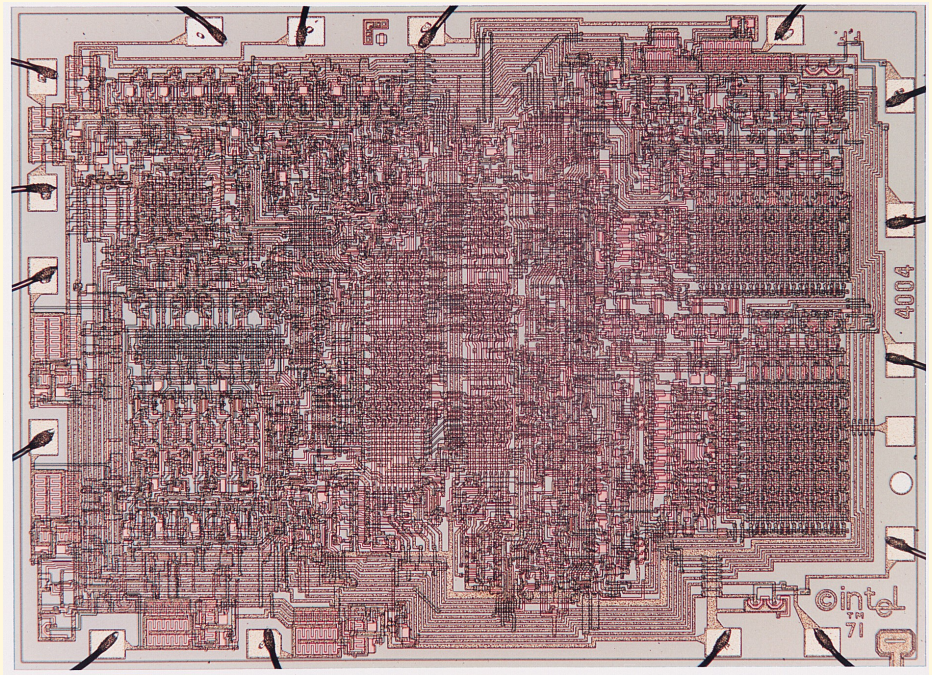
Computer Systems Research in the Post-Dennard Scaling Era

