



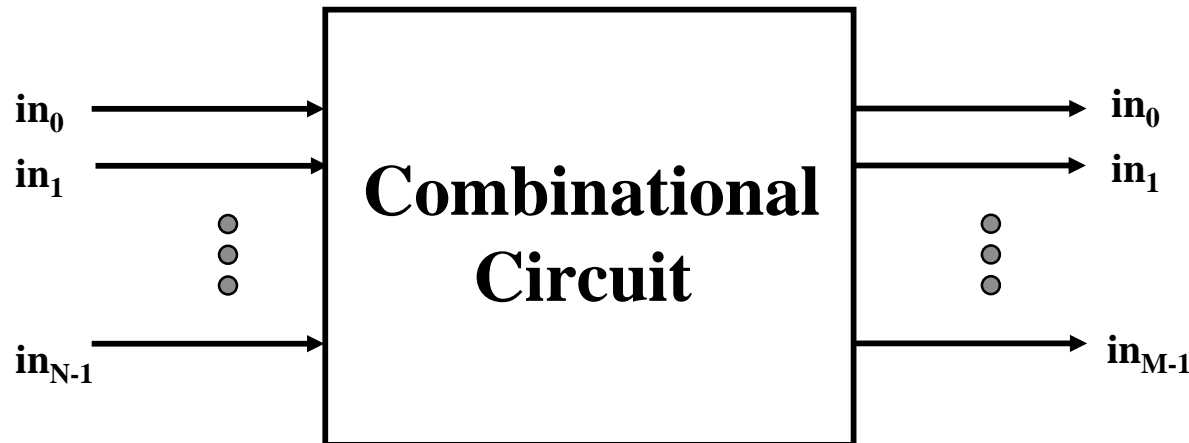
# L4: Construction and Analysis of Sequential Building Blocks



Some (most) lecture material derived from R. Katz, *“Contemporary Logic Design”*, Addison Wesley Publishing Company, Reading, MA, 1993. *Some slides are derived from slides used in past terms of 6.111*



# Combinational Logic Review

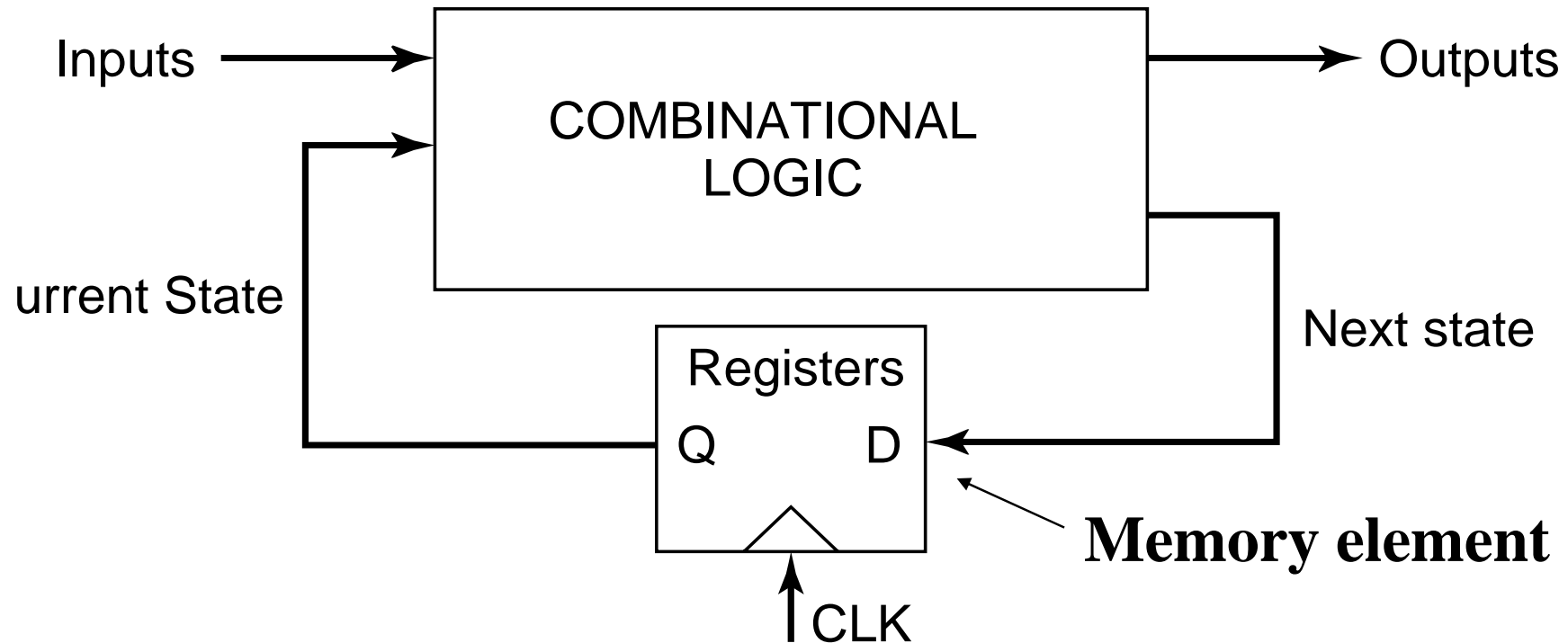


***No feedback in combinational circuits***

- **Combinational logic circuits are memoryless.**
- **There is no feedback from inputs and outputs.**
- **The output assumes the function implemented by the logic network, assuming that the switching transients have settled.**



# A Sequential System



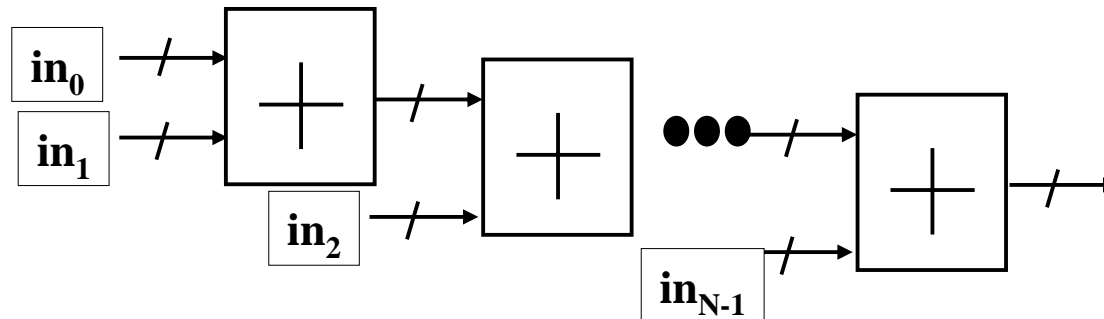
- **Sequential circuits have memory (i.e., remember the past).**
- **The current state is “held” in memory and the next state is computed based on the current state and the current inputs.**
- **In a synchronous system, the clock signal orchestrates the sequence of events.**



