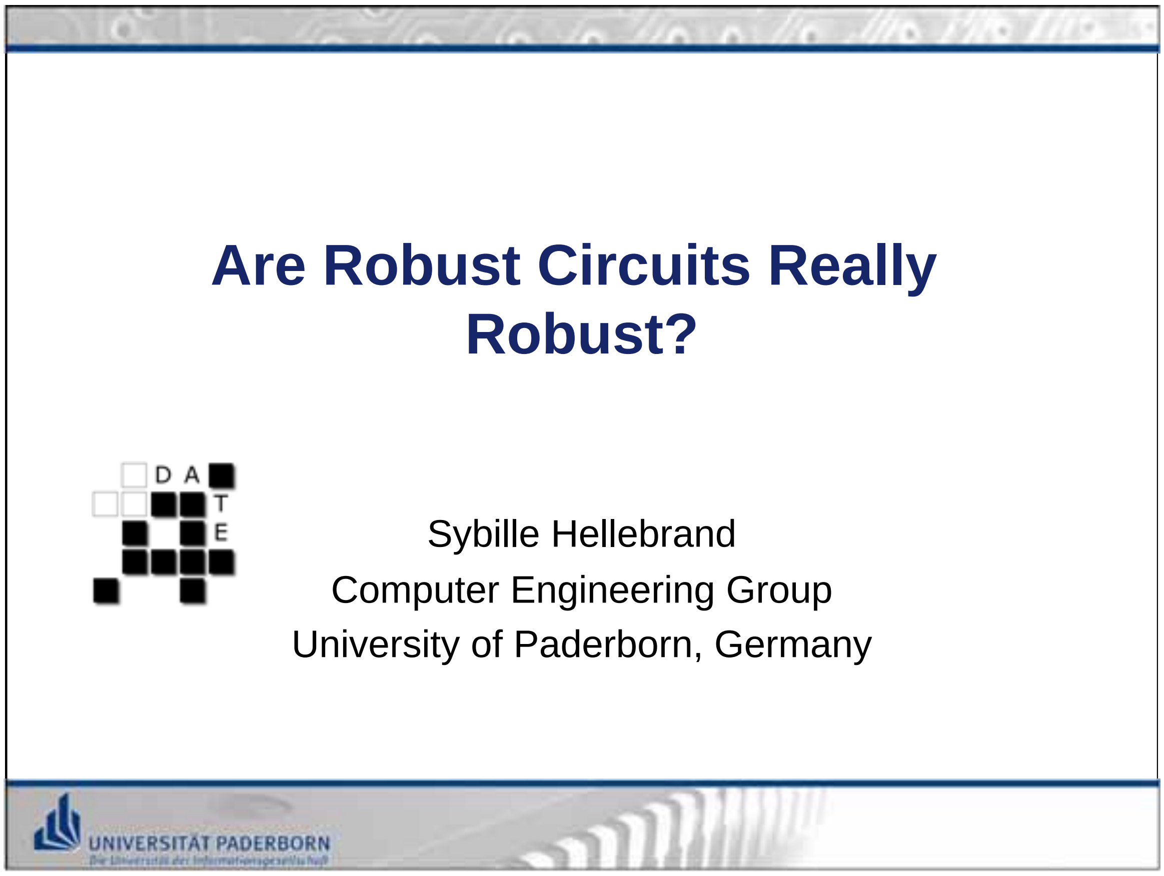
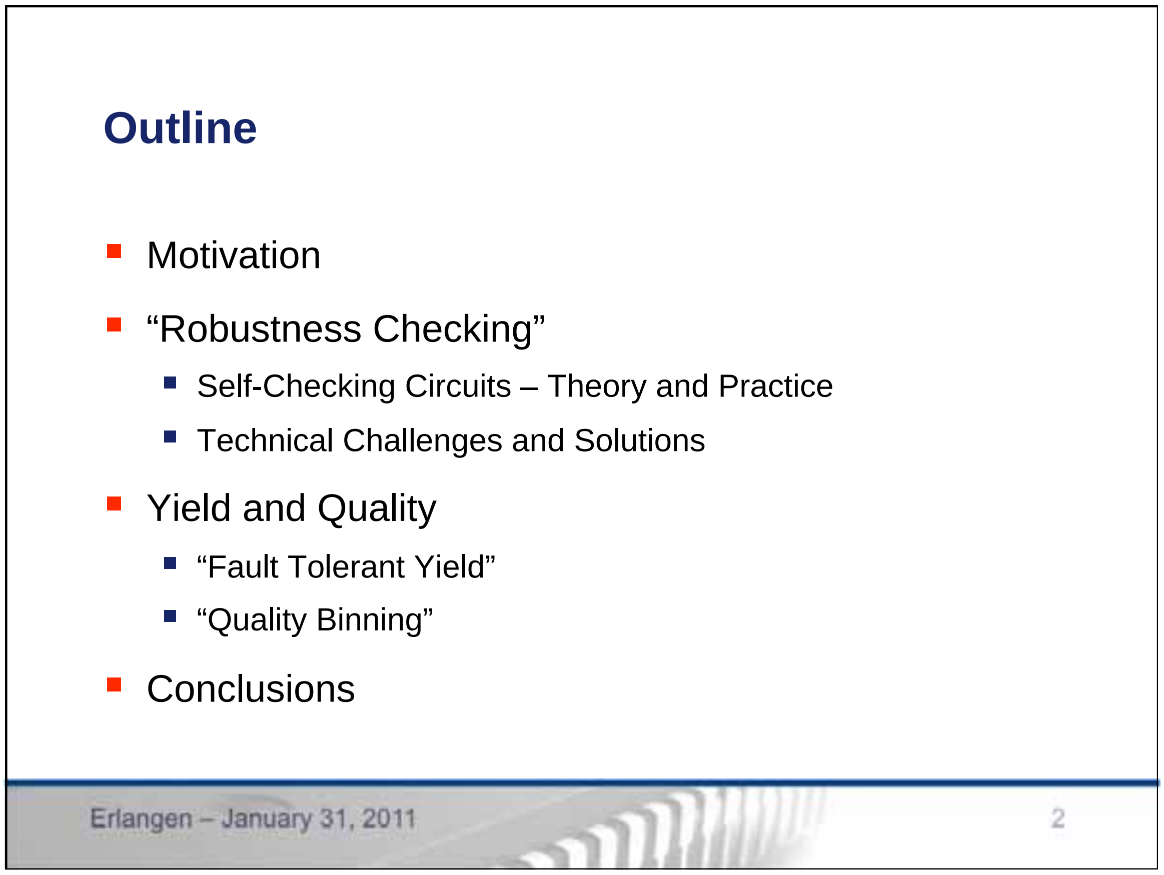


Are Robust Circuits Really Robust?

