TECHNISCHE
UNIVERSITÄT
WIEN
Vienna University of Technology

How to Synchronize a Pausible Clock to a Reference

Robert Najvirt, Andreas Steininger

GALS Communication

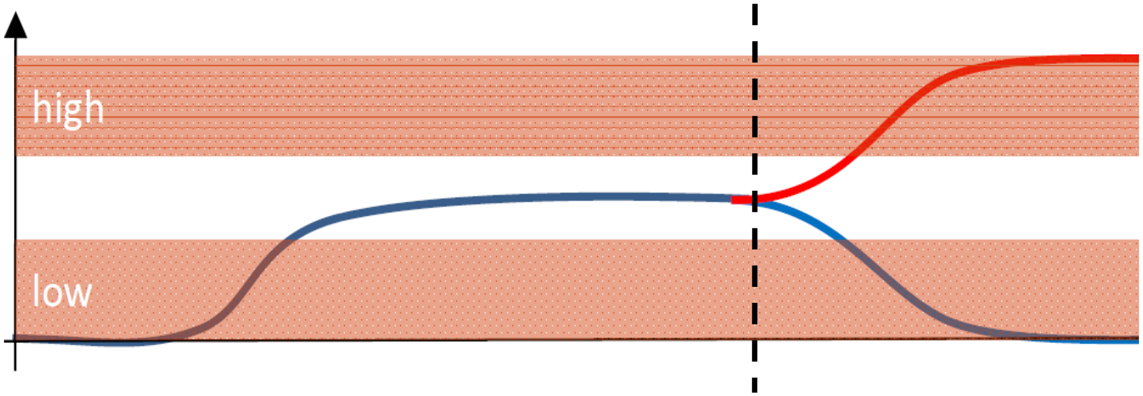
- The communication between two (locally) synchronous modules has inevitable potential for metastable upsets
- Except for smartly designed systems utilizing assumptions on the clock sources
- Two ways of handling metastability:
  Time safe vs. value safe

Time safe solution

- Allow for fixed resolution time
- Use result no matter what

👍 Allows to use free running clock sources

👎 A non-zero probability of failure always remains

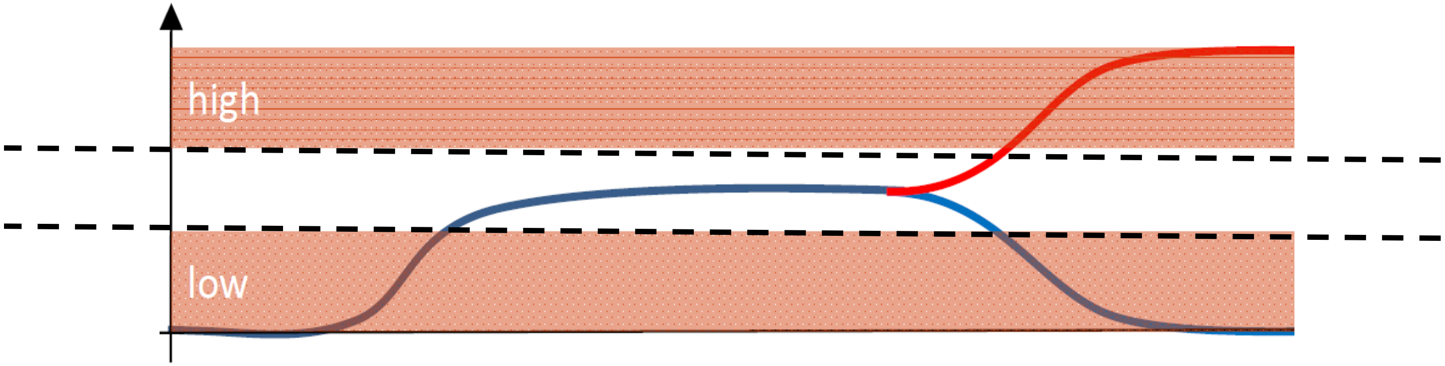


Value safe solution

- Wait for metastability to resolve
- Use result when ready

👍 Allows to design circuits free of metastable upsets

👎 Requires stretching of clock period



Ideal solution

- 👍 Use free running clock sources
- 👍 Communicate without risk of metastable upsets

