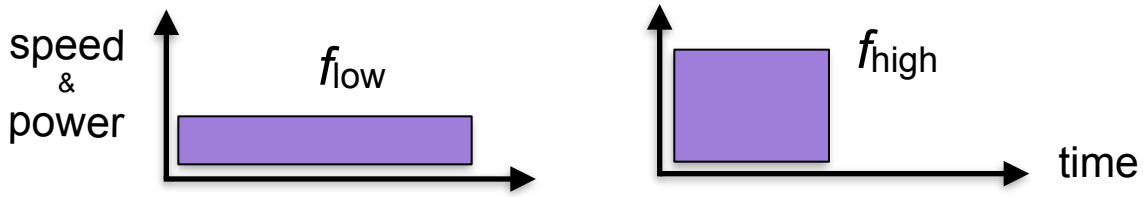
Unifying DVFS and Offlining in Mobile Multicores

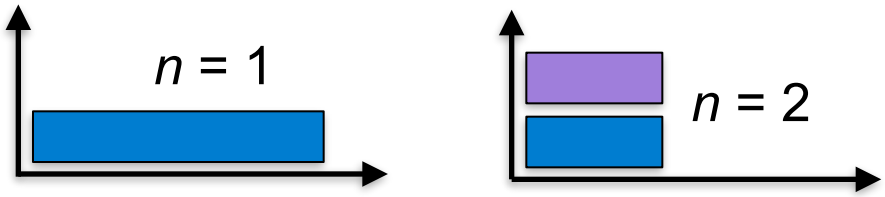
Aaron Carroll
Gernot Heiser

Power control knobs

DVFS:



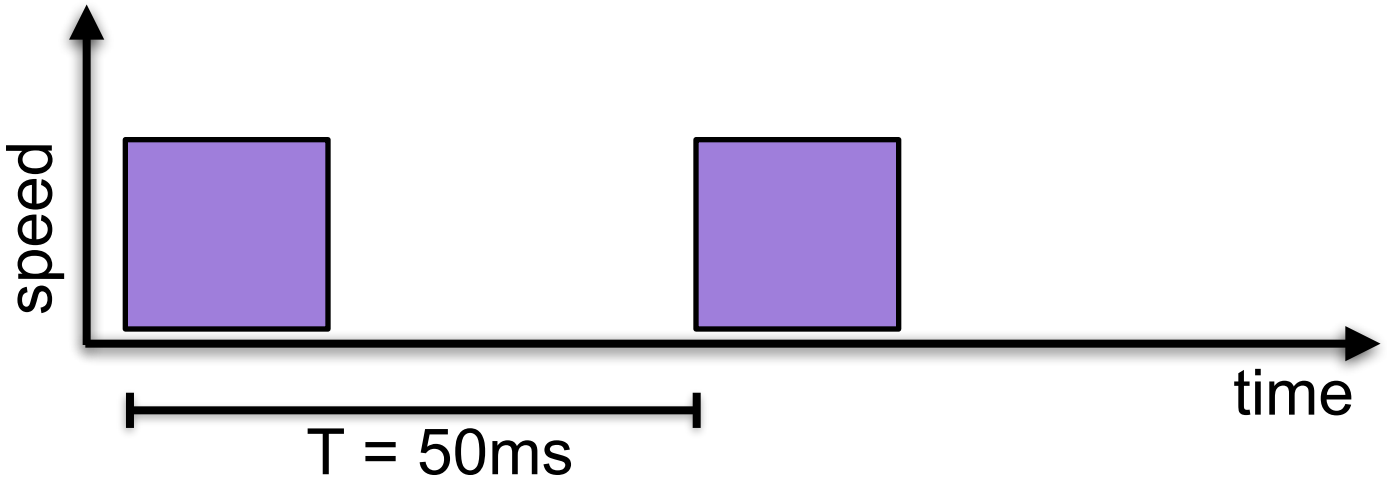
#online cores:



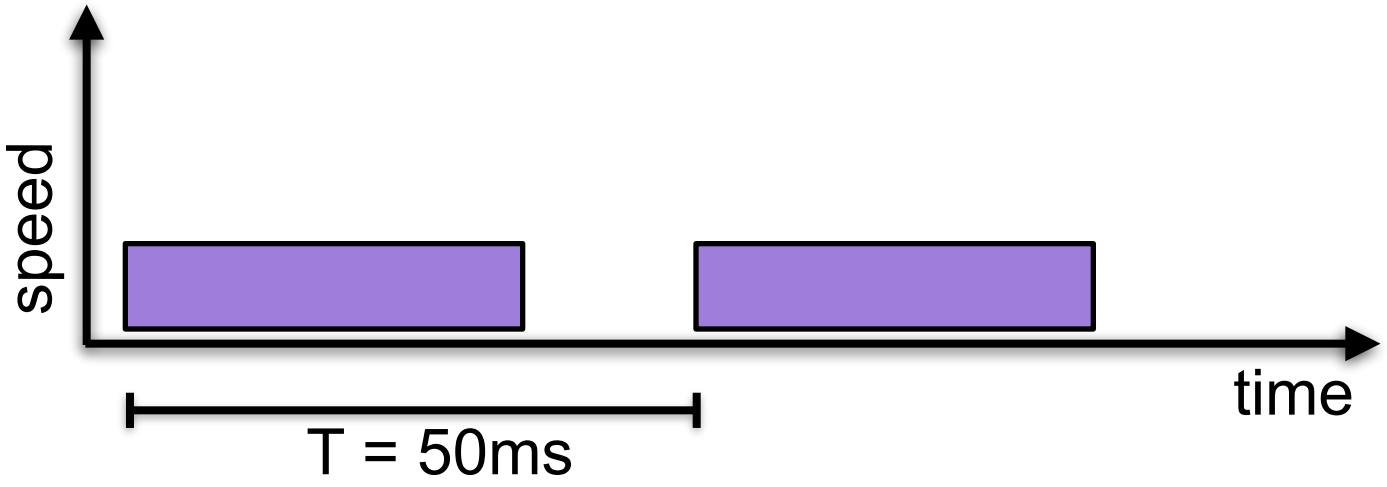
Problem

- trade-off between the two mechanisms
- optimal **operating point**
  - frequency and #online cores
- **idle** energy management
  - under-utilised CPU
  - reduce energy
  - performance unaffected (ideally…)

