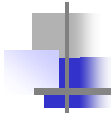






Robust Computing in the Nanoscale Era





Todd Austin
University of Michigan
austin@umich.edu

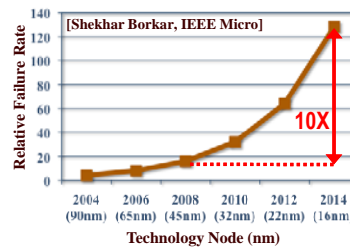
A Scenario Not That Far Away

Scenario: The year is 2016, in the whole world we are using more than 100 billion devices with microprocessors and suddenly microprocessors start to fail. They fail in big numbers...

"In the future we will need to design reliable systems with unreliable components"
Pradip Bose, IBM Research



"Reliability will be the barrier to future scaling"
Shekhar Borkar, Intel Fellow



"Chips will have tens of billions of transistors, but many of them might be unusable and others will slowly age and degrade over time"
Shekhar Borkar, Intel Fellow

"Reliability will be a first-class design constraint"
Chuck Moore, AMD Senior Fellow

What If That Scenario Happened Today?

Consumer Electronics



- Affect user experience
- Frequent system crashes
- Lower customer satisfaction
- Stain company credibility

Corporate Computing

- Millions of dollars for downtime
- Lower productivity
- Higher IT management costs
- Less trust in computing systems



Data Centers

- Lower performance
- Break quality of service contracts
- Dissatisfy customers with lower availability
- Higher repair and management cost



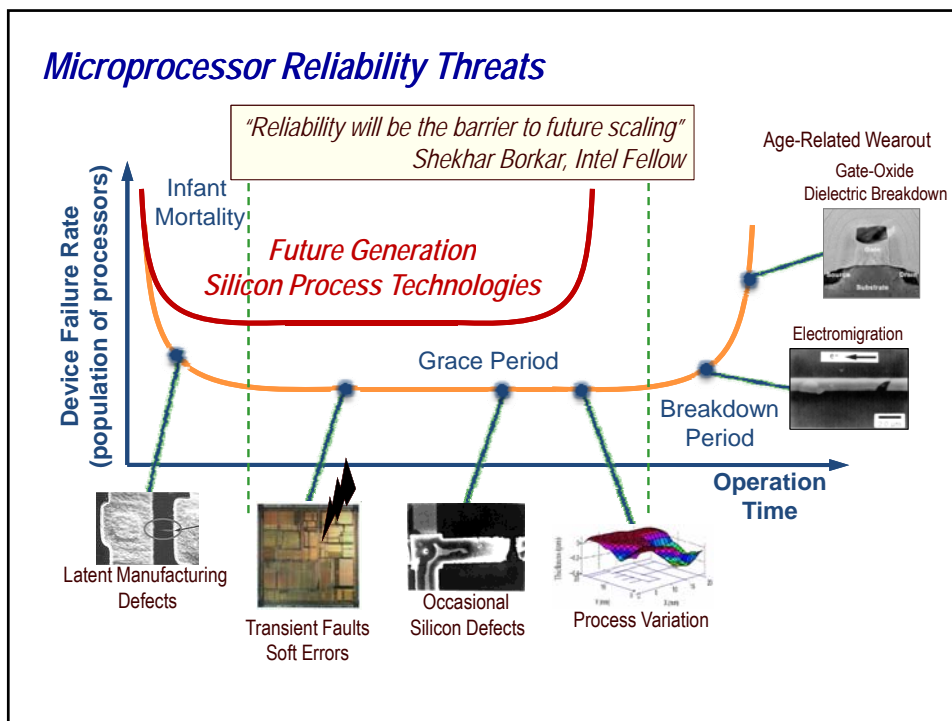
Why does robustness matter?

- ... the ability to consistently resolve critical dimensions of 30nm is severely compromised creating substantial uncertainty in device performance. ... at 30nm design will enter an era of “probabilistic computing,” with the behavior of logic gates no longer deterministic...
- susceptibility to single event upsets from radiation particle strikes will grow due to supply voltage scaling while power supply integrity (IR drop, inductive noise, electromigration failure) will be exacerbated by rapidly increasing current demand
- new approaches to robust and low power design will be crucial to the successful continuation of process scaling ...

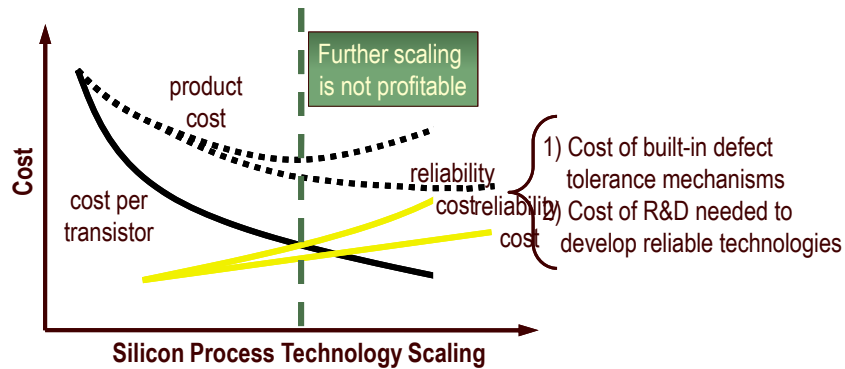
Intel chairman Andrew Grove *Int. Electron Devices Meeting* keynote
Dec. 2002

Tutorial Schedule

- **Reliability Issues: SER, Variability and Defects**
- **Fault Tolerant Design Techniques**
 - Classical Techniques
 - SER Specific Techniques
 - Full-Spectrum Techniques
 - Research Topic: Self-Healing Systems
- **Robust Low-Power Design Techniques**



Reliability Challenges of Technology Scaling



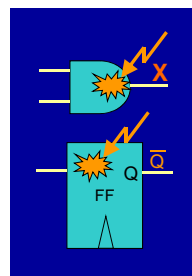
- 1) Build microprocessors out of unreliable transistors/technologies
- 2) Provide reliability through very low cost defect tolerance techniques

Fault Classes

- **Permanent fault (hard fault)**
 - Irreversible physical change
 - Latent manufacturing defects, Electromigration
- **Intermittent fault**
 - Hard to differentiate from transient faults
 - Repeatedly occurs at the same location
 - Occurs in bursty manners when fault is activated
 - Replacing the offending circuit removes faults
- **Transient faults (Soft Errors)**
 - Neutron/Alpha particle strikes
 - Power supply and Interconnect noises
 - Electromagnetic interference
 - Electrostatic discharge

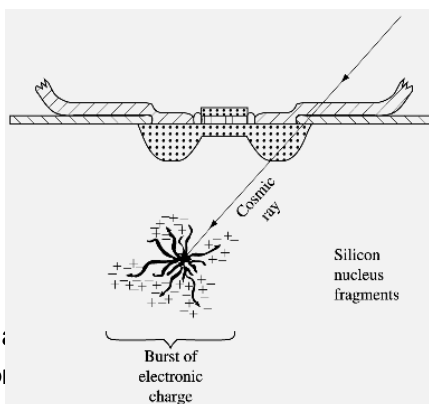
Introduction – Soft Errors

- ◆ *Soft errors*, also called *transient faults* and *single-event upsets*(SEU)
 - Processor execution errors caused by high-energy neutrons resulting from cosmic radiation and alpha particles radiation
 - Appears to be a reliability threat for future technology processors
- ◆ When a particle strikes a circuit element a small amount of charge is deposited
 - *Combinational logic node*: a very short duration pulse of current is formed at the circuit node
 - *State holding element (FF/SRAM cell)*: flip the stored value
- ◆ Unlike permanent faults the effects of soft errors are transient



Soft Errors (SER)

