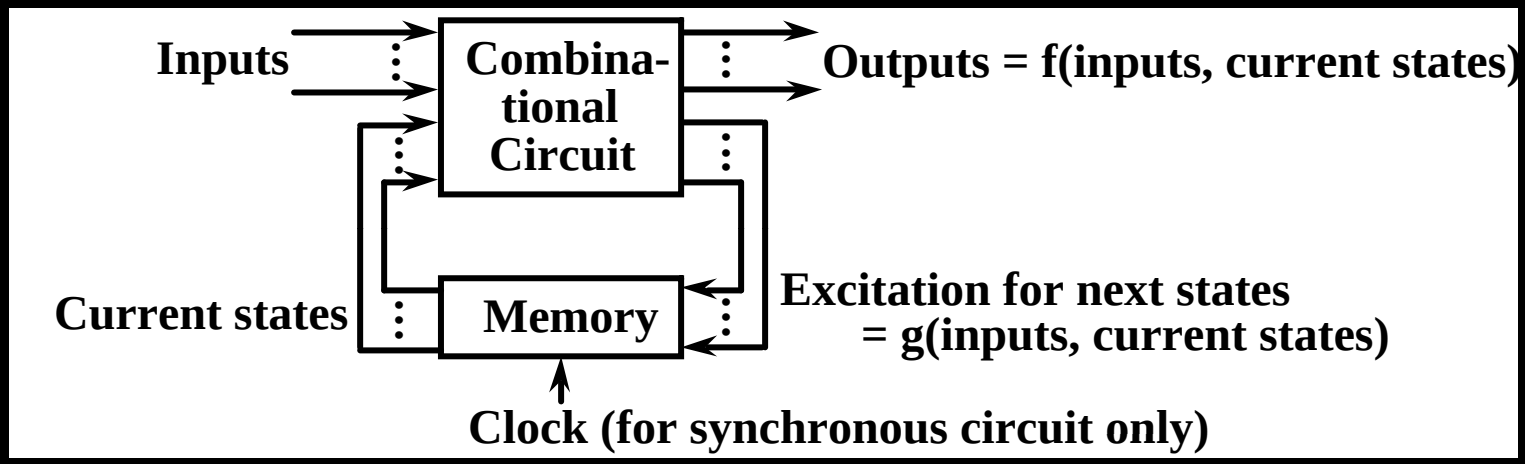
LOGIC DESIGN OF SEQUENTIAL CIRCUITS

- **General Model of Sequential Circuit**
- **Flip-Flop as Basic Memory Unit**
- **Analysis of Sequential Circuits**
- **Design of Sequential Circuits**
- **Sequential MSI Modules**

General Model of Sequential Circuit

- Output depends on circuit's history stored as state variables in memory.
- Mealy machine: outputs = F(inputs, current states)
- Moore machine: outputs = F(current states)
  It has no input variables other than the clock.
- Excitation signals act on memory for next states.

General Model of Sequential Circuit



- Synchronous circuit — State changes only at the occurrence of clock.
- Asynchronous circuit — State changes in response to the change of inputs.

FLIP-FLOP AS BASIC MEMORY UNIT

A Simple SR Latch

A simple SR Latch with two stable states:

Q=0, Q′=1 in ‘0’ state;

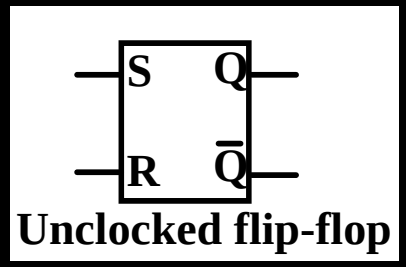
Q=1, Q′=0 in ‘1’ state.

S(set), R(reset) — excitation signals.

FLIP-FLOP AS BASIC MEMORY UNIT

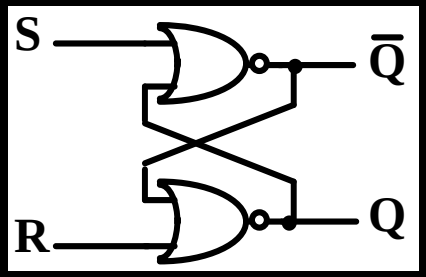
A Simple SR Latch

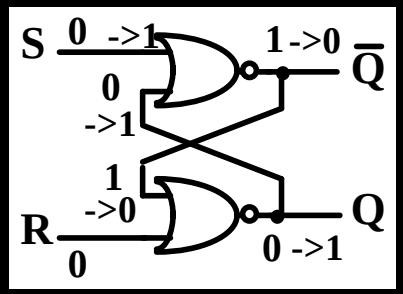
- Conceptual circuit: symbol, state transition table & analysis

