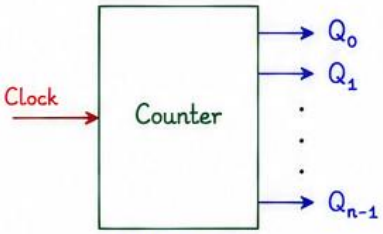
Concept of Counter

Sequential Logic Circuits

★ What is a Counter?

- A counter is a sequential circuit that goes through a prescribed sequence of states on the application of clock pulses.
- It is used for counting the number of pulses, measuring time intervals, frequency division, sequence generation, etc.



n-bit Counter

★ Example

A 3-bit counter has 3 flip-flops.

Total states = $2^3 = 8$

Sequence (Up Counter):

Count	Q_2	Q_1	Q_0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Then it returns to 0.

★ Basic Operation

- A counter has 'n' flip-flops, so it can have 2^n different states.
- With each clock pulse, it moves from one state to the next state in a sequence.
- After the last state, it returns to the first state and the process repeats.

★ Types of Counters

- Asynchronous (Ripple) Counter
  - Flip-flops are triggered one after another (ripple effect).
- Synchronous Counter
  - All flip-flops are triggered simultaneously by the same clock.
- Up Counter
  - Counts in increasing order (0, 1, 2, 3, ...).
- Down Counter
  - Counts in decreasing order (... 3, 2, 1, 0).
- Up/Down Counter
  - Can count in both directions based on a control input.

Difference Between Asynchronous and Synchronous Counter

Feature	Asynchronous (Ripple) Counter	Synchronous Counter
1. Clocking	Only the first flip-flop receives the external clock. Other flip-flops are triggered by the output of the previous flip-flop.	All flip-flops receive the same clock signal simultaneously.
2. Operation	Flip-flops change state one after another, creating a ripple effect.	All flip-flops change state at the same time.
3. Speed	Slower due to cumulative propagation delay in each flip-flop.	Faster since there is no ripple delay.
4. Max. Operating Frequency	Low	High
5. Hardware Complexity	Simple design, requires fewer gates.	More complex, requires additional gates for combinational logic.
6. Power Consumption	Generally lower.	Generally higher.
7. Glitches	More prone to glitches due to propagation delay.	Fewer glitches.
8. Use	Used in simple, low-speed applications like frequency division.	Used in high-speed applications like microprocessors, memory address counters.
9. Example	3-bit Ripple Counter using JK/T flip-flops.	3-bit Synchronous Counter using JK/T flip-flops.

★ Note

Synchronous counters are preferred in modern digital systems due to their high speed and reliability, while asynchronous counters are used in simple and low-cost designs.

Operation of 3-bit Ripple UP/DOWN Counter

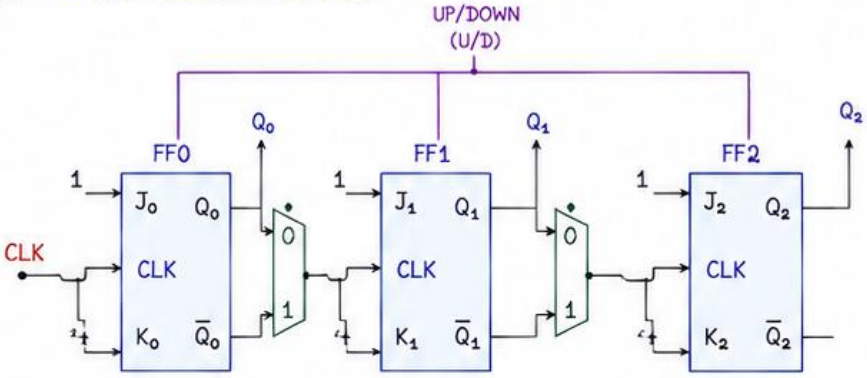
with Timing Diagram – Programmable Ripple Counter

Sequential Logic Circuits

★ Operation of 3-bit Ripple UP/DOWN Counter

- A 3-bit ripple counter uses three flip-flops (FF_0, FF_1, FF_2) to count through $2^3 = 8$ states.
- All flip-flops are JK flip-flops with J = K = 1 (toggle mode).
- UP/DOWN control input (U/D):
  - U/D = 1 → Up counting
  - U/D = 0 → Down counting
- When U/D = 1 (Up), each flip-flop is clocked by the Q output of the previous flip-flop.
- When U/D = 0 (Down), each flip-flop is clocked by the $\overline{Q}$ (complement) output of the previous flip-flop.

★ 3-bit Ripple UP/DOWN Counter Circuit



U/D = 1 (Up) → Select '1' input → Use Q of previous FF
U/D = 0 (Down) → Select '0' input → Use $\overline{Q}$ of previous FF

0 → $\overline{Q}$ (Down)
1 → Q (Up)

★ State Table (Up Count) – U/D = 1

Count (Decimal)	Q_2 (MSB)	Q_1	Q_0 (LSB)
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Then repeat to 0

★ State Table (Down Count) – U/D = 0

Count (Decimal)	Q_2 (MSB)	Q_1	Q_0 (LSB)
0	0	0	0
7	1	1	1
6	1	1	0
5	1	0	1
4	1	0	0
3	0	1	1
2	0	1	0
1	0	1	1

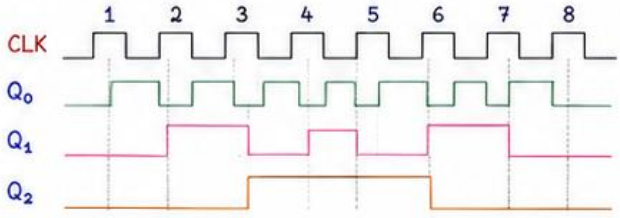
Then repeat to 0

Features

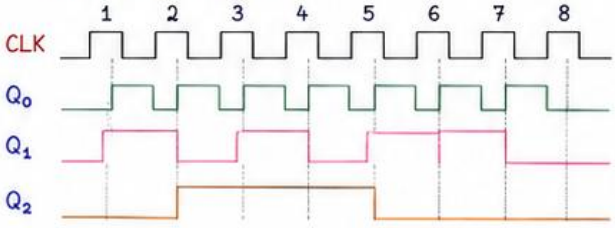
- Counts in both upward and downward direction.
- Simple design using **ripple** (asynchronous) configuration.
- Speed is limited by **ripple propagation** delay.

★ Timing Diagram

(a) UP Count (U/D = 1)

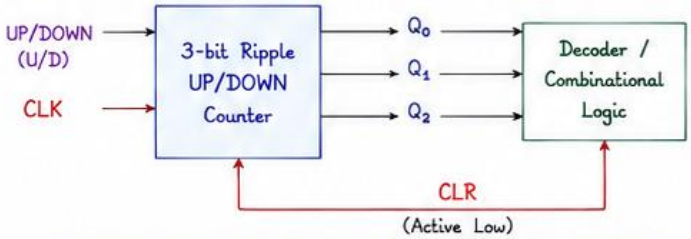


(b) DOWN Count (U/D = 0)



★ Programmable Ripple Counter

- A counter can be made programmable to count from any initial value to any final value.
- When the counter reaches the desired final count, it is automatically reset to the initial count.
- This is achieved using combinational decoding and clearing.



Example

To count from 2 (010) to 6 (110) in UP mode
- Load initial count = 2 (010)
- Decode final count = 6 (110) using a 3-input AND gate
- When 110 occurs, CLR = 0 → Counter resets to 010

★ Note

Ripple counters are easy to design but slower due to cumulative propagation delay.
Synchronous counters are preferred in high-speed applications.