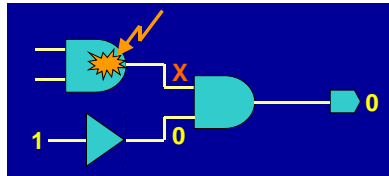
- **Alpha particles stemming from radioactive decay of packaging materials**
- **Neutrons (cosmic rays) are always present in the atmosphere**
- **Soft errors are transient non-recurring faults (also called single event upsets, SEUs) where added/deleted charge on a node results in a functional error**
 - Charge is added/removed by electron/hole pairs absorbed by source/drain diffusion areas



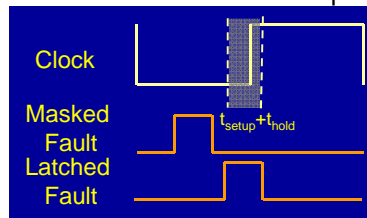
Source: S. Mukherjee, Intel

Soft Error Masking

- ◆ *Logic Masking*: the fault gets blocked by a following gate whose output is completely determined by its other inputs

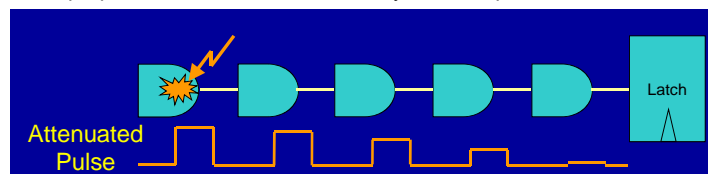


- ◆ *Timing Masking*: the fault affects the input of a latch only in the period of time that the latch is not sensitive to its input



Soft Error Masking

- ◆ *Electrical Masking*: the fault's pulse is attenuated by subsequent logic gates due to electrical properties, and does not affect any latch's input



- ◆ *Microarchitectural Masking*: the fault alters a value of at least one flip-flop, but the incorrect values get overwritten without being used in any computation affecting the design's output
- ◆ *Software Masking*: the fault propagates to the design's output but is subsequently masked by software without affecting the application's correct execution

How To Measure Reliability: Soft Error Rate (FIT)

- **Failure In Time (FIT) : Failures in 10^9 hours**
 - **114 FIT means**
 - 1 failure every 1000 years
 - It sounds good, but
 - If 100,000 units are shipped in market, 1 end-user per week will experience a failure
- **Mean Time to Failure : $1 / \text{FIT}$**

Soft Error Considerations

- **Highly elevation dependent (3-5X higher in Denver vs. sea-level, or 100X higher in airplane)**
- **Critical charge of a node (Q_{crit}) is an important value**
 - Node requires Q_{crit} to be collected before an error will result
 - The more charge stored on a node, the larger Q_{crit} is (Q_{crit} must be an appreciable fraction of stored Q)
 - Implies scaling problems → caps reduce with scaling, voltage reduces, so stored Q reduces as S^2 (~ 2X) per generation
 - Ameliorated somewhat by smaller collection nodes (S/D junctions)
 - But exacerbated again by 2X more devices per generation

Soft Error Rate Trends, ITRS03

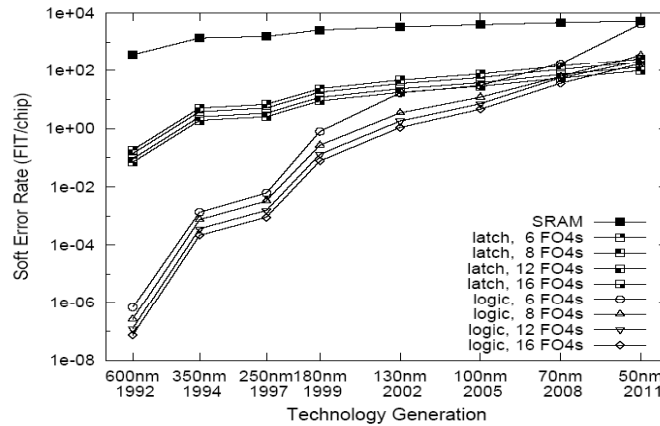
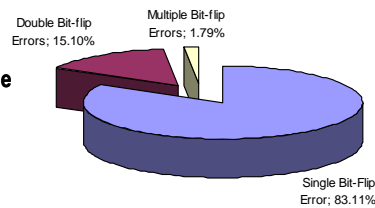


Figure 14. SER/chip for SRAM/latches/logic

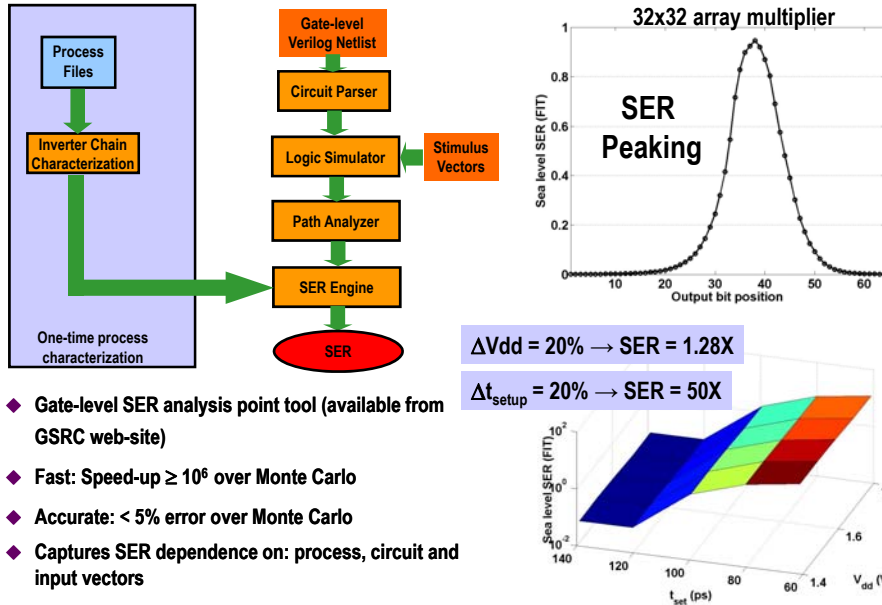
Impact of Soft Errors in Processors [Iyer]

- ◆ How do soft errors in processors propagate and impact applications?
- ◆ Approach
 - Fault injections (with *i-Measure*, hardware level fault injection framework) in combinational logic and flip-flops of MIPS and Alpha-like processors
 - Study fault propagation to the application level
- ◆ Major findings:
 - Nearly 5% of faults in combinational logic propagate to state of the processor
 - Errors in *Control* contribute to 79% of application hangs
 - Errors in *Execution* blocks a major factor in application crashes (45%) and silent data corruption (40%)
 - Faults in combinational logic can cause double and multiple bit errors

Multiple Bit-flip Distribution in Alpha processor

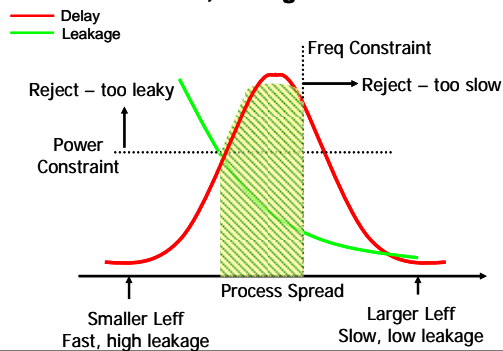
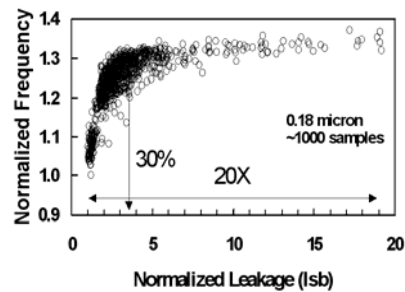


