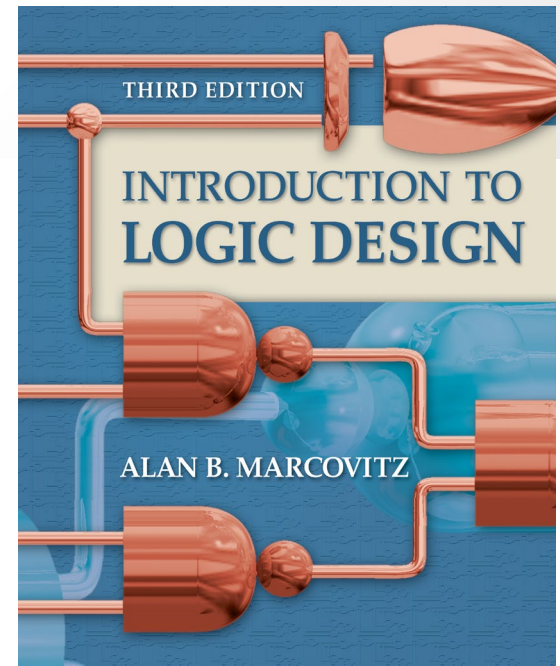


EE2000: Logic Circuit Design

Chapter 8

Acknowledgements: Dr. Wang Hong, Dr. Ray Cheung, Dr. Ricky Lau
Dr. Chang Chip Hong, Dr. Goh Wang Ling



Outline

10.1 Registers

- Registers

 - Parallel load, Bidirectional shift

 - Serial Adder, Accumulator

 - FIFO & FILO registers

10.2 Counters

- Asynchronous Counters

 - Ripple counters

- Synchronous Counters

 - Up-down counters,

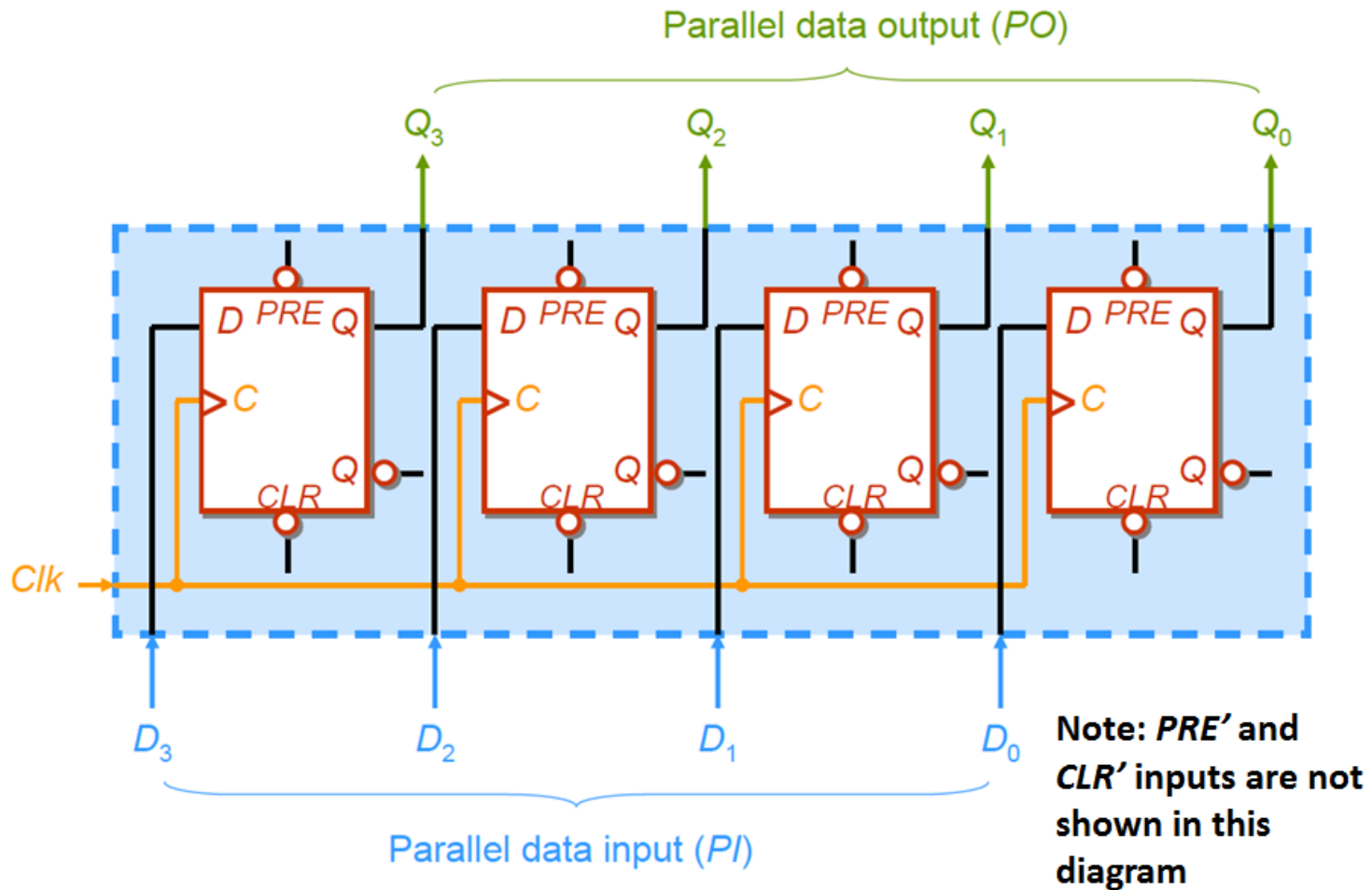
 - Modulo counters,

 - Ring counters, Twist Ring counters

10.1 Registers

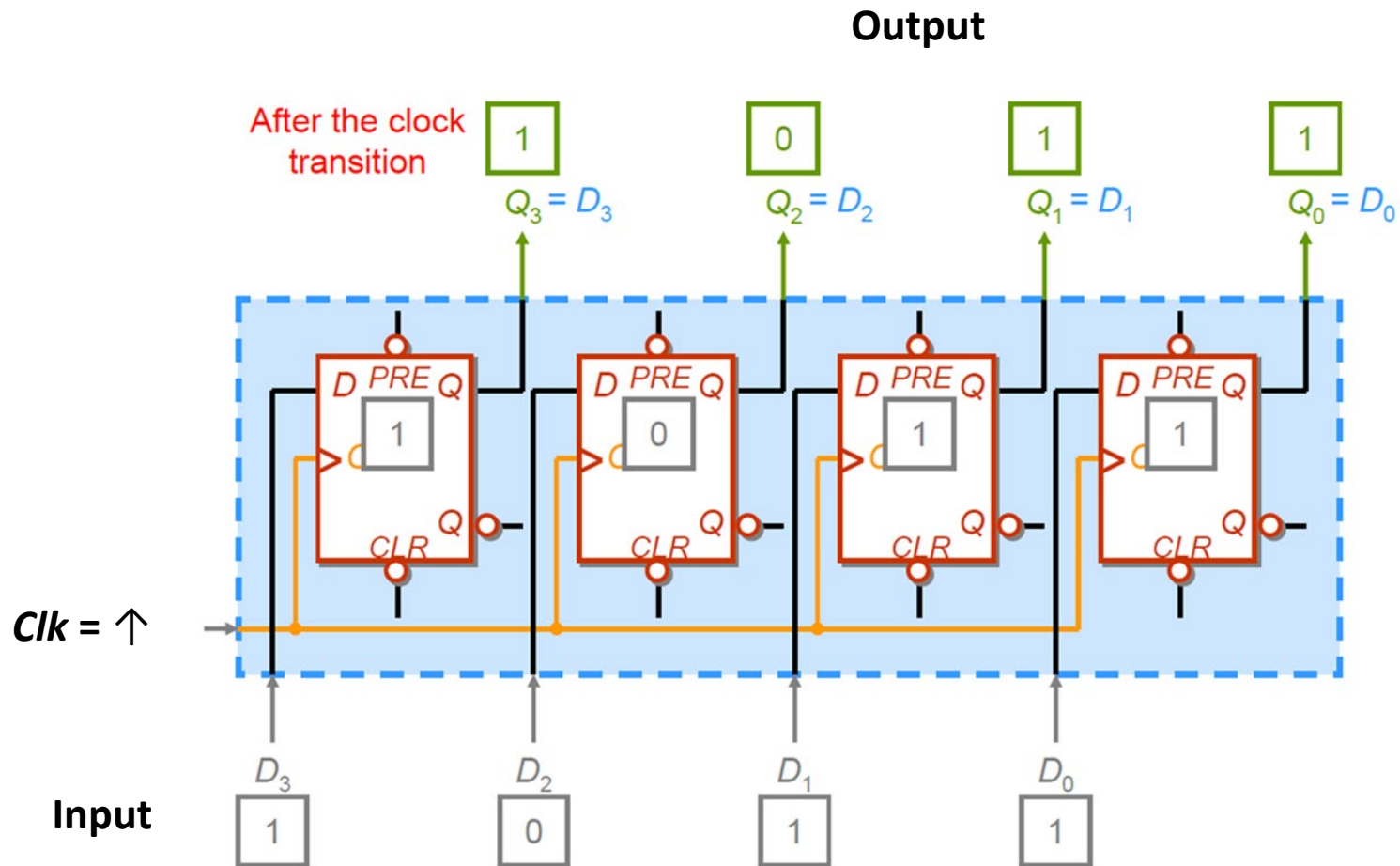
- Memory used for storing (binary) information during the processing of data **(used as buffer)**
- To store n -bit data, require n flip-flops
- n -bit register
 - Composed of a set of n edge-triggered flip-flops
 - Data will be latched to the flip-flops during the clock active period (pos. edge or neg. edge)
- Two designs
 - Parallel, or Serial