Operation of 4-bit Ripple UP/DOWN Counter

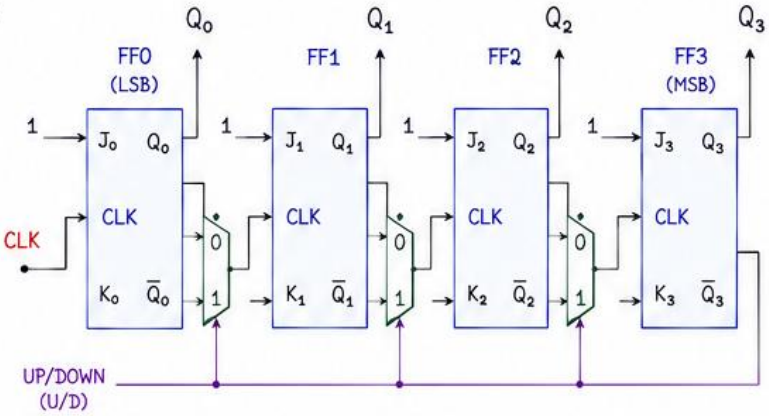
with Timing Diagram – Programmable Ripple Counter

Sequential Logic Circuits

★ Operation of 4-bit Ripple UP/DOWN Counter

- A 4-bit ripple counter uses four JK flip-flops (FF_0, FF_1, FF_2, FF_3) with J = K = 1 (toggle mode) to count through $2^4 = 16$ states.
- All flip-flops are triggered on the falling edge of the clock from the previous stage (ripple connection).
- UP/DOWN control input (U/D):
  - U/D = 1 → Up counting (use Q output of previous FF)
  - U/D = 0 → Down counting (use $\overline{Q}$ output of previous FF)
- When U/D = 1 (Up), each flip-flop is clocked by the Q output of the previous flip-flop.
- When U/D = 0 (Down), each flip-flop is clocked by the $\overline{Q}$ (complement) output of the previous flip-flop.

★ 4-bit Ripple UP/DOWN Counter Circuit



U/D = 1 (Up) → Select '1' input → Use Q of previous FF
U/D = 0 (Down) → Select '0' input → Use $\overline{Q}$ of previous FF

0 → $\overline{Q}$ (Down)
1 → Q (Up)

★ State Table (Up Count) – U/D = 1

Count (Decimal)	Q_3 (MSB)	Q_2	Q_1	Q_0 (Deimal)	Q_3 (MSB)	Q_2	Q_1	Q_0 (LSB)
0	0	0	0	8	1	0	0	0
1	0	0	0	9	1	0	0	1
2	0	0	1	10	1	0	1	0
3	0	0	1	11	1	0	1	1
4	0	1	0	12	1	1	0	0
5	0	1	0	13	1	1	0	1
6	0	1	1	14	1	1	1	0
7	0	1	1	15	1	1	1	1

Then repeat to 0.

★ State Table (Down Count) – U/D = 0

Count (Decimal)	Q_3 (MSB)	Q_2	Q_1	C_{0B} (Deinal)	Q_3 (MSB)	Q_1	Q_1 (LSB)
15	1	1	1	7	0	1	1
14	1	1	1	6	0	1	0
13	1	1	0	1	0	1	0
12	1	1	0	4	0	1	0
11	1	0	1	3	0	0	1
10	1	0	1	2	0	1	0
9	1	0	0	1	0	0	1
8	1	0	0	0	0	0	0

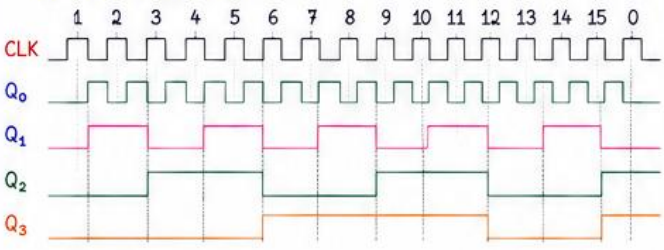
Then repeat to 15.

Features

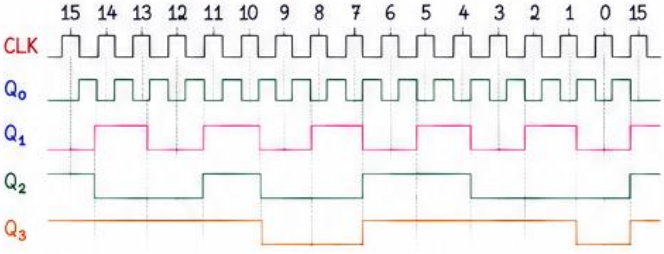
- Counts in both upward and downward direction.
- Simple ripple (asynchronous) design.
- Uses JK flip-flops in toggle mode.
- Speed is limited by ripple propagation delay.

★ Timing Diagram

(a) UP Count (U/D = 1)

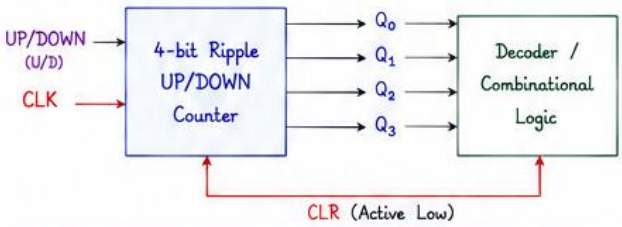


(b) DOWN Count (U/D = 0)



★ Programmable Ripple Counter

- A programmable counter can count from any initial value to any final value.
- When the counter reaches the desired final count, it is automatically reset to the initial count.
- This is achieved using combinational decoding and clearing.



Example

To count from 5 (0101) to 13 (1101) in UP mode

- Load initial count = 5 (0101)
- Decode final count = 13 (1101) using a 4-input AND gate
- When 1101 occurs ($Q_3Q_2Q_1Q_0$ = 1 1 0 1), CLR = 0 → Counter resets to 0101
- Counter will repeat the sequence 5, 6, 7, 8, 9, 10, 11, 12, 13, 5 ....

★ Note

Ripple counters are easy to design and require fewer gates, but they are slower due to cumulative propagation delay. Synchronous counters are preferred for high-speed applications.

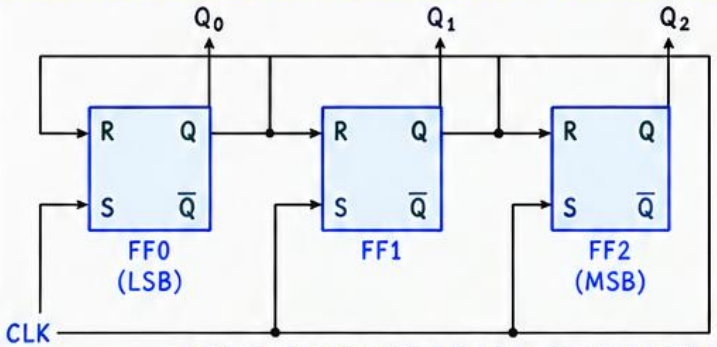
Design of Synchronous Counter with the help of RS Flip-Flop

Sequential Logic Circuits

★ Concept :

- A synchronous counter is a counter in which all flip-flops are triggered by the same clock pulse (common clock).
- Here, RS flip-flops are used to design a synchronous 3-bit binary counter (MOD-8).
- The counter counts in binary sequence from 000 to 111 and then repeats.
- All flip-flops receive the same clock signal, so the state changes simultaneously.

★ Circuit Diagram :



All flip-flops are triggered by the same clock pulse (synchronous).

