Software, Architecture, and VLSI Co-Design for Efficient Task-Based Parallel Runtimes

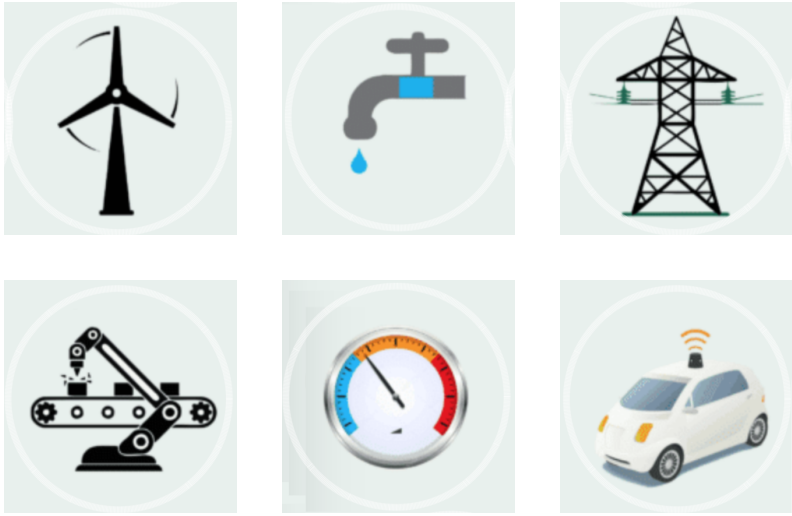
Christopher Torng

Computer Systems Laboratory
School of Electrical and Computer Engineering
Cornell University

Emerging New Contexts Demand Better Hardware

Pushing Intelligence to the Edge

- Better local security
- Faster response times
- Lower data-movement energy
- Many more...

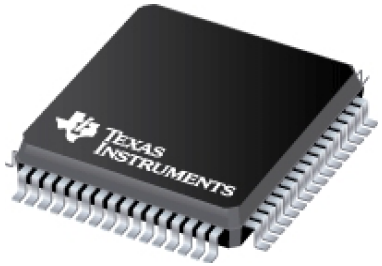


Source: Lanner

Peak Performance ⟵ ⟶ Energy Efficiency

Inference 2.5 sec
Image 28 x 28
MNIST dataset

Source: Gobieski ASPLOS'19

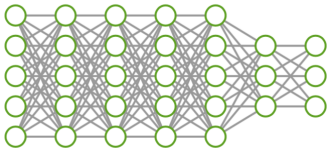


TI MSP430

10+ years
CR2032 coin
Standby mode

Emerging New Contexts Demand Better Hardware

Machine Learning

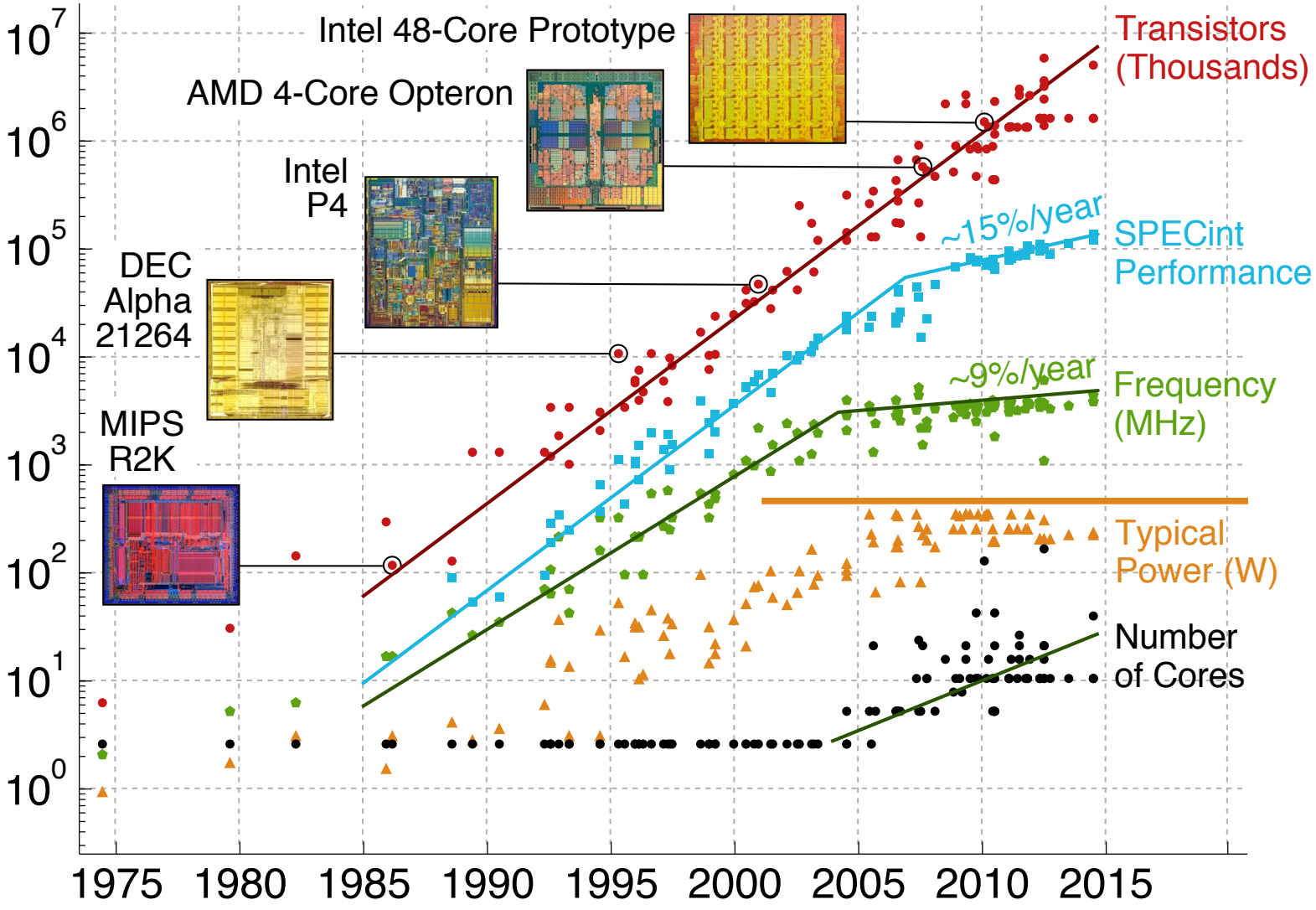


Graph Analytics

- Cybersecurity
- Smart Healthcare
- Smart Home
- Augmented Reality
- Virtual Reality
- Autonomous Driving

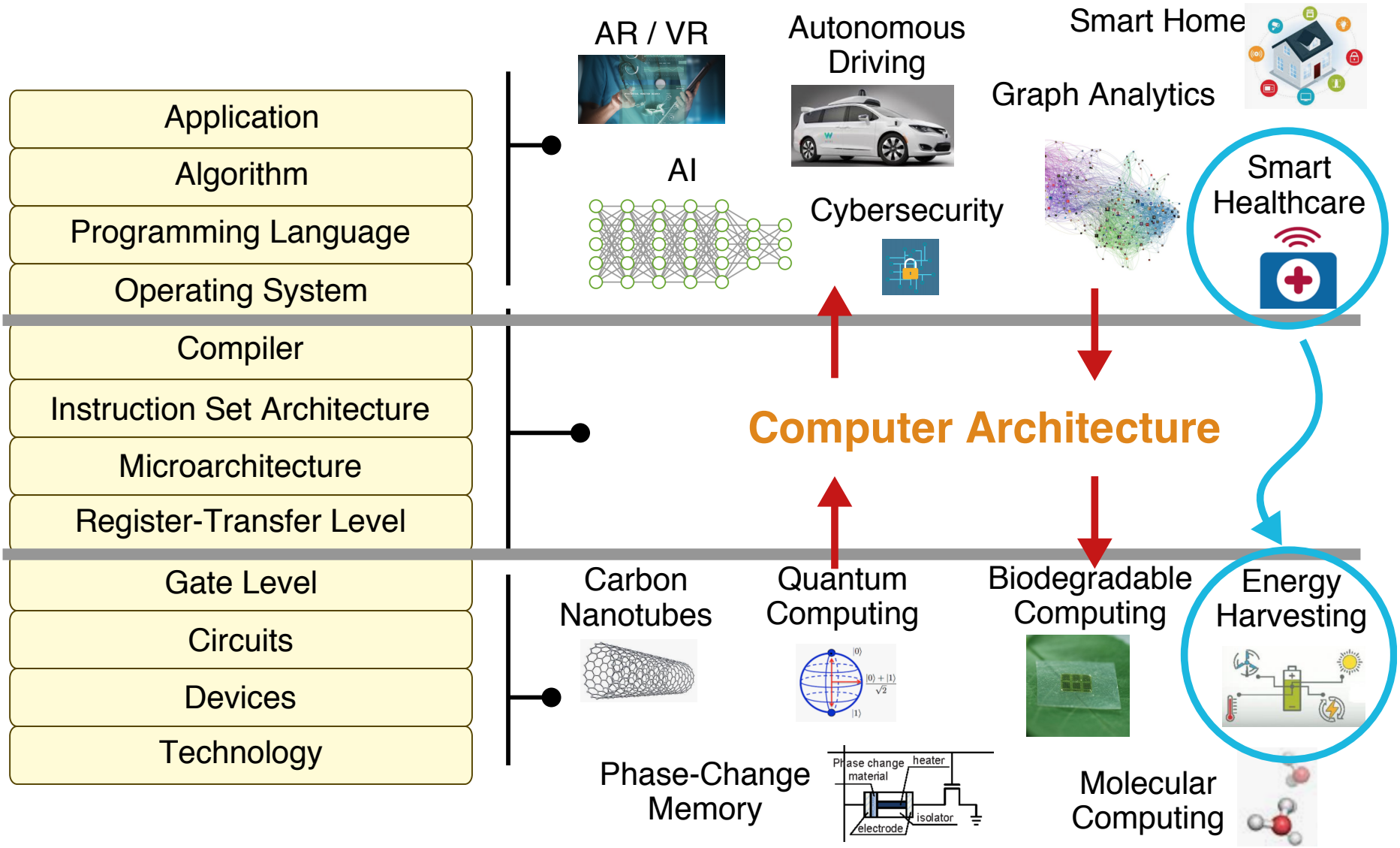
How can we drastically improve performance and energy efficiency for these new emerging contexts?

Motivating Trends in Computer Architecture

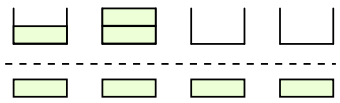


Data collected by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, C. Batten

Excitement After Moore's Law

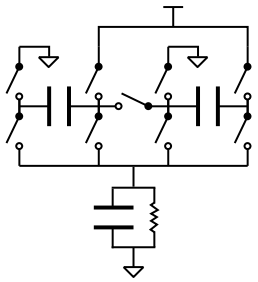


Building Future Computing Systems that Bridge Software, Architecture, and VLSI



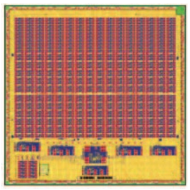
Cross-Stack Co-Design for

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



Cross-Stack Co-Design for

Integrated Voltage Regulation - **MICRO'14**, IEEE TCAS I'18

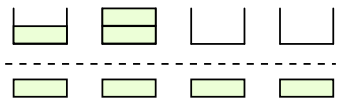


Cross-Stack Co-Design for

Rapid ASIC Design - **IEEE MICRO'18**, DAC'18, Hotchips'17

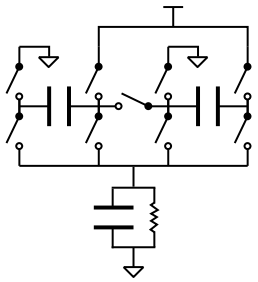
Future Research

Building Future Computing Systems that Bridge Software, Architecture, and VLSI



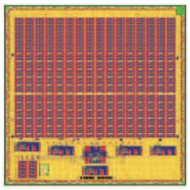
Cross-Stack Co-Design for

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



Cross-Stack Co-Design for

Integrated Voltage Regulation - **MICRO'14**, IEEE TCAS I'18

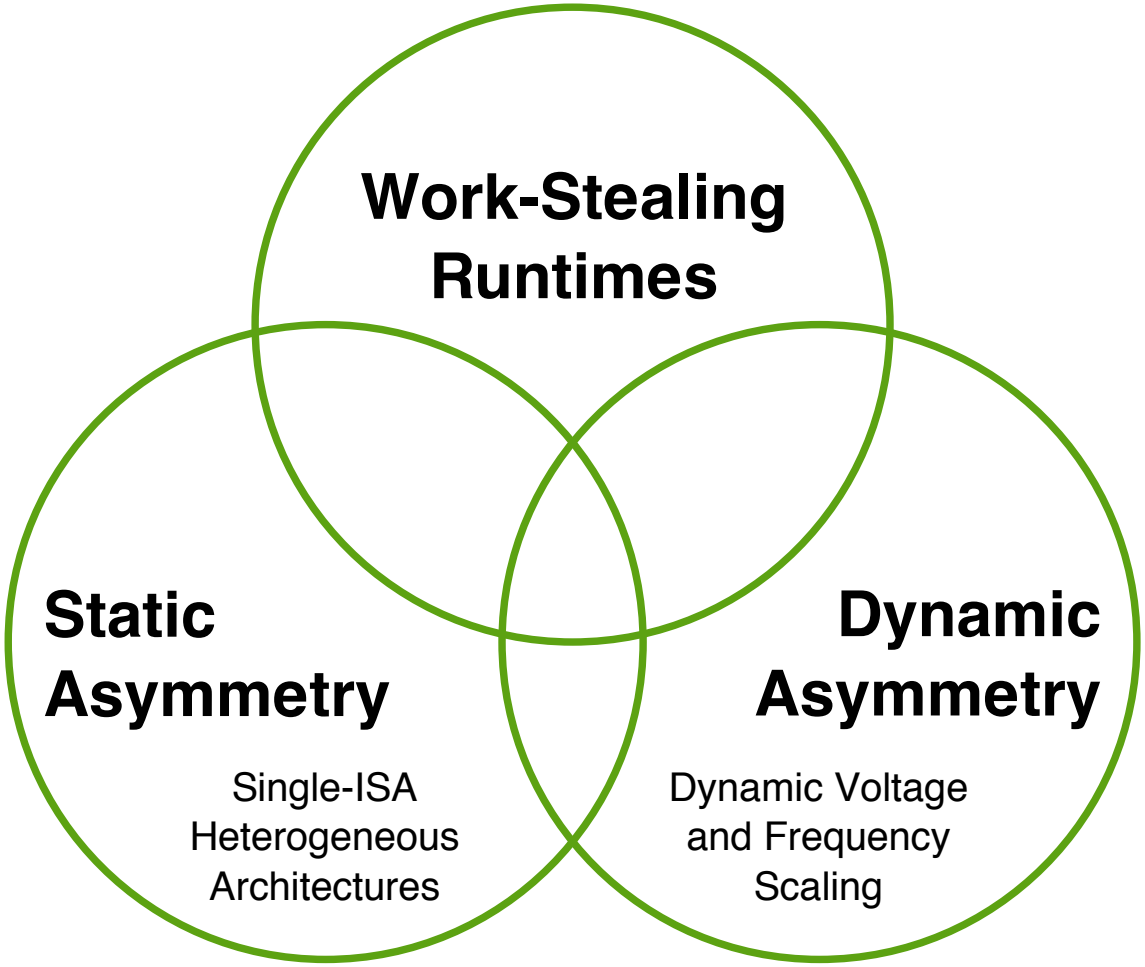


Cross-Stack Co-Design for

Rapid ASIC Design - **IEEE MICRO'18**, DAC'18, Hotchips'17

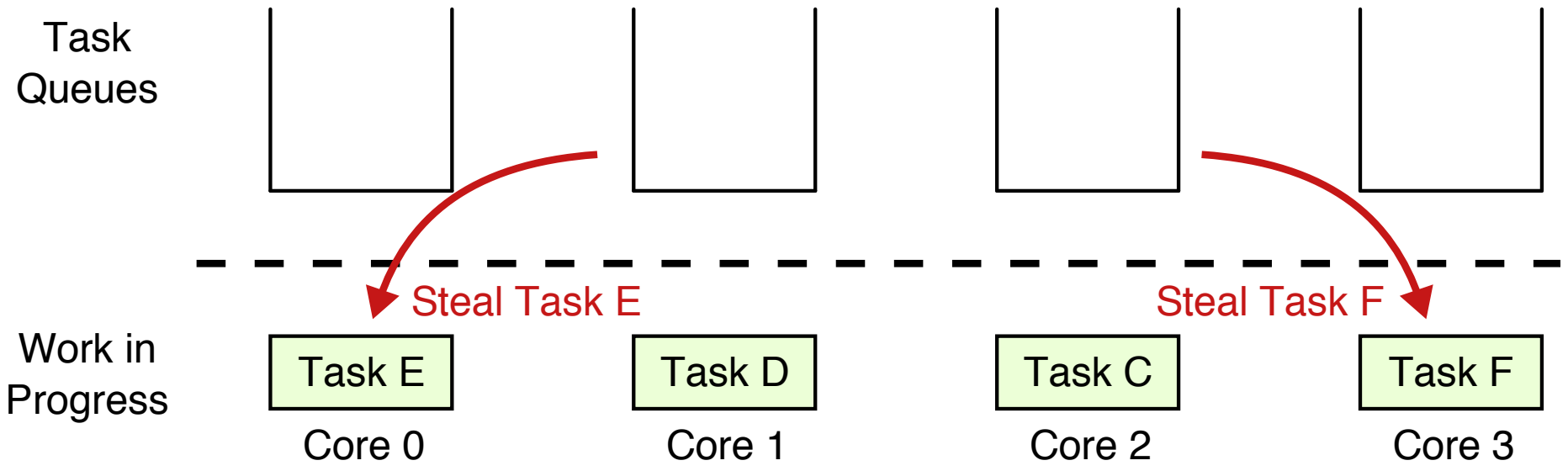
Future Research

Cross-Stack Co-Design for Task-Based Parallelism



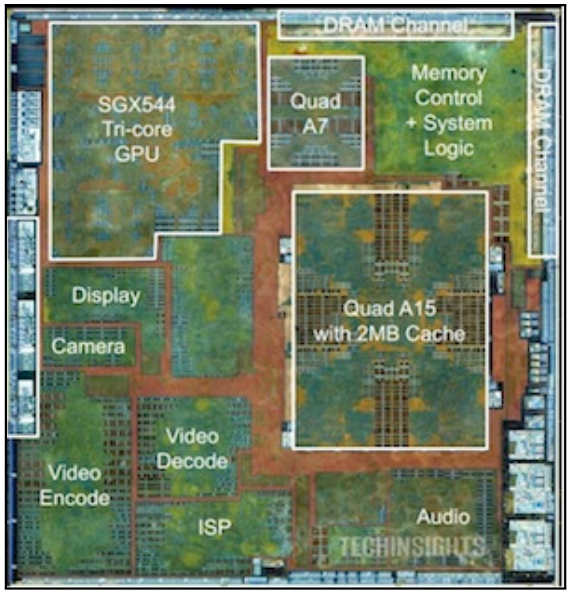
How can we use asymmetry awareness to improve the performance and energy efficiency of a work-stealing runtime?