Sybille Hellebrand
Computer Engineering Group
University of Paderborn, Germany

UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft



Outline

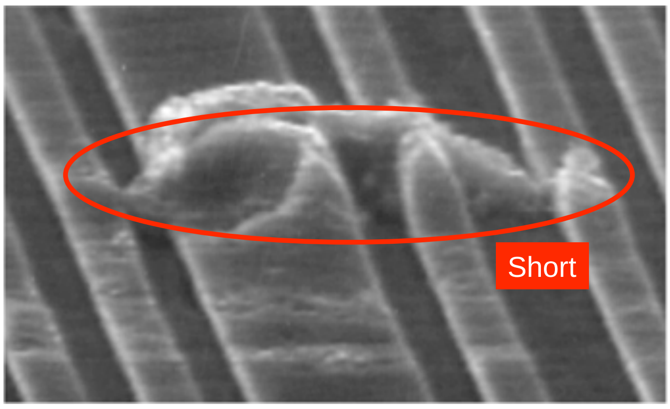
- Motivation
- “Robustness Checking”
  - Self-Checking Circuits – Theory and Practice
  - Technical Challenges and Solutions
- Yield and Quality
  - “Fault Tolerant Yield”
  - “Quality Binning”
- Conclusions

µElectronic is everywhere

more than 80 µProcessors to control various functions (ABS, ..., Infotainment, ...)

Major Problem so far

- „Spot defects“, „random defects“ during manufacturing



[http://www.icyield.com]

Nanoscale Integration

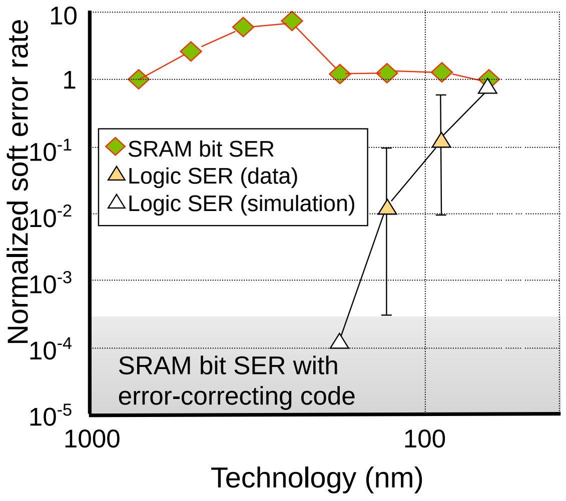
- Potential for integrating highly complex innovative products into single chip (SoC) or package (SiP)
- Problems
  - Soft errors
  - Parameter variations
    cf. Borkar, IEEE Micro 2005

DESIGNING RELIABLE SYSTEMS FROM UNRELIABLE COMPONENTS: THE CHALLENGES OF TRANSISTOR VARIABILITY AND DEGRADATION

Soft Errors

- Caused by
  - Alpha particles, cosmic radiation
- Measures
  - SER (Soft Error Rate) given in
  - FIT (Failure in Time)
    1 FIT = 1 failure in 10^9 hours (≈ 114,155 years)
- Example
  - SER for μProcessor with embedded SRAM is 50,000 FIT
    (1 soft error every 2 years)
  - But: Multiprocessor system with 100 chips has 1 failure per week

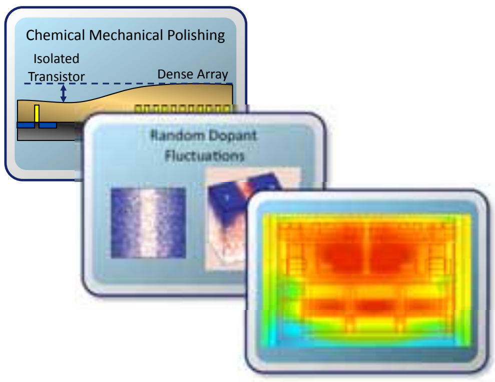
SER for Latches/Flipflops in Random Logic



[Baumann, IEEE Design&Test 2005]

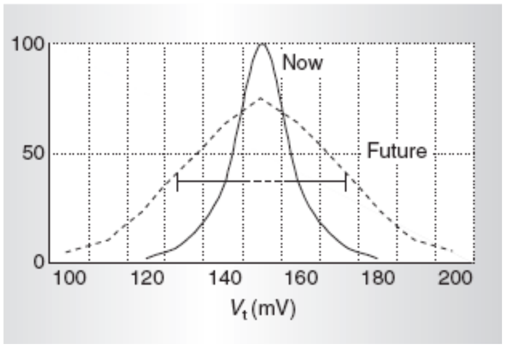
Parameter Variations

- Static variations
  - Systematic
  - Random
- Dynamic variations
- Variations over time (aging)



Example: Random Dopant Fluctuations

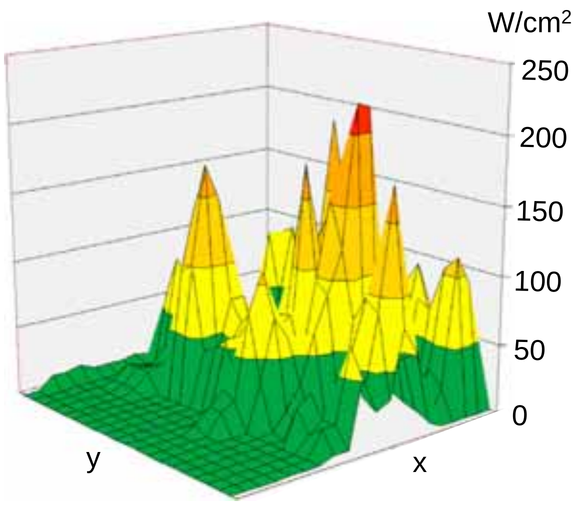
- Threshold voltage V_{th}
  - Determined by the concentration of dopant atoms in the channel
  - Only a few dopant atoms in nano scale transitors
  - Law of large numbers is no longer valid, quantum effects must be considered



[Borkar, IEEE Micro 2005]

Dynamic Parameter Variations

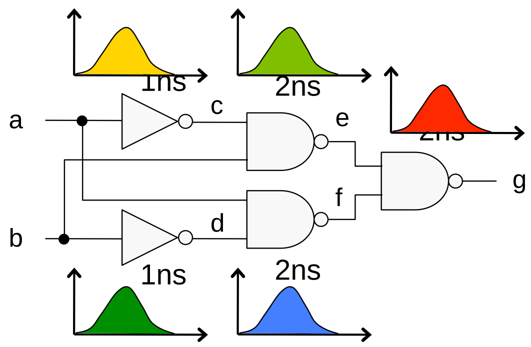
- „Power density“ in a µProcessor chip
- Problems
  - Hot spots
  - Varying supply voltage
  - ...



