ITI1100B

CHAPTER-6

REGISTER

A SEQUENTIAL CIRCUIT WHICH CONSISTS OF A GROUP OF FF'S AND (POSSIBLY) EXTERNAL GATES CONTROLLING THE FF INPUTS WHERE

- **FF'S** HOLD BINARY INFO
  AND
- **EXTERNAL GATES** CONTROL WHEN & HOW
  NEW BINARY INFO IS TRANSFERRED TO FF'S

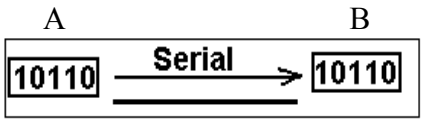
* **(THE CONTENT OF A REGISTER = WORD** = BIT STREAM STORED IN FF'S**)**
* **(THE LENGTH OF A REGISTER = WORD LENGTH** = # OF FF'S IN A REGISTER**)**

* ALL REGISTERS (EXCEPT GATED LATCHES) ARE BUILT BY USING FF'S
  THAT ARE SENSITIVE TO **PULSE TRANSITION** RATHER THAN TO **PULSE DURATION**

* THE TRANSFER OF NEW BINARY INFO INTO A REGISTER IS CALLED **LOADING**

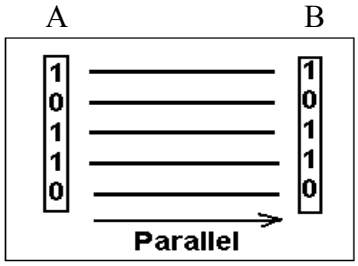
* **SERIAL LOAD** **(TRANSFER IN SERIAL MODE)**
TRANSFER OF NEW BINARY INFO INTO A REGISTER BIT BY BIT
(# OF CLOCK PULSES NEEDED TO LOAD THE REGISTER = # OF FF'S IN THE REGISTER)

e.g., Let A and B be two 5-bit registers



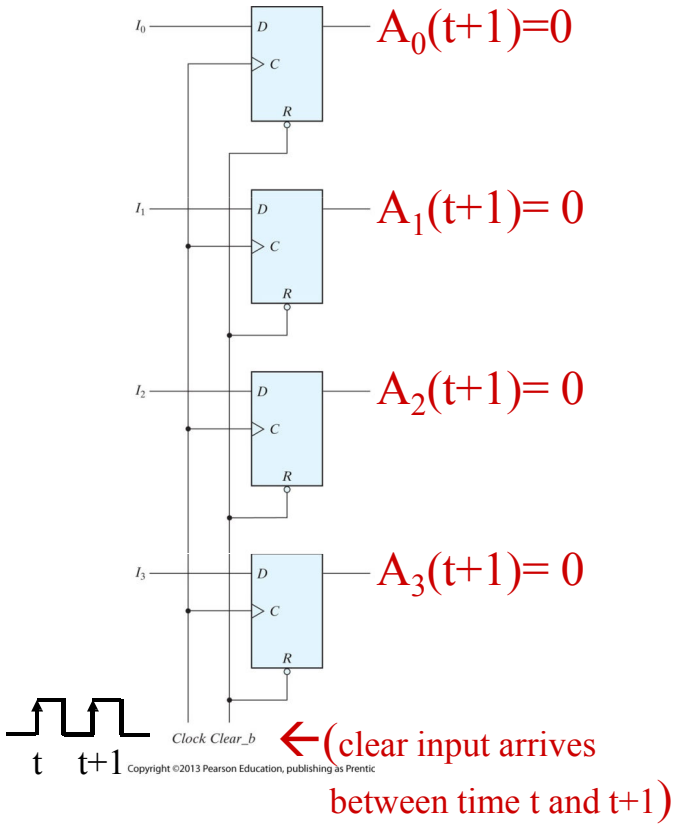
* **PARALLEL LOAD** **(TRANSFER IN PARALLEL MODE)**
TRANSFER OF NEW BINARY INFO INTO A REGISTER AS A WHOLE WORD
(# OF CLOCK PULSES NEEDED TO LOAD THE REGISTER = 1)

