25 February 2021 وَمَا أُوتِيتُمْ مِنَ الْعِلْمِ إِلَّا قَلِيلًا 13 رجب 1442

Digital IC Design

Lecture 19
Metastability and Synchronizers

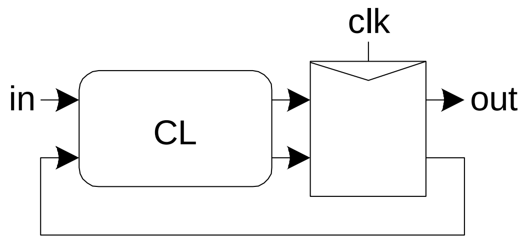
Dr. Hesham A. Omran

Integrated Circuits Laboratory (ICL)
Electronics and Communications Eng. Dept.
Faculty of Engineering
Ain Shams University

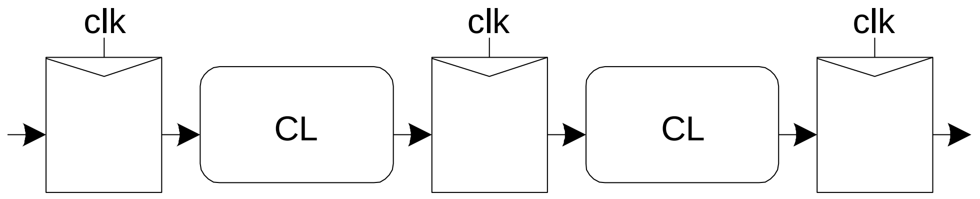
This lecture is mainly based on “CMOS VLSI Design”, 4th edition, by N. Weste and D. Harris and its accompanying lecture notes

Combinational vs. Sequential Logic

- Combinational logic
  - Output depends on current inputs
- Sequential logic
  - Output depends on current and previous inputs
  - Uses flip-flops or latches (memory/sequencing elements)
  - Separates previous, current, and next states (tokens)
  - Ex: FSM, pipeline



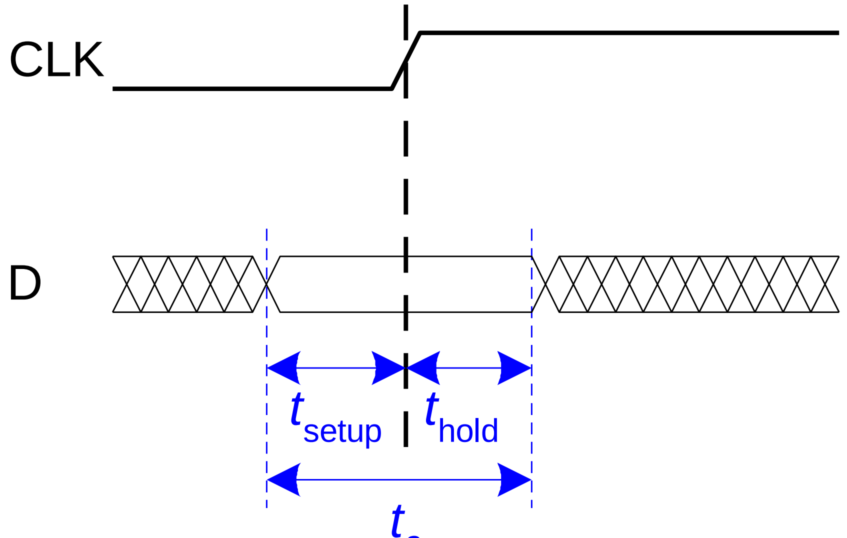
Finite State Machine



Pipeline

FF Input Timing Constraints

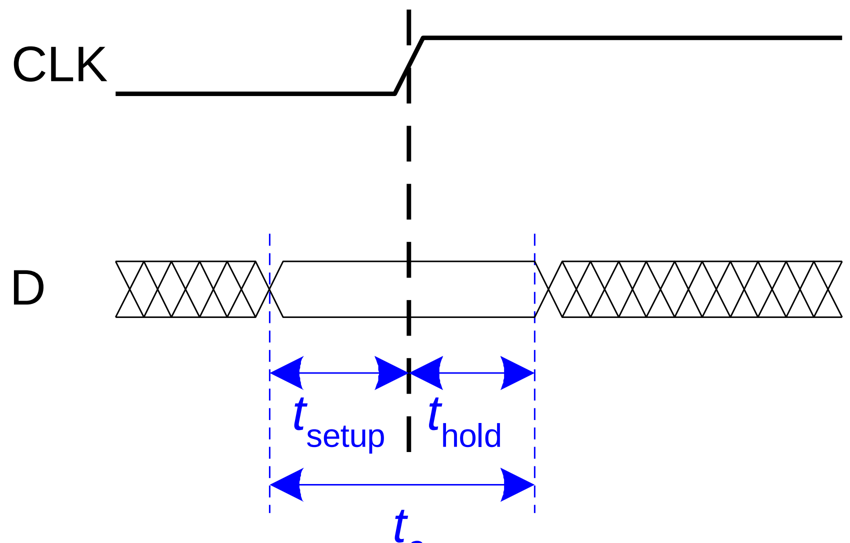
- Flip-flop samples *D* at clock edge
  - *D* must be stable when sampled
  - Similar to a photograph, *D* must be stable around clock edge
  - If not, metastability can occur
- **Setup time:** t_{setup} = time *before* clock edge data must be stable (i.e. not changing)
- **Hold time:** t_{hold} = time *after* clock edge data must be stable
- **Aperture time:** t_a = time *around* clock edge data must be stable ($t_a = t_{setup} + t_{hold}$)



[Harris & Harris, DDCA, 2012]

FF Input Timing Constraints

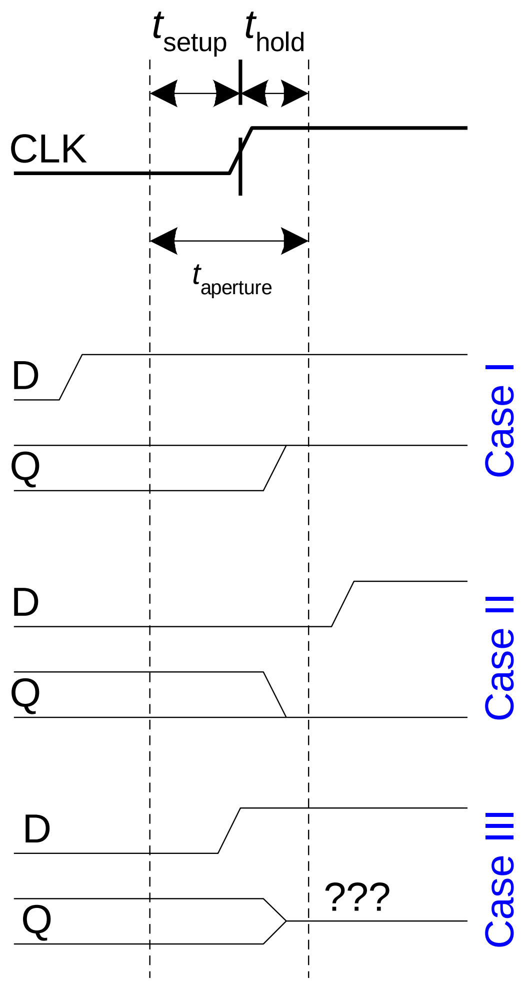
- If data changes in the aperture between the setup and hold times
  - Latch enters a metastable state
  - Output may be unpredictable
  - Time to settle to a good logic level may be unbounded

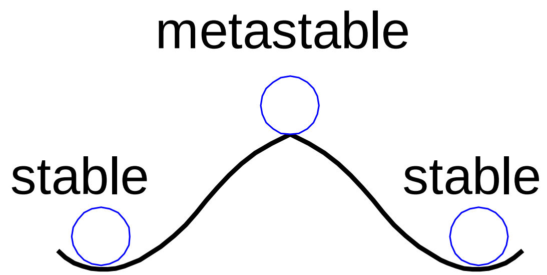


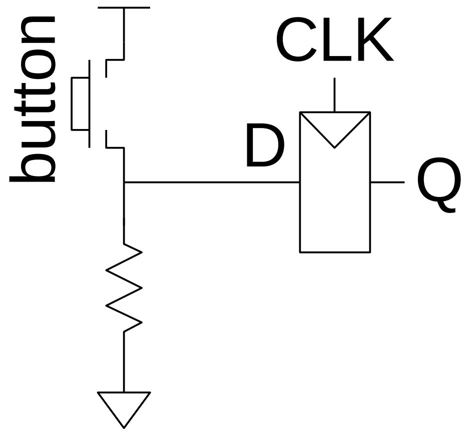
[Harris & Harris, DDCA, 2012]

Violating the Dynamic Discipline

- Asynchronous inputs may violate the dynamic discipline
  - Asynchronous inputs are inevitable (user inputs, MCU interrupts, systems with different clocks interacting, etc.)
  - Multiple clock domains exist on the same chip: on-chip data crosses the clock domain boundaries.
- Flip-flop is a bistable device
  - Two stable states (1 and 0) and one metastable state
  - If flip-flop lands in metastable state, could stay there for an undetermined amount of time



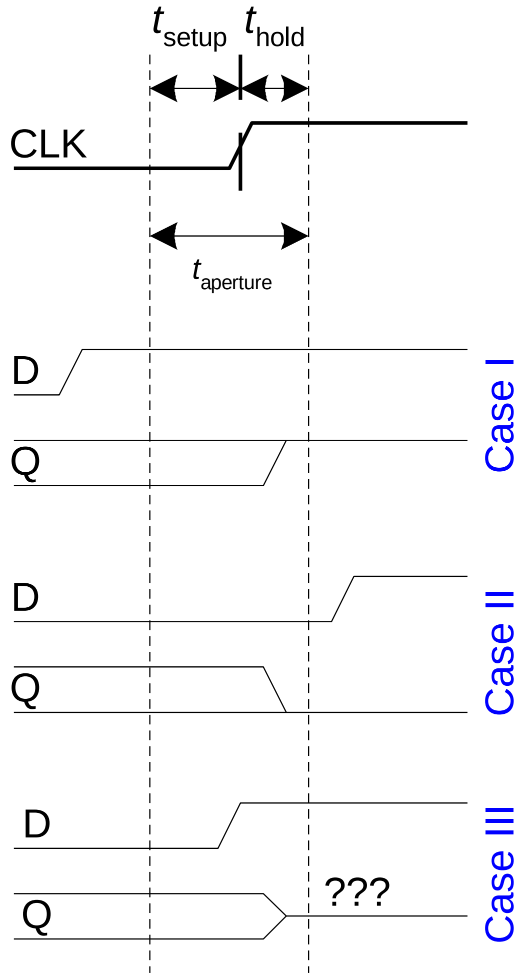


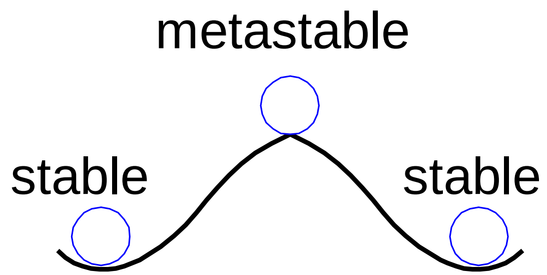


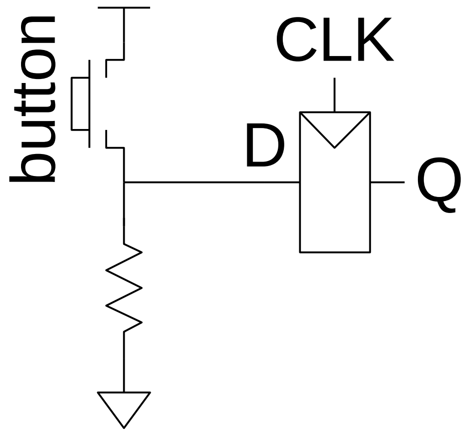
[Harris & Harris, DDCA, 2012]

