SOFTWARE DESIGN FOR EMBEDDED HETEROGENEOUS SOC

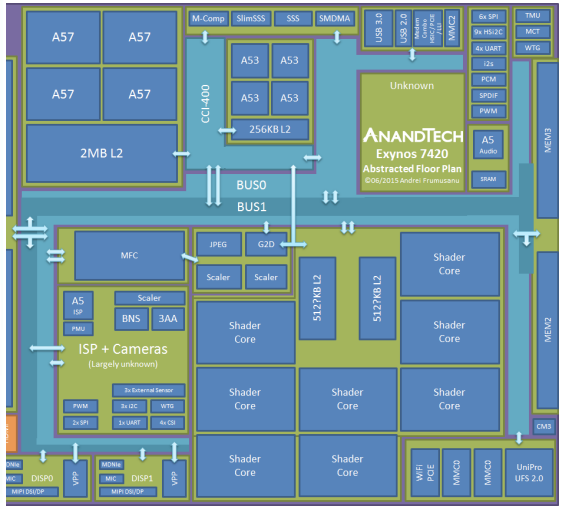
Tulika Mitra

National University of Singapore

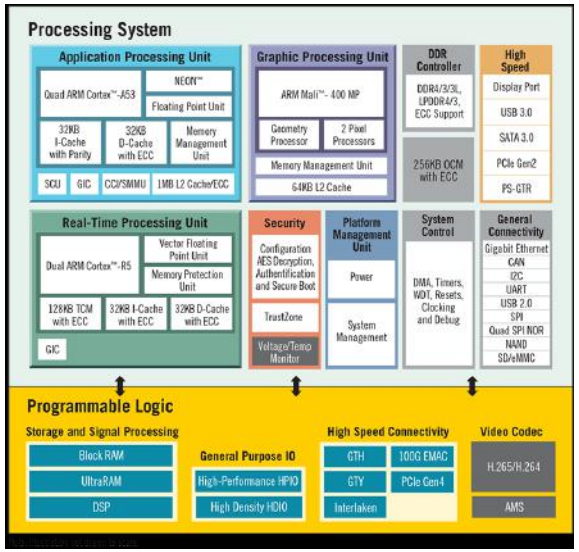
Embedded Heterogeneous Computing

- A System-on-a-Chip (SoC) comprising of heterogeneous computing elements, such as different CPU cores, GPU cores, DSP cores, and accelerators

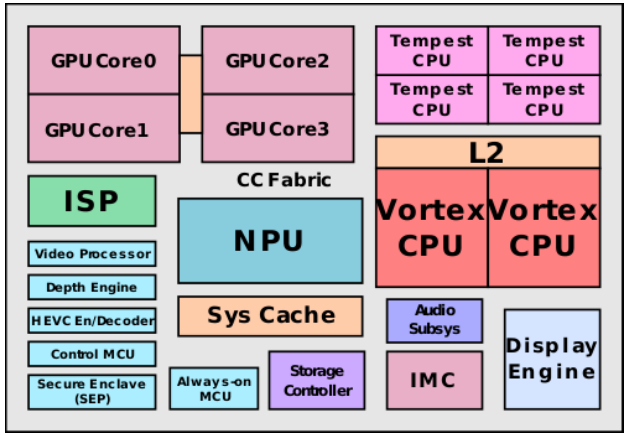
Qualcomm Snapdragon 835



Samsung Exynos 7420



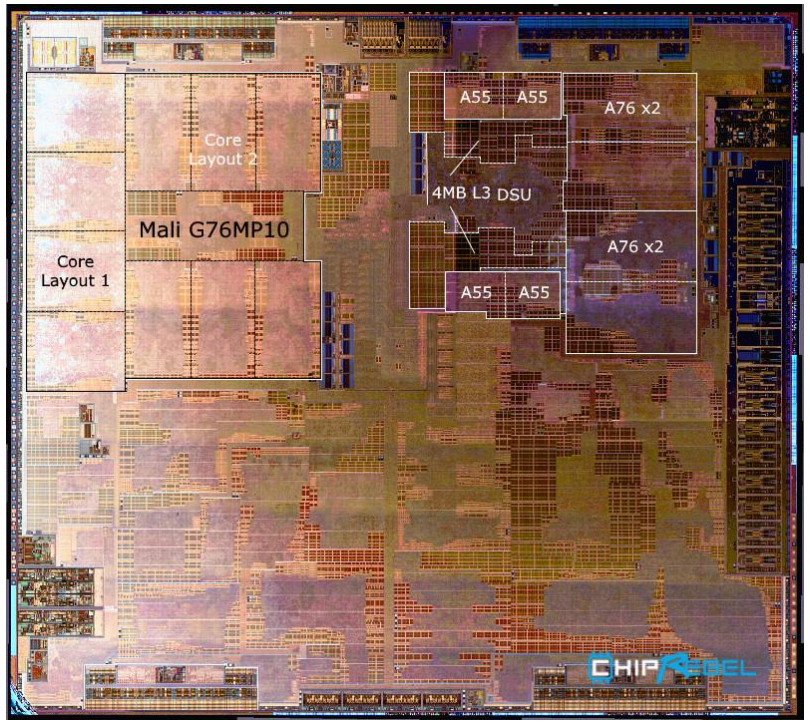
Xilinx Zynq Ultrascale+



Apple A12 Bionic

Image credit: Qualcomm, AnandTech, Xilinx, Wikichip

Mobile Application processor



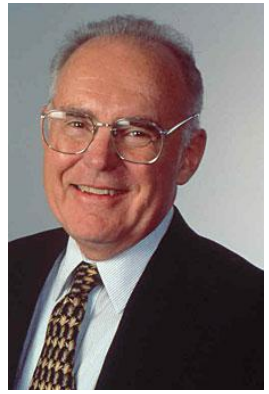
HiSilicon Kirin 980 Die-photo

HiSilicon Kirin 980

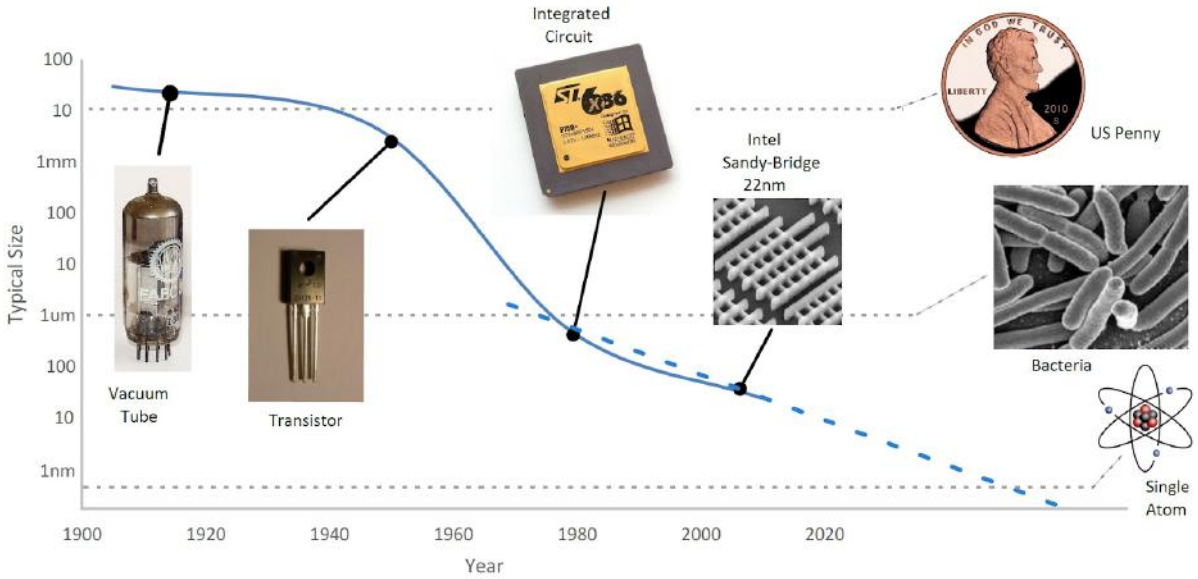
Huawei Mate 20

Image credit: AnandTech, Chip Rebel, Huawei

Moore's Law



- Prediction by Intel co-founder Gordon Moore in 1965:
- Transistor density in an integrated circuit will double every year or two
- Transistor's linear dimension shrinks by a factor of λ, area shrinks by a factor of λ²
  Number of transistors increase by a factor of λ² per technology generation



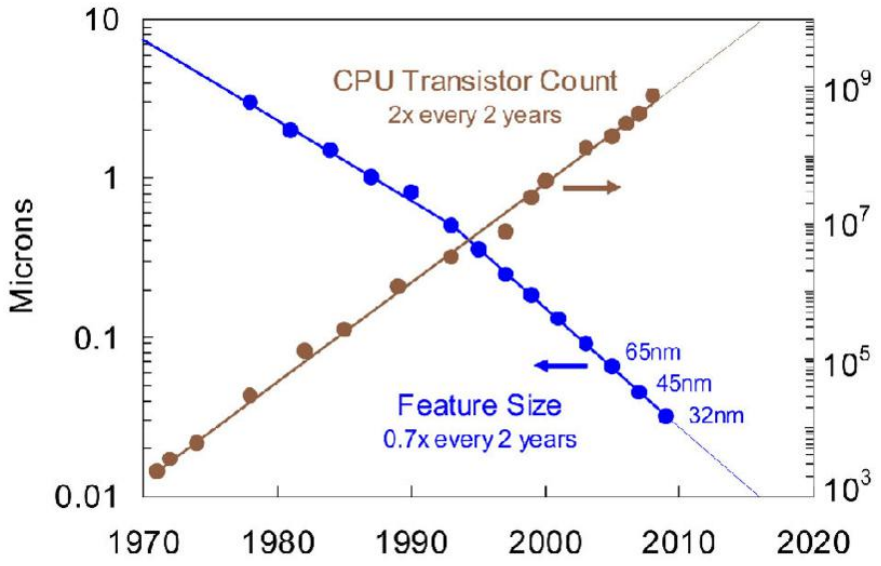


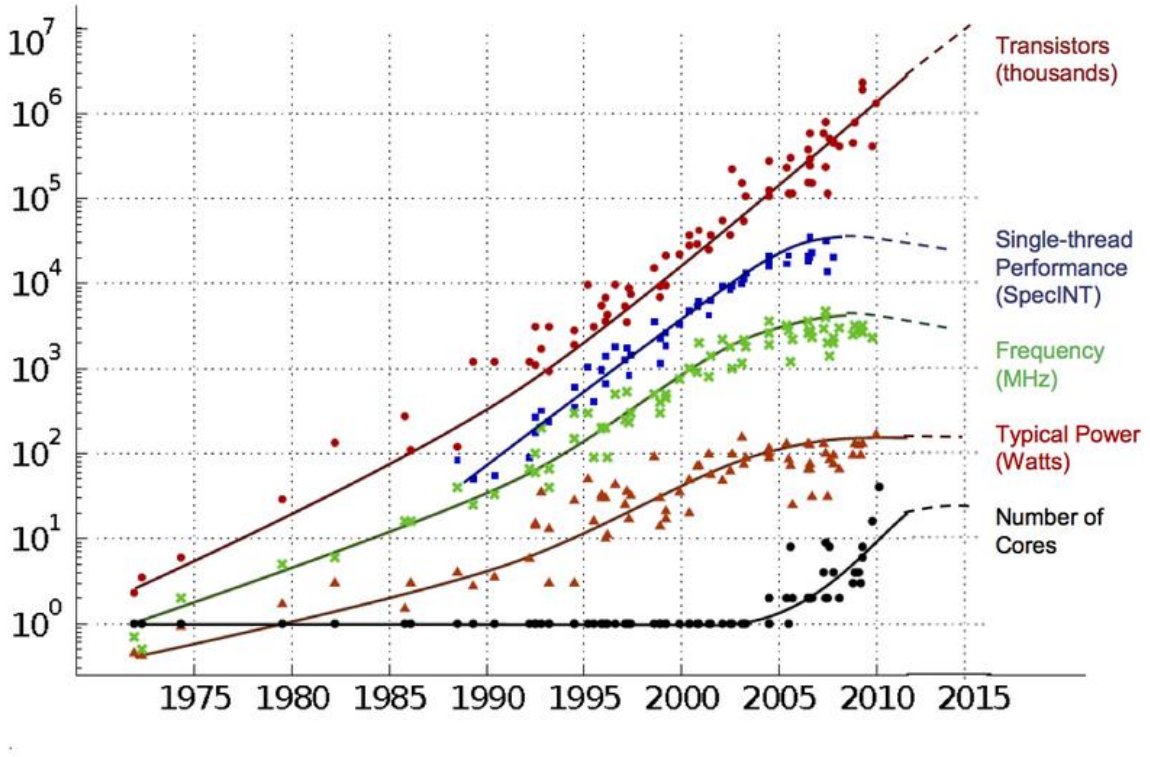
Image credit: Soham Chatterjee, KJ Kuhn

Dennard Scaling

- Complementary to Moore's Law formulated by Robert Dennard in 1974
- As transistors get smaller, their power density stays constant
- Transistor's linear dimension scaling factor: $\frac{1}{\lambda}$ ($\approx 0.7\text{x}$ per technology generation)
- Transistor area scaling factor: $\frac{1}{\lambda^2}$ ($\approx 0.5\text{x}$ per technology generation)
- Both current and voltage scaling factor: $\frac{1}{\lambda}$
- Transistor frequency scaling factor: λ
- Power dissipation scaling factor: $\frac{1}{\lambda^2}$
- Power density scaling factor: 1

End of Dennard Scaling

- Dennard scaling broke down around 2004 with unscaled interconnect delay and our inability to scale down voltage and current due to reliability concerns



Original data collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond and C. Batten
Dotted line extrapolations by C. Moore

Transition to Multi- and Many- Cores

- End of frequency scaling
- Increased performance through multi- and many-core parallelism

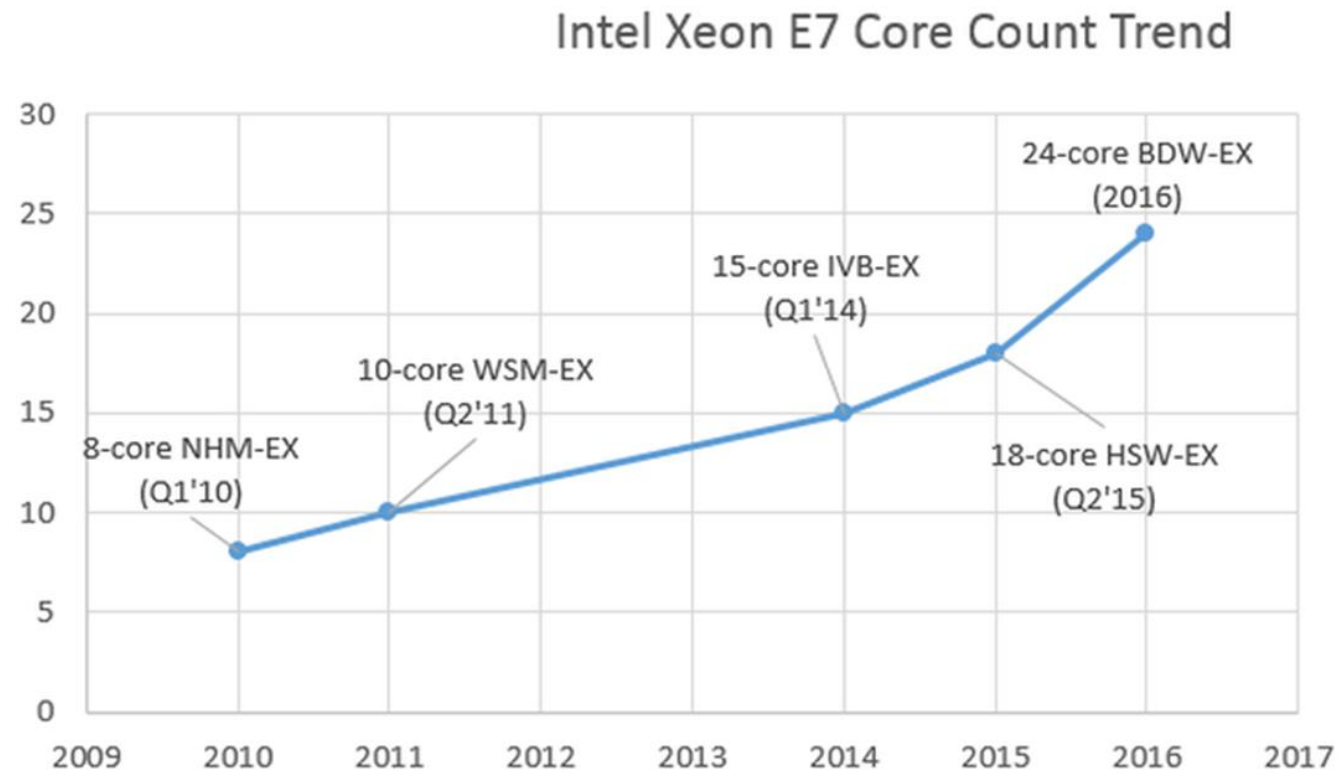
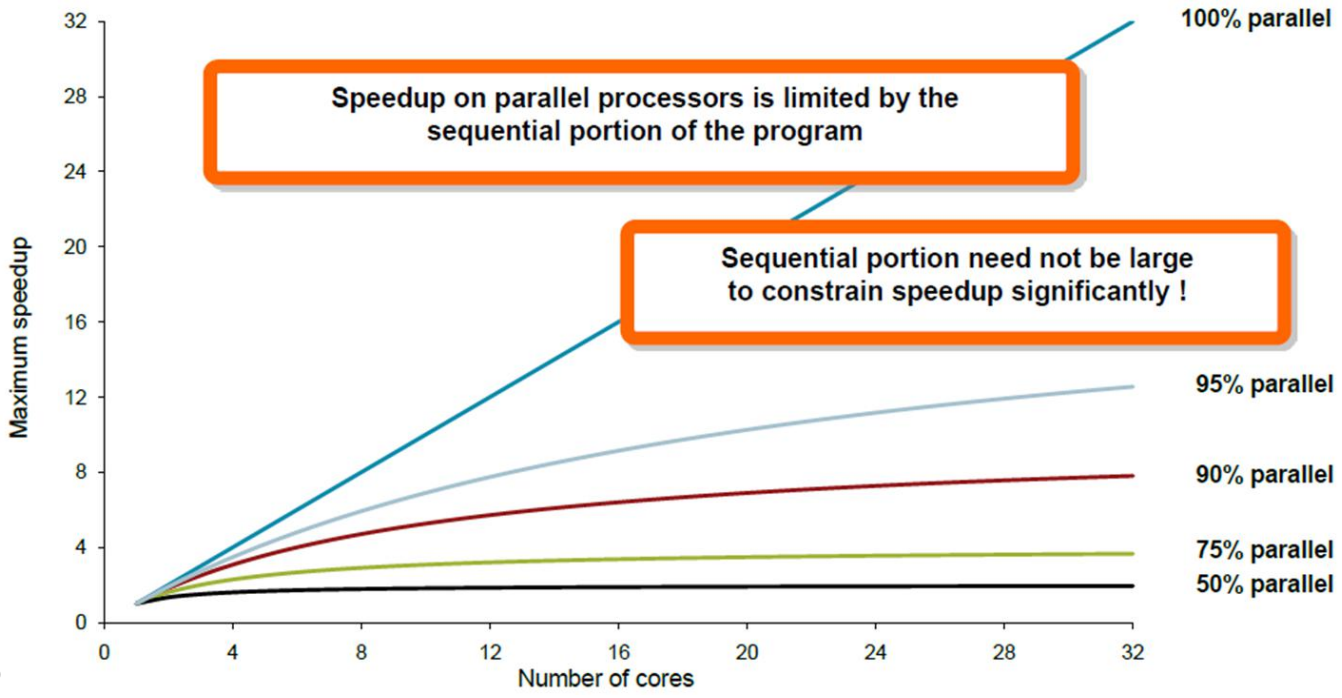


Image credit: Ashraf Eassa

Amdahl's Law is Alive and Well

- Speedup on a parallel system with *N* cores is limited by the serial fraction *s* that cannot be parallelized

$$\lim_{N \to \infty} speedup\ (N) = \frac{1}{s}$$



Slowdown of Moore's Law

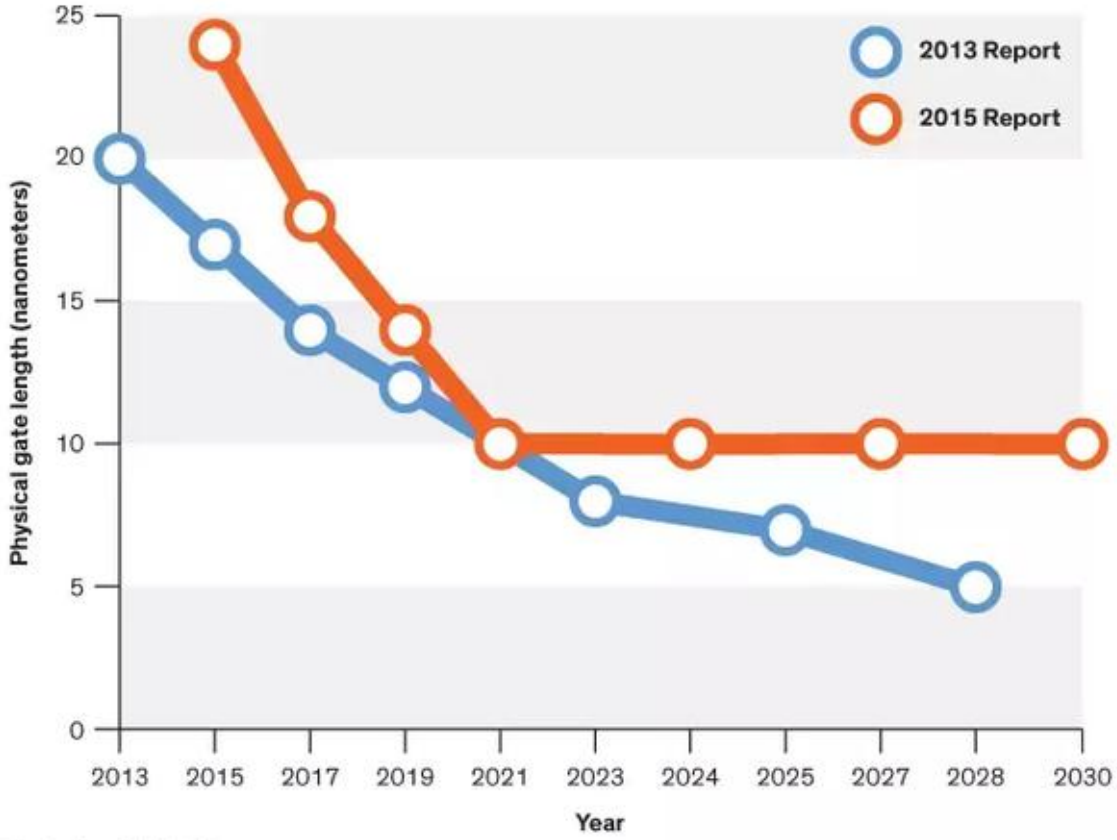
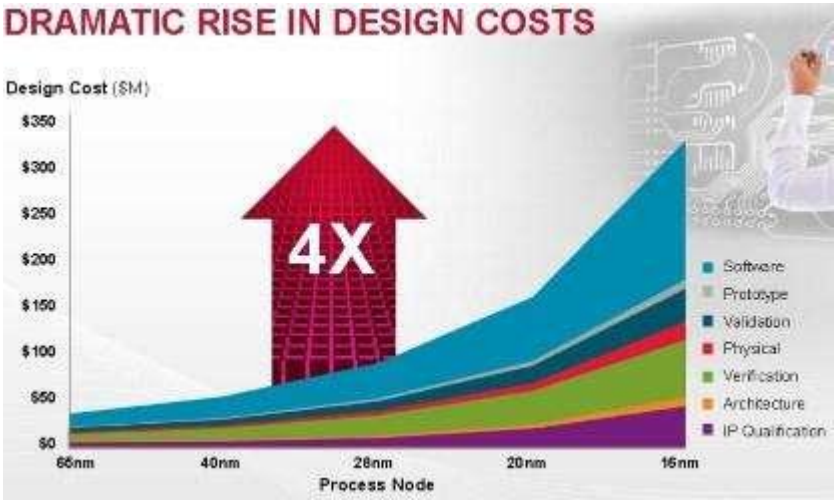


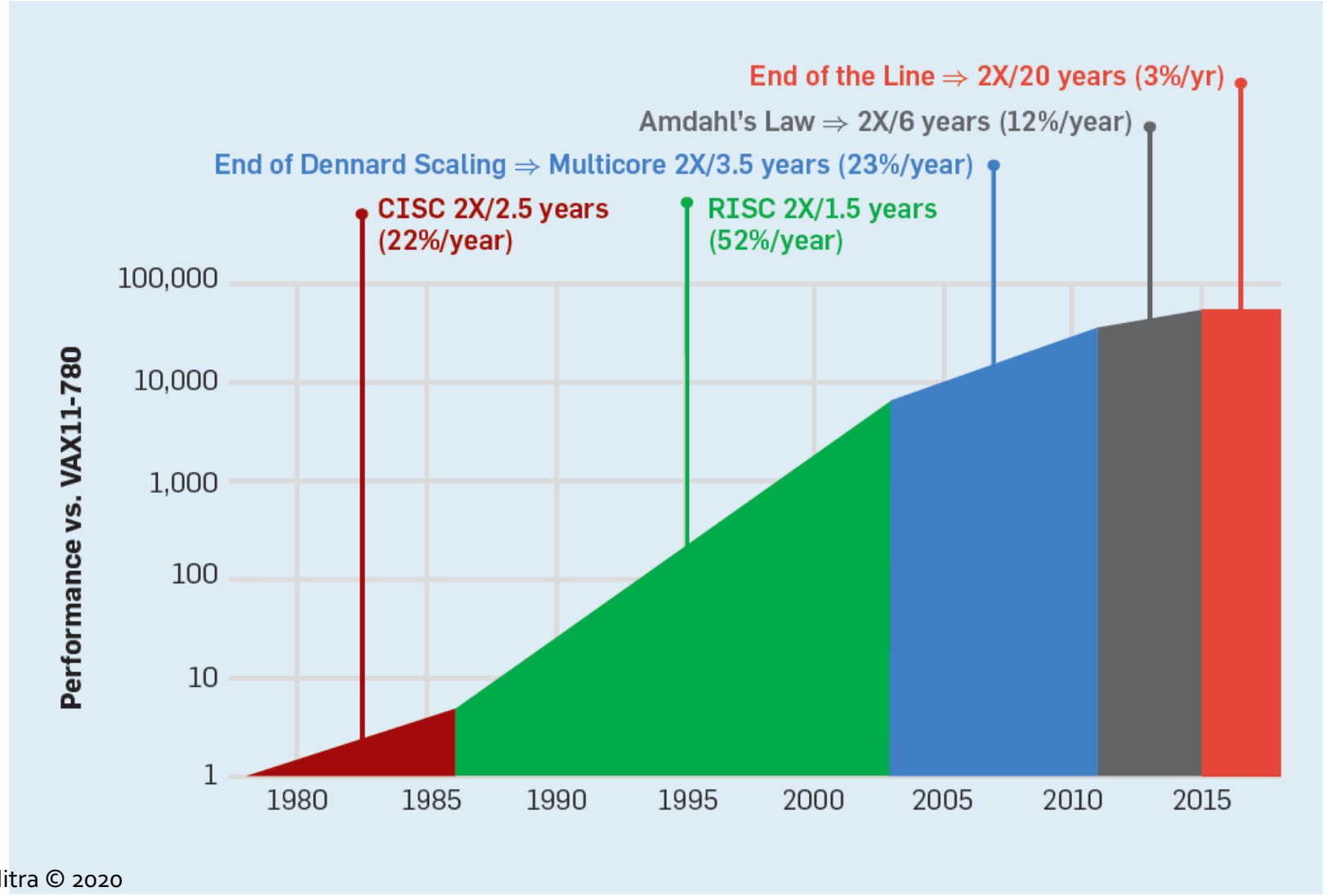
Illustration: Erik Vrielink

The trajectory of transistor feature sizes (the physical gate length of transistors in high-performance logic is shown here) could take a sharp turn in 2021.