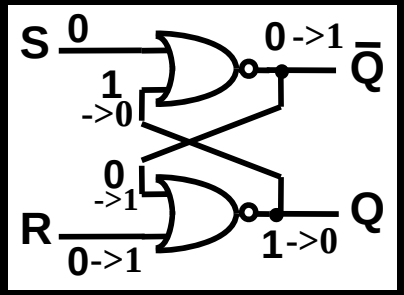


Unclocked flip-flop

S	R	Q(t)	Q(t+1)	
0	0	0	0	} Q(t+1)=Q(t)
0	0	1	1	
0	1	0	0	} Q(t+1)=0
0	1	1	0	
1	0	0	1	} Q(t+1)=1
1	0	1	1	
1	1	0	not defined	
1	1	1	not defined	





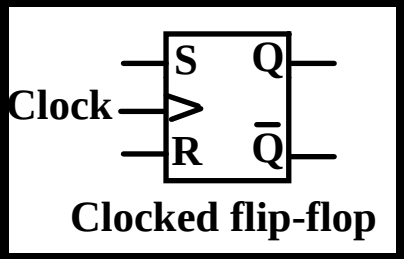


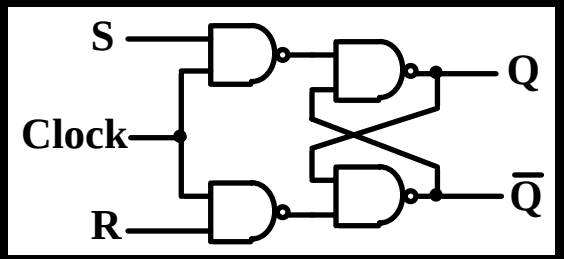
FLIP-FLOP AS BASIC MEMORY UNIT (continued)

Clocked SR flip-flop

- Flip-flop changes state only when clock = 1.

  Conceptual ckt, symbol, state transition table & equation:



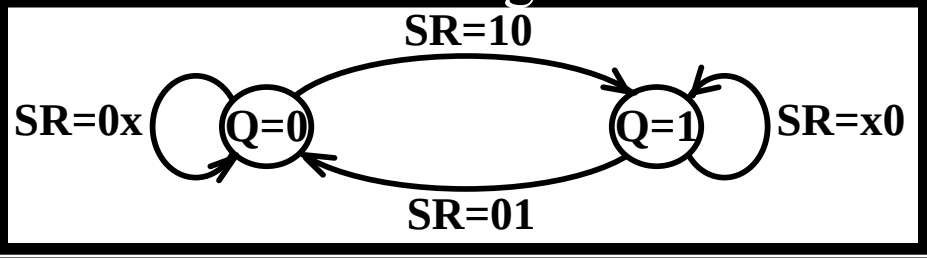


S	R	Q(t)	Q(t+1)	
0	0	0	0	} Q(t+1)=Q(t)
0	0	1	1	
0	1	0	0	} Q(t+1)=0
0	1	1	0	
1	0	0	1	} Q(t+1)=1
1	0	1	1	
1	1	0	not defined	
1	1	1	not defined	

$Q(t+1) = S+R' Q(t)$

$SR = 0$ (restriction)

- State transition diagram & excitation table:



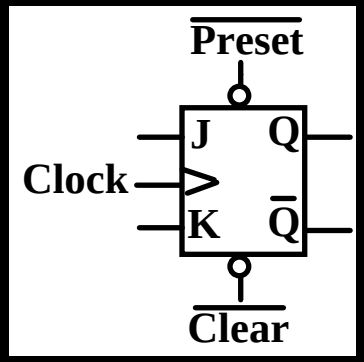
Q(t)	Q(t+1)	S	R
0	0	0	x
0	1	1	0
1	0	0	1
1	1	x	0

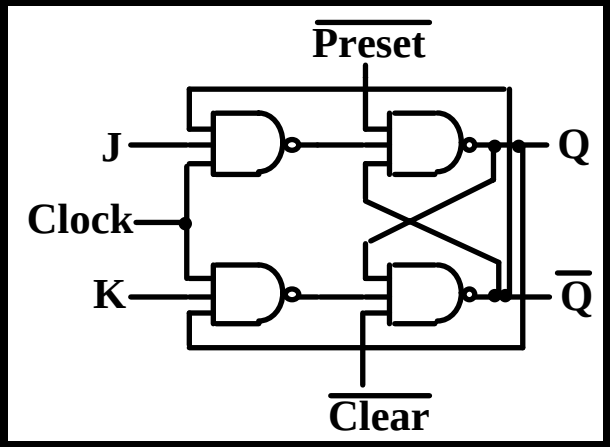
FLIP-FLOP AS BASIC MEMORY UNIT (continued)

Clocked JK flip-flop

- Extension of SR FF by allowing J=1 & K=1.

