

Designing Languages for Designing Hardware

Adrian Sampson

Cornell

How do we harness the power of computing?

HCI

What are the secrets of human intelligence?

ML

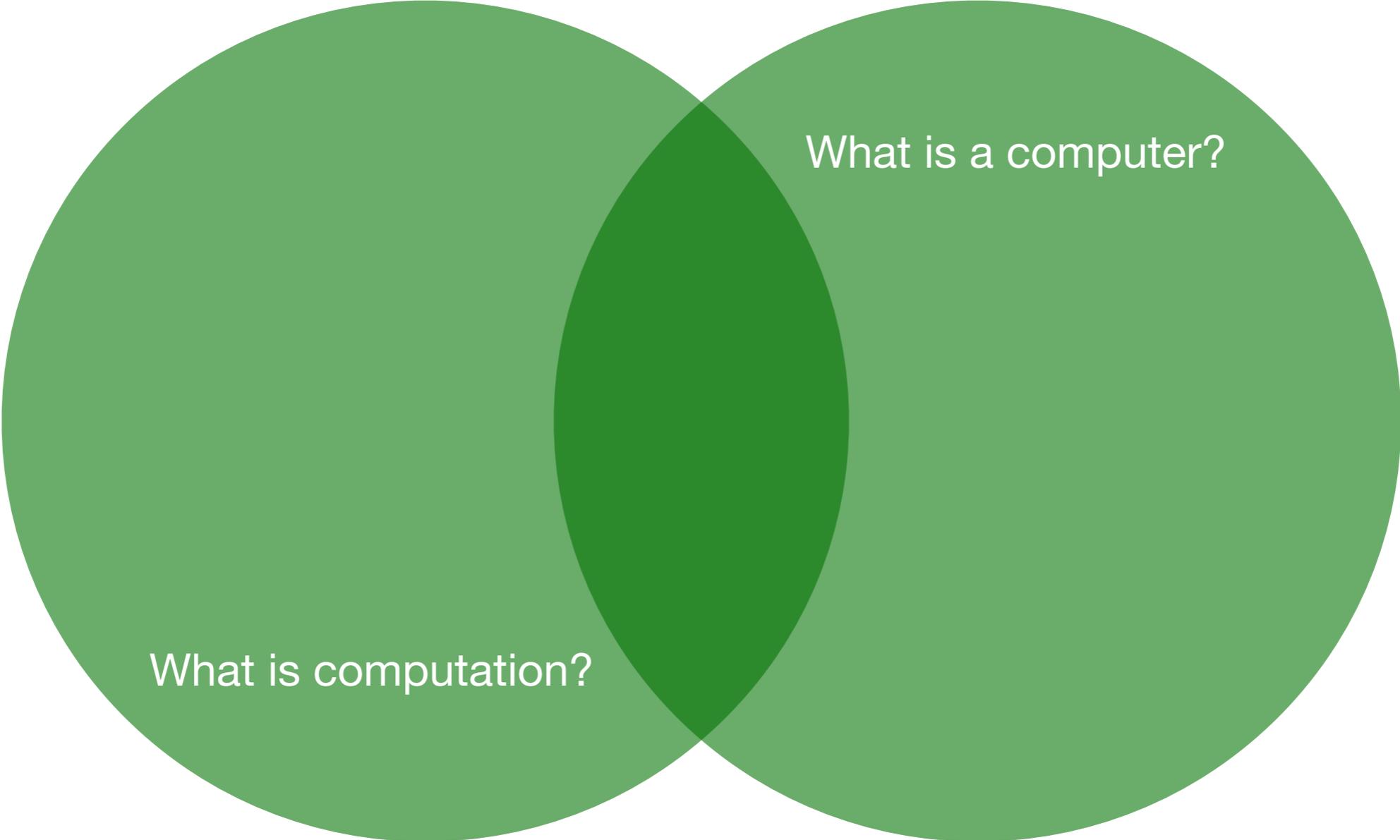
How do computers work?
What is computation, really?

What is computation, really?

PL

What is a computer?

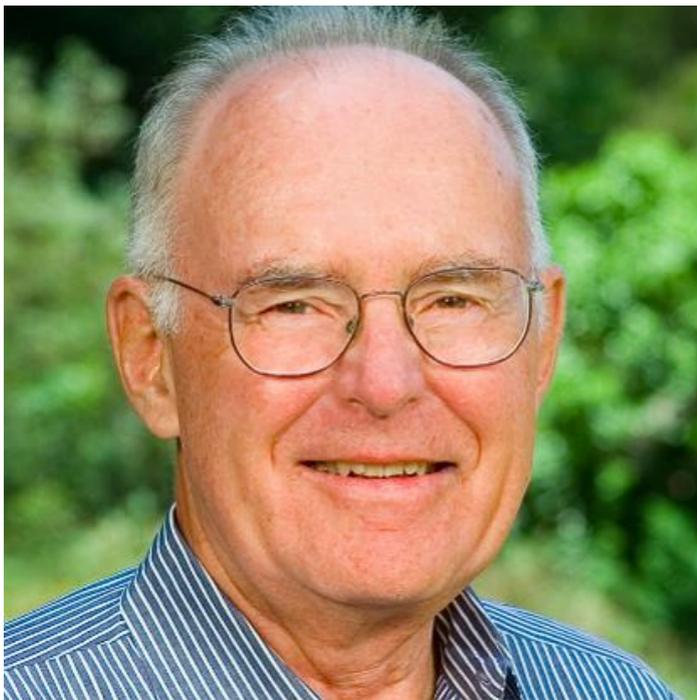
Arch



What is computation?

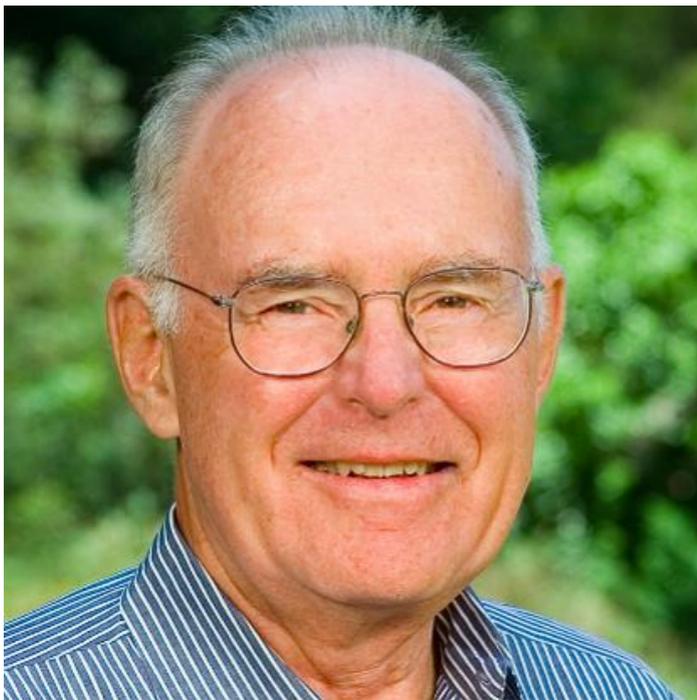
What is a computer?

“The complexity for minimum component costs has increased at a rate of roughly a factor of two per year.”



Gordon Moore
co-founder of Intel
1965

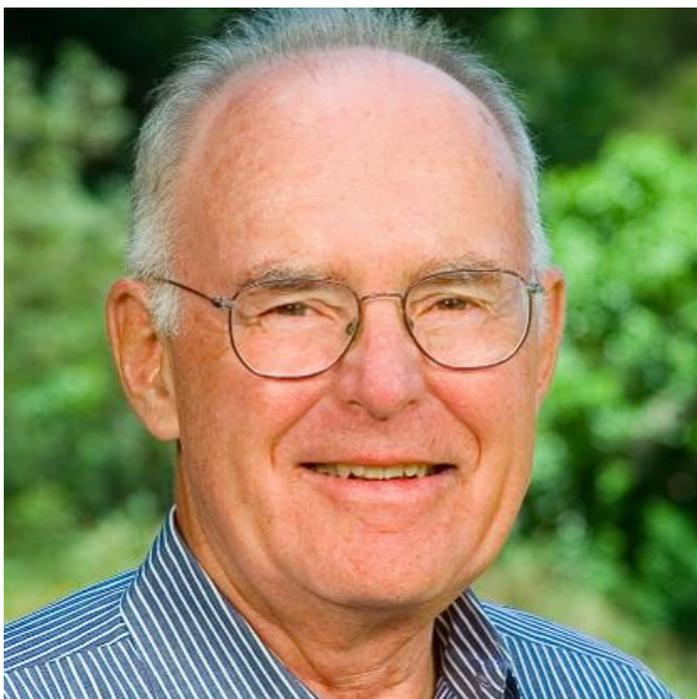
The size of a single transistor decreases by half every 18 months.