[Borkar, IEEE Micro 2005]

Consequences

Most parameter variations result in timing variations

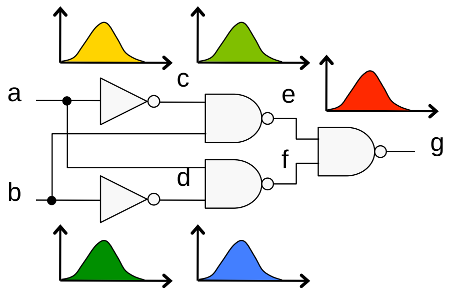


Traditional view: nominal or worst case delay

Now: probability density functions (PDF) for delay

Variation-Aware and Robust Design

- Statistical timing analysis
  - More and more commercial EDA support
- Redundancy
  - Hardware
  - Time
  - Information
  - Algorithmic
- Self-calibrating architectures



Example

Razor

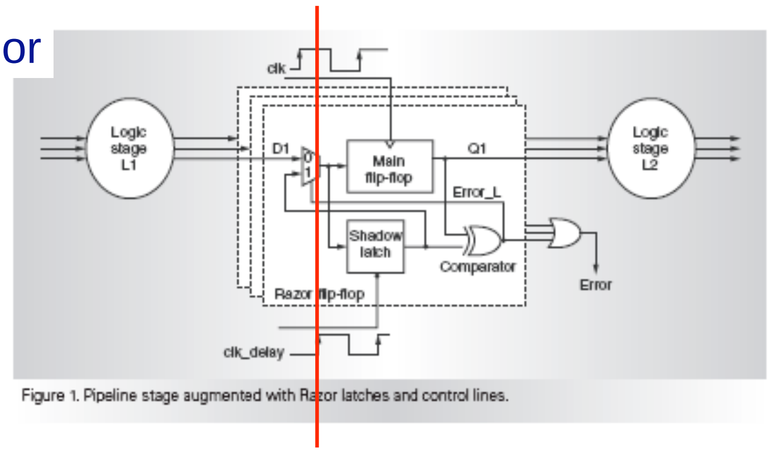


Figure 1. Pipeline stage augmented with Razor latches and control lines.

[D. Ernst et al., IEEE Micro, 2004]

Razor – Error Rates

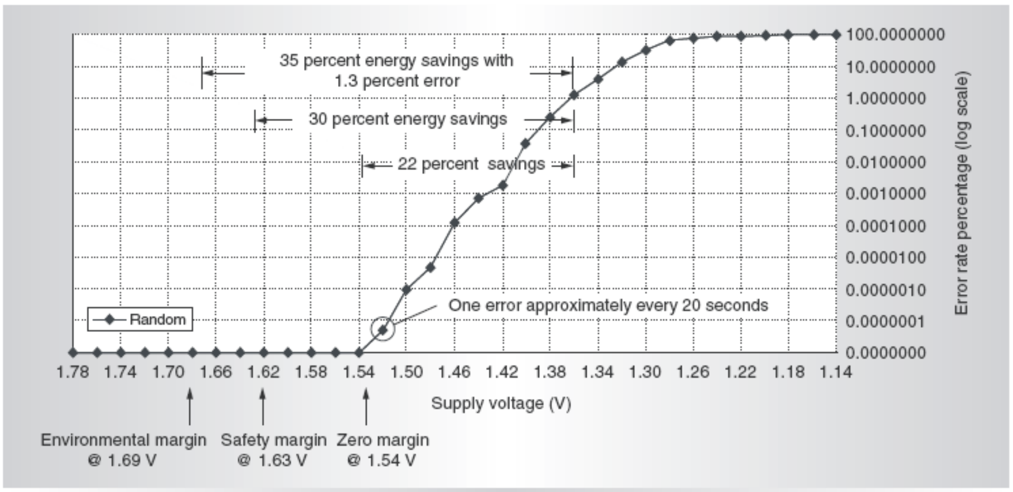
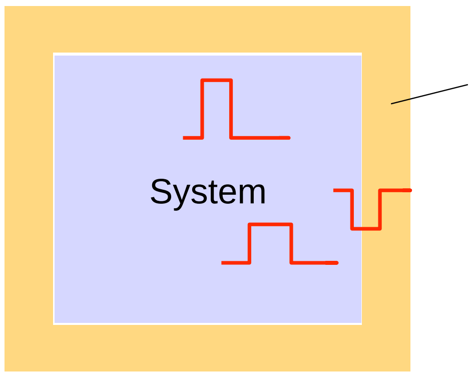


Figure 3. Measured error rates for an 18 × 18-bit field-programmable gate array multiplier block at 90 MHz and 27° C.

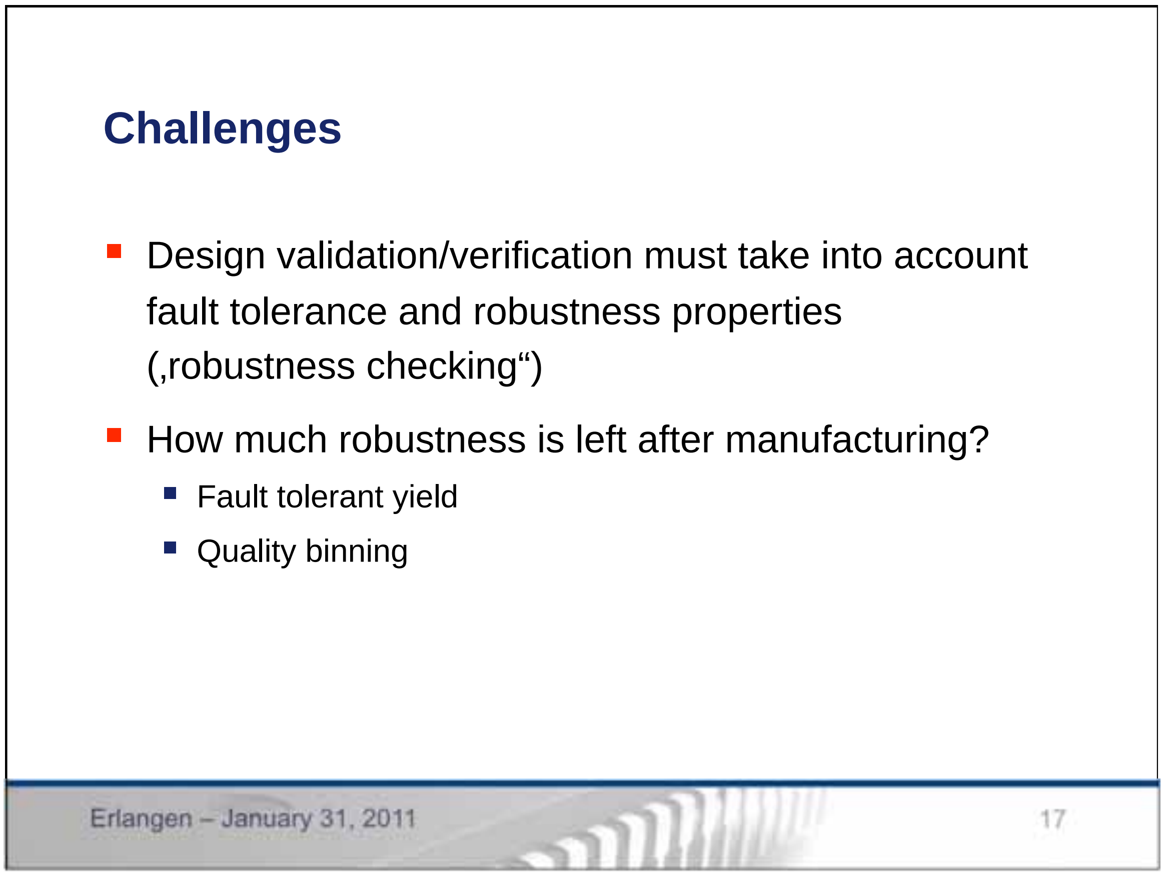
[D. Ernst et al., IEEE Micro, 2004]

