LOWERING POWER CONSUMPTION OF HEVC DECODING

Chi Ching Chi | Techinische Universität Berlin - AES | PEGPUM 2014

Introduction

How to achieve low power HEVC video decoding?

- Modern processors expose many low power techniques
  - Clock gating, power gating, DVFS, asymmetric cores
- Increase application performance
  - SIMD, multithreading
- Two (contradicting) general approaches:
  - Exploit slack
  - Race to idle

What combination is best for video decoding?

Outline

Introduction

Architectural Low Power Modes

Power Optimization HEVC Decoder

Platforms and Methodology

Results

- Offline Decoding
- Real-time Decoding
- Efficiency “Out-of-the-box”

Conclusions

Architectural Techniques

Technische Universität Berlin

Power consumption processor:

$$P_{cpu} = \underbrace{\alpha N K V_{dd} I_{leak}}_{P_{static}} + \underbrace{(1-\alpha) N C f V_{dd}^2}_{P_{dynamic}} \quad (1)$$

Modern architecture incorporate various low power techniques to reduce the **active** and **idle** power

- Active power
  - Dynamic Voltage Frequency Scaling (DVFS)
  - Asymmetric cores
- Idle power
  - Clock gating
  - Power gating

Low Power Active

Technische Universität Berlin

- DVFS lowers $V_{dd}, I_{leak}, f \rightarrow P_{static} \downarrow\downarrow P_{dynamic} \downarrow\downarrow$
- Asymmetric ARM big.Little extend DVFS in lower performance regions
- In Linux `cpufreq` subsystem controls both DVFS and asymmetric core switching
  - Governors change frequency depending on load
  - `exynos_cpufreq` driver maps Little cores to lower frequencies
  - In general responds to load quickly to avoid performance regression

Low Power Idle I

Technische Universität Berlin

- Clock gating and power gating used to implement idle states (C-states)
  - ACPI defines C0 - C3, Intel extends up to C10
  - Intel core C-states

C0	Core is execution code
C1	Core is halted most clocks are stopped
C1E	Voltage reduced to Pn
C3	Core L1/L2 flushed to L3, PLL stopped
C6-C10	Core state saved and power gated

Low Power Idle II

Technische Universität Berlin

- Intel package C-states

C3	Mem self-refresh, some uncore clocks stopped, some voltages reduced
C6	All uncore clocks stopped
C7	L3 power gated, more voltages reduced
C8	Most uncore power gated
C9-C10	FIVR in low power/off state

- In Linux `cpuidle` subsystem controls C-states
  - Trade off power saving in lower C-state vs. transition cost
  - Predicts duration of next idle period to select C-state
  - C-state, transition time, transition energy model specific
  - Vendor supplies this in model/series specific drivers

Outline

Introduction

Architectural Low Power Modes

Power Optimization HEVC Decoder

Platforms and Methodology

Results

- Offline Decoding
- Real-time Decoding
- Efficiency “Out-of-the-box”

Conclusions

Power Optimization HEVC Decoder

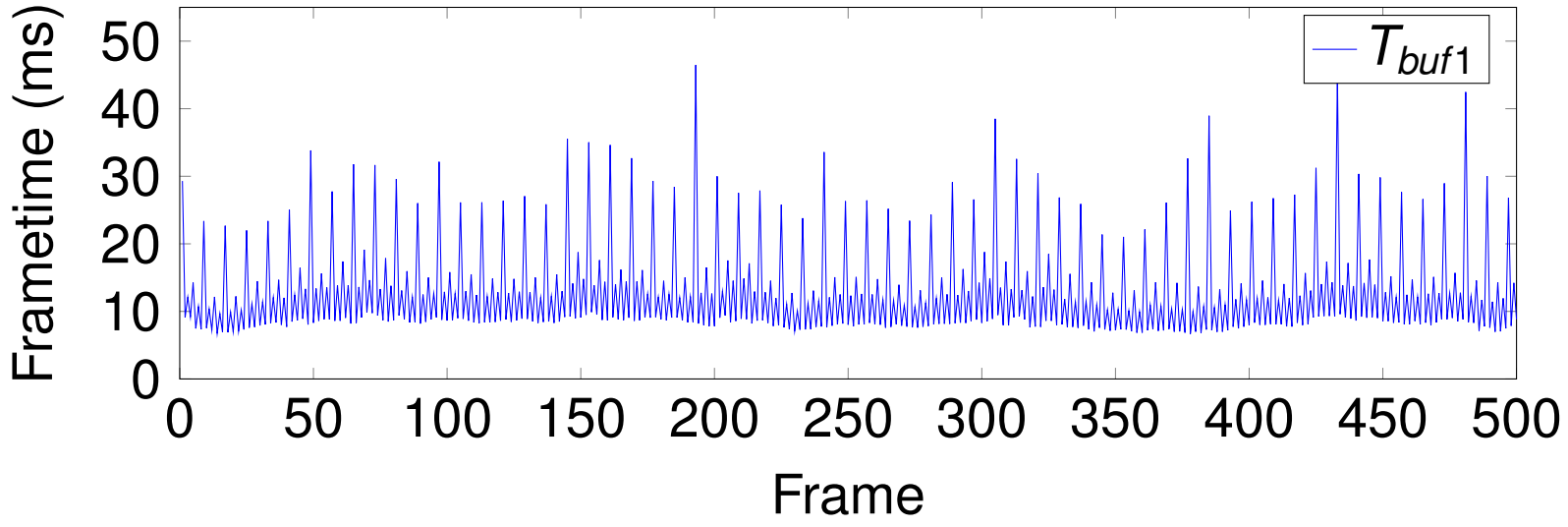
Two usage models HEVC decoding : *offline* and *real-time*

- *Offline* decoding
  - Improve application performance
    - General optimization
    - ISA/Architecture specific (e.g. SIMD)
    - Multithreading
- Additionally for *real-time* decoding
  - Decrease worst-case complexity
  - Coalesce computation

Real-time Video Decoding I

Technische Universität Berlin

Decrease worst-case complexity using playback buffers

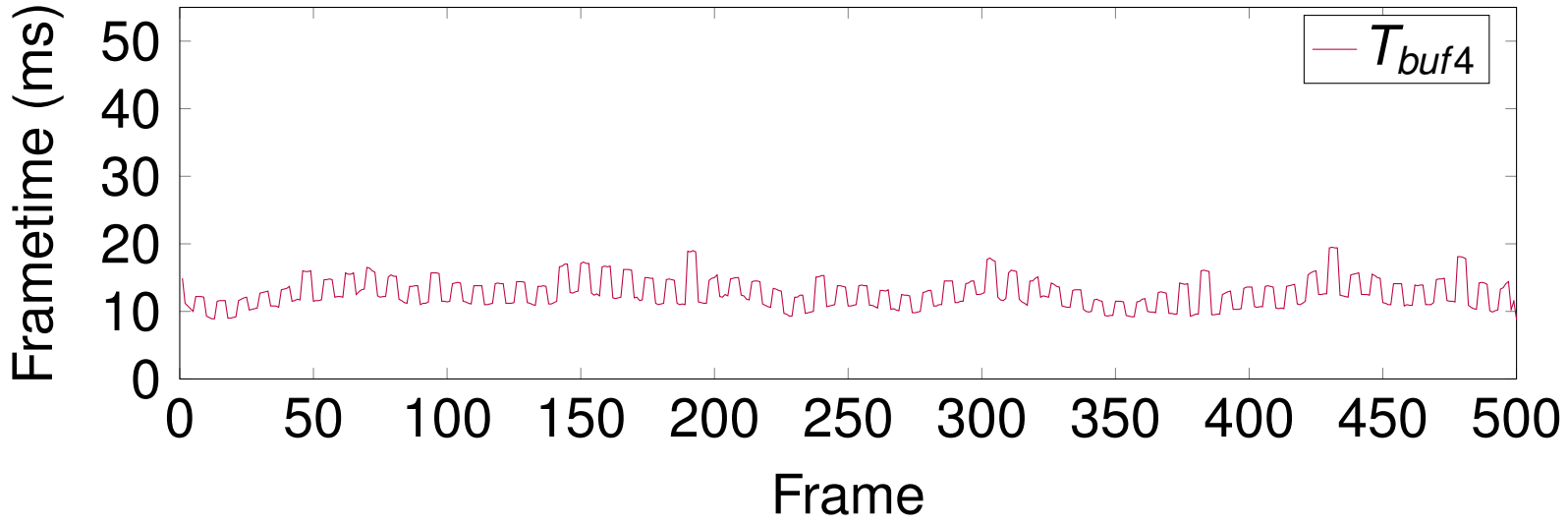


- Lower peaks allow for execution on lower frequency/slower cores
- Improves possibilities for exploiting slack

