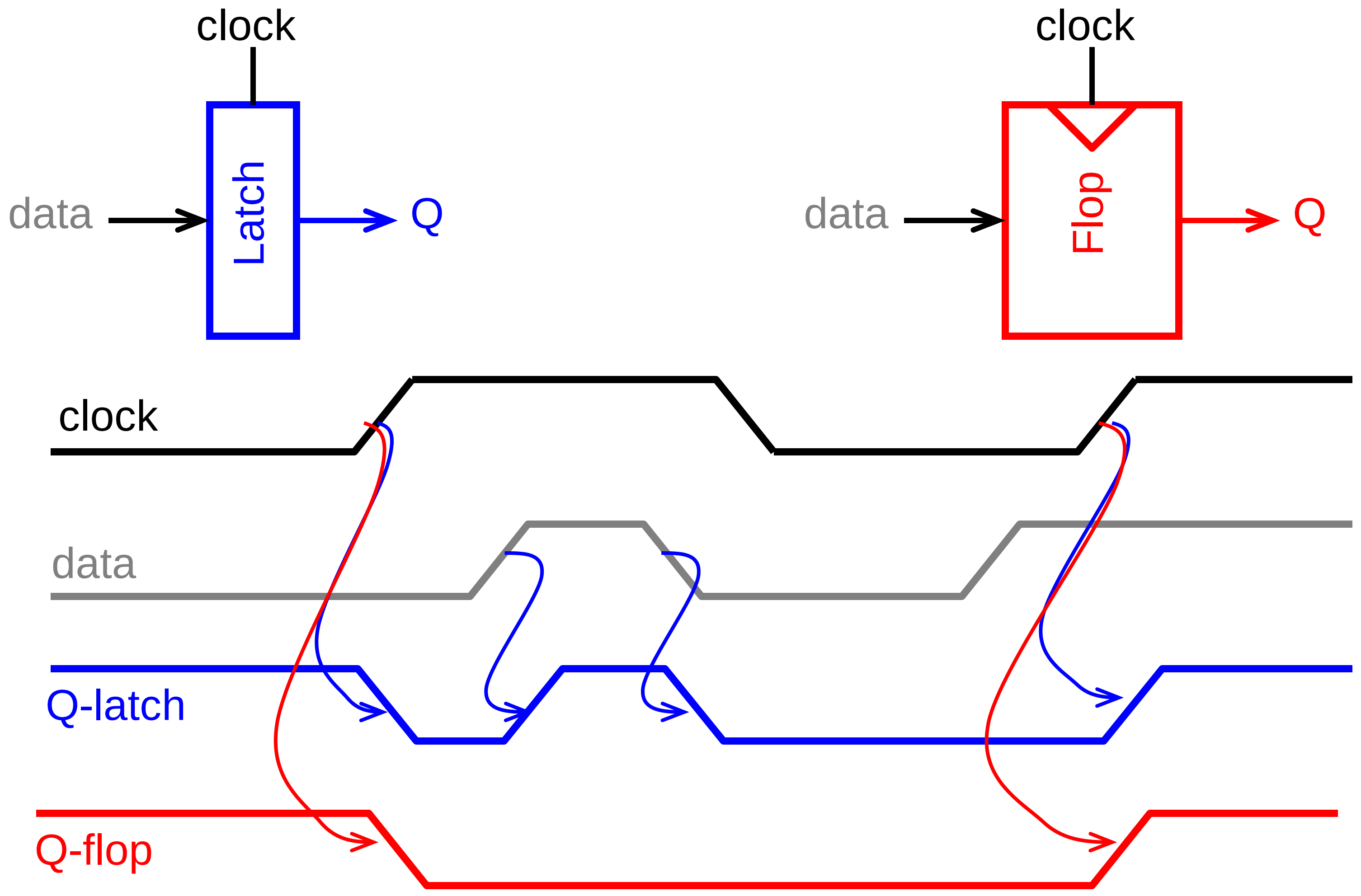
Sequential Circuit Design

Shmuel Wimer
Bar Ilan University, Engineering Faculty
Technion, EE Faculty

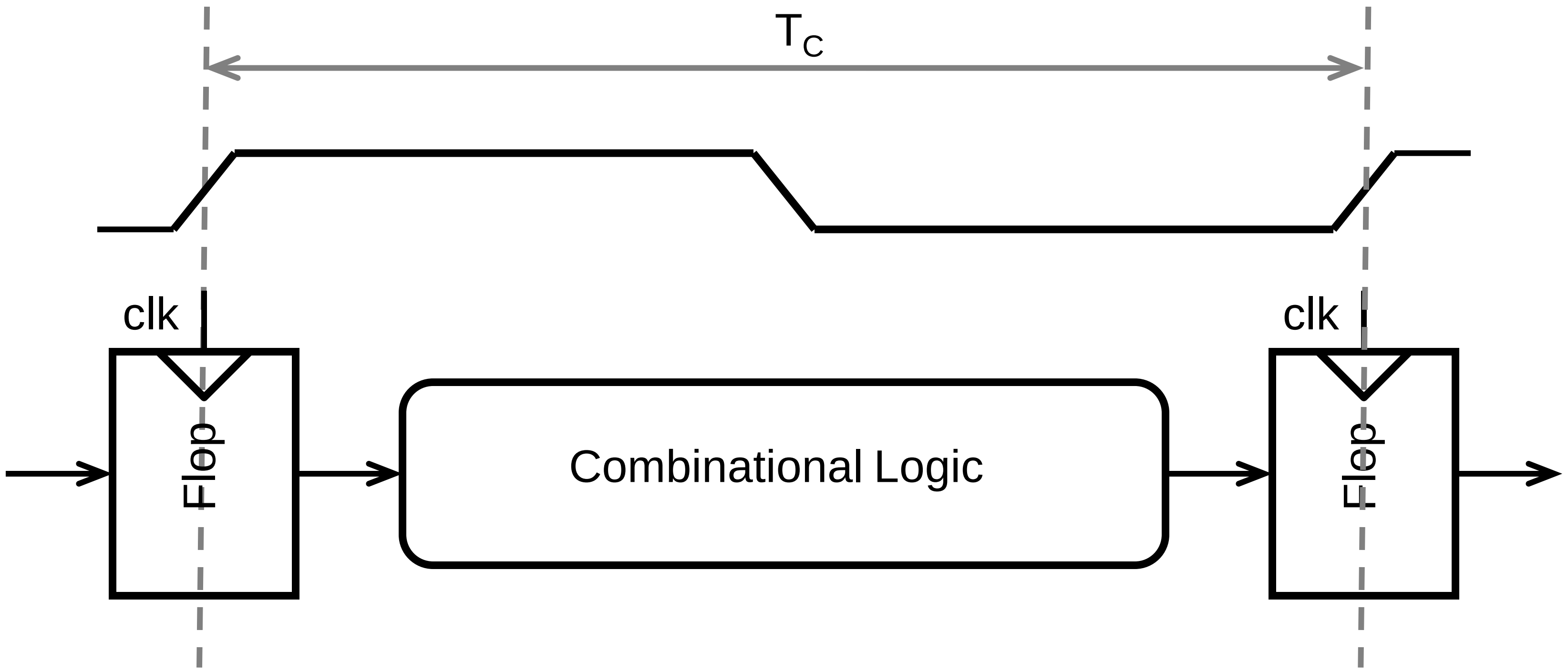
Memory Elements



Latch is transparent on high clock and opaque on low clock.

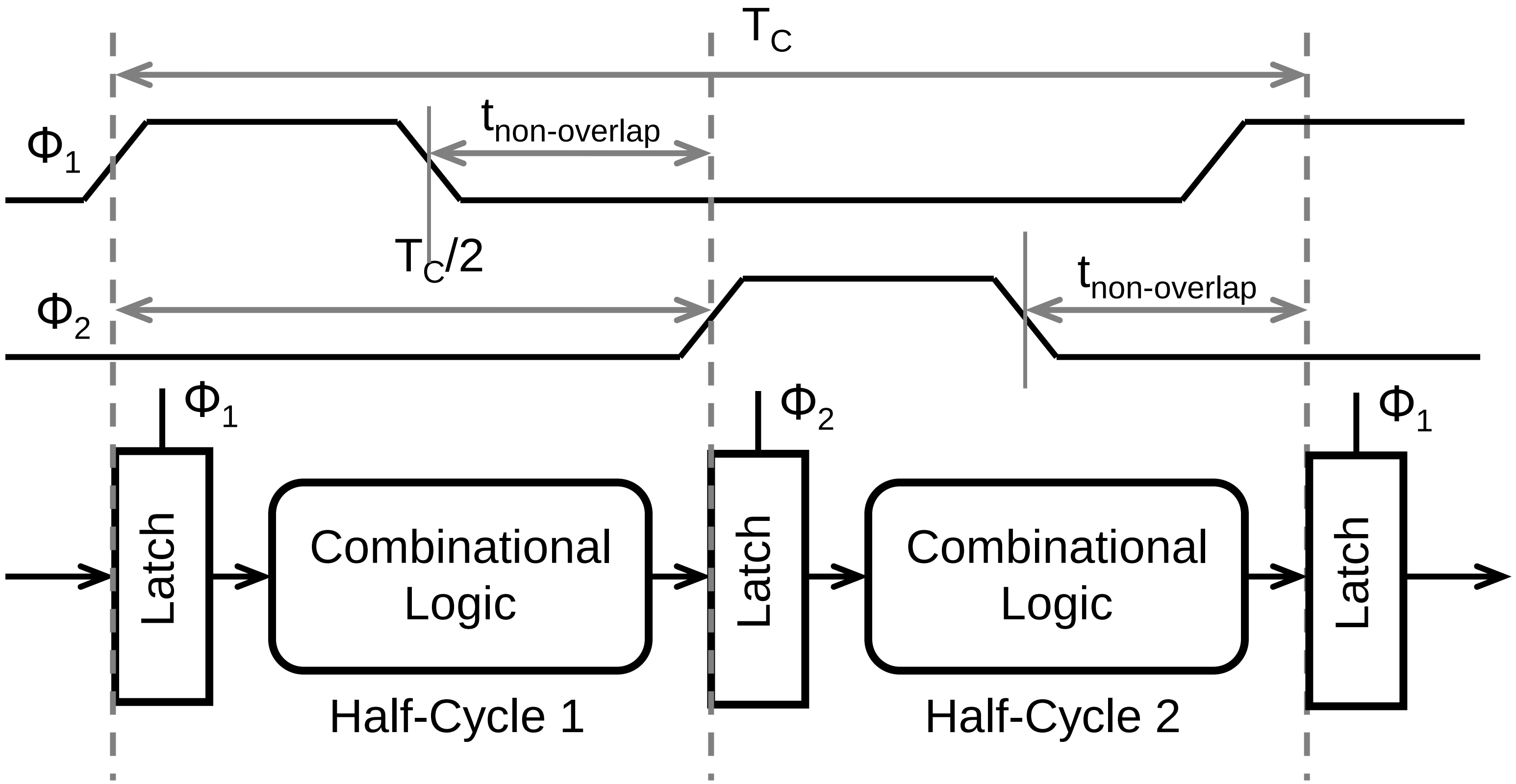
Flip-flop is edge triggered. It transfers input data to Q on clock rising edge.

Static Sequencing by Flip-Flops



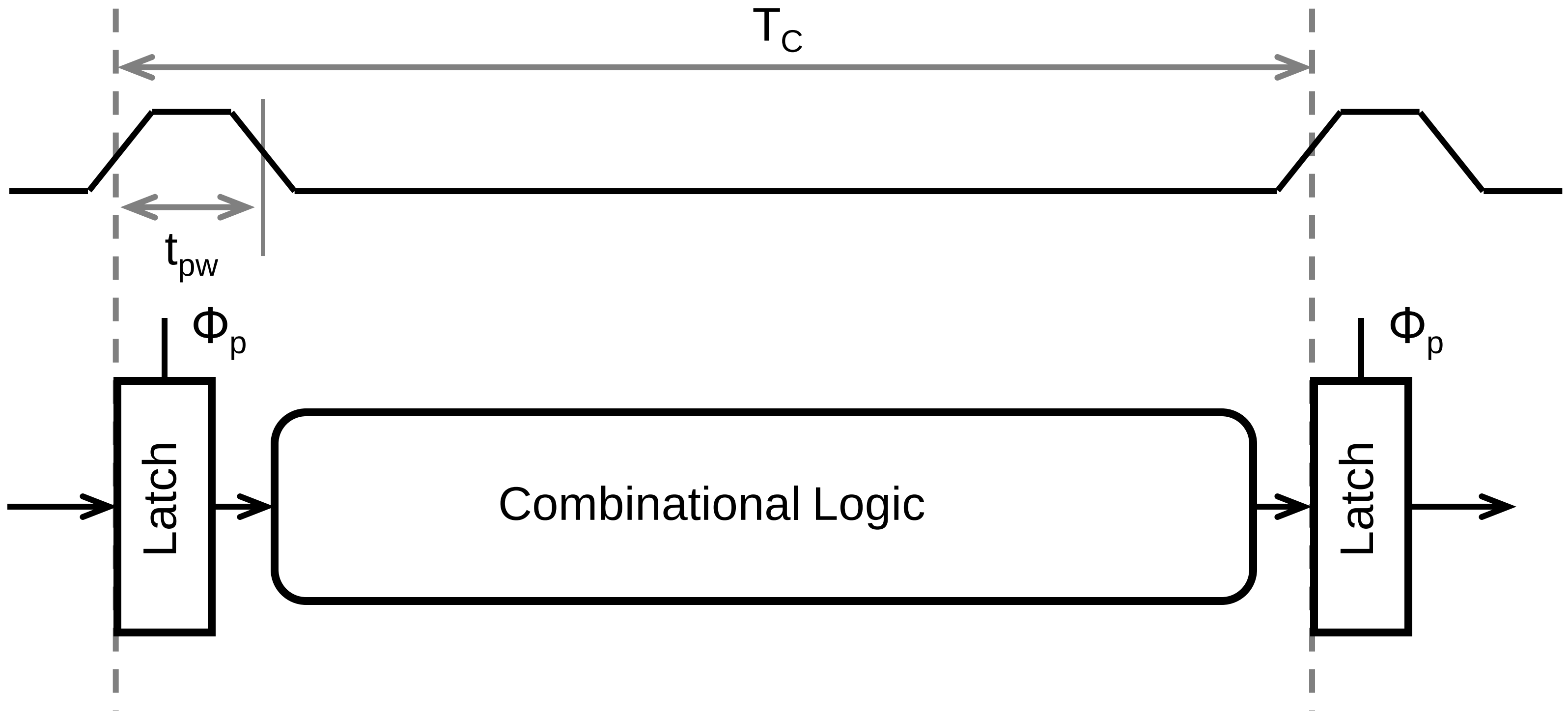
One flip-flop is used on each cycle boundary. Tokens advance from one cycle to the next on rising edge.