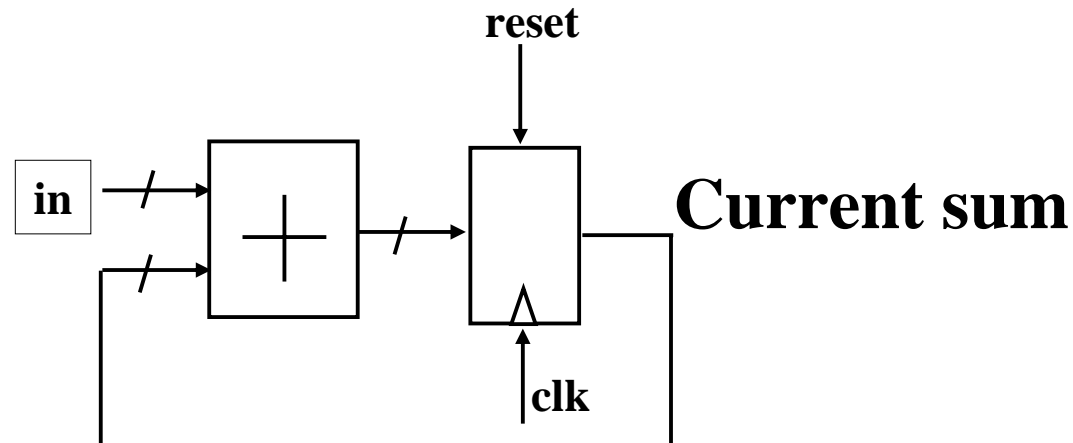
# A Simple Example



## Adding N inputs (N-1 Adders)

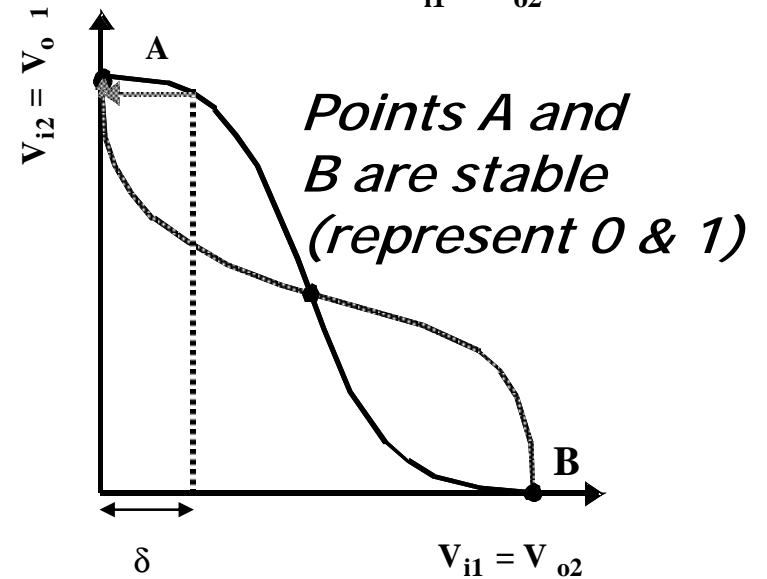
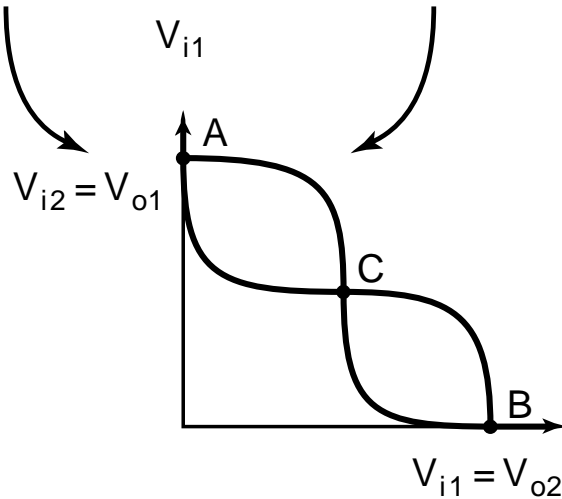
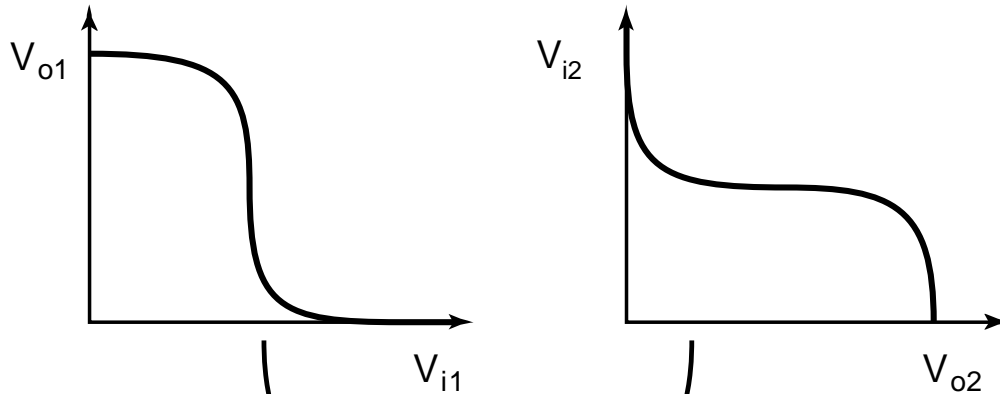
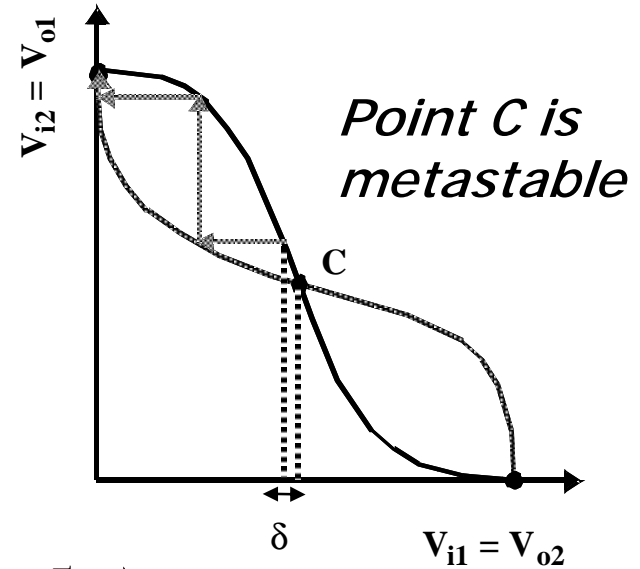
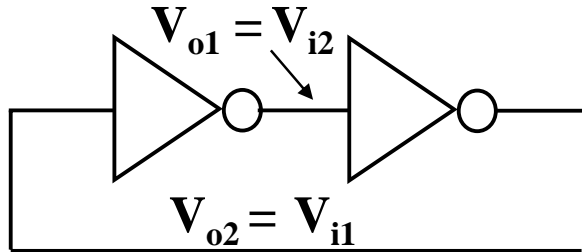


## Using a sequential (serial) approach



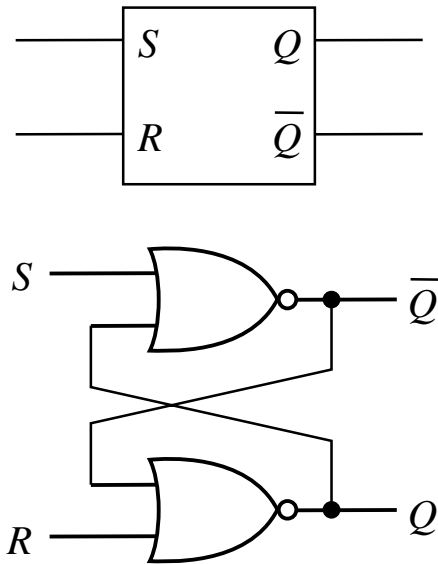


# Implementing State: Bi stability



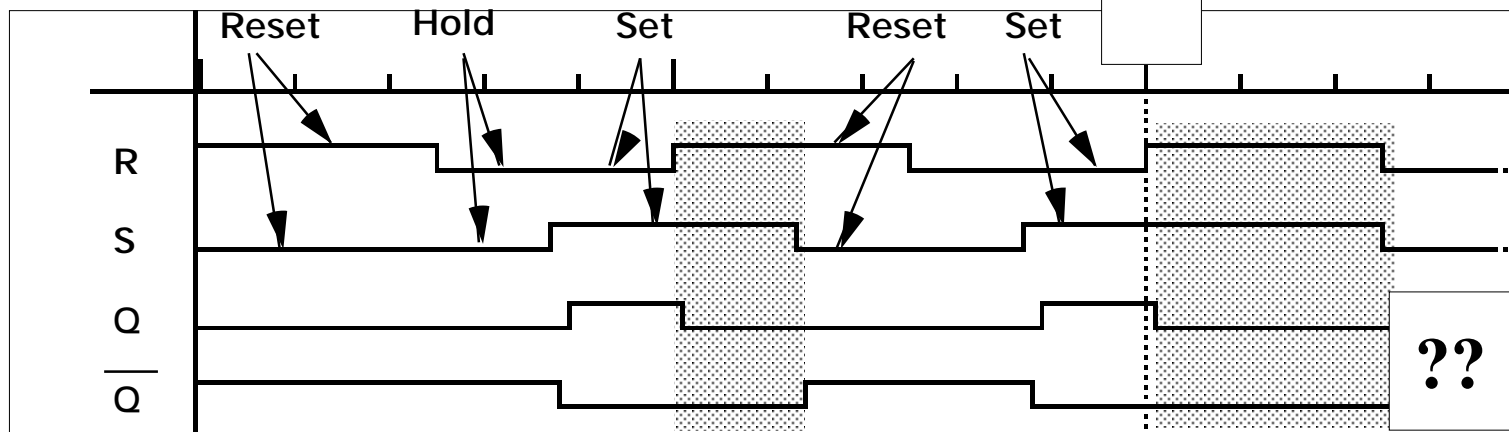
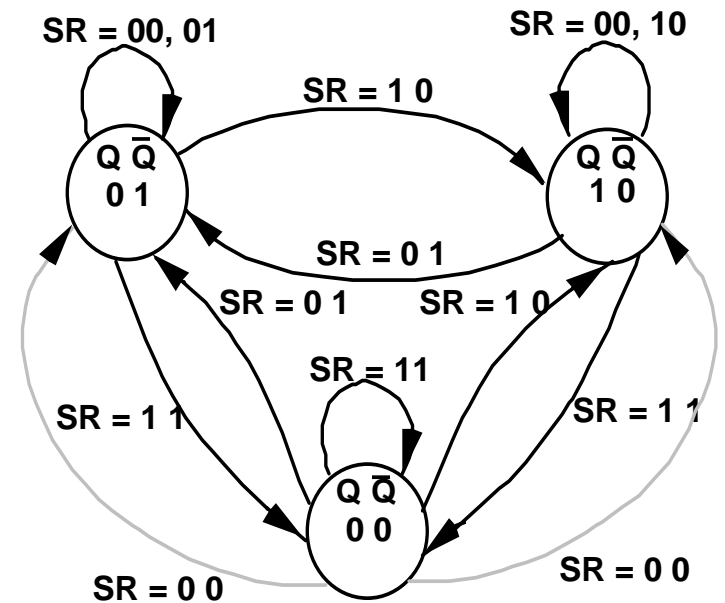


