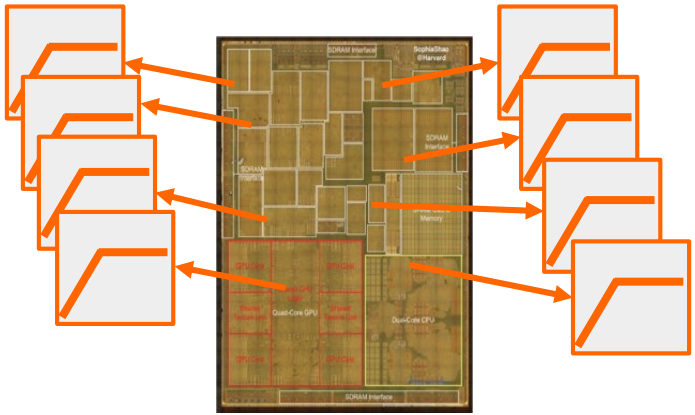
Gables: A Roofline Model for Mobile SoCs

Mark D. Hill, Wisconsin & **Former Google Intern**
Vijay Janapa Reddi, Harvard & **Former Google Intern**

HPCA, Feb 2019

Outline

- Motivation
- Gables Model
- Example Balanced Design
- Wrap Up



Executive Summary

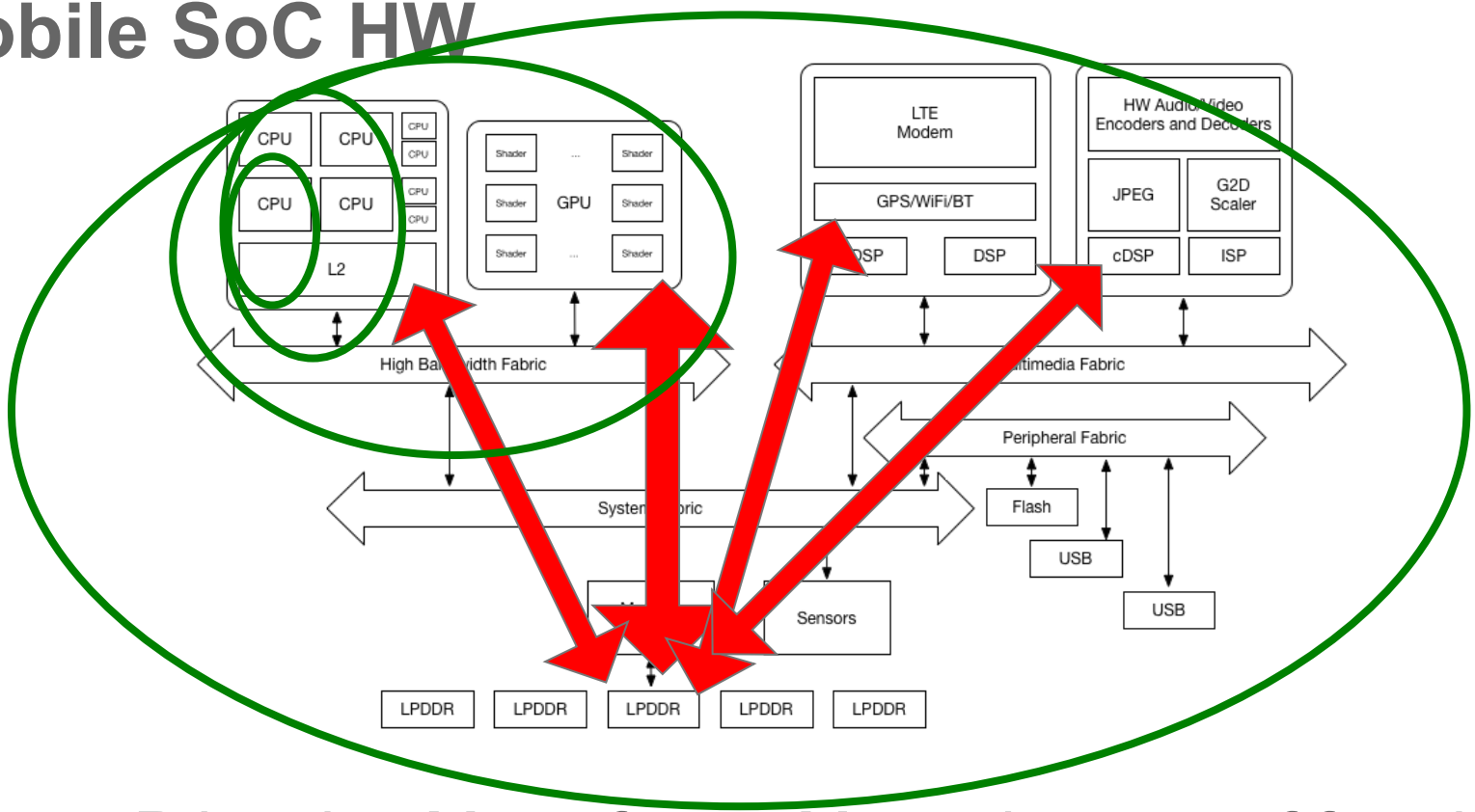
Mobile SoCs have “extreme heterogeneity”

- CPUs, GPUs, DSPs, & 10+ other “IPs” (accelerators)
- Which IPs have potential? How big? How many?
- Need initial answers before authoring IP HW/SW

Gables

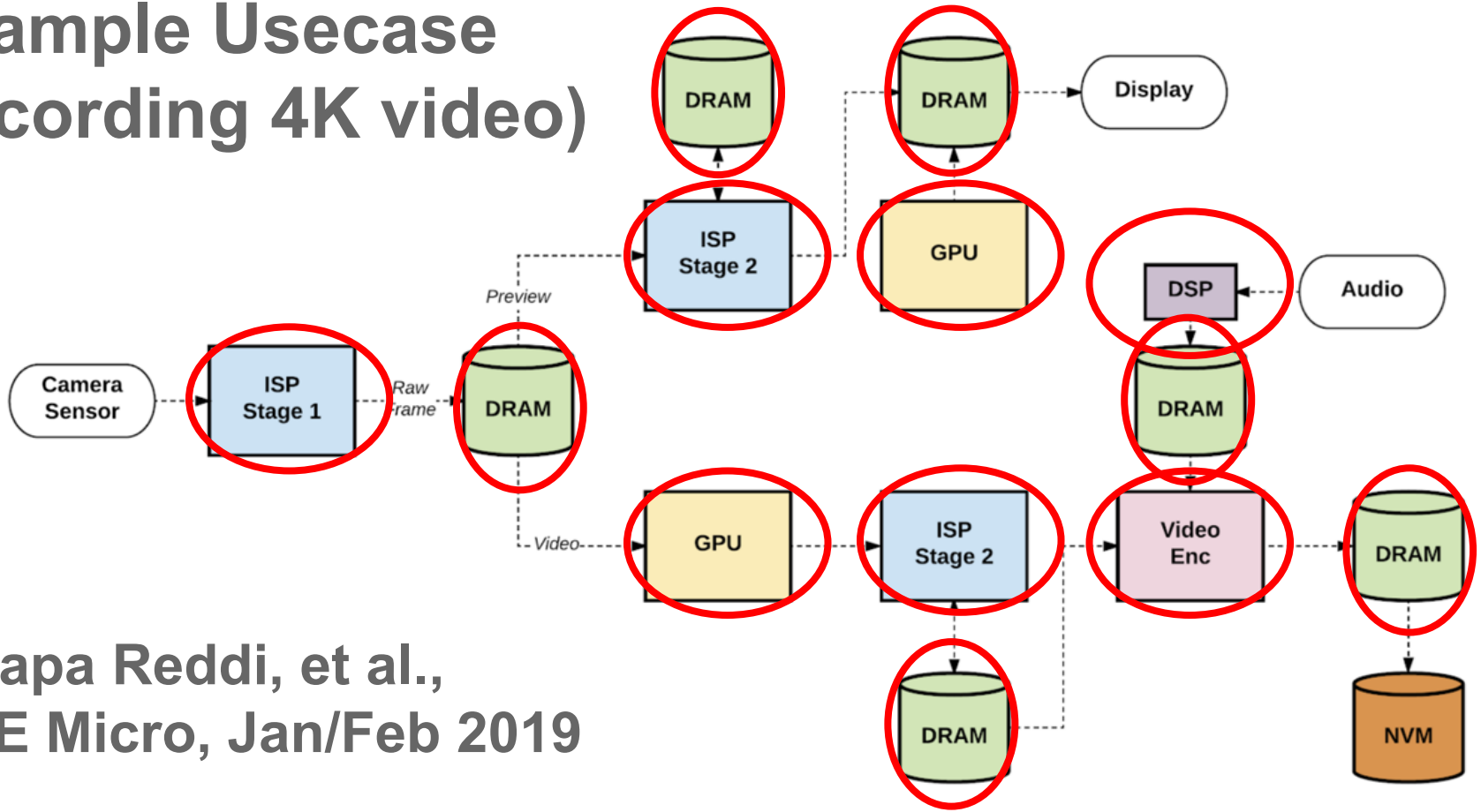
- Models give initial answers: Amdahl’s Law & Roofline
- Gables: Roofline per IP & apportion concurrent work
- E.g., how much IP[i] acceleration needed?

