CS3351 DIGITAL PRINCIPLES AND COMPUTER ORGANIZATION L T P C
3 0 2 4

COURSE OBJECTIVES:

- To analyze and design combinational circuits.
- To analyze and design sequential circuits
- To understand the basic structure and operation of a digital computer.
- To study the design of data path unit, control unit for processor and to familiarize with the hazards.
- To understand the concept of various memories and I/O interfacing.

UNIT I COMBINATIONAL LOGIC 9

Combinational Circuits – Karnaugh Map - Analysis and Design Procedures – Binary Adder – Subtractor – Decimal Adder - Magnitude Comparator – Decoder – Encoder – Multiplexers - Demultiplexers

UNIT II SYNCHRONOUS SEQUENTIAL LOGIC 9

Introduction to Sequential Circuits – Flip-Flops – operation and excitation tables, Triggering of FF, Analysis and design of clocked sequential circuits – Design – Moore/Mealy models, state minimization, state assignment, circuit implementation - Registers – Counters.

UNIT III COMPUTER FUNDAMENTALS 9

Functional Units of a Digital Computer: Von Neumann Architecture – Operation and Operands of Computer Hardware Instruction – Instruction Set Architecture (ISA): Memory Location, Address and Operation – Instruction and Instruction Sequencing – Addressing Modes, Encoding of Machine Instruction – Interaction between Assembly and High Level Language.

UNIT IV PROCESSOR 9

Instruction Execution – Building a Data Path – Designing a Control Unit – Hardwired Control, Microprogrammed Control – Pipelining – Data Hazard – Control Hazards.

UNIT V MEMORY AND I/O 9

Memory Concepts and Hierarchy – Memory Management – Cache Memories: Mapping and Replacement Techniques – Virtual Memory – DMA – I/O – Accessing I/O: Parallel and Serial Interface – Interrupt I/O – Interconnection Standards: USB, SATA

45 PERIODS

PRACTICAL EXERCISES: 30 PERIODS

1. Verification of Boolean theorems using logic gates.
2. Design and implementation of combinational circuits using gates for arbitrary functions.
3. Implementation of 4-bit binary adder/subtractor circuits.
4. Implementation of code converters.
5. Implementation of BCD adder, encoder and decoder circuits
6. Implementation of functions using Multiplexers.
7. Implementation of the synchronous counters
8. Implementation of a Universal Shift register.
9. Simulator based study of Computer Architecture

COURSE OUTCOMES:

At the end of this course, the students will be able to:

CO1 : Design various combinational digital circuits using logic gates

CO2 : Design sequential circuits and analyze the design procedures

CO3 : State the fundamentals of computer systems and analyze the execution of an instruction

CO4 : Analyze different types of control design and identify hazards

CO5 : Identify the characteristics of various memory systems and I/O communication

TOTAL: 75 PERIODS

TEXT BOOKS:

1. M. Morris Mano, Michael D. Ciletti, “Digital Design : With an Introduction to the Verilog HDL, VHDL, and System Verilog”, Sixth Edition, Pearson Education, 2018.

2. David A. Patterson, John L. Hennessy, “Computer Organization and Design, The Hardware/Software Interface”, Sixth Edition, Morgan Kaufmann/Elsevier, 2020.

REFERENCES:

1. Carl Hamacher, Zvonko Vranesic, Safwat Zaky, Naraig Manjikian, “Computer Organization and Embedded Systems”, Sixth Edition, Tata McGraw-Hill, 2012.

2. William Stallings, “Computer Organization and Architecture – Designing for Performance”, Tenth Edition, Pearson Education, 2016.

3. M. Morris Mano, “Digital Logic and Computer Design”, Pearson Education, 2016.

CS3351 - DIGITAL PRINCIPLES & COMPUTER ORGANIZATION

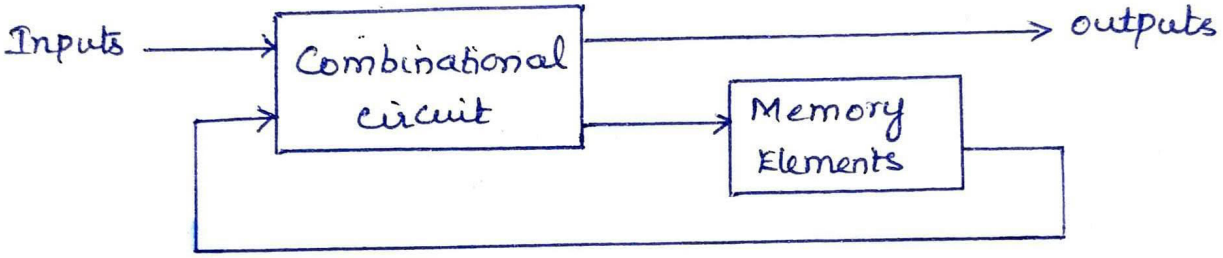
UNIT II

SYNCHRONOUS SEQUENTIAL LOGIC

Introduction to Sequential Circuits – Flip-Flops – operation and excitation tables, Triggering of FF, Analysis and design of clocked sequential circuits – Design – Moore/Mealy models, state minimization, state assignment, circuit implementation - Registers – Counters.

Sequential Circuits

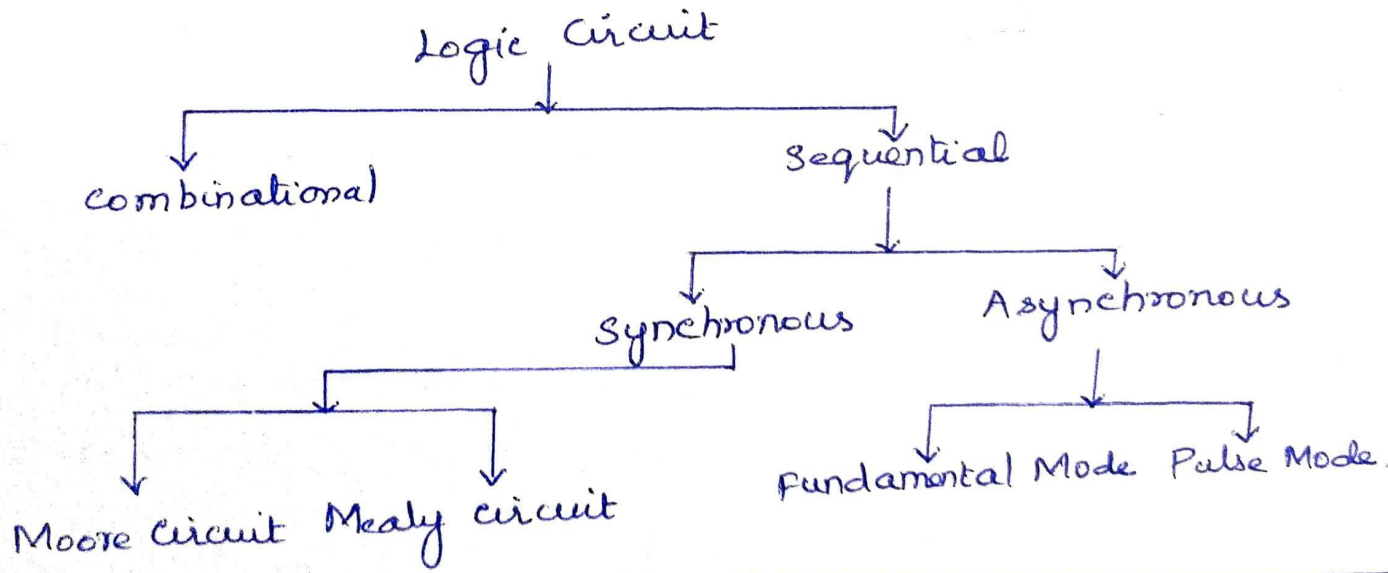
Block Diagram of Sequential Circuit



* Consists of a combinational circuit to which storage elements are connected to form a feedback path.
  - The storage elements are devices capable of storing binary information.
  - Binary information stored in these elements at any given time defines the State of the sequential circuit at that time.
* The outputs in a sequential circuit are a function not only of the inputs, but also of the present state of the storage elements.
  - The next state of the storage elements is also a function of external inputs and the present state.
* A sequential circuit is specified by a time sequence of inputs, outputs and internal states.

Sl No.	Combinational circuits	Sequential circuits
1.	The outputs at any time are determined from the present combination of inputs	The outputs are a function not only of the inputs, but also of the present state of the storage elements
2.	Memory unit is not required	Memory unit is required to store the past information
3	Faster in Speed	Slower than Combinational circuits
4.	Easy to design	Comparatively harder to design
5.	Ex: Parallel Adder	Ex: Serial Adder.

Classification of Logic circuits :



Sequential < Synchronous / Asynchronous

Synchronous sequential circuits :

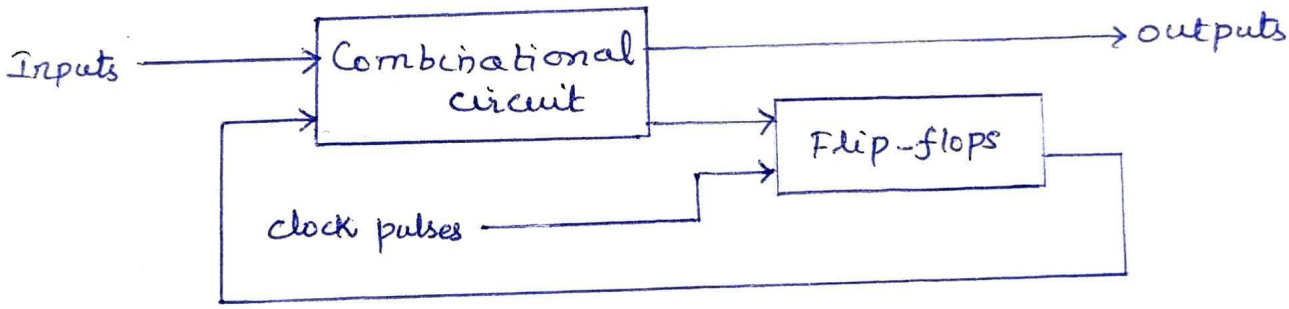
* A synchronous sequential circuit is a system whose behaviour can be defined from the knowledge of its signals at discrete instants of time.

* employs signals that affect the storage elements only at discrete instants of time.
* Synchronization is achieved by a timing device called a clock generator that provides a periodic train of clock pulses.
* Use clock pulses in the inputs of storage elements called clocked sequential circuits.

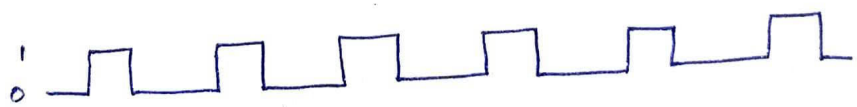
→ The storage elements used in clocked sequential circuits are called flip-flops.

* A flip-flop is a binary storage device capable of storing one bit of information.

Synchronous clocked sequential Circuit



Timing diagram of clock pulses



* The outputs can come either from the combinational circuit or from the flip-flops or both
* The flip-flops receive their inputs from the combinational circuit and also from a clock signal with pulses that occur at fixed intervals of time.

→ The state of the flip-flops can change only during a clock pulse transition.

* When a clock pulse is not active, the feedback loop is broken because the flip-flop outputs cannot change even if the outputs of the combinational circuit driving their inputs change in value.

→ The transition from one state to the next occurs only at predetermined time intervals dictated by the clock pulses.

S.No	Synchronous sequential circuits	Asynchronous Sequential circuits
1.	Memory elements are clocked Flip-flops	Memory elements are either unclocked Flip-flops or time-delay elements
2.	The change in input signals can affect memory element upon activation of clock signal.	The change in input signals can affect memory element at any instant of time
3.	The maximum operating speed of clock depends on time delays involved	Because of the absence of clock, it can operate faster than synchronous circuits
4.	Easier to design	More difficult to design

Storage Elements

1. Latches
2. Flip-flops

* A flip-flop is a binary storage device capable of storing one bit of information.
* can maintain a binary state indefinetely until directed by an input signal to switch states.

- Types based on i) the number of inputs
  ii) the manner in which the inputs affect the binary state.

Latches:

* Basic types of flip-flops operate with signal levels and are referred to as latches.
- basic circuits from which all flip-flops are constructed.
* They are not practical for use in synchronous sequential circuits.

Latches

① SR Latch
② D latch

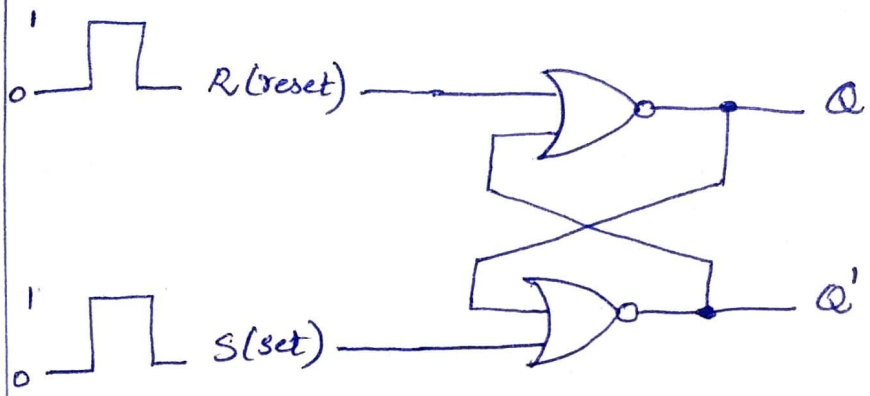
① SR Latch

* The SR latch is a circuit with two cross-coupled NOR gates or two cross-coupled NAND gates.
* Two inputs: S → set
  R → Reset

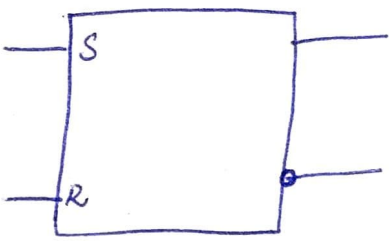
a) SR Latch using NOR gates:

Two states
i) Set state: $Q=1, Q'=0$
ii) Reset State: $Q=0, Q'=1$
*Undefined State: i/p → both = 1 , o/p → 0

Logic Diagram



Logic Symbol



Function Table

S	R	Q	Q'	
1	0	1	0	
0	0	1	0	(after S=1, R=0) — Set State
0	1	0	1	
0	0	0	1	(after S=0, R=1) — Reset State
1	1	0	0	

S=0, R=0

* Under normal conditions, both inputs of the latch remain at 0 unless the state has to be changed.
* latch can be in either the set or reset state.

S=1, R=0 (Set State)

* S=1, input causes the latch to go to the Set state
* The S input must go back to 0 before any other changes to avoid the occurrence of the undefined state.

S=0, R=1 (Reset state)

* R=1, the latch shifts to reset state.
* R - go back to 0, the circuit remains in the reset state.

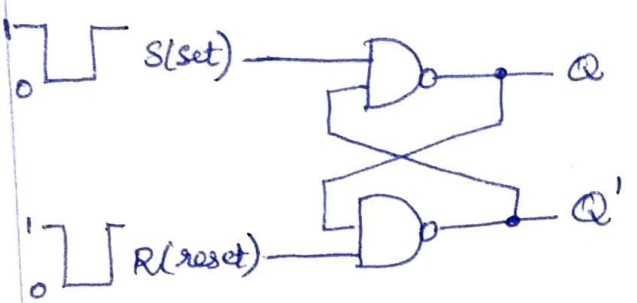
S=1, R=1

* outputs go to 0
* undefined state, unpredictable next state

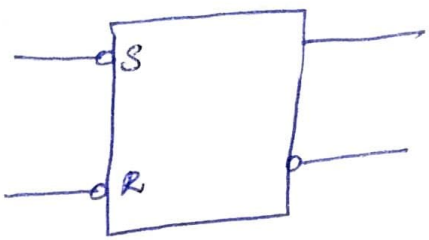
Outputs: Q & Q' → complement of each other.

b) SR Latch using NAND gates : (S'-R' Latch)

Logic diagram



Logic Symbol $\bar{S}\bar{R}$



Function Table

S	R	Q	Q'	
1	0	0	1	(after S=1, R=0)
1	1	0	1	
0	1	1	0	(after S=0, R=1)
1	1	1	0	
0	0	1	1	

S=1, R=1

* Inputs normally at 1 unless the state of the latch has to be changed

S=0, R=1 (set state)

* Apply S=0, input causes output Q to go to 1.
* latch in the set state
* S input goes back to 1, the circuit remains in the set state.

S=1, R=0 (Reset state)

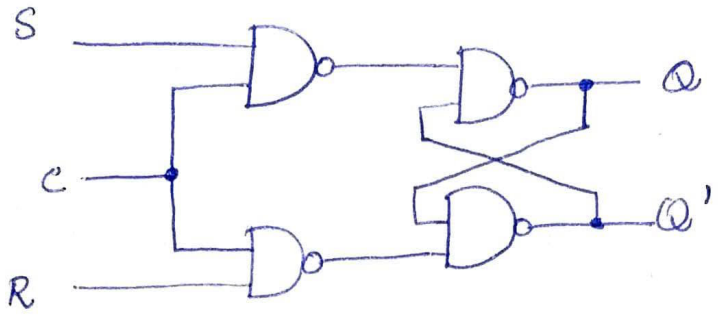
* change R=0, the circuit goes to reset state.
* R goes back to 1, the circuit remains in the reset state

S=0, R=0

* undefined state.

c) SR Latch with Control Input

Logic diagram



Function Table

C	S	R	Next State of Q
0	X	X	No change
1	0	0	No change
1	0	1	Q = 0, Reset State
1	1	0	Q = 1, Set State
1	1	1	Indeterminate

* SR Latch with two additional NAND gates
* control input C acts as an enable signal for the other two inputs.

C = 0 :
* NAND → logic 1
* The circuit remains in its current state
* disables the circuit, the output does not change regardless of the values of S and R.

C = 1 :
* information from the S or R input is allowed to affect the SR latch.

i) **Set State:**
S = 1, R = 0, C = 1

ii) **Reset State:**
S = 0, R = 1, C = 1

iii) **Undefined state / indeterminate condition**
S = 1, R = 1, C = 1

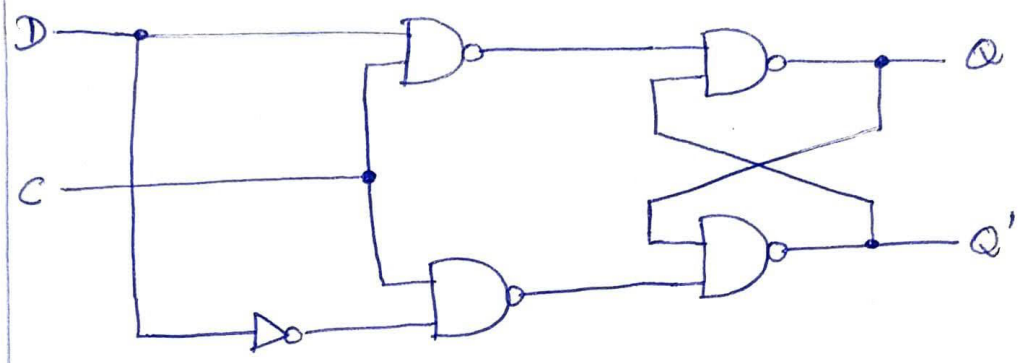
iv) **C = 1, S = 0, R = 0 :**
* circuit does not change.

② D Latch : (Transparent Latch)

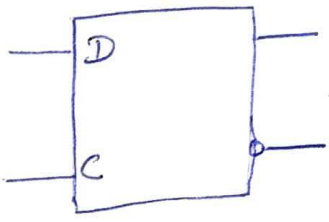
* To eliminate the undesirable condition of the indeterminate state in the SR latch is to ensure that inputs S and R are never equal to 1 at the same input.

→ done in the D latch.

Logic diagram



Logic Symbol



Function Table

C	D	Next state of Q
0	X	No change
1	0	Q = 0, Reset state
1	1	Q = 1, Set State

* Latch has two inputs : D (data) and C (control)

C = 0

* The circuit cannot change state regardless of the value of D.

C = 1

* The D input is sampled

D = 1 :

* The Q output goes to 1 → set state

D = 0 :

* The output Q goes to 0 → reset state

D Latch:

* Use as a temporary storage for binary information between a unit and its environment
* Binary information present at the data input of the D latch is transferred to the Q output when the control input is enabled.

Flip-flops

* The state of a latch or flip-flop is switched by a change in the control input
  - The momentary change is called a trigger
  - transition it causes is said to trigger the flip flop.
* A sequential circuit has a feedpack path from the outputs of the flip-flops to the input of the combinational circuit.

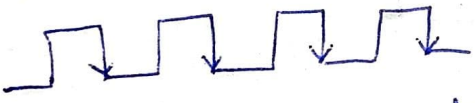
Latch → responds to a change in the level of a clock pulse
flip-flop → trigger it only during a signal transition.

Clock Response in Latch and Flip-flop

(a) Response to positive level

(b) Positive-edge response

(c) Negative-edge response



* A clock pulse goes through two transitions:
  → from 0 to 1 and return from 1 to 0.

