

Helping Moore's Law: Architectural Techniques to Address Parameter Variation

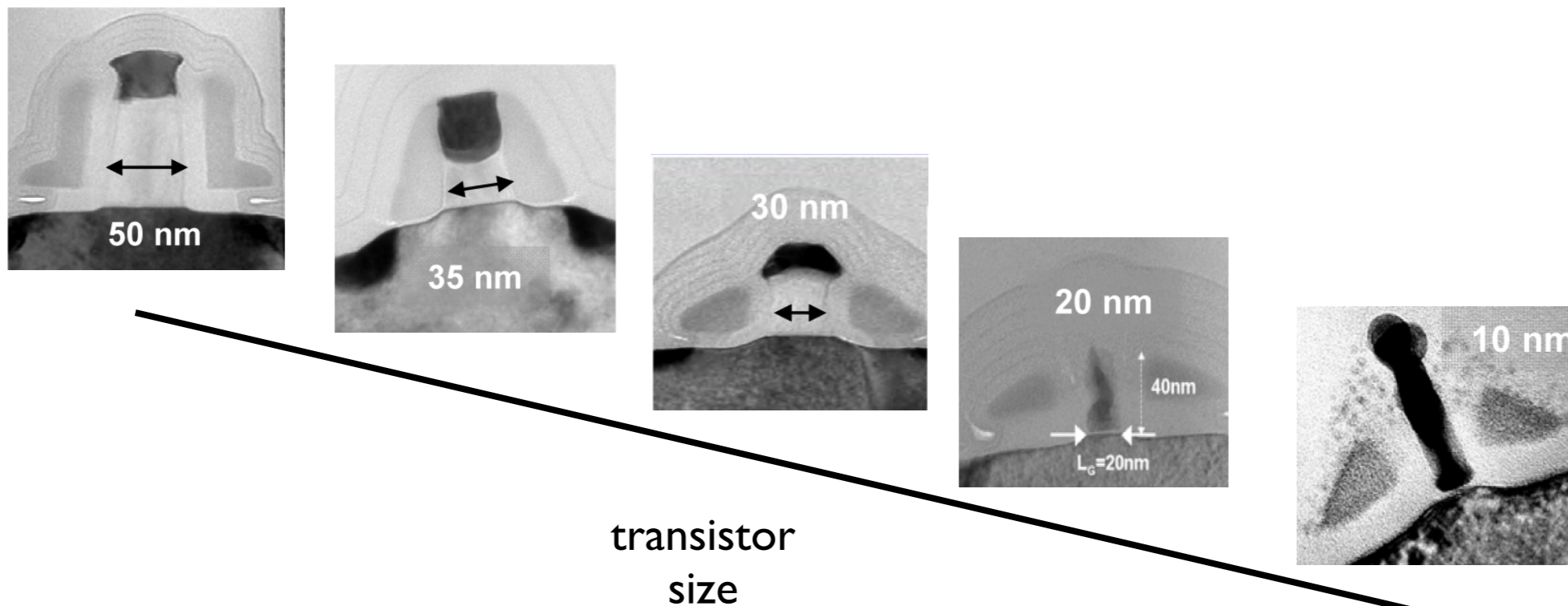
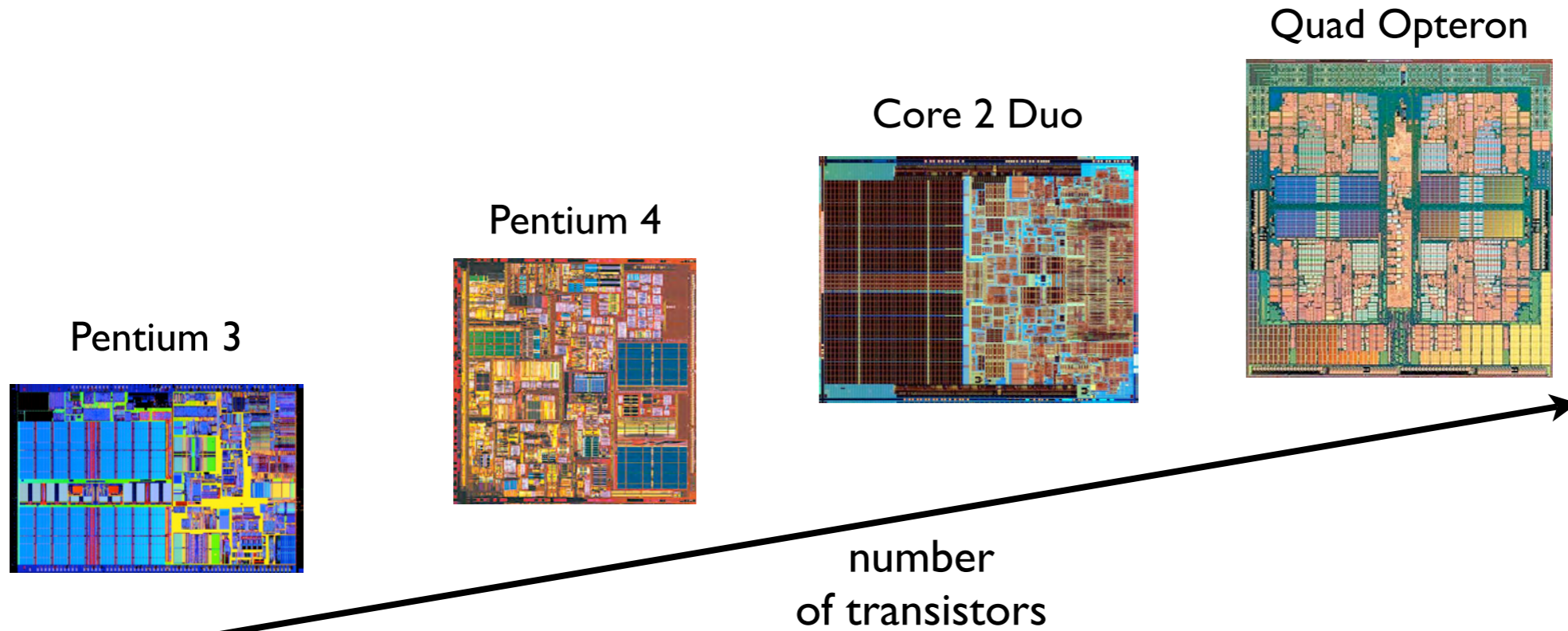
Radu Teodorescu

