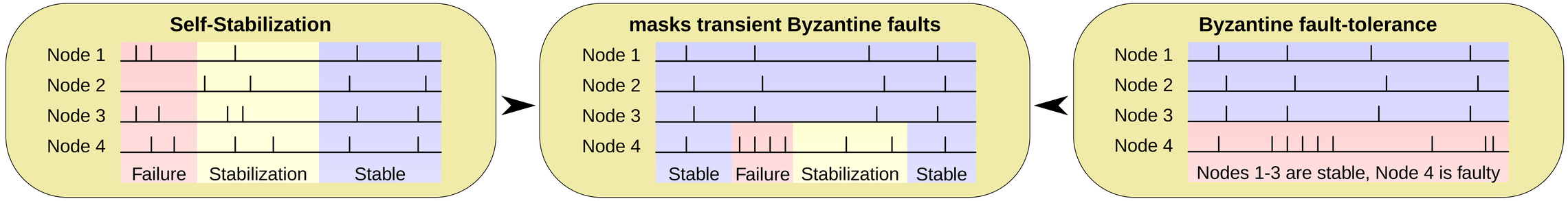
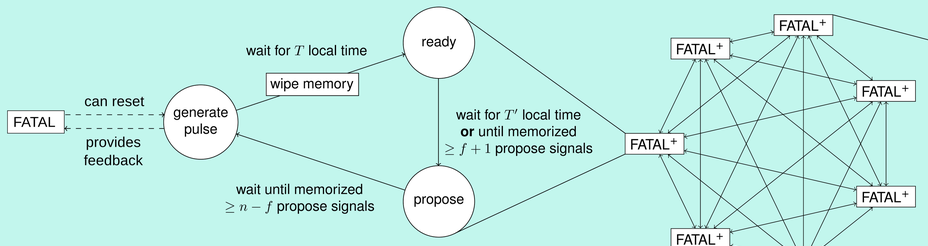
FATAL+HEX:
Fault-tolerant Self-Stabilizing Clock Generation+Distribution

End Goal: highly depenable architecture

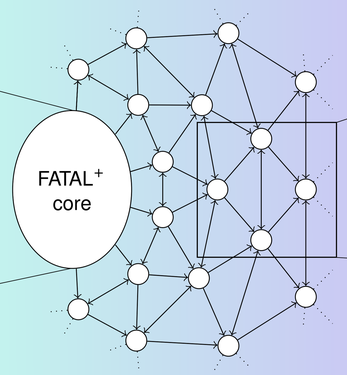


FATAL$^+$: Clock Generation



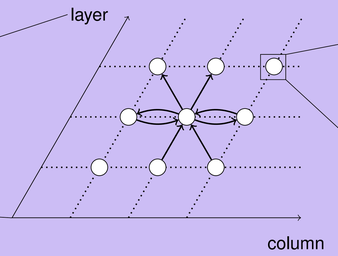
The quick cycle of the FATAL$^+$ protocol, for n nodes and up to $f < n/3$ Byzantine faults. Nodes continuously broadcast whether they are in state propose or not. To estimate the progress of time, nodes are equipped with ring oscillators.

communication topology of FATAL$^+$ core



HEX clock distribution grid

HEX: Clock Distribution



local structure of the HEX grid
layer index = distance from core

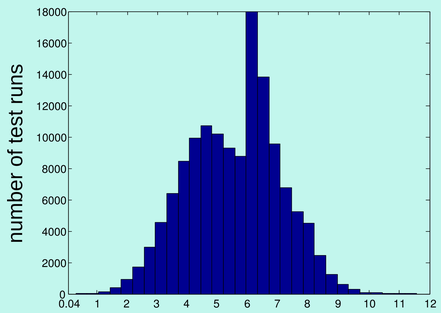
Pulse forwarding algorithm for HEX-nodes.

once *received trigger messages from (left and lower left) or (lower left and lower right) or (lower right and right) neighbors* **do**
  broadcast trigger message; // local clock pulse
  sleep for some time within $[T^-, T^+]$;
  forget previously received trigger messages

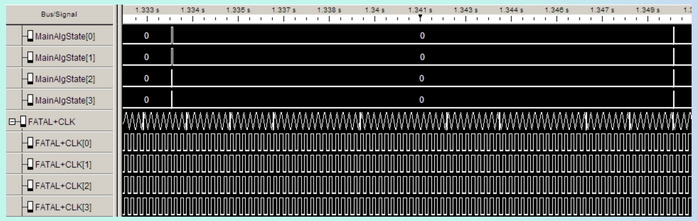
- can tolerate one Byzantine fault in each neighborhood:
  - triggers pulse once both neighbors on previous layer have
  - if one of them failed, neighbors on same layer can fill in
- self-stabilizing: directed pulse propagation "flushs out" false residual states from transient faults
- local oscillators drive high-frequency "fast clocks"
  - resynchronized with every pulse flooded through the grid
  - can be leveraged for fast and efficient communication within a small number of clock cycles