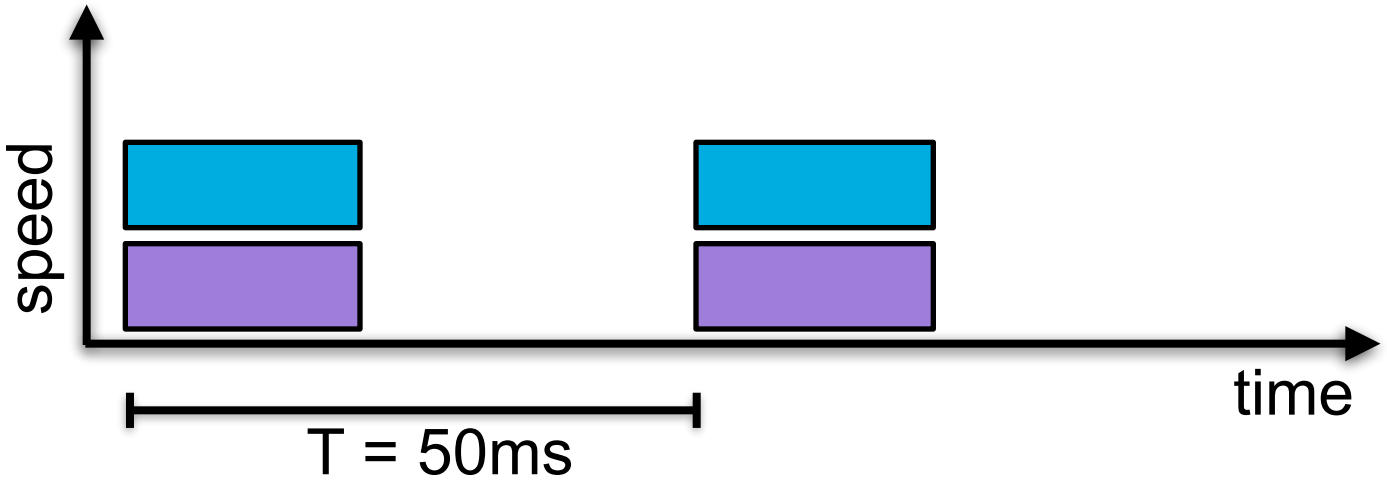
loadcpu



loadcpu

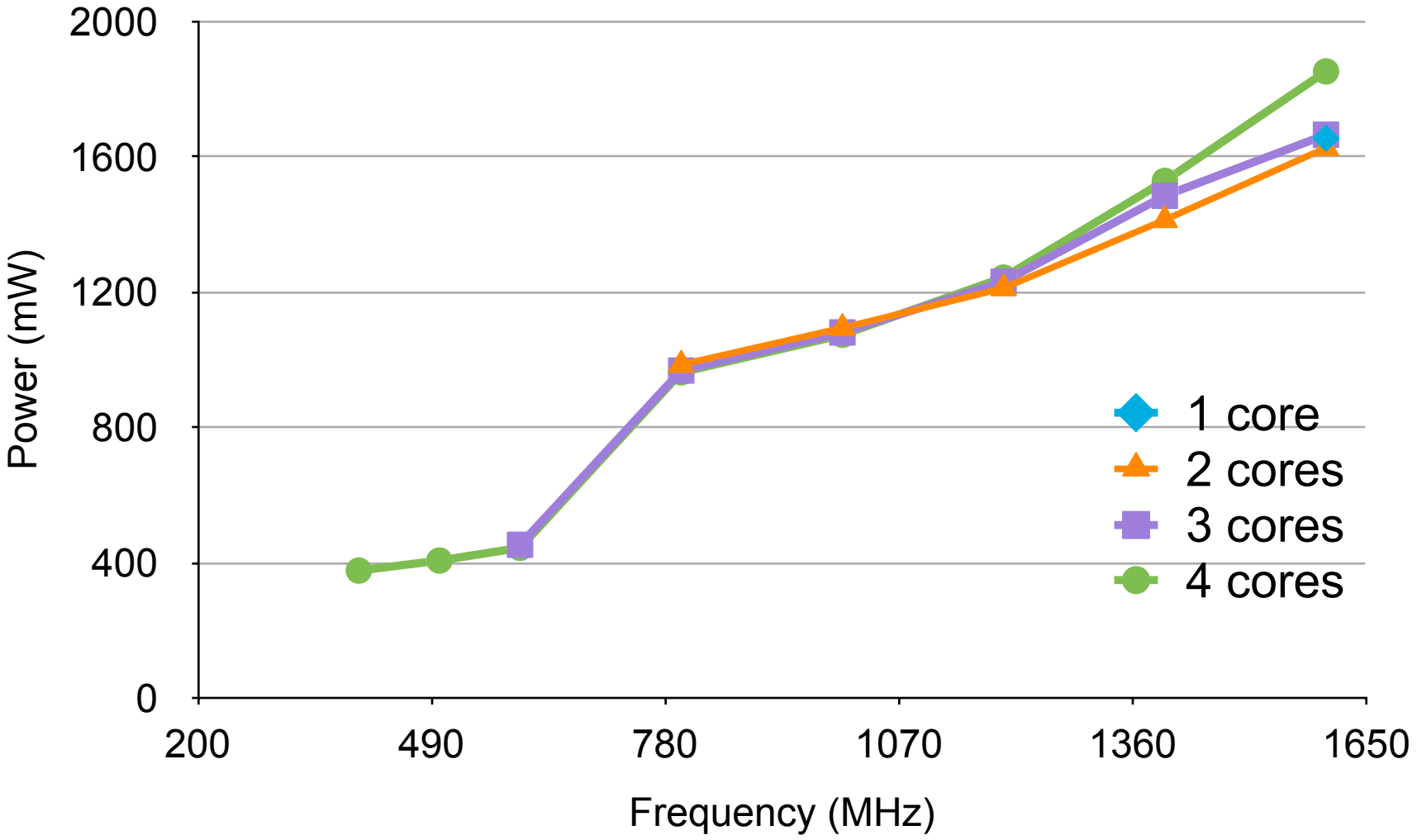


loadcpu

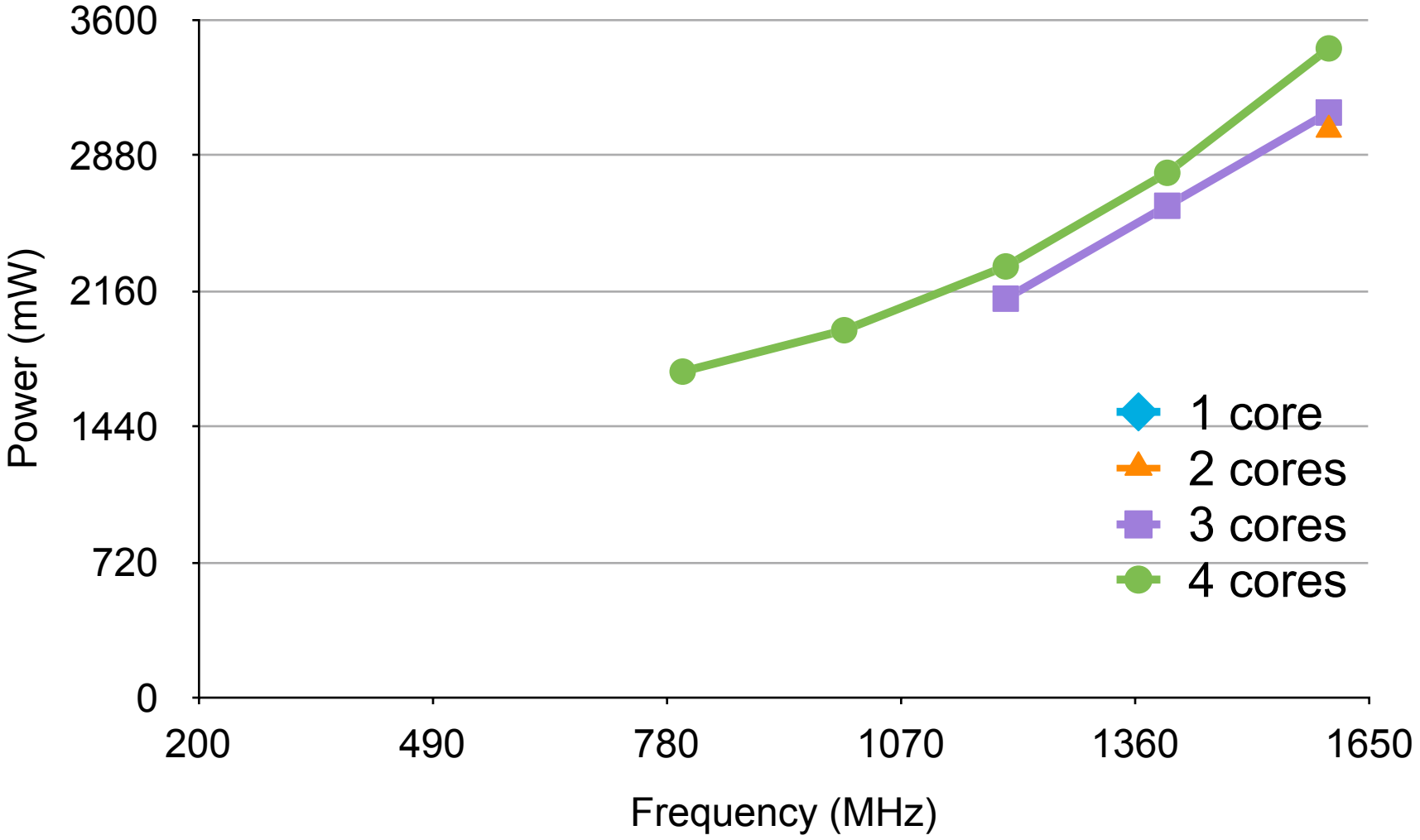


Devices

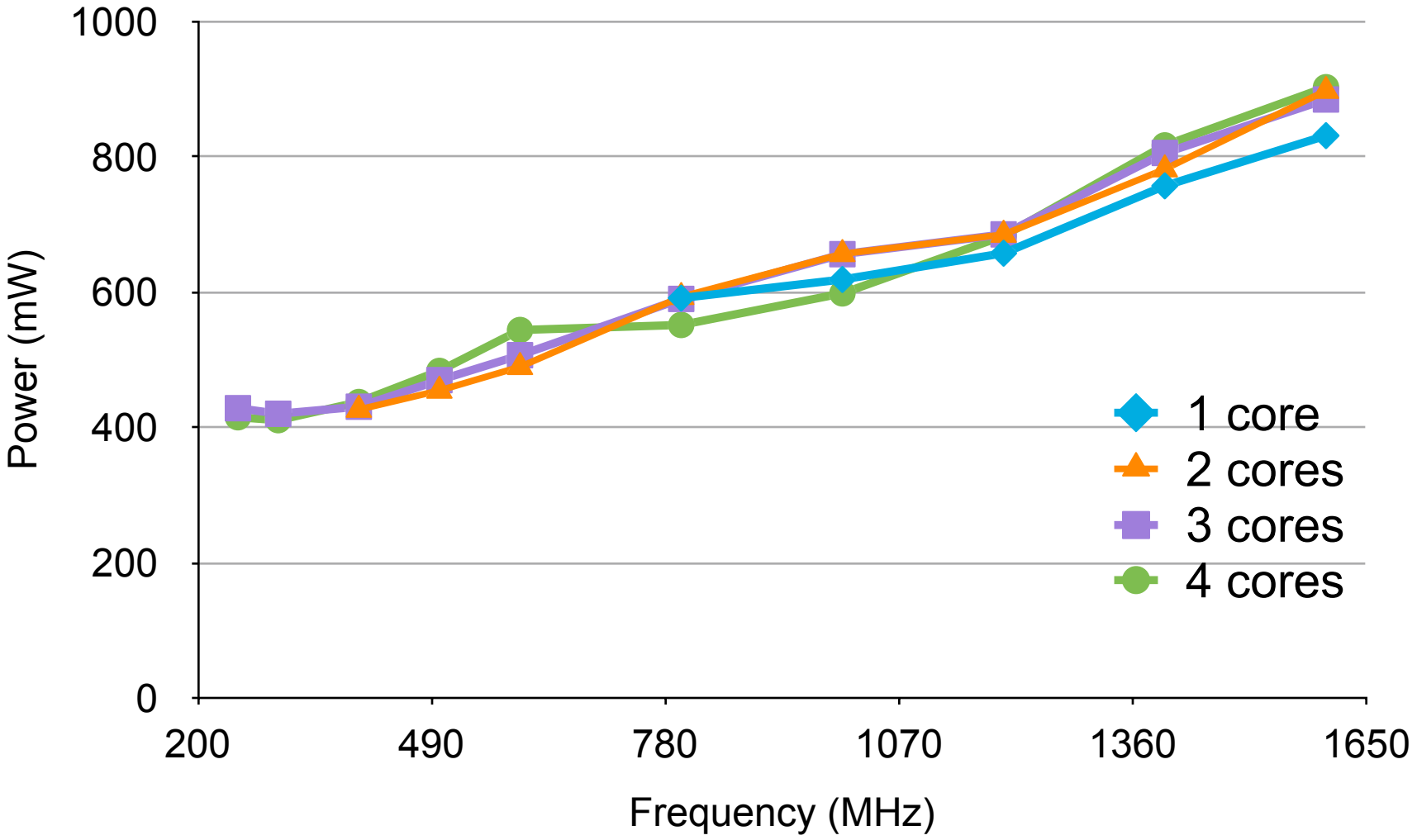
Exynos: loadcpu 25%



Exynos: loadcpu 50%



