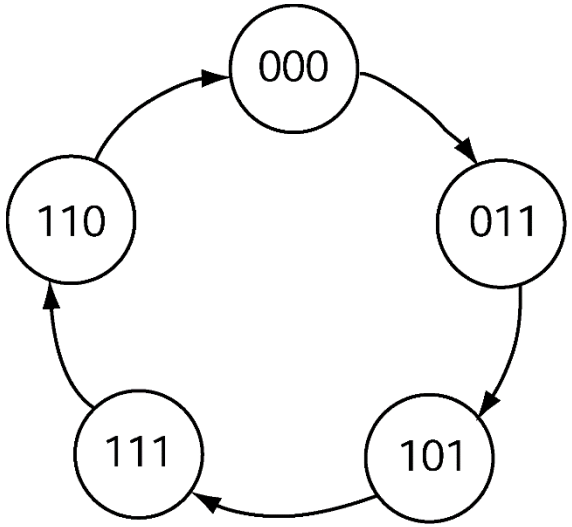
Sequential Circuit Design

Part II

General Counter Design

- Design a counter with the following sequence

0 → 3 → 5 → 7 → 6 → 0



Present state			Next state			JK 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	1	1	0	d	1	d	1	d
0	0	1	–	–	–	d	d	d	d	d	d
0	1	0	–	–	–	d	d	d	d	d	d
0	1	1	1	0	1	1	d	d	1	d	0
1	0	0	–	–	–	d	d	d	d	d	d
1	0	1	1	1	1	d	0	1	d	d	0
1	1	0	0	0	0	d	1	d	1	0	d
1	1	1	1	1	0	d	0	d	0	d	1

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

$$J_A = B$$

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

$$K_A = \overline{C}$$

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

$$J_B = 1$$

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

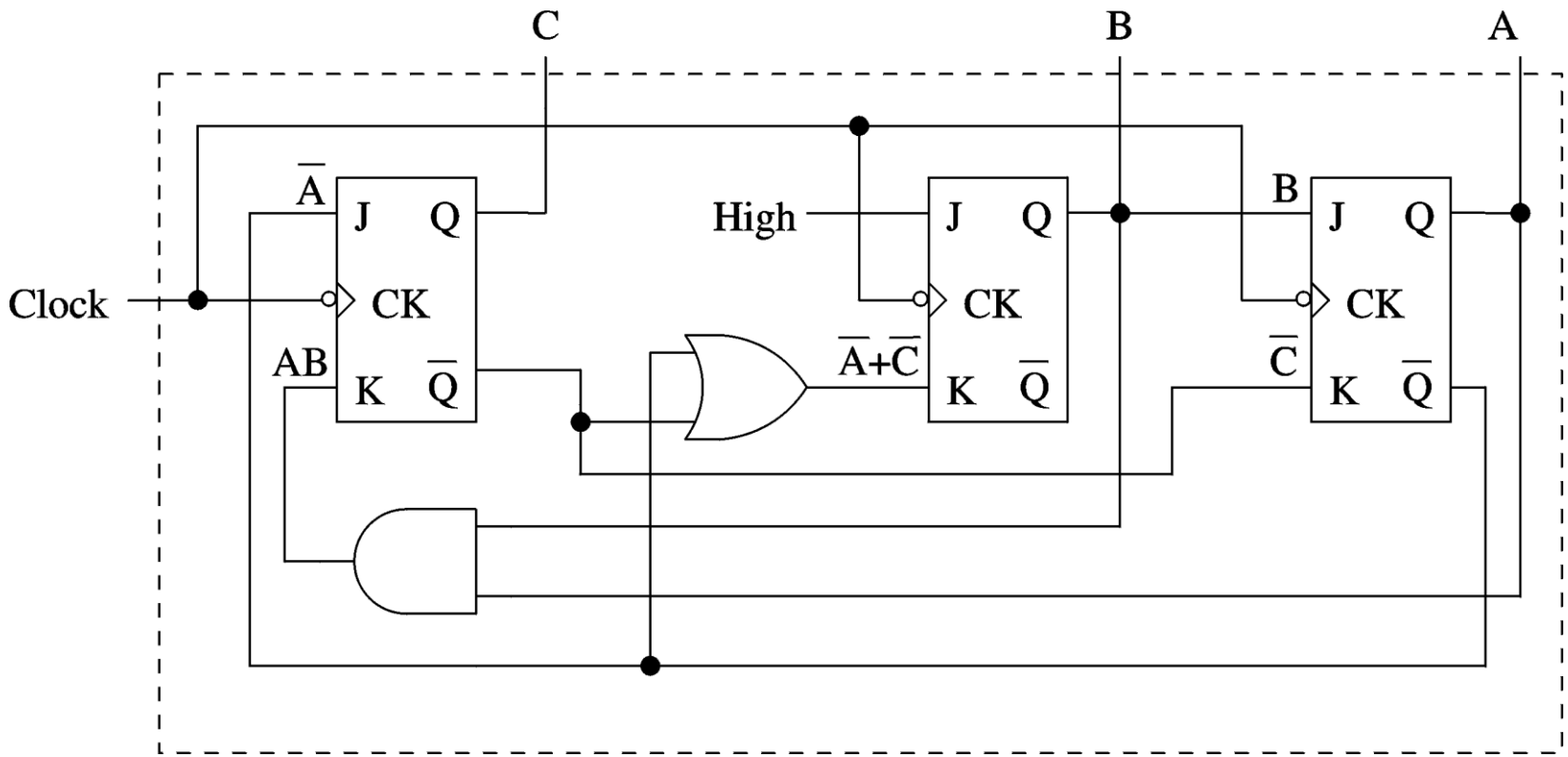
$$K_B = \overline{A} + \overline{C}$$

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

$$J_C = \overline{A}$$

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

$$K_C = A\,B$$


C
B
A
Clock
$\overline{A}$
J
Q
CK
AB
K
$\overline{Q}$
High
J
Q
CK
$\overline{A}+\overline{C}$
K
$\overline{Q}$
B
J
Q
CK
$\overline{C}$
K
$\overline{Q}$