Work-Stealing Runtimes

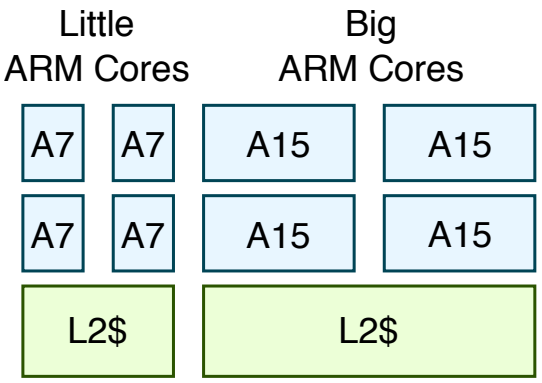


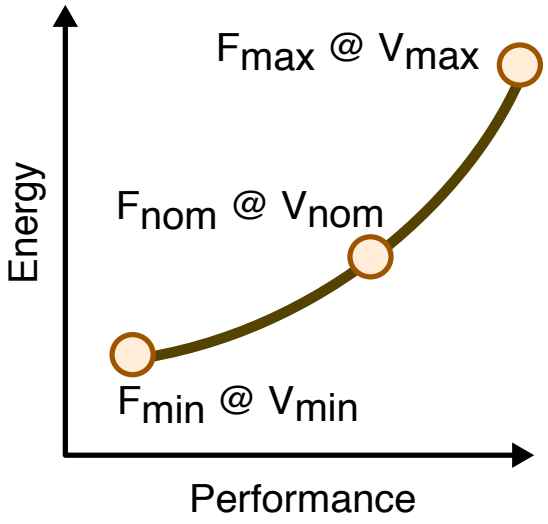
- Work stealing has good performance, space requirements, and communication overheads in both theory and practice
- Supported in many popular concurrency platforms including: Intel's Cilk Plus, Intel's C++ TBB, Microsoft's .NET Task Parallel Library, Java's Fork/Join Framework, and OpenMP

Static Asymmetry vs. Dynamic Asymmetry

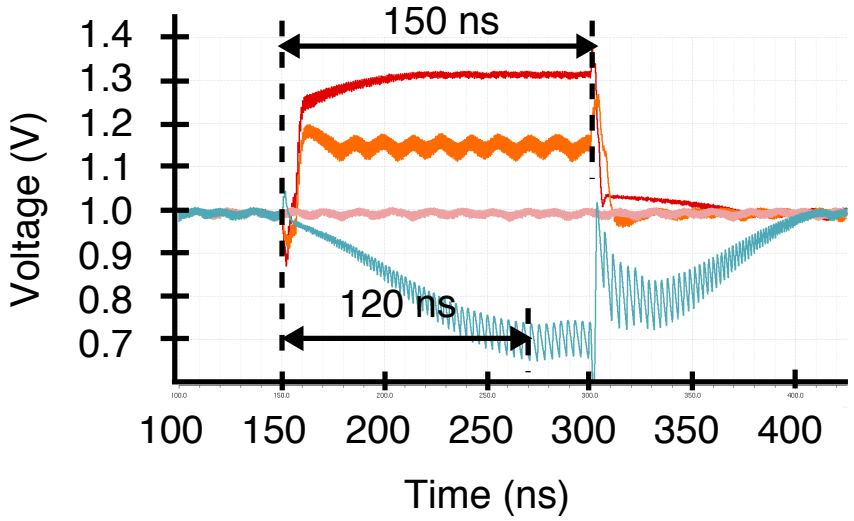


Samsung Exynos Octa Mobile Processor





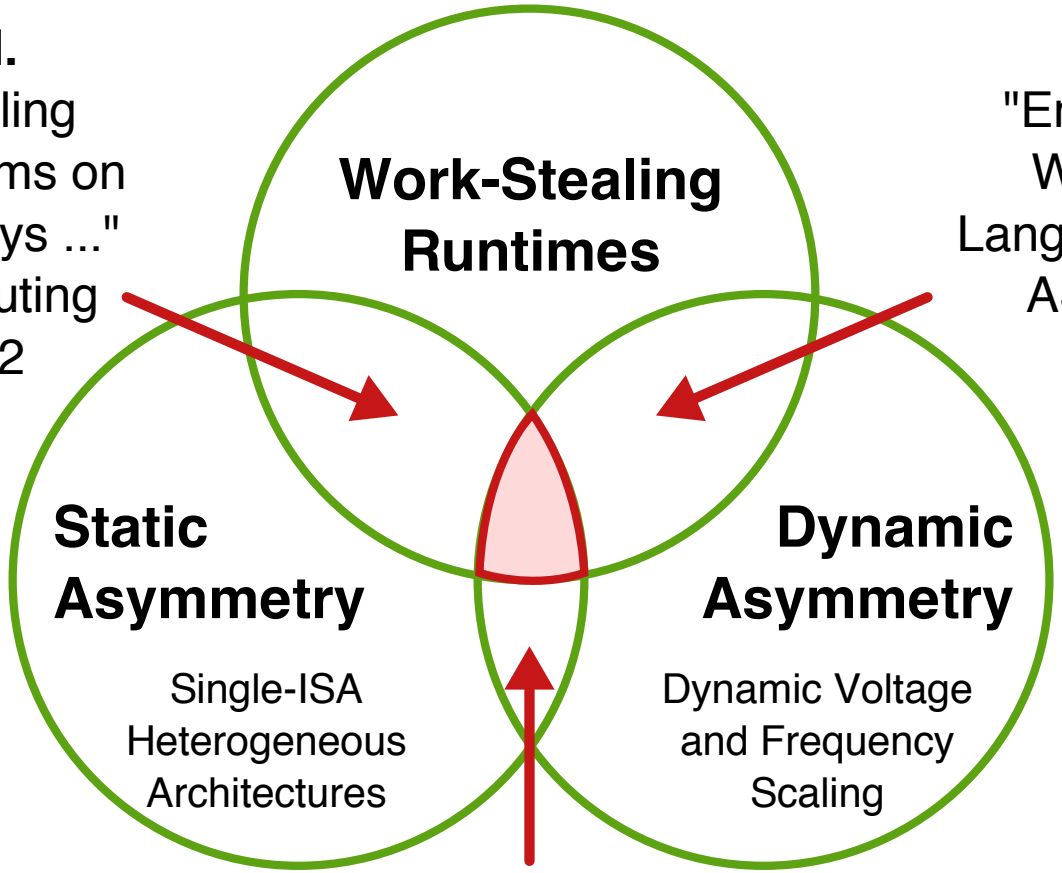
Integrated Voltage Regulation



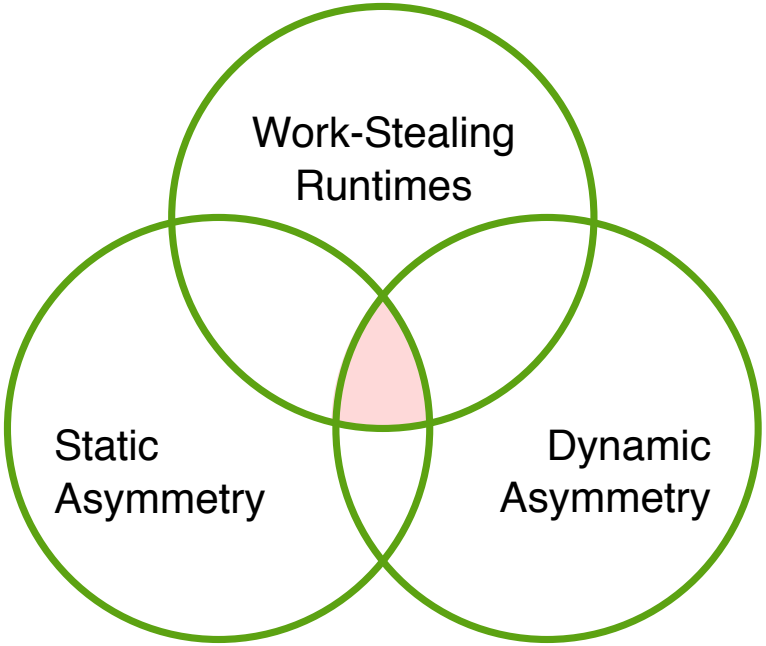
How can we use asymmetry awareness to improve the performance and energy efficiency of a work-stealing runtime?

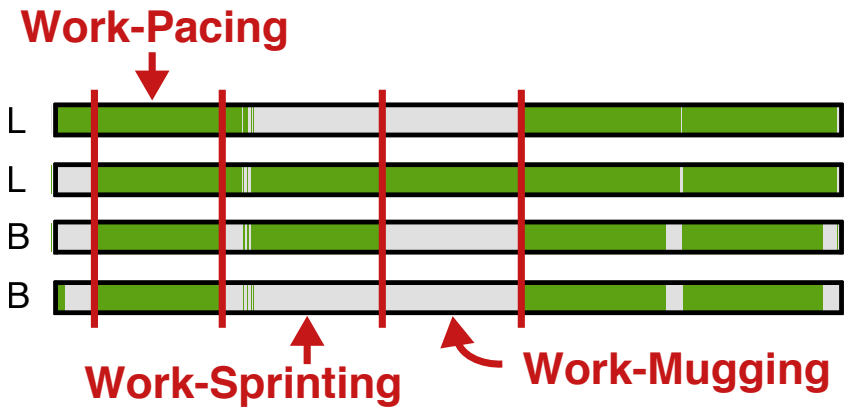
Bender et al.
"Online Scheduling of Parallel Programs on Heterogeneous Sys ..." Theory of Computing Systems 2002

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

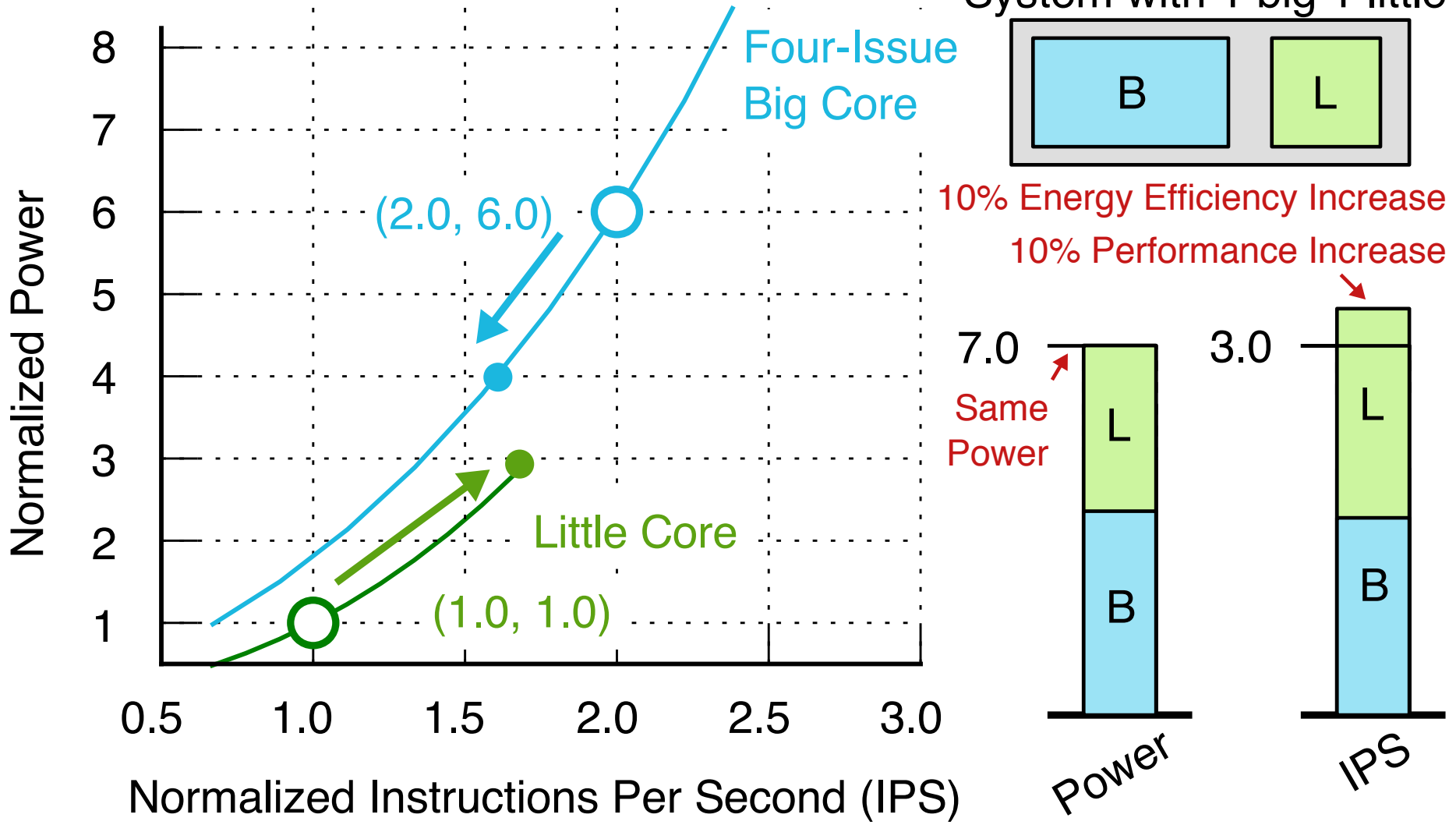




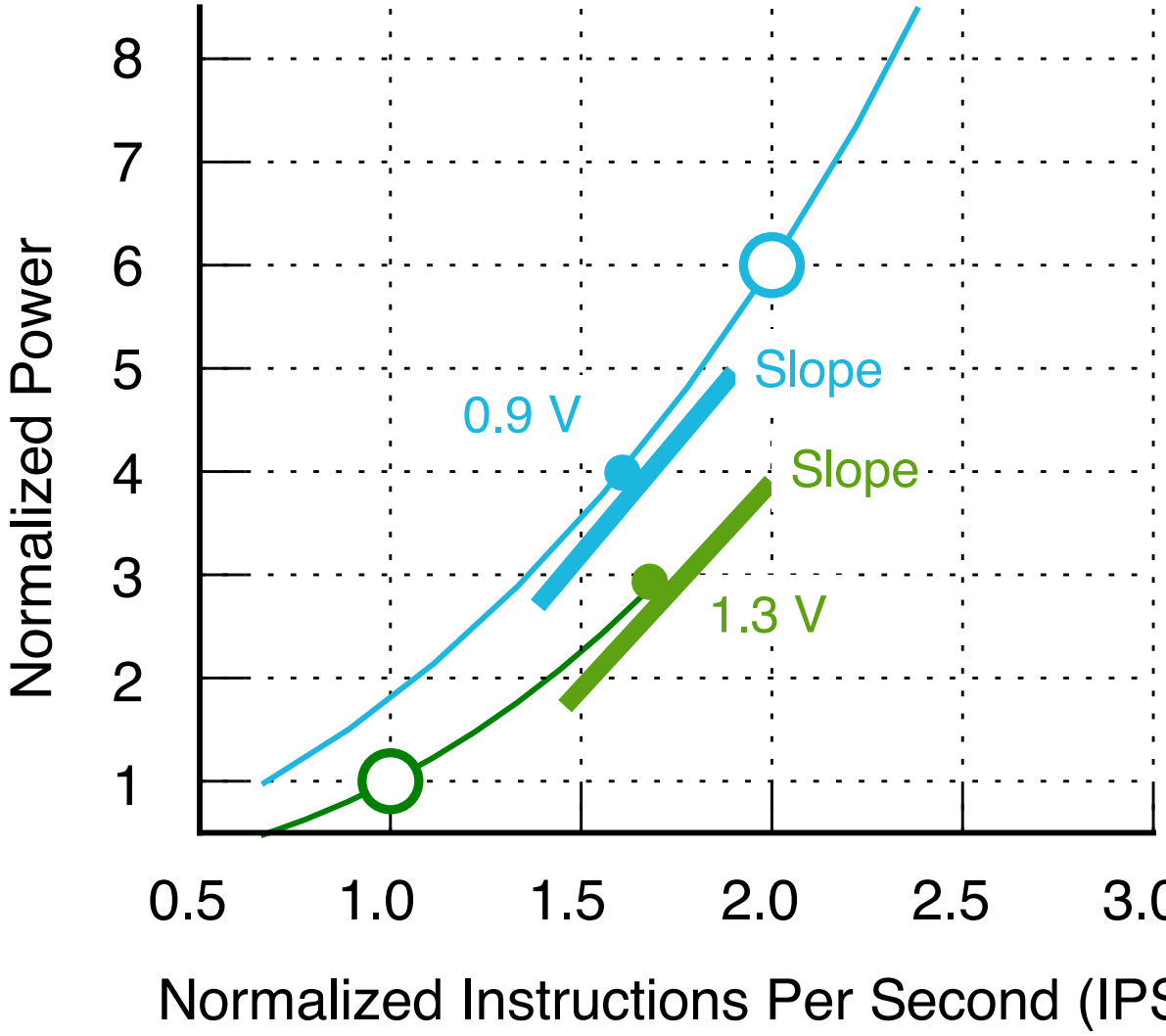
Cross-Stack Co-Design for Task-Based Parallelism

Let's start with some first-order modeling to build intuition

Building Intuition by Exploring a 1 Big 1 Little System



The Law of Equi-Marginal Utility



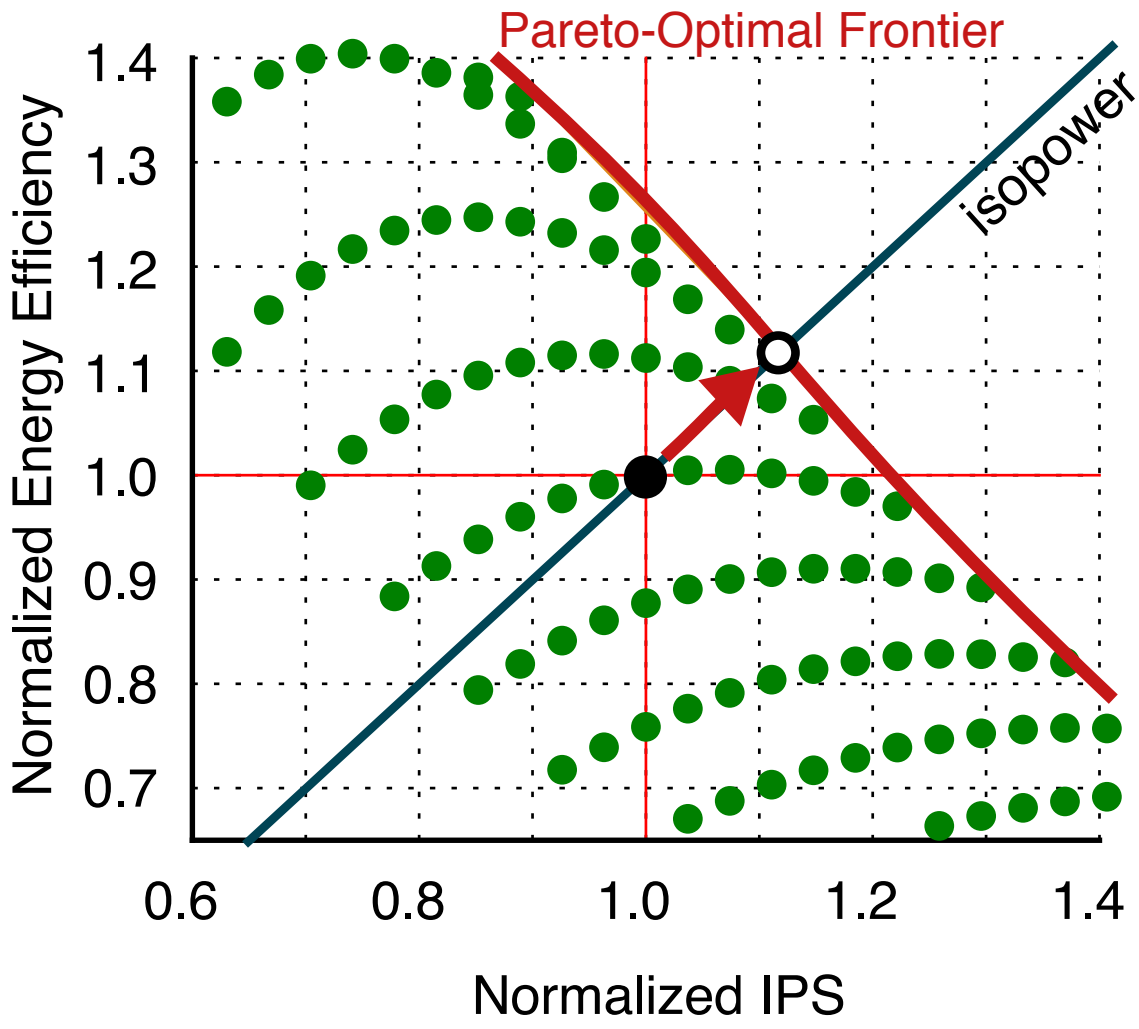
British Economist
Alfred Marshall (1824 - 1924)

"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**."