Real-time Video Decoding I

Technische Universität Berlin

Decrease worst-case complexity using playback buffers

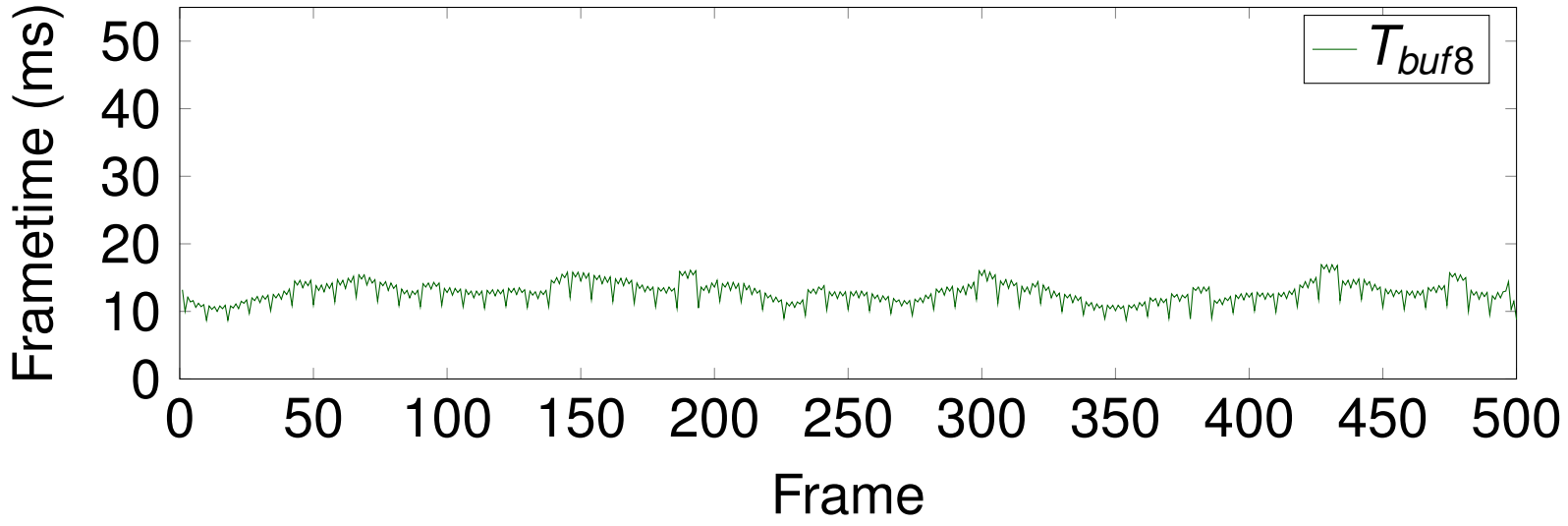


- Lower peaks allow for execution on lower frequency/slower cores
- Improves possibilities for exploiting slack

Real-time Video Decoding I

Technische Universität Berlin

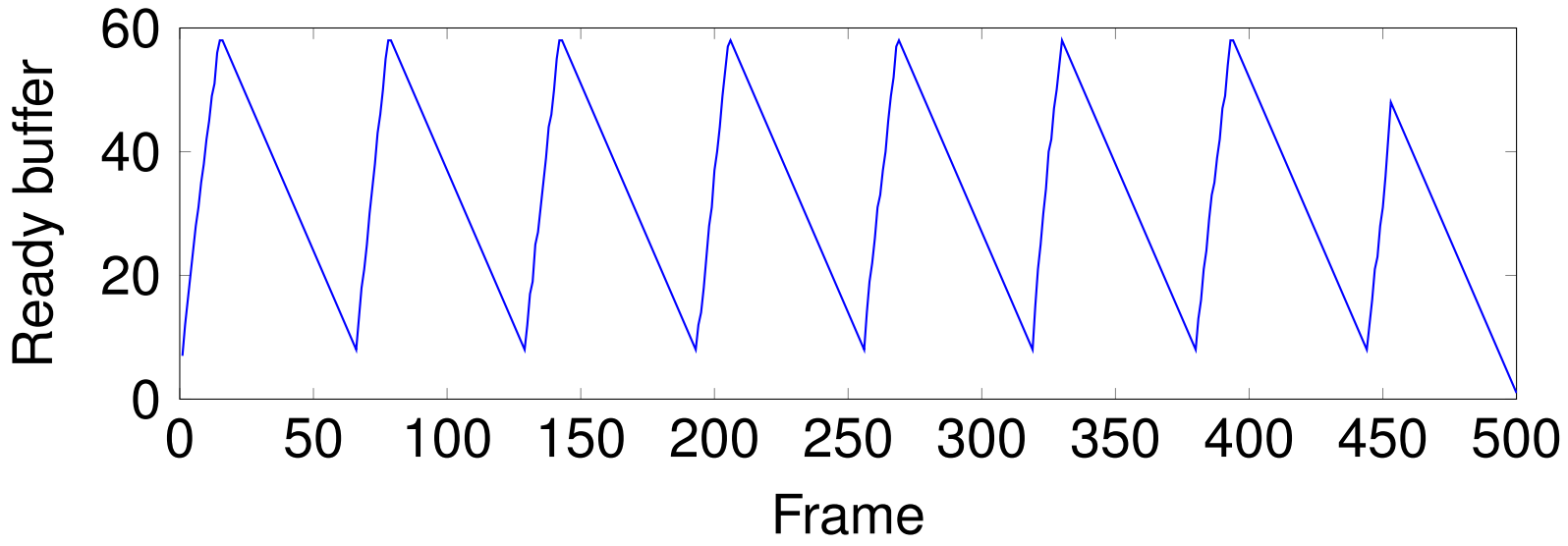
Decrease worst-case complexity using playback buffers



- Lower peaks allow for execution on lower frequency/slower cores
- Improves possibilities for exploiting slack

Real-time Video Decoding II

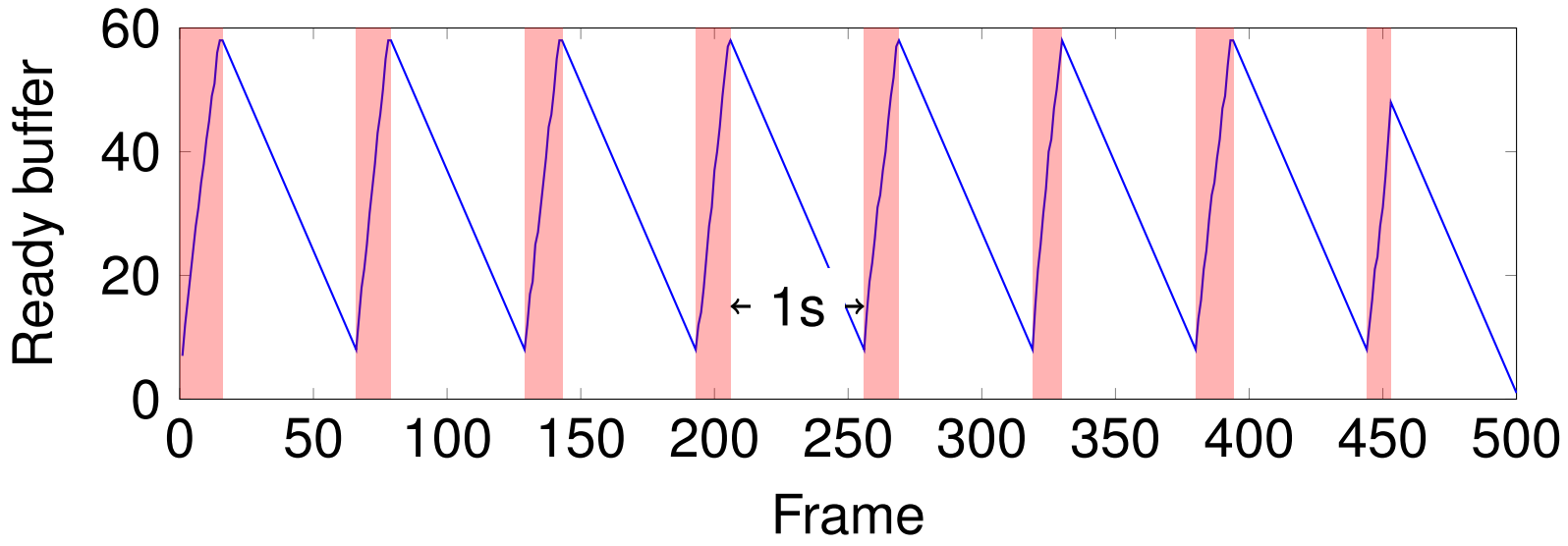
Improve C-state residency with burst decoding



- Perform decoding bursts to improve deeper C-state residency
- Improves effectiveness of race to idle in theory
- In practice has no significant benefit except rare configurations

