Gables: A Roofline Model for Mobile SoCs

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

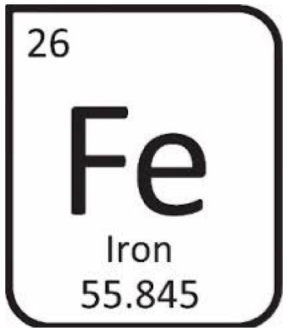
HPCA
Feb 2019

ASPLOS
Tutorial

Gables: A Roofline Model for Mobile SoCs

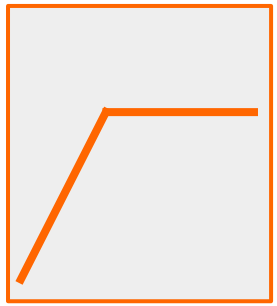
Mark D. Hill, Wisconsin & Former Google Intern
Vijay Janapa Reddi, Harvard & Former Google Intern

Models give insight & first answer



CPU &
Iron Law

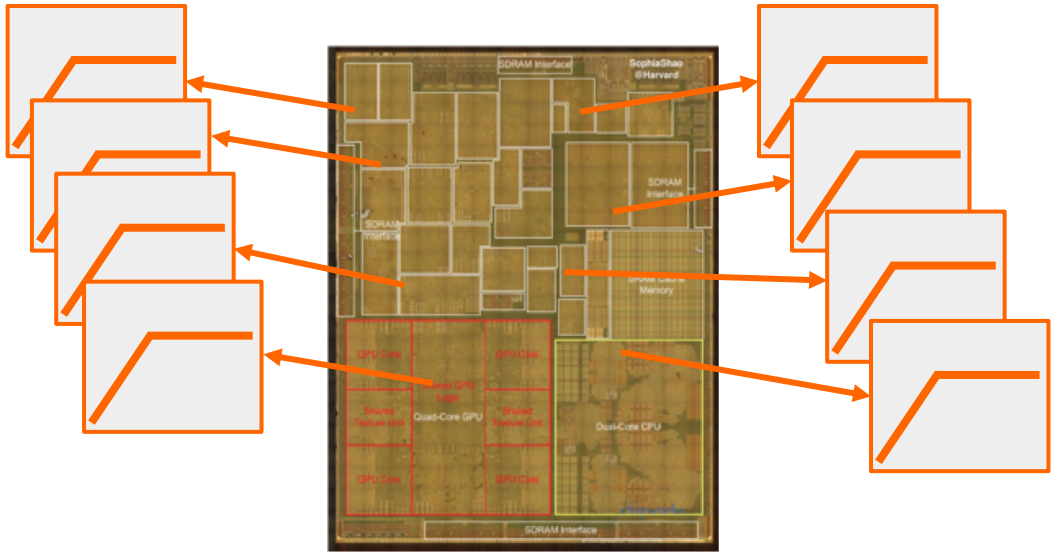
Multiprocessor &
Amdahl's Law



Multicore &
Roofline

Mobile System on a Chip (SoC) & What???

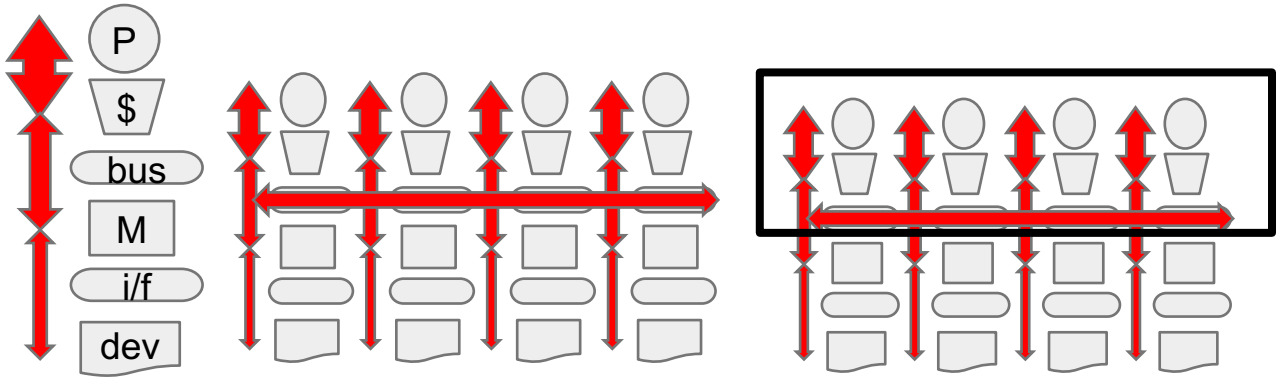
Mobile System on Chip (SoC) & Gables



Dozens of Accelerators. Which?
Provision “Goldilocks” Accelerators?
Whither workload communication?

Gables @ ~5:55pm today!