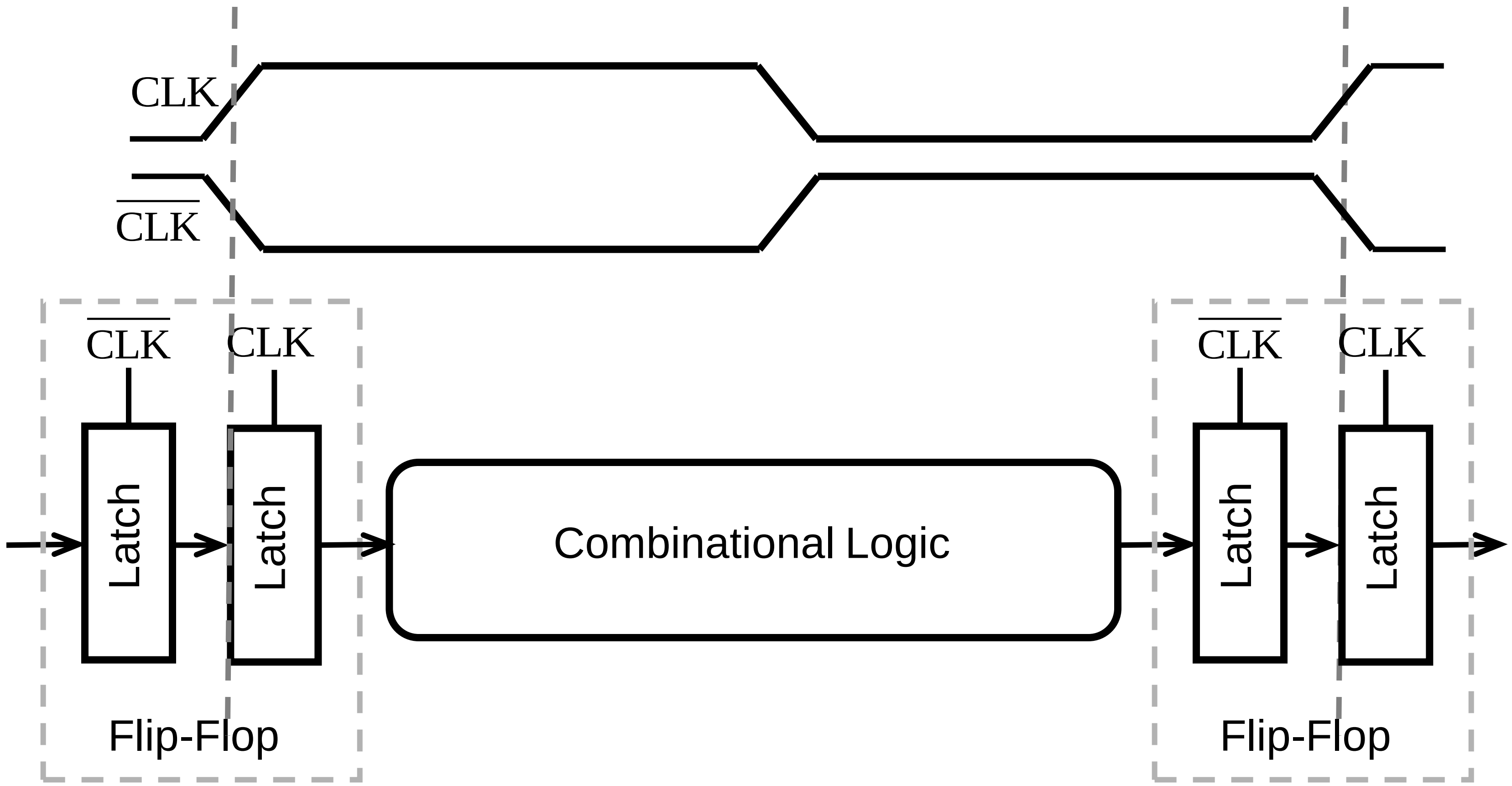
Static Sequencing by Latches



2-phase system. Phases may be separated by some non overlapping time.

Static Sequencing by Pulsed Latches





Flip-flop sequencing can be viewed as a back-to-back latch pair

Sequencing Elements Timing Notations

t_{pd}

Logic Propagation Delay

t_{cd}

Logic Contamination Delay

t_{pcq}

Latch / Flop Clock-to-*Q* Propagation Delay

t_{ccq}

Latch / Flop Clock-to-*Q* Contamination Delay

t_{pdq}

Latch / Flop *D*-to-*Q* Propagation Delay

t_{cdq}

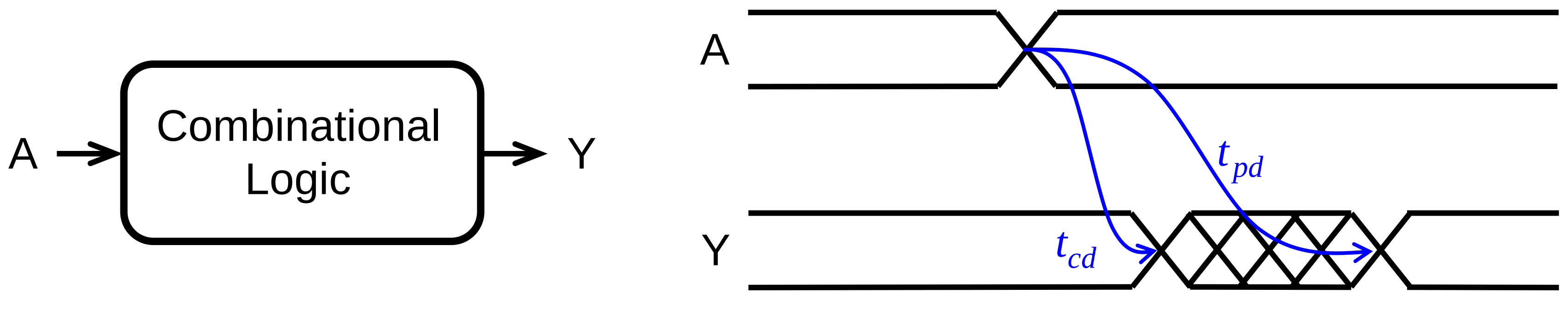
Latch / Flop *D*-to-*Q* Contamination Delay

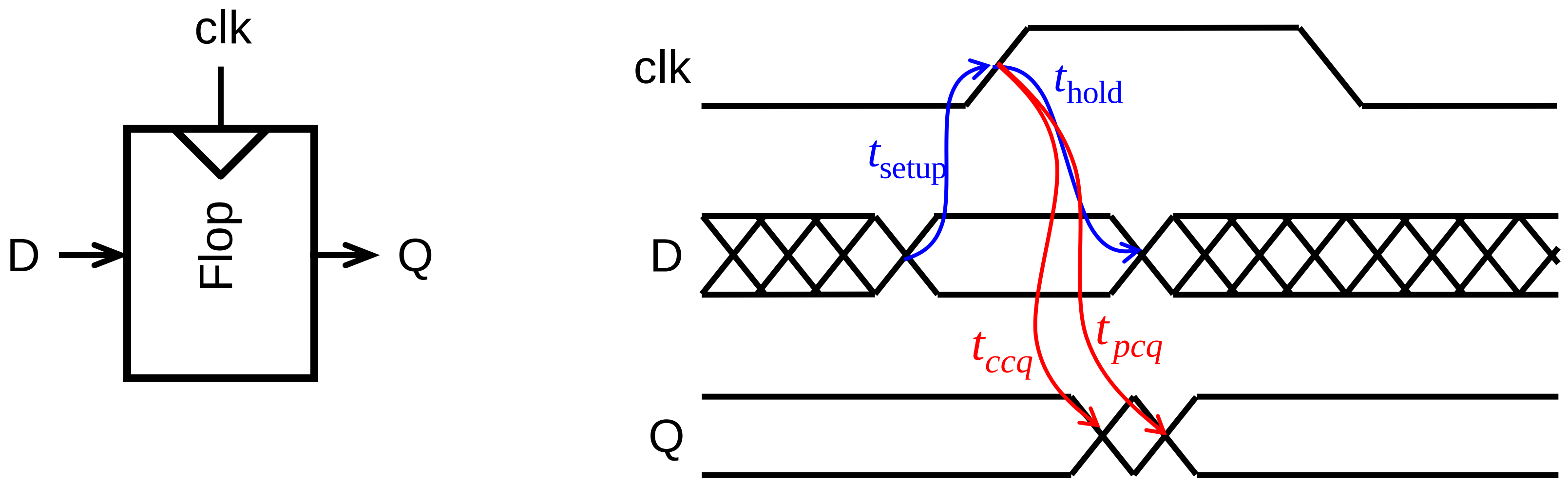
t_{setup}

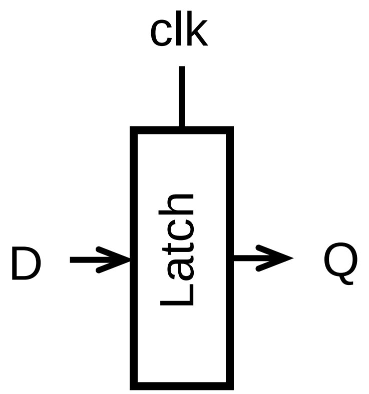
Latch / Flop Setup Time

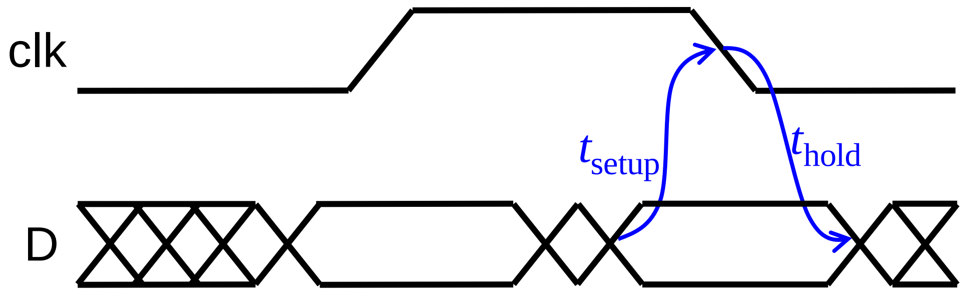
t_{hold}

Latch / Flop Hold Time

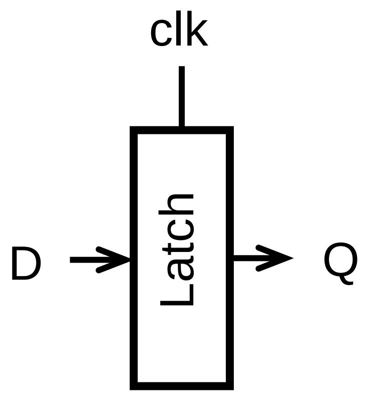

A
Combinational Logic
Y
A
Y
t_{pd}
t_{cd}

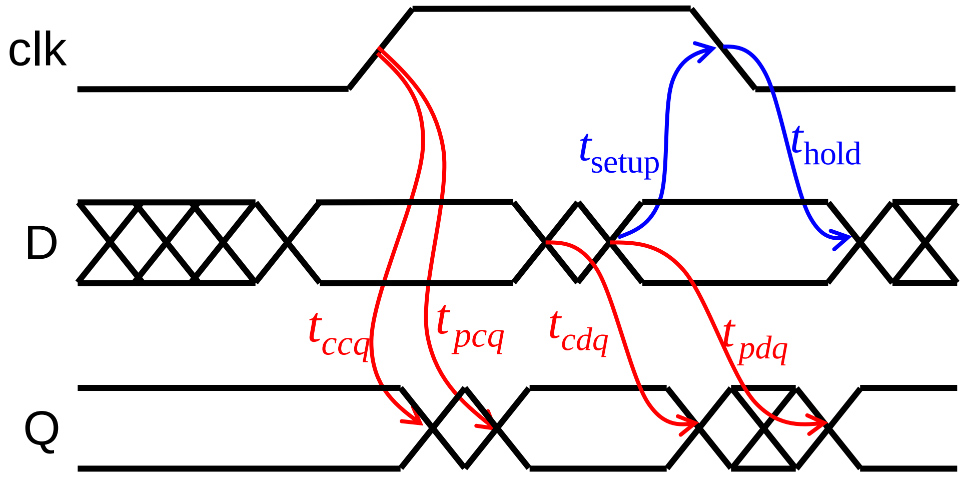

clk
D
Flop
Q
clk
D
Q
t_{setup}
t_{hold}
t_{ccq}
t_{pcq}





Latch is transparent when clock is high. In order for a data change to transfer to output, the latest change must occur at t_{setup} before latch turns to opaque. It must sustain t_{hold} after latch turns opaque.

