TSN1101
Computer Architecture and Organization

Section A (Digital Logic Design)

Lecture 06

[redacted] Sequential Logic Circuits

- Counter and Shift Registers

TOPIC COVERAGE IN THE LECTURE

- Asynchronous/Ripple Counters
  - 2-bit, 3-bit, 4-bit, 6-bit and 10-bit Counters
- Design of Synchronous/Parallel Counters
- Basic Shift Register Functions
- Types of Shift Registers
  - Serial In/Serial Out (SISO)
  - Serial In/Parallel Out (SIPO)
  - Parallel In/Serial Out (PISO)
  - Parallel In/Parallel Out (PIPO)
- Shift Register Counters
  - Johnson Counter
  - Ring Counter

Sequential Logic Circuits - Asynchronous Counters

* Difference between asynchronous and synchronous counters
* Reason for calling asynchronous counter as ripple counter
* Asynchronous binary UP or DOWN counter
    - Examples
    * 2-bit (Mod 4)
    * 3-bit (Mod 8)
    * 4- bit (Mod 16)
* Asynchronous (< Mod 2^N)counters - Examples
    * 6- bit (Mod 6)
    * 10-bit (Mod 10)

Counters
- Introduction

- The flip-flop is a basic building block for counters, registers, and other sequential control logic circuits.
  - Flip-flops can be connected together to perform counting operations.
- A sequential circuit that goes through a prescribed sequence of states upon the application of input pulses is called a counter
  - Sequence of states can follow binary number sequence (called binary counter) or any other sequence of states.
  - An n-bit binary counter consists of n flip-flops and can count in binary from 0 through 2^n - 1
- There are two types of counters according to the way they are clocked.
  - ❖ asynchronous (or ripple) counters
  - ❖ synchronous (or parallel) counters
- Difference between Asynchronous and Synchronous Counters
  - In asynchronous (ripple) counters, the first flip flop is clocked by the external clock pulse and then each successive flip-flop is clocked by the output of the preceding flip-flop.
  - In synchronous (parallel) counters, the clock input is connected to all the flip-flops so that they are clocked simultaneously.

Asynchronous Counters - Introduction

- Asynchronous counters (ripple counters) do not have a common clock pulse hence they do not change states at the same time.
- The asynchronous counter are also known as **ripple counters** because the input clock pulse ripples through the counter, which makes cumulative delay. The effect of clock pulse is felt on each consecutive stage of counter like ripples in water.
- Up counters
  - Example: 2-bit: count the sequence 00 to 11
- Down counters
  - Example: 2-bit: count the sequence 11 to 00

Asynchronous Counters
- 2-bit (Mod 4) Asynchronous binary UP counter (Construction)

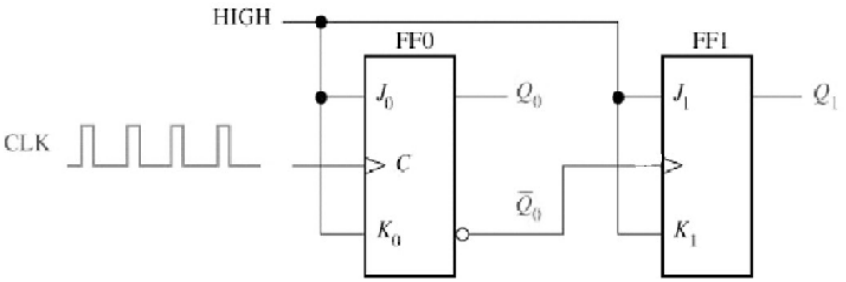
- A binary ripple counter consists of a series connection of complementing flip-flops, with the output of each flip-flop connected to the Clock (C) input of the next higher-order flip-flop.
- The flip-flop holding the LSB receives the incoming count pulses.
- Obtaining complementing flip-flop
  - Using T flip-flop by connecting T input to logic 1.
  - Using J-K flip-flops in Toggle mode by connecting J and K inputs together and giving logic 1.
  - Using D flip-flop with complementary output connected to D (Data) input.
    - D input is always the complement of present state and next clock pulse will cause the flip-flop to complement

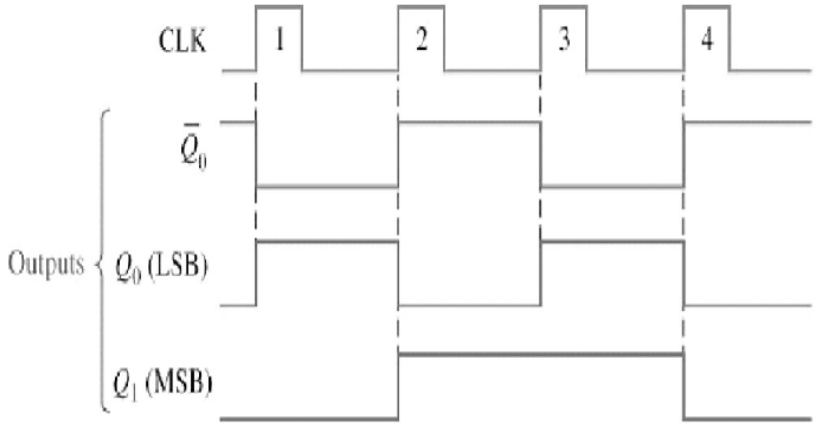
.

Asynchronous Counters

- 2-bit (Mod 4) Asynchronous binary UP counter (Implementation - Example)

N(clock)	Q_1	Q_0
0(initial)	0	0
1	0	1
2	1	0
3	1	1
4(recycle)	0	0





- Two positive-edge-triggered T flip-flops or JK flip-flops are connected in cascade configuration.
- Clock pulse is applied to the clock input C of the first flip flop which is always the LSB.
- The second flip flop FF1 is triggered by the Q_0' output of FF0.
- There is a propagation delay time through a flip-flop hence a transition of the Q_0' output of FFO can never occur at exactly the same time.
- If the counter uses negative edge-triggered flip-flops, Q_0 output can be connected to clock pulse.

Asynchronous Counters
- 3-bit (Mod 8)Asynchronous binary UP counter

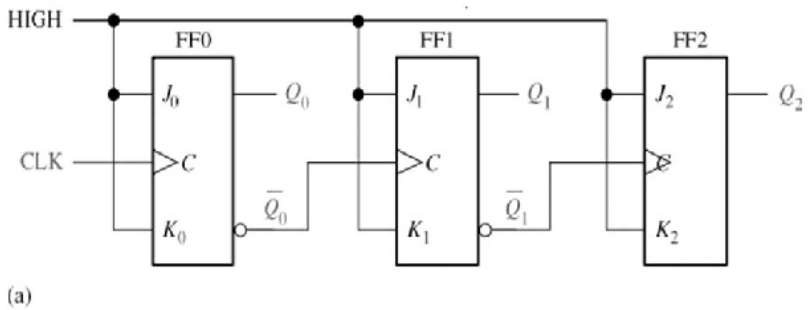
- A 3 bit counter has eight states. The counter starts through a binary count of zero through seven and then recycles to the zero state.
- The state sequence for a 3 bit binary counter is given below

CLOCK PULSE	Q_2 (MSB)	Q_1	Q_0 (LSB)
0 (initial)	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8 (recycles)	0	0	0

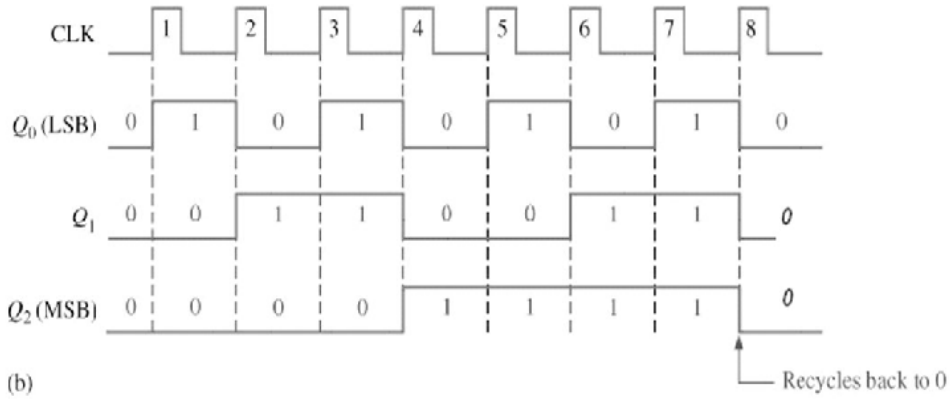
- The least significant bit Q_0 is complemented with each count pulse input
- Every time that Q_0 goes from 1to 0, it complements Q_1
- Every time that Q_1 goes from 1 to 0, it complements Q_2
- This continues for any other higher order bits of a ripple counter.

- For example, consider the transition from count 011 to 100.
- Q_0 is complemented with every count pulse.
- Since Q_0 goes from 1 to 0, it triggers Q_1, and complements it.
- As a result, Q_1 goes from 1 to 0, which in turn complements Q_2 ,changing it from 0 to 1.

Asynchronous Counters
- 3-bit (Mod 8) Asynchronous binary UP counter (Implementation - Example)



(a)



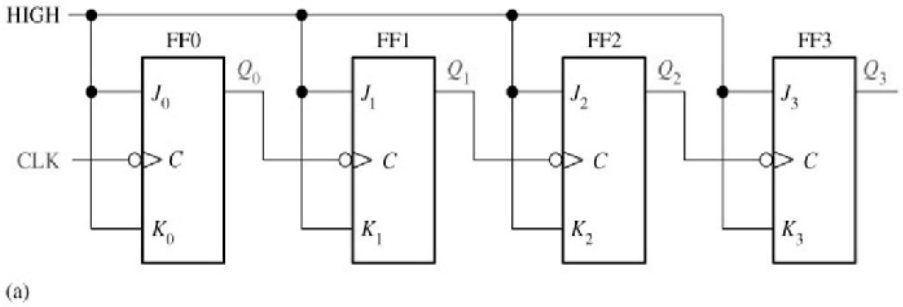
(b)

- Three positive-edge-triggered T flip-flops or JK flip-flops are connected in cascade configuration.
- Clock pulse is applied to the clock input C of the first flip flop which is always the LSB.
- The second flip flop FF1 is triggered by the Q_0' output of FF0.
- The third flip flop FF2 is triggered by the Q_1' output of FF1.

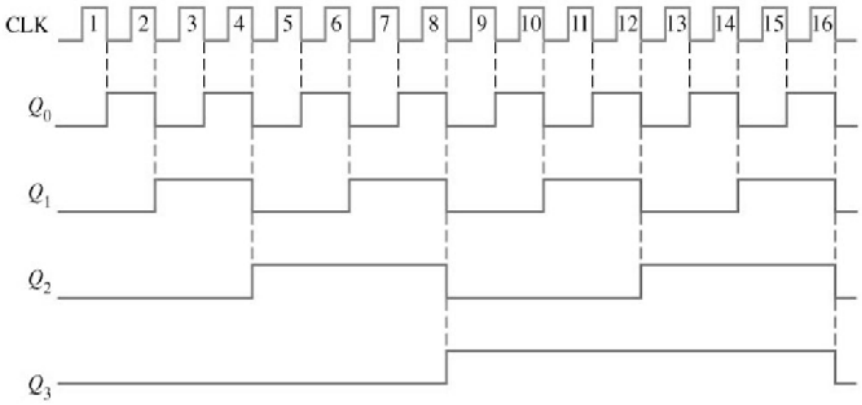
- If the counter uses negative edge-triggered flip-flops, Q_0 output can be connected to clock pulse of FF1 instead of Q_0' and Q_1 output can be connected to clock pulse of FF2 instead of Q_1'

Asynchronous Counters

- 4-bit (Mod 16) Asynchronous binary UP counter (Implementation - Example)



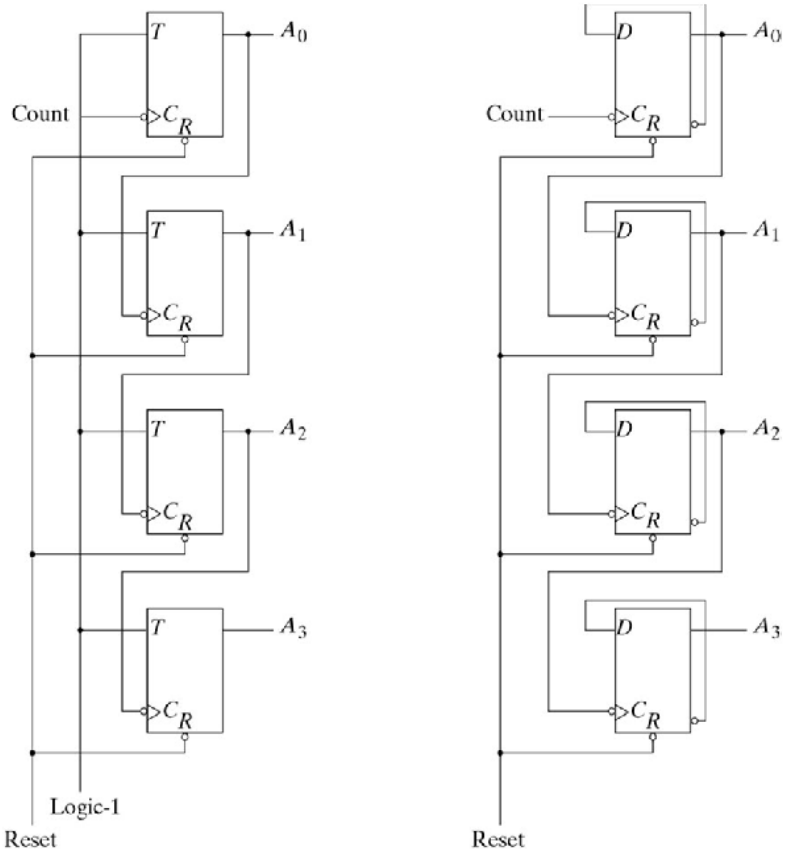
(a)



(b)

Asynchronous Counters

- 4-bit (Mod 16)Asynchronous binary UP counter (Construction with T flip-flops and D flip-flops)



(a) With T flip-flops

(b) With D flip-flops

Asynchronous Counters
- Asynchronous binary DOWN counter

- A **down, or backward**, counter is one whose state transitions are reversed from those of the **standard counter**, which is also known as an **up, or forward** counter .
- Now, if the Q_0 output is connected to T input of FF1 (positive edge triggered), or Q_0' output connected to T input of FF1 (negative edge triggered) and so on, the counter is a down counter.
- Here, we assume the counter is SET (Q = 1) before counting.

