

Toward Extreme-Scale Processor Chips

Josep Torrellas

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

HiPC 2016

Hyderabad, India

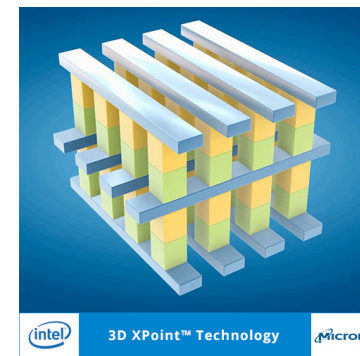


Accelerated Progress in Transistor Integration

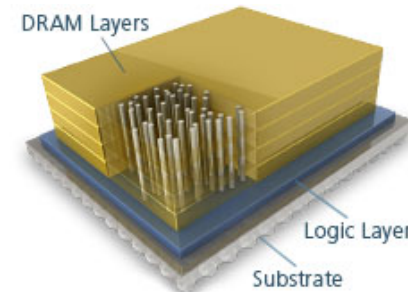
- Large multicores for data centers and cloud
- 3D stacked chips



Intel Xeon Phi 7290F (Oct 2016)
72 cores, 288 contexts, 260W



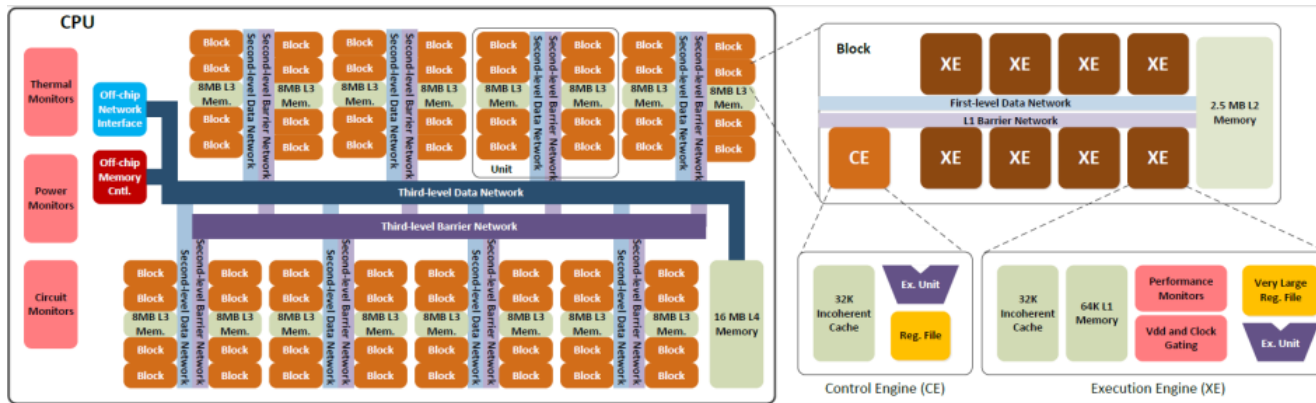
Intel 3D Xpoint memory



Micron's Hybrid Memory Cube

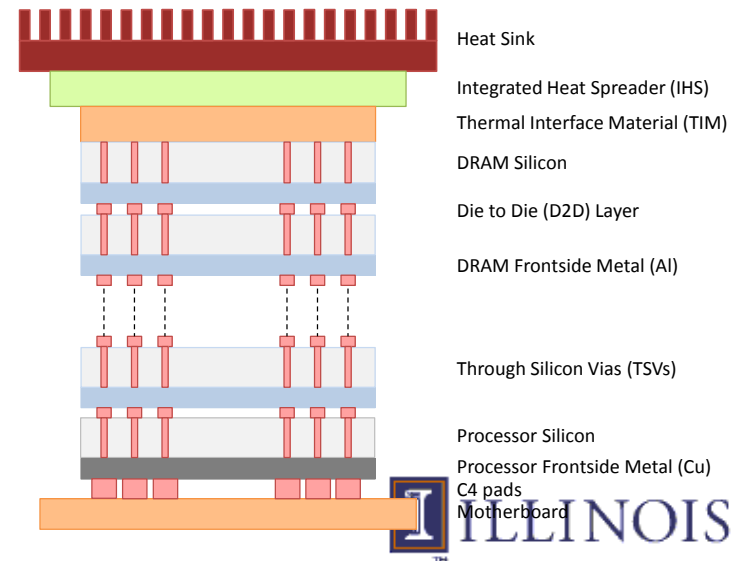
Research is Pushing Ever Farther Ahead

- More integration → 1,000 cores/chip



Runnemedde prototype [HPCA-13]

- Research on stacking multiple processor and memory dies



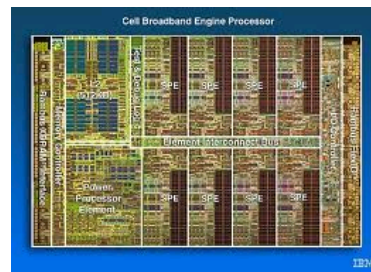
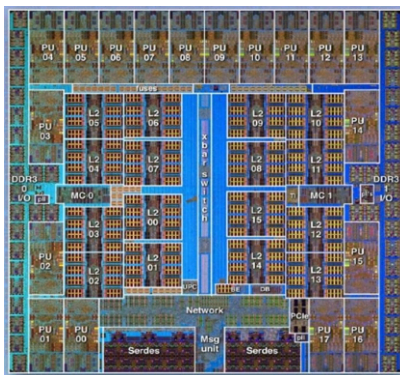
Meanwhile: Energy Wall... and Performance Wall

- University of Illinois Blue Waters Supercomputer



Performance: 11 PF
Power: 6-11 MW (idle to loaded)
1MW = \$1M per year electricity

- Technology improvements in speed and power slowing down



What We Need

- Very high energy efficiency
- Faster communication and synchronization
- Ease of programming

Energy Wall: How Did We Get Here?

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

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

Constant dynamic power density

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

Power density increases rapidly

Energy Efficiency: Low Voltage Operation

- V_{dd} reduction is **the best lever** for energy efficiency

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

Static power: $P_{sta} \propto V_{dd} T^2 e^{-qV_t/kT}$

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

Attaining Very High Energy Efficiency

- Voltage-scalable cores
- Dynamic voltage speculation
- Pervasive power gating
- Control-theoretic controllers

Three rules:



Reduce voltage

Turn-off if unused

Minimize waste

Attaining Very High Energy Efficiency

- **Voltage-scalable cores**
- Dynamic voltage speculation
- Pervasive power gating
- Control-theoretic controllers

Three rules:

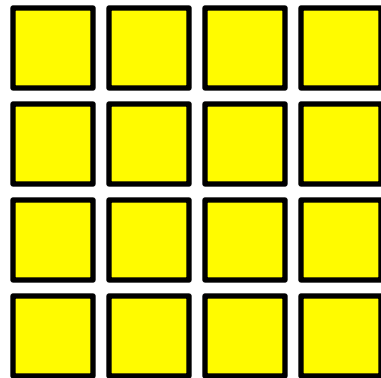
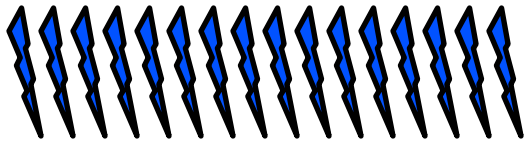


Reduce voltage

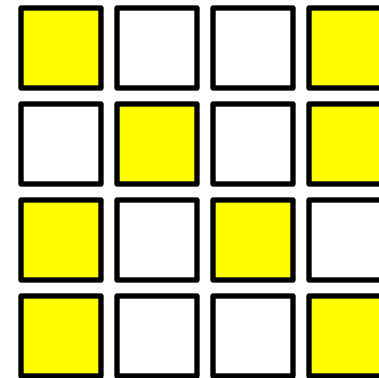
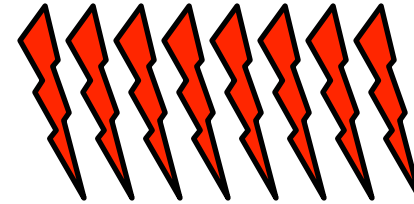
Turn-off if unused

Minimize waste

Goal: A Voltage-Scalable Core



Go to low voltage ($\sim 0.6V$)
and attain high energy
efficiency “EEMode”

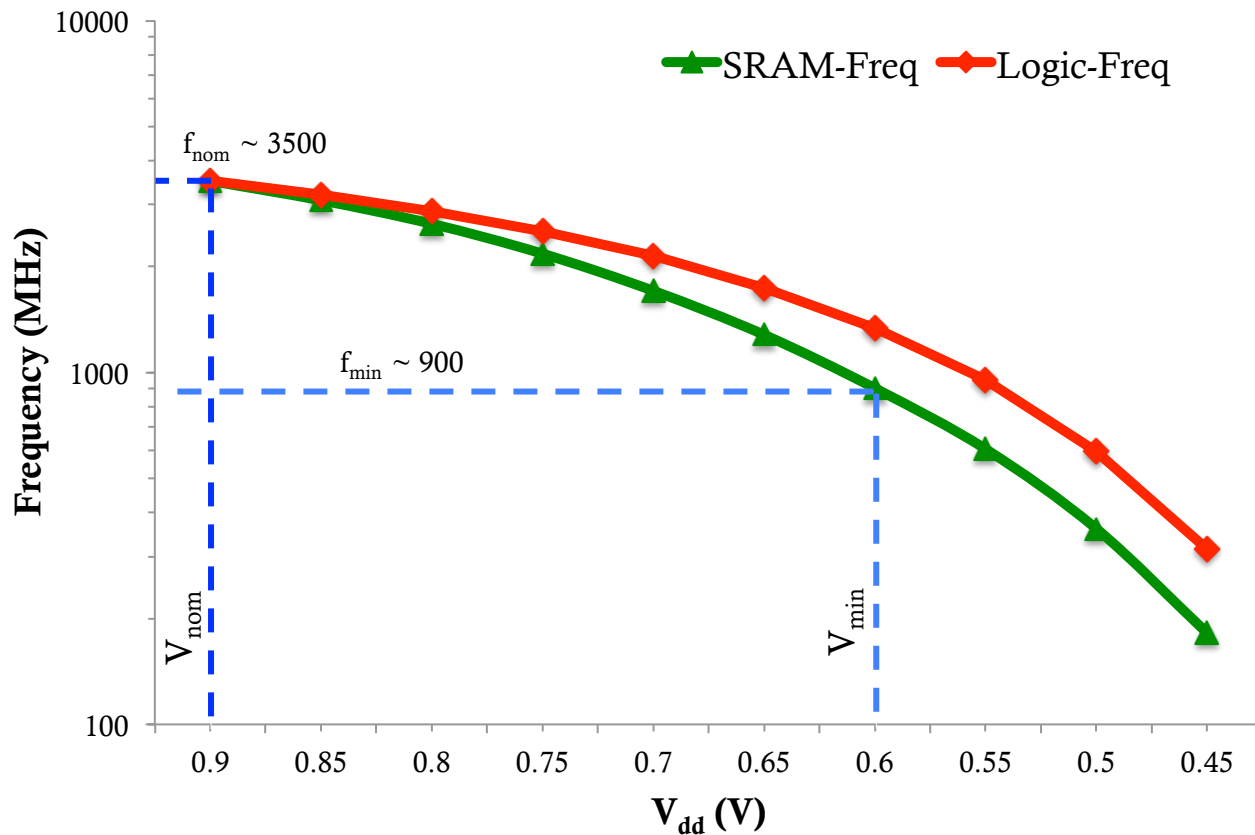


Deliver high performance
at nominal voltage ($\sim 0.9V$)
“HPMode”

