DESIGNING RELIABLE SYSTEMS with UNRELIABLE COMPONENTS

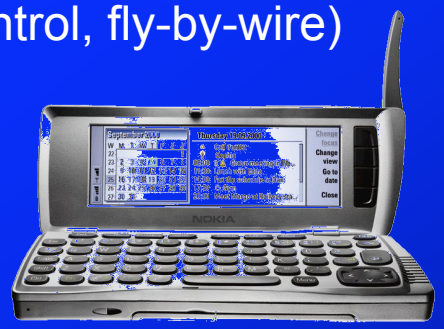
Giovanni De Micheli
Centre Systèmes Intégrés

Outline

- Introduction and motivation
- Variability and robustness
  - Self-calibrating circuits
- Soft failures
  - Error detection and correction
- Hard failures
  - Redundancy
  - System-level management
- Hardware system integration issues
  - On-chip networks
  - Self-healing
- Nanotechnology challenges
- Summary and conclusions

Reliable design:
where do we need it ?

- **Traditional applications**
  - Long-life applications (space missions)
  - Life-critical, short-term applications (aircraft engine control, fly-by-wire)
  - Defense applications (aircraft, guidance & control)
  - Nuclear industry
  - Telecommunications



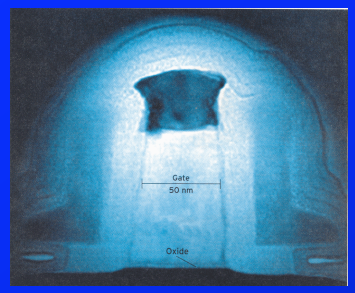
- **New computation-critical applications**
  - Health industry
  - Automotive industry
  - Industrial control systems and production lines
  - Banking, reservations, commerce

The economic perspective

- Availability is a critical business metric for commercial systems and services
  - Nearly 100% availability (“five nines+”) is almost mandatory
- Service outages are frequent
  - 65% website managers report outages over a 6-month period
  - 25% report three or more outages [Internet week 2000]
- High cost of downtime of systems providing vital services
  - Lost opportunities and revenues, non-compliance penalties, potential loss of lives
  - Cost per an hour of downtime varies from $89K for cellular services to $6.5M for stock brokerage [Gartner Group 1998]
- Revenue for *high availability* products in the data/telecom/computer server market is over $100B (≈ $15B for servers alone) [IMEX Research 2003]

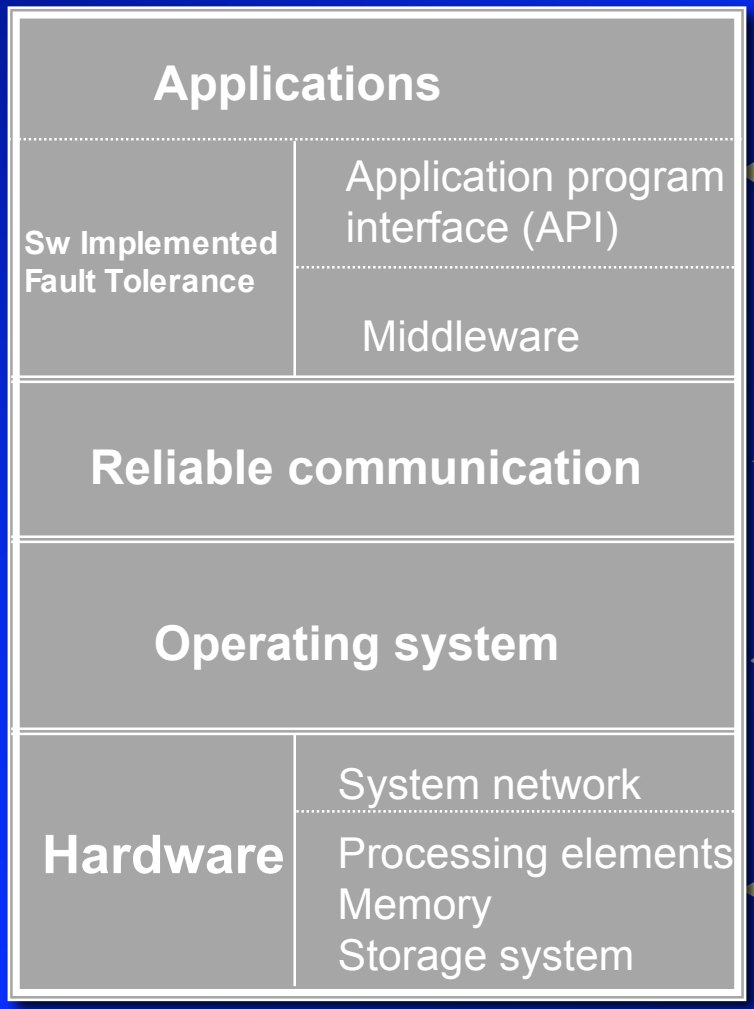
The physical perspective

- Malfunctions are more likely to happen
  - Manufacturing imperfections
    - Design close to scaling limits
    - Smaller devices, larger variances



- Hostile environments
  - Embedded systems exposed to harsh conditions
  - System fragility due to low-voltage operation
- Material aging
  - Oxide breakdown, electromigration failures
  - More likely with scaled-down lithography

Reliability is a system issue



Checkpointing and rollback, application replication, software, voting (fault masking), process pairs, robust data structures, recovery blocks, N-version programming,

CRC on messages , acknowledgment, watchdogs, heartbeats, consistency protocols

Memory management and exception handling, detection of process failures, checkpoint and rollback

Error correcting codes, M-out-of-N and standby redundancy , voting, watchdog timers, reliable storage (RAID, mirrored disks)

[Iyer]

Hardware reliability challenges

- Extremely small device size
  - Coping with deep submicron (DSM) technologies
    - Spreading of parameters
- Extremely large circuit scale
  - System complexity
    - Interaction of computing and storage components
- New computing materials
  - Higher device and defect density
    - How to make the new technologies viable