Synchronizers

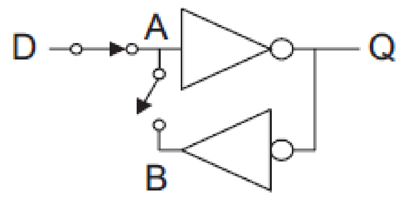
- Unsynchronized inputs can cause strange and sporadic system failures that are very difficult to locate
- Synchronizer: a circuit that accepts an input that can change at arbitrary times and produces an output aligned to the synchronizer's clock
- Synchronizer goal: make the probability of failure (the output Q still being metastable) **low**
  - **Synchronizer cannot make the probability of failure 0**



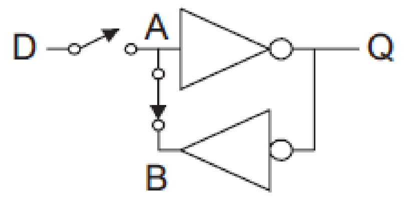




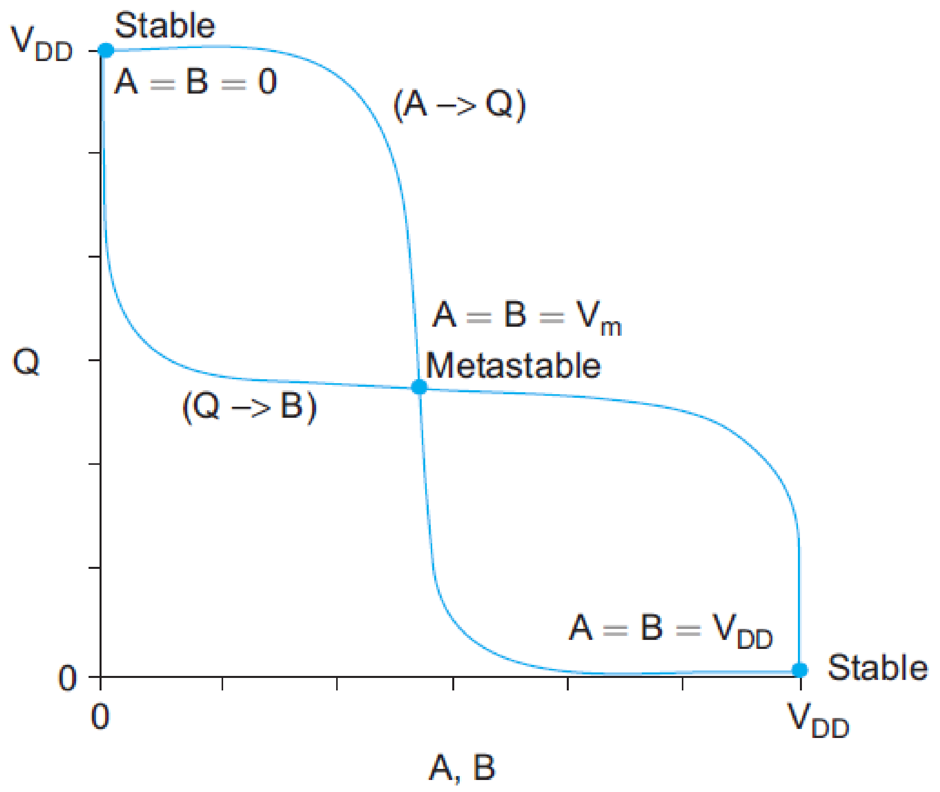
Metastability



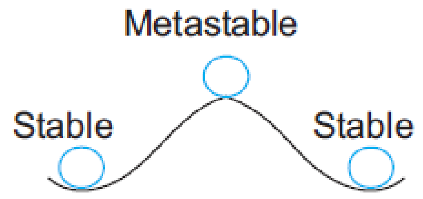
(a)



(b)

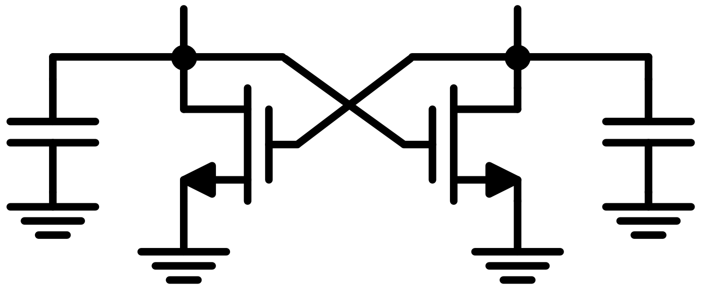


(c)



(d)

Regenerative Latch Delay



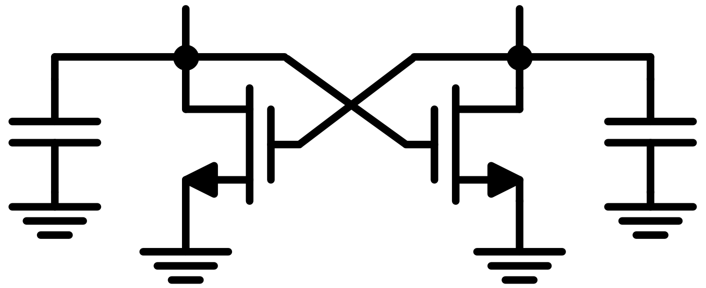
Regenerative Latch Delay

- Assume the pre-amplified latch input voltage $= \Delta V_{in}$

$$\tau \approx \frac{C}{g_m}$$

$$\Delta V_{out}\,(t) = \Delta V_{in} e^{+t/\tau}$$

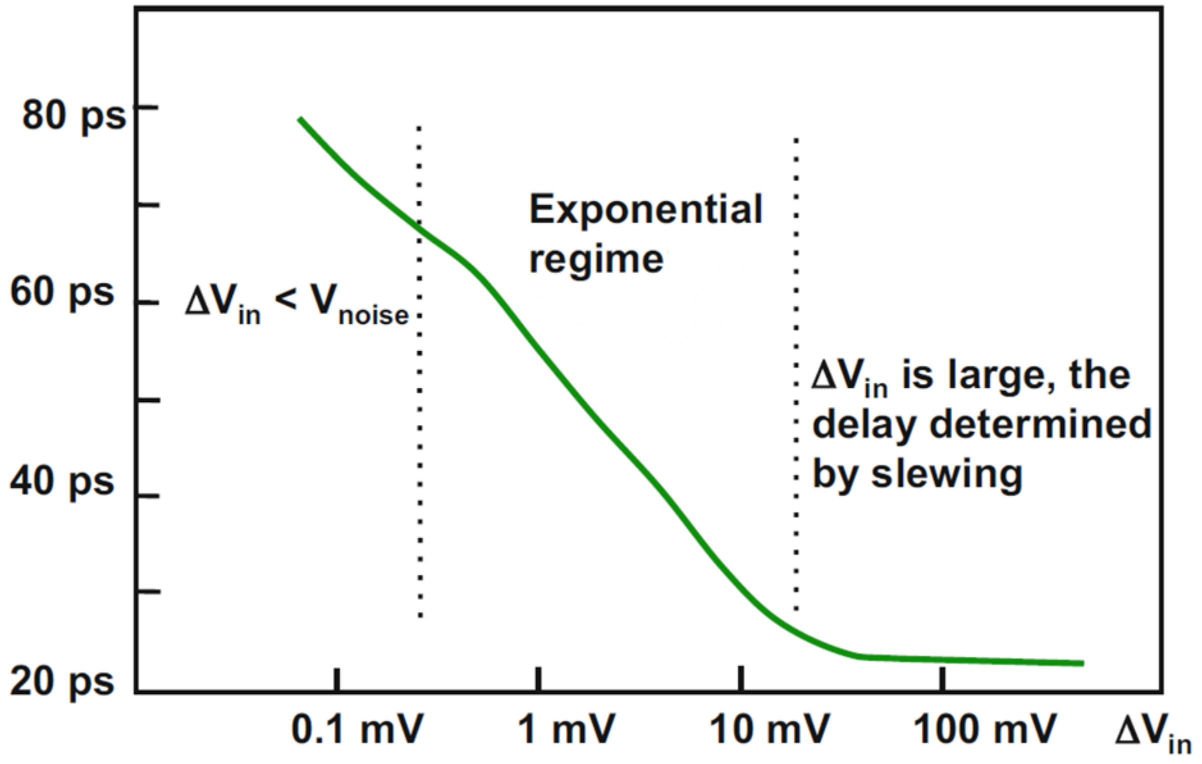
- The exponential signal growth makes a latch a fast decision element
- The exponential growth will continue until some non-linear limiting mechanism takes action (e.g., approaching power rails)



Regenerative Latch Delay

- The time a latch needs to form a digital signal depends on the initial over-drive voltage
  - The input differential voltage (ΔV_{in}) determines the delay of the latch

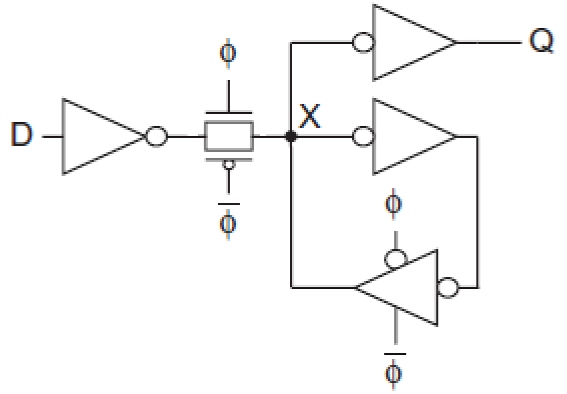
$$\Delta V_{out}(t) = \Delta V_{in} e^{+t/\tau} \rightarrow \; t_d = \tau \ln \frac{V_{DD}}{\Delta V_{in}} = \tau(\ln V_{DD} - \ln \Delta V_{in})$$

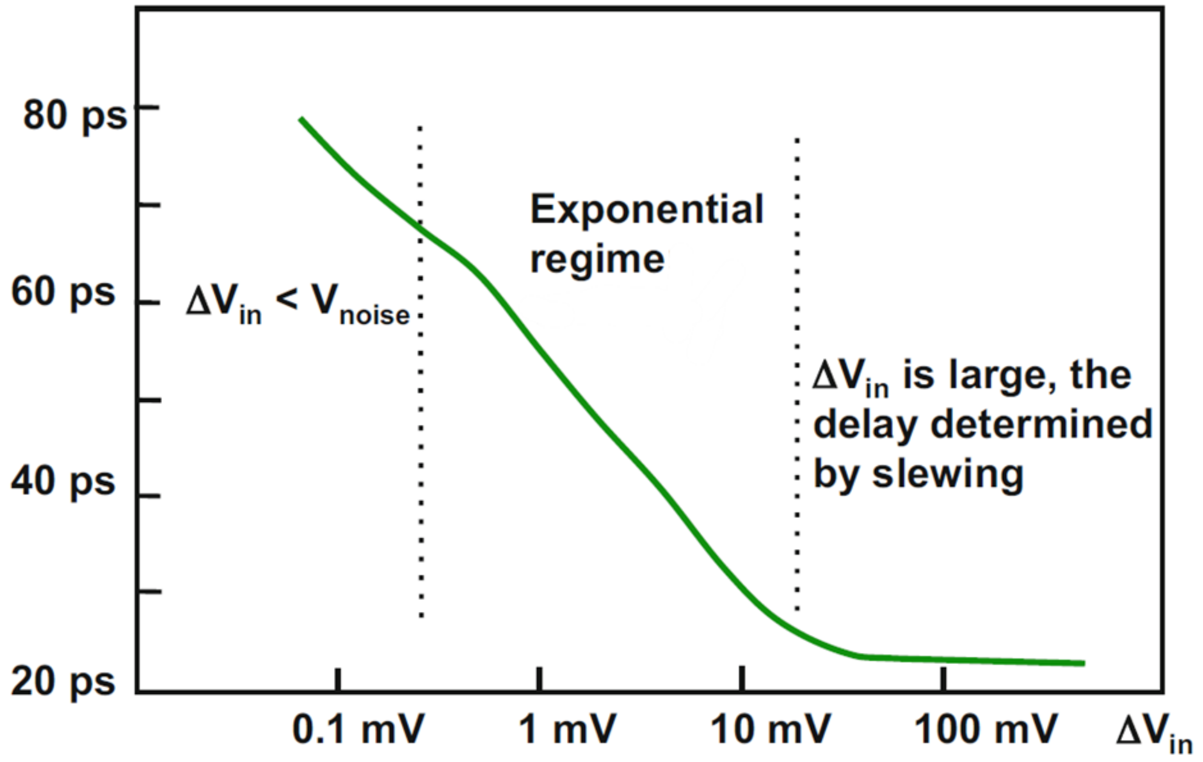


[M. Pelgrom, 2017]