★ Truth Table :

Present State (Q_2 Q_1 Q_0)	Next State (Q_2^+ Q_1^+ Q_0^+)	Input to RS Flip-Flops		
		FF2 (S_2 R_2)	FF1 (S_1 R_1)	FF0 (S_0 R_0)
0 0 0	0 0 1	0 0	0 0	1 0
0 0 1	0 1 0	0 0	1 0	0 0
0 1 0	0 1 1	0 0	1 0	0 0
0 1 1	1 0 0	1 0	0 1	0 0
1 0 0	1 0 1	0 0	0 0	1 0
1 0 1	1 1 0	0 0	0 0	1 0
1 1 0	1 1 1	0 0	0 0	1 0
1 1 1	0 0 0	1 1	0 1	0 1

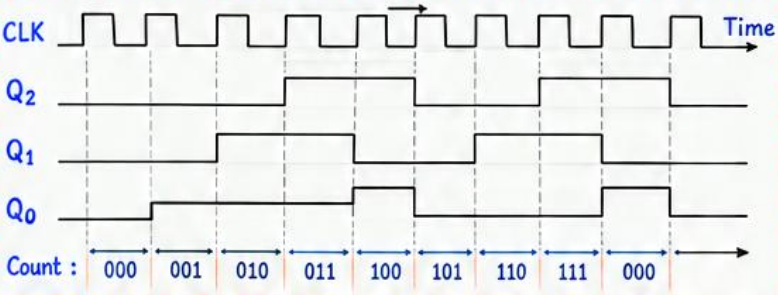
Total States = 8 (2^3)

★ Working Principle :

- The counter starts from 000.
- For each clock pulse, the next state is applied to the RS inputs according to the truth table.
- Since all flip-flops get the same clock pulse, the state changes simultaneously.
- The counter goes through 8 states (000 to 111) and then repeats.

Note :
The input combinations for RS flip-flops are based on the state transition table.

★ Output Waveform :



★ Key Points :

- Uses RS flip-flops.
- Synchronous counter (common clock).
- Counts in binary sequence (MOD-8).
- All flip-flops receive the same clock pulse.
- Can be extended for more bits (e.g., 4-bit, 5-bit, ...).

★ Summary / Comparison :

Feature	Description
Type	Synchronous counter
Flip-Flop	RS flip-flop
Number of Bits	3 (mod-8)
Clocking	Common clock for all flip-flops
Counting Sequence	000 → 001 → 010 → 011 → 100 → 101 → 110 → 111 → 000

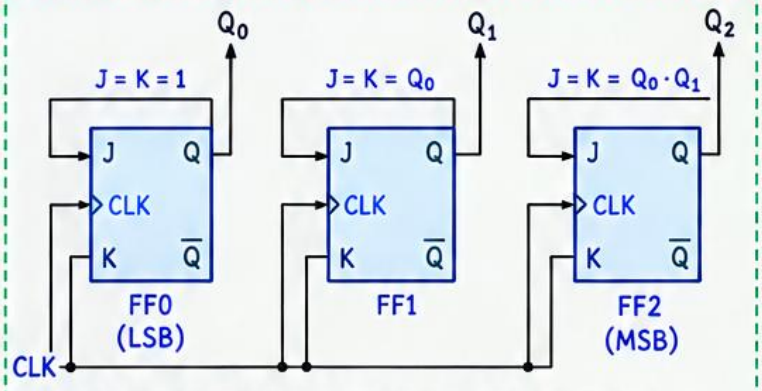
Design of Synchronous Counter with the help of JK Flip-Flop

Sequential Logic Circuits

★ Concept :

- A synchronous counter is a counter in which all flip-flops are triggered by the same clock pulse (common clock).
- Here, JK flip-flops are used to design a synchronous 3-bit binary counter (MOD-8).
- All flip-flops receive the same clock signal, so the state changes simultaneously.

★ Circuit Diagram :



All flip-flops are triggered by the same clock pulse (synchronous).

★ Truth Table (General) :

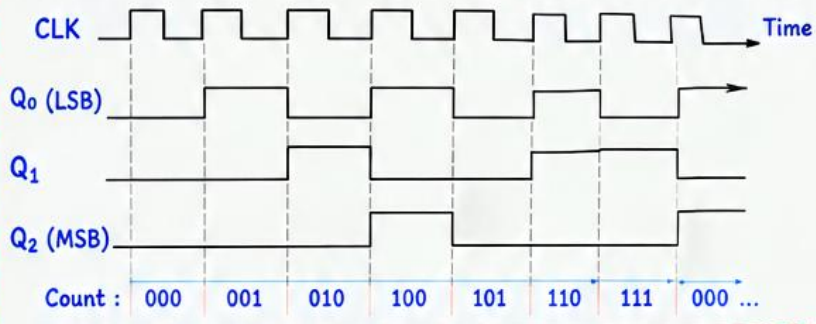
Present State (Q_2 Q_1 Q_0)	Next State (Q_{2+} Q_{1+} Q_{0+})	Input to JK Flip-Flops					
		J_0	K_0	J_1	K_1	J_2	K_2
0 0 0	0 0 1	1	0	1	0	1	0
0 0 1	0 1 0	1	0	1	0	1	0
0 1 0	0 1 1	1	0	1	0	1	0
0 1 1	1 0 0	1	1	0	1	1	0
1 0 0	1 0 1	1	0	1	0	0	1
1 0 1	1 1 0	1	0	1	0	0	1
1 1 0	1 1 1	1	0	0	1	0	1
1 1 1	0 0 0	1	0	0	1	0	1

Total States = $2^3 = 8$

★ Example : 3-bit Synchronous Binary Counter (MOD-8)

Present State (Q_2 Q_1 Q_0)	Next State (Q_{2+} Q_{1+} Q_{0+})
0 0 0	0 0 1
0 0 1	0 1 0
0 1 0	0 1 1
0 1 1	1 0 0
1 0 0	1 0 1
1 0 1	1 1 0
1 1 0	1 1 1
1 1 1	0 0 0

★ Output Waveform :



★ Key Points :

- Uses JK flip-flops.
- Synchronous counter (common clock).
- $J_0 = K_0 = 1$ (for LSB).
- $J_1 = K_1 = Q_0$.
- $J_2 = K_2 = Q_0 \cdot Q_1$.
- All flip-flops receive the same clock pulse.
- Can be extended for more bits (e.g., 4-bit, 5-bit, ...).

★ Summary / Comparison :

Feature	Description
Type	Synchronous counter
Flip-Flop	JK flip-flop
Number of Bits	3 (mod-8)
Clocking	Common clock for all flip-flops
Counting Sequence	000 → 001 → 010 → 011 → 100 → 101 → 110 → 111 → 000

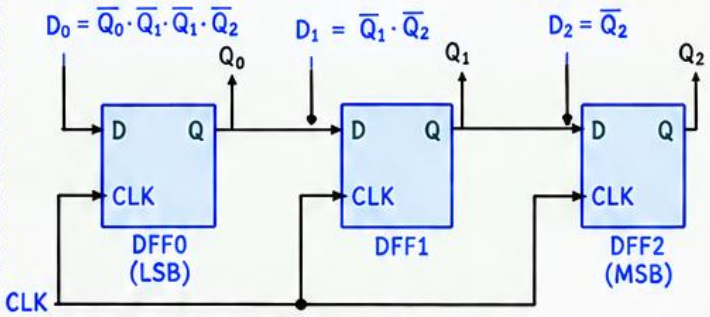
Design of Synchronous Counter with the help of D Flip-Flop

Sequential Logic Circuits

★ Concept :

- A synchronous counter is a counter in which all flip-flops are triggered by the same clock pulse (common clock).
- Here, D flip-flops are used to design a synchronous 3-bit binary counter (MOD-8).
- All flip-flops receive the same clock signal, so the state changes simultaneously.