Mobile SoC HW



Many IP blocks; Many flows, Many degrees of freedom

Example Usecase (recording 4K video)



Janapa Reddi, et al.,
IEEE Micro, Jan/Feb 2019

Multiple IPs compute & repeated off-chip bandwidth use!

Mobile SoCs Run Usecases

	AP	Display	G2DS	GPU	ISP	JPEG	IPU	VDEC	VENC	DSP
HDR+	X	X		X	X	X	X			
Videocapture	X	X		X	X				X	
VideocaptureHDR	X	X		X	X				X	
VideoplaybackUI	X	X	X	X				X		
Google Lens	X	X	X	X						X

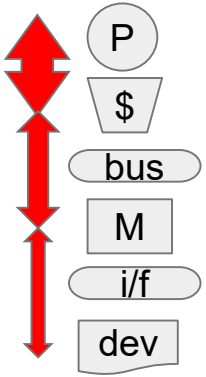
Must run each usecase sufficiently fast -- no need faster

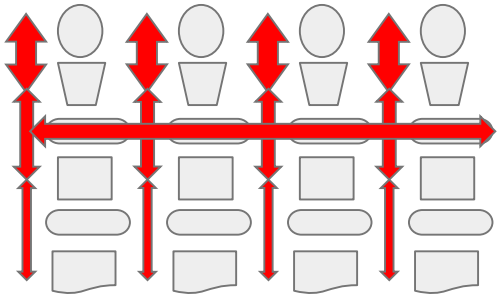
Must run all usecases – average irrelevant

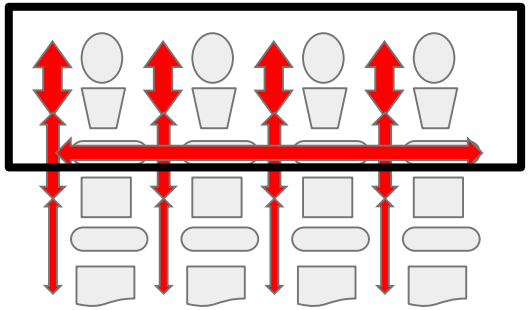
A usecase uses IPs **concurrently** – more than serially

For each usecase, how much IP[i] acceleration needed?

Computer Architecture & Models



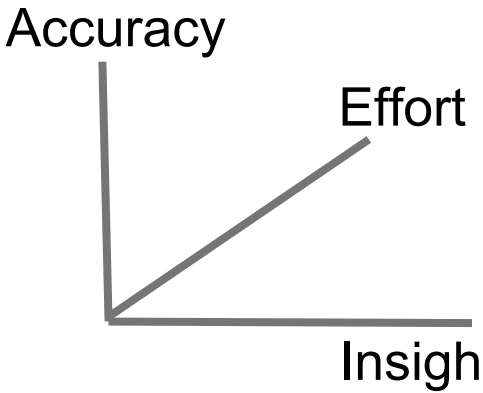




CPU &
Iron Law

Multiprocessor &
Amdahl's Law

Multicore &
Roofline



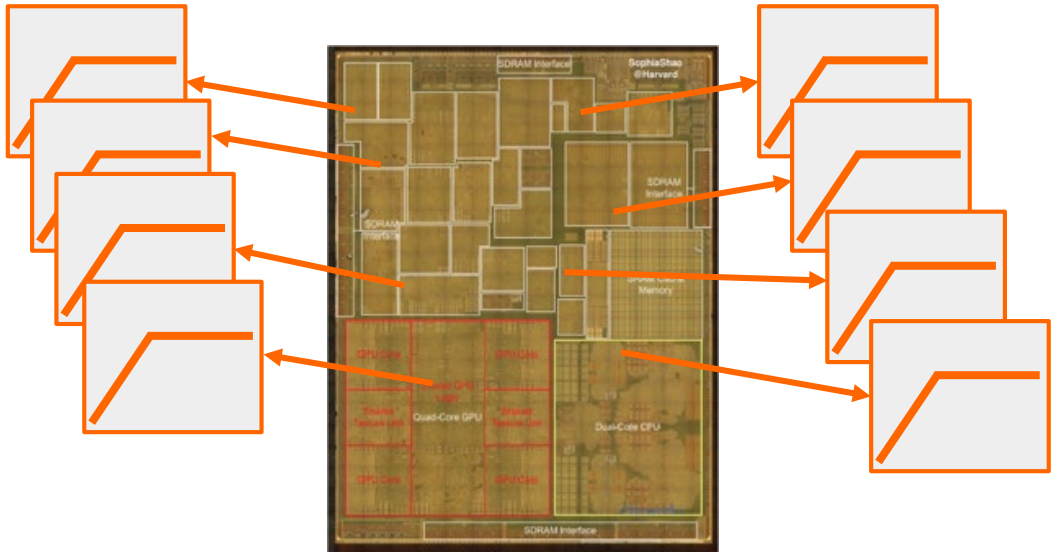
Models vs Simulation

- More insight
- Less effort
- But less accuracy

Models give first answer, not final answer

Gables builds on Roofline → SoC "first answer"