Asynchronous Counters
- Asynchronous Modulo-N or Divide-by-N counters

- Modulus of a counter is the number of unique states that the counter will sequence through.

- The value 2^N is actually the maximum MOD number that can be obtained using N flip–flops.

- The basic counter can be changed to produce MOD numbers less than 2^N by skipping states of the counting sequence.

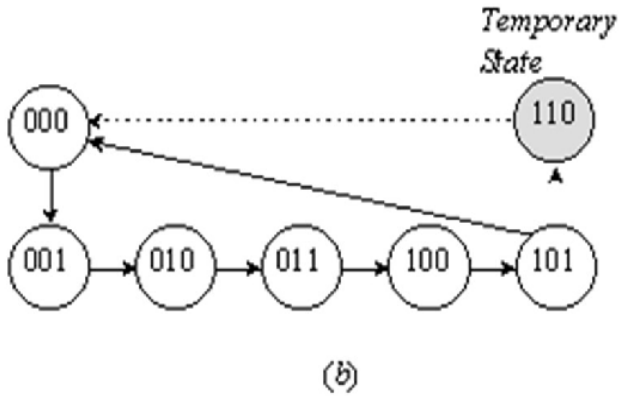
- Commonly asynchronous inputs *Reset* (*R*) or *Clear* (*CLR*) of the flip–flops can be used to force counter to recycle before going thorough all of the states.

Asynchronous Counters
- Asynchronous Mod-6 or Divide-by-6 UP counters (with negative edge triggered flip-flops)

- The MOD–6 counter cycle through 6 states.

C	B	A
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0

(a)



(b)

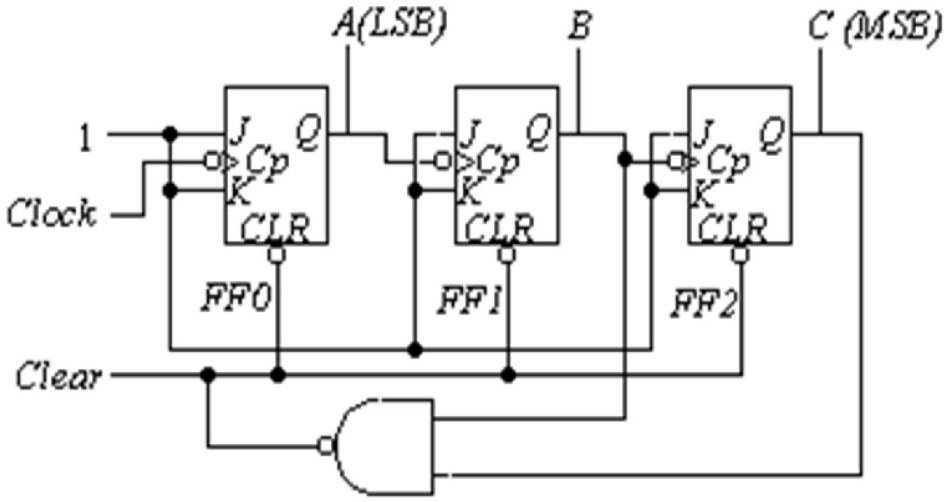
(a) Counting sequence; (b)State transition diagram of MOD–6

MOD-6
Decimal: 0,1,2,3,4,5,**6**,0, …
Binary:
000,001,010,011,100,101,**11**0,
000,…
Temporary state: 6_{10} or 110_2
Reset condition: $Q_2Q_1 = 11$

- The MOD–6 has a counting sequence as shown in above figure.
- The temporary state will not appear in the counting sequence because it happens only for a few nanoseconds.

Asynchronous Counters

- Asynchronous Mod-6 or Divide-by-6 UP counters (with negative edge triggered flip-flops)

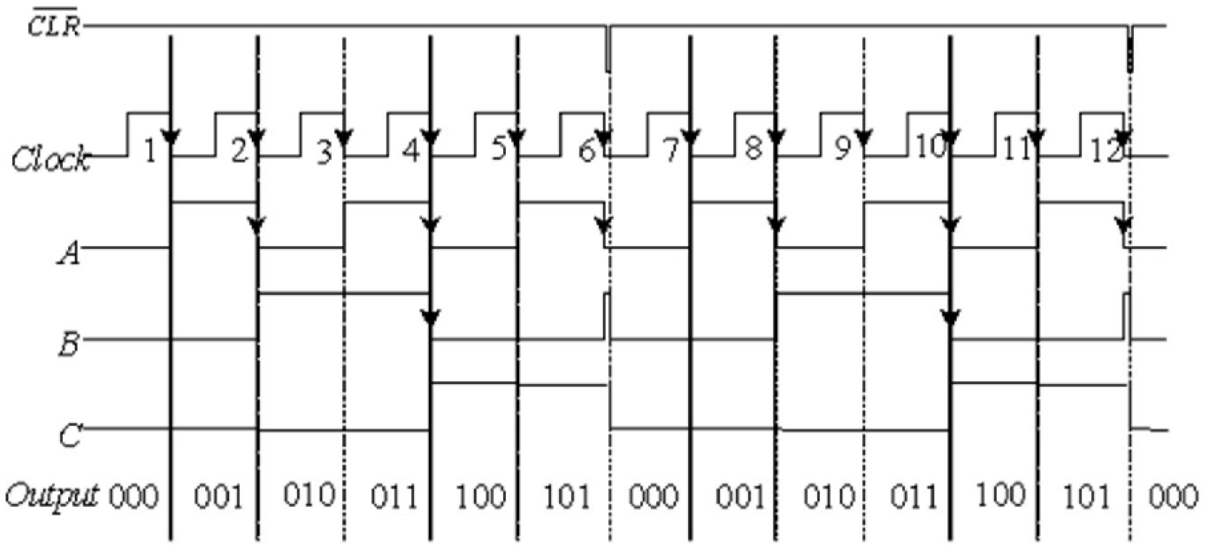


MOD–6 asynchronous up–counter logic diagram

- It is constructed using three negative edge triggered JK flip–flops with active LOW *Clear* inputs and 1 NAND gate.
- The *Clear* input is used to reset all the outputs of the flip–flops to 0.
- The output of NAND gate is used to force counter to recycle sequence back to 000 (*CBA*) state if both the inputs are 1
- This can be done by connecting all the *CLR* of the flip–flops to NAND gate output.

Asynchronous Counters

- Asynchronous Mod-6 or Divide-by-6 UP counters (with negative edge triggered flip-flops)



Timing diagram for MOD–6 asynchronous up–counters

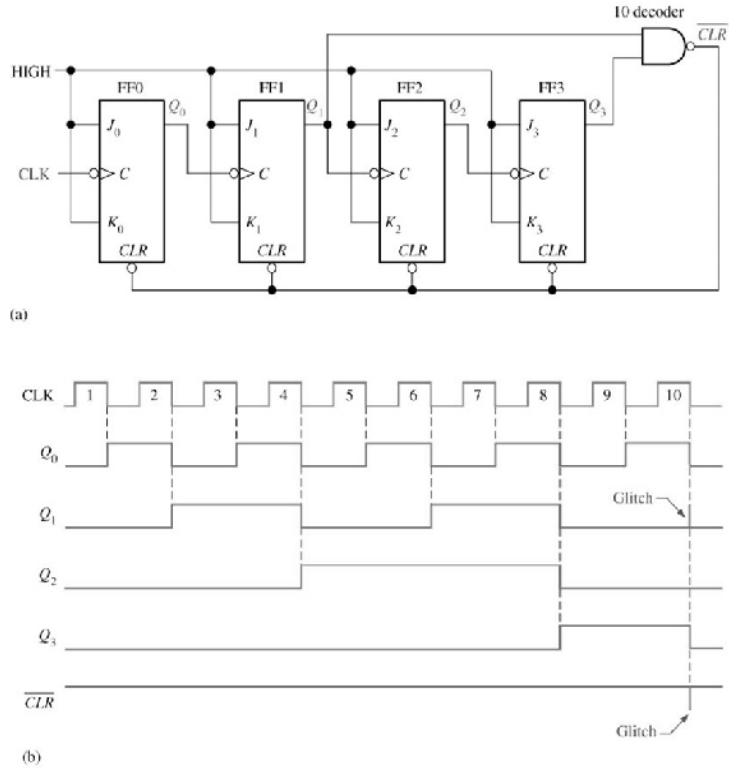
Asynchronous Counters

- Asynchronous Mod-10 or Divide-by-10 or BCD or Decade UP counters (with negative edge triggered flip-flops)

- Counters with ten states in their sequence are called decade counters.
- A decade counter with a counting sequence of zero (0000) through nine (1001) is a *BCD counter* because its ten-state sequence produces the BCD code.
- This type of counter is useful in display applications in which BCD is required for conversion to a decimal readout (7-segment).
- To obtain a truncated sequence, it is necessary to force the counter to recycle before going through all of its possible states.
- The CLR asynchronous input will be used for this purpose.

Asynchronous Counters

- Asynchronous Mod-10 or Divide-by-10 or BCD or Decade UP counters (with negative edge triggered flip-flops)



(a) Logic Diagram (b) Timing Diagram

- One way to make the counter recycle after the count of nine (1001) is to decode count ten (1010) with a NAND gate and connect the output of the NAND gate to the clear (CLR') inputs of the flip-flops.
- Since there is a unique combination of Q_1Q_3 (reset condition), a partial decoding is sufficient to decode the count of ten because none of the other states (0 to 9) have both Q_1 and Q_3 HIGH at the same time.

Decimal: 0, 1, 2, 3, …, 9, 10, 0, 1, …
Binary: 0000, 0001, 0010, 0011, …, 1001, **1**0**1**0, 0000, 0001, …
Here the 10_{10} or 1010_2 is a temporary state.

Reset condition: $Q_3Q_1 = 11$

Asynchronous Counters
- Limitation

- The asynchronous (ripple) counter has a problem with propagation delay that propagate from the first flip–flop to second flip–flop, and so on.
- All the flip–flops cannot change states simultaneously in synchronisation with the input pulse.
- The maximum cumulative delay in an asynchronous counter must be less than the period of the clock waveform or period.
- The limitation of the time lag in triggering all the flip–flops can be overcome with the use of synchronous (parallel) counters.
  - Here all the Flip-flops are triggered simultaneously by a clock input pulse

Sequential Logic Circuits - Synchronous Counters

* Design Procedure for Synchronous Counters
* Flip-flop Excitation Tables
* Design of Synchronous (Parallel) Counters - Examples
    * 2-bit down counter (using J-K flip-flops)
    * 2-bit UP/down counter (using J-K flip-flops)
    * Mod 6 UP counter (using J-K flip-flops)
    * Counter with arbitrary sequence (using J-K flip-flops)
    * Decade counter (using T flip-flops)
    * Counter with arbitrary sequence (using T flip-flops)

Synchronous Counters
- Design Procedure

- In synchronous counters all the flip–flops are clocked at the same time.
- The **design procedure** is as follows:

Step-1: Understand the required circuit specification and draw the state transition diagram

Step-2: Draw the Flip-flop excitation table

Step-3: Draw the state table (or circuit excitation table) showing present state, next state and Flip-flop inputs. Present State and next state are determined from state transition diagram from step 1.
Flip-flop inputs are determined based on Flip-flop excitation table in Step2.

Step-4: Use K-maps to simplify the boolean expressions for Flip- flop inputs

Step-5: Draw the implementation diagram for the counter using the simplified boolean expressions obtained in Step 4 and using the required number of flip–flops.