Emilio G. Cota

Candidacy Exam

April 30, 2013



Intel 4004, 1971

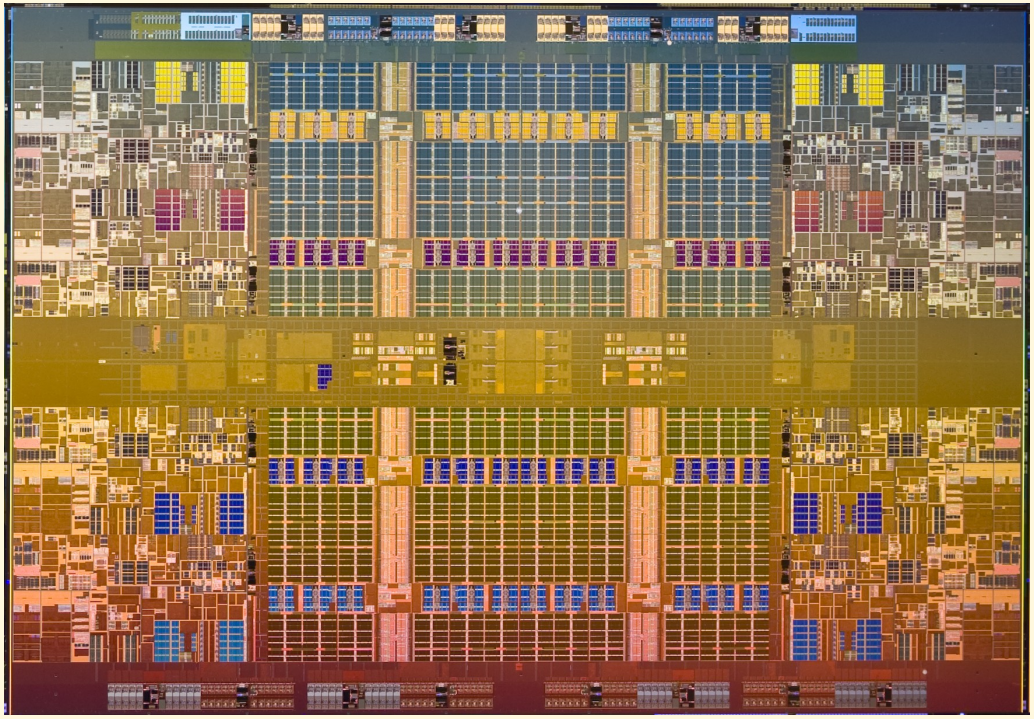
1 core, no cache

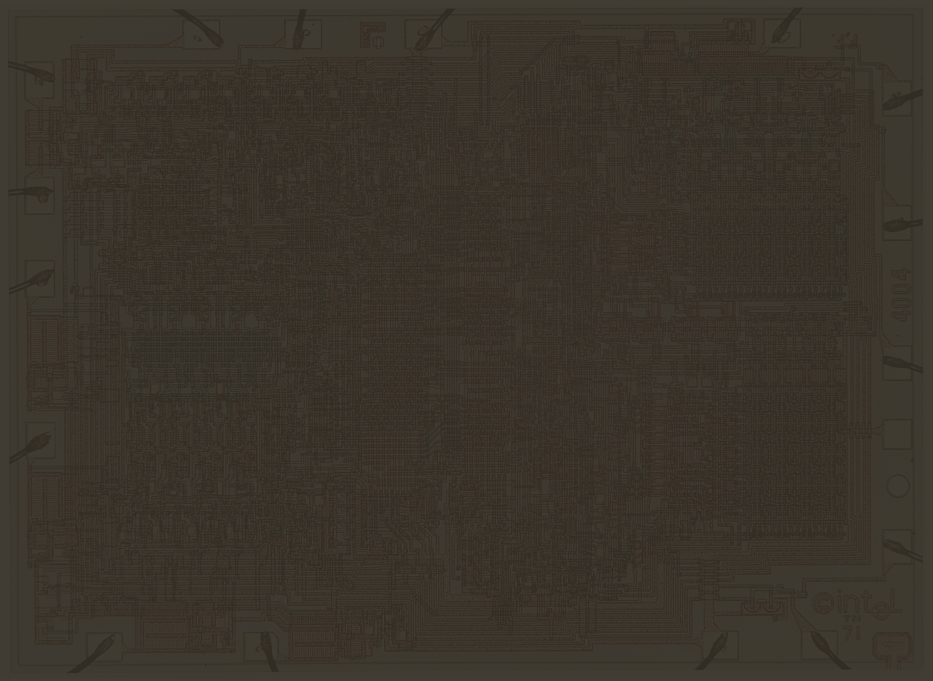
23K 10um transistors

Intel Nehalem EX, 2009

8c, 24MB cache

2.3B 45nm transistors



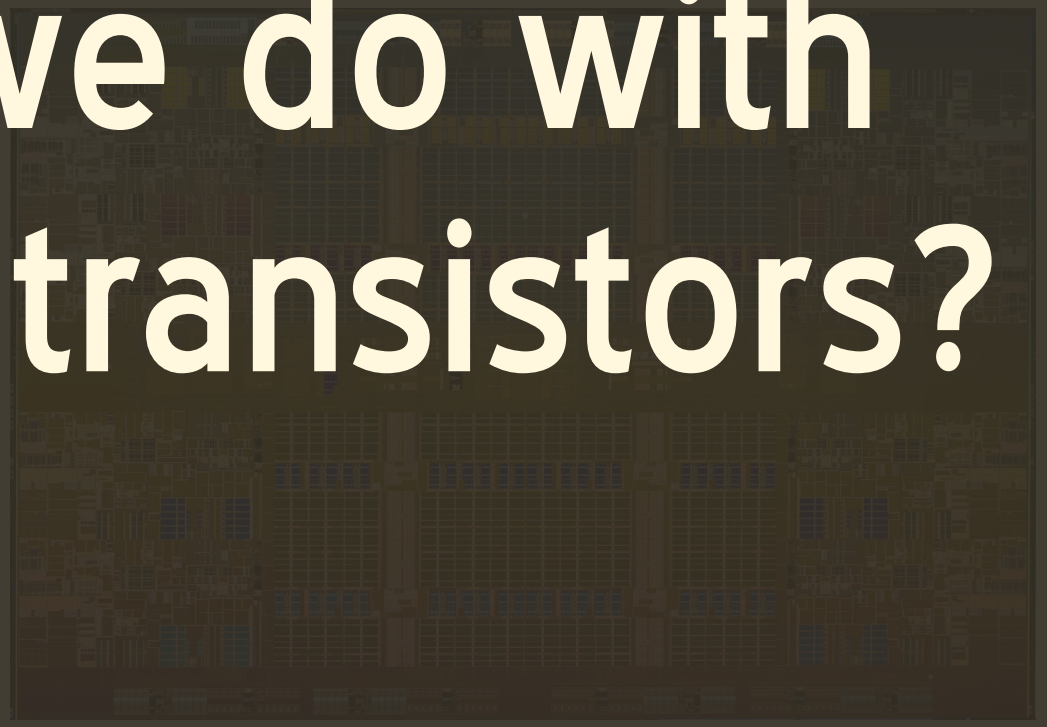


