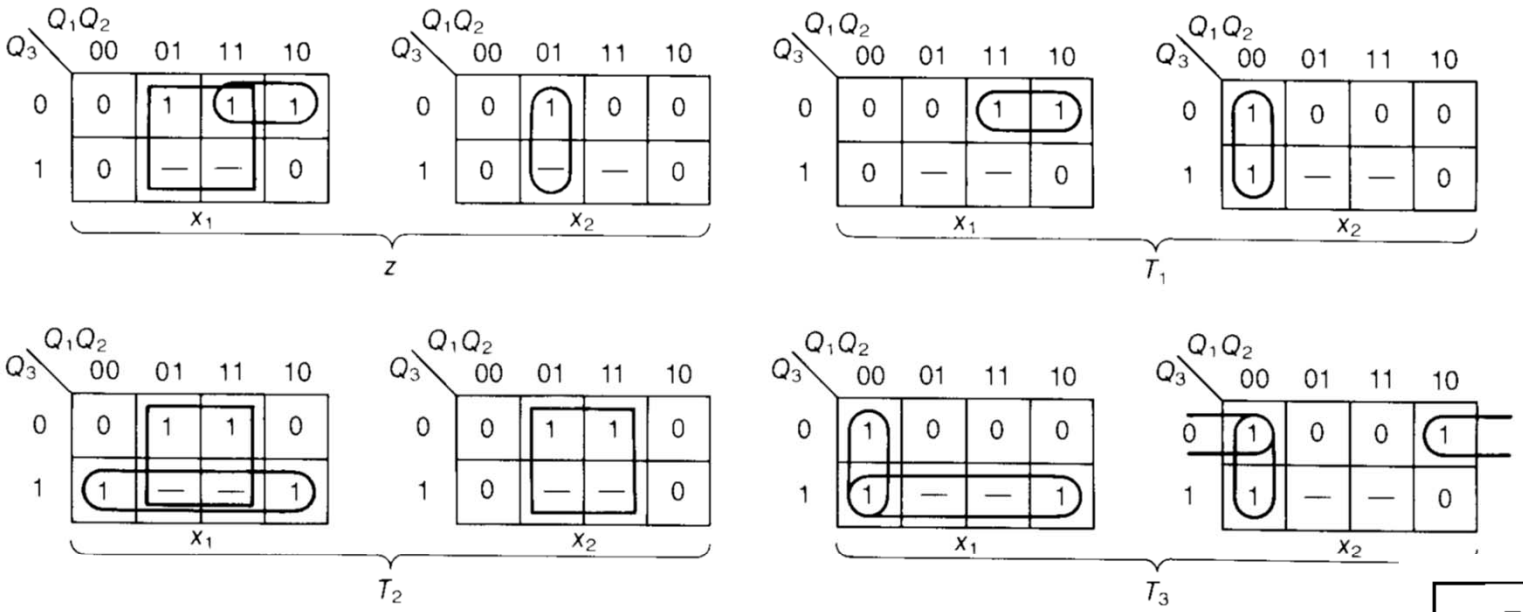
PS	NS, z	
	x_1	x_2
A	B	E
B	C	D
C	A,z	A,z
D	A,z	E
E	F	E
F	A,z	D

Arbitrary state assignment

A=000
B=001
C=010
D=100
E=101
F=110

State Transition Table

PS	NS, z	
$Q_1 Q_2 Q_3$	x_1	x_2
000	001	101
001	010	100
010	000,z	000,z
100	000,z	101
101	110	101
110	000,z	100