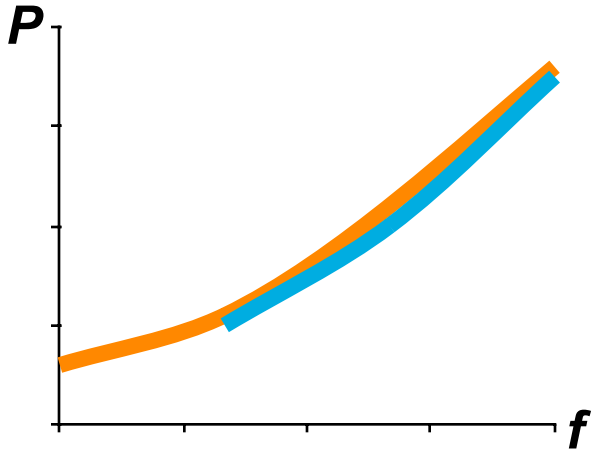
Exynos: loadcache 10%



Exynos: observations

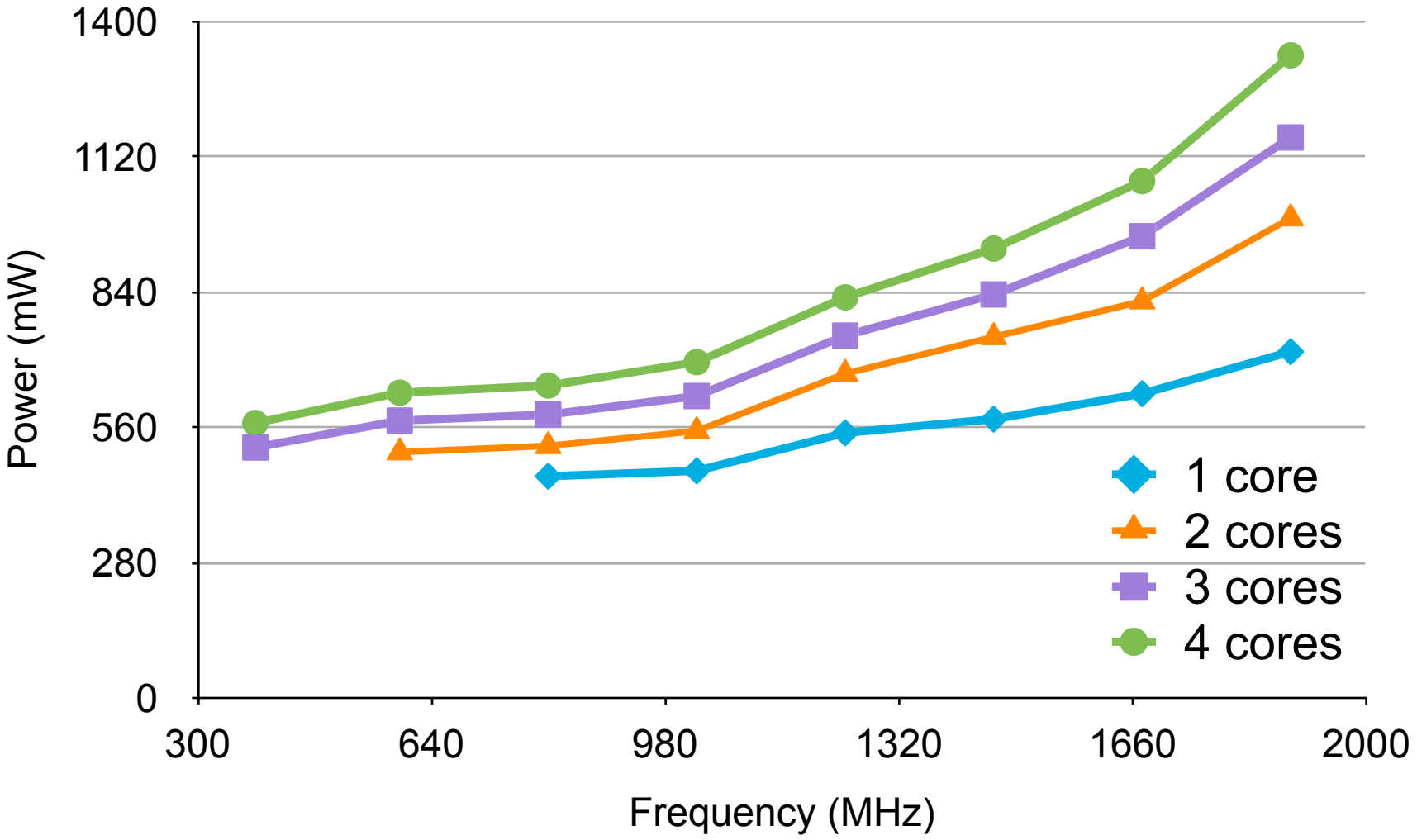
NICTA

- Fix ***n*** : lower ***f*** $\Rightarrow$ lower ***P***
- Fix ***f*** : ***P*** is independent of ***n***

