Lecture

5

ECE216B

Architecture Flexibility

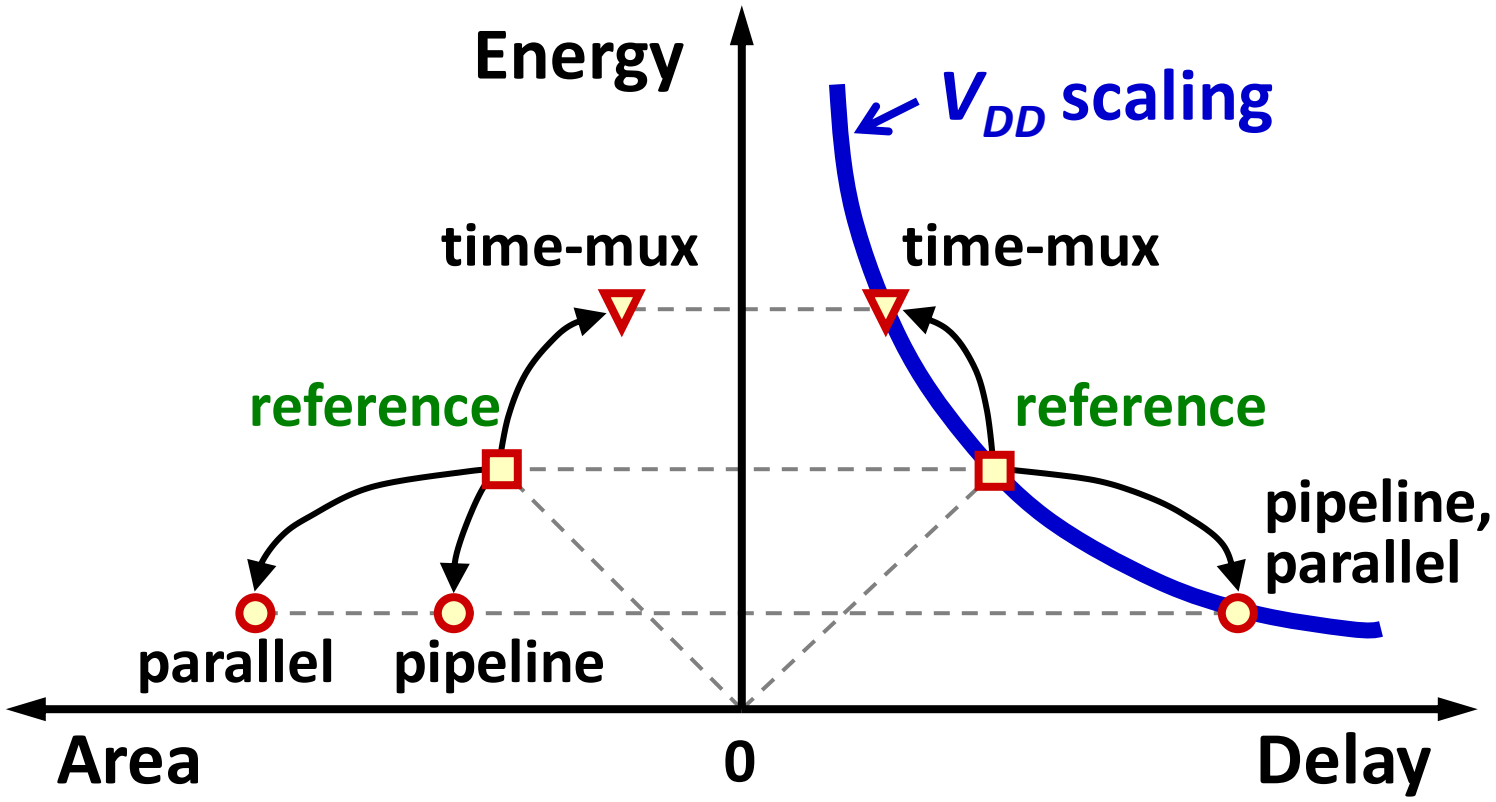
Prof. Dejan Marković

ee216b@gmail.com

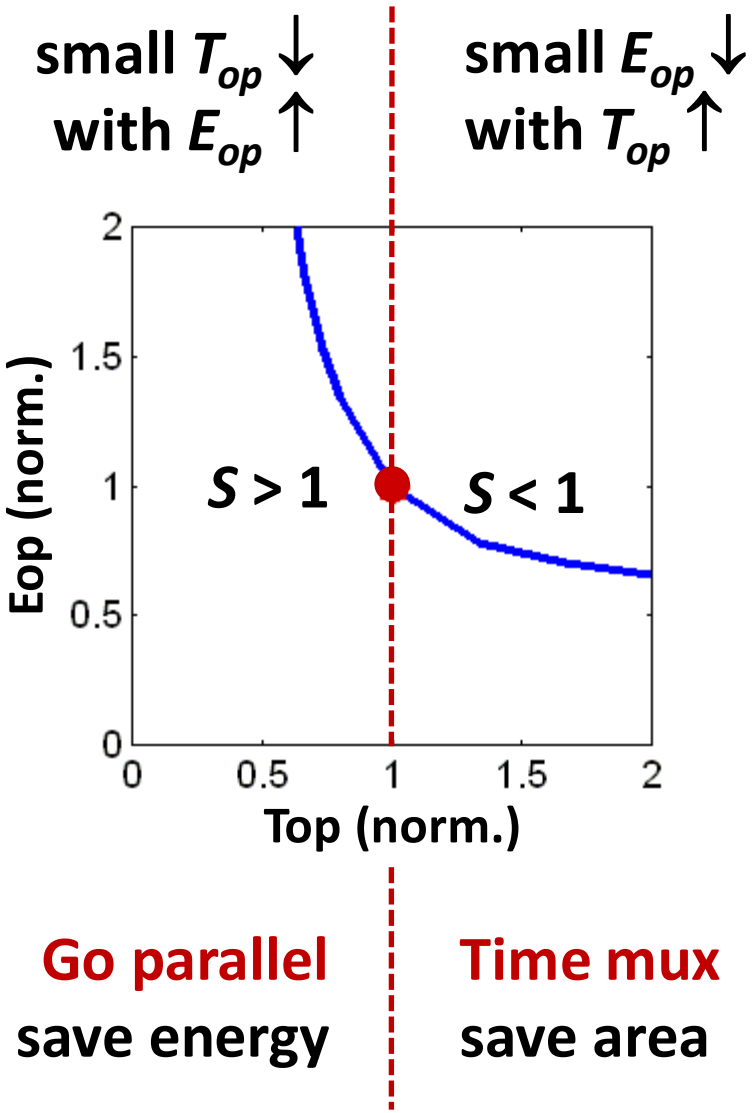
Parallelism vs. Time Multiplexing

? Energy efficiency

- **Parallelism:** energy efficient
- **Time multiplexing:** energy inefficient



Recap: Sensitivity Analysis



Agenda

- **Understand hardware efficiency metrics**
  - Energy efficiency
  - Area efficiency
- **Architectural case study**
  - CPU, GPU, DSP, ASIC
- **System examples**

Architecture Flexibility

- **Determining how much to include and how to do it in the most efficient way possible**
- **Claims (to be shown)**
  - There are good reasons for flexibility
  - The "cost" of flexibility is orders of magnitude of inefficiency over an optimized solution
  - There are different ways to provide flexibility

Material based on ISSCC 2002 evening session lecture (updated to include most recent chips):
R.W. Brodersen, "Technology, Architecture, and Applications," in *Proc. Int. Solid-State Circuits Conf.,* Special Topic Evening Session: Low Voltage Design for Portable Systems, Feb. 2002.

Good Reasons for Flexibility

- **One design for a number of SoC customers: more sales volume**
- **Customers able to provide added value and uniqueness**
- **Unsure of specification or can't make a decision**
- **Backwards compatibility with (debugged) software**
- **Risk, cost and time of implementing hardwired solutions**

Important to note:

these are business, not technical reasons

So, What is the Cost of Flexibility?

- **We need technical metrics that we can use to compare flexible and non-flexible implementations**
  - A power metric because of thermal limitations
  - An energy metric for portable operation
  - A cost metric related to the area of the chip
  - Performance (computational throughput)

Let's use metrics normalized to the amount of computation being performed – so now let's define computation

Definitions

Computation

- Operation = OP =algorithmically interesting computation (i.e. multiply, add, delay)
- MOPS = Millions of OP's per Second
- N_{op} = Number of parallel OP's in ***each*** clock cycle

Power

- P_{chip} = Total power of chip = $A_{chip} \cdot C_{sw} \cdot V_{DD}^2 \cdot f_{clk}$
- C_{sw} = Switched Cap / mm^2 = $P_{chip} / (A_{chip} \cdot V_{DD}^2 \cdot f_{clk})$

Area

- A_{chip} = Total area of chip
- A_{op} = Average area of each operation = A_{chip} / N_{op}

Energy Efficiency Metric: MOPS/mW

- **How much computing (number of operations) can we can do with a finite energy source (e.g. battery)?**

$$\text{Energy efficiency} = \frac{\text{Number of useful operations}}{\text{Energy required}}$$

$$= \frac{\text{Number of operations}}{\text{NanoJoule}} = \frac{\text{OP}}{\text{nJ}}$$

$$= \frac{\text{OP/sec}}{\text{nJ/sec}} = \frac{\text{MOPS}}{\text{mW}}$$

$$= \text{Power efficiency}$$

⇨ **Energy efficiency = Power efficiency**

Energy and Power Efficiency

OP/nJ = MOPS/mW

- **Interestingly, the energy efficiency metric for energy constrained applications (OP/nJ) for a fixed number of operations, is the same as that for thermal (power) considerations when maximizing throughput (MOPS/mW)**

- **So let's look at a number of chips to see how these efficiency numbers compare**

Chip Archeology:
Architecture Case Studies

ISSCC Chips (22nm – 0.18µm)

Chip	Year	Paper	Description
1	2009	3.8	Dunnington
2	2010	5.7	MSG-Passing
3	2010	5.5	Wire-speed
4	2011	4.4	Godson-3B
5	2013	3.5	Godson-3B1500
6	2011	15.1	Sandy Bridge
7	2012	3.1	Ivy Bridge
8	2011	15.4	Zacate
9	2013	9.4	ARM-v7A

Chip	Year	Paper	Description
10	2012	10.6	3D Proc.
11	2013	9.3	H.264
12	2012	28.8	Razor SIMD
13	2011	7.1	3DTV
14	2011	7.3	Multimedia
15	2011	19.1	ECG/EEG
16	2010	18.4	Obj. Recog.
17	2012	12.4	Obj. Recog.
18	2013	9.8	Obj. Recog.
19	2011	7.4	Neural Network
20	2013	28.2	Visual. Recog.