Parallel Registers



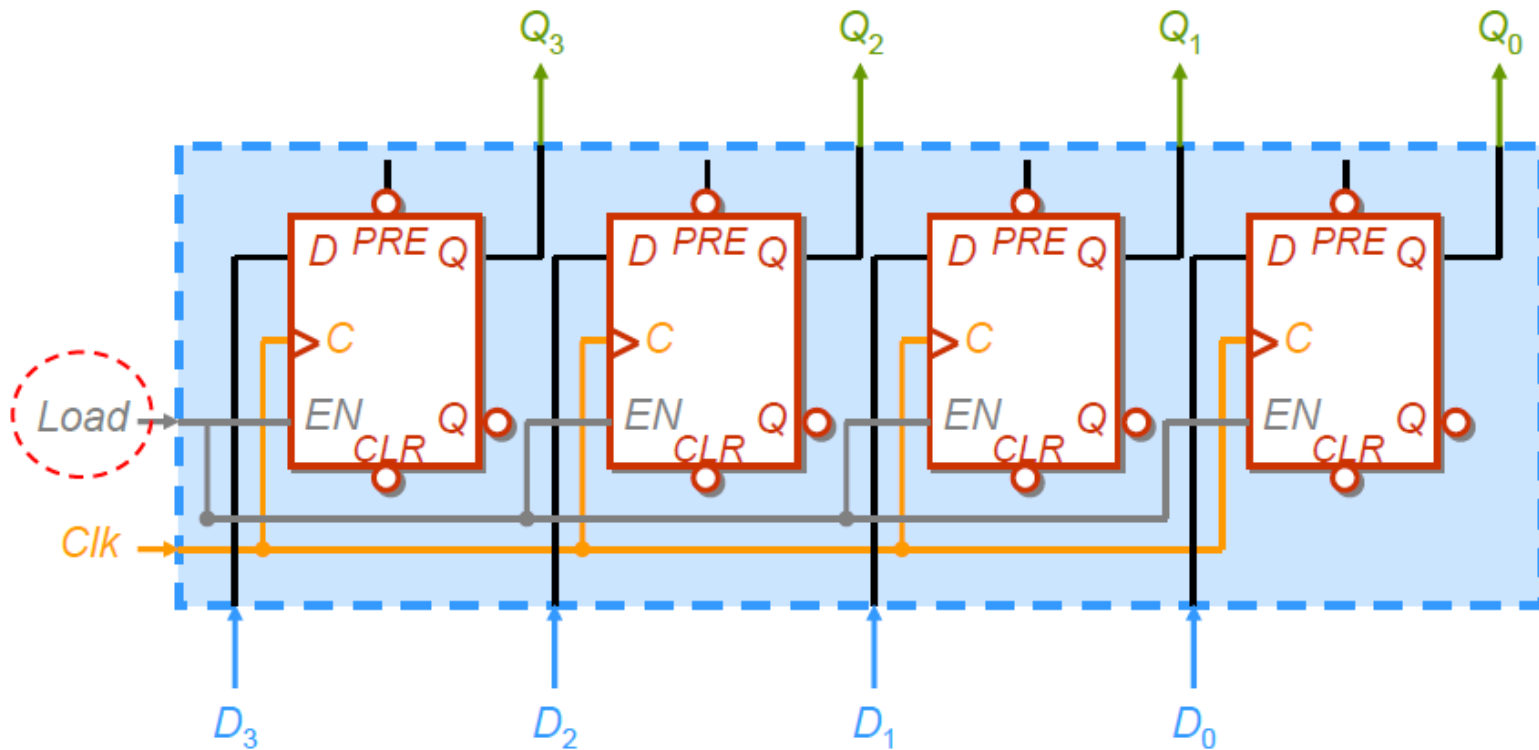
Multiple (n) inputs and multiple (n) outputs

Example



Note: PRE' and CLR' inputs are not shown in this diagram

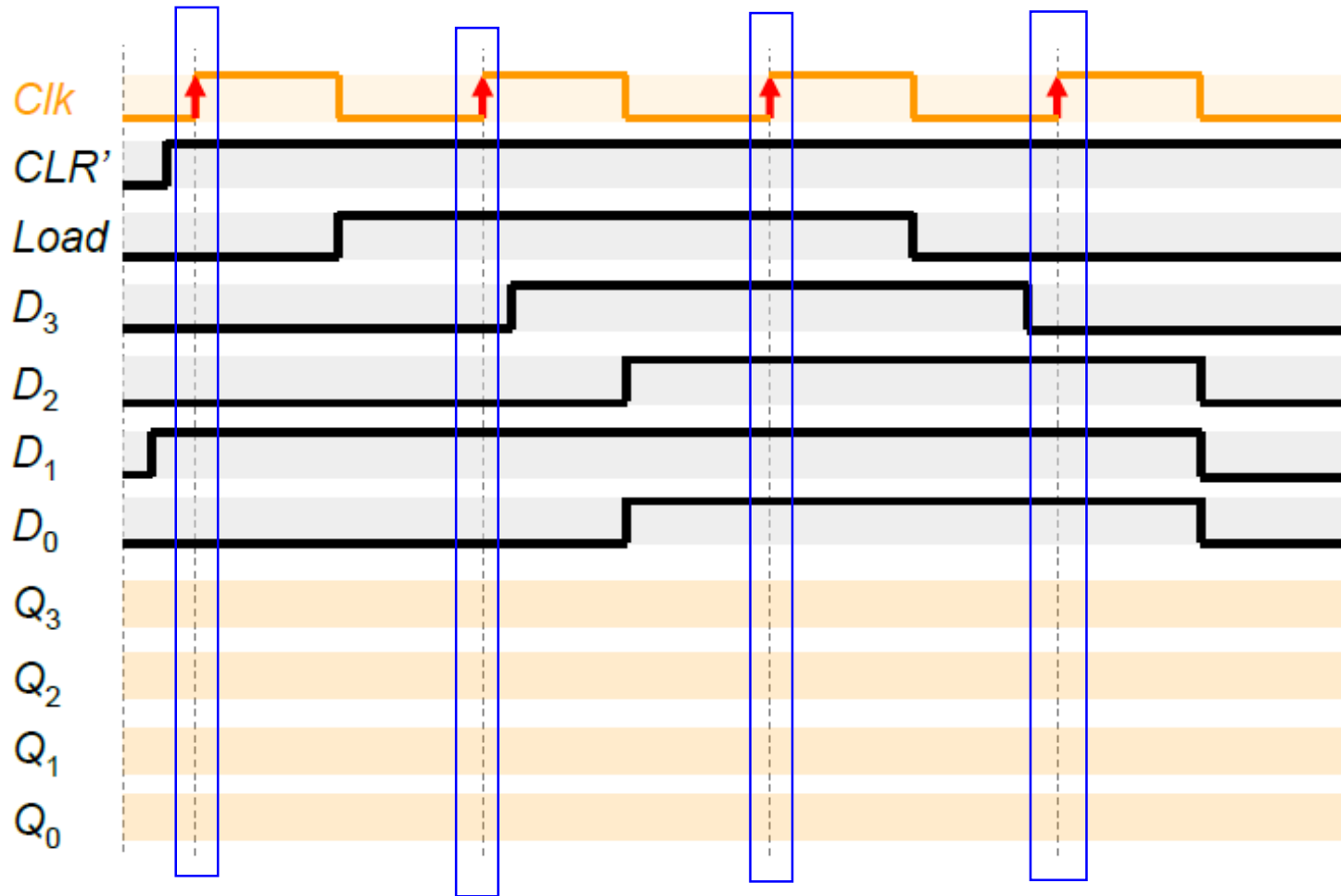
Register with Parallel Load



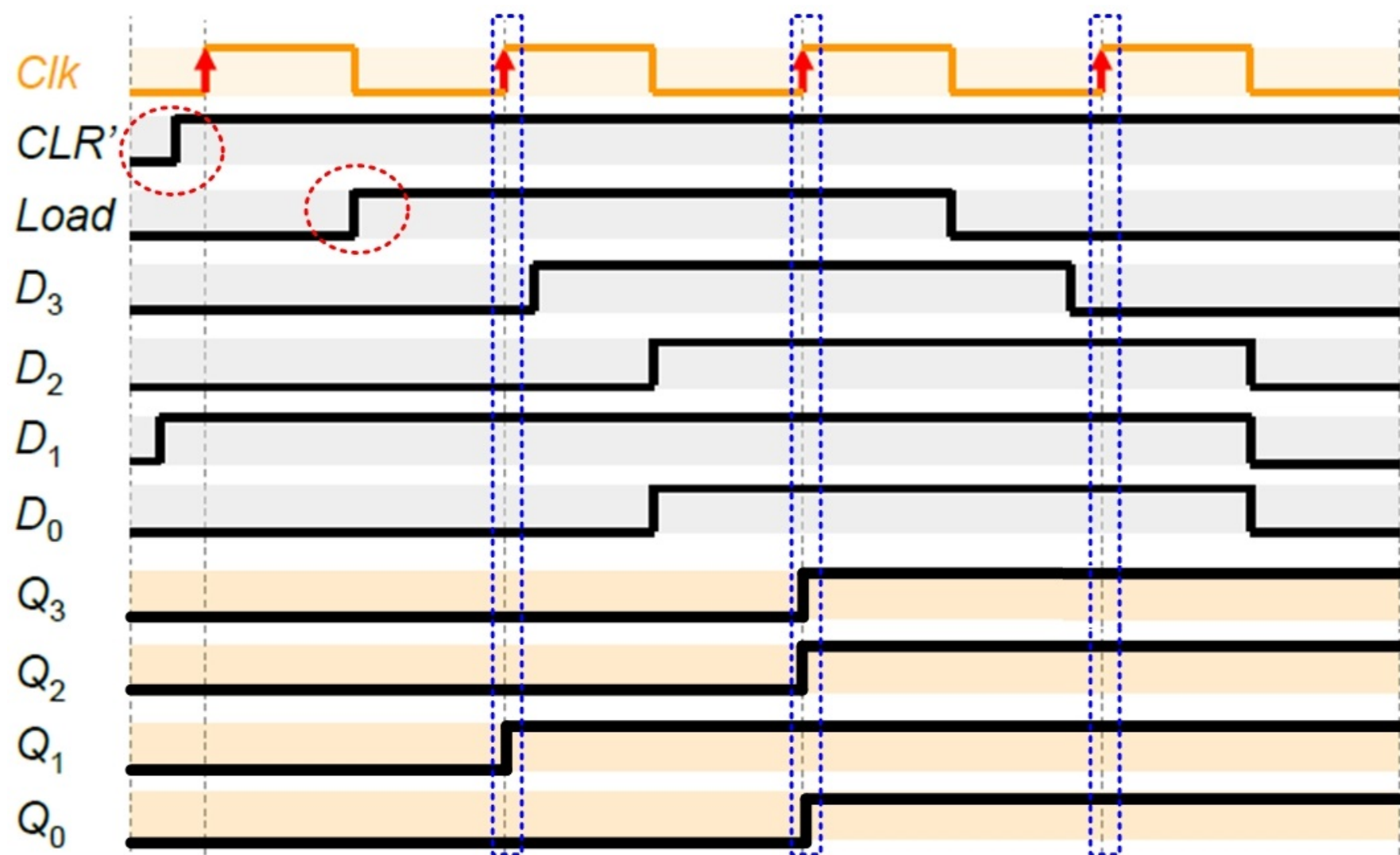
The register will load the inputs into the flip-flops only when "Clk is at the rising edge" & "Load = 1"

Note: PRE' & CLR' inputs are not drawn in this diagram

Timing Diagram- Register with Parallel Load



The D_i input must satisfy the set-up and hold time.

