Energy-Efficient Manycore Architectures for Big Data

Josep Torrellas

Department of Computer Science
University of Illinois at Urbana-Champaign
http://iacoma.cs.uiuc.edu

BPOE

April 2015

Wanted: Energy-Efficient Computing

- State of the Art:

Performance: 11 PF
Power: 6-11 MW (idle to loaded)
10MW = $10M per year electricity

University of Illinois Blue Waters Supercomputer

- Extreme Scale computing: 100x more capable for the same power consumption and physical footprint

Recap: How Did We Get Here?

- Ideal Scaling (or Dennard Scaling): Every semicond. generation:
  - Dimension: 0.7
  - Area of transistor: 0.7x0.7 = 0.49
  - Supply Voltage V_{dd}, C: 0.7
  - Frequency: 1/0.7 = 1.4

$$P_{dyn} \propto CV_{dd}^2 f$$

Constant dynamic power density

- Real Scaling: V_{dd} does not decrease much
  - If too close to threshold voltage (V_{th}) → slow transistor
  - Dynamic power density increases with smaller tech
  - Additionally: There is the static power

Power density increases rapidly

Low Voltage Operation

- V_{dd} reduction is the best lever for energy efficiency:
  - Big reduction in dynamic power; also reduction in static power

- Advantages:
  - Reduces energy per operation quickly
- Drawbacks:
  - Lower speed
  - Higher variation in gate delay and power consumption

Let's Start with the Cores

- Application characteristics vary dynamically
- Goal: Single core design that attains
  - High performance at nominal voltage (~0.9V)
  - High energy efficiency at low voltage (~0.5V)

A Voltage-Scalable Core

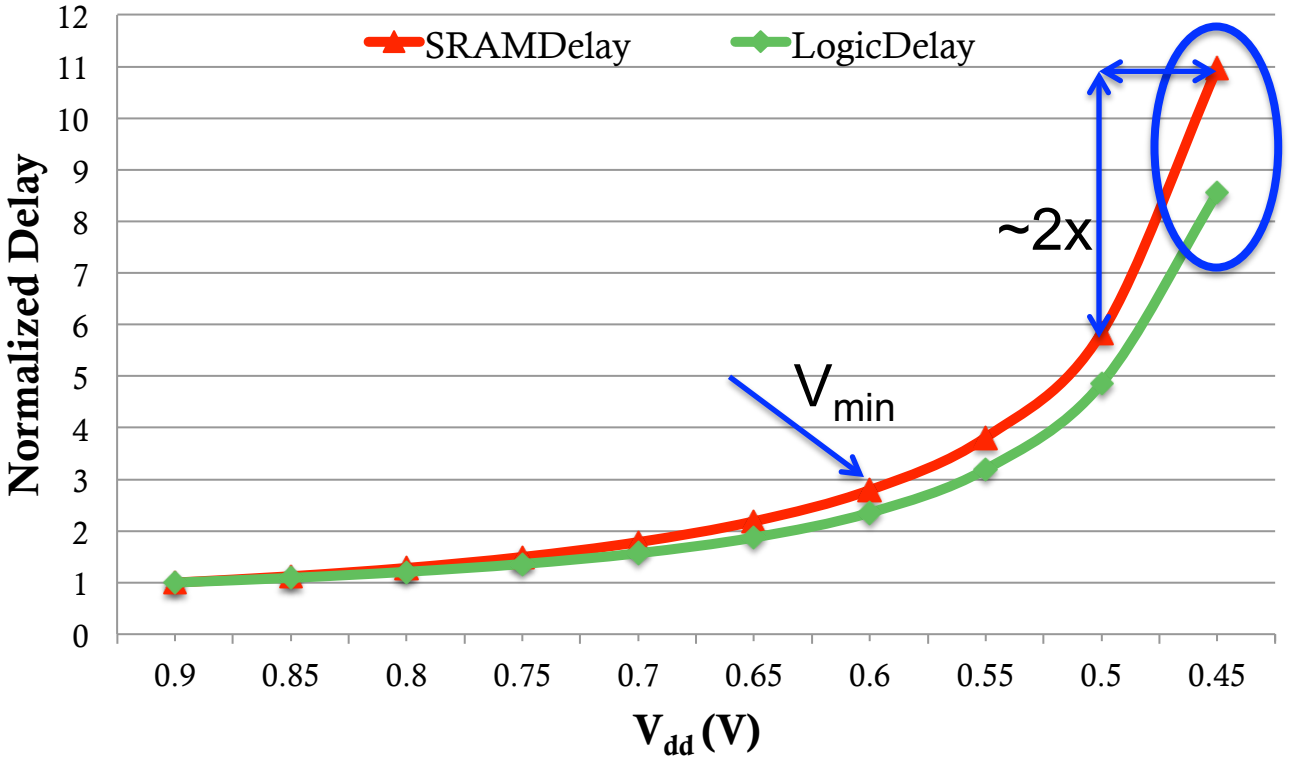
Why not Big/Little ?

- Heterogeneous cores on a chip ideally suited for different tasks 

- Fixed partitioning of cores 
- A fraction of chip unused 
- Migration overhead 

Observations

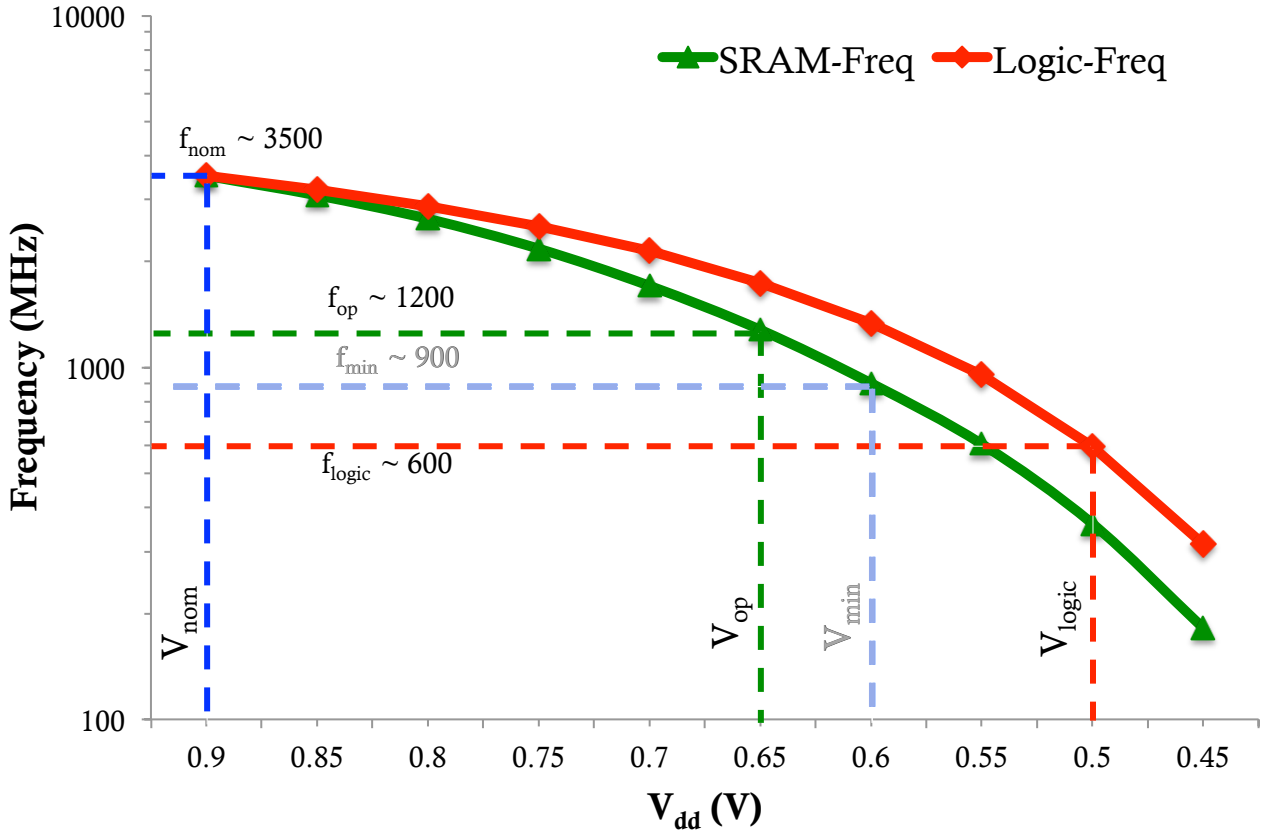
- SRAM vs Logic delay scaling
- Small increase in V → large improvement in delay



ScalCore Idea

[Gopireddy HPCA'16]

- Design a Voltage-Scalable core based on the two observations



ScalCore: A Voltage-Scalable Core

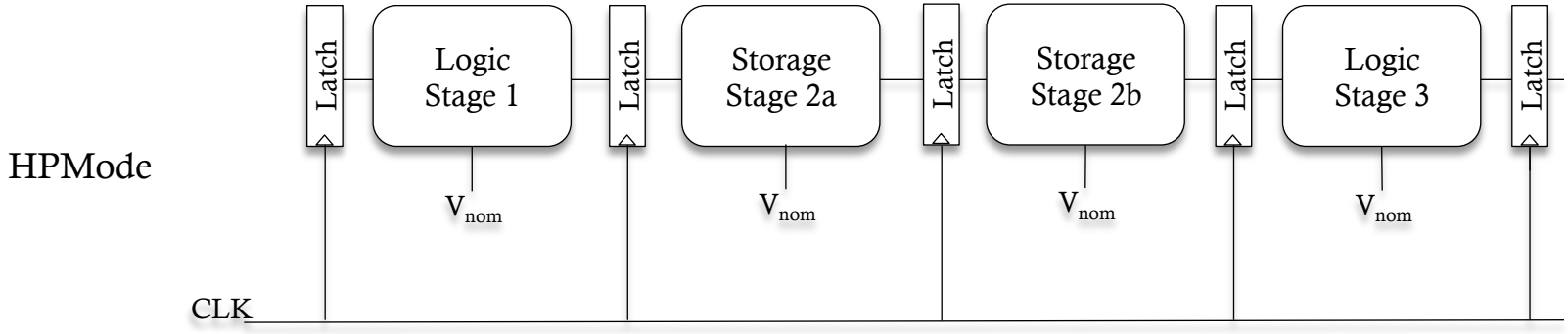
- Decouple V_{dd} of logic and storage structures in the pipeline
  - Improve energy efficiency
- Raise V_{dd} of storage structures a little
  - Reconfigure the pipeline to take advantage of faster storage structures
  - Further improve performance and reduce leakage energy !!