Chip type:

Microprocessor

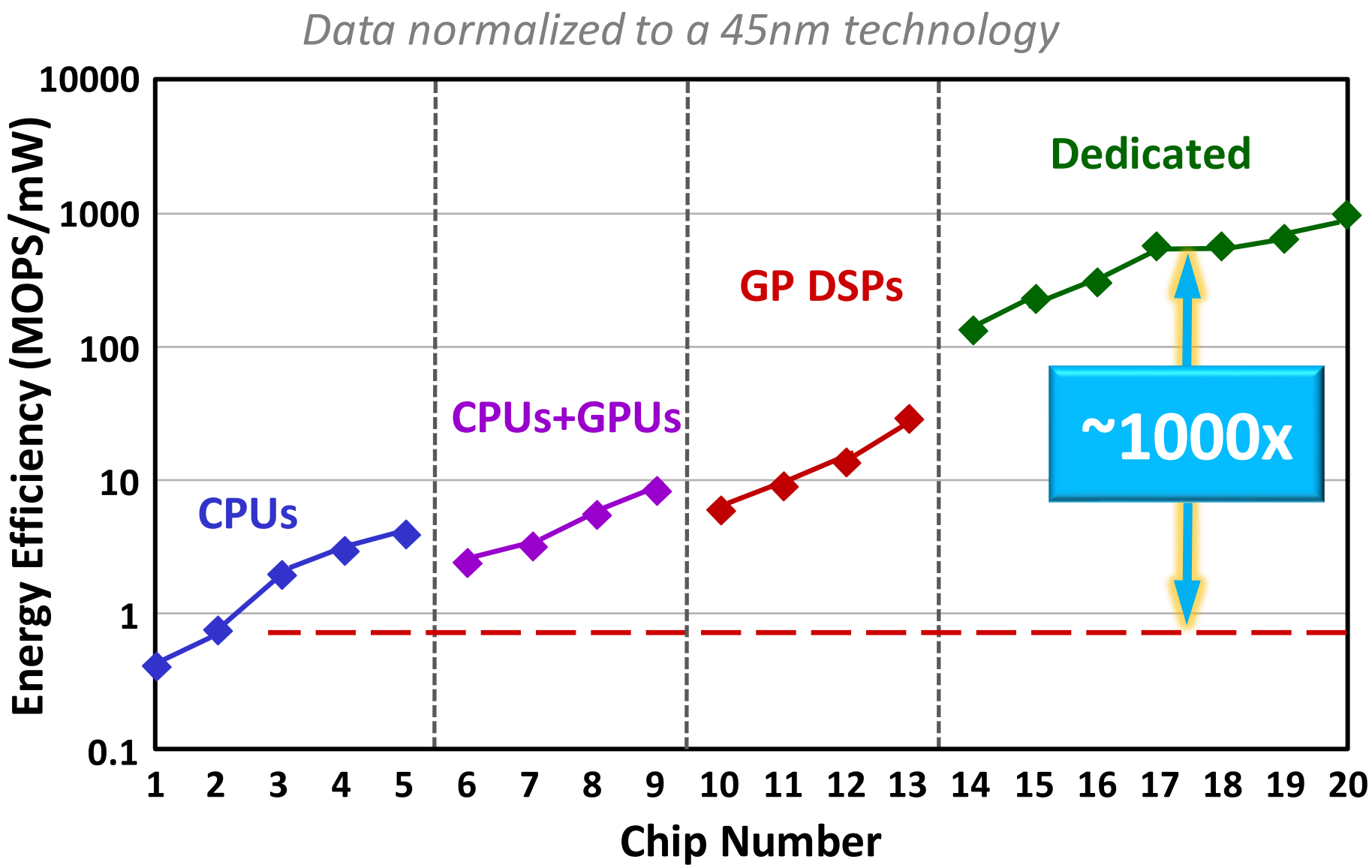
Microprocessor + GPU

General purpose DSP

Dedicated design

Chips published at ISSCC over a 5-year span

Energy Efficiency (MOPS/mW or OP/nJ)



Why Such a Big Difference?

Lets look at the components of MOPS/mW

- **The operations per second:**

  $$\text{MOPS} = f_{clk} \cdot N_{op}$$

- **The power:**

  $$P_{chip} = A_{chip} \cdot C_{sw} \cdot V_{DD}^2 \cdot f_{clk}$$

- **The ratio (MOPS / P_{chip}) gives the MOPS/mW**

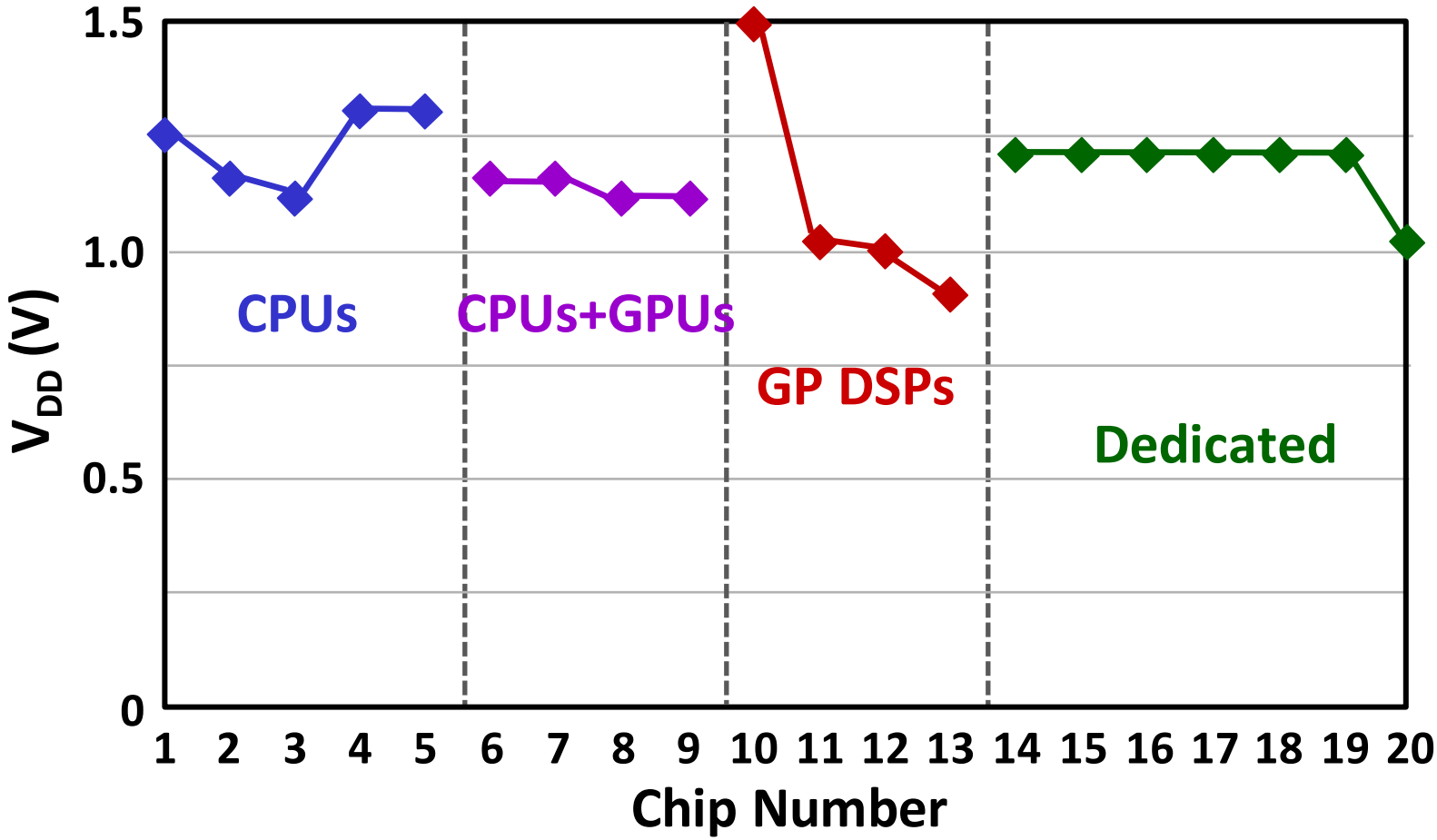
  $$= (f_{clk} \cdot N_{op}) / (A_{chip} \cdot C_{sw} \cdot V_{DD}^2 \cdot f_{clk})$$

Simplifying, $\text{MOPS/mW} = 1 / (A_{op} \cdot C_{sw} \cdot V_{DD}^2)$

So lets look at the 3 components: V_{DD}, C_{sw} and A_{op}

Supply Voltage, V_{DD}

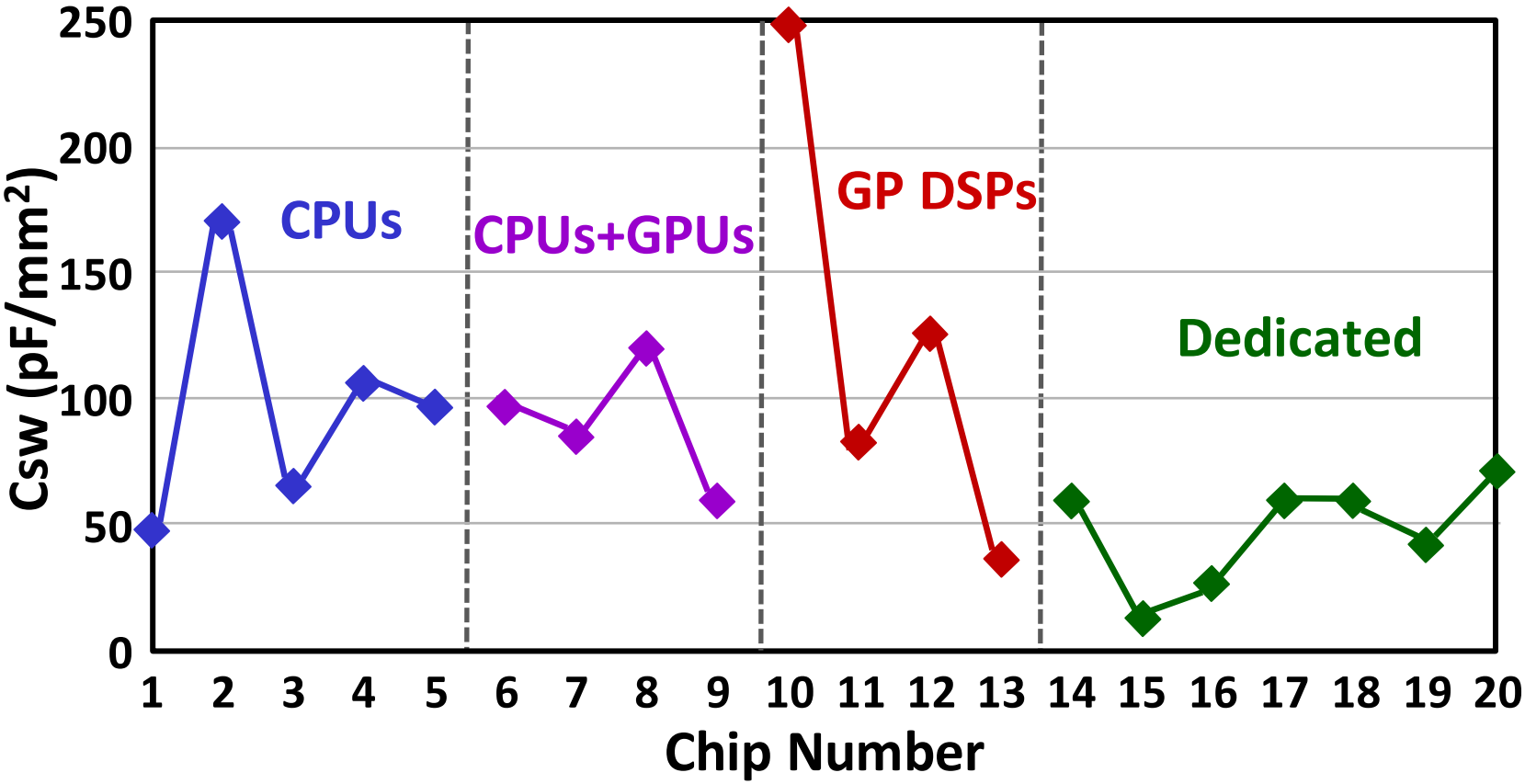
$$\text{MOPS/mW} = 1 / (A_{op} \cdot C_{sw} \cdot V_{DD}^2)$$



V_{DD} isn't the cause (actually a bit higher for dedicated)

Switched Capacitance, C_{sw} (pF/mm^2)