Metastability

- For very small overdrive voltages, the latch/FF cannot reach a decision **within its propagation delay** (t_{pDQ} / t_{pCQ})
  - The latch/FF will not generate a clear “zero” or “one” output level before t_{pDQ} / t_{pCQ}



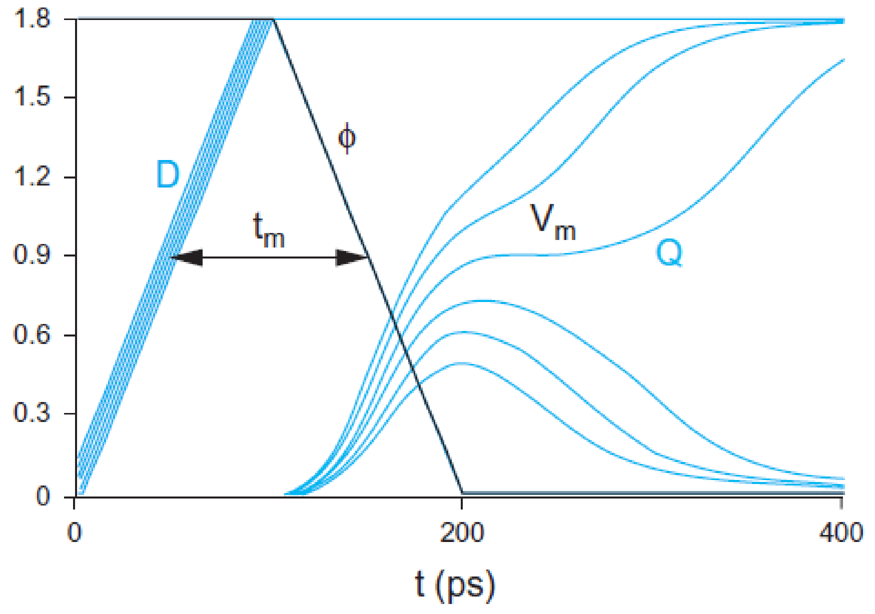


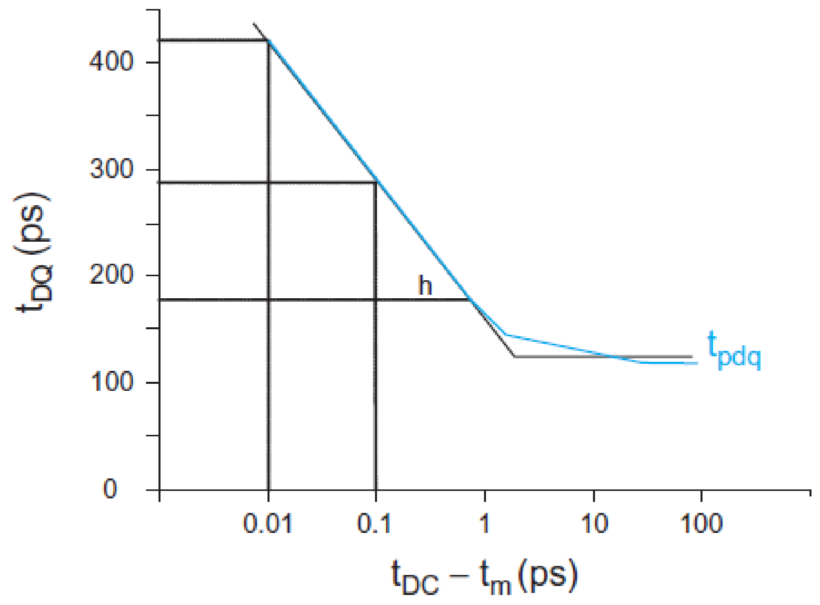
[M. Pelgrom, 2017]

Metastability

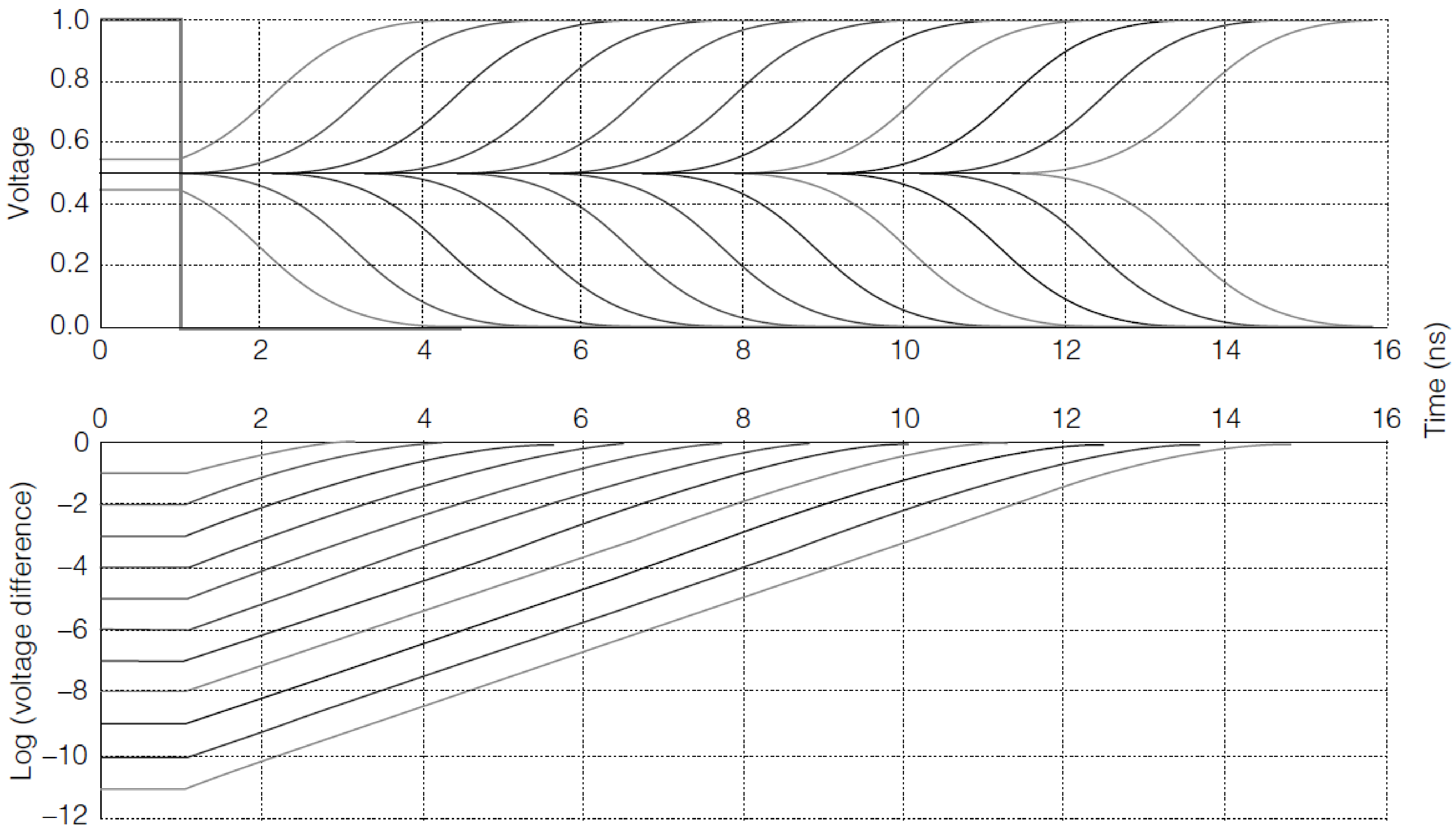
- Delay increases for inputs that arrive too close to t_m
- The time to resolve from metastability depends on the gain-bandwidth product ($GBW = 1/\tau$) of the latch feedback loop

- Delay increases for inputs that arrive too close to t_m
- The time to resolve from metastability depends on the gain-bandwidth product ($GBW = 1/\tau$) of the latch feedback loop





Metastability



Probability of Metastability

- Probability of Metastability (MS) is the probability that the asynchronous input edge occurs in the metastability window (T_W)
  - Metastability can be simply defined as FF output delay > t_{pCQ}

$$T_W \sim t_a = t_{setup} + t_{hold}$$

$$P(MS) = \frac{T_W}{T_C} = T_W f_C$$

Mean Time Between Failures (MTBF)

$$P(MS) = \frac{T_W}{T_C} = T_W f_C$$

- Assume the asynchronous input changes once per second
  - Probability of MS per second = probability of failure per second = Failure Rate = $P(MS)$
- If input changes at a rate of f_{in} (i.e., f_{in} times per second)
  - Probability of failure per second = Failure Rate (FR) = BER = $f_{in} \frac{T_W}{T_C} = T_W f_{in} f_C$
- Failure happens (on average) every $1/FR$
  - Called mean time between failures (MTBF)