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





Technology scaling continues



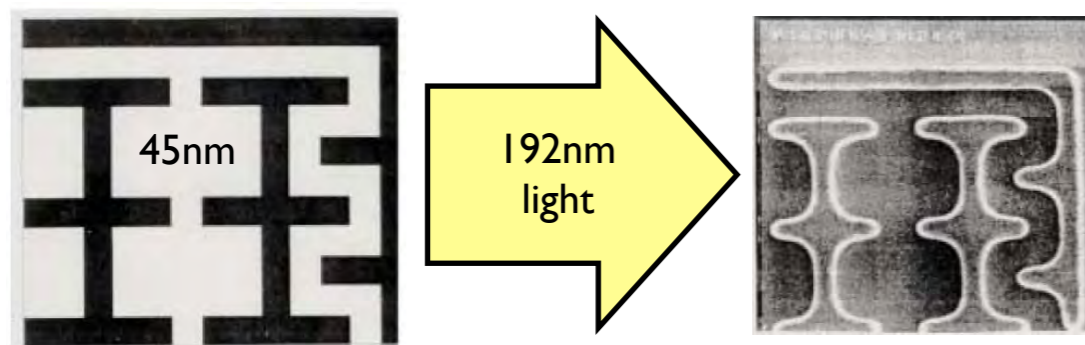


Challenges to scaling

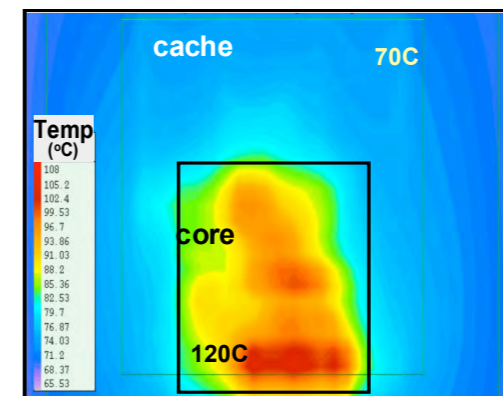
Manufacturing process

Environmental

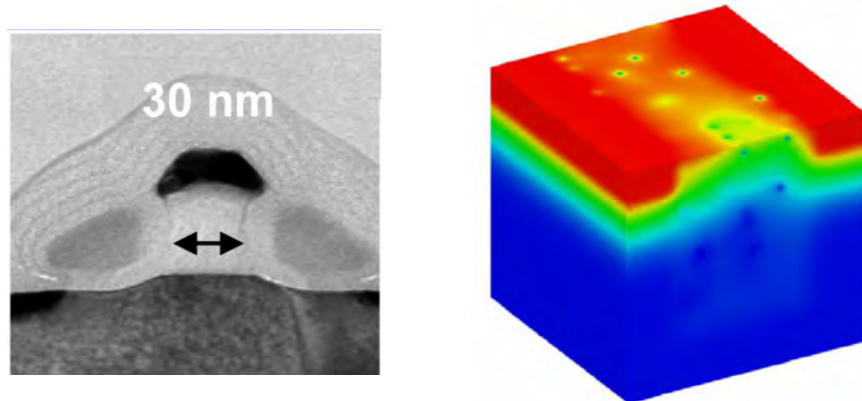
Sub-wavelength lithography



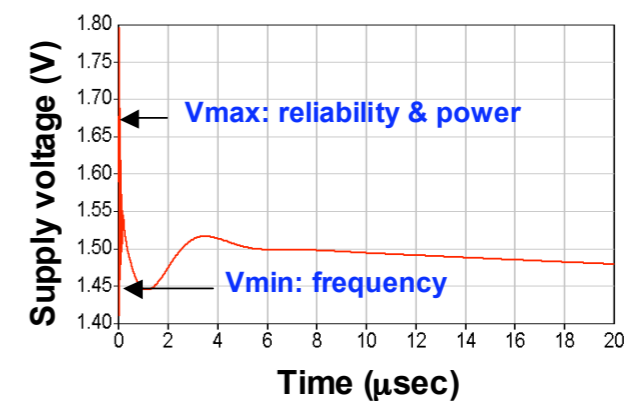
Temperature variation



Dopant density fluctuations

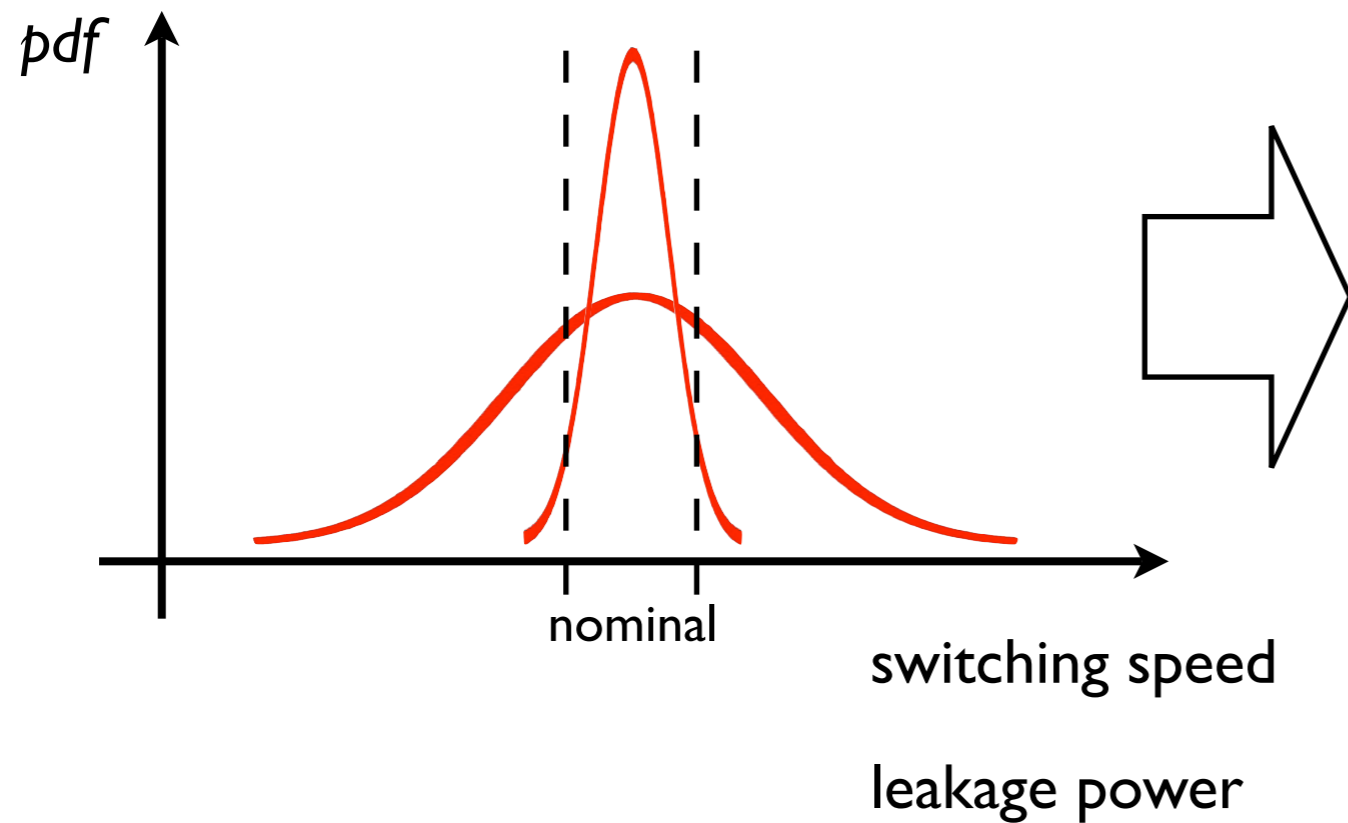


Supply voltage fluctuations



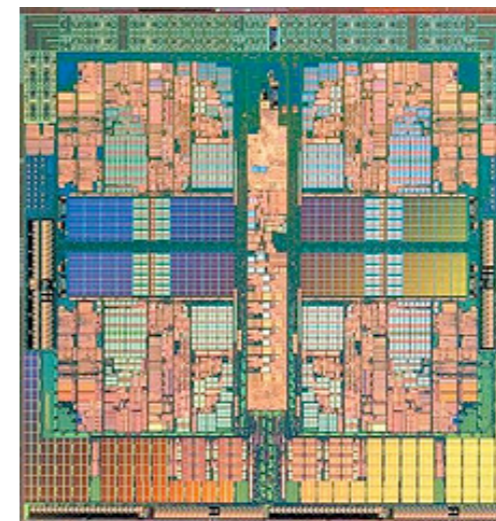


Variation in transistor parameters

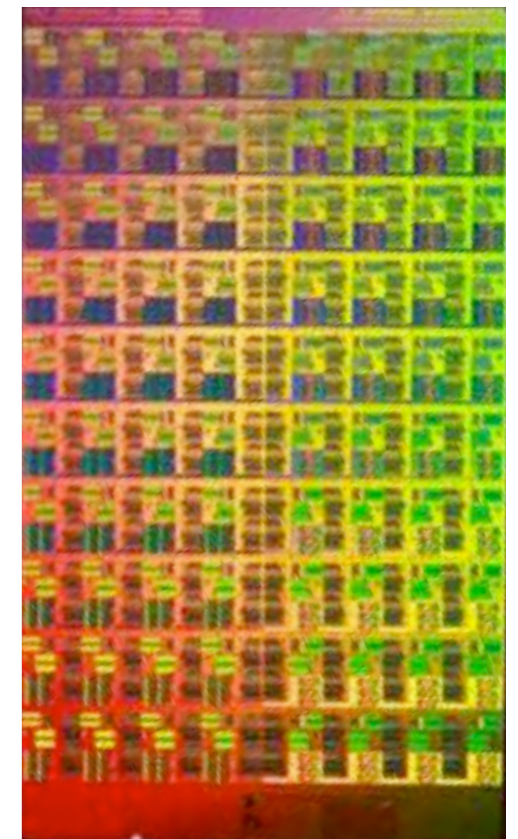


↓ Frequency
↓ Reliability

↑ Power



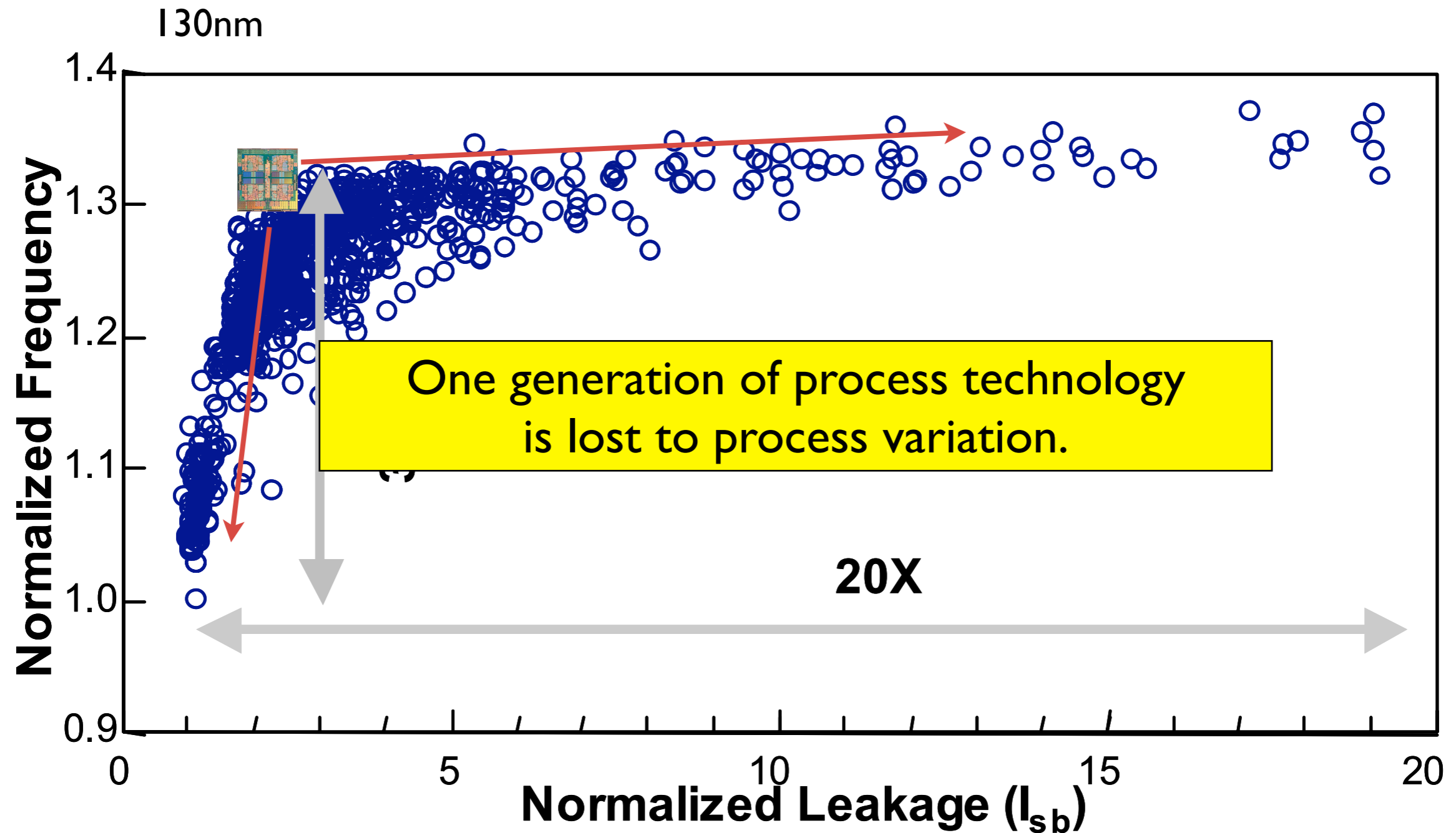
AMD Quad-core
Opteron



Intel 80-core
Polaris



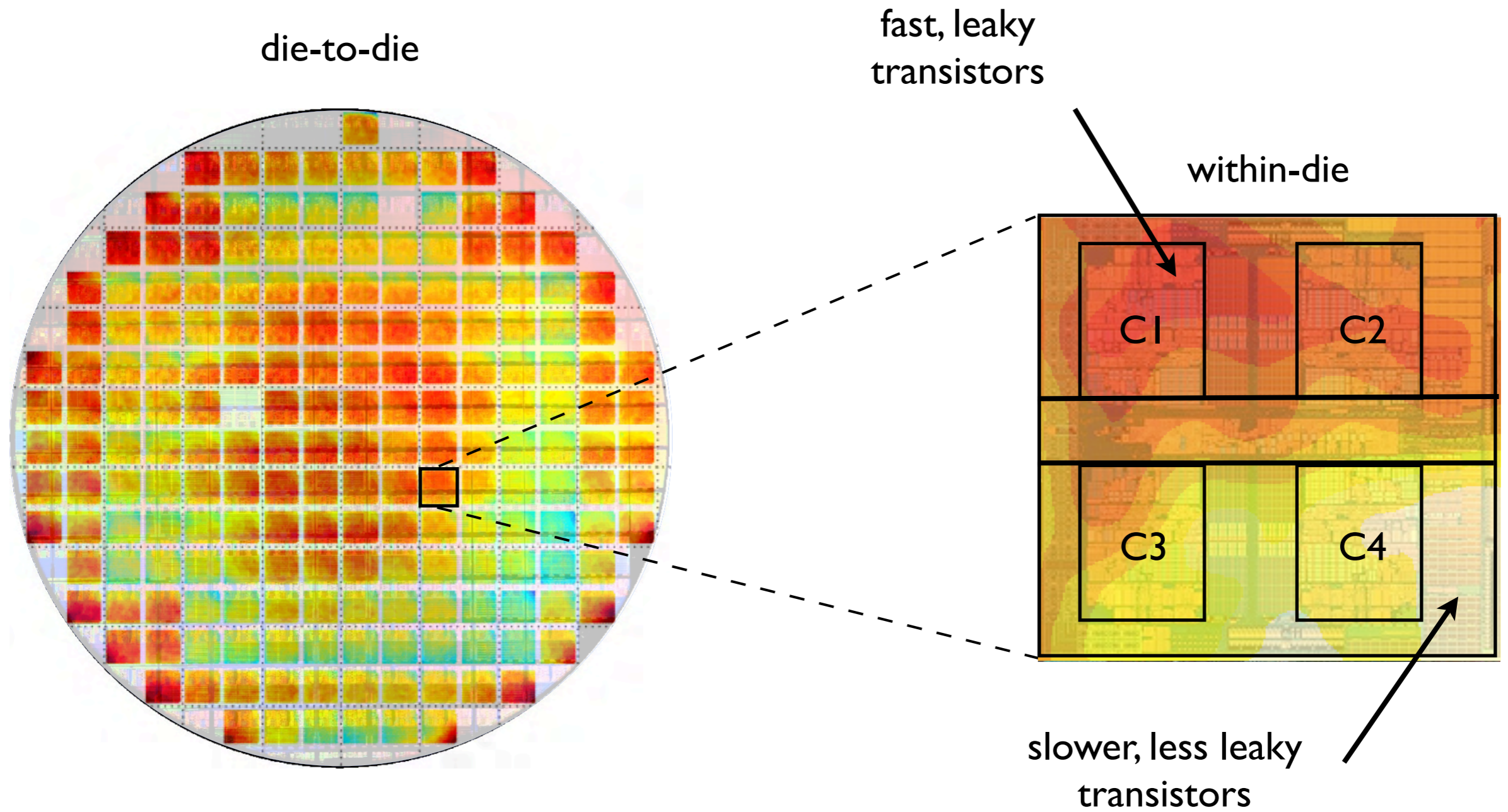
Process variation effects



Shekhar Borkar et al, Intel, DAC 2003



Variation components



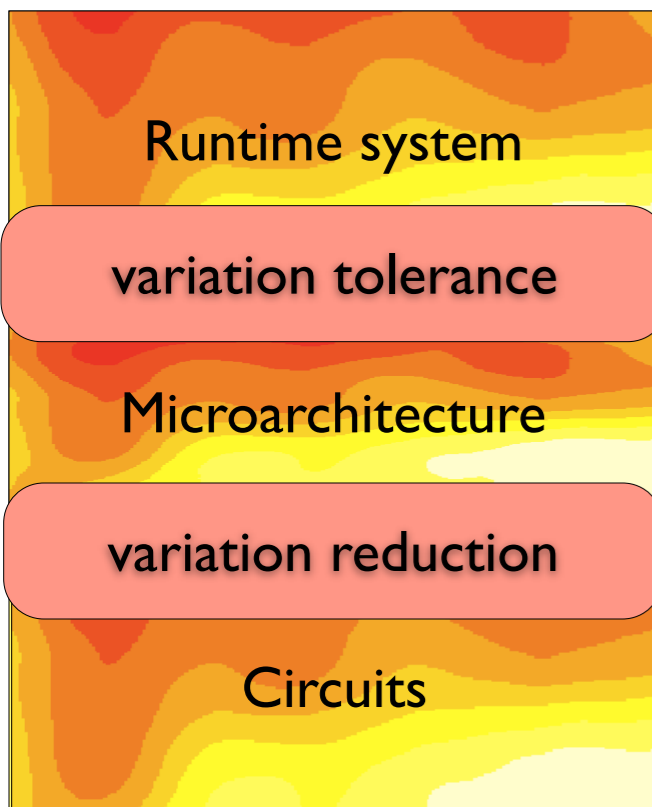


Addressing parameter variation

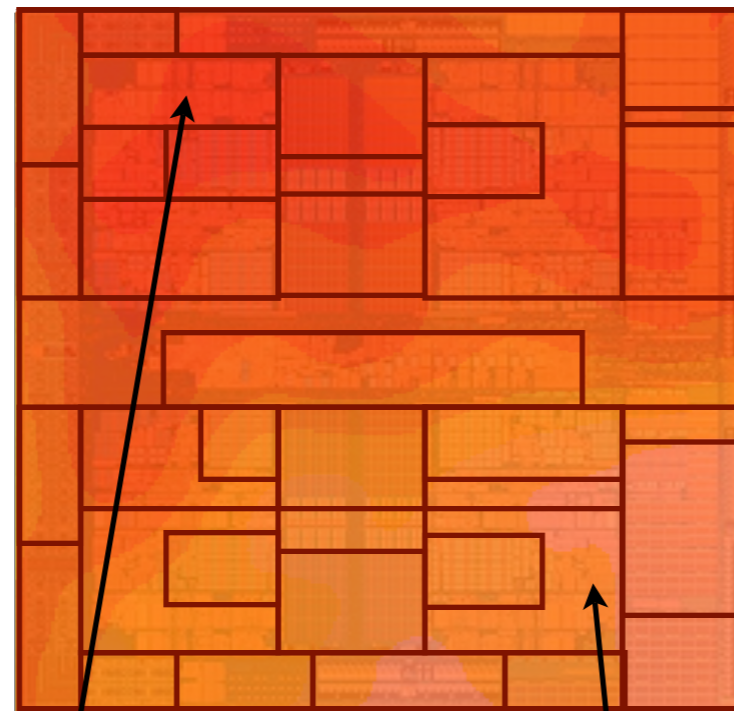
Variation reduction

Variation tolerance

computing stack



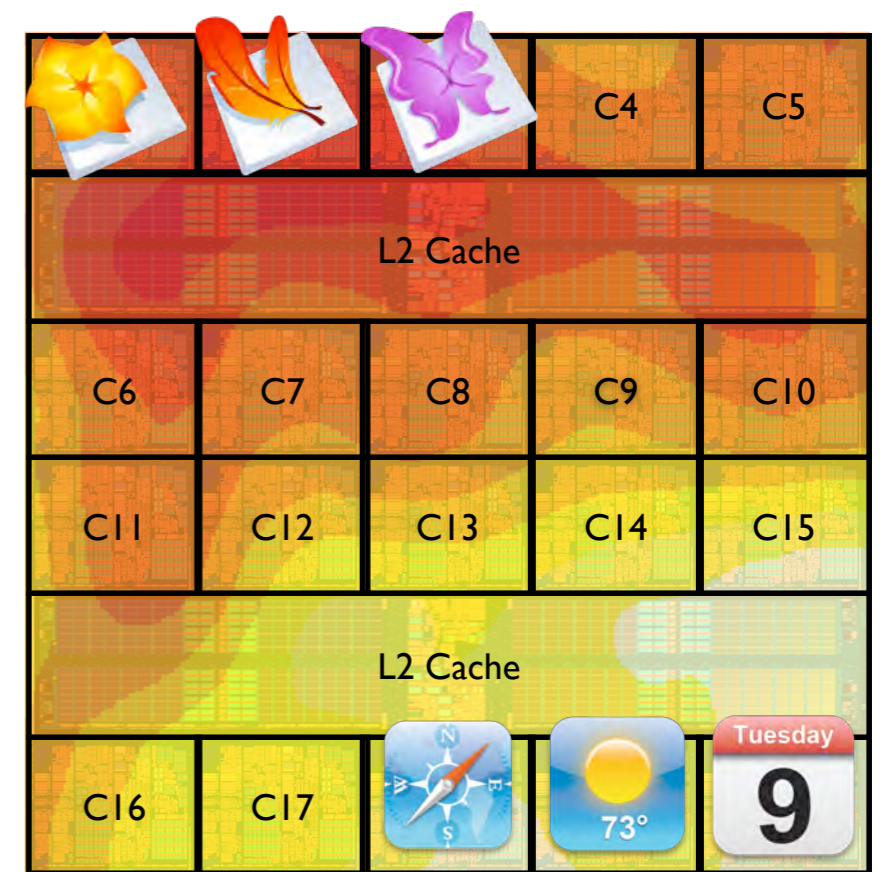
dynamic fine-grain body biasing



reduce power
of high power cells

speed up
slow cells

variation-aware application
scheduling and power management





Outline

- Two solutions:
 - Dynamic fine-grain body biasing [MICRO'07]
 - Variation aware scheduling and power management [ISCA'08]
- Evaluation
- Future work

Runtime system

variation tolerance

Microarchitecture

variation reduction

Circuits



Outline

- Two solutions:
 - **Dynamic fine-grain body biasing**
 - Variation aware scheduling and power management
- Evaluation
- Future work

Runtime system

variation tolerance





Microarchitecture

variation reduction

Circuits



Body biasing

- A voltage is applied between source/drain and substrate of a group of transistors
- Forward body bias (FBB) Frequency  Leakage 
- Reverse body bias (RBB) Frequency  Leakage 
- Key knob to trade off frequency for leakage power

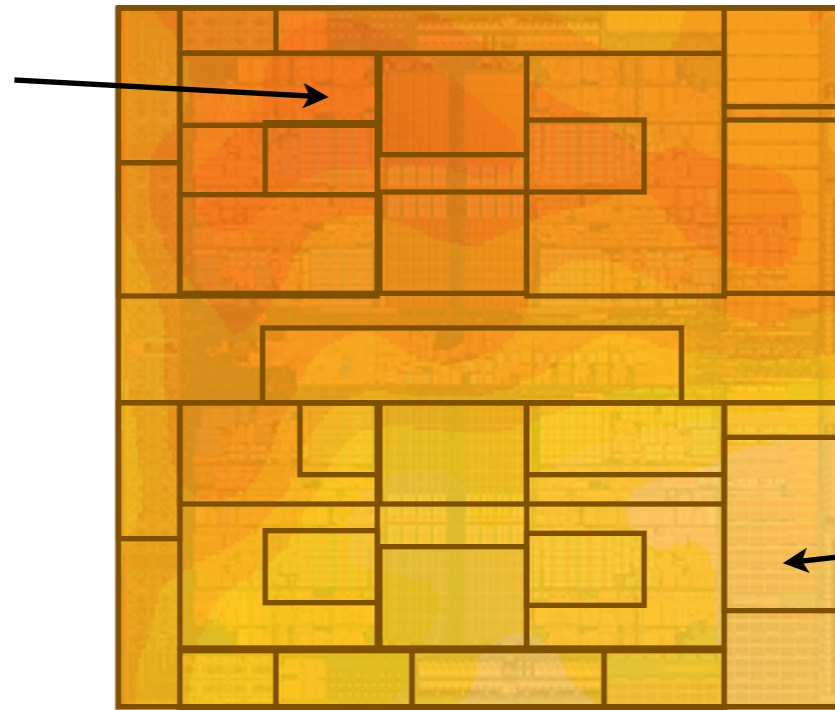




Static fine-grain body biasing (S-FGBB)

[Tschanz et al, Intel]