Robust Systems



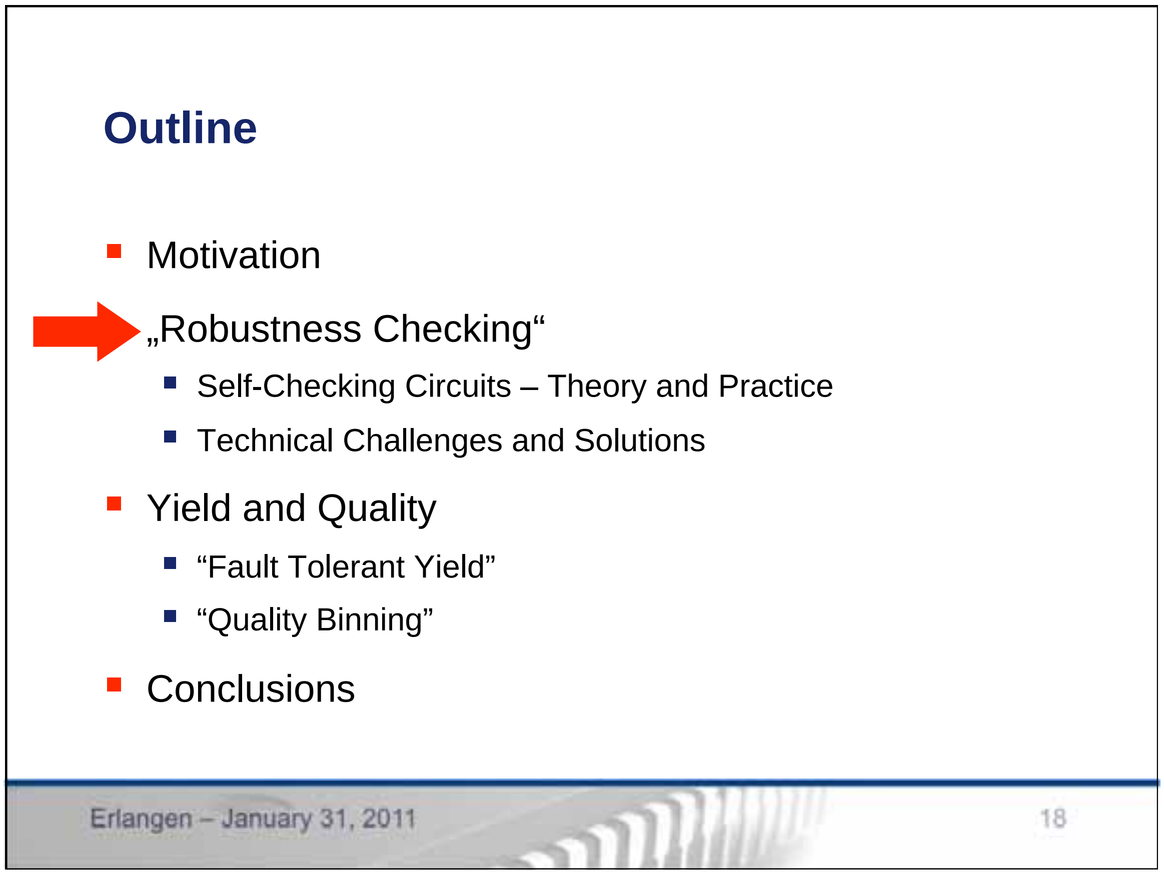
Robust implementation compensates static and/or dynamic parameter variations and/or soft errors

- Classical fault tolerant architectures (Self-checking circuits, TMR, …)
- New self-calibrating, self-adaptive solutions