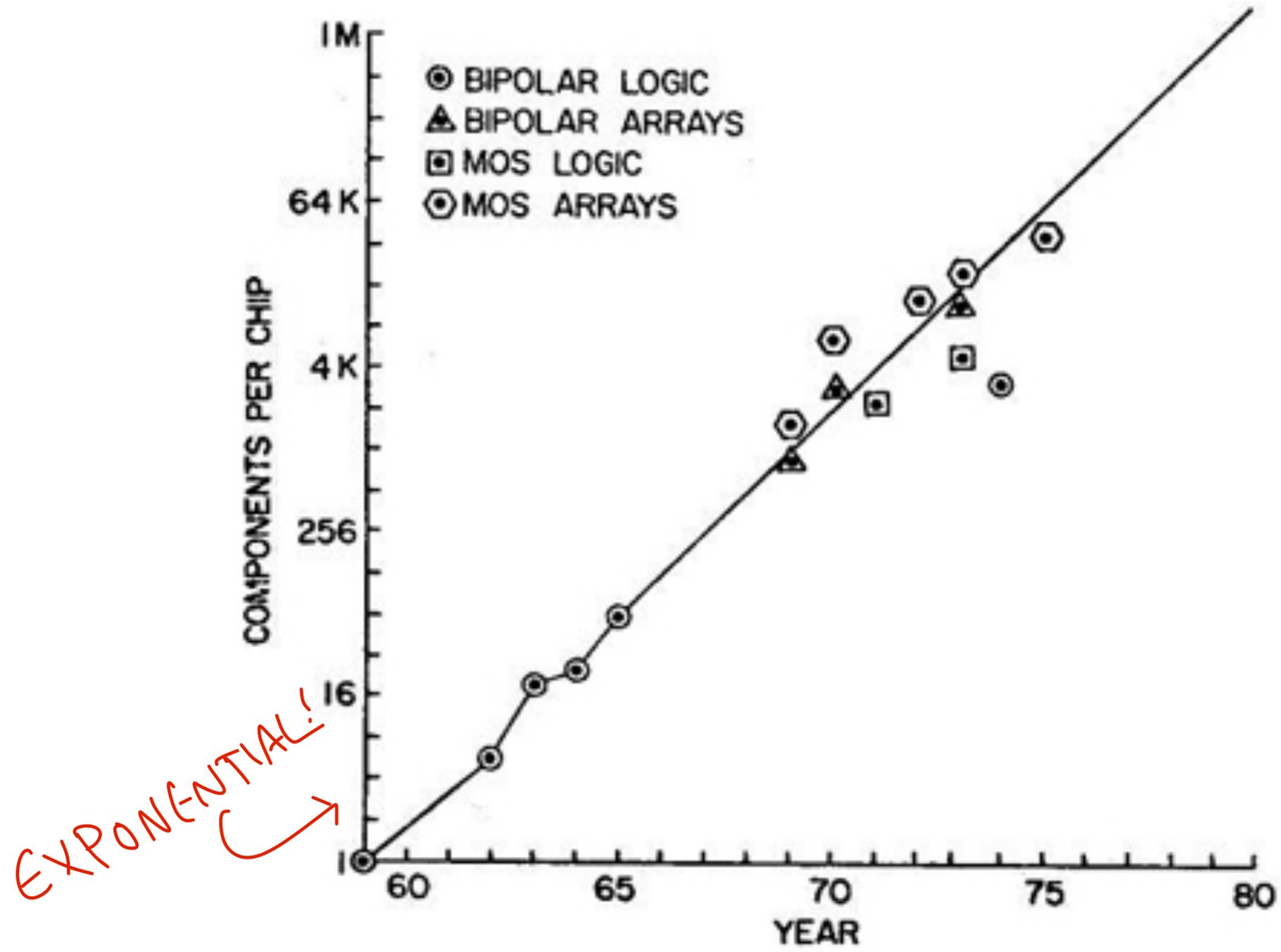
Gordon Moore
co-founder of Intel
1965

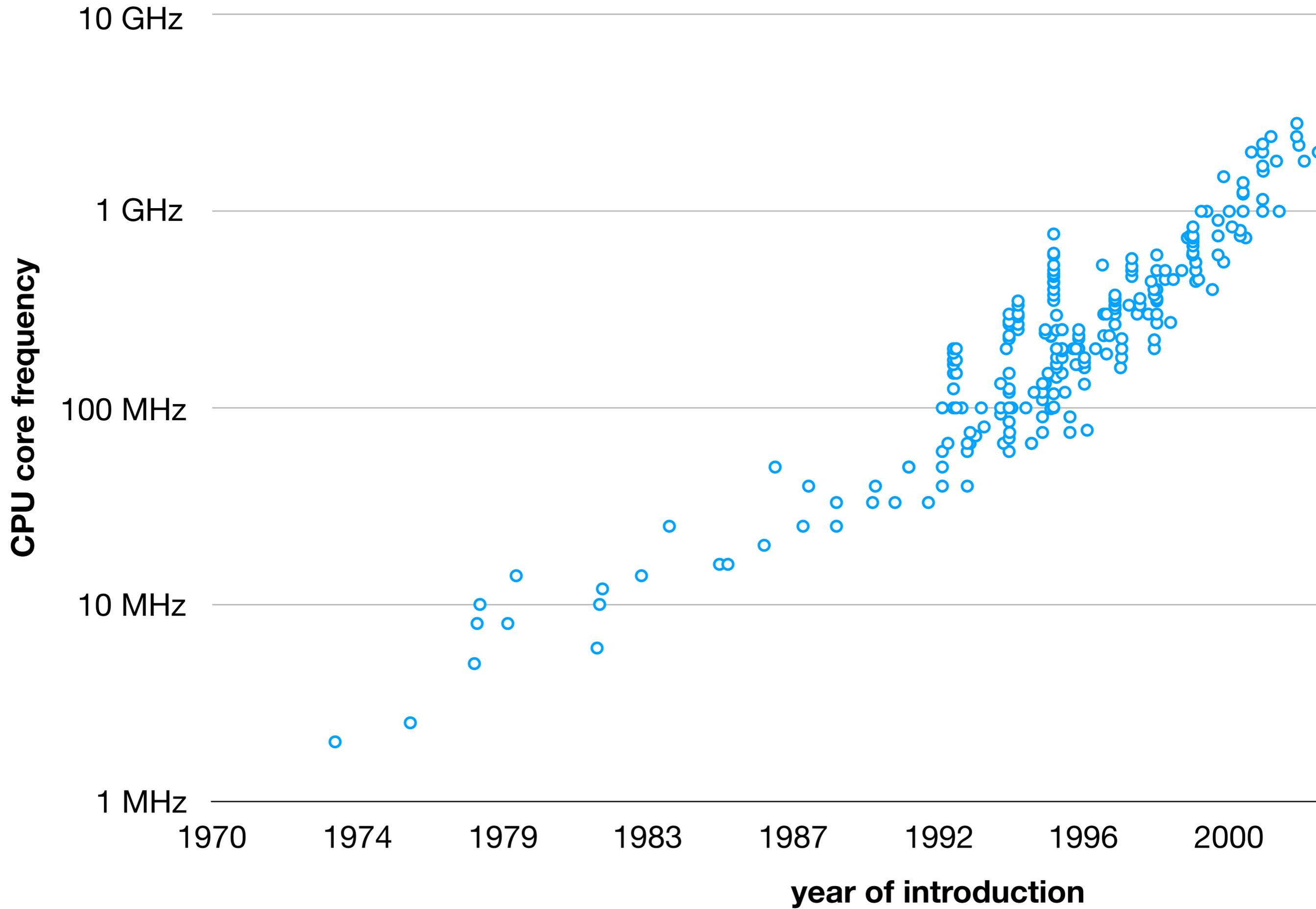
and cost, and power...