RBB
reduces static power
of leaky cells



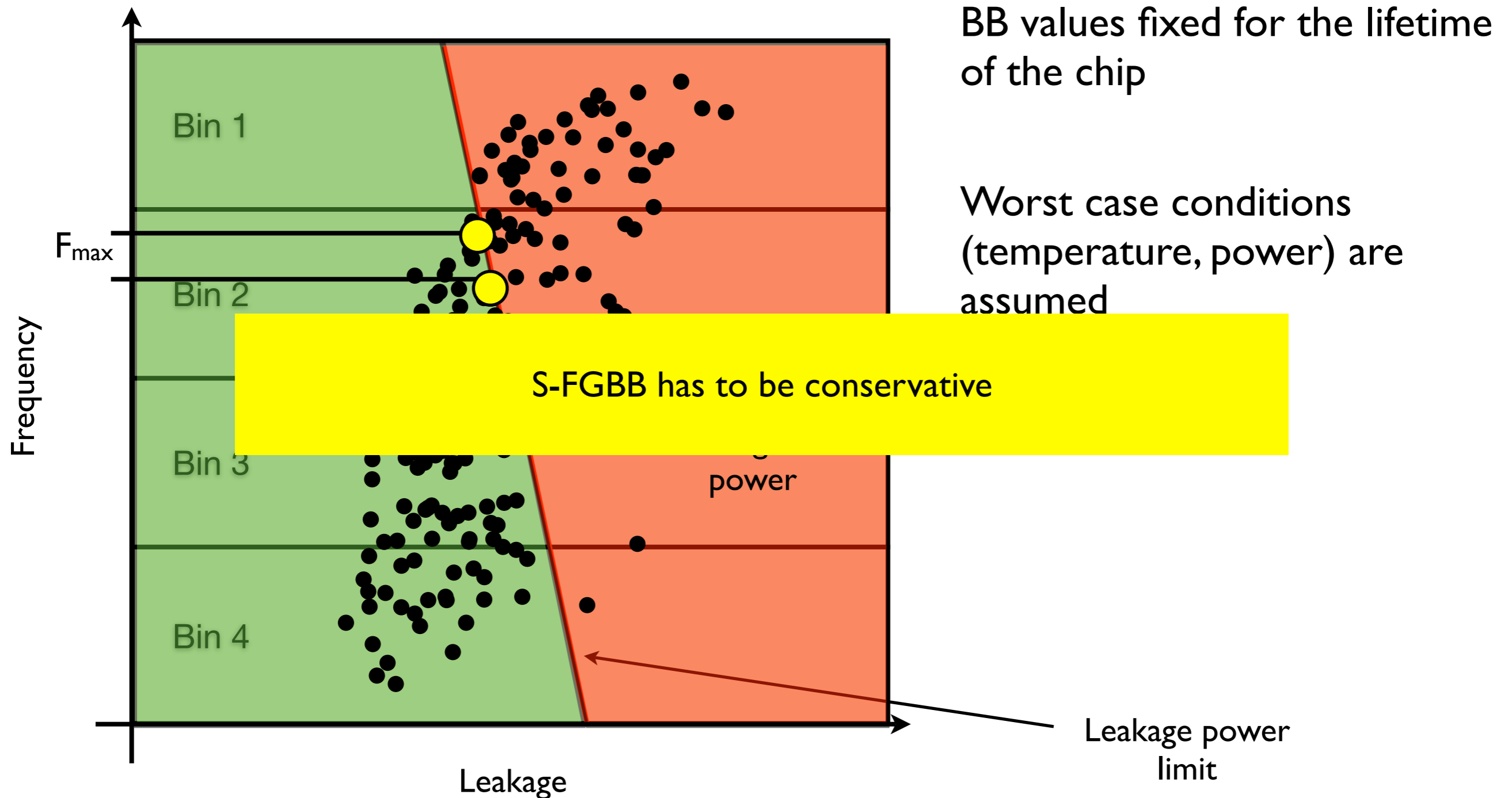
FBB
speeds up
slow cells

- The result is reduced WID variation
- improved processor frequency, lower power
- Additional control over a chip's frequency and power





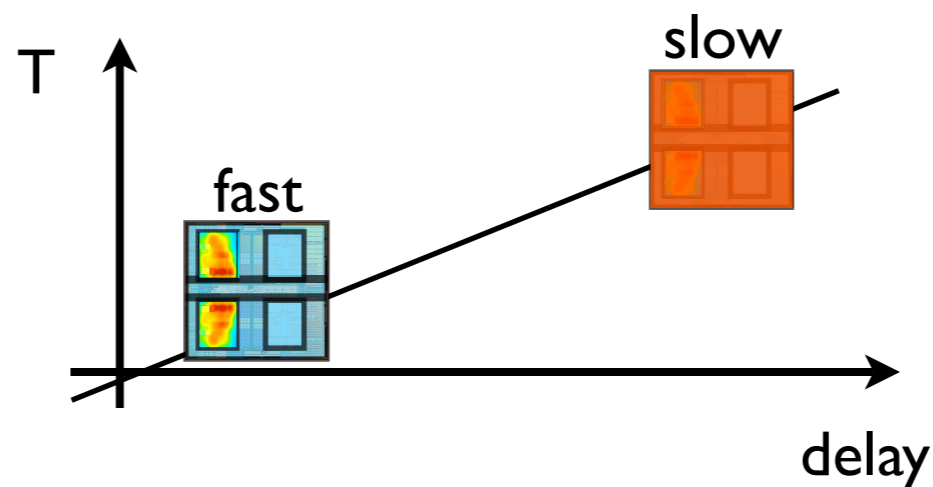
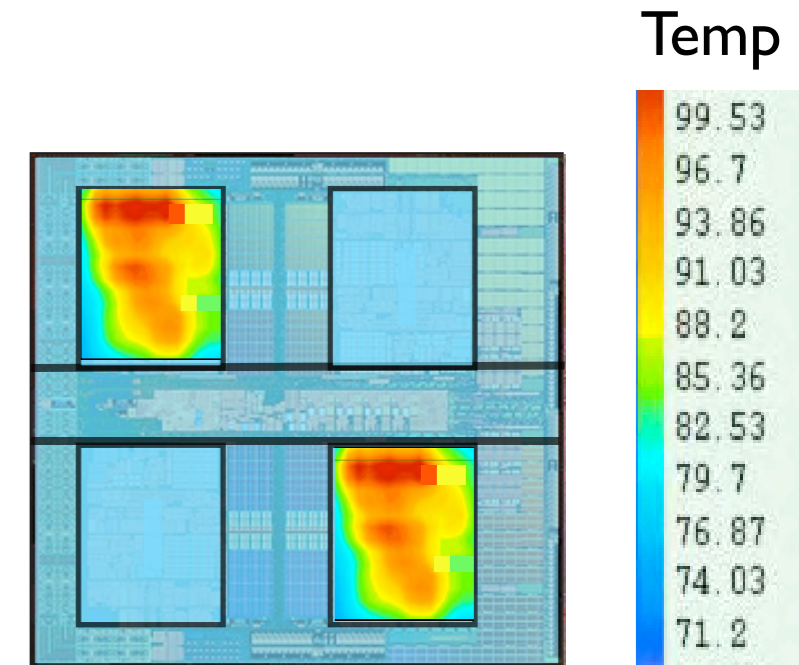
Static fine-grain body biasing





Dynamic fine-grain body biasing (D-FGGB)

- Significant temperature variation:
 - **Space**: across different cores
 - **Time**: as the activity factor of the workload changes
- Circuit delay increases with temperature:



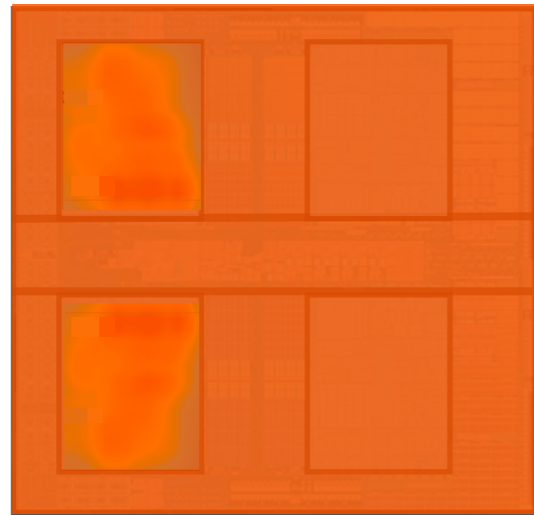


Dynamic fine-grain body biasing

S-FGBB

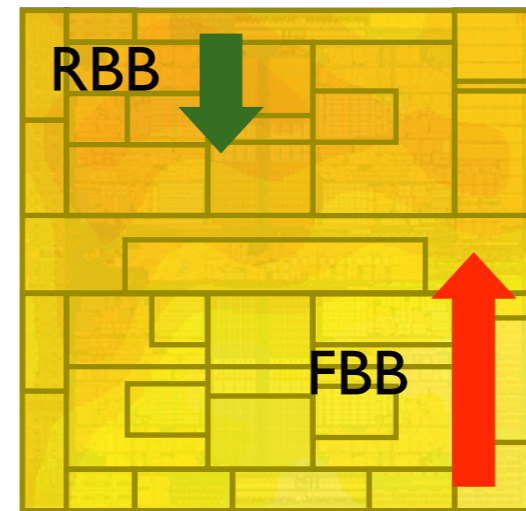
BB - fixed

max T



slow

Target: F_{max}

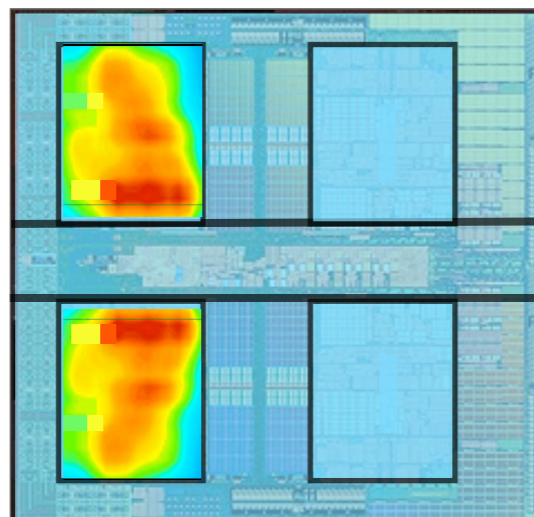


Higher power consumption

D-FGBB

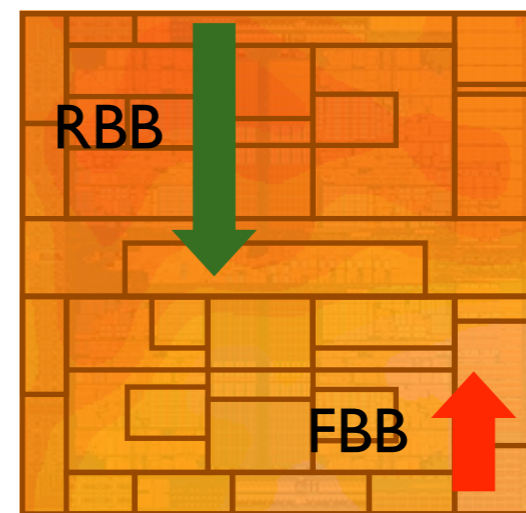
BB - variable

average T



fast

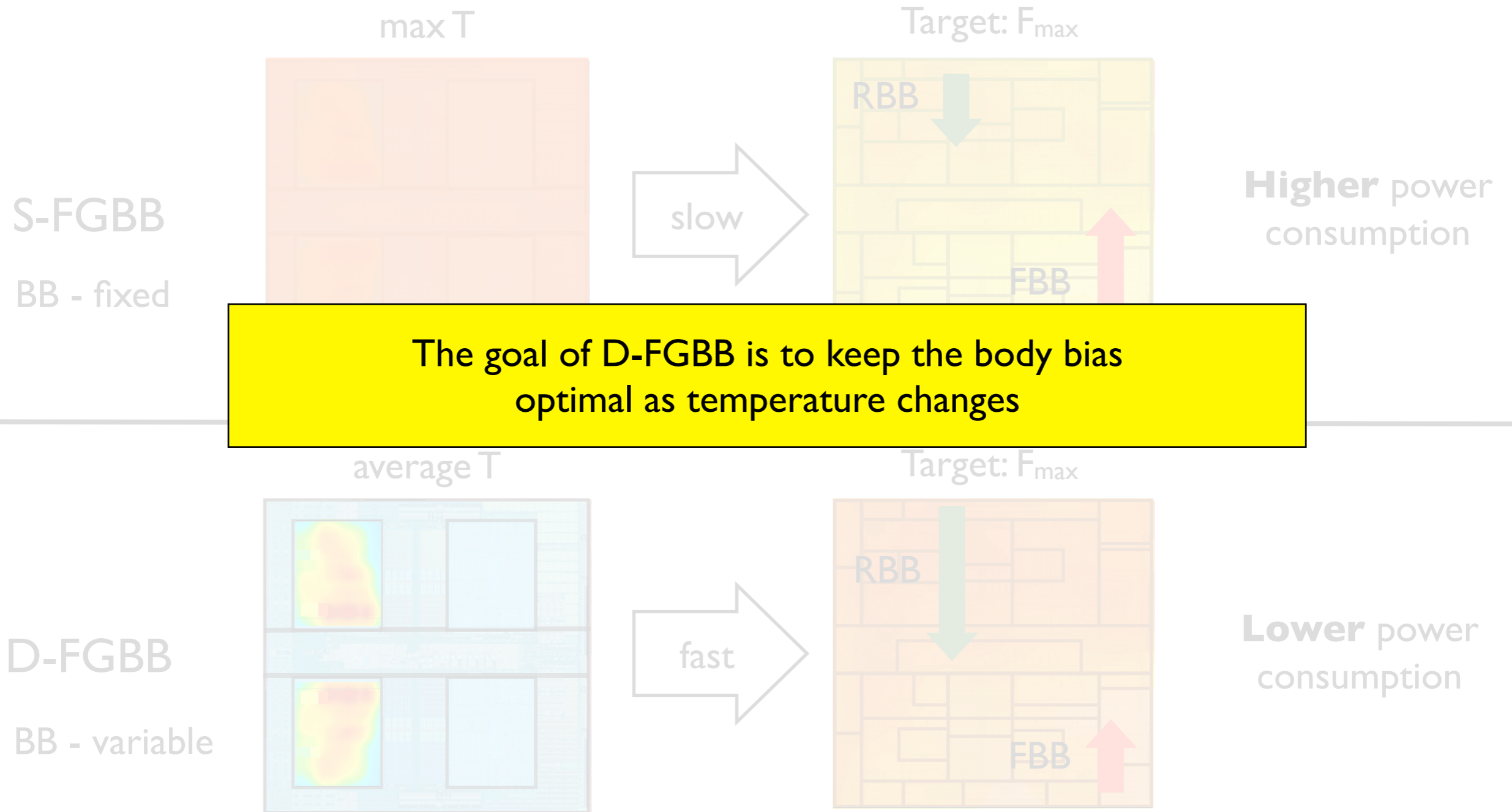
Target: F_{max}



Lower power consumption



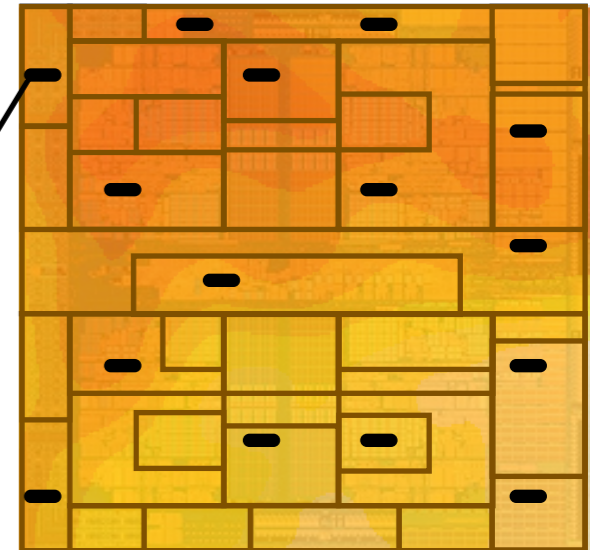
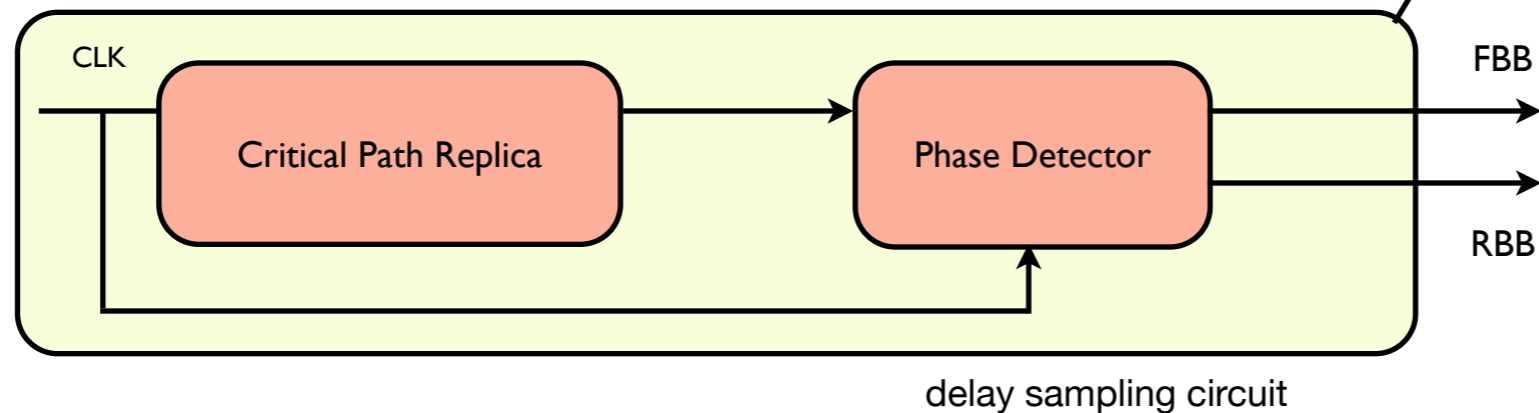
Dynamic fine-grain body biasing