e.g., Let A and B be two 5-bit registers



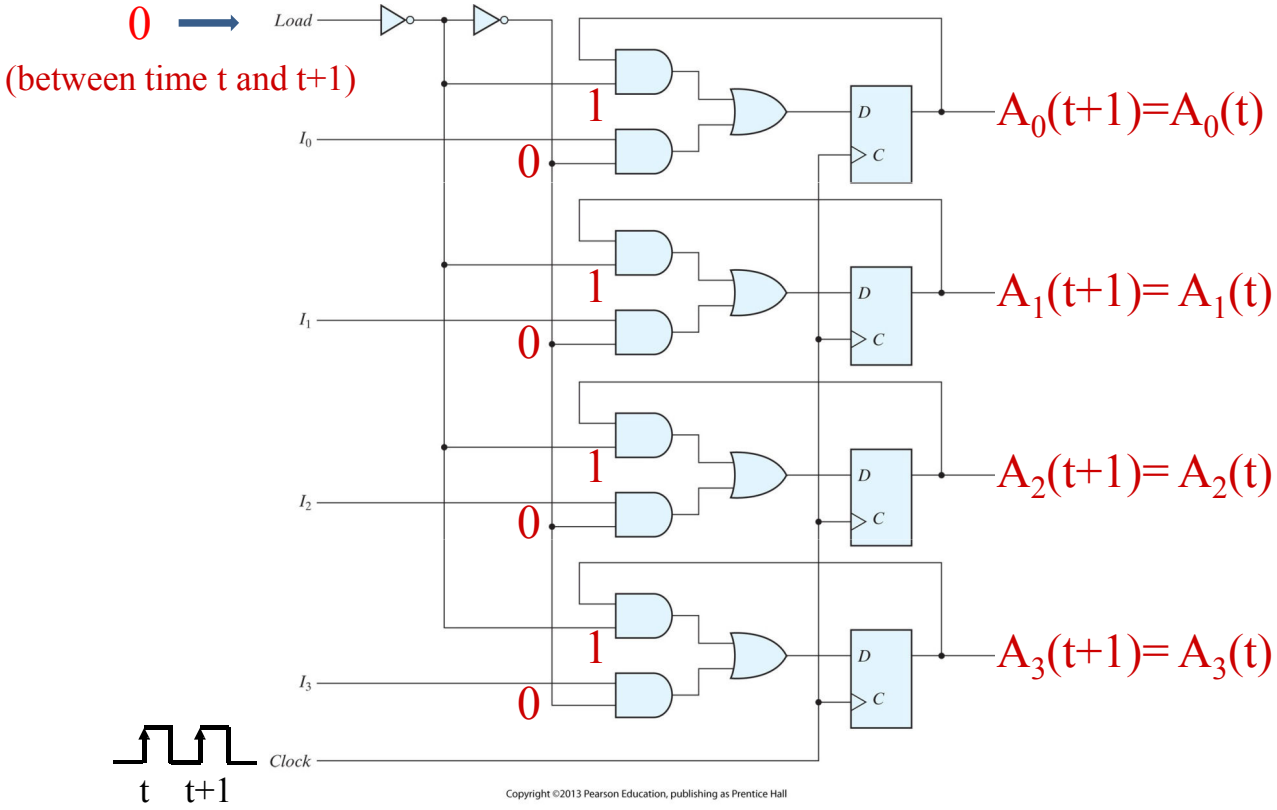
e.g., 4-BIT REGISTER WITH PARALLEL LOAD

RESETTING



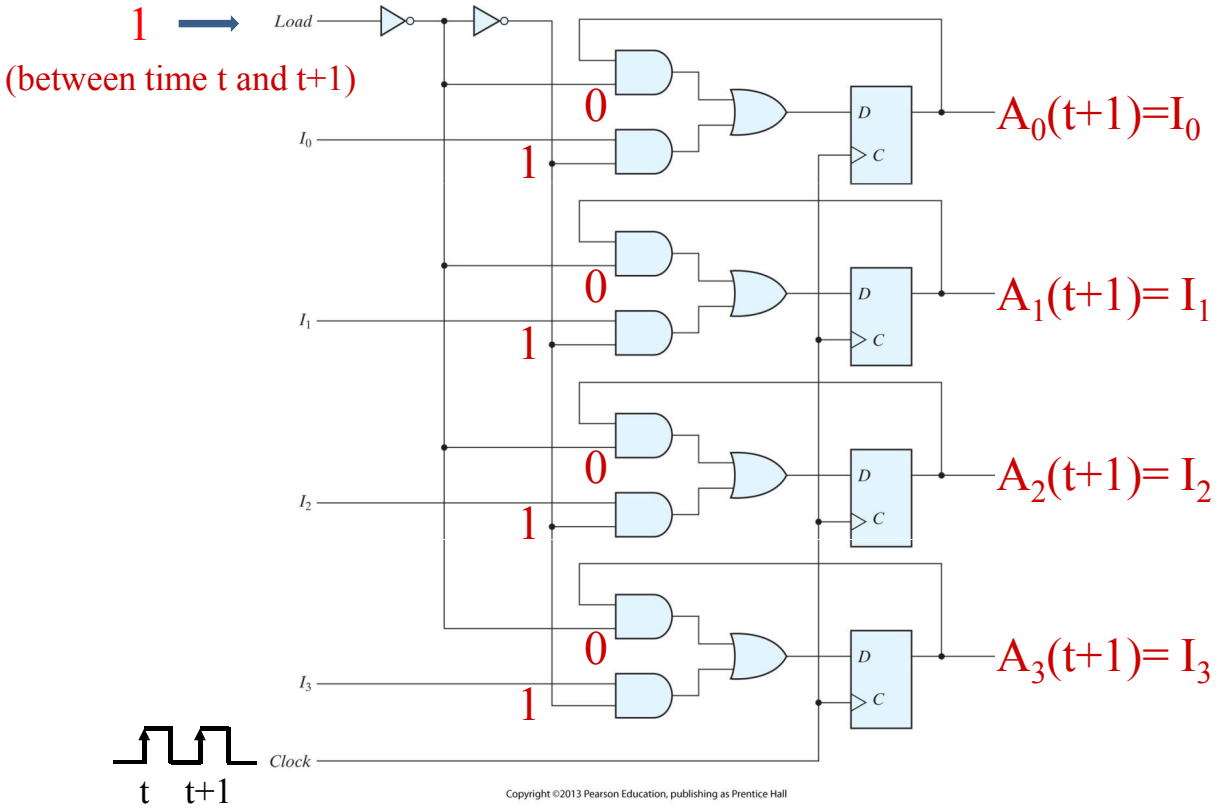
e.g., 4-BIT REGISTER WITH PARALLEL LOAD

REFRESHING



e.g., 4-BIT REGISTER WITH PARALLEL LOAD

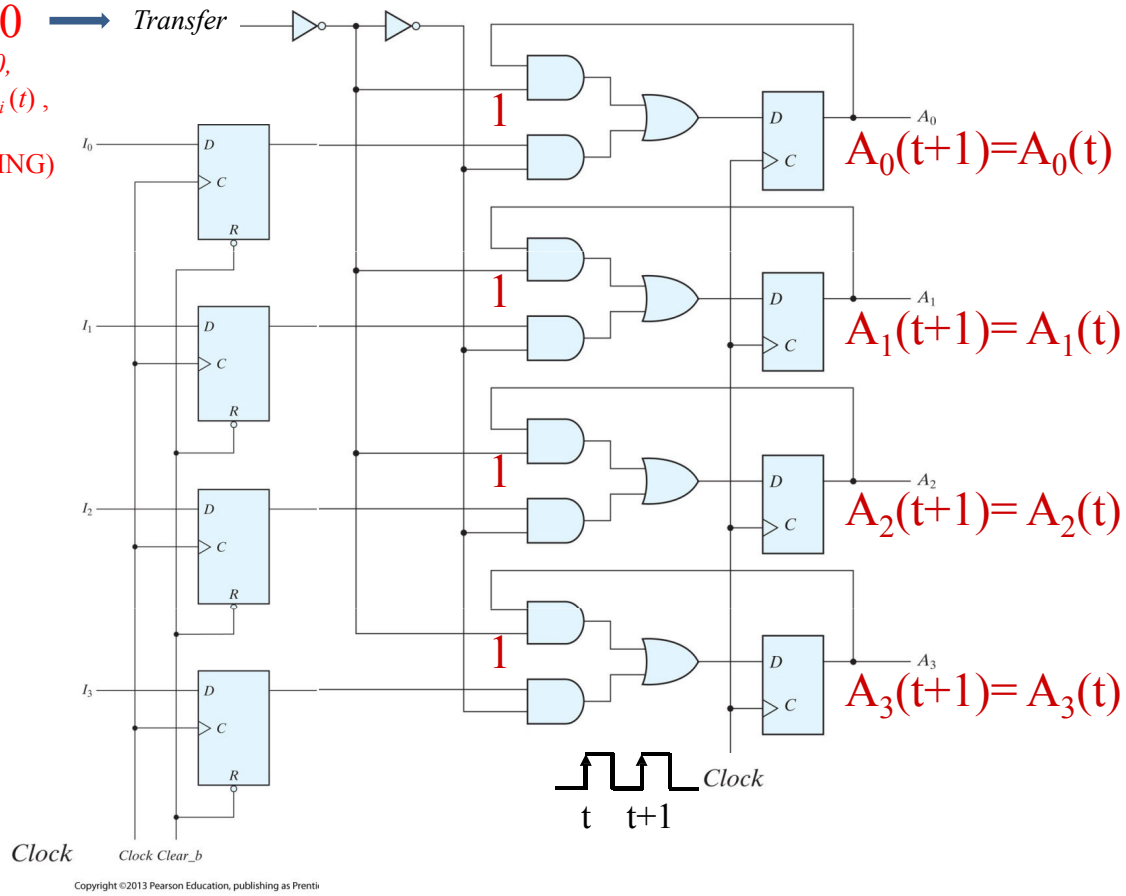
LOADING



TRANSFER IN PARALLEL MODE

0 → *Transfer*

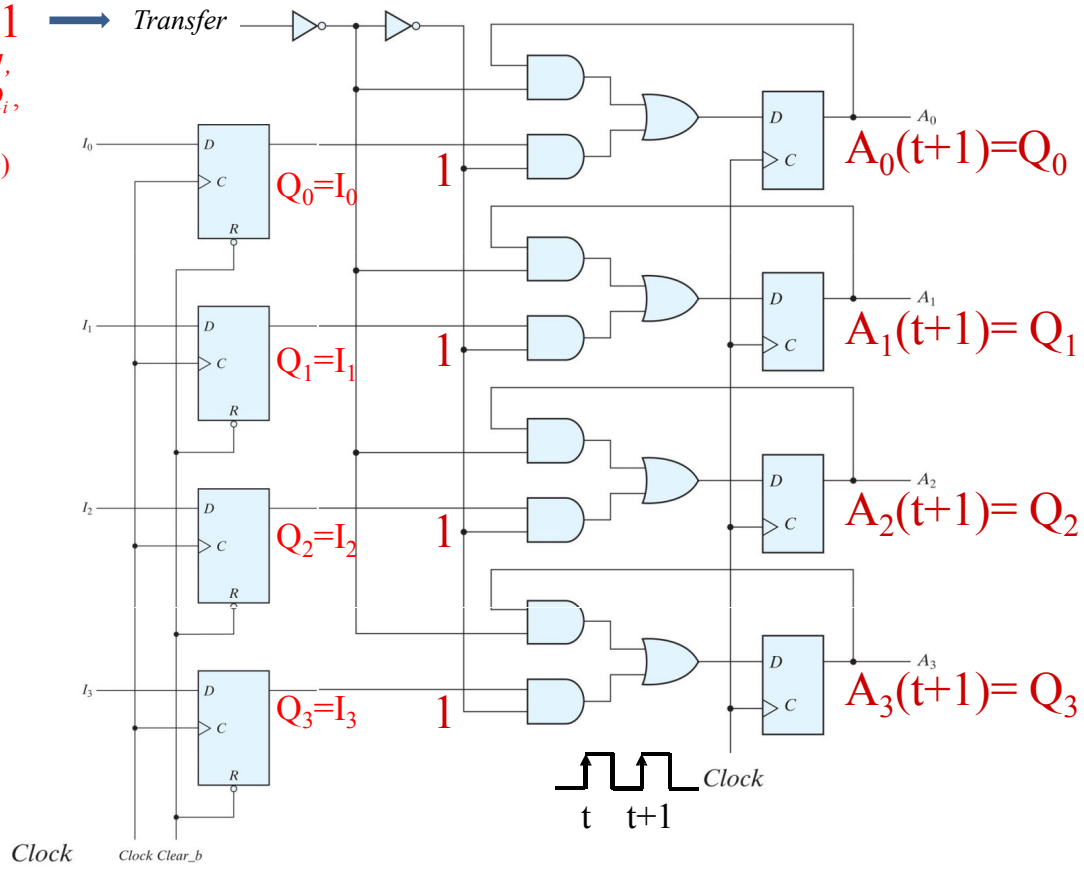
Transfer = 0,
$A_i(t+1) = A_i(t)$,
for $0 \le i \le 3$
(REFRESHING)



TRANSFER IN PARALLEL MODE

1 → *Transfer*

Transfer = 1,
$A_i(t+1) = Q_i$,
for $0 \le i \le 3$
(LOADING)



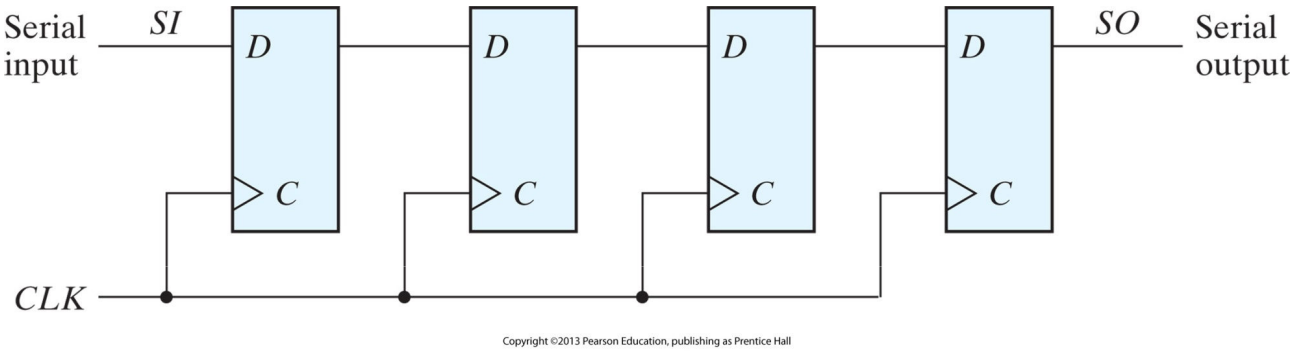
SHIFT REGISTER

A REGISTER CAPABLE OF SHIFTING ITS BINARY INFO TO ITS LEFT OR RIGHT

A REGISTER WHERE RS-FF OR D-FF'S ARE CONNECTED IN CASCADE

- UNIDIRECTIONAL
- BIDIRECTIONAL

UNIDIRECTIONAL 4-BIT SHIFT REGISTER (Shifts RIGHT)

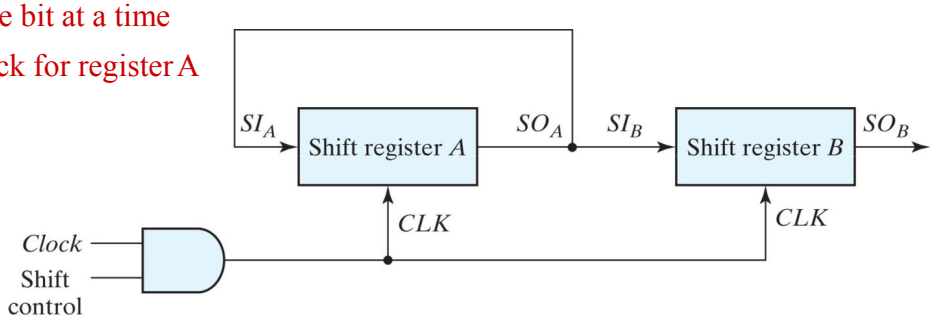


TRANSFER IN SERIAL MODE

(**BIT TIME**) : TIME INTERVAL BETWEEN CLOCK PULSES

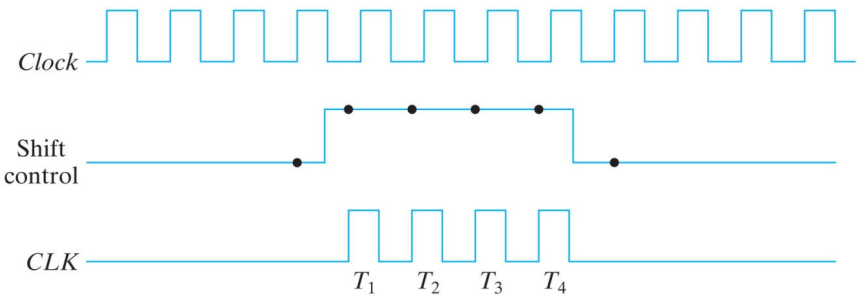
(**WORD TIME**) : TIME INTERVAL NEEDED TO SHIFT THE ENTIRE CONTENT OF A REGISTER

- Data is transferred one bit at a time
- Note the data loop back for register A



(a) Block diagram

Time	A	B
T0	1011	0011
T1	1101	1001
T2	1110	1100
T3	0111	0110
T4	1011	1011



(b) Timing diagram

BIDIRECTIONAL SHIFT REGISTER WITH PARALLEL LOAD

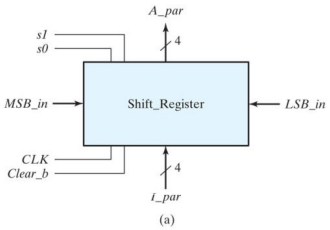


Table 6.3
Function Table for the Register of Fig. 6.7

Mode Control		
s_1	s_0	Register Operation
0	0	No change
0	1	Shift right
1	0	Shift left
1	1	Parallel load

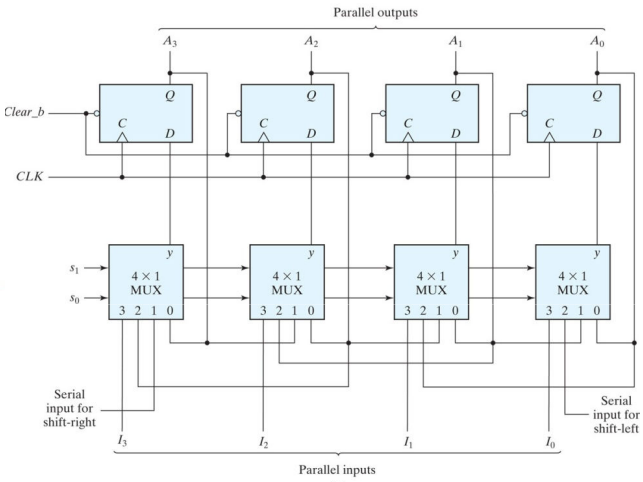
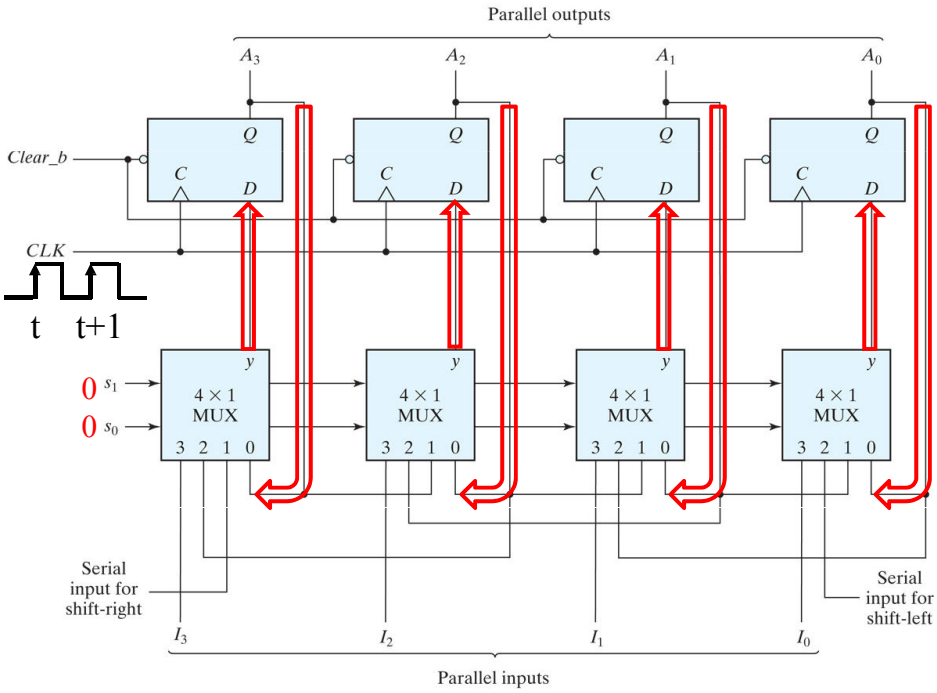