General Design Process

1. Derive FSM
2. State assignment
   * Assign flip-flop states to the FSM states
     * Necessary to get an efficient design
3. Design table derivation
   * Derive a design table corresponding to the assignment in the last step
4. Logical expression derivation
   * Use K-maps as in our previous examples
5. Implementation

General Design Process

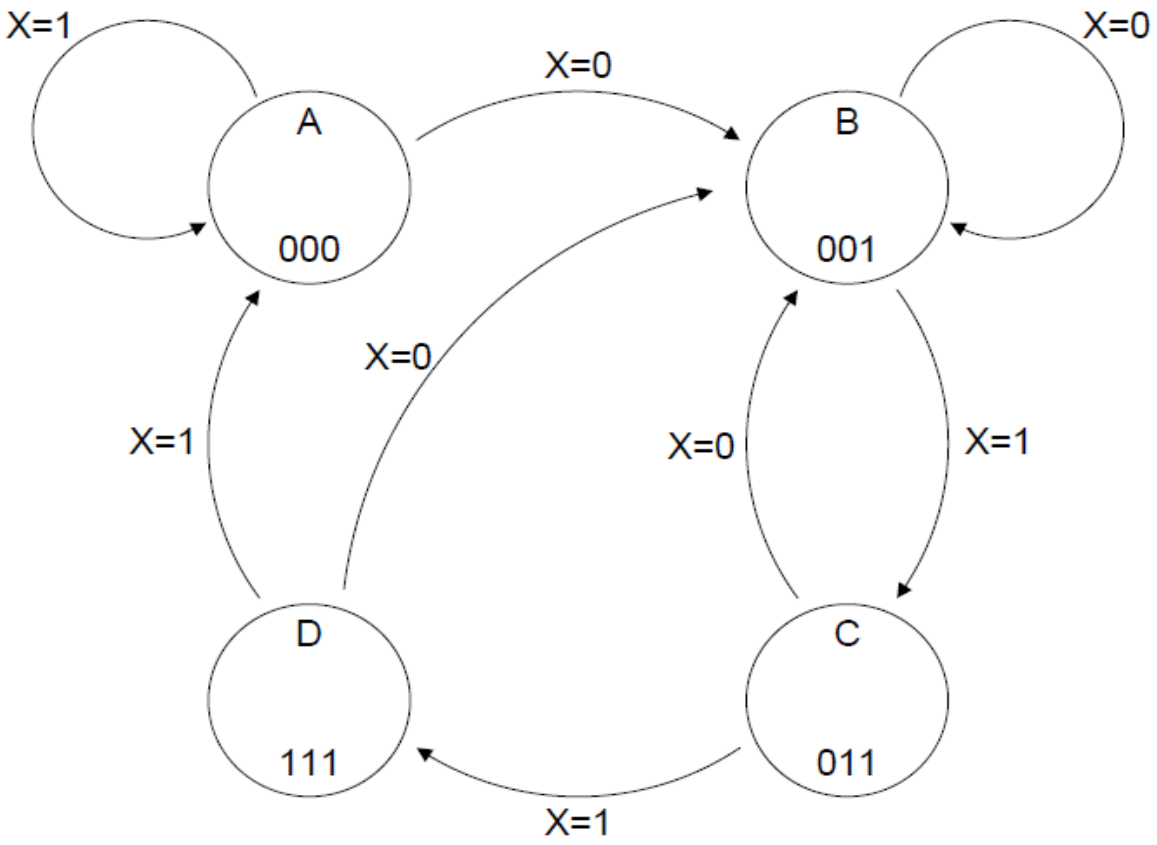
1. Draw a State Diagram
2. Make a Next State Truth Table
3. Pick Flip-Flop type
4. Add Flip-Flop inputs to Next State Truth Table using Flip-Flop excitation equation (This creates an Excitation Table.)
5. Solve equations for Flip-Flop inputs (K-maps)
6. Solve equations for Flip-Flop outputs (K-maps)
7. Implement the circuit

Example

- Design a simple sequence detector for the sequence 011. Include three outputs that indicate how many bits have been received in the correct sequence (For example, each output could be connected to an LED.).
- Step 1

Draw a State Diagram (Moore) and then assign binary State Identifiers.

MOORE SEQUENCE DETECTOR FOR 011



STATES
A=00
B=01
C=11
D=10

State ‘A’ is the starting state for this diagram.

