Architecture and Design Methodology for Autonomic System-on-Chip (ASoC)

A. Bernauer, W. Rosenstiel
Wilhelm-Schickard Institute for Informatics
Computer Engineering
University of Tübingen

O. Bringmann, A. Braun
Forschungszentrum Informatik
Karlsruhe

A. Bouajila, J. Zeppenfeld,
W. Stechele, A. Herkersdorf
Institute for Integrated Systems
Munich University of Technology

asoc@informatik.uni-tuebingen.de

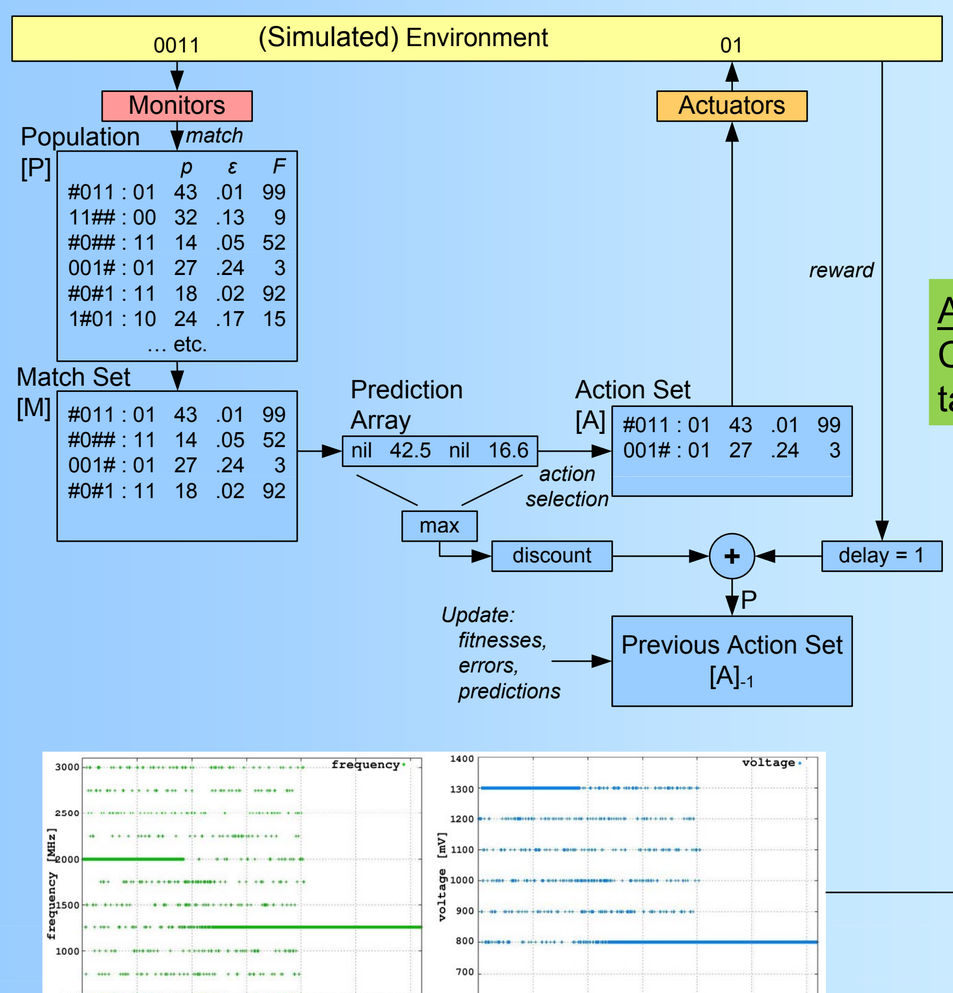
Learning for Autonomic System-on-Chip

XCS

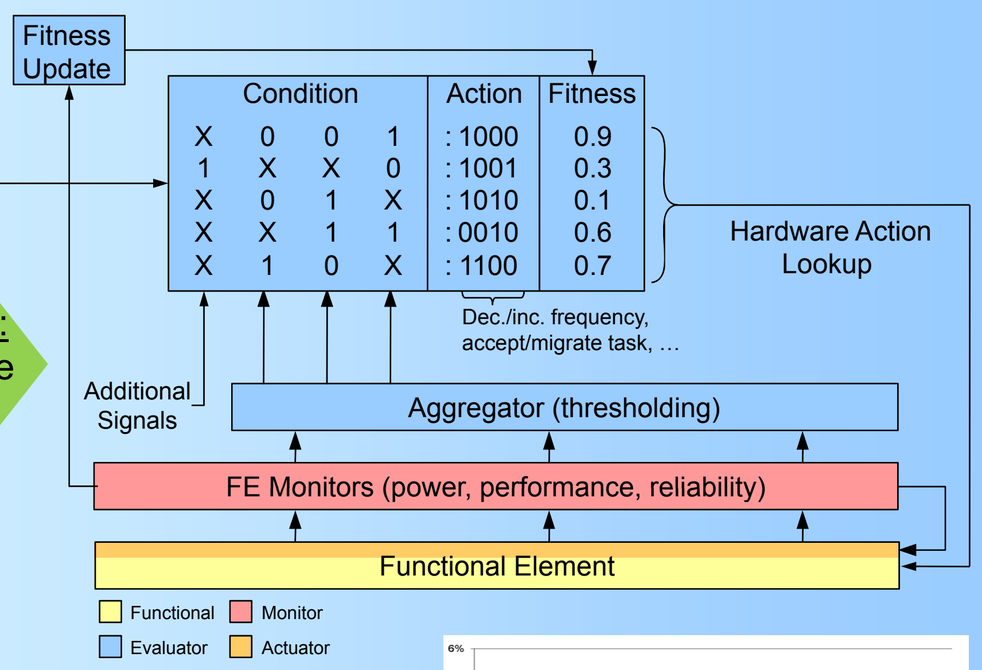
- At design time
- Completely in Software
- Full power of software learning