Goal: Operate at very low V_{dd} when we have parallelism

Some Observations

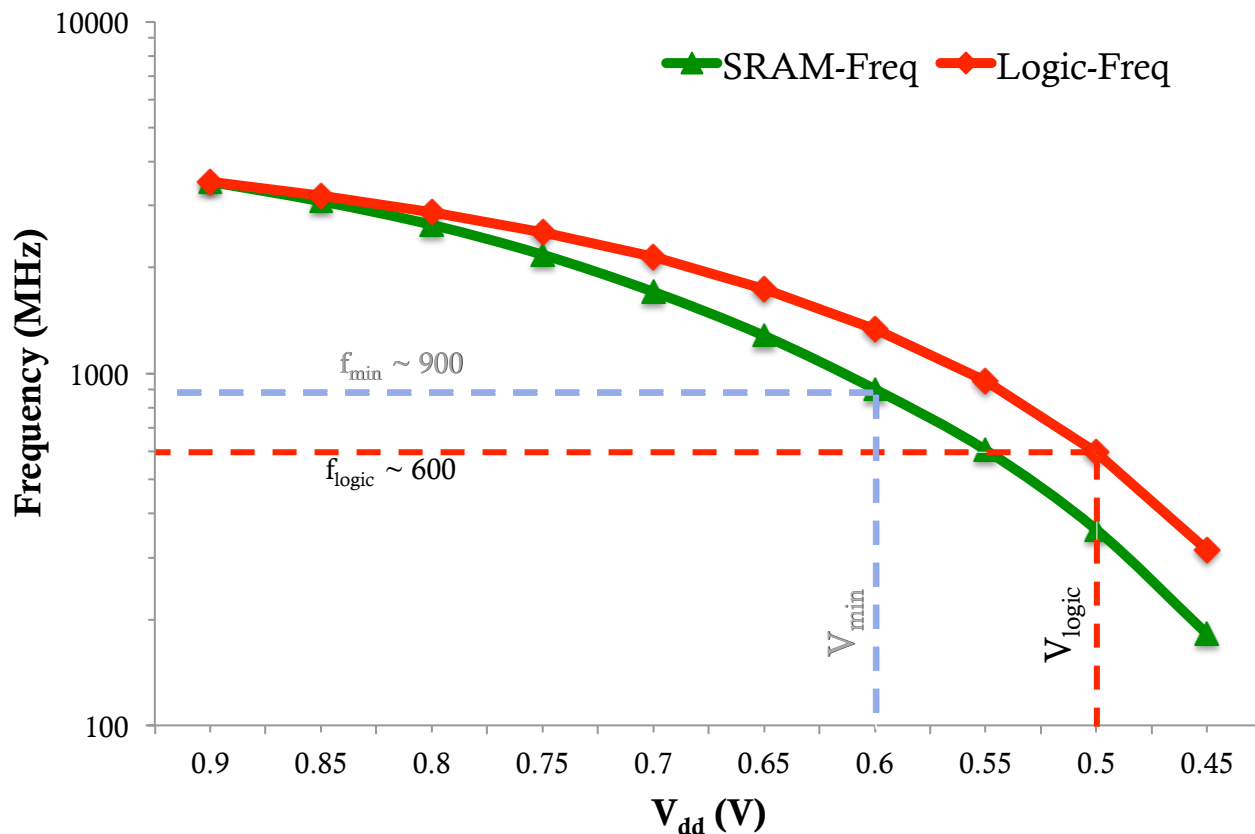
- SRAM and logic scale differently with V_{dd}
- Small increase in V_{dd} \rightarrow large reduction in delay



ScalCore Idea (I)

[Gopireddy HPCA'16]

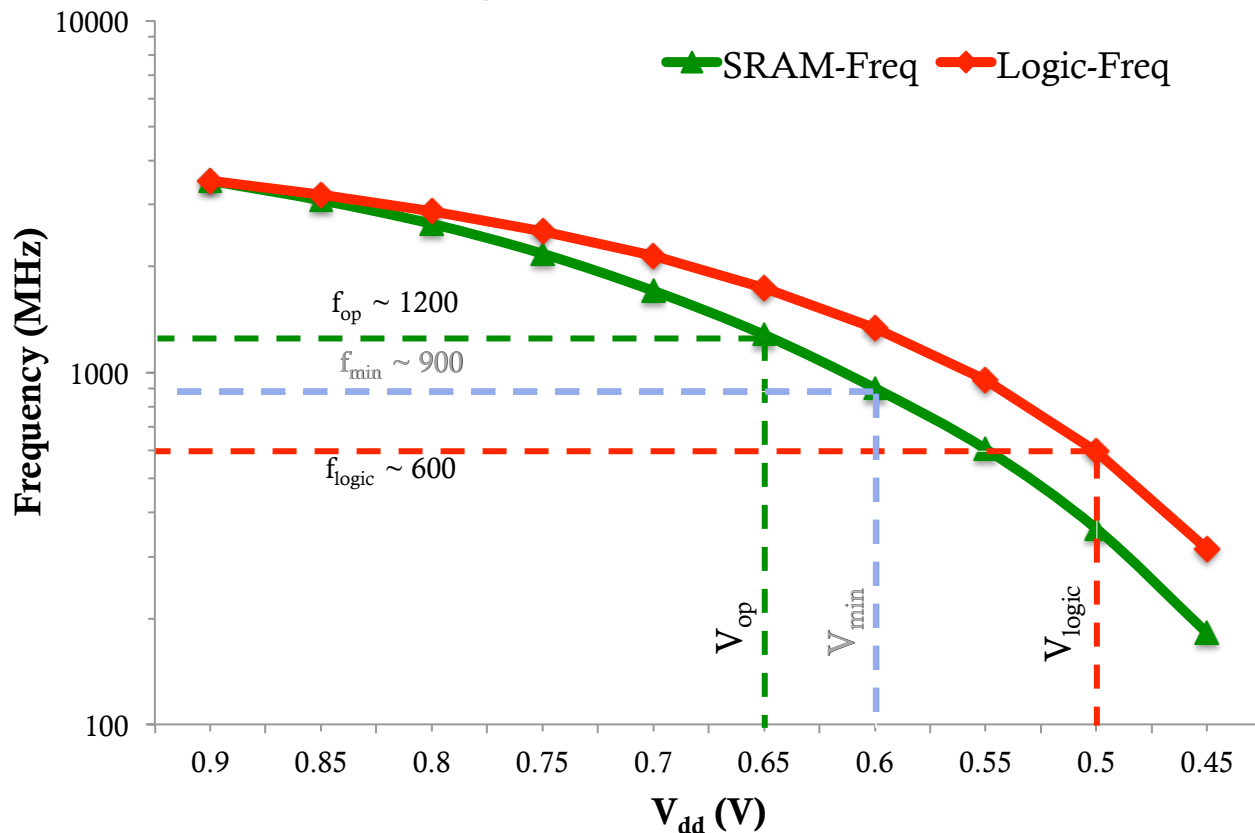
- Decouple the V_{dd} of logic and storage structures in the pipeline
 - Can reduce the V_{dd} of logic more → higher energy efficiency



ScalCore Idea (II)

[Gopireddy HPCA'16]

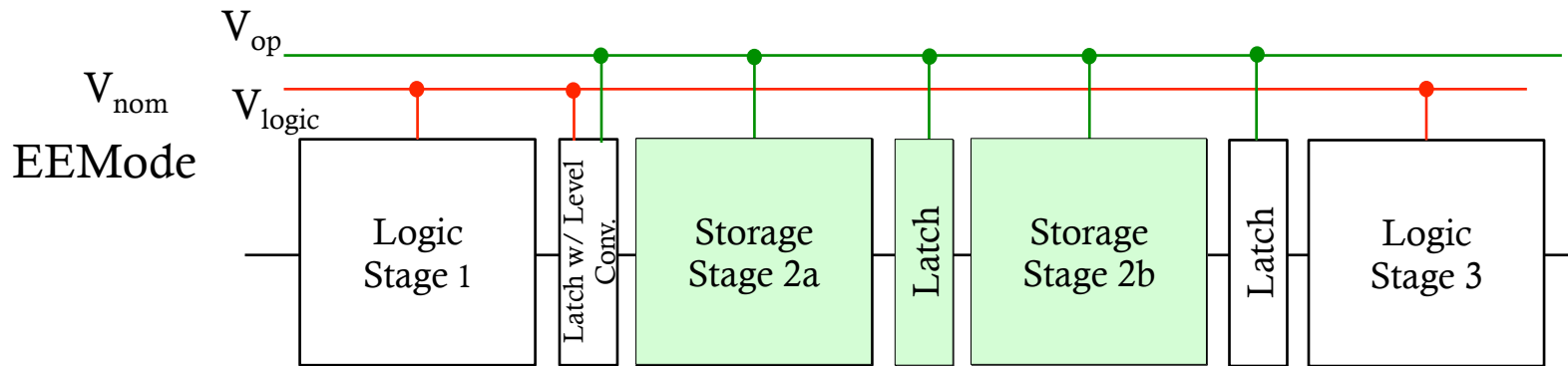
- Raise V_{dd} of storage structures a little: faster at low E cost
 - Reconfigure the pipeline to leverage the faster storage structures and improve IPC



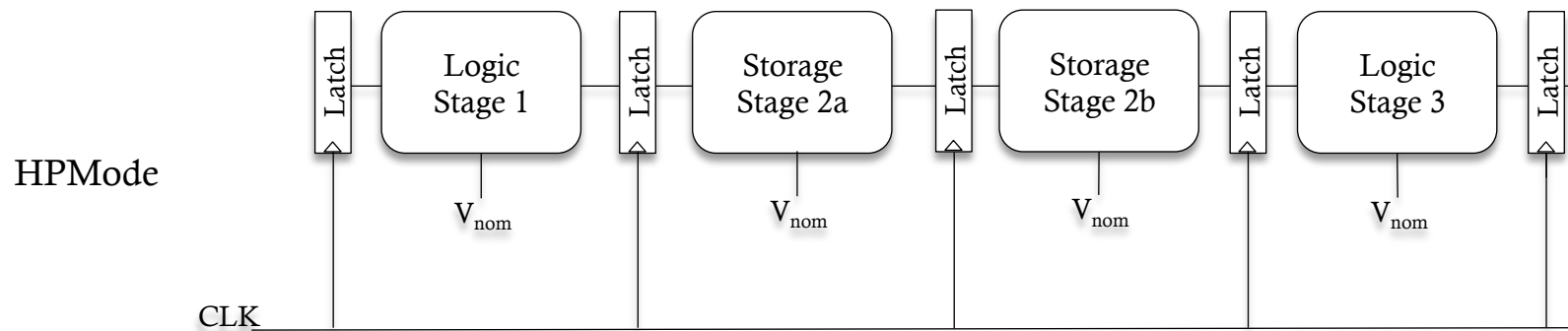
How Does it Work?