Next State Truth Table

Step 2

State	X	O_2	O_1	O_0	State$^+$
A	0	0	0	0	B
A	1	0	0	0	A
B	0	0	0	1	B
B	1	0	0	1	C
D	0	1	1	1	B
D	1	1	1	1	A
C	0	0	1	1	B
C	1	0	1	1	D

Q_1	Q_0	X	O_2	O_1	O_0	Q_1^+	Q_0^+
0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0
0	1	0	0	0	1	0	1
0	1	1	0	0	1	1	1
1	0	0	1	1	1	0	1
1	0	1	1	1	1	0	0
1	1	0	0	1	1	0	1
1	1	1	0	1	1	1	0

Step 3

Pick Flip-Flop type

- Pick D Flip-Flop

Excitation Table

Step 4

Q_1	Q_0	X	O_2	O_1	O_0	Q_1^+	Q_0^+	D_1	D_0
0	0	0	0	0	0	0	1	0	1
0	0	1	0	0	0	0	0	0	0
0	1	0	0	0	1	0	1	0	1
0	1	1	0	0	1	1	1	1	1
1	0	0	1	1	1	0	1	0	1
1	0	1	1	1	1	0	0	0	0
1	1	0	0	1	1	0	1	0	1
1	1	1	0	1	1	1	0	1	0

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

K-maps

Step 5

Solve equations for Flip-Flop inputs (K-maps)

$X\backslash Q_1Q_0$	00	01	11	10
0	0	0	0	0
1	0	1	1	0

$$D_1 = XQ_0$$

$X\backslash Q_1Q_0$	00	01	11	10
0	1	1	1	1
1	0	1	0	0

$$D_0 = \overline{X} + \overline{Q}_1Q_0$$

Step 6

Solve equations for Flip-Flop inputs (K-maps)