$$MTBF = \frac{1}{FR} = \frac{1}{T_W f_{in} f_C}$$

Example

- Assume $f_C = 1GHz, f_{in} = 100MHz, T_W = 20ps$

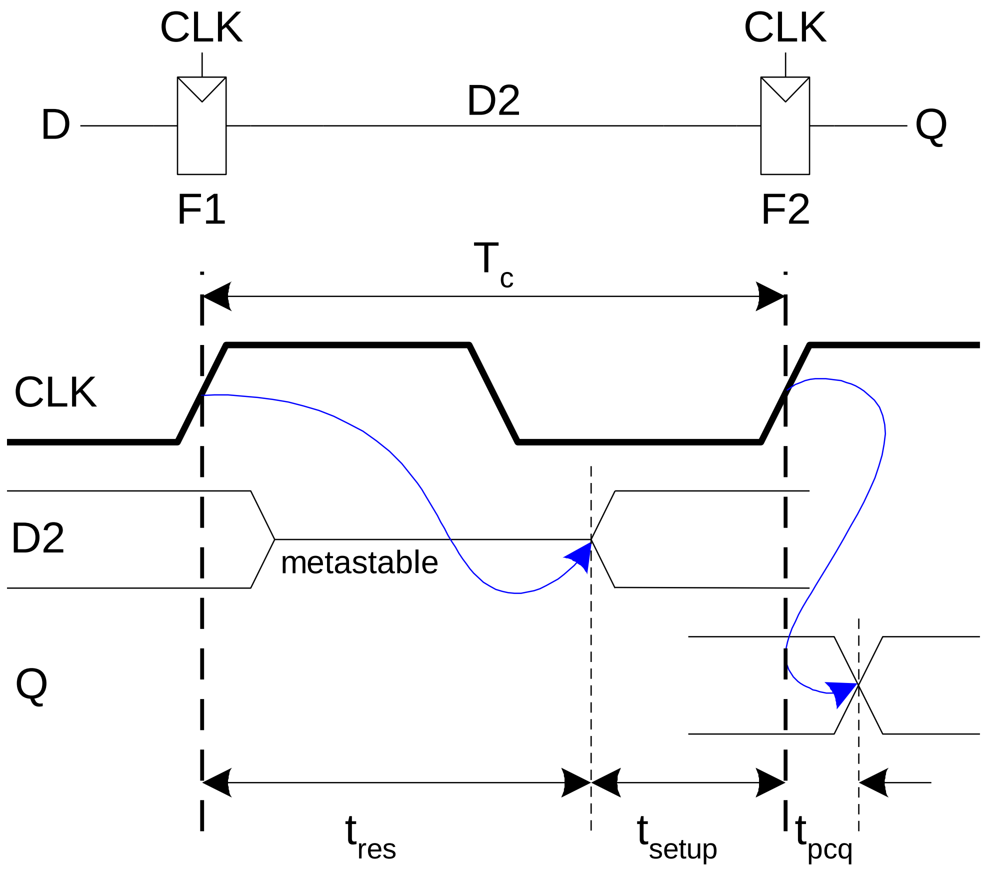
$$P(MS) = \frac{T_W}{T_C} = T_W f_C = 0.02$$

$$FR = T_W f_{in} f_C = 2M/s$$

$$MTBF = \frac{1}{FR} = \frac{1}{T_W f_{in} f_C} = 0.5\mu s$$

Simple Synchronizer

- Give the metastable signal extra time to resolve
  - Add one more FF to give one more clock cycle
  - The more FFs you add, the more time you give



[Harris & Harris, DDCA, 2012]

Probability of Synchronizer Failure

- Assume ambiguity happens when $\Delta V_{out}(t_{res}) < V_{DD}$

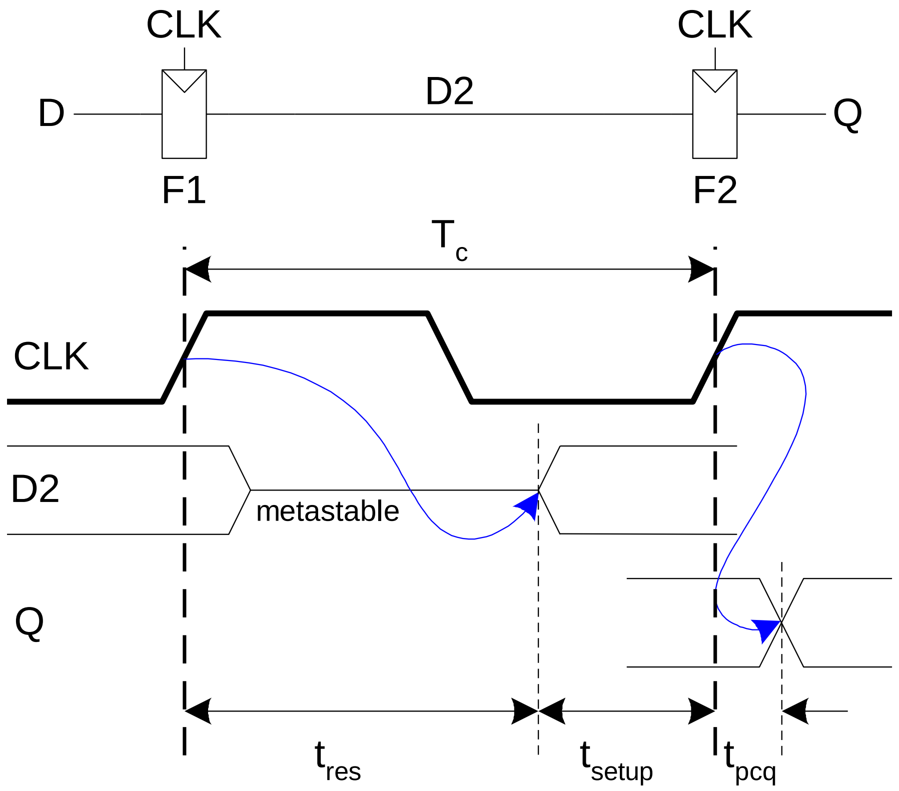
$$\Delta V_{out}\ (t_{res}) = \Delta V_{in} e^{+t_{res}/\tau} < V_{DD}$$

$$\Delta V_{in,min} = V_{DD} e^{-t_{res}/\tau}$$

- Assume ΔV_{in} is uniformly distributed:

$$0 < \Delta V_{in} < V_{DD}$$
$$0 < \Delta V_{in}/V_{DD} < 1$$

$$P(Sync\ Failure) = \frac{\Delta V_{in,min}}{V_{DD}} = e^{-(T_C - t_{setup})/\tau}$$



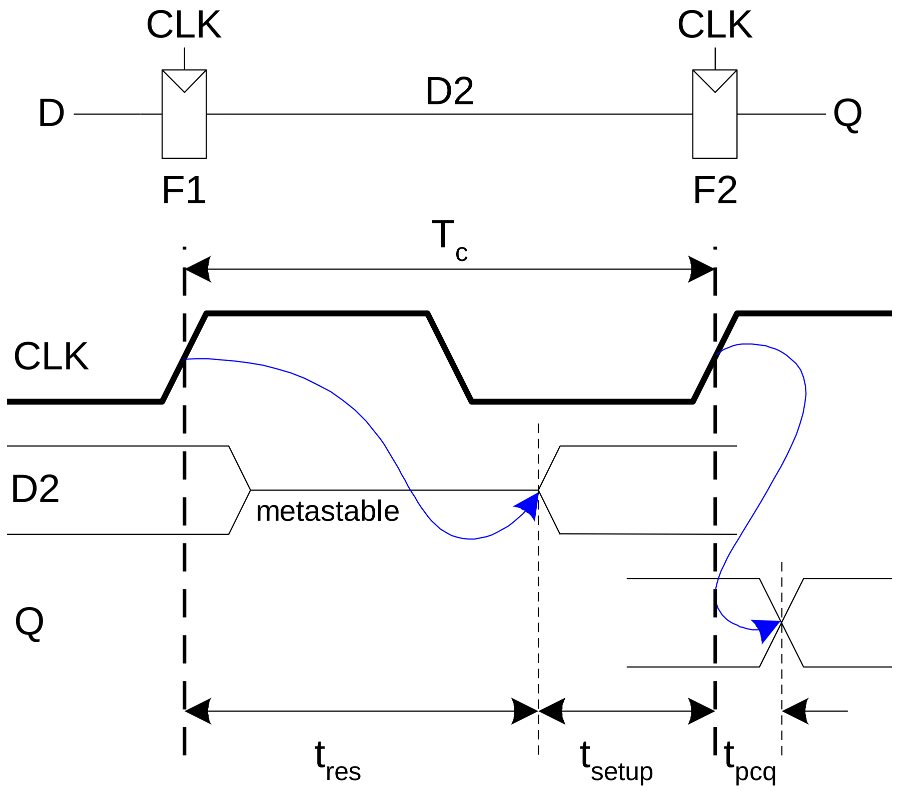
Probability of Failure

$$P(Sync\ Failure) = e^{-(T_C - t_{setup})/\tau}$$

- But the problem occurs only when the latch is metastable!

$$P(Failure) = P(MS) \times P(Sync\ Failure)$$

$$= T_W f_C e^{-(T_C - t_{setup})/\tau}$$



Synchronizer Mean Time Between Failures (MTBF)

- If input changes at a rate of f_{in} (i.e., f_{in} times per second)
  - Probability of failure per second = Failure Rate (FR) = $T_W f_{in} f_C e^{-(T_C - t_{setup})/\tau}$
- Failure happens (on average) every $1/FR$
  - Called mean time between failures (MTBF)

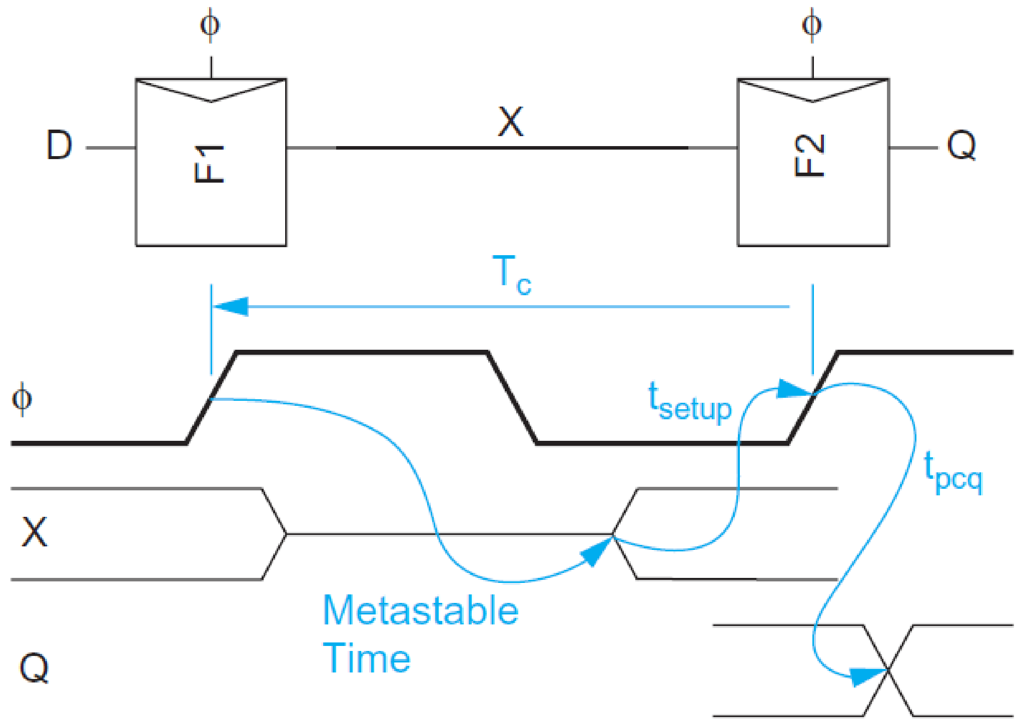
$$MTBF = \frac{1}{FR} = \frac{e^{+(T_C - t_{setup})/\tau}}{T_W f_{in} f_C}$$

Simple Synchronizer Summary

- The synchronizer has a latency of one clock cycle (T_C).
- Fails if X does not settle t_{setup} before the second clock edge.
- MTBF: Mean time between failures
- f_{in}: asynchronous input toggling rate
- $T_W/T_c = T_W f_C$ describes the probability of change during the aperture
- τ: latch time constant (= 1/GBW)

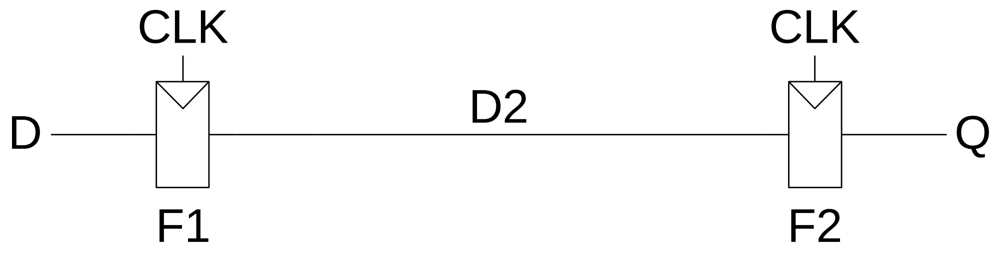
$$MTBF = \frac{1}{FR}$$

$$= \frac{e^{+(T_C - t_{setup})/\tau}}{T_W f_{in} f_C}$$



Example

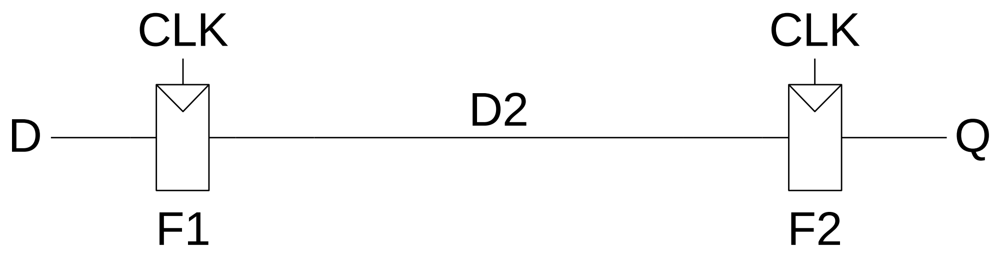
- Assume: $T_c = 1/500$ MHz = 2 ns $\tau = 200$ ps
  $T_W = 150$ ps $t_{setup} = 100$ ps
  $f_{in} = 10$ events per second
- What is the probability of failure? MTBF?