Edge Triggered Flip-flops:

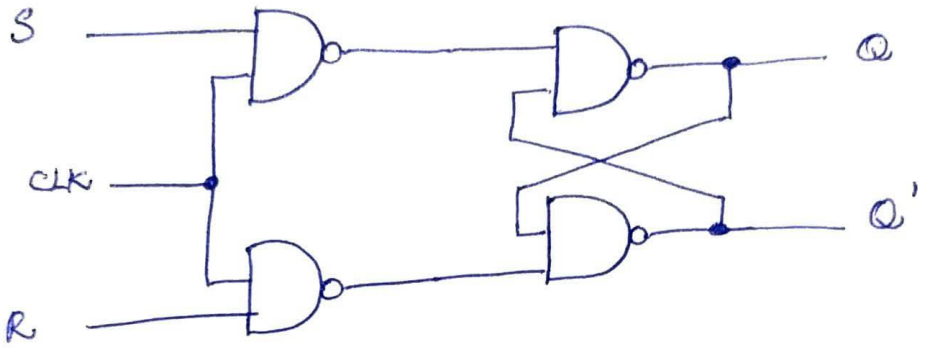
* change state either at the positive edge or at the negative edge of the clock pulse and is sensitive to its inputs only at the transition of the clock.

Types:
1) SR Flip-flop
2) J-K Flip-flop
3) D Flip-flop
4) T Flip-flop

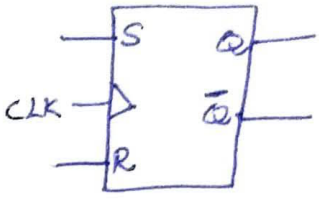
1) SR Flip-flop :

* Similar to S-R Latch
  - enable signal is replaced by clock pulse (CLK)
* The S and R inputs are called synchronous inputs because data on the inputs are transferred to the Flip-Flop's output only on the triggering edge of the clock pulse.

Logic diagram



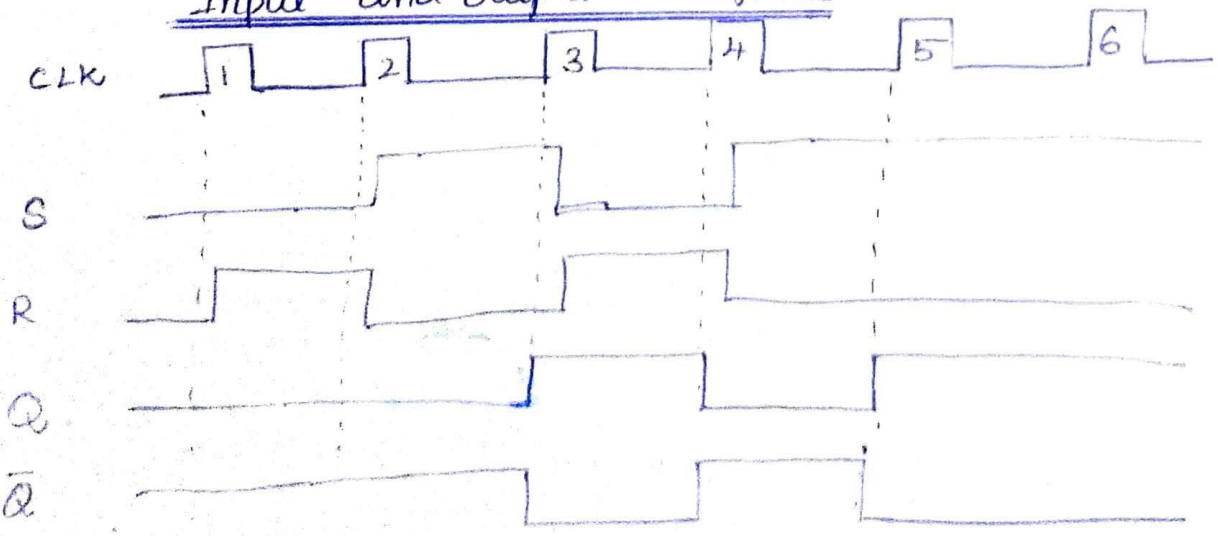
Logic symbol



Function Table

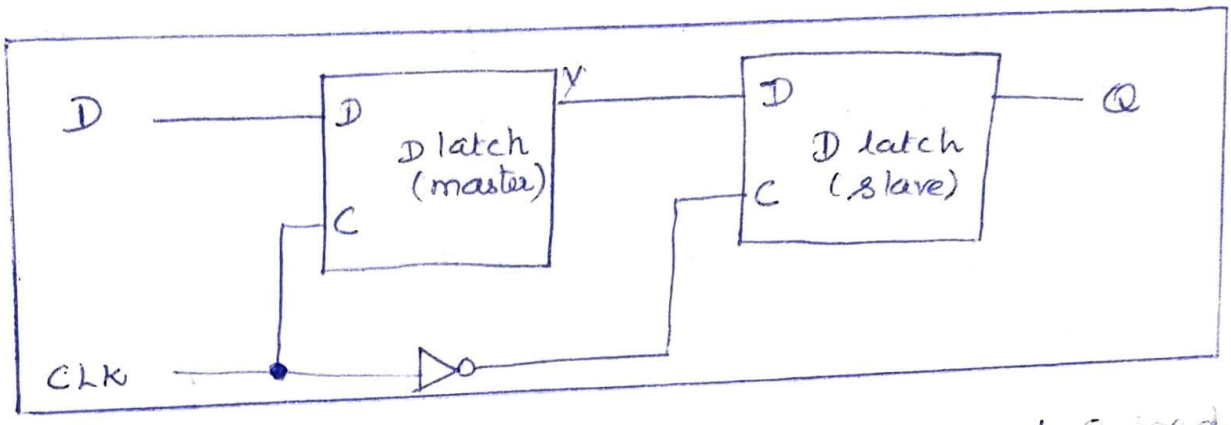
CLK	S	R	State
↓	0	0	No change
↓	0	1	Reset
↓	1	0	Set
↓	1	1	Indeterminate

Input and output Waveforms



2) D Flip-flop :

* Constructed with two D latches and an inverter.



* The first latch is called the master and second the slave.

– The circuit samples the D input and changes its output Q only at the negative-edge of the clock.

CLK=0 :

* When the clock is 0, the output of the inverter is 1.
* The slave latch is enabled and its output Q is equal to the master output Y.

– The master latch is disabled because CLK=0.

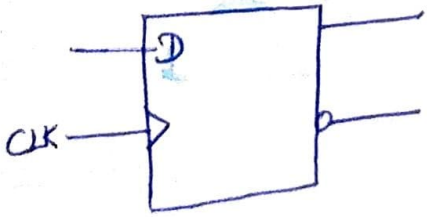
CLK=1 :

* When the input pulse changes to the logic 1 level, the data from the external D input is transferred to the master.

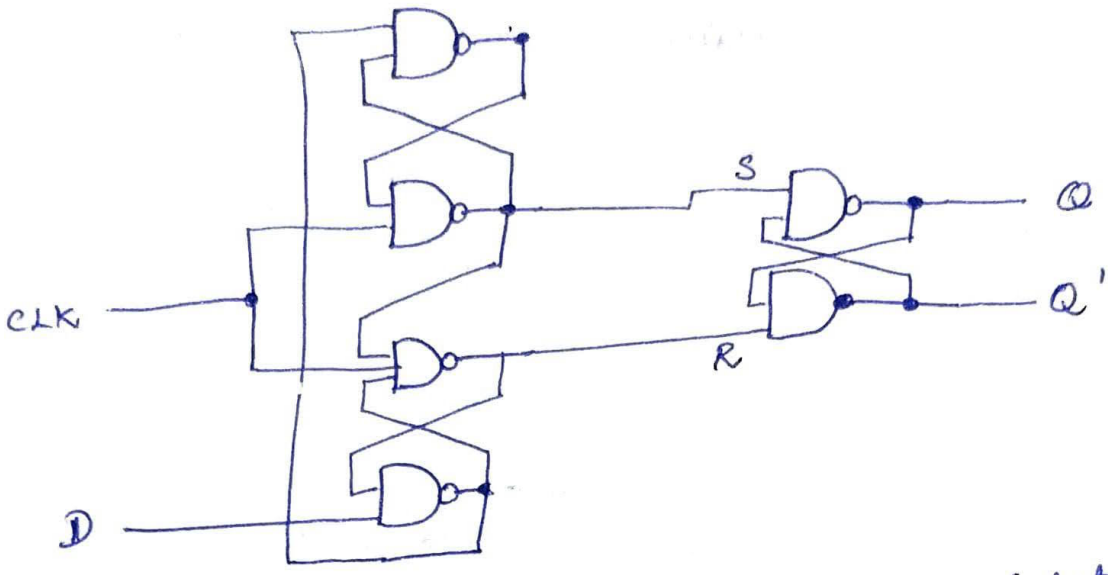
→ The slave is disabled as long as the clock remains in the 1 level.

* The output of the flip-flop can change only during the transition of the clock from 1 to 0.

Graphic Symbol



Construction of an edge-triggered D flip-flop uses three SR Latches:



* Two latches respond to the external D (data) and CLK (clock) inputs
* Third latch provides the outputs of the flipflop.

CLK = 0

* logic 1 level.
* The output to remain in its present state.

D = 0, CLK = 1

* R = 0
* Flip-flop goes to reset state
* Q = 0

D = 1, CLK = 1

* S changes to 0
* Flip-flop goes to set state
* Q = 1

* When the input clock in the positive-edge-triggered flip-flop makes a positive transition
  - the value of D is transferred to Q.
* A negative transition from 1 to 0 does not affect the output.

Characteristic Table:

D	Q(t+1)	
0	0	Reset
1	1	Set

Q(t+1) → next state one clock period later.

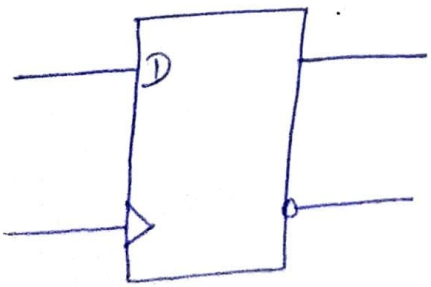
Characteristic Equation:

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

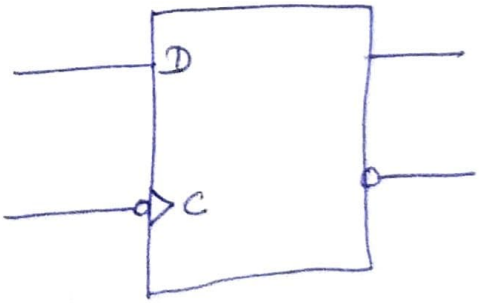
* The next state of the output will be equal to the value of input D in the present state.

Graphic Symbol for Edge-Triggered D Flip-Flop

(a) Positive-edge



(b) Negative-edge



* Most economical and efficient flip-flop construction is the edge-triggered D flip-flop
  - It requires the smallest number of gates.
* Other types of flip-flops can be constructed by using the D flip-flop and external logic.

3) J-K Flip-flop:

J-K → Jack Kilby.

* Two inputs J (set) and K (reset)

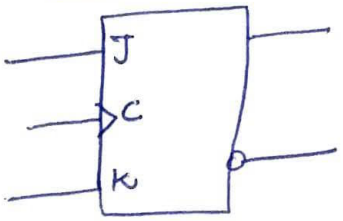
Three operations:

i) set it to 1

ii) reset it to 0

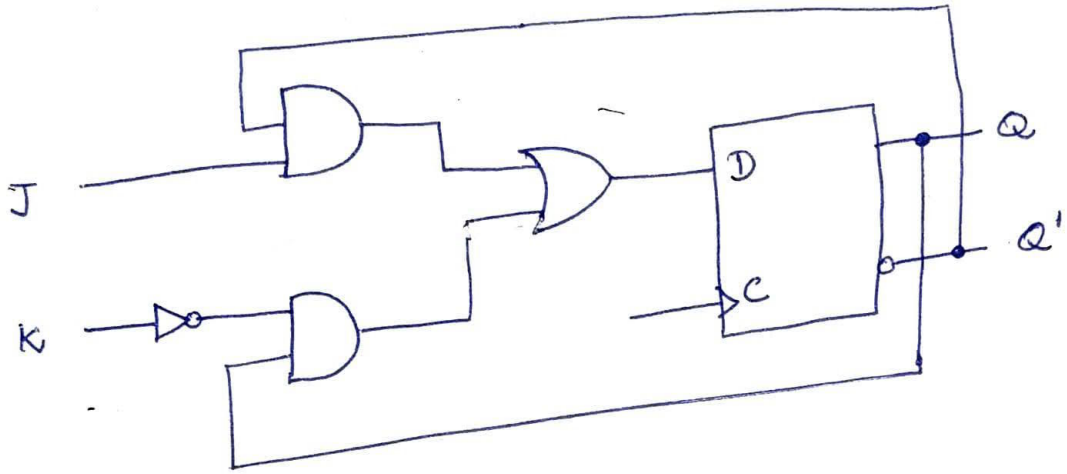
iii) complement the output.

Graphic Symbol



Circuit diagram

constructed with a D flip-flop and gates



Characteristic Table

J	K	Q(t+1)	
0	0	Q(t)	No change
0	1	0	Reset
1	0	1	Set
1	1	Q'(t)	complement

Characteristic Equation :

$$Q(t+1) = JQ' + K'Q$$

$$D = JQ' + K'Q$$

J = 1, K = 0

* The next clock edge sets the output to 1
* J input sets the flip-flop to 1

$$D = Q' + Q = 1$$

J = 0, K = 1

* next clock edge resets the output to 0
* K input resets it to 0

J = K = 1

* The next clock edge complements the output.

$$D = Q'$$

J = K = 0

* The clock edge leaves the output unchanged

$$D = Q.$$

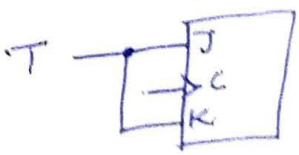
4) **T Flip-flop:**

* **Toggle** flip-flop, complementing flip-flop

Using J-K Flip-flop:

* obtained from a JK flip-flop when inputs J and K are tied together.

Graphic Symbol



T = 0

* J = K = 0
* a clock edge does not change the output.

T = 1

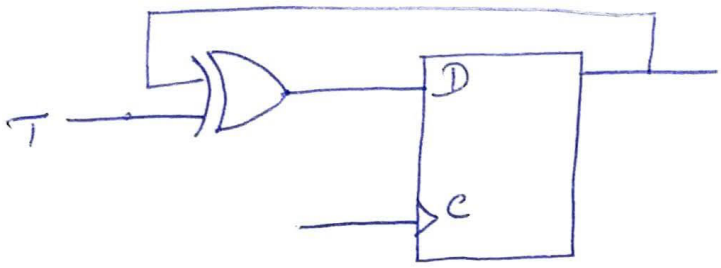
* J = K = 1
* a clock edge complements the output.

Application: useful for designing binary counters

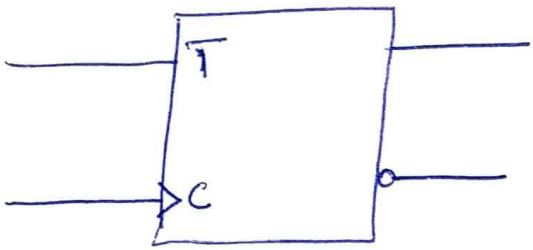
Using D flip-flop.

* T flip-flop can be constructed with a D flip-flop and an exclusive OR gate.

circuit diagram



Graphic Symbol



characteristic Table:

T	Q(t+1)	
0	Q(t)	No change
1	Q'(t)	Complement

characteristic Equation

$$Q(t+1) = T \oplus Q = TQ' + T'Q$$

T=0

* D = Q
* No change in the output

T=1

* D = Q'
* The output complements

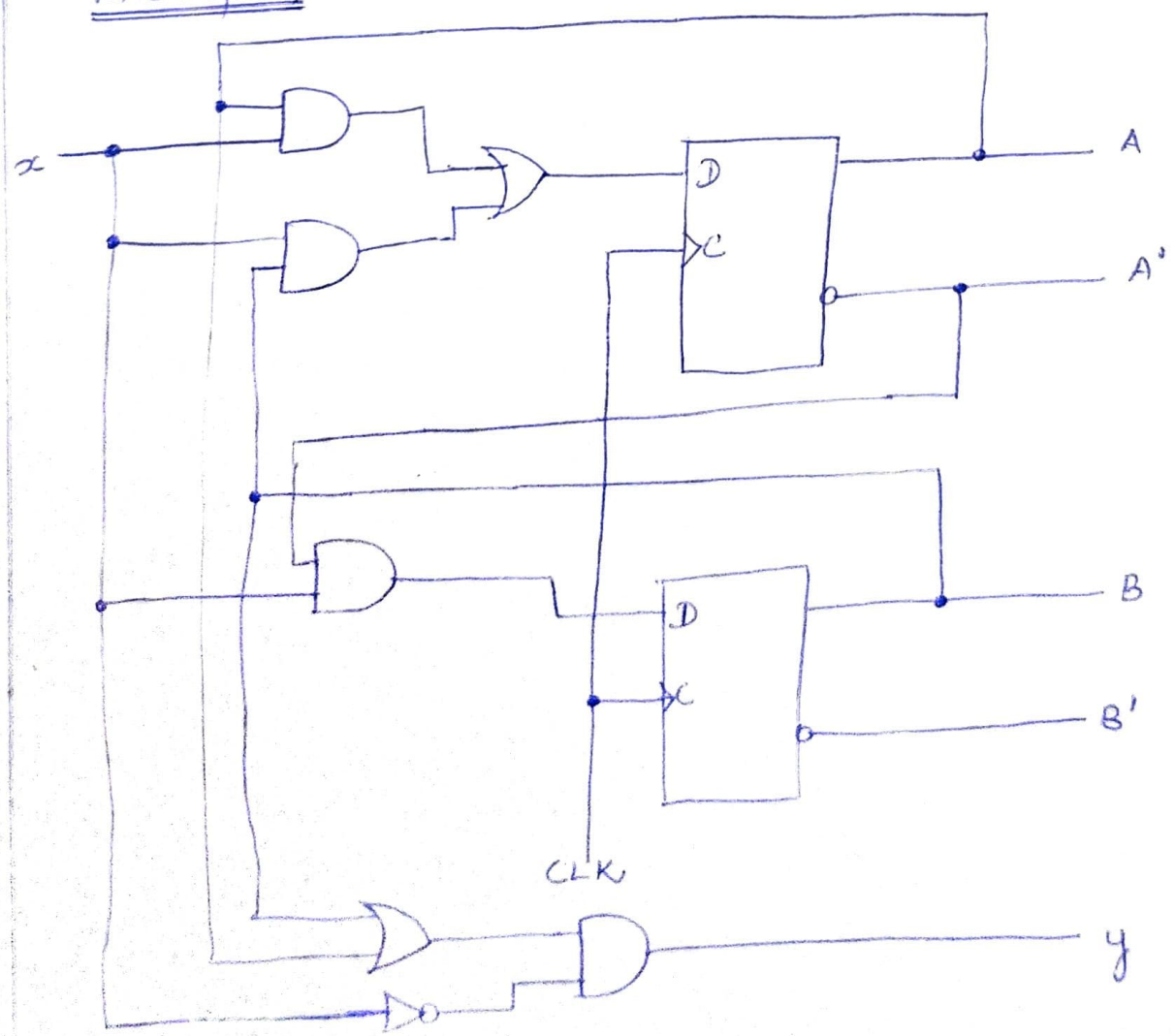
Analysis of clocked Sequential circuits

* Analysis describes what a given circuit will do under certain operating conditions
* The behavior of a clocked sequential circuit is determined from the inputs, the outputs and the state of its flip-flops

→ The outputs and the next state are both a function of the inputs and the present state.

* **Analysis**
  * consists of obtaining a table or a diagram for the time sequence of inputs, outputs and internal states.
    * State Equation
    * State Table
    * State Diagram.

Example:



Step 1:

State Equations

* Specify the next state as a function of the present state and inputs.

Ex: 2 D flip-flops → A, B
input → x
output → y

$$A(t+1) = A(t)\,x(t) + B(t)\,x(t)$$
$$B(t+1) = A'(t)\,x(t)$$
$$y(t) = [A(t) + B(t)]\,x'(t)$$

(t+1) → next state of the flip-flop one clock edge later.

* All the variables in the Boolean expressions are a function of the present state.

$$A(t+1) = Ax + Bx$$
$$B(t+1) = A'(x)$$
$$y = (A + B)x'$$

Step 2:

State Table / Transition Table

* The time sequence of inputs, outputs and flip-flop states can be enumerated in a state table.

→ The table consists of four sections

i) present state
ii) input
iii) next state
iv) output.

Present state → states of flip-flops A and B at any given time t.

input → a value of x for each possible present state.

Next state → states of the flip-flops one clock cycle later at time t+1

output → value of y at time t for each present state and input condition.

Present state		Input	Next state		output
A	B	x	A	B	y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

* Listing all binary combinations of present state and inputs
* Next state values are determined from the logec diagram or from the state equations

$$A(t+1) = Ax + Bx$$

$$B(t+1) = A'x$$

$$y = Ax' + Bx'$$

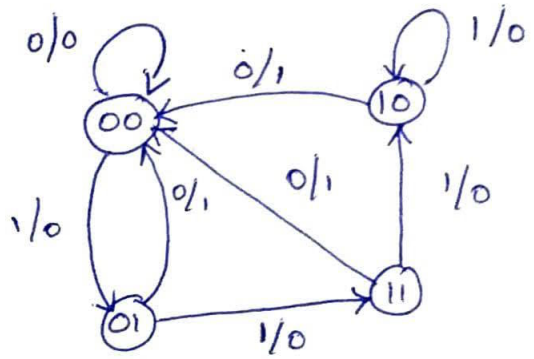
Second Form of the State Table

m flip-flops
n inputs
} $\Rightarrow 2^{m+n}$ rows

Present State	Next State		Output	
	$x=0$	$x=1$	$x=0$	$x=1$
A B	A B	A B	y	y
0 0	0 0	0 1	0	0
0 1	0 0	1 1	1	0
1 0	0 0	1 0	1	0
1 1	0 0	1 0	1	0

State Diagram

* The information available in a state table can be represented graphically in the form of a state diagram.
  - → A state is represented by a circle
  - → the transitions between states are indicated by directed lines connecting the circles.