# NOR based Set Reset (SR) Flip Flop



S	R	Q	$\bar{Q}$
0	0	Q	$\bar{Q}$
1	0	1	0
0	1	0	1
1	1	0	0

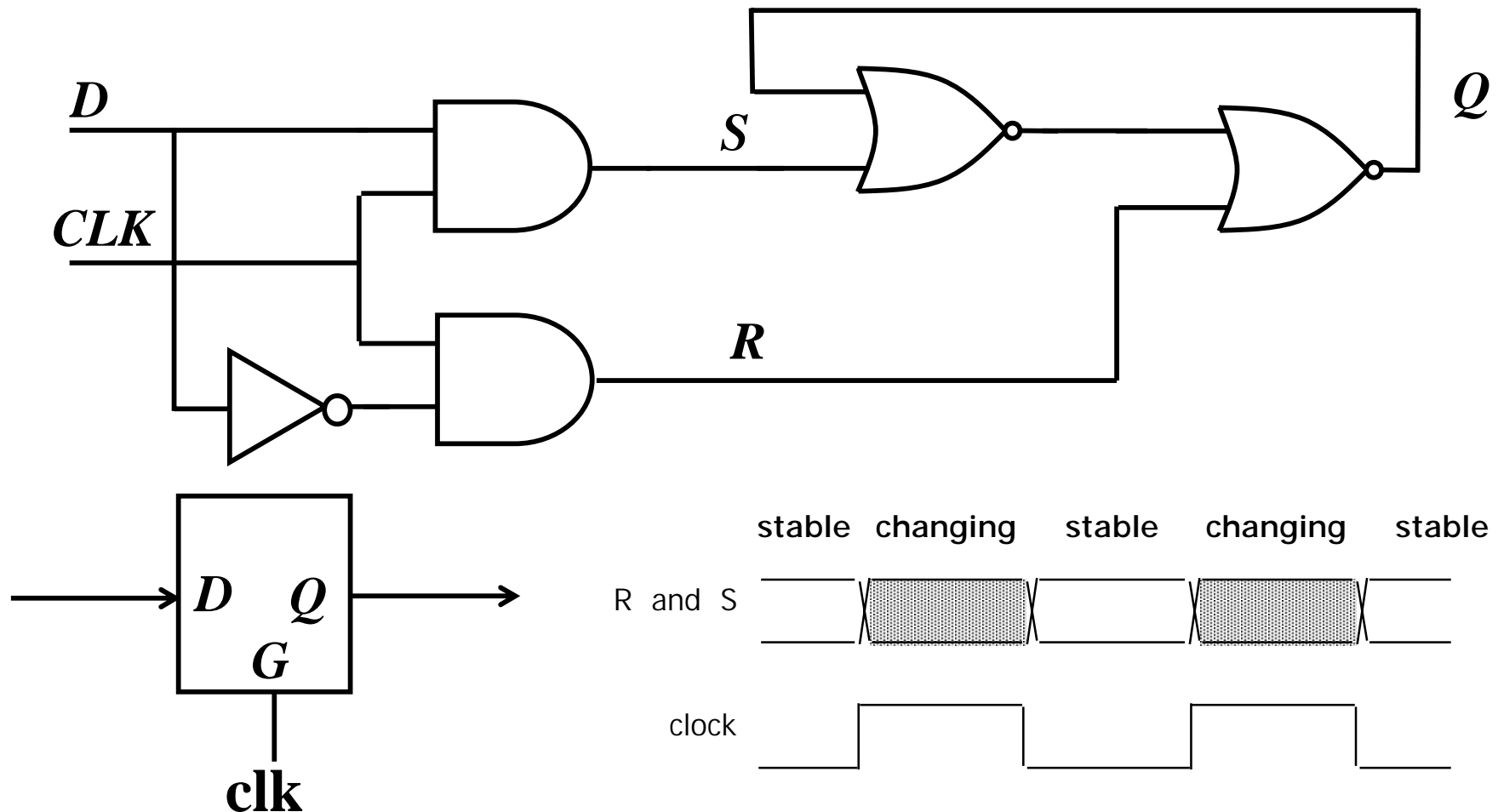
Forbidden State



- Flip flop refers to a bi stable element (edge triggered registers are also called flip flops). This circuit is not clocked and outputs change “asynchronously” with the inputs.



# Making a Clocked Memory Element: Positive $\mathcal{D}$ Latch



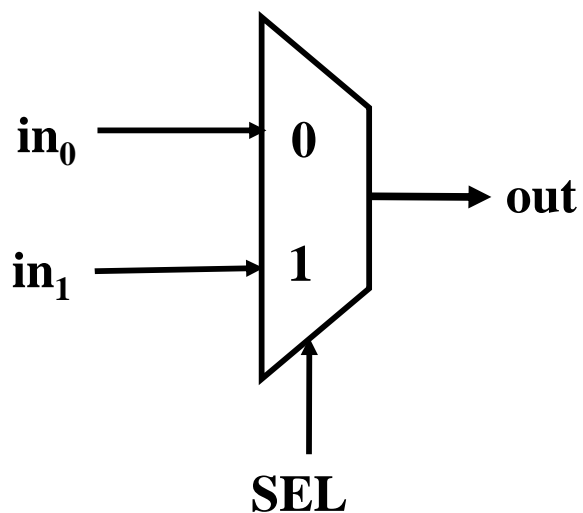
- A Positive  $\mathcal{D}$  Latch: passes input  $D$  to  $Q$  when  $clk$  is high and holds state when clock is low (i.e., ignores input  $D$ ).
- A Latch is level sensitive: invert clock for a negative latch.



# Multiplexor Based Positive & Negative Latch

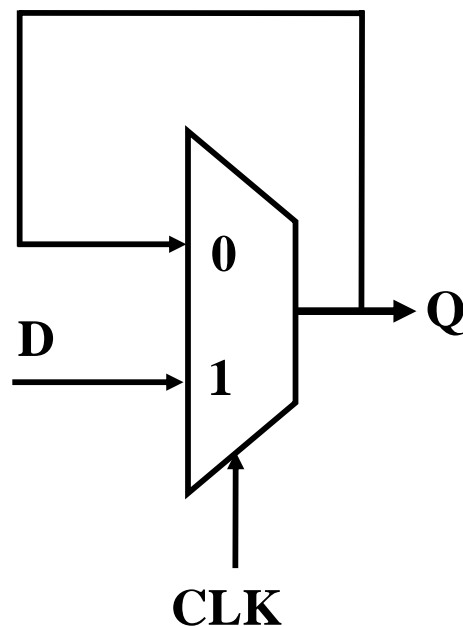


## 2:1 multiplexor

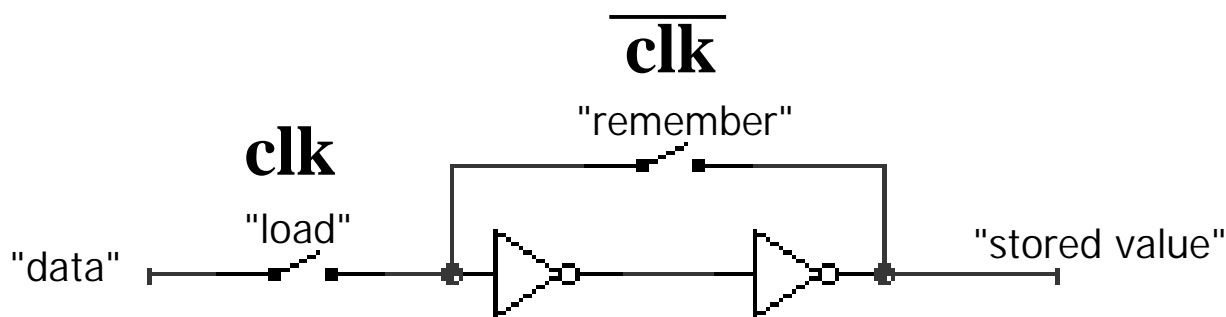
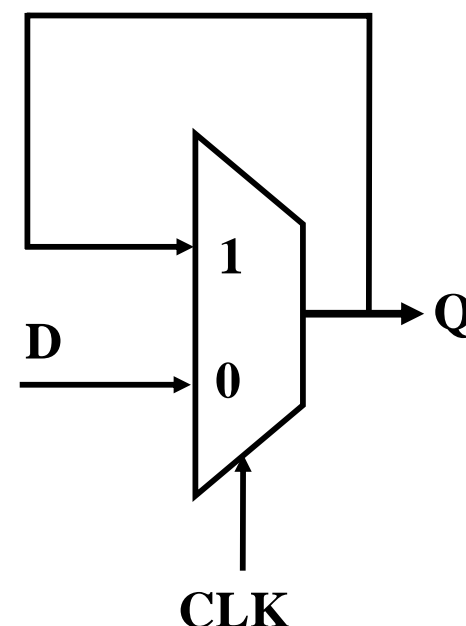


$$\text{Out} = \text{sel} * \text{in}_1 + \overline{\text{sel}} * \text{in}_0$$

## Positive Latch

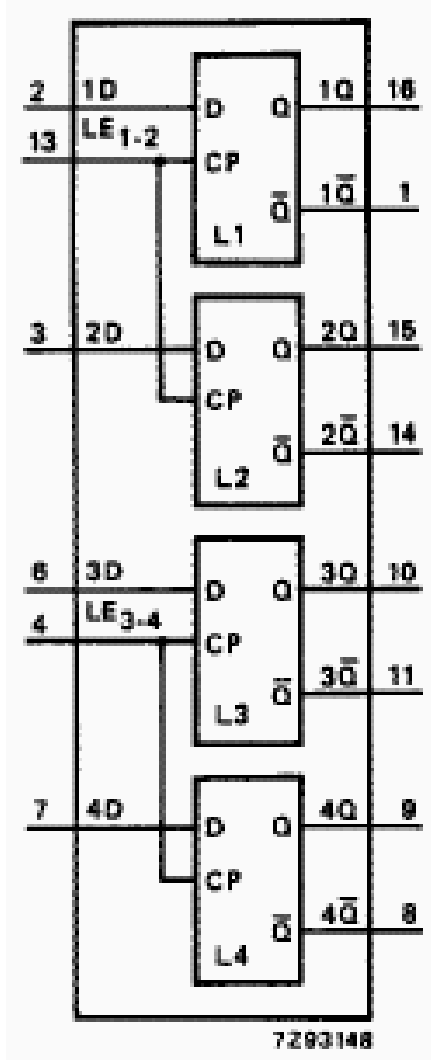


## Negative Latch





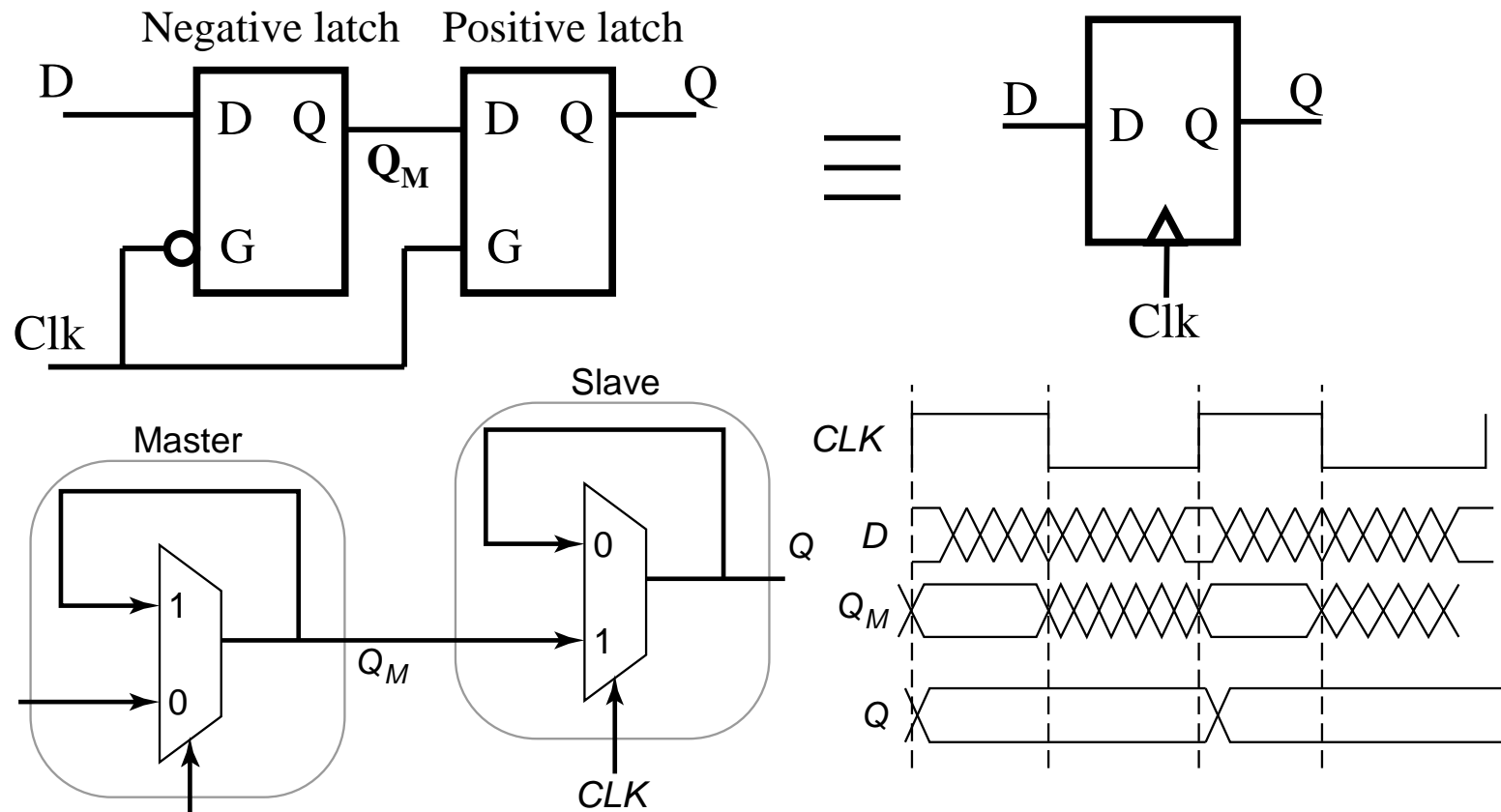
# 74HC75 (Positive Latch)