Source: Scott McGregor, Broadcom

Processor performance over the years



Jouppi, Norman P., et al.
"A domain-specific architecture for deep neural networks."
Communications of the ACM 2018

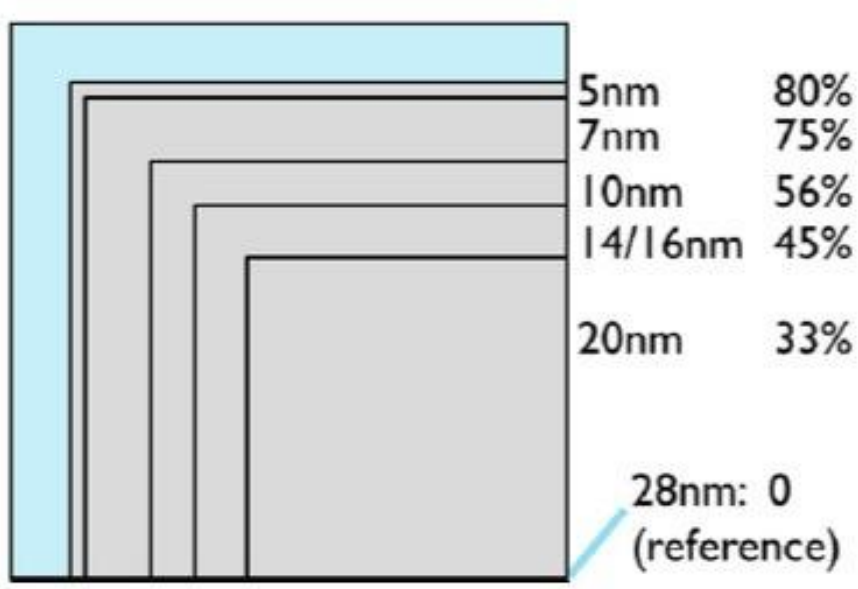
Why Heterogeneity? Rise of Dark Silicon

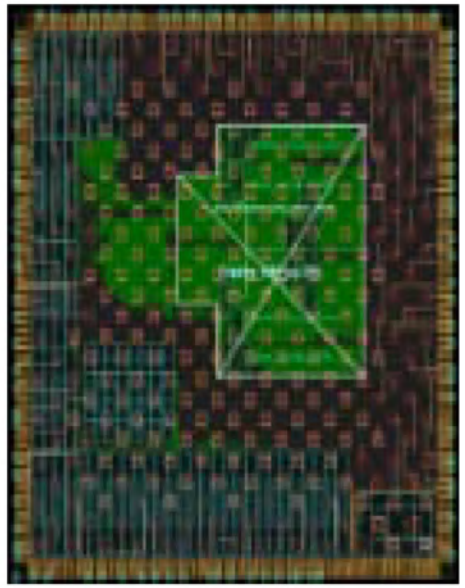
Node	45nm	22nm	11nm
Year	2008	2014	2020
Area^{-1}	1	4	16
Peak freq	1	1.6	2.4
Power	1	1	0.6
		$(4 \times 1)^{-1} = 25\%$	$(16 \times 0.6)^{-1} = 10\%$
Exploitable Si (in 45nm power budget)		25%	10%

Source: Jem Davies, ARM

Dark Silicon

- We can have more transistors and cores due to Moore's Law
- We cannot power them all due to Thermal Design Power (TDP) constraint
- At 5nm, 80% of the chip will be dark

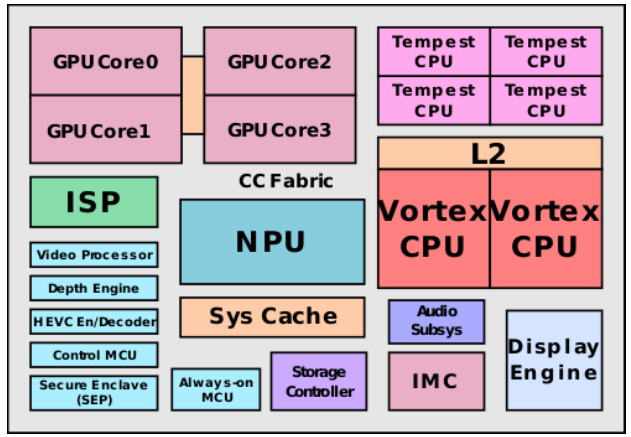




Source: Paul McLellan, Cadence

Heterogeneous cores towards rescue

- Need heterogeneous cores
- Power on only the most appropriate cores
- Energy-efficient, high-performance computation



Apple A12 Bionic

Image credit: Wikichip

Dark Silicon in Practice: Samsung Exynos 7420 SoC

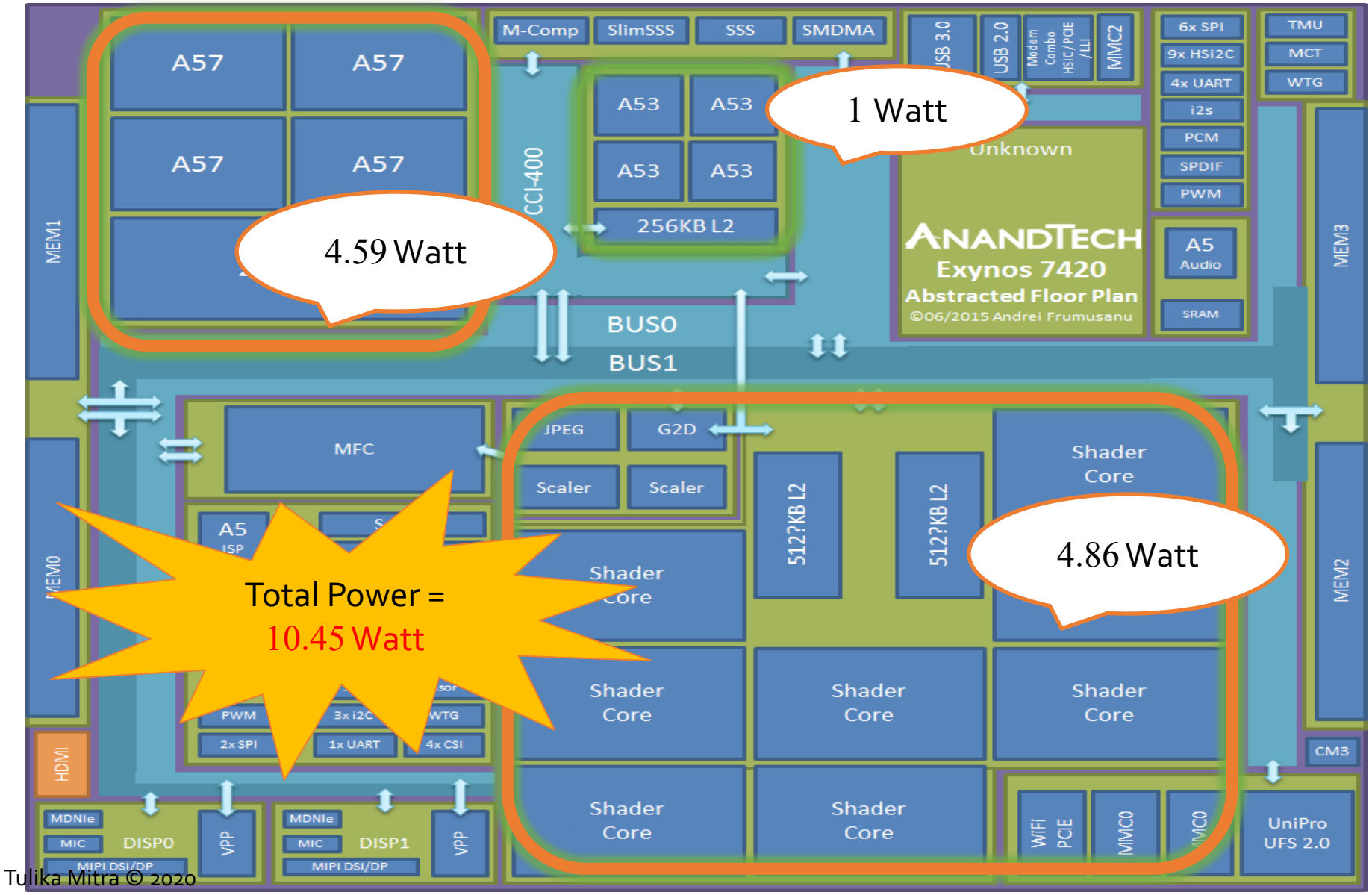
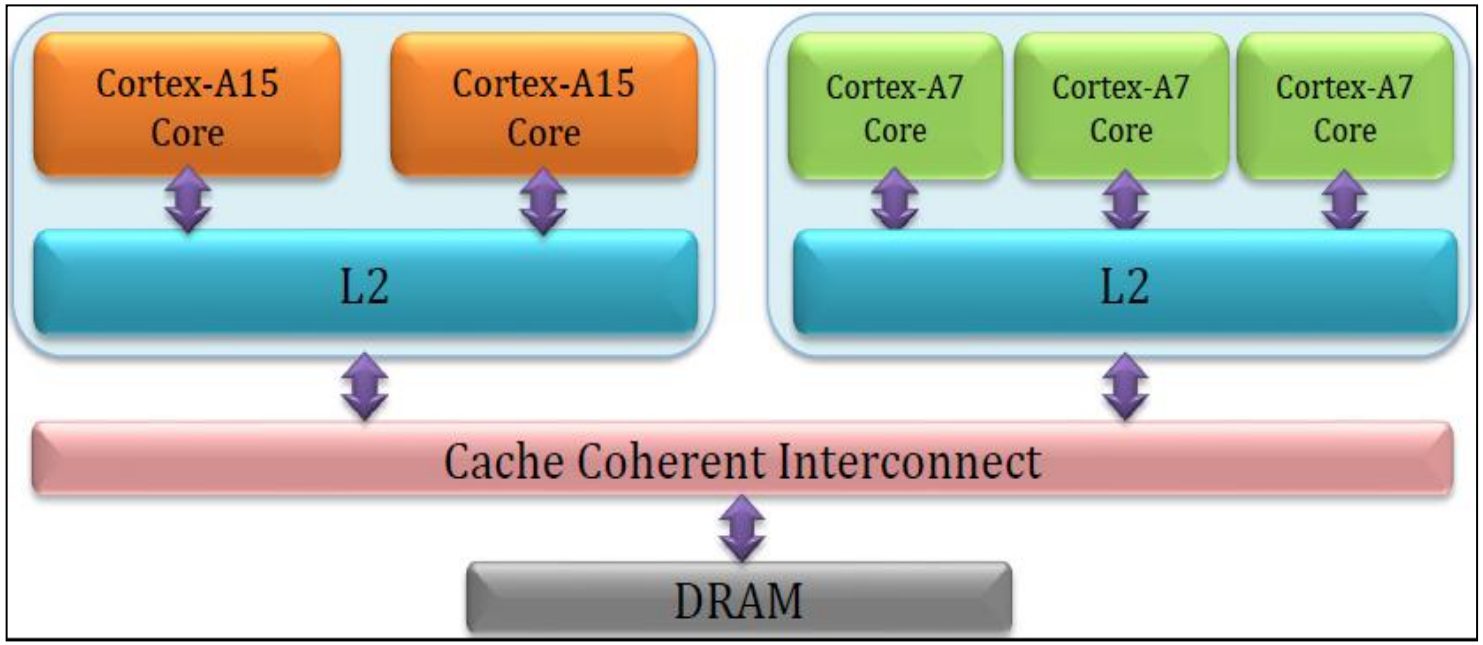


Image credit: AnandTech

Performance Heterogeneity

- Cores with the same instruction-set architecture (functionality) but different micro-architecture and hence power-performance characteristics
- Why? ILP versus TLP
  - Homogeneous collection of simple cores is excellent match for applications with high thread-level parallelism (TLP)
  - Applications with large sequential fractions suffer from lack of instruction-level parallelism support (ILP): Amdahl's Law
- Why? Power versus Performance
  - Some workloads need low-performance at low-power on simple cores (email client) while some need high-performance at the cost of high-power on complex cores (flash websites)

ARM big.LITTLE Architecture

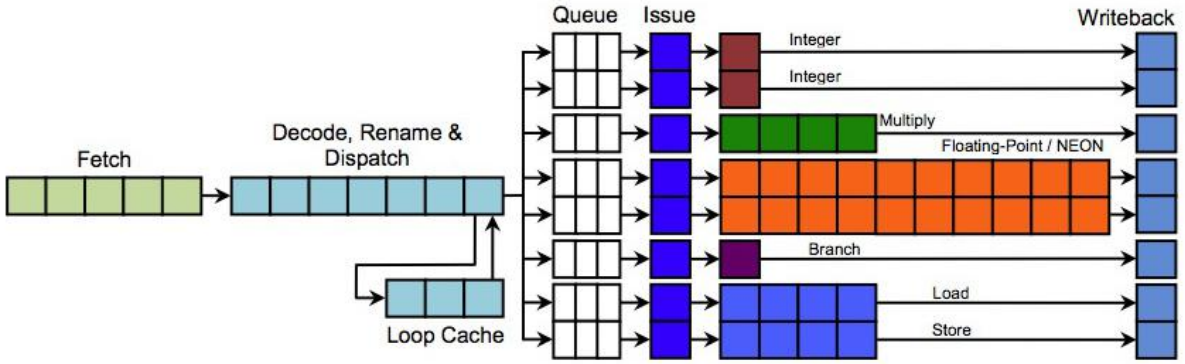


CPU	Pipeline	Issue Width	L2$ size	Frequency Range
small	In-order	2	512 KB	350 – 1000 MHz
big	Out-of-order	3	1 MB	500 – 1200 MHz

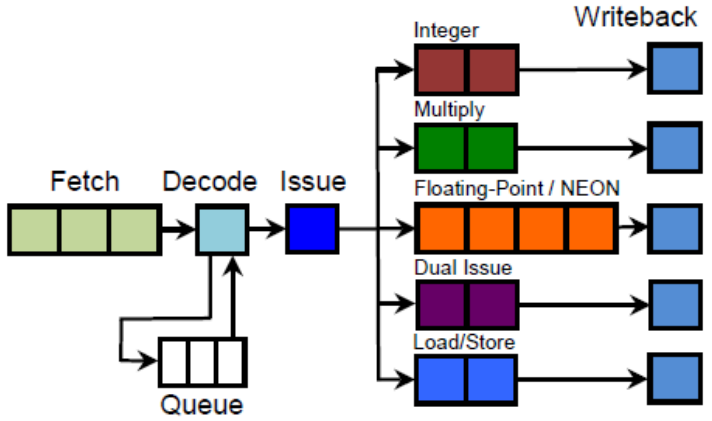
* Test chip at 45nm technology

Micro-architectural Diversity

Cortex A15 out-of-order pipeline

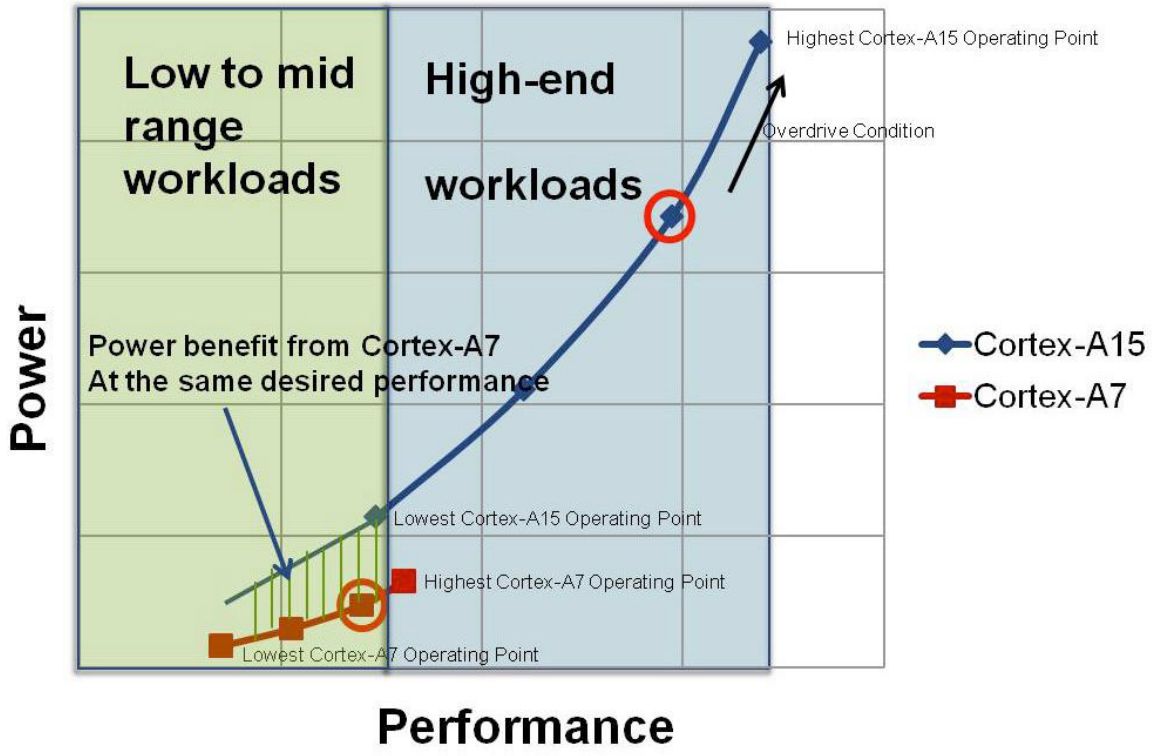


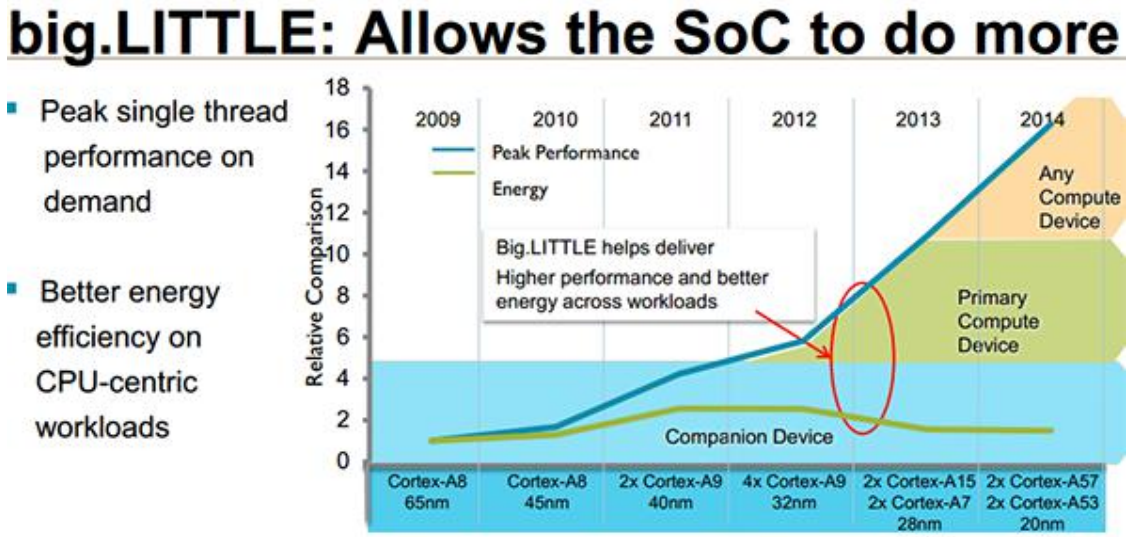
Cortex A7 in-order pipeline



Source: ARM

Advantage of Performance Heterogeneity

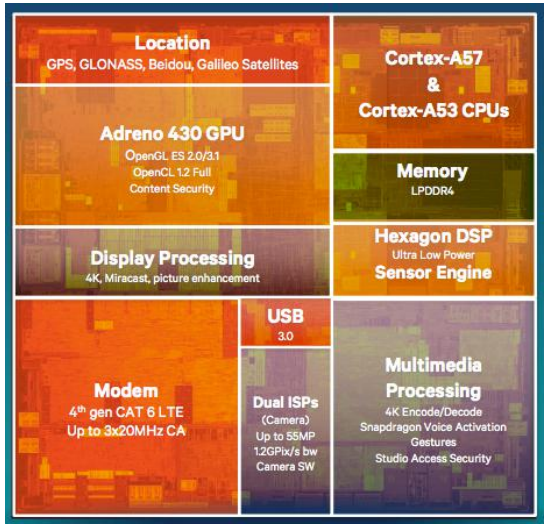




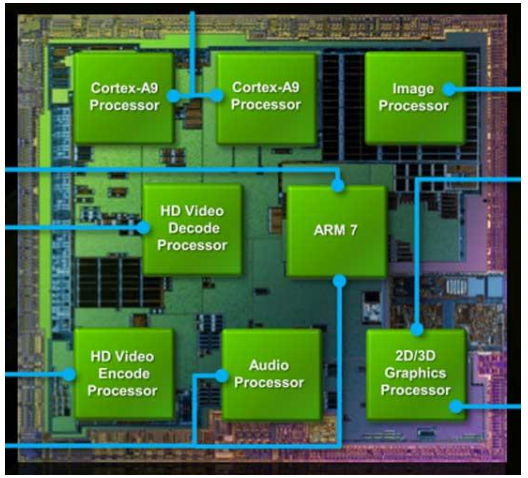
Source: ARM

Functional Heterogeneity

- Domain-Specific Processors and accelerators
- Different programming models (DSP, GPU, FPGA, CGRA) or non-programmable accelerators (video encoder, machine learning accelerator)



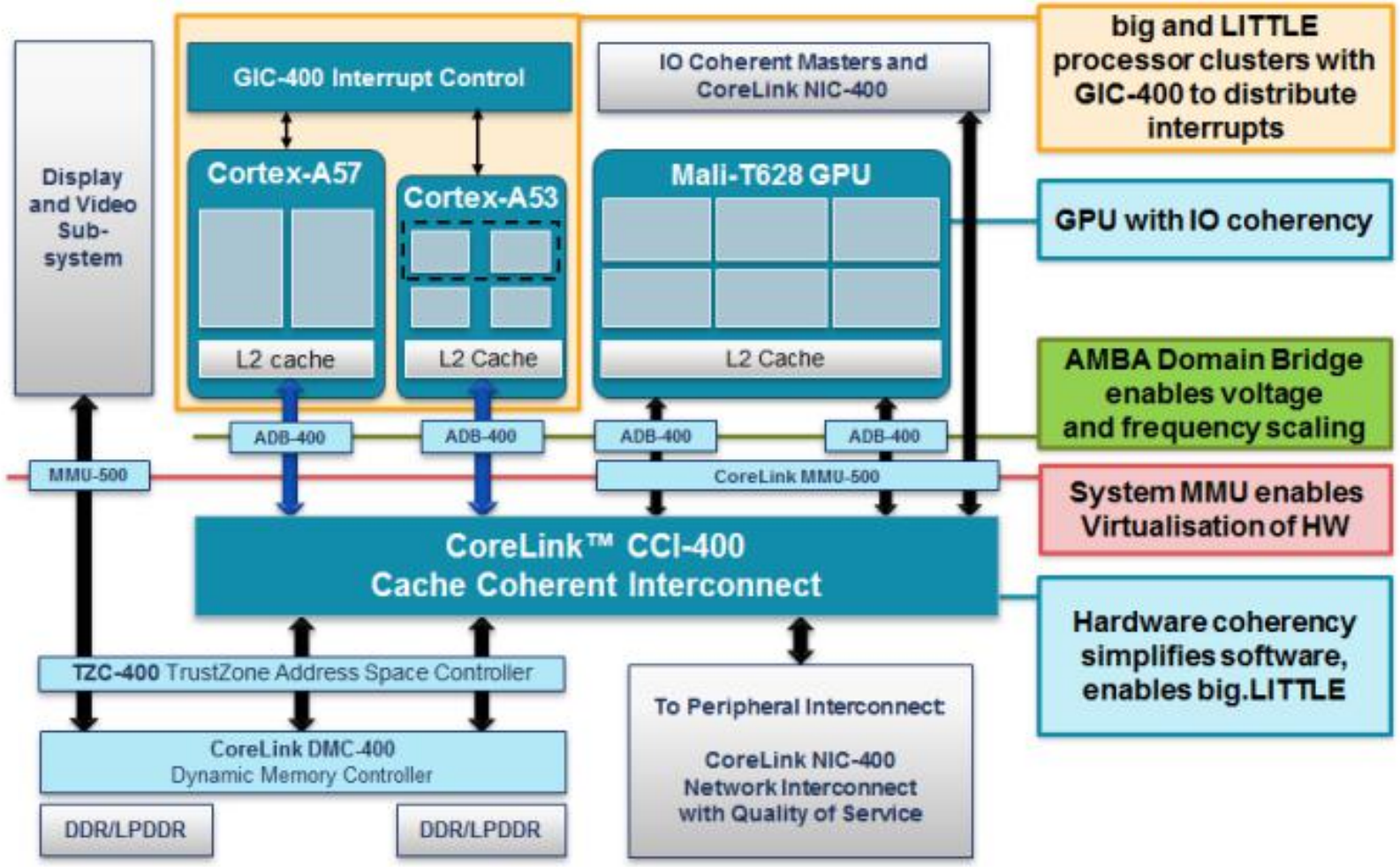
Qualcomm Snapdragon



NVIDIA Tegra

Image credit: Qualcomm, NVIDIA

Functional and Performance Heterogeneity: Samsung Exynos 7420 SoC



Source: Samsung

SOFTWARE CHALLENGES

Best of Times… Worst of Times..

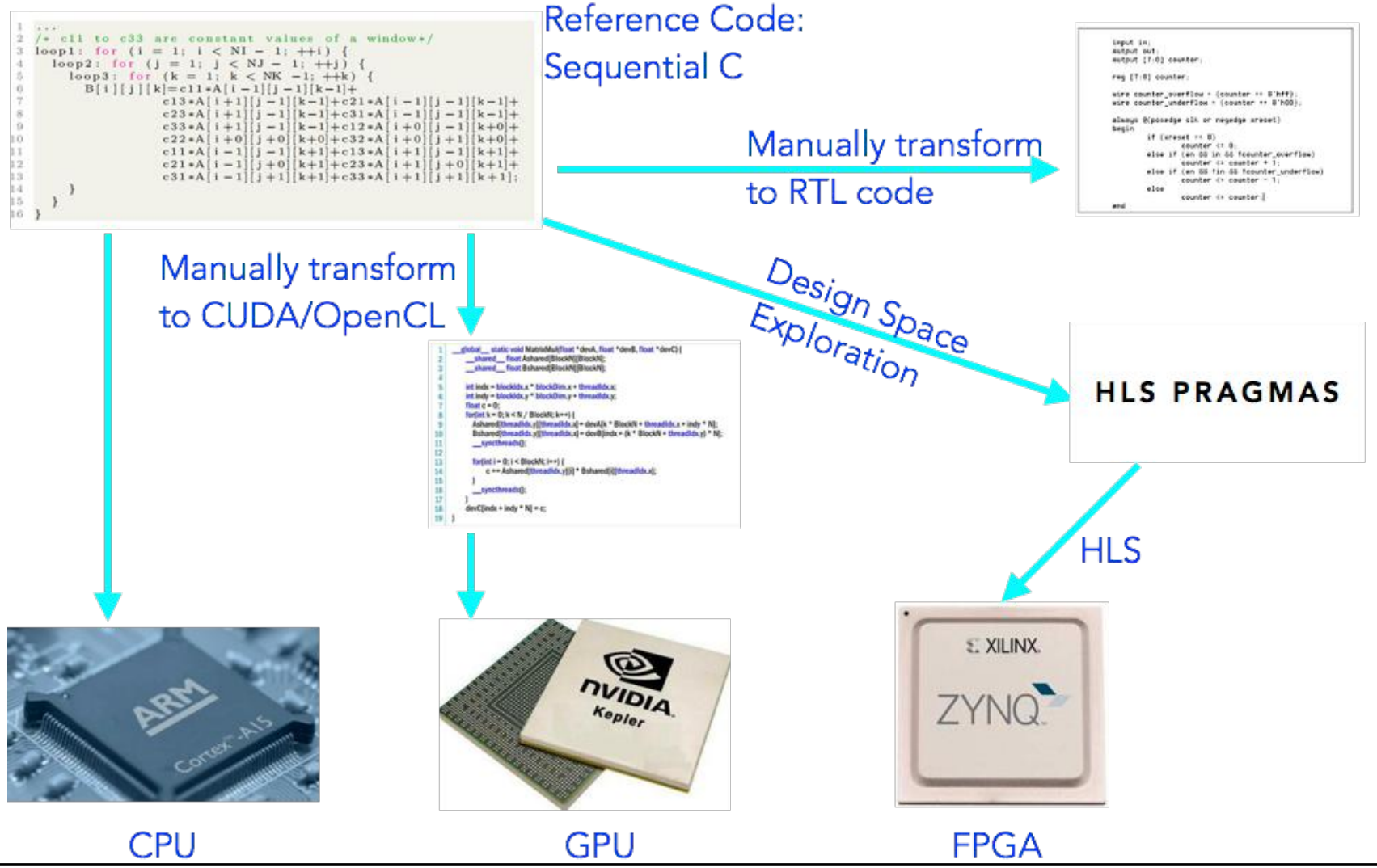
- Tremendous progress in mobile SoC design with heterogeneous computing over the past five years

- Software development remains as challenging (if not more)

Software Developers' Dilemma

- Which core or accelerator to use for my app?
- How to orchestrate application execution across multiple cores or accelerators?
- How to remain within TDP (thermal design power) in the dark silicon era?
- How to maximize battery life?

