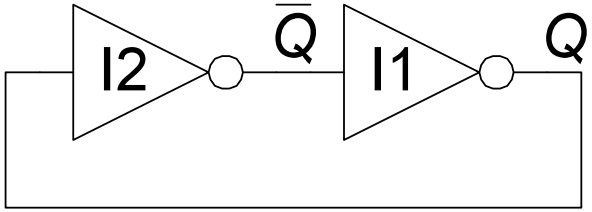
Chapter 3 :: Sequential Logic Design

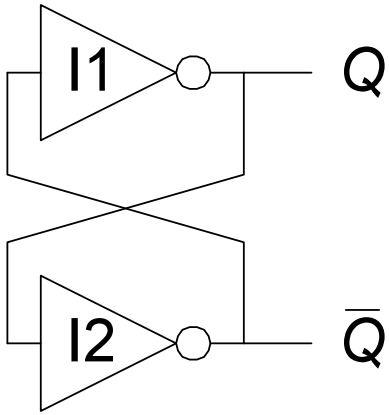
Digital Design and Computer Architecture

David Money Harris and Sarah L. Harris

Bistable Circuit

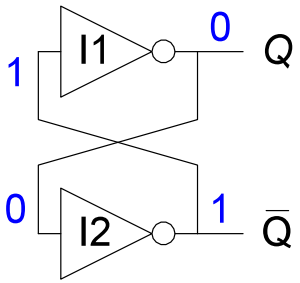
- Fundamental building block of other state elements
- Two outputs: Q, $\overline{Q}$
- No inputs



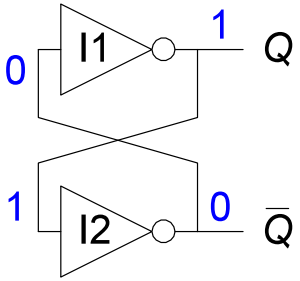


Bistable Circuit Analysis

- Consider the two possible cases:
  - $Q = 0$: then $\overline{Q} = 1$ and $Q = 0$ (consistent)



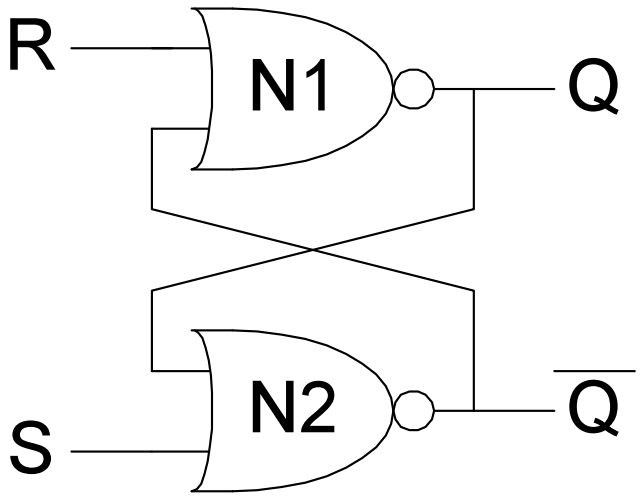
  - $Q = 1$: then $\overline{Q} = 0$ and $Q = 1$ (consistent)



- Bistable circuit stores 1 bit of state in the state variable, Q (or $\overline{Q}$)
- But there are **no inputs to control the state**

SR (Set/Reset) Latch

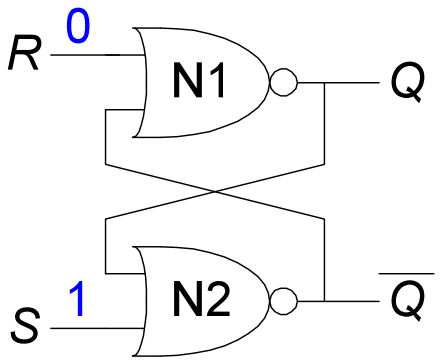
- SR Latch



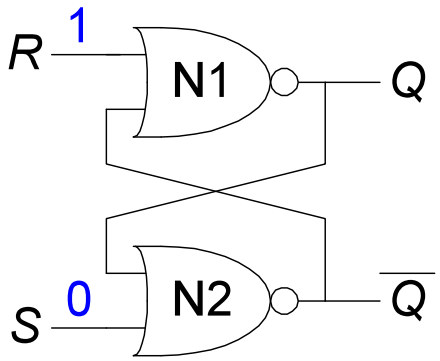
- Consider the four possible cases:
  - $S = 1, R = 0$
  - $S = 0, R = 1$
  - $S = 0, R = 0$
  - $S = 1, R = 1$

SR Latch Analysis

– $S = 1, R = 0$: then $Q = 1$ and $\overline{Q} = 0$

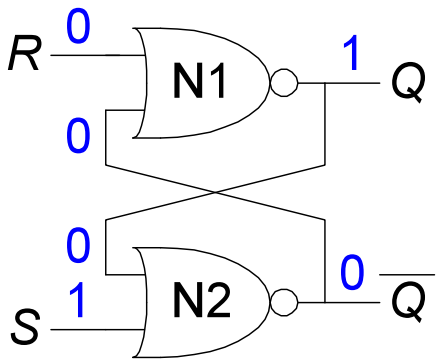


– $S = 0, R = 1$: then $Q = 0$ and $\overline{Q} = 1$

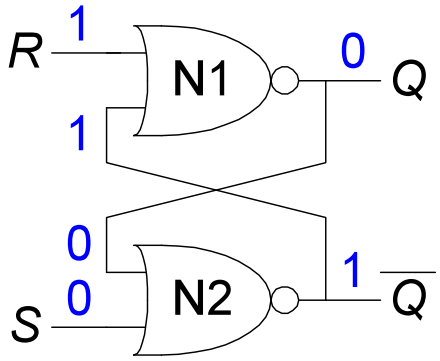


SR Latch Analysis

– $S = 1, R = 0$: then $Q = 1$ and $\overline{Q} = 0$



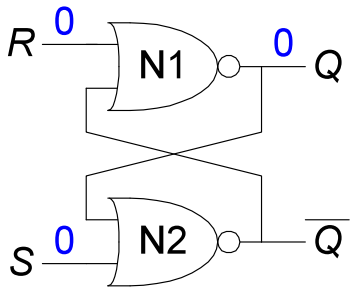
– $S = 0, R = 1$: then $Q = 0$ and $\overline{Q} = 1$



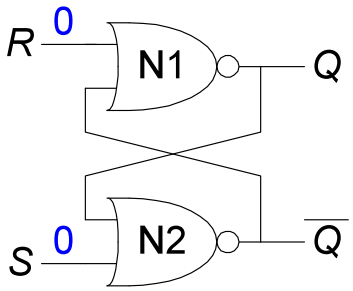
SR Latch Analysis

– $S = 0, R = 0$: then $Q = Q_{prev}$

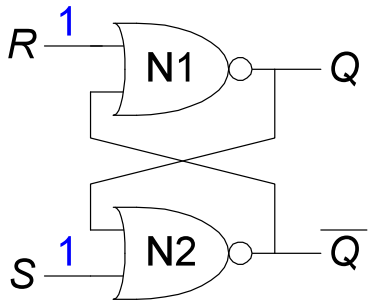
$Q_{prev} = 0$



$Q_{prev} = 1$



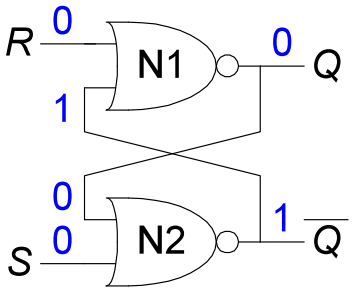
– $S = 1, R = 1$: then $Q = 0$ and $\overline{Q} = 0$



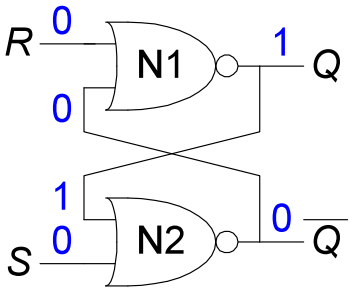
SR Latch Analysis

– $S = 0, R = 0$: then $Q = Q_{prev}$ and $\overline{Q} = \overline{Q_{prev}}$ (memory!)