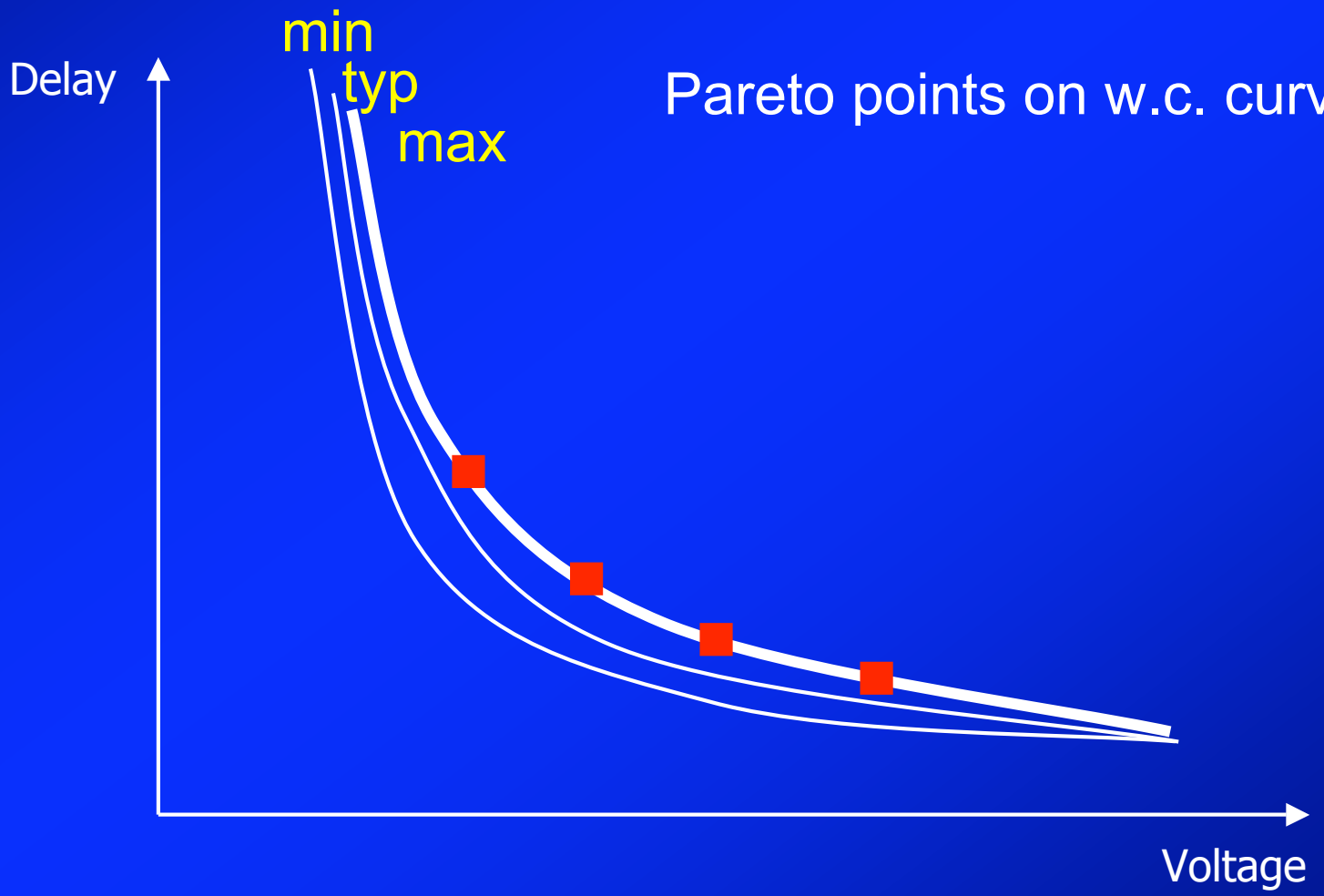
Outline

- Introduction and motivation
- Variability and robustness
  - Self-calibrating circuits
- Soft errors
  - Error detection and correction
- Hard failures
  - Redundancy
  - System-level management
- Hardware system integration issues
  - On-chip networks
  - Self-healing
- Nanotechnology challenges
- Summary and conclusions

Roadmap qualitative trends

- Continued gate downscaling
- Increased transistor density and frequency
  - Power and thermal management
- Lower supply voltage
  - Reduced noise immunity
- Increased spread of physical parameters
  - Inaccurate modeling of physical behavior

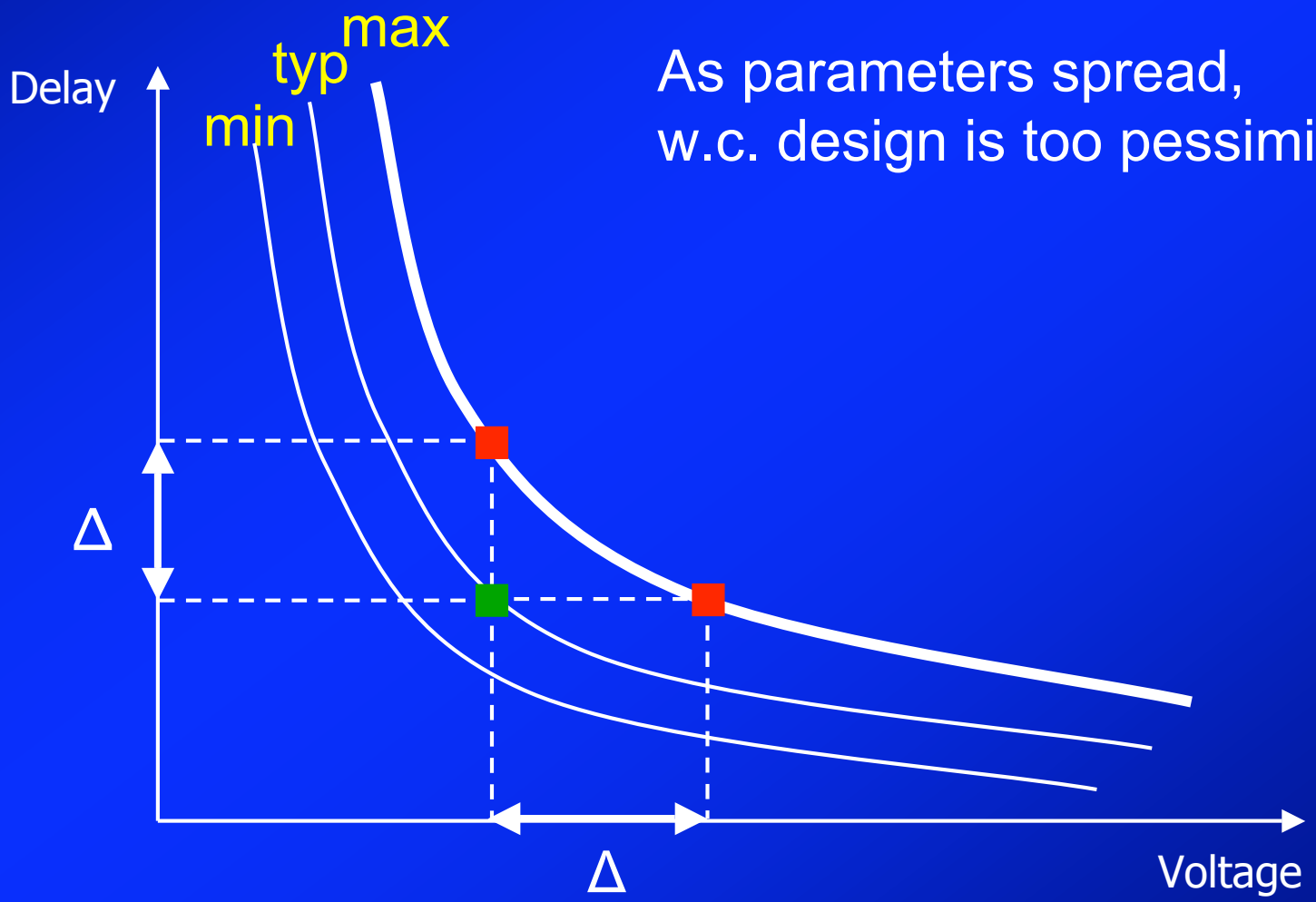
Design space exploration
worst case analysis



Pareto points on w.c. curve

Adaptive design space

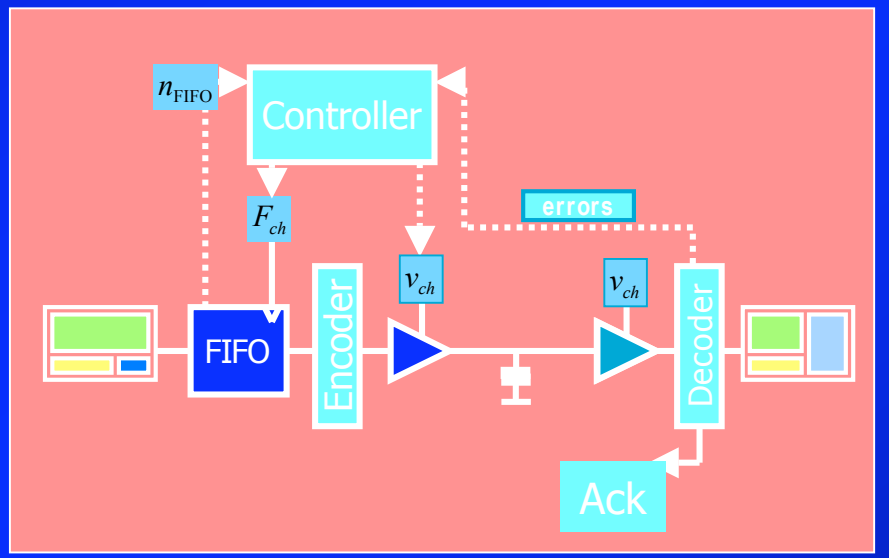
worst case analysis

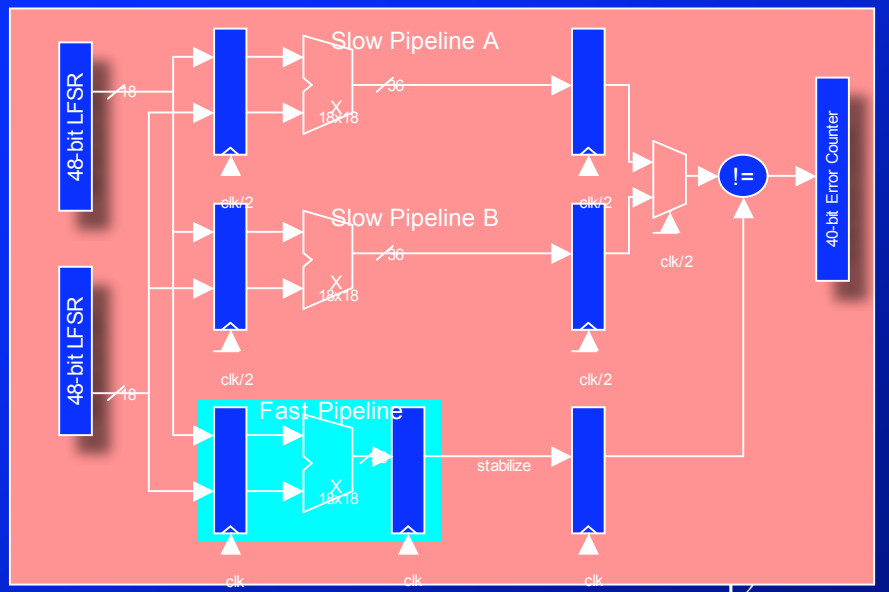


As parameters spread,
w.c. design is too pessimistic

Self-calibrating circuits

- Address variability and robustness
- Design self-calibrating circuits operating at the edge of failure
- Examples:
  - Dynamic voltage scaling of bus swings [Ienne –EPFL]
  - Dynamic voltage scaling in processors
    - Razor [Austin – U Mich]
  - Dynamic latency adjustment
    - Terror [Stanford]