- At nominal, high-performance conditions (HPMode):
 - Conventional processor
- When energy efficiency matters (EEMode):
 - Decouple V_{dd} for storage and logic stages in the pipeline
 - Storage stages ~2x faster than logic stages
 - Reconfigure pipeline 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)

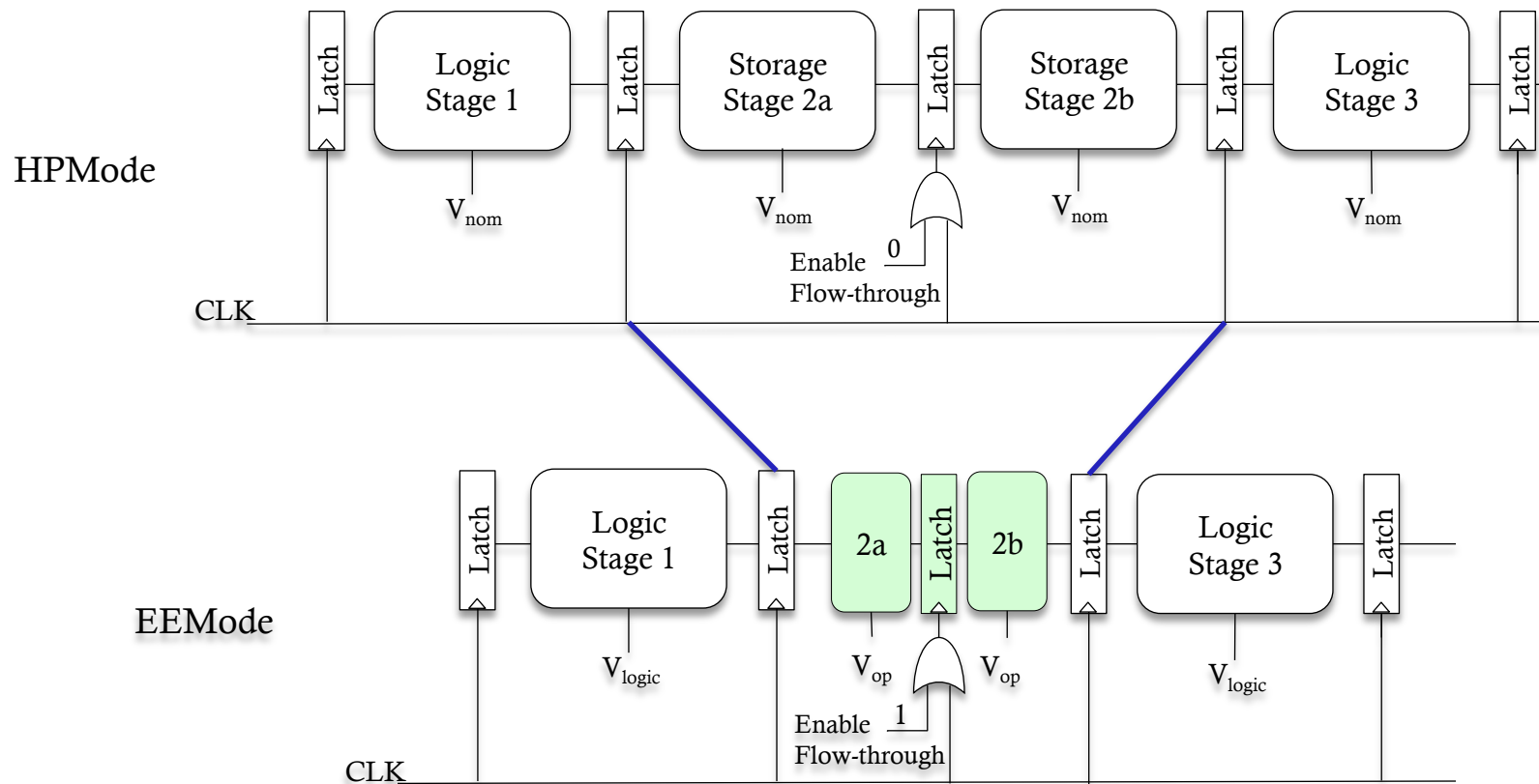
ScalCore: a Core for Voltage Scalability



Fusing Two Pipeline Stages into One



Fusing Two Pipeline Stages into One



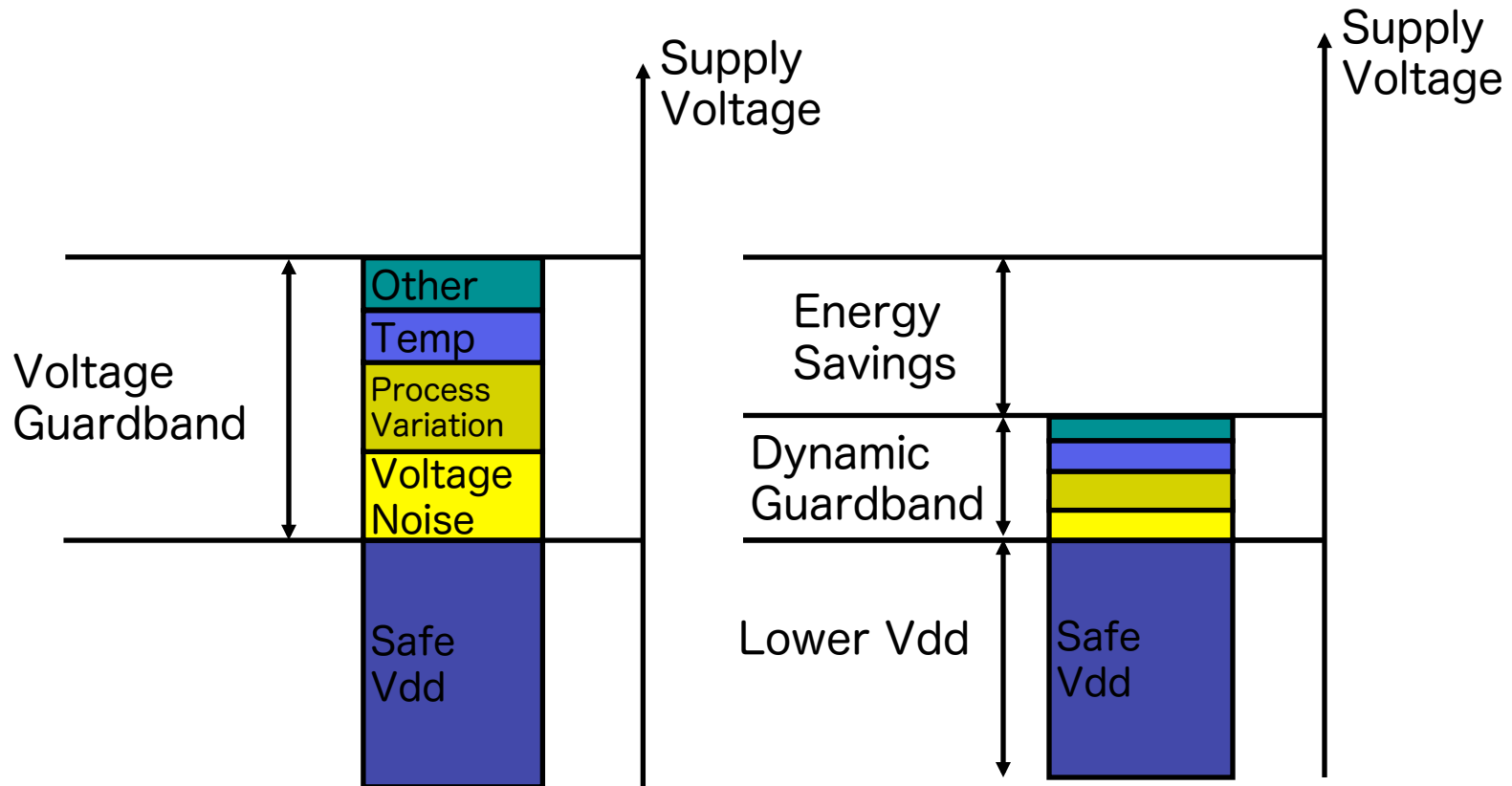
ScalCore Summary

- Highly energy-efficient when needed (parallel sections):
 - Vdd of logic stages very low
 - Reconfigured to fuse stages to increase IPC
- High performance at nominal conditions (serial sections):
 - Unmodified pipeline

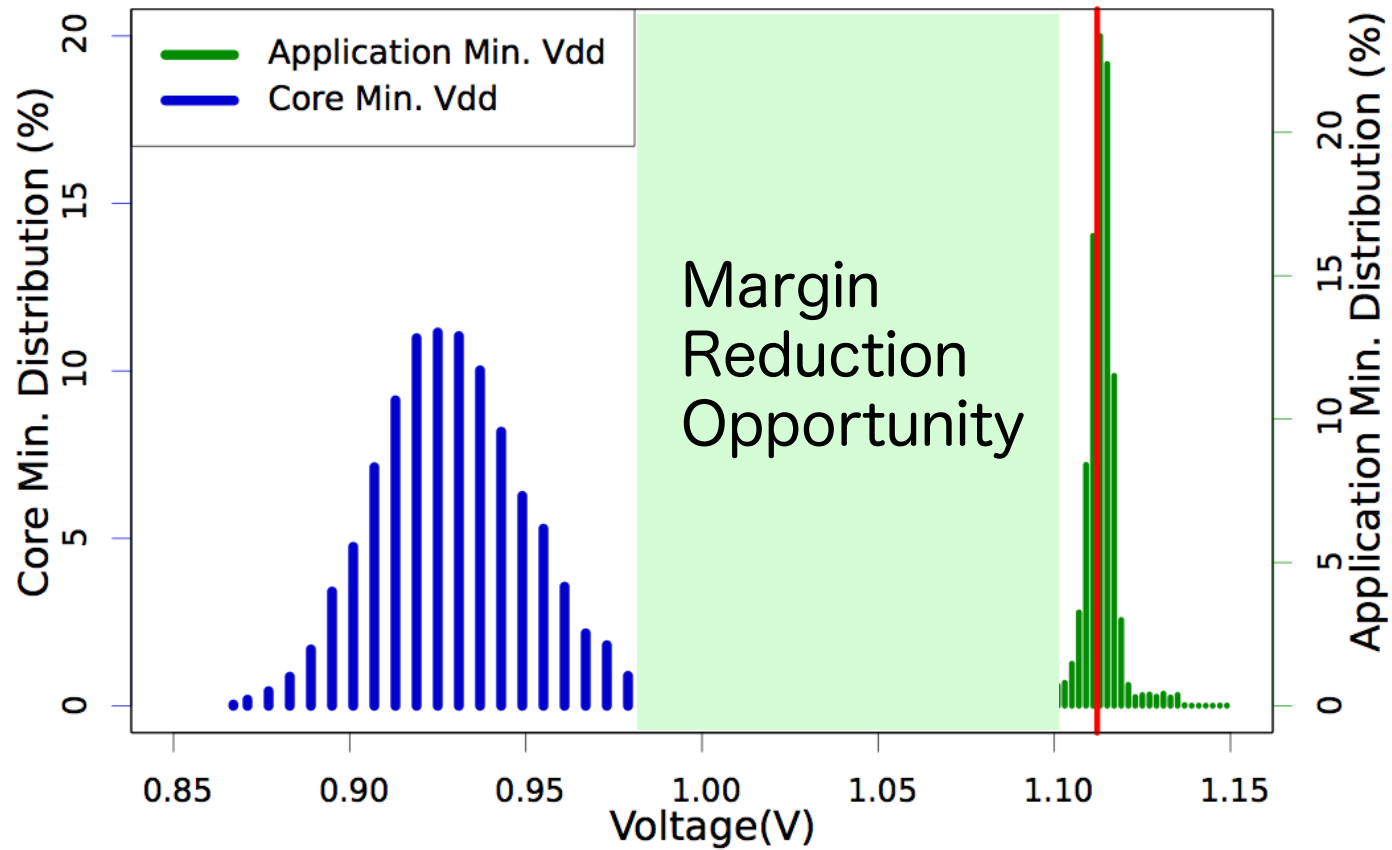
Attaining Very High Energy Efficiency

- Voltage-scalable cores
- **Dynamic voltage speculation**
- Pervasive power gating
- Control-theoretic controllers