★ Circuit Diagram :



All flip-flops are triggered by the same clock pulse (synchronous).

★ Truth Table (General) :

Present State (Q_2 Q_1 Q_0)	Next State (Q_2+ Q_1+ Q_0+)	Input to D Flip-Flops		
		D_2	D_1	D_0
0 0 0	0 0 1	0	0	1
0 0 1	0 1 0	0	1	0
0 1 0	0 1 1	0	1	1
0 1 1	1 0 0	1	0	0
1 0 0	1 0 1	1	0	1
1 0 1	1 1 0	1	1	0
1 1 0	1 1 1	1	1	1
1 1 1	0 0 0	0	0	0

Total States = $2^3 = 8$ (MOD-8)

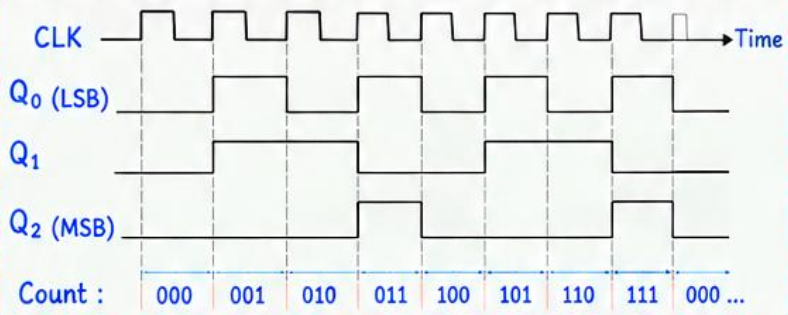
★ Input Logic for D Flip-Flops :

$$D_0 = \overline{Q_0} \cdot \overline{Q_1} \cdot \overline{Q_2}$$
$$D_1 = \overline{Q_1} \cdot \overline{Q_2}$$
$$D_2 = \overline{Q_2}$$

★ Example (First few states) :

Clock Pulse	Q_2 Q_1 Q_0 (Present)	Q_2 Q_1 Q_0 (Next)
0	0 0 0 →	0 0 1
1	0 0 1 →	0 1 0
2	0 1 0 →	0 1 1
3	0 1 1 →	1 0 0
...	... ...	... ...
7	1 1 1 →	0 0 0

★ Output Waveform :



★ Key Points :

- Uses D flip-flops.
- Synchronous counter (common clock).
- D_0, D_1, D_2 are derived from present state (Q_2, Q_1, Q_0).
- All flip-flops receive the same clock pulse.
- Can be extended for more bits (e.g., 4-bit, 5-bit, ...).

★ Summary / Comparison :

Feature	Description
Type	Synchronous counter
Flip-Flop	D flip-flop
Number of Bits	3 (mod-8)
Counting Sequence	000 → 001 → 010 → 011 → 100 → 101 → 110 → 111 → 000
Clocking	Common clock for all flip-flops
Input Logic	$D_0 = \overline{Q_0} \cdot \overline{Q_1} \cdot \overline{Q_2}$, $D_1 = \overline{Q_1} \cdot \overline{Q_2}$, $D_2 = \overline{Q_2}$

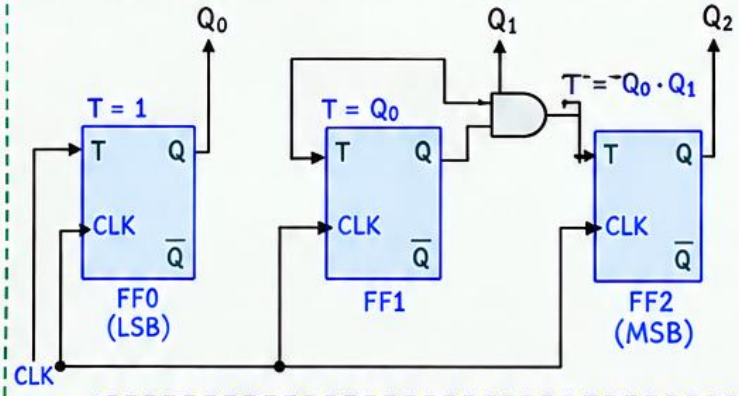
Design of Synchronous Counter with the help of T Flip-Flop

Sequential Logic Circuits

★ Concept :

- A synchronous counter is a counter in which all flip-flops are triggered by the same clock pulse (common clock).
- Here, T flip-flops are used to design a synchronous 3-bit binary counter (MOD-8).
- All flip-flops receive the same clock signal, so the state changes simultaneously.

★ Circuit Diagram :



All flip-flops are triggered by the same clock pulse (synchronous).

★ Truth Table (General) :

Present State (Q_2 Q_1 Q_0)	Next State (Q_{2+} Q_{1+} Q_{0+})	TInputs (T_2 T_1 T_0)
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 1 1
0 1 1	1 0 0	1 1 1
1 0 0	1 0 1	1 0 1
1 0 1	1 1 0	1 0 1
1 1 0	1 1 1	1 0 1
1 1 1	0 0 0	1 0 1

Total States = $2^3 = 8$ (MOD-8)

★ Input Logic for T Flip-Flops :

For a synchronous counter using T flip-flops :

$T_0 = 1$

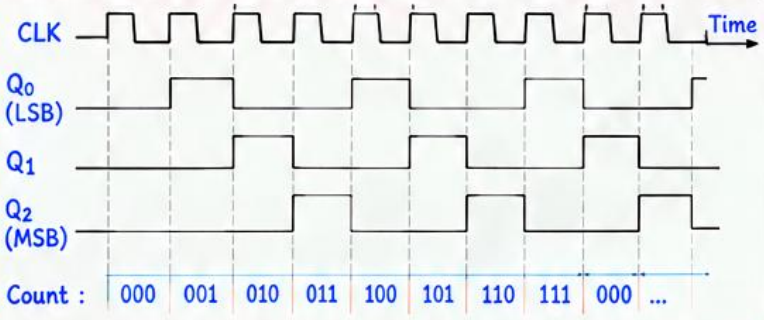
$T_1 = Q_0$

$T_2 = Q_0 \cdot Q_1$

★ Example (First few states) :

Clock Pulse	Q_2 Q_1 Q_0 (Present)	Q_2 Q_1 Q_0 (Next)
0	0 0 0 →	0 0 1
1	0 0 1 →	0 1 0
2	0 1 0 →	0 1 1
3	0 1 1 →	1 0 0
...	... ...	... ...
7	1 1 1 →	0 0 0

★ Output Waveform :



★ Key Points :

- Uses T flip-flops.
- Synchronous counter (common clock).
- $T_0 = 1$, $T_1 = Q_0$, $T_2 = Q_0 \cdot Q_1$.
- All flip-flops receive the same clock pulse.
- Can be extended for more bits (e.g., 4-bit, 5-bit, ...).