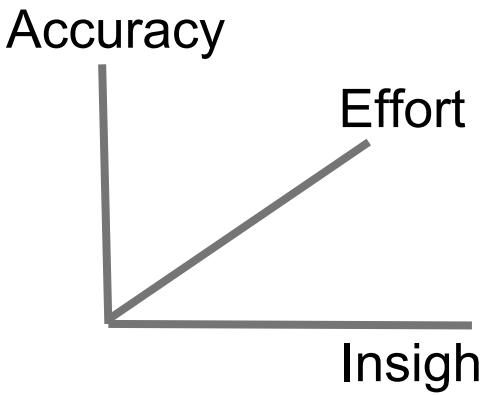
Computer Architecture & Models



CPU &
Iron Law

Multiprocessor &
Amdahl's Law

Multicore &
Roofline

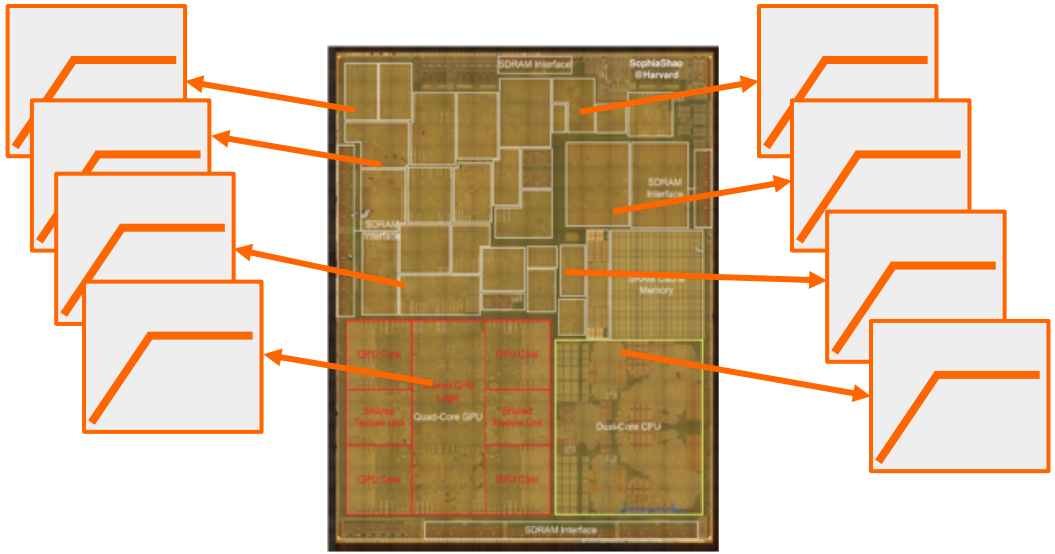


Models v.v. Simulation

- More insight
- Less effort
- But less accuracy

Models give first answer, not final answer

Mobile System on Chip (SoC) & Gables



1. Include Accelerator IP[i]?
2. IP[i] over-provisioned?
3. IP[i] over-communicates?

Gables provides first answer!

Outline

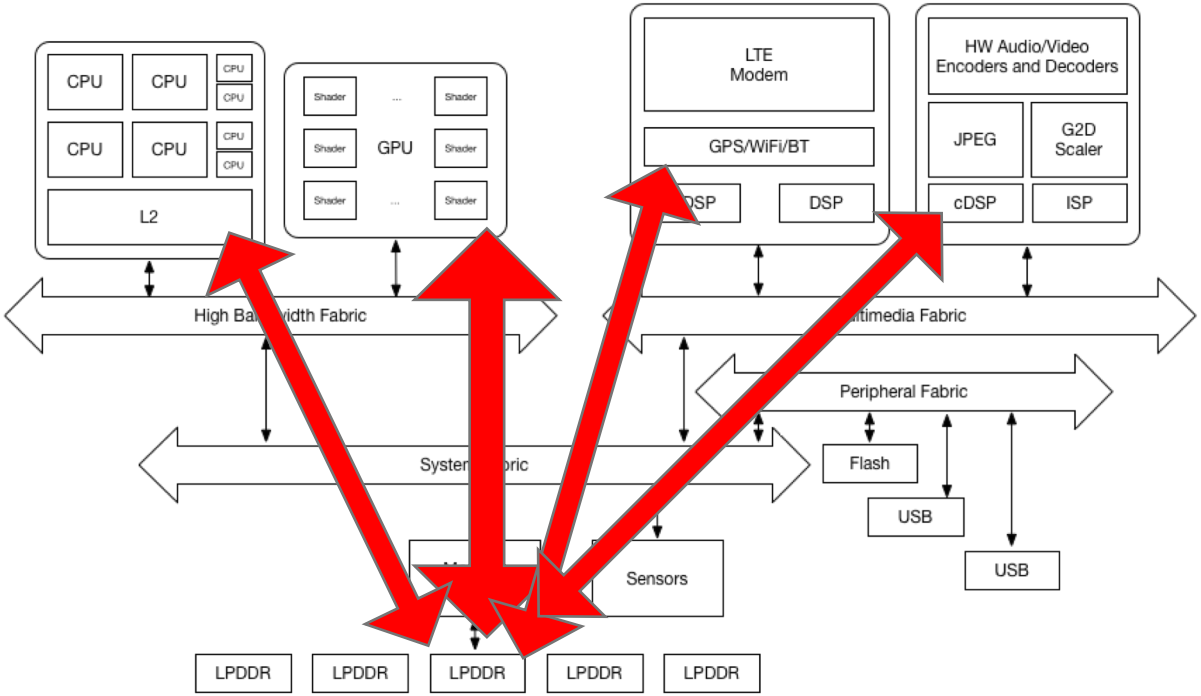
Motivation: Mobile SoCs, Usecases, & IP Blocks