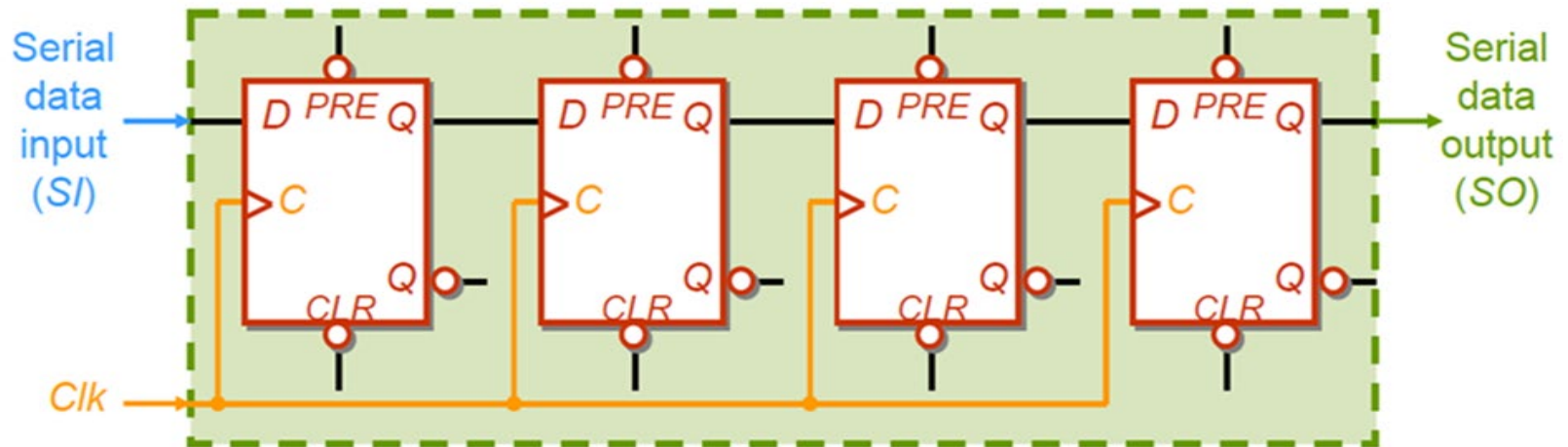


Serial Registers (Shift Registers)

- A chain of flip-flops
- When the flip-flops are triggered, the stored bits will be **shifted** to the next flip-flops
- A first-in-first-out (FIFO) structure
- Shift directions:
 - Right shift
 - Left shift
 - Bidirectional shift

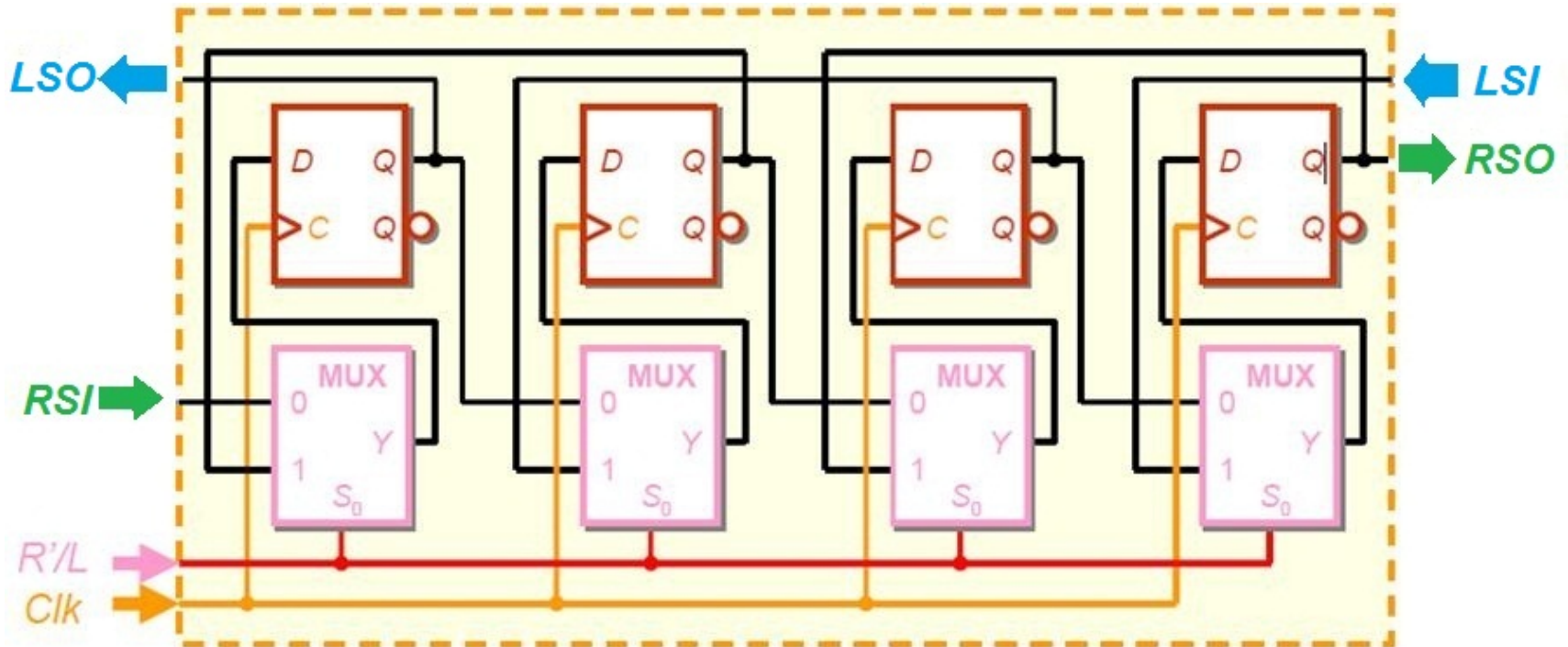
Serial Registers (Shift Registers)

A 4-bit Serial-In-Serial-Out Register



Single input and single output – Right shift operation

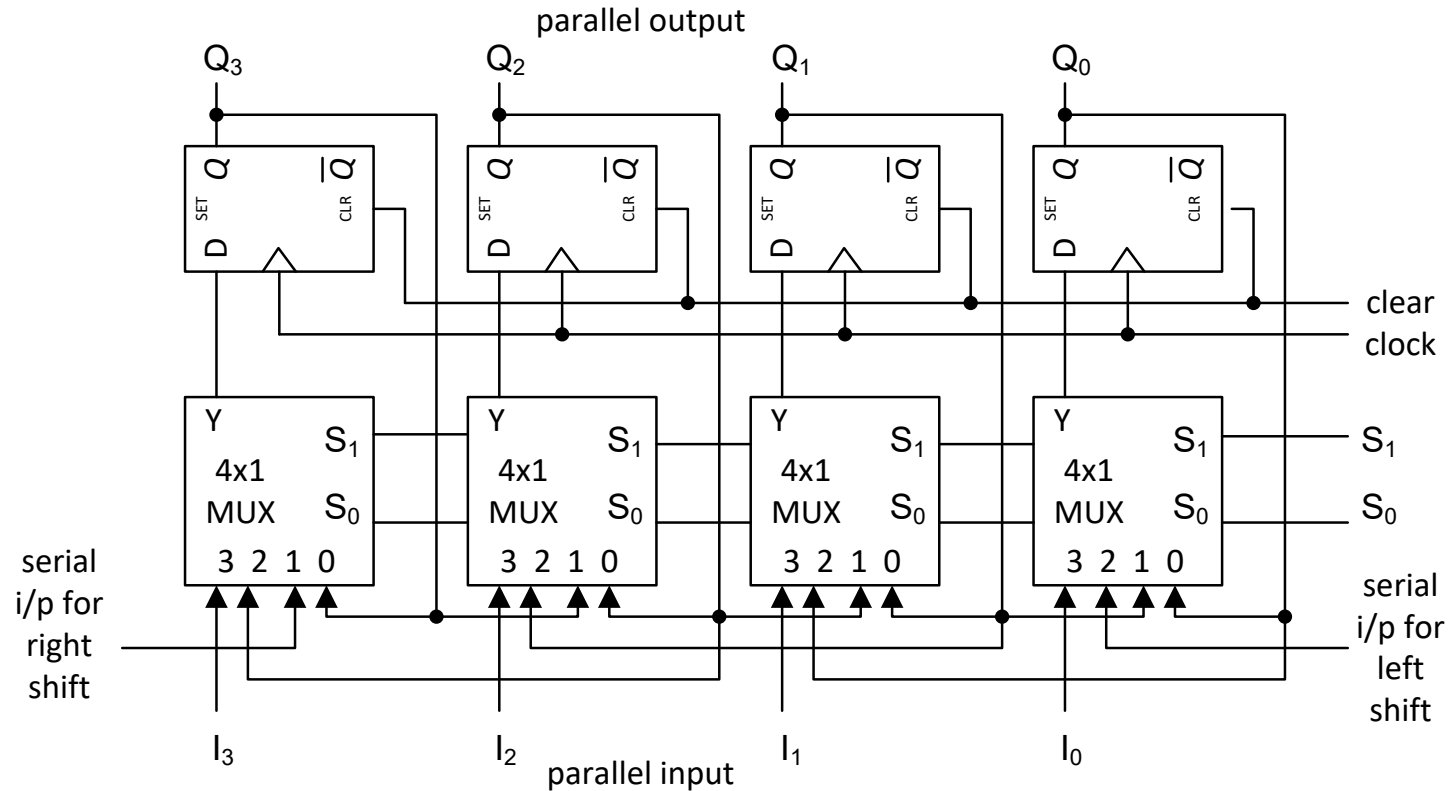
Bidirectional Shift Registers



Right shift for $R' = 0$

Left shift for $R' = 1$

Bidirectional Shift Register with Parallel Load

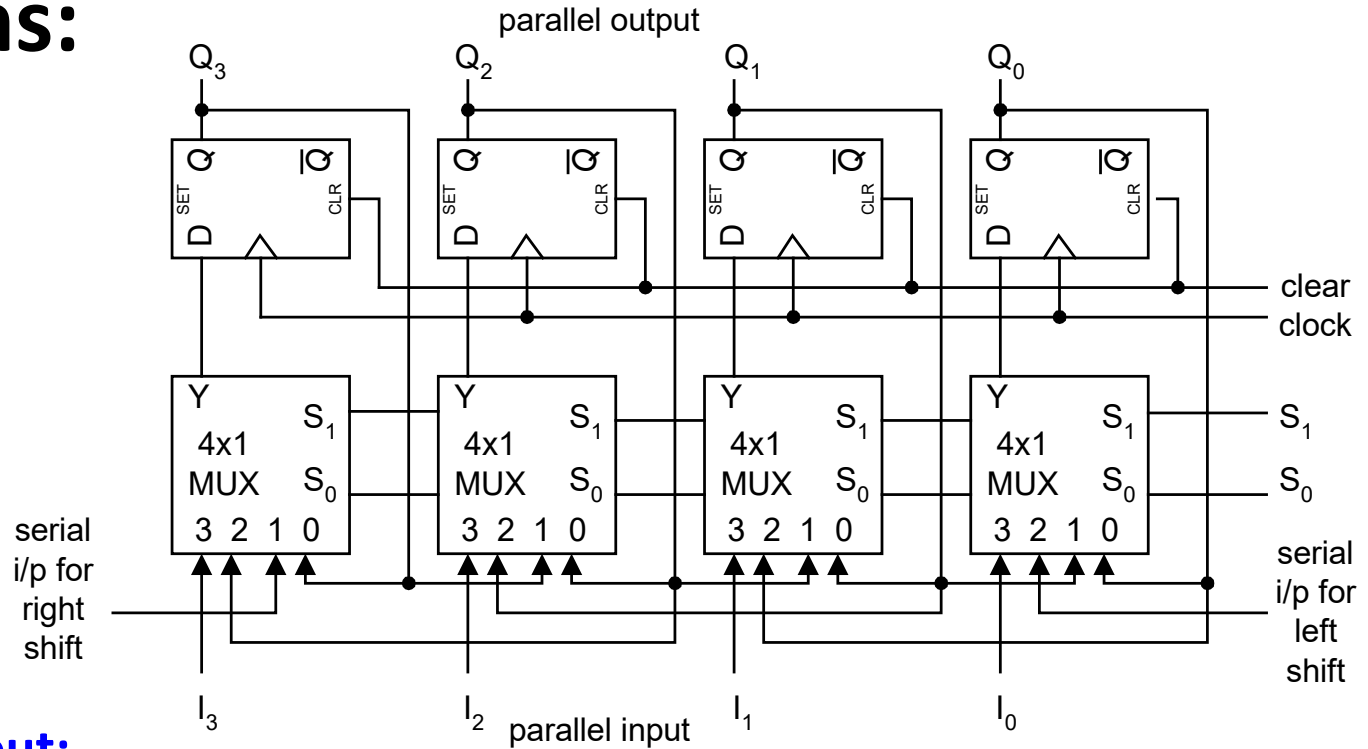


$\overline{\text{clear}}$	S ₁	S ₀	clock	Q _i	Description
L	X	X	X	0	Clear output
H	L	L	X	Q _i	Data hold (inhibit)
H	L	H	↑	Q _{i+1}	Shift right
H	H	L	↑	Q _{i-1}	Shift left
H	H	H	↑	I _i	Parallel load