$$z = x_1(Q_2 + Q_1\,\overline{Q}_3) + x_2(\overline{Q}_1 Q_2)$$

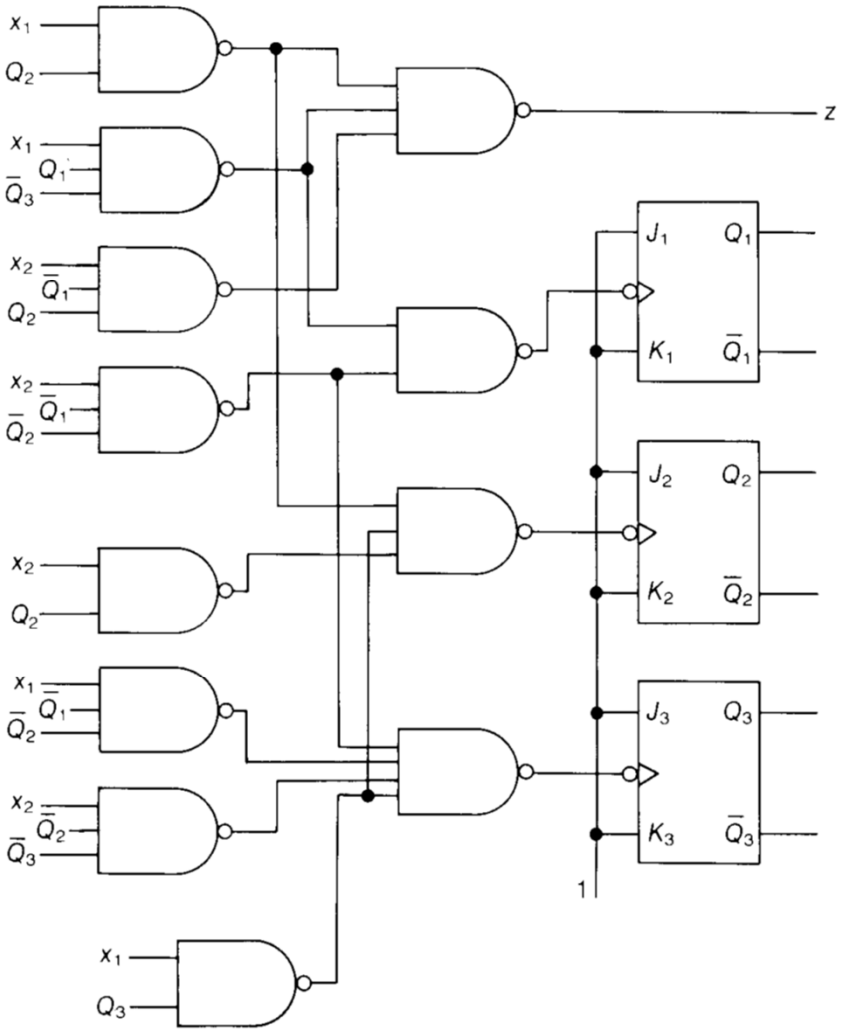
$$T_1 = x_1(Q_1\overline{Q}_3) + x_2(\overline{Q}_1\overline{Q}_2)$$

$$T_2 = x_1(Q_2 + Q_3) + x_2(Q_2)$$

$$T_3 = x_1(Q_3 + \overline{Q}_1\overline{Q}_2) + x_2(\overline{Q}_1\overline{Q}_2 + \overline{Q}_2\overline{Q}_3)$$

PS	NS, z	
$Q_1Q_2Q_3$	x_1	x_2
000	001	101
001	010	100
010	000,z	000,z
100	000,z	101
101	110	101
110	000,z	100