Intel 4004, 1971

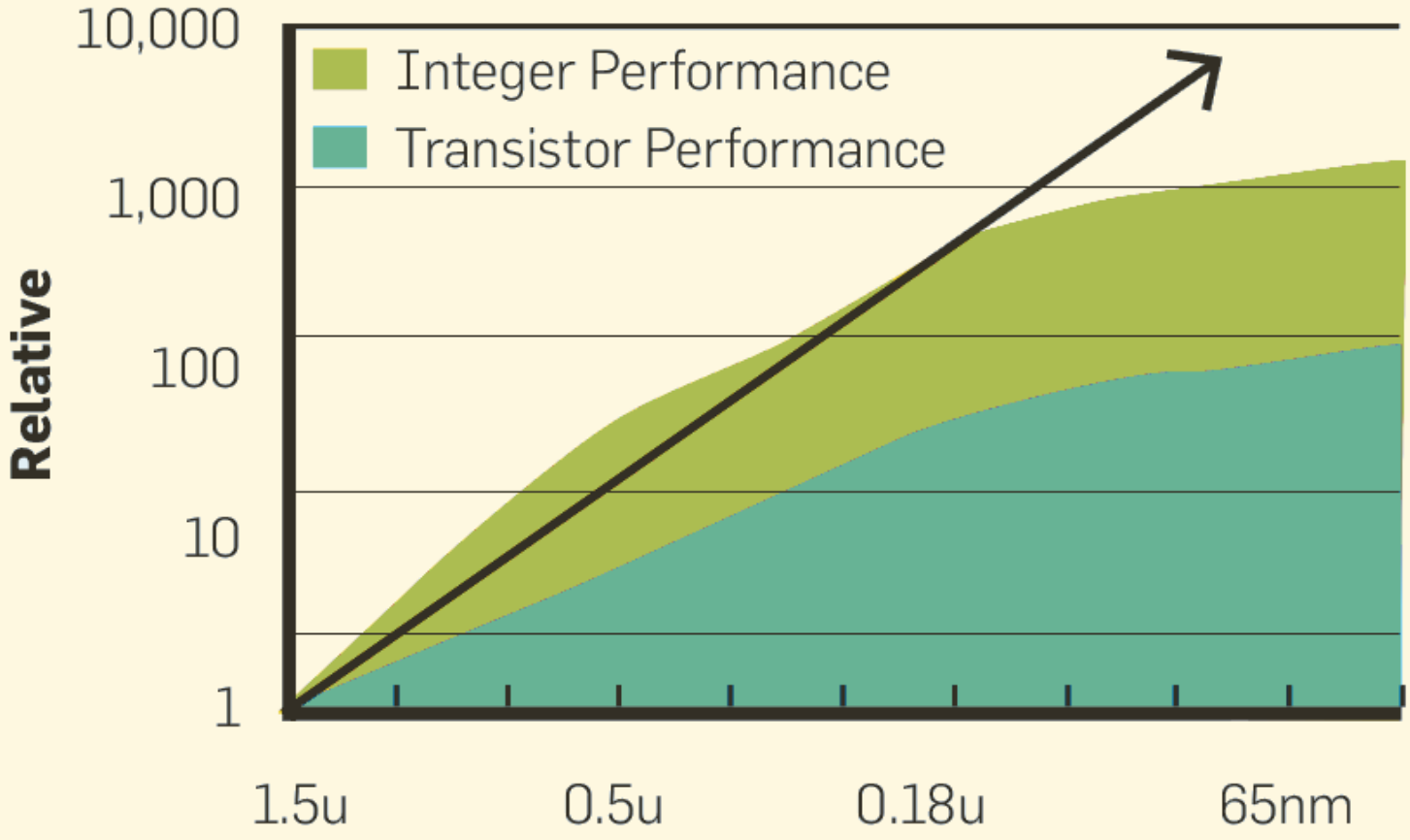
1 core, no cache

23K 10um transistors

What did we do with those 2B+ transistors?



2.3B 45nm transistors



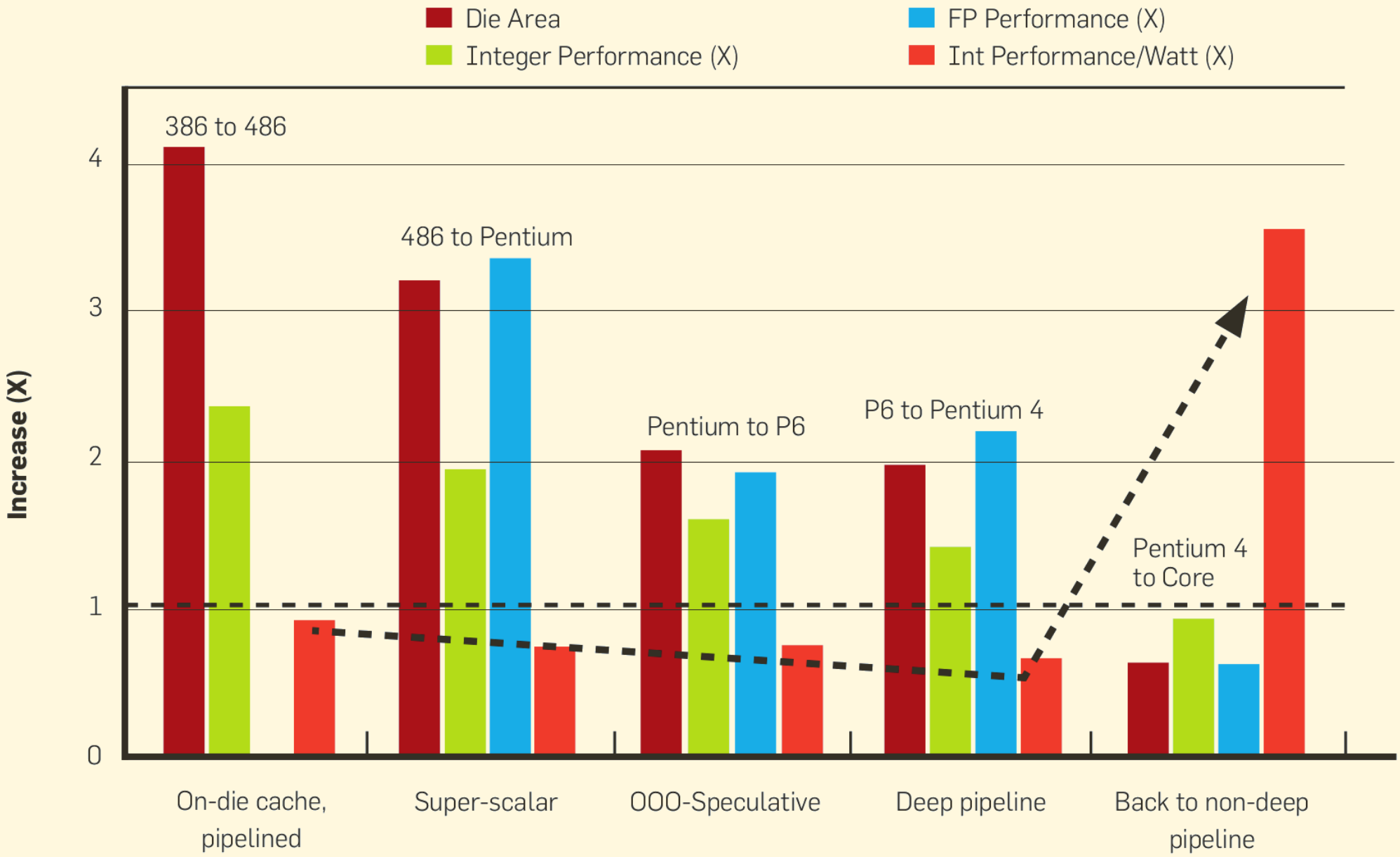
20 years

1000x speedup

10x: architectural innovations

100x: transistor scaling

[Borkar 2011]