Risky Ways to Reduce V_{dd}

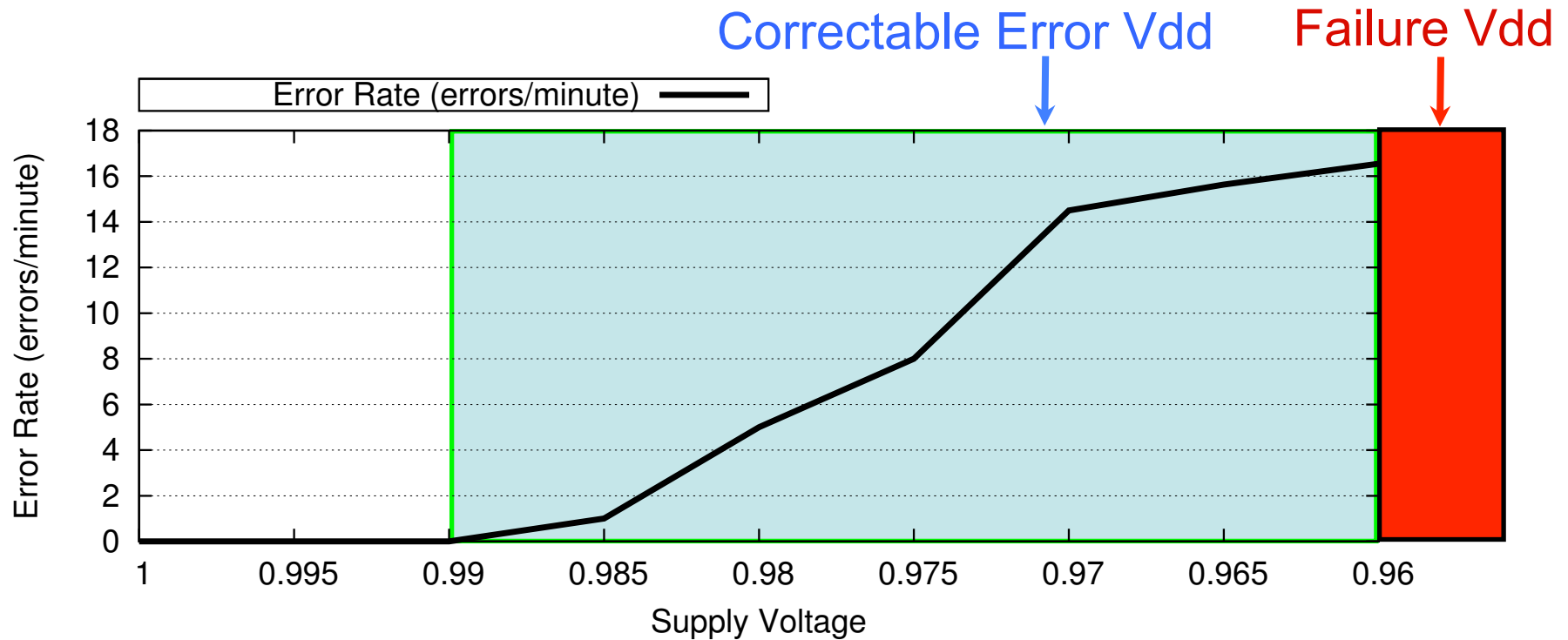


Reducing the Voltage of Cores



How Much Can We Reduce the Vdd?

[Bacha ISCA'13]



Observation: Correctable errors always triggered before uncorrectable ones, while running a stress test workload.

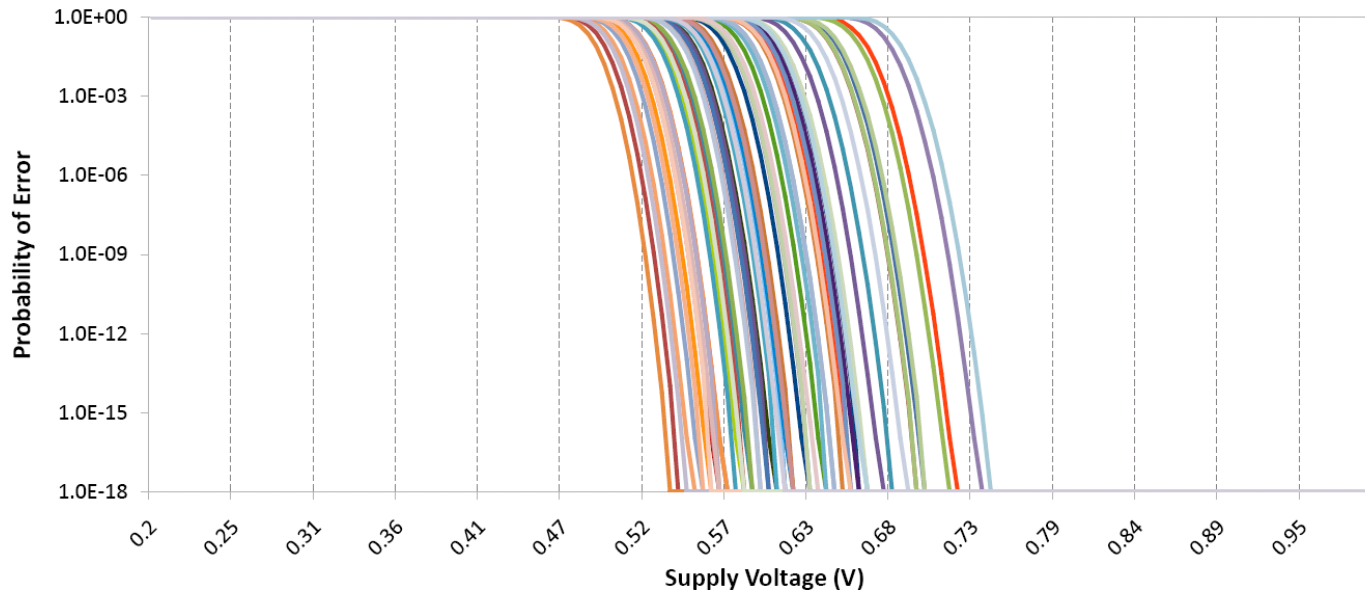
Reducing the Voltage of On-Chip Network

[Ansari HPCA'14]

- Networks typically have error detection capabilities
- Networks connect slow and fast parts of the chip (due to process variation)
- Propose:
 - Dynamically reduce Vdd of different parts of the network
 - Detect and handle errors