FIGURE 6.7 Four-bit universal shift register

$S_1\ S_0 = 00$ ---> NO CHANGE

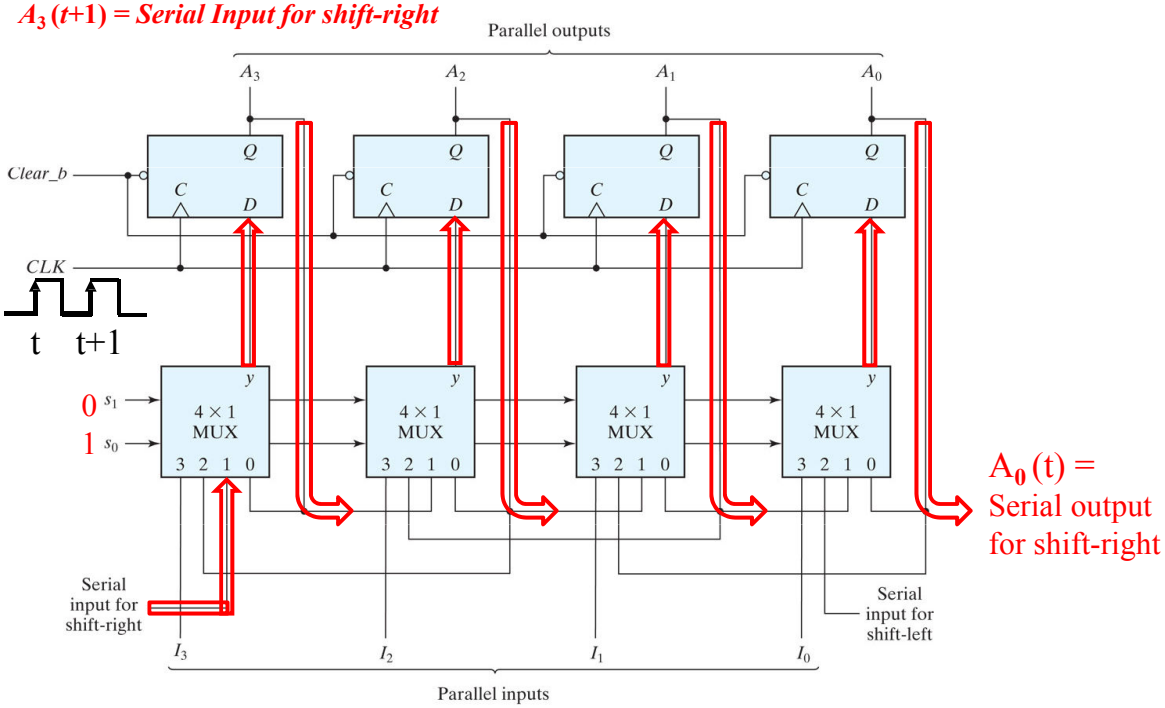
$A_i(t+1) = A_i(t)$ for $0 \le i \le 3$



$S_1\ S_0 = 01$ ---> SHIFT RIGHT

$A_i(t+1) = A_{i+1}(t)$ for $0 \le i \le 2$

$A_3(t+1) =$ *Serial Input for shift-right*



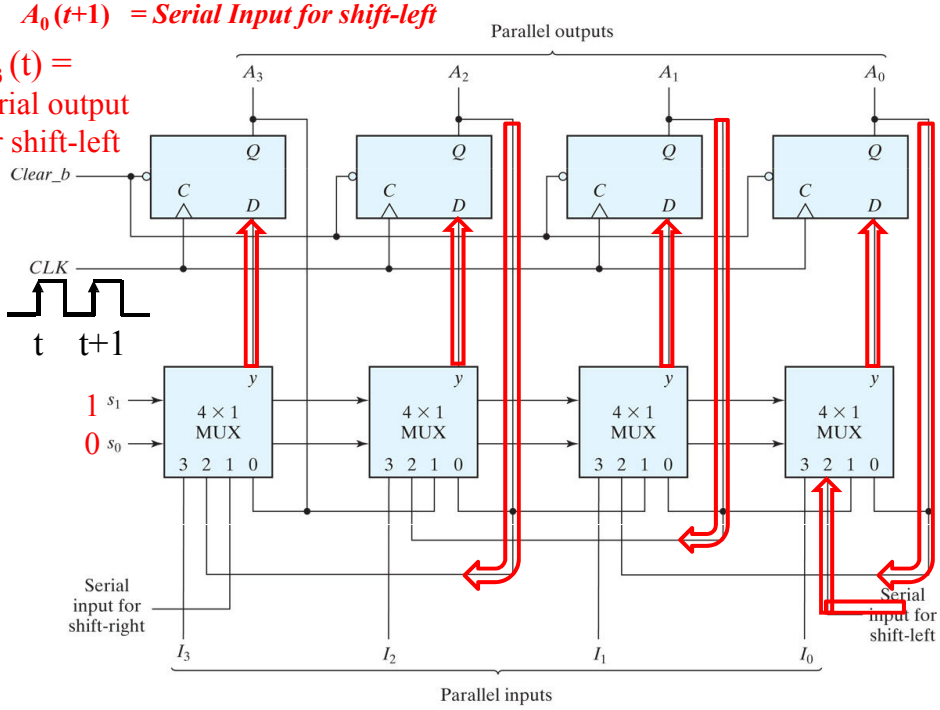
$A_0(t) =$ Serial output for shift-right

$S_1\ S_0 = 10$ ---> SHIFT LEFT

$A_{i+1}(t+1) = A_i(t)$ for $0 \le i \le 2$

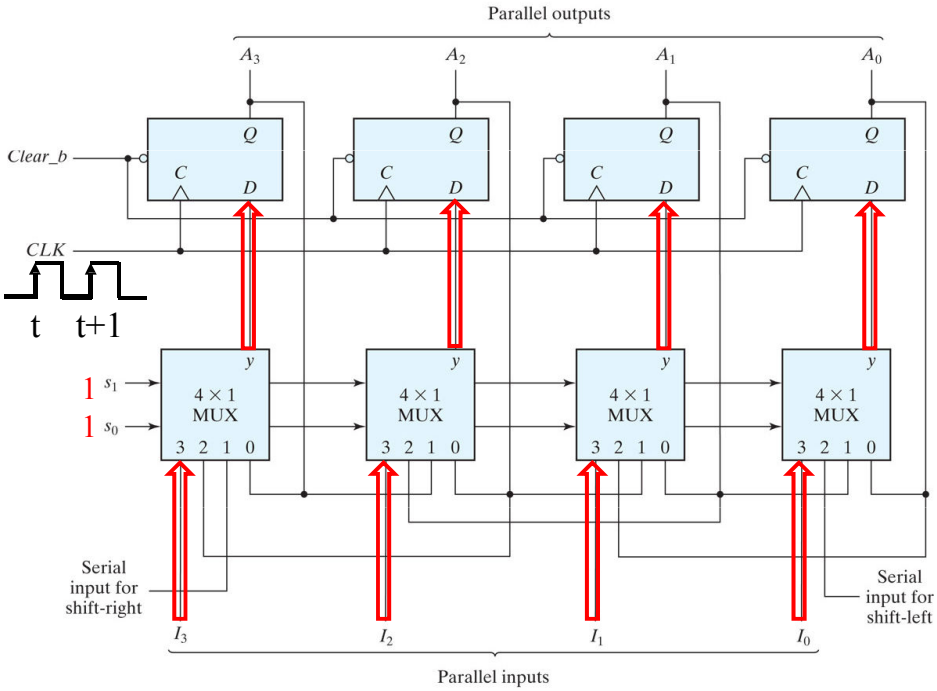
$A_0(t+1)$ = *Serial Input for shift-left*

$A_3(t) =$ Serial output for shift-left



$S_1\ S_0 = 11$ ---> PARALLEL LOAD

$A_i(t+1) = I_i(t)$ for $0 \le i \le 3$



COUNTER

A SEQUENTIAL CIRCUIT
(a cascade of JK or T FF's, such as a register)
THAT GOES THROUGH A PREDETERMINED SEQUENCE OF STATES

A) RIPPLE COUNTER

A counter in which CP inputs of all FF's (except the 1st FF) are triggered not by incoming clock pulses but rather by state transitions that occur in the preceding FF's

B) SYNCHRONOUS COUNTER

A counter in which CP inputs of all FF's are triggered by incoming clock pulses

Terminology for Counters:

MODULUS (or MOD) : The number of states in the count sequence

Maximum MOD : The maximum number of states = 2^n
where n is the number of bits used to represent a state

e.g., Mod-8 Binary Counter : Count Sequence is 0, 1, 2, 3, 4, 5, 6, 7, and repeat
Each state requires 3 bits to be represented, thus n =3 which means that maximum MOD is 8.

Sequential count sequence : natural numeric count

e.g., Count Sequence 0, 1, 2, 3, 4, 5, 6, 7 is a sequential count sequence.

Up/Down count sequence: Increasing (up) or decreasing (down) count sequence

e.g., 0, 1, 2, 3, 4, 5, 6, 7 is Up and 7, 6, 5, 4, 3, 2, 1, 0 is Down count sequence

Full count sequence : The number of states = Maximum Mod.

e.g., Mod-8 Binary counter with 3 bits used for representing states.

Truncated count sequence : The number of states < Maximum Mod.

e.g., Mod-6 Binary counter (0, 1, 2, 3, 4, 5) with 3 bits used for representing states
where states 6 and 7 are not used (missing states).
A truncated count sequence may or may not be sequential:
0, 1, 2, 3, 4, 5 or 1, 3, 4, 5, 6, 7 or 0, 2, 3, 5, 6, 7, etc.