20 years of architectural innovation for a 10x speedup

[Borkar 2011]

Dennard scaling

Every technology generation brings:

50% area reduction

40% speed increase

50% less power consumption

[Borkar 2011]

Dennard scaling is no more

Leakage current grows exponentially with $\downarrow V_{th}$

To mitigate leakage power, threshold voltage is now increasing, limiting speed

Further, supply voltage scaling is severely restricted by process variability

Result: below 130nm power density grows every generation

[Borkar 2011]

growing power density +
fixed power budgets =
increasingly large portions of

dark silicon

as technology scales

[Borkar 2011]

Fighting dark silicon

Process innovations (!= traditional scaling)

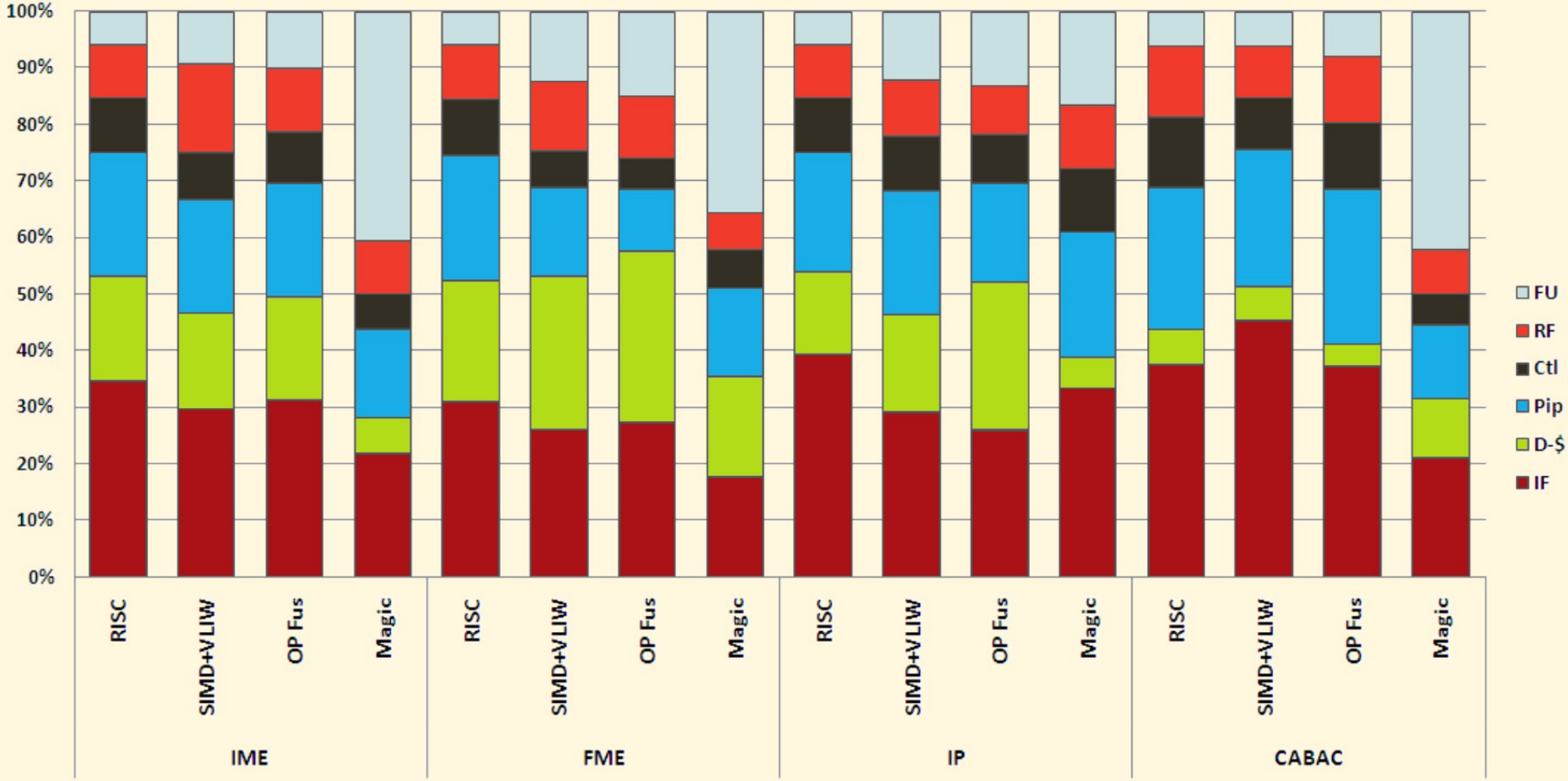
beyond this talk's scope

Increase locality and reduce bandwidth per op

how inefficient are we right now?