Finding the optimal BB

- Dynamically measure the delay of each BB cell
- Delay sampling circuit:



- BB for each cell is adjusted as temperature changes
- Until optimal delay is reached



D-FGGB environments

environment

goal



Standard

Improve **frequency** and **power**



High performance

Maximize **frequency**

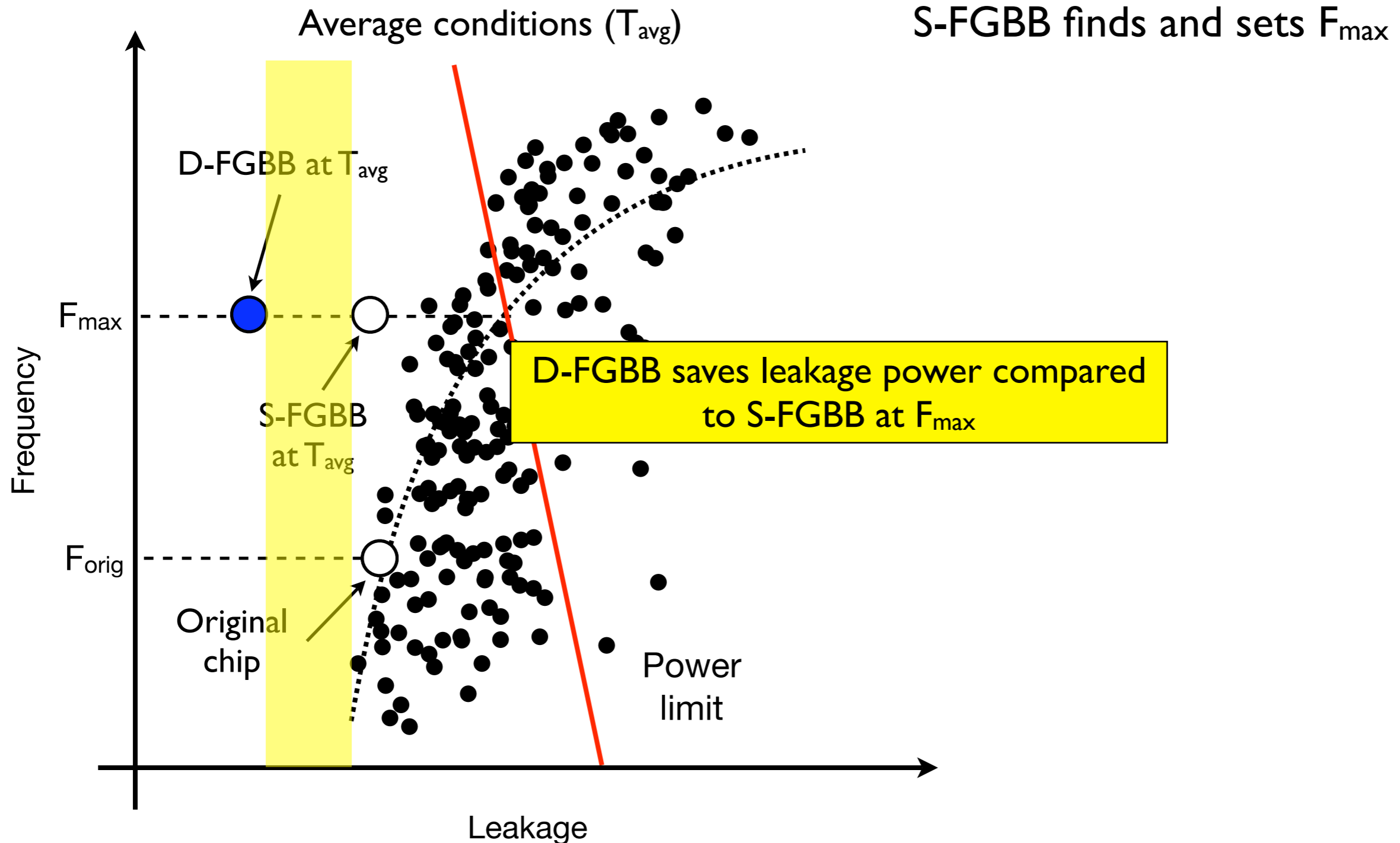


Low power

Minimize **leakage power**



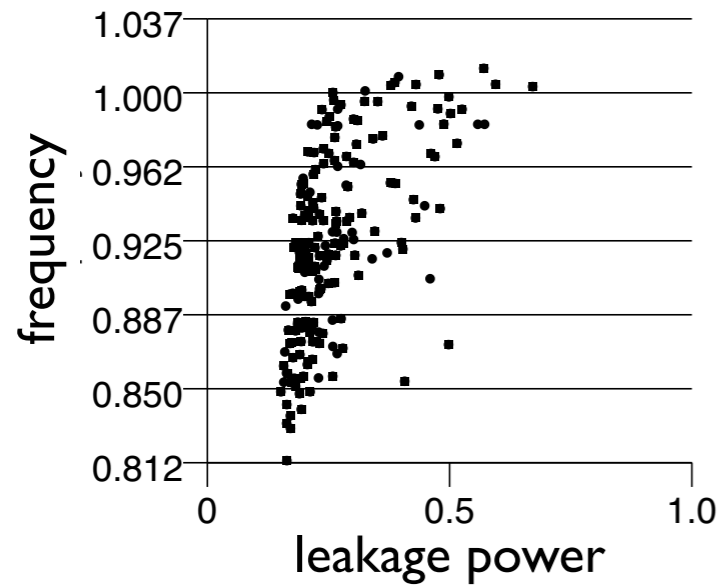
Standard environment



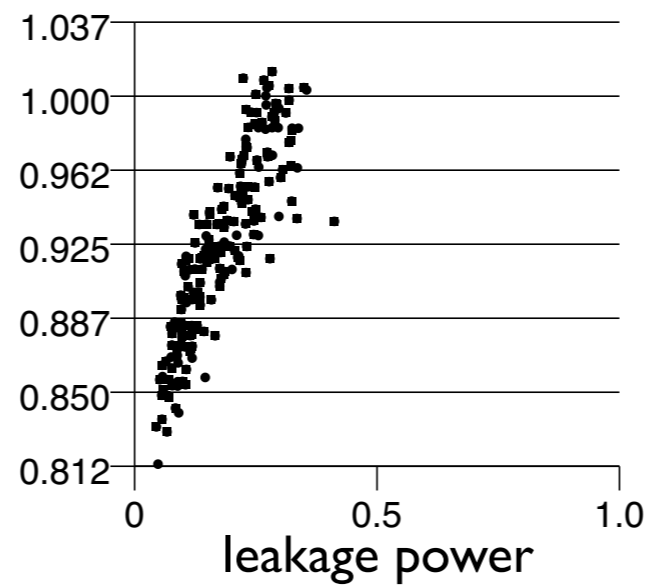


D-FGGB Summary

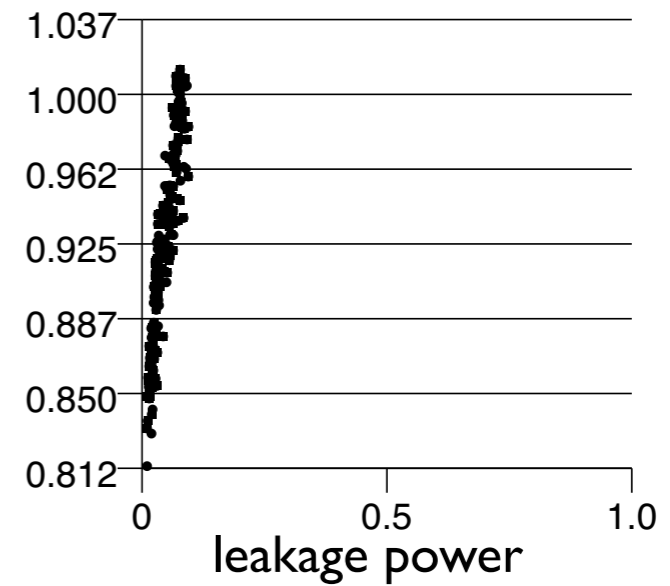
D-FGGB is very effective at reducing WID variation:



NoBB



S-FGGB



D-FGGB

- 40% lower leakage
- 10% higher frequency





Outline

- Two solutions:
 - Dynamic fine-grain body biasing
 - Variation aware scheduling and power management [ISCA'08]
- Evaluation
- Future work

Runtime system

variation tolerance

Microarchitecture

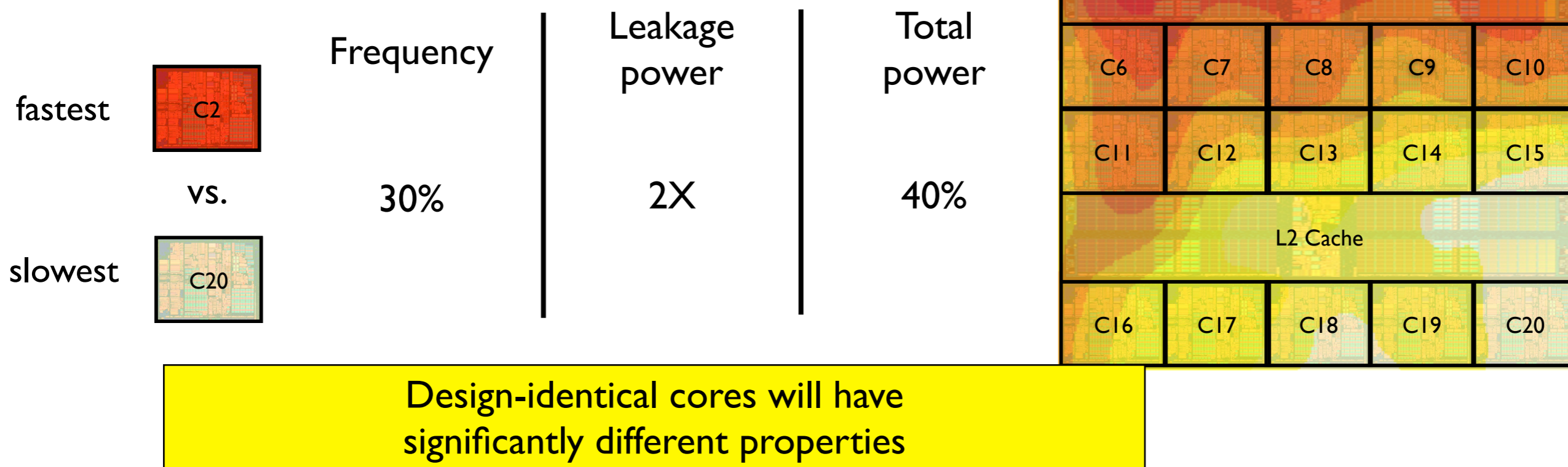
variation reduction

Circuits



Motivation

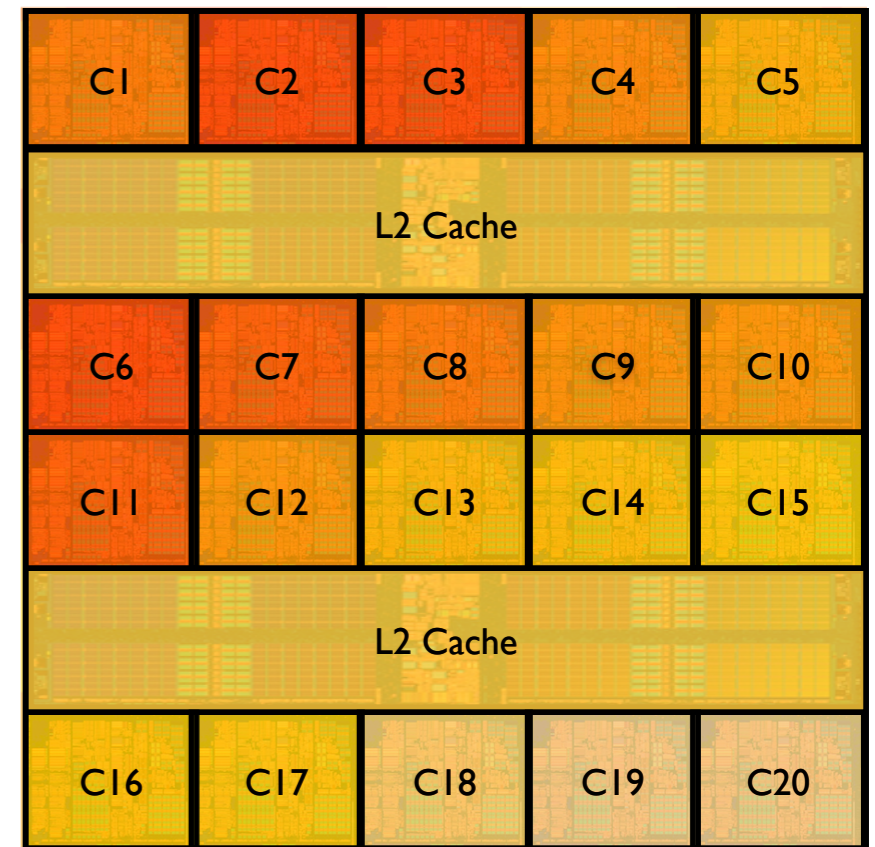
- Large CMPs will have significant core-to-core variation
- We model a 20-core CMP, 32nm





How can we exploit this variation?

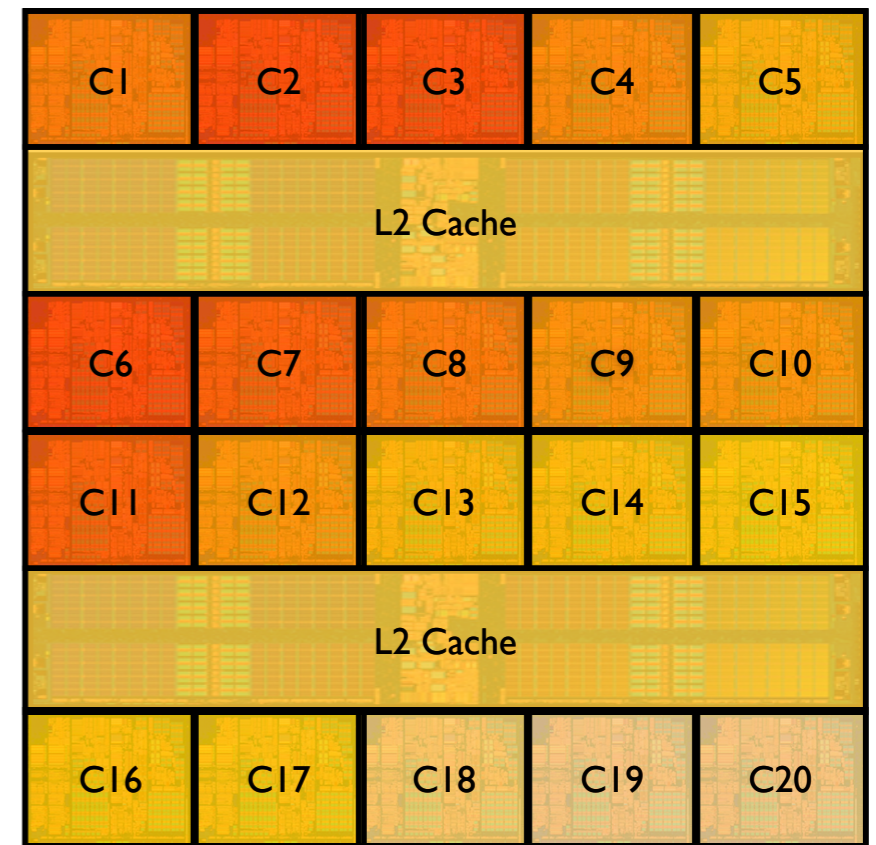
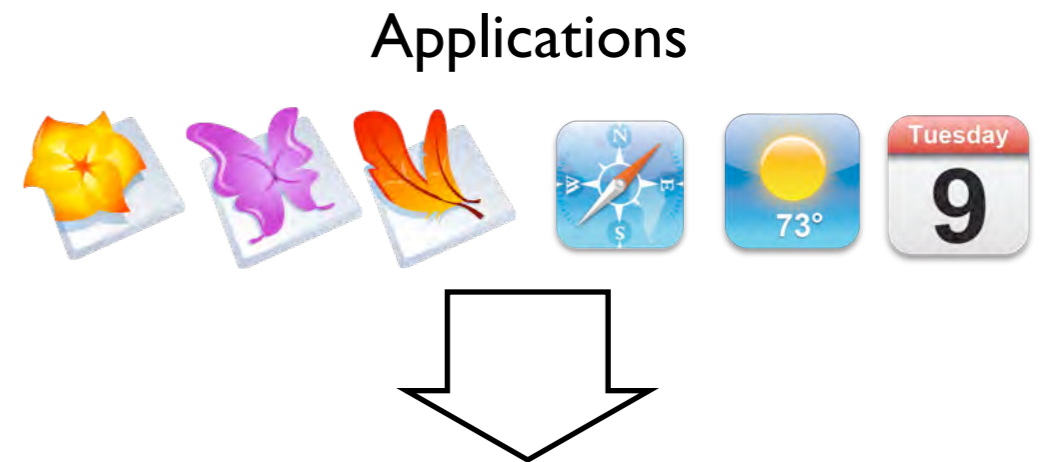
- Current CMPs run at the frequency of the slowest core
- We can run each core at the maximum frequency it can achieve
 - 15% average frequency increase
- Heterogeneous system
 - Variation-aware scheduling
 - Variation-aware power management