Synchronous Counters
- Flip-flop Excitation Tables

- Excitation tables give the required transition from present state (Q) to next state ($Q+$) and determine the flip–flop input(s).
- Excitation Tables for the different flip-flops are shown below:

JK Excitation table

Q	Q^+	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

D Excitation table

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

T Excitation table

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

Synchronous Counters
- Analysis of JK Flip-flop Excitation Table

0 → 0 Transition
Flip–flop present state (Q) is 0 and it should remain 0 in the next state (Q+) when a clock pulse applied.
For this to happen, $J = K = 0$ or $J = 0$ and $K = 1$.
This means $J = 0$ and $K = 0$ **or 1** so $J = 0$ and $K = X$ (i.e don't care).

0 → 1 Transition
Q is 0 and Q+ is 1.
Either **J = 1** and **K = 0** or **J=K = 1.**
So **J = 1** and **K = X**.

1 → 0 Transition
Q is 1 and Q+ is 0.
Either **J = 0 and K = 1** or **J = K = 1**.
So **J = X** and **K = 1**.

1 → 1 Transition
Q is 1 and Q+ is 1.
Either **J = K = 0** or **J = 1 and K = 0**.
So **J = X** and **K=0**.

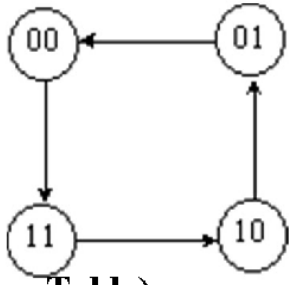
Synchronous Counters
- Design Example 1 (2-bit DOWN counter)

- **Problem 1:**
  Design a synchronous (parallel) counter using J-K Flip-flops so that the counting sequence is : 11-10-01-00 and then recycles to 11.

- **Step 1: Circuit specification:** Sequence 11-10-01-00 and then recycles to 11.

B	A
1	1
1	0
0	1
0	0

State Transition Diagram



- **Step 2: (Identification of Flip-flop excitation Table)**
  - Identify the number of flip-flops needed:
    Since it has two outputs A and B, we can construct the circuit with two flip–flops.
  - Identify the type of flip-flop to be used:
    JK flip–flops are used to construct this circuit.
  - Then, **construct the excitation table for JK flip-flop**

JK Excitation table

Q	Q^+	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Synchronous Counters
- Design Example 1 (2-bit DOWN counter)

- **Step 3: (Construction of State Table)**
  Use the state transition diagram and JK flip–flop excitation table to set up a table that lists all Present States and Next States.

  State Table given below indicates the level required by each input J and K in order to produce the transition from present state to the Next State

JK Excitation table

Q	Q^+	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

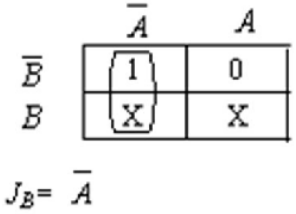
Present State		*Next State*		*Inputs J, K*			
B	A	B^+	A^+	J_B	K_B	J_A	K_A
1	1	1	0	X	0	X	1
1	0	0	1	X	1	1	X
0	1	0	0	0	X	X	1
0	0	1	1	1	X	1	X

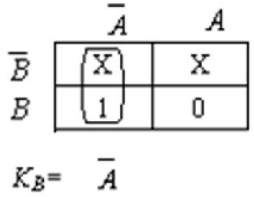
Synchronous Counters
- Design Example 1 (2-bit DOWN counter

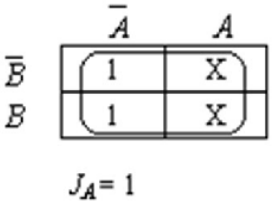
Present State		Next State		Inputs J, K			
B	A	B+	A+	J_B	K_B	J_A	K_A
1	1	1	0	X	0	X	1
1	0	0	1	X	1	1	X
0	1	0	0	0	X	X	1
0	0	1	1	1	X	1	X

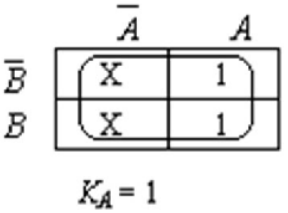
- **Step 4 (Simplification using Karnaugh maps)**

Based on the table above, the minimum logic required for the J and K inputs of each flip-flop in the counter can be generated using K–maps. There is a K-map for the J input and a K-map for the K-input of each flip-flop. Each cell in K-map represents one of the present states in the counter sequence.



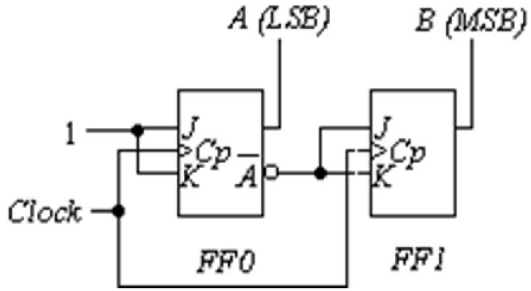






- **Step 5: (Logic Diagram)**

Complete the design by drawing the logic diagram as the following figure. using $J_A = K_A = 1$ and $J_B = K_B = \overline{A}$



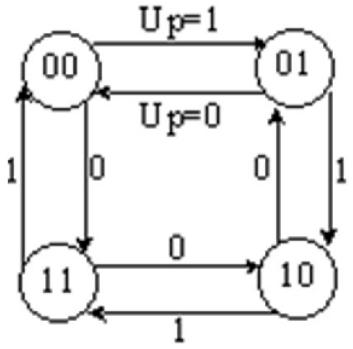
Synchronous Counters
- Design Example 2 (2-bit UP/DOWN counter)

Problem 2: Design a parallel counter using J-K Flip-flops, that has the following sequence:

- **If the input Up = 1, it will count up 00-01-10-11 then recycle to 00.**
- **If the input Up = 0, it will count down 11-10-01-00 then recycle to 11.**

Step–1: Specification and State transition diagram:

State Transition Diagram



Inputs	*Present State*		*Next State*	
Up	*B*	*A*	B^+	A^+
0	0	0	1	1
0	0	1	0	0
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

JK Excitation table

Q	Q^+	*J*	*K*
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Step–2

Based on the above steps, numbers of flip–flops required are two.
J-K flip-flops have to be used. Construct JK flip-flop excitation table

Synchronous Counters

- Design Example 2 (2-bit UP/DOWN counter)

JK Excitation table

Q	Q^+	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

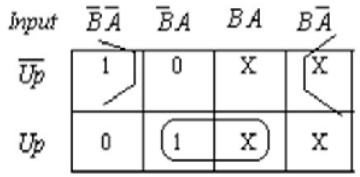
Step–3: Construction of State Table:

Input	*Present State*		*Next State*		*Input J,K*			
Up	B	A	B^+	A^+	J_B	K_B	J_A	K_A
0	0	0	1	1	1	X	1	X
0	0	1	0	0	0	X	X	1
0	1	0	0	1	X	1	1	X
0	1	1	1	0	X	0	X	1
1	0	0	0	1	0	X	1	X
1	0	1	1	0	1	X	X	1
1	1	0	1	1	X	0	1	X
1	1	1	0	0	X	1	X	1

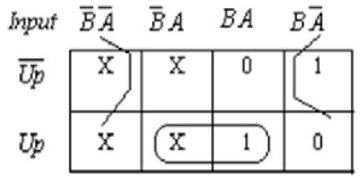
Step–4: K-map simplification :

$J_A = 1$

$K_A = 1$



$J_B = \overline{Up \oplus A}$



$K_B = \overline{Up \oplus A}$

Synchronous Counters

- Design Example 2
(2-bit UP/DOWN counter)

Input	$\overline{B}\overline{A}$	$\overline{B}A$	BA	$B\overline{A}$
$\overline{Up}$	1	X	X	1
Up	1	X	X	1

$J_A = 1$

Input	$\overline{B}\overline{A}$	$\overline{B}A$	BA	$B\overline{A}$
$\overline{Up}$	X	1	1	X
Up	X	1	1	X

$K_A = 1$

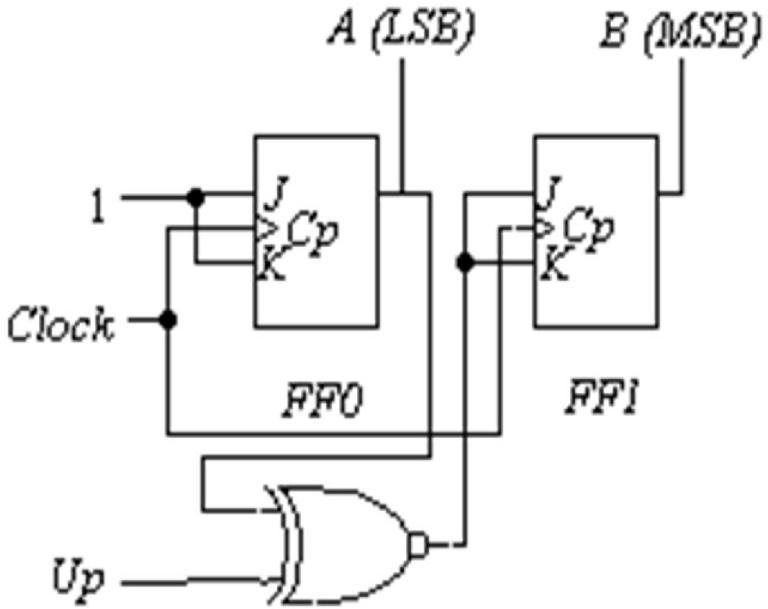
Input	$\overline{B}\overline{A}$	$\overline{B}A$	BA	$B\overline{A}$
$\overline{Up}$	1	0	X	X
Up	0	1	X	X

$J_B = \overline{Up \oplus A}$

Input	$\overline{B}\overline{A}$	$\overline{B}A$	BA	$B\overline{A}$
$\overline{Up}$	X	X	0	1
Up	X	X	1	0

$K_B = \overline{Up \oplus A}$

Step–5: Construction of Logic Diagram



Synchronous Counters
- Design Example 3 (Mod 6 UP counter)

Problem 3:

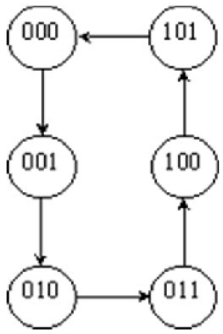
Design MOD–6 parallel up–counter using J-K flip-flops that has the following sequence: 000-001-010-011-100 101and then recycle to 000. The undesired states 110 and 111 go to don't care on the next clock pulse.

Solution:

Step 1: Specification and State Transition Diagram:

C	*B*	*A*
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1

State Transition Diagram:



Step–2

Based on the above steps, numbers of flip–flop required are three.
J-K flip flop is required. Construct the excitation table for JK flip-flop

Synchronous Counters
- Design Example 3
(Mod 6 UP counter)

JK Excitation table

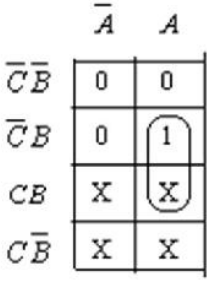
Q	Q^+	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Step–3

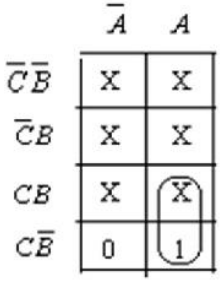
Construction of State Table:

Present State			*Next State*			*Input FF2*		*Input FF1*		*Input FF0*	
C	B	A	C^+	B^+	A^+	J_C	K_C	J_B	K_B	J_A	K_A
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	0	1	1	0	X	X	0	1	X
0	1	1	1	0	0	1	X	X	1	X	1
1	0	0	1	0	1	X	0	0	X	1	X
1	0	1	0	0	0	X	1	0	X	X	1
1	1	0	X	X	X	X	X	X	X	X	X
1	1	1	X	X	X	X	X	X	X	X	X

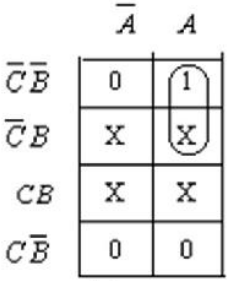
Step–4 K-Map Simplification:



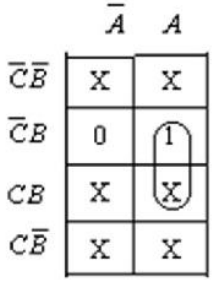
$J_C = AB$



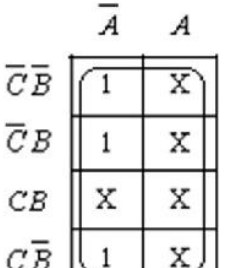
$K_C = A$



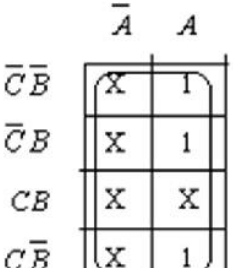
$J_B = A\overline{C}$



$K_B = A$

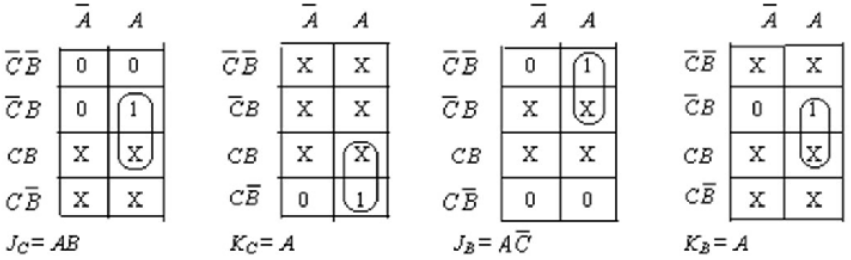


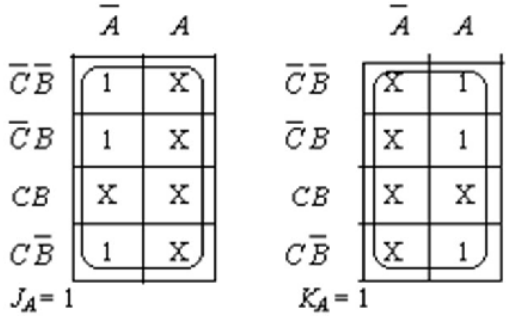
$J_A = 1$



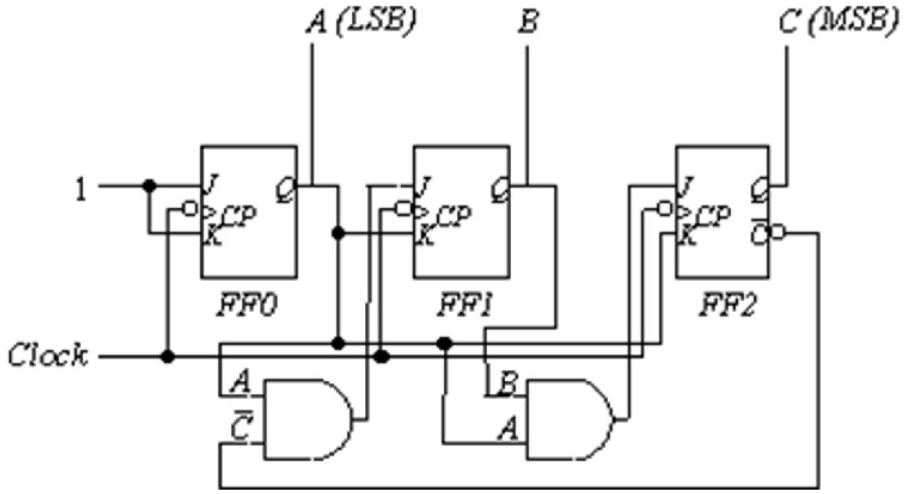
$K_A = 1$

Synchronous Counters
- Design Example 3
(Mod 6 UP counter)





Step–5 Implementation Diagram



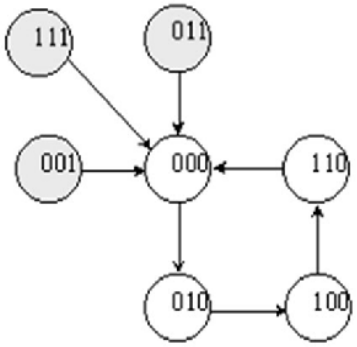
Synchronous Counters
- Design Example 4

Problem 4: Design a parallel up–counter that has the following sequence: 000-010-100-110 and then recycle to 000. The undesired states 001, 011,101 and 111 go to 000 on the next clock pulse.

Solution:

Step–1: Specification and State Transition Diagram

C	B	A
0	0	0
0	1	0
1	0	0
1	1	0



Synchronous Counters
- Design Example 4

Step–2

Based on the above steps, numbers of flip–flop required are three.
J-K flip-flop is used. Construct JK Flip-flop excitation table

JK Excitation table

Q	Q^+	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Step–3 Construct State table showing present state, next state and Flip- flop inputs

Present State			*Next State*			Input *FF2*		Input *FF1*		Input *FF0*	
C	B	A	C^+	B^+	A^+	J_C	K_C	J_B	K_B	J_A	K_A
0	0	0	0	1	0	0	X	1	X	0	X
0	0	1	0	0	0	0	X	0	X	X	1
0	1	0	1	0	0	1	X	X	1	0	X
0	1	1	0	0	0	0	X	X	1	X	1
1	0	0	1	1	0	X	0	1	X	0	X
1	0	1	0	0	0	X	1	0	X	X	1
1	1	0	0	0	0	X	1	X	1	0	X
1	1	1	0	0	0	X	1	X	1	X	1

Synchronous Counters
- Design Example 4

Step–4 K-Map Simplification for the Flip-flop inputs

$\bar{C}\bar{B}$	$\bar{A}$	A
$\bar{C}\bar{B}$	0	0
$\bar{C}B$	1	0
CB	X	X
$C\bar{B}$	X	X

$J_C = \bar{A}B$

$\bar{C}\bar{B}$	$\bar{A}$	A
$\bar{C}\bar{B}$	X	X
$\bar{C}B$	X	X
CB	1	1
$C\bar{B}$	0	1

$K_C = A+B$

$\bar{C}\bar{B}$	$\bar{A}$	A
$\bar{C}\bar{B}$	1	0
$\bar{C}B$	X	X
CB	X	X
$C\bar{B}$	1	0

$J_B = \bar{A}$

$\bar{C}\bar{B}$	$\bar{A}$	A
$\bar{C}\bar{B}$	X	X
$\bar{C}B$	1	1
CB	1	1
$C\bar{B}$	X	X

$K_B = 1$

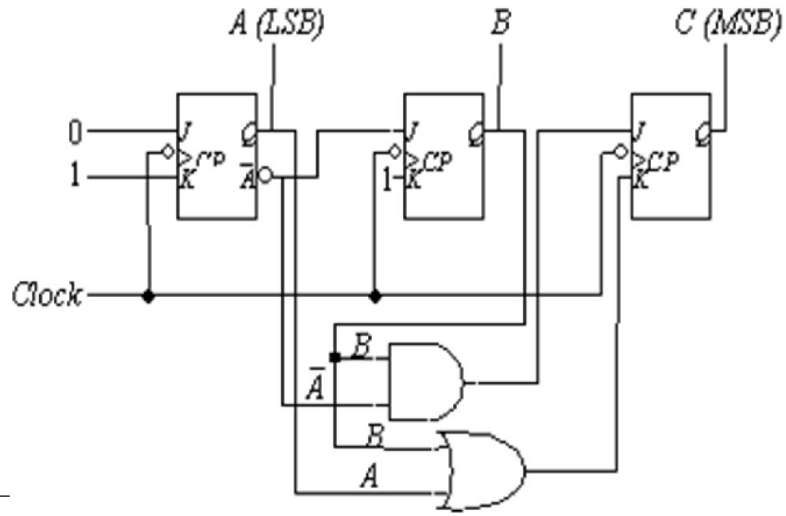
$\bar{C}\bar{B}$	$\bar{A}$	A
$\bar{C}\bar{B}$	0	X
$\bar{C}B$	0	X
CB	0	X
$C\bar{B}$	0	X

$J_A = 0$

$\bar{C}\bar{B}$	$\bar{A}$	A
$\bar{C}\bar{B}$	X	1
$\bar{C}B$	X	1
CB	X	1
$C\bar{B}$	X	1

$K_A = 1$

Step–5: Complete the design by drawing the logic diagram as shown in the following figure.



Synchronous Counters
- Design Example 5 (Mod-10 UP Counter)

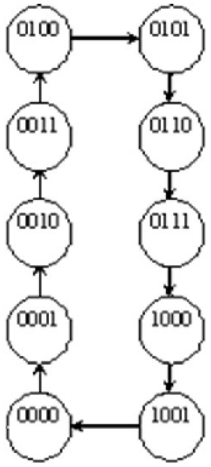
Problem 4: Design a MOD–10 parallel up–counters using T-flip-flops that has the following sequence: 0000-000 0010-0011-0100-0101-0110- 0111-1000-1001 and then recycle to 0000. The undesired states 1010, 1011,110 1101, 1110 and 1111 go to don't care on the next clock pulse.

Solution: Step 1: Specification and State Transition Diagram

.

D	*C*	*B*	*A*
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

STATE TRANSITION DIAGRAM:



Synchronous Counters
- Design Example 5 (Mod-10 UP Counter)

T Excitation table

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

Step–2

Based on the above steps, numbers of flip–flop required are four.

T flip–flops are used. Construct T Flip-flop excitation table.

Step–3 Construct State Table

Present State				*Next State*				*FF3*	*FF2*	*FF1*	*FF0*
D	C	B	A	D^+	C^+	B^+	A^+	T_D	T_C	T_B	T_A
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	1	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	1
0	0	1	1	0	1	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	1
0	1	0	1	0	1	1	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	1
0	1	1	1	1	0	0	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	1
1	0	0	1	0	0	0	0	1	0	0	1
1	0	1	0	X	X	X	X	X	X	X	X
1	0	1	1	X	X	X	X	X	X	X	X
1	1	0	0	X	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X	X

Synchronous Counters
- Design Example 5 (Mod-10 UP Counter)

Step–4 K-Map Simplification for T Flip-flop inputs:

	$\overline{B}\overline{A}$	$\overline{B}A$	BA	$B\overline{A}$
$\overline{D}\overline{C}$	0	0	0	0
$\overline{D}C$	0	0	1	0
DC	X	X	X	X
$D\overline{C}$	0	1	X	X

$T_D = ABC + AD$

	$\overline{B}\overline{A}$	$\overline{B}A$	BA	$B\overline{A}$
$\overline{D}\overline{C}$	0	0	1	0
$\overline{D}C$	0	0	1	0
DC	X	X	X	X
$D\overline{C}$	0	0	X	X

$T_C = AB$

	$\overline{B}\overline{A}$	$\overline{B}A$	BA	$B\overline{A}$
$\overline{D}\overline{C}$	0	1	1	0
$\overline{D}C$	0	1	1	0
DC	X	X	X	X
$D\overline{C}$	0	0	X	X

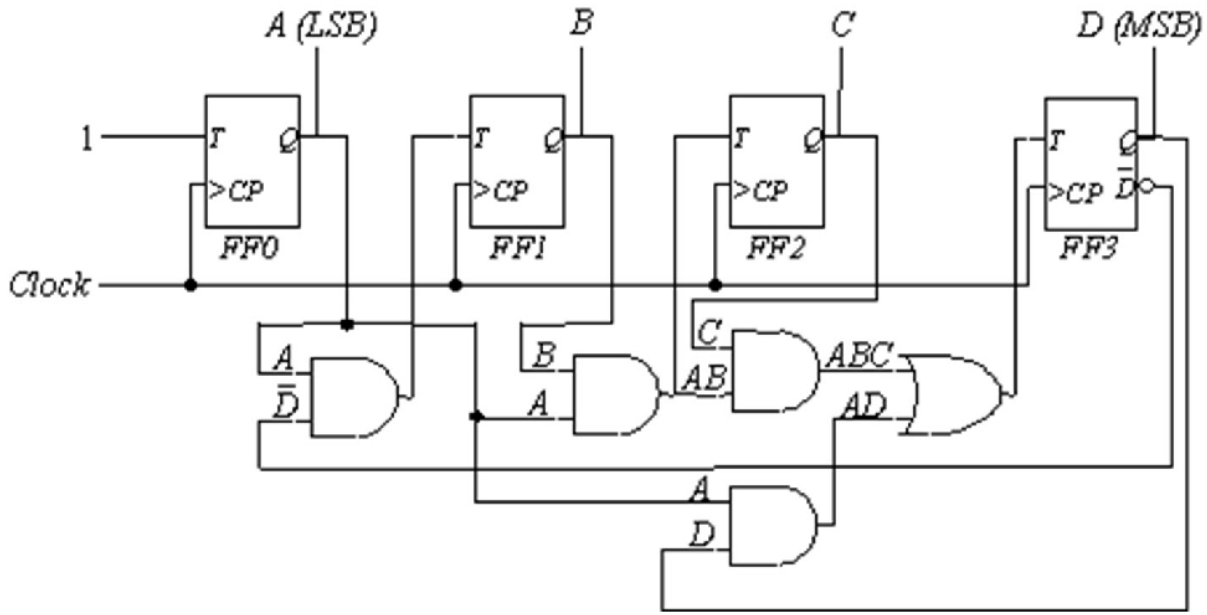
$T_B = A\overline{D}$

	$\overline{B}\overline{A}$	$\overline{B}A$	BA	$B\overline{A}$
$\overline{D}\overline{C}$	1	1	1	1
$\overline{D}C$	1	1	1	1
DC	1	1	1	1
$D\overline{C}$	1	1	1	1

$T_A = 1$

Synchronous Counters
- Design Example 5 (Mod-10 UP Counter)

Step–5: Complete the design by drawing the logic diagram



Synchronous Counters
- Design Example 6

Problem 6:

Design a synchronous (parallel) up-counter (using T flip-flops) that sequences as follows:
000 – 010 – 100 -110 and then recycle to 000.