[Harris & Harris, DDCA, 2012]

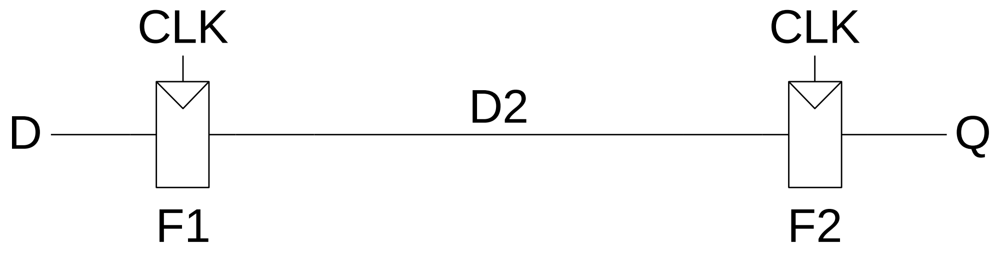
Example

- Assume: $T_c = 1/500$ MHz = 2 ns $\tau = 200$ ps
  $T_W = 150$ ps $t_{setup} = 100$ ps
  $f_{in} = 10$ events per second
- What is the probability of failure? MTBF?
- P(failure) $= (150 \text{ ps}/2 \text{ ns})\, e^{-(1.9 \text{ ns})/200 \text{ ps}}$
  $= 5.6\text{e-}6$
- FR = P(failure)/second $= 10 \times (5.6\text{e-}6)$
  $= 5.6\text{e-}5$ / second
- MTBF = 1/FR ≈ 5 hours



Example

- A synchronizer flip-flop in a 0.25 μm process has τ = 20 ps and T_W = 15 ps. Assuming the input toggles at f_{in} = 50 MHz and the setup time is negligible, what is the minimum clock period T_c for which the MTBF exceeds one year?
  - 1 year $\approx \pi \times 10^7$ seconds

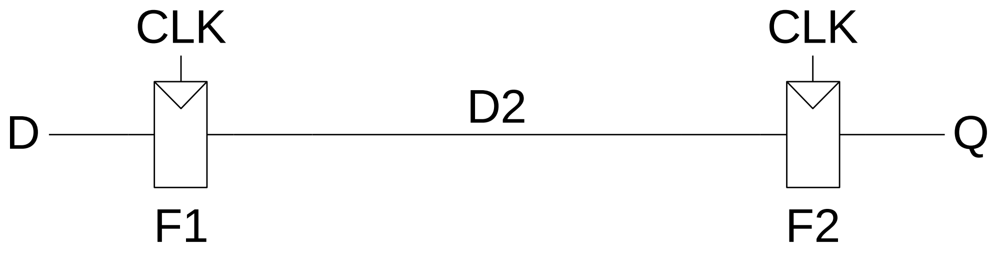


Example

- A synchronizer flip-flop in a 0.25 μm process has τ = 20 ps and T_W = 15 ps. Assuming the input toggles at f_{in} = 50 MHz and the setup time is negligible, what is the minimum clock period T_c for which the MTBF exceeds one year?
  - 1 year $\approx \pi \times 10^7$ seconds
  - Solve numerically
  - $T_c \approx 625ps\ (1.6GHz)$

$$\pi \times 10^7 = \frac{T_c e^{\frac{T_c}{20\times 10^{-12}}}}{\left(5\times 10^7\right)\left(15\times 10^{-12}\right)}$$

- If MTBF = 1000 years
  - $T_c \approx ?$

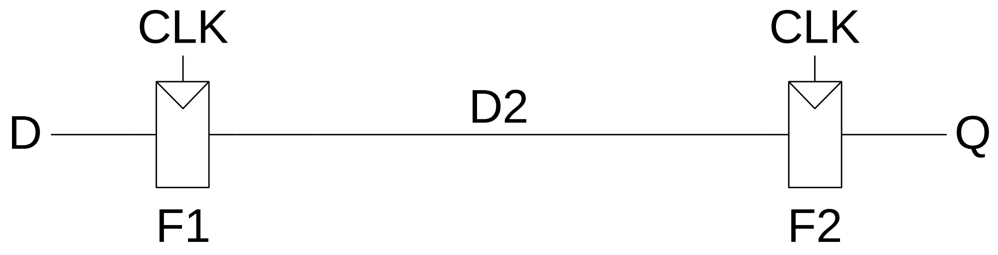


Example

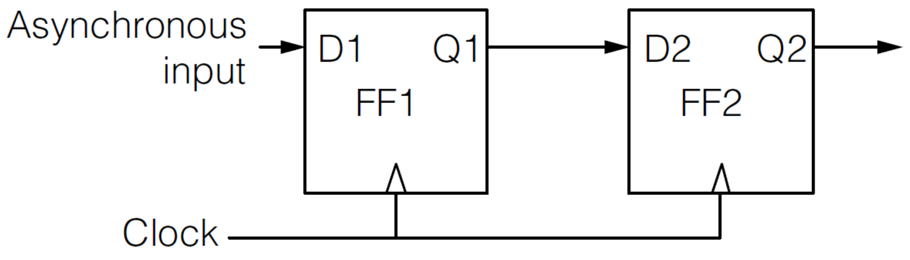
- A synchronizer flip-flop in a 0.25 μm process has τ = 20 ps and T_W = 15 ps. Assuming the input toggles at f_{in} = 50 MHz and the setup time is negligible, what is the minimum clock period T_c for which the MTBF exceeds one year?
  - 1 year $\approx \pi \times 10^7$ seconds
  - Solve numerically
  - $T_c \approx 625ps\ (1.6GHz)$

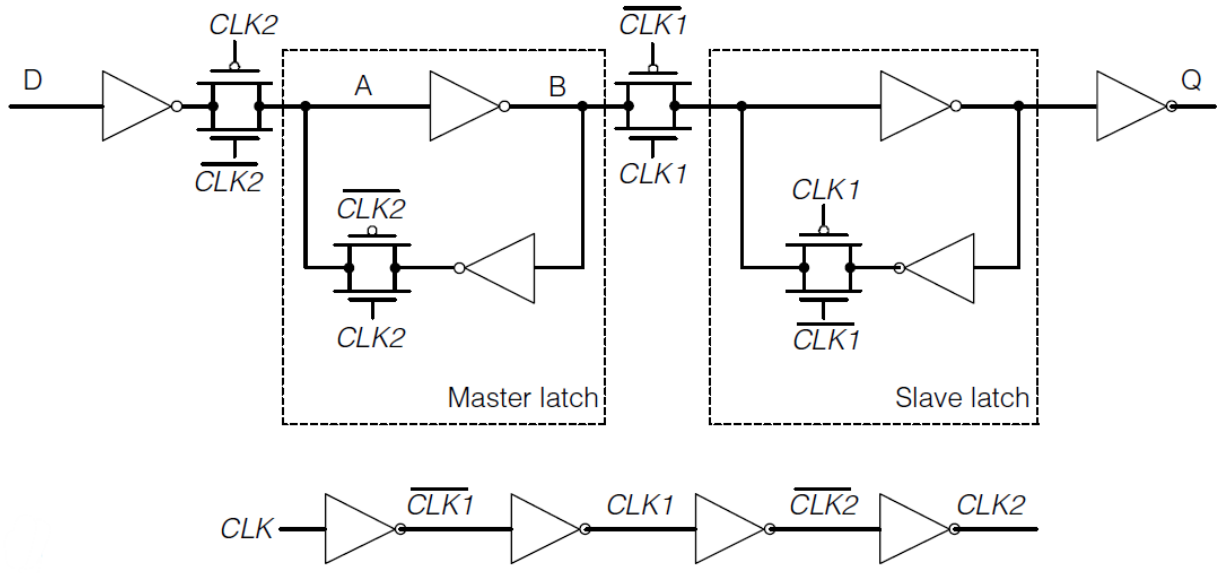
$$\pi \times 10^7 = \frac{T_c e^{\frac{T_c}{20 \times 10^{-12}}}}{\left(5 \times 10^7\right)\left(15 \times 10^{-12}\right)}$$

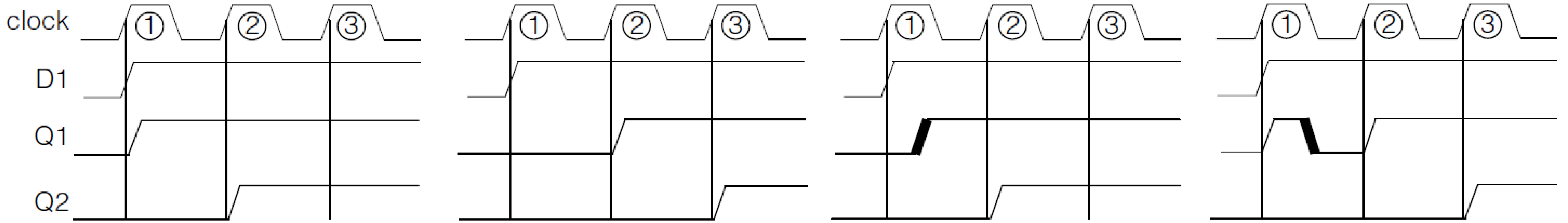
- If MTBF = 1000 years
  - $T_c \approx 760ps\ (1.3GHz)$
  - Exponential dependence on T_c!!!