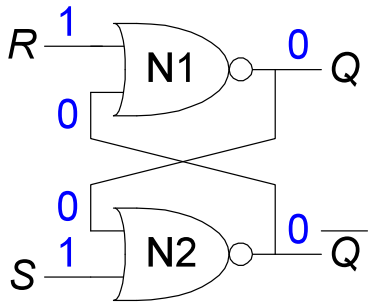
$Q_{prev} = 0$



$Q_{prev} = 1$



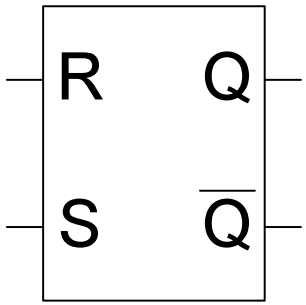
– $S = 1, R = 1$: then $Q = 0$ and $\overline{Q} = 0$ (invalid state: $\overline{Q} \neq$ NOT Q)



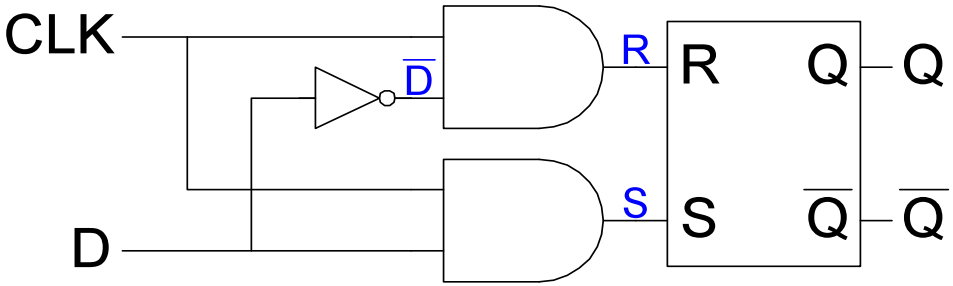
SR Latch Symbol

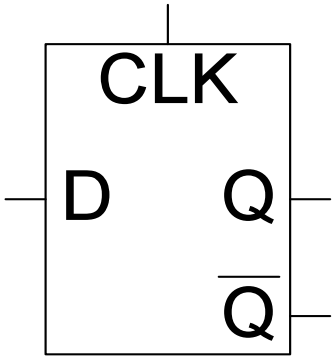
- SR stands for Set/Reset Latch
  - Stores one bit of state (Q)
- Control what value is being stored with S, R inputs
  - Set: Make the output 1 ($S = 1$, $R = 0$, $Q = \mathbf{1}$)
  - Reset: Make the output 0 ($S = 0$, $R = 1$, $Q = \mathbf{0}$)
- **Must do something to avoid invalid state (when $S = R = 1$)**

SR Latch Symbol



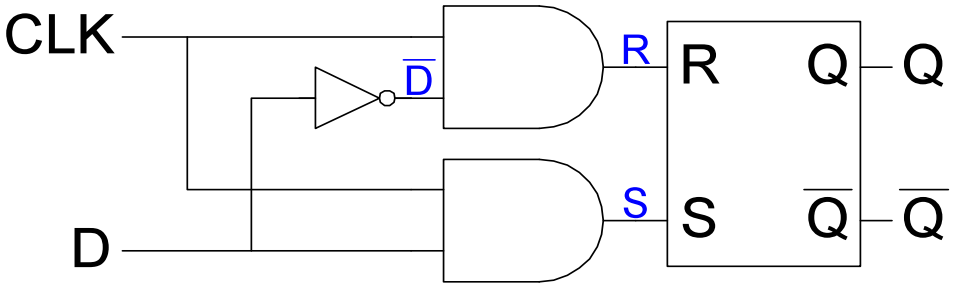
D Latch Internal Circuit

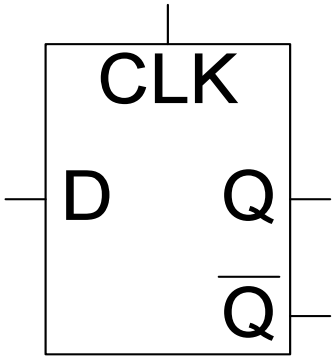




CLK	*D*	$\overline{D}$	*S*	*R*	Q	$\overline{Q}$
0	X					
1	0					
1	1					

D Latch Internal Circuit



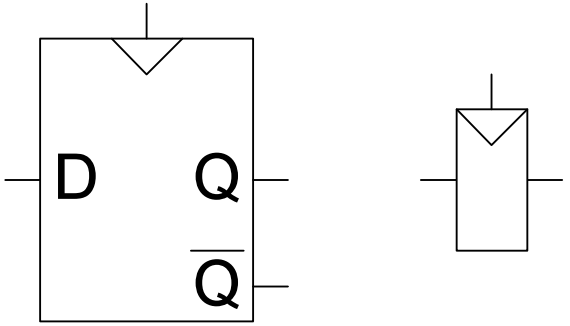


CLK	*D*	$\overline{D}$	*S*	*R*	Q	$\overline{Q}$
0	X	$\overline{X}$	0	0	Q_{prev}	$\overline{Q}_{prev}$
1	0	1	0	1	0	1
1	1	0	1	0	1	0

D Flip-Flop

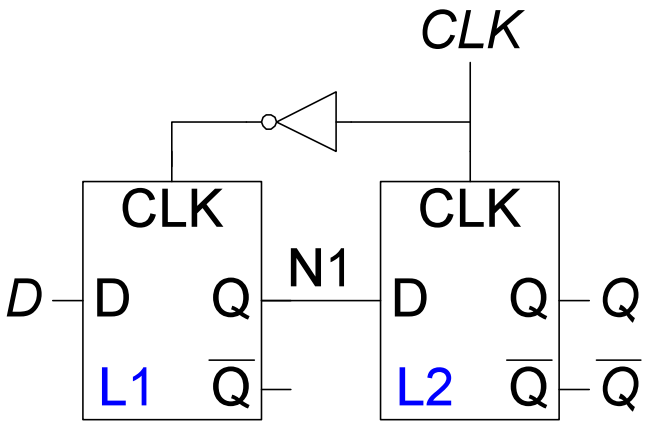
- Two inputs: *CLK*, *D*
- *Function*
  - The flip-flop "samples" *D* on the rising edge of *CLK*
    - When *CLK* rises from 0 to 1, *D* passes through to *Q*
    - Otherwise, *Q* holds its previous value
  - *Q* changes only on the rising edge of *CLK*
- A flip-flop is called an *edge-triggered* device because it is activated on the clock edge

D Flip-Flop Symbols



D Flip-Flop Internal Circuit

- Two back-to-back latches (L1 and L2) controlled by complementary clocks
- When *CLK* = 0
  - L1 is transparent
  - L2 is opaque
  - *D* passes through to N1
- When *CLK* = 1
  - L2 is transparent
  - L1 is opaque
  - N1 passes through to *Q*
- Thus, on the edge of the clock (when *CLK* rises from 0→1)
  - *D* passes through to *Q*

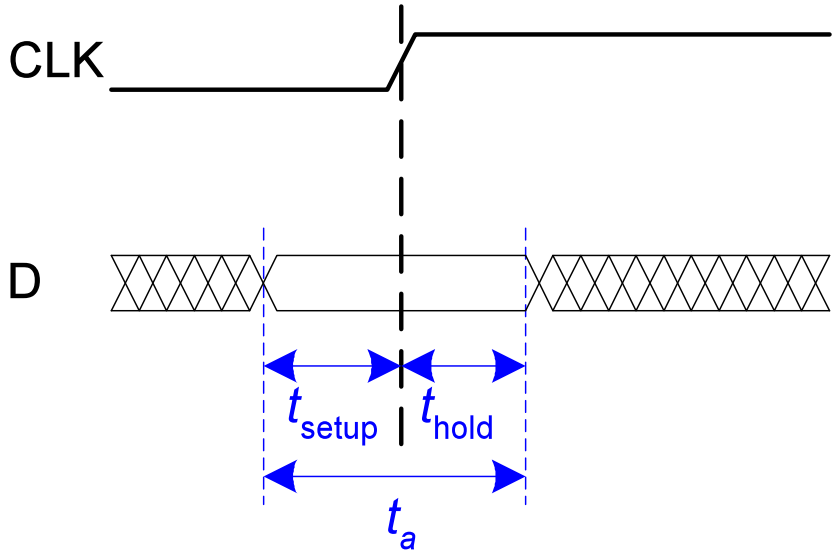


Timing

- Flip-flop samples *D* at clock edge
- *D* must be stable when it is sampled
- Similar to a photograph, *D* must be stable around the clock edge
- If *D* is changing when it is sampled, metastability can occur

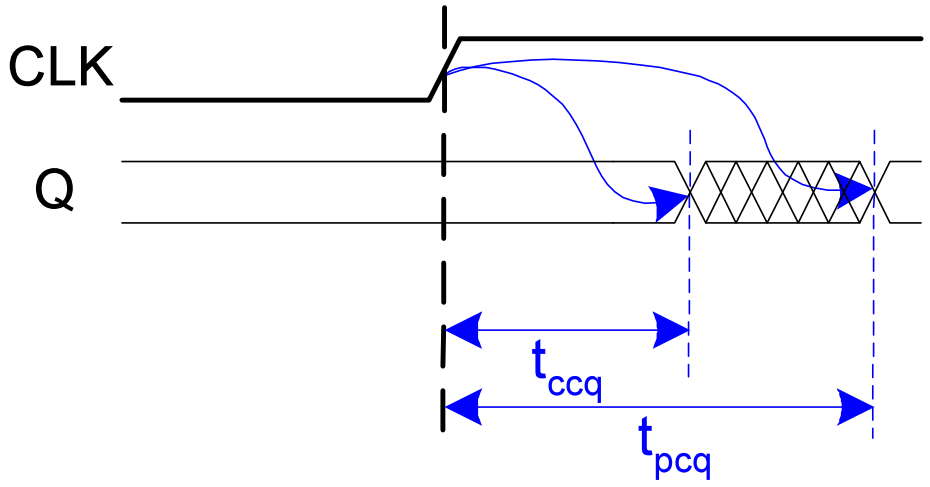
Input Timing Constraints

- Setup time: t_{setup} = time *before* the clock edge that data must be stable (i.e. not changing)
- Hold time: t_{hold} = time *after* the clock edge that data must be stable
- Aperture time: t_a = time around clock edge that data must be stable ($t_a = t_{setup} + t_{hold}$)