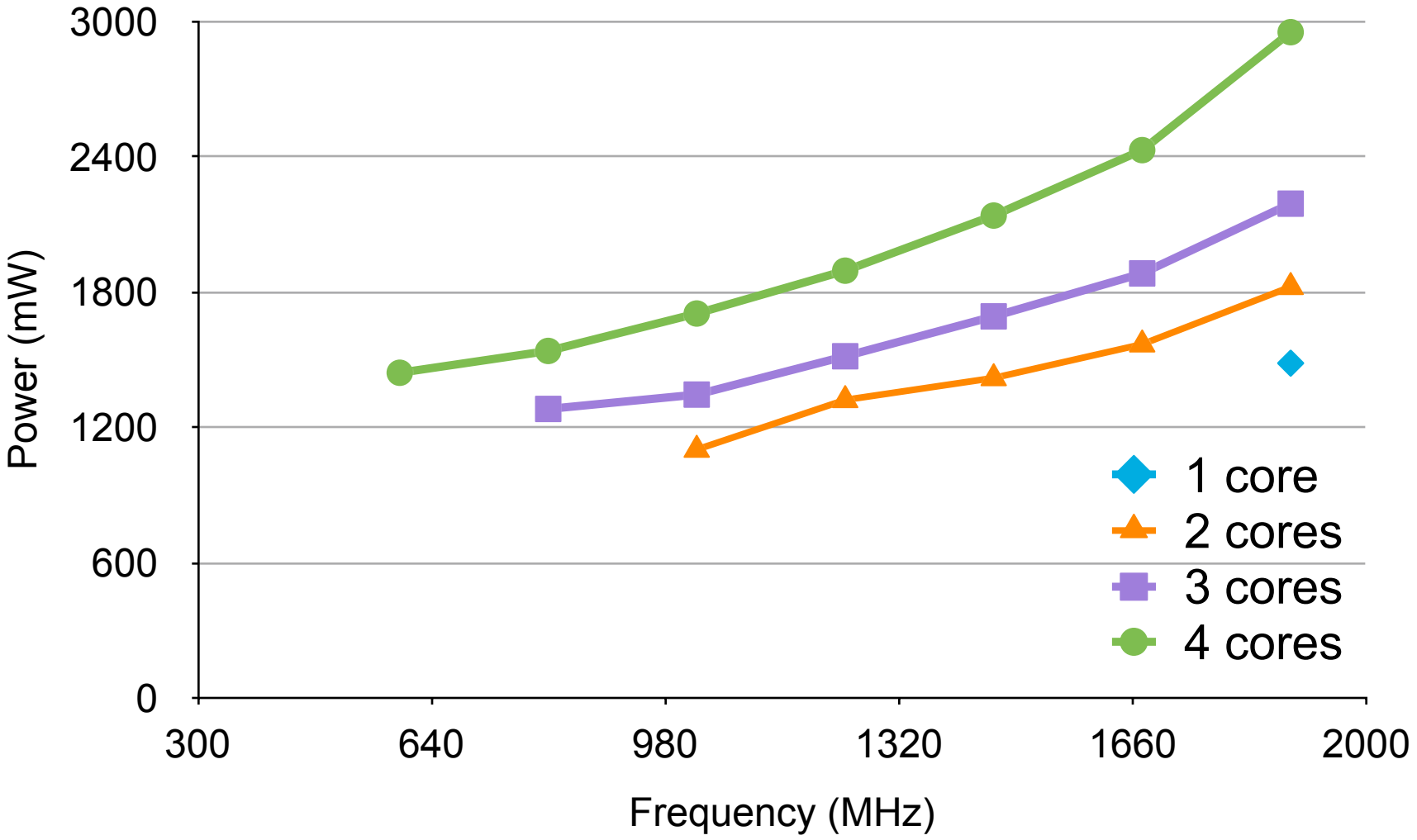


- Run at **max** { ***n*** }, **min** { ***f*** }

Snapdragon: loadcpu 10%

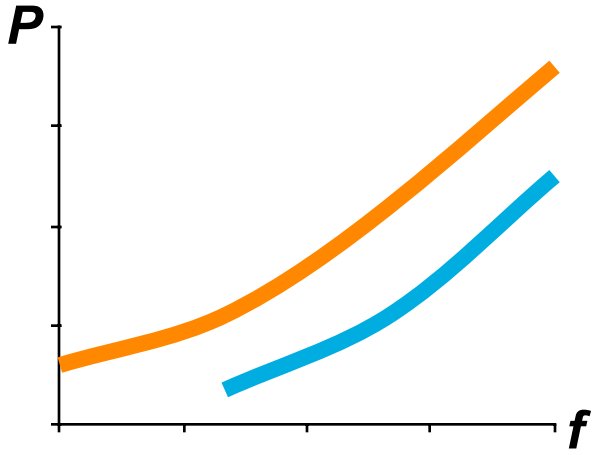


Snapdragon: loadmem 25%



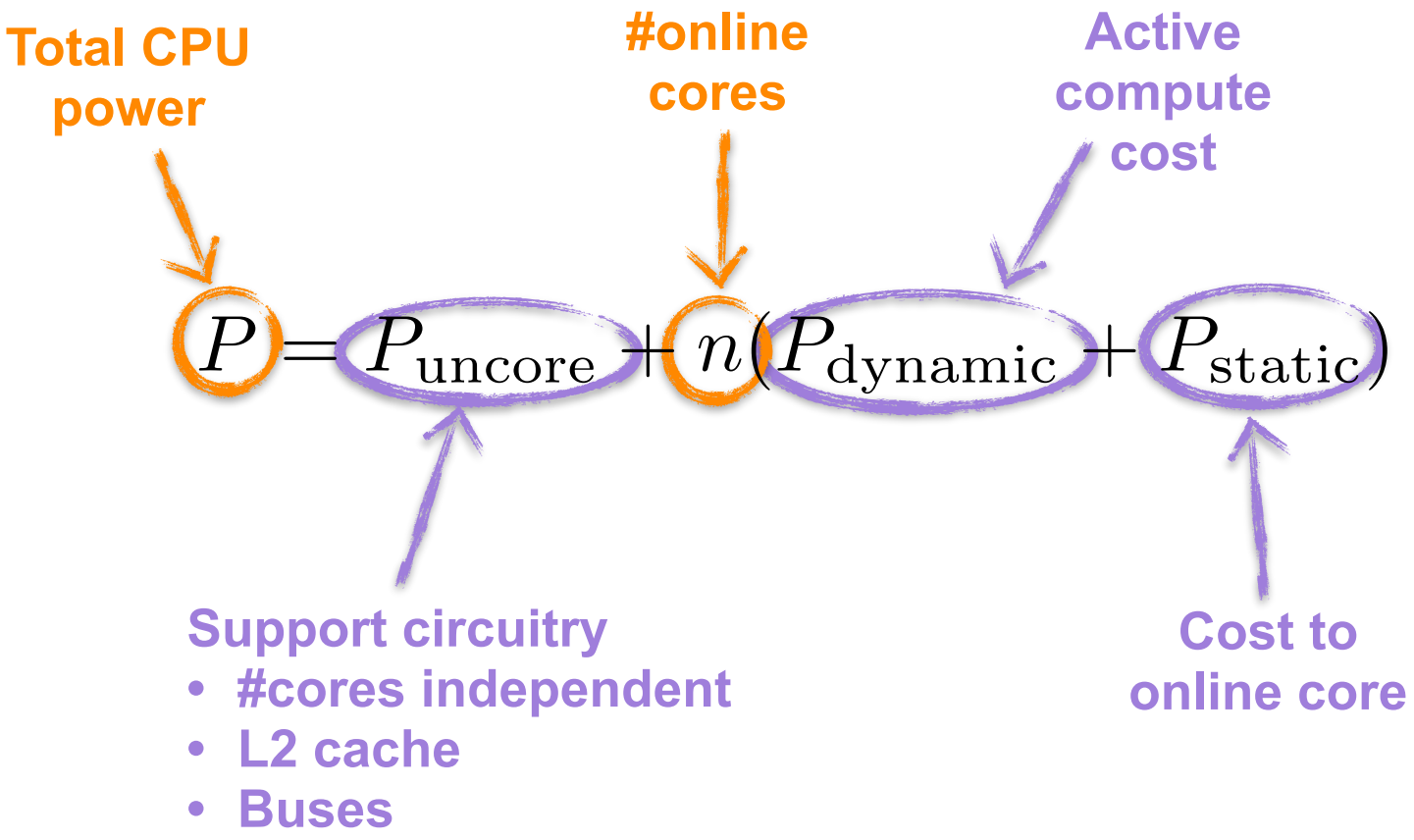
Observations

- Fix ***n*** : lower ***f*** $\Rightarrow$ lower ***P***
- Fix ***f*** : lower ***n*** $\Rightarrow$ lower ***P***



- Run at **???**

CPU model


Total CPU power
#online cores
Active compute cost
$P = P_{\text{uncore}} + n(P_{\text{dynamic}} + P_{\text{static}})$
Support circuitry
• #cores independent
• L2 cache
• Buses
Cost to online core

Analysis

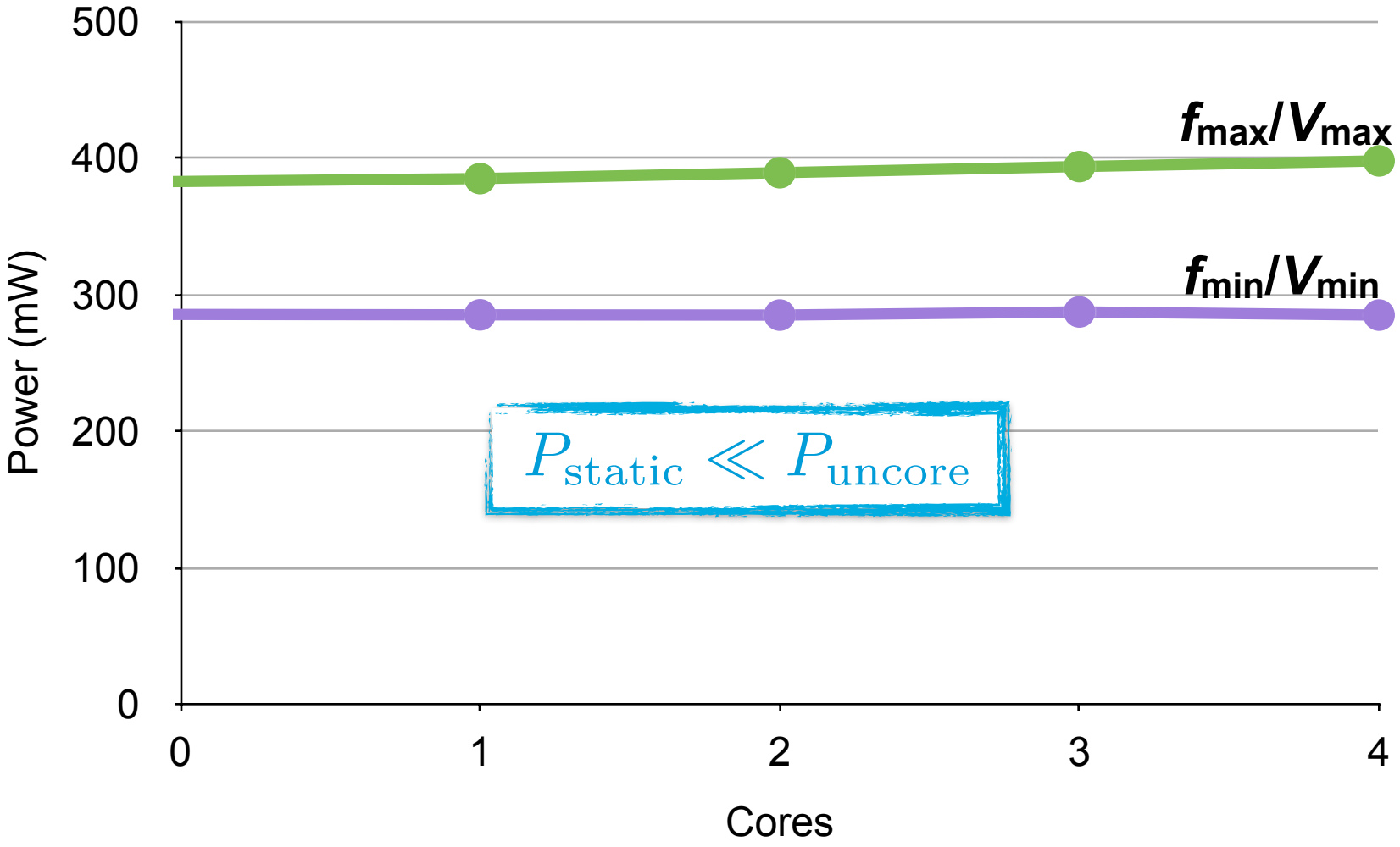
$$P = P_{\text{uncore}} + n(P_{\text{dynamic}} + P_{\text{static}})$$

$$E = (P_{\text{uncore}} + nP_{\text{static}})T + k$$ (***T*** = period)