Gables: A Roofline Model for Mobile SoCs

Gables 2-IP Example

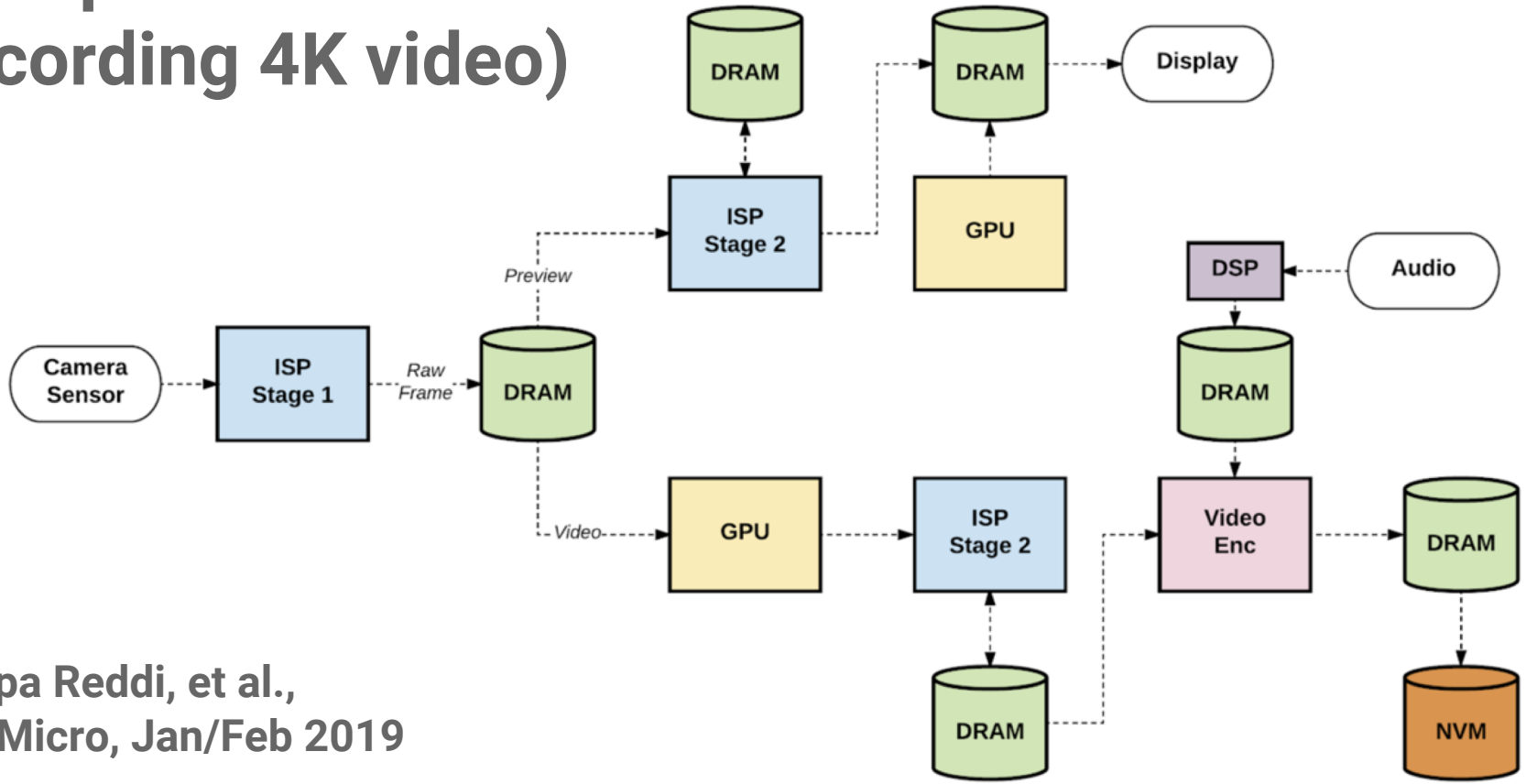
Wrap Up

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

Mobile SoCs Run Usecases

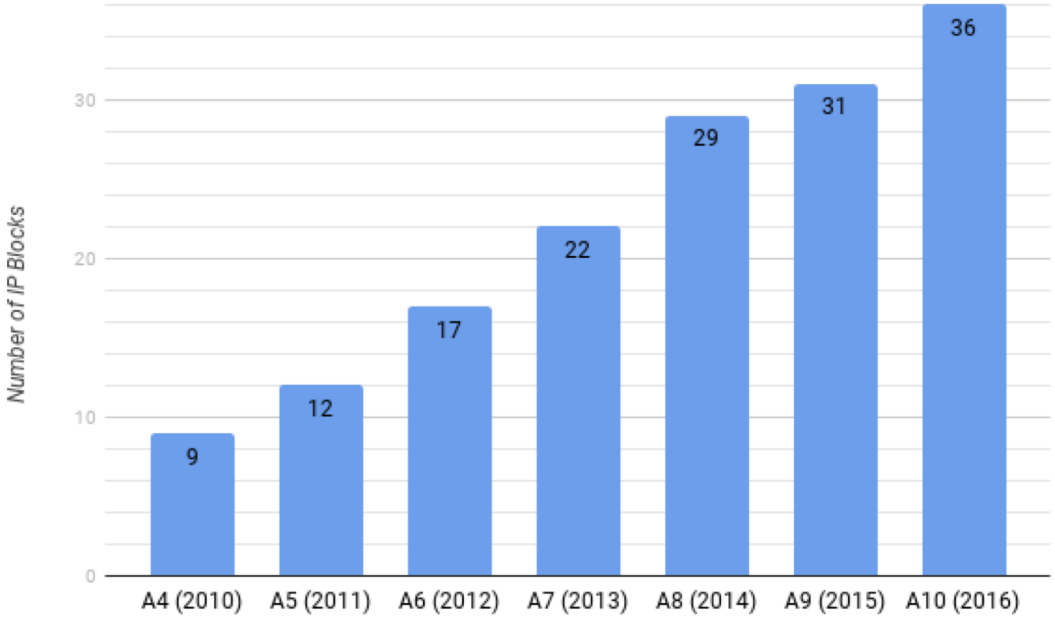
	AP	Display	G2DS	GPU	ISP	JPEG	IPU	VDEC	VENC	DSP
HDR+	X	X		X	X	X	X			
Videocapture	X	X		X	X				X	
VideocaptureHDR	X	X		X	X				X	
VideoplaybackUI	X	X	X	X				X		
Google Lens	X	X	X	X						X

Must run each usecase sufficiently fast -- no need faster
Must run all usecases – average irrelevant
A usecase uses IPs concurrently – more than serially

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?

Outline

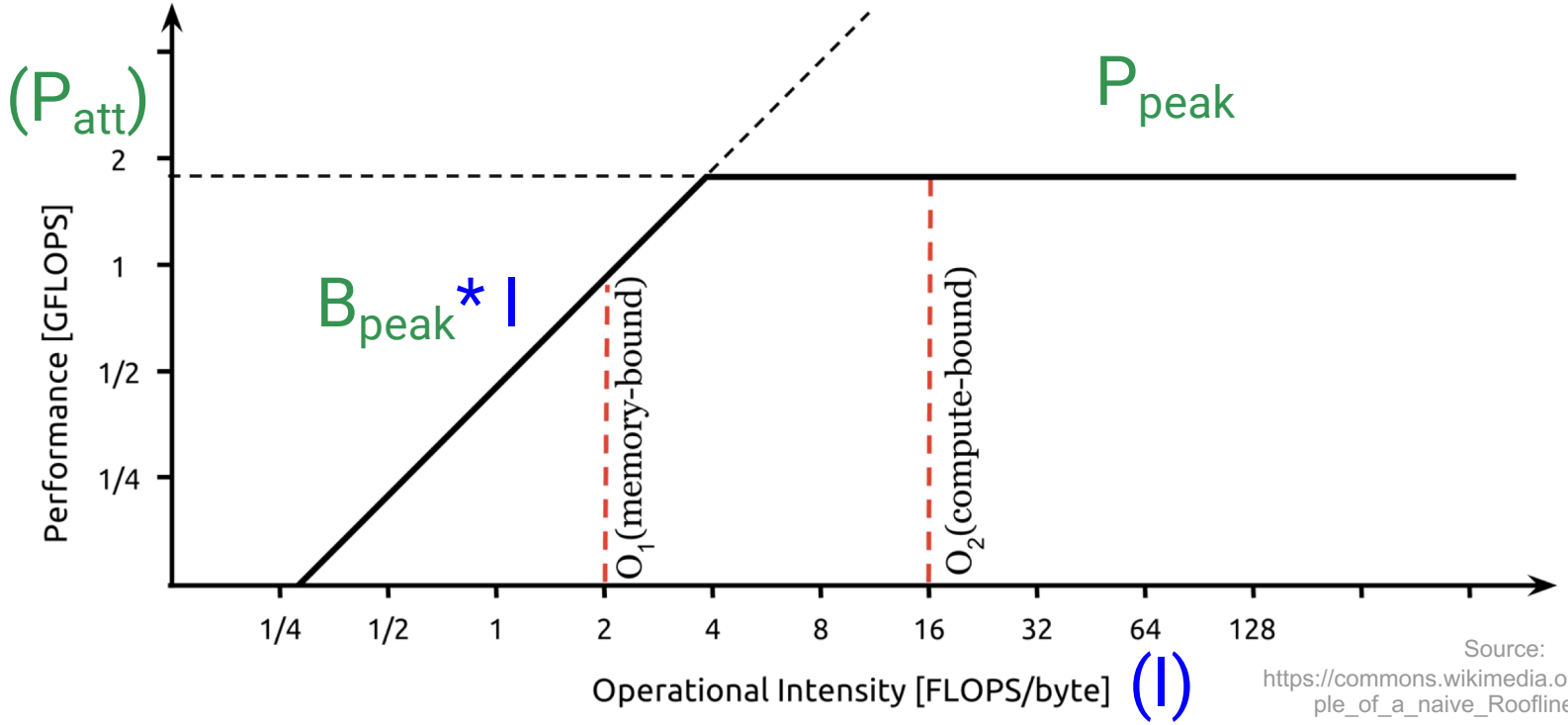
Motivation: Mobile SoCs, Usecases, & IP Blocks

