Software, Architecture, and VLSI Co-Design for Exploiting Fine-Grain Asymmetry

Christopher Torng

Thesis Defense

Computer Systems Laboratory
School of Electrical and Computer Engineering
Cornell University

Emerging Applications Demand Performance

AR / VR

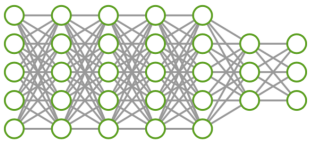
Autonomous Driving

Graph Analytics

Smart Home

Smart Healthcare

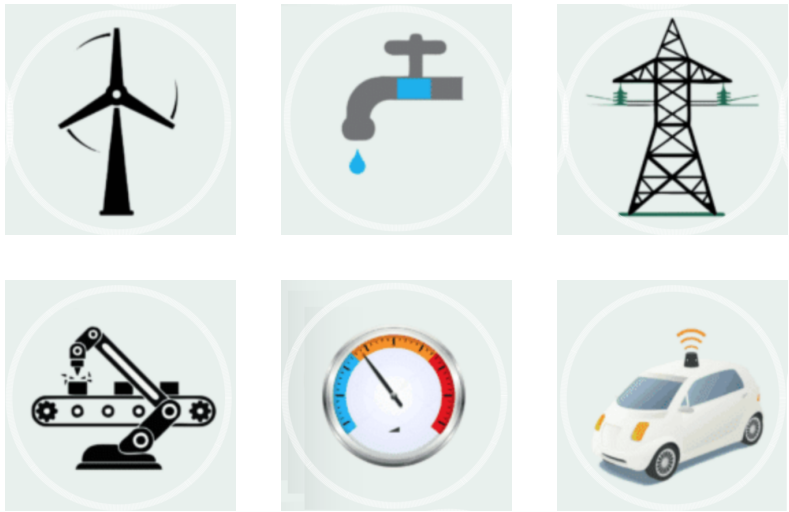
AI



Cybersecurity

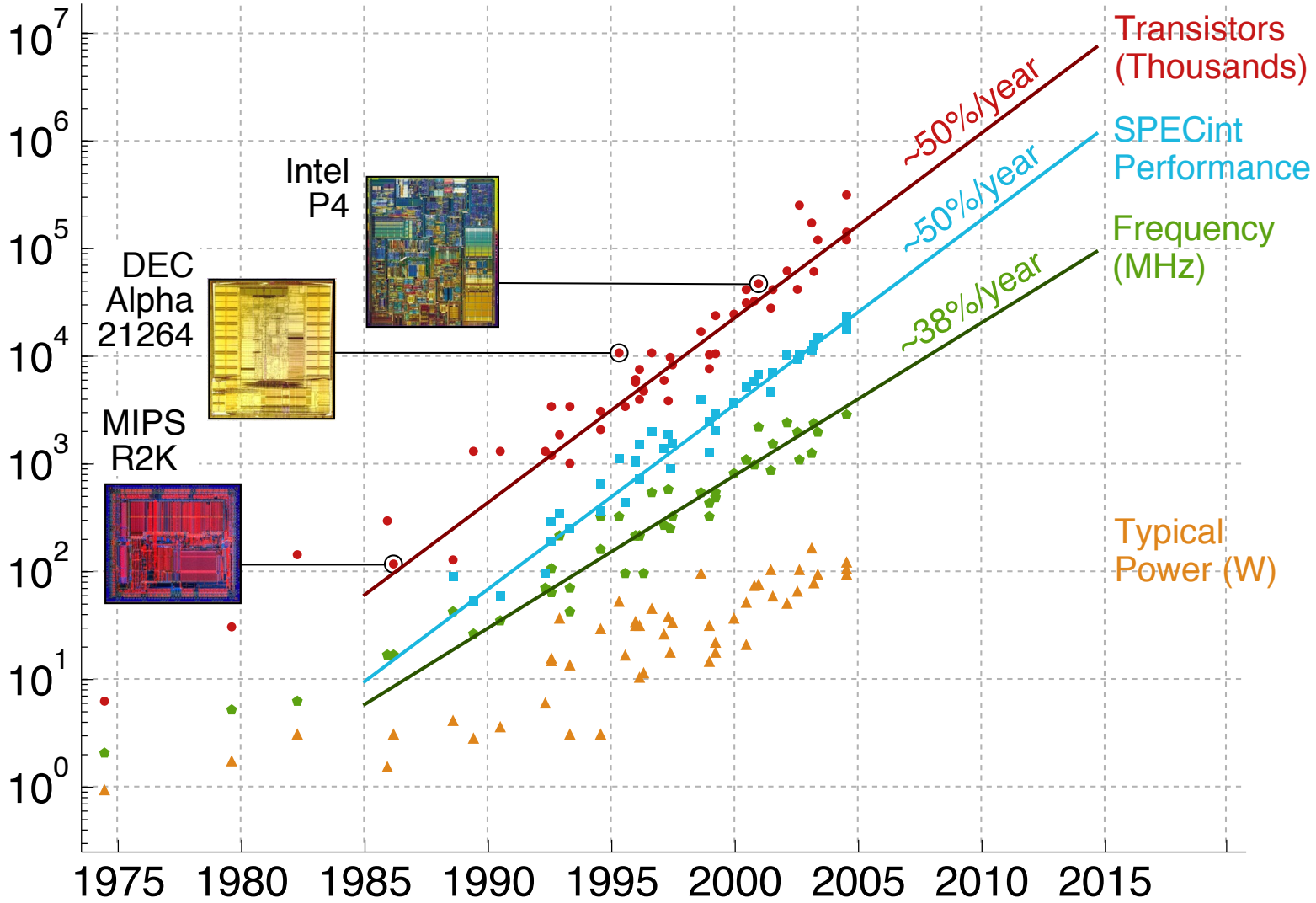
Intelligence on the Edge

Emerging domains have either promised or have already delivered real-world impact



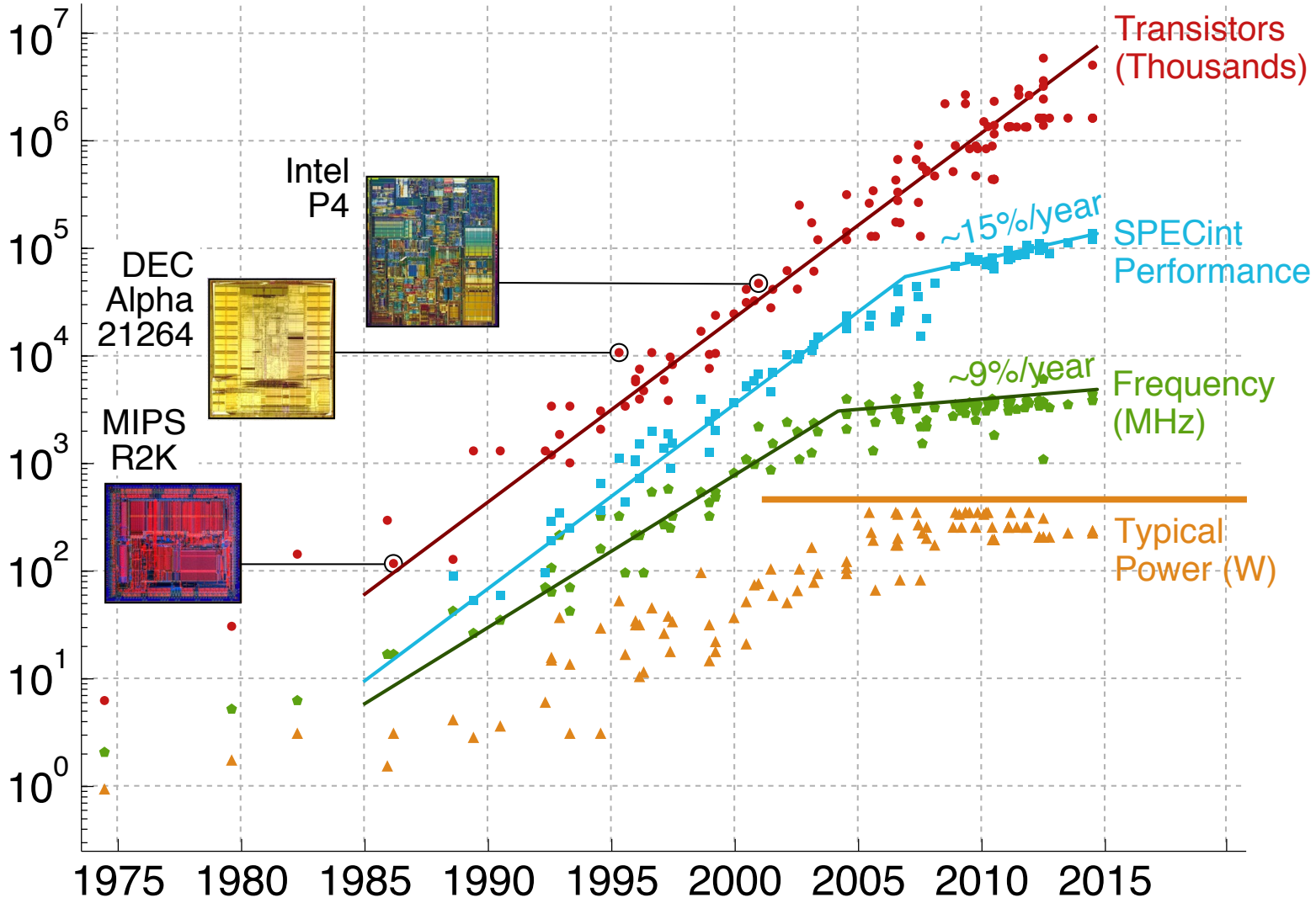
Source: Lanner

The Power Wall



Data collected by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, C. Batten

The Power Wall



Data collected by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, C. Batten

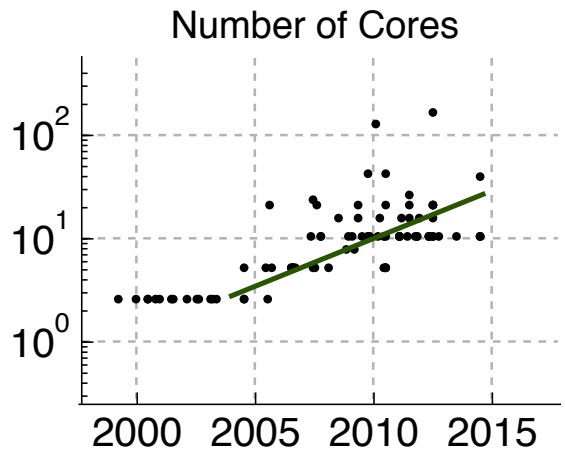
Parallelism and Specialization

Parallelism

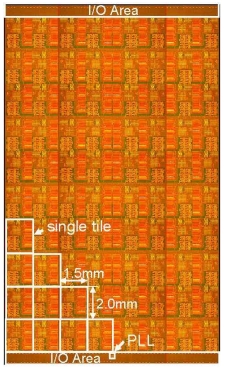
Specialization

Parallelism and Specialization

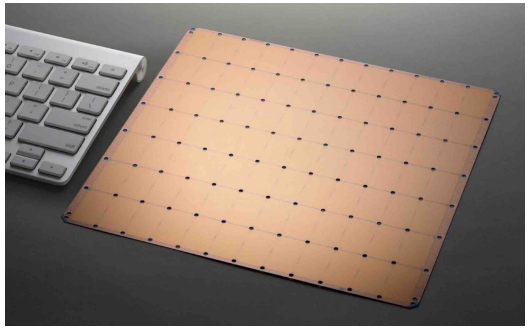
Parallelism



Data collected by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, C. Batten



Intel Polaris (2007)
21.7 mm x 12.6 mm
80 cores

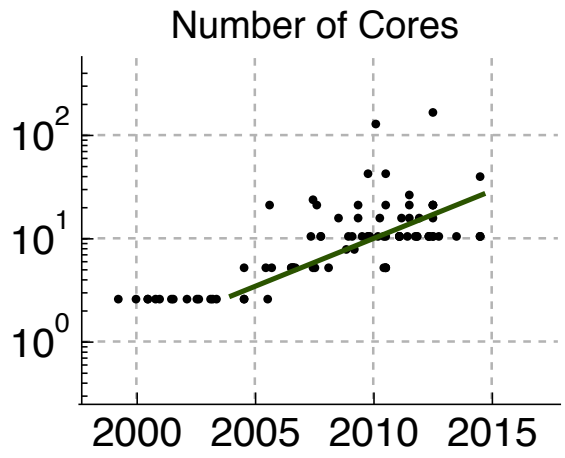


Cerebras Wafer-Scale Engine (2019)
215 mm x 215 mm
400,000 AI-Optimized Cores

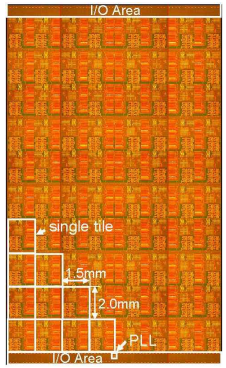
Specialization

Parallelism and Specialization

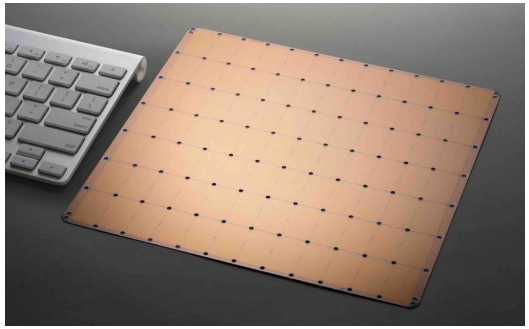
Parallelism



Data collected by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, C. Batten



Intel Polaris (2007)
21.7 mm x 12.6 mm
80 cores



Cerebras Wafer-Scale Engine (2019)
215 mm x 215 mm
400,000 AI-Optimized Cores

Specialization

General Purpose	Lane-Based Accelerators	CGRA-Based Accelerators	Domain-Specific Accelerators	Application-Specific Accelerators
Multicore Manycore	Packed-SIMD Vector Vector-thread	DySER Triggered Instructions	Convolution Engines CNN/DNN Accelerators GPGPU	Crypto / Compression Video encoders Video decoders

Flexible ⟷ Efficient

Parallelism and Specialization

Parallelism

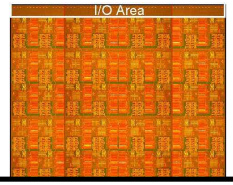
Number of Cores

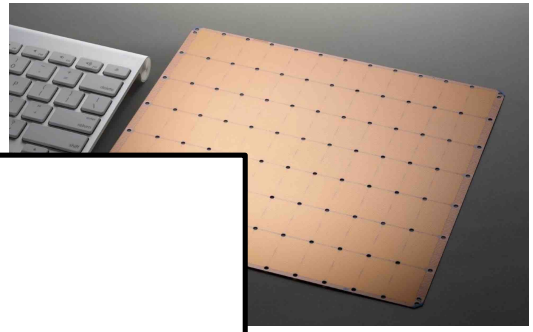
10^2

10^1

10^0

Data colle





Scale Engine (2019)
x 215 mm
Optimized Cores

Key Observation

The combination of parallelism and specialization is steadily increasing *on-chip asymmetry*, which motivates *fine-grain power-control techniques*

Specialization

Gene
Purpo

Multic
Manycore

Vector
Vector-thread

Triggered Instructions

CNN/DNN Accelerators
GPGPU

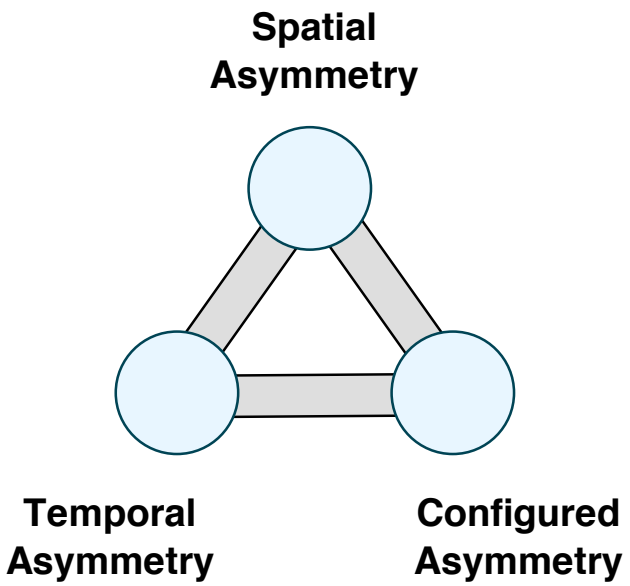
plication-Specific
Accelerators

ypto / Compression
Video encoders
Video decoders

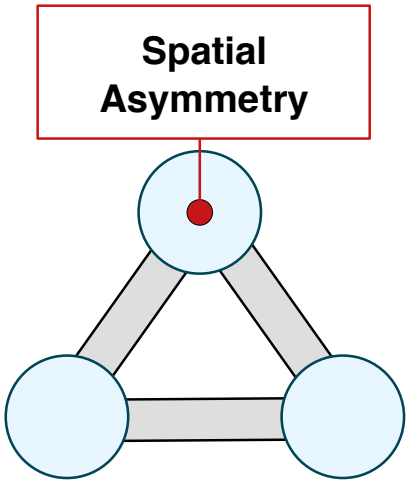
Flexible

Efficient

A Trend of Increasing On-Chip Asymmetry



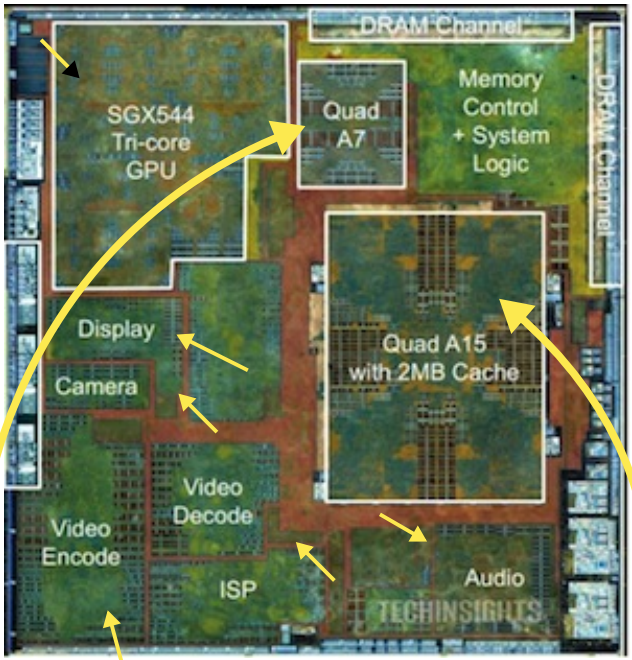
A Trend of Increasing On-Chip Asymmetry



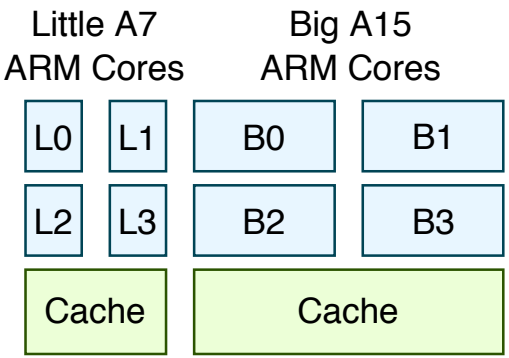
Temporal Asymmetry

Configured Asymmetry

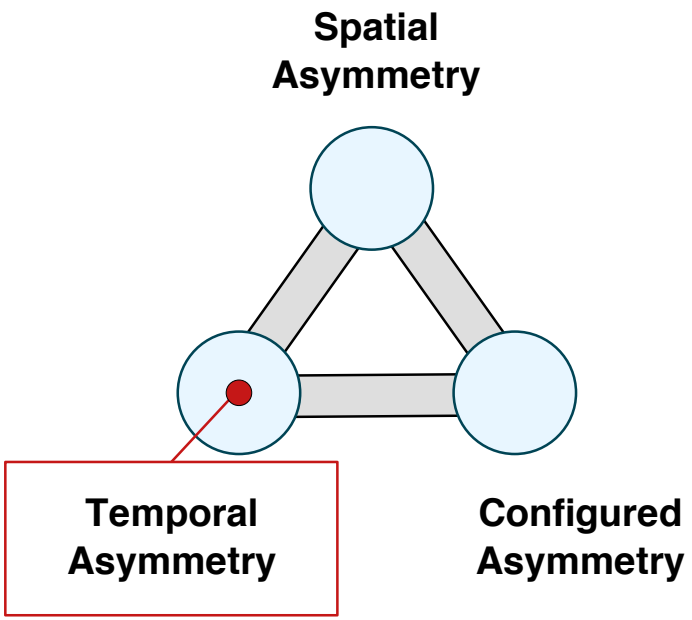
Spatial Asymmetry - Varies across chip

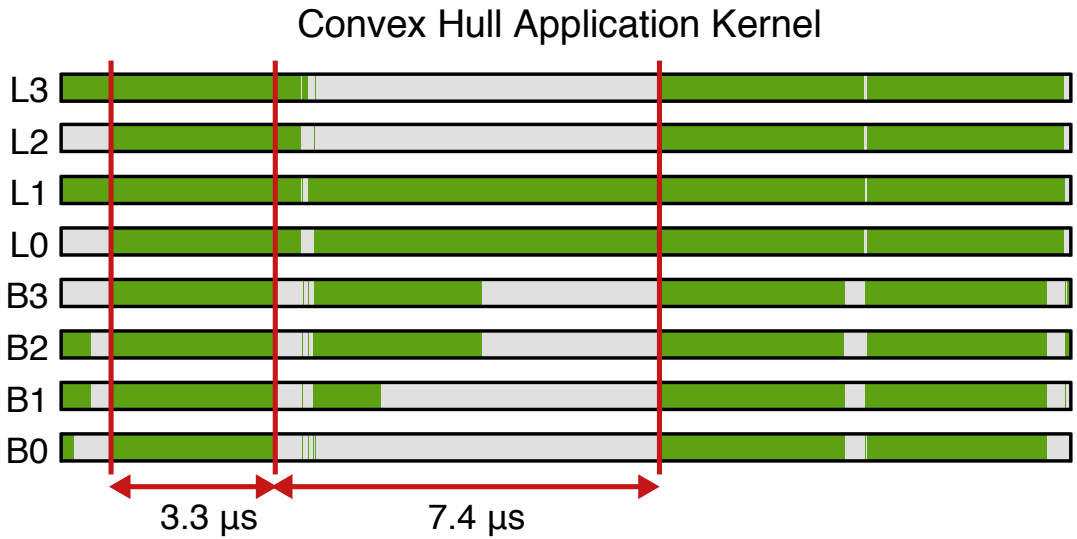


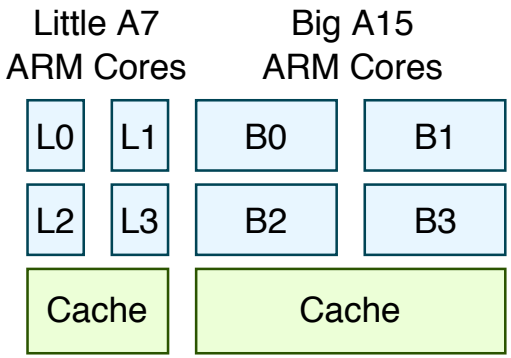
Samsung Exynos Octacore



A Trend of Increasing On-Chip Asymmetry





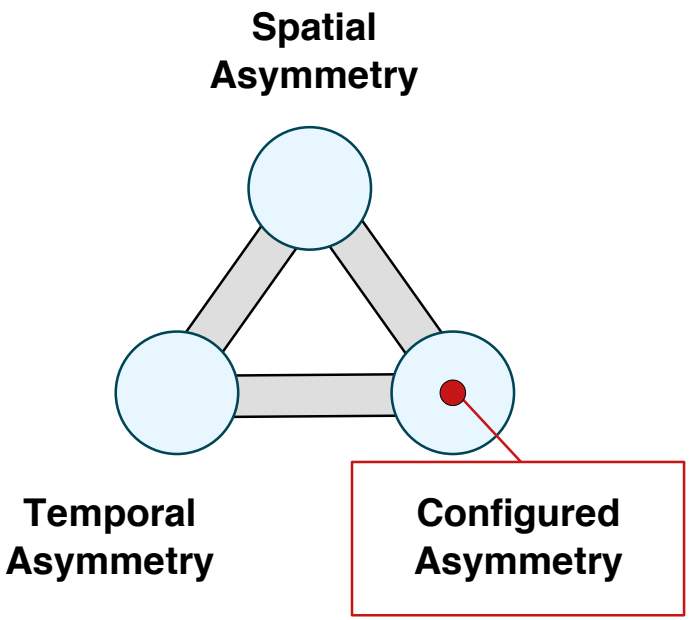


Spatial Asymmetry - Varies across chip

Temporal Asymmetry - Varies at runtime

See also: Kim et al. "System Level Analysis of Fast, Per-Core DVFS" HPCA 2008

A Trend of Increasing On-Chip Asymmetry

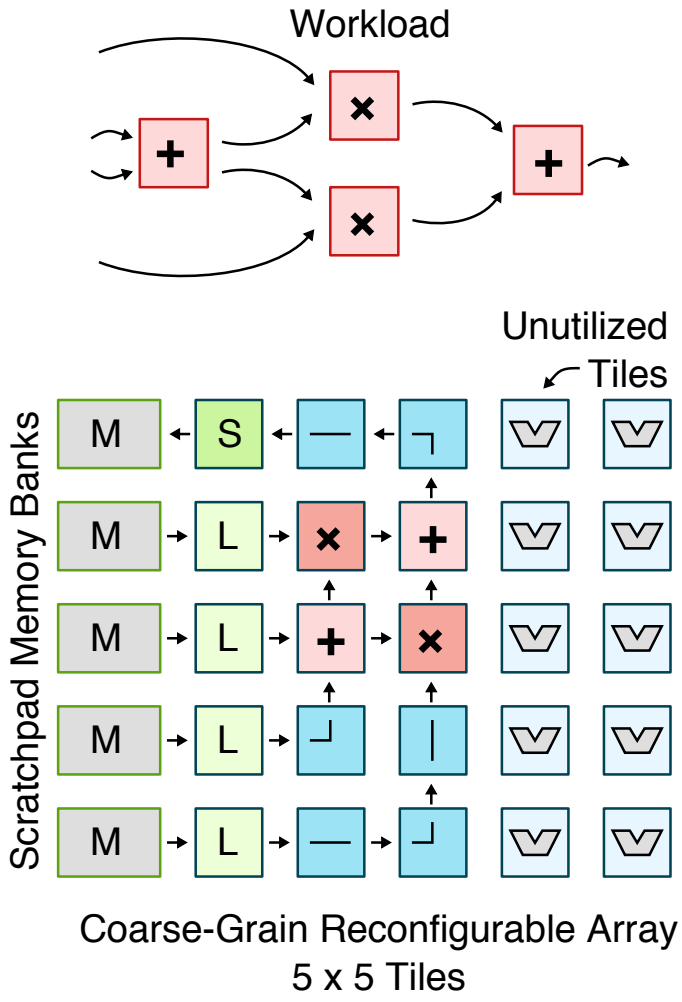


Spatial Asymmetry - Varies across chip

Temporal Asymmetry - Varies at runtime

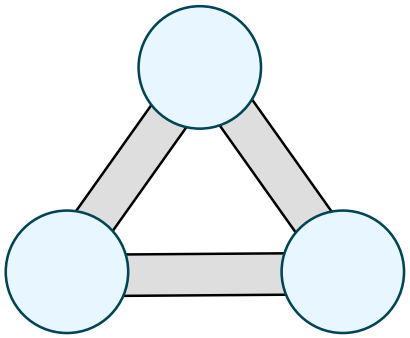
Configured Asymmetry - Set at configure time

Limited Reconfiguration - Similar to purely spatial asymmetry
Rapid Reconfiguration - Increases temporal asymmetry



See also: Mei et al. "ADRES", FPL 2003

A Trend of Increasing On-Chip Asymmetry



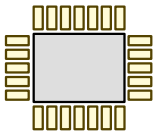
Increasing on-chip asymmetry motivates
fine-grain power-control techniques

Spatial Asymmetry - Varies across chip

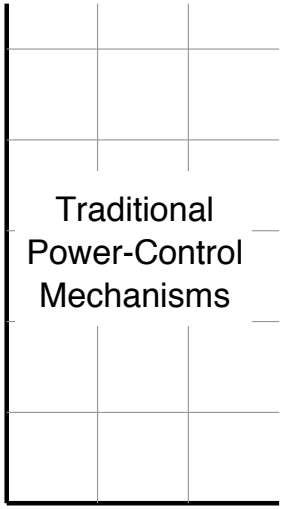
Temporal Asymmetry - Varies at runtime

Configured Asymmetry - Set at configure time

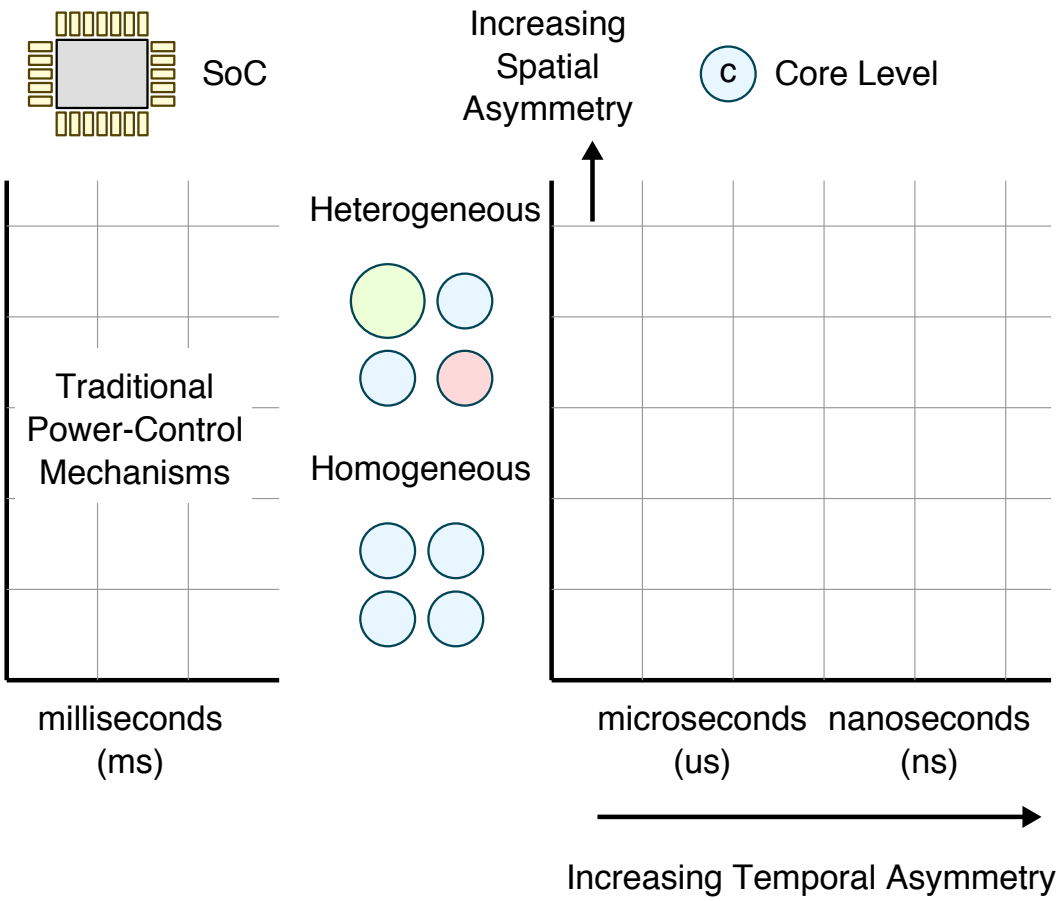
Motivating Fine-Grain Power Control



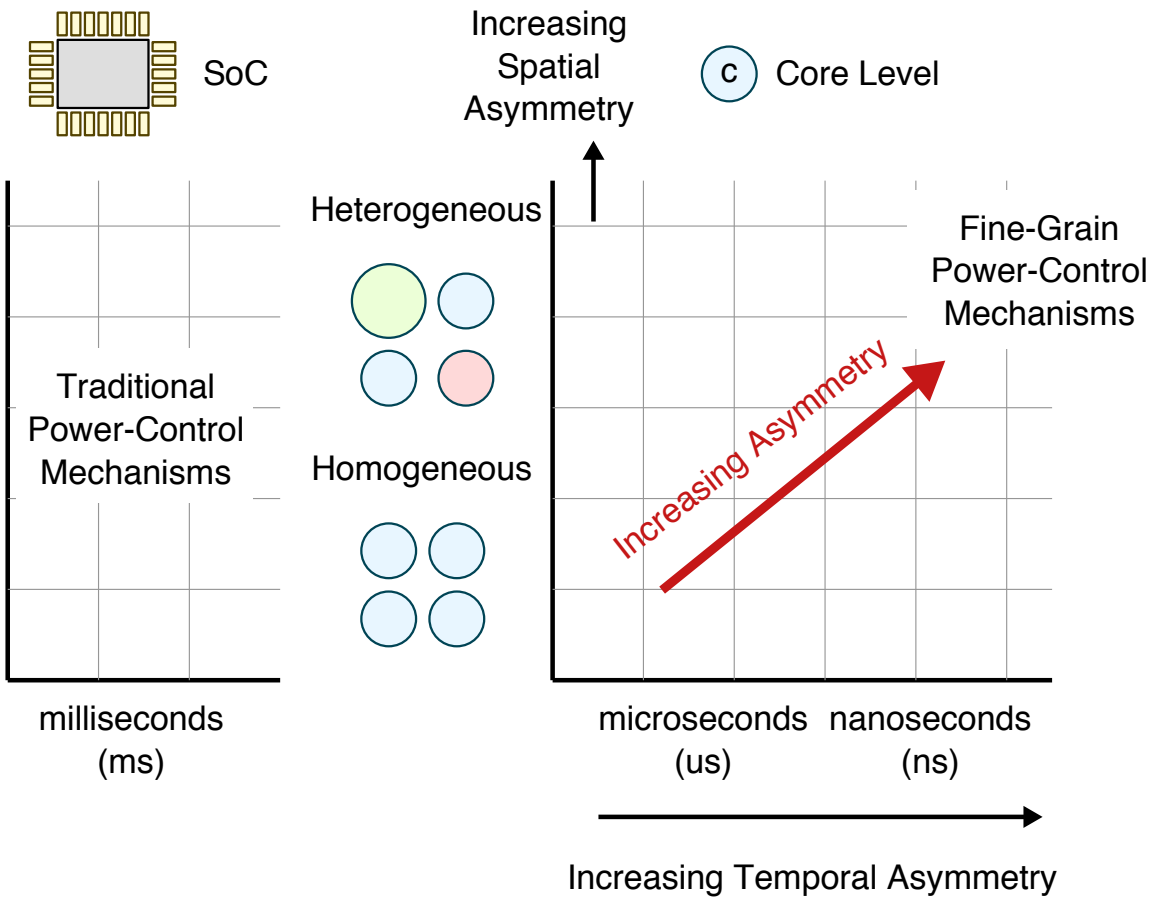
SoC



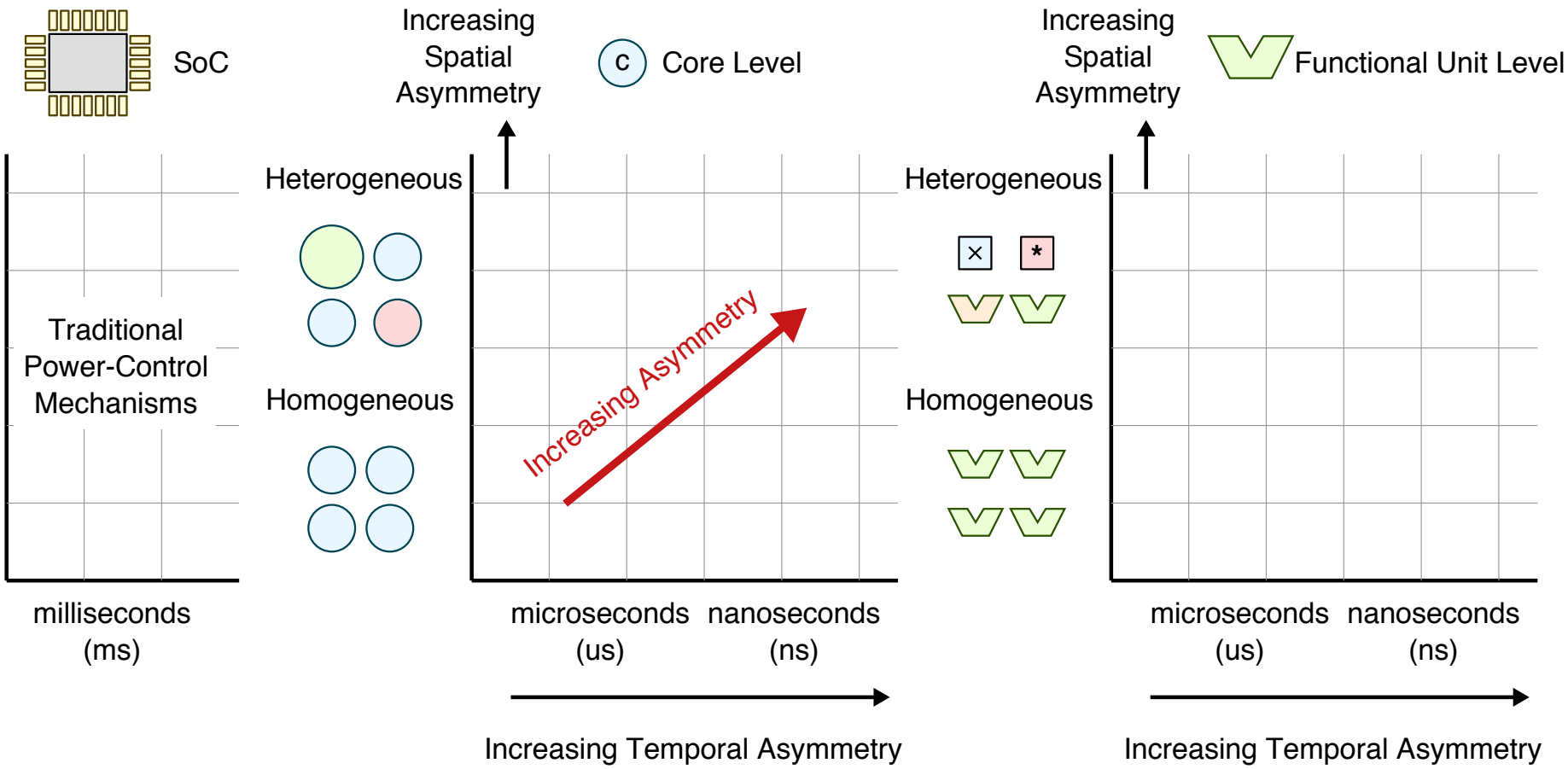
Motivating Fine-Grain Power Control



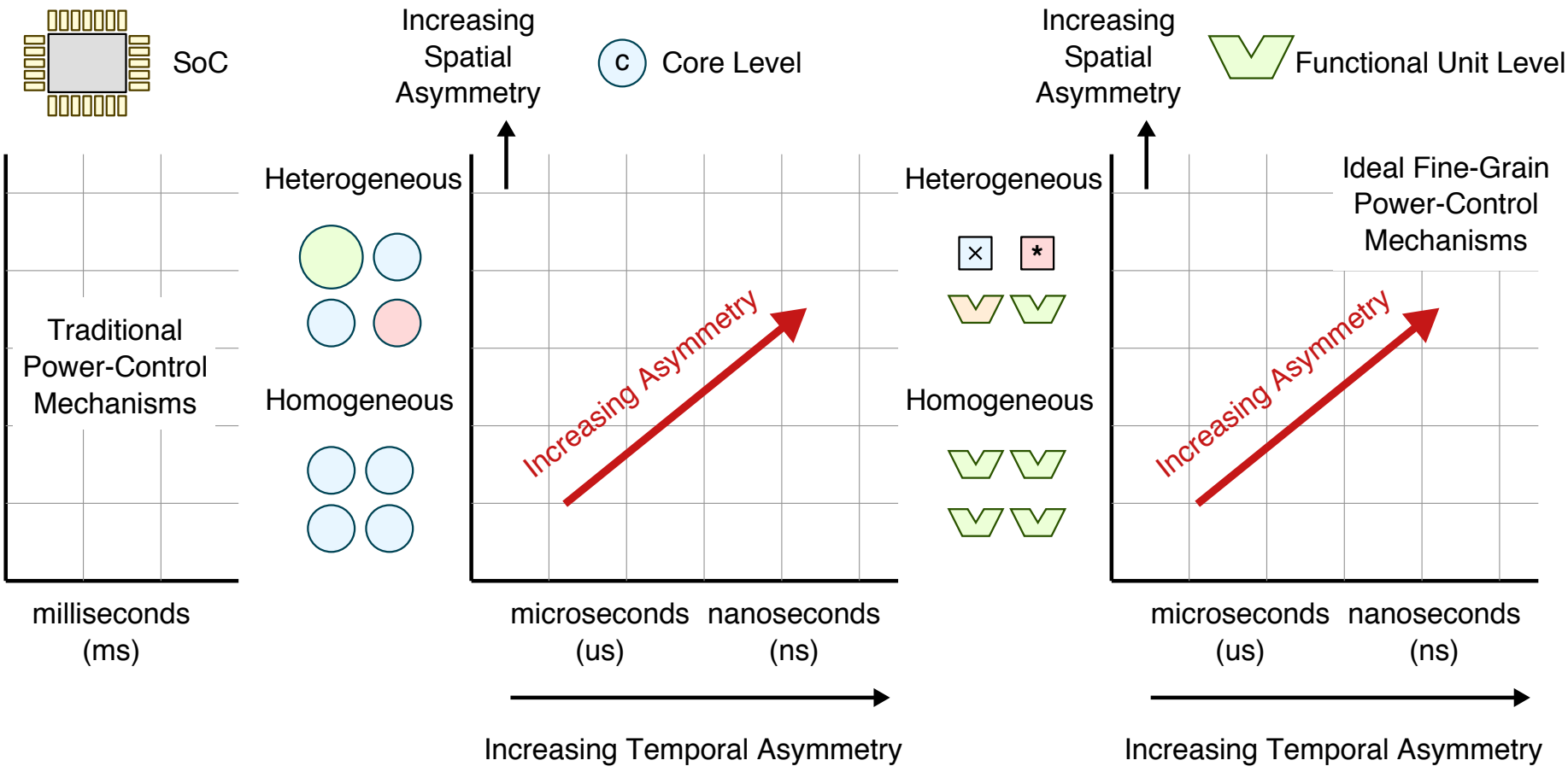
Motivating Fine-Grain Power Control



Motivating Fine-Grain Power Control

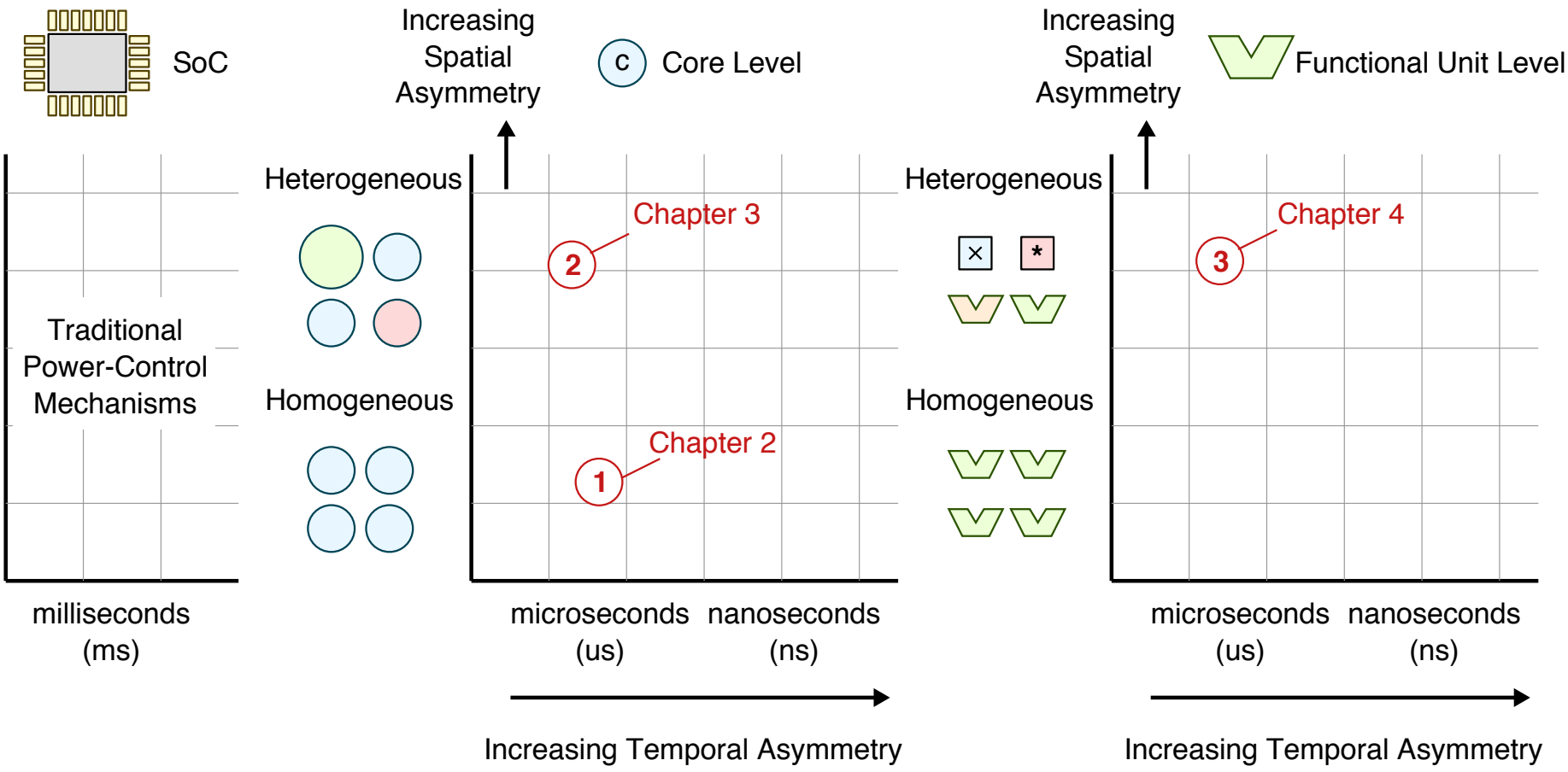


Motivating Fine-Grain Power Control



Caveat: Balance flexibility against overheads

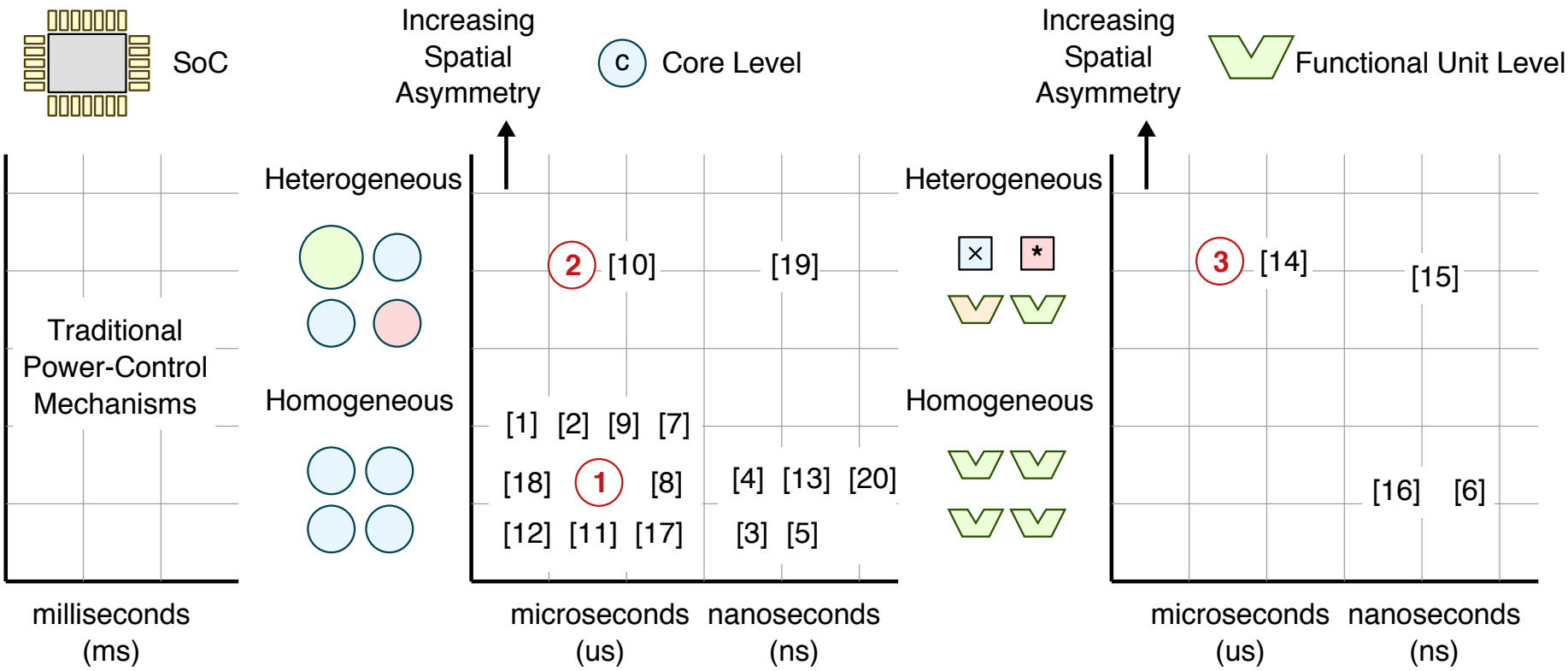
Motivating Fine-Grain Power Control



Thesis Goal

Explore a range of fine-grain power-control techniques across different application domains at both the core- and functional-unit levels

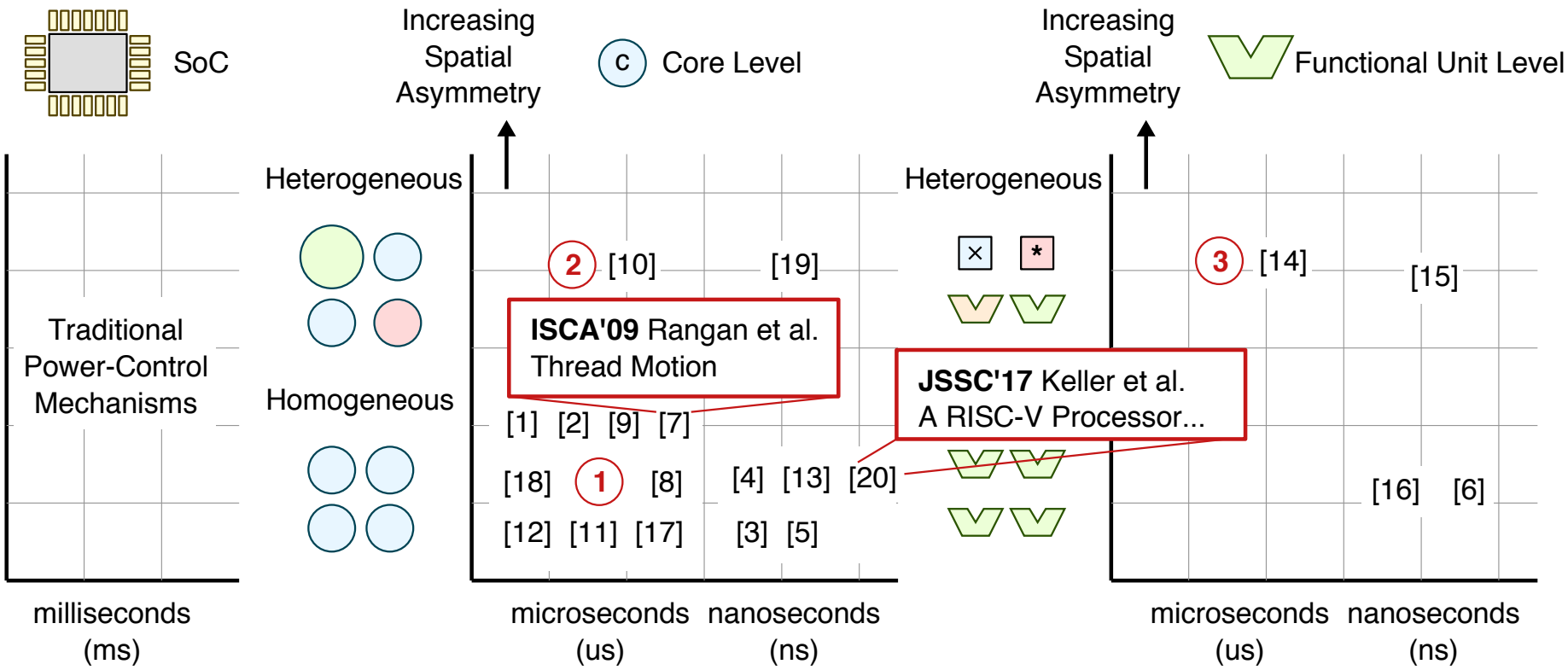
Motivating Fine-Grain Power Control



Selected Works Assuming Fine-Grain Power Control

[1] ISCA'09
[2] HPCA'08
[3] JSSC'09
[4] MICRO'03
[5] PROCIEEE'10
[6] JSSC'06
[7] ISCA'09
[8] ISCA'06
[9] DAC'09
[10] ISCA'05
[11] IEEEMICRO'13
[12] HPCA'14
[13] HPCA'12
[14] MICRO'02
[15] SAMOS'13
[16] ESSCIRC'11
[17] ASPLOS'14
[18] HPCA'15
[19] PACT'14
[20] JSSC'17

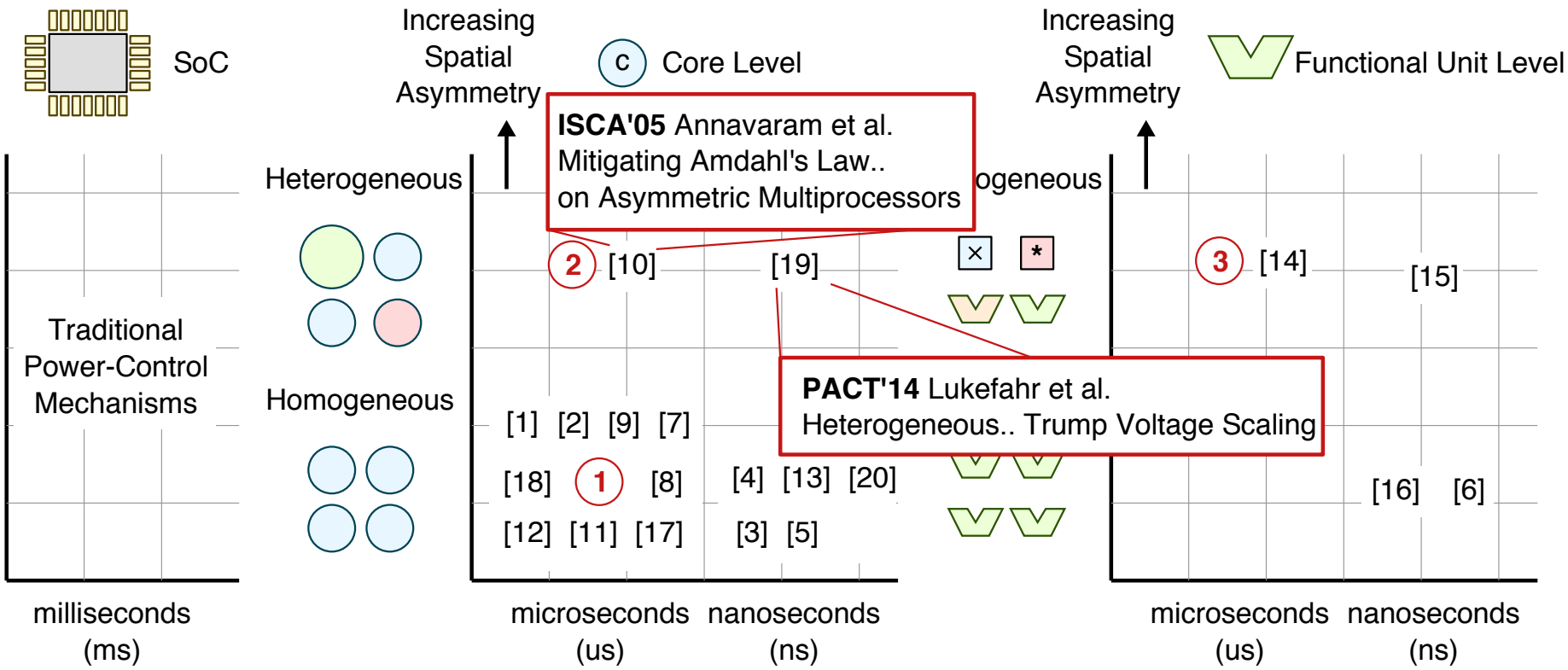
Motivating Fine-Grain Power Control



Selected Works Assuming Fine-Grain Power Control

[1] ISCA'09
[2] HPCA'08
[3] JSSC'09
[4] MICRO'03
[5] PROCIEEE'10
[6] JSSC'06
[7] ISCA'09
[8] ISCA'06
[9] DAC'09
[10] ISCA'05
[11] IEEEMICRO'13
[12] HPCA'14
[13] HPCA'12
[14] MICRO'02
[15] SAMOS'13
[16] ESSCIRC'11
[17] ASPLOS'14
[18] HPCA'15
[19] PACT'14
[20] JSSC'17

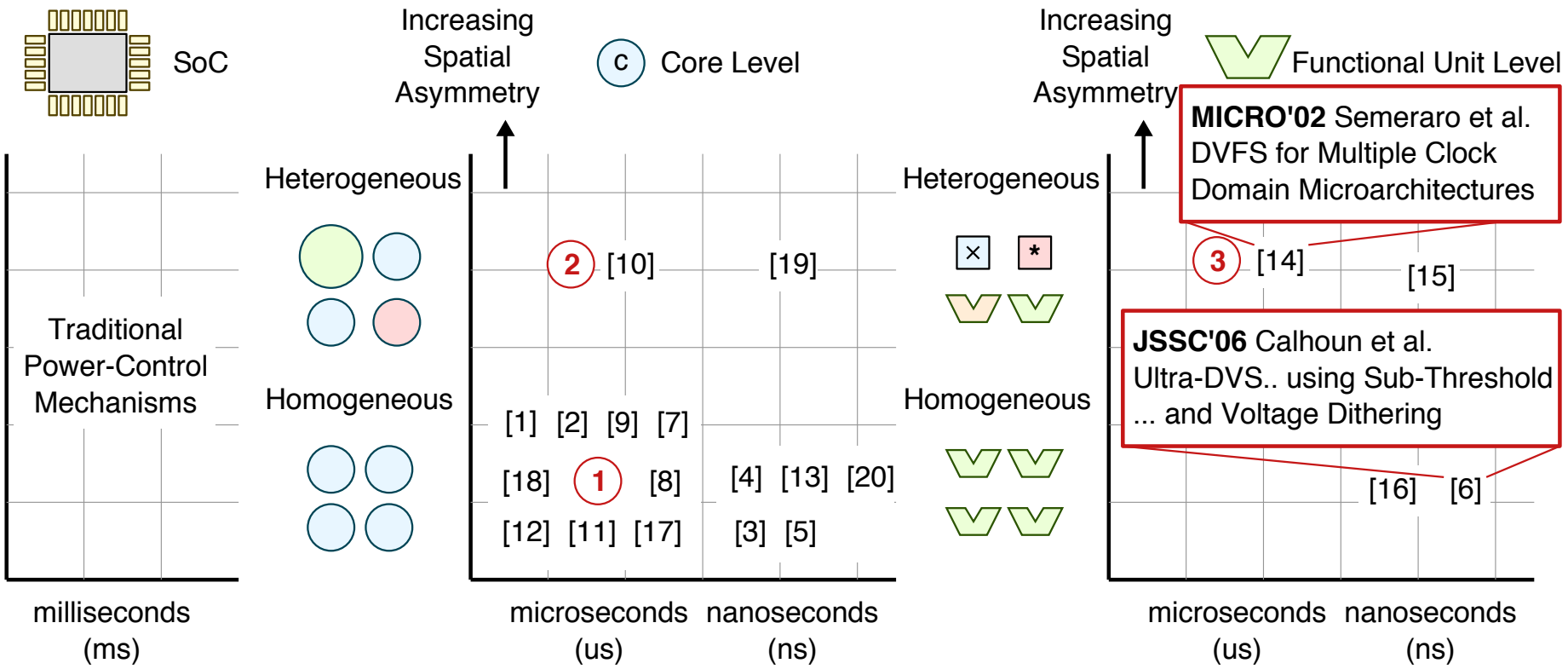
Motivating Fine-Grain Power Control



Selected Works Assuming Fine-Grain Power Control

[1] ISCA'09
[2] HPCA'08
[3] JSSC'09
[4] MICRO'03
[5] PROCIEEE'10
[6] JSSC'06
[7] ISCA'09
[8] ISCA'06
[9] DAC'09
[10] ISCA'05
[11] IEEEMICRO'13
[12] HPCA'14
[13] HPCA'12
[14] MICRO'02
[15] SAMOS'13
[16] ESSCIRC'11
[17] ASPLOS'14
[18] HPCA'15
[19] PACT'14
[20] JSSC'17

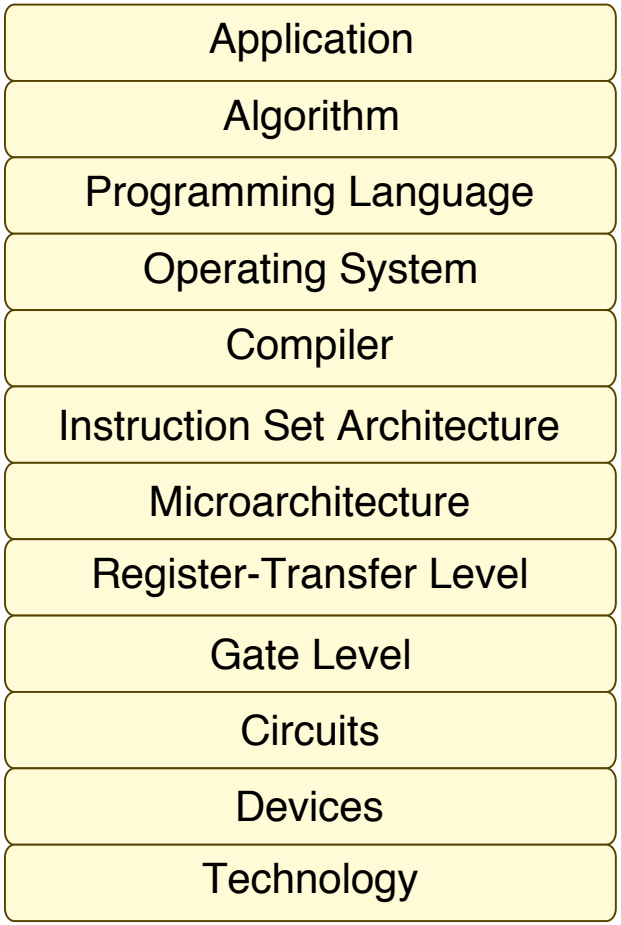
Motivating Fine-Grain Power Control



Selected Works Assuming Fine-Grain Power Control

[1] ISCA'09
[2] HPCA'08
[3] JSSC'09
[4] MICRO'03
[5] PROCIEEE'10
[6] JSSC'06
[7] ISCA'09
[8] ISCA'06
[9] DAC'09
[10] ISCA'05
[11] IEEEMICRO'13
[12] HPCA'14
[13] HPCA'12
[14] MICRO'02
[15] SAMOS'13
[16] ESSCIRC'11
[17] ASPLOS'14
[18] HPCA'15
[19] PACT'14
[20] JSSC'17

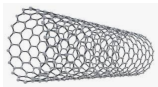
Broken Walls of Abstraction



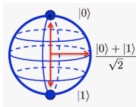
Broken Walls of Abstraction

- Application
- Algorithm
- Programming Language
- Operating System
- Compiler
- Instruction Set Architecture
- Microarchitecture
- Register-Transfer Level
- Gate Level
- Circuits
- Devices
- Technology

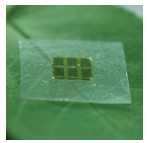
Carbon Nanotubes



Quantum Computing

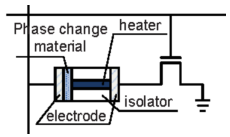


Biodegradable Computing

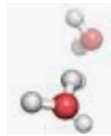


Energy Harvesting

Phase-Change Memory



Molecular Computing



Broken Walls of Abstraction

- Application
- Algorithm
- Programming Language
- Operating System
- Compiler
- Instruction Set Architecture
- Microarchitecture
- Register-Transfer Level
- Gate Level
- Circuits
- Devices
- Technology

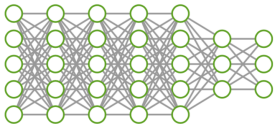
AR / VR

Autonomous Driving

Smart Home

Graph Analytics

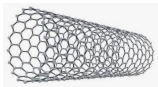
AI



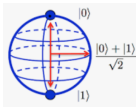
Cybersecurity

Smart Healthcare

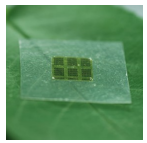
Carbon Nanotubes



Quantum Computing

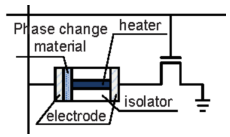


Biodegradable Computing

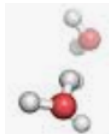


Energy Harvesting

Phase-Change Memory



Molecular Computing



Broken Walls of Abstraction

- Application
- Algorithm
- Programming Language
- Operating System
- Compiler
- Instruction Set Architecture
- Microarchitecture
- Register-Transfer Level
- Gate Level
- Circuits
- Devices
- Technology

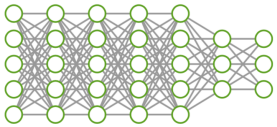
AR / VR

Autonomous Driving

Smart Home

Graph Analytics

AI

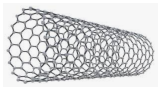


Cybersecurity

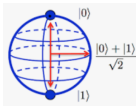
Smart Healthcare

Computer Architecture

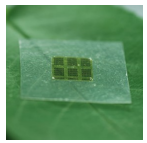
Carbon Nanotubes



Quantum Computing

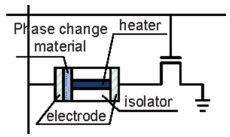


Biodegradable Computing

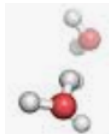


Energy Harvesting

Phase-Change Memory



Molecular Computing



Broken Walls of Abstraction

Application
Algorithm
Programming Language
Operating System
Compiler
Instruction Set Architecture
Microarchitecture
Register-Transfer Level
Gate Level
Circuits
Devices
Technology

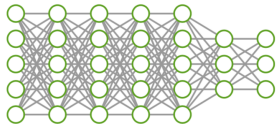
AR / VR

Autonomous Driving

Smart Home

Graph Analytics

AI

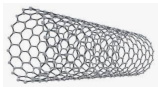


Cybersecurity

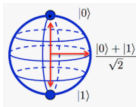
Smart Healthcare

Computer Architecture

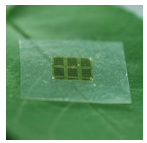
Carbon Nanotubes



Quantum Computing

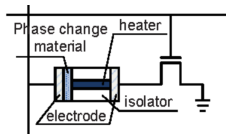


Biodegradable Computing

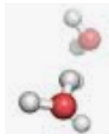


Energy Harvesting

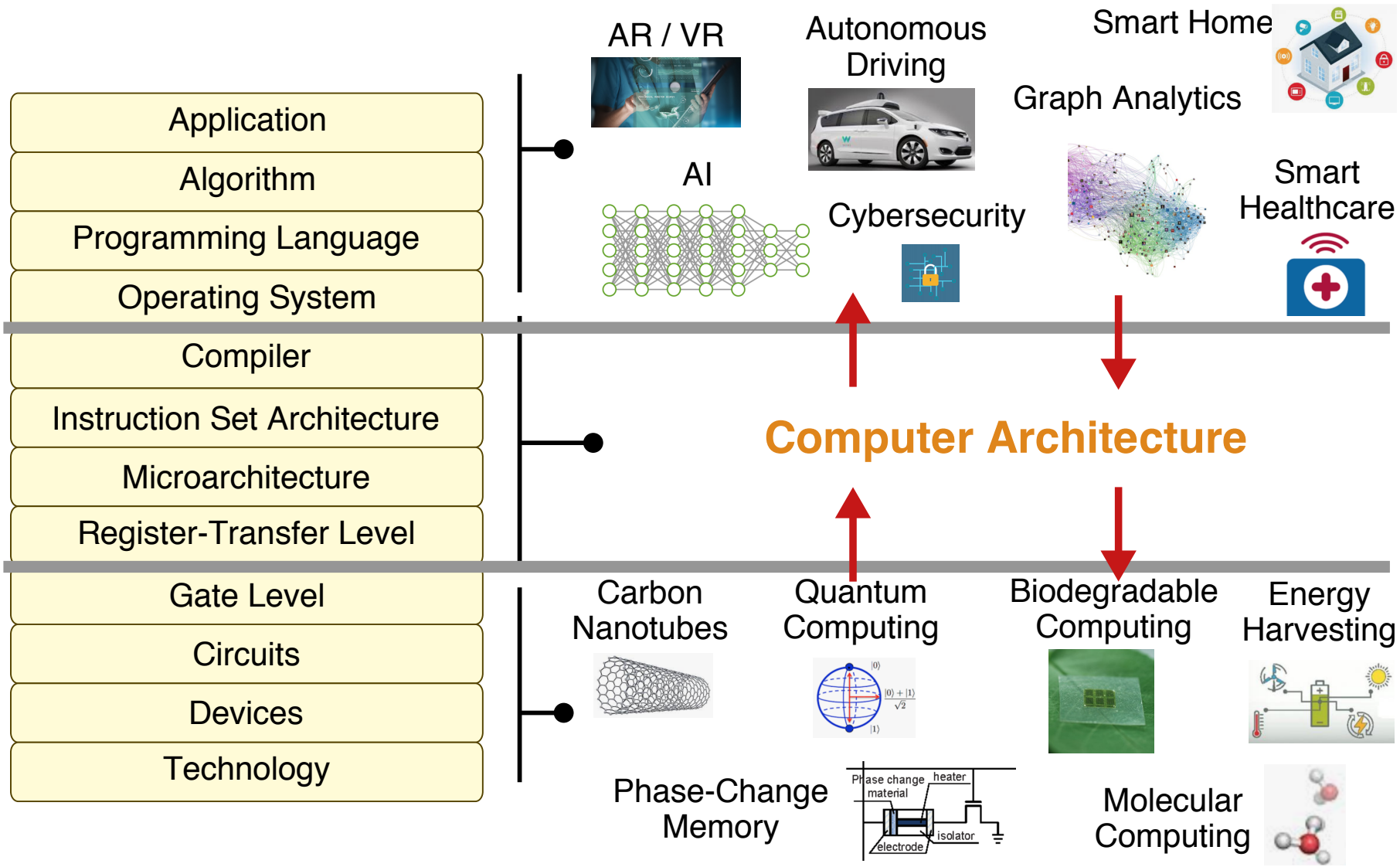
Phase-Change Memory



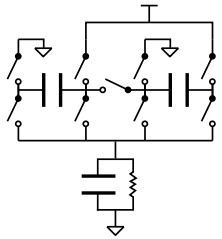
Molecular Computing



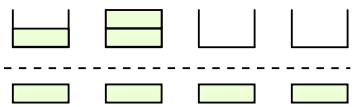
Broken Walls of Abstraction



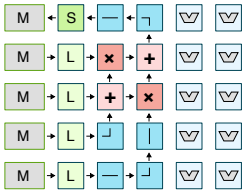
Software, Architecture, and VLSI Co-Design for Exploiting Fine-Grain Asymmetry



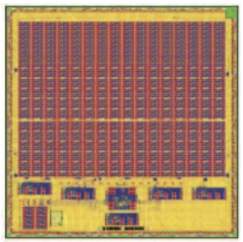
Exploiting Fine-Grain Asymmetry with
On-Chip Voltage Regulation - **MICRO'14**, IEEE TCAS I'18



Exploiting Fine-Grain Asymmetry in
Task-Based Parallel Runtimes - **ISCA'16**, MICRO'17, RISCV'18



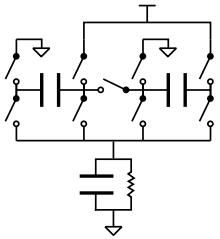
Exploiting Fine-Grain Asymmetry in
Coarse-Grain Reconfigurable Arrays - **Unpublished**



Exploiting Fine-Grain Asymmetry with
Silicon Prototyping - **IEEE MICRO'18**, DAC'18, Hotchips'17

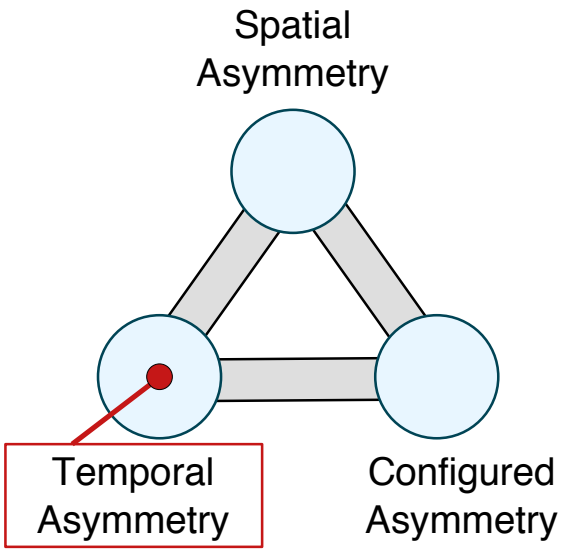
Conclusion

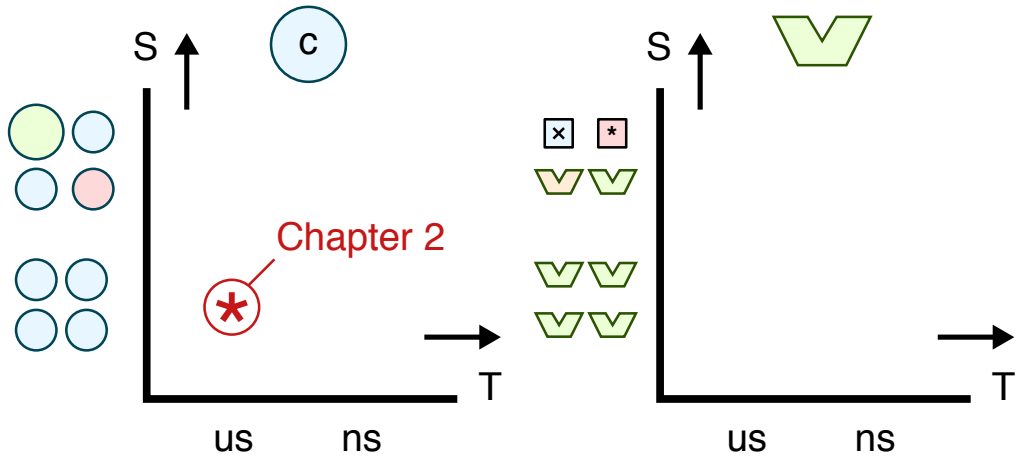
Software, Architecture, and VLSI Co-Design for Exploiting Fine-Grain Asymmetry



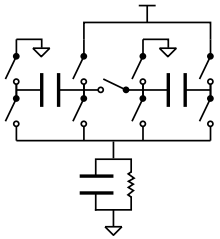
Exploiting Fine-Grain Asymmetry with

On-Chip Voltage Regulation - MICRO'14, IEEE TCAS I'18



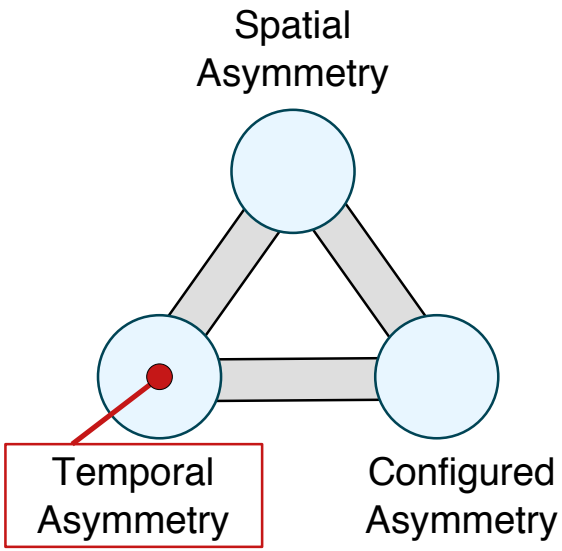


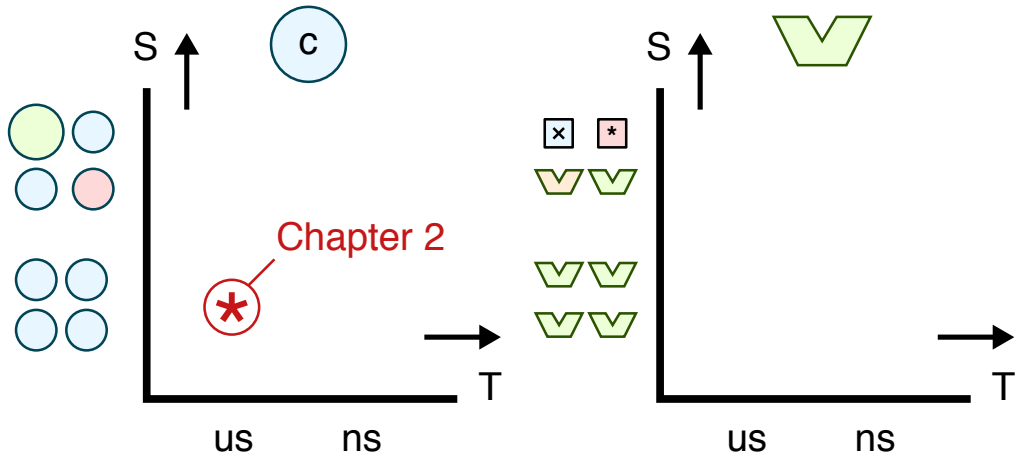
Software, Architecture, and VLSI Co-Design for Exploiting Fine-Grain Asymmetry



Exploiting Fine-Grain Asymmetry with

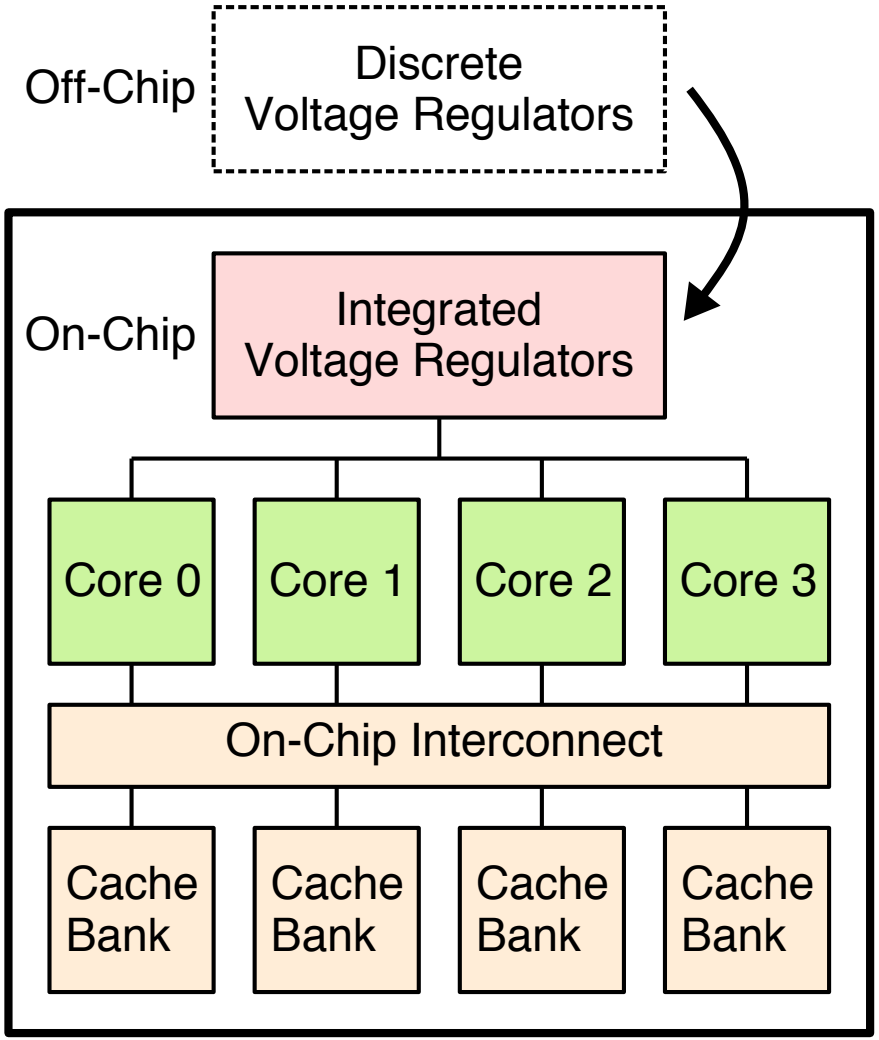
On-Chip Voltage Regulation - MICRO'14, IEEE TCAS I'18





Research Question – Can we use architecture-circuit co-design to enable more efficient fine-grain voltage regulation?