A) RIPPLE COUNTER

1) BINARY RIPPLE COUNTER

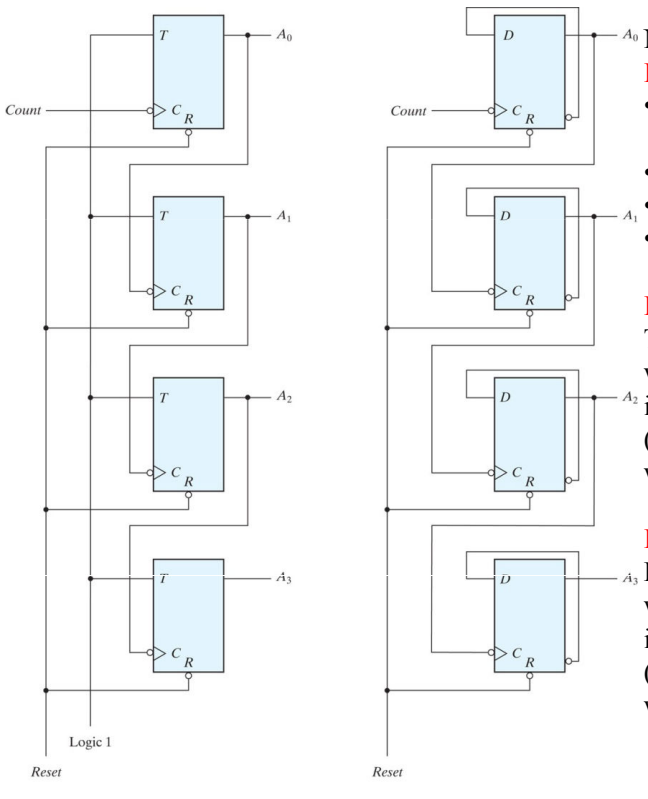
e.g., a binary ripple counter counting from 0 to 15 and repeating

Count Sequence

A_3	A_2	A_1	A_0
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

Note that
- Count starts with binary 0 and increments by 1 with each clock pulse
- A_0 is alternating with each clock pulse
- Each time A_0 goes from 1 to 0, it complements A_1
- Each time A_1 goes from 1 to 0, it complements A_2
- Each time A_2 goes from 1 to 0, it complements A_3

FIGURE 6.8 Four-bit binary ripple counter



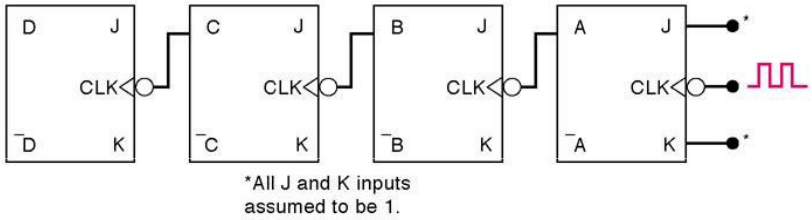
Note that

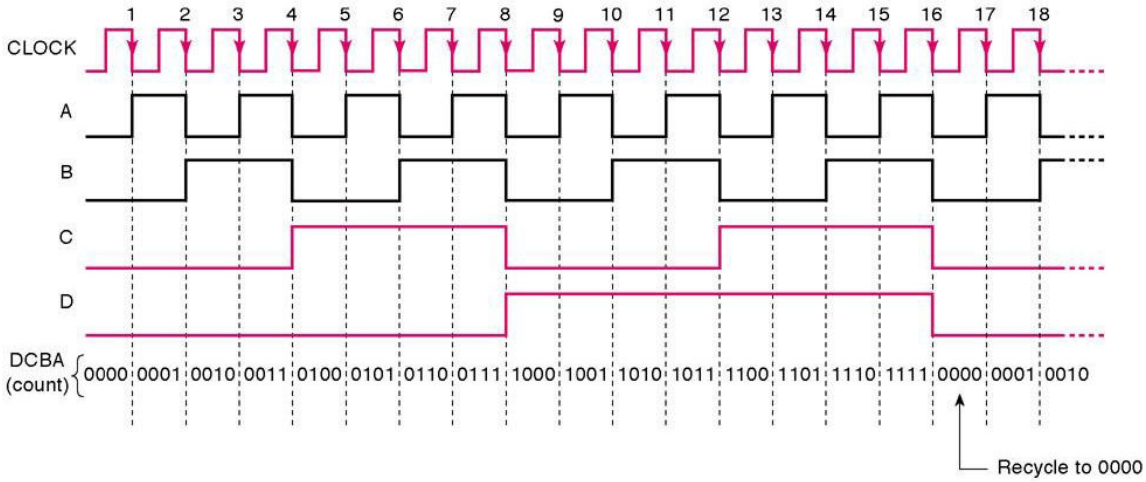
In both circuits ,
- FFs respond to the negative edge transition (from 1 to 0) (notice circle at C input of each FF)
- Output of each FF goes to the C input of the next FF
- A_0 (the FF holding the LSB) gets the count pulses
- When a FF goes from 1 to 0, it complements the next FF

In Figure 6.8.a),
T inputs of all FFs are connected to a permanent logic 1 which makes each FF complement if C input goes from 1 to 0 (this happens when the output of the previous FF goes from 1 to 0)

In Figure 6.8.b),
D input of each FFs is connected to its complement output which makes each FF complement its state if C input goes from 1 to 0 (this happens when the output of the previous FF goes from 1 to 0)

e.g., Four-bit binary ripple counter with JK-FFs
(with J and K inputs of each FF is connected to Logic 1)
Note that
if J=K=1, then JK-FF toggles (like T FF toggles when T=1)





2) BCD RIPPLE COUNTER

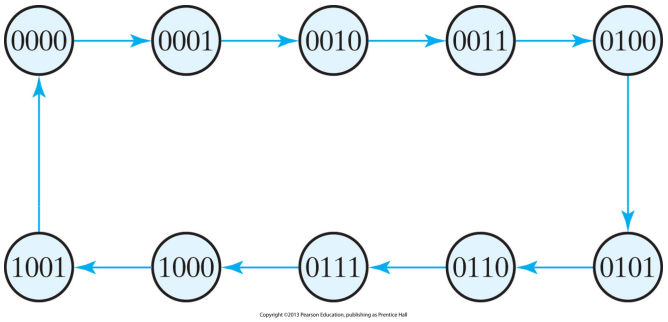


FIGURE 6.9 State diagram of a decimal BCD counter

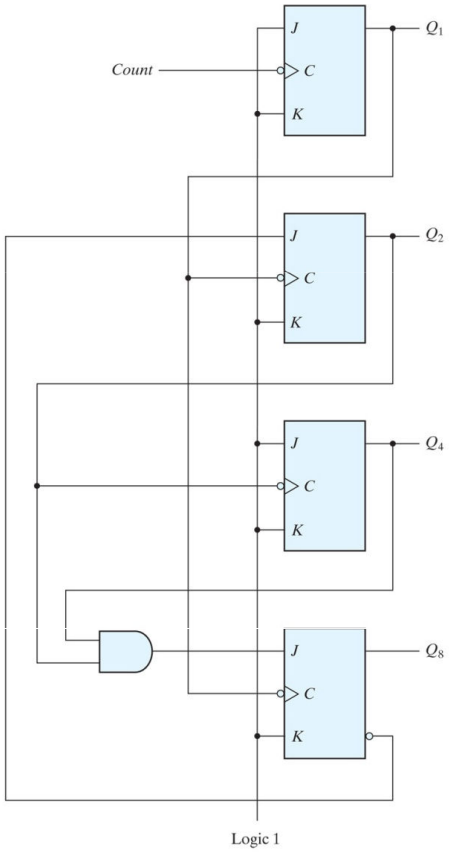


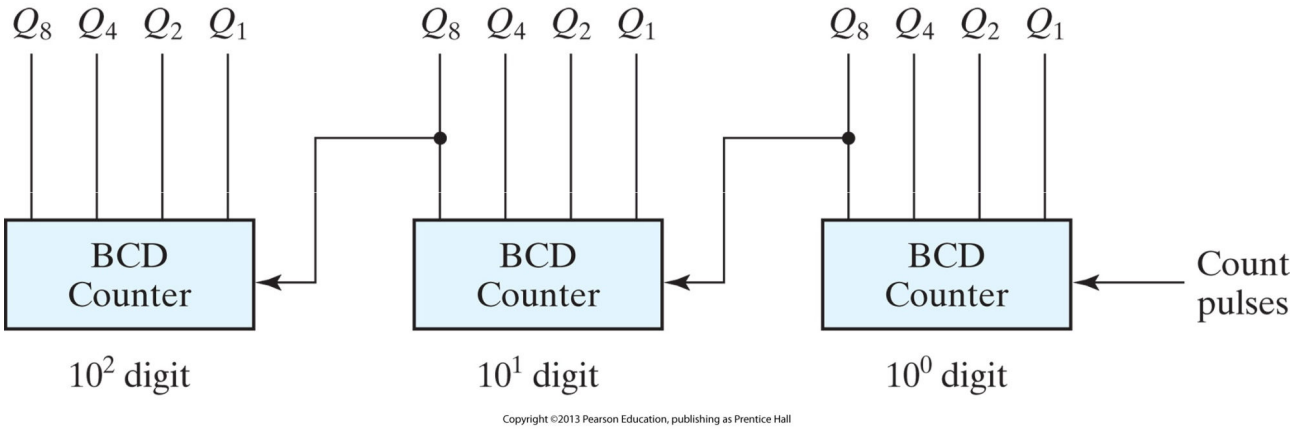
FIGURE 6.10 BCD ripple counter

if J = 1, SET. if K = 1, RESET
if J=K=0 then no change (hold)
if J=K=1 then toggle (complement)

Note that in Figure 6.10,
K = 1 for all FFs.
Thus, if J=1 also, the FF toggles
when C goes from 1 to 0.

From Figure 6.9, observe that
Q1 changes state after each count pulse.
Q2 complements each time Q1 goes from 1 to 0, as long as Q8 =0
Q2 remains at 0 when Q8 becomes 1.
Q4 complements each time Q2 goes from 1 to 0.
Q8 remains at 0 as long as Q2 or Q4 is 0.
When both Q2 and Q4 are 1, Q8 complements when Q1 goes from 1 to 0.
Q8 is reset on the next transition of Q1.

FIGURE 6.11 Block diagram of a three-decade decimal BCD counter



B) SYNCHRONOUS COUNTER

A counter in which CP inputs of all FF's are triggered by incoming clock pulses

1) BINARY SYNCHRONOUS COUNTER

LSB is complemented at each clock pulse.

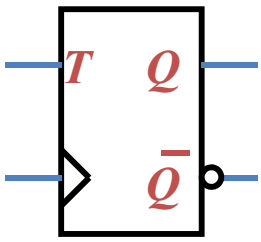
Other bits are complemented when all of its lower significant bits are 1.

e.g., a binary synchronous counter counting from 0 to 7 and repeating

Count Sequence	Present State	Next State
	A B C	A B C
0 0 0	0 0 0	0 0 1
0 0 1	0 0 1	0 1 0
0 1 0	0 1 0	0 1 1
0 1 1	0 1 1	1 0 0
1 0 0	1 0 0	1 0 1
1 0 1	1 0 1	1 1 0
1 1 0	1 1 0	1 1 1
1 1 1	1 1 1	0 0 0

e.g., Using T FFs, design a mod-8 binary synchronous counter

Count Sequence	Present State A B C	Next State A B C	Flip - Flop Inputs $T_A T_B T_C$
0 0 0	0 0 0	0 0 1	0 0 1
0 0 1	0 0 1	0 1 0	0 1 1
0 1 0	0 1 0	0 1 1	0 0 1
0 1 1	0 1 1	1 0 0	1 1 1
1 0 0	1 0 0	1 0 1	0 0 1
1 0 1	1 0 1	1 1 0	0 1 1
1 1 0	1 1 0	1 1 1	0 0 1
1 1 1	1 1 1	0 0 0	1 1 1



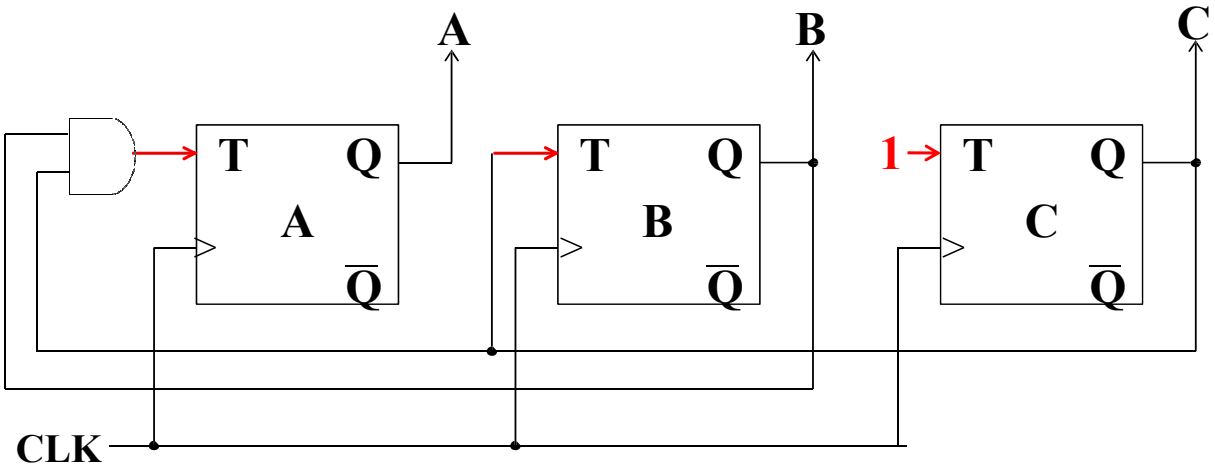
$Q(t)$	$Q(t+1)$	T
0	0	0
0	1	1
1	0	1
1	1	0

$$T_A = \Sigma\, m(3,7)$$
$$T_B = \Sigma\, m(1,3,5,7)$$
$$T_C = 1$$

$T_A = \Sigma\, m(3,7)$ = BC (using K-Maps)