Synchronizer Behavior



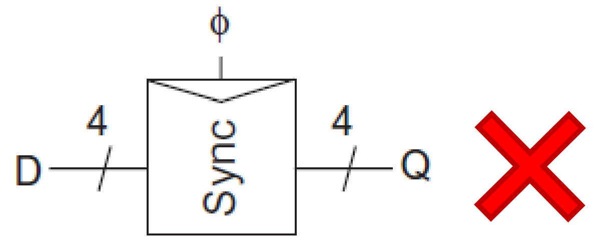


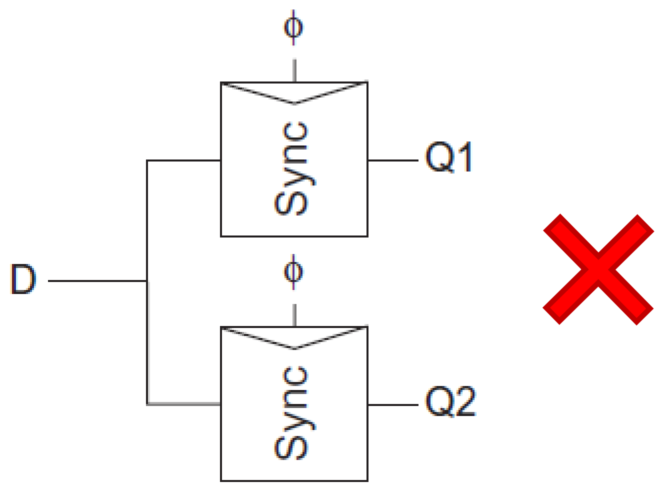


[Ginosar, 2011]

Synchronization Pitfalls

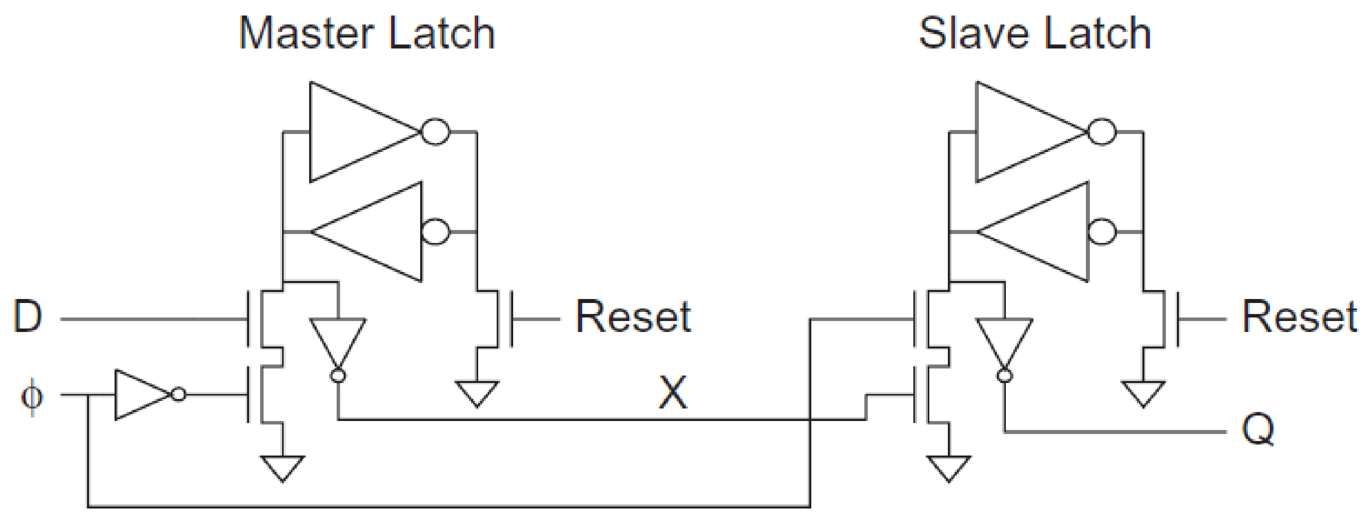
- ❑ Do NOT synchronize data lines
  - Unless gray codes are used (only one bit changes at a time)
  - Some bits will pass after one cycle and others after two cycles
- ❑ Do NOT drive two synchronizers with the same signal
  - The two synchronizers can resolve to different values
- ❑ Do NOT put the sync FFs far apart
  - The wire delay will eat from your resolution time
- ❑ Synchronizers must have good feedback loops
  - You cannot use dynamic FFs
- ❑ Overall MTBF depends on number of synchronizers





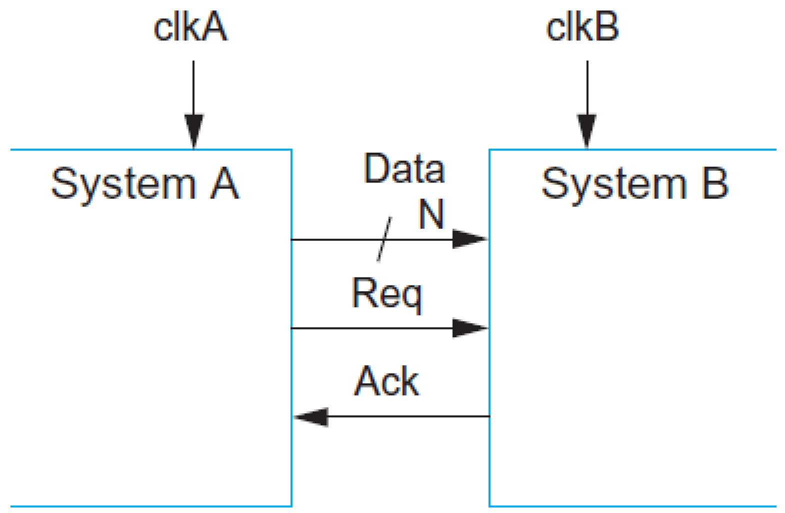
Synchronizer Flip-Flop

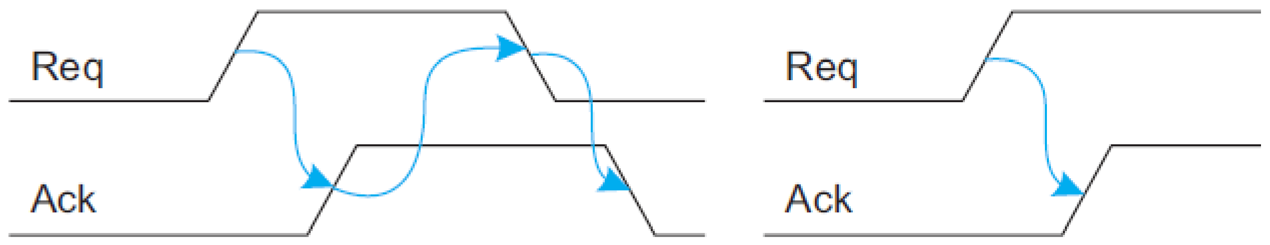
- Master and slave jamb latches
- Maximize feedback loop GBW (minimizes $\tau = C_L/g_m$)
  - Simple cross-coupled inverter pair
  - Minimize loading on feedback loop
    - Reset to 0 and only set to 1 if D = 1
    - PDN just large enough to overpower the cross-coupled pair
    - X and Q are buffered with small inverters



Asynchronous Communication

- Multiple clock domains
- Handshake protocol
  - Four-phase: level-sensitive
  - Two-phase: edge-sensitive
- Synchronizers required for handshake lines (Request & Ack)