  Conceptual ckt, symbol, state transition table & equation:

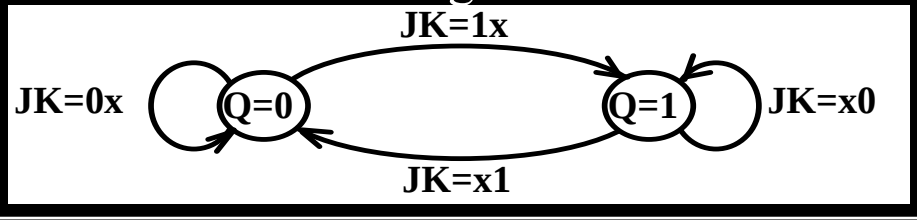




J	K	Q(t)	Q(t+1)	
0	0	0	0	} Q(t+1)=Q(t)
0	0	1	1	
1	0	0	0	} Q(t+1)=0
1	0	1	0	
0	1	0	1	} Q(t+1)=1
0	1	1	1	
1	1	0	1	} Q(t+1)=Q(t)′
1	1	1	0	

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

- State transition diagram & excitation table:

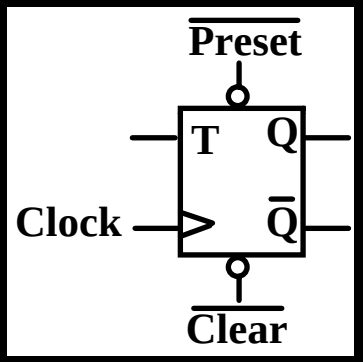


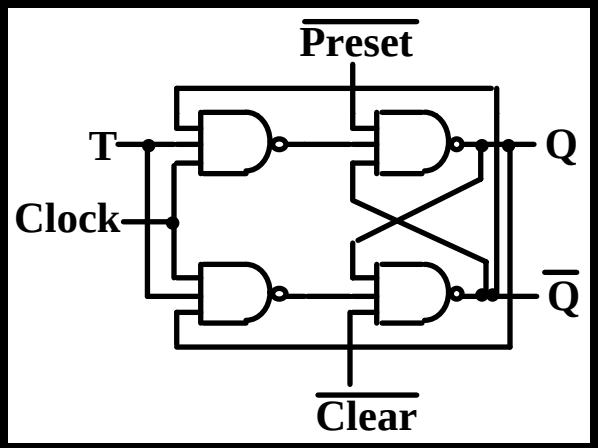
Q(t)	Q(t+1)	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

FLIP-FLOP AS BASIC MEMORY UNIT (continued)

Clocked T flip-flop

- A special case of JK FF by setting J=K=T.
  Conceptual ckt, symbol, state transition table & equation:

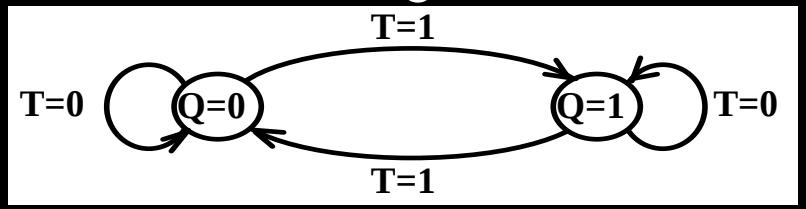




T	Q(t)	Q(t+1)	
0	0	0	} Q(t+1)=Q(t)
0	1	1	
1	0	1	
1	1	0	} Q(t+1)=Q(t)′

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

- State transition diagram & excitation table:



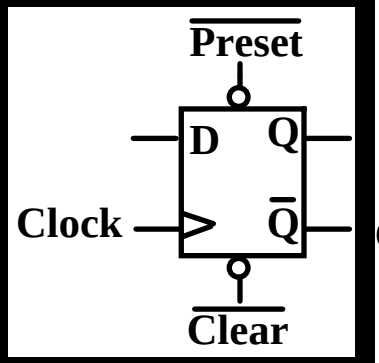
Q(t)	Q(t+1)	T
0	0	0
0	1	1
1	0	1
1	1	0

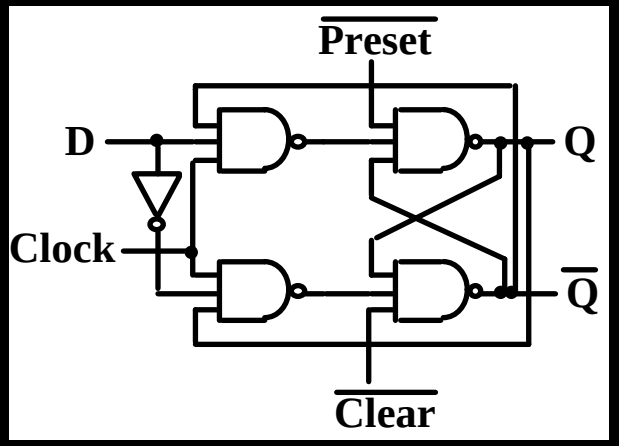
FLIP-FLOP AS BASIC MEMORY UNIT (continued)

Clocked D flip-flop

- A special case of JK FF by setting $J=\overline{K}=D$.

