Computer Architecture

Spring 2016

Lecture 22: Power & Reliability in Computer Architecture

Shuai Wang

Department of Computer Science and Technology

Nanjing University

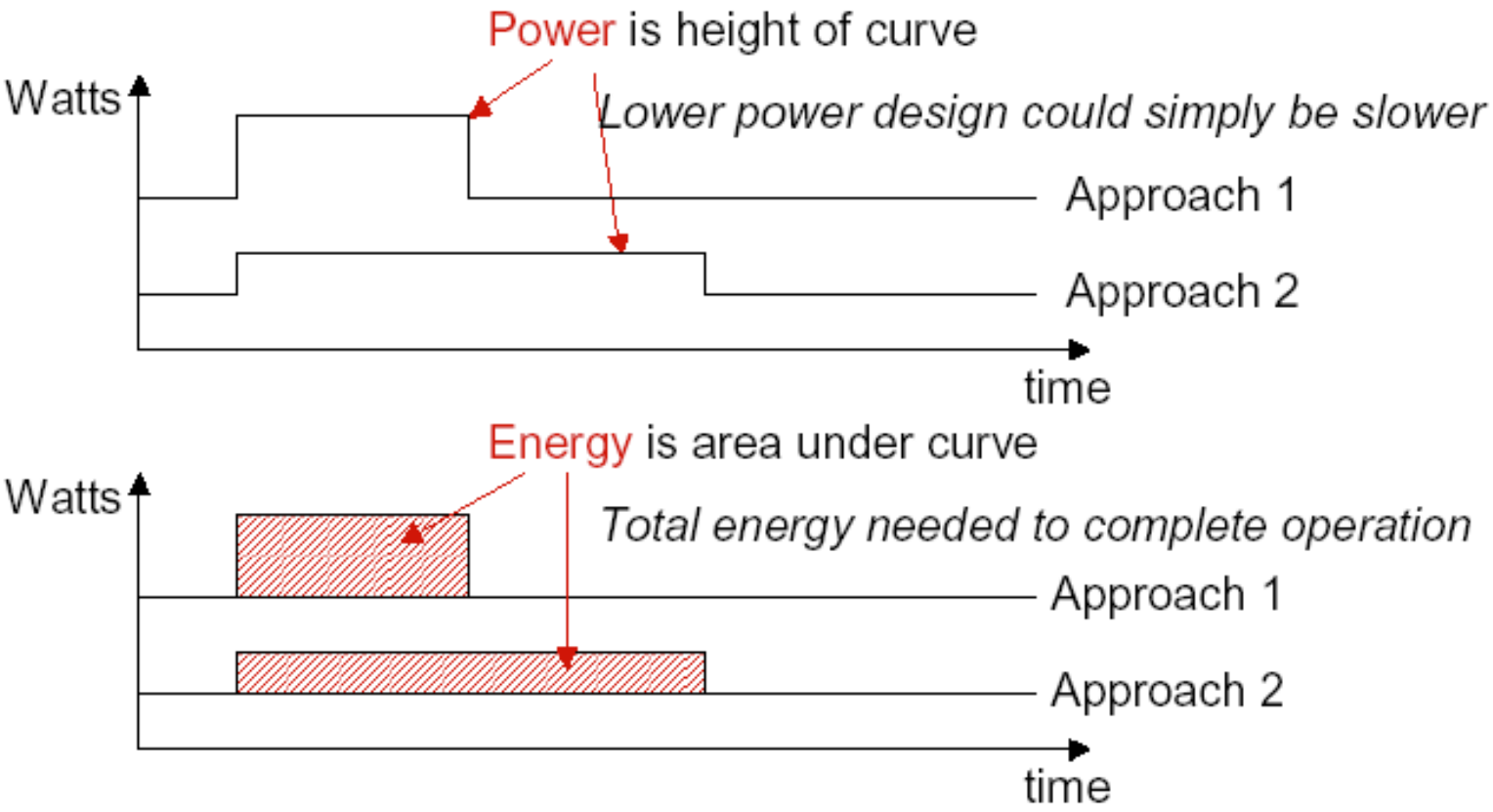
[Slides adapted from CSE 502 Stony Brook University and EECS 6.823 MIT]

Power vs. Energy (1/2)

- *Power*: instantaneous rate of energy transfer
  - Expressed in Watts
  - In Architecture, implies conversion of electricity to heat

- *Energy*: measure of using power for some time
  - Expressed in Joules
  - power * time (joules = watts * seconds)

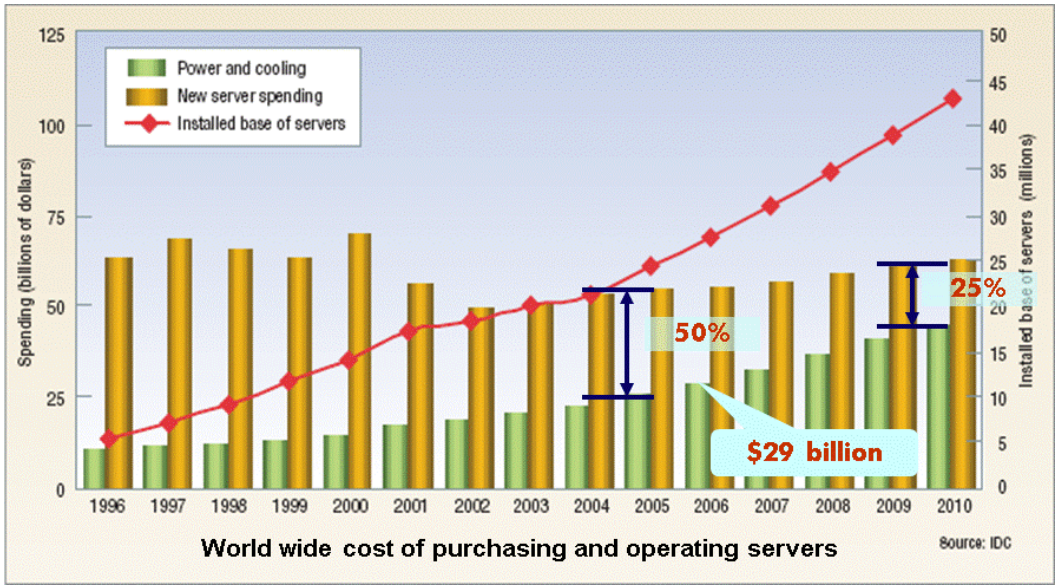
Power vs. Energy (2/2)



- Does this example help or hurt?

Why is energy important?

- Because electricity consumption has costs
  - Impacts battery life for mobile
  - Impacts electricity costs for tethered
    - Delivering power for buildings, countries
    - Gets worse with larger data centers ($7M for 1000 racks)



Why is power important?

