On the Stability and Robustness of Non-Synchronous Circuits with Timing Loops

Matthias Függer, Gottfried Fuchs, Ulrich Schmid and Andreas Steininger

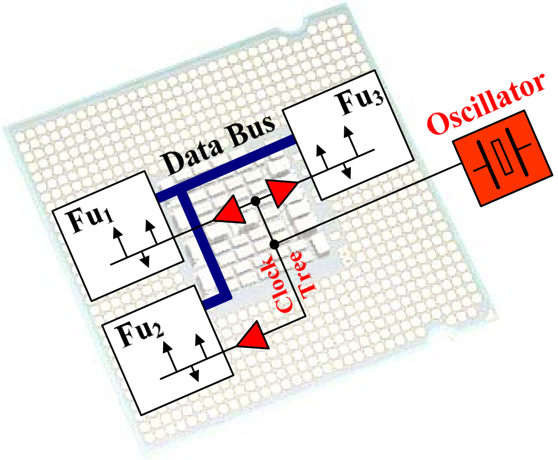
Vienna University of Technology
Embedded Computing Systems Group
{fuegger, fuchs, schmid, steininger}@ecs.tuwien.ac.at

The main messages

- sketch DARTS clocking scheme for SoC
- study systematic timing variations in asynchronous circuits
- influence of k-of-n voting (fault tolerance) on
  - tractability of analysis
  - stability
  - robustness

Clocking in SoCs

synchronous SoC



✓ global synchrony (< 1 tick)

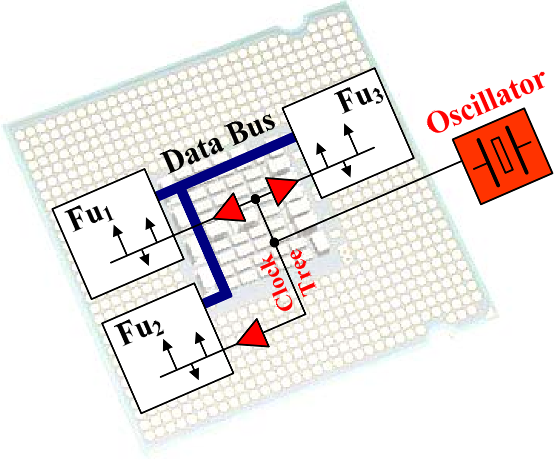
☺ very powerful abstraction
efficient metastability-free communication

💣 single point of failure

☹ nanoscale requires fault tolerance
for clocks as well [Seifert et al.]

Clocking in SoCs

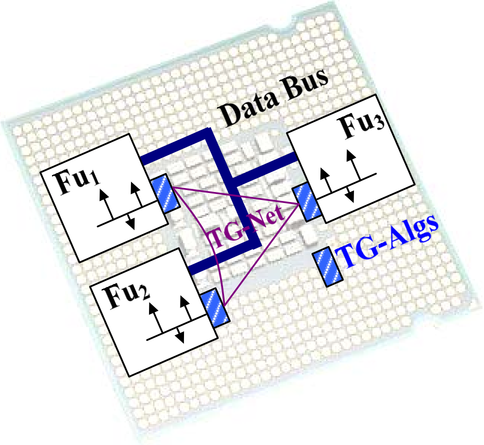
synchronous SoC



✓ global synchrony (< 1 tick)

💣 single point of failure

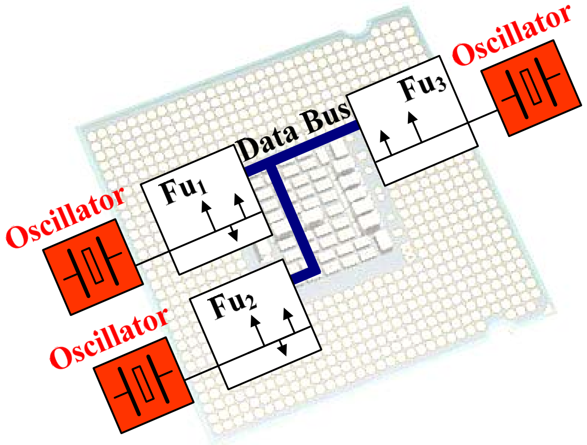
DARTS



✓ global synchrony (> 1 tick)