How to calibrate?

- General paradigm
  - A circuit may be in *correct* or *faulty* operational state, depending on a parameter (e.g., voltage)
  - Computed/transmitted data need checks
    - If data is faulty, data is recomputed and/or retransmitted
  - Error rate is monitored on line
  - Feedback loop to control *operational state parameter* based on error rate

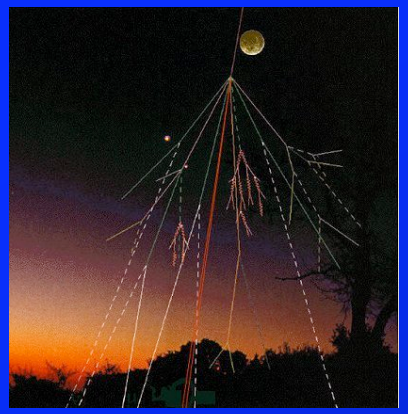
Circuits can generate errors:
- Errors must be detected and corrected
- Correction rate is used for calibration

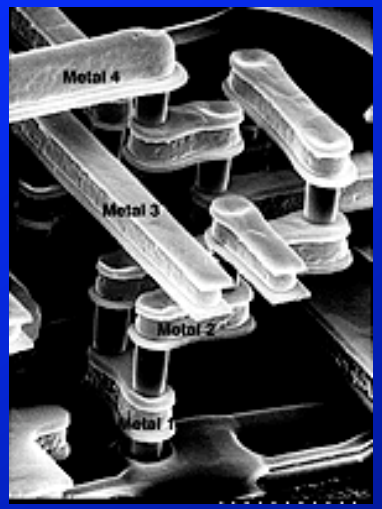
Outline

- Introduction and motivation
- Variability and robustness
  - Self-calibrating circuits
- Soft failures
  - Error detection and correction
- Hard failures
  - Redundancy
  - System-level management
- Hardware system integration issues
  - On-chip networks
  - Self-healing
- Nanotechnology challenges
- Summary and conclusions

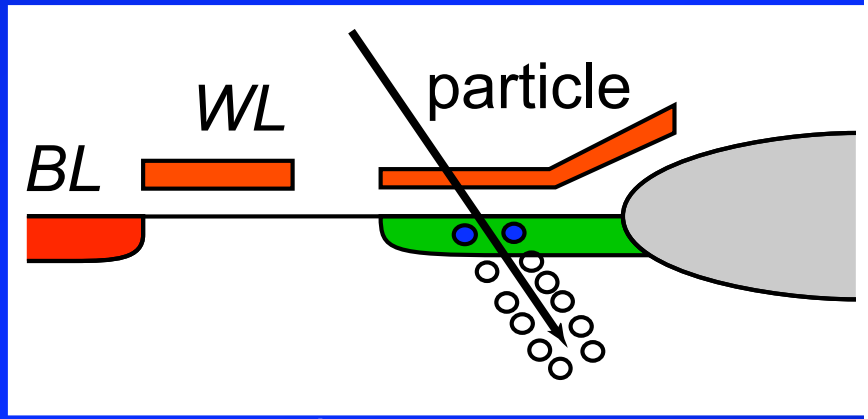
Dealing with transient malfunctions

- Soft errors
  - Data corruption due external radiation exposure
- Crosstalk
  - Data corruption due to internal field exposure
- Both malfunctions manifest themselves as timing errors
  - Error containment