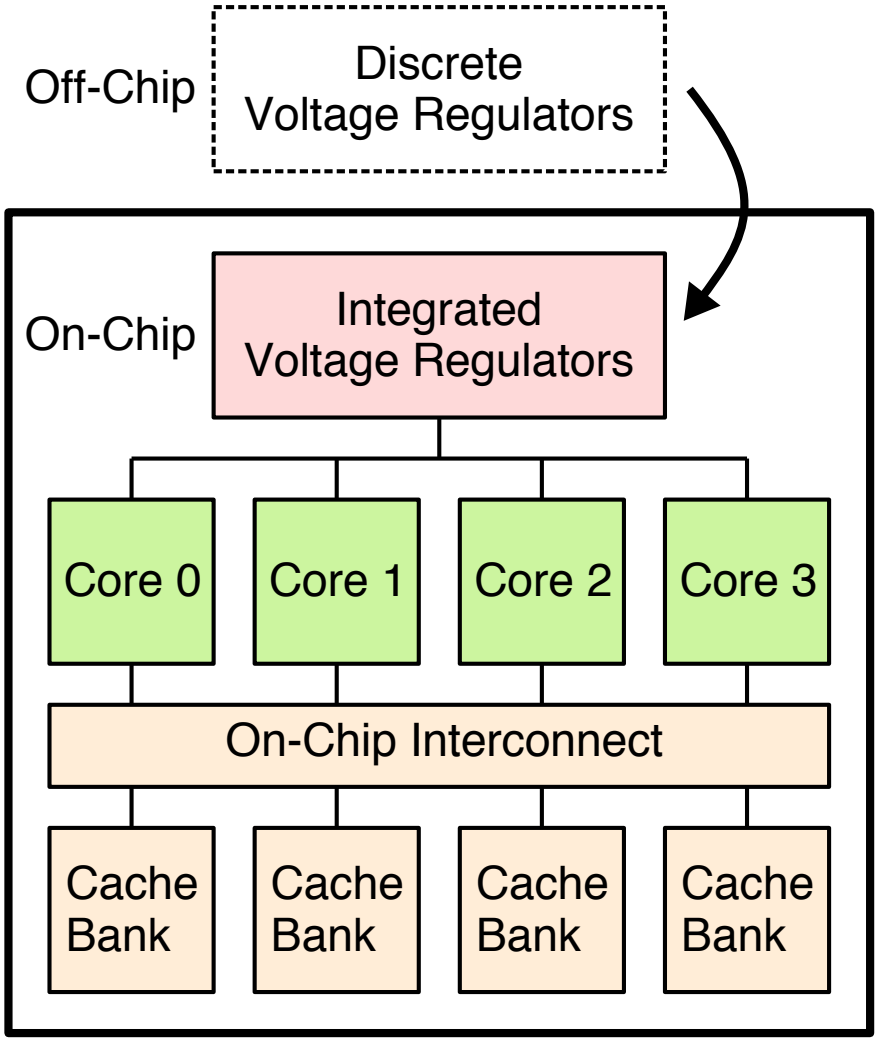
Motivation: Integrated Voltage Regulation (IVR)



Key Benefit of IVR

- Reduced System Cost

Motivation: Integrated Voltage Regulation (IVR)



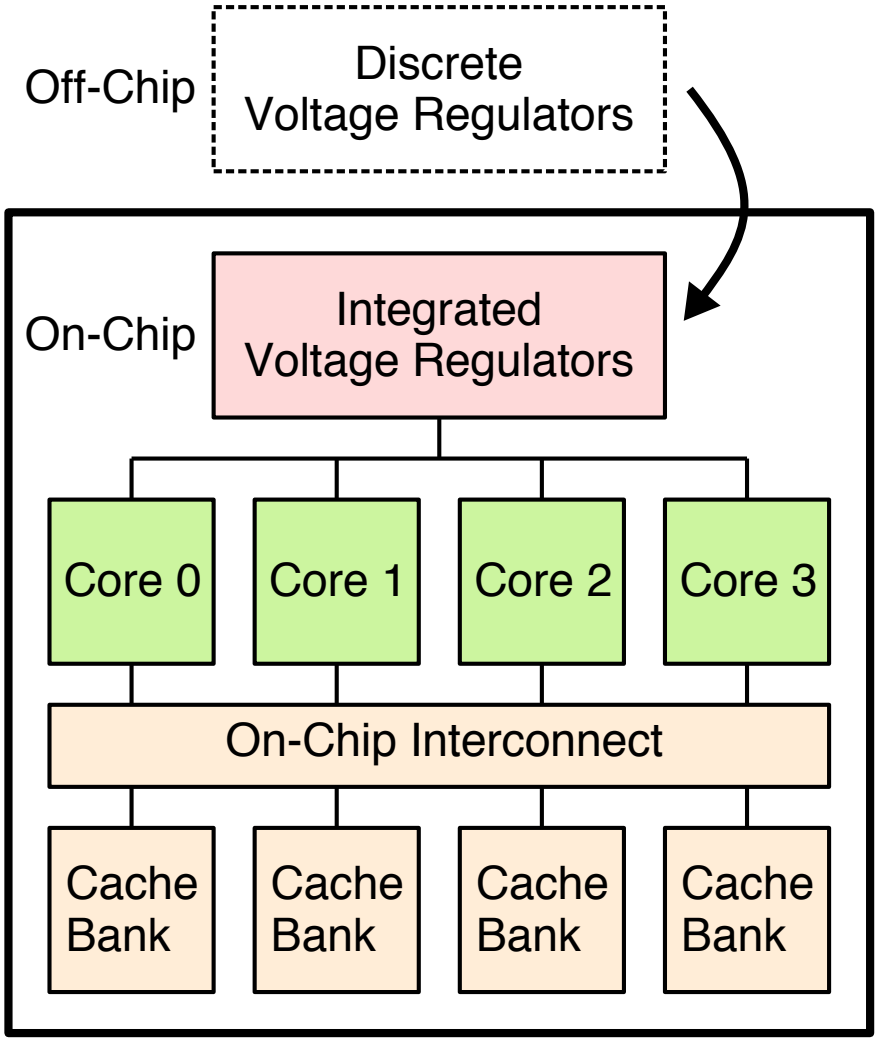
Key Benefit of IVR

- Reduced System Cost

Challenges of IVR

- Integrated energy-storage elements have low energy densities
- Low switching speeds with high parasitic losses

Motivation: Integrated Voltage Regulation (IVR)



Key Benefit of IVR

- Reduced System Cost

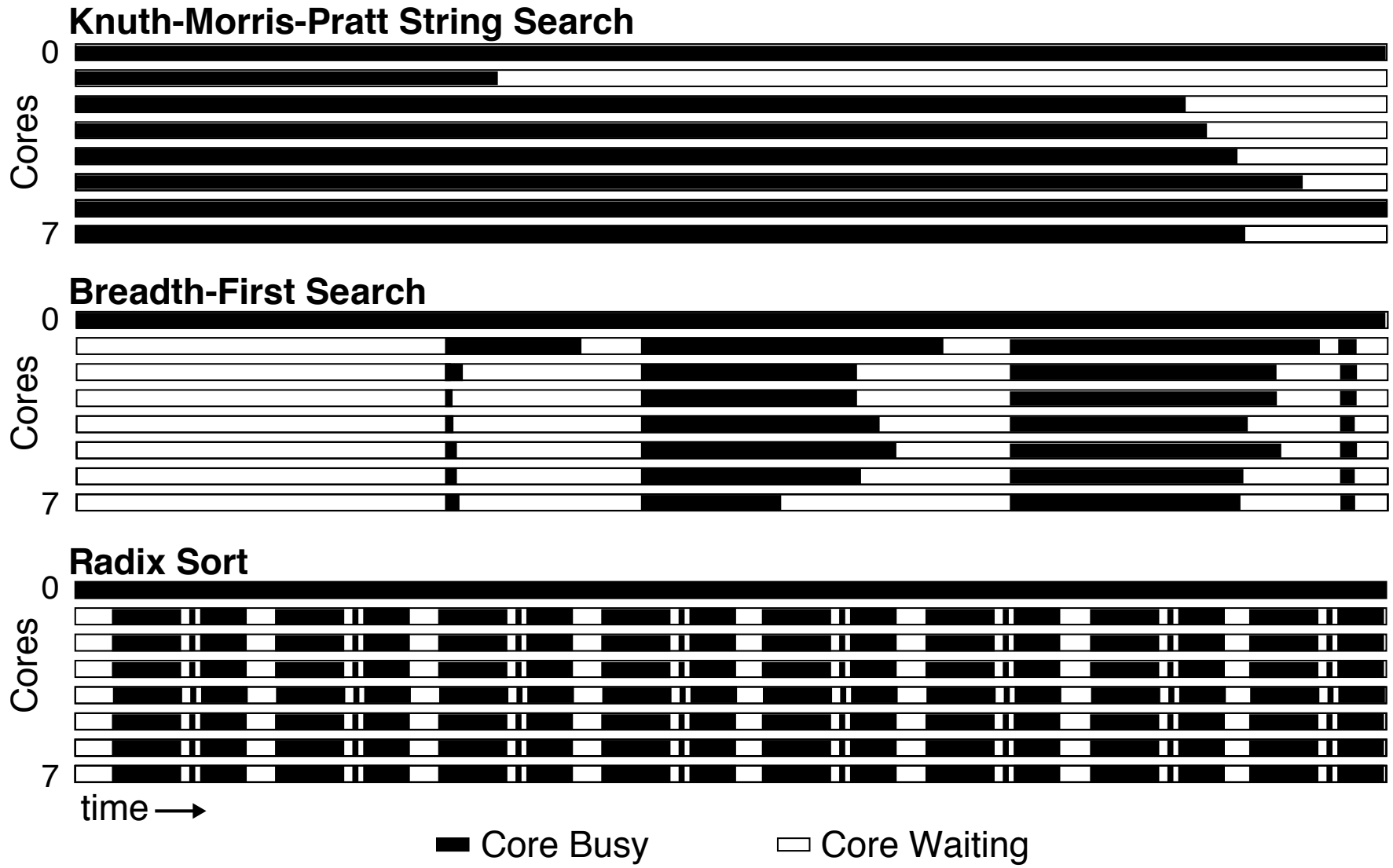
Challenges of IVR

- Integrated energy-storage elements have low energy densities
- Low switching speeds with high parasitic losses

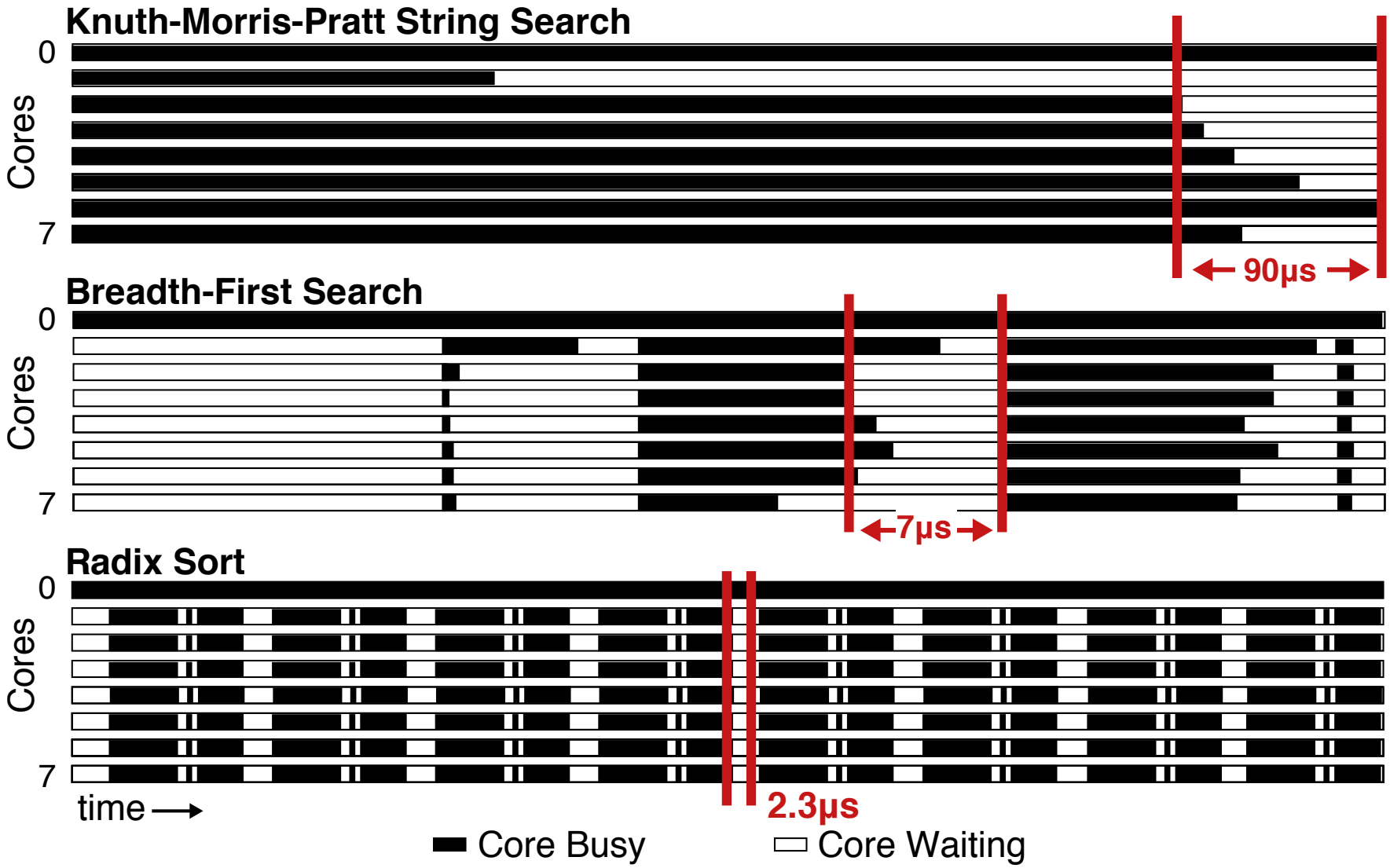
A New Era of IVR

- Energy storage elements have slightly improved energy densities
- Faster switches with low parasitic losses

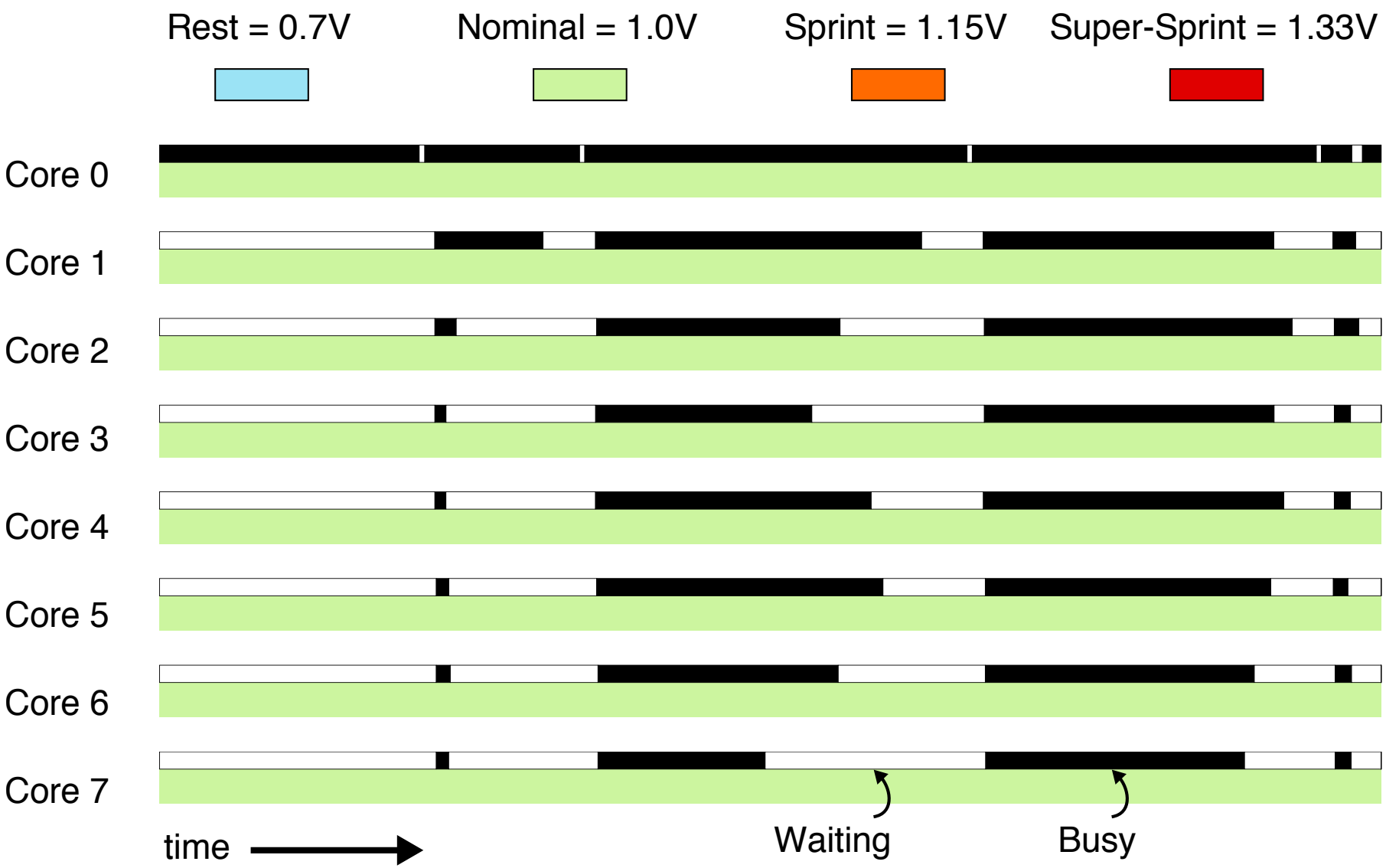
Fine-Grain Voltage Scaling Opportunities



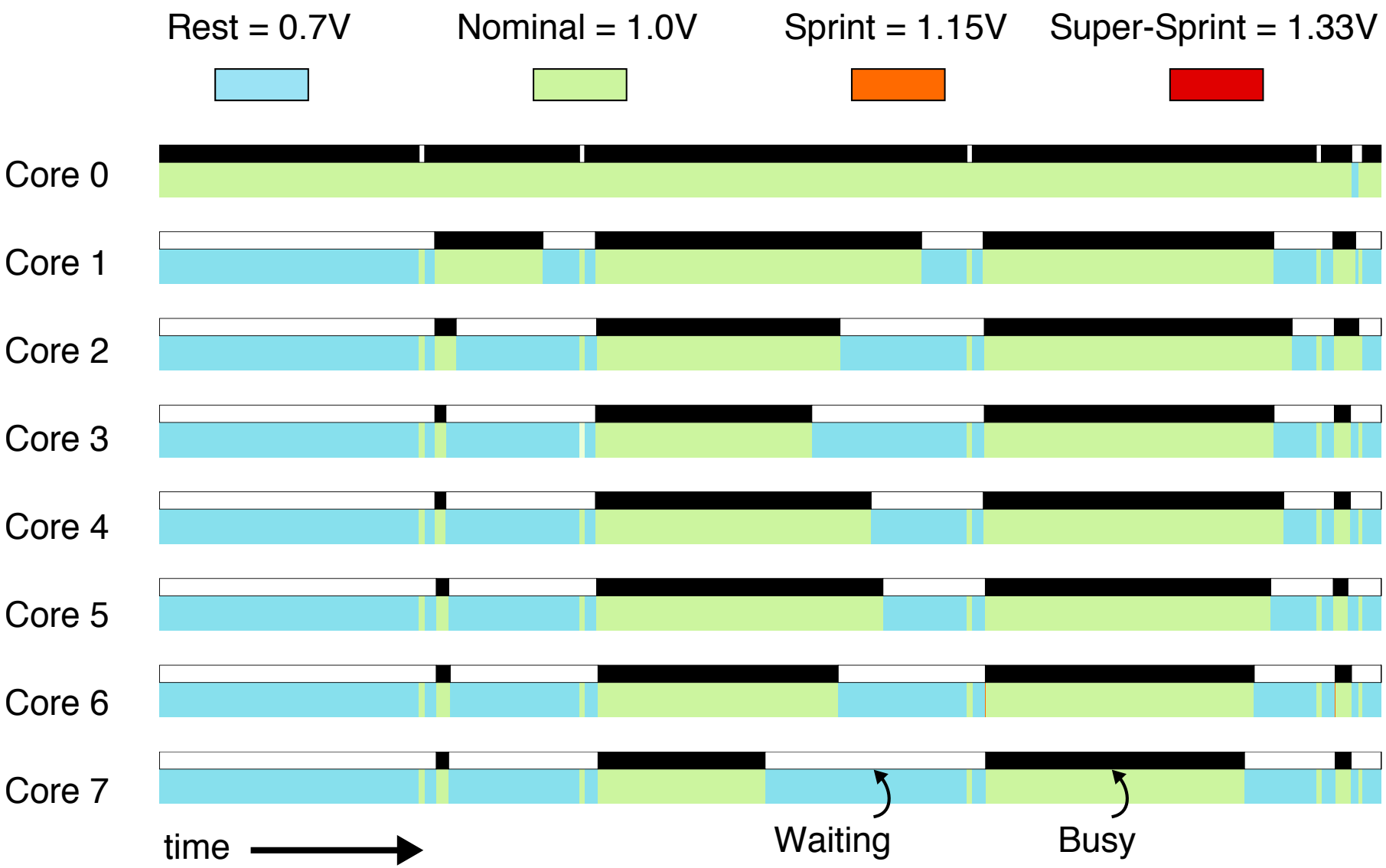
Fine-Grain Voltage Scaling Opportunities



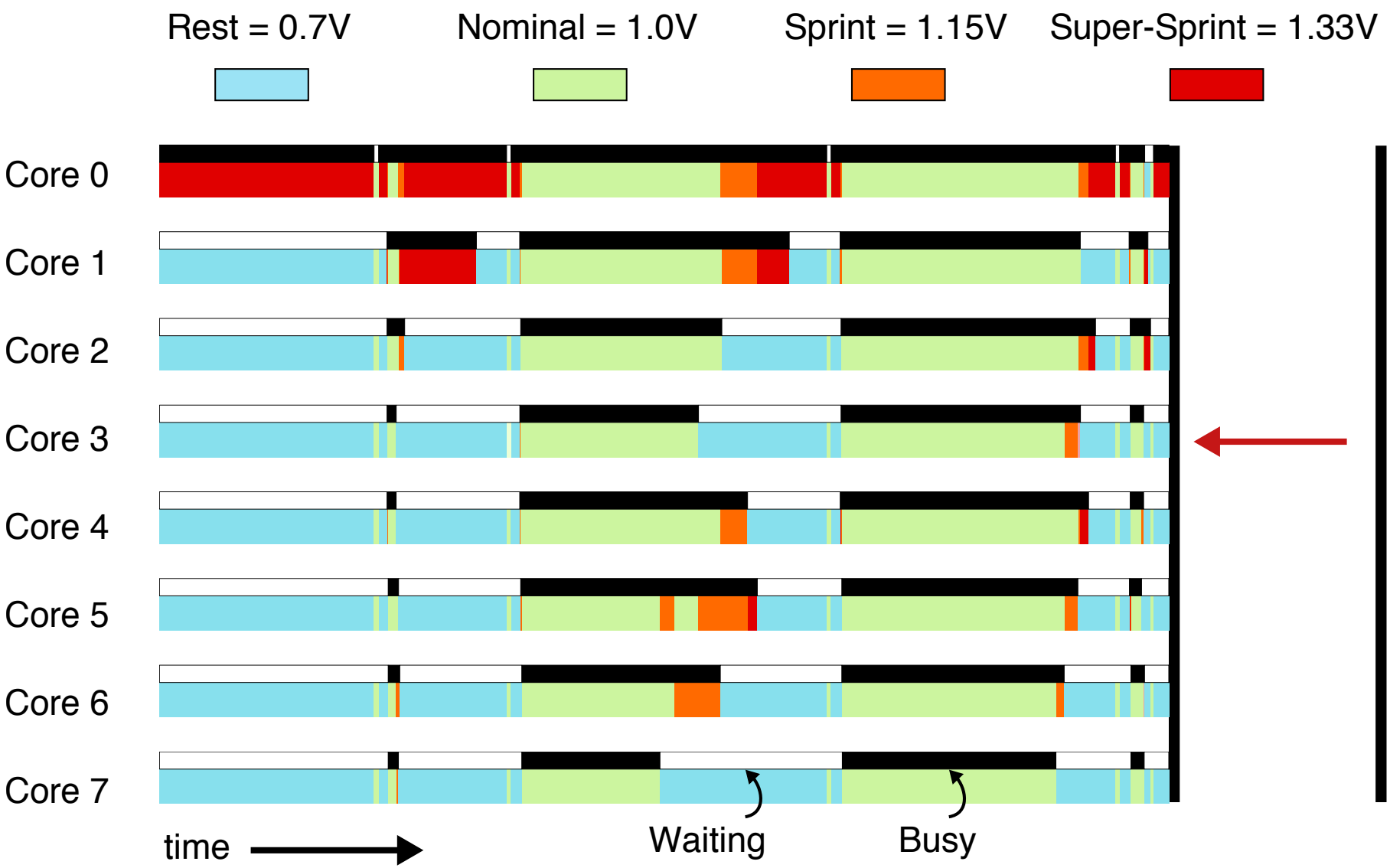
Ideal Fine-Grain Voltage Scaling: Breadth-First Search



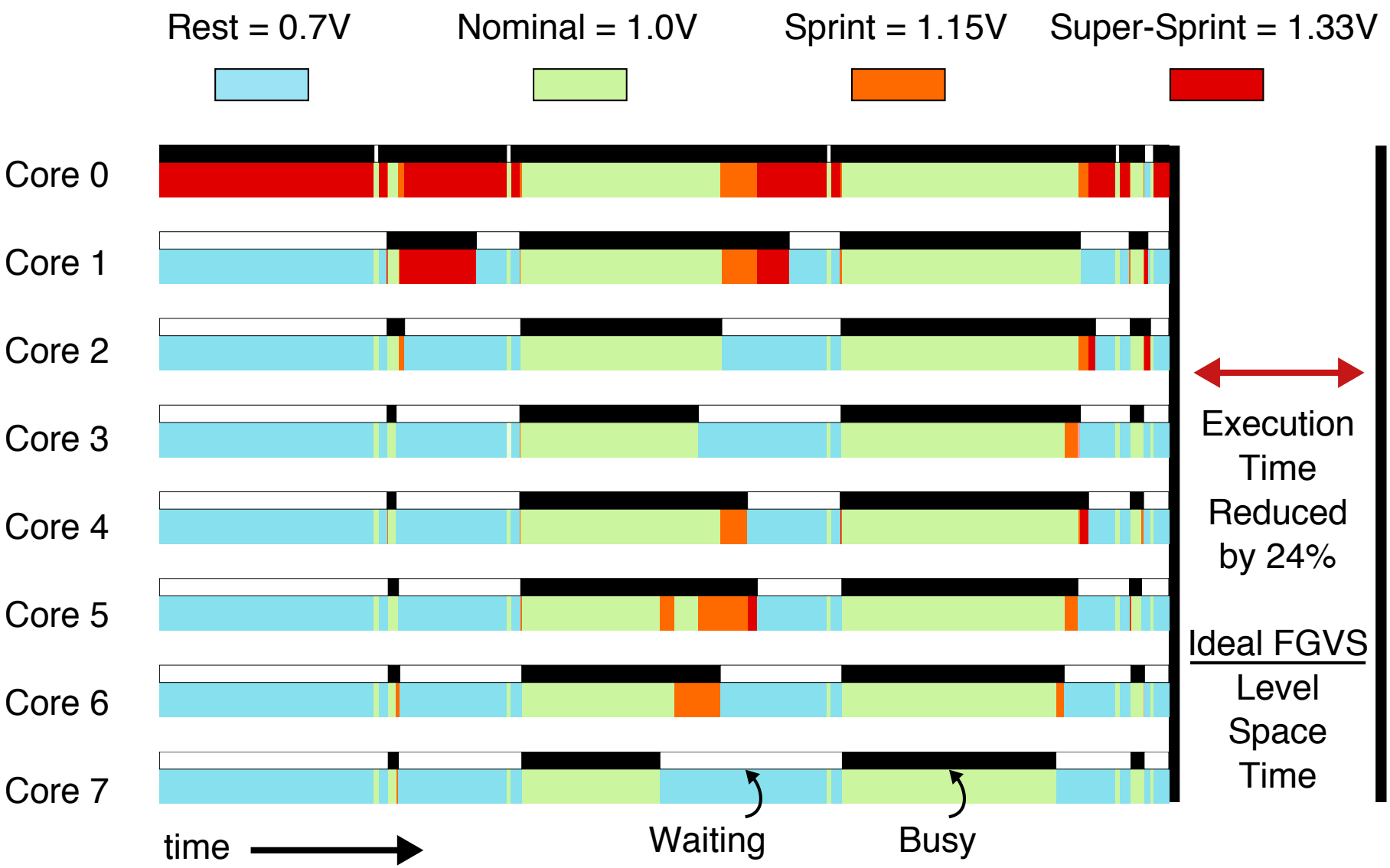
Ideal Fine-Grain Voltage Scaling: Breadth-First Search



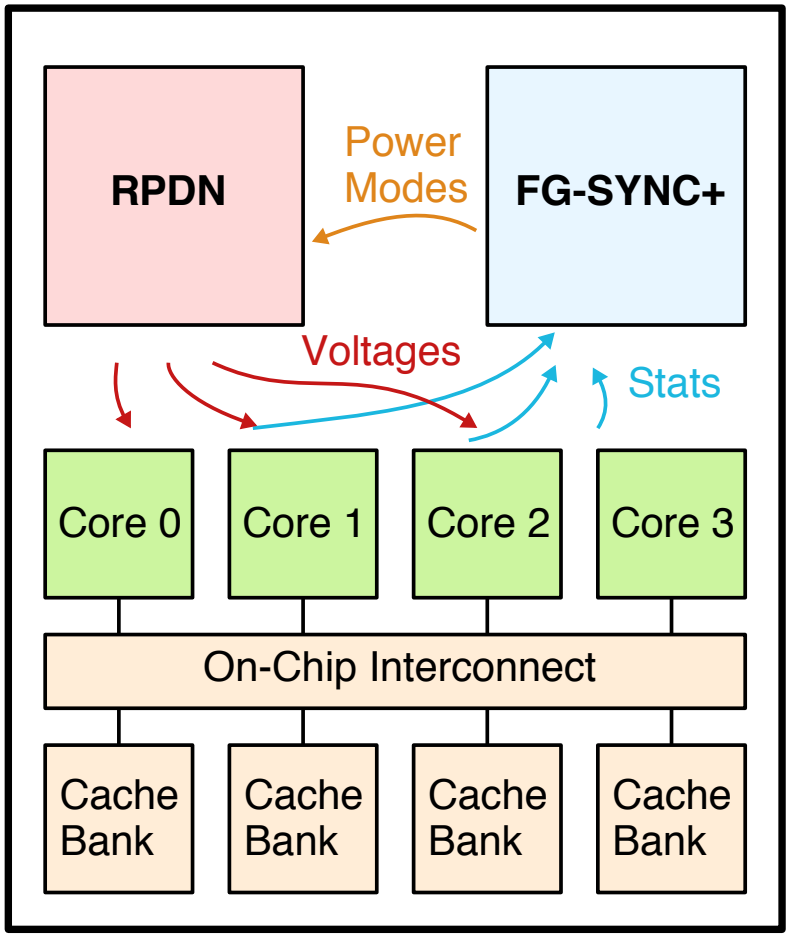
Ideal Fine-Grain Voltage Scaling: Breadth-First Search



Ideal Fine-Grain Voltage Scaling: Breadth-First Search



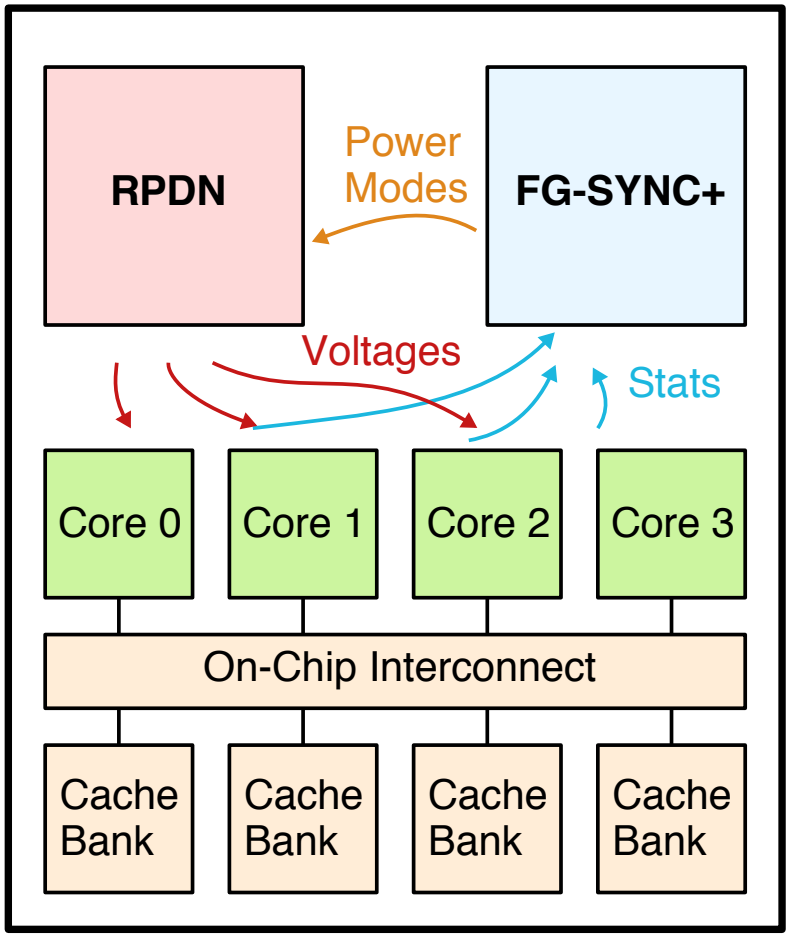
Enabling Realistic Fine-Grain Voltage Scaling



FGVS Architecture: FG-SYNC+

Use lightweight *software hints* and lookup tables derived offline to enable fast multi-level voltage configuration

Enabling Realistic Fine-Grain Voltage Scaling



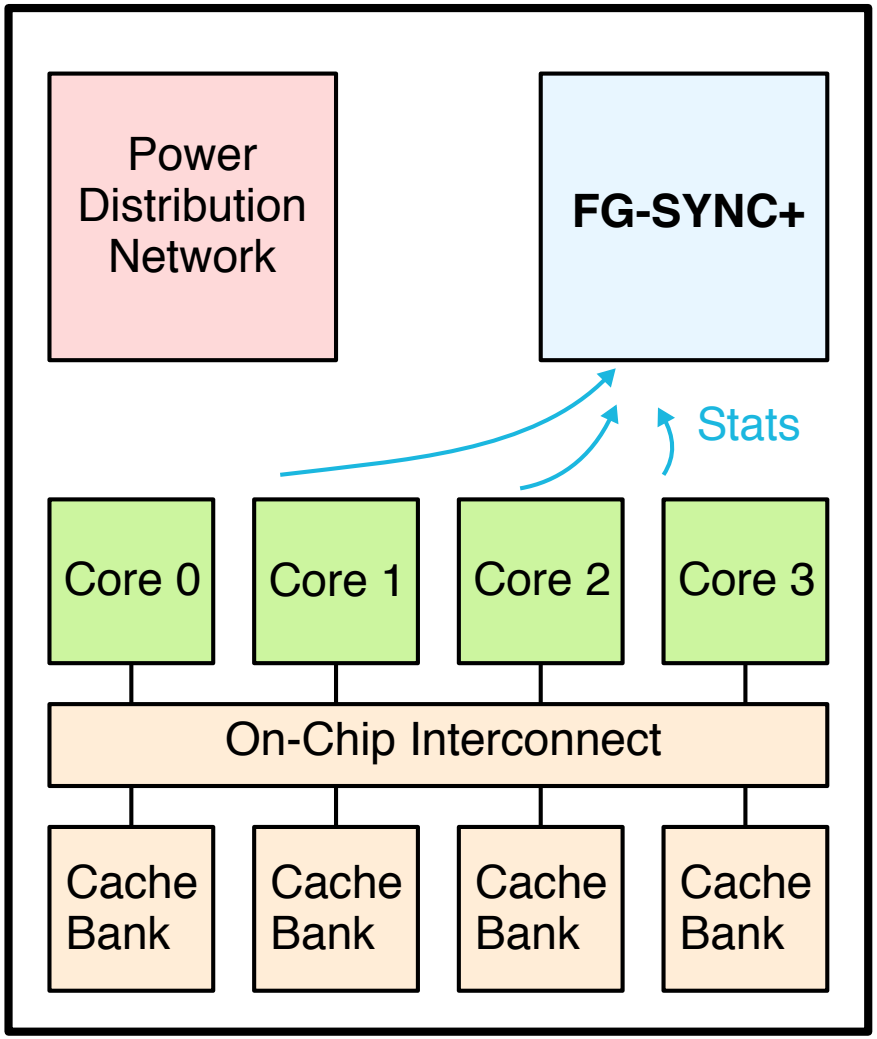
FGVS Architecture: FG-SYNC+

Use lightweight *software hints* and lookup tables derived offline to enable fast multi-level voltage configuration

FGVS Circuits: RPDN

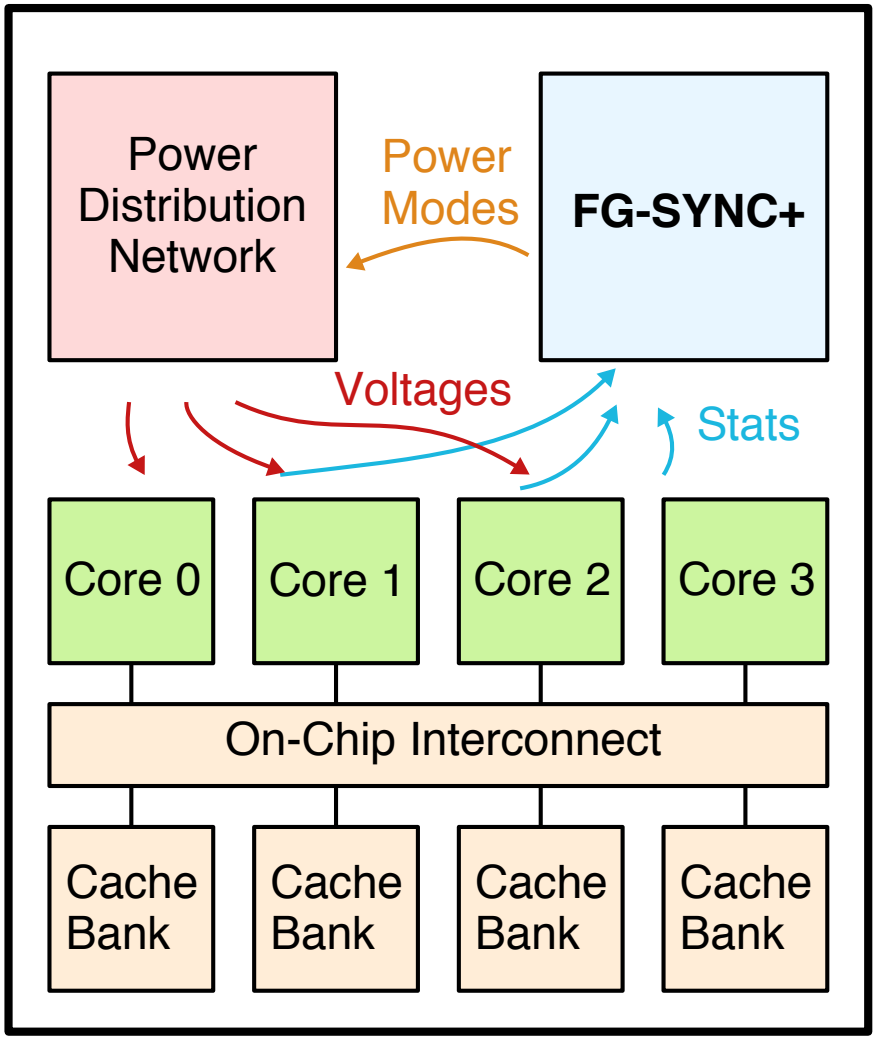
Enable sprinting cores to dynamically borrow energy storage from resting cores

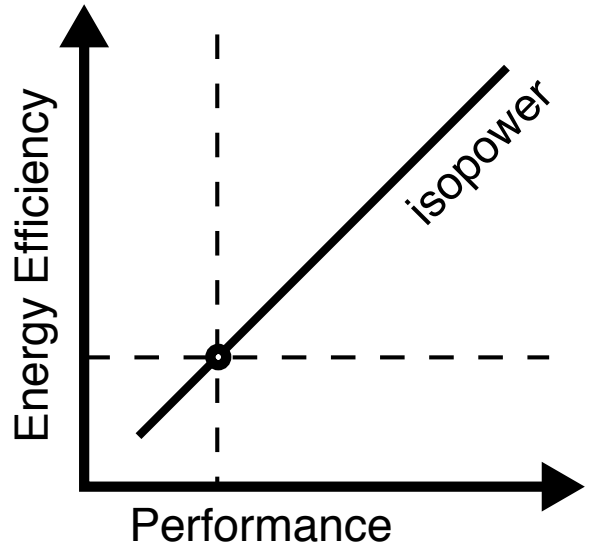
Fine-Grain Synchronization Controller (FG-SYNC+)

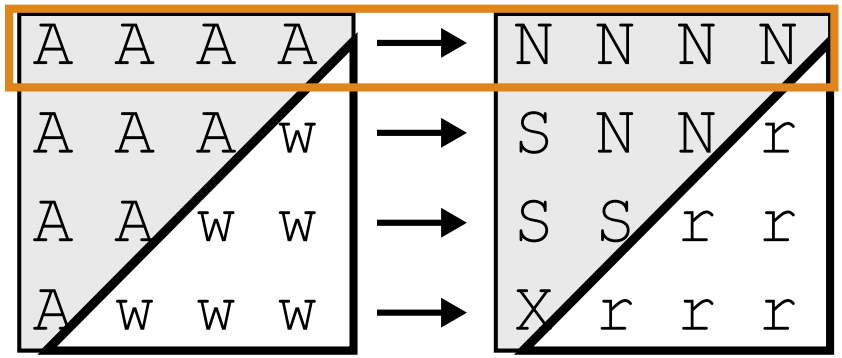



```
parallel_for(int i=0; i<N; n++){
   Before Start: Work Left Hint
   Loop Start: Activity Hint -- busy
   < loop body >
   Loop Ends: Activity Hint -- waiting
}
```

Fine-Grain Synchronization Controller (FG-SYNC+)



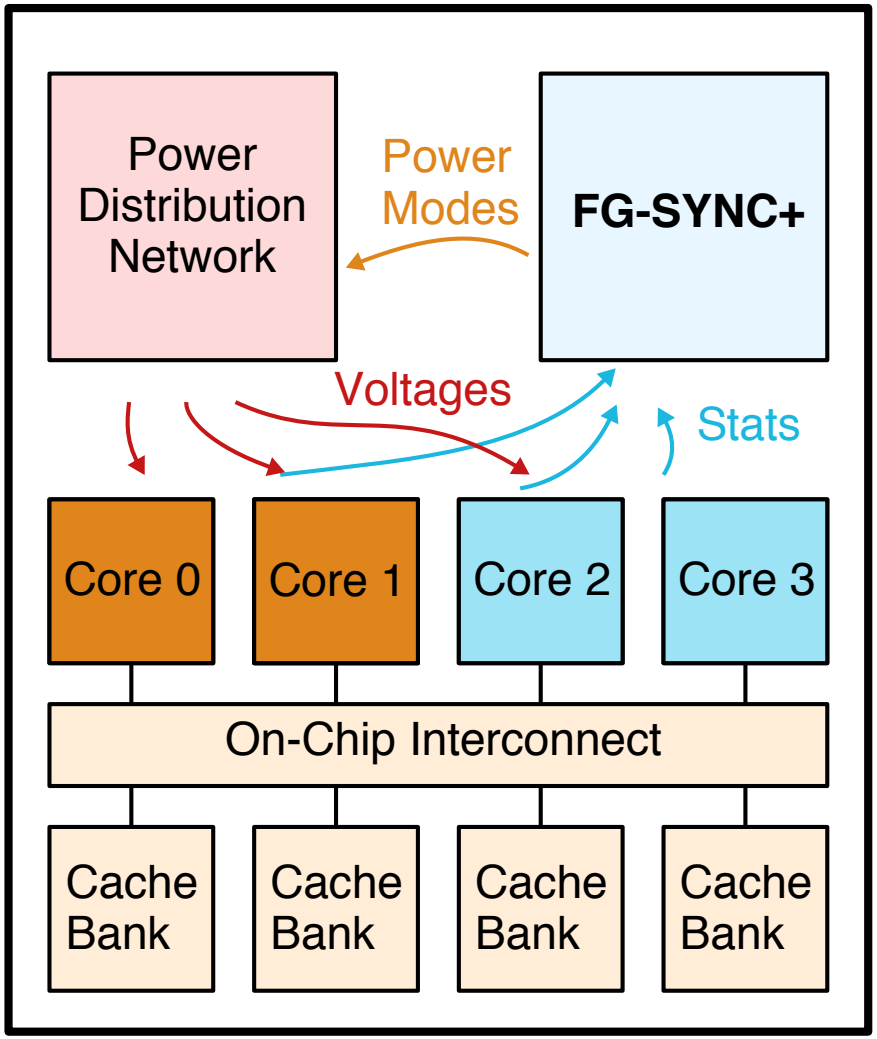


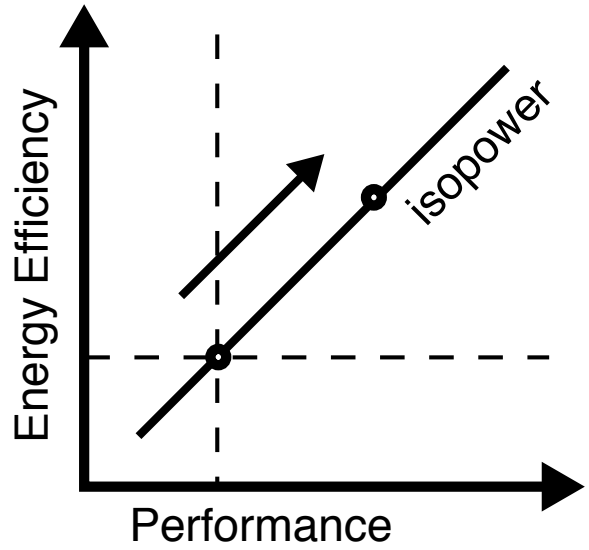


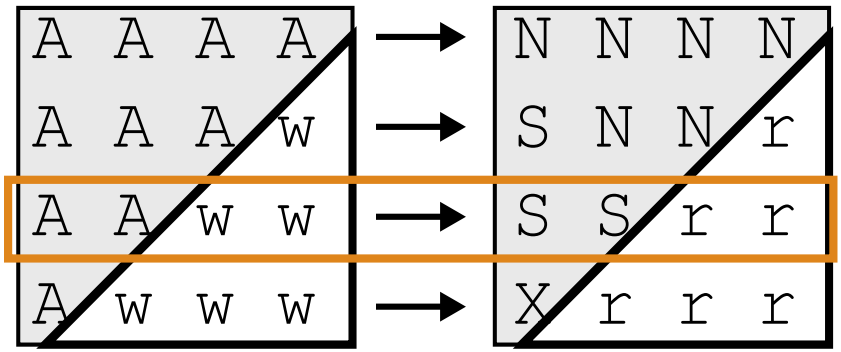
(**A**ctive **W**aiting)

(**N, r** Nominal, rest)
(**S, X** Sprint modes)

Fine-Grain Synchronization Controller (FG-SYNC+)



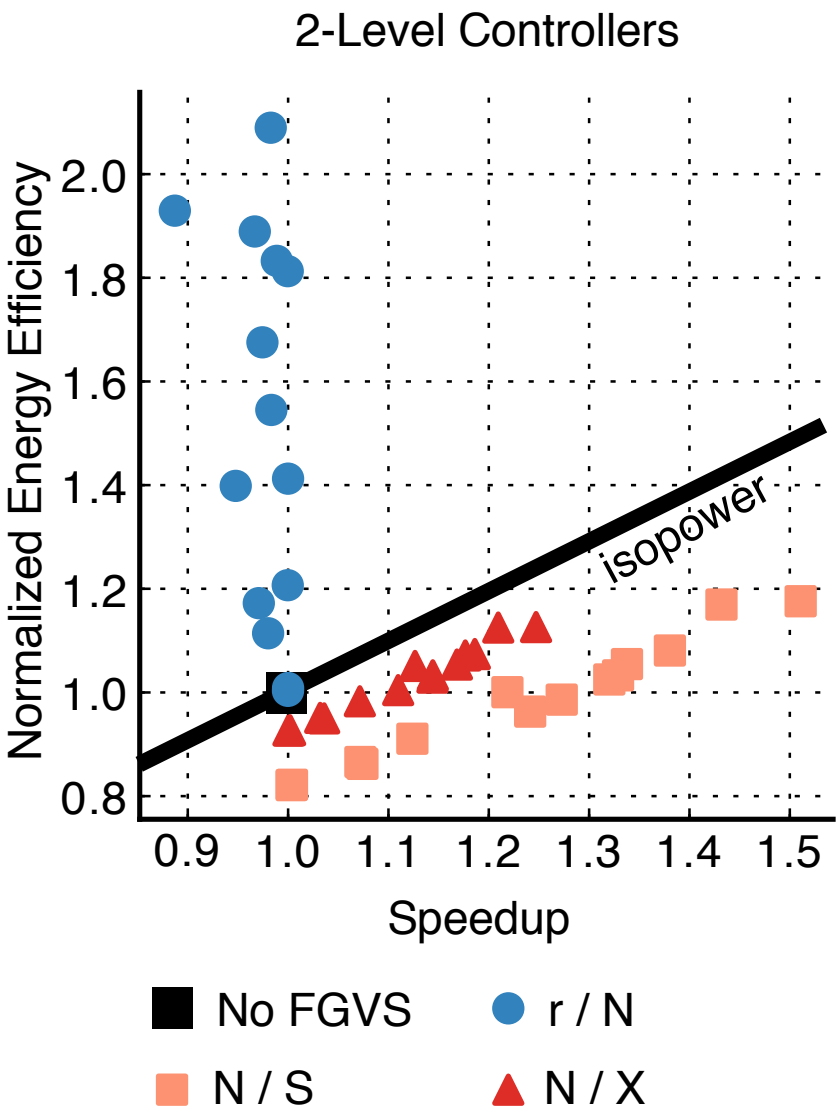


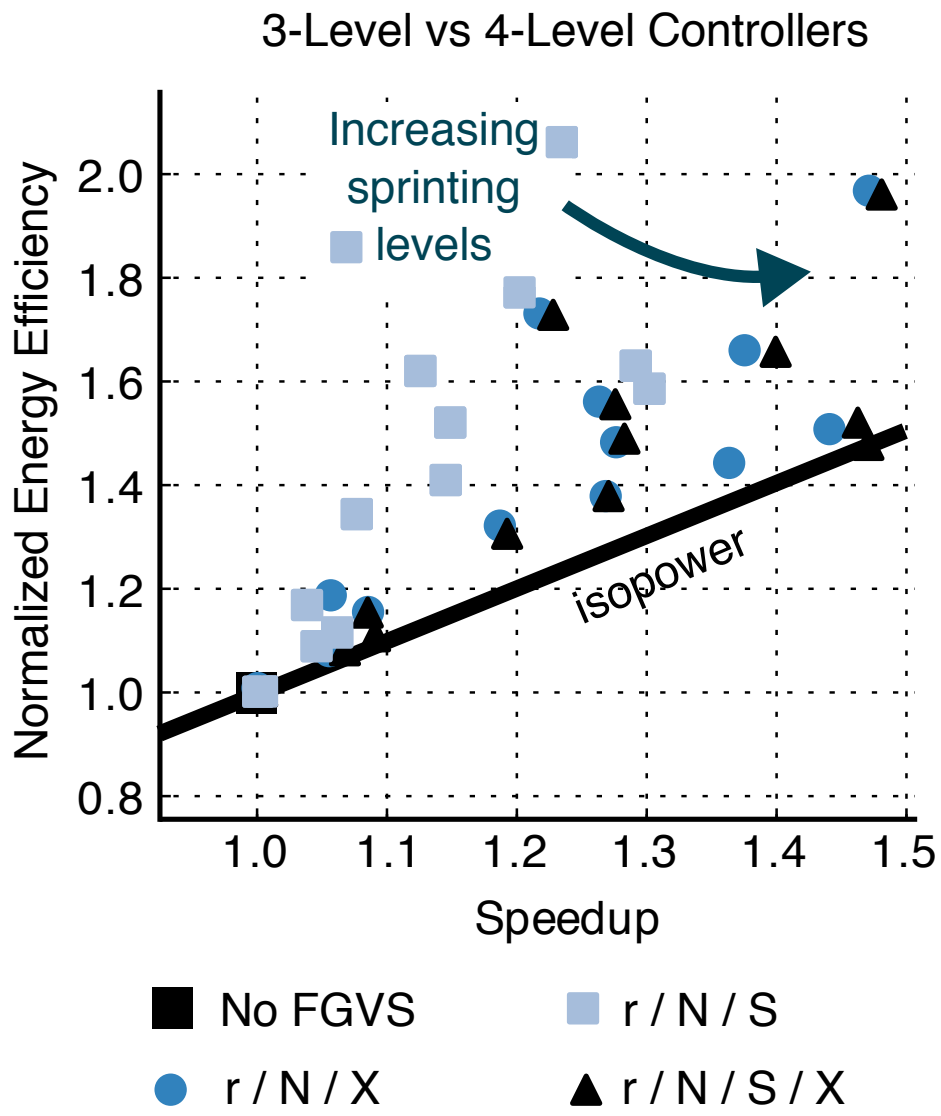


(**A**ctive **W**aiting)

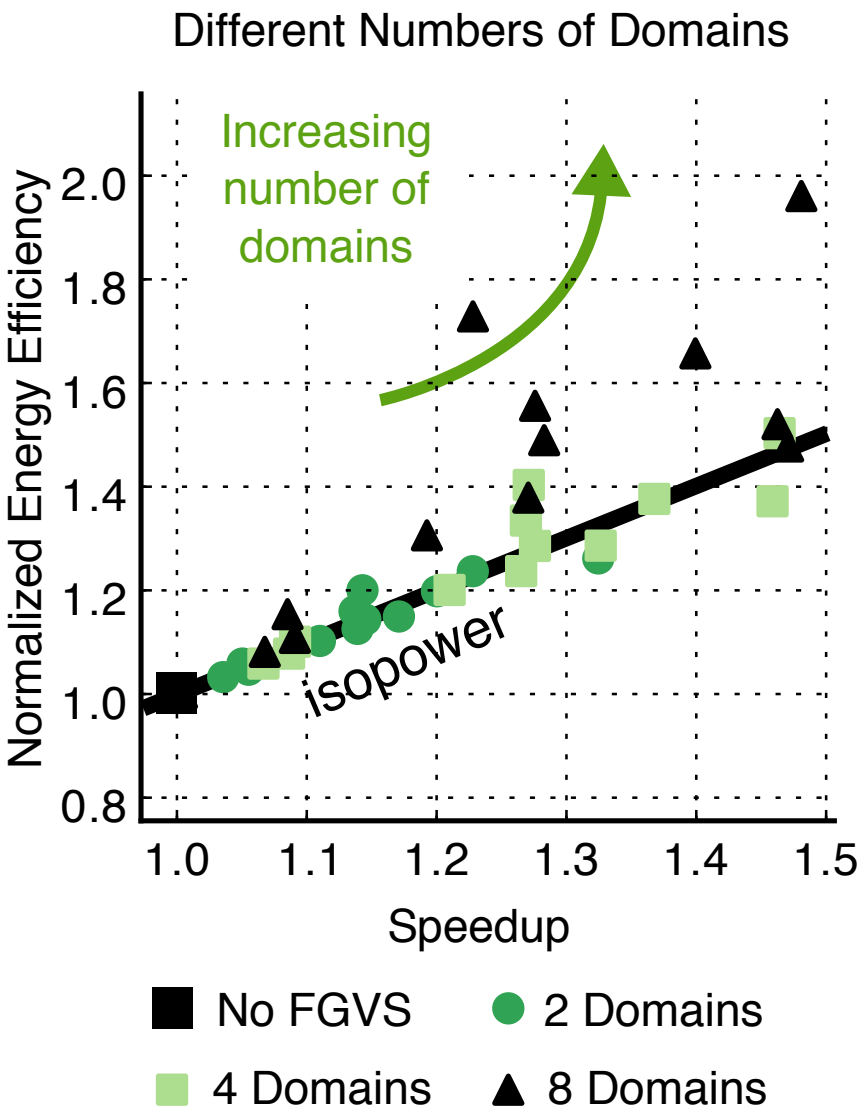
(**N, r** Nominal, rest)
(**S, X** Sprint modes)

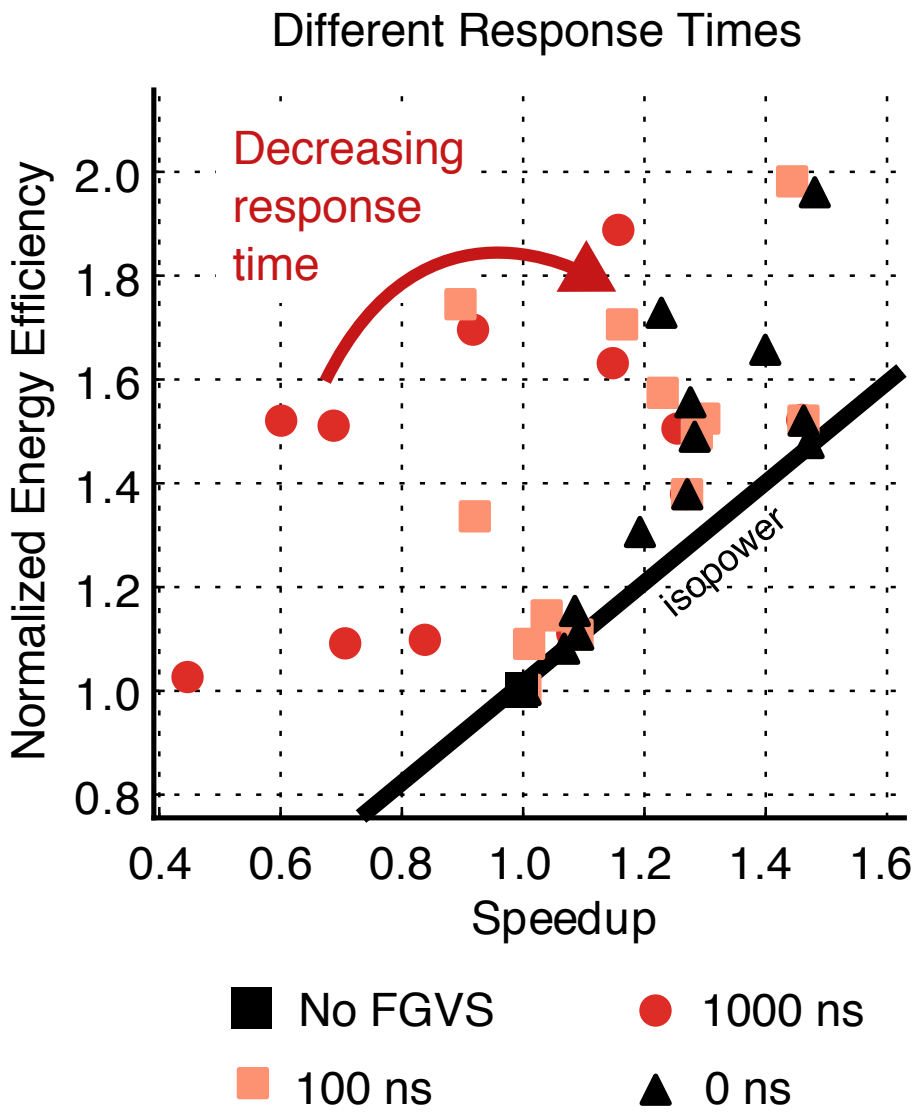
Non-Ideal FGVS: Different Levels



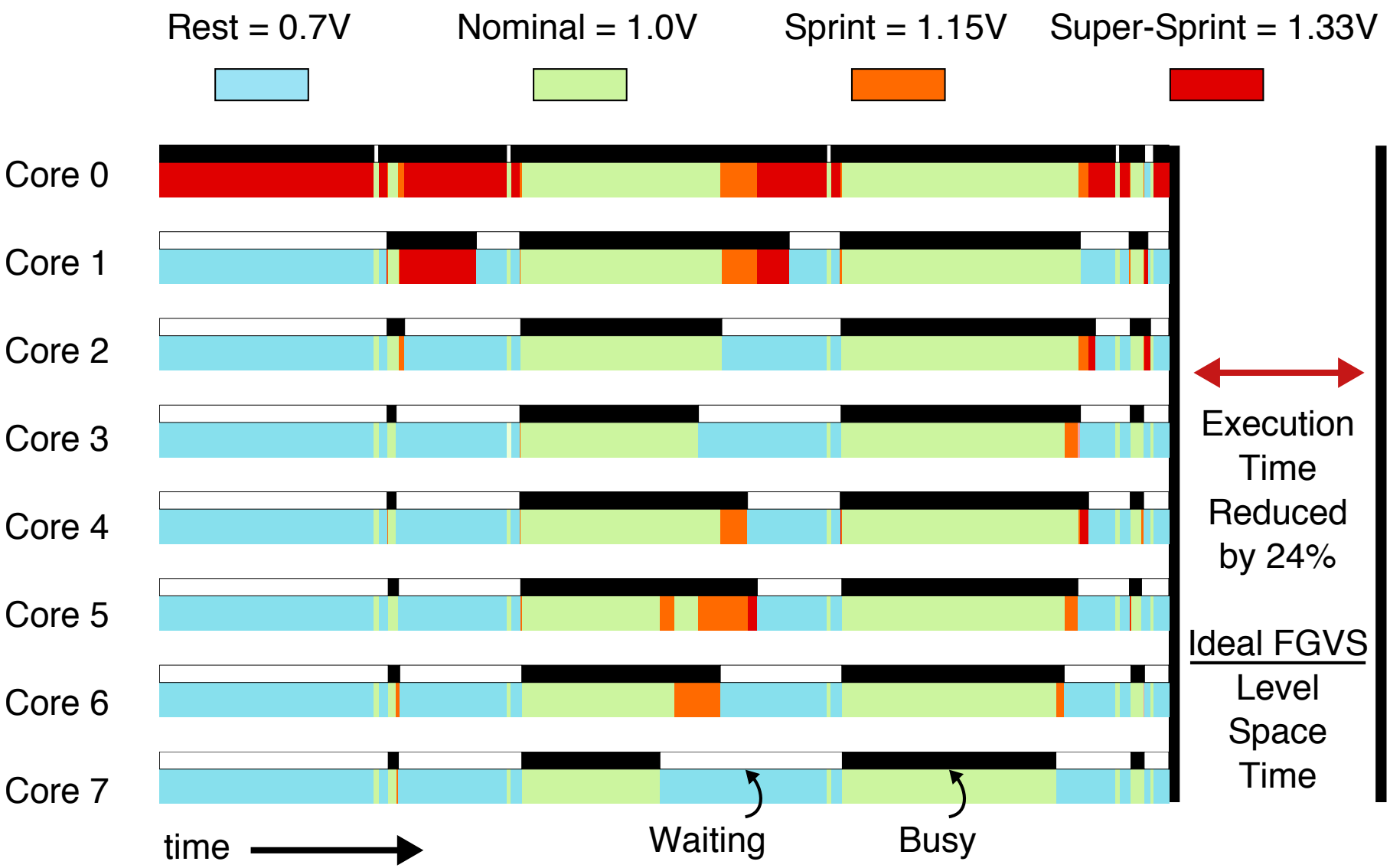


Non-Ideal FGVS: Space and Time

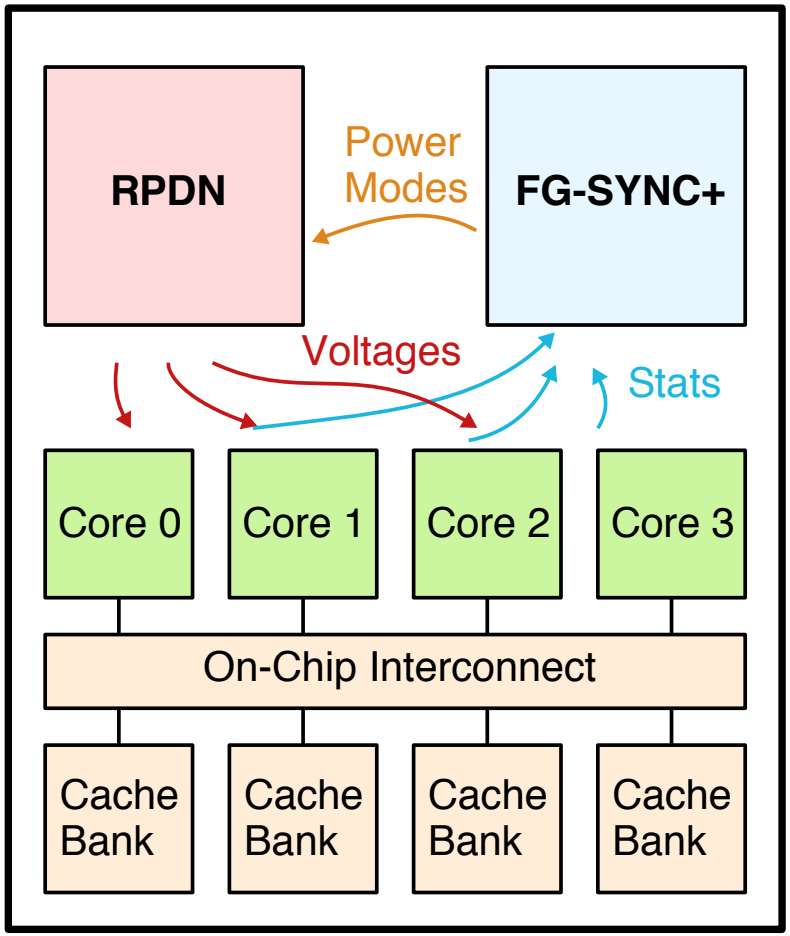




Skipping Details.. FG-SYNC+ Achieved Its Goal



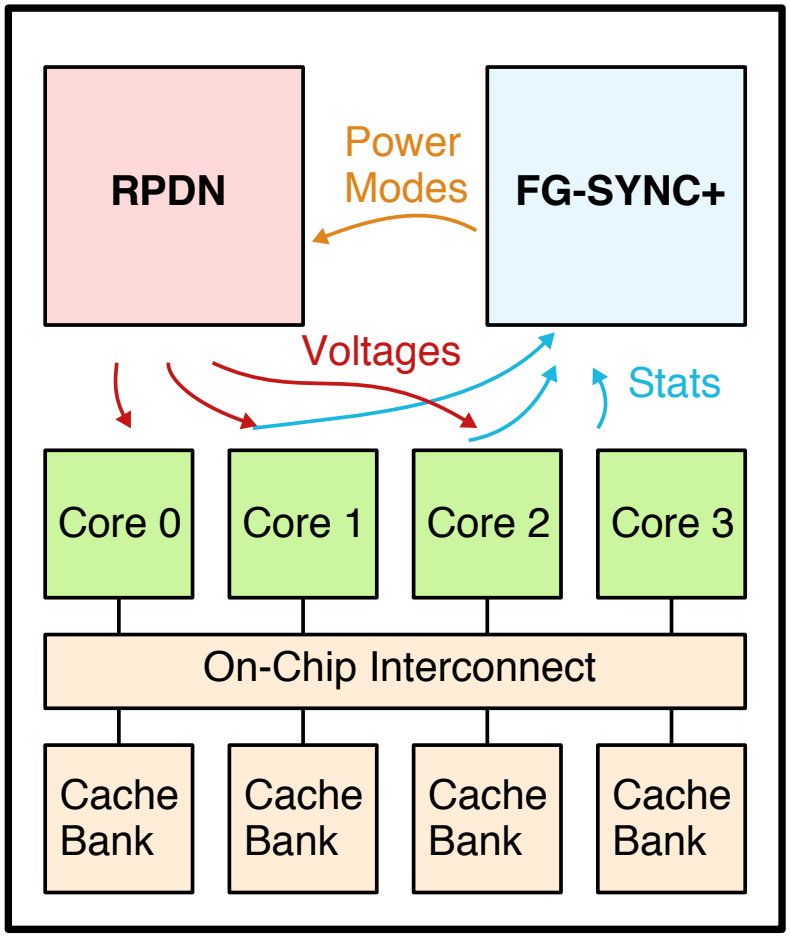
Enabling Realistic Fine-Grain Voltage Scaling



FGVS Architecture: FG-SYNC+

Use lightweight *software hints* and lookup tables derived offline to enable fast multi-level voltage configuration

Enabling Realistic Fine-Grain Voltage Scaling



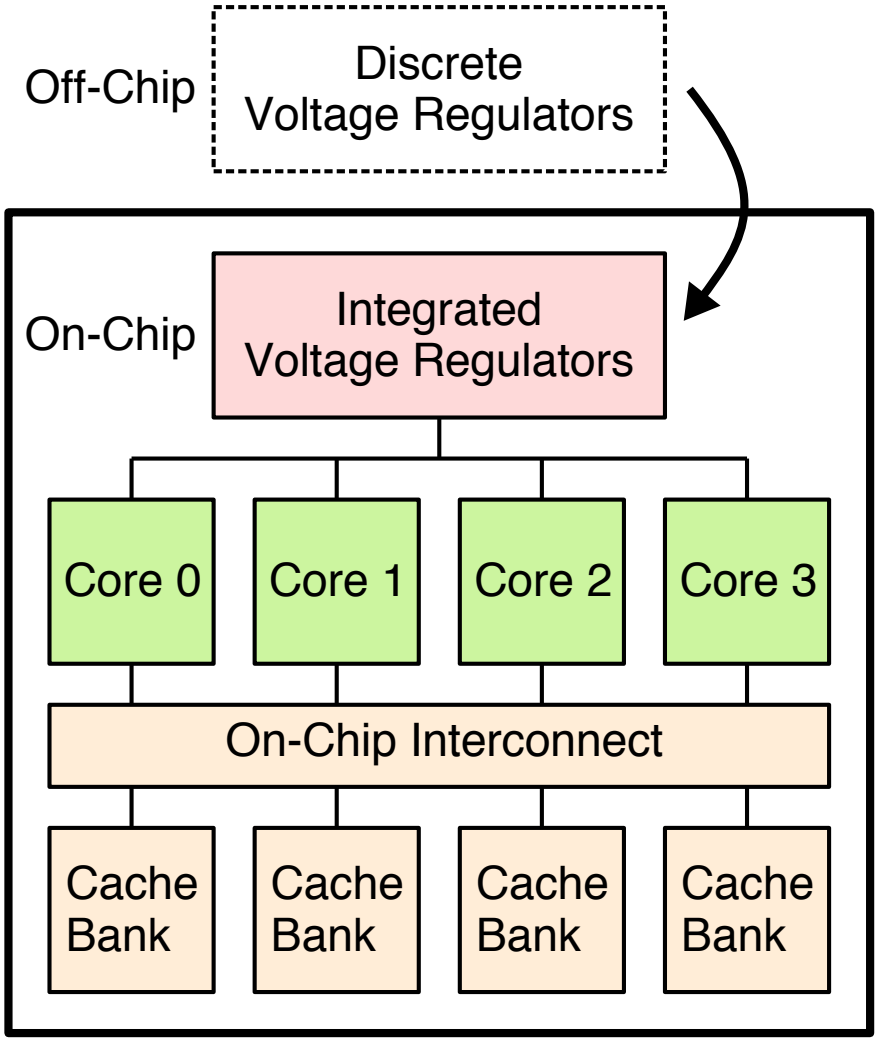
FGVS Architecture: FG-SYNC+

Use lightweight *software hints* and lookup tables derived offline to enable fast multi-level voltage configuration

FGVS Circuits: RPDN

Enable sprinting cores to dynamically borrow energy storage from resting cores

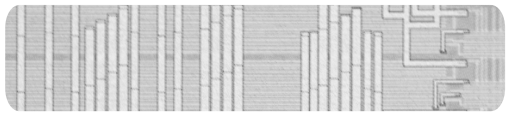
Key Challenge of Integrated Voltage Regulation



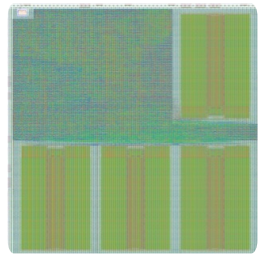
What's the Problem?