✓ no single point of failure

GALS

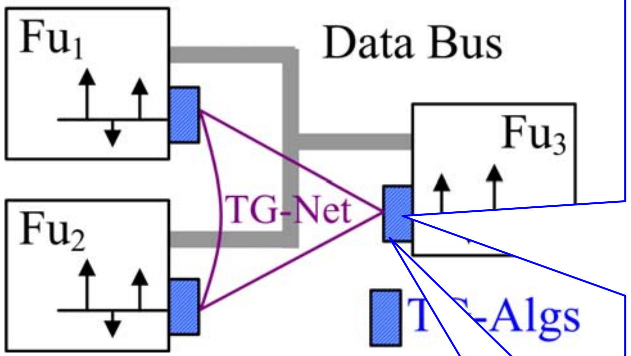


💣 NO (inherent) global synchrony

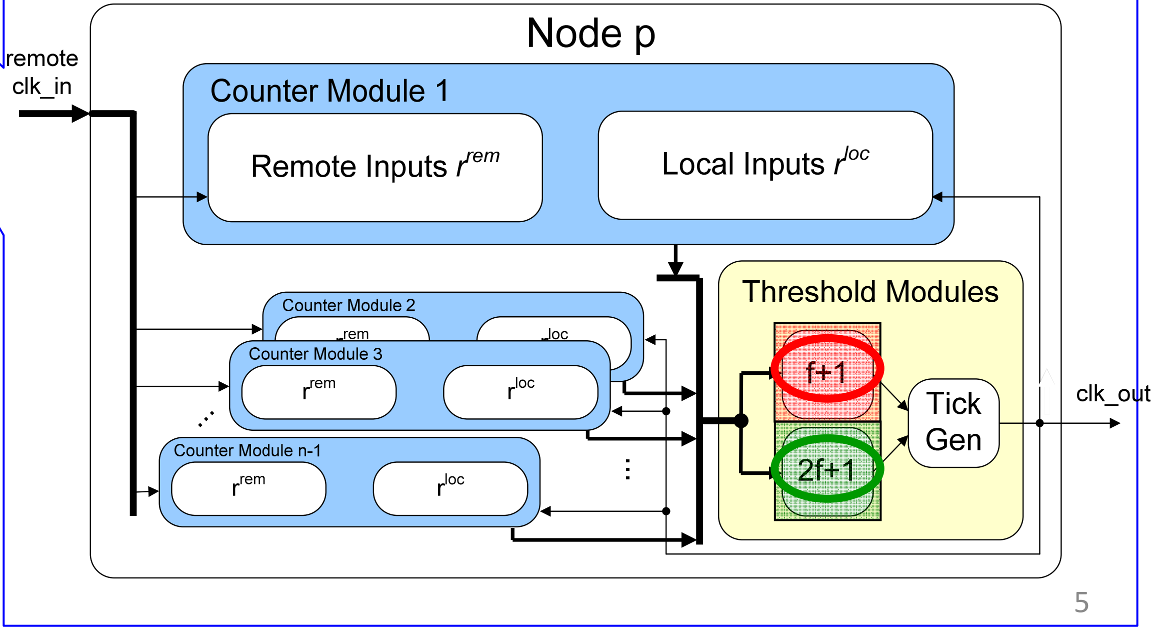
✓ no single point of failure

DARTS Principle & Implementation

Distributed Algorithms for Robust Tick Synchronization



```
(1) Initially:
(2)         send tick(0) to all; clock:= 0;
(3) If received tick(m) from at least f+1 remote nodes and m > clock:
(4)         send tick(clock+1),...., tick(m) to all; clock:= m;
(5) If received tick(m) from at least 2f+1 remote nodes and m >= clock:
(6)         send tick(m+1) to all; clock:= m+1;
```



- asynchronous HW-implementation
- two concurrent rules
- k-of-n- thresholds for fault tolerance

Properties of the DARTS Clock

✓ precision of a few clock cycles & bounded accuracy

- can be guaranteed by formal proof [EDCC06, PODC09]
- on condition of some (weak) routing constraints

✓ a system of 3*f*+2 nodes can tolerate *f* Byzantine faults (nodes and interconnect)

