CS221: Digital Design

Flip Flops and Register

A. Sahu

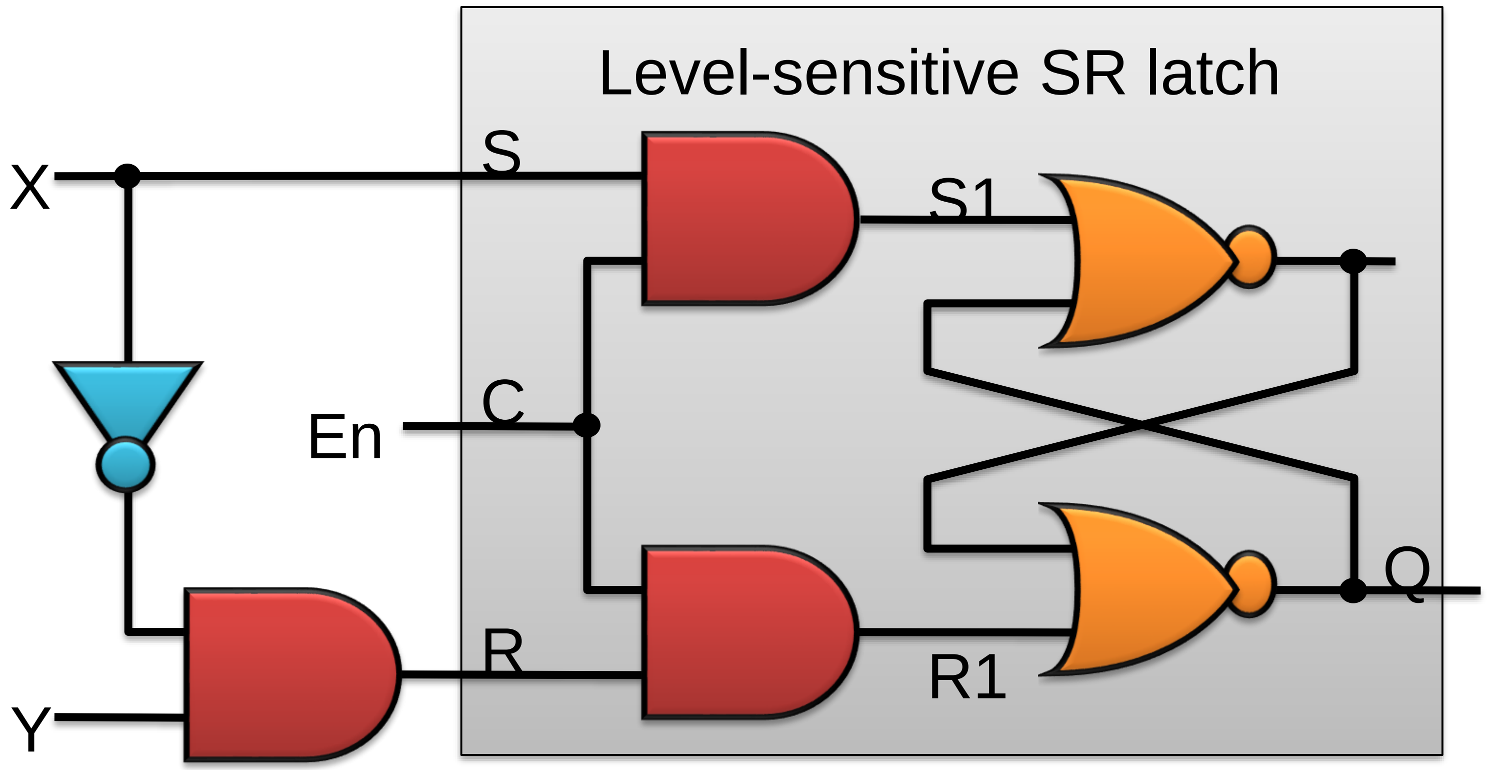
Dept of Comp. Sc. & Engg.

Indian Institute of Technology Guwahati

Outline

- ✓ Latch Stabilizing RS latch : Level Sensitive
- ✓ Clocked Latch : Flip Flop- Edge Sensitive
  - Master Slave Latches
- RS, D, JK, T flip flops
- Characterization Table and Equation
  - RS, D, JK and T Flip flop
- Excitation Table and Equation
  - RS, D, JK and T Flip flop
- Registers, MF Register and Memory

Solution: Ensure, Stabilize, Store



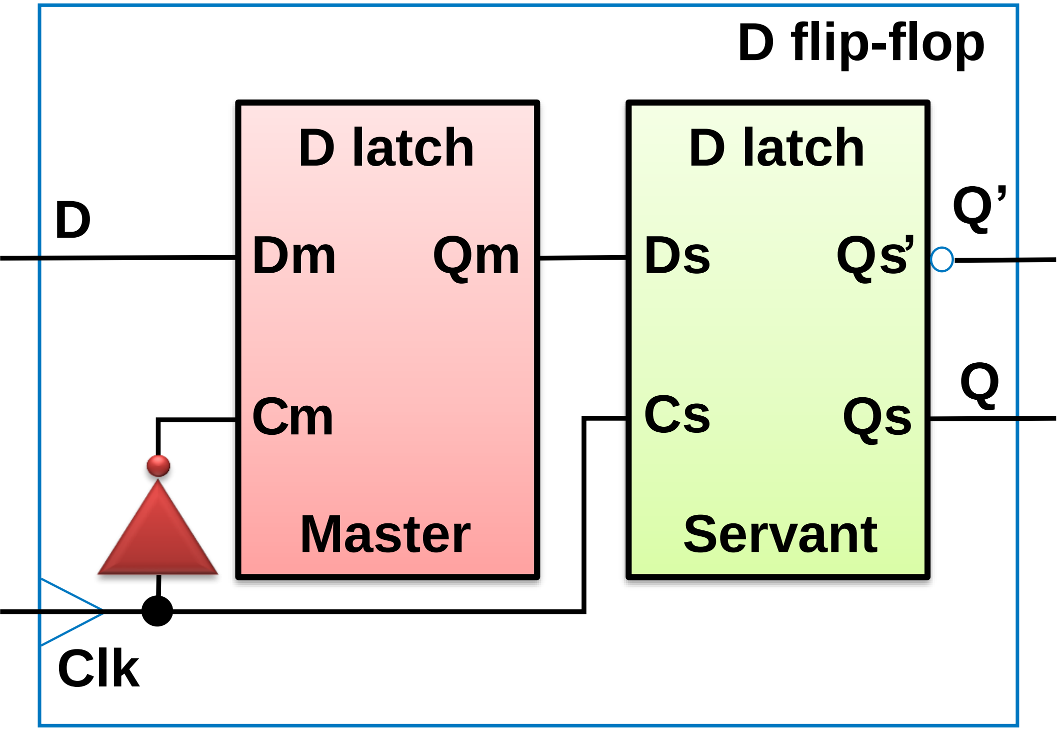
Ensure
Stage
Never Happens
SR=11

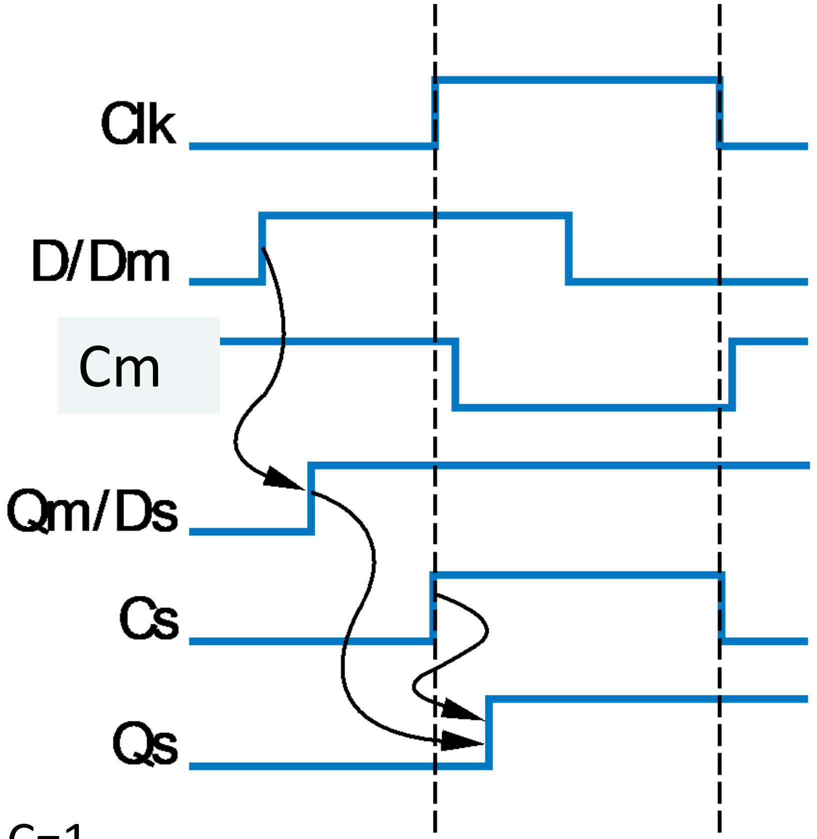
Stabilize
Stage
When C=0
Stabilize SR and
Use when C=1

Store
Stage
Store bit

Master -Slave D Flip-Flop

- ***Flip-flop***: stores on clock edge, not level

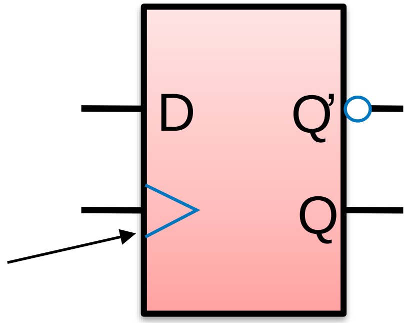




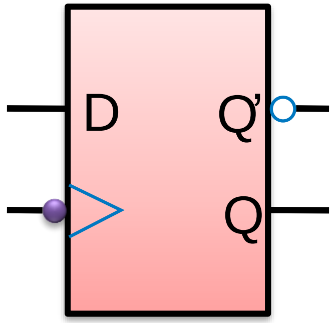
Master loaded when C=0, then servant when C=1
When C changes from 0 to 1, master disabled, servant loaded with value that was at D just before C changed -- i.e., value at D during rising edge of C

D Flip-Flop (Rising & Falling Edges)