Applications:

Data format conversion:

1. Parallel to serial
2. Serial to parallel



parallel-in-serial-out:-

1st step: issue “parallel load” control

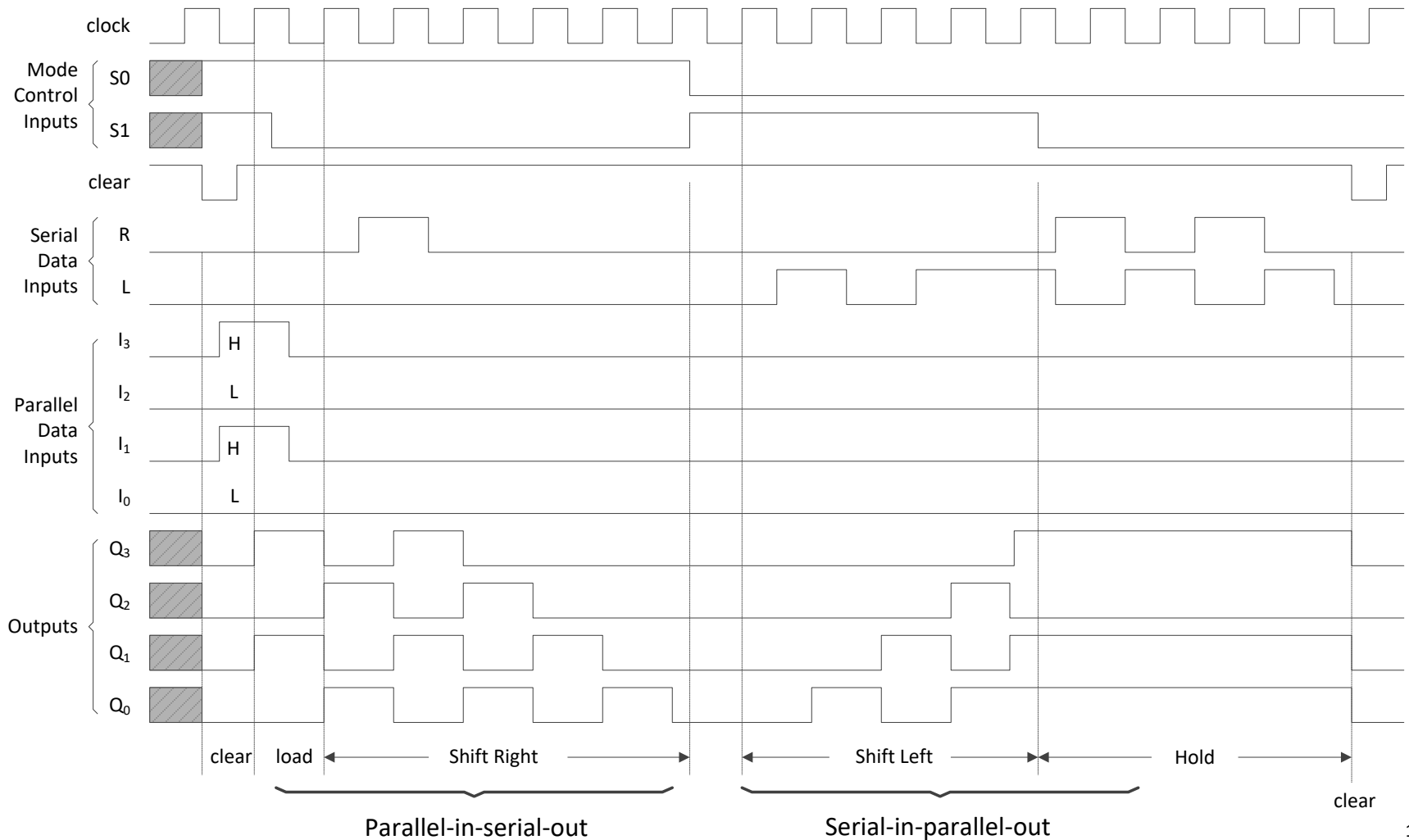
2nd step: issue “right shift” or “left shift” control

serial-in-parallel-out:-

1st step: issue “right shift” or “left shift” control for 4 clock pulses

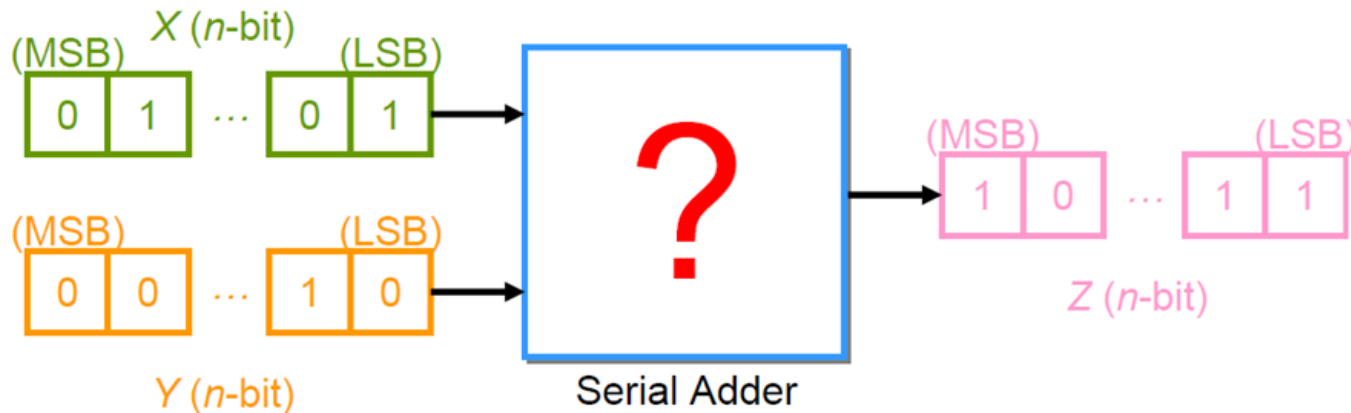
2nd step: “read” parallel output

Bidirectional Shift Register with Parallel Load – data format conversion



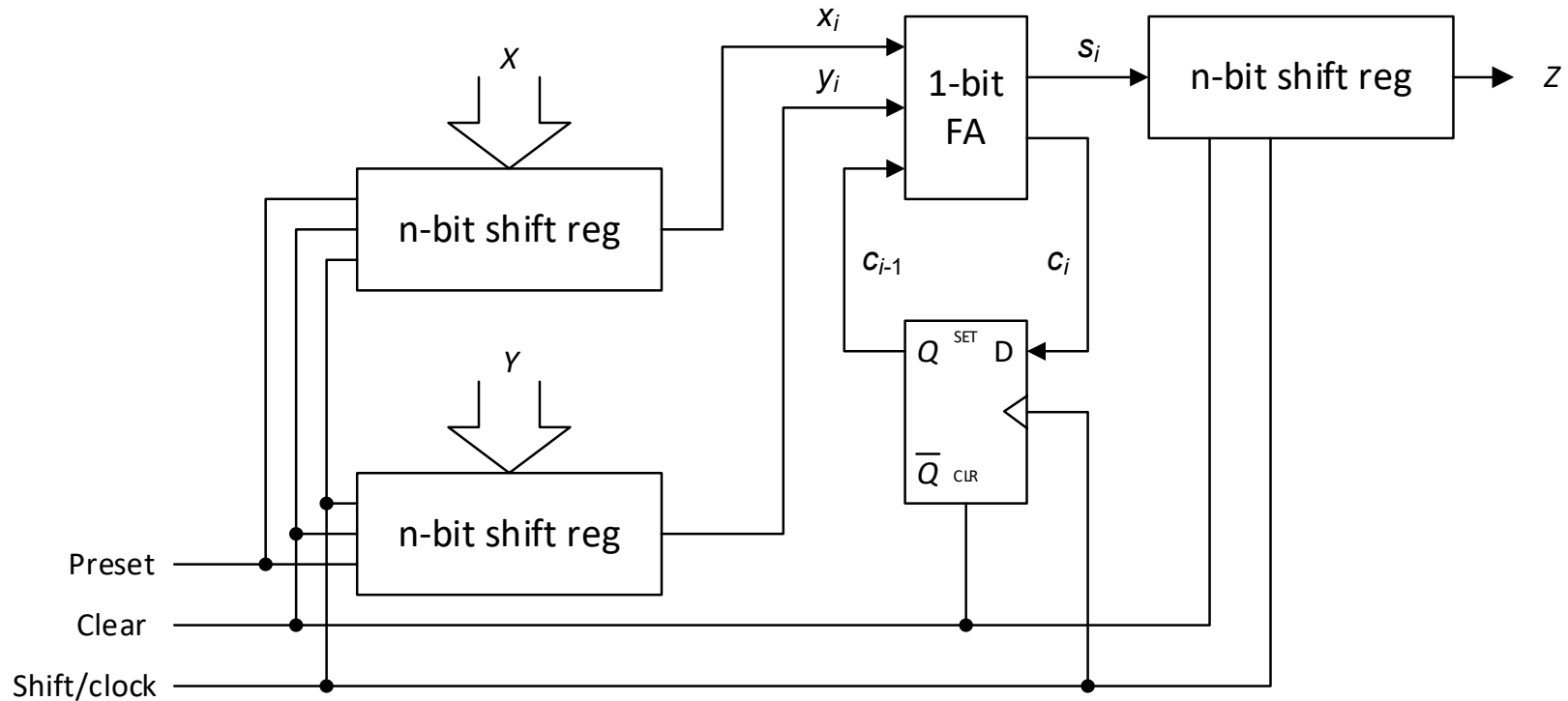
Application - Serial Binary Adder

- Perform addition sequentially one bit at a time starting from the LSB
- To perform *n*-bit addition with **one 1-bit FAs**, instead of *n* 1-bit FAs in parallel binary adder.
- Disadvantage – **time = $n \times \text{clock period}$**