Compute Core Selection



Performance Prediction on Accelerators

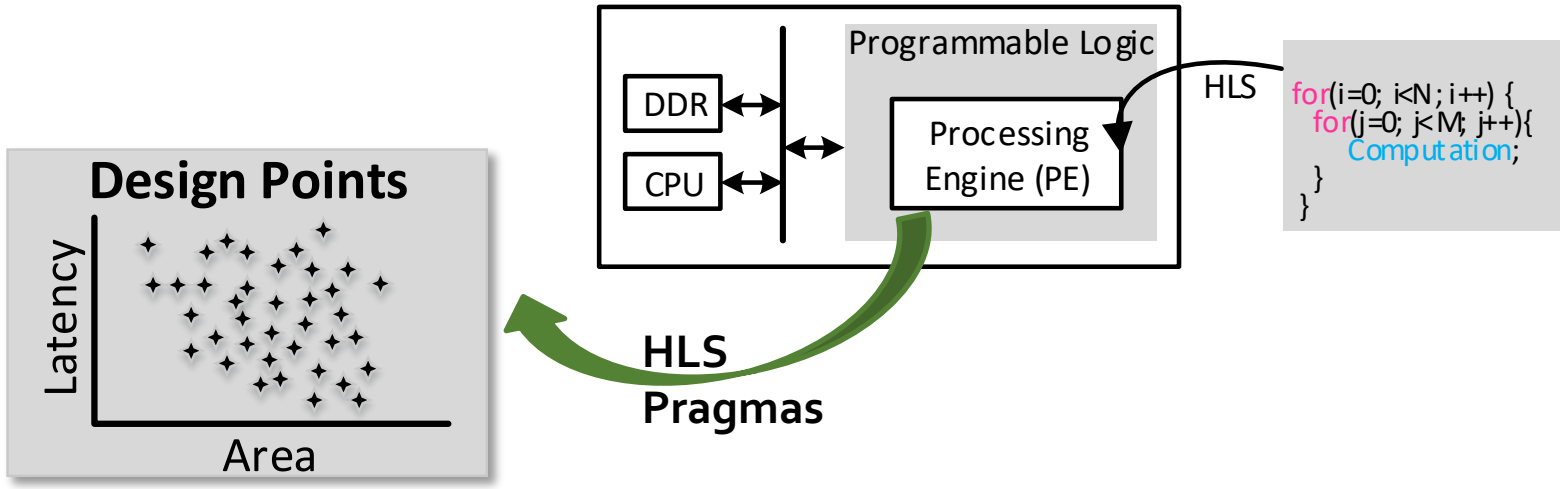
- Lin-Analyzer: Performance prediction for FPGAs from sequential C code
- CGPredict: Performance prediction for GPUs from sequential C code
- Provide performance insights to the software developers

FPGA High-Level Synthesis (HLS)

- **FPGA**
  - Re-programmability, high parallelism
  - Implementation difficulty, long synthesis flow
- **High-Level Synthesis (HLS)**

  - Multiple design choices (Pragmas)
    - Loop unrolling, loop pipelining, array partitioning, etc.

Design Space Exploration



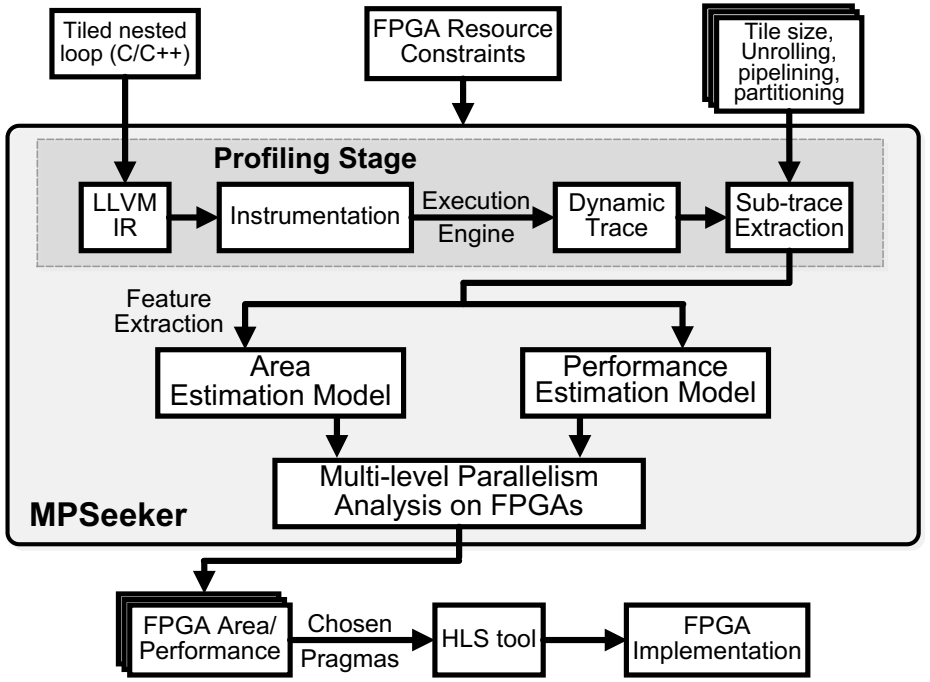
- **Finding a good-quality design with HLS is difficult**
  - Huge design space
  - Non-negligible HLS runtime

Lin-Analyzer

- Input: C/C++ Code
- Output:
  - Estimated performance at a chosen set of pragma values for loop unrolling, loop pipelining, array partitioning
  - Best set of pragma values or configuration
- Does not invoke High-Level Synthesis (HLS) in the estimation
  - Hybrid static-dynamic analysis
- HLS tool, like Vivado, is invoked with the chosen configuration at the end to generate RTL code

Lin-Analyzer and MPSeeker

- State-of-the-art FPGA performance/area estimation and design space exploration from C code



Zhong, Guanwen, et al. "Lin-analyzer: a high-level performance analysis tool for FPGA-based accelerators." *DAC* 2016
Zhong, Guanwen, et al. "Design Space exploration of FPGA-based accelerators with multi-level parallelism." *DATE* 2017
https://github.com/zhguanw/lin-analyzer

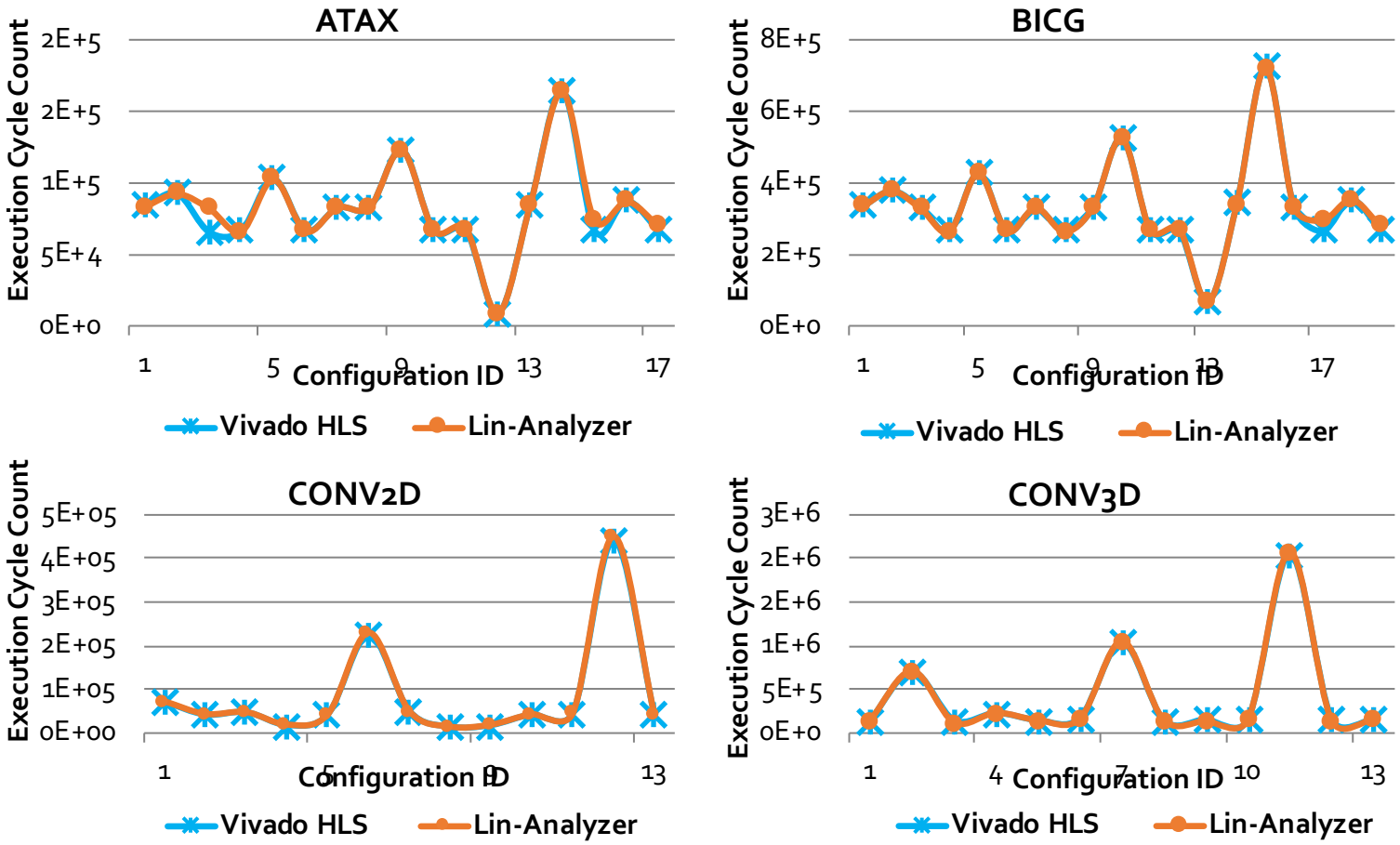
Power of Lin-Analyzer

```
...
/* c11 to c33 are constant values of a window*/
loop1: for (i = 1; i < NI - 1; ++i) {
  loop2: for (j = 1; j < NJ - 1; ++j) {
    loop3: for (k = 1; k < NK -1; ++k) {
      B[i][j][k]=c11*A[i-1][j-1][k-1]+
                 c13*A[i+1][j-1][k-1]+c21*A[i-1][j-1][k-1]+
                 c23*A[i+1][j-1][k-1]+c31*A[i-1][j-1][k-1]+
                 c33*A[i+1][j-1][k-1]+c12*A[i+0][j-1][k+0]+
                 c22*A[i+0][j+0][k+0]+c32*A[i+0][j+1][k+0]+
                 c11*A[i-1][j-1][k+1]+c13*A[i+1][j-1][k+1]+
                 c21*A[i-1][j+0][k+1]+c23*A[i+1][j+0][k+1]+
                 c31*A[i-1][j+1][k+1]+c33*A[i+1][j+1][k+1];
    }
  }
}
```

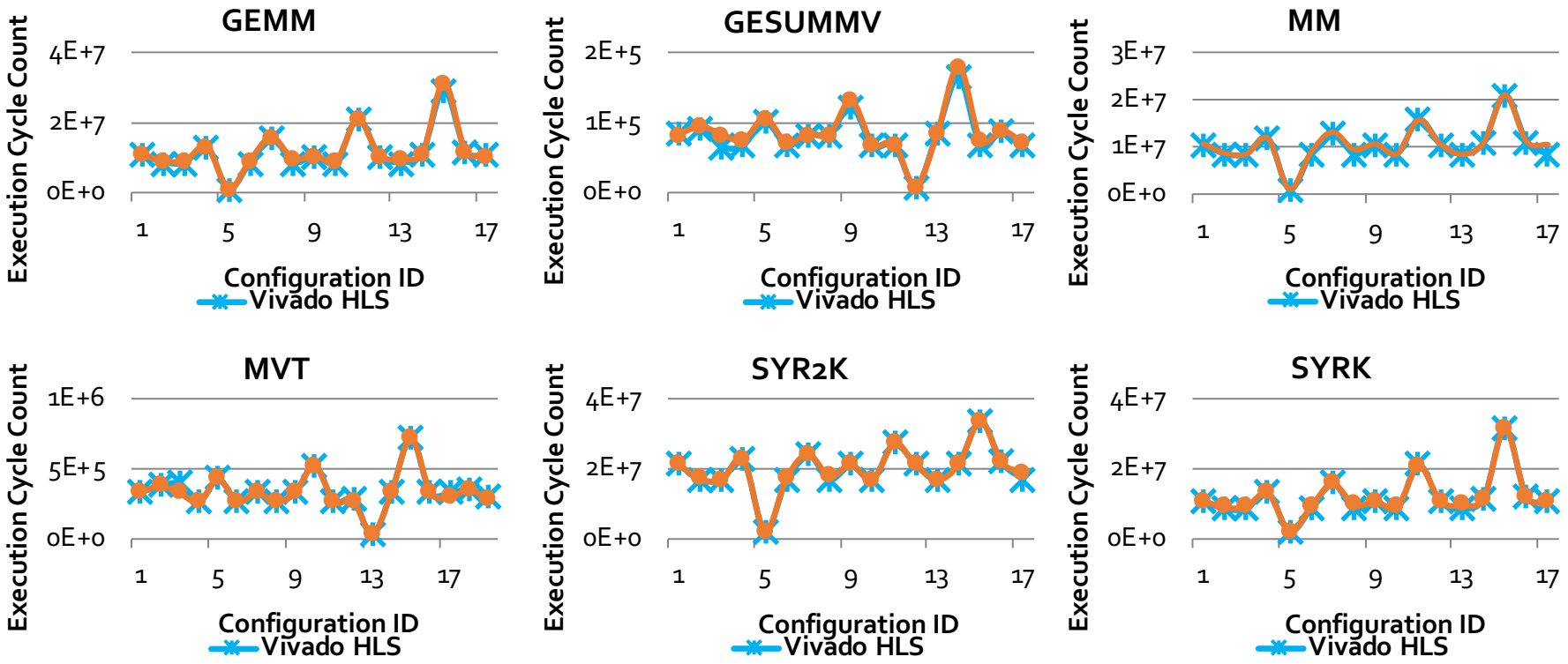
Input Size	Loop Pipelining	Loop Unrolling	Array Partitioning	Vivado HLS Runtime
32*32*32	Disabled	loop3 factor=30	A, cyclic, 2 B, cyclic, 2	44.25 seconds
	loop3, yes	loop3 factor=15	A, cyclic, 16 B, cyclic, 16	1.78 hours
	loop3, yes	loop3 factor=16	A, cyclic, 16 B, cyclic, 16	3.25 hours

Input Size	Design Space	DSE Time	
		Exhaustive HLS-based	Lin-Analyzer
32*32*32	120	10 days*	29.30 seconds

Estimation Accuracy



Estimation Accuracy



Benchmark	ATAX	BICG	CONV2D	CONV3D	GEMM
Difference (%)	2.68	1.24	1.63	3.75	3.25

Benchmark	GESUMMV	MM	MVT	SYR2K	SYRK
Difference (%)	5.15	2.69	2.05	1.32	2.78

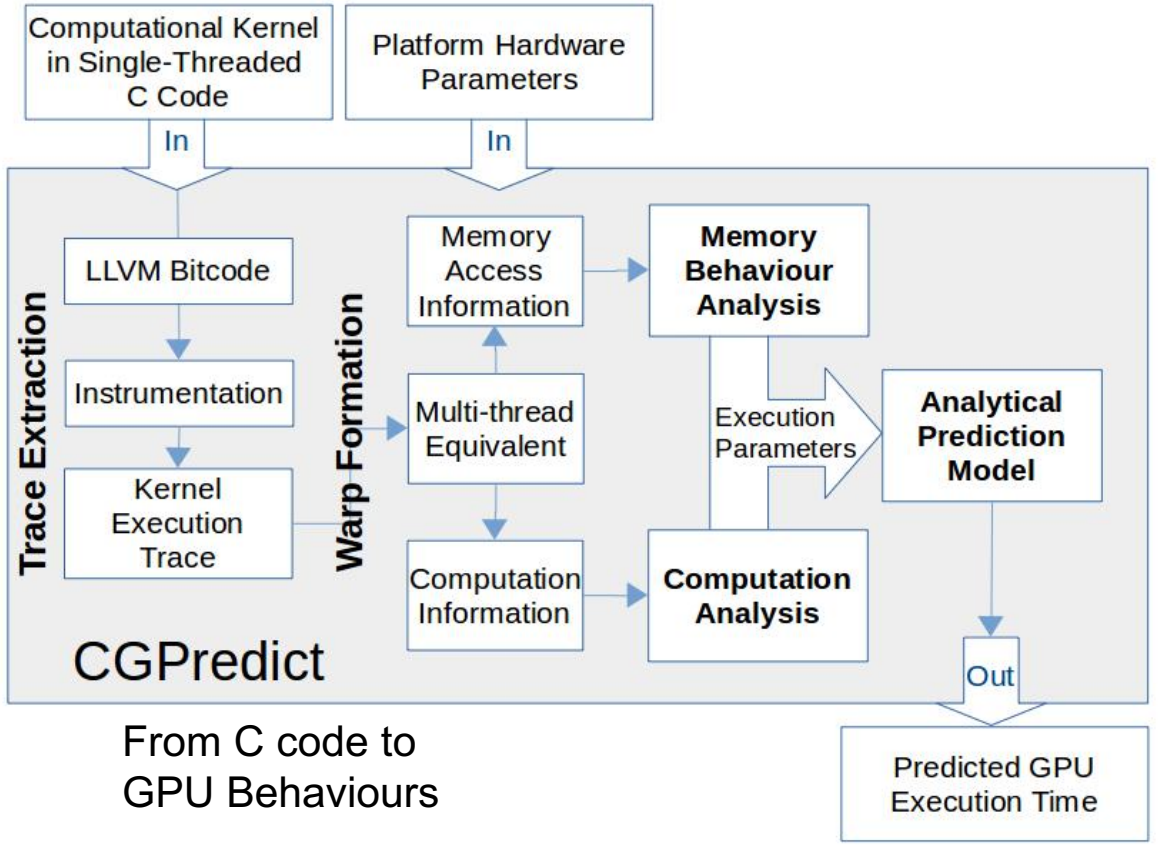
Design Space Exploration (DSE) Time

- Training time (one-off): 660 seconds

Benchmark	Design Space Exploration Time	
	HLS (hours)	Lin-Analyzer (min.)
DCT1D	31.7	3.0
DERICHE1	26.7	0.4
GEMVER1	23.2	3.3
MM	35.9	0.5
MVT	22.9	0.6
Average	28.1	1.6

CGPredict: C to GPU Performance Estimation

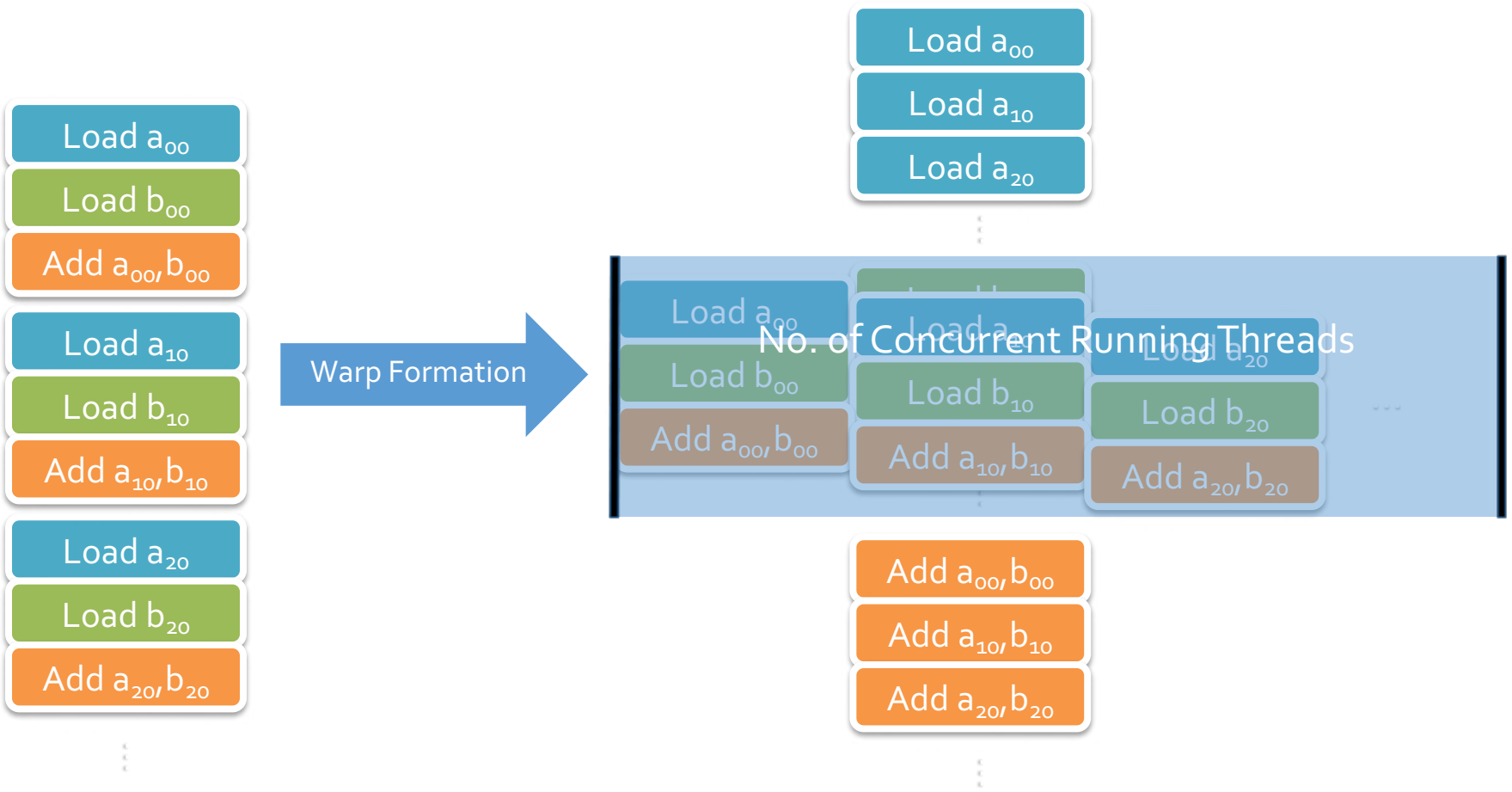
- An analytical framework that predicts the performance of a computational kernel on an embedded GPU architecture from un-optimized, single-threaded C code



From C code to GPU Behaviours

Wang, Siqi, Guanwen Zhong, and Tulika Mitra. "CGPredict: Embedded GPU Performance Estimation from Single-Threaded Applications." *ACM TECS* 2017

Warp Formation



WARP Formation

Trace format: matrix_name, type, address, loop-index-i, loop-index-j

```
A,load,139904633909248,0,0
B,load,139904633839616,0,0
A,load,139904633909252,0,1
B,load,139904633840128,0,1
A,load,139904633909256,0,2
B,load,139904633840640,0,2
A,load,139904633909260,0,3
B,load,139904633841152,0,3
.
.
A,load,139904633909372,0,31
B,load,139904633855488,0,31
A,load,139904633909376,0,32
B,load,139904633856000,0,32
A,load,139904633909380,0,33
B,load,139904633856512,0,33
A,load,139904633909384,0,34
B,load,139904633857024,0,34
.
.
```

```
A,load,139904633909248,0,0   }
A,load,139904633909252,0,1   }
A,load,139904633909256,0,2   }  warp 0
A,load,139904633909260,0,3   }
.                            }
A,load,139904633909372,0,31  }
A,load,139904....,1,0        }
A,load,139904....,1,1        }  warp 1
.                            }
A,load,139904....,1,31       }
.
.
A,load,139904....,31,0       }
A,load,139904....,31,1       }  warp 31
.                            }
A,load,139904....,31,31      }
```

Block (0, 0)

```
A,load,139904....,32,0       }  warp 0
A,load,139904....,32,1       }
.                            }
A,load,139904....,32,31      }
A,load,139904....,33,0       }
A,load,139904....,33,1       }  warp 1
.                            }
A,load,139904....,33,31      }
.
.
.
```

Block (1, 0)

Performance Estimation

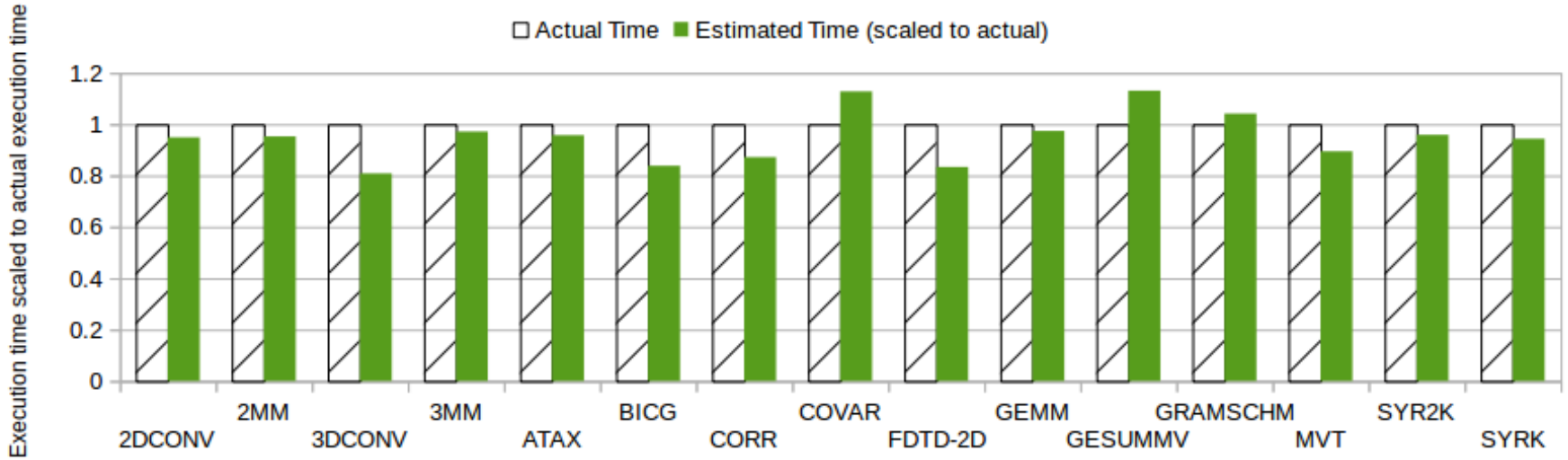
- Computation time estimation
- Memory access time estimation by memory hierarchy modeling
- Memory latency hiding

Experimental Setup

- NVIDIA Jetson TK1 platform
  - 192 CUDA cores
  - 852 MHz
- Polybench benchmark suite
  - C implementation
  - Unoptimized CUDA implementation