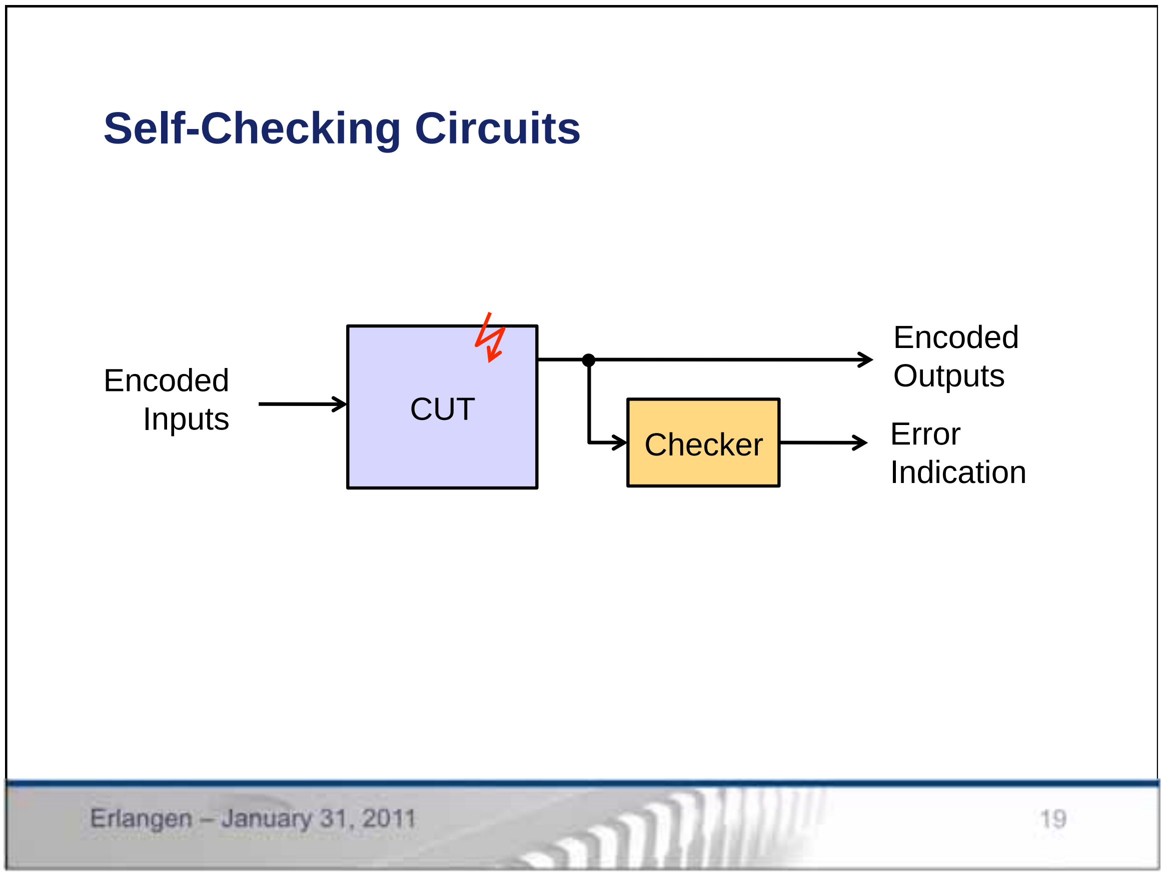
Challenges

- Design validation/verification must take into account fault tolerance and robustness properties („robustness checking“)
- How much robustness is left after manufacturing?
  - Fault tolerant yield
  - Quality binning

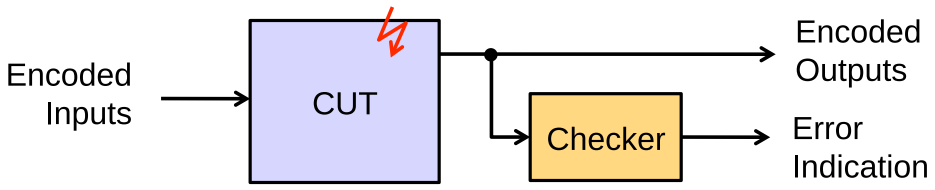


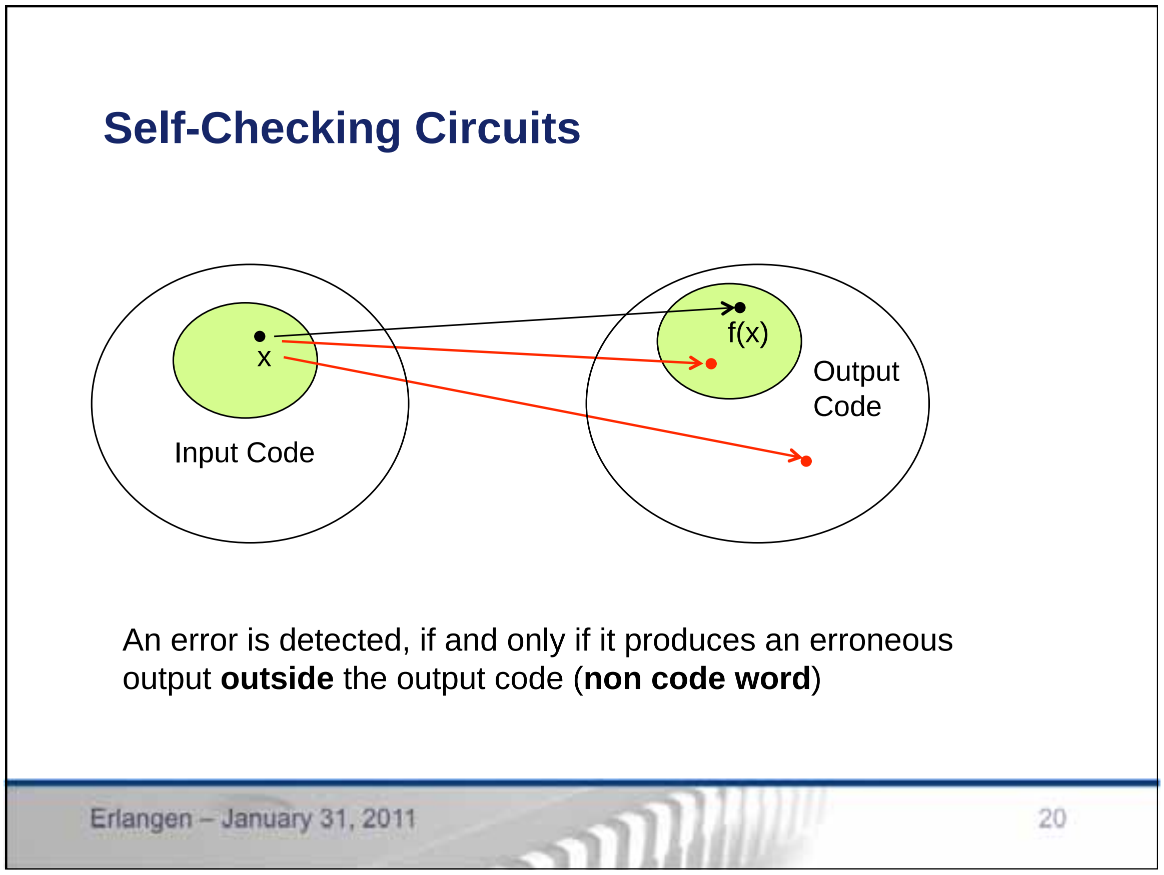
Outline

- Motivation
- „Robustness Checking“
  - Self-Checking Circuits – Theory and Practice
  - Technical Challenges and Solutions
- Yield and Quality
  - “Fault Tolerant Yield”
  - “Quality Binning”
- Conclusions

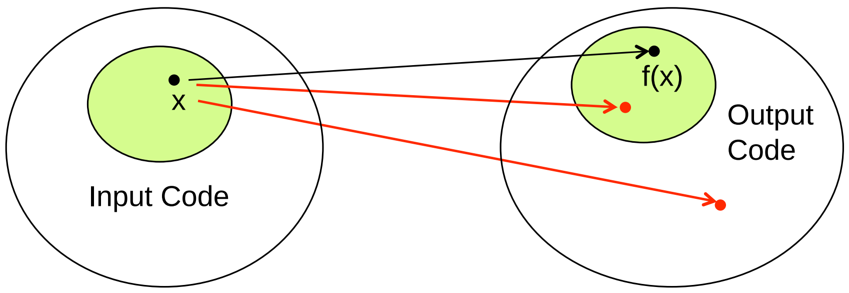


Self-Checking Circuits





Self-Checking Circuits



An error is detected, if and only if it produces an erroneous output **outside** the output code (**non code word**)