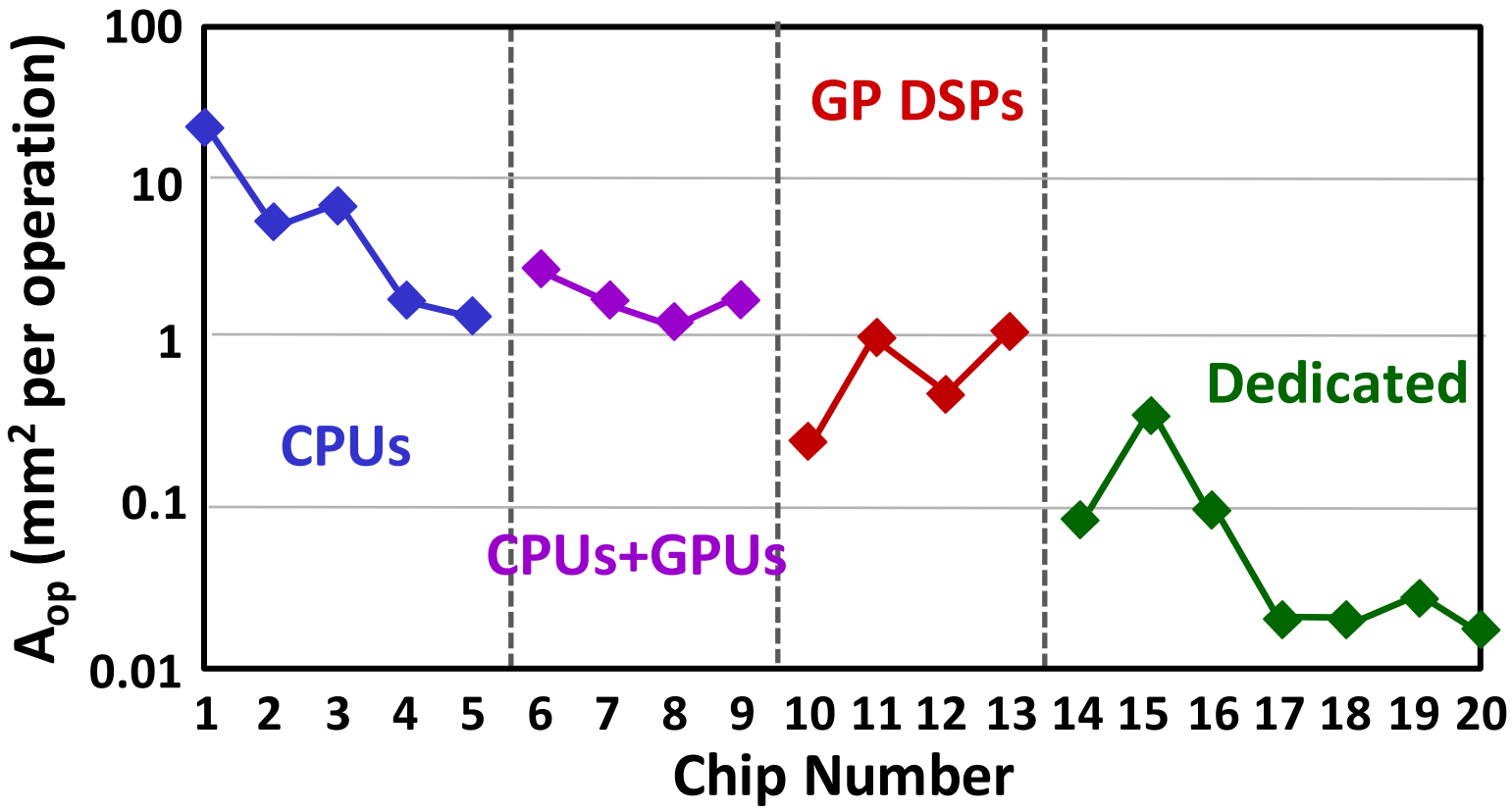
$$\text{MOPS/mW} = 1 / (A_{op} \cdot C_{sw} \cdot V_{DD}^2)$$



C_{sw} is lower for dedicated, but only by a factor of 2-3

A_{op} = Area per Operation (A_{chip}/N_{op})

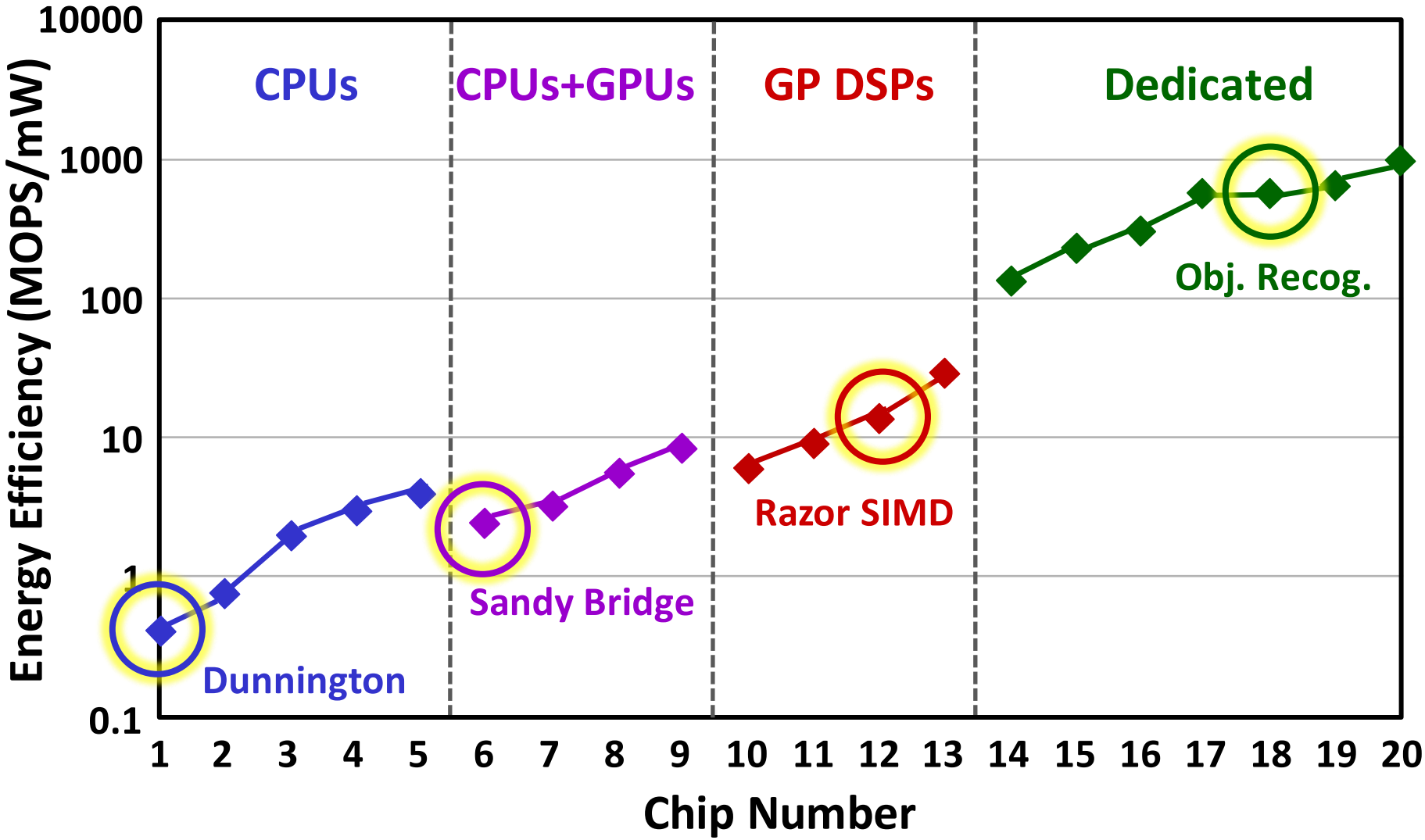
$$\text{MOPS/mW} = 1 / (A_{op} \cdot C_{sw} \cdot V_{DD}^2)$$



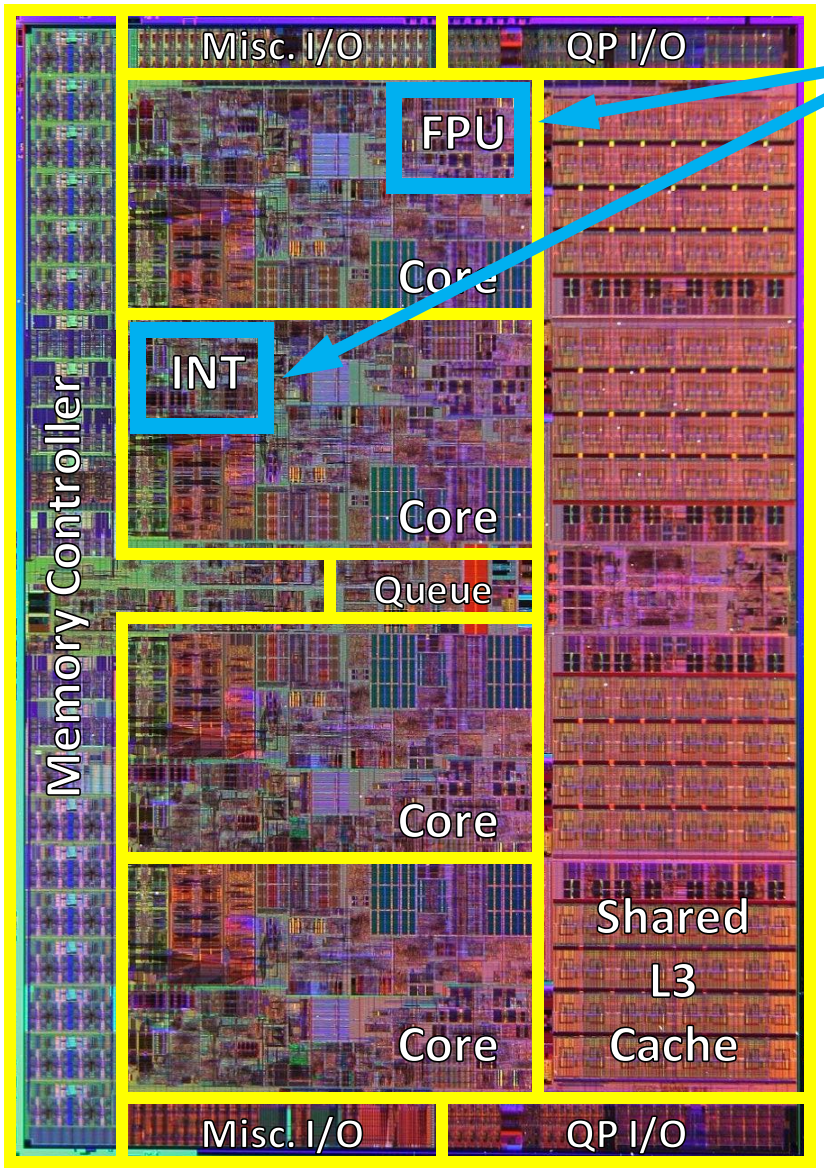
A_{op} explains the difference: more parallelism (higher N_{op}) in a smaller chip area (less overhead)