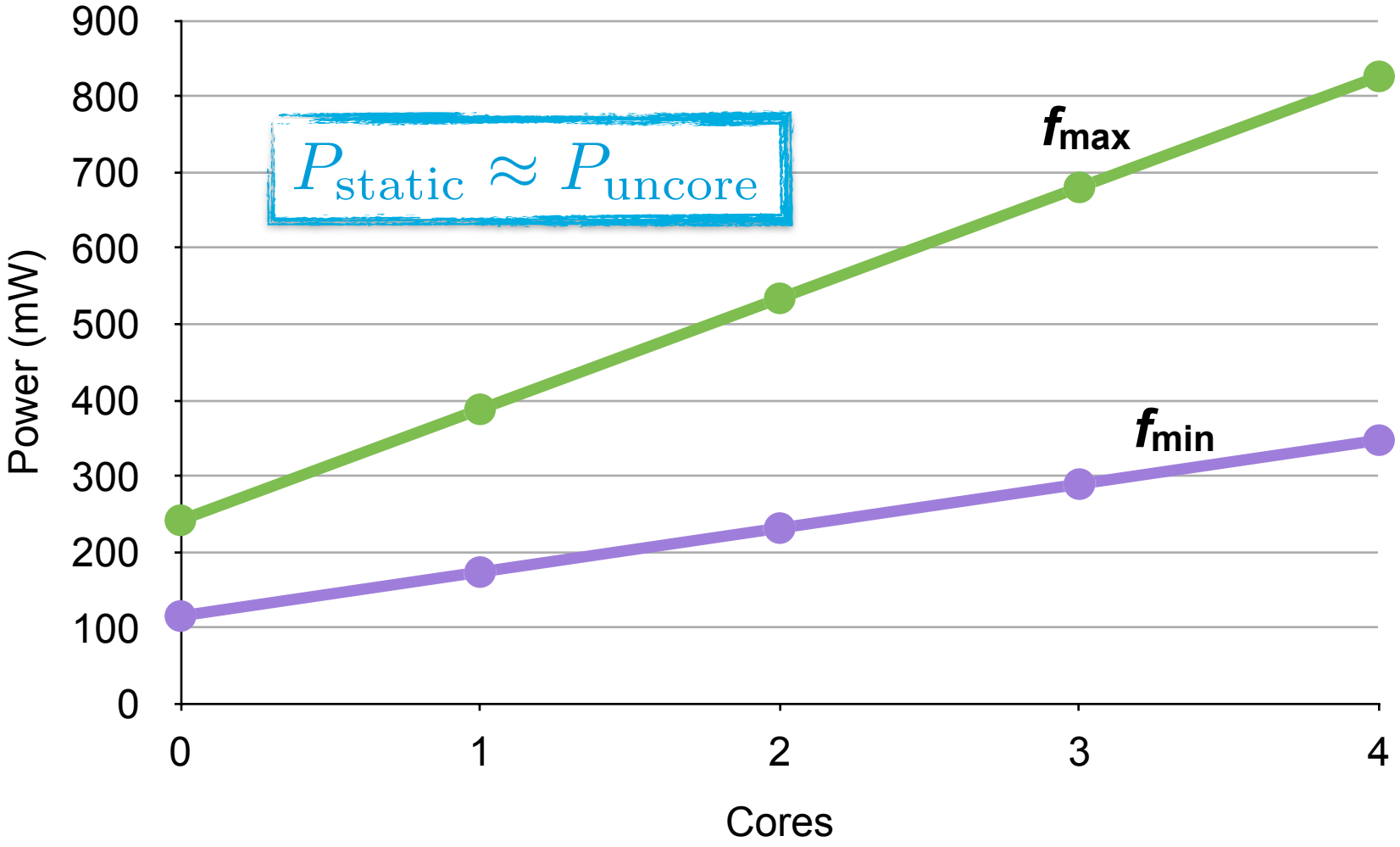
So if $P_{\text{static}} \ll P_{\text{uncore}}$:

E is **independent** of ***n***

Exynos: idle power



Snapdragon: idle power

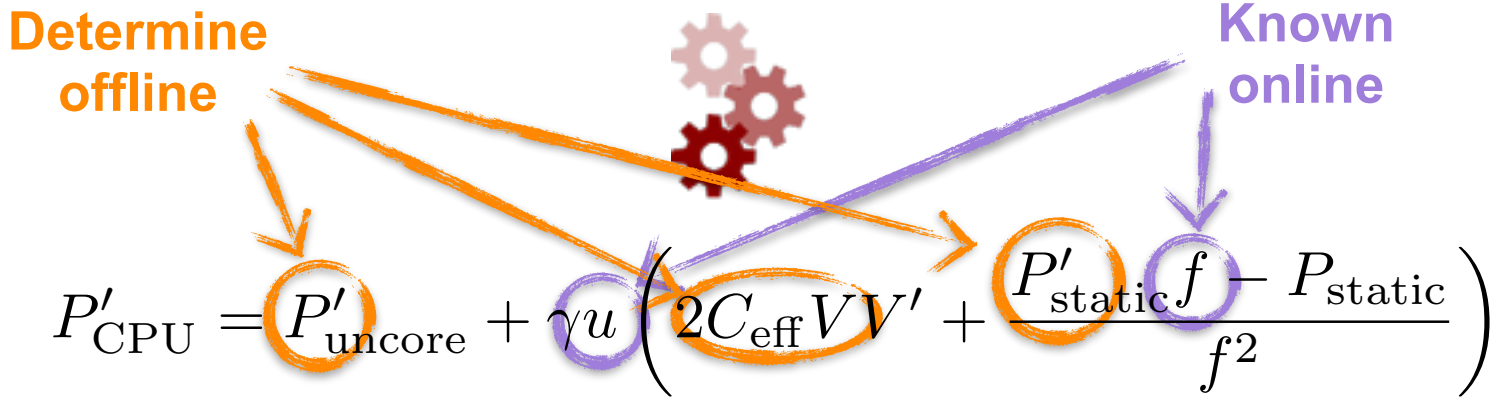


Analysis

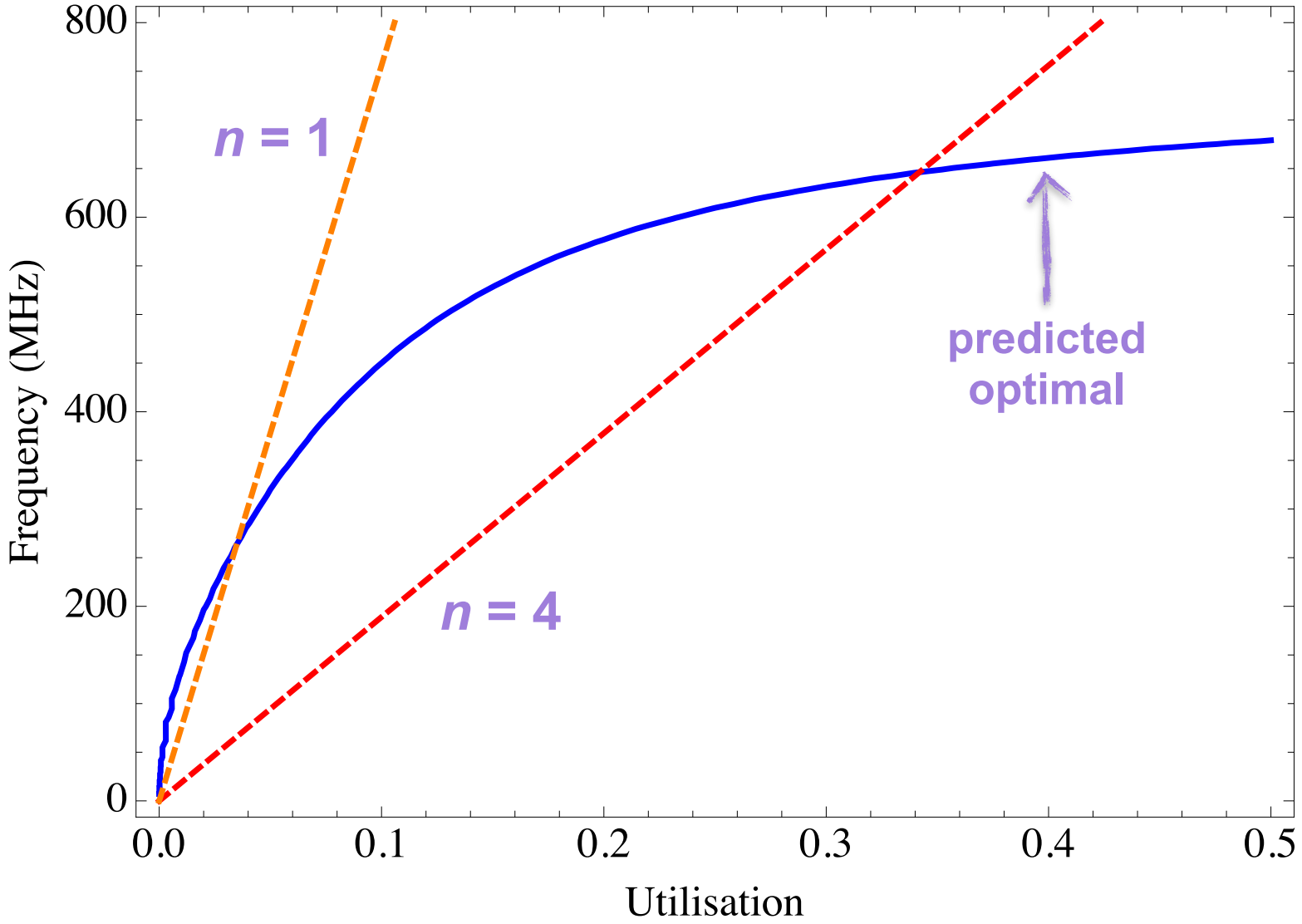
- What if P_{static} is significant?