  Conceptual ckt, symbol, state transition table & equation:

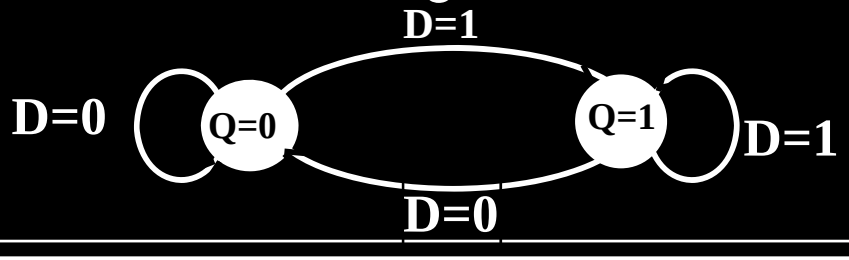




D	Q(t)	Q(t+1)
0	0	0
0	1	0
1	0	1
1	1	1

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

- State transition diagram & excitation table:



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

FLIP-FLOP AS BASIC MEMORY UNIT (continued)

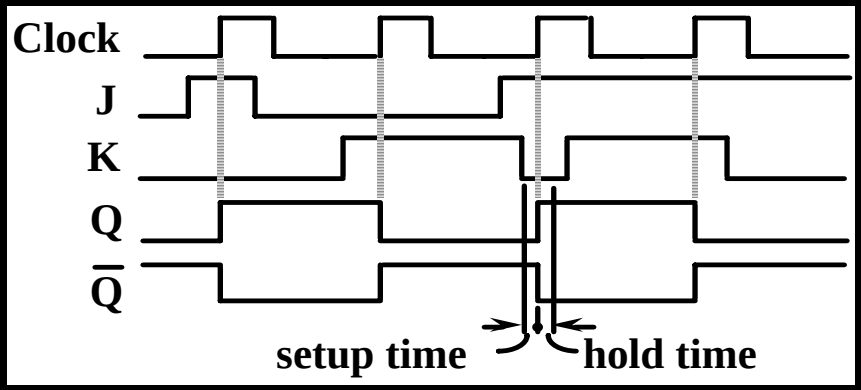
Edge-triggered flip-flop

- FF changes state only at the edge of the clock.

solves the problem of multiple triggering of level-controlled FF when level is active. Two methods of circuit implementation:

- ◊ master-slave flip-flop
- ◊ special circuit

Typical waveforms of edge-triggered JK FF:



- In the setup time, J,K must be constant prior to the active edge of the clock.
- In the hold time, J,K must not change after the active edge of the clock.

ANALYSIS OF SEQUENTIAL CIRCUITS

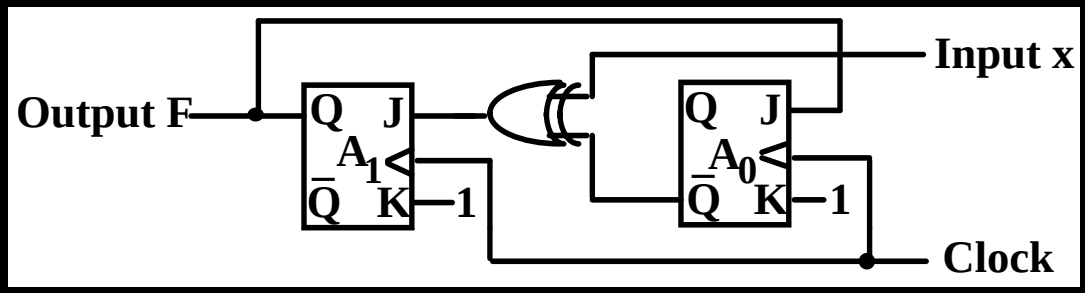
Write logical expressions for excitation and output in terms of input and current state. Write state transition table and output table. Draw the state transition diagram.

Example 4.1

$J_1 = x \oplus A_0'$ $K_1 = 1$

$J_0 = A_1$ $K_0 = 1$

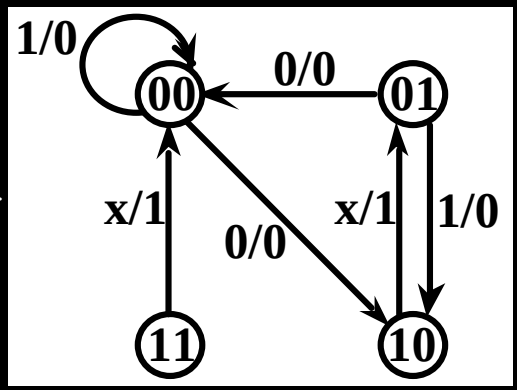
$F = A_1$



	Current						Next State		
x	A_1	A_0	J_1	K_1	J_0	K_0	$A_1(t+1)$	$A_0(t+1)$	F
0	0	0	1	1	0	1	1	0	0
0	0	1	0	1	0	1	0	0	0
0	1	0	1	1	1	1	0	1	1
0	1	1	0	1	1	1	0	0	1
1	0	0	0	1	0	1	0	0	0
1	0	1	1	1	0	1	1	0	0
1	1	0	0	1	1	1	0	1	1
1	1	1	1	1	1	1	0	0	1

When x=0,
00->10->01->00->.
When x=1,
01->10->01->…

Self-correcting:
x=0, 11->00
x=1, 11->00, 00->00



DESIGN OF SEQUENTIAL CIRCUITS

Design of a Gray Code Counter (Example 4.2)

- Problem specification.
  For the counter,
  - find state transition table.
  - Excitation table for the type of chosen FF.
  - Simplest expressions of excitation signals.
  - Check the self-correcting property. If not correct, modify the design.
  - Implement the counter, then the system.