The triangle means clock input, edge triggered



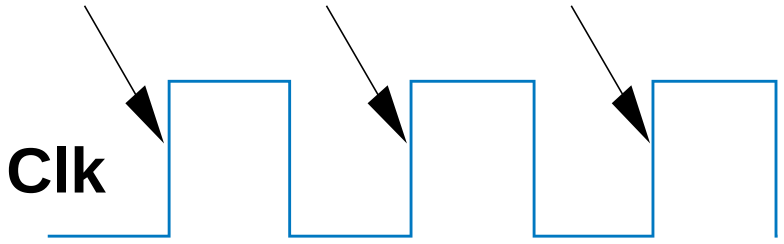
Symbol for rising-edge triggered D flip-flop



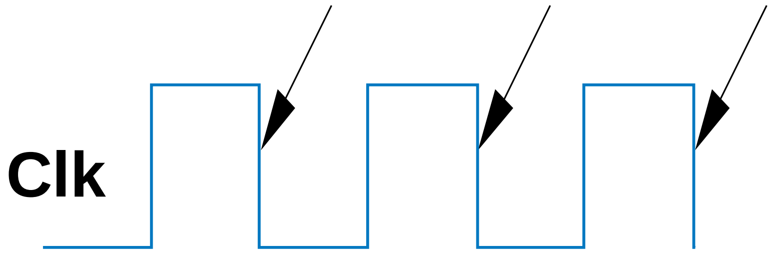
Symbol for falling-edge triggered D flip-flop

Internal design: Just invert servant clock rather than master

Rising edges

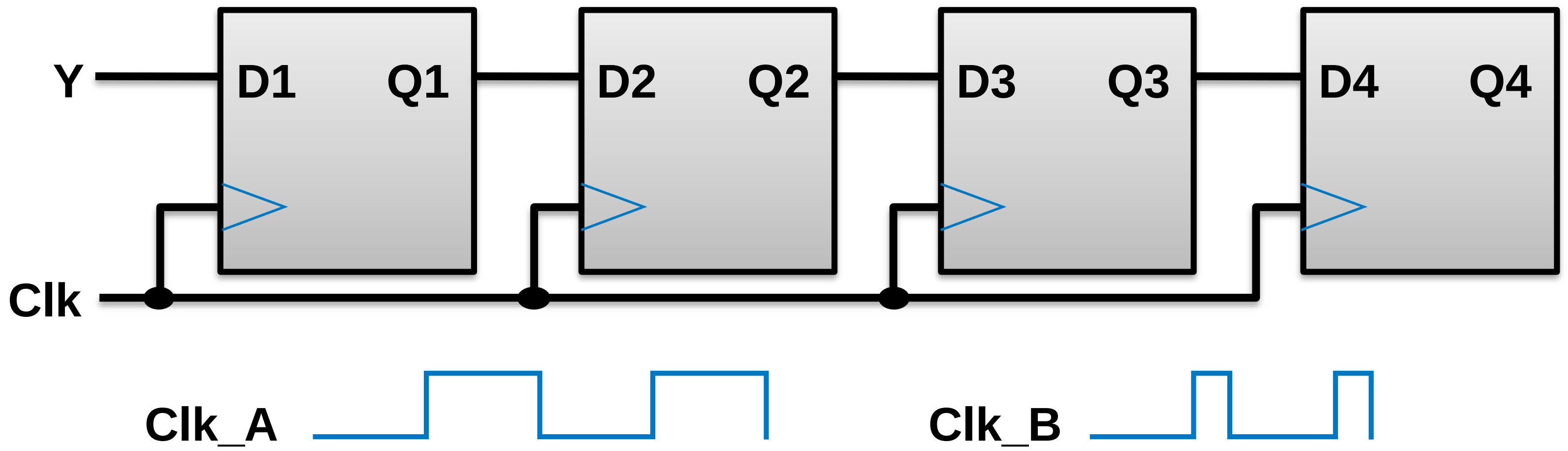


Falling edges



D Flip-Flops

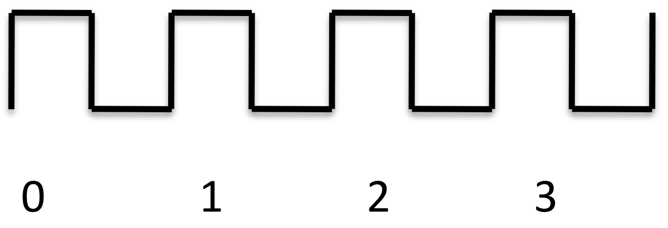
- Rising edge of Clk
  - All four flip-flops are loaded simultaneously -- then all four no longer pay attention to their input, until the next rising edge.
  - Doesn't matter how long Clk is 1.



Two latches inside each flip-flop

Conventions

- The circuit is *set* means output = 1
- The circuit is *reset* means output = 0
- Flip-flops have two output Q and Q’
- Due to time related characteristic of the flip-flop:
  - Q_t or Q: present state
  - Q_{t+1} or Q^+: next state

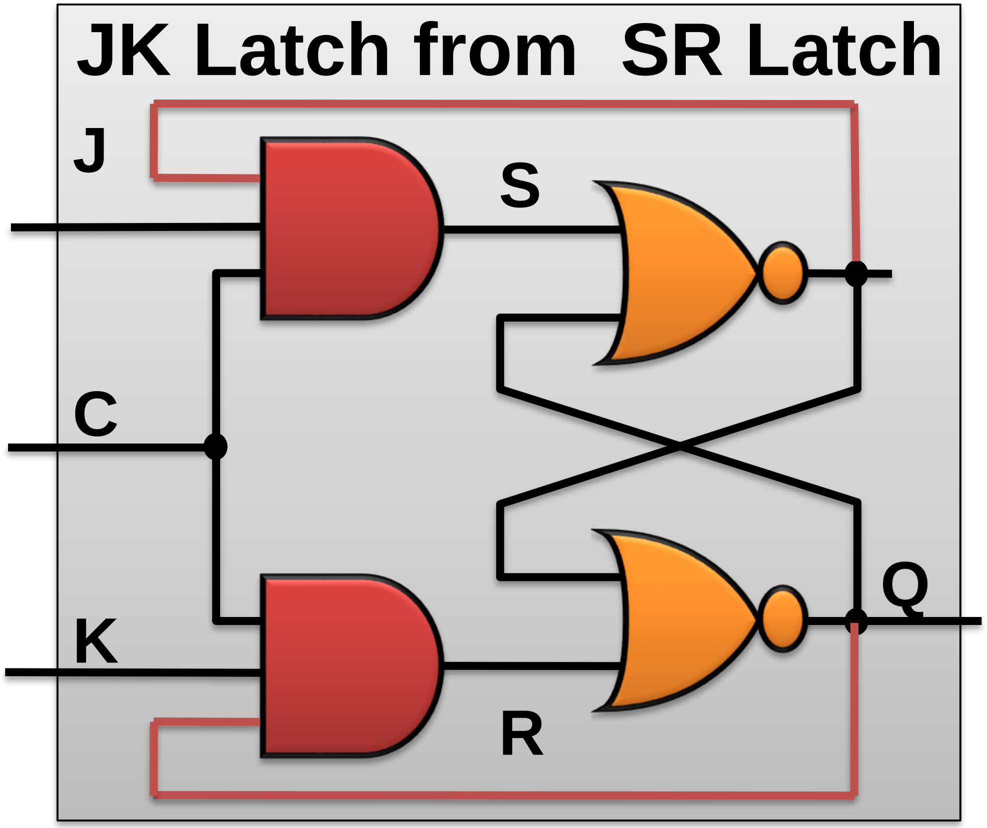


4 Type of Flip Flop

- **SR Flip Flop** : Set/Reset Flip Flop
- **D Flip Flop** : Data Flip Flop to store Bit
- **J-K Flip Flop**: Unavoidable SR=11 state to Toggle (All input values are useful)
  - The JK Flip Flop was named to honuor "**Jack Kilby**" of Texas Instrument engineer who invented the concept of IC.
- **T Flip Flop**: Toggle Flip Flop

J-K Latches

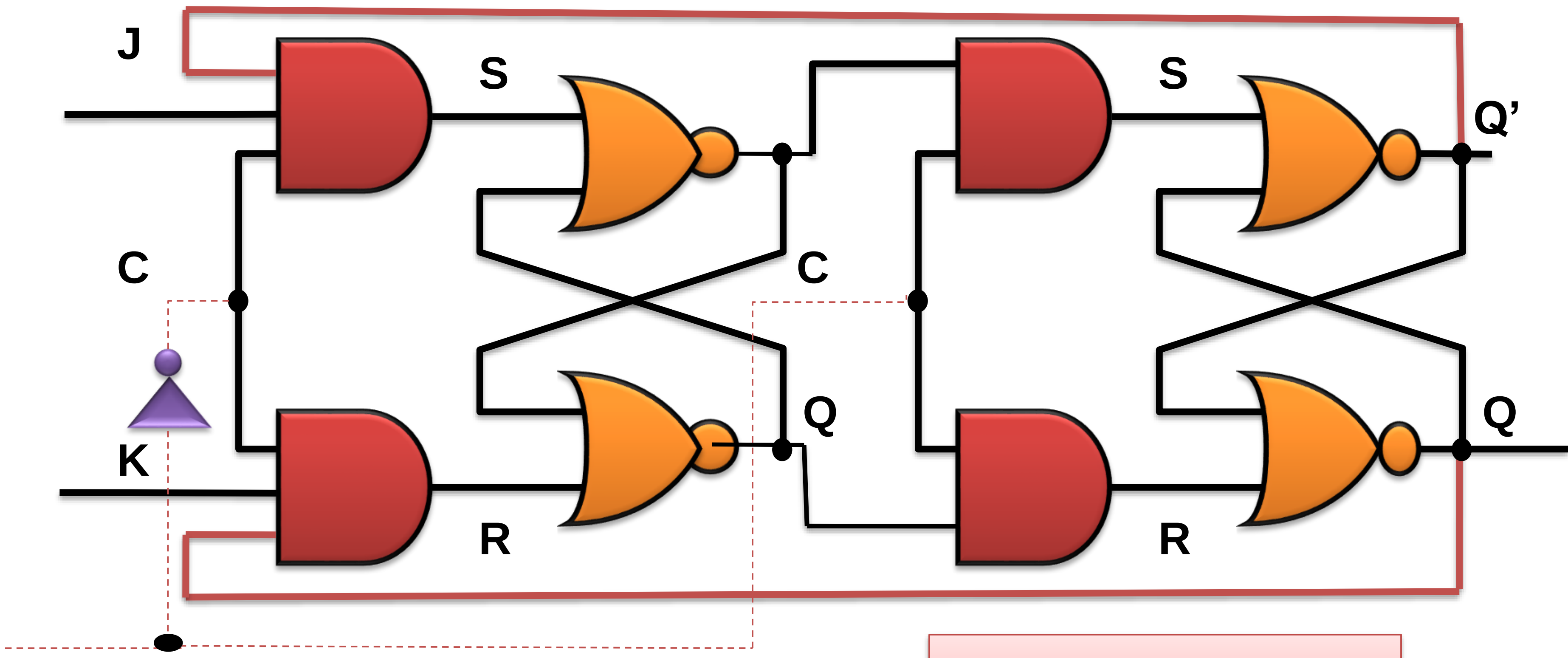
- The JK flip-flop augments the behavior of the SR flip-flop (J=Set, K=Reset) by interpreting the S = R = 1 condition as a "flip" or toggle command.