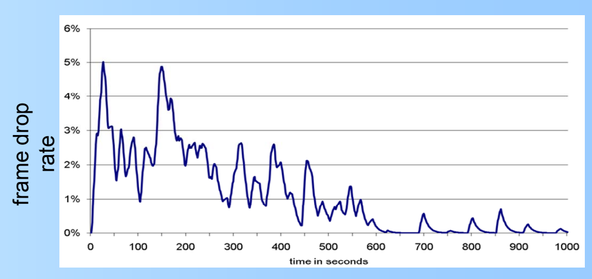
Learning Classifier Table

- At run time
- Mostly in Hardware
- Design time rules adapted to actual hardware



After design and training:
Compression of XCS rule table to fit in LCT



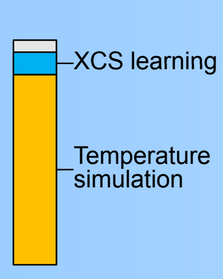


Best of both worlds:
✓At design time: powerful software learning with XCS
✓At run time: rules adapted to actual hardware

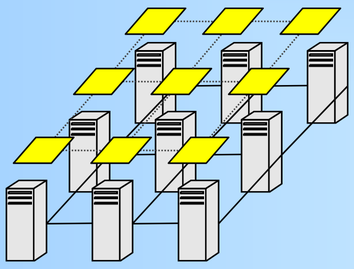
Scalability of ASoC

Developing ASoCs with many cores

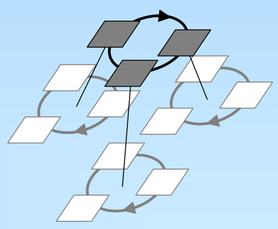
- How to efficiently train XCS for systems with many cores?
- What is the effect of many distributed evaluators (LCTs)?



Reduce simulation time



Transfer knowledge during design-time learning

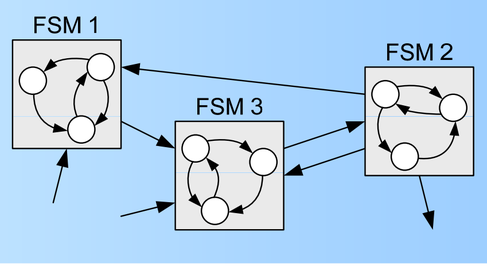


Hierarchical AE interconnect

Robustness of the Autonomic Layer

Protecting the Autonomic layer

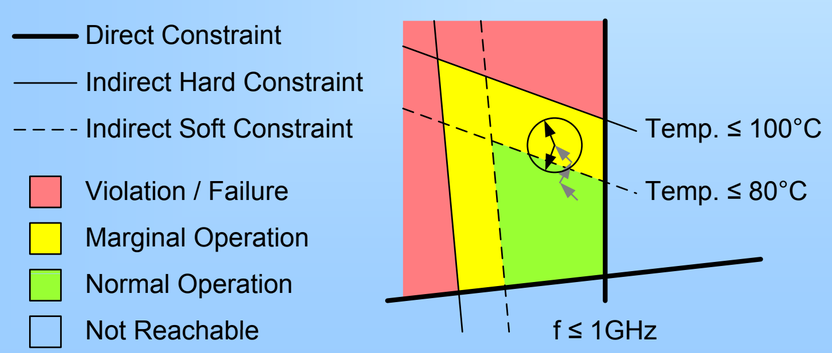
- Adding error monitors in the autonomic layer
- One AE's LCT will supervise neighboring AEs.
- Study how to protect the different communicating FSMs in the Autonomic layer
- Adding AE monitoring interface to an organic software middleware layer



Satisfying Constraints

Guarding against undesirable conditions

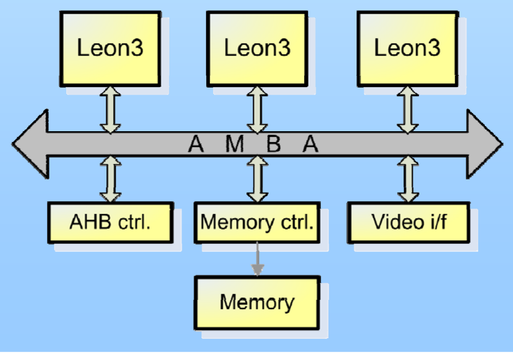
- Violating soft constraints leads to performance degradation.
- Violating hard constraints could lead to a system failure.

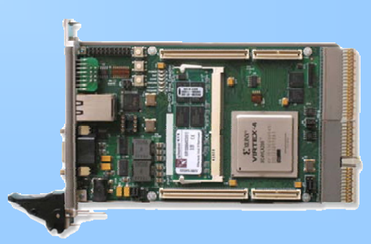


- Add guard to detect and prevent violation of constraints.

Use-Case Exploration in FPGA

- Real-word applications running on Leon3-based prototype:
  - Networking (varying inter-arrival rate of packets)
  - Autovision (video frames >300 Kb @ 25 frames/sec)
- Check that the ASoC FPGA prototype performs self-optimization, self-correction and learning.





This work is partially funded by Deutsche Forschungsgemeinschaft (DFG) within the priority program "Organic Computing" (SPP 1183) under the grants HE 4584/3-2 and RO 1030/14-2.