Real-time Video Decoding II

Improve C-state residency with burst decoding



- Perform decoding bursts to improve deeper C-state residency
- Improves effectiveness of race to idle in theory
- In practice has no significant benefit except rare configurations

Outline

Introduction

Architectural Low Power Modes

Power Optimization HEVC Decoder

Platforms and Methodology

Results

- Offline Decoding
- Real-time Decoding
- Efficiency “Out-of-the-box”

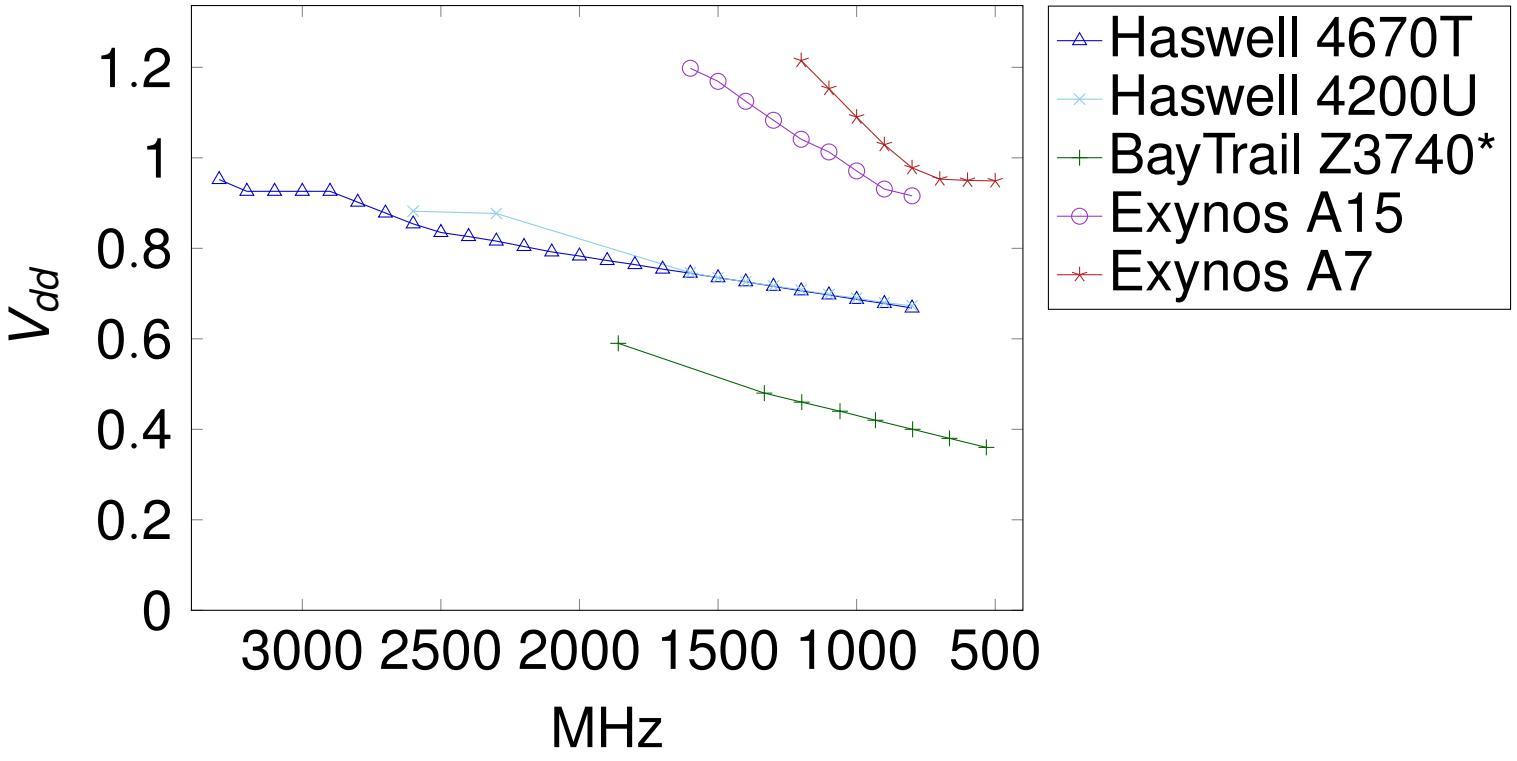
Conclusions

Platforms I

- Used 4 platform representative of modern PC and consumer electronics processors
- Power measurable but not completely comparable

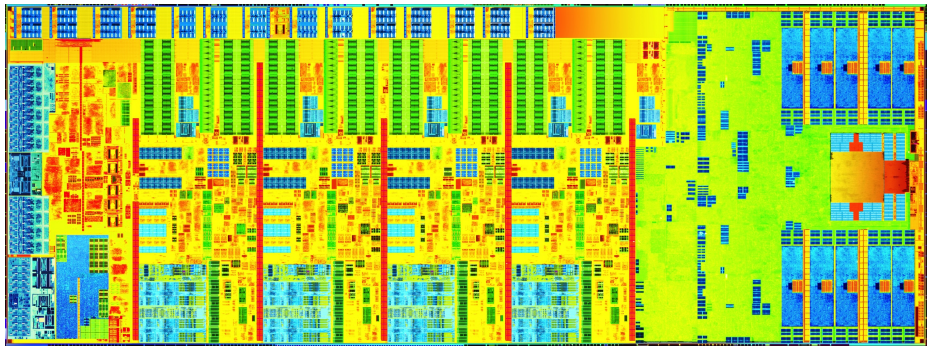
Series	Model	Process	Die size	Measurable power		
				Cores	Uncore	DRAM
Haswell	i5-4670T	22nm	177mm^2	✓	✓	✓
Haswell ULT	i5-4200U	22nm	134mm^2	✓	✓	✓
Baytrail-T	Z3740	22nm	102mm^2	✓	✓	✗
Exynos	5410	28nm	122mm^2	✓	(GPU)	✓