A Voltage-Scalable Core

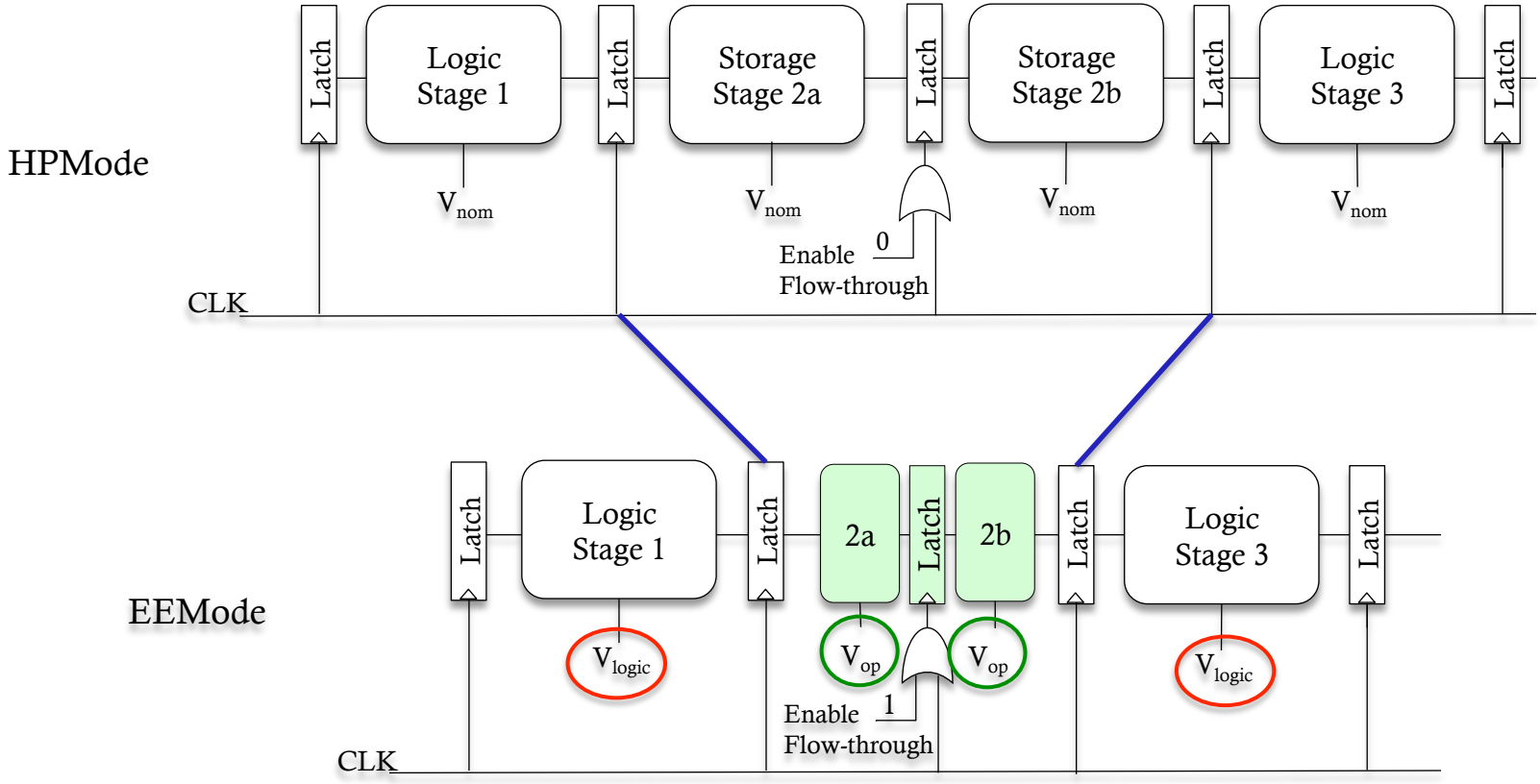
How Does it Work?

- At nominal, high-performance conditions (HPMode):
  - No impact on performance or energy
- When energy efficiency matters (EEMode):
  - V_{dd}s for storage and logic stages in the pipeline decouple
    - Storage stage ~2x faster than logic stage
  - Pipeline is reconfigured in one of the two ways
    - Fuse storage stages in the pipeline (e.g., access register file)
    - Increase storage structure sizes (e.g., load-store queue)

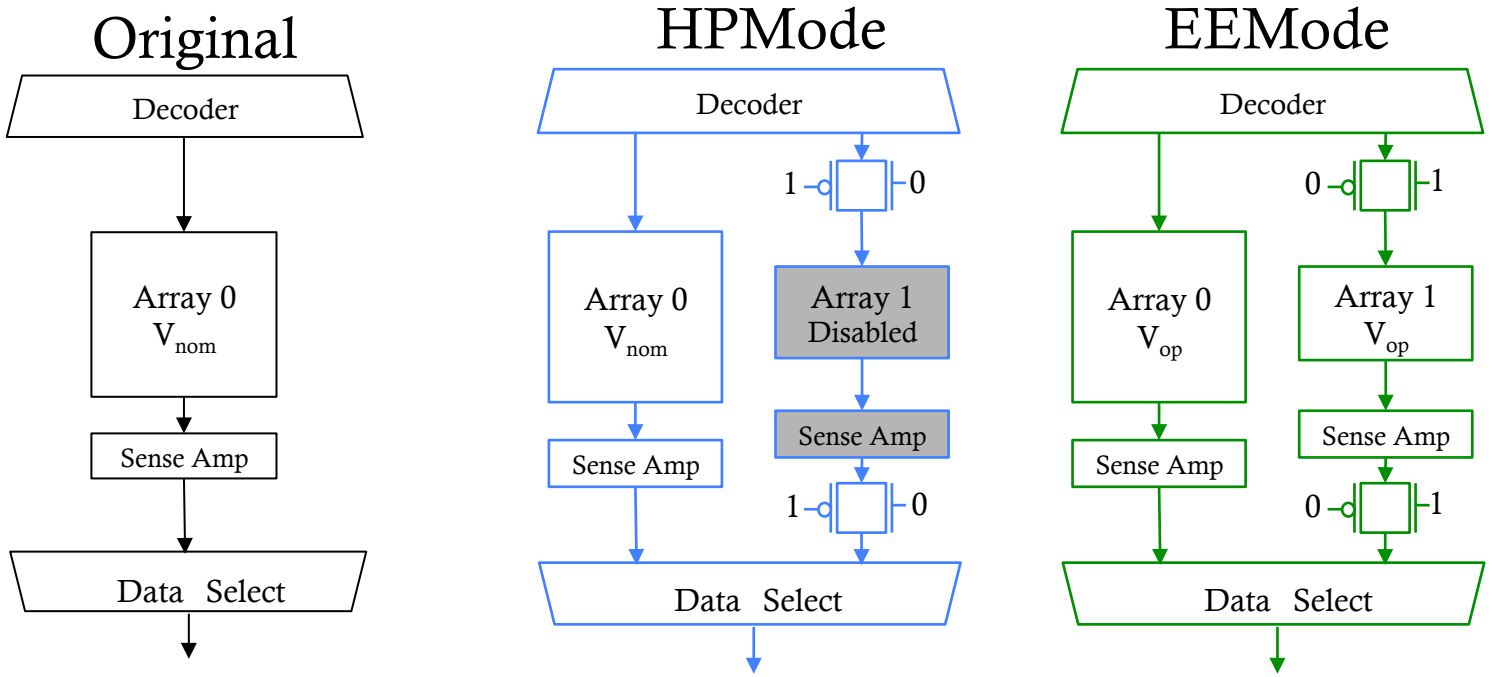
Fusing Two Pipeline Stages into One



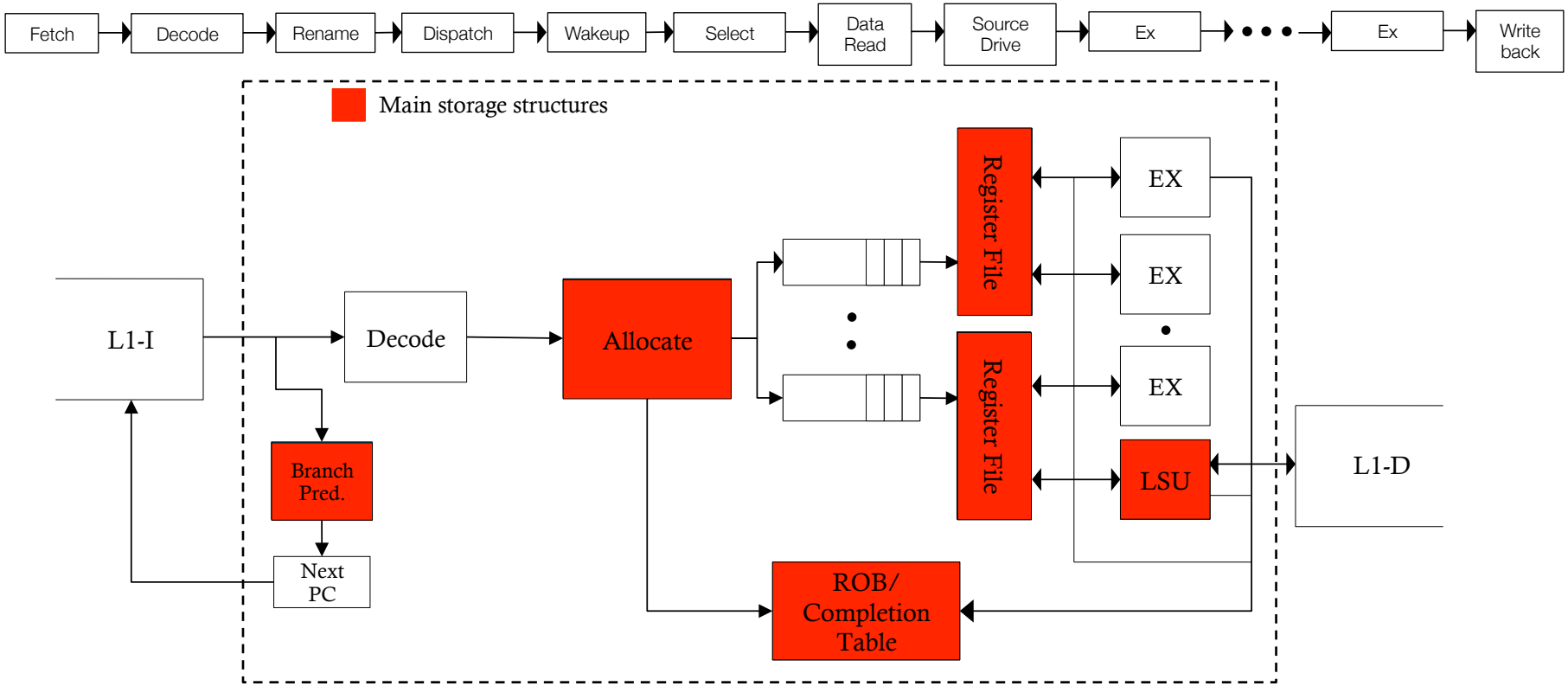
Fusing Two Pipeline Stages into One



Increasing Size of Structures



ScalCore Pipeline



Pipeline Structure Changes

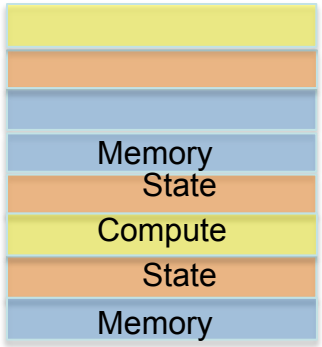
	Fuse Stages	Increase Size
Register File	Array access + Source drive	1.5X PRF
Allocation	Rename + Dispatch	---
Load Store Unit	Addr. generation + Memory disambiguation	1.5X Load/Store Queue and Store buffer
Reorder Buffer	---	1.5X ROB
Branch Prediction	---	---

Power-Gating: A Real Requirement

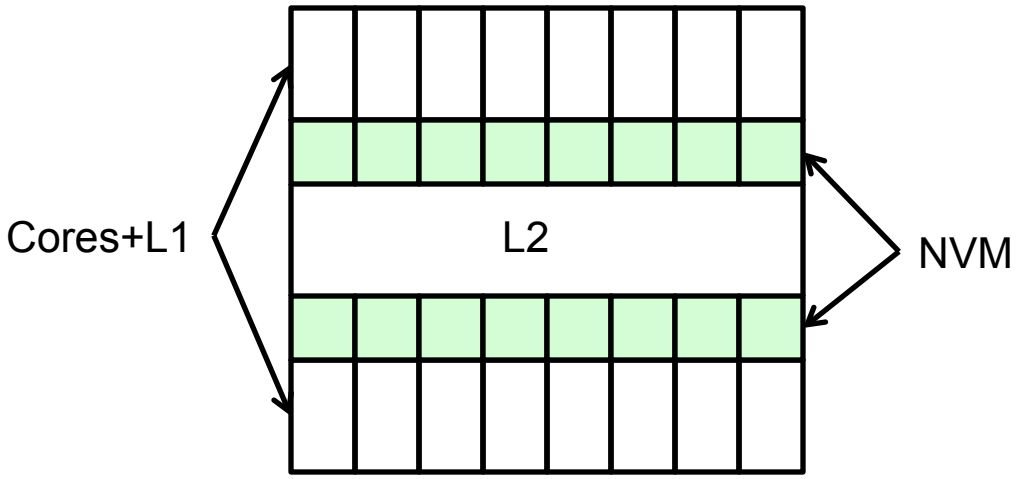
- Components not in use need to be power-gated
  - OS/software can do it sometimes, but has overhead
  - Many short idle periods; need HW-based power gating
    - Last-level cache miss
- Pipeline has significant state: Reg file, ROB
  - Need to micro-checkpoint the pipeline