- Example
  Design a counter for the given Gray code subsequence:

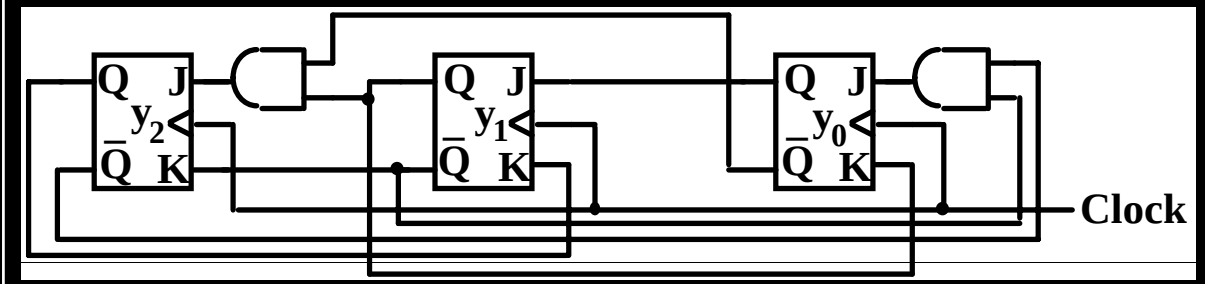
  000 ➡ 001 ➡ 011 ➡ 010 ➡ 110 ➡ 100 ➡ 000 (repeat)

DESIGN OF SEQUENTIAL CIRCUITS (continued)

Design of the counter using JK FF's

Counting sequence state transition table
State transition table Excitation table

current y_2 y_1 y_0	next y_2^+ y_1^+ y_0^+	Excitation for JK FF's J_2 K_2	J_1 K_1	J_0 K_0
0 0 0	0 0 1	0 x	0 x	1 x
0 0 1	0 1 1	0 x	1 x	x 0
0 1 0	1 1 0	1 x	x 0	0 x
0 1 1	0 1 0	0 x	x 0	x 1
1 0 0	0 0 0	x 1	0 x	0 x
1 0 1	x x x	x x	x x	x x
1 1 0	1 0 0	x 0	x 1	0 x
1 1 1	x x x	x x	x x	x x



y_2y_1 / y_0	00	01	11	10
0	0	1	x	x
1	0	0	x	x

$J_2 = y_1\overline{y_0}$

y_2y_1 / y_0	00	01	11	10
0	x	x	0	1
1	x	x	x	x

$K_2 = \overline{y_1}$

y_2y_1 / y_0	00	01	11	10
0	0	x	x	0
1	1	x	x	x

$J_1 = y_0$

y_2y_1 / y_0	00	01	11	10
0	x	0	1	x
1	x	0	x	x

$K_1 = y_2$

y_2y_1 / y_0	00	01	11	10
0	1	0	0	0
1	x	x	x	x

$J_0 = \overline{y_2}\overline{y_1}$

y_2y_1 / y_0	00	01	11	10
0	x	x	x	x
1	0	1	x	x

$K_0 = y_1$

- Self-correcting:
  101 ->011
  111 -> 100

DESIGN OF SEQUENTIAL CIRCUITS (continued)

Design of the counter using D FF's

State transition table & Excitation table

current y_2 y_1 y_0	next y_2^+ y_1^+ y_0^+	Excitation D_2 D_1 D_0
0 0 0	0 0 1	0 0 1
0 0 1	0 1 1	0 1 1
0 1 0	1 1 0	1 1 0
0 1 1	0 1 0	0 1 0
1 0 0	0 0 0	0 0 0
1 0 1	x x x	x x x
1 1 0	1 0 0	1 0 0
1 1 1	x x x	x x x

