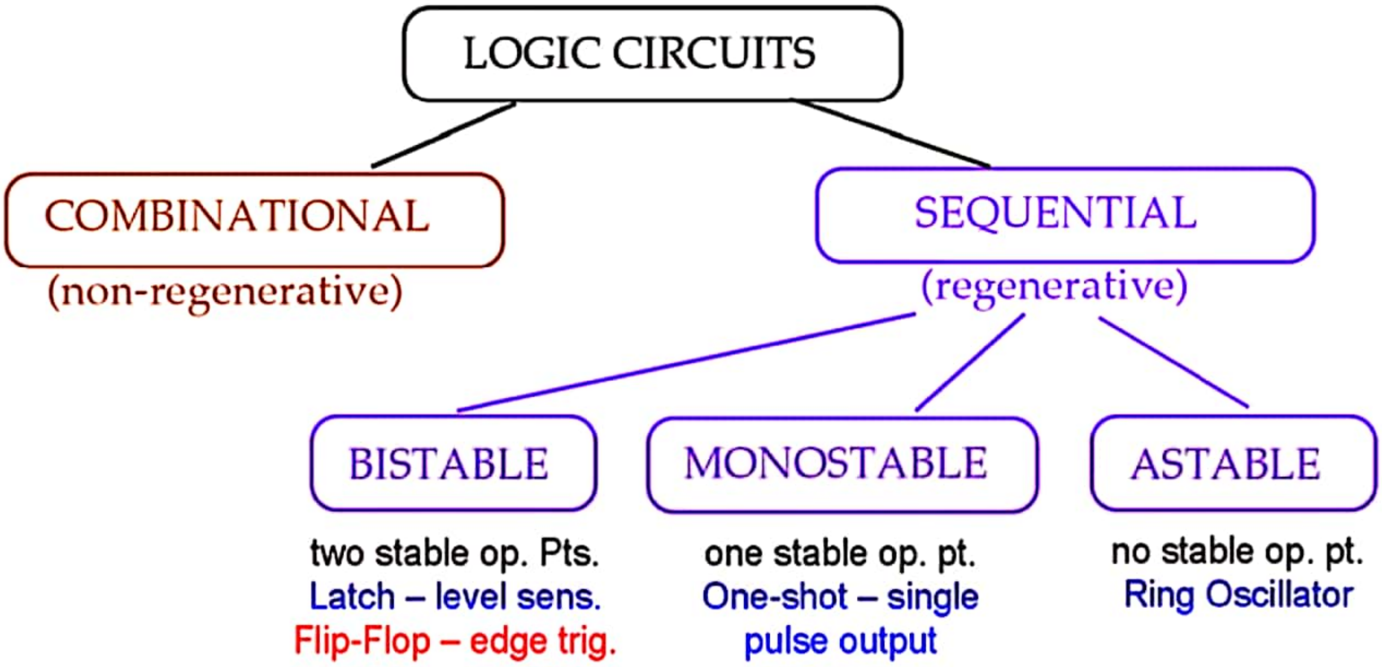
Sequential Circuits

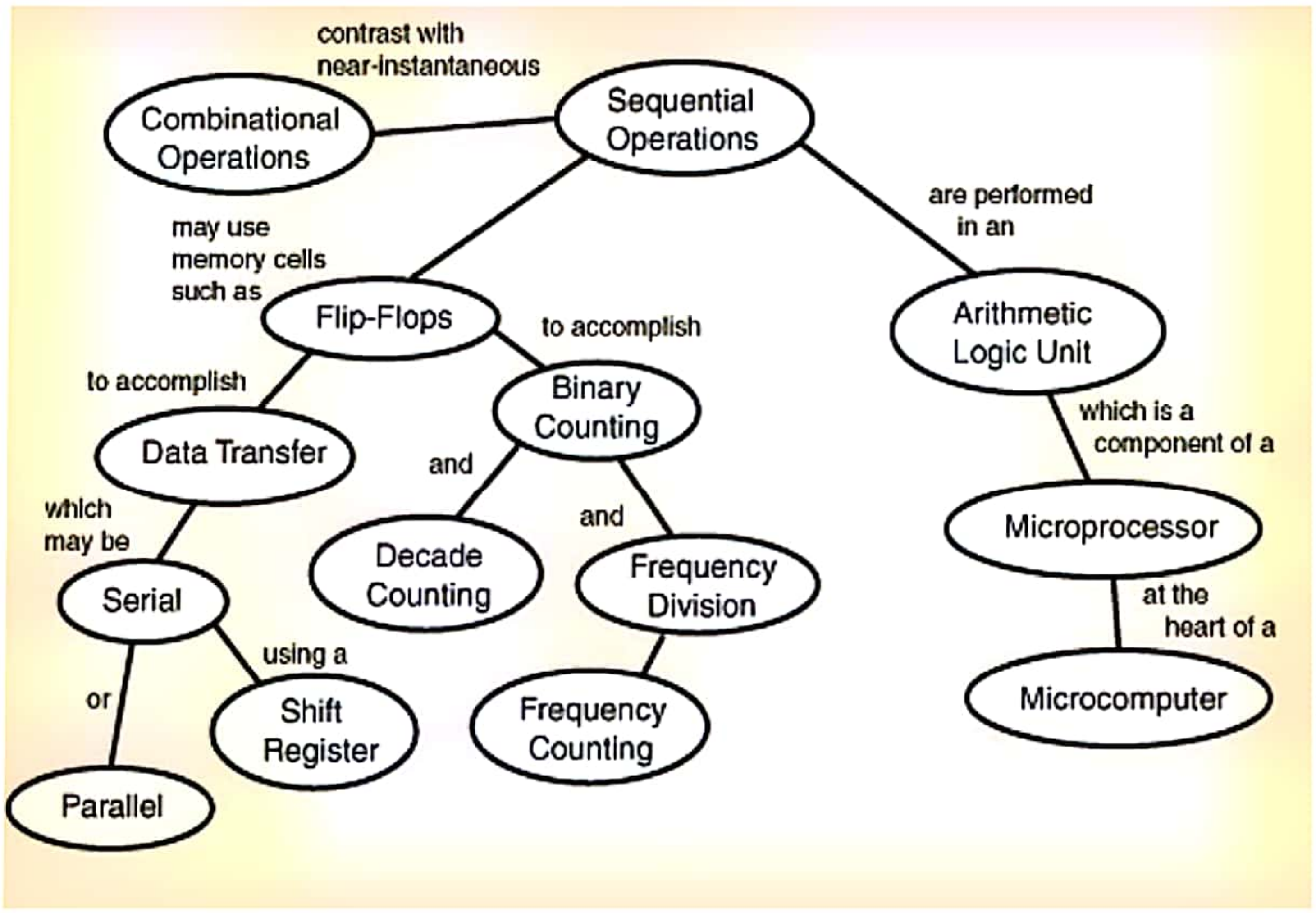
Classes of Logic Circuits



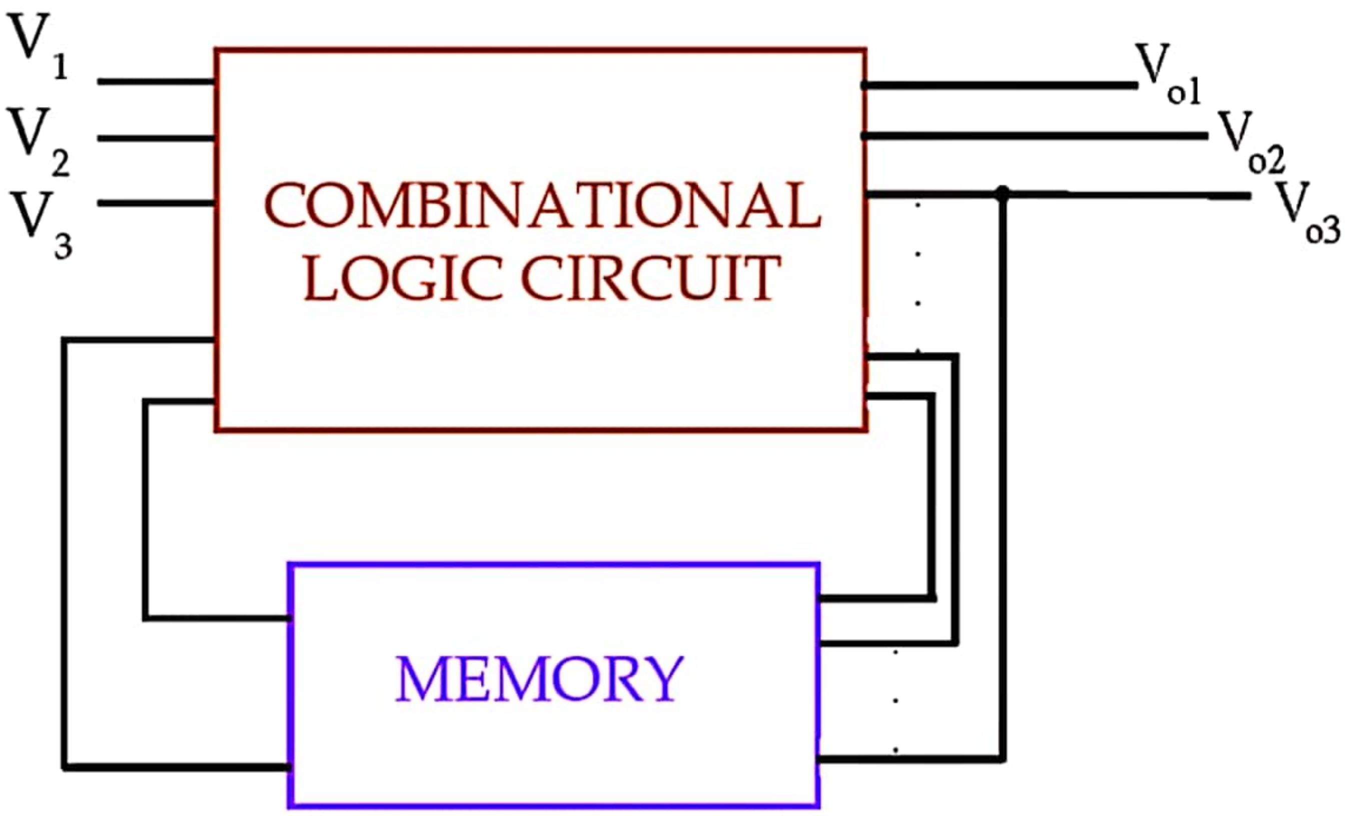
Combinational Circuits: Current Output(s) depend ONLY on Current Inputs.

Sequential Circuits: Current Output(s) depend on Current Inputs and PAST Output(s).