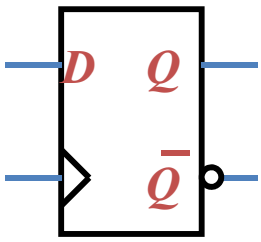
$T_B = \Sigma\, m(1,3,5,7)$ = C (using K-Maps)

$T_C = 1$



e.g., Using D FFs, design a mod-8 binary synchronous counter

Count Sequence	Present State A B C	Next State A B C	Flip - Flop Inputs $D_A D_B D_C$
0 0 0	0 0 0	0 0 1	0 0 1
0 0 1	0 0 1	0 1 0	0 1 0
0 1 0	0 1 0	0 1 1	0 1 1
0 1 1	0 1 1	1 0 0	1 0 0
1 0 0	1 0 0	1 0 1	1 0 1
1 0 1	1 0 1	1 1 0	1 1 0
1 1 0	1 1 0	1 1 1	1 1 1
1 1 1	1 1 1	0 0 0	0 0 0

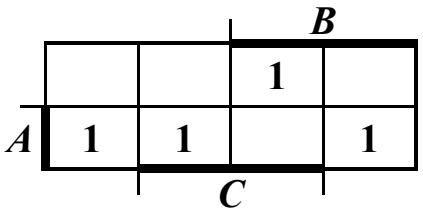


$Q(t)$	$Q(t+1)$	D
0	0	0
0	1	1
1	0	0
1	1	1

$$D_A = \Sigma\, m(3,4,5,6)$$
$$D_B = \Sigma\, m(1,2,5,6)$$
$$D_C = \Sigma\, m(0,2,4,6)$$

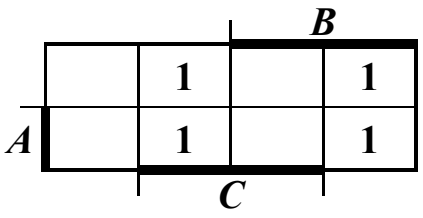
$$D_A = \Sigma\, m(3,4,5,6)$$

$$D_A = AB' + AC' + A'BC$$



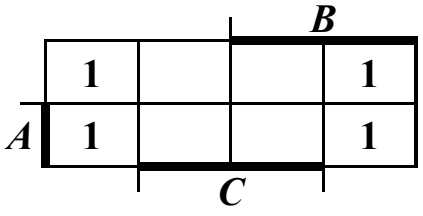
$$D_B = \Sigma\, m(1,2,5,6)$$

$$D_B = BC' + B'C$$



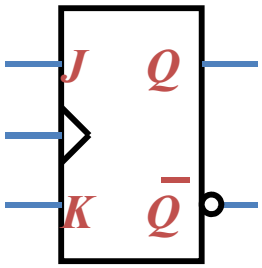
$$D_C = \Sigma\, m(0,2,4,6)$$

$$D_C = C'$$



e.g., Using JK FFs, design a mod-8 binary synchronous counter

Count Sequence	Present State	Next State	Flip - Flop Inputs
	A B C	A B C	J_AK_A J_BK_B J_CK_C
0 0 0	0 0 0	0 0 1	0 X 0 X 1 X
0 0 1	0 0 1	0 1 0	0 X 1 X X 1
0 1 0	0 1 0	0 1 1	0 X X 0 1 X
0 1 1	0 1 1	1 0 0	1 X X 1 X 1
1 0 0	1 0 0	1 0 1	X 0 0 X 1 X
1 0 1	1 0 1	1 1 0	X 0 1 X X 1
1 1 0	1 1 0	1 1 1	X 0 X 0 1 X
1 1 1	1 1 1	0 0 0	X 1 X 1 X 1



$Q(t)$	$Q(t+1)$	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

$J_A = \Sigma\, m(3), d(4,5,6,7)$ $K_A = \Sigma\, m(7), d(0,1,2,3)$

$J_B = \Sigma\, m(1,5), d(2,3,6,7)$ $K_B = \Sigma\, m(3,7), d(0,1,4,5)$

$J_C = 1$ $K_C = 1$

$J_A = \Sigma\, m(3), d(4,5,6,7)$ $K_A = \Sigma\, m(7), d(0,1,2,3)$

$J_B = \Sigma\, m(1,5), d(2,3,6,7)$ $K_B = \Sigma\, m(3,7), d(0,1,4,5)$

$J_C = 1$ $K_C = 1$

A \ BC	00	01	11	10
0			1	
1	X	X	X	X

$J_A = BC$

A \ BC	00	01	11	10
0	X	X	X	X
1			1	

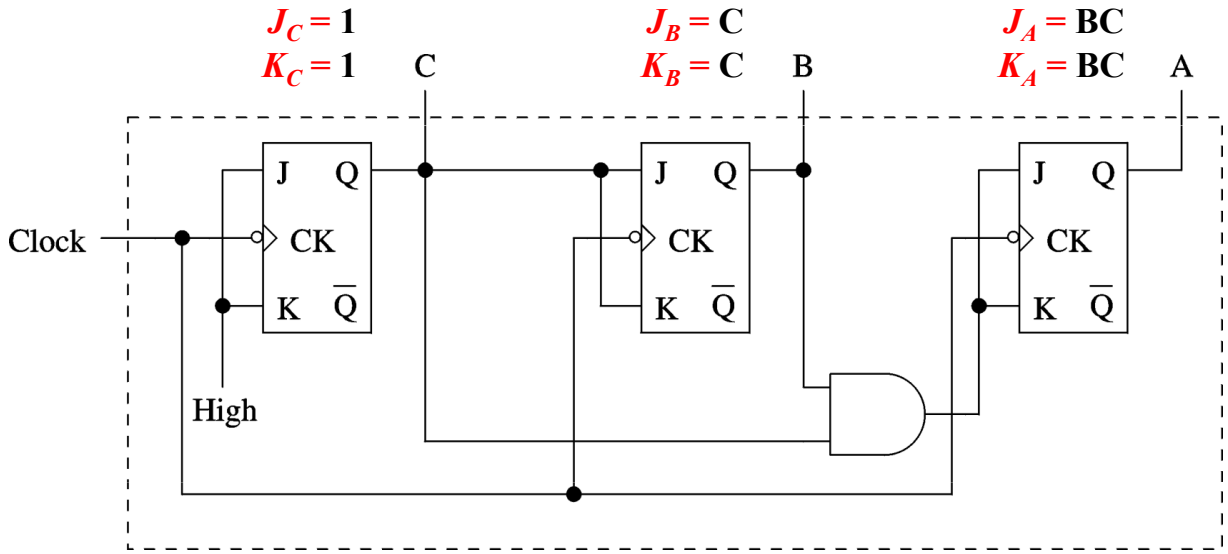
$K_A = BC$

A \ BC	00	01	11	10
0		1	X	X
1		1	X	X

$J_B = C$

A \ BC	00	01	11	10
0	X	X	1	
1	X	X	1	

$K_B = C$

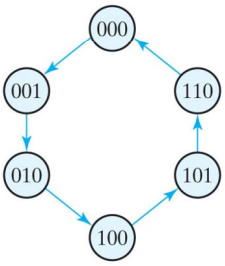


Binary Synchronous Counter with Missing States

e.g., Design a counter with count sequence 0, 1, 2, 4, 5, 6 (and repeat) by using JK-FFs

Note that states 3 and 7 (i.e., 011 and 111 are missing (not used))

State Diagram



State Table

Present State			Next State			Flip-Flop Inputs					
A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	1	0	0	1	X	X	1	0	X
1	0	0	1	0	1	X	0	0	X	1	X
1	0	1	1	1	0	X	0	1	X	X	1
1	1	0	0	0	0	X	1	X	1	0	X

$Q(t)$	$Q(t+1)$	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Minterms	Present State			Next State			Flip-Flop Inputs					
	A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	0	1	0	X	0	X	1	X
1	0	0	1	0	1	0	0	X	1	X	X	1
2	0	1	0	1	0	0	1	X	X	1	0	X
4	1	0	0	1	0	1	X	0	0	X	1	X
5	1	0	1	1	1	0	X	0	1	X	X	1
6	1	1	0	0	0	0	X	1	X	1	0	X
3							X	X	X	X	X	X
7							X	X	X	X	X	X

For minterms 3 and 7, J and K inputs of all FFs are DONT CARES

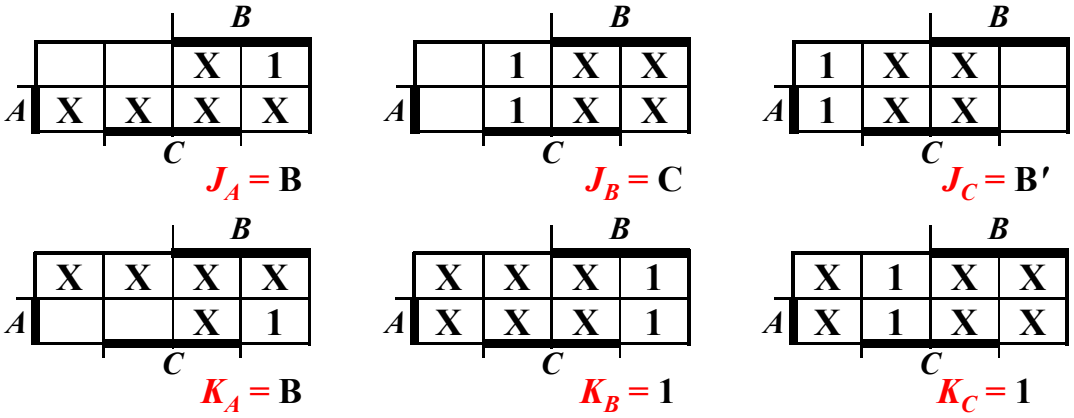
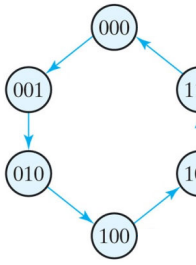


FIGURE 6.16 Counter with unused states



$J_A = B$

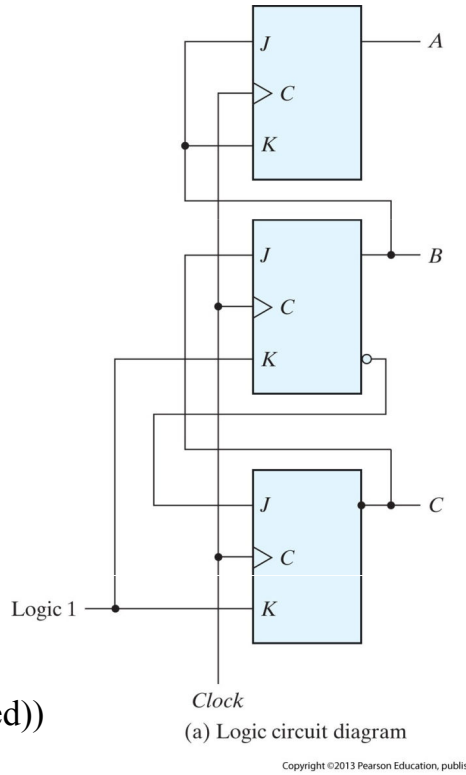
$K_A = B$

$J_B = C$

$K_B = 1$

$J_C = B'$

$K_C = 1$



(a) Logic circuit diagram

Note that states 3 and 7
(i.e., 011 and 111 are missing (not used))

What if the counter goes into state 3 or state 7?
Recall that states 3 and 7 (i.e., 011 and 111 are missing (not used))

If the counter goes into state 3 (i.e., 011) or state 7 (i.e. 11110,
what is the next state, when the next clock pulse arrives?

