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

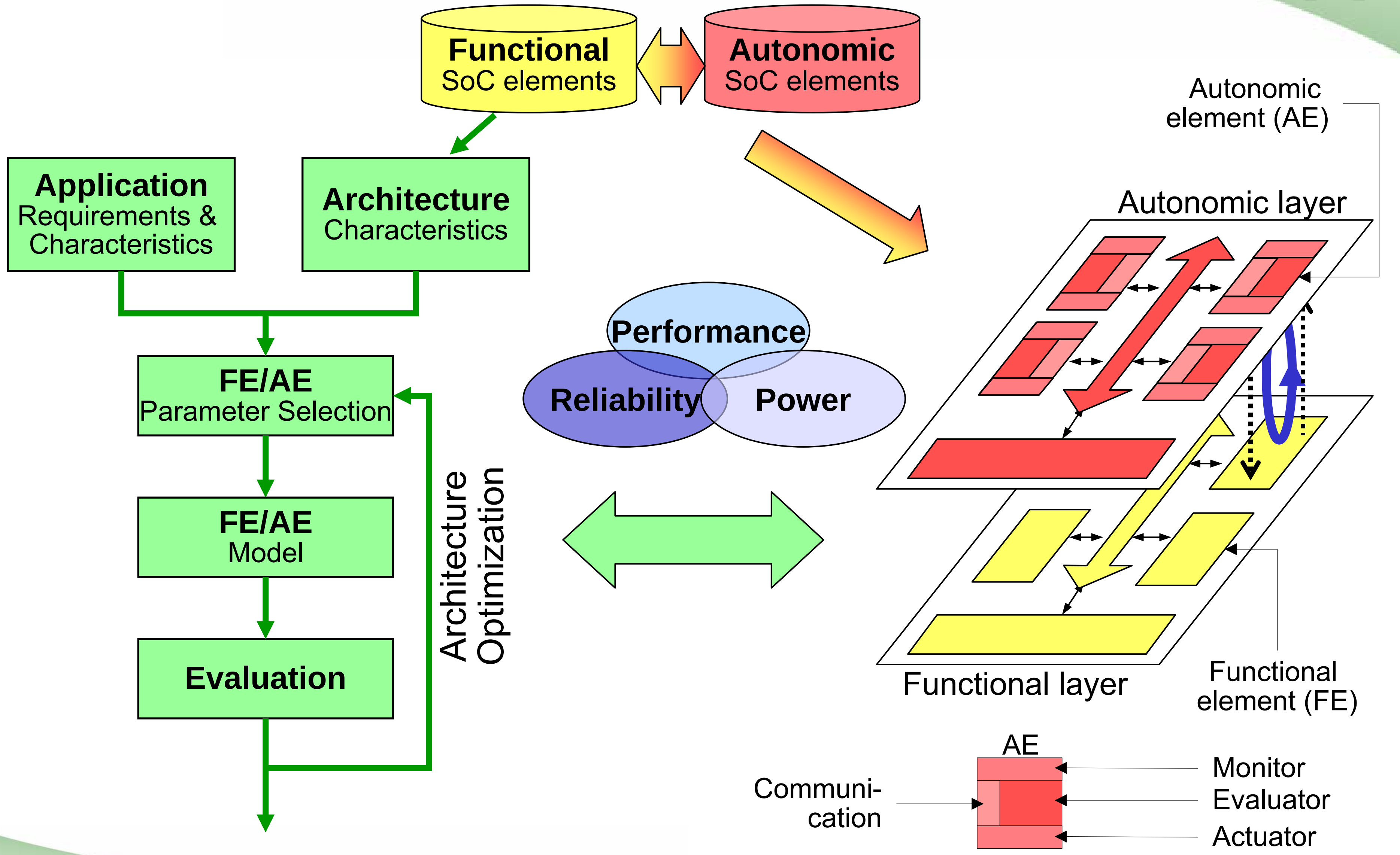
Andreas Bernauer[1], Abdelmajid Bouajila[2], Oliver Bringmann[3], Walter Stechele[2],
Andreas Herkersdorf[2], Wolfgang Rosenstiel[1,3]

[1]Universität Tübingen

[2]Technische Universität München

[3]FZI - Forschungszentrum Informatik

# Project reminder



Functional SoC elements

Autonomic SoC elements

Application Requirements & Characteristics

Architecture Characteristics

FE/AE Parameter Selection

FE/AE Model

Evaluation

Architecture Optimization

Performance

Reliability

Power

Autonomic element (AE)

Autonomic layer

Functional layer

Functional element (FE)

AE

Communi-cation

Monitor

Evaluator

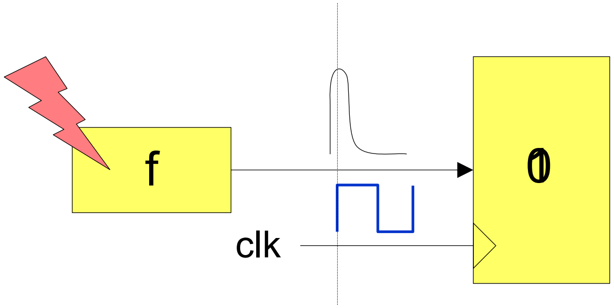
Actuator

# Overview

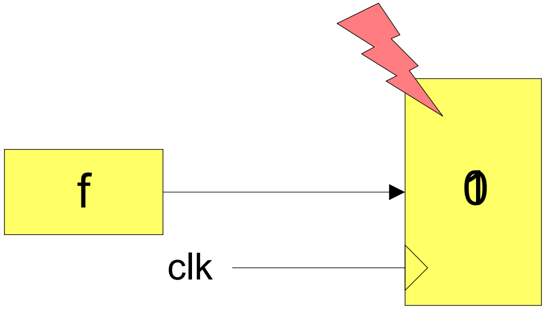
- State of the art
- Error detection techniques (monitors)
- Self-healing CPU-pipeline (monitor & actuator)
- Reliability estimation (evaluator)
- Summary
- Future work