J	K	Q+
0	**0**	**Qt**
0	1	0
1	0	1
1	**1**	**Qt’**

$$Q^{+} = K'Q + JQ'$$

Master Slave J-K Flip Flop



$Q^{+} = K'Q + JQ'$

J-K Flipflop

J	K	Q+
0	**0**	**Qt**
0	1	0
1	0	1
1	**1**	**Qt’**

$$
\begin{aligned}
Q+ &= J'K'Q + JKQ' + \mathbf{JK'} \\
&= J'K'Q+JKQ'+\mathbf{JK'Q'}+\mathbf{JK'Q} \\
&= J'K'Q+\mathbf{JK'Q} + JKQ'+\mathbf{JK'Q'} \\
&= (J+J')K'Q+J(K+K')Q' \\
&= (1)K'Q+J(1)Q'
\end{aligned}
$$

$$\boxed{Q^+ = K'Q + JQ'}$$

J-K Flip Flop

- To synthesize a D flip-flop, simply set K equal to the complement of J.
- **The JK flip-flop is a universal flip-flop**
  - Because it can be configured to work as any FF
  - T flip-flop or D flip-flop or SR flip-flop.

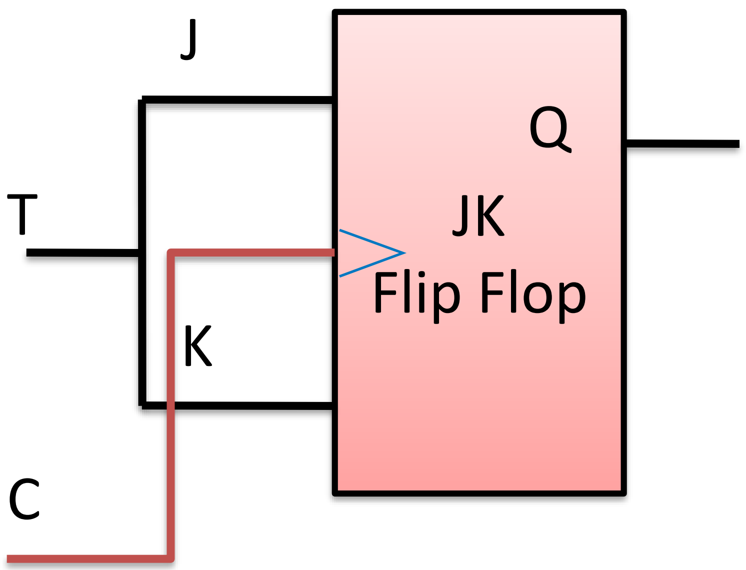
J=T	K=T	Q+
0	**0**	**Qt**
0	1	0
1	0	1
1	**1**	**Qt’**

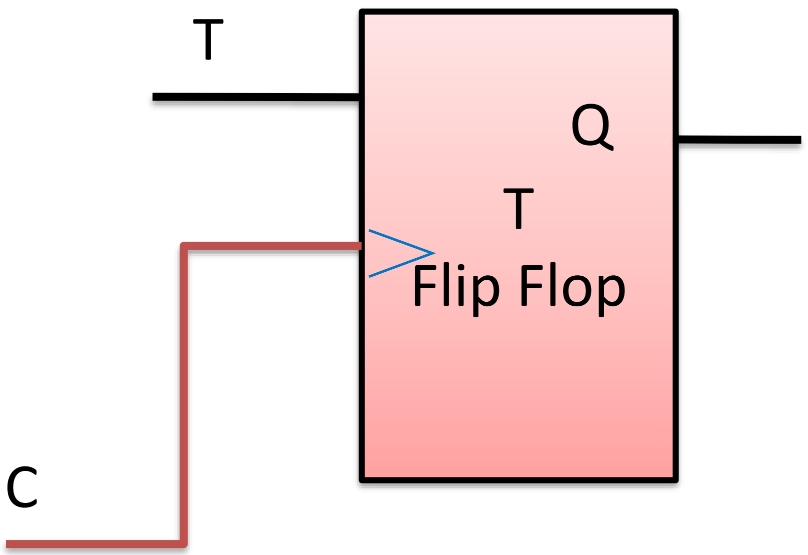
J=D	K=D’	Q+
0	0	Qt
0	**1**	**0**
1	**0**	**1**
1	1	Qt’

J=S	K=R	Q+
0	**0**	**Qt**
0	**1**	**0**
1	**0**	**1**
1	1	Qt’

Toggle Flip-Flop: T FF

- J=K=1, Q^+= Q'





4 Types of Flip-Flops

S	R	Q+
0	0	Qt
0	**1**	**0**
1	**0**	**1**
1	**1**	**U**

D	Q+
0	0
1	1

J	K	Q+
0	**0**	**Qt**
0	1	0
1	0	1
1	**1**	**Qt’**

T	Q+
0	Qt
1	Qt’

Characteristic Equations

- A descriptions of the next-state table of a flip-flop
- Constructing from the Karnaugh map for Q_{t+1} in terms of the present state and input

Characteristic tables

- The tables that we've made so far are called characteristic tables.
  - They show the next state Q(t+1) in terms of the current state Q(t) and the inputs.
  - For simplicity, the control input C is not usually listed.
  - Again, these tables don't indicate the positive edge-triggered behavior of the flip-flops that we'll be using.

J	K	Q+
0	**0**	**Qt**
0	1	0
1	0	1
1	**1**	**Qt'**

D	Q+
0	0
1	1

T	Q+
0	Qt
1	Qt'

JK FF: Characteristic equations

- We can also write characteristic equations, where the next state Q(t+1) is defined in terms of the current state Q(t) and inputs.

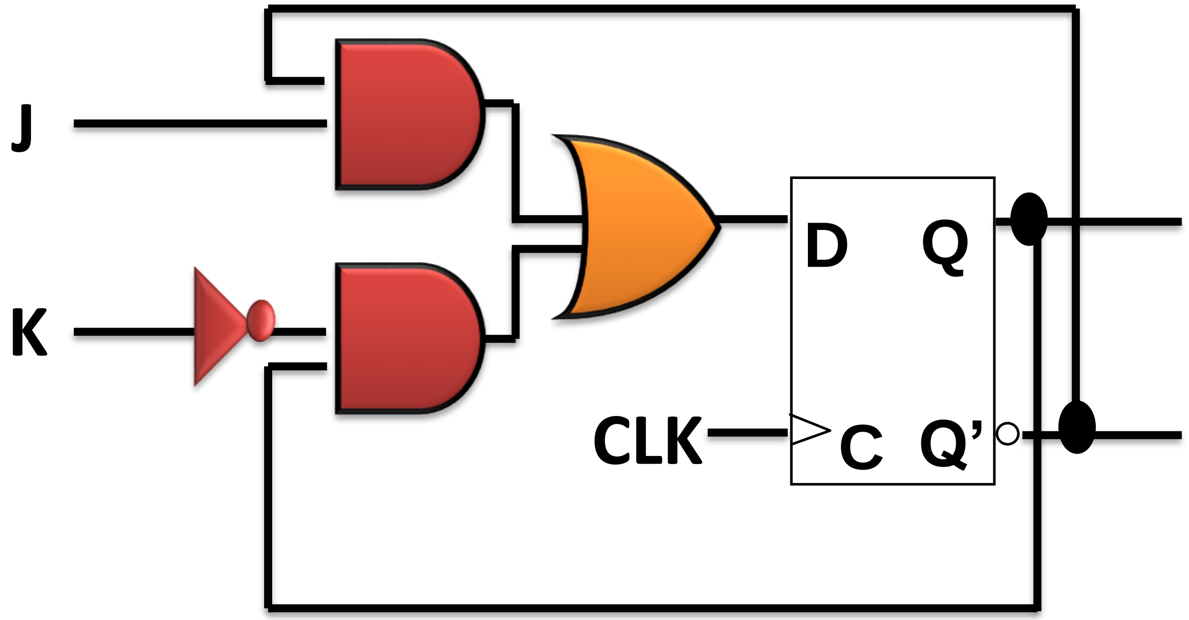
J	K	Q+
0	**0**	**Qt**
0	1	0
1	0	1
1	**1**	**Qt'**

$$
\begin{aligned}
Q+ &= J'K'Q + JKQ' + JK' \\
&= J'K'Q+JKQ'+JK'Q'+JK'Q \\
&= J'K'Q+JK'Q + JKQ'+JK'Q' \\
&= (J+J')K'Q+J(K+K')Q' \\
&= (1)K'Q+J(1)Q'
\end{aligned}
$$

$$\mathbf{Q^{+}= K'Q + JQ'}$$

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

Given a D FF: Construct JK FF



J	K	D	K'Q+JQ'
0	0	Q	1Q+0Q'
0	1	0	0Q+0Q'
1	0	1	1Q+1Q'
1	1	Q'	0Q+1Q'

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

D FF and T FF: Characteristic equations

- We can also write characteristic equations, where the next state Q(t+1) is defined in terms of the current state Q(t) and inputs.