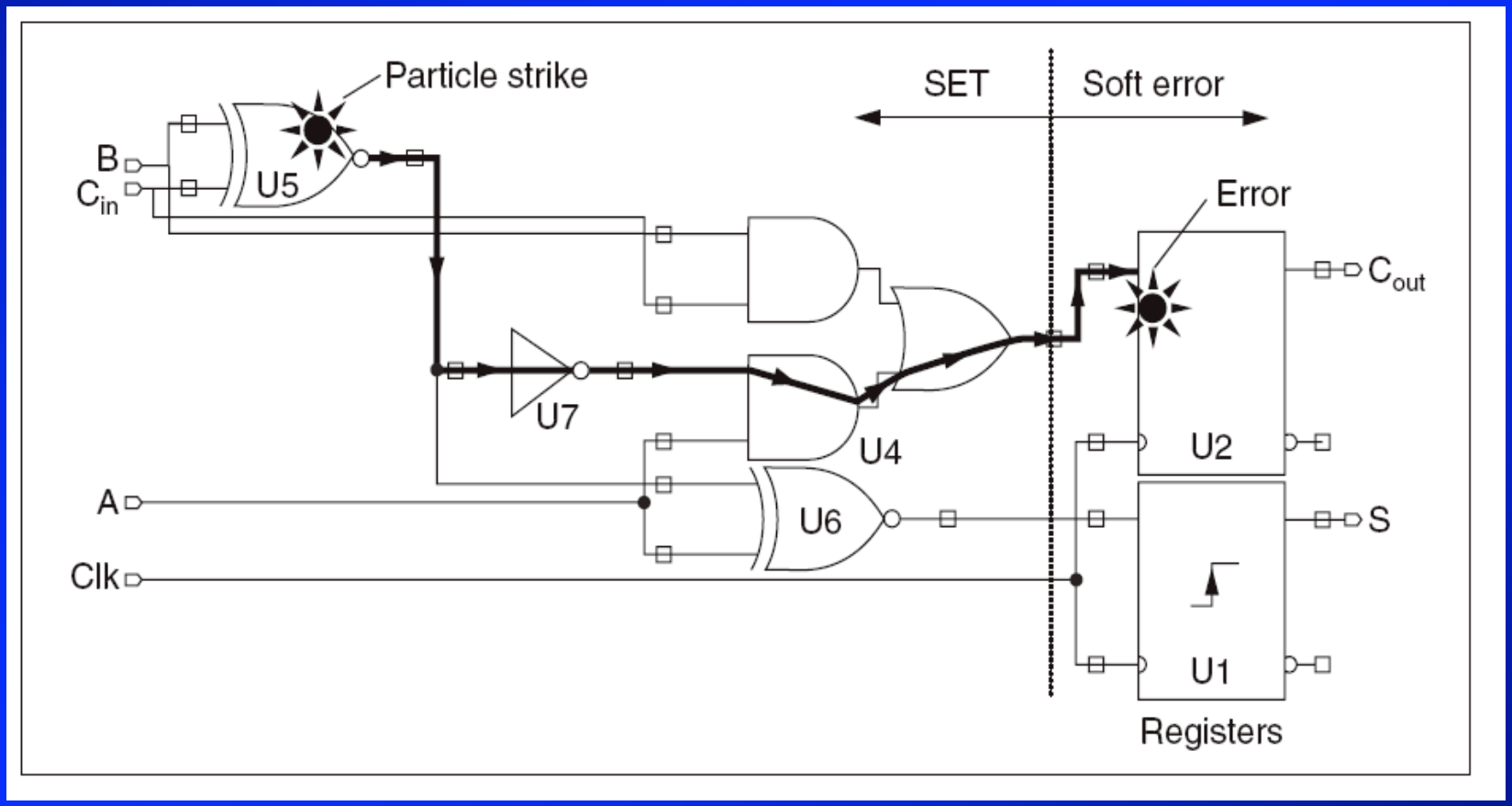


Soft errors



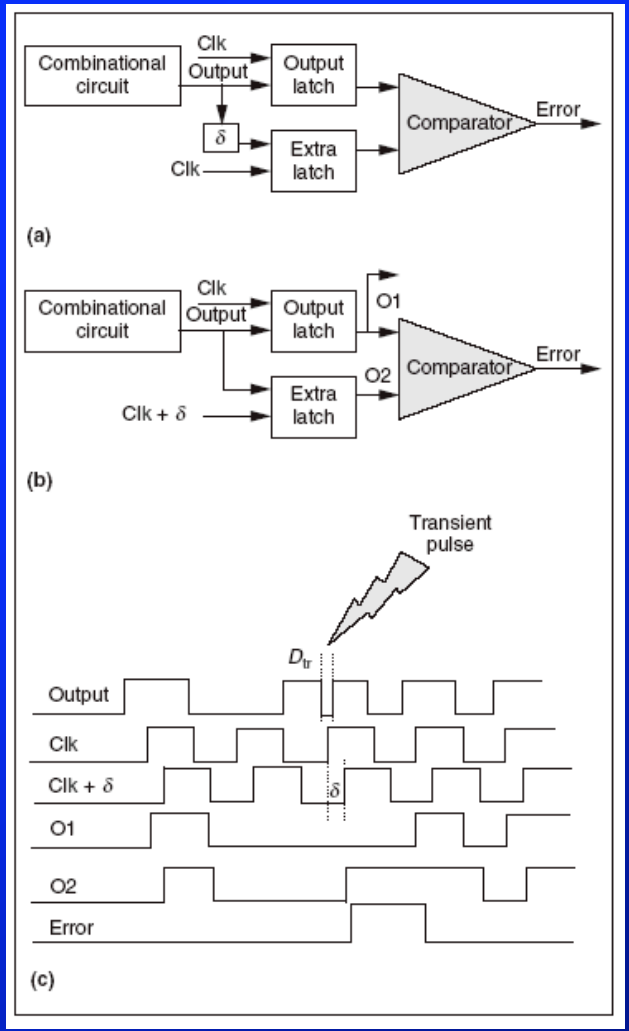
- Due to charge injection:
  - Charged ion:
    - Alpha particle
    - Neutron scattering (from cosmic rays)
- Soft error rate increase with:
  - Environment (e.g., altitude, latitude)
  - Decrease of node critical charge (capacitance*voltage)
- Used to be a problem for DRAMs
  - Now important also for SRAM, registers, etc.

Propagation of soft error



Radiation-hardened registers

- Protection against soft errors
  - Timing errors
- Each latch is duplicated
  - Shadow latch has delayed clock
- Comparison between original and shadow latch detects error
  - Error correction is possible



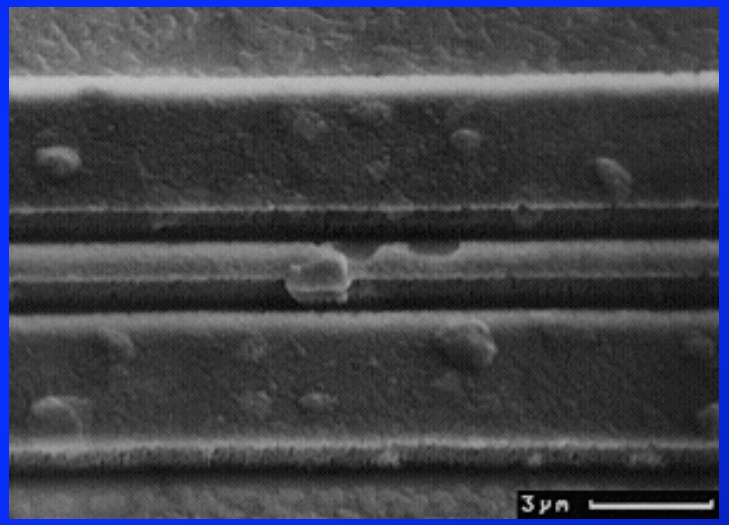
[IROC Technologies]

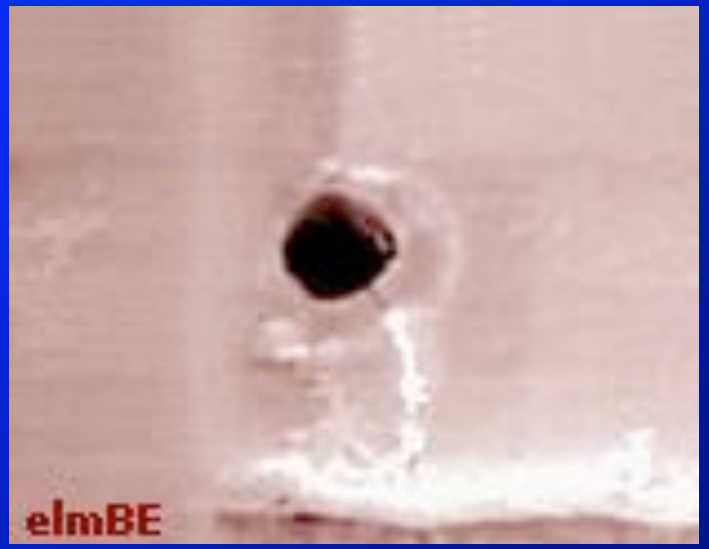
Outline

- Introduction and motivation
- Variability and robustness
  - Self-calibrating circuits
- Soft failures
  - Error detection and correction
- Hard failures
  - Redundancy
  - System-level management
- Hardware system integration issues
  - On-chip networks
  - Self-healing
- Nanotechnology challenges
- Summary and conclusions

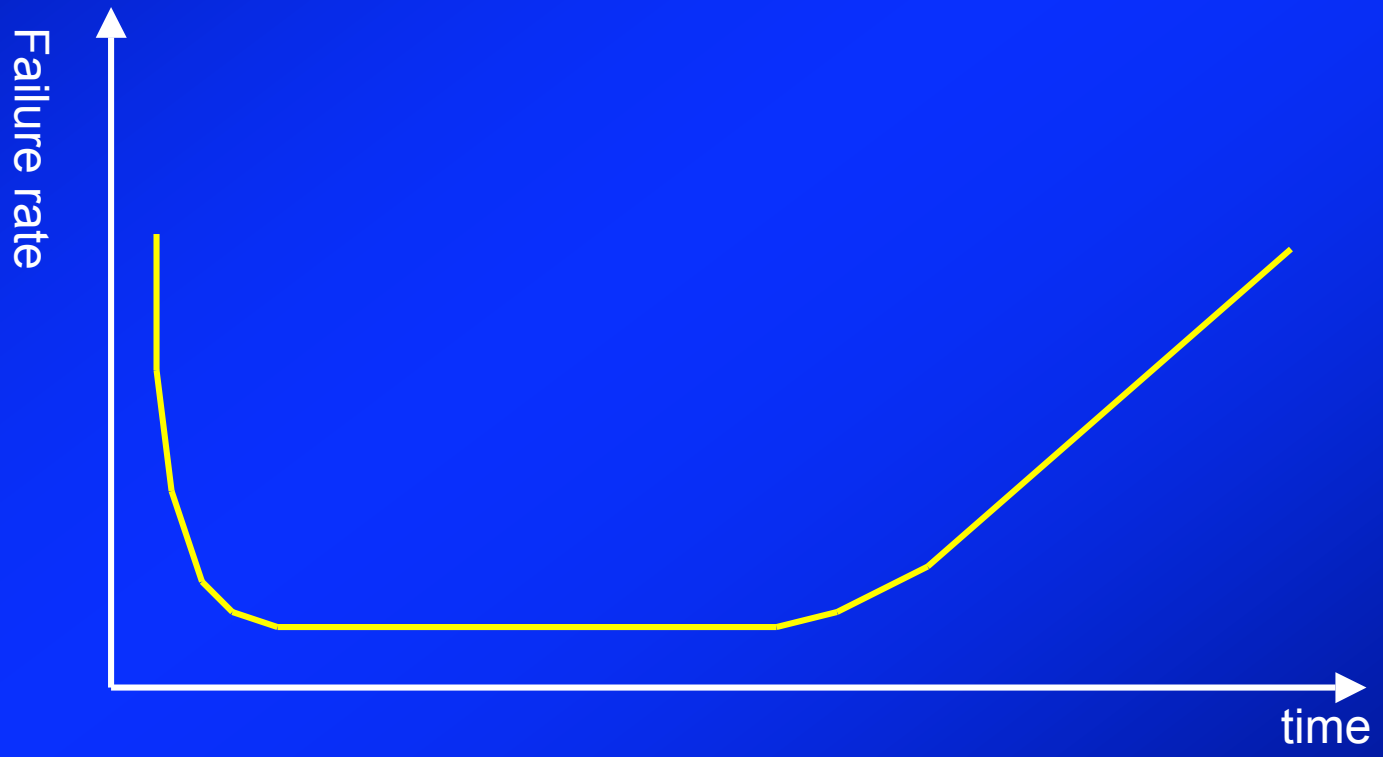
Aging of materials

- Failure mechanisms
  - Electromigration
  - Oxide Breakdown
  - Thermo-mechanical stress
- Temperature dependence
  - Arrhenius law