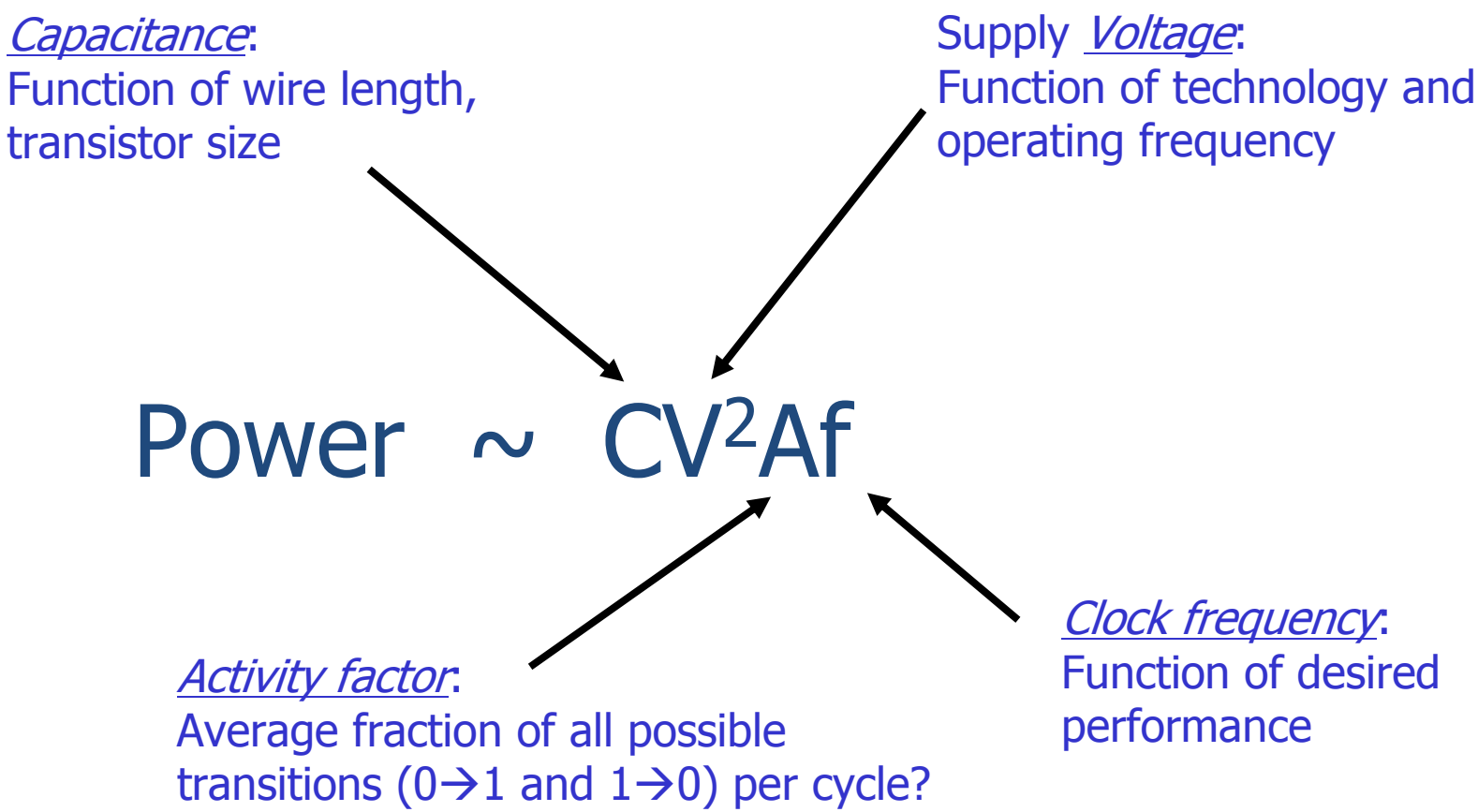
- Because power has a peak
- All power "spent" is converted to heat
  - Must dissipate the heat
  - Need heat sinks and fans
- What if fans not fast enough?
  - Chip powers off (if it's smart enough)
  - Melts otherwise
- Thermal failures even when fans OK
  - 50% server reliability degradation for +10^{o}C
  - 50% decrease in hard disk lifetime for +15^{o}C

Power: The Basics

- Dynamic power vs. Static power
  - Dynamic: “switching” power
  - Static: “leakage” power

- Dynamic power: transitions from 0→1 and 1→0
- Static power: steady, constant energy cost

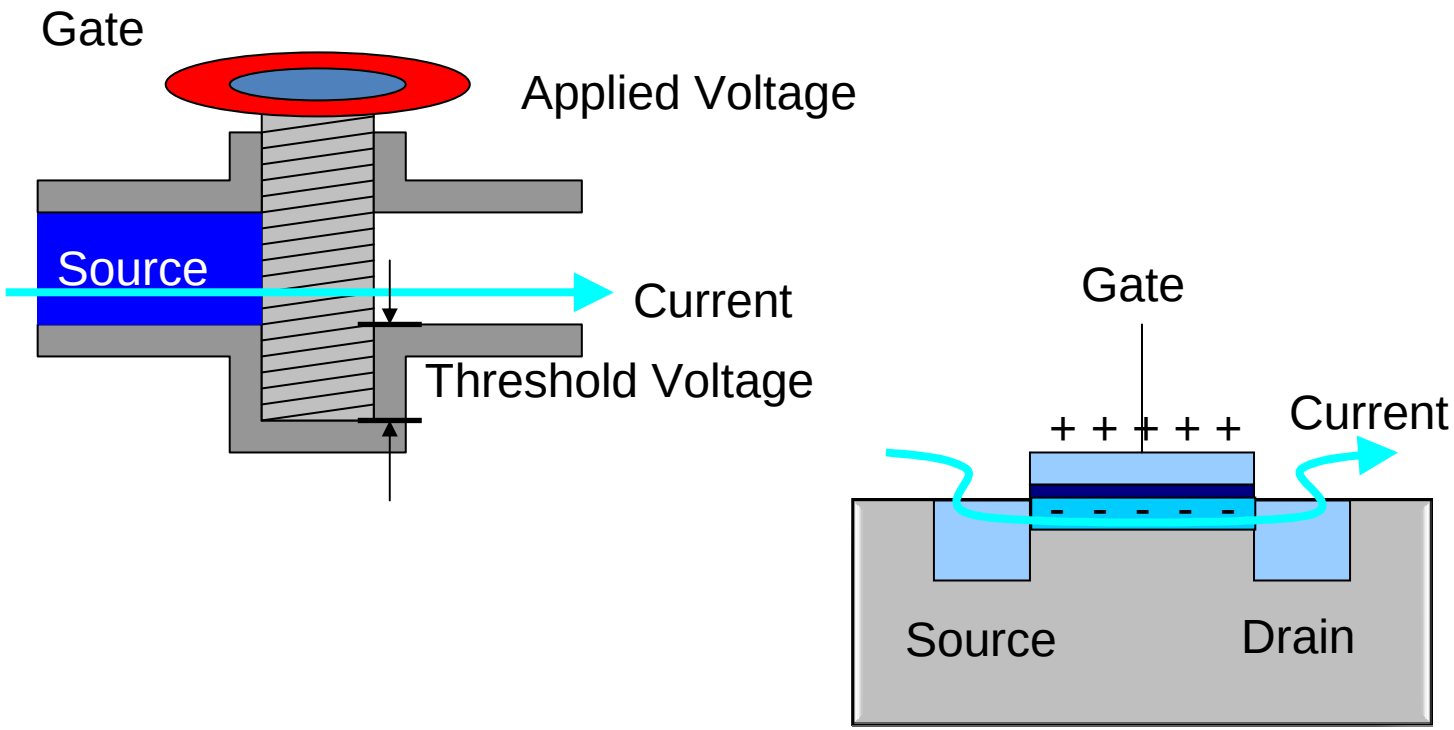
Dynamic Power Dissipation (Capacitive)



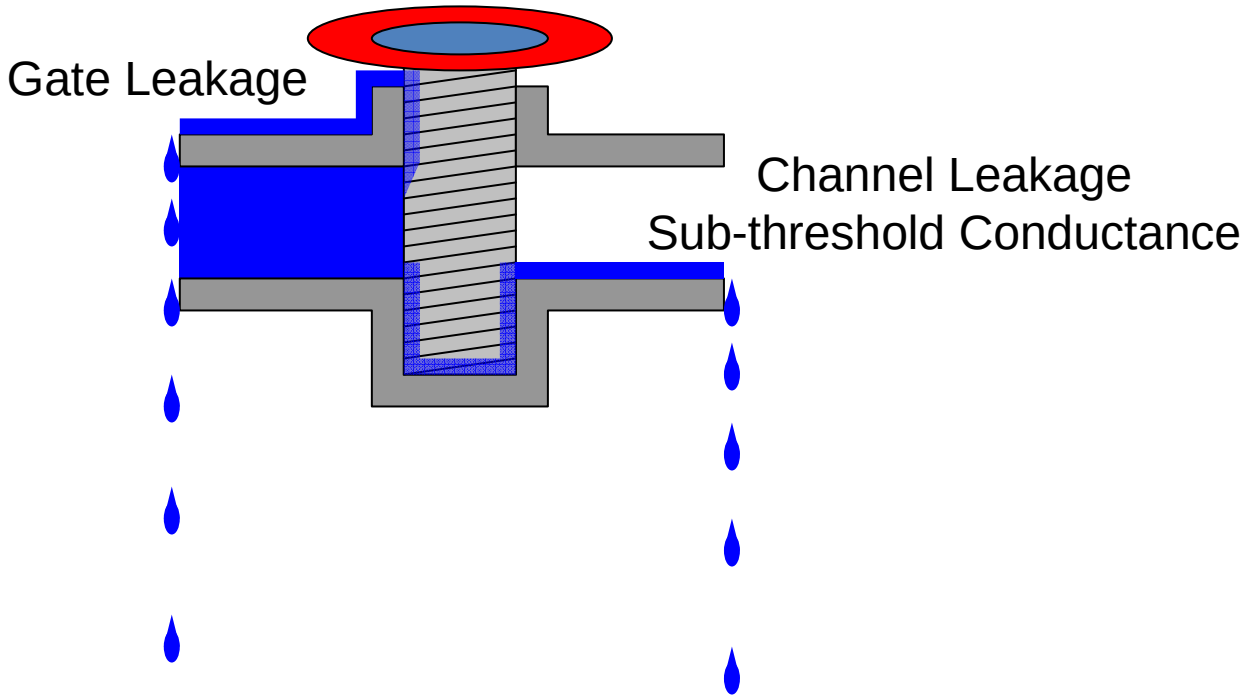
Lowering Dynamic Power

- Reducing Voltage (V) has quadratic effect
  - Has a negative (~linear) effect on frequency
  - Limited by technology (insufficient difference of 1 & 0)
- Lowering Capacitance (C) has linear effect
  - May improve frequency
  - Limited by technology (small transistors, short wires)
- Reducing switching Activity (A) has linear effect
  - A function of signal transition stats
  - Turns off idle units to reduce activity
  - Impacted by logic and architecture decisions