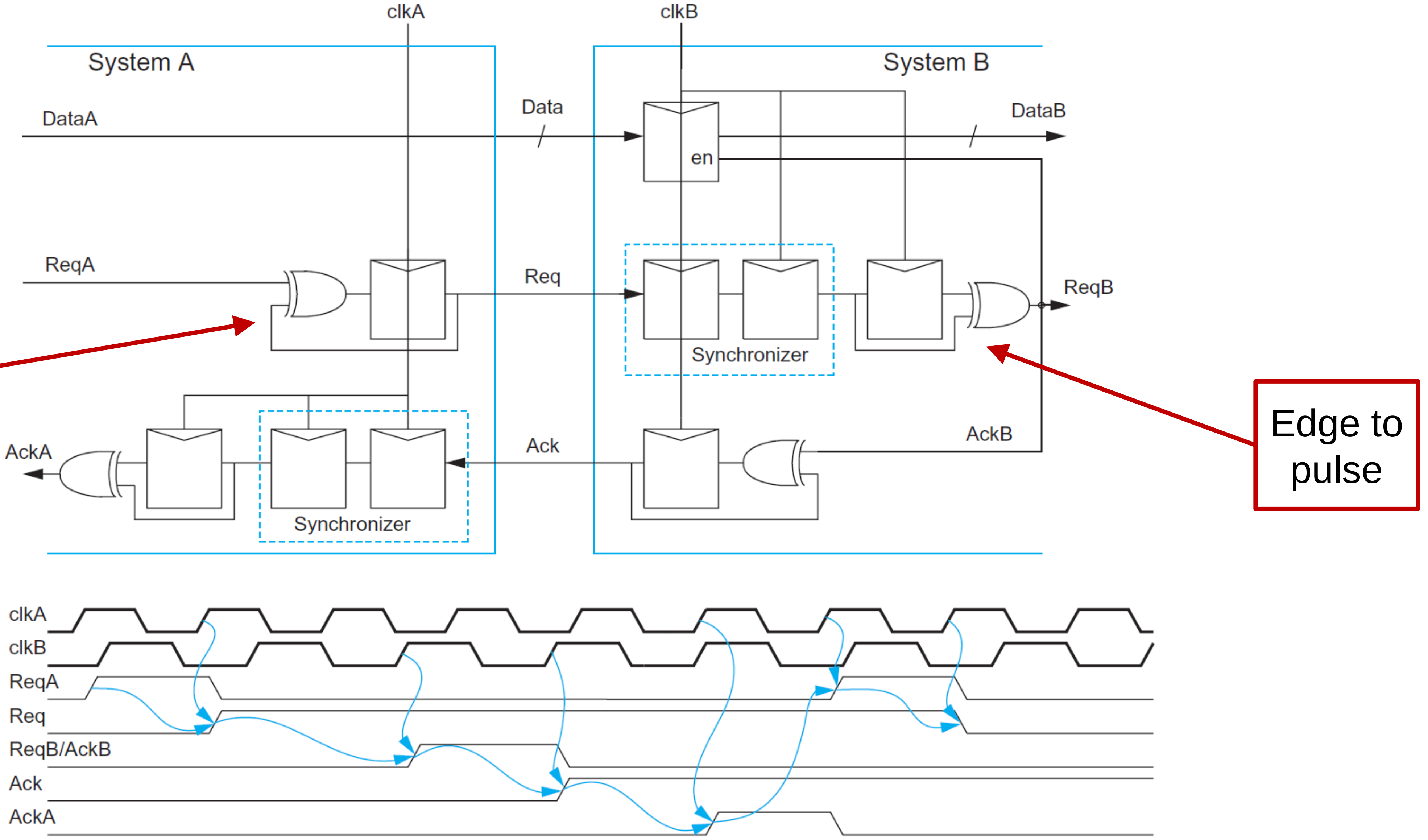




(a) Four-Phase (b) Two-Phase

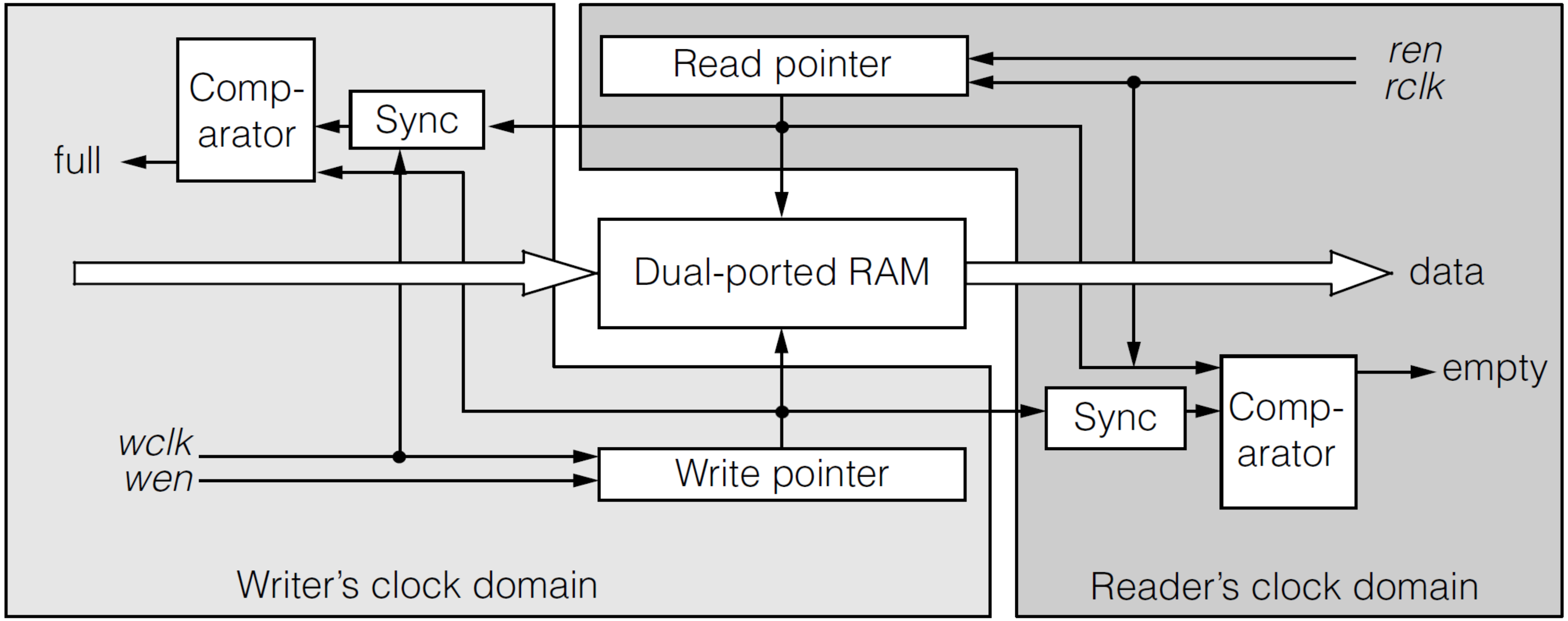
Two-Phase Handshake Example

- Significant latency added, much lower throughput



Two-clock FIFO Buffer

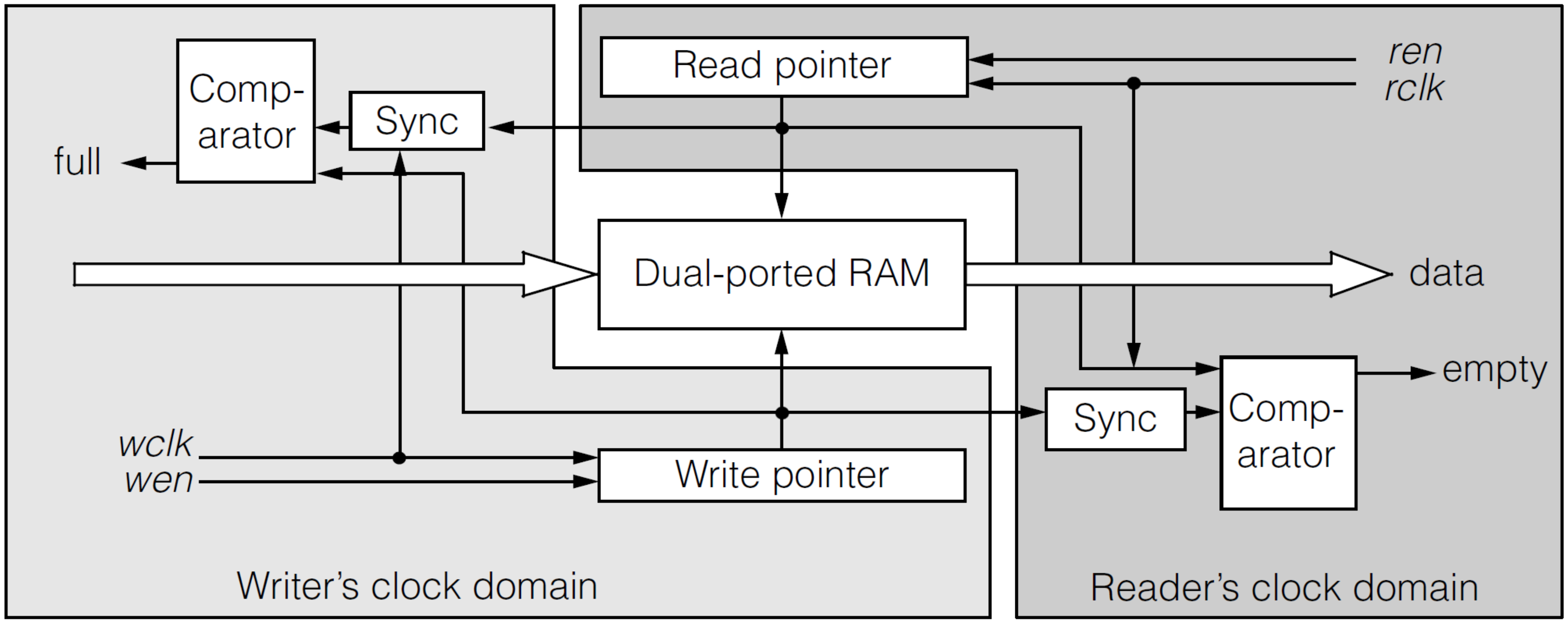
- High-throughput asynchronous communication
- Typically available as library element or IP core
- On write and on read, the write pointer and the read pointer are respectively incremented.



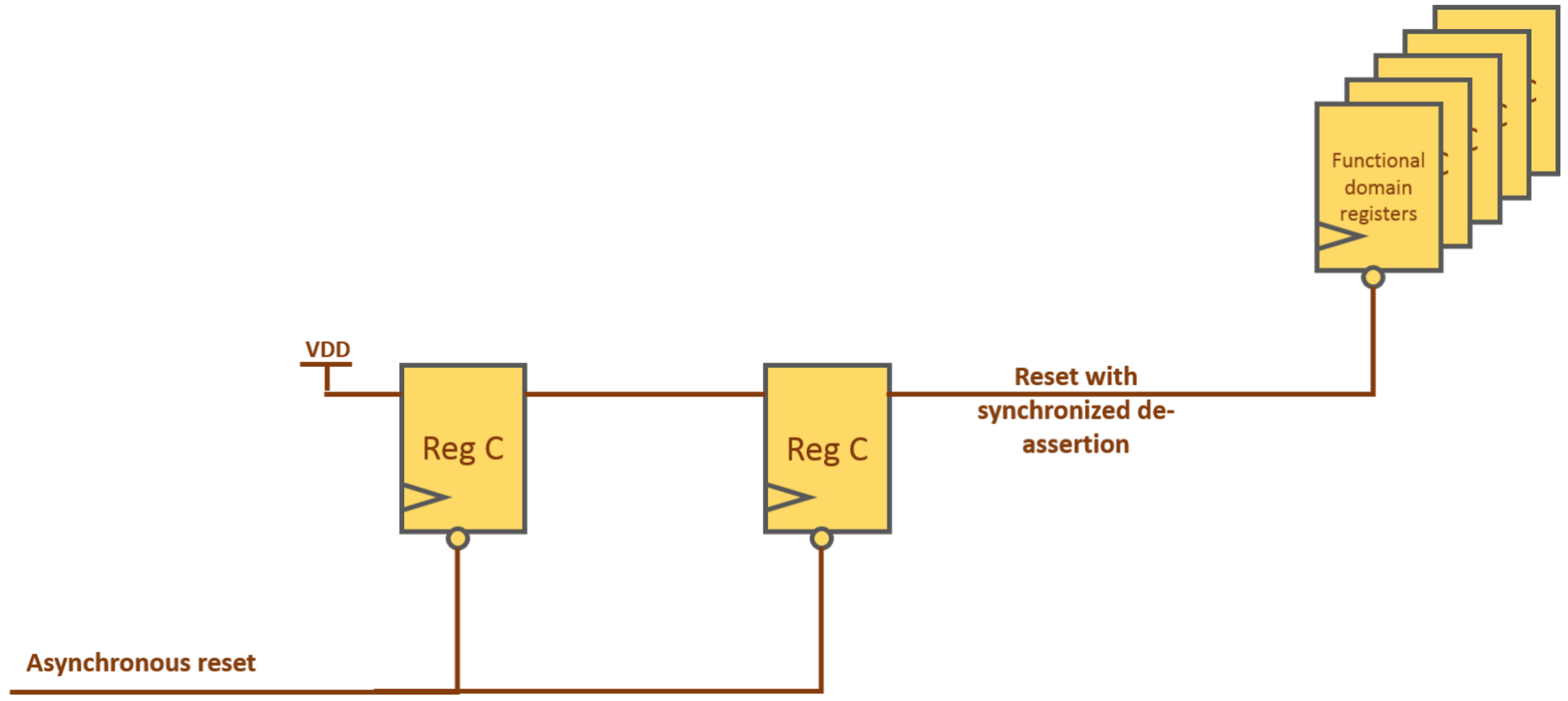
[Ginosar, 2011]

Two-clock FIFO Buffer

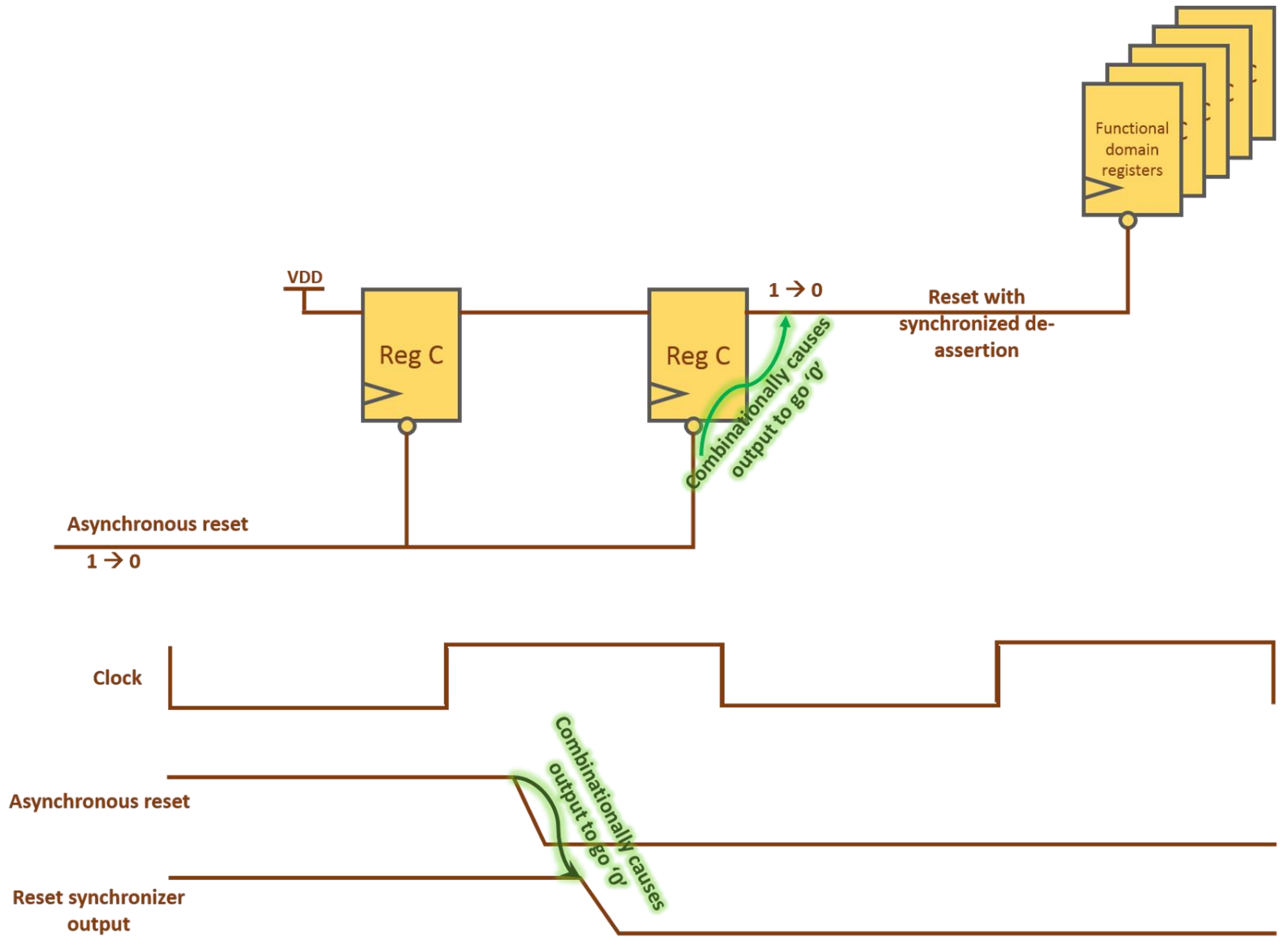
- Empty: When the read pointer points to the same word as the write pointer
- Synchronization is applied to the pointers NOT to the data (pointers are in gray code)
- When the two pointers are far from each other, no synchronization latency is incurred