$$P = P_{\text{uncore}} + n(P_{\text{dynamic}} + P_{\text{static}})$$

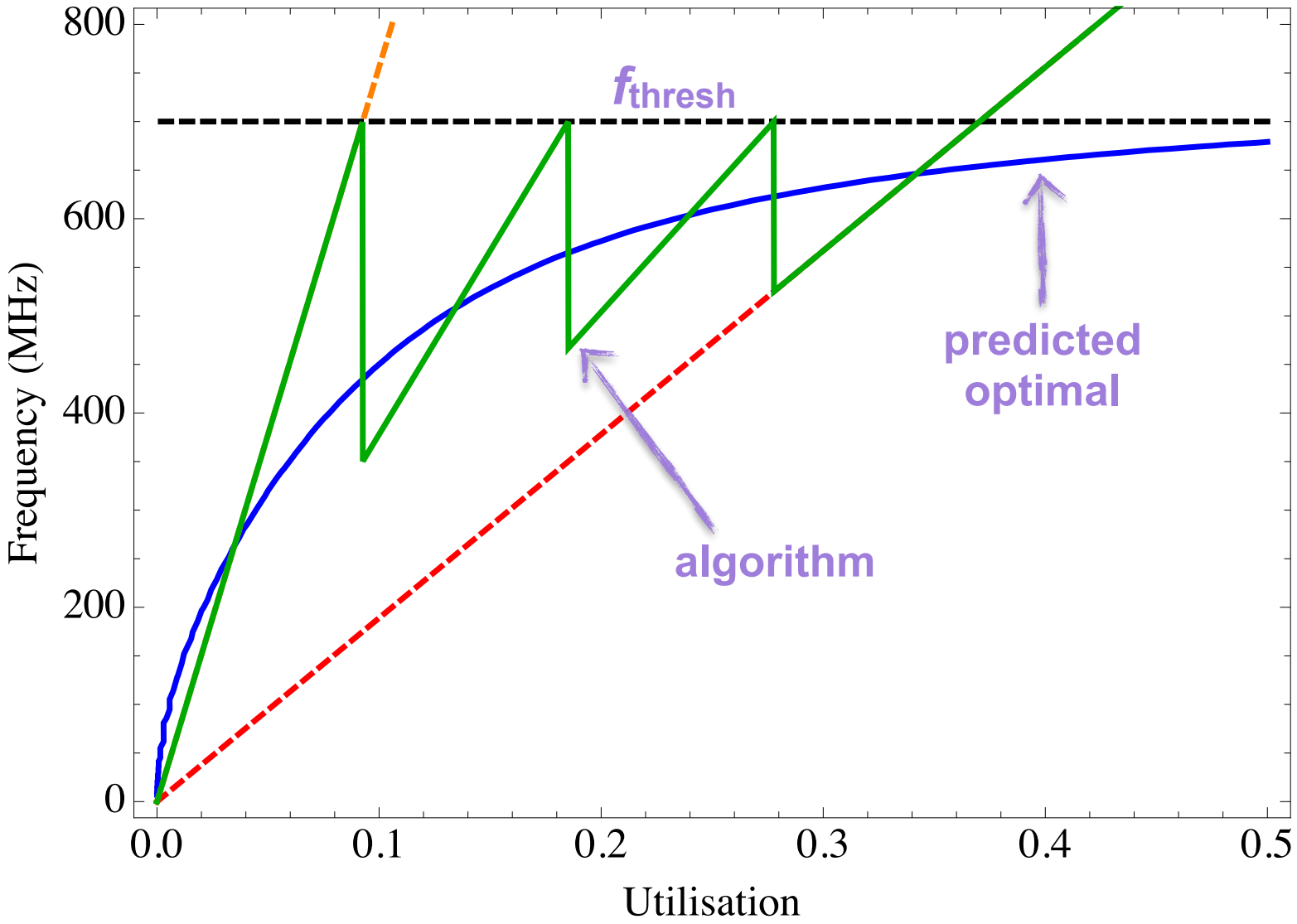
- Assumption: $n \propto 1/f$ (i.e. scalability)
- Then solve $\frac{\mathrm{d}}{\mathrm{d}f} P(f) = 0$



Analysis



Analysis



medusa

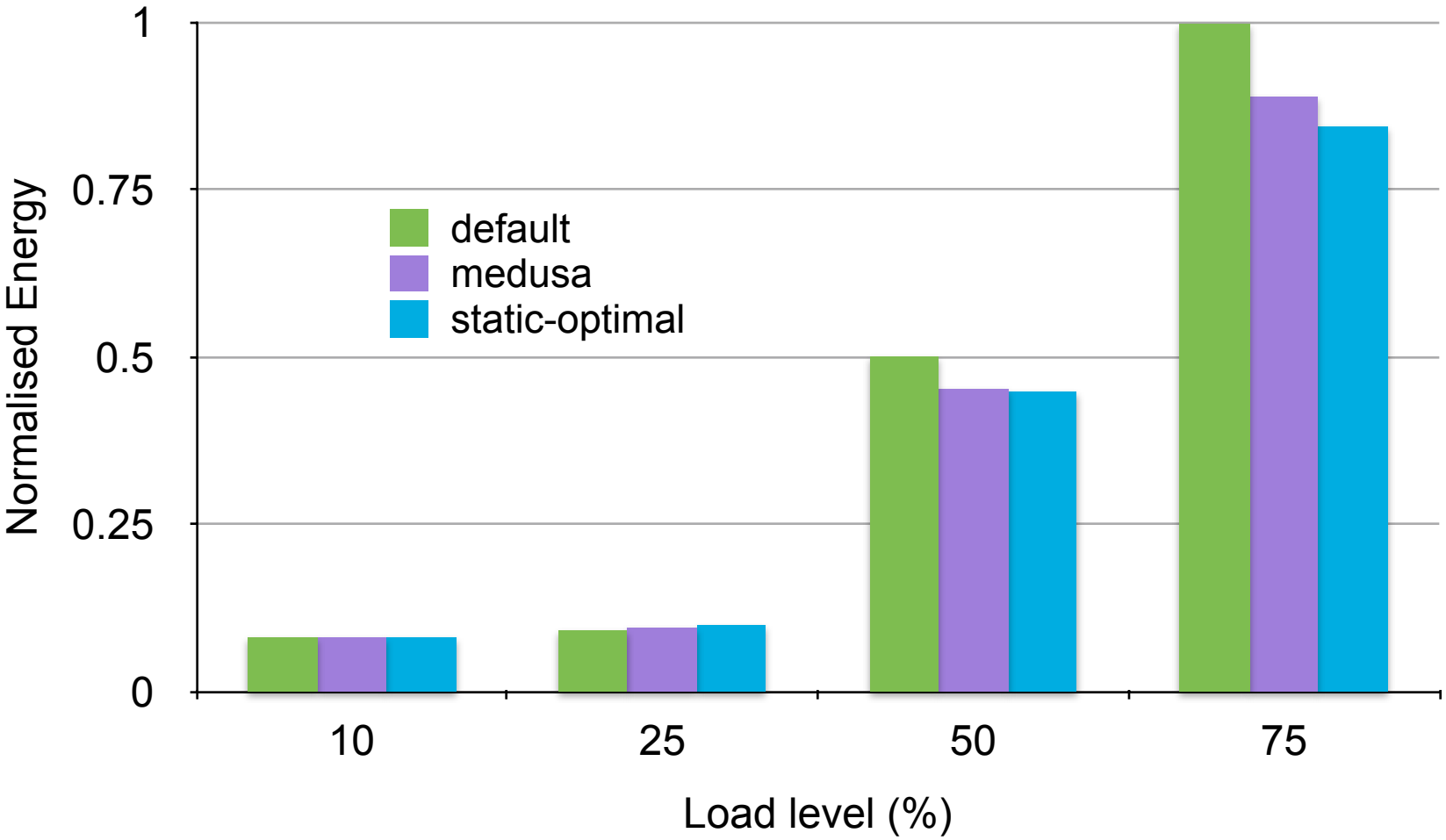
- **medusa**: Linux offline-aware governor
- keep utilisation < 100%
- parameterised by f_{thresh}
  - maintain $f < f_{thresh}$ by onlining cores as required
- practical considerations
  - available parallelism
  - changing workload
  - predicting effect of OP change
  - limiting OP switching