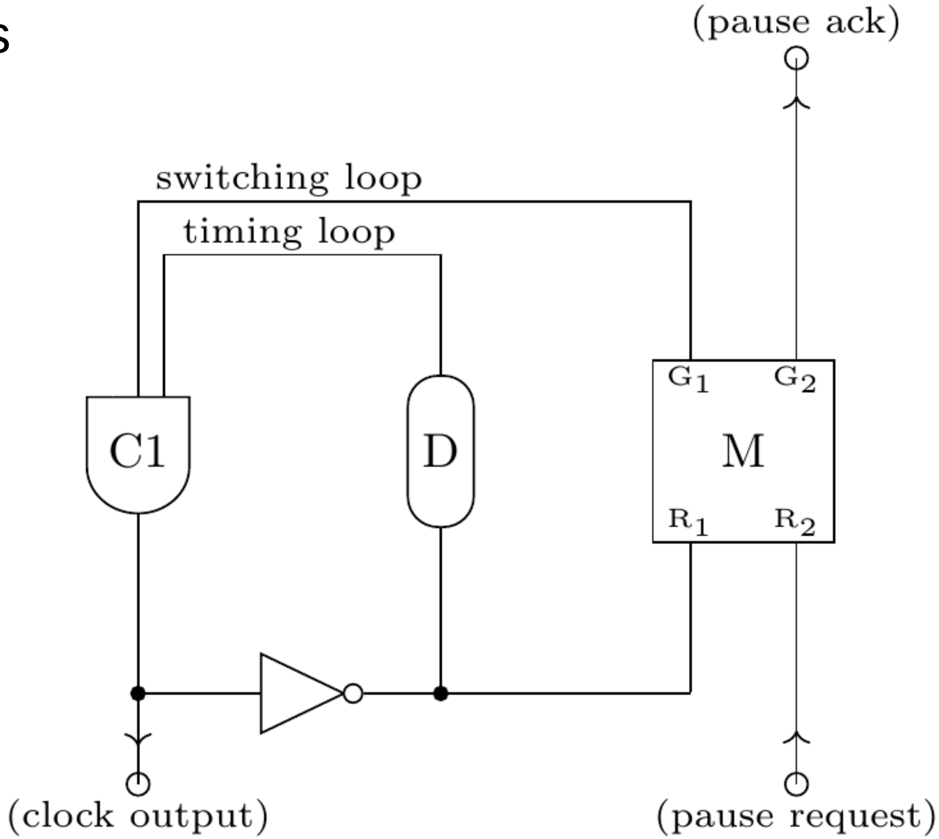
➜ Unfortunately impossible

- Synchronization impossibility
- Pausing and starting a free running clock impossibility