HW Implementation

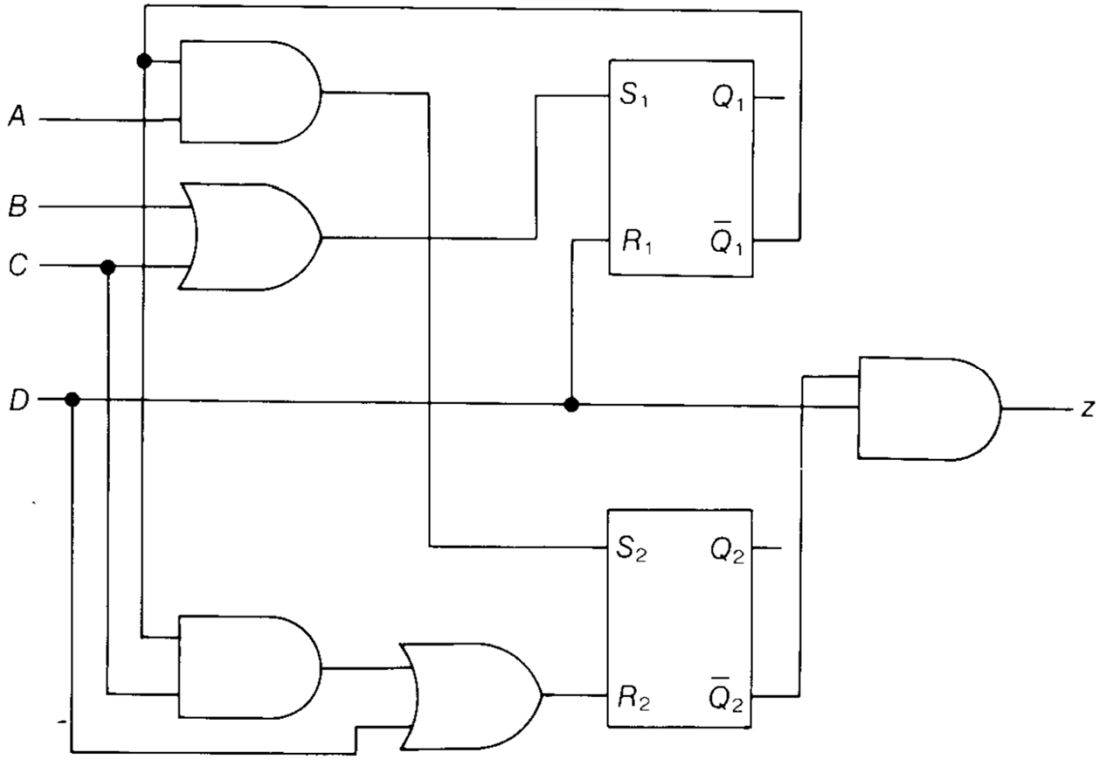


Example 3

Analyze the pulse-mode circuit of Figure

The equations of the states and the output are easily written as:

$$S_1 = B + C$$
$$R_1 = D$$
$$S_2 = A\bar{Q}_1$$
$$R = C\bar{Q}_1 + D$$
$$Z = D\bar{Q}_2$$



Since it is a pulse-mode circuit the only input conditions that need to be considered are $ABCD$ = 1000, 0100, 0010, and 0001.

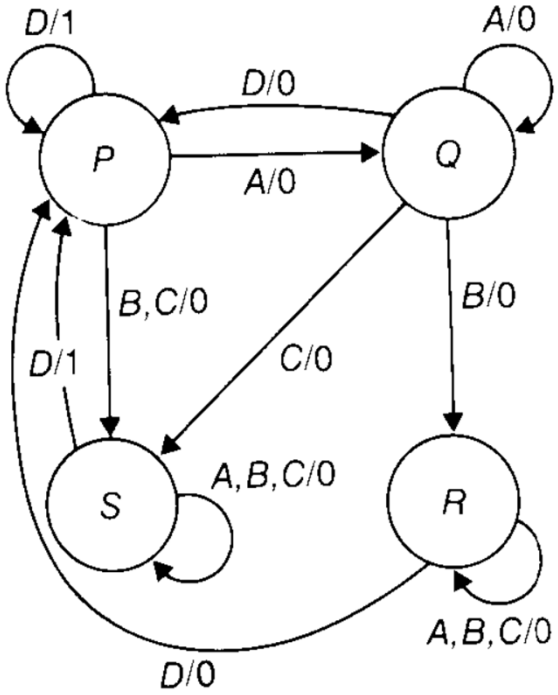
$$S_1 = B + C$$

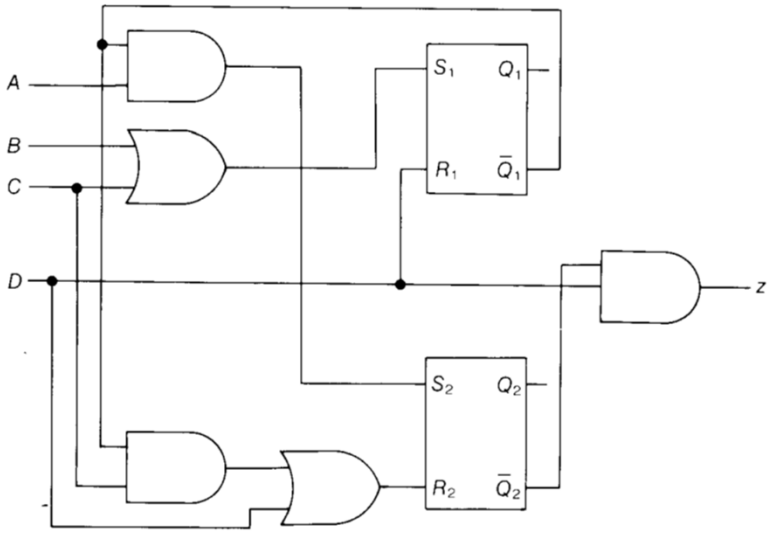
$$R_1 = D$$

$$S_2 = A\bar{Q}_1$$

$$R = C\bar{Q}_1 + D$$

$$Z = D\bar{Q}_2$$





PS	NS, z			
$Q_1 Q_2$	A	B	C	D
P 00	01,0	10,0	10,0	00,1
Q 01	01,0	11,0	10,0	00,0
R 11	11,0	11,0	11,0	00,0
S 10	10,0	10,0	10,0	00,1