* The binary number inside each circle identifies the state of the flip-flops.
* The directed lines are labeled with two binary numbers separated by a slash.
  - input value during the present state is labeled first
  - the number after the slash gives the output during the present state with the given input

* The state diagram gives a pictorial view of state transitions

- more suitable for human interpretation of the circuit operation

Output Equations:

* The part of the combinational circuit that generates external outputs is described algebraically by a set of Boolean functions called output equations.

Input Equations:

* The part of the circuit that generates the inputs to flip-flops is described algebraically by a set of Boolean functions called the flip-flop input equations (excitation equations)

$$D_Q = x + y$$

↓ name of the flip flop.

EX:

$$D_A = Ax + Bx$$
$$D_B = A'x$$
$$y = (A+B)x'$$

a) Analysis with D Flip-flops:

EX:

Input equation : $D_A = A \oplus x \oplus y$

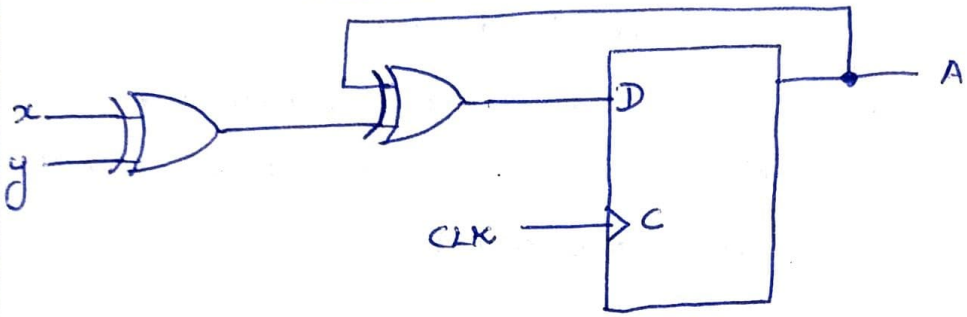
$D_A \rightarrow$ D flip-flop with output A.

$x, y \rightarrow$ inputs.

* No output equations are given.

Step 1:

Logic diagram



Step 2: **State Table**

* One column for the present state for flip-flop A
* Two columns for the two inputs
* one column for the next state of A.

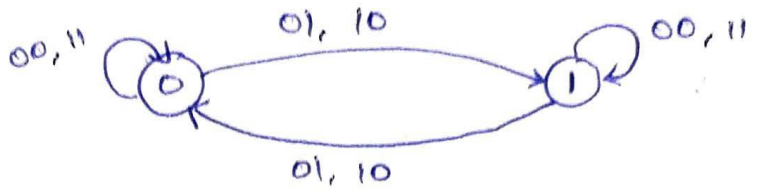
Present State	Inputs		Next State
A	x	y	A
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

* Next state values are obtained from the state equation:

$$A(t+1) = A \oplus x \oplus y$$

* State equation is the same as the input equation.

Step 3: State Diagram



* The circuit has one flip-flop and two states 0, 1
* A slash on the directed lines is not needed because there is no output from a combinational circuit

b) Analysis with JK Flip-Flops:

* To obtain the next state values,
  - refer corresponding characteristic table or characteristic equation.

i) * Procedure:

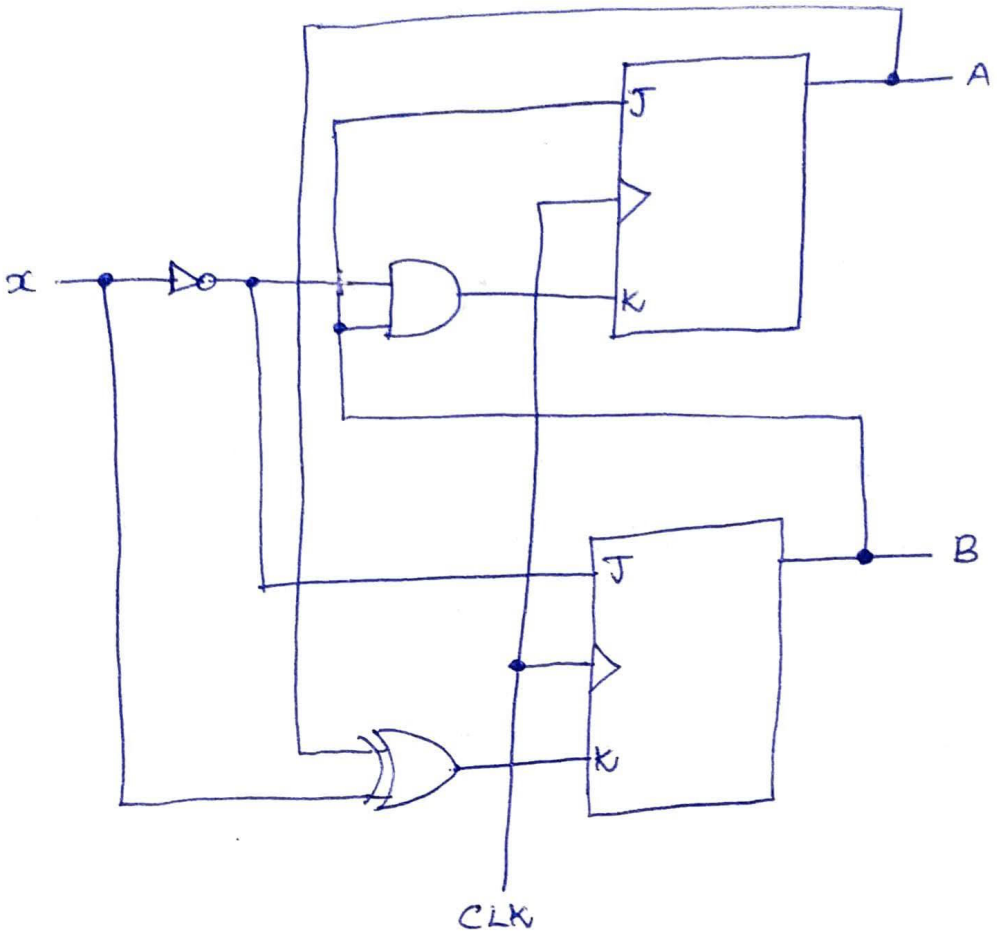
→ The next-state values of a sequential circuit that uses flip-flops such as JK or T type can be derived using the following procedure.

1. Determine the flip-flop input equation in terms of the present state and input variables.
2. List the binary values of each input equation
3. Use the corresponding flip-flop characteristic table to determine the next state values in the state table.

Example:

* Two JK flip-flops A and B
* one input x
* no outputs.

Sequential circuit with JK Flip-Flop



Soln:

Step 1: Flip-flop Input equations:

$J_A = B$ $\quad K_A = Bx'$

$J_B = x'$ $\quad K_B = A'x + Ax' = A \oplus x$

Characteristic Table:

J	K	Next Q(t)
0	0	Q(t)
0	1	0
1	0	1
1	1	Q'(t)

Step 2: State Table

Present State		Input	Next State		Flip-Flop Inputs			
A	B	x	A	B	J_A	K_A	J_B	K_B
0	0	0	0	1	0	0	1	0
0	0	1	0	0	0	0	0	1
0	1	0	1	1	1	1	1	0
0	1	1	1	0	1	0	0	1
1	0	0	1	1	0	0	1	1
1	0	1	1	0	0	0	0	0
1	1	0	0	0	1	1	1	1
1	1	1	1	1	1	0	0	0

i) Next state values using characteristic Table.

* J_A, K_A } evaluated using input equations
  J_B, K_B }

* Next state – evaluated using characteristic Table.

$A \rightarrow J_A, K_B$

$B \rightarrow J_B, K_B$

J	K	Next state
0	0	Q(t)
0	1	0
1	0	1
1	1	Q'(t)

ii) Next State values using State Equations:

* The next state values can be obtained also by evaluating the state equations from the characteristic equation.

Procedure:

1. Determine the flip-flop input equations in terms of the present state and input variables.
2. Substitute the input equations into the flip-flop characteristic equation to obtain the state equations
3. Use the corresponding state equations to determine the next state values in the state table.

Example

Step 1: Input equations:

Step 2: State equations:

characteristic equation for JK F/F : $Q(t+1) = JQ' + K'Q$

$A(t+1) = JA' + K'A$

$B(t+1) = JB' + K'B$

Substitute the input equations in characteristic eqns

$J_A = B$
$K_A = Bx'$
$J_B = x'$
$K_B = A \oplus x$

$$A(t+1) = J_A A' + K_A' A$$
$$= BA' + (Bx')'A$$
$$= BA' + (B' + x)A$$
$$= BA' + B'A + xA$$

$$\boxed{A(t+1) = A'B + AB' + Ax}$$

$$B(t+1) = J_B B' + K_B' B$$

$$= x' B' + (A \oplus x)' B$$

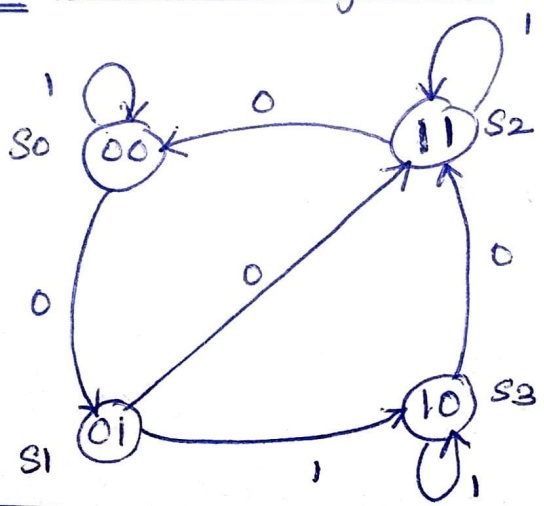
$$= B'x' + [Ax + A'x'] B$$

$$\boxed{B(t+1) = B'x' + ABx + A'Bx'}$$

Step 3: State Table.

Refer. Previous Table

Step 4: State Diagram:

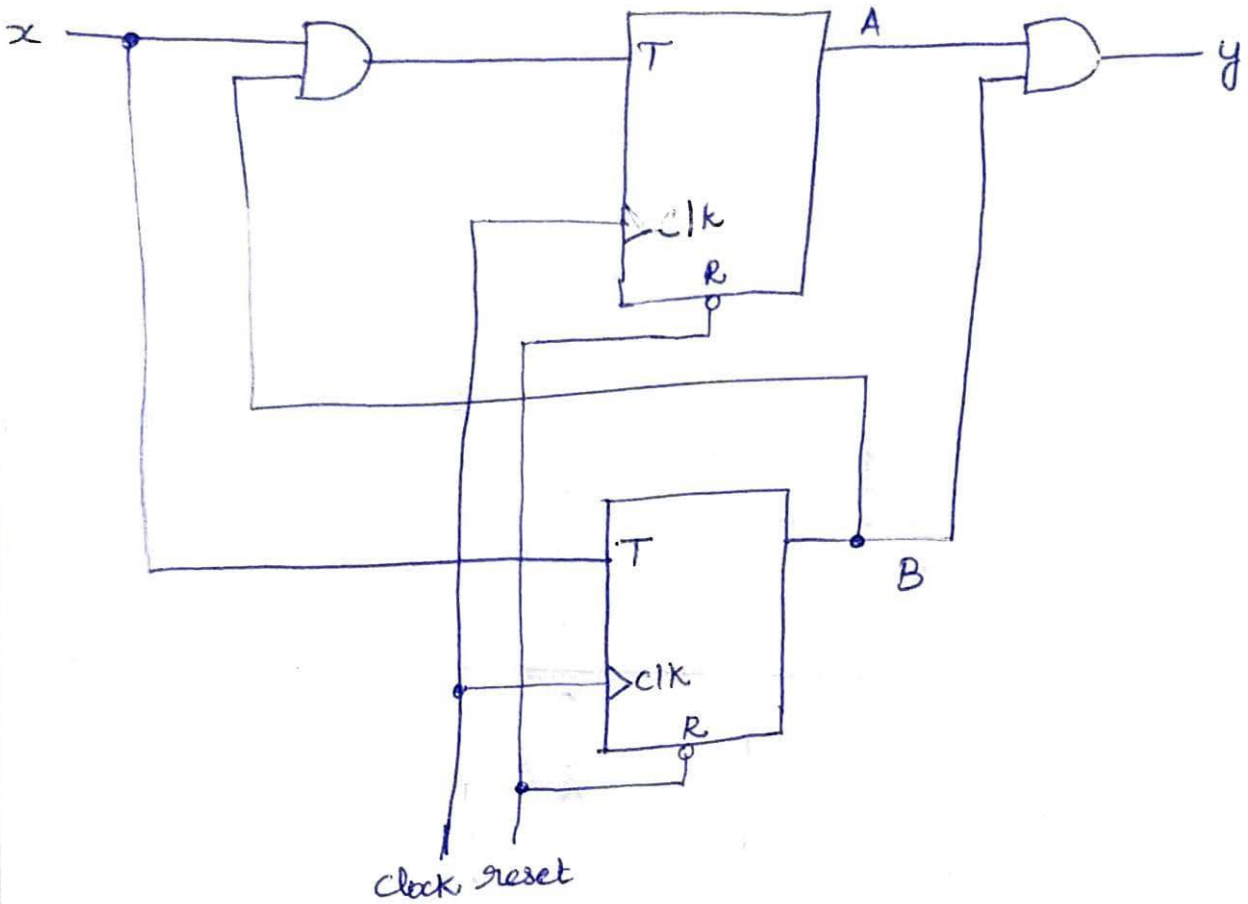


→ no outputs

c) Analysis with T Flip-flops:

* Same as JK F/F

Example:



Soln:

Two T Flip-flops → A, B
input → x
output → y

Step 1: Input equations and output equation:

$$T_A = Bx$$
$$T_B = x$$
$$y = AB$$

Step 2: State Table.

* Present state, input, next state, output
* Values for the next states can be derived from the state equations

– Substitute T_A and T_B in characteristic equation.

$Q(t+1) = T \oplus Q$
$= TQ' + T'Q$

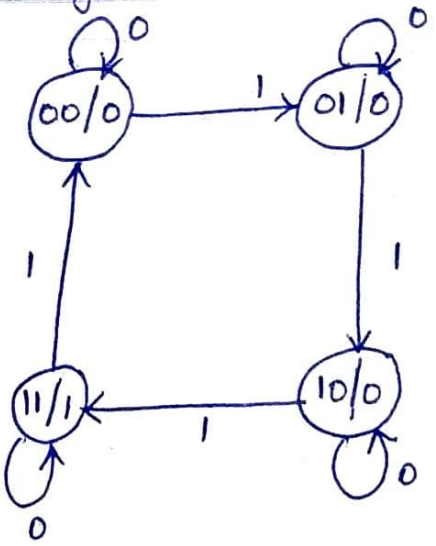
$$A(t+1) = T_A A' + T_A' A$$
$$= (Bx)A' + (Bx)'A$$
$$= A'Bx + (B'+x')A$$
$$\boxed{A(t+1) = A'Bx + AB' + Ax'}$$

$$B(t+1) = T_B \oplus B$$
$$\boxed{B(t+1) = x \oplus B}$$

State Table for sequential circuit with T flip-flops

Present state		Input	Next state		Output
A	B	x	A	B	y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	1	0	0
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	1

Step 3: State Diagram



* o/p depends on the present state only – independent of i/p
* Present state / output

Mealy and Moore Models of Finite State Machines

Two models of Sequential circuits

i) Mealy model

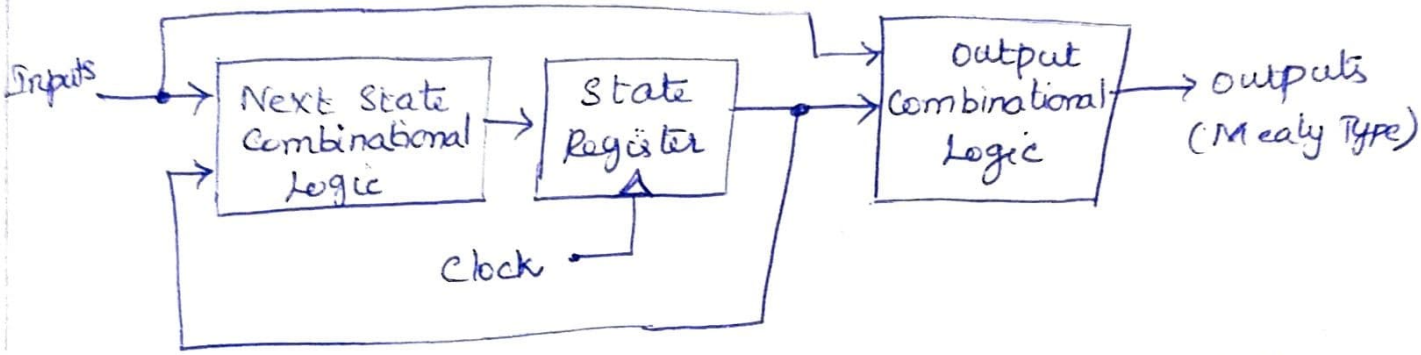
ii) Moore Model.

- They differ only in the way the output is generated.

i) Mealy Model:

* The output is a function of both the present state and the input

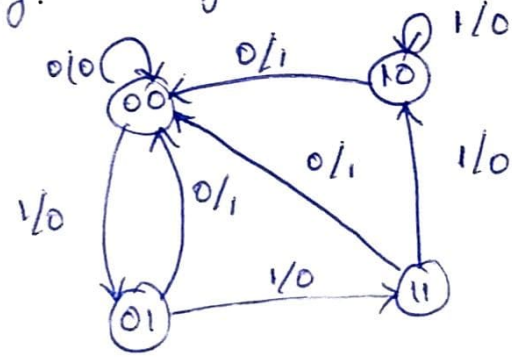
Block diagram



EX:

* output y is a function of both input x and the present state of A and B

Fig. Analysis of clocked Sequential Circuits — ISI dgm.

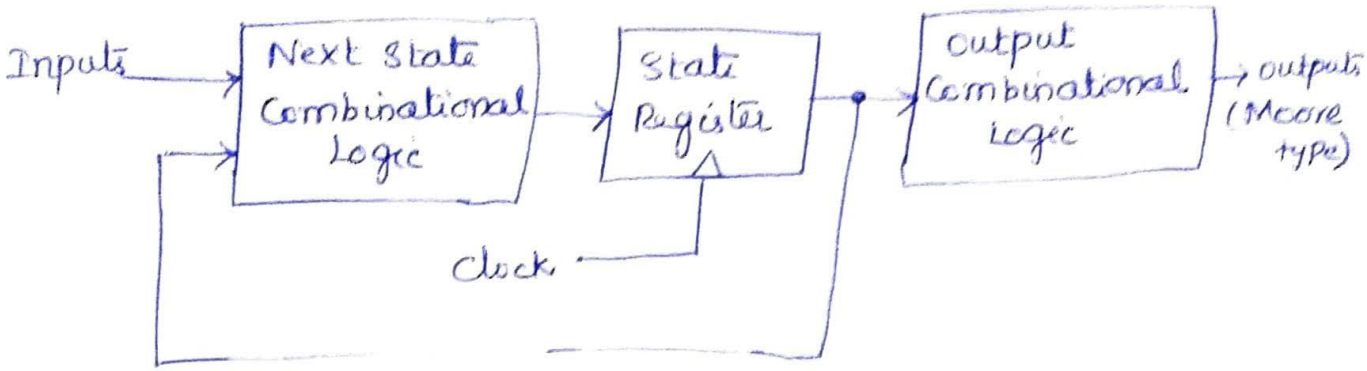


* The outputs may change if the inputs change during the clock cycle.
* The output is the value that is present immediately before the active edge of the clock.

ii) Moore Model:

* The output is a function of only the present state

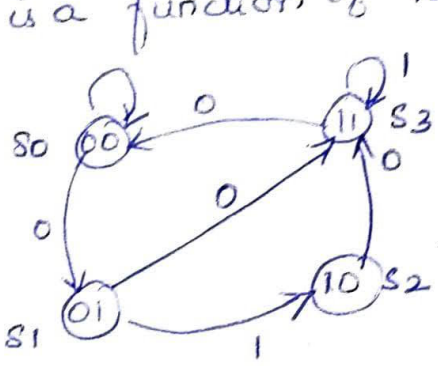
Block Diagram:

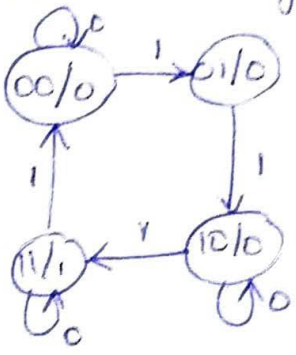


Ex:

Analysis with JK Flip-flops – circuit diagram

* output is a function of the present state only.





* The outputs of the sequential circuit are synchronized with the clock
  - because they depend only on flip-flop outputs that are synchronized with the clock.

State Reduction and Assignment

* Simplify a design by reducing the number of gates and flip-flops it uses.

→ Reducing the number of flip-flops reduces the cost of a circuit.

State Reduction

* The reduction in the number of flip-flops in a sequential circuit is referred to as the state-reduction problem.