$Q_1\backslash Q_0$	0	1
0	0	0
1	1	0

$$O_2 = Q_1\overline{Q}_0$$

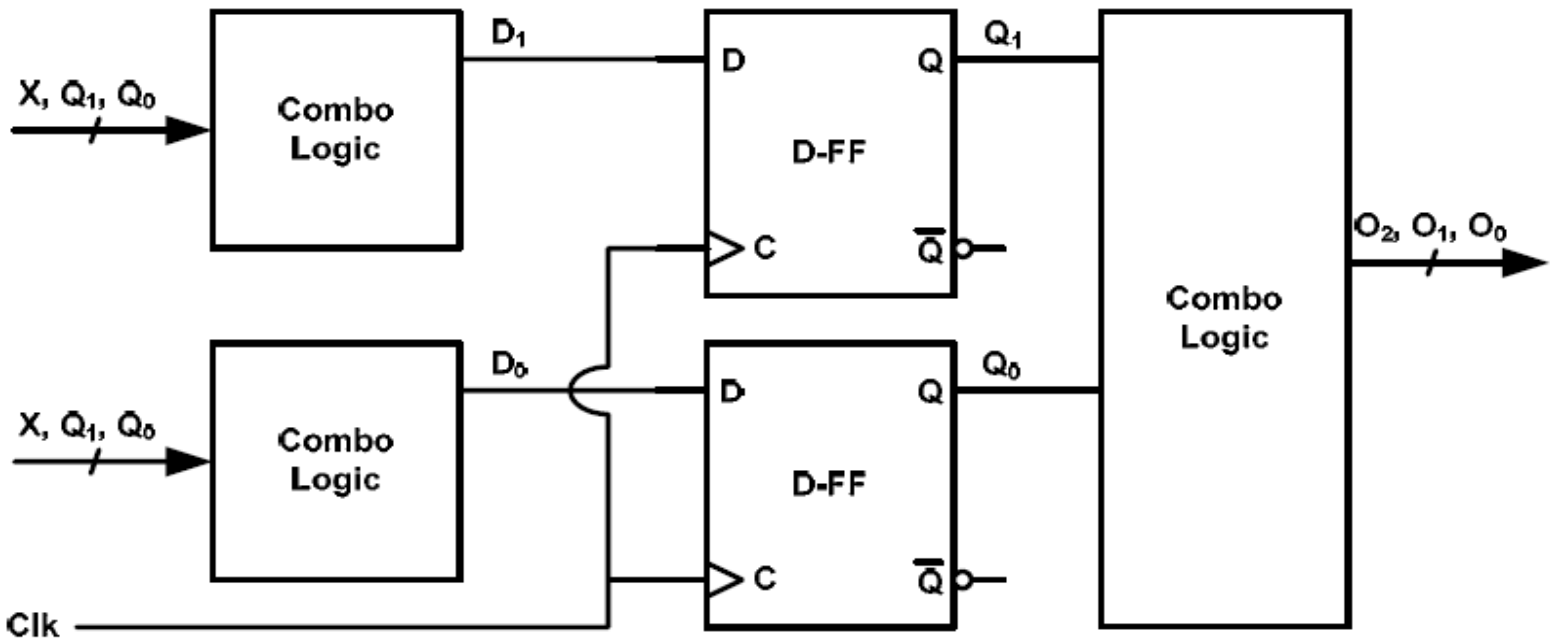
$Q_1\backslash Q_0$	0	1
0	0	0
1	1	1

$$O_1 = Q_1$$

$Q_1\backslash Q_0$	0	1
0	0	1
1	1	1

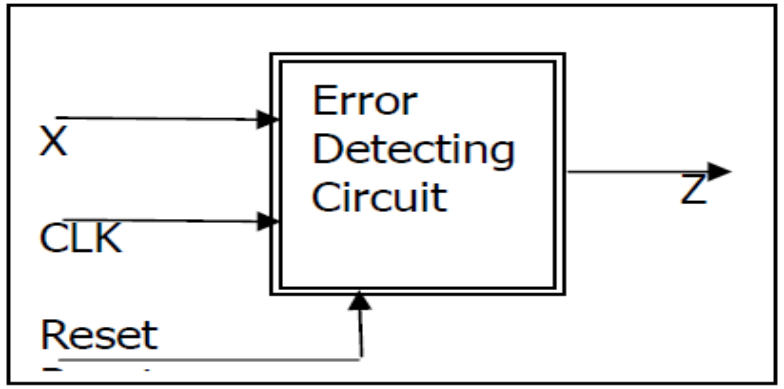
$$O_0 = Q_1 + Q_0$$

Implement the circuit



Example of Mealy Machine

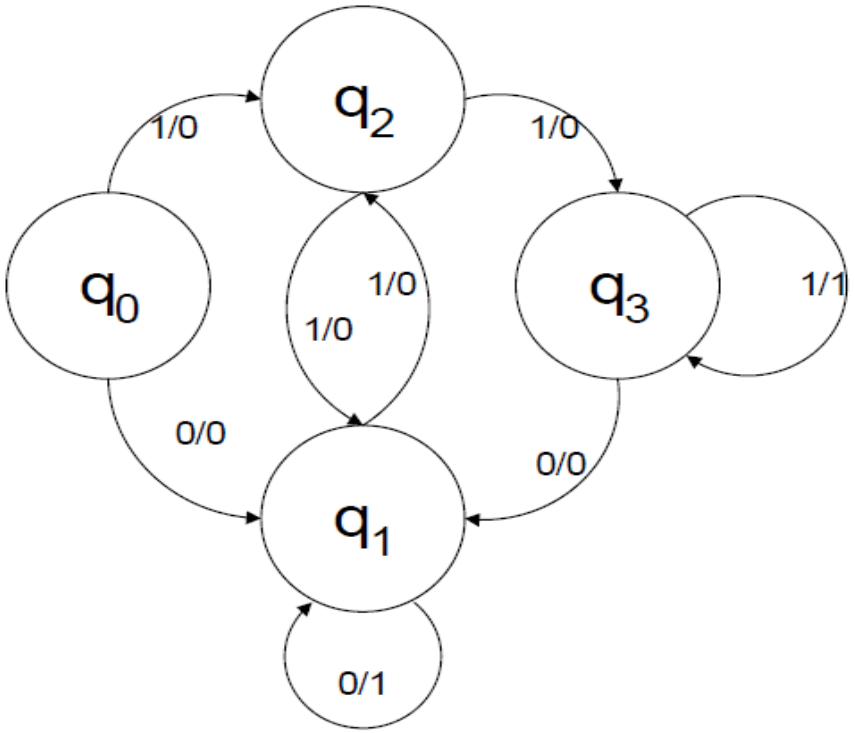
- Design an error detector for the following sequential circuit. The circuit has a single input x and a single output z. Data arrive serially on x synchronized with the clock. The output z (an error) should be “1” whenever two consecutive zeroes or three consecutive ones appear on line x. Implement the circuit using D, JK, RS and T flip-flops.



Example – contd..

x	0	0	1	1	1	1	1	0	0	0	1	1	1	0	0	1
z	0	1	0	0	1	1	1	0	1	1	0	0	1	0	1	0
clk	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑

State Diagram:



Example – contd..

State assignment

Assign the following state arbitrary:

q_0=00

q_1=01

q_2=10

q_3=11

Present state	Next state, output	
	x=0	x=1
y_1 y_0	y_1 y_0 / z	y_1 y_0 / z
0 0	0 1, 0	1 0, 0
0 1	0 1, 1	1 0, 0
1 0	0 1, 0	1 1, 0
1 1	0 1, 0	1 1, 1

$$\text{error} = z = \bar{y}_1 y_0 \bar{x} + y_1 y_0 x$$

Example – contd..