# Fault cause: particle strike

- Two models
  - Single event transient (SET): gate output changed

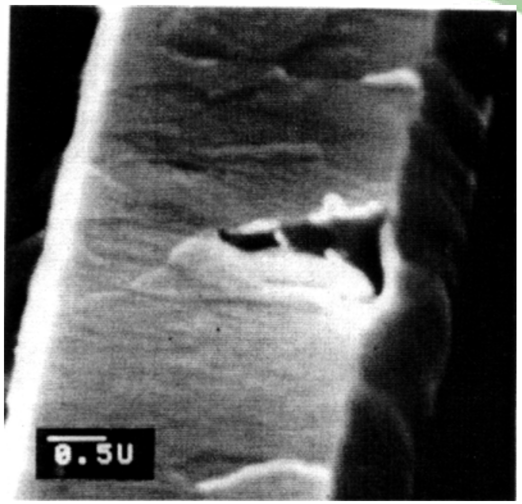


  - Single event upset (SEU): flip-flop value changed



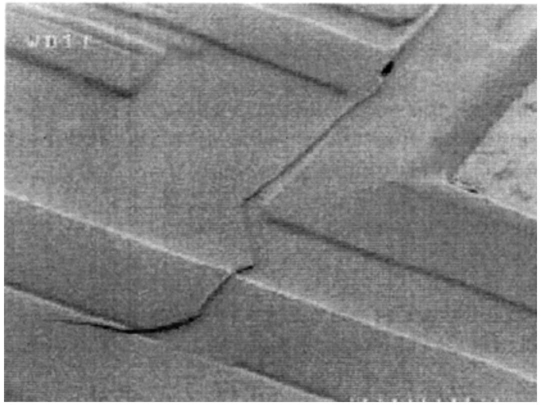
# Other common fault causes

- Migration of metal atoms
  - Electromigration: Atoms hit by electrons
  - Stress migration: Mechanical stress
  - → resistance increase and shorts
- Temperature cycling and thermal shock
  - Accumulated deformations
  - → cracks and lifts
- Time-dependent dielectric breakdown
  - Conductive path in the dielectric
  - → lowered thresholds



© JEDEC

Metal line with stress migration voids



© JEDEC

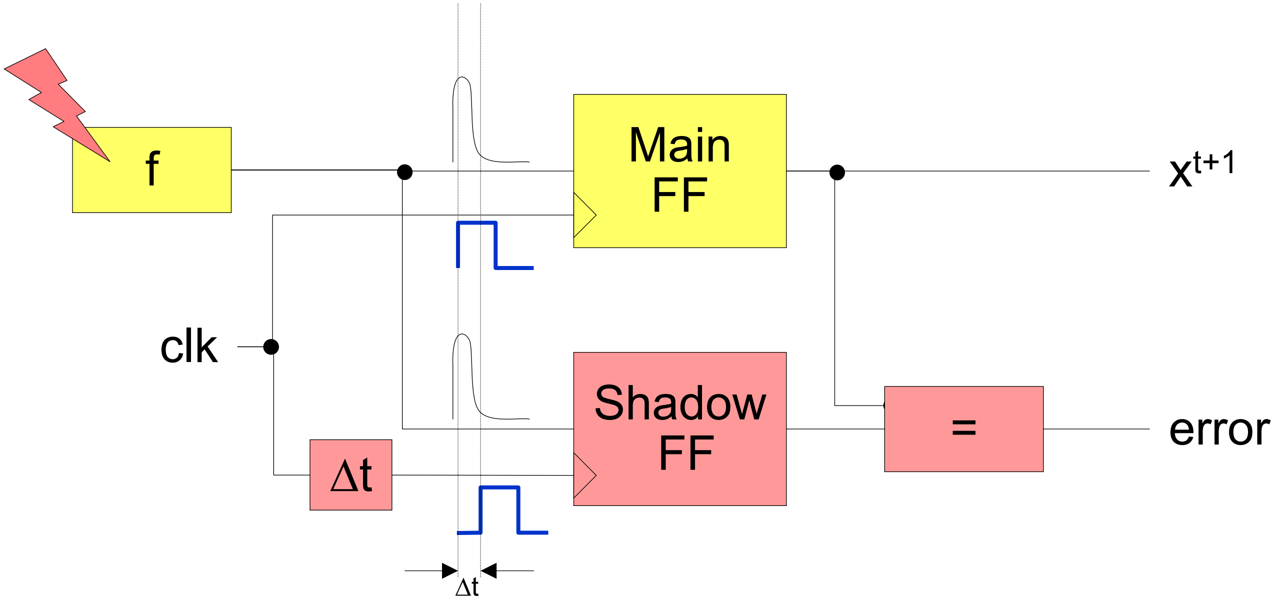
Temperature cycle damage

# Fault models

| Class | Causes | Models |
|---|---|---|
| Transient error | Particle strike | SET: single event transient<br>SEU: single event upset |
| Timing error | E.g. increased line resistance | Timing constraints violations |
| Permanent error | Shorts, breaks | stuck-at-1/-0,<br>stuck-open, stuck-on |
| Design error | E.g. incompetence, logic errors | |

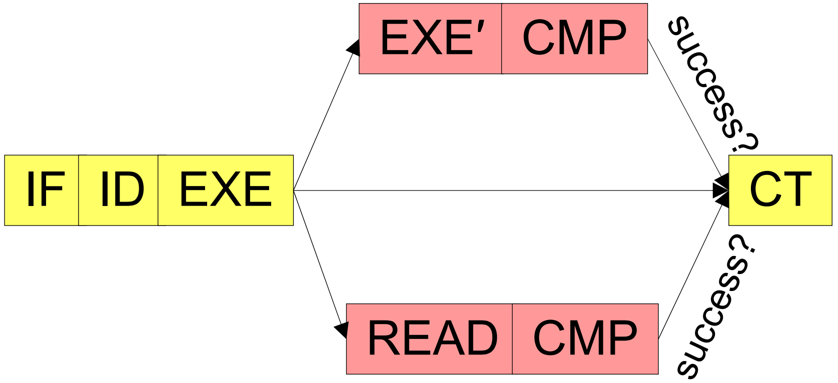
# Error detection techniques

- Time redundancy [Nic99]