Functions Using Sequential Operations



Sequential Circuit Construct

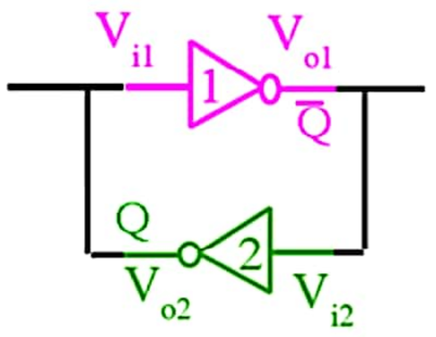


-> Sequential Circuits: Current Output(s) depend on Current Inputs and PAST inputs (via the feedback of some past State(s) and Output(s) to inputs).

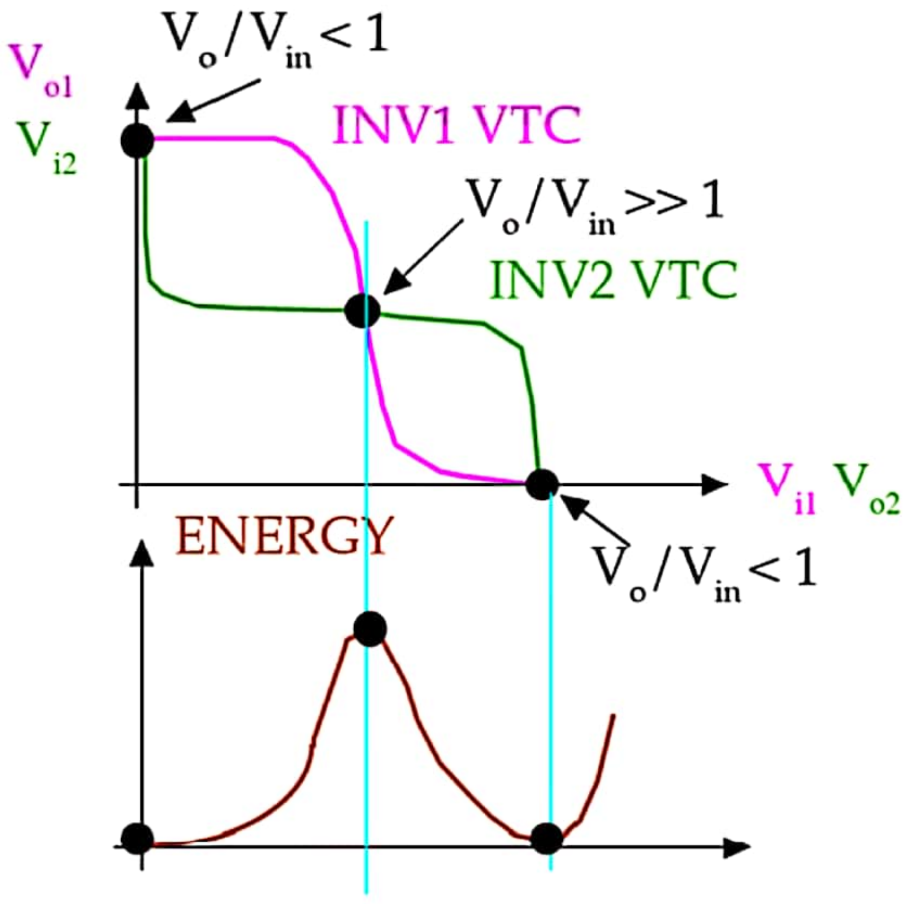
-> Memory is used to Store Past Values of State(s) and Output(s).

Bistable Sequential Circuits

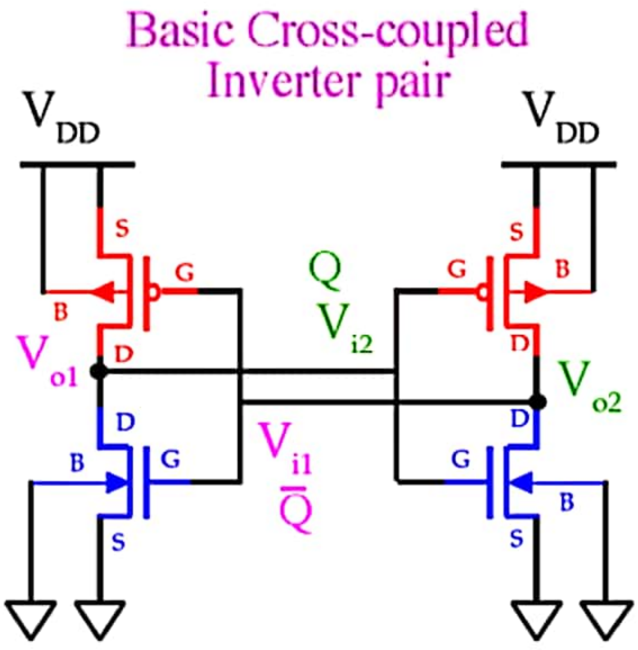
Basic Cross-coupled Inverter pair

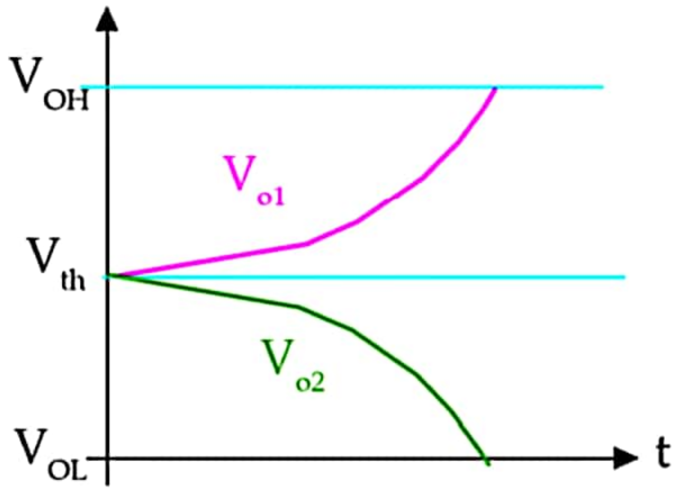


BISTABLE BEHAVIOR



Bistable Sequential Circuits - cont.

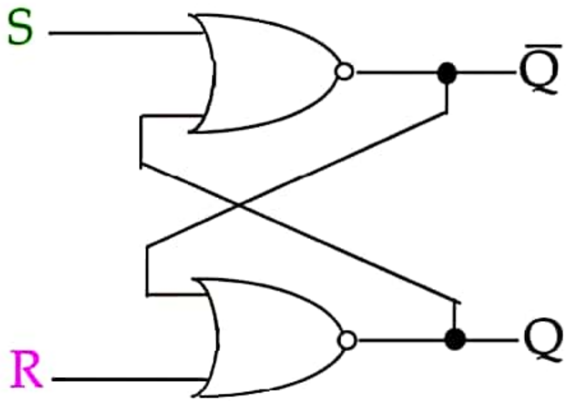


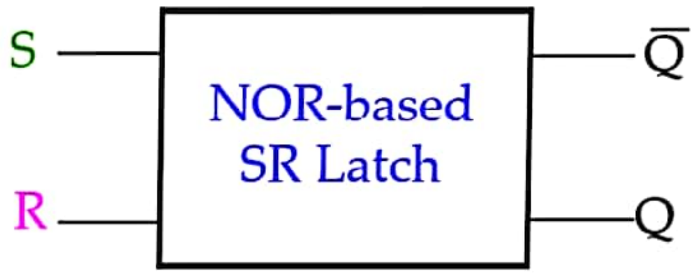


STATIC: V_{DD} is required to maintain stable state.

Basic Bistable Cross-coupled Inverter Pair has no means to apply input(s) to change the circuit's State.

Unclocked Latch Circuits





t = t1 > t = t0

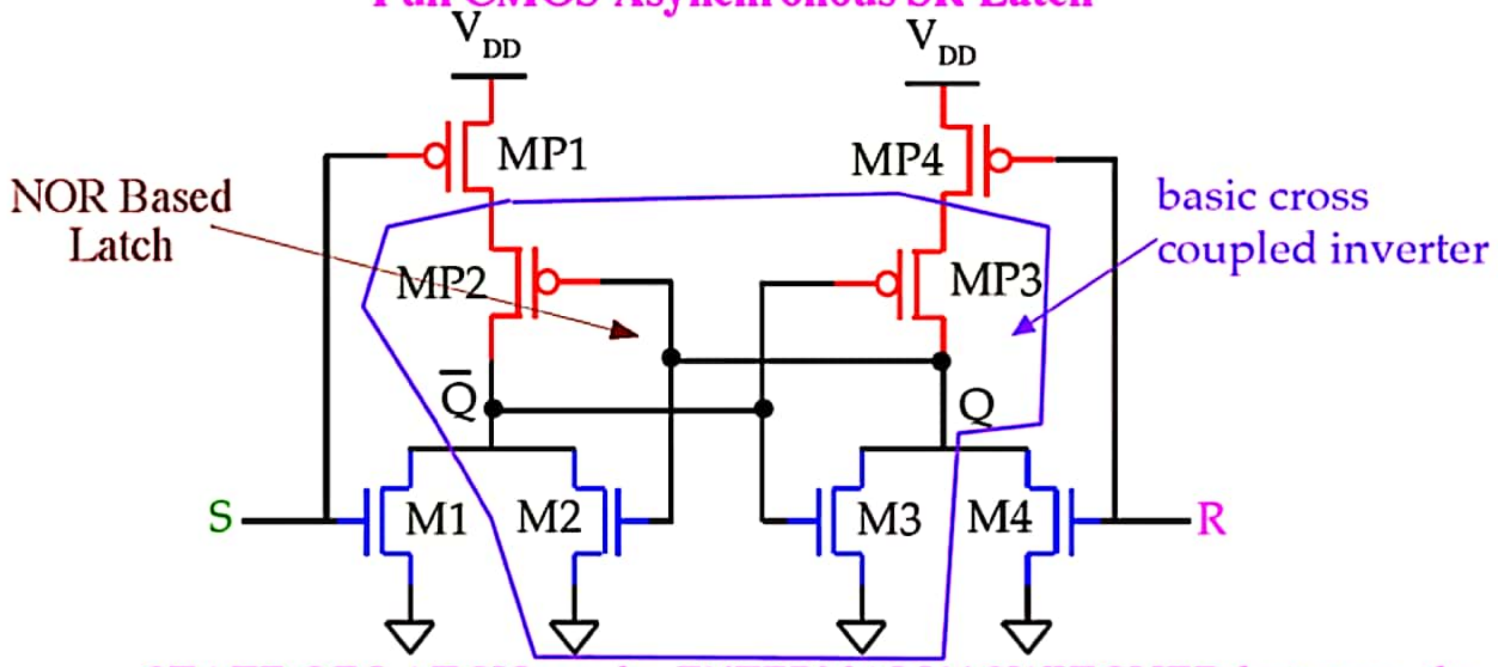
S_{t1}	R_{t1}	Q_{t1}	$\overline{Q}_{t1}$	Operation
0	0	Q_n	$\overline{Q}_n$	hold
1	0	1	0	set*
0	1	0	1	reset*
1	1	0	0	NOT allowed

ACTIVE HIGH

*Data is written by over powering the feedback loop using S, R inputs.