Implementation using D Flip flop

present→next state	D
0→0	0
0→1	1
1→0	0
1→1	1

y_1 y_0 \ x	0	1
00	0	1
01	0	1
11	0	1
10	0	1

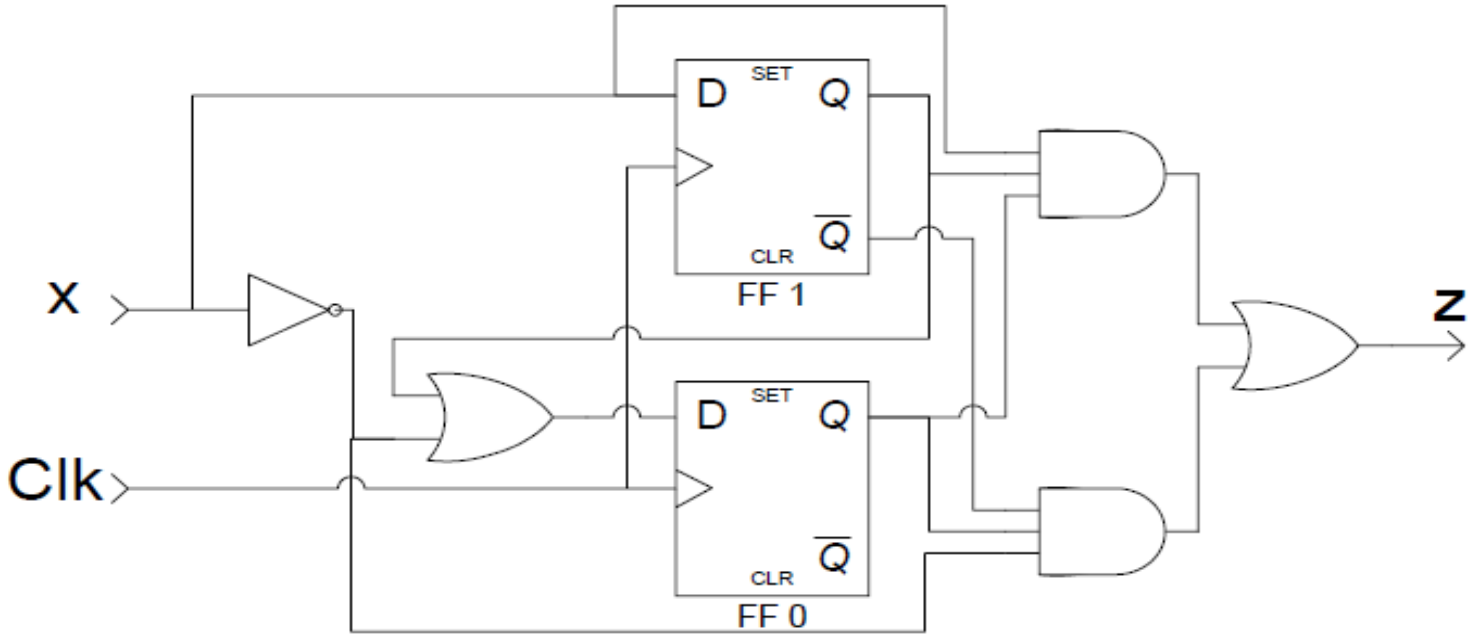
$$y_1^+ = D_1 = x$$

y_1 y_0 \ x	0	1
00	1	0
01	1	0
11	1	1
10	1	1

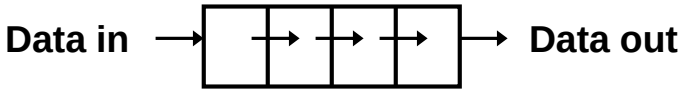
$$y_0^+ = D_0 = \bar{x} + y$$

Example – contd..

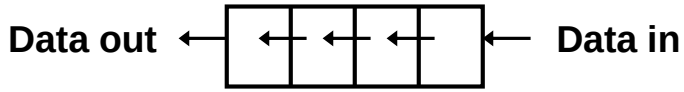
Implementation using D Flip flop



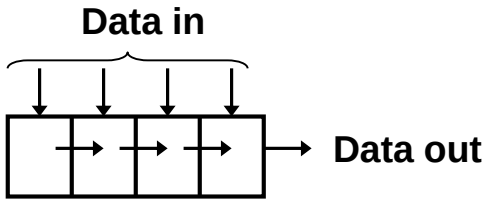
Shift Registers



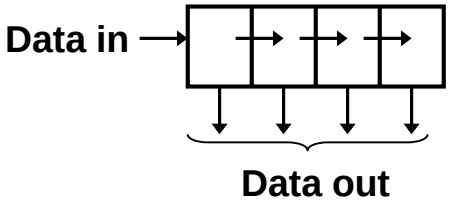
(a) Serial in/shift right/serial out



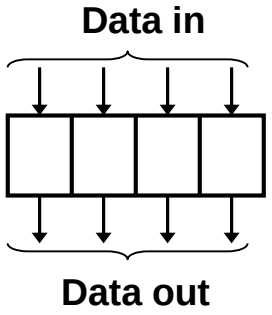
(b) Serial in/shift left/serial out



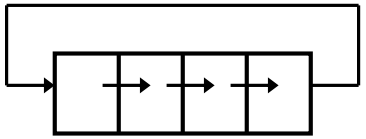
(c) Parallel in/serial out



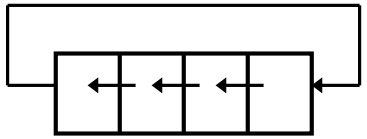
(d) Serial in/parallel out



(e) Parallel in / parallel out



(f) Rotate right

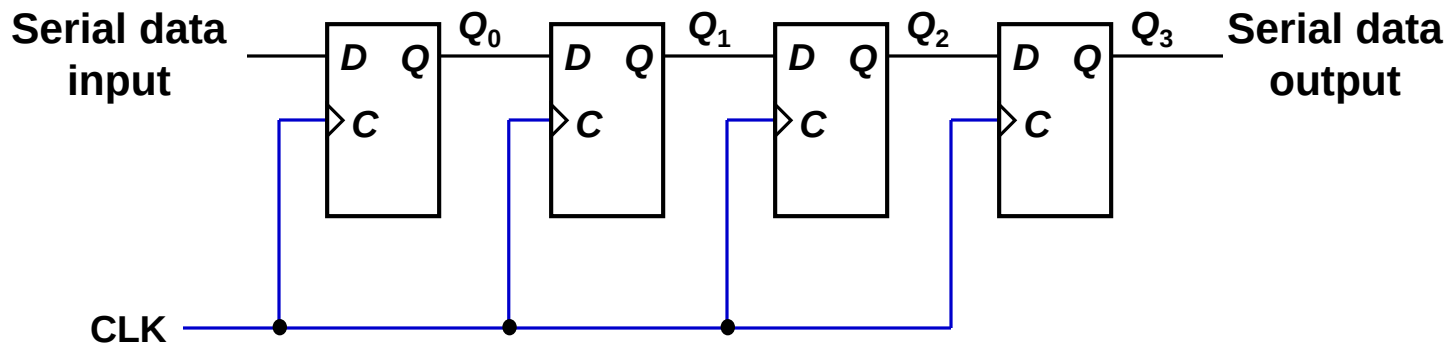


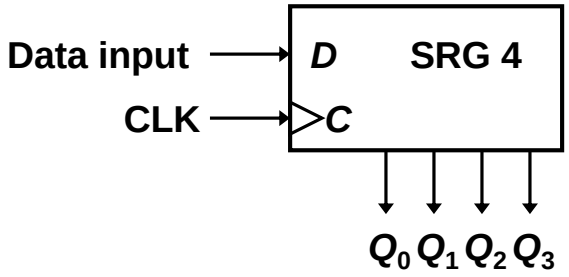
(g) Rotate left

- Registers are used for storing data
- Shift registers are used for storage and data movement
- Each stage (flip-flop) in a shift register represents one bit of storage
- shifting capability of a register permits the movement of data
  - from stage to stage within the register,
  - or into or out of the register upon application of clock pulses.