Compromise solution: Use ring oscillator referenced with a free running clock

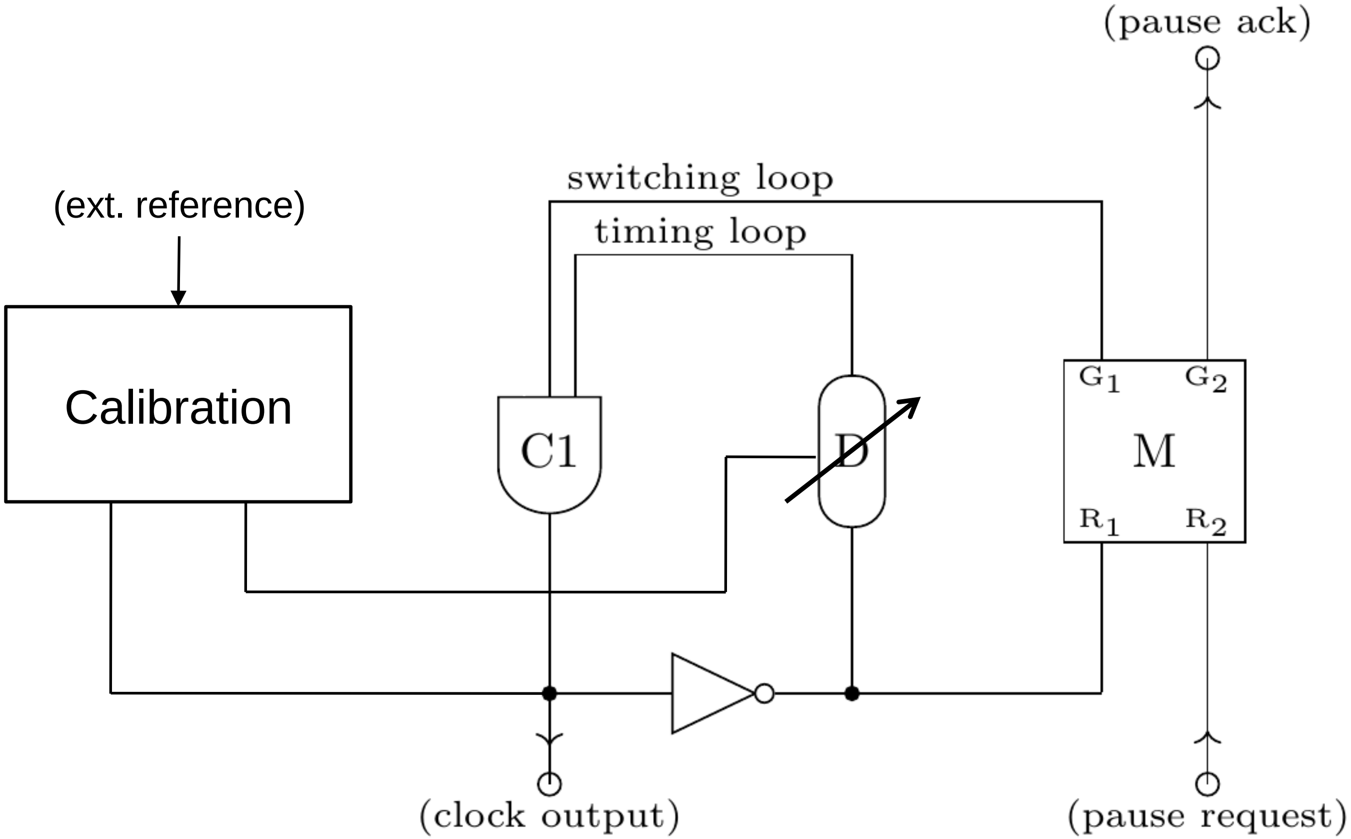
The standard pausible clock

- Two loops
  - Timing loop guarantees minimum pulse widths
  - Switching loop handles pausing and restarting by delaying a transition