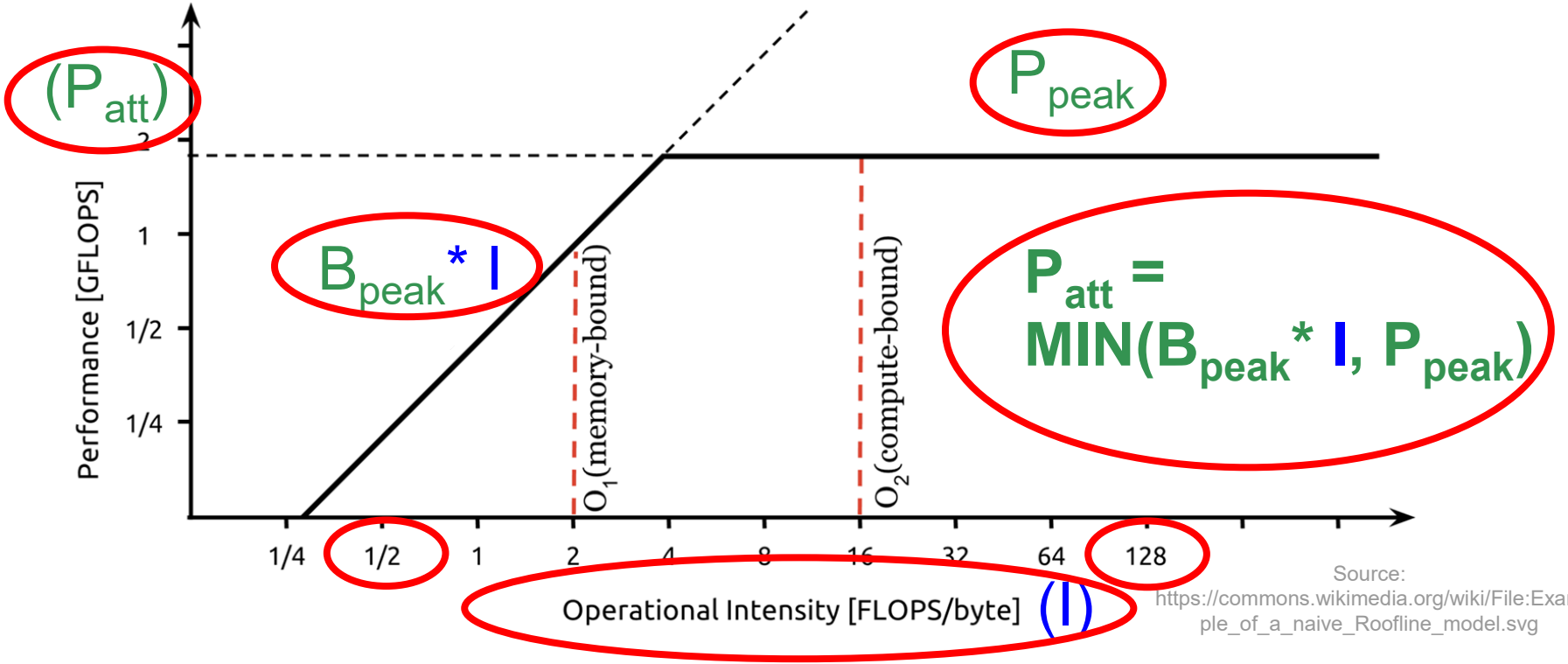
Moble System on Chip (SoC) & Gables



Gables uses Roofline per IP to provide first answer!

What's a Roofline?

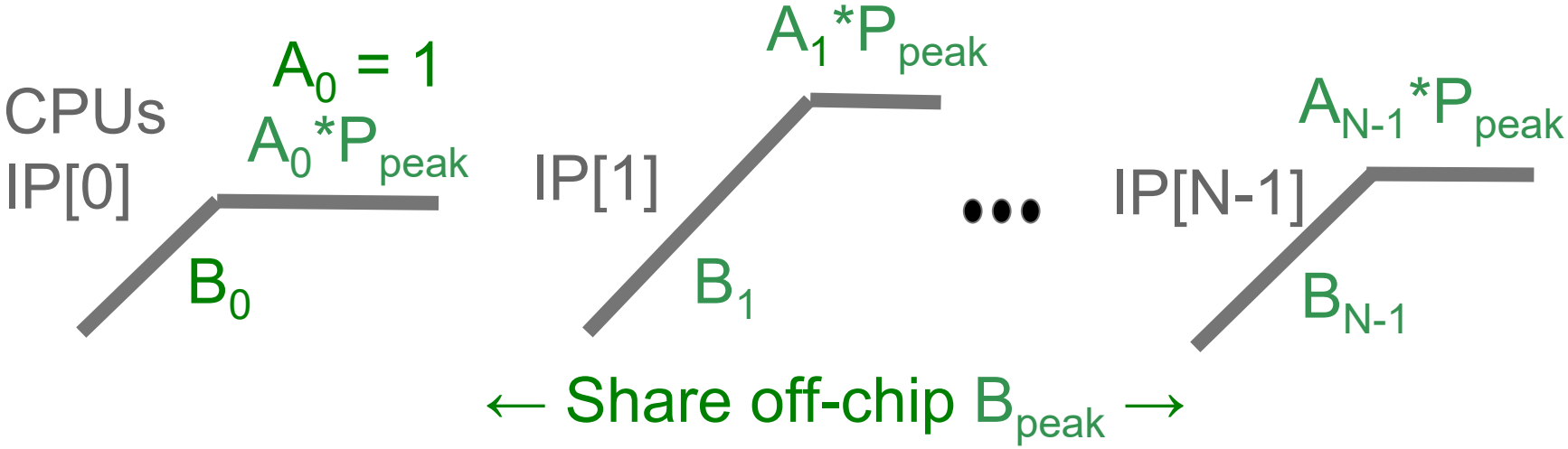
Williams et al., Roofline, CACM 4/2009



Source: https://commons.wikimedia.org/wiki/File:Example_of_a_naive_Roofline_model.svg

Compute v. Communication: Op. Intensity (I) = #operations / #off-chip bytes

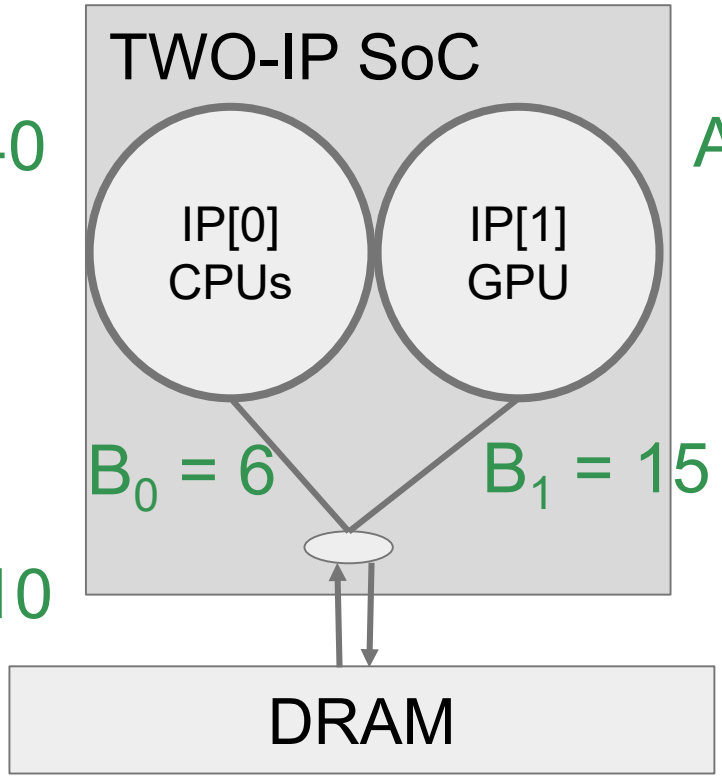
Gables for N IP SoC



Usecase at each IP[i]

- Non-negative work f_i (f_i's sum to 1) w/ IPs in parallel
- Operational intensity I_i operations/byte