Error Rate as Function of V_{dd}

- On-chip network with many routers
- Error rate per router as we change V_{dd}

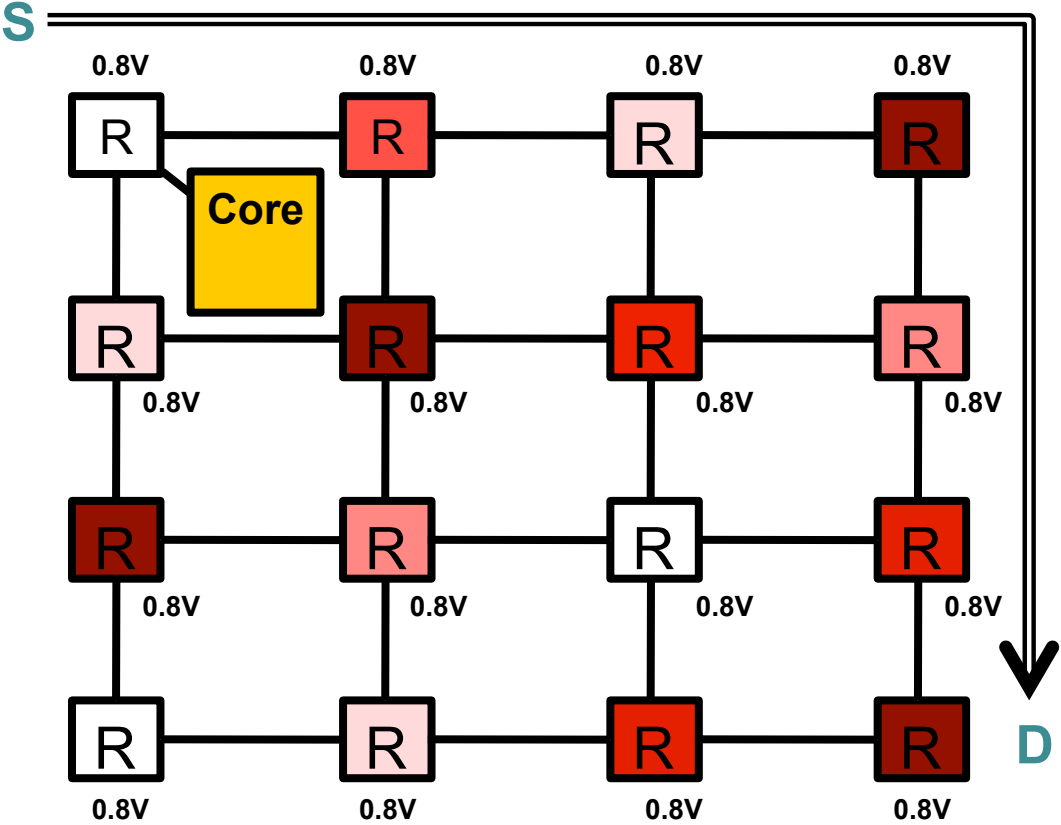


- Process variation has a major impact on the routers

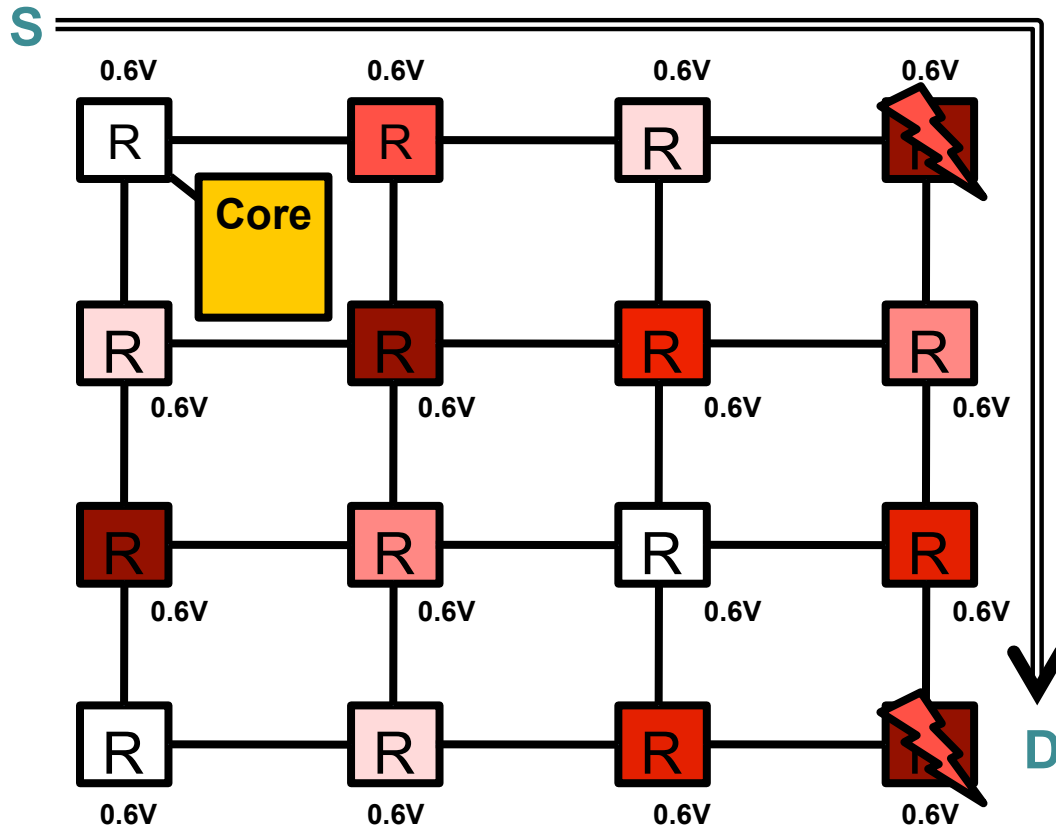
Leveraging the Error Handling of Networks

- Reduce V_{dd} of clusters of routers based on their tolerance
 - Continuously monitor for errors (and handle them)
 - Dynamically adapt V_{dd} of each cluster of routers based on errors
- Highly energy efficient
 - Remove V_{dd} margins added for variation

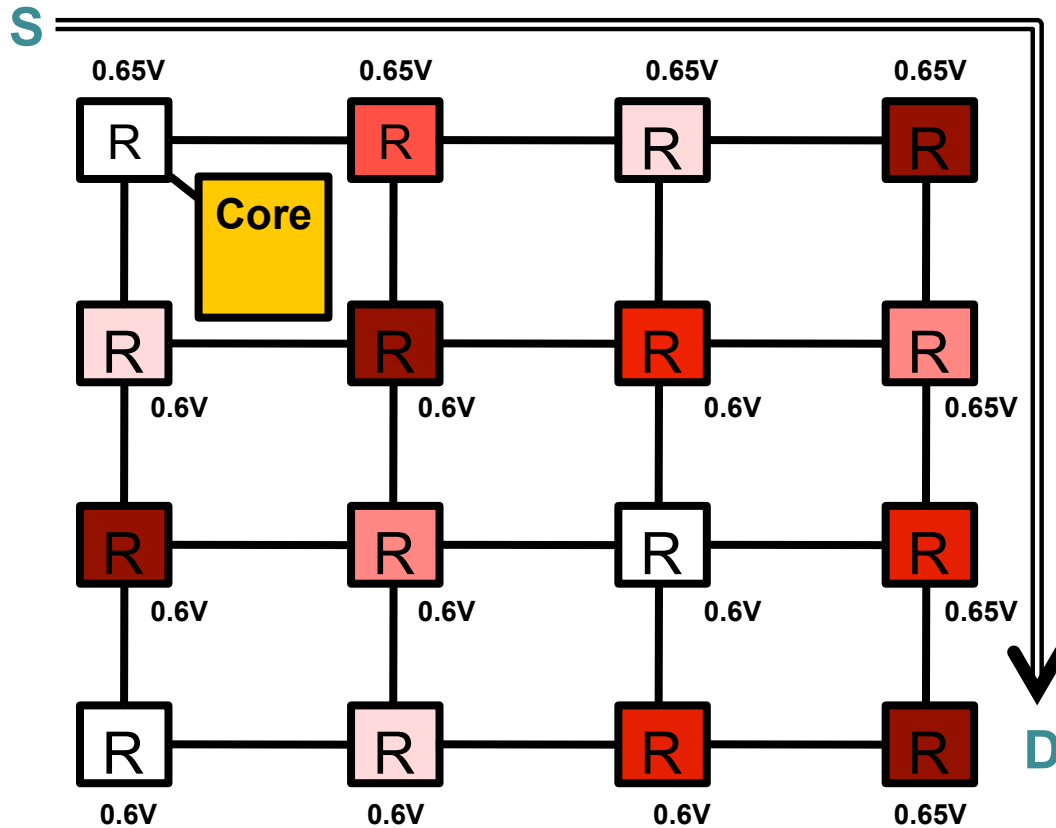
Scheme Operation (Initial)



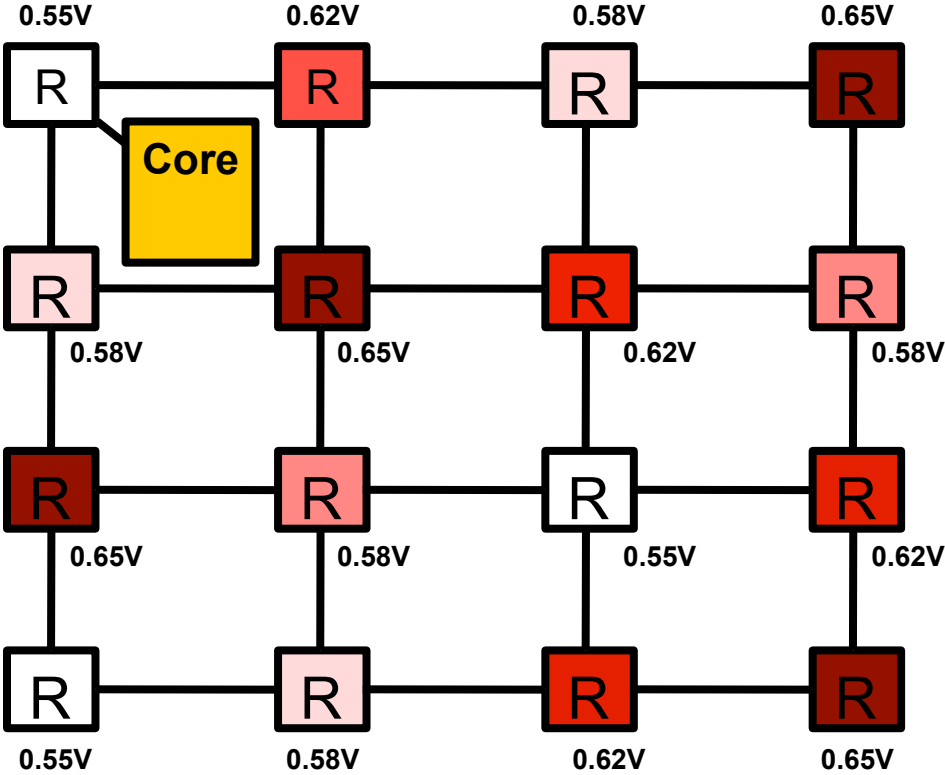
Scheme Operation (Lowering Voltage)



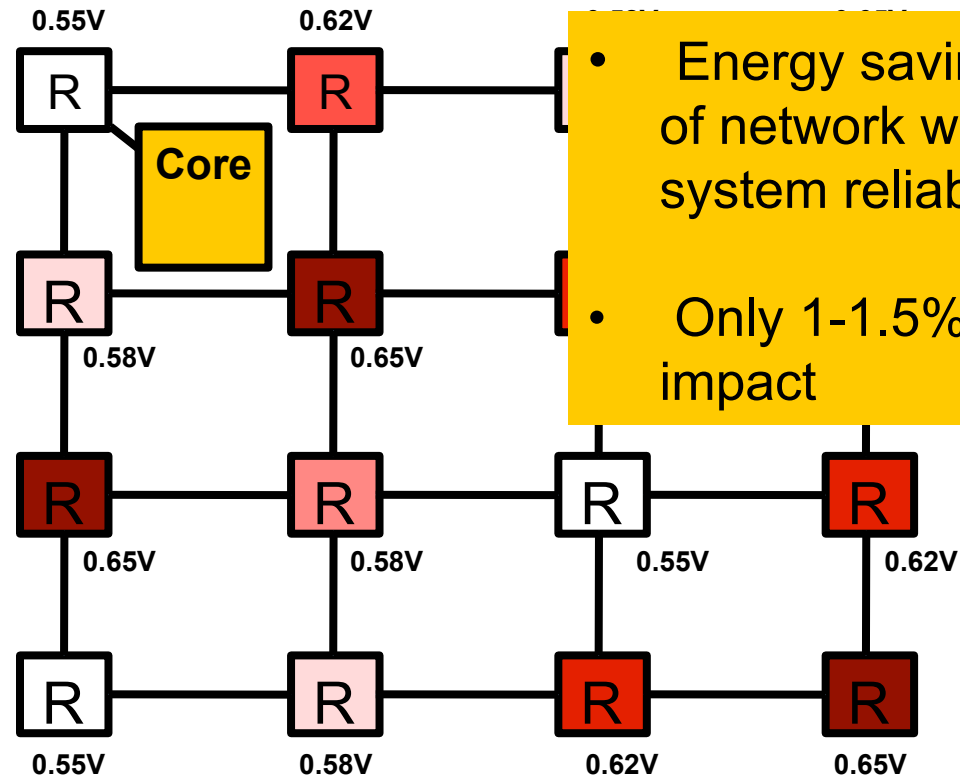
Scheme Operation (V_{dd} Tuning on a Path)



Scheme Operation (Convergence)



Scheme Operation (Convergence)



Attaining Very High Energy Efficiency

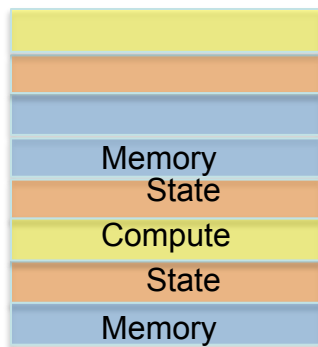
- Voltage-scalable cores
- Dynamic voltage speculation
- **Pervasive power gating**
- Control-theoretic controllers