Output Timing Constraints

- Propagation delay: t_{pcq} = time after clock edge that the output Q is guaranteed to be stable (i.e., to stop changing)
- Contamination delay: t_{ccq} = time after clock edge that Q might be unstable (i.e., start changing)

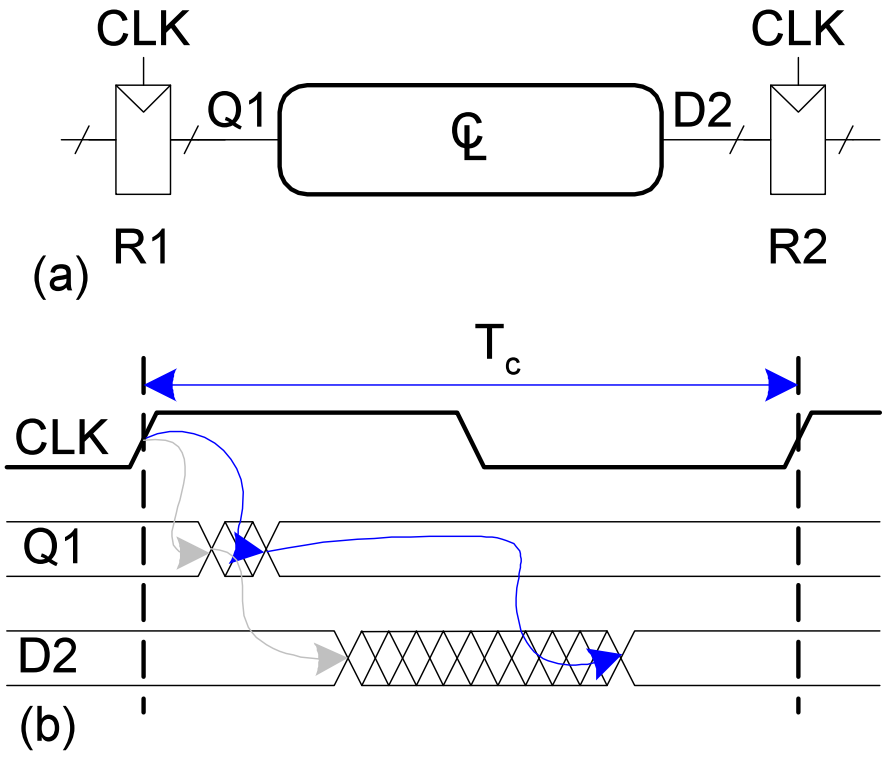


Dynamic Discipline

- The input to a synchronous sequential circuit must be stable during the aperture (setup and hold) time around the clock edge.
- Specifically, the input must be stable
  - at least t_{setup} before the clock edge
  - at least until t_{hold} after the clock edge

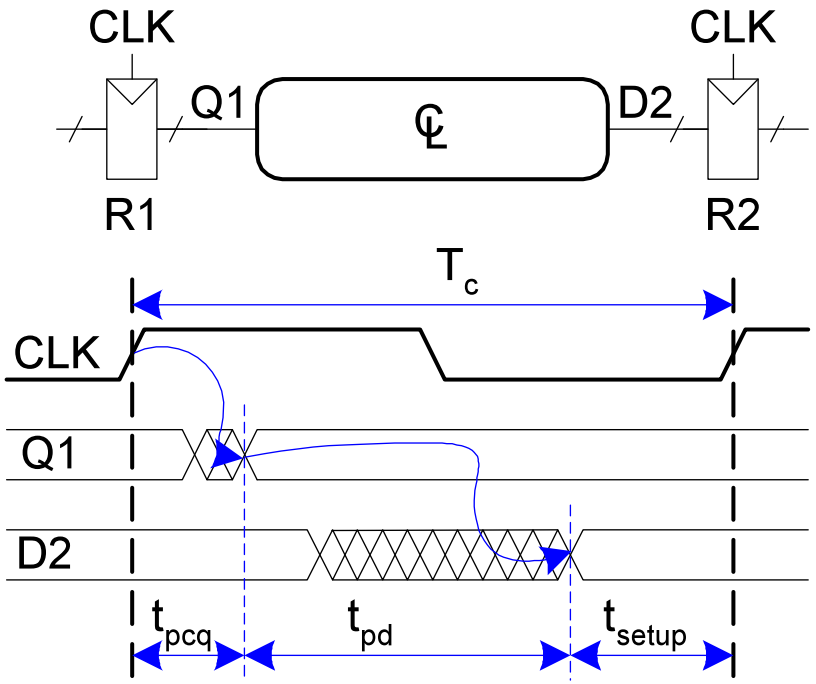
Dynamic Discipline

- The delay between registers has a **minimum** and **maximum** delay, dependent on the delays of the circuit elements



Setup Time Constraint

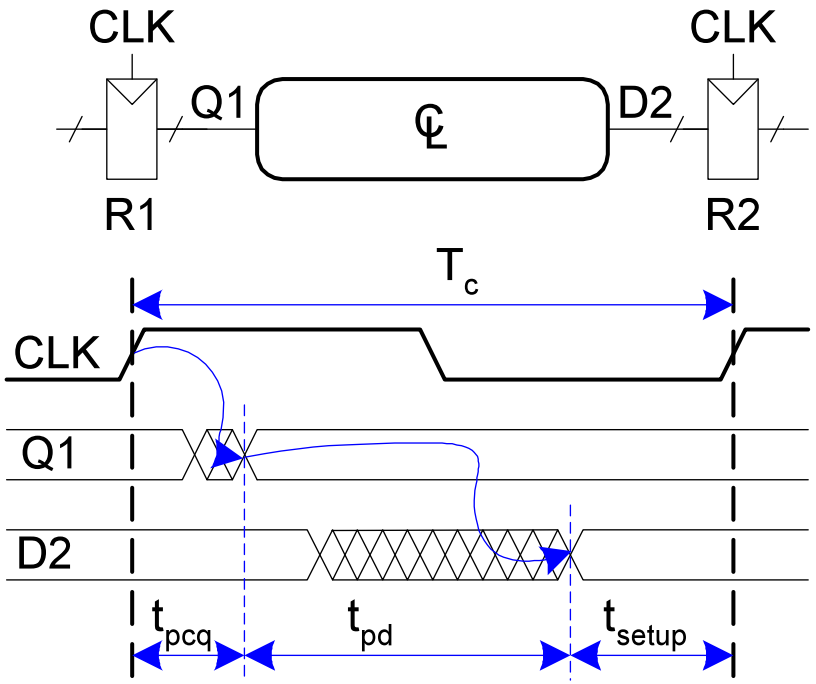
- The setup time constraint depends on the **maximum** delay from register R1 through the combinational logic.
- The input to register R2 must be stable at least t_{setup} before the clock edge.



$$T_c \geq$$

Setup Time Constraint

- The setup time constraint depends on the **maximum** delay from register R1 through the combinational logic.
- The input to register R2 must be stable at least t_{setup} before the clock edge.

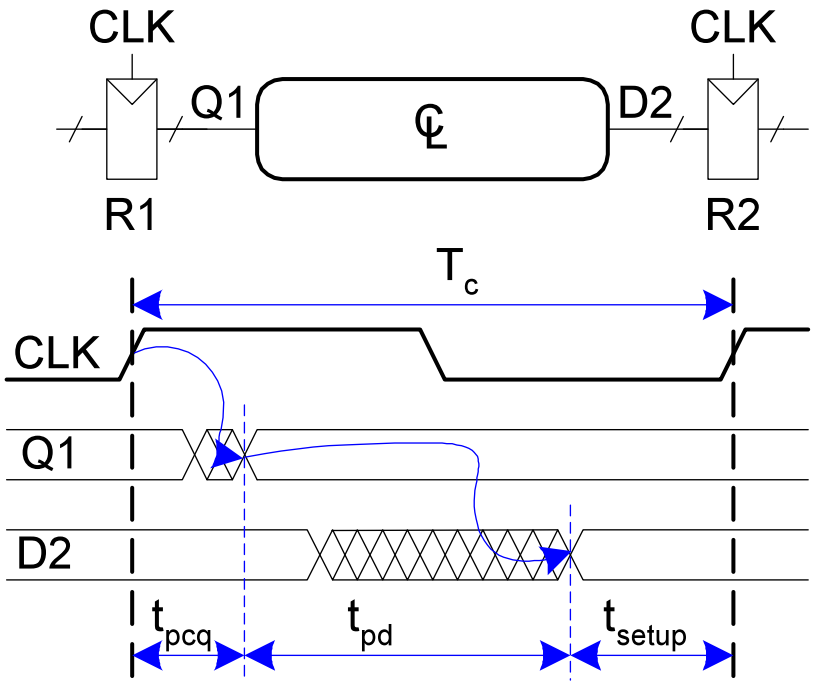


$$T_c \geq t_{pcq} + t_{pd} + t_{\text{setup}}$$

$$t_{pd} \leq$$

Setup Time Constraint

- The setup time constraint depends on the **maximum** delay from register R1 through the combinational logic.
- The input to register R2 must be stable at least t_{setup} before the clock edge.