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
- Individual (V_B, V_L) pair

Assumptions

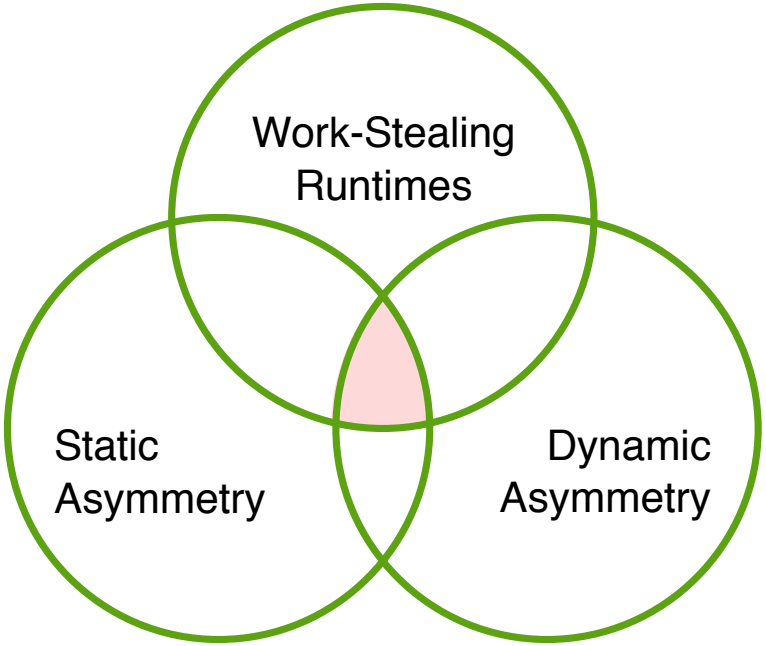
Perfectly parallel application
Ideal load balancing

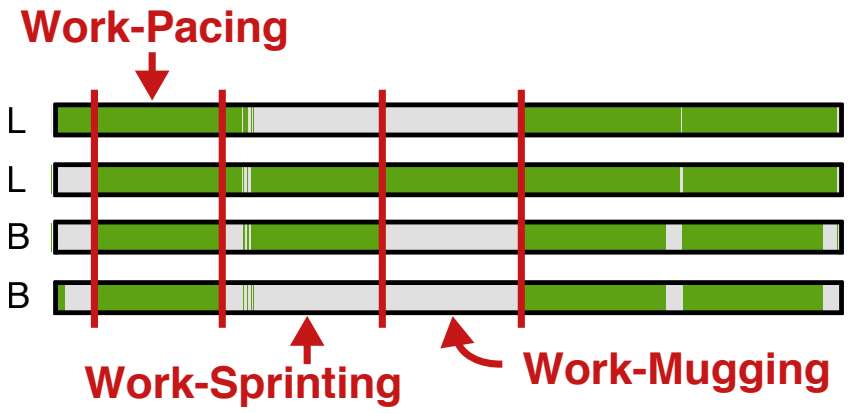
Marginal Utility-Based Optimization Problem

Constraint: *isopower line*

Objective: *maximize performance*

(Solved numerically)



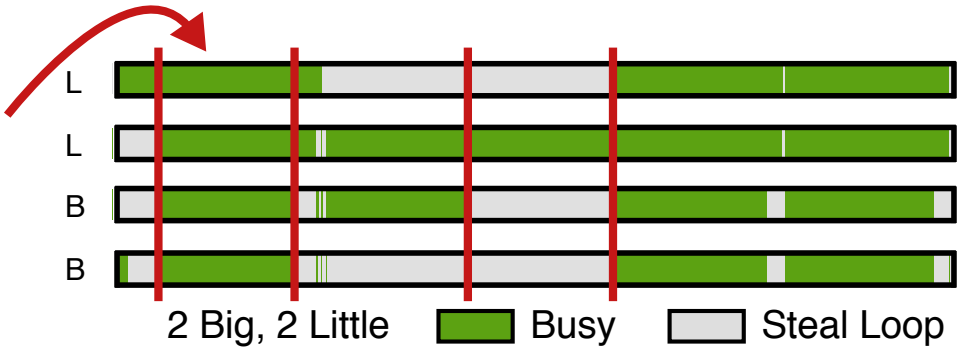


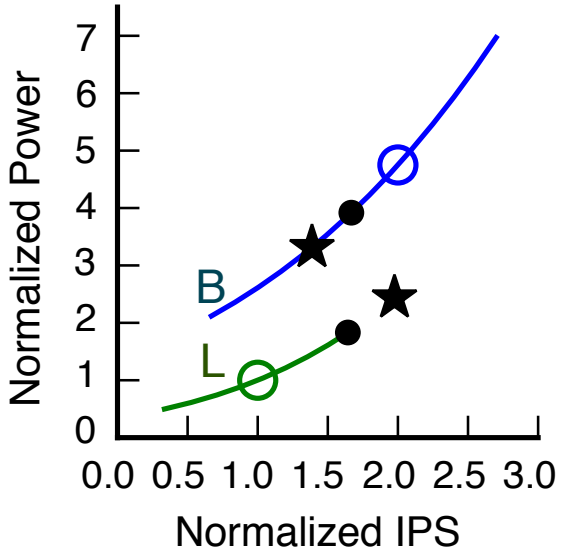
Cross-Stack Co-Design for Task-Based Parallelism

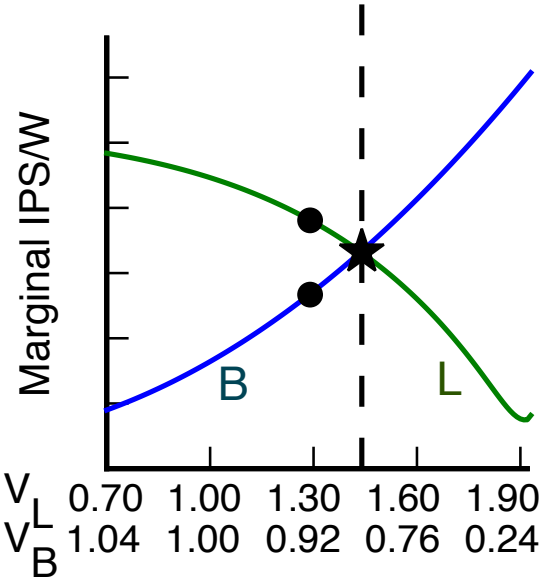
Let's explore three **specific techniques** to balance marginal utility in a work-stealing runtime

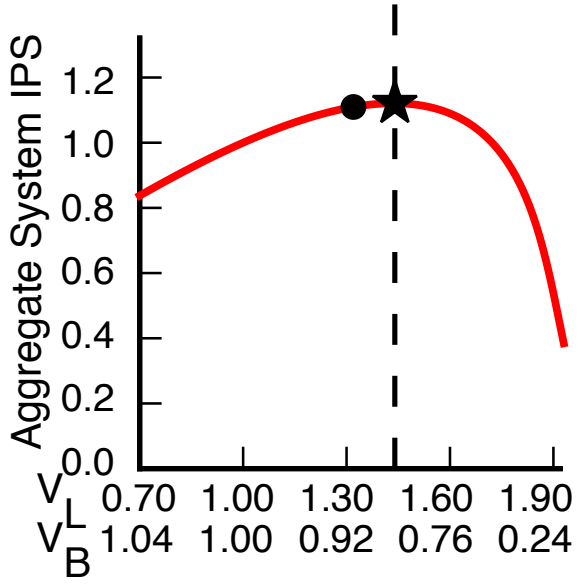
Work-Pacing: Building Intuition

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



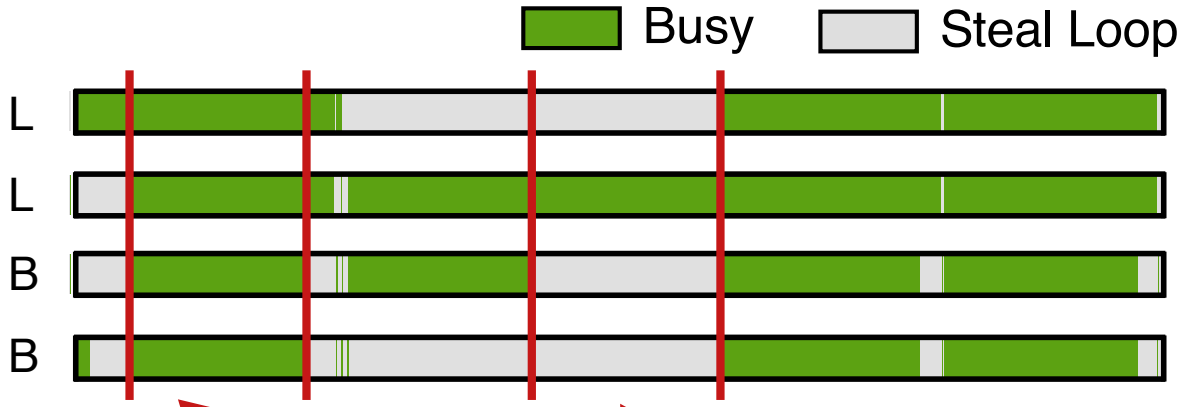






System with both big cores active and both little cores active

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



Work-Pacing

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

Work-Sprinting

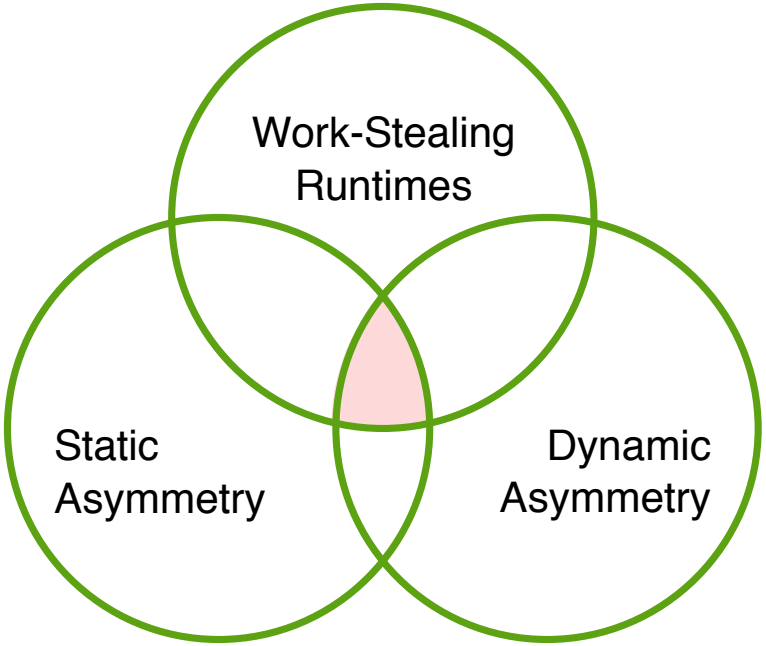
Rest cores in the steal loop to the lowest voltage

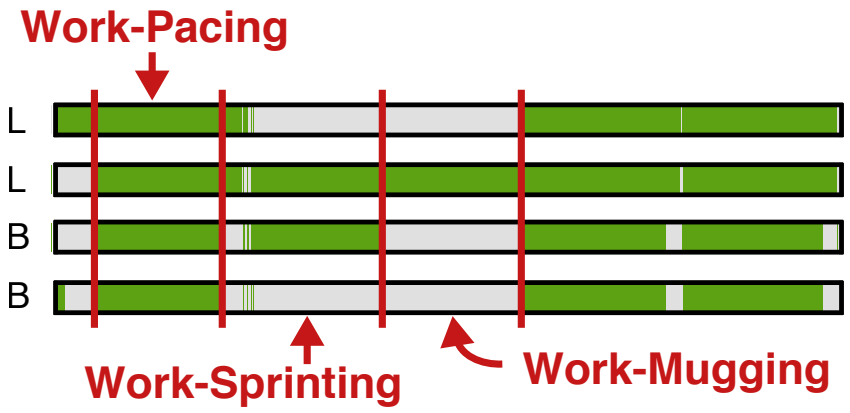
With additional power slack, balance performance/power across busy cores in the low-parallel (LP) region

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



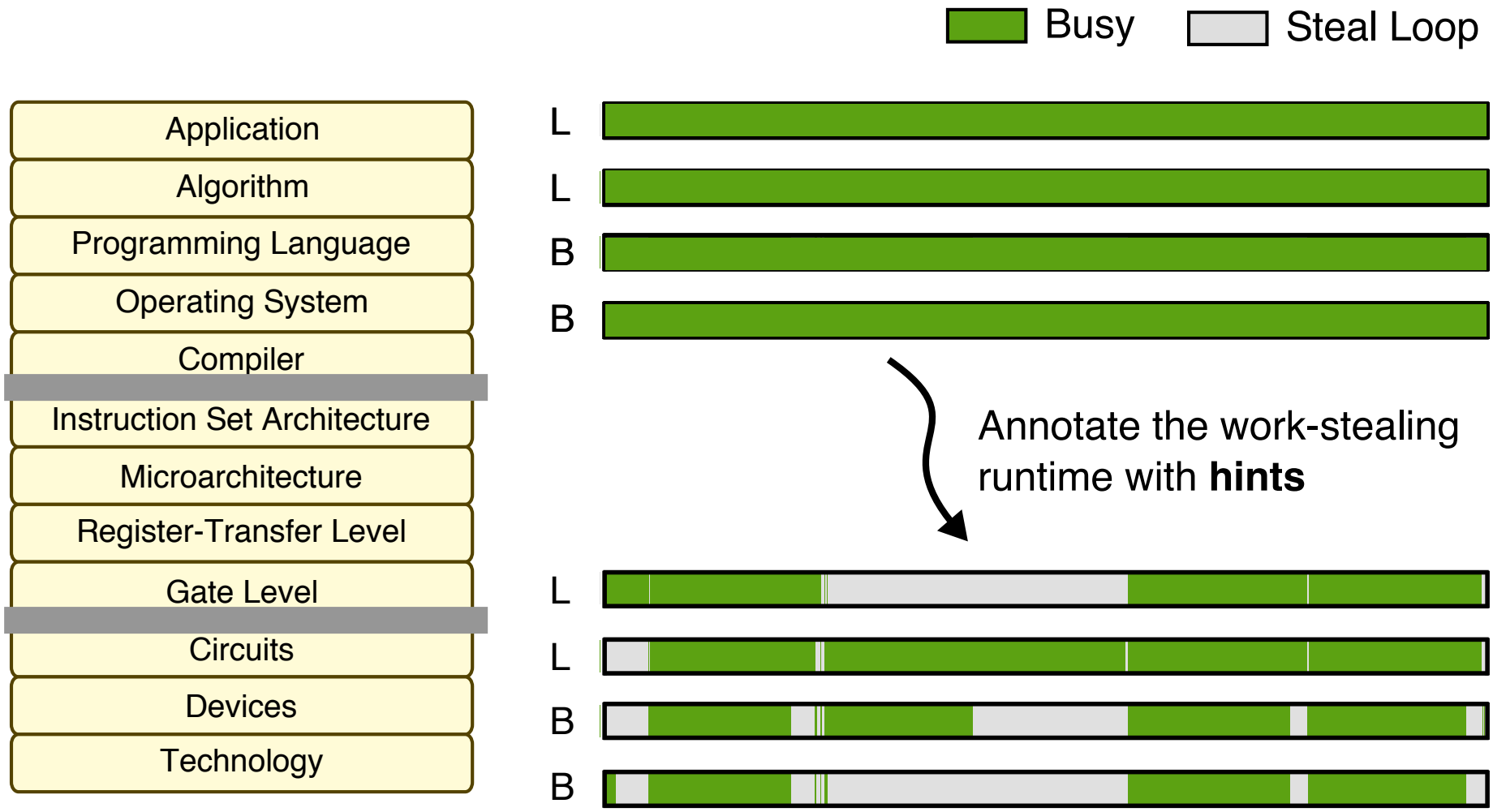


Cross-Stack Co-Design for Task-Based Parallelism

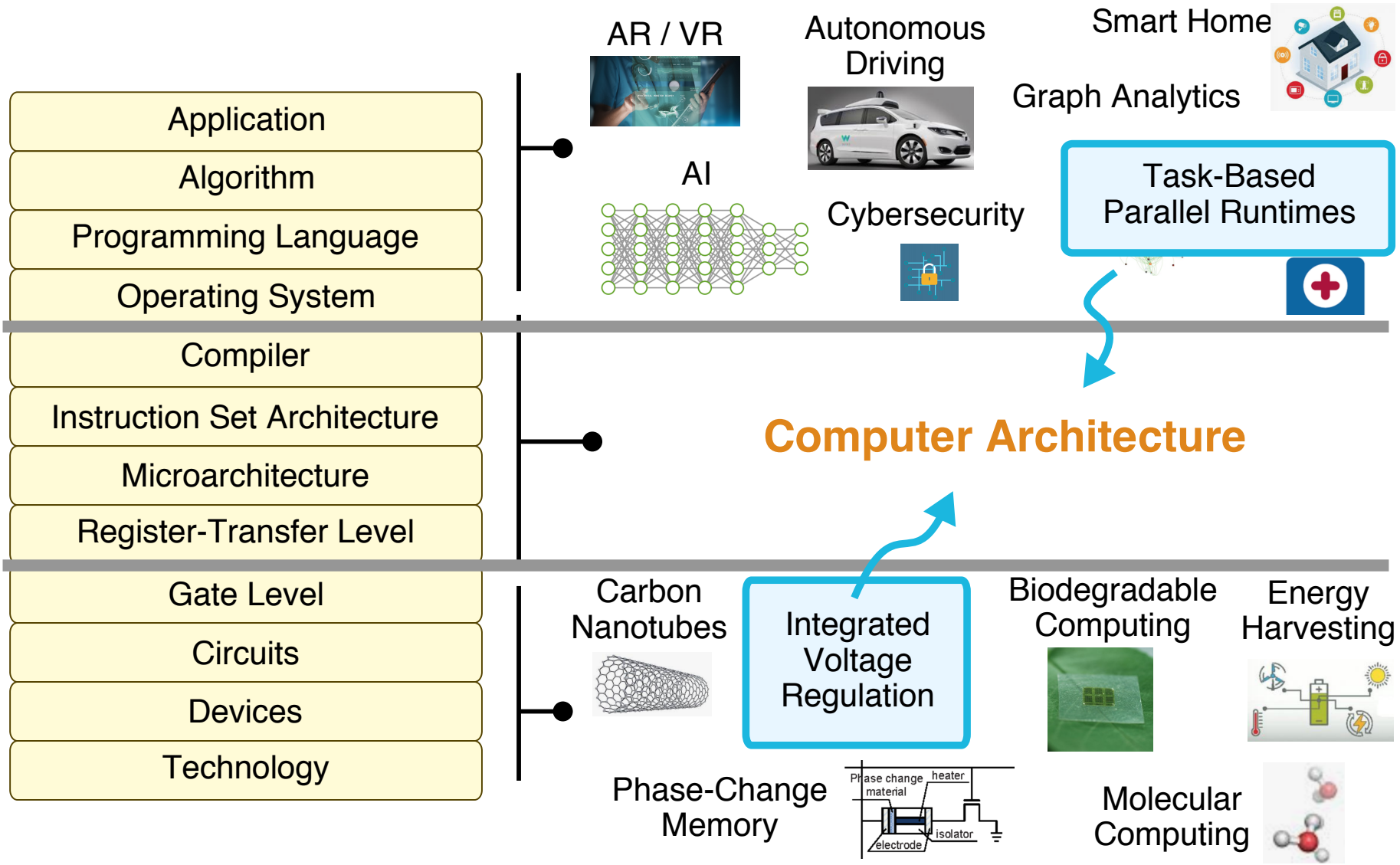
We have three techniques for balancing marginal utility

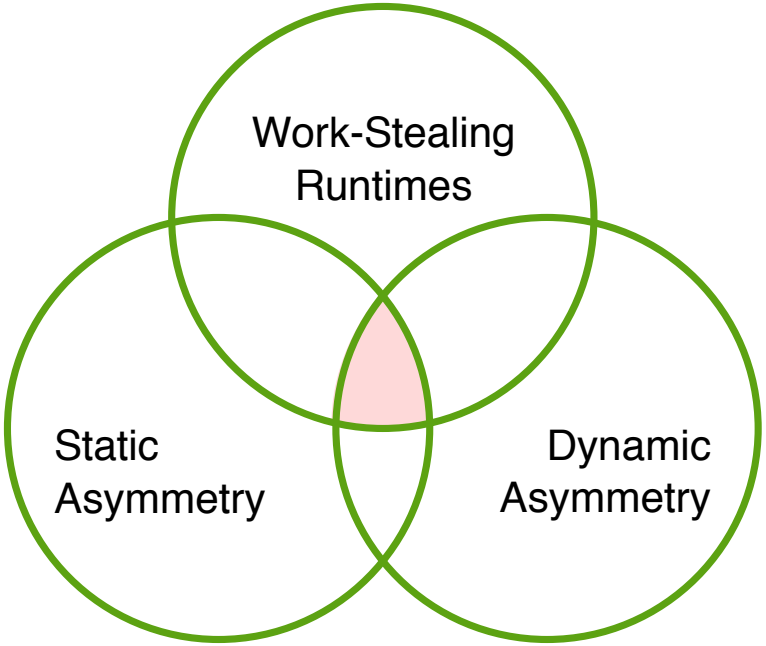
(but we're missing something)