Variation-aware scheduling

- Variation in core frequency and power
- Application behavior
 - dynamic power consumption
 - instructions per cycle (IPC)
- System goals:
 - reduce power
 - improve performance





Variation-aware scheduling

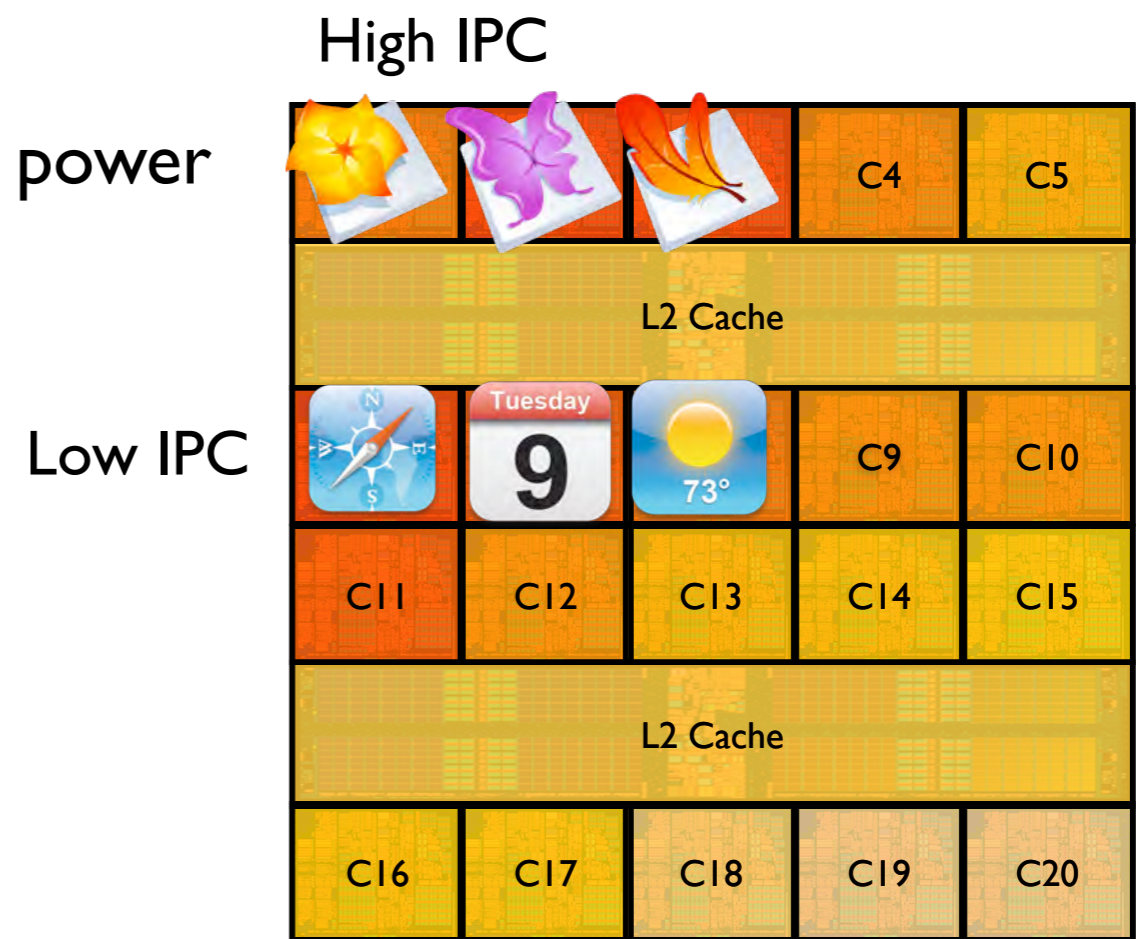
Variation-aware scheduling algorithms:

- Reduce power:

Assign applications with high dynamic power to low power cores (**VarPower**)

- Improve performance:

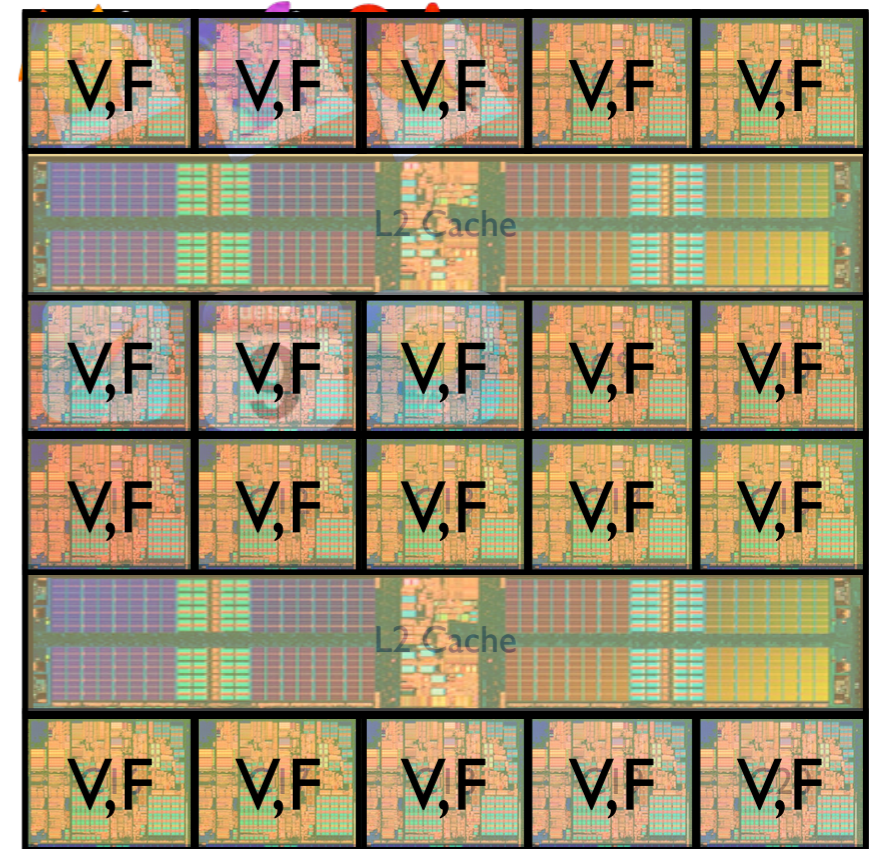
Assign high IPC applications to high frequency cores (**VarPerf**)





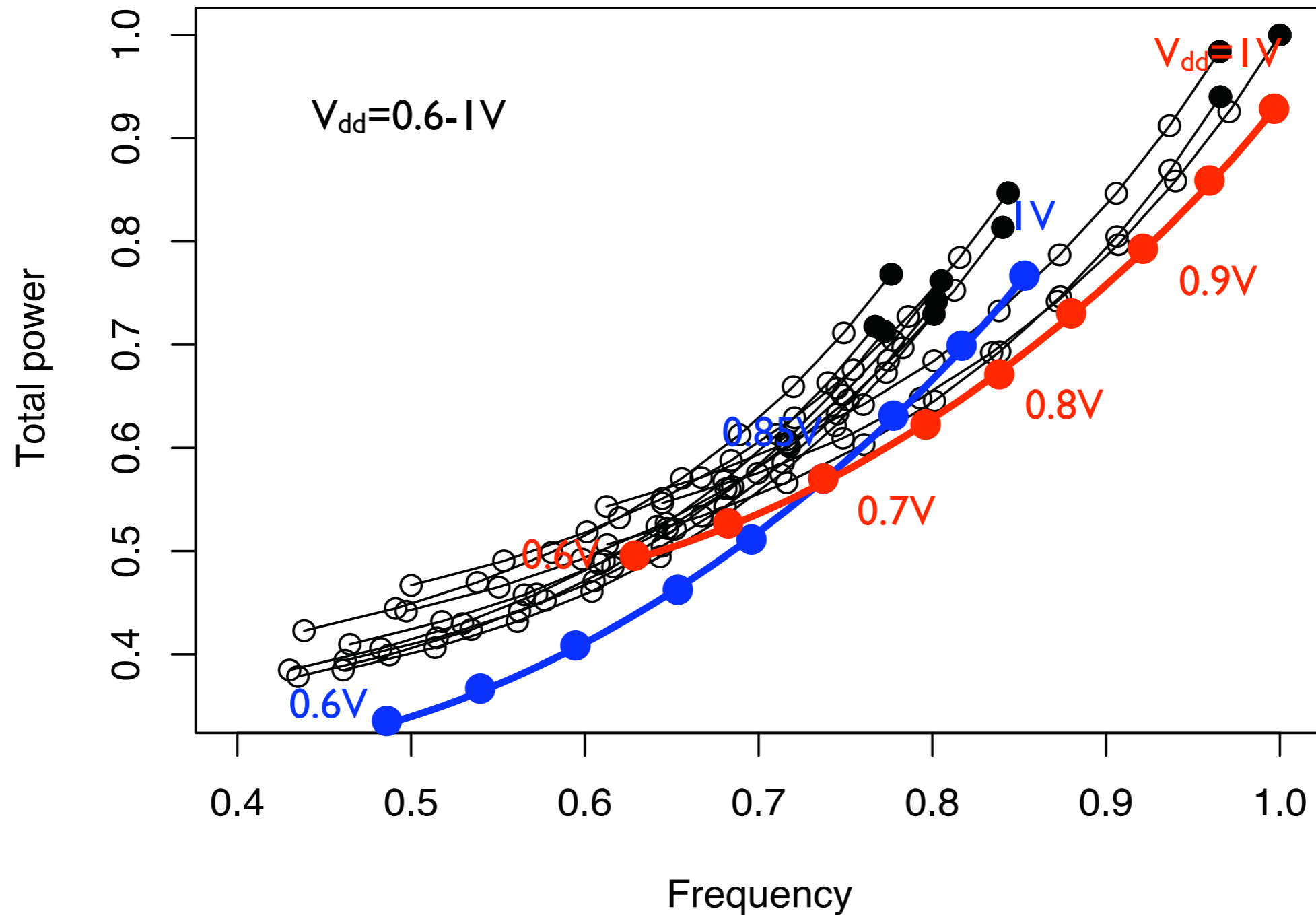
Variation-aware power management

- Dynamic voltage and frequency scaling (DVFS)
- Core-level control over voltage and frequency
- The challenge:
 - Find optimal (V,F) for each core
- Variation makes the problem more difficult





DVFS under variation

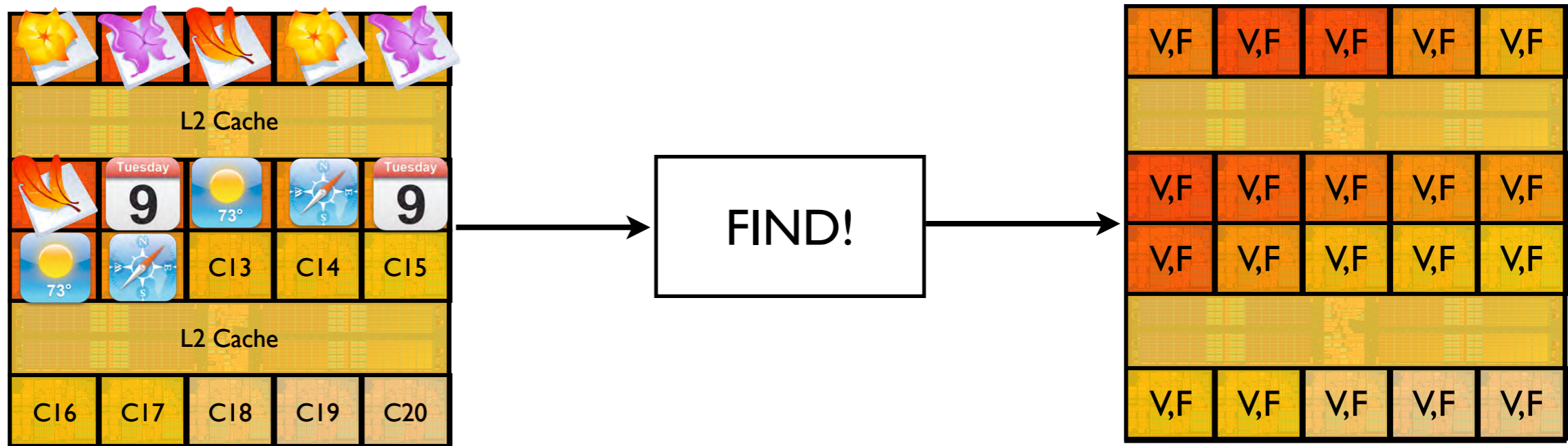




Optimization problem

Given a mapping of threads to cores (**VarPerf**):

best (V_i, F_i) of each core



- **Goal:** maximize system throughput (MIPS)
- **Constraint:** keep total power below budget

50W



75W



100W

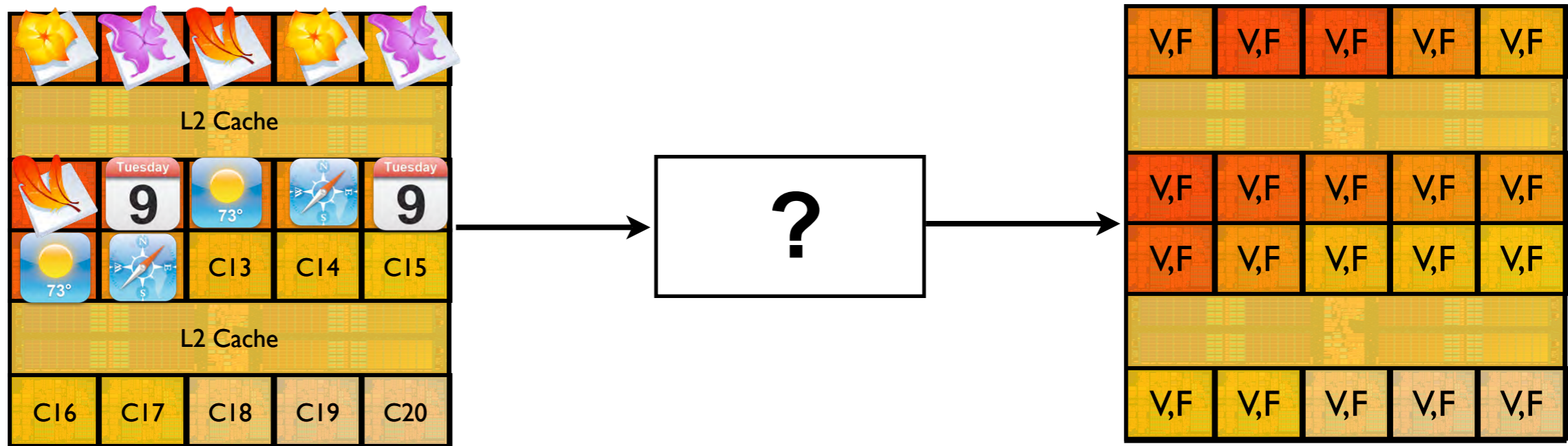




Optimization problem

Given a mapping of threads to cores (**VarPerf**):

best (V_i, F_i) of each core



- **Goal:** maximize system throughput (MIPS)
- **Constraint:** keep total power below budget

50W



75W



100W





Possible solutions

LinOpt

- Exhaustive search: too expensive
- Simulated annealing (***SAnn***)
 - not practical at runtime
- Linear programming (***LinOpt***)
 - simpler, faster
 - requires some approximations