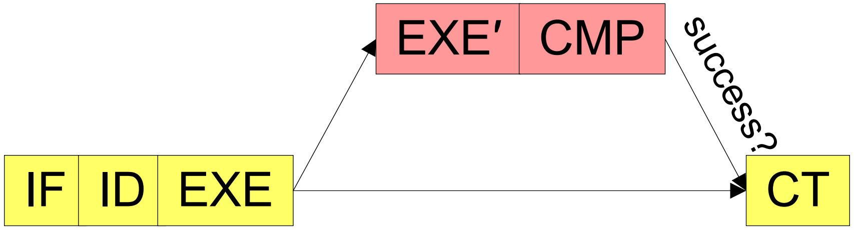


# DIVA technique

- DIVA [Austin01]
  - EXE′ re-executes
  - READ re-reads
  - Less area overhead than duplication but comparable fault coverage



- Modified DIVA [BICC06]
  - Improvement: No READ to avoid hazards



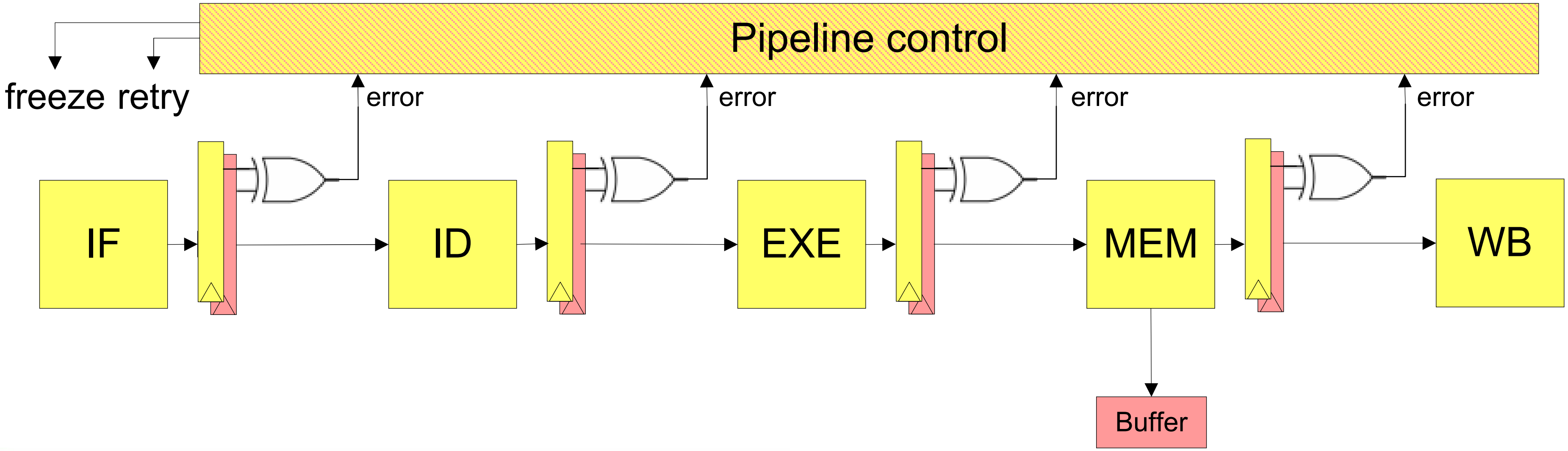
# Error detection techniques

| | Technique | Transient | Timing | Permanent | Design | IP reusable? |
|---|---|---|---|---|---|---|
| Hardware | e.g. duplication | √ | √ | √ | √ | Yes, systematically |
| Information | Self-checking operators [Nic93] | | | √ | | No |
| | Path-fault secure circuits [Tuba97] | √ | √ | √ | | No |
| Time | Nicolaidis flip-flop [Nic99] | √ | √ | | | Yes, systematically |
| | Razor [Austin03] | | √ | | | Yes, systematically |
| Mixed | DIVA [Austin01] | √ | √ | √ | √ | Yes, but not systematically |

Published at BICC06

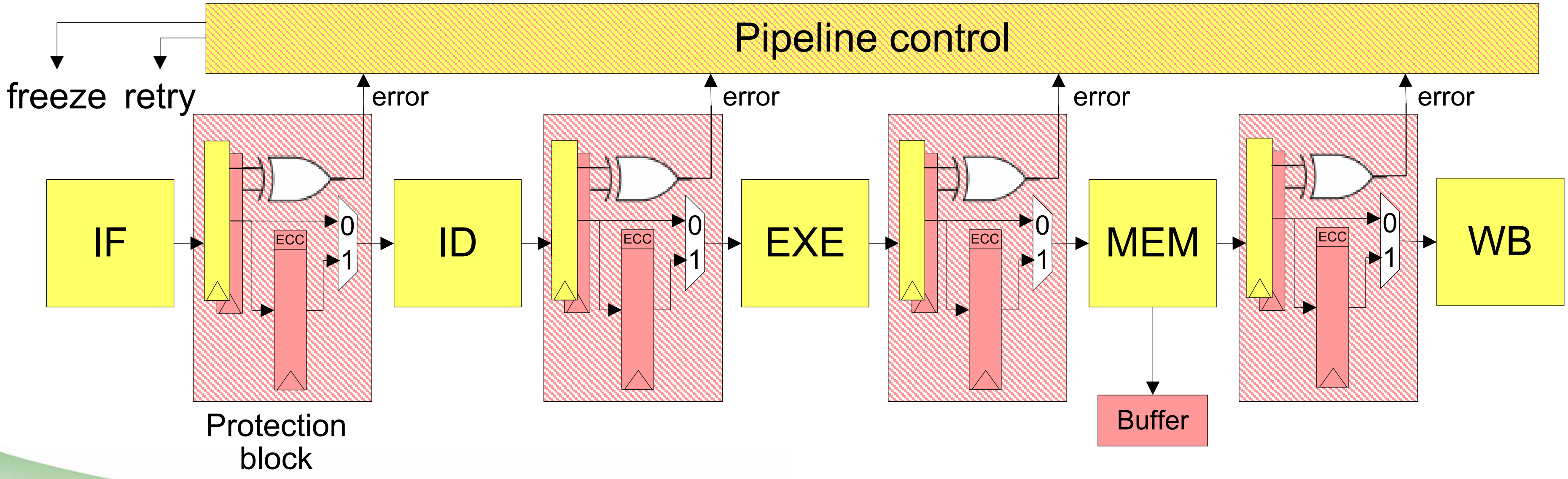
# Protecting a CPU pipeline against transient and timing errors