State Reduction Algorithm

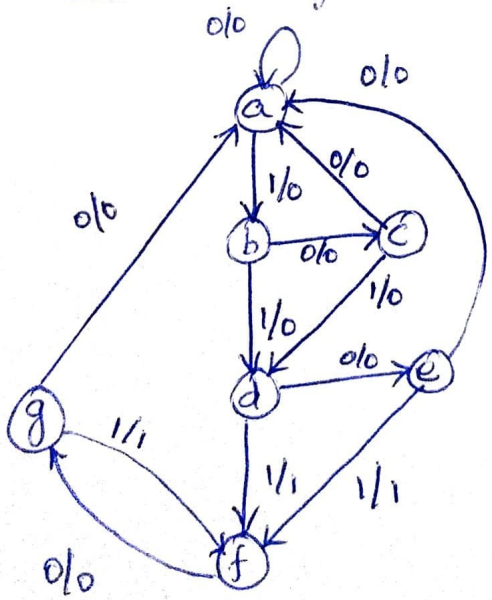
→ Procedures for reducing the number of states in a state table, while keeping the external input-output requirements unchanged.

* m flip-flops produce 2^m states

– A reduction in the number of states may result in the reduction of flip-flops

Example

State diagram



Input sequence: 01010110100

Initial state: a

Step 1: Find the complete sequence

* Each input of 0 or 1 produces an output of 0 or 1
  - causes the circuit to go to the next state

state	a	a	b	c	d	e	f	f	g	f	g	a
input	0	1	0	1	0	1	1	0	1	0	0	
output	0	0	0	0	0	1	1	0	1	0	0	

* The next state is written on the top of the next column.

Step 2: Reduce the number of states

State Table

* obtain directly from the state diagram.

Present State	Next State		Output	
	x=0	x=1	x=0	x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	g	f	0	1
g	a	f	0	1

* Two states are said to be equivalent if
  - for each member of the set of inputs, they give exactly the same output and send the circuit either to the same state or to an equivalent state.

* When two states are equivalent
  - one of them can be removed without altering input-output relationships.

i) States e and g are equivalent

* Both go to states a & f
  outputs 0 & 1

Reducing the state table.

Present State	Next State x=0	Next State x=1	Output x=0	Output x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	e	f	0	1

* Remove 'g' state and replace it with the equivalent state 'e'

ii) States f and d are equivalent

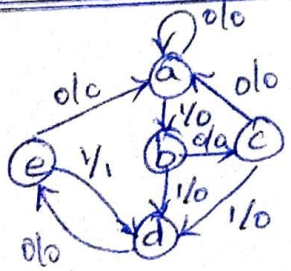
- state f can be removed and replaced by d

Reduced state Table

Present State	Next State x=0	Next State x=1	Output x=0	Output x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	d	0	1
e	a	d	0	1

* consists of only 5 states

Step 3: Reduced State diagram



Step 4: List - Input sequence

State	a	a	b	c	d	e	d	d	e	d	e	a
Input	0	1	0	1	0	1	1	0	1	0	0	
output	0	0	0	0	0	1	1	0	1	0	0	

* Same as the original

* Reduced the number of states from seven to five.

State Assignment:

* Assign unique coded binary values to the states
* For a circuit with m states, the codes must contain n bits, $2^n \geq m$.

3 bits → 8 states from 000 to 111

Step 5: Assign binary values to states.

* Use binary counting order
* Use Gray code assignment
* Use one-hot assignment

Three Possible binary State Assignments

State	Assignment 1 Binary	Assignment 2 Gray code	Assignment 3 One-hot
a	000	000	00001
b	001	001	00010
c	010	011	00100
d	011	010	01000
e	100	110	10000

* Gray code → only one bit in the code group changes when going from one number to the next.

one-hot assignment → one bit is equal to 1
— uses one flip-flop per state

* One-hot encoding leads to simpler decoding logic for the next state and output
* one-hot machines can be faster than machines with sequential binary encoding.

Reduced State Table with Binary Assignment:

Present State	Next State		Output	
	x=0	x=1	x=0	x=1
000	000	001	0	0
001	010	011	0	0
010	000	011	0	0
011	100	011	0	1
100	000	011	0	1

DESIGN PROCEDURE

- Design procedures or methodologies specify hardware that will implement a desired behavior.
- The design effort for small circuits may be manual, but industry relies on automated synthesis tools for designing massive integrated circuits.
- The sequential functionality that is to be implemented by the synthesis tool.
- Illustrate manual methods using D, JK, and T flip-flops.

✔ The design of a clocked sequential circuit starts from a set of specifications and culminates in a logic diagram or a list of Boolean functions from which the logic diagram can be obtained.

➔ The first step in the design of sequential circuits is to obtain a state table or an equivalent representation, such as a state diagram.

➔ A synchronous sequential circuit is made up of flip-flops and combinational gates.

➔ The design of the circuit consists of choosing the flip-flops and then finding a combinational gate structure that, together with the flip-flops, produces a circuit which fulfills the stated specifications.

➔ The number of flip-flops is determined from the number of states needed in the circuit and the choice of state assignment codes

➔ Once the type and number of flip-flops are determined, the design process involves a transformation from a sequential circuit problem into a combinational circuit problem.

The procedure for designing synchronous sequential circuits can be summarized by a list of recommended steps:

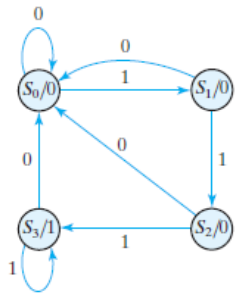
1. From the word description and specifications of the desired operation, derive a state diagram for the circuit.
2. Reduce the number of states if necessary.
3. Assign binary values to the states.
4. Obtain the binary-coded state table.
5. Choose the type of flip-flops to be used.
6. Derive the simplified flip-flop input equations and output equations.
7. Draw the logic diagram.

a) Synthesis Using D Flip-Flops

Example:

- To design a circuit that detects a sequence of three or more consecutive 1's in a string of bits coming through an input line

Step 1: State diagram for sequence detector



- Starting with state S0, the reset state.
- If the input is 0, the circuit stays in S0,
- If the input is 1, it goes to state S1 to indicate that a 1 was detected.
- If the next input is 1, the change is to state S2 to indicate the arrival of two consecutive 1's,
- If the input is 0, the state goes back to S0.
- The third consecutive 1 sends the circuit to state S3.
- If more 1's are detected, the circuit stays in S3.
- Any 0 input sends the circuit back to S0.

✔ In this way, the circuit stays in S3 as long as there are three or more consecutive 1's received.

✔ This is a Moore model sequential circuit, since the output is 1 when the circuit is in state S3 and is 0 otherwise.

Step 2:

a) Synthesis Using D Flip-Flops

- Assign binary codes to the states and list the state table.

State Table for Sequence Detector

Present State		Input	Next State		Output
A	*B*	*x*	*A*	*B*	*y*
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	0	0
1	0	0	0	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	1	1	1	1

Step 3:

- Choose two D flip-flops to represent the four states,
- Label their outputs A and B.
- one input x
- one output y.

✔ The characteristic equation of the D flip-flop is $Q(t + 1) = D_Q$

➔ the next-state values in the state table specify the D input condition for the flip-flop.

✔ The flip-flop input equations can be obtained directly from the next-state columns of A and B and expressed in sum-of-minterms form as

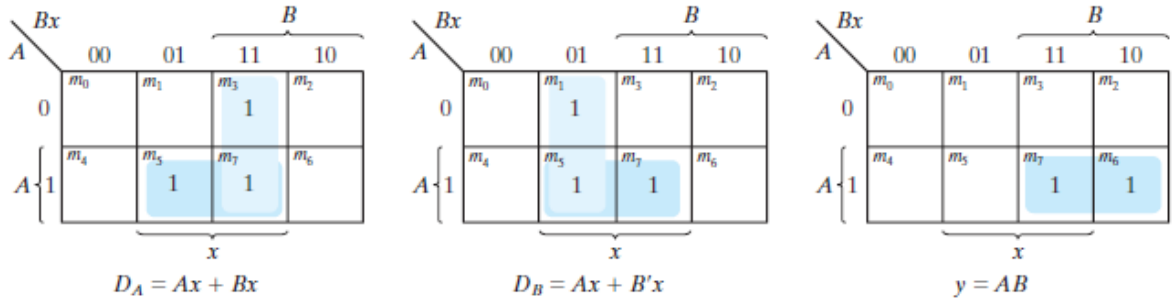
$$A(t + 1) = D_A(A, B, x) = \Sigma(3, 5, 7)$$
$$B(t + 1) = D_B(A, B, x) = \Sigma(1, 5, 7)$$
$$y(A, B, x) = \Sigma(6, 7)$$

A, B - present-state values of flip-flops A and B,
x – input
D_A, D_B -input equations.
y - output

- The minterms for output y are obtained from the output column in the state table.

Step 4:

K-Maps for sequence detector

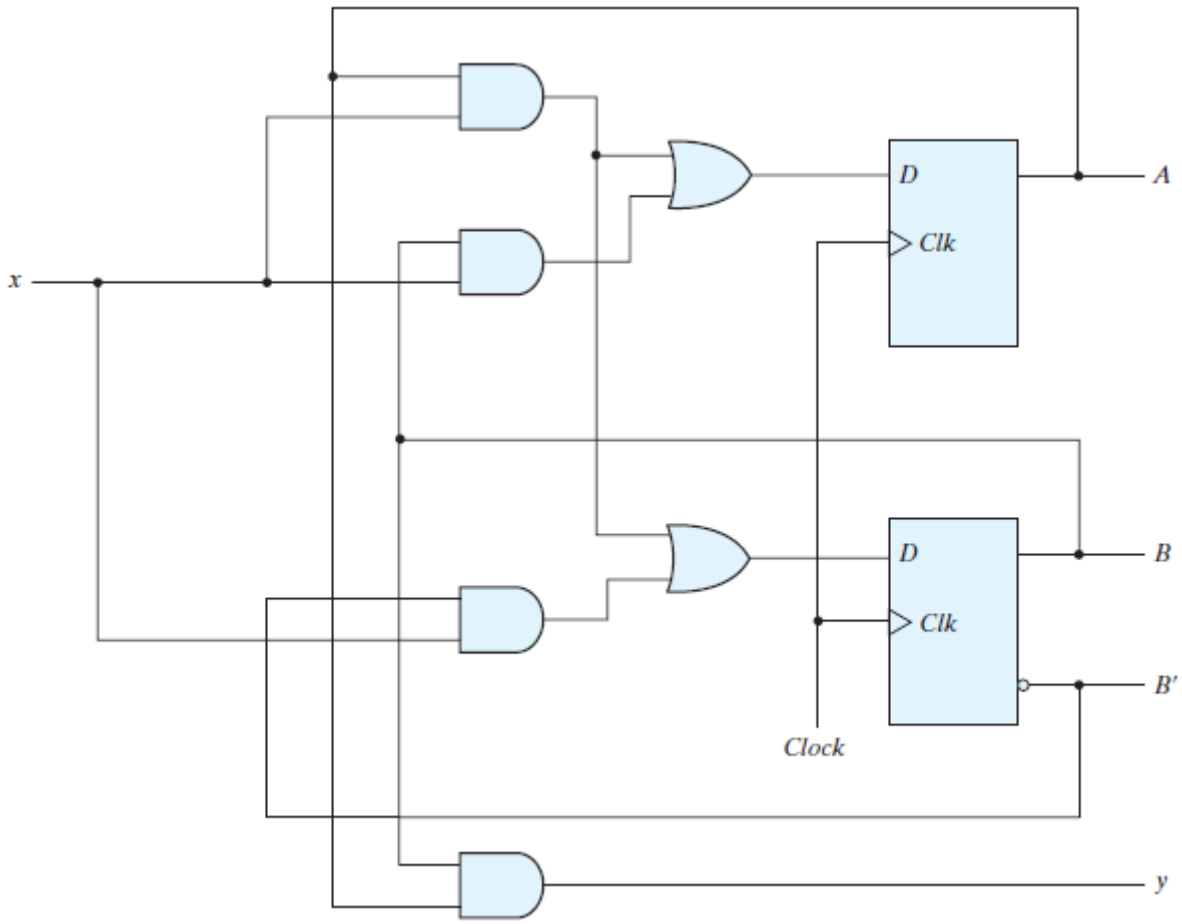


- The Boolean equations are simplified by means of the maps
- The simplified equations are

$$D_A = Ax + Bx$$
$$D_B = Ax + B'x$$
$$y = AB$$

Step 5:

Logic diagram of a Moore-type sequence detector



Excitation Tables

- D-type flip-flops
  - The input equations are obtained directly from the next state.
- JK and T types of flip-flops
  - In order to determine the input equations for these flip-flops, it is necessary to derive a functional relationship between the state table and the input equations.

Excitation Table for J-K Flip-flop:

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

- ✔ There are four possible transitions from the present state to the next state.
- ✔ The required input conditions for each of the four transitions are derived from the information available in the characteristic table.
- ✔ The symbol X in the tables represents a don't-care condition, which means that it does not matter whether the input is 1 or 0.
- ✔ When both present state and next state are 0, the J input must remain at 0 and the K input can be either 0 or 1.
- ✔ Similarly, when both present state and next state are 1, the K input must remain at 0, while the J input can be 0 or 1.
- ✔ If the flip-flop is to have a transition from the 0-state to the 1-state, J must be equal to 1, since the J input sets the flip-flop.
- ✔ Input K may be either 0 or 1. If K = 0, the J = 1 condition sets the flip-flop as required;
- ✔ If K = 1 and J = 1, the flip-flop is complemented and goes from the 0-state to the 1-state as required.
- ✔ Therefore, the K input is marked with a don't-care condition for the 0-to-1 transition.
- ✔ For a transition from the 1-state to the 0-state, we must have K = 1, since the K input clears the flip-flop. However, the J input may be either 0 or 1, since J = 0 has no effect and J = 1 together with K = 1 complements the flip-flop with a resultant transition from the 1-state to the 0-state.

Excitation table for the T flip-flop:

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

- From the characteristic table,
- when input T = 1, the state of the flip-flop is complemented, and
- when T = 0, the state of the flip-flop remains unchanged.

- ✔ when the state of the flip-flop must remain the same, the requirement is that T = 0.
- ✔ When the state of the flip-flop has to be complemented, T must equal 1.

b) Synthesis Using JK Flip-Flops

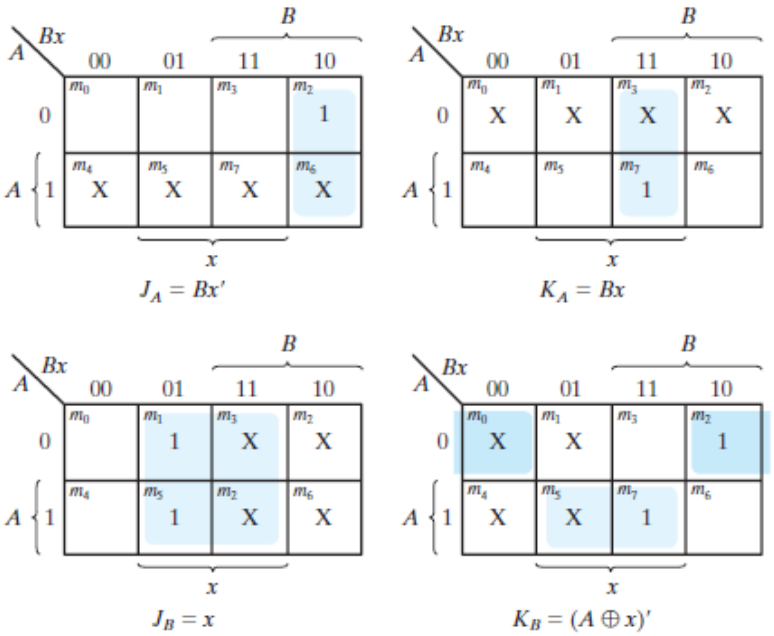
Example:

State Table and JK Flip-Flop Inputs

Present State		Input	Next State		Flip-Flop Inputs			
A	B	x	A	B	J_A	K_A	J_B	K_B
0	0	0	0	0	0	X	0	X
0	0	1	0	1	0	X	1	X
0	1	0	1	0	1	X	X	1
0	1	1	0	1	0	X	X	0
1	0	0	1	0	X	0	0	X
1	0	1	1	1	X	0	1	X
1	1	0	1	1	X	0	X	0
1	1	1	0	0	X	1	X	1

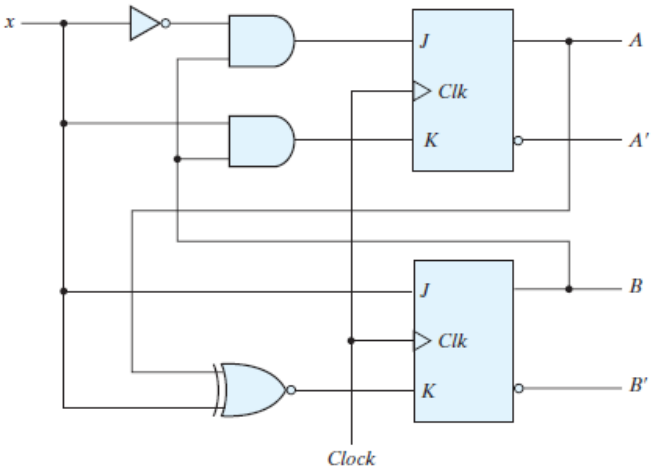
Step 1:

Maps for J and K input equations



Step 2:

Logic diagram for sequential circuit with JK flip-flops

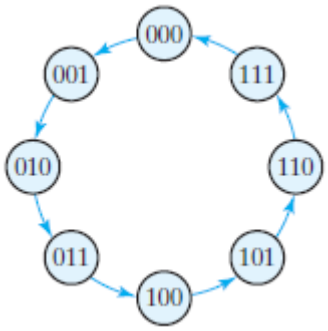


c) Synthesis Using T Flip-Flops

Example:

- Designing a binary counter.
- An n -bit binary counter consists of n flip-flops that can count in binary from 0 to $2^n - 1$.

State Diagram



- ✔ Binary states indicated inside the circles,
- ✔ The flip-flop outputs repeat the binary count sequence with a return to 000 after 111.
- ✔ The directed lines between circles are not marked with input and output values as in other state diagrams.
- ✔ State transitions in clocked sequential circuits are initiated by a clock edge;
- ✔ the flip-flops remain in their present states if no clock is applied.
- ✔ The only input to the circuit is the clock, and the outputs are specified by the present state of the flip-flops.
- ✔ The next state of a counter depends entirely on its present state, and the state transition occurs every time the clock goes through a transition.

Step 1:

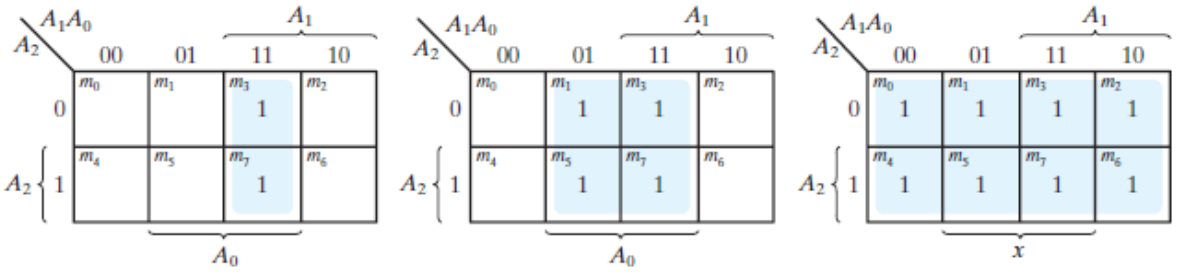
State Table for Three-Bit Counter

Present State			Next State			Flip-Flop Inputs		
A_2	A_1	A_0	A_2	A_1	A_0	T_{A2}	T_{A1}	T_{A0}
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1	1
1	1	0	1	1	1	0	1	1
1	1	1	0	0	0	1	1	1

- The three flip-flops are symbolized by A2, A1, and A0.
- Binary counters are constructed most efficiently with T flip-flops because of their complement property. The flip-

Step 2:

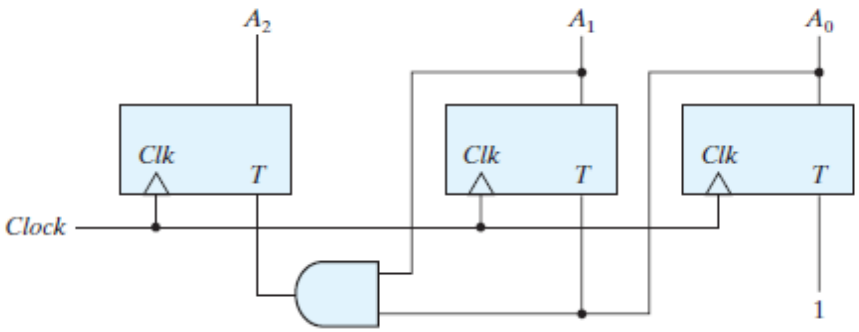
Maps for three-bit binary counter



$T_{A2} = A_1A_0$ $T_{A1} = A_0$ $T_{A0} = 1$

Step 3:

Logic diagram of three-bit binary counter



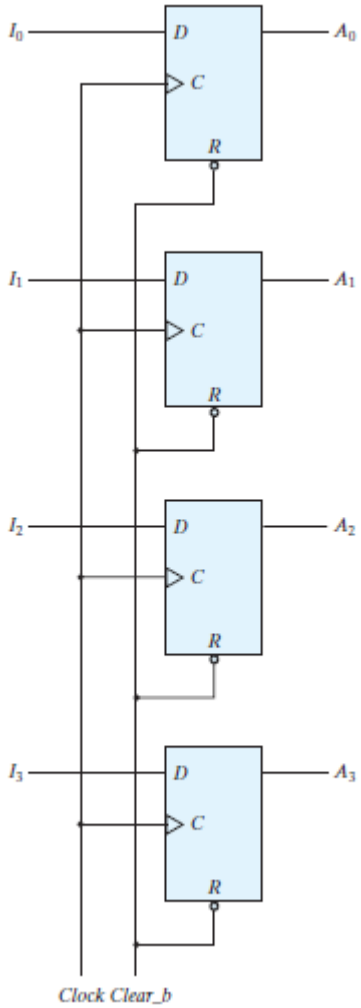
Registers and Counters

- ✔ Registers
- ✔ Shift Registers

  - A clocked sequential circuit consists of a group of flip-flops and combinational gates.
  - Circuits that include flip-flops are usually classified by the function they perform
  - Two such circuits are
    - registers and
    - counters.