OPERATING MODES	INPUTS		OUTPUTS	
	LE <sub>n-n</sub>	nD	nQ	n $\bar{Q}$
data enabled	H	L	L	H
	H	H	H	L
data latched	L	X	q	$\bar{q}$



# Building an Edge Triggered Register



- **Master-Slave Register**

- Use negative clock phase to latch inputs into first latch.
- Use positive clock to change outputs with second latch.

- **View pair as one basic unit.**

- master-slave flip-flop twice as much logic



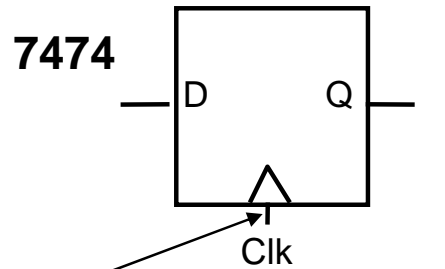
# Latches vs. Edge Triggered Register



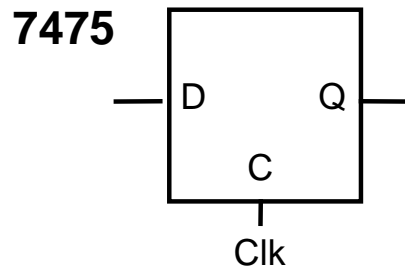
*Edge triggered* devices sample inputs on the event edge.

*Transparent latches* sample inputs as long as the clock is asserted.

Timing Diagram:

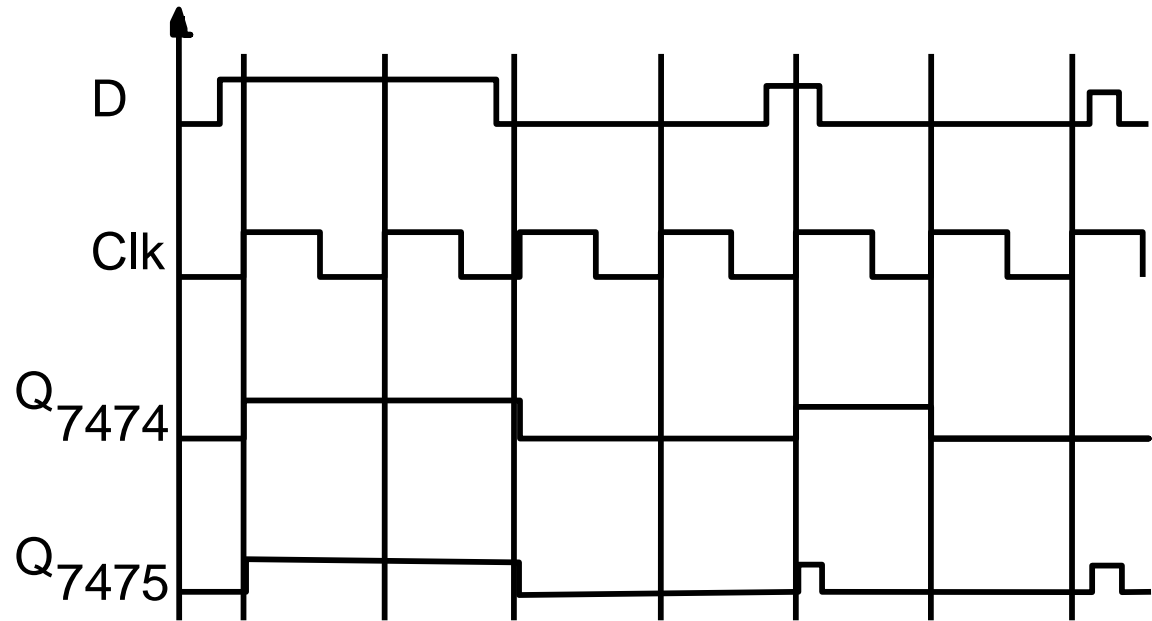


Positive edge triggered register



Level sensitive latch

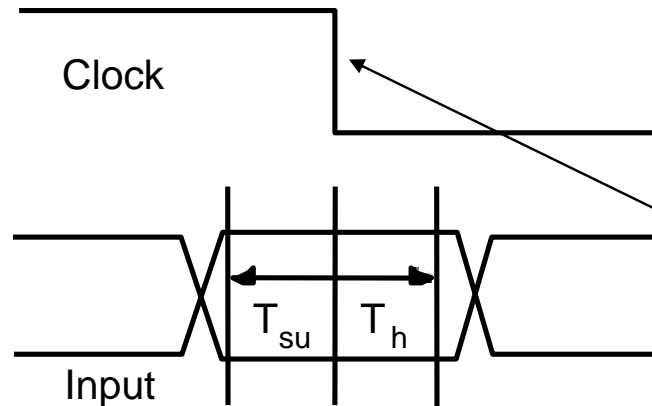
Bubble here for negative edge triggered register



*Behavior the same unless input changes while the clock is high*



# Important Timing Parameters



## ***Clock:***

**Periodic event, causes state of memory element to change**

**Memory element can be updated on the rising edge, falling edge, high level, low level.**

## ***Setup Time ( $T_{su}$ )***

**Minimum time before the clocking event by which the input must be stable**

## ***Hold Time ( $T_h$ )***

**Minimum time after the clocking event during which the input must remain stable**

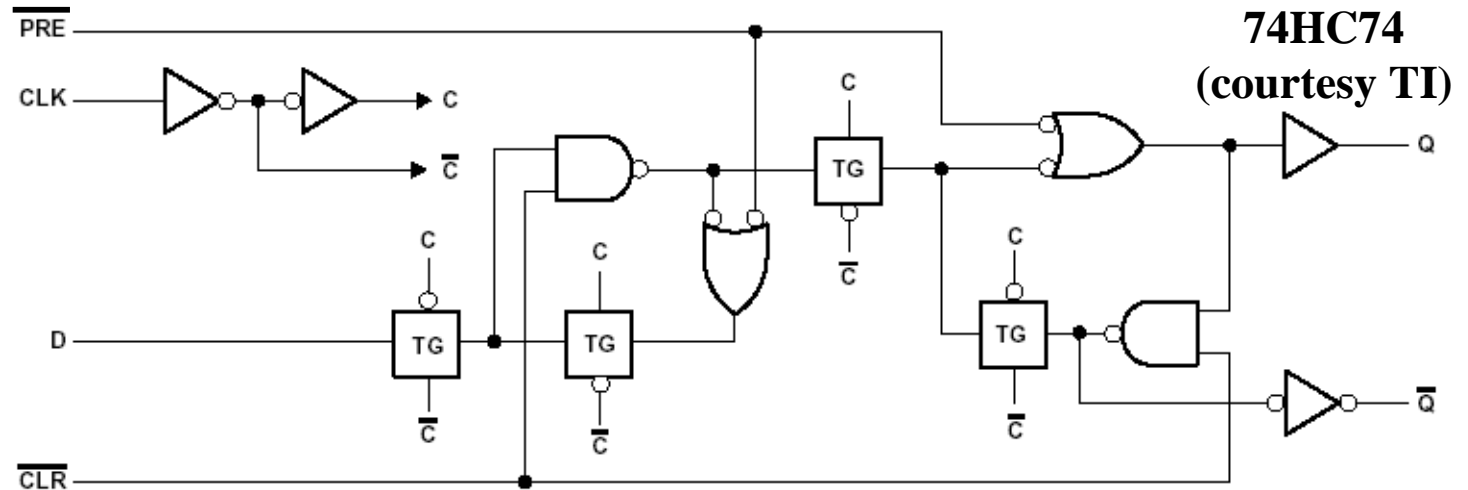
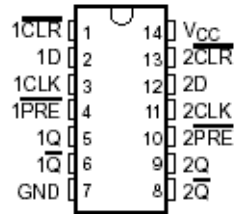
***Propagation Delay ( $T_{cq}$  for an edge triggered register and  $T_{dq}$  for a latch)***

**Delay overhead of the memory element**

**There is a timing "window" around the clocking event during which the input must remain stable and unchanged for the output to be guaranteed.**