Unclocked Latch Circuits

Full CMOS Asynchronous SR Latch



STATE OF LATCH can be EXTERNALLY SWITCHED between the 2 STABLE STATES

SET STATE:	$S_{t1} = 1, R_{t1} = 0 \Rightarrow Q_{t1} = 1, \overline{Q}_{t1} = 0$
RESET STATE:	$S_{t1} = 0, R_{t1} = 1 \Rightarrow Q_{t1} = 0, \overline{Q}_{t1} = 1$
HOLD:	$S_{t1} = 0, R_{t1} = 0 \Rightarrow Q_{t1} = Q_{t0}, \overline{Q}_{t1} = \overline{Q}_{t0}$

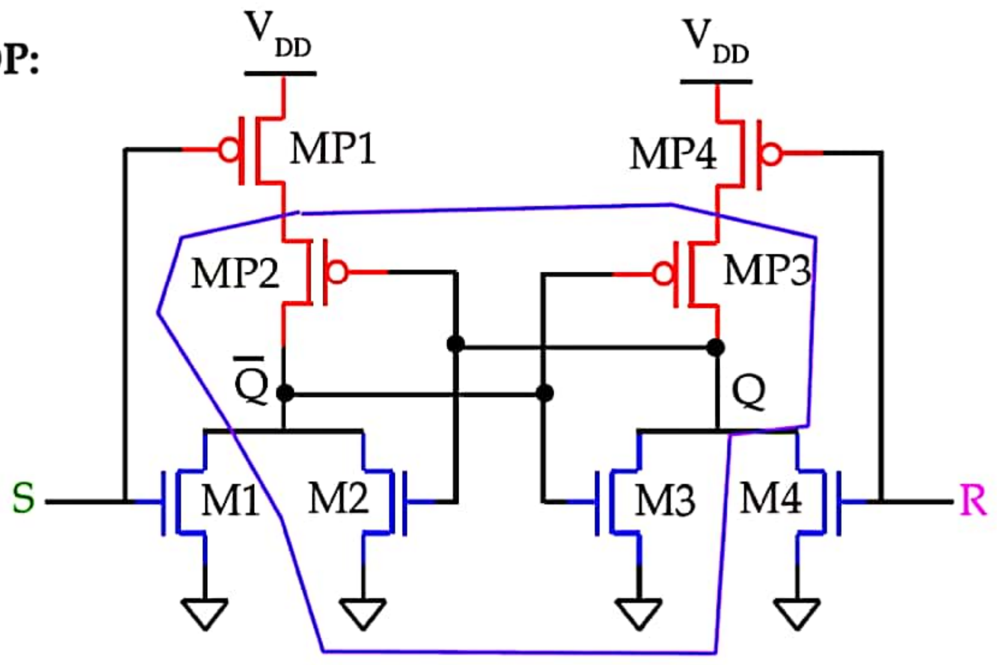
$t - t1 > t - t0$

(two cross-coupled Inverters)
(M2, MP2 and M3, MP3)

NOT ALLOWED: $S = 1, R = 1 \longrightarrow$ state Q_{n+1}, Q_{n+1} is <u>indeterminate</u>

Unclocked CMOS NOR Based SR Latch Operation

SET OP:



Let at t = t0: $Q_{t0} = 0$, $\overline{Q}_{t0} = 1$

At t = t1 > t0

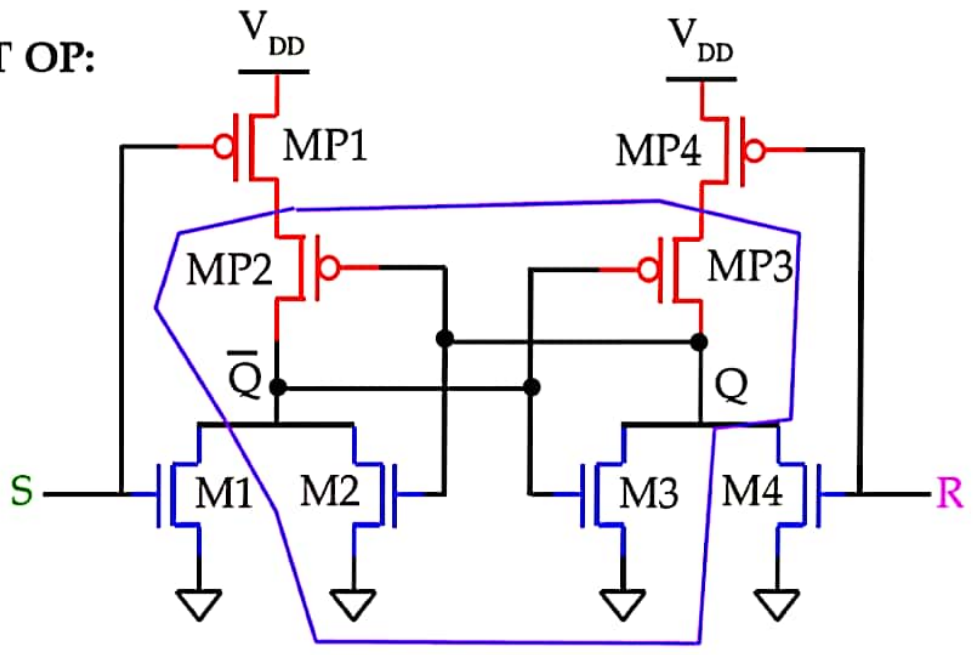
1. $S_{t1} = V_{DD}$ => M1 ON, MP1 OFF => $\overline{Q}_{t1} = 0$

2. $R_{t1} = 0$ and $\overline{Q}_{t1} = 0$ => M4 OFF, M3 OFF, MP3 ON, MP4 ON => $Q_{t1} = V_{DD}$

3. $Q_{t1} = V_{DD}$ => M2 ON, MP2 OFF => $\overline{Q}_{t1} = 0$

Unclocked CMOS NOR Based SR Latch Operation - cont.

RESET OP:



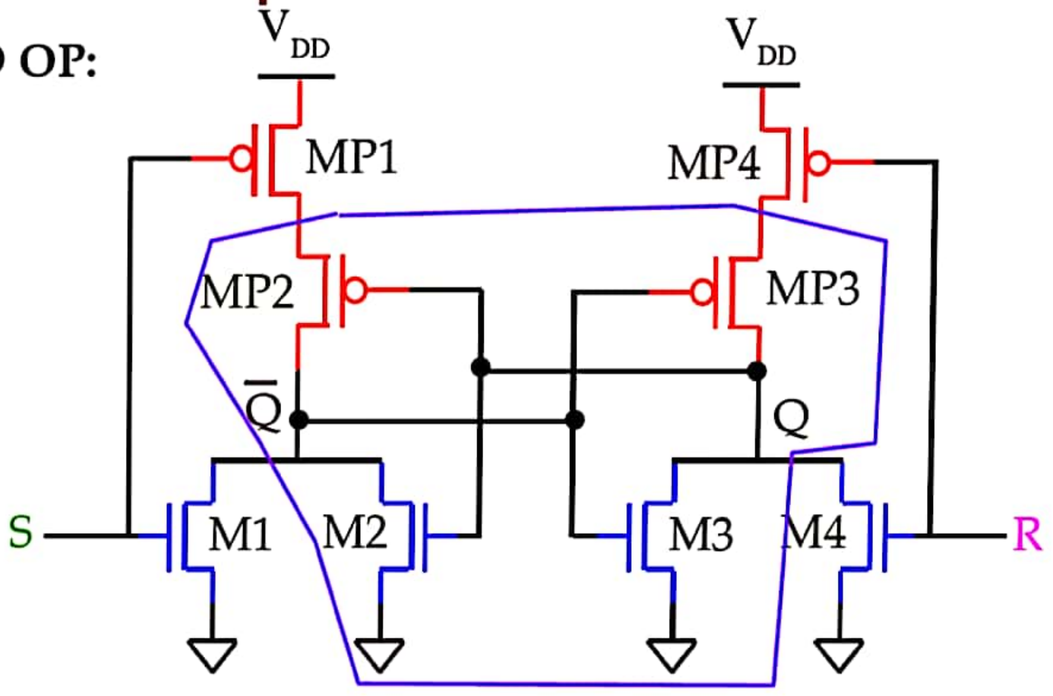
Let at $t = t0$: $Q_{t0} = 1$, $\overline{Q}_{t0} = 0$

At $t = t1 > t0$

1. $R_{t1} = 1$ => M4 ON, MP4 OFF => $Q_{t1} = 0$
2. $S_{t1} = 0$ and $Q_{t1} = 0$ => M1 OFF, M2 OFF, MP1 ON, MP2 ON => $\overline{Q}_{t1} = V_{DD}$
3. $\overline{Q}_{t1} = V_{DD}$ => M3 ON, MP3 OFF => $Q_{t1} = 0$

Unclocked CMOS NOR Based SR Latch Operation - cont.

HOLD OP:



At t = t1 > t0

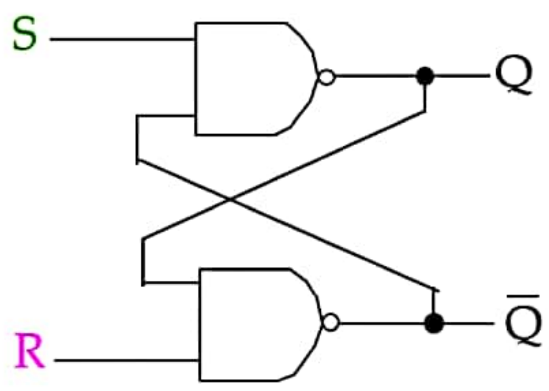
1. $S_{t1} = 0$ => M1 OFF, MP1 ON; $R_{t1} = 0$ => M4 OFF, MP4 ON

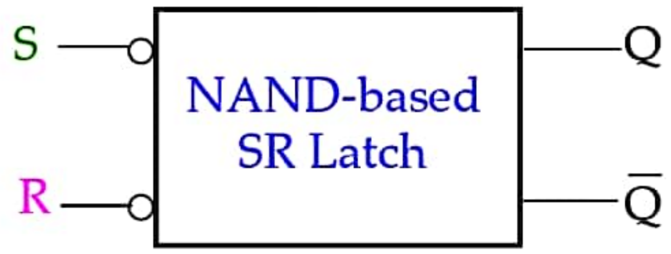
2a. $Q_{t1} = Q_{t0} = V_{DD}$, $\overline{Q}_{t1} = \overline{Q}_{t0} = 0$ => M2 ON, MP2 OFF, M3 OFF, MP3 ON

or

2b. $Q_{t1} = Q_{t0} = 0$, $\overline{Q}_{t1} = \overline{Q}_{t0} = V_{DD}$ => M2 OFF, MP2 ON, M3 ON, MP3 OFF