Leakage Power (1/3)

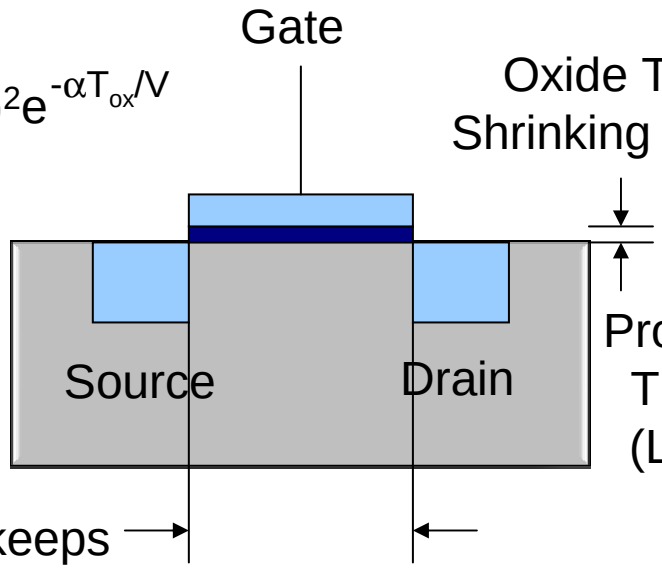


Leakage Power (2/3)



Leakage Power (3/3)

$I_{ox} = K_2 W(V/T_{ox})^2 e^{-\alpha T_{ox}/V}$



Oxide Thickness keeps Shrinking (faster transistors)

Probability of Quantum Tunneling Increases (Leakage increases)

Channel Length keeps Shrinking (faster transistors)

Channel resistance decreases (Leakage increases)

$I_{sub} = K_1 W e^{-V_{th}/nV_\theta}(1-e^{-V/V_\theta})$

Thermal Voltage

Thermal Runaway

- Leakage is a function of temperature

- ↑ Temp leads to ↑ Leakage $I_{sub} = K_1 W e^{-V_{th}/nV_\theta} (1-e^{-V/V_\theta})$
- Which burns more power
- Which leads to ↑ Temp, which leads to…

Metrics

- Energy: in joules
- Power: rate of energy dissipation, in watts

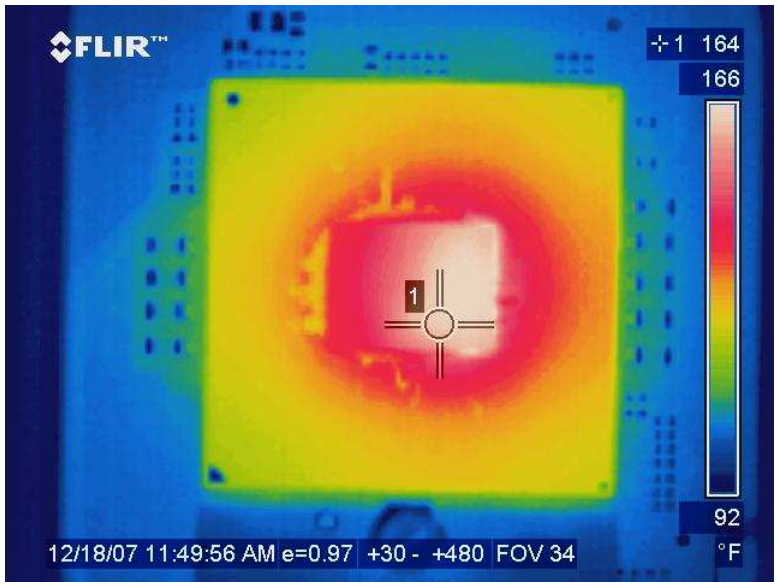
- Energy-Per-Instruction (EPI)
- Energy-Delay Product (EDP):
  - EDP = Energy * Delay
    = $ICount^2$ * 1/($MIPS^2$/Watt)

Measurement

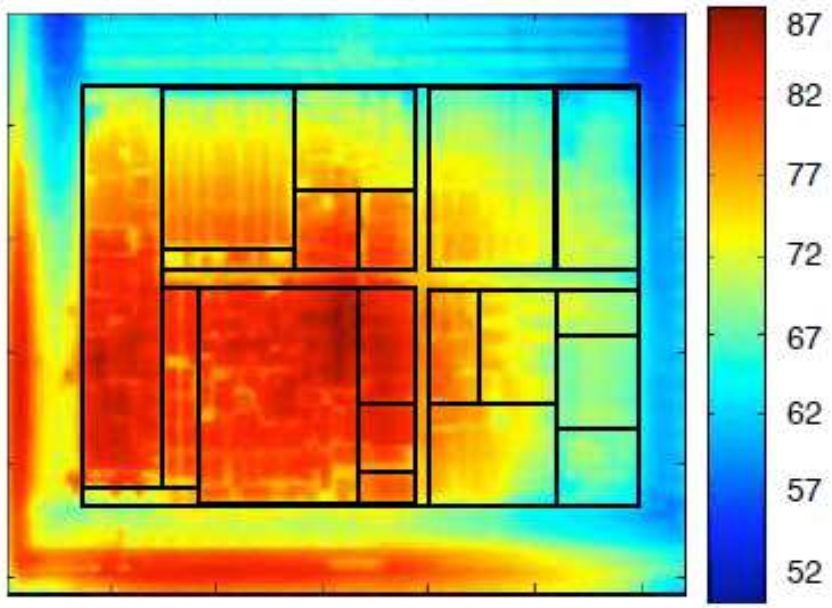
- Challenges:
  - Off-chip ammeters can be used to deduce how much total power the chip dissipates.
  - No straightforward approach to determine a unit-by-unit power breakdown on chip.

- Methods:
  - Performance-Counter-based Power and Thermal Estimates
    - cycle-level simulators to estimate the *activity factors*
    - hardware performance counters
  - Imaging Techniques
    - Power estimates from thermal images: thermal -> power

Imaging Techniques for Power Measurement



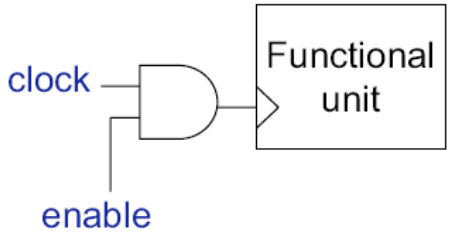
Thermal Image of Processor



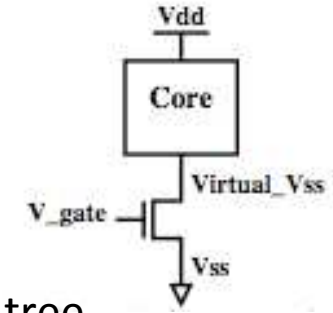
Temperature with overlapped floorplan

Power Management in Processors

- *Clock gating (Dynamic power)*
  - Stop switching in unused components
  - Done automatically in most designs
  - Near instantaneous on/off behavior



- *Power gating (Leakage control)*
  - Turn off power to unused cores/caches
  - High latency for on/off
    - Saving SW state, flushing dirty cache lines, turning off clock tree
    - Carefully done to avoid voltage spikes or memory bottlenecks
  - Issue: Area & power consumption of power gate
  - Opportunity: use thermal headroom for other cores



DVFS: Dynamic Voltage/Frequency Scaling

- Scaling down supply voltage
  - reduces dynamic power (quadratic)
  - reduces leakage power (linear)

- Reducing supply voltage often slows transistors such that reducing the clock frequency is also required.

- Design Issues of DVFS:
  - At what level?
    - System? Program? or Hardware level?
  - How to choose DVFS setting?
  - What hardware granularity?

System-Level DVFS

- Usually at operating system level
- Eliminating Idle Time
  - Do DVFS rather than shut-down
- Discovering and Exploiting Deadlines
  - Especially for real-time systems

Program-Level DVFS

- Offline Compiler Analysis
  - Profile-assisted compiler approach
  - Analytic techniques (heuristic)
- Online Dynamic Compiler analysis
  - run-time analytic model
    - Hot code detection (loop)
    - Memory bound decision
- Coarse-Grained Analysis Based on Power Phases
  - Counter based
  - Make Prediction
- DVFS for Multi-Core Processors

Hardware-Level DVFS

- Razor – Hardware-level DVS
  - Indentify critical flip-flops
  - Add Shadow Latch (driven by time-delayed clock)
  - Dynamic Voltage
  - Time error detection and correction.