- Integrated VRs are BIG

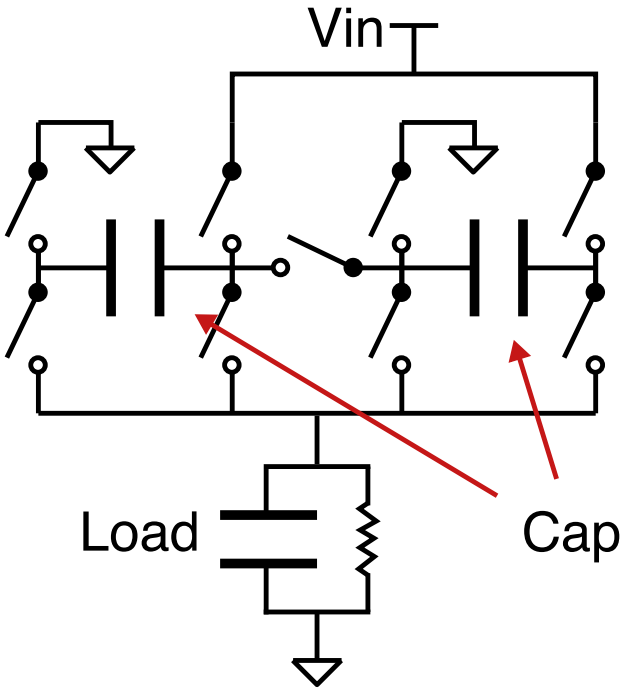
Technology-Normalized
Integrated Voltage Regulator

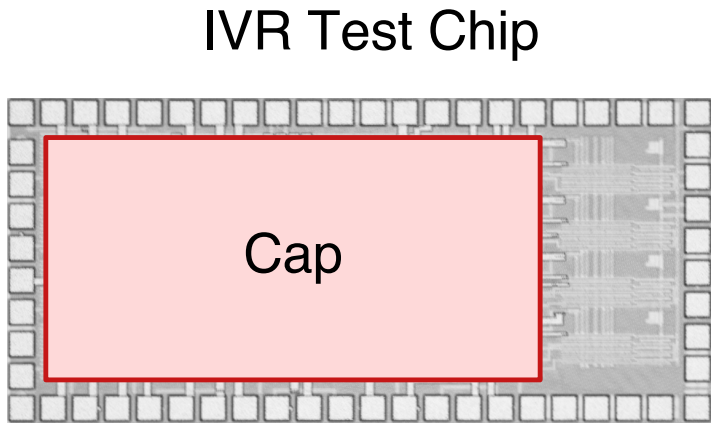


Technology-Normalized
Simple RISC Core

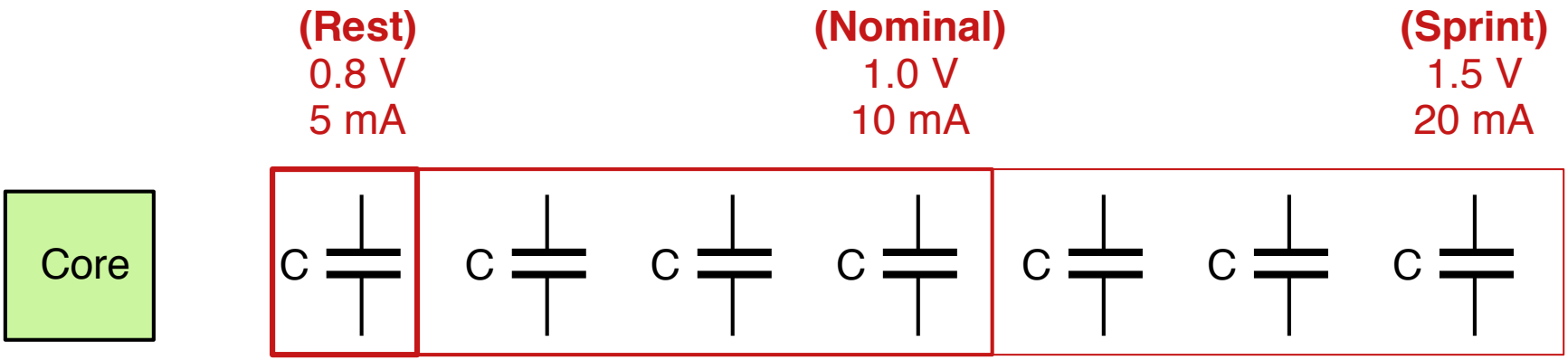


What's taking all that area?

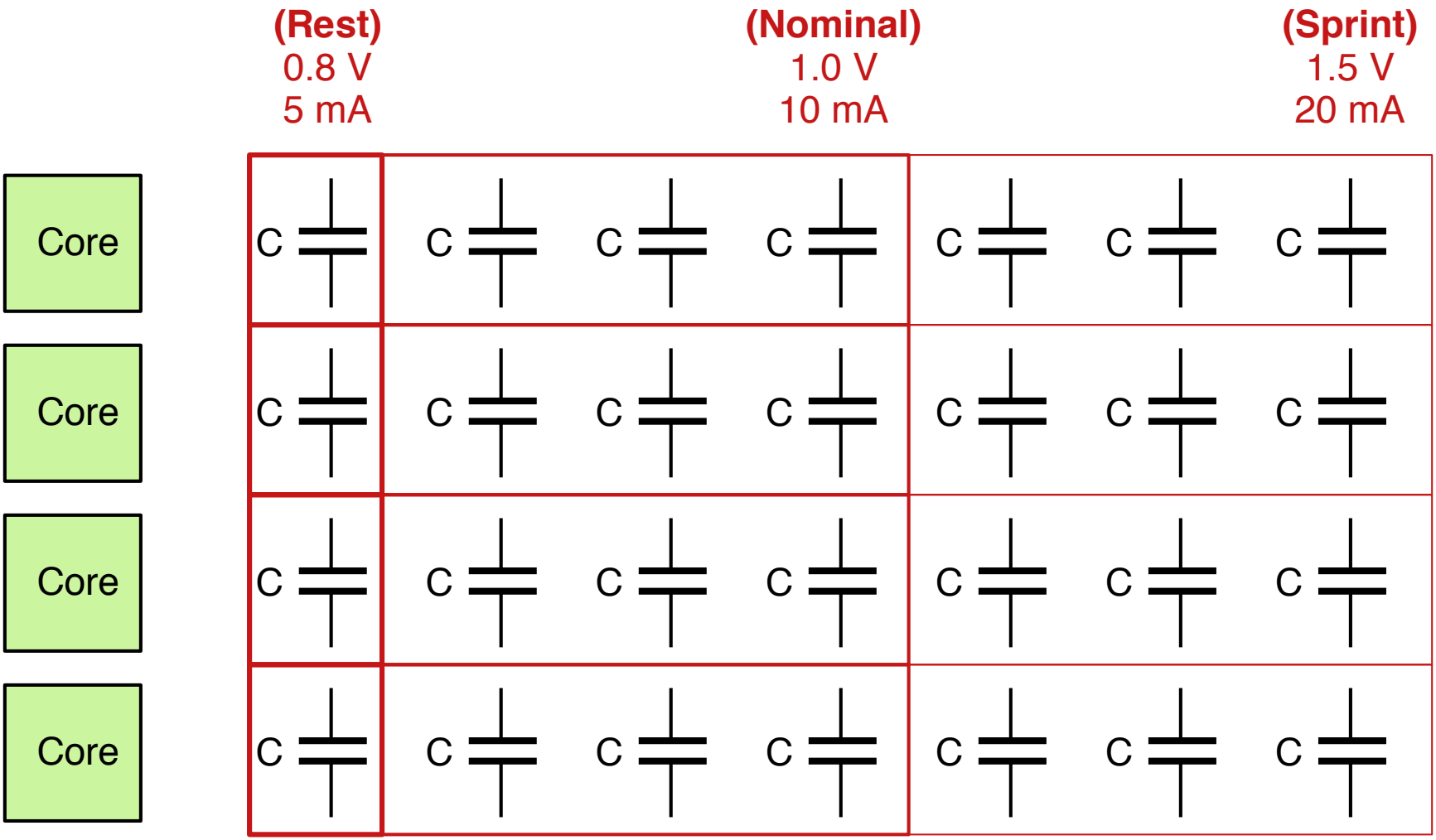




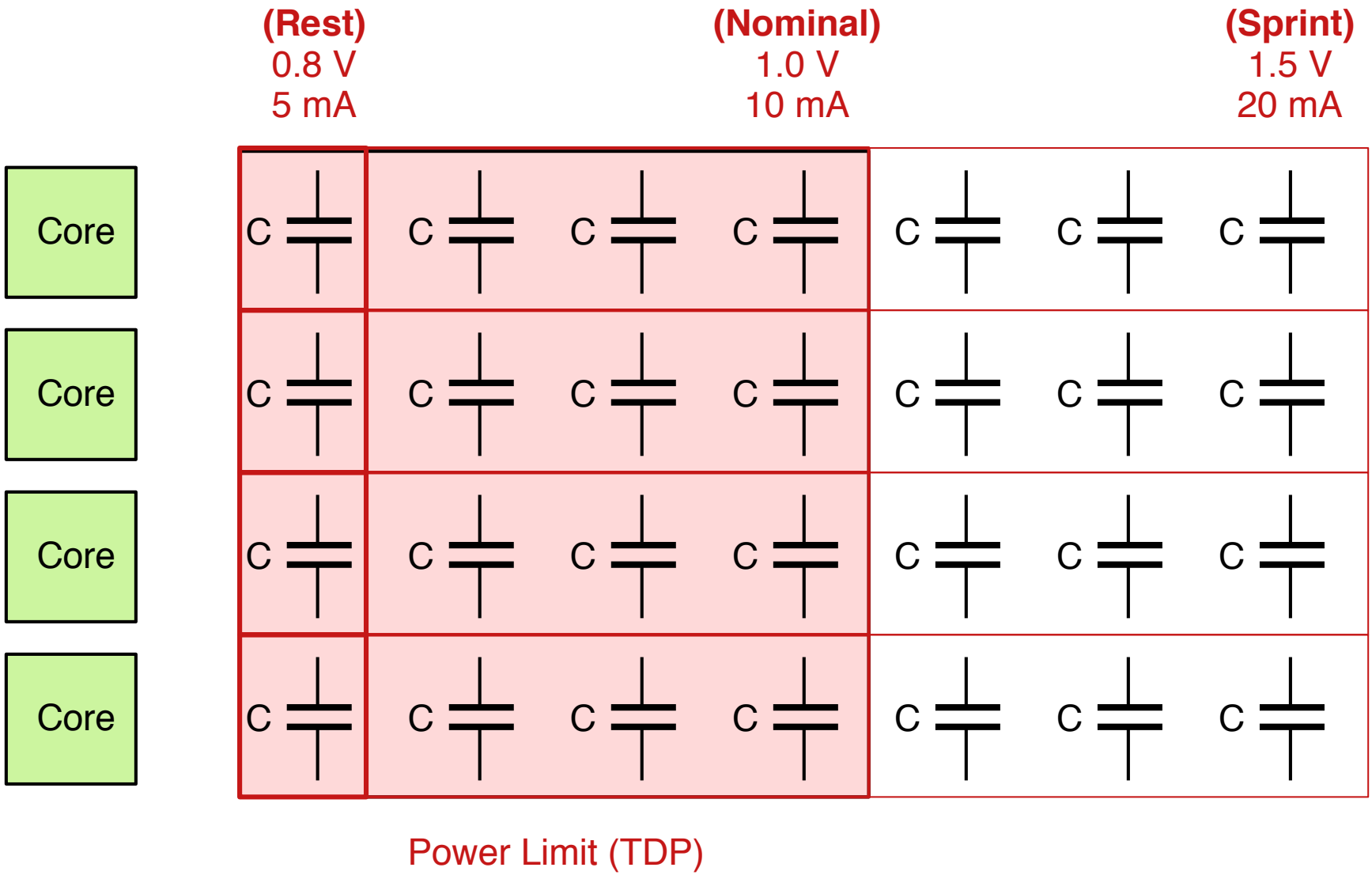
Key Architecture-Level Intuition



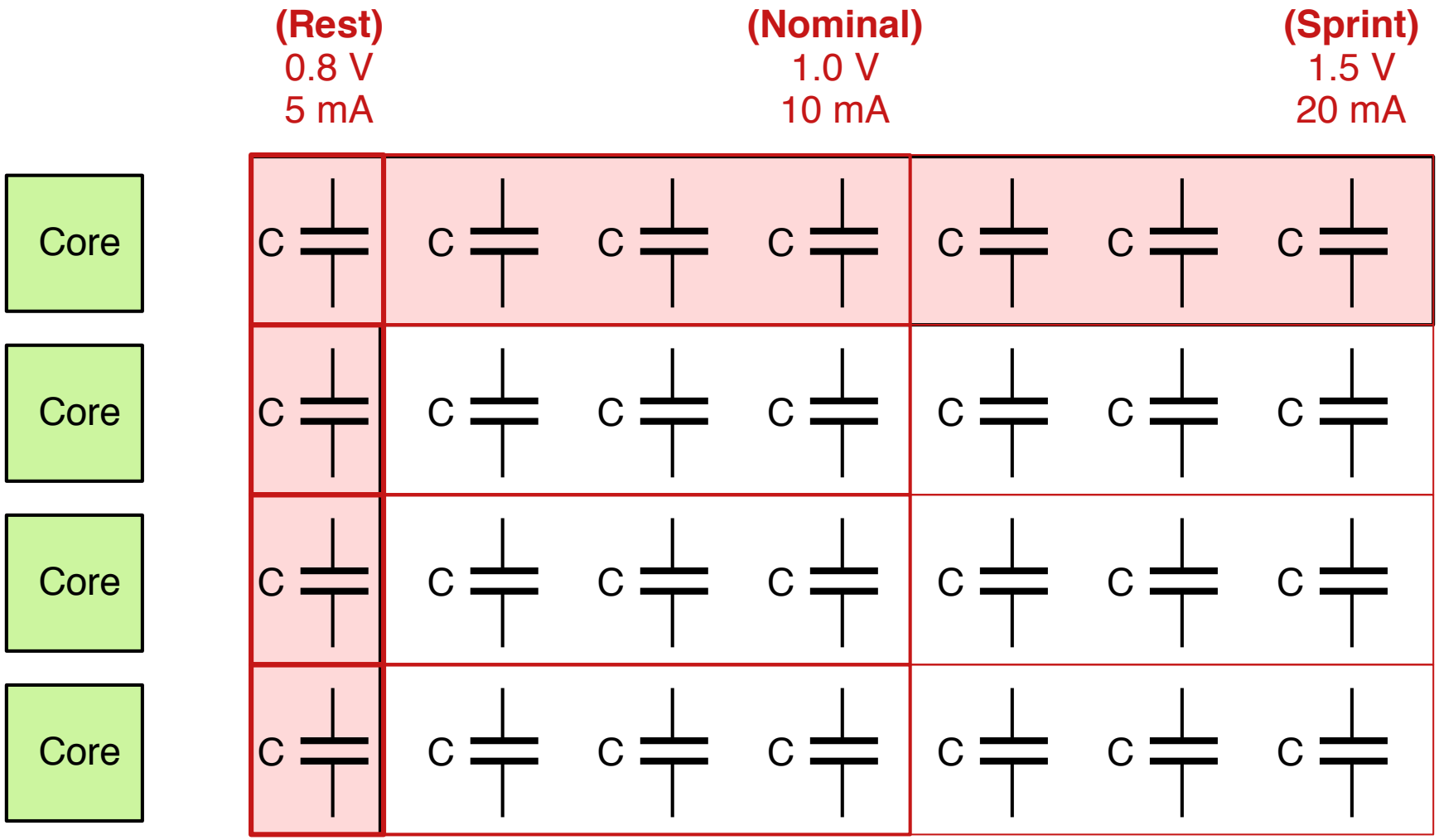
Key Architecture-Level Intuition



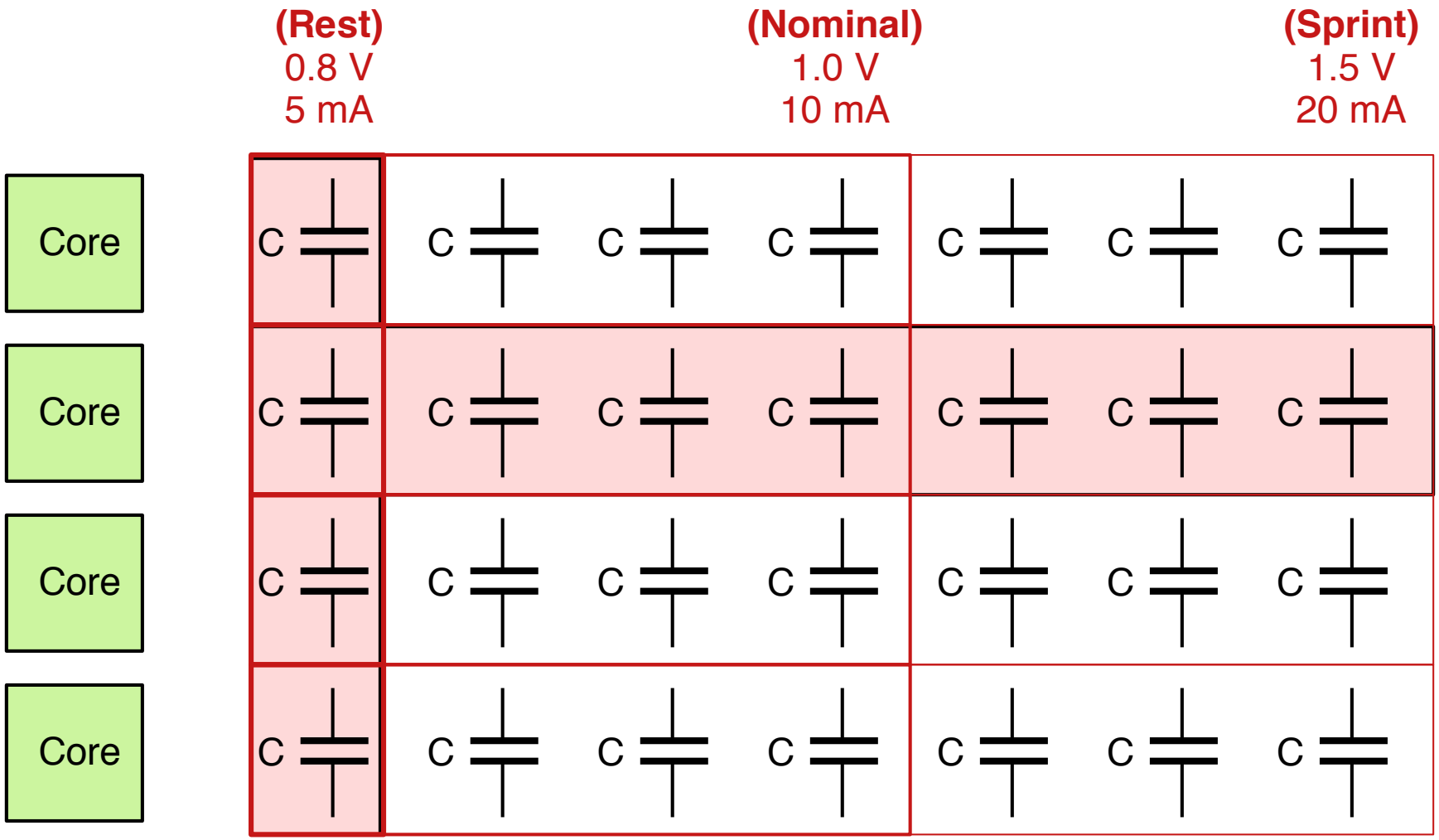
Key Architecture-Level Intuition



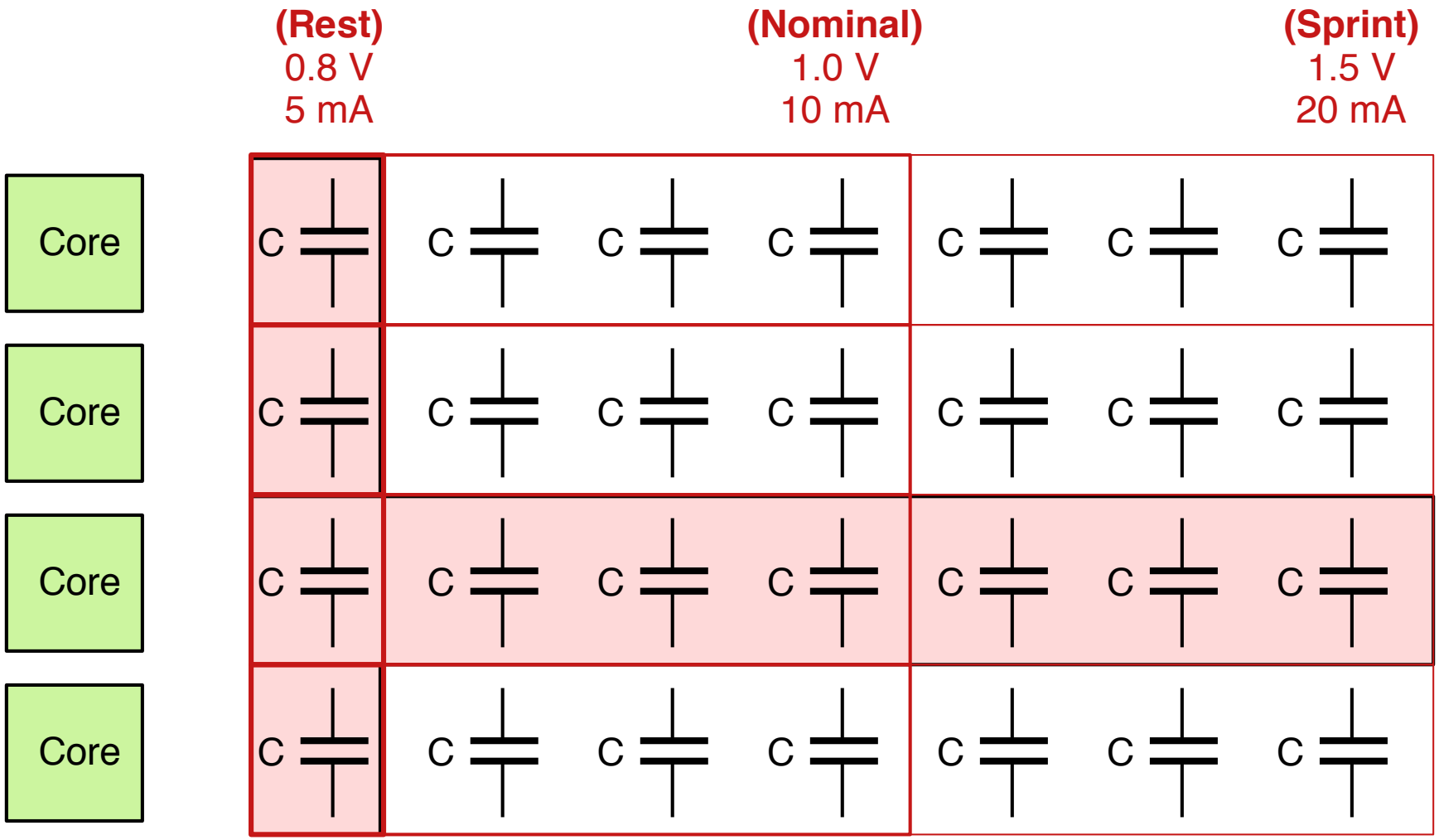
Key Architecture-Level Intuition



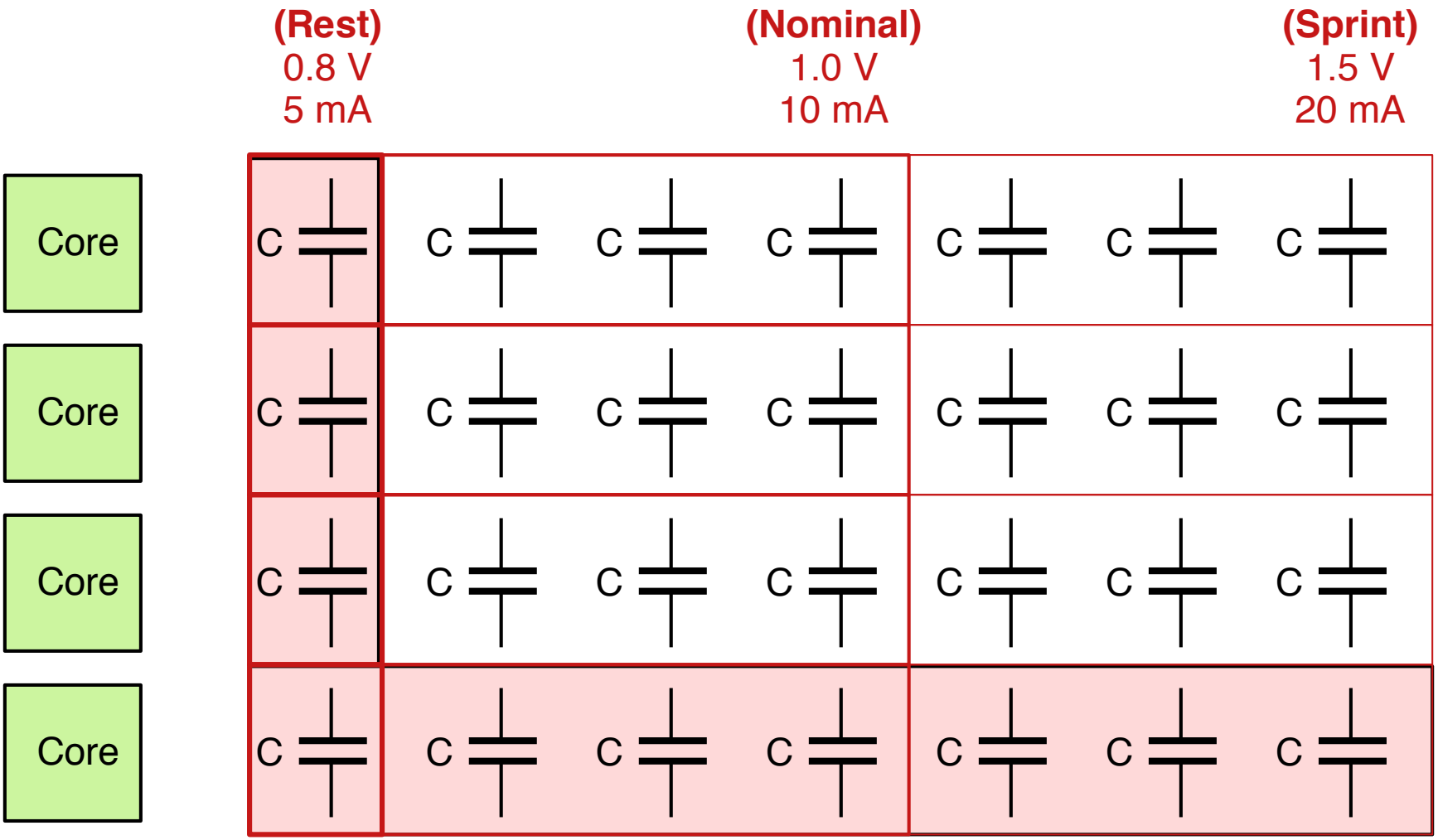
Key Architecture-Level Intuition



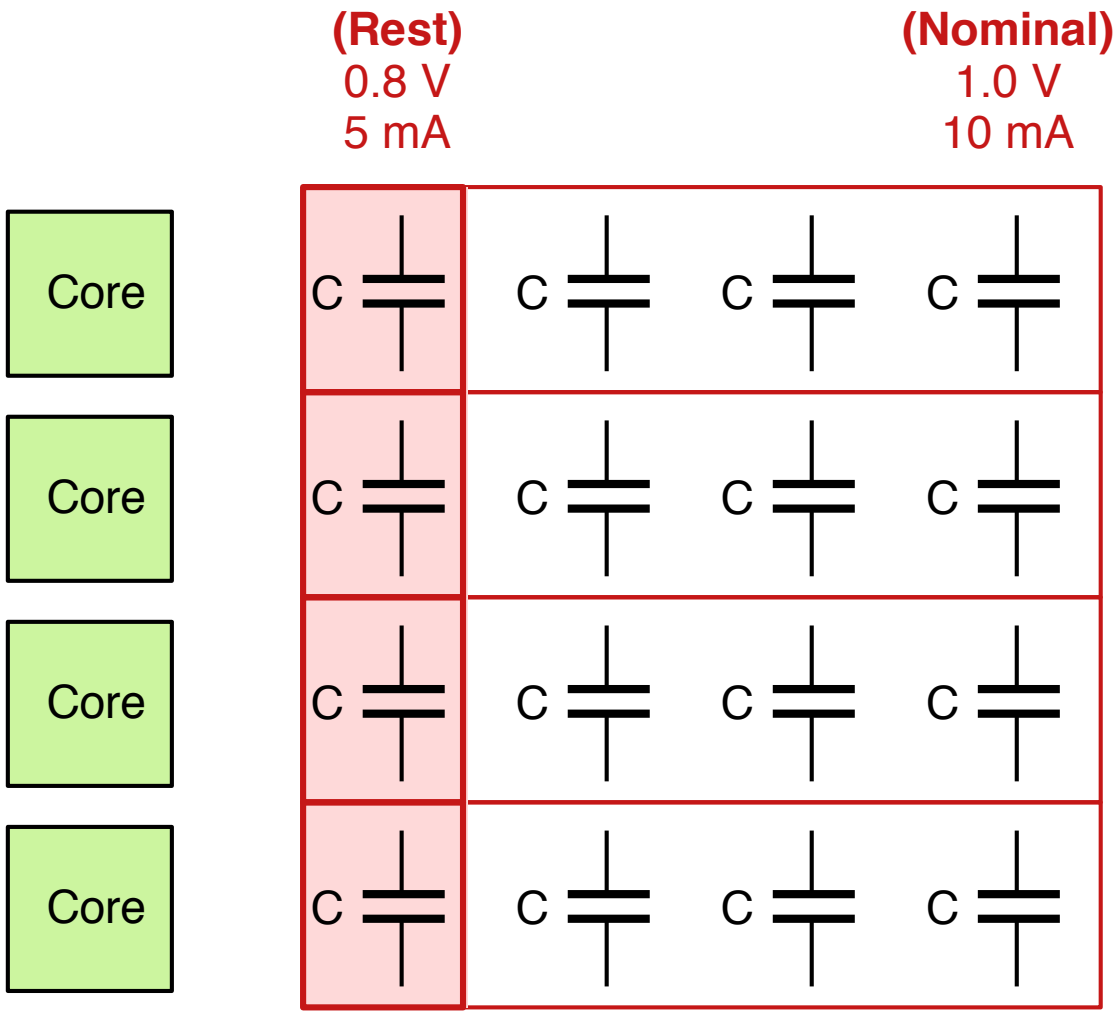
Key Architecture-Level Intuition



Key Architecture-Level Intuition



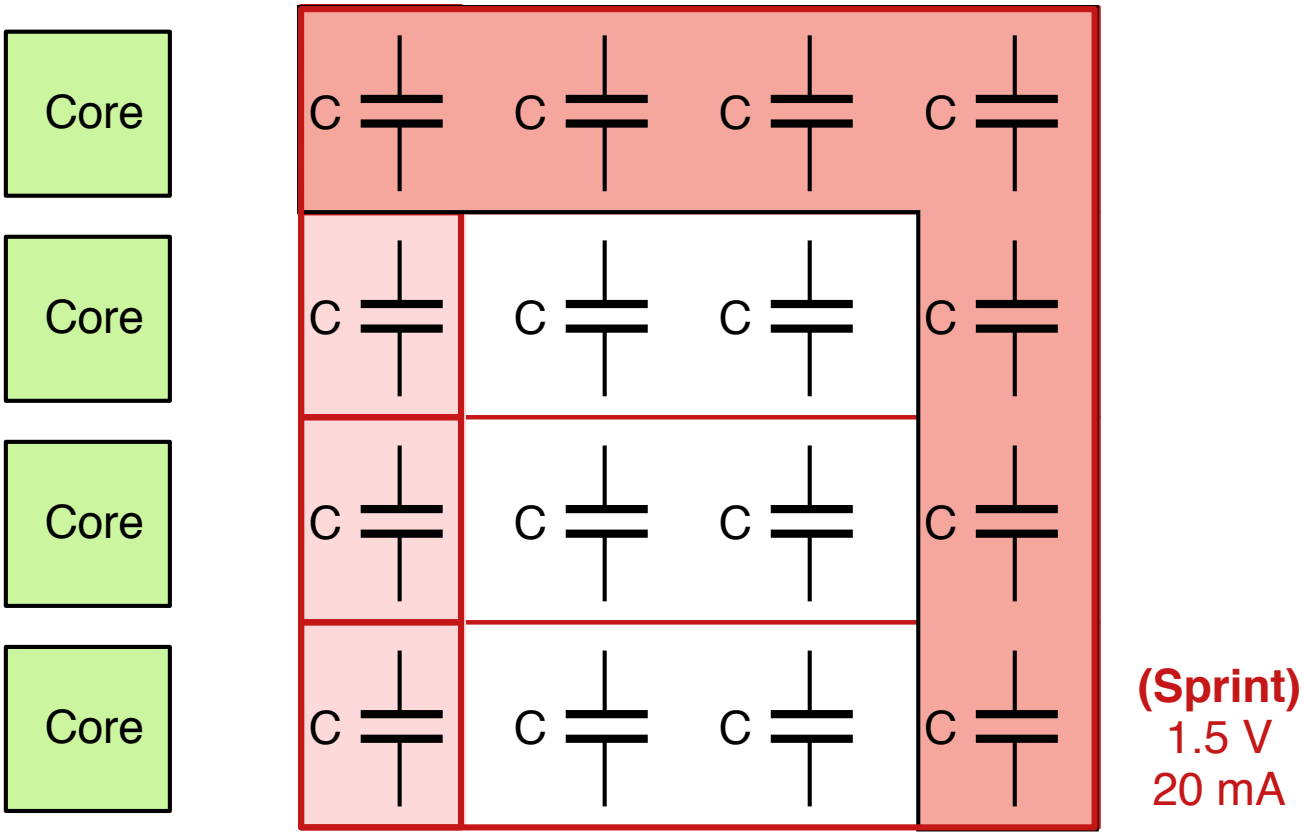
Key Architecture-Level Intuition



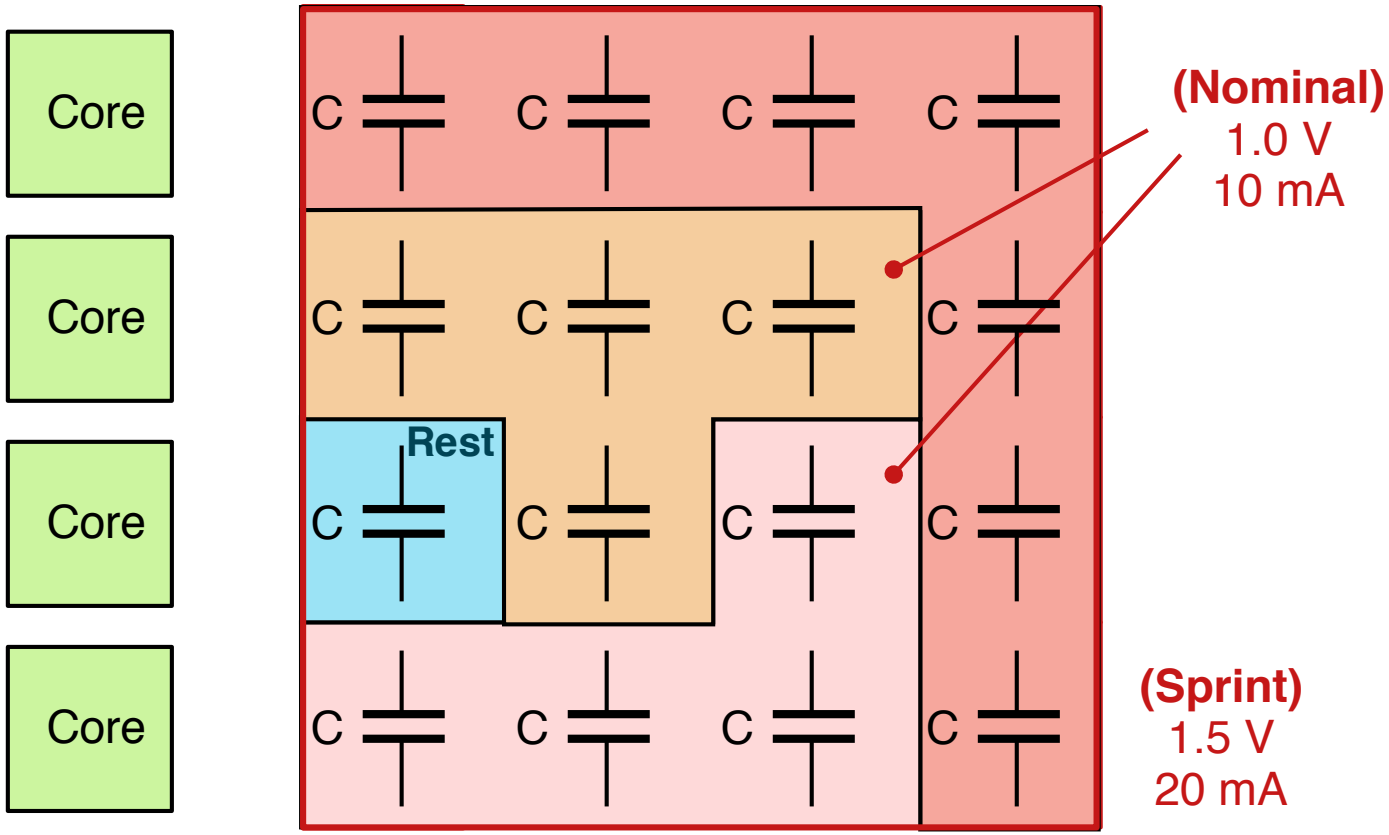
Key Goal

Reduce area without impacting performance and energy efficiency

Key Architecture-Level Intuition

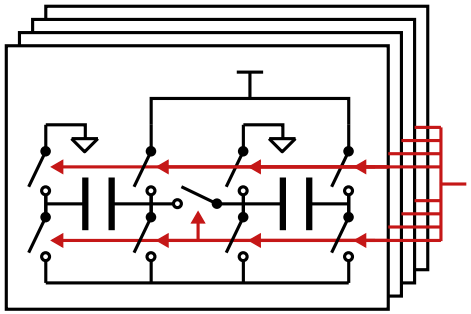


Key Architecture-Level Intuition

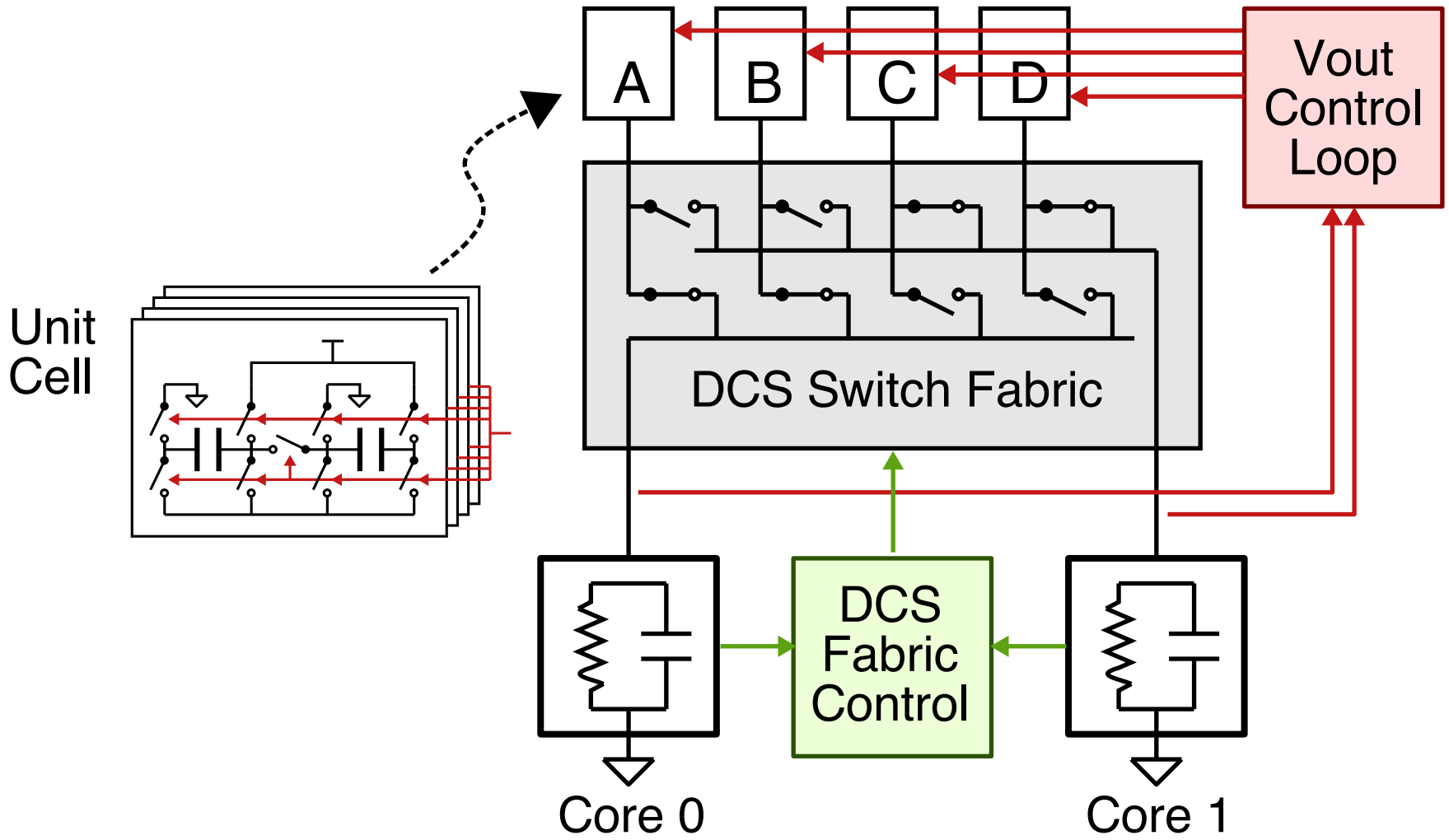


Dynamic Capacitance Sharing

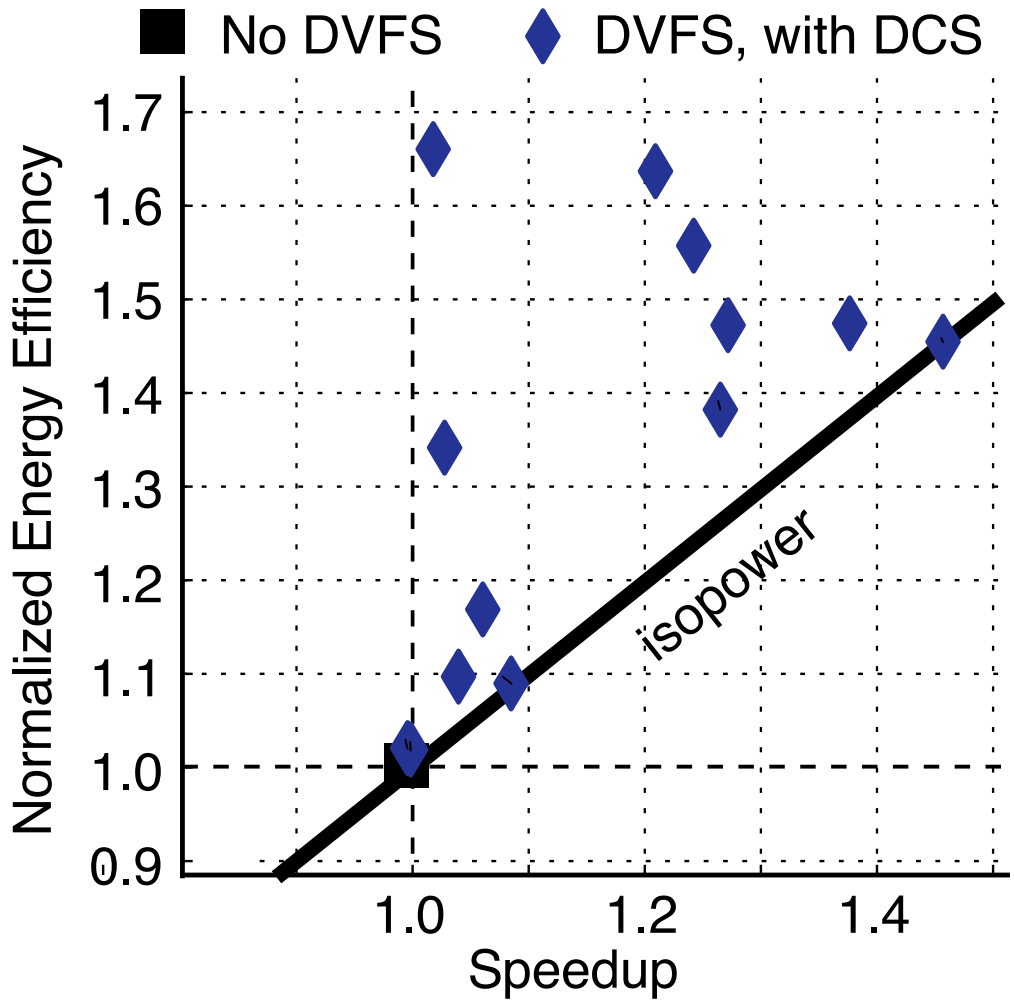
Unit Cell



Dynamic Capacitance Sharing



Evaluation

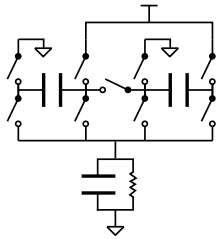


10-50% Speedup and 10-70% Energy Efficiency with Area-Optimized On-Chip VRs (40% area savings)

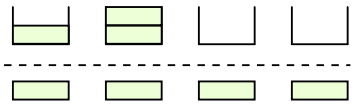
Benchmarks

- bfs
- bilateral
- dither
- kmeans
- mriq
- pbbs-dr
- pbbs-knn
- pbbs-mm
- rsort
- splash2-fft
- splash2-lu
- strsearch
- viterbi

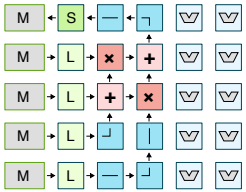
Software, Architecture, and VLSI Co-Design for Exploiting Fine-Grain Asymmetry



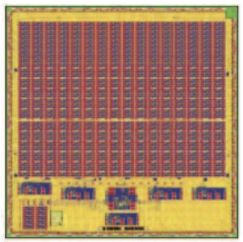
Exploiting Fine-Grain Asymmetry with
On-Chip Voltage Regulation - **MICRO'14**, IEEE TCAS I'18



Exploiting Fine-Grain Asymmetry in
Task-Based Parallel Runtimes - **ISCA'16**, MICRO'17, RISCV'18



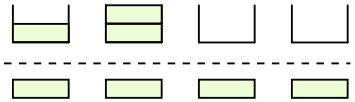
Exploiting Fine-Grain Asymmetry in
Coarse-Grain Reconfigurable Arrays - **Unpublished**



Exploiting Fine-Grain Asymmetry with
Silicon Prototyping - **IEEE MICRO'18**, DAC'18, Hotchips'17

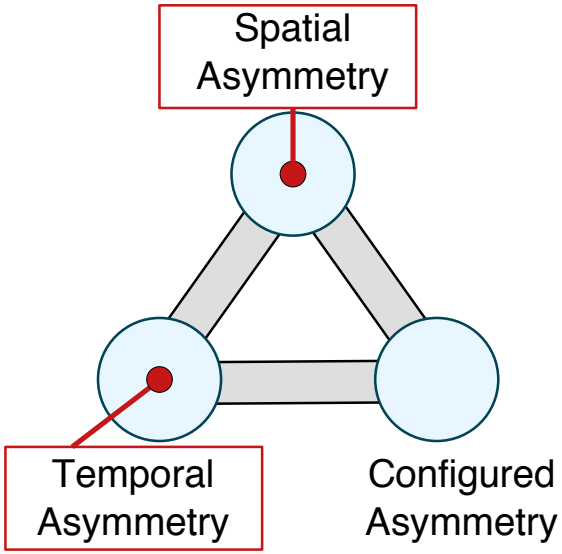
Conclusion

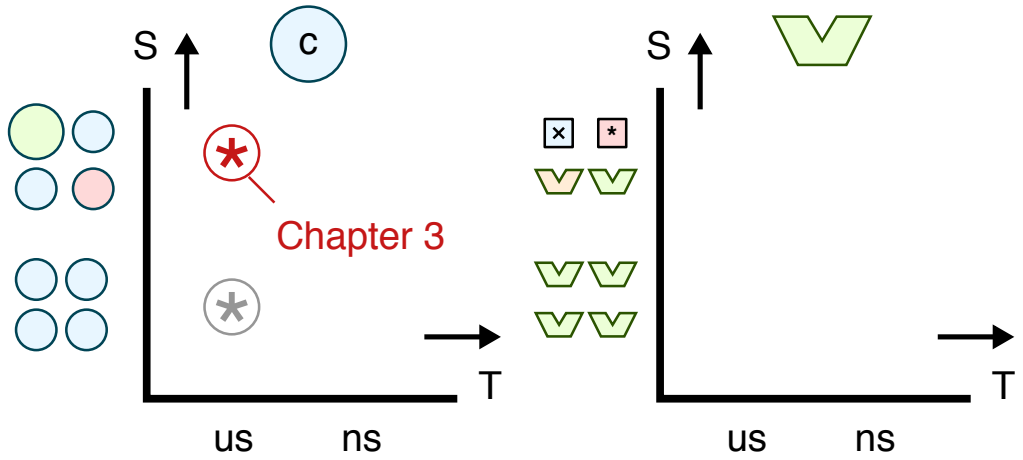
Software, Architecture, and VLSI Co-Design for Exploiting Fine-Grain Asymmetry



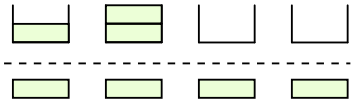
Exploiting Fine-Grain Asymmetry in

Task-Based Parallel Runtimes - ISCA'16, MICRO'17, RISCV'18



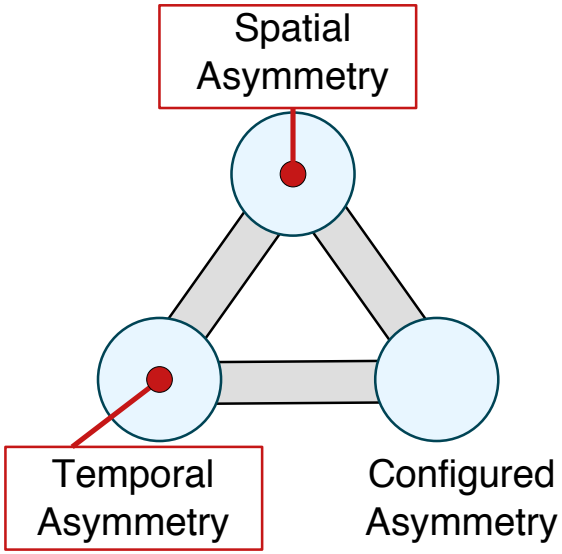


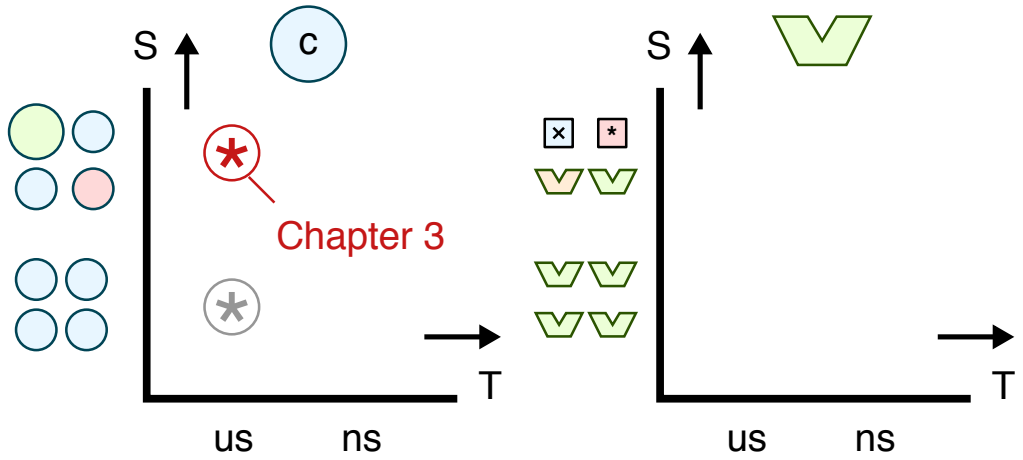
Software, Architecture, and VLSI Co-Design for Exploiting Fine-Grain Asymmetry



Exploiting Fine-Grain Asymmetry in

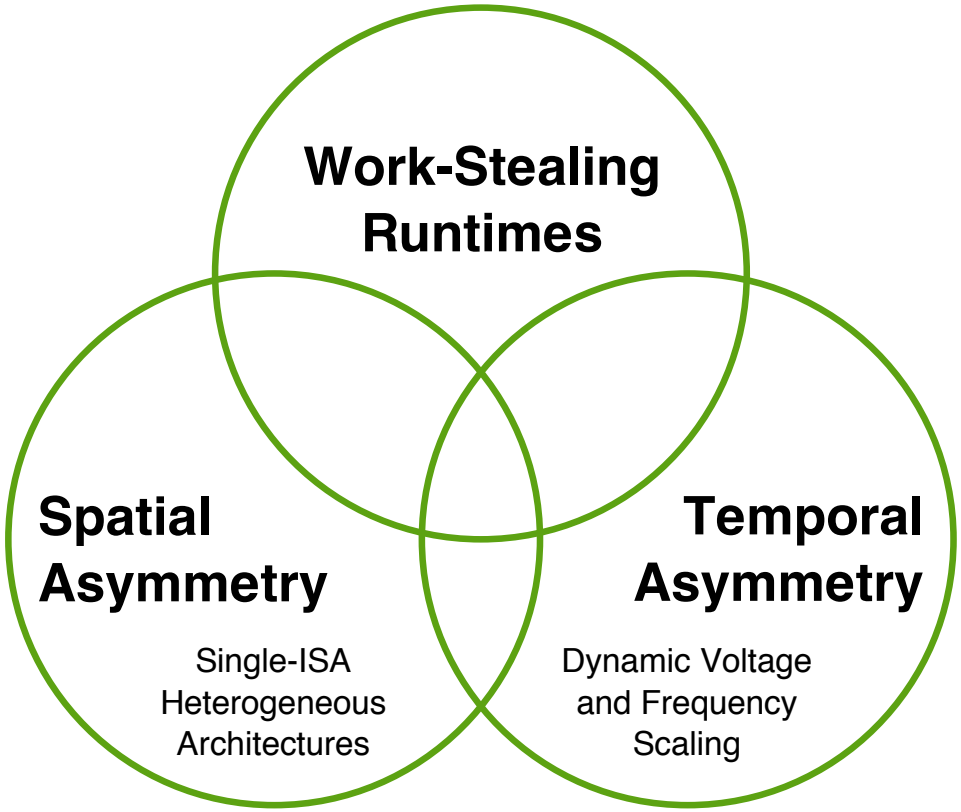
Task-Based Parallel Runtimes - ISCA'16, MICRO'17, RISCV'18





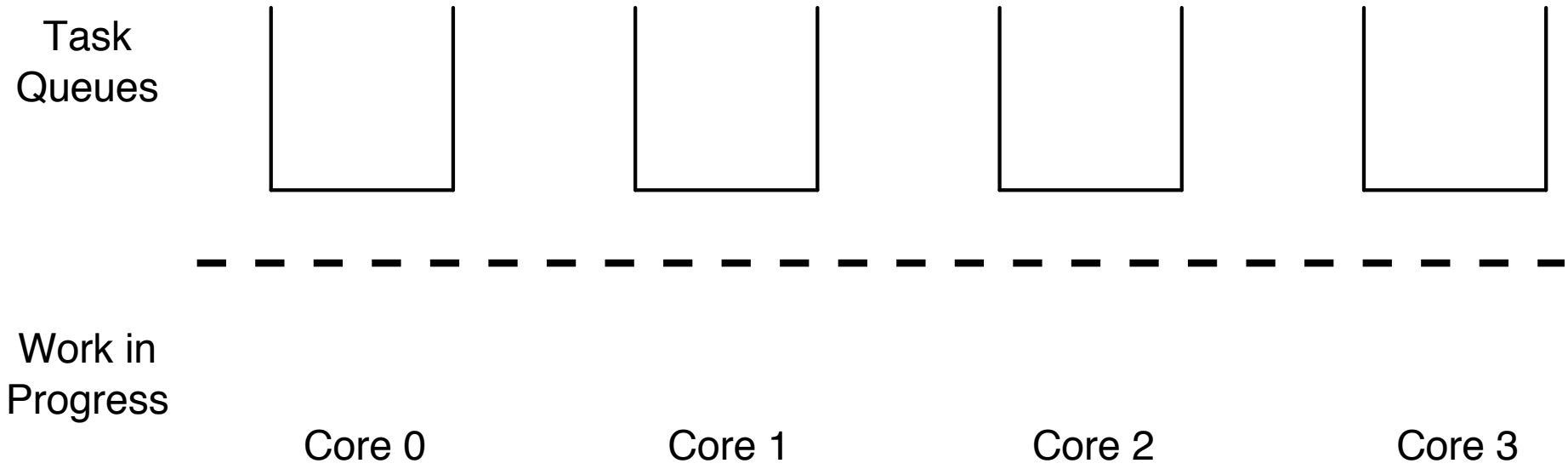
Research Question – Can we leverage fine-grain power control to enable more efficient task-based parallel runtimes executing on heterogeneous systems?

Focusing on Task-Based Parallel Runtimes

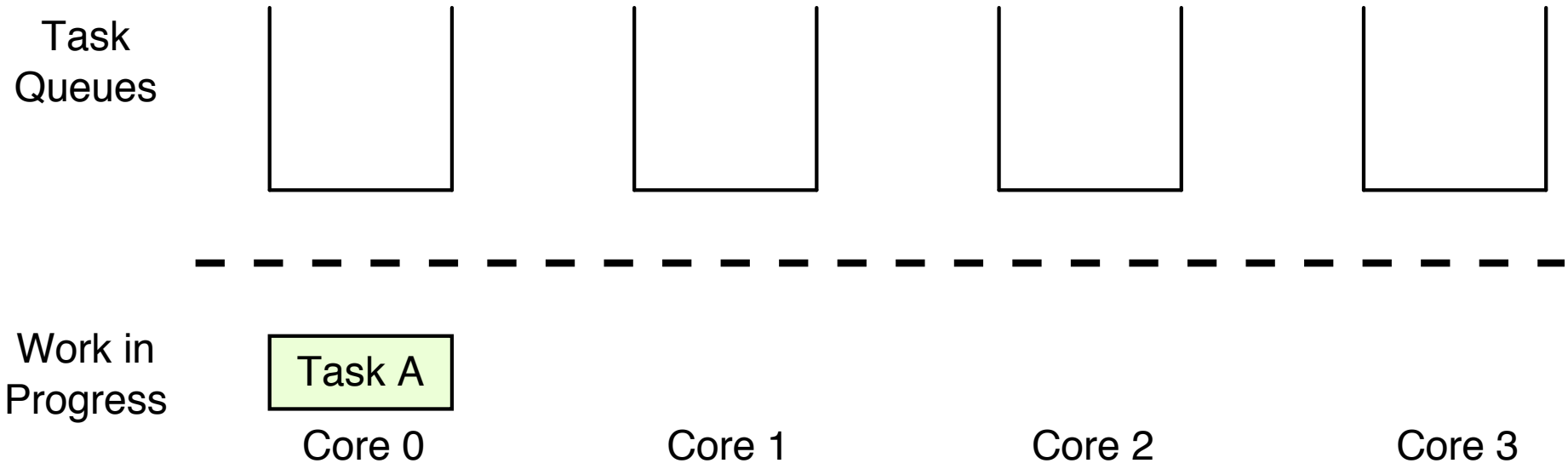


Research Question – Can we leverage fine-grain power control to enable more efficient task-based parallel runtimes executing on heterogeneous systems?