Power-Gating: A Real Requirement

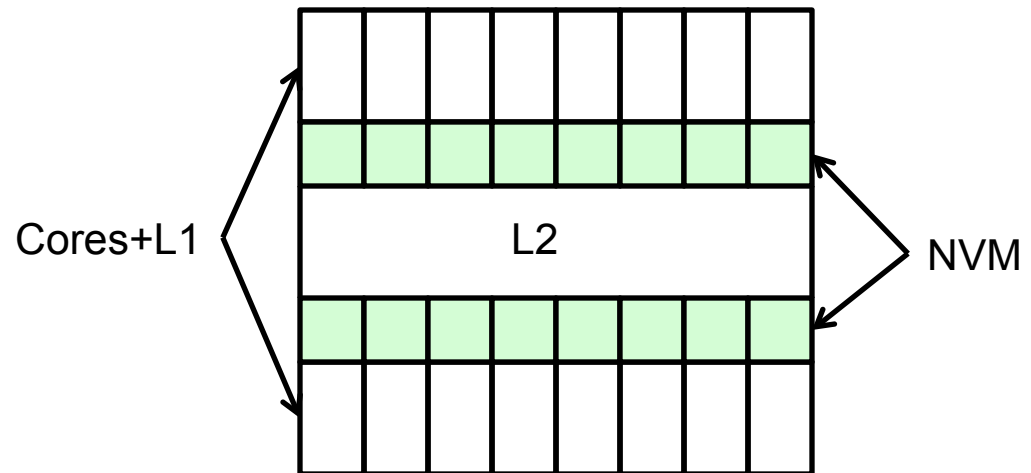
- Components not in use **need to be power-gated**
 - OS/software can do it sometimes, but has overhead
 - Many short idle periods; need **HW-based power gating**
 - Example: Last-level cache miss
- When we power-gate a structure, we lose its state
- Propose **micro-checkpoint** the pipeline: fast restoration of state

Use Non-Volatile Memory for Micro-Checkpointing

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



Monolithic integration

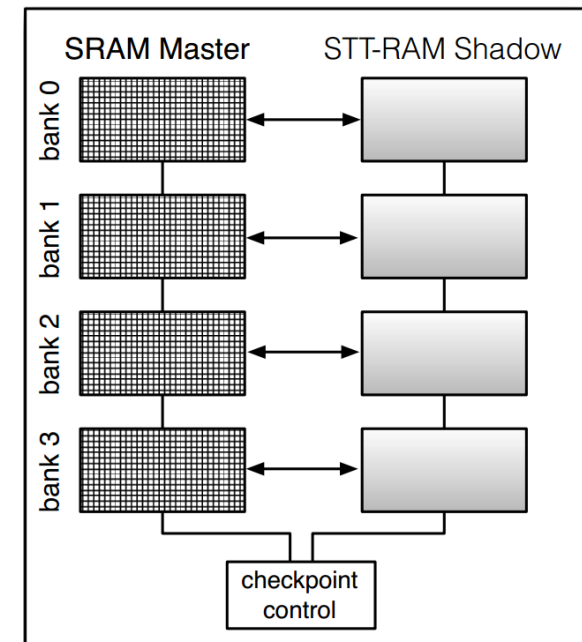


Same die integration

Shadow Latency-Sensitive Structures

[Pan ICCD'14]

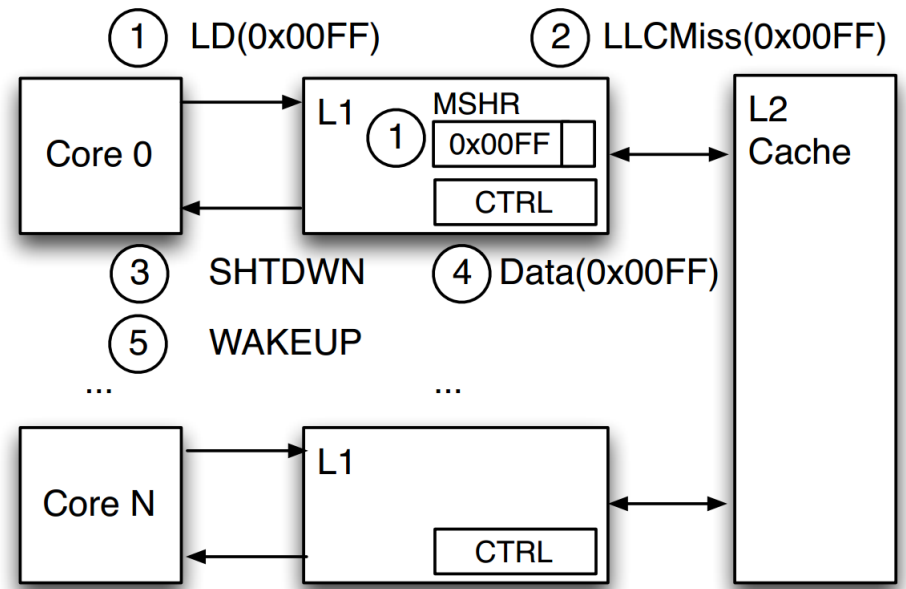
- Write-latency sensitive units:
 - Reg file, Inst window, ROB, Ld/st queue, pipeline regs
 - Implemented in SRAM + shadow in STTRAM
- Hybrid SRAM/STTRAM
 - SRAM for primary storage
 - STTRAM shadow of identical size
 - Data moved to shadow lazily



Hardware Implementation

- Checkpointing and wakeup of cores managed by L1 cache controller
- Checkpoint/wakeup sequence:

1. Ld issued, missed in L2
2. Sleep signal sent to core
3. Push data into shadow + power gate
4. Missing data returns
5. Wakeup signal sent to core
6. Reload data from shadow + wakeup



Attaining Very High Energy Efficiency

- Voltage-scalable cores
- Dynamic voltage speculation
- Pervasive power gating
- **Control-theoretic controllers**

The Need for Optimal Control

- Extreme scale manycores **need effective controllers**
 - Power, energy, temperature, utilization...
- Current approaches for architectural control and tuning
 - Heuristics
 - Machine learning
 - **Control theory**

Heuristics


 Lightweight

 Popular with architects

 No guarantees

 No formal methodology

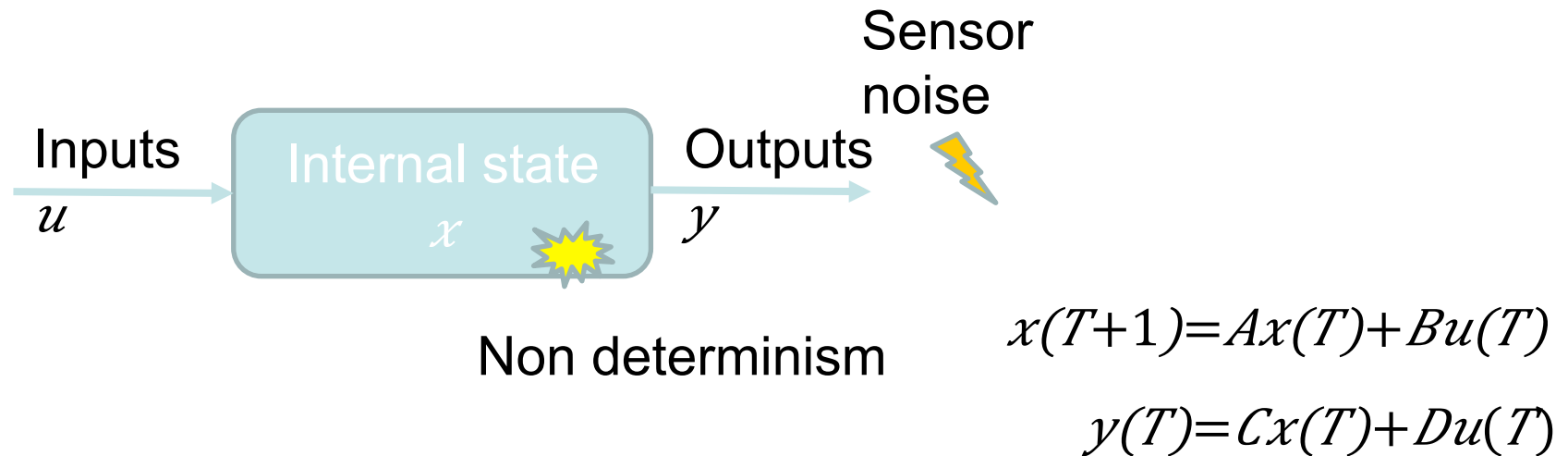
 Hard to add learning

 Prone to errors

 Hard to deal with multiple inputs and/or outputs

Controlling a System with Control Theory

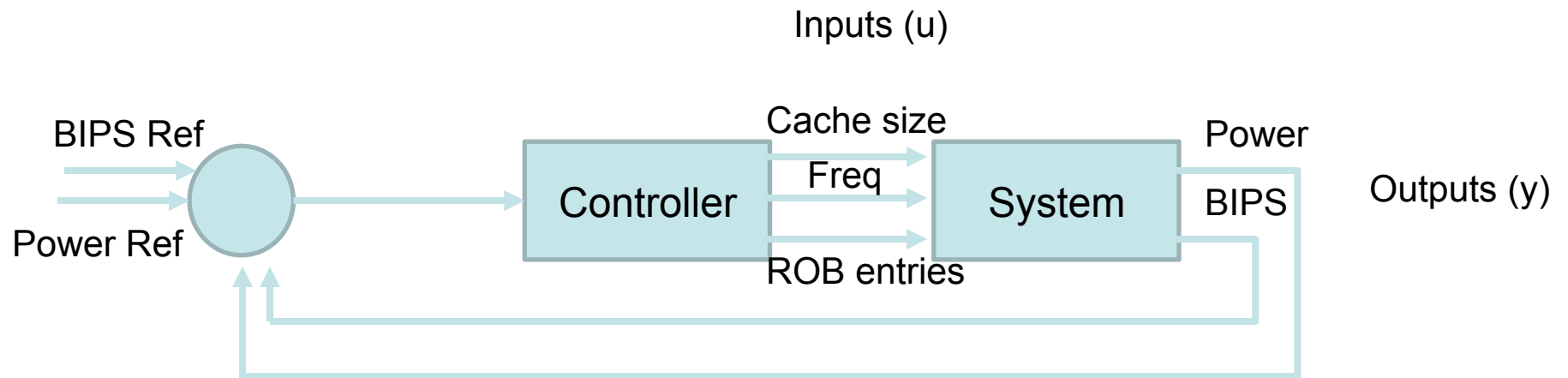
[Pothukuchi ISCA'16]



- The model is $\{A, B, C, D\}$ + Unpredictability matrices
 - Obtained from analytical formulas or experimental characterization
- Want **MIMO control** (Multiple Inputs and Multiple Outputs)

MIMO Control

- Actuate on multiple inputs: cache size, frequency, #ROB entries
- Control multiple outputs: performance (BIPS), power



Control Theory

- 😊 Feedback loop: runtime adaptation to conditions not seen during training
- 😊 Guarantees: Convergence, Stability, Optimality
- 😊 Easy to add/remove a new input