Dynamic Power Control Techniques

- Idle-Width (Data) Switching Activity Control

- Idle-Capacity Switching Activity Control

Idle-Width Switching Activity

- Narrow-Width Operands
  - Dynamically detecting narrow-width operands
  - Value gating: disabling the unused width
- Caches:
  - Dynamic Zero Compression
  - Value Compression and the Frequent Value Cache
  - Compressed Cache Lines
  - Instruction Compression

Idle-Capacity Switching Activity

- Power-inefficiency of Out-of-order Processors
- Resource Partitioning
- Instruction Queue:
  - Physical Resizing
  - Readiness Feedback Control
  - Occupancy Feedback Control
  - Logical Resizing Without Partitioning

Idle-Capacity Switching Activity -cont'd

- Core:
  - Adaptive Issue width: 8-issue -> 4-, 6-issue
    - Disable half cluster
- Caches:
  - Physical Memory Segment
  - Selective Cache Ways
  - Phased Cache (tag -> data)
  - Sequentially Accessed Set-Associative Cache (sequential based on MRU)
  - Way Prediction

Idle-Capacity Switching Activity -cont'd

- More Low Power Caches:
  - Filter Cache
  - Loop Cache
  - Trace Cache

- Pipeline gating
  - Gate and stall the whole pipeline when confidence in branch prediction is low

Leakage Power Management

- Leakage Power
  - Subthreshold Leakage
    - W/L: transistor size
    - V_{ds} : voltage differential between the drain and the source
    - V_T : threshold voltage (the smaller the V_T, the higher is the leakage)
    - T: temperature
  - Gate Leakage
    - gate oxide thickness
    - V_{gs} and V_{gd}

Leakage Power Control

- Dynamically Resized (DRI) Cache
- Cache Decay
  - Predict dead lifetime of cache
- Drowsy Caches
  - Instead of cutting off the supply voltage, using a low $V_{dd.}$ (DVS)
- Temperature-driven adaptation

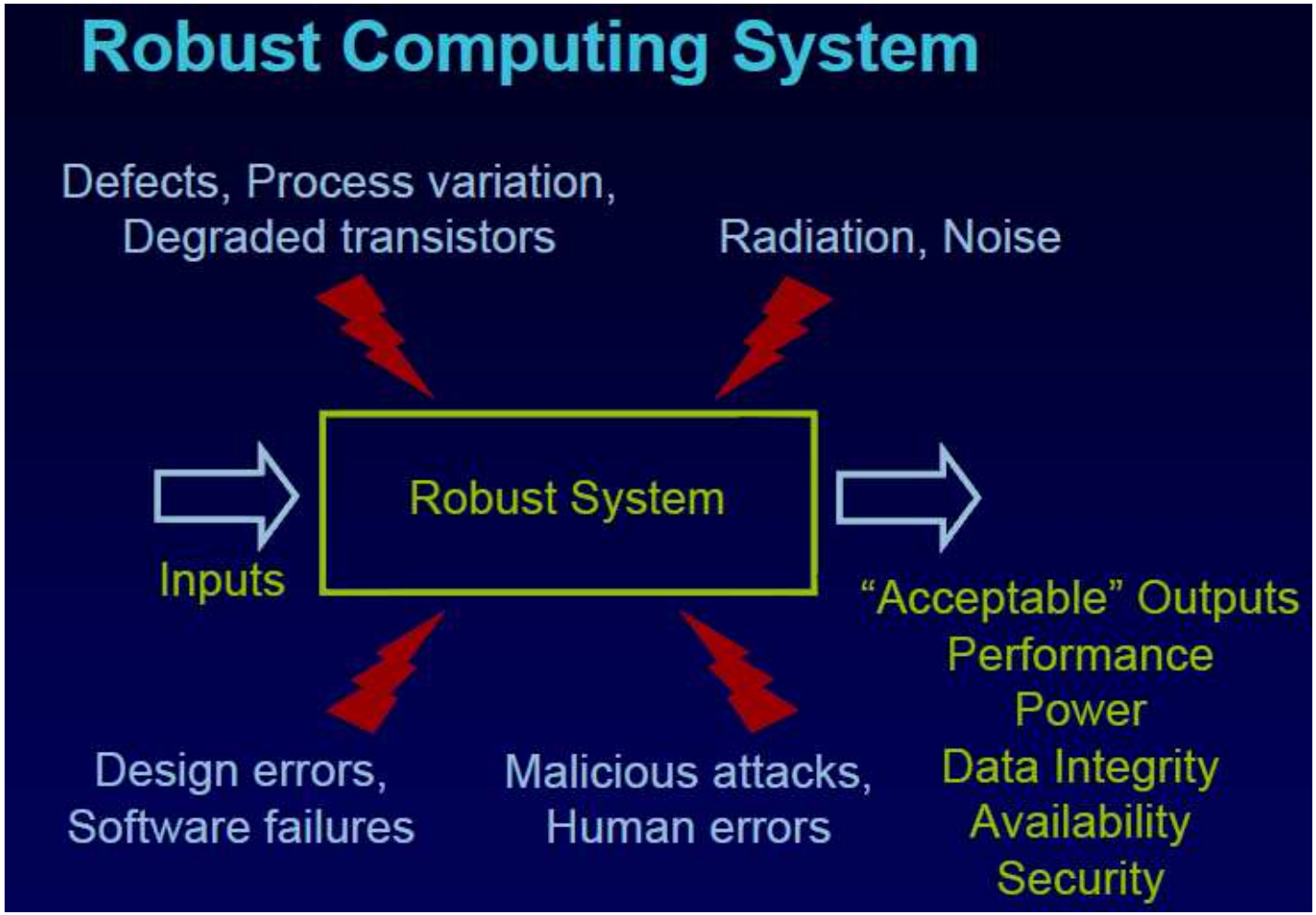
Software Level Power Reduction

- Algorithm (Software) level power reduction techniques focus on minimizing the number of operations weighted by the cost of those operations.
  - First-Order Goal: Reducing the number of operations to be performed.
  - Algorithm selected based on the underlying hardware implementation.
    - E.g., addition versus a logical operation, cost of a memory access, etc.

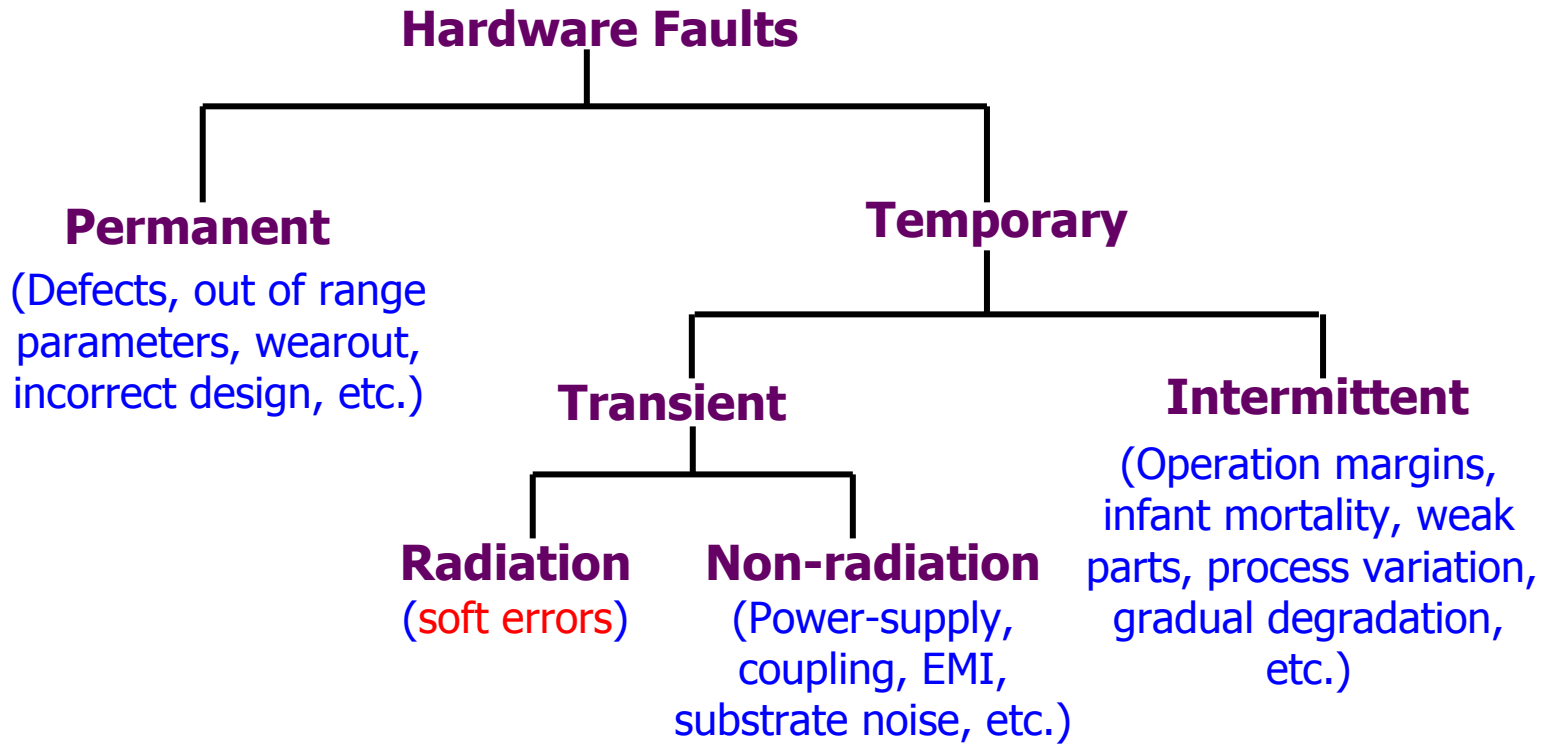
- Compiler Optimization
  - Strength reduction, common subexpression elimination, minimize memory traffic, loop unrolling, etc.