Accurate Performance Estimates

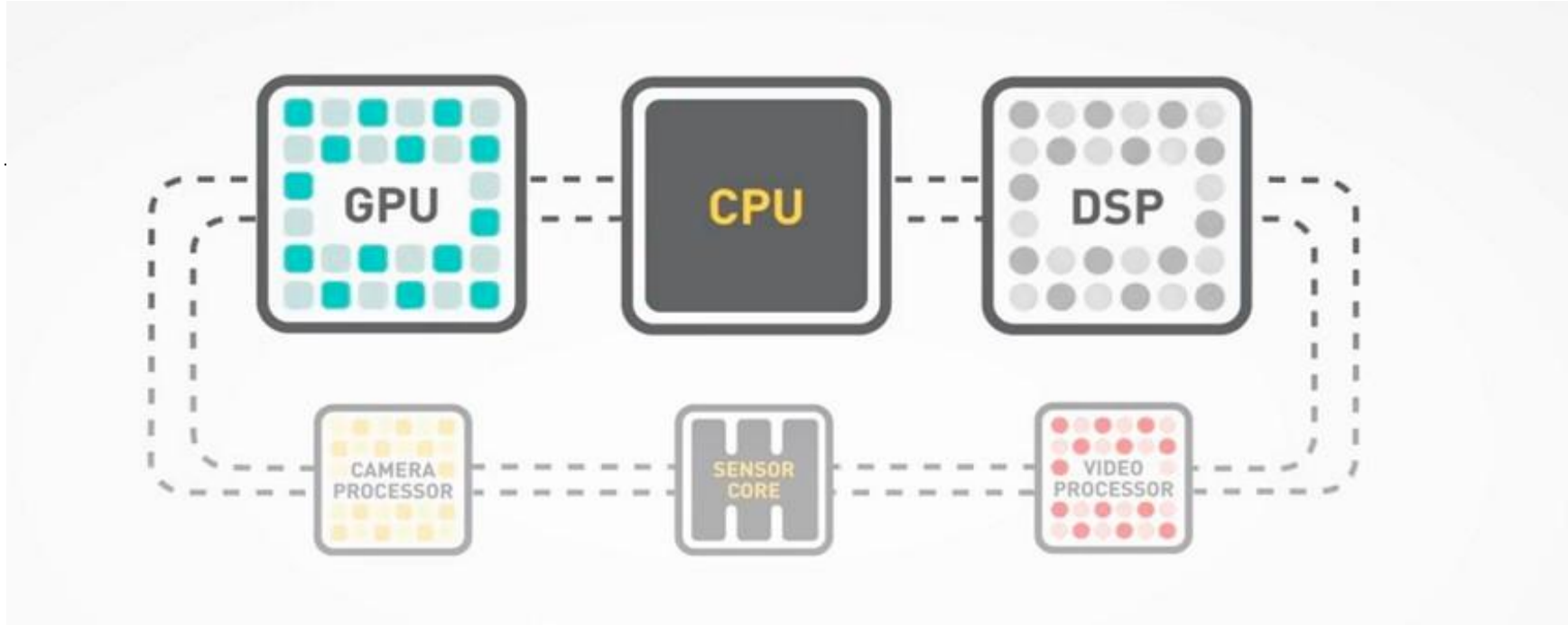
- Estimation Error: As low as 2.66%, avg 9%



Choice between GPU and FPGA via Analysis of C programs

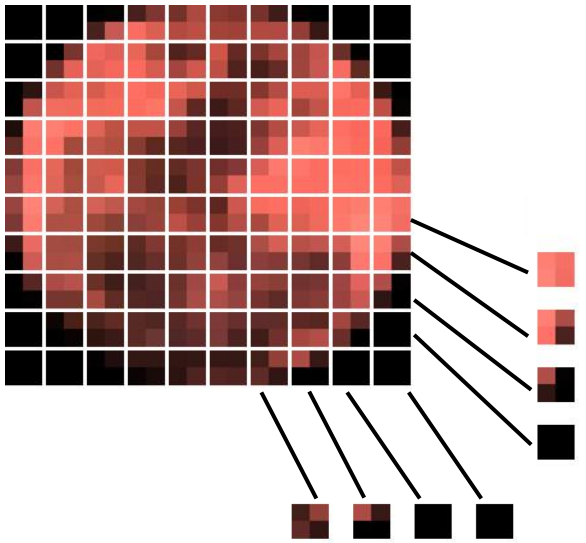
Benchmark Name	Input Size	Estimated Time (ms)		Actual Time (ms)		Choice of Platform
		GPU	FPGA	GPU	FPGA	
MM	1024	**242.51**	1180	**250.27**	1450	GPU
MVT	2048	48.31	**9.09**	42.371	**10.41**	FPGA
GEMVER1	2048	**2.61**	16.55	**4.57**	19.81	GPU
DERICHE1	1024	**0.95**	2.99	**1.53**	3.37	GPU
DCT1D	1024	2697.75	**636.47**	2685.362	**650.8**	FPGA

ORCHESTRATING HETEROGENEOUS CORES

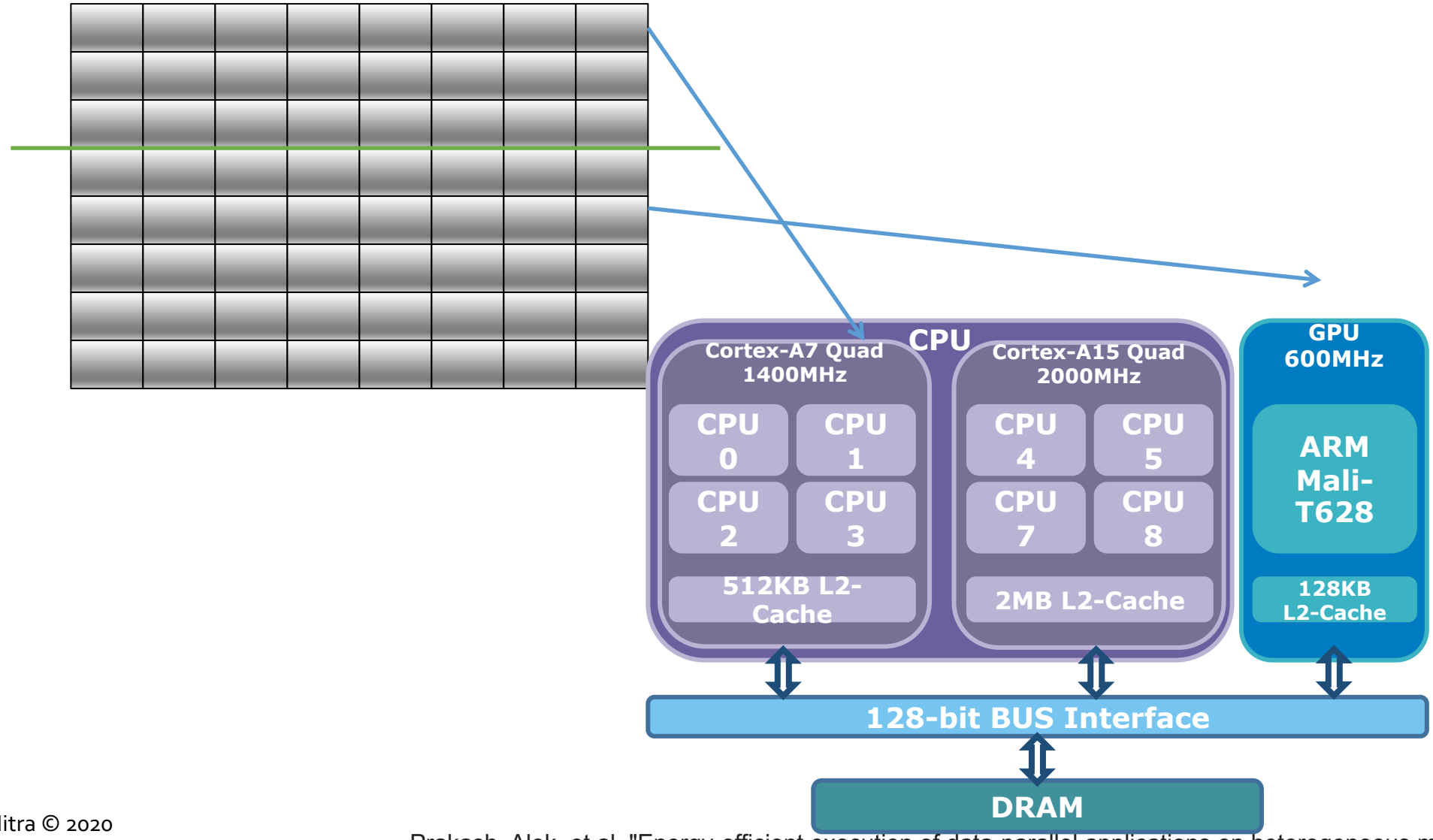


OpenCL to Support Heterogeneity

- Free, open standard for cross-platform, parallel programming on CPU, GPU, FPGAs
- Fine grained parallelism to allow thousands of active threads
- Smartphone camera performs all image processing on Mali GPU

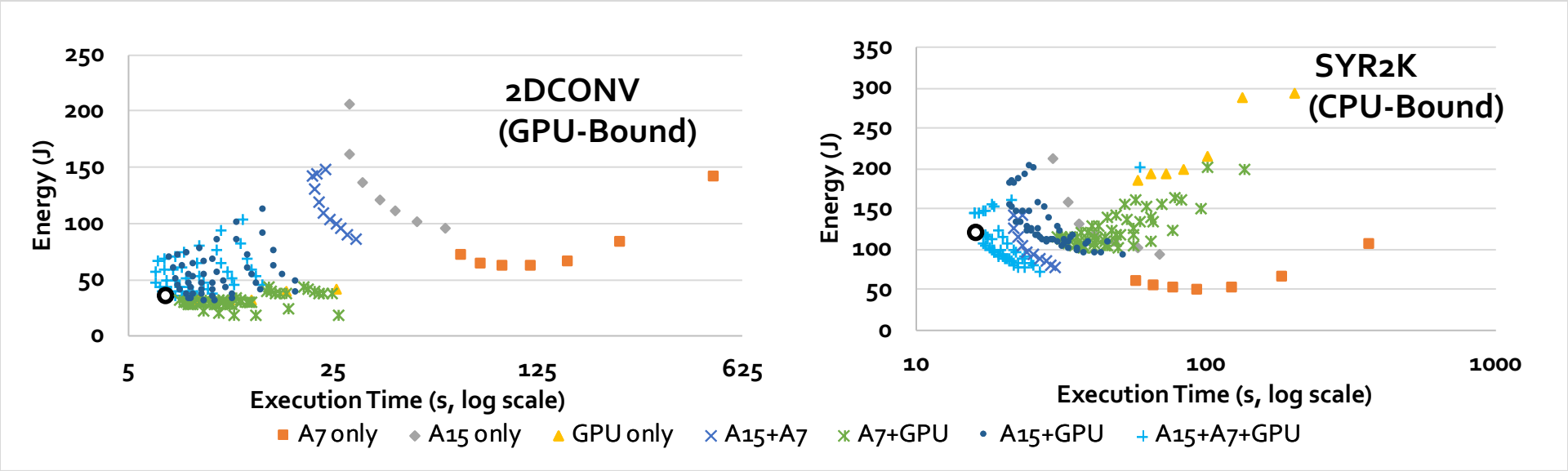


Workload Partition: CPU + GPU



Prakash, Alok, et al. "Energy-efficient execution of data-parallel applications on heterogeneous mobile platforms." *ICCD 2015*

Design Space: Energy-Delay

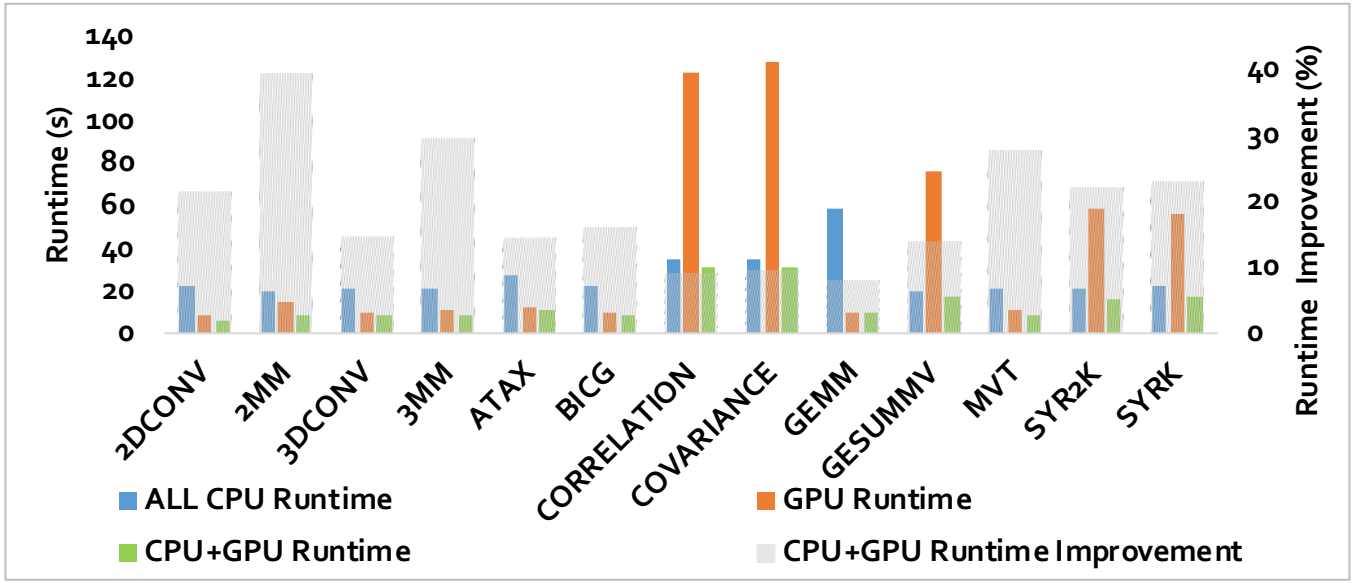


- 2DCONV: Works well with Functional Heterogeneity (A7+GPU)
- SYR2K: Works well with Performance Heterogeneity (A7+A15)
- Main Factors: DVFS, Workload Partition & Memory Contention

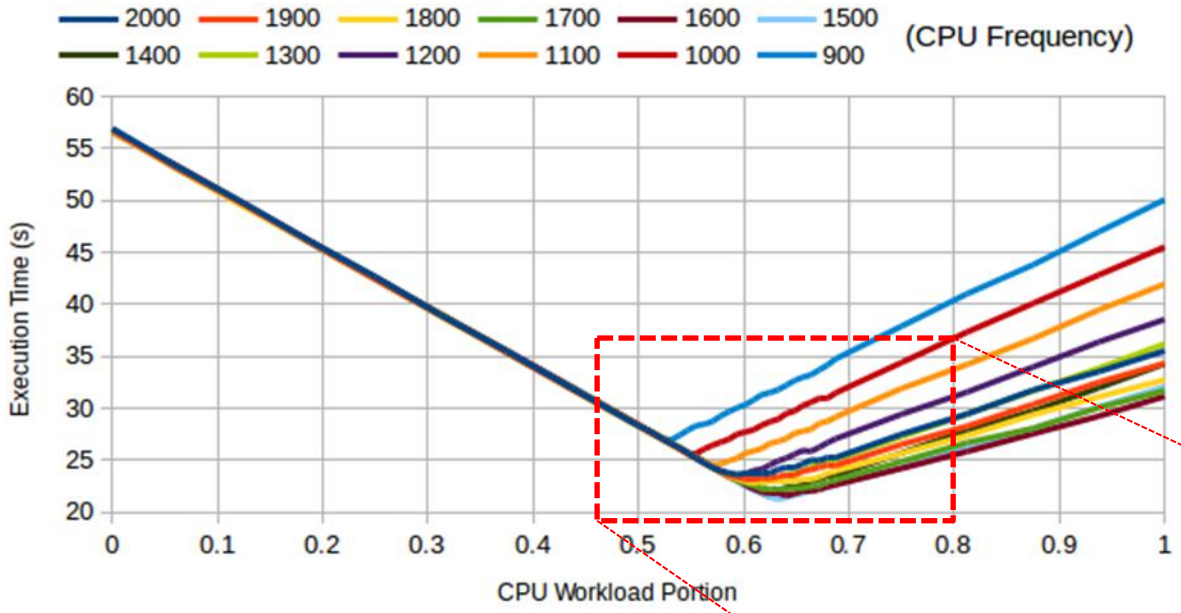
Finding optimal configuration: Models

Gain with Heterogeneity

- Optimal partitioning for CPU and GPU + DVFS
- Average 19% runtime improvement

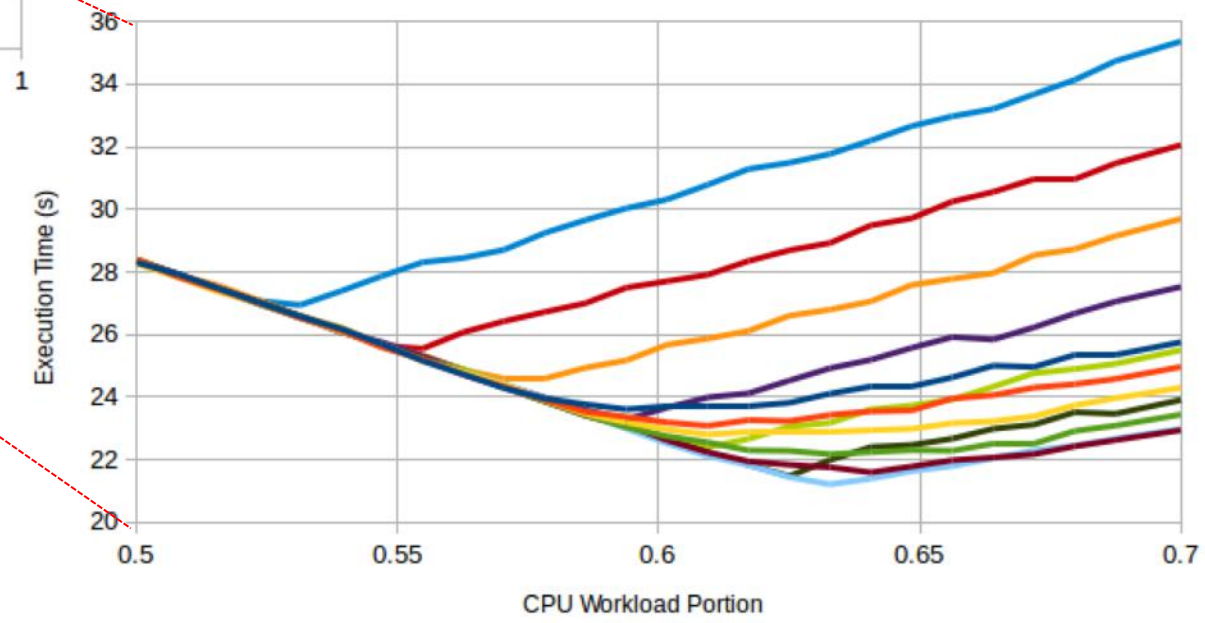


Thermal Constraint



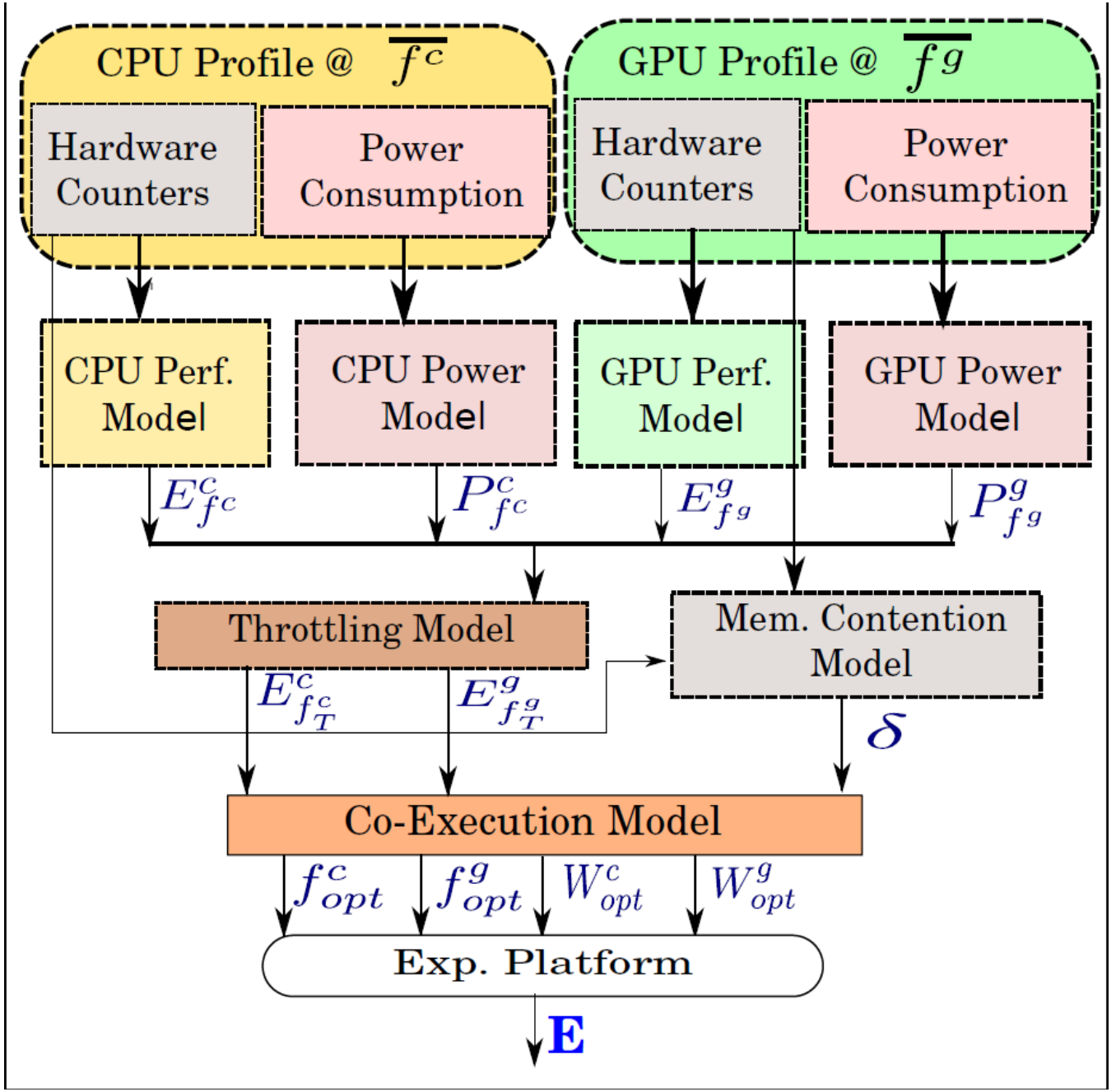
Thermal Throttling impacts co-execution performance Significantly

Lower frequency may induce less thermal throttling and lead to lower runtime for co-execution



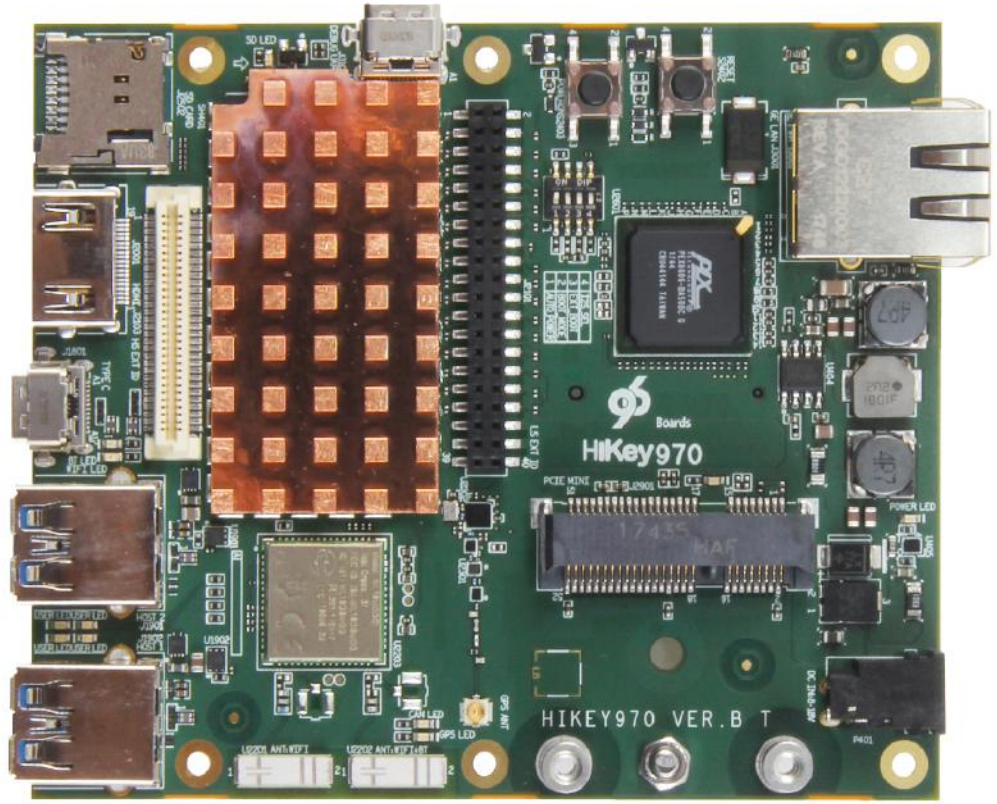
OPTiC

- A framework to select CPU-GPU workload partition and corresponding frequency to minimize thermal throttling and maximize performance



Wang, Siqi et al. OPTiC: Optimizing Collaborative CPU-GPU Computing on Mobile Devices with Thermal Constraints. IEEE TCAD 2019