- guaranteed by the same formal proof

? stability of clock frequency

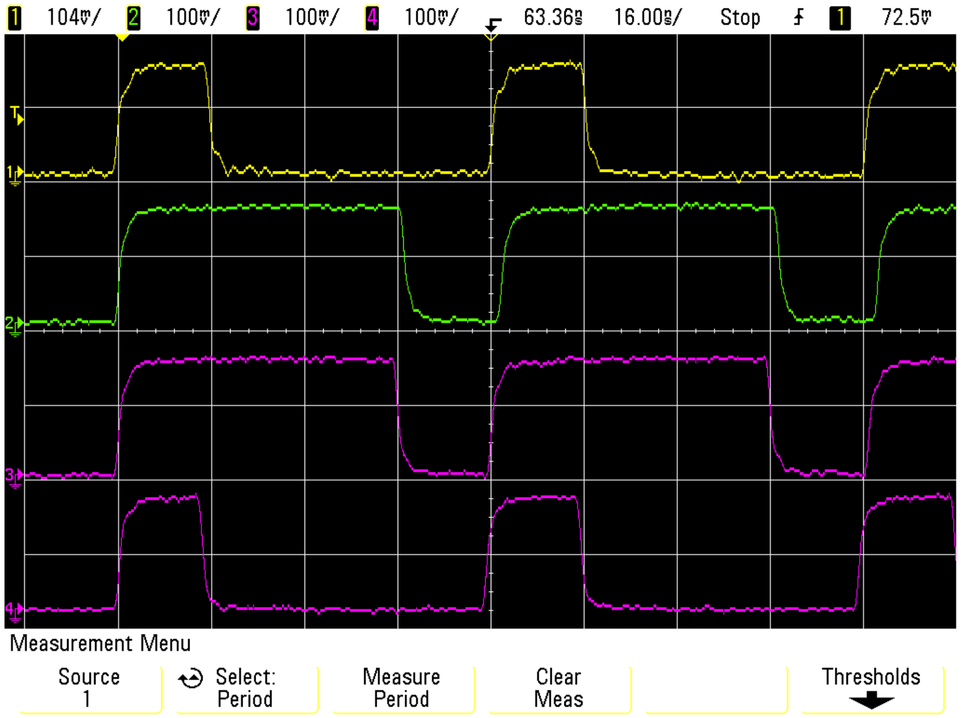
- important for many applications
- BUT: *adaptive* systems cannot be completely *stable*

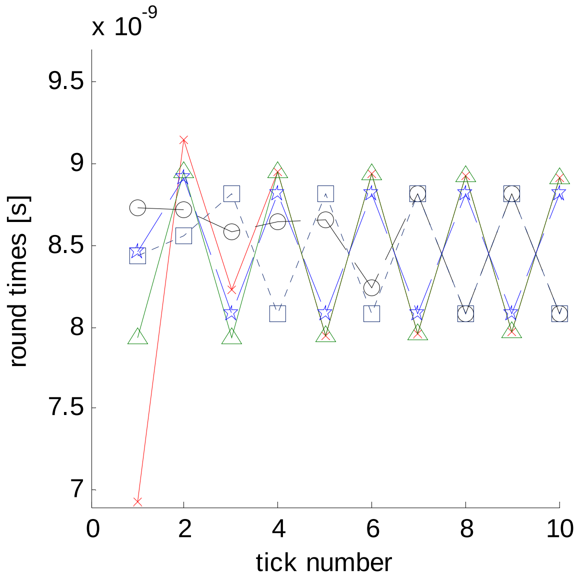
? robustness of clock frequency

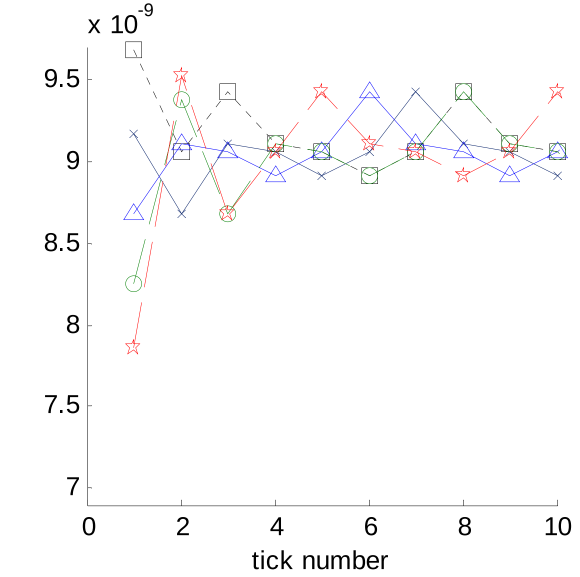
- important for nanotechnology
- variations (tolerances, environment,memb-ship) affect frequency

An Interesting Observation

Setup: - 5 node DARTS system
- pronounced wire delays





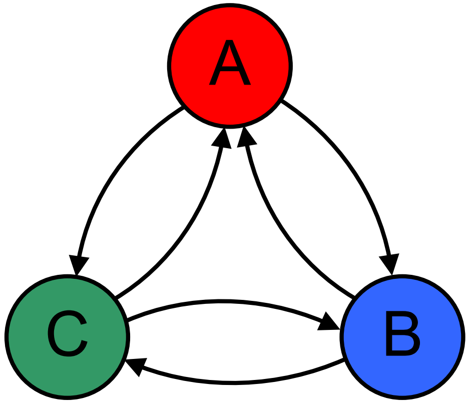


- permanent „oscillation“ of round time
- systematic, not random effect!
- strong dependence on wire delays