- Data representation
  - Data type: fixed-point vs. floating-point
  - Bit width reduction

Reliable System



Hardware Faults & Soft Errors



- Permanent faults can be avoided by thorough validation, testing, and early life failure screening.
- Soft errors cannot be captured by traditional verification and testing process due to the irrelevancy to the correctness of the logic.

Metrics

- Interval-based
  - MTTF = Mean Time to Failure
  - MTTR = Mean Time to Repair
  - MTBF = Mean Time Between Failures = MTTF + MTTR
  - Availability = MTTF / MTBF

- Rate-based
  - FIT = Failure in Time = 1 failure in a billion (10^9) hours
  - 1 year MTTF = 10^9 / (24 * 365) FIT = 114,155 FIT
  - $FIT_{system} = \sum FIT_{components}$

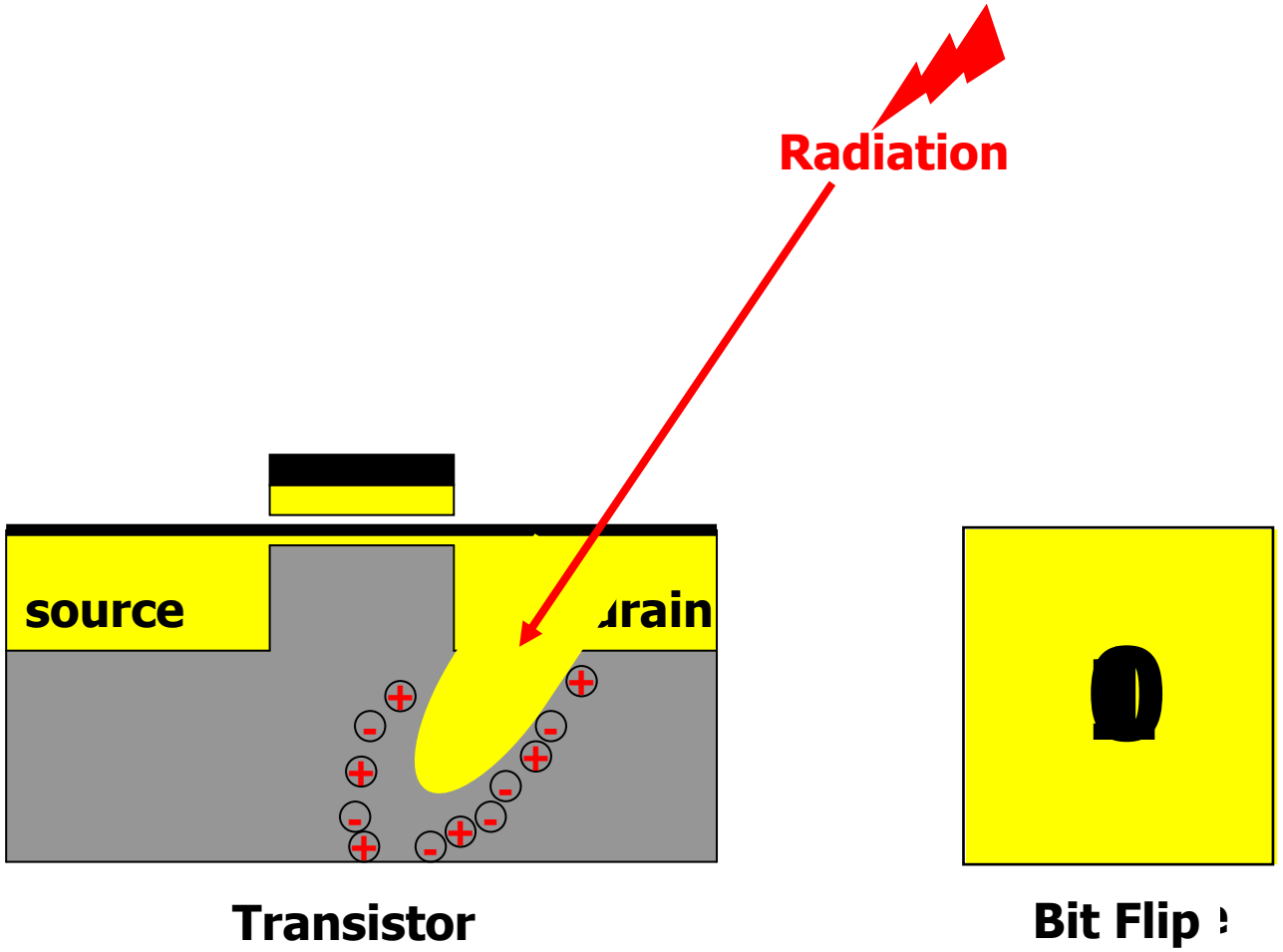
Soft Errors

- Soft errors are that the internal states of nodes are flipped due to excess charge carriers induced primarily by external radiations.

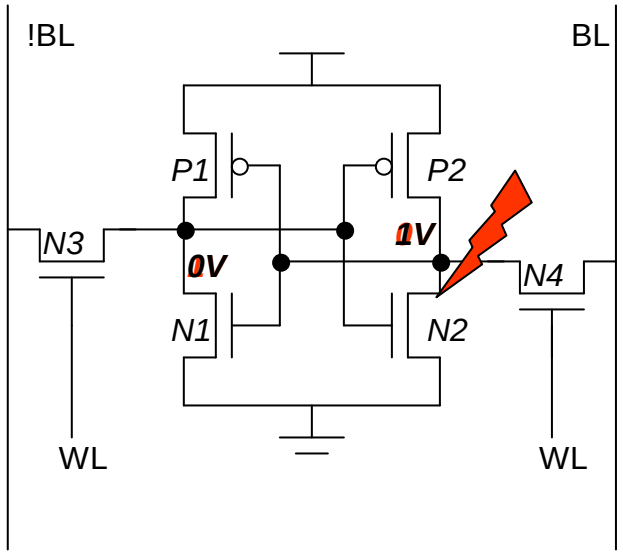
- These errors cause an upset event but the circuit itself is not damaged.

- Soft Errors can cause problems in different ways
  - Change the data value in the Caches and Memory
  - Corrupt the execution of instruction due to the flip of data in the pipeline registers
  - Change the character of a SRAM-based FPGA circuit. (Firm Error)
  - Datapath logic SET (Single Event Transient) caught by registers/memory

A Soft Error Scenario



A Soft Error Scenario



- An example of an alpha particle that hits the transistor in a SRAM cell and flips the stored value 1 into 0, resulting soft errors.