Augmenting the Software/Architecture Interface



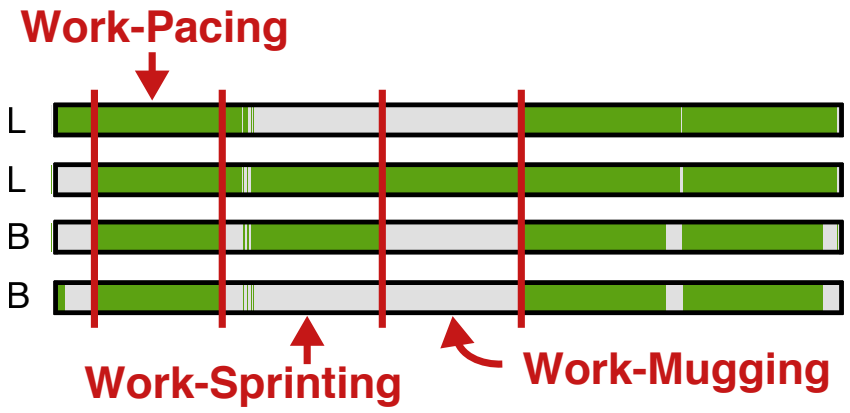
Popping Back Up a Level



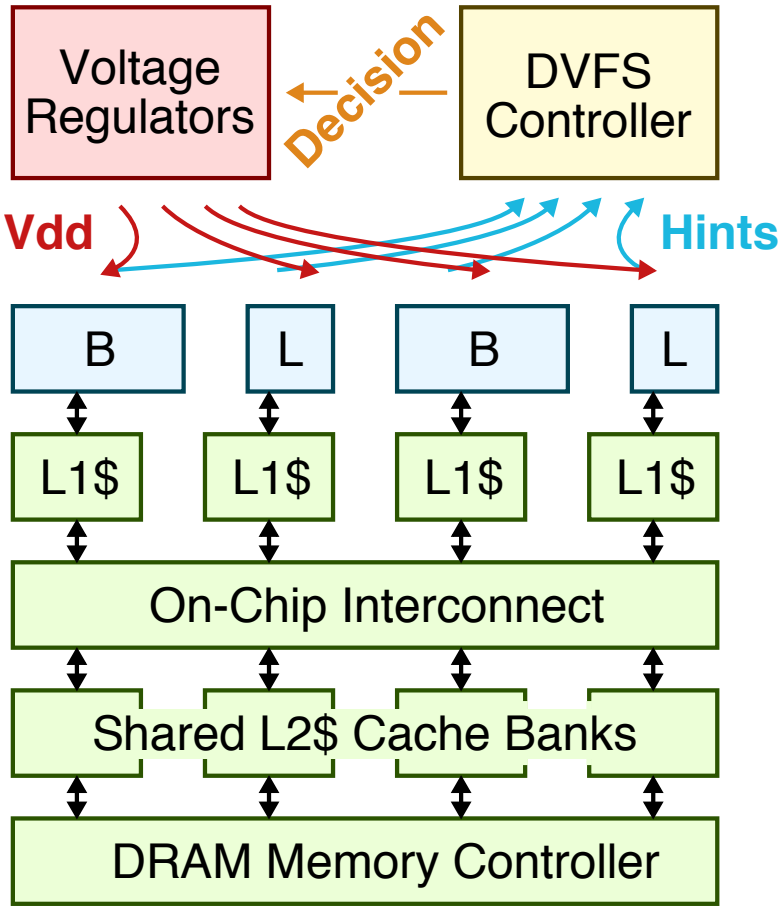


Cross-Stack Co-Design for Task-Based Parallelism

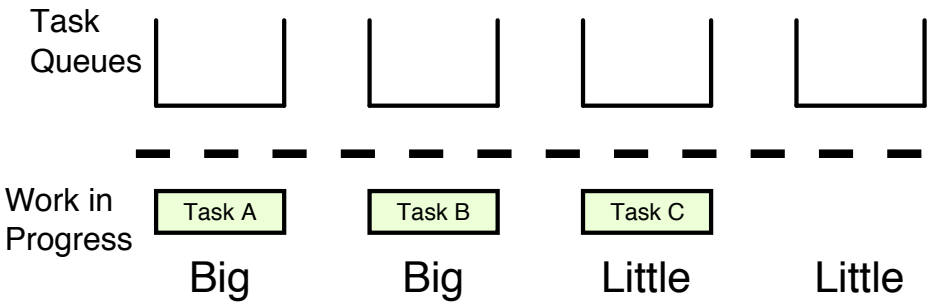
For the detail-minded, here are the specific mechanisms



Work-Pacing and Work-Sprinting Mechanisms



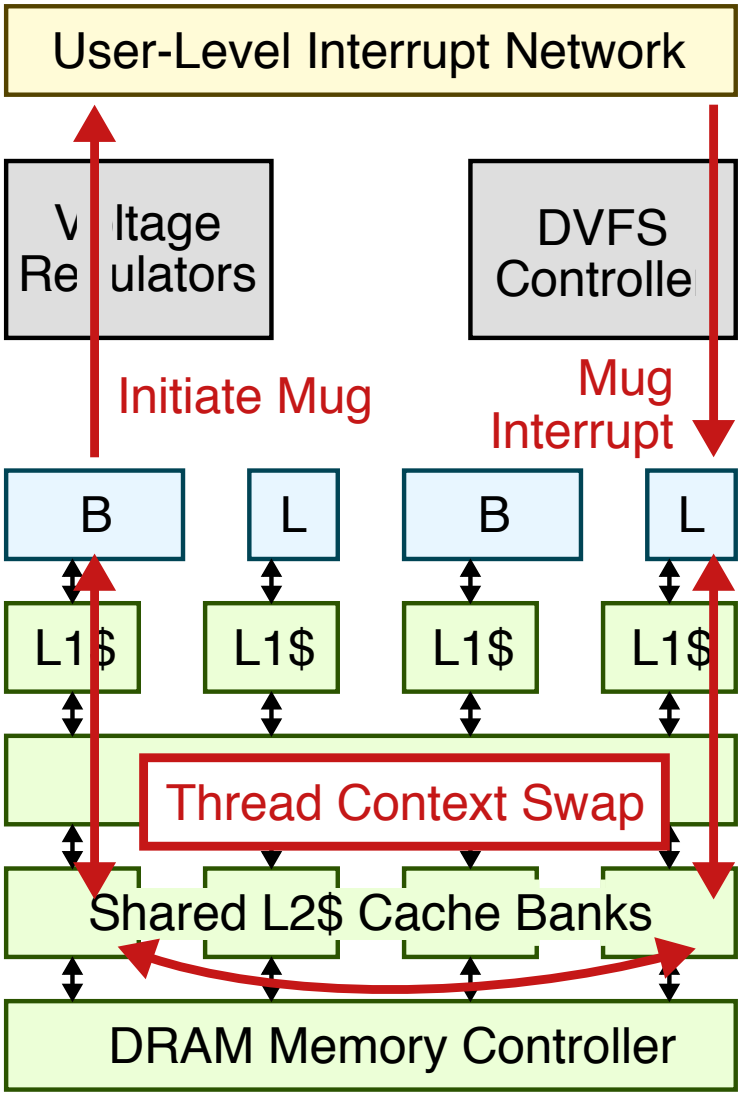
Which cores are stealing? Big or little?



Activity Pattern				Voltages		
B	B	L	L	V_B	V_L	Stealing
A	A	A	A	0.91V	1.30V	
A	A	A	S	0.98V	1.30V	0.70V
A	A	S	S	1.03V	1.30V	0.70V
A	S	A	A	1.04V	1.30V	0.70V
A	S	A	S	1.13V	1.30V	0.70V
A	S	S	S	1.21V	0.70V	0.70V
S	S	A	A	0.70V	1.30V	0.70V
S	S	A	S	0.70V	1.30V	0.70V
S	S	S	S	0.70V	0.70V	0.70V

A = Active, **S** = Stealing

Work-Mugging Mechanisms



Mug Instruction

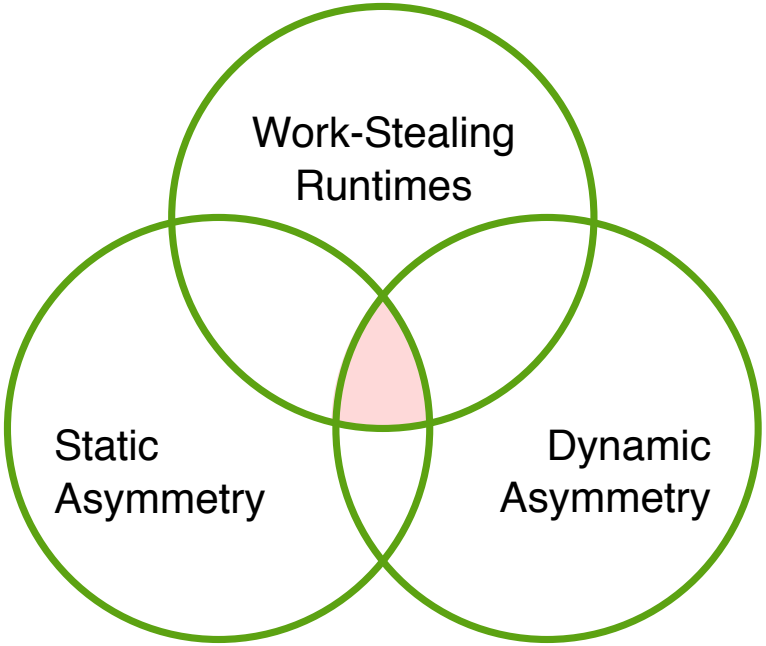
- Thread ID to mug
- Address of thread-swapping handler

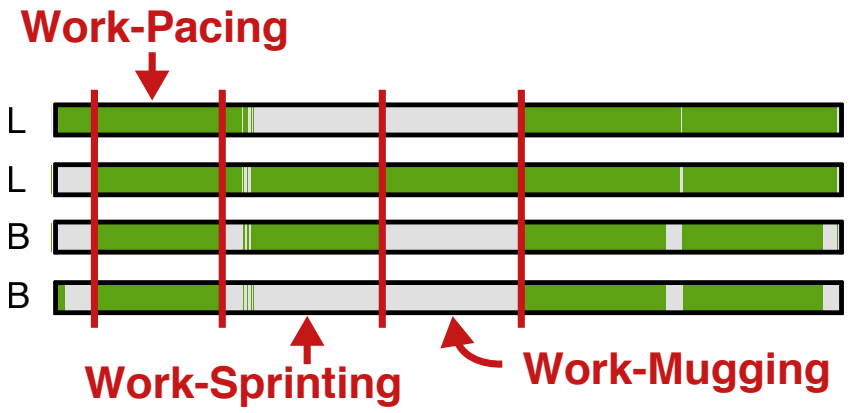
User-Level Interrupt Network

- Simple, low-bandwidth inter-core network
- Latency on order of 20 cycles

Thread Context Swap

- Threads store architectural state to separate locations in shared memory
- Both threads sync
- Threads load architectural state from other location





Cross-Stack Co-Design for Task-Based Parallelism

Let's see an asymmetry-aware work-stealing runtime in action

Evaluation Methodology: Modeling

Work-Stealing Runtime

- State-of-the-art Intel TBB-inspired work-stealing scheduler
- Chase-Lev task queues with occupancy-based victim selection
- Instrumented with activity hints

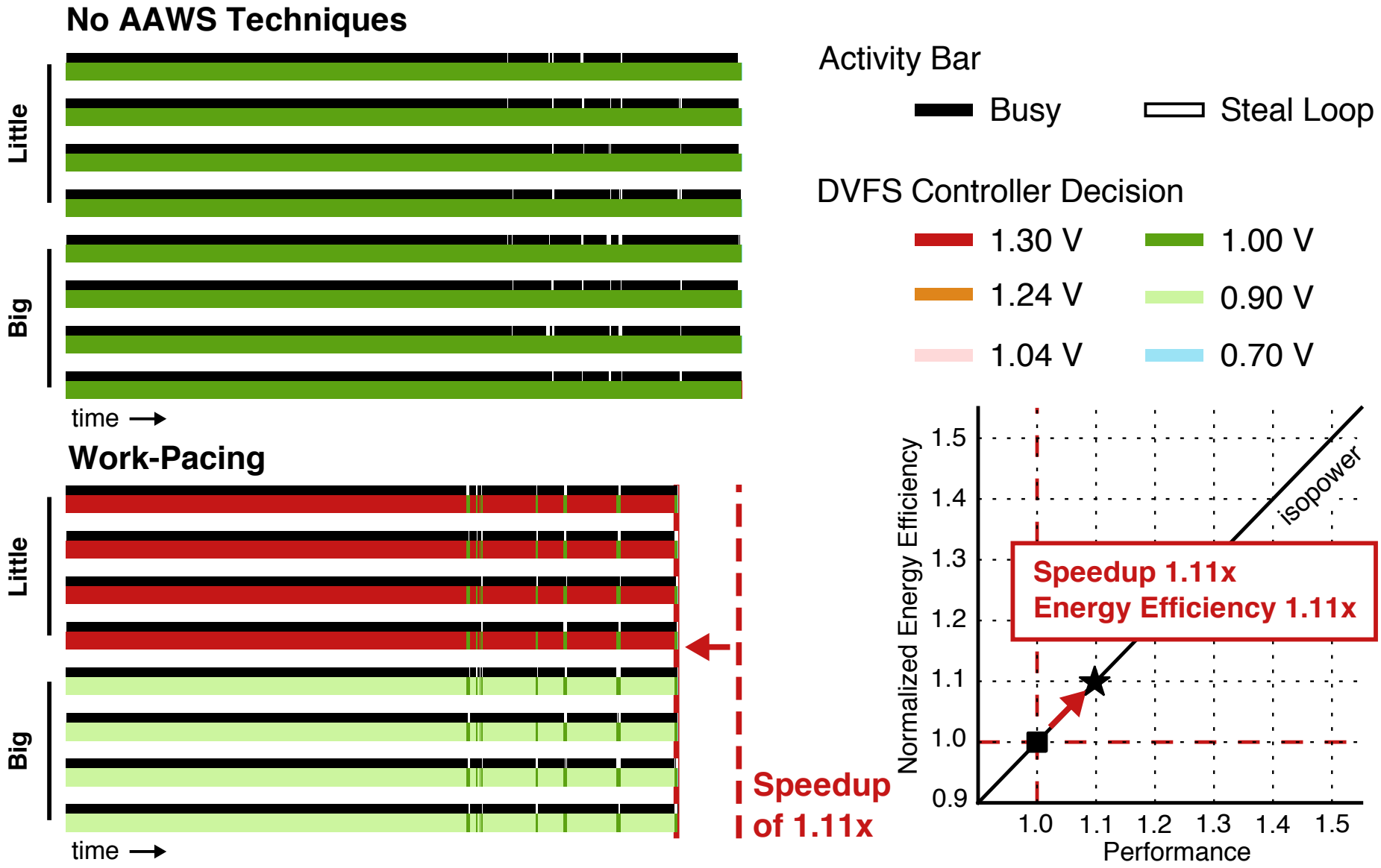
Cycle-Level Modeling

- Heterogeneous system modeled in gem5 cycle-approximate simulator
- Support for scaling per-core frequencies + central DVFS Controller

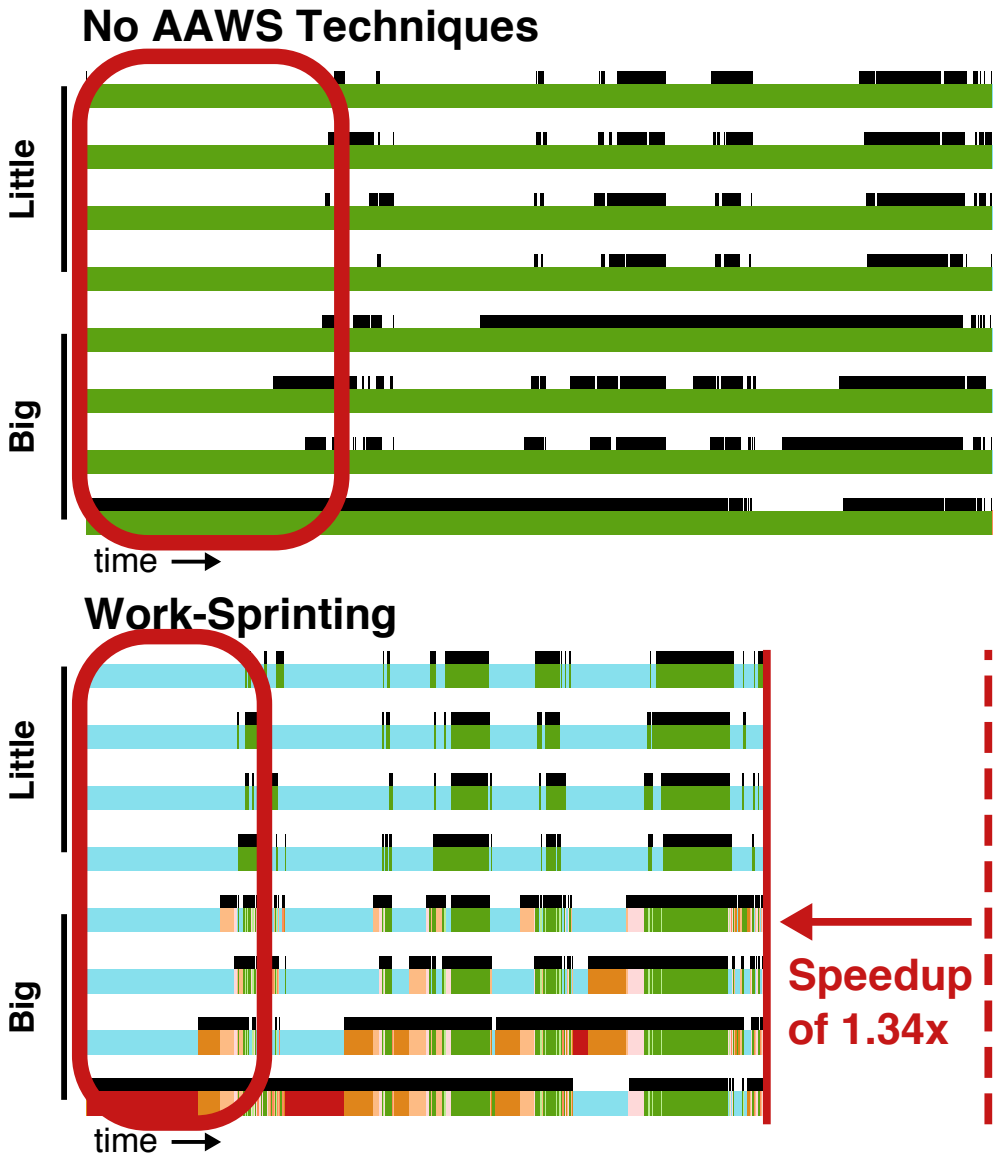
Energy Modeling

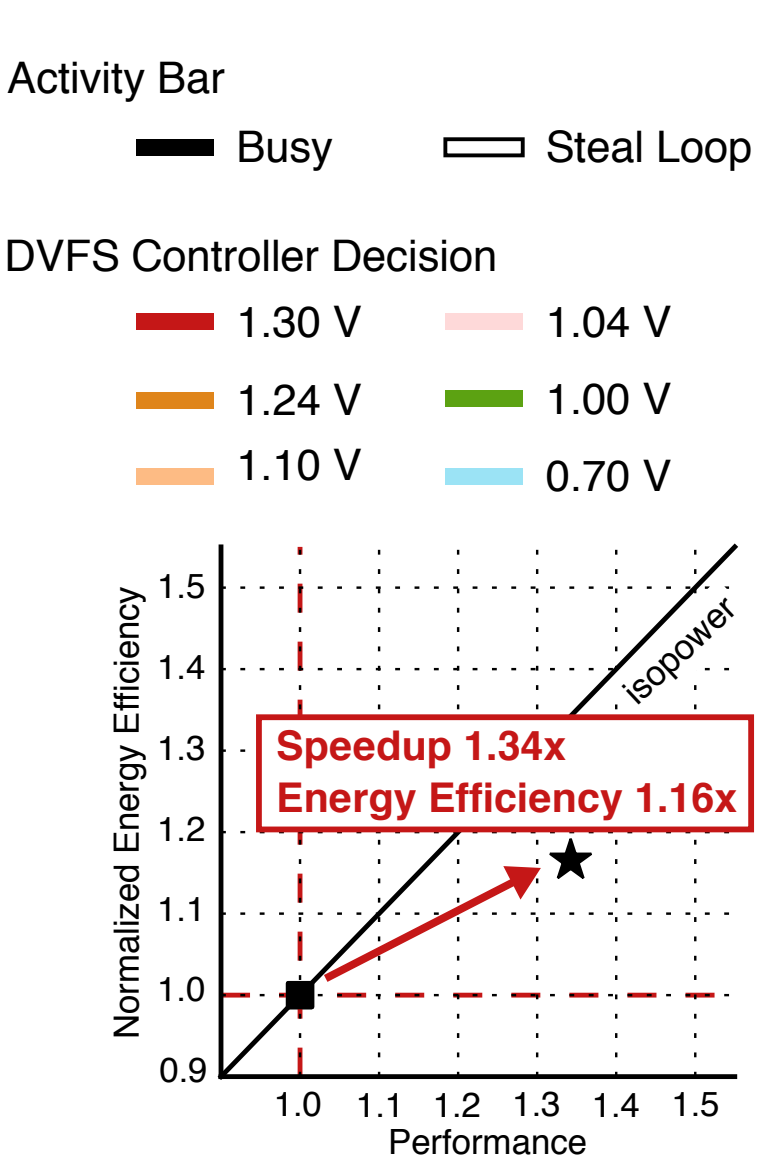
- Event-based energy modeling based on detailed RTL/gate-level sims (Synopsys ASIC toolflow, TSMC LP, 65 nm 1.0 V)
- Carefully selected subset of McPAT results tuned to our µarchitecture