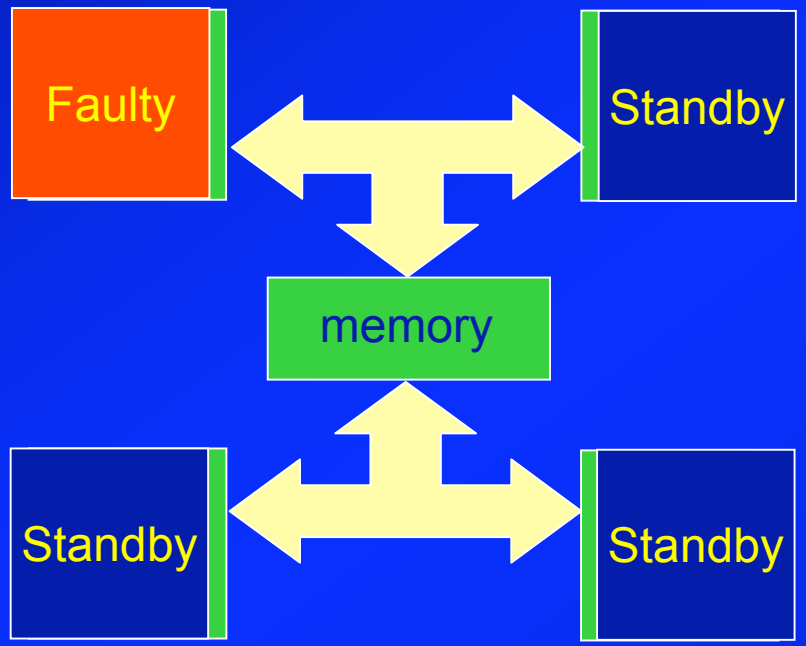
Failure rate
the bathtub curve



Component redundancy

- Component redundancy for enhanced reliability
  - Energy consumption penalty may be severe
- Power-managed standby components
  - Provide for temporary/permanent back-up
  - Provide for load and stress sharing
- Power management and reliability are intertwined:
  - PM allows reasonable use of redundancy on chip
  - Failure rates depend on effect of PM on components
- A programmable and flexible interconnection means is required

Example



When core operates
failure rate is higher as compared to standby unit

When core fails,
it is replaced by standby core

System management may alternate cores at high frequency, voltage and failure rate, to optimize long term reliability

Issues

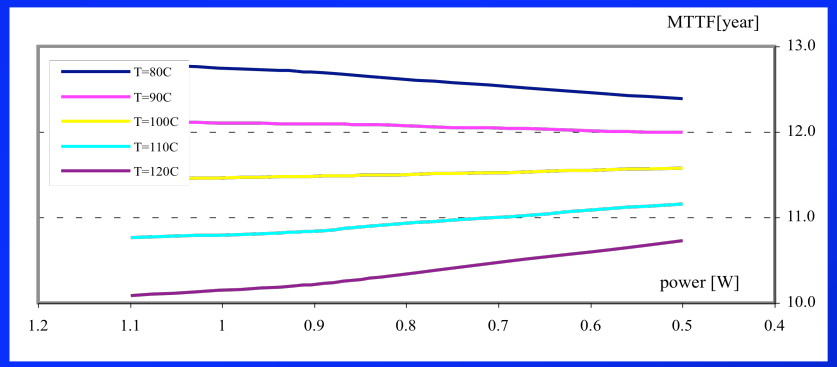
- Analyze system-level reliability
  - as a function of a power management policy
- Determine a system management policy
  - to maximize reliability (over a time interval) and minimize energy consumption
- Determine a system management policy and system topology
  - to maximize reliability (over a time interval) and minimize energy consumption

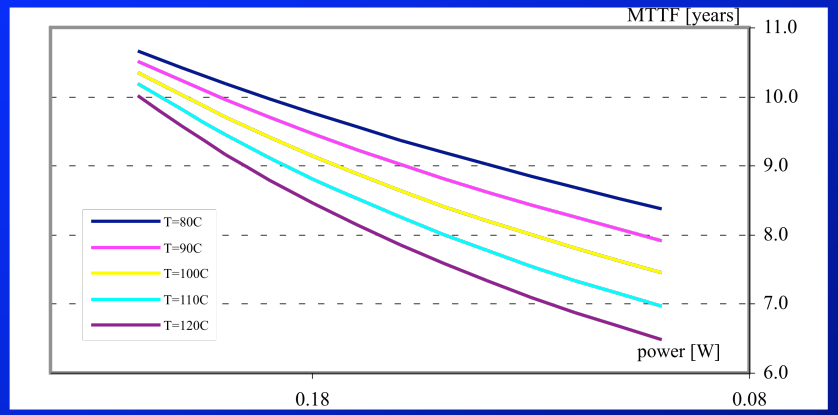
System-level management

- Reliability and energy management can be modeled by stochastic processes
  - Stochastic optimum control for policy design
  - As more accurate models are required, policy design is harder
- Simulation of system management policies is useful for assessing effectiveness of redundancy and energy cost
  - Simulation results show dominant effect of temperature and its cycling on system reliability
- Optimal policy design is also possible

Effect of DPM policy on MTTF

- Power and temperature gap between active and sleep state
- Small gap
  - Thermal cycle effects dominate EM and TDDB only in the lower temperature spectrum
  - MTTF decreases/increases as DPM gets more aggressive
- Wider gap
  - Thermal cycles effects dominate
  - MTTF decreases always as DPM gets more aggressive





[Simunic – UCSD]

Outline

- Introduction and motivation
- Variability and robustness
  - Self-calibrating circuits
- Soft failures
  - Error detection and correction
- Hard failures
  - Redundancy
  - System-level management
- Hardware system integration issues
  - On-chip networks
  - Self-healing
- Nanotechnology challenges
- Summary and conclusions

Extremely large scale circuits
Component-based design

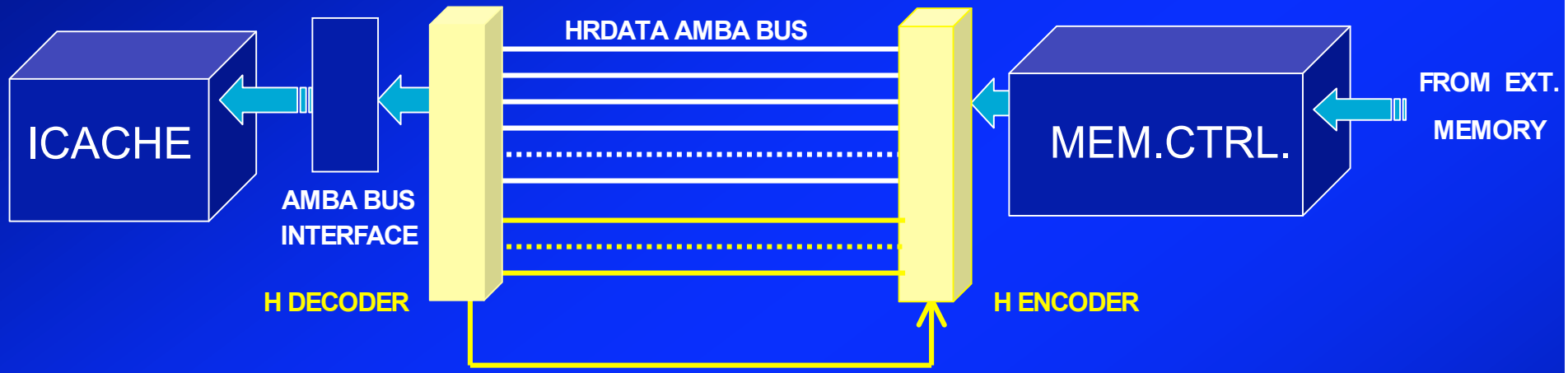
- SoCs are designed (re)-using large macrocells
  - Processors, controllers, memories…
  - *Plug and play* methodology is very desirable
  - Components are qualified before use
- Design challenge:
  - Provide a functionally-correct, reliable operation of the interconnected components