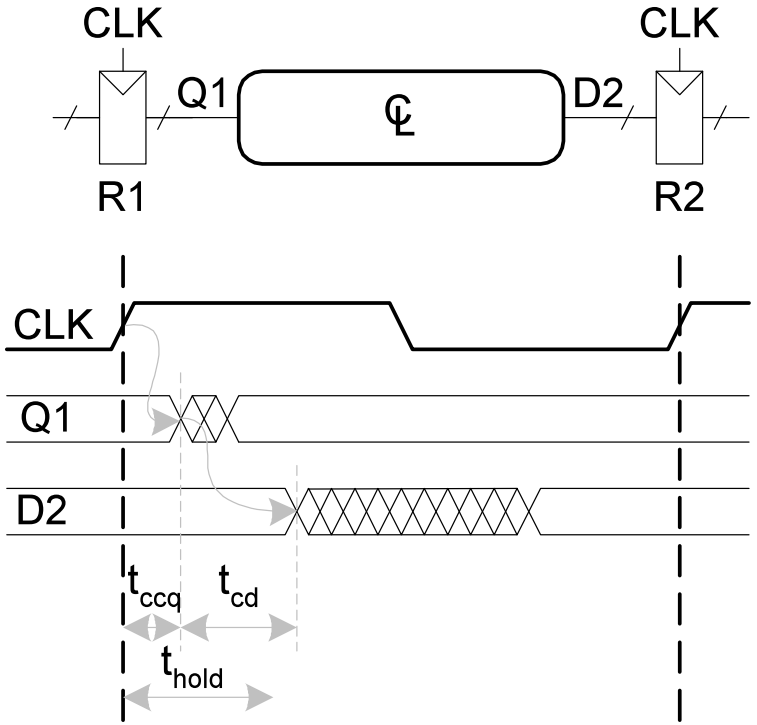


$$T_c \geq t_{pcq} + t_{pd} + t_{\text{setup}}$$

$$t_{pd} \leq T_c - (t_{pcq} + t_{\text{setup}})$$

Hold Time Constraint

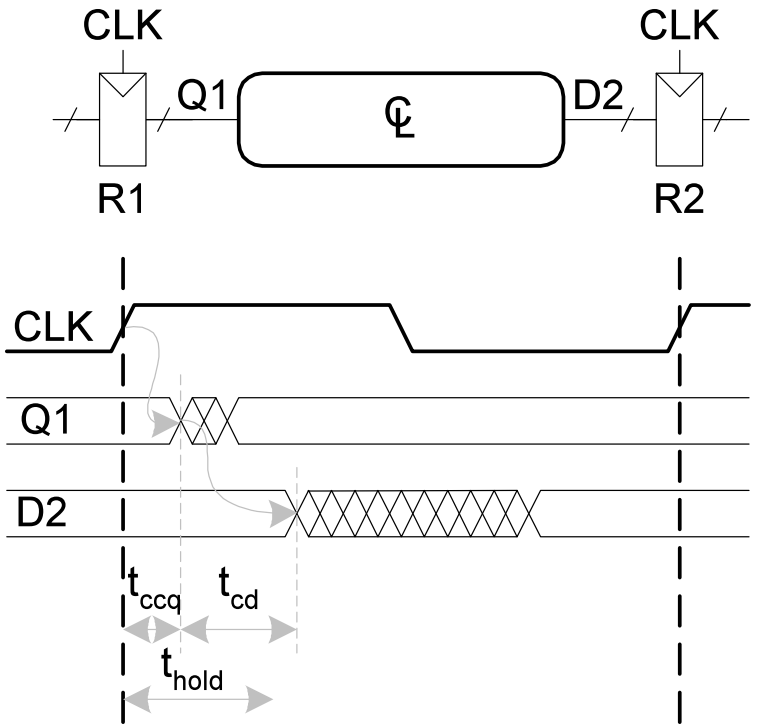
- The hold time constraint depends on the **minimum** delay from register R1 through the combinational logic.
- The input to register R2 must be stable for at least t_{hold} after the clock edge.



$t_{hold} <$

Hold Time Constraint

- The hold time constraint depends on the **minimum** delay from register R1 through the combinational logic.
- The input to register R2 must be stable for at least t_{hold} after the clock edge.

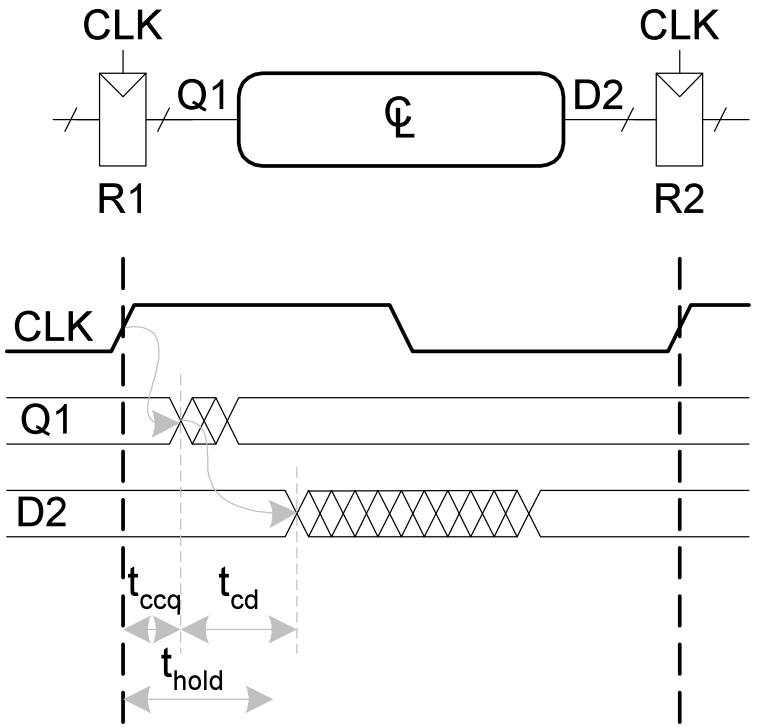


$$t_{\text{hold}} < t_{ccq} + t_{cd}$$

$$t_{cd} >$$

Hold Time Constraint

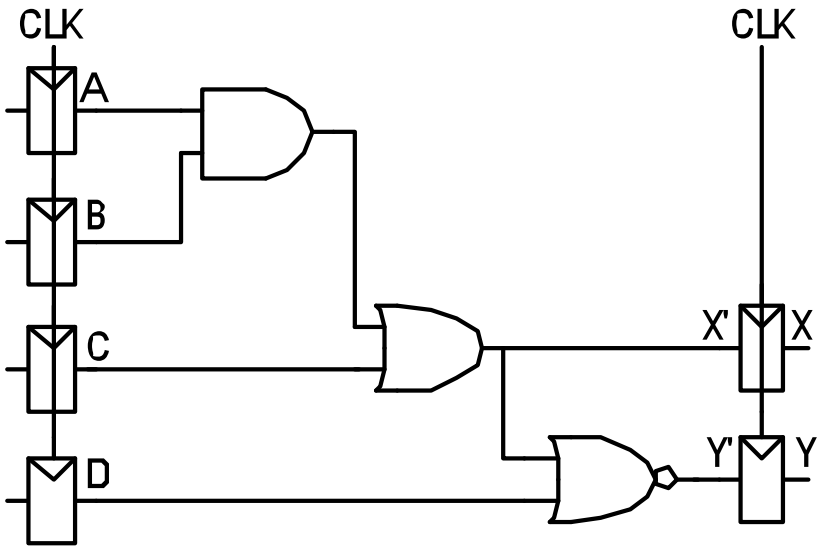
- The hold time constraint depends on the **minimum** delay from register R1 through the combinational logic.
- The input to register R2 must be stable for at least t_{hold} after the clock edge.



$$t_{hold} < t_{ccq} + t_{cd}$$

$$t_{cd} > t_{hold} - t_{ccq}$$

Timing Analysis



Timing Characteristics

t_{ccq} = 30 ps

t_{pcq} = 50 ps

t_{setup} = 60 ps

t_{hold} = 70 ps

per gate

t_{pd} = 35 ps

t_{cd} = 25 ps

t_{pd} =

t_{cd} =

Setup time constraint:

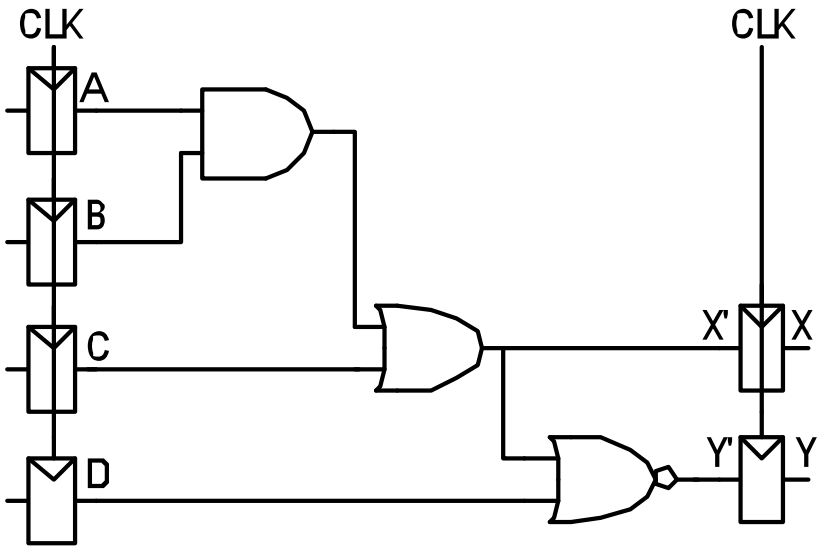
$T_c \geq$

$f_c = 1/T_c =$

Hold time constraint:

$t_{ccq} + t_{pd} > t_{hold}$?