Work-Pacing in *cilk-sort*

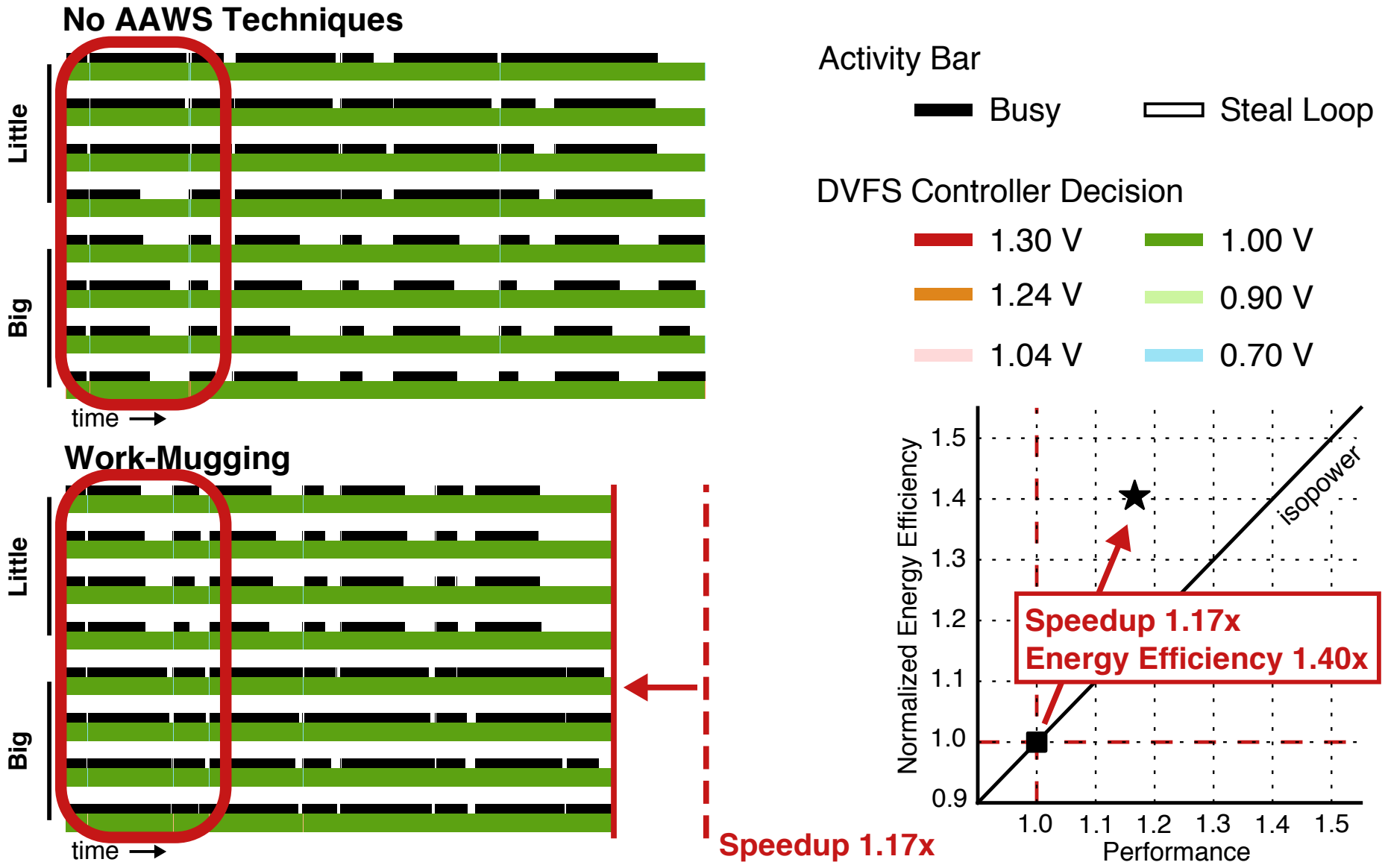


Work-Sprinting in *quicksort*

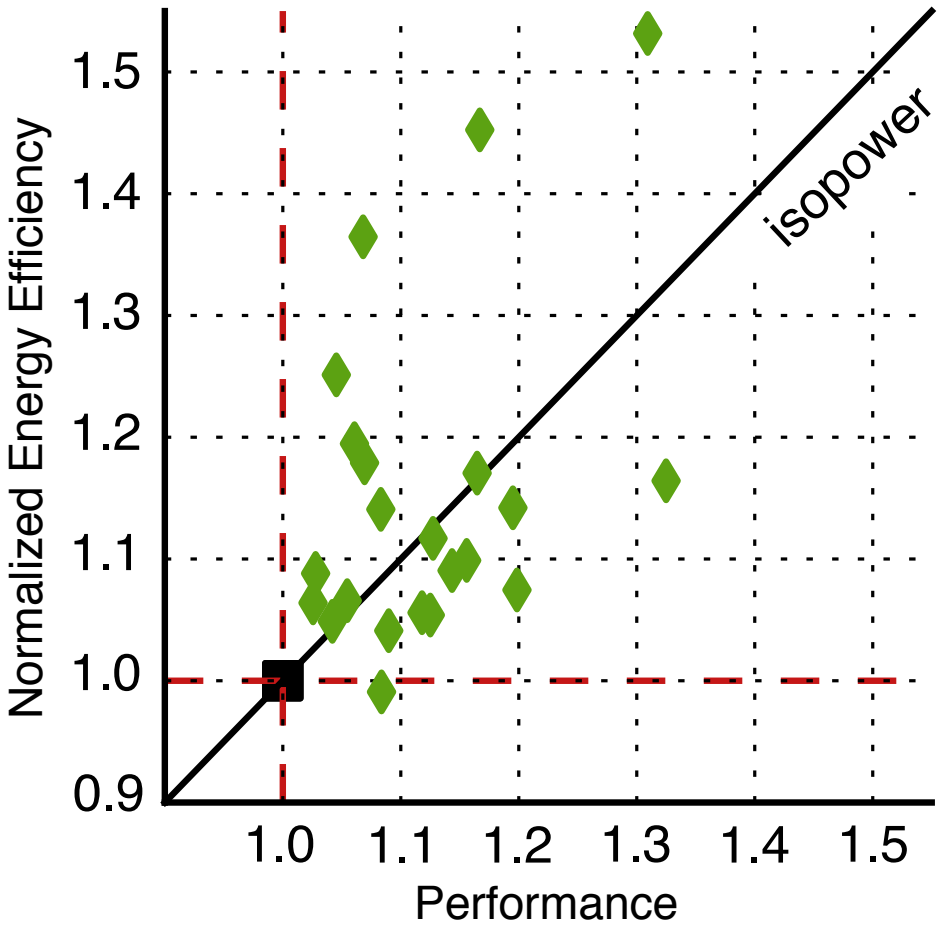




Work-Mugging in *radix sort*



Evaluation of Complete AAWS Runtime

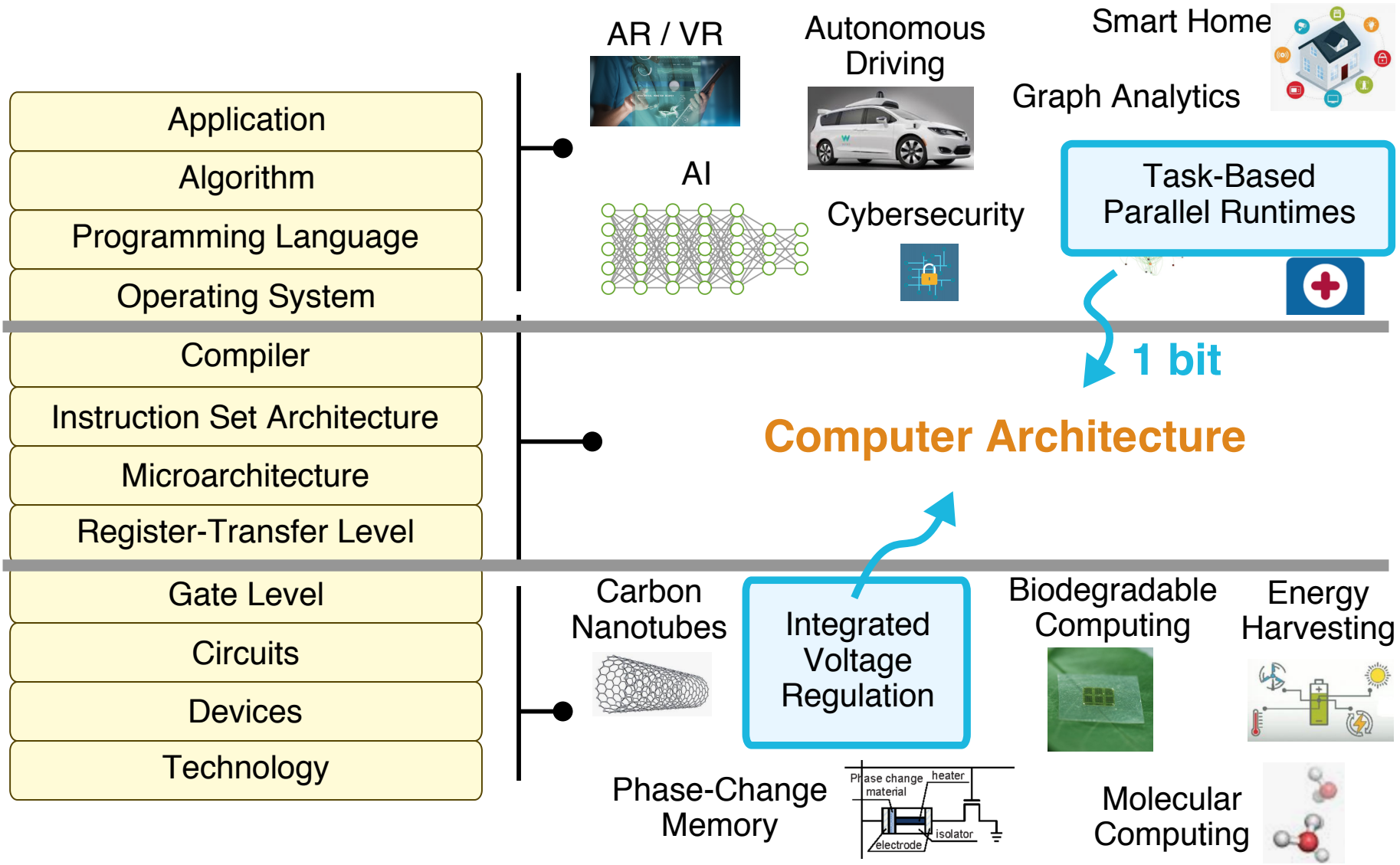


Application Kernels

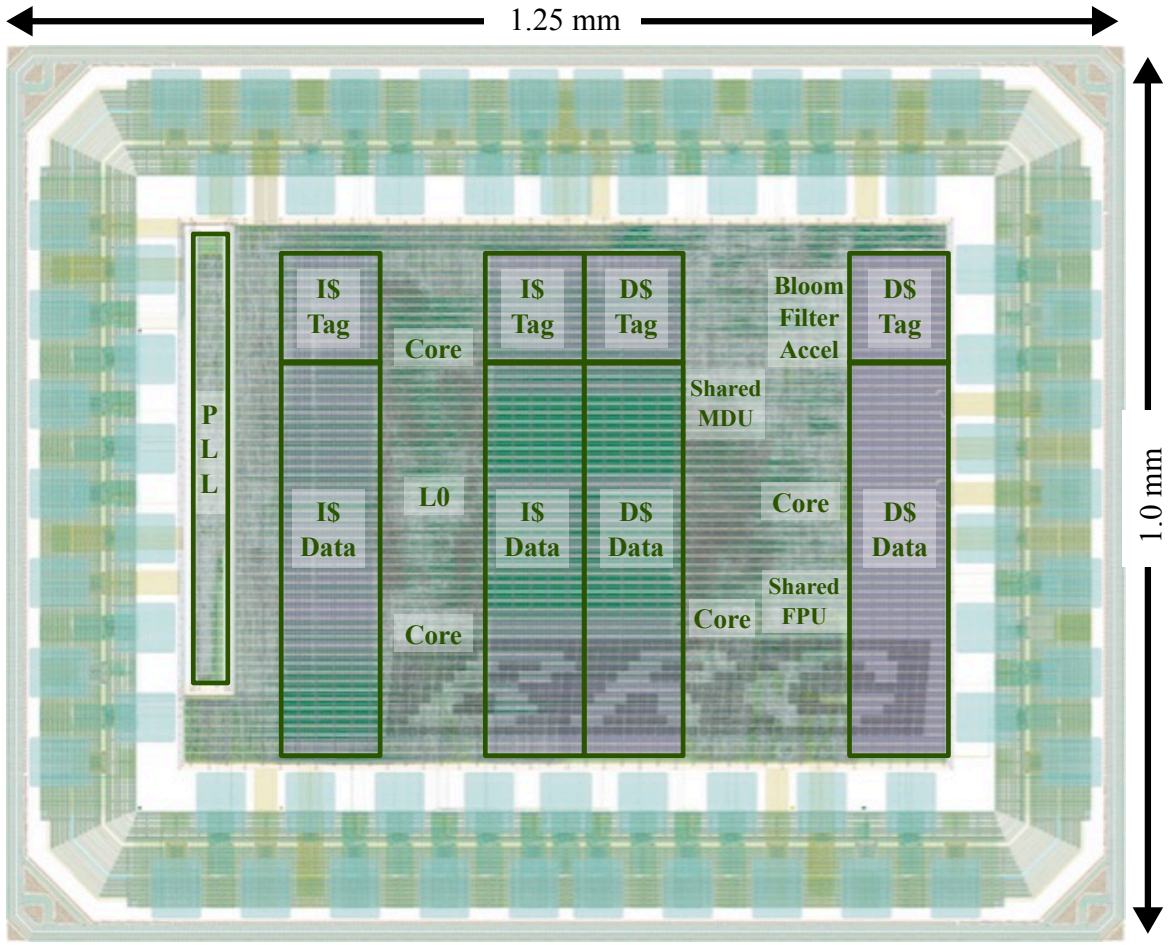
- pbbs-bfs
- pbbs-quicksort
- pbbs-samplesort
- pbbs-dictionary
- pbbs-convex-hull
- pbbs-radix-sort
- pbbs-knn
- pbbs-max-independent-set
- pbbs-nbody
- pbbs-remove-duplicates
- pbbs-suffix-array
- pbbs-spanning-tree
- cilk-cholesky
- cilk-cilksort
- cilk-heat
- cilk-knapsack
- cilk-matrix-multiply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

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

Popping Back Up a Level



Prototyping to Support Research Results

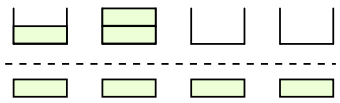


Batten Research Group Test Chip 2
Digital Test Chip, TSMC 28 nm
Static Timing Analysis @ 500 MHz

- Completed in 2 months
- Runs a work-stealing runtime (RISC-V XCC)
- Four RISC-V cores + 32kB L1 caches
- Aggressively shared long-latency resources
- Microarchitectural smart sharing mechanisms
- Synthesizable PLL

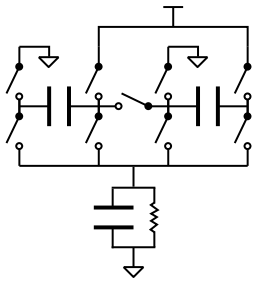
Results supporting multiple paper submissions to top-tier venues