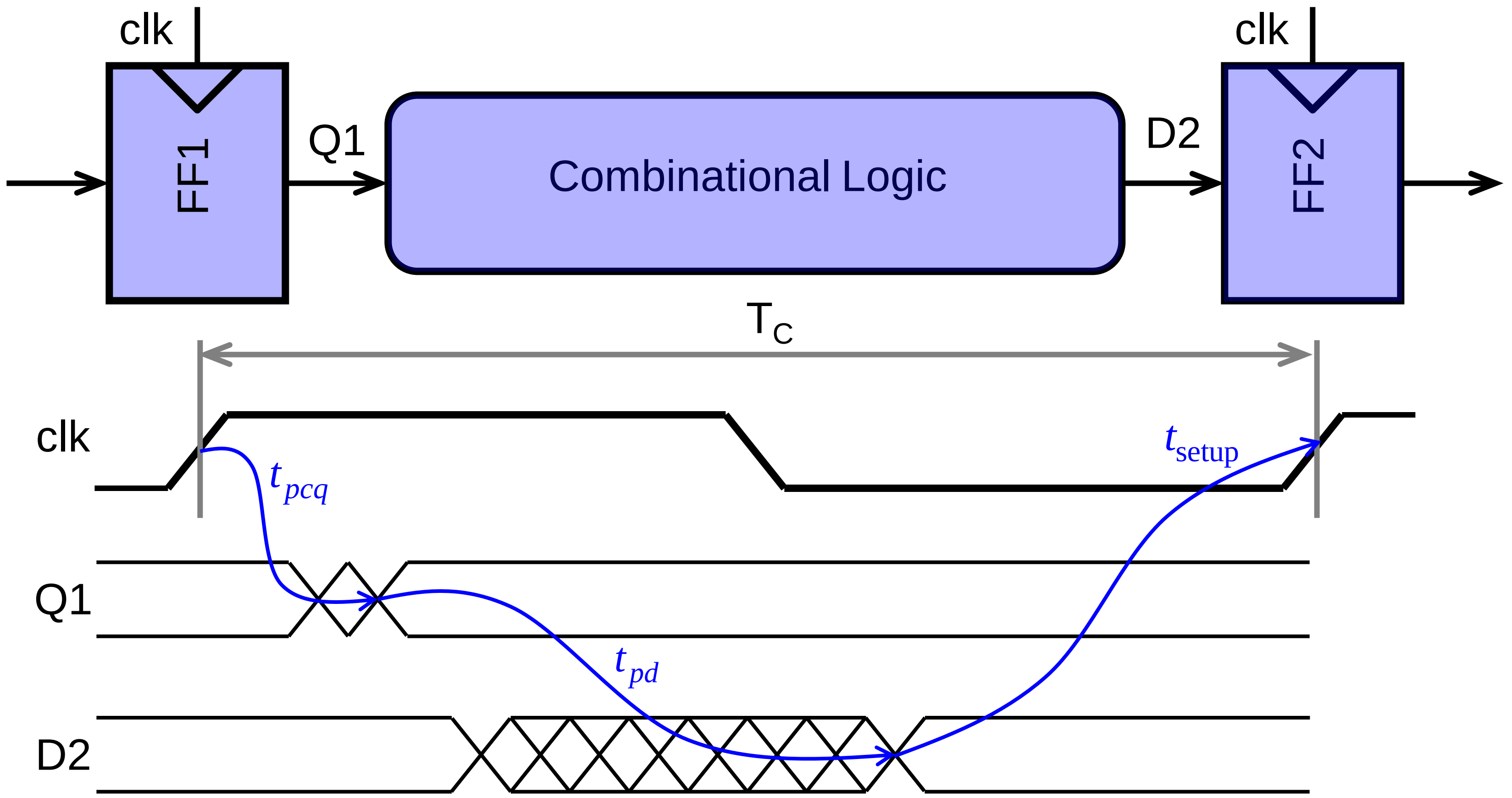




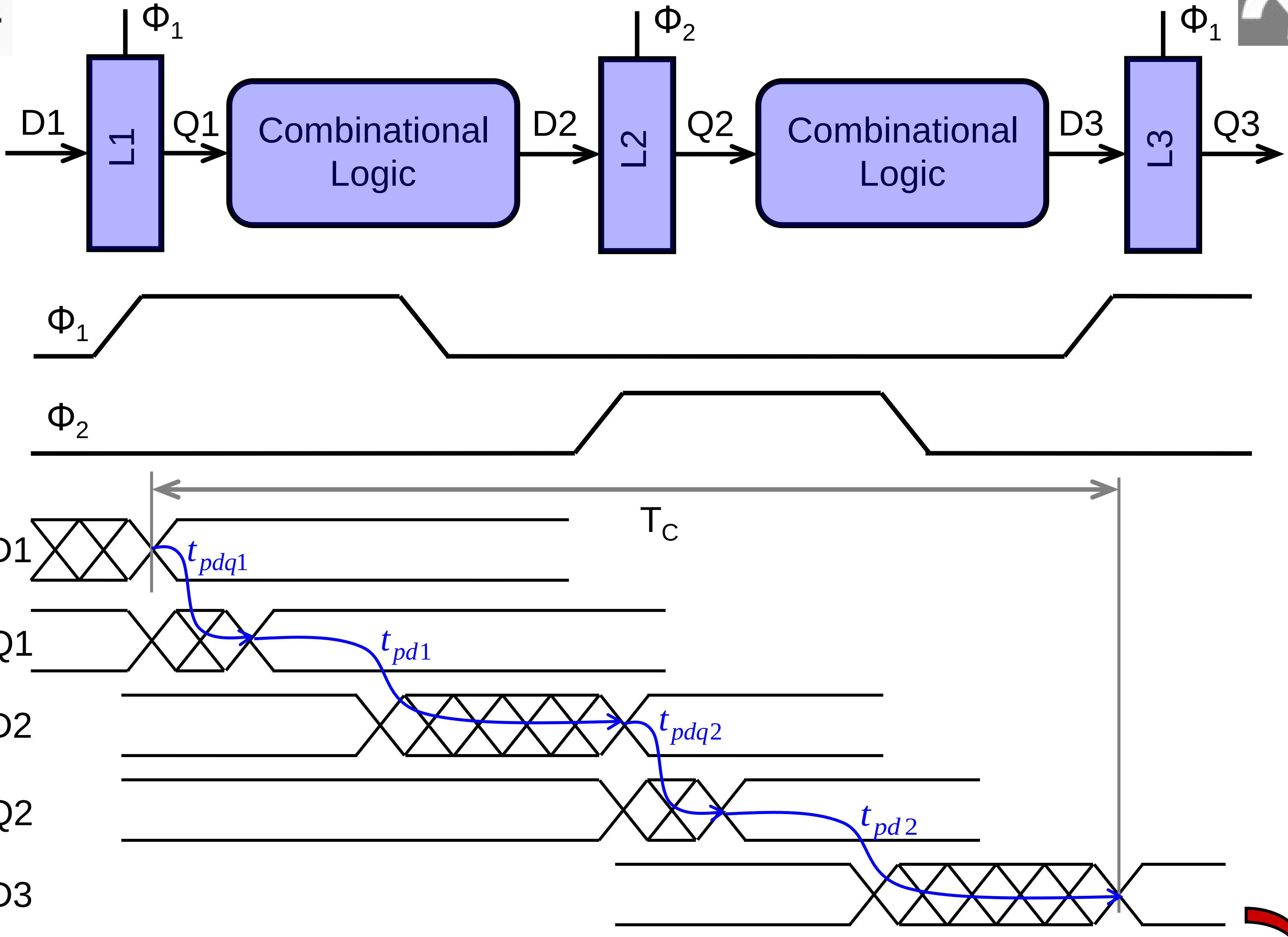
When a latch turns transparent the data is transferred to output at min delay of t_{ccq} and max delay of t_{pcq}.

Data change at transparency is transferred to output at min delay of t_{cdq} and max delay of t_{pdq} .

Max-Delay Constraints



$$T_c \geq t_{pcq} + t_{pd} + t_{\text{setup}} \quad \Rightarrow \quad t_{pd} \leq T_c - \underbrace{\left(t_{\text{setup}} + t_{pcq}\right)}_{\text{sequencing overhead}}$$



Φ_1
Φ_2
Φ_1
D1
L1
Q1
Combinational Logic
D2
L2
Q2
Combinational Logic
D3
L3
Q3
Φ_1
Φ_2
T_C
D1
Q1
D2
Q2
D3
t_{pdq1}
t_{pd1}
t_{pdq2}
t_{pd2}

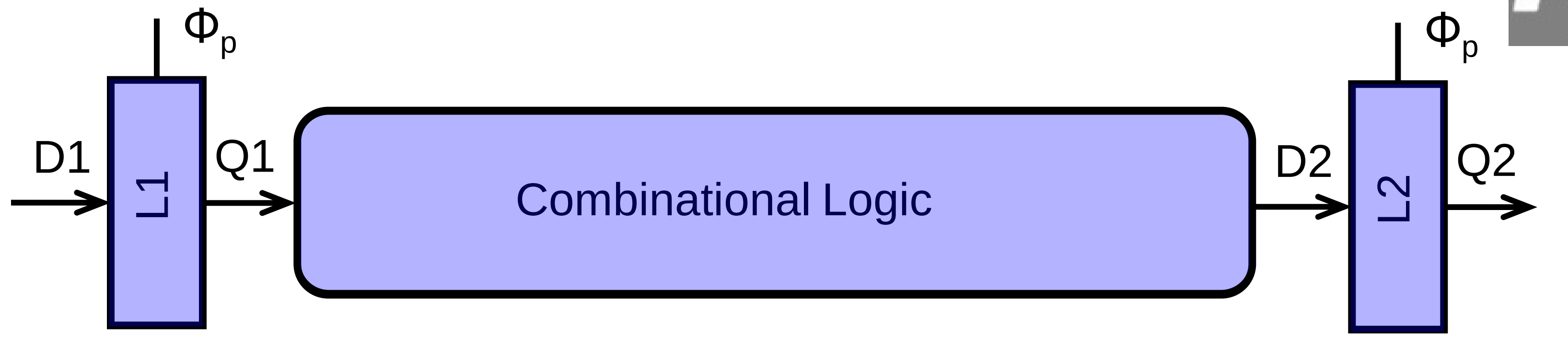
$$T_c \geq t_{pdq1} + t_{pd1} + t_{pdq2} + t_{pd2}$$

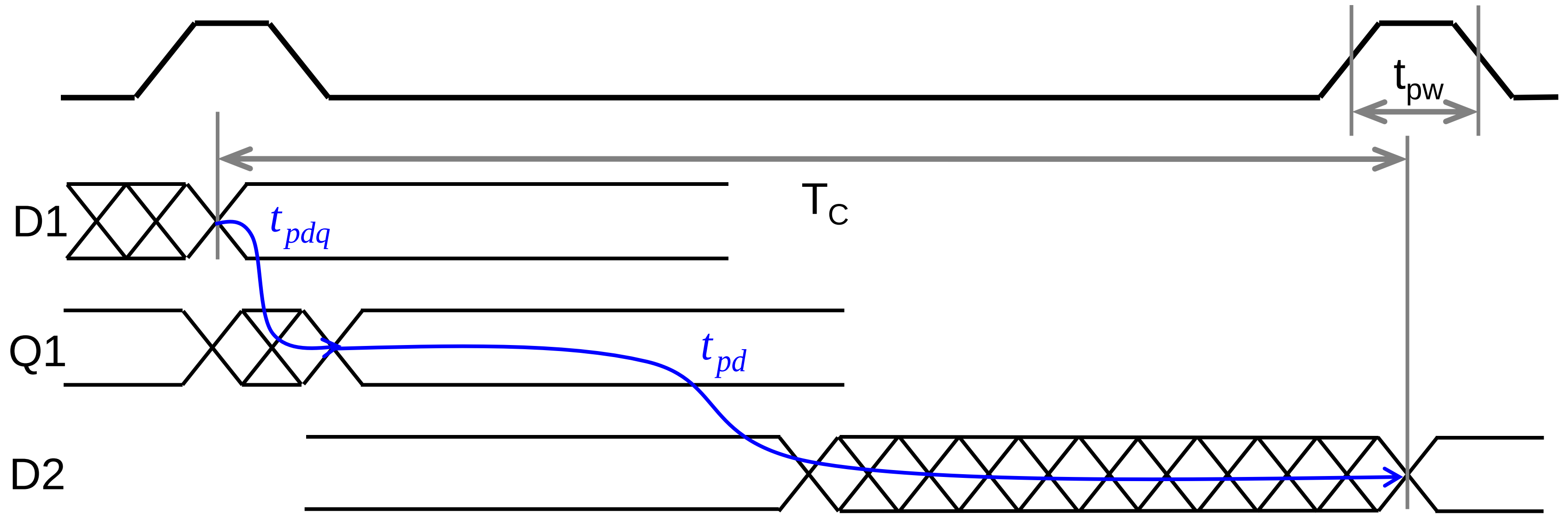
$$\Rightarrow \; t_{pd1} + t_{pd2} \leq T_c - \underbrace{\left(t_{pdq1} + t_{pdq2}\right)}_{\text{sequencing overhead}}$$

Notice that the non overlap between clocks doesn't degrade performance.

Flip-Flop can be realized by two latches connected back to back, yielding expression similar to Flip-Flop sequencing.

$$t_{pd} = t_{pd1} + t_{pd2} \leq T_c - \underbrace{\left(2t_{pdq}\right)}_{\substack{\text{sequencing} \\ \text{overhead}}}$$

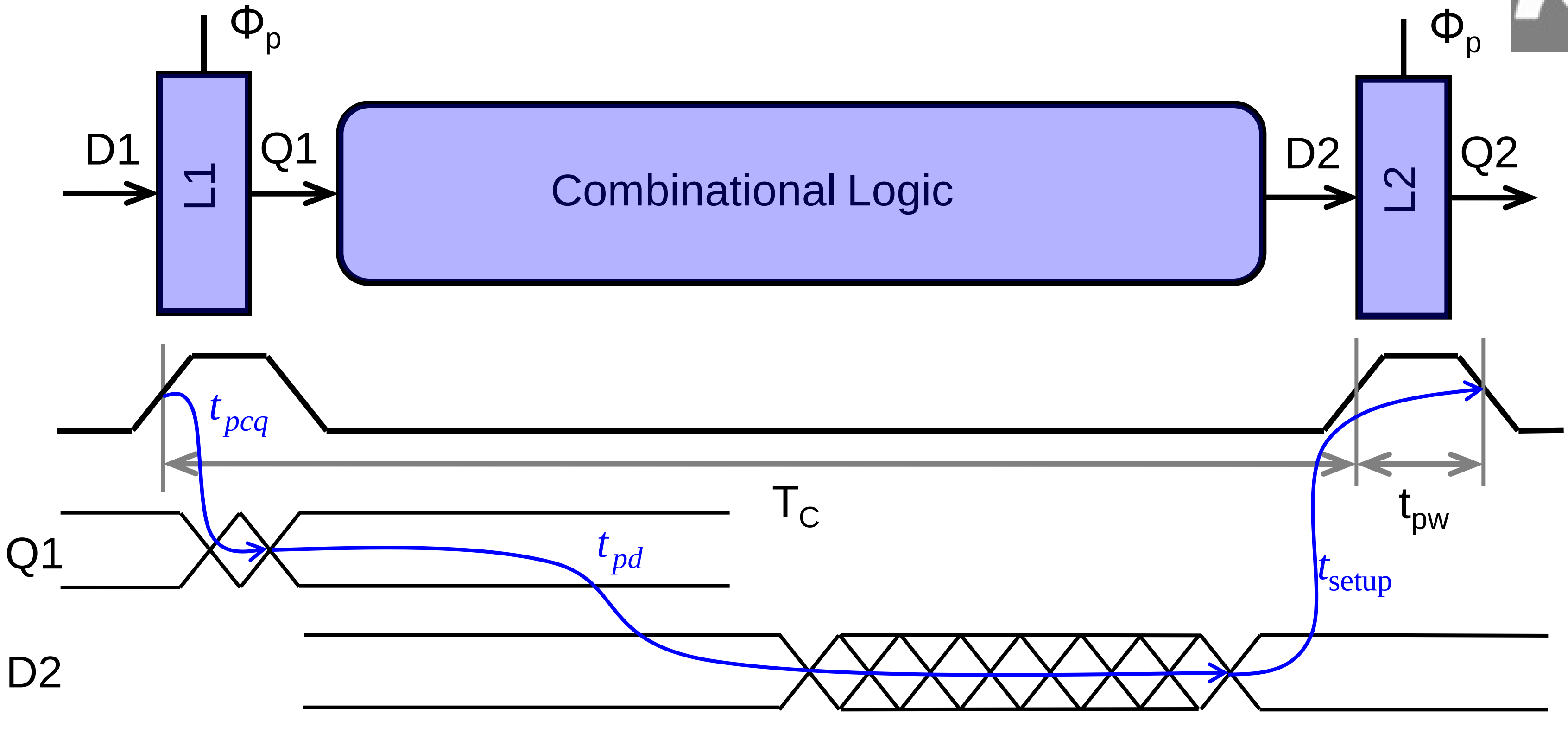

Φ_p
Φ_p
D1
L1
Q1
Combinational Logic
D2
L2
Q2


t_{pw}
T_C
D1
t_{pdq}
Q1
t_{pd}
D2

If the pulse width is wide enough the max delay is similar to two-phase latches, except that only one latch is in the critical path.

$$t_{pw} \geq t_{\text{setup}}: \quad \Rightarrow \quad T_c \geq t_{pdq} + t_{pd}$$

If the pulse width is narrower than the setup time, the data must be set up before the pulse falls.



$$t_{pw} < t_{\text{setup}}: \quad \Rightarrow \quad T_c \geq t_{pcq} + t_{pd} + t_{\text{setup}} - t_{pw}$$

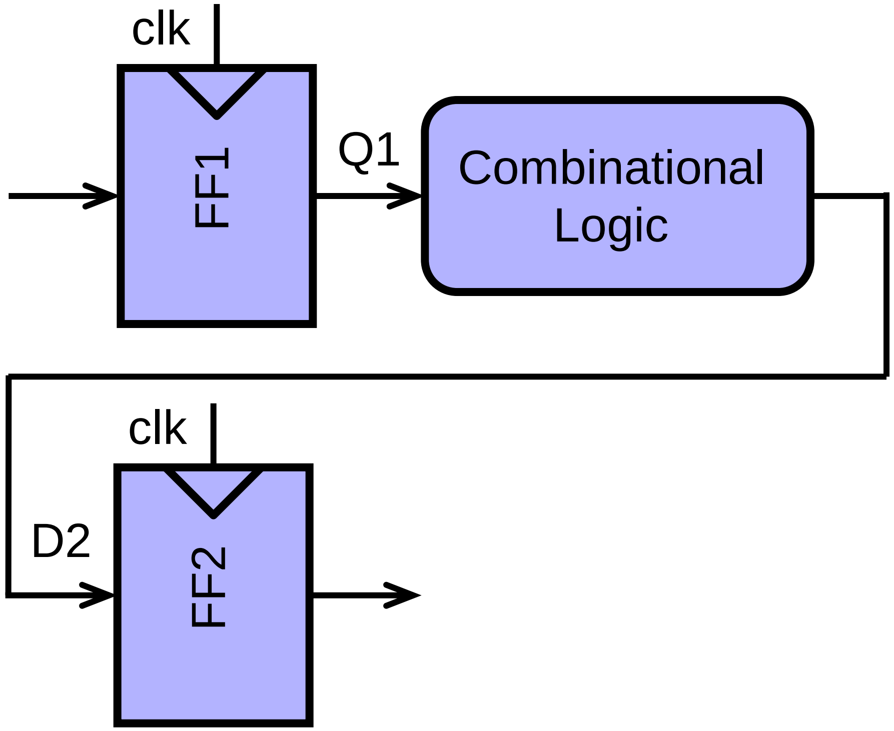
Consequently: $$t_{pd} \leq T_c - \underbrace{\max\left\{t_{pdq}, t_{pcq} + t_{\text{setup}} - t_{pw}\right\}}_{\text{sequencing overhead}}$$

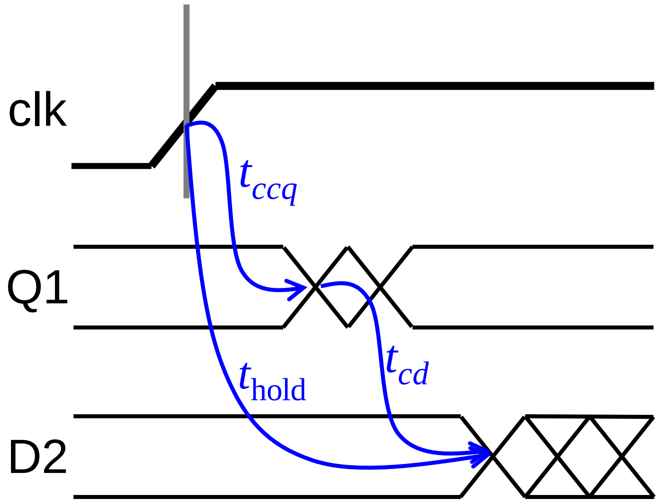
Min-Delay Constraints

Logic circuits cannot be too fast.