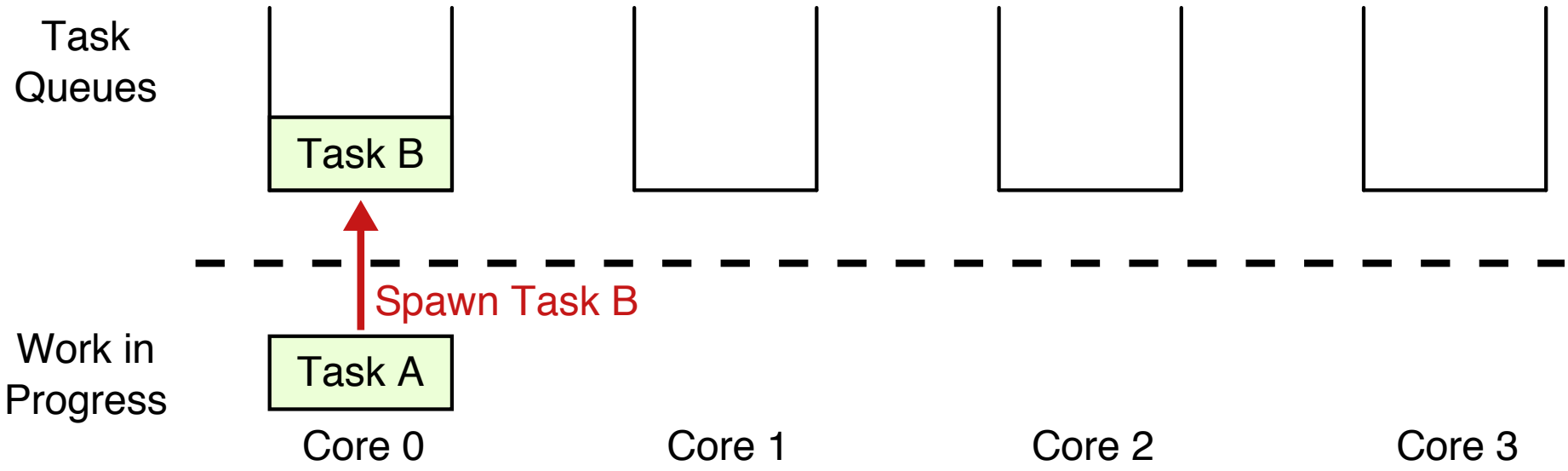
Work-Stealing Runtimes



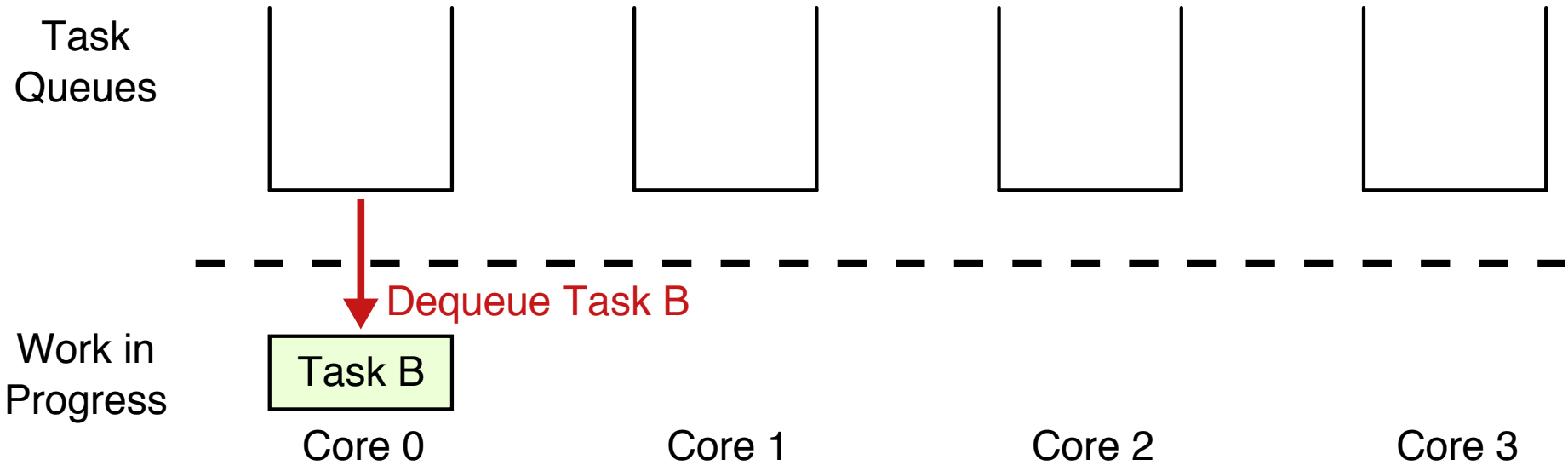
Work-Stealing Runtimes



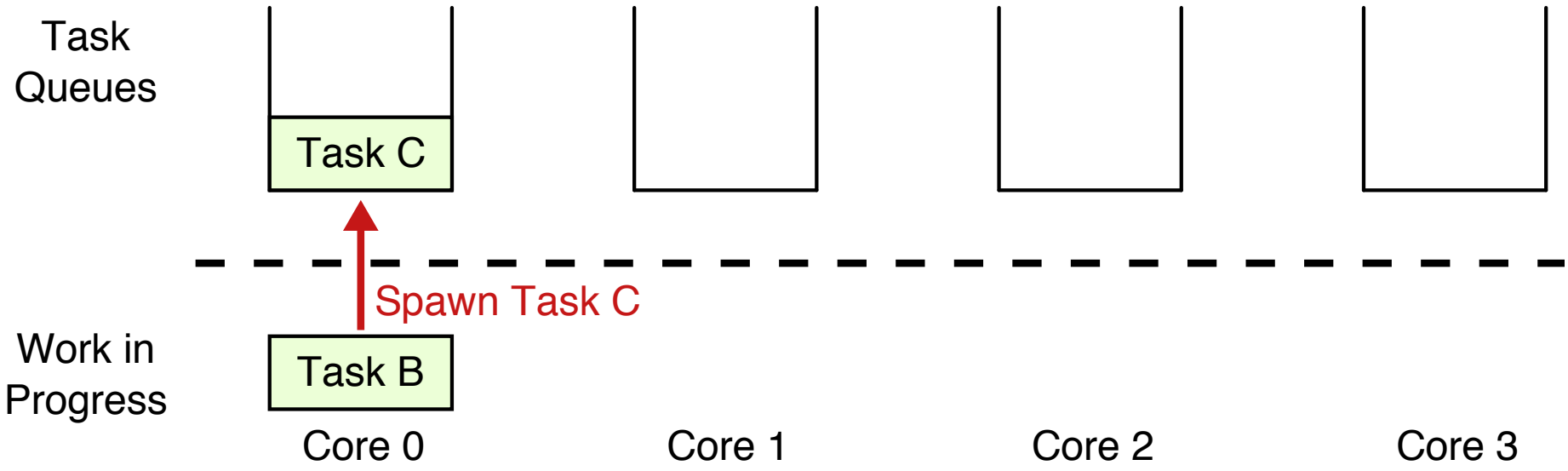
Work-Stealing Runtimes



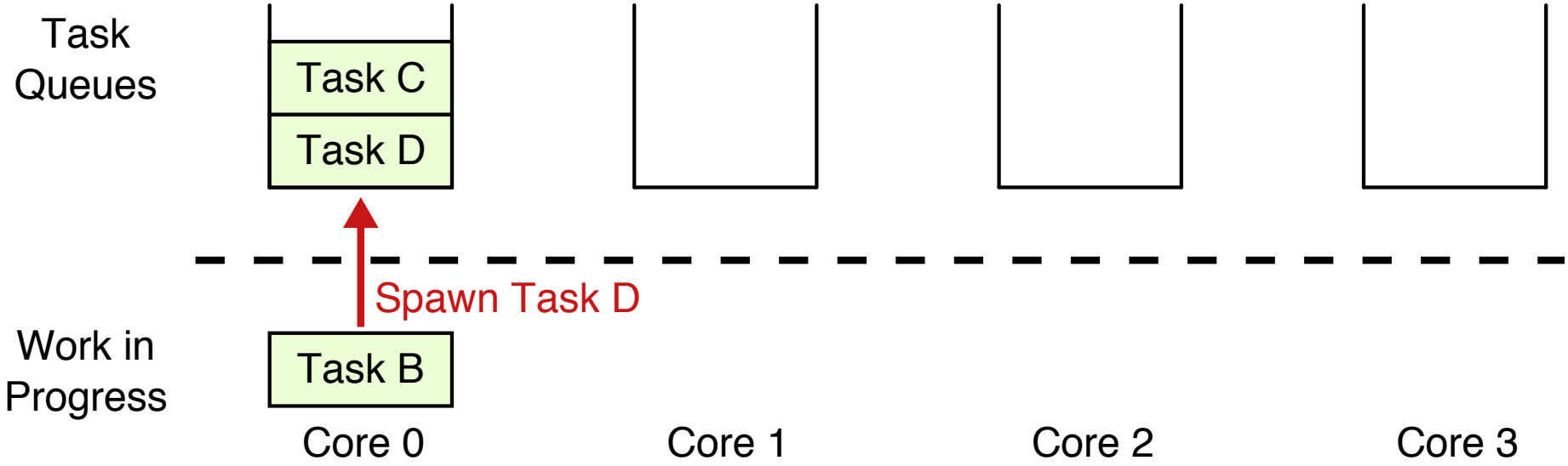
Work-Stealing Runtimes



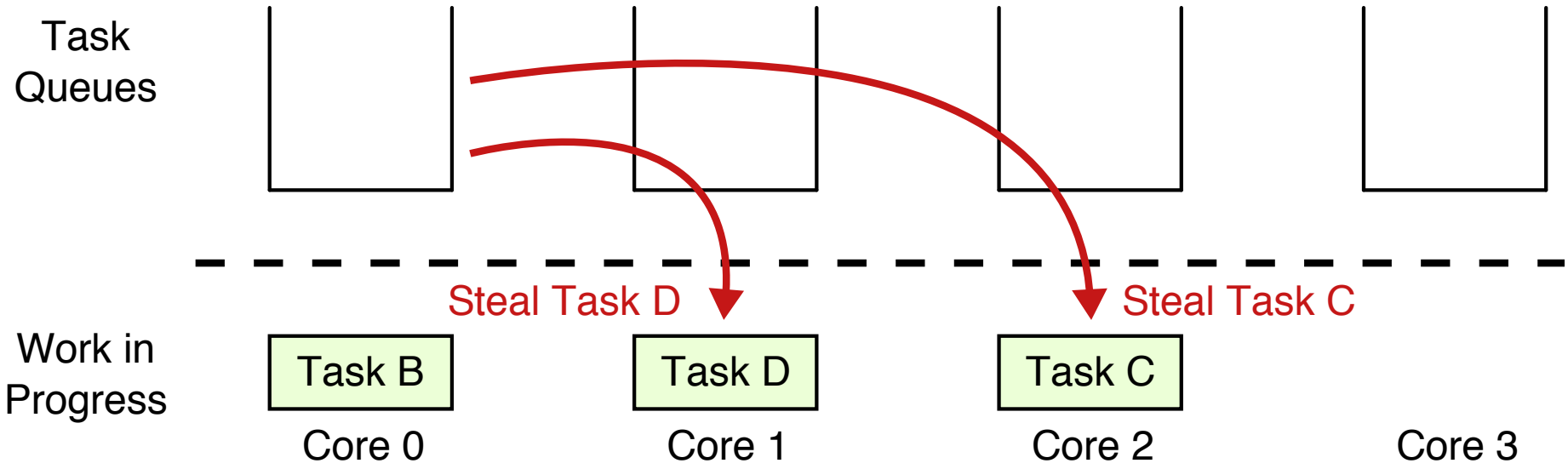
Work-Stealing Runtimes



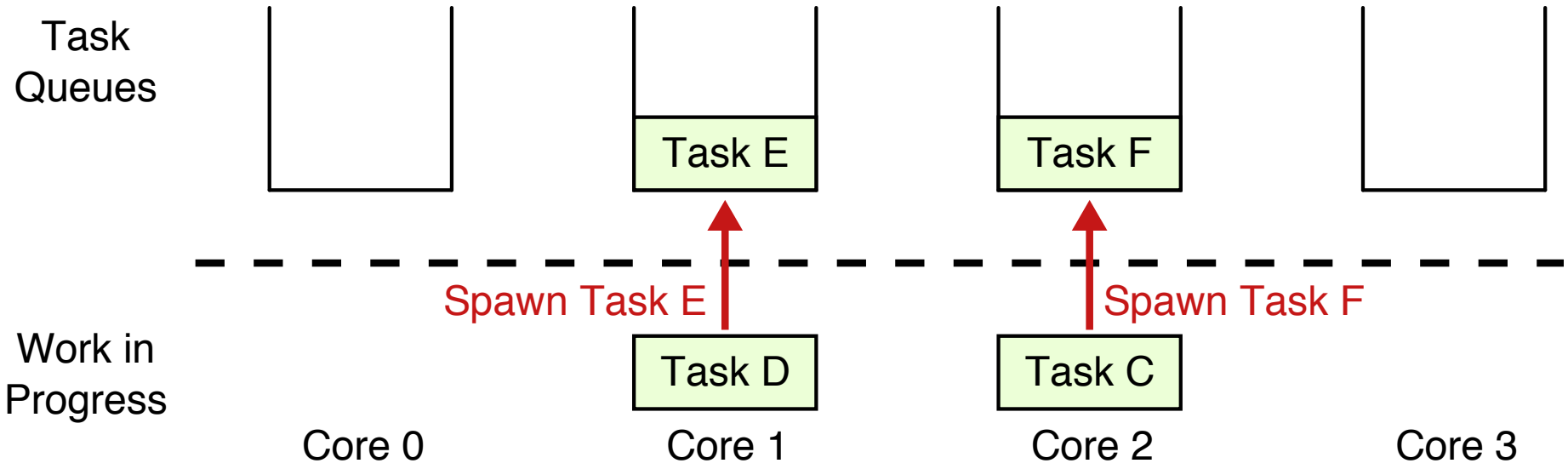
Work-Stealing Runtimes



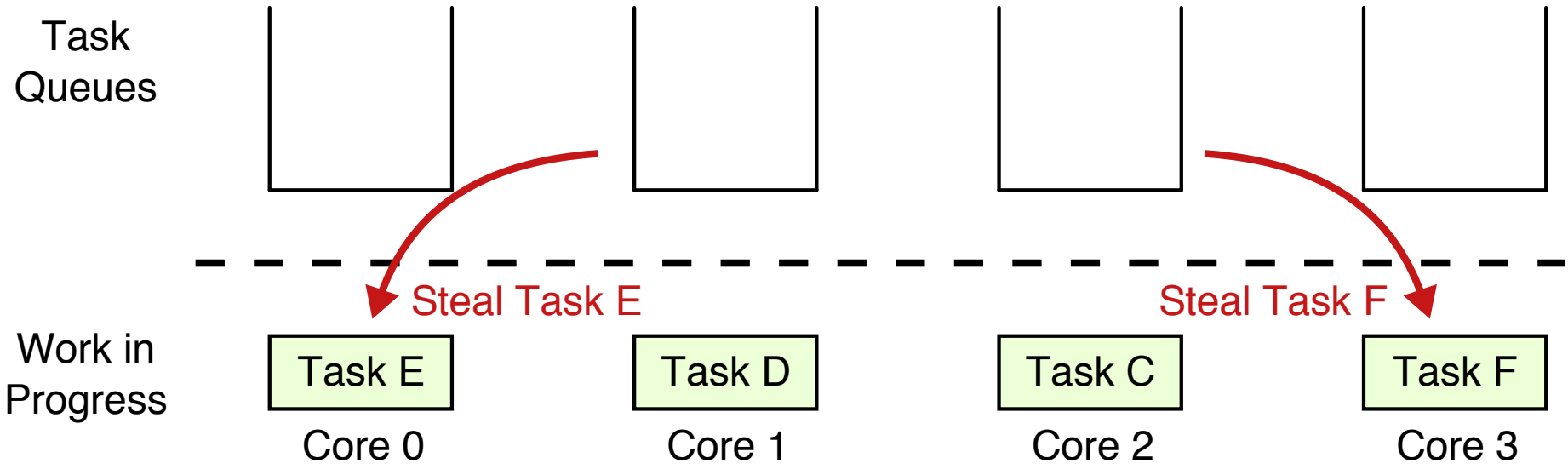
Work-Stealing Runtimes



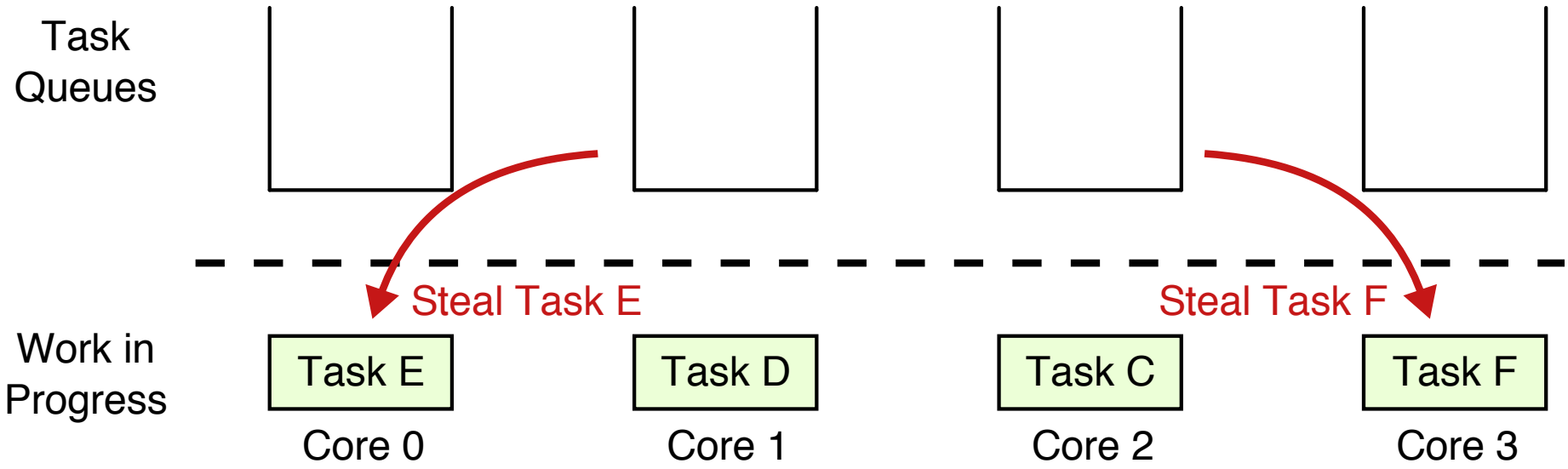
Work-Stealing Runtimes



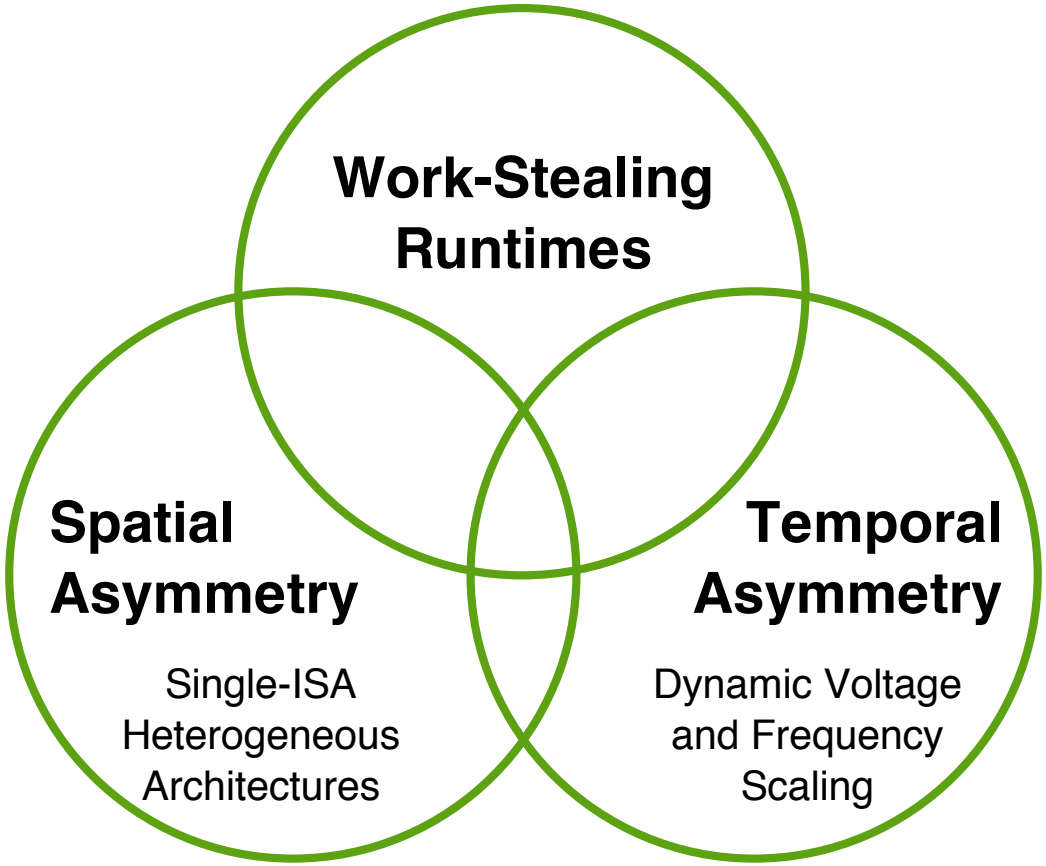
Work-Stealing Runtimes



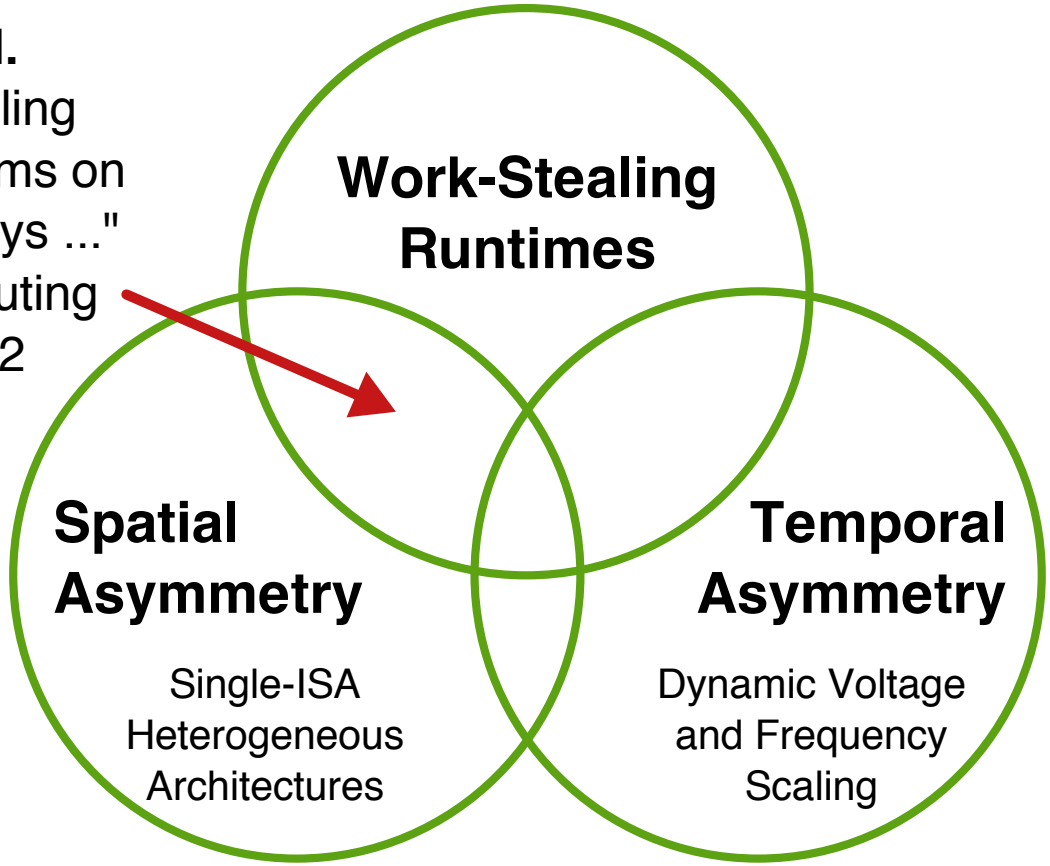
Work-Stealing Runtimes



- Work stealing has good performance, space requirements, and communication overheads in both theory and practice
- Supported in many popular concurrency platforms including: Intel's Cilk Plus, Intel's C++ TBB, Microsoft's .NET Task Parallel Library, Java's Fork/Join Framework, and OpenMP

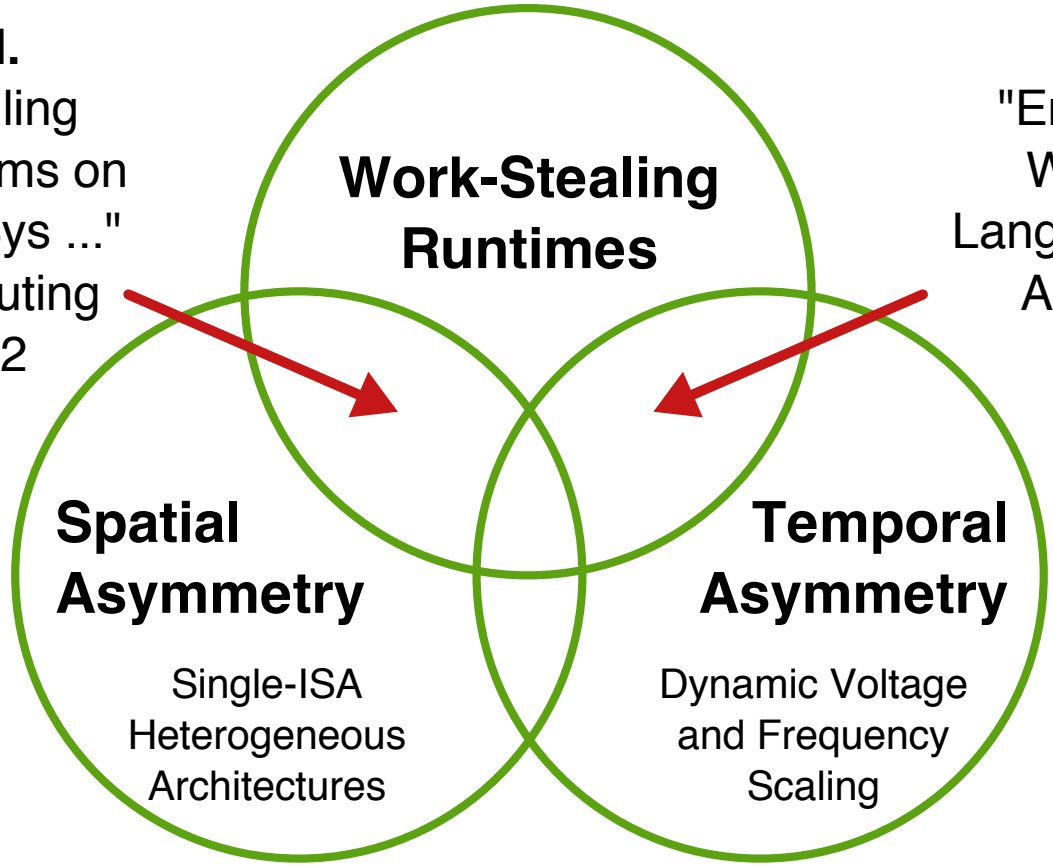

Work-Stealing Runtimes
Spatial Asymmetry
Single-ISA Heterogeneous Architectures
Temporal Asymmetry
Dynamic Voltage and Frequency Scaling

Bender et al.
"Online Scheduling of Parallel Programs on Heterogeneous Sys ..."
Theory of Computing Systems 2002



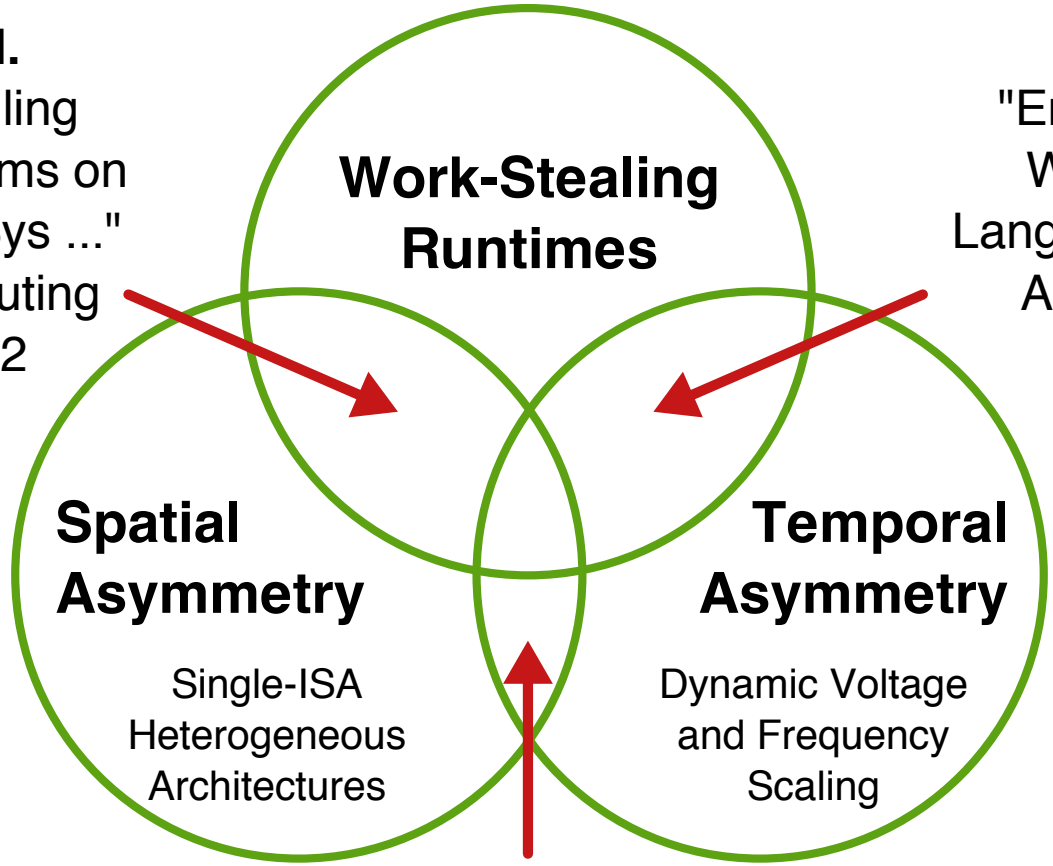
Bender et al.
"Online Scheduling of Parallel Programs on Heterogeneous Sys ..."
Theory of Computing Systems 2002

Ribic et al.
"Energy-Efficient Work-Stealing Language Runtimes"
ASPLOS 2014



Bender et al.
"Online Scheduling of Parallel Programs on Heterogeneous Sys ..." Theory of Computing Systems 2002

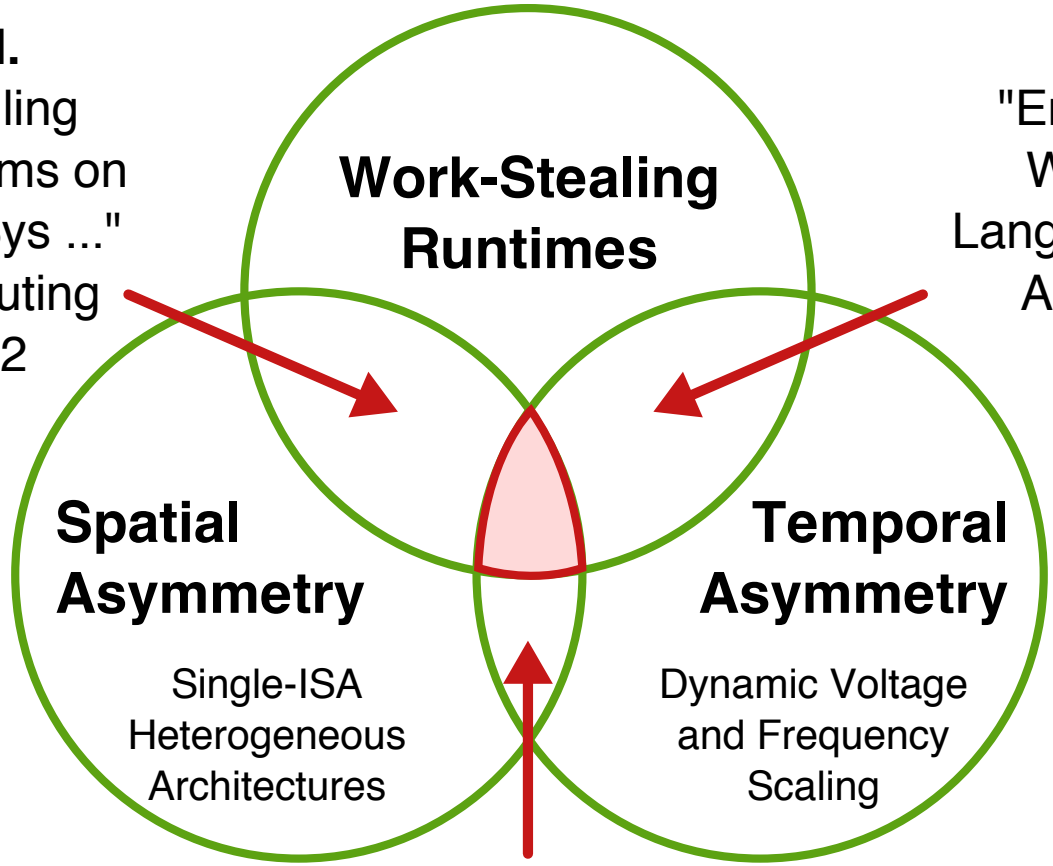
Ribic et al.
"Energy-Efficient Work-Stealing Language Runtimes" ASPLOS 2014



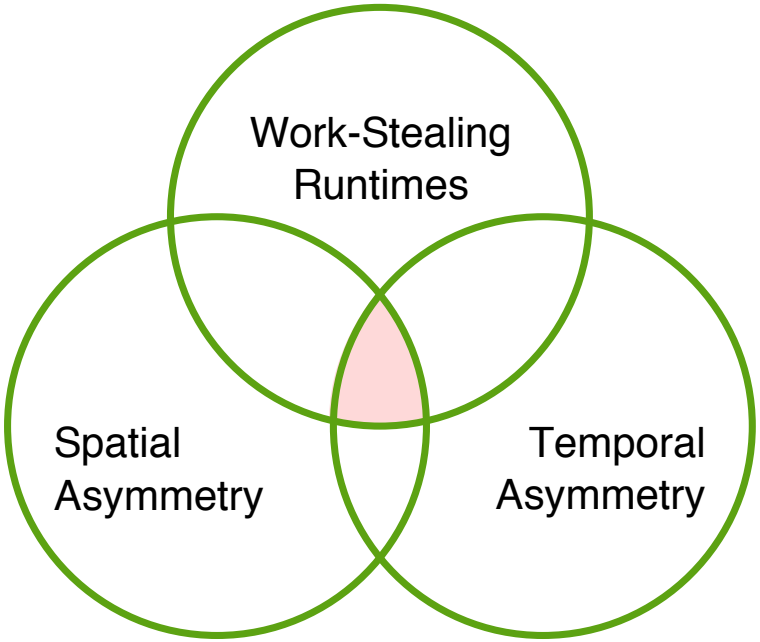
Azizi et al.
"Energy-performance Tradeoffs in Processor Architecture and Circuit Design: A Marginal Cost Analysis" ISCA 2010

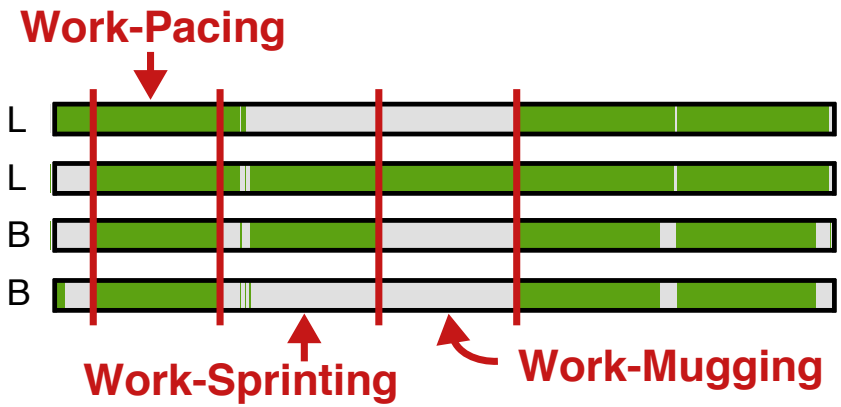
Bender et al.
"Online Scheduling of Parallel Programs on Heterogeneous Sys ..." Theory of Computing Systems 2002

Ribic et al.
"Energy-Efficient Work-Stealing Language Runtimes" ASPLOS 2014



Azizi et al.
"Energy-performance Tradeoffs in Processor Architecture and Circuit Design: A Marginal Cost Analysis" ISCA 2010

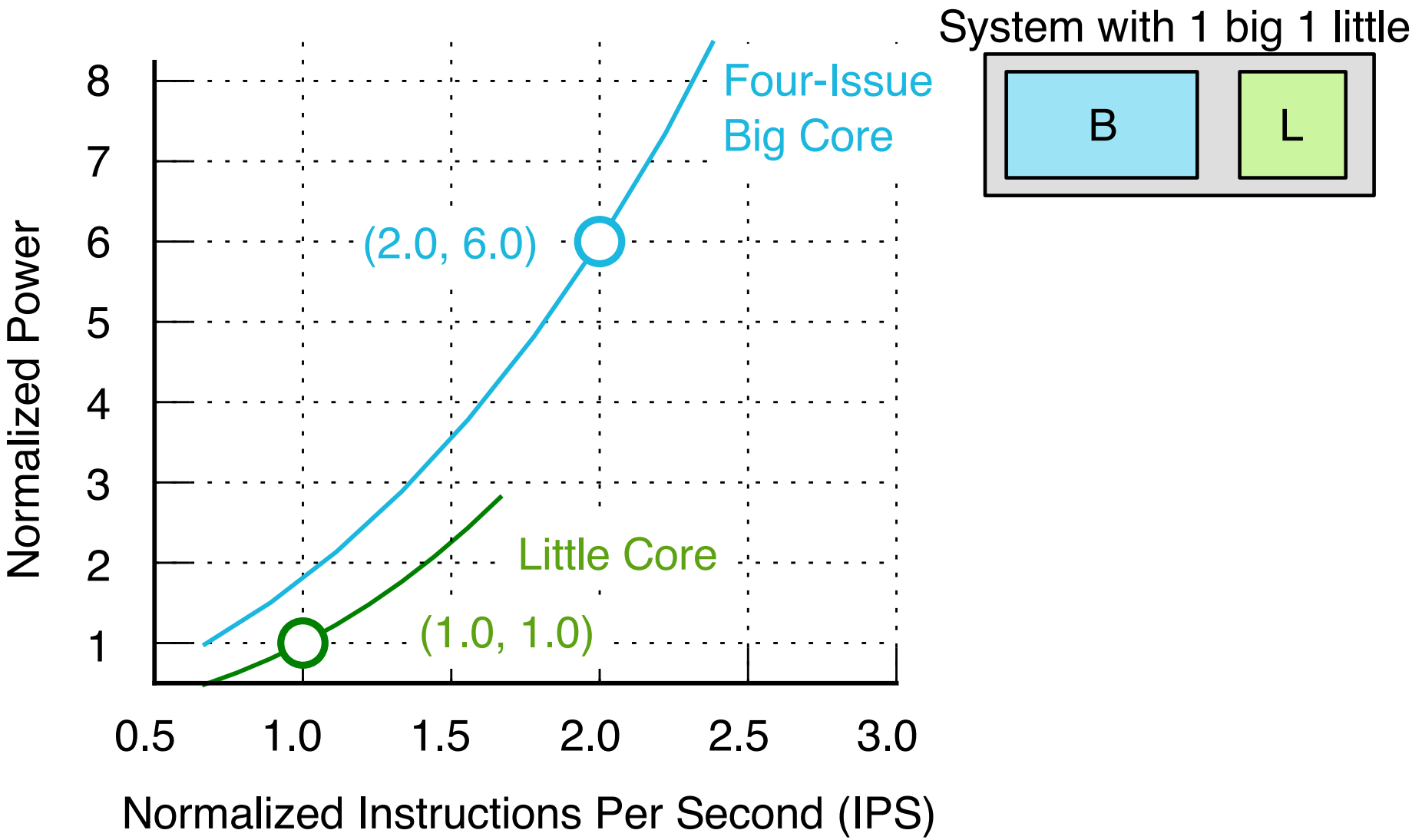




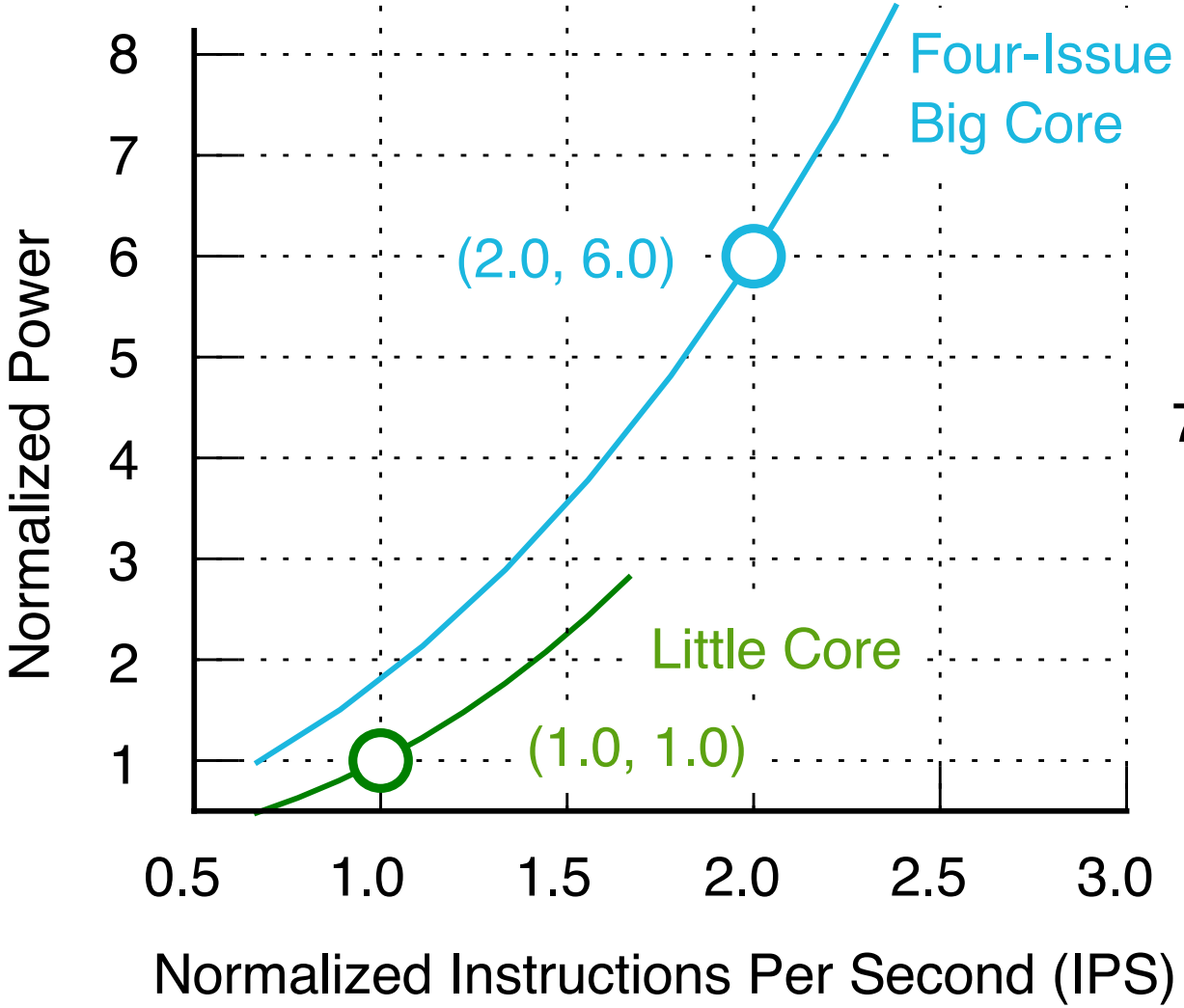
Exploiting Fine-Grain Asymmetry for Task-Based Parallelism

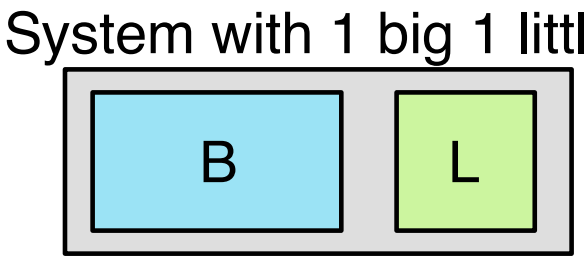
- Intuition
- Work-Pacing
- Results

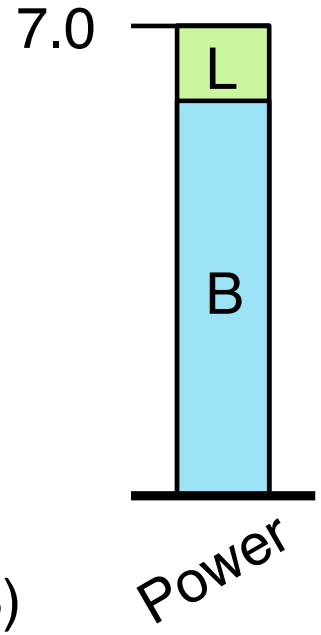
Building Intuition by Exploring a 1 Big 1 Little System



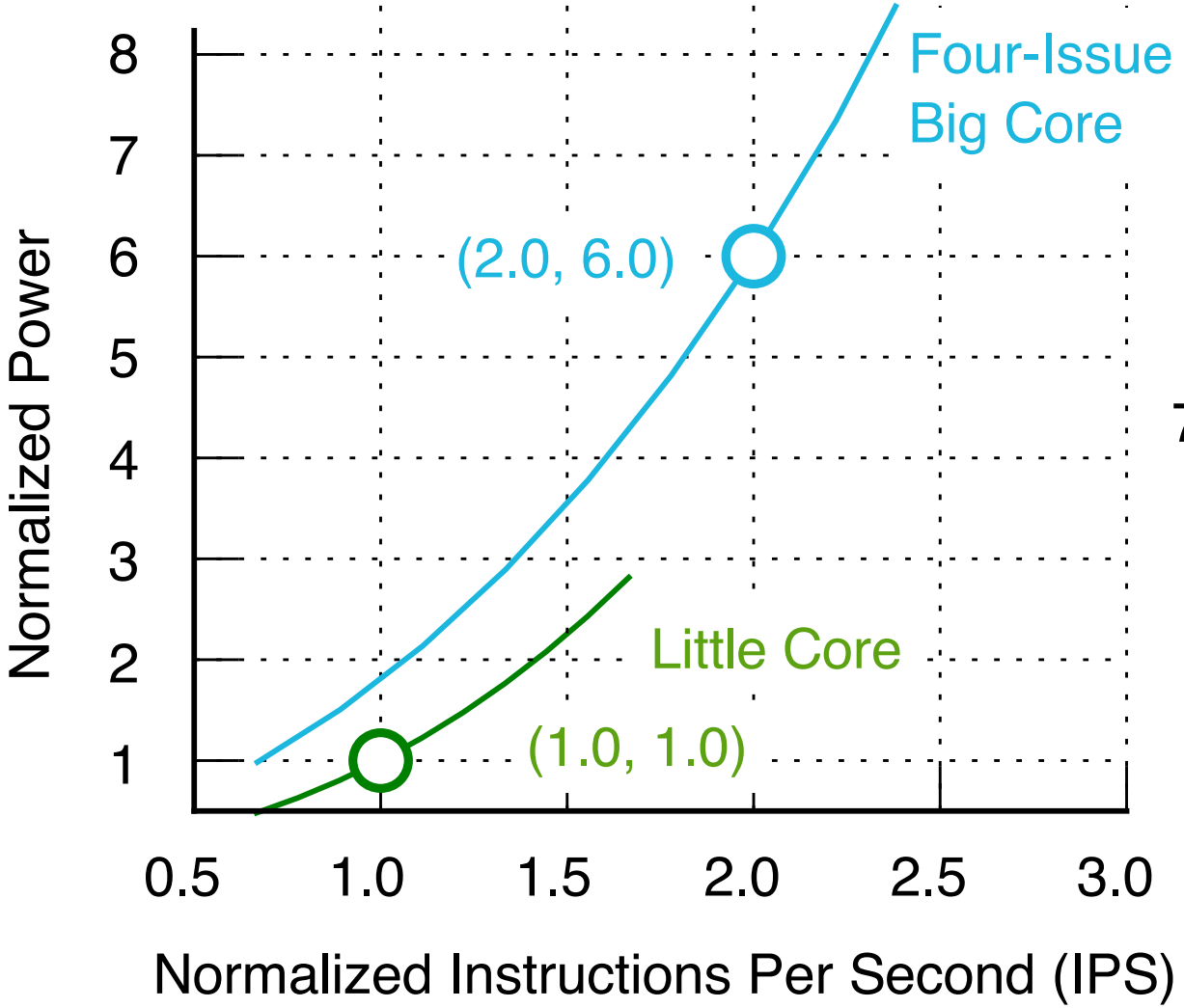
Building Intuition by Exploring a 1 Big 1 Little System

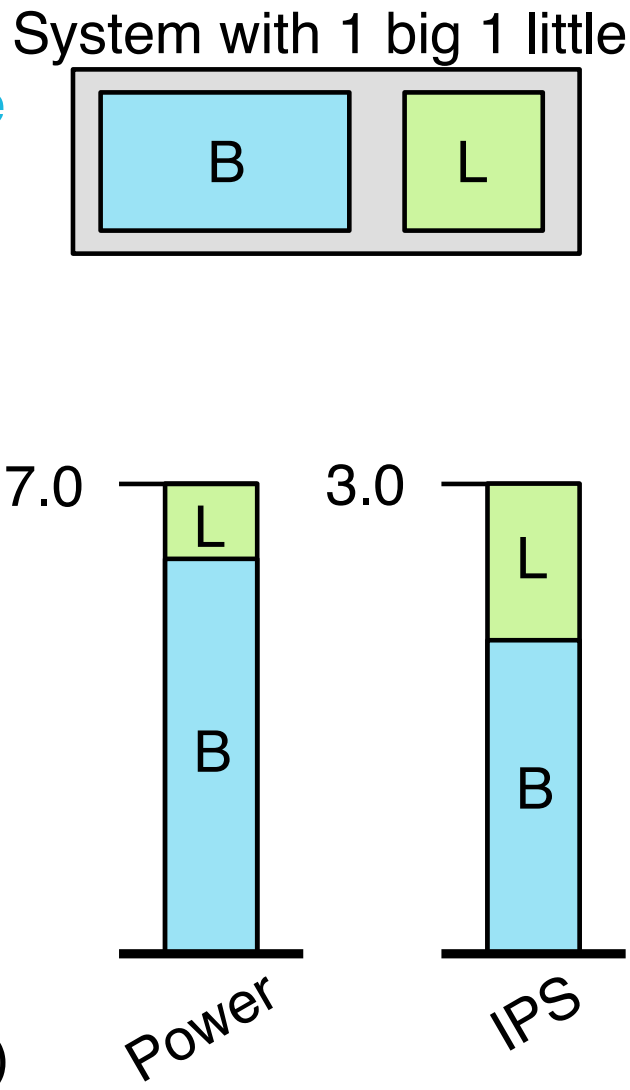




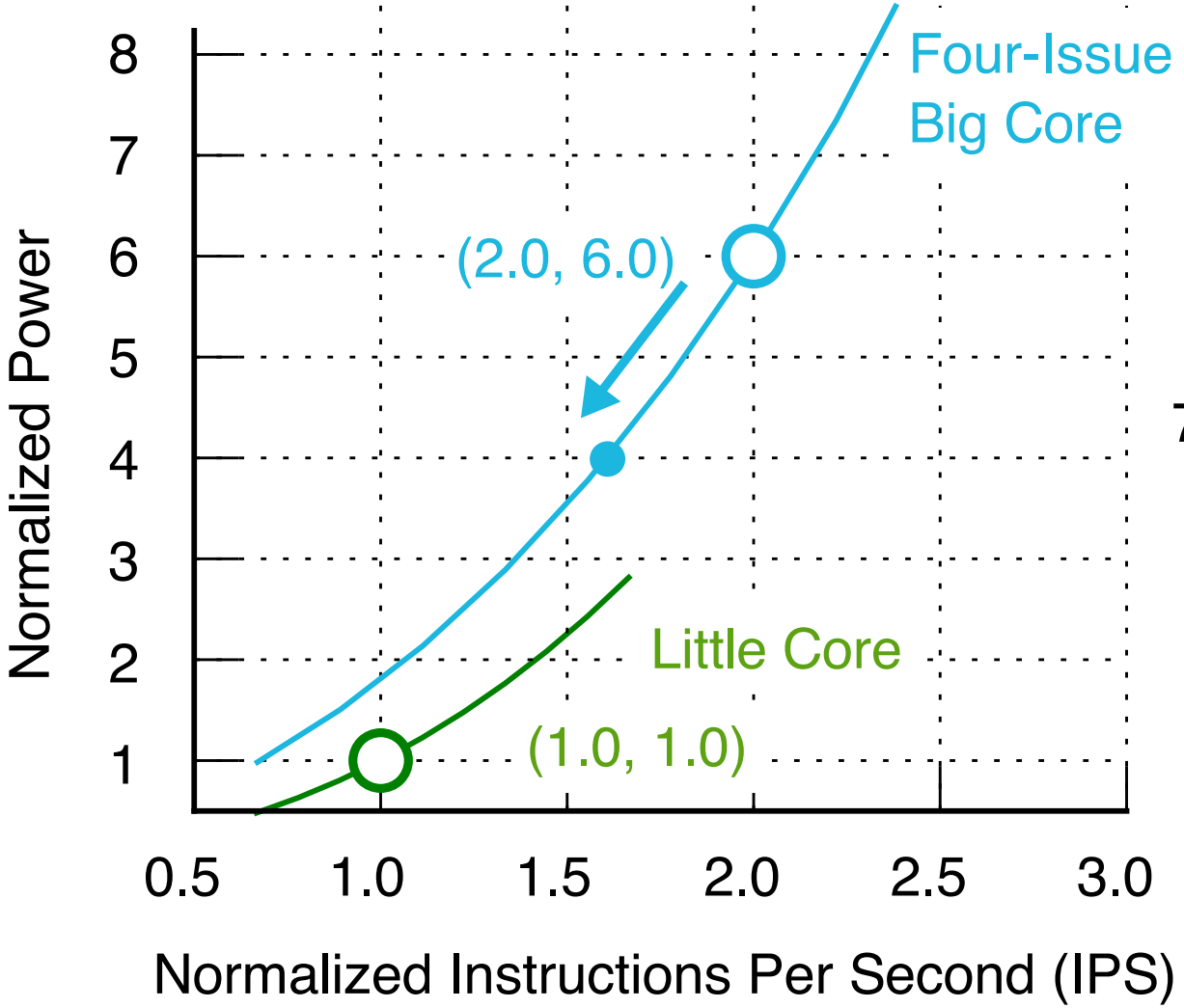


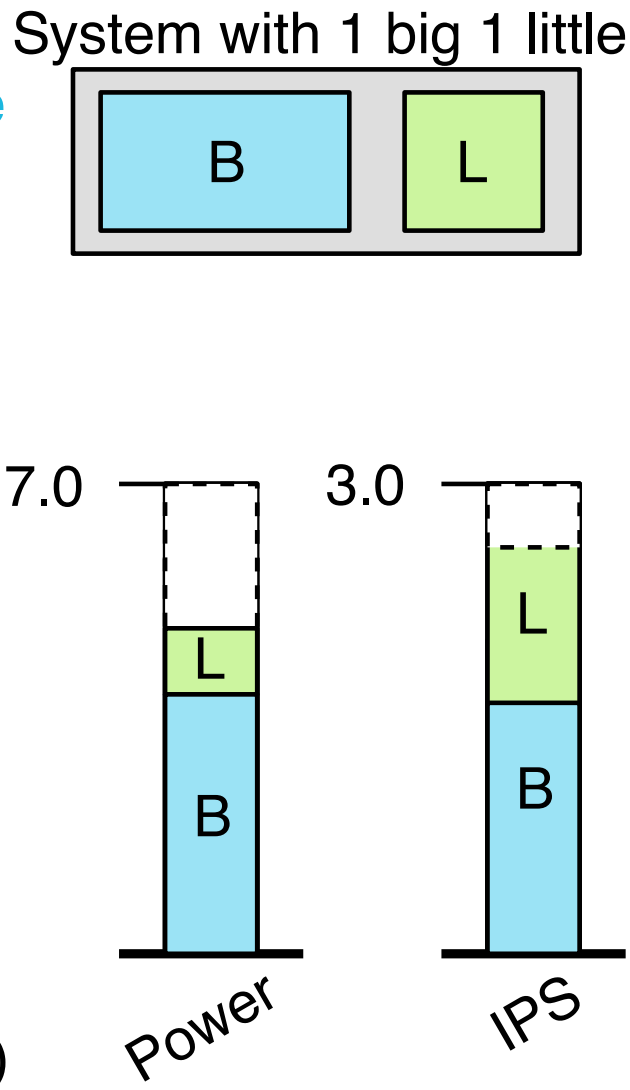
Building Intuition by Exploring a 1 Big 1 Little System



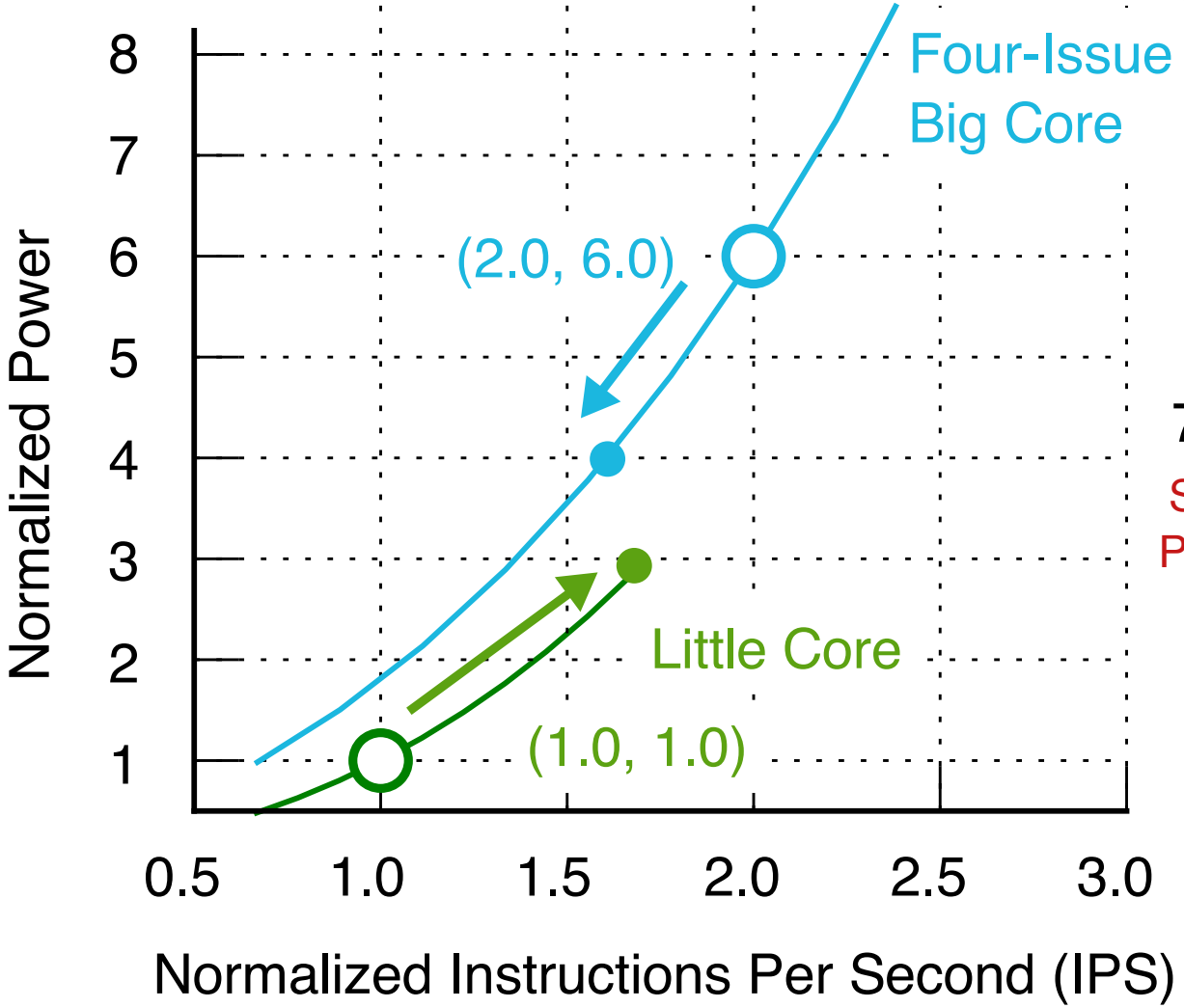


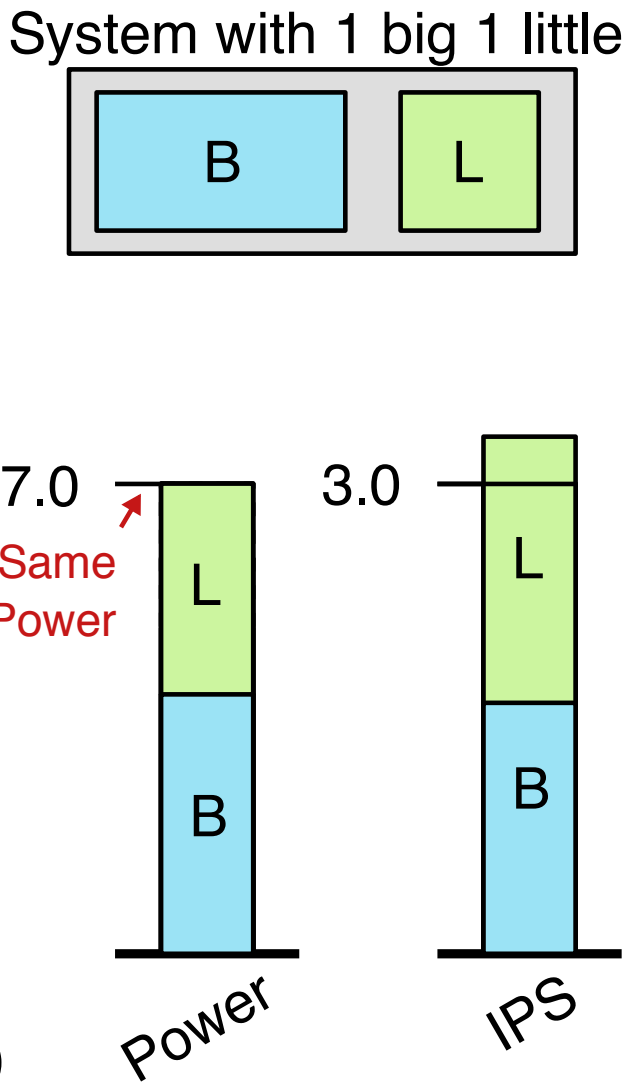
Building Intuition by Exploring a 1 Big 1 Little System



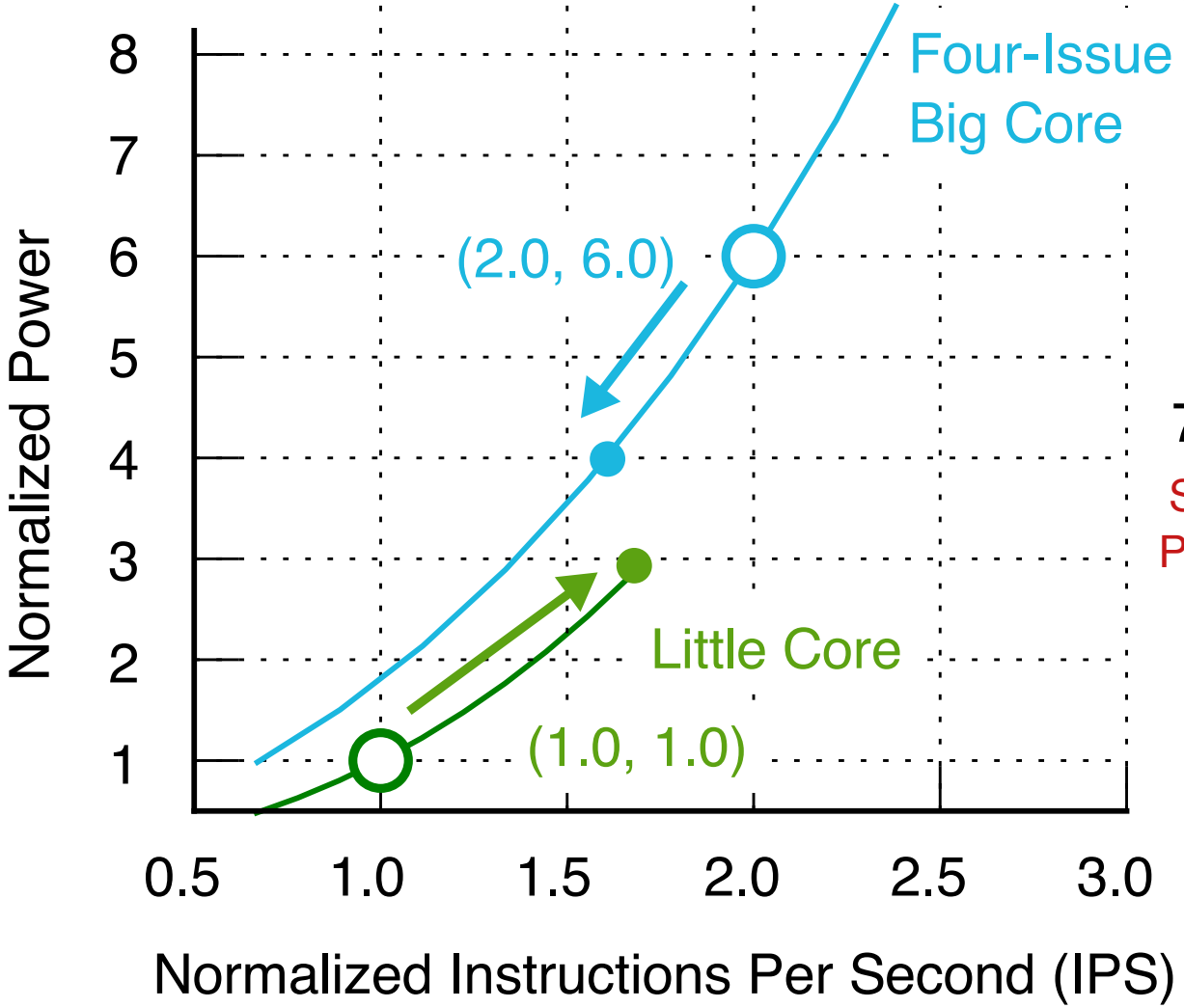


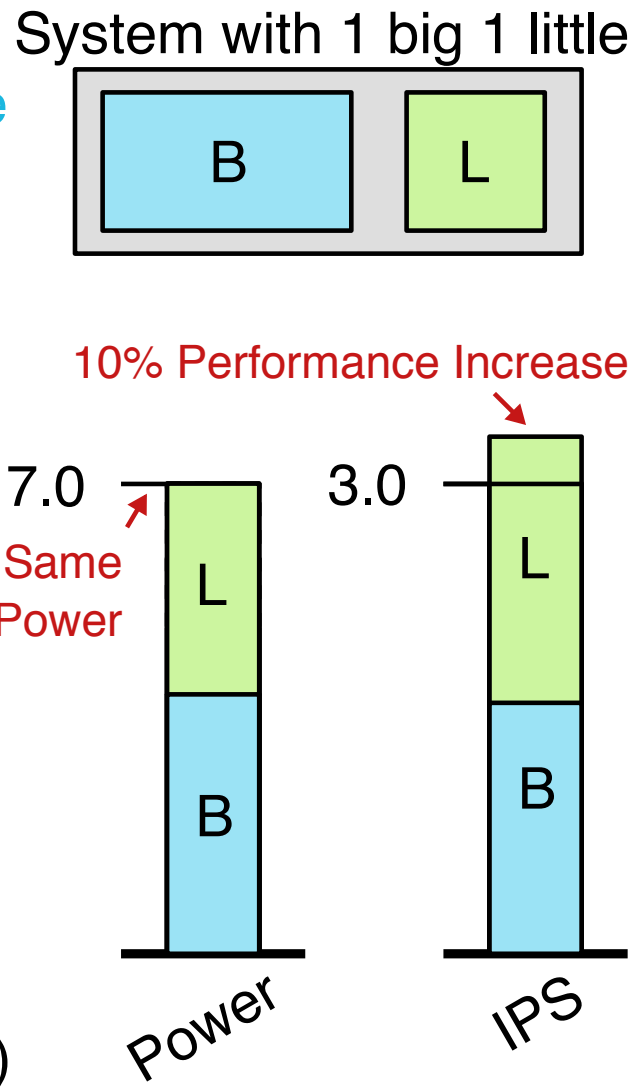
Building Intuition by Exploring a 1 Big 1 Little System



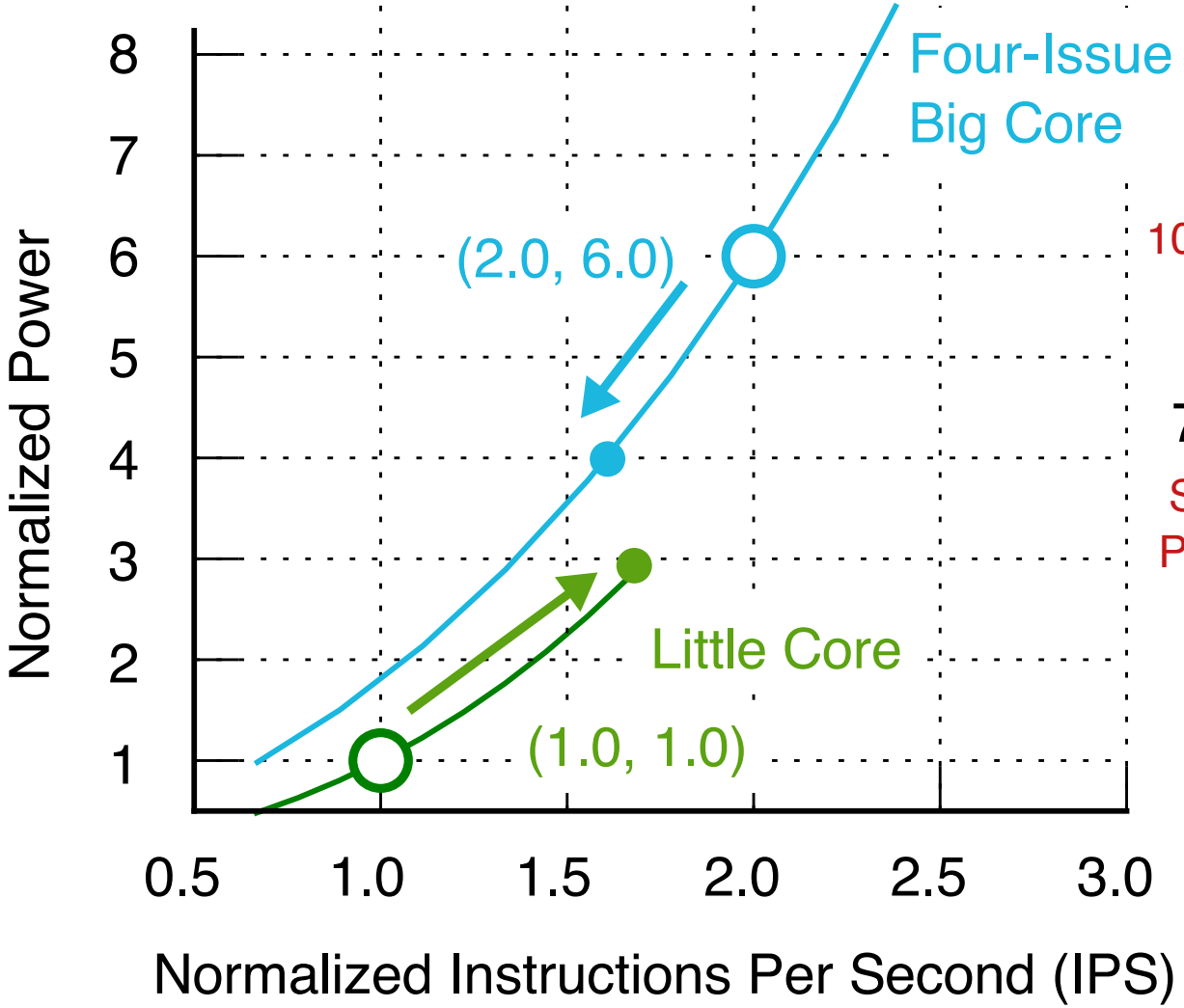


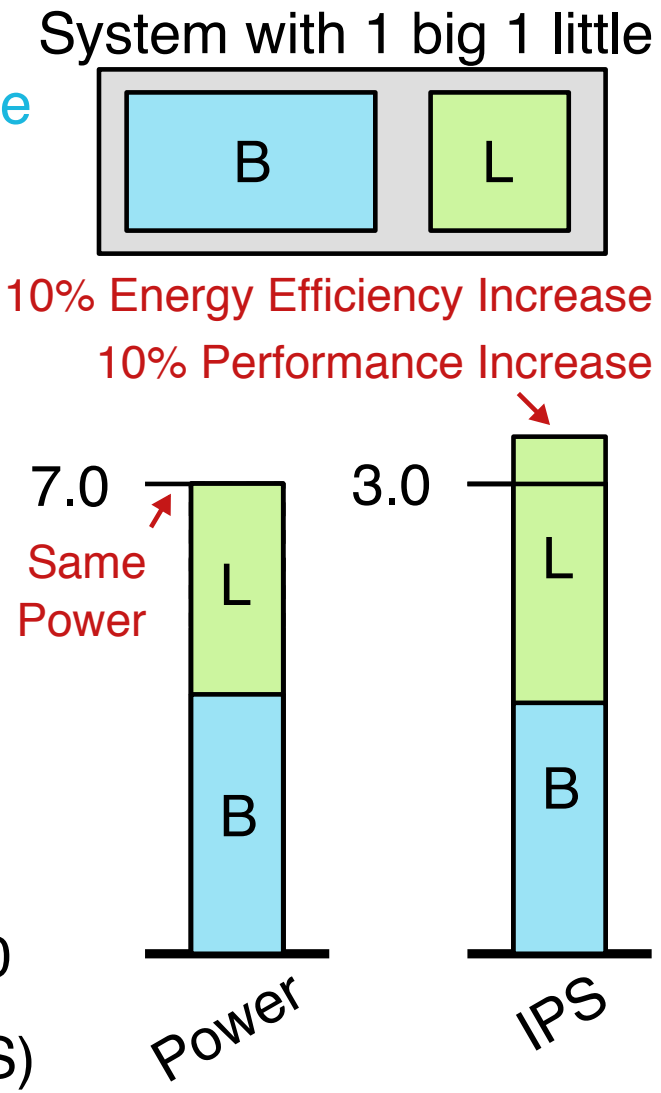
Building Intuition by Exploring a 1 Big 1 Little System



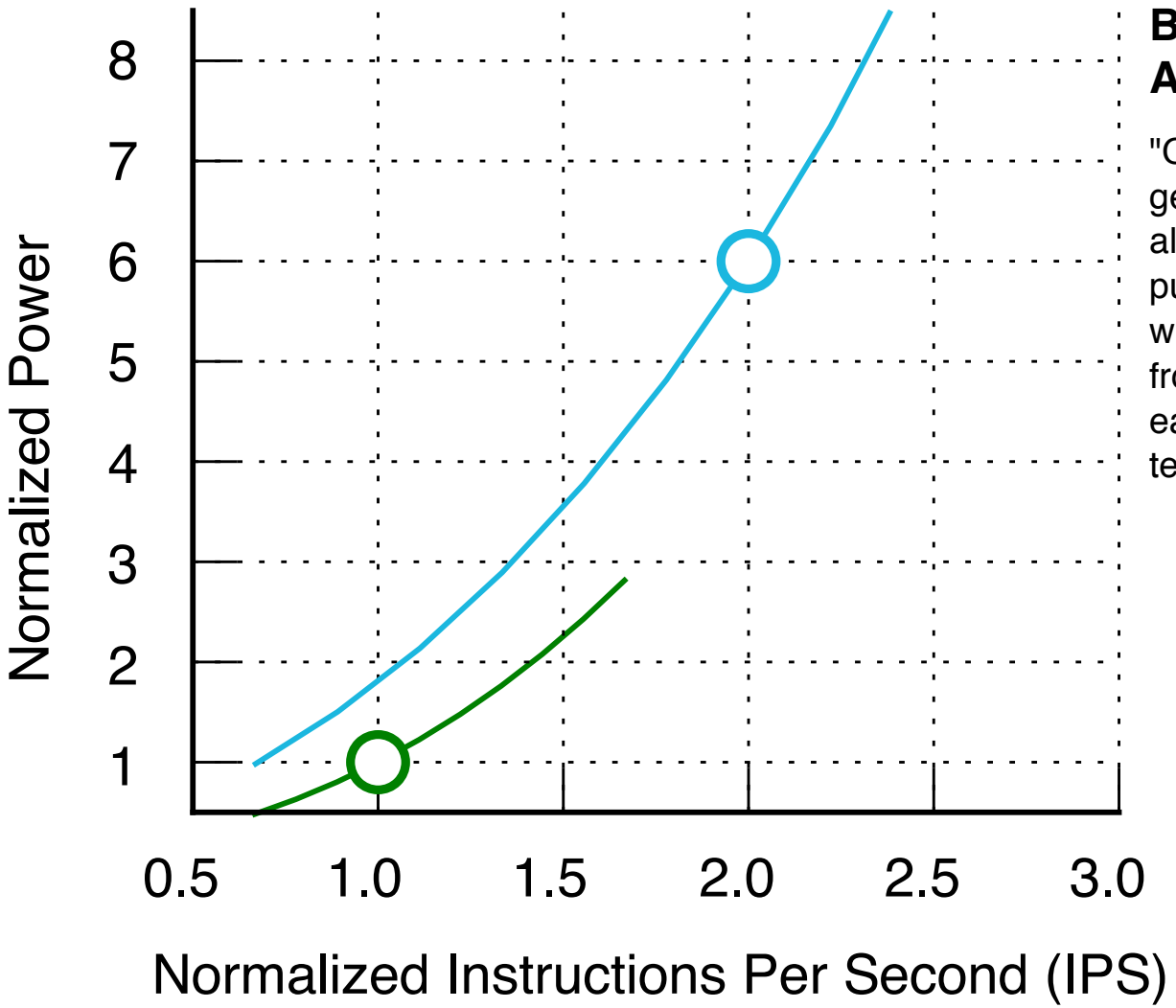


Building Intuition by Exploring a 1 Big 1 Little System





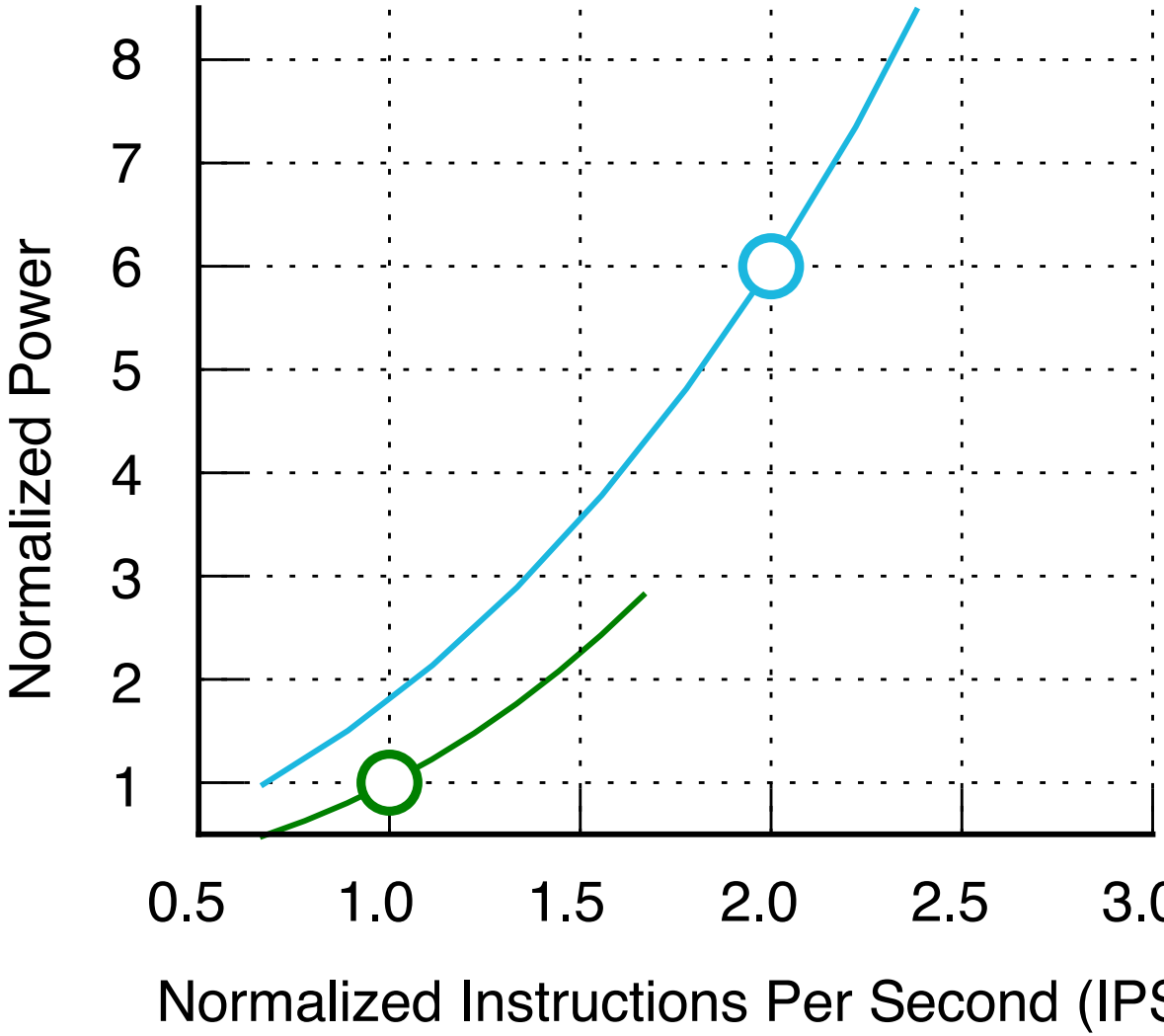
The Law of Equi-Marginal Utility



British Economist
Alfred Marshall (1824 - 1924)

"Other things being equal, a consumer gets **maximum satisfaction** when he allocates his **limited income** to the purchase of different goods in such a way that the **Marginal Utility** derived from the last unit of money spent on each item of expenditure tend **to be equal**."

The Law of Equi-Marginal Utility

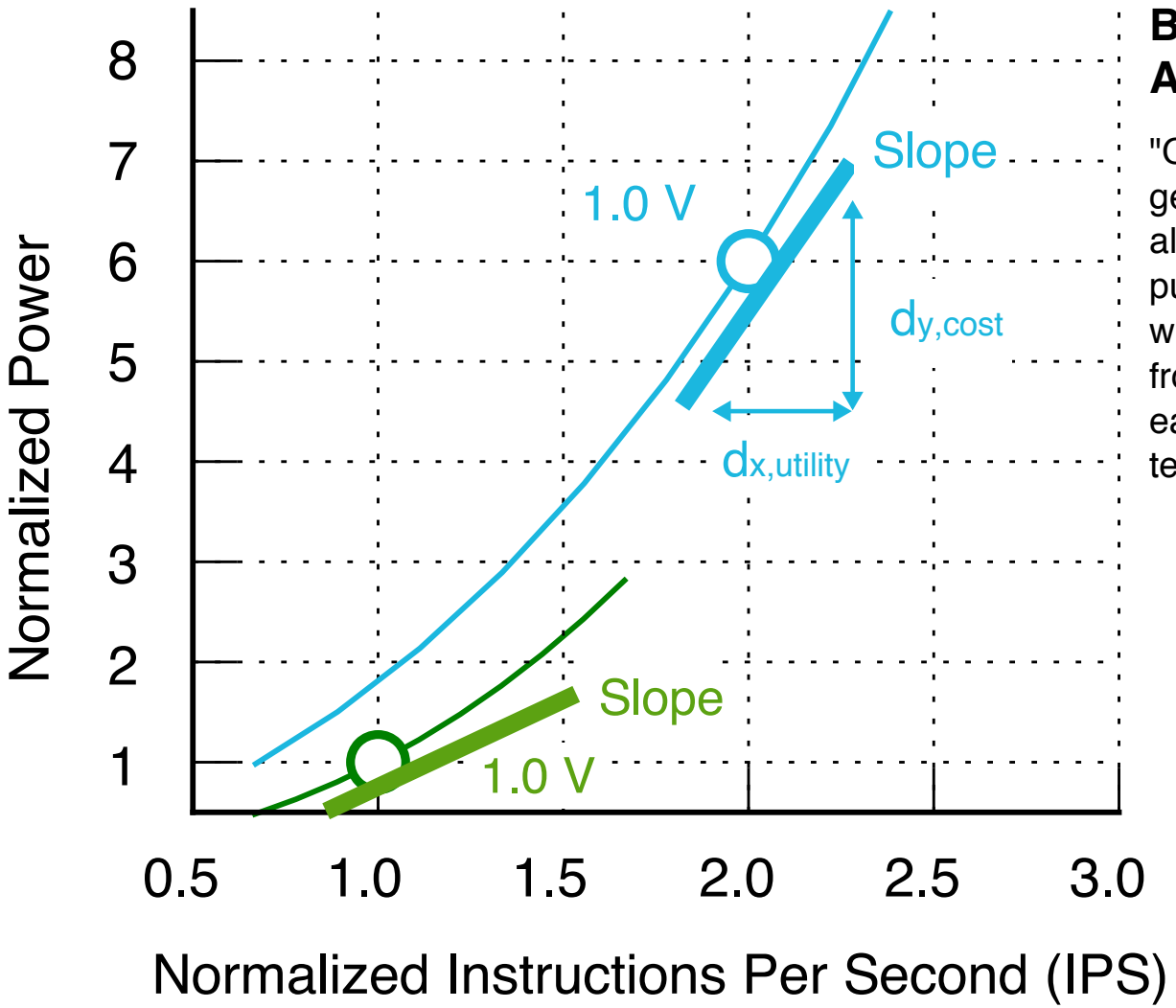


British Economist
Alfred Marshall (1824 - 1924)

"Other things being equal, a consumer gets **maximum satisfaction** when he allocates his **limited income** to the purchase of different goods in such a way that the **Marginal Utility** derived from the last unit of money spent on each item of expenditure tend **to be equal**."