Platforms II

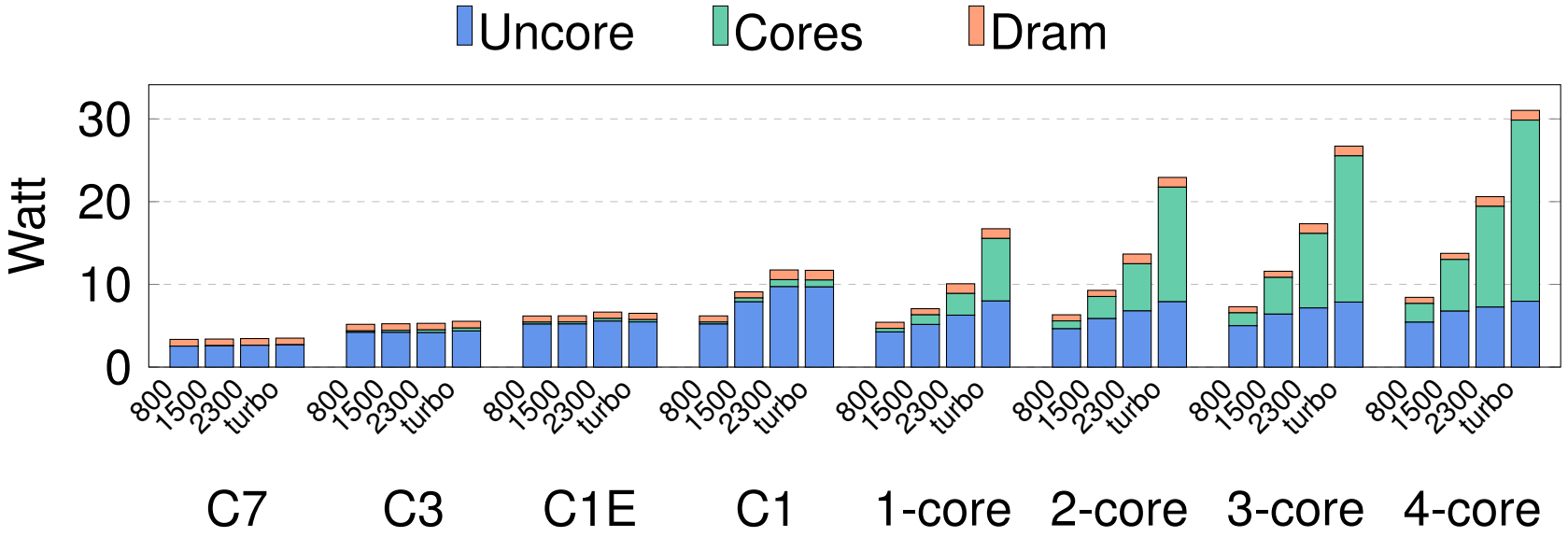


Haswell i5-4670T

- Haswell
- 4 cores
- TDP 45 W
- Low power desktop/ all-in-one

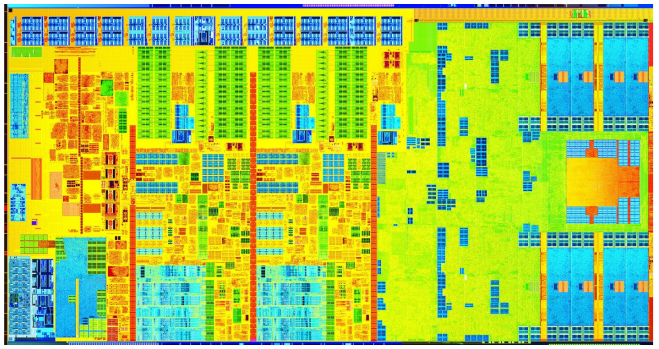


Source: Intel

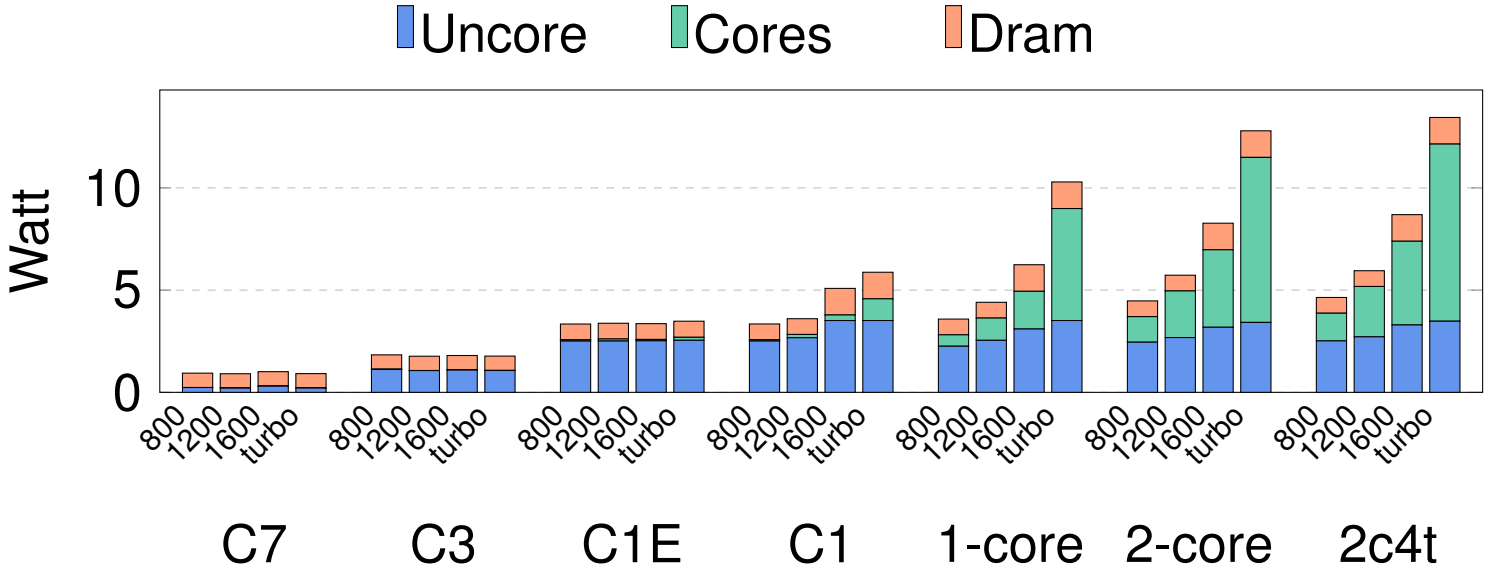


Haswell ULT i5-4200U

- Haswell
- 2 cores (SMT)
- TDP 15 W
- Ulltrabook/convertibles

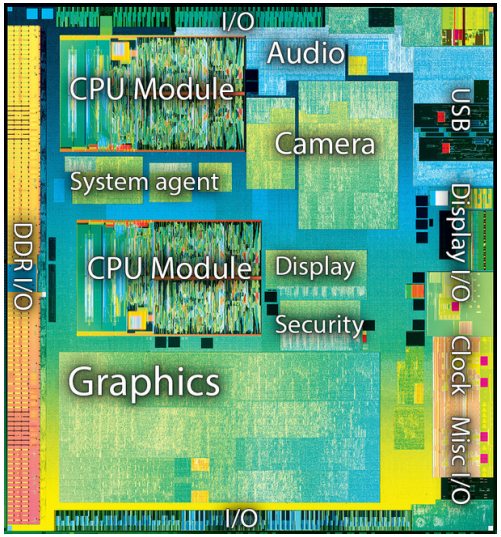


Source: Intel

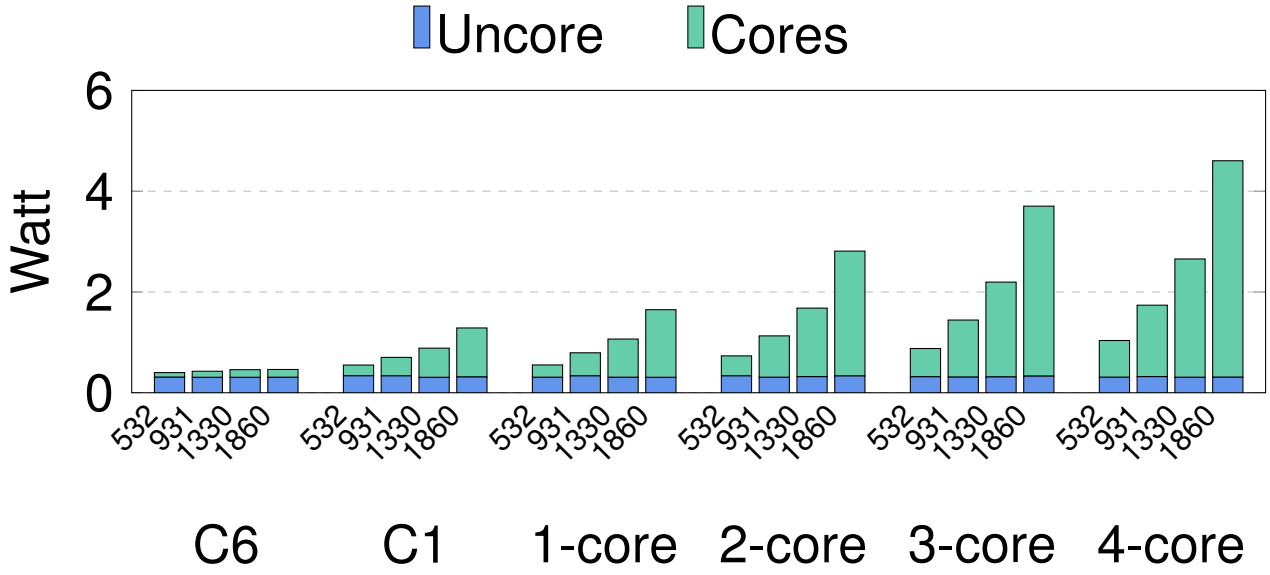


Baytrail-T Z3740

- Silvermont
- 4 cores
- SDP 2 W/ TDP 5 W?
- Tablets

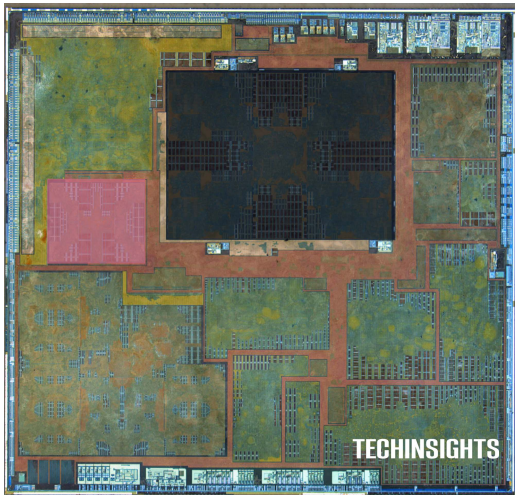


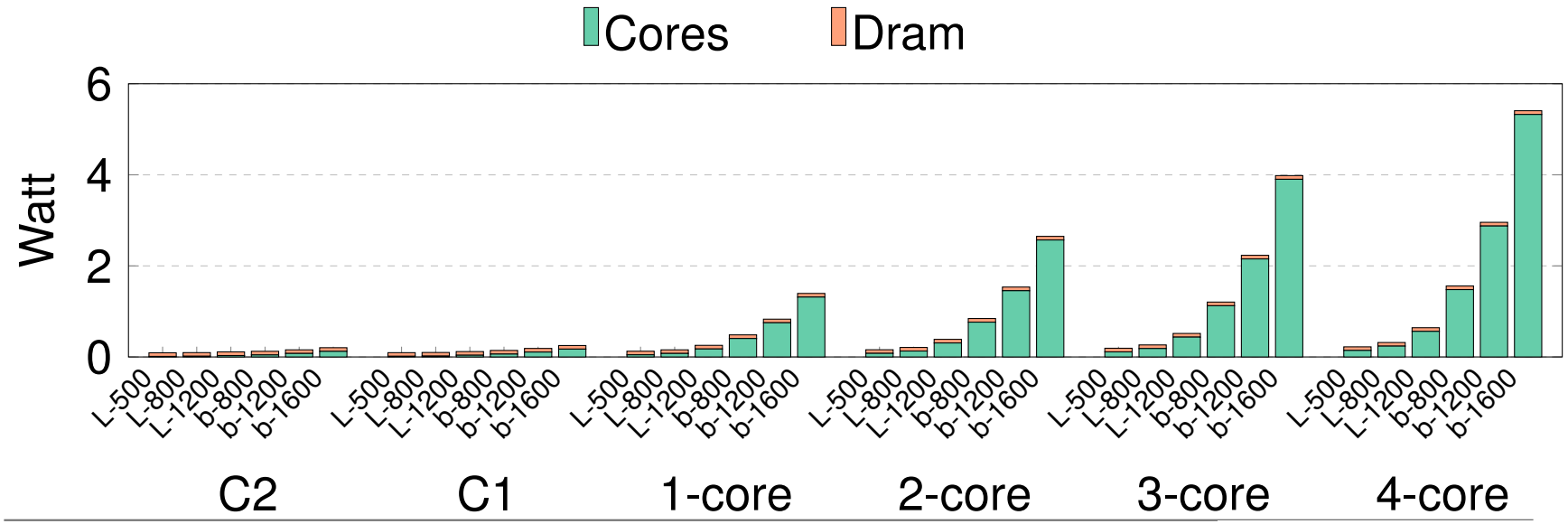
Source: Intel, tweakers.net



Exynos 5410

- A15 + A7 big.Little
- 4 + 4 cores
- TDP 6 W?
- Smartphones/Tablets