Deep Learning on Embedded Heterogeneous SoC



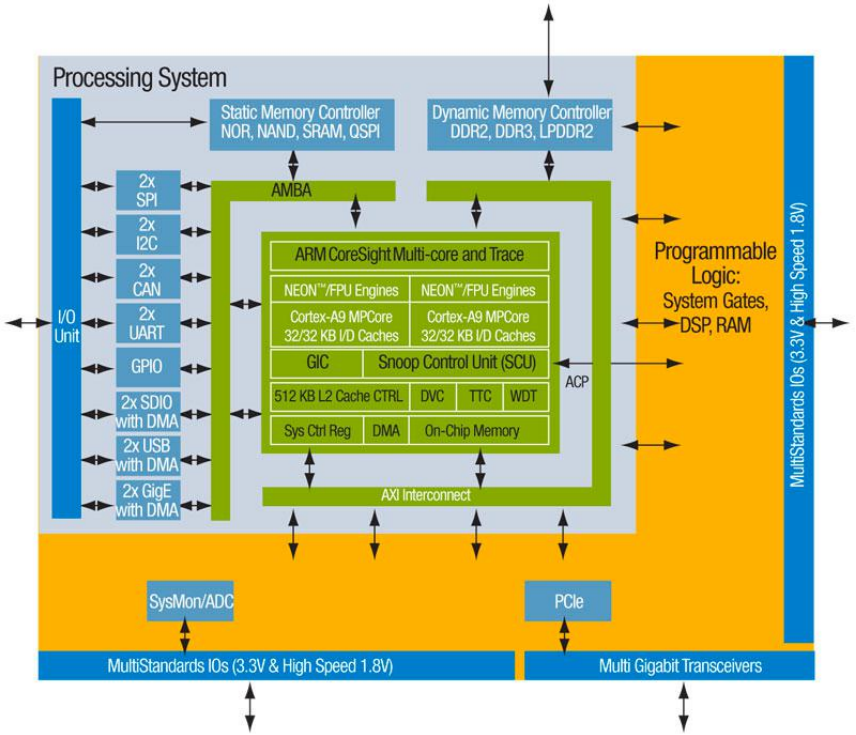
Huawei HiKey 970 Mobile Platform

HiSilicon Kirin 970
big.LITTLE CPU, GPU, NPU, ISP, Security Engine

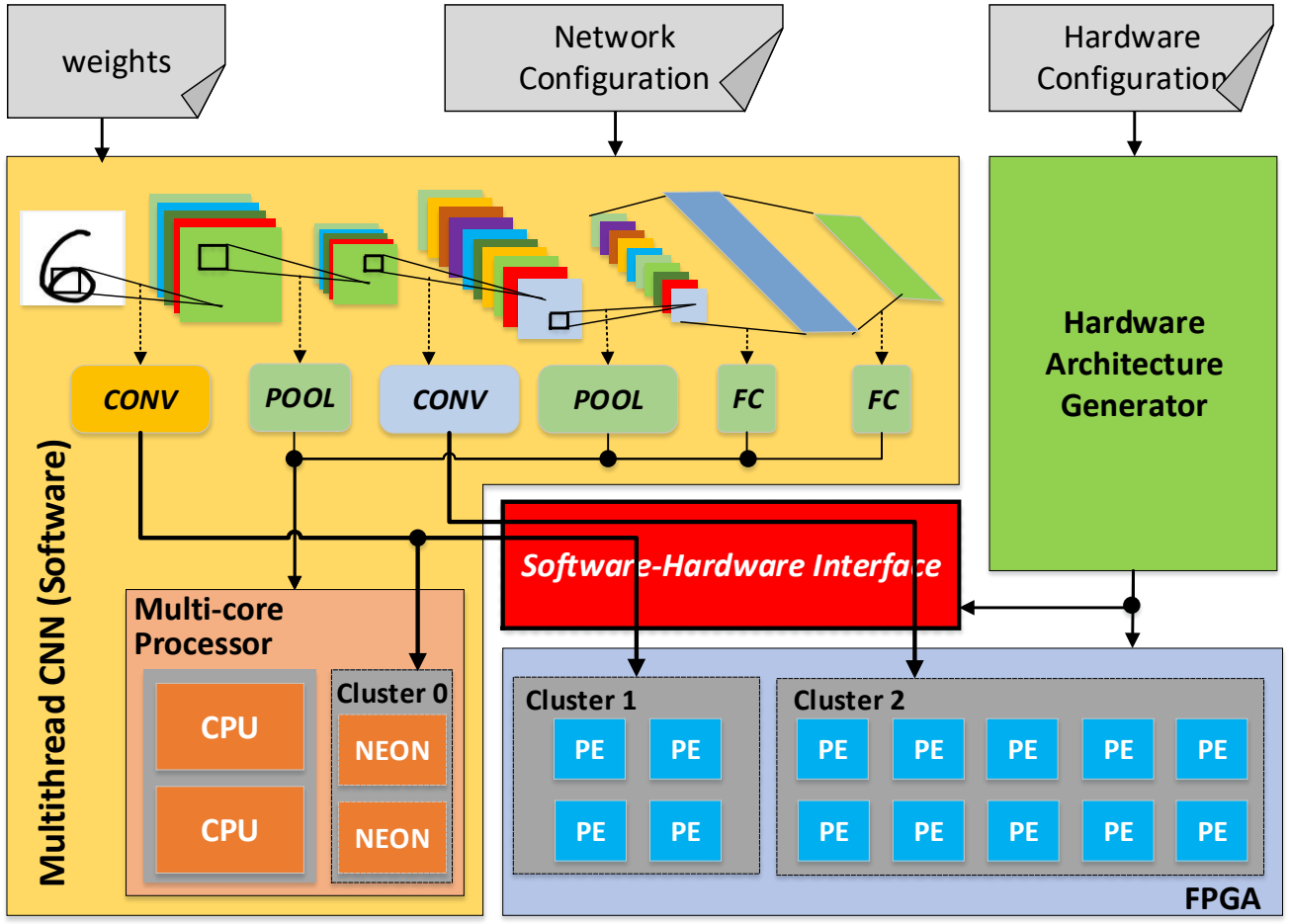
Image Credit: Huawei

CNN on CPU, FPGA, Neon

- Xilinx ZedBoard with Zynq XC7Z020 low-cost SoC
- 63.5 frames/sec for CIFAR
- 1.67 GOPS 32-bit floating-point
- 54 mJ/frame
- Highest throughput, energy-efficiency at low-cost

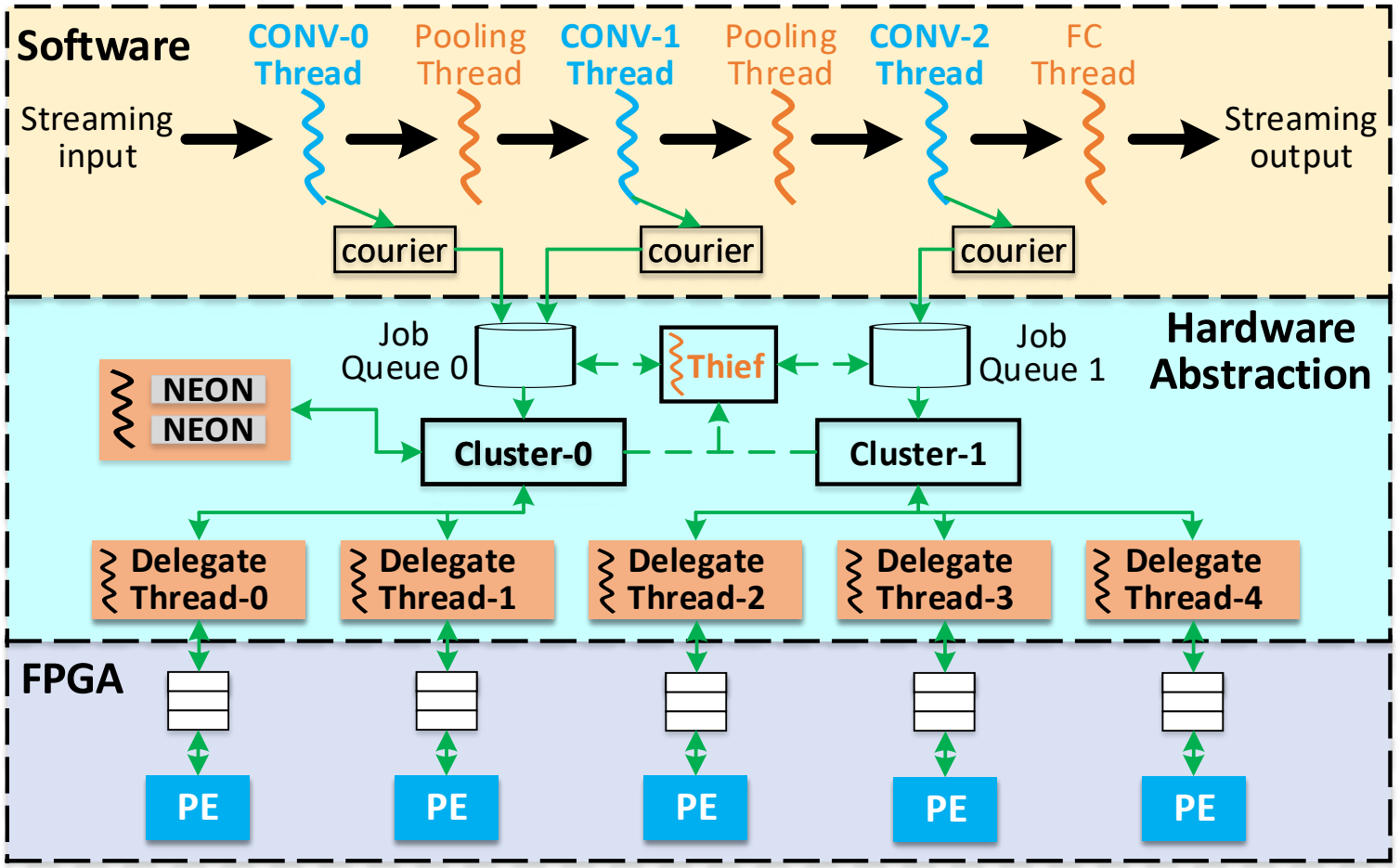


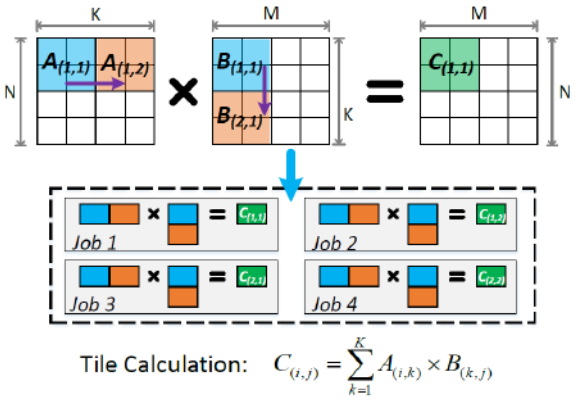
Synergy Framework



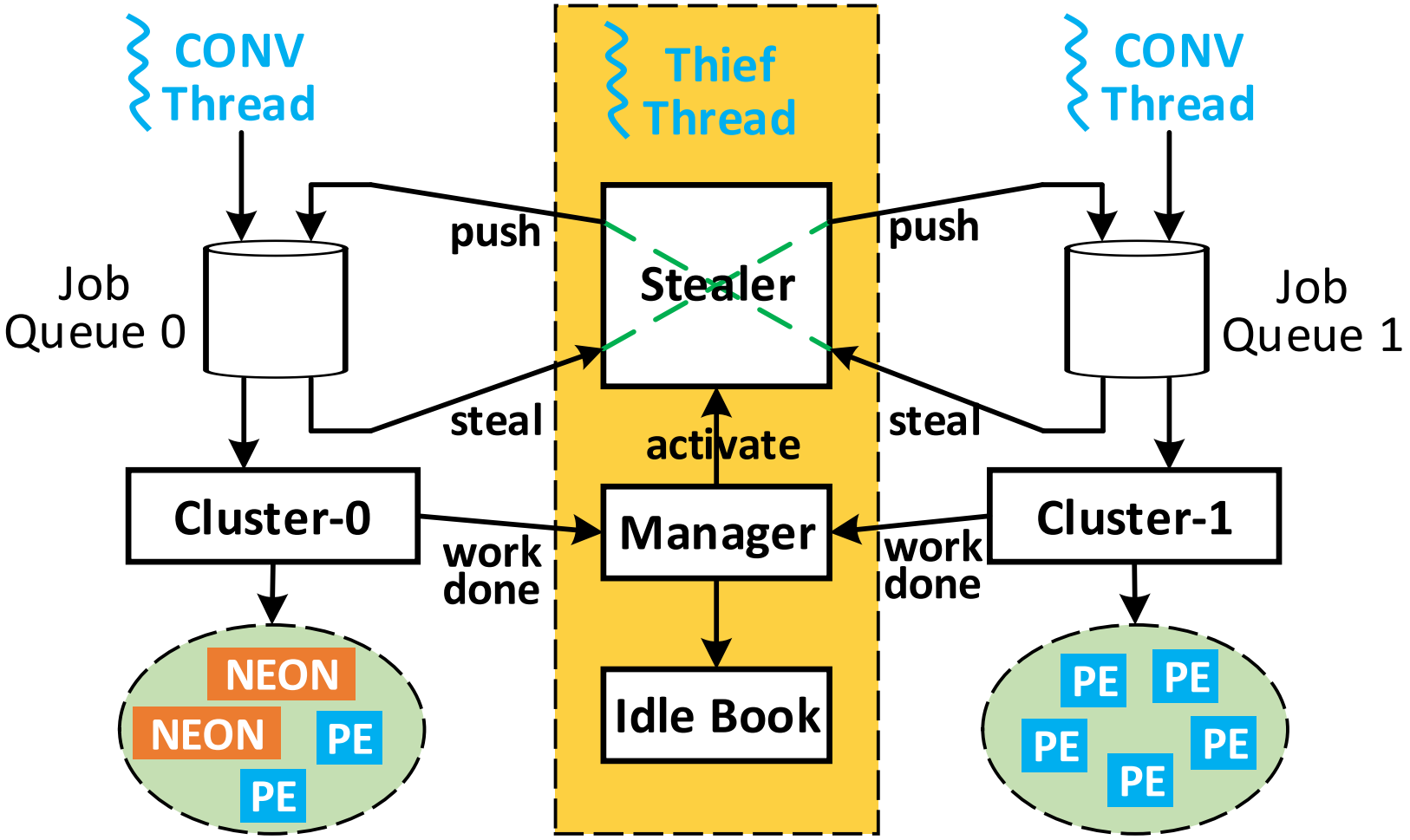
Zhong, Guanwen et al. "Synergy: A HW/SW Framework for High Throughput CNNs on Embedded Heterogeneous SoC" *ACM TECS 2019*

Overview of Synergy Execution Flow

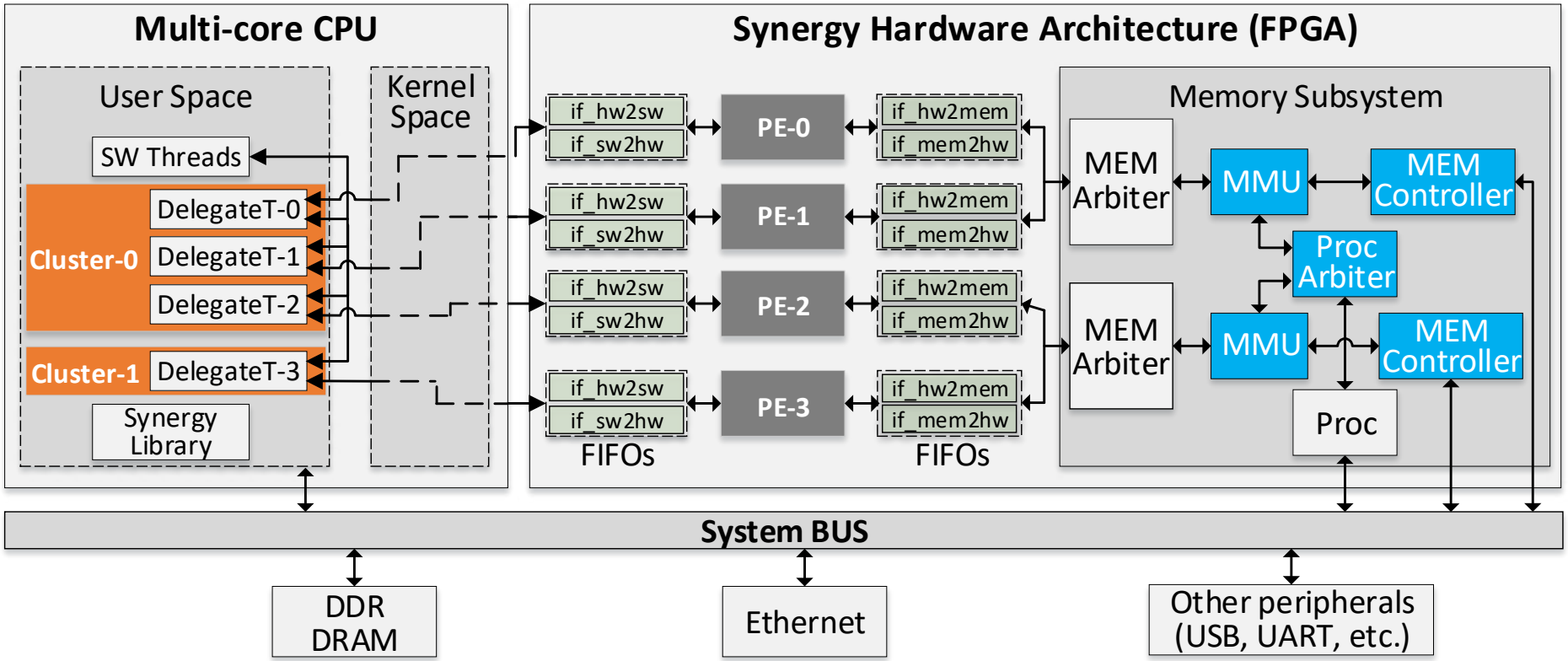




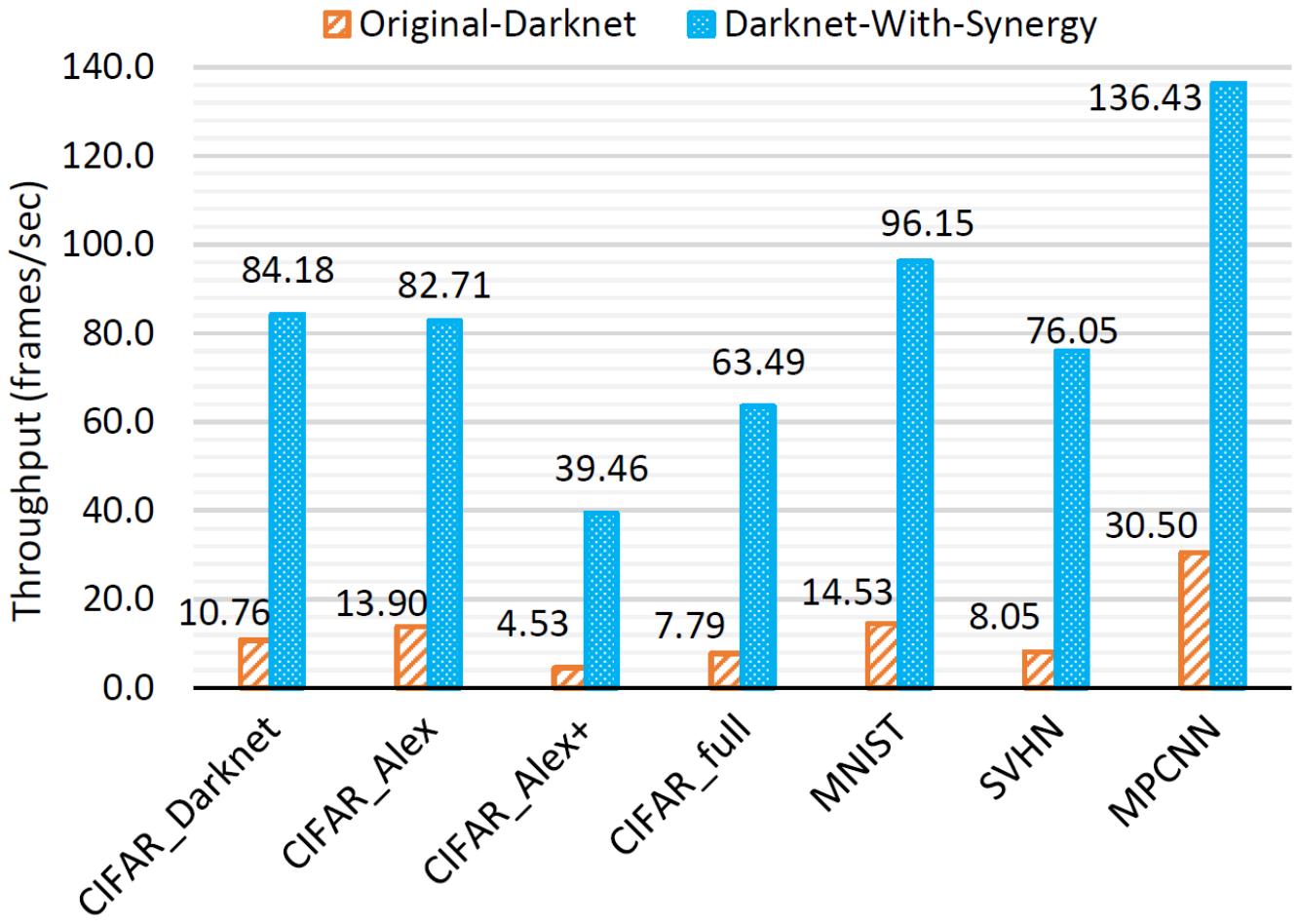
Self-balancing: Work Stealing



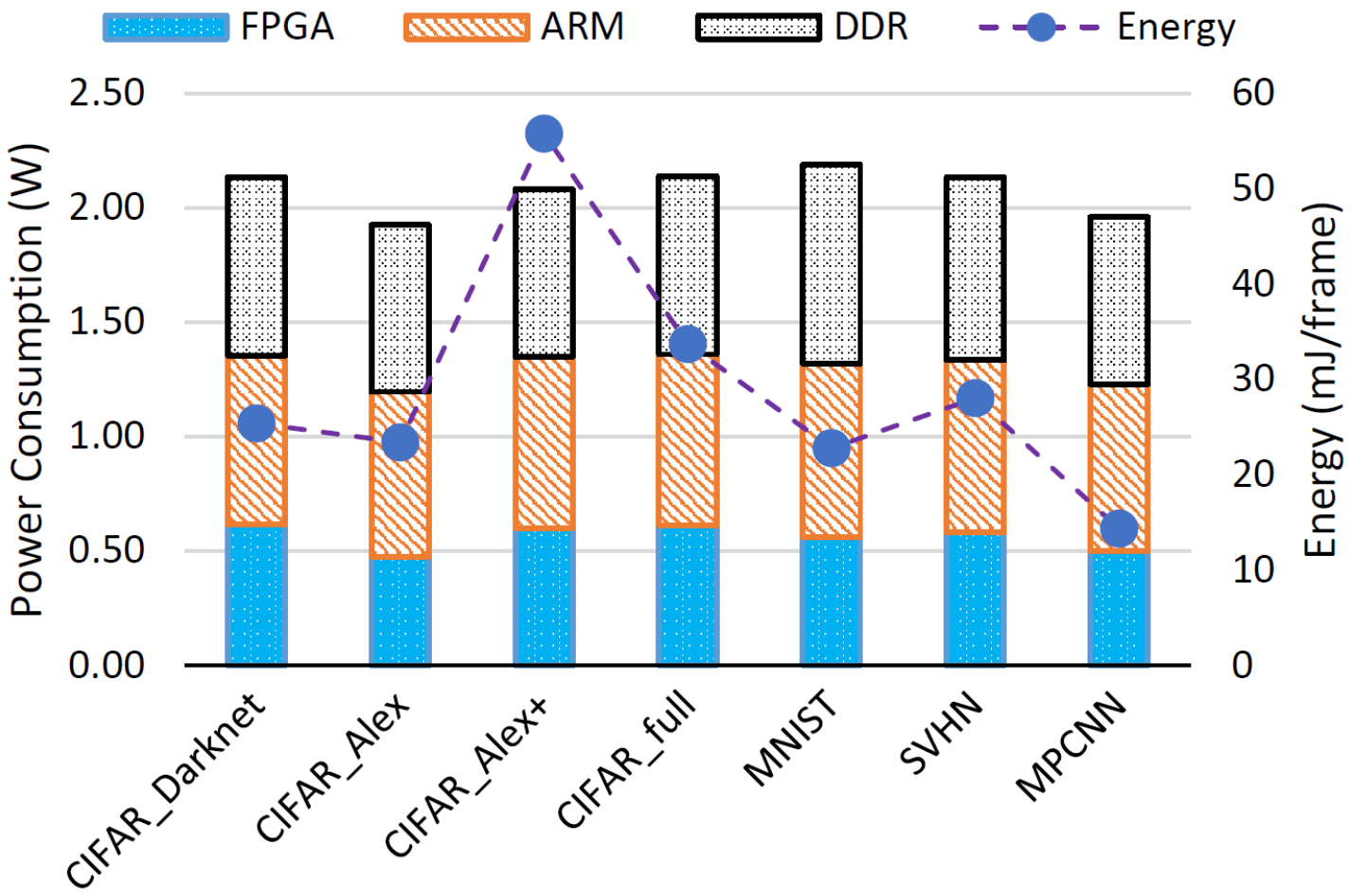
HW Architecture



Throughput improvement using Synergy

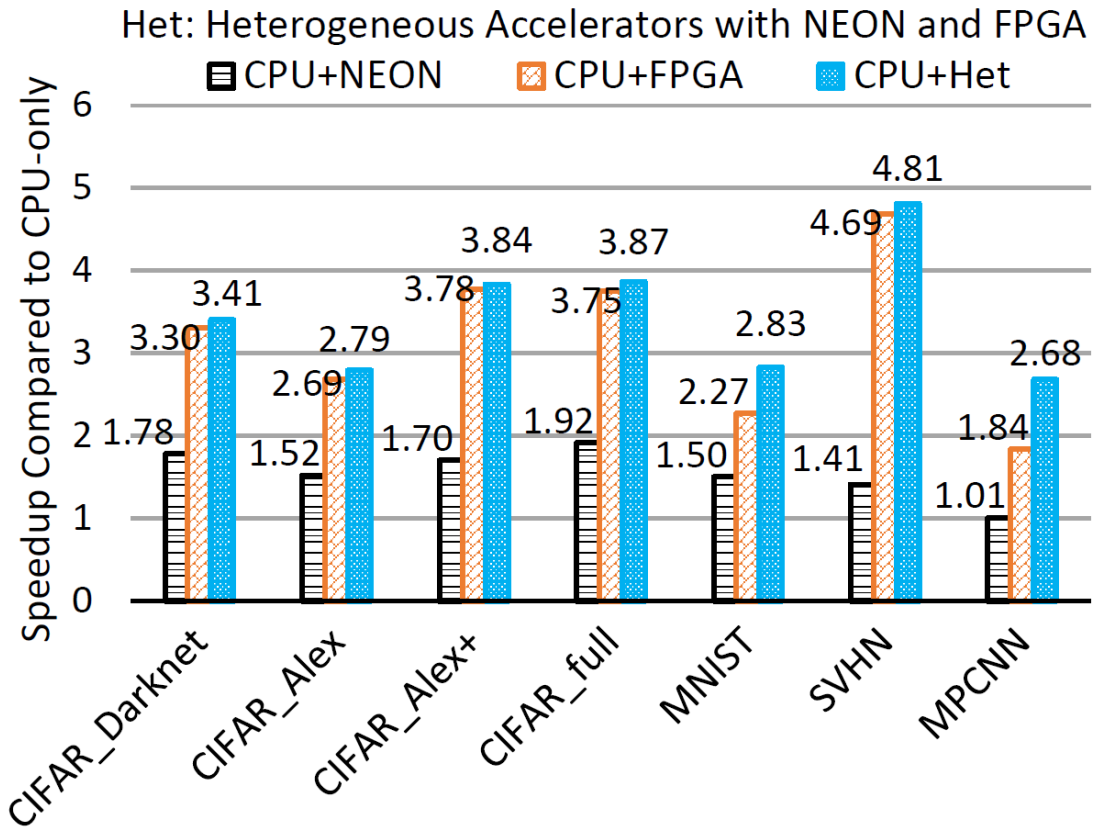


Power Distribution and Energy Consumption



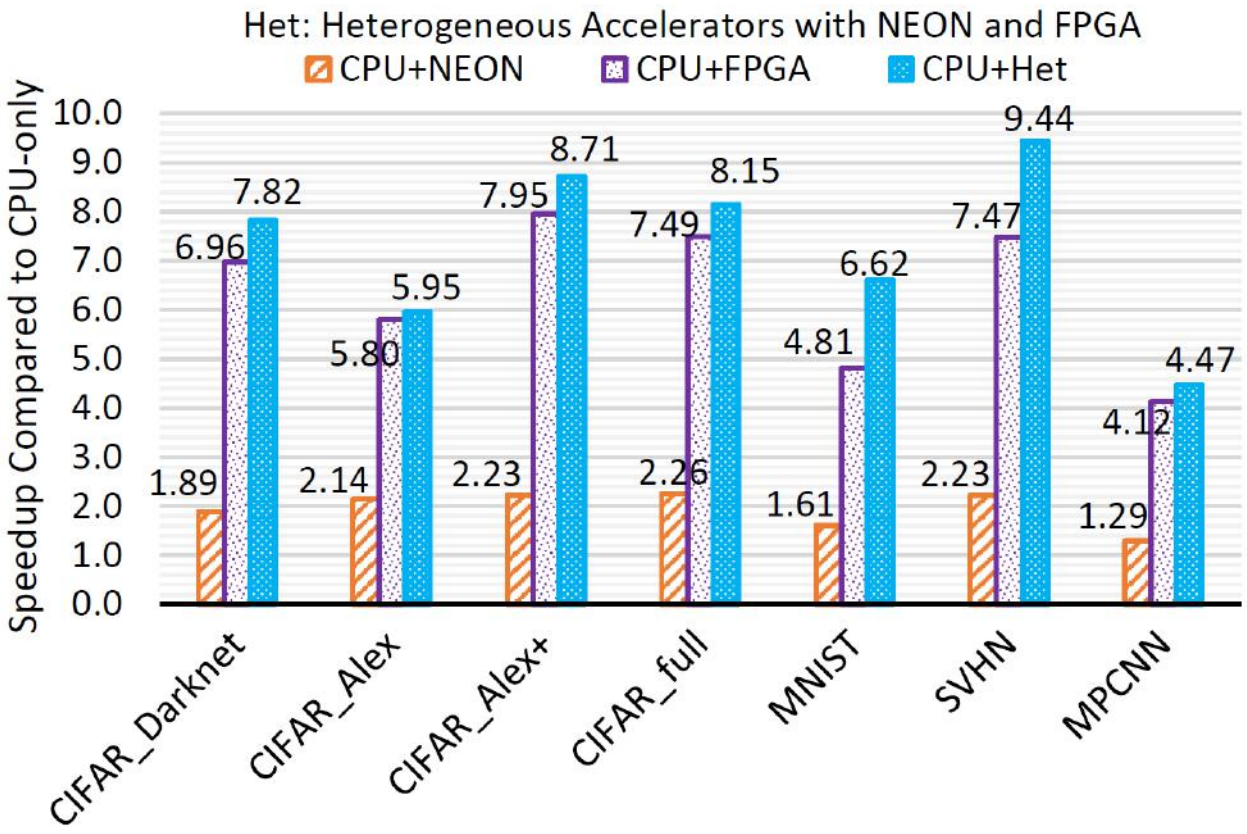
Impact of Heterogeneity

- **Latency** improvement with accelerators compared to CPU-only solutions for non-pipelined designs