Balance the ratio of utility (IPS) to cost (power)

The Law of Equi-Marginal Utility

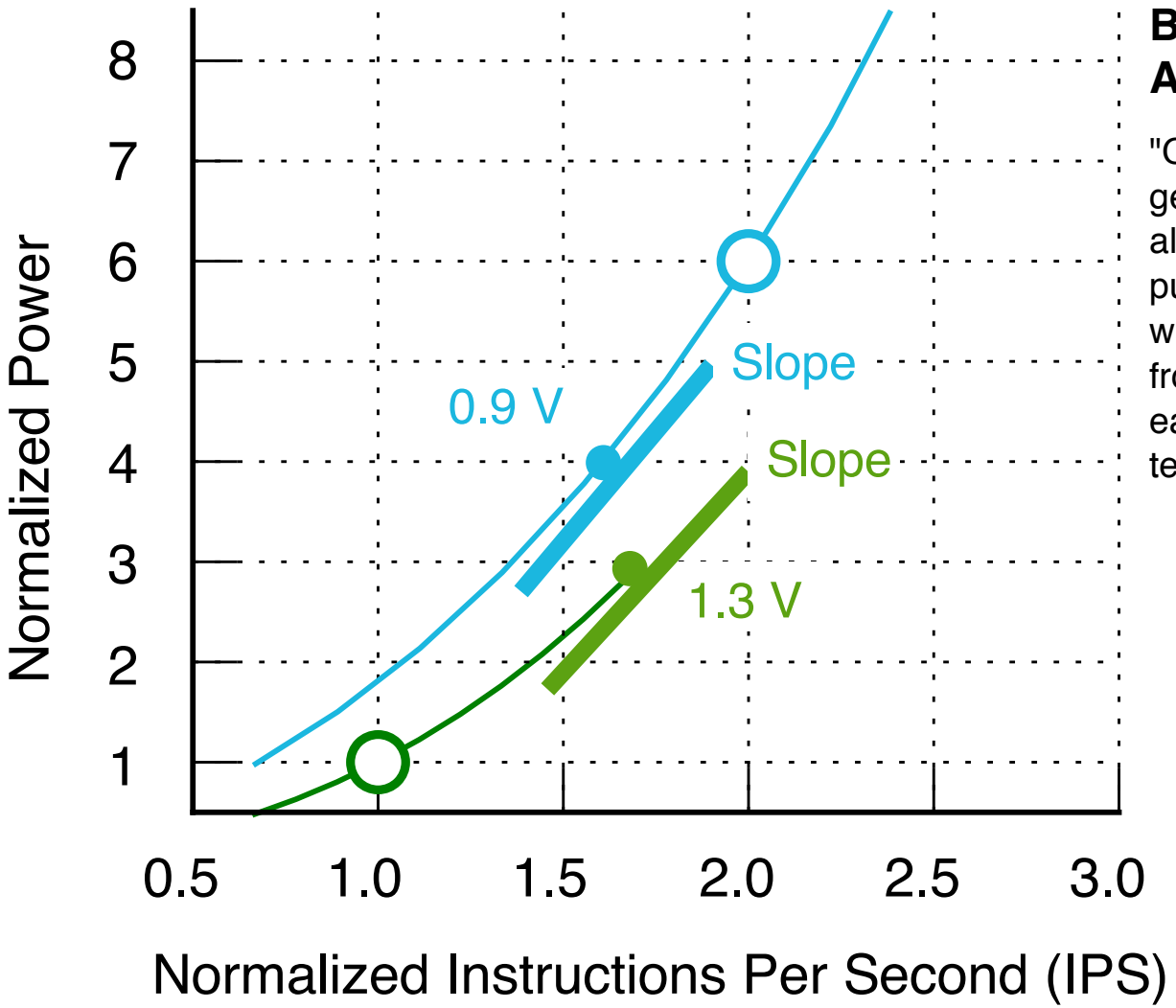


British Economist
Alfred Marshall (1824 - 1924)

"Other things being equal, a consumer gets **maximum satisfaction** when he allocates his **limited income** to the purchase of different goods in such a way that the **Marginal Utility** derived from the last unit of money spent on each item of expenditure tend **to be equal**."

Balance the ratio of utility (IPS) to cost (power)

The Law of Equi-Marginal Utility

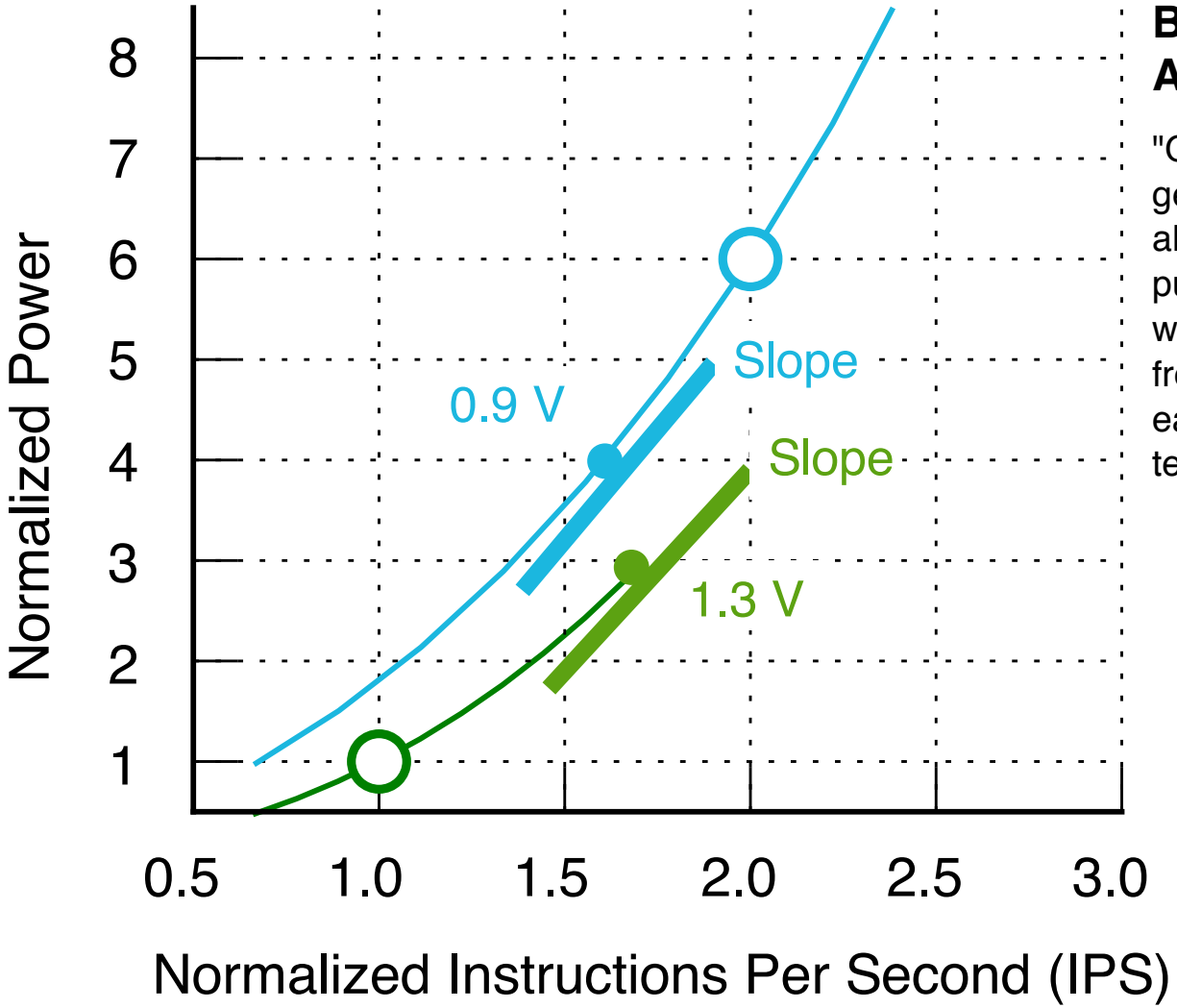


British Economist
Alfred Marshall (1824 - 1924)

"Other things being equal, a consumer gets **maximum satisfaction** when he allocates his **limited income** to the purchase of different goods in such a way that the **Marginal Utility** derived from the last unit of money spent on each item of expenditure tend **to be equal**."

Balance the ratio of utility (IPS) to cost (power)

The Law of Equi-Marginal Utility



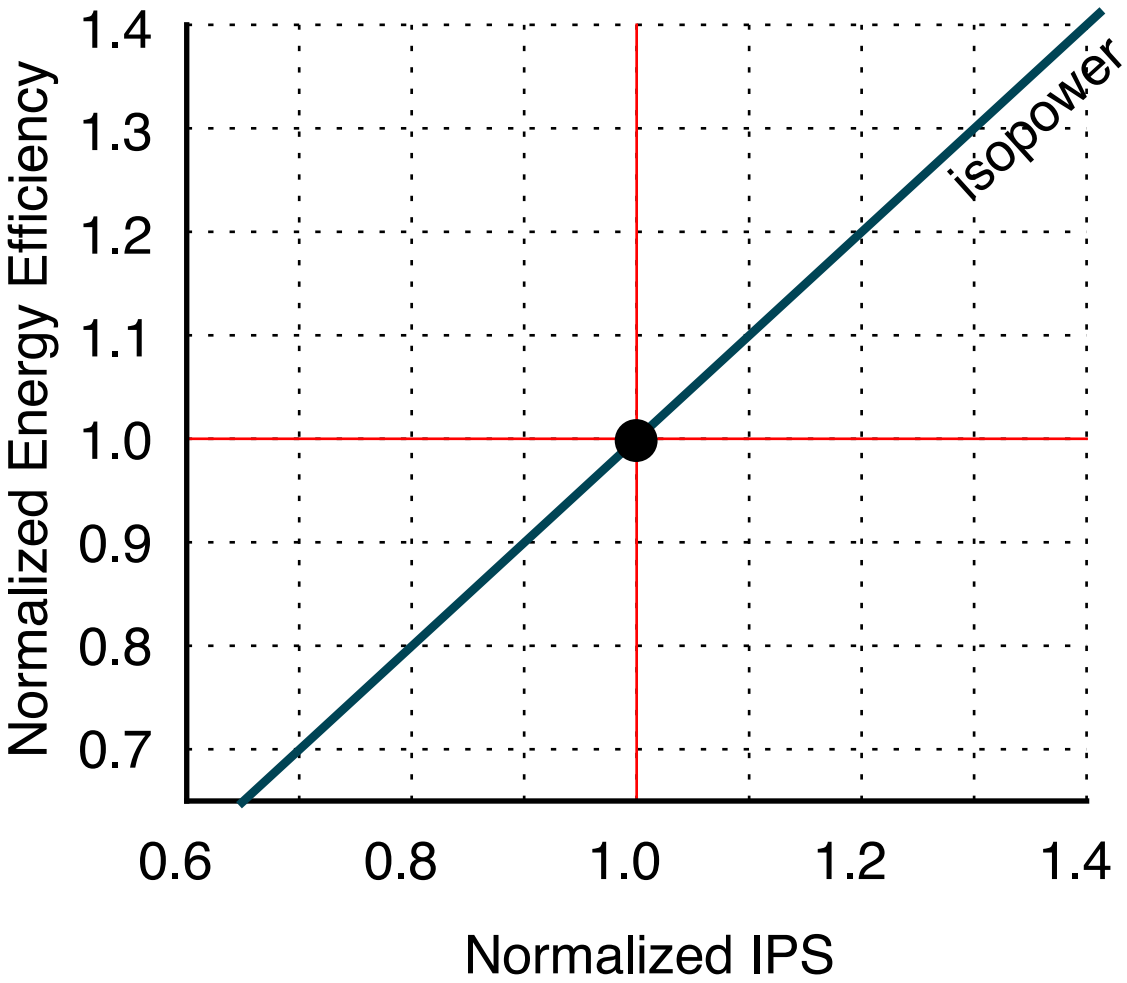
British Economist
Alfred Marshall (1824 - 1924)

"Other things being equal, a consumer gets **maximum satisfaction** when he allocates his **limited income** to the purchase of different goods in such a way that the **Marginal Utility** derived from the last unit of money spent on each item of expenditure tend **to be equal**."

Balance the ratio of utility (IPS) to cost (power)

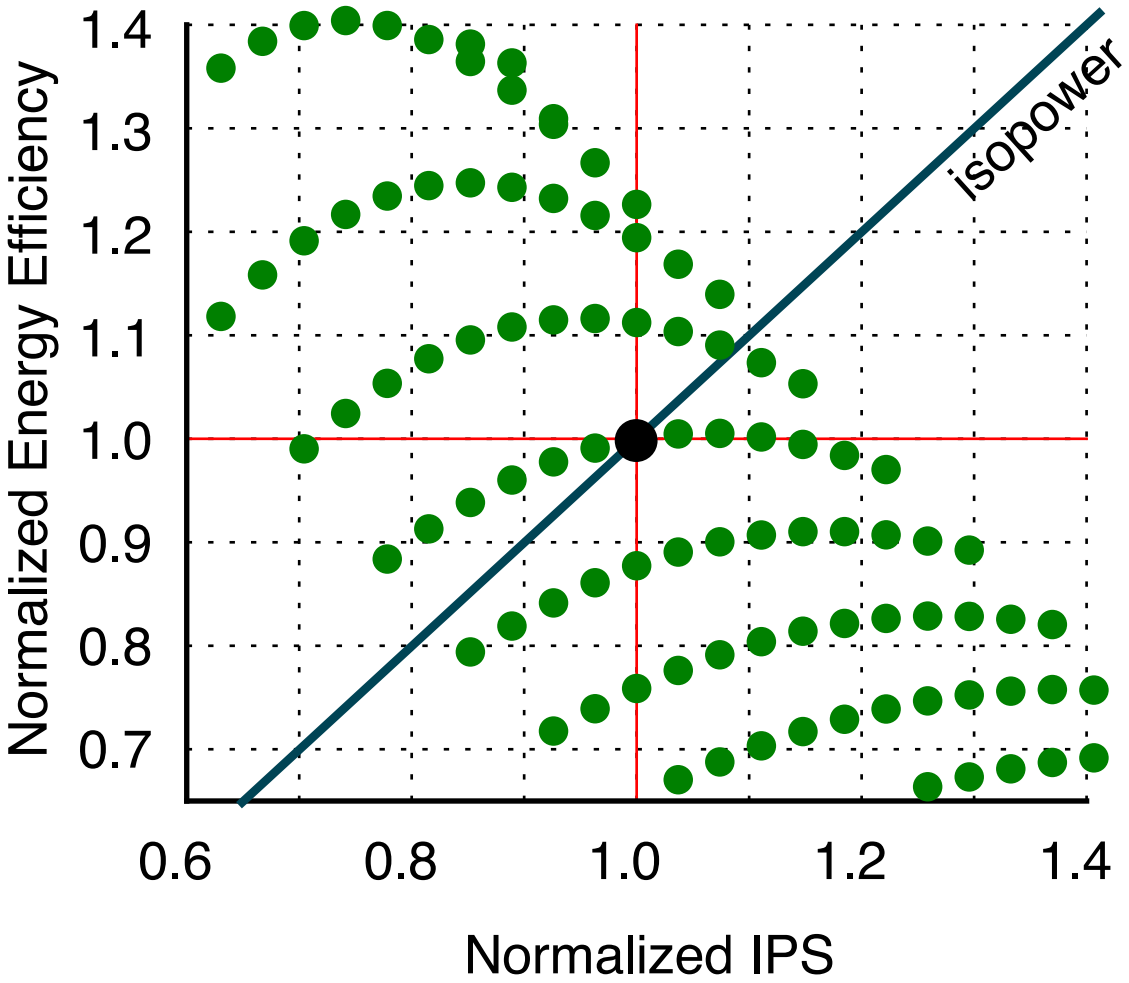
Arbitrage
"Buy Low, Sell High"

Systematic Approach for Balancing Marginal Utility



- 1 Big 1 Little System at Nominal voltage

Systematic Approach for Balancing Marginal Utility

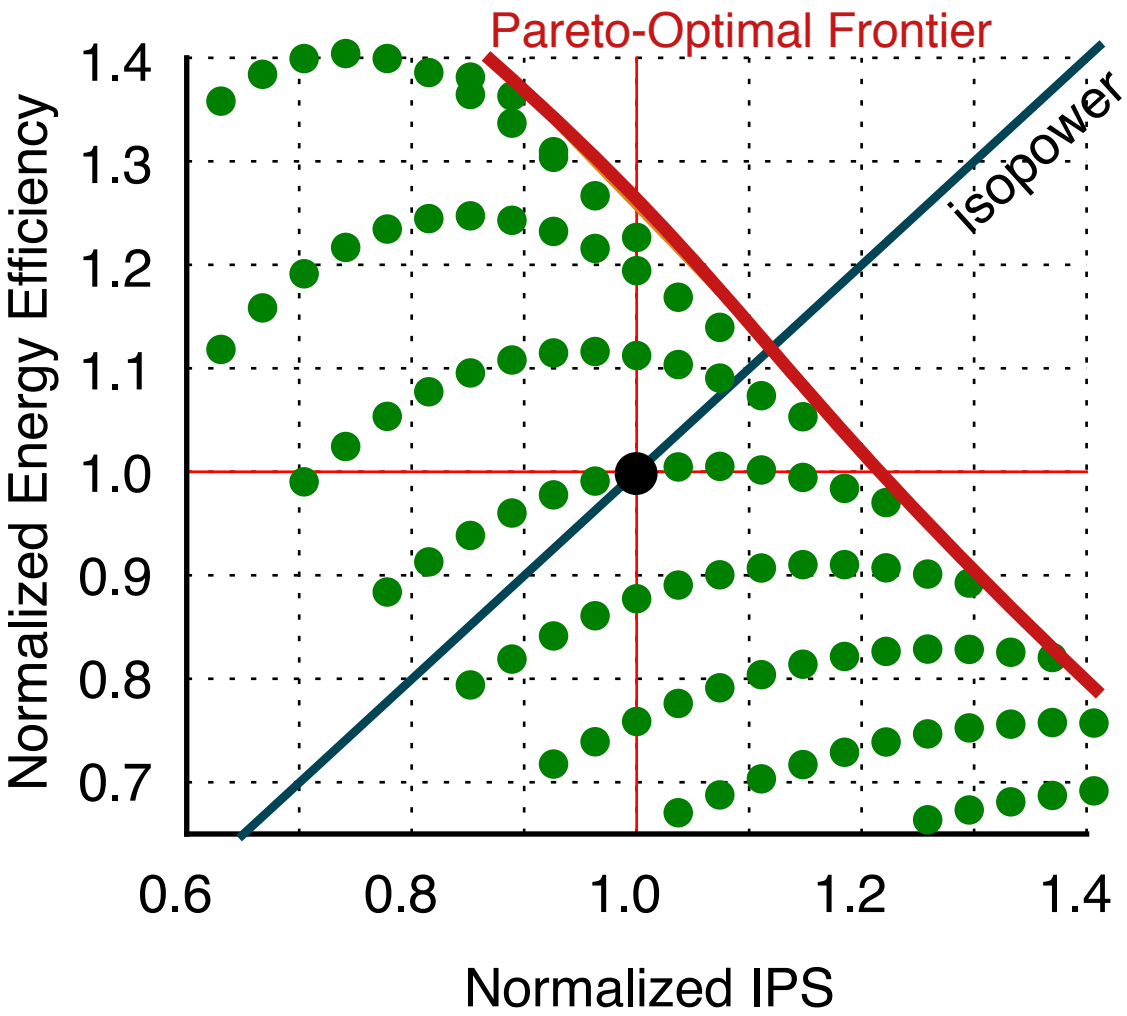


- 1 Big 1 Little System at Nominal voltage
- Individual (V_B, V_L) pair

Assumptions

Perfectly parallel application
Ideal load balancing

Systematic Approach for Balancing Marginal Utility

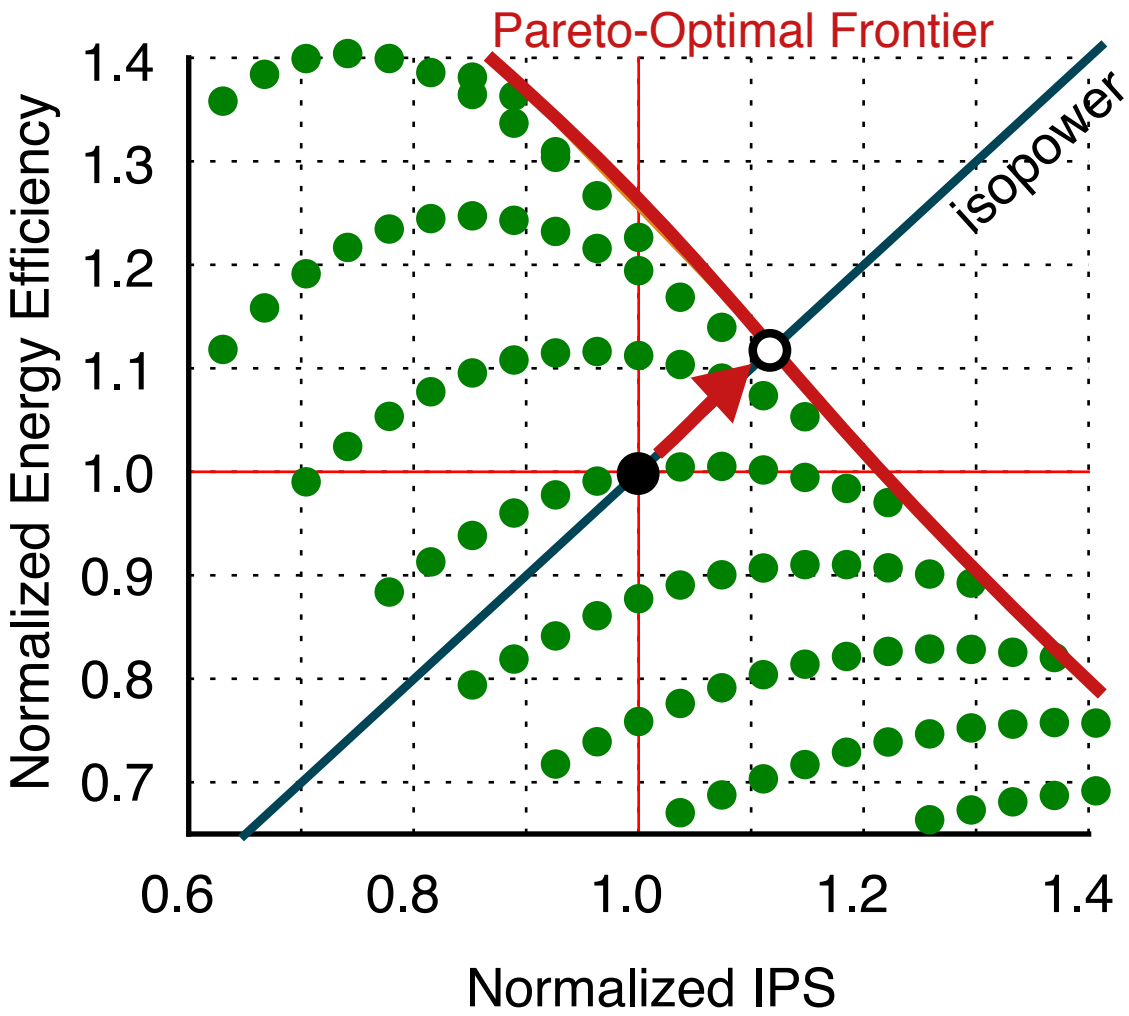


- 1 Big 1 Little System at Nominal voltage
- Individual (V_B, V_L) pair

Assumptions

Perfectly parallel application
Ideal load balancing

Systematic Approach for Balancing Marginal Utility



- 1 Big 1 Little System at Nominal voltage
- Individual (V_B, V_L) pair

Assumptions

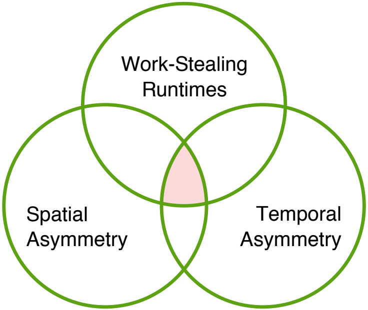
Perfectly parallel application
Ideal load balancing

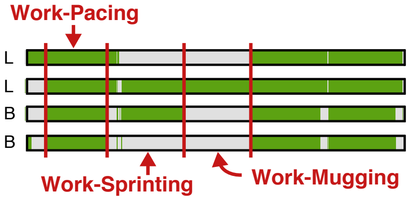
Marginal Utility-Based Optimization Problem

Constraint: *isopower line*

Objective: *maximize performance*

(Solved numerically)



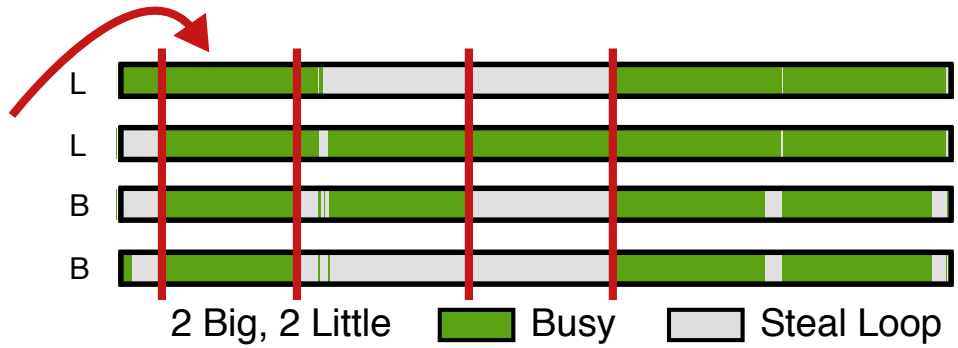


Exploiting Fine-Grain Asymmetry for Task-Based Parallelism

- Intuition
- **Work-Pacing**
- Results

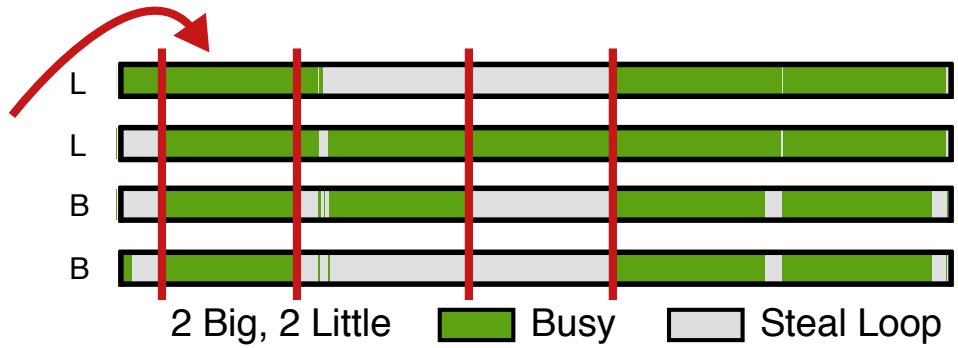
Work-Pacing: Building Intuition

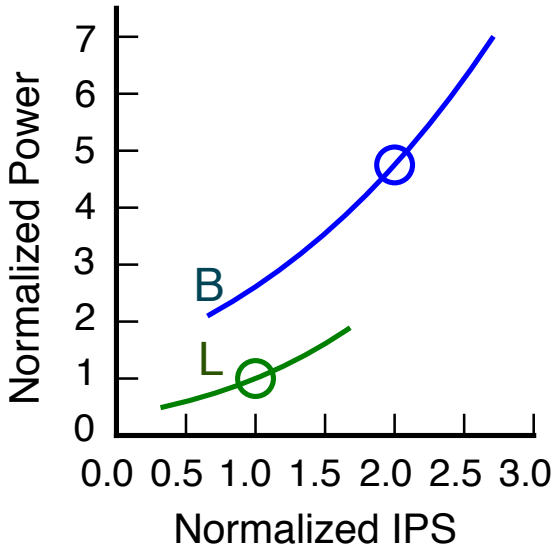
Balance performance/power across cores in the high-parallel (HP) region



Work-Pacing: Building Intuition

Balance performance/power across cores in the high-parallel (HP) region

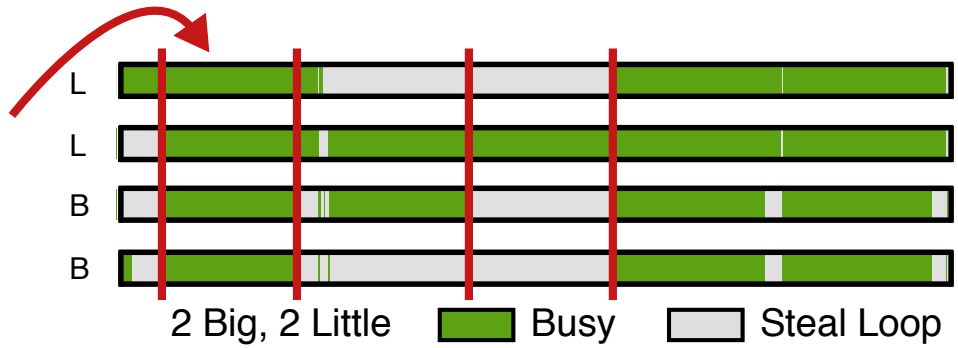


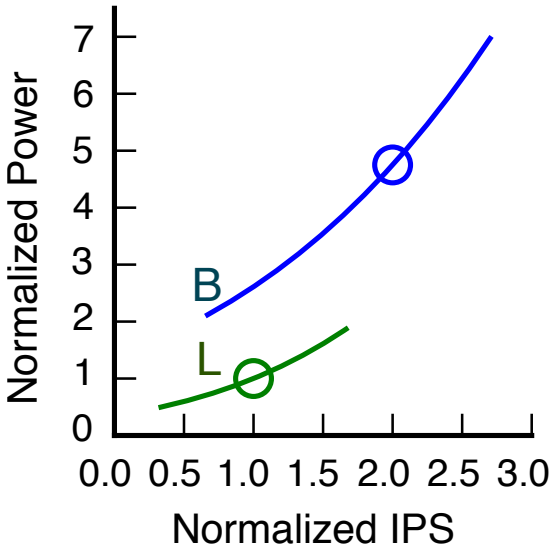


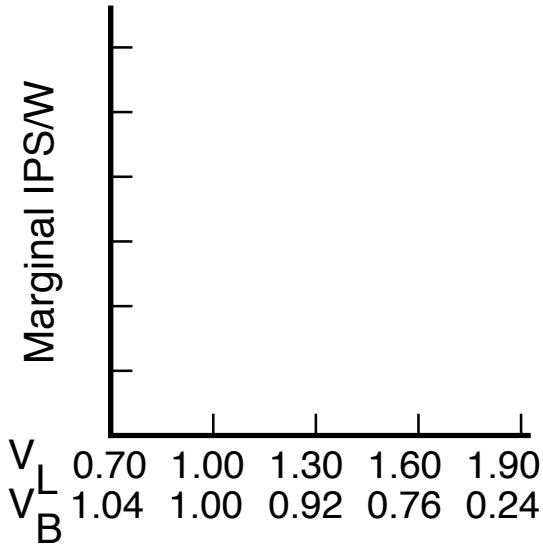
System with both big cores active and both little cores active

Work-Pacing: Building Intuition

Balance performance/power across cores in the high-parallel (HP) region



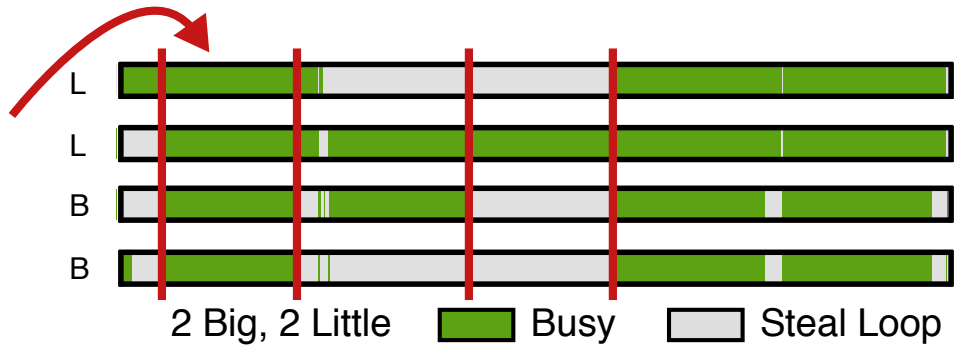


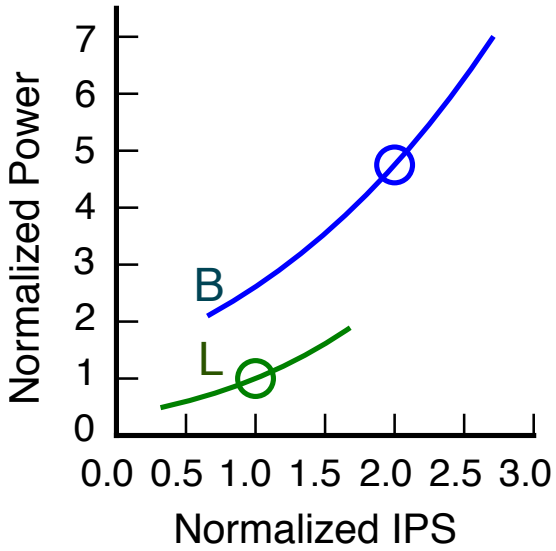


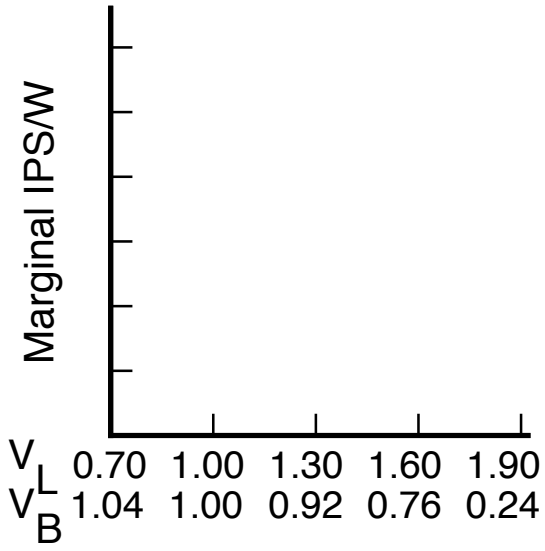
System with both big cores active and both little cores active

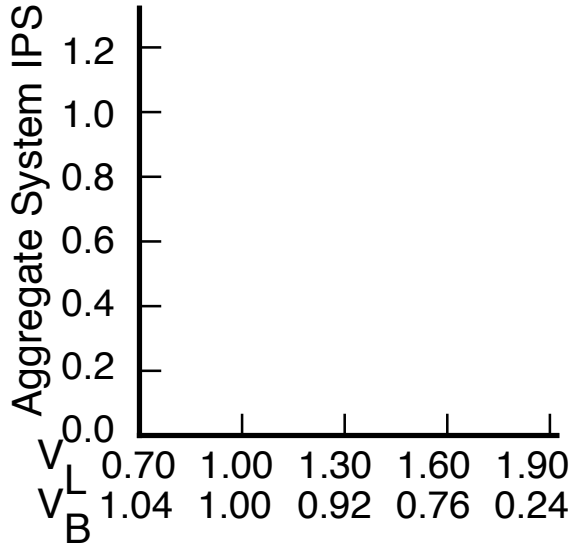
Work-Pacing: Building Intuition

Balance performance/power across cores in the high-parallel (HP) region





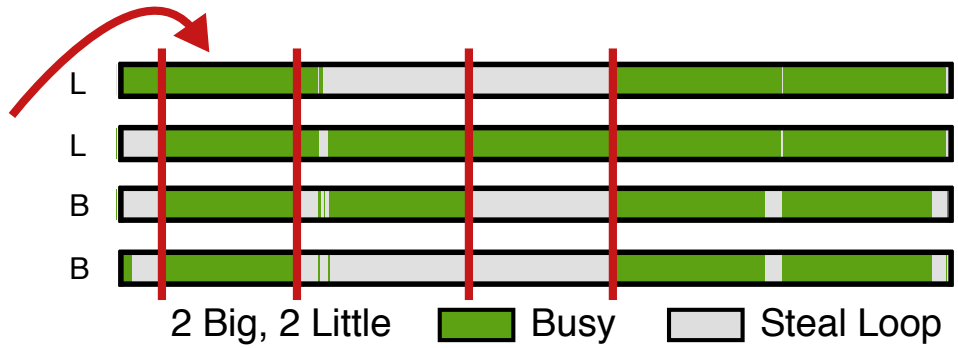


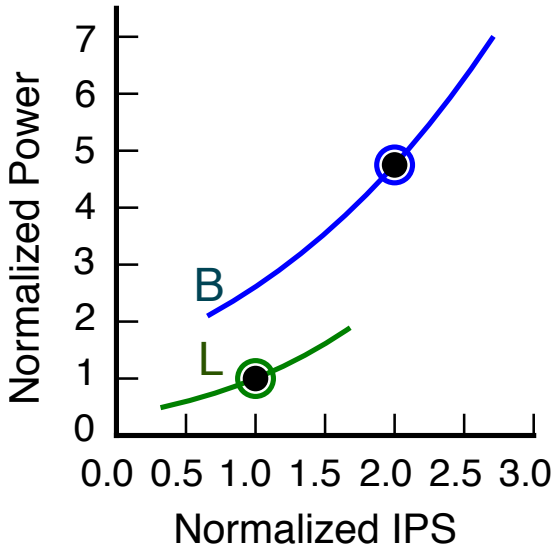


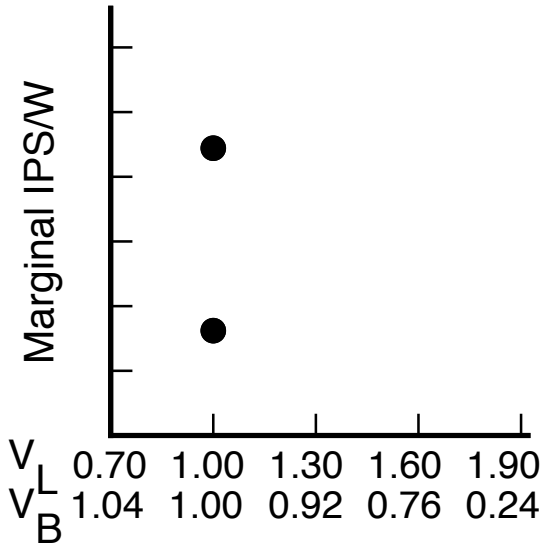
System with both big cores active and both little cores active

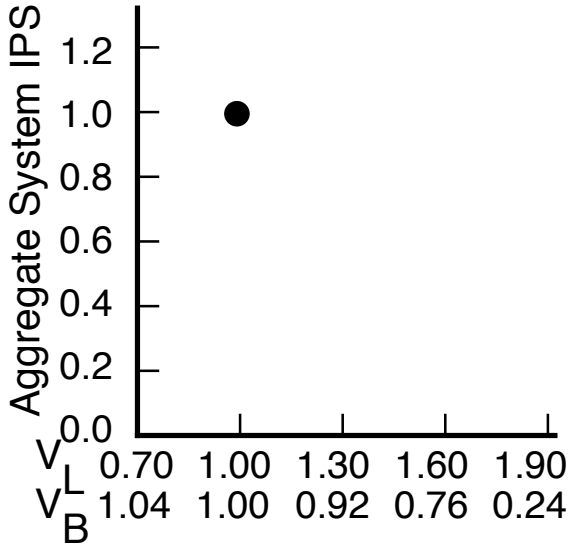
Work-Pacing: Building Intuition

Balance performance/power across cores in the high-parallel (HP) region





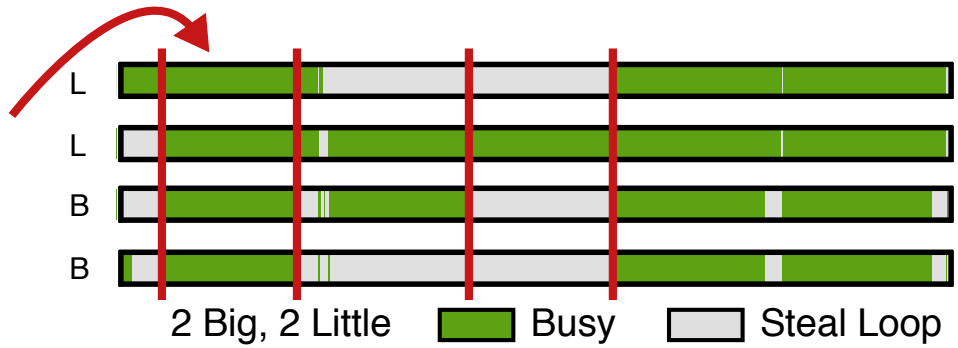


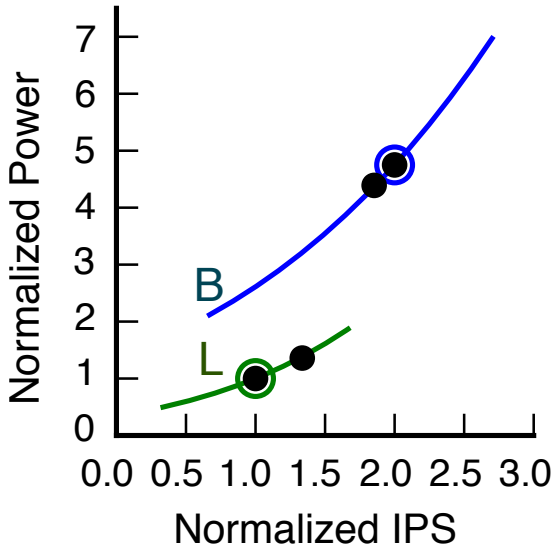


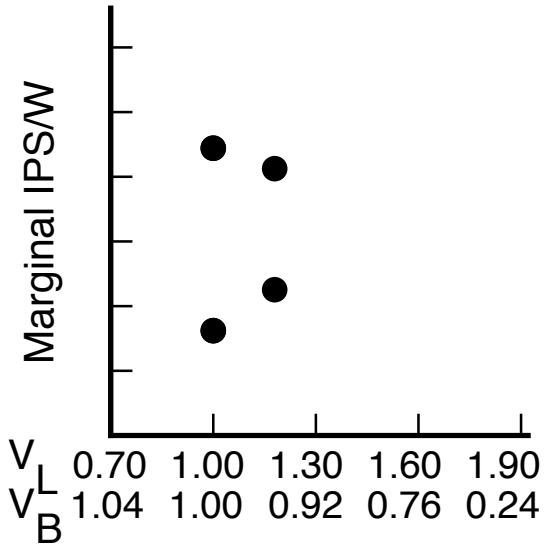
System with both big cores active and both little cores active

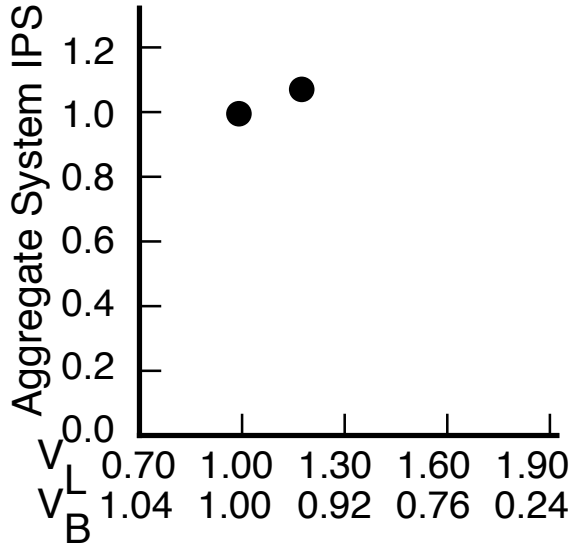
Work-Pacing: Building Intuition

Balance performance/power across cores in the high-parallel (HP) region





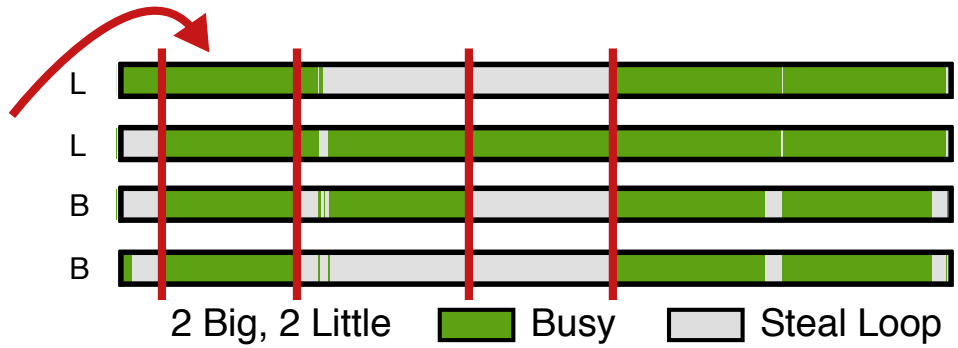


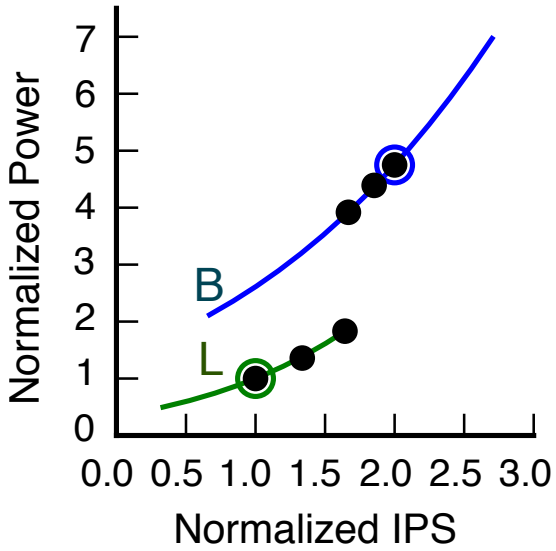


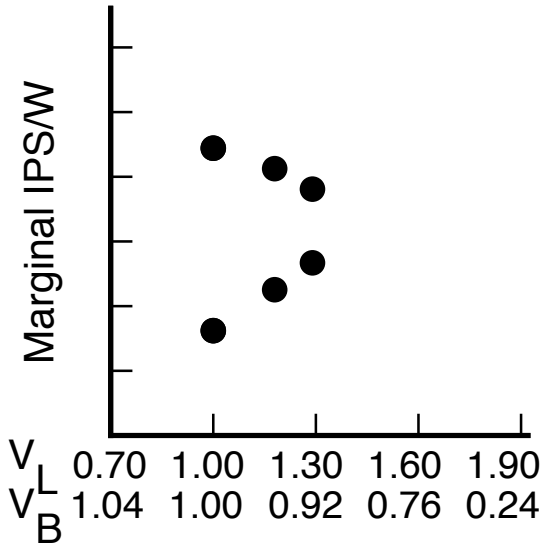
System with both big cores active and both little cores active

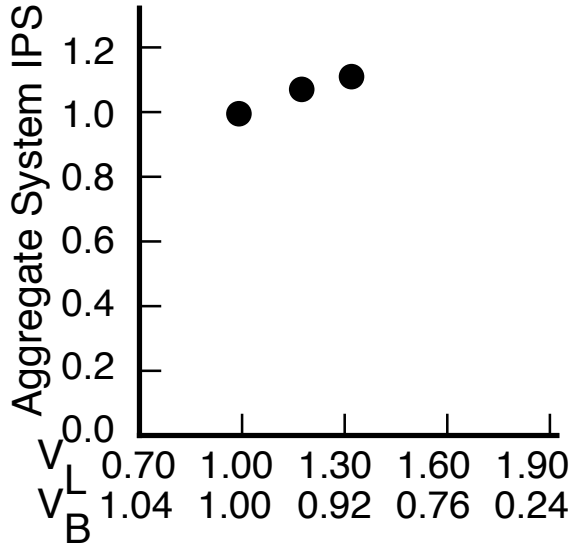
Work-Pacing: Building Intuition

Balance performance/power across cores in the high-parallel (HP) region





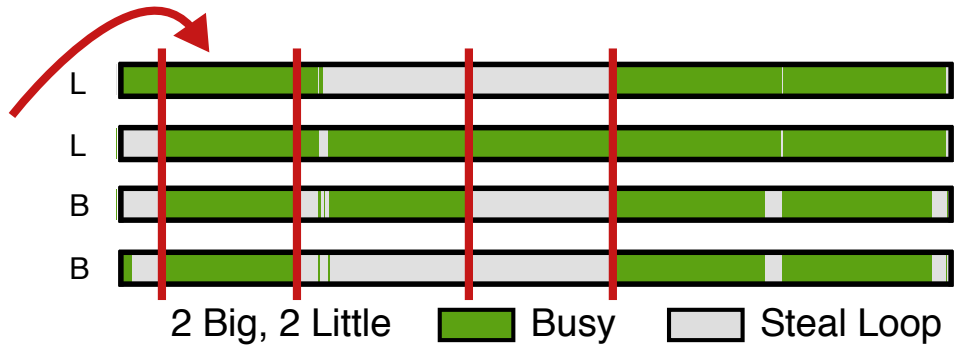


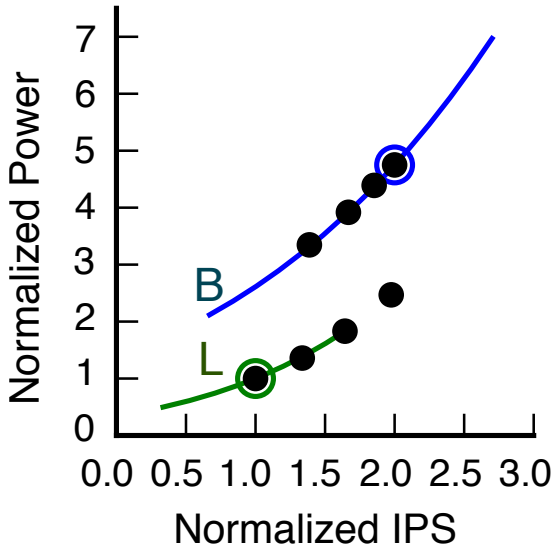


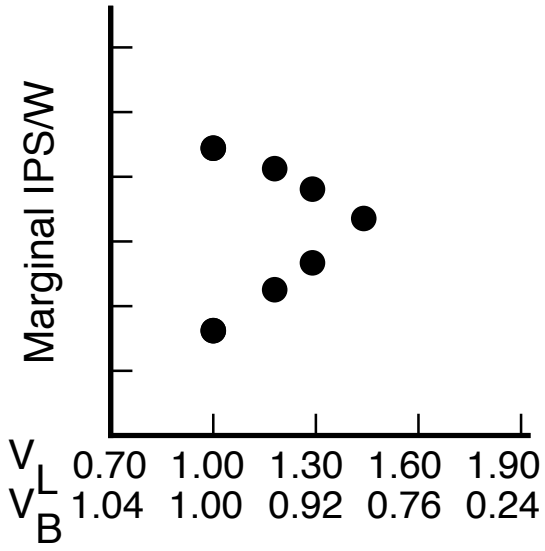
System with both big cores active and both little cores active

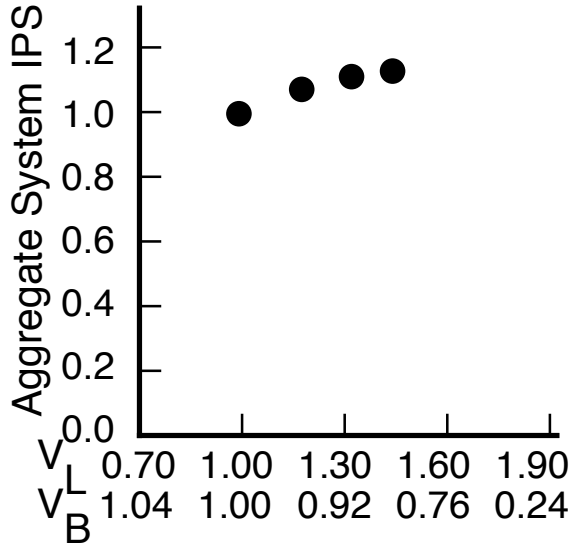
Work-Pacing: Building Intuition

Balance performance/power across cores in the high-parallel (HP) region





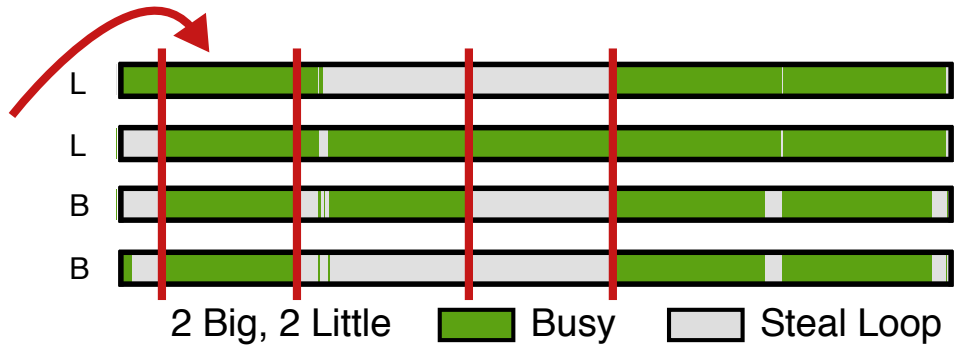


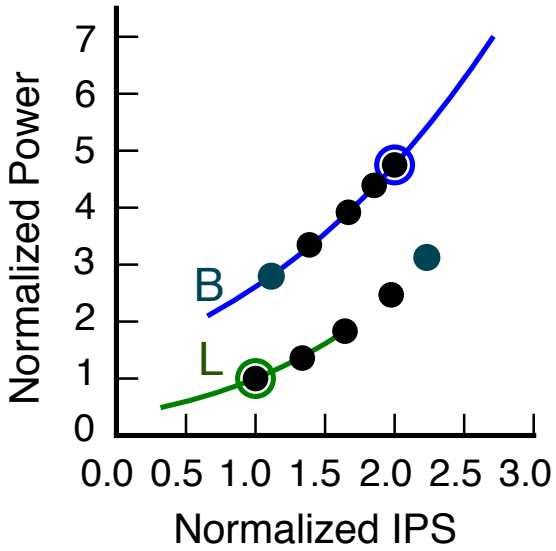


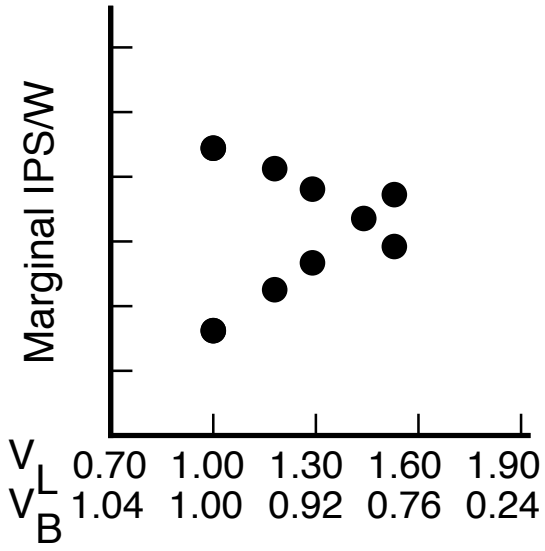
System with both big cores active and both little cores active

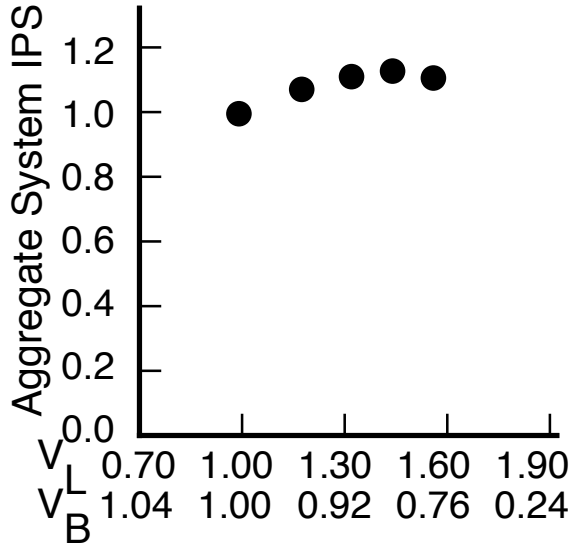
Work-Pacing: Building Intuition

Balance performance/power across cores in the high-parallel (HP) region





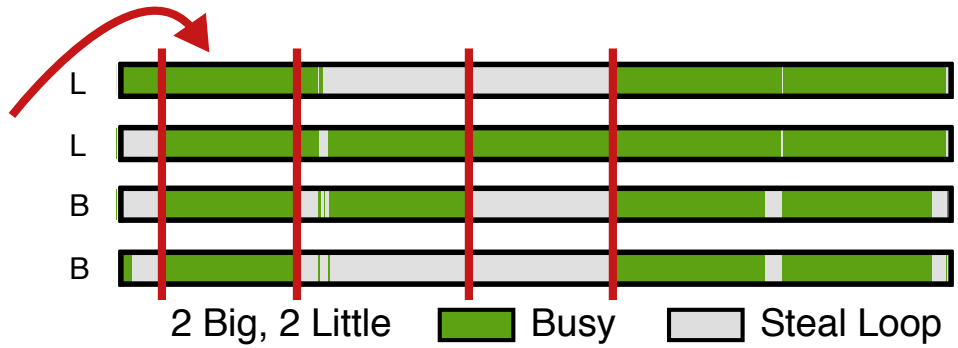


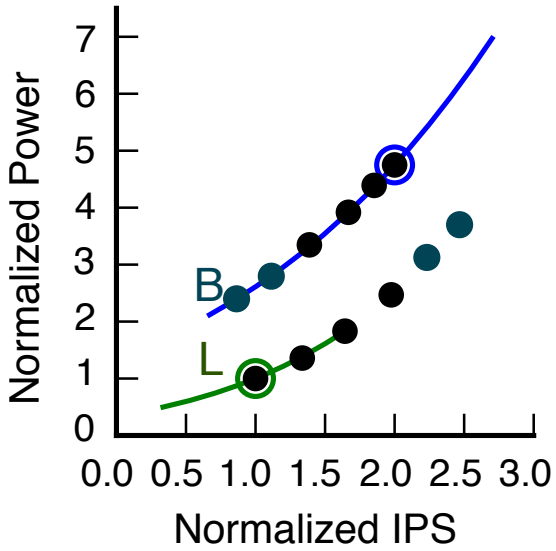


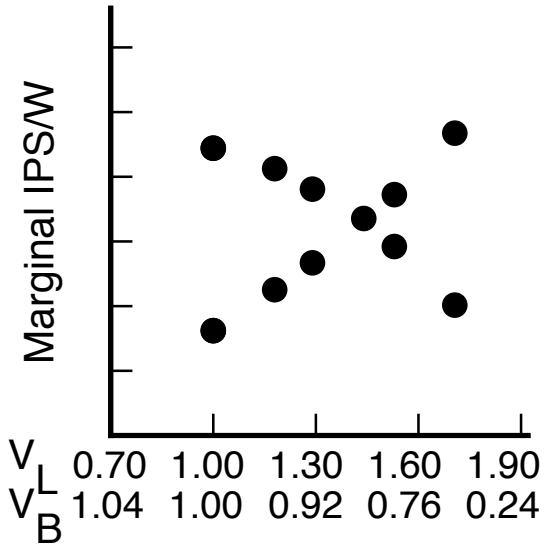
System with both big cores active and both little cores active

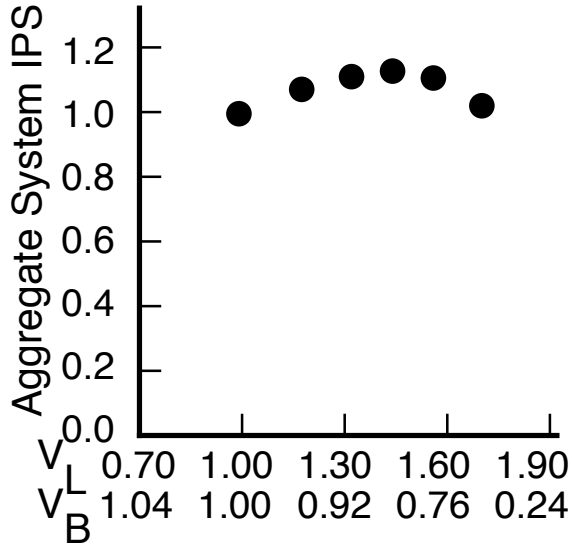
Work-Pacing: Building Intuition

Balance performance/power across cores in the high-parallel (HP) region





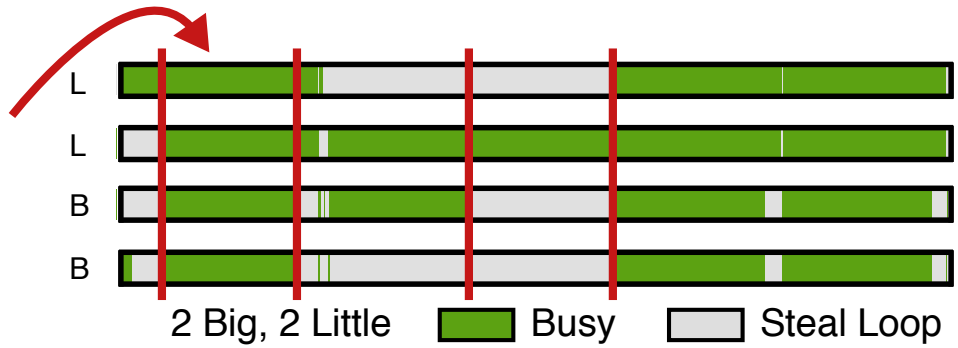


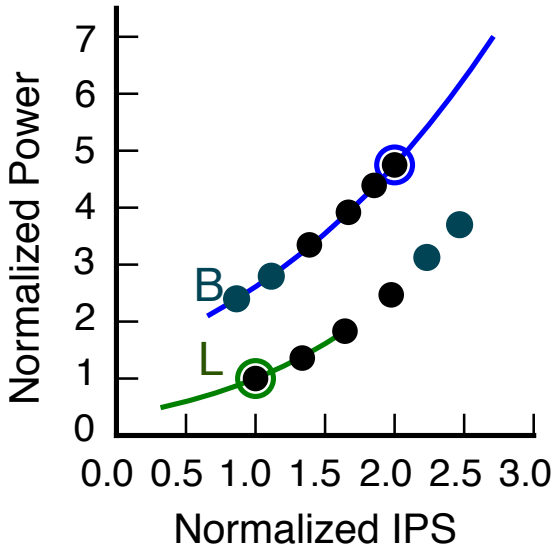


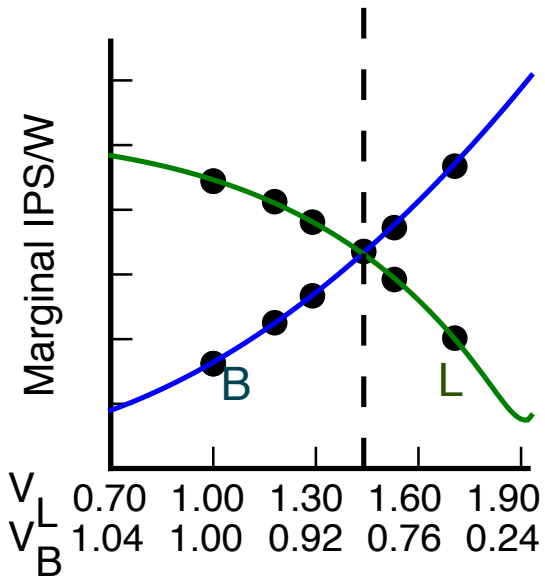
System with both big cores active and both little cores active

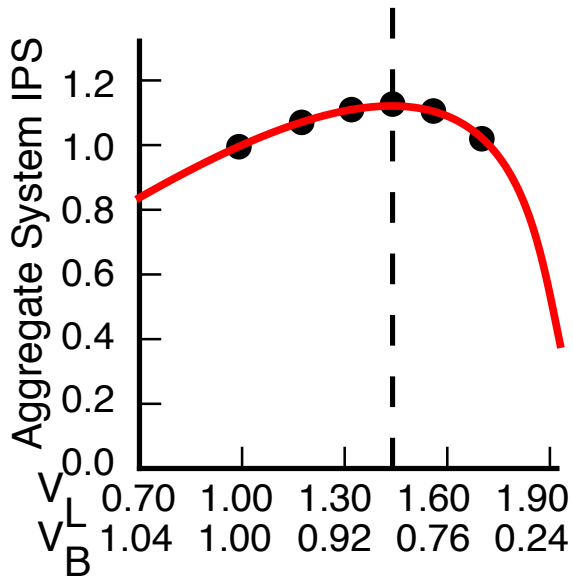
Work-Pacing: Building Intuition

Balance performance/power across cores in the high-parallel (HP) region





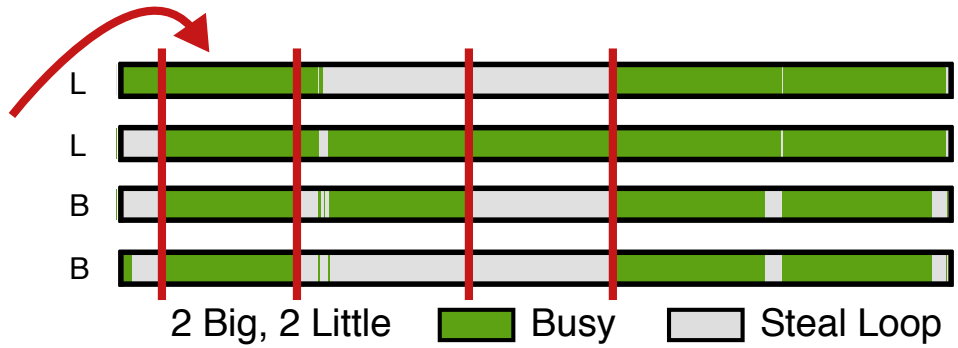


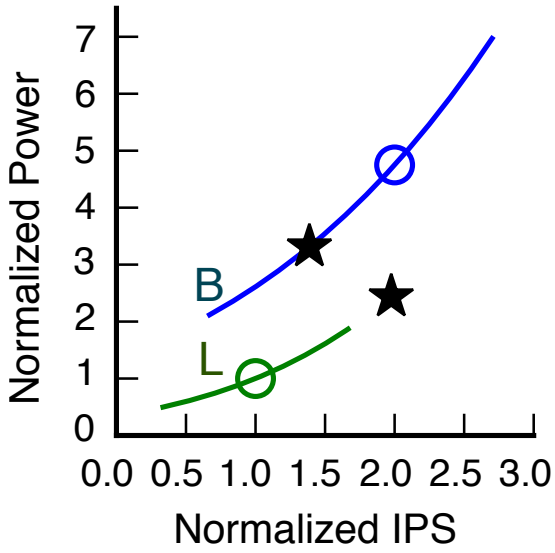


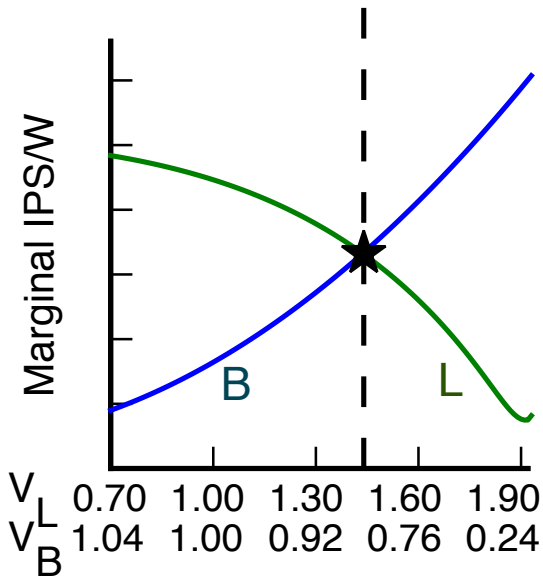
System with both big cores active and both little cores active

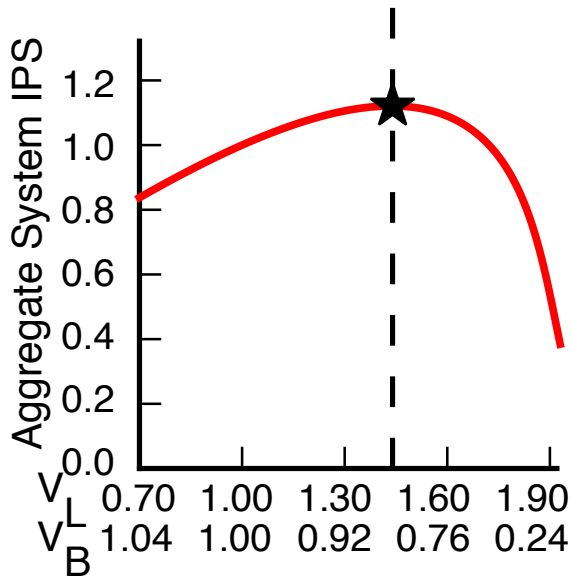
Work-Pacing: Building Intuition

Balance performance/power across cores in the high-parallel (HP) region



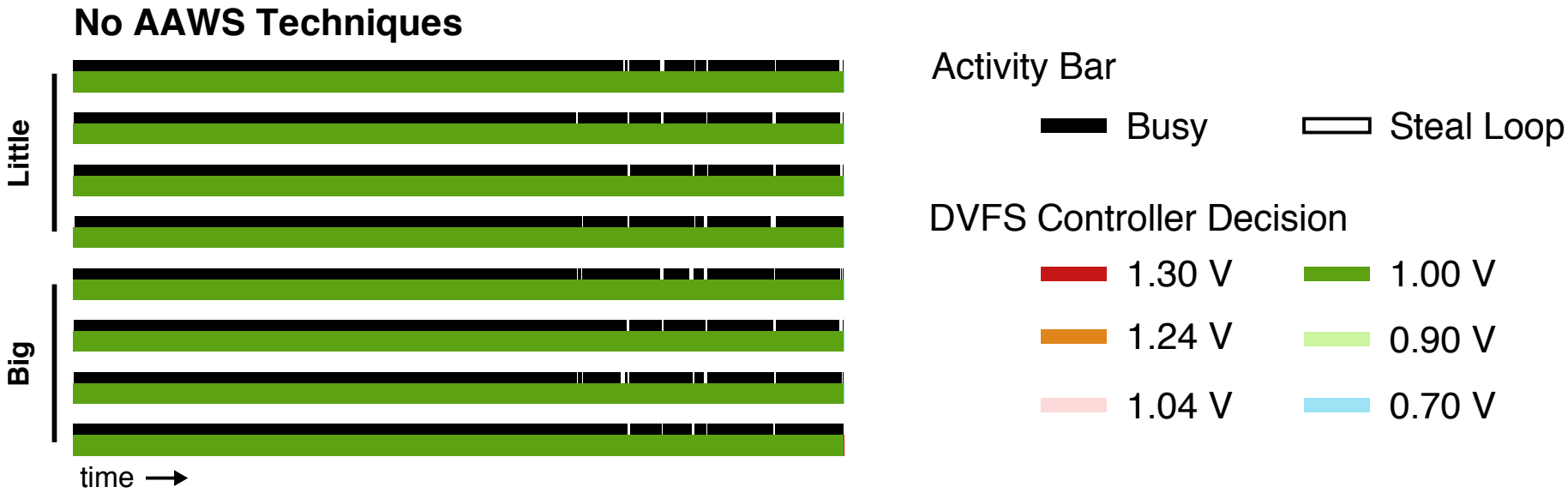




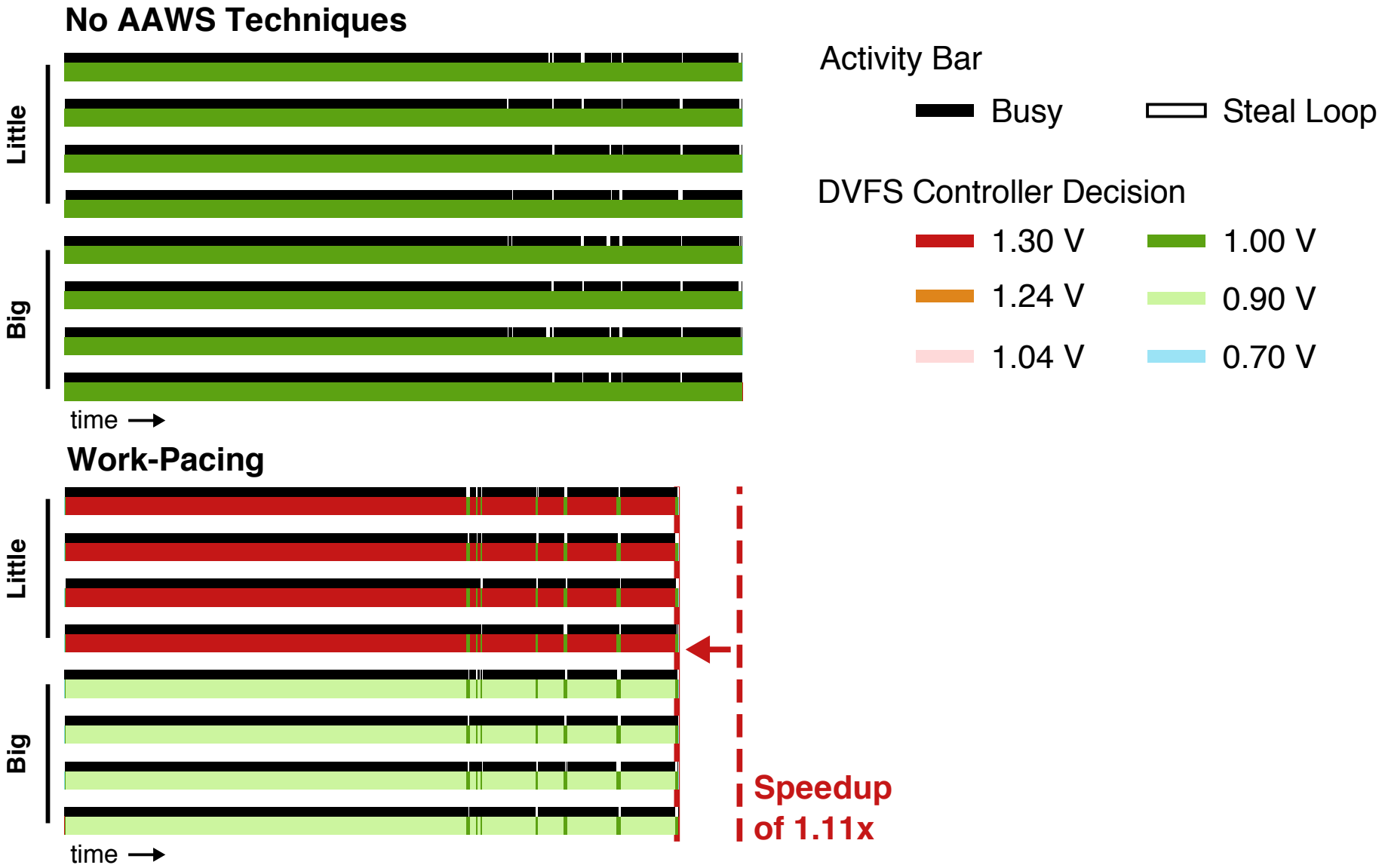


System with both big cores active and both little cores active

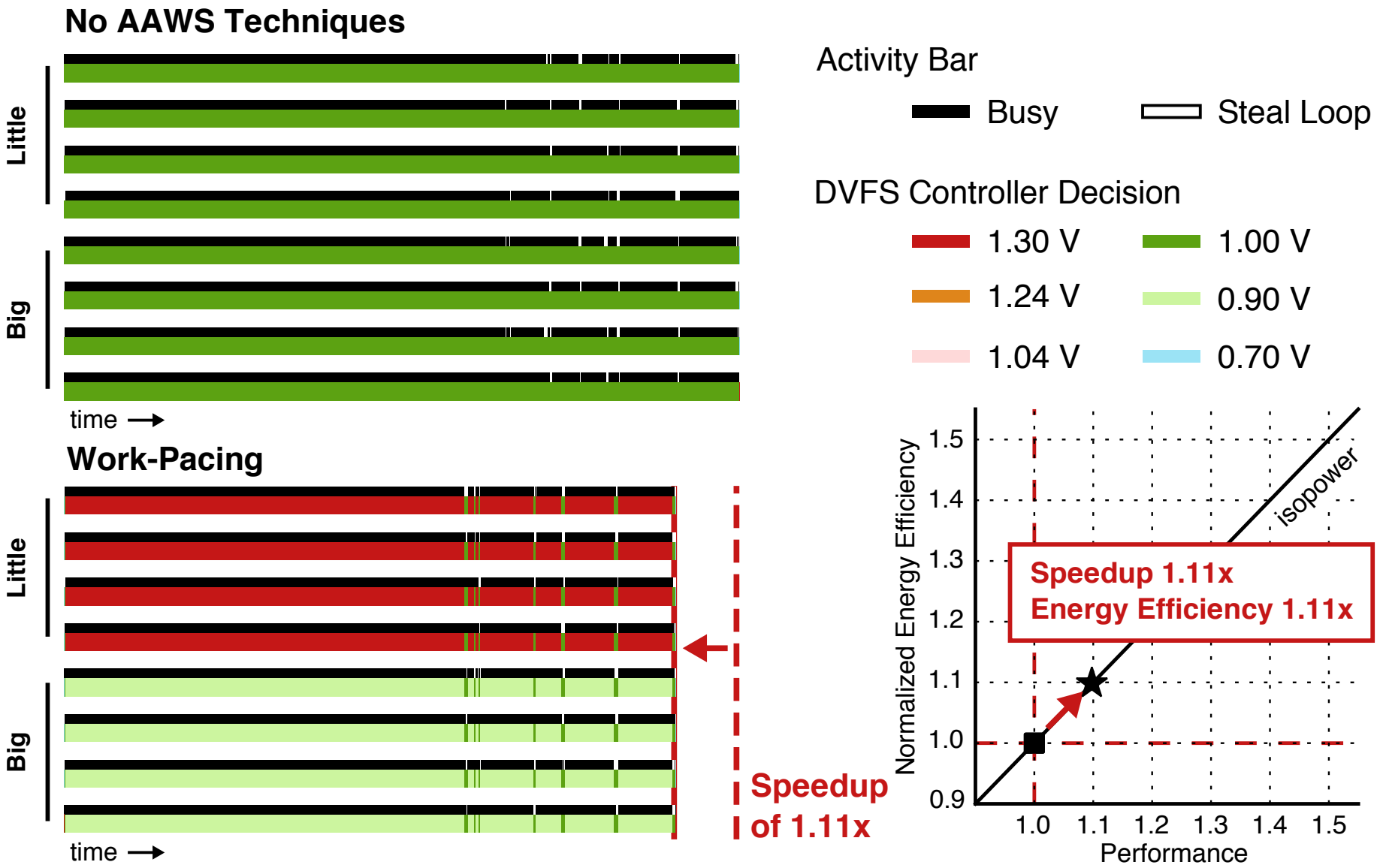
Work-Pacing in *cilk-sort*



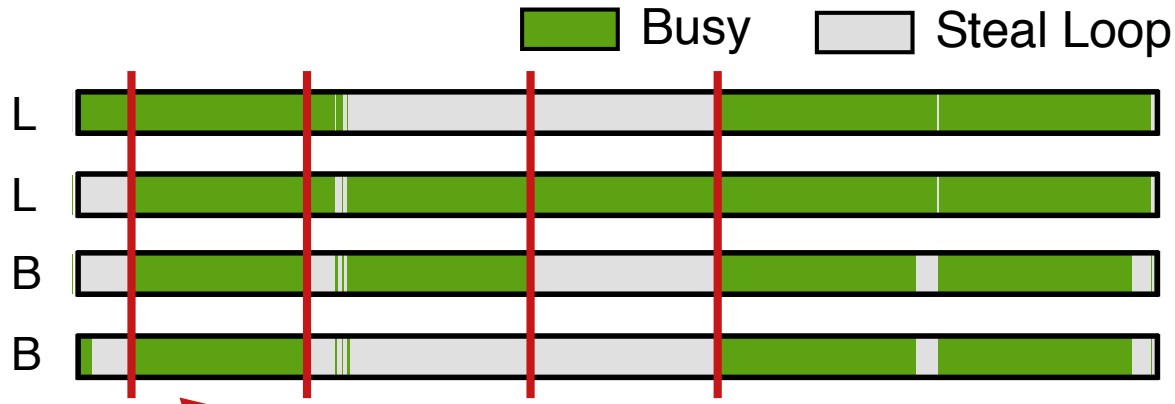
Work-Pacing in *cilk-sort*



Work-Pacing in *cilk-sort*



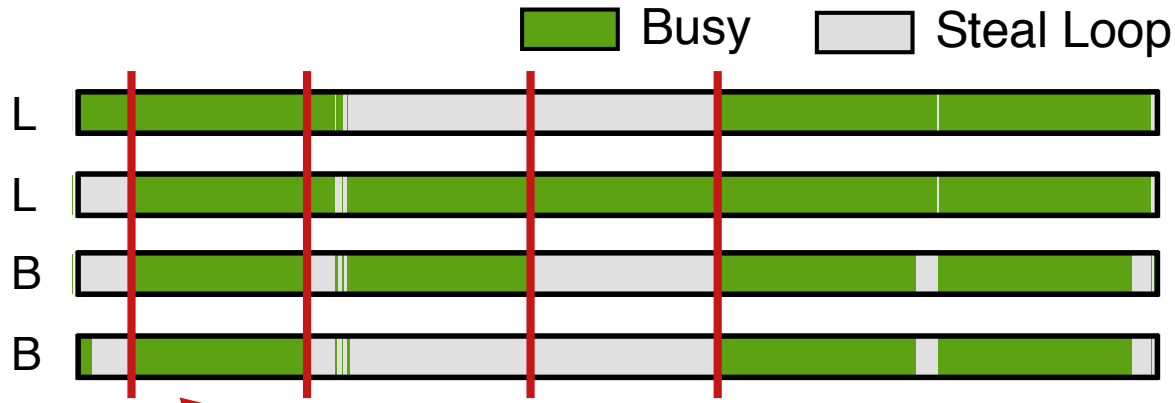
Work-Pacing, Work-Sprinting, and Work-Mugging