SERA SER Analysis Tool [Shanbhag]



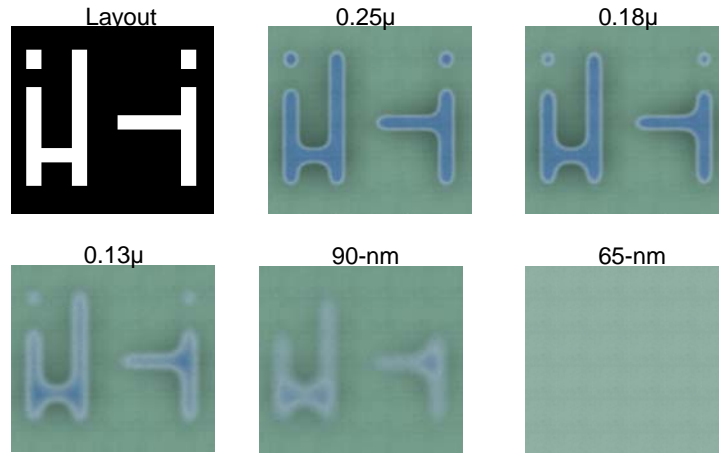
Effects Of Variability

- High-performance processors are speed-binned
 - Faster == more \$\$\$
 - These parts have small L_{eff}
- Exponential dependence of leakage on V_{th}
 - And L_{eff} , through V_{th}



Since leakage is now appreciable, parametric yield is being squeezed on both sides

Printing in the Subwavelength Regime



Figures courtesy Synopsys Inc.

Variation: Across-Wafer Frequency

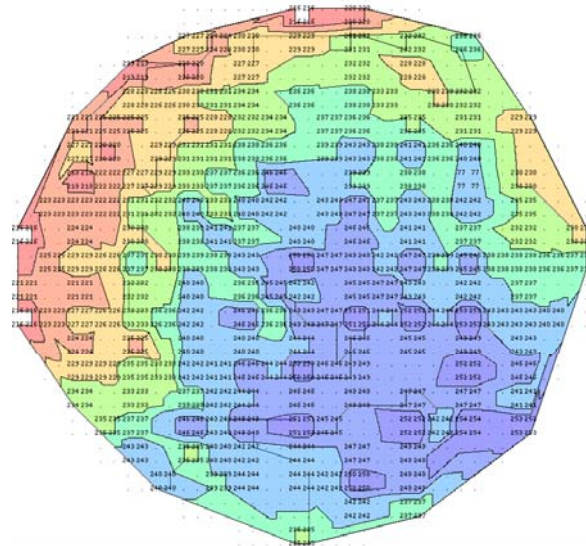
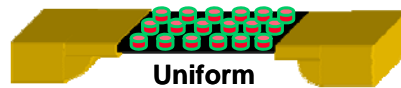
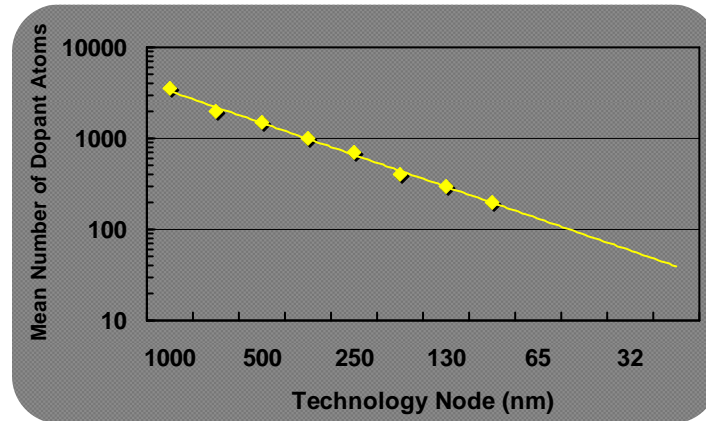
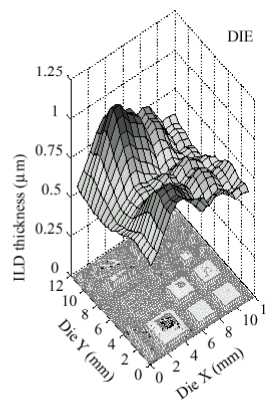
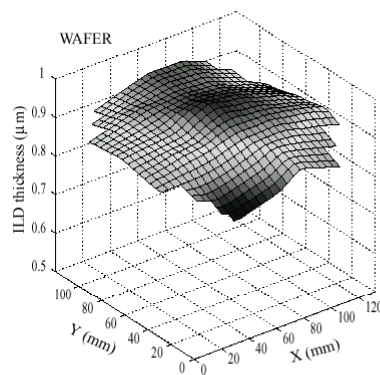


Figure courtesy S. Nassif, IBM

Random Dopant Fluctuations, Intel's View



Inter-die vs. Intra-die Variation

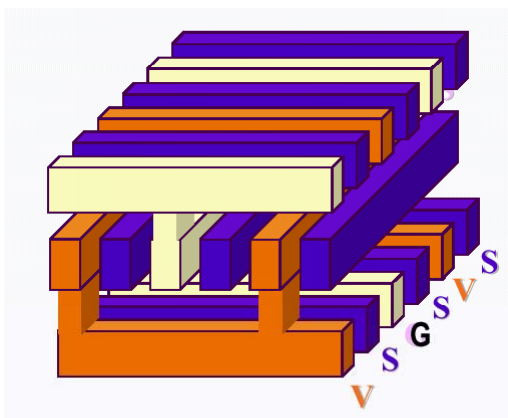


Inter-die variation is not always larger than intra-die (ILD)

Design/EDA for Highly Variable Technologies

- **Critical need: Move away from deterministic CAD flow and worst-case corner approaches**
- **Examples:**
 - **Probabilistic dual-Vth insertion**
 - **Low-Vth devices exhibit large process spreads; speed improvements and leakage penalties are thus highly variable**
 - **Parametric yield optimization**
 - **Making design decisions (in sizing, circuit topology, etc.) that quantitatively target meeting a delay spec AND a power spec with given confidence**
 - **Avoid designing to unrealistic worst-case specs**
 - **Use other design tweaks such as gate length biasing (next)**

Noise Immune Layout Fabric



Major area penalty (>60%)

This layout style trades off area for:

- **Noise immunity (both C and L)**
- **Minimizes variations (CMP)**
- **Predictable**
- **Easy layout**
- **Simplifies power distribution**

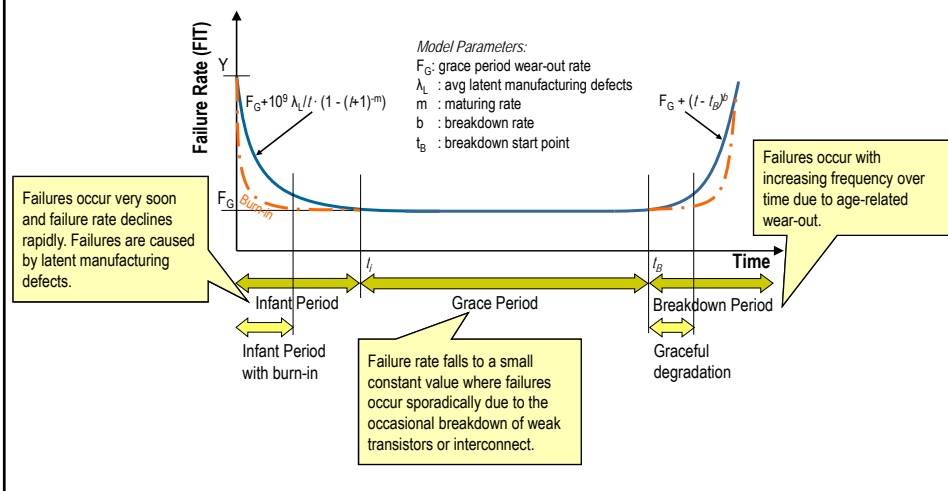
Ref: Khatri, DAC99

Defects: The (Bumpy) Road Ahead for Silicon

- ◆ What is the failure model of silicon 2-3 generations out?
 - ◆ What the literature says...
 - ◆ “Expected failure rate of 10^{12} hours/device”, this would give a high end NVidia graphics part an expected lifetime of less than 1 year
 - ◆ “Failure rates higher than 10^{20} hours/device”, which eliminates the problem
 - ◆ What the experts say...
 - ◆ Intel [Borkar] and IBM [Bernstein]: critical problem for future silicon
- ◆ Key failure modes
 - ◆ Transistor wear-out (aggravated by scaling)
 - ◆ SER-related upsets (especially in logic)
 - ◆ Early transistor failures (due to ineffective burn-in)
 - ◆ Untestable defects (compounded by complexity)

Silicon Defects: Sources and Trajectory

- ◆ Sources: gate wearout, NBTI, hot electrons, electro-metal migration, etc...