# 74HC74 (Positive Edge Triggered Register)



FUNCTION TABLE

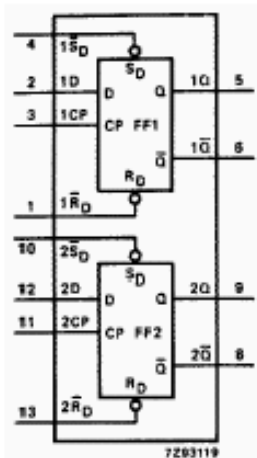
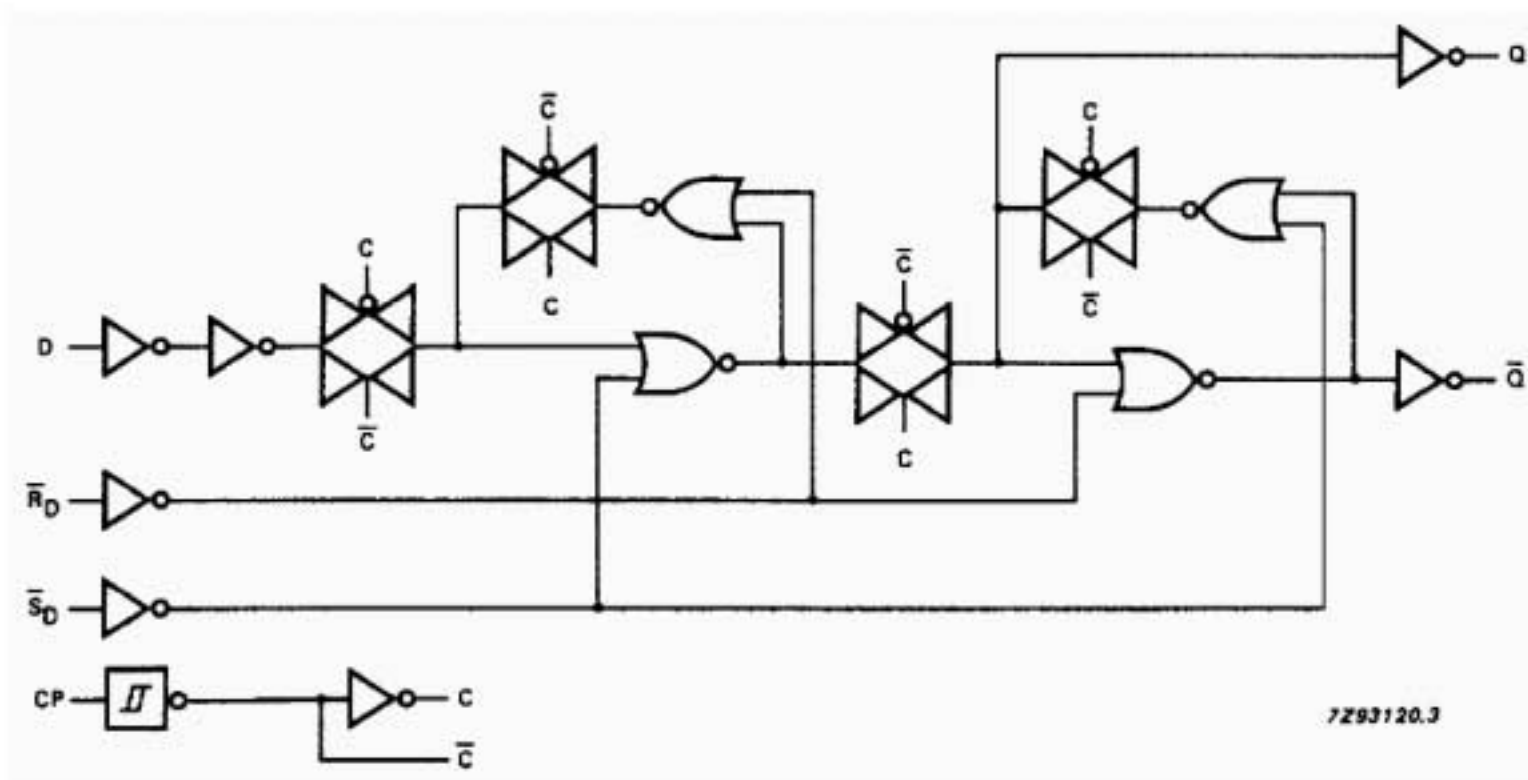
INPUTS				OUTPUTS	
PRE	CLR	CLK	D	Q	Q̄
L	H	X	X	H	L
H	L	X	X	L	H
L	L	X	X	H↑	H↑
H	H	↑	H	H	L
H	H	↑	L	L	H
H	H	L	X	Q <sub>0</sub>	Q̄ <sub>0</sub>

**D-FF with preset and clear**

		V <sub>CC</sub>	T <sub>A</sub> = 25°C		SN54HC74		SN74HC74		UNIT
			MIN	MAX	MIN	MAX	MIN	MAX	
f <sub>clock</sub>	Clock frequency	2 V	0	6	0	4.2	0	5	MHz
		4.5 V	0	31	0	21	0	25	
		6 V	0	36	0	25	0	29	
t <sub>w</sub>	Pulse duration	PRE or CLR low	2 V	100	150	125	ns		
			4.5 V	20	30	25			
			6 V	17	25	21			
		CLK high or low	2 V	80	120	100			
			4.5 V	16	24	20			
			6 V	14	20	17			
t <sub>su</sub>	Setup time before CLK↑	Data	2 V	100	150	125	ns		
			4.5 V	20	30	25			
			6 V	17	25	21			
		PRE or CLR inactive	2 V	25	40	30			
			4.5 V	5	8	6			
			6 V	4	7	5			
t <sub>h</sub>	Hold time, data after CLK↑	2 V	0	0	0	ns			
		4.5 V	0	0	0				
		6 V	0	0	0				



# Philips 74HC74 Implementation

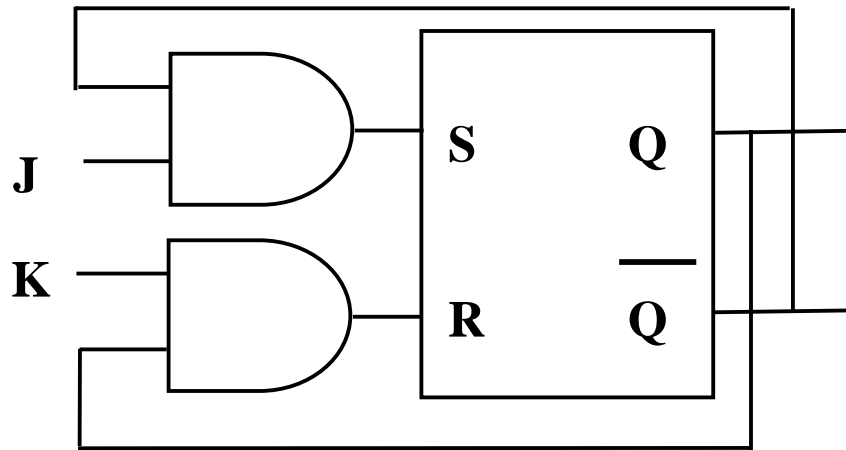


INPUTS				OUTPUTS	
$\bar{S}_D$	$\bar{R}_D$	CP	D	Q	$\bar{Q}$
L	H	X	X	H	L
H	L	X	X	L	H
L	L	X	X	H	H

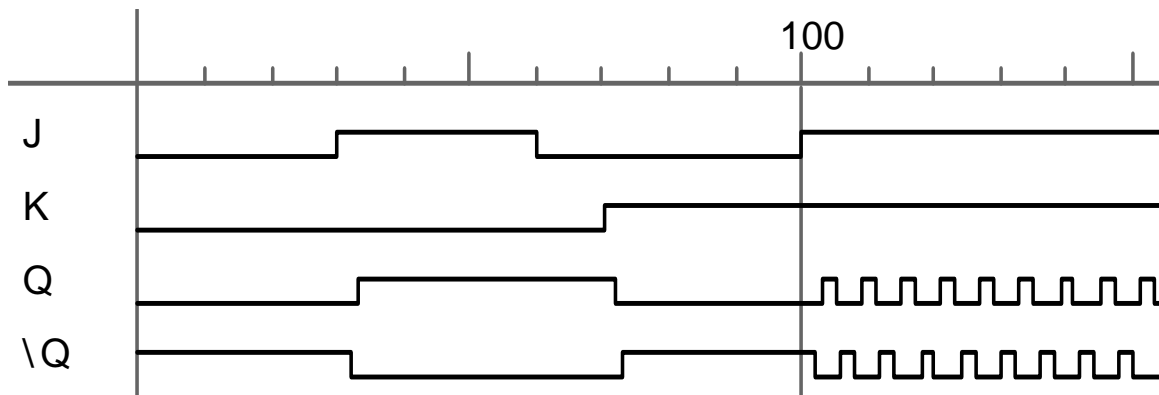
INPUTS				OUTPUTS	
$\bar{S}_D$	$\bar{R}_D$	CP	D	$Q_{n+1}$	$\bar{Q}_{n+1}$
H	H	↑	L	L	H
H	H	↑	H	H	L



# The JK Flip Flop

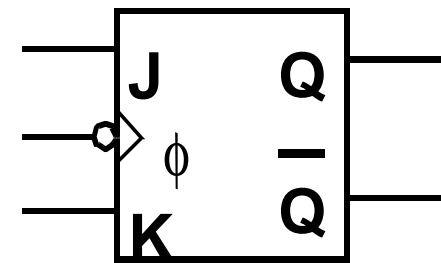
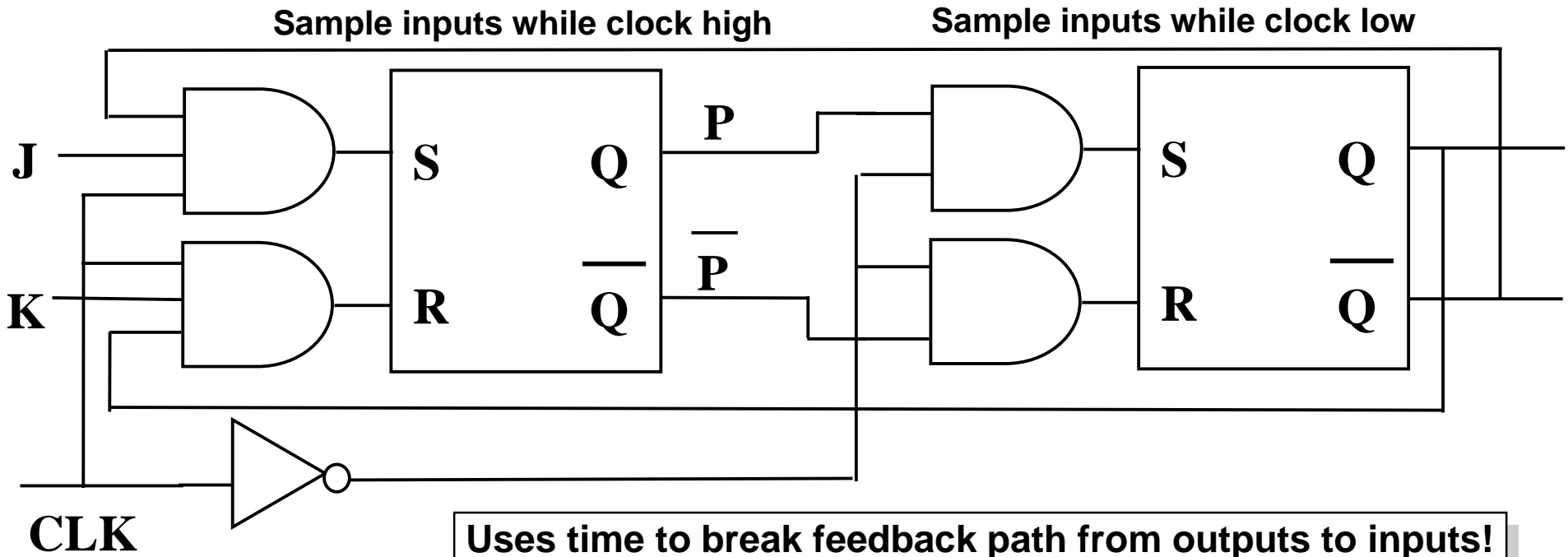