Work-Pacing

Balance performance/power across cores in the high-parallel (HP) region

Work-Pacing, Work-Sprinting, and Work-Mugging



Work-Pacing

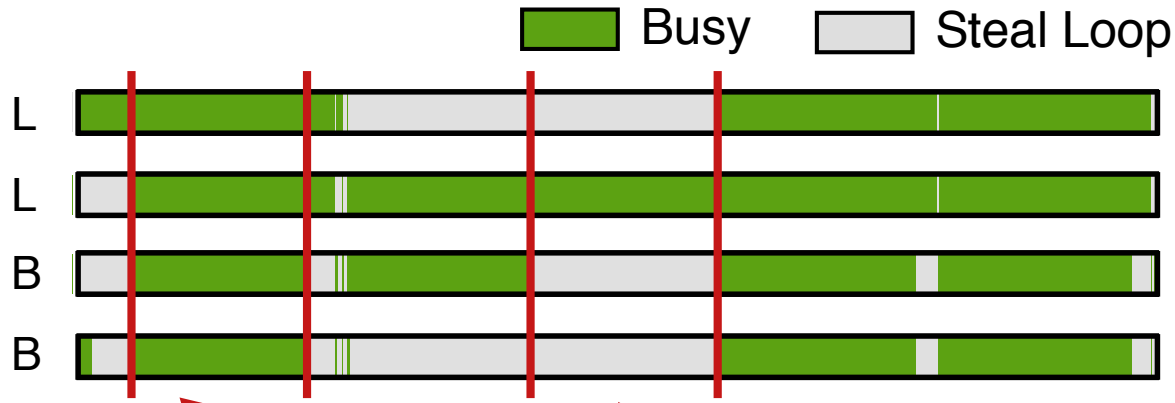
Balance performance/power across cores in the high-parallel (HP) region

Work-Sprinting

Rest cores in the steal loop to the lowest voltage

With additional power slack, balance performance/power across busy cores in the low-parallel (LP) region

Work-Pacing, Work-Sprinting, and Work-Mugging



Work-Pacing

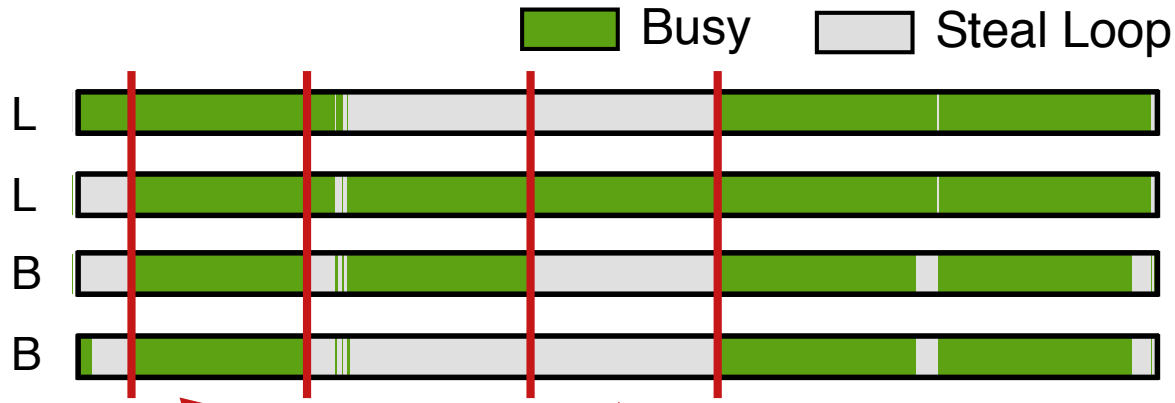
Balance performance/power across cores in the high-parallel (HP) region

Work-Sprinting

Rest cores in the steal loop to the lowest voltage

With additional power slack, balance performance/power across busy cores in the low-parallel (LP) region

Work-Pacing, Work-Sprinting, and Work-Mugging



Work-Pacing

Balance performance/power across cores in the high-parallel (HP) region

Work-Sprinting

Rest cores in the steal loop to the lowest voltage

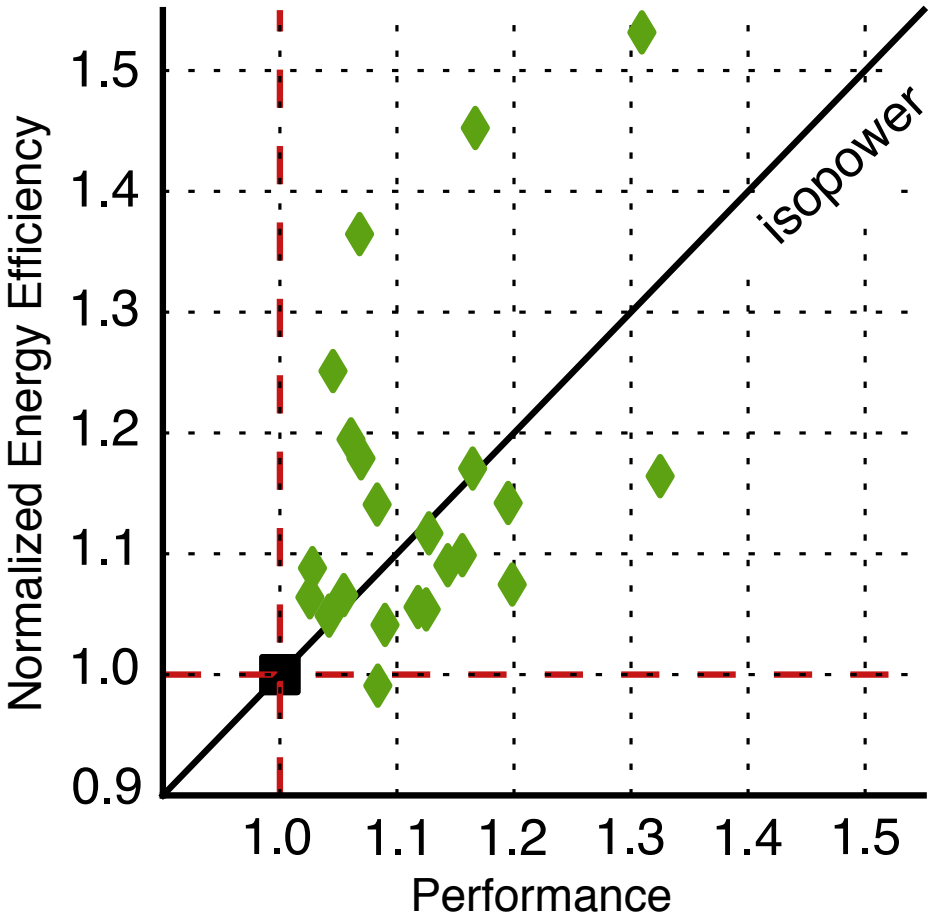
With additional power slack, balance performance/power across busy cores in the low-parallel (LP) region

Work-Mugging

Move work from slow little cores to fast big cores in the low-parallel (LP) region

Inspired by theoretical work - Bender et al. Theory of Computing '02

Evaluation of Complete AAWS Runtime

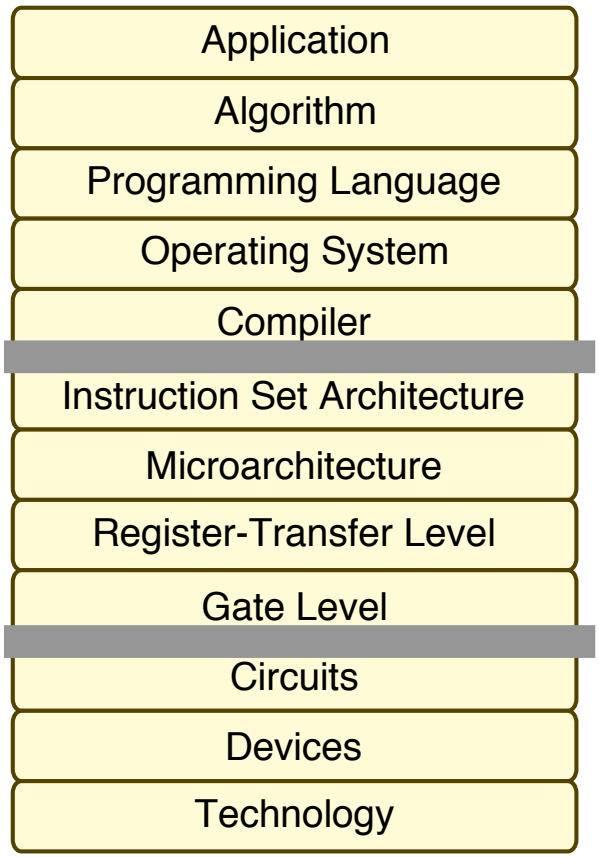


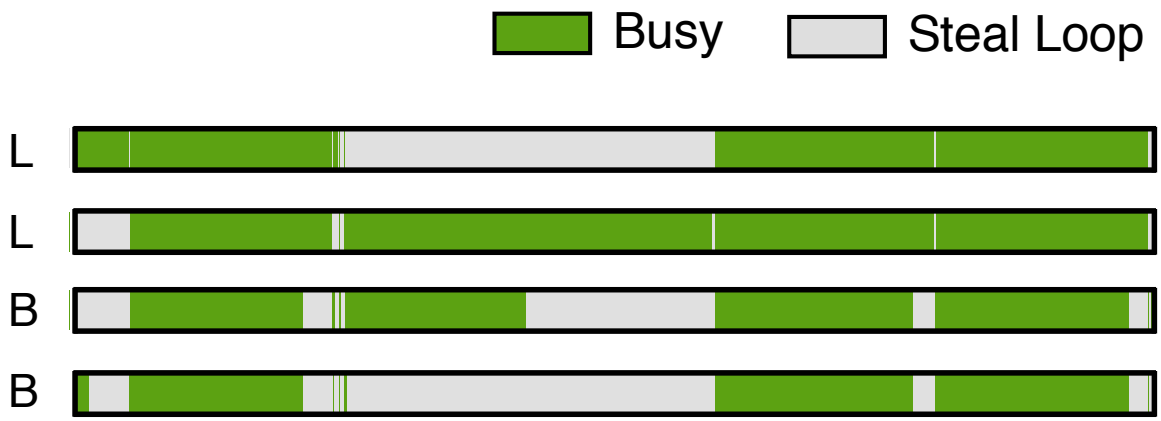
Application Kernels

- pbbs-bfs
- pbbs-quicksort
- pbbs-samplesort
- pbbs-dictionary
- pbbs-convex-hull
- pbbs-radix-sort
- pbbs-knn
- pbbs-max-independent-set
- pbbs-nbody
- pbbs-remove-duplicates
- pbbs-suffix-array
- pbbs-spanning-tree
- cilk-cholesky
- cilk-cilksort
- cilk-heat
- cilk-knapsack
- cilk-matrix-multiply
- parsec-blackscholes
- unbalanced-tree-search

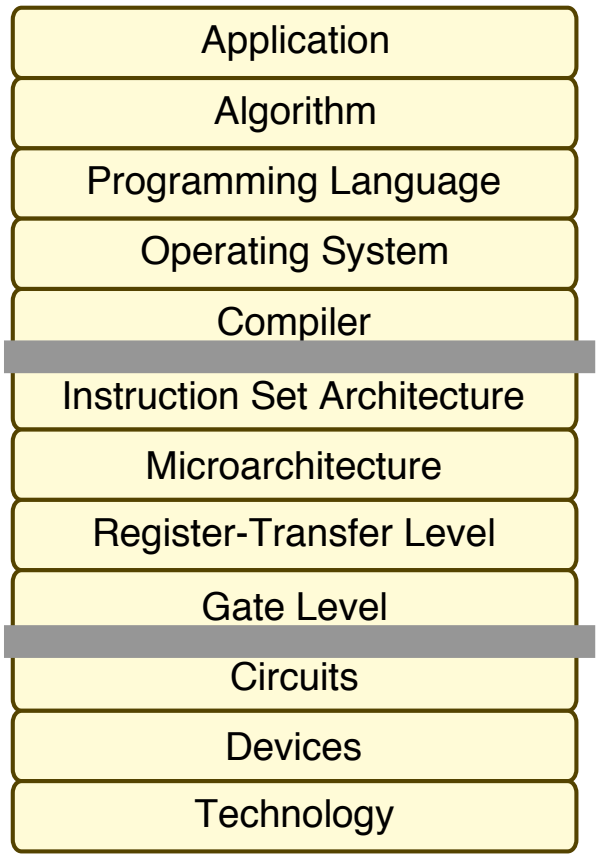
Performance	Median: 1.10 x	Max: 1.32 x
Energy Efficiency	Median: 1.11 x	Max: 1.53 x

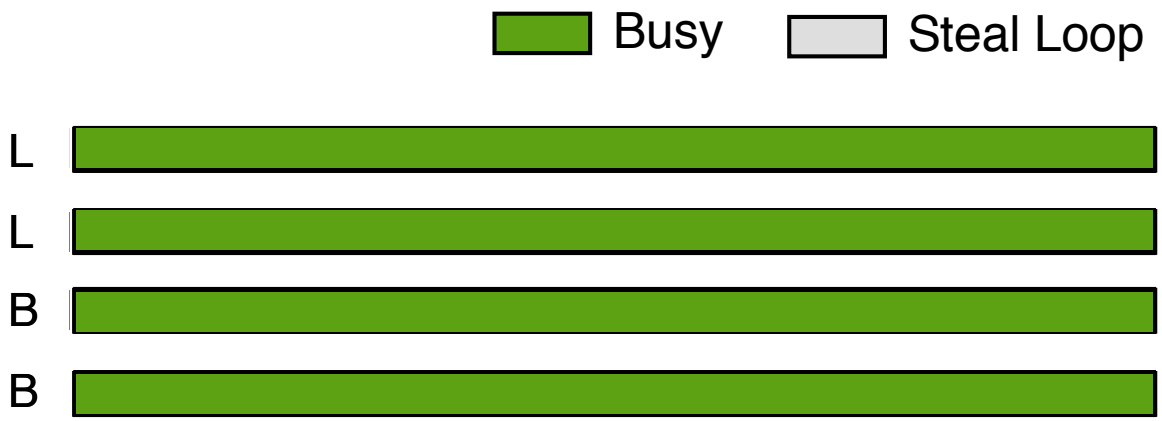
Augmenting the Software/Architecture Interface



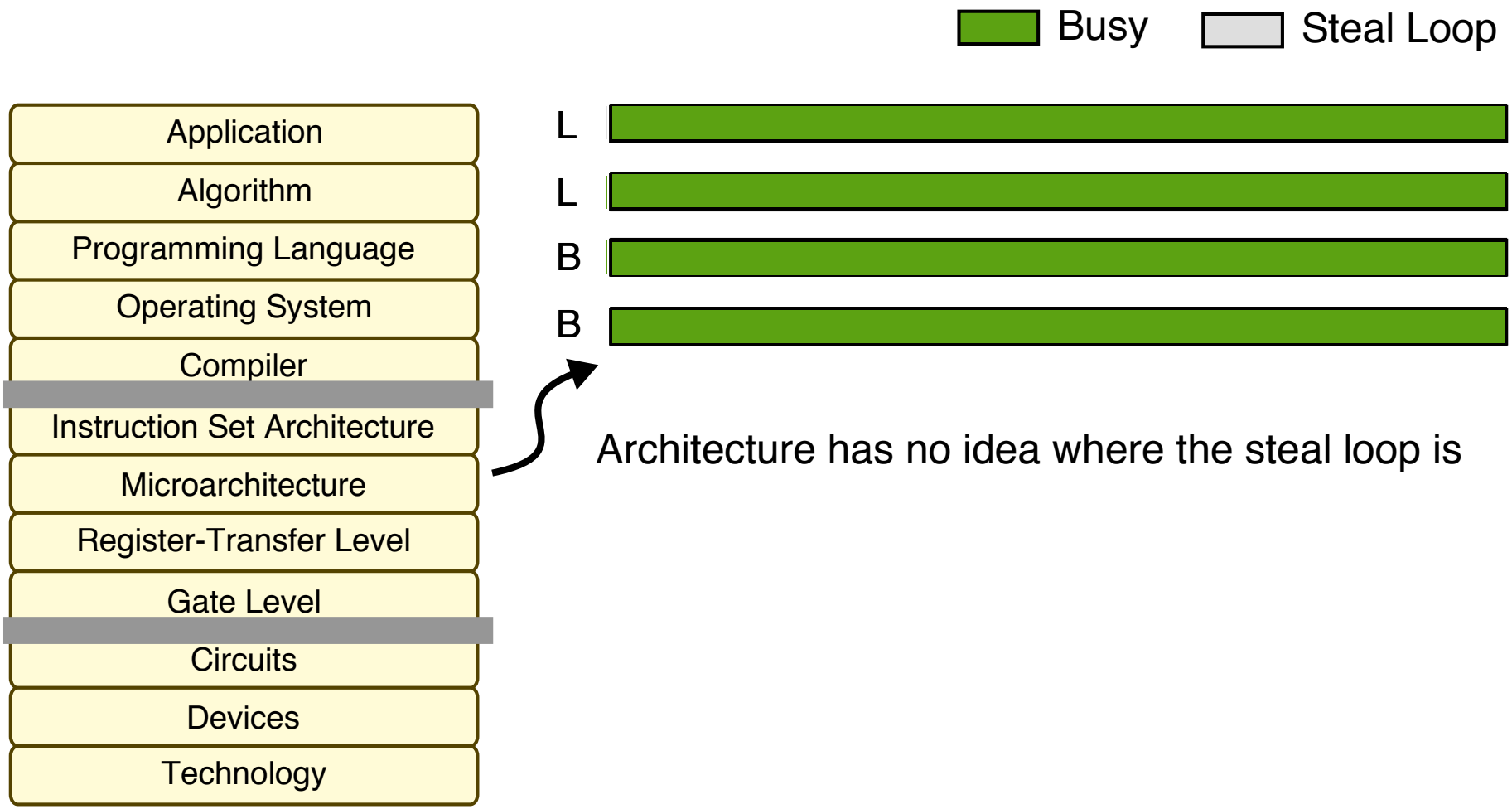


Augmenting the Software/Architecture Interface

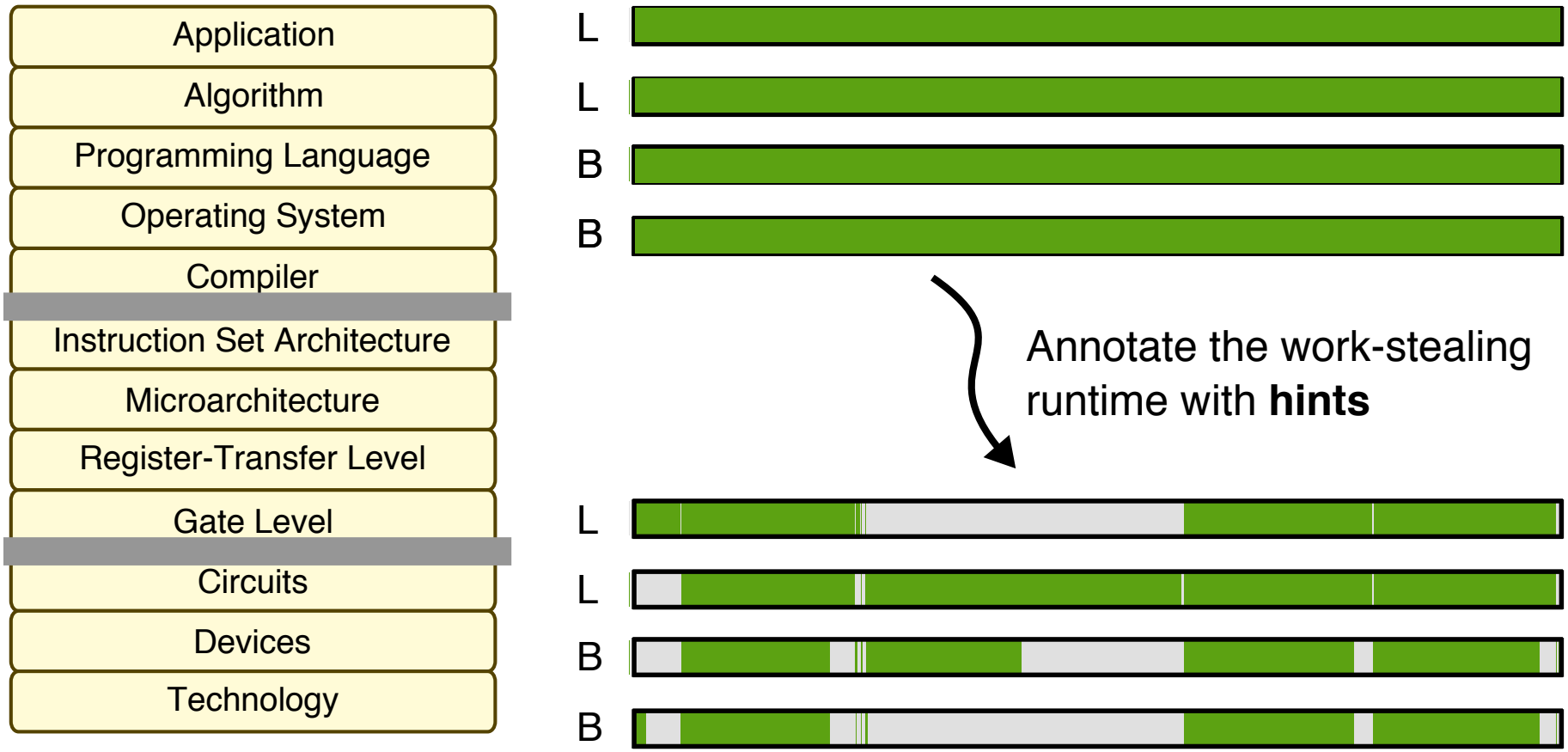




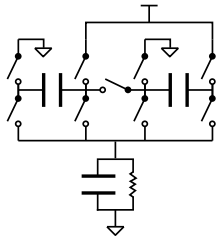
Augmenting the Software/Architecture Interface



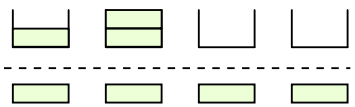
Augmenting the Software/Architecture Interface



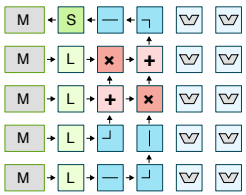
Software, Architecture, and VLSI Co-Design for Exploiting Fine-Grain Asymmetry



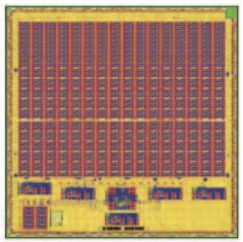
Exploiting Fine-Grain Asymmetry with
On-Chip Voltage Regulation - **MICRO'14**, IEEE TCAS I'18



Exploiting Fine-Grain Asymmetry in
Task-Based Parallel Runtimes - **ISCA'16**, MICRO'17, RISCV'18



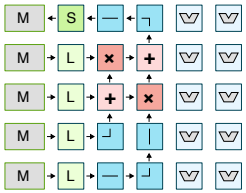
Exploiting Fine-Grain Asymmetry in
Coarse-Grain Reconfigurable Arrays - **Unpublished**



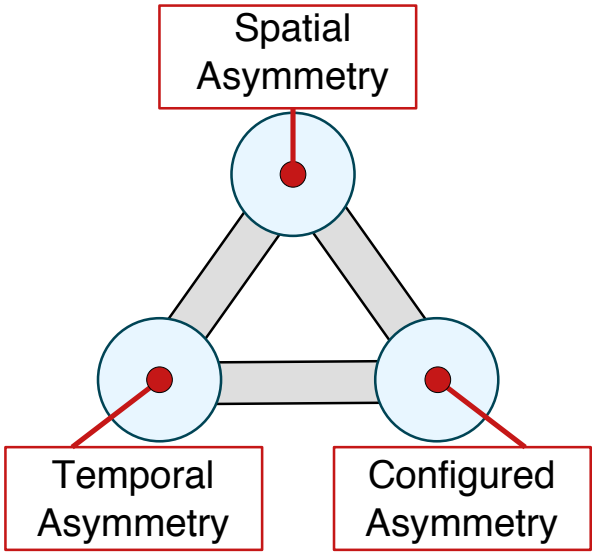
Exploiting Fine-Grain Asymmetry with
Silicon Prototyping - **IEEE MICRO'18**, DAC'18, Hotchips'17

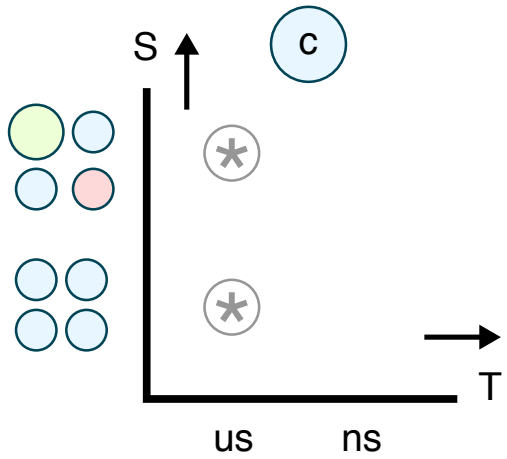
Conclusion

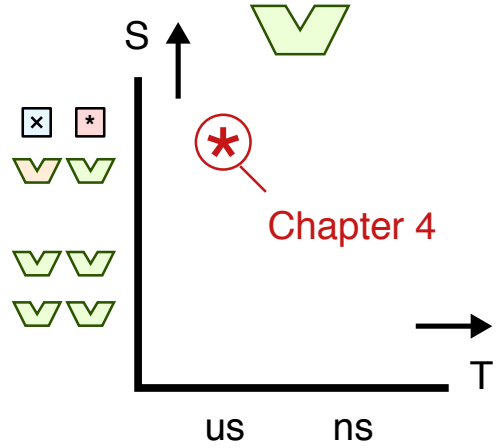
Software, Architecture, and VLSI Co-Design for Exploiting Fine-Grain Asymmetry



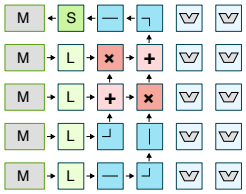
Exploiting Fine-Grain Asymmetry in **Coarse-Grain Reconfigurable Arrays** - Unpublished





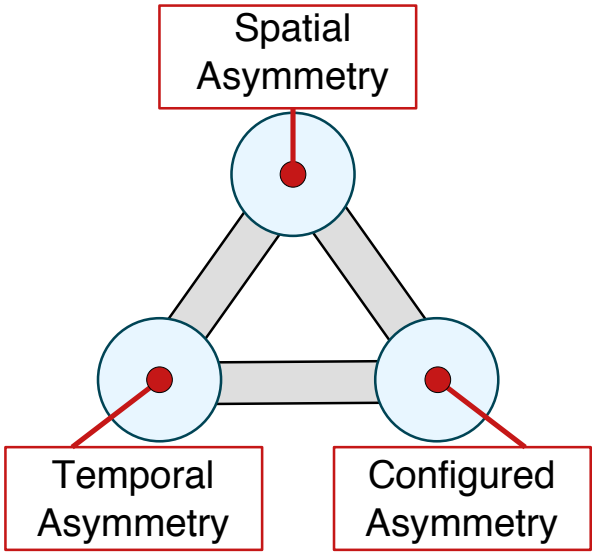


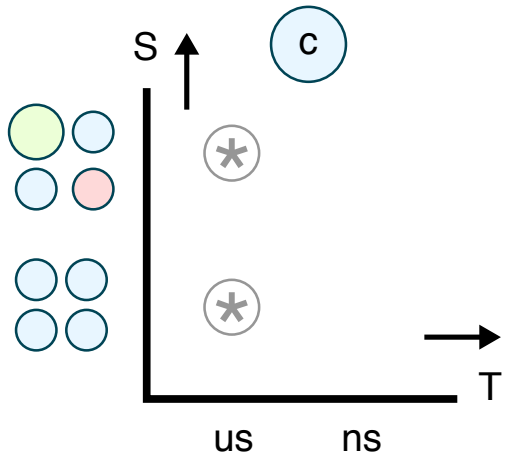
Software, Architecture, and VLSI Co-Design for Exploiting Fine-Grain Asymmetry

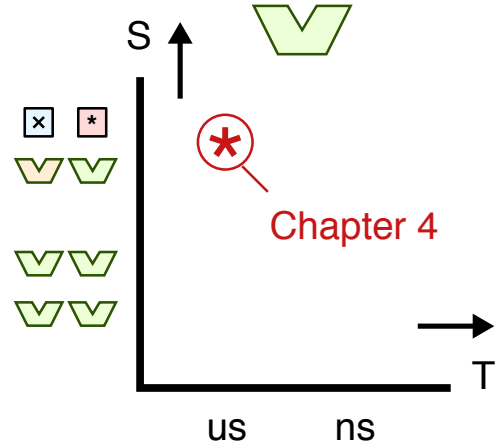


Exploiting Fine-Grain Asymmetry in

Coarse-Grain Reconfigurable Arrays - **Unpublished**

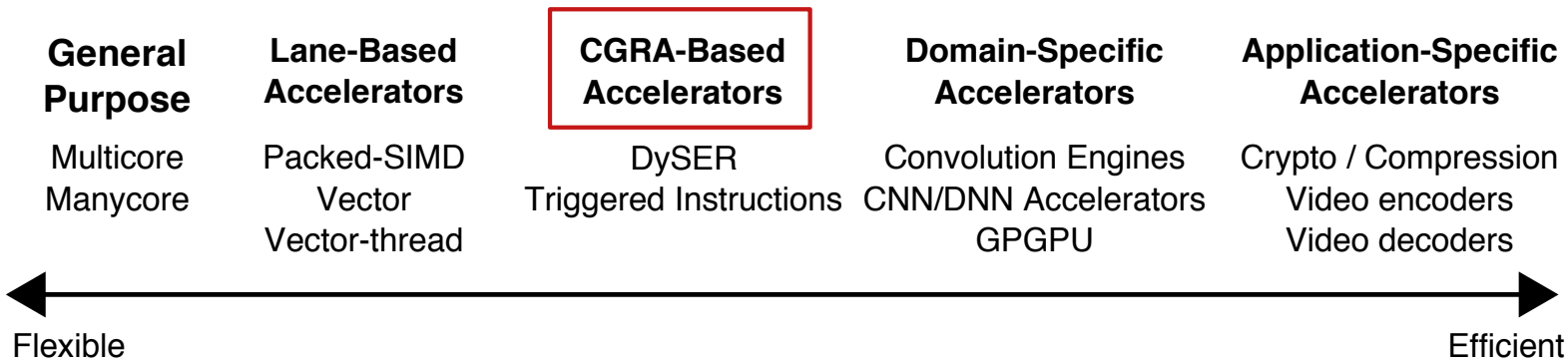




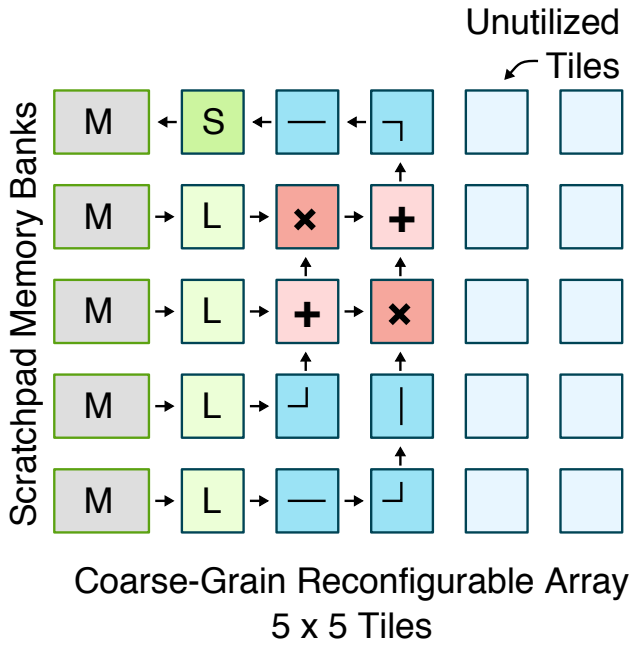


Research Question – Can we leverage fine-grain power-control techniques to optimize CGRAs at configure time?

Coarse-Grain Reconfigurable Arrays (CGRAs)



Coarse-Grain Reconfigurable Arrays (CGRAs)

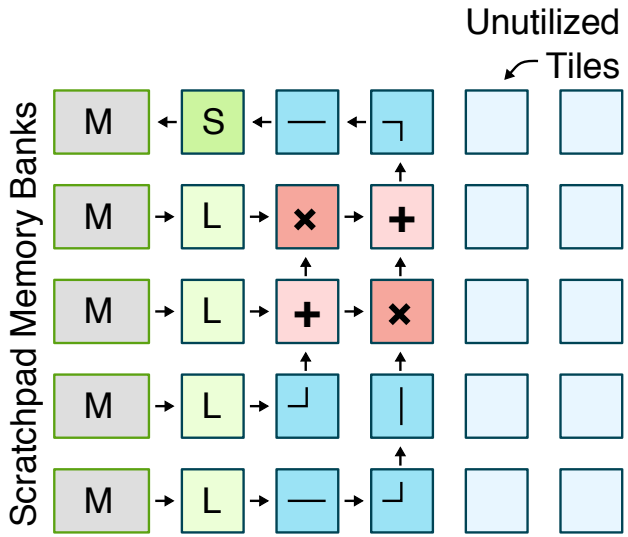


See also: Mei et al. "ADRES", FPL 2003

General Purpose	Lane-Based Accelerators	CGRA-Based Accelerators	Domain-Specific Accelerators	Application-Specific Accelerators
Multicore Manycore	Packed-SIMD Vector Vector-thread	DySER Triggered Instructions	Convolution Engines CNN/DNN Accelerators GPGPU	Crypto / Compression Video encoders Video decoders

Flexible ⟷ Efficient

Coarse-Grain Reconfigurable Arrays (CGRAs)



Data movement is more energy-consuming than computation, especially for deep learning

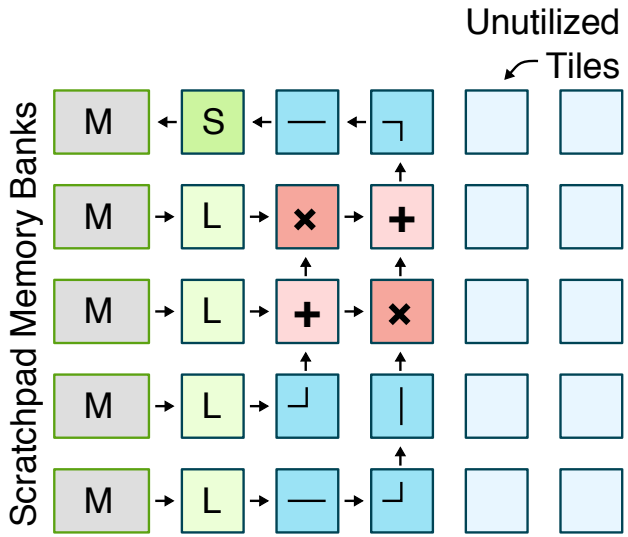
Chen et al. JSSC'16 "Eyeriss..."

See also: Mei et al. "ADRES", FPL 2003

General Purpose	Lane-Based Accelerators	CGRA-Based Accelerators	Domain-Specific Accelerators	Application-Specific Accelerators
Multicore Manycore	Packed-SIMD Vector Vector-thread	DySER Triggered Instructions	Convolution Engines CNN/DNN Accelerators GPGPU	Crypto / Compression Video encoders Video decoders

Flexible ⟷ Efficient

Coarse-Grain Reconfigurable Arrays (CGRAs)



Data movement is more energy-consuming than computation, especially for deep learning

Chen et al. JSSC'16 "Eyeriss..."

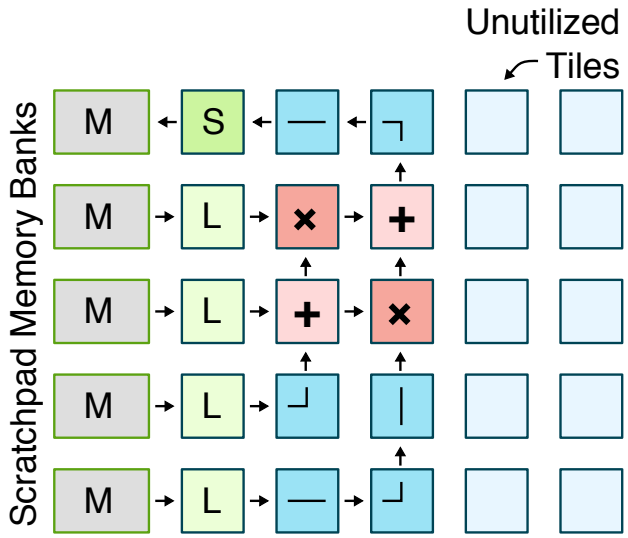
CGRAs are **dataflow architectures** that reduce data-movement energy to and from the memory hierarchy

See also: Mei et al. "ADRES", FPL 2003

General Purpose	**Lane-Based Accelerators**	**CGRA-Based Accelerators**	**Domain-Specific Accelerators**	**Application-Specific Accelerators**
Multicore Manycore	Packed-SIMD Vector Vector-thread	DySER Triggered Instructions	Convolution Engines CNN/DNN Accelerators GPGPU	Crypto / Compression Video encoders Video decoders

Flexible ⟷ Efficient

Coarse-Grain Reconfigurable Arrays (CGRAs)



Data movement is more energy-consuming than computation, especially for deep learning

Chen et al. JSSC'16 "Eyeriss..."

CGRAs are **dataflow architectures** that reduce data-movement energy to and from the memory hierarchy

CGRAs can efficiently map flexible workloads beyond those suitable for deep learning accelerators

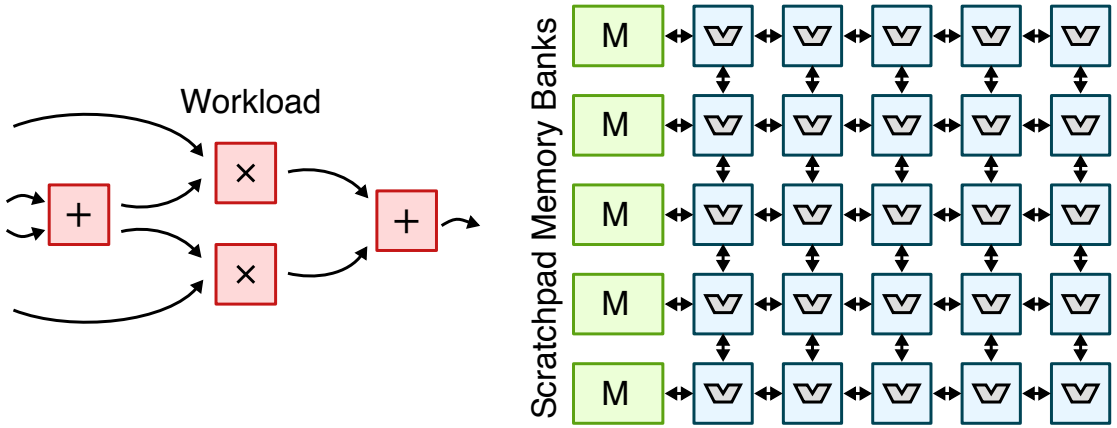
Fan et al. TVLSI'18
"Dual-Track CGRA for Object Inference"

See also: Mei et al. "ADRES", FPL 2003

General Purpose	**Lane-Based Accelerators**	**CGRA-Based Accelerators**	**Domain-Specific Accelerators**	**Application-Specific Accelerators**
Multicore Manycore	Packed-SIMD Vector Vector-thread	DySER Triggered Instructions	Convolution Engines CNN/DNN Accelerators GPGPU	Crypto / Compression Video encoders Video decoders

Flexible ← → Efficient

Key Challenges Facing CGRAs

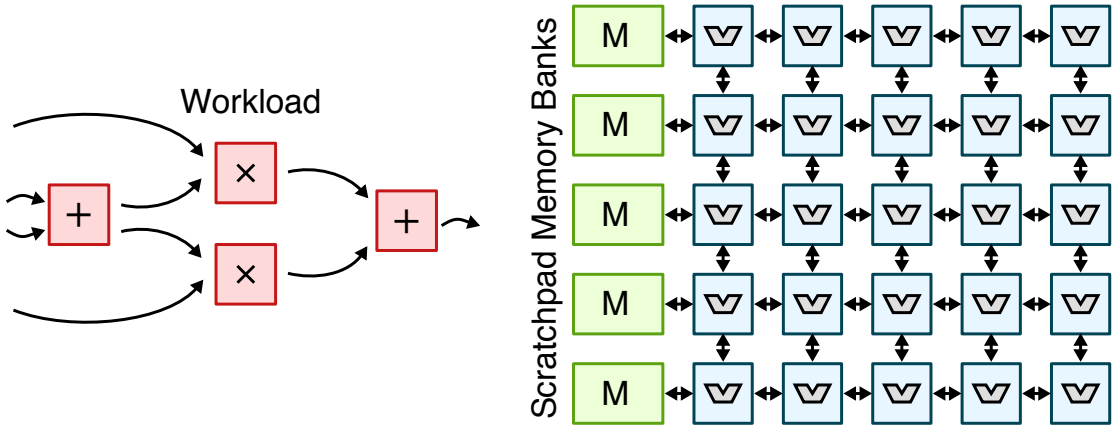


Key Challenge #1 – CGRA compilers must manage *complexity* with

- Resource constraints (routing, functional units, heterogeneity)
- Support for predication
- Dynamic timing behavior unknown at compile time
- Efficient targeting of kernels with high reuse (avoid reconfiguration)
- Over both space and time

See: HPCA'16, HPCA'11, IEEEMICRO'12, LCTES'09, TCAD'18, FPGA'13, FPL'03, FPL'96

Key Challenges Facing CGRAs



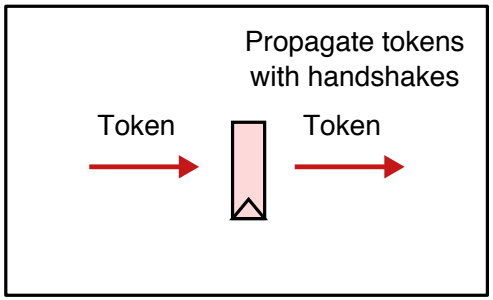
Key Challenge #2 – Complex compilers must optimize *utilization* with

- Abundant physical resources
- Potential heterogeneity across tiles
- Targeting kernels with high utilization (use more of the array)
- Over both space and time

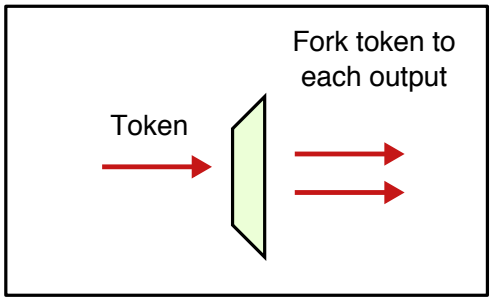
See: HPCA'16, HPCA'11, IEEEMICRO'12, LCTES'09, TCAD'18, FPGA'13, FPL'03, FPL'96

Elastic CGRAs – Huang et al. FPGA 2013

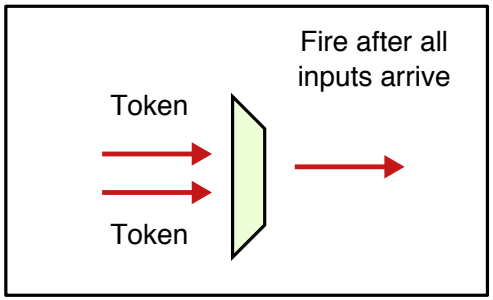
Elastic Buffers



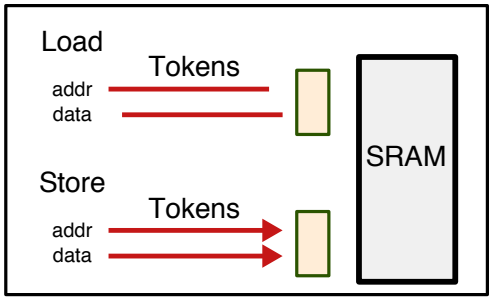
Eager Fork



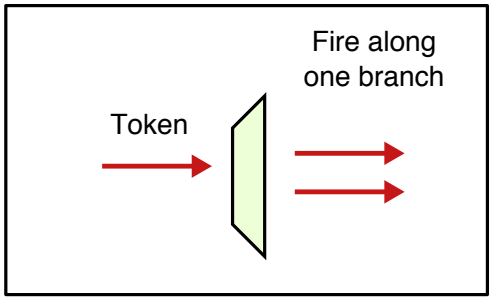
Join



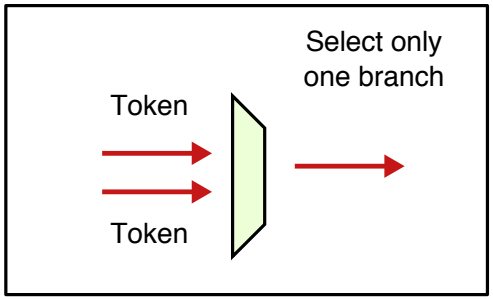
Memory Interfaces



Branch

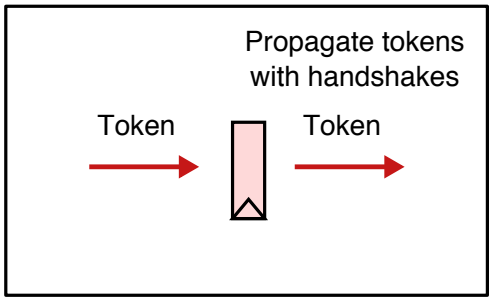


Merge

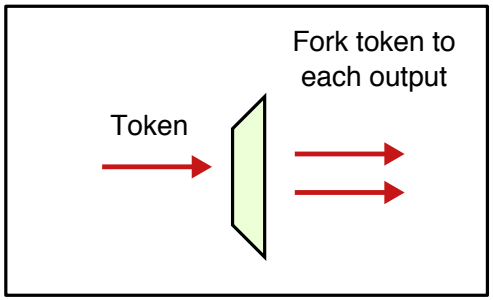


Elastic CGRAs – Huang et al. FPGA 2013

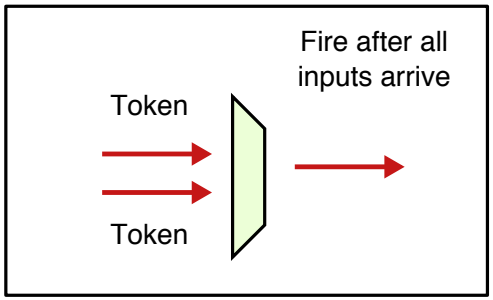
Elastic Buffers



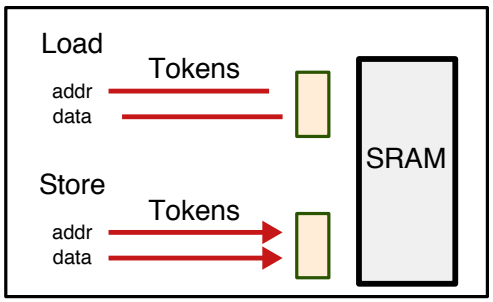
Eager Fork



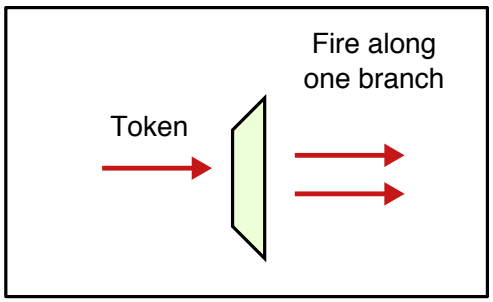
Join



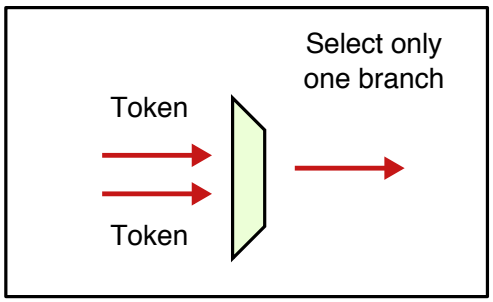
Memory Interfaces



Branch

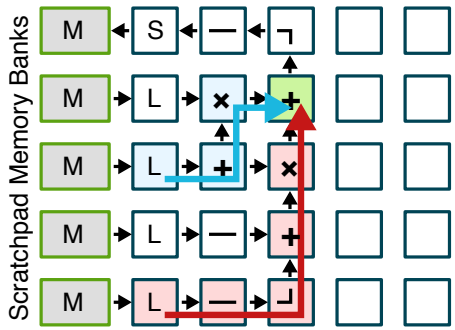


Merge



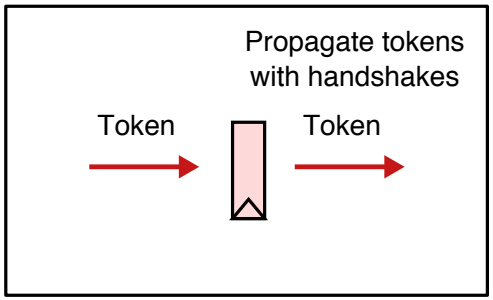
Hardware-managed elastic control flow can drastically simplify compiler complexity

Timing no longer affects **functionality**, only **performance**

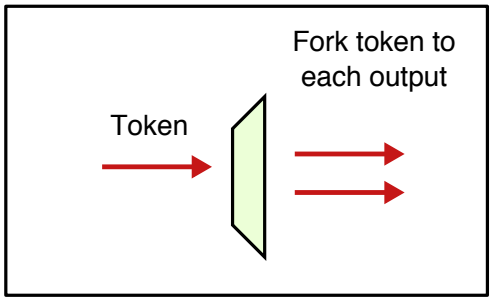


Elastic CGRAs – Huang et al. FPGA 2013

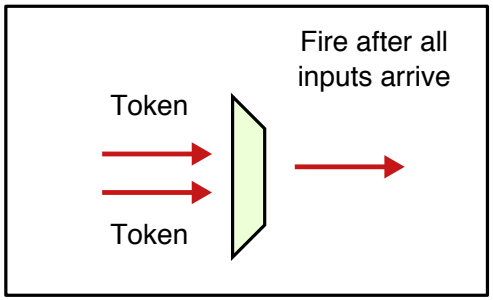
Elastic Buffers



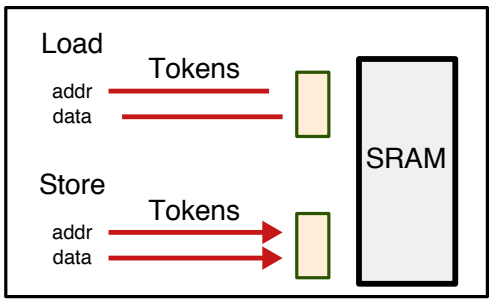
Eager Fork



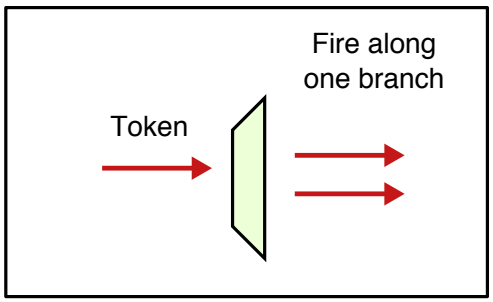
Join



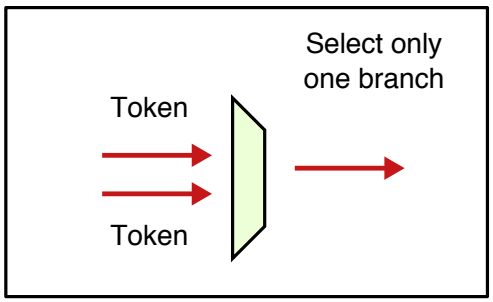
Memory Interfaces



Branch



Merge

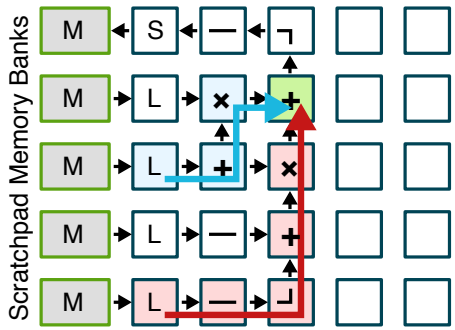


Hardware-managed elastic control flow can drastically simplify compiler complexity

Timing no longer affects **functionality**, only **performance**

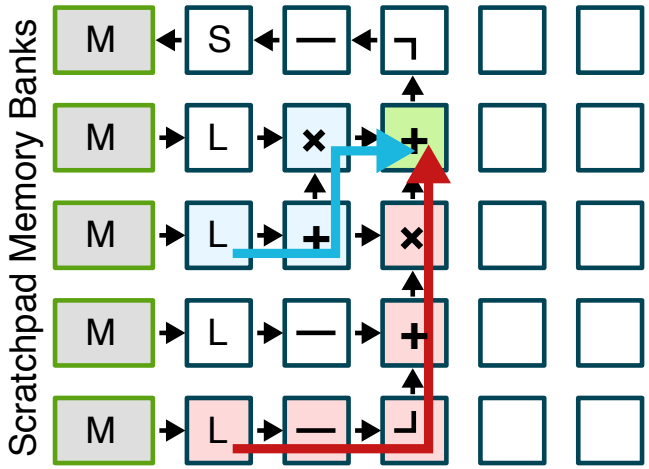
Two other examples of elasticity in CGRAs

Govindaraju et al. [HPCA'11] Fan et al. [TVLSI'18]



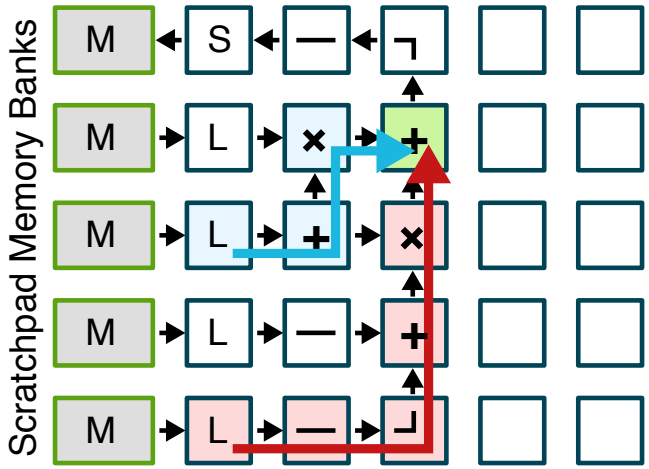
Fine-Grain Power Control Challenges

Can we use fine-grain DVFS to optimize execution on an elastic CGRA?



Fine-Grain Power Control Challenges

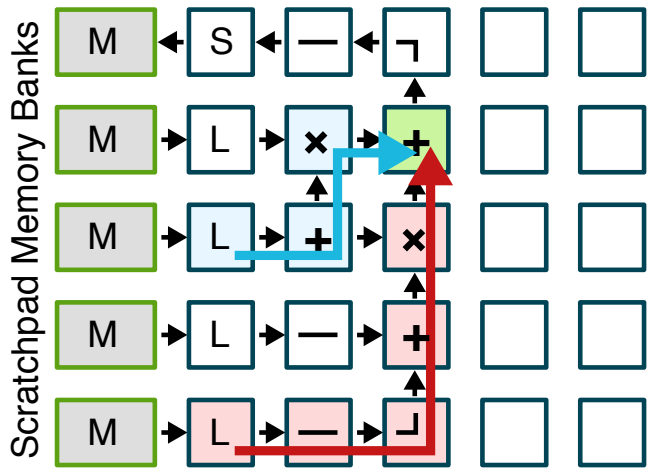
Can we use fine-grain DVFS to optimize execution on an elastic CGRA?



- Per-tile integrated voltage regulators are too large

Fine-Grain Power Control Challenges

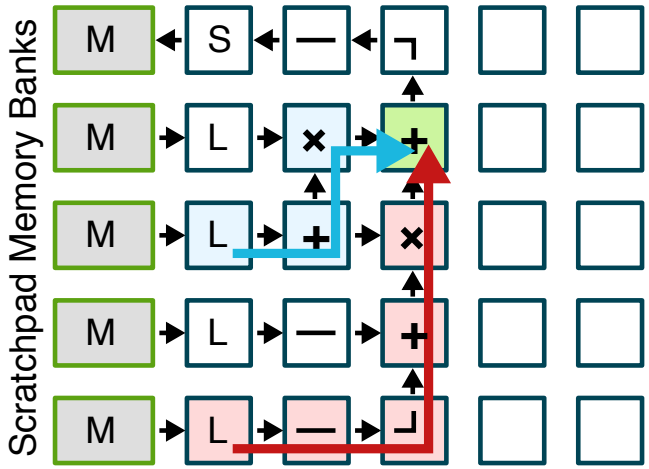
Can we use fine-grain DVFS to optimize execution on an elastic CGRA?



- Per-tile integrated voltage regulators are too large
- Per-tile PLLs are too large

Fine-Grain Power Control Challenges

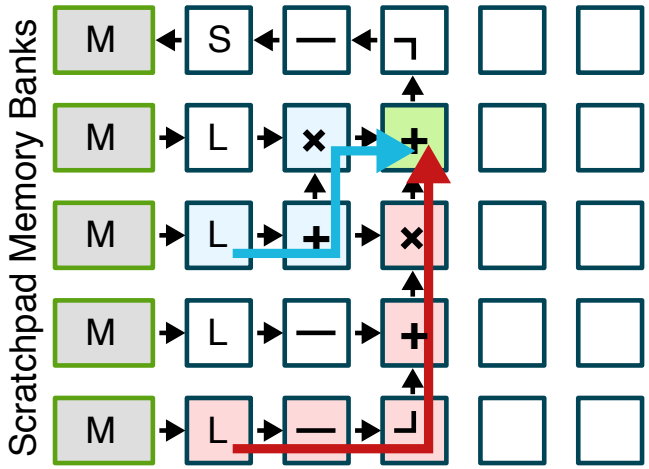
Can we use fine-grain DVFS to optimize execution on an elastic CGRA?



- Per-tile integrated voltage regulators are too large
- Per-tile PLLs are too large
- Local ring oscillators suffer from high phase noise

Fine-Grain Power Control Challenges

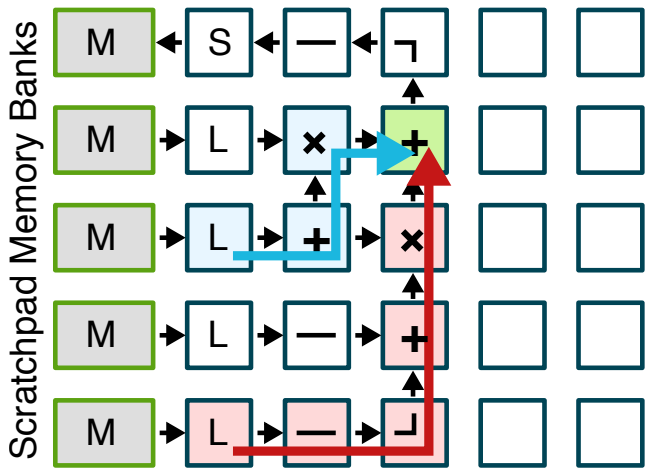
Can we use fine-grain DVFS to optimize execution on an elastic CGRA?

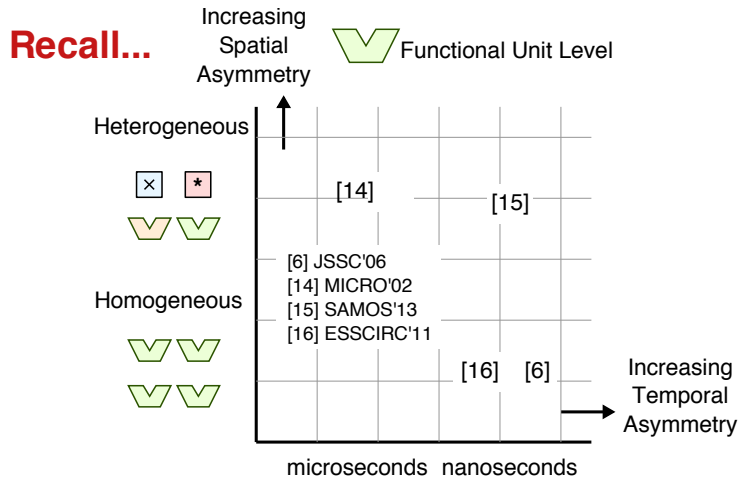


- Per-tile integrated voltage regulators are too large
- Per-tile PLLs are too large
- Local ring oscillators suffer from high phase noise
- Asynchronous crossings add synchronization latency

Fine-Grain Power Control Challenges

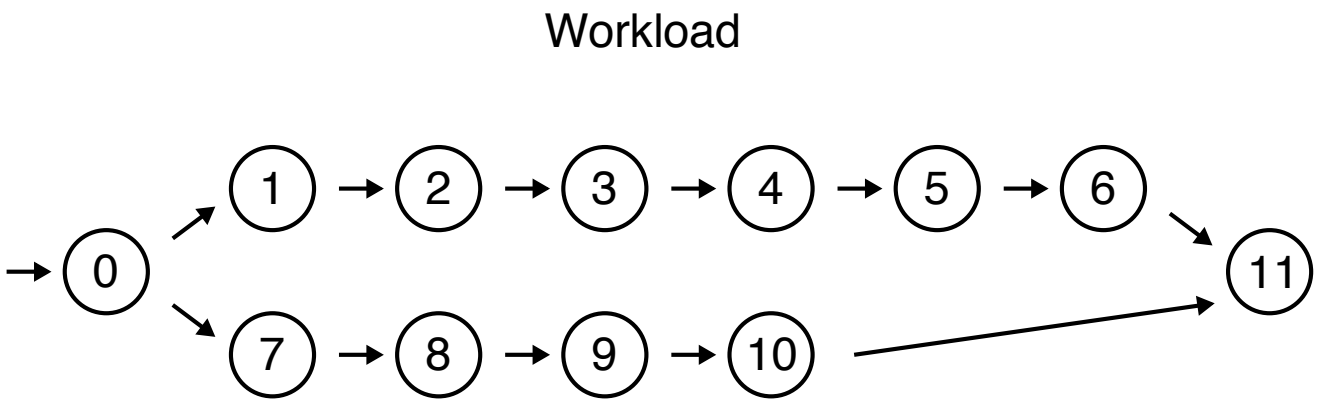
Can we use fine-grain DVFS to optimize execution on an elastic CGRA?