Adding a reference to a pausible clock

Calibration of the delay line



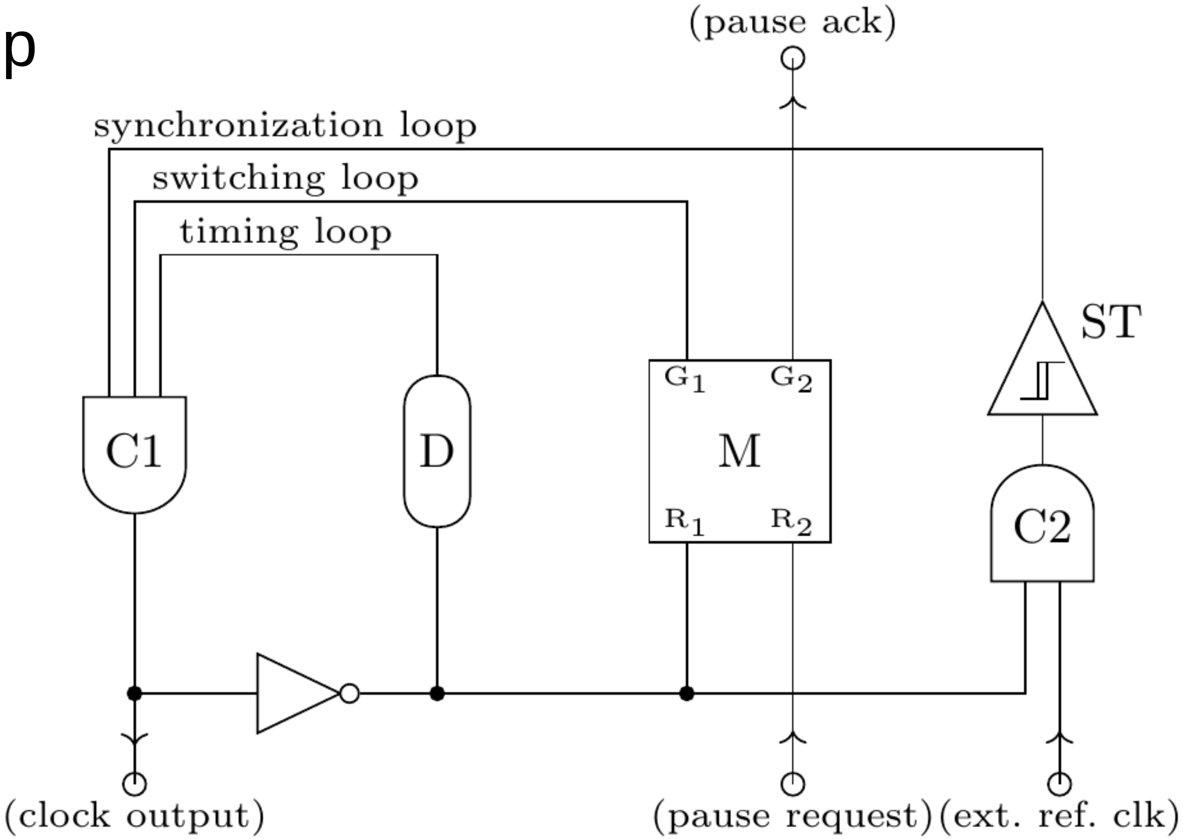
Adding a reference to a pausible clock

Our approach:

Add another loop

for direct

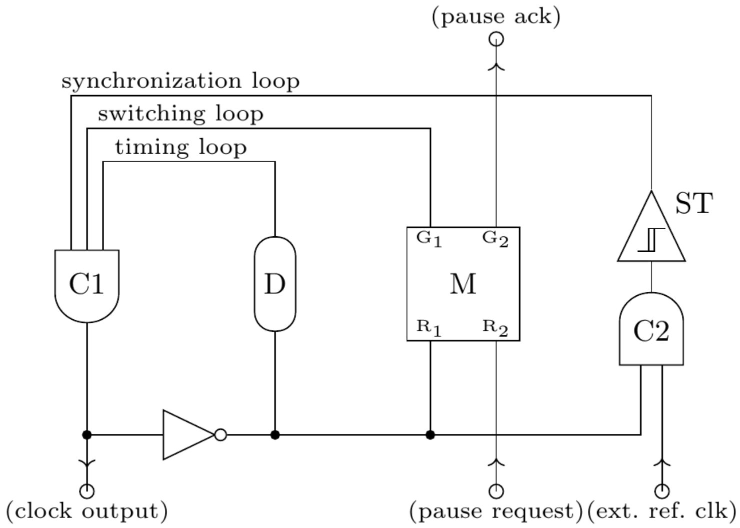
synchronization



Analysis – Pausing

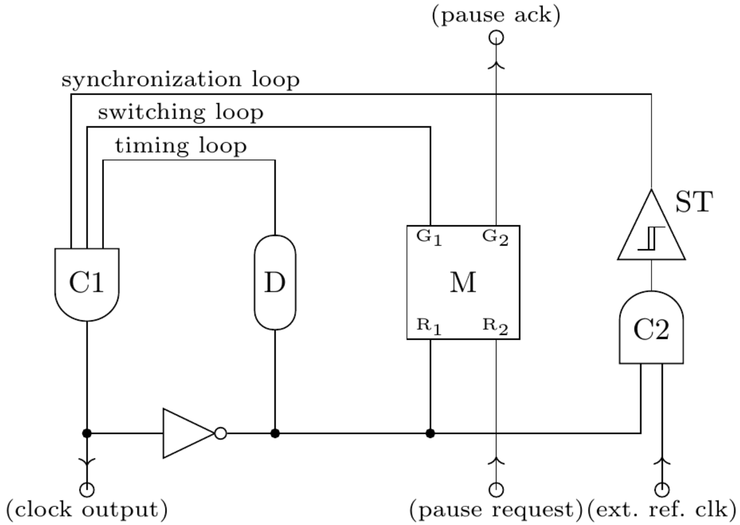
Behavior when being paused not affected by third loop:

- The mutex delays the switching loop while paused
- Metastability is correctly handled in a value safe manner