J	K	Q+	$\overline{Q}$ +
0	0	Q	$\overline{Q}$
0	1	0	1
1	0	1	0
1	1	$\overline{Q}$	Q



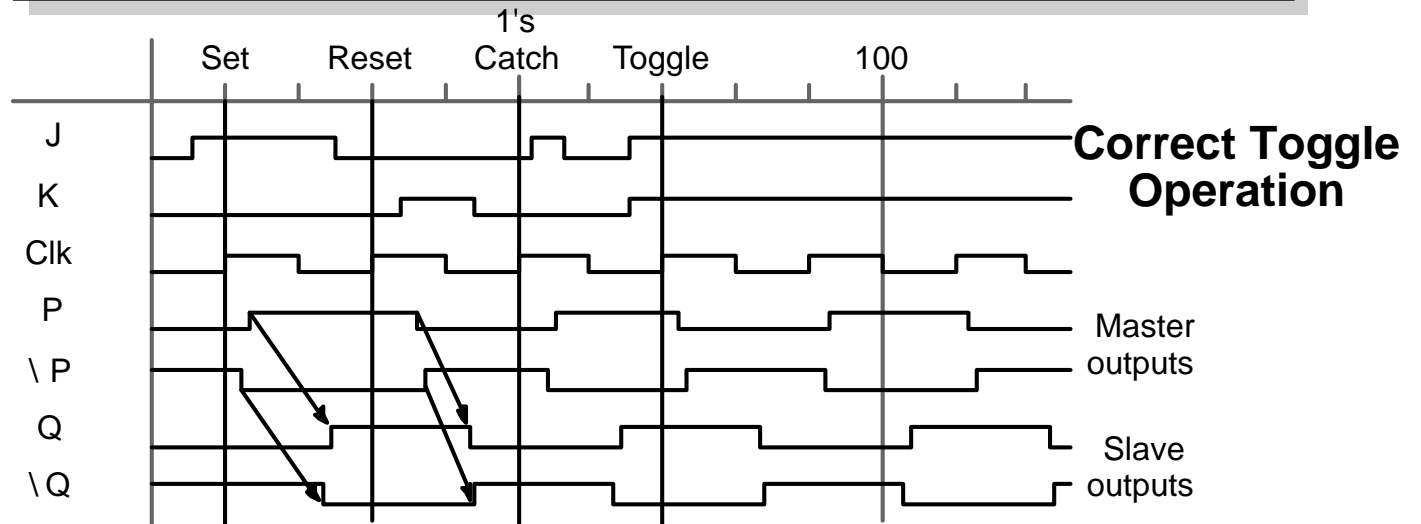
- Eliminate the forbidden state of the SR flip flop.
- Use output feedback to guarantee that R and S are never both one.

# JK Master-Slave Register



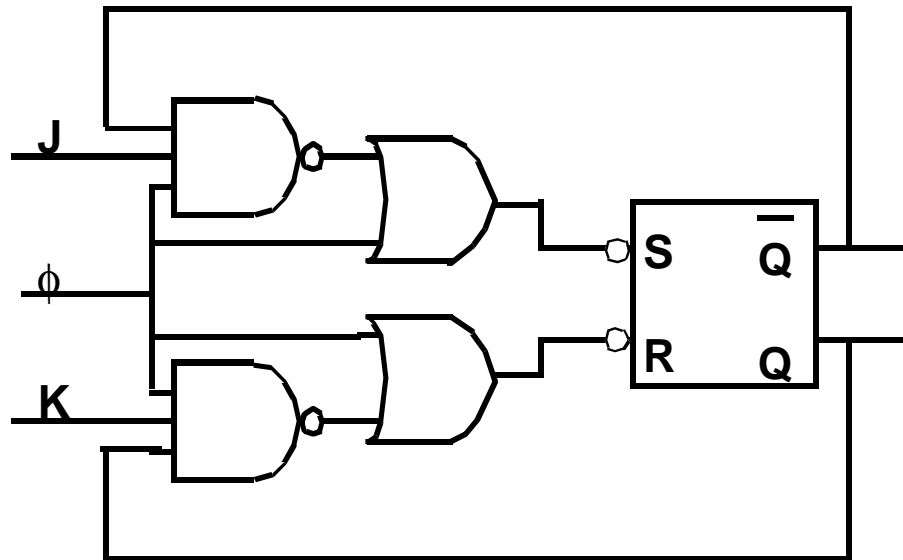
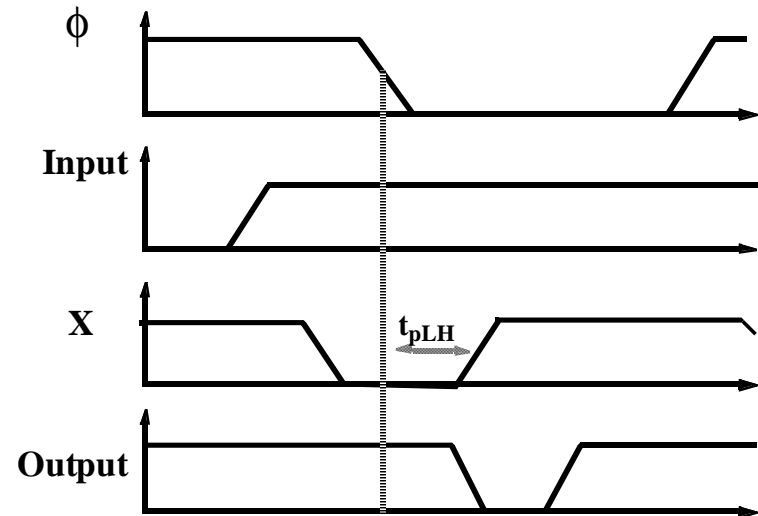
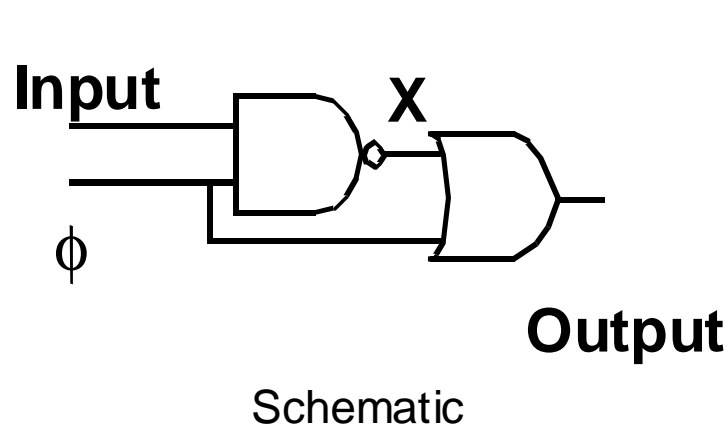
**JK Logic Symbol**

**Uses time to break feedback path from outputs to inputs!**

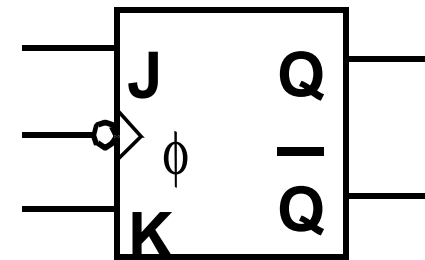




# Pulse Based Edge Triggered JK Register



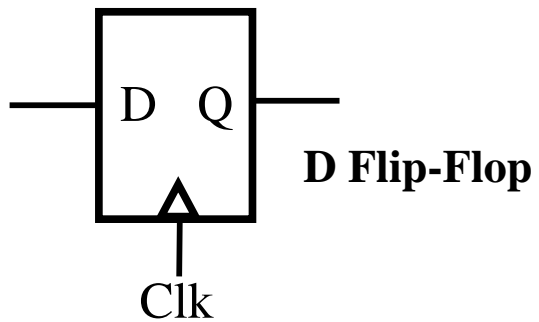
JK Register Schematic



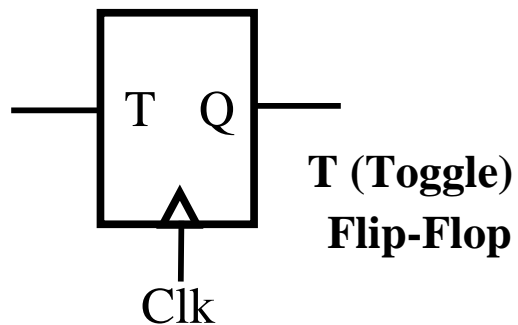
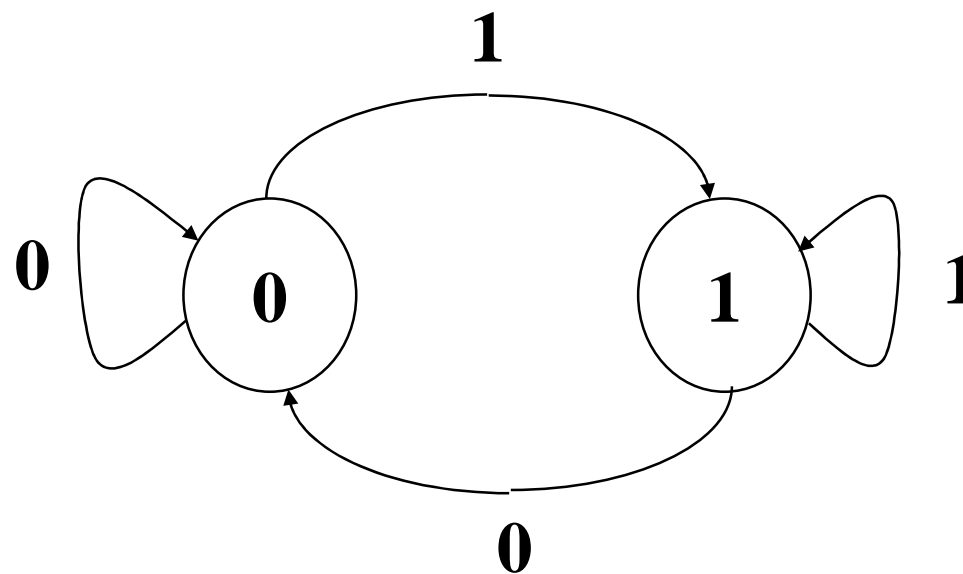
JK Register Logic Symbol