Unclocked CMOS NAND Based SR Latch Circuit - cont





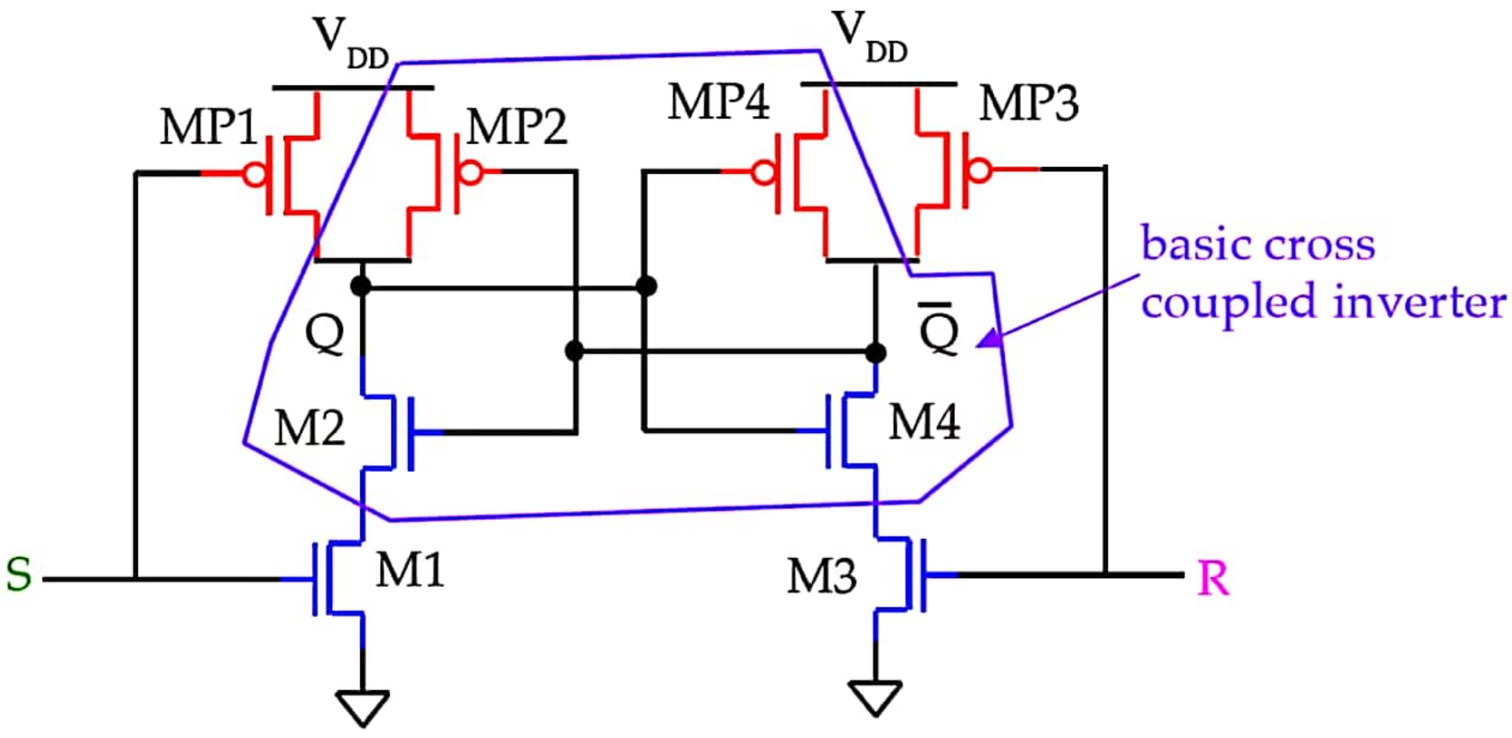
t = t1 > t = t0

S_{t1}	R_{t1}	Q_{t1}	$\overline{Q}_{t1}$	Operation
0	0	0	0	NOT allowed
0	1	1	0	set
1	0	0	1	reset
1	1	Q_{t0}	$\overline{Q}_{t0}$	hold

– –

SR- Latches
+ Simplest form of latch
- Asynchronous
- Not Allowed Input Sequence

ASYNCHRONOUS NAND BASED SR LATCH



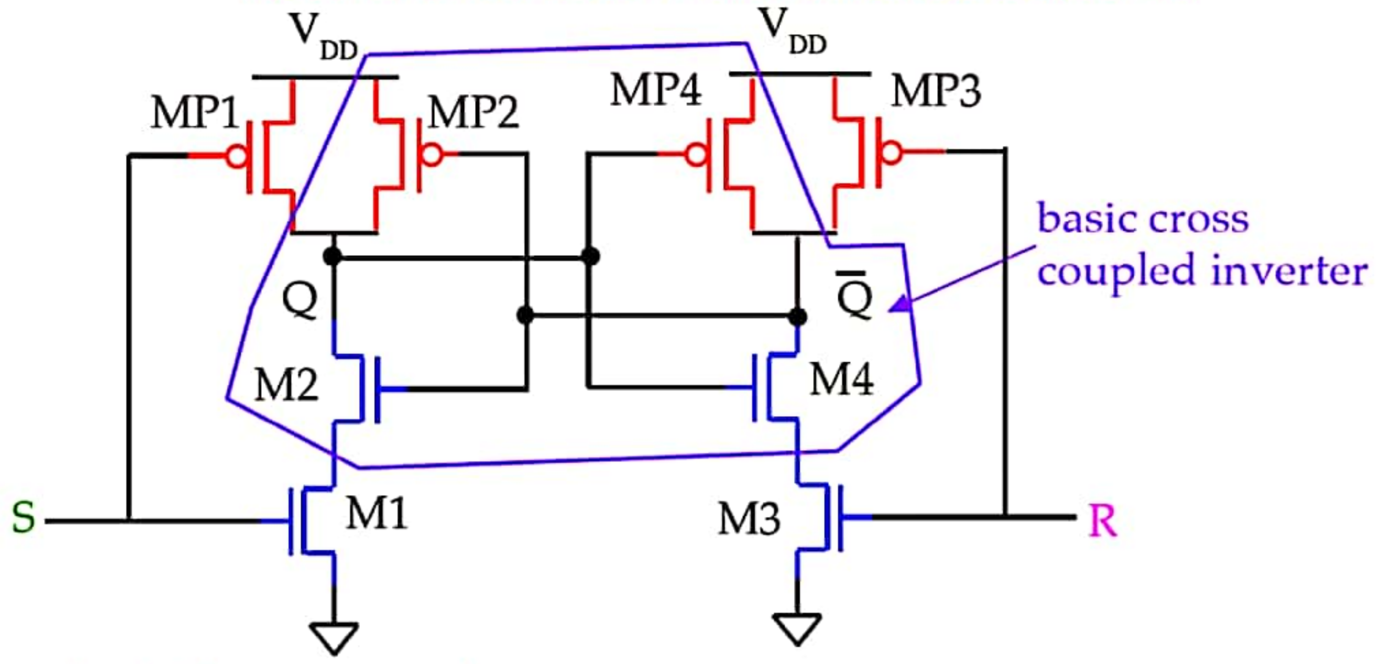
t = t1 > t = t0

S_{t1}	R_{t1}	Q_{t1}	$\overline{Q}_{t1}$	Operation
0	0	0	0	NOT allowed
0	1	1	0	set
1	0	0	1	reset
1	1	Q_{t0}	$\overline{Q}_{t0}$	hold

ACTIVE LOW

Unclocked CMOS NAND Based SR Latch Circuit

ASYNCHRONOUS NAND BASED SR LATCH



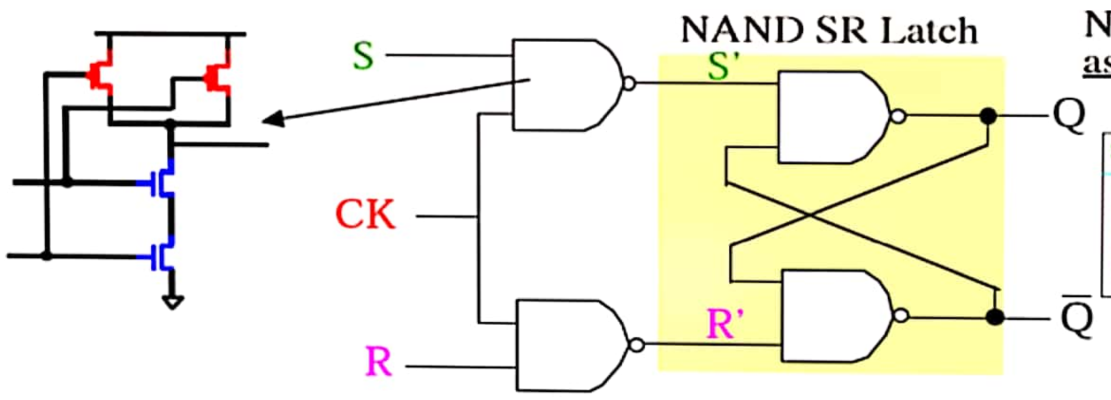
$t = t1 > t = t0$

S_{t1}	R_{t1}	Q_{t1}	$\overline{Q}_{t1}$	Operation
0	0	0	0	NOT allowed
0	1	1	0	set
1	0	0	1	reset
1	1	Q_{t0}	$\overline{Q}_{t0}$	hold

ACTIVE LOW

Clocked CMOS Latch Circuits

CLOCKED SR LATCH: Synchronization is introduced through clock CK.



NOTE: S and R are <u>asynchronous</u>.