D	Q+
0	0
1	1

$Q^{+}=D$

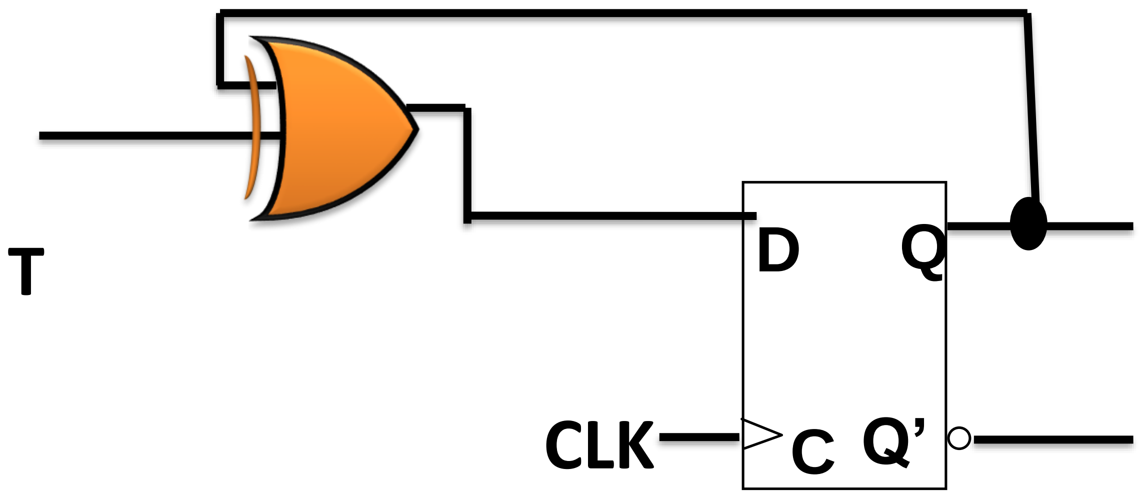
$Q(t+1) = D$

T	Q+
0	Qt
1	Qt'

$Q+= T'Q+TQ' = T \oplus Q$

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

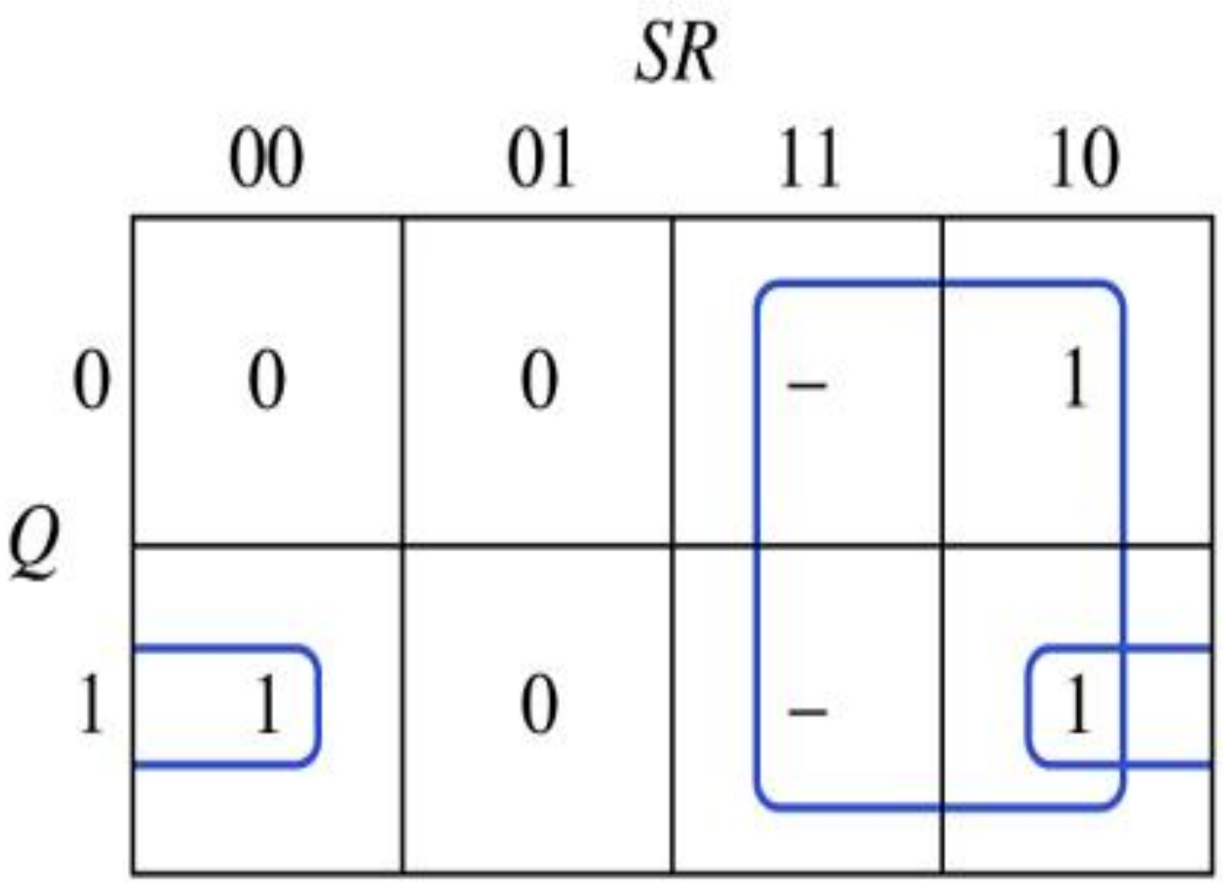
Given a D FF: Construct T FF



T	D	TQ’+T’Q
0	Q	0Q’+1Q
1	Q’	1Q’+0Q

D= TQ’ + TQ

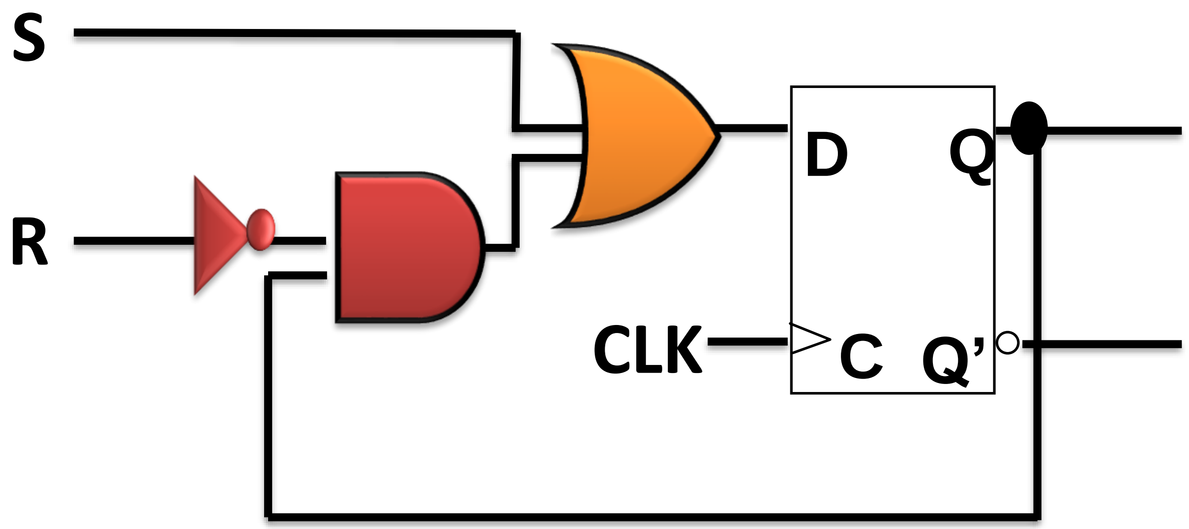
RS FF: Characteristic equations



Q \ SR	00	01	11	10
0	0	0	–	1
1	1	0	–	1

$$Q^+ = S + \bar{R}Q \quad (SR = 0)$$

Given a D FF: Construct RS FF



S	R	D	$Q^{+}= S + R'Q$ (SR=0)
0	0	Q	1Q+0Q’
0	1	0	0Q+0Q’
1	0	1	1Q+1Q’
1	1	x	x

$$D= S+R'Q$$

All FF: Characteristic equations

Flip Flop Type	Characteristic Equation
SR	$Q^{+}= S + R'Q \quad (SR=0)$
JK	$Q^{+}= JQ'+K'Q$
D	$Q^{+}=D$
T	$Q^{+}=TQ'+T'Q=T\oplus Q$

FF with Asynchronous Inputs

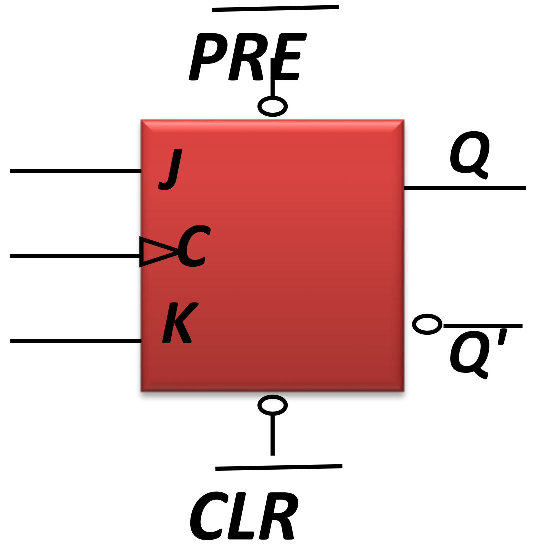
- S-R, D and J-K inputs are synchronous inputs
  - As data on these inputs are transferred to the flip-flop's output
  - Only on the triggered edge of the clock pulse.
- Asynchronous inputs affect the state of the flip-flop independent of the clock
- Example:
  - *Preset* (*PRE*) and *clear* (*CLR*)
  - or *direct set* (*SD*) and *direct reset* (*RD*)