Use Non-Volatile Memory for Micro-Checkpointing

- Challenge: NVM write latency
  - Need to bring NVM to < 10 cycles away



Monolithic integration

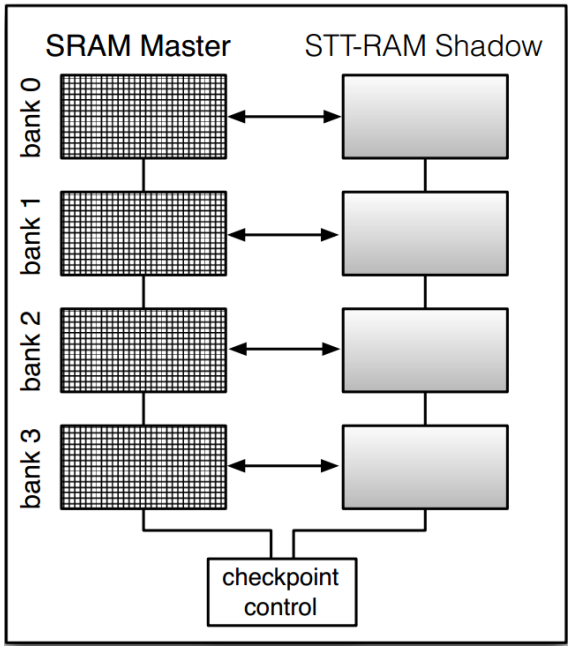


Same die integration

Shadow Latency-Sensitive Structures

[Teodorescu ICCD'14]

- Write-latency sensitive units:
  - Reg file, Inst window, ROB, Ld/st queue, pipeline regs
  - Implemented with hybrid SRAM/STTRAM
- Hybrid SRAM/STTRAM
  - SRAM for primary storage
  - STTRAM shadow of identical size used for micro-checkpointing
  - Banked design to parallelize checkpointing process

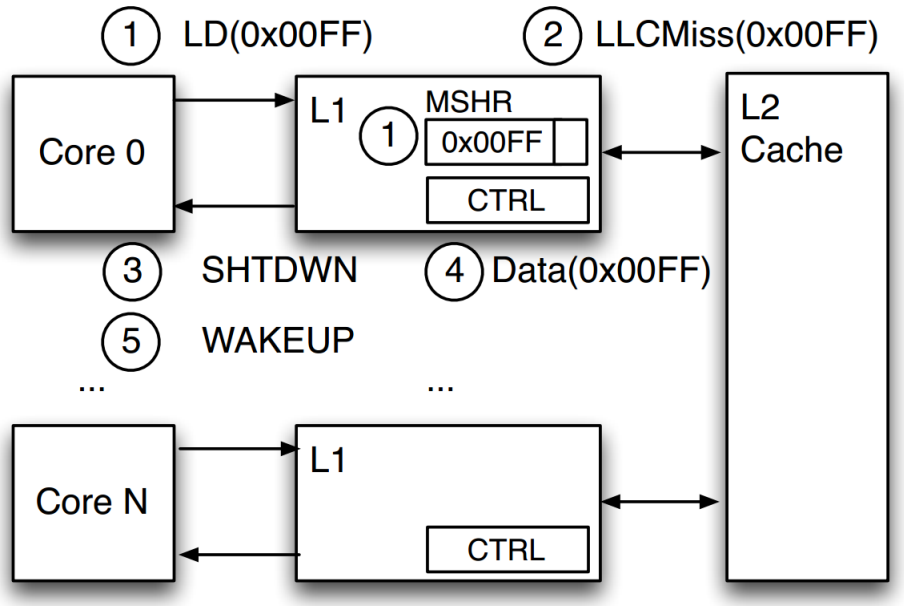


Hardware-Driven Implementation

– Checkpointing and wakeup of cores are coordinated by the L1 cache controller of each core

– Checkpoint/wakeup sequence:

1. LD issued, missed in L1
2. LLC miss reported by LLC
3. Sleep signal sent to Core 0
4. Missing data returns
5. Wakeup signal sent to Core 0



Sleep and Wakeup

- After receiving the sleep signal:
  - Checkpointing process initiates→ core waits until pipeline actually stalls → any newly modified entries will be checkpointed into SST-RAM → go to sleep
- During sleep period:
  - Caches and controllers are always kept on to maintain coherence requests when cores are shutdown
- After receiving the wakeup signal:
  - Core waits until the voltage supply is stabilized → restore saved data from shadow STT-RAM structures → resume normal execution

Software APIs are also Possible

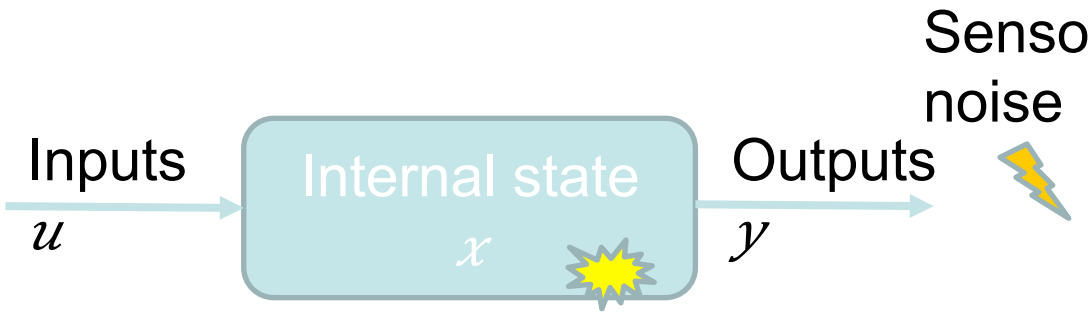
- Expose check-pointing to the system software
- Can be used by the OS or applications to “suspend” cores quickly
- Ideal for software observable idle events such as blocking on synchronization (e.g. barriers, locks, etc.)

Optimal Control

- Extreme scale computers need effective controllers [Skylake Package Control Unit]
  - Power, energy, temperature, utilization…

- Current approaches
  - Heuristics (with and without models)
  - Optimization
  - Machine learning
  - Control theory
    - Uses feedback to learn
    - Provides guarantees

Modeling a System with Control Theory

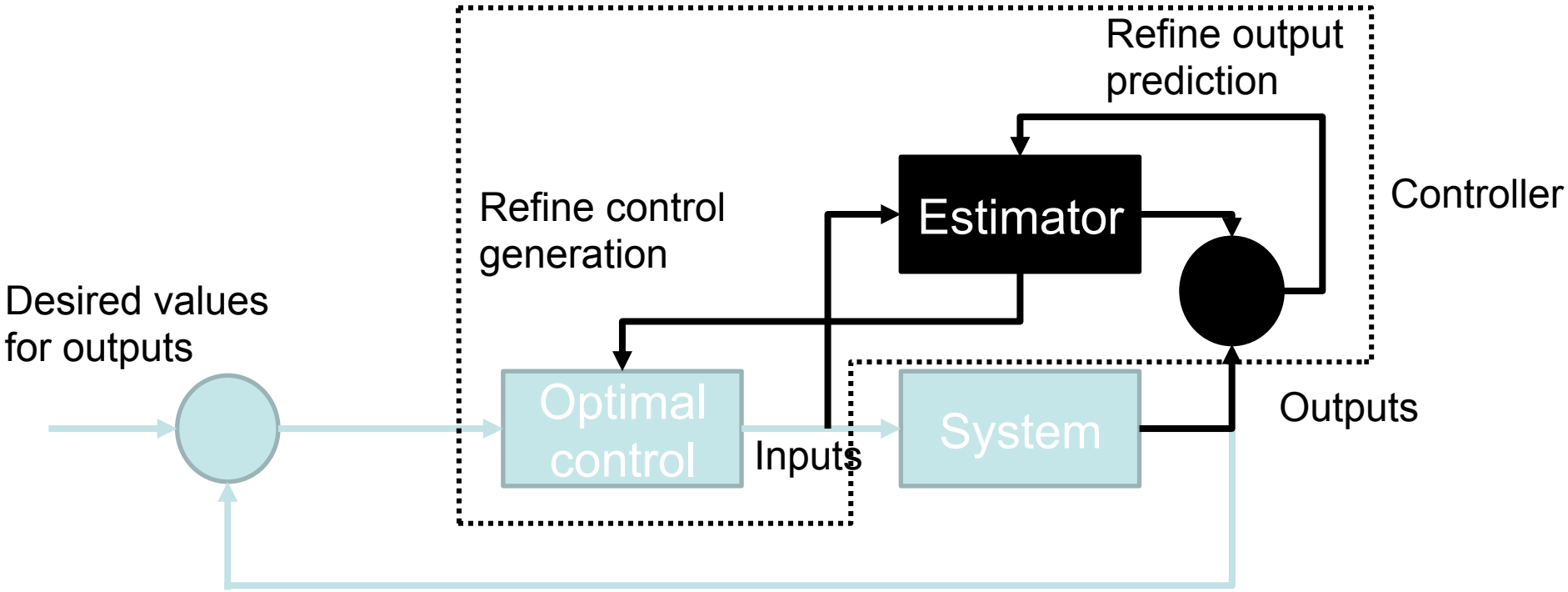


$$x(T+1)=Ax(T)+Bu(T)$$

$$y(T)=Cx(T)+Du(T)$$

- State evolves depending on the current state and inputs
- Outputs are a function of the state and inputs
- The model is {A, B, C, D} + Unpredictability matrices
  - Obtained from analytical formulas or experimental characterization

Control Overview

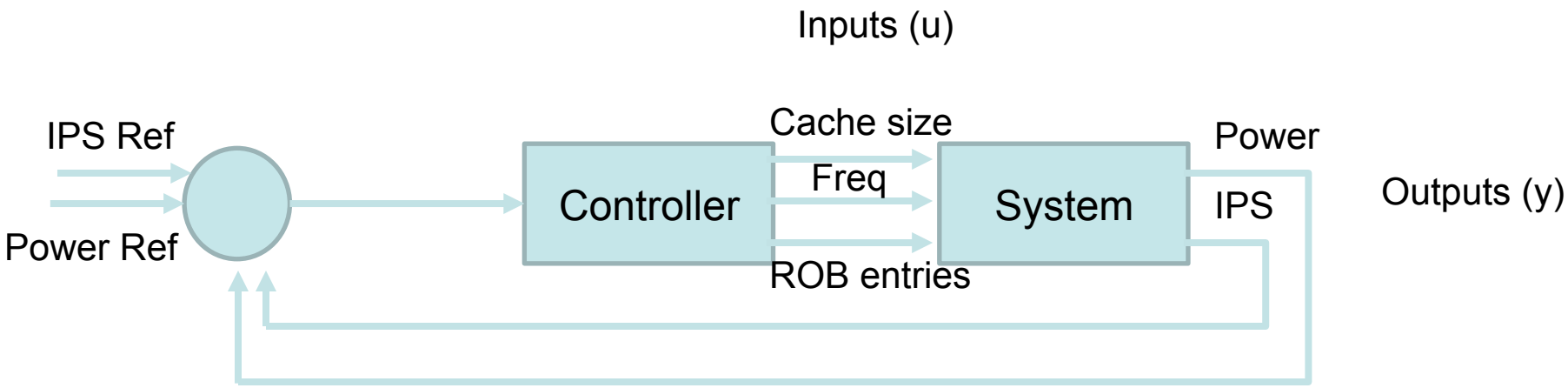


With control theory, the output eventually matches the desired value

Applying Control Theory to Architecture

[Pothukuchi ISCA'16]