Example Balanced Design Start w/ Gables



$P_{peak} = 40$

$A_1 * P_{peak} = 5*40 = 200$

$B_0 = 6$

$B_1 = 15$

$B_{peak} = 10$

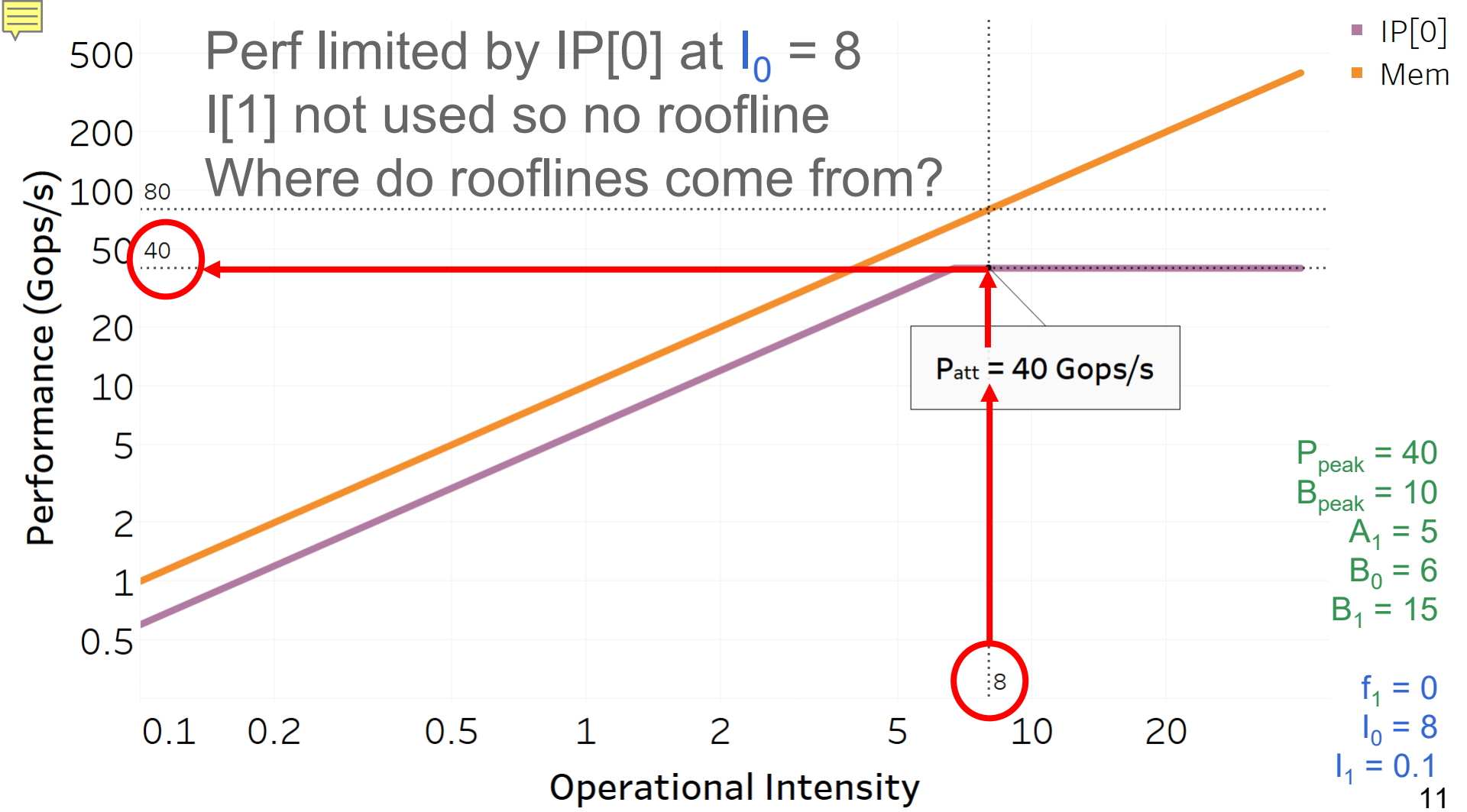
Workload (Usecase):

$f_0 = 1$ & $f_1 = 0$

$I_0 = 8$ = good caching

$I_1 = 0.1$ = latency tolerant

Performance?


Perf limited by IP[0] at I_0 = 8
I[1] not used so no roofline
Where do rooflines come from?
IP[0]
Mem
Performance (Gops/s)
500
200
100
80
50
40
20
10
5
2
1
0.5
0.1
0.2
0.5
1
2
5
10
20
8
Operational Intensity
Patt = 40 Gops/s
P_{peak} = 40
B_{peak} = 10
A_1 = 5
B_0 = 6
B_1 = 15
f_1 = 0
I_0 = 8
I_1 = 0.1