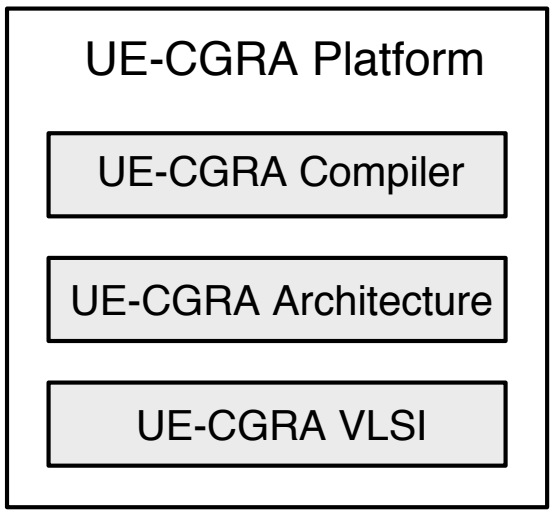


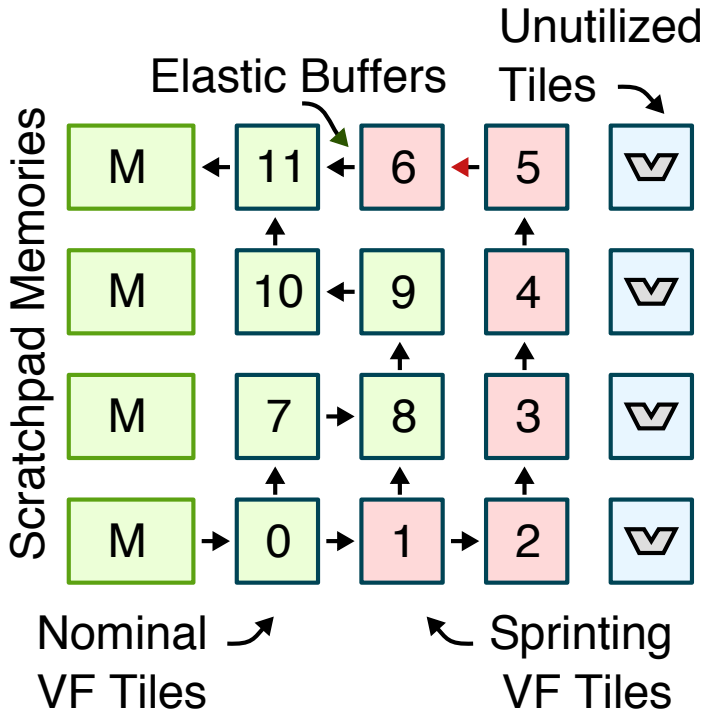


- Per-tile integrated voltage regulators are too large
- Per-tile PLLs are too large
- Local ring oscillators suffer from high phase noise
- Asynchronous crossings add synchronization latency

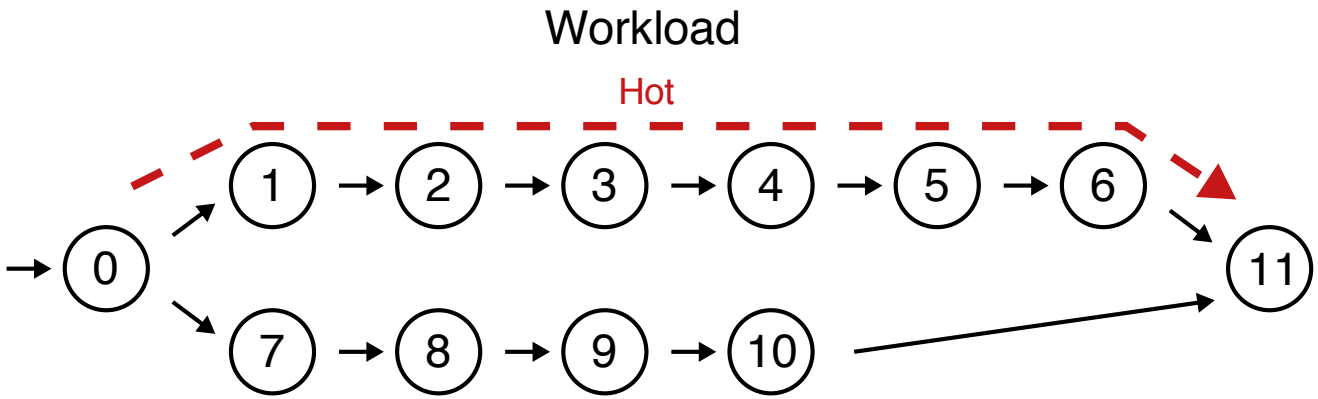
Ultra-Elastic CGRAs (UE-CGRAs)

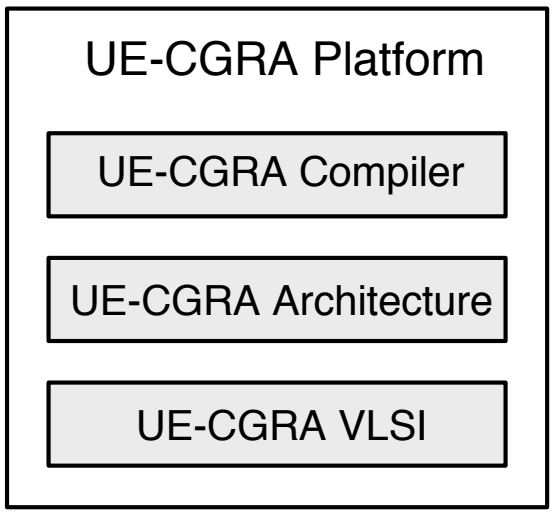


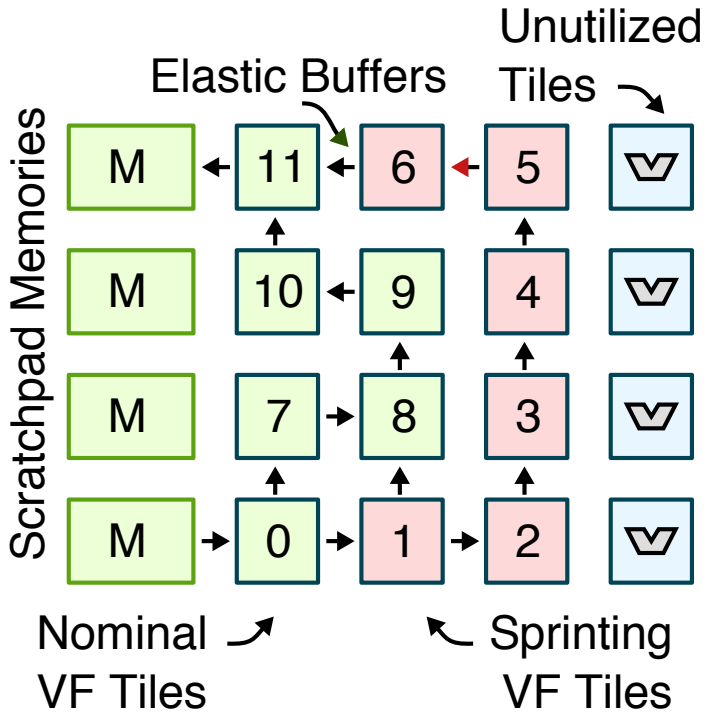


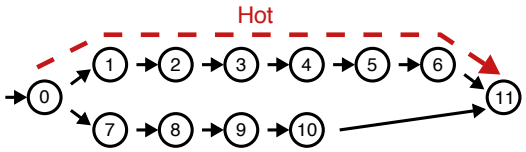


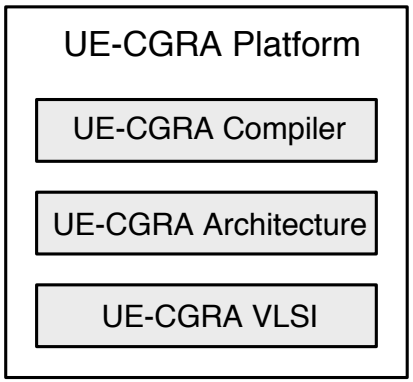
Ultra-Elastic CGRAs (UE-CGRAs)

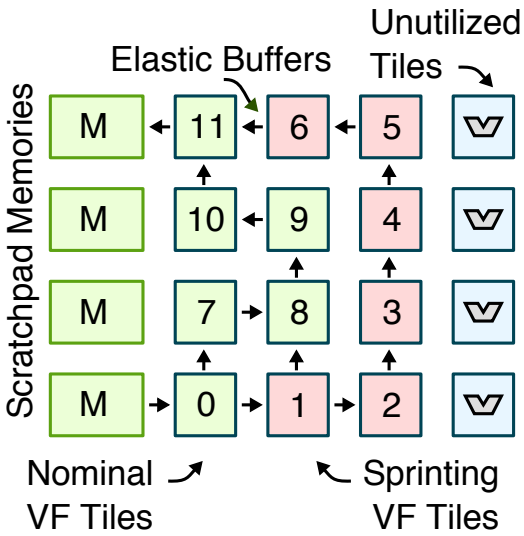








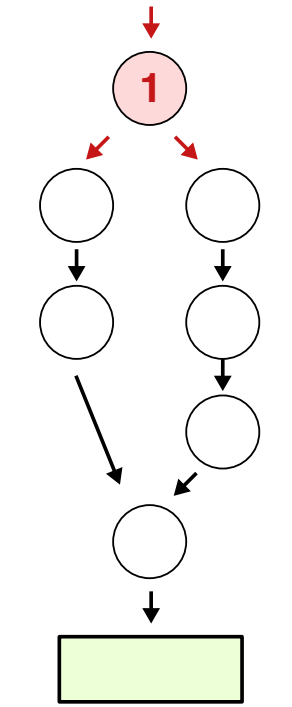




Exploiting Fine-Grain Asymmetry in Ultra-Elastic CGRAs

- Analytical Modeling
- UE-CGRA Compiler and Architecture
- UE-CGRA VLSI
- Methodology
- Evaluation

Elastic CGRA Analytical Performance Model

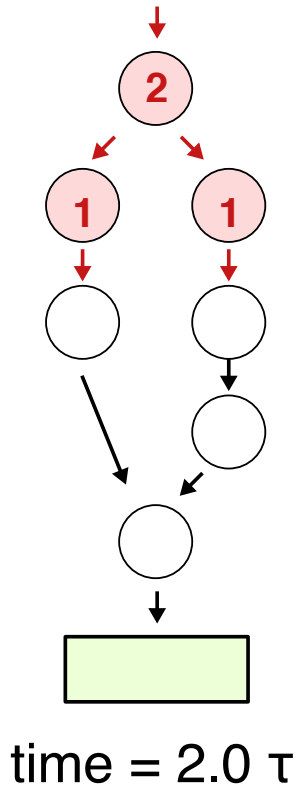


time = 1.0 τ

Discrete-event performance simulator

- Tokens flow and exit the dataflow graph

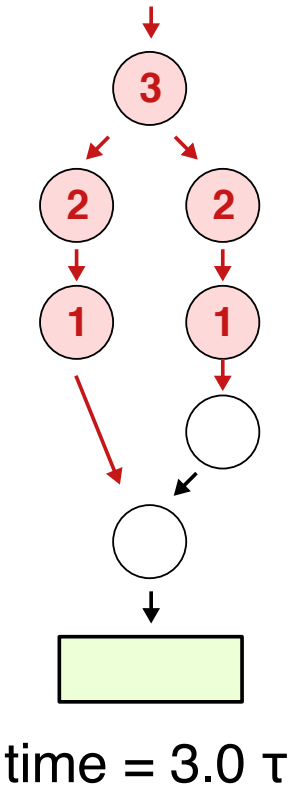
Elastic CGRA Analytical Performance Model



Discrete-event performance simulator

- Tokens flow and exit the dataflow graph

Elastic CGRA Analytical Performance Model

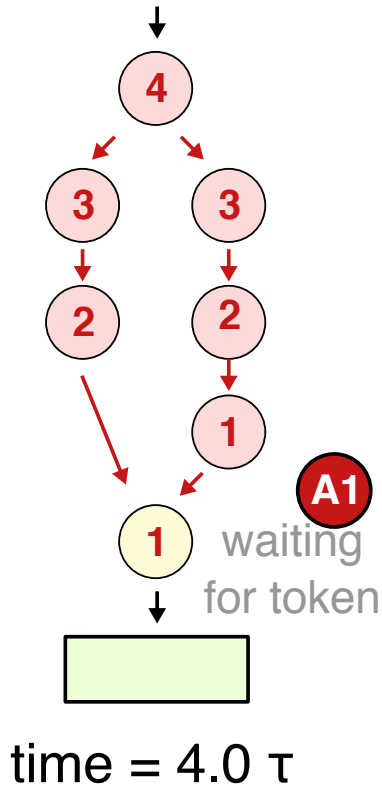


time = 3.0 τ

Discrete-event performance simulator

- Tokens flow and exit the dataflow graph

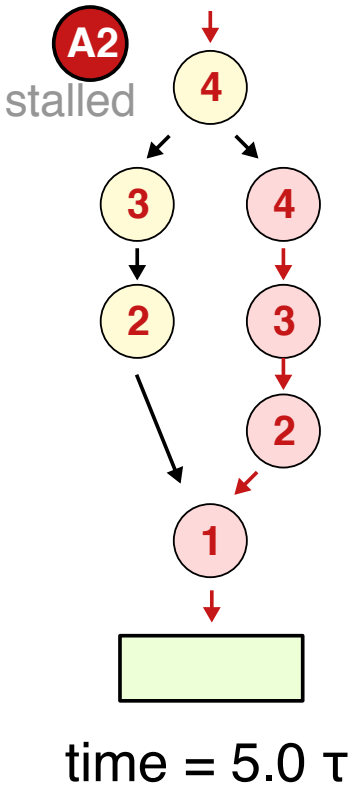
Elastic CGRA Analytical Performance Model



Discrete-event performance simulator

- Tokens flow and exit the dataflow graph
- Nodes fire when all input tokens arrive

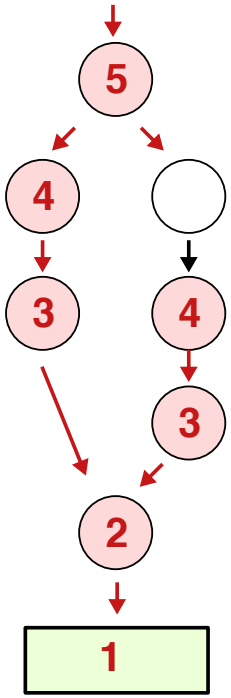
Elastic CGRA Analytical Performance Model



Discrete-event performance simulator

- Tokens flow and exit the dataflow graph
- Nodes fire when all input tokens arrive
- Nodes apply backpressure

Elastic CGRA Analytical Performance Model

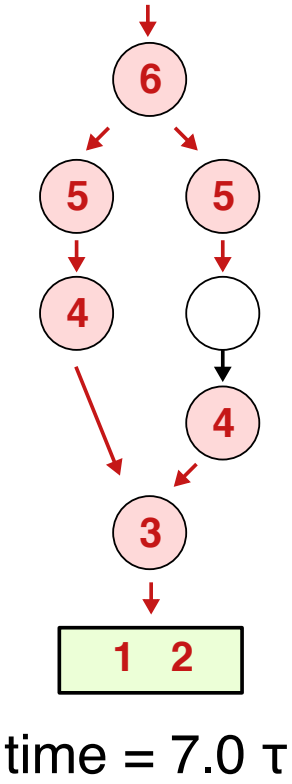


time = 6.0 τ

Discrete-event performance simulator

- Tokens flow and exit the dataflow graph
- Nodes fire when all input tokens arrive
- Nodes apply backpressure

Elastic CGRA Analytical Performance Model

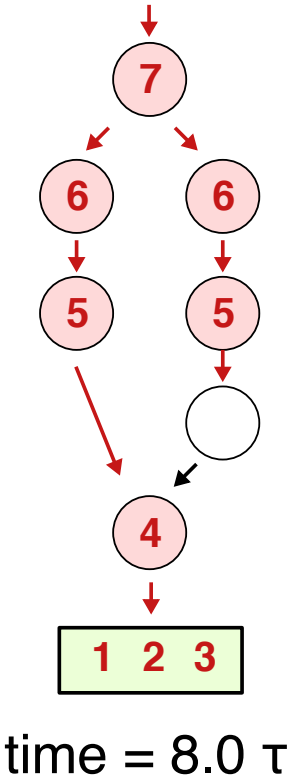


time = 7.0 τ

Discrete-event performance simulator

- Tokens flow and exit the dataflow graph
- Nodes fire when all input tokens arrive
- Nodes apply backpressure

Elastic CGRA Analytical Performance Model

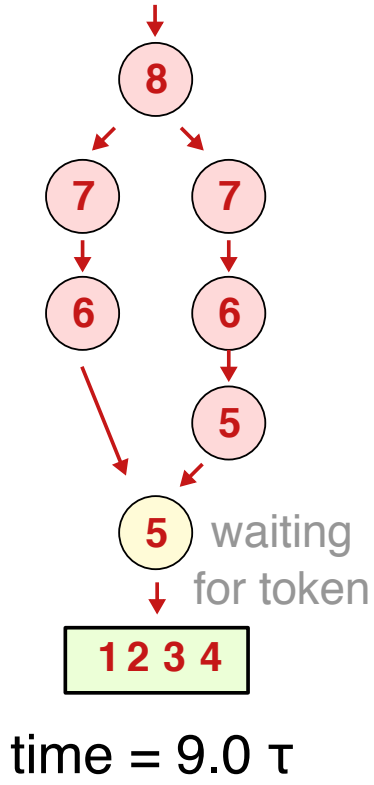


time = 8.0 τ

Discrete-event performance simulator

- Tokens flow and exit the dataflow graph
- Nodes fire when all input tokens arrive
- Nodes apply backpressure

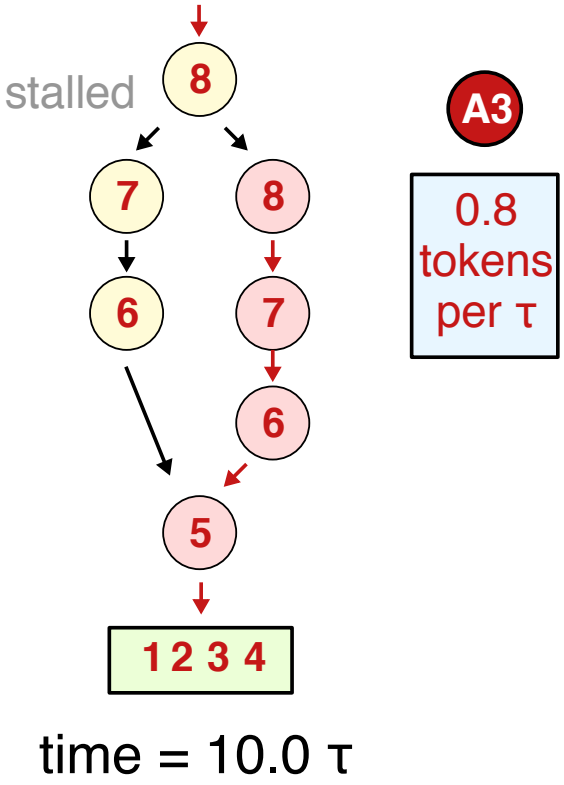
Elastic CGRA Analytical Performance Model



Discrete-event performance simulator

- Tokens flow and exit the dataflow graph
- Nodes fire when all input tokens arrive
- Nodes apply backpressure

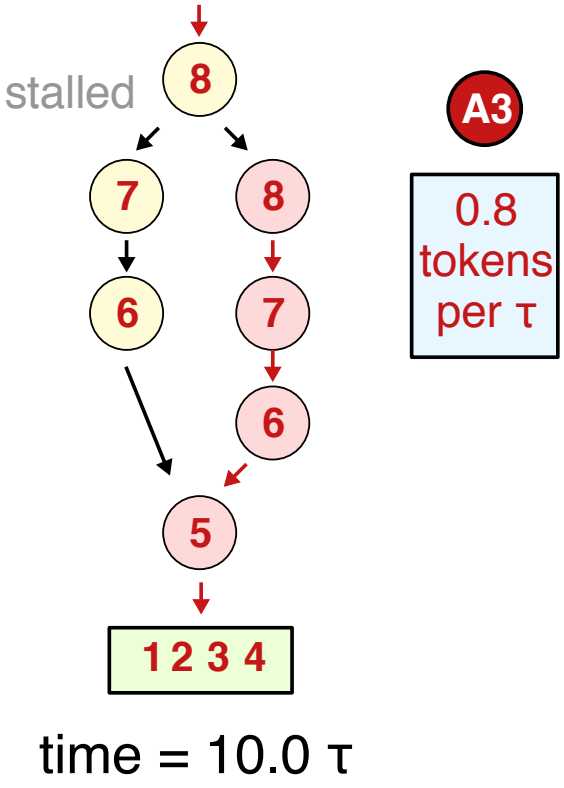
Elastic CGRA Analytical Performance Model



Discrete-event performance simulator

- Tokens flow and exit the dataflow graph
- Nodes fire when all input tokens arrive
- Nodes apply backpressure

Elastic CGRA Analytical Performance Model



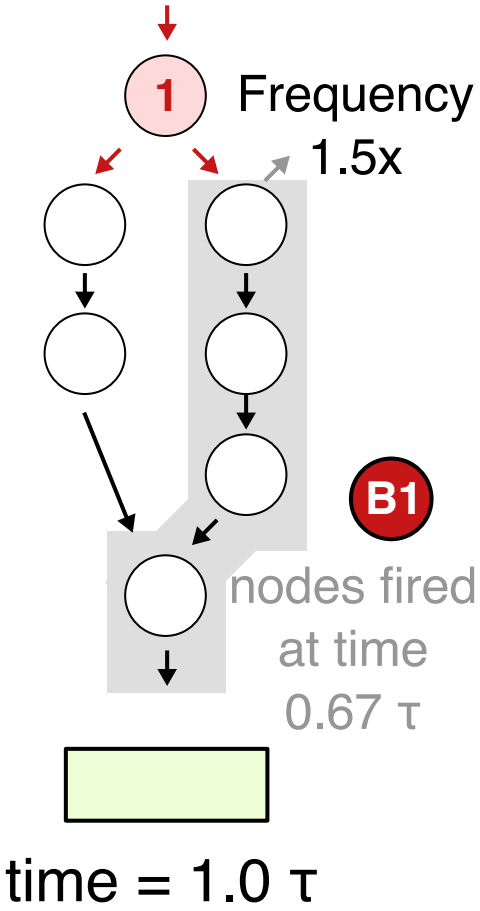
Discrete-event performance simulator

- Tokens flow and exit the dataflow graph
- Nodes fire when all input tokens arrive
- Nodes apply backpressure

Mapping to a real elastic CGRA

- All nodes map to a single unique tile
- Any tile can communicate with any other
- All tiles run at the same VF mode

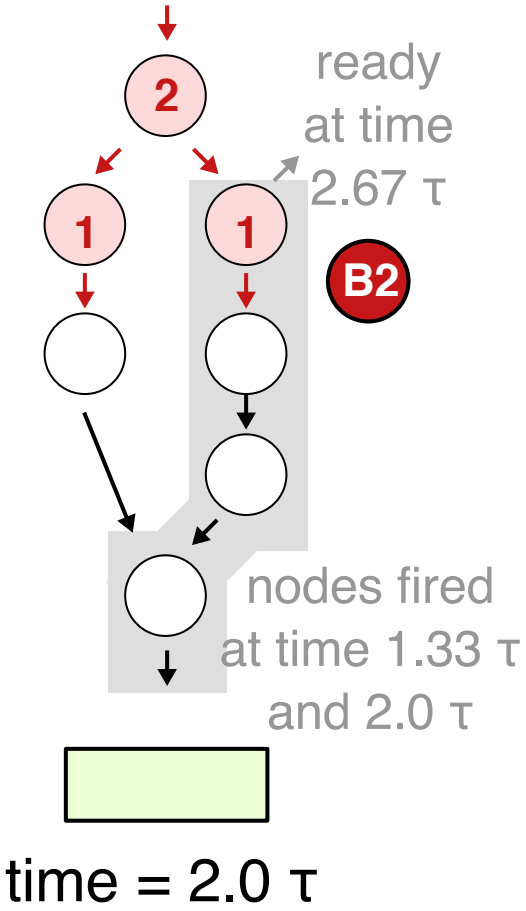
Ultra-Elastic CGRA Analytical Performance Model



Discrete-event performance simulator

- Nodes produce tokens at different rates
- Slow nodes produce at 1.0τ intervals

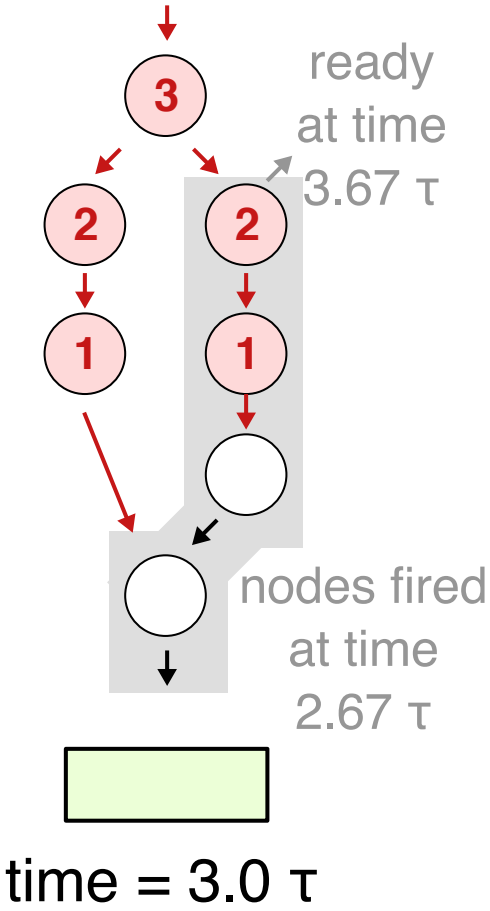
Ultra-Elastic CGRA Analytical Performance Model



Discrete-event performance simulator

- Nodes produce tokens at different rates
- Slow nodes produce at 1.0τ intervals
- Wire delays are modeled!

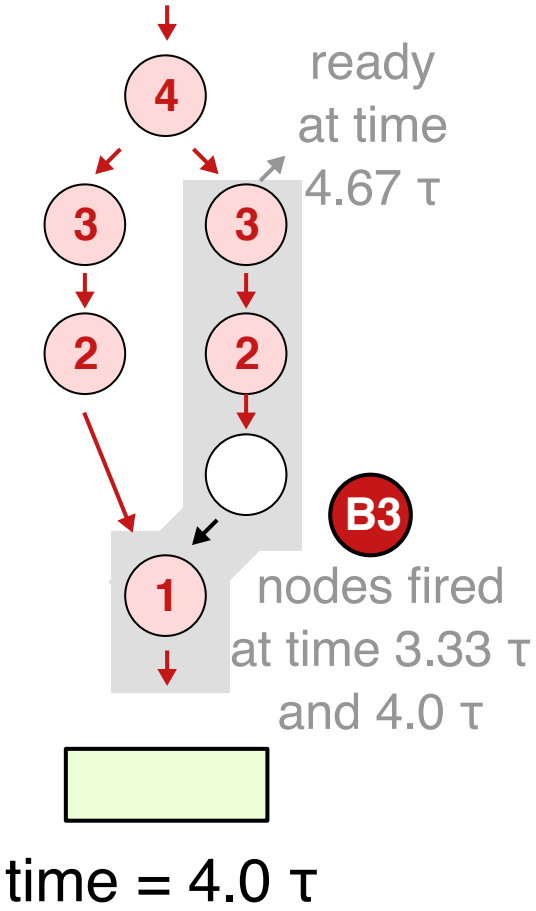
Ultra-Elastic CGRA Analytical Performance Model



Discrete-event performance simulator

- Nodes produce tokens at different rates
- Slow nodes produce at 1.0τ intervals
- Wire delays are modeled!

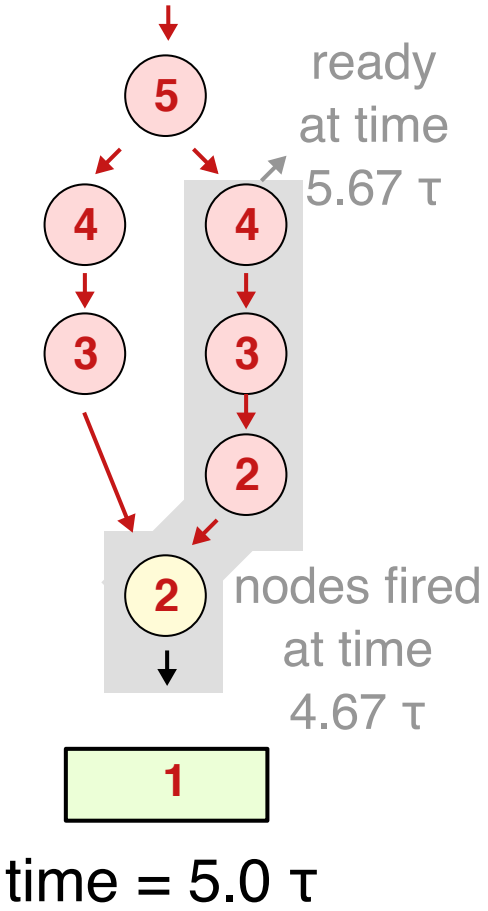
Ultra-Elastic CGRA Analytical Performance Model



Discrete-event performance simulator

- Nodes produce tokens at different rates
- Slow nodes produce at 1.0τ intervals
- Wire delays are modeled!
- Nodes may fire twice within 1.0τ

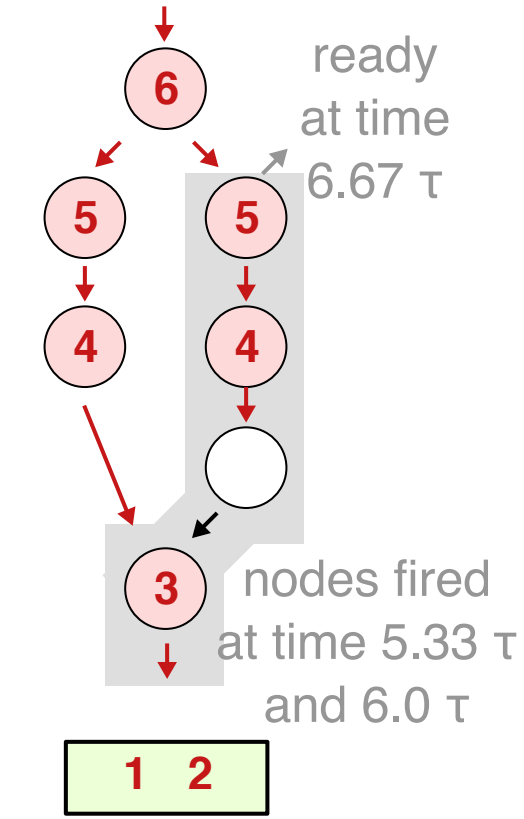
Ultra-Elastic CGRA Analytical Performance Model



Discrete-event performance simulator

- Nodes produce tokens at different rates
- Slow nodes produce at 1.0τ intervals
- Wire delays are modeled!
- Nodes may fire twice within 1.0τ
- Nodes may wait for short subperiods

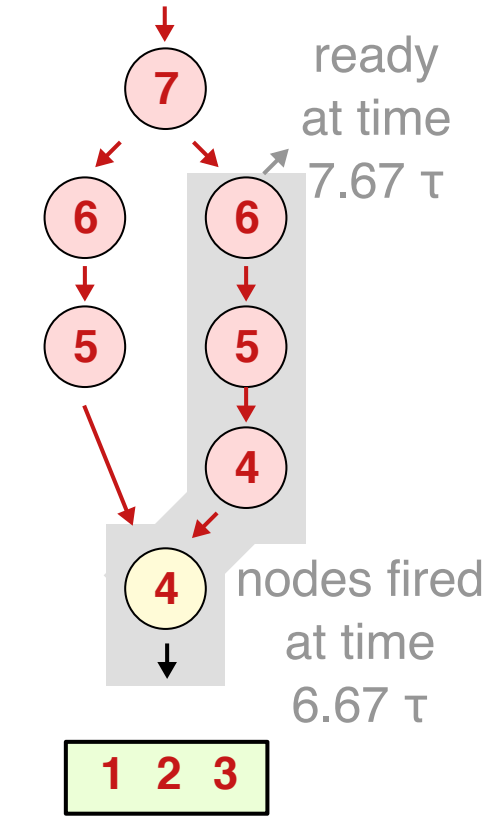
Ultra-Elastic CGRA Analytical Performance Model



Discrete-event performance simulator

- Nodes produce tokens at different rates
- Slow nodes produce at 1.0τ intervals
- Wire delays are modeled!
- Nodes may fire twice within 1.0τ
- Nodes may wait for short subperiods
- In this dataflow graph, backpressure is never applied, and the throughput is higher than in the elastic CGRA (0.8 tokens/τ vs. 1.0 tokensτ)

Ultra-Elastic CGRA Analytical Performance Model

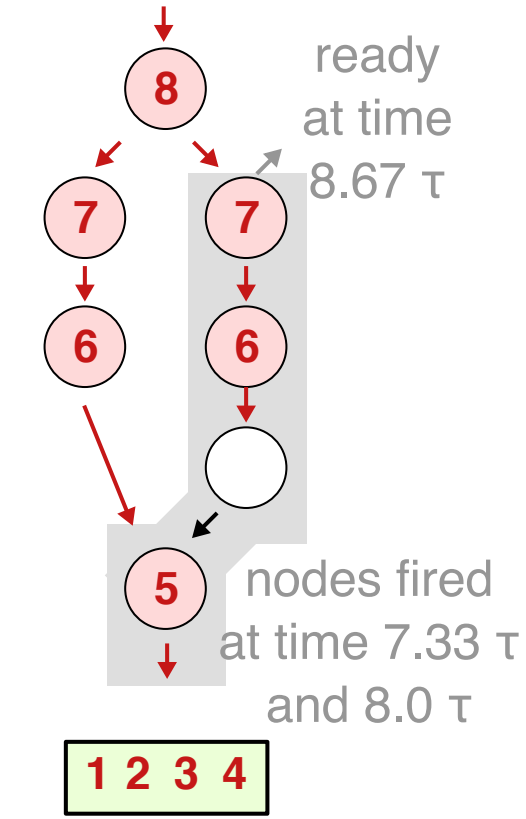


time = 7.0 τ

Discrete-event performance simulator

- Nodes produce tokens at different rates
- Slow nodes produce at 1.0τ intervals
- Wire delays are modeled!
- Nodes may fire twice within 1.0τ
- Nodes may wait for short subperiods
- In this dataflow graph, backpressure is never applied, and the throughput is higher than in the elastic CGRA (0.8 tokens/τ vs. 1.0 tokensτ)

Ultra-Elastic CGRA Analytical Performance Model

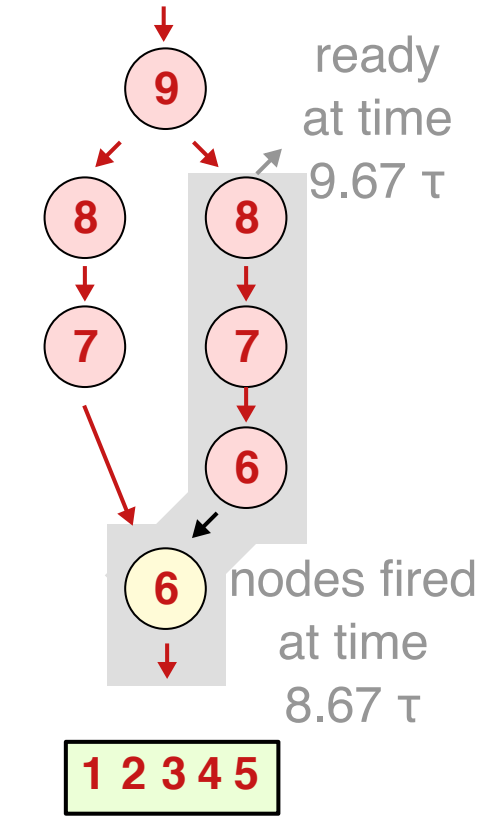


time = 8.0 τ

Discrete-event performance simulator

- Nodes produce tokens at different rates
- Slow nodes produce at 1.0τ intervals
- Wire delays are modeled!
- Nodes may fire twice within 1.0τ
- Nodes may wait for short subperiods
- In this dataflow graph, backpressure is never applied, and the throughput is higher than in the elastic CGRA (0.8 tokens/τ vs. 1.0 tokensτ)

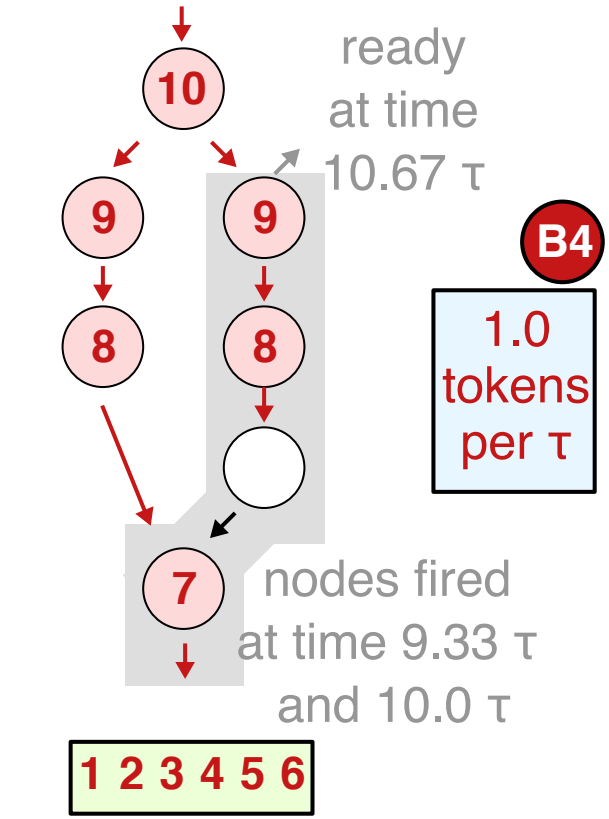
Ultra-Elastic CGRA Analytical Performance Model



Discrete-event performance simulator

- Nodes produce tokens at different rates
- Slow nodes produce at 1.0τ intervals
- Wire delays are modeled!
- Nodes may fire twice within 1.0τ
- Nodes may wait for short subperiods
- In this dataflow graph, backpressure is never applied, and the throughput is higher than in the elastic CGRA (0.8 tokens/τ vs. 1.0 tokensτ)

Ultra-Elastic CGRA Analytical Performance Model



Discrete-event performance simulator

- Nodes produce tokens at different rates
- Slow nodes produce at 1.0τ intervals
- Wire delays are modeled!
- Nodes may fire twice within 1.0τ
- Nodes may wait for short subperiods
- In this dataflow graph, backpressure is never applied, and the throughput is higher than in the elastic CGRA (0.8 tokens/τ vs. 1.0 tokensτ)

Analytical Energy Modeling

- First-order power equations

Analytical Energy Modeling

- First-order power equations
- Model parameters
  - Voltage and frequency relationship characterized in SPICE 28 nm

Analytical Energy Modeling

- First-order power equations
- Model parameters
  - Voltage and frequency relationship characterized in SPICE 28 nm
  - Leakage power estimated for both tiles and SRAM banks as a fraction of a tile executing a multiply at nominal VF

Analytical Energy Modeling

- First-order power equations
- Model parameters
  - Voltage and frequency relationship characterized in SPICE 28 nm
  - Leakage power estimated for both tiles and SRAM banks as a fraction of a tile executing a multiply at nominal VF
  - Tiles executing different operations (e.g., `mul`, `add`, `shift`, `SRAM`) have different multiplicative factors (e.g., tile executing `add` uses 0.23 $\times$ power compared to multiply)

Analytical Energy Modeling

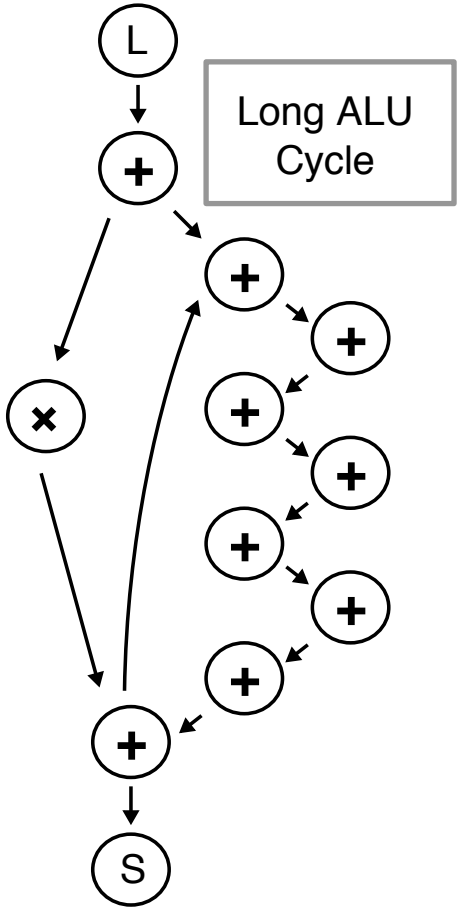
- First-order power equations
- Model parameters
  - Voltage and frequency relationship characterized in SPICE 28 nm
  - Leakage power estimated for both tiles and SRAM banks as a fraction of a tile executing a multiply at nominal VF
  - Tiles executing different operations (e.g., `mul`, `add`, `shift`, `SRAM`) have different multiplicative factors (e.g., tile executing `add` uses 0.23 $\times$ power compared to multiply)
- Leakage *energy* is estimated given the performance from the discrete-event model

Analytical Energy Modeling

- First-order power equations
- Model parameters
  - Voltage and frequency relationship characterized in SPICE 28 nm
  - Leakage power estimated for both tiles and SRAM banks as a fraction of a tile executing a multiply at nominal VF
  - Tiles executing different operations (e.g., `mul`, `add`, `shift`, `SRAM`) have different multiplicative factors (e.g., tile executing `add` uses 0.23 $\times$ power compared to multiply)
- Leakage *energy* is estimated given the performance from the discrete-event model
- Voltage and frequency levels
  - Rest voltage – 0.61 V at 3 $\times$ slower frequency
  - Nominal voltage – 0.90 V at 1 $\times$ frequency
  - Sprint voltage – 1.23 V at 1.5 $\times$ frequency

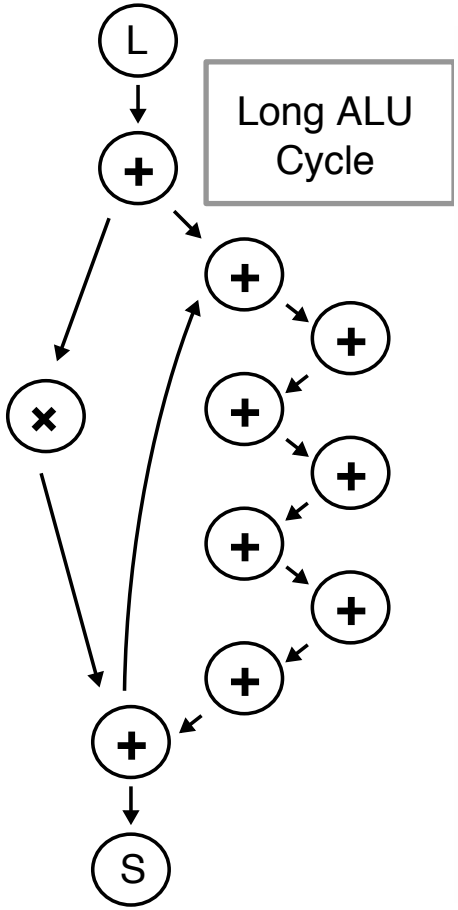
UE-CGRA Analytical Case Study #1

Dataflow Graph

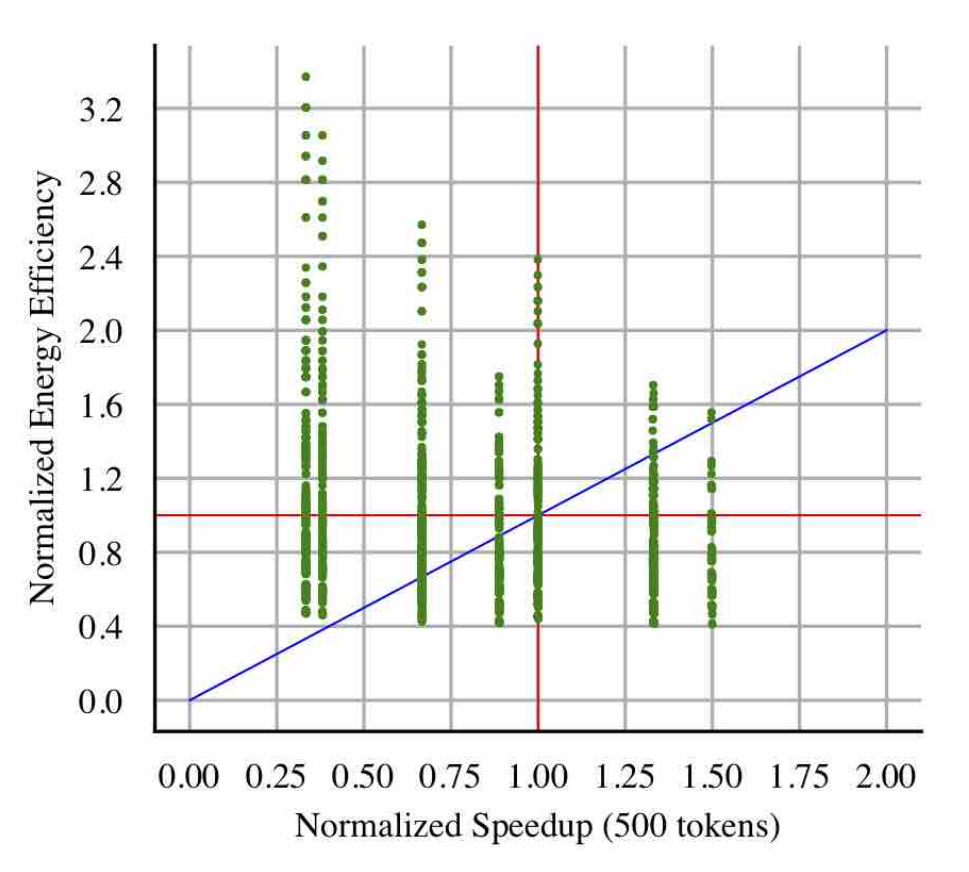


UE-CGRA Analytical Case Study #1

Dataflow Graph

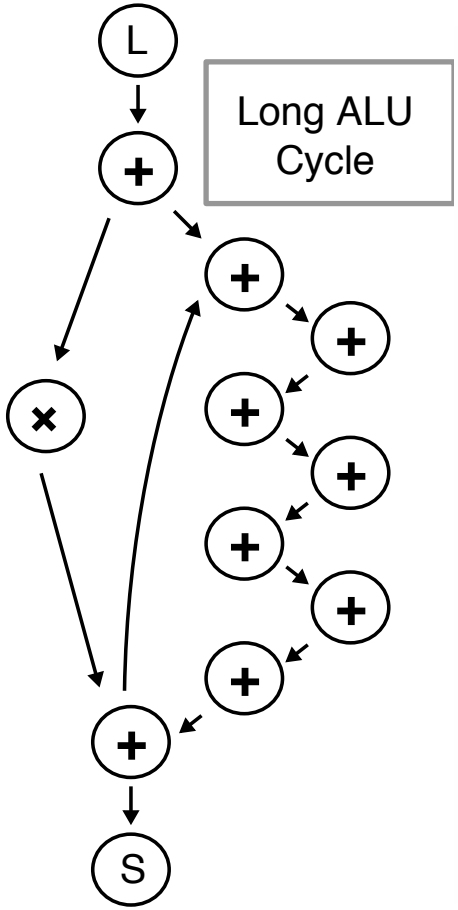


Analytical Sweep

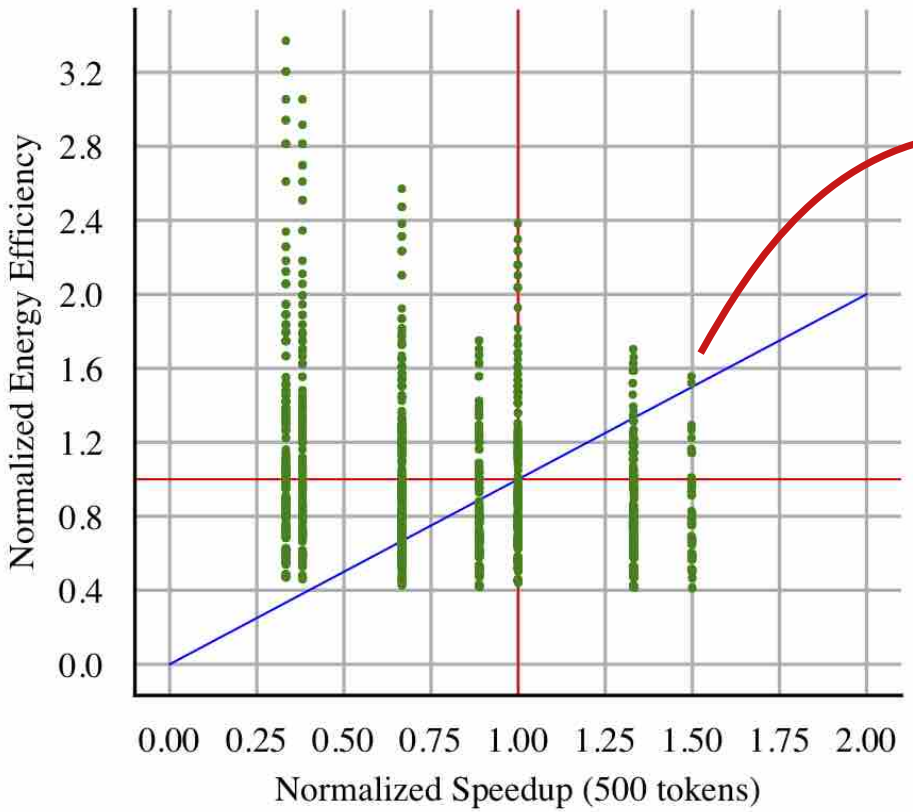


UE-CGRA Analytical Case Study #1

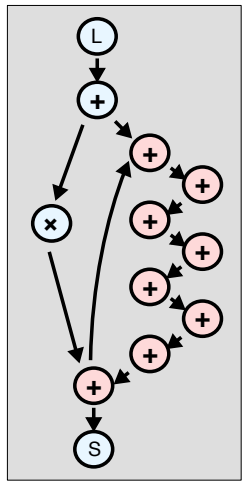
Dataflow Graph



Analytical Sweep



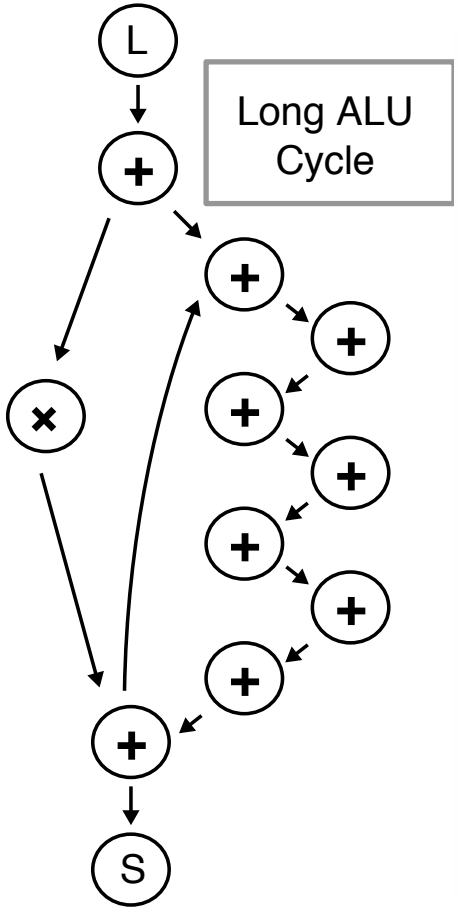
1.5x speedup
1.6x energy eff.



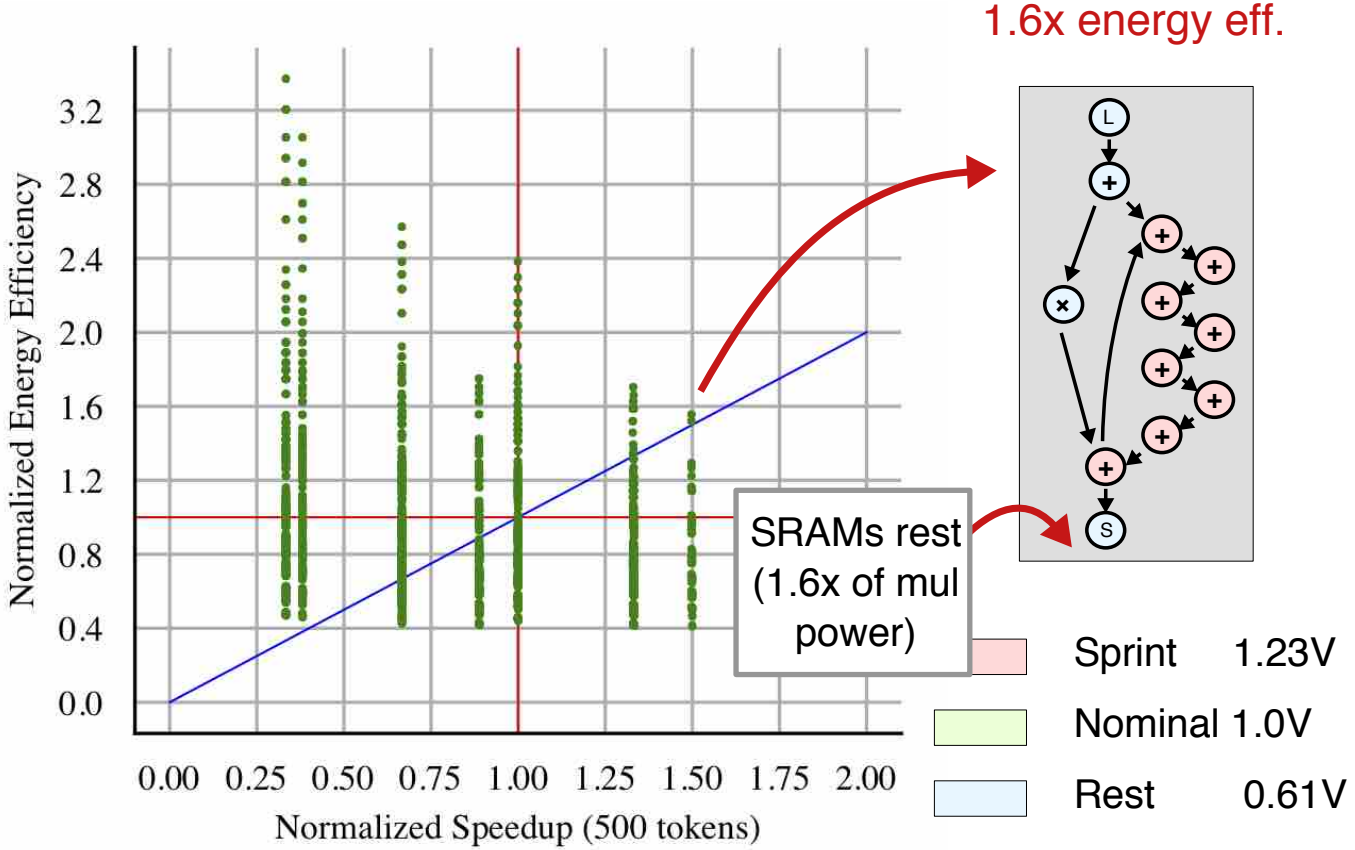
Sprint 1.23V
Nominal 1.0V
Rest 0.61V

UE-CGRA Analytical Case Study #1

Dataflow Graph

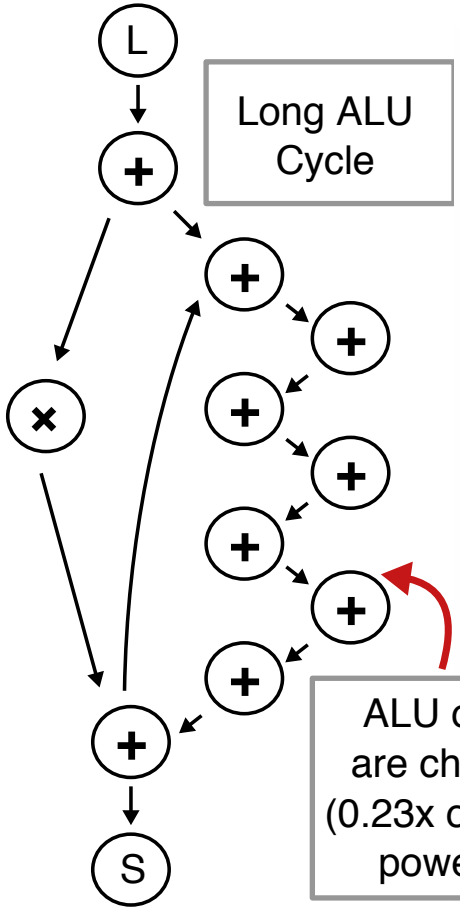


Analytical Sweep

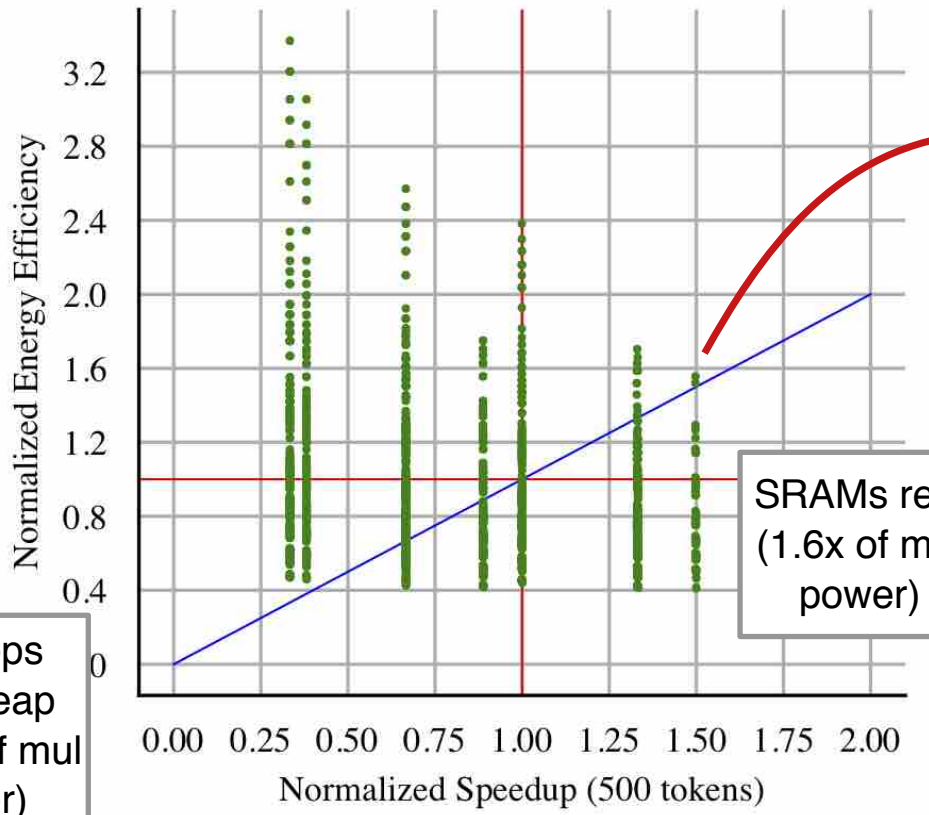


UE-CGRA Analytical Case Study #1

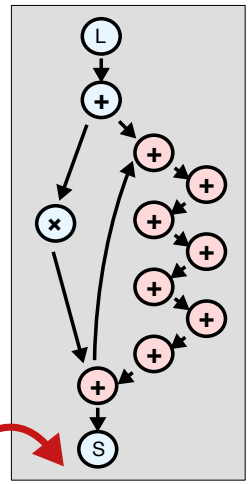
Dataflow Graph



Analytical Sweep



1.5x speedup
1.6x energy eff.

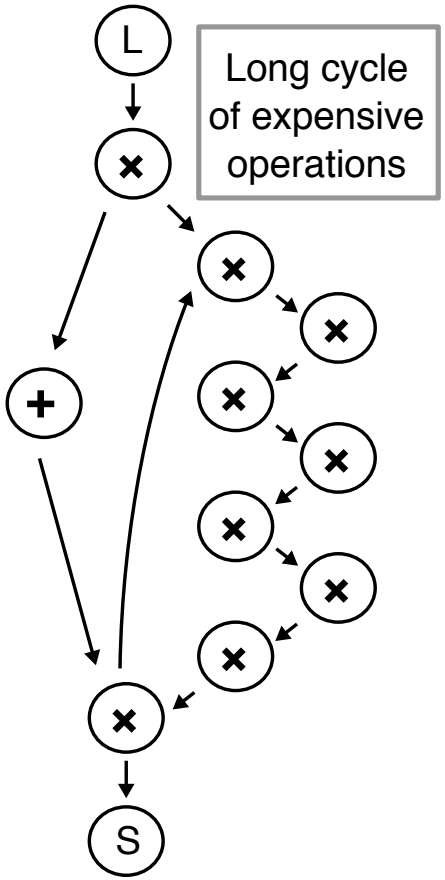


SRAMs rest (1.6x of mul power)

Sprint 1.23V
Nominal 1.0V
Rest 0.61V

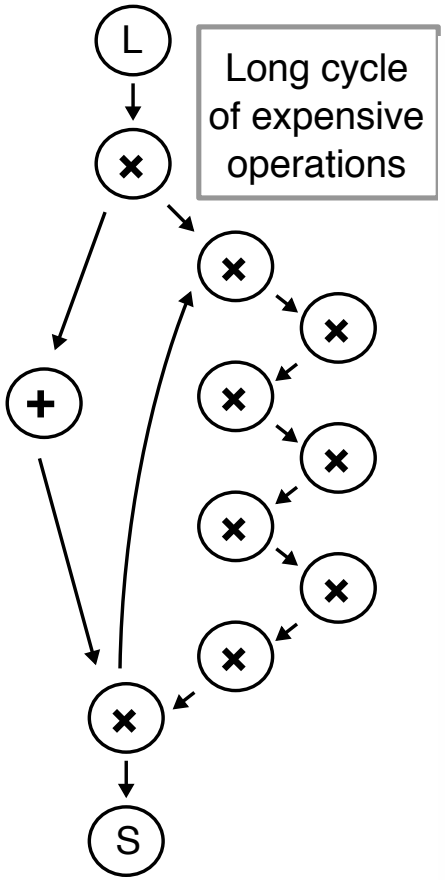
UE-CGRA Analytical Case Study #2

Dataflow Graph

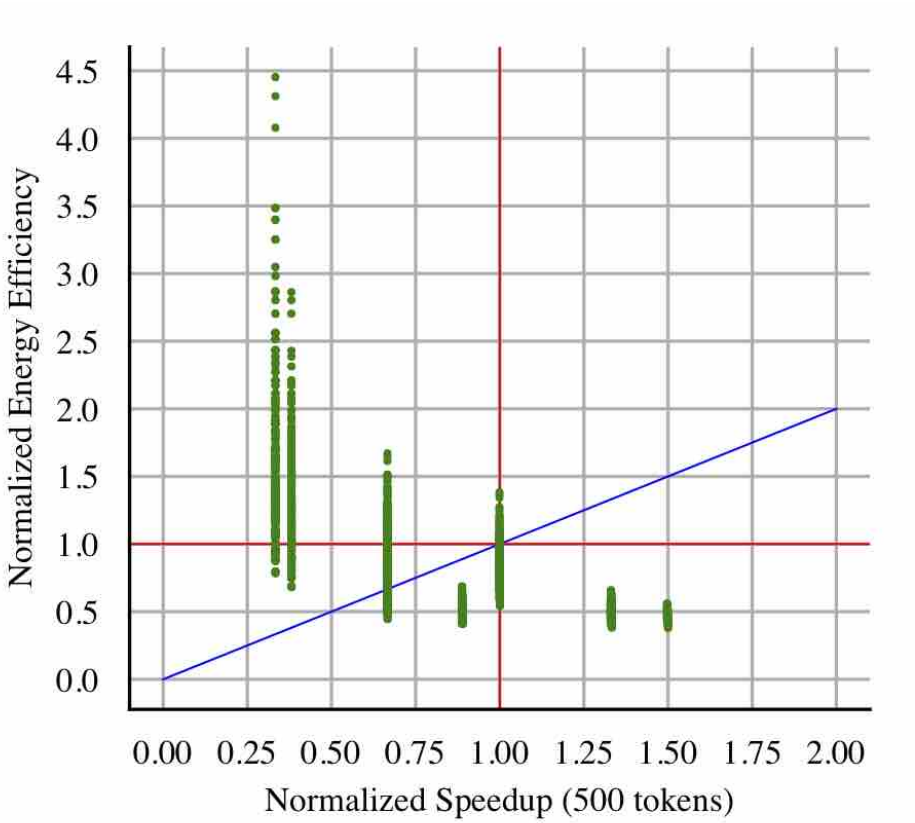


UE-CGRA Analytical Case Study #2

Dataflow Graph

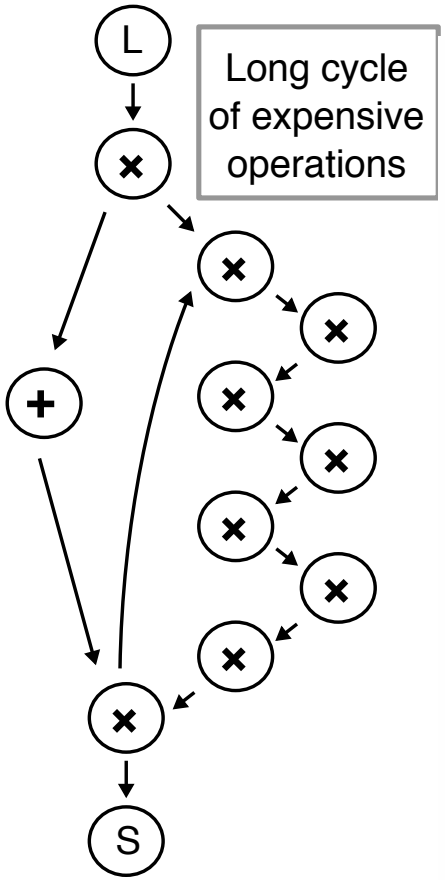


Analytical Sweep

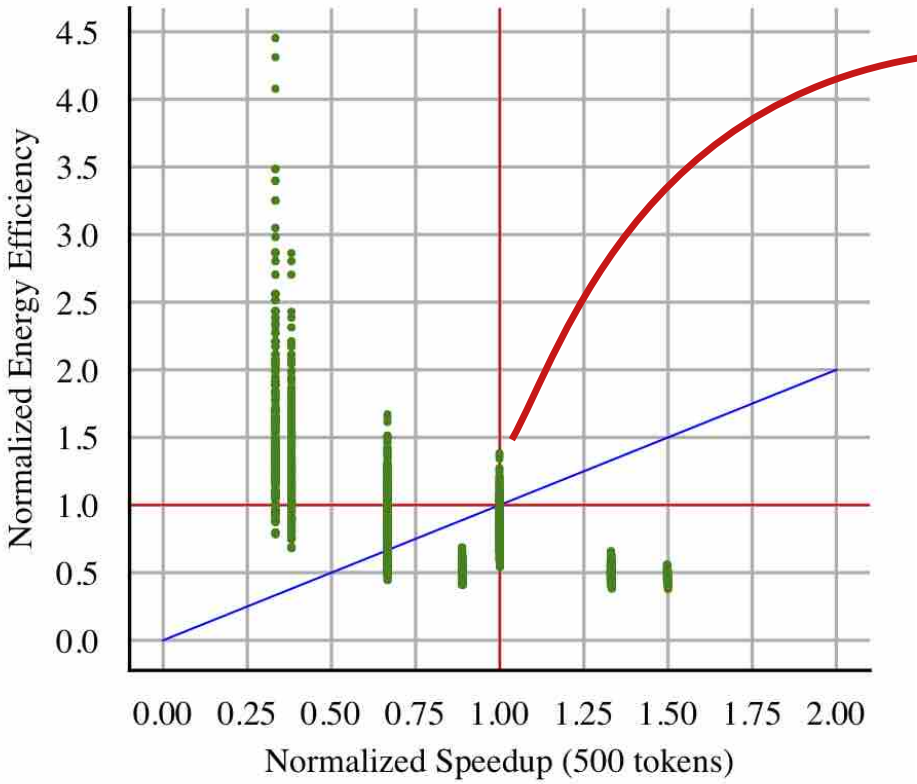


UE-CGRA Analytical Case Study #2

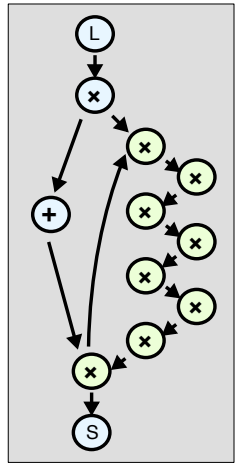
Dataflow Graph



Analytical Sweep



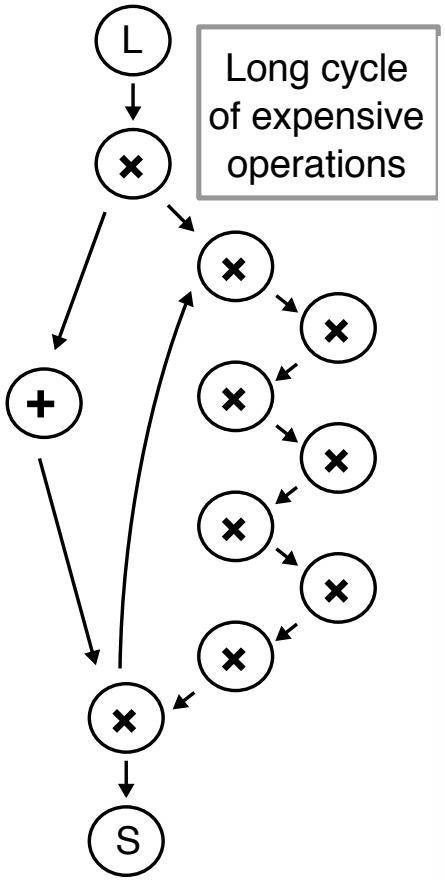
1.0x speedup
1.4x energy eff.



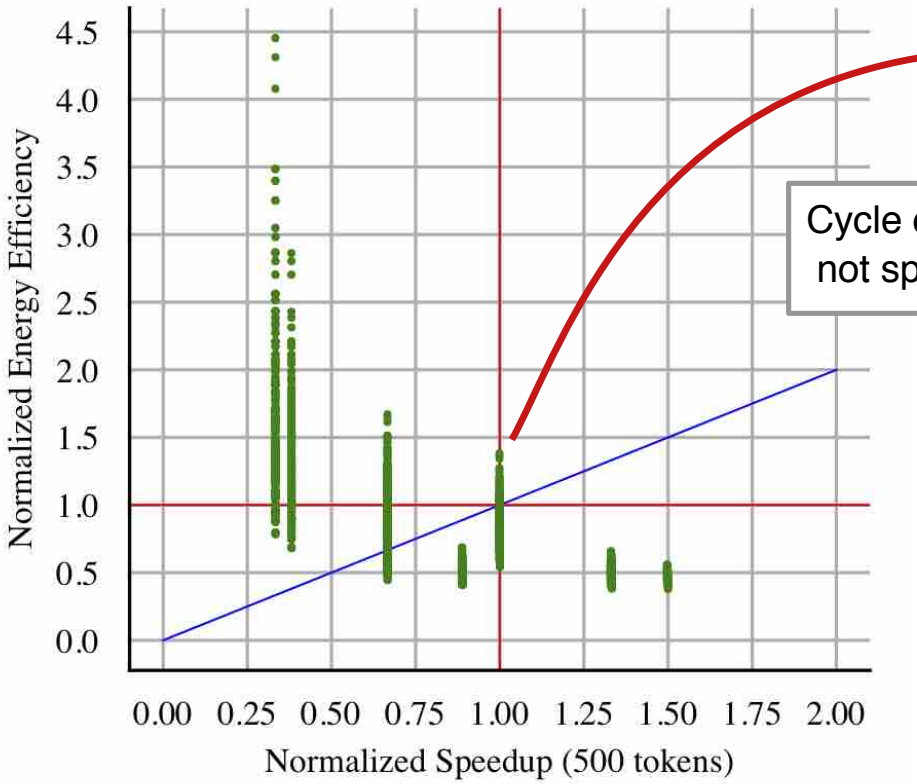
Sprint 1.23V
Nominal 1.0V
Rest 0.61V

UE-CGRA Analytical Case Study #2

Dataflow Graph

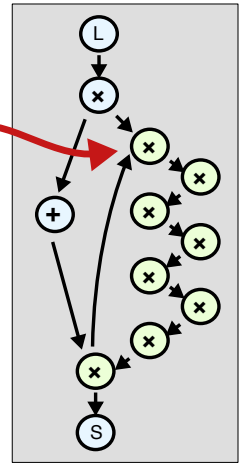


Analytical Sweep



1.0x speedup
1.4x energy eff.

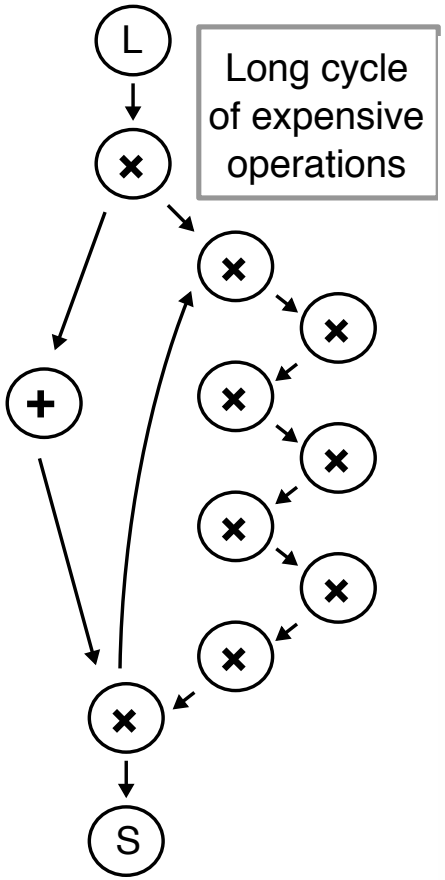
Cycle does not sprint!



Sprint 1.23V
Nominal 1.0V
Rest 0.61V

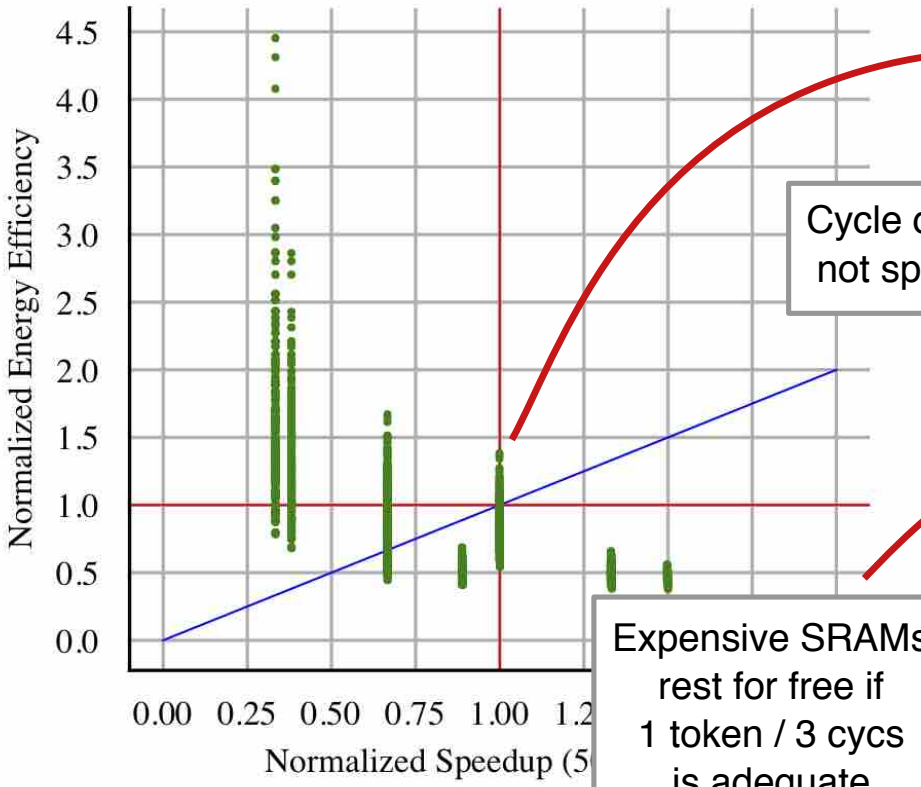
UE-CGRA Analytical Case Study #2

Dataflow Graph



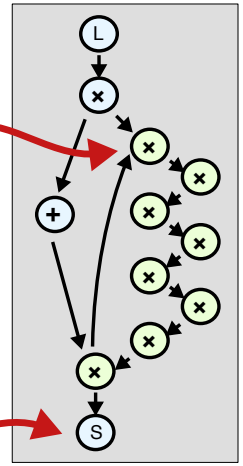
Long cycle of expensive operations

Analytical Sweep



1.0x speedup
1.4x energy eff.

Cycle does not sprint!