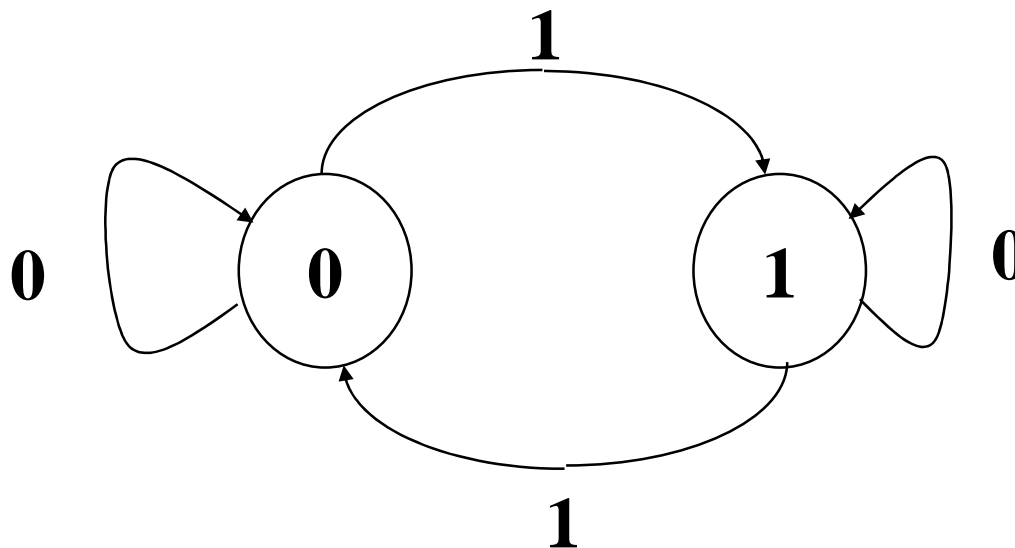
# D Flip Flop vs. Toggle Flip Flop



D	$Q_N$
0	0
1	1



T	$Q_N$
0	$Q_{N-1}$
1	$\overline{Q_{N-1}}$





# Realizing Different Types of Memory Elements



## Characteristic Equations

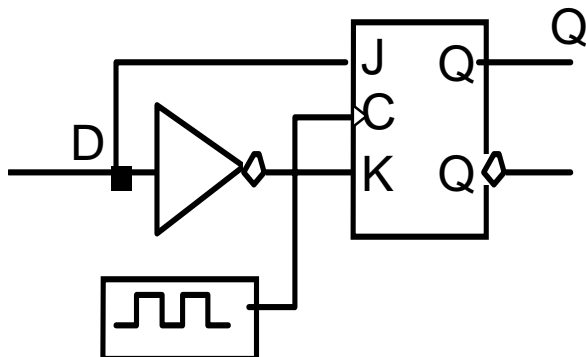
D:  $Q_+ = D$

JK:  $Q_+ = J \bar{Q} + \bar{K} Q$

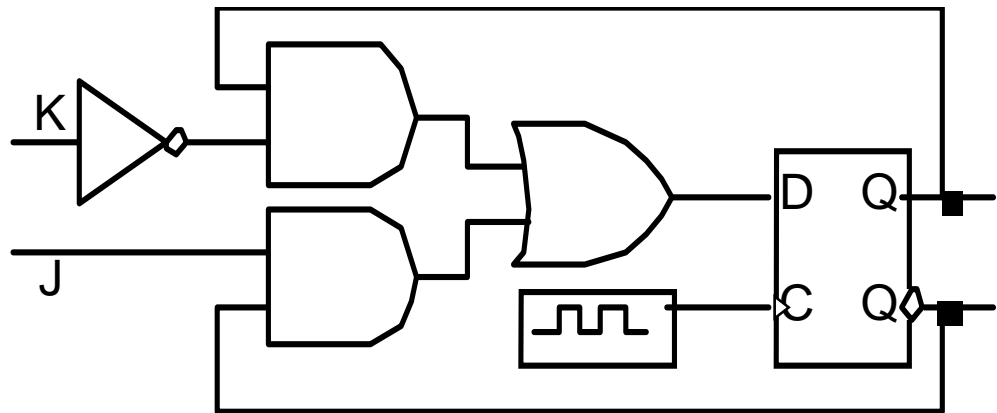
T:  $Q_+ = T \bar{Q} + \bar{T} Q$

E.g., J=K=0, then  $Q_+ = Q$   
J=1, K=0, then  $Q_+ = 1$   
J=0, K=1, then  $Q_+ = \bar{Q}$   
J=1, K=1, then  $Q_+ = \bar{Q}$

## Implementing One FF in Terms of Another



D implemented with JK



JK implemented with D



# Design Procedure



Excitation Tables: What are the necessary inputs to cause a particular kind of change in state?

Q	Q+	J	K	T	D
0	0	0	X	0	0
0	1	1	X	1	1
1	0	X	1	1	0
1	1	X	0	0	1

Implementing D FF with a J K FF:

- 1) Start with K map of  $Q^+ = f(D, Q)$
- 2) Create K maps for J and K with same inputs (D, Q)
- 3) Fill in K maps with appropriate values for J and K to cause the same state changes as in the original K map

Q \ D	0	1
0	0	1
1	0	1

$Q^+ = D$

E.g.,  $D = Q = 0, Q^+ = 0$   
then  $J = 0, K = X$

Q \ D	0	1
0	0	1
1	X	X

$J = D$

Q \ D	0	1
0	X	X
1	1	0

$K = \overline{D}$

# Design Procedure (cont.)



Implementing JK FF with a D FF:

1) K-map of  $Q^+ = F(J, K, Q)$

2,3) Revised K-map using D's excitation table

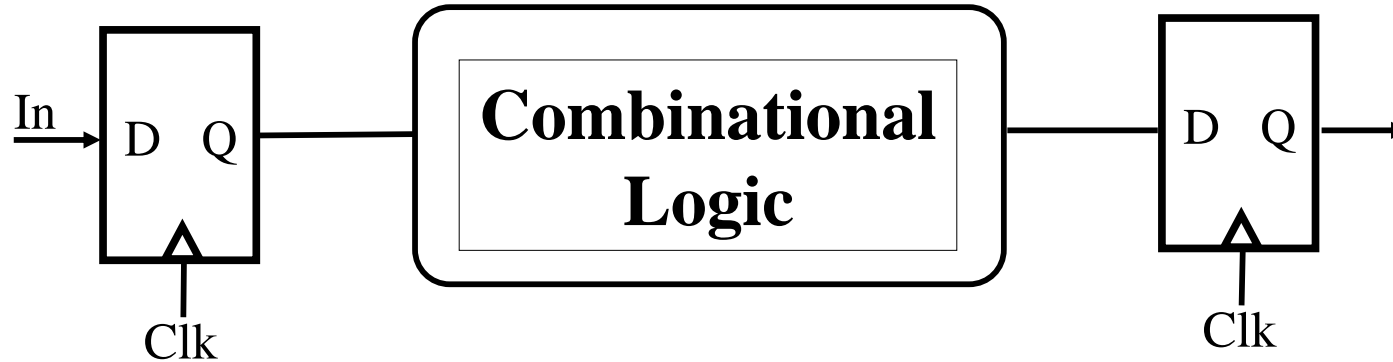
It's the same! That is why design procedure with D FF is simple!

Q	JK		J	
	00	01	11	10
0	0	0	1	1
1	1	0	0	1

$$Q^+ = D = J\bar{Q} + \bar{K}Q$$

Resulting equation is the combinational logic input to D to cause same behavior as JK FF. Of course, it is identical to the characteristic equation for a JK FF.