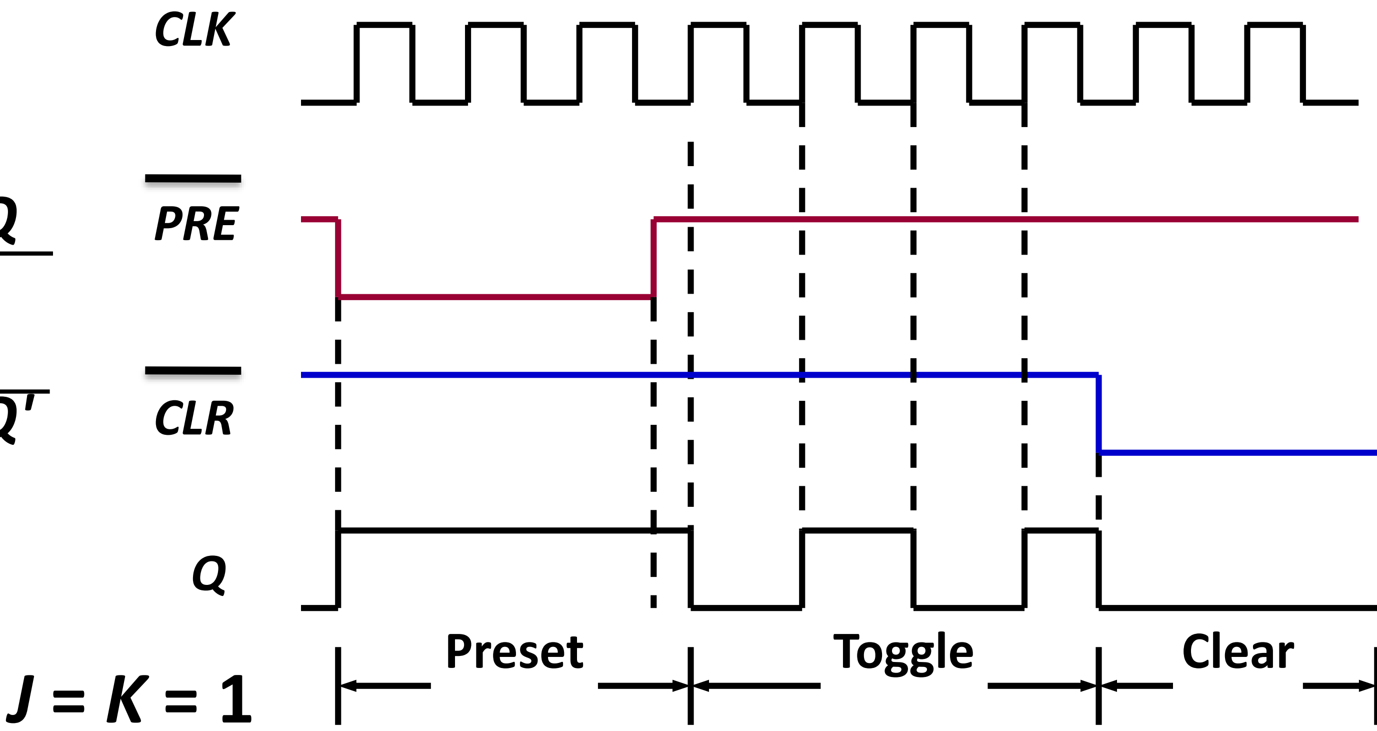
FF with Asynchronous Inputs

- When *PRE*=HIGH, *Q* is immediately set to HIGH.
- When *CLR*=HIGH, *Q* is immediately cleared to LOW.
- Flip-flop in normal operation mode when both *PRE* and *CLR* are LOW.

Asynchronous Inputs

- A J-K flip-flop with active-LOW preset and clear inputs.





Register

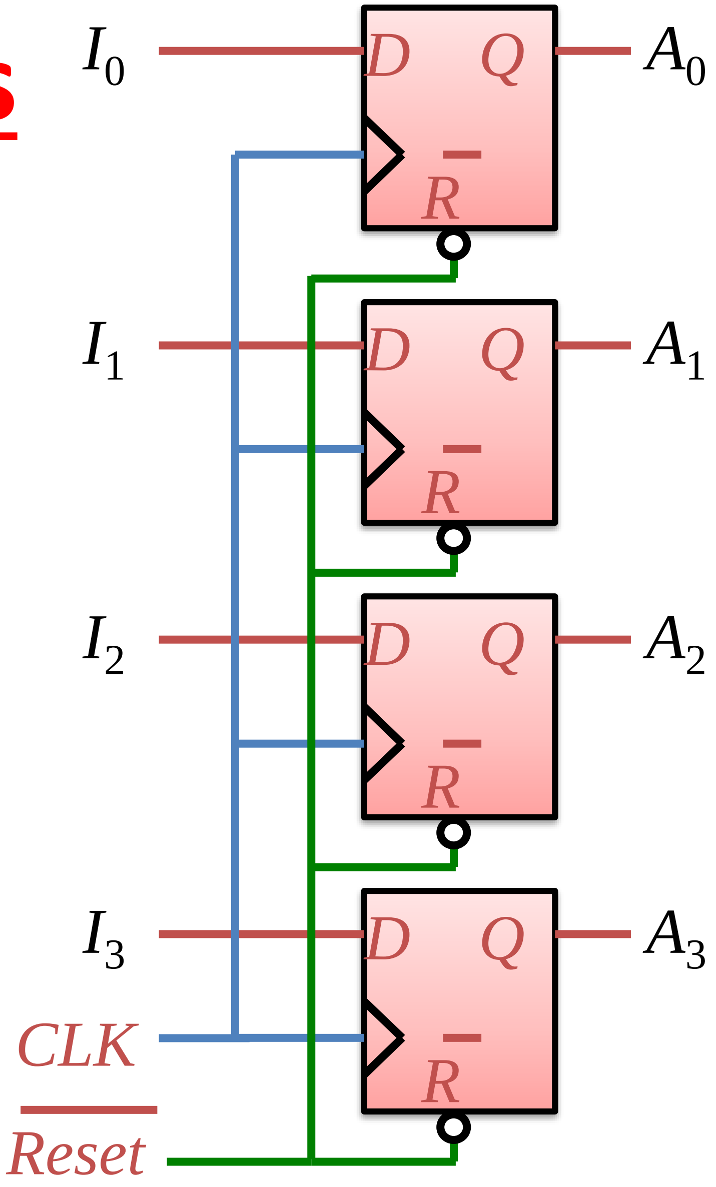
- D FF store one bit
- Register : Store multiple bit simultaneously

Register

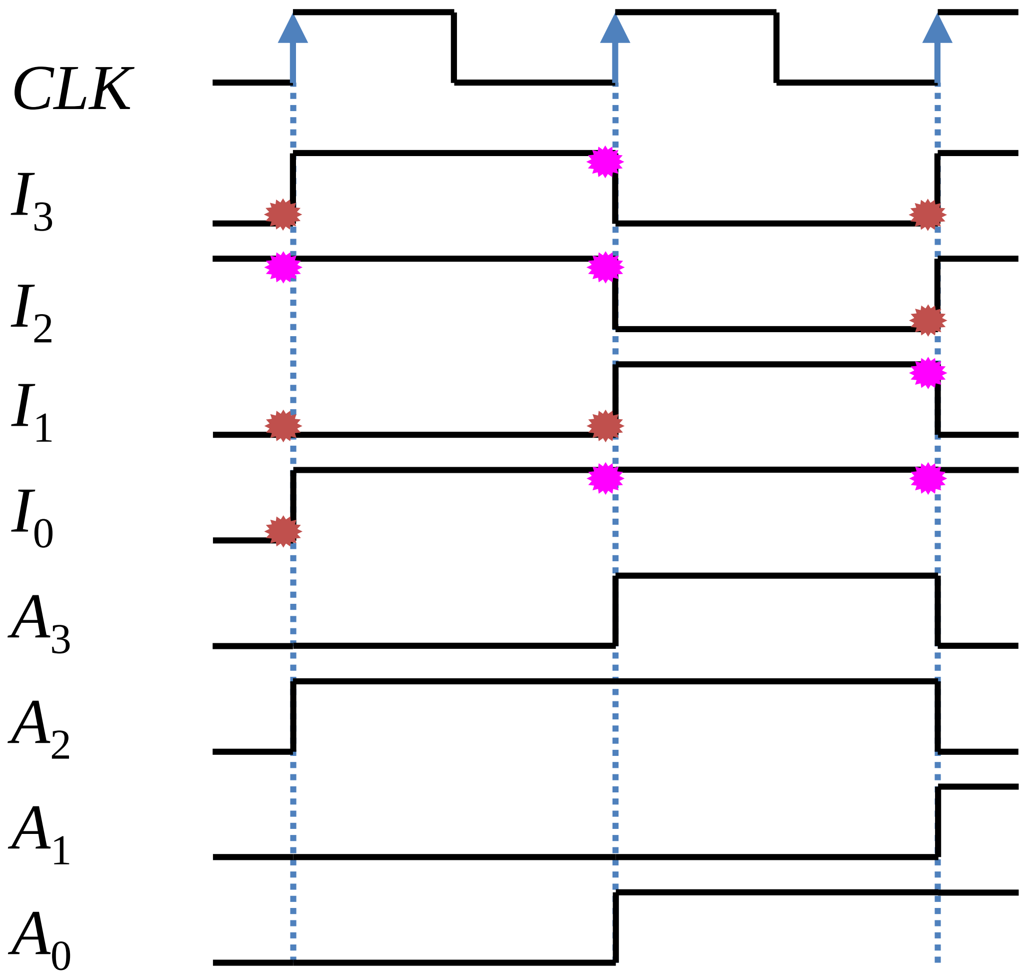
- Register
  - Parallel Load, Parallel out : (PIPO)
  - Serial Load, Wrap around load, Serial out (SISO)
  - PISO, SIPO Register
- Multifunction Register : How to design?
- Memory Design using array of PIPO registers