Tutorial Schedule

- ◆ **Reliability Issues: SER, Variability and Defects**

- ◆ **Fault Tolerant Design Techniques**
 - ◆ Classical Techniques
 - ◆ SER Specific Techniques
 - ◆ Full-Spectrum Techniques
 - ◆ Research Topic: Self-Healing Systems

- ◆ **Robust Low-Power Design Techniques**

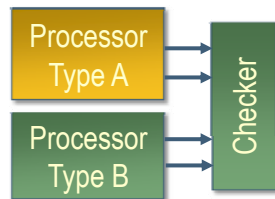
Techniques For Improving Reliability

- **Fault avoidance (Process / Circuit)**
 - Improving materials
 - Low Alpha Emission interconnect and Packaging materials
 - Manufacturing process
 - Silicon On Insulator (SOI)
 - Triple Well design process to protect SRAM
- **Fault tolerance (robust design in presence of Soft Error) : Circuit / Architecture**
 - Error Detection & Correction relies mostly on “Redundancy”
 - Space : DMR, TMR
 - Time : Temporal redundant sampling (Razor-like)
 - Information : Error coding (ECC)

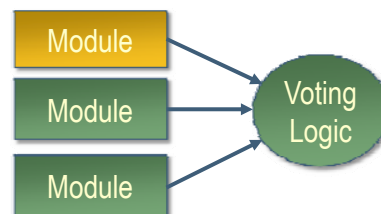
How Do We Protect The Systems Today?

- ◆ Defect tolerance techniques are limited to high-end systems
 - Life-critical applications (*e.g.*, aviation, medical systems)
 - Mission-critical applications (*e.g.*, military, NASA's space exploration)
 - Business-critical applications (*e.g.*, banks, financial sector)

2-Version Hardware



Triple Modular Redundancy



Redundant computation/hardware is too expensive to deploy into cost-sensitive mainstream systems

DMR Error Detection

- Context: **Dual-modular redundancy for computation**
- Problem: **Error detection across blades**

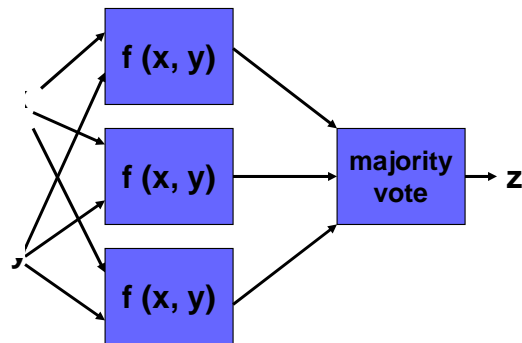


Triple Modular Redundancy (von Neumann)

Voter assumed
reliable!

⇒ voter small

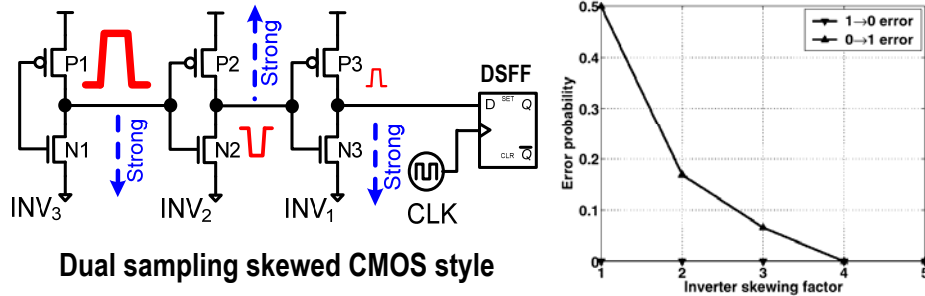
⇒ coarse-grained



Error Coding : Information Redundancy

- **Coding: representation of information**
 - Sequence of code words or symbols
 - Shannon's theorem in 1948
 - In noisy channels, errors can be reduced to a certain degree
 - Golay(1949), Hamming(1950), Stepian(1956), Prange(1957), Huffman
- **Overheads**
 - Spatial overhead : Additional bits required
 - Temporal overhead : Time to encode and decode
- **Terminology**
 - Distance of code
 - Minimum hamming distance between any two valid codewords
 - Code separability (e.g. Parity Code)
 - Code is separable if code has separate code and data fields

SER-Tolerant Circuit Design [Shanbhag]

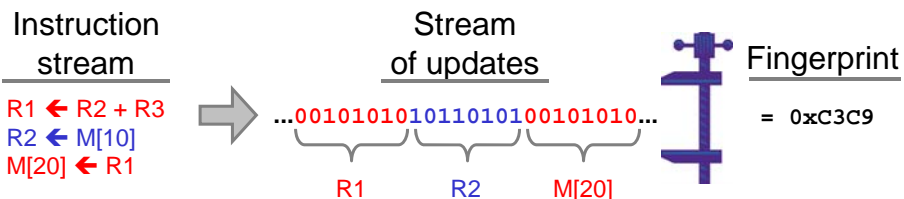


Dual sampling skewed CMOS style

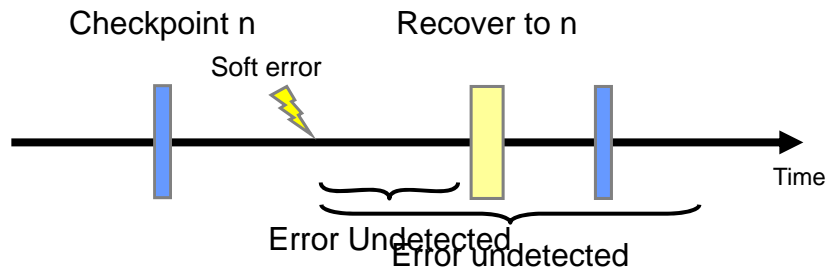
- ◆ Employs skewed CMOS for logic and dual sampling FF (DSFF)
- ◆ Both 0→1 and 1→0 errors are eliminated if skewing factor ≥ 4.
- ◆ Speed penalty
 - depends on Δ (maximum SET width)
 - can be made a design parameter.
 - equals 300ps (for 0.18um process) if zero SER wanted.
- ◆ Power penalty: 17% (DSFF) + 20% (Skewed CMOS)

Fingerprinting [Falsafi/Hoe]

- Hash updates to architectural state
- Fingerprints compared across DMR pair
- Bounded error detection latency
- Reduced comparison bandwidth



Recovery Model



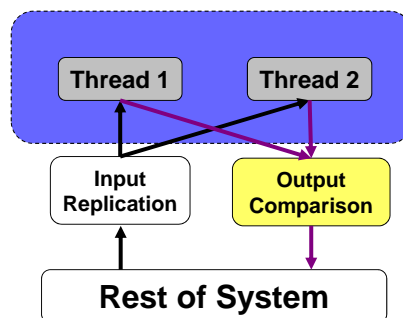
➤ *Rollback-recovery to last checkpoint upon detection*