Serial In/Serial Out Shift Registers

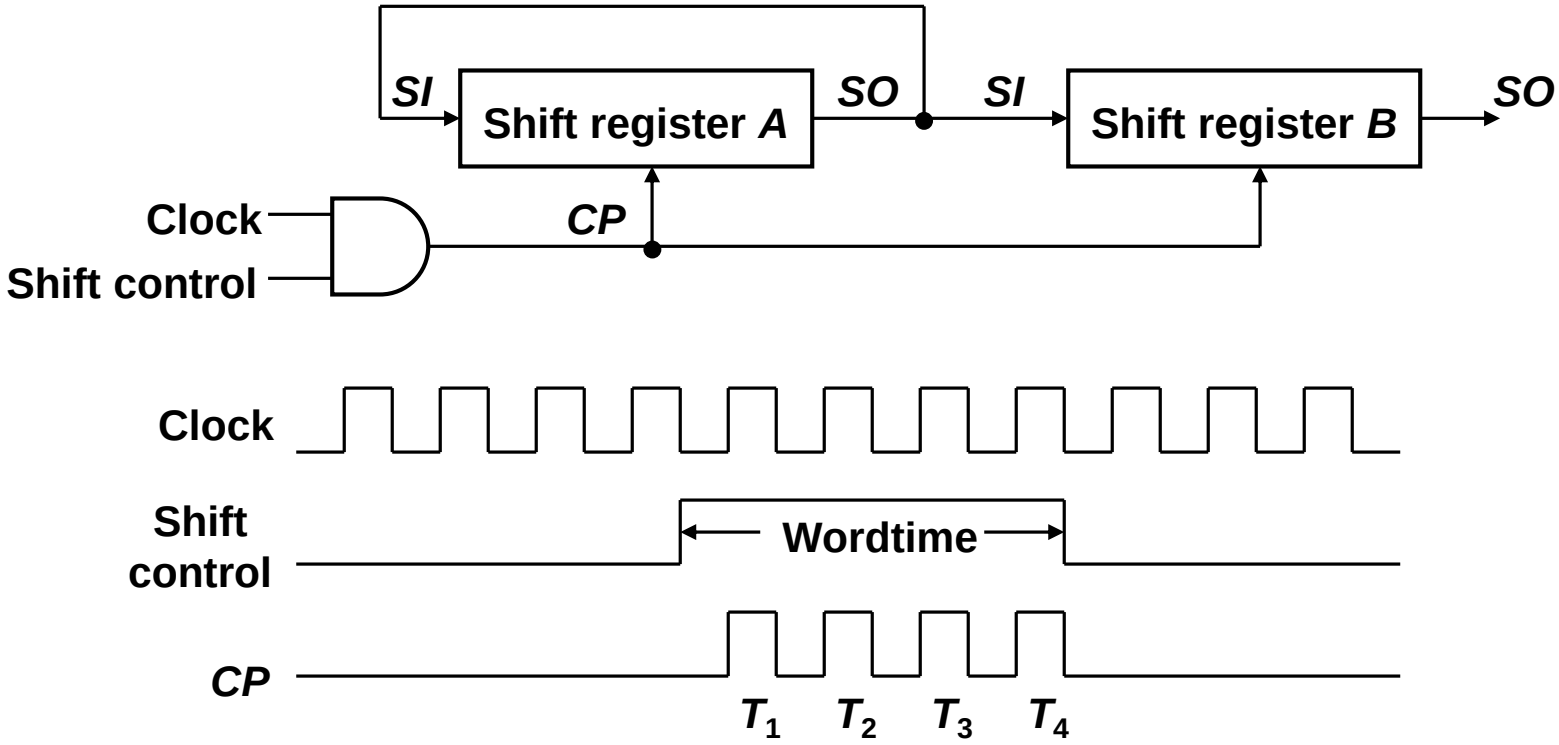
- Accepts data serially – one bit at a time – and also produces output serially.





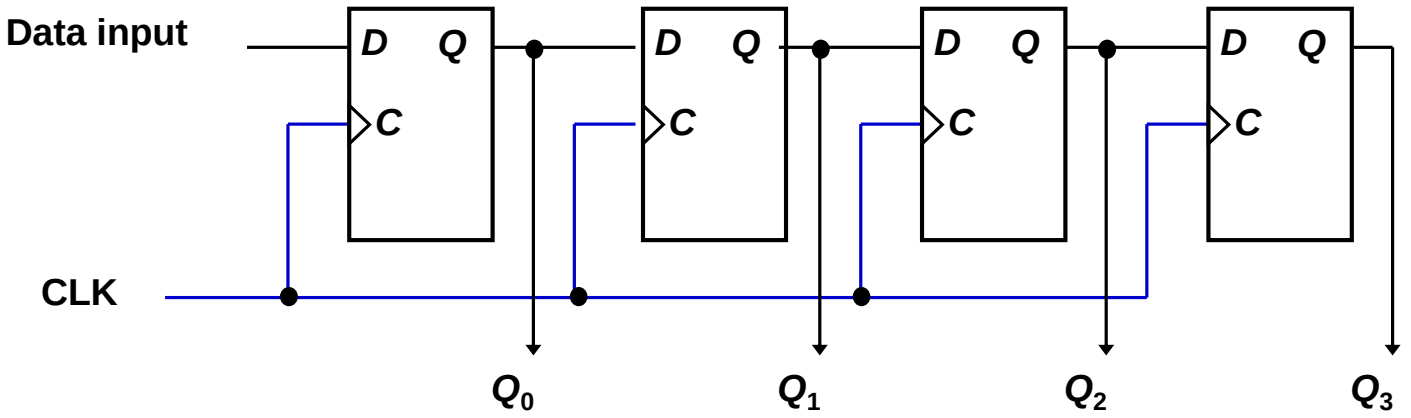
Serial In/Serial Out Shift Registers

- Application – Serial transfer of data from one register to another.