Gables: A Roofline Model for Mobile SoCs

Gables 2-IP Example

Wrap Up

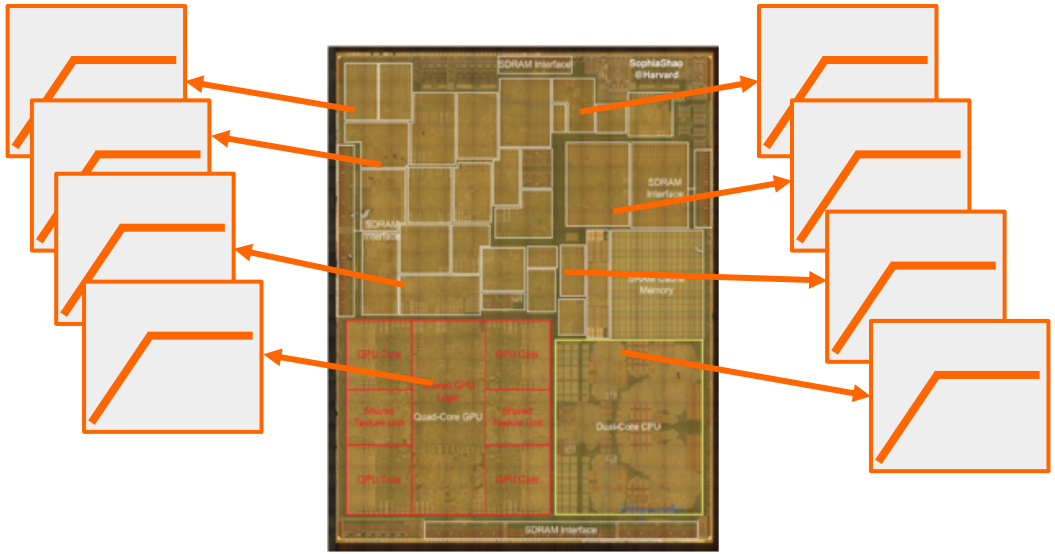
Williams et al., Roofline, CACM 4/2009



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

$$P_{att} = MIN(B_{peak} * I, P_{peak})$$

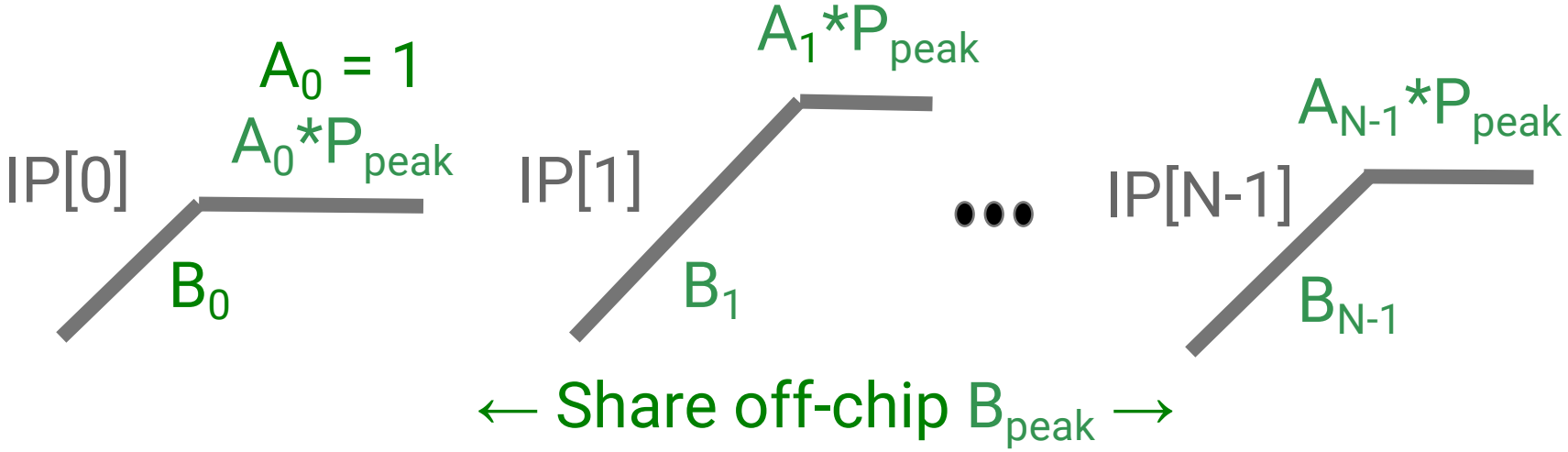
Consumer System on Chip (SoC) & Gables



1. Include Accelerator IP[i]?
2. IP[i] over-provisioned?
3. IP[i] over-communicates?

Gables provides first answer!