$J_A = B$ $J_B = C$ $J_C = B'$

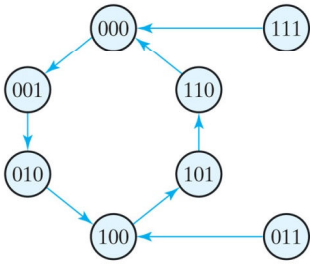
$K_A = B$ $K_B = 1$ $K_C = 1$

For state 011, we put A(t)=0, B(t)=1, C(t)=1
into above FF input functions and come up with:
A(t+1)= A(t)′ =1 because J_A =1 and K_A =1.
B(t+1)= B(t)′ =0 because J_B =1 and K_B =1.
C(t+1)= 0 because J_C = 0 and K_C =1.

Next state is 100

For state 111, we put A(t)=1, B(t)=1, C(t)=1
into above FF input functions and come up with:
A(t+1)= A(t)′ =0 because J_A =1 and K_A =1.
B(t+1)= B(t)′ =0 because J_B =1 and K_B =1.
C(t+1)= 0 because J_C = 0 and K_C =1.

Next state is 000



SELF CORRECTING COUNTER

e.g., Using JK FFs, design a mod-16 binary synchronous counter

Present State	Next State	Flip Flop Input Functions			
$A_3 A_2 A_1 A_0$	$A_3 A_2 A_1 A_0$	$J_{A3}K_{A3}$	$J_{A2}K_{A2}$	$J_{A1}K_{A1}$	$J_{A0}K_{A0}$
0 0 0 0	0 0 0 1	0 X	0 X	0 X	1 X
0 0 0 1	0 0 1 0	0 X	0 X	1 X	X 1
0 0 1 0	0 0 1 1	0 X	0 X	X 0	1 X
0 0 1 1	0 1 0 0	0 X	1 X	X 1	X 1
0 1 0 0	0 1 0 1	0 X	X 0	0 X	1 X
0 1 0 1	0 1 1 0	0 X	X 0	1 X	X 1
0 1 1 0	0 1 1 1	0 X	X 0	X 0	1 X
0 1 1 1	1 0 0 0	1 X	X 1	X 1	X 1
1 0 0 0	1 0 0 1	X 0	0 X	0 X	1 X
1 0 0 1	1 0 1 0	X 0	0 X	1 X	X 1
1 0 1 0	1 0 1 1	X 0	0 X	X 0	1 X
1 0 1 1	1 1 0 0	X 0	1 X	X 1	X 1
1 1 0 0	1 1 0 1	X 0	X 0	0 X	1 X
1 1 0 1	1 1 1 0	X 0	X 0	1 X	X 1
1 1 1 0	1 1 1 1	X 0	X 0	X 0	1 X
1 1 1 1	0 0 0 0	X 1	X 1	X 1	X 1

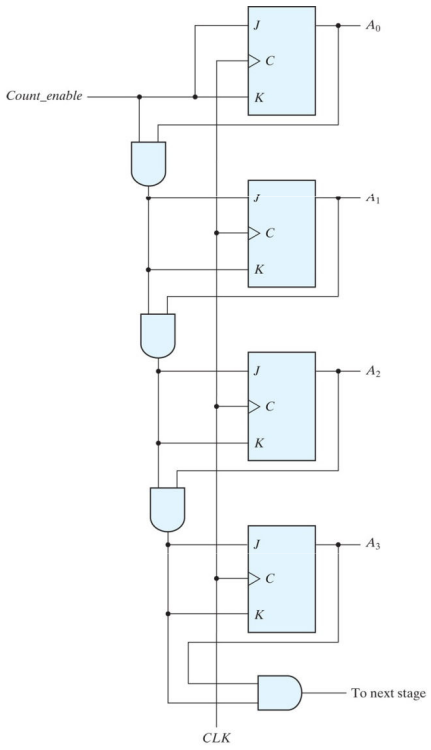
$Q(t)$	$Q(t+1)$	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Using K-Maps,
one can simplify
Flip Flop Input Functions

$J_{A3} = A_2 A_1 A_0$
$K_{A3} = A_2 A_1 A_0$
$J_{A2} = A_1 A_0$
$K_{A2} = A_1 A_0$
$J_{A1} = A_0$
$K_{A1} = A_0$
$J_{A0} = 1$
$K_{A0} = 1$

FIGURE 6.12 Four-bit synchronous binary counter

$J_{A3} = A_2 A_1 A_0$
$K_{A3} = A_2 A_1 A_0$
$J_{A2} = A_1 A_0$
$K_{A2} = A_1 A_0$
$J_{A1} = A_0$
$K_{A1} = A_0$
$J_{A0} = 1$
$K_{A0} = 1$



2) BINARY SYNCHRONOUS UP-DOWN COUNTER

Count-Up Counter

Goes trough the binary states in ascending (increasing) order.

LSB is complemented at each clock pulse.

Each other bit is complemented when all of its lower significant bits are 1.

e.g., all counters seen so far are countup counters

Count-Down Counter

Goes trough the binary states in descending (decreasing) order

LSB is complemented at each clock pulse.

Each other bit is complemented when all of its lower significant bits are 0.

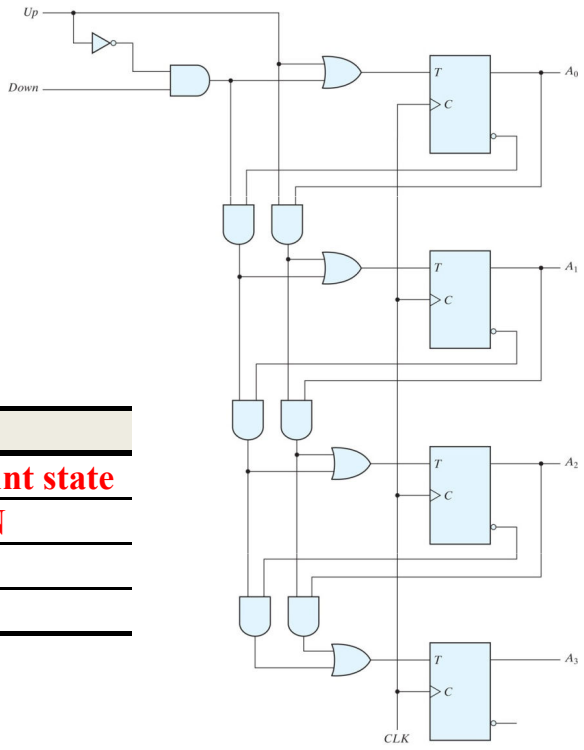
e.g., A mod-16 binary synchronous countdown counter with JK FFs

Same as the one shown in the previous slide (Figure 6.12),
except that
the inputs to the AND gates must come from
the complemented outputs, instead of the normal outputs of the FFs.

Up-Down Counter

A counter that counts either up or down

FIGURE 6.13 Four-bit up–down binary counter



Up	*Down*	Circuit
0	0	Remains at the same count state
0	1	COUNTS DOWN
1	0	COUNTS UP
1	1	COUNTS UP

3) BCD SYNCHRONOUS COUNTER

Counts from 0 to 9

e.g., A BCD Synchronous Counter using T FFs

$Q(t)$	$Q(t+1)$	T
0	0	0
0	1	1
1	0	1
1	1	0

Table 6.5
State Table for BCD Counter

Present State				Next State				Output	Flip-Flop Inputs			
Q_8	Q_4	Q_2	Q_1	Q_8	Q_4	Q_2	Q_1	y	TQ_8	TQ_4	TQ_2	TQ_1
0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	1	0	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	0	1
0	0	1	1	0	1	0	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	0	1
0	1	0	1	0	1	1	0	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	0	1
0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	0	1
1	0	0	1	0	0	0	0	1	1	0	0	1

Using the State Table in Table 6.5,
where minterms m10, to m15 are DONT CARES are not explicitly shown
one can form the K-Maps for flipflop input functions