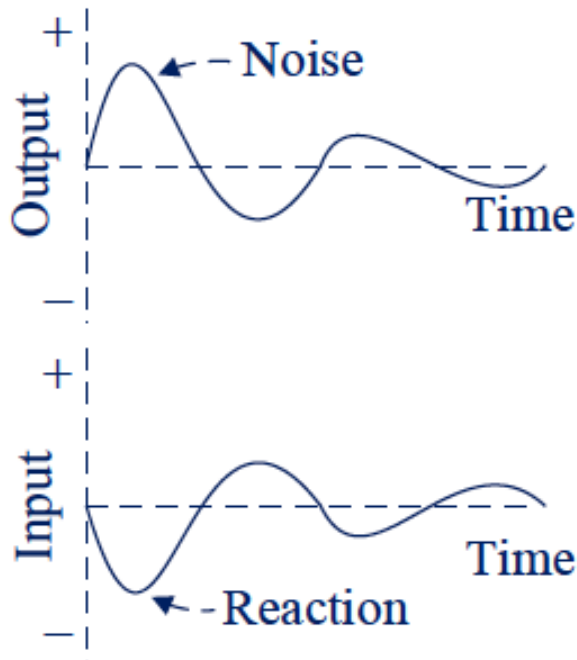
- 😞 Hard to obtain model
- 😞 Specifying the target values of outputs is not obvious (Power, performance)

MIMO Controller Details

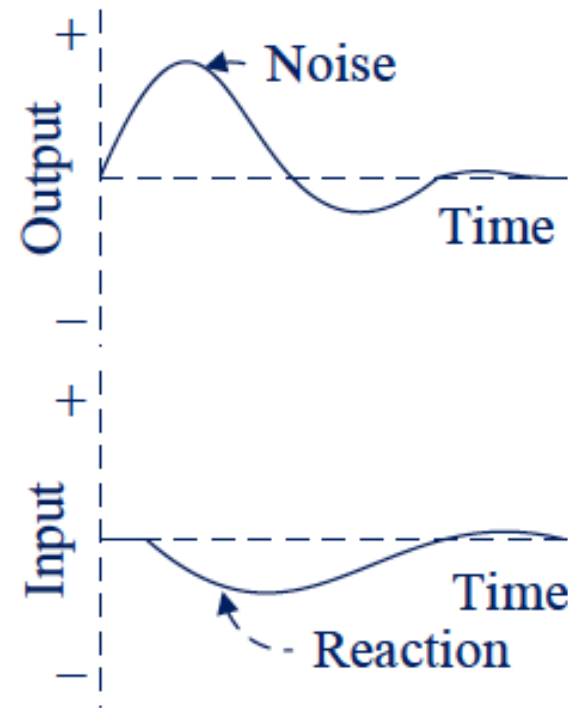
- Each input and each output has a cost (or weight)
 - Cost of an input: How hard it is to change it from its current value
 - Cost of an output: Cost of not meeting the target of the output
- System will try to minimize the changes to costly inputs/outputs

MIMO Controller Details

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



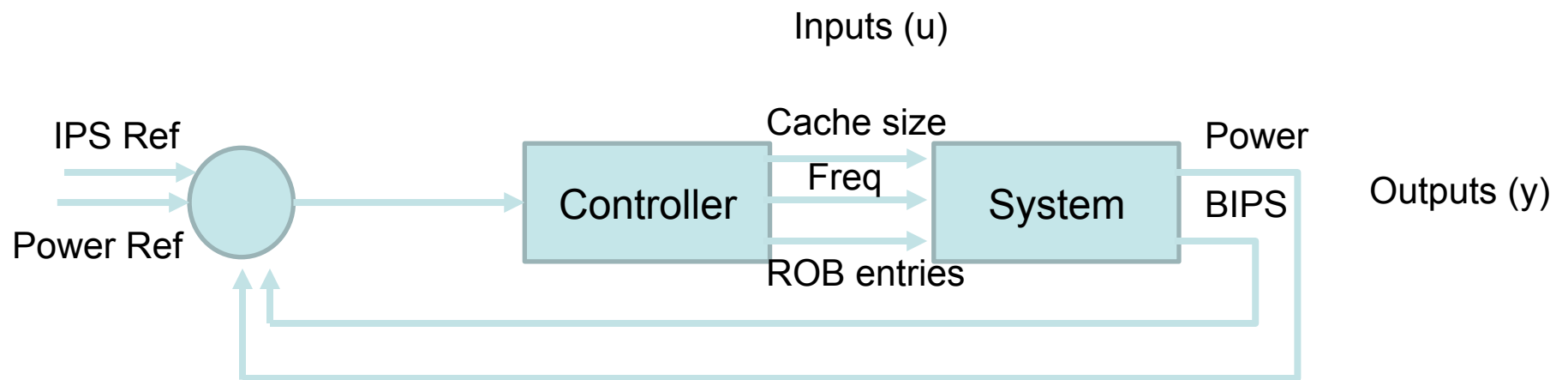
Input weights \ll output weights:
Ripply system



Input weights \gg output weights:
System with inertia

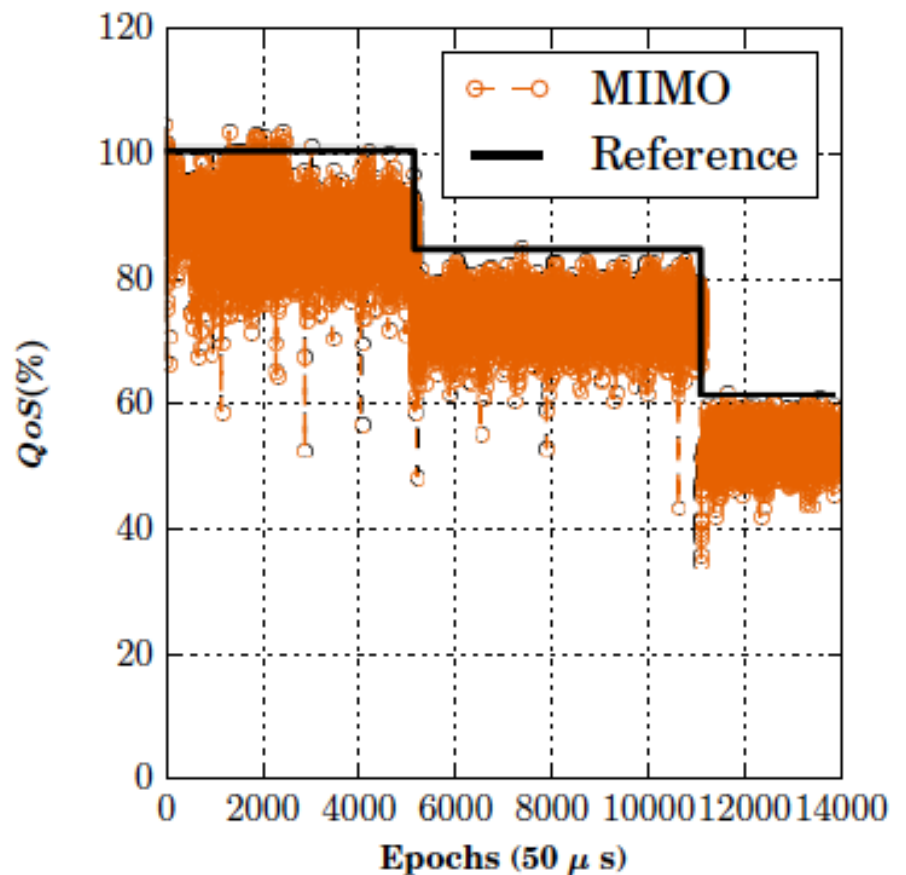
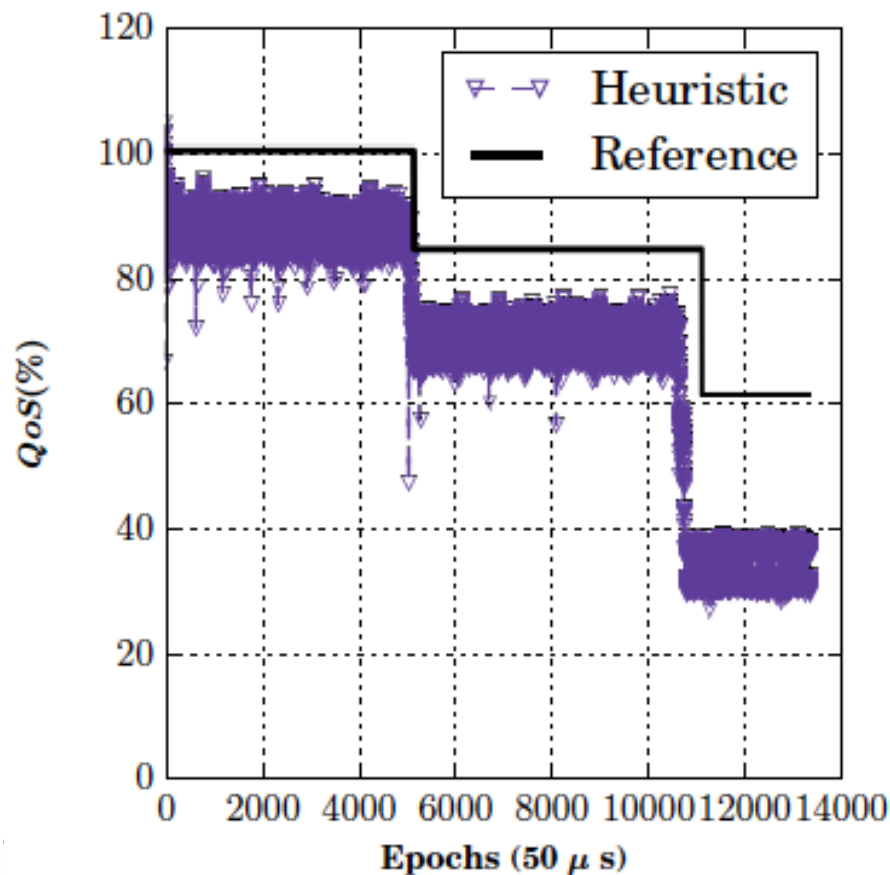
Uses of the Controller (I)

- Set outputs to target values:
 - Performance ($BIPS_0$) and Power (P_0)



Uses of the Controller (II)