Determining f_{thresh}

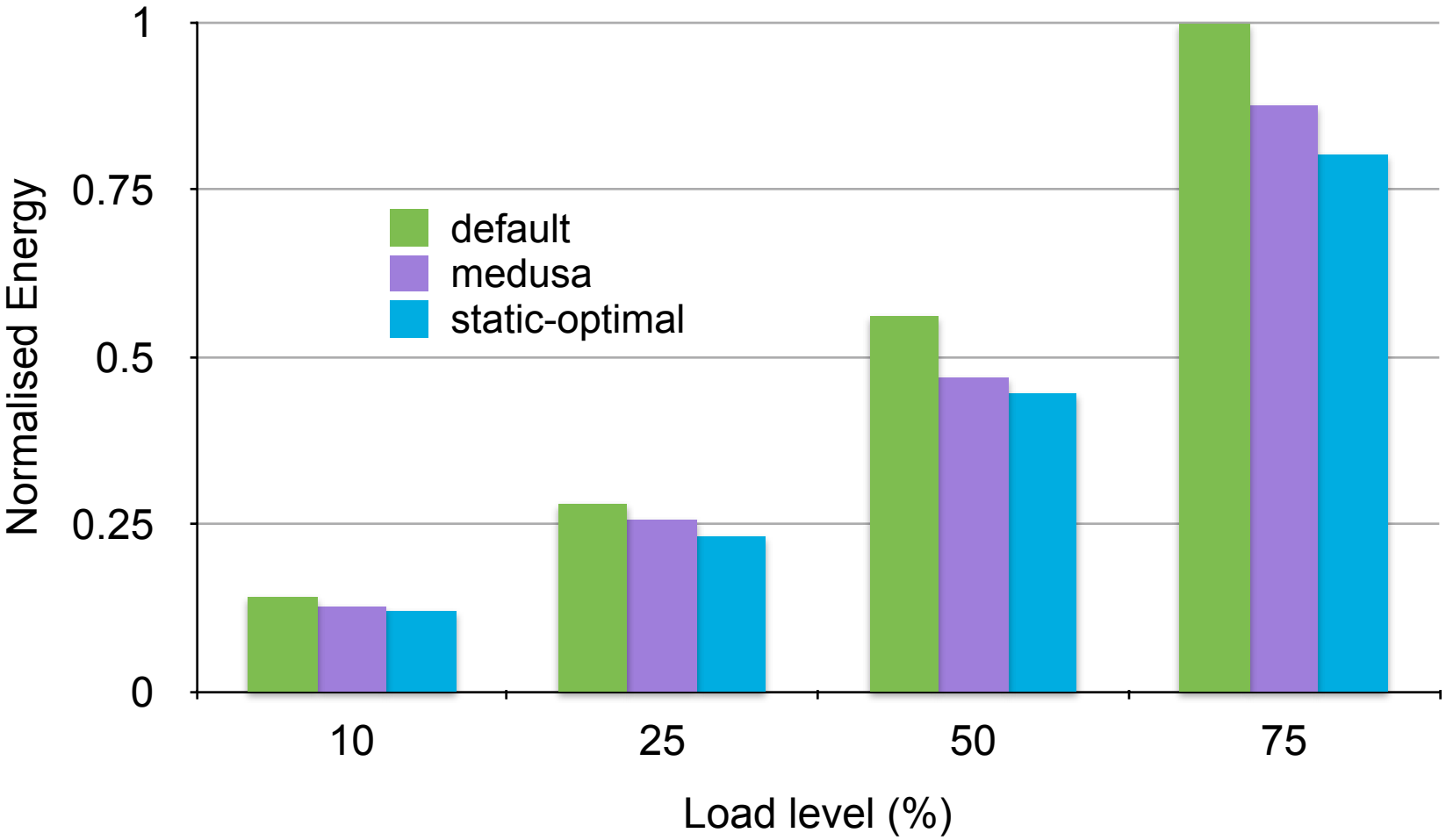
- Low static power
  - zero!
  - always online cores before increasing ***f***
- Otherwise
  - experimental
  - analytical

Exynos: medusa loadgen

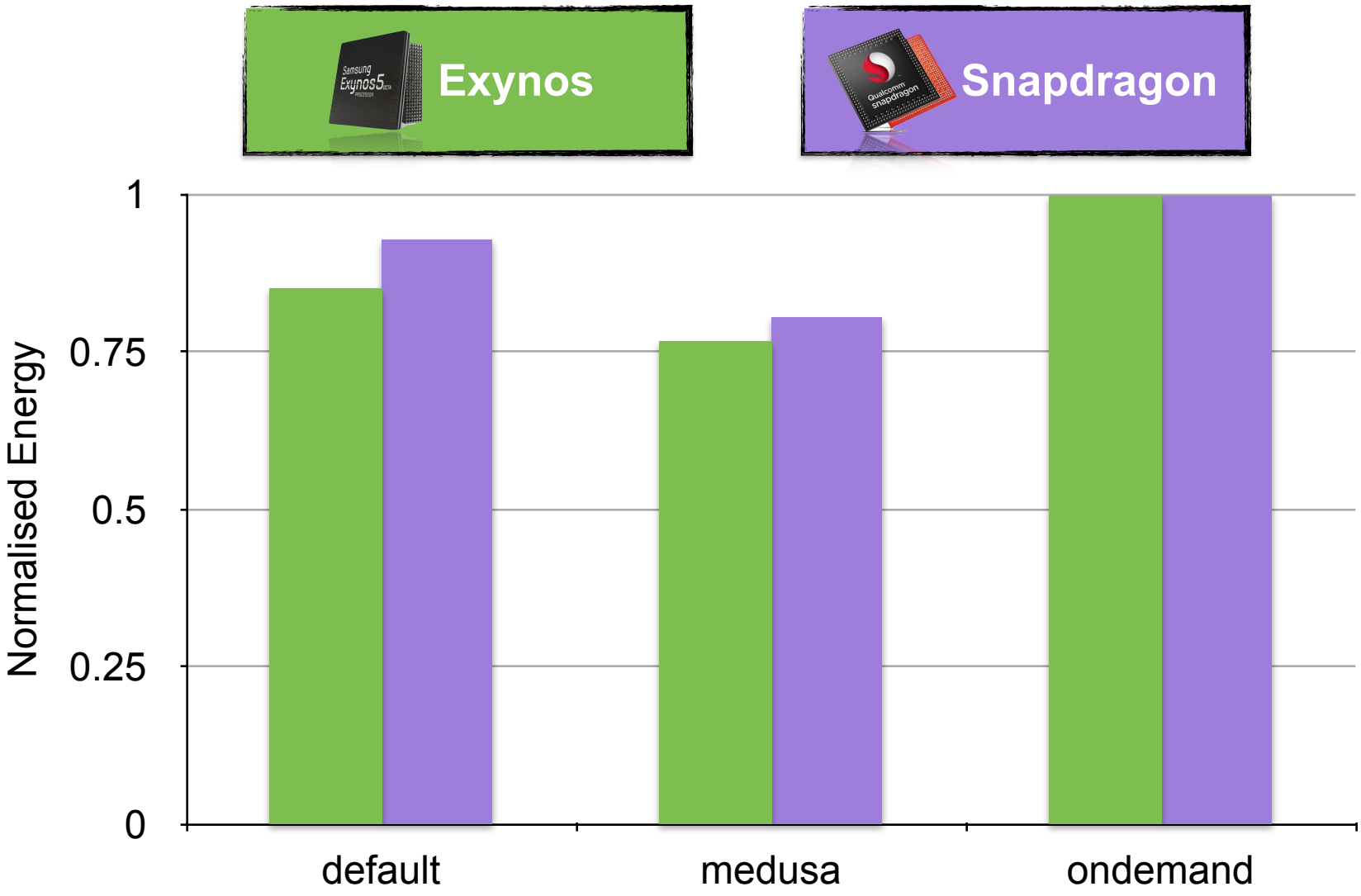
NICTA



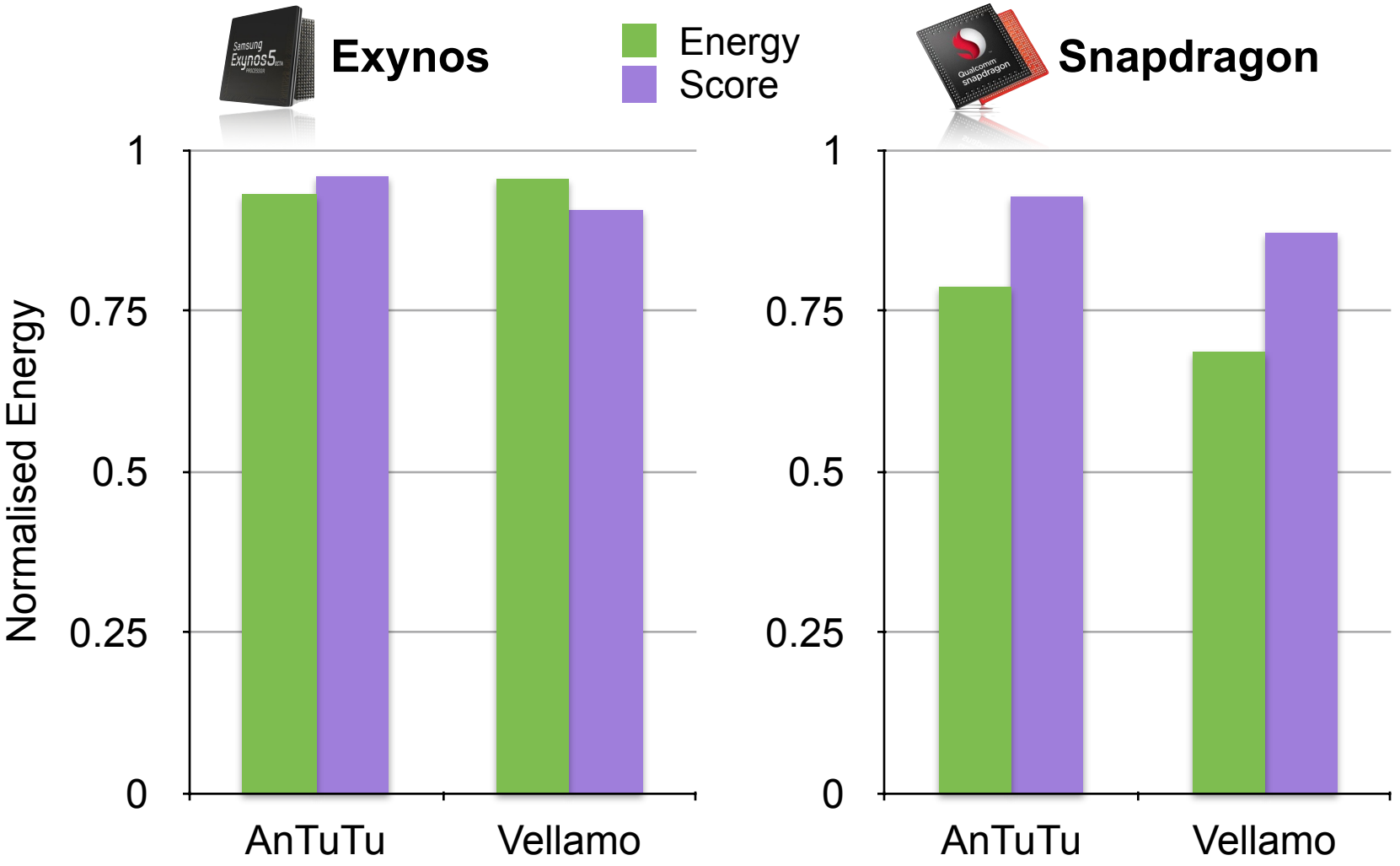
Snapdragon: medusa loadgen



medusa H.264 playback



medusa benchmarks



Summary

- Online cores for more efficient ***f***
- Low $\boldsymbol{P}_{\text{static}}$:
  - ***P*** independent of ***n***
  - increase ***n*** before ***f***
- High $\boldsymbol{P}_{\text{static}}$:
  - increase ***f*** to a point ($\boldsymbol{f}_{\text{thresh}}$)
  - then prefer to increase ***n***