Gables Math: Roofline / Work Fraction

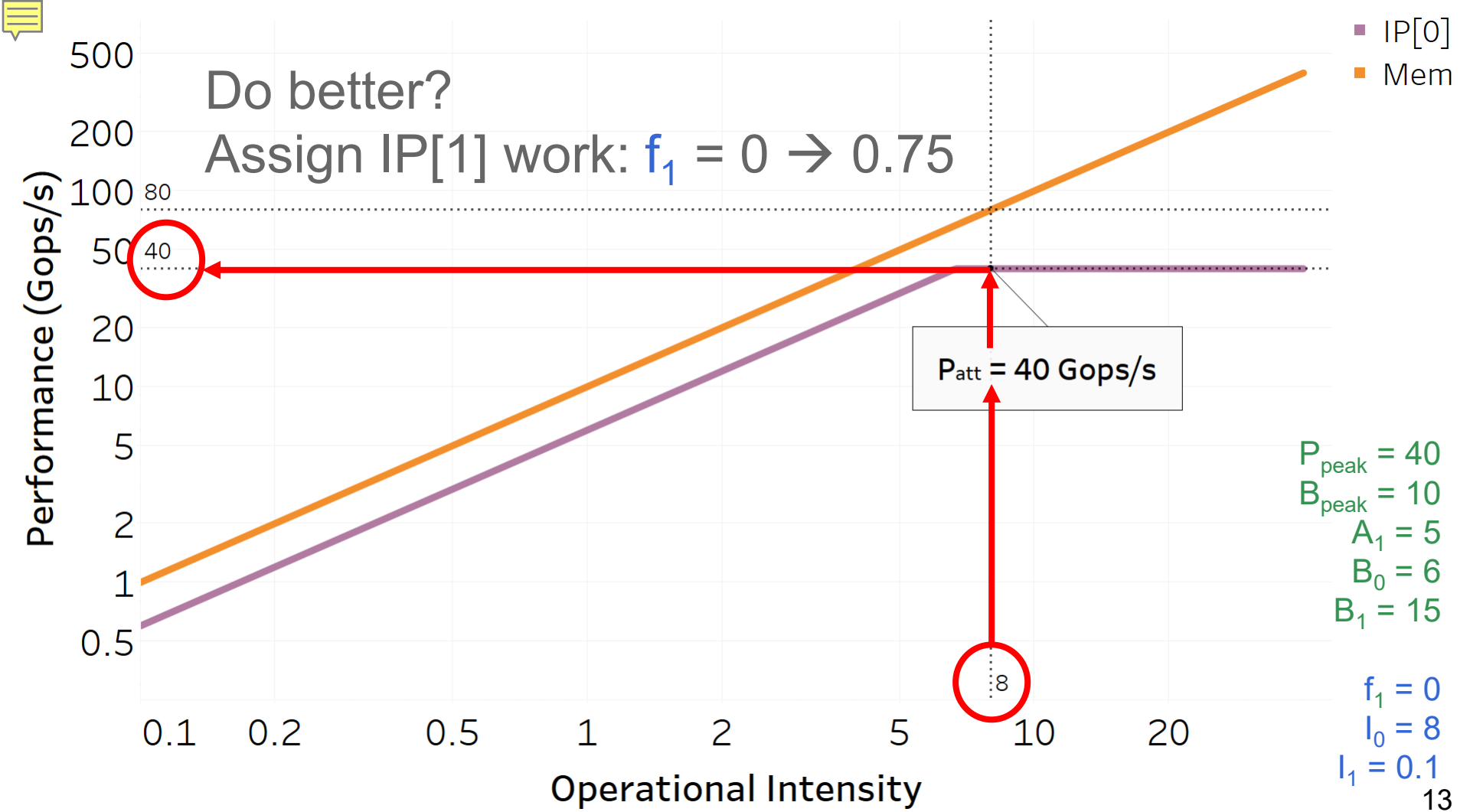
Roofline: $MIN(B_{peak} * I, P_{peak})$

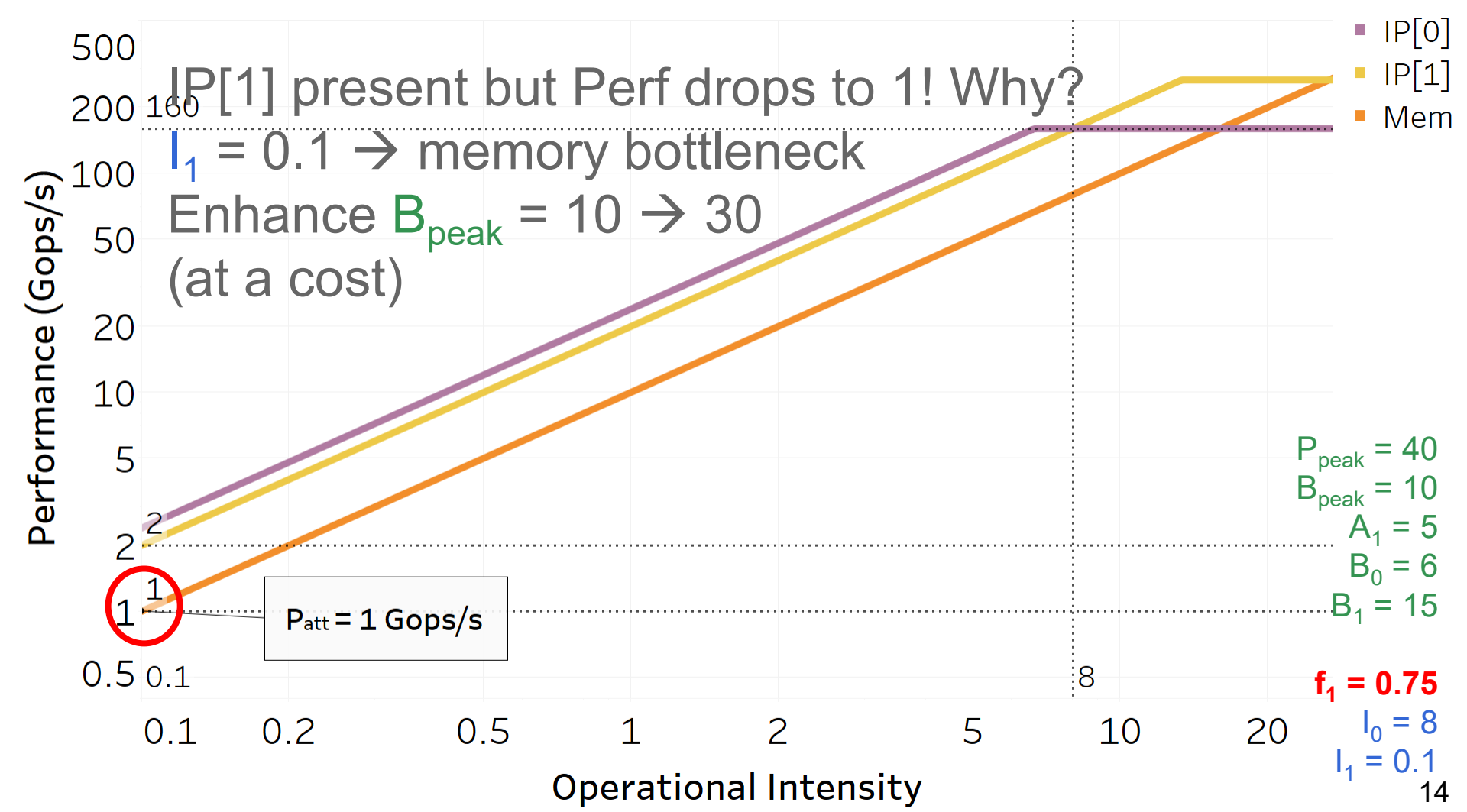
$MIN(B_{peak} * I, 1 * P_{peak}) \;/\; 1$

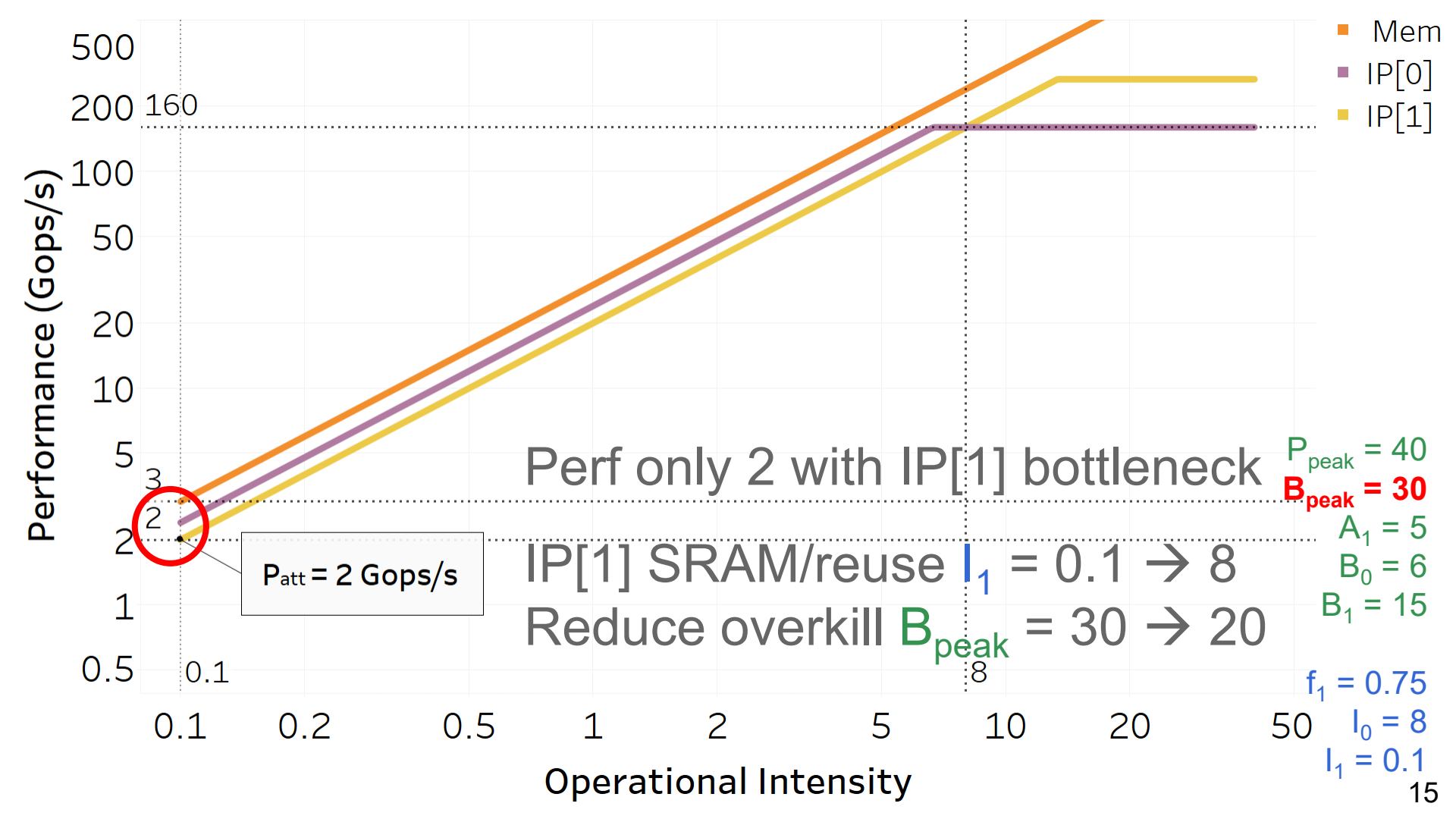
$1 / T_{IP[i]} = MIN(B_i * I_i, A_i * P_{peak}) \;/\; f_i \qquad f_i \neq 0$