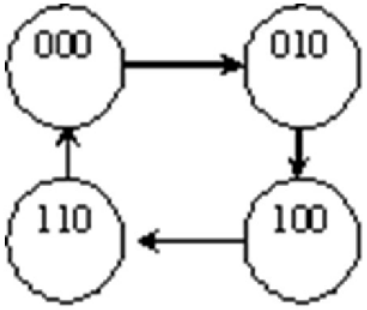
Solution

Step 1:

Since the LSB does not change, i.e. it is always 0, we need not design any circuitry for it.
We can set it to 0 straightaway (i.e. A=0) and concentrate on other outputs (i.e. C and B).

C	*B*	*A*
0	0	0
0	1	0
1	0	0
1	1	0

State Transition Diagram:



Synchronous Counters
- Design Example 6

Step–2

Based on the above steps, number of flip–flops required are two ONLY.

Since T-flip-flops are used for construction, draw T Flip-flop excitation table.

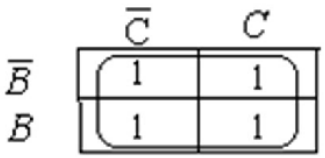
T Excitation table

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

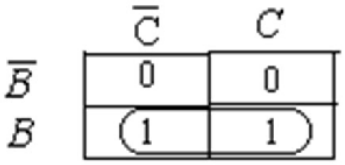
Step–3 State Table:

Present state		Next state		FF_C	FF_B
C	B	C^+	B^+	T_C	T_B
0	0	0	1	0	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	0	0	1	1

Step–4 K-Map Simplification



$T_B=1$

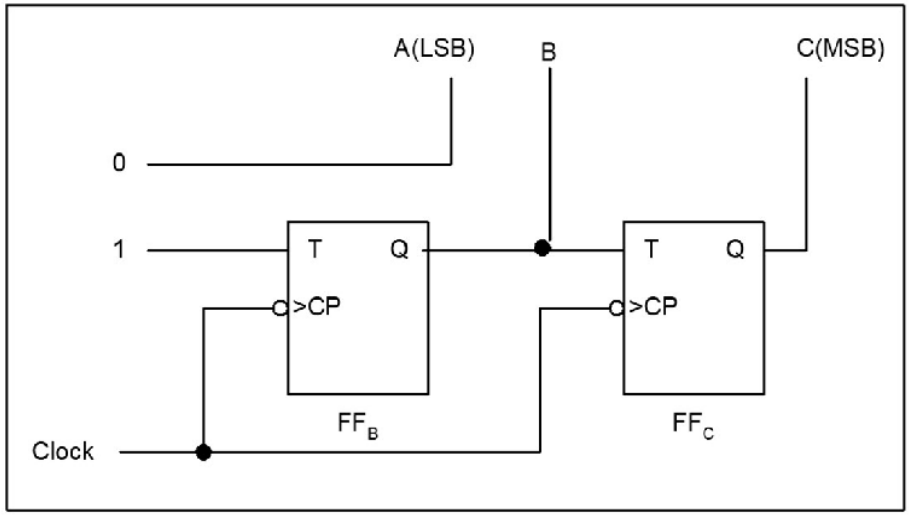


$T_C=B$

Synchronous Counters
- Design Example 6

Step–5 **Logic Diagram:**

Note that A=0.



Sequential Logic Circuits - Shift Registers

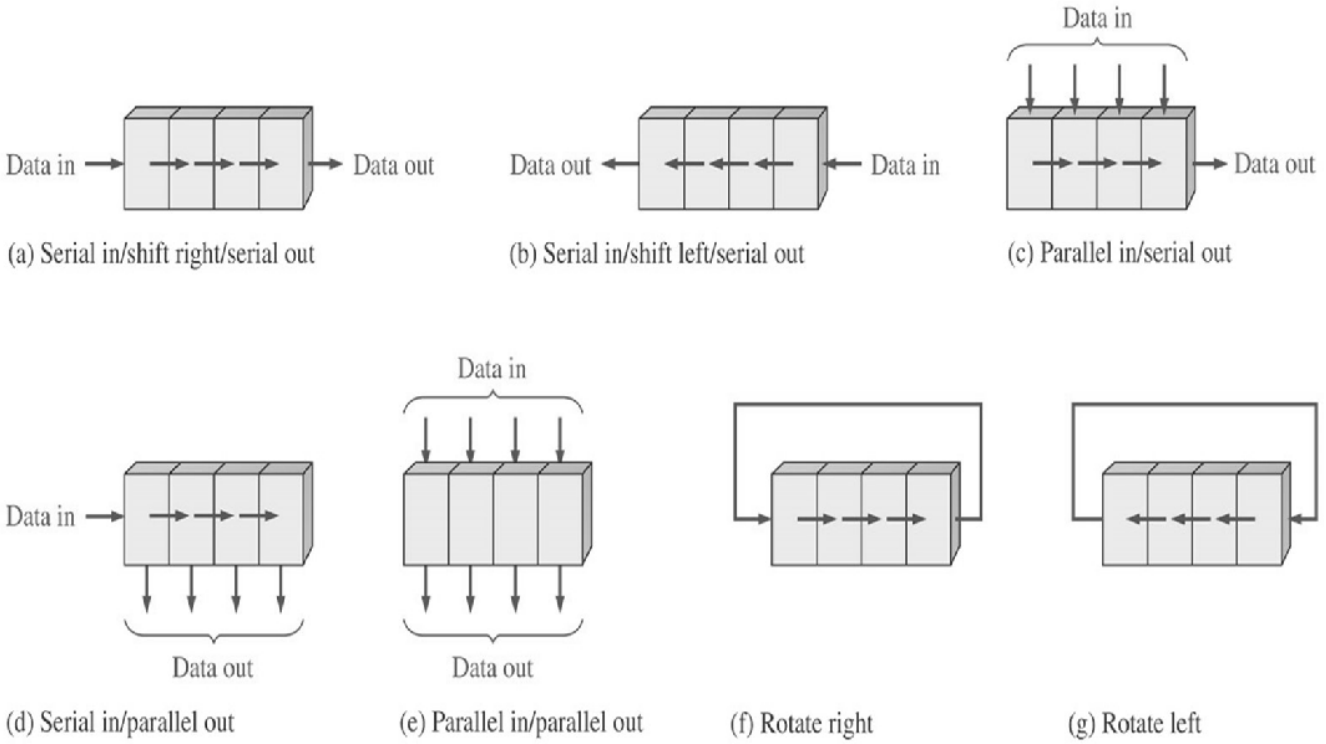
* Basic Shift Register Functions
* Construction and working of different Shift Registers
    * Serial In Serial Out (SISO)
    * Serial In Parallel Out (SIPO)
    * Parallel In Serial Out (PISO)
    * Parallel In Parallel Out (PIPO)

Shift Registers
- Basic Functions

- Constructed with arrangement of flip-flops
- Two Basic Functions of Shift Registers
  - Data Storage
  - Data Movement
- Data storage
  - This capability makes it an important type of memory device
  - For example, in a D flip-flop, if 1 is applied to data input and a clock pulse is applied, then it stores 1 by setting the flip-flop. When the 1 on the input is removed, the flip-flop remains in the SET state, thereby storing 1.
  - Similarly, 0 also can be stored by resetting the flip-flop
- Storage capacity of a register (total number of bits it can retain) is determined by the number of stages (flip-flops).
  - Each stage can store one bit .
- No specified sequence of states – unlike counter
- Shift capability permits the movement of data from stage to stage within register or into or out of register upon application of clock pulses.

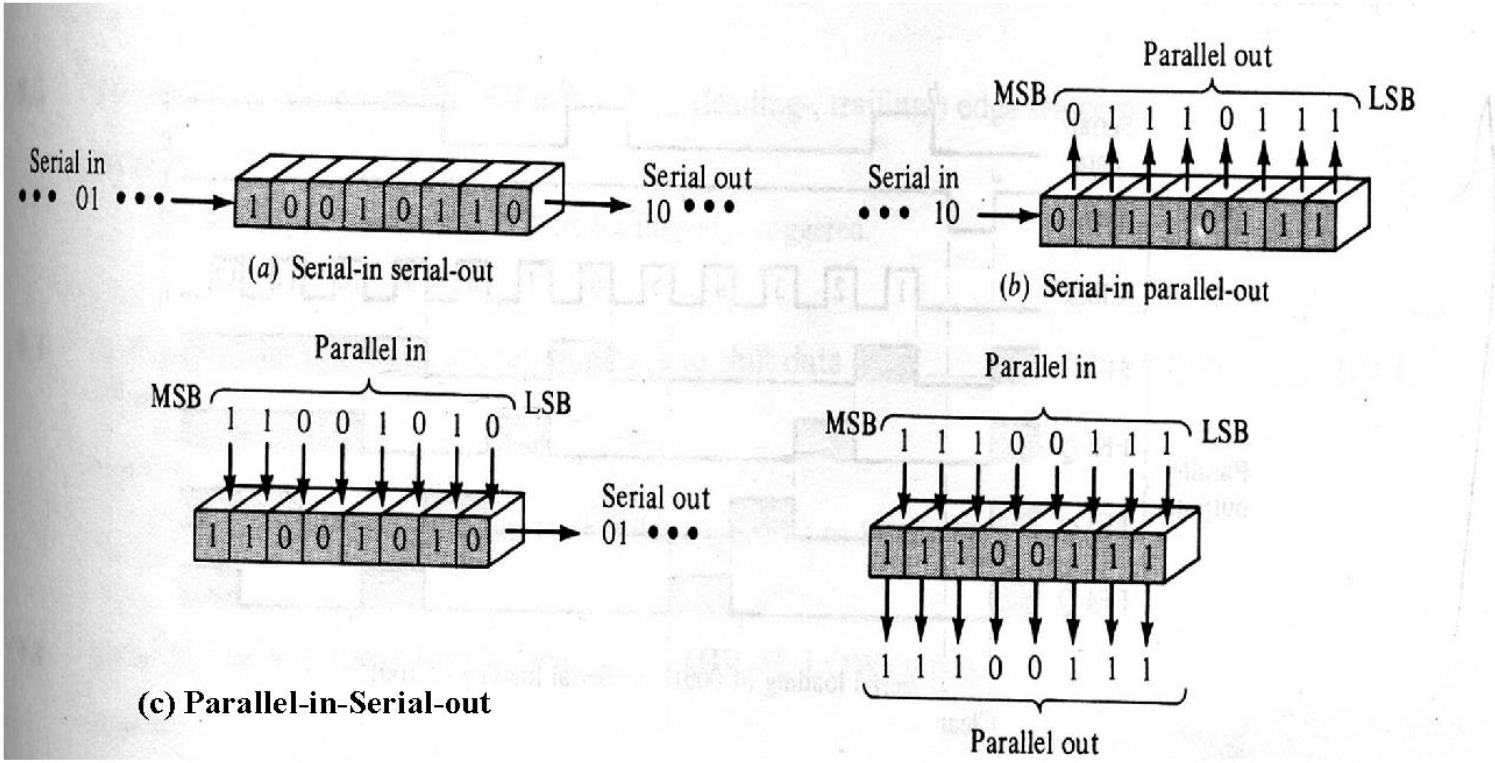
Shift Registers
- Basic Data Movements

For illustration, four bits are used. Bits move in the direction of the arrows.