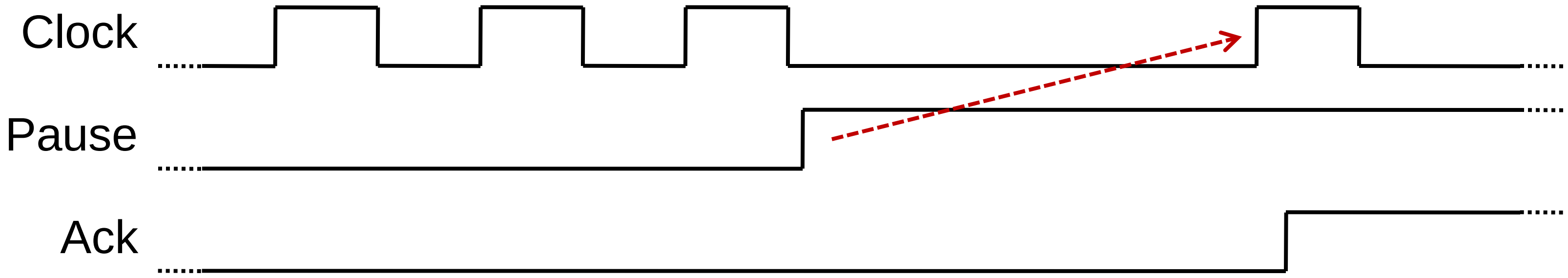
Analysis – Starting

- First clock transition at an arbitrary phase
- The next aligned by synchronization loop to coincide with corresponding ones from the reference
- However, minimum pulse widths still dictated by timing loop
- Inevitable metastability of C2 handled by Schmitt-Trigger in a value safe manner



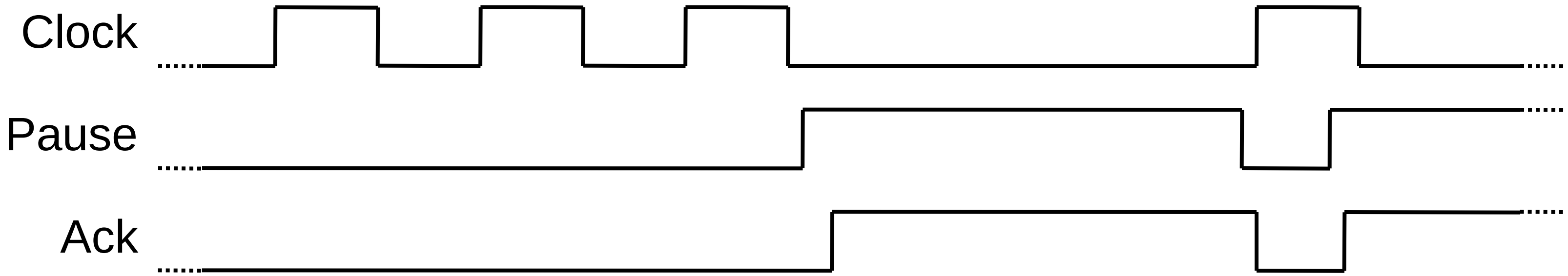
Analysis – The missing case

- When, after a metastabe period, the mutex decides against a pause request



Analysis – The missing case

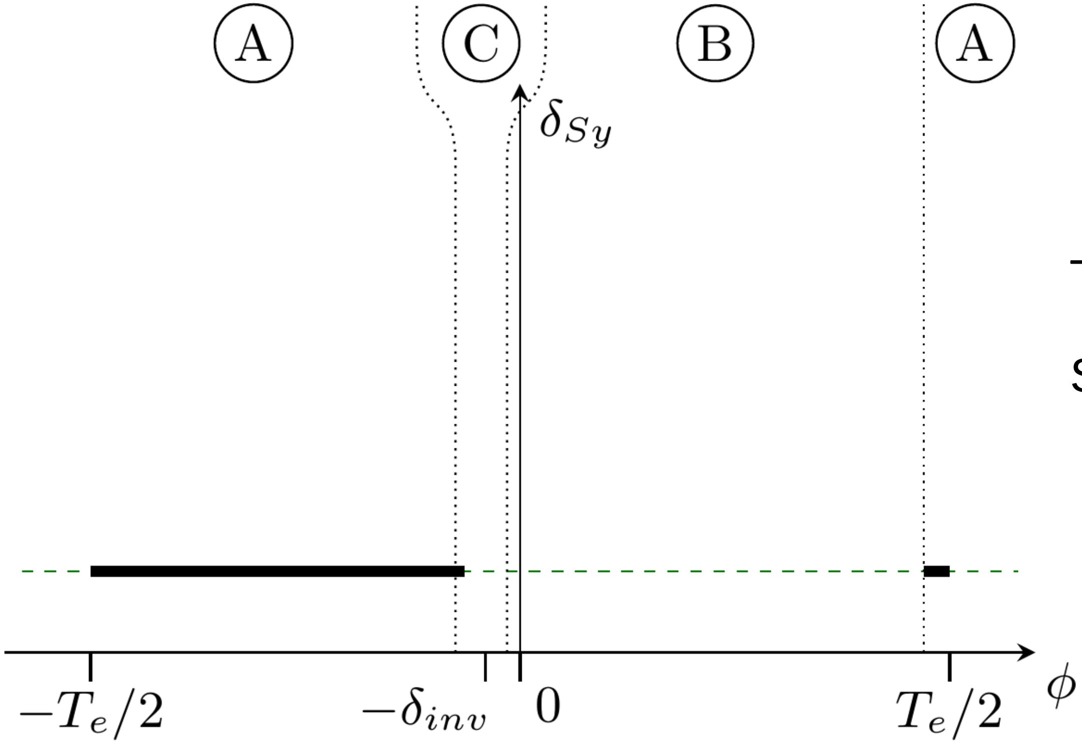
- When, after a metastabe period, the mutex decides against a pause request