- Control theory has been used for decades in other areas of engineering and sciences
- Its use in architecture has largely been limited to single output (SISO, MISO) controllers
- Need to use of Multiple Input Multiple Output (MIMO) control



MIMO Controller Details

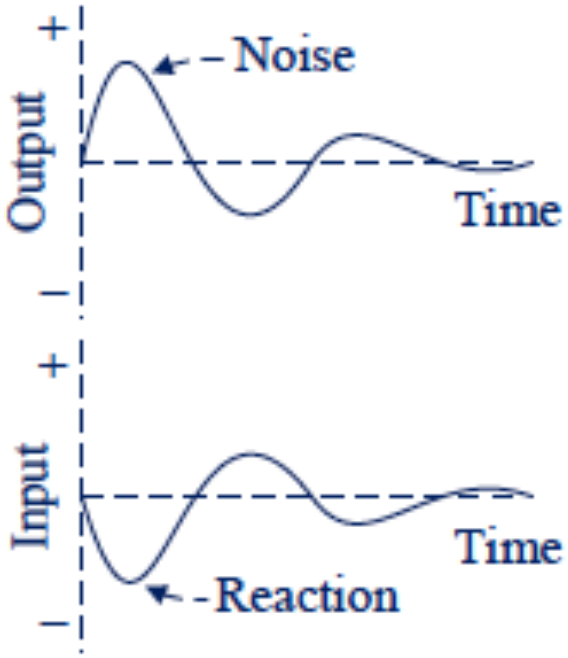
- We use a design called Linear Quadratic Gaussian controller
  - Each input and each output has a cost (or weight)
- Cost of an input: How hard it is to change it from its current value
- Cost of an output: Cost of not meeting the target of the output

Type of Weight	Qualitative Weight Ranking (From High to Low)
System Outputs	Voltage guardband, Temperature, Power, Core Utilization, Energy, Frame rate, Instructions per Second (IPS)
System Inputs	Cache power gating, core power gating, frequency, issue width, ld/st queue entries

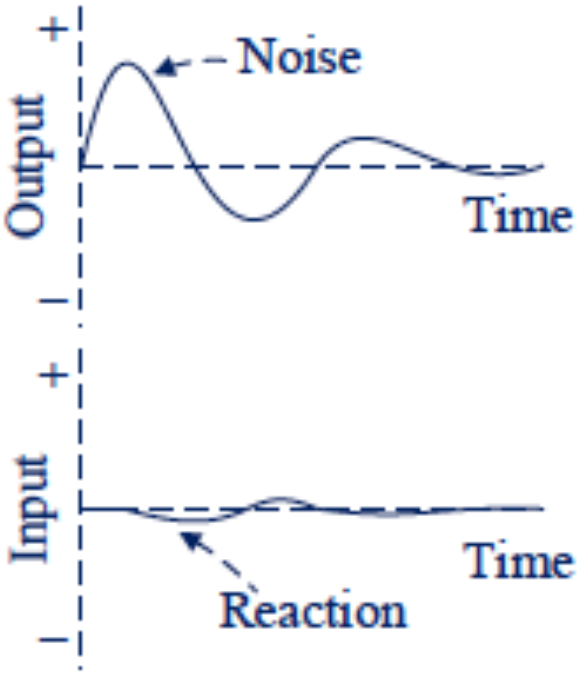
- System will try to minimize the changes to costly inputs/outputs

MIMO Controller Details

- Relative cost of inputs & outputs controls the inertia of the system

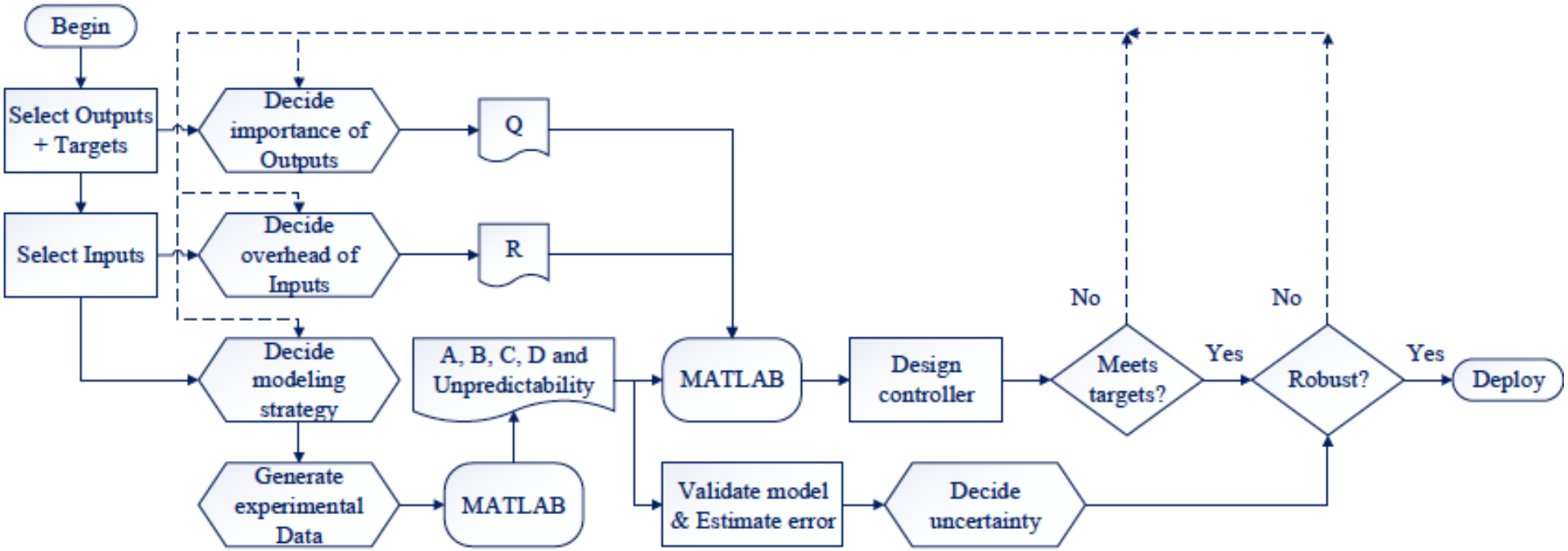


Input weights are low wrt outputs:
Ripply system



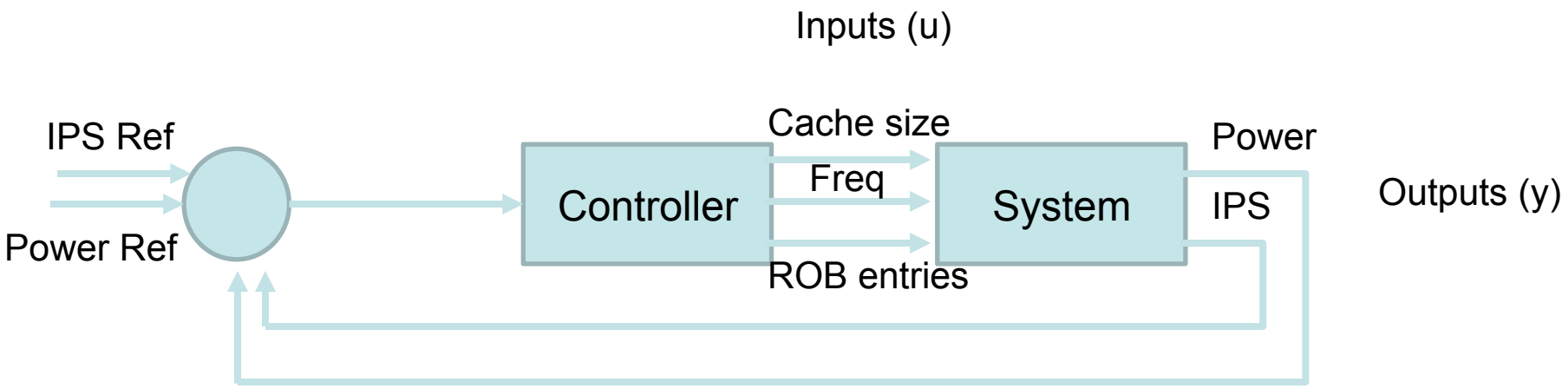
Input weights are high wrt outputs:
System with inertia

Automated Controller Design with MATLAB



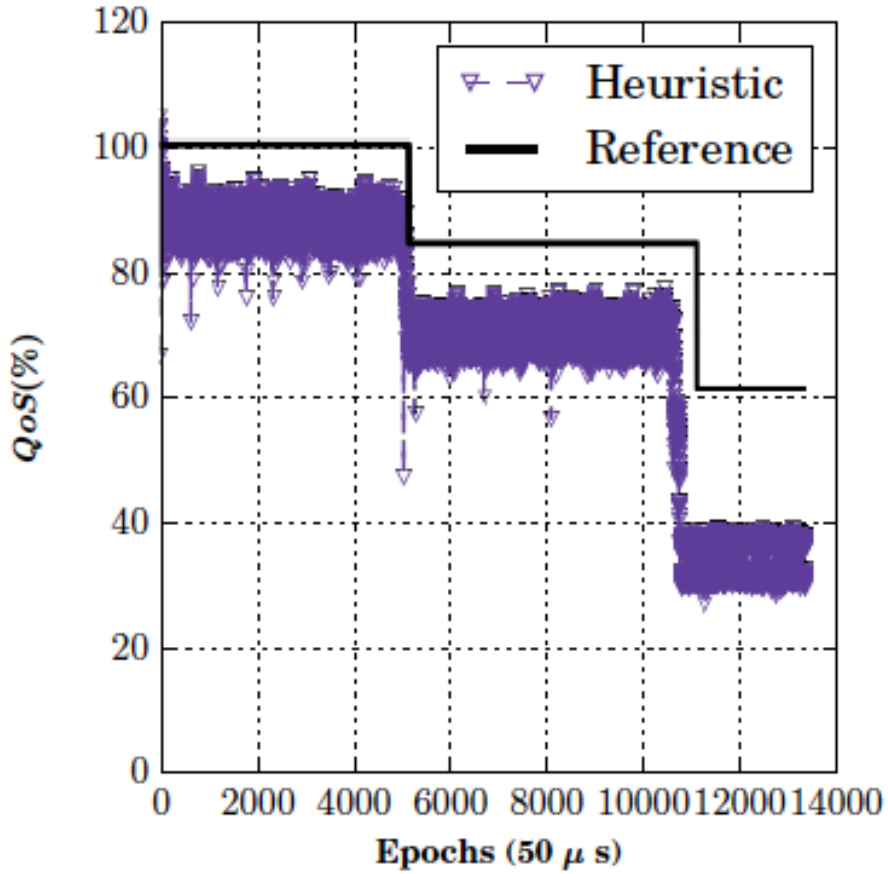
Uses of the Controller (I)