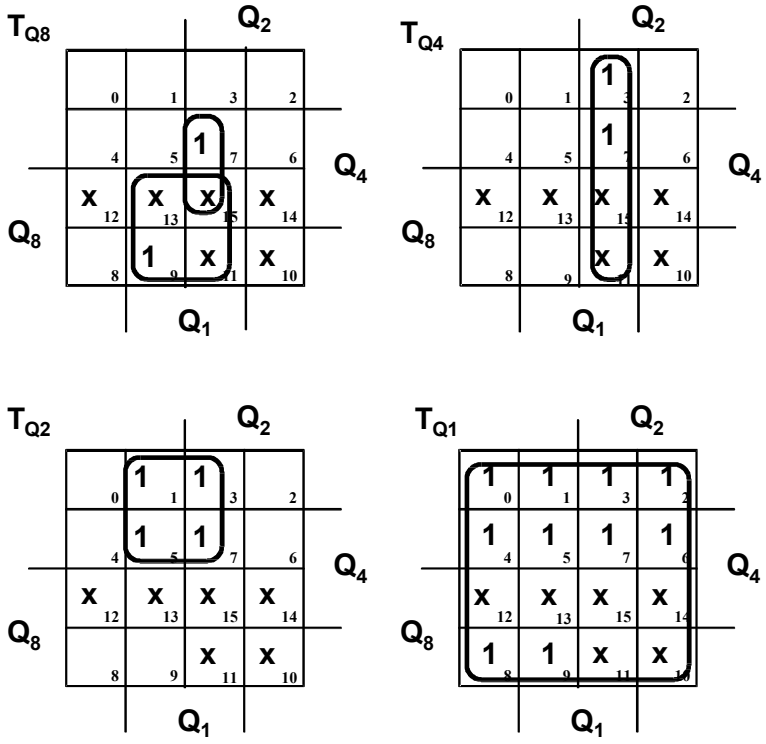
$T_{Q8} = Q_8Q_1 + Q_4Q_2Q_1$

$T_{Q4} = Q_2Q_1$

$T_{Q2} = Q_8'Q_1$

$T_{Q1} = 1$

$y = Q_8Q_1$



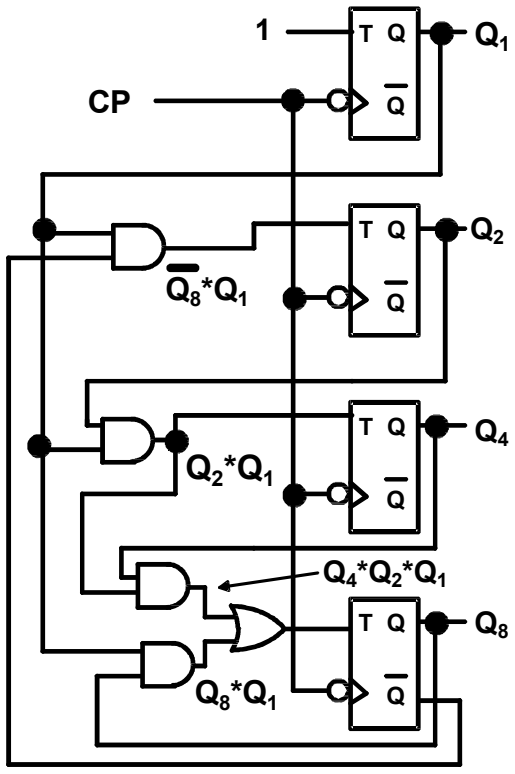
$T_{Q8} = Q_8Q_1 + Q_4Q_2Q_1$

$T_{Q4} = Q_2Q_1$

$T_{Q2} = Q_8'Q_1$

$T_{Q1} = 1$

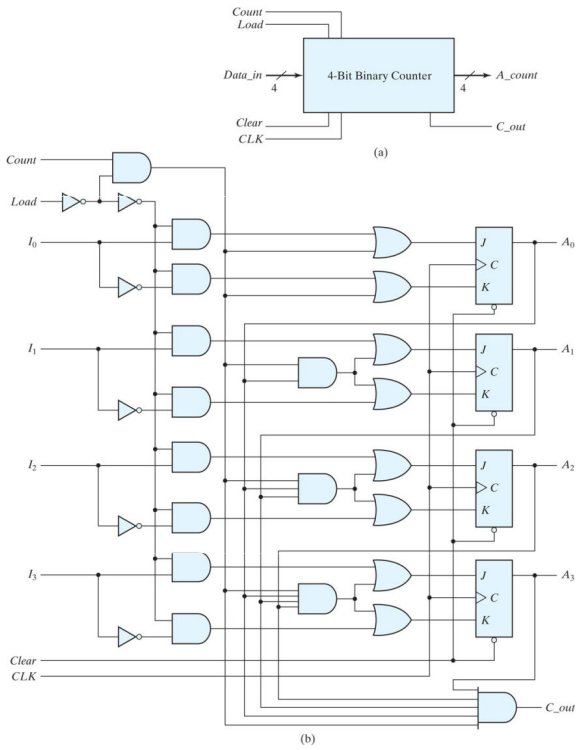
$y = Q_8Q_1$



4) BINARY SYNCHRONOUS COUNTER WITH PARALLEL LOAD

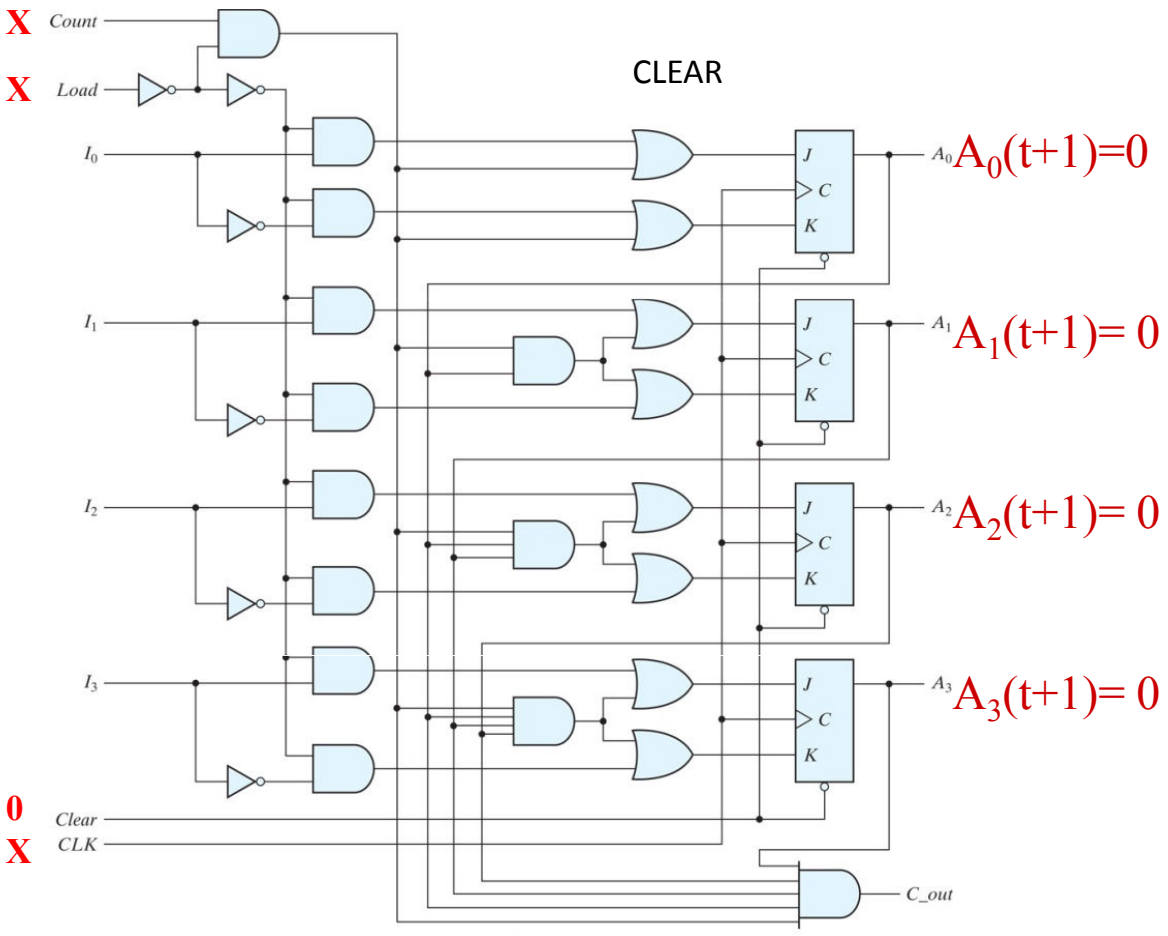
Table 6.6
Function Table for the Counter of Fig. 6.14

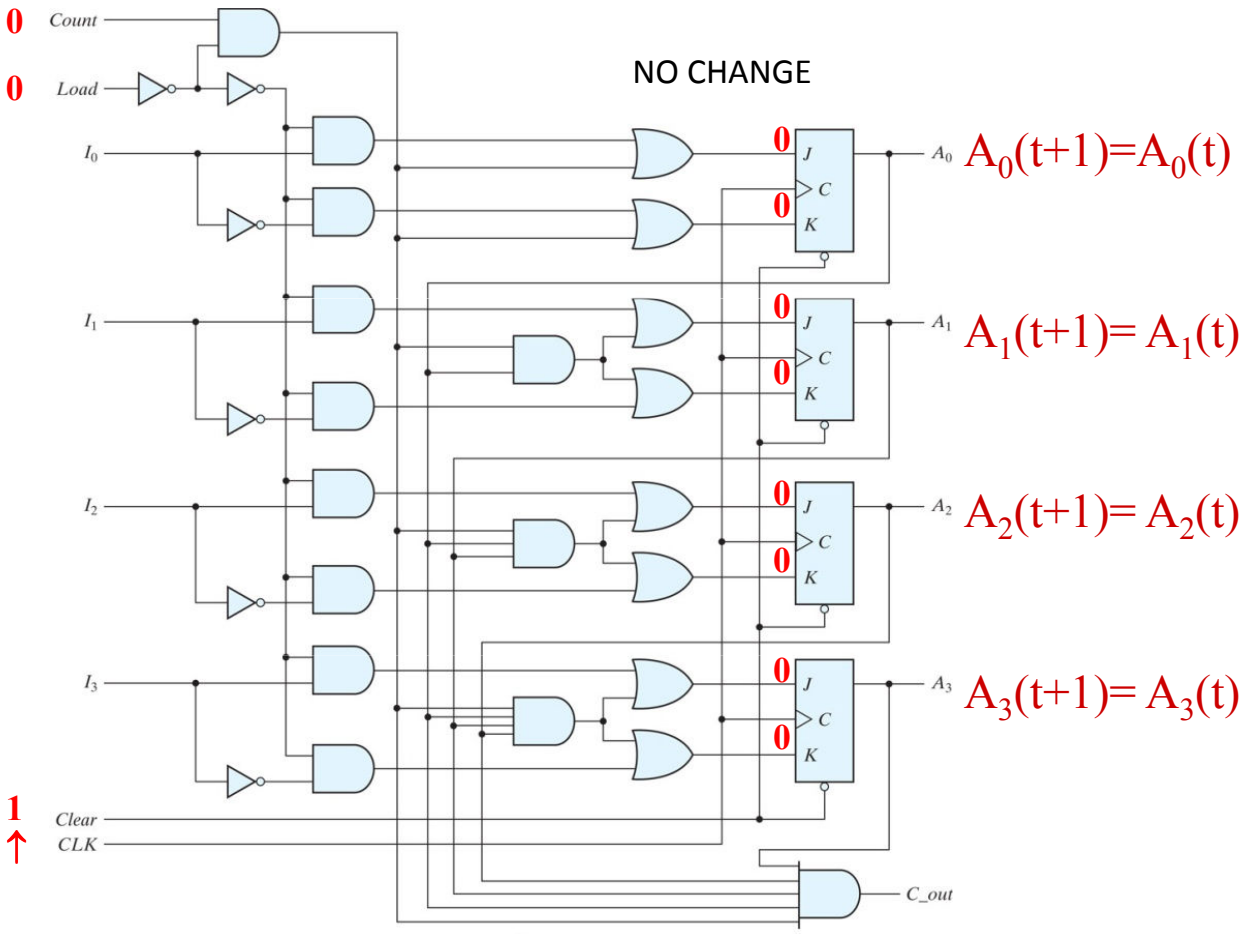
Clear	CLK	Load	Count	Function
0	X	X	X	Clear to 0
1	↑	1	X	Load inputs
1	↑	0	1	Count next binary state
1	↑	0	0	No change