Methodology

- Use *highly optimized* HEVC Decoder developed by AES TUB
- All experiments executed under Linux (3.12+ for x86, 3.4 arm)
- GCC 4.8.1
- In total 100 FHD and UHD short 10 sec test sequences
  - 1 second intra period, GOP size 8
  - FHD 24, 50, 60 fps with 40 to 200 kb/frame
  - UHD 50 fps with 100 to 500 kb/frame
- Multithreading using wavefronts (WPP)
- Real-time execution simulated with 8 output picture buffer

Outline

Introduction

Architectural Low Power Modes

Power Optimization HEVC Decoder

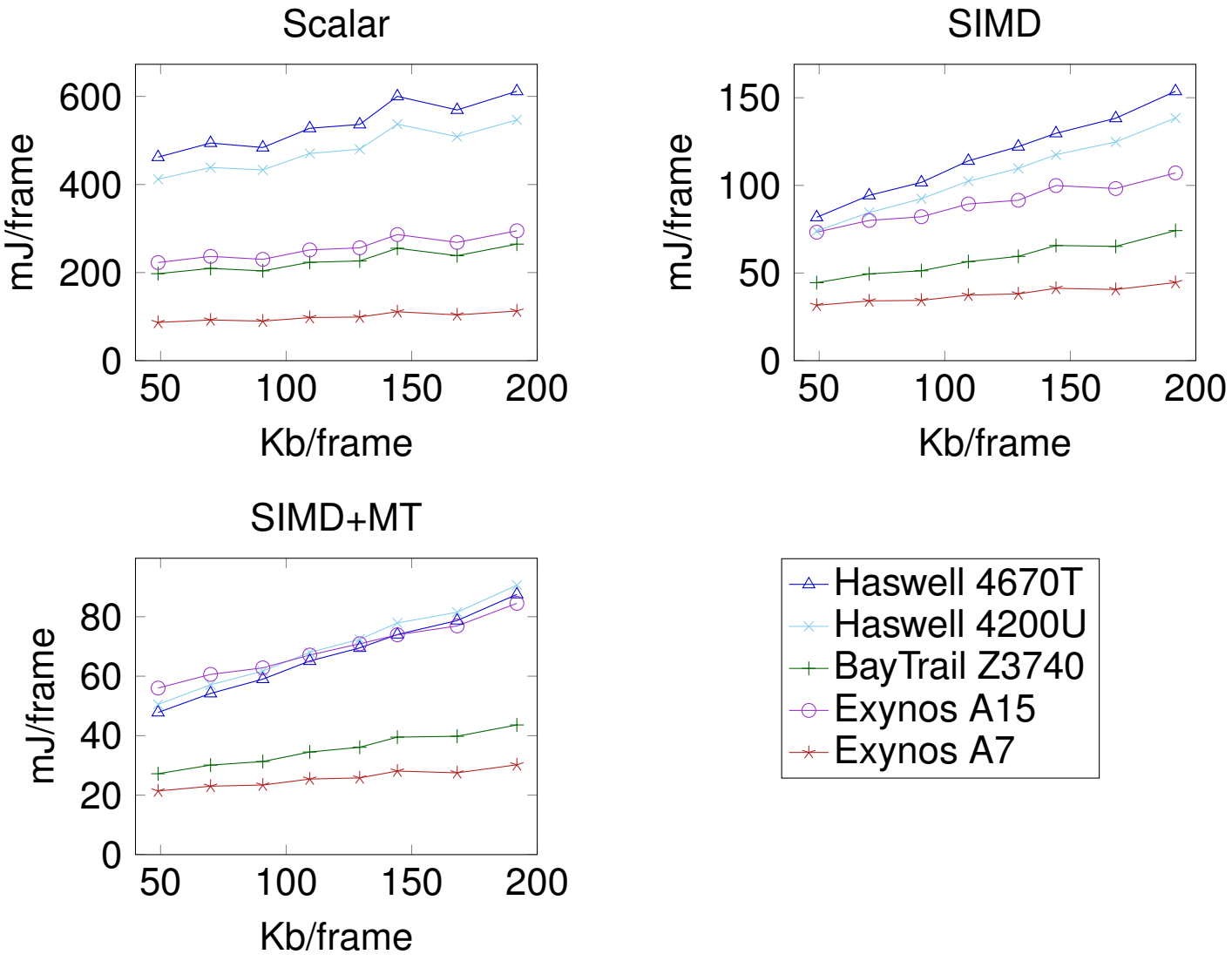
Platforms and Methodology

Results

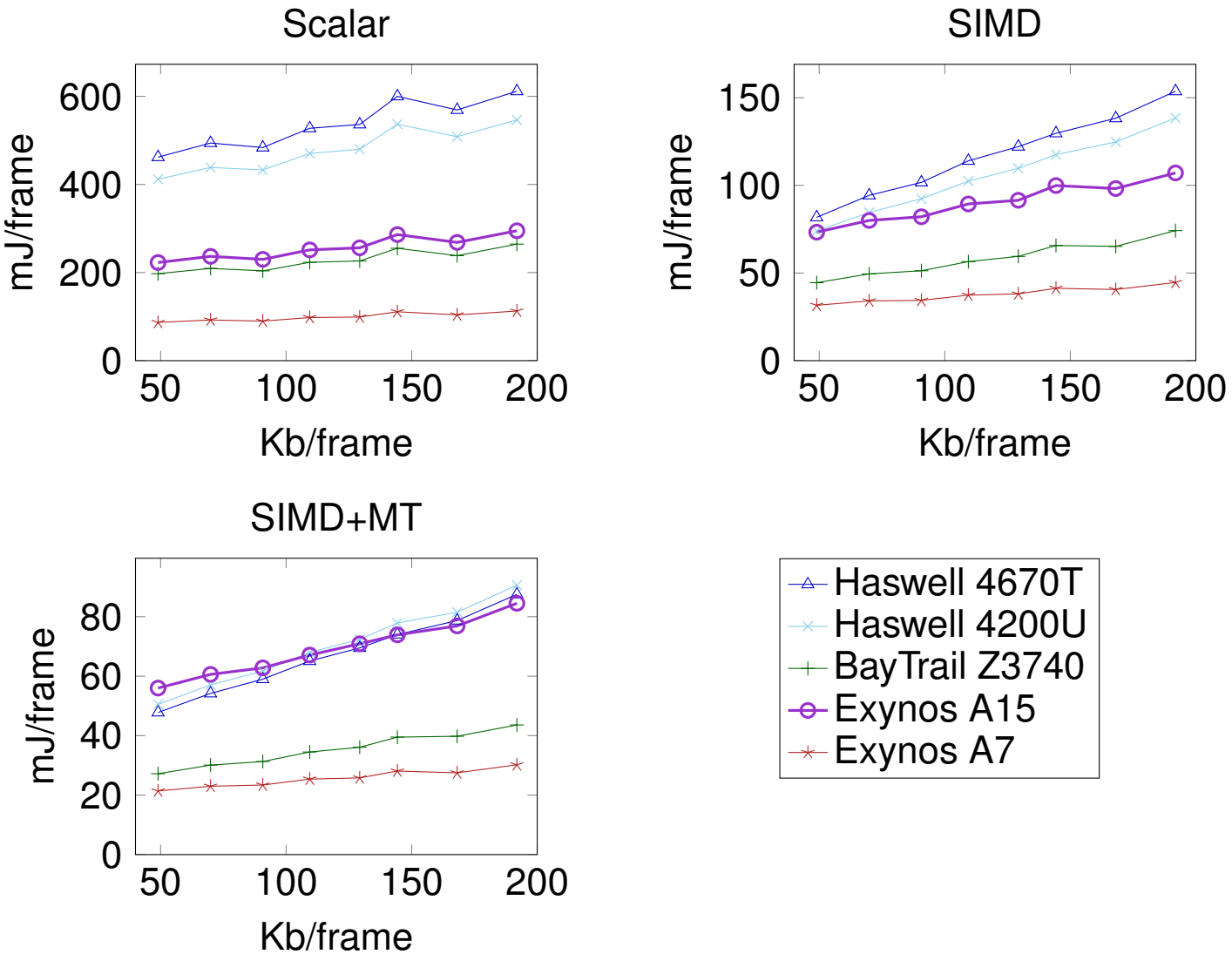
- Offline Decoding
- Real-time Decoding
- Efficiency “Out-of-the-box”