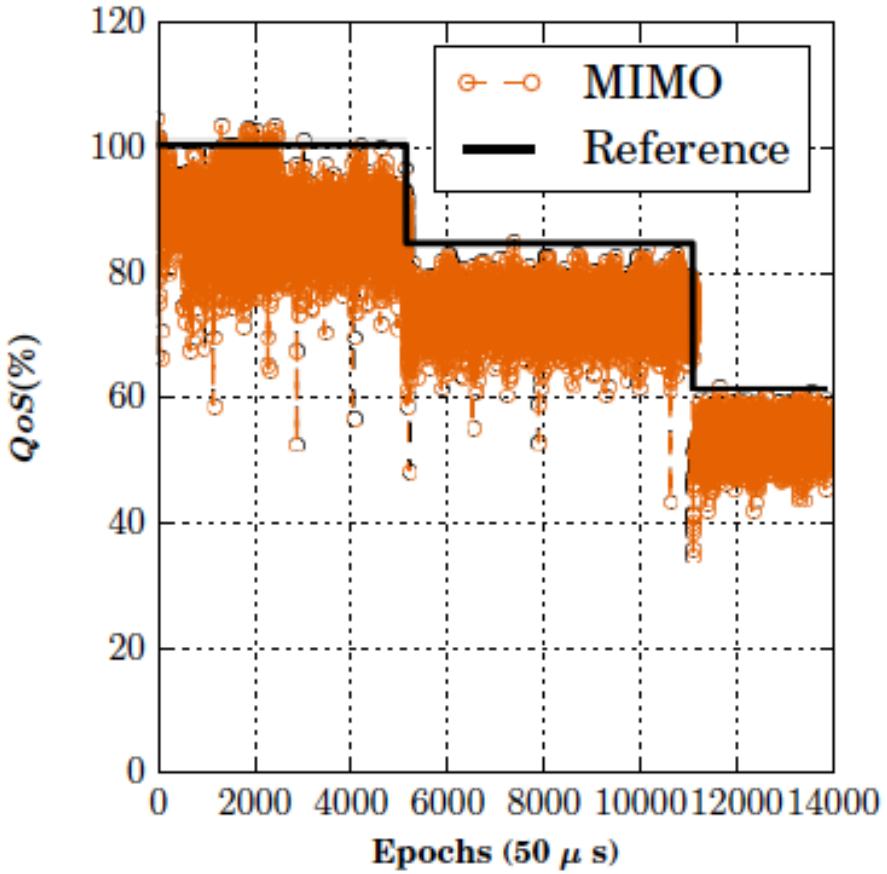
- Track multiple output references
  - Performance (IPS) and Power (P)



Uses of the Controller (II)

- Time varying tracking
  - Changing the QoS and power levels as the battery is depleted in a mobile device

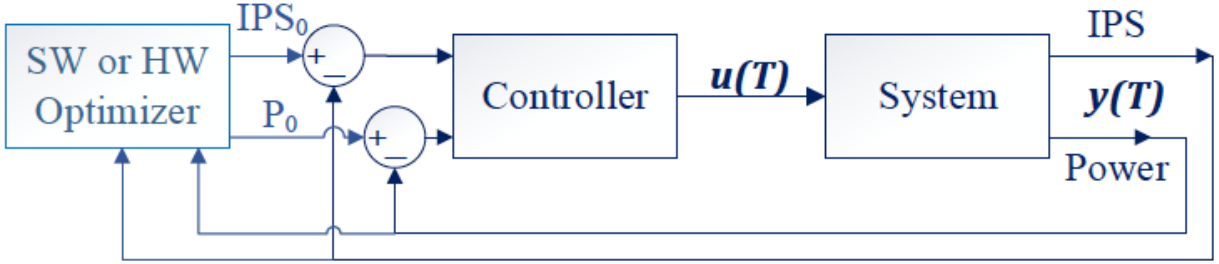


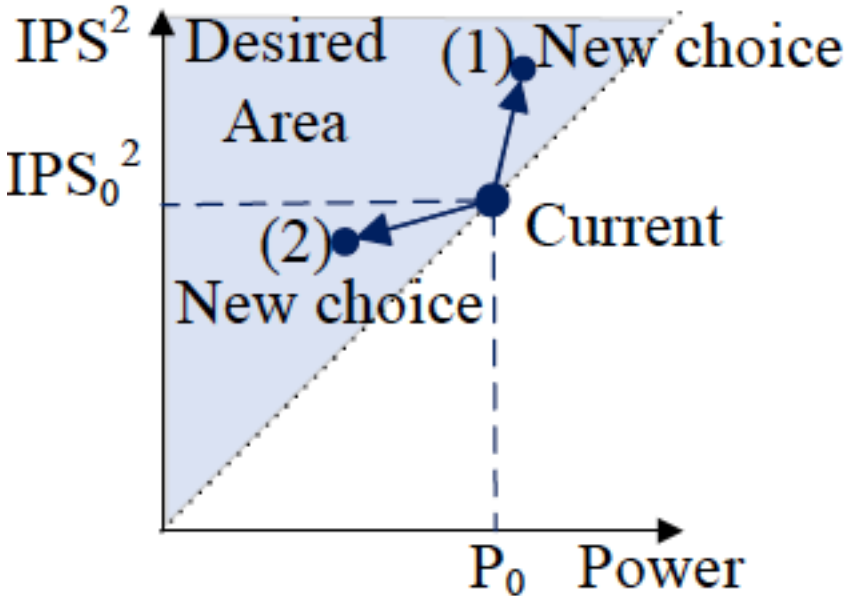


Uses of the Controller (III)

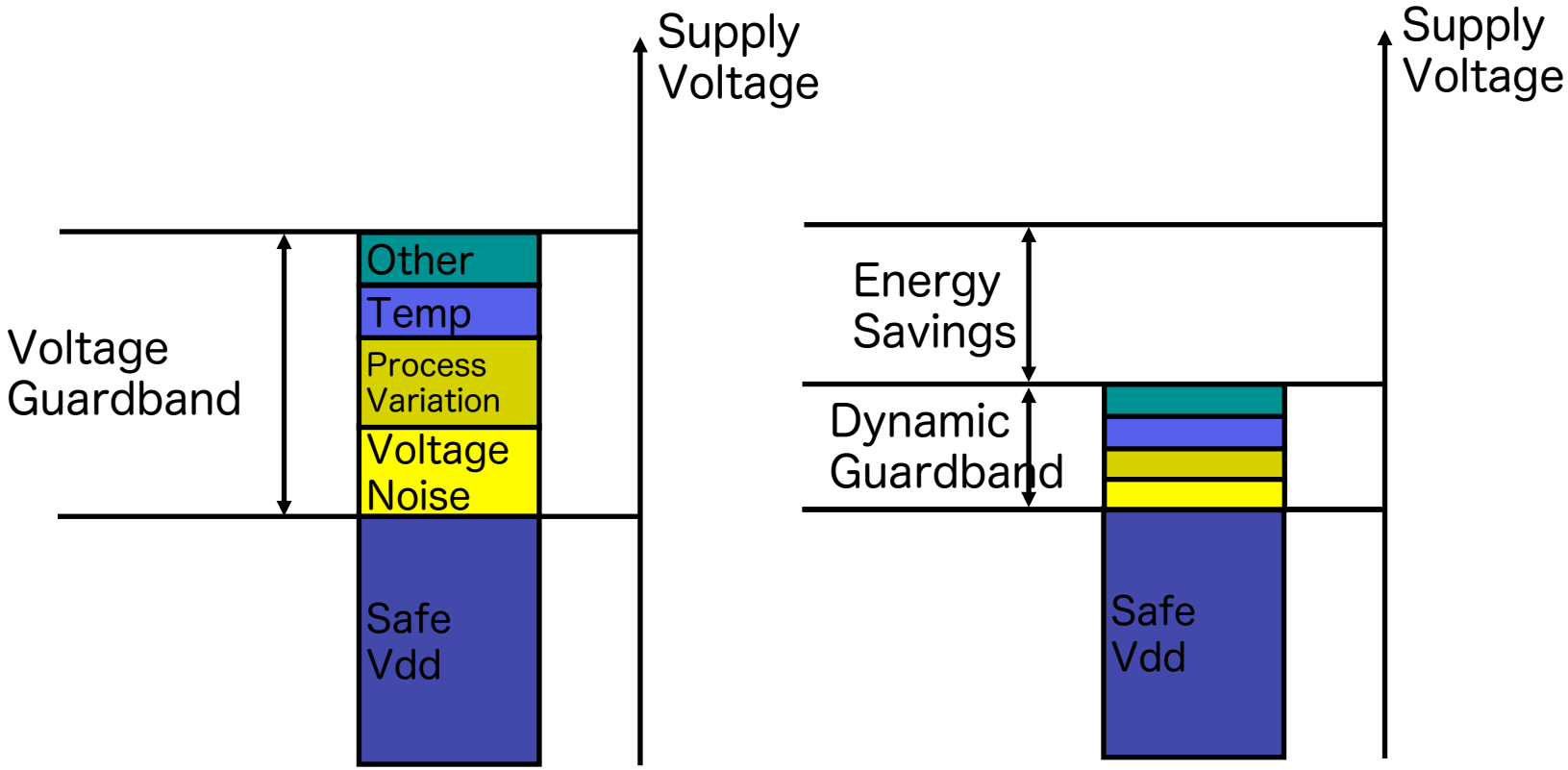
- Fast optimization of a composite measure
  - Minimizing (ED^n)= maximize($IPS^{(n+1)}$/Power)