Let's Look at Some Chips to Actually See the Different Architectures

We'll look at one from each category…



Microprocessor: MOPS/mW = 0.33



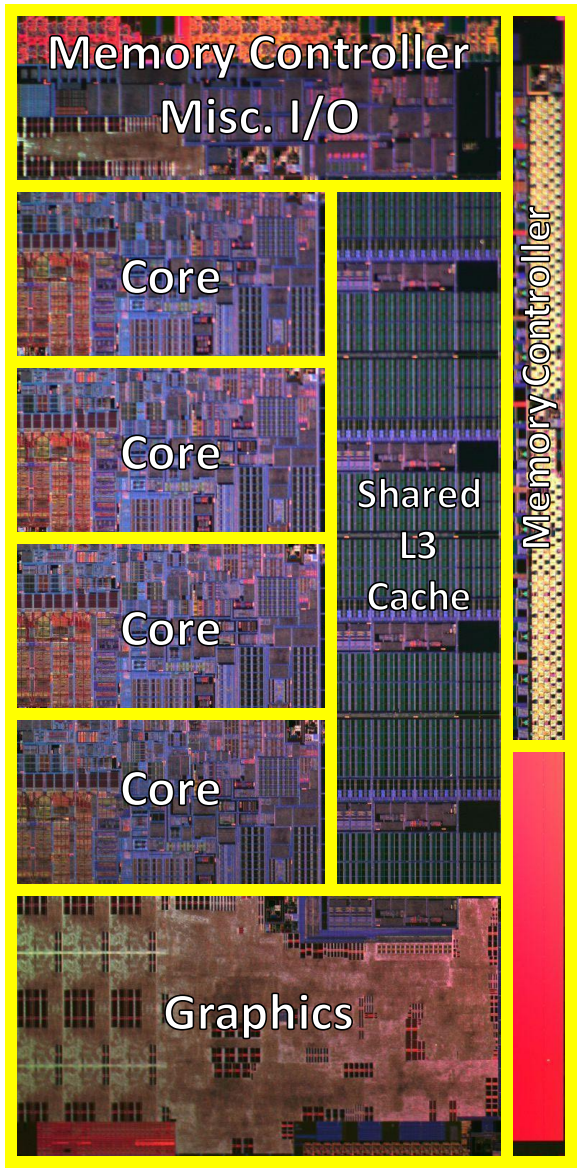
The only circuitry which supports "useful operations"
All the rest is overhead
to support the time multiplexing

N_{op} = 16
f_{clk} = 2.66 GHz
=> 42.56 GIPS

Sixteen operations each clock cycle, so
$A_{op} = A_{chip}/16$ = 31.4 mm^2

Power = 130 Watts

Microprocessor + GPU: MOPS/mW = 2.46



CPU: 4 cores (8 threads)

=> 4 ops per thread (SIMD)

=> $N_{op} = 32$

f_{clk} = 3.4 GHz => 108.8 GIPS

GPU: 12 cores => 8 ops per core (SIMD)

=> $N_{op} = 96$

f_{clk} = 1.3 GHz => 124.8 GIPS

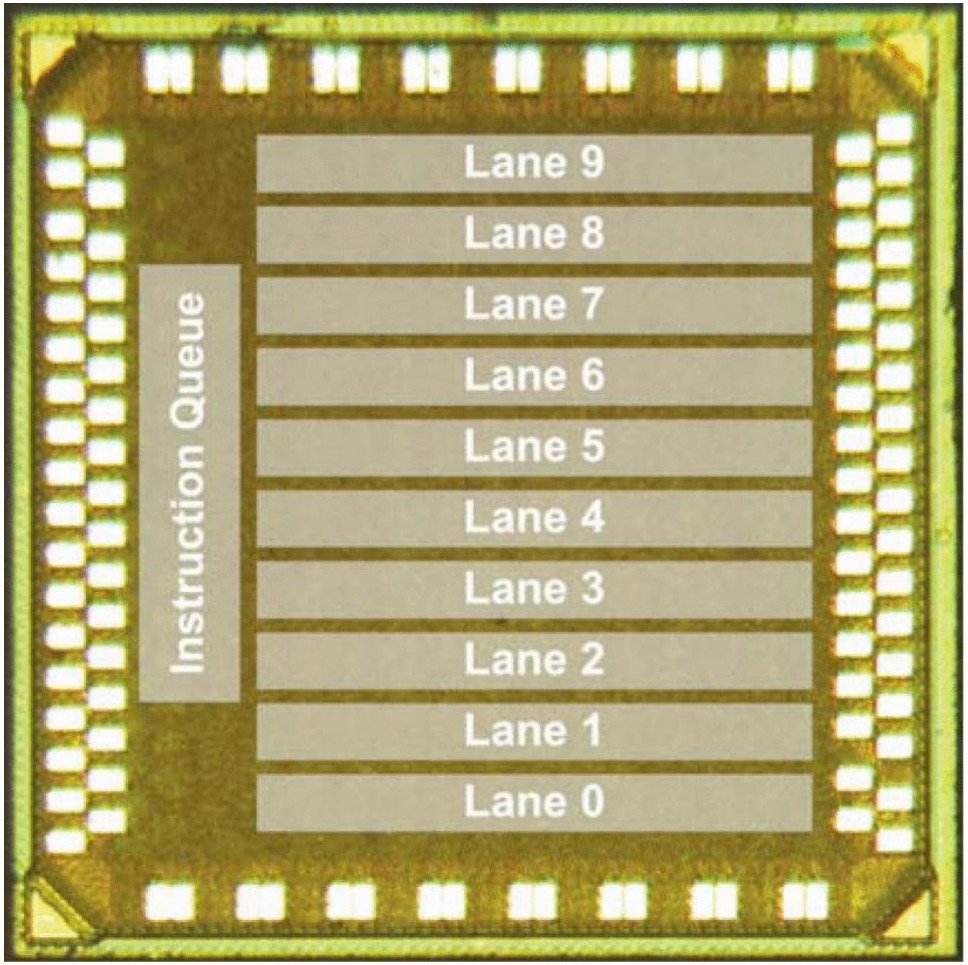
TOTAL: 233.6 GIPS

~69 operations each clock cycle (CPU), so

$A_{op} = A_{chip} / 69 = 3.14\ mm^2$

Power = 95 Watts

General Purpose DSP: MOPS/mW = 16



Same granularity (a datapath), more parallelism

10 Parallel processors
(2 for estimation and ECC)

$N_{op} = 8$

$f_{clk} = 550$ MHz

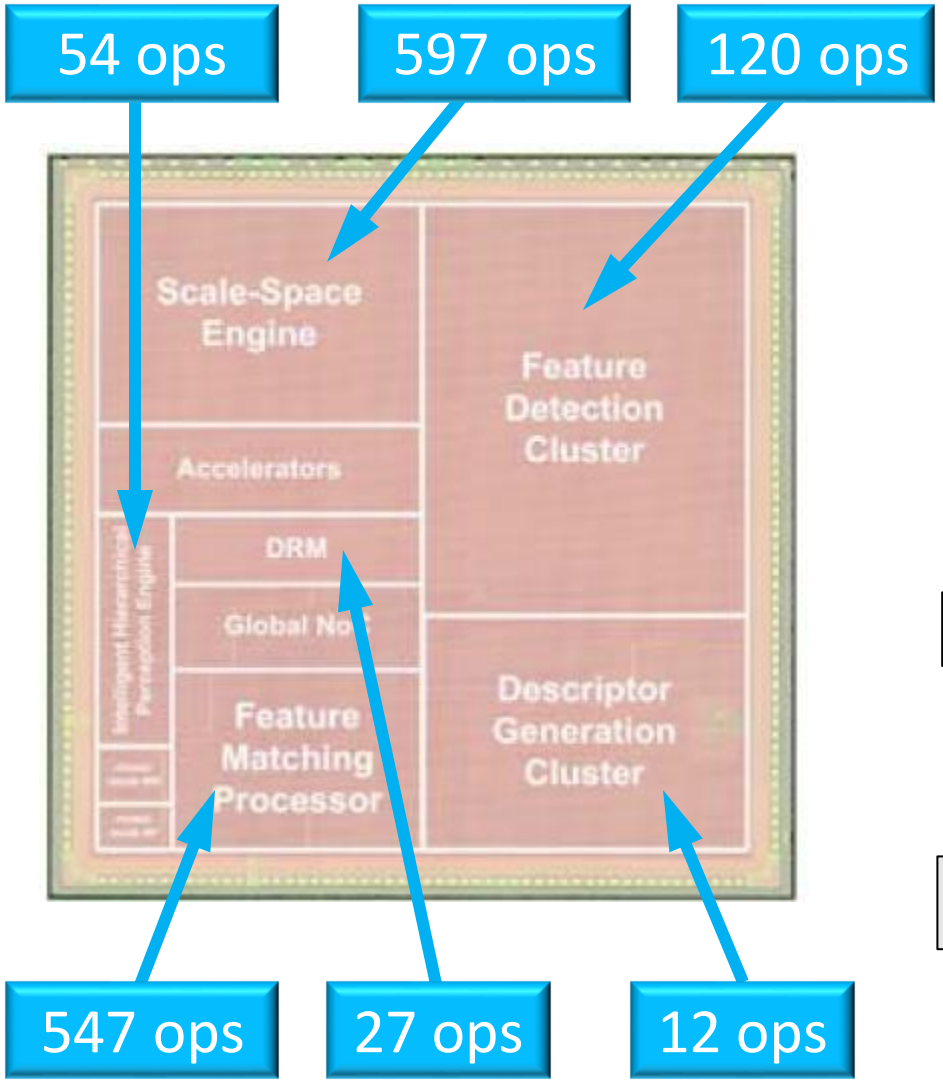
=> 4.4 GOPS

Eight operations each clock cycle, so

$A_{op} = A_{chip} / 8 = 0.5$ mm^2

Power = 275 mW

Dedicated Design: MOPS/mW = 650



Fully parallel mapping of object recognition algorithm. No time multiplexing.

N_{op} = 1357

f_{clk} = 200 MHz

=> 271.4 GOPS

$A_{op} = A_{chip}$ /1357 = 0.02 mm^2

Power = 420 mW

The Basic Problem is Time Multiplexing

- **CPU architectures obtain performance by increasing the clock rate, because the parallelism is low***
- **Results in ever increasing memory on the chip, high control overhead and fast area consuming logic**

But doesn't time mux give better area efficiency?

*T. A.C.M. Claasen, "High Speed: Not the Only Way to Exploit the Intrinsic Computational Power of Silicon," in *Proc. IEEE Int. Solid-State Circuits Conf.*, Feb. 1999, pp. 22-25.