S'_{t1}	R'_{t1}	Q_{t1}	$\overline{Q}_{t1}$	Operation
0	0	0	0	NOT allowed
0	1	1	0	set
1	0	0	1	reset
1	1	Q_{t0}	$\overline{Q}_{t0}$	hold

When CK = 0, S' = R' =1 independent of the values of S and R => HOLD

HOLD STATE: CK = 0, S = x, R = x => $Q_{n+1} = Q_n$, $\overline{Q}_{n+1} = \overline{Q}_n$

SET STATE: CK = 1, S = 1, R = 0 => $Q_{n+1} = 1$, $\overline{Q}_{n+1} = 0$

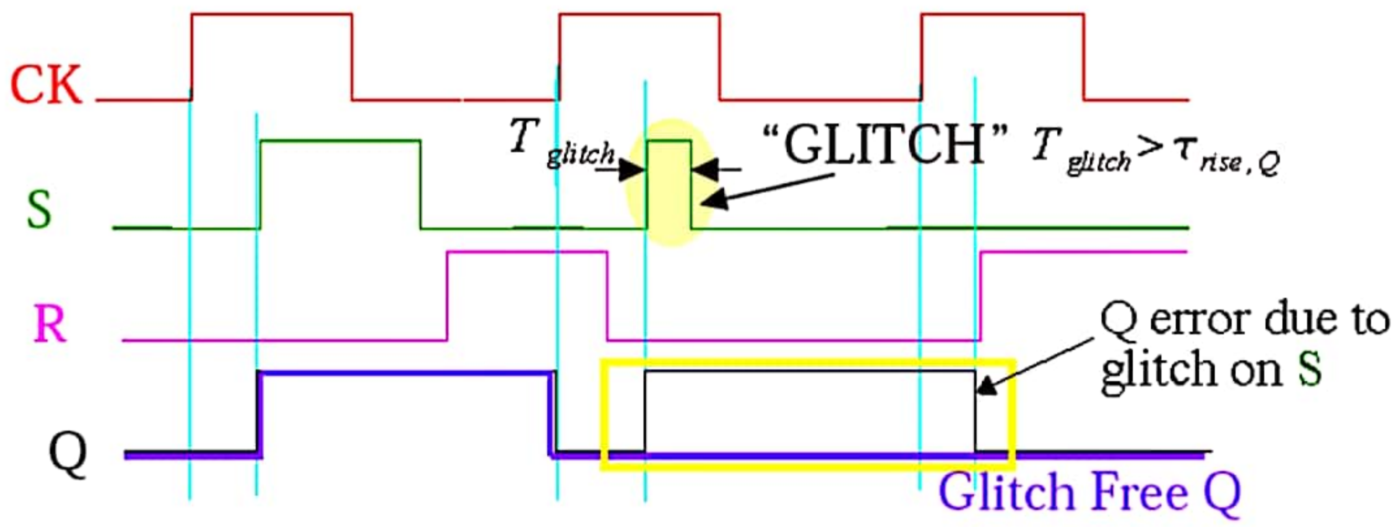
RESET STATE: CK = 1, S = 0, R = 1 => $Q_{n+1} = 0$, $\overline{Q}_{n+1} = 1$

NOT ALLOWED: CK = 1, S = 1, R = 1 => S' = 0, R' = 0

"ACTIVE HIGH"

Clocked CMOS Latch Circuits - cont.

HOLD STATE:	$CK = 0, S = x, R = x \Rightarrow$	$Q_{n+1} = Q_n$,	$\overline{Q}_{n+1} = \overline{Q}_n$
SET STATE:	$CK = 1, S = 1, R = 0 \Rightarrow$	$Q_{n+1} = 1$,	$\overline{Q}_{n+1} = 0$
RESET STATE:	$CK = 1, S = 0, R = 1 \Rightarrow$	$Q_{n+1} = 0$,	$\overline{Q}_{n+1} = 1$
NOT ALLOWED:	$CK = 1, S = 1, R = 1$		

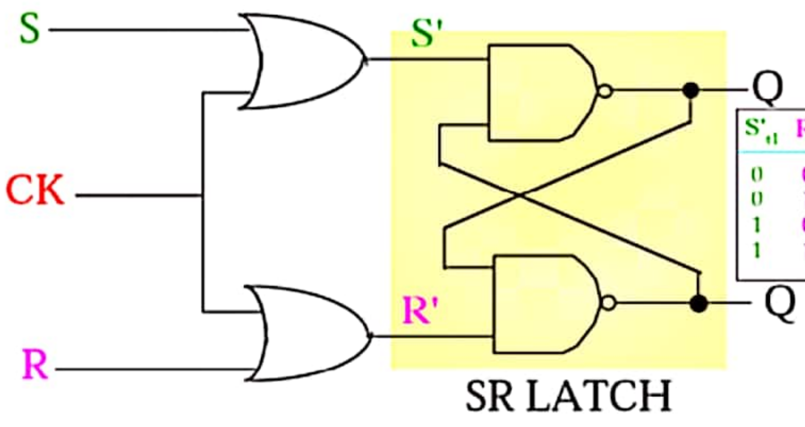

CK
S
R
Q
T_{glitch}
"GLITCH" $T_{glitch} > \tau_{rise,Q}$
Q error due to glitch on S
Glitch Free Q

WHEN "GLITCH" ON S (OR R) OCCURS DURING CK = 1, Q IS SET (OR RESET)

LEVEL SENSITIVE: WHEN CK = 1, ANY CHANGES IN S, R WILL EFFECT Q.

Clocked CMOS Latch Circuits - cont.

Another Gate Level schematic of a Clocked NAND Based SR Latch



S'_{t1}	R'_{t1}	Q_{t1}	$\overline{Q}_{t1}$	Operation
0	0	0	0	NOT allowed
0	1	1	0	set
1	0	0	1	reset
1	1	Q_{t0}	$\overline{Q}_{t0}$	hold

When CK = 1, S' = R' =1 independent of the values of S and R => HOLD

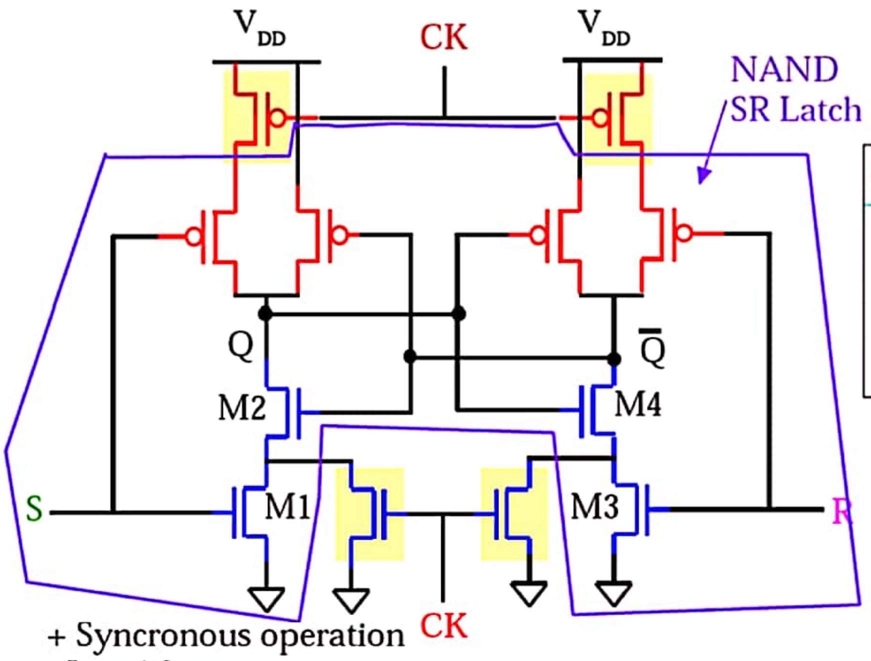
"ACTIVE LOW"

CK	S	R	Q_{n+1}	$\overline{Q}_{n+1}$	Operation
0	0	0	0	0	NOT allowed
0	0	1	1	0	set
0	1	0	0	1	reset
1	x	x	Q_n	$\overline{Q}_n$	hold

S' = R' = 0

Clocked CMOS Latch Circuits - cont.

CMOS Clocked NAND Based SR Latch or Flip-Flop



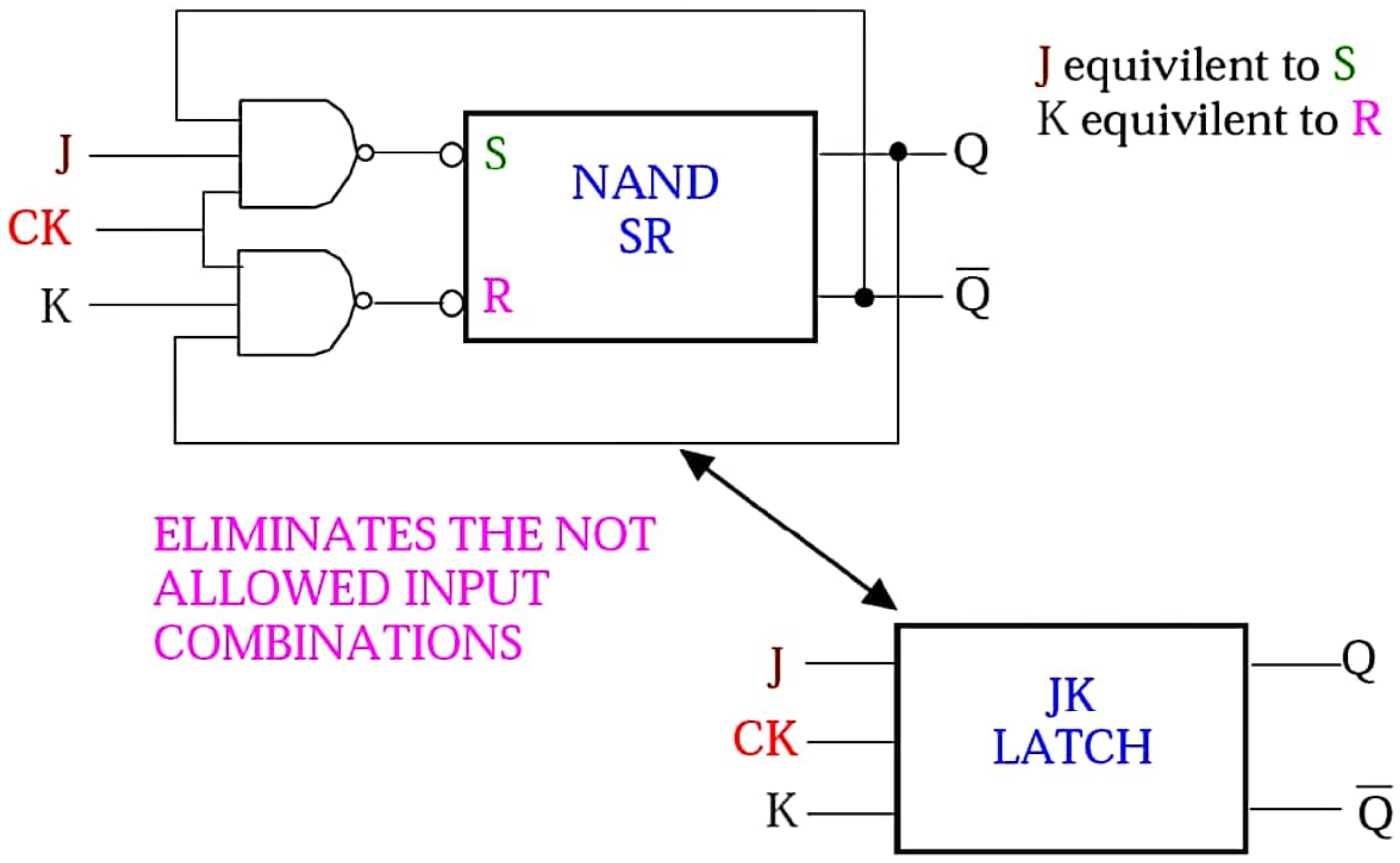
CK	S	R	Q_{n+1}	$\overline{Q}_{n+1}$	Operation
0	0	0	0	0	NOT allowed
0	0	1	1	0	set
0	1	0	0	1	reset
1	x	x	Q_n	$\overline{Q}_n$	hold

"ACTIVE LOW"

+ Syncronous operation
- Level Sensitive
- Not Allowed Input Sequence

CMOS Clocked Latch Circuits - cont.