Summary

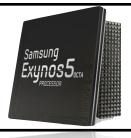
Download medusa at

ssrg.nicta.com.au/projects/energy-management

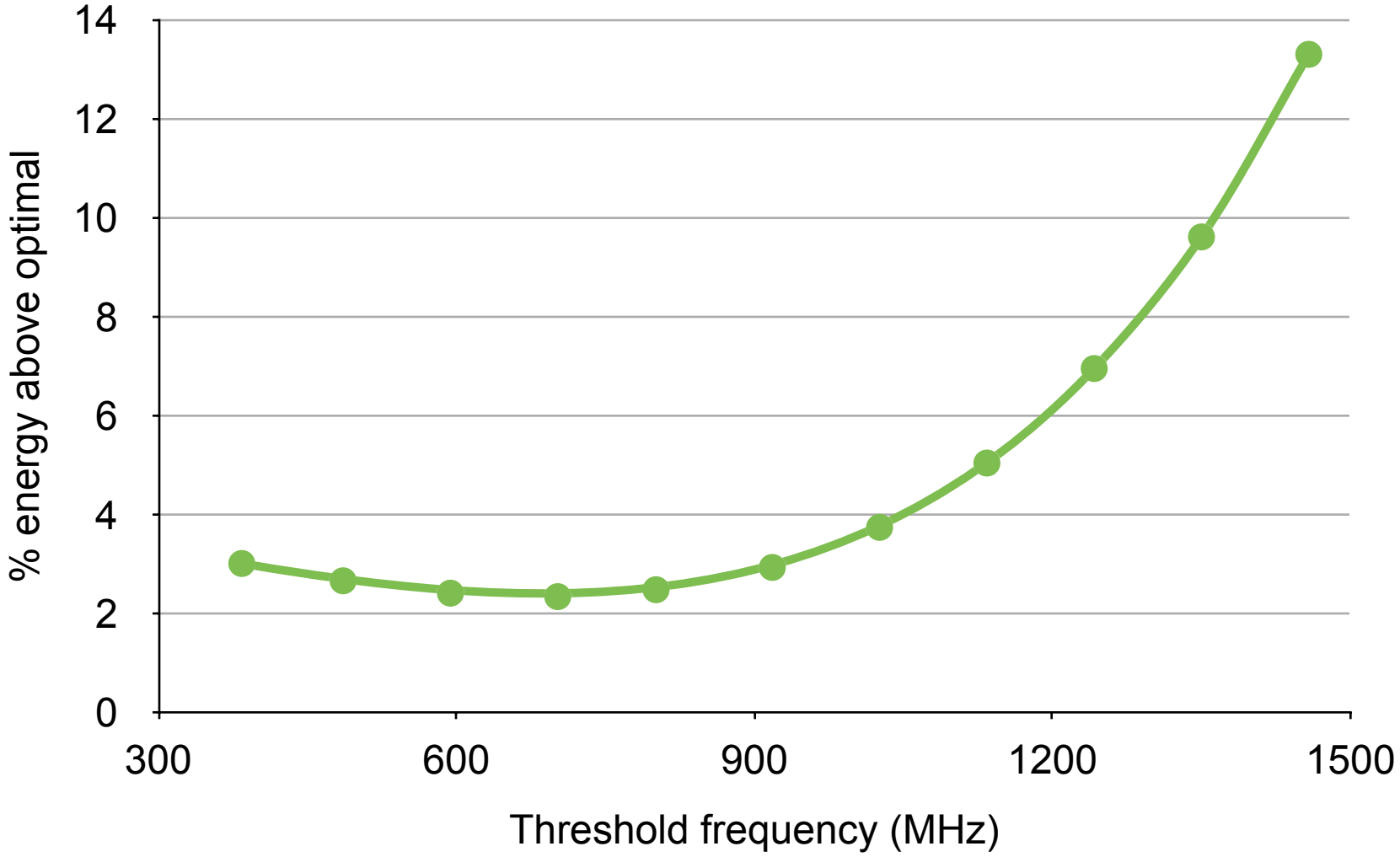
Unifying DVFS and Offlining in Mobile Multicores

Aaron Carroll
Gernot Heiser

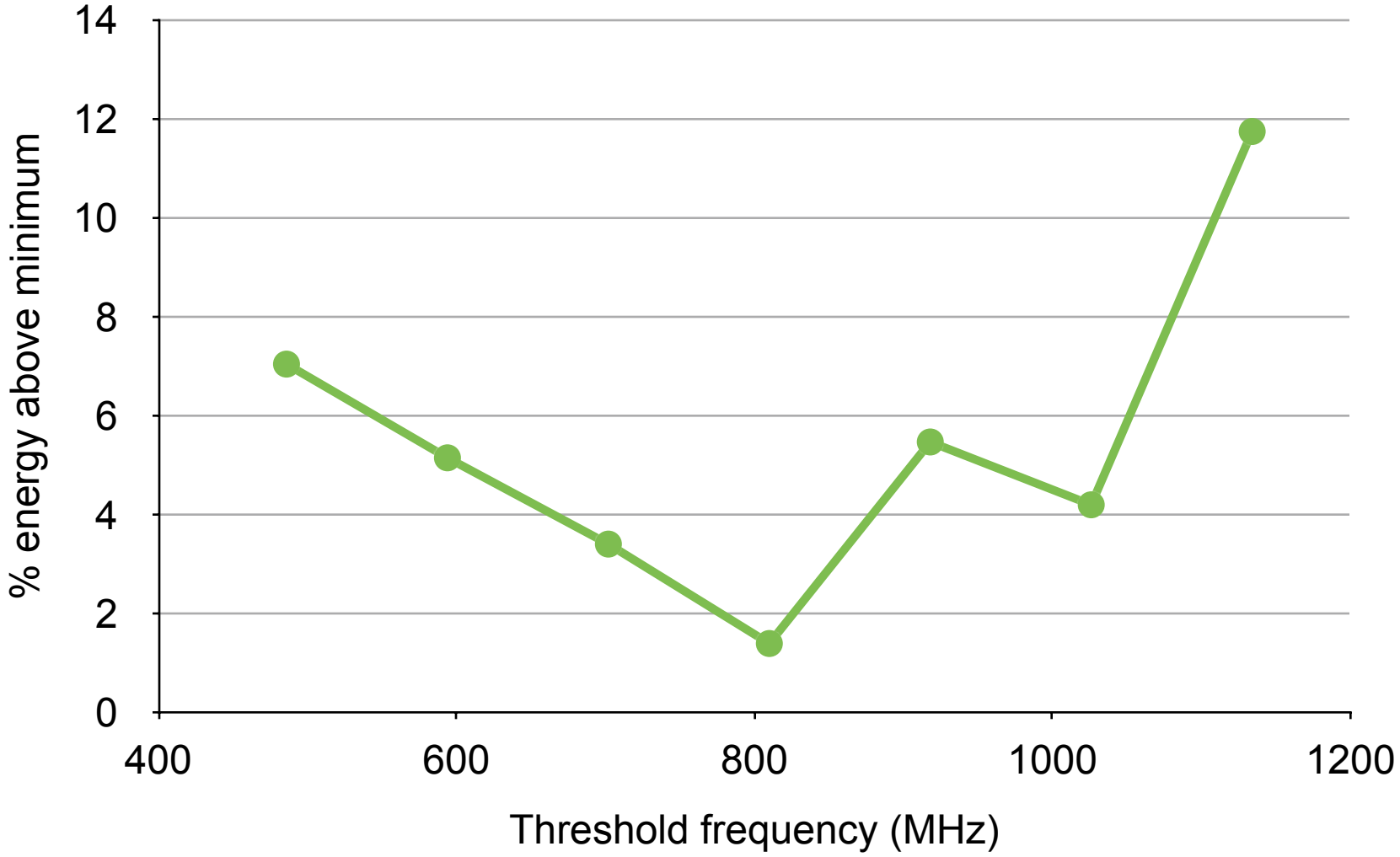
OP transition cost

	 Exynos		 Snapdragon	
	t (ms)	*P* (mW)	*t* (ms)	*P* (mW)
online/offline	22.6	357	11.7	454
online/offline	16.5	1305	4.09	1926
$f_{min} \rightarrow$	2.10	1033	11.1	420
$f_{min} \rightarrow$	0.39	320	6.05	317
$f_{max} \rightarrow$	1.20	1400	0.24	838

f_{thresh} prediction



f_{thresh} sensitivity



P_{static} variation