Building Future Computing Systems that Bridge Software, Architecture, and VLSI



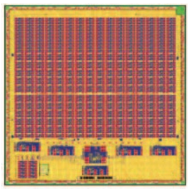
Cross-Stack Co-Design for

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



Cross-Stack Co-Design for

Integrated Voltage Regulation - MICRO'14, IEEE TCAS I'18

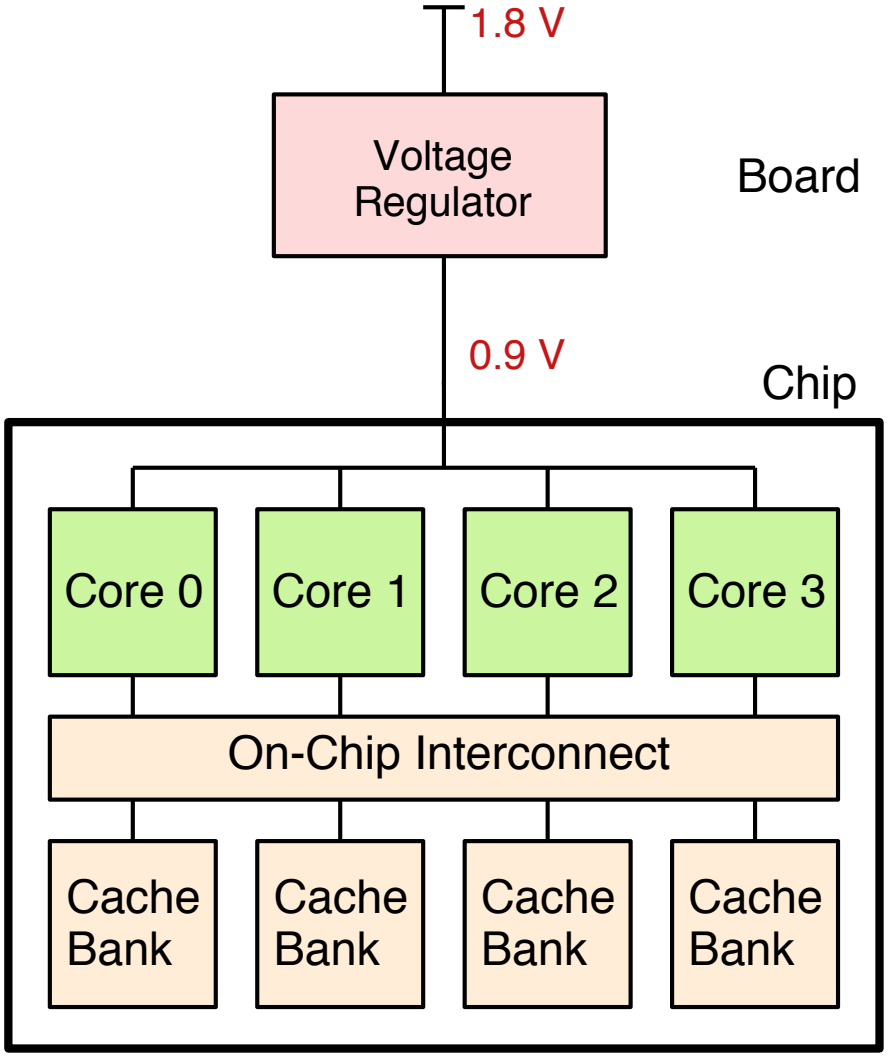


Cross-Stack Co-Design for

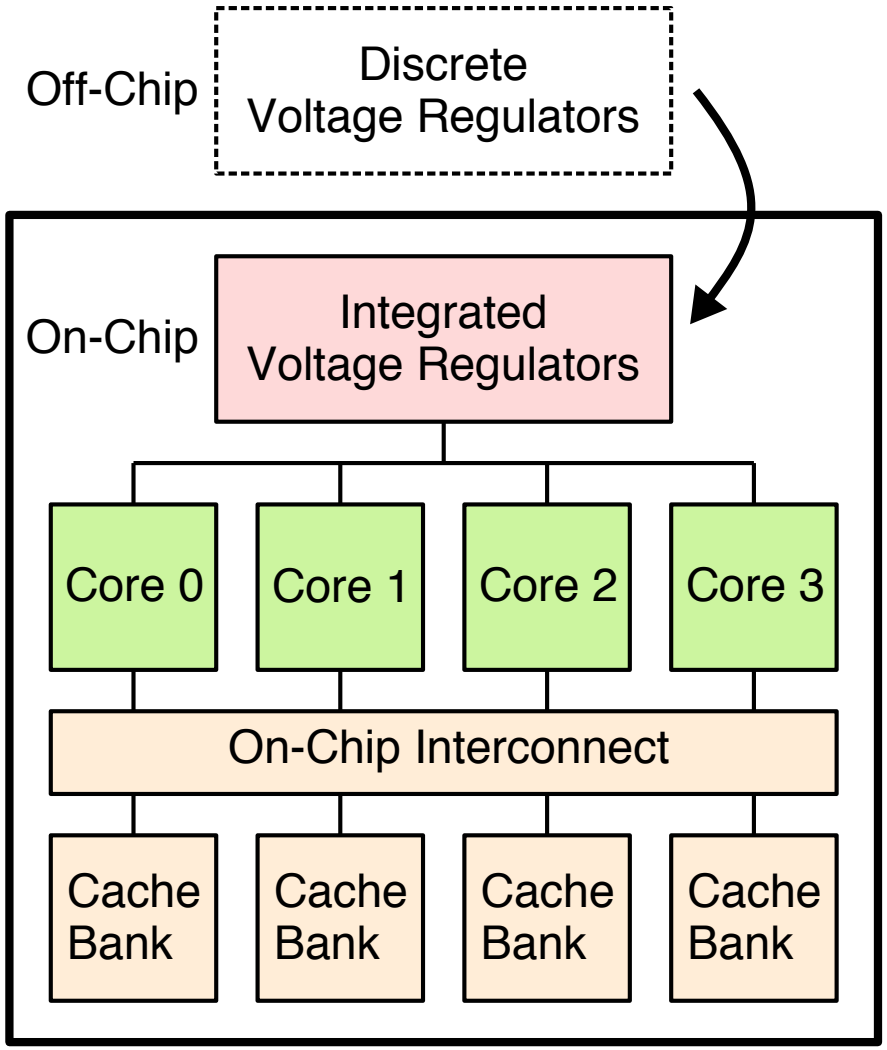
Rapid ASIC Design - IEEE MICRO'18, DAC'18, Hotchips'17

Future Research

What is a Voltage Regulator?



Why is Integrated Voltage Regulation Important?



Key Benefit of IVR

- Reduced System Cost

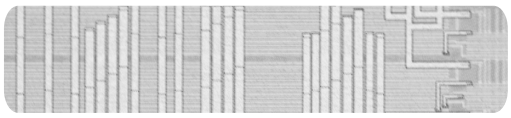
Why Integrate Now?

- Technology scaling.. on-chip switches and capacitors have gotten better

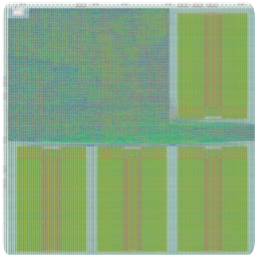
What's the Problem?

- Integrated voltage regulators are BIG

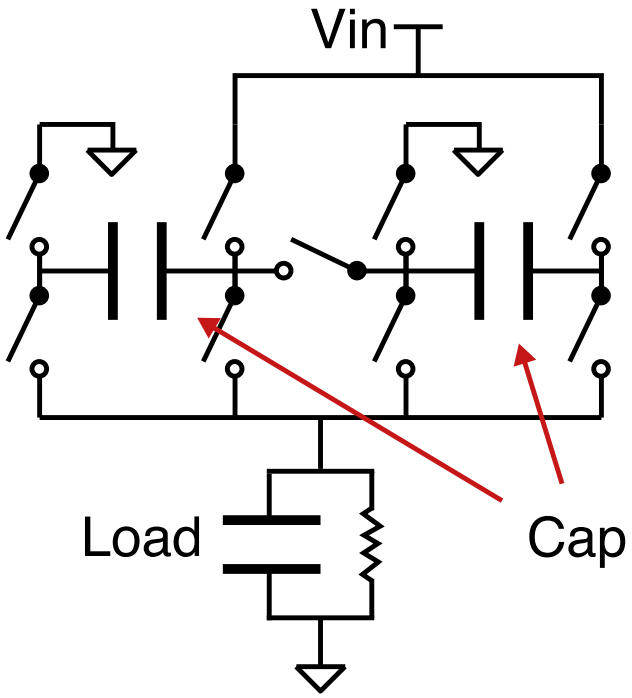
Technology-Normalized Integrated Voltage Regulator



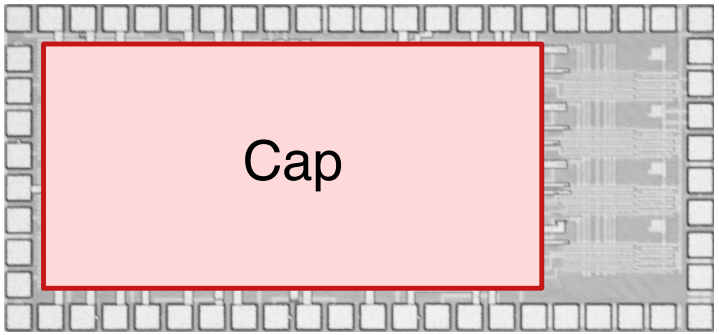
Technology-Normalized Simple RISC Core



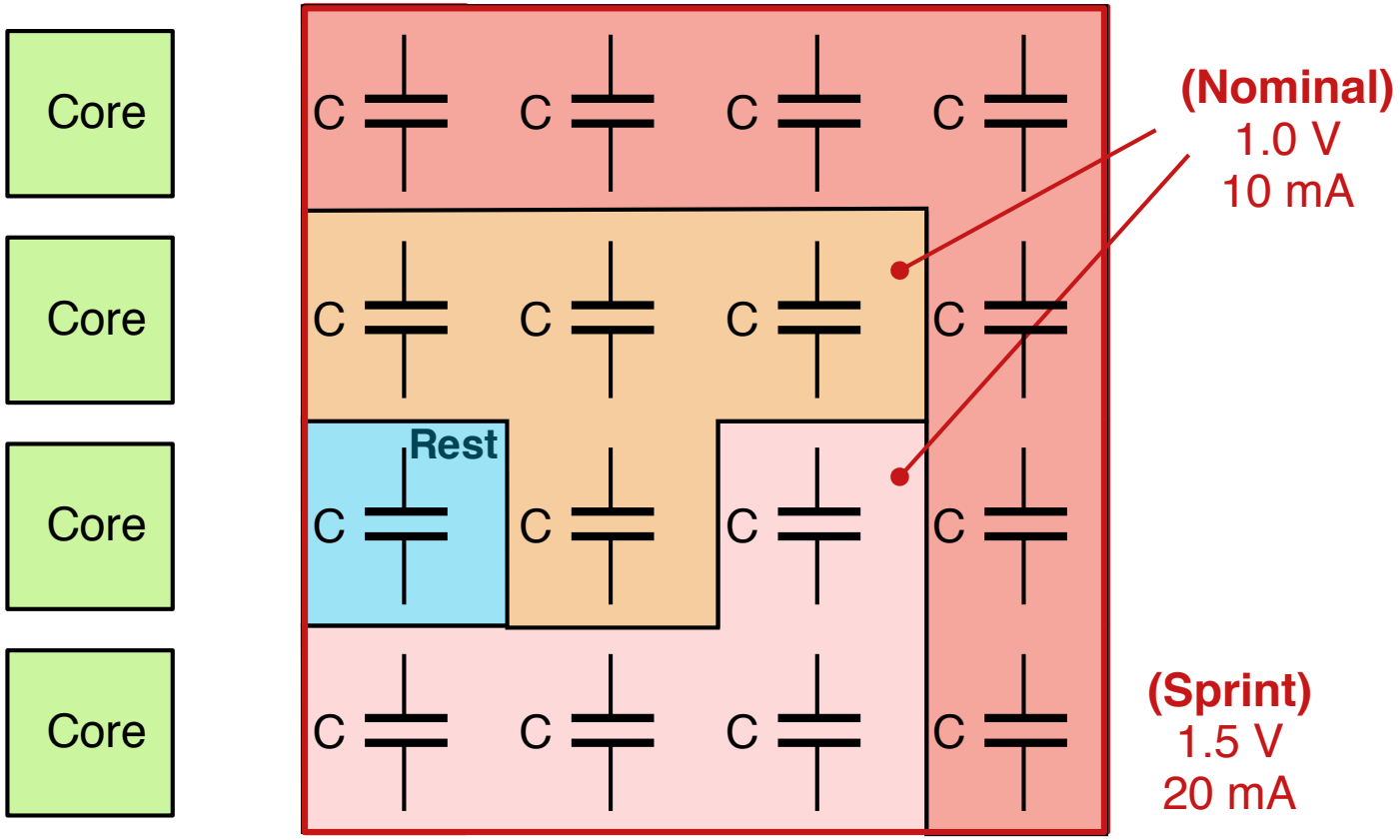
What's taking all that area?



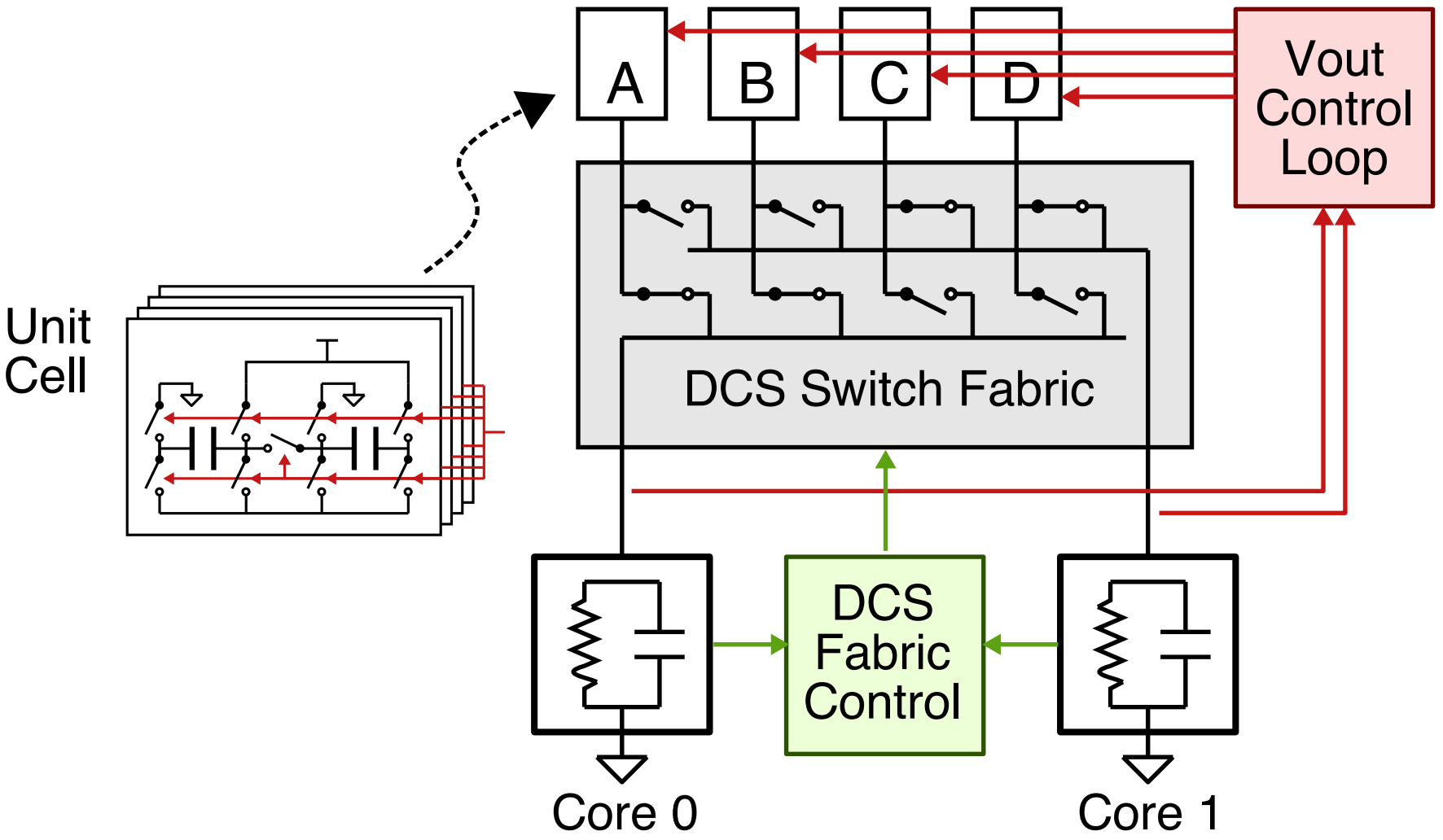
IVR Test Chip



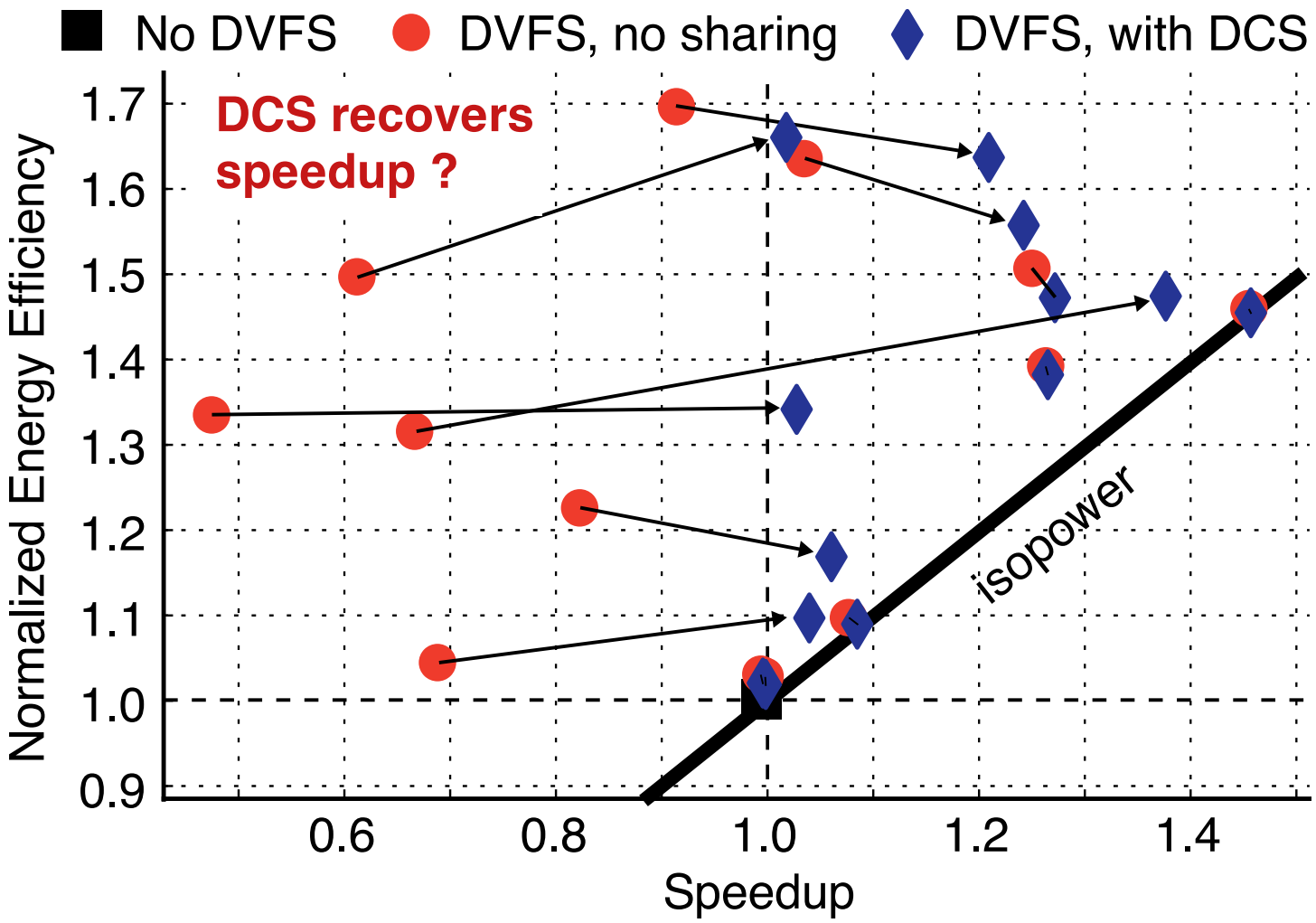
Key Architecture-Level Intuition



Key Idea: Dynamic Capacitance Sharing



Evaluation

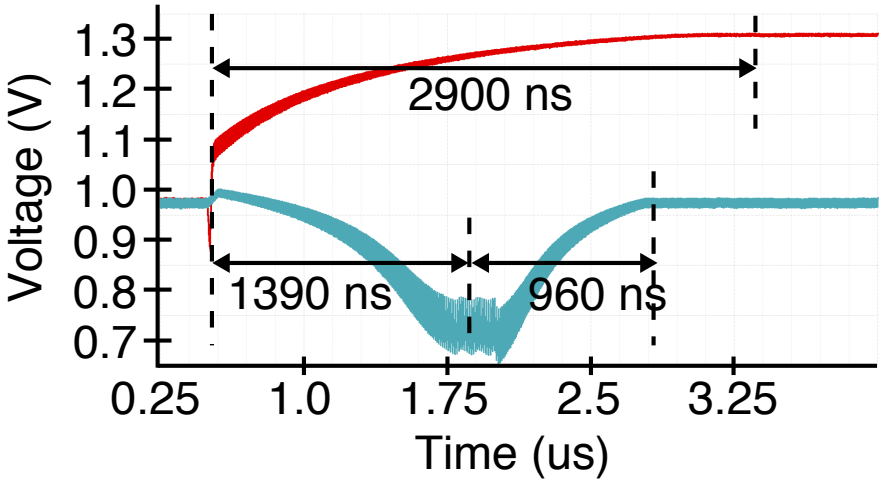


Benchmarks

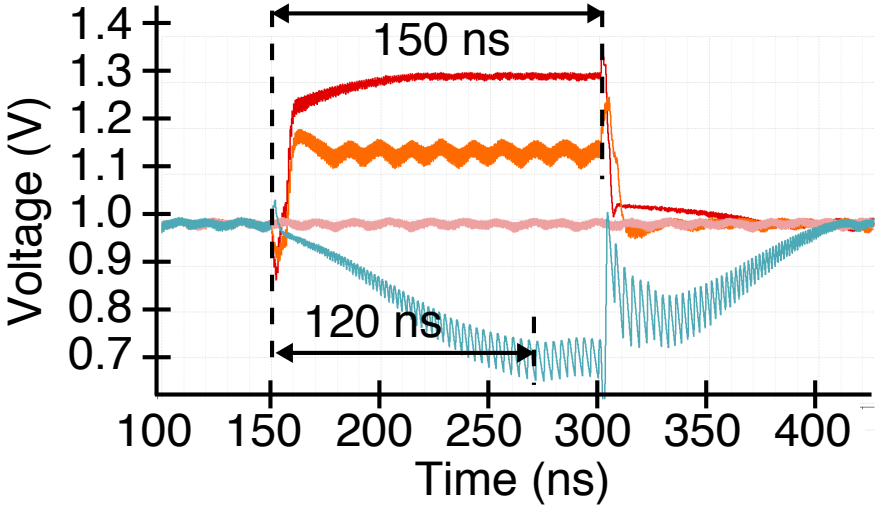
- bfs
- **bilateral**
- dither
- kmeans
- **mriq**
- pbbs-dr
- **pbbs-knn**
- pbbs-mm
- rsort
- **splash2-fft**
- **splash2-lu**
- **strsearch**
- viterbi
- bfs
- bilateral
- dither