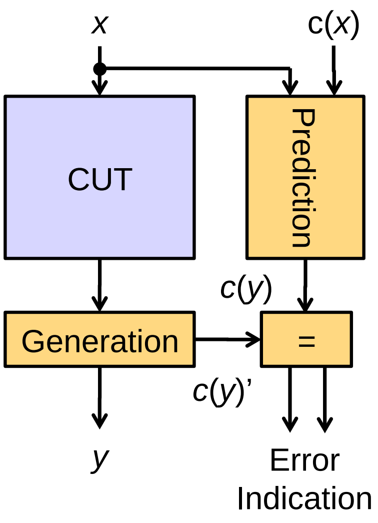
Properties

- Totally self-checking (TSC) goal
  - Faults must be detected when they produce the first erroneous output
- Fault secure (FS)
  - Faults are detected or do not propagate to outputs
- Self-Testing (ST)
  - Every fault can be detected with at least one input

Avoid fault accumulation

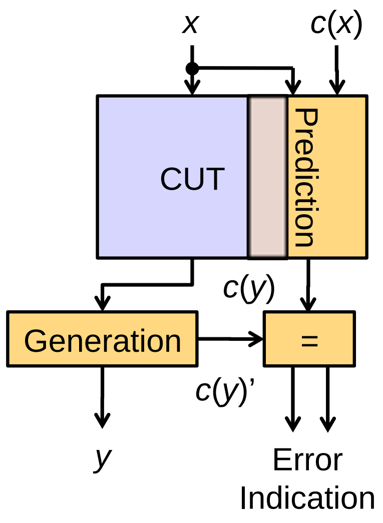
Problem

- Design strategies for self-checking circuits well-known
- But: synthesis may destroy self-checking properties, e.g. by logic sharing



Problem

- Design strategies for self-checking circuits well-known
- But: synthesis may destroy self-checking properties, e.g. by logic sharing

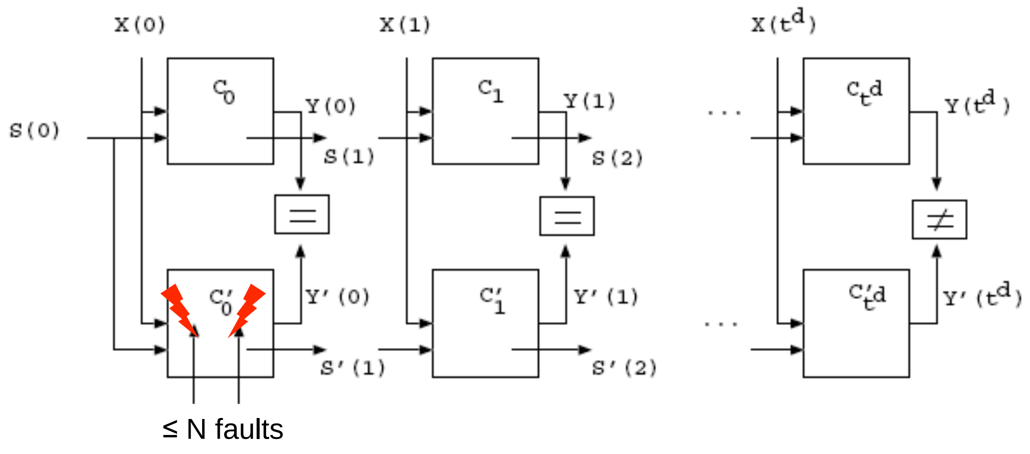


Consequences

- Analysis of circuit robustness is required to
  - check robustness properties after synthesis
  - identify critical nodes / regions
  - compare different circuit implementations

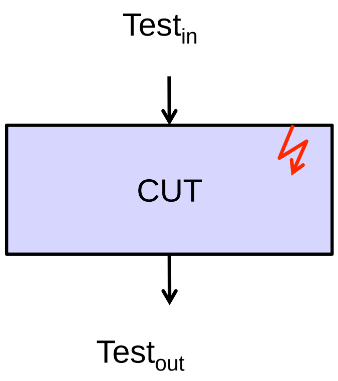
Formal Robustness Checking

[G. Fey et al. 2008, 2009]



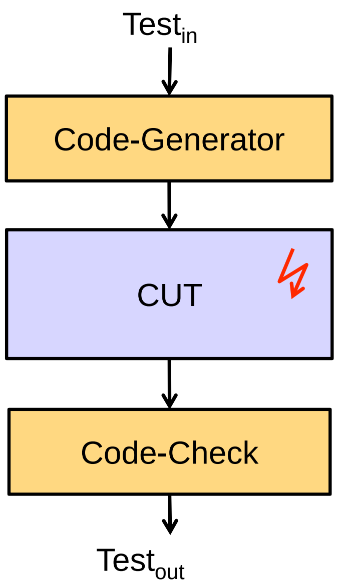
ATPG-Based Analysis

- Reuse efficient tools for manufacturing test
- **A**utomatic **T**est **P**attern **G**eneration (ATPG) can
  - Generate test patterns
  - Identify redundant faults



Example: Self-Testability

- Self-testable = every fault is detectable
- Use test bench to constrain ATPG
  - Only input codes as patterns
  - Detection only for non code outputs



Strongly Fault-Secure Circuits (SFS)

- Discussed so far
  - Fault-secure (FS)
  - Self-testing (ST)

Avoid fault accumulation

- Strongly fault-secure (SFS)
  Circuit is SFS w.r.t. fault set F:
  For all f in F
  - ST and FS w.r.t. {f} or
  - FS for {f} and SFS for all sequences {<f, g>}

Secure fault accumulation

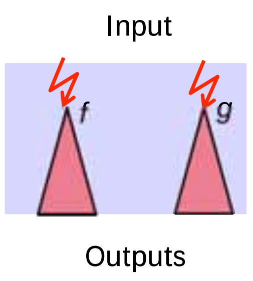
Challenges

- Multiple fault analysis required
- How to compare circuits which are not 100% SFS?

Iterative Robustness Grading

- Classify multiple fault *f* as
  - insecure (!FS)
  - secure (FS & ST) or
  - unknown (else)
- Study $\{f\} \times F$, if fault is unknown
- At each iteration compute upper and lower bounds

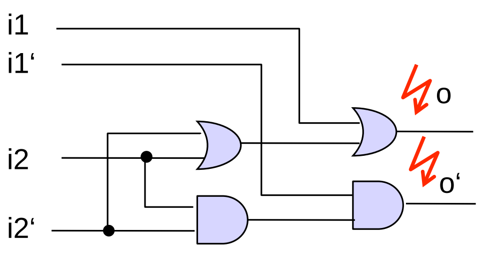
Multiple Fault Analysis



- Unconstrained multiple fault analysis:
  - Rules to determine detectability of multiple faults from properties of single faults
  - E.g. Faults *f* and *g* with disjoint output cones:
    DT(<*f*, *g*>) = DT(*f*) or DT(*g*)