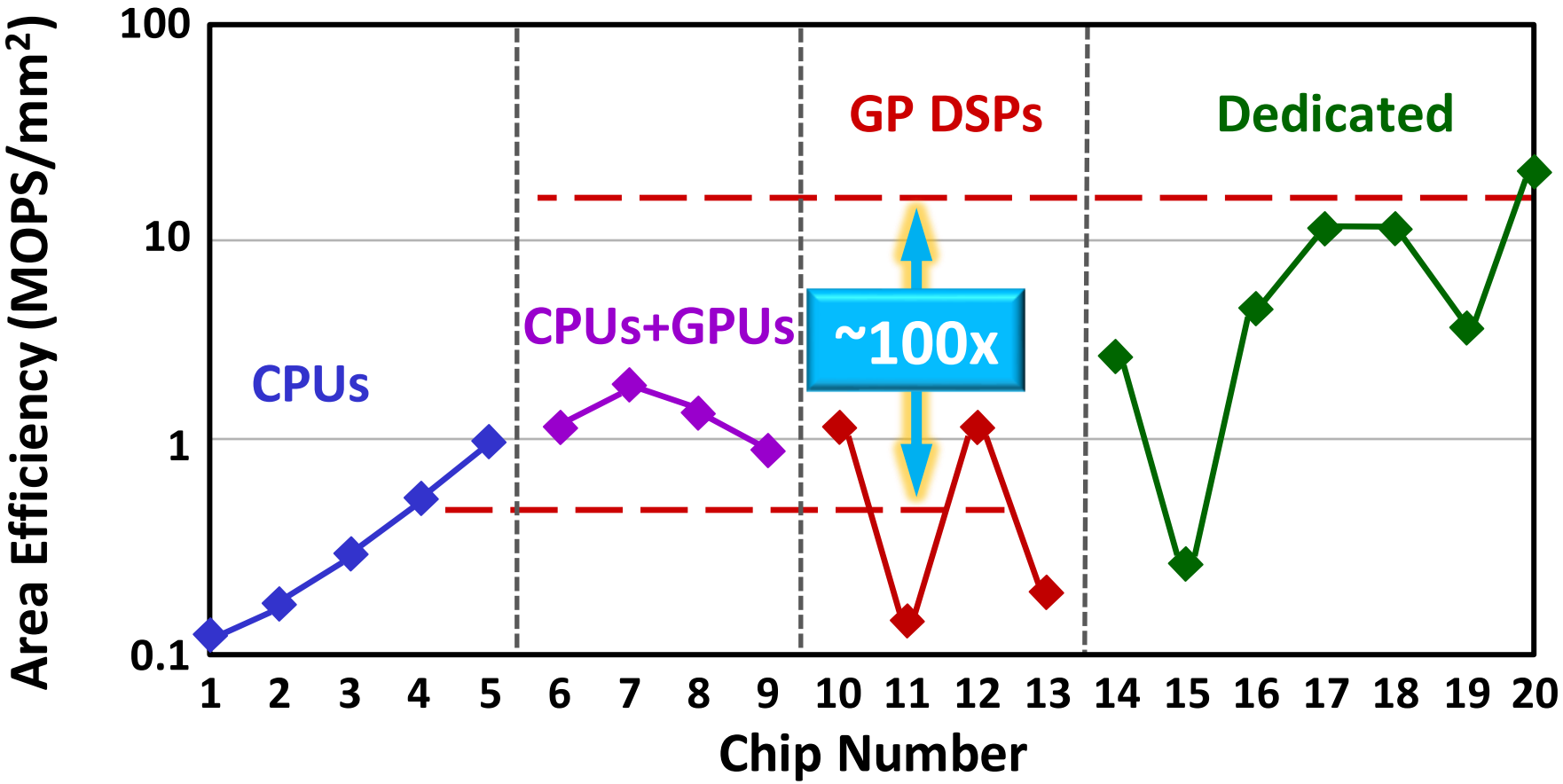
Area Efficiency

- SOC based devices are often very cost sensitive
- So we need a $ cost metric => for SOC's that is equivalent to the efficiency of area utilization
- Area-efficiency metric:

  Computation per unit area = MOPS/mm^2

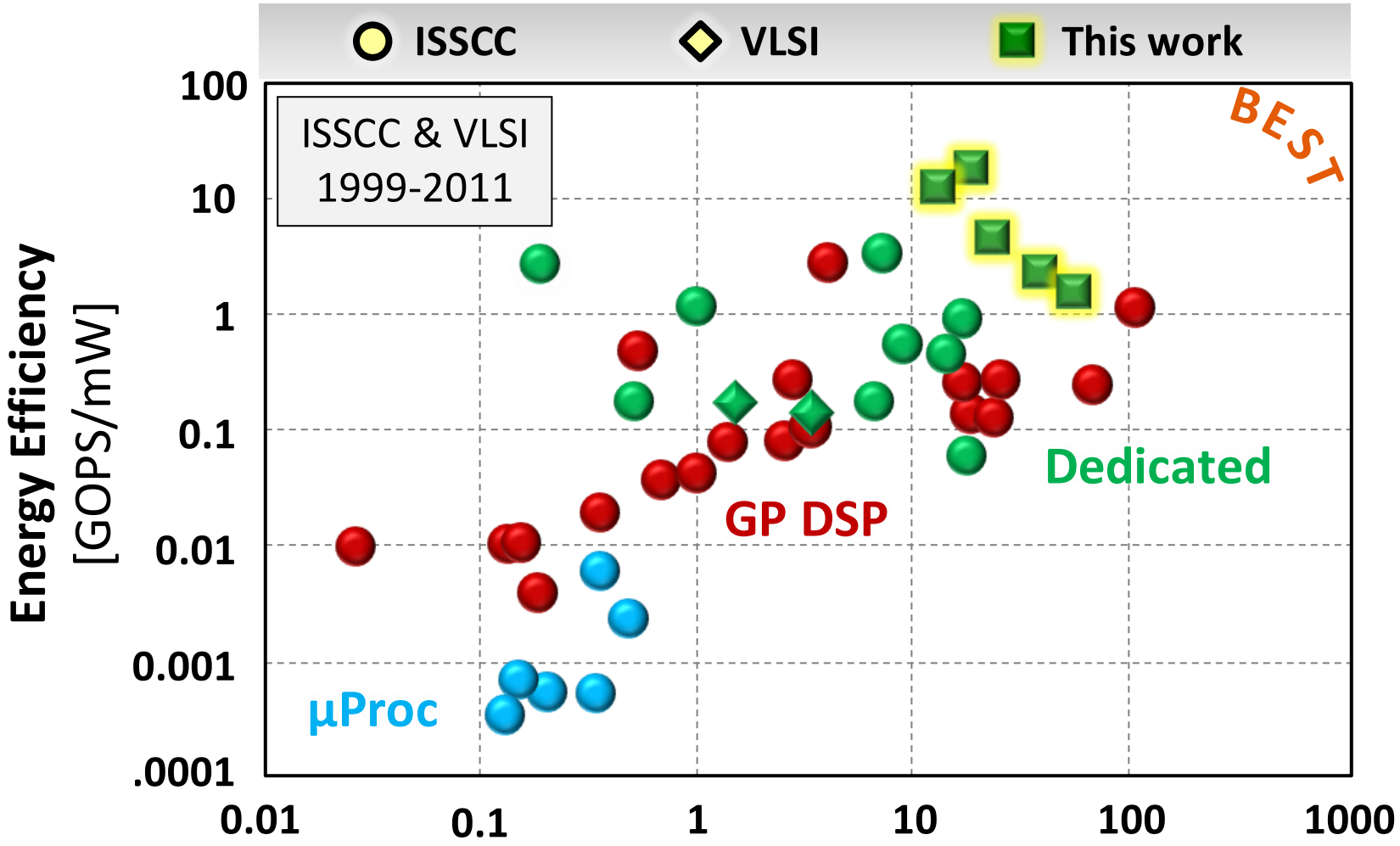
How much of a $ cost (area) penalty will we have if we put down many parallel hardware units and have limited time multiplexing?

Surprisingly, the Area Efficiency Roughly Tracks the Energy Efficiency



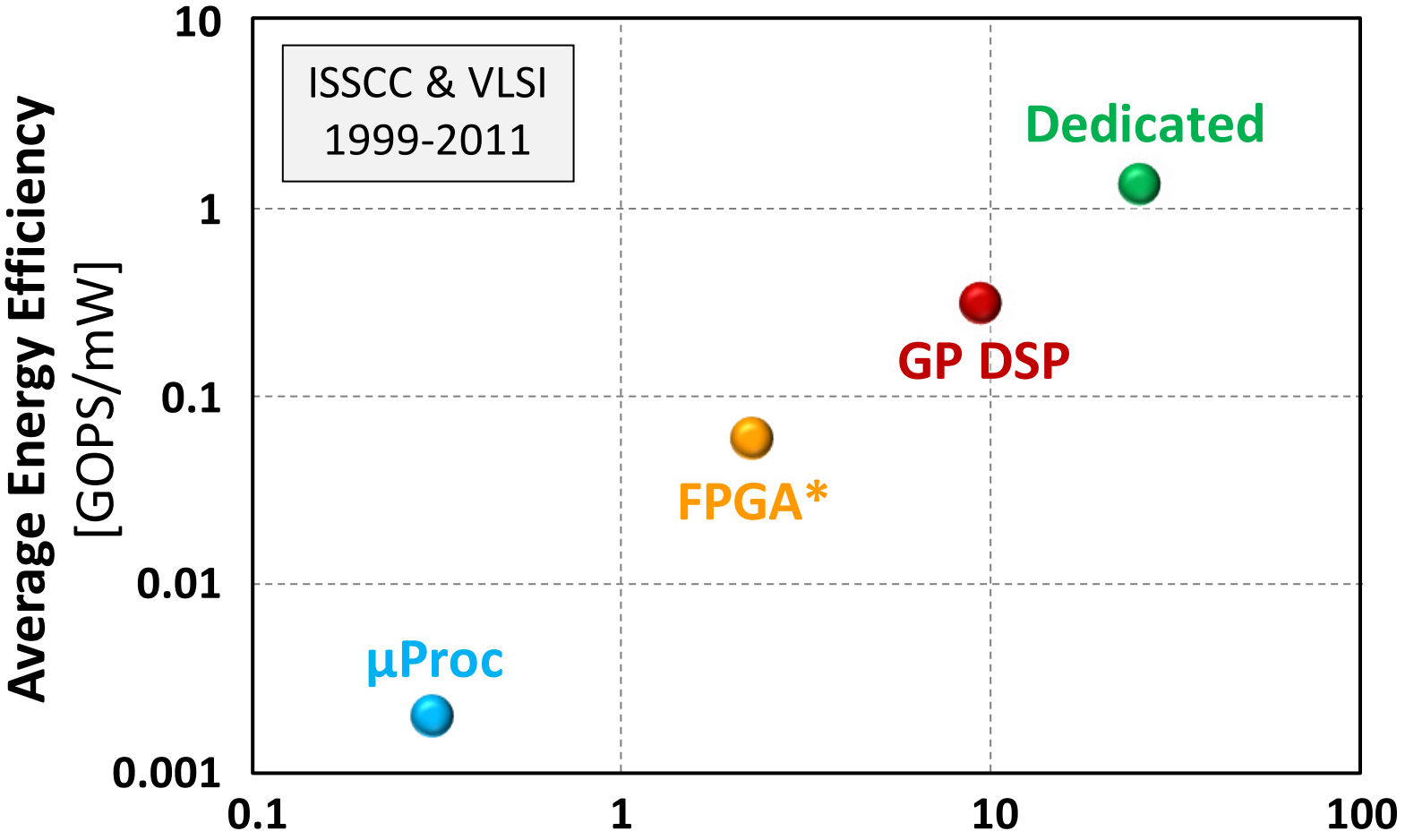
The overhead of flexibility in processor architectures is so high that there is even an area penalty