Serial Binary Adder

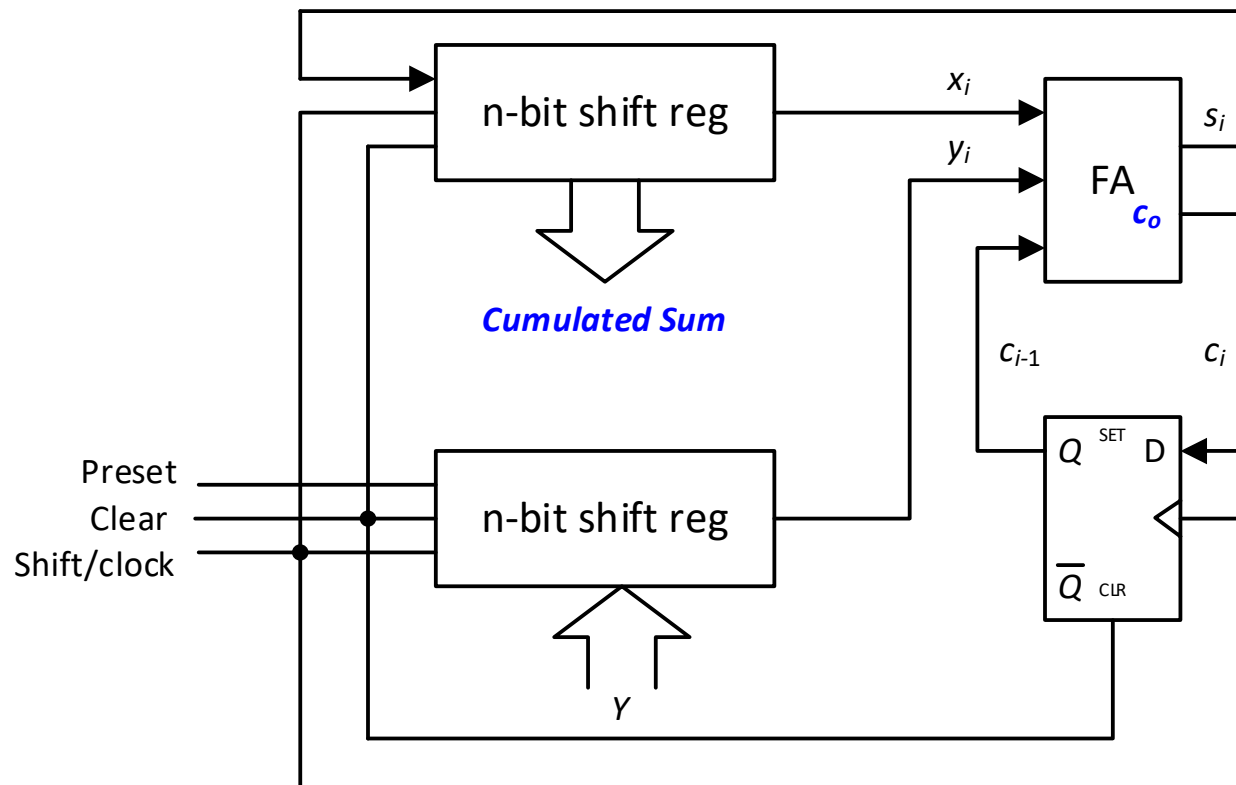
Only a 1-bit full adder is required.



The D-FF is used to hold the carry-out generated from previous bit addition by the 1-bit full adder.

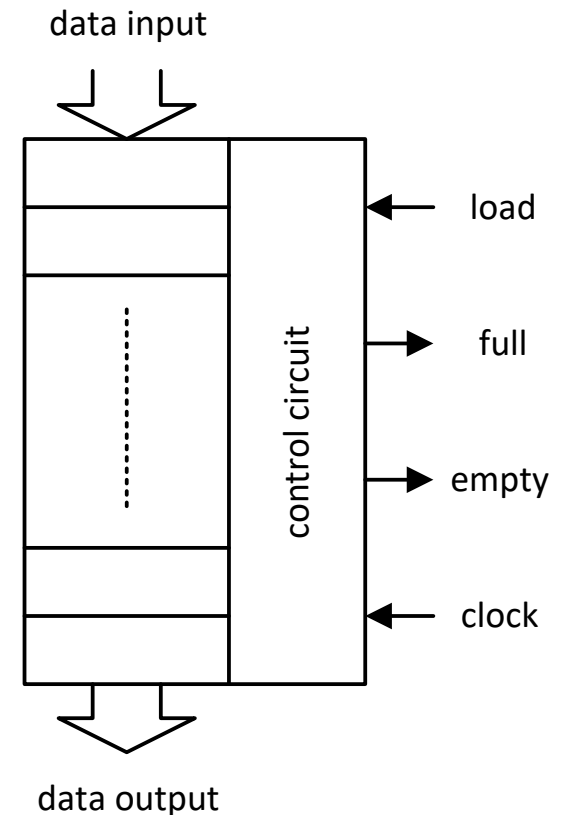
Serial Binary Accumulator

If the input of the adder, x_i , is connected to shift registers holding the cumulated sum, y_i will be added to the cumulated sum – the circuit becomes an **accumulator**.



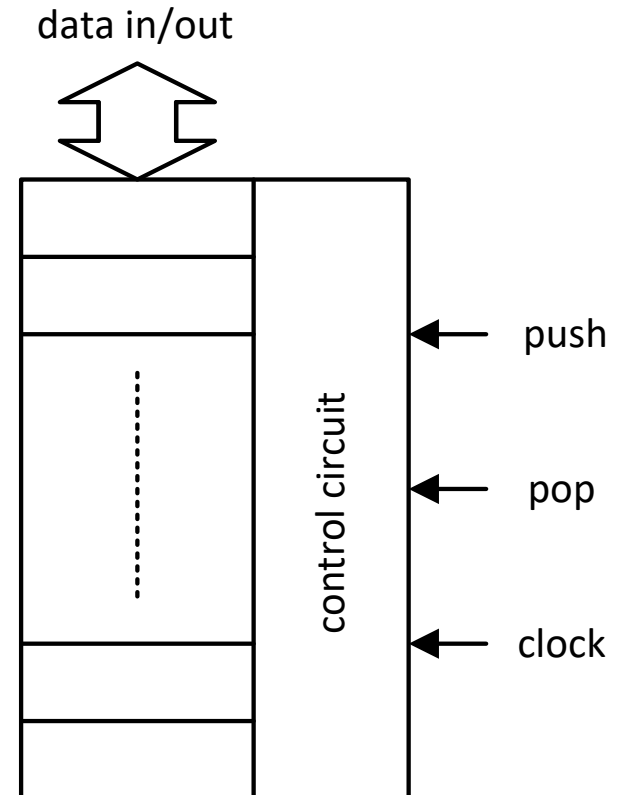
First-in-first-out register

- Abbreviated as FIFO.
- The first data goes into the register will be passed to the output first (just like a queue).
- It has two pins to indicate its fullness and emptiness.
- The operational speed is very fast and is normally used as buffer between a fast host system and **slow** peripheral (I/O) device to avoid data bus congestion.
- FIFO may be written into or read from its array at independent rates.



First-in-last-out register

- Abbreviated as FILO.
- The first data goes into the register will be passed to output last (just like a stack inside a CPU).
- It has two control signals, *push* and *pop*, for storing to and reading data from the register, respectively.
- The main function is used for temporary storage (in orderly sequence) and its main application is for serving interrupts inside a CPU.



Summary of 10.1

- Registers used to store a group of binary bits (known as word)
- A n -bit register can store a n -bit word
- Functions of shift registers
 - Hold (00)
 - Parallel load (11)
 - Serial input / output, left / right shift
 - Parallel-to-serial, serial-to-parallel conversion

10.2 Counters

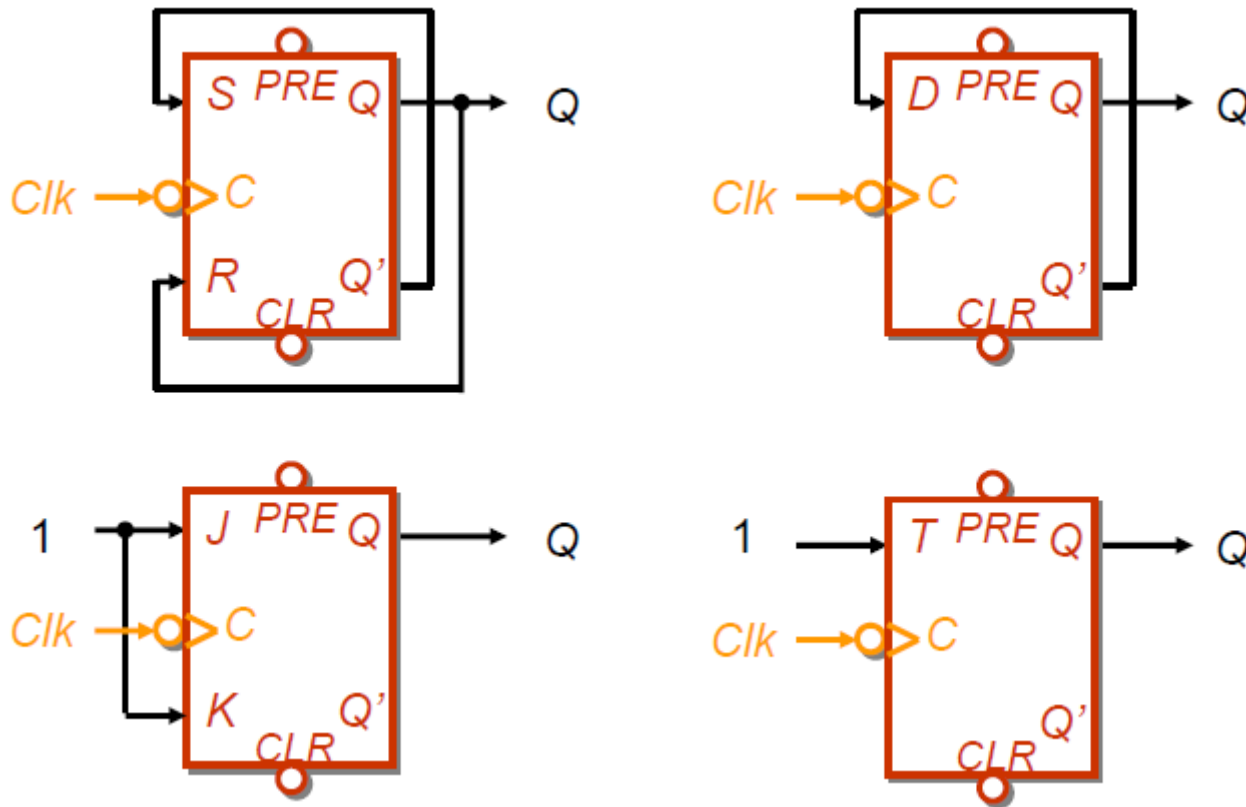
- What is a counter?
 - A special type of register
 - Goes through a predetermined sequence of states
 - Used to sequence and control operations in a circuit

Counters

- Two classes of counters
 - Asynchronous (binary) counters
 - Synchronous (binary) counters
- Asynchronous counters
 - e.g. ripple counters: the flip-flops are triggered by the **transition of other flip-flops**
- Synchronous Counters
 - All flip-flops are triggered by a **common clock**

Asynchronous Counters

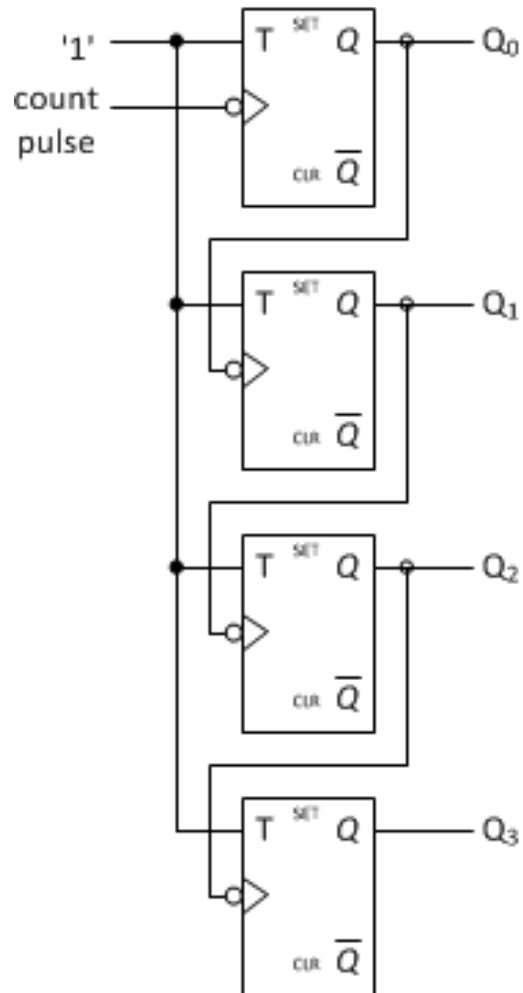
- How to toggle the state of different FFs ?