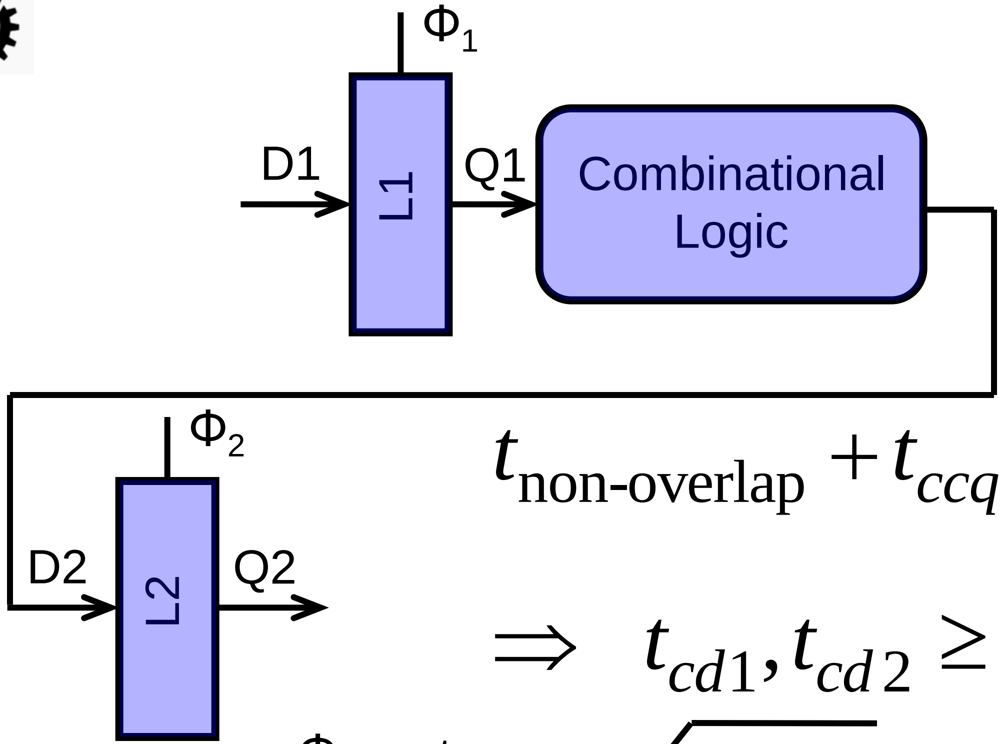
Otherwise the input data to next sequential circuit will change while it is still holding its current data.

Such malfunction is called ***race condition***, ***hold time failure*** or ***min-delay failure***.



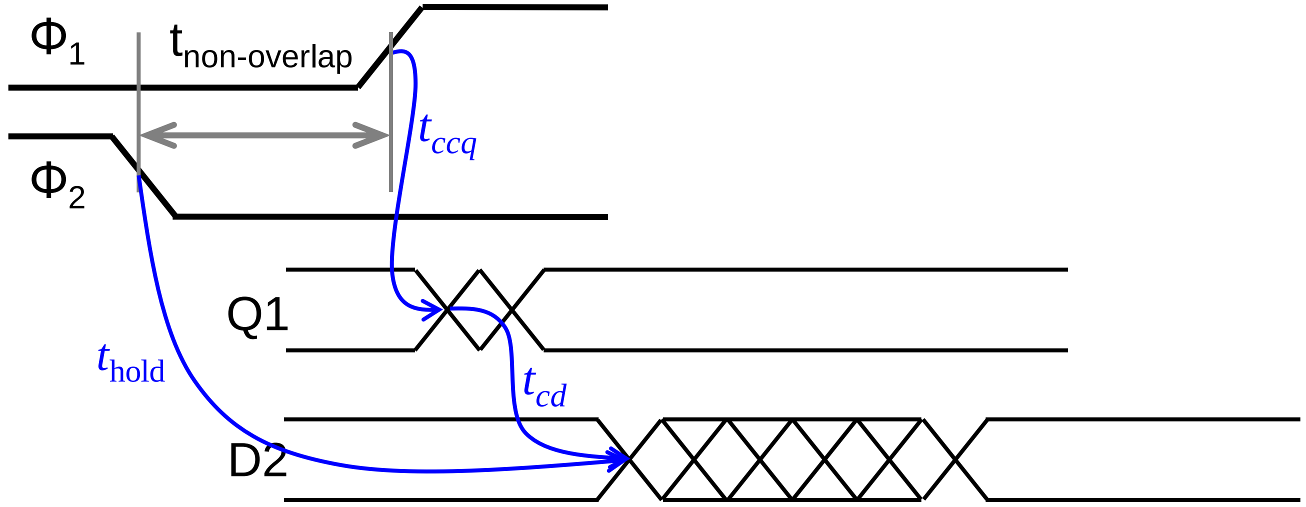


$$t_{ccq} + t_{cd} \geq t_{\text{hold}} \quad \Rightarrow \quad t_{cd} \geq t_{\text{hold}} - t_{ccq}$$



$$t_{\text{non-overlap}} + t_{ccq} + t_{cd} \geq t_{\text{hold}}$$

$$\Rightarrow \quad t_{cd1}, t_{cd2} \geq t_{\text{hold}} - t_{ccq} - t_{\text{non-overlap}}$$

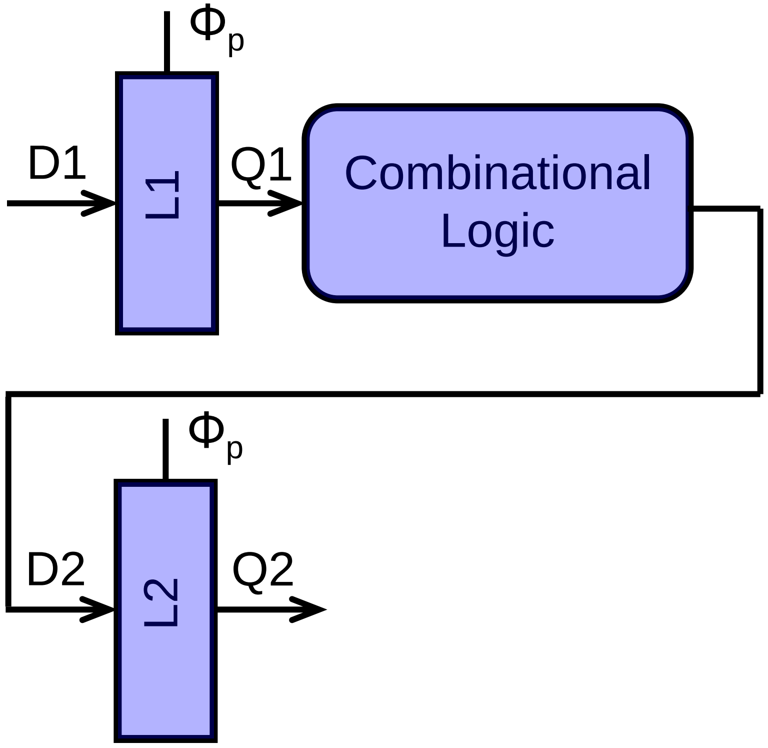


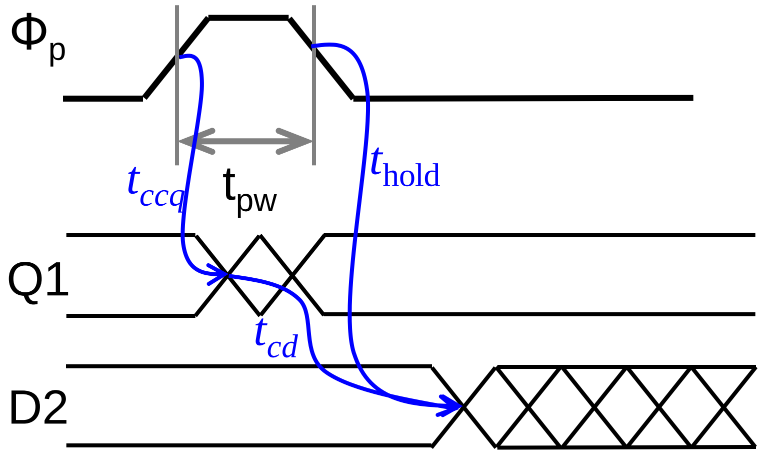
Taking large enough non overlapping time will avoid min delay problems, but distributing two clocks and controlling non overlapping time is difficult and expensive.

Latch-based systems are usually using single clock and its complement, making non overlapping time be zero. In that case min delay constraint for flip-flop and latches is the same.

Here is a paradox: The logic in latch-based system requires twice min-delay as in flip-flop. On the other hand flip-flop can be built by a pair of latches!

The resolution follows from the fact that a flip-flop has an internal race condition, making its hold time longer than in latch.





$$t_{ccq} + t_{cd} - t_{pw} \geq t_{\text{hold}} \quad \Rightarrow \quad t_{cd} \geq t_{\text{hold}} - t_{ccq} + t_{pw}$$

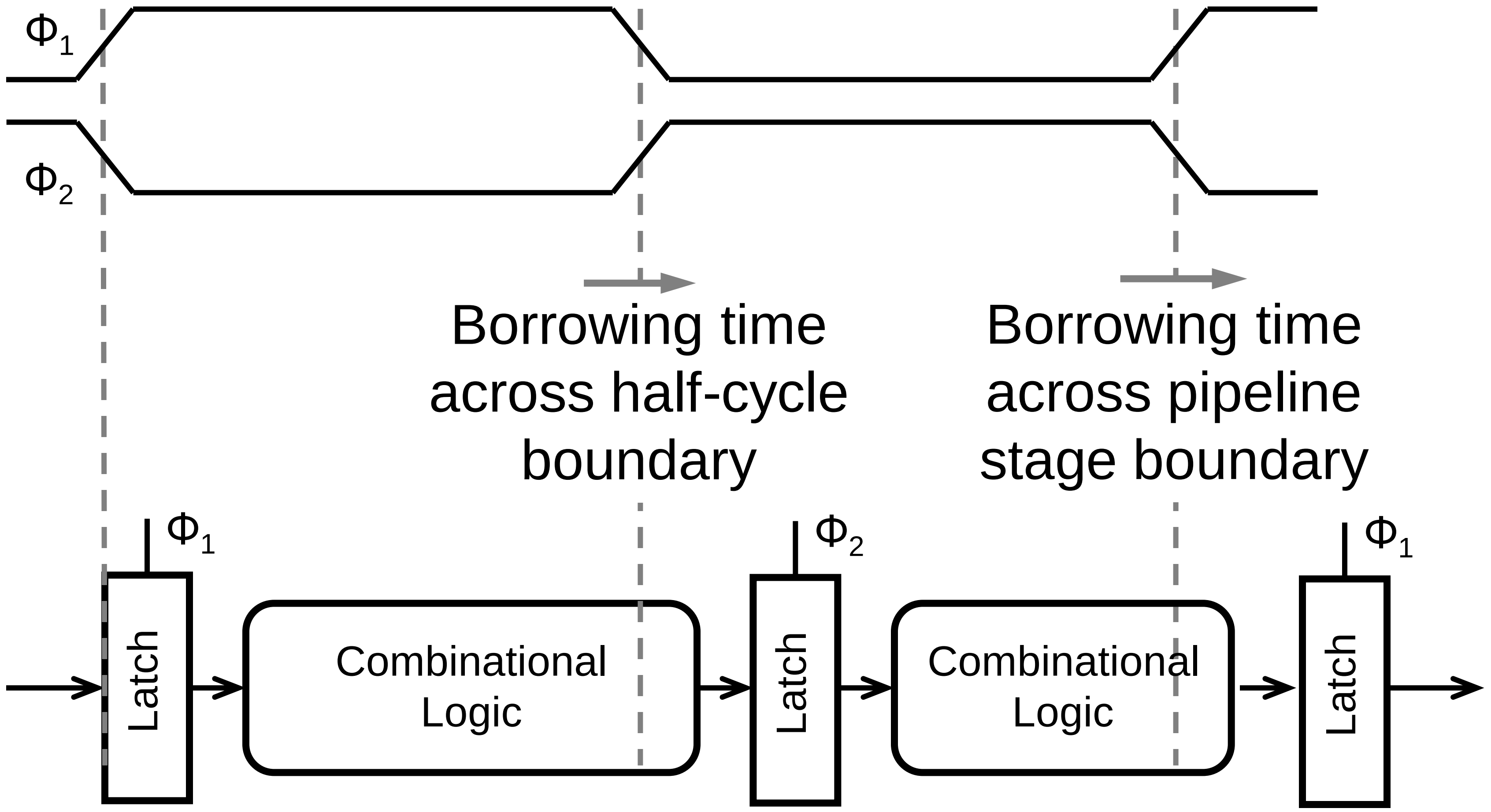
Time Borrowing

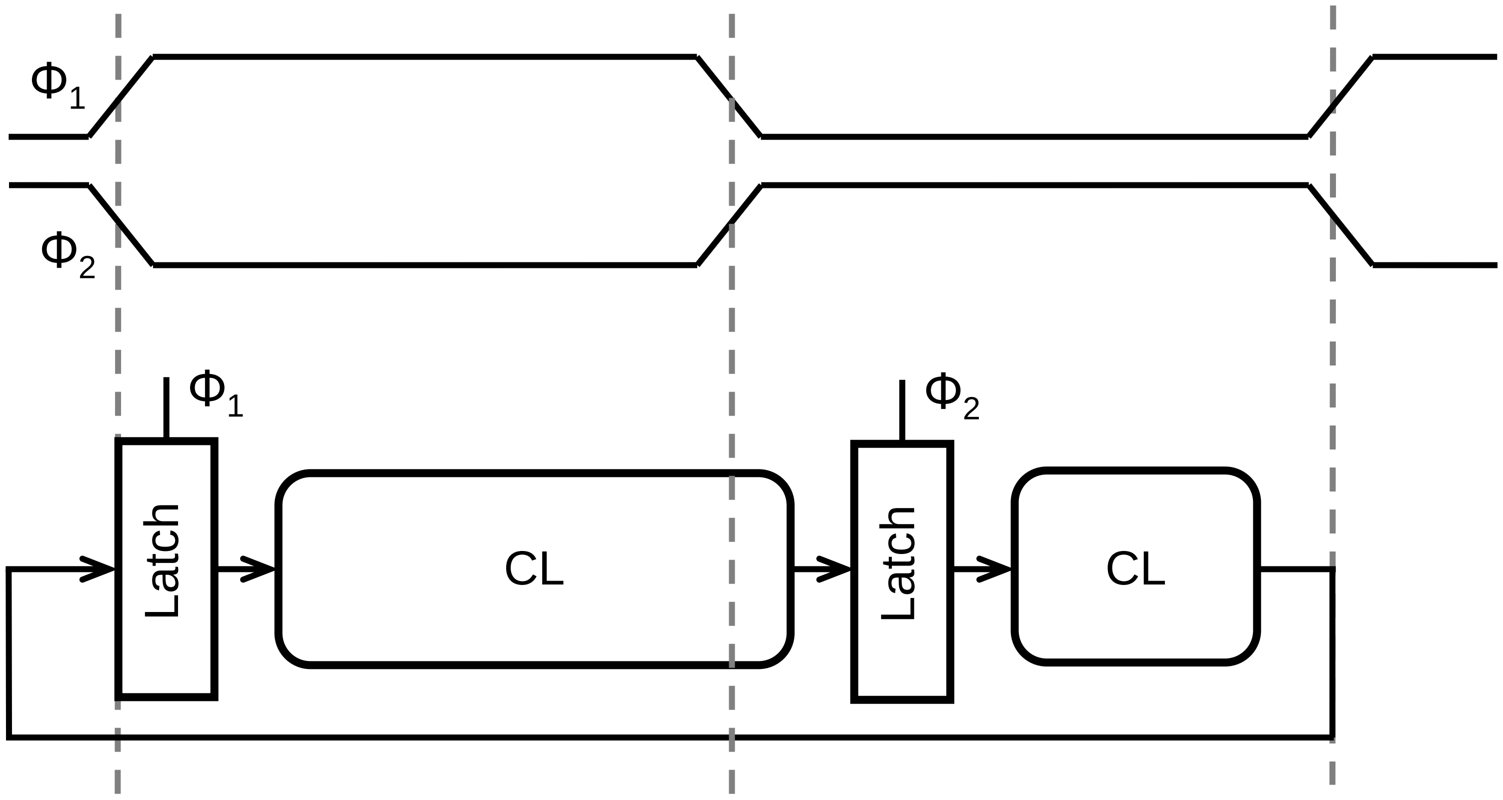
In flip-flop systems clock sharply delineates the cycles.