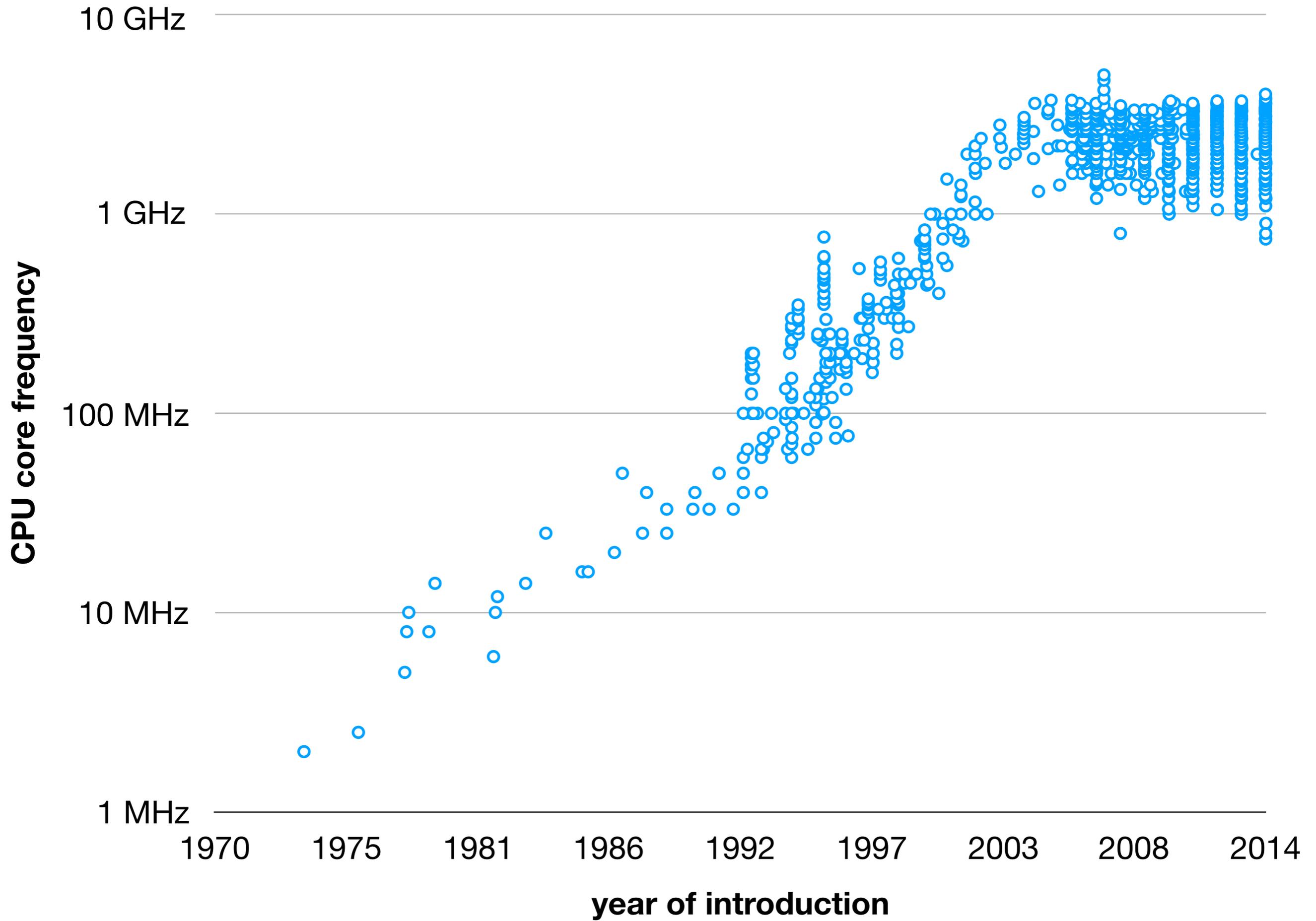
**The size[^] of a single
transistor decreases by
half every 18 months.**



Gordon Moore
co-founder of Intel
1965







free lunch

time
immemorial

2005

2015

exponential
single-threaded
performance
scaling!

(not to scale)