Timing Analysis



Timing Characteristics

t_{ccq} = 30 ps

t_{pcq} = 50 ps

t_{setup} = 60 ps

t_{hold} = 70 ps

per gate

t_{pd} = 35 ps

t_{cd} = 25 ps

t_{pd} = 3 x 35 ps = 105 ps

t_{cd} = 25 ps

Setup time constraint:

$T_c \geq (50 + 105 + 60)$ ps = 215 ps

$f_c = 1/T_c$ = 4.65 GHz

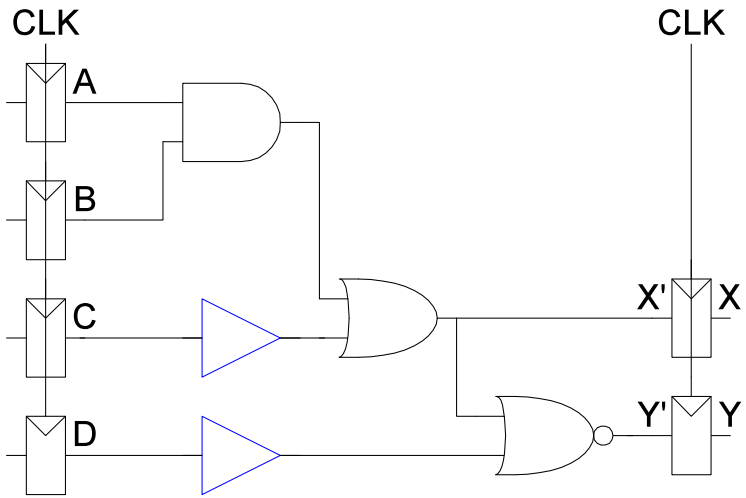
Hold time constraint:

$t_{ccq} + t_{pd} > t_{hold}$?

(30 + 25) ps > 70 ps ? No!

Fixing Hold Time Violation

Add buffers to the short paths:



Timing Characteristics

t_{ccq} = 30 ps

t_{pcq} = 50 ps

t_{setup} = 60 ps

t_{hold} = 70 ps

per gate

t_{pd} = 35 ps

t_{cd} = 25 ps

t_{pd} =

t_{cd} =

Setup time constraint:

$T_c \geq$

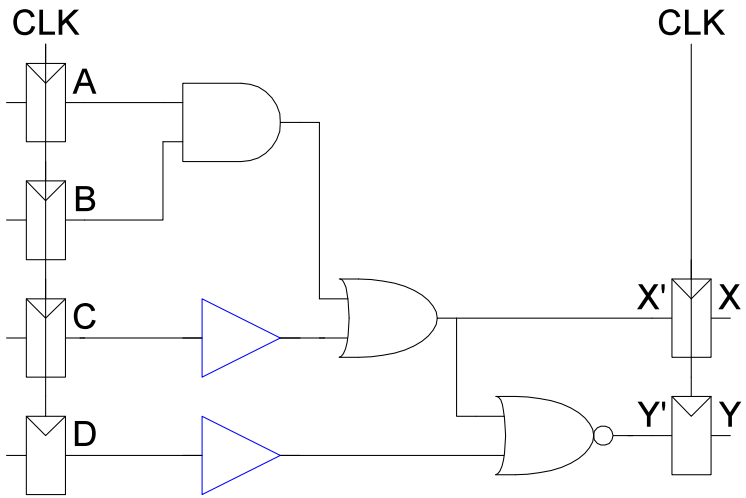
f_c =

Hold time constraint:

$t_{ccq} + t_{pd} > t_{hold}$?

Fixing Hold Time Violation

Add buffers to the short paths:



t_{pd} = 3 x 35 ps = 105 ps

t_{cd} = 2 x 25 ps = 50 ps

Setup time constraint:

$T_c \geq (50 + 105 + 60)$ ps = 215 ps

$f_c = 1/T_c$ = 4.65 GHz

Timing Characteristics

t_{ccq} = 30 ps

t_{pcq} = 50 ps

t_{setup} = 60 ps

t_{hold} = 70 ps

per gate:

t_{pd} = 35 ps

t_{cd} = 25 ps

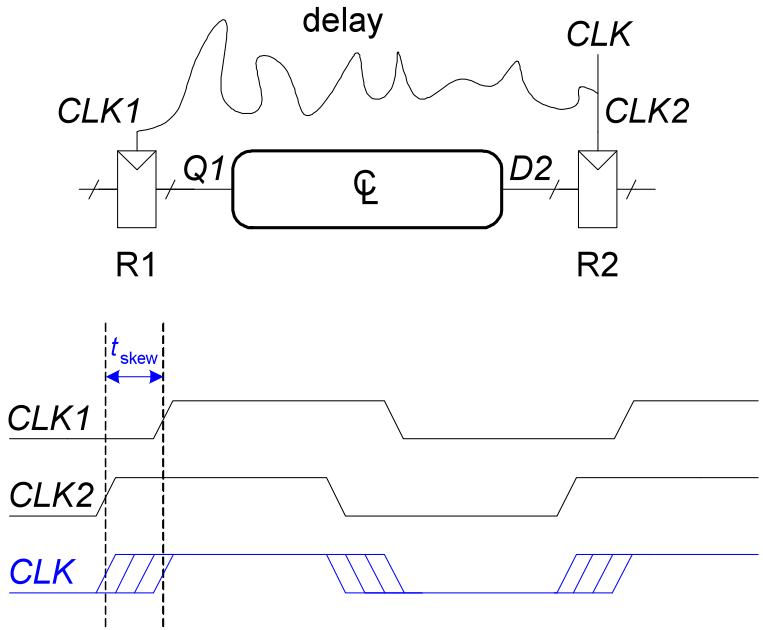
Hold time constraint:

$t_{ccq} + t_{pd} > t_{hold}$?

(30 + 50) ps > 70 ps ? Yes!

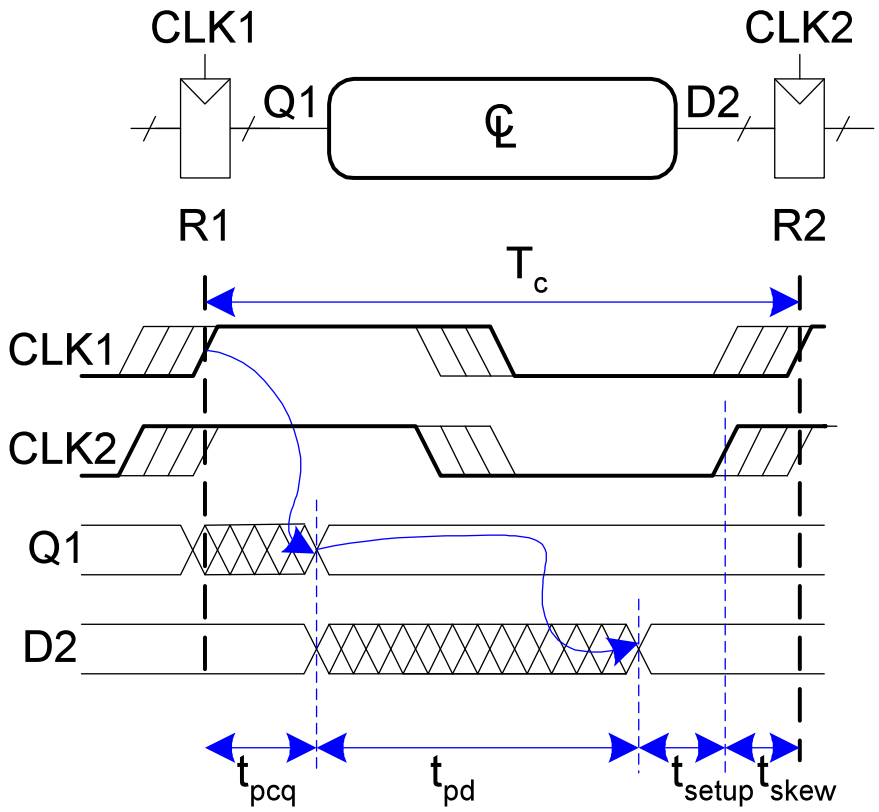
Clock Skew

- The clock doesn't arrive at all registers at the same time
- Skew is the difference between two clock edges
- Examine the worst case to guarantee that the dynamic discipline is not violated for any register – many registers in a system!