- Prior-art:
  - Transient and timing error detection [Chardonnereau02]
  - Timing error detection and correction with Razor flip-flops [Austin03]
  - Error correction with pipeline flushing
- Detection at the next cycle → pipeline stage input registers overwritten

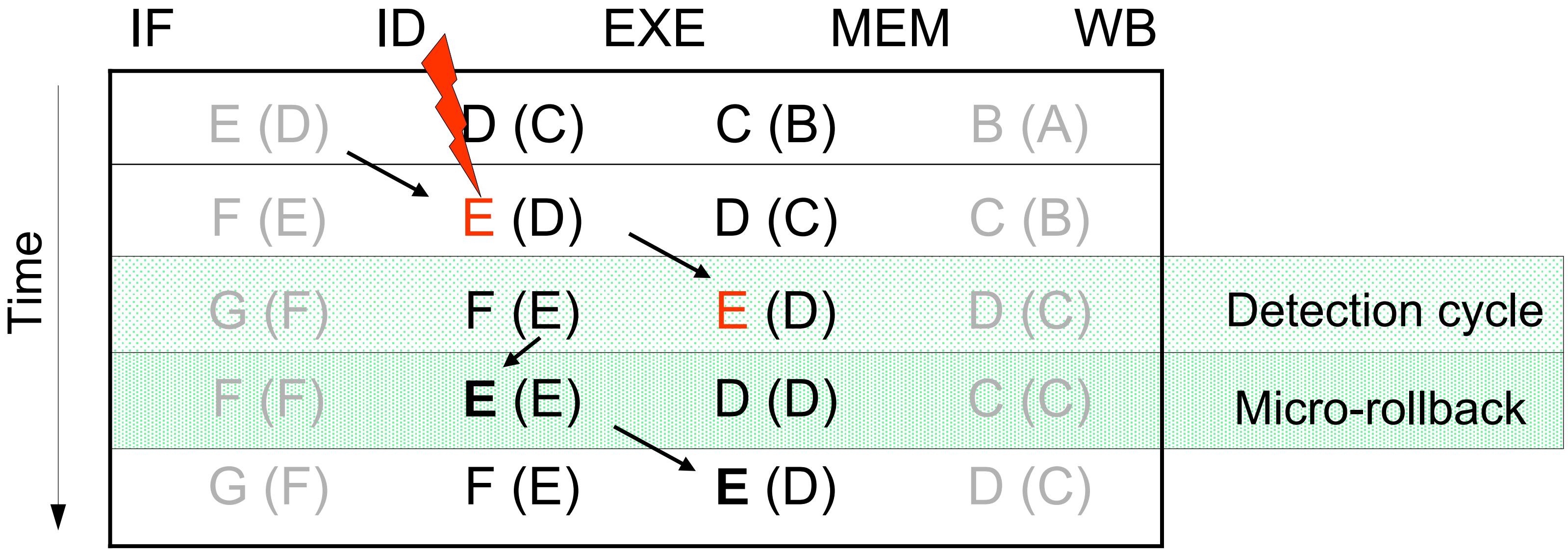


# Self-healing CPU pipeline [VLSI06]

- Error detection using Nicolaidis flip-flops
- History registers keep track of latest pipeline stage registers
- No pipeline flushing necessary → fixed 2-cycle penalty