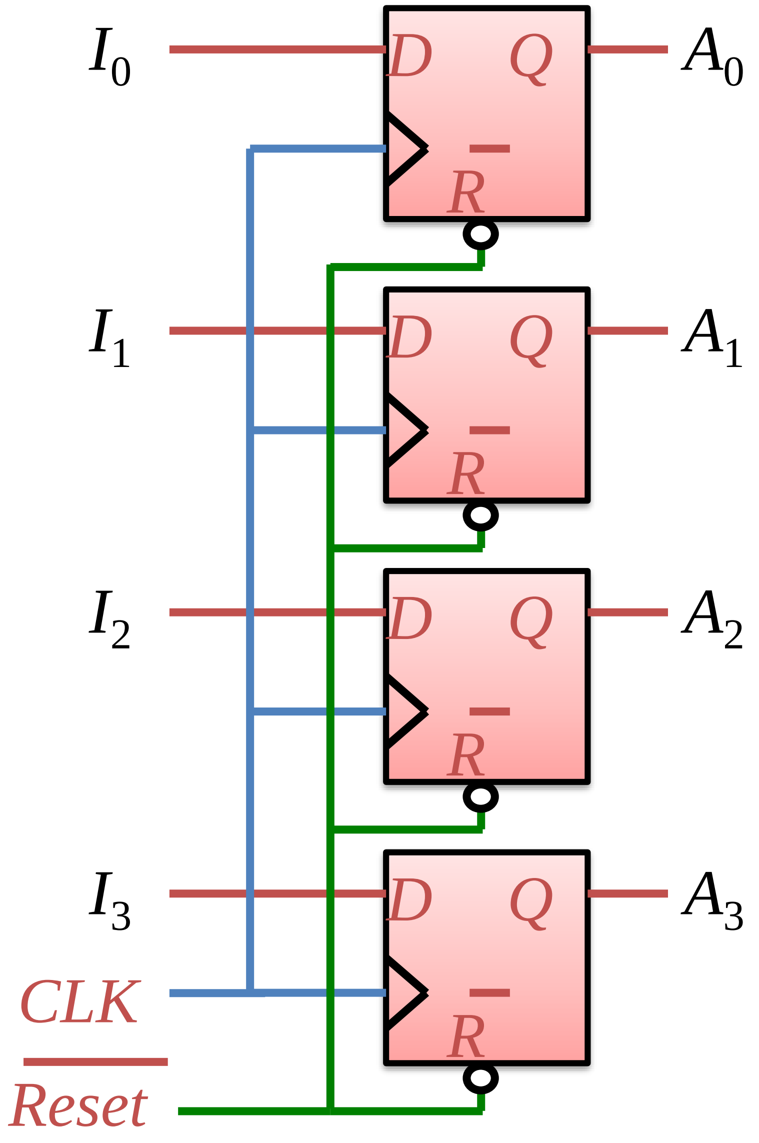
Registers

- Group of *D* Flip-Flops
- **Synchronized (Single Clock)**
- Store Data



Registers



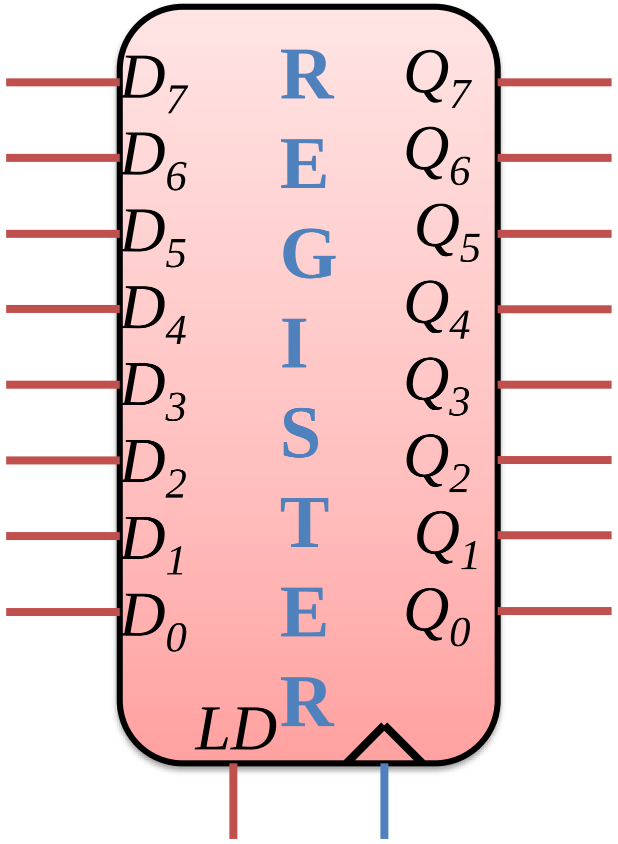


Note: New data has to go in with every clock

See carefully: Input at the dotted will be reflected to output : Just before rising edges

Registers with Parallel Load

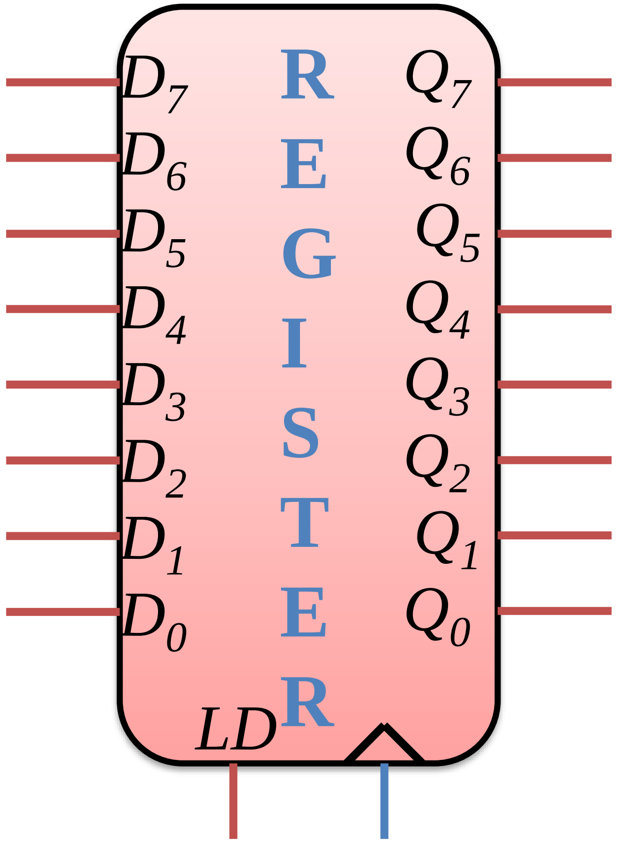
- Control ***Loading*** the Register with New Data