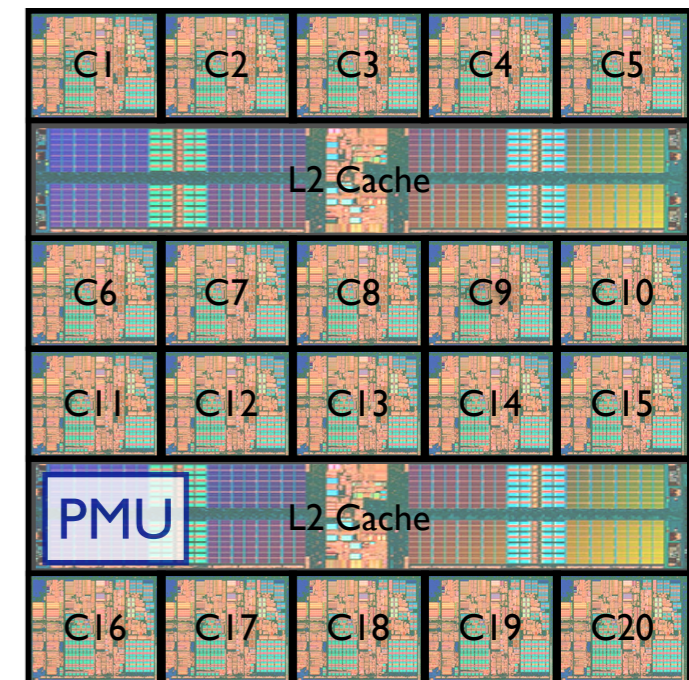
LinOpt problem definition

- **Linear programming:**
 - Maximize objective function: $f(x_1, \dots, x_n)$, with x_1, \dots, x_n independent
 - Subject to constraints such as: $g(x_1, \dots, x_n) < C$
 - f, g are linear functions
- **Variables:** voltages V_1, \dots, V_n for all cores
- **Objective function:** maximize throughput
 - Throughput (MIPS) = Frequency \times IPC = $f(V_1, \dots, V_n)$
- **Constraint:** keep power under P_{target}
 - Power = $g(V)$



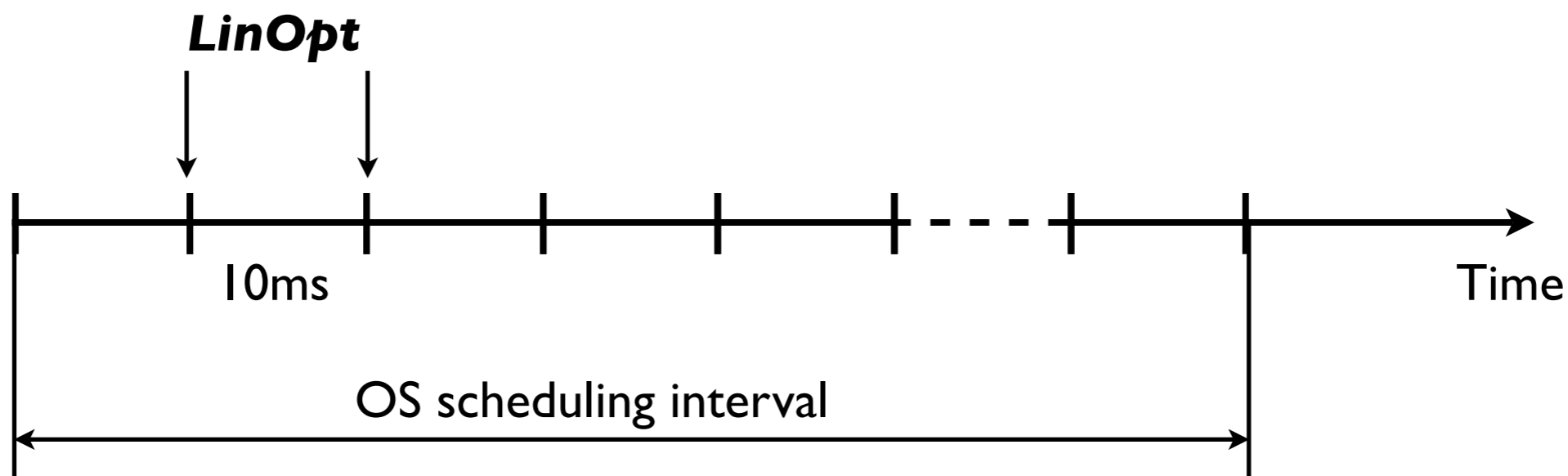
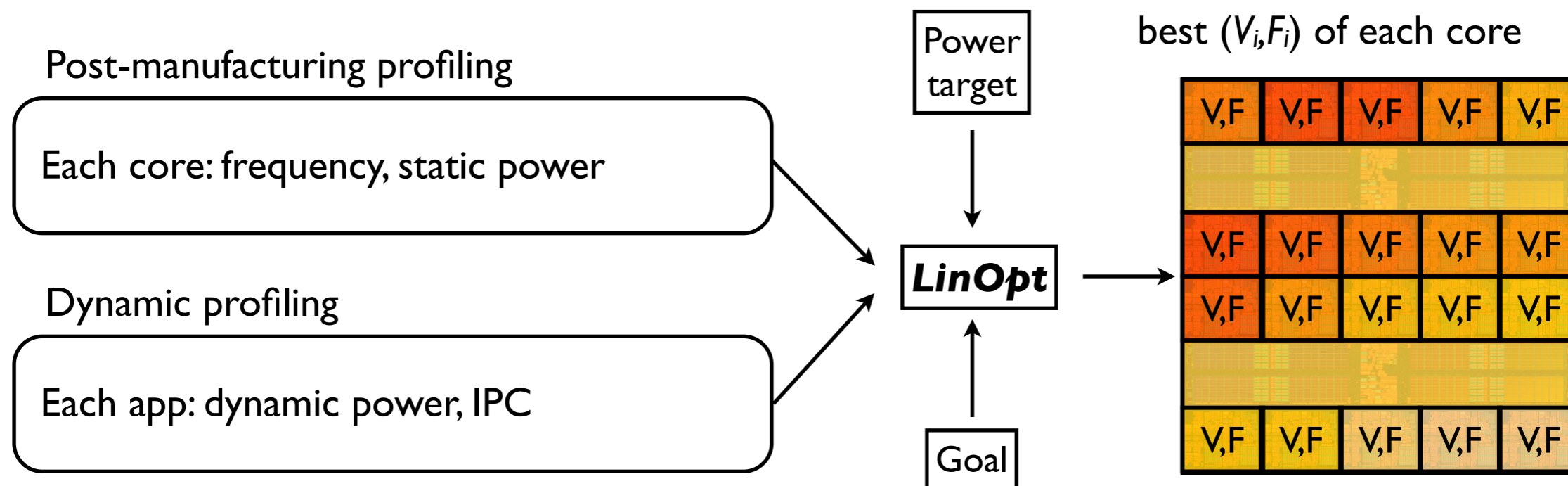
LinOpt implementation

- ***LinOpt*** works together with the OS scheduler
 - OS scheduler maps applications to cores (e.g. ***VarPerf***)
 - ***LinOpt*** then finds (V,F) settings for each core
- ***LinOpt*** runs periodically as a system process
 - on a spare core
 - Power management unit (PMU)
 - on-chip microcontroller (*Foxton*)
- ***LinOpt*** uses profile information as input





LinOpt implementation





Outline

- Two solutions:
 - Dynamic fine-grain body biasing
 - Variation aware scheduling and power management
- Evaluation
- Future work

Runtime system

variation tolerance

Microarchitecture

variation reduction

Circuits

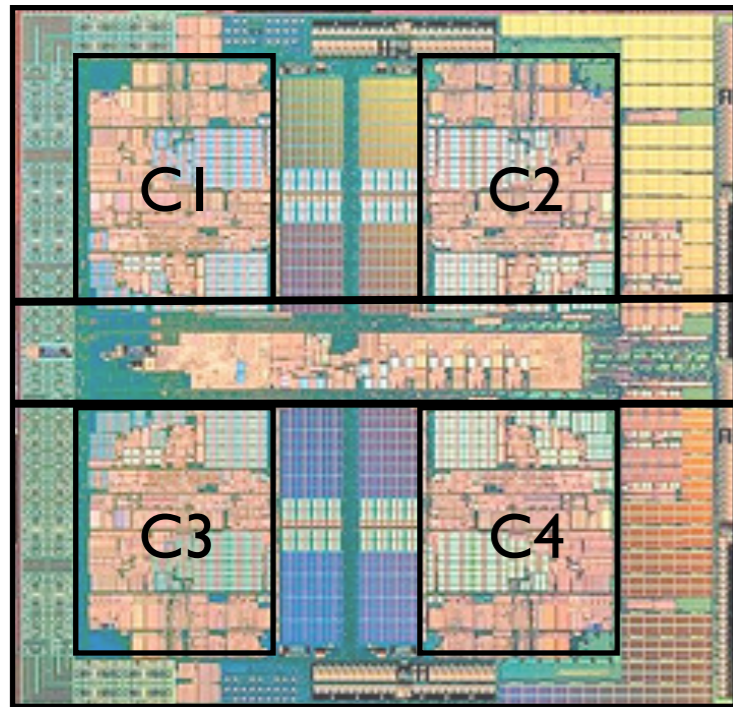


Evaluation infrastructure

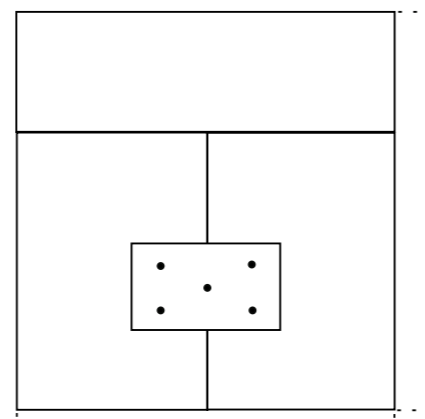
- Process variation model - *VARIUS* [IEEE TSM'08]
 - Monte Carlo simulations for 200 chips
- SESC - cycle accurate microarchitectural simulator
- HotLeakage, SPICE model - leakage power
- Hotspot - temperature estimation
- Mix of SPECint and SPECfp benchmarks



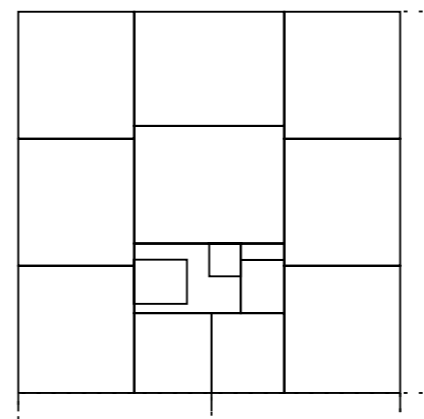
Dynamic fine-grain body biasing



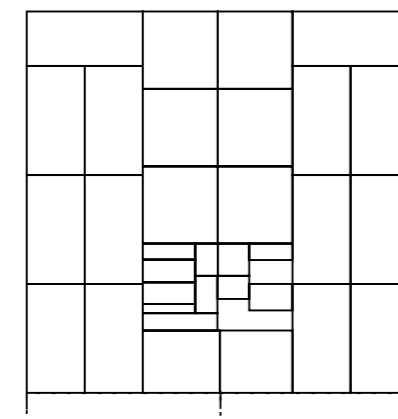
- 4-core CMP
- 45nm technology, 4GHz
- We evaluate FGBB at different granularities (1-144 cells)



FGBB16



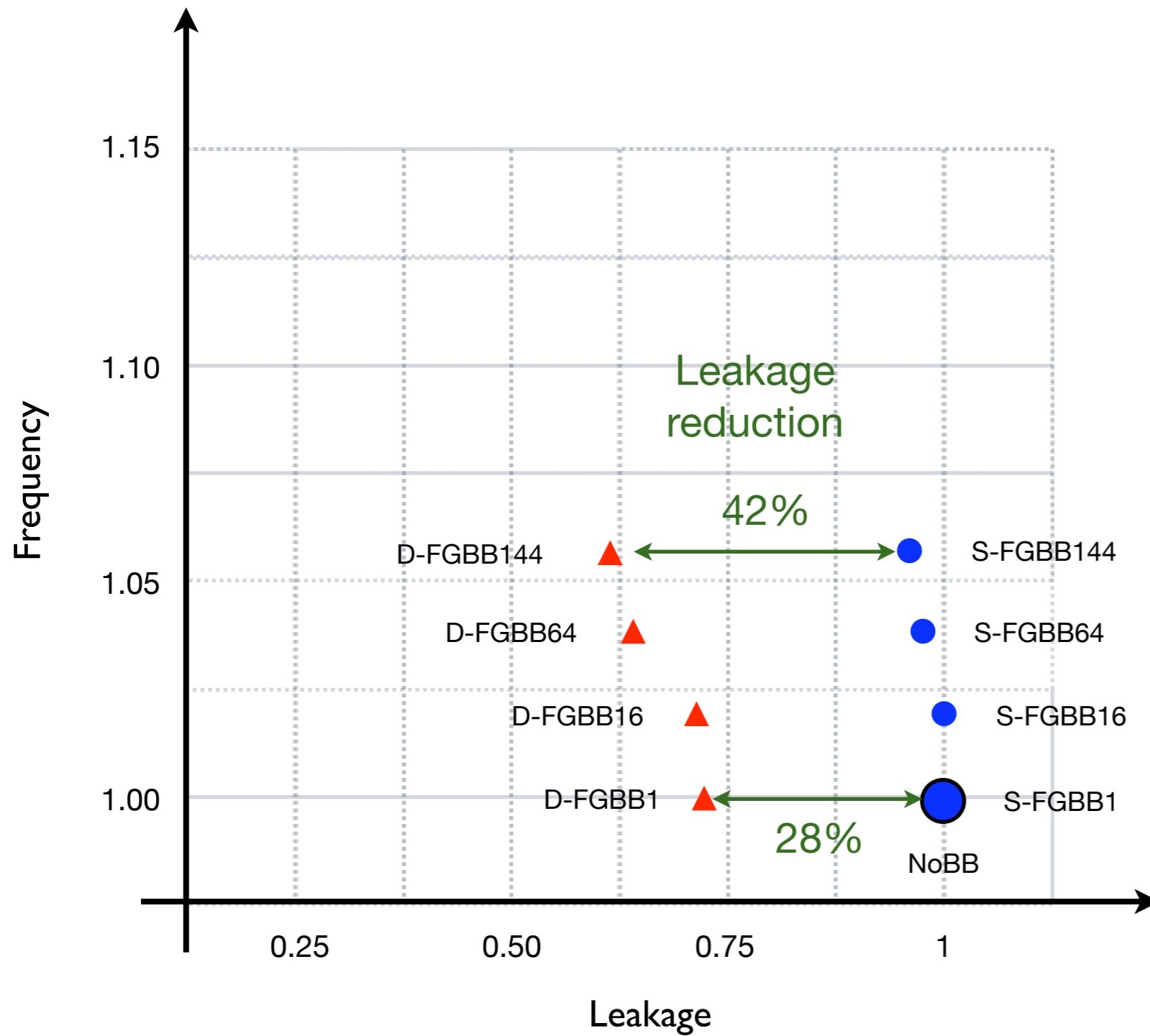
FGBB64



FGBB144



D-FGGB Standard



More BB cells result in higher frequency and lower leakage



Other environments



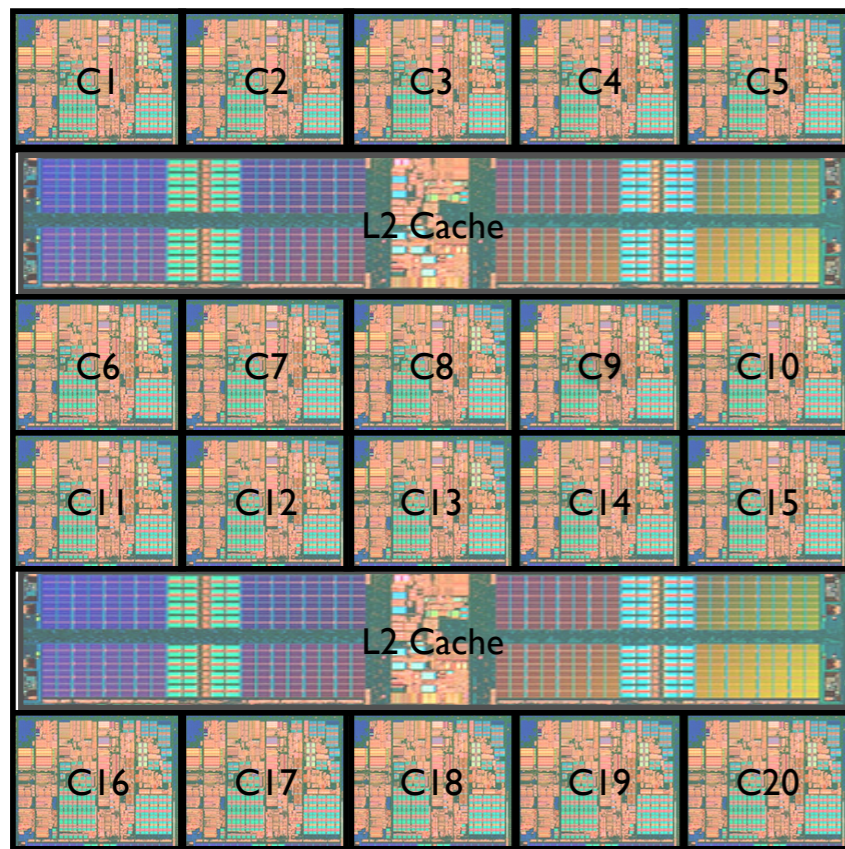
- D-FGGB High Performance: 7-10% frequency increase compared to S-FGGB