Asynchronous binary counter

- An asynchronous counter is also called ripple counter.
- In a ripple counter, the FF output transition serves as a source for triggering other FFs.
- In other words, the clock inputs of all FFs (except the first) are triggered not by the incoming pulse, but rather by the transition that occurs in other FFs.

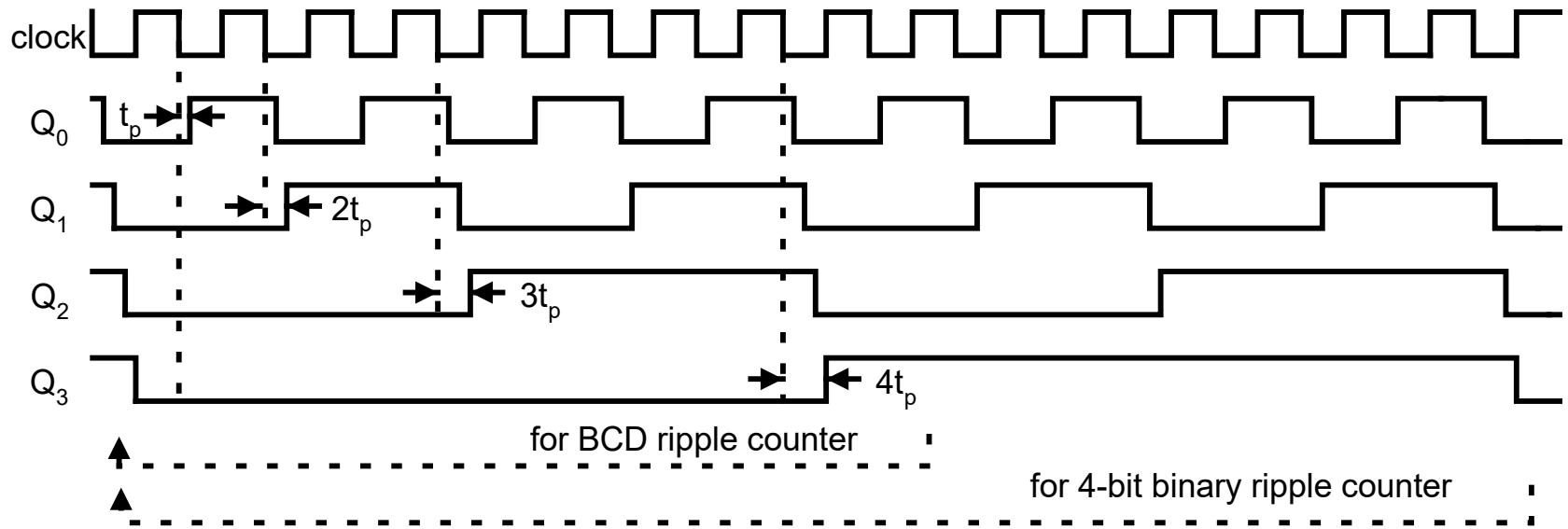
4-bit Ripple Counter (Async.)



Upward Counting Sequence			
Q ₃	Q ₂	Q ₁	Q 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
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

$CLR' = 0$
 Triggered at the falling edge (i.e. the moment that previous FF's Q falling from 1 to 0)

4-bit Ripple Counter (Async.)

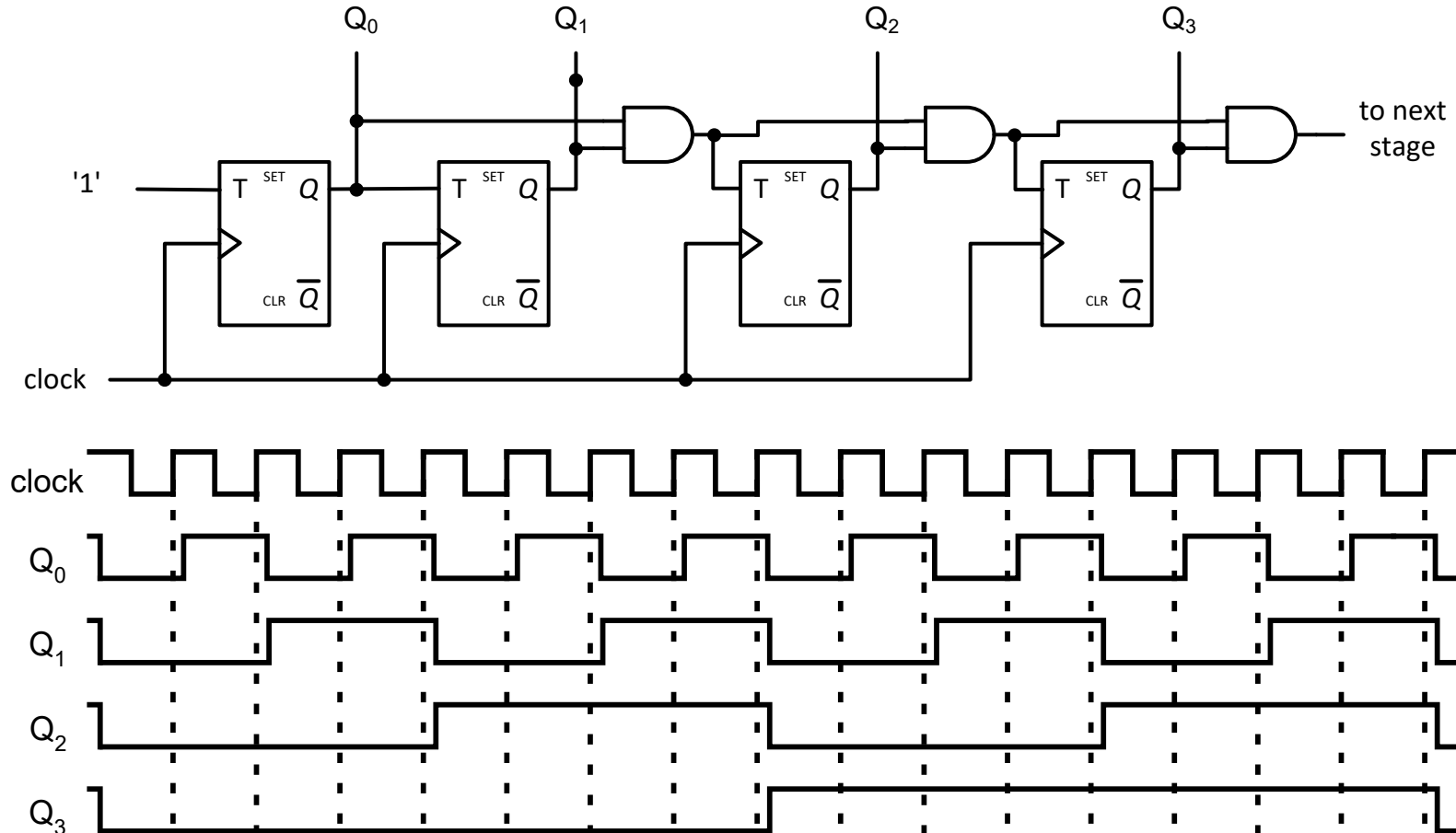


- The change of states does not occur at the same time
- Propagation delay of the flip-flops will ripple through the “clock input” of FFs
- The Q_i of the last flip-flop has the largest delay

Synchronous Counter

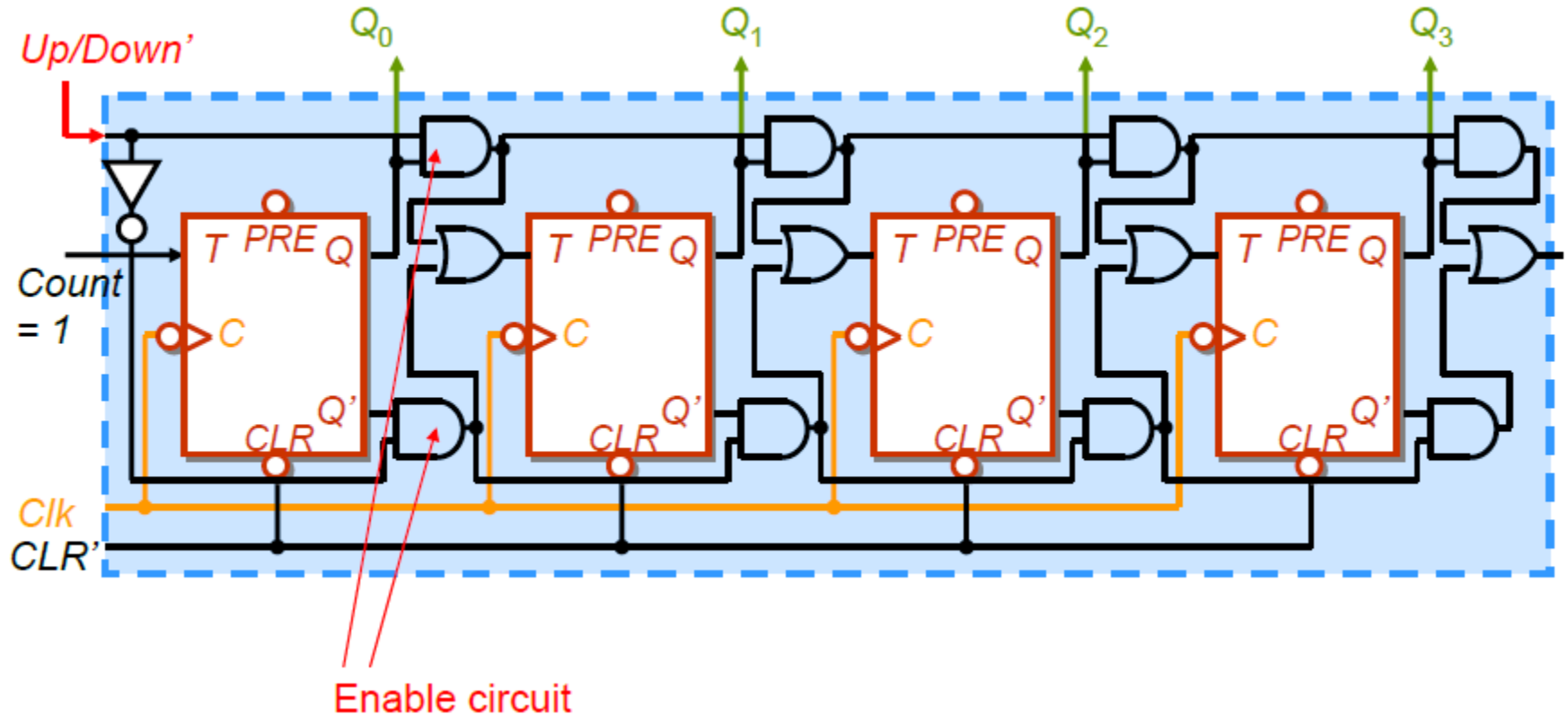
- In synchronous binary counter, all FFs are driven by a common clock.
- The change of FFs' states occurs at the same time (if their propagation times are same).
- It can also be used to provide the frequency division function.