- ■ A **register** is a group of flip-flops, each one of which shares a common clock and is capable of storing one bit of information.
- ■ An n-bit register consists of a group of n flip-flops capable of storing n bits of binary information.
- ■ In addition to the flip-flops, a register may have combinational gates that perform certain data-processing tasks.
- ■ The flip-flops hold the binary information, and the gates determine how the information is transferred into the register.

Four-bit register



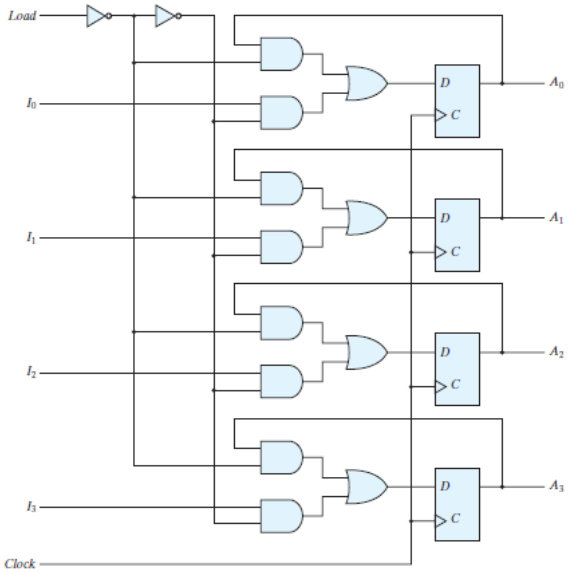
- The common clock input triggers all flip-flops on the positive edge of each pulse, and the binary data available at the four inputs are transferred into the register.
- The value of (I_3,I_2,I_1,I_0) immediately before the clock edge determines the value of (A_3, A_2 , A_1 , A_0) after the clock edge.
- The four outputs can be sampled at any time to obtain the binary information stored in the register.
- The input Clear_b goes to the active-low R (reset) input of all four flip-flops.
- When this input goes to 0, all flip-flops are reset asynchronously.
- The Clear_b input is useful for clearing the register to all 0's prior to its clocked operation.
- The R inputs must be maintained at logic 1 (i.e., de-asserted) during normal clocked operation

Register with Parallel Load

- Registers with parallel load are a fundamental building block in digital systems.
- Synchronous digital systems have a master clock generator that supplies a continuous train of clock pulses.
- The pulses are applied to all flip-flops and registers in the system.
- The master clock acts like a drum that supplies a constant beat to all parts of the system.
- A separate control signal must be used to decide which register operation will execute at each clock pulse.
- The transfer of new information into a register is referred to as loading or updating the register.
- If all the bits of the register are loaded simultaneously with a common clock pulse, then loading is done in parallel.
- A clock edge applied to the C inputs of the register, load all four inputs in parallel.
- To fully synchronize the system, ensure that all clock pulses arrive at the same time anywhere in the system, so that all flip-flops trigger simultaneously.

Four-bit register with parallel load

- A four-bit data-storage register with a load control input that is directed through gates and into the D inputs of the flip-flops



- The additional gates implement a two-channel mux whose output drives the input to the register with either the data bus or the output of the register.
- The load input to the register determines the action to be taken with each clock pulse.
  - When the load input is 1, the data at the four external inputs are transferred into the register with the next positive edge of the clock.
  - When the load input is 0, the outputs of the flip-flops are connected to their respective inputs.
- The feedback connection from output to input is necessary because a D flip-flop does not have a "no change" condition.
- With each clock edge, the D input determines the next state of the register.
- To leave the output unchanged, it is necessary to make the D input equal to the present value of the output
- The transfer of information from the data inputs or the outputs of the register is done simultaneously with all four bits in response to a clock edge.

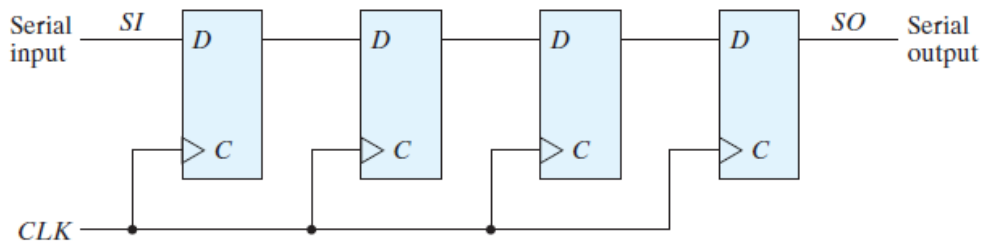
SHIFT REGISTERS

➔ A register capable of shifting the binary information held in each cell to its neighboring cell, in a selected direction, is called a shift register.

➔ The logical configuration of a shift register consists of a chain of flip-flops in cascade, with the output of one flip-flop connected to the input of the next flip-flop.

➔ All flip-flops receive common clock pulses, which activate the shift of data from one stage to the next. The simplest possible shift register is one that uses only flip-flops

Four-bit shift register



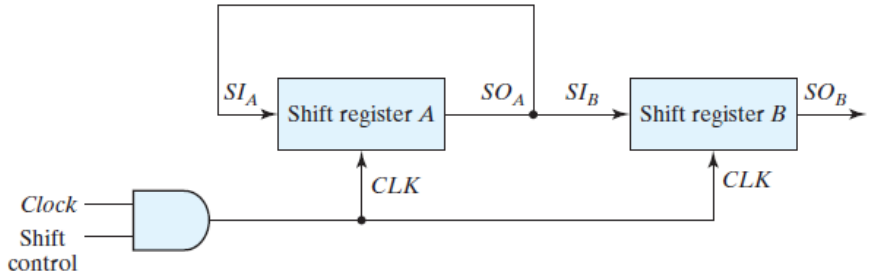
- The output of a given flip-flop is connected to the D input of the flip-flop at its right.
- This shift register is unidirectional (left-to-right).
- Each clock pulse shifts the contents of the register one bit position to the right.
- The configuration does not support a left shift.
  - The serial input determines what goes into the leftmost flip-flop during the shift.
  - The serial output is taken from the output of the rightmost flip-flop the clock's signal can be suppressed by gating the clock signal to prevent the register from shifting.

a) Serial Transfer:

- The datapath of a digital system is said to operate in serial mode when information is transferred and manipulated one bit at a time.
- Information is transferred one bit at a time by shifting the bits out of the source register and into the destination register.
- parallel transfer - all the bits of the register are transferred at the same time.

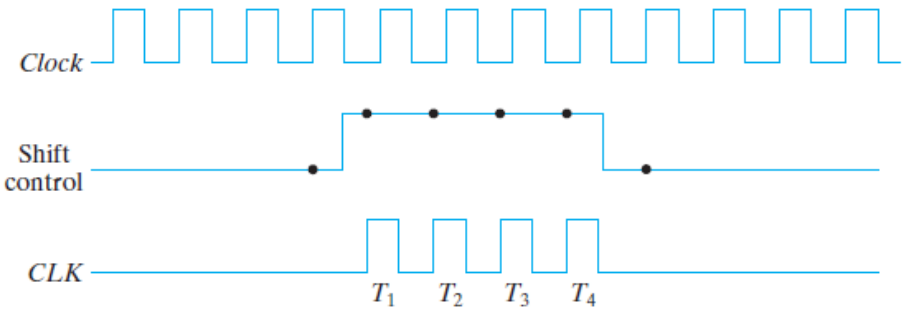
- The serial transfer of information from register A to register B is done with shift registers,

Serial transfer from register *A* to register *B*



- The serial output (SO) of register A is connected to the serial input (SI) of register B.
- To prevent the loss of information stored in the source register, the information in register A is made to circulate by connecting the serial output to its serial input.
- The initial content of register B is shifted out through its serial output and is lost unless it is transferred to a third shift register.
- The shift control input determines when and how many times the registers are shifted.
- an AND gate that allows clock pulses to pass into the CLK terminals only when the shift control is active the shift registers have four bits each.
- Then the control unit that supervises the transfer of data must be designed in such a way that it enables the shift registers, through the shift control signal, for a fixed time of four clock pulses in order to pass an entire word

Timing diagram



✔ The shift control signal is synchronized with the clock and changes value just after the negative edge of the clock.

✔ The next four clock pulses find the shift control signal in the active state, so the output of the AND gate connected to the CLK inputs produces four pulses: T_1, T_2, T_3, and T_4.

✔ Each rising edge of the pulse causes a shift in both registers.

✔ The fourth pulse changes the shift control to 0, and the shift registers are disabled.

Example:

- The binary content of A before the shift is 1011 and that of B is 0010.

◆ The serial transfer from A to B occurs in four steps

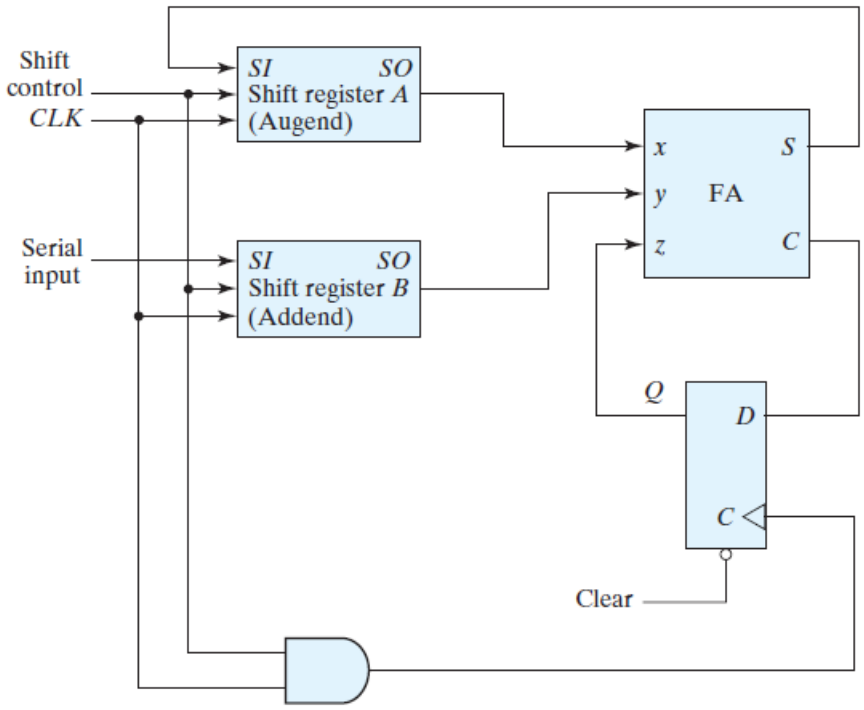
Timing Pulse	Shift Register *A*	Shift Register *B*
Initial value	1 0 1 1	0 0 1 0
After T_1	1 1 0 1	1 0 0 1
After T_2	1 1 1 0	1 1 0 0
After T_3	0 1 1 1	0 1 1 0
After T_4	1 0 1 1	1 0 1 1

➔ With the first pulse, T_1, the rightmost bit of A is shifted into the leftmost bit of B and is also circulated into the leftmost position of A.
➔ At the same time, all bits of A and B are shifted one position to the right.
➔ The previous serial output from B in the rightmost position is lost, and its value changes from 0 to 1.
➔ The next three pulses perform identical operations, shifting the bits of A into B, one at a time.
➔ After the fourth shift, the shift control goes to 0, and registers A and B both have the value 1011.
➔ The contents of A are copied into B, so that the contents of A remain unchanged i.e., the contents of A are restored to their original value.

➢ In the parallel mode, information is available from all bits of a register and all bits can be transferred simultaneously during one clock pulse.
➢ In the serial mode, the registers have a single serial input and a single serial output.
➢ The information is transferred one bit at a time while the registers are shifted in the same direction.

b) Serial Addition:

- The two binary numbers to be added serially are stored in two shift registers.
- Beginning with the least significant pair of bits, the circuit adds one pair at a time through a single full-adder (FA) circuit



- The carry out of the full adder is transferred to a D flip-flop, the output of which is then used as the carry input for the next pair of significant bits.
- The sum bit from the S output of the full adder could be transferred into a third shift register.
- By shifting the sum into A while the bits of A are shifted out, it is possible to use one register for storing both the augend and the sum bits.

- The serial input of register B can be used to transfer a new binary number while the addend bits are shifted out during the addition.

➔ <u>The operation of the serial adder is as follows:</u>

  - Initially, register A holds the augend, register B holds the addend, and the carry flip-flop is cleared to 0.
  - The outputs (SO) of A and B provide a pair of significant bits for the full adder at x and y.
  - Output Q of the flip-flop provides the input carry at z.

- The shift control enables both registers and the carry flip-flop, so at the next clock pulse, both registers are shifted once to the right, the sum bit from S enters the leftmost flip-flop of A, and the output carry is transferred into flip-flop Q.
- The shift control enables the registers for a number of clock pulses equal to the number of bits in the registers.
- For each succeeding clock pulse, a new sum bit is transferred to A, a new carry is transferred to Q, and both registers are shifted once to the right.
- This process continues until the shift control is disabled.
- The addition is accomplished by passing each pair of bits together with the previous carry through a single full-adder circuit and transferring the sum, one bit at a time, into register A.
- Initially, register A and the carry flip-flop are cleared to 0, and then the first number is added from B.
- While B is shifted through the full adder, a second number is transferred to it through its serial input.
- The second number is then added to the contents of register A , while a third number is transferred serially into register B.
- This can be repeated to perform the addition of two, three, or more four-bit numbers and accumulate their sum in register A.

✔ The parallel adder uses registers with a parallel load, whereas the serial adder uses shift registers.

✔ The number of full-adder circuits in the parallel adder is equal to the number of bits in the binary numbers, whereas the serial adder requires only one full-adder circuit and a carry flip-flop.

✔ Excluding the registers, the parallel adder is a combinational circuit, whereas the serial adder is a sequential circuit which consists of a full adder and a flip-flop that stores the output carry.

<u>The state table that specifies the sequential circuit</u>

Present State	Inputs		Next State	Output	Flip-Flop Inputs	
Q	x	y	Q	S	J_Q	K_Q
0	0	0	0	0	0	X
0	0	1	0	1	0	X
0	1	0	0	1	0	X
0	1	1	1	0	1	X
1	0	0	0	1	X	1
1	0	1	1	0	X	0
1	1	0	1	0	X	0
1	1	1	1	1	X	0

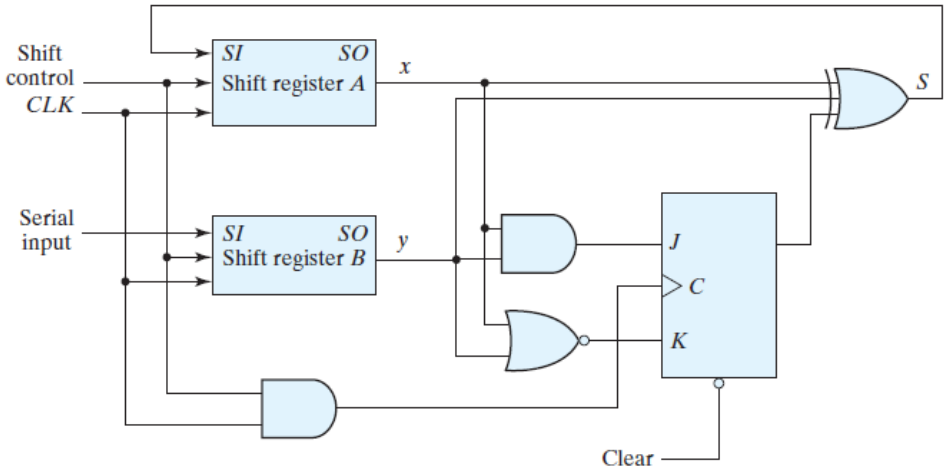
- The present state of Q is the present value of the carry.
- The present carry in Q is added together with inputs x and y to produce the sum bit in output S.
- The next state of Q is equal to the output carry.
- If a D flip-flop is used for Q, the circuit reduces to the one

➔ If a JK flipflop is used for Q, it is necessary to determine the values of inputs J and K by referring to the excitation table

➔ The two flip-flop input equations and the output equation can be simplified by means of maps to

$$J_Q = xy$$
$$K_Q = x'y' = (x + y)'$$
$$S = x \oplus y \oplus Q$$

Second form of serial adder



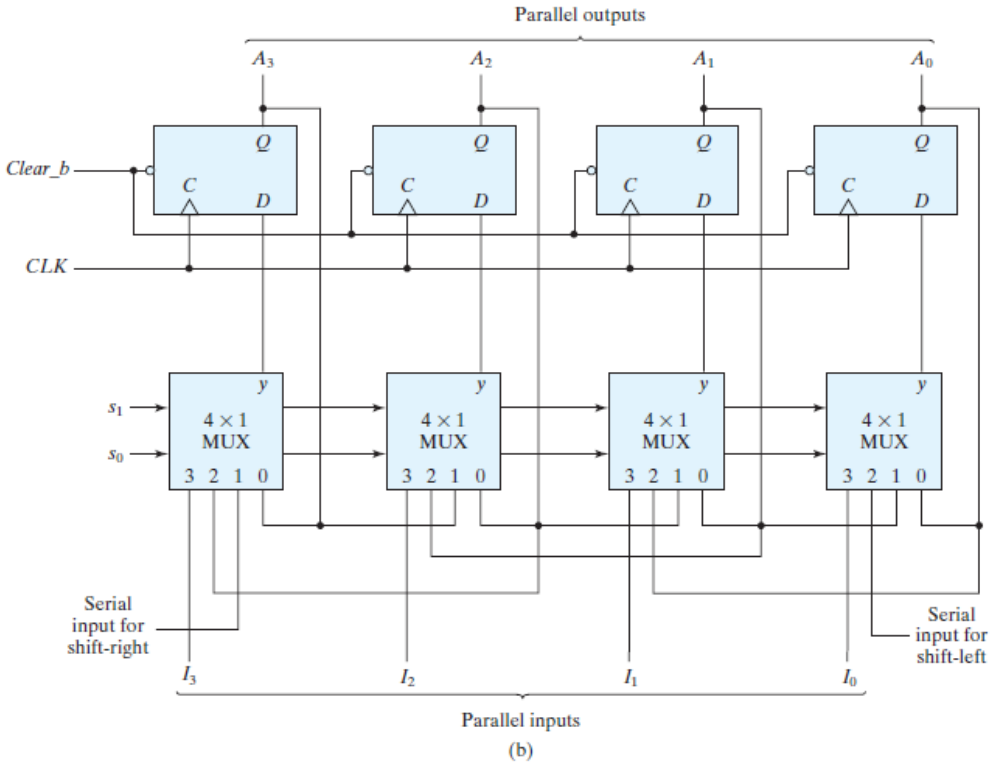
c) Universal Shift Register:

- If the flip-flop outputs of a shift register are accessible, then information entered serially by shifting can be taken out in parallel from the outputs of the flip-flops.
- If a parallel load capability is added to a shift register, then data entered in parallel can be taken out in serial fashion by shifting the data stored in the register.
- Some shift registers provide the necessary input and output terminals for parallel transfer.

- Shift register has the following capabilities:
  1. A clear control to clear the register to 0.
  2. A clock input to synchronize the operations.
  3. A shift-right control to enable the shift-right operation and the serial input and output lines associated with the shift right.
  4. A shift-left control to enable the shift-left operation and the serial input and output lines associated with the shift left.
  5. A parallel-load control to enable a parallel transfer and the n input lines associated with the parallel transfer.
  6. n parallel output lines.
  7. A control state that leaves the information in the register unchanged in response to the clock. Other shift registers may have only some of the preceding functions, with at least one shift operation.

- A register capable of shifting in one direction only is a unidirectional shift register.
- One that can shift in both directions is a bidirectional shift register.
- If the register has both shifts and parallel-load capabilities, it is referred to as a universal shift register.

The block diagram symbol and the circuit diagram of a four-bit universal shift register that has all the capabilities



(b)

Function Table for the Register

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

➔ When s1s0 = 01, terminal 1 of the multiplexer inputs has a path to the D inputs of the flip-flops. This causes a shift-right operation, with the serial input transferred into flip-flop A3.

➔ When s1s0 = 10, a shift-left operation results, with the other serial input going into flip-flop A0.