Chip Olympics



Data normalized to a 65nm technology

Chip Olympics: Average E/op and A/op



Data normalized to a 65nm technology

**optimistic estimate*

Fixed vs. Reconfigurable Architecture

Fixed architectures

- **CPU, GPU, DSP, Dedicated**

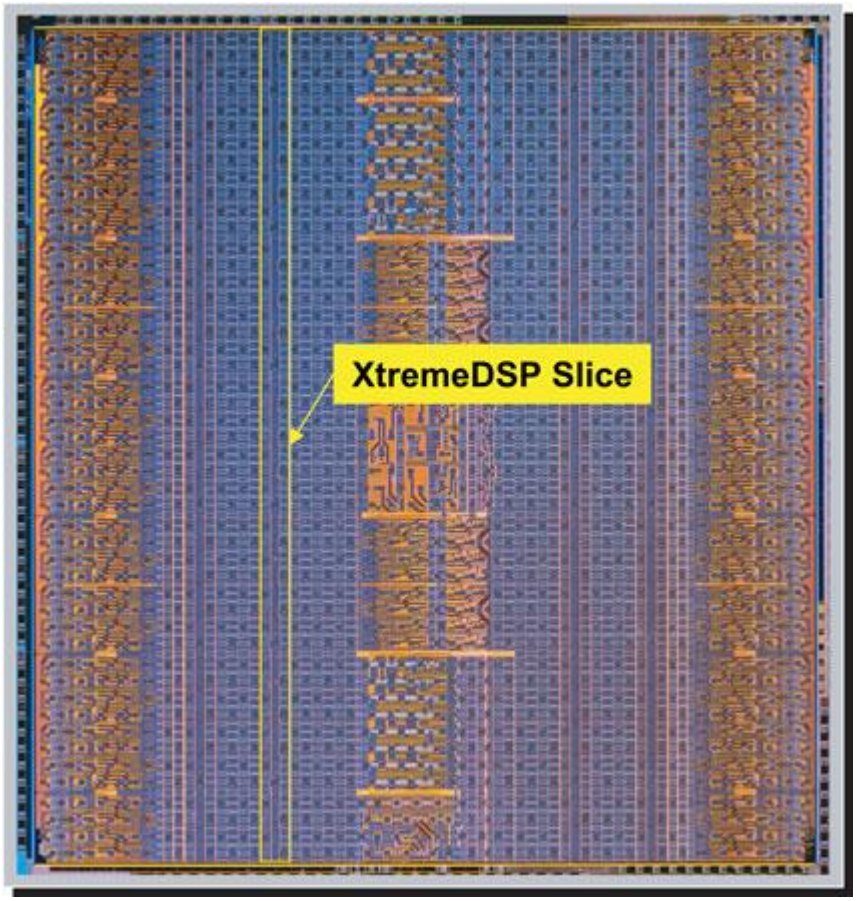
Reconfigurable architecture

- **FPGA**

Ways to Achieve FLEXIBILITY

	Software	Hardware
Feature	Programmable DSP	FPGA (Flexible DSP)
Architecture	Fixed	Reconfigurable
Operations	Conditional	Repetitive
Multi-core	Hard	Easy
Throughput	Low/mid	High

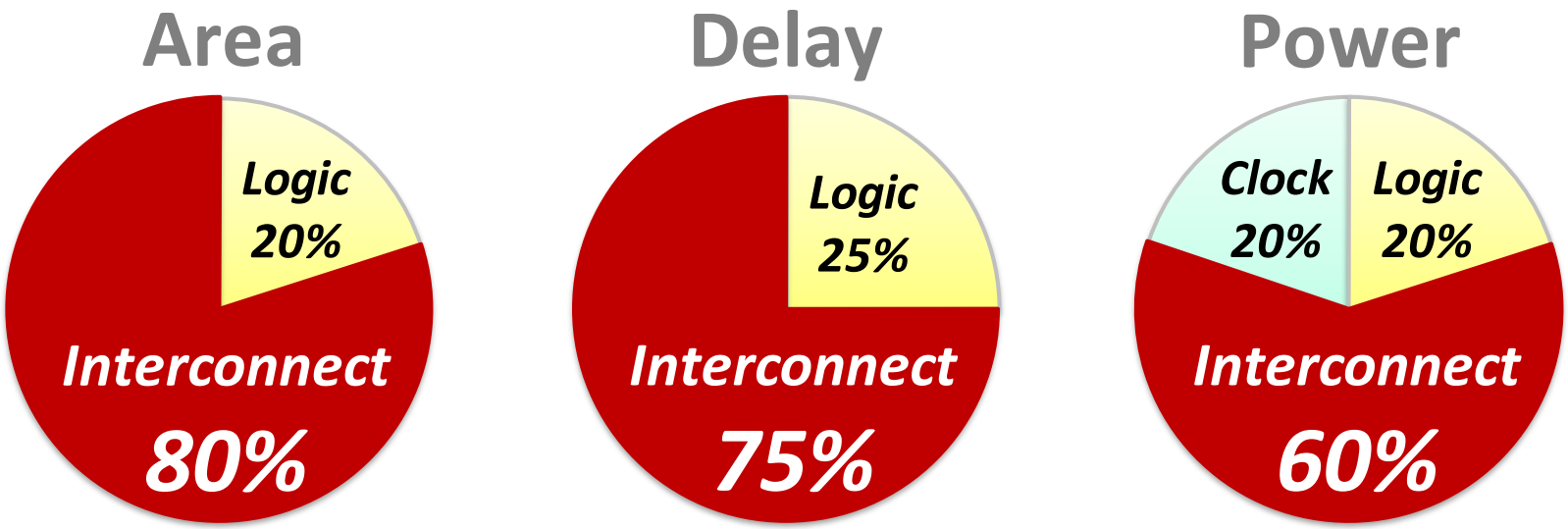
Reconfigurable Architecture: Virtex-4 Chip



- Architecture than can adapt to data
- Large degrees of parallelism possible

FPGAs are Flexible but Inefficient

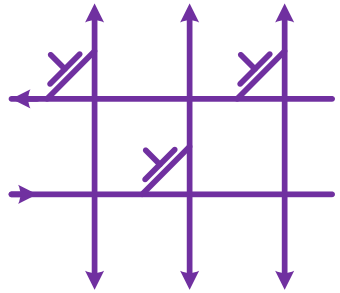
- **Compared to dedicated chips, FPGAs incur penalties in**
  - Area (17 – 54 x)
  - Speed (3 – 7 x)
  - Power (6 – 62 x)
- **Main culprit: interconnect!**



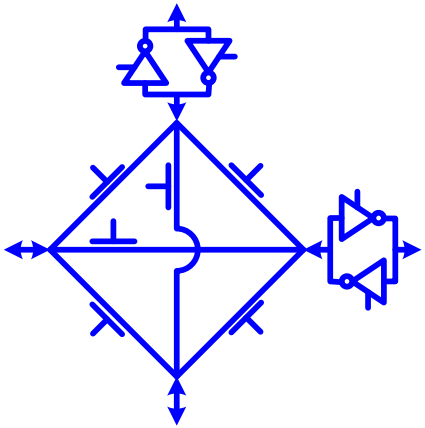
I. Kuon, et al., Found. & Trends in Elec. Design Automation 2007
I. Bolsens, MPSOC 2006; B. Calhoun, et al., Proc. IEEE 2010

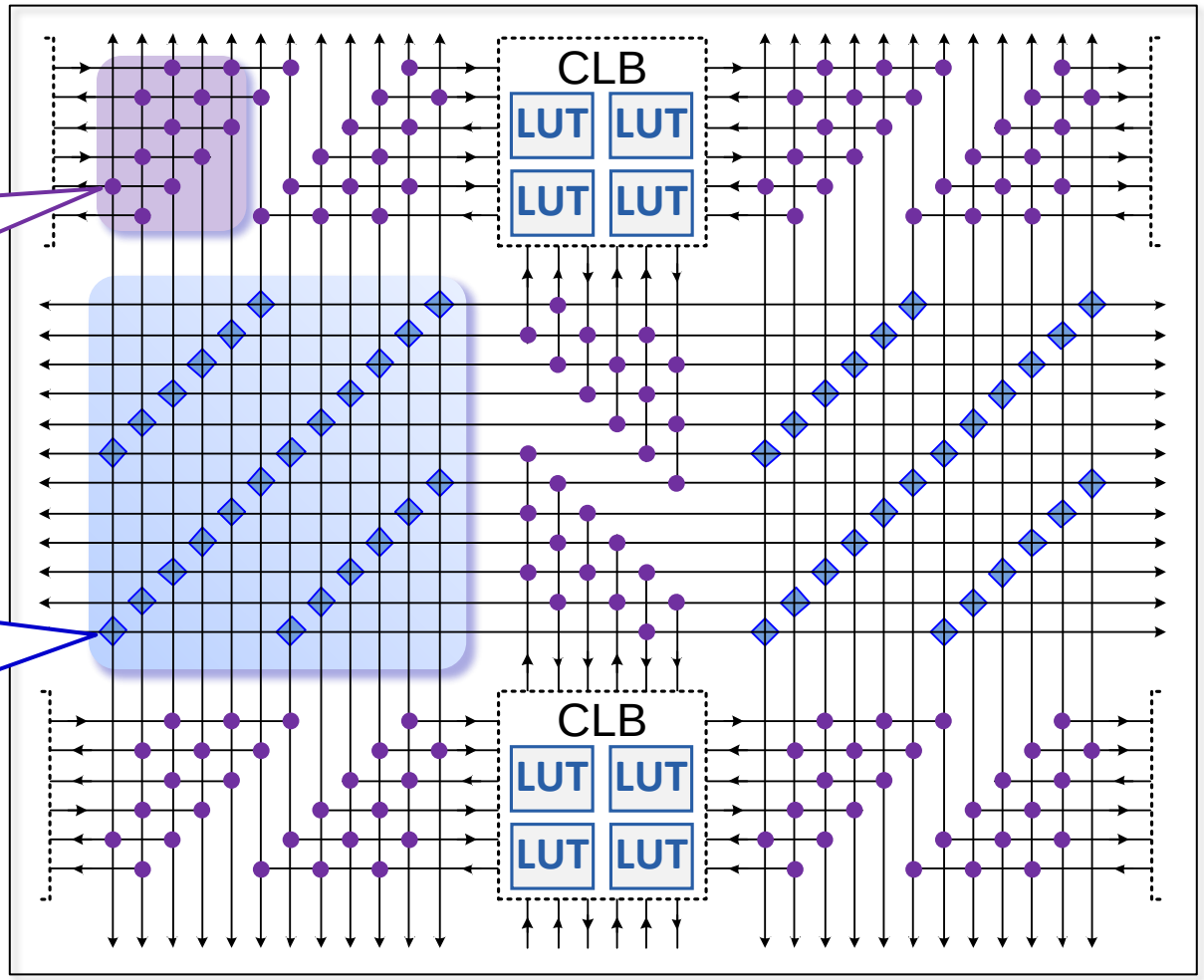
2D Mesh Architecture: 80% Interconnect Area

Connection box



Switch box





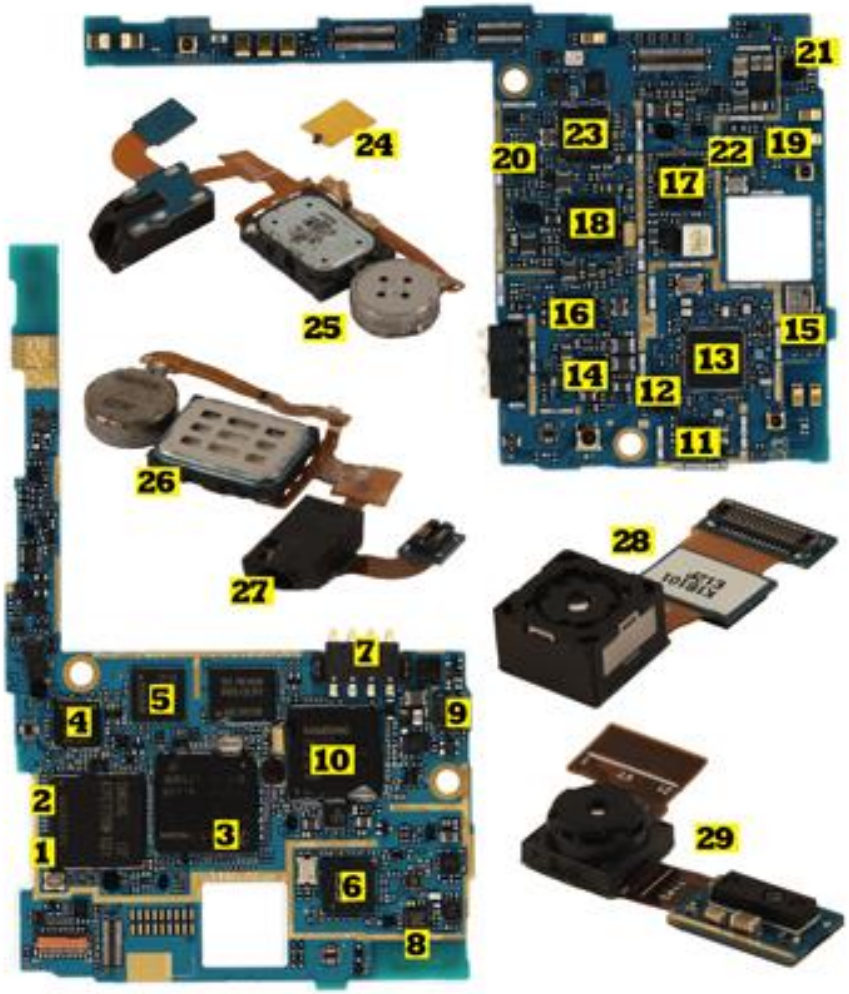
Hardware / Software

There is no software/hardware tradeoff!