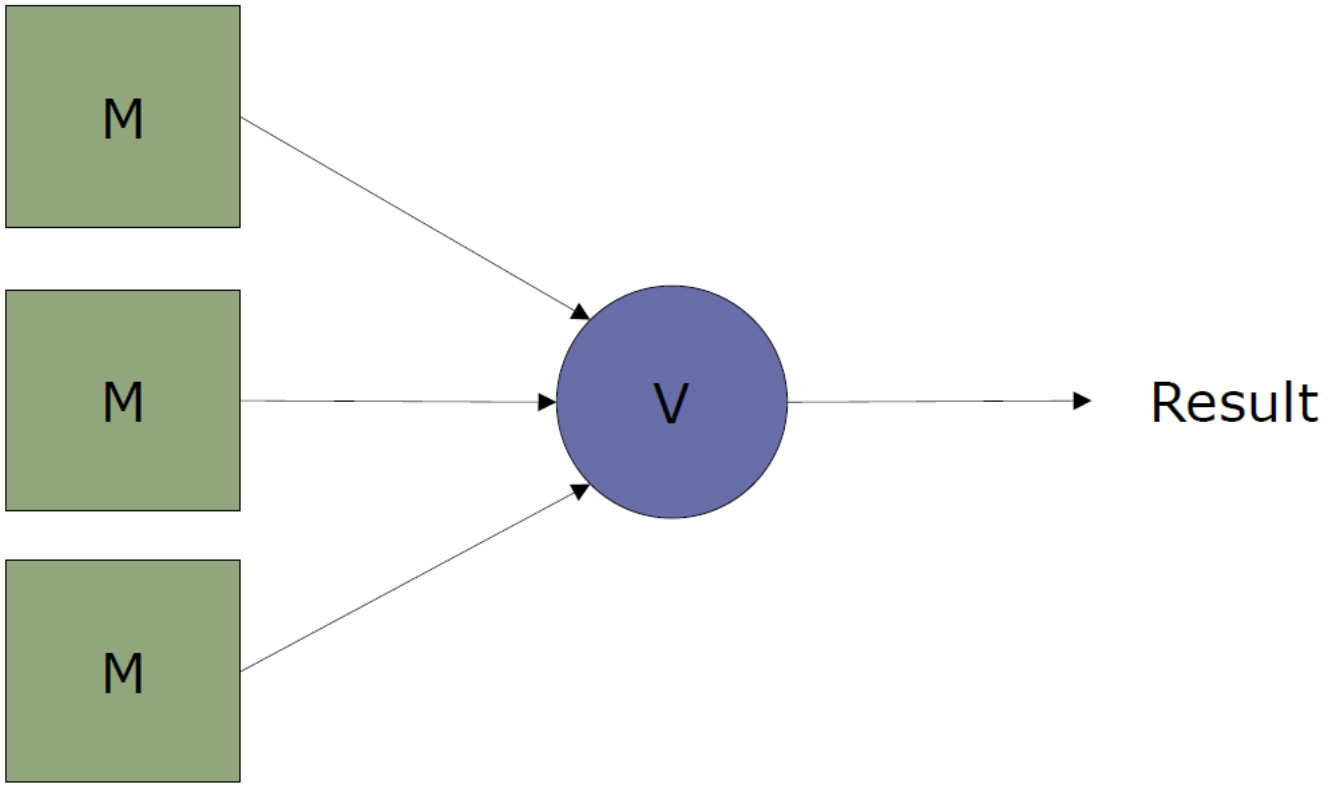
Soft Errors: An In Depth Focus

- Cosmic Particles (becoming a largest contributor, mainly neutrons)
  - The upset rate in usually shown at 300 meters above sea level, increases with altitude by a factor of 2.36 every 1K meters.
- Package Component
  - Alpha particles emitted by decaying radioactive impurities in packing and interconnect materials. (Plastic packages is the worst)
- Solder-Bump Component
  - Alpha particles emitted by isotopes of the lead and other elements used to attach die and package
- Boron 10 Component
  - Starting from 2002, IBM began using new materials with negligible Boron 10. Consequently, it is not a concern for IBM anymore.

Physical Solutions Are Hard

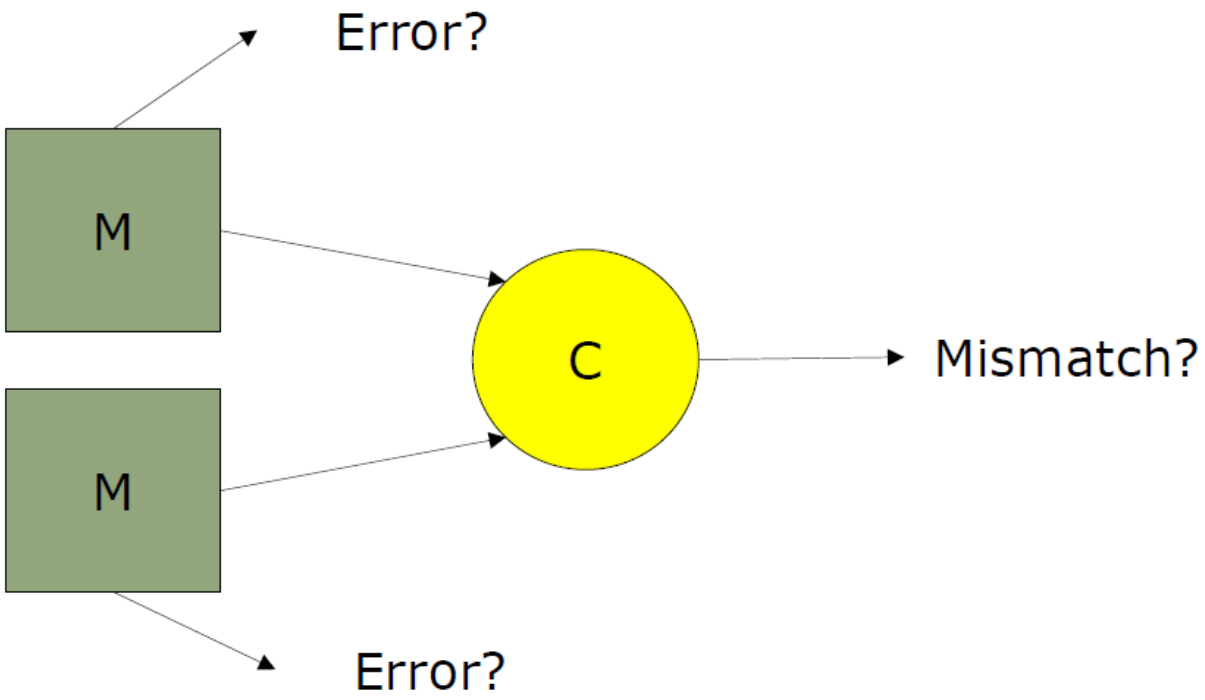
- Shielding?
  - No practical absorbent (e.g., approximately > 10 ft of concrete)
  - This is unlike Alpha particles which are easily blocked
- Technology solution: SOI?
  - Partially-depleted SOI of some help, effect on logic unclear
  - Fully-depleted SOI may help, but is challenging to manufacture
- Circuit level solution?
  - Radiation hardened circuits can provide 10x improvement with significant penalty in performance, area, cost
  - 2-4x improvement may be possible with less penalty

Triple Modular Redundancy (Von Neumann, 1956)