Conclusions

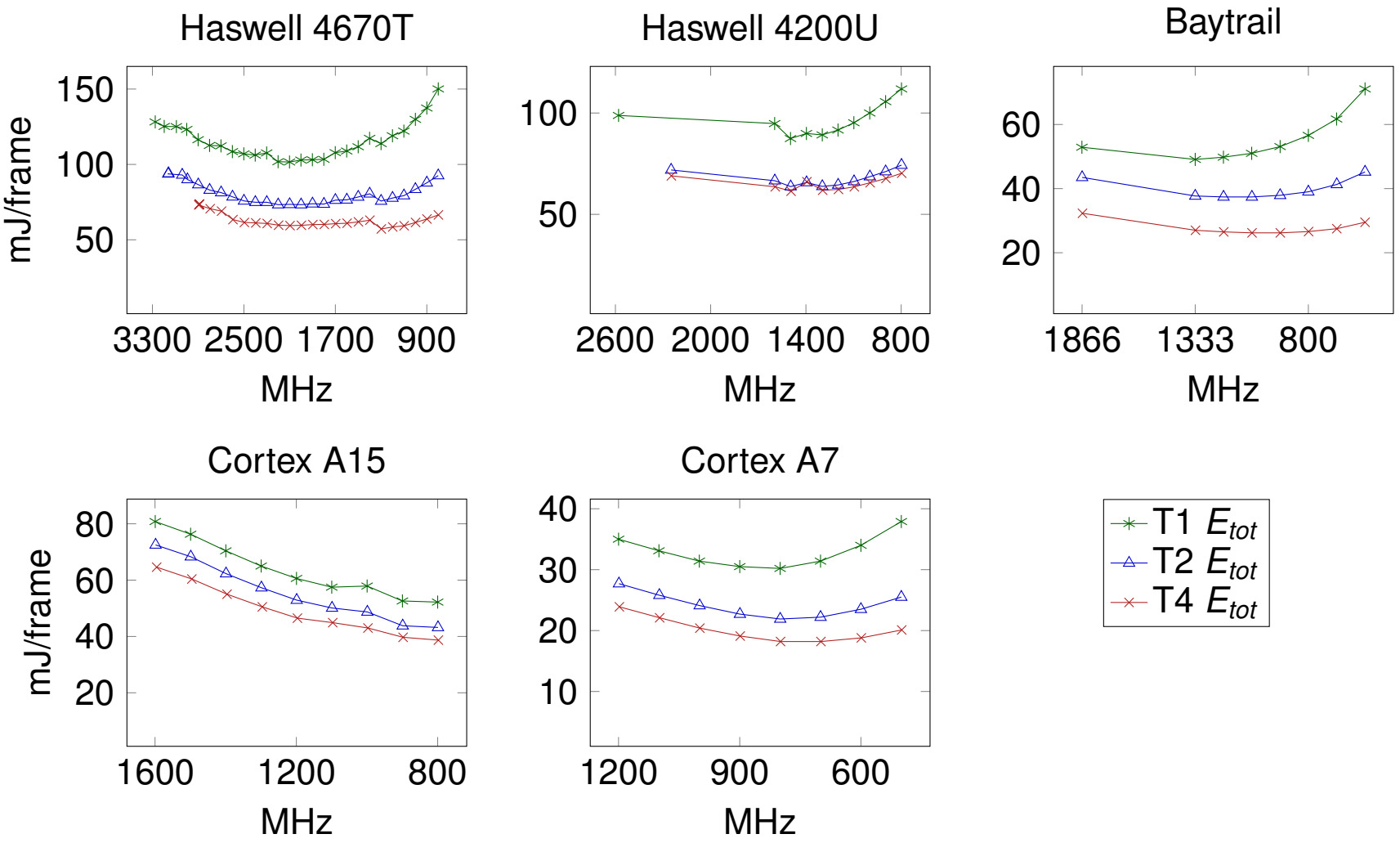
SIMD and Multithreading (1080p)



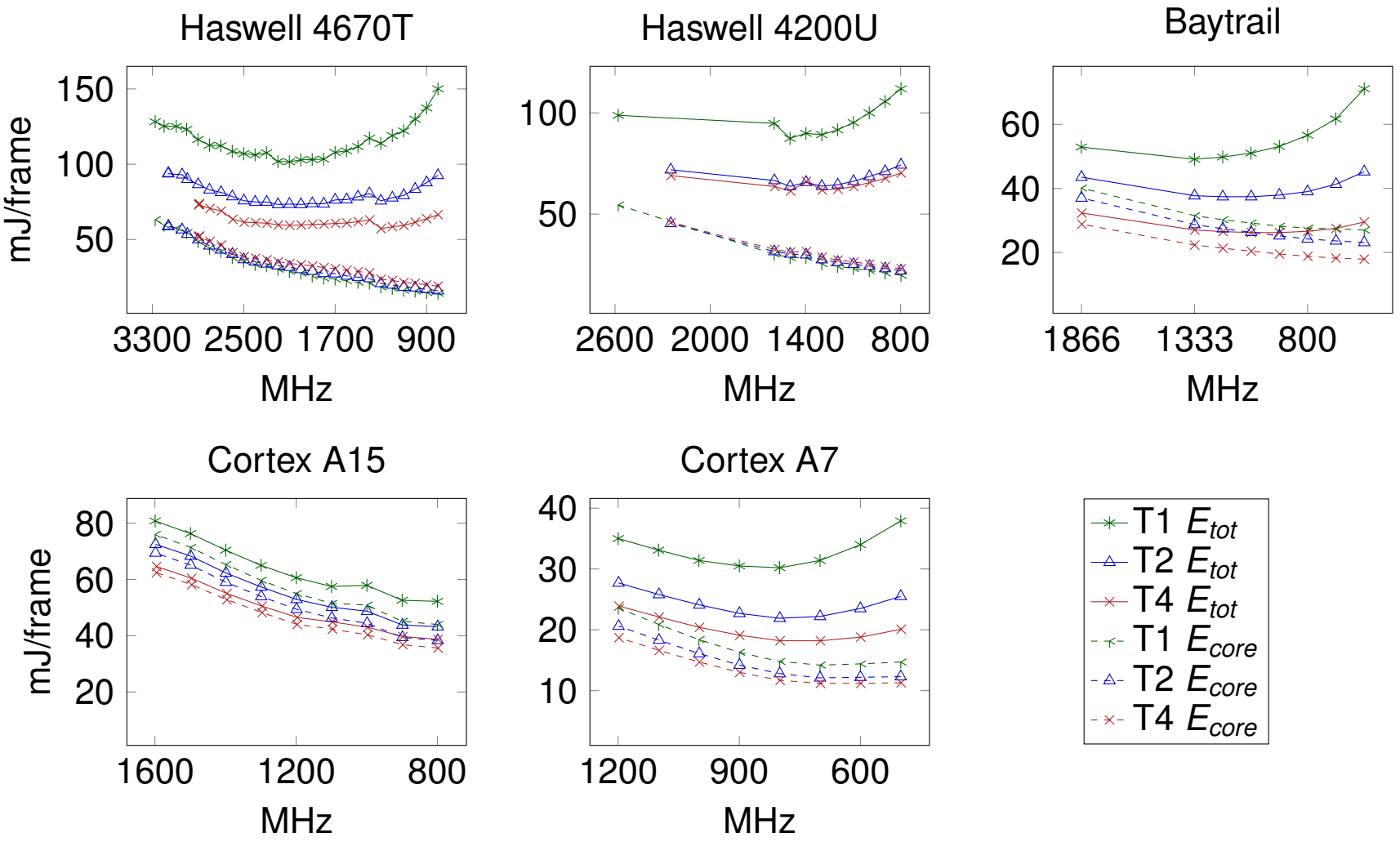
SIMD and Multithreading (1080p)