- The difference between hardware and software in performance, power and area is so large that there is no "tradeoff"
- It is reasons other than energy, performance or cost that drives a software solution (e.g. business, legacy, …)
- The "Cost of Flexibility" is extremely high, so the other reasons better be good!

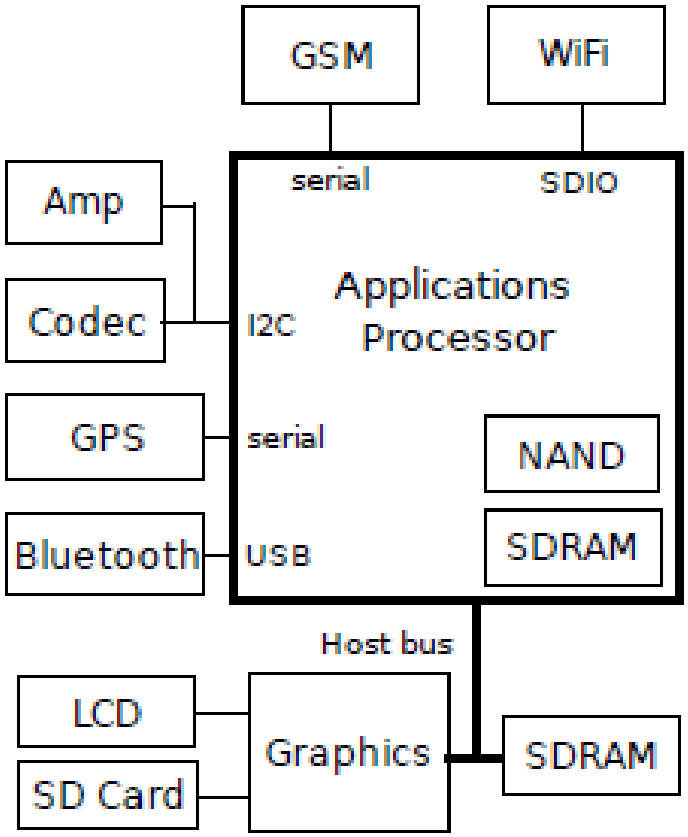
System Examples

Case 5.1: Smartphone | Power Breakdown?



What matters?

Analysis Framework*

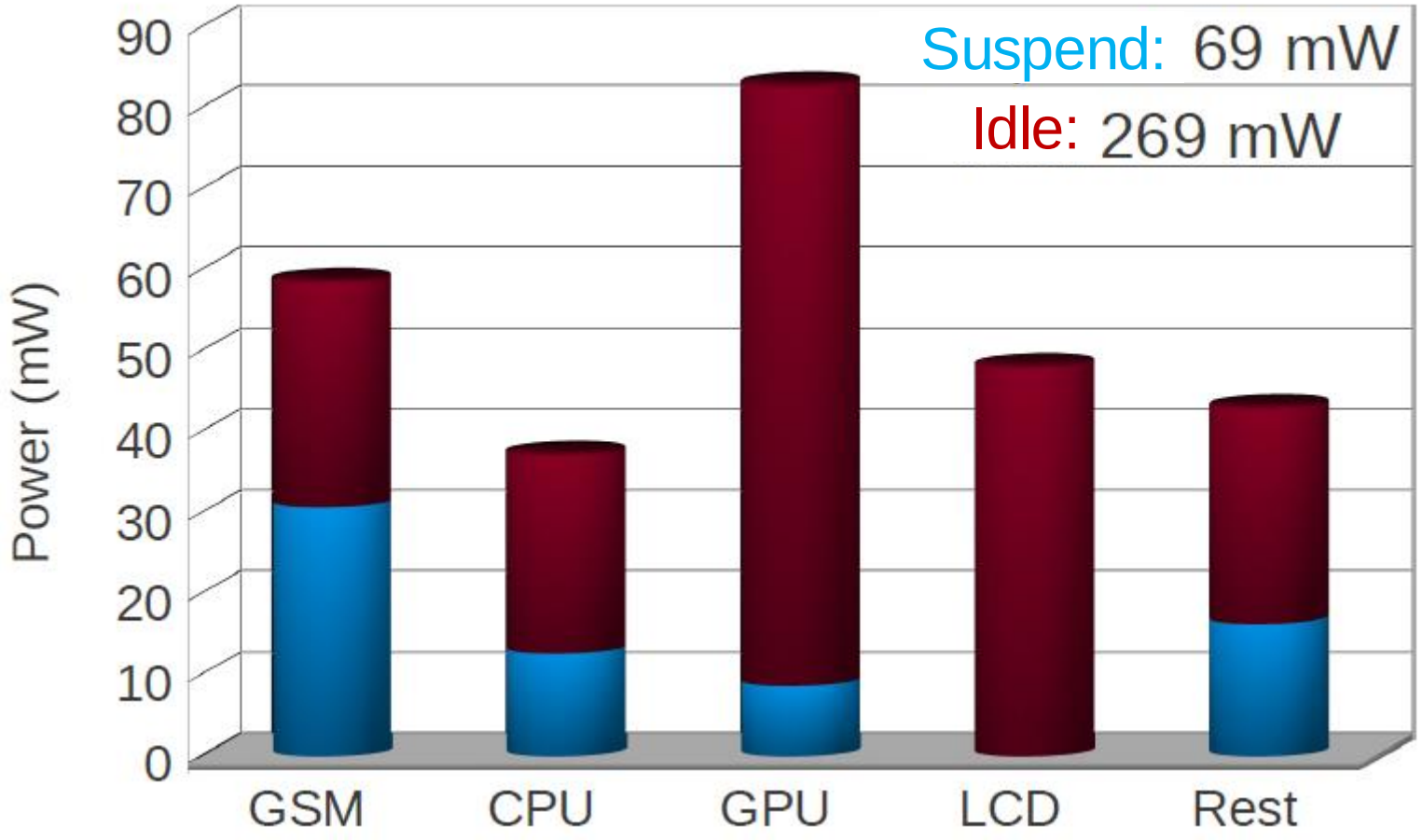


*A. Carroll and G. Heiser, "An Analysis of Power Consumption in a Smartphone," USENIX ATC 2010.

- **OpenMoko Freerunner**
  - 2.5G smartphone (2008)
  - 400 MHz ARM9
  - No camera, 3G or 4G modem
  - Open design | Android 1.5
  - Factory-configured for power measurements
- **Measured battery power**
  - HTC Dream
  - Google Nexus One

System Components | Backlight Off

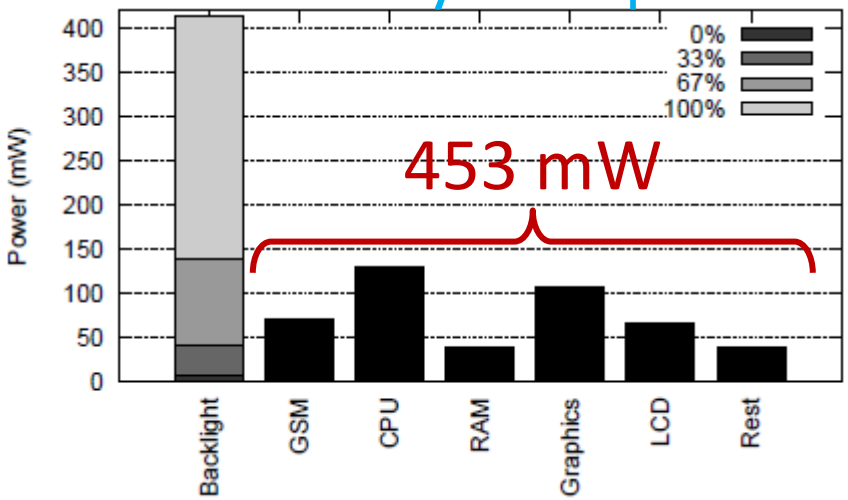
Backlight: **8mW** | **75mW** | **414mW**
min 50% max