Gables for N IP SoC



Usecase at each each IP[i]

- Parallel non-negative work f_i (f_i's sum to 1)
- Operational intensity I_i operations/byte

Gables Math: Roofline / Work Fraction

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

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

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

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

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

Outline

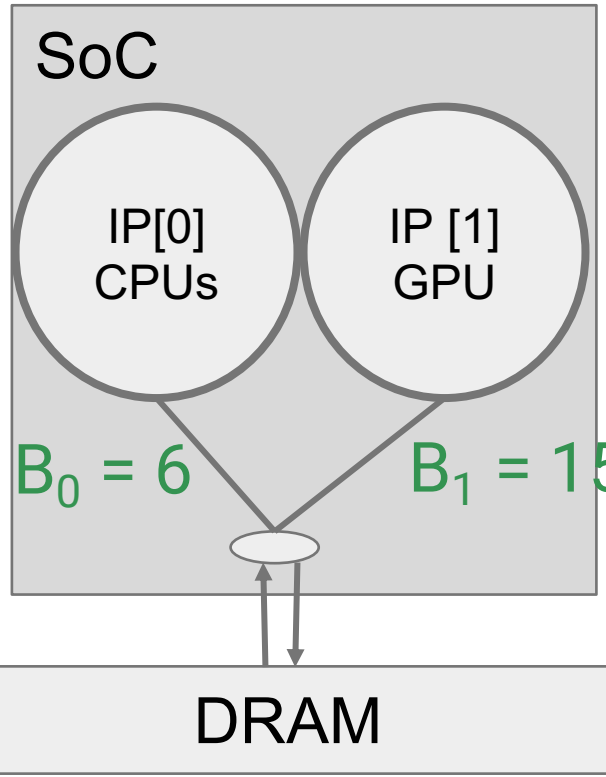
Motivation: Mobile SoCs, Usecases, & IP Blocks

Gables: A Roofline Model for Mobile SoCs

Gables 2-IP Example

Wrap Up

Gables Example with 2 IP SoC System



P_{peak} = 40

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

B_0 = 6

B_1 = 15

B_{peak} = 10

Workload:

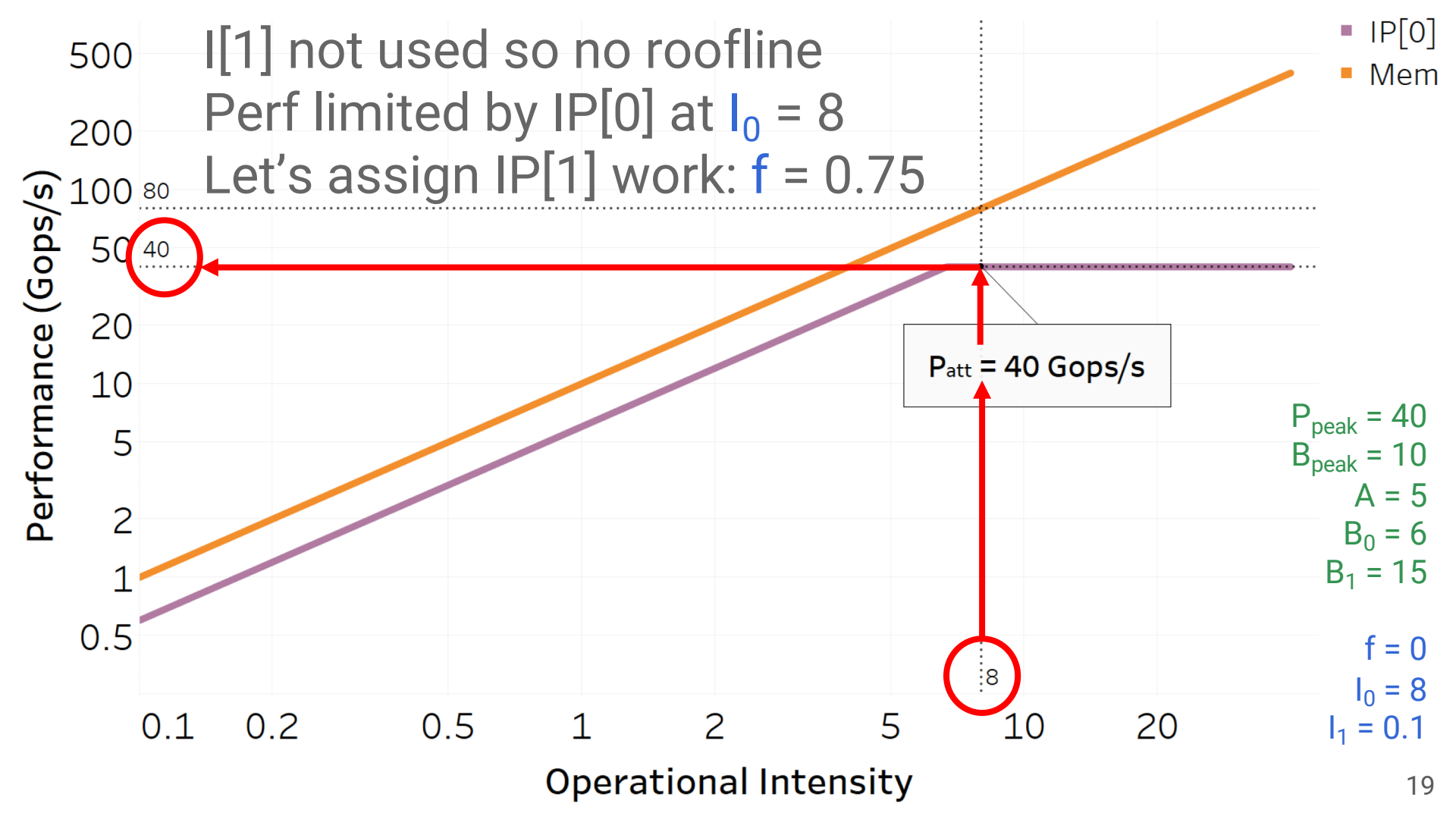
f = 0

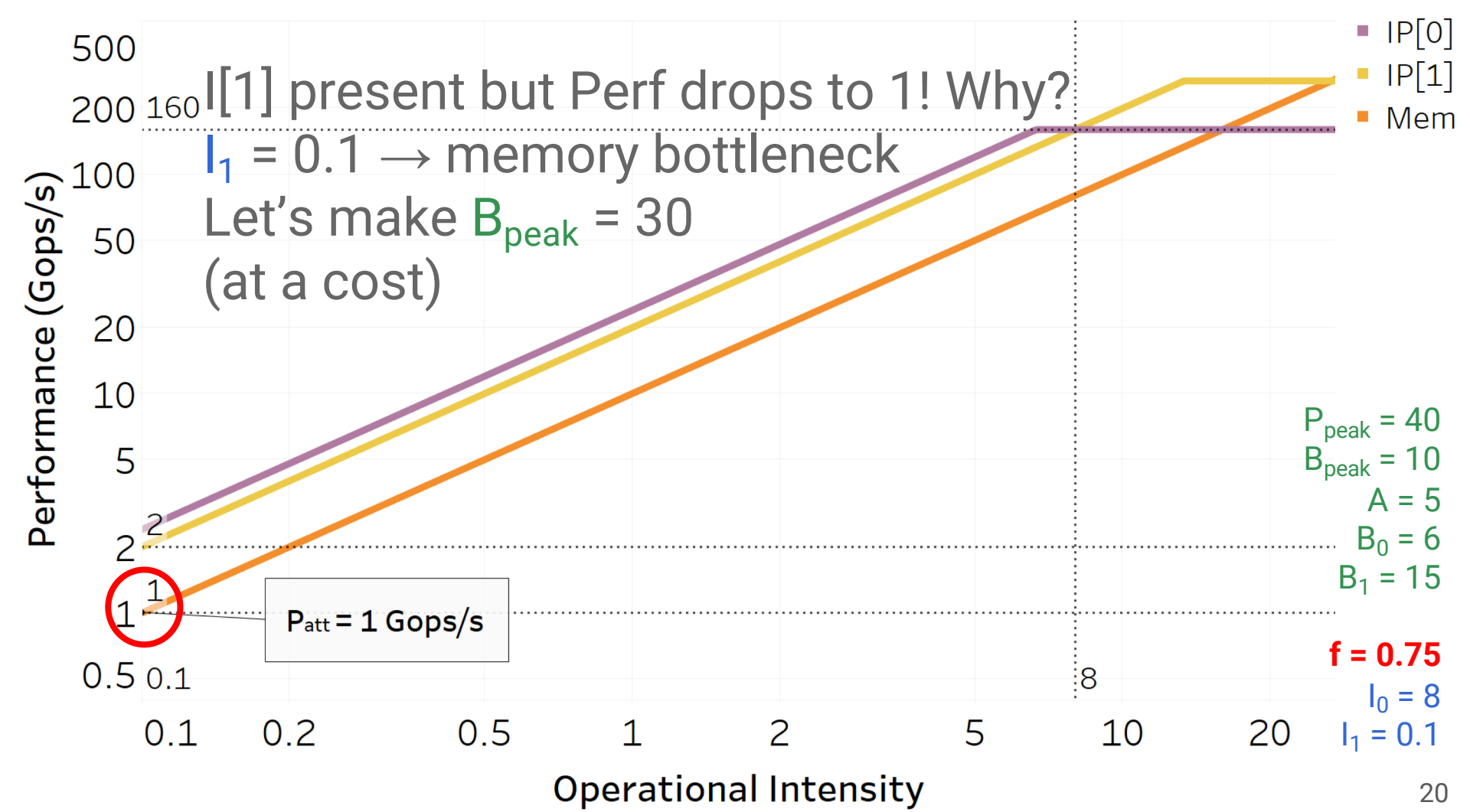
I_0 = 8 = good caching

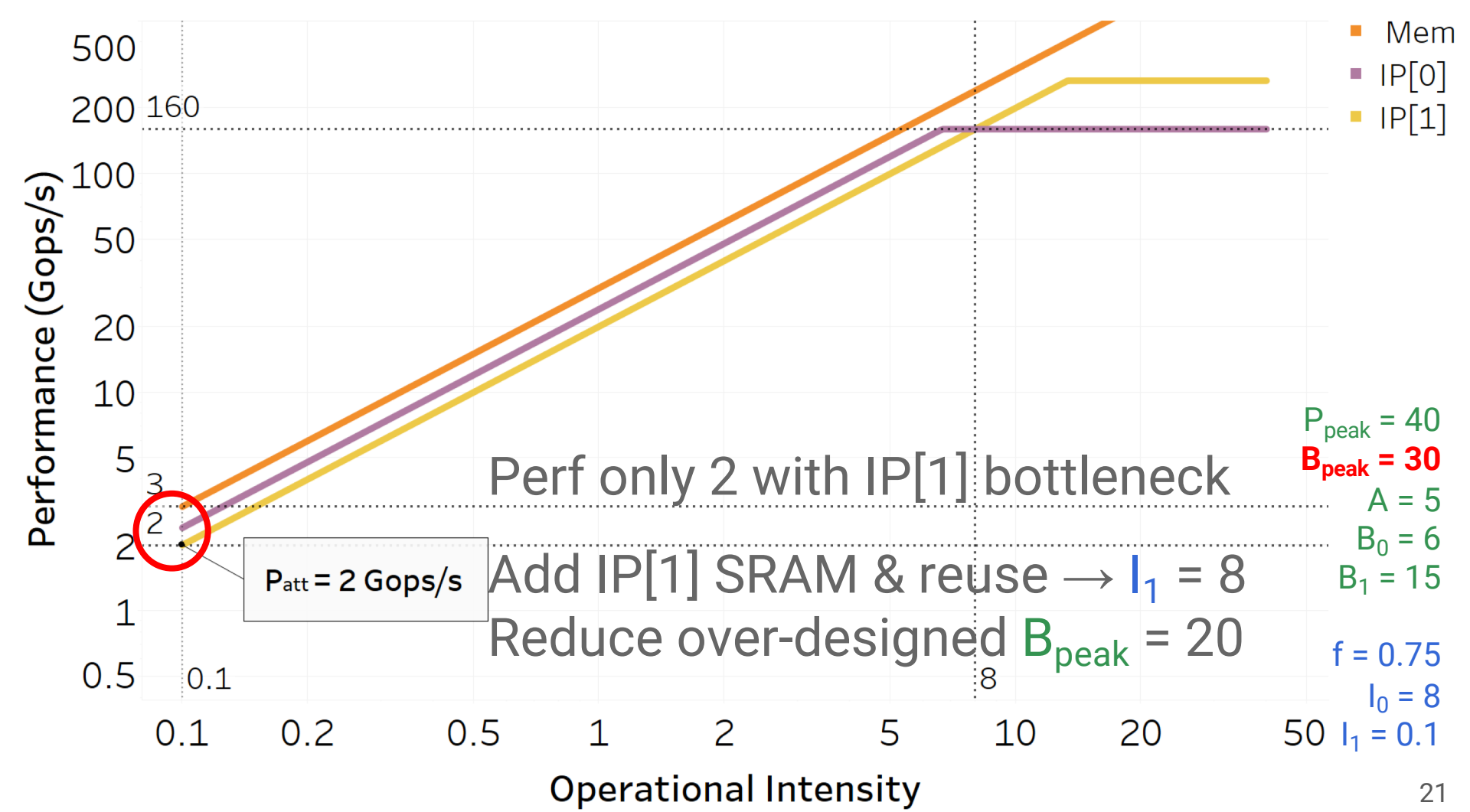
I_1 = 0.1 = latency tolerant

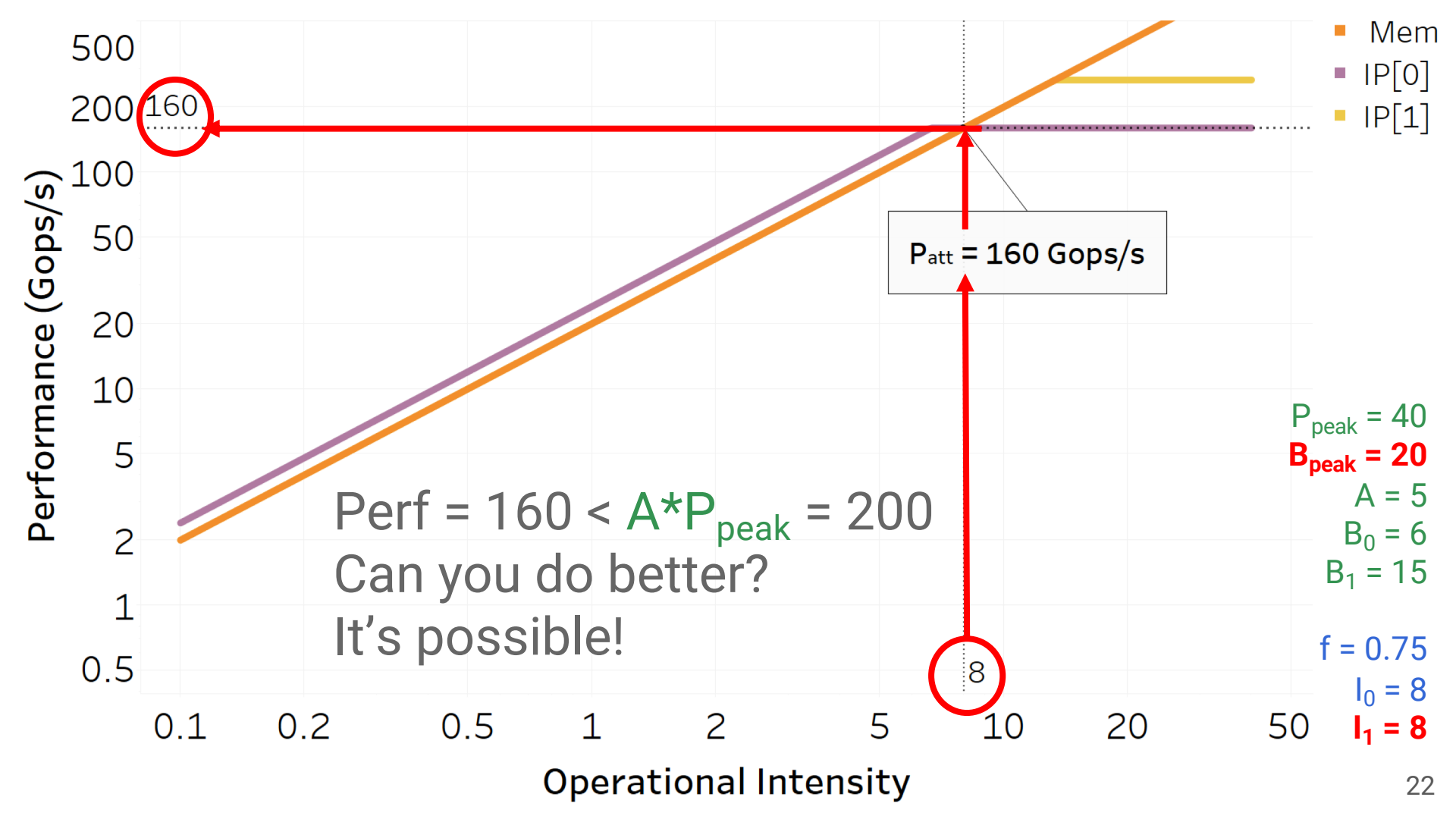
Performance?

Go to research.cs.wisc.edu/multifacet/gables/

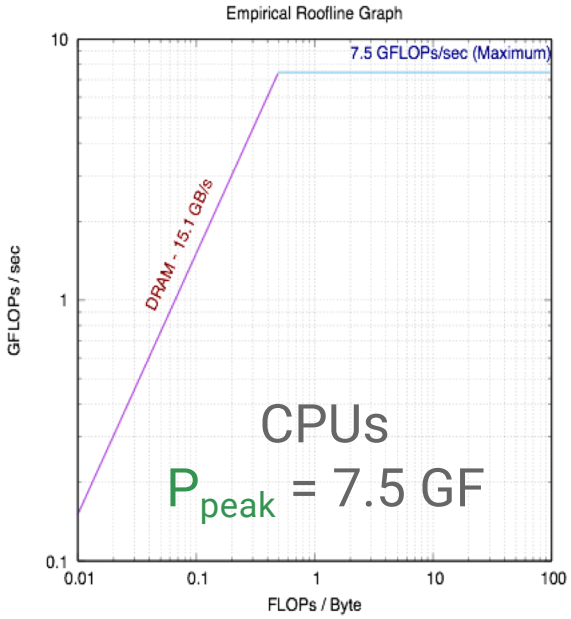

I[1] not used so no roofline
Perf limited by IP[0] at I_0 = 8
Let's assign IP[1] work: f = 0.75
IP[0]
Mem
Performance (Gops/s)
500
200
100
80
50
40
20
10
5
2
1
0.5
0.1
0.2
0.5
1
2
5
10
20
8
Operational Intensity
Patt = 40 Gops/s
P_{peak} = 40
B_{peak} = 10
A = 5
B_0 = 6
B_1 = 15
f = 0
I_0 = 8
I_1 = 0.1


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


Mem
IP[0]
IP[1]
Performance (Gops/s)
500
200
160
100
50
20
10
5
3
2
2
1
0.5
0.1
8
0.1
0.2
0.5
1
2
5
10
20
50
Operational Intensity
Patt = 2 Gops/s
Perf only 2 with IP[1] bottleneck
Add IP[1] SRAM & reuse → I1 = 8
Reduce over-designed Bpeak = 20
Ppeak = 40
Bpeak = 30
A = 5
B0 = 6
B1 = 15
f = 0.75
I0 = 8
I1 = 0.1