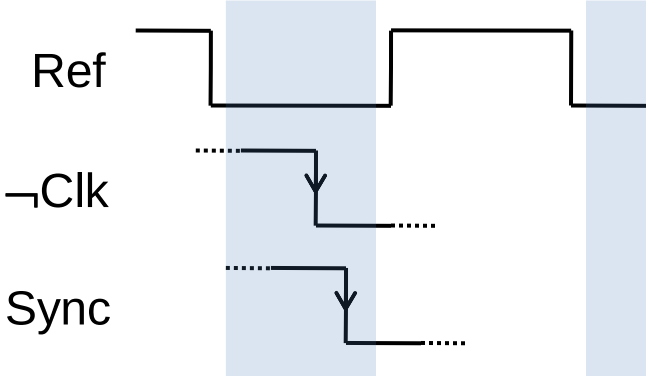


➔ Equivalent to two successive pause requests

Analysis – Synchronization loop delay

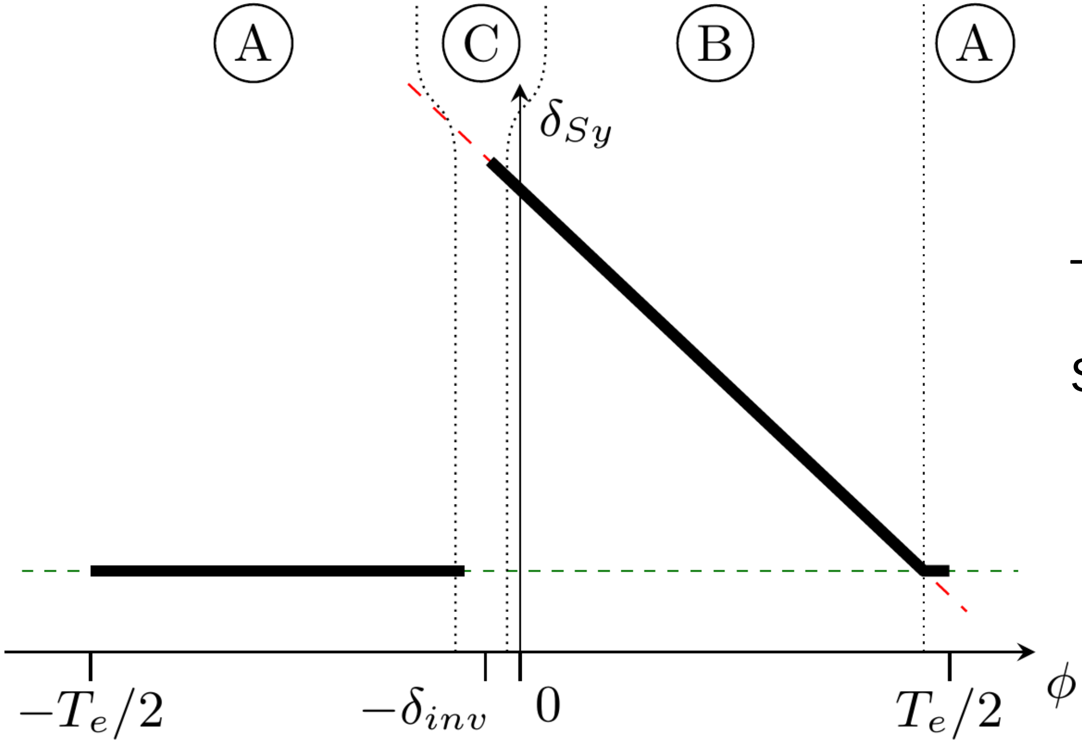
1) Direct propagation

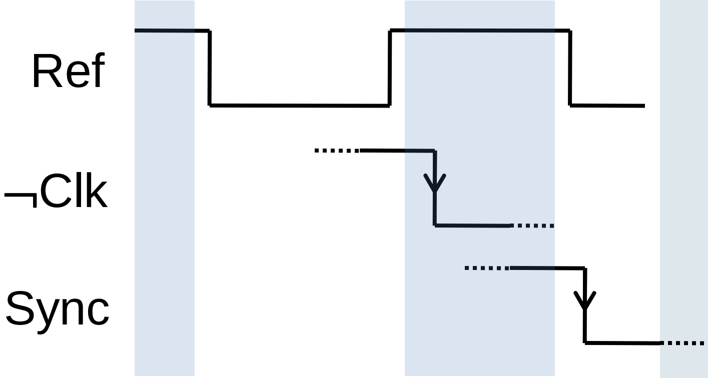




Analysis – Synchronization loop delay

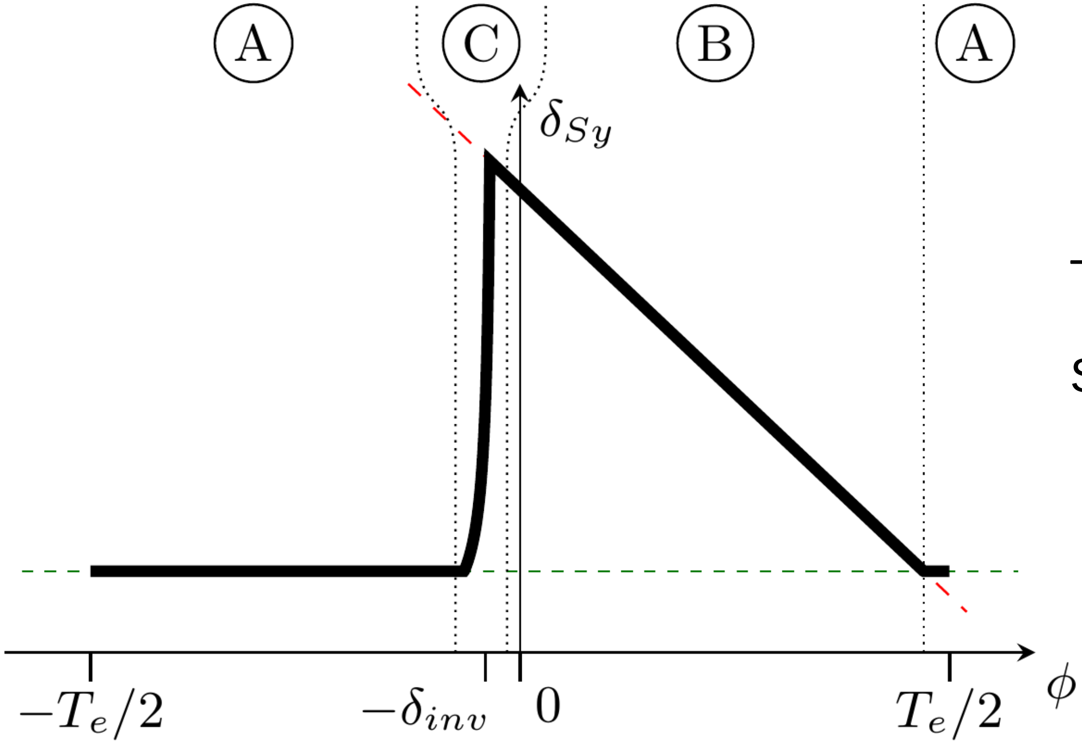
2) Synchronized propagation

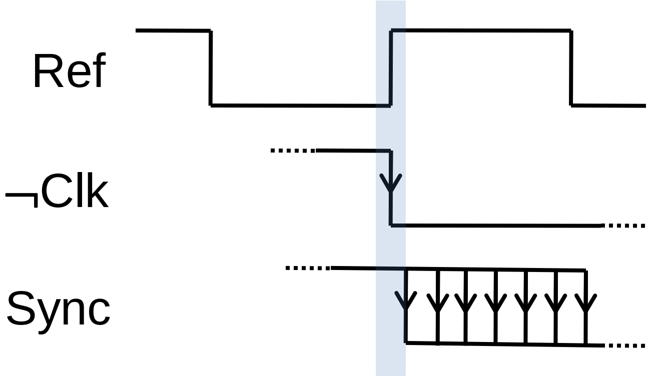




Analysis – Synchronization loop delay

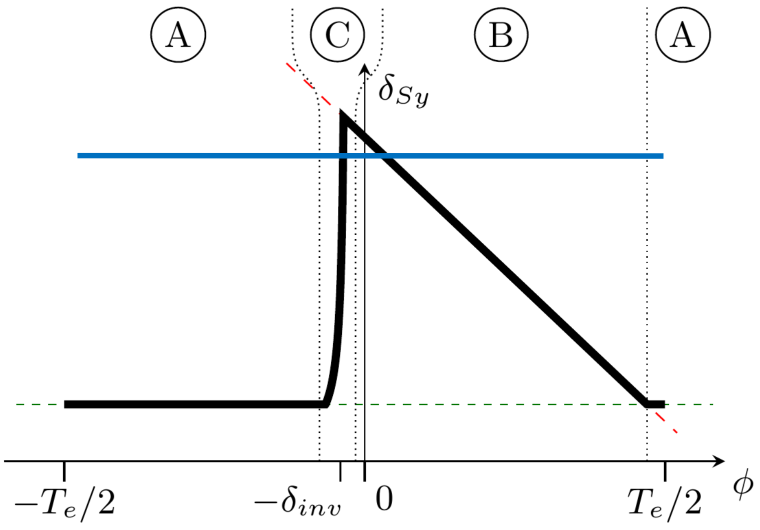
3) Metastability