free lunch

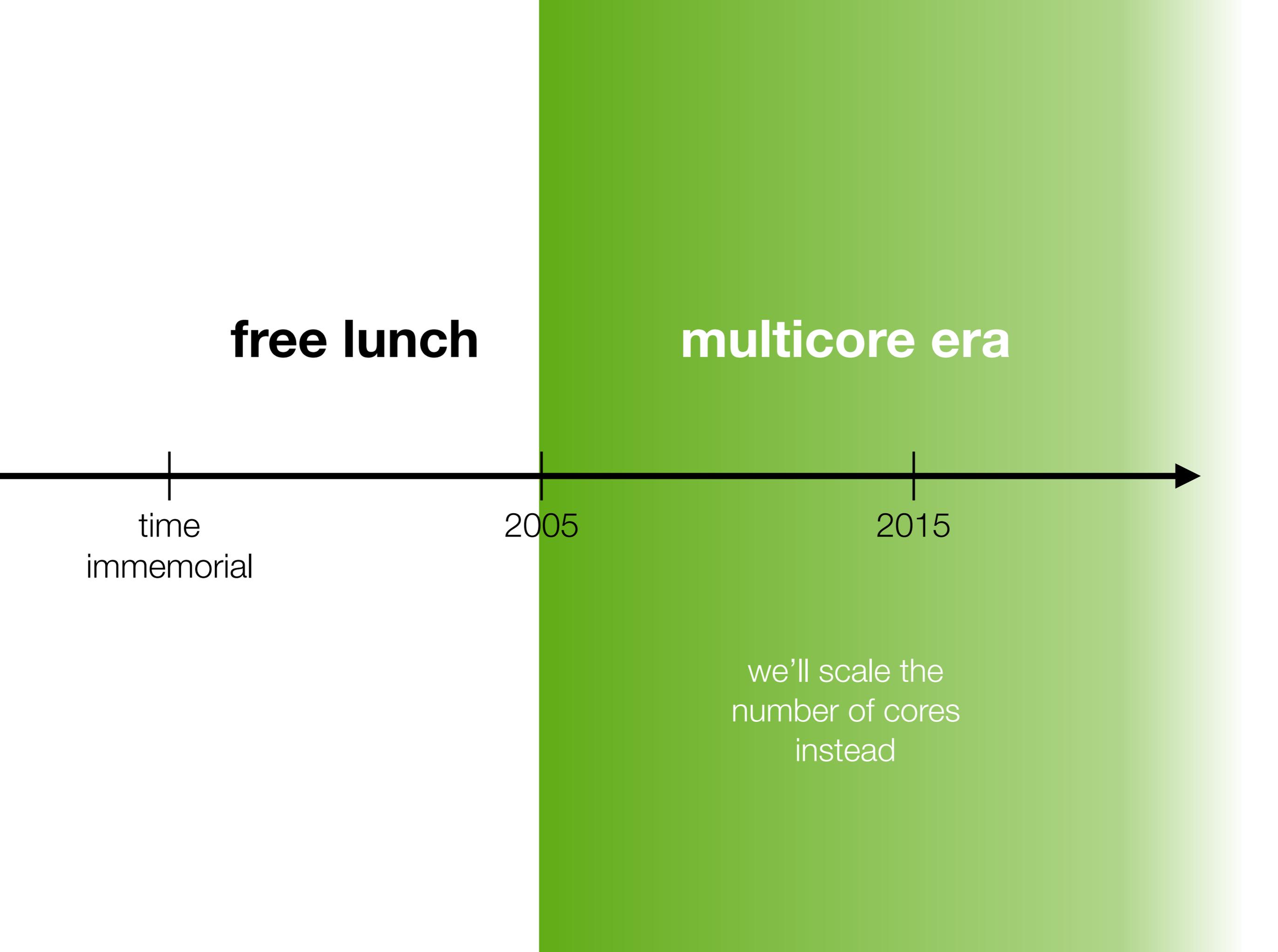
multicore era

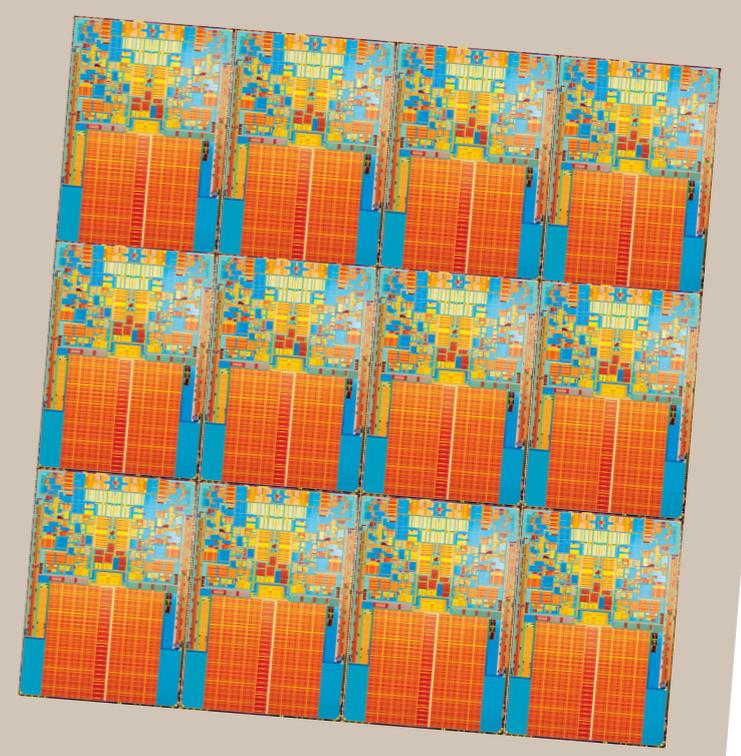
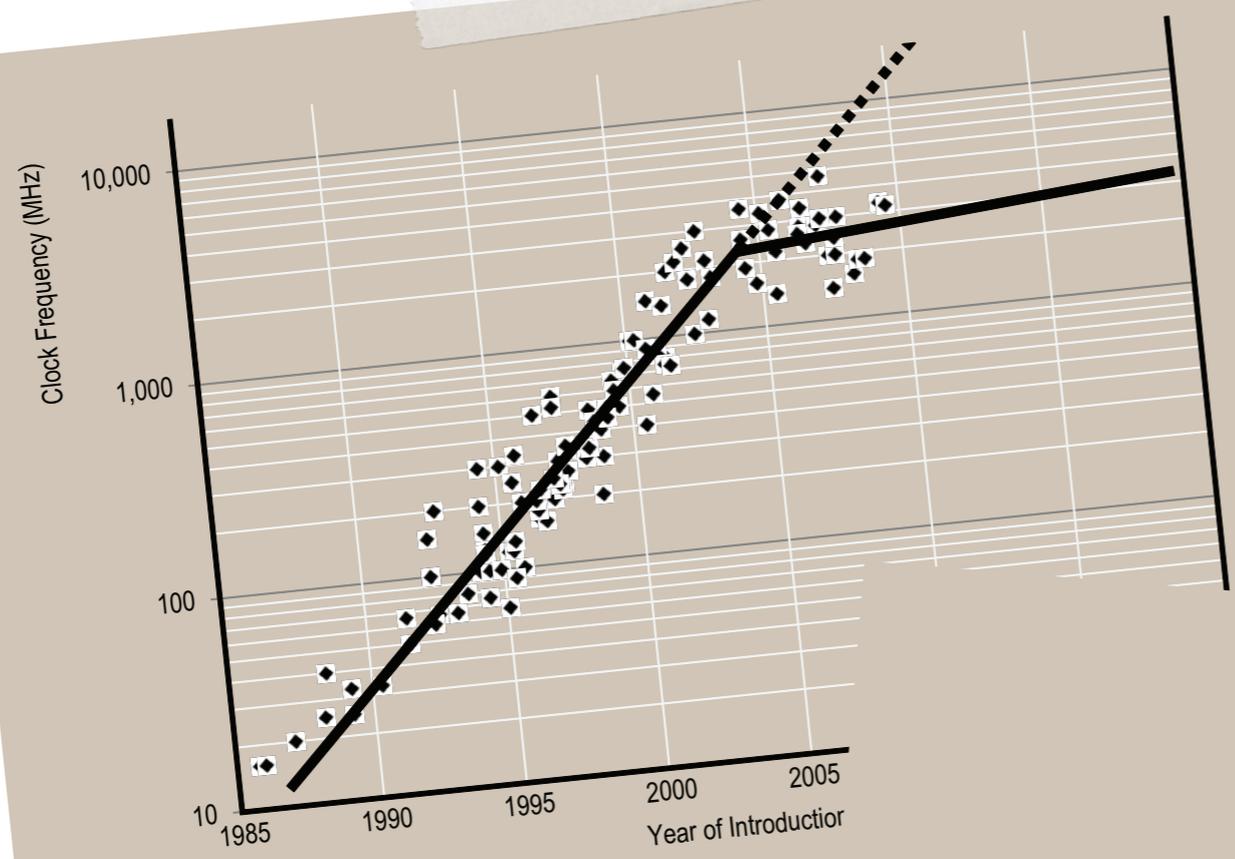
time
immemorial