NAND BASED CLOCKED JK FLIP-FLOP



J equivilent to S
K equivilent to R

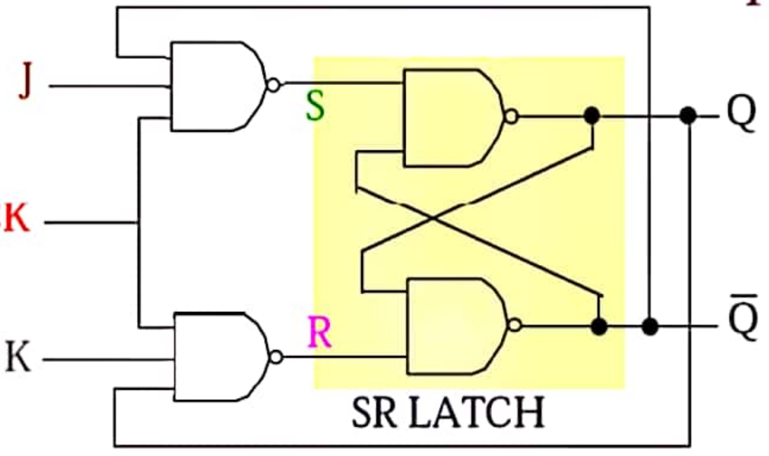
ELIMINATES THE NOT ALLOWED INPUT COMBINATIONS

Clocked NAND Based JK Latch Operation

CK = 0 => hold

CK = 1=> active

"ACTIVE HIGH"



CK = 1

J	K	Q_n	$\overline{Q}_n$	S	R	Q_{n+1}	$\overline{Q}_{n+1}$	Operation
0	0	0	1	1	1	0	1	hold
0	0	1	0	1	1	1	0	hold
0	1	0	1	1	1	0	1	reset (hold)
0	1	1	0	1	0	0	1	reset
1	0	0	1	0	1	1	0	set
1	0	1	0	1	1	1	0	set (hold)
1	1	0	1	0	1	1	0	toggle
1	1	1	0	1	0	0	1	toggle

The not-allowed S, R values S = R = 0 do not occur for any values of J, K, CK.

OSC $\rightarrow$ not desirable, but the state Q_{n+1}, Q_{n+1} is determinate

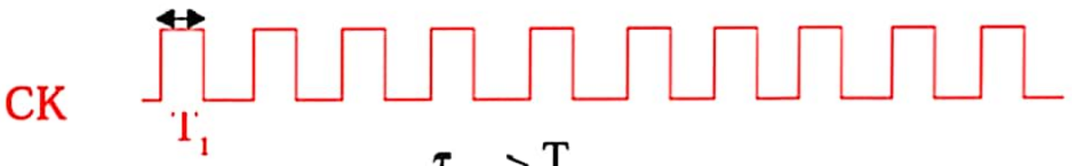
Clocked NAND Based JK Latch Operation

CK = 1

J	K	Q_n	$\overline{Q}_n$	S	R	Q_{n+1}	$\overline{Q}_{n+1}$	Operation
0	0	0	1	1	1	0	1	hold
0	0	1	0	1	1	1	0	hold
0	1	0	1	1	1	0	1	reset
0	1	1	0	1	0	0	1	reset
1	0	0	1	0	1	1	0	set
1	0	1	0	1	1	1	0	set
1	1	0	1	0	1	1	0	toggle
1	1	1	0	1	0	0	1	toggle

OSC

TO PREVENT OSICLLATION WHEN J = K = 1:

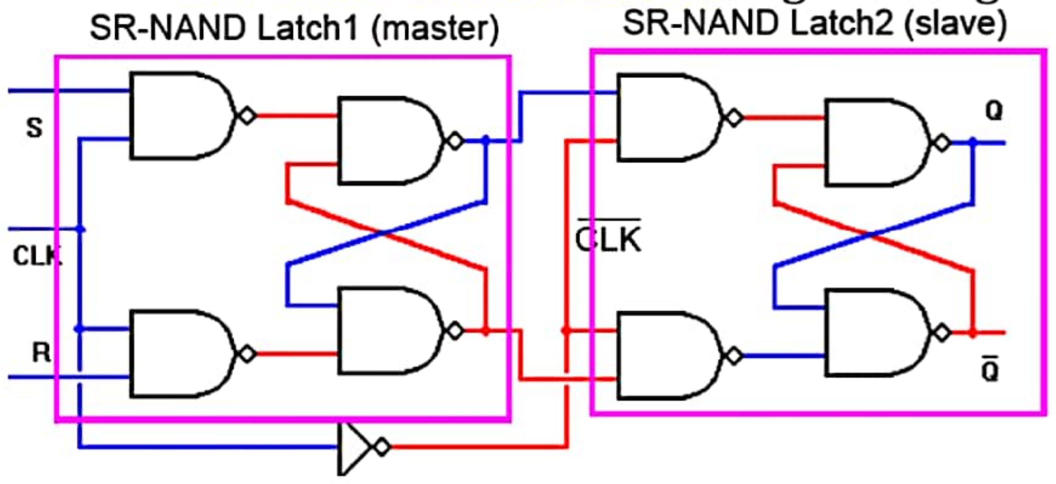


$\tau_{JKP} > T_1$

(high speed clock may be impractical)

τ_{JKP} = INPUT-OUTPUT PROP DELAY OF JK LATCH
(CK 1 -> 0 BEFORE Q, $\overline{Q}$ CAN SWITCH 2nd TIME)

CLOCKED SR FLIP FLOP: Negative Edge Triggered.



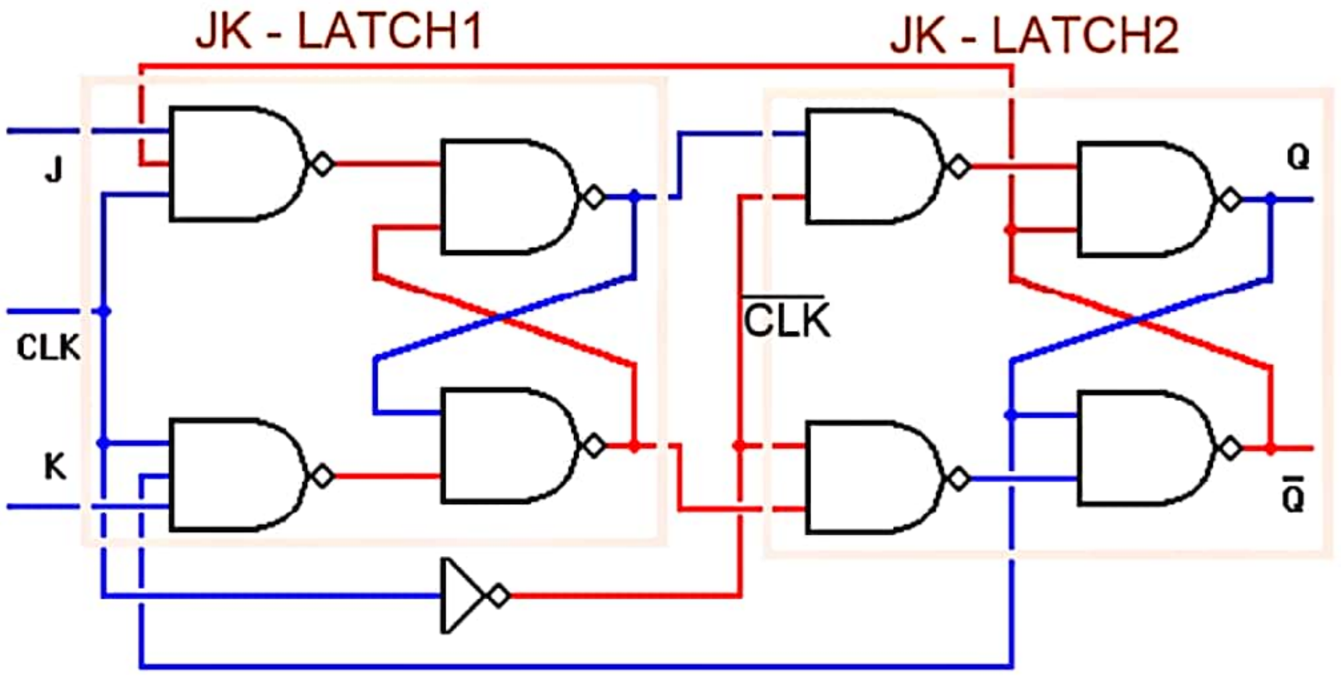
+ Syncronous operation
+ Not Level Sensitive
- Not Allowed S, R Sequence

- Start with CLK = 0 , the S , R inputs are disconnected from the input Latch1.
- Changes in S, R cannot affect the state of Q, $\overline{Q}$.

When CLK = 1, S, R are able to control the state of Latch1.
- Inverted $\overline{CLK}$ applied to Latch2 prevents the state of Latch1 from effecting Q, $\overline{Q}$.
- Any changes to R, S are tracked by Latch1 while CLK = 1, but not reflected at Q, $\overline{Q}$.

When CLK = 0, S, R are again isolated from Latch1.
- Inverted $\overline{CLK}$ allows the current state of Latch1 to reach Latch2.
- Q, $\overline{Q}$ can only change state when the CLK signal falls from 1 to 0.
- This is the falling (negative) edge of the CLK signal.

CLOCKED EDGE TRIGGERED JK FLIP-FLOP

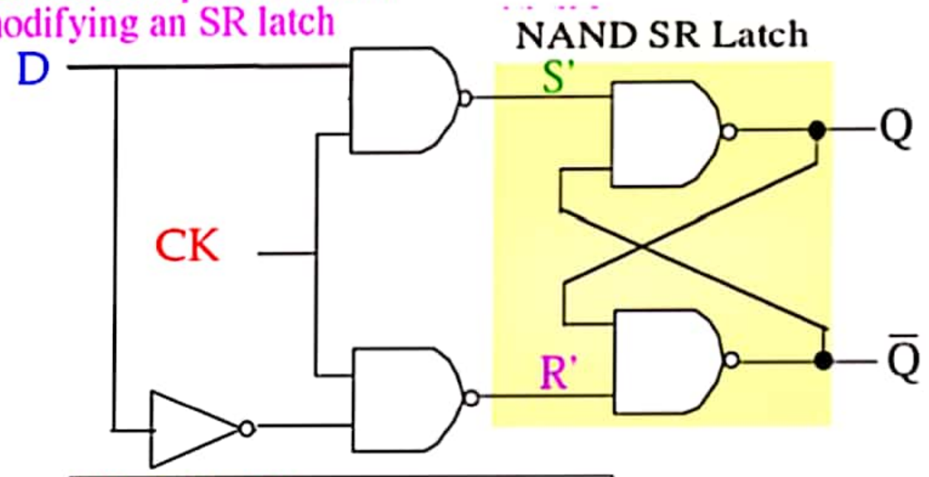


+ Synchronous Operation
+ No Not-Allowed Inputs
+ Not Level Sensitive
+ No Q, $\overline{Q}$ Oscillation when J = K = 1

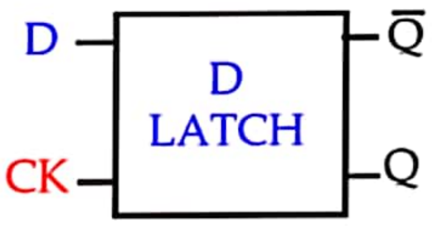
Since the behavior of the JK flip-flop is completely predictable under all conditions, it is the preferred type of flip-flop for most logic circuit designs.