[Borkar 2011]

H.264 energy breakdown



"Magic" is a highly specialized implementation

yet it only achieves

up to 50% of "real" (FU and RF) work

[Hameed 2010]

Computer Systems Research in the Post-Dennard Scaling Era

Outline

Computer Systems Research in the Post-Dennard Scaling Era

Outline

I The dark silicon era

How the end of Dennard scaling shifted focus from performance to energy efficiency

Computer Systems Research in the Post-Dennard Scaling Era

Outline

I The dark silicon era

How the end of Dennard scaling shifted focus from performance to energy efficiency

II Multicore Scalability

Memory hierarchy innovations
Potential bottlenecks: Coherence & Heterogeneity

Computer Systems Research in the Post-Dennard Scaling Era

Outline

I The dark silicon era

How the end of Dennard scaling shifted focus from performance to energy efficiency

II Multicore Scalability

Memory hierarchy innovations
Potential bottlenecks: Coherence & Heterogeneity

III Heterogeneous architectures

Drastic energy savings through specialization

Part II

Multicore Scalability

Memory Hierarchy Innovations

Performance gains with little or no transistor expense

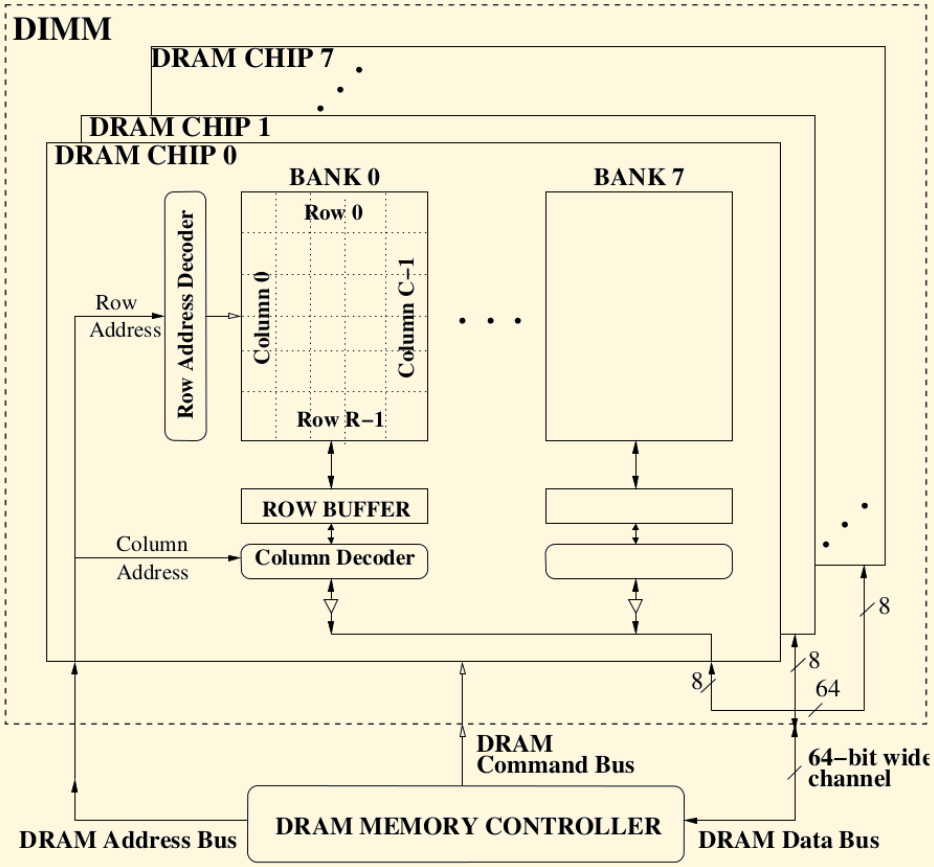
Memory Controller

Scheduling & Placement

Non-Uniform Caches

Latency reduction on last-level caches

Memory Controller Scheduling



Per bank, only one row can be accessed at any given time

Every access must go through the row buffer

Consecutive accesses to the same row are thus faster

row hit: t_{CL} < row closed: $t_{RCD} + t_{CL}$ < row conflict: $t_{RP} + t_{RCD} + t_{CL}$

[Mutlu 2007]

Memory Controller Scheduling

Traditional solution: FR-FCFS

Maximizes row hits by prioritizing *column* accesses over *row* ones

Is unfair: threads with infrequent accesses of low row locality are severely slowed down

$$\underset{\text{hit}}{t_{CL}} < \underset{\text{closed}}{t_{RCD} + t_{CL}} < \underset{\text{conflict}}{t_{RP} + t_{RCD} + t_{CL}}$$

[Mutlu 2007]

Memory Controller Scheduling

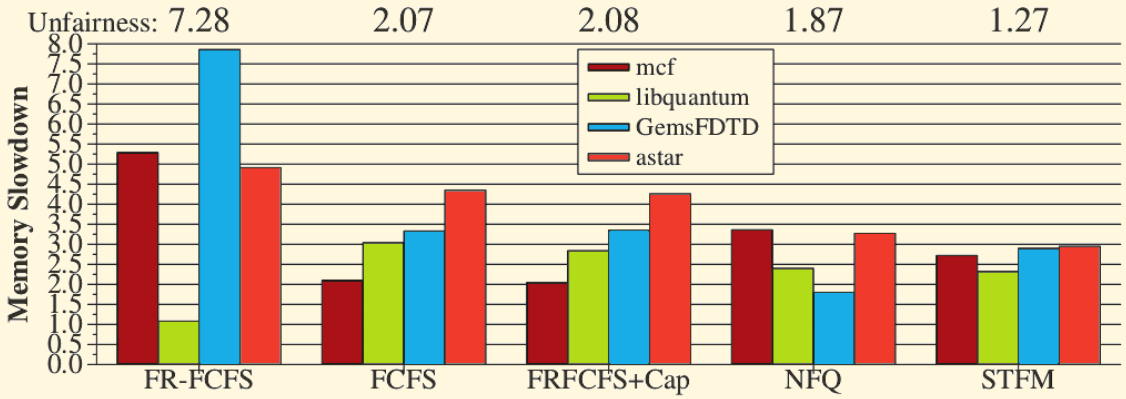
Goal: equalize memory-related slowdown across threads