# Error handling mechanism

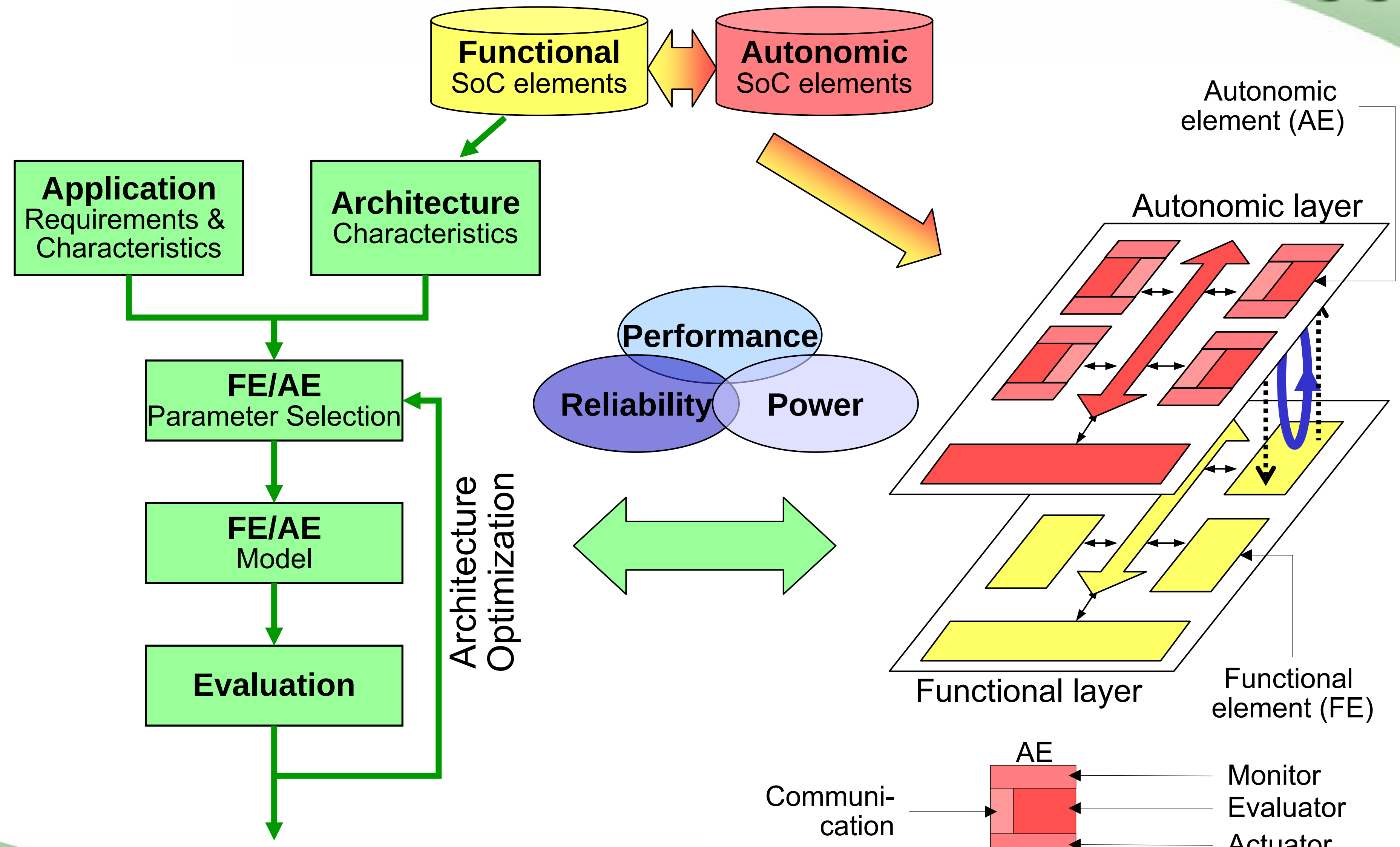


Time

| IF | ID | EXE | MEM | WB | |
|---|---|---|---|---|---|
| E (D) | D (C) | C (B) | B (A) | | |
| F (E) | E (D) | D (C) | C (B) | | |
| G (F) | F (E) | E (D) | D (C) | Detection cycle | |
| F (F) | **E** (E) | D (D) | C (C) | Micro-rollback | |
| G (F) | F (E) | **E** (D) | D (C) | | |

- Implemented in Leon2 pipeline (VHDL)
- Simulation with ModelSim (fault injections) confirms
  - Detection and correction of transient and timing errors
  - Fixed 2-cycle penalty
- 23% area overhead on a Virtex-II-Pro FPGA
  - Area of XOR-trees in FPGA

# Project overview



**Functional**
SoC elements

**Autonomic**
SoC elements

**Application**
Requirements & Characteristics

**Architecture**
Characteristics

**FE/AE**
Parameter Selection

**FE/AE**
Model

**Evaluation**

Architecture Optimization

**Performance**

**Reliability**

**Power**

Autonomic element (AE)

Autonomic layer

Functional layer

Functional element (FE)

AE

Communication

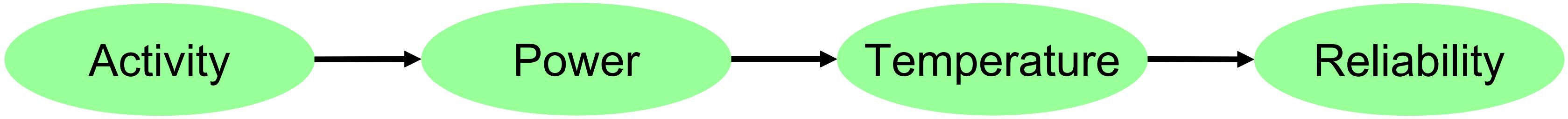
Monitor

Evaluator

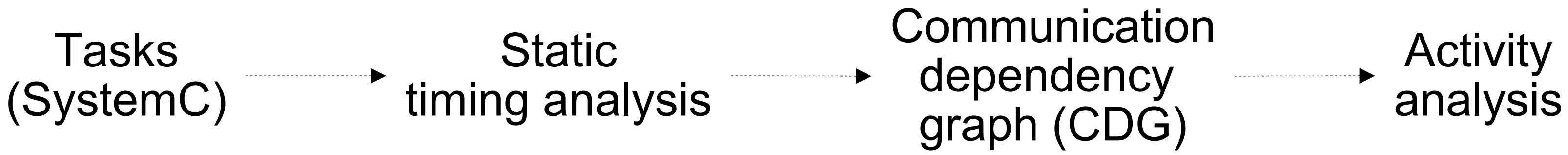
Actuator

# Design time reliability estimation

- Factors influencing reliability:

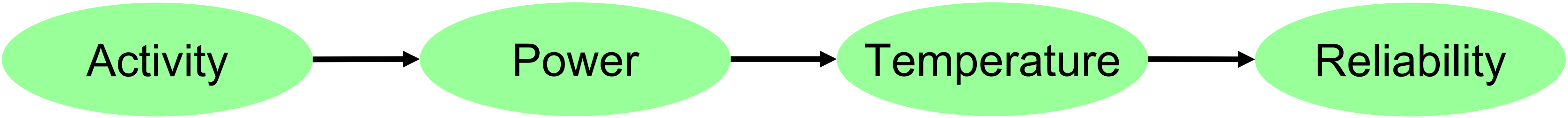


- → Activity



- Activity → Power: power state machines
- Power → Temperature
  - HotSpot tool [SSS04] : accurate run time temperature estimates
  - Inputs: geometry, power consumption, material constants
  - Outputs: spatial and temporal temperature distribution