★ Summary / Comparison :

Feature	Description
Type	Synchronous counter
Flip-Flop	T flip-flop
Number of Bits	3 (mod-8)
Clocking	Common clock for all flip-flops
Input Logic	$T_0 = 1$, $T_1 = Q_0$, $T_2 = Q_0 \cdot Q_1$

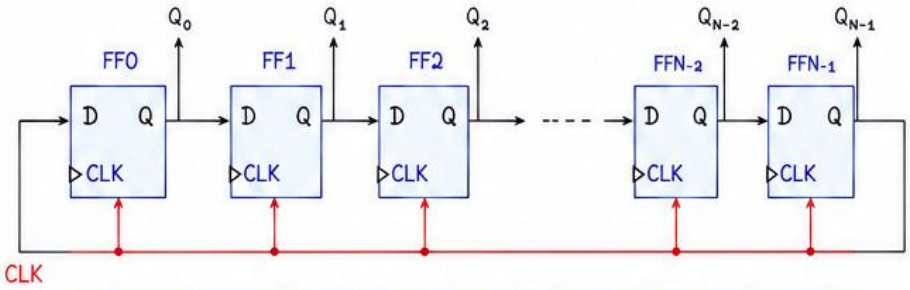
Design of Ring (N:1) Counter

Sequential Logic Circuits

★ Ring (N:1) Counter

- A Ring counter is a circular shift register in which a single '1' (logic high) circulates through N flip-flops.
- It is a MOD-N counter (has N states).
- It requires N flip-flops and N AND gates (to generate one-hot outputs).
- It is a type of self-starting counter.

★ Circuit Diagram of Ring (N:1) Counter (using D Flip-Flops)



The Q output of each flip-flop is connected to the D input of the next flip-flop.
The Q output of the last flip-flop is connected to the D input of the first flip-flop.

★ Operation

- Initially, one flip-flop is preset to 1 and all others are 0.
- On each clock pulse, the single '1' shifts to the next flip-flop in the ring.
- After N clock pulses, the '1' returns to the first flip-flop and the sequence repeats.
- Thus, it has N unique states.

★ Truth Table (General)

Present State					Next State				
Q_{N-1}	Q_{N-2}	...	Q_1	Q_0	Q'_{N-1}	Q'_{N-2}	...	Q'_1	Q'_0
1	0	0 ...	0	0	0	1	0 ...	0	0
0	1	0 ...	0	0	0	0	1 ...	0	0
0	0	1 ...	0	0	0	0	0 ...	1	0
:	:	: :	:	:	:	:	: :	:	:
0	0	0 ...	0	1	1	0	0 ...	0	0

(Total States = N)

★ Example: 4:1 Ring Counter

State Table

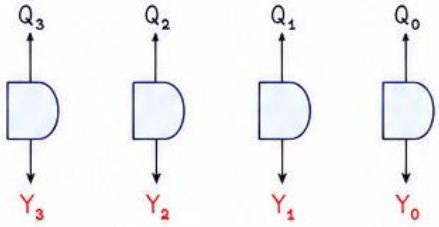
Clock	Count	Q_3	Q_2	Q_1	Q_0
Initial	0	1	0	0	0
1st Pulse	1	0	1	0	0
2nd Pulse	2	0	0	1	0
3rd Pulse	3	0	0	0	1
4th Pulse	4	1	0	0	0

(Sequence repeats after N = 4 states)

★ Outputs

- The counter is decoded so that only the stage holding '1' produces a high output.
- For N flip-flops, N decoded outputs are available.
- At any time, only one output is high (one-hot output).

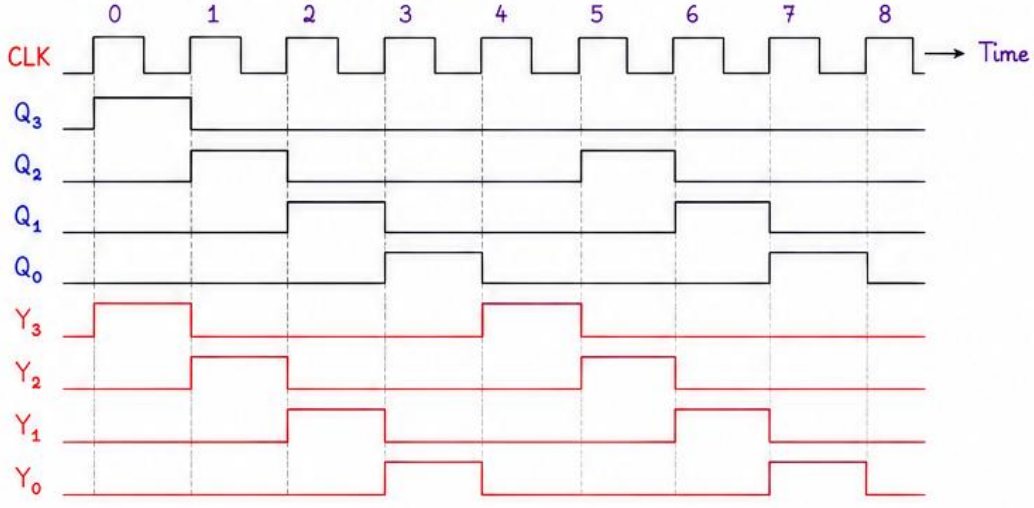
★ Decoding Logic (For 4:1 Ring Counter)



$Y_i = Q_i$ (i = 0 to 3)
i.e., Output is HIGH only when the corresponding flip-flop output is 1.

★ Waveform Diagram (4:1 Ring Counter)

(Assume Initial State: $Q_3 Q_2 Q_1 Q_0 = 1000$)



Key Points

- Ring counter is a MOD-N counter.
- It has N states.
- Only one flip-flop is HIGH at a time.
- It is self-starting.
- Used in sequence generation, timing signals, and control applications.

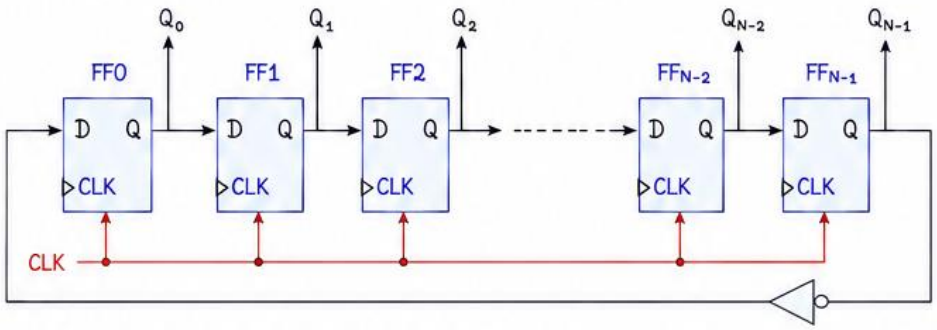
Johnson Counter (2N:1)

Sequential Logic Circuits

★ Johnson Counter (2N:1)

- Johnson counter is a modified ring counter.
- In Johnson counter, the inverse ($\overline{1}$) of the last flip-flop output is fed back to the input of the first flip-flop.
- It uses N flip-flops and has 2N unique states.
- It is a self-starting counter.

★ Circuit Diagram of Johnson Counter using N Flip-Flops



The inverse ($\overline{Q}_{N-1}$) of the last flip-flop output is connected to the D input of the first flip-flop.

★ Operation

- Initially, the counter is cleared (all outputs = 0).
- On each clock pulse, bits shift to the right.
- The complement of the last flip-flop output is fed back to the first flip-flop.
- After 2N clock pulses, the sequence repeats.
- It has 2N states.
- Example: For N = 4, total states = 2N = 8.