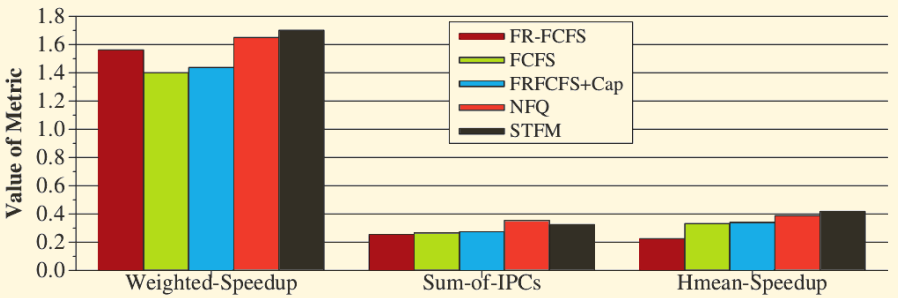
Technique

Estimate slowdown of each thread

Compute system unfairness

Prioritize commands based on the slowdowns of their threads





[Mutlu 2007]

Memory Controller Placement

Constraints

Pin count: many cores, few controllers

Uniform spread of traffic across ports

Physical considerations, e.g. thermal

Best placement: diamond

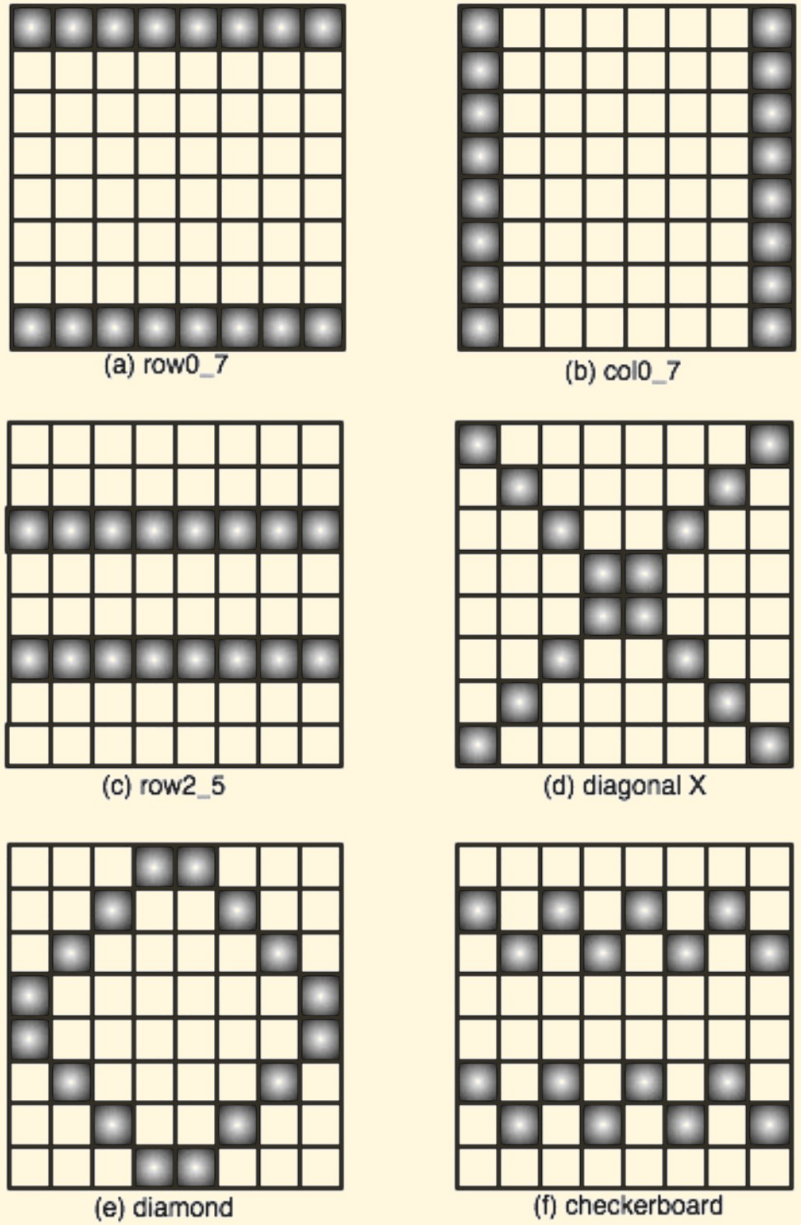
Lowest contention (<33% than row07)

Lowest latency & latency variance

Better thermal distribution than diag. X

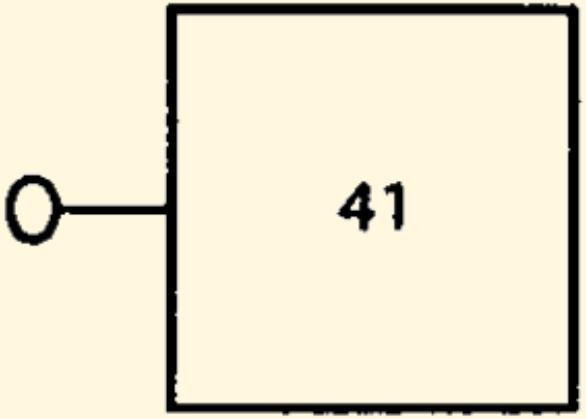
Best routing: Class-Based

XY request, YX response packets



[Abts 2009]