Hence clock imposes hard edge.

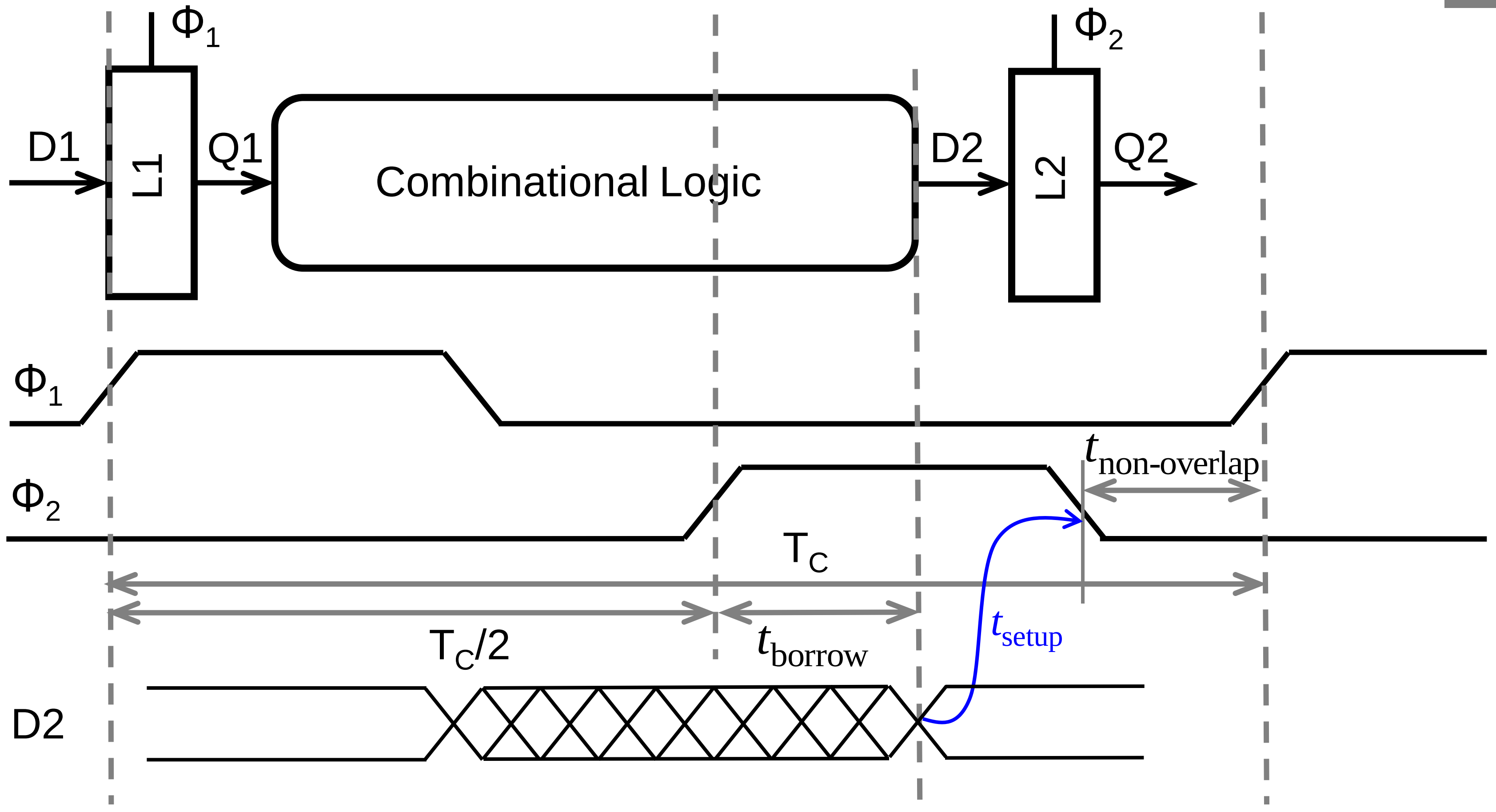
Latch systems are more flexible due to latch transparency.

Data input of a latch must set up before falling edge.


Φ_1
Φ_2
Borrowing time across half-cycle boundary
Borrowing time across pipeline stage boundary
Φ_1
Latch
Combinational Logic
Φ_2
Latch
Combinational Logic
Φ_1
Latch



Loops may borrow time internally but must complete within the cycle.

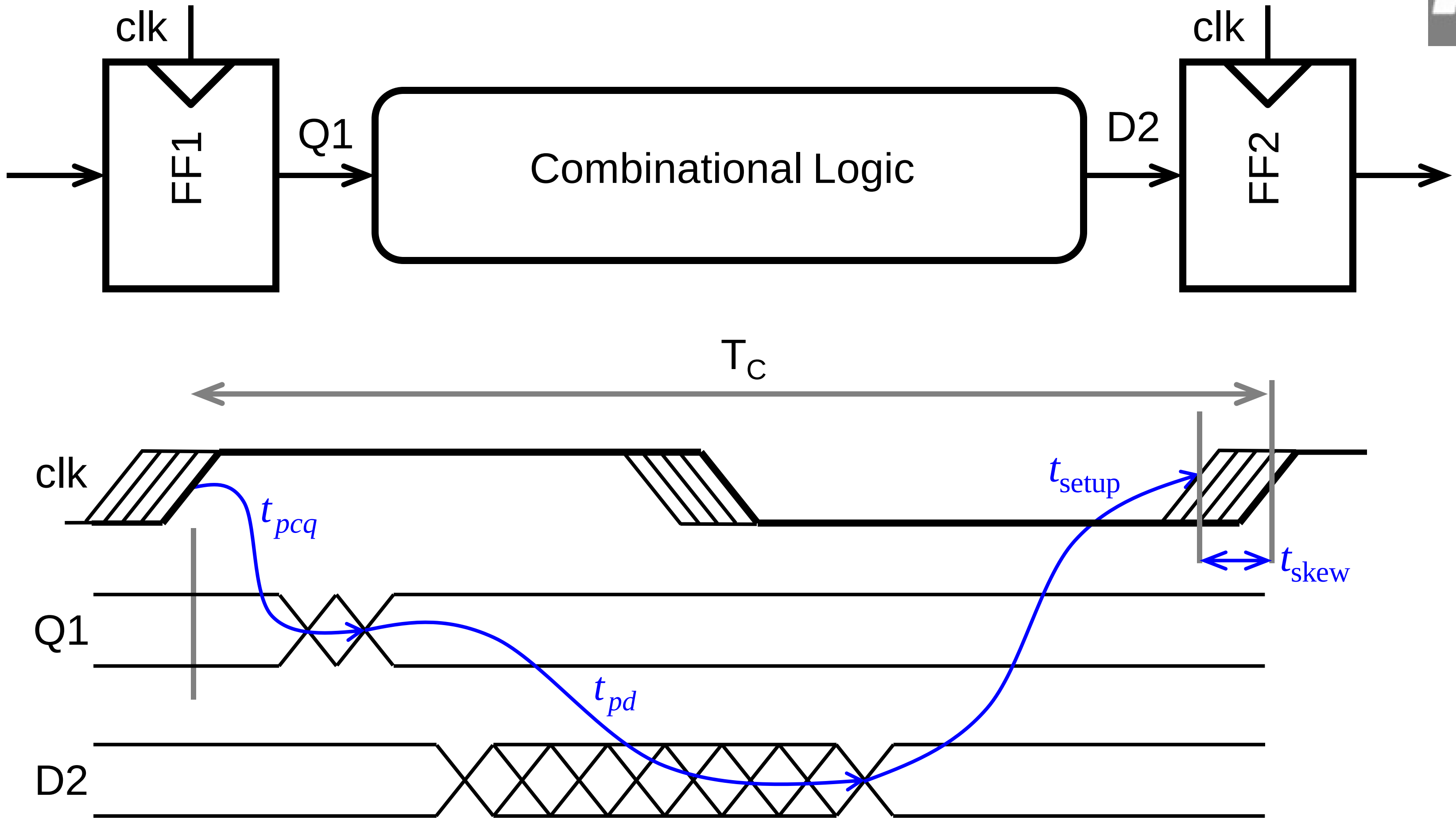


$$T_c \geq T_c/2 + t_{\text{borrow}} + t_{\text{setup}} + t_{\text{non-overlap}}$$

$$\Rightarrow \quad t_{\text{borrow}} \leq T_c/2 - \left(t_{\text{setup}} + t_{\text{non-overlap}}\right)$$

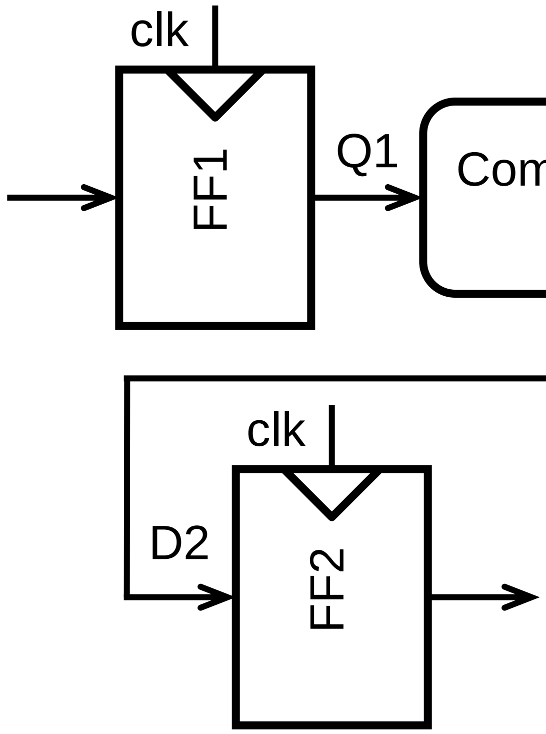
Clock Skew

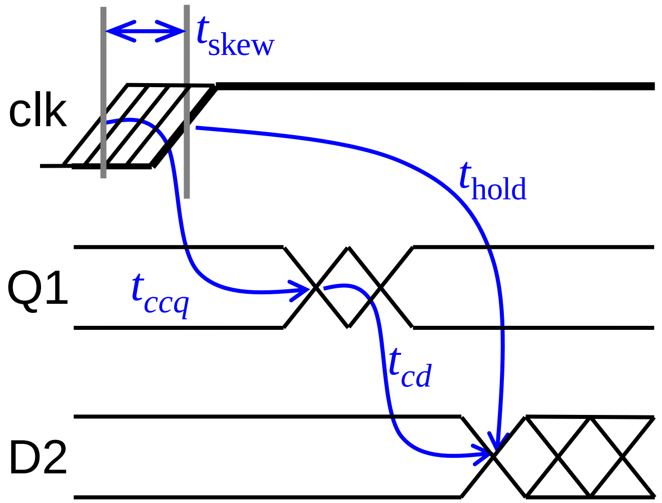
- Clock should theoretically arrive simultaneously to all sequential circuits.
- Practically it arrives in different times. The differences are called ***clock skews***.
- Clock skew consists of the following components:
  - ***Systematic*** is the portion existing under nominal conditions. It can be minimized by appropriate design.
  - ***Random*** is caused by process variations like devices' channel length, oxide thickness, threshold voltage, wire thickness, width and space. It can be measured on silicon and adjusted by delay components.



$$t_{pd} \le T_c - \underbrace{\left(t_{\text{setup}} + t_{pcq} + t_{\text{skew}}\right)}_{\text{sequencing overhead}}$$

Clock skew worsen max and min delay constraints. Max delay constraint becomes shorter. Min delay constraint becomes longer.





$$t_{cd} \geq t_{hold} - t_{ccq} + t_{skew}$$

In **transparent latch** system max delay constraints are not hooked to clock edge since it is assumed that transparency period is long enough. Hence clock skew doesn't affect max delay constrained.

Min delay constraints are worsen however since they depend on the non overlap time, which may effectively be shortened by skew.

$$t_{cd1}, t_{cd2} \geq t_{\text{hold}} - t_{ccq} - t_{\text{non-overlap}} + t_{\text{skew}}$$

Max delay constraint in **pulsed latch** is not affected if pulse is wide enough.

$$t_{pd} \le T_c - \underbrace{\max\left\{t_{pdq}, t_{pcq} + t_{\text{setup}} - t_{pw} + t_{\text{skew}}\right\}}_{\text{sequencing overhead}}$$

Min delay constrained **in pulsed** latch is increased since skew effectively increases hold time.

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

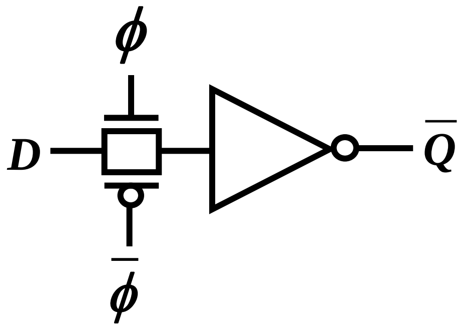
It is also reducing time borrowing.