# Design time reliability estimation



- Temperature → Reliability
  - Arrhenius relationship between failure rate $\lambda$ and temperature T

$$\lambda(T) = \text{const} \cdot e^{-\frac{E_a}{kT}}$$

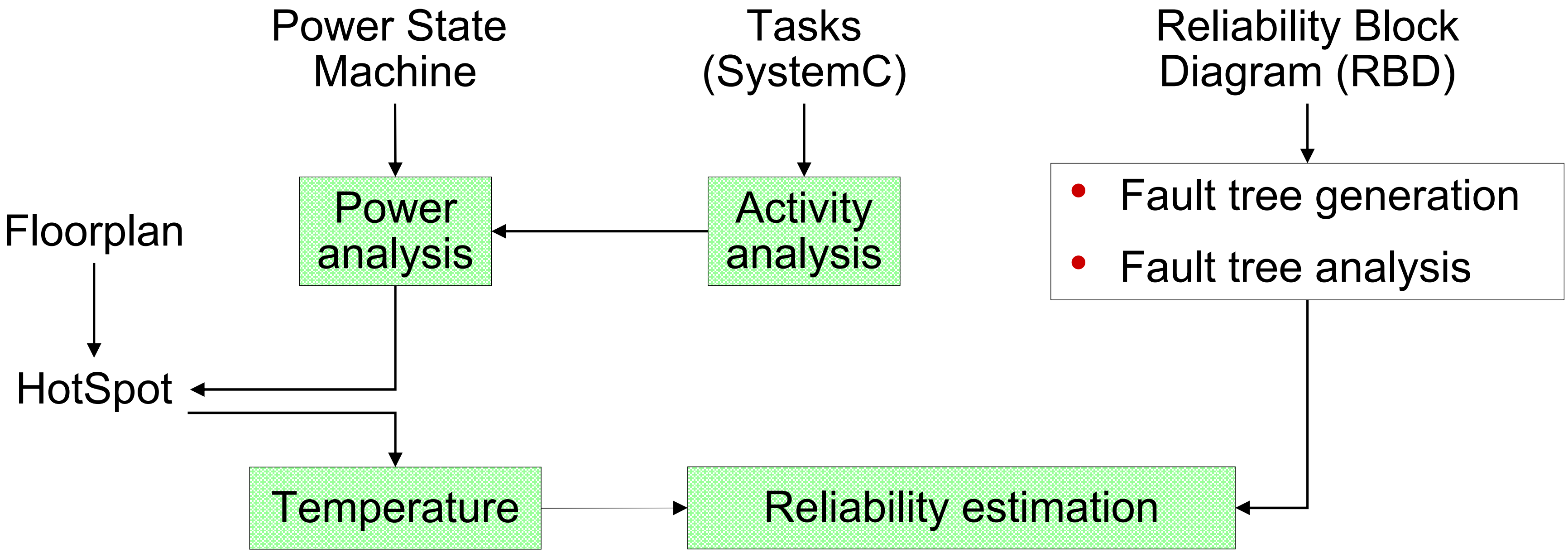
| | |
|---|---|
| const | material constant |
| $E_a$ | activation energy for electromigration |
| k | Boltzmann‘s constant |
| T | absolute temperature |

  - Coffin-Manson equation for average number of temperature cycles $N_f$ until failure

$$N_f = C_0(\Delta T)^{-q}$$

| | |
|---|---|
| $C_0$, q | material constants |
| $\Delta$T | size of temperature cycle |

# Design time reliability estimation



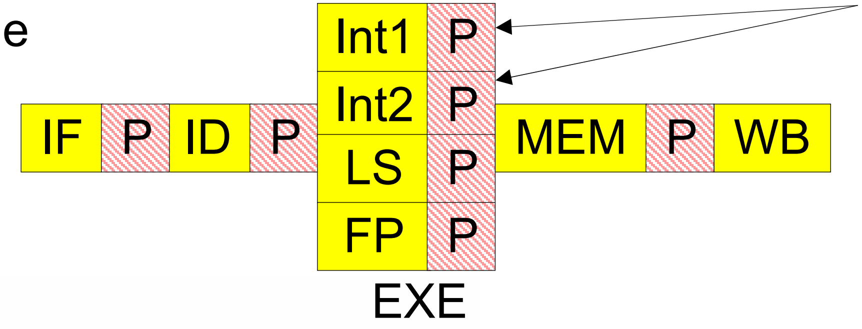
- Fault tree generation
- Fault tree analysis

- Comparison to system at 60°C eliminates constants
- Example with Viterbi decoder showed: Power Management...
  - decreased aging acceleration factor by 2%, but
  - increased aging due to temperature cycling by 40%

# Run time reliability estimation [OC06]

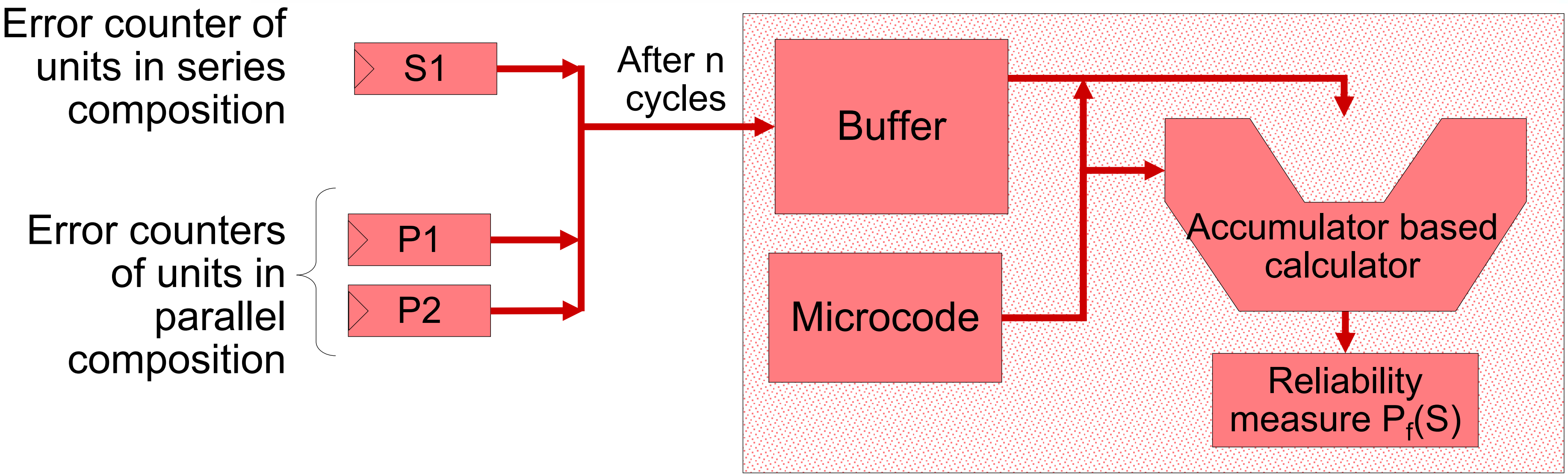
- Design time reliability estimation considers average case:
  - Allows to choose the right architecture
- Run time reliability estimation considers particular case:
  - Allows to evaluate the current chip condition
  - Allows to evaluate the effectiveness of taken actions
- Reliability calculation:
  - Count errors during fixed time interval.
  - Estimate failure probability for next time interval.
  - Account for redundancies.