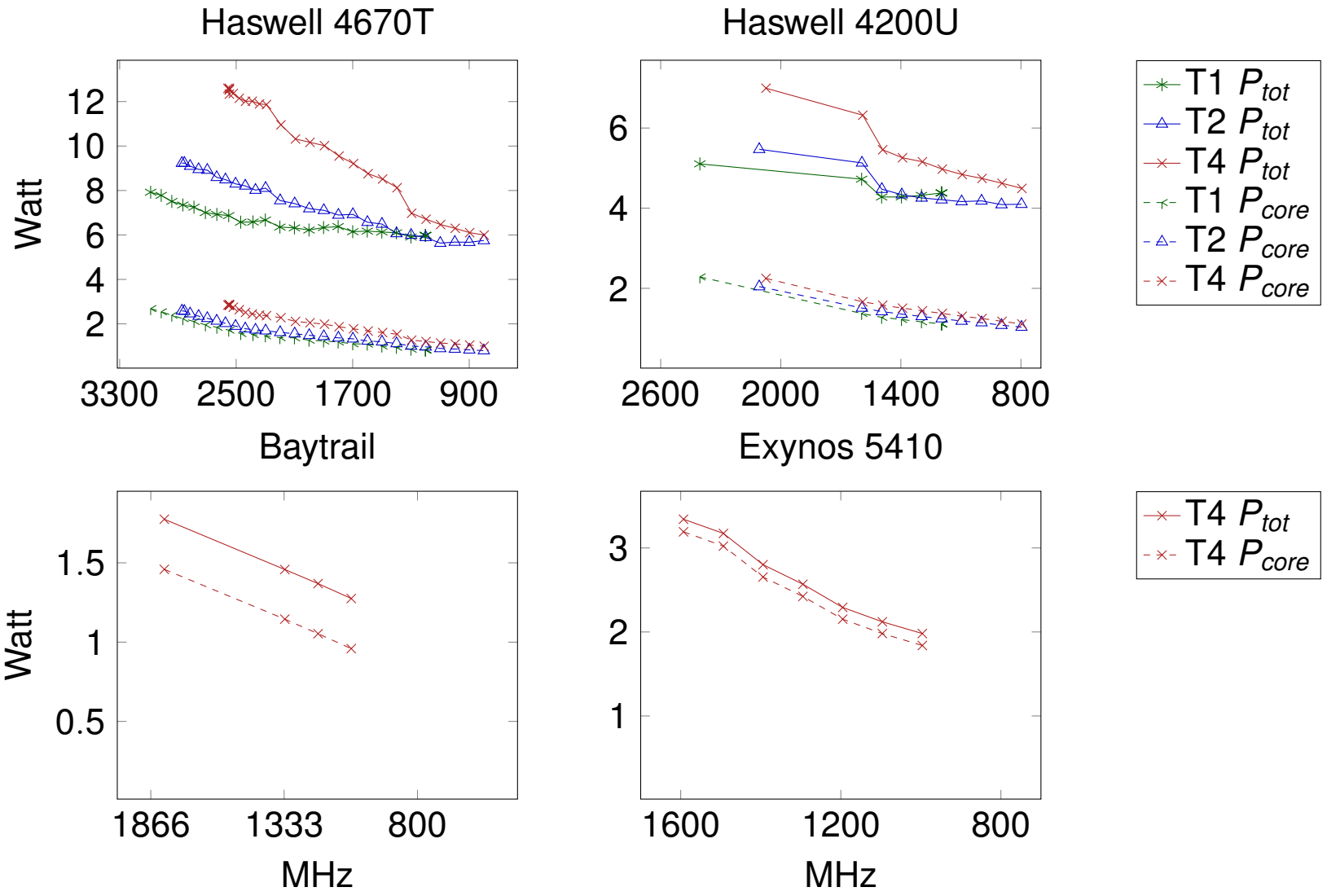
DVFS in Offline Decoding (1080p)



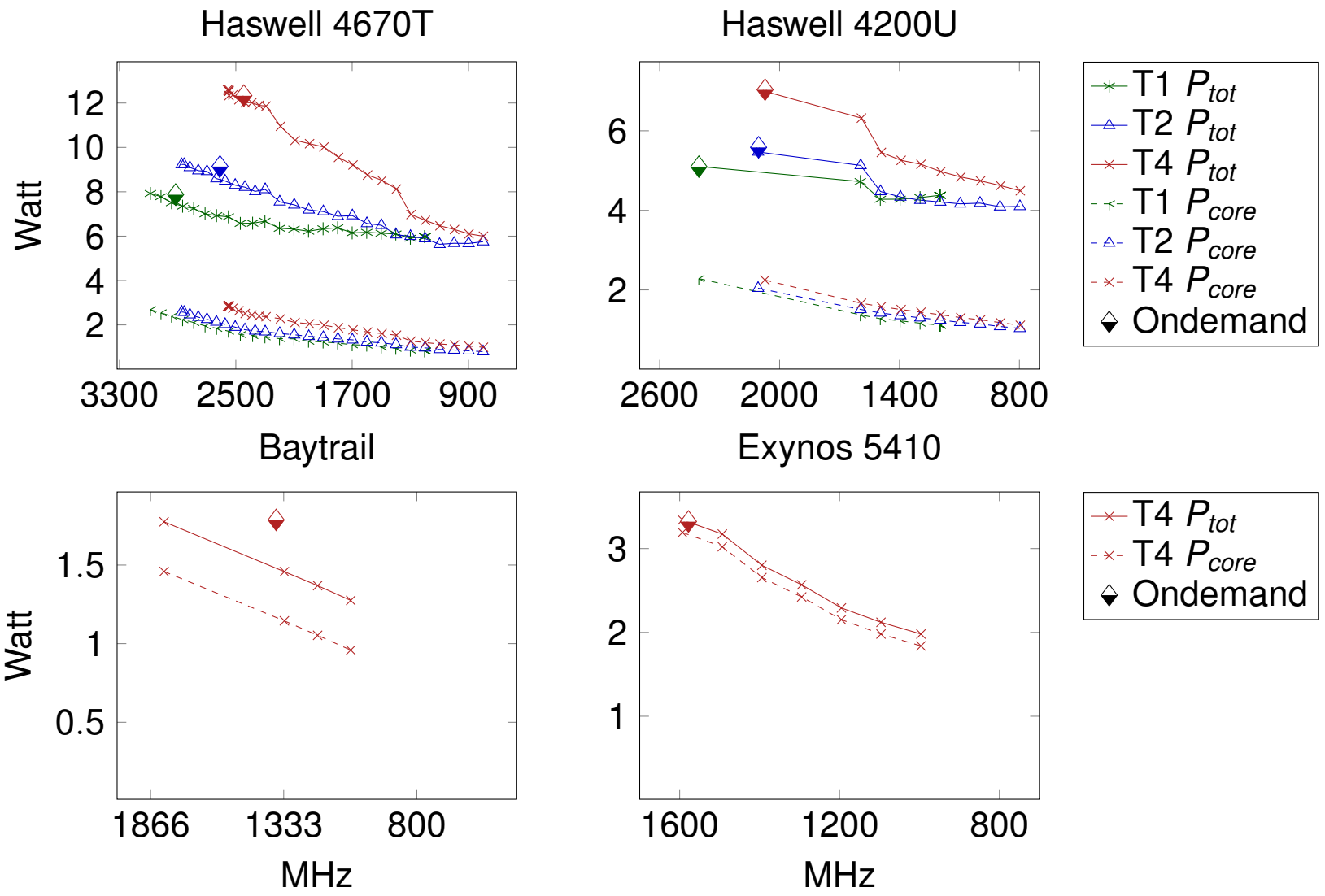
DVFS in Offline Decoding (1080p)