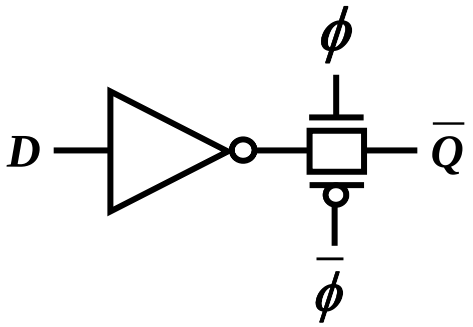
$$t_{\text{borrow}} \le t_{pw} - \left(t_{\text{setup}} + t_{\text{skew}}\right)$$

Latches and Flip-flops

Buffered output inverting latch

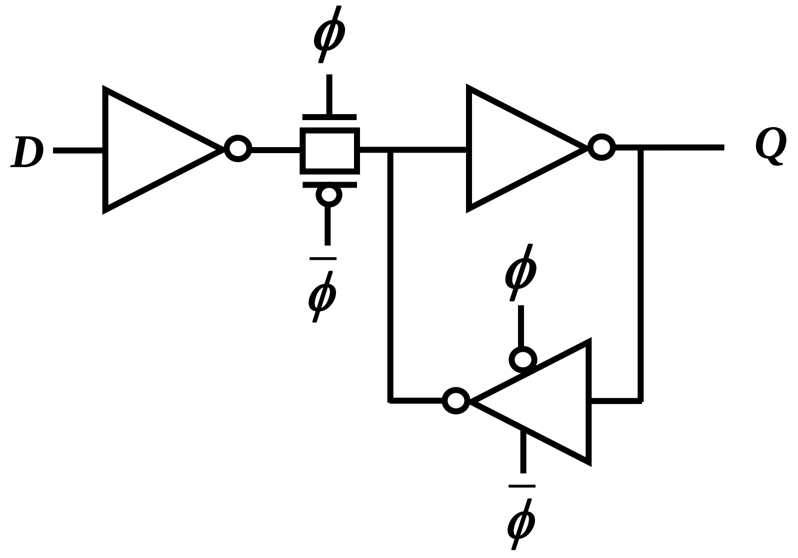
Buffered input inverting latch. It is a tristate inverter.





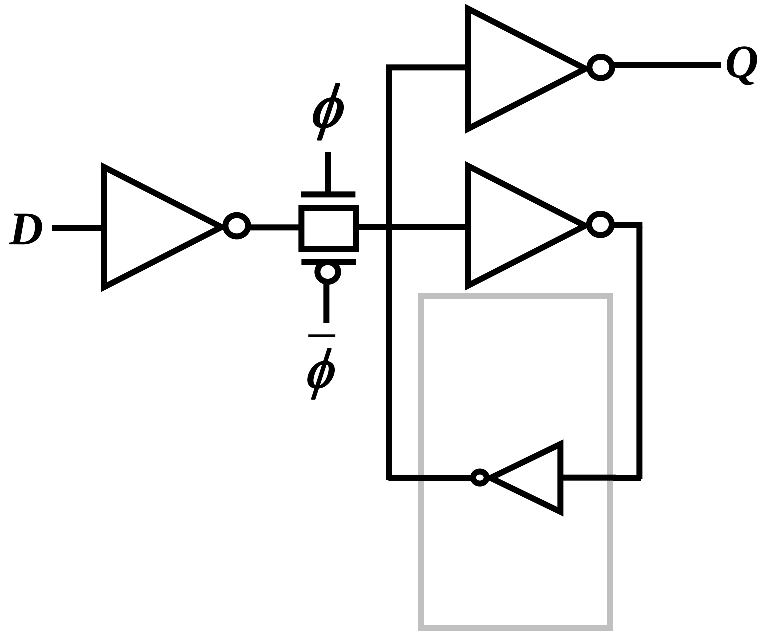
Both are fast dynamic latches.

In high leakage, the dynamic latches retain their output value only for a short period and in order to sustain it, latches must be static to avoid floating output.



Tristate feedback inverter sustains output voltage when clock is low (latch is opaque). Input was also be isolated.

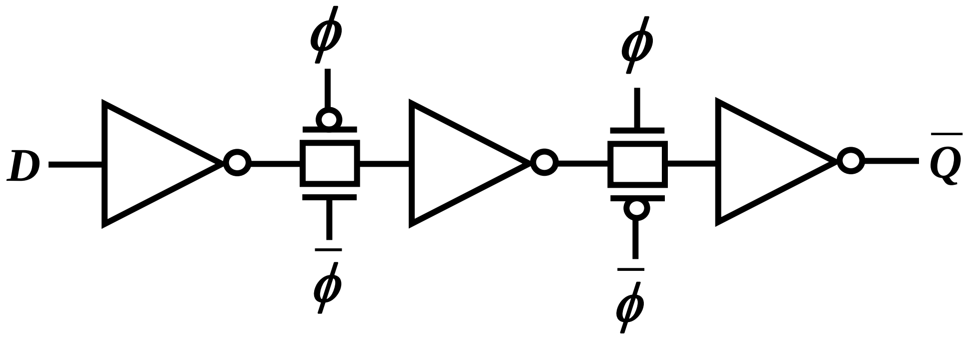
A noise spike at output may invert the latch output.



This is a robust non inverting latch addressing all deficiencies

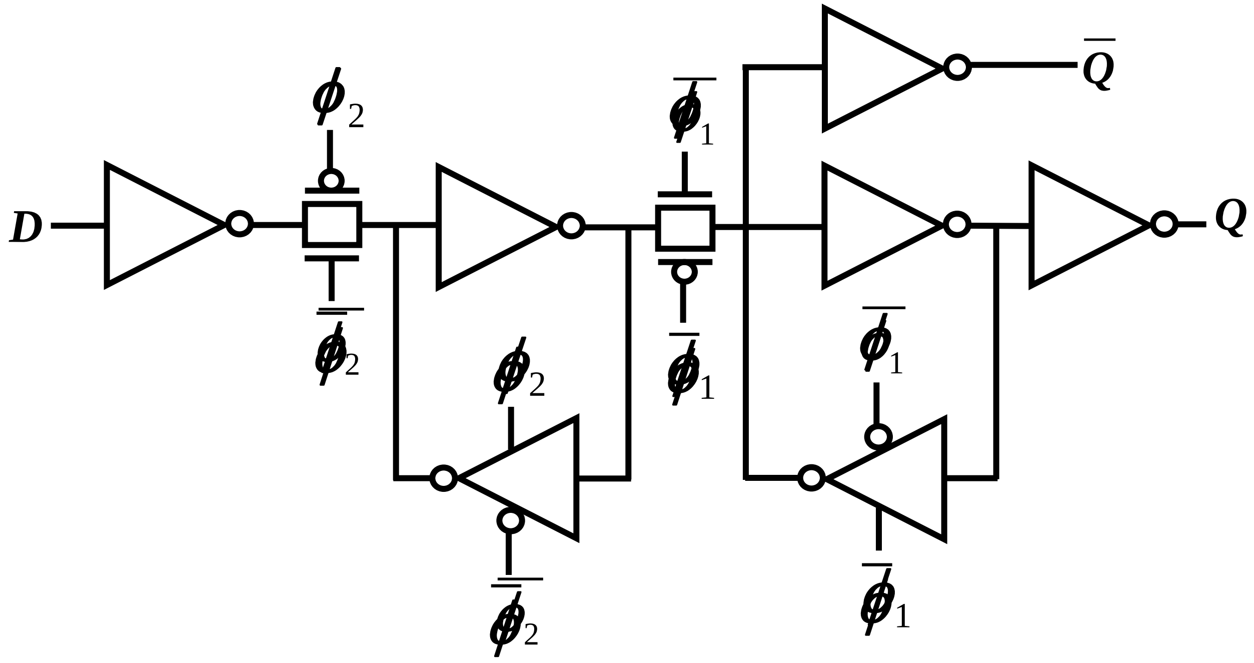
To reduced clock load and save two transistors, the tristate can be replaced by a weak inverter, called ***jamb latch***.

Dynamic inverting flip-flop can be constructed by a pair of back-to-back dynamic latches.



To reduce delay at the expense of noise sensitivity, either first or last inverters can be removed.

Static non inverting flip-flop is constructed from two static latches.



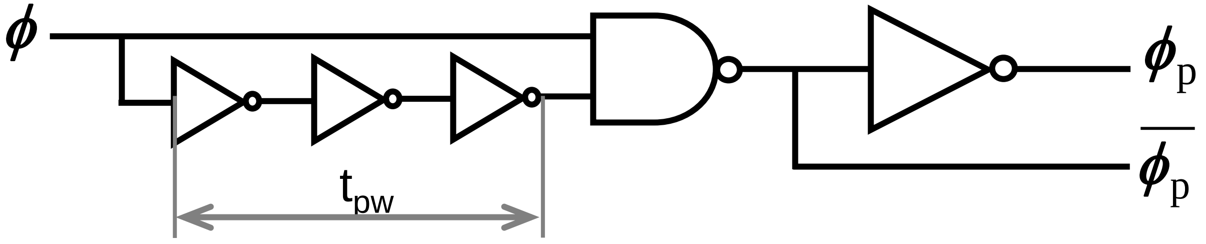
For very slow rise / fall time both latches will be simultaneously transparent, which will require to increase FF’s hold time.

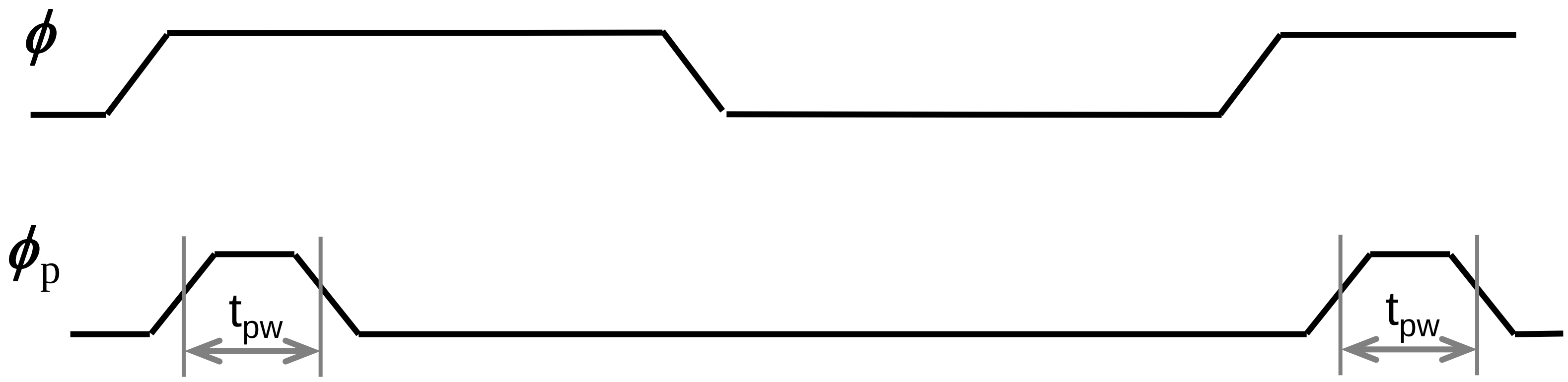
To sharpen clock edges local clock buffers can be used. The expense is more area and clock load (power).

Non-overlapping clocks avoid simultaneous transparency. Making it large enough, large skew can also be tolerated.

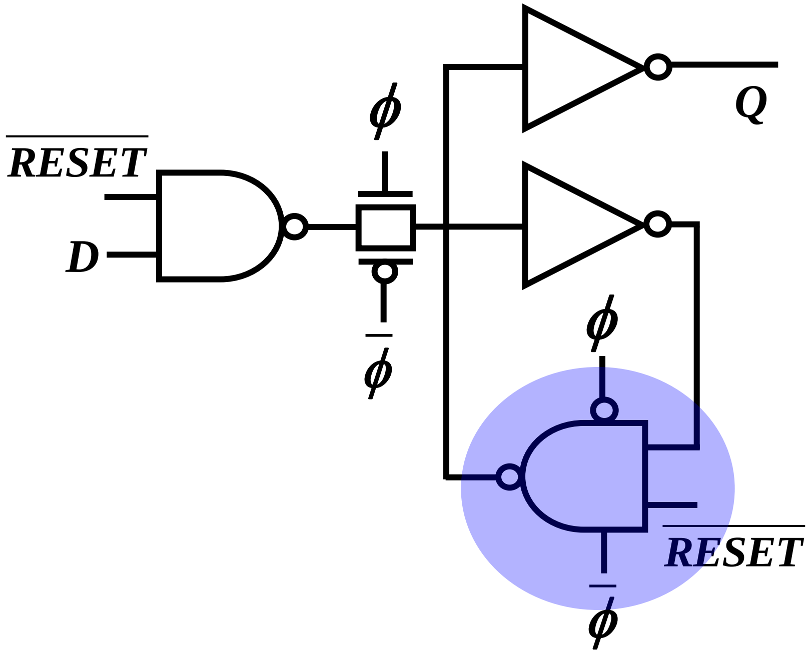
Large non-overlapping causes however large setup time (sequencing overhead).

Pulsed latch is built from conventional latch driven by a brief clock pulse, generated by a ***clock chopper***.





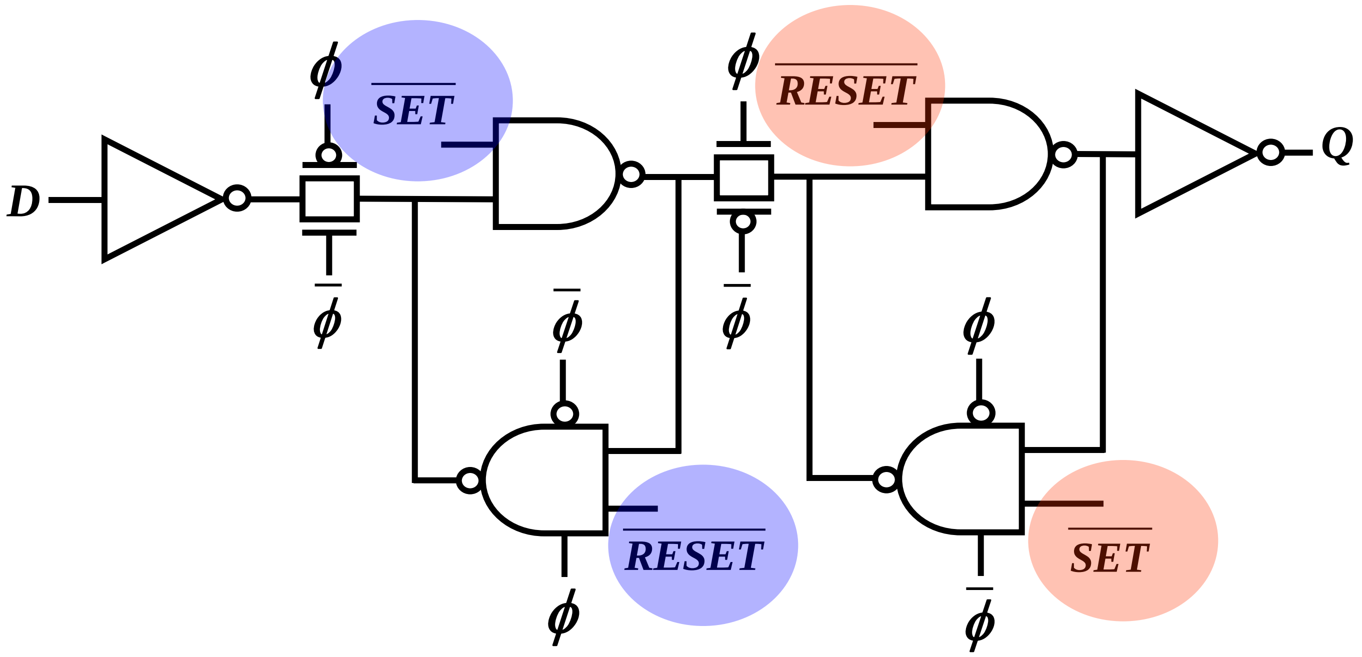
Sequencing elements require a **reset** signal to enter a known initial state or startup. Asynchronous reset forces Q immediately, while synchronous reset waits for clock edge.