Simultaneous Redundant Multithreading [Reinhardt]

Logical boundary of redundant execution within a system

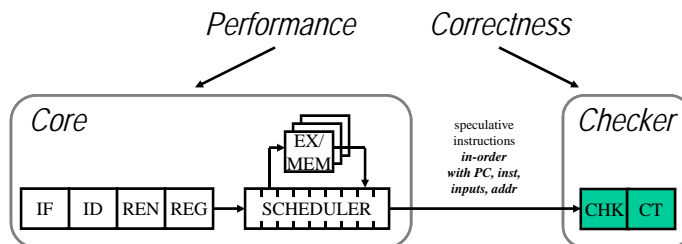
- Trade-off between information, time, & space redundancy

Sphere of Replication



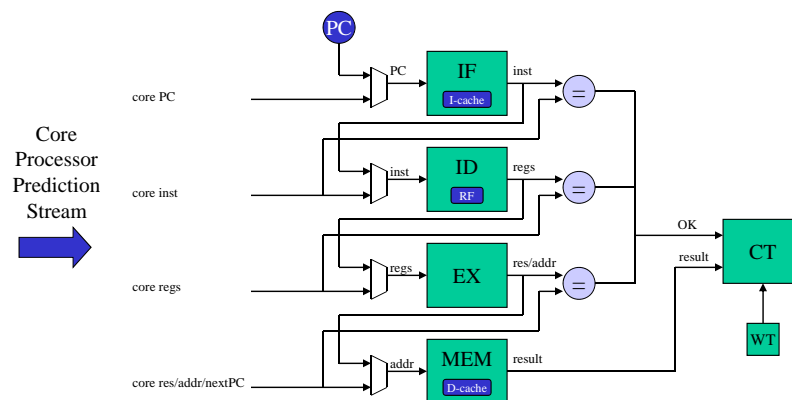
Compare & validate output before sending it outside the SoR

Full-Spectrum Fault Tolerance: DIVA Checker [Austin]

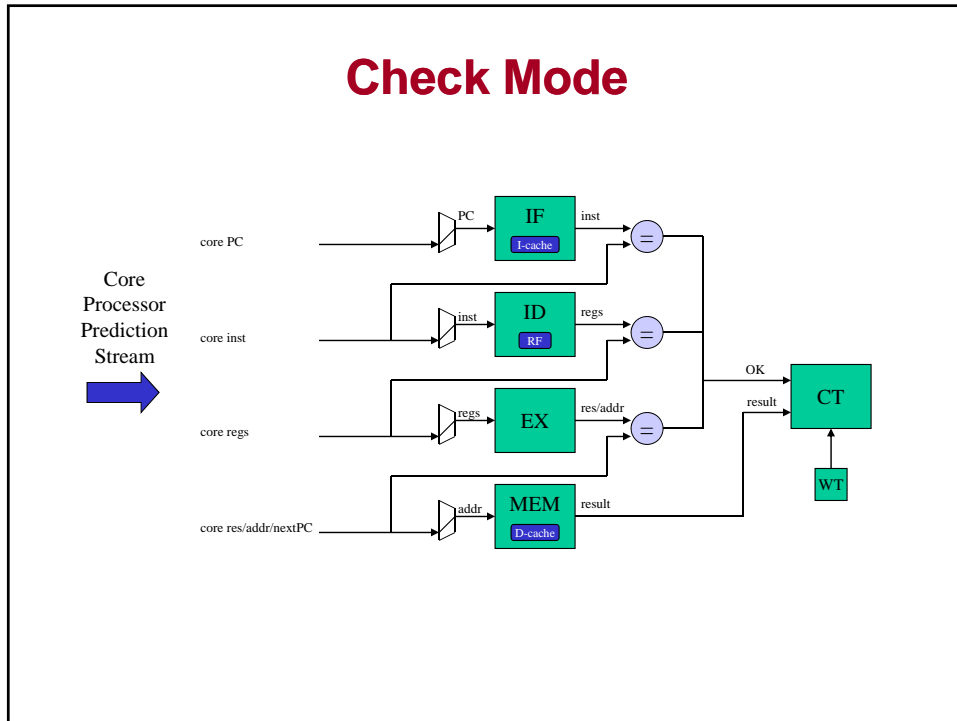


- **All core function is validated by checker**
 - Simple checker *detects* and *corrects* faulty results, restarts core
- **Checker relaxes burden of correctness on core processor**
 - Tolerates design errors, electrical faults, defects, and failures
 - Core has burden of accurate prediction, as checker is 15x slower
- **Core does heavy lifting, removes hazards that slow checker**

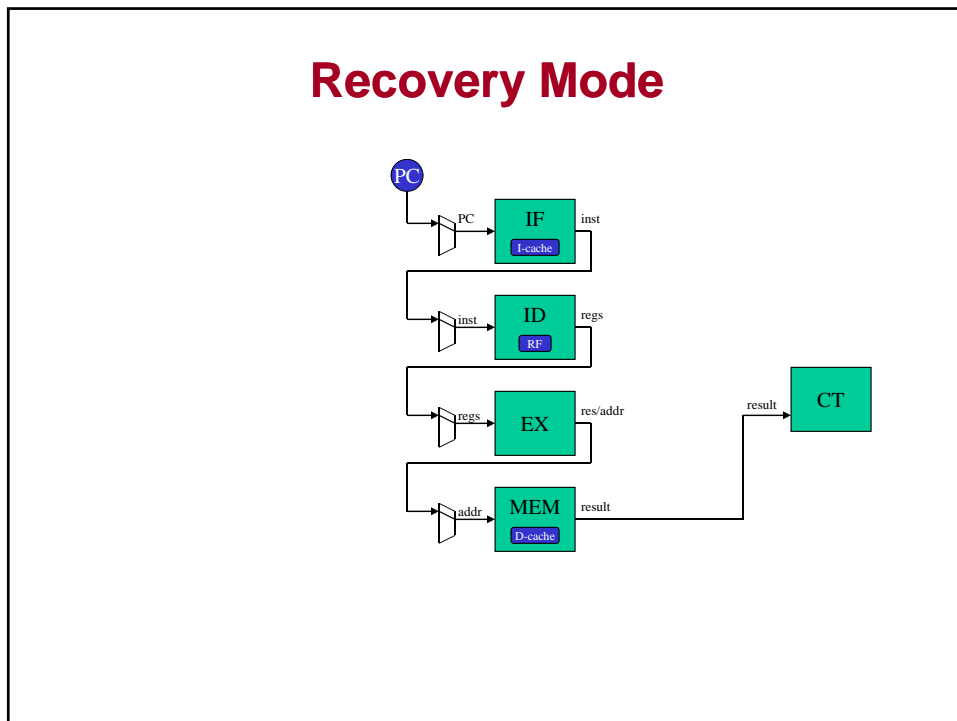
Checker Processor Architecture



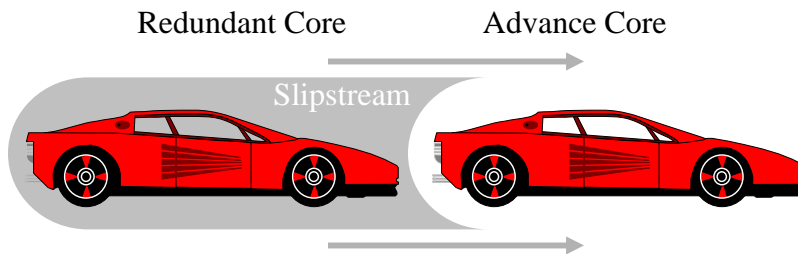
Check Mode



Recovery Mode

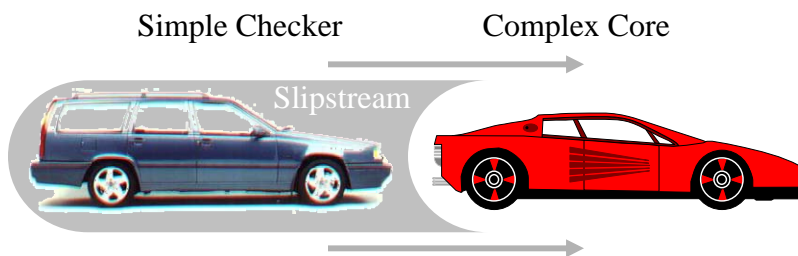


How Can the Simple Checker Keep Up?