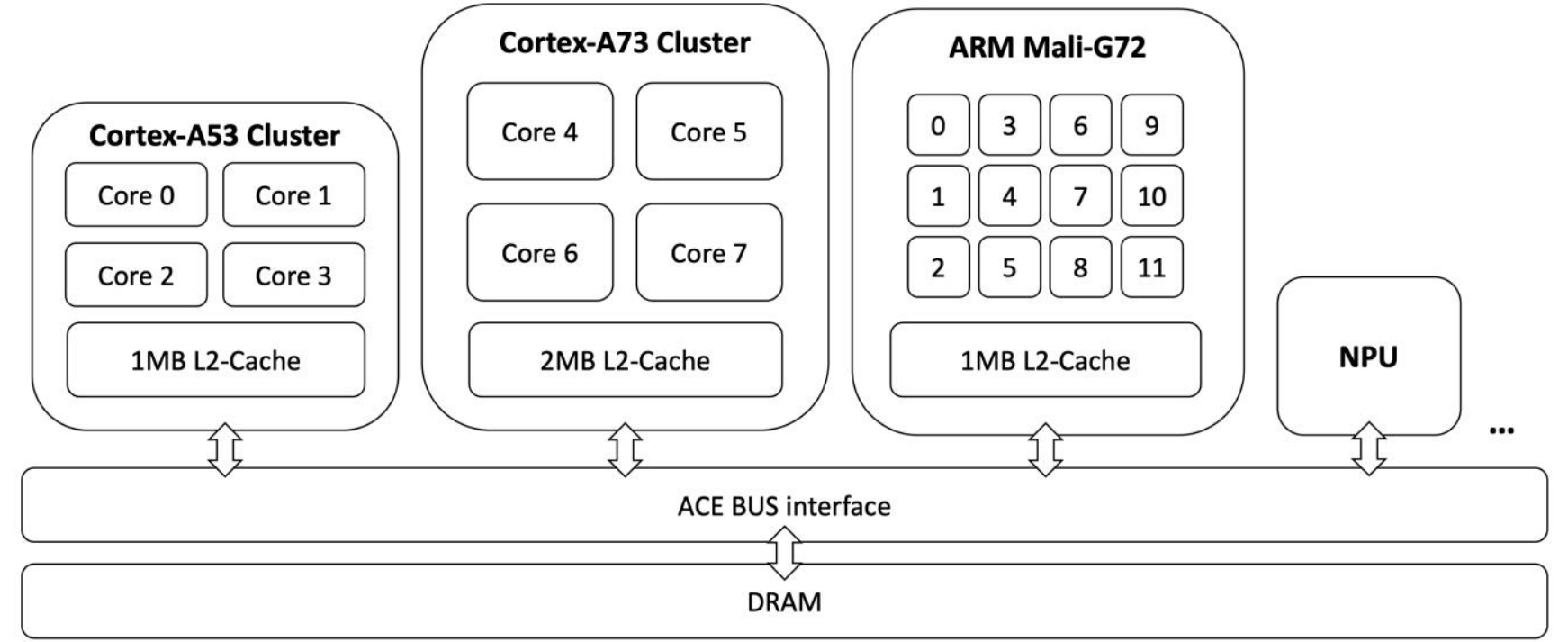


Impact of Heterogeneity

- **Throughput** improvement with accelerators compared to CPU-only solutions for pipelined designs

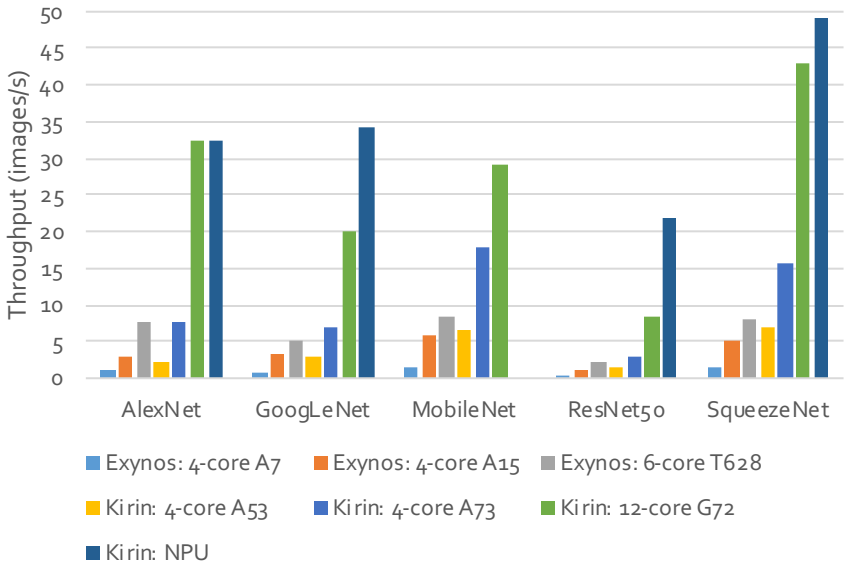


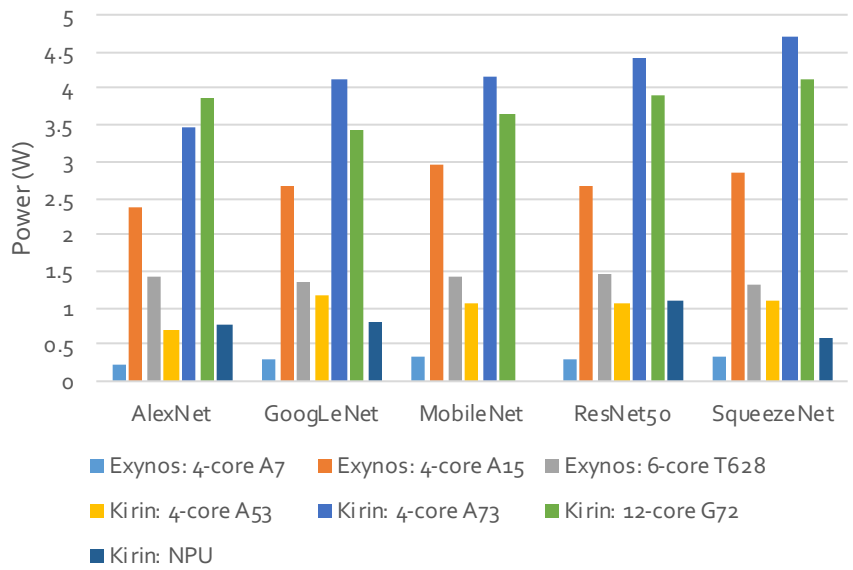
Mobile SoC Closeup



Performance and Power Characteristics

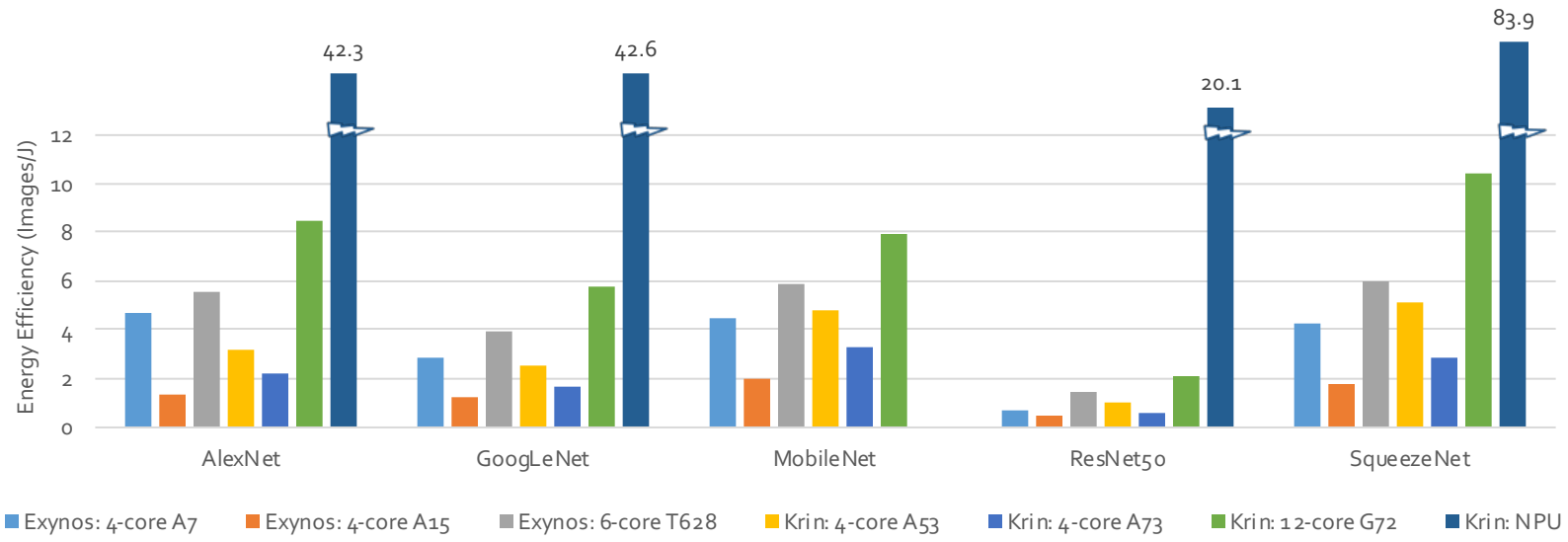
- GPU is comparable to NPU
- Performance gap between GPU and CPU clusters is about 2.5X
- High-end GPU comsumes high power comparable to big CPU cluster
- Low-end SoCs have no single component sustaining ML workloads





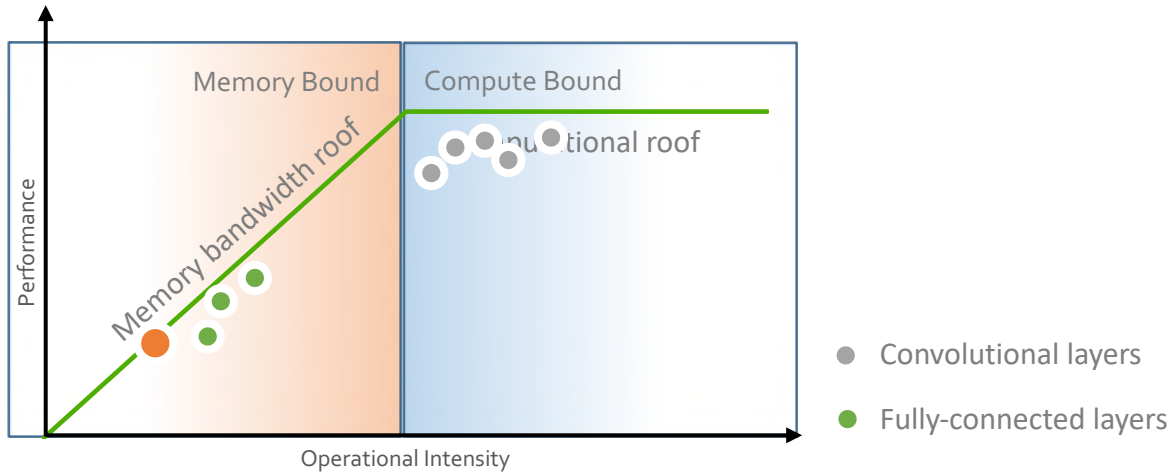
Energy-Efficiency Characterization

- Energy efficiency is defined as Images/Joule
- NPU is superior in both performance and energy efficiency



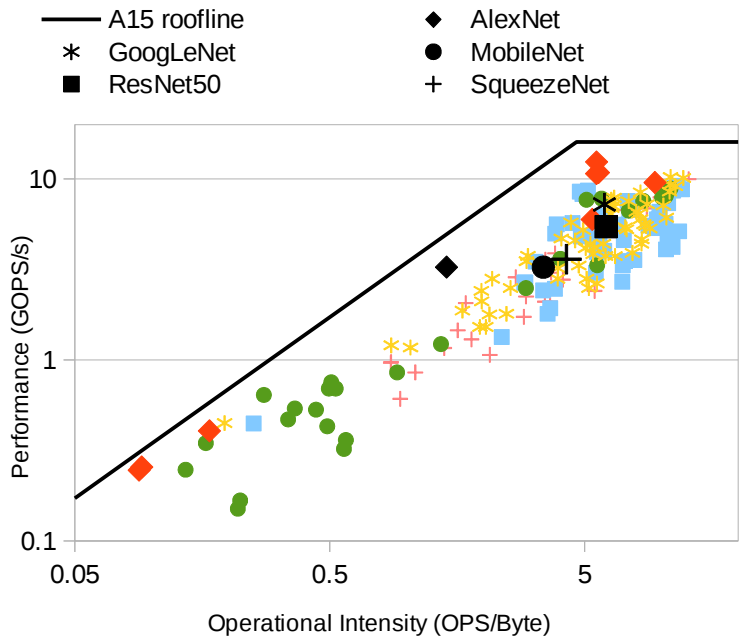
Roofline Analysis

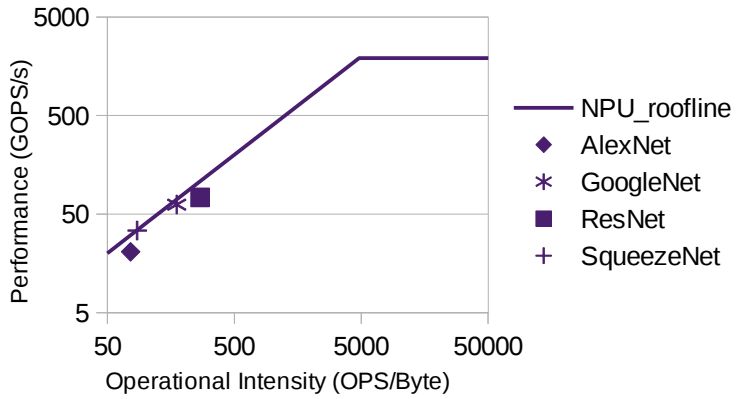
- Theoretical vs Empirical operational intensity (OI)
  - Theoretical OI captures application characteristics
  - Empirical OI reflects actual platform performance
    - Multiple level of caches moves the layers further to the right
  - Fully-connected layers have lower OI compared to convolutional layers



Roofline Analysis

- Within an application, layers behave differently
- NPU is always in the memory bound region

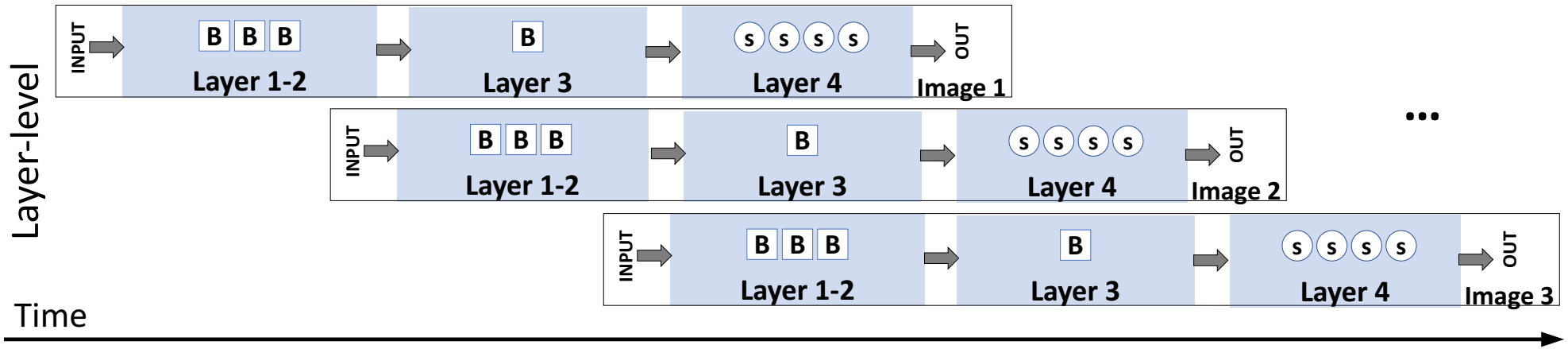


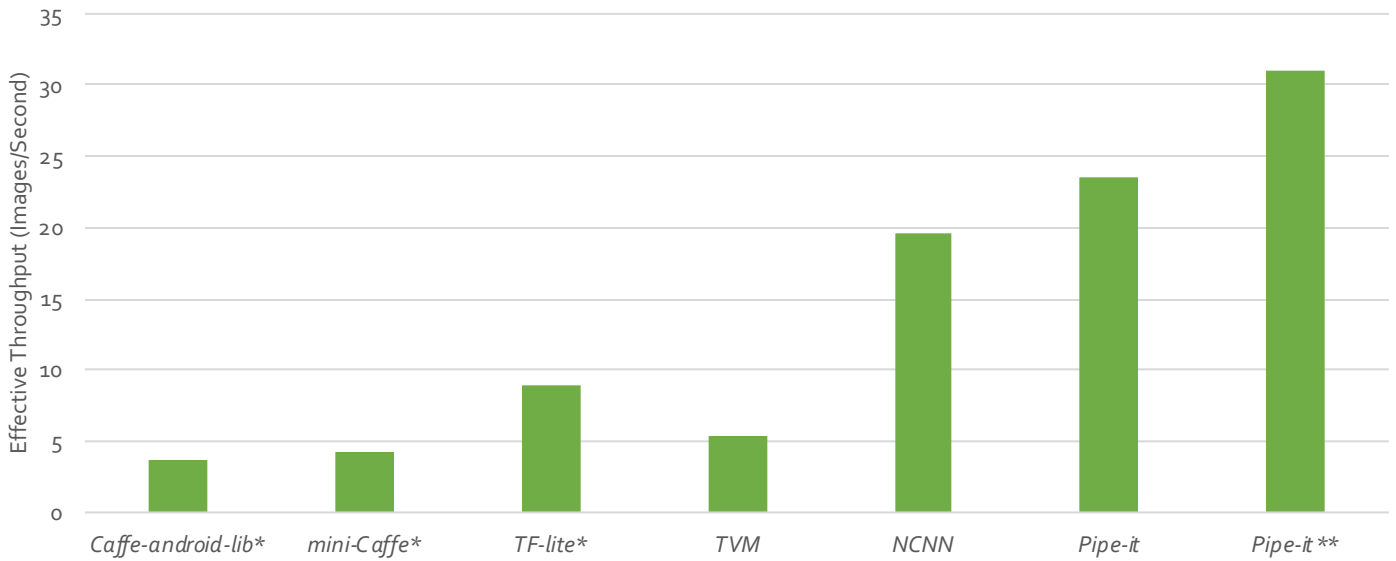


Synergistic Coarse-grained Co-execution

Siqi Wang et al. "Neural Network Inference on Mobile SoC" *IEEE Design & Test 2020*

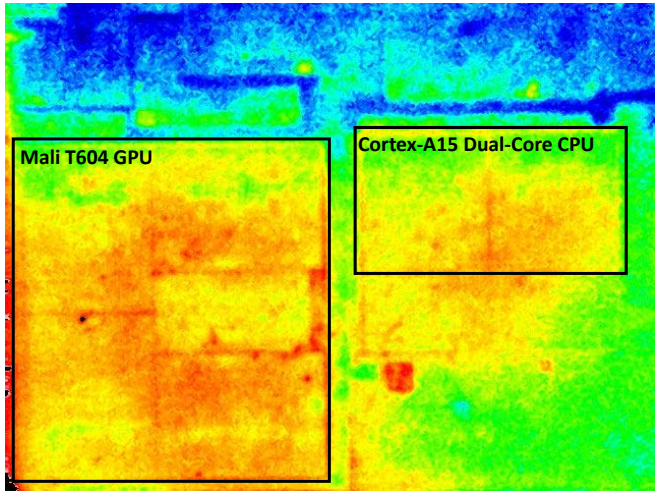
Pipe-It: Synergistic Pipelined Execution on ARM big.LITTLE





High-Throughput CNN Inference on Embedded ARM big. LITTLE Multi-Core Processors. TCAD 2020

RUNTIME POWER-THERMAL MANAGEMENT



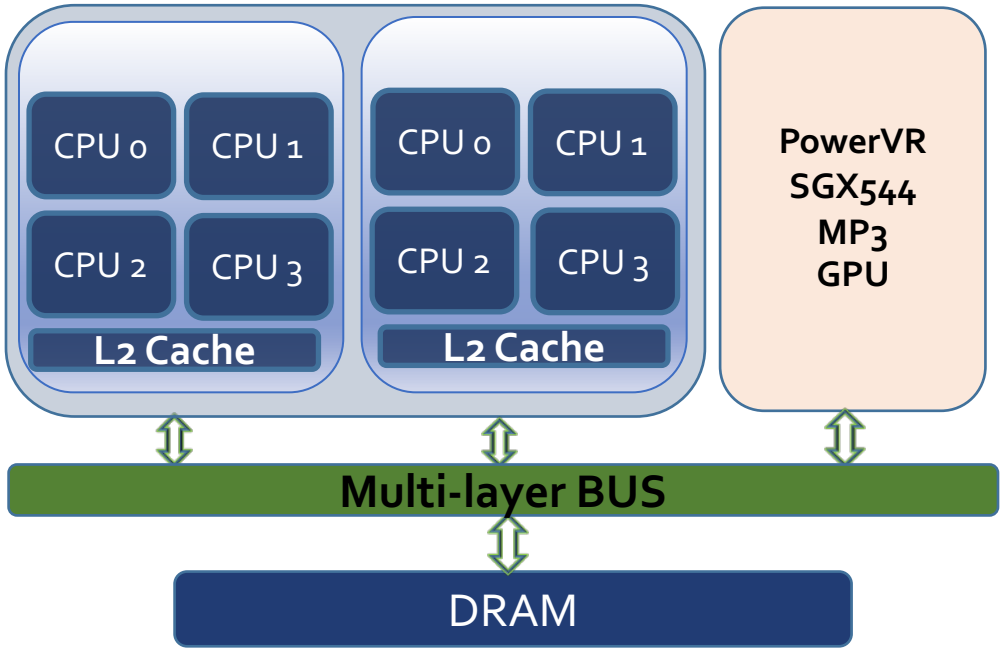
Hardware side of the story

- Hardware design aims to provide the appropriate set of power management knobs
- Heterogeneous cores/clusters
  - Functional, Micro-architectural, Interconnect, Reliability, Technology, Process-variation induced heterogeneity
- Multiple voltage-frequency levels per cluster
- Multiple power states per cluster

Performance + Functional Heterogeneity: Exynos 5422 SoC

Low Power, In Order **LITTLE Core** (ARM Cortex™-A7 Quad 0.2-1.2 GHz)

High Performance, Out of Order **big Core** (ARM Cortex™-A15 Quad) 0.8-1.6 GHz



GPU (Imagination PowerVR Multi-core 177-640 MHz)

Multiple avenues to save power

Software side of the story

- Constraint: Maintain total chip power below Thermal Design Power (TDP)

- Knobs: Orchestrate in synergistic fashion

  Task to core mapping and migration

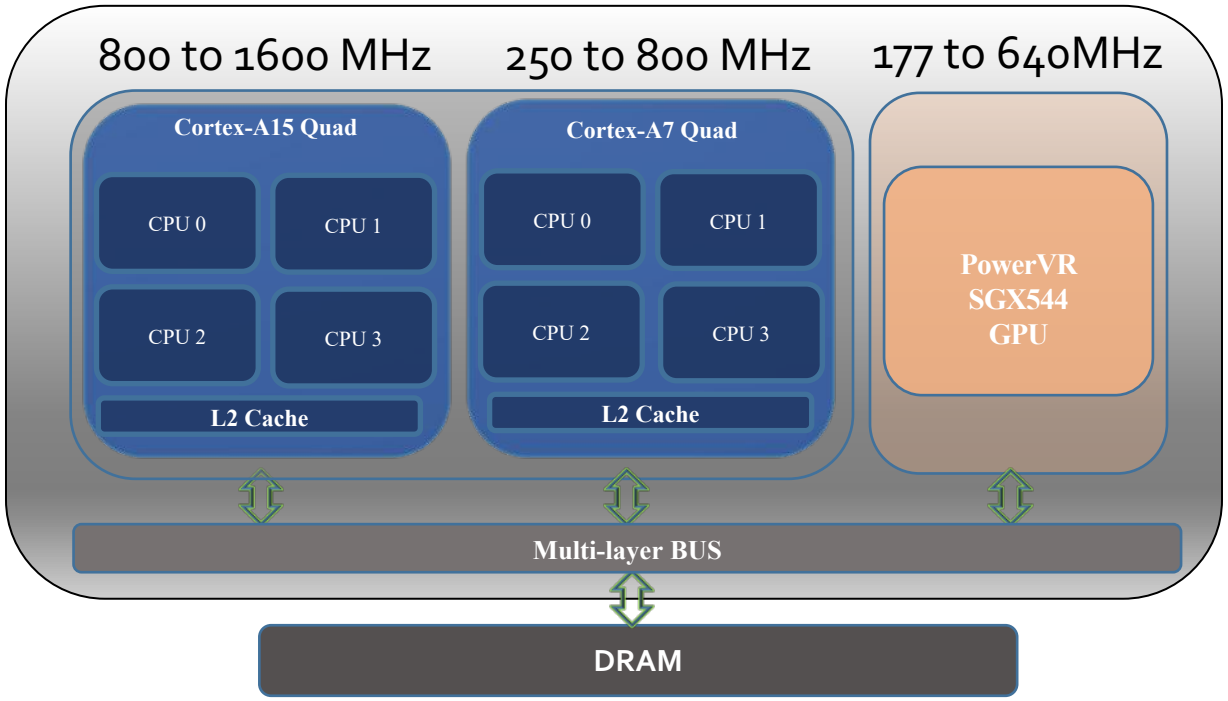
  DVFS and power states per core/cluster

- Goal: Provide best user experience while minimizing energy

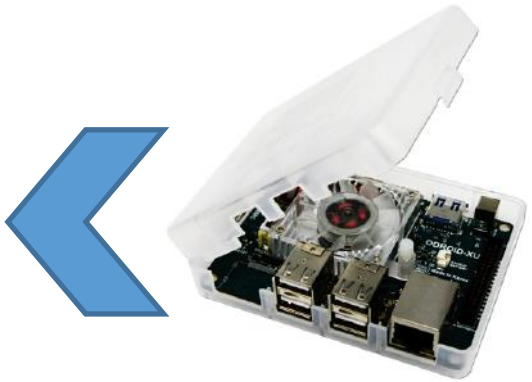
User experience in Interactive applications

- Quality of Service (QoS) is the key metric
  - Example: Frames per second in games, video/audio encoding/decoding
- QoS should be maintained within a range for each task
- Going beyond the maximum QoS bound is wasteful from power/energy perspective