Setup Time Constraint with Clock Skew

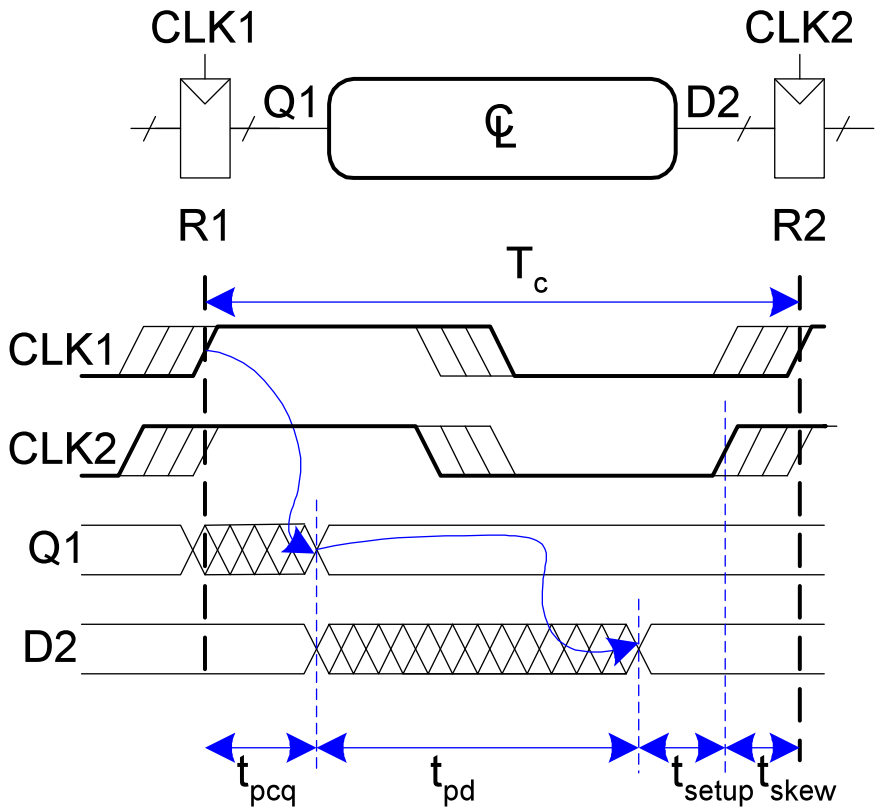
- In the worst case, the CLK2 is earlier than CLK1



$$T_c \geq$$

Setup Time Constraint with Clock Skew

- In the worst case, the CLK2 is earlier than CLK1

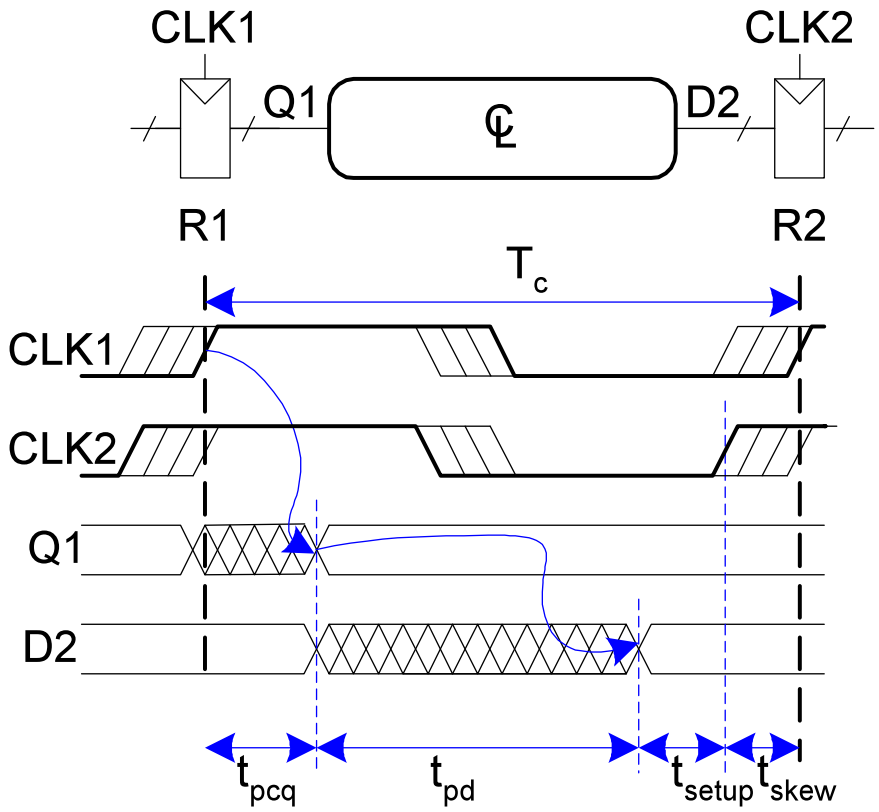


$$T_c \geq t_{pcq} + t_{pd} + t_{\text{setup}} + t_{\text{skew}}$$

$$t_{pd} \leq$$

Setup Time Constraint with Clock Skew

- In the worst case, the CLK2 is earlier than CLK1

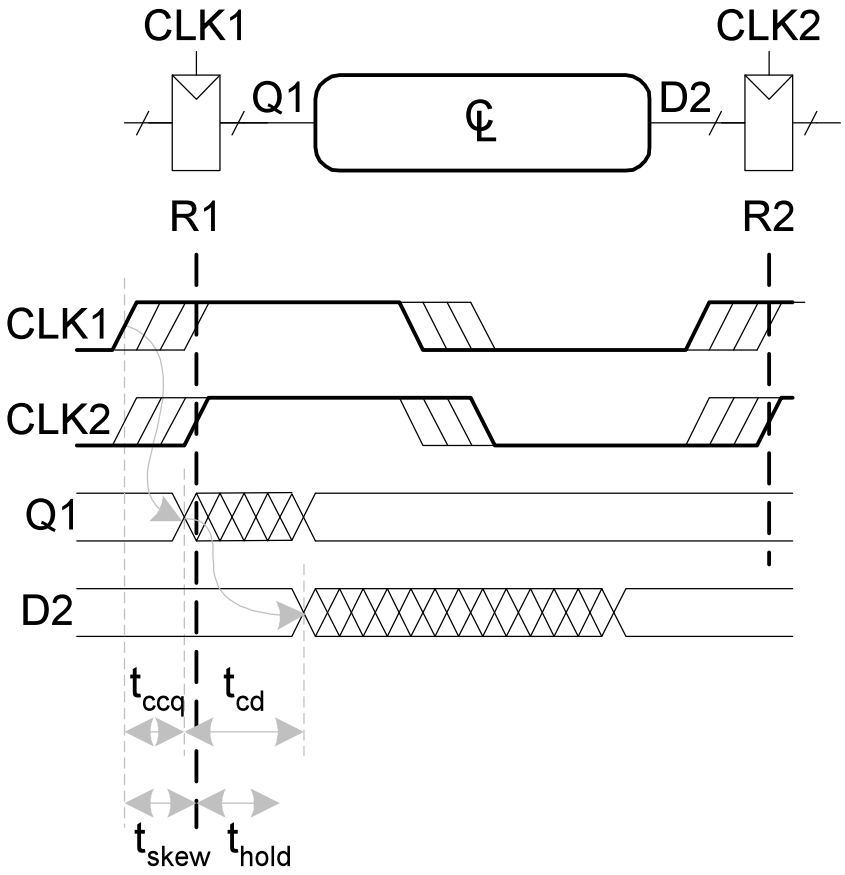


$$T_c \geq t_{pcq} + t_{pd} + t_{\text{setup}} + t_{\text{skew}}$$

$$t_{pd} \leq T_c - (t_{pcq} + t_{\text{setup}} + t_{\text{skew}})$$

Hold Time Constraint with Clock Skew

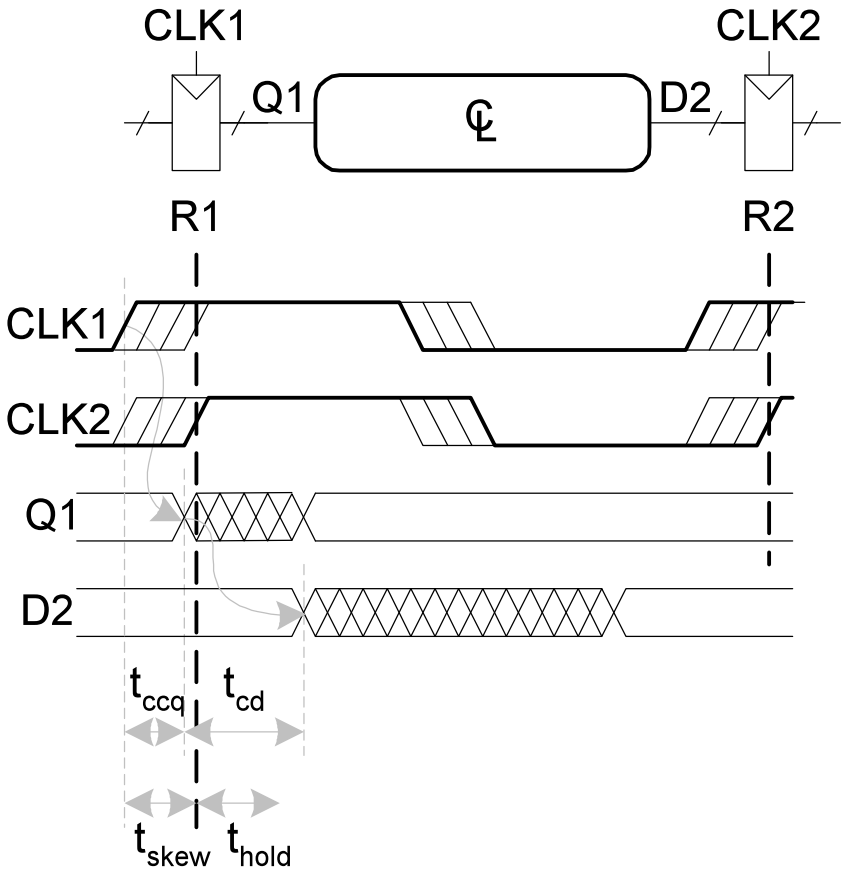
- In the worst case, CLK2 is later than CLK1