Search directly in the (IPS,P) space rather than in the hardware input space



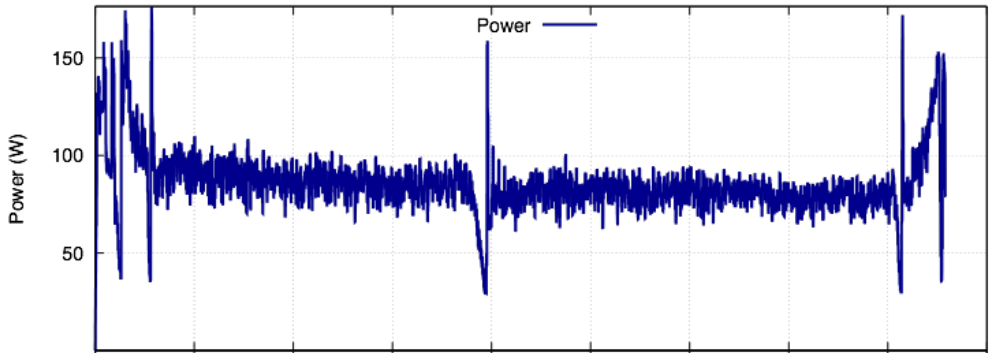


Key: Reduce the Voltage Guard-band

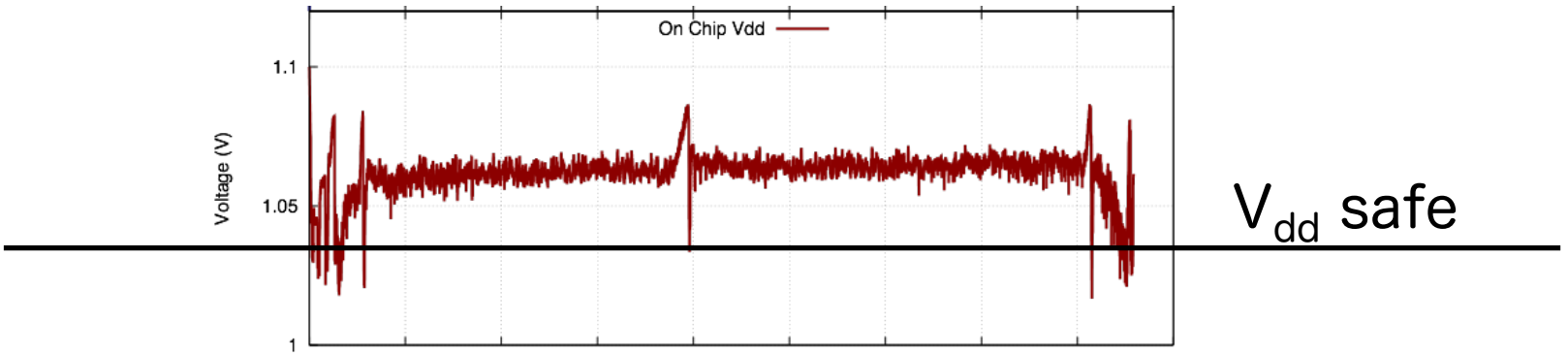


Many Dangers Lurking: di/dt

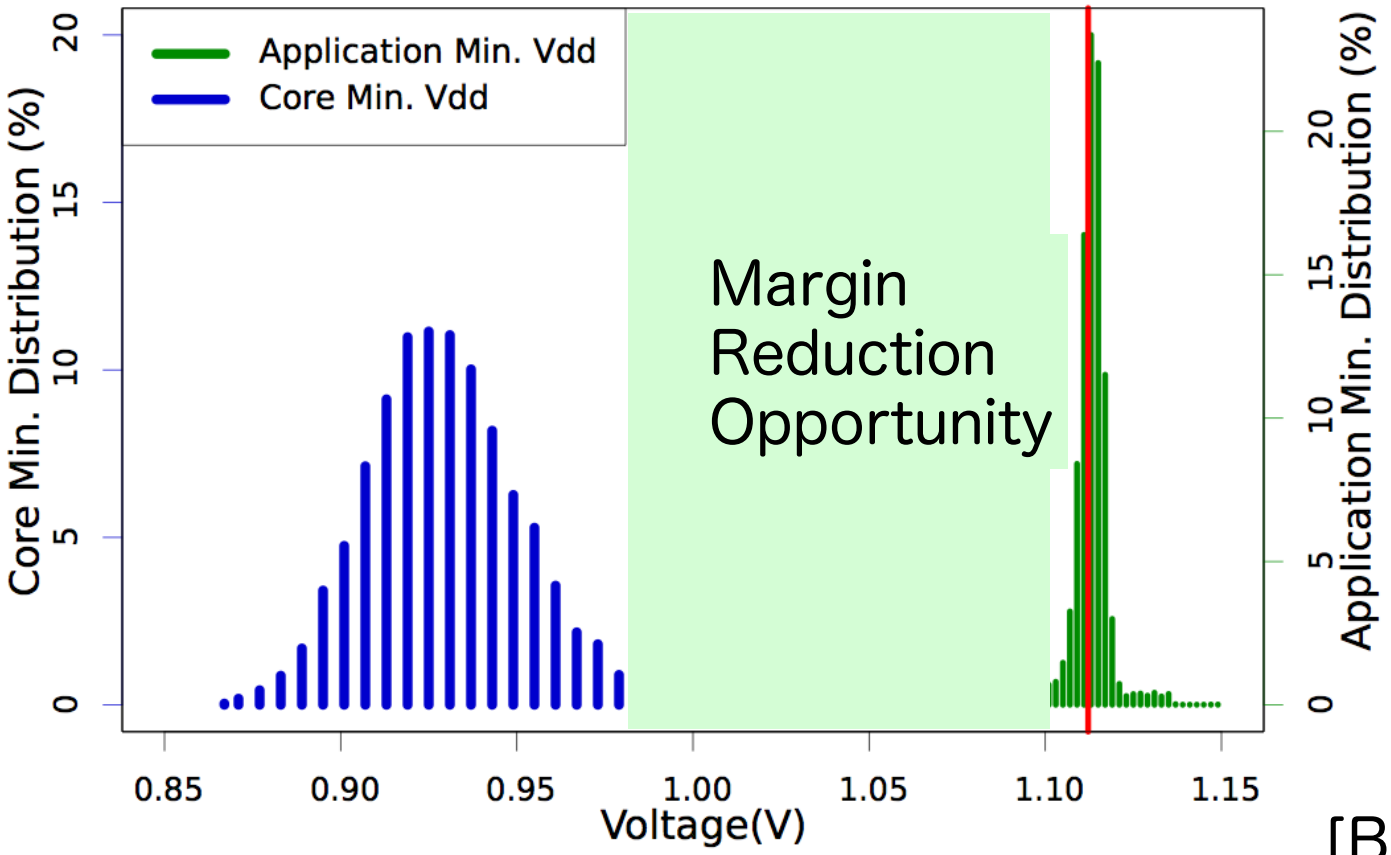
- Power demand fluctuates with workload



- Supply voltage is affected by changes in power demand



Noise and Variation Effects



[BFS]

Core Tunneling

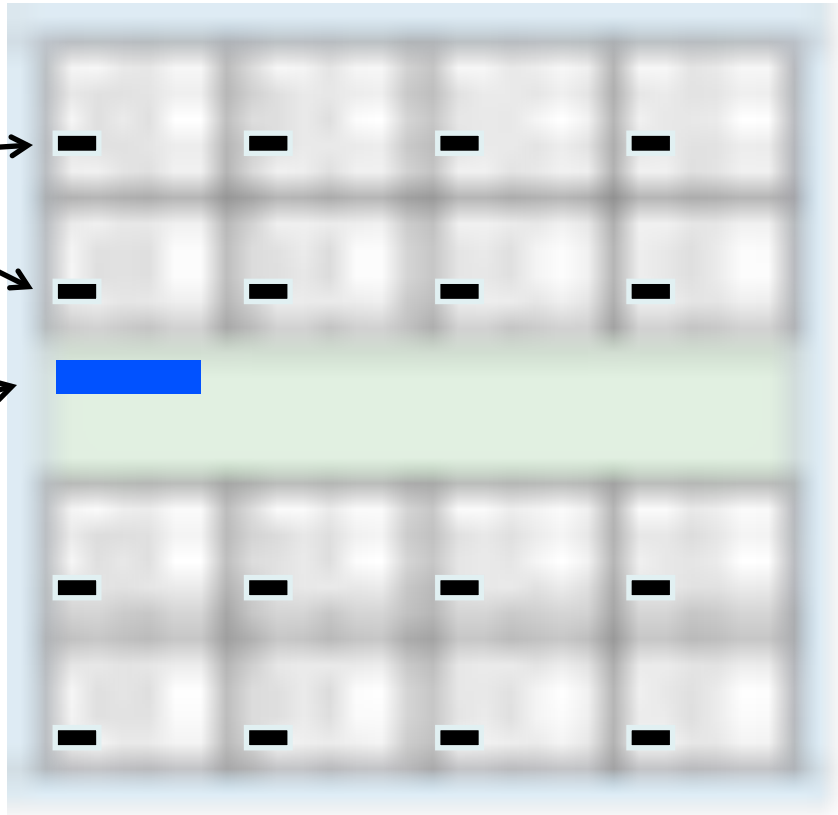
[Teodorescu HPCA'16]

- Reduce margins dynamically by reducing the voltage
- Safety margins will be exceeded *sometimes* by some units
- When critical slowdown is detected, an entire unit is clock-gated

Core Tunneling

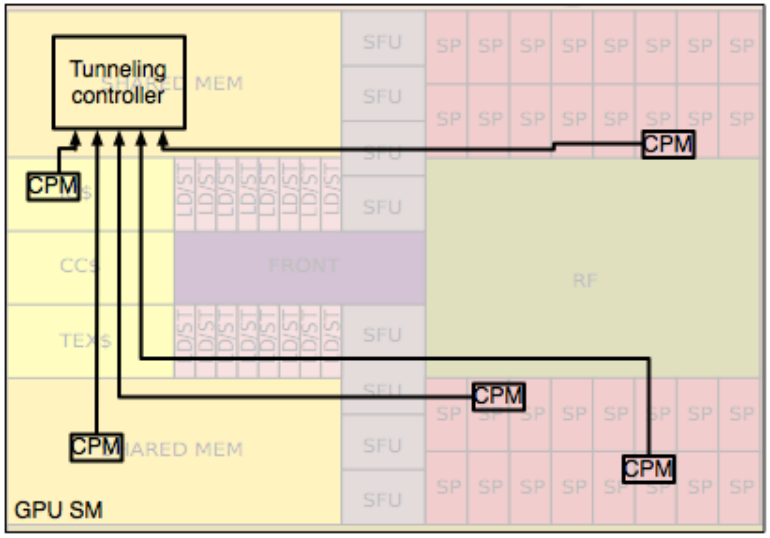
Tunneling Controller:
Noise Detection
and Mitigation

Chip-level Voltage
Management



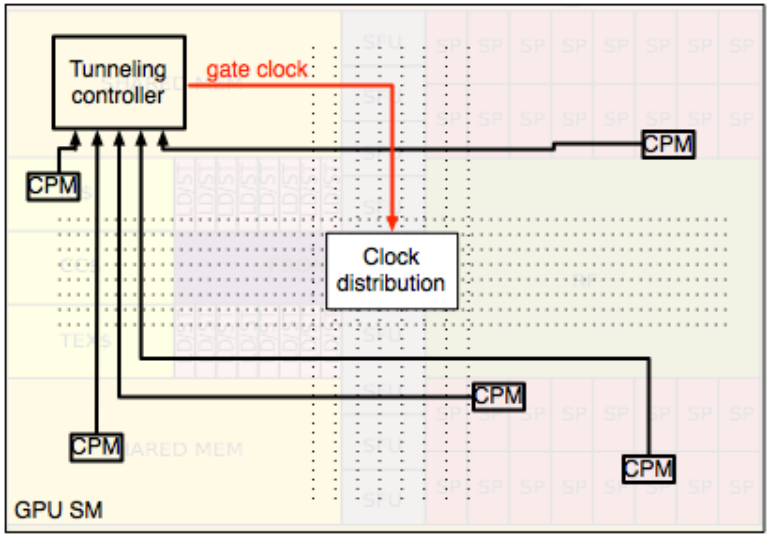
Noise Detection

- Critical Path Monitors (CPMs) used to detect critical circuit slowdowns due to voltage droops
  - Fast delay monitoring circuits, simple logic, low-cost
  - Deployed in production on the IBM POWER 7