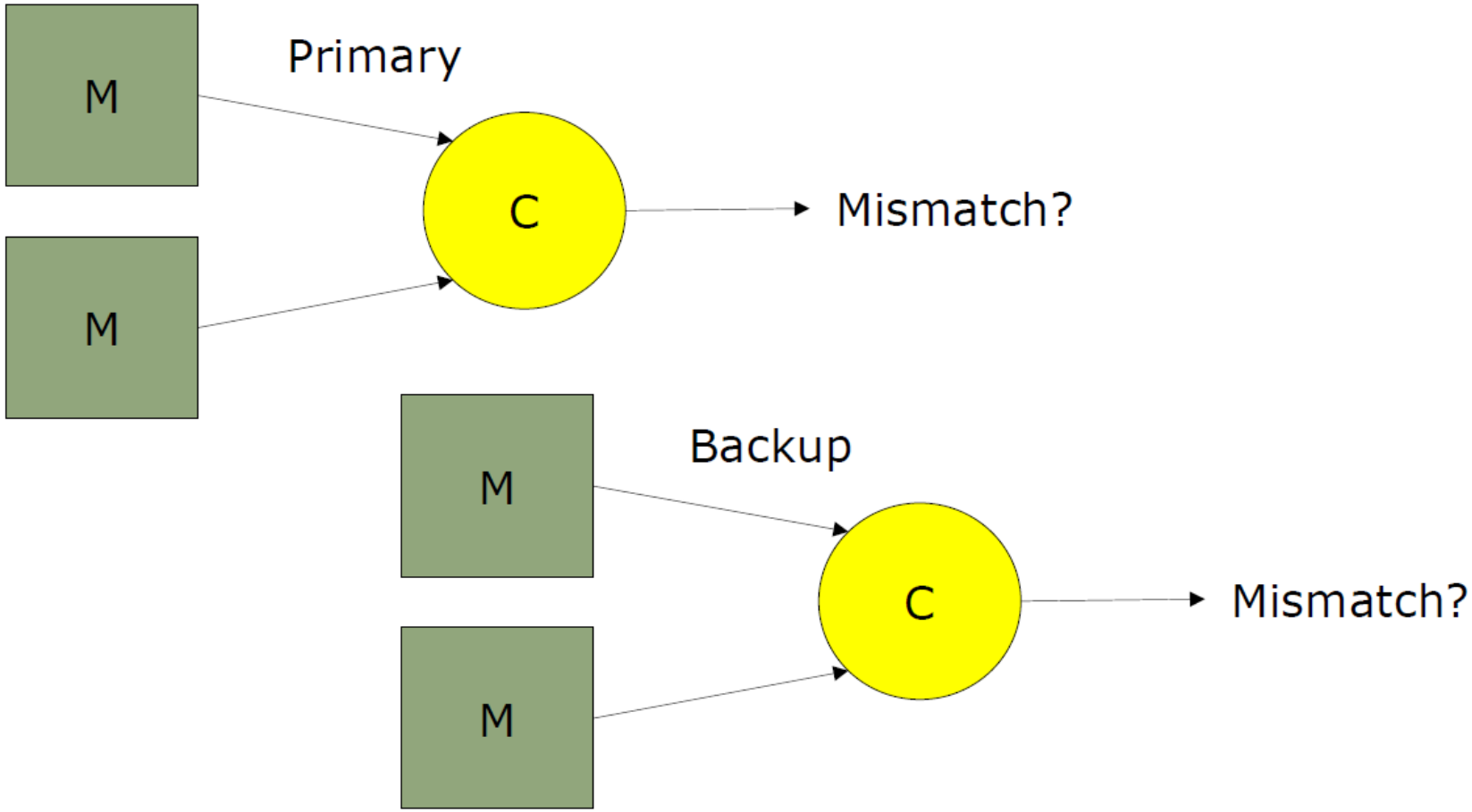
V does a majority vote on the results

Dual Modular Redundancy (eg., Binac, Stratus)



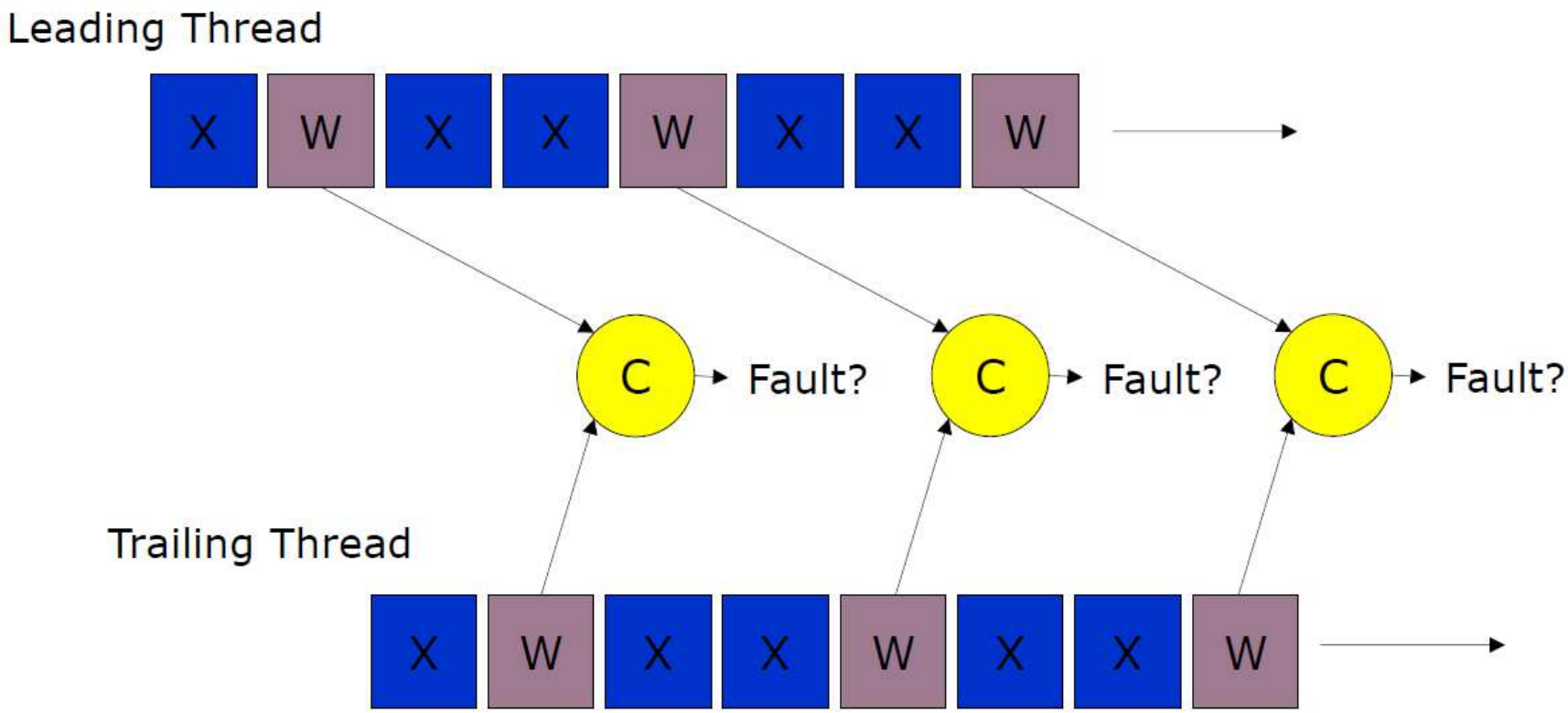
- Processing stops on mismatch
- Error signal used to decide which processor be used to restore state to other

Pair and Spare Lockstep (e.g., Tandem, 1975)



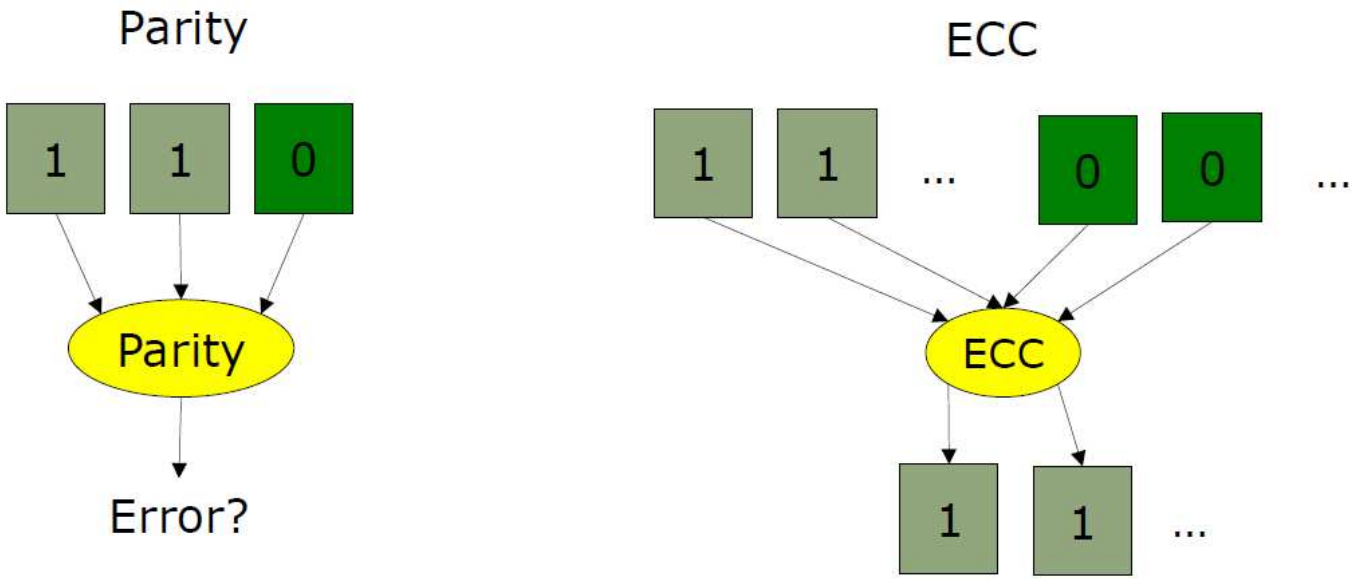
- Primary creates periodic checkpoints
- Backup restarts from checkpoint on mismatch

Redundant Multithreading (e.g., Reinhardt, Mukherjee, 2000)