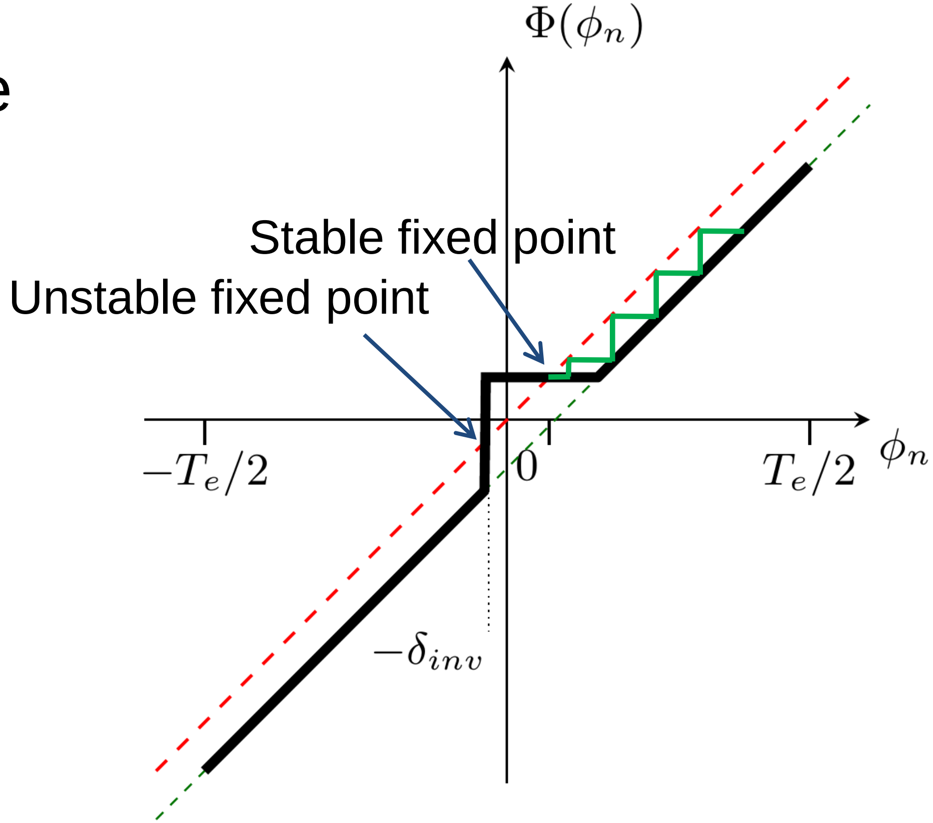




Analysis – The three loops together

- Timing loop adds lower bound
- Recursion to describe synchronization phase





Observations

- When being paused and during resynchronization, the circuit behaves just like the standard pausible clock generator
- In steady state operation, its clock output is synchronous with the reference
- Phase of generated clock vs. reference clock unrestricted in synchronization phase, synchronization time unbounded
- Decoupling and other measures can make desynchronization arbitrarily improbable

The disadvantages

- Requires both a delay line and a clock source
- Both clock sources must be active
- The guaranteed minimum clock period is determined by the delay line – circuit must tolerate it

Conclusion

- The circuit is
  - Simple
  - Value safe
  - When stabilized generating a synchronous clock to the reference
  - Providing the same guarantees as a standard pausible clock

Thank you!