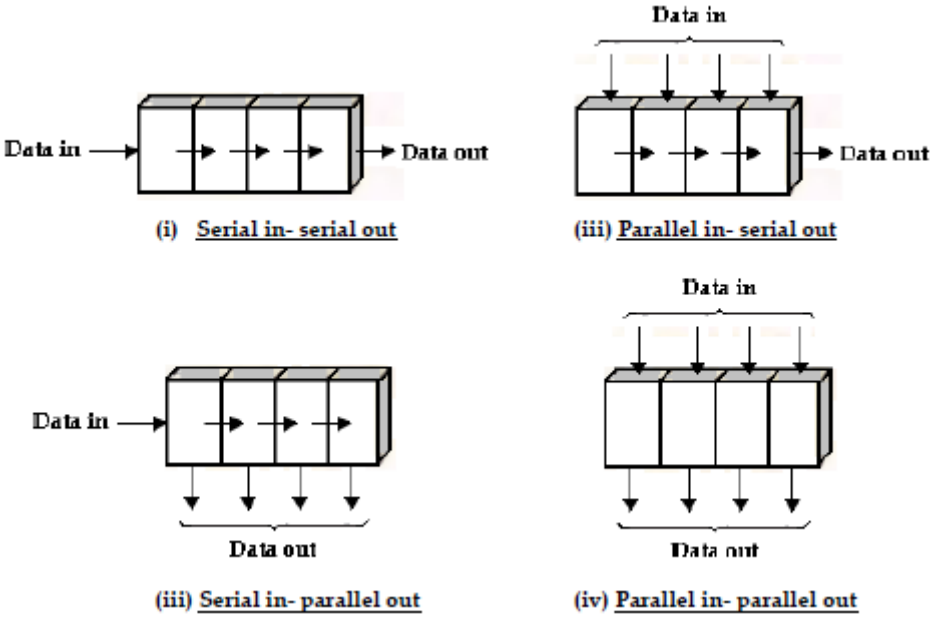
➔ Finally, when s1s0 = 11, the binary information on the parallel input lines is transferred into the register simultaneously during the next clock edge.

- It is more economical to use a single line and transmit the information serially, one bit at a time.
- The transmitter accepts the n -bit data in parallel into a shift register and then transmits the data serially along the common line.
- The receiver accepts the data serially into a shift register.
- When all n bits are received, they can be taken from the outputs of the register in parallel.

- The transmitter performs a parallel-to-serial conversion of data and the receiver does a serial-to-parallel conversion.

d) Types of Shift Register:

i. Serial in/Serial out (SISO)
ii. Serial in/Parallel out (SIPO)
iii. Parallel in/Serial out (PISO)
iv. Parallel in/Parallel out (PIPO)



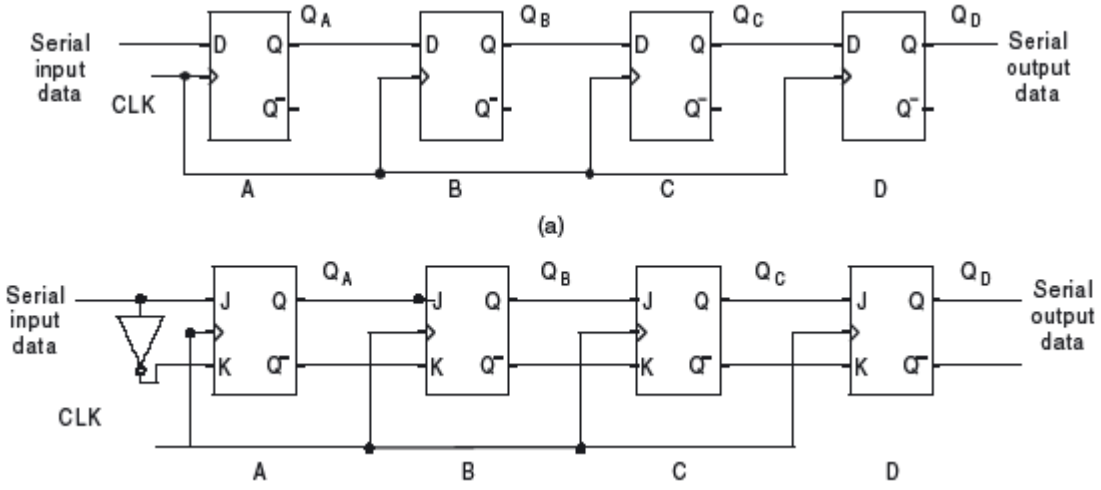
(i) Serial in- serial out (iii) Parallel in- serial out

(iii) Serial in- parallel out (iv) Parallel in- parallel out

i) Serial-In–-Serial-Out Shift Register:

Shift-right Register:

using D flip-flops, using J-K fl ip-fl ops

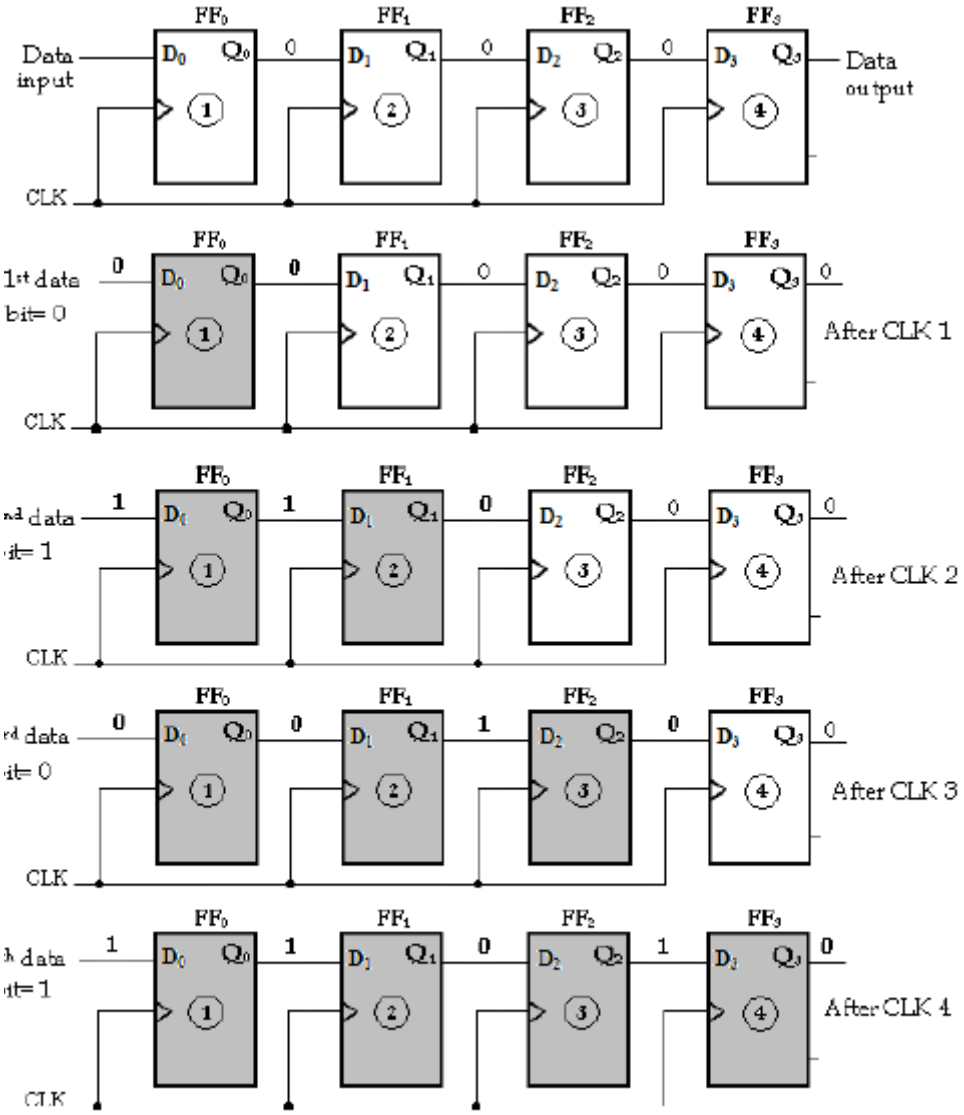


Operation of the Shift-right Register:

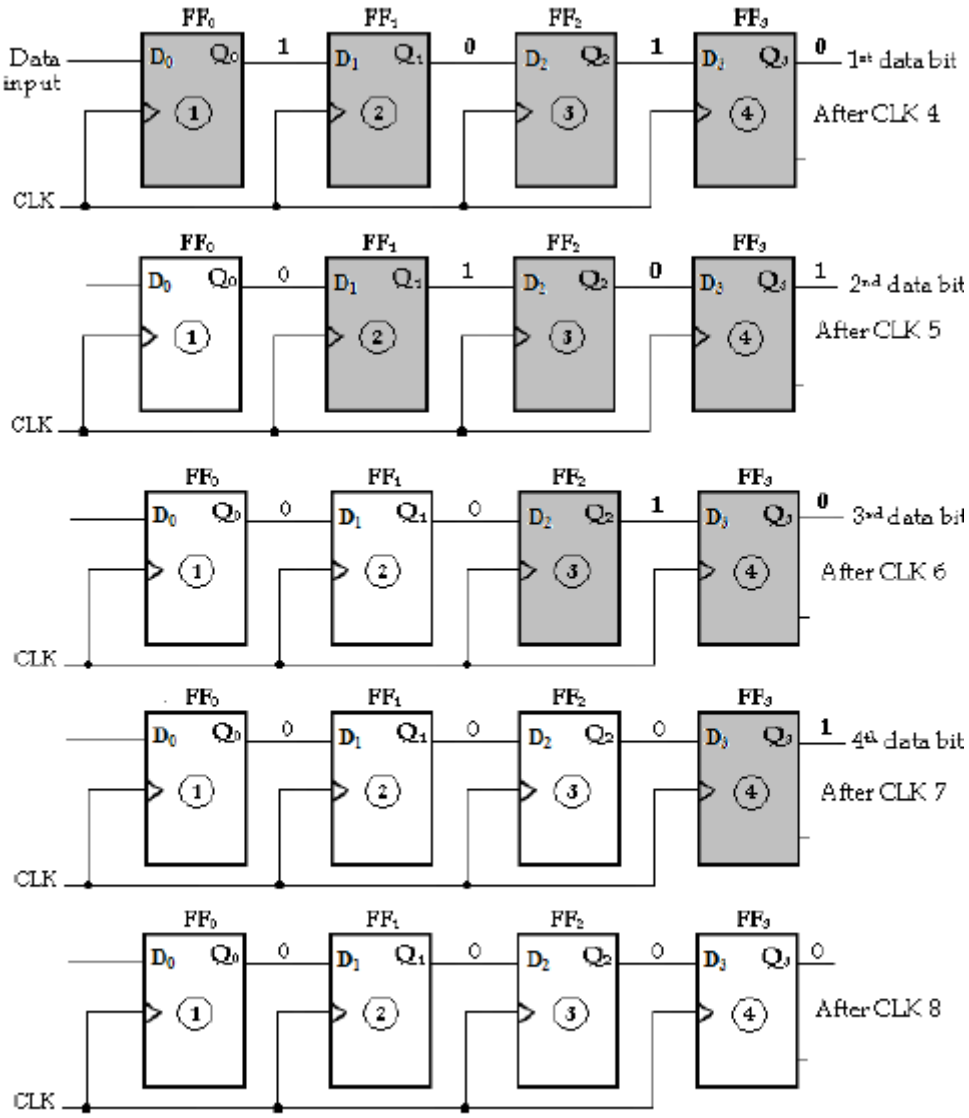
1. To shift a 1 into the flip-flop, J = 1 and K = 0,
2. To shift a 0 into the flip-flop, J = 0 and K = 1.

Timing pulse	Q_A	Q_B	Q_C	Q_D	*Serial output at* Q_D
Initial value	0	0	0	0	0
After 1st clock pulse	1	0	0	0	0
After 2nd clock pulse	1	1	0	0	0
After 3rd clock pulse	0	1	1	0	0
After 4th clock pulse	1	0	1	1	1

Example: The entry of the four bits 1010 into the register - Illustration



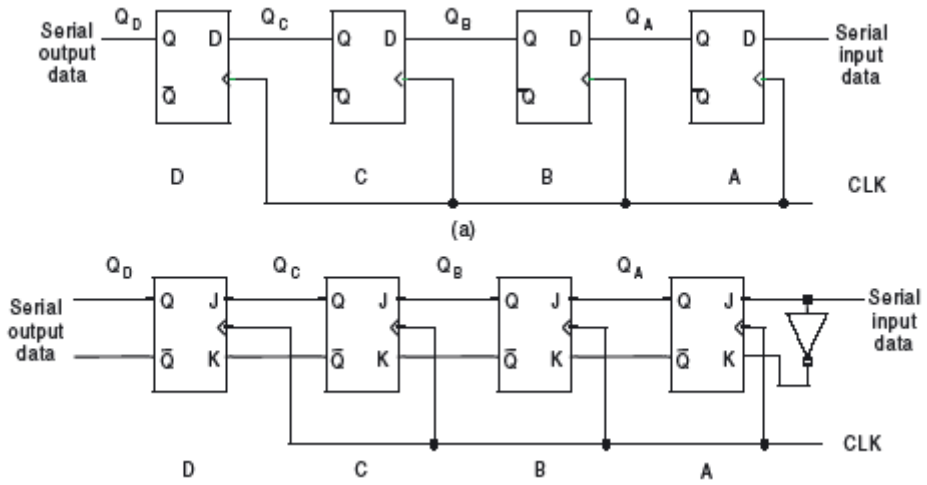
Four bits (1010) being entered serially into the register



Four bits (1010) being entered serially-shifted out of the register and replaced by all zeros

Shift-left Register

(*a*) using D flip-flops, (*b*) using J-K flip-flops

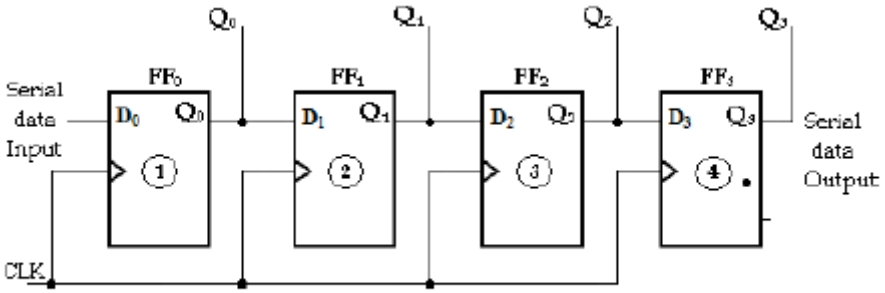


Operation of the Shift-left Register

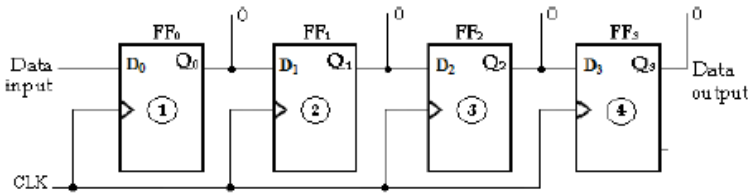
Timing pulse	Q_D	Q_C	Q_B	Q_A	*Serial output at* Q_D
Initial value	0	0	0	0	0
After 1st clock pulse	0	0	0	0	0
After 2nd clock pulse	0	0	0	1	0
After 3rd clock pulse	0	0	1	1	0
After 4th clock pulse	0	1	1	1	0

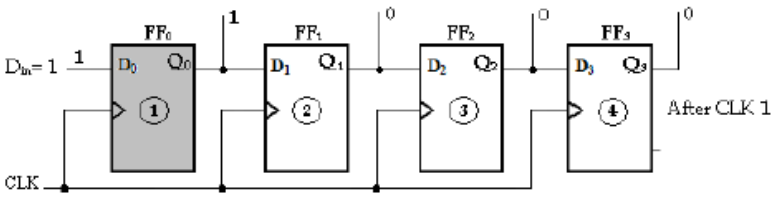
ii) Serial-In–Parallel-Out Register:

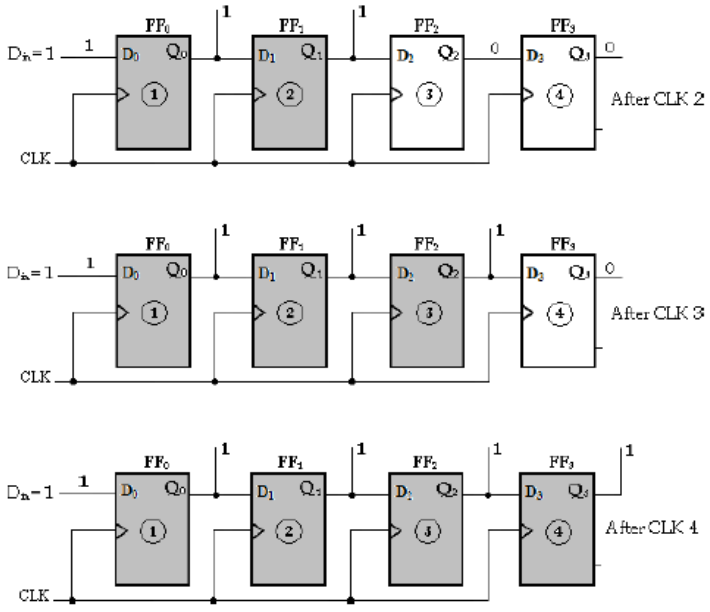
- Data bits are entered into the register in the same as serial-in serial-out shift register.
- But the output is taken in parallel.
- Once the data are stored, each bit appears on its respective output line and all bits are available simultaneously instead of on a bit-by-bit.



Serial-In parallel-Out Shift Register

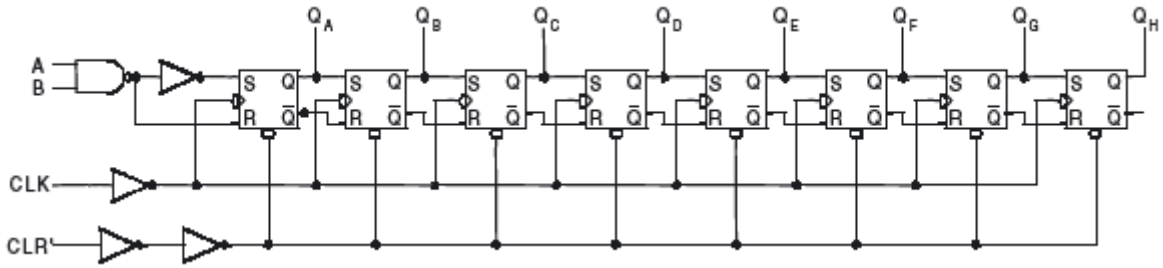






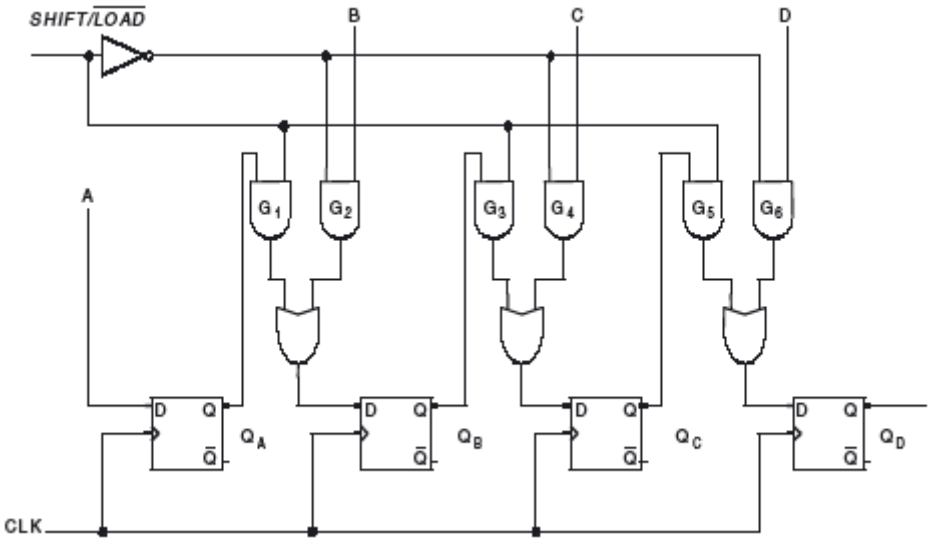
Four bits (1111) being serially entered into the register

8-bit shift register – Logic Diagram



iii) Parallel-In–Serial-Out Register:

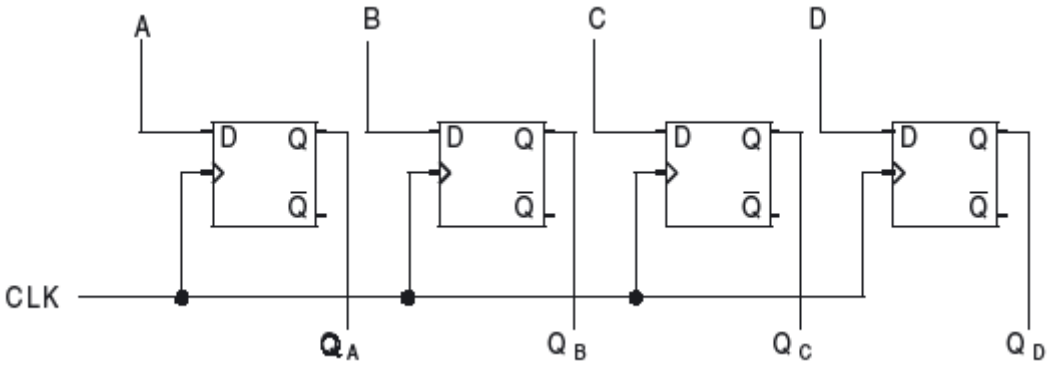
- the bits are entered in parallel i.e., simultaneously into their respective stages on parallel lines.
- There are four data input lines, X0, X1, X2 and X3 for entering data in parallel into the register.
- SHIFT/ LOAD input is the control input, which allows four bits of data to load in parallel into the register.
- When SHIFT/LOAD is LOW, gates G1, G2, G3 and G4 are enabled, allowing each data bit to be applied to the D input of its respective Flip-Flop.
- When a clock pulse 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/LOAD is HIGH, gates G1, G2, G3 and G4 are disabled and gates G5, G6 and G7 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 the parallel data-entry operation, depending on which AND gates are enabled by the level on the SHIFT/LOAD input.

iv) Parallel-In–Parallel-Out Register:

- In this type, there is simultaneous entry of all data bits and the bits appear on parallel outputs simultaneously.