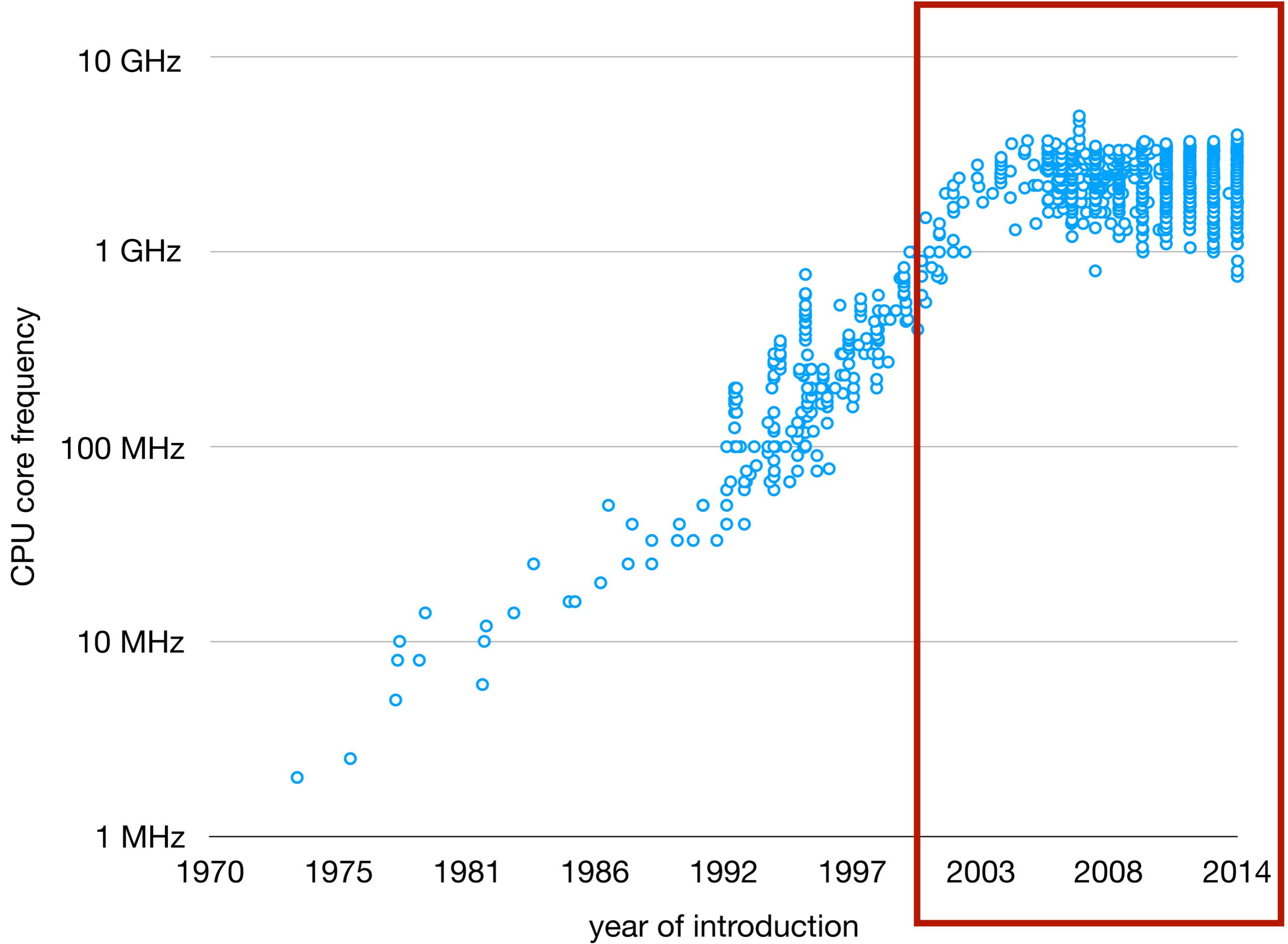
2005

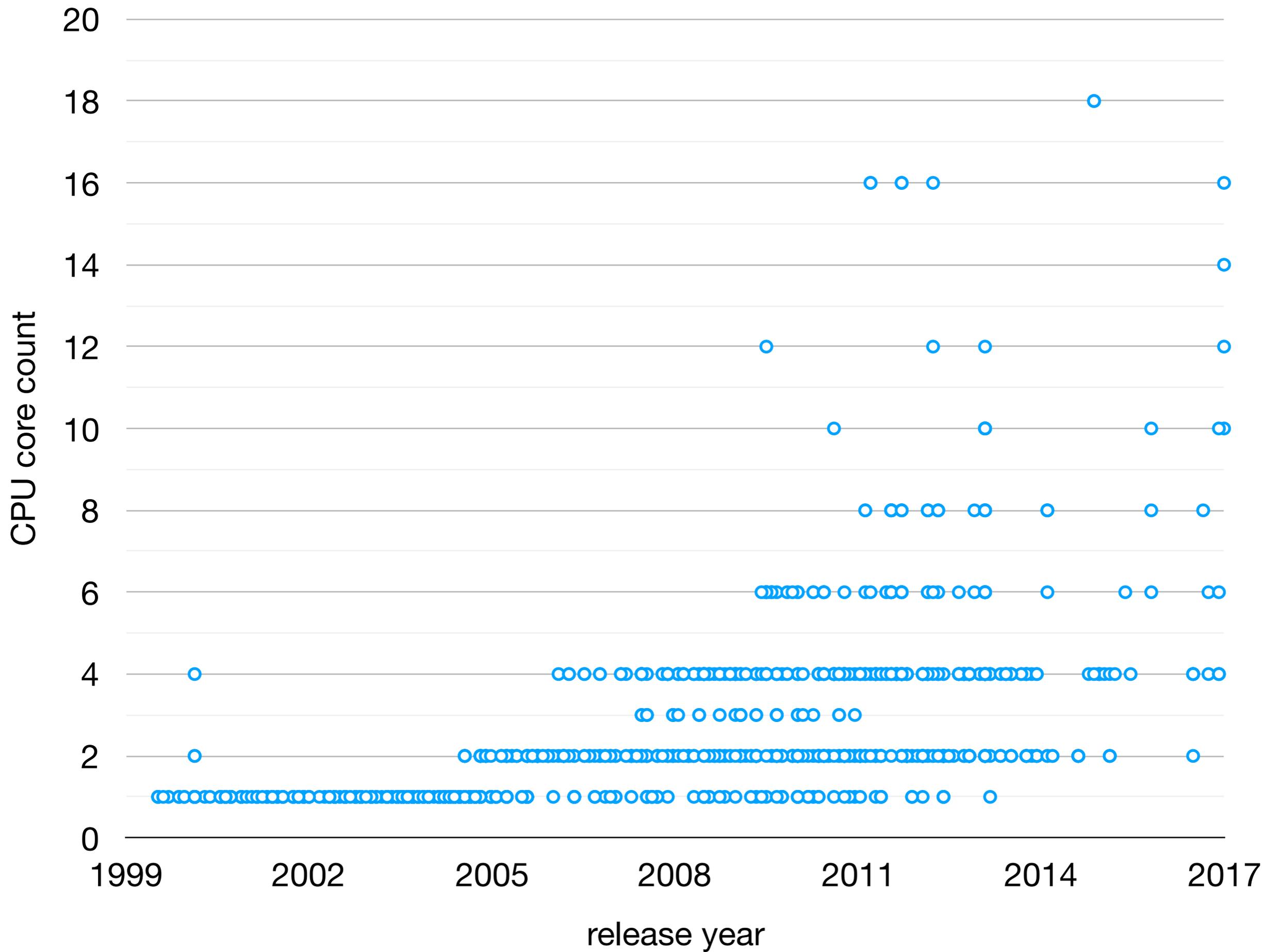
2015

we'll scale the
number of cores
instead









free lunch

multicore era

who knows?

time
immemorial

2005

2015

?

?

?

?

?

**The performance
returns from Moore's
Law ended in 2015!**

**The only way
forward is to trade
off generality for
efficiency!**

A New Golden Age for

**Domain-Specific Hardware,
Enhanced Security, Open Instruction Sets,
Agile Chip Development**

John L. Hennessy and David A. Patterson

CATAPULT



DOUG BURGER



(MICROSOFT,
FORMERLY
UT)



A Reconfigurable Fabric for Accelerating Large-Scale Datacenter Services

Andrew Putnam Adrian M. Caulfield Eric S. Chung Derek Chiou¹
Kypros Constantinides² John Demme³ Hadi Esmaeilzadeh⁴ Jeremy Fowers
Gopi Prashanth Gopal Jan Gray Michael Haselman Scott Hauck⁵ Stephen Heil
Amir Hormati⁶ Joo-Young Kim Sitaram Lanka James Larus⁷ Eric Peterson
Simon Pope Aaron Smith Jason Thong Phillip Yi Xiao Doug Burger

Microsoft

Abstract

Datacenter workloads demand high computational capabilities, flexibility, power efficiency, and low cost. It is challenging to improve all of these factors simultaneously. To advance datacenter capabilities beyond what commodity server designs can provide, we have designed and built a composable, reconfigurable fabric to accelerate portions of large-scale software services. Each instantiation of the fabric consists of a 6x8 2-D torus of high-end Stratix V FPGAs embedded into a half-rack of 48 machines. One FPGA is placed into each server, accessible through PCIe, and wired directly to other FPGAs with pairs of 10 Gb SAS cables.

In this paper, we describe a medium-scale deployment of this fabric on a bed of 1,632 servers, and measure its efficacy in accelerating the Bing web search engine. We describe the requirements and architecture of the system, detail the

desirable to reduce management issues and to provide a consistent platform that applications can rely on. Second, datacenter services evolve extremely rapidly, making non-programmable hardware features impractical. Thus, datacenter providers are faced with a conundrum: they need continued improvements in performance and efficiency, but cannot obtain those improvements from general-purpose systems.

Reconfigurable chips, such as Field Programmable Gate Arrays (FPGAs), offer the potential for flexible acceleration of many workloads. However, as of this writing, FPGAs have not been widely deployed as compute accelerators in either datacenter infrastructure or in client devices. One challenge traditionally associated with FPGAs is the need to fit the accelerated function into the available reconfigurable area. One could virtualize the FPGA by reconfiguring it at run-time to support more functions than could fit into a single device. However, current reconfiguration times for standard FPGAs