Synchronous Counter



The delays of all FFs are the propagation delays of their own.

Sync. Up-Down Counter



Up/Down' used to select feeding either Q or Q' to next T

If $Up/Down' = 1$, upward counting (the same as previous counter). If

Up/Down' = 0, downward counting (feeding Q' to T)

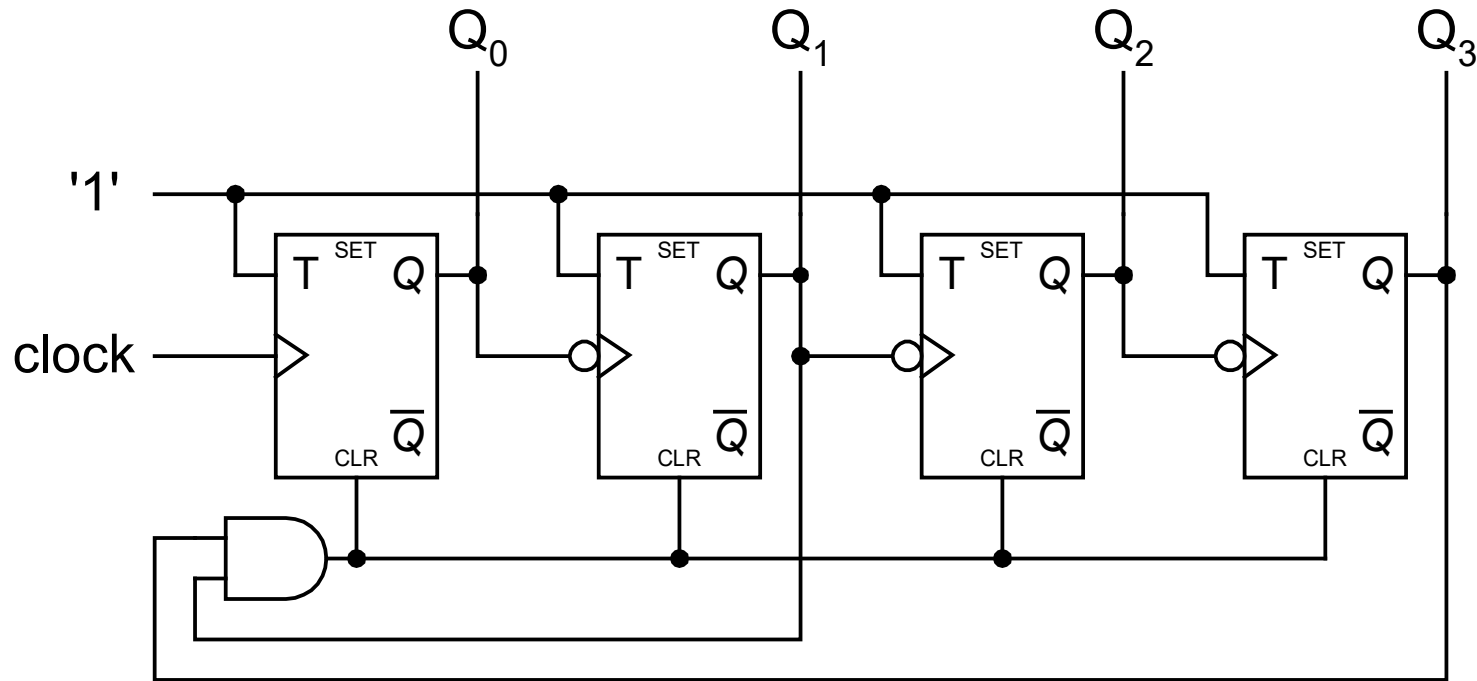
Sync. Up-Down Counter

Upward Counting Sequence			
Q_3	Q_2	Q_1	Q_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
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

Downward Counting Sequence			
Q_3	Q_2	Q_1	Q_0
1	1	1	1
1	1	1	0
1	1	0	1
1	1	0	0
1	0	1	1
1	0	1	0
1	0	0	1
1	0	0	0
0	1	1	1
0	1	1	0
0	1	0	1
0	1	0	0
0	0	1	1
0	0	1	0
0	0	0	1
0	0	0	0

BCD ripple counter

The BCD ripple counter shown below counts from 0 to 9 and then returns back to 0 and starts the counting all over again.



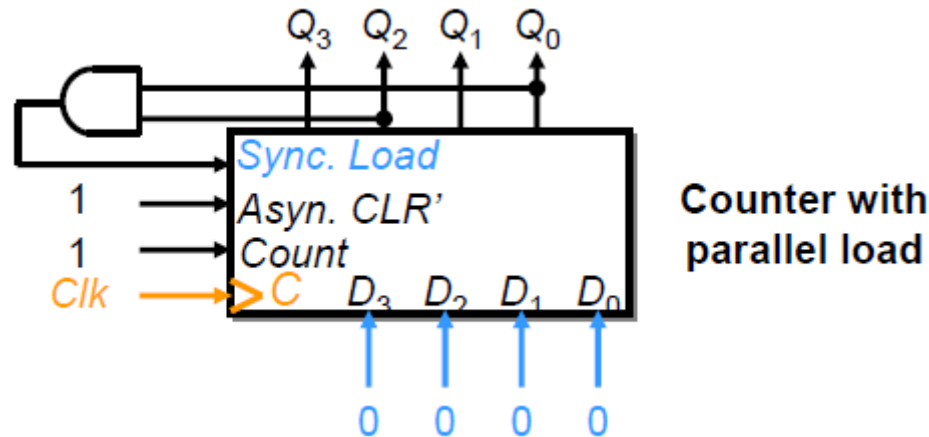
Note: CLR is an async. Clear control input

Modulo counter

- A modulo- N (abbreviated mod- N) counter is a counter that goes through a repeated sequence of N counts (from 0 to $N-1$).
- When choosing counter device, observe
 - **Syn** or **asyn** operation of the **LOAD** or **CLR** signals
 - This affects the **load/clear** circuit design

Modulo Counter Example

Mod-6 counter : 0, 1, 2, 3, 4, 5 and repeats

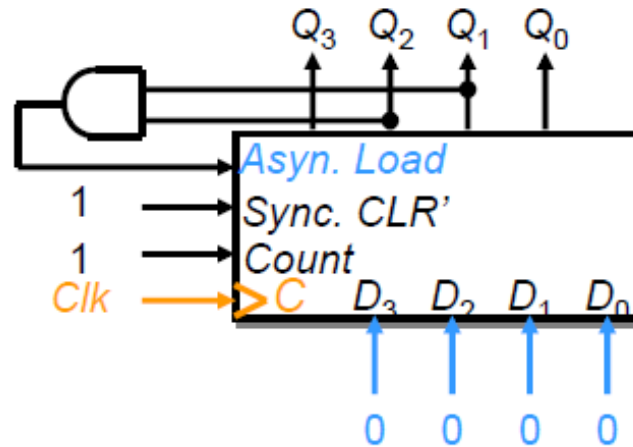


With **Sync Load** control, its counts

0000 $\rightarrow$ 0001 $\rightarrow$ 0010 $\rightarrow$ 0011 $\rightarrow$ 0100 $\rightarrow$ 0101

When next clock edge appears after 0101, Sync Load will load $D_3D_2D_1D_0$ to $Q_3Q_2Q_1Q_0$. The counter output becomes 0000 and the counter repeat its normal operation.

Modulo Counter Example

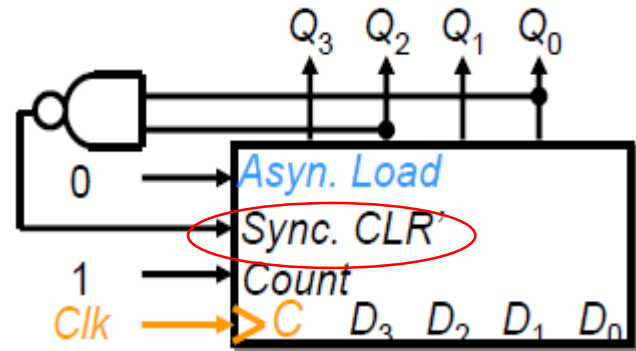
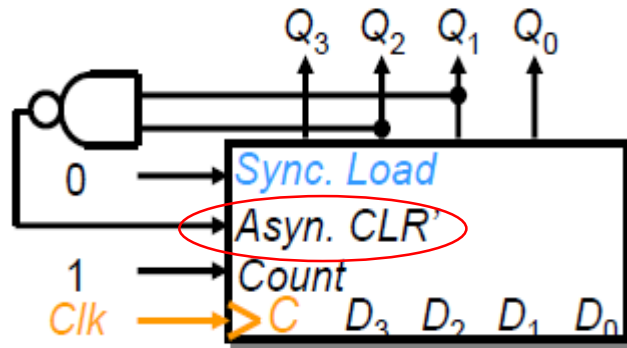


With *Async Load* control, its counts

0000 $\rightarrow$ 0001 $\rightarrow$ 0010 $\rightarrow$ 0011 $\rightarrow$ 0100 $\rightarrow$ 0101 $\rightarrow$ (0110/0000)

When 0110 appears, $D_3D_2D_1D_0$ (0000) will immediately be loaded to $Q_3Q_2Q_1Q_0$. **0110 will only appear for a negligible period.** The counter output becomes 0000 and the counter repeat its normal operation.

Modulo Counter Example

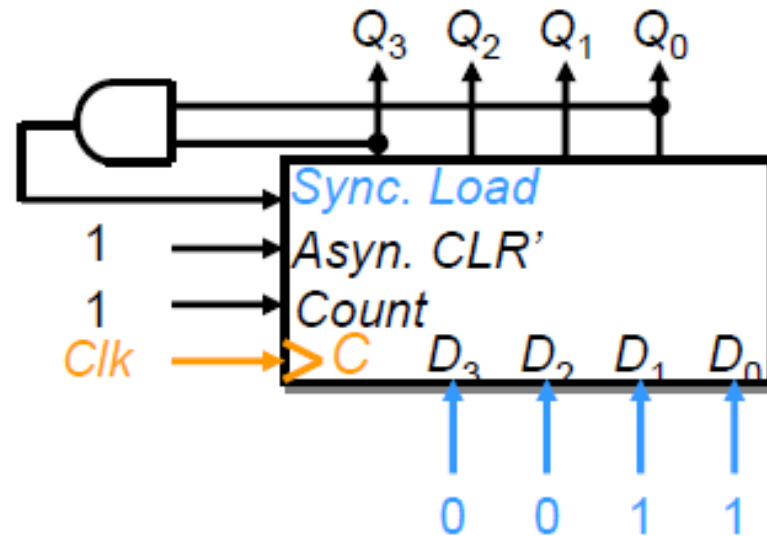


With ***Sync & Sync CLR*** control, the clear conditions are different.