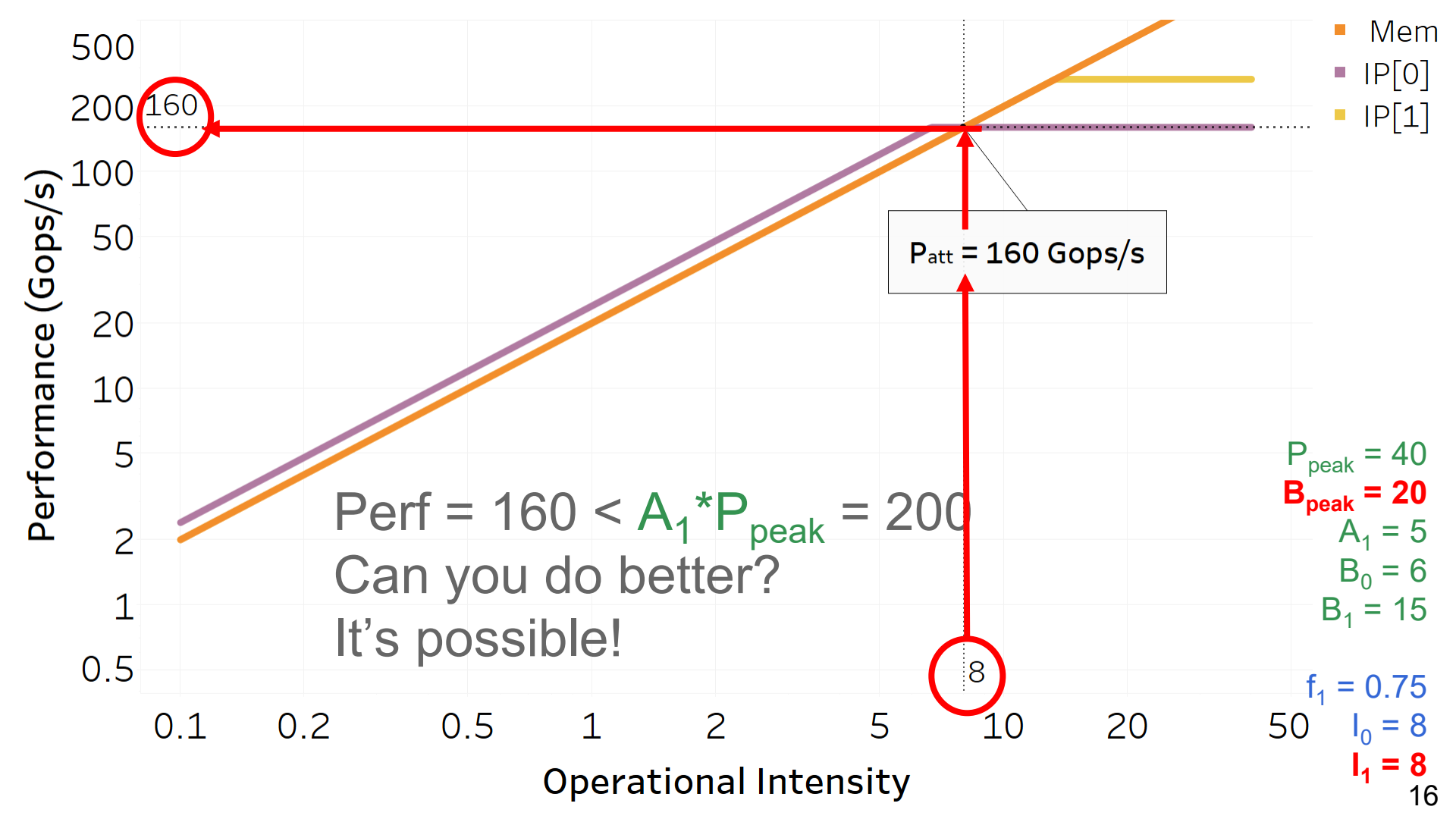
$1 / T_{memory} = B_{peak} * I_{avg} \qquad I_{avg} = 1 / \Sigma_{i=1,N-1}(f_i / I_i)$

$Perf = MIN(1/T_{IP[0]}, \ldots 1/T_{IP[N-1]}, 1/T_{memory})$


Do better?
Assign IP[1] work: f_1 = 0 → 0.75
IP[0]
Mem
Performance (Gops/s)
500
200
100
80
50
40
20
10
5
2
1
0.5
0.1
0.2
0.5
1
2
5
10
20
8
Operational Intensity
Patt = 40 Gops/s
P_{peak} = 40
B_{peak} = 10
A_1 = 5
B_0 = 6
B_1 = 15
f_1 = 0
I_0 = 8
I_1 = 0.1


IP[1] present but Perf drops to 1! Why?
I_1 = 0.1 → memory bottleneck
Enhance B_{peak} = 10 → 30
(at a cost)
IP[0]
IP[1]
Mem
Performance (Gops/s)
500
200
160
100
50
20
10
5
2
2
1
1
0.5
0.1
P_{att} = 1 Gops/s
P_{peak} = 40
B_{peak} = 10
A_1 = 5
B_0 = 6
B_1 = 15
f_1 = 0.75
I_0 = 8
I_1 = 0.1
8
0.1
0.2
0.5
1
2
5
10
20
Operational Intensity


Mem
IP[0]
IP[1]
Performance (Gops/s)
500
200
160
100
50
20
10
5
3
2
2
1
0.5
0.1
0.1
0.2
0.5
1
2
5
10
20
50
8
Operational Intensity
Perf only 2 with IP[1] bottleneck
IP[1] SRAM/reuse I_1 = 0.1 → 8
Reduce overkill B_{peak} = 30 → 20
Patt = 2 Gops/s
P_{peak} = 40
B_{peak} = 30
A_1 = 5
B_0 = 6
B_1 = 15
f_1 = 0.75
I_0 = 8
I_1 = 0.1


Mem
IP[0]
IP[1]
500
200
160
100
50
20
10
5
2
1
0.5
Performance (Gops/s)
0.1
0.2
0.5
1
2
5
10
20
50
Operational Intensity
Patt = 160 Gops/s
Perf = 160 < A1*Ppeak = 200
Can you do better?
It's possible!
8
Ppeak = 40
Bpeak = 20
A1 = 5
B0 = 6
B1 = 15
f1 = 0.75
I0 = 8
I1 = 8

A Gables Workflow for a 1st SoC Answer

	AP	Display	G2DS	GPU	ISP	JPEG	IPU	VDEC	VENC	DSP
HDR+	X	X		X	X	X	X			
Videocapture	X	X		X	X				X	
VideocaptureHDR	X	X		X	X				X	
VideoplaybackUI	X	X	X	X				X		
Google Lens	X	X	X	X						X

For each usecase repeat until sufficiently fast

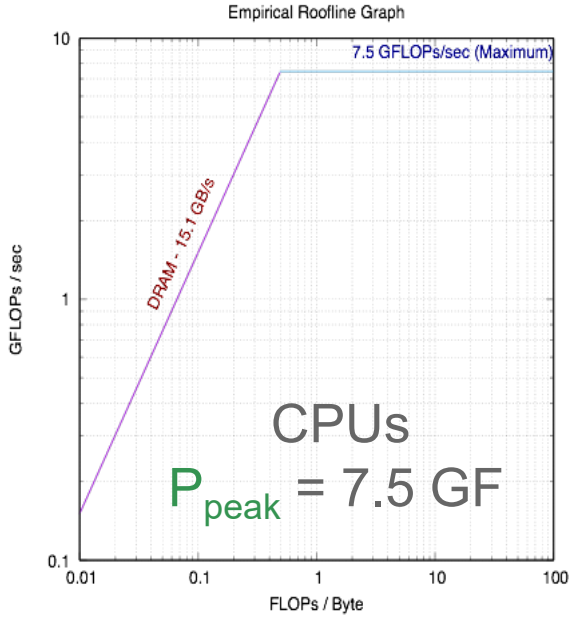
- Pick bottleneck IP[i] improve compute/communication
  Pick non-bottleneck IP[i] reduce cost