$$t_{ccq} + t_{cd} >$$
$$t_{cd} >$$

Hold Time Constraint with Clock Skew

- In the worst case, CLK2 is later than CLK1

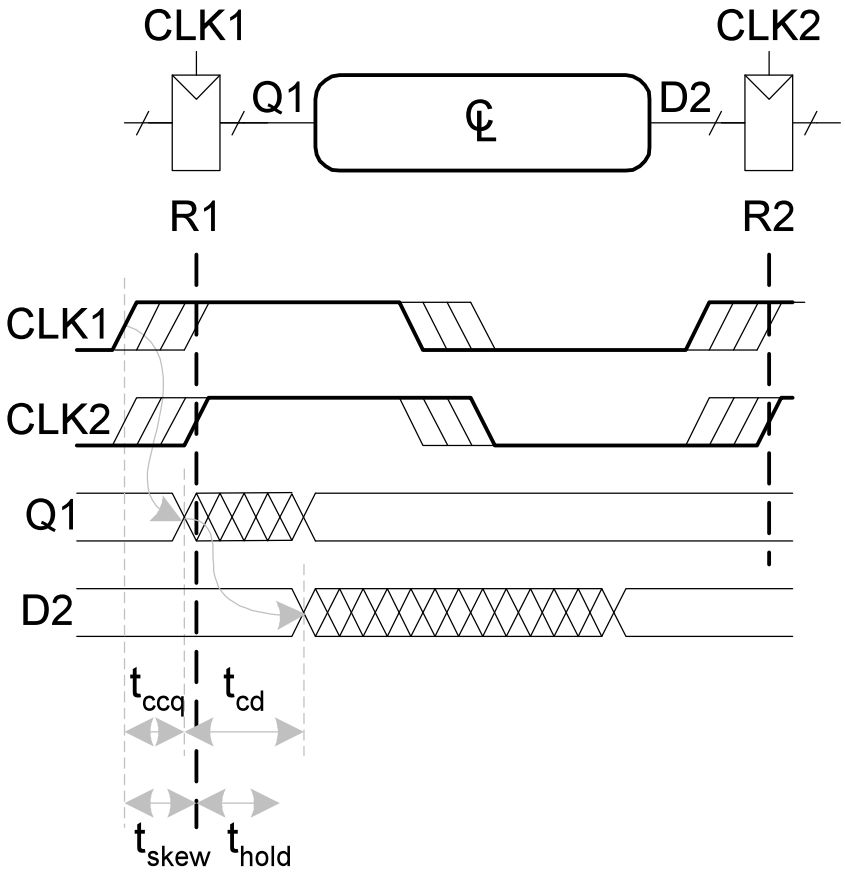


$$t_{ccq} + t_{cd} > t_{\text{hold}} + t_{\text{skew}}$$

$$t_{cd} >$$

Hold Time Constraint with Clock Skew

- In the worst case, CLK2 is later than CLK1

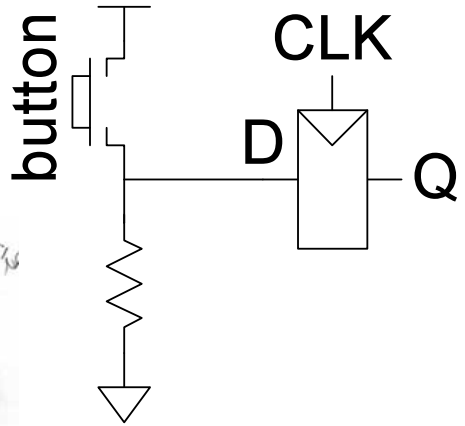


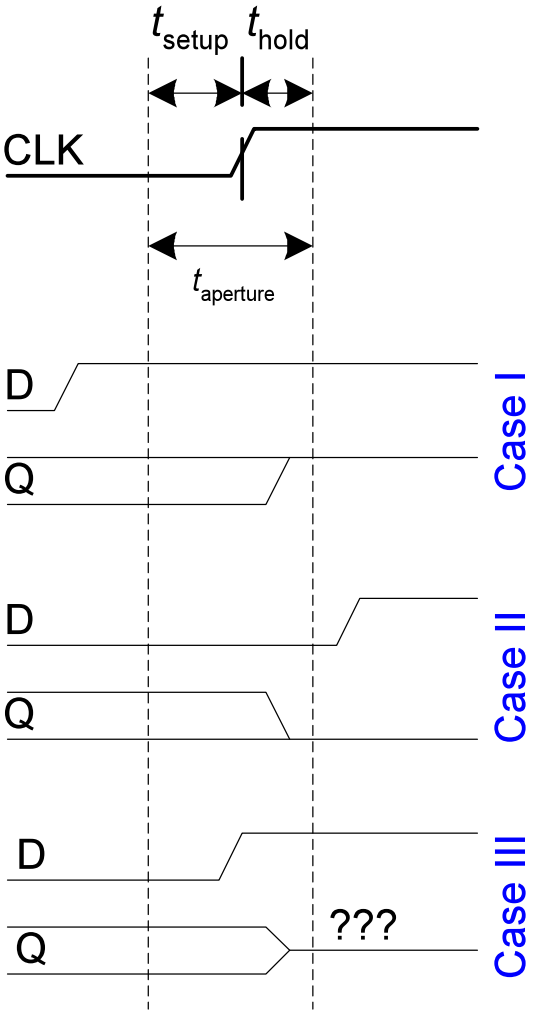
$$t_{ccq} + t_{cd} > t_{\text{hold}} + t_{\text{skew}}$$

$$t_{cd} > t_{\text{hold}} + t_{\text{skew}} - t_{ccq}$$

Violating the Dynamic Discipline

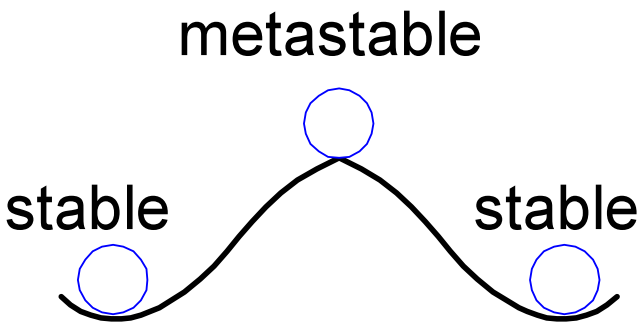
- Asynchronous (for example, user) inputs might violate the dynamic discipline





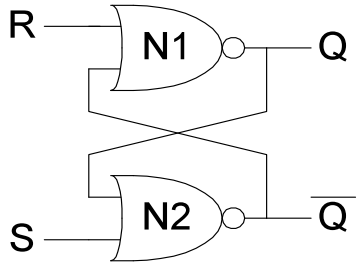
Metastability

- Any bistable device has two stable states and a metastable state between them
- A flip-flop has two stable states (1 and 0) and one metastable state
- If a flip-flop lands in the metastable state, it could stay there for an undetermined amount of time



Flip-flop Internals

- Because the flip-flop has feedback, if Q is somewhere between 1 and 0, the cross-coupled gates will eventually drive the output to either rail (1 or 0, depending on which one it is closer to).



- A signal is considered metastable if it hasn't resolved to 1 or 0
- If a flip-flop input changes at a random time, the probability that the output Q is metastable after waiting some time, t, is:

$$P(t_{res} > t) = (T_0/T_c)\, e^{-t/\tau}$$

t_{res} : time to resolve to 1 or 0

T_0, τ : properties of the circuit

Metastability

- Intuitively:
  - T_0/T_c describes the probability that the input changes at a bad time, i.e., during the aperture time

$$P(t_{res} > t) = (T_0/T_c)\, e^{-t/\tau}$$

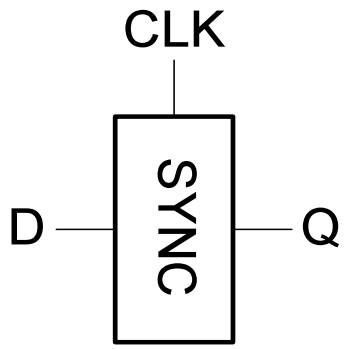
  - τ is a time constant indicating how fast the flip-flop moves away from the metastable state; it is related to the delay through the cross-coupled gates in the flip-flop

$$P(t_{res} > t) = (T_0/T_c)\, e^{-t/\tau}$$

- In short, if a flip-flop samples a metastable input, if you wait long enough (t), the output will have resolved to 1 or 0 with high probability.