★ Truth Table (General)

Present State						Next State					
Q_{N-1}	Q_{N-2}	⋯	Q_2	Q_1	Q_0	Q'_{N-1}	Q'_{N-2}	⋯	Q'_2	Q'_1	Q'_0
0	0	⋯	0	0	0	0	0	⋯	0	0	0
0	0	⋯	0	0	1	0	0	⋯	0	0	0
0	0	⋯	0	1	1	0	0	⋯	0	0	1
0	0	⋯	1	1	1	0	0	⋯	0	1	1
:	:	:	:	:	:	:	:	:	:	:	:
1	1	⋯	1	1	0	1	1	⋯	1	0	0
1	1	⋯	1	0	0	1	1	⋯	0	0	0

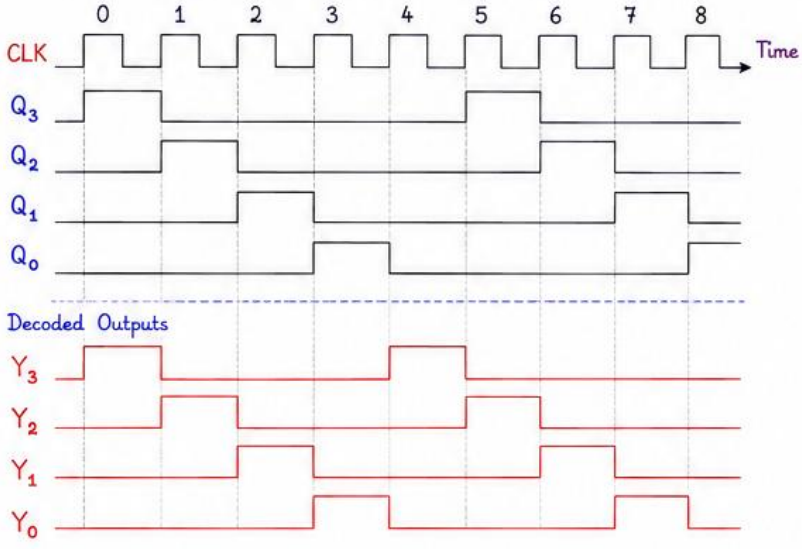
Total States = 2N

★ Example: 4-bit Johnson Counter (N = 4, 2N = 8 states)

Clock Pulse	Q_3	Q_2	Q_1	Q_0
Initial	0	0	0	0
1st	0	0	0	1
2nd	0	0	1	1
3rd	0	1	1	1
4th	1	1	1	0
5th	1	1	0	0
6th	1	0	0	0
7th	0	0	0	0
8th	0	0	0	0 (repeat)

★ Waveform Diagram (Example: 4-bit Johnson Counter)

Assume Initial State: $Q_3\ Q_2\ Q_1\ Q_0 = 0000$



At any time, either all outputs are 0, or the outputs contain a group of 1's followed by a group of 0's.

★ Key Points

- Johnson counter uses N flip-flops.
- It has 2N states.
- The complement of the last flip-flop output is fed back to the first flip-flop.
- It is self-starting.
- It is widely used in frequency division, sequence generation, and control circuits.

★ Comparison with Ring Counter

Feature	Ring Counter (N:1)	Johnson Counter (2N:1)
Number of States	N	2N
Feedback	Last Q to First D	Complement of Last Q to First D
Self-Starting	No	Yes
State Sequence	One-hot	Twisted Ring (Group of 1's & 0's)

Registers in Digital Electronics

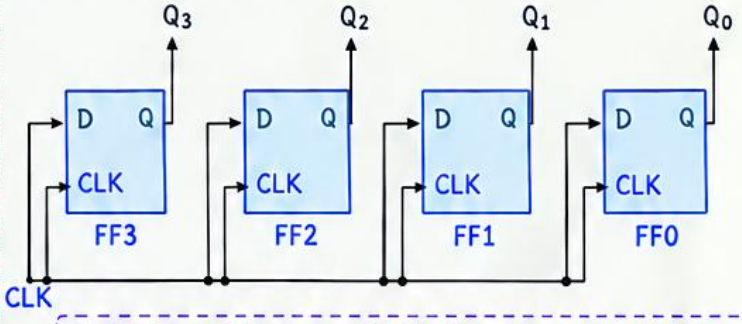
Sequential Logic Circuits

★ Definition :

A register is a group of flip-flops used to store and temporarily hold binary data.

Since each flip-flop can store 1 bit, an n-bit register requires n flip-flops.

★ Circuit Diagram (4-bit Register) :



A 4-bit register uses 4 flip-flops to store 4 bits of data (Q_3, Q_2, Q_1, Q_0).

★ Key Points :

- A register is a collection of flip-flops.
- Each flip-flop stores one binary bit (0 or 1).
- Registers are used for data storage and data transfer.
- Data can be shifted left or right in shift registers.
- Registers are important building blocks of digital systems, CPUs, counters and memory circuits.

★ Example :

A 4-bit register can store 4 binary bits :

1 0 1 1

It uses 4 flip-flops, with each flip-flop storing one bit :

1	0	1	1
FF3	FF2	FF1	FF0

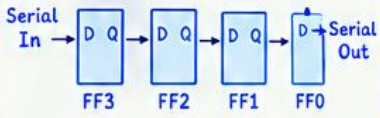
★ Types of Registers :

Common types include :

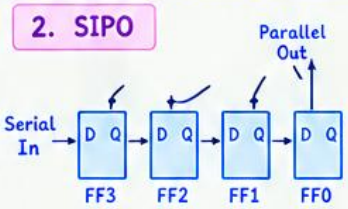
1. **SISO** (Serial-In Serial-Out)
   - Data enters and exits serially.
2. **SIPO** (Serial-In Parallel-Out)
   - Data enters serially, exits in parallel.
3. **PISO** (Parallel-In Serial-Out)
   - Data enters in parallel, exits serially.
4. **PIPO** (Parallel-In Parallel-Out)
   - Data enters and exits in parallel.

★ Types with Diagram :

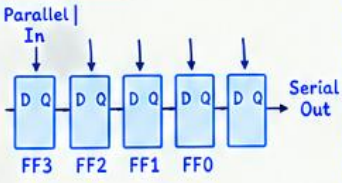
1. SISO



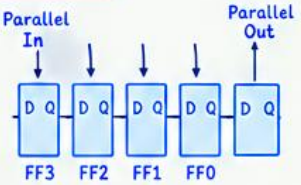
2. SIPO



3. PISO



4. PIPO



★ Applications :

- Used in data storage and transfer.
- Used in CPUs, microcontrollers and microprocesors.
- Used in counters, shift registers and memory circuits.
- Helps in temporary data holding and synchronization.

★ Summary :

A register is a group of flip-flops that stores binary information.

Register = n flip-flops = n bits.

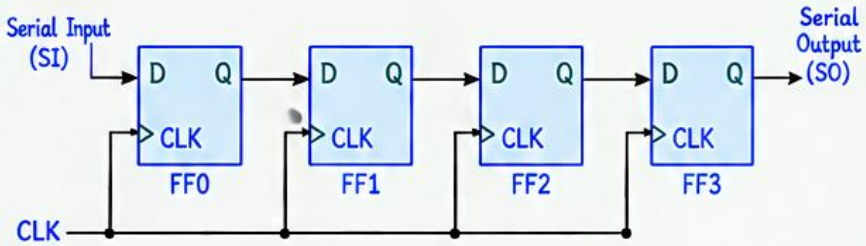
Serial In Serial Out (SISO) Shift Register
Operation with Circuit Diagram

Sequential Logic Circuits

★ Concept :

- A Serial In Serial Out (SISO) shift register takes input data in serial form and provides output data in serial form.
- It consists of a chain of flip-flops (usually D flip-flops) connected in series.
- With each clock pulse, the data moves from one flip-flop to the next.

★ Circuit Diagram :



Here, 4 D flip-flops are connected in series. The clock signal is common to all flip-flops.

★ Operation :

1. Initially, all flip-flops are cleared (Q = 0).
2. A serial input bit is applied to the first flip-flop (FF0).
3. On each clock pulse, the data moves one stage to the right.
4. After 4 clock pulses, the first input bit appears at the output (SO).
5. The next bits follow in the same manner.
6. Thus, the data is shifted out serially in the same order as it was applied.