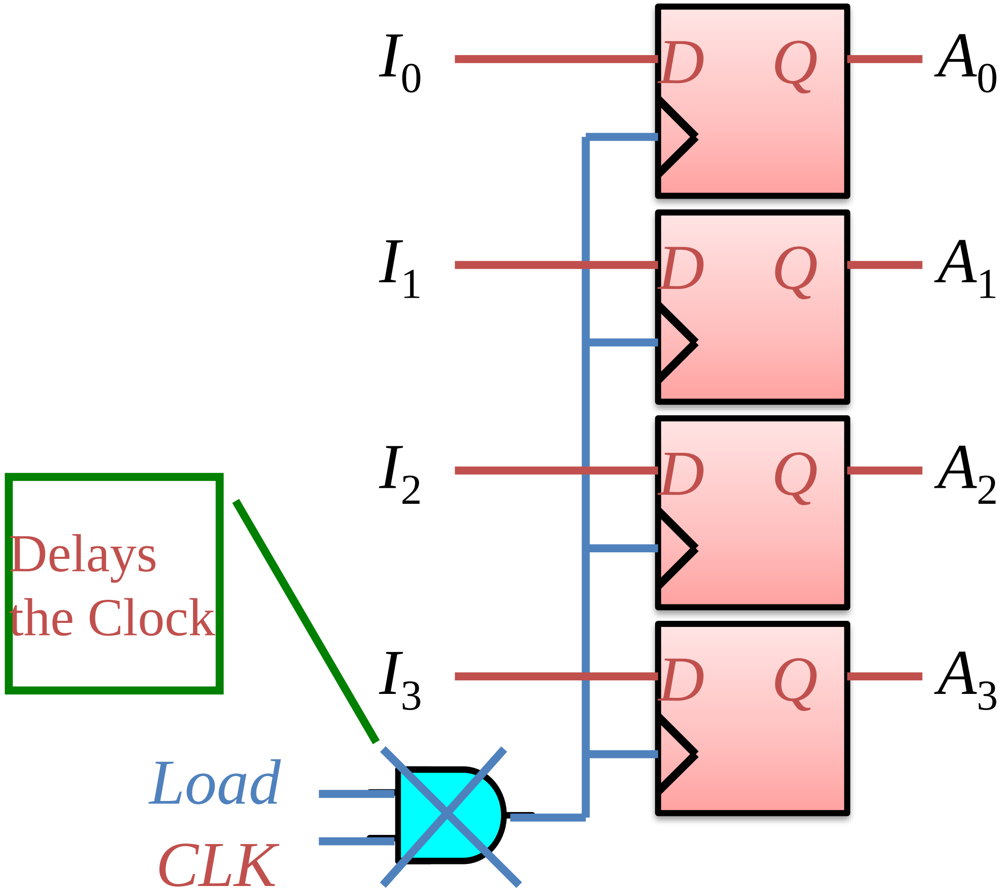


LD	Q(t+1)
0	Q(t)
1	D

Registers with Parallel Load

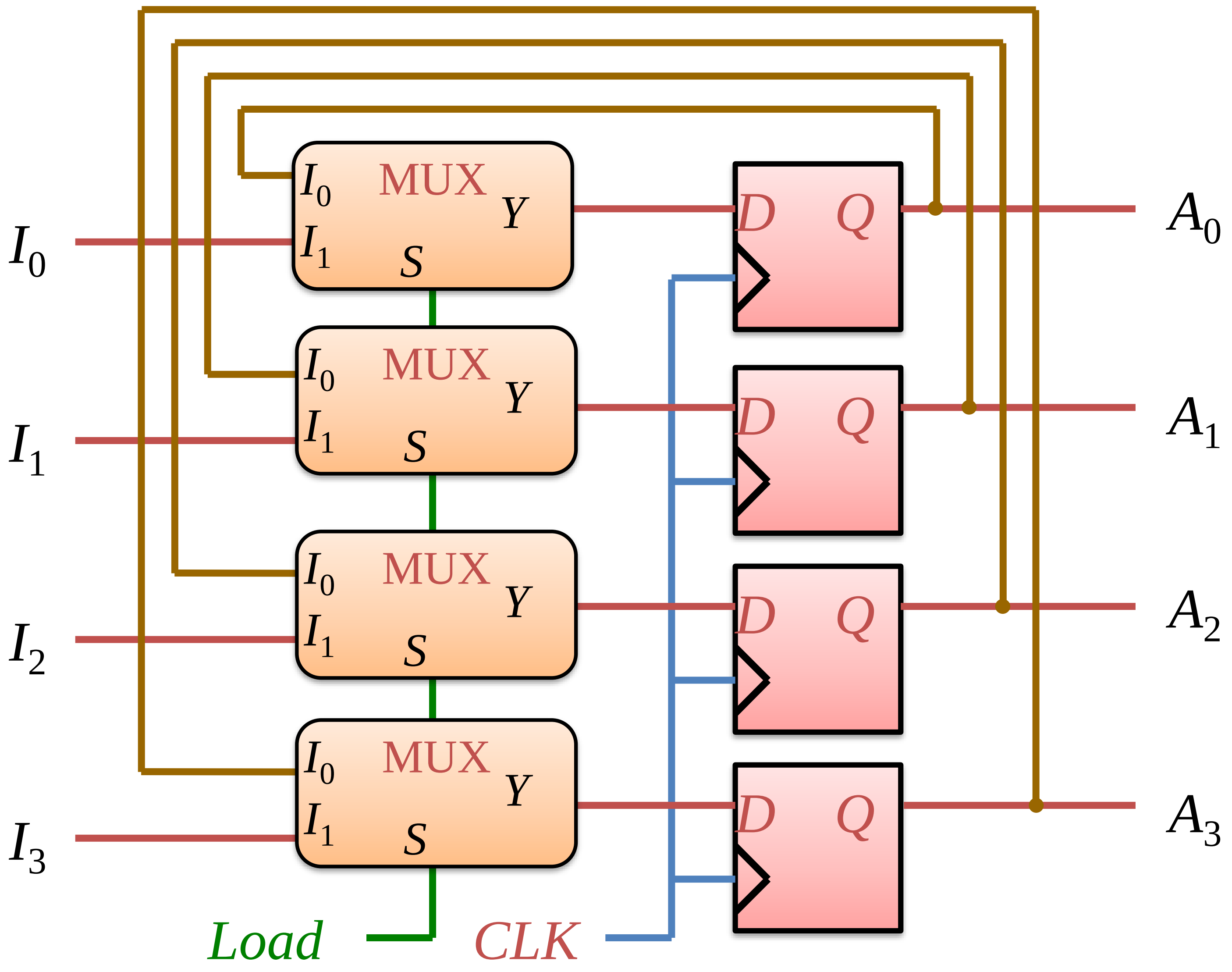
- Should we block the "Clock" to keep the "Data"?





Registers with Parallel Load

- Circulate the “old data”



Shift Registers

- Register (Set of FFs)
- 4-Bit Shift Register (Example)
- Serial in Serial Out (SISO)

Shift Register

- Right Shift Example (Left shift is similar)
  – Move each bit one position right
  – Shift in 0 to leftmost bit

Q: Do four right shifts on 1001, showing value after each shift

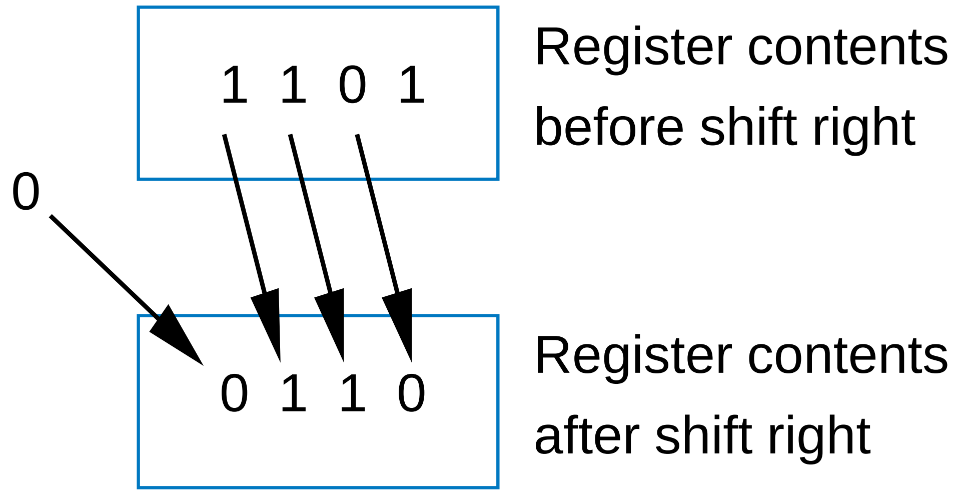
A: 1001 (original)

0100

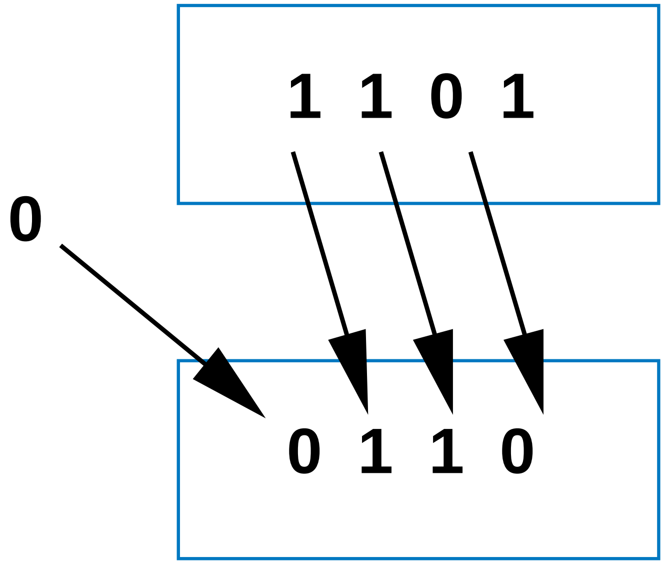
0010

0001

0000


1 1 0 1
Register contents before shift right
0
0 1 1 0
Register contents after shift right