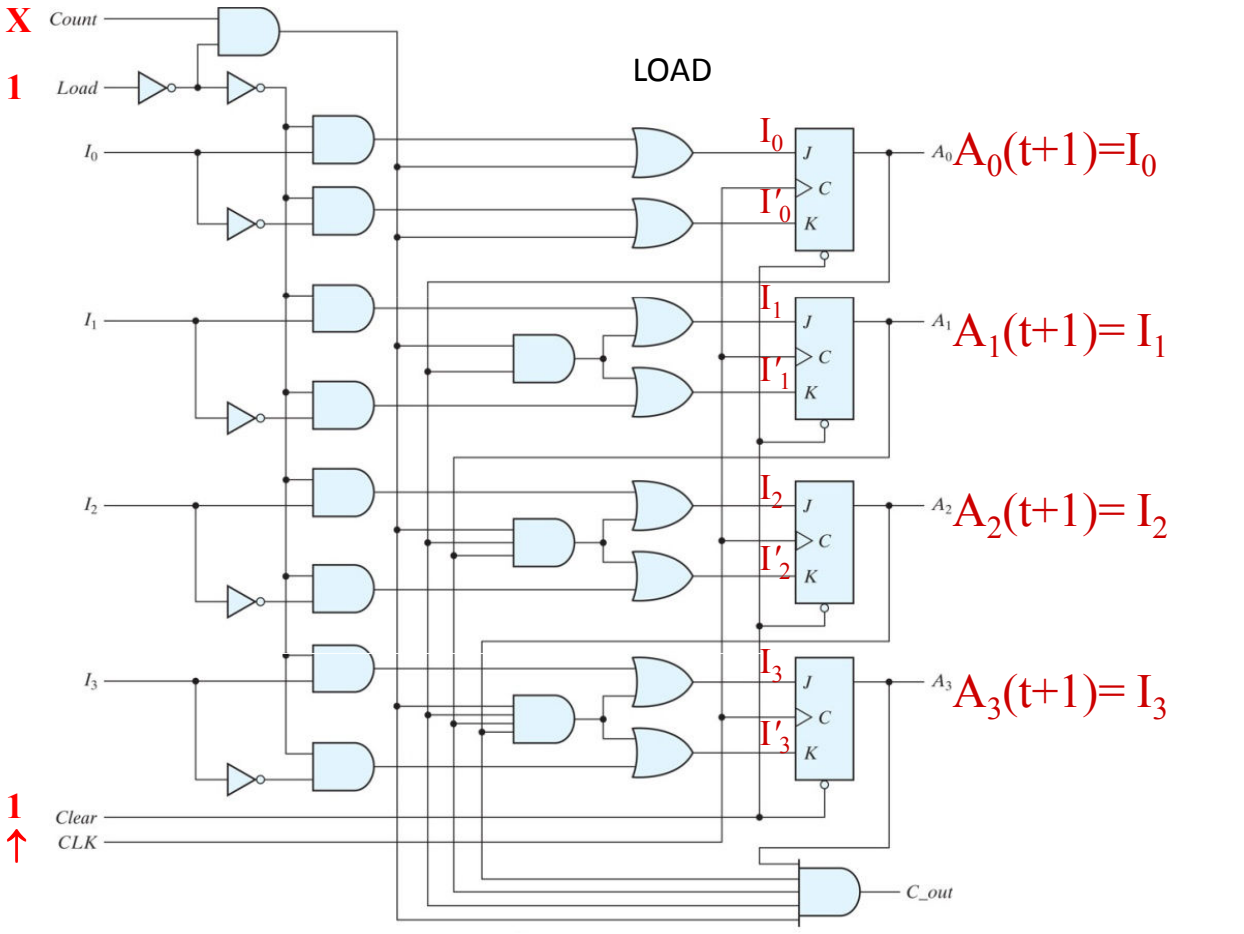


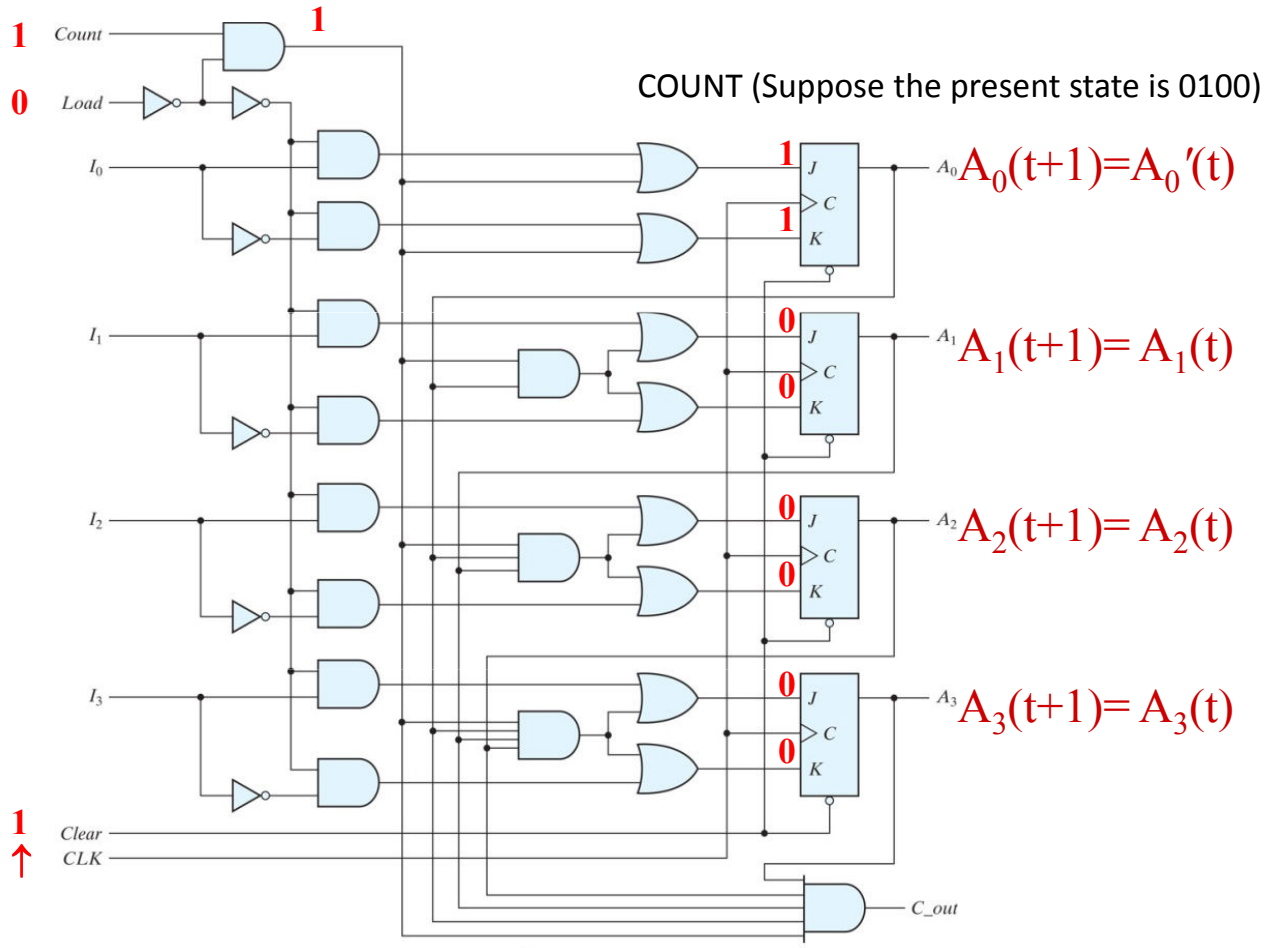
C-out= Count $A_3 A_2 A_1 A_0$

FIGURE 6.14 Four-bit binary counter with parallel load




NO CHANGE
0
Count
0
Load
I_0
I_1
I_2
I_3
1
Clear
CLK
J
C
K
A_0
A_1
A_2
A_3
C_out
$A_0(t+1)=A_0(t)$
$A_1(t+1)= A_1(t)$
$A_2(t+1)= A_2(t)$
$A_3(t+1)= A_3(t)$


LOAD
X
Count
1
Load
I_0
I_1
I_2
I_3
1
Clear
CLK
J
C
K
A_0
A_1
A_2
A_3
C_out
$A_0(t+1)=I_0$
$A_1(t+1)= I_1$
$A_2(t+1)= I_2$
$A_3(t+1)= I_3$



A BINARY SYNCHRONOUS COUNTER WITH PARALLEL LOAD
CAN BE USED TO CONSTRUCT ANY MOD-N COUNTER
e.g., BCD Synchronous Counter

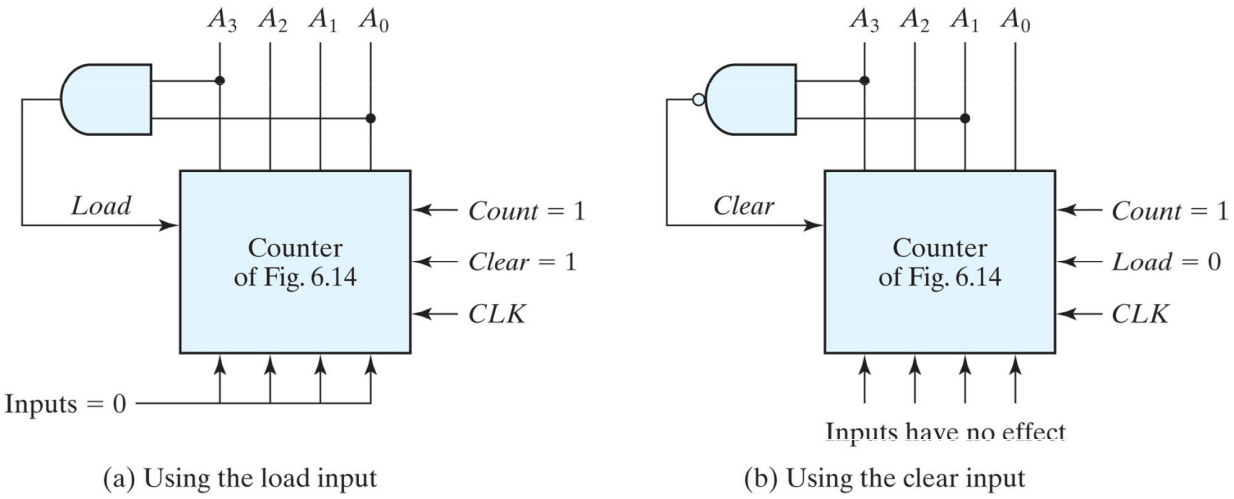


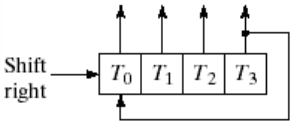
FIGURE 6.15 Two ways to achieve a BCD counter using a counter with parallel load

RING COUNTER

A ring counter takes the serial output of the last Flip-Flop of a shift register and submits it to the serial input of the first Flip-Flop.

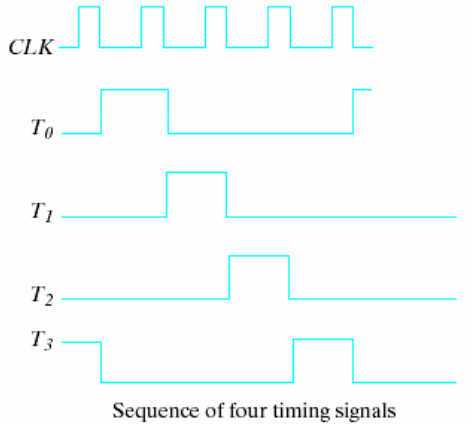
It is also called a re-circulating shift register where only one flip flop is set at any particular time while all other flip flops are cleared.

e.g., 4-bit shift register connected as a ring counter



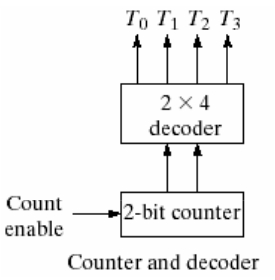
Initial state is 1000.
1is shifted right at each clock pulse and circulates from T3 to T0.

Since each FF is 1 only once at every 4 clock pulses, it generates four timing signals:

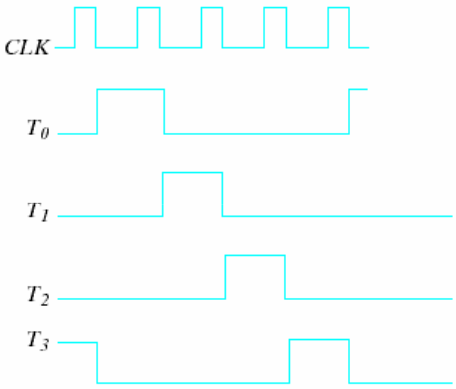


Sequence of four timing signals

The same four timing signals can be generated by 2-bit counter :



Counter and decoder



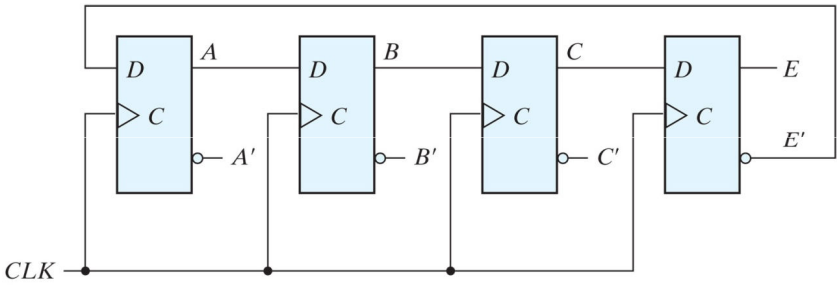
Sequence of four timing signals

The 2-bit counter goes through states 00, 01, 10, 11
and
the 2x4 decoder decodes these four states and
generates the required timing sequence.

In summary, to generate 2^n timing signals
we need either a shift register with 2^n FFs
or
an n-bit binary counter and an $nx2^n$ decoder.

JOHNSON COUNTER

A Johnson Counter re-circulates the complement output of the last flip-flop back to the input of the first Flip-Flop.



(a) Four-stage switch-tail ring counter

A K-bit switch-tail ring counter goes through a sequence of 2K states (in contrast, a K-bit ring counter goes through a sequence of K states).

Sequence number	Flip-flop outputs A	B	C	E
1	0	0	0	0
2	1	0	0	0
3	1	1	0	0
4	1	1	1	0
5	1	1	1	1
6	0	1	1	1
7	0	0	1	1
8	0	0	0	1

Starting from all 0s, each shift operation inserts 1s from left until all FFs are filled with 1s.
Then, 0s are inserted from left until all FFs are filled with 0s.

Johnson counter can provide 2k timing signals with 2k decoding gates. Decoding gates are specified in the table below.

Sequence number	Flip-flop outputs A	B	C	E	AND gate required for output
1	0	0	0	0	$A'E'$
2	1	0	0	0	AB'
3	1	1	0	0	BC'
4	1	1	1	0	CE'
5	1	1	1	1	AE
6	0	1	1	1	$A'B$
7	0	0	1	1	$B'C$
8	0	0	0	1	$C'E$

Johnson counter can be constructed for any no. of timing sequences.

Number of flip flops needed is half the number of timing signals.

Number of decoding gates is equal to number of timing signals and only 2-input gates are needed.