A Closer Look…

- theoretical model
  - min/max/+ algebra difference equation

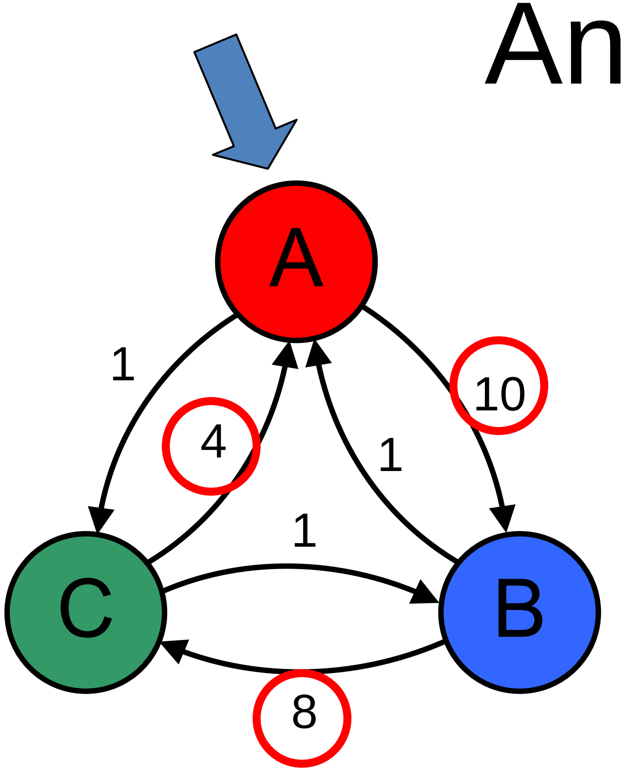
- simplify problem
  - simplify algorithm
    - ✓ wait-for-all instead of k-of-n
    - ✓ no concurrency (one rule)
  - simplify model topology
    - ✓ 3 nodes only