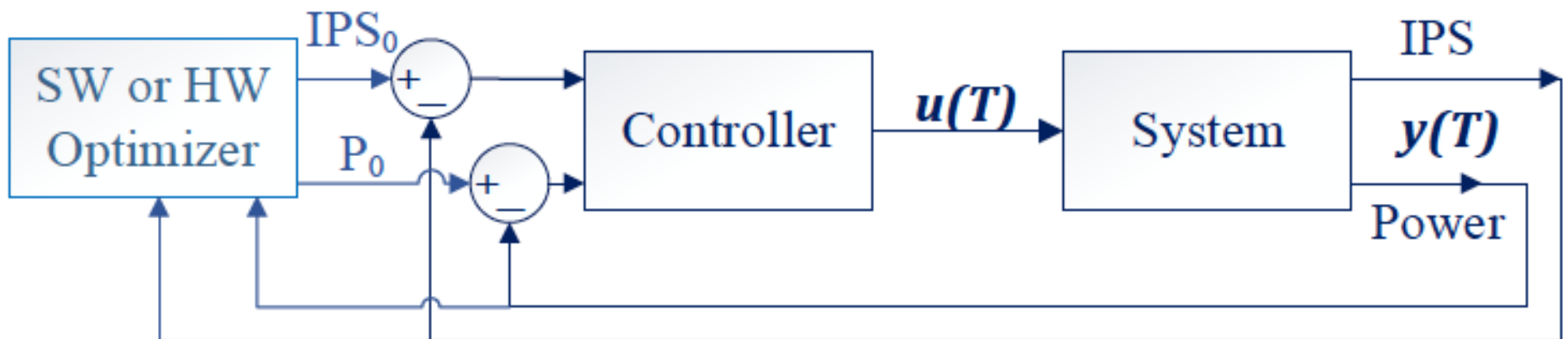
- Set outputs to varying target values:
 - Changing the quality of service (QoS) as the battery is depleted in a mobile device



Uses of the Controller (III)

- Optimize a combination of output measures:
 - Minimize (ExD)= maximize (IPS²/Power)

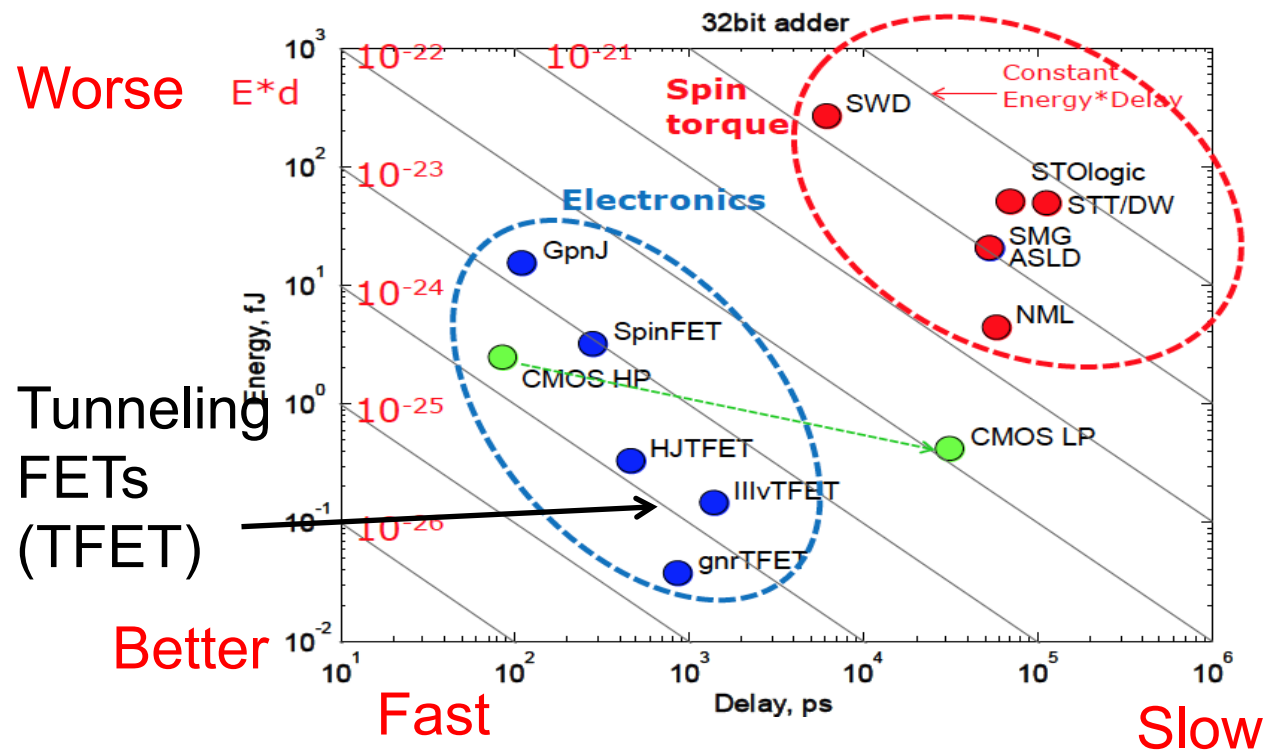
Propose: Optimizer that searches directly in the space of (IPS²,P)



Attaining Very High Energy Efficiency

- Voltage-scalable cores
- Dynamic voltage speculation
- Pervasive power gating
- Control-theoretic controllers

More Energy Efficiency? Need New Technologies



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

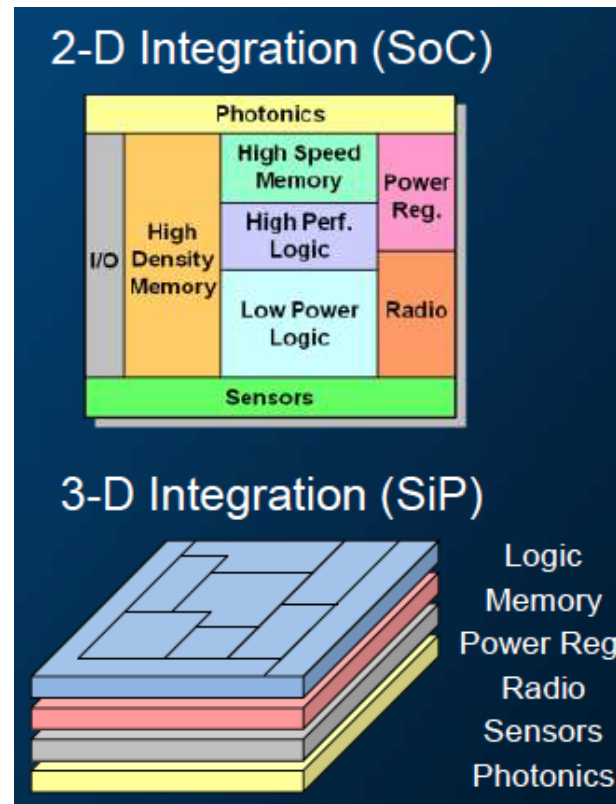
TFET Characteristics

- 😊 Can be fabricated on the same die as CMOS
- 😊 Consumes much less power (4—8x less)
- 😊 Scalable

- 😞 Not as fast as CMOS (2—4x slower)

What Will Happen?

- Heterogeneous architectures?



Conclusion

- Energy and power efficiency are the strongest constraints in future computer architectures
- There is no silver bullet (or perhaps it is V_{dd} reduction)
- Some principles:
 - Reduce voltage (safely or taking risks)
 - Turn-off if unused
 - Minimize waste

Toward Extreme-Scale Processor Chips

Josep Torrellas

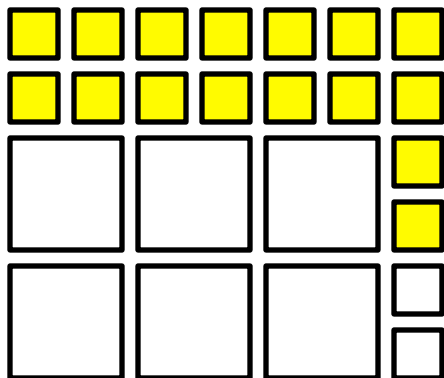
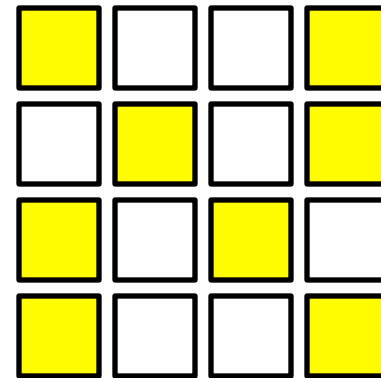
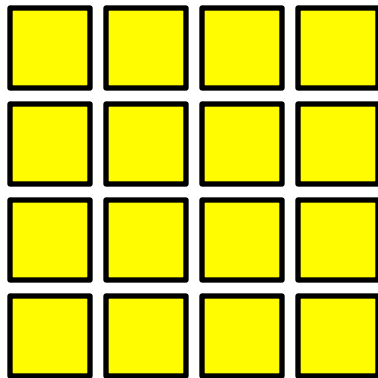
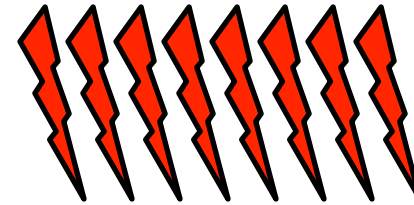
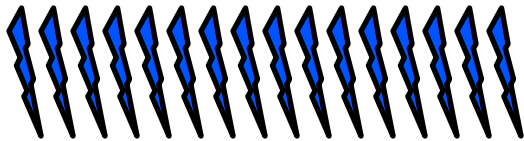
Department of Computer Science
University of Illinois at Urbana-Champaign
<http://iacoma.cs.uiuc.edu>

HiPC 2016

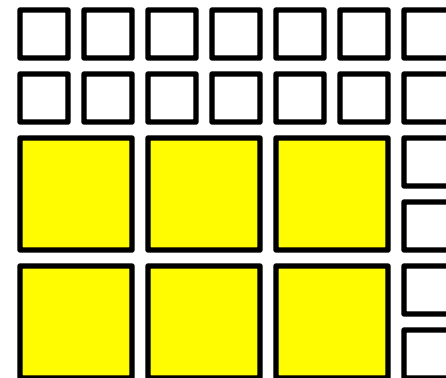
Hyderabad, India



Currently: Big/Little



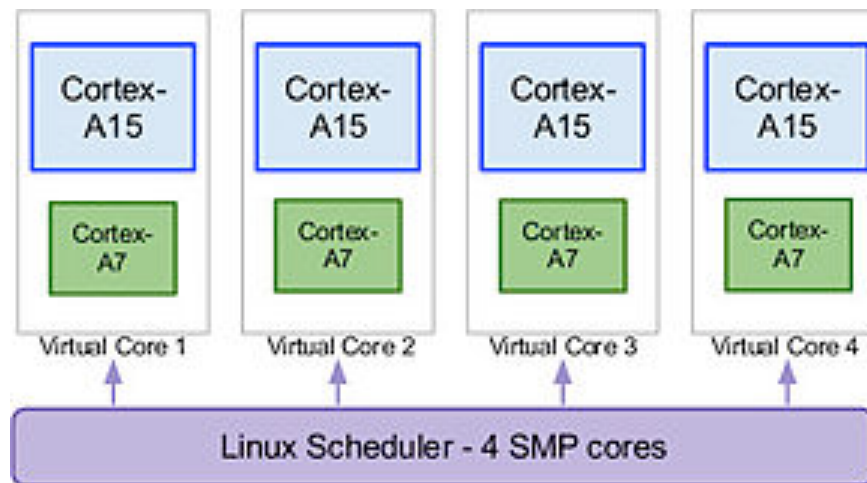
Little



Big

Big/Little Not Optimal

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



ARM System