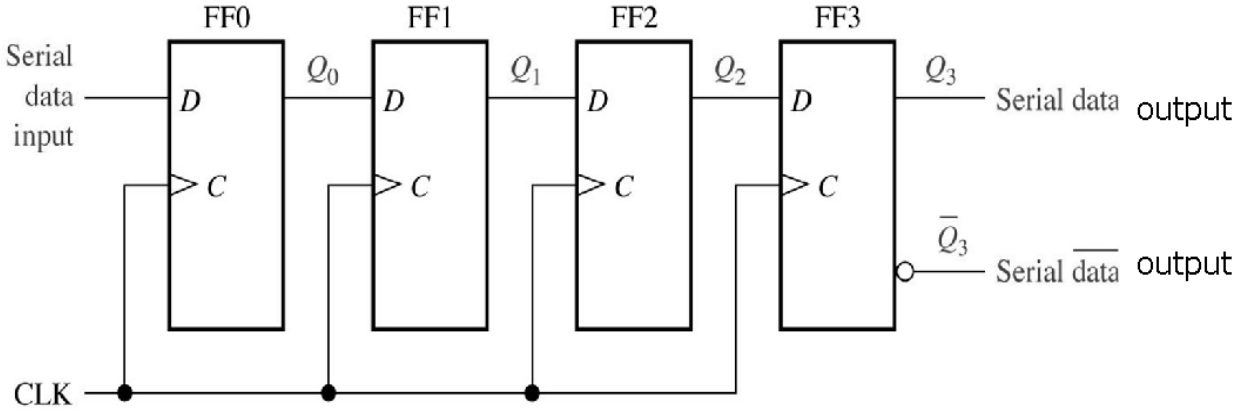
Shift Registers

-Classification (Illustration with 8-bit data)

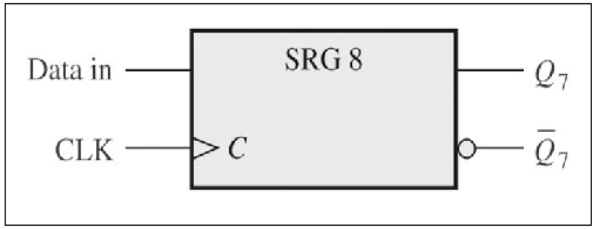


Shift Registers
- Serial-in-Serial-out (Implementation)

Logic Diagram for 4-bit serial-in-serial-out shift register (implemented with D flip-flops)



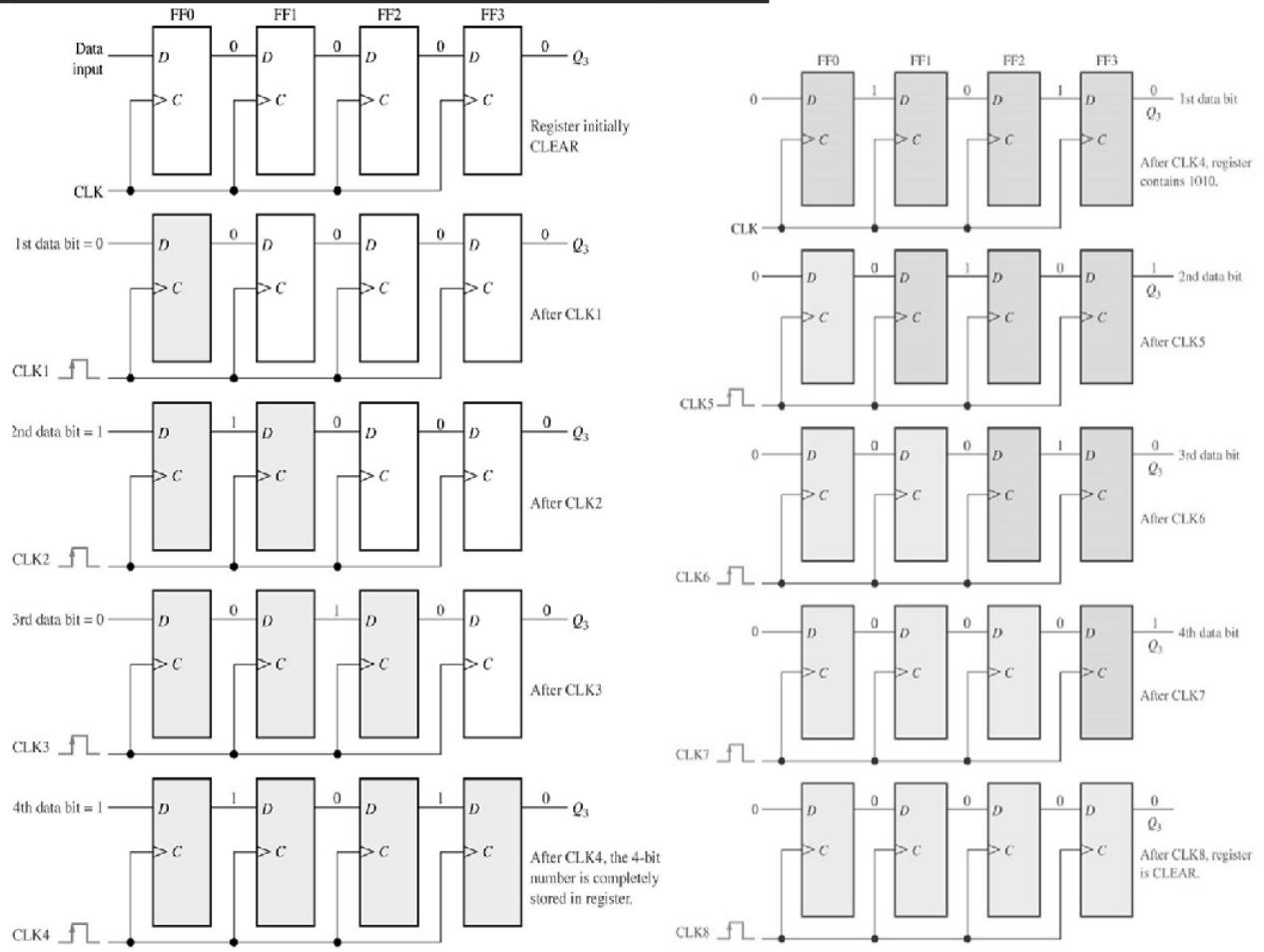
Logic symbol for 8-bit serial-in-serial-out shift register



Shift Registers
- 4-bit Serial-in-Serial-out (Illustration)

➢Four bits **(1010)** being entered serially into the register is shown.

The data input is on the left. Each clock pulse will move the input bit to the next flip-flop. (Shifting occurs from left to right)

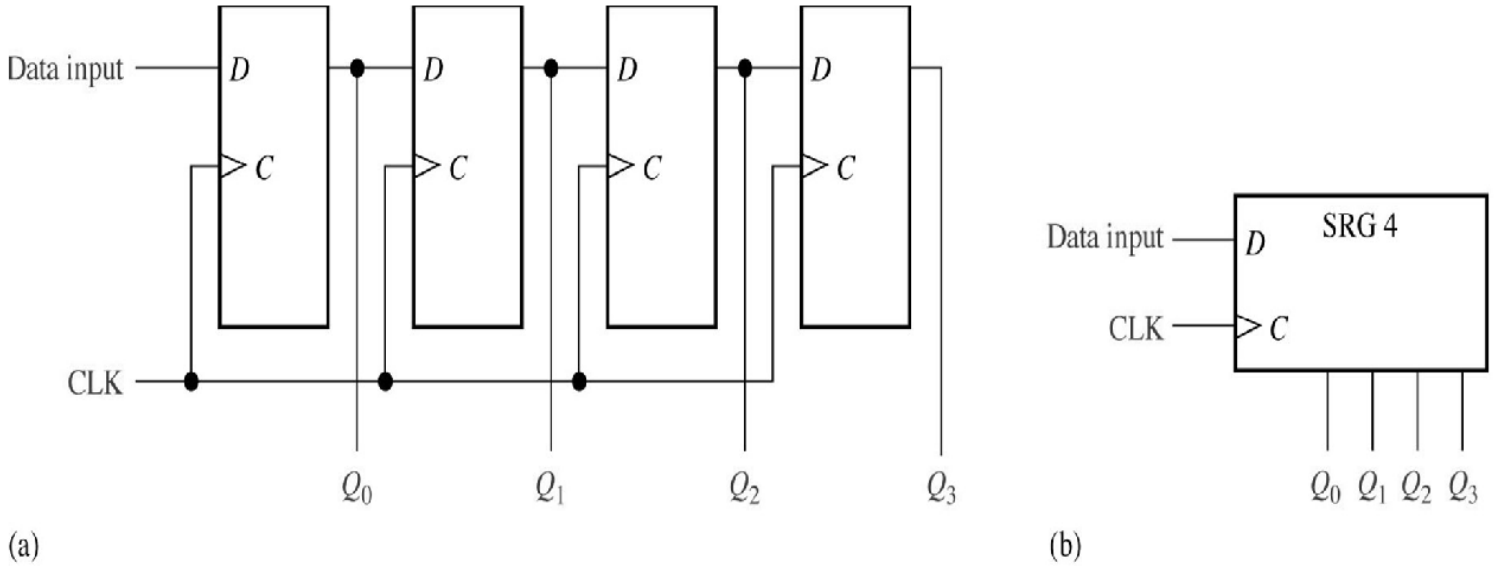


Four bits **(1010)** being serially shifted out of the register to display device and replaced by all zeros

Shift Registers

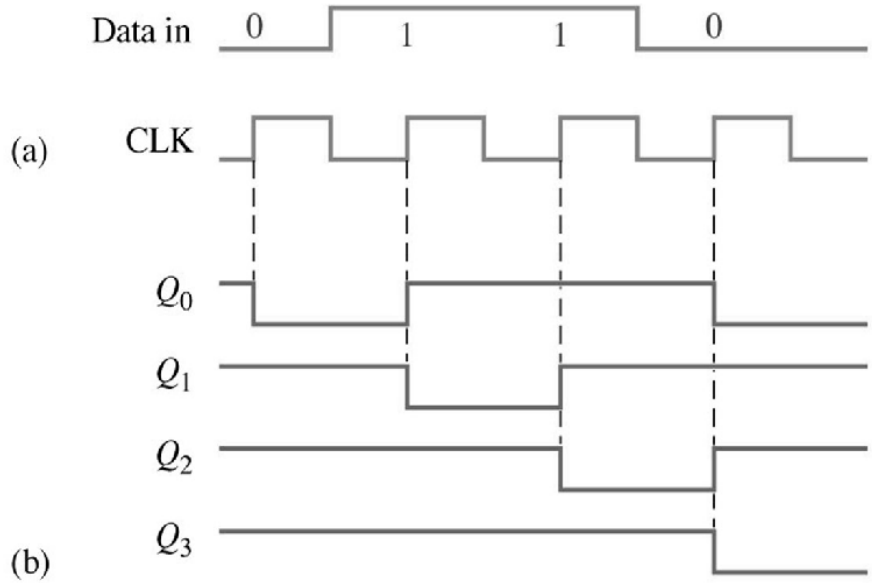
- 4-bit Serial-in-Parallel-out (Implementation)

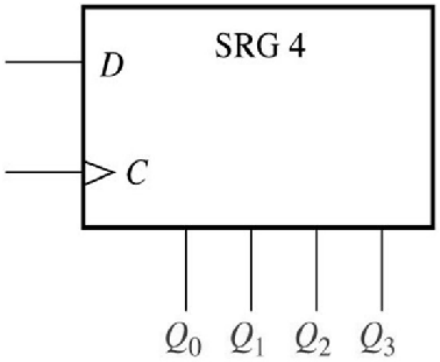
Figure shows a 4-bit serial in/parallel out shift register and its logic block symbol.



Shift Registers
- 4-bit Serial-in-Parallel-out (Example)

Show the state of the 4-bit SIPO register for the data input and clock waveforms given. The register initially contains all 1's.

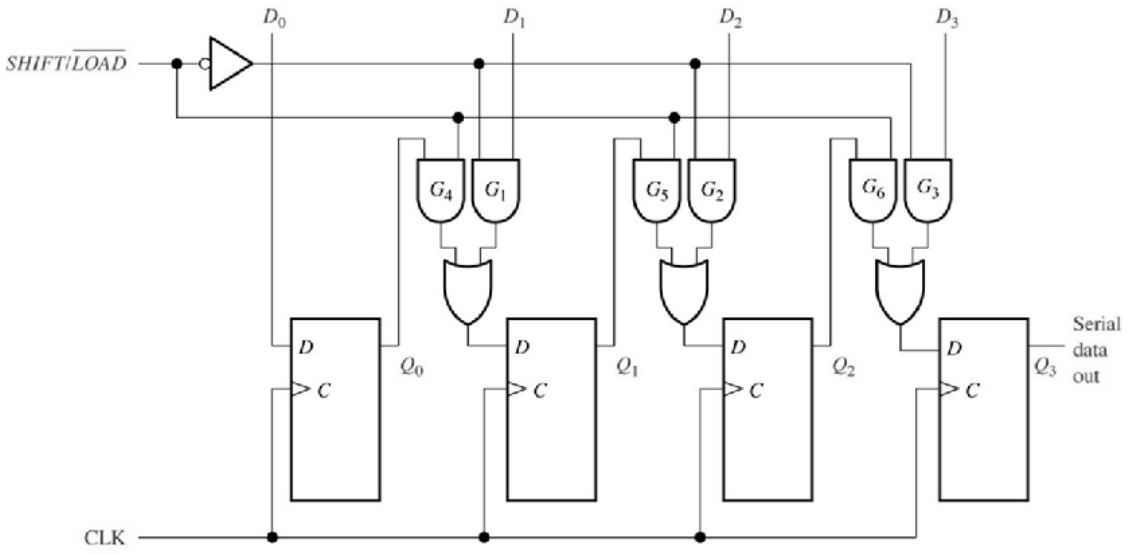




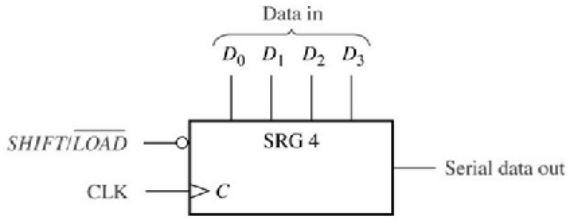
The register contains 0110 after 4 clock pulses.