E.g.: CPU pipeline



Can tolerate permanent failure of either integer unit

# Run time reliability estimation

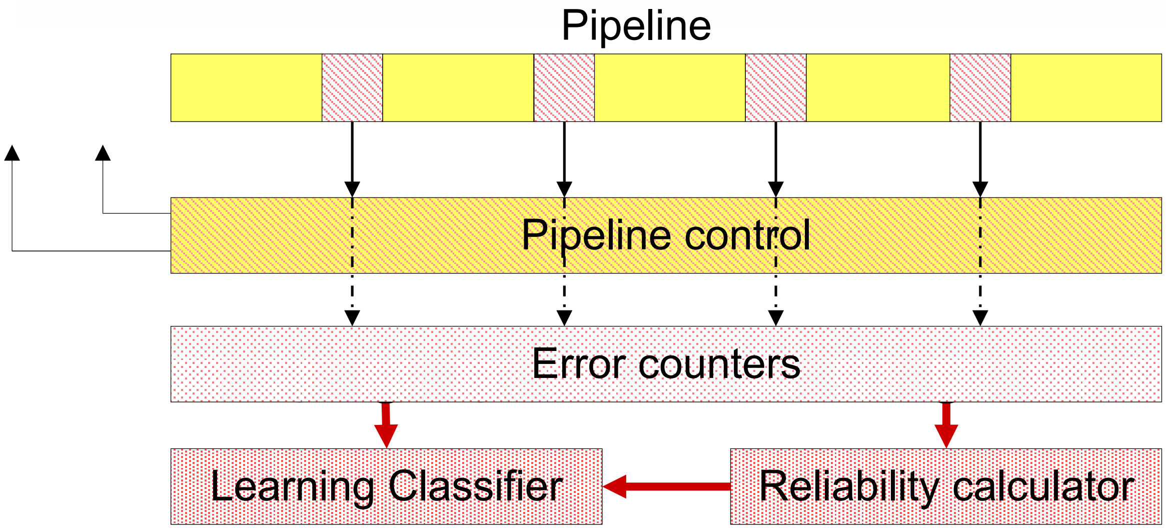

Error counter of units in series composition
S1
Error counters of units in parallel composition
P1
P2
After n cycles
Buffer
Microcode
Accumulator based calculator
Reliability measure $P_f(S)$

- Approximation of failure probability $P_f$ of units in series composition
  - OK if ignoring common mode failures
  - Allows single error counter for all units in series composition
- Reliability of system S until time $T = n\Delta t$:
  - $R_S(T) = (1 – P_f(S))^n$
  - → $P_f(S)$ can serve as a reliability measure
- Calculator needs only addition, shifting & multiplication.

# Summary

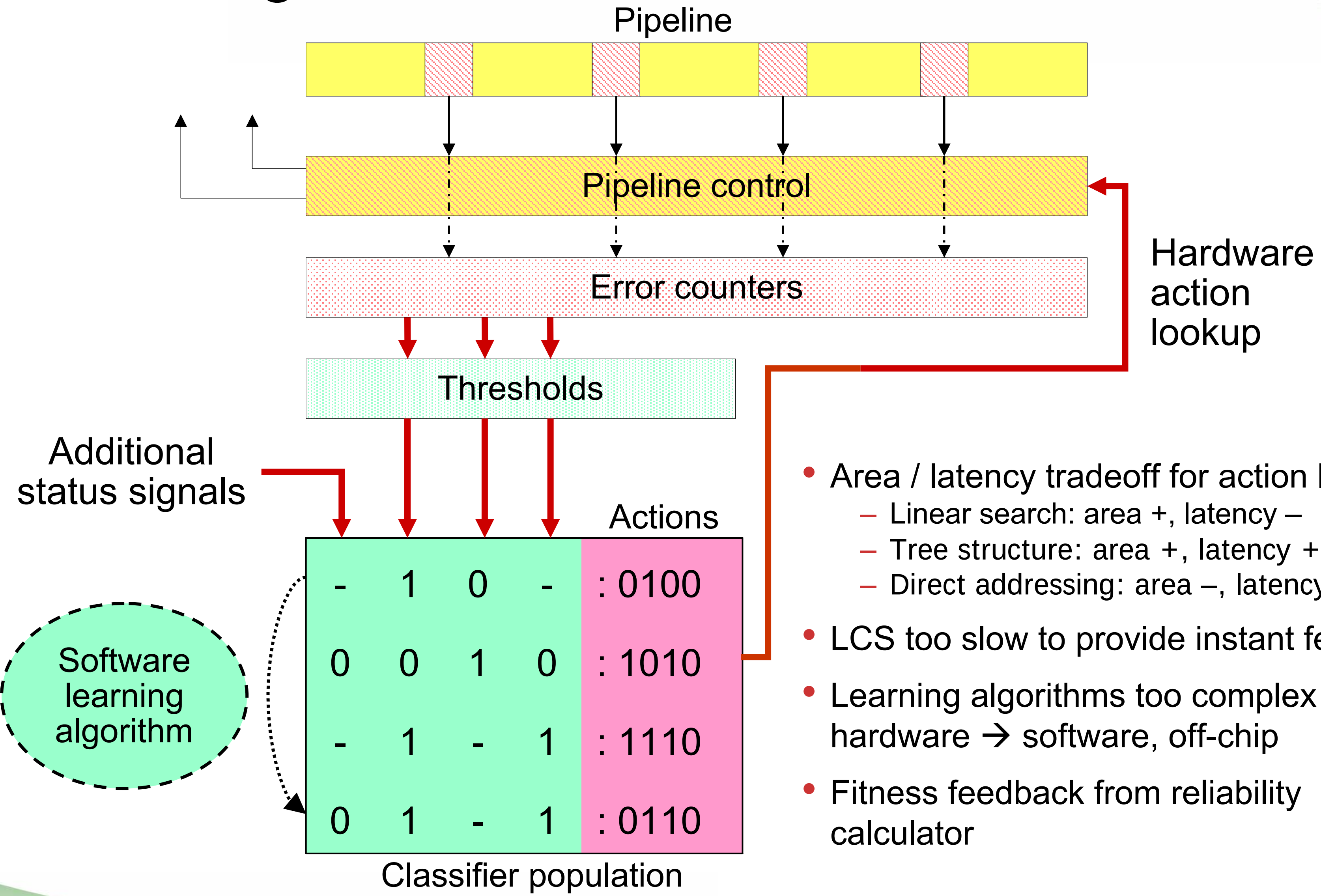
- Investigated existing error detection techniques [BICC06]
- Built a self-healing Leon2 CPU pipeline [VLSI06]
  - Incorporated monitors
  - Pipeline is resistant against timing and transient errors
- Created tool for design time reliability estimation
- Runtime reliability estimator [OC06]