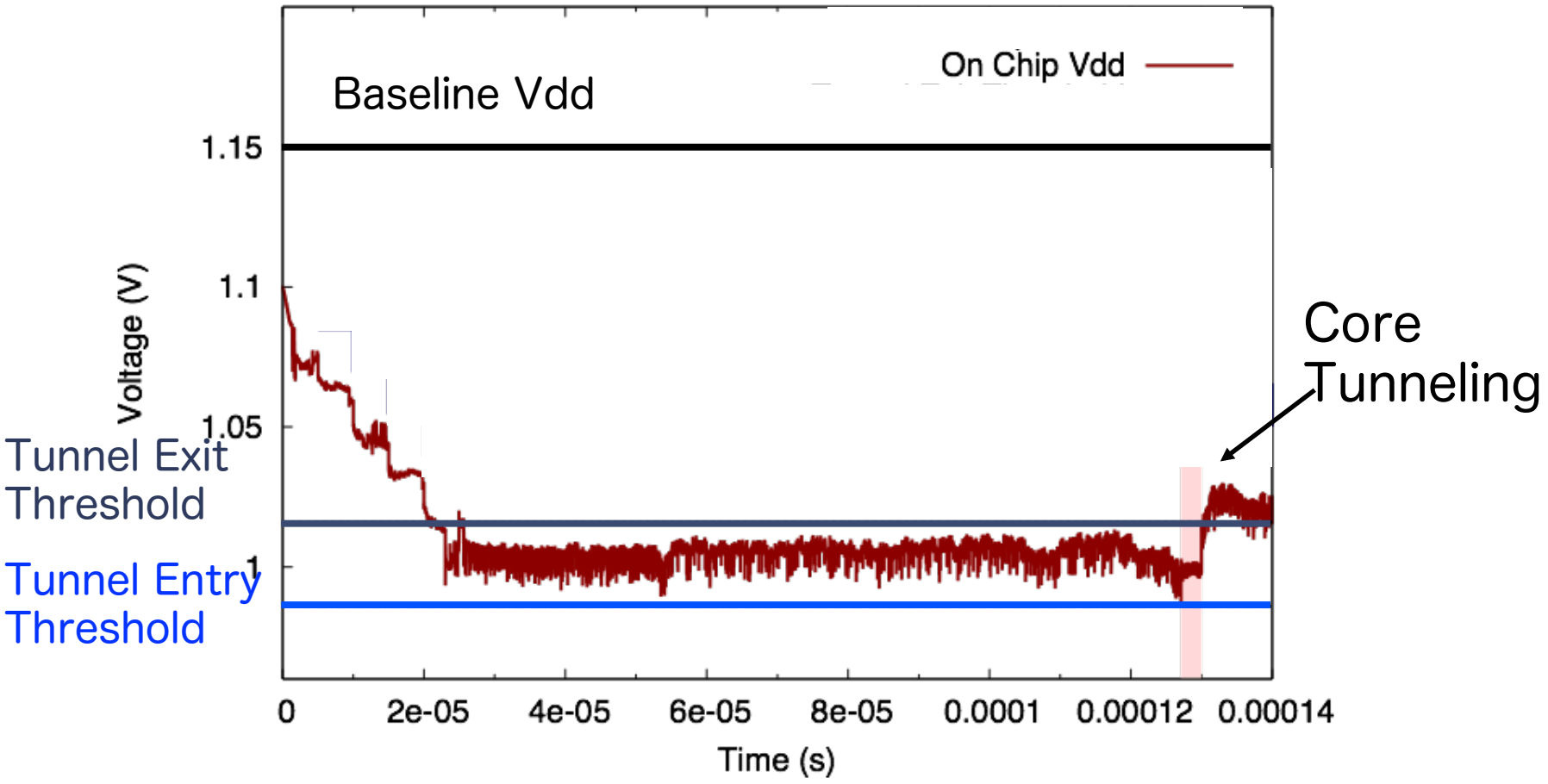


Noise Mitigation

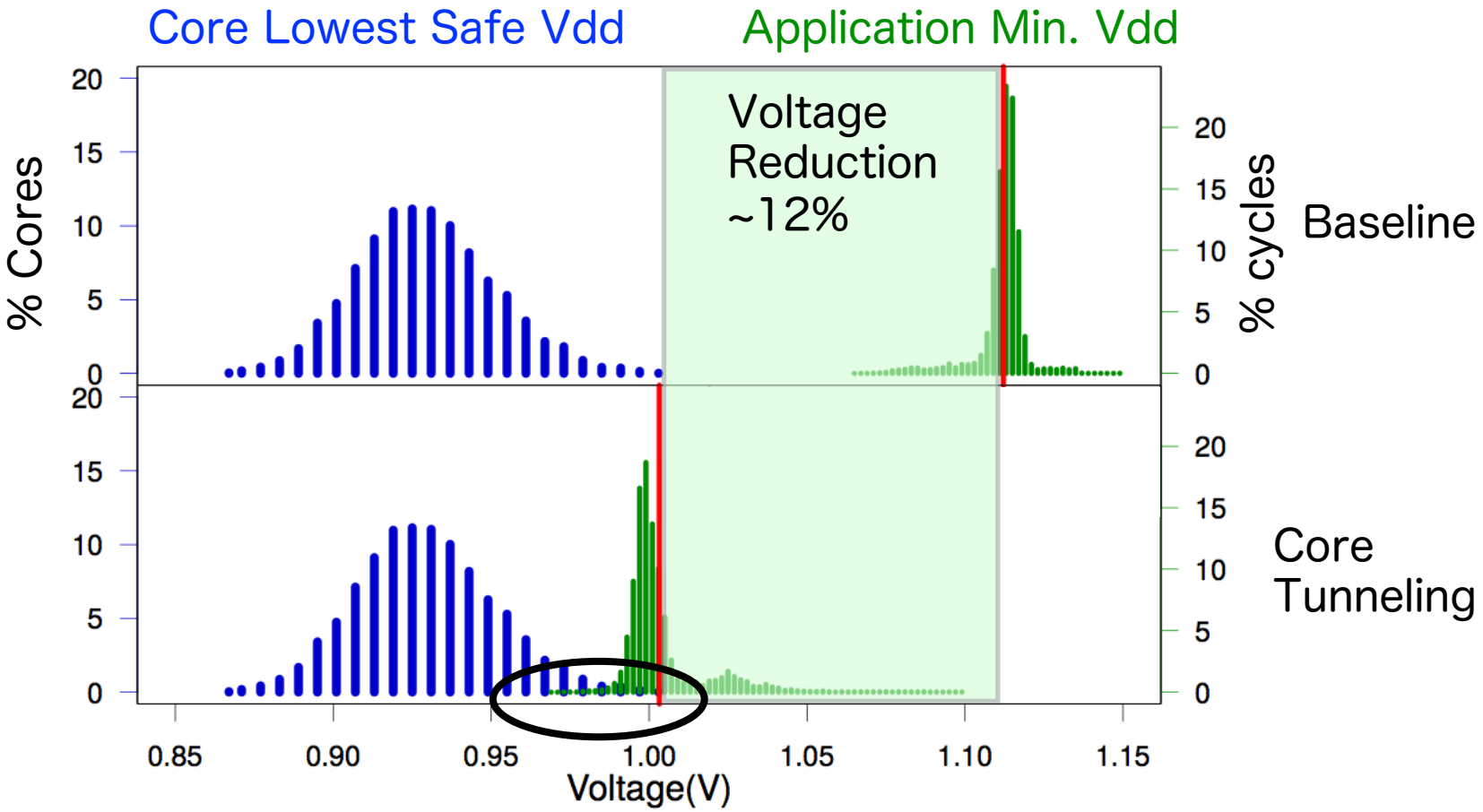
- When critical slowdown is detected, the entire core is clock-gated
  - Since no computation while clock gated, the core's voltage can drop below "safe" margin
  - SRAM is also safe since voltage is not lowered below retention



On-Chip Level Voltage Management



Voltage Margin Reduction



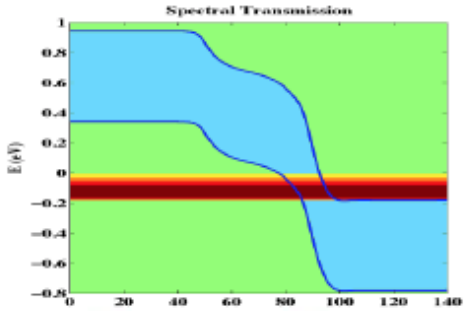
Upcoming Technologies

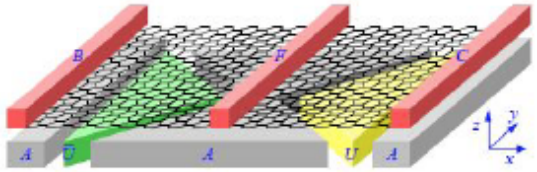
- Picture is unclear
- If we want energy efficiency, we get low performance and need to rely on more parallelism (many more cores)
- Likely a combination of technologies in the same die/stack