Problem

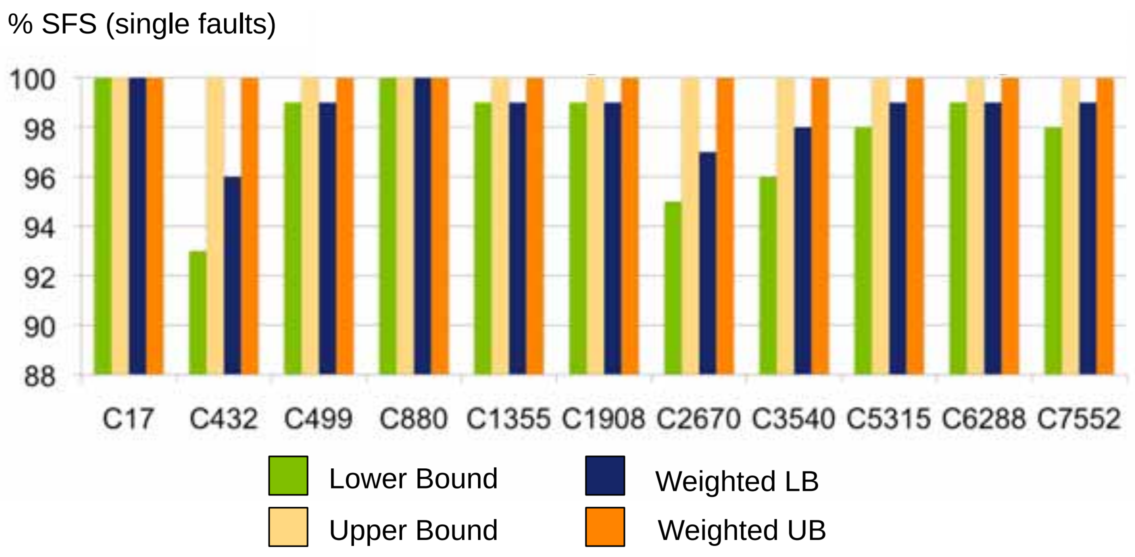
- Rules cannot be directly applied, new code specific rules are applied
- Example: dual-rail circuit



Experimental Results

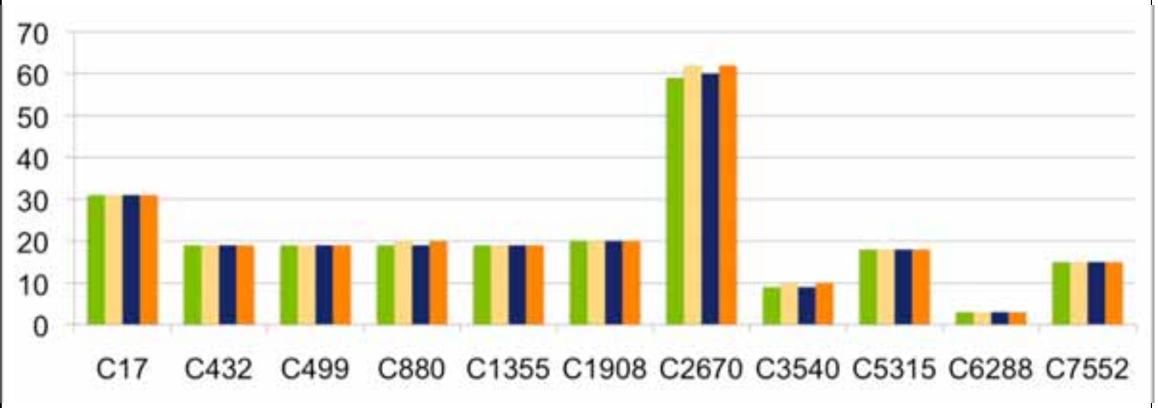
- Unordered input and output encoding & inverter-free implementation
- Parity output encoding
- Thread-parallel SAT-based ATPG tool TIGUAN (Freiburg)

Unordered Input and Output Coding



Parity Output Coding

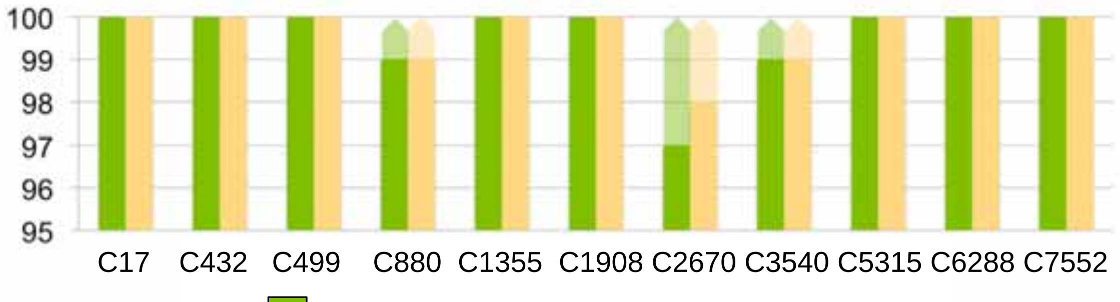
% SFS (single faults)



🟩 Lower Bound 🟦 Weighted LB

🟨 Upper Bound 🟧 Weighted UB

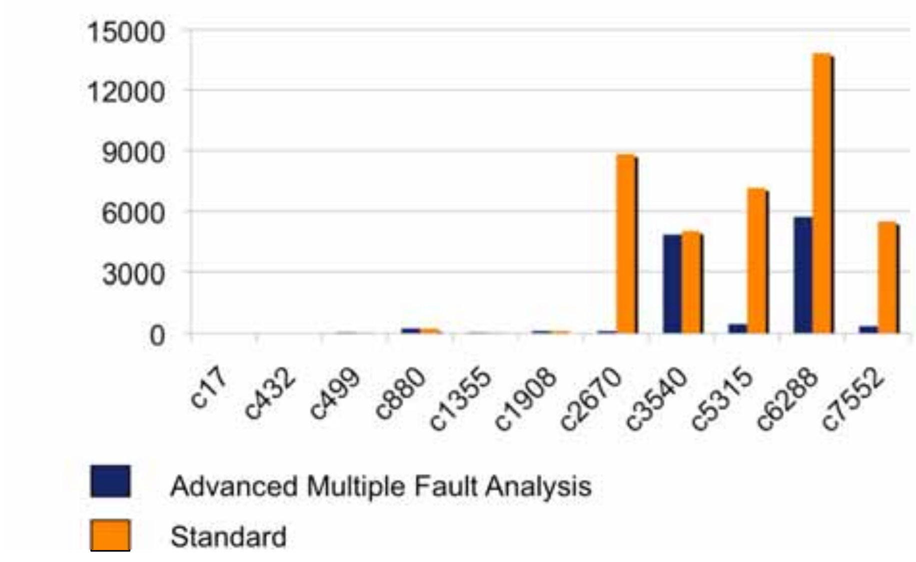
Precision for Single and Double Faults



🟩 Precision

🟨 Precision (weighted bounds)

Run Time for Double Faults (Seconds)