★ Truth Table :

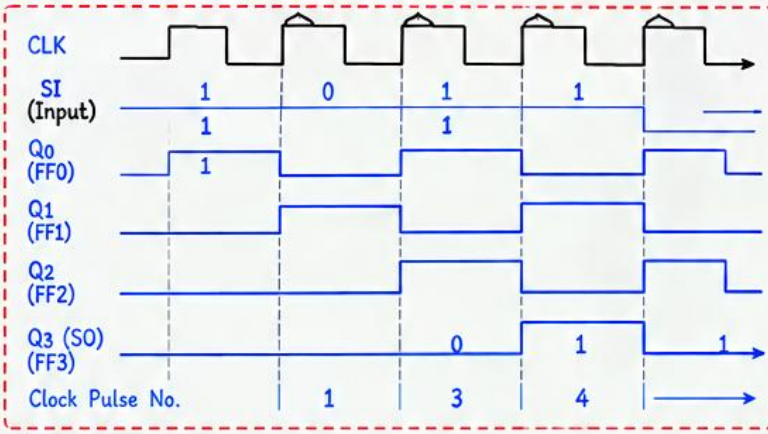
Clock Pulse	SI (Input)	Q_0	Q_1	Q_3	Q_3 (SO)
Initial	–	0	0	0	0
1	1	1	0	0	0
2	0	0	1	0	0
3	1	1	0	1	0
4	1	1	1	0	1

(Example: Input sequence = 1 0 1 1)

★ Note :

- The output appears after 4 clock pulses (for 4-bit register).
- The data moves from left to right.
- The output is delayed by the number of flip-flops.

★ Timing Diagram / Waveform :



★ Key Points :

- SISO means Serial In Serial Out.
- Consists of n D flip-flops.
- Data moves from left to right with each clock pulse.
- Output is available after n clock pulses.
- Used in serial data transmission, data storage and conversion.

★ Summary :

Feature	Description
Full Form	Serial In Serial Out (SISO) Shift Register
Flip-flops used	n D flip-flops (in series)
Input	Serial input (SI)
Output	Serial output (SO)
Data movement	Left to right (with clock pulses)
Clocking	Common clock for all flip-flops

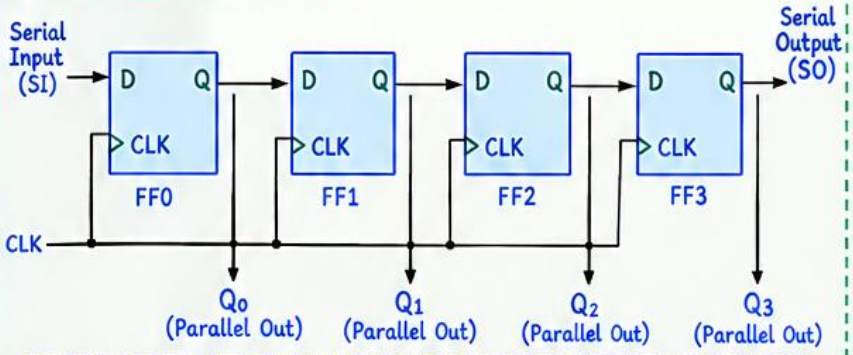
Serial In Parallel Out (SIPO) Shift Register Operation with Circuit Diagram

Sequential Logic Circuits

★ Concept :

- A Serial In Parallel Out (SIPO) shift register takes input data in serial form and provides output data in parallel form.
- It consists of a chain of flip-flops (usually D flip-flops) connected in series.
- After n clock pulses, the data appears at all the output lines simultaneously (in parallel).

★ Circuit Diagram :



Here, 4 D flip-flops are connected in series.
The Q outputs give parallel data, and the output can also be taken from the last flip-flop for serial output (optional).

★ Operation :

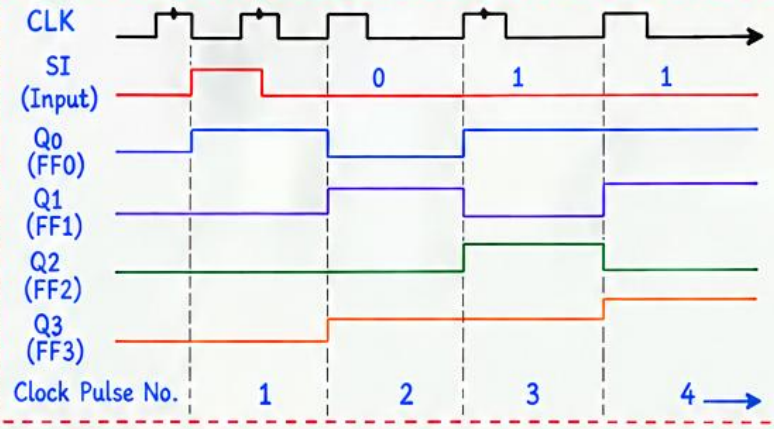
1. Initially, all flip-flops are cleared (Q = 0).
2. A serial input bit is applied to the first flip-flop (FF0).
3. On each clock pulse, the data moves one stage to the right.
4. After 4 clock pulses (for 4-bit register), the data appears at all the output lines simultaneously, (Q_0, Q_1, Q_2, Q_3).
5. The output can also be taken from the last flip-flop as serial output (SO).

★ Truth Table :

Clock Pulse	SI (Input)	Q_0 (FF0)	Q_1 (FF1)	Q_2 (FF2)	Q_3 (FF3)
Initial	-	0	0	0	0
1	1	1	0	0	0
2	0	0	1	0	0
3	1	1	0	1	0
4	1	1	1	0	1

(Example: Input sequence = 1 0 1 1)

★ Timing Diagram / Waveform :



★ Key Points :

- SIPO means Serial In Parallel Out.
- Input is serial, output is parallel.
- Consists of n D flip-flops.
- Data moves from left to right with each clock pulse.
- After n clock pulses, the data is available at all output lines simultaneously.
- Used in data transmission, ADC, DAC, memory systems, and communication systems.

★ Summary :

Feature	Description
Full Form	Serial In Parallel Out (SIPO) Shift Register
Flip-flops used	n D flip-flops (in series)
Input	Serial input (SI)
Output	Parallel output (Q_0, Q_1, Q_2, ..., Q_{n-1})
Data movement	Left to right (with clock pulses)
Clocking	Common clock for all flip-flops

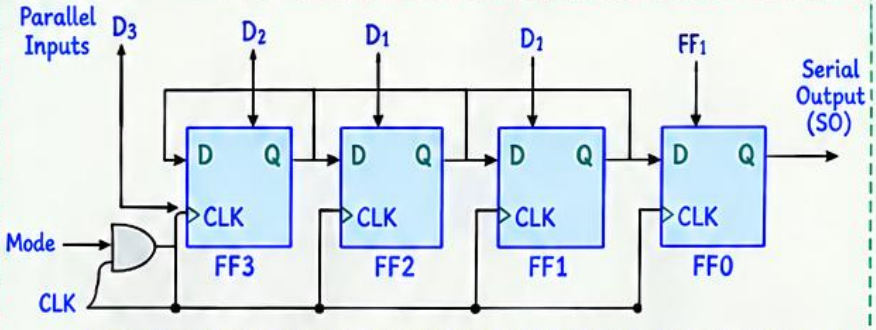
Parallel In Serial Out (PISO) Shift Register Operation with Circuit Diagram

Sequential Logic Circuits

★ Concept :

- A Parallel In Serial Out (PISO) shift register takes input data in parallel form and provides output data in serial form.
- It consists of a chain of flip-flops (usually D flip-flops) with a common clock.
- The data loaded in parallel is shifted out one bit at a time after a control signal is applied.