23
AUTHORS!

RTL

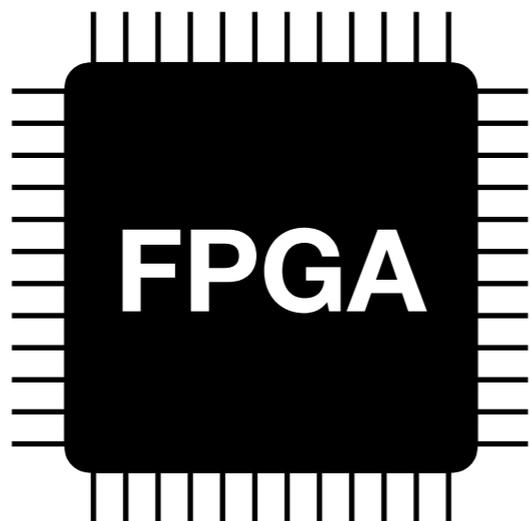
register-transfer level

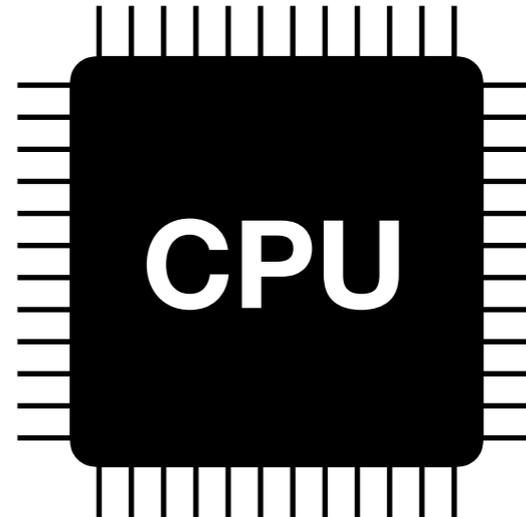
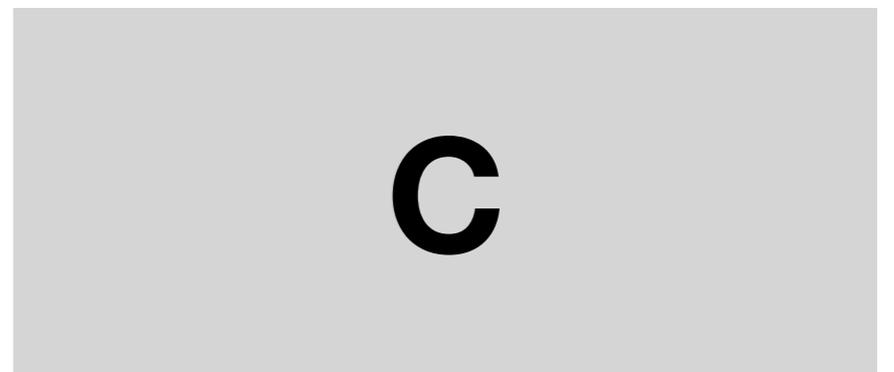
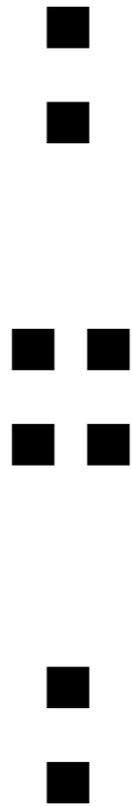
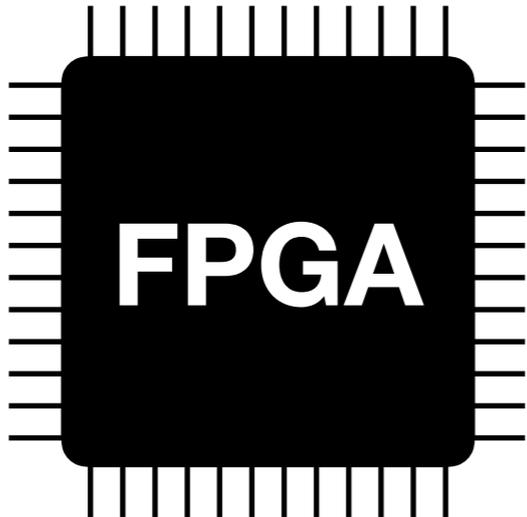
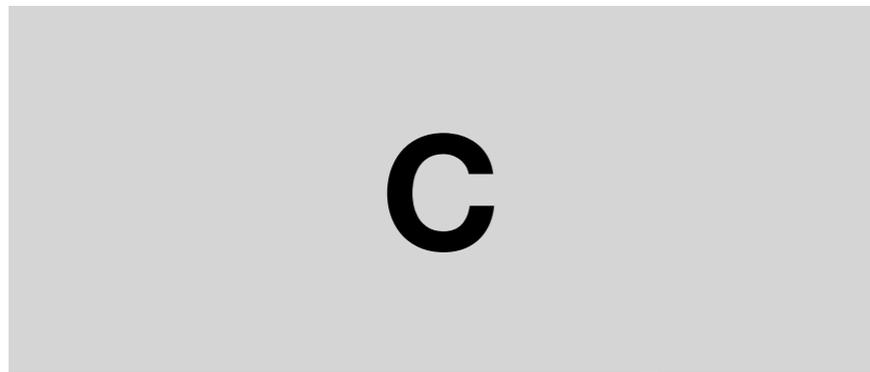
Verilog

VHDL

Bluespec

Chisel





C



High-Level Synthesis

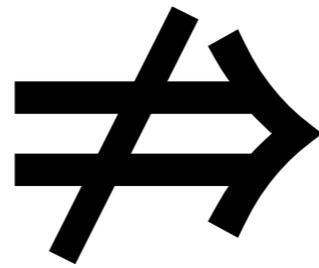
RTL

register-transfer level

image and video processing, financial analytics, bioinformatics, and scientific computing applications. Since RTL programming in VHDL or Verilog is unacceptable to most application software developers, it is essential to provide a highly automated compilation/synthesis flow from C/C++ to FPGAs.