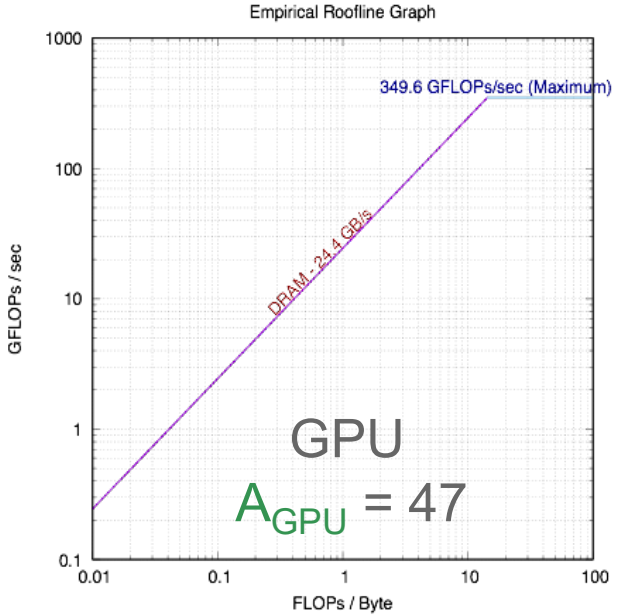
Pick IP[i] configs that satisfy all usecases; done if cost ok

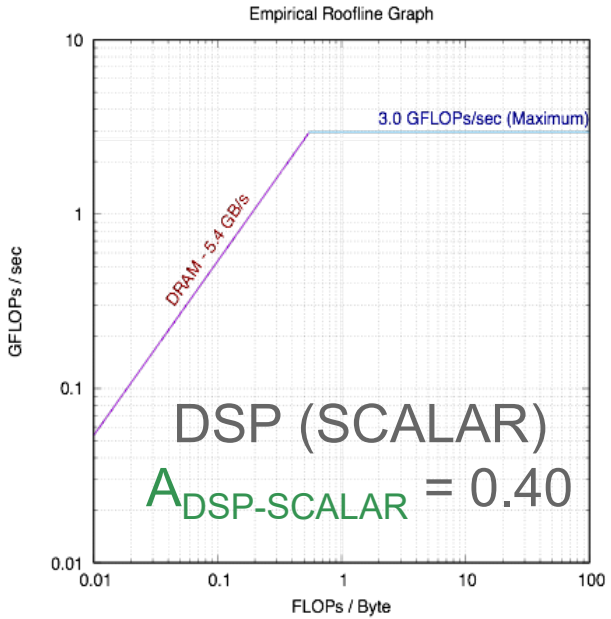
Pixel 2 (Snapdragon 835) w/ Aux. Thermal Mangmt

µBenchmark w/ Qualcomm Snapdragon™ 835

- All elements load from array & vary FP SP op intensity
- Finds empirical lower bound on rooflines







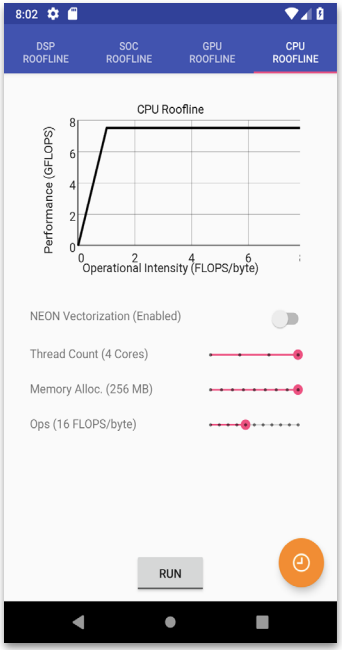
- Preliminary evidence that multiple rooflines useful

Gables Paper & Home Page

Extensions: memory-side buffer, interconnect, serial work

Interactive tool for 2-IP & 3-IP SoCs

Gables Android Source at GitHub



http://research.cs.wisc.edu/multifacet/gables/

Caveats

Base Assumptions

- SW perfectly parallel
- All IP's concurrent w/ each other & memory BW
- BW limits of Roofline appropriate (proxy for power?)

But

- Insight but not cycle-level accuracy
- Omits interrupt latencies, etc., to manage IPs
- IP acceleration varying w/ usecase (Roofline ceiling?)
- <your concern here>

Conjectures

Gables provides a way to conceptualize many-IP SoCs

- Roofline per IP forces early parameter estimation
- Insight for much less work than porting usecases

Operational intensity I_i zeros in on SRAM utility & reuse

Understanding work fraction f_i valuable to estimate the acceleration A_i necessary for each usecase

SoCs harbinger of extreme heterogeneity elsewhere

Executive Summary

All models are wrong, but some are useful.
–George Box, Statistician, 1987

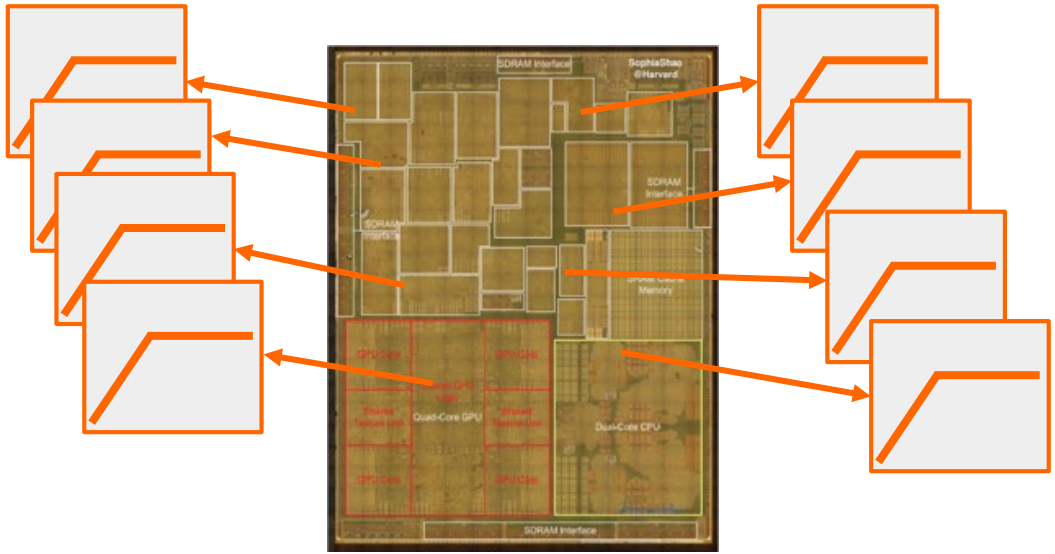
Gables Mobile SoC Model

- Models give initial answers: Amdahl's Law & Roofline
- Gables: Roofline per IP & apportion concurrent work
- E.g., how much IP[i] acceleration needed?

Thanks to Mobile Silicon Team @ Google

Backup Slides

Consumer System on Chip (SoC) & Gables

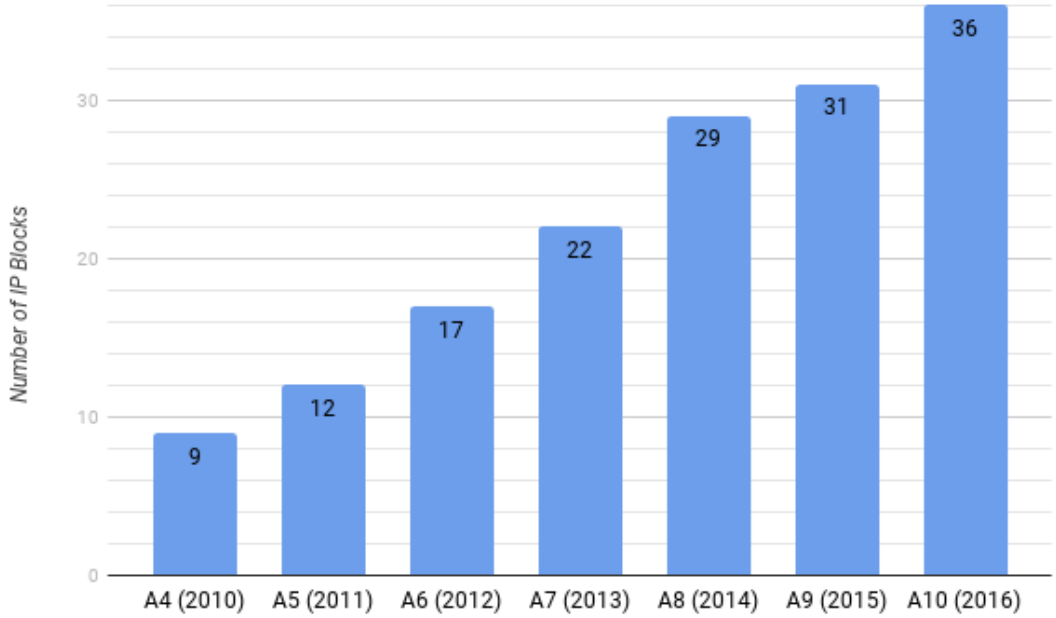


1. Include Accelerator IP[i]? Or give work to enhanced CPUs
2. IP[i] over-provisioned? Make IP[i] acceleration less
3. IP[i] over-communicates? IP[i] less compute; more SRAM

A. Commercial SoCs Hard To Design

Envision usecases
(2-3 years ahead)
Select IPs
Size IPs
Design Uncore

Cycle-level simulation later, but….