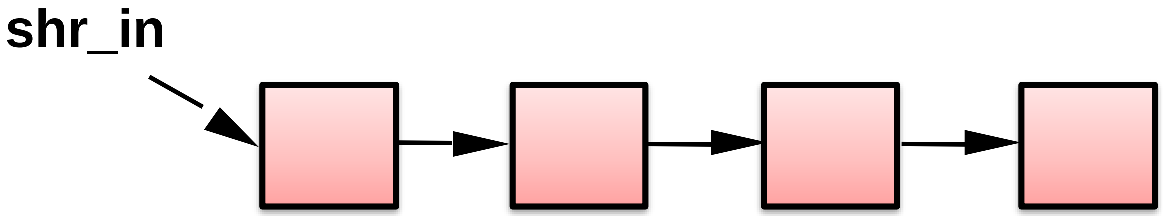
Shift Register



Register contents before shift right

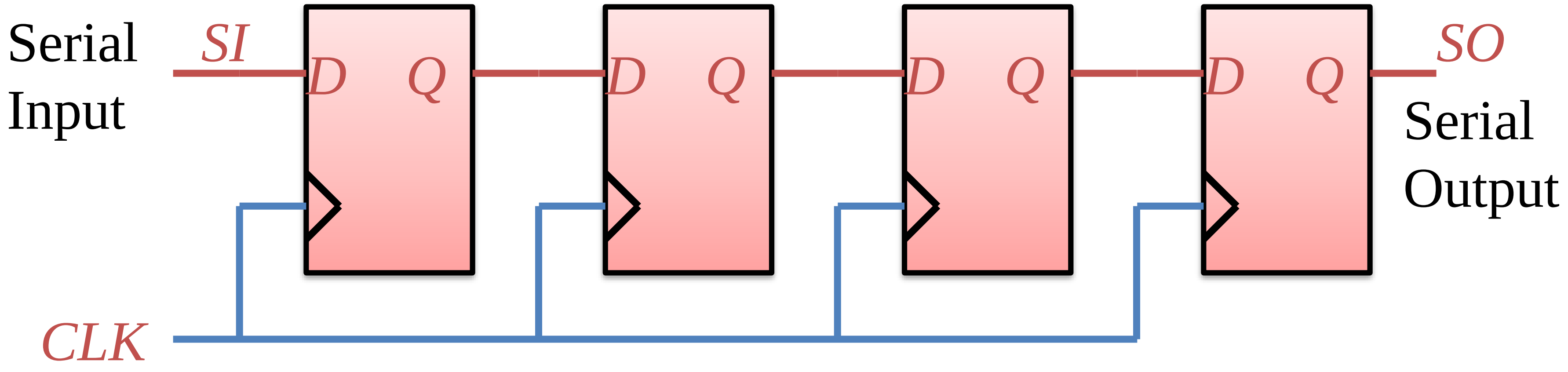
Register contents after shift right

- **Implementation:** Connect flip-flop output to next flip-flop's input

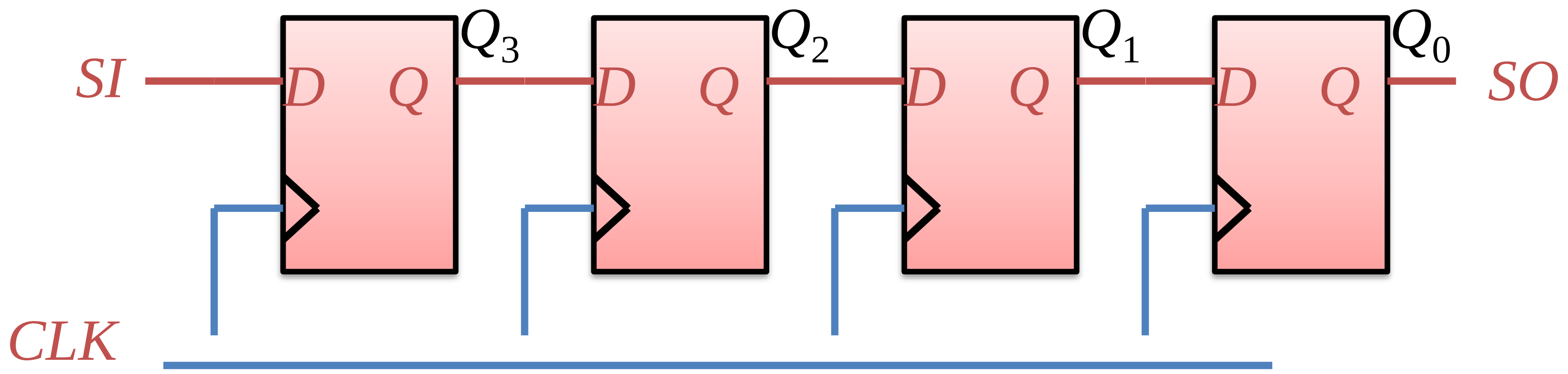


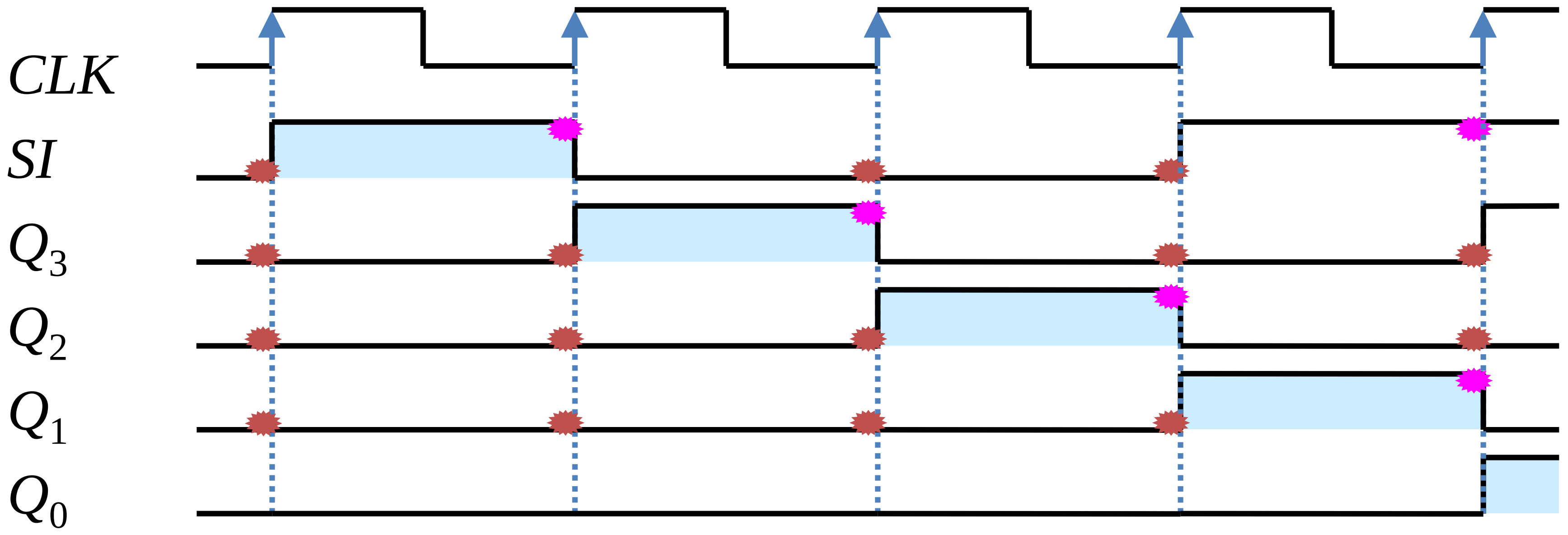
Shift Registers

- 4-Bit Shift Register
- Serial in Serial Out (SISO)



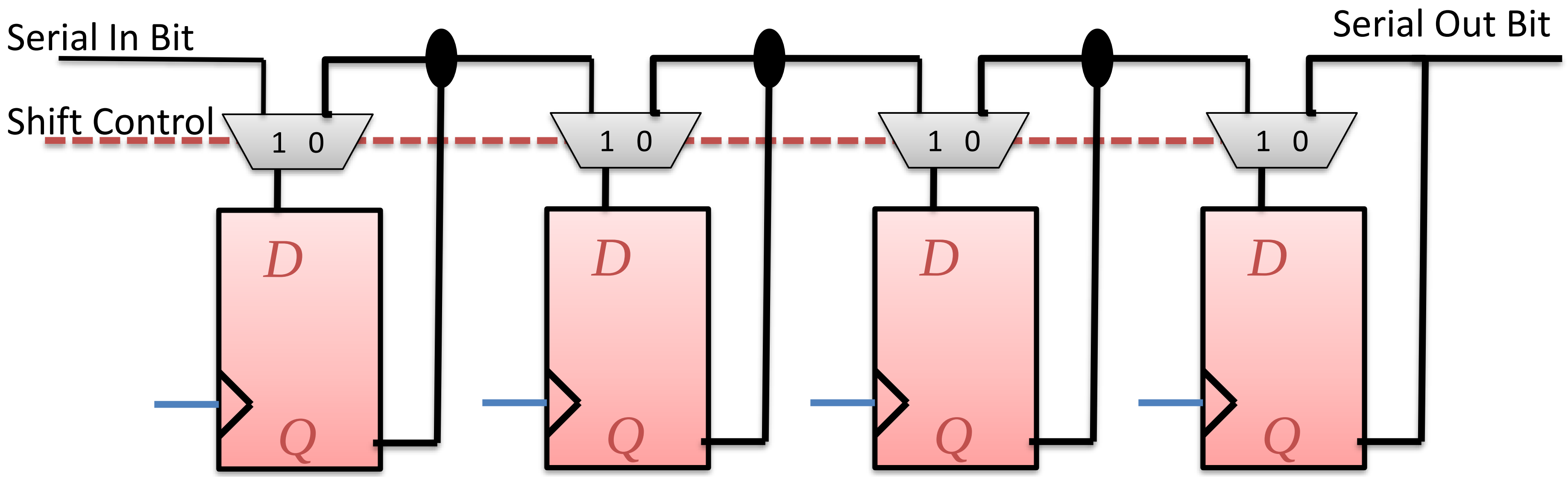
Shift Registers (SISO)





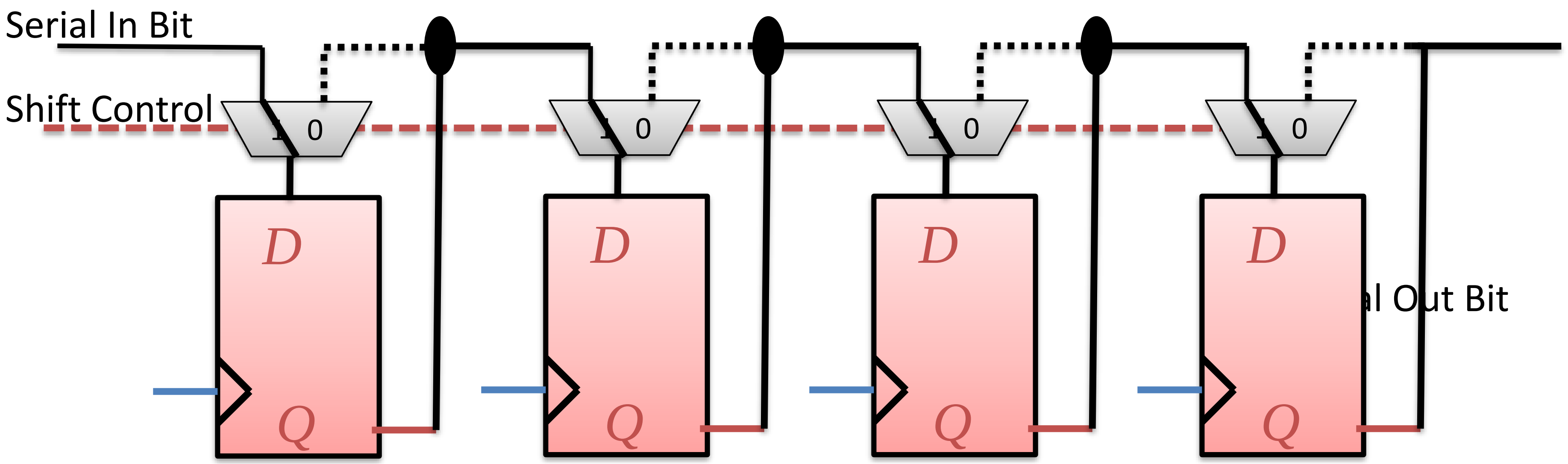
Shift Register with Control

- To allow register to either shift or retain, use 2x1 muxes
  - **Shift Control: shr: 0 means retain, 1 shift**
  - shr_in: value to shift in : May be 0/1, or right most bit



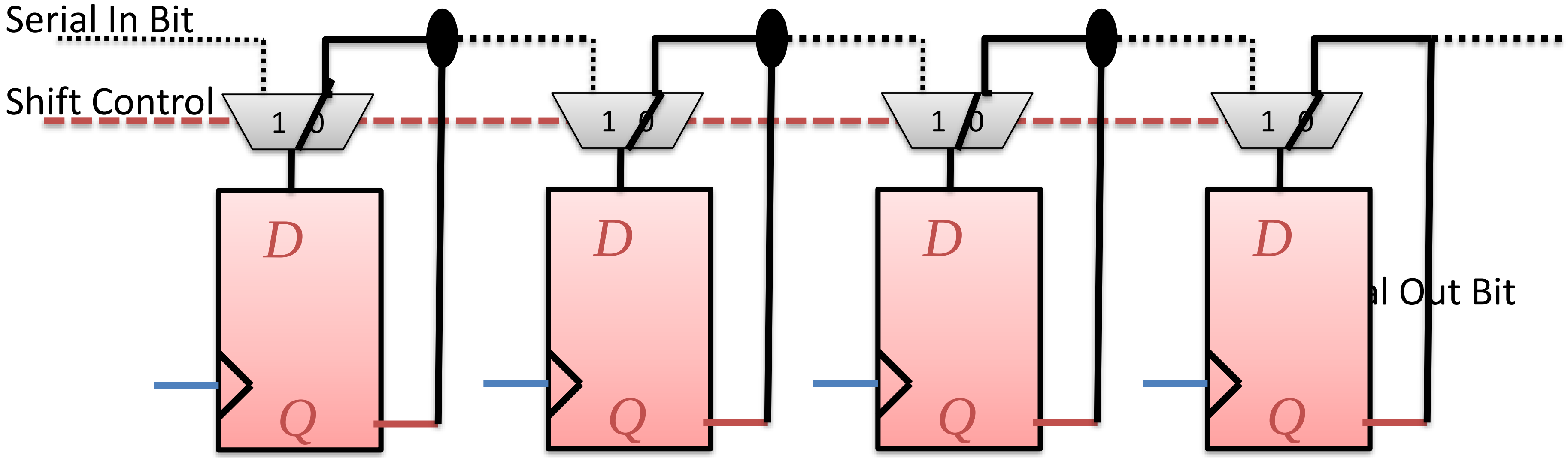
Shift Register with Control

Shift Control=1, Do one right shift per cycle



Shift Register with Control

Shift Control=0, No change to Data



Thanks