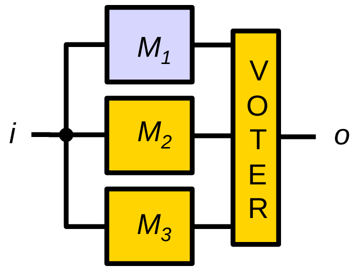


Outline

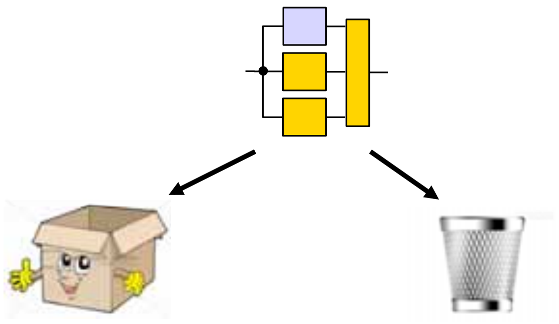
- Motivation
- „Robustness Checking“
  - Self-Checking Circuits – Theory and Practice
  - Technical Challenges and Solutions
- Yield and Quality
  - “Fault Tolerant Yield”
  - “Quality Binning”
- Conclusions

Example: Triple Modular Redundancy

- Can compensate both permanent and transient faults
- Used both for yield and reliability improvement

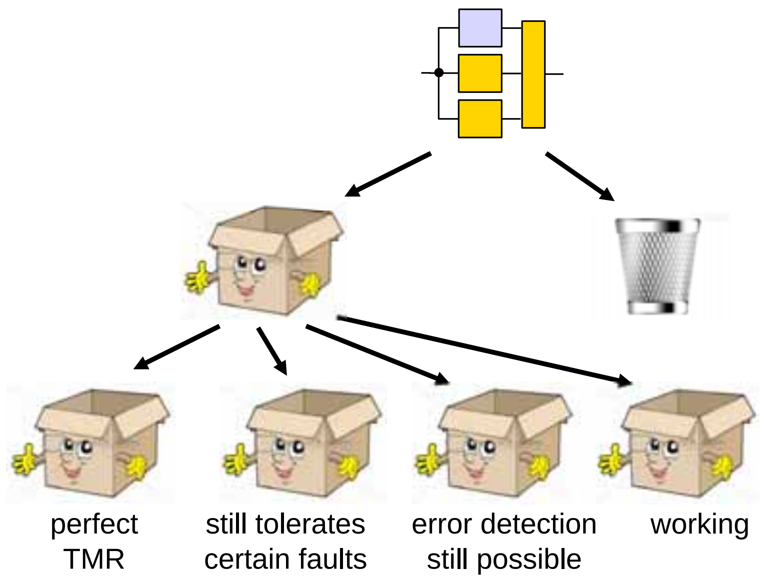


Yield of a TMR System



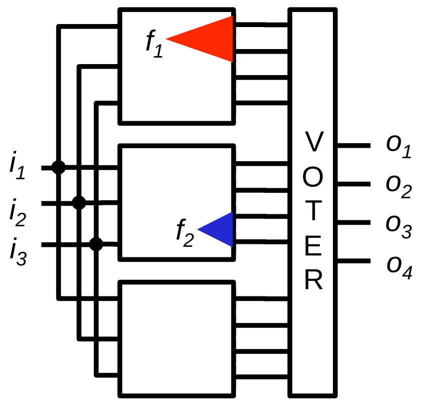
$$Yield = \sum_{i=0}^{\infty} \underbrace{r(i)}_{i \text{ faults tolerated}} \underbrace{p(i)}_{i \text{ faults occur}}$$

Fault Tolerance?


perfect TMR
still tolerates certain faults
error detection still possible
working

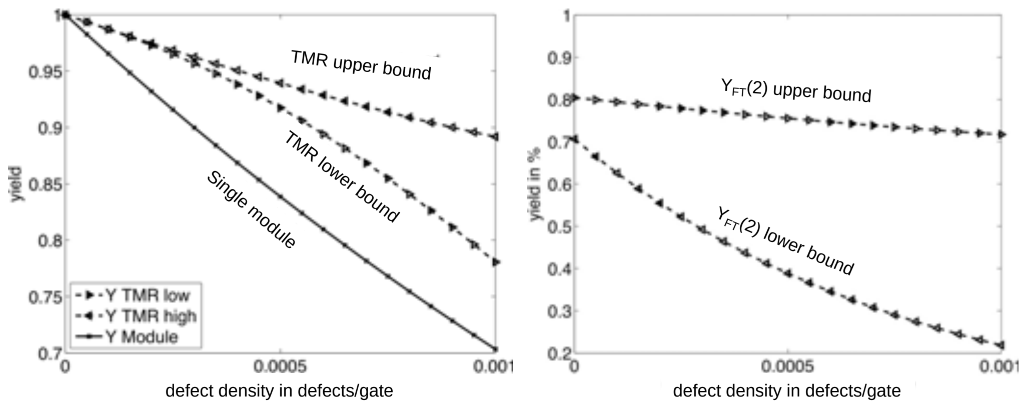
“Fault Tolerant” Yield

- Necessary:
  refined yield estimation



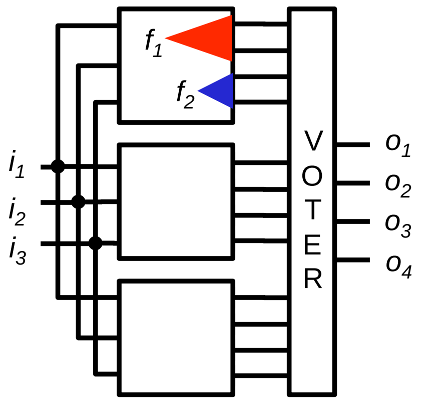
$$Y_{FT}(k) = \sum_{i=0}^{\infty} \underbrace{r(i+k \mid i)}_{k \text{ additional faults tolerated}} r(i)\, p(i)$$

Example b13



Quality Binning

- Enhanced manufacturing test must classify chips according to quality levels
- Two steps
  - "Functional" Test: Go/NoGo
  - Diagnostic Test with DfT
    - Reveals "functionally redundant" faults
    - Critical faults must be distinguished from tolerable faults



DFG-Project RealTest

- Topics
  - Variation-Aware Testing
  - Design and Test of Robust Systems
- Partners
  - IIS-EAS Dresden (Straube, Vermeiren), U. Freiburg (Becker), U. Stuttgart (Wunderlich), U. Paderborn (Hellebrand), U. Passau (Polian)
- Industrial Board
  - Mentor Graphics Hamburg, Infineon München

Conclusions

- Soft errors and parameter variations require a robust system design
- Robust circuit design comes along with new challenges in
  - design validation/verification
  - yield estimation (traditional vs. “fault tolerant yield”)
  - testing (pass/fail vs. “quality binning”)