This design is simpler since the **connections to inputs are not required.**

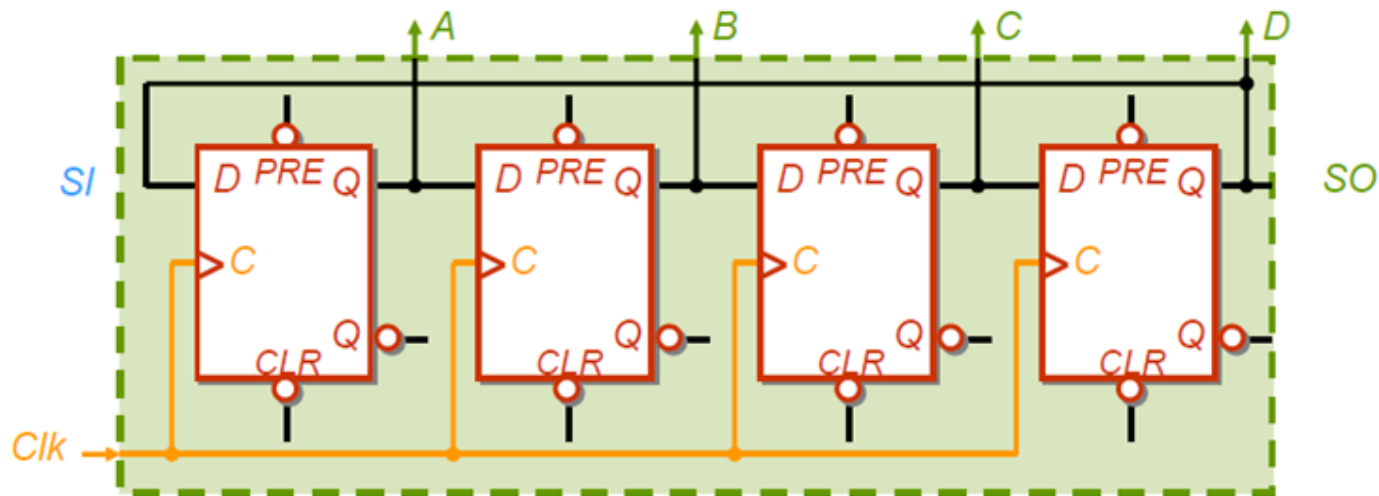
Modulo Counter Example

- Can perform specific sequence generator
- e.g. counting from 3, 4, 5, 6, 7, 8, 9 and repeats



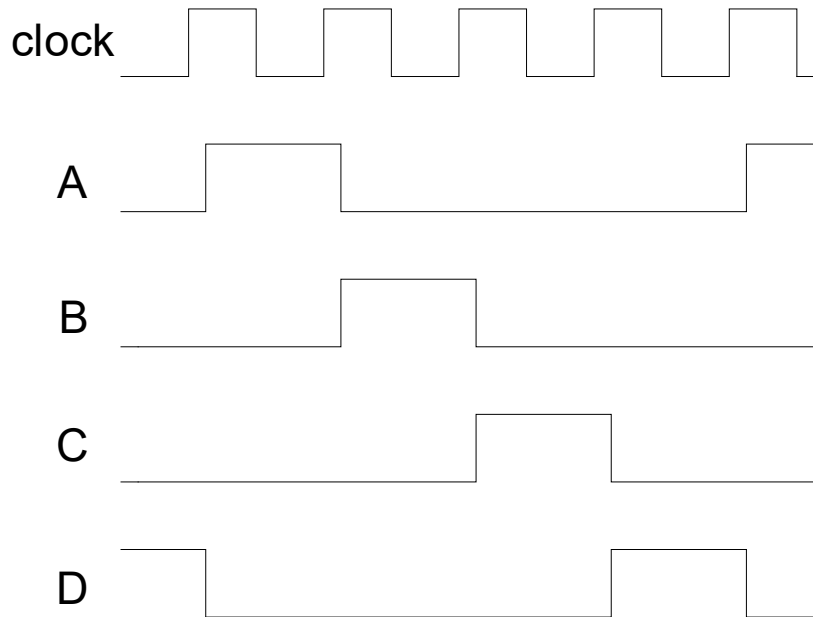
Ring Counters

- Special class of synchronous counters
- Basic Ring Counters
 - Circular shift register with only 1 flip-flops being set (i.e. = 1) at all time



It requires to initialize the circuit with only **ONE** output = 1 and all other = 0.

Basic Ring Counters

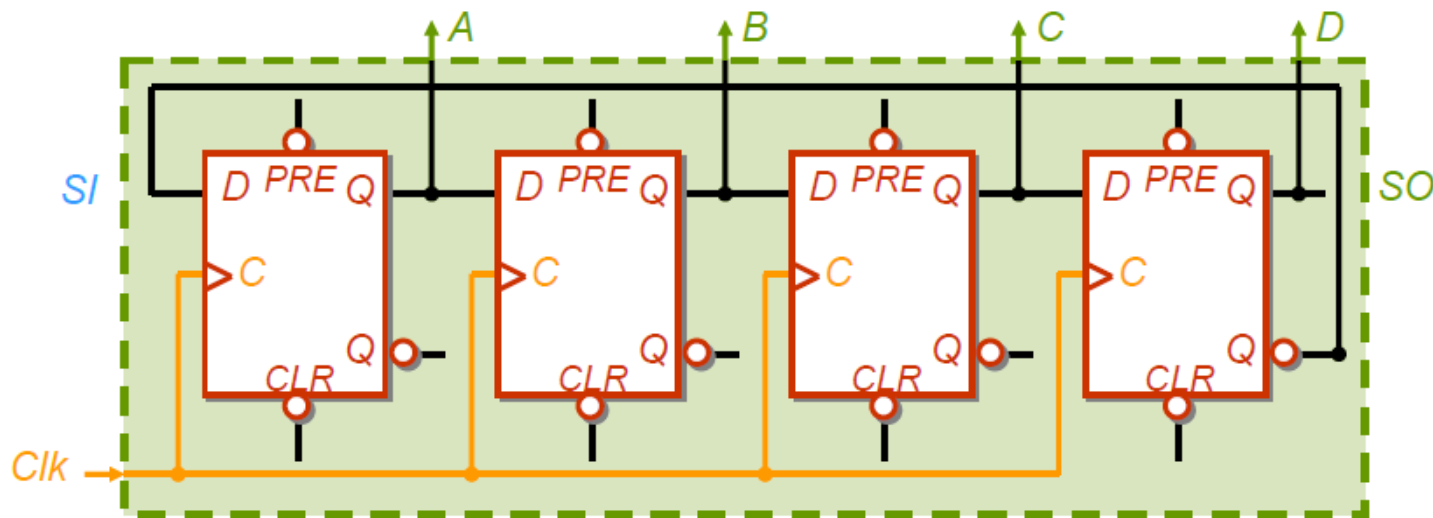


A	B	C	D	Decoded Output
1	0	0	0	A
0	1	0	0	B
0	0	1	0	C
0	0	0	1	D

- N-bit counter with N-state
- Particularly useful in time division multiplexing (TDM) applications
 - Enable only one Q_i (only one user) at a time

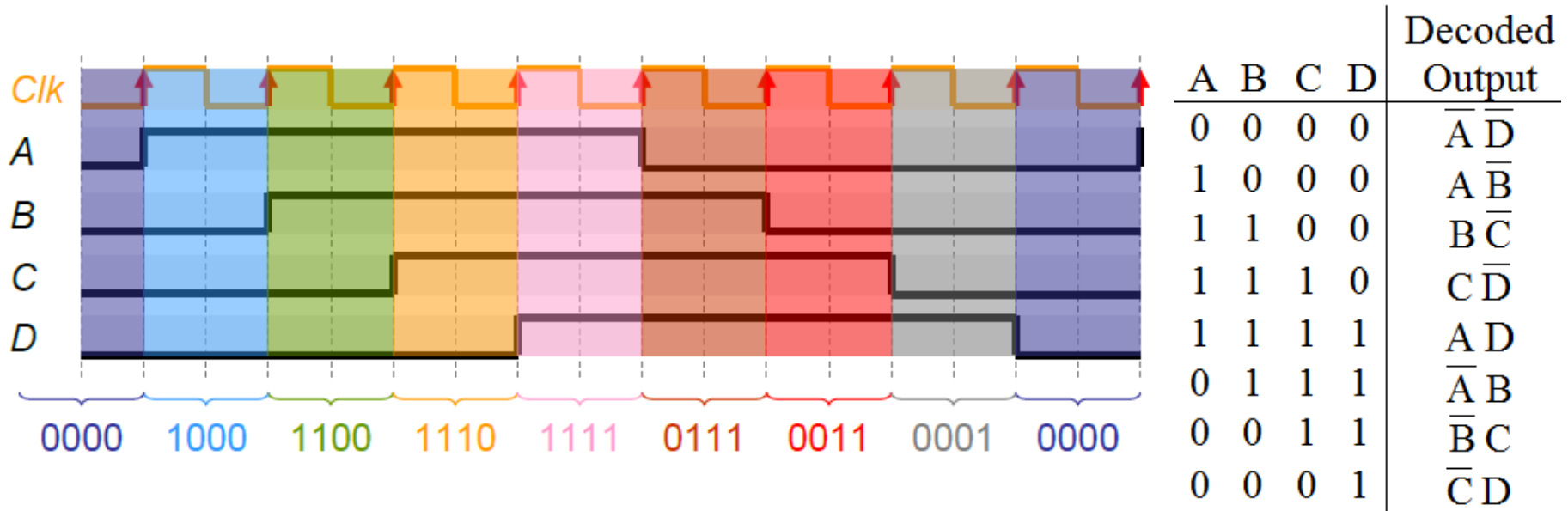
Twist Ring Counters (Johnson counter)

The twisted ring counter is a circular shift register with the complement output of the last stage connected to the input of its first stage.



Two possible initial conditions: all outputs = 0000 or 1111;
i.e., simply clear or preset all outputs.

Twist Ring Counters



- The number of states for a n -bit twisted ring counter is $2n$.
- Only one output bit changes between consecutive count states.
- The count sequences can be decoded with just 2 bits from each count state, no matter how many stages are in the counter.

Summary of 10.2

- Synchronous Counters
 - Outputs (Q_i) appear at the same time
- Asynchronous Counters
 - Outputs (Q_i) do not appear at the same time
 - The final flip-flop has the largest propagation delay
- Counters, modulo counters, ring counters, twisted ring counters
 - Simple design to construct frequency divisor, irregular frequency divisor, time division multiplexer, period shifter respectively