# System Timing Parameters



## Register Timing Parameters

$T_{cq}$  : worst case rising edge  
clock to q delay

$T_{cq, cd}$  : contamination or  
minimum delay from  
clock to q

$T_{su}$  : setup time

$T_h$  : hold time

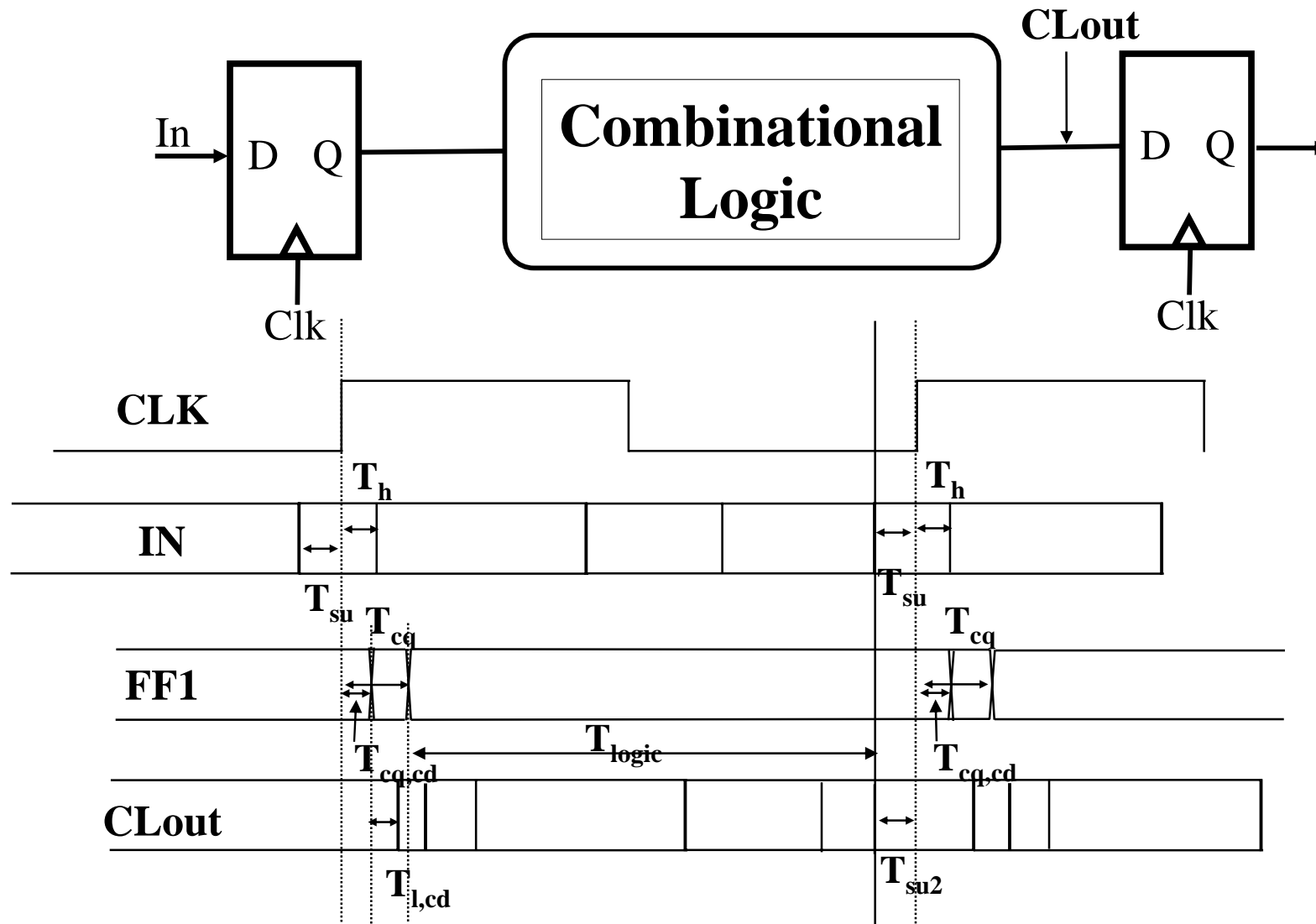
## Logic Timing Parameters

$T_{logic}$  : worst case delay  
through the combinational  
logic network

$T_{logic, cd}$  : contamination or  
minimum delay  
through logic network



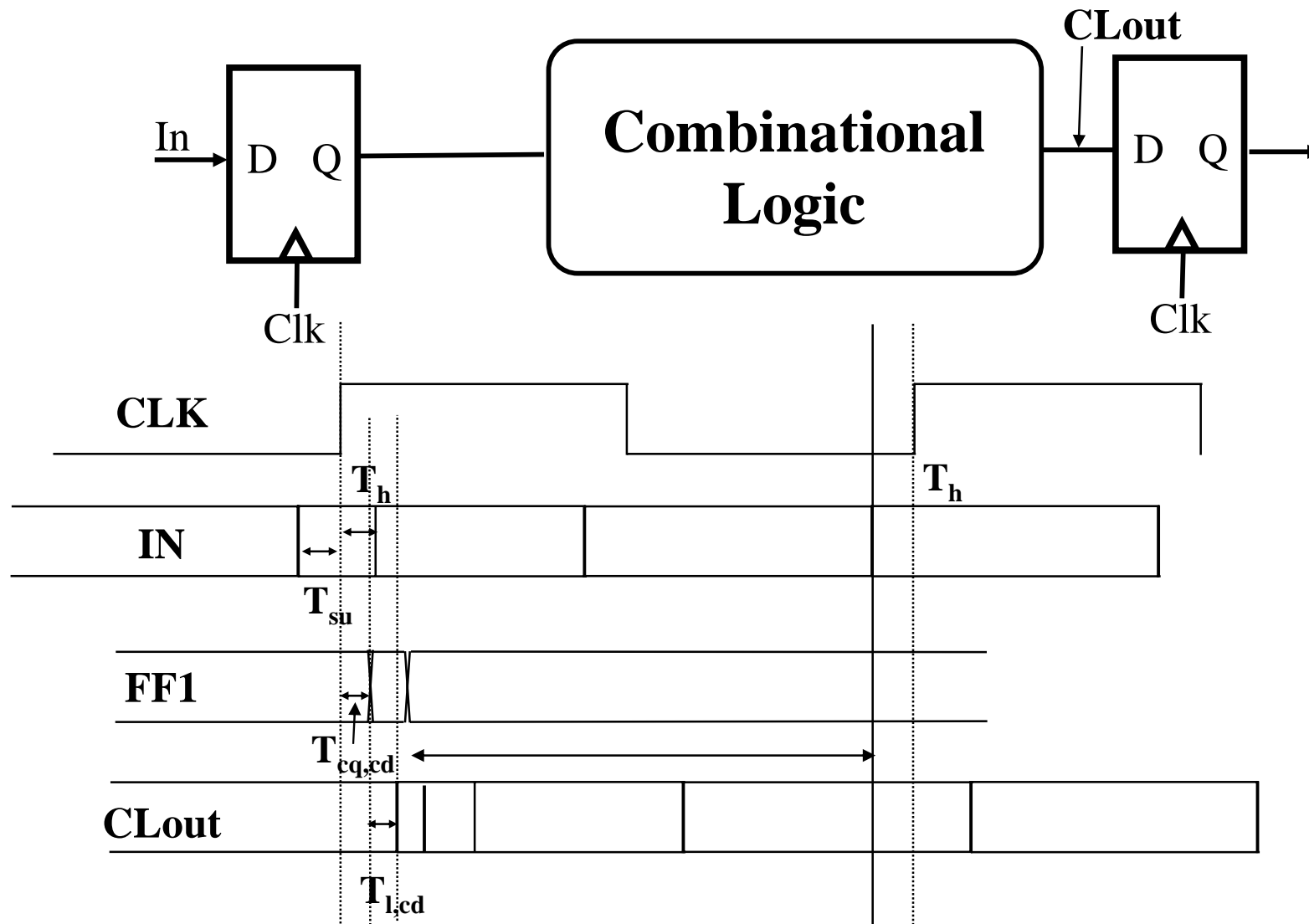
# System Timing (I): Minimum Period



$$T > T_{cq} + T_{logic} + T_{su}$$



# System Timing (II): Minimum Delay



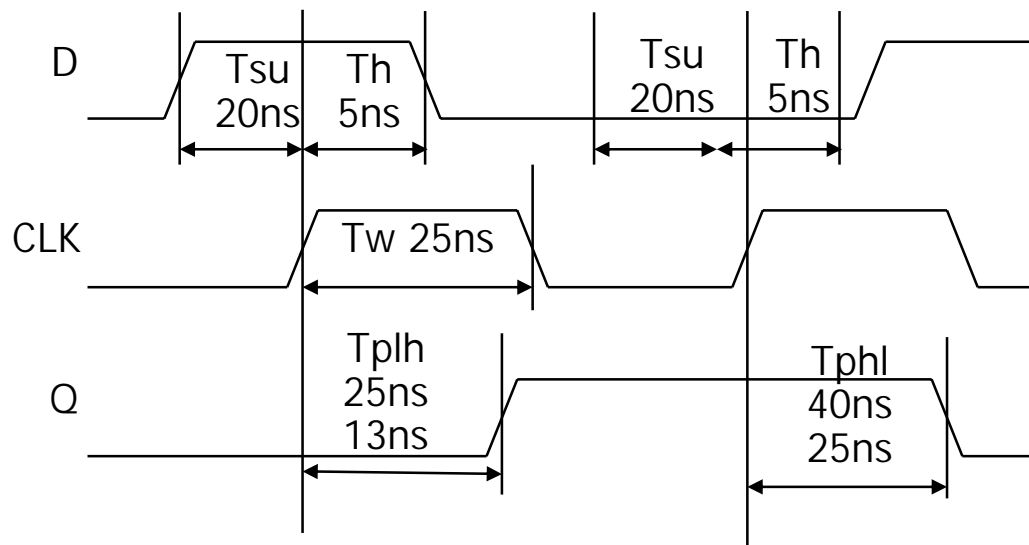
$$T_{cq,cd} + T_{logic,cd} > T_{hold}$$



# Shift Register

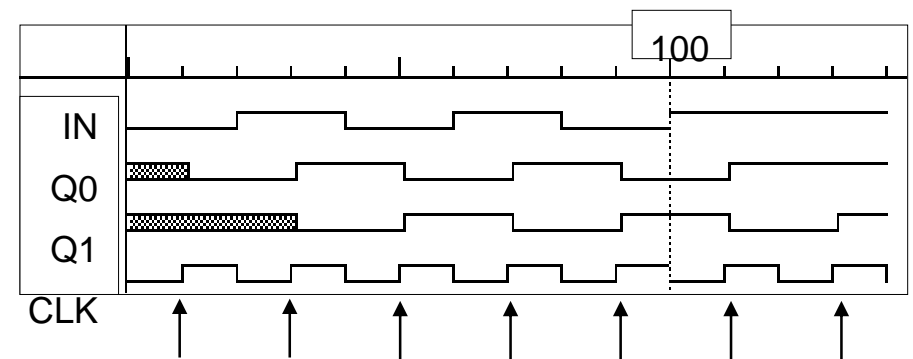
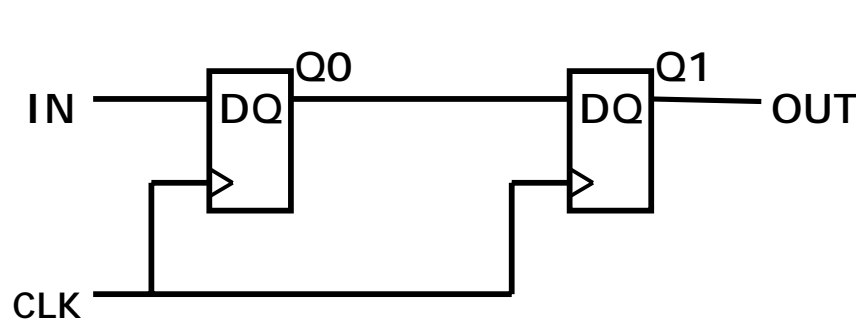


## ■ Typical parameters for positive edge triggered D Register



All measurements are made from the clocking event, that is, the rising edge of the clock.

## ■ Shift register





# Clocks are Not Perfect: Clock Skew