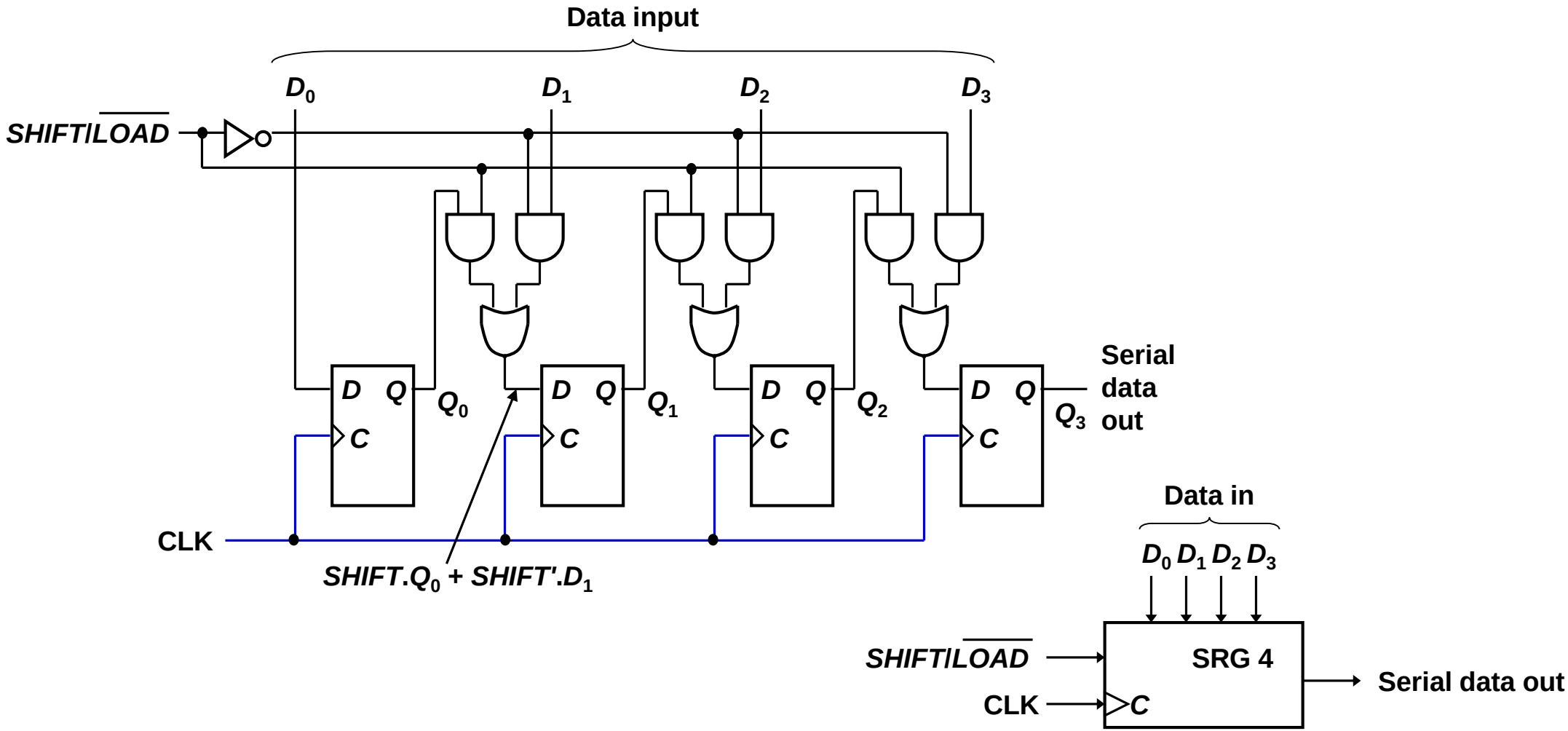


Serial In/Parallel Out Shift Registers

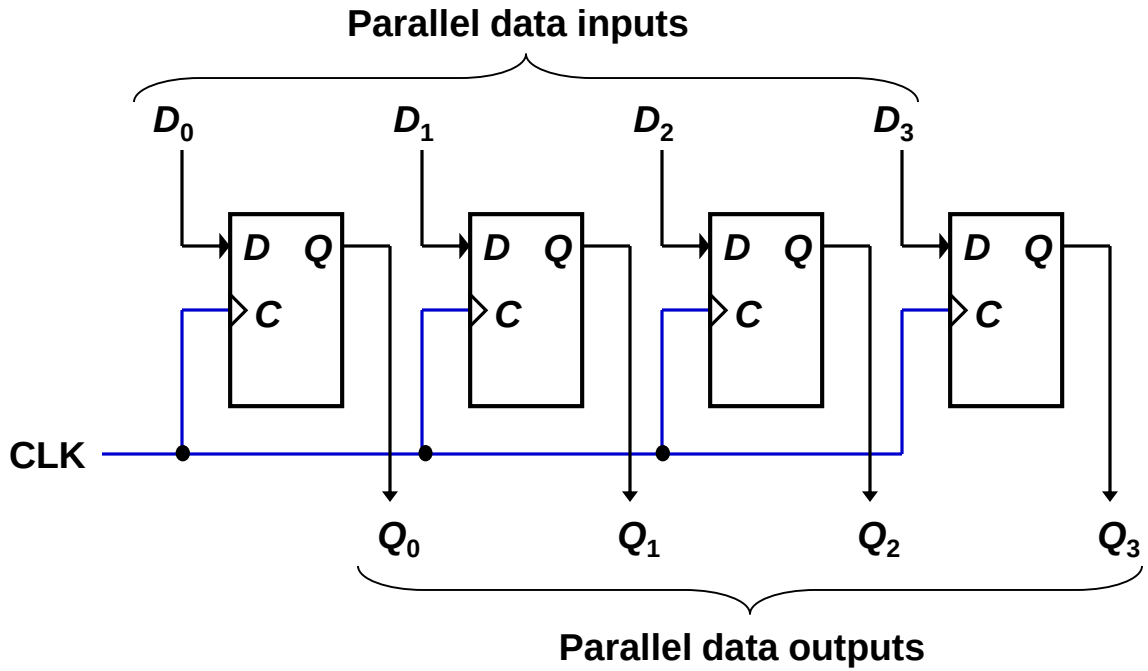
- Accepts data serially.
- Outputs of all stages are available simultaneously.



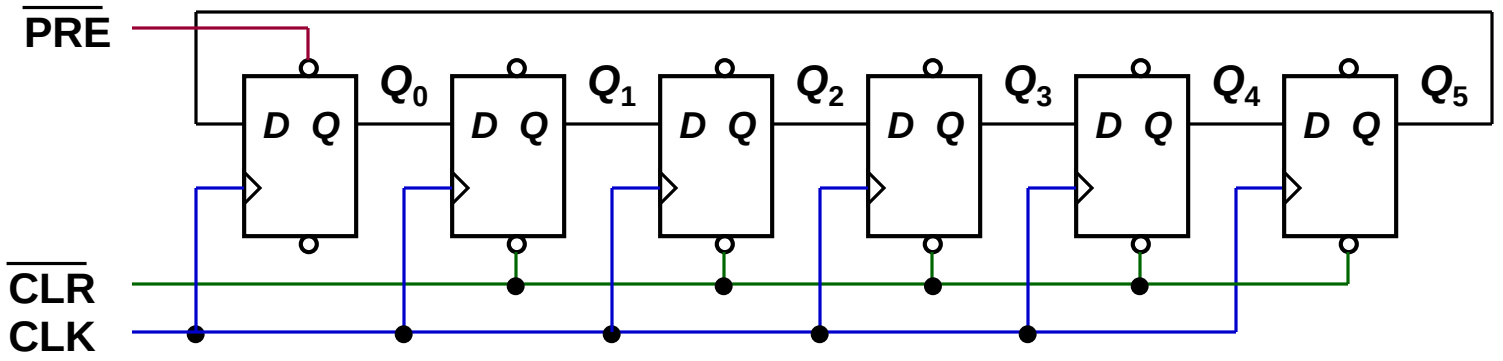
Parallel In/Serial Out Shift Registers



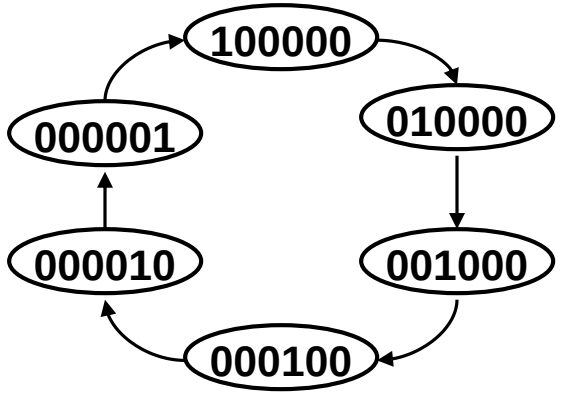
Parallel In/Parallel Out Shift Registers