- **Slipstream effects reduce power requirements of trailing car**
 - Checker processor executes in the core processor slipstream
 - fast moving air \Rightarrow branch/value predictions and cache prefetches
 - Core processor slipstream reduces complexity requirements of checker
- **Symbiotic effects produce a higher combined speed**

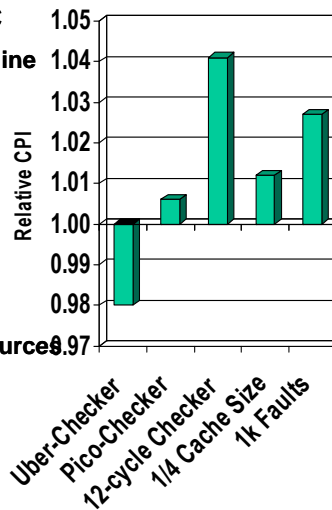
How Can the Simple Checker Keep Up?



- **Slipstream effects reduce power requirements of trailing car**
 - Checker processor executes in the core processor slipstream
 - fast moving air \Rightarrow branch/value predictions and cache prefetches
 - Core processor slipstream reduces complexity requirements of checker
- **Symbiotic effects produce a higher combined speed**

Checker Performance Impacts

- **Checker *throughput* bounds core IPC**
 - Only cache misses stall checker pipeline
 - Core warms cache, leaving few stalls
- **Checker *latency* stalls retirement**
 - Stalls decode when speculative state buffers fill (LSQ, ROB)
 - Stalled instructions mostly nuked!
- **Storage hazards stall core progress**
 - Checker may stall core if it lacks resources
- **Faults flush core to recover state**
 - Small impact if faults are infrequent



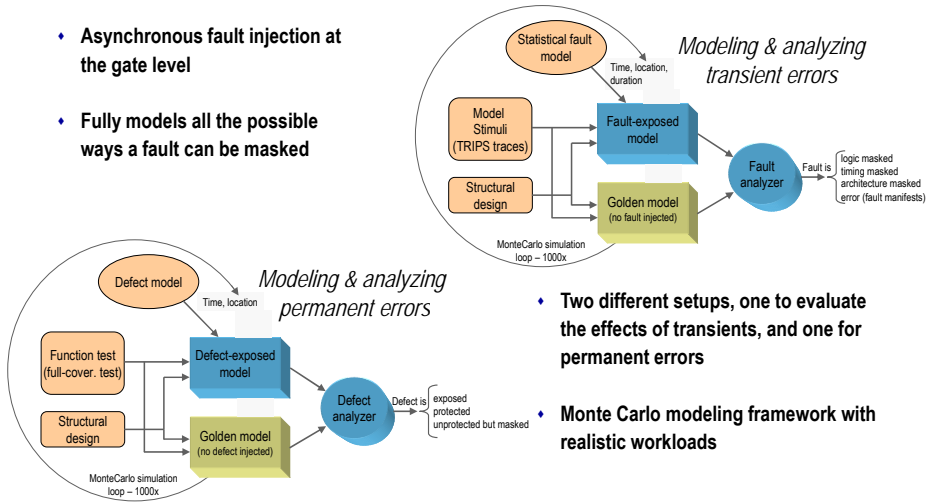
Research Topic: Self-Repairing Systems

- ◆ Defect-tolerant self-repairing systems need to support:
 - Error Detection
 - System Diagnosis (locate the origin of the error)
 - System Repair
 - System Recovery
- ◆ Key idea:
 - Error detection must be performance efficient
 - ◆ Continuously check execution for errors
 - Diagnosis, repair and recovery are insensitive on performance
 - ◆ Get invoked only when an error is detected (rare scenario)
 - ◆ Trade-off performance for more cost efficient techniques

Fault Modeling & Analysis Infrastructure

- ◆ High-performance, high-fidelity, fault modeling simulation infrastructure

- Asynchronous fault injection at the gate level
- Fully models all the possible ways a fault can be masked



- Two different setups, one to evaluate the effects of transients, and one for permanent errors
- Monte Carlo modeling framework with realistic workloads

Self-Repairing BulletProof Silicon [Austin, Bertacco]

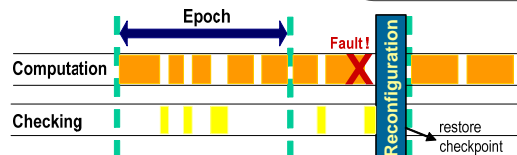
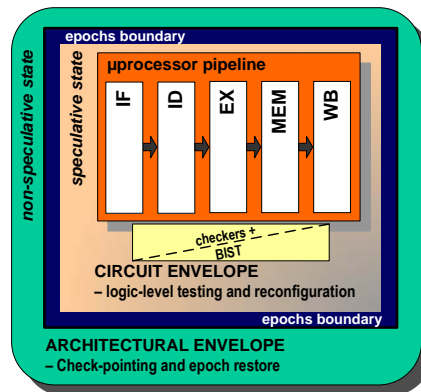
Goal: Single-defect tolerance for 5% area overhead

Key ideas:

- No expensive computation checking
- Protect computation and test Hw
- Repair by disabling redundant parts

Approach:

1. Execute and protect state
2. Test concurrently when Hw idle
3. If **tests fails** → roll back state
 - disable component
 - restart



Tutorial Schedule

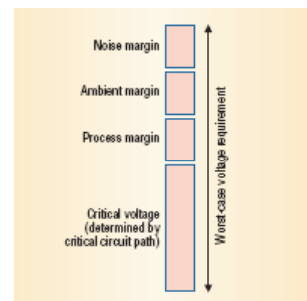
- ◆ Reliability Issues: SER, Variability and Defects

- ◆ Fault Tolerant Design Techniques
 - ◆ Classical Techniques
 - ◆ SER Specific Techniques
 - ◆ Full-Spectrum Techniques
 - ◆ Research Topic: Self-Healing Systems

- ◆ Robust Low-Power Design Techniques

Power and Reliability: How are they related?

- ◆ **The move to smaller features can help with power – with qualifications**
- ◆ **Smaller features increase design margins**
 - ◆ reduce power savings
 - ◆ reduce performance gains
 - ◆ reduced area benefits



Why does power matter?