Expensive SRAMs rest for free if 1 token / 3 cycs is adequate

Sprint 1.23V
Nominal 1.0V
Rest 0.61V

UE-CGRA Analytical Summary

We can summarize our high-level intuition for UE-CGRAs

UE-CGRA Analytical Summary

We can summarize our high-level intuition for UE-CGRAs

- SRAMs become natural candidates for resting in the presence of long critical paths
  - Literature implies that long critical paths are common in CGRAs (e.g., initiation intervals of 3–20)
  - [LCTES'09], [CASES'06], [IPCDT'03]

UE-CGRA Analytical Summary

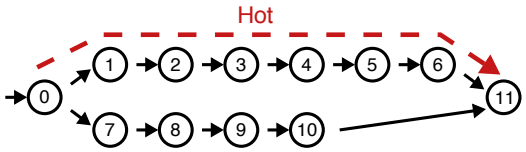
We can summarize our high-level intuition for UE-CGRAs

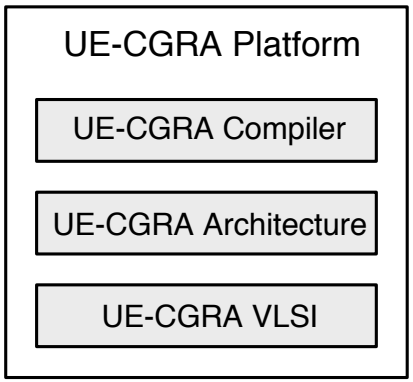
- SRAMs become natural candidates for resting in the presence of long critical paths
  - Literature implies that long critical paths are common in CGRAs (e.g., initiation intervals of 3–20)
  - [LCTES'09], [CASES'06], [IPCDT'03]
- Long dependent chains of *expensive* operators will be unlikely to allow any performance benefit, but still enable energy efficient resting

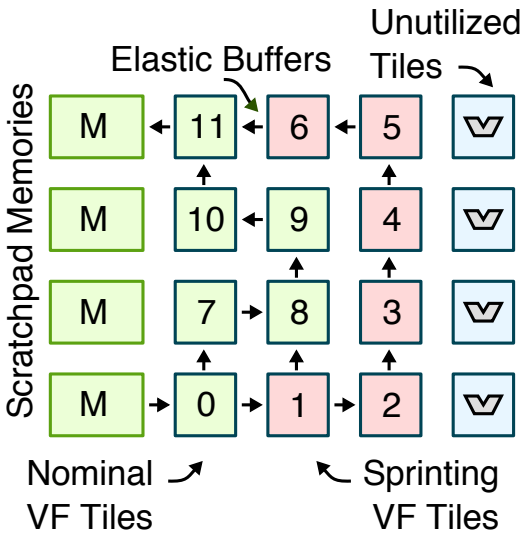
UE-CGRA Analytical Summary

We can summarize our high-level intuition for UE-CGRAs

- SRAMs become natural candidates for resting in the presence of long critical paths
  - Literature implies that long critical paths are common in CGRAs (e.g., initiation intervals of 3–20)
  - [LCTES'09], [CASES'06], [IPCDT'03]
- Long dependent chains of *expensive* operators will be unlikely to allow any performance benefit, but still enable energy efficient resting
- Long dependent chains of *cheap* operators (whether from cycles or from series-parallel joins) are excellent candidates for performance improvement with minimal impact to energy



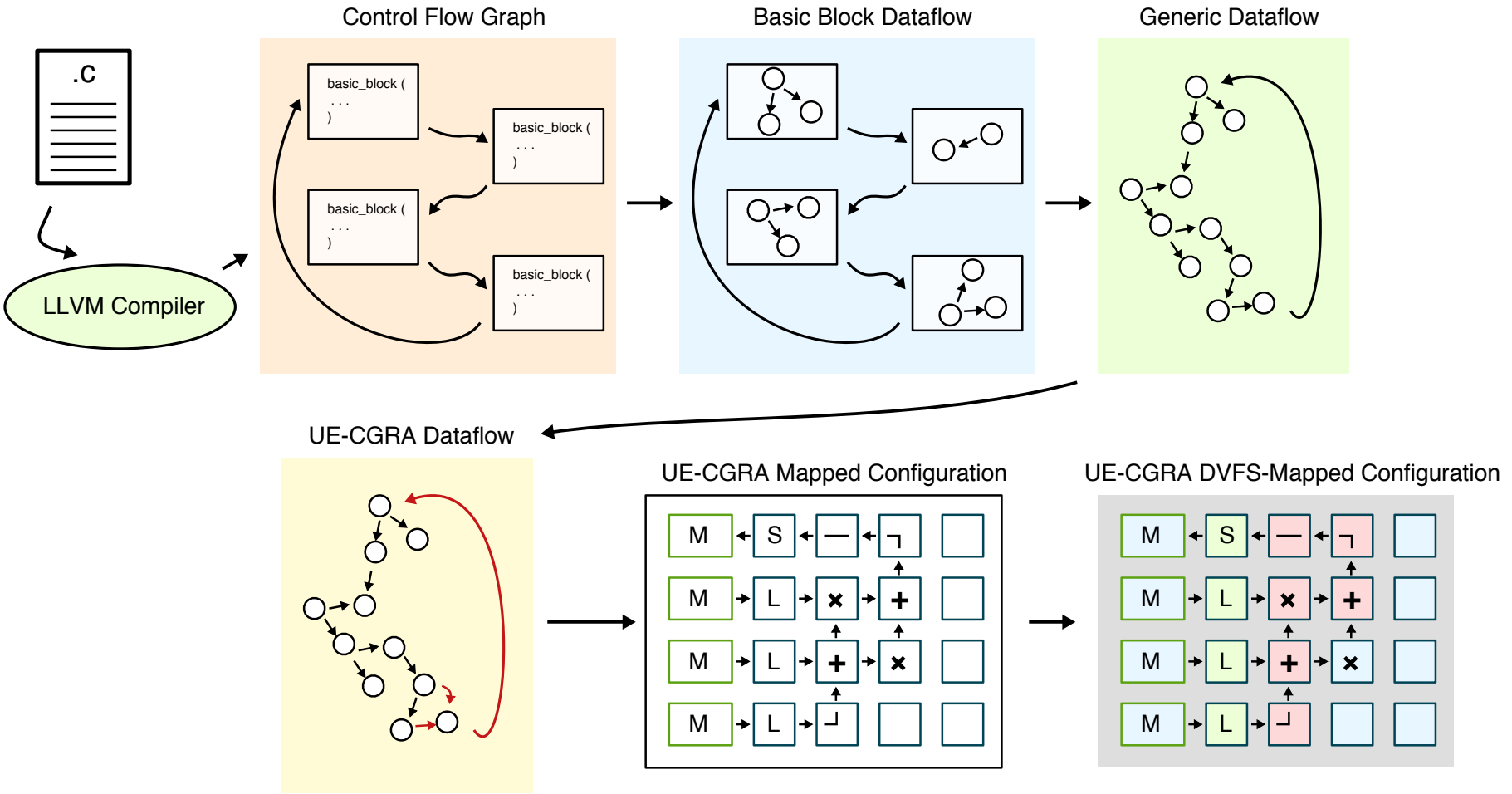




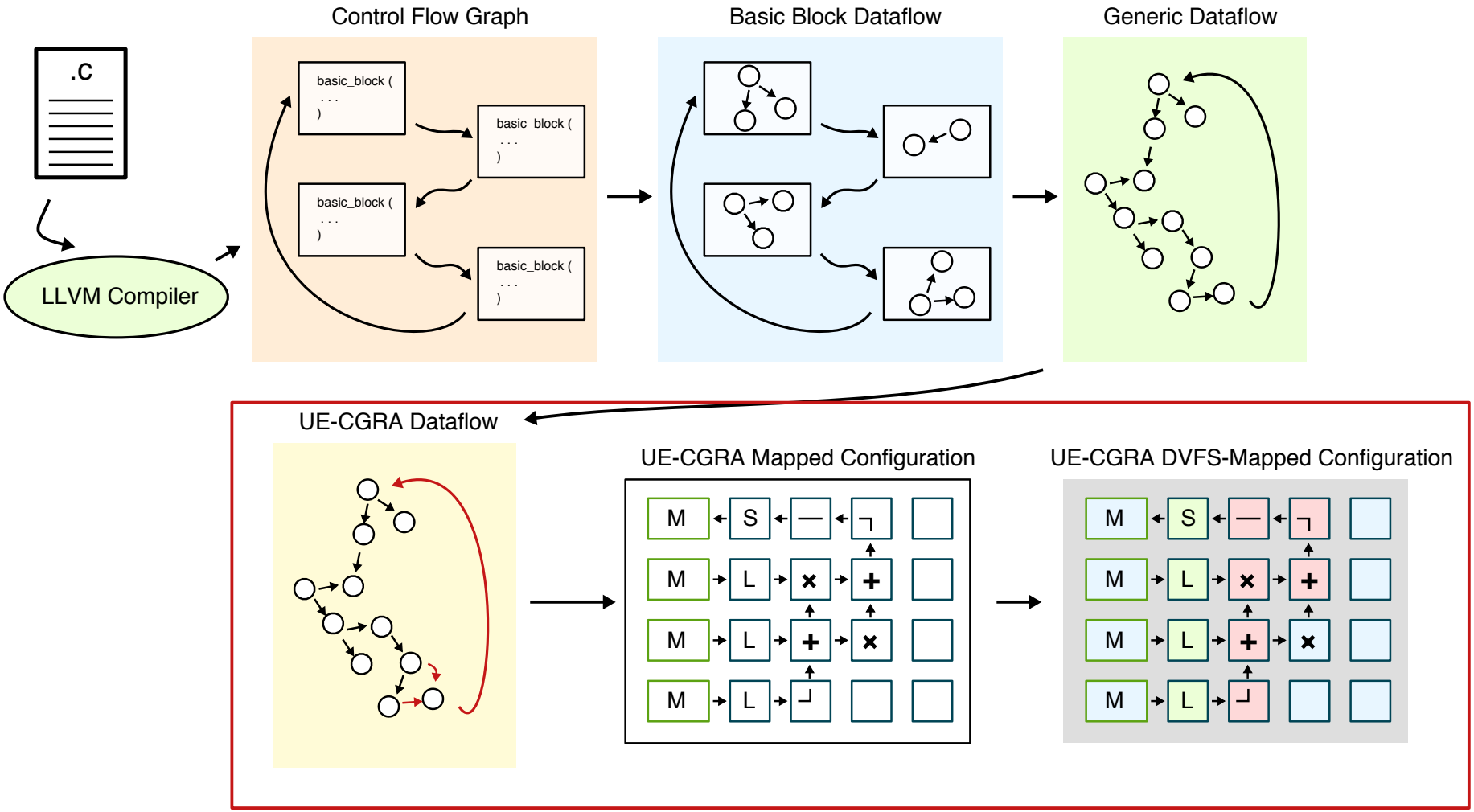
Exploiting Fine-Grain Asymmetry in Ultra-Elastic CGRAs

- Analytical Modeling
- UE-CGRA Compiler and Architecture
- UE-CGRA VLSI
- Methodology
- Evaluation

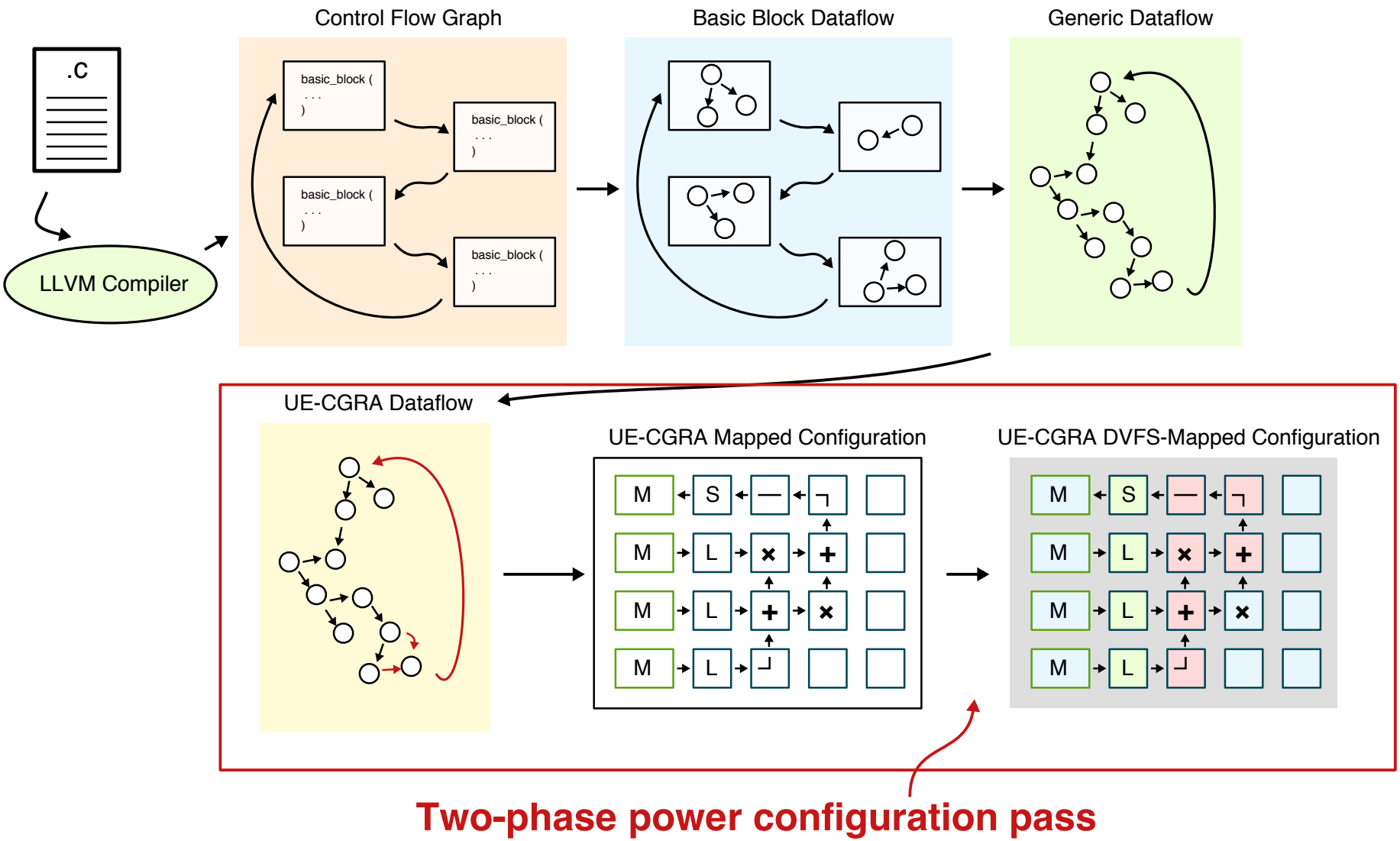
UE-CGRA Compiler Flow



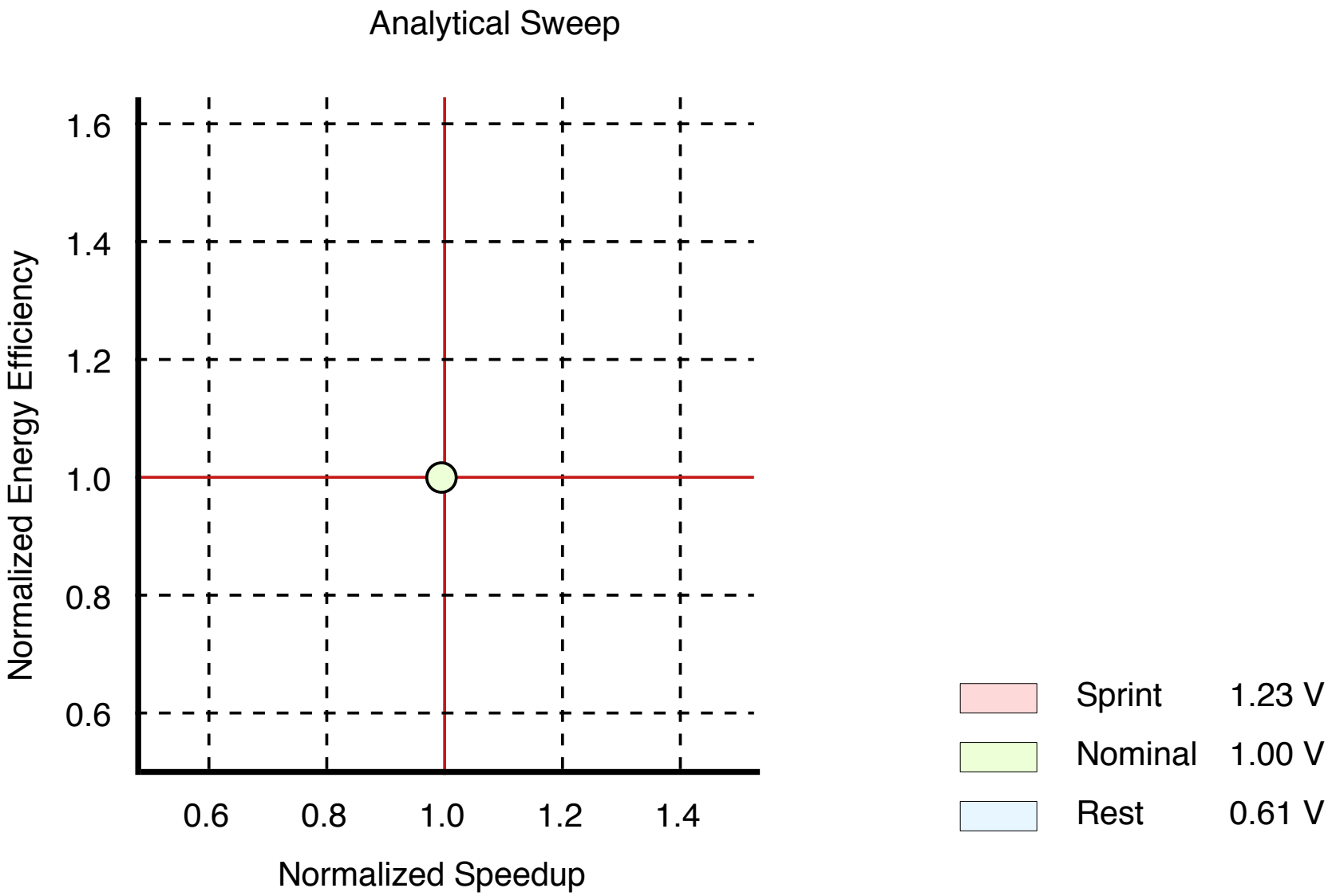
UE-CGRA Compiler Flow



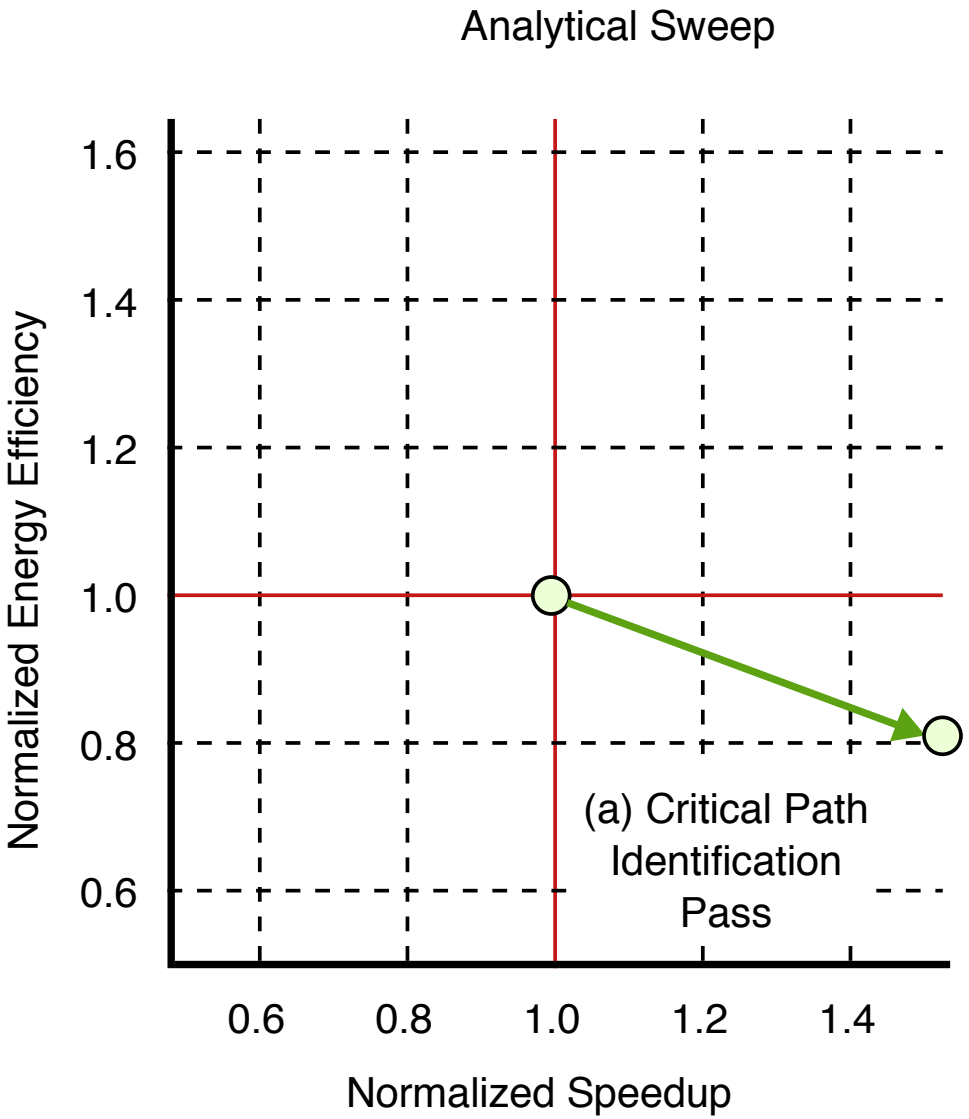
UE-CGRA Compiler Flow



Two-Phase Power Configuration Pass

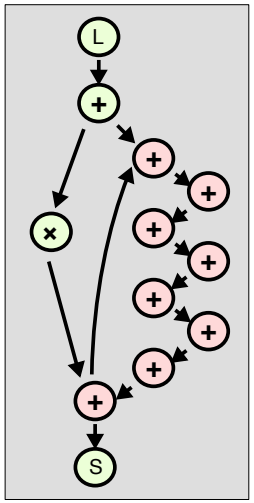


Two-Phase Power Configuration Pass



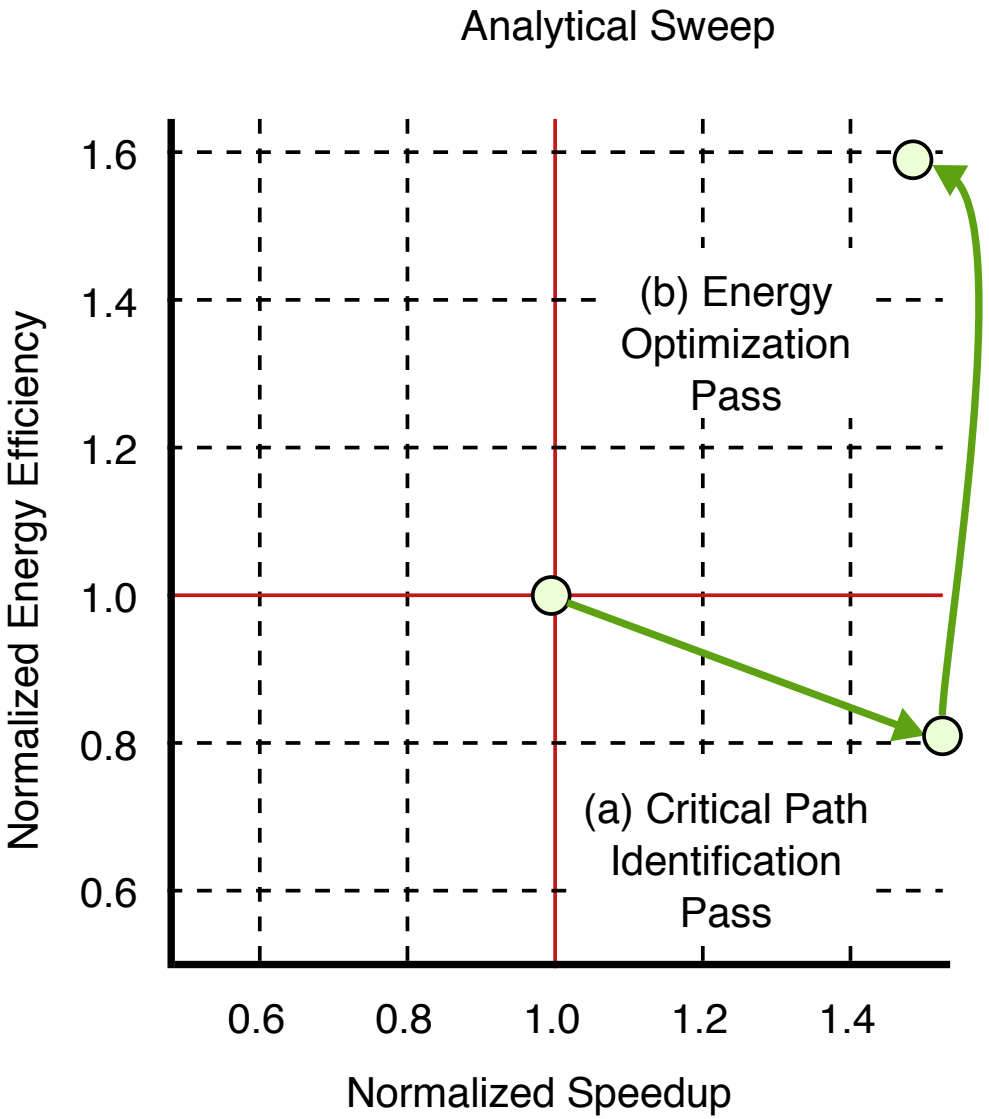
After (a)

1.5x speedup
0.8x energy eff.



	Sprint	1.23 V
	Nominal	1.00 V
	Rest	0.61 V

Two-Phase Power Configuration Pass

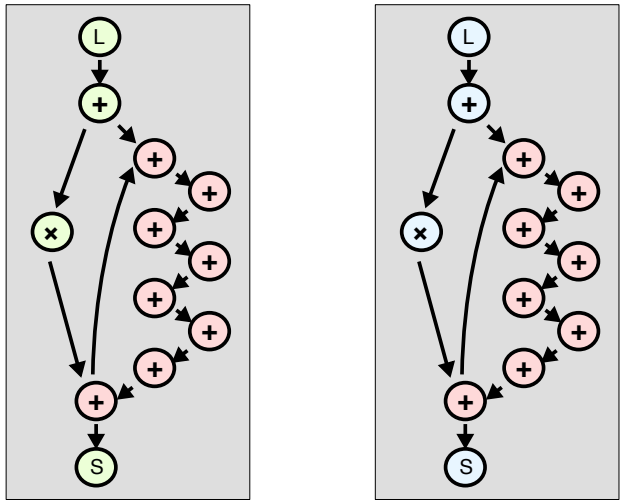


After (a)

1.5x speedup
0.8x energy eff.

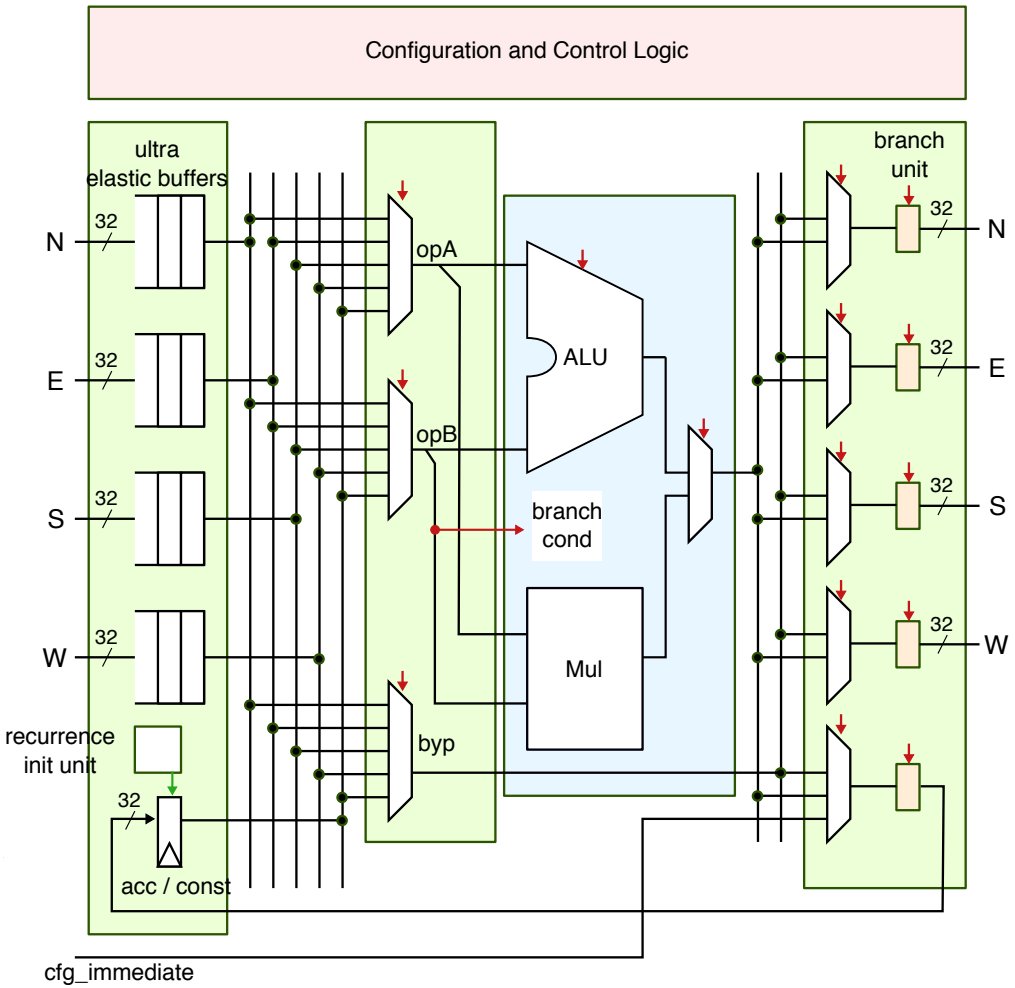
After (b)

1.5x speedup
1.6x energy eff.



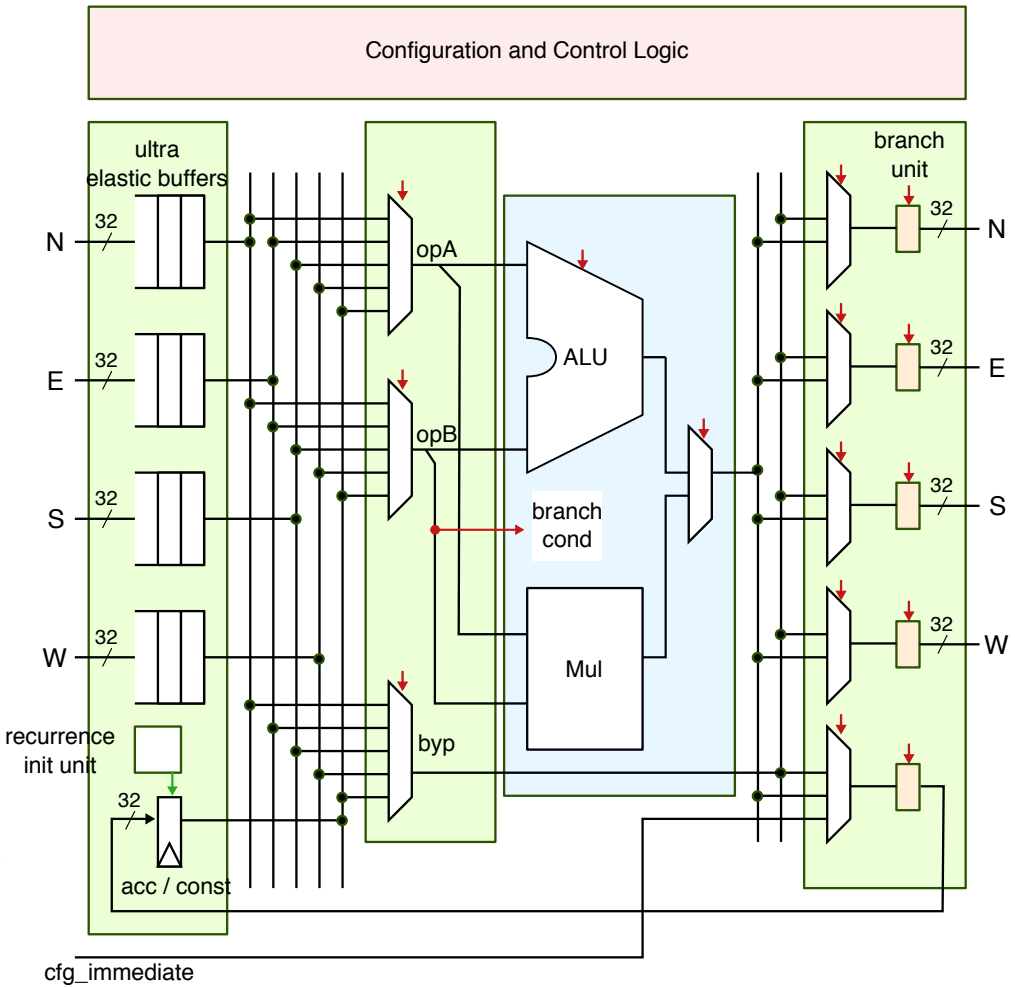
	Sprint	1.23 V
	Nominal	1.00 V
	Rest	0.61 V

UE-CGRA Tile



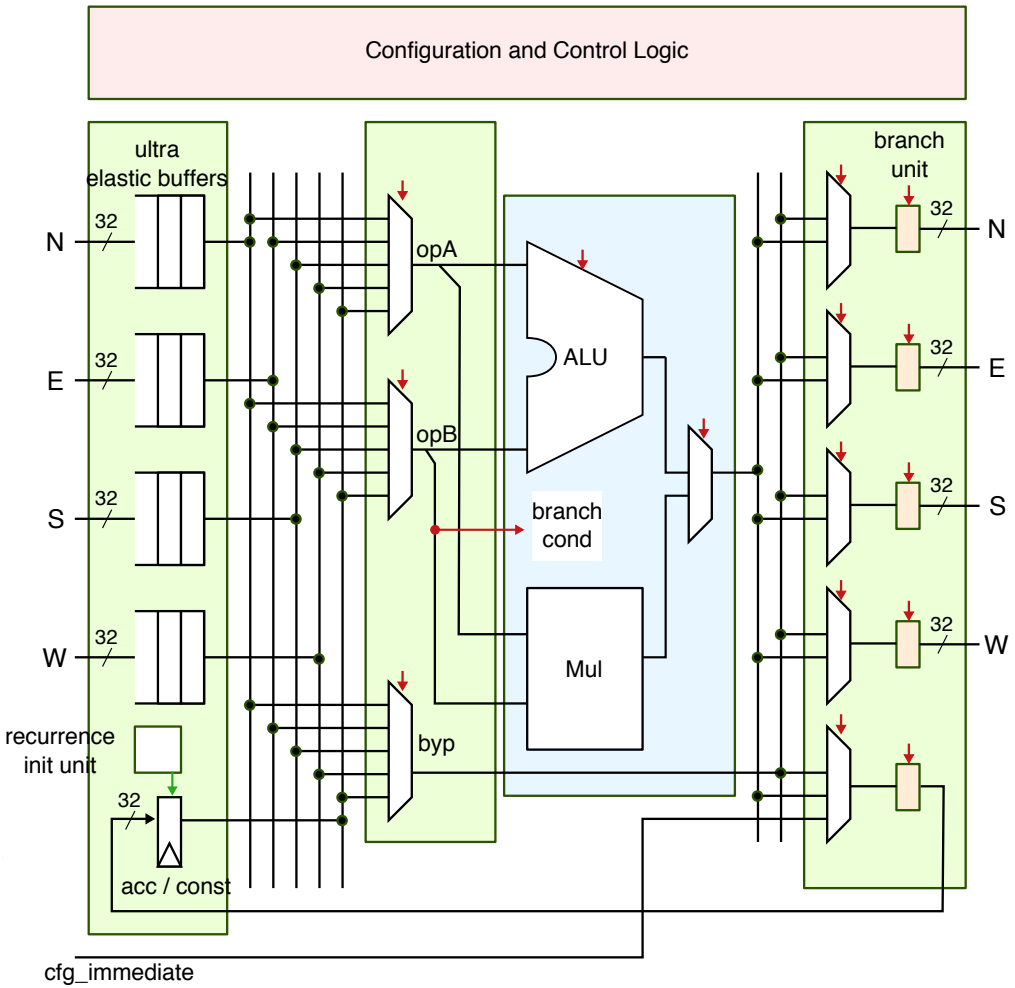
- Input queues
  - Two-entry normal queues for full throughput

UE-CGRA Tile



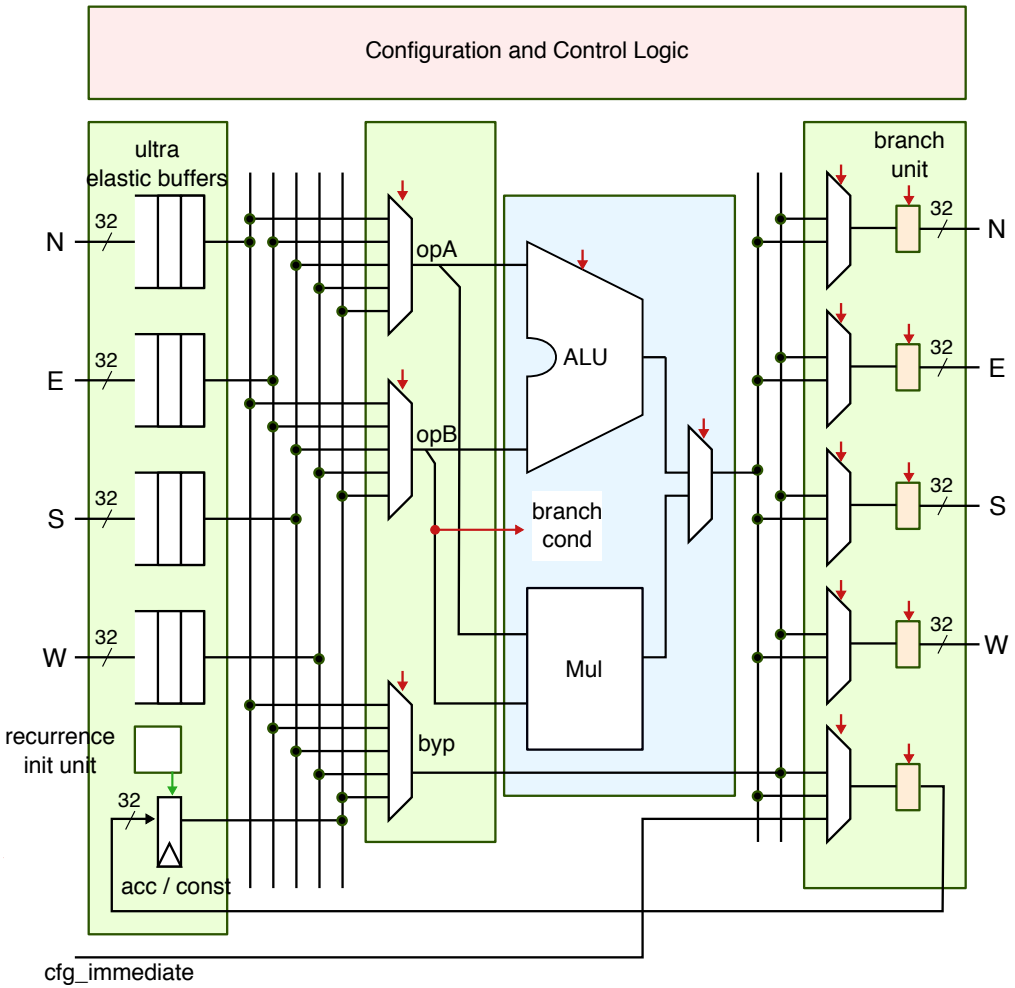
- Input queues
  - Two-entry normal queues for full throughput
- Multi-purpose register
  - Constants, accumulation

UE-CGRA Tile



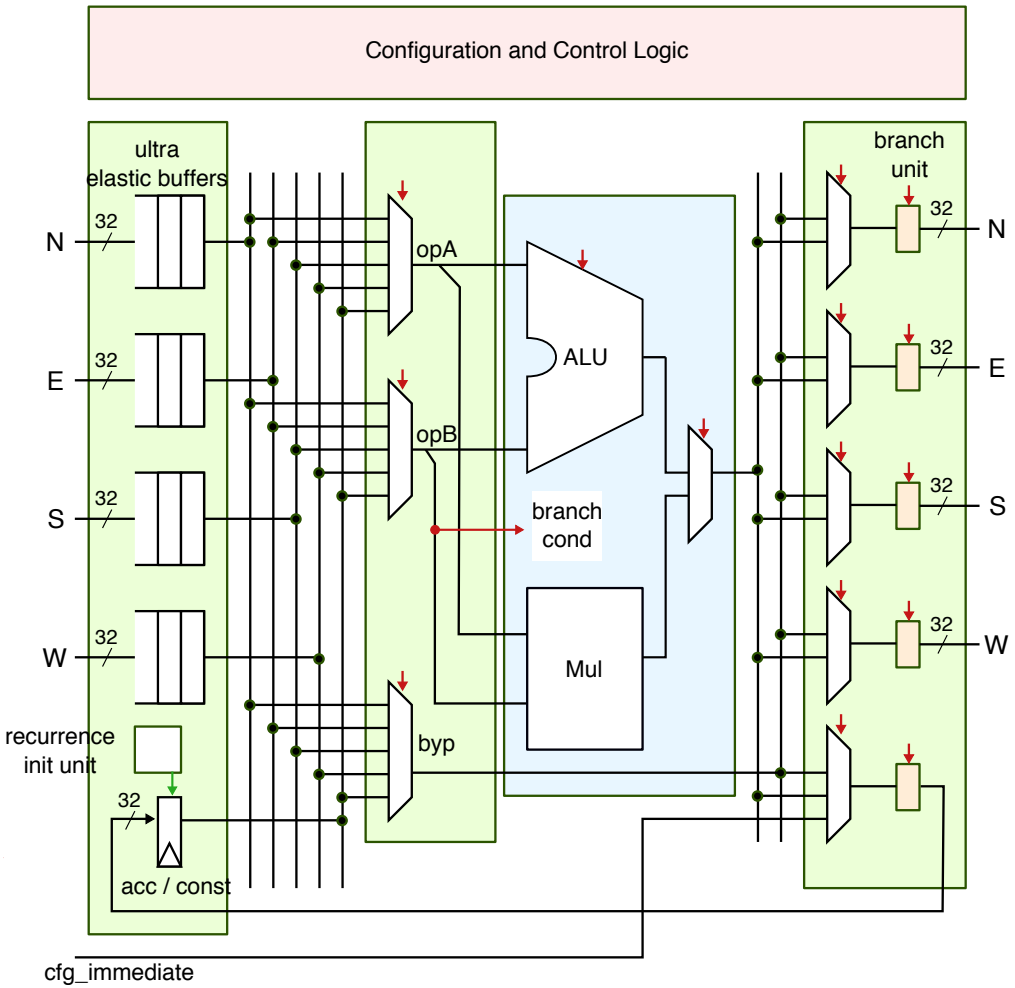
- Input queues
  - Two-entry normal queues for full throughput
- Multi-purpose register
  - Constants, accumulation
- Muxing
  - Routing and bypassing

UE-CGRA Tile



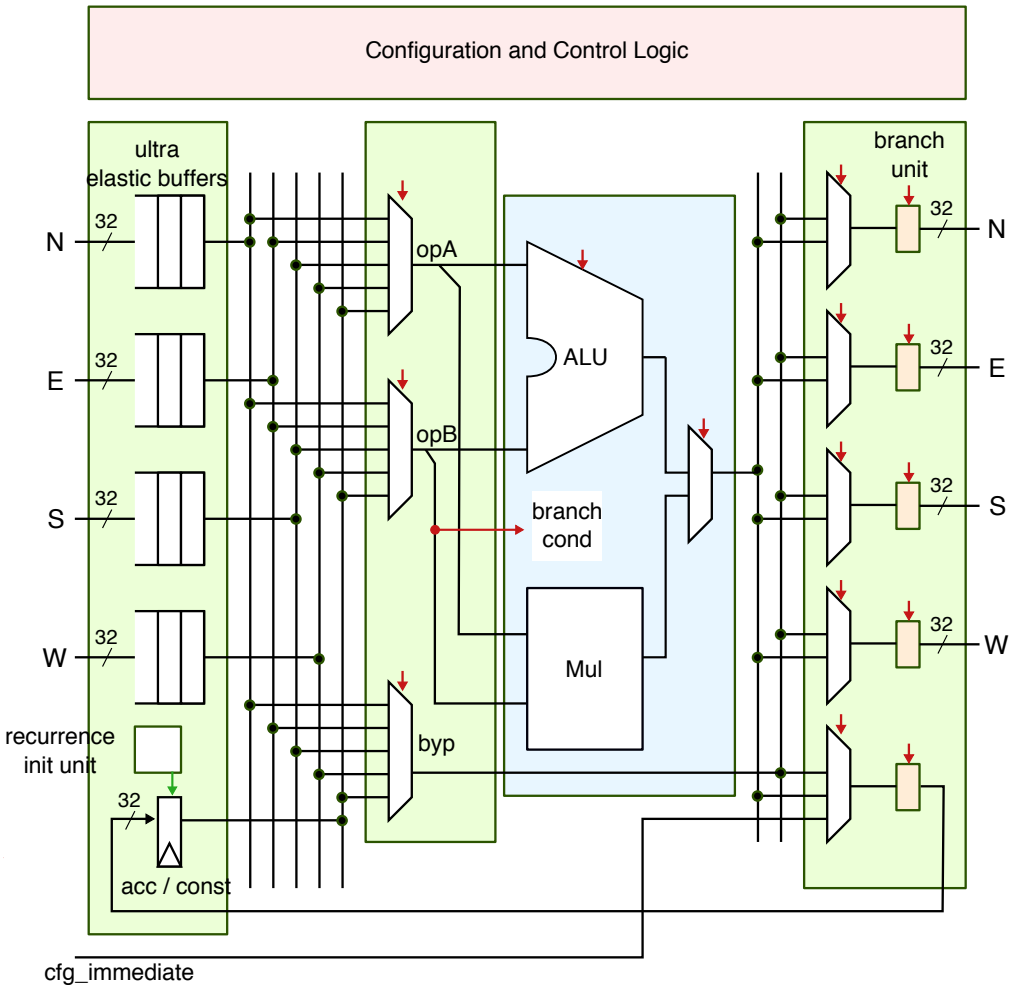
- Input queues
  - Two-entry normal queues for full throughput
- Multi-purpose register
  - Constants, accumulation
- Muxing
  - Routing and bypassing
- Operator
  - Supports 19 operations

UE-CGRA Tile



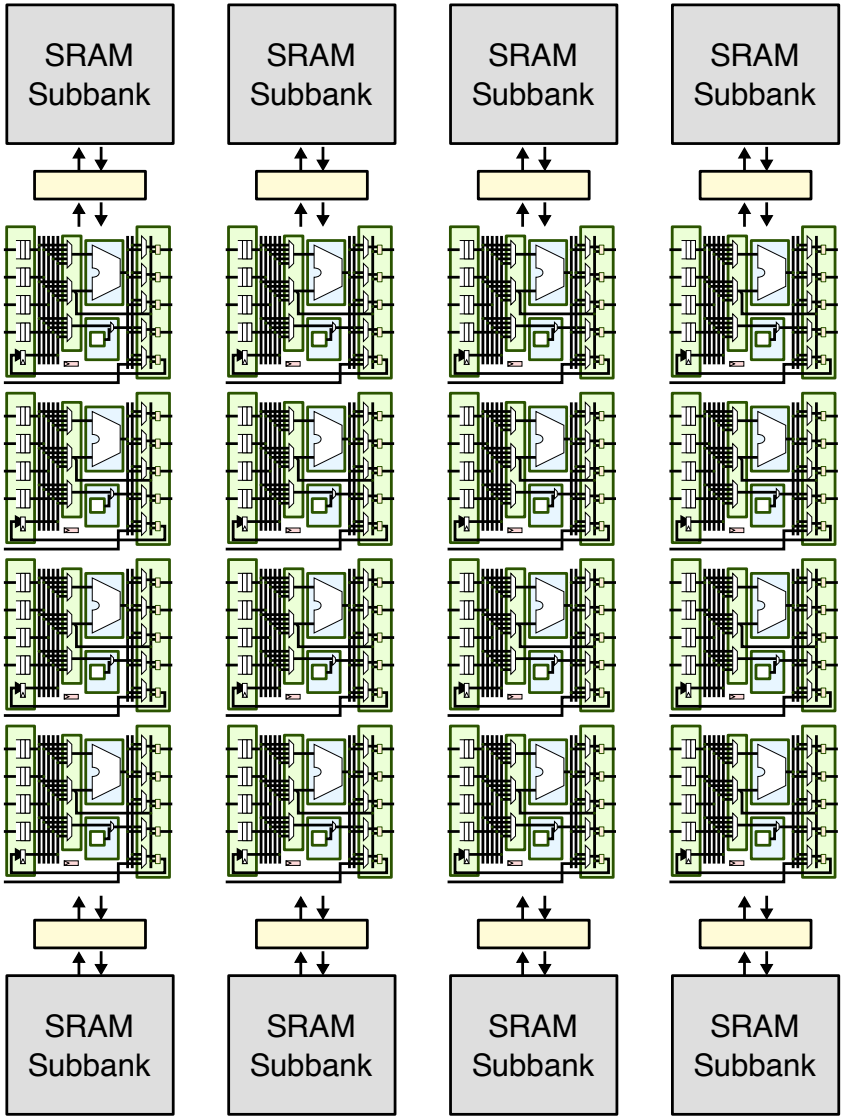
- Input queues
  - Two-entry normal queues for full throughput
- Multi-purpose register
  - Constants, accumulation
- Muxing
  - Routing and bypassing
- Operator
  - Supports 19 operations
- Phi support
  - Recurrence edges

UE-CGRA Tile



- Input queues
  - Two-entry normal queues for full throughput
- Multi-purpose register
  - Constants, accumulation
- Muxing
  - Routing and bypassing
- Operator
  - Supports 19 operations
- Phi support
  - Recurrence edges
- Branching support
  - Control flow is data flow

UE-CGRA Architecture

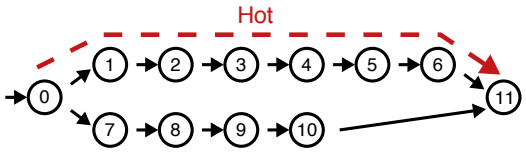


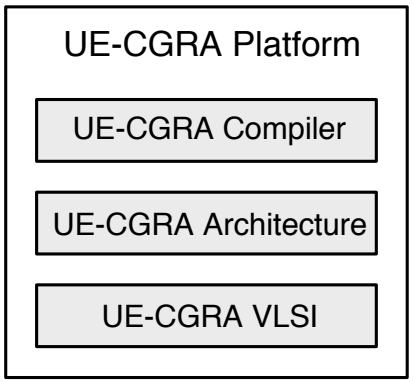
UE-CGRA composed of

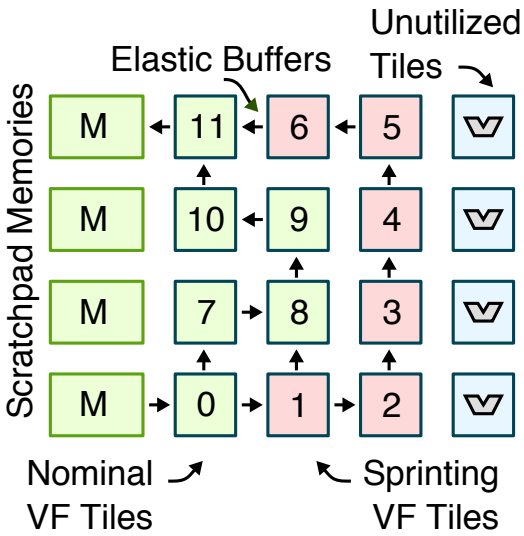
- Tiles with elastic interfaces
- SRAM subbanks
- Elastic memory adapters (similar to Huang et al. FPGA 2013)

Systolic configuration phase reuses the data network

UE-CGRA VLSI circuitry is not shown here







Exploiting Fine-Grain Asymmetry in Ultra-Elastic CGRAs

- Analytical Modeling
- UE-CGRA Compiler and Architecture
- UE-CGRA VLSI
- Methodology
- Evaluation

UE-CGRA VLSI – Strict Requirements

- DVFS Levels – At least three to improve both performance and energy

UE-CGRA VLSI – Strict Requirements

- DVFS Levels – At least three to improve both performance and energy
- Low Synchronization Latency – Must remain *very* low
  - Dataflow throughput is sensitive to critical-path latency
  - Queues are almost always empty (worst case for asynchronous FIFOs)

UE-CGRA VLSI – Strict Requirements

- DVFS Levels – At least three to improve both performance and energy
- Low Synchronization Latency – Must remain *very* low
  - Dataflow throughput is sensitive to critical-path latency
  - Queues are almost always empty (worst case for asynchronous FIFOs)
- Low Reconfiguration Overhead
  - Enable rapid reconfiguration for different kernels

UE-CGRA VLSI – Strict Requirements

- DVFS Levels – At least three to improve both performance and energy
- Low Synchronization Latency – Must remain *very* low
  - Dataflow throughput is sensitive to critical-path latency
  - Queues are almost always empty (worst case for asynchronous FIFOs)
- Low Reconfiguration Overhead
  - Enable rapid reconfiguration for different kernels
- Low-Leakage Clock-Domain Crossings
  - Level shifters are slow at 1-10 ns (TVLSI 2015)
  - 32b data bus leaking 30 uA – 1 pJ to exist (tile `mul` is 2.5 pJ)

UE-CGRA VLSI – Strict Requirements

- DVFS Levels – At least three to improve both performance and energy
- Low Synchronization Latency – Must remain *very* low
  - Dataflow throughput is sensitive to critical-path latency
  - Queues are almost always empty (worst case for asynchronous FIFOs)
- Low Reconfiguration Overhead
  - Enable rapid reconfiguration for different kernels
- Low-Leakage Clock-Domain Crossings
  - Level shifters are slow at 1-10 ns (TVLSI 2015)
  - 32b data bus leaking 30 uA – 1 pJ to exist (tile `mul` is 2.5 pJ)
- Area and Power Overheads – Low
  - PEs are small – small overheads are significant

UE-CGRA VLSI – Strict Requirements

- DVFS Levels – At least three to improve both performance and energy
- Low Synchronization Latency – Must remain *very* low
  - Dataflow throughput is sensitive to critical-path latency
  - Queues are almost always empty (worst case for asynchronous FIFOs)
- Low Reconfiguration Overhead
  - Enable rapid reconfiguration for different kernels
- Low-Leakage Clock-Domain Crossings
  - Level shifters are slow at 1-10 ns (TVLSI 2015)
  - 32b data bus leaking 30 uA – 1 pJ to exist (tile `mul` is 2.5 pJ)
- Area and Power Overheads – Low
  - PEs are small – small overheads are significant
- Verification Challenges
  - NREs rise exponentially
  - Asynchronous clock-domain crossings are hard to verify

UE-CGRA VLSI

Key features of VLSI support for ultra elasticity

UE-CGRA VLSI

Key features of VLSI support for ultra elasticity

- Configured Ratiochronous Clocking (adapted to elastic CGRAs) [ISLPED'10]
  - Compiler maps to synchronous quantized VF levels

UE-CGRA VLSI

Key features of VLSI support for ultra elasticity

- Configured Ratiochronous Clocking (adapted to elastic CGRAs) [ISLPED'10]
  - Compiler maps to synchronous quantized VF levels
  - Previous work on unsafe edges [ICCD'95] augmented for elasticity

UE-CGRA VLSI

Key features of VLSI support for ultra elasticity

- Configured Ratiochronous Clocking (adapted to elastic CGRAs) [ISLPED'10]
  - Compiler maps to synchronous quantized VF levels
  - Previous work on unsafe edges [ICCD'95] augmented for elasticity
  - System-level ratiochronous design (advance from [ASYNC'12] in isolation)

UE-CGRA VLSI

Key features of VLSI support for ultra elasticity

- Configured Ratiochronous Clocking (adapted to elastic CGRAs) [ISLPED'10]
  - Compiler maps to synchronous quantized VF levels
  - Previous work on unsafe edges [ICCD'95] augmented for elasticity
  - System-level ratiochronous design (advance from [ASYNC'12] in isolation)
  - Coordinated (glitchless) clock switching and alignment

UE-CGRA VLSI

Key features of VLSI support for ultra elasticity

- Configured Ratiochronous Clocking (adapted to elastic CGRAs) [ISLPED'10]
  - Compiler maps to synchronous quantized VF levels
  - Previous work on unsafe edges [ICCD'95] augmented for elasticity
  - System-level ratiochronous design (advance from [ASYNC'12] in isolation)
  - Coordinated (glitchless) clock switching and alignment
- Multi-Rail Supply Voltages (traditional approach w/ PMOS header switches)

UE-CGRA VLSI

Key features of VLSI support for ultra elasticity

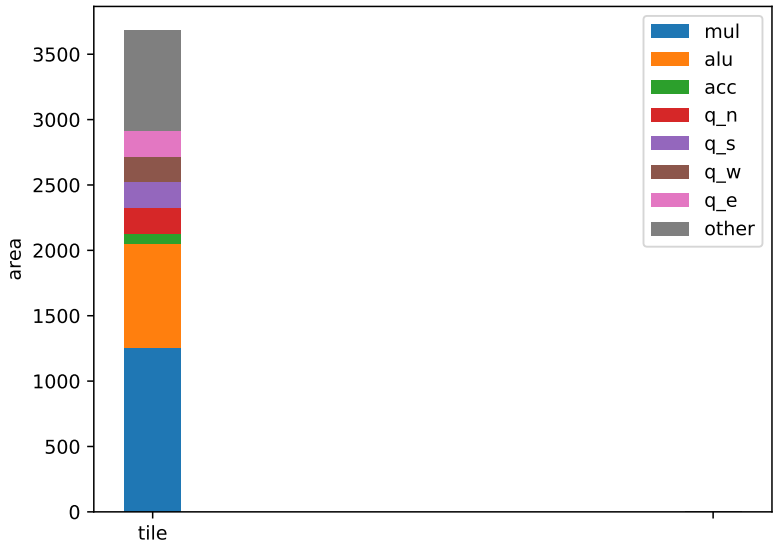
- Configured Ratiochronous Clocking (adapted to elastic CGRAs) [ISLPED'10]
  - Compiler maps to synchronous quantized VF levels
  - Previous work on unsafe edges [ICCD'95] augmented for elasticity
  - System-level ratiochronous design (advance from [ASYNC'12] in isolation)
  - Coordinated (glitchless) clock switching and alignment
- Multi-Rail Supply Voltages (traditional approach w/ PMOS header switches)
- Ratiochronous Crossings and Static Timing Analysis Flow
  - Simple bisynchronous fifos are sufficient
  - UE-CGRA VLSI tool generates ratiochronous timing constraints
  - No level shifters are used

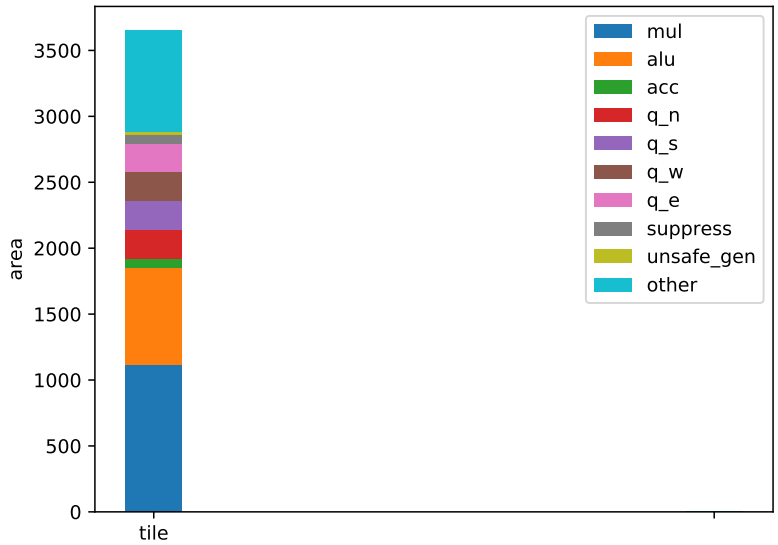
UE-CGRA VLSI

Key features of VLSI support for ultra elasticity

- Configured Ratiochronous Clocking (adapted to elastic CGRAs) [ISLPED'10]
  - Compiler maps to synchronous quantized VF levels
  - Previous work on unsafe edges [ICCD'95] augmented for elasticity
  - System-level ratiochronous design (advance from [ASYNC'12] in isolation)
  - Coordinated (glitchless) clock switching and alignment
- Multi-Rail Supply Voltages (traditional approach w/ PMOS header switches)
- Ratiochronous Crossings and Static Timing Analysis Flow
  - Simple bisynchronous fifos are sufficient
  - UE-CGRA VLSI tool generates ratiochronous timing constraints
  - No level shifters are used
- Compiler optimizations
  - Functionality and leakage-aware mapping (0.61V never talks to 1.23V)
  - Energy-aware optimization pass to rest expensive operators

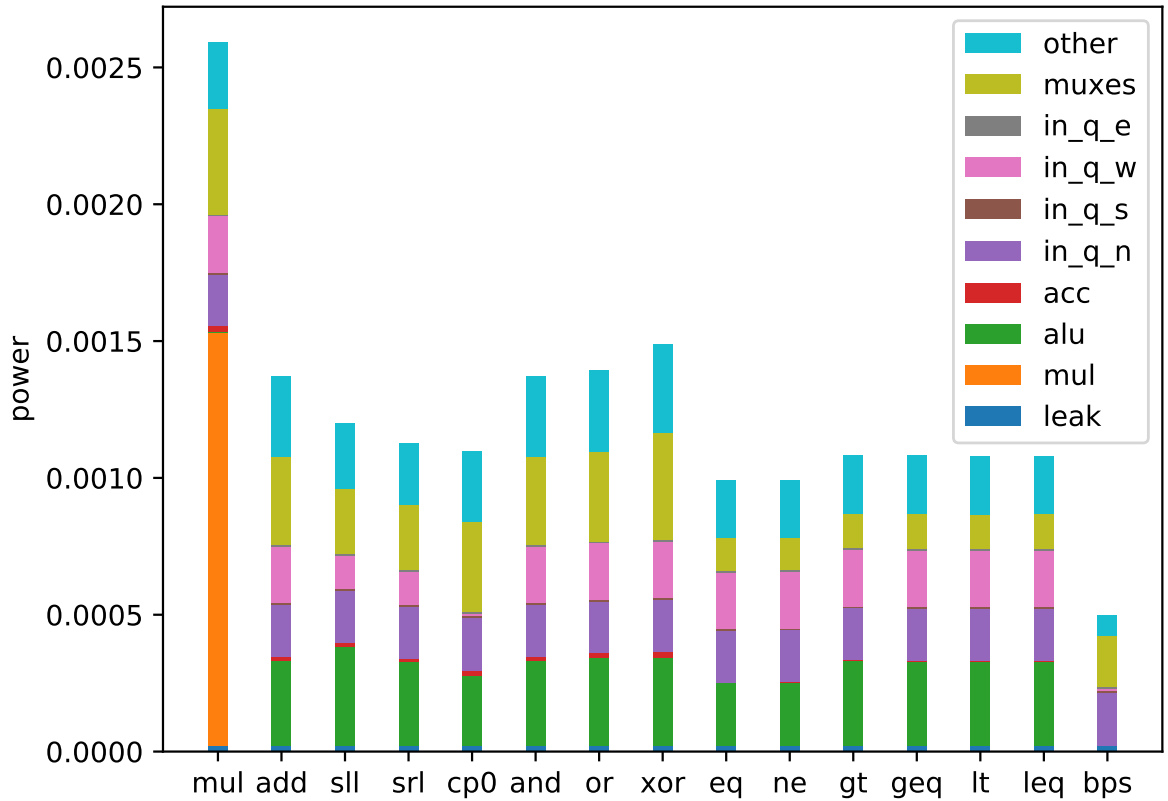
UE-CGRA Methodology – Area





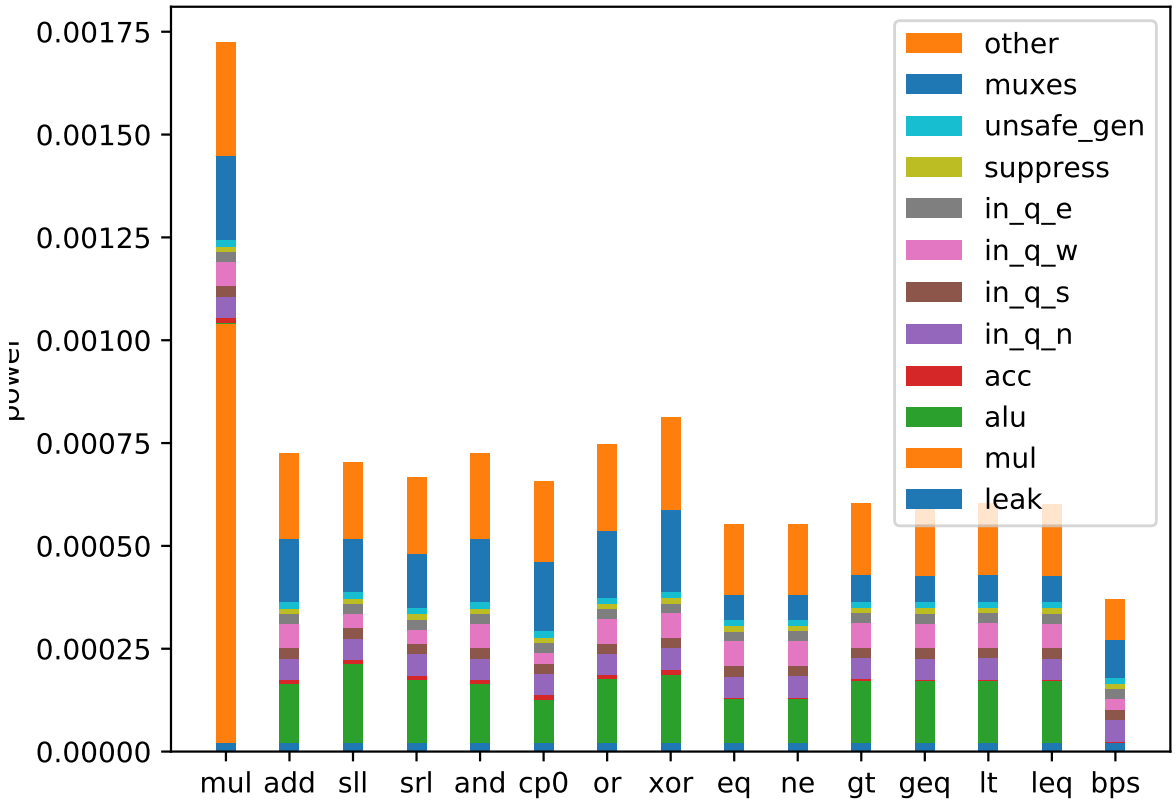
Preliminary area overhead estimates for ultra-elastic circuitry is small, but does not account for multi-rail supply voltage islands (power headers, margining between domains, more congested metal routing)

UE-CGRA Methodology – Elastic Tile Power



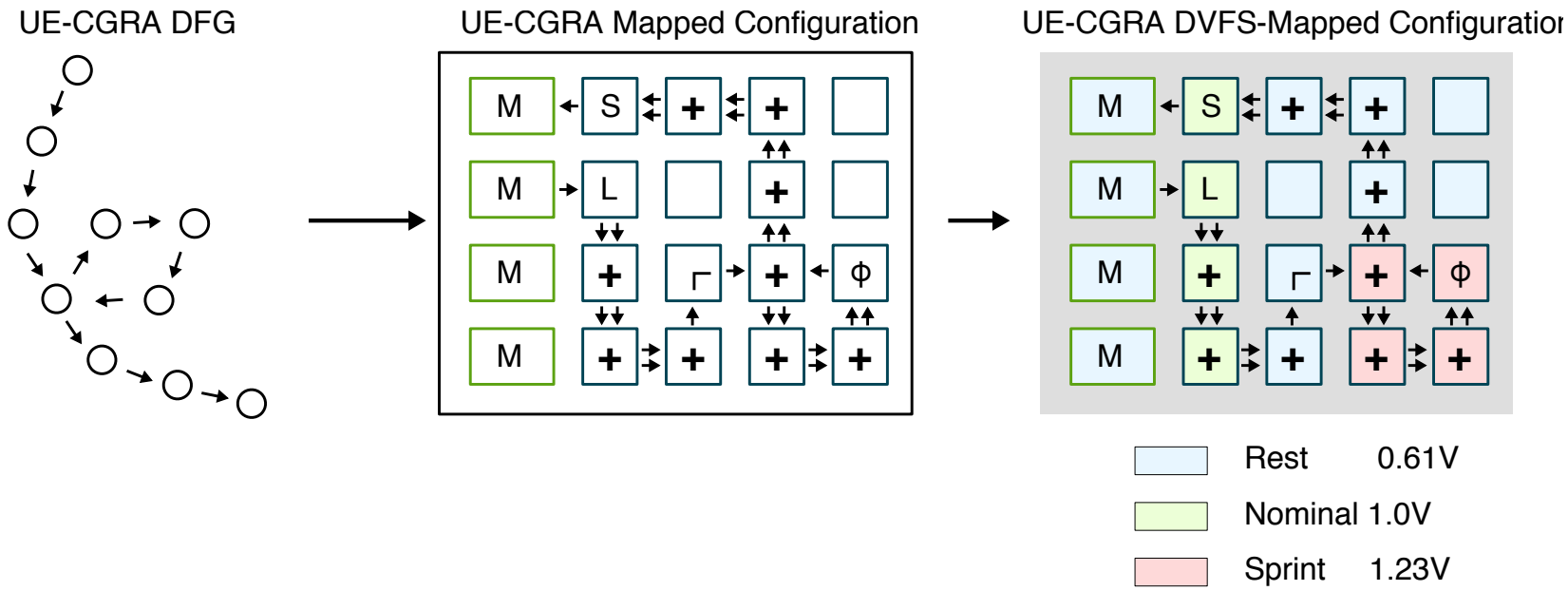
Preliminary power estimates show only small differences in tile-level power breakdowns with ultra-elastic circuitry, but does not account for clock tree power, leakage at crossings, power header overheads.

UE-CGRA Methodology – Ultra-Elastic Tile Power



Preliminary power estimates show only small differences in tile-level power breakdowns with ultra-elastic circuitry, but does not account for clock tree power, leakage at crossings, power header overheads.

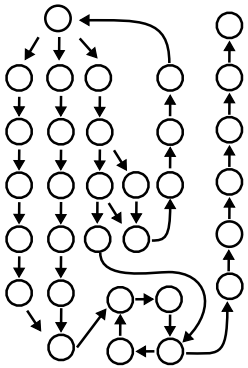
UE-CGRA Results



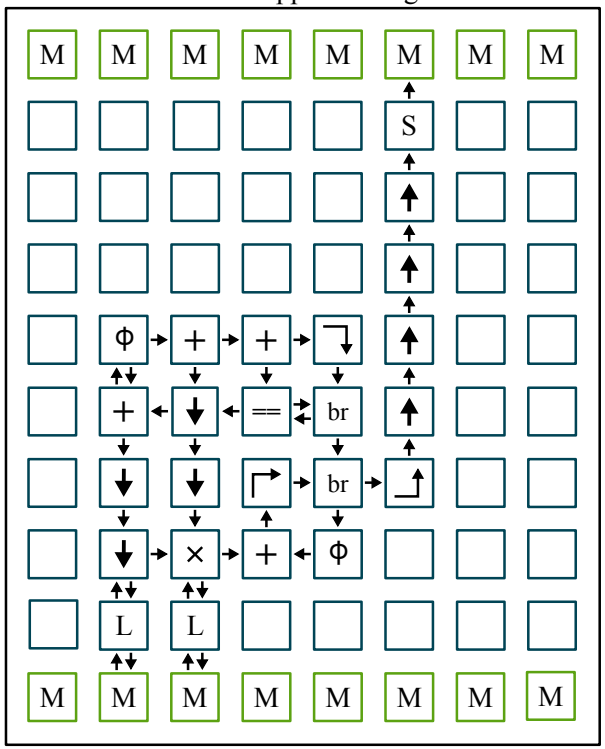
An example series-cyclic microbenchmark dataflow graph

UE-CGRA Results

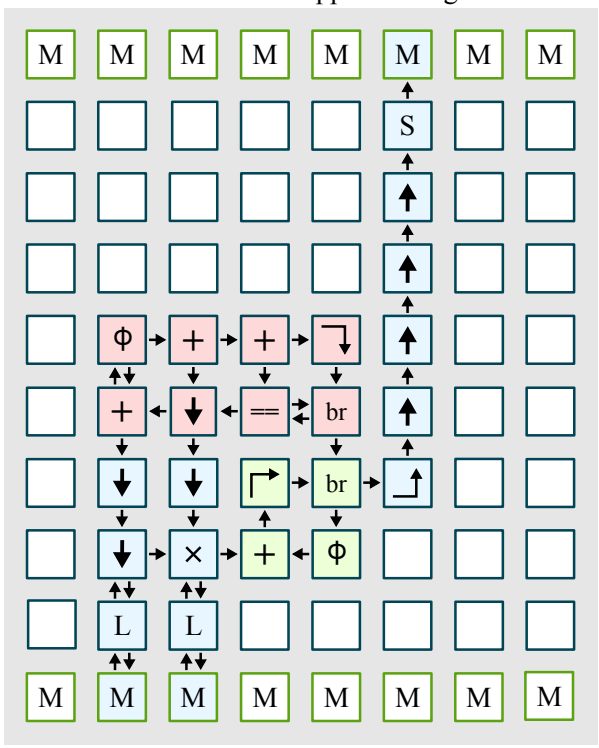
UE-CGRA DFG



UE-CGRA Mapped Configuration



UE-CGRA DVFS-Mapped Configuration



Rest 0.61V

Nominal 0.90V

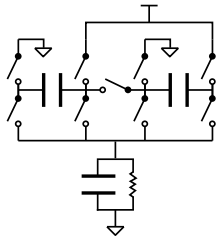
Sprint 1.23V

UE-CGRA Results

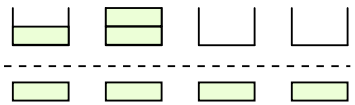
Benchmark	Iterations per Second	Energy
Series-Cyclic (base)	124018K	29.91 pJ per token
Series-Cyclic (fast)	184729K	30.56 pJ per token
Improvement	1.49 ×	0.98 ×
Series-Parallel (base)	242522K	87.21 pJ per token
Series-Parallel (fast)	275988K	93.42 pJ per token
Improvement	1.14 ×	0.93 ×
FIR (base)	62176K	43.58 pJ per token
FIR (fast)	92735K	45.81 pJ per token
Speedup	1.49 ×	0.95 ×

Preliminary results for both performance and energy efficiency for an 8×8 array

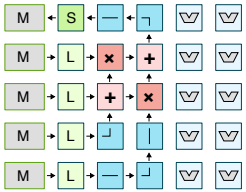
Software, Architecture, and VLSI Co-Design for Exploiting Fine-Grain Asymmetry



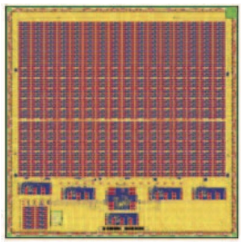
Exploiting Fine-Grain Asymmetry with
On-Chip Voltage Regulation - **MICRO'14**, IEEE TCAS I'18



Exploiting Fine-Grain Asymmetry in
Task-Based Parallel Runtimes - **ISCA'16**, MICRO'17, RISCV'18



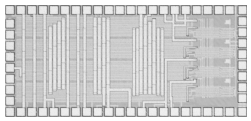
Exploiting Fine-Grain Asymmetry in
Coarse-Grain Reconfigurable Arrays - **Unpublished**



Exploiting Fine-Grain Asymmetry with
Silicon Prototyping - **IEEE MICRO'18**, DAC'18, Hotchips'17

Conclusion

Silicon Prototyping

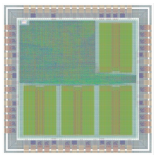


DCS (2014)

TSMC 65nm
1mm x 2.2mm
[TCAS'18, MICRO'14]

Research Purpose

Directly implement and characterize the voltage regulator designs supporting dynamic capacitance sharing

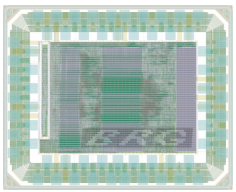


BRGTC1 (2016)

IBM 130nm
2mm x 2mm
[Poster at HOTCHIPS'16]

Research Purpose

Silicon-validate PyMTL-generated RTL

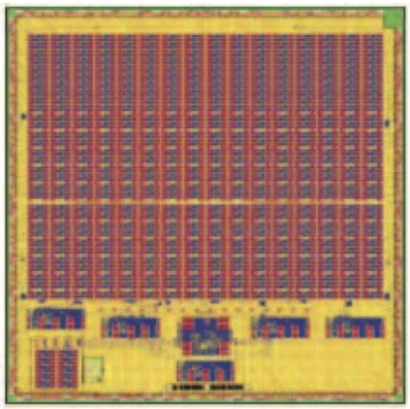


BRGTC2 (2018)

TSMC 28nm
1mm x 1.25mm
[WOSET'18, RISCV'18]

Research Purpose

Explore preliminary new ideas on hardware optimization for task-based parallel runtimes and also silicon-validate PyMTL in a more advanced node



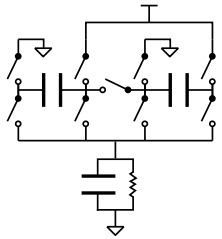
Celerity (2017)

TSMC 16nm FinFET
5mm x 5mm
[IEEE-MICRO'18
CARRV'17
HOTCHIPS'17]

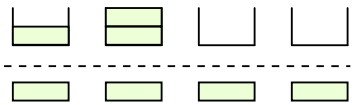
Research Purpose

Explore a range of productive hardware design and verification tools -- synthesizable analog IP (PLL, digital LDO), high-level synthesis (complex HLS-generated BNN accelerator), Python-based HDLs, area-efficient and high-bandwidth manycore networks (496-core RISC-V tiled manycore), and open-source generators.

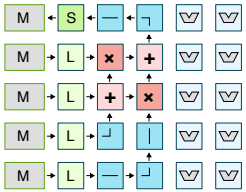
Software, Architecture, and VLSI Co-Design for Exploiting Fine-Grain Asymmetry



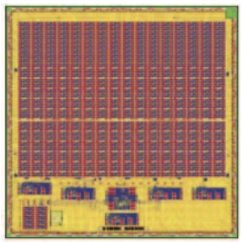
Exploiting Fine-Grain Asymmetry with
On-Chip Voltage Regulation - **MICRO'14**, IEEE TCAS I'18



Exploiting Fine-Grain Asymmetry in
Task-Based Parallel Runtimes - **ISCA'16**, MICRO'17, RISCV'18



Exploiting Fine-Grain Asymmetry in
Coarse-Grain Reconfigurable Arrays - **Unpublished**



Exploiting Fine-Grain Asymmetry with
Silicon Prototyping - **IEEE MICRO'18**, DAC'18, Hotchips'17

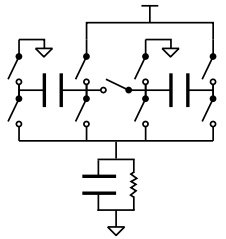
Conclusion

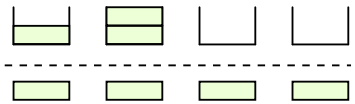
Acknowledgements and Funding

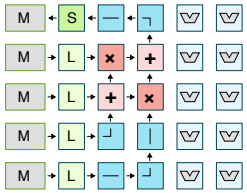
Acknowledgements and Funding

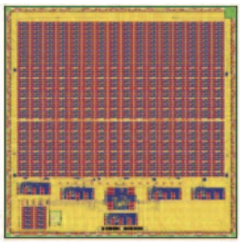
- **Batten Research Group**: Derek Lockhart, Ji Kim, Shreesha Srinath, Berkin Ilbeyi, Moyang Wang, Shunning Jiang, Khalid Al-Hawaj, Tuan Ta, Lin Cheng, Yanghui Ou, Peitian Pan, Cheng Tan, Shady Agwa, Christopher Batten
- **Apsel Research Group**: Waclaw Godycki, Ivan Bukreyev, Alyssa Apsel
- **UCSD / University of Washington**: Scott Davidson, Paul Gao, Atieh Lotfi, Julian Puscar, Loai Salem, Anuj Rao, Ningxiao Sun, Luis Vega, Bandhav Veluri, Xiaoyang Wang, Shaolin Xie, Chun Zhao, Michael B. Taylor
- **University of Michigan**: Tutu Ajayi, Aporva Amarnath, Austin Rovinski, Ronald G. Dreslinski
- **NVIDIA**: Brucek Khailany, Evgeni Krimer, Rangharajan Venkatesan, Jason Clemons, Joel Emer, Matthew Fojtik, Alicia Klinefelter, Michael Pellauer, Nathaniel Pinckney, Yakun Sophia Shao, Shreesha Srinath, Sam (Likun) Xi, Yanqing Zhang, Brian Zimmer
- **Celerity**: Tutu Ajayi, Khalid Al-Hawaj, Aporva Amarnath, Steve Dai, Scott Davidson, Paul Gao, Gai Liu, Atieh Lotfi, Julian Puscar, Anuj Rao, Austin Rovinski, Loai Salem, Ningxiao Sun, Luis Vega, Bandhav Veluri, Xiaoyang Wang, Shaolin Xie, Chun Zhao, Ritchie Zhao, Christopher Batten, Ronald G. Dreslinski, Ian Galton, Rajesh K. Gupta, Patrick P. Mercier, Mani Srivastava, Michael B. Taylor, Zhiru Zhang
- **BRGTC1/2**: Shunning Jiang, Khalid Al-Hawaj, Ivan Bukreyev, Berkin Ilbeyi, Tuan Ta, Lin Cheng, Julian Puscar, Ian Galton, Moyang Wang, Bharath Sudheendra, Nagaraj Murali, Suren Jayasuriya, Shreesha Srinath, Taylor Pritchard, Robin Ying, Christopher Batten

Takeaway Points









Parallelism and specialization are increasing on-chip **fine-grain asymmetry**, which must be managed by similarly **fine-grain power-control techniques**

Controlling these techniques through **cross-stack co-design** can enable:

- Smaller area for *integrated voltage regulation*
- Asymmetry-aware *task-based parallel runtimes*
- More efficient *elastic CGRAs*

Thank you!