CMOS D-Latch

gate level implementation modifying an SR latch

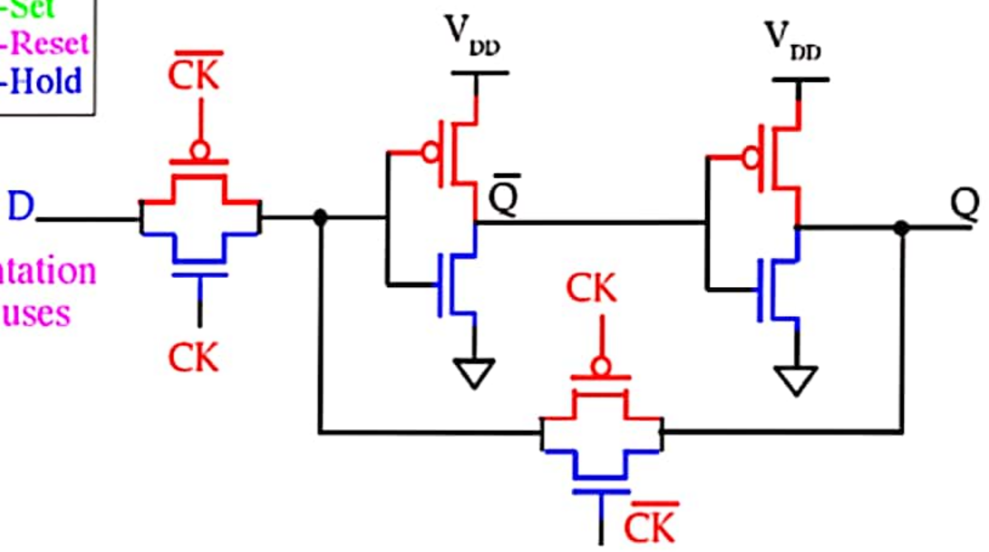


If CK = 1: $Q_{n+1} = D$
If CK = 0: $Q_{n+1} = Q_n$

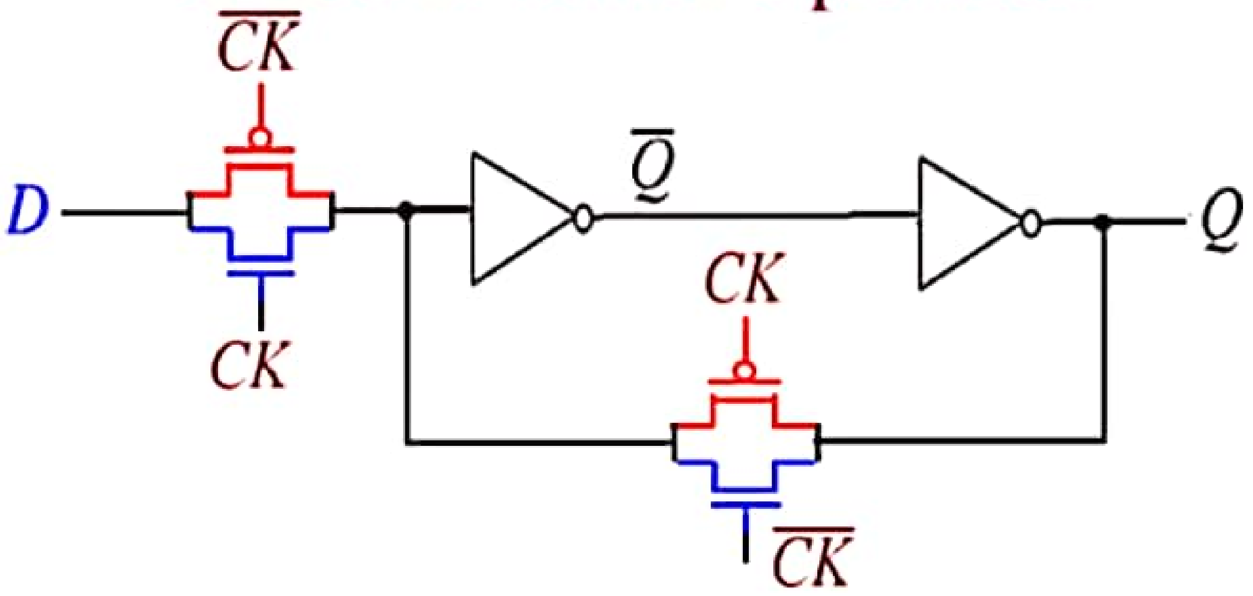


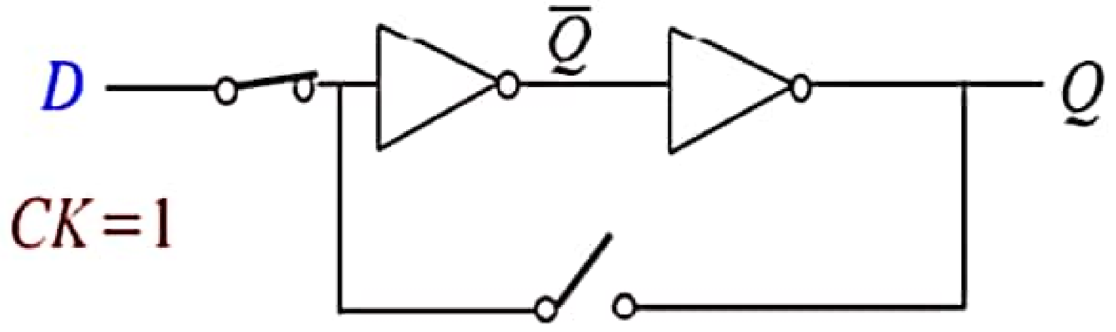
CK	D	S'	R'	Q_{n+1}	$\overline{Q}_{n+1}$	
1	1	0	1	1	0	SR-Set
1	0	1	0	0	1	SR-Reset
0	x	0	0	Q_n	$\overline{Q}_n$	SR-Hold

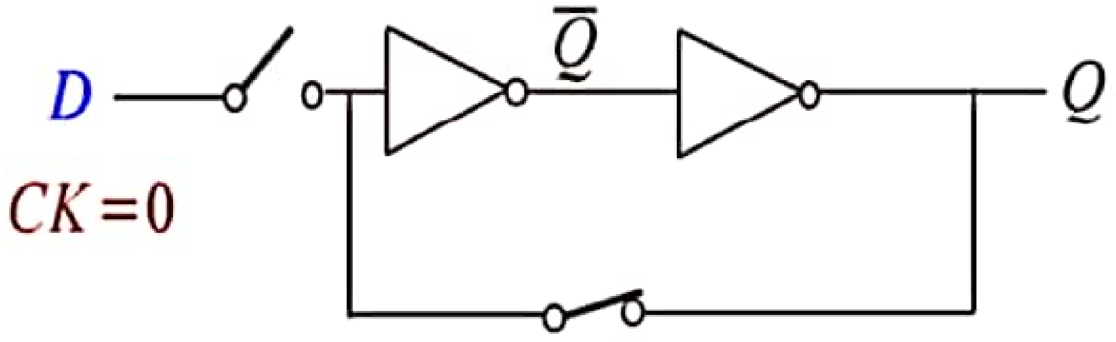
transistor level implementation using transmission gates uses fewer transistors



CMOS D-Latch Operation

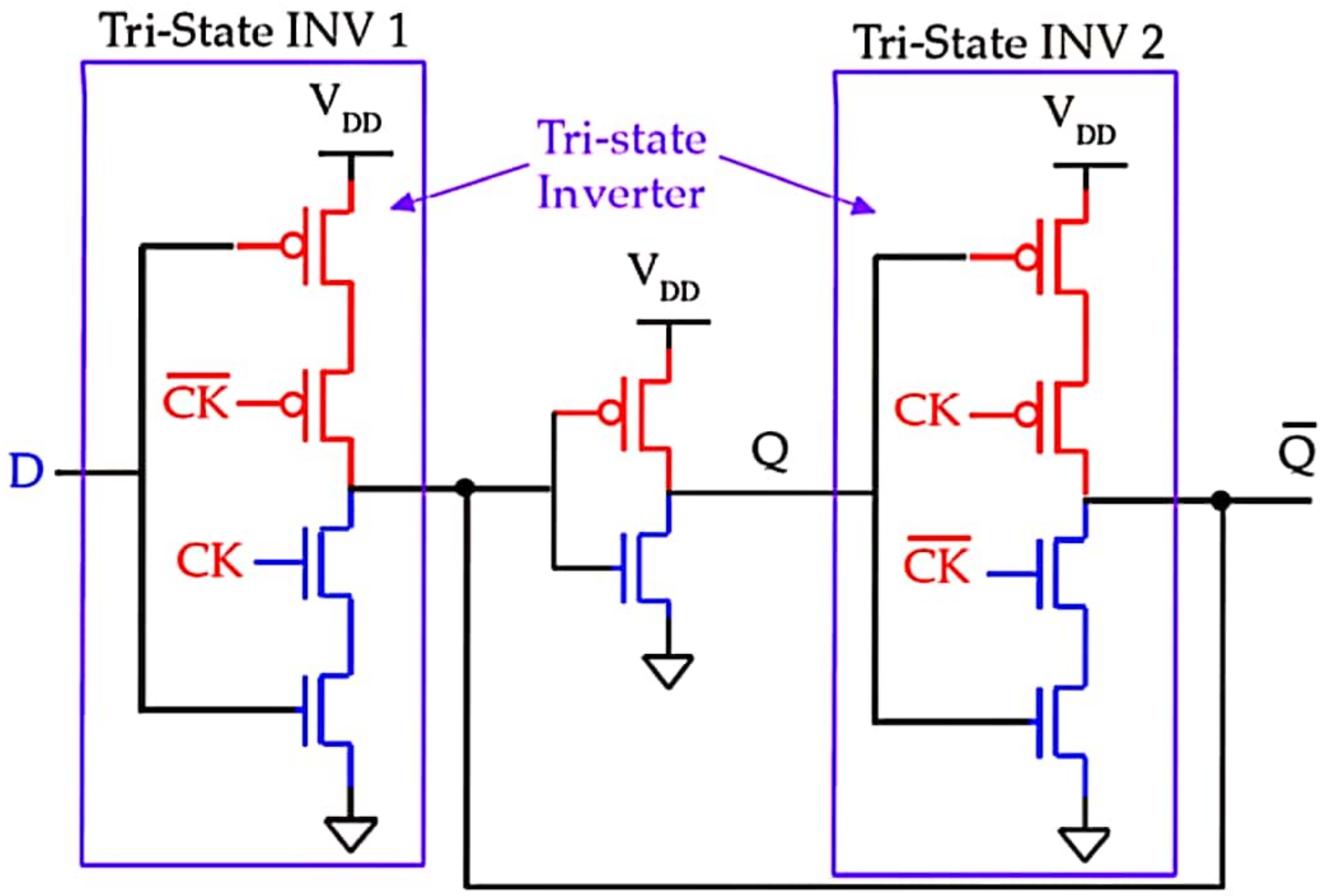






+ Much simpler then JK Latch.
+ Does not require Edge Triggering for Safe Operation.