Non-Uniform Caches (NUCA)



Uniform caches

High latency due to wire delay

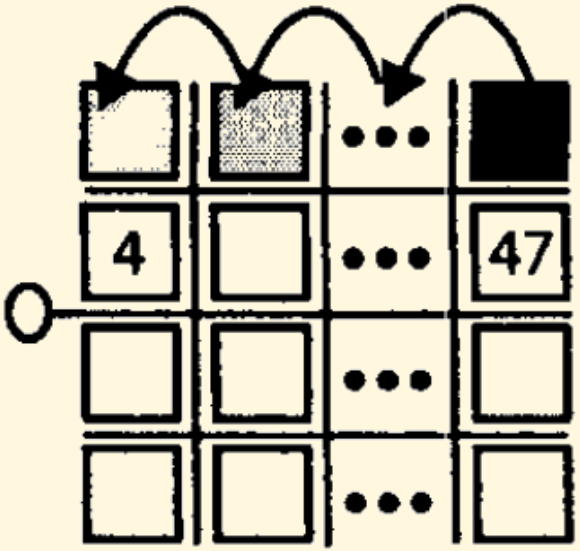
Aggressive sub-banking not enough

Port-limited

Non-Uniform caches

Small, fast banks over a switched network

Good average latency



Challenge: efficient bank partitioning in CMPs

[Kim 2002]

NUCA slicing in CMPs

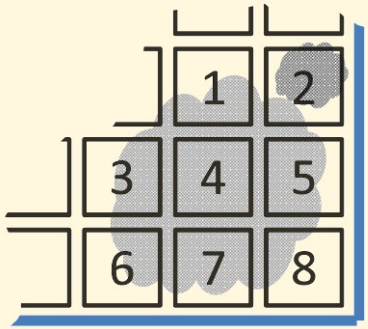
ESP-NUCA [Merino 2010]
Token-based directory
Limited per-core priv slices

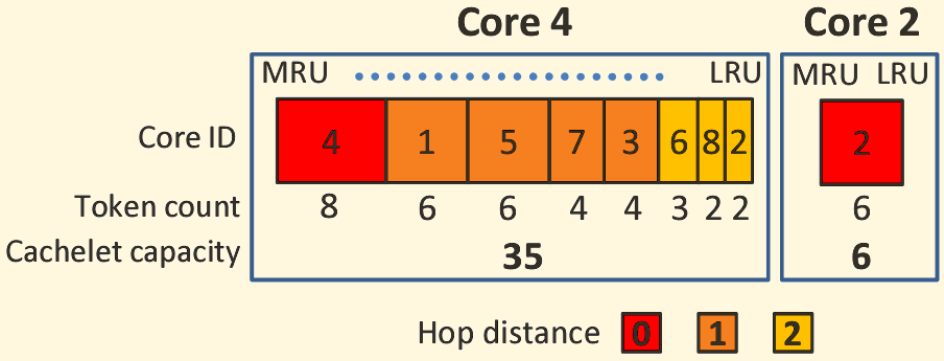
Elastic CC [Herrero 2010]
Address-based split of directory & data

Both: Utility-based spilling of replicas/victims

CloudCache

Utility-based dynamic partitioning
Distance-aware borrowing from neighbors
Address-based distributed directory





[Lee 2011]

OS-level allocation: Slice = Phys. PN % (nr. of slices)

[Cho 2006]

Multicore Scalability

Where is the bottleneck?

Coherence

may be too costly to maintain

&

Heterogeneity

could become too hard to manage

e.g. NUMA

[Baumann 2009]

Communication Models

Coherent shared memory

Hybrid

e.g. scale-out (coherence only among groups of cores) [Lofti-Kamran 2012]

Scratchpad

e.g. local stores in the IBM Cell

Entirely distributed

message-passing across cores

Time to give up coherence?

It may make sense

Cores are already nodes in a network – why not just exchange messages? [Baumann 2009]

Conventional wisdom says coherence cannot scale

but

we better have a very good reason

Most existing code relies on coherence

Plenty of man-years of optimizations [Martin 2012]

Many programmers' brains would have to be rewired

- But my program doesn't scale today...

Is it the algorithm, the implementation, or coherence?

Software bottlenecks often to blame

7 system applications shown to scale when using standard parallel programming techniques [Boyd-Wickizer 2010]

Scalable locks do exist and are just as simple as non-scalable ticket locks (e.g. Linux spin locks) [Boyd-Wickizer 2012]

Too many readers will always cause trouble

Though lockless mechanisms like RCU are an increasingly popular alternative to most Reader-Writer locks [Clements 2012]

It seems coherence will live on

Coherence can scale

Judicious choices can lead to slow growth of traffic, storage, latency and energy with core count [Martin 2012]

Likely to coexist with other models

Research on these issues still at its infancy

Coherence won't solve all problems

Heterogeneity is a challenge

Problems here seem less threatening, but they exist, e.g. management of memory-controller traffic on NUMA systems [Dashti 2013]

Part III

Heterogeneous Architectures

and beyond

Performance and energy efficiency require

Specialization via Heterogeneity

Flexible-core CMPs

~1.5x speedup, ~2x power savings over GP

Granularity of processors determined at runtime

Pose interesting challenge to thread schedulers [Kim 2007]

Greendroid

Synthesis of code segments into "conservation cores"

~No speedup, ~16X energy savings for segments

[Goulding-Hotta 2011]

Accelerator-rich CMPs

~50X speedup, ~20X energy improv.