As a result a growing number of FPGA designs are

**Verilog
is unacceptable**



**we must program
FPGAs in C**

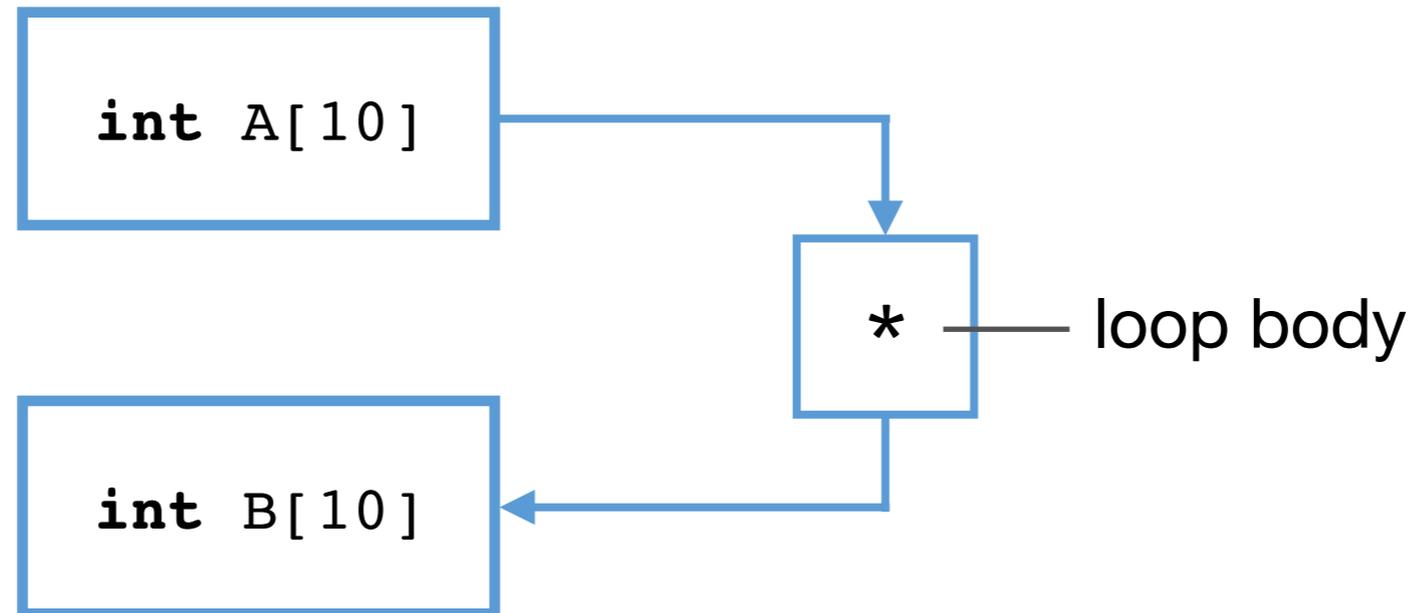
HLS

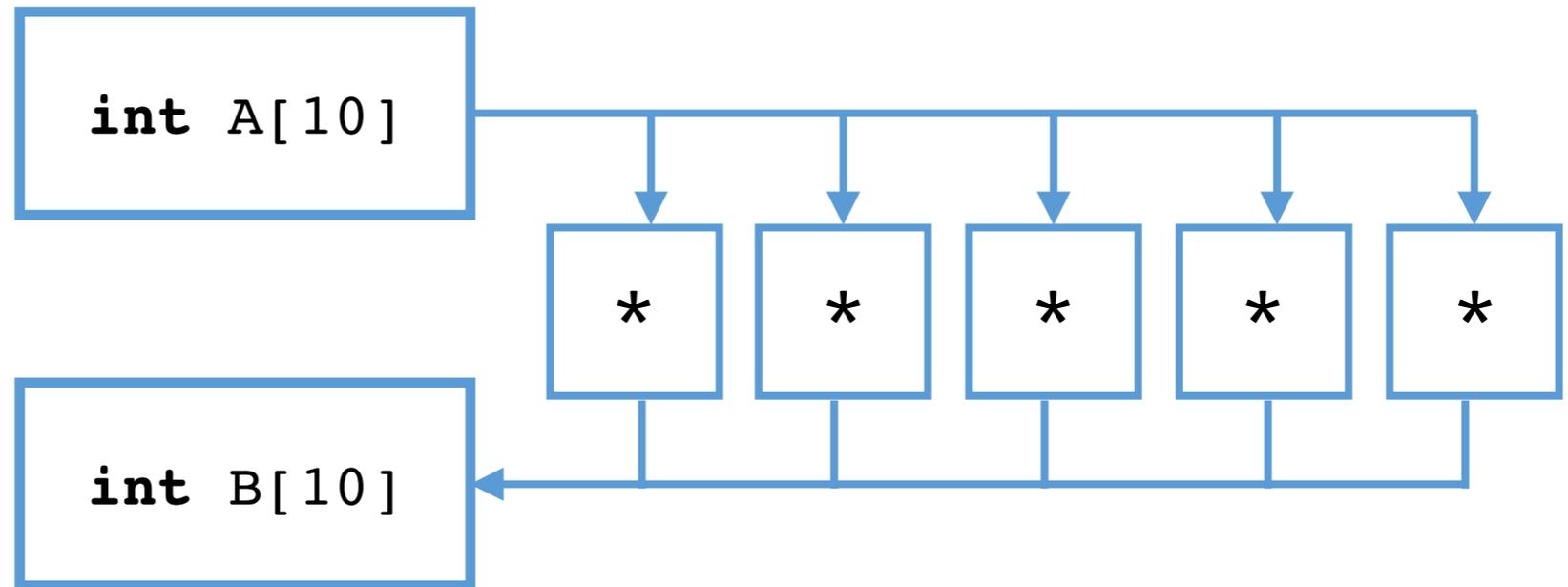
An enormous series of *ad hoc* consistency checks, hacks, and workarounds to compile some C programs to Verilog.

Seashell

A new language for hardware accelerator design with a **type system** that defines which programs are realizable on FPGAs.

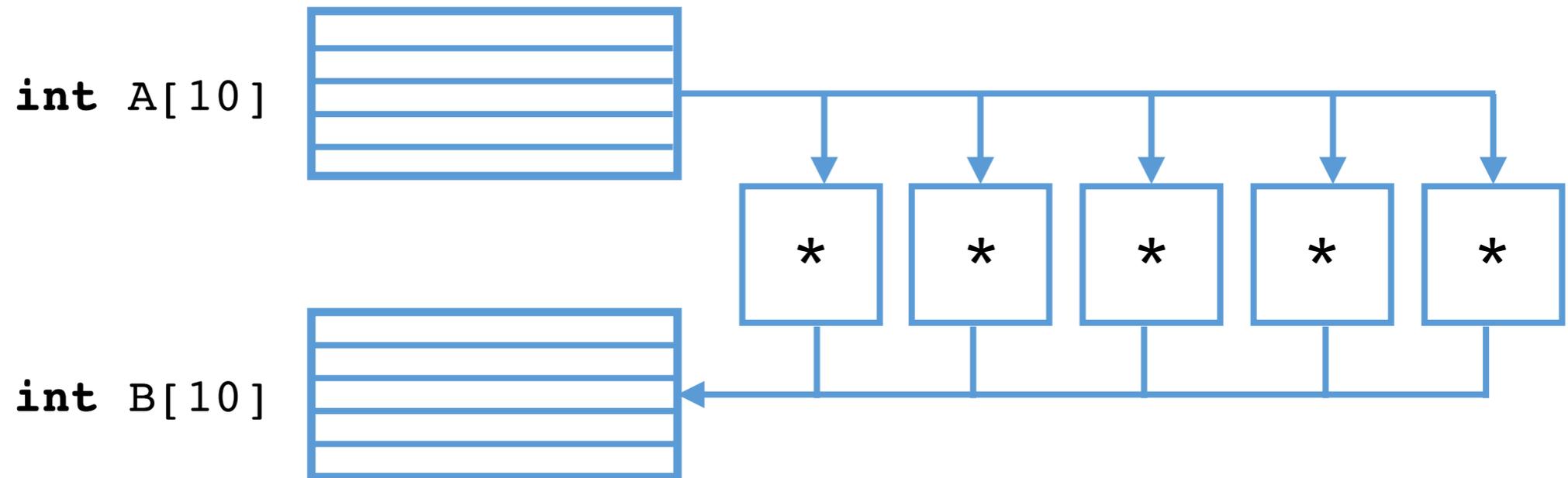
```
int A[10];  
int B[10];  
for (int i = 0; i < 10; i++) {  
    int x = A[i];  
    int y = x * 5;  
    B[i] = y;  
}
```





```
#pragma HLS ARRAY_PARTITION variable=A factor=5
#pragma HLS ARRAY_PARTITION variable=B factor=5

int A[10];
int B[10];
for (int i = 0; i < 10; i++) {
    #pragma HLS UNROLL factor=5
    int x = A[i];
    int y = x * 5;
    B[i] = y;
}
```



```
#pragma HLS ARRAY_PARTITION variable=A factor=5
#pragma HLS ARRAY_PARTITION variable=B factor=5
int A[10];
int B[10];
for (int i = 0; i < 10; i++) {
    #pragma HLS UNROLL factor=5
    int x = A[i];
    int y = x * 5;
    B[i] = y;
}
```

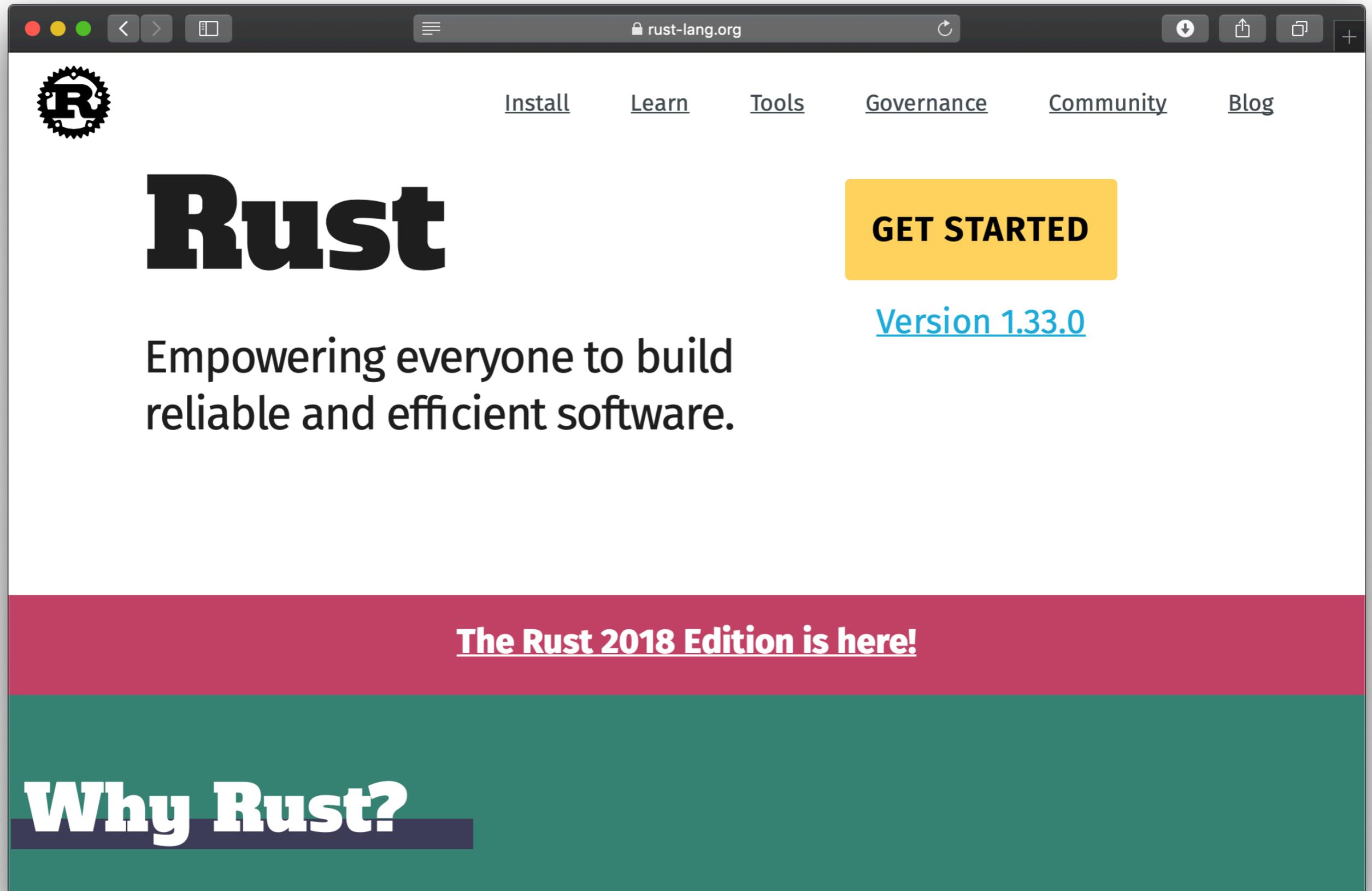
Memory types

```
memory A : int[10];  
for (...) {  
    access A[i];  
    access A[i+1];  
}
```