Shift Registers

- 4-bit Parallel-in-Serial-out (Implementation)



(a) Logic diagram



(b) Logic symbol

Shift Registers

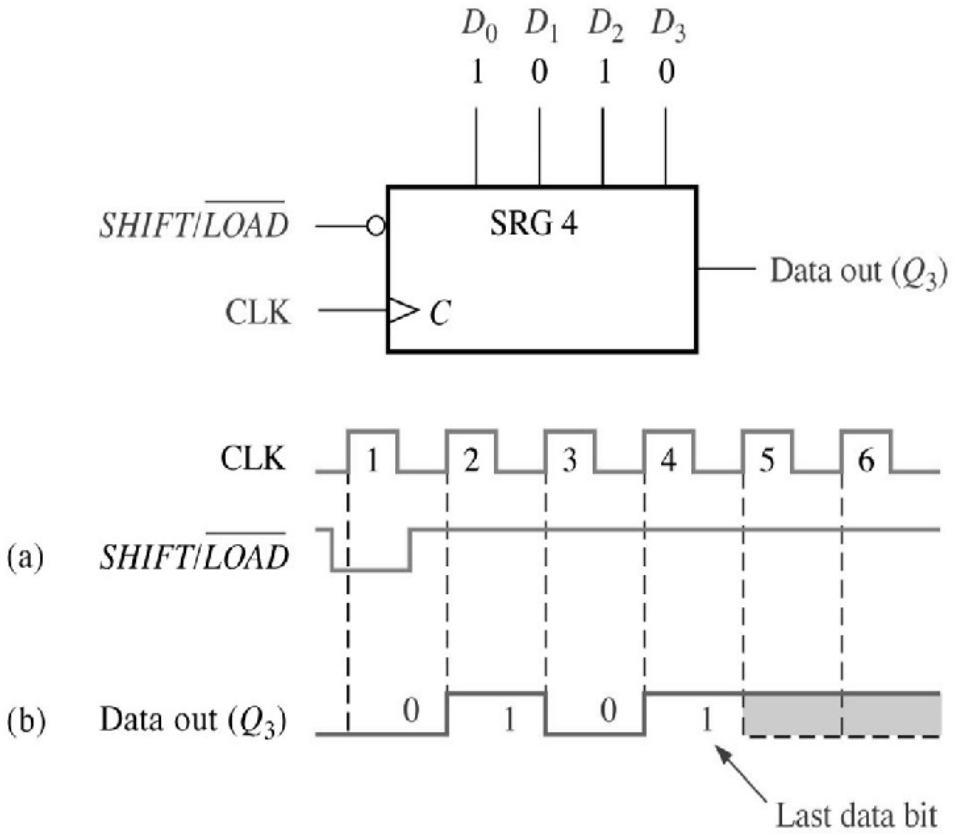
- 4-bit Parallel-in-Serial-out (Explanation)

- There are four data-input lines, D_0, D_1, D_2, D_3 and a SHIFT/$\overline{\text{LOAD}}$ input, which allows four bits of data to load in parallel into the register.
- When SHIFT/$\overline{\text{LOAD}}$ is LOW, gates G_1 through G_3 are enabled, allowing each data bit to be applied to the D input of its respective flip-flop.
- When a clock is applied, the flip-flops with D=1 will set and those with D=0 will reset, thereby storing all four bits simultaneously.
- When SHIFT/$\overline{\text{LOAD}}$ is HIGH, gates G_1 through G_3 are disabled and G_4 through G_6 are enabled, allowing the data bits to shift right from one stage to the next.
- The OR gates allow either the normal shifting operation or parallel data-entry operation, depending on which AND gates are enabled by the level on the SHIFT/$\overline{\text{LOAD}}$ input.

Shift Registers

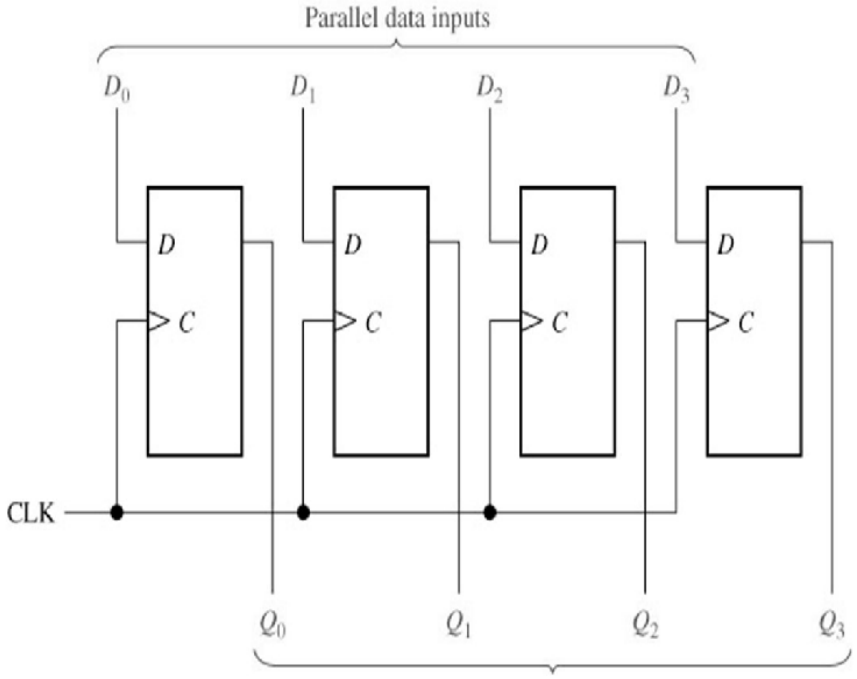
- 4-bit Parallel-in-Serial-out (Example)

Show the data-output waveform for a 4-bit register with the parallel input data and the clock and SHIFT/$\overline{\text{LOAD}}$ waveforms given.



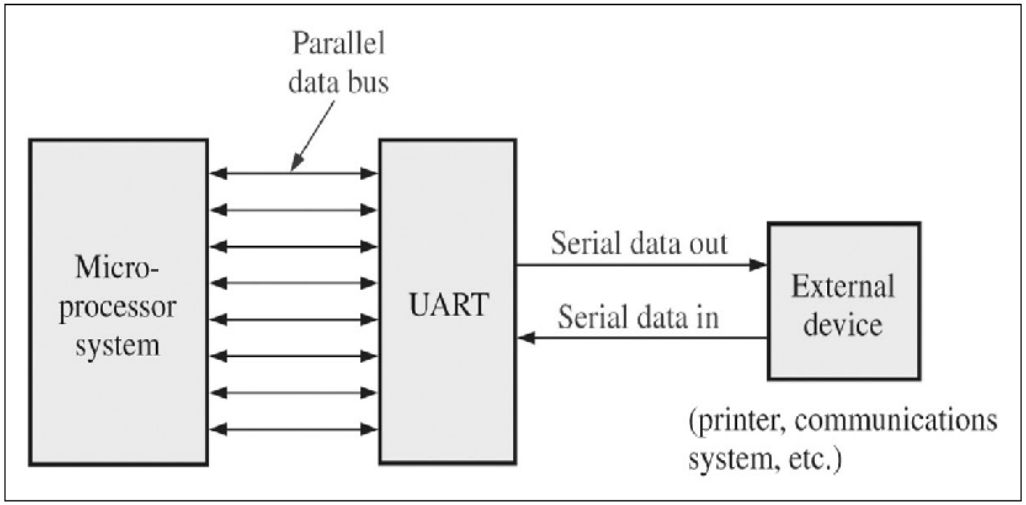
Shift Registers

- 4-bit Parallel-in-Parallel-out (Implementation)



Used for synchronous parallel data transfer whereby the logic levels present at the D inputs are transferred to the corresponding Q outputs when a positive going transition occurs at the clock C.

Shift Registers
- An application (Example-UART)

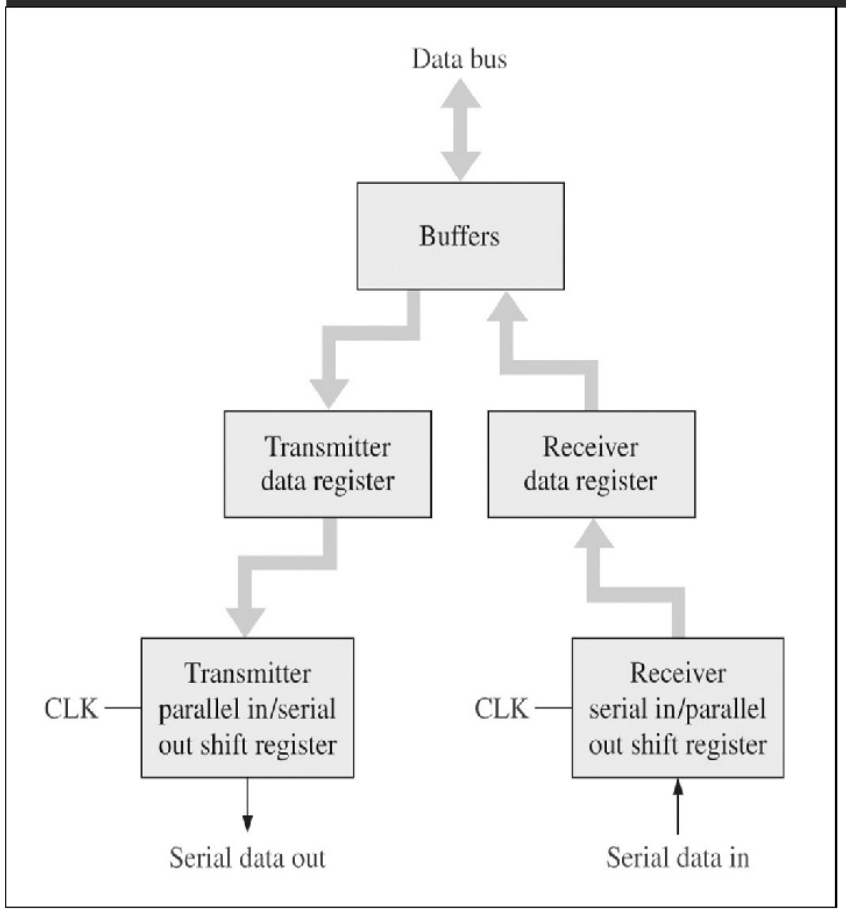


Universal Asynchronous Receiver-Transmitter (UART)

- Computers and microprocessor-based systems send and receive data in parallel format
- To communicate with external devices that send and/or receive serial data, interfacing device such as UART can be used.

Shift Registers

- An application (Example-UART)

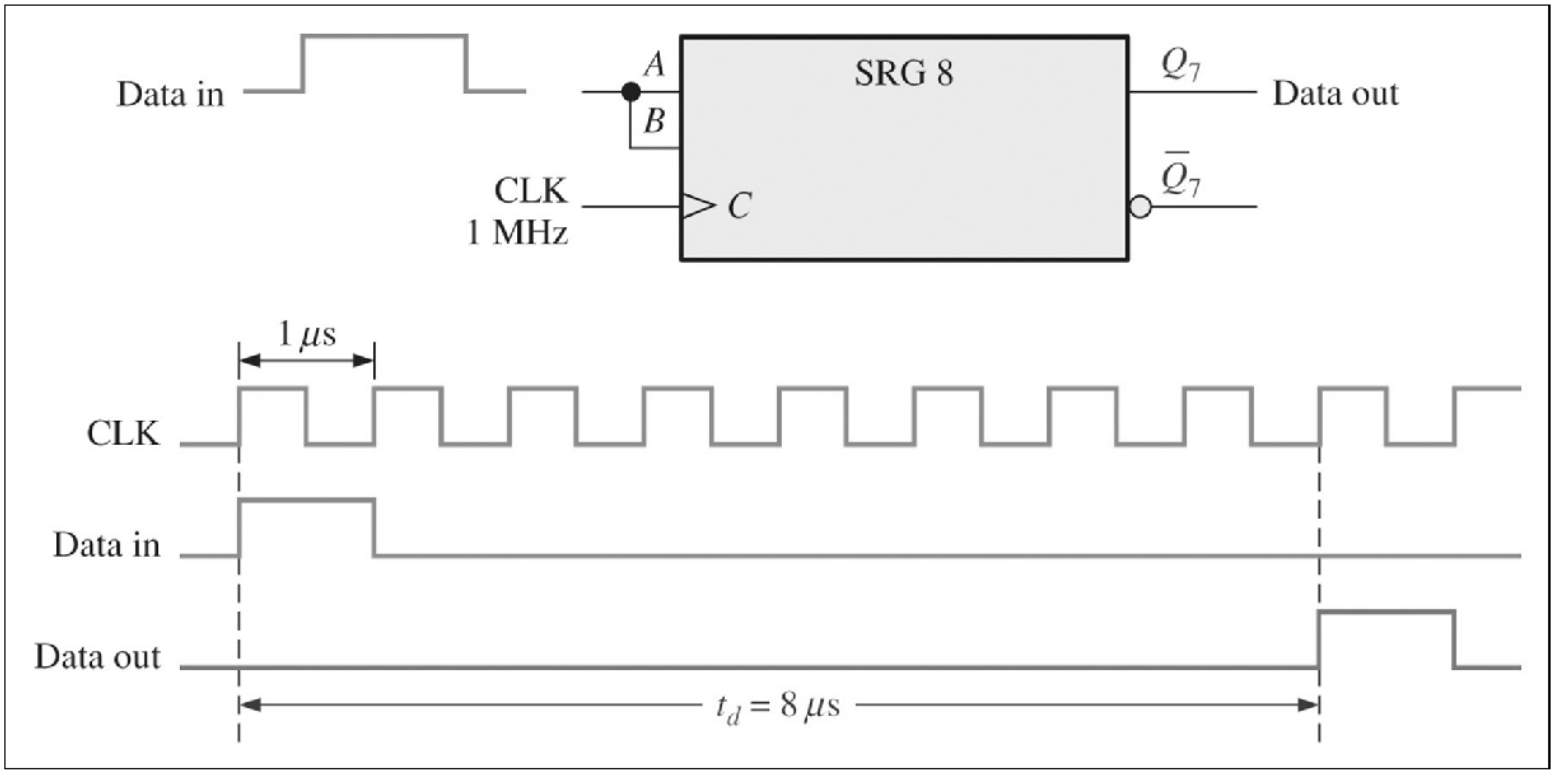


(UART) Block Diagram

- UART includes both serial-to-parallel and parallel-to-serial conversion.
- The data bus is a set of parallel conductors along which data move between UART and microprocessor system.
- Buffers interface the data registers with the data bus
- UART receives data in serial format, converts the data to parallel format, and places them on the data bus.
- UART also accepts parallel data from the data bus, converts the data to serial format, and transmits them to an external device

Shift Registers

- An application (Example-Time Delay circuit)



SISO shift register used to provide a time delay from input to output
that is a function of both the number of stages (n) in the register and clock frequency

Sequential Logic Circuits - Shift Register Counters

* Construction of Shift Register Counters - Examples
    * Johnson Counter
    * Ring Counter

Shift Register Counters

- Johnson Counter

- A Johnson counter will produce a modulus of 2n.
- A 4-bit Johnson counter has a total of 8 states and the 5-bit counter has a total of 10 states.
- The implementation of a Johnson counter is the same regardless of the number of stages.
- The Q output of each stage is connected to the D input of the next stage, except the $\overline{Q}$ output of the last stage is connected back to the D input of the first stage.