Still unclear to what extent general-purpose computing could benefit: opportunity cost of integrating accelerators may be prohibitive

[Cong 2012]

Heterogeneity
is not the only way out

Disciplined Approximate Computing

Trade off accuracy for energy [Esmaeilzadeh 2012]

Vision, clustering, etc. don't need 100% accuracy

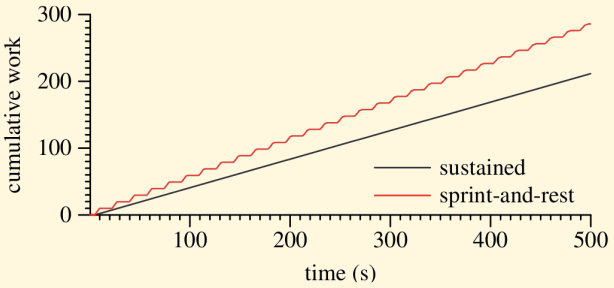
2x energy savings, average error rate 3-10%, peak 80%

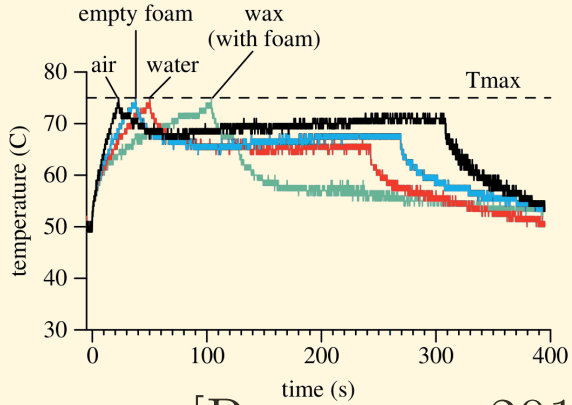
Computational Sprinting

Leverage dark silicon to provide short bursts of intense computation by exploiting thermal capacitance

6x responsiveness gain, 5% less energy

Smart sprint pacing can yield performance improvements





[Raghavan 2013]

Conclusion

Conclusion

In the post-Dennard scaling era,

performance

is determined by energy efficiency

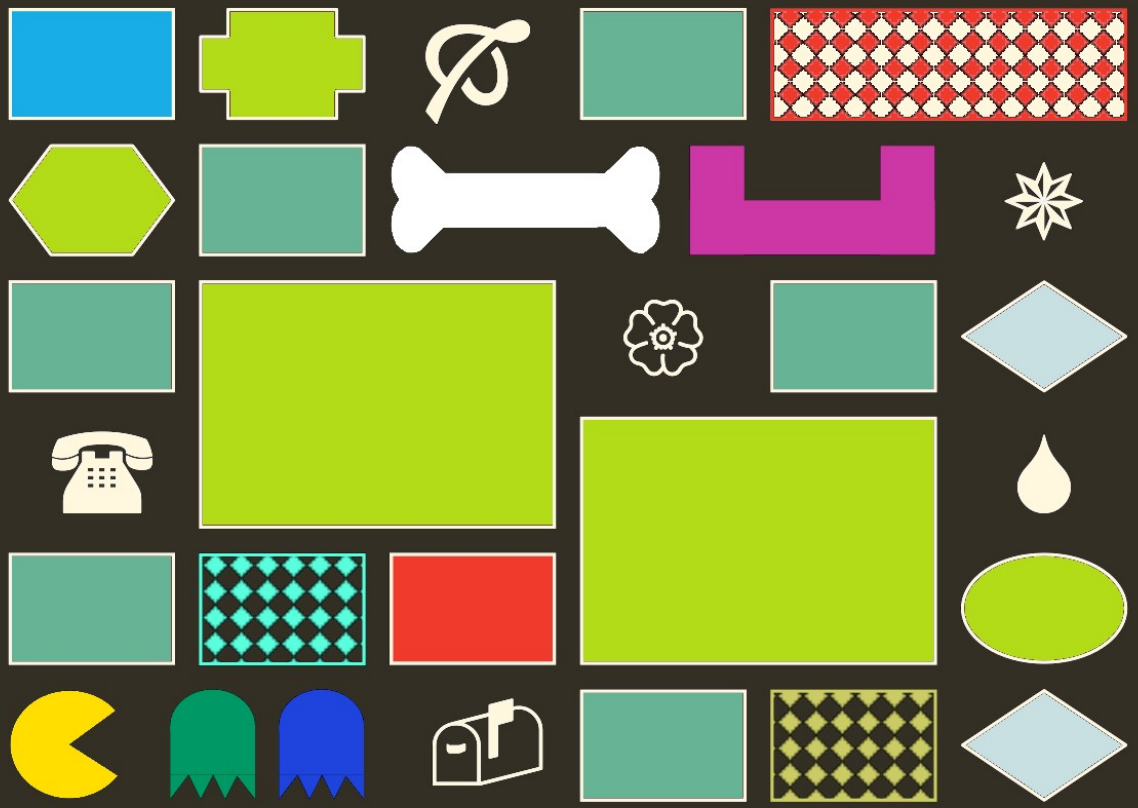
Future computer systems will be

parallel & heterogeneous

Various GPCPUs will coexist with
custom logic, GPGPUs and even FPGAs

[Consortium 2013] [Chung 2010]

Thanks



Frankenchip by Ryan Johnson