- quick cycle by itself not self-stabilizing
- solution: stabilize by (infrequent) forced reset
  - FATAL generates a reset signal
  - after stabilization:
    - feedback brings reset signal into phase

  => clock generation is not compromised
  - FATAL uses randomization for stabilization



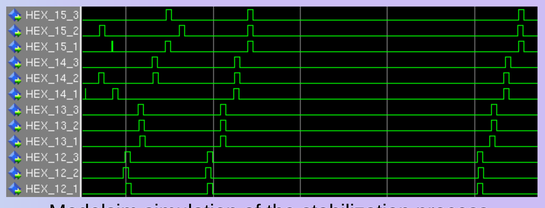
stabilization time (unit depends on technology)



"slow pulses" of the stabilization logic (top)
and high-frequency clock pulses of the quick cycle (bottom)

Proven properties of an n-node FATAL$^+$ system:
- recovery from arbitrarily corrupted system states
- can sustain $f < n/3$ (persistent) Byzantine faults
- stabilization logic succeeds in $\mathcal{O}(n)$ time w.p. $1 - 2^{-n}$

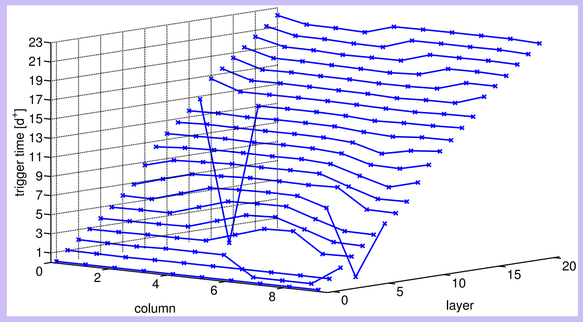
$\Rightarrow$ recovery time $\mathcal{O}(n)$ in the worst case
- recovery in constant time in typical cases:
  - if state of stabilization logic is only partially inconsistent
  - if $n - f$ or more correct nodes are still synchronized



Modelsim simulation of the stabilization process
of the HEX pulse forwarding mechanism

Proven properties of an L-layer HEX grid:
- assuming link delays in $[d^-, d^+]$ and fast clocks of nominal rate r
  - neighbors' clocks differ by $\mathcal{O}(r(d^+ - d^-)^2 L/d^+)$ ticks
  - each fault increases this by $\mathcal{O}(rd^+)$ ticks
- stabilization within $\mathcal{O}(L)$ HEX pulses

Simulations:
- excellent average-case performance:
  - average clock differences much better than worst case
  - tolerance of multiple randomly distributed faults



pulse propagation wave in a simulated HEX grid
with multiple faults (fake trigger time 0)

Future Work

- develop novel hardware building blocks to:
  - increase operational frequency
  - have cheap self-stabilizing low-level building blocks
- bottom-to-top formal verification of FATAL+HEX compound system
- provide fault-tolerant communication and application logic
- build and test fully functional ASIC prototype

Danny Dolev	Hebrew University of Jerusalem
Matthias Függer	Vienna University of Technology
Markus Hofstätter	Vienna University of Technology
Christoph Lenzen	Massachusetts Institute of Technology
Martin Perner	Vienna University of Technology
Markus Posch	Vienna University of Technology
Ulrich Schmid	Vienna University of Technology
Martin Sigl	Vienna University of Technology
Andreas Steininger	Vienna University of Technology

Fault-tolerant Algorithms for Tick-Generation in Asynchronous Logic: Robust Pulse Generation
Under submission to Journal of the ACM (JACM), first revision.
HEX: Scaling Honeycombs is Easier than Scaling Clock Trees
25th Symposium on Parallelism in Algorithms and Architectures (SPAA), 2013.
FATAL$^+$: An Ultra-Robust Clocking Scheme for Systems-on-Chip
Under submission to Journal of Computer and System Sciences (JCSS).
Byzantine Self-Stabilizing Clock Distribution with HEX: Implementation, Simulation, Clock Multiplication
6th Conference on Dependability (DEPEND), 2013.
Efficient Construction of Global Time in SoCs despite Arbitrary Faults
16th Euromicro Conference on Digital System Design (DSD), 2013.
Fault-tolerant Algorithms for Tick-Generation in Asynchronous Logic: Robust Pulse Generation
13th Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS), 2011.