Critical issue:
- Design of the communication fabric

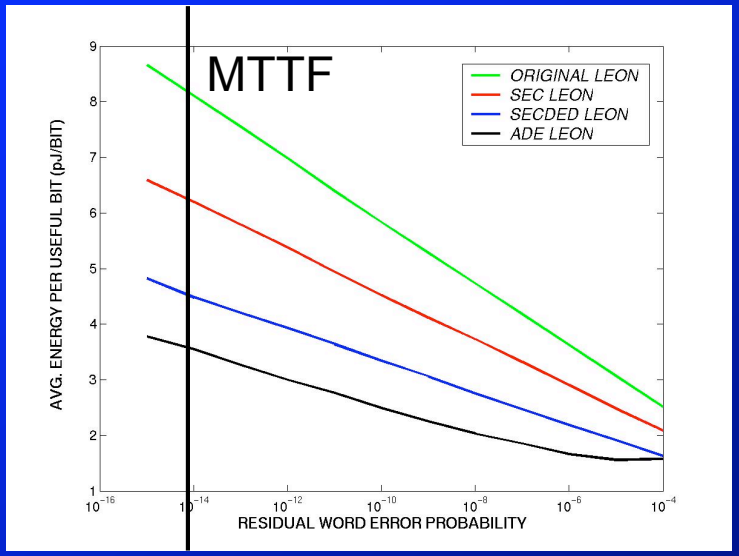
Why on-chip networking ?

- Provide a structured methodology for realizing on-chip communication schemes
  - Modularity
  - Flexibility
- Cope with inherent limitations of busses
  - Performance and power of busses do not scale up
- Support reliable operation
  - Layered approach to error detection and correction

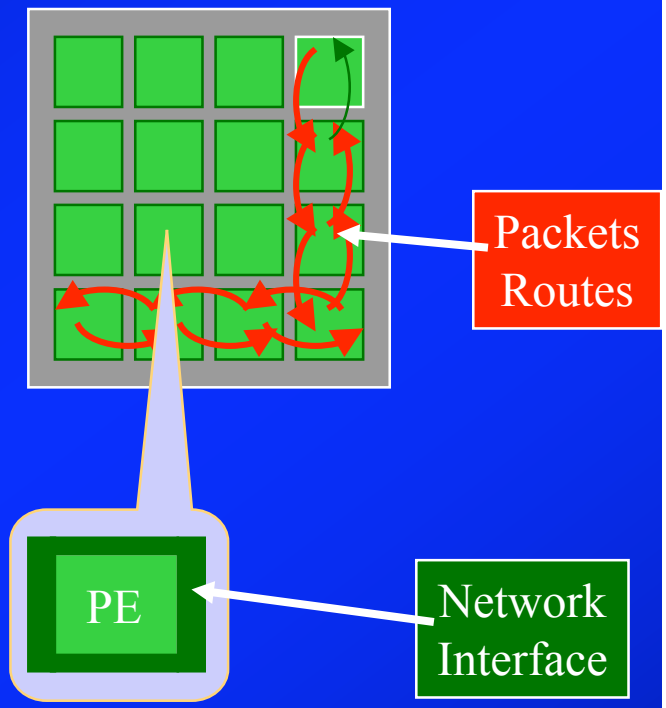
Data-link protocol example: error-resilient coding



- Compare original AMBA bus to extended bus with error detection and correction or retransmission
  - SEC coding
  - SEC-DED coding
  - ED coding
- Explore energy efficiency [Bertozzi]

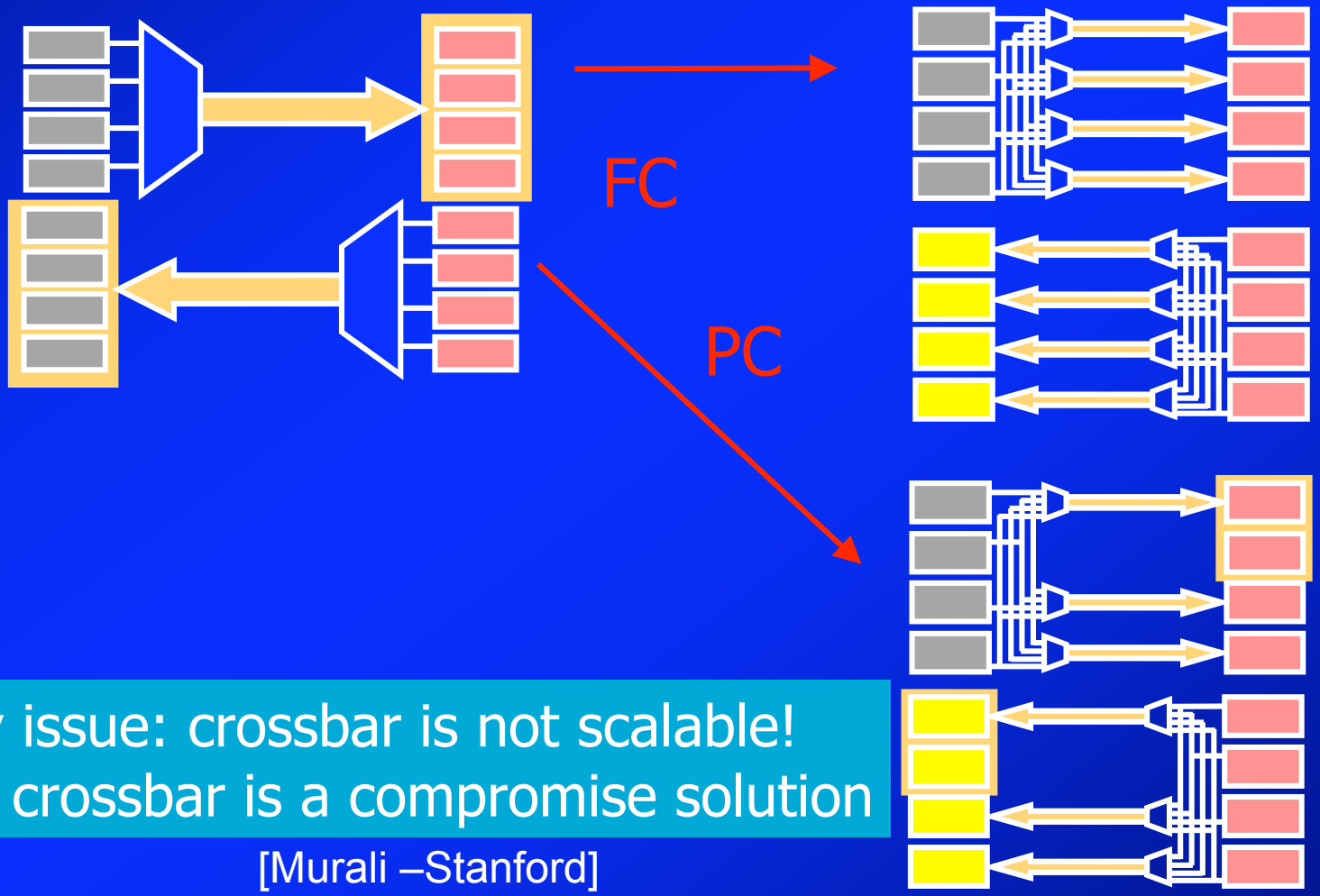


Going beyond buses



- Buses:
  - Pro: simple, existing standards
  - Con: performance, energy, arbitration
- Other network topologies:
  - Pro: higher performance, experience with MP
  - Con: physical routing, need for network and transport layers
- Challenges:
  - Exploit network architecture and corresponding protocols
  - Design network and transport protocols with small overhead
    - Switching, routing, packetization, …