CMOS D-Latch - cont.



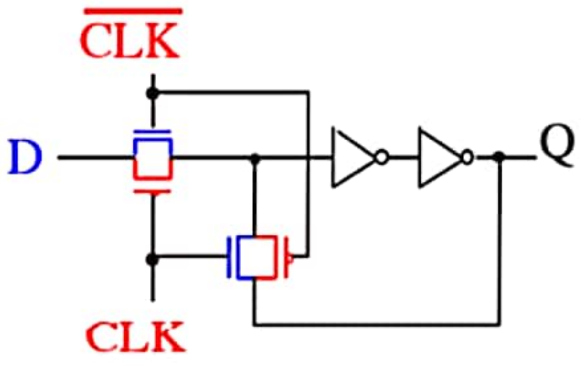
alternative implementation using clocked tri-state inverters

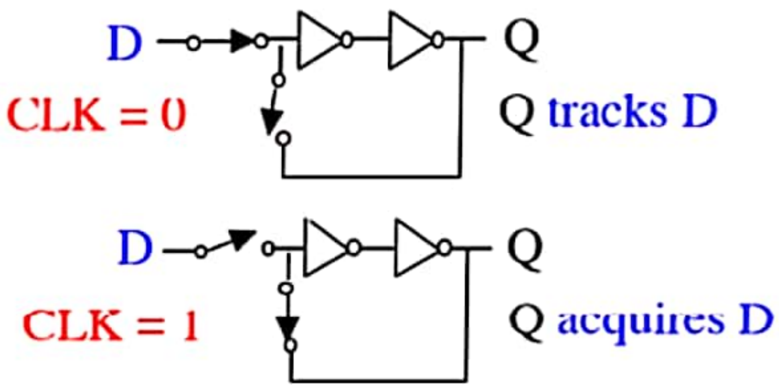
CK = 1: Tri-state INV 1 is active, Tri-state INV 2 is Hi-Z and $Q_n = D_n$

CK = 0: Tri-state INV 1 is Hi-Z, Tri-state INV 2 is active and Q_n and $\overline{Q}_n$ are held

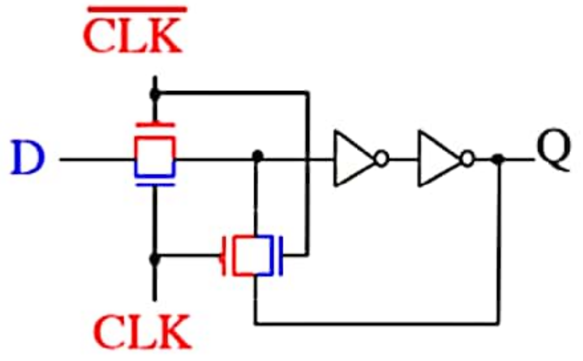
CMOS D Flip-Flop

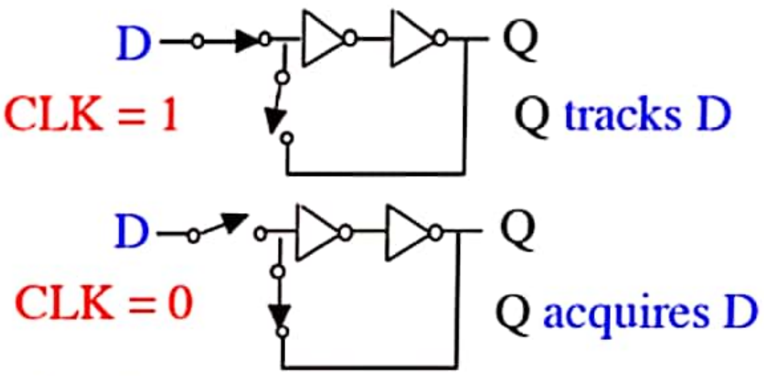
Positive D - Latch





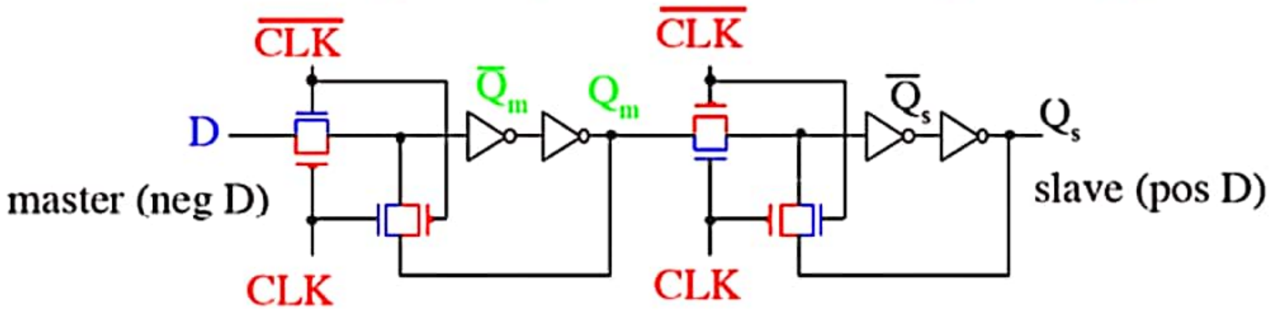
Negative D - Latch



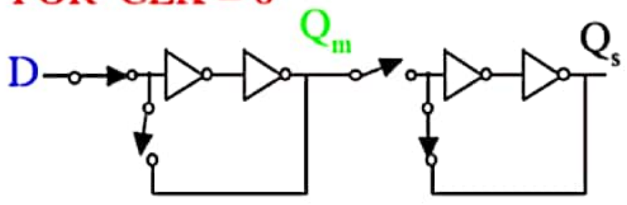


D Flip-Flop = Positive D-Latch + Negative D-Latch

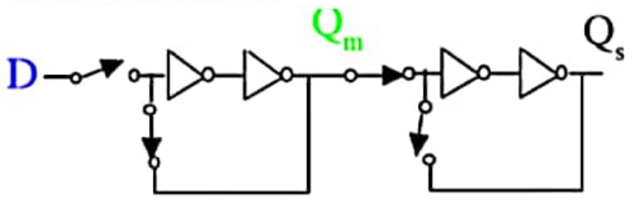
CMOS D Flip-Flop – Positive Edge Triggered



FOR CLK = 0



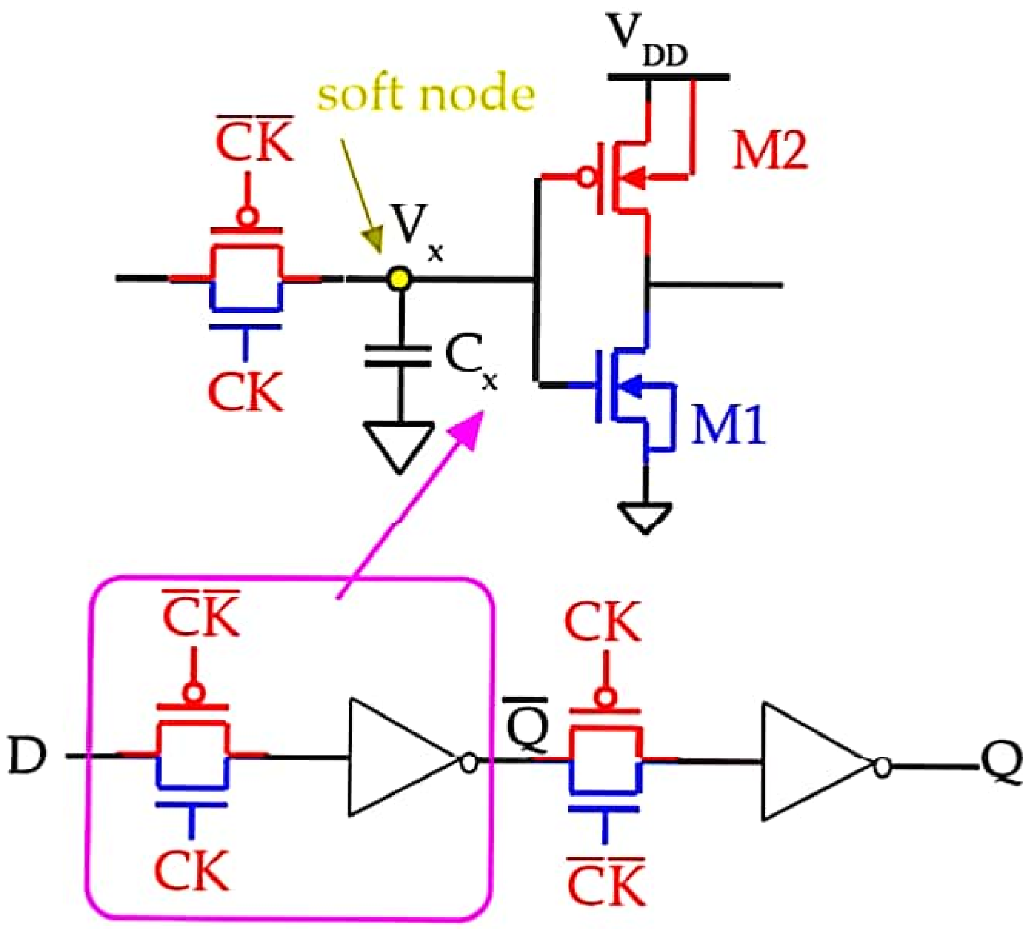
FOR CLK = 1



1. CLK = 0: master Q_m tracks input D; slave Q_s = previous D_{n-1} sample (Q_s is transparent to variations in D).
2. CLK = 0 -> 1: master stores $Q_m = D_n$ (new D sample).
3. CLK = 1: master passes $Q_m = D_n$ to slave output Q_s (Q_m and Q_s are transparent to variations in D).
4. CLK = 1 -> 0: slave locks in new D_n.
5. CLK = 0: master Q_m begins tracking D. (Q_s is transparent to variations in D)
6. CLK = 0 -> 1: master stores $Q_m = D_{n+1}$.

Scanned with CamScanner

CMOS Dynamic D Flip-Flop



1. NO FEEDBACK REGENERATIVE FEEDBACK LOOP
2. STATES STORED ON SOFT NODES