y_0 \ y_2y_1	00	01	11	10
0	0	1	1	0
1	0	0	x	x

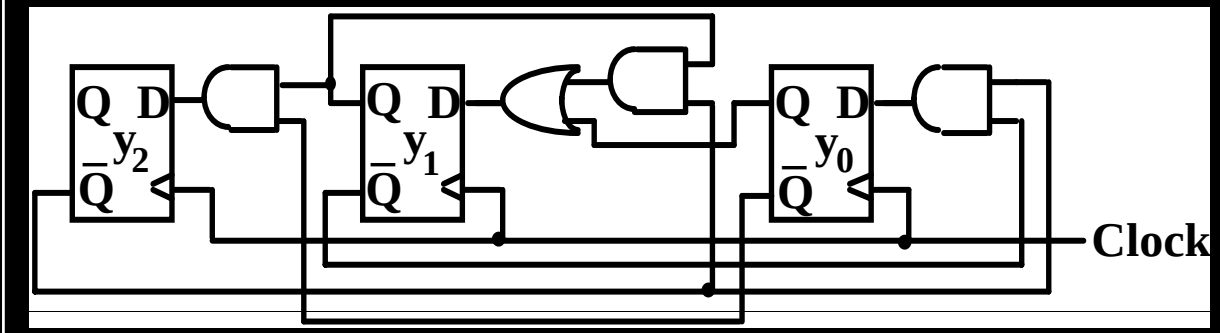
$D_2 = y_1\bar{y}_0$

y_0 \ y_2y_1	00	01	11	10
0	0	1	0	0
1	1	1	x	x

$D_1 = \bar{y}_2 y_1 + y_0$

y_0 \ y_2y_1	00	01	11	10
0	1	0	0	0
1	1	0	x	x

$D_0 = \bar{y}_2\bar{y}_1$

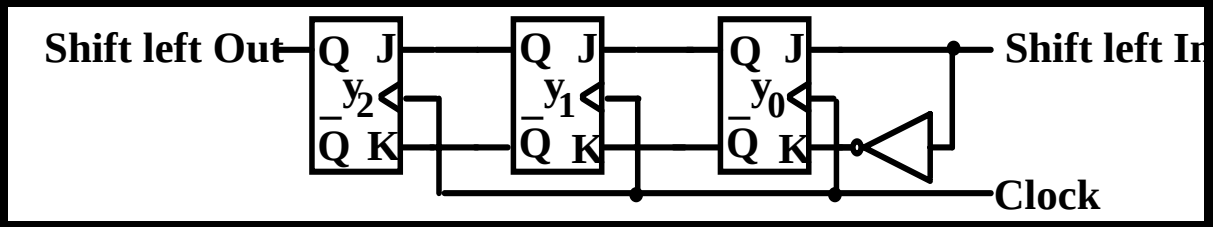


- Self-correcting:
  - 101 -> 010
  - 111 -> 010

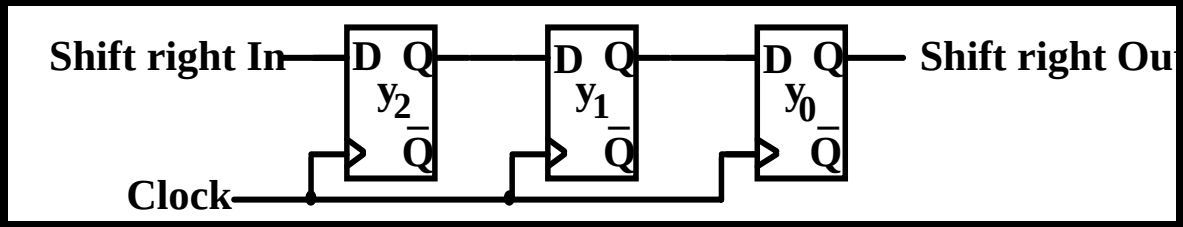
SEQUENTIAL MSI MODULES

Shift Register & Shift Operations

- Left shift, e.g., on JK FF's



- Logical right shift, e.g., on D FF's



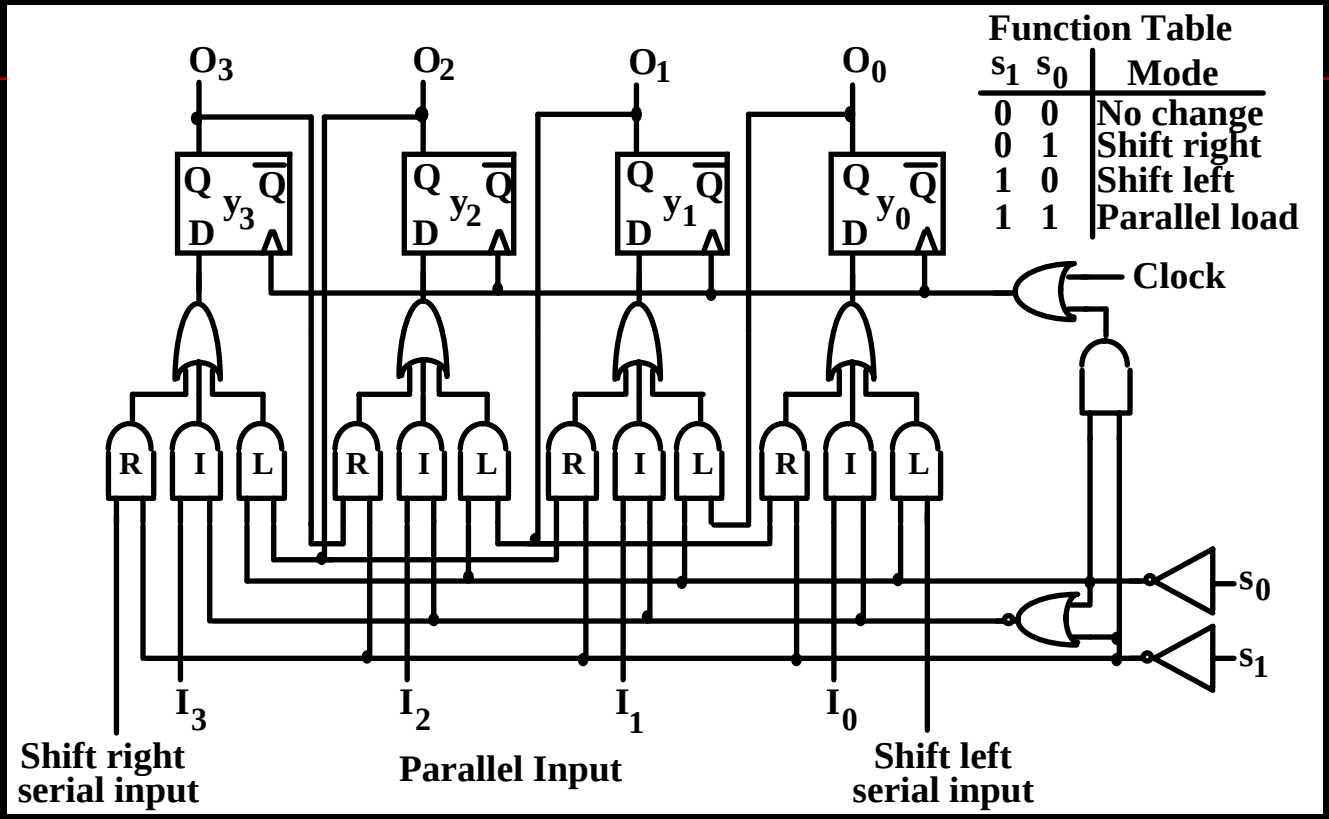
- Arithmetic right shift: connect MSB to shift in

  Arithmetic left shift: signal overflow if the sign changes.