Asynchronous resettable latch

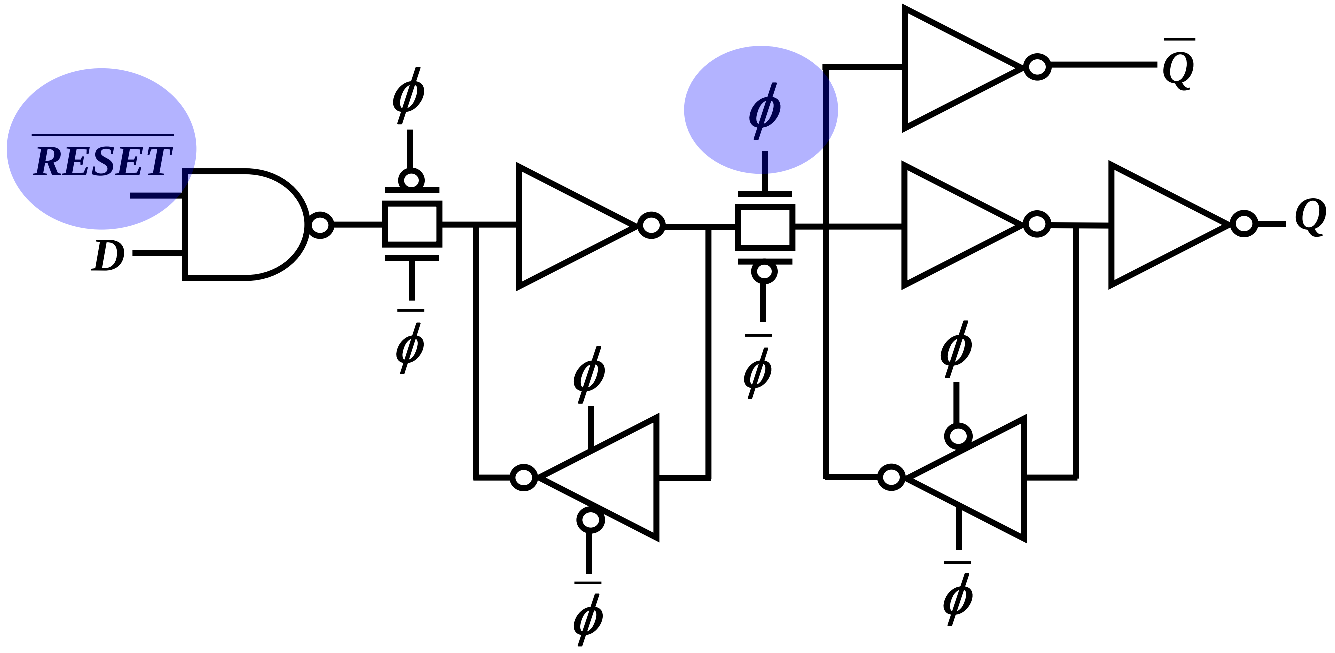
Asynchronous settable and resettable flip-flop

Set and reset must enter in both master and slave stages



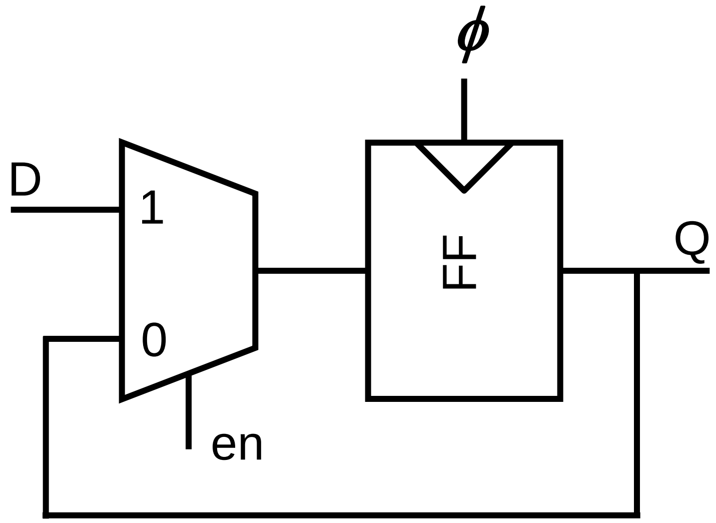
Synchronous resettable flip-flop

Reset affects Q only at clock rising



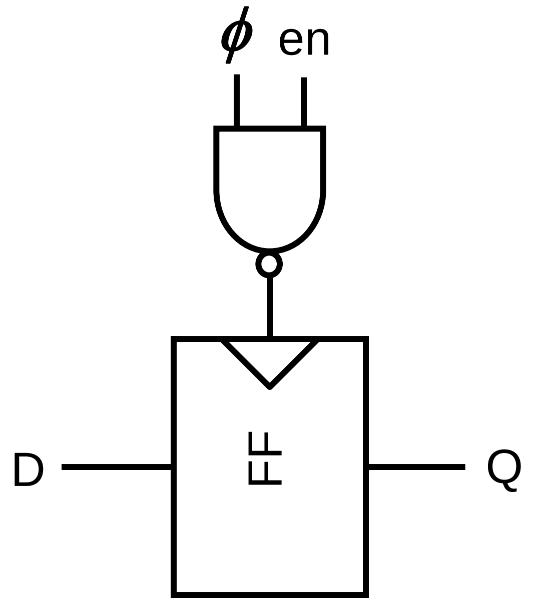
Reset must be stable for setup and hold time around clock edge.

Enabling latches and flip-flops



Enabling is done on data change. The savings is mostly in suppressing the D generation, known to be unchanged.

It doesn’t affect clock, but affects delay, adds area and power.



Clock gating doesn’t affect delay but may add clock skew. The savings is on the CLK network and FFs.

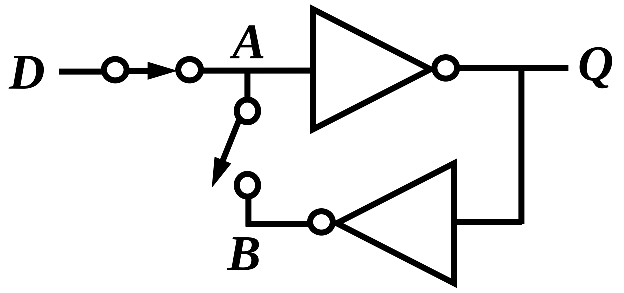
It significantly reduces power consumption since clock is not toggling on disabled element. AND gate is added to clock driver.

Static Sequencing Methodology

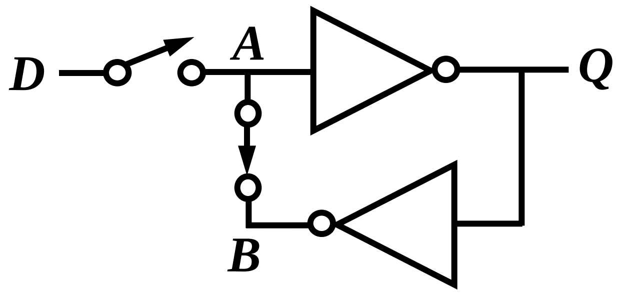
- **Flip-flops:** Most popular in non aggressive deigns.
  - Simple and robust.
  - Setup and hold times are penalties.
  - No time borrowing across clock cycle (unless clock is shifted).
- **Pulsed Latches:** Similar behavior to flip-flops.
  - Simpler than FF or transparent latches, less area and power.
  - Faster than FF.
  - Higher effective hold time, min-delay constraints more difficult.

- **Transparent Latches:** Use in high-end designs
  - Lower sequencing overhead, faster than FF.
  - Allows nearly half cycle time borrowing.
  - Complex clock design, sensitive to clock slew rate

Metastability in a latch

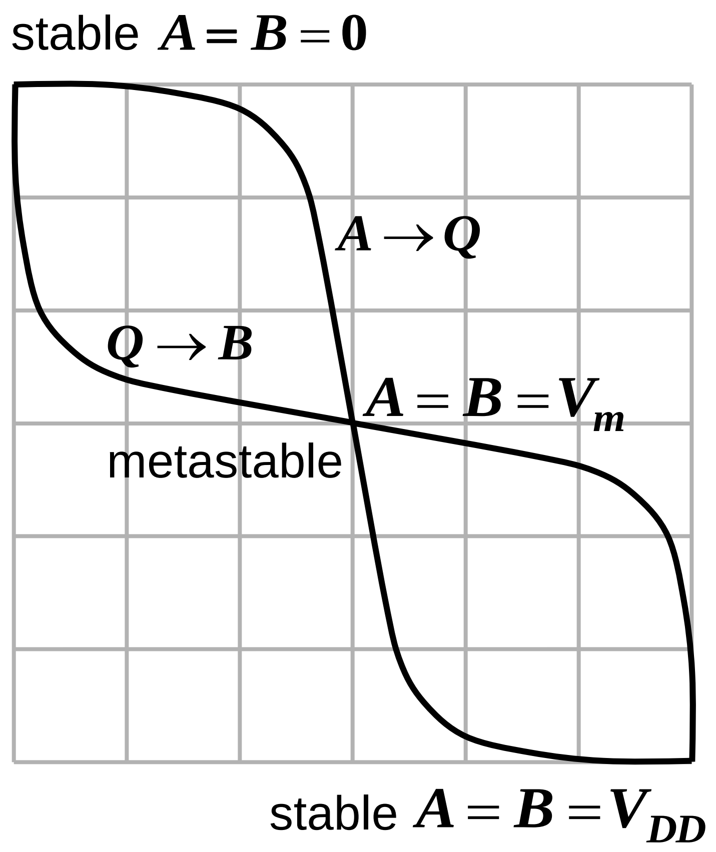


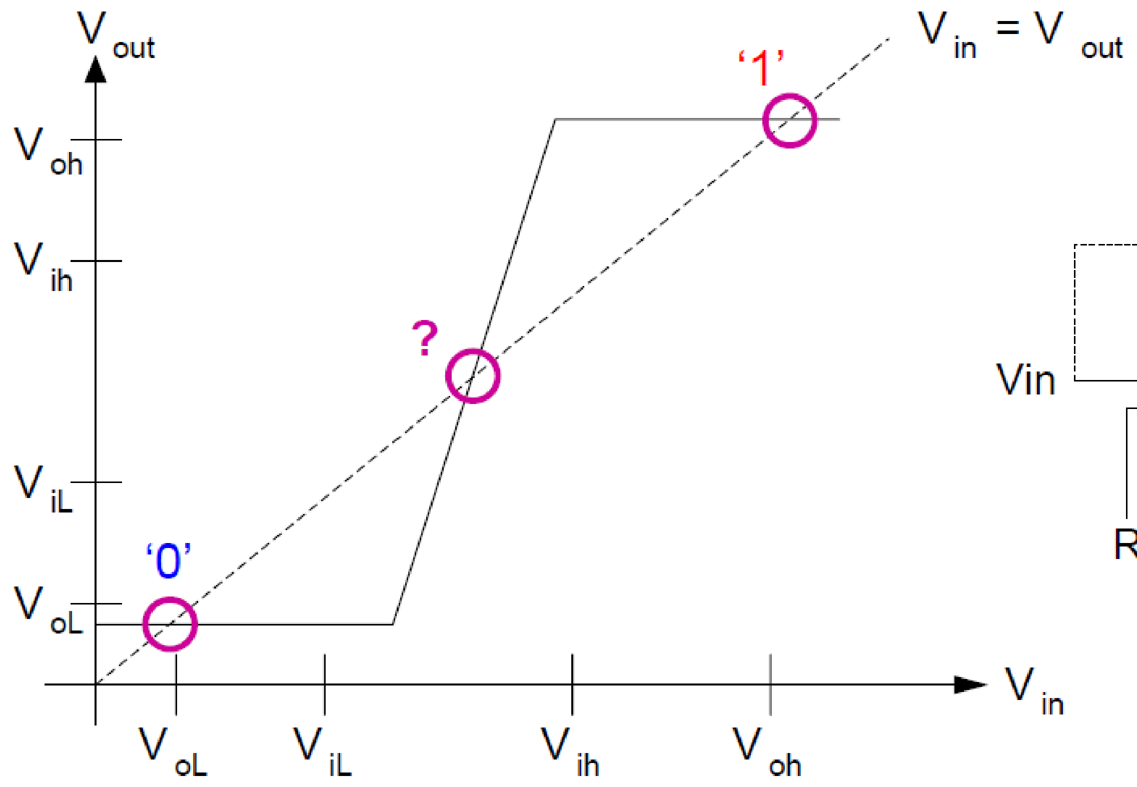
At transparency sample switch is closed and hold switch is open.



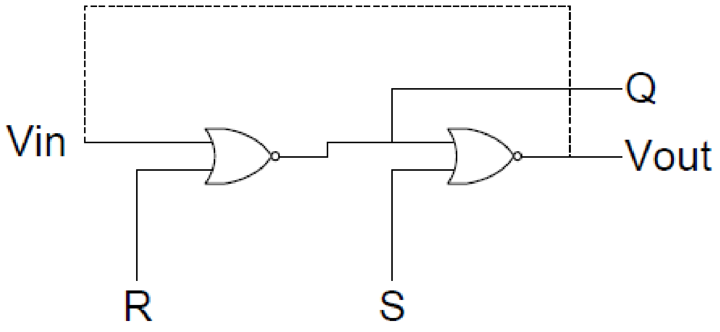
When latch is opaque sample switch is open and hold switch is closed, and the two inverters are connected in a feedback loop.

Latch can enter into metastable state where voltages are consistent. It can remain so unbounded time!

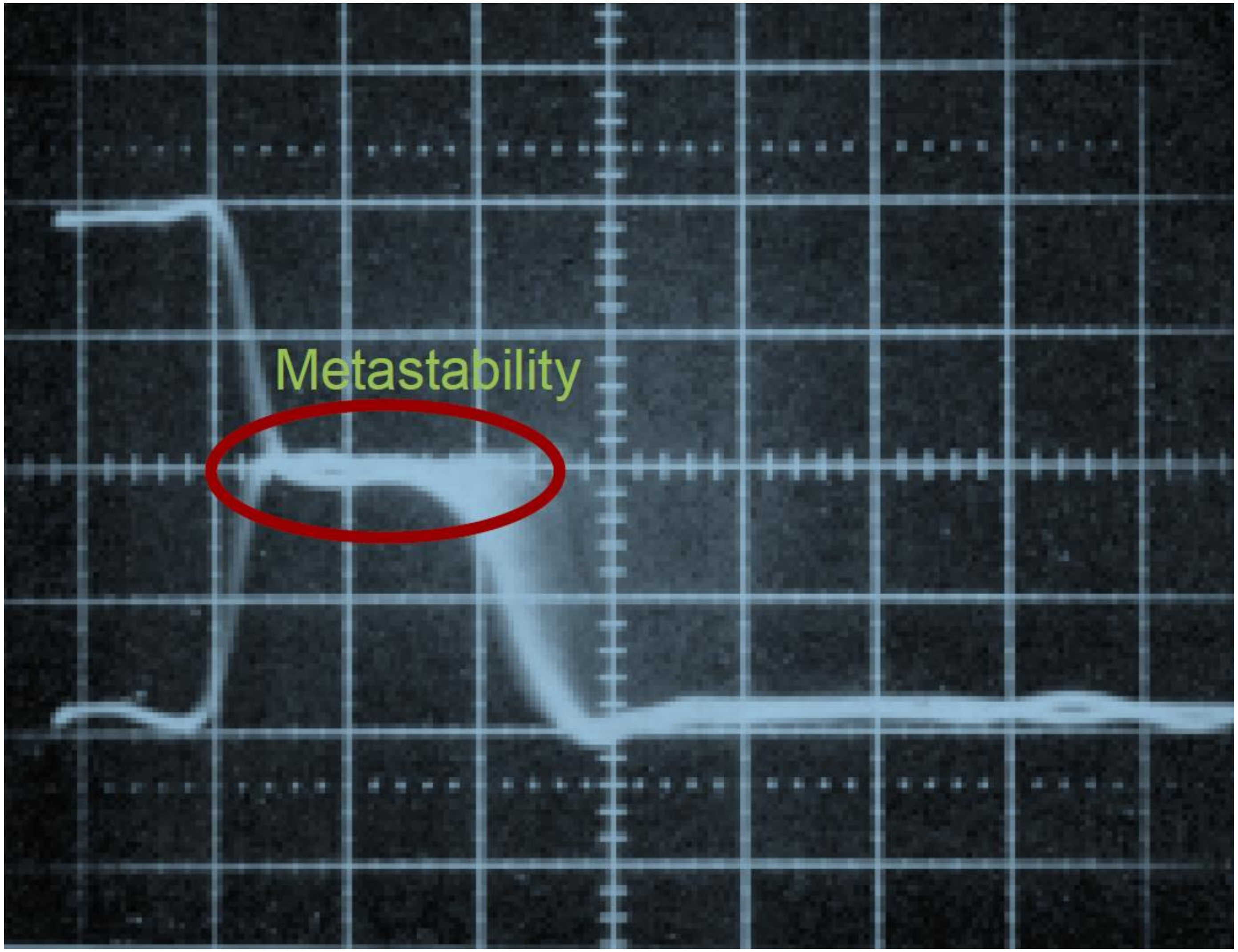




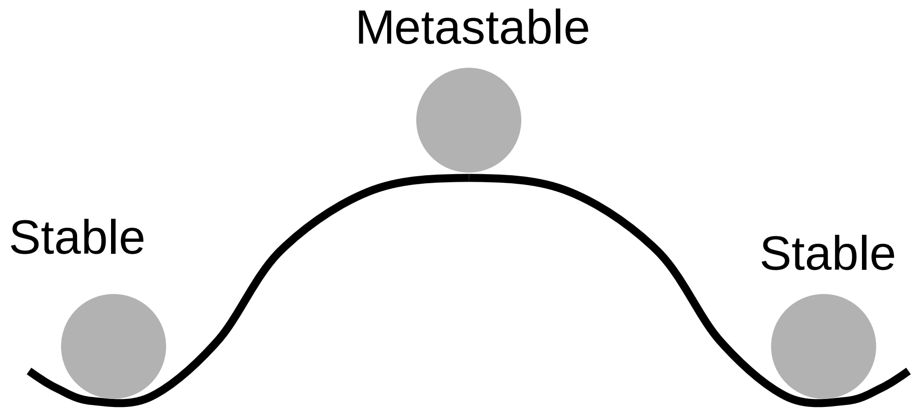
V_{out}
$V_{in} = V_{out}$
'1'
?
'0'
V_{oh}
V_{ih}
V_{iL}
V_{oL}
V_{in}
V_{oL}
V_{iL}
V_{ih}
V_{oh}