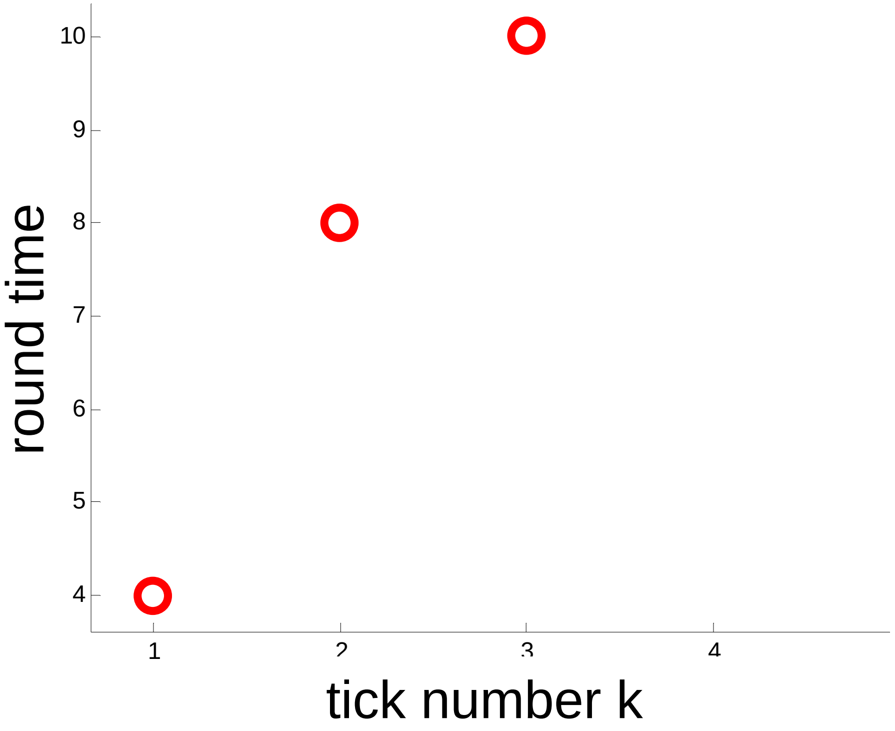


can use max/+ algebra diff equ

from
- nonlinear control theory
- game theory
- asynchronous logic

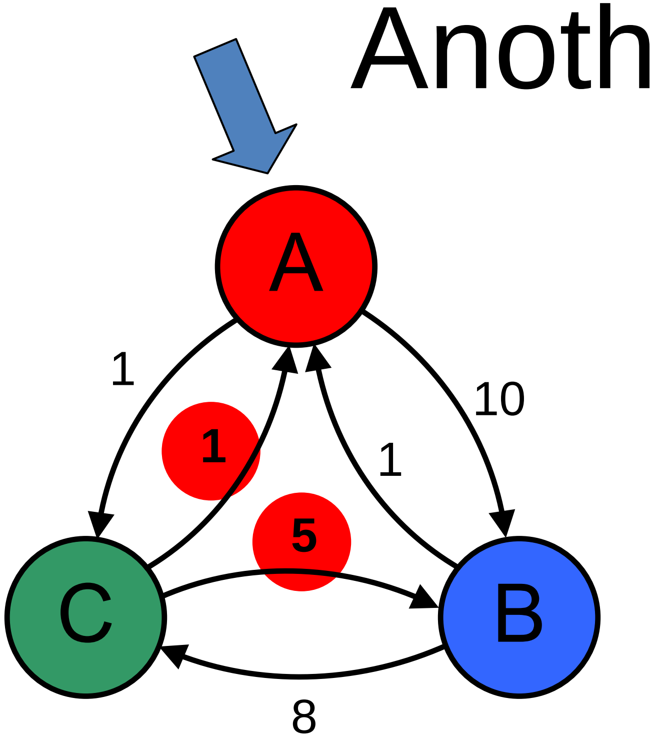
An Example

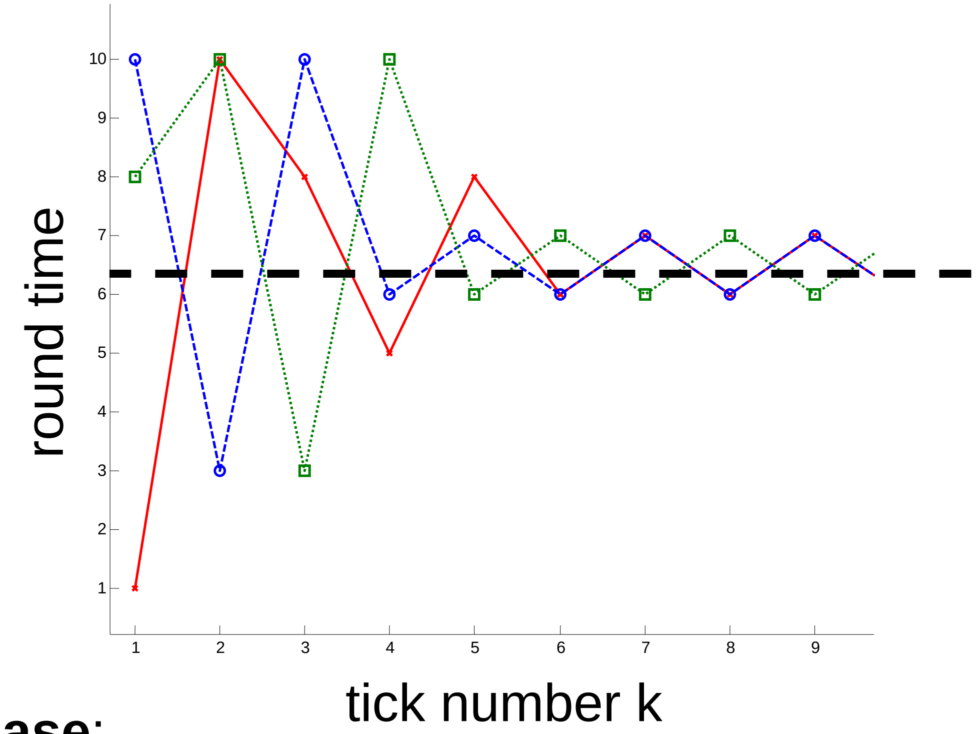




longest paths of length k ending in node P
determine P‘s round times

Another Example





characteristics:

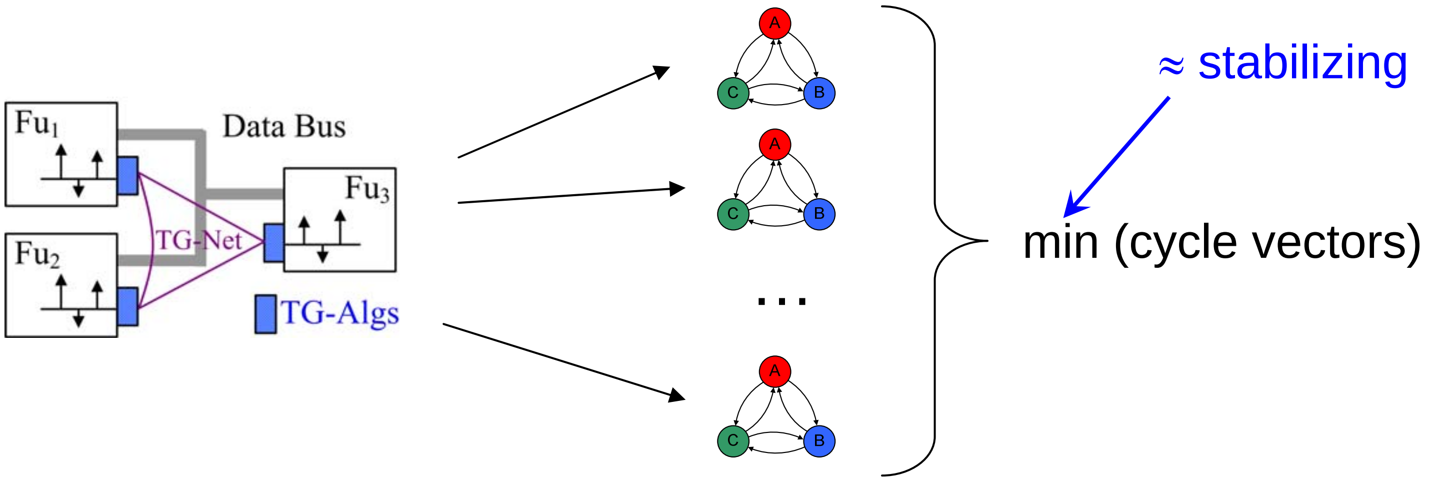
- <u>length</u> of **initial transient phase**:
  depends on delays & initial phase alignment
- <u>mean rate</u> („cycle vector“) during **periodic phase**:
  determined by cycle with maximum mean cycle weight
- <u>peak-to-peak variation</u> (inluding/excluding transient phase)

Is this Relevant for HW Designers?

- timing oscillations in **all asynchronous architectures**
  - with reasonable complexity
  - under some conditions that may not always apply
  - usually not an issue in asynchronous designs
  - still should be known and considered
  - theory largely available (max/+ algebra)
- the specific problems when **fault tolerance** is required
  - concurrent execution of two (or more) rules
  - use of k-of-n thresholds instead of wait-for-all
  - same principle but requires complex min/max/+ algebra

Our Current Status wrt Stability

- ✓ identified appropriate formalism
- ✓ developed simulation environment
- ✓ derived conditions for oscillation for simplified case (max/+)
- ✓ applying „Duality Conjecture“ to explore complex case (min/max/+)

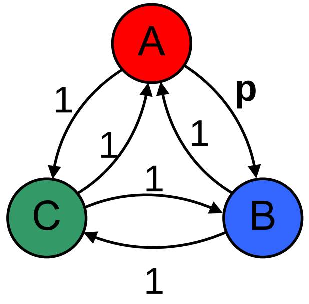


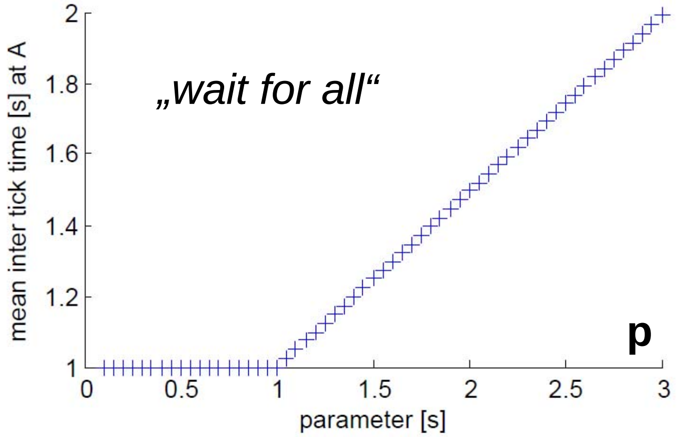
≈ stabilizing

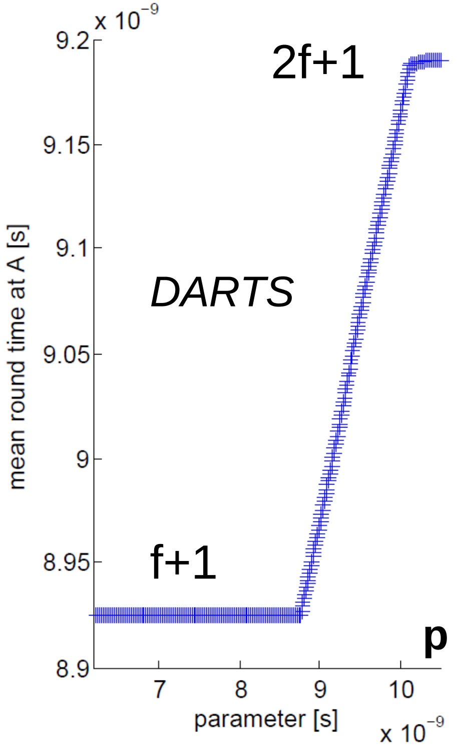
min (cycle vectors)