★ Circuit Diagram :



When Mode = 0 → Parallel Load (data enters in parallel).
When Mode = 1 → Shift (data shifts out serially).

★ Operation :

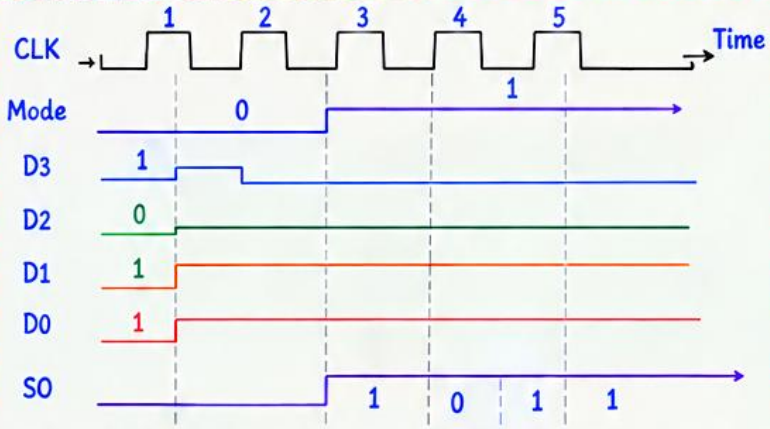
1. Initially, Mode = 0 (Parallel Load).
2. On the first clock pulse, the parallel input data (D_3, D_2, D_1, D_0) is loaded into the flip-flops.
3. Then, Mode is changed to 1 (Shift).
4. With each clock pulse, the data shifts one position to the right.
5. The last bit (D_0) appears at the serial output (SO) after 4 clock pulses.

★ Truth Table :

Clock Pulse	Mode	Parallel Input				Serial Output (SO)
		D_3	D_2	D_1	D_0	
Initial	–	Q_3	Q_2	Q_1	Q_0	–
1	0	D_3	D_2	D_1	D_0	–
2	1	–	–	–	–	D_3
3	1	–	–	–	–	D_2
4	1	–	–	–	–	D_1
5	1	–	–	–	–	D_0

(Example: Parallel input = 1 0 1 1 → Serial output = 1 0 1 1)

★ Timing Diagram / Waveform :



★ Key Points :

- PISO means Parallel In Serial Out.
- Consists of n D flip-flops.
- Parallel data is loaded when Mode = 0.
- Data shifts right when Mode = 1 (using clock pulses).
- Serial output appears at the last flip-flop (Q_0).
- Widely used in data transmission, ADC, DAC, memory systems and communication systems.

★ Summary :

Feature	Description
Full Form	Parallel In Serial Out (PISO) Shift Register
Flip-flops used	n D flip-flops (in series)
Input	Parallel input (D_3, D_2, D_1, D_0)
Output	Serial output (SO)
Data movement	Right shift (after loading)
Clocking	Common clock for all flip-flops

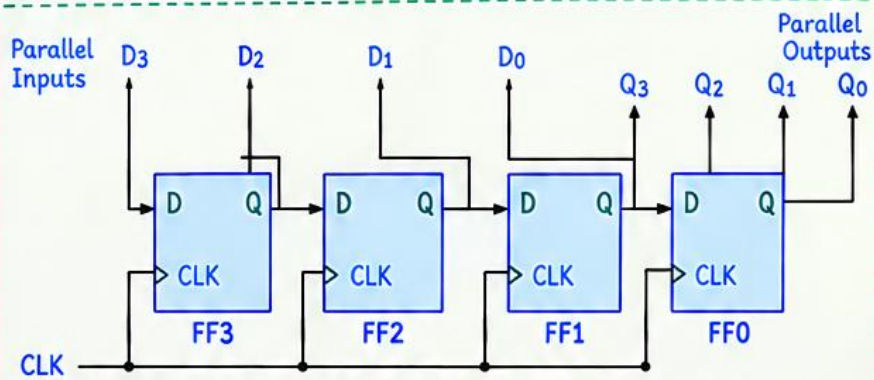
Parallel In Parallel Out (PIPO) Shift Register
Operation with Circuit Diagram

Sequential Logic Circuits

★ Concept :

- A Parallel In Parallel Out (PIPO) shift register takes input data in parallel form and provides output data in parallel form.
- It consists of a chain of flip-flops (usually D flip-flops) connected in parallel.
- All flip-flops receive the same clock pulse simultaneously.
- The data is transferred from input to output without shifting.

★ Circuit Diagram :



All flip-flops are connected in parallel and receive the same clock pulse. The input data is given in parallel and the output is obtained in parallel.

★ Operation :

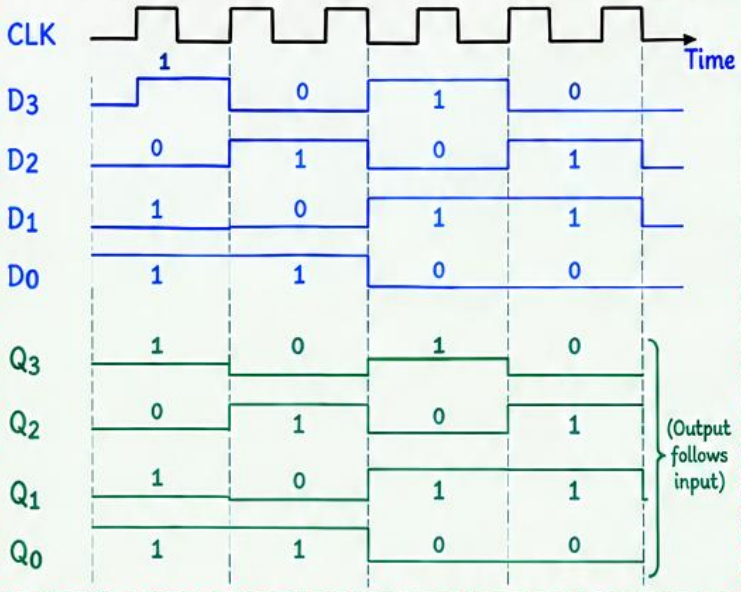
1. Initially, all flip-flops are cleared (Q = 0).
2. The parallel input data (D_3, D_2, D_1, D_0) is applied to the respective flip-flops.
3. On each clock pulse, all flip-flops load the input data simultaneously.
4. The same data appears at the output (Q_3, Q_2, Q_1, Q_0).
5. The data remains at the output until the next clock pulse with new data.

★ Truth Table :

Clock Pulse	Input (Parallel)				Output (Parallel)			
	D_3	D_2	D_1	D_0	Q_3	Q_2	Q_1	Q_0
Initial	-	-	-	-	0	0	0	0
1	1	0	1	1	1	0	1	1
2	0	1	0	1	0	1	0	1
3	1	1	1	0	1	1	1	0
4	0	0	1	0	0	0	1	0

(Example: Input sequence changes at each clock pulse)

★ Timing Diagram / Waveform :



★ Key Points :

- PIPO means Parallel In Parallel Out.
- Consists of n D flip-flops.
- All flip-flops receive the same clock pulse.
- Parallel data is loaded when clock pulse is applied.
- Output is available at the same time as input (no shifting).
- Widely used in data storage, microprocessors, registers and memory systems.

★ Summary :

Feature	Description
Full Form	Parallel In Parallel Out (PIPO) Shift Register
Flip-flops used	n D flip-flops (in parallel)
Input	Parallel input (D_3, D_2, D_1, D_0)
Output	Parallel output (Q_3, Q_2, Q_1, Q_0)
Data movement	No shifting (direct transfer)
Clocking	Common clock for all flip-flops