Crossbar & Partial CB cost

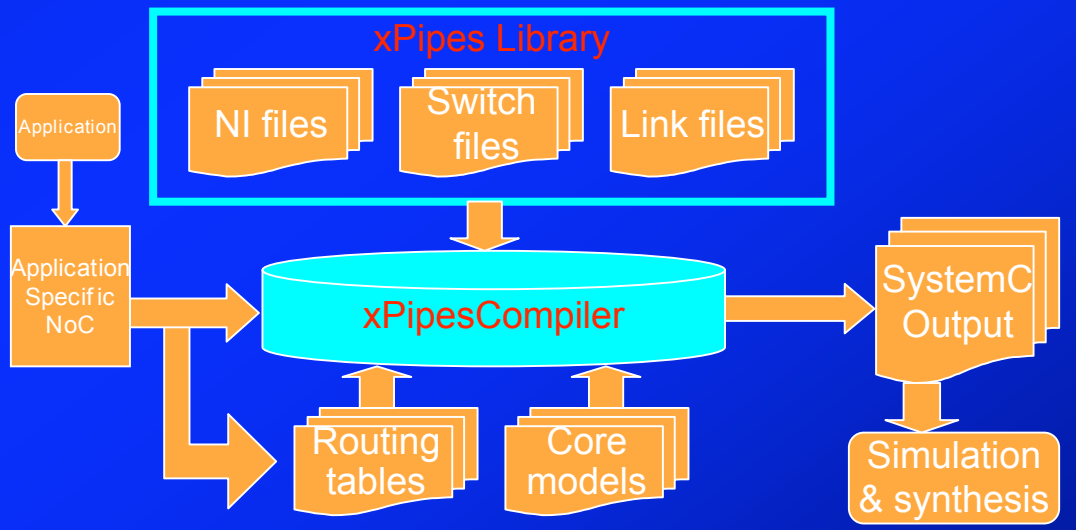


Key issue: crossbar is not scalable!
Partial crossbar is a compromise solution

[Murali –Stanford]

Micro-network synthesis

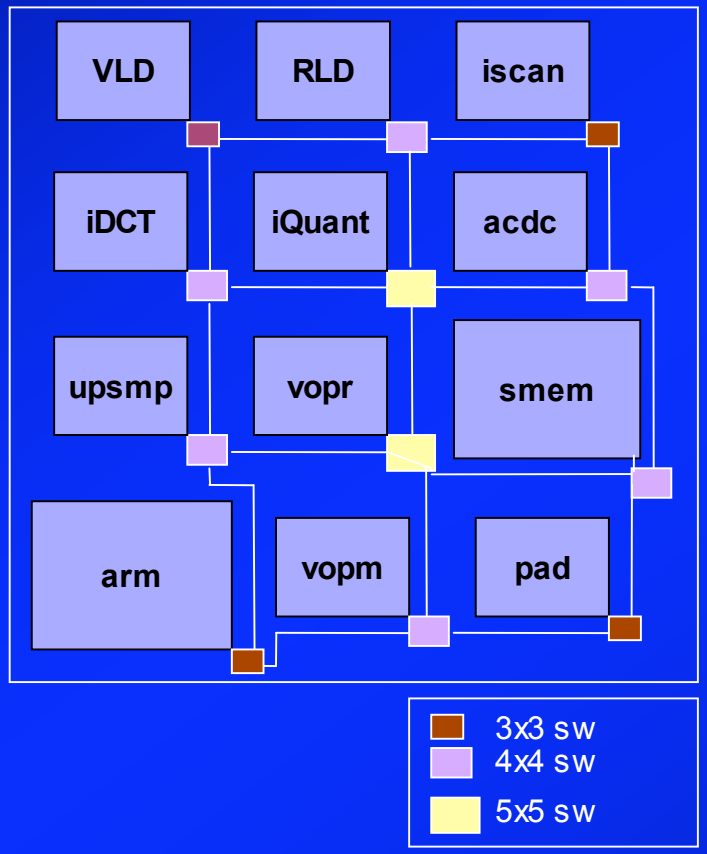
- xPipes:
  - A library of components for micro-network synthesis
    - Network interfaces
    - Switches
    - Communication links
- xPipesCompiler:
  - An assembler to synthesize micro-network from a structural description
  - Output is a SystemC micro-network model



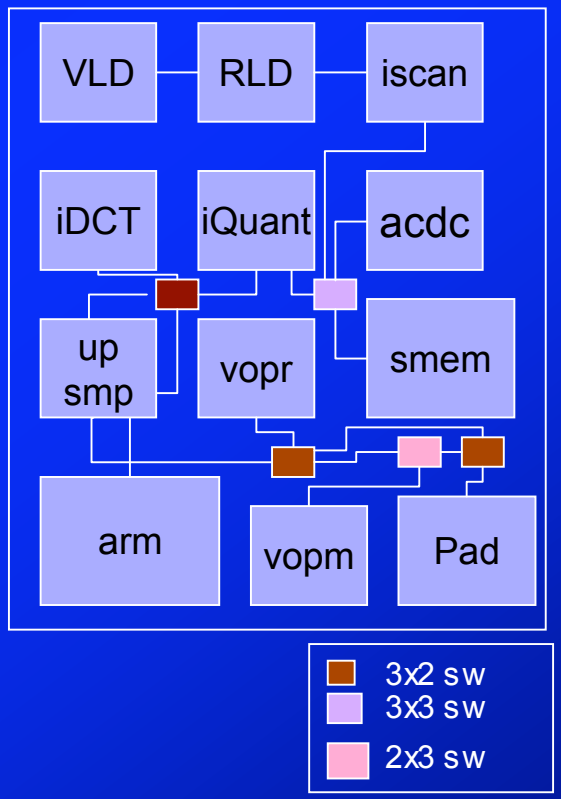
Provide an integrated flow to support micro-network design