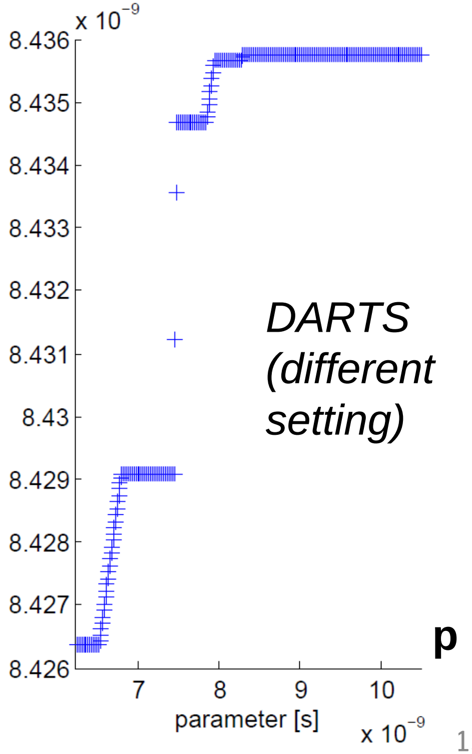
Our Current Status wrt Robustness

- ✓ good robustness against delay variations
- ✓ wait-for-all causes saturation (masks „faster than slowest“)
- ✓ k-of-n causes 2nd saturation („also masks slower than kth“)









Still a Far Way to Go

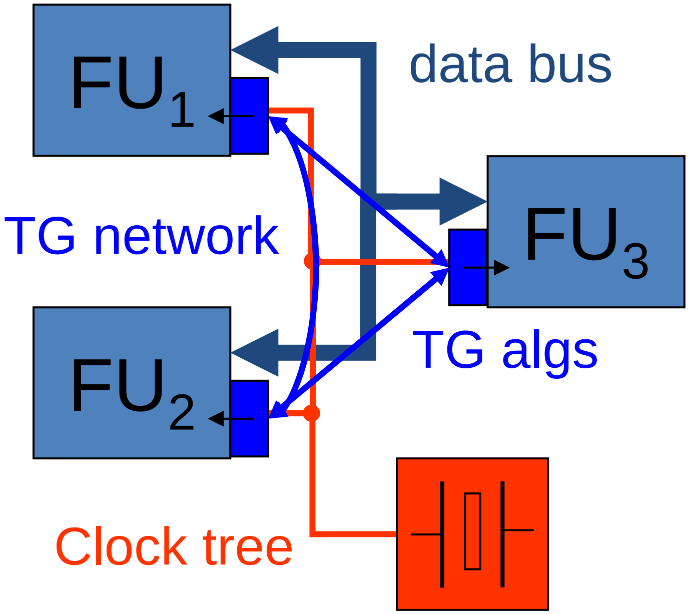
- ➢ need efficient algorithms for min/max/+ algebra to characterize
  - – mean rate
  - – maximum swing
  - – transient phase length
- ➢ explore more complicated cases
- ➢ consider real worls effects
  - – noise and jitter
  - – rise/fall asymmetry, …

Conclusion

- nanoelectronics needs
  - adaptive timing
  - fault tolerance
  - robustness
- round times in asyn loops can show systematic variance
- characterization possible for wait-for-all architectures (max/+ algebra)
- FT solutions need k-of-n architectures
- this severely complicates the analysis (min/max/+ alg.)
- k-of-n improves asynchronous designs‘ robustness

Thank you!

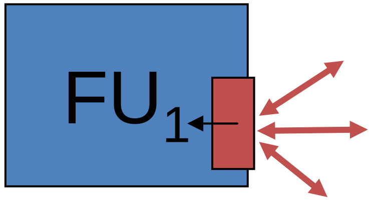
The DARTS Architecture



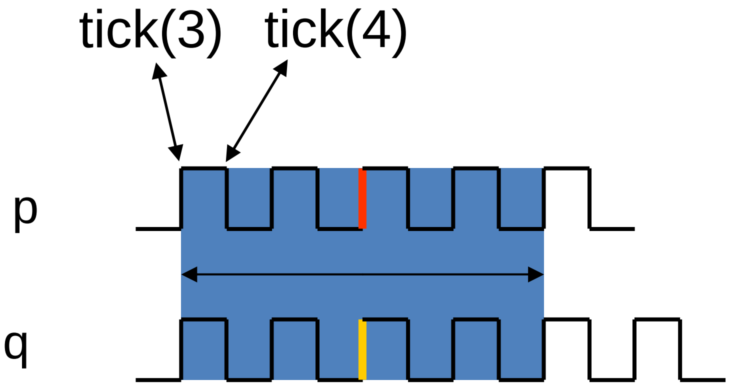
Distributed clock
Synchronous solution

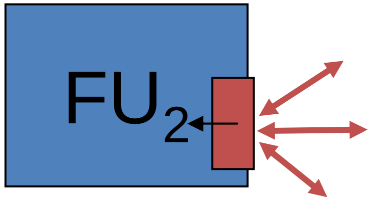
- modules FU_i augmented with simple local clock unit (TG alg)
- TG algs communicate over dedicated bus (TG network) to generate local clocks
- need 3*f*+1 modules to tolerate *f* arbitrary clock faults