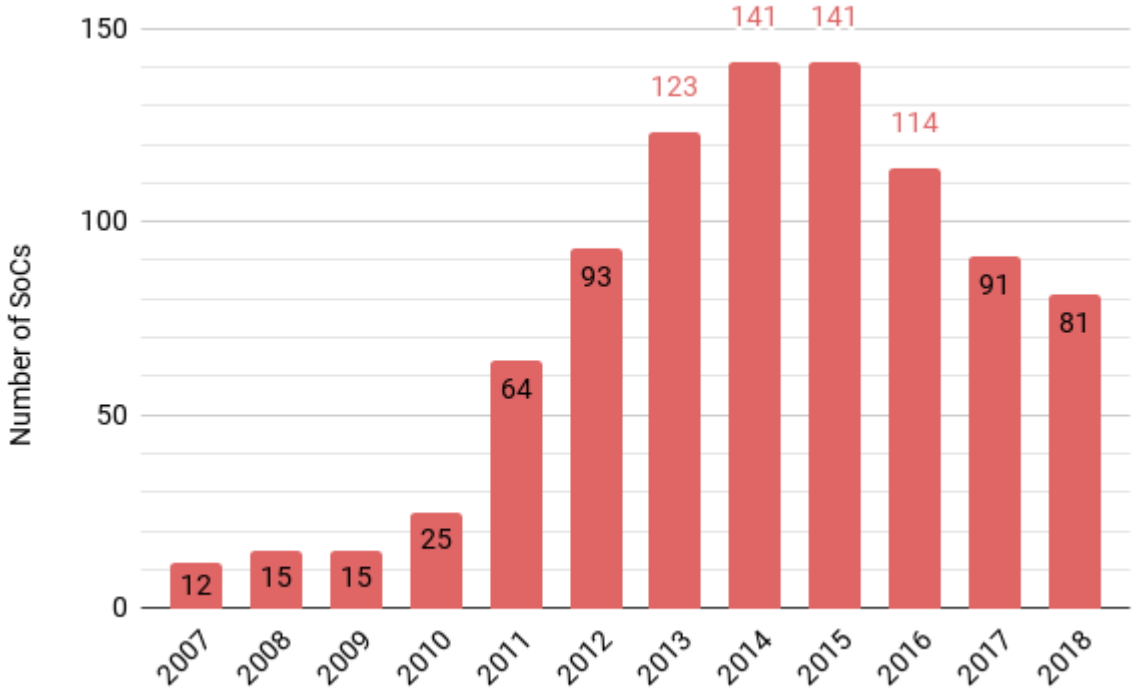
What about early before SW written?

B. Commercial SoCs Hard To Select

Envision usecases (years ahead)
Port to many SoCs??

Port to few finalists?

Early downselect?



SoC MODEL to help early SoC design & selection?

Conjectures

1. Gables is useful for early Mobile SoC planning
2. Valuable to scrutinize each IP's Acceleration & BW
3. Estimating work fraction for "Goldilocks" IP design
4. Operation intensity illuminates IP memory reuse

Operational Intensity (I_i) is Function of IP[i] SRAM (M_i)

Non-linear function that increases when new footprint/working-set fits

Should consider these plots when sizing IP[i] SRAM

Later evaluation can use simulation performance on y-axis

Related Work

Builds on Roofline & Amdahl's Law

Closest: SoC MultiAmdahl [Kelassy et al., CAL'12]
Gables adds BW per-IP & chip & uses concurrent work

Gables can be extended

- CPU-GPU "Valley" [Guz et al., CAL'09]
- LogCA interaction overheads [Altaf & Wood, ISCA'17]
- Richer IP models, e.g., [Jog et al., ISMS'15]

Toward Validating Gables: Methods

Experimental Results w/ Qualcomm Snapdragon™ 835

- Kyro™ 280 CPU, Adreno™ 540 GPU, Hexagon™ 682 DSP

Micro-benchmark: All IP processing elements

- Load 32b word from array (vary memory footprint)
- Do some SP-FP ops (vary operational intensity)

Metric: Roofline Estimate

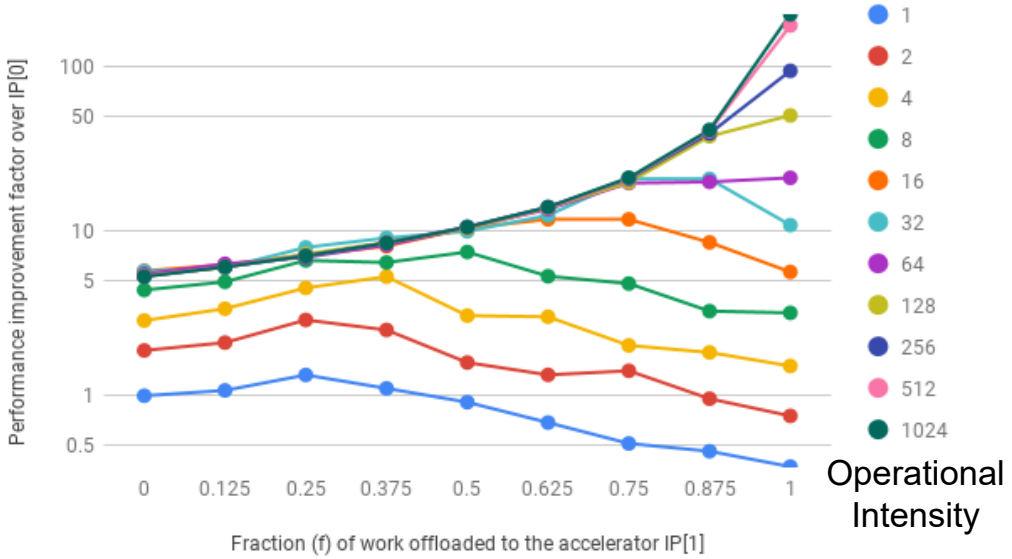
- Op speed × #concurrent ops → upper bound **not used**
- Empirically probe --> lower bound **used**

Toward Validating Gables: Results

CPU IP[0]: P_{peak} = 7.5 GF/s & min(B_0, B_{peak}) = 15.1 GB/s
GPU IP[1]: A_1 = 349.6/7.5 = 46.6 & B_{peak} = 24.4 GB/s

Perf. v. f=0 & I_0=I_1=1

1. Data noisy
2. Diff. work each line
3. Low I: CPU better
4. High I: GPU speedup I=1K, 39X



Gables Glossary

SoC HW Inputs

- P_{peak} & B_{peak} CPU perf. & off-chip BW from Roofline
- A_i & B_i acceleration & BW for each IP[i]

SW Usecase Inputs

- f_i fraction work at each IP[i]
- I_i operational intensity at each IP[i]

Output

- $P_{attainable}$ SoC performance upper bound