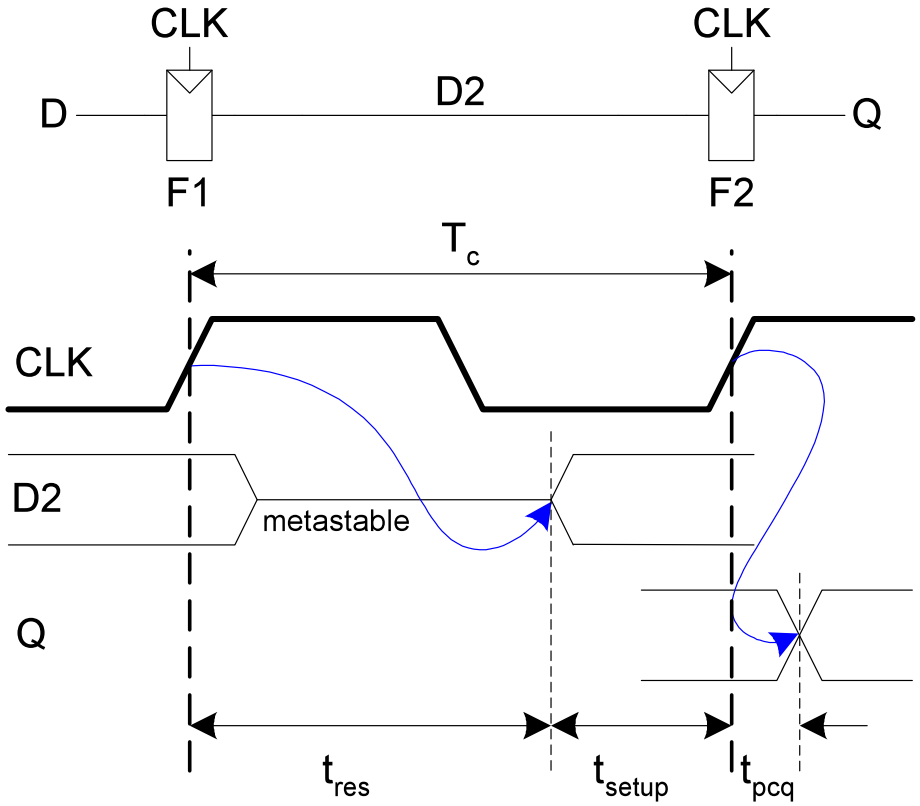
Synchronizers

- Asynchronous inputs (D) are inevitable (user interfaces, systems with different clocks interacting, etc.).
- The goal of a synchronizer is to make the probability of failure (the output Q still being metastable) low.
- A synchronizer cannot make the probability of failure 0.



Synchronizer Internals

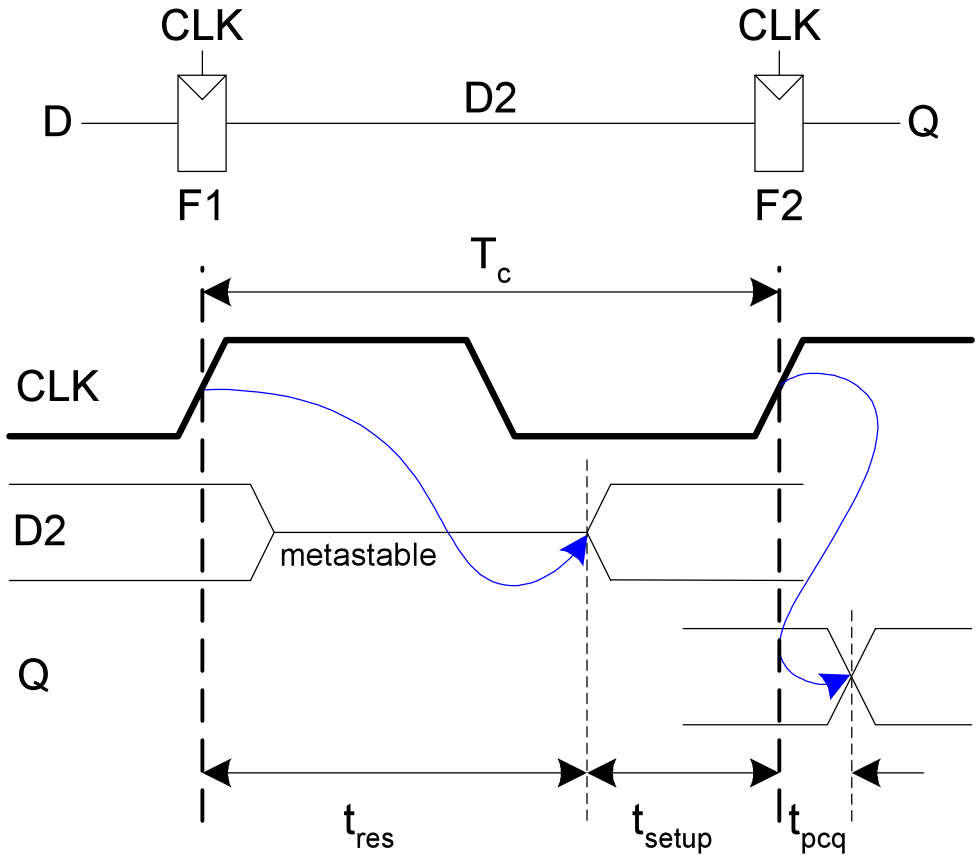
- A synchronizer can be built with two back-to-back flip-flops.
- Suppose the input D is transitioning when it is sampled by flip-flop 1, F1.
- The amount of time the internal signal D2 can resolve to a 1 or 0 is (T_c - t_{setup}).



Synchronizer Probability of Failure

For each sample, the probability of failure of this synchronizer is:

$$P(\text{failure}) = (T_0/T_c)\, e^{-(T_c - t_{setup})/\tau}$$



Synchronizer Mean Time Before Failure

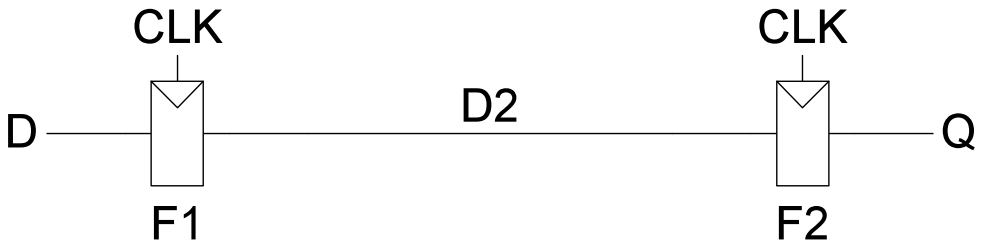
- If the asynchronous input changes once per second, the probability of failure per second of the synchronizer is simply *P*(failure).
- In general, if the input changes *N* times per second, the probability of failure per second of the synchronizer is:

$$P(\text{failure})/\text{second} = (NT_0/T_c)\, e^{-(T_c - t_{setup})/\tau}$$

- Thus, the synchronizer fails, on average, 1/[*P*(failure)/second]
- This is called the *mean time between failures*, MTBF:

$$\text{MTBF} = 1/[P(\text{failure})/\text{second}] = (T_c/NT_0)\, e^{(T_c - t_{setup})/\tau}$$

Example Synchronizer



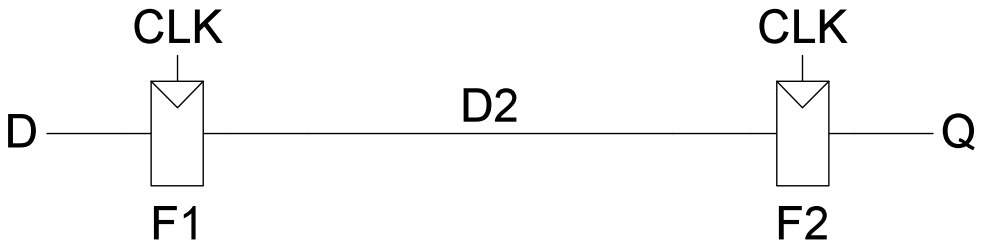
- Suppose: T_c = 1/500 MHz = 2 ns τ = 200 ps
  T_0 = 150 ps t_{setup} = 100 ps
  N = 10 events per second
- What is the probability of failure? MTBF?

$P(\text{failure}) =$

$P(\text{failure})/\text{second} =$

$\text{MTBF} =$

Example Synchronizer



- Suppose: T_c = 1/500 MHz = 2 ns τ = 200 ps
  T_0 = 150 ps t_{setup} = 100 ps
  N = 10 events per second
- What is the probability of failure? MTBF?

$$P(\text{failure}) = (150 \text{ ps}/2 \text{ ns})\, e^{-(1.9 \text{ ns})/200 \text{ ps}}$$
$$= 5.6 \times 10^{-6}$$
$$P(\text{failure})/\text{second} = 10 \times (5.6 \times 10^{-6})$$
$$= 5.6 \times 10^{-5} / \text{second}$$
$$\text{MTBF} = 1/[\text{P(failure)/second}] \approx 5 \text{ hours}$$