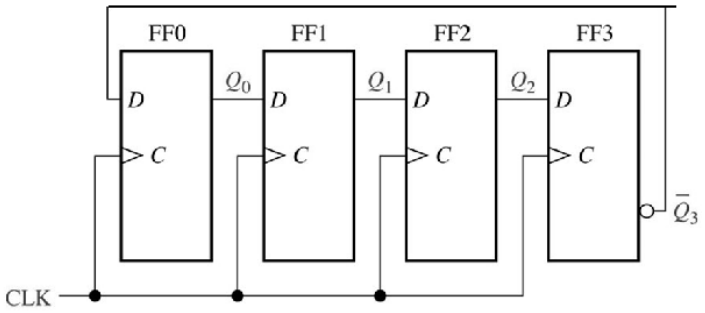
Shift Register Counters

- 4- bit Johnson Counter

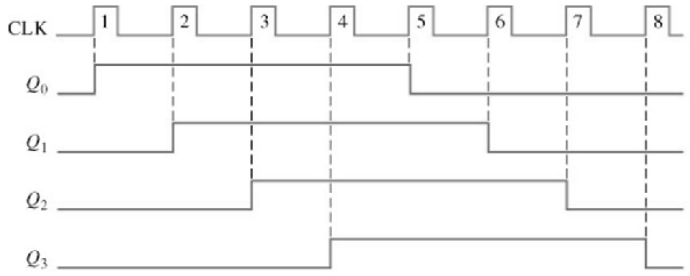
Four-bit Johnson sequence:

Clock Pulse	Q0	Q1	Q2	Q3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1

Logic diagram (4-bit Johnson counter)



Timing diagram (4-bit Johnson counter)

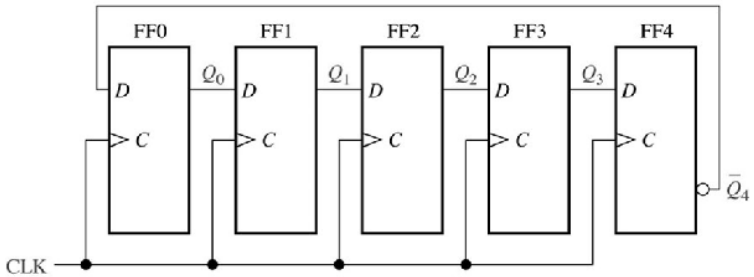


Shift Register Counters
- 5- bit Johnson Counter

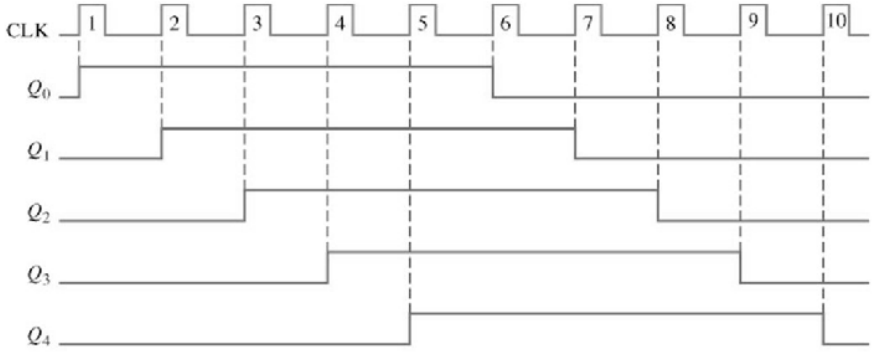
Five-bit Johnson sequence:

Clock Pulse	Q0	Q1	Q2	Q3	Q4
0	0	0	0	0	0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	1	0
5	0	1	1	1	1
6	0	1	1	1	1
7	0	0	1	1	1
8	0	0	0	1	1
9	0	0	0	0	1

Logic diagram (5-bit Johnson counter)



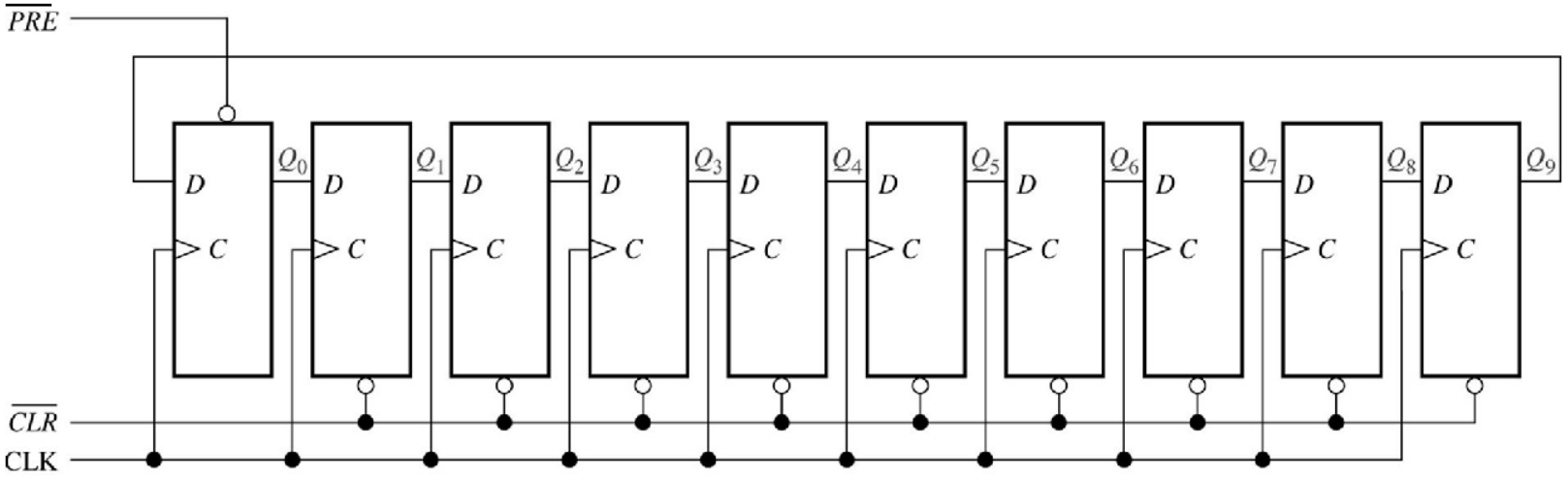
Timing diagram (5-bit Johnson counter)



Shift Register Counters
- Ring Counter (Logic Diagram)

▪Ring counter utilizes one flip flop for each state in its sequence. In the case of 10-bit ring counter, there is a unique output for each decimal digit. Example: FF0 represents a zero, FF1 represents a one, FF5 represents a five and so on.

•The inter-stage connections are the same as those for a Johnson counter, except that Q rather than $\overline{Q}$ is fed back from the last stage.



Logic diagram for a 10-bit ring counter

Initially preset = 1, therefore Q_0 will be 1 and the others will be cleared (0)

Shift Register Counters

- Ring Counter (Ten-bit counting sequence)

CLOCK PULSE	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9
0	1	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0
4	0	0	0	0	1	0	0	0	0	0
5	0	0	0	0	0	1	0	0	0	0
6	0	0	0	0	0	0	1	0	0	0
7	0	0	0	0	0	0	0	1	0	0
8	0	0	0	0	0	0	0	0	1	0
9	0	0	0	0	0	0	0	0	0	1

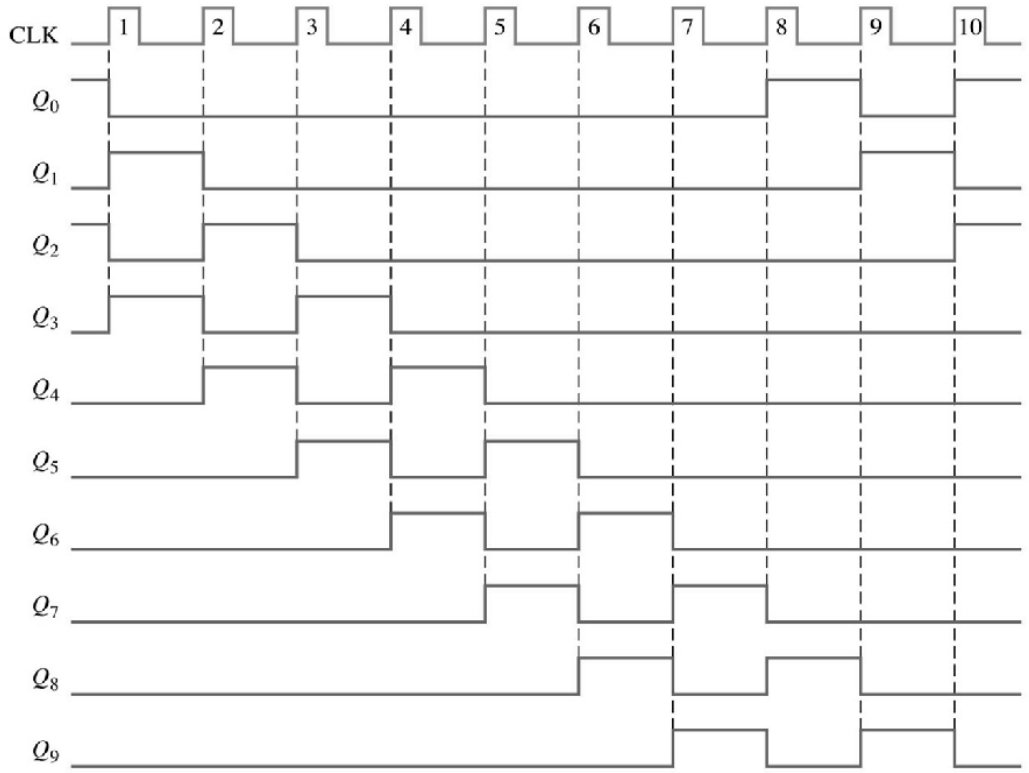
Shift Register Counters

- Ring Counter (Timing Diagram-Example)

Modified sequences can be achieved by having more than a single 1 in the counter.

Example:

If a 10-bit ring counter has the initial state 1010000000, determine the waveform for each of the Q outputs.



Flow table

	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_8	Q_8	Q_9
0	1	0	1	0	0	0	0	0	0	0
1	0	1	0	1	0	0	0	0	0	0
2	0	0	1	0	1	0	0	0	0	0
3	0	0	0	1	0	1	0	0	0	0
4	0	0	0	0	1	0	1	0	0	0
5	0	0	0	0	0	1	0	1	0	0
6	0	0	0	0	0	0	1	0	1	0
7	0	0	0	0	0	0	0	1	0	1
8	1	0	0	0	0	0	0	0	1	0
9	0	1	0	0	0	0	0	0	0	1
10	1	0	1	0	0	0	0	0	0	0
11			And	repeat						

References

Slides adopted from the books

- Thomas L.Floyd, “Digital Fundamentals,” 10th Edition, Pearson Education International, 2009
- Ronald J.Tocci, Neal S.Widmer, and Gregory L.Moss, "Digital Systems- Principles and Application"- 10th Edition, Pearson Education International, 2007
- M.Morris Mano and Michael D. Ciletti, " Digital Design," 5th Edition, Pearson Education International, 2012