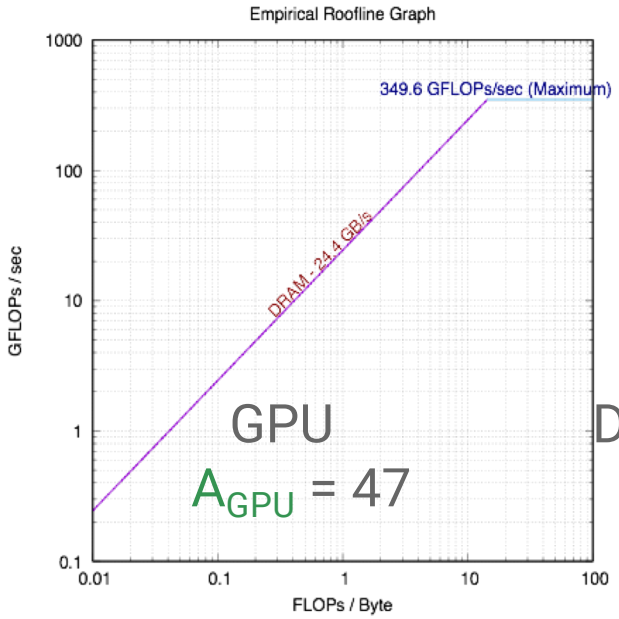

Mem
IP[0]
IP[1]
Performance (Gops/s)
500
200
160
100
50
20
10
5
2
1
0.5
0.1
0.2
0.5
1
2
5
10
20
50
Operational Intensity
Patt = 160 Gops/s
Perf = 160 < A*Ppeak = 200
Can you do better?
It's possible!
8
Ppeak = 40
Bpeak = 20
A = 5
B0 = 6
B1 = 15
f = 0.75
I0 = 8
I1 = 8

µBenchmark w/ Qualcomm Snapdragon™ 835



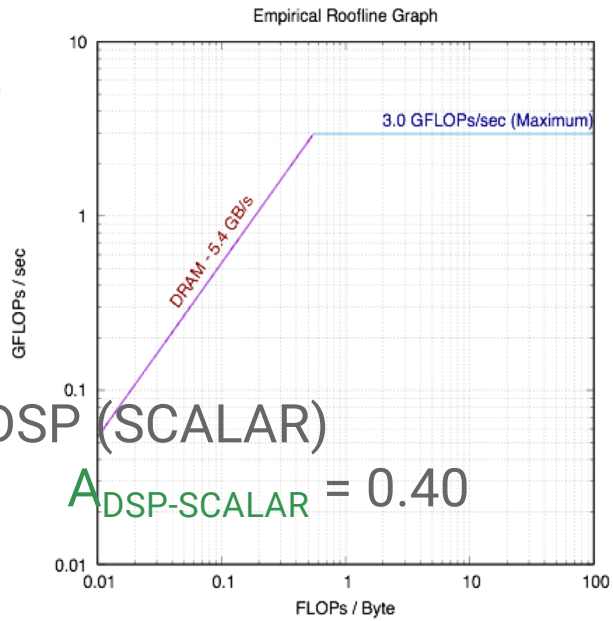
CPUs

P_{peak} = 7.5 GF



GPU

A_{GPU} = 47



DSP (SCALAR)

$A_{DSP\text{-}SCALAR}$ = 0.40

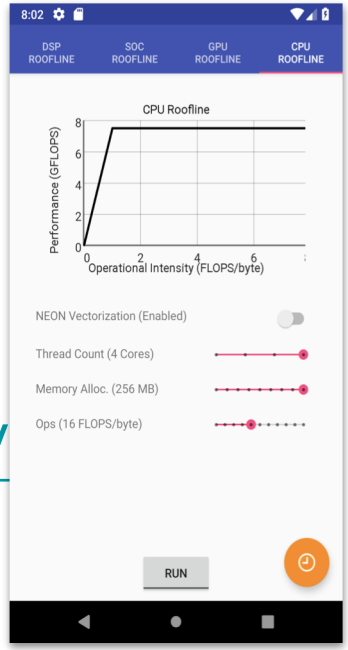
- All elements load from array & vary FP SP op intensity
- Finds empirical lower bound on rooflines

Gables Paper & Home Page

Extensions: memory-side buffer, interconnect, serial work

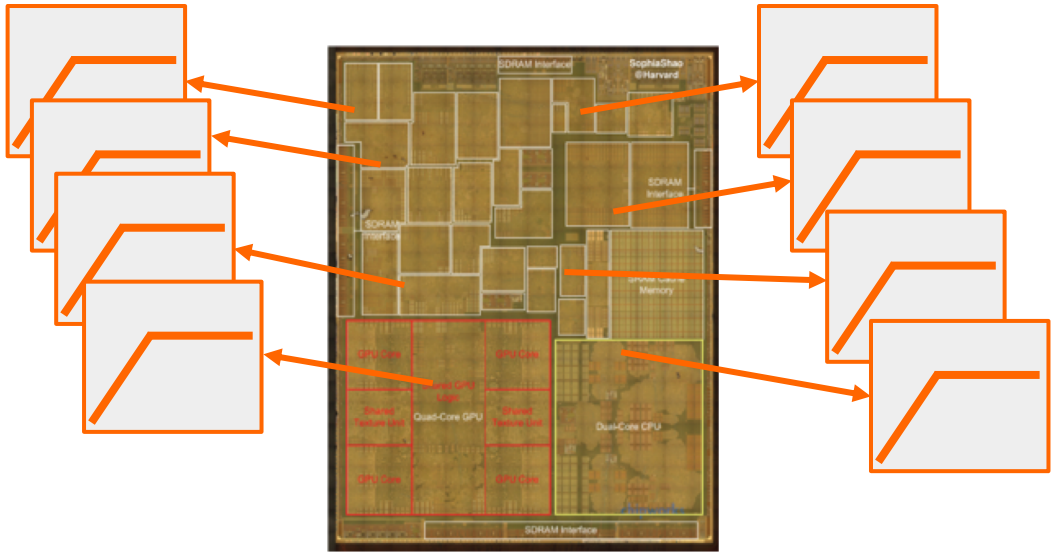
Interactive tool for 2-IP & 3-IP SoCs

Gables: Open-source Android App



http://research.cs.wisc.edu/multifacet/gables/
https://github.com/harvard-edge/Gables

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

Conjectures

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