MPSoC with CPU + GPU



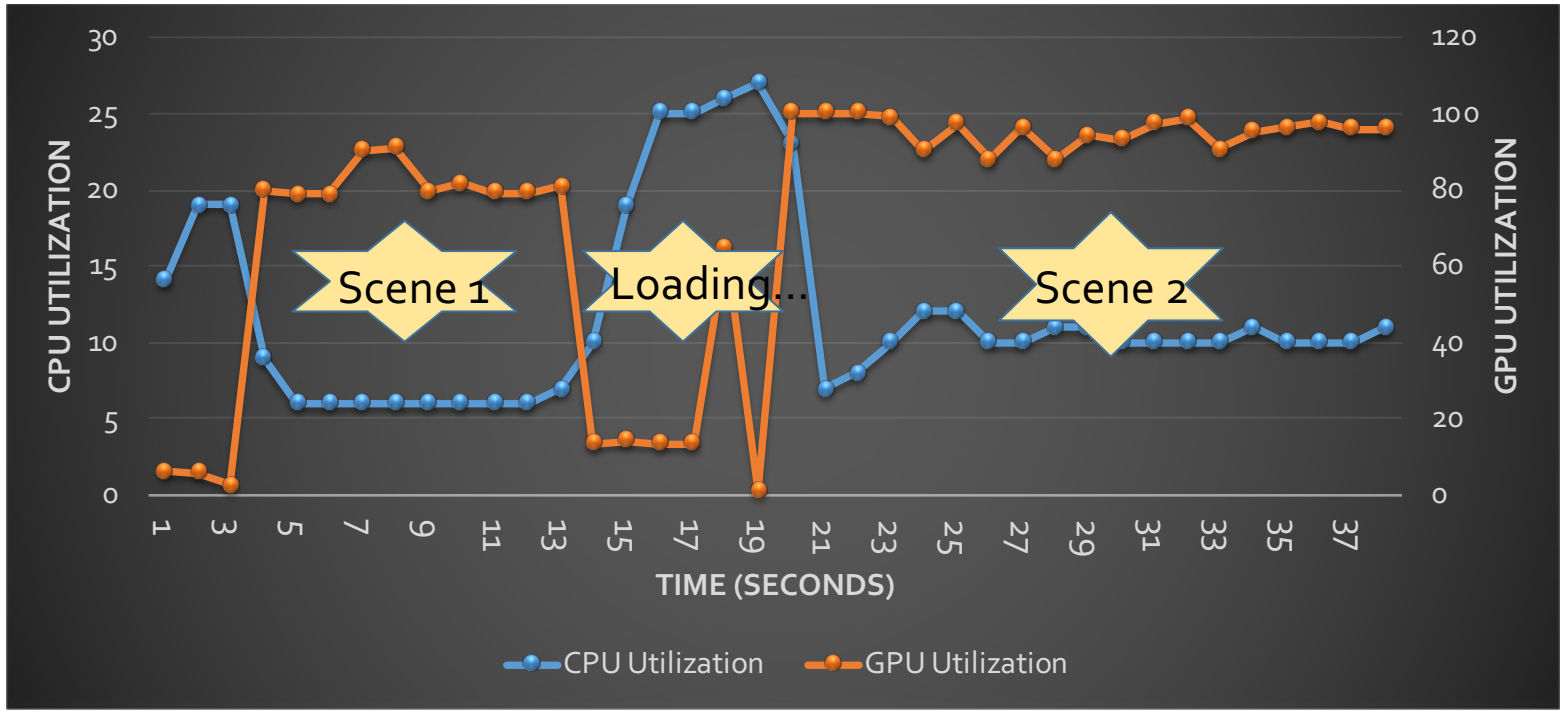
Exynos 5410 Octa SoC



Odroid XU+E

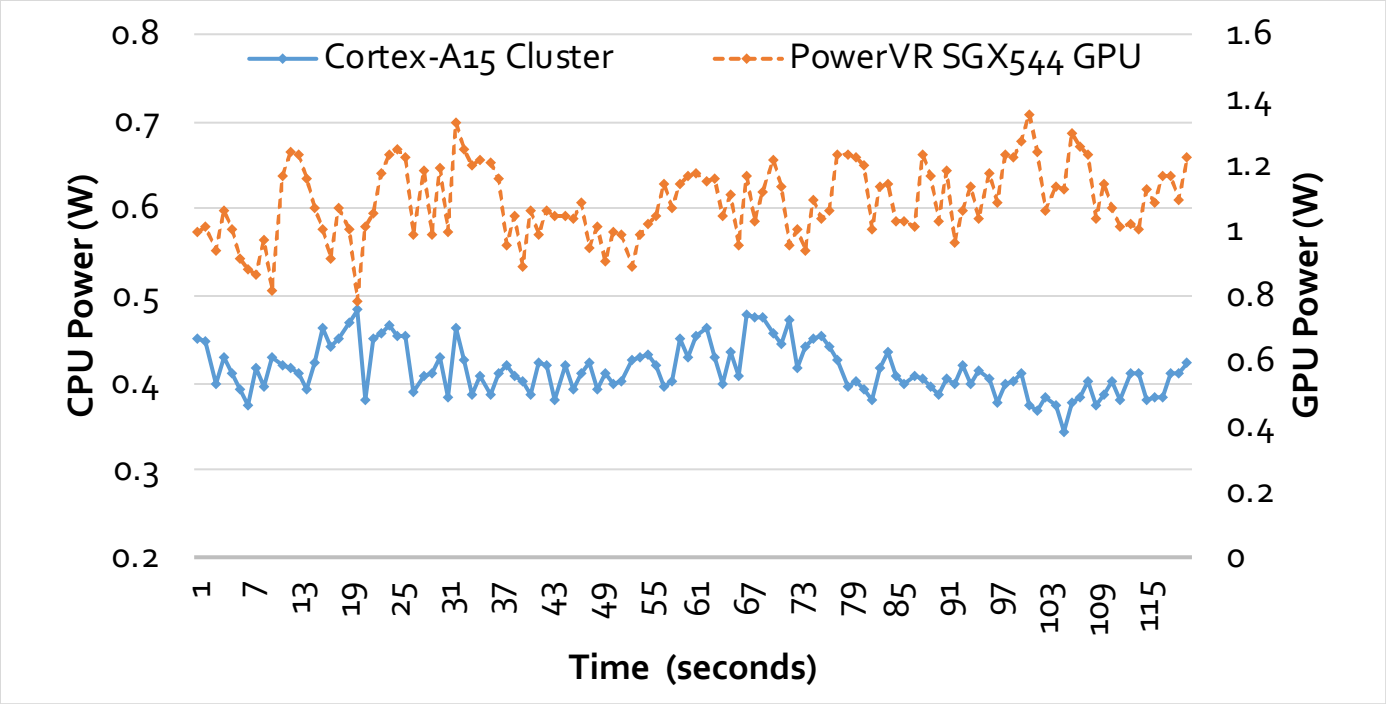
Galaxy S4

3D Gaming on Mobile Platform

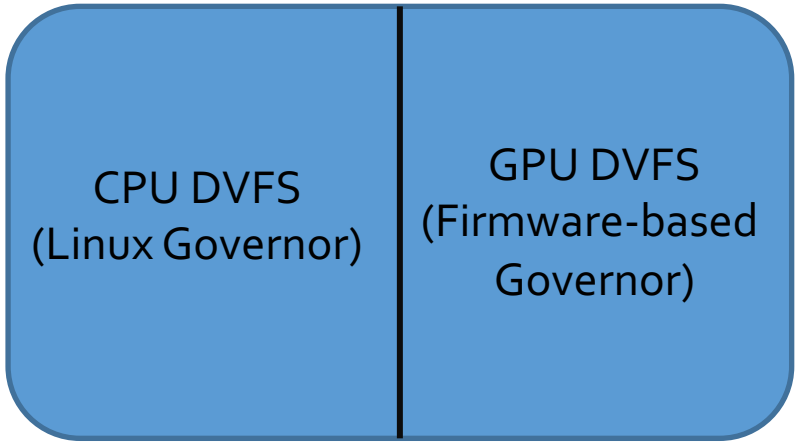


Pathania, Anuj, et al. "Integrated CPU-GPU power management for 3D mobile games." *DAC 2014*
Pathania, Anuj, et al. "Power-performance modelling of mobile gaming workloads on heterogeneous MPSoCs." *DAC 2015*
Prakash, Alok, et al. "Improving mobile gaming performance through cooperative CPU-GPU thermal management." *DAC* 2016

CPU and GPU Power Consumption

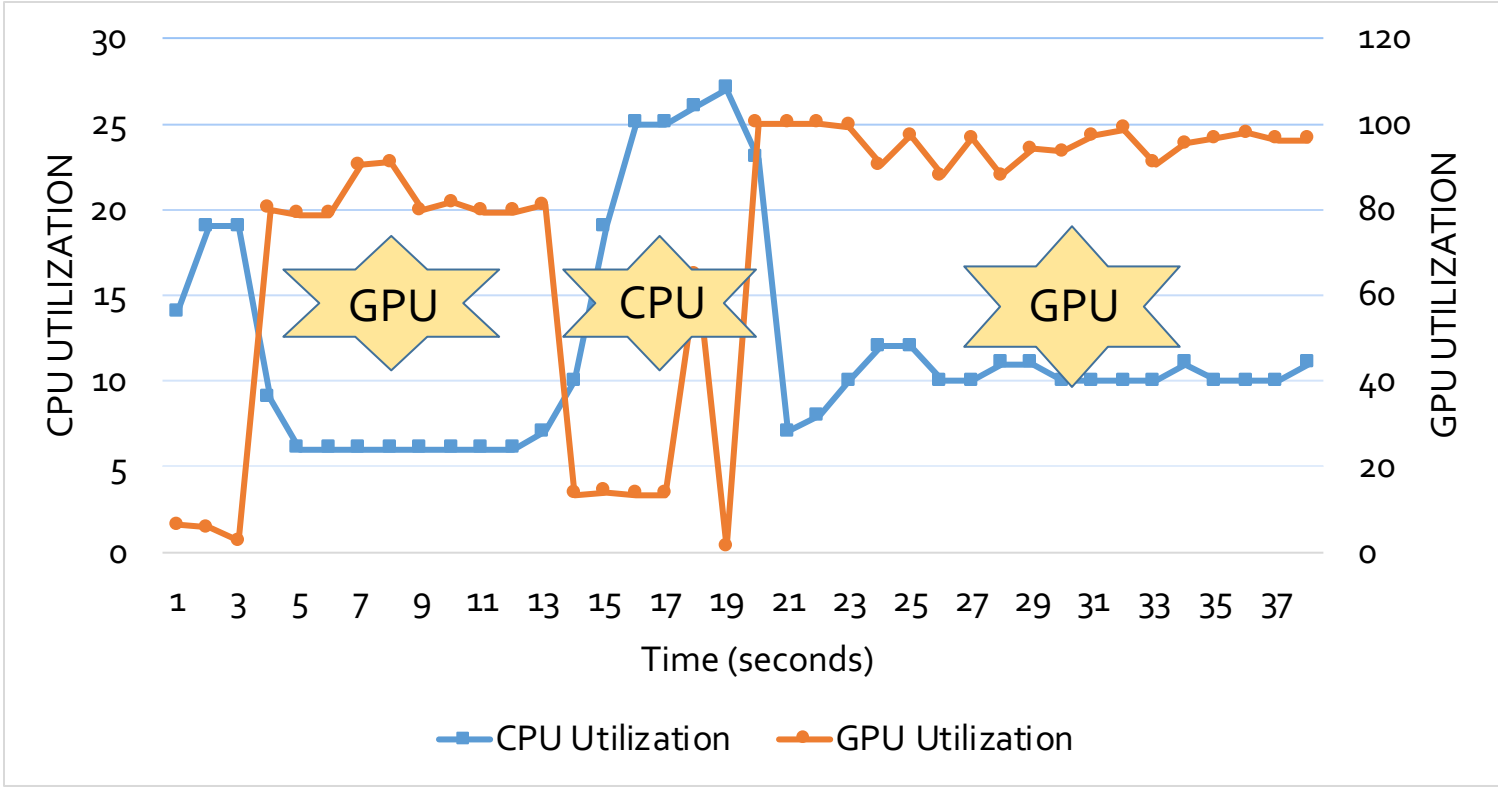


Android Power/Thermal Management



No synergy between the two DVFS managers!

Main Idea



Either CPU or GPU is bottleneck at any point in time

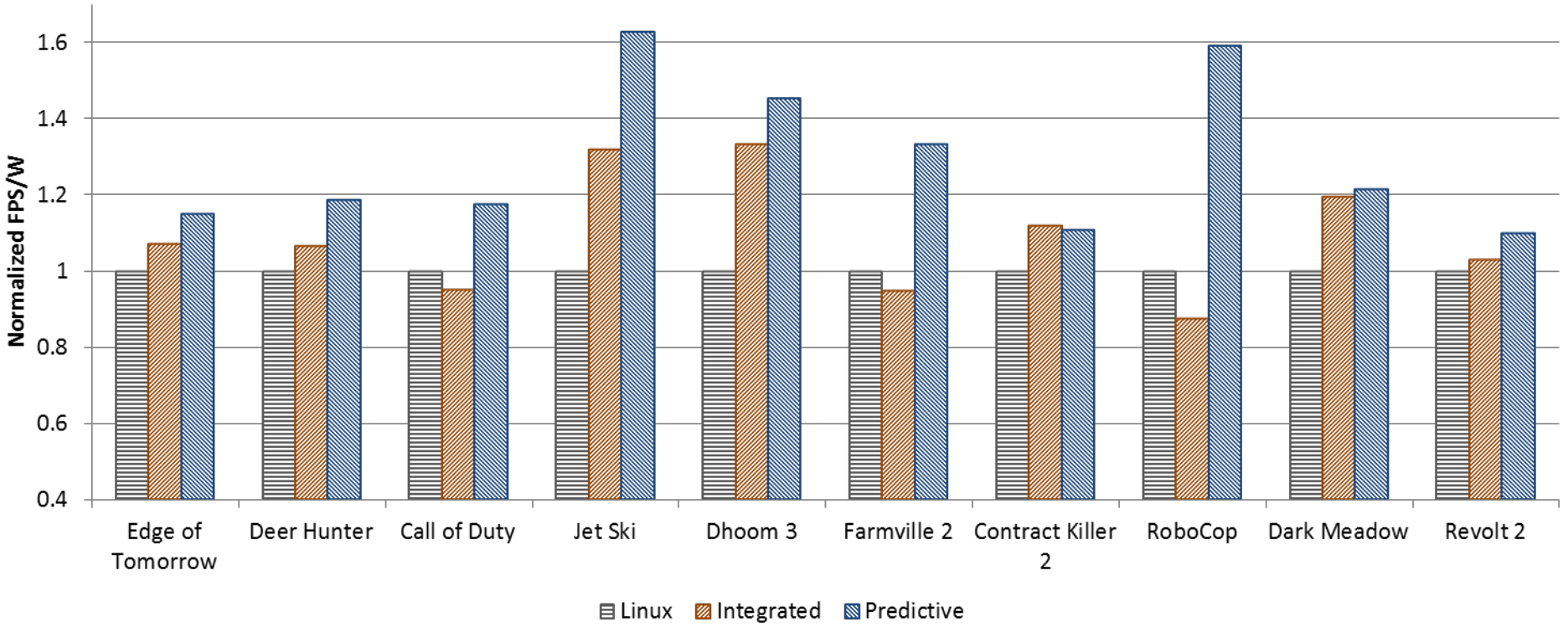
Reduce voltage-frequency for the other component

Challenges

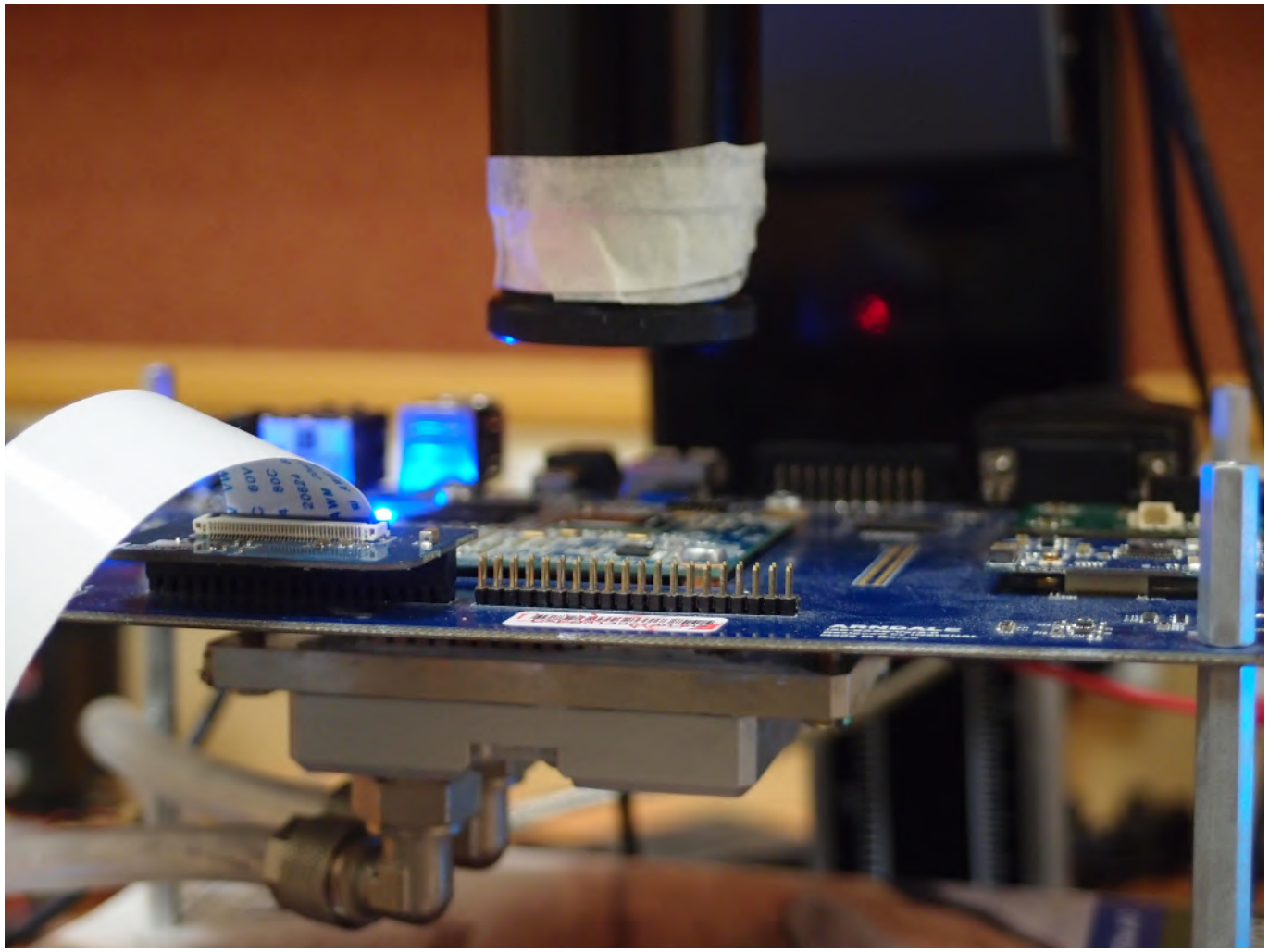
- Identify the bottleneck component
  - Utilization and performance counters

- Reduce voltage-frequency level of the non-bottleneck component appropriately
  - Predict entire system performance at different voltage-frequency level of the non-bottleneck component