SPICE-Level Transient Response

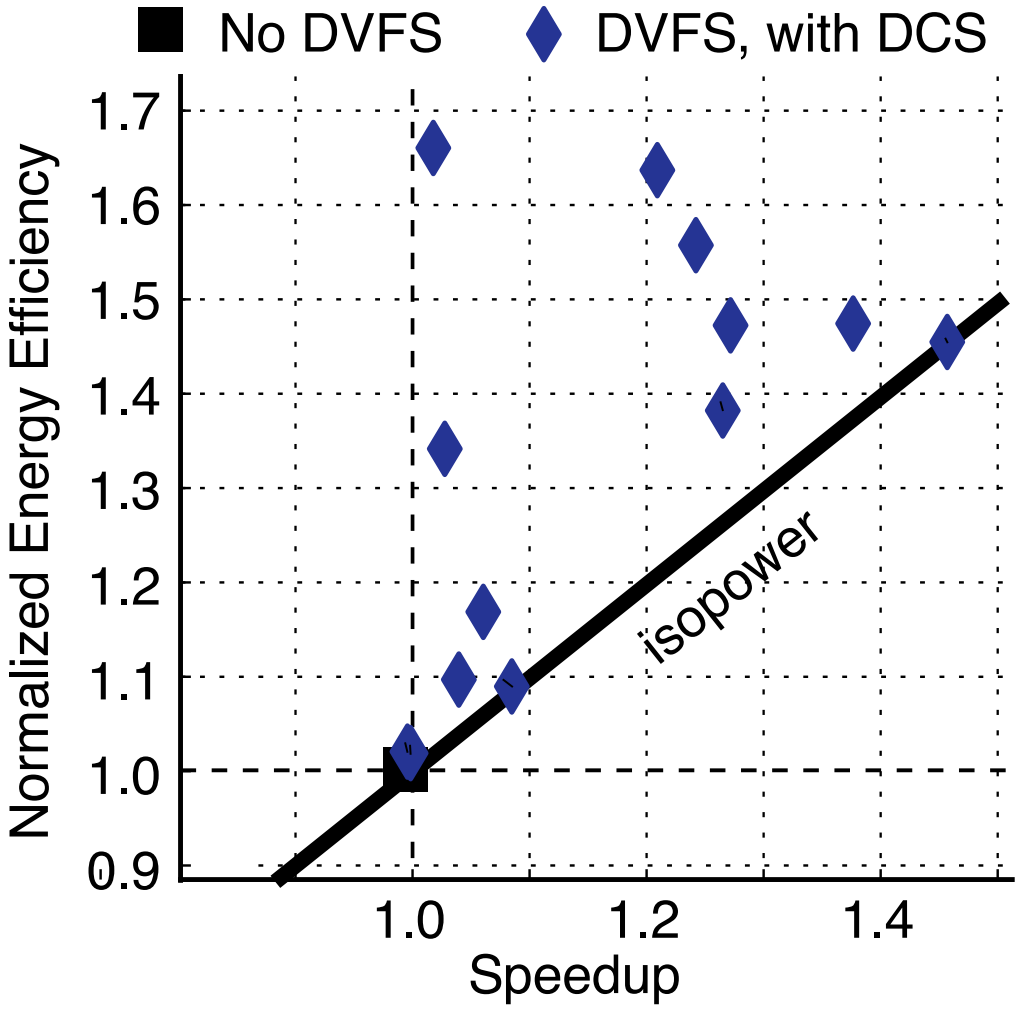
Transient Response
DVFS, no sharing



Transient Response
DVFS, with DCS



Evaluation

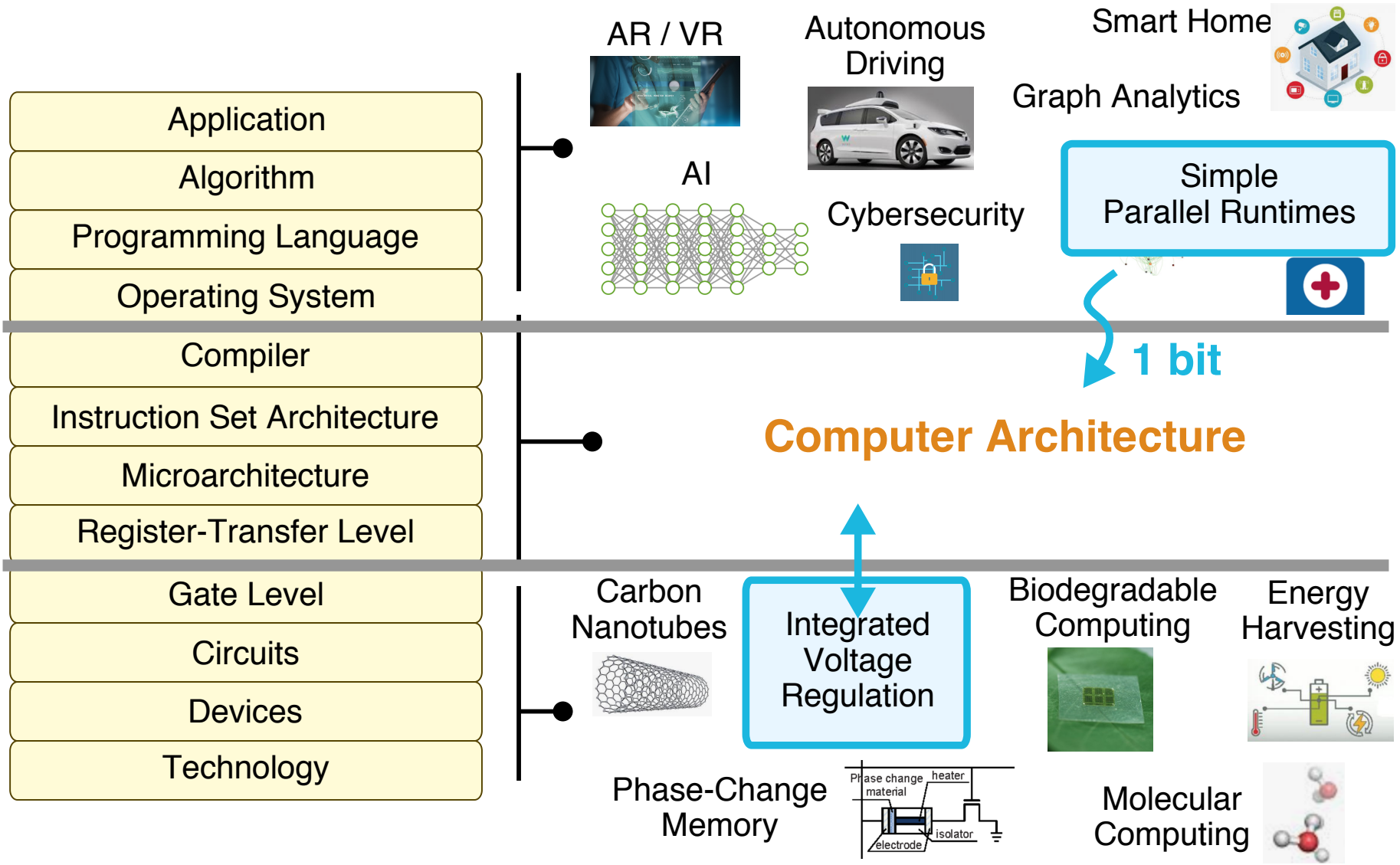


10-50% Speedup and 10-70% Energy Efficiency with Area-Optimized On-Chip VRs

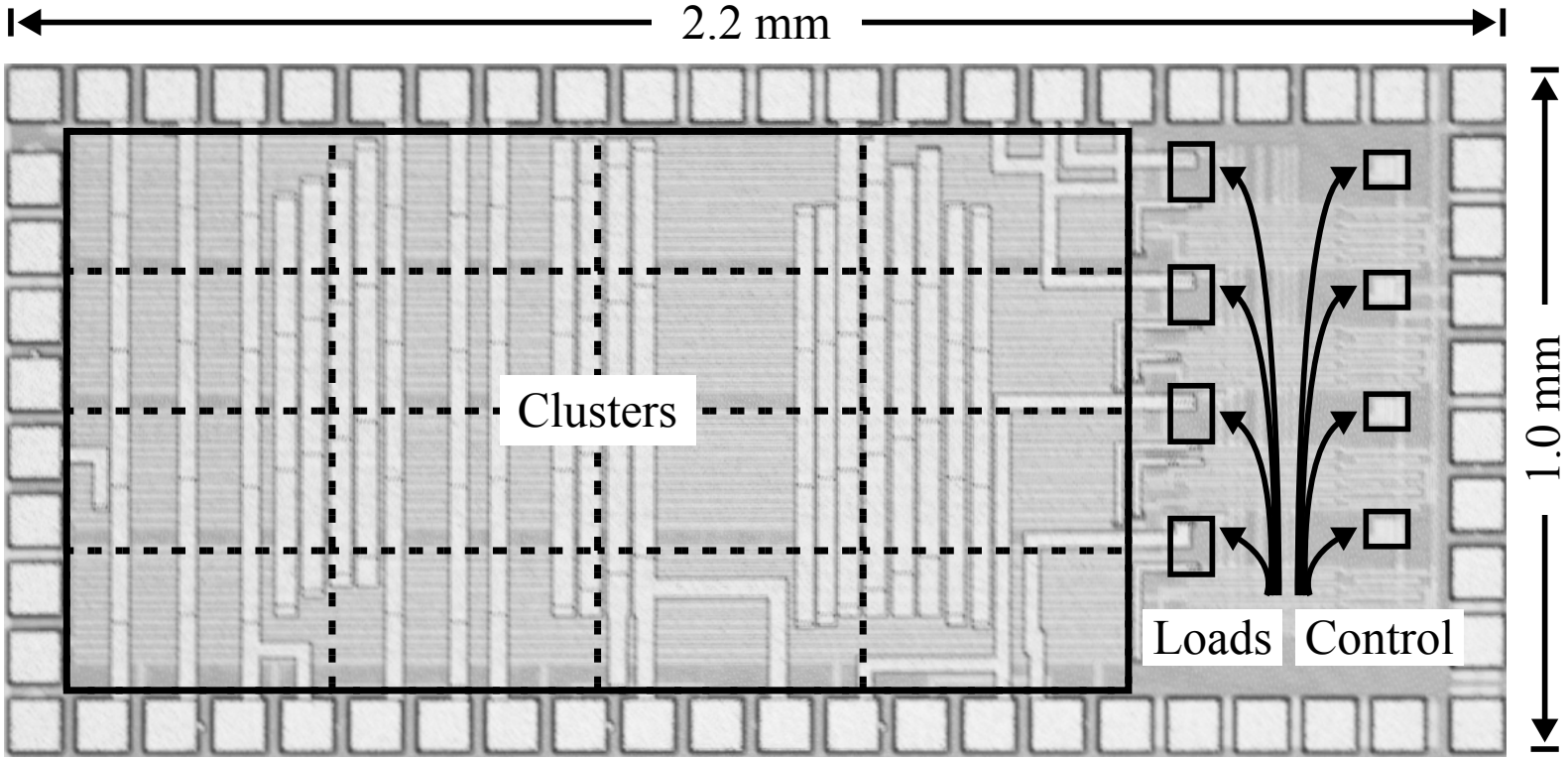
Benchmarks

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

Popping Back Up a Level



Dynamic Capacitance Sharing Analog Test Chip

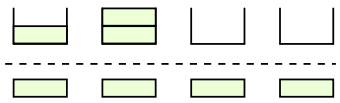


Four monolithically integrated switched-capacitor DC-DC converters with the **dynamic capacitance sharing** technique in 65 nm CMOS

Collaboration with Waclaw Godycki, Ivan Bukreyev, and Professor Alyssa Apsel

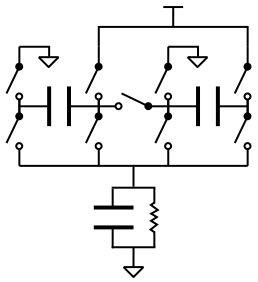
[MICRO 2014] [IEEE TCAS I 2018]

Building Future Computing Systems that Bridge Software, Architecture, and VLSI



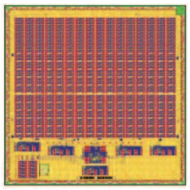
Cross-Stack Co-Design for

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



Cross-Stack Co-Design for

Integrated Voltage Regulation - MICRO'14, IEEE TCAS I'18

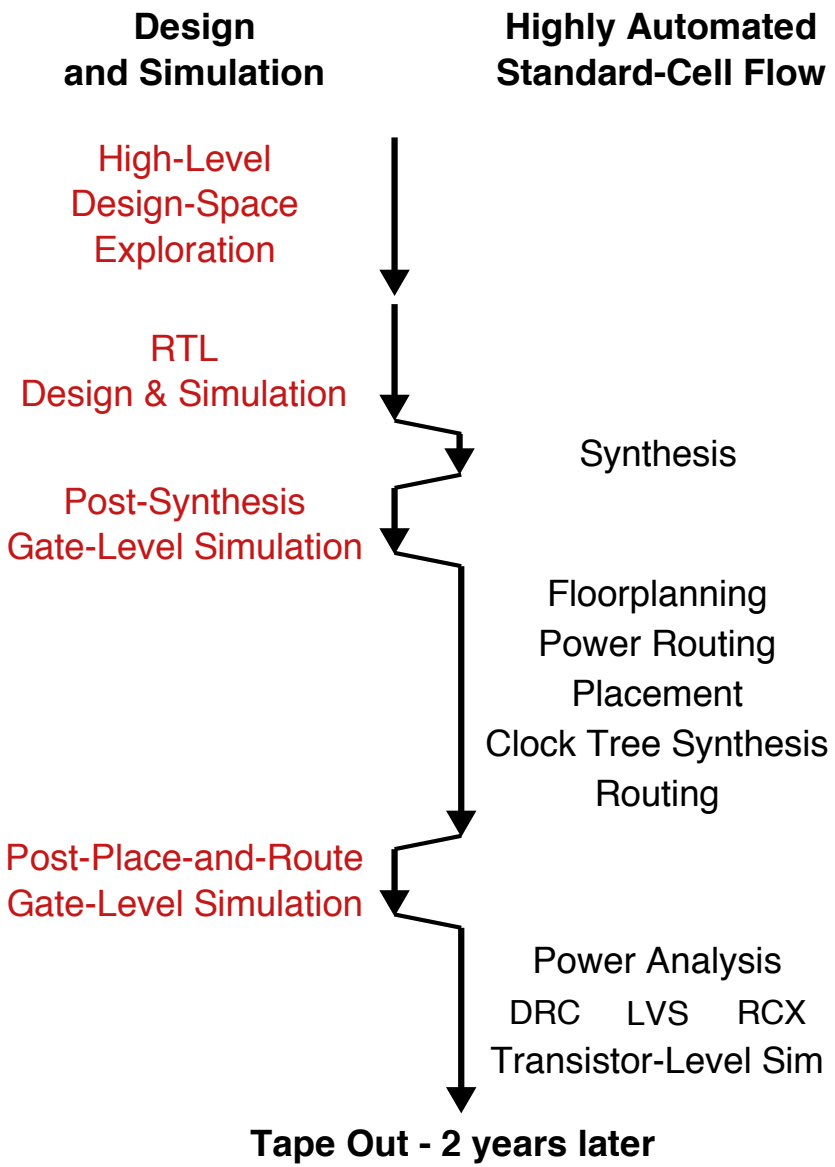


Cross-Stack Co-Design for

Rapid ASIC Design - IEEE MICRO'18, DAC'18, Hotchips'17

Future Research

Challenges in Building ASIC Prototypes



Costly in terms of...

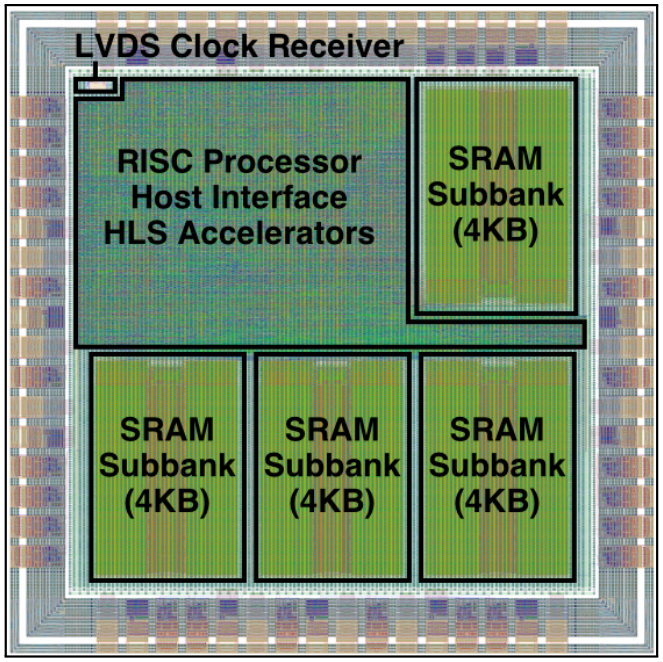
- $$ to fabricate
- $$ for licensing of IP
- $$ for expertise and workforce

Time and effort...

- RTL design and verification
- ASIC frontend / ASIC backend
  - Weeks to months for one iteration
- 1.5 year timeline is too late!

By the end, the frontier of the accelerator design space will have already moved on...