- Rotate: connect shift in & shift out

SEQUENTIAL MSI MODULES (continued)

Bidirectional Shift Register



Use AND or OR gate to block the clock.

To maintain D FF unaltered, either feedback Q to D or block the clock.

SEQUENTIAL MSI MODULES (continued)

Ripple-Carry Binary Counter

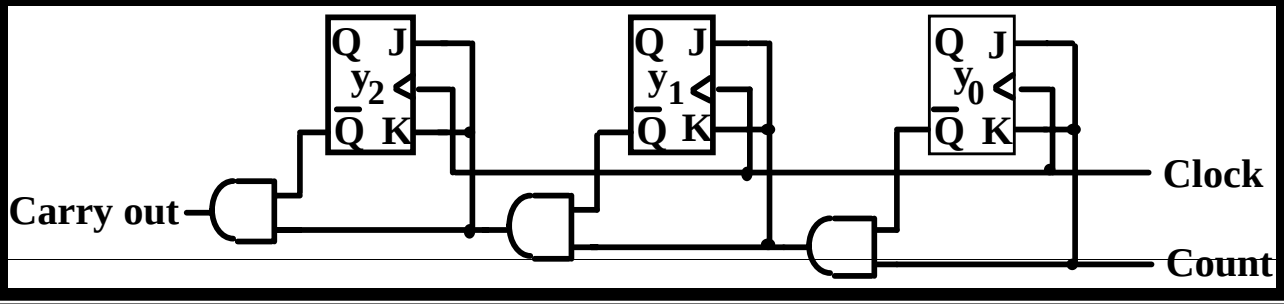
current $y_2\ y_1\ y_0$	Count up next $y_2^+\ y_1^+\ y_0^+$	Count down next $y_2^+\ y_1^+\ y_0^+$
0 0 0	0 0 1	1 1 1
0 0 1	0 1 0	0 0 0
0 1 0	0 1 1	0 0 1
0 1 1	1 0 0	0 1 0
1 0 0	1 0 1	0 1 1
1 0 1	1 1 0	1 0 0
1 1 0	1 1 1	1 0 1
1 1 1	0 0 0	1 1 0

Count up

y_0 triggers every clock.
y_1 triggers if y_0=1.
y_2 triggers if y_1y_0=1.
y_i triggers if $y_{i-1}\ldots y_0$=1.