Power Savings

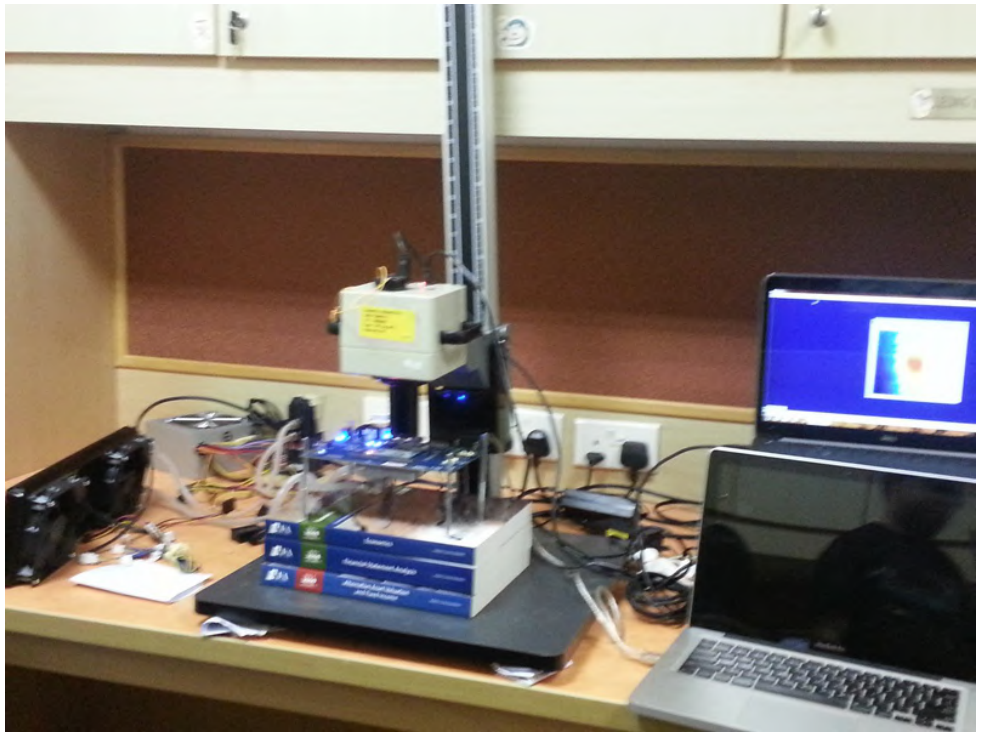


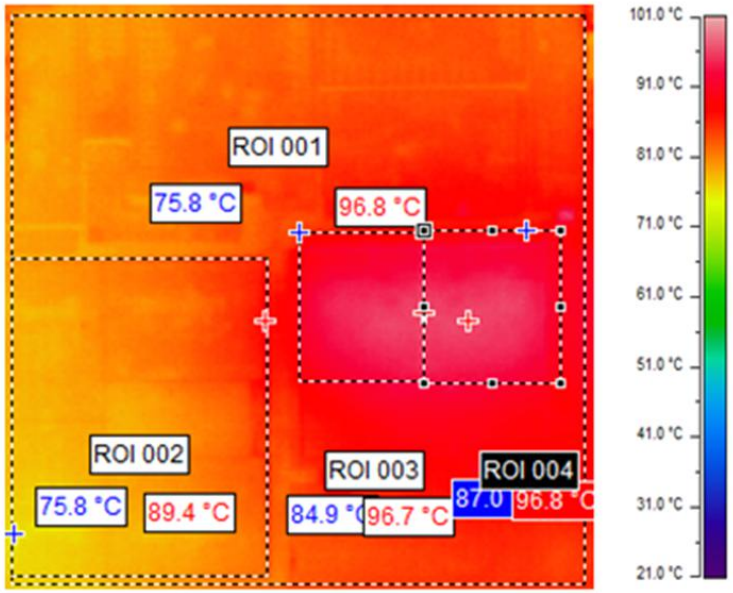
Thermal Camera with Heat-sink



Collaboration with Chair for Embedded Systems , KIT

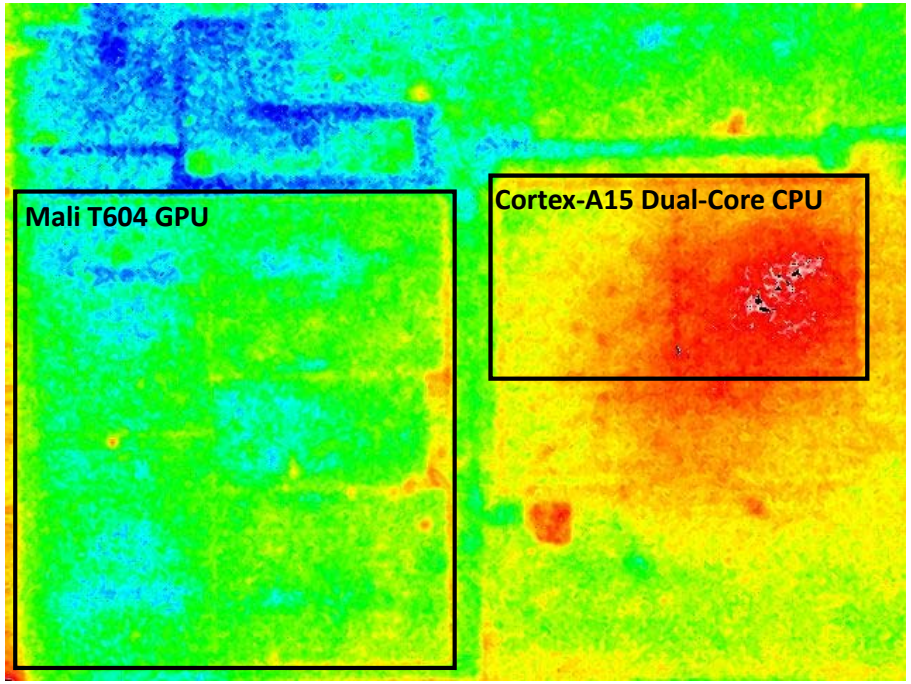
Thermal & Reliability Management

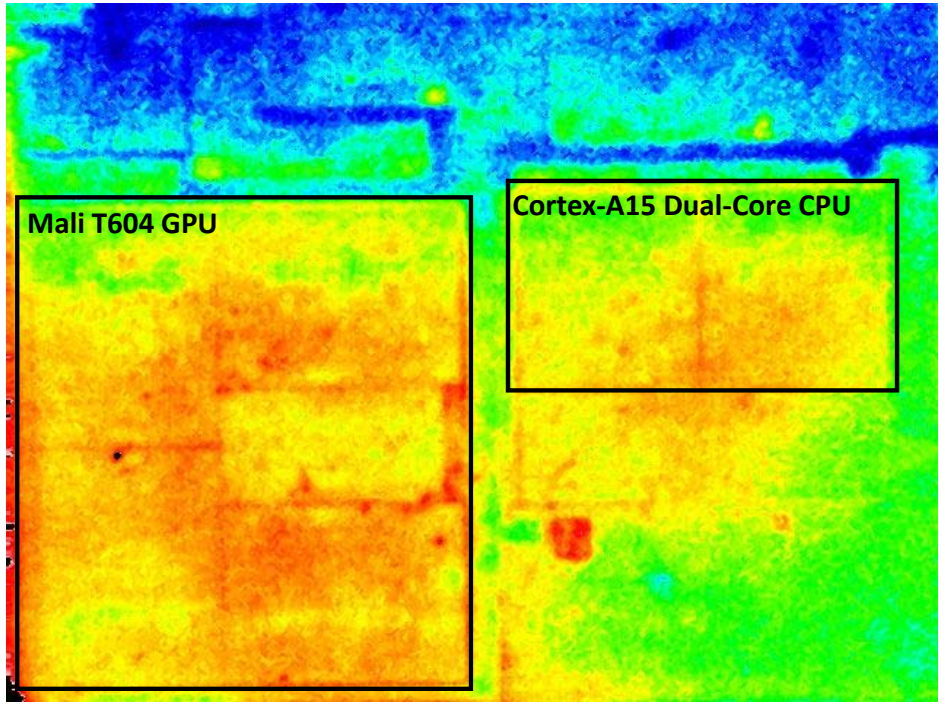


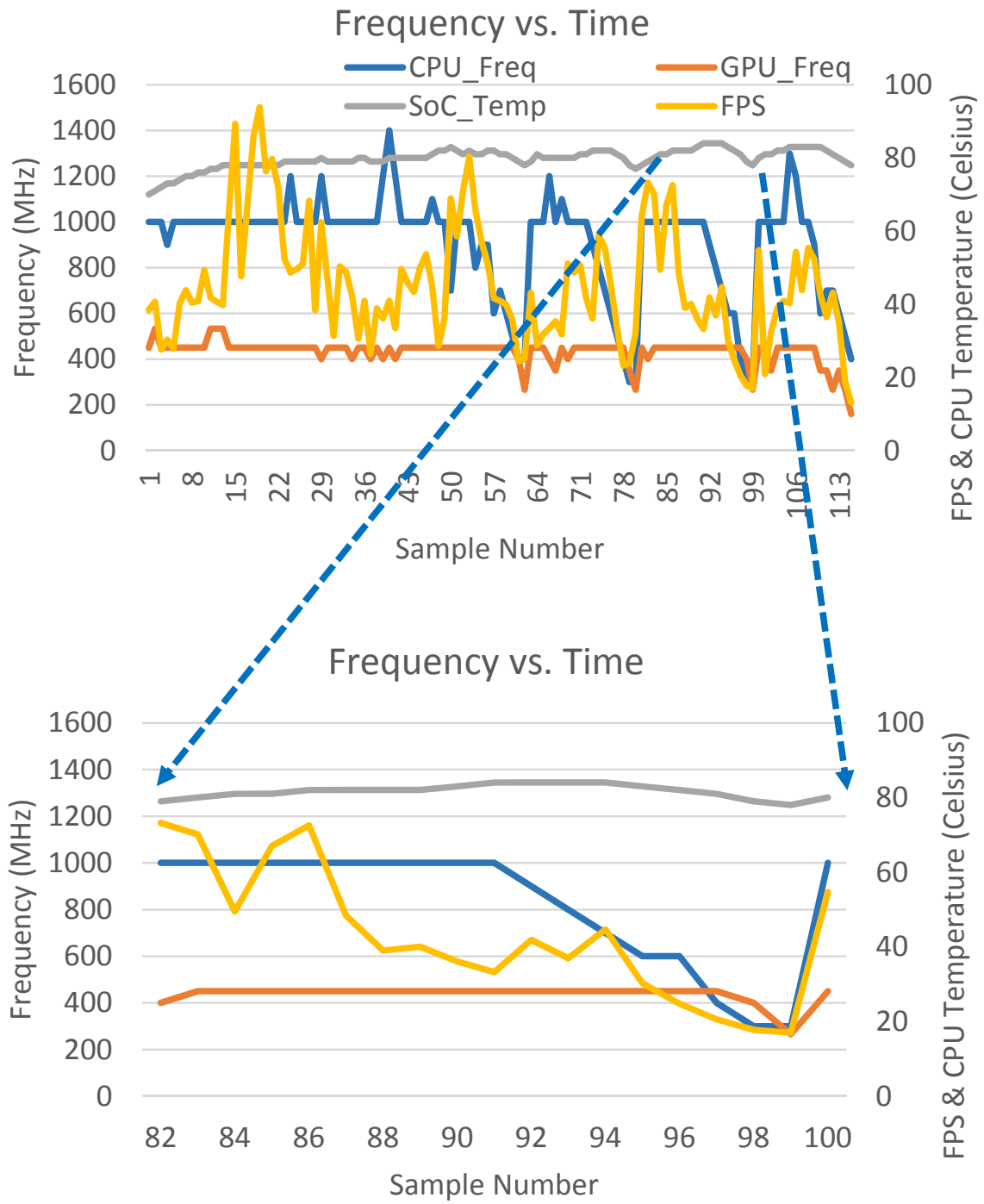


- Runtime management of workload for temperature and reliability

Hotspot moves between CPU, GPU



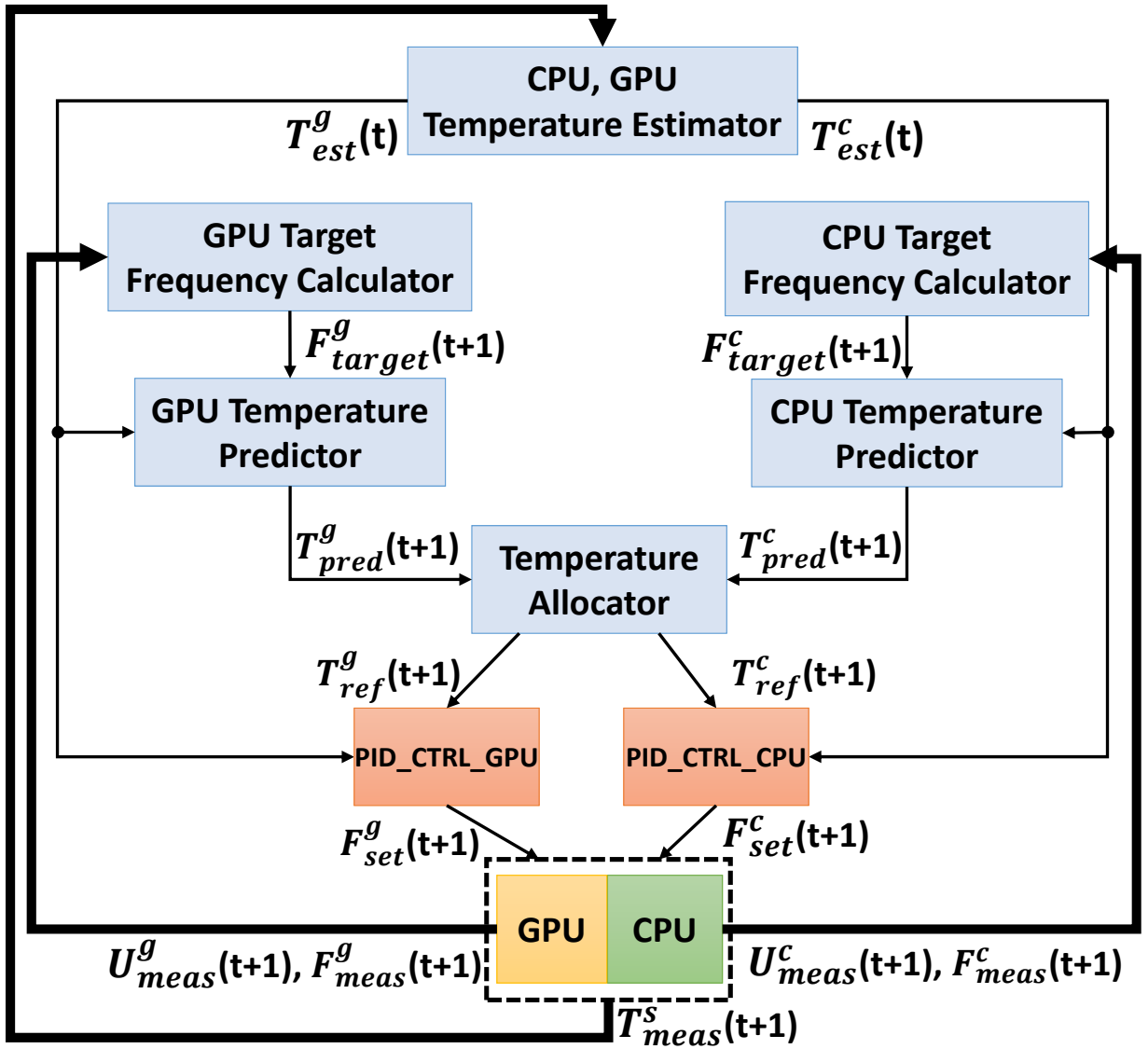




Android Behavior

- Recurring changes in CPU, GPU frequency, temperature
- High jitter, low FPS, high power, high peak temperature

Coordinated Thermal Management

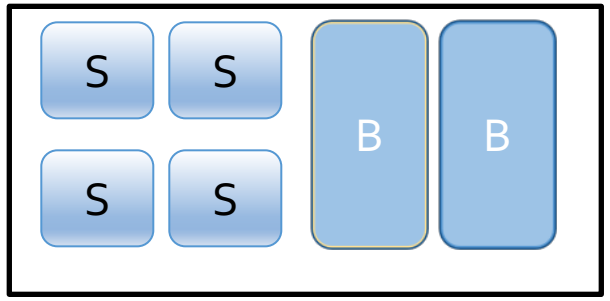


Results Highlights

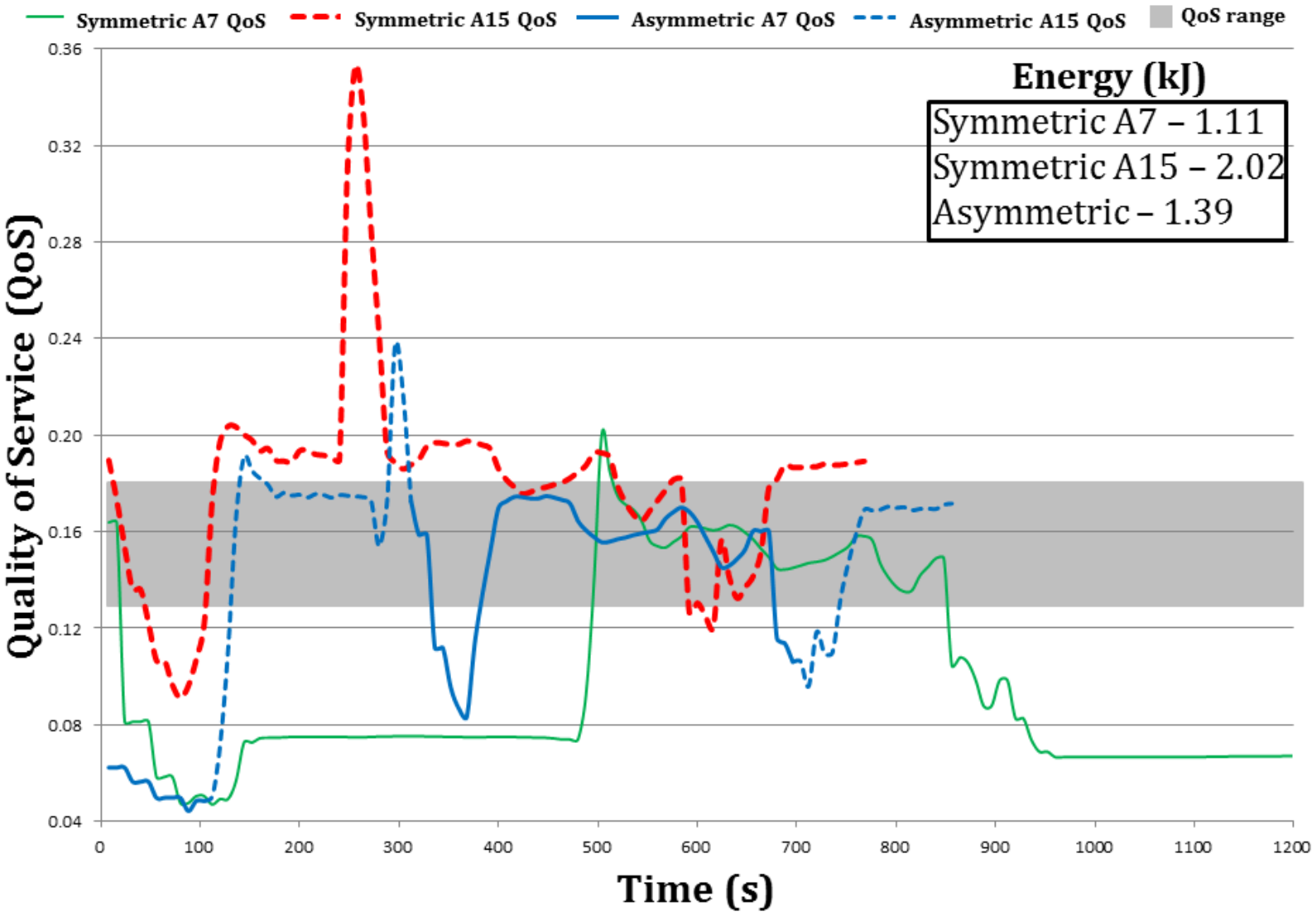
- Average 19% improvement in FPS over Linux
- Peak temperature reduced from 83°C to 77°C
- Minimal temperature variance compared to 10°C variance for Linux
- Similar average temperature

Smart Runtime Management for big.LITTLE

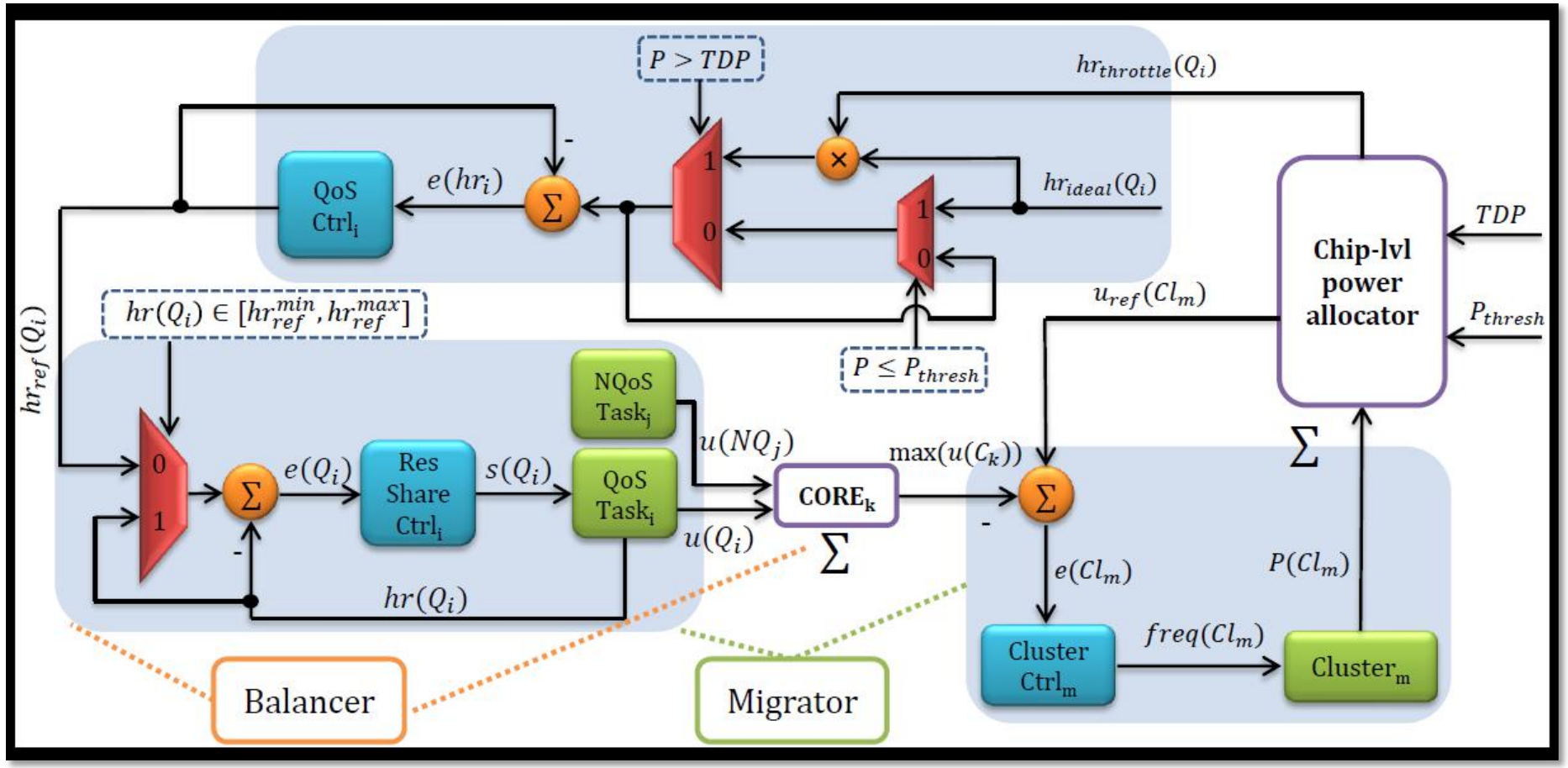
- Task to core mapping is decided per application basis and changes dynamically
- All the cores can potentially be active at the same time but at lower DVFS level
- Graceful performance degradation if power requirement exceeds TDP budget



Advantage of Heterogeneity



Reactive Power Management: Control-theory, Centralized



Muthukaruppan, Thannirmalai Somu, et al. "Hierarchical power management for asymmetric multi-core in dark silicon era." DAC 2013

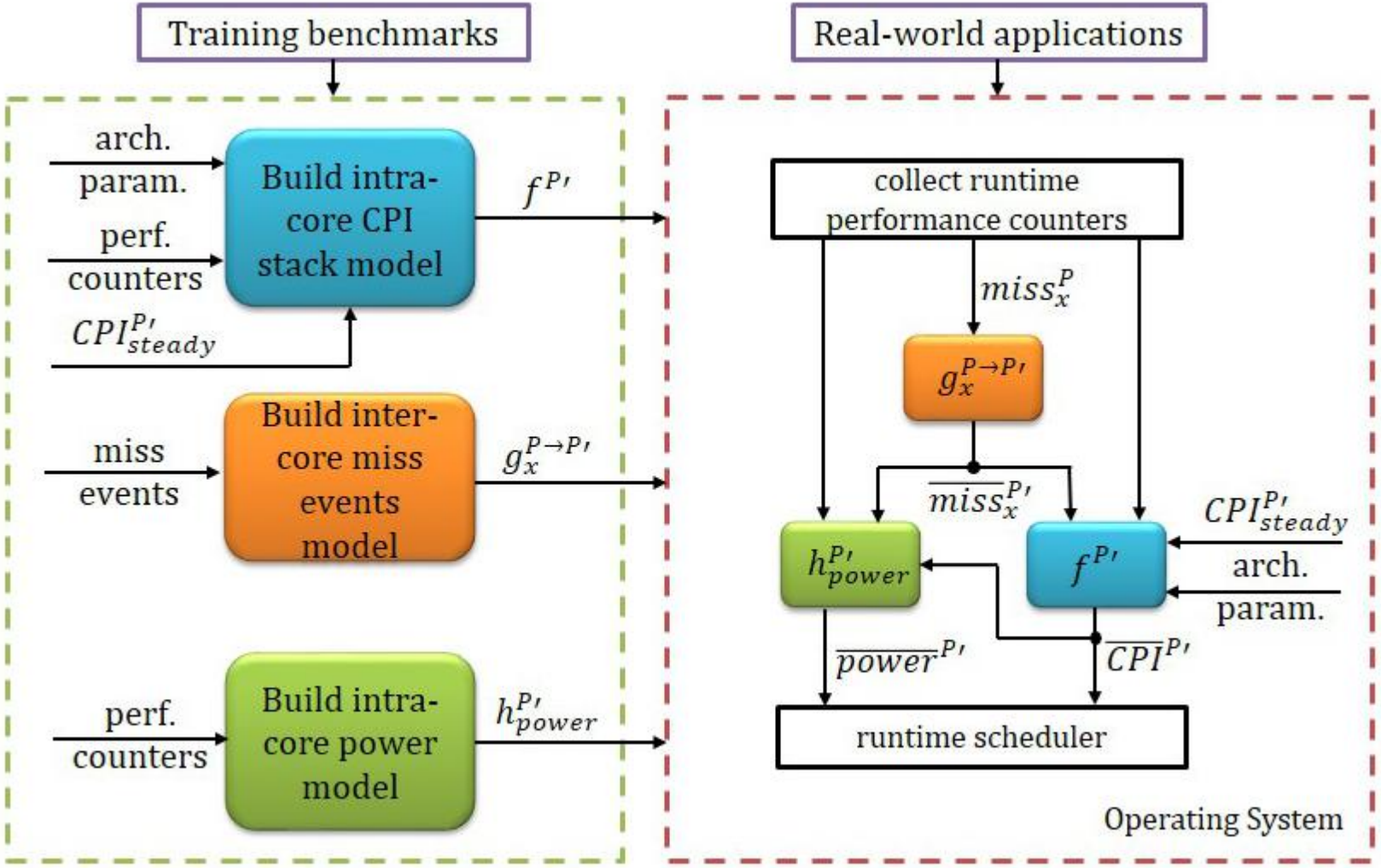
Energy Savings

Benchmark	HPM Scheduler	Linaro
Swap-H264-X264	3.35 W	6.18 W
Swap-H264-Body	3.88 W	6.06 W
H264-Body-Black	4.19 W	6.00 W
Black-X264-H264	4.21 W	6.19 W
X264-Body-H264	2.27 W	5.83 W

Power-Performance Prediction

- Given an execution on small core, predict power, performance behavior on big core or vice versa
- Collect performance counters on core C
- Predict performance counters on core C’
- Predict power, performance on C’

Inter-core Power-Performance Prediction



Pricopi, Mihai, et al. "Power-performance modeling on asymmetric multi-cores." CASES, 2013

Proactive Power Management: Price Theory, Distributed

- Transaction of virtual money in a market place to purchase computational resources
- Allocation of virtual money to tasks, cores control power-performance trade-off
- Tasks demand computational resources to meet Performance (QoS) target
- Cores supply computational resources; Heterogeneity and frequency change supply
- When supply is equal to demand, system reaches equilibrium at minimal energy
- Reduce supply to satisfy TDP constraint at the cost of performance

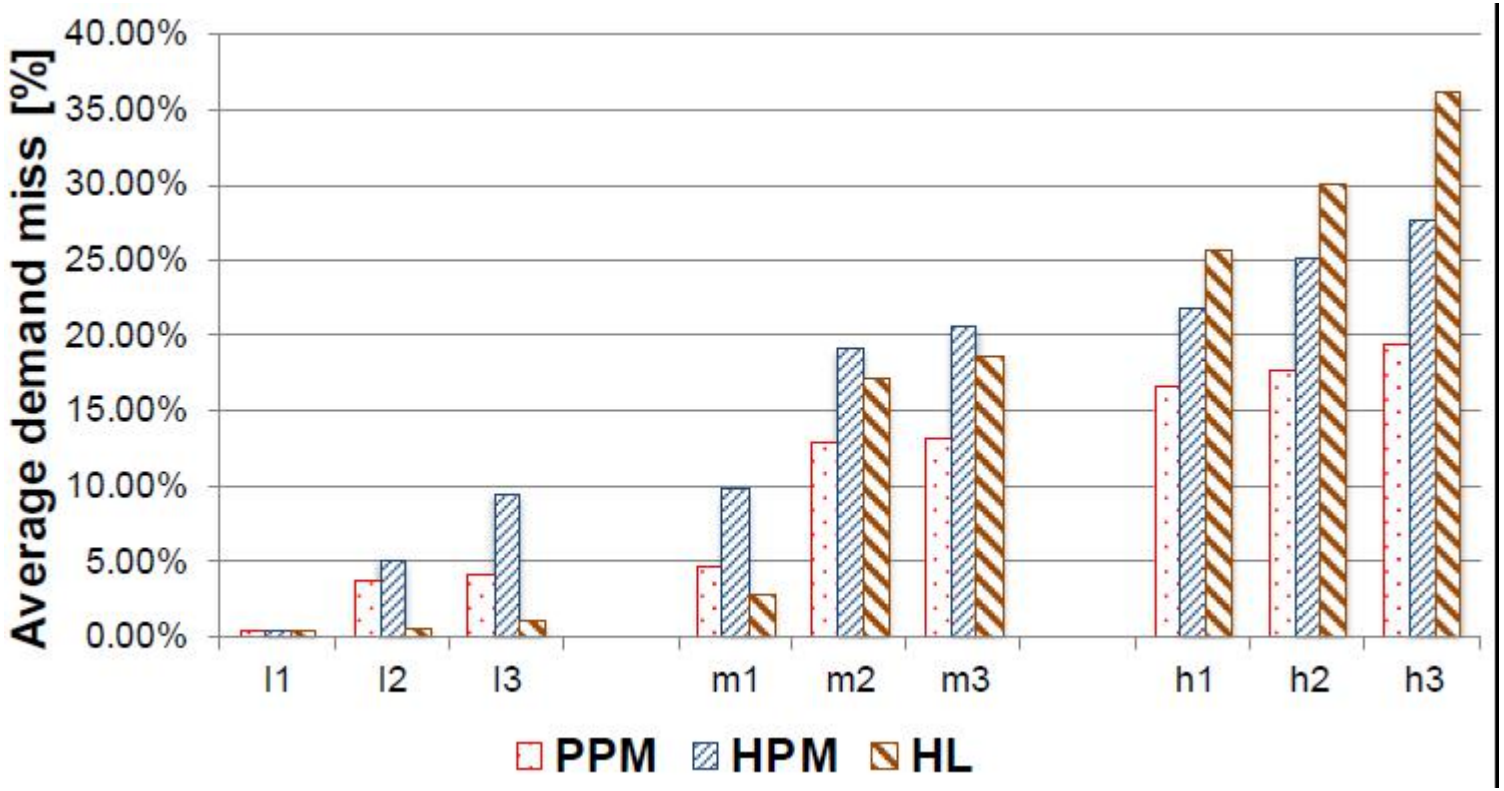
Somu Muthukaruppan, Thannirmalai, Anuj Pathania, and Tulika Mitra. "Price theory based power management for heterogeneous multi-cores." *ASPLOS 2014*

Energy Savings

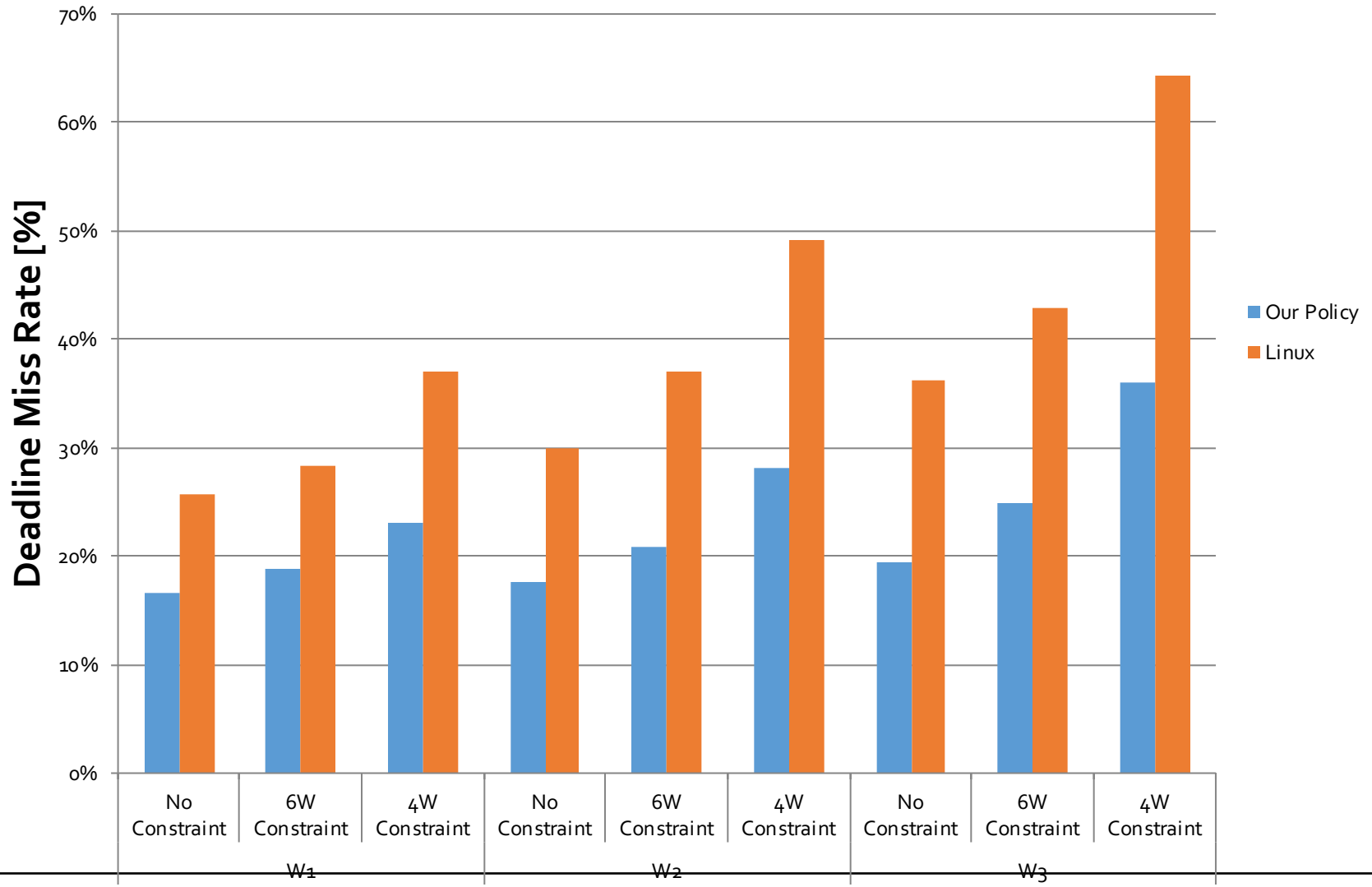
Price Theory (PPM): 2.96 W

Control Theory (HPM): 3.43 W

Linux Governor (HL): 5.99 W



Impact of TDP Constraint



Summary

- Architectural landscape for embedded heterogeneous computing is very rich
  - Heterogeneous CPU, GPU, DSP, reconfigurable computing, non-programmable accelerators

- Software development is really challenging
  - Different programming models and interfaces
  - Choice of the appropriate compute element for an application
  - Collaborative execution across multiple compute elements
  - Power and thermal management