Counters:

- A register that goes through a prescribed sequence of states upon the application of input pulses is called a counter.
- A counter is essentially a register that goes through a predetermined sequence of binary states.
- The input pulses may be clock pulses, or they may originate from some external source and may occur at a fixed interval of time or at random.
- The sequence of states may follow the binary number sequence or any other sequence of states.
- The gates in the counter are connected in such a way as to produce the prescribed sequence of states.

✗ Various types of registers are available commercially.
- ◆ The simplest register is one that consists of only flip-flops, without any gates.
- ◆ A counter that follows the binary number sequence is called a binary counter .
  - ➔ An n -bit binary counter consists of n flip-flops and can count in binary from 0 through 2n – 1.

Counters are available in two categories:

1. Ripple counters (Asynchronous Counters)
2. Synchronous counters.

1. Ripple counters:

- In a ripple counter, a flip-flop output transition serves as a source for triggering other flip-flops.

a) Binary Ripple Counter

- The count starts with binary 0 and increments by 1 with each count pulse input.
- After the count of 15, the counter goes back to 0 to repeat the count.

Binary Count Sequence

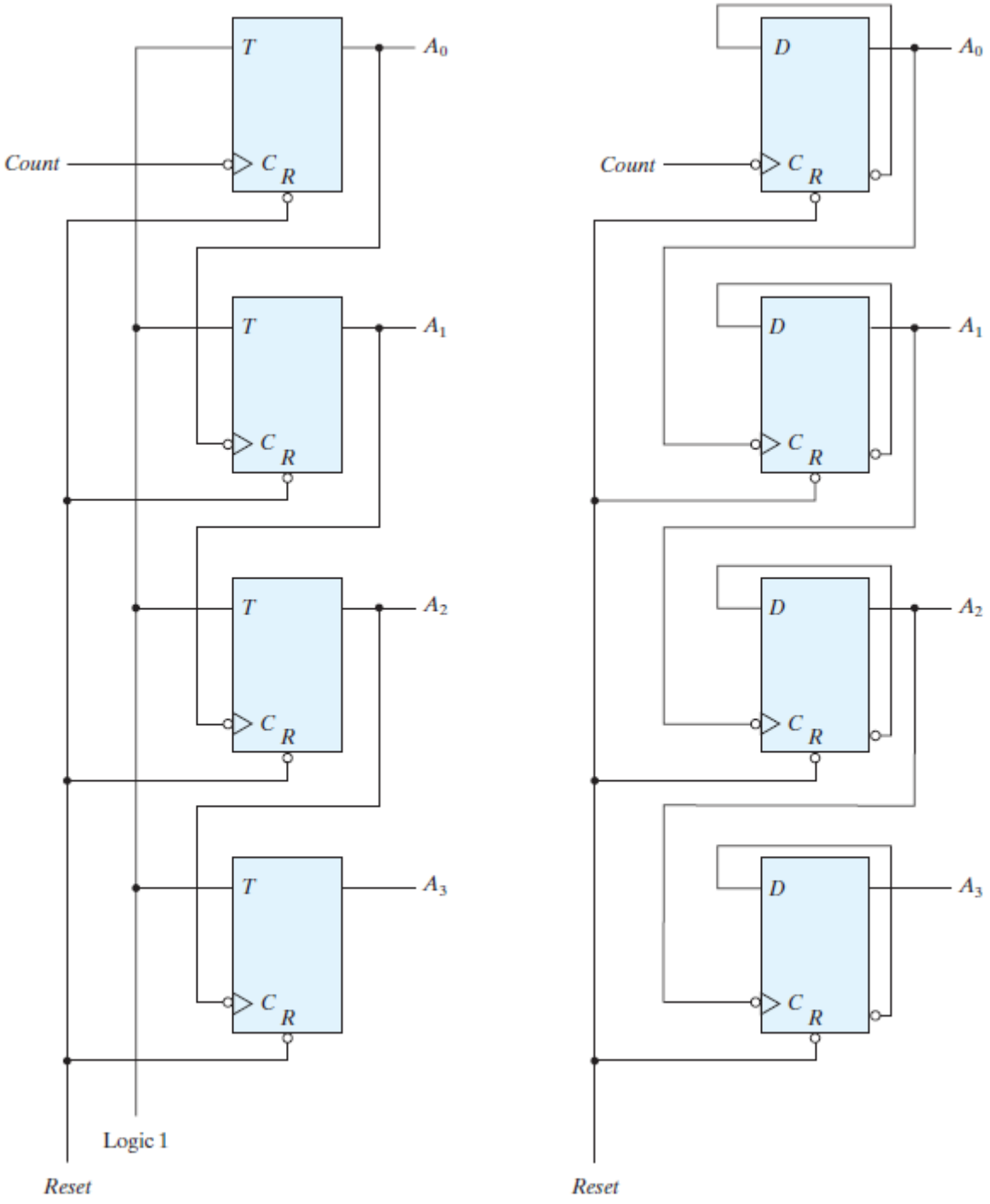
A_3	A_2	A_1	A_0
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0

✔ A binary ripple counter consists of a series connection of complementing flip-flops, with the output of each flip-flop connected to the C input of the next higher order flip-flop.

✔ The flip-flop holding the least significant bit receives the incoming count pulses

- ✔ Every time that A0 goes from 1 to 0, it complements A1.
- ✔ Every time that A1 goes from 1 to 0, it complements A2.
- ✔ Every time that A2 goes from 1 to 0, it complements A3 and so on for any other higher order bits of a ripple counter.

Four-bit binary ripple counter



(a) With *T* flip-flops

(b) With *D* flip-flops

Example: The transition from count 0011 to 0100.

- ✔ A0 is complemented with the count pulse.

- ✔ Since A0 goes from 1 to 0, it triggers A1 and complements it.
- ✔ As a result, A1 goes from 1 to 0, which in turn complements A2, changing it from 0 to 1.
- ✔ A2 does not trigger A3, because A2 produces a positive transition and the flip-flop responds only to negative transitions.
- ✔ Thus, the count from 0011 to 0100 is achieved by changing the bits one at a time, so the count goes from 0011 to 0010, then to 0000, and finally to 0100.
- ✔ The flip-flops change one at a time in succession, and the signal propagates through the counter in a ripple fashion from one stage to the next.

- The output of each flip-flop is connected to the C input of the next flip-flop in sequence.
- The flip-flop holding the least significant bit receives the incoming count pulses.
- The T inputs of all the flip-flops in (a) are connected to a permanent logic 1, making each flip-flop complement if the signal in its C input goes through a negative transition.
- The bubble in front of the dynamic indicator symbol next to C indicates that the flip-flops respond to the negative-edge transition of the input.
- The negative transition occurs when the output of the previous flip-flop to which C is connected goes from 1 to 0.

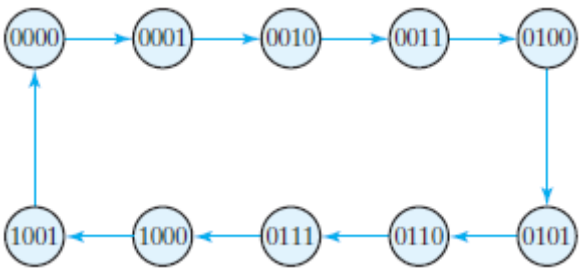
b) Binary countdown counter:

- A binary counter with a reverse count is called a binary countdown counter .
- ✔ In a countdown counter, the binary count is decremented by 1 with every input count pulse.
- ✔ The count of a four-bit countdown counter starts from binary 15 and continues to binary counts 14, 13, 12, . . . , 0 and then back to 15.
- ✔ A list of the count sequence of a binary countdown counter shows that the least significant bit is complemented with every count pulse.

c) BCD Ripple Counter:

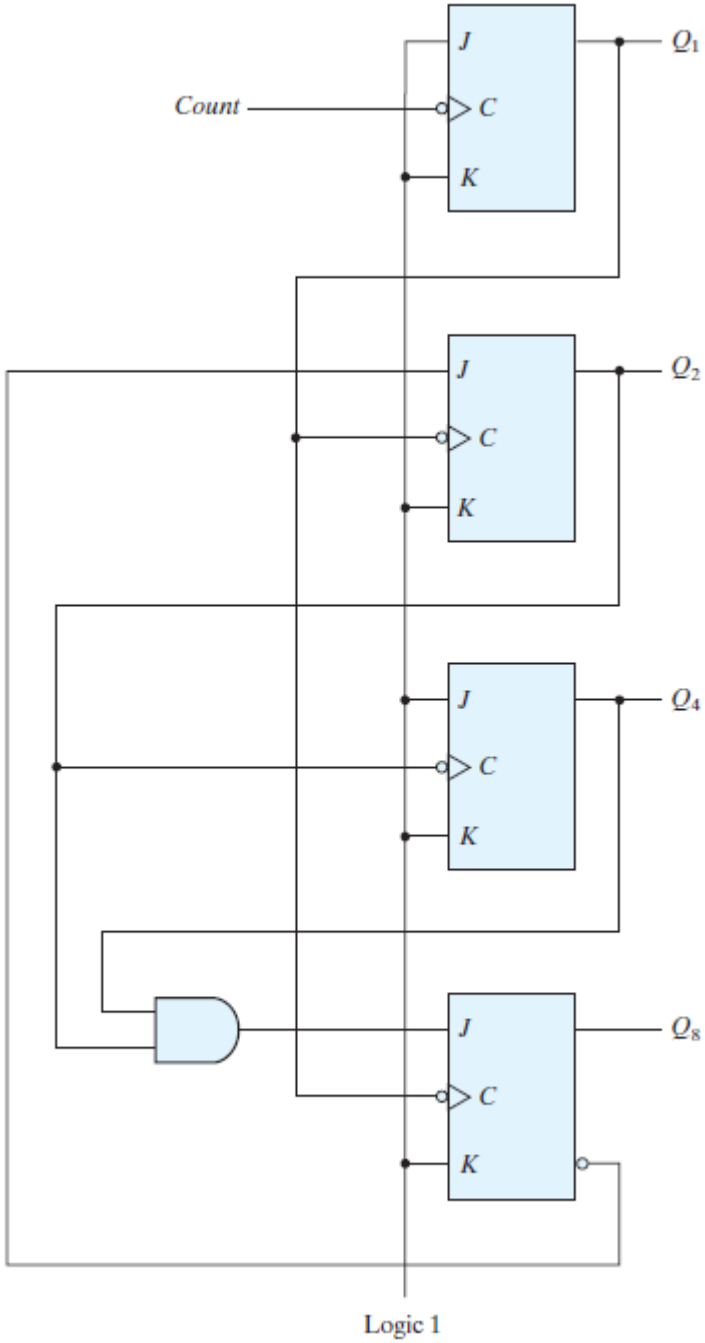
- A decimal counter follows a sequence of 10 states and returns to 0 after the count of 9.
- A counter must have at least four flip-flops to represent each decimal digit, since a decimal digit is represented by a binary code with at least four bits.
- The sequence of states in a decimal counter is dictated by the binary code used to represent a decimal digit.

State diagram of a decimal BCD counter



- A decimal counter is similar to a binary counter, except that the state after 1001 (the code for decimal digit 9) is 0000 (the code for decimal digit 0).

The logic diagram of a BCD ripple counter using JK flip-flops

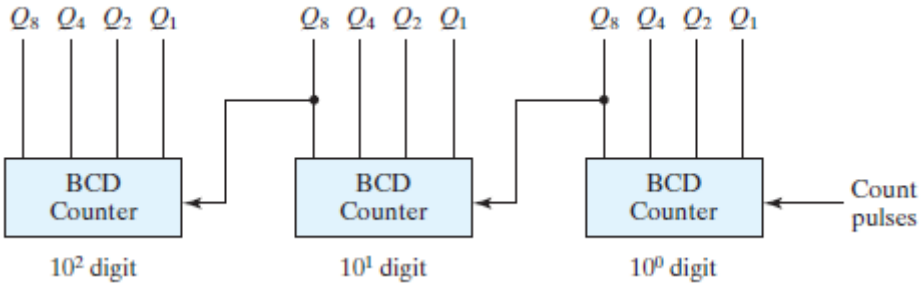


- The four outputs are designated by the letter symbol Q, with a numeric subscript equal to the binary weight of the corresponding bit in the BCD code.
- The output of Q1 is applied to the C inputs of both Q2 and Q8 and the output of Q2 is applied to the C input of Q4.
- The J and K inputs are connected either to a permanent 1 signal or to outputs of other flip-flops.
- Signals that affect the flip-flop transition depend on the way they change from 1 to 0.
- The operation of the counter can when the C input goes from 1 to 0, the flip-flop is
  - set if J = 1
  - cleared if K = 1,
  - complemented if J = K = 1, and
  - left unchanged if J = K = 0.

d) Decade counter:

- The BCD counter is a decade counter

Block diagram of a three-decade decimal BCD counter



- Multiple decade counters can be constructed by connecting BCD counters in cascade, one for each decade.
- The inputs to the second and third decades come from Q8 of the previous decade.
- When Q8 in one decade goes from 1 to 0, it triggers the count for the next higher order decade while its own decade goes from 9 to 0.

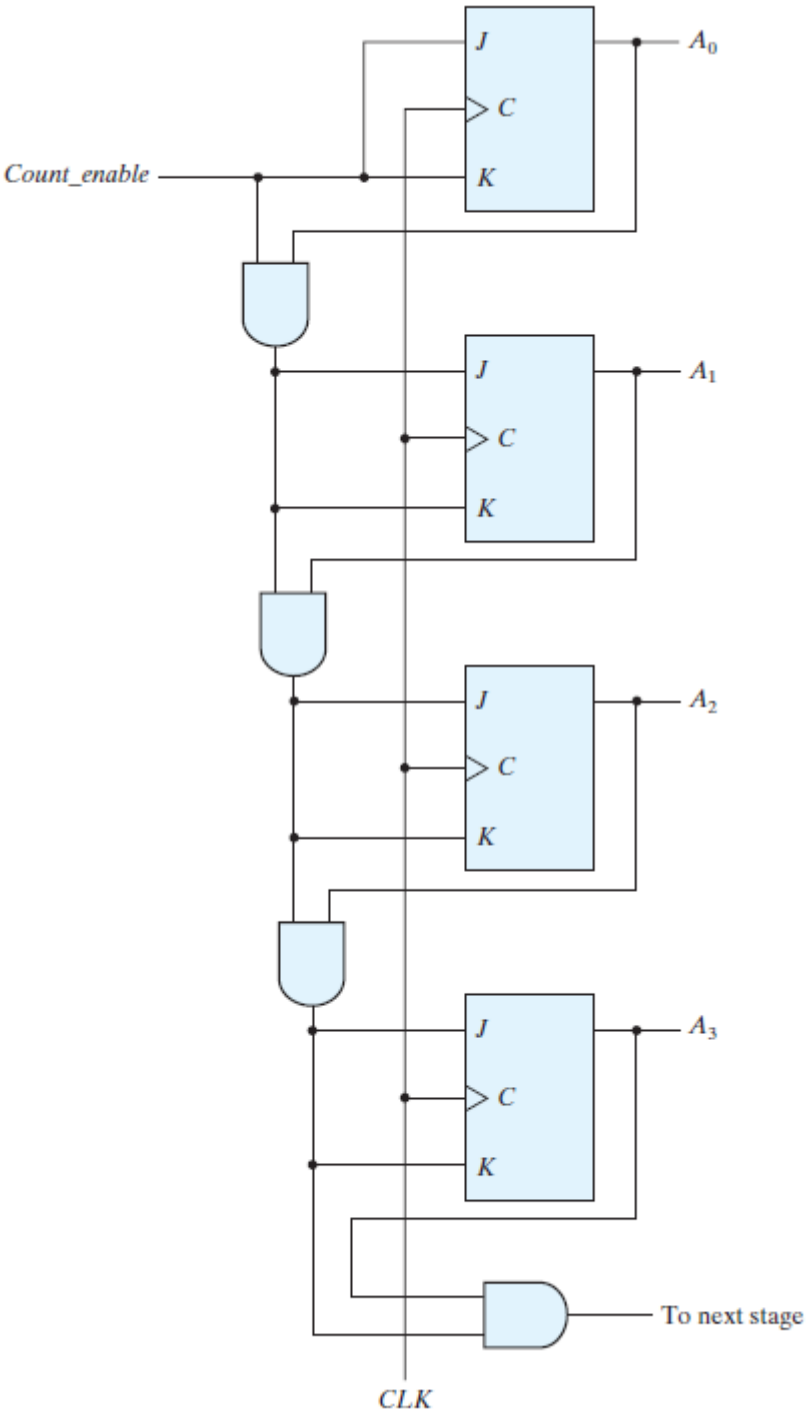
2. Synchronous Counters

- Clock pulses are applied to the inputs of all flip-flops.
- A common clock triggers all flip-flops simultaneously

a) Binary Counter:

- the flip-flop in the least significant position is complemented with every pulse.
- A flip-flop in any other position is complemented when all the bits in the lower significant positions are equal to 1 .

Four-bit synchronous binary counter



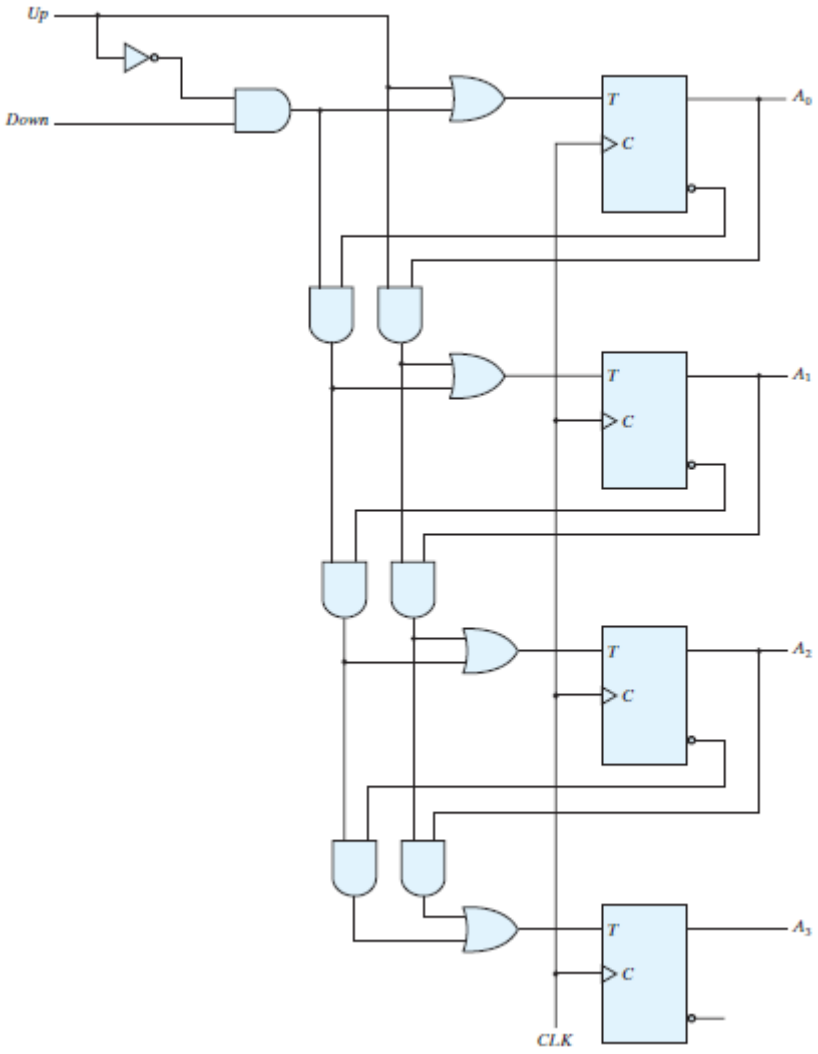
- The C inputs of all flip-flops are connected to a common clock.
- The counter is enabled by Count_enable.
- ✔ If the enable input is 0, all J and K inputs are equal to 0 and the clock does not change the state of the counter.

- ✗ The first stage, A0, has its J and K equal to 1 if the counter is enabled.
- ✗ The other J and K inputs are equal to 1 if all previous least significant stages are equal to 1 and the count is enabled.
- ✗ The chain of AND gates generates the required logic for the J and K inputs in each stage.

- The counter can be extended to any number of stages, with each stage having an additional flip-flop and an AND gate that gives an output of 1 if all previous flip-flop outputs are 1.
- The flip-flops trigger on the positive edge of the clock.

b) Up–Down Binary Counter:

- A synchronous countdown binary counter goes through the binary states in reverse order, from 1111 down to 0000 and back to 1111 to repeat the count.
- The bit in the least significant position is complemented with each pulse.
- A bit in any other position is complemented if all lower significant bits are equal to 0.



- It has an up control input and a down control input.
  - When the up input is 1,
    - the circuit counts up, since the T inputs receive their signals from the values of the previous normal outputs of the flip-flops.
  - When the down input is 1 and the up input is 0,
    - the circuit counts down, since the complemented outputs of the previous flip-flops are applied to the T inputs
  - When the up and down inputs are both 0,
    - the circuit does not change state and remains in the same count.
  - When the up and down inputs are both 1,
    - the circuit counts up.
- This set of conditions ensures that only one operation is performed at any given time.
- The up input has priority over the down input.

c) BCD Counter

- A BCD counter counts in binary-coded decimal from 0000 to 1001 and back to 0000.