Count down

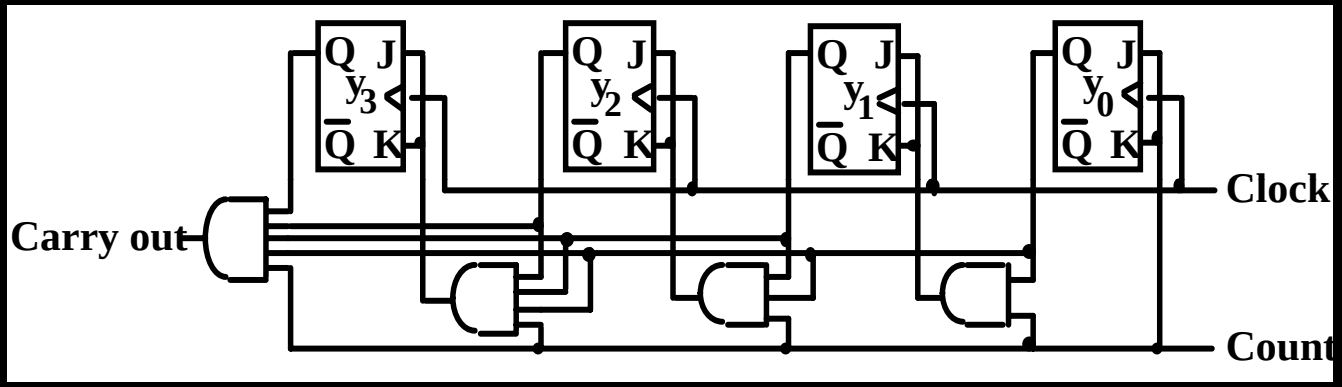
y_0 triggers every clock.
y_1 triggers if y_0=1.
y_2 triggers if y_1y_0=1.
y_i triggers if $y_{i-1}\ldots y_0$=1.



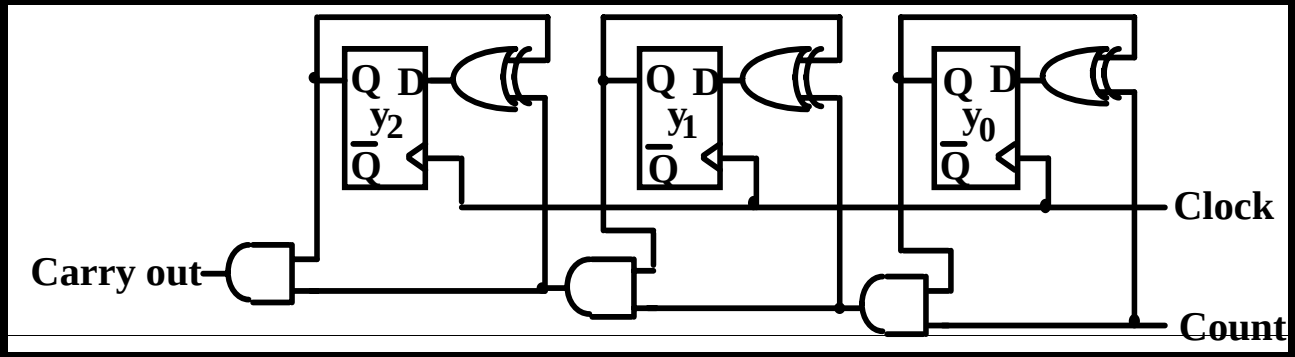
SEQUENTIAL MSI MODULES (continued)

Counter on D FF's & Parallel Counter

Parallel Counter



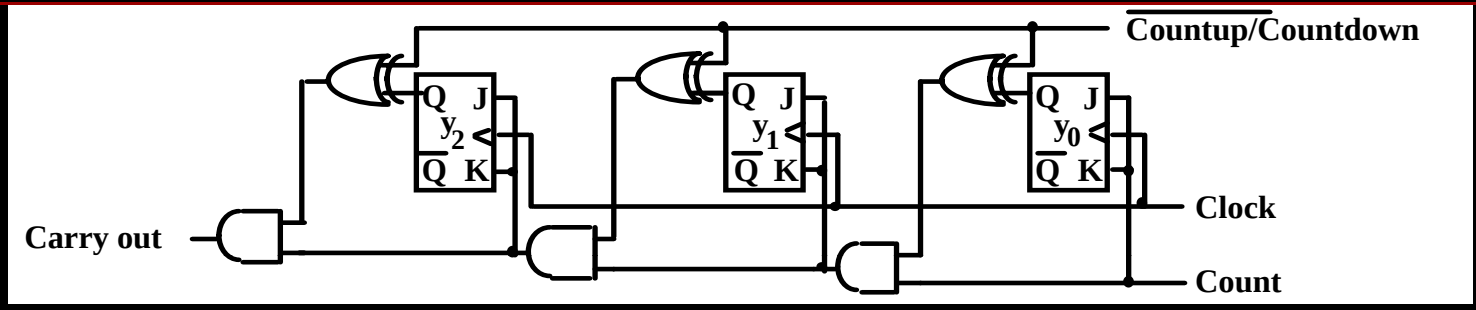
Counter on D FF's



SEQUENTIAL MSI MODULES (continued)

Up/Down Counter

Up/Down Counter selected by XOR gates



Multifunctional Up/Down Counter

Function Table

s_1	s_0	Mode
0	0	Clear
0	1	No action
1	0	Count up
1	1	Count down