Vin
Q
Vout
R
S


Metastability

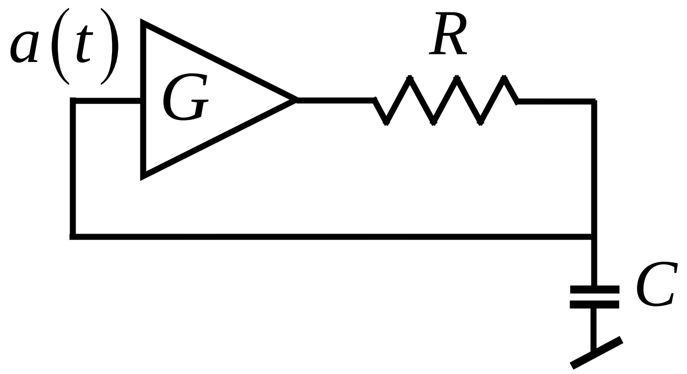
Though the latch can stay metastable infinite time, the probability of remaining metastable drops off exponentially with time.



Any noise or other disturbance will cause ***A*** and ***B*** node to switch into one of the stable states.

When ***A*** is near the metastable voltage ***V_m*** the cross-coupled inverters behave like a linear amplifier with gain ***G***.

Inverter delay can be modeled by output resistance ***R*** and load capacitance ***C***.



Small signal model of bistable element in metastability

To predict metastability behavior, let latch turn opaque at $t=0$. The voltage $A(0)$ at point A is $A(0)=V_m+a(0)$, where $a(0)$ is a small signal offset.

The current throug R is charging C,

$$\frac{Ga(t)-a(t)}{R}=C\frac{da(t)}{dt},$$

which solves to $a(t)=a(0)\exp\left[t(G-1)/RC\right]$.

If the node is defined to reach legal logic level

when $\left|a(t)\right| > \Delta V,$ the time to reach this level is

$t_{DQ} = \left[RC/(G-1) \right] \left[\ln \Delta V - \ln a(0) \right].$

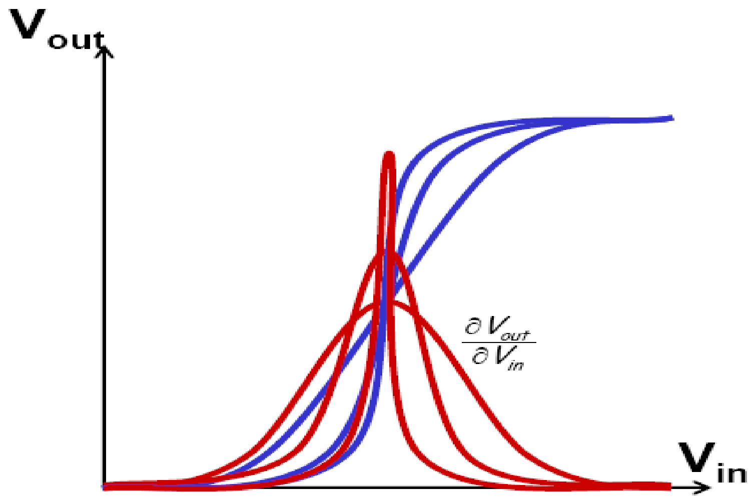
Latch propagation delay may reach infinity

if $a(0) = 0,$ which cannot happen in reality

because of noise. There's however no upper

bound for the time until output becomes valid.

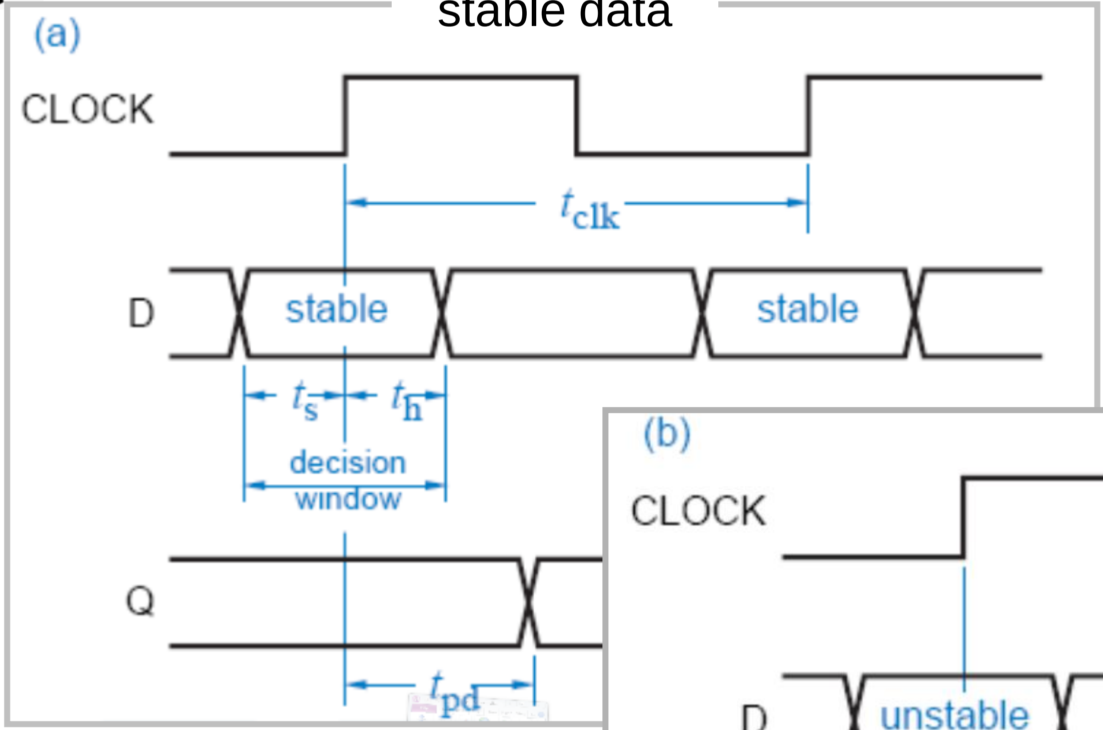
To quickly recover from metastability the term $(G-1)/RC$ should be sufficiently large.



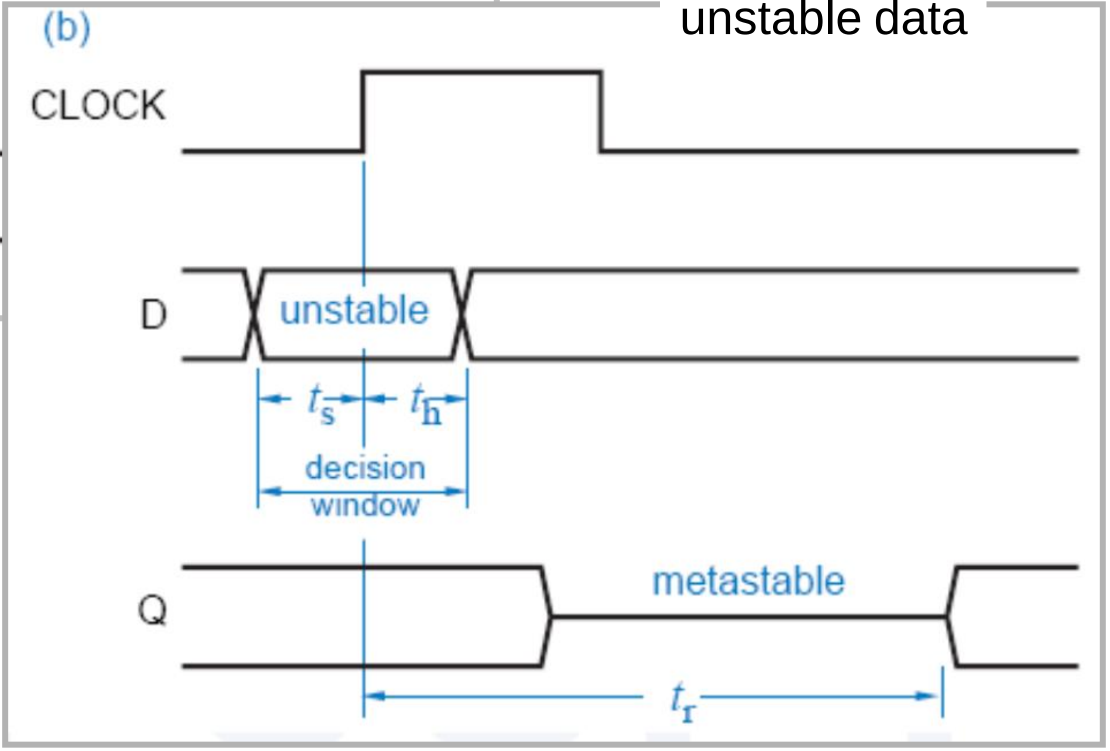
Latch propagation delay t_{DQ} is the time until its output becomes valid. The probability it will be longer than t' is $P\left(t_{DQ} > t'\right) = \frac{T_0}{T_C}\exp\left[\frac{-t'(G-1)}{RC}\right]$.

T_0 and $(G-1)/RC$ are obtained by measurements or simulations.

stable data



unstable data

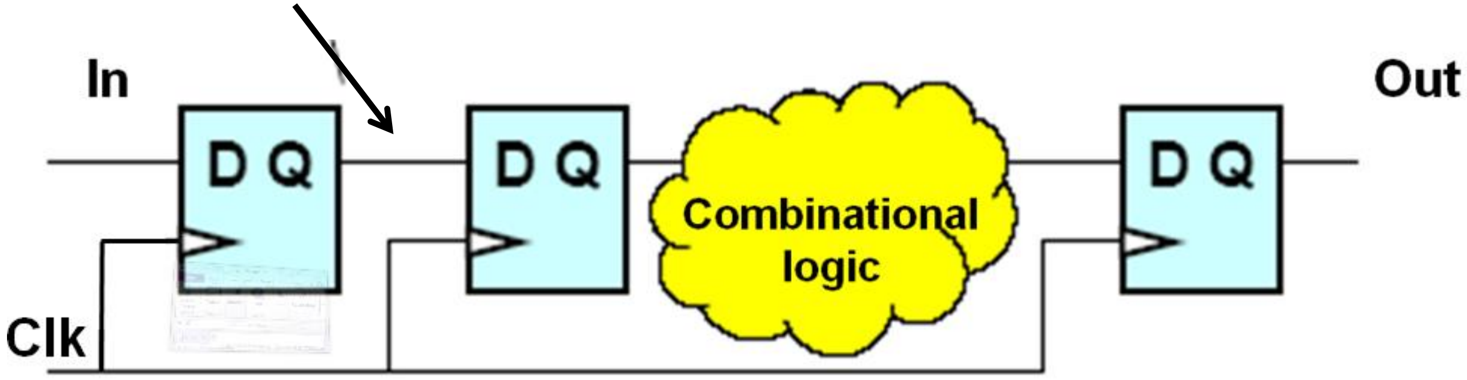


Synchronizers and Metastability

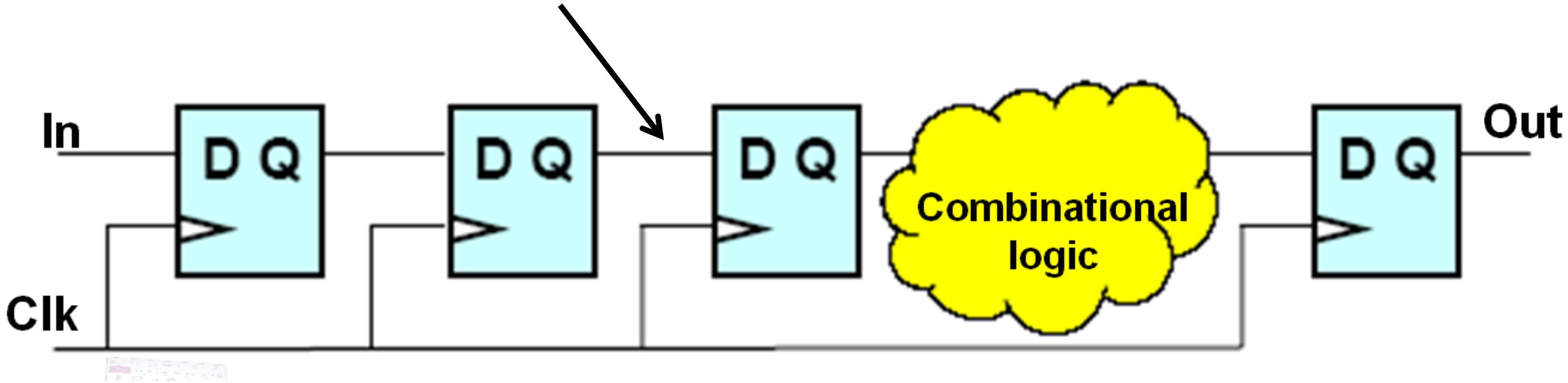
- Proper operation of synchronous circuits requires that data is stable around the clock rising edge.
- Connection to an external input may not satisfy it.
  - Input devices like keyboard or mouse are blind to internal system.
  - Two systems of different clocks may feed each other.
- What happens when data is changed in the aperture between setup and hold times?
  - Output may be unpredictable and the time for settling to a good logic level may be unbounded. This is called ***metastability***.

- A synchronizer is a circuit that accept an input that can change arbitrarily and produce an output aligned to the synchronizer's clock.
- Because the input can change arbitrarily in the synchronizer's aperture, there is non zero probability that synchronizer's output is still ***metastable***.
- Synchronizers are built to make this probability sufficiently small.
  - It is measured by mean time between failures (MTBF), which can be set arbitrarily long.

A metastable state here will probably resolve itself to a valid level before it gets into my circuit.



And one here will almost certainly get resolved.


In
D Q
D Q
D Q
Combinational logic
D Q
Out
Clk