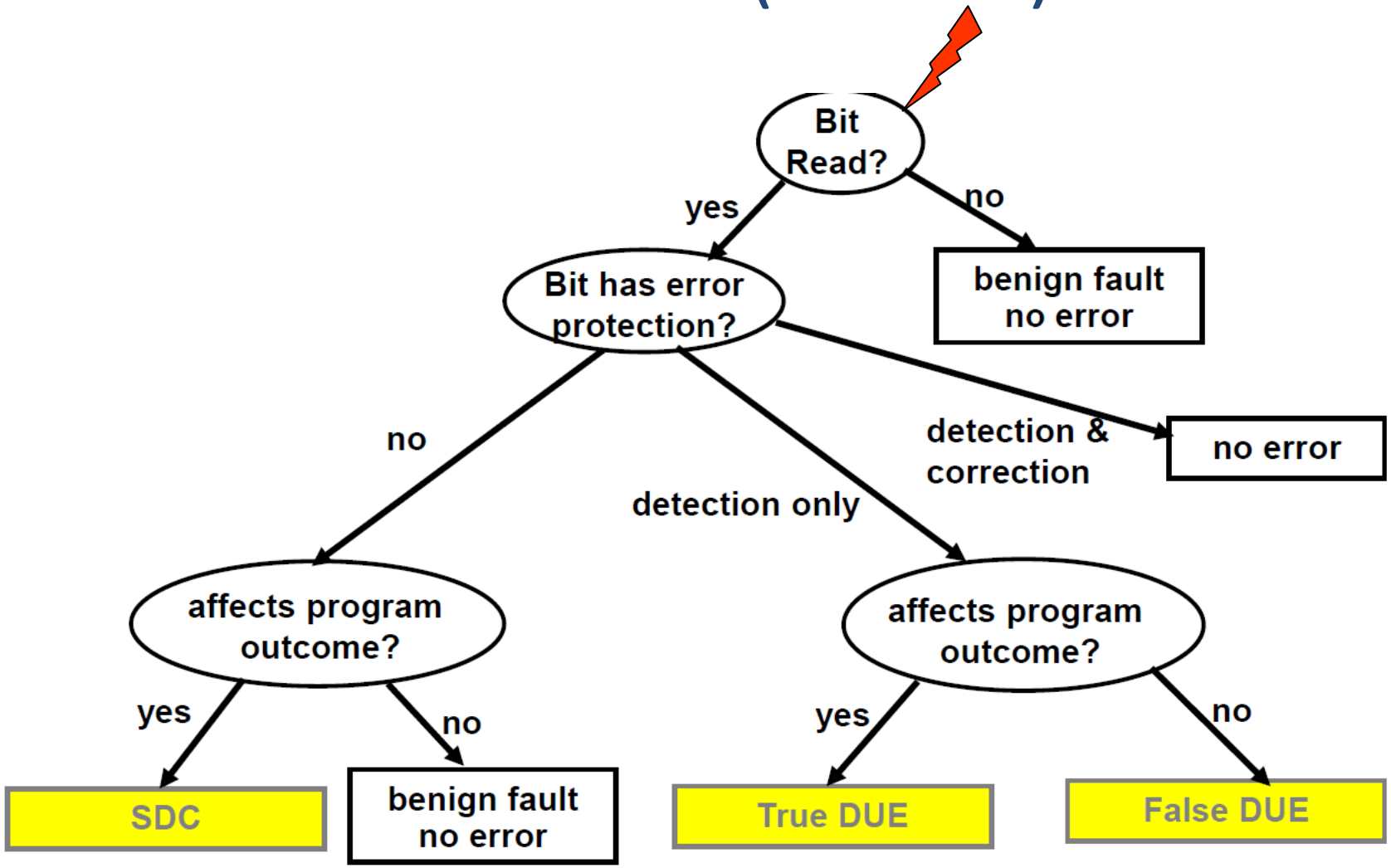
- Writes are checked

Component Protection (Coding)



- Fujitsu SPARC in 130 nm technology (ISSCC 2003)
  - 80% of 200k latches protected with parity
  - versus very few latches protected in commodity microprocessors

Soft Error (one bit)



SDC = Silent Data Corruption, DUE = Detected Unrecoverable Error

Architectural Vulnerability Factor (AVF)

- AVF_{bit} = Probability Bit Matters

= # of Visible Errors / # of Bit Flips from Particle Strikes

- $FIT_{effective\_bit} = FIT_{raw\_bit} * AVF_{bit}$

- AVF = fraction of cycles a bit contains ACE state

= Average number of ACE bits in a cycle / Total number of bits in the structure

  - ACE: Architecturally Correct Execution

Architectural Vulnerability Factor

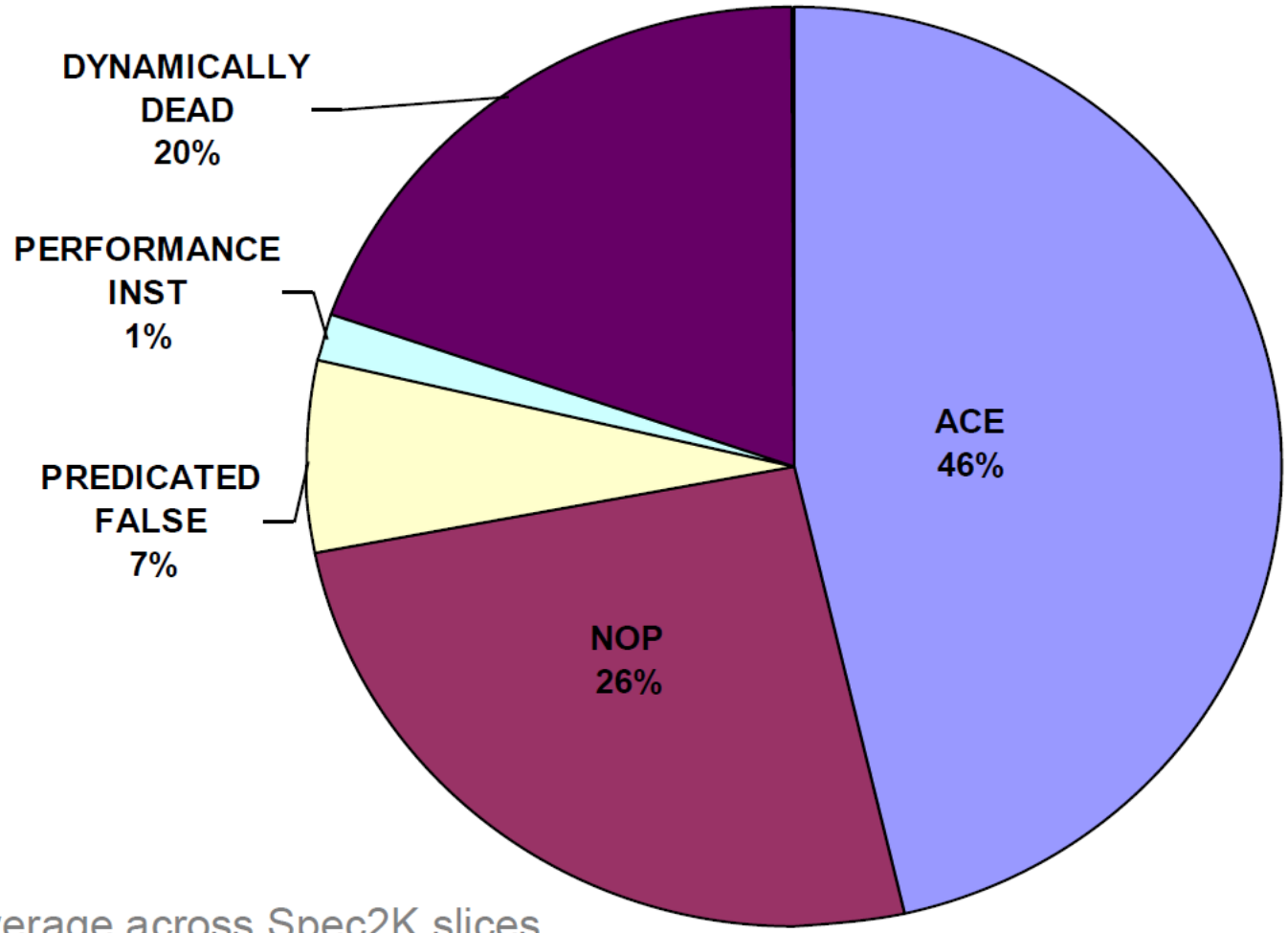
- Branch Predictor
  - Doesn’t matter at all (AVF = 0%)

- Program Counter
  - Almost always matters (AVF ~ 100%)

Computing AVF

- Approach is conservative
  - Assume every bit is ACE unless proven otherwise
- Data Analysis using a Performance Model
  - Prove that data held in a structure is un-ACE
- Timing Analysis using a Performance Model
  - Tracks the time this data spent in the structure

Dynamic Instruction Breakdown



Average across Spec2K slices