# Future: More on evaluating errors



- Self-healing CPU pipeline: fast, short term response.
- Learning classifier makes long term response.
- Reliability calculator as input and/or feedback to learning classifier.

# Learning classifier



| | | | | Actions |
|---|---|---|---|---|
| - | 1 | 0 | - | : 0100 |
| 0 | 0 | 1 | 0 | : 1010 |
| - | 1 | - | 1 | : 1110 |
| 0 | 1 | - | 1 | : 0110 |

- Area / latency tradeoff for action lookup:
  - Linear search: area +, latency –
  - Tree structure: area +, latency +, complex
  - Direct addressing: area –, latency +
- LCS too slow to provide instant feedback
- Learning algorithms too complex for hardware → software, off-chip
- Fitness feedback from reliability calculator

# Future work

- Based on work with Leon2:
  Automate insertion of protection blocks in CPU pipeline
- Measure accuracy and area of runtime reliability estimator
- Combine design and run time reliability estimation:
  Identify spots that need thorough monitoring
- Identify further actuators

# Cooperations

- team of Prof. Reif, Augsburg (has been initiated)
  - Verification of self-x properties based on logic model
  - Tools for reliability estimations
- team of Prof. Brinkschulte, Karlsruhe
  - close link to organic middleware
- team of Prof. Karl, Karlsruhe
  - exchange experiences in monitoring HW
- team of Prof. Fey, Jena
  - marching pixels as an application to reliable design

# Publications and References

- [VLSI06] A. Bouajila, J. Zeppenfeld, W. Stechele, A. Herkersdorf, A. Bernauer,O. Bringmann, and W. Rosenstiel. *Organic Computing at the System-on-Chip Level*. In VLSI-SoC, October 2006. Invited paper, to be published.
- [OC06] A. Bernauer, O. Bringmann, W. Rosenstiel, A. Bouajila, W. Stechele, andA. Herkersdorf. *An Architecture for Runtime Evaluation of SoC Reliability*. In Organic Computing Workshop, October 2006, to be published.
- [BICC06] A. Bouajila, A. Bernauer, A. Herkersdorf, W. Rosenstiel, O. Bringmann,and W. Stechele. *Error Detection Techniques Applicable in an Architecture Framework and Design Methodology for Autonomic SoC*. In IFIP International Federation for Information Processing, Biologically Inspired Cooperative Computing. August 2006.
- [Lipsa05a] G. Lipsa, A. Herkersdorf, W. Rosenstiel, O. Bringmann, W. Stechele. *Towards a Framework and a Design Methodology for Autonomic SoC*, 2nd IEEE International Conference on Autonomic Computing (ICAC-05), 13-16 June, Seattle, USA
- [Lipsa05b] G. Lipsa, A. Herkersdorf, W. Rosenstiel, O. Bringmann, Walter Stechele. *Towards a Framework and a Design Methodology for Autonomic SoC* , Proceedings Dynamically Reconfigurable Systems, Self-Organization and Emergence, Architecture of Computing Systems (ARCS) 2005, pages 101-108.

- Credits go to Björn Sander and Johannes Zeppenfeld for their contribution to this project.

- [SSS04] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, D. Tarjan. *Temperature Aware Microarchitecture: modeling and implementation*. TACO 1(1):94-125, 2004.
- [Austin03] D. Ernst, N. S. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, T. Mudge. *Razor: A Low-Power Pipeline Based on Circuit-Level Timing Speculation*, 36th Annual International Symposium on Microarchitecture (MICRO-36), December 2003.
- [Chardonnerau02] D. Chardonnereau et al., *"32-Bit RISC Processor Implementing Transient Fault-Tolerant Mechanisms and its Radiation Test Campaign Results"*, Single-Event Effects Symposium, NASA, Apr. 2002.
- [Austin01] C. Weaver and T. Austin, *A Fault Tolerant Approach to Microprocessor Design. I*in: Proc. Intl. Conf. Dependable Systems and Networks, pp. 411–420 (2001).
- [Tuba97] N. Touba and E. McCluskey, *Logic Synthesis of Multilevel Circuits with Concurrent Error Detection*, IEEE Trans. CAD 16(7), 783–789 (1997).
- [Nic99] M. Nicolaidis, *Time Redundancy Based Soft-Error Tolerance to Rescue Nanometer Technologies.* In: Proc. 17th IEEE VLSI Test Symposium, pp. 86–94 (1999).
- [Nic93] M. Nicolaidis, *Efficient Implementations of Self-Checking Adders and ALUs.* In: Proc.23rd Intl. Symp. Fault-Tolerant Computing, pp. 586–595 (1993).