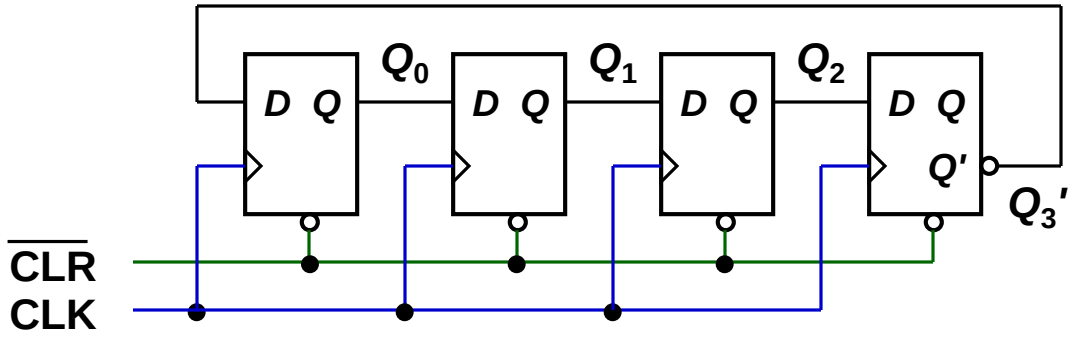
Ring Counters – A 6 bit ring counter



Clock	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5
0	1	0	0	0	0	0
1	0	1	0	0	0	0
2	0	0	1	0	0	0
3	0	0	0	1	0	0
4	0	0	0	0	1	0
5	0	0	0	0	0	1



Johnson Counters – A 4-bit Johnson Counter



Clock	Q_0	Q_1	Q_2	Q_3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1