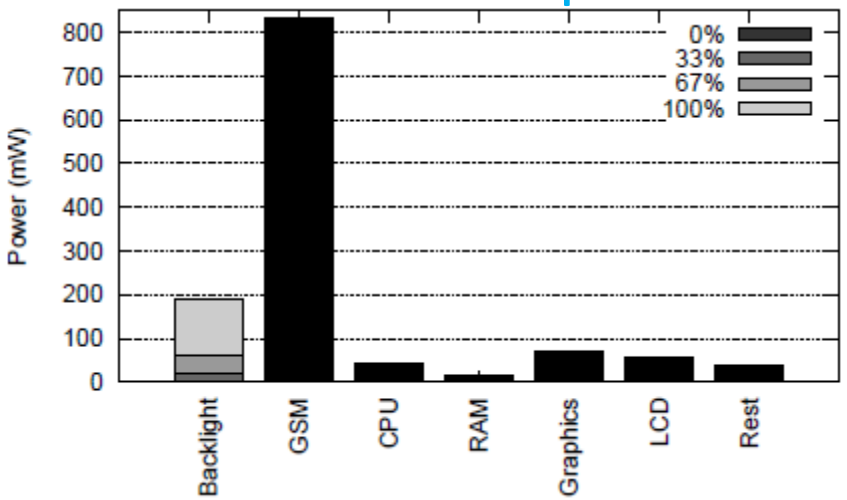
Courtesy: A. Carroll and G. Heiser

Use Scenarios | Common Apps

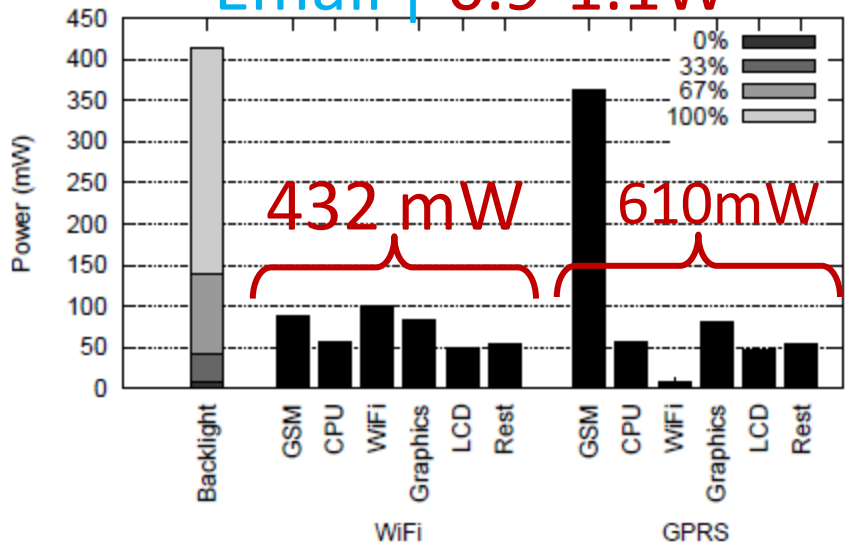
Video Playback | 0.9W



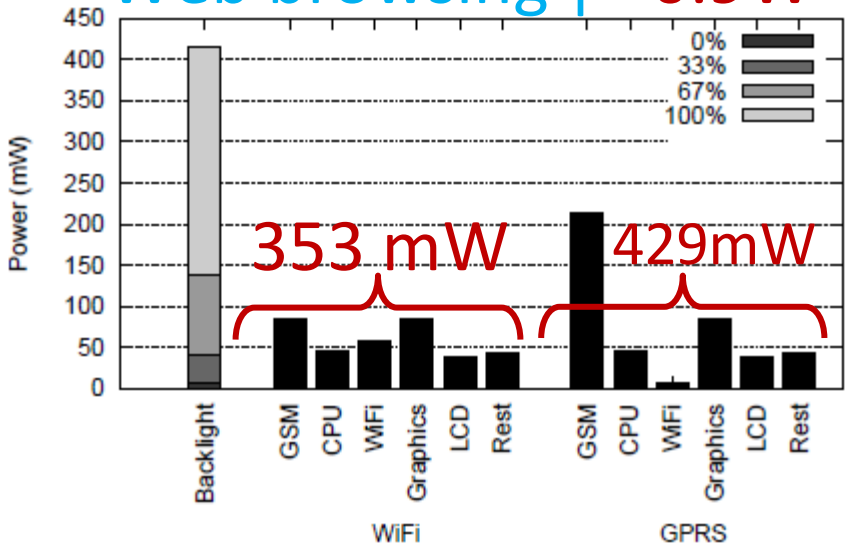
Phone Call | 1.1W



Email | 0.9-1.1W



Web browsing | <0.9W



Courtesy: A. Carroll and G. Heiser

Freerunner, HTC Dream, Google Nexus

Benchmark	Average System Power (mW)		
	Freerunner	G1	N1
Suspend	103.2	26.6	24.9
Idle	333.7	161.2	333.9
Phone call	1135.4	822.4	746.8
Email (cell)	690.7	599.4	-
Email (WiFi)	505.6	349.2	-
Web (cell)	500.0	430.4	538.0
Web (WiFi)	430.4	270.6	412.2
Network (cell)	929.7	1016.4	825.9
Network (WiFi)	1053.7	1355.8	884.1
Video	558.8	568.3	526.3
Audio	419.0	459.7	322.4

$$
\begin{aligned}
E_{\text{audio}}(t) &= 0.32W \times t \\
E_{\text{video}}(t) &= (0.45W + P_{\text{BL}}) \times t \\
E_{\text{sms}}(t) &= (0.3W + P_{\text{BL}}) \times t \\
E_{\text{call}}(t) &= 1.05W \times t \\
E_{\text{web}}(t) &= (0.43W + P_{\text{BL}}) \times t \\
E_{\text{email}}(t) &= (0.61W + P_{\text{BL}}) \times t
\end{aligned}
$$

Additional notes

- **RAM power insignificant in real workloads**
- **Bluetooth: < 40mW**
- **DVFS works for N1**

Recommendations

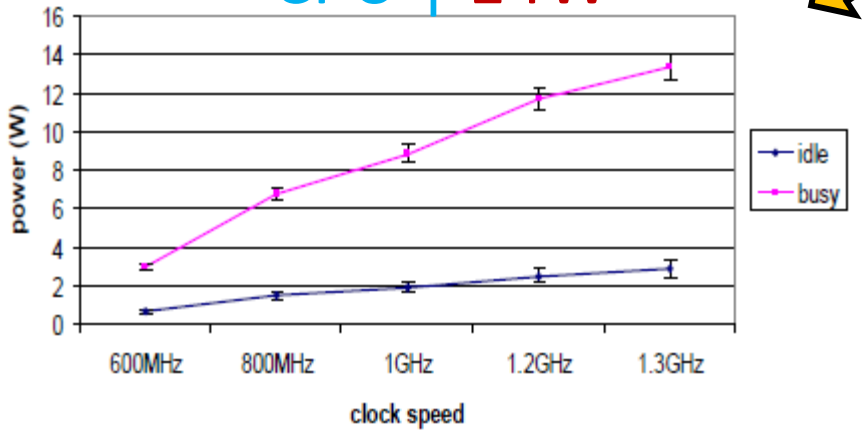
- **Use WiFi when possible**
- **Dim the display**
  - OLED display: 1.1W
  - Light-on-dark color scheme saves power

Case 5.2: Laptop | Power Breakdown?

System Components

Component	Details
Processor	1.3 GHz Pentium M
Memory	256 MB
Hard Drive	40 GB @ 4200 RPM
Optical Drive	CD-R/RW, DVD
Wireless Networking	Intel Pro Wireless 2100
Screen	14.1" 1048 x 768

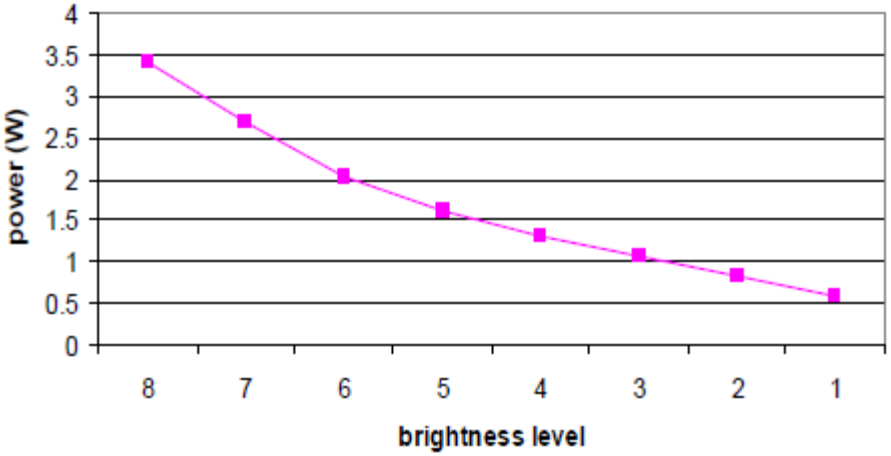
CPU | 14W



WiFi | 3W

Wireless Card States	Power Consumption
Power Saver (Idle)	0.14 W
Base (Idle)	1.0 W
Transmit	3.12 W total at 4.2 Mb/s
Receive	2.55 W total at 2.9 Mb/s

Display | 3W



HD | 2.8W

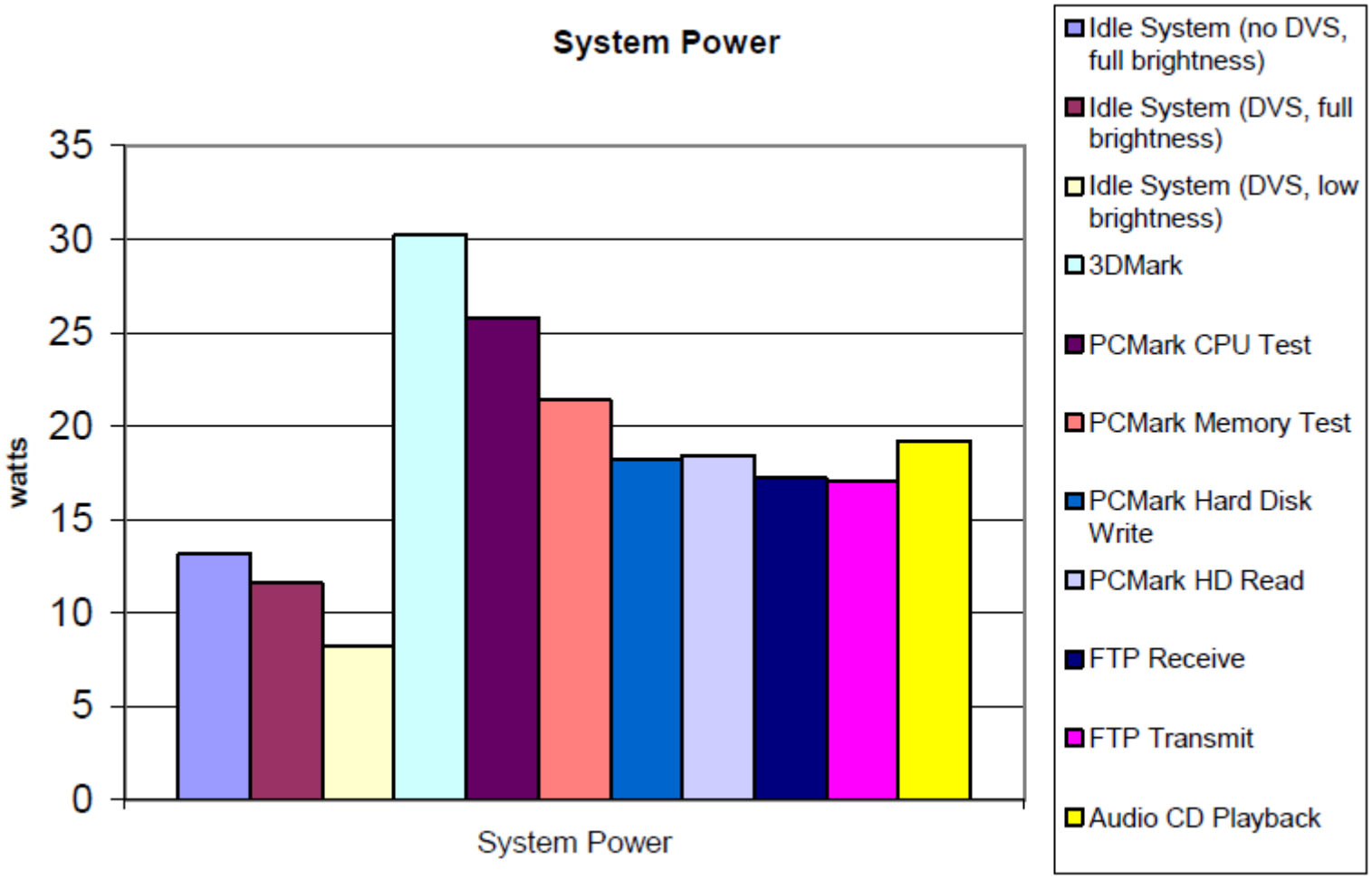
Hard Drive State	Power Consumption
Idle	.575 W
Standby	.173 W
Read	2.78 W
Write	2.19 W
Copy	2.29 W

Optical D | 5.3W

Optical drive state	Power (W)
Initial spin up	3.34
Steady spin	2.78
Reading data	5.31

Courtesy: A. Mahersi, V. Vardhan

App Benchmarking (PCMark, 3DMark)



A. Mahersi, V. Vardhan, "Power Consumption on a Modern Laptop," in Proc. Workshop on Power-Aware Computing Systems, Dec 2004, pp. 165-180.

The Laptop Case

- CPU power dominates for many apps
- Display dominates during system idle
- Graphics, WiFi and optical D matter in specific workloads

Summary and Next Lecture

- **Parallelism provides energy efficiency**
- **Design problem: flexibility and efficiency**
- **Efficiency is not just about computations**
- **Rules of thumb (stuff we'll cover next time)**
  - Fixed → Floating (1.5-5x penalty)
  - Single → Double (3.5x penalty)
  - Math → MEM access (2x penalty, SRAM $)
  - Math → MEM access (5.5x penalty, DRAM)
  - Sequential → Random (5x penalty)

⊕ O(N)
⊗ O(N²)
MAC 3.6x penalty to double precision

E.g. 16b fix-pt —3x→ 16b float —3.6x→ 32b float —3.6x→ 64b float ⇒ ~38x total

1,000x CPU vs. ASIC
40x arith.
25x arch