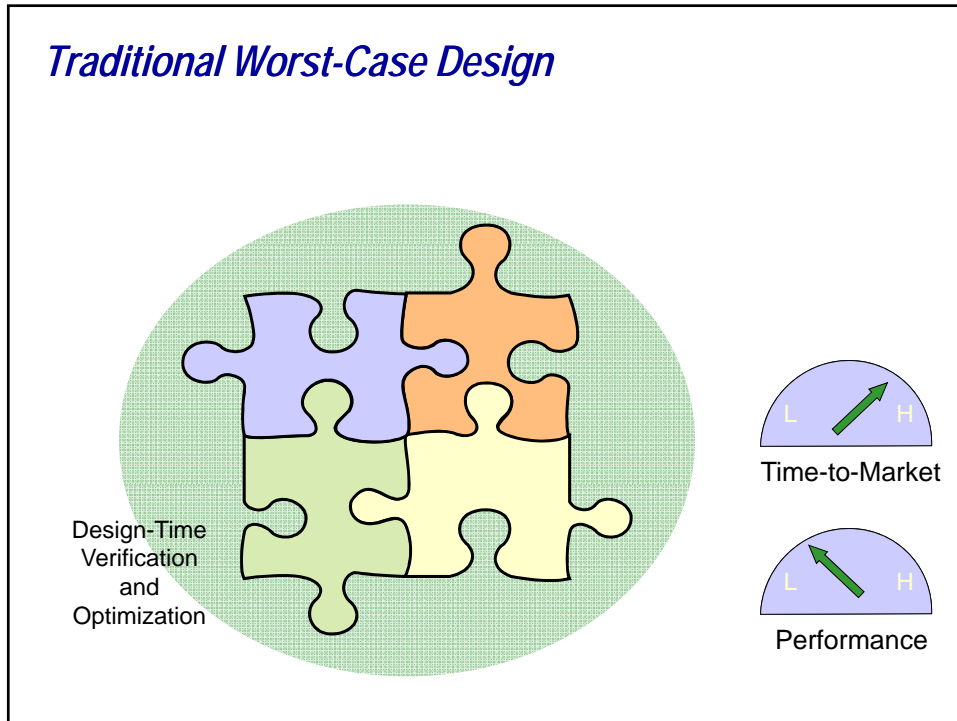
- “... left unchecked, power consumption will reach 1200 Watts for high-end processors in 2018. ... power consumption [is] a major shows topper with off-state current leakage ‘a limiter of integration’.”

Intel chairman Andrew Grove *Int. Electron Devices Meeting* keynote Dec. 2002

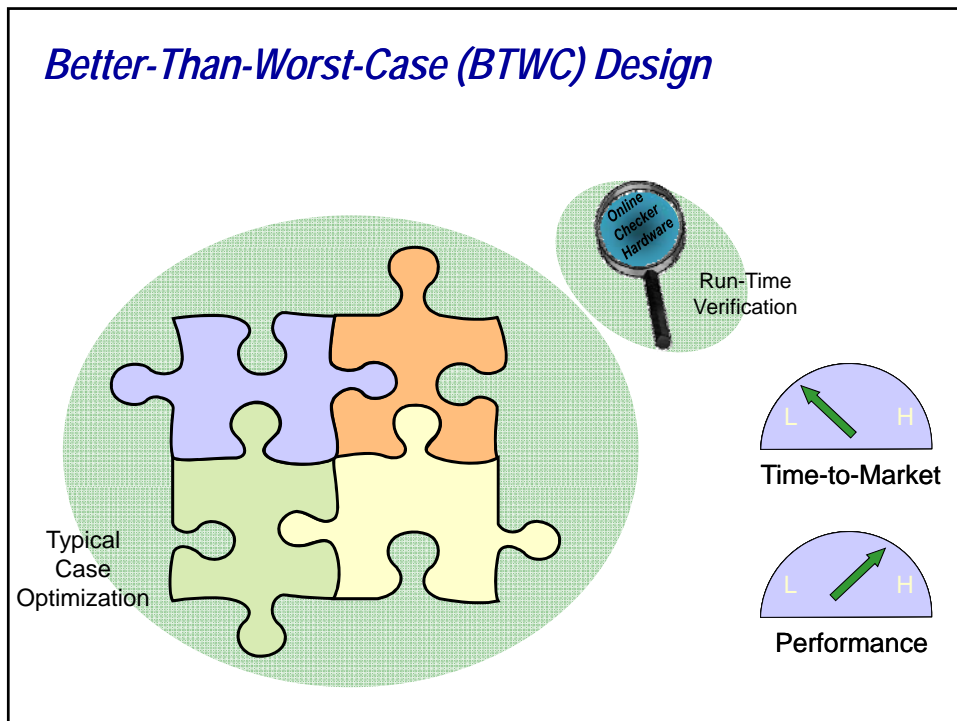
Total Power of CPUs in PCs

- Early '90's – 100M CPUs @ 1.8W = 180MW
- Early 21st – 500M CPUs @ 18W = 10,000MW
- Exponential growth
- Recent comment in a Financial Times article: 10% of US's energy use is for computers
 - exponentially growth implies it will overtake cars/homes/manufacturing
- NOT! – why we're here

Traditional Worst-Case Design



Better-Than-Worst-Case (BTWC) Design



Algorithmic SER-Tolerance [Shanbhag]

◆ **Voltage Overscale Main Block**

◆ **Error Control via Estimator**

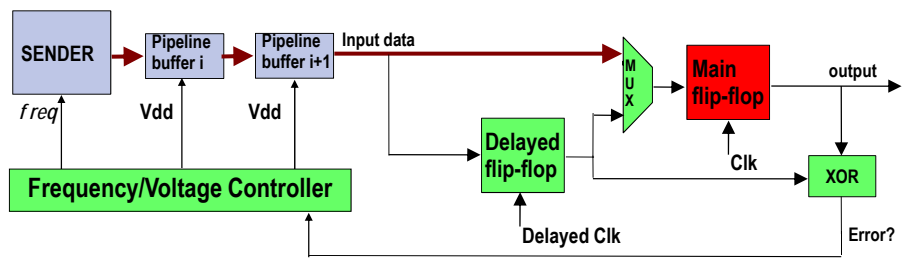
◆ **Estimators: Prediction, Reduced Precision Replica, MAP, Error Canceller and others**

◆ **Employ two estimators in SEU/MEU scenario**

◆ **Robust to error frequencies up to:**

- 1 in 100 samples for SEU
- 1 in 1000 samples for MEU

Timing Error Tolerant Links [De Micheli]

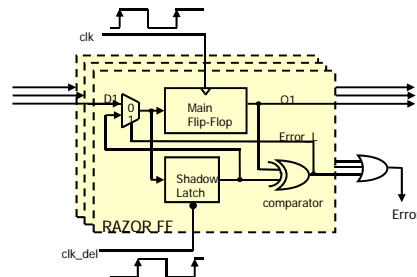


- ◆ **Aggressively clock on-chips links with high frequency/low voltage**
 - Double-sample link output
 - Once speculatively, then again with reliable timing
- ◆ **Stall receiver for recovery data if samples disagree**
 - Non-speculative if receiver incurs additional delay
 - Otherwise, receiver must perform internal recover

Research Topic: Razor Error Resilient Circuits [Austin/Blaauw]

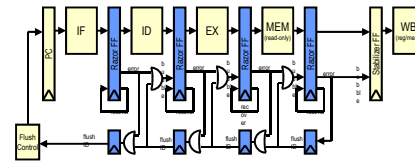
◆ In-situ detection/correction of timing errors

- Tune processor voltage based on errors
- Eliminate process, temperature, and noise margins (tune for near-zero errors)
- Purposely run *below* critical voltage to capture **data-dependent latency margins**

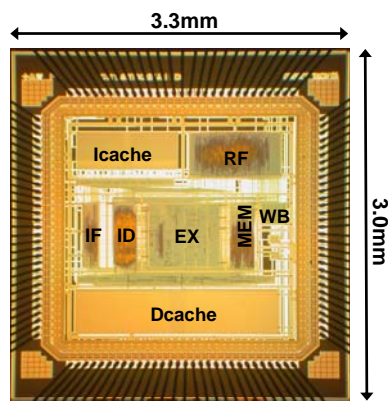


◆ Implemented with architecture and circuit support

- Double-sampling metastability-tolerant Razor flip-flops validate pipeline results
- Pipeline initiates recovery after timing errors, forward progress is guaranteed