PyMTL ASIC Tapeouts

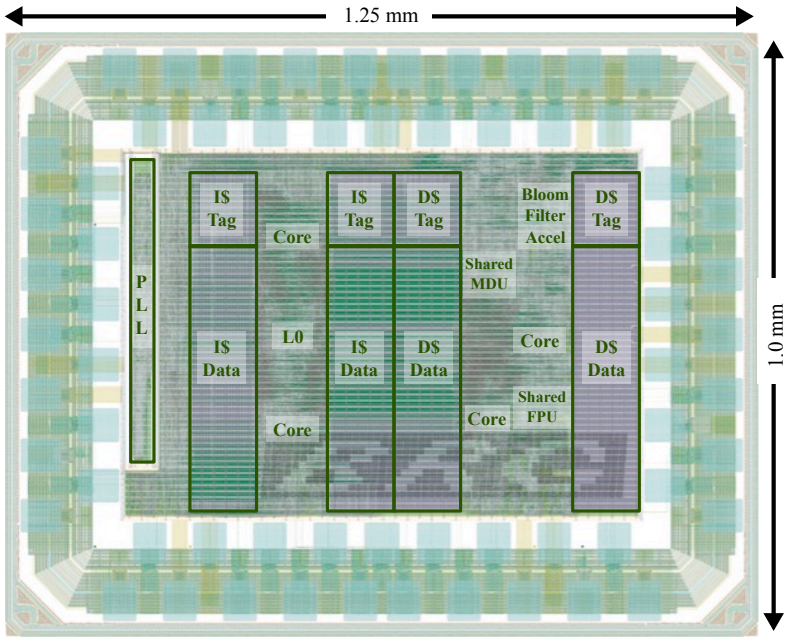


BRGTC1 in 2016

RISC processor, 16KB SRAM
HLS-generated accelerator

2x2mm, 1.2M-trans, IBM 130nm

[Poster at Hotchips 2016]



BRGTC2 in 2018

Four RISC-V cores with “smart”
sharing L1$/LLFU, PLL

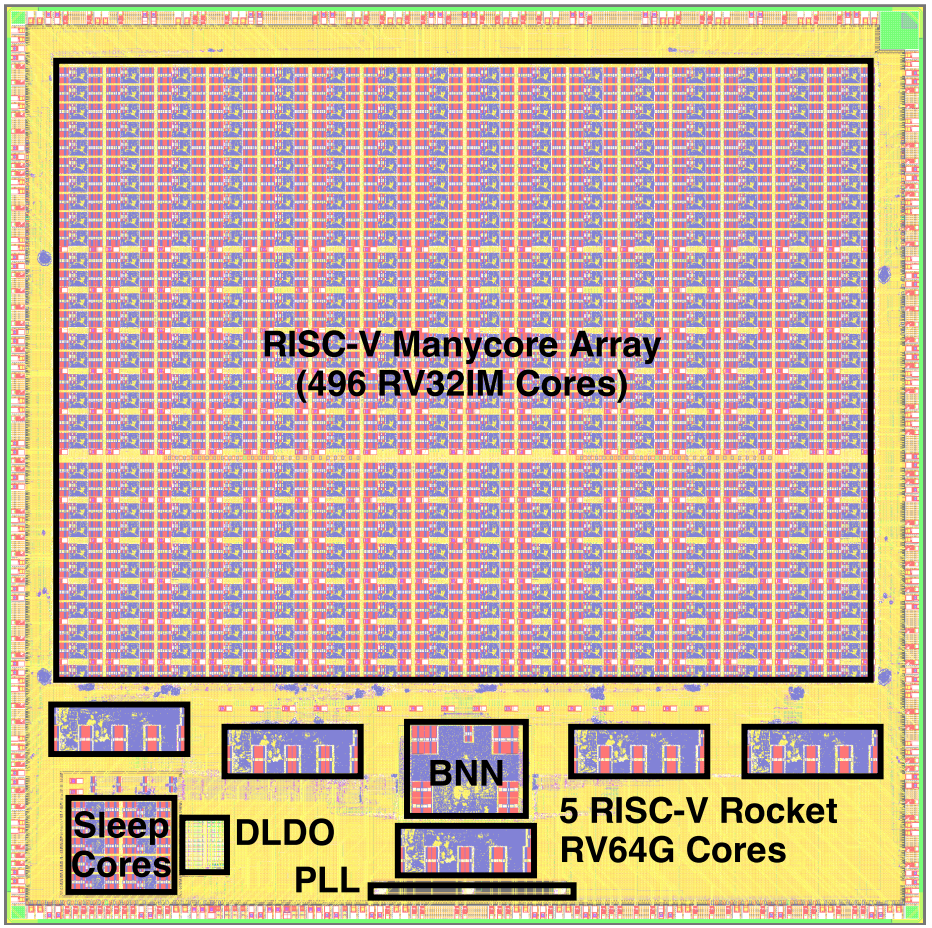
1x1.2mm, ≈10M-trans, TSMC 28nm

[RISCV 2018]

The Celerity SoC

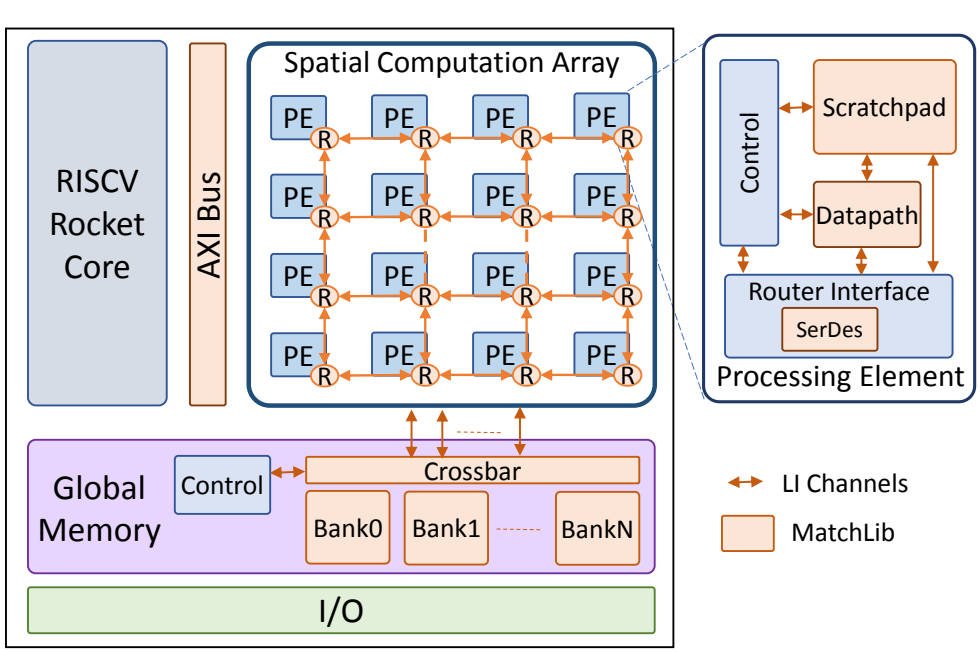
Target Workload: High-Performance Embedded Computing

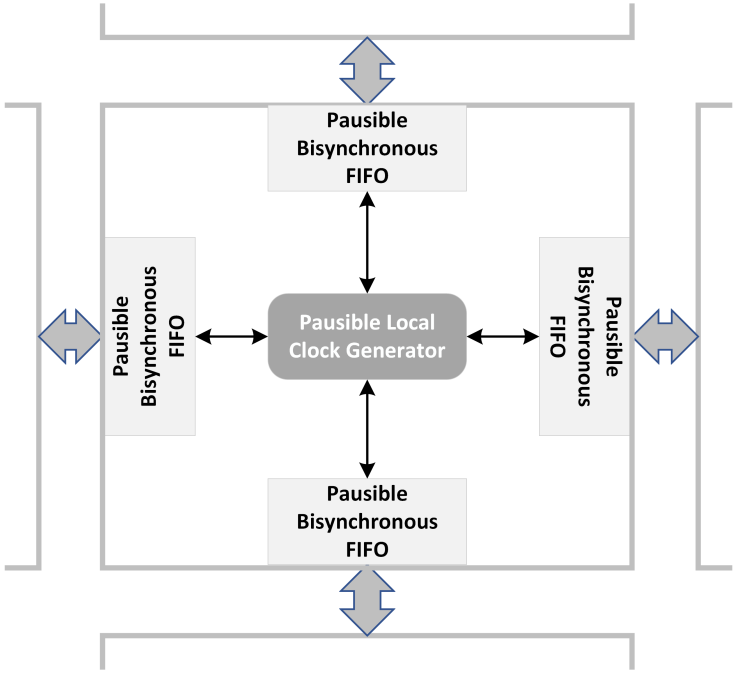
- Multiple universities under *DARPA CRAFT*
- 5 × 5mm in TSMC 16 nm FFC
- 385 million transistors
- 511 RISC-V cores
  - 5 Linux-capable Rocket cores
  - 496-core tiled manycore
  - 10-core low-voltage array
- 1 BNN accelerator
- 1 synthesizable PLL
- 1 synthesizable LDO Vreg
- 3 clock domains
- 672-pin flip chip BGA package
- 9-months from PDK access to tape-out



[Hotchips 2017] [IEEE MICRO 2018]

High-Productivity SoC Design Flow based on HLS



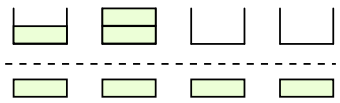


Internship at NVIDIA Research in Summer'17, led by Brucek Khailany

Lightly involved in their MatchLib project under DARPA CRAFT

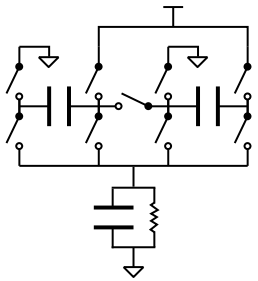
[DAC 2018]

Building Future Computing Systems that Bridge Software, Architecture, and VLSI



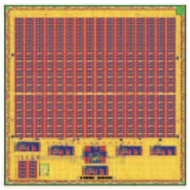
Cross-Stack Co-Design for

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



Cross-Stack Co-Design for

Integrated Voltage Regulation - MICRO'14, IEEE TCAS I'18



Cross-Stack Co-Design for

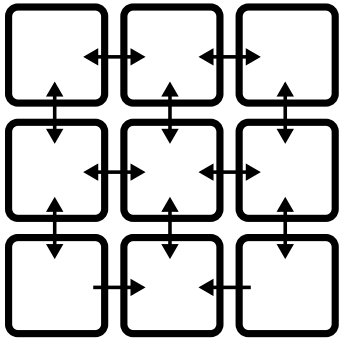
Rapid ASIC Design - IEEE MICRO'18, DAC'18, Hotchips'17

Future Research

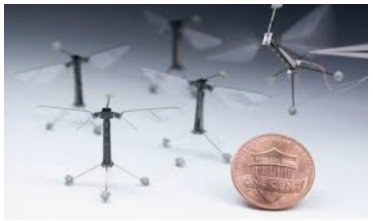
Future Research

Apply a **cross-stack research approach** to many different problems using a **vertically integrated methodology**

Intelligence on the Edge

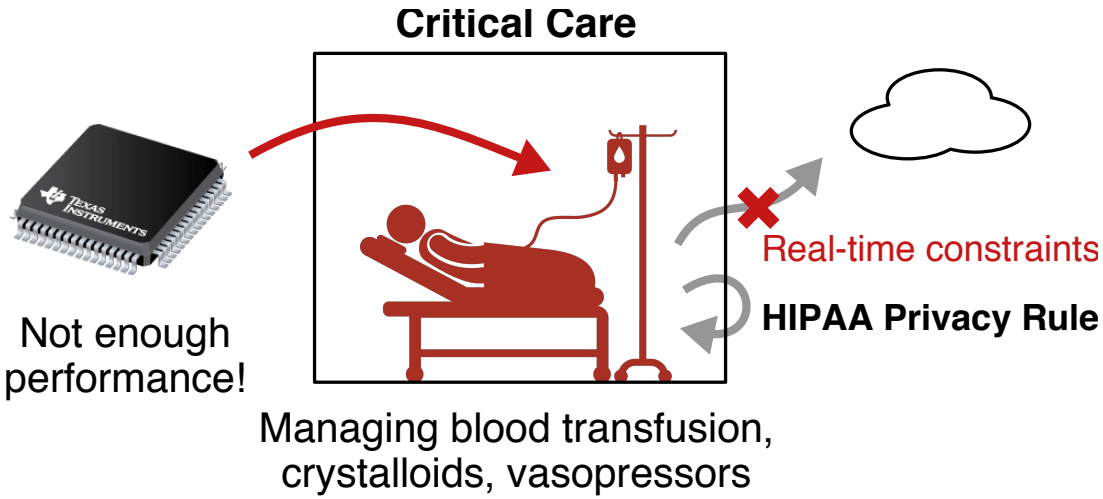


Tiling-Based Designs

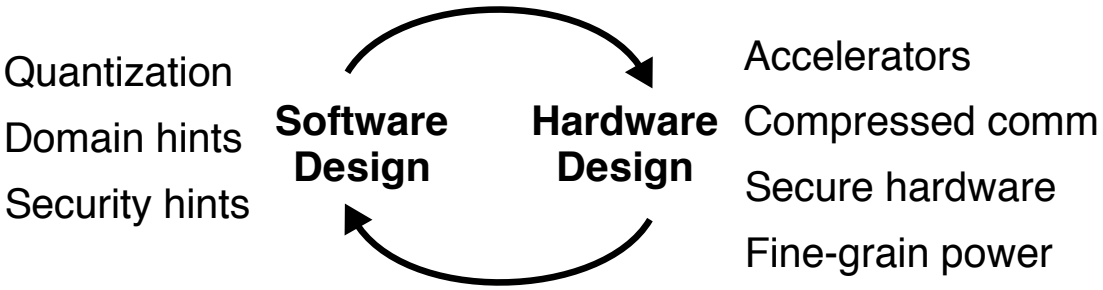


Cyber-Physical Systems

Cross-Stack Co-Design for IoT on the Edge



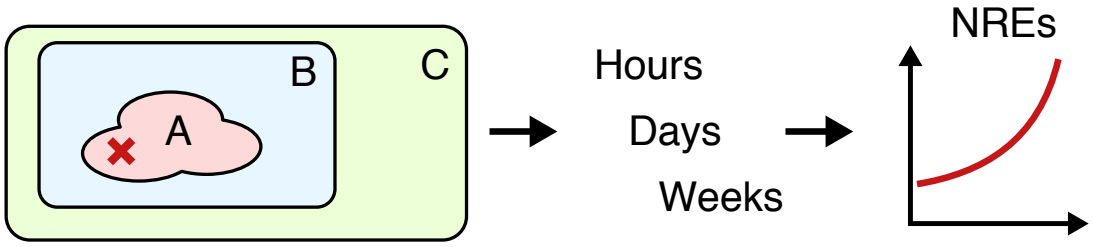
Specialize



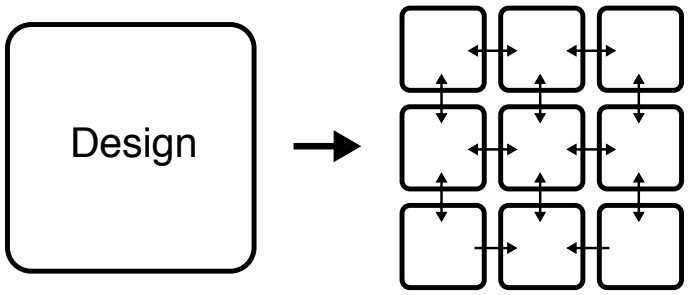
Software-Hardware Co-Design

- Intelligence on the edge is **required** in many different application domains
- We need **specialization**
- **Concretely**: Build new accelerator-centric SoCs to enable new applications
- **Pipe dreams**
  - Emerging applications: smart healthcare + infra
  - Emerging technologies: hybrid CMOS-TFET, emerging memories
- **Modular** system design

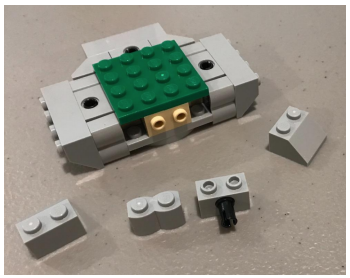
Cross-Stack Co-Design for Tiling Designs



- We need to build more hardware and make hardware **easier to build**

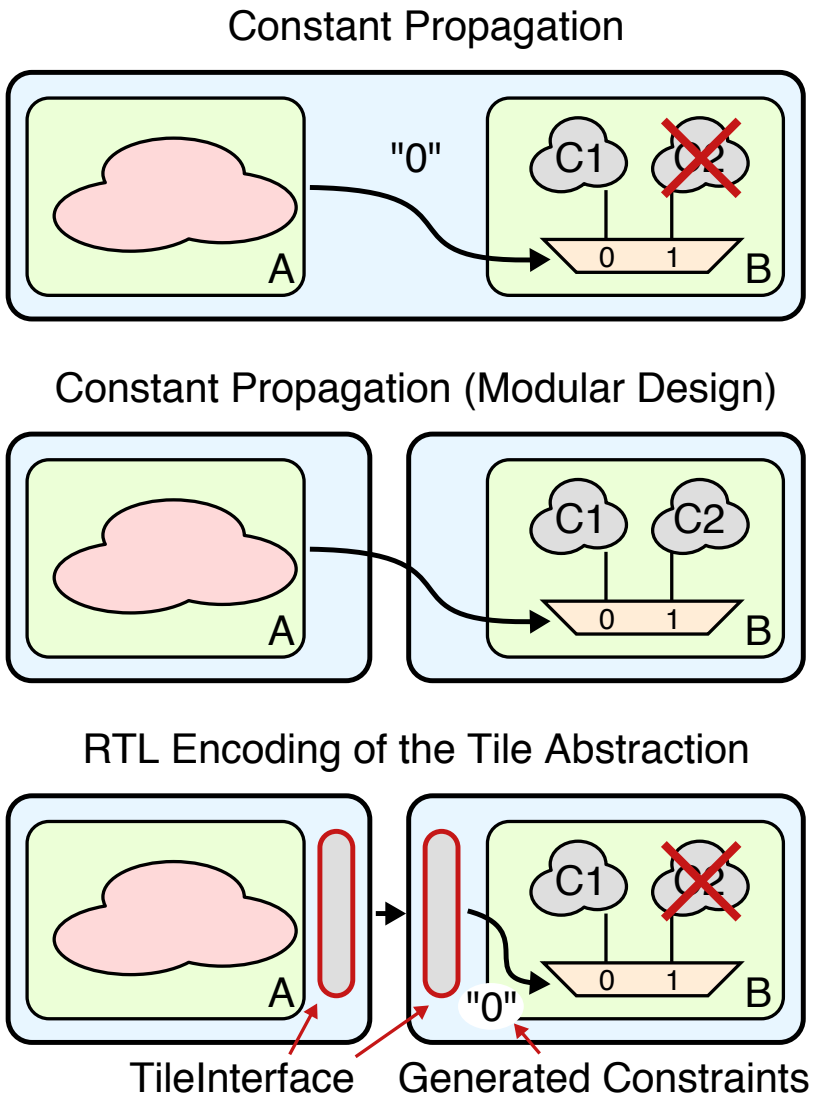


- Do our best to **avoid monolithic designs**
  - GALS – pre-silicon
  - Chiplets – post-silicon



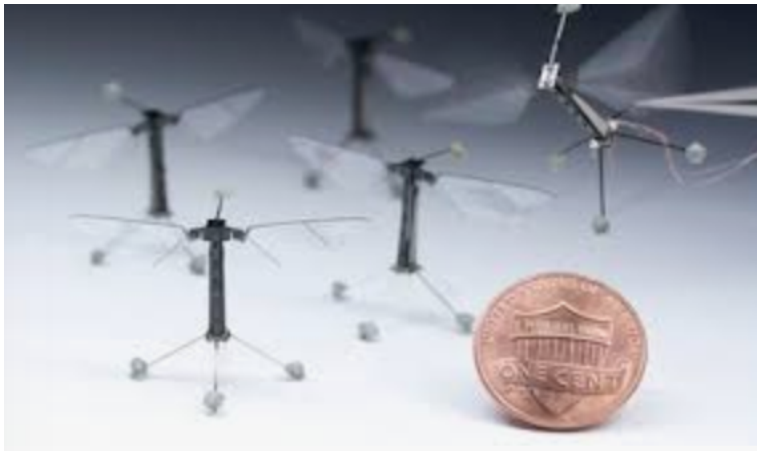
- **Concretely**: Build new accelerator-centric SoCs with a methodology that extends the tiling abstraction across the stack

Cross-Stack Co-Design for Tiling Designs



- We need to build more hardware and make hardware **easier to build**
- Do our best to **avoid monolithic designs**
  - GALS – pre-silicon
  - Chiplets – post-silicon
- **Concretely**: Build new accelerator-centric SoCs with a methodology that extends the tiling abstraction across the stack

Cross-Stack Co-Design with Cyber-Physical Systems

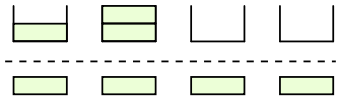


- **Concretely**: Explore new SoCs that can be embedded into cyber-physical systems
- **Pipe dreams**
  - Inspired by projects like Harvard RoboBee
  - Architectures + cyber-physical systems where custom acceleration and silicon prototyping can make a real difference

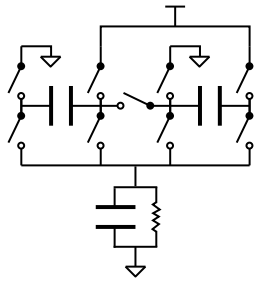
Acknowledgements and Funding

- **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, 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

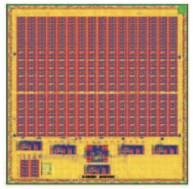


Emerging new contexts demand much higher performance and energy efficiency



Cross-stack co-design can make a real impact

- More efficient *task-based parallel runtimes*
- Smaller area for *integrated voltage regulation*
- Better methodologies for *rapid ASIC design*



I am excited to **explore future cross-stack research** applied to **intelligence on the edge**, driving methodologies for **tiling-based designs**, and also supporting **cyber-physical systems**.