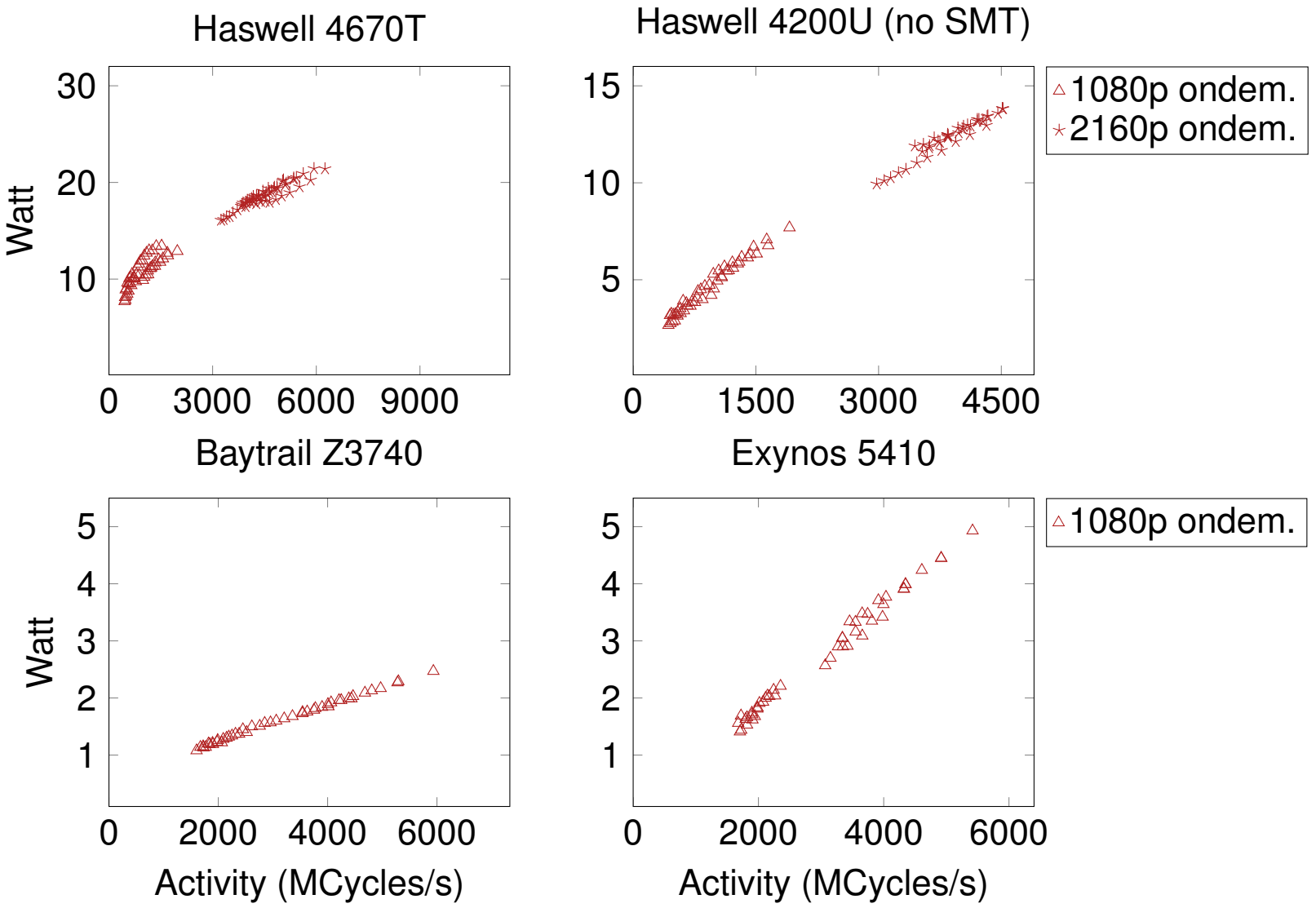
DVFS in Real-time Decoding (1080p50 4.5Mbps)



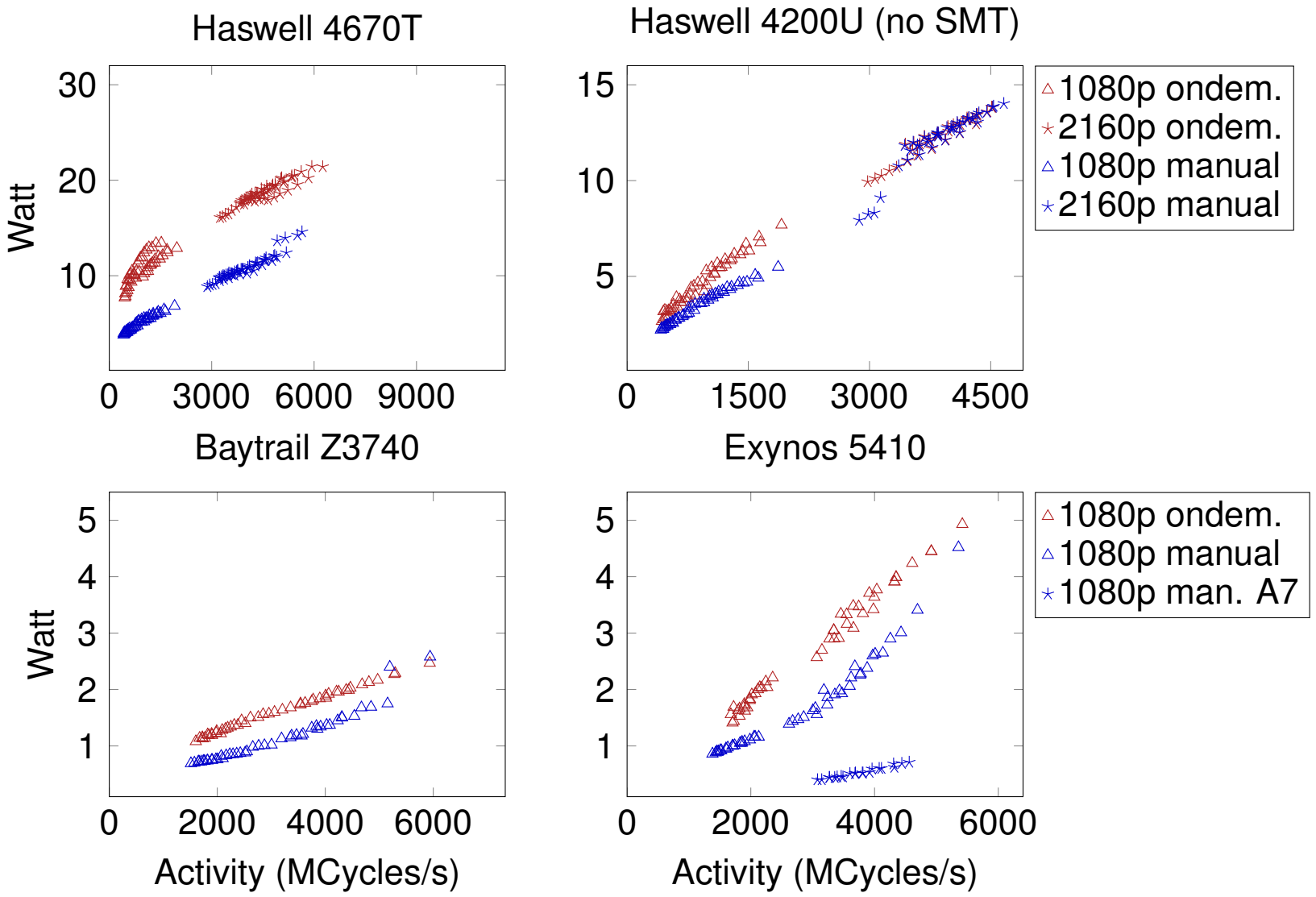
DVFS in Real-time Decoding (1080p50 4.5Mbps)



Efficiency “Out-of-the-box”



Efficiency “Out-of-the-box”



Outline

Introduction

Architectural Low Power Modes

Power Optimization HEVC Decoder

Platforms and Methodology

Results

- Offline Decoding
- Real-time Decoding
- Efficiency “Out-of-the-box”

Conclusions

Conclusions I

How to achieve low power HEVC decoding:

- Offline decoding
  - Application optimization (SIMD+MT)
  - DVFS only when cores are consuming most of energy
- Real-time decoding
  - Exploiting slack preferred over race to idle
  - Multithreading only helpful when lowering clock
  - $<$ 2W for 1080p50 in software

Limitations low power modes:

- OS not selecting lower frequencies
  - Not able to determine when exploiting slack is desired

Conclusions II

- Around 30%! power savings possible
- Architectures not efficient at low activity
  - DVFS targets mainly cores, but cores consume little power at low frequency
  - big.Little solves reduced effectiveness DVFS closer to V_{th}, but not main problem (for Intel)
  - Deeper idle states + race to idle not a replacement

Future Work

- Include rendering power in analysis
- Extend towards more OS (Windows, Android, IOS, etc..)
- Investigate towards application driven DFVS control