State table of a 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

- The input conditions for the T flip-flops are obtained from the present- and next-state conditions
- The flip-flop input equations can be simplified by means of maps.
- The unused states for minterms 10 to 15 are taken as don't-care terms.
- The simplified functions are

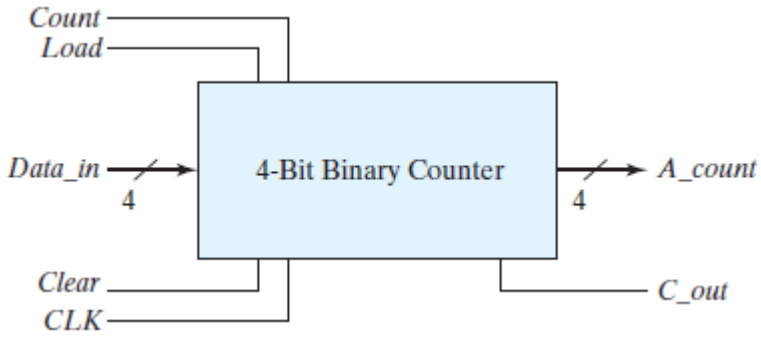
$$T_{Q1} = 1$$
$$T_{Q2} = Q_8' Q_1$$
$$T_{Q4} = Q_2 Q_1$$
$$T_{Q8} = Q_8 Q_1 + Q_4 Q_2 Q_1$$
$$y = Q_8 Q_1$$

- The circuit can easily be drawn with four T flip-flops, five AND gates, and one OR gate.
- Synchronous BCD counters can be cascaded to form a counter for decimal numbers of any length.

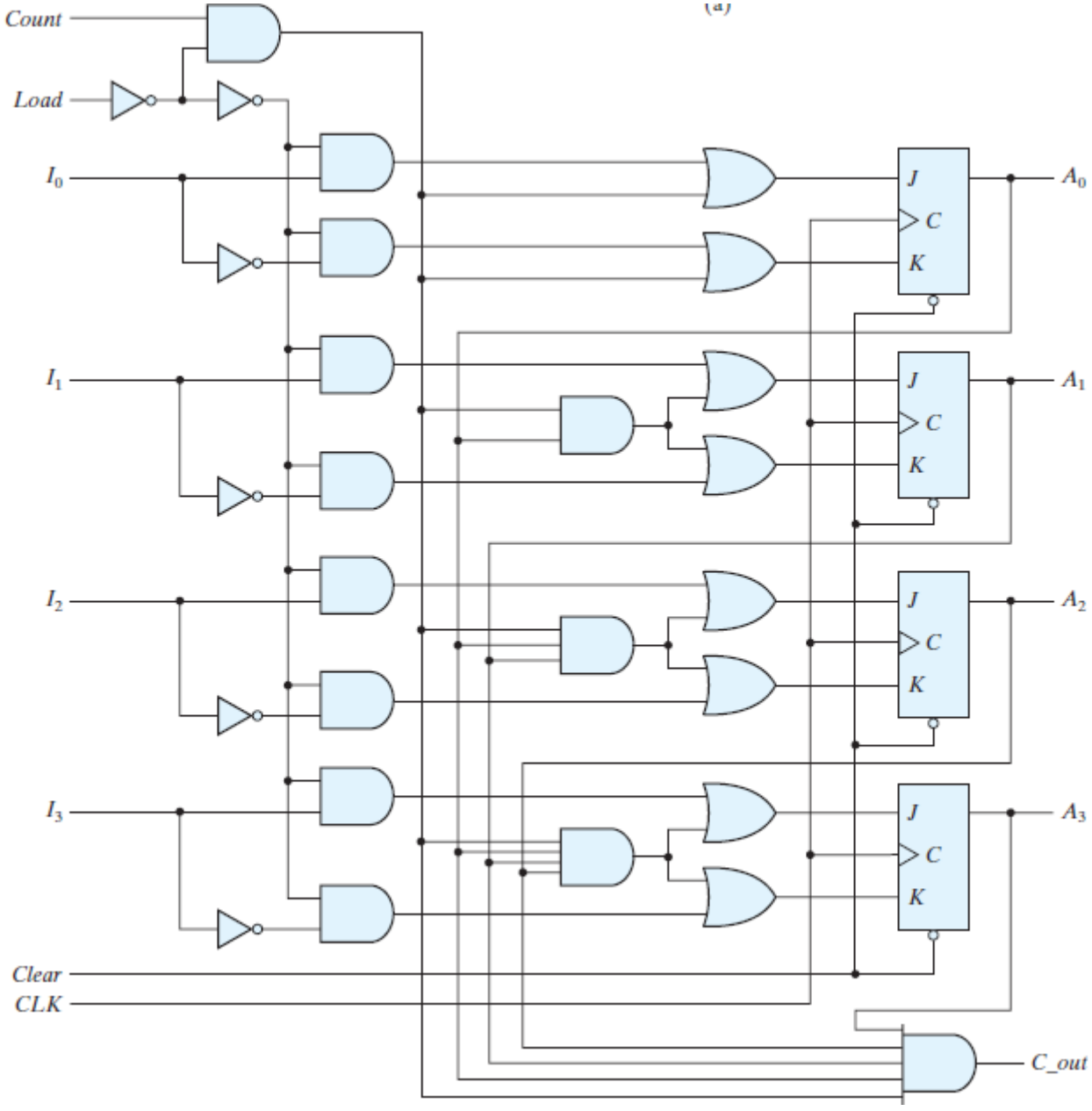
d) Binary Counter with Parallel Load:

- Counters employed in digital systems quite often require a parallel-load capability for transferring an initial binary number into the counter prior to the count operation.

Top-level block diagram symbol



The logic diagram of a four-bit register



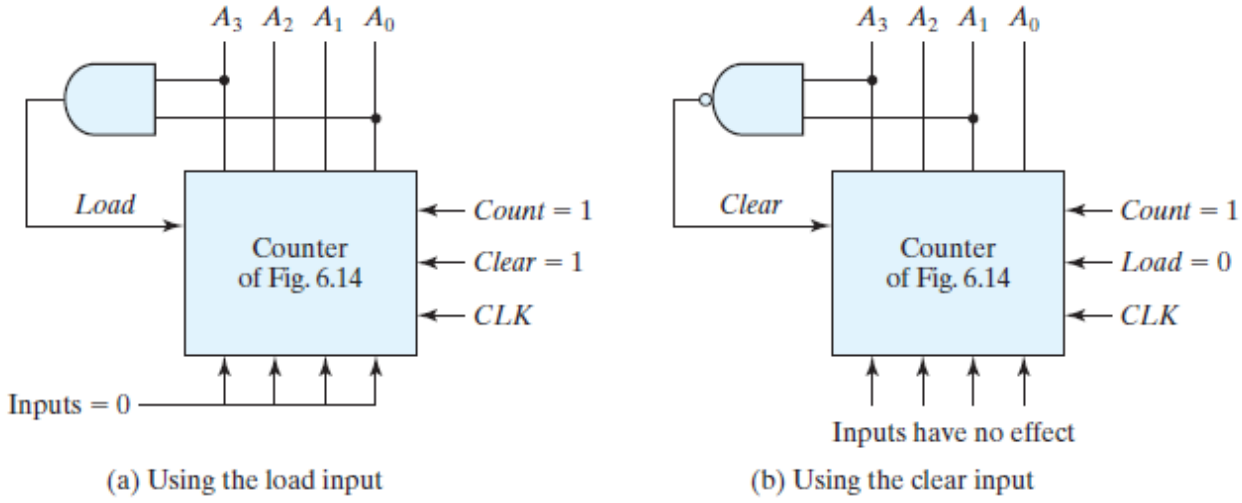
- When equal to 1, the input load control disables the count operation and causes a transfer of data from the four data inputs into the four flip-flops.
- If both control inputs are 0, clock pulses do not change the state of the register.
- The carry output becomes a 1 if all the flip-flops are equal to 1 while the count input is enabled.
- This is the condition for complementing the flip-flop that holds the next significant bit.
- The carry output is useful for expanding the counter to more than four bits

Function Table for the Counter

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

- ✔ The four control inputs— Clear, CLK, Load, and Count —determine the next state.
  - ✗ The Clear input is asynchronous and, when equal to 0, causes the counter to be cleared regardless of the presence of clock pulses or other inputs.
    - ✔ indicated in the table by the X entries, which symbolize don't-care conditions for the other inputs.
  - ✗ The Clear input must be in the 1 state for all other operations.
  - ✗ With the Load and Count inputs both at 0, the outputs do not change, even when clock pulses are applied.
  - ✗ A Load input of 1 causes a transfer from inputs I0 - I3 into the register during a positive edge of CLK .
- ✔ The input data are loaded into the register regardless of the value of the Count input, because the Count input is inhibited when the Load input is enabled.
- ✔ The Load input must be 0 for the Count input to control the operation of the counter.
- ✔ A counter with a parallel load can be used to generate any desired count sequence.

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



(a) Using the load input

(b) Using the clear input

OTHER COUNTERS

- Counters can be designed to generate any desired sequence of states.
- Counters are used to generate timing signals to control the sequence of operations in a digital system.
- Counters can also be constructed by means of shift registers

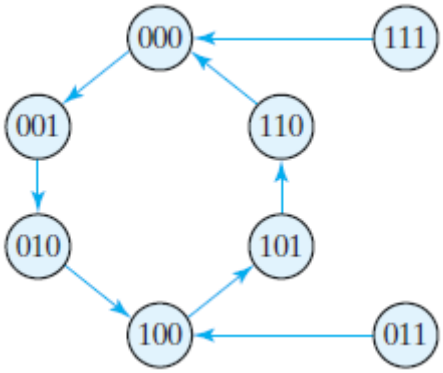
a) Divide-by- N counter (Modulo- N counter)

- A divide-by- N counter (also known as a modulo- N counter) is a counter that goes through a repeated sequence of N states.
- The sequence may follow the binary count or may be any other arbitrary sequence

Counter with Unused States

- A circuit with n flip-flops has 2n binary states

Step 1: State Transition Diagram



Step 2:

State Table for Counter

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

Step 3:

- Flip-flop input equations can be simplified by using minterms 3 and 7 as don't-care conditions.
- Use K-map

J_A:

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

$J_A = B$

K_A:

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

$K_A = B$

J_B:

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

$J_B = C$

K_B:

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

$K_B = 1$

J_C:

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

$J_C = B'$

K_C:

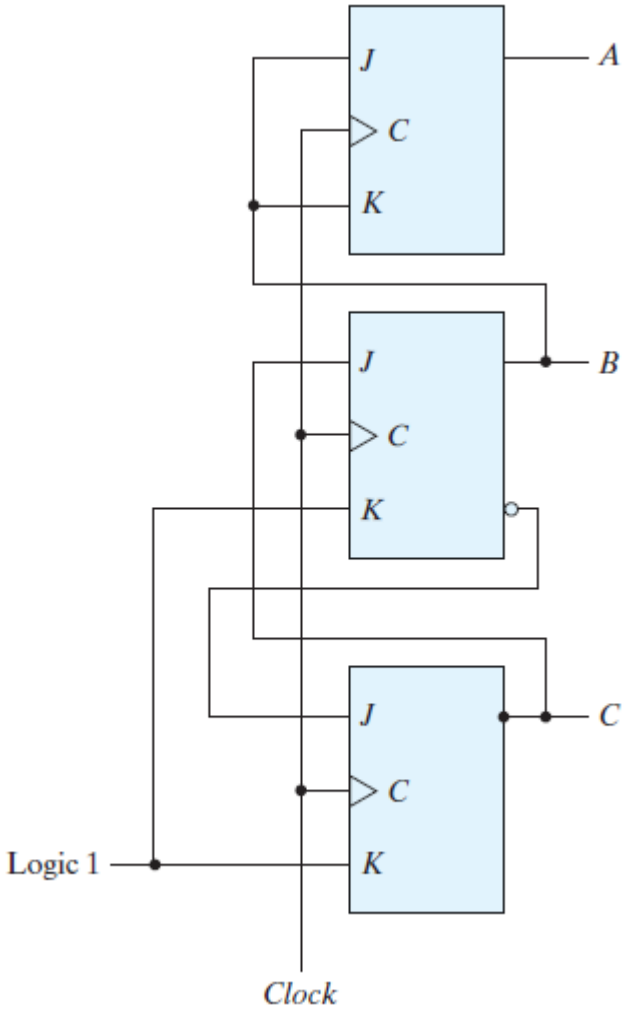
A \ BC	00	01	11	10
0	X	1	X	X
1	X	1	X	X

$K_C = 1$

The simplified equations are

$J_A = B$ $K_A = B$
$J_B = C$ $K_B = 1$
$J_C = B$ $K_C = 1$

Step 4: Logic Circuit Diagram

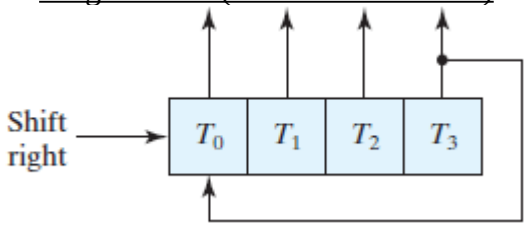


b) Ring Counter:

- A ring counter is a circular shift register with only one flip-flop being set at any particular time; all others are cleared.
- The single bit is shifted from one flip-flop to the next to produce the sequence of timing signals.

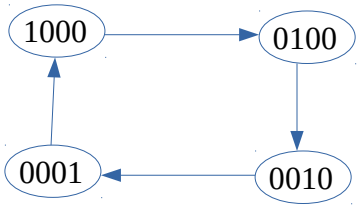
four-bit shift register connected as a 8-4-2-1 ring counter

Ring Counter (Initial value = 1000)



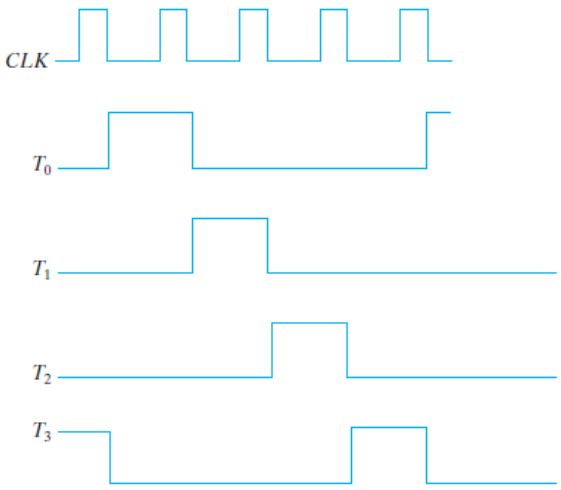
- The initial value of the register is 1000 and requires Preset/Clear flip-flops.
- The single bit is shifted right with every clock pulse and circulates back from T3 to T0.

State Diagram



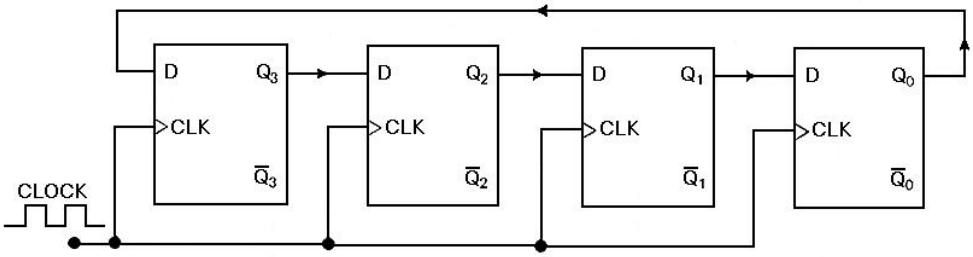
- Each flip-flop is in the 1 state once every four clock cycles and produces one of the four timing signals

Sequence of four timing signals



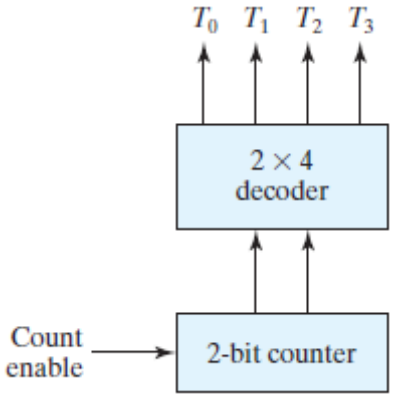
✗ Each output becomes a 1 after the negative-edge transition of a clock pulse and remains 1 during the next clock cycle.

Logic Circuit Diagram (Using D Flip flops)



Counter and Decoder

- ✗ For an alternative design, the timing signals can be generated by a two-bit counter that goes through four distinct states.



- ✗ The decoder decodes the four states of the counter and generates the required sequence of timing signals.
- ✗ To generate 2^n timing signals, we need either a shift register with 2^n flip-flops or an n -bit binary counter together with an n -to-2^n -line decoder.

Example:

- 16 timing signals can be generated with a 16-bit shift register connected as a ring counter or with a 4-bit binary counter and a 4-to-16-line decoder.
  - In the first case, 16 flip-flops are needed.
  - In the second, 4 flip-flops are needed and 16 four-input AND gates for the decoder.
- It is also possible to generate the timing signals with a combination of a shift register and a decoder.

c) Johnson Counter:

- ✗ A k -bit ring counter circulates a single bit among the flip-flops to provide k distinguishable states.
- ✗ The number of states can be doubled if the shift register is connected as a *switch-tail* ring counter.
- ✗ A k -bit switch-tail ring counter will go through a sequence of 2 k states.
- ✗ Starting from all 0's, each shift operation inserts 1's from the left until the register is filled with all 1's. In the next sequences, 0's are inserted from the left until the register is again filled with all 0's.

- ✔ A switch-tail ring counter is a circular shift register with the complemented output of the last flip-flop connected to the input of the first flip-flop.
- ✗ The circular connection is made from the complemented output of the rightmost flip-flop to the input of the leftmost flip-flop.

- ✔ The register shifts its contents once to the right with every clock pulse, and at the same time, the complemented value of the E flip-flop is transferred into the A flip-flop.

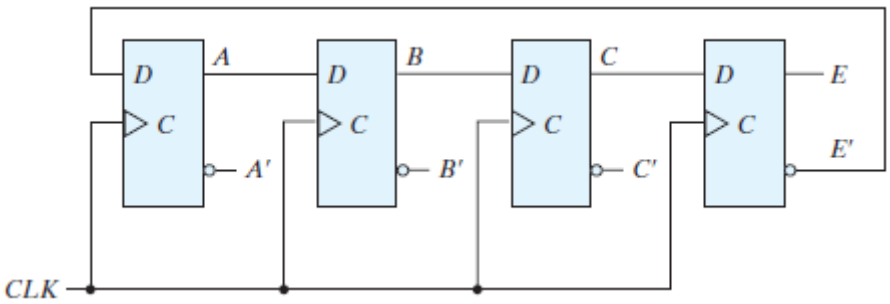
- ✗ Starting from a cleared state, the switch-tail ring counter goes through a sequence of eight states

Count Sequence and required decoding

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	$C'E$

- ✗ A Johnson counter is a k -bit switch-tail ring counter with 2 k decoding gates to provide outputs for 2 k timing signals

Four-stage switch-tail ring counter



- ✗ Each gate is enabled during one particular state sequence, the outputs of the gates generate eight timing signals in succession.
- ✗ The decoding of a *k* -bit switch-tail ring counter to obtain 2 *k* timing signals follows a regular pattern.
- ✗ The all-0's state is decoded by taking the complement of the two extreme flip-flop outputs.
- ✗ The all-1's state is decoded by taking the normal outputs of the two extreme flip-flops.
- ✗ All other states are decoded from an adjacent 1, 0 or 0, 1 pattern in the sequence.

Example:

- sequence 7 has an adjacent 0, 1 pattern in flip-flops *B* and *C* .
- The decoded output is then obtained by taking the complement of *B* and the normal output of *C,* or *BC*.

- ➢ Modifying the circuit to avoid undesirable condition when finding itself in an unused states..
- ➢ One correcting procedure is to disconnect the output from flip-flop *B* that goes to the *D* input of flip-flop *C* and instead enable the input of flip-flop *C* by the function

$$D_C = (A + C)B$$

D_C is the flip-flop input equation for the *D* input of flip-flop *C*.

- Johnson counters can be constructed for any number of timing sequences.

- The number of flip-flops needed is one-half the number of timing signals.
- The number of decoding gates is equal to the number of timing signals, and only two-input gates are needed.

Modulus-N-Counters:

- The counter with 'n' Flip-Flops has maximum MOD number 2^n.
- Find the number of Flip-Flops (n) required for the desired MOD number (N) using the equation,

$$\mathbf{2^n \geq N}$$

MOD 10 Counter:

$2^n = N = 10$

$2^3 = 8$ less than N.

$2^4 = 16 > N(10)$.

S.No	Asynchronous (ripple) counter	Synchronous counter
1	All the Flip-Flops are not clocked simultaneously.	All the Flip-Flops are clocked simultaneously.
2	The delay times of all Flip-Flops are added. Therefore there is considerable propagation delay.	There is minimum propagation delay.
3	Speed of operation is low	Speed of operation is high.
4	Logic circuit is very simple even for more number of states.	Design involves complex logic circuit as number of state increases.
5	Minimum numbers of logic devices are needed.	The number of logic devices is more than ripple counters.
6	Cheaper than synchronous counters.	Costlier than ripple counters.