What we want…



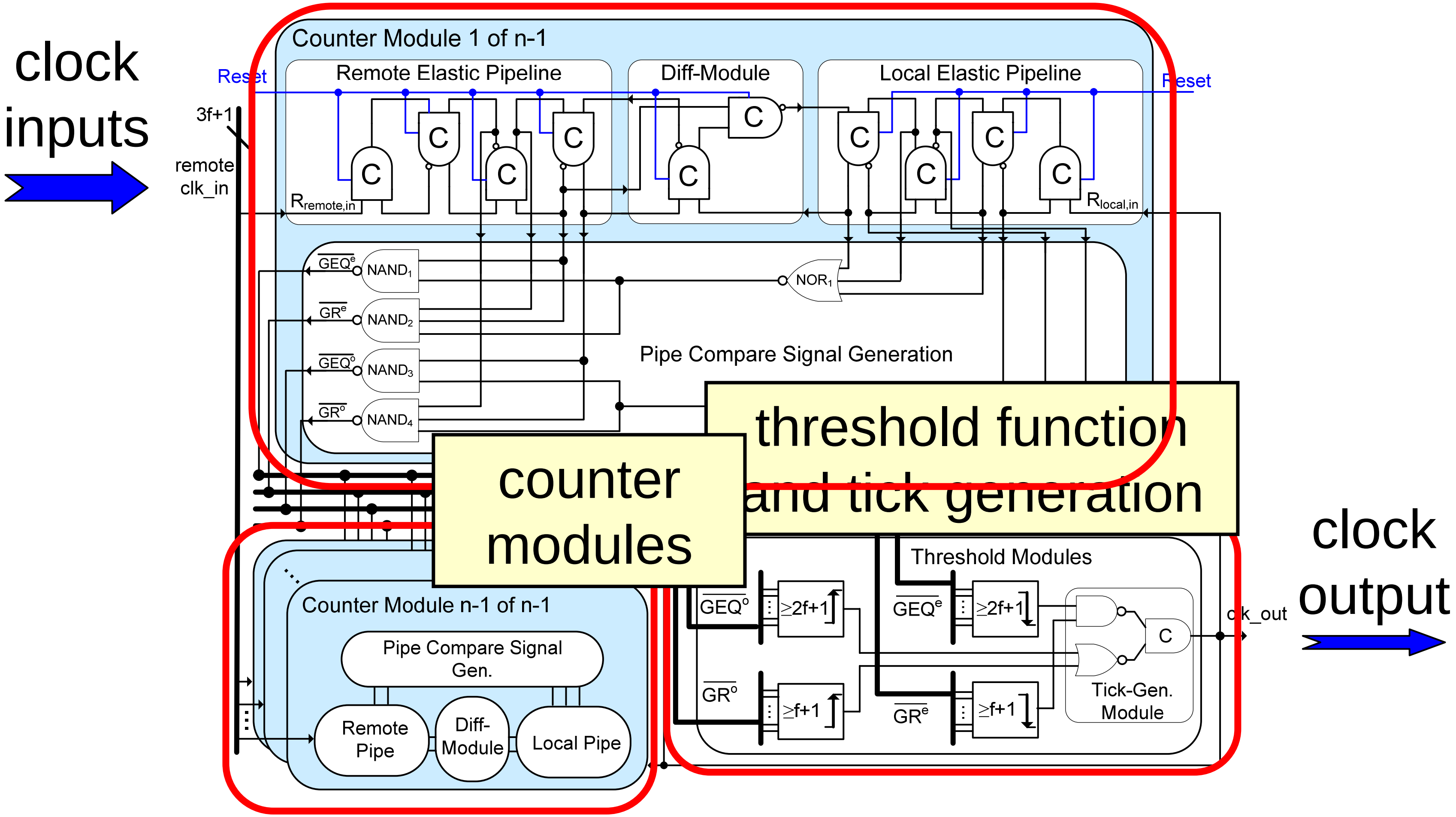
TG alg p





TG alg q

The DARTS clock-generation node



clock inputs

counter modules

threshold function and tick generation

clock output