Beyond CMOS Devices

1. Electronic

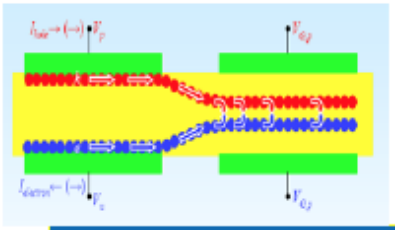
Tunneling FET





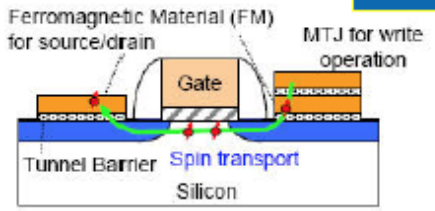
Graphene pn Junction

BisFET



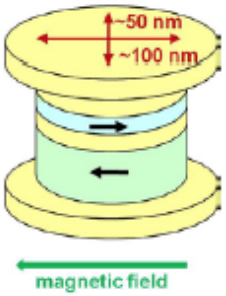
3. Orbitronic

2. Spintronic

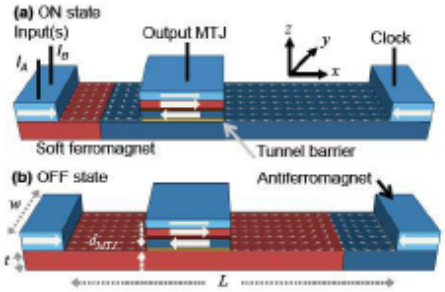


SpinFET

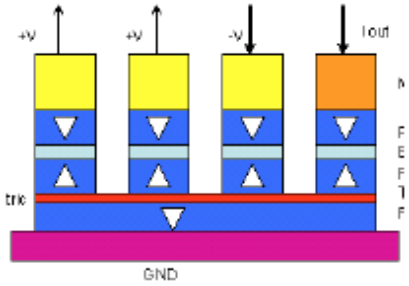
All Spin Logic



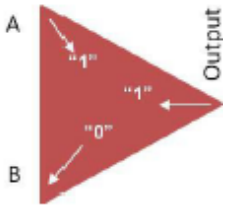
Spin Torque Oscillator



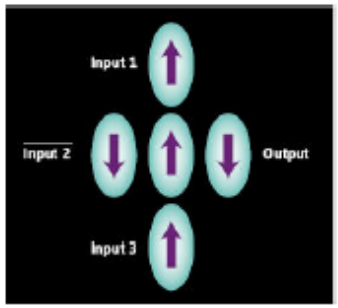
Domain Wall Logic



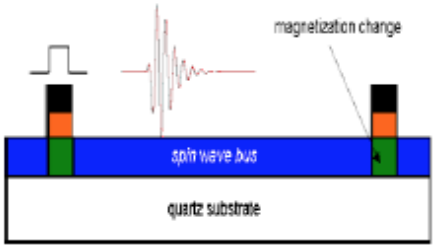
Spintronic Majority



Spin Torque Triad

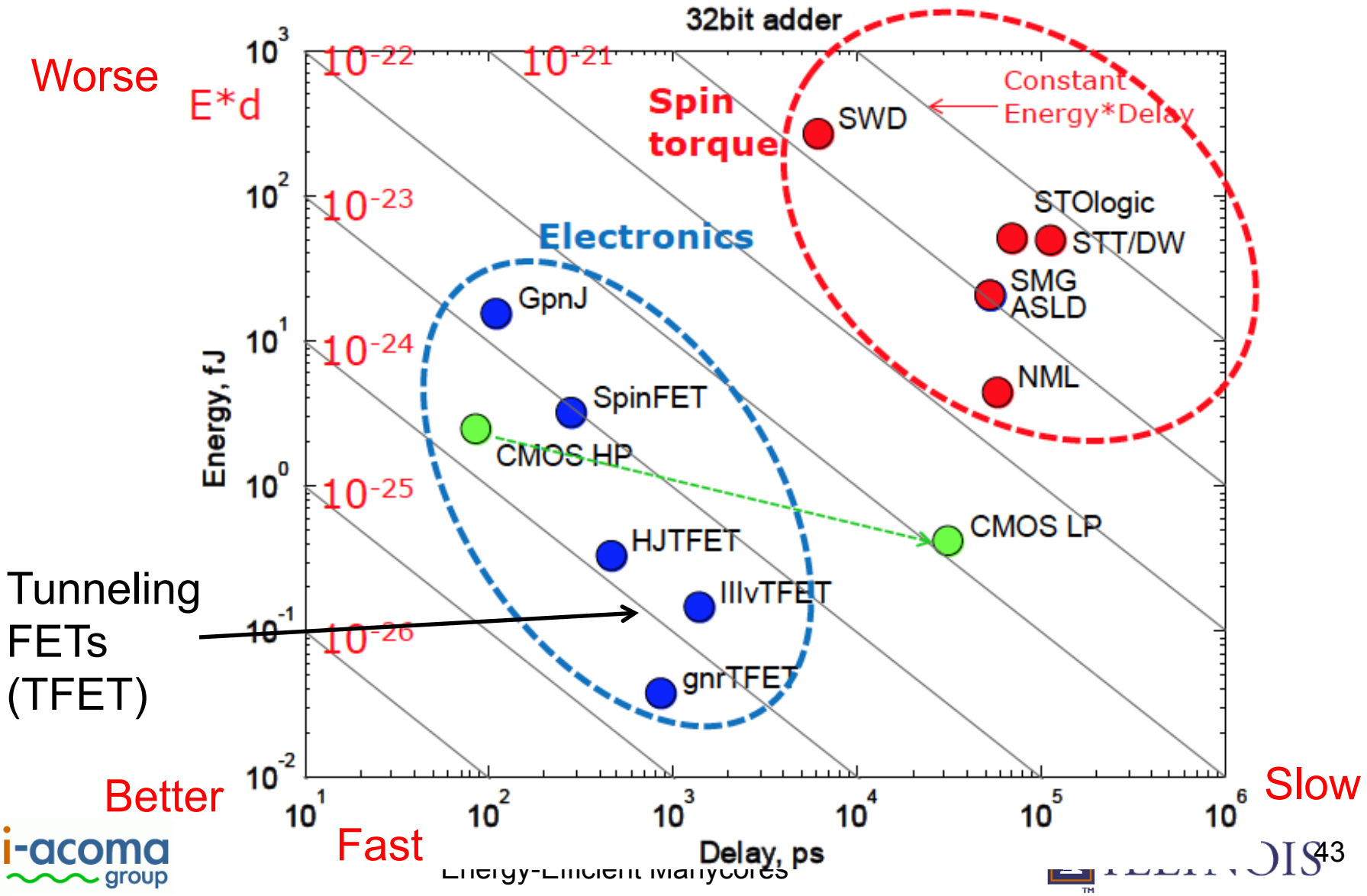


Nano Magnet Logic

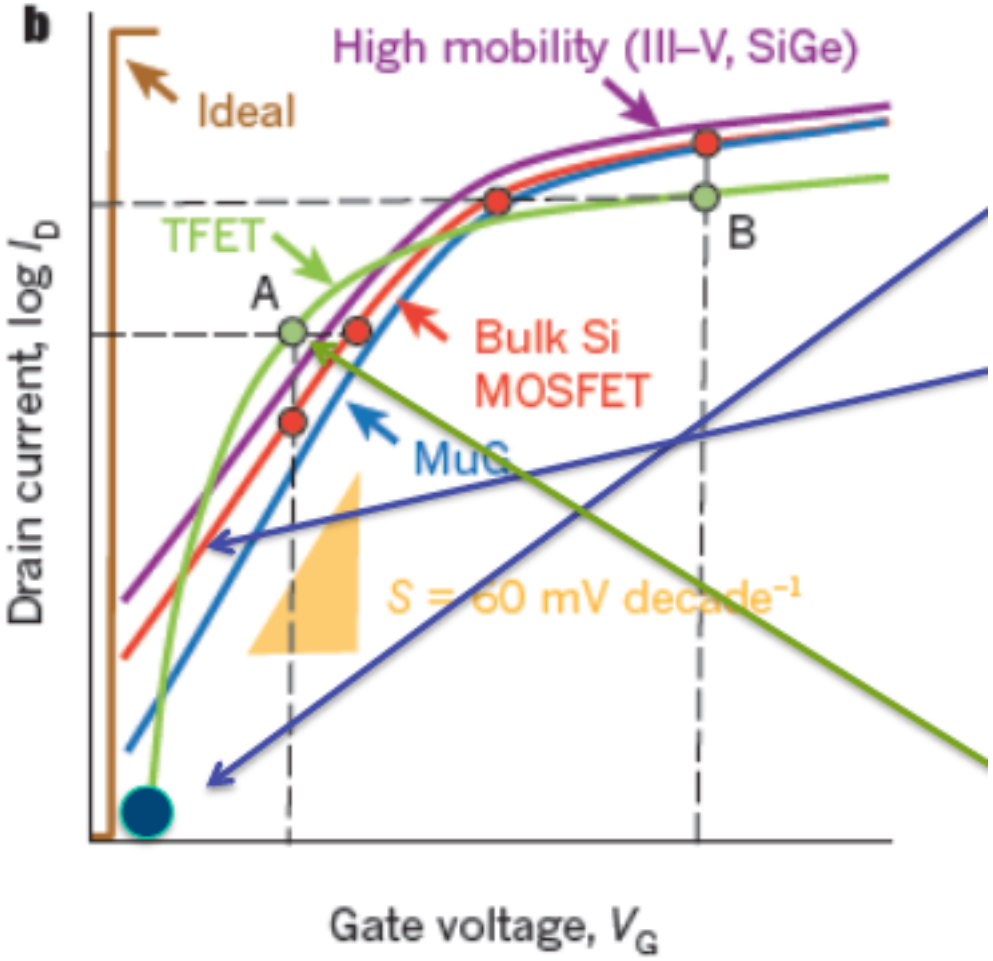


Spin Wave Device

Energy vs Delay



TFET vs CMOS



A.M. Ionescu, H. Riel, Nature, Vol. 479, pp.329-337, 2011.

Low I_{OFF}

Steep characteristic, hence improved I_{ON}/I_{OFF} ratio

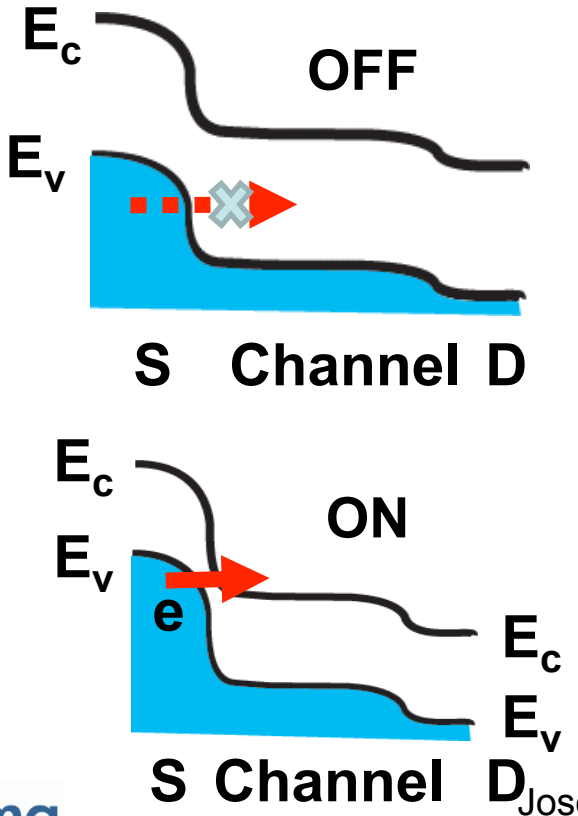
Working point carefully optimized

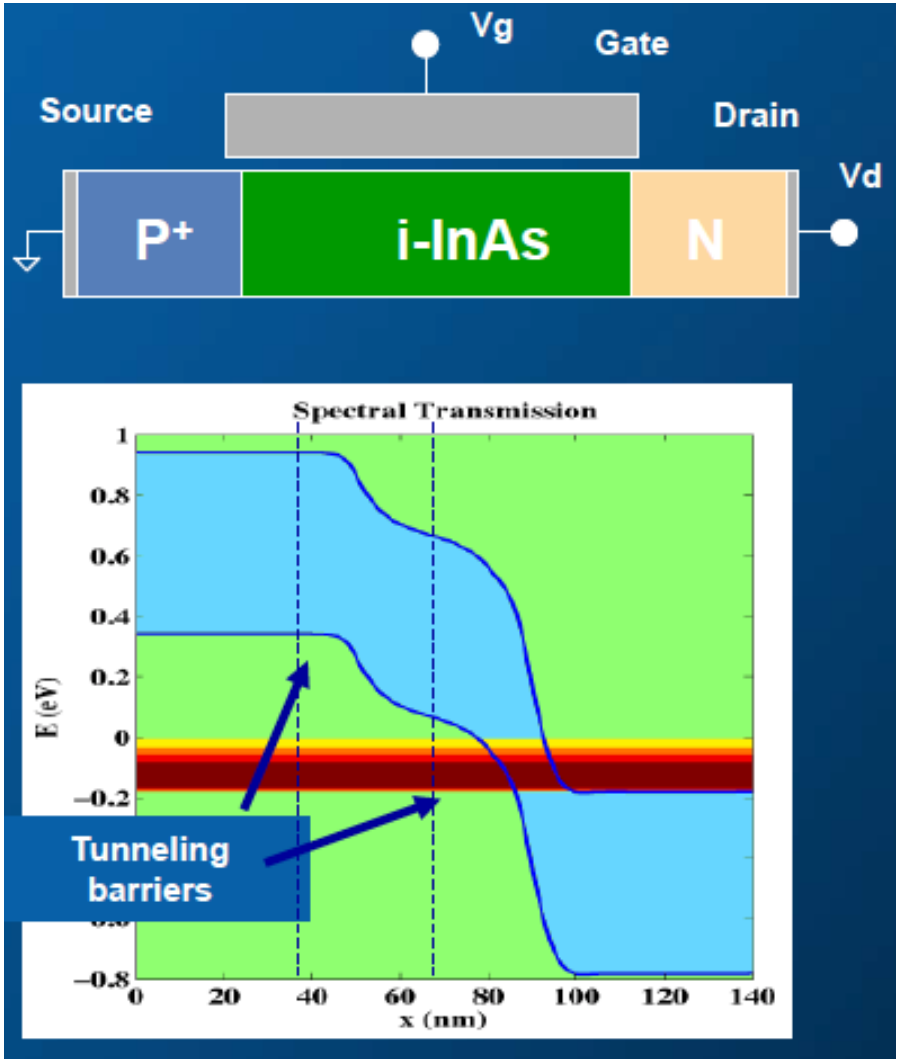
TFET vs CMOS

- TFET:
  - Not as fast as CMOS
  - More energy efficient at low Vdd
  - Dissipate much less static power
  - Need R&D to make it to production
  - There are different types (homo- and hetero-junction)
  - For logic, and maybe SRAM
  - Scalable

Why TFET Has Steep Characteristic

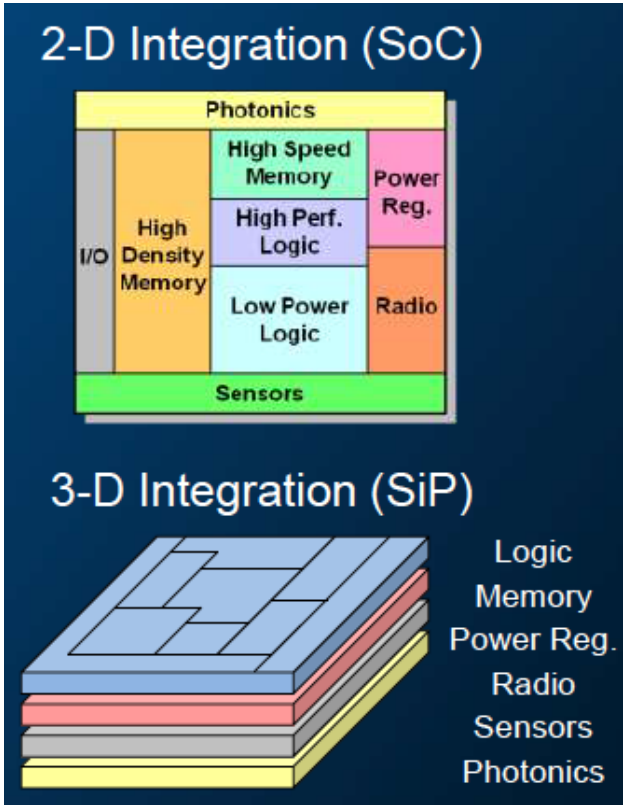
- TFETs operate by tunneling through the S/D barrier, rather than diffusion over the barrier





What Will Happen?

- TFET can be fabricated on the same die as CMOS
- Heterogeneous architectures?



Conclusion

- Lots of room to innovate in computer architectures, to make them more energy efficient
  - Voltage-scalable cores
  - Pervasive power gating
  - Control-theoretic controllers
  - Beware of voltage noise
  - Learning to deal with new technologies
- Luckily, Big Data workloads have a lot of parallelism

Energy-Efficient Manycore Architectures for Big Data

Josep Torrellas

Department of Computer Science
University of Illinois at Urbana-Champaign
http://iacoma.cs.uiuc.edu

BPOE

April 2015