Razor Prototype Chip



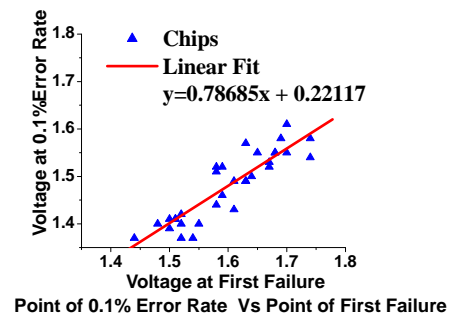
◆ 4 stage 64-bit Alpha pipeline

- 120 - 160MHz operation, 0.18μm

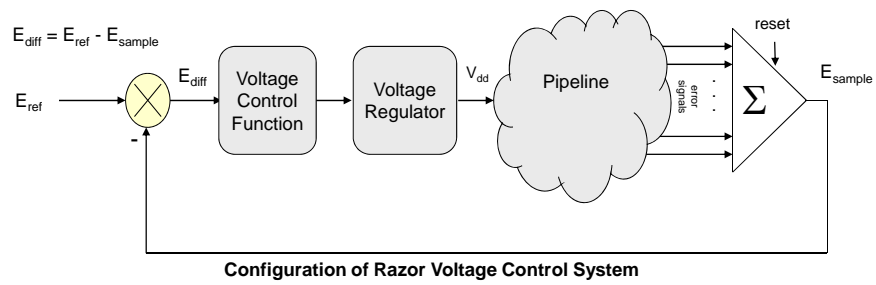
◆ Percentage of FF Razorized: 9%

- Error free Razor overhead ~3%

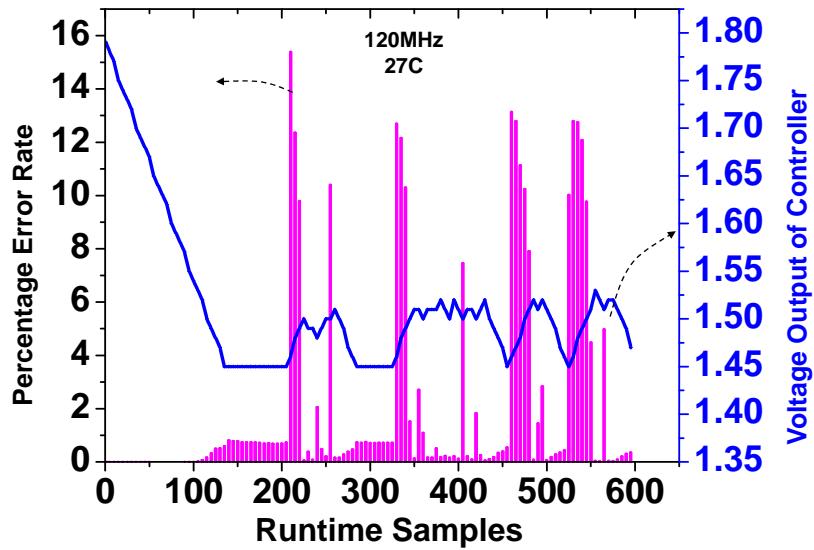
◆ 54% energy reduction



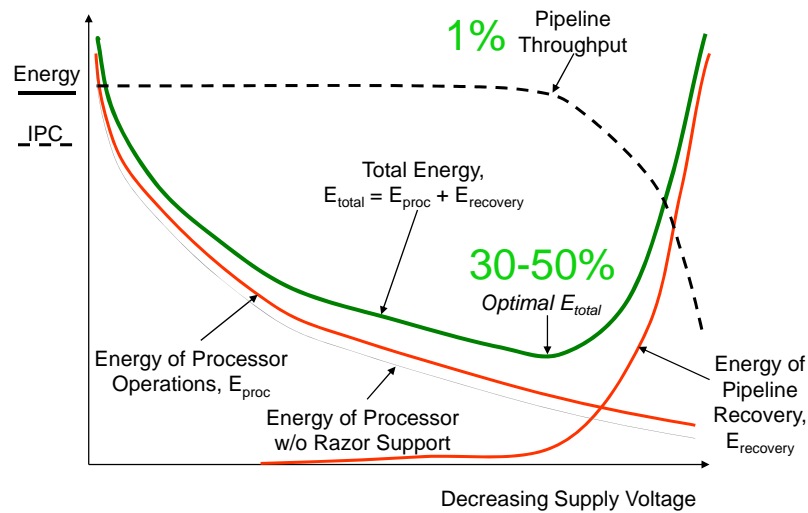
Configuration of the Razor Voltage Controller



Run-Time Response of Razor Voltage Controller



Energy/Performance Characteristics



References

1. C. Constantinescu 'Trend and Challenge in VLSI Circuit Reliability' intel
2. H. T. Nguyen 'A Systematic Approach to Processor SER Estimation and Solutions'
3. P. Shivakumar et. al. 'Modeling the effect of Technology trends on Soft Error Rate of Combinational Logic'
4. P. Shivakumar 'Fault-Tolerant Computing for Radiation Environment' Ph.D. Thesis Stanford University
5. M. Nicolaidis 'Time Redundancy Based Soft-Error Tolerance to Rescue Nanometer Technologies'
6. L. Anghel, et. al. 'Cost Reduction and Evaluation of a Temporary Faults Detecting Technique'
7. L. Anghel, et. al. 'Evaluation of Soft Error Tolerance Technique based on Time and/or Space Redundancy' ICSD
8. I. Koren, University of Massachusetts ECE 655 Lecture Notes 4-5 'Coding'
9. ITRS 2003 Report
10. J. von Neumann, "Probabilistic logic and the synthesis of reliable organisms from unreliable components,"
11. R. E. Lyons, et. al. 'The Use of Triple-Modular Redundancy to Improve Computer Reliability'
12. D. G. Mavis, et. al. 'Soft Error Rate Mitigation Techniques for Modern Microcircuits.' IEEE 40th Annual International Reliability Physics Symposium 2002.
13. C. Weaver, et. al. 'A Fault Tolerant Approach to Microprocessor Design' DSN'01
14. J. Ray, et. al. 'Dual Use of Superscalar Datapath for Transient-Fault Detection and Recovery', Proceedings of the 34th Annual Symposium on Microarchitecture (MICRO'01).
15. J. B. Nickel, et. al. 'REESE: A Method of Soft Error Detection in Microprocessors', Proceedings of the International Conference on Dependable Systems and Networks (DSN'01).
16. S. Reinhardt, et. al. 'Transient Fault Detection Simultaneous Multithreading'

References

1. D. Siewiorek 'Fault Tolerance in Commercial Computers' CMU
2. W. Bartlett, et. al. 'Commercial Fault Tolerance: A Tale of Two Systems' IEEE Dependable and Secure Computing 2004
3. T. Siegel et.al 'IBM's S/390 G5 Microprocessor Design'
4. L. Spainhower, et.al, 'IBM S/390 Parallel Enterprise Server G5 fault tolerance: A historical approach'
5. D. Bossen et.al 'Fault tolerant design of the IBM pSeries 690 system using POWER4 processor technology'
6. 'Tandem HP Himalaya' White Paper
7. Fujitsu SPARC64 V Microprocessor Provides Foundation for PRIMEPOWER Performance and Reliability Leadership
8. D. J. Sorin, et. al. 'SafetyNet: Improving the Availability of SharedMemory Multiprocessors with Global Checkpoint/Recovery.'
9. Milos Prvulovic, et. al. 'ReVive: Cost-Effective Architectural Support for Rollback Recovery in Shared-Memory Multiprocessors'
10. J. Smolens, et.al 'Fingerprinting: Bounding SoftError Detection Latency and Bandwidth'
11. D. Sorin, et.al 'Dynamic Verification of End-to-End Multiprocessor Invariants'