- D-FGGB Low Power - 10-50% leakage reduction compared to S-FGGB



Variation-aware scheduling and power management



- 20-core CMP
- 32nm technology, 4GHz



- Multiprogrammed workload: 1-20 applications
 - from a pool of SPECint and SPECfp benchmarks



Power management schemes

Goal: - maximize throughput

Constraint: - keep power below budget (75W)

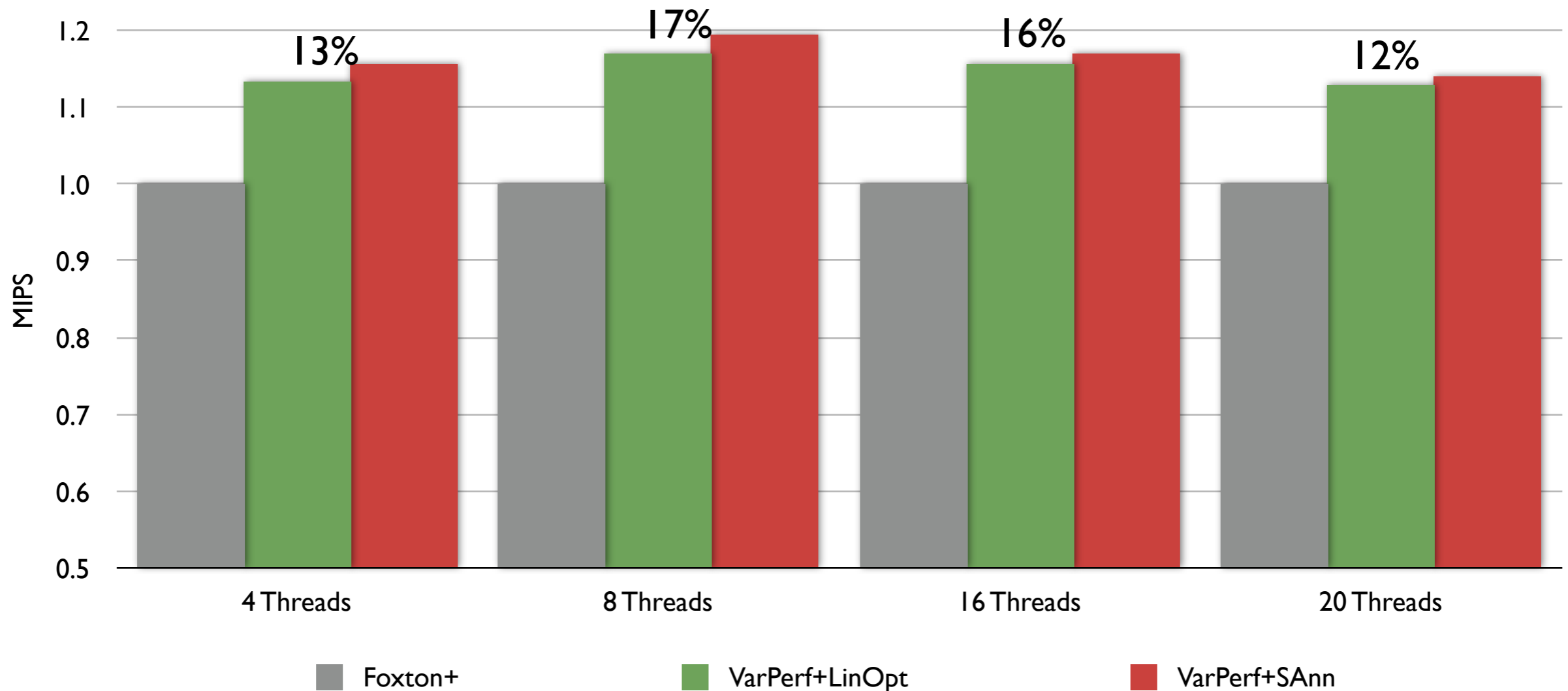
Foxtan+: baseline

VarPerf+LinOpt: proposed scheme

VarPerf+SAnn: approximate upper bound



Throughput improvements



- ***VarPerf+LinOpt***: 12-17% over ***Foxtan+***
- ***LinOpt***: within 2% of ***SAnn***

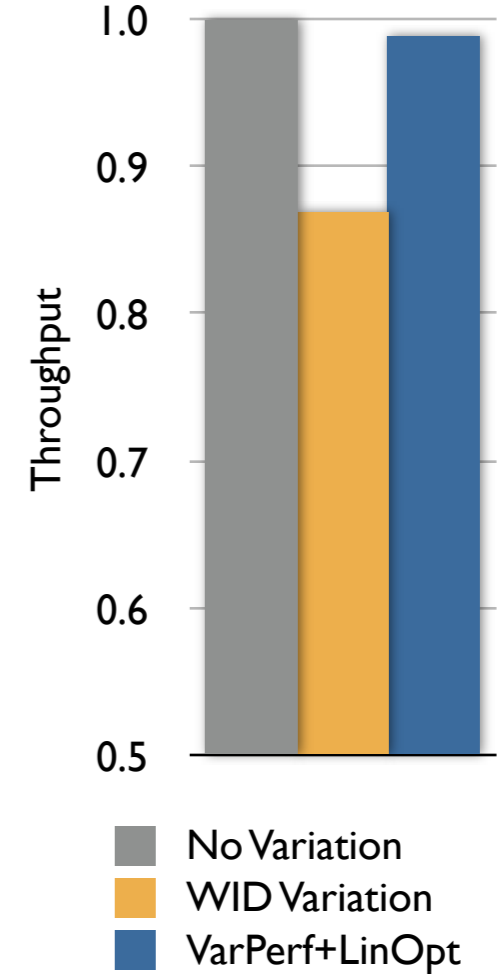
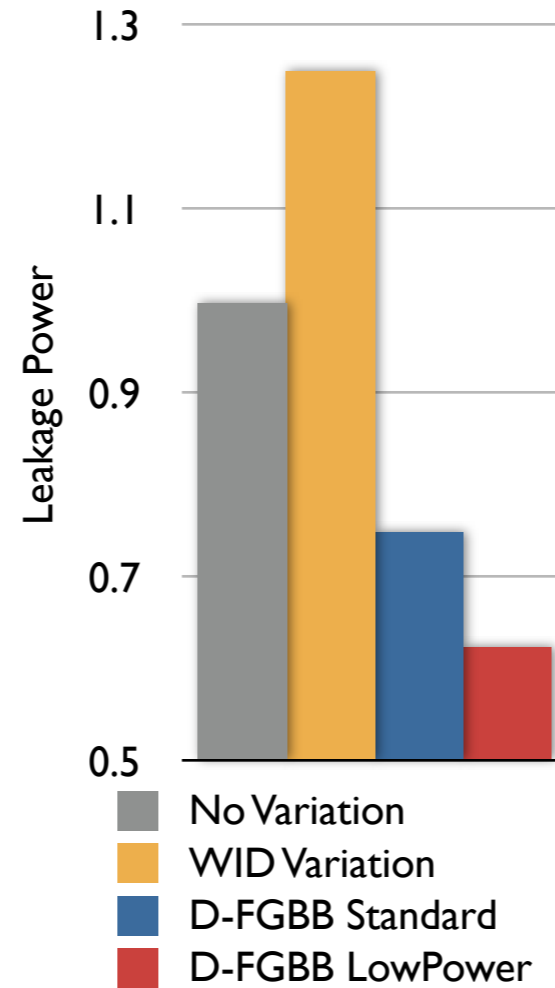
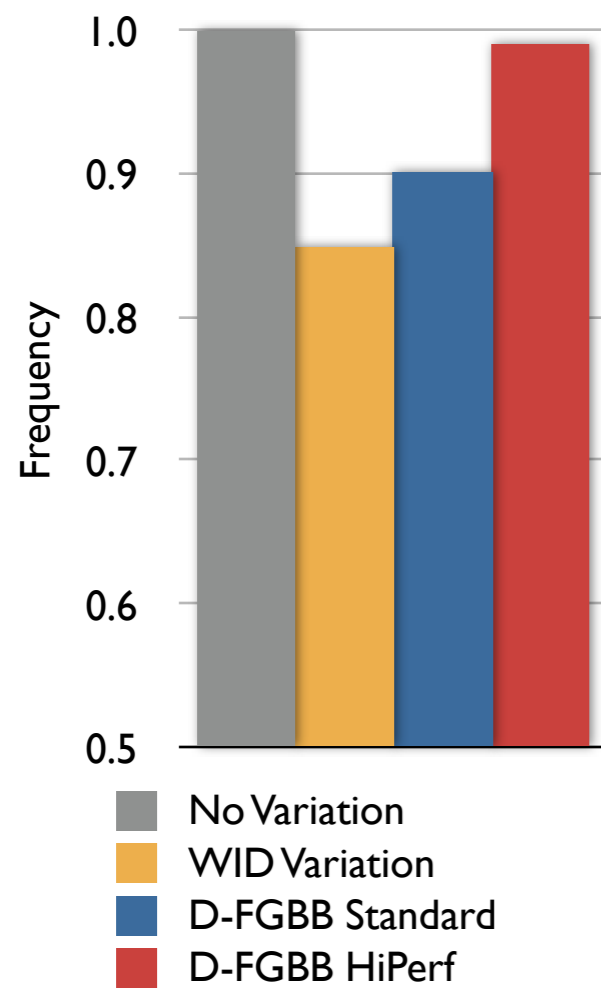


To sum up...

How much of the performance/power have we recovered?

dynamic fine-grain body biasing

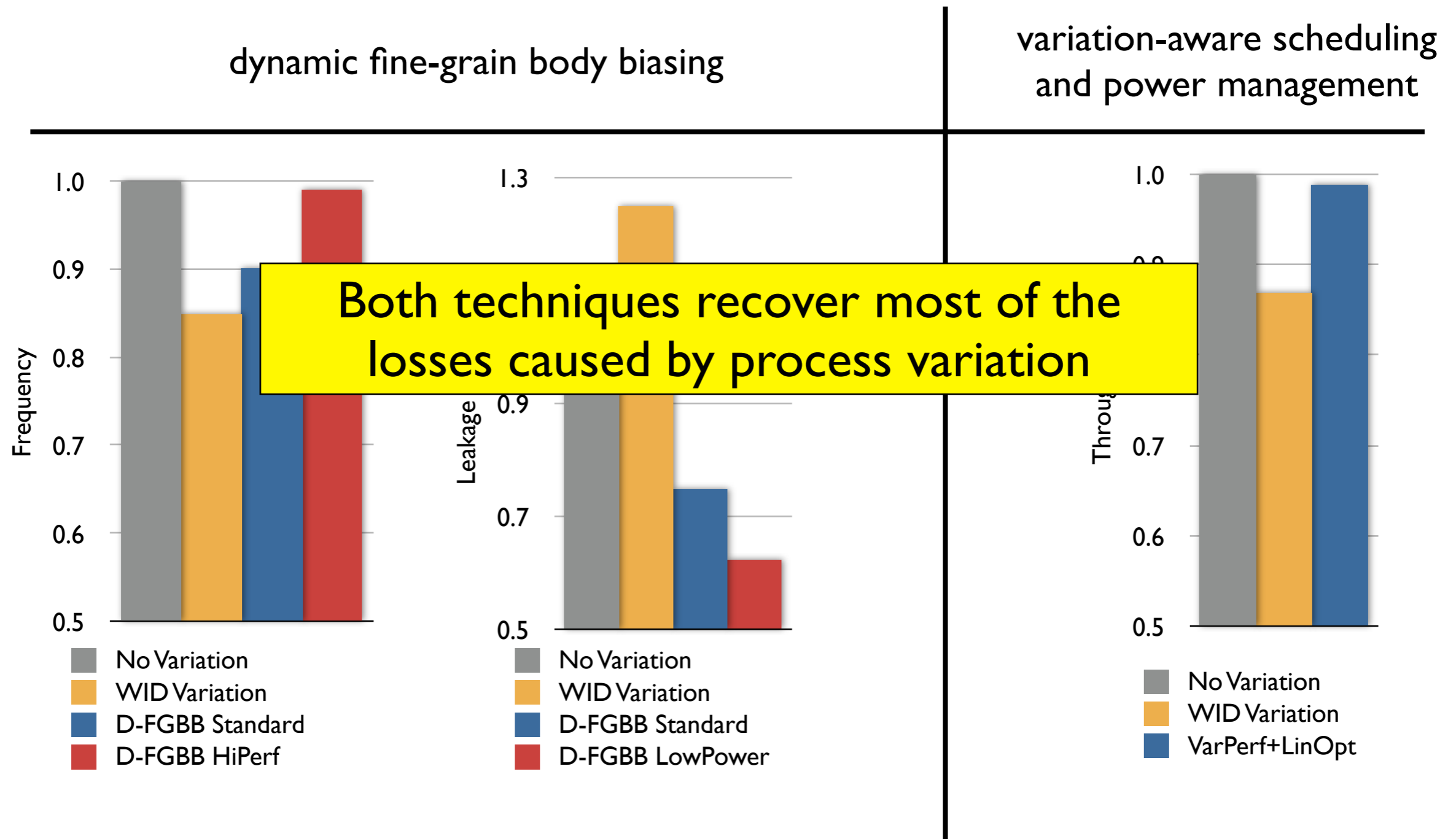
variation-aware scheduling
and power management





To sum up...

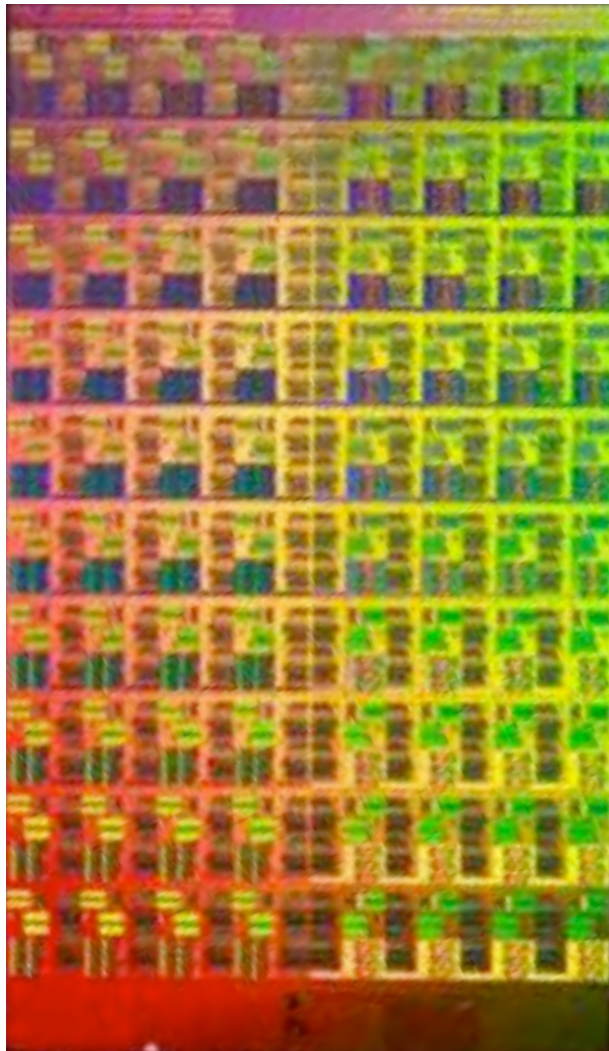
How much of the performance/power have we recovered?