Reset Synchronization

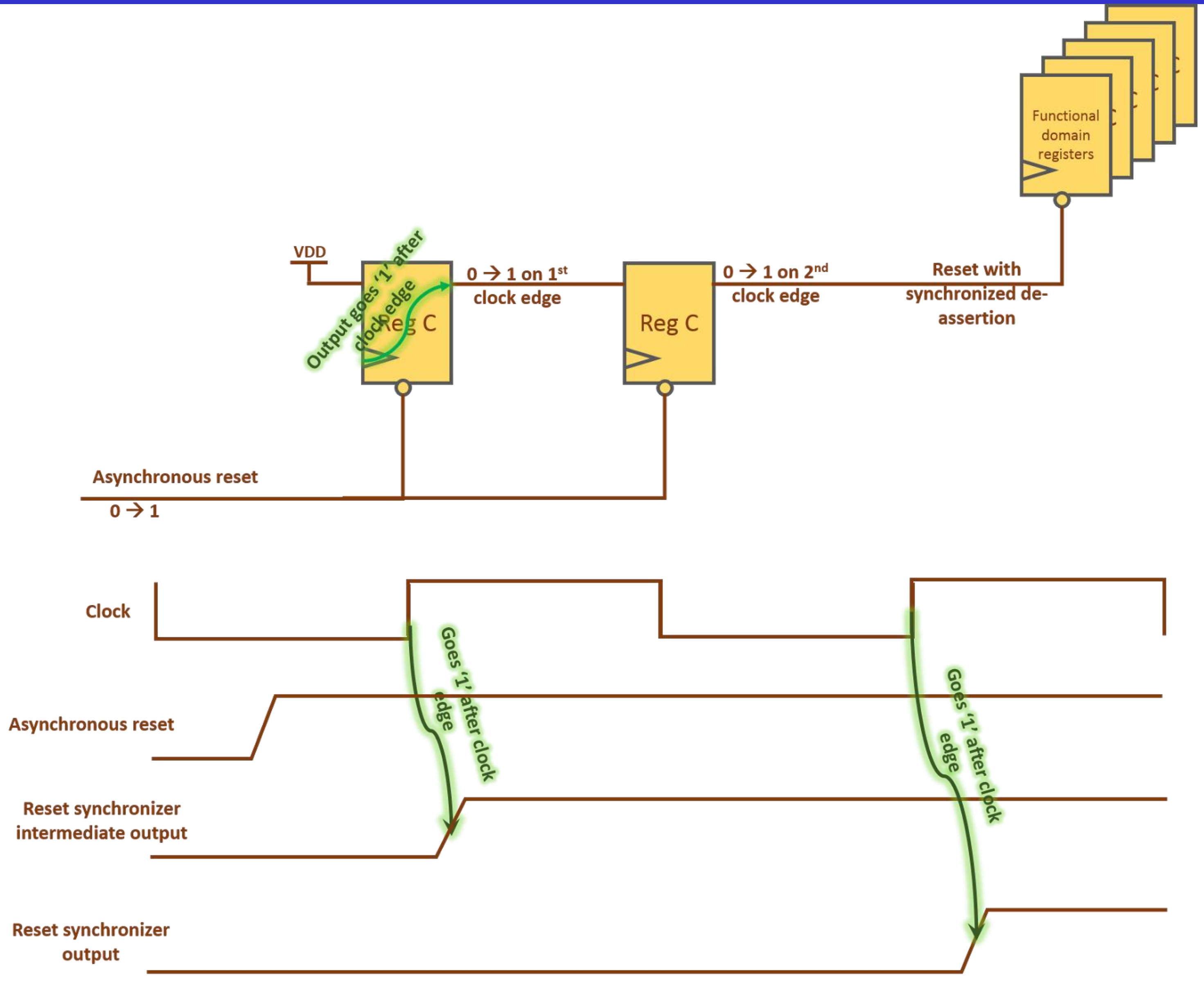

Functional domain registers
VDD
Reg C
Reg C
Reset with synchronized de-assertion
Asynchronous reset

Reset Synchronization


Functional domain registers
VDD
Reg C
Reg C
1 → 0
Reset with synchronized de-assertion
Combinationally causes output to go '0'
Asynchronous reset
1 → 0
Clock
Asynchronous reset
Reset synchronizer output
Combinationally causes output to go '0'

[https://vlsiuniverse.blogspot.com/2016/09/reset-synchronizer.html]

Reset Synchronization

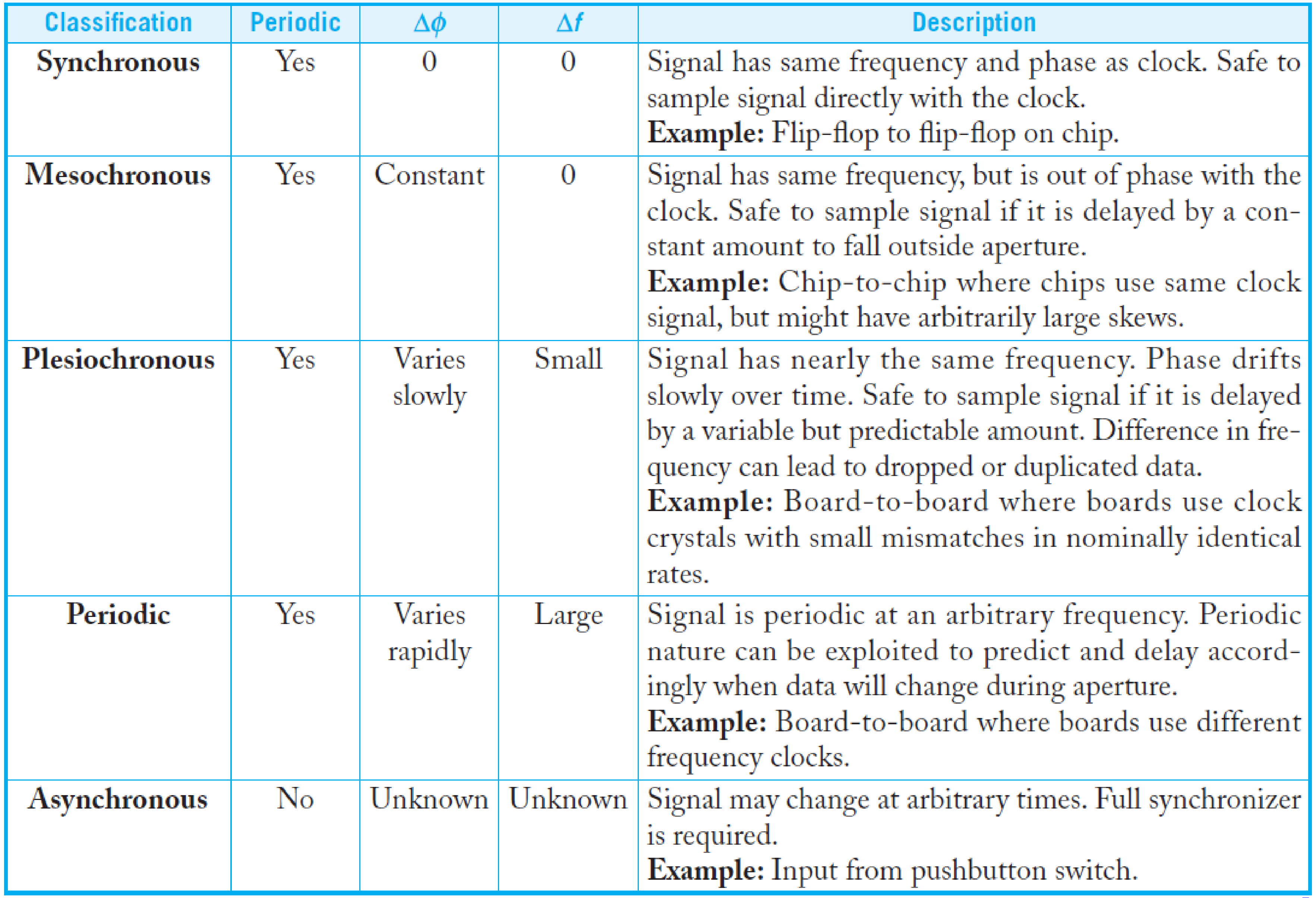


Functional domain registers
VDD
Output goes '1' after clock edge
Reg C
0 → 1 on 1st clock edge
Reg C
0 → 1 on 2nd clock edge
Reset with synchronized de-assertion
Asynchronous reset
0 → 1
Clock
Asynchronous reset
Goes '1' after clock edge
Reset synchronizer intermediate output
Goes '1' after clock edge
Reset synchronizer output

[https://vlsiuniverse.blogspot.com/2016/09/reset-synchronizer.html]

Degrees of Synchrony

- ❑ Several "smart" solutions were proposed for the spectrum between sync and async
- ❑ But the two-clock FIFO is the simplest, safest, and most popular solution



Classification	Periodic	$\Delta\phi$	Δf	Description
Synchronous	Yes	0	0	Signal has same frequency and phase as clock. Safe to sample signal directly with the clock. **Example:** Flip-flop to flip-flop on chip.
Mesochronous	Yes	Constant	0	Signal has same frequency, but is out of phase with the clock. Safe to sample signal if it is delayed by a constant amount to fall outside aperture. **Example:** Chip-to-chip where chips use same clock signal, but might have arbitrarily large skews.
Plesiochronous	Yes	Varies slowly	Small	Signal has nearly the same frequency. Phase drifts slowly over time. Safe to sample signal if it is delayed by a variable but predictable amount. Difference in frequency can lead to dropped or duplicated data. **Example:** Board-to-board where boards use clock crystals with small mismatches in nominally identical rates.
Periodic	Yes	Varies rapidly	Large	Signal is periodic at an arbitrary frequency. Periodic nature can be exploited to predict and delay accordingly when data will change during aperture. **Example:** Board-to-board where boards use different frequency clocks.
Asynchronous	No	Unknown	Unknown	Signal may change at arbitrary times. Full synchronizer is required. **Example:** Input from pushbutton switch.

More on Metastability

- M. Arora, *The Art of Hardware Architecture: Design Methods and Techniques for Digital Circuits*, Springer, 2012.

  https://link.springer.com/book/10.1007%2F978-1-4614-0397-5
- R. Ginosar, "Metastability and Synchronizers: A Tutorial," *IEEE Design & Test of Computers*, 2011.

Thank you!