Video object plane decoder

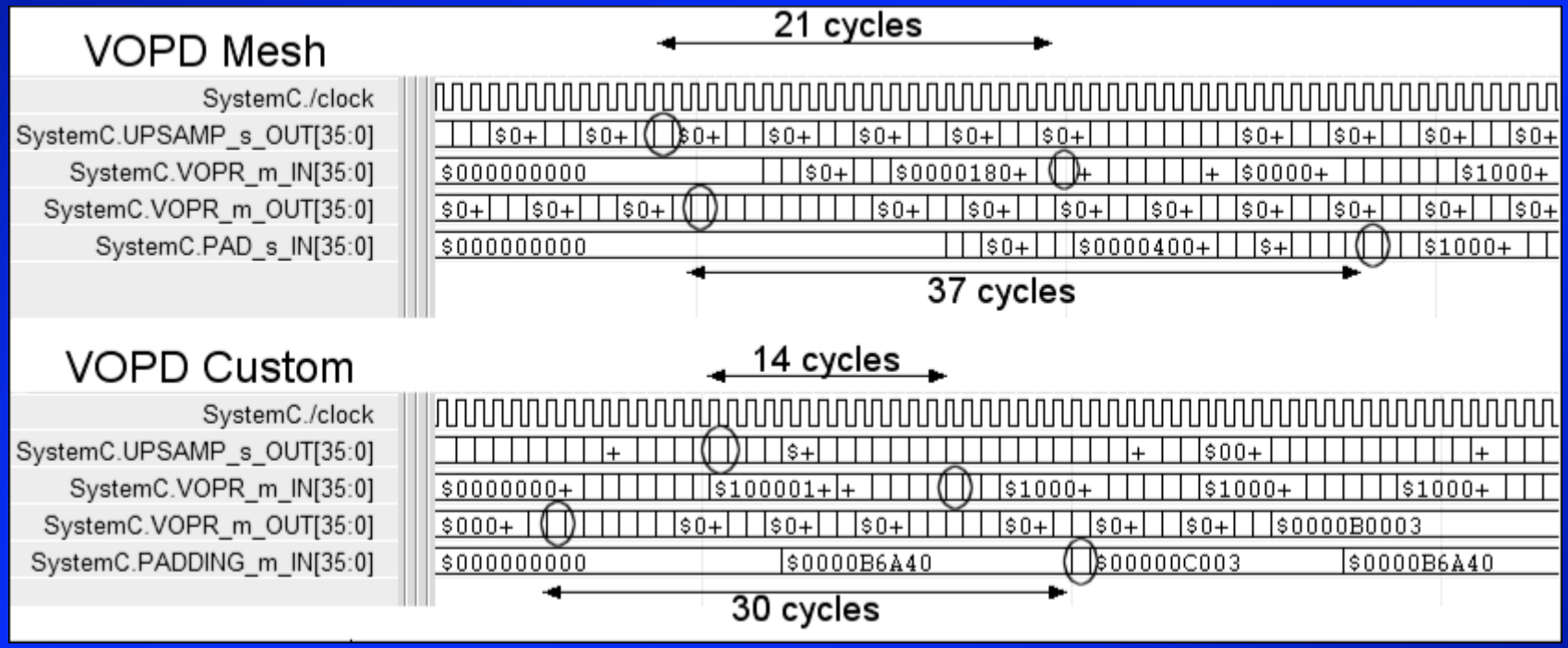
Regular network



Ad hoc network



VOPD Simulation

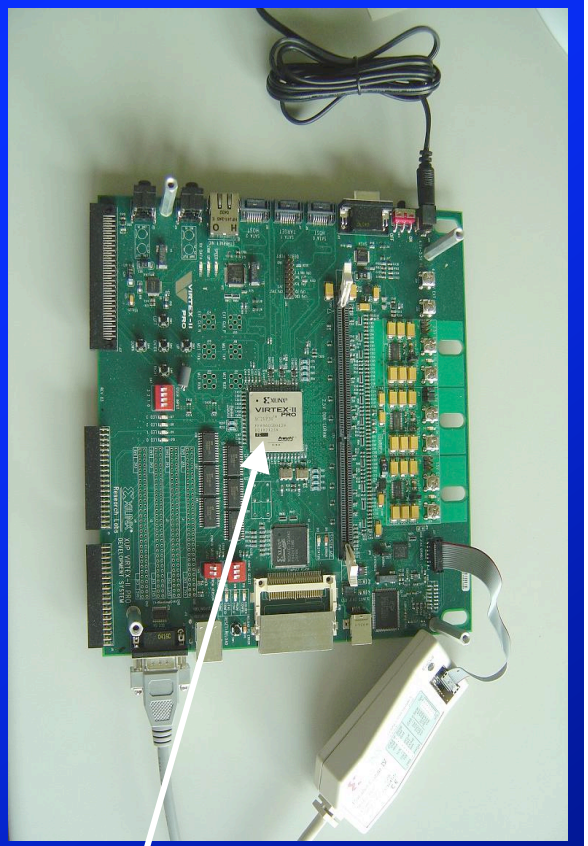


Arrows show the time that packet heads take from source NI to destination

NoC Emulation on FPGA

- Emulation on FPGA enables functional and performance validation of NoC based systems
  - Accurate execution model
  - Probing for profiling and gathering of statistics
- The emulation can achieve important speedups compared to cycle accurate simulation:
  - Up to four orders of magnitude faster
  - Real inputs with millions of packets can be used
- Two-level network emulation
  - Network backbone in configurable hardware
  - Network protocol and parameters in software

  [Genko – EPFL]



Virtex-II Pro FPGA
- 2 Power PC Cores
- 3 M programmable gates

Micro-networks and reliability

- Micro-networks are the platform to integrate multi-processor systems
- Micro-networks support seamlessly redundant stand-by components for achieving high availability
- Micro-networks provide fault-tolerant communication by supporting alternative paths

Self-healing

- Bio-wall project [Mange – EPFL]
- Cellular design with redundancy
  - Each cell programmed by a string (gene)
  - FPGA technology
- Self-healing property:
  - Upon cell failure, neighbors reconfigure to take over function

Cellular self-repair

Cellular self-repair

Autonomic computing

- Broad R&D project launched by IBM
- Self-healing
  - Design computer and software that perform self-diagnostic functions and can fix themselves without human intervention
  - Strong analogies to biological systems
- Reduced cost of design and maintenance

Autonomics principles

- An autonomic system:
  - must know itself
  - reconfigures itself under varying condition
  - optimizes its operations at run time
  - must support self-healing
  - must defend itself against attacks
  - must know the environment
  - manages and optimizes internal resources without human intervention

Outline

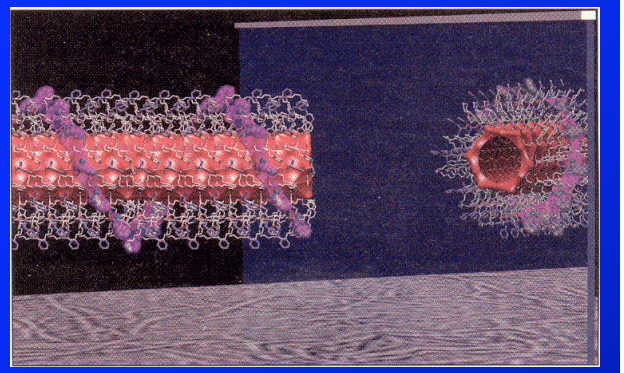
- Introduction and motivation
- Variability and robustness
  - Self-calibrating circuits
- Soft failures
  - Error detection and correction
- Hard failures
  - Redundancy
  - System-level management
- Hardware system integration issues
  - On-chip networks
  - Self-healing
- Nanotechnology challenges
- Summary and conclusions

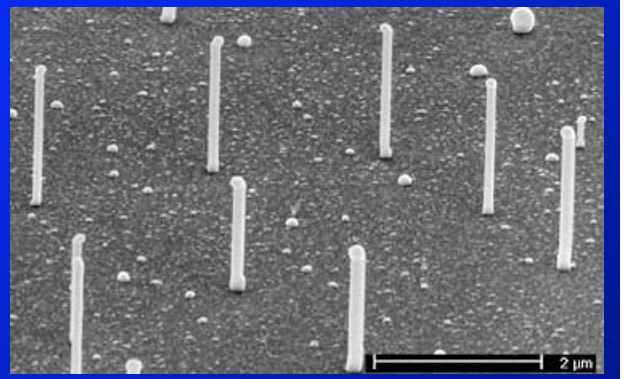
New computing materials

- When will current semiconductor technologies run out of steam?
- What factor will provide a radical change in technology?
  - Performance, power density, cost?
- Several emerging technologies:
  - Carbon nanotubes, nanowires, quantum devices, molecular electronics, biological computing, …
- Are these technologies compatible with silicon?
  - What is the transition path?
- What are the common characteristics, from a design technology standpoint?

Common characteristics of nano-devices

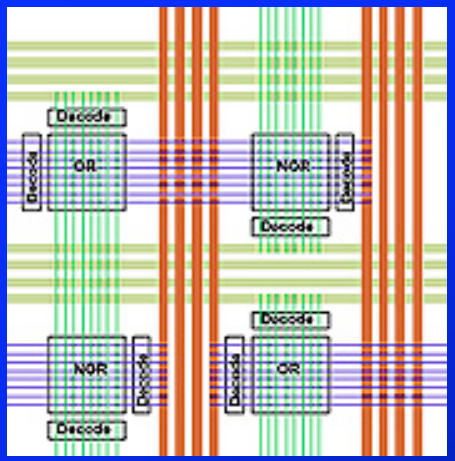
- Self-assembly used to create structures
  - Manufacturing paradigm is bottom-up
- Significant presence of physical defects
  - Massively fault-tolerant design style
- Competitive advantage stems from the high density of computing elements
  - Two orders up as compared to scaled CMOS





Robust nano-design

- Key ingredients:
  - Massive parallelism and redundancy
  - Exploit properties of crosspoint architectures
    - E.g., Programmable Logic Arrays (PLAs)
  - Local and global reconfiguration
  - Some nanotechnologies are compatible with CMOS



Some design technologies for robust DSM CMOS design can be applied to nanotechnology

Outline

- Introduction and motivation
- Variability and robustness
  - Self-calibrating circuits
- Soft failures
  - Error detection and correction
- Hard failures
  - Redundancy
  - System-level management
- Hardware system integration issues
  - On-chip networks
  - Self-healing
- Nanotechnology challenges
- Summary and conclusions

Summary

- The electronic market is driven by embedded applications where reliability and robustness are key figures of merit
- Hardware systems are more prone to fail
  - Variations in manufacturing
  - Hard and soft malfunctions
- Reliability can be enhanced by component and communication redundancy
  - System management is critical for long-lasting operation
  - On-chip networks support redundancy
- Self-healing, massive parallelism and redundancy are key to design highly-dependable circuits with nano-technologies
  - Sub 45nm CMOS technologies
  - Novel silicon and non-silicon based nano-technologies


Thank You!
Merci!