Both techniques recover most of the losses caused by process variation



Outline



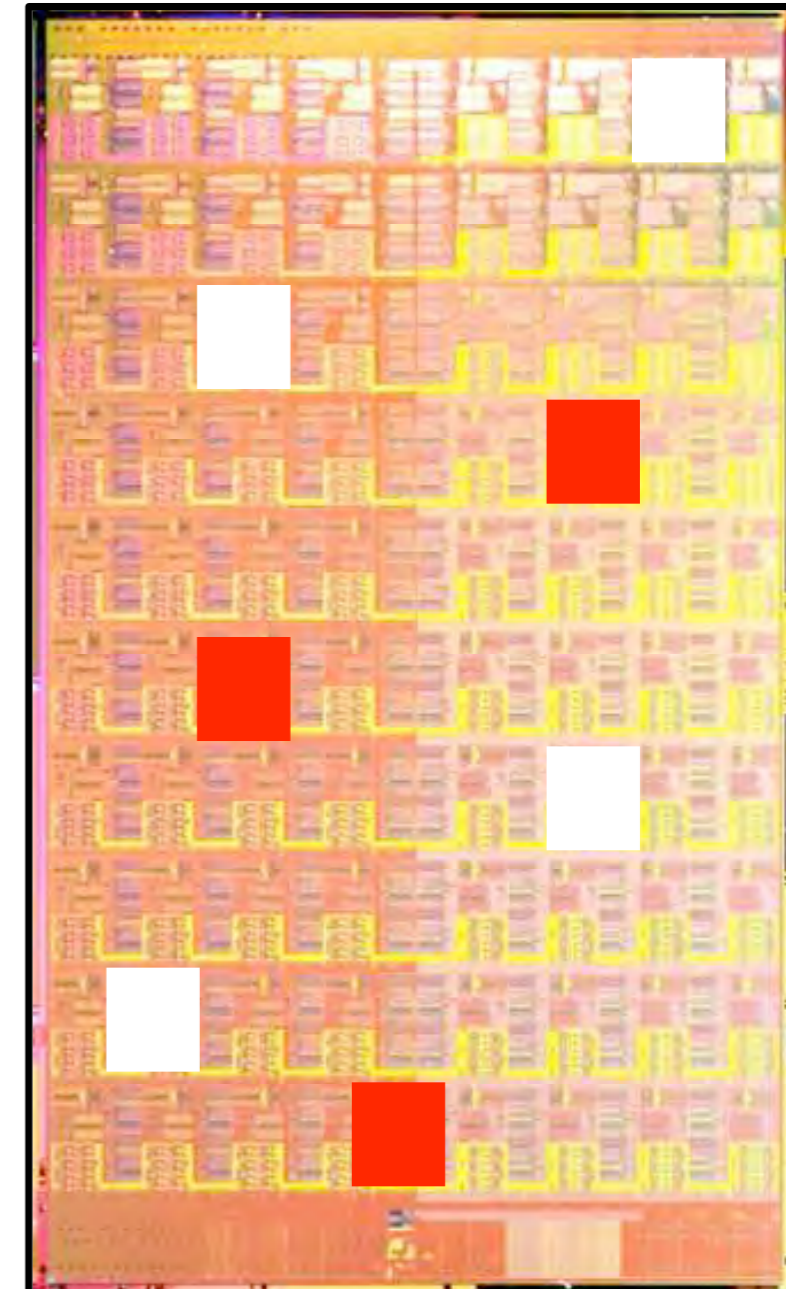
Intel 80-core Polaris

- Two solutions:
 - Dynamic fine-grain body biasing
 - Variation aware scheduling and power management
- Evaluation
- Future work



Future work

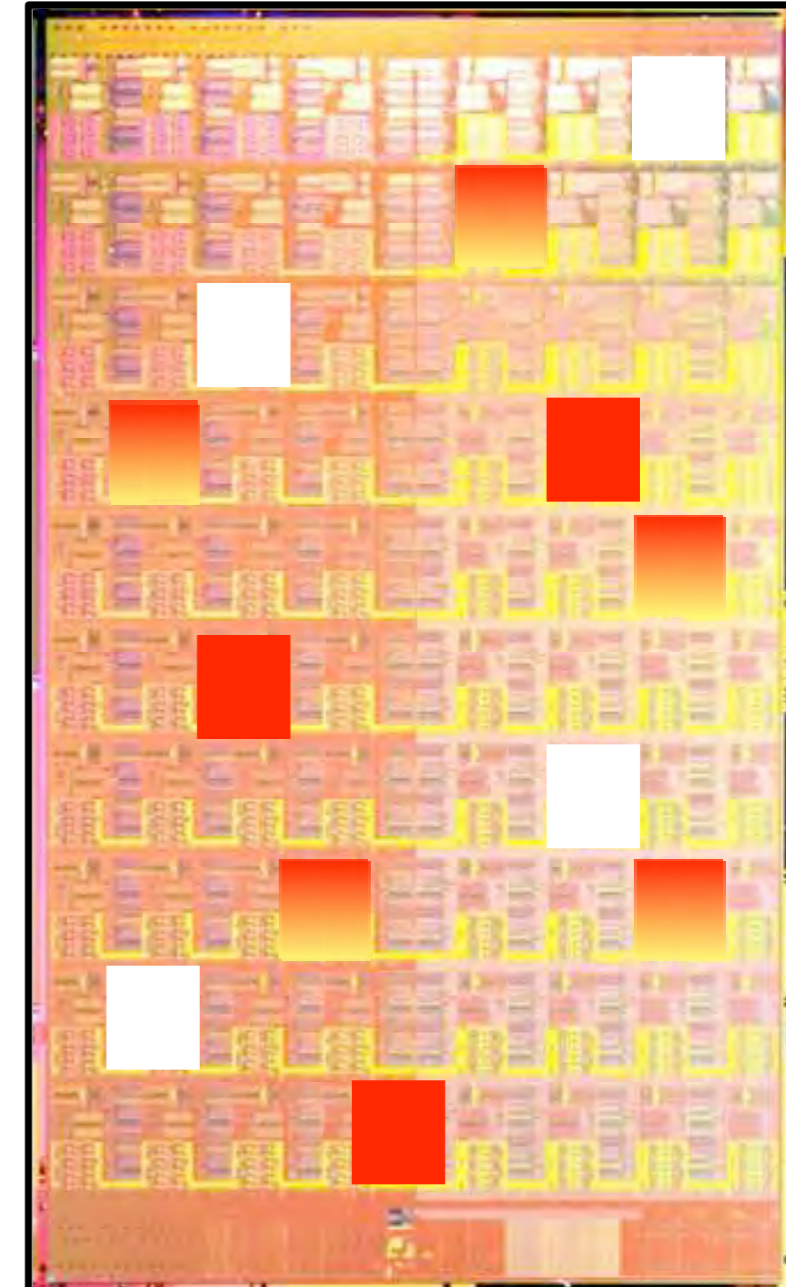
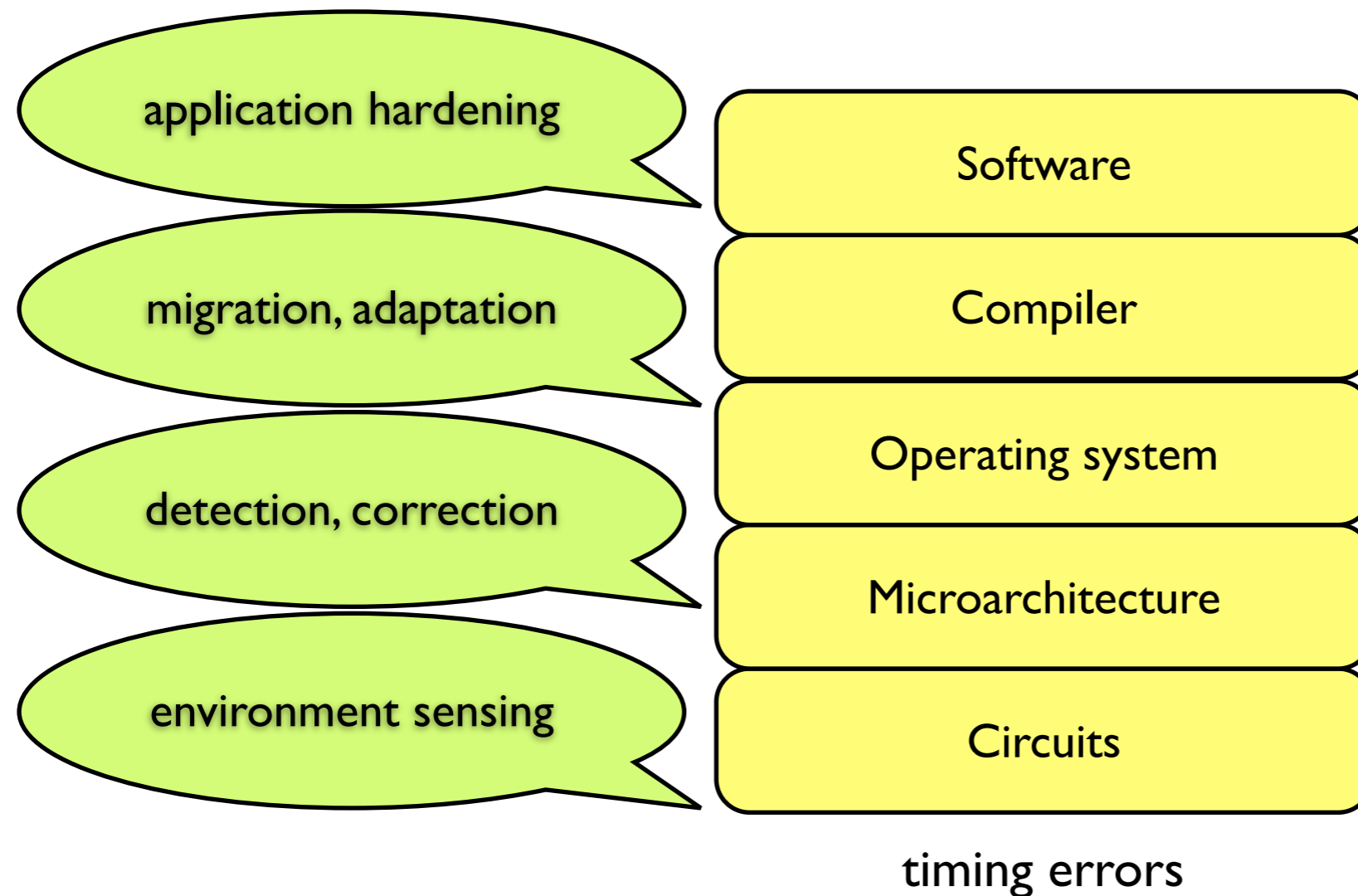
- Semiconductor roadmaps predict:
 - 11nm - 128 billion transistor chips
 - Hundreds of cores on a die
 - Reliability problems will get worse
 - some cores will fail immediately
 - others over time





Future work

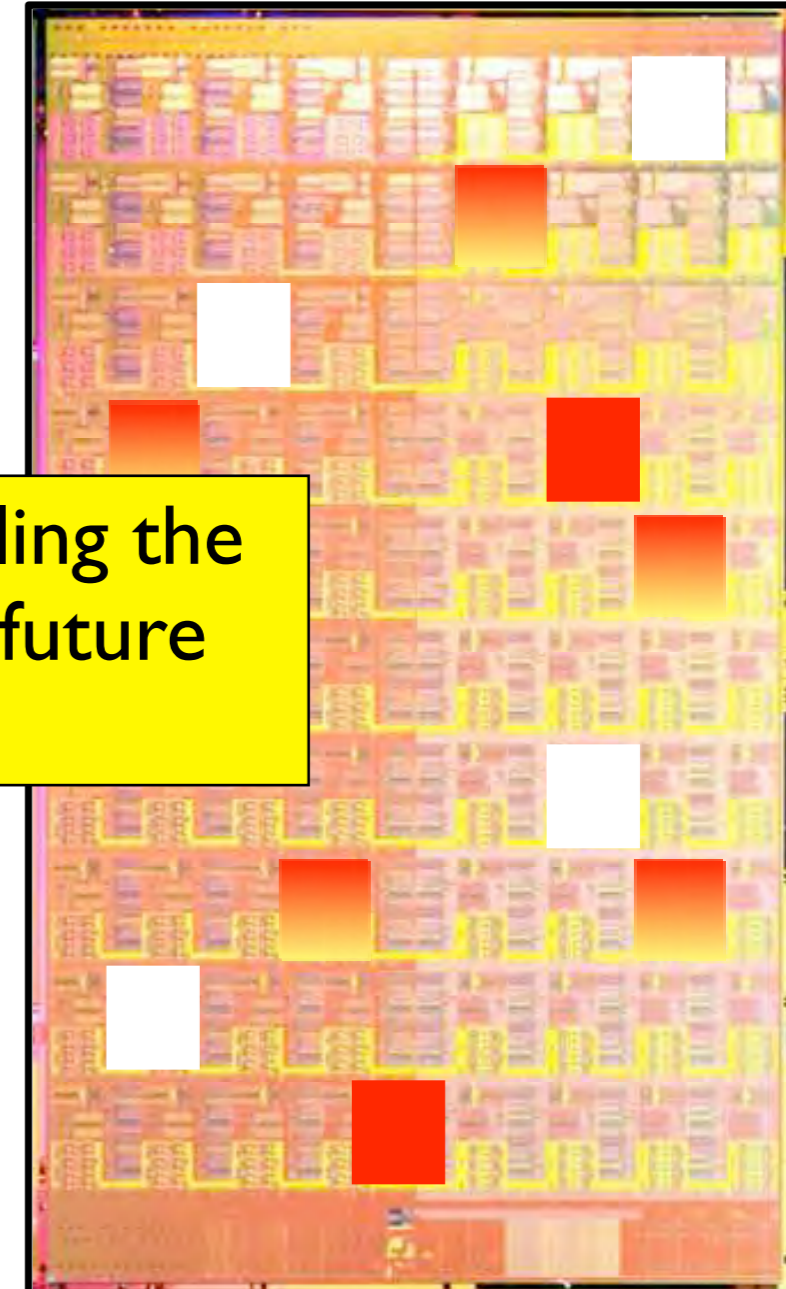
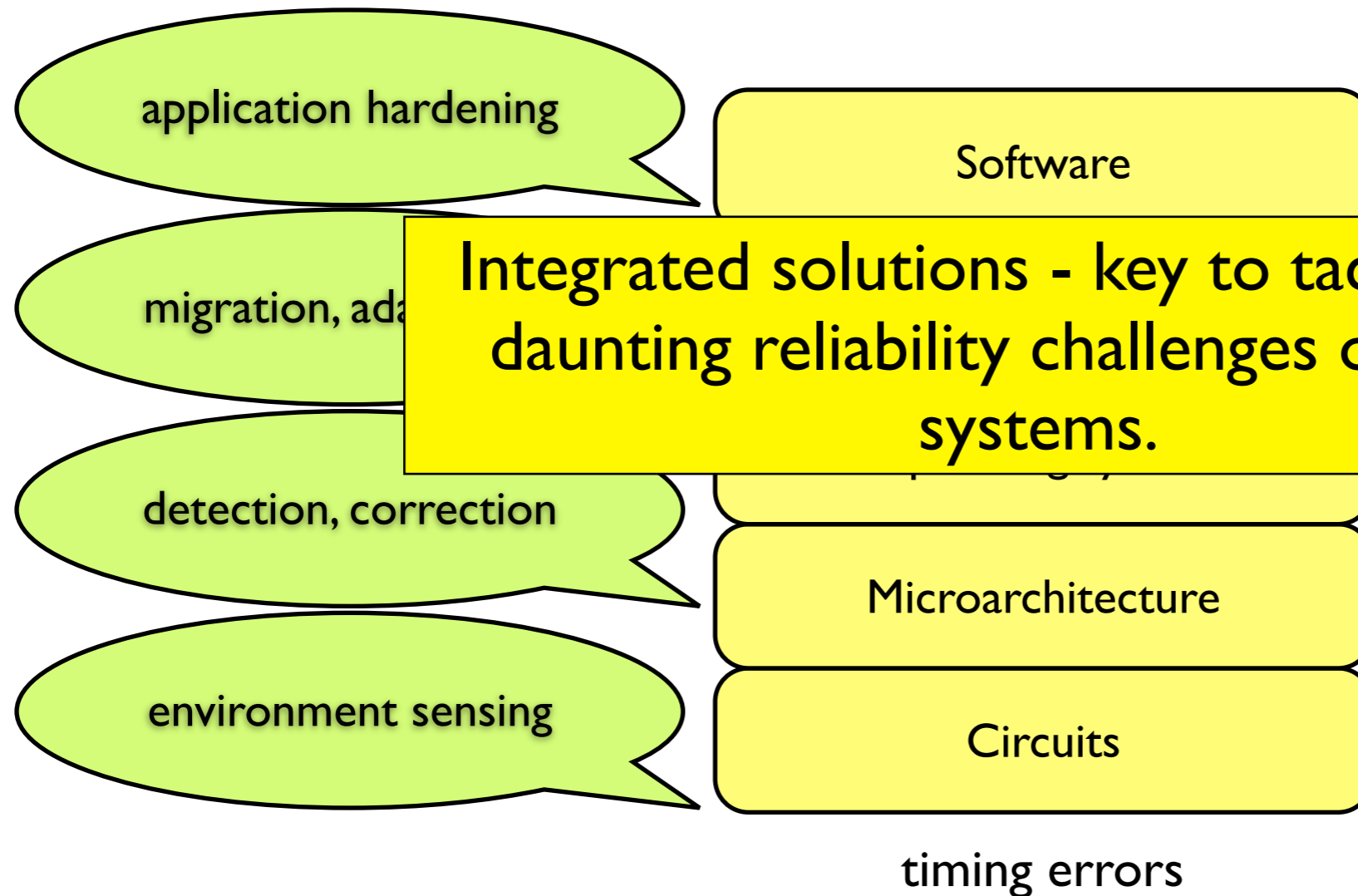
Integrated approach to system reliability





Future work

Integrated approach to system reliability





Other work

Hardware support for on-line software debugging

- Prototype of a processor with fast, software controlled checkpointing and rollback, in FPGA [FCCM'05][WCED'05][BUGS'05][Micro Magazine'06]
- Hardware implementation of a data race detection algorithm [HPCA'07]
- Log-based architectures for lightweight monitoring of production code [ASID'06]

Helping Moore's Law: Architectural Techniques to Address Parameter Variation

Radu Teodorescu

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