Affine memory types

```
memory A : int[10];  
for (...) {  
    access A[i];  
    access A[i+1]; ← error: A already used in this context  
}
```

Affine types and **linear types**, as made famous recently by **Rust**.

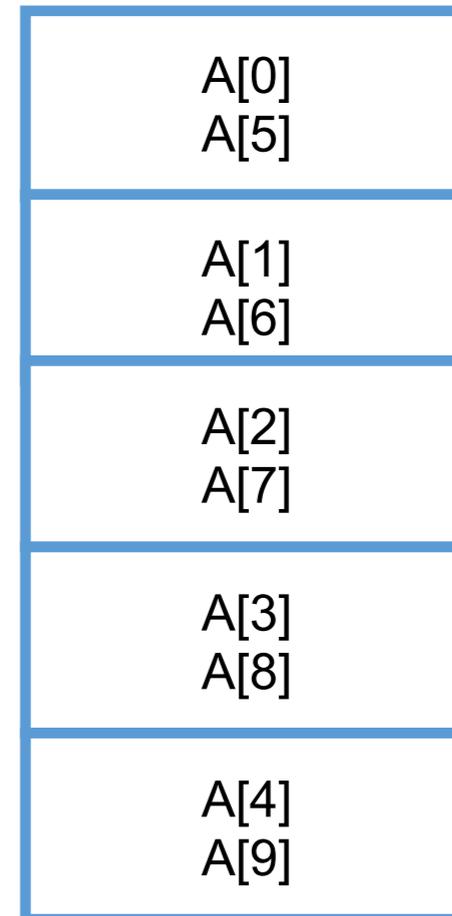


The image shows a screenshot of the Rust website homepage. At the top, there is a navigation bar with links for [Install](#), [Learn](#), [Tools](#), [Governance](#), [Community](#), and [Blog](#). The Rust logo is in the top left corner. The main heading is "Rust" in a large, bold, black font. Below it, the tagline reads "Empowering everyone to build reliable and efficient software." To the right of the tagline is a yellow button that says "GET STARTED" and a link for "[Version 1.33.0](#)". At the bottom of the page, there is a dark red banner with the text "[The Rust 2018 Edition is here!](#)" and a dark green banner with the text "Why Rust?" in white.

Why Rust?

Banked memory types

```
memory bank(5) A : int[10];
```



Banked memory types

```
memory bank(5) A : int[10];  
for (let i in 0..1) {  
  access A[0][i];  
  access A[1][i];  
  access A[2][i];  
  access A[3][i];  
  access A[4][i];  
}
```

// one access to each
A[j] allowed here

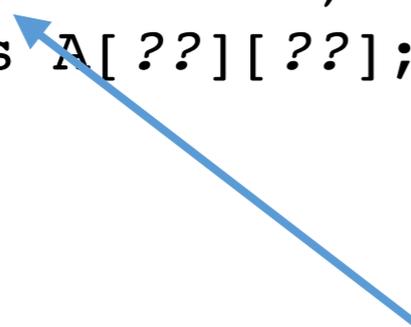
STATIC

DYNAMIC

A[0][0] A[0][1]
A[1][0] A[1][1]
A[2][0] A[2][1]
A[3][0] A[3][1]
A[4][0] A[4][1]

Hybrid indices for unrolling

```
memory bank(5) A : int[10];  
for (let i in 0..9) unroll 5 {  
  access A[??][??];  
}
```



$i : \text{idx}\langle 0..5, 0..2 \rangle$

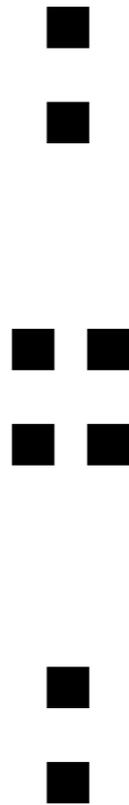
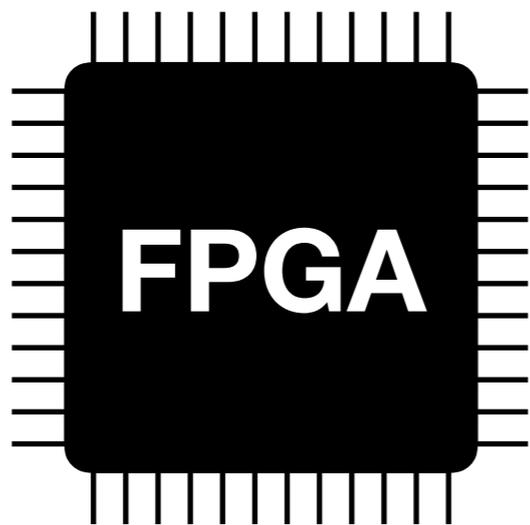
A pair of a **static index** from 0 through 4 and a **dynamic index** that's either 0 or 1.

Seashell



RTL

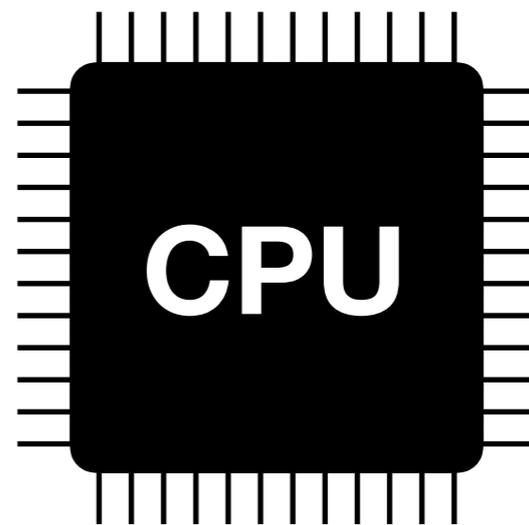
register-transfer level



C



Assembly



github.com/cucapra/seashell

The screenshot shows the GitHub repository page for `cucapra/seashell`. The page includes a navigation bar with links for Pull requests, Issues, Marketplace, and Explore. The repository name is `cucapra / seashell`, with 2 Unwatch, 10 Star, and 2 Fork actions. The repository description is "A typed programming language for safe high-level synthesis" with a link to <https://capra.cs.cornell.edu/fuse>. The repository statistics show 1,006 commits, 7 branches, 0 releases, 1 environment, 5 contributors, and MIT license. The repository is currently on the `master` branch. The commit history shows the latest commit by `tedbauer` and `rachitnigam` adding vim syntax highlighting support (#92) 12 hours ago. The file list includes `.circleci`, `buildbot`, `docs`, `examples`, `notes`, `paper`, `project`, and `src`.

github.com

Search or jump to... Pull requests Issues Marketplace Explore

cucapra / seashell Unwatch 2 Star 10 Fork 2

Code Issues 22 Pull requests 8 Projects 0 Wiki Insights Settings

A typed programming language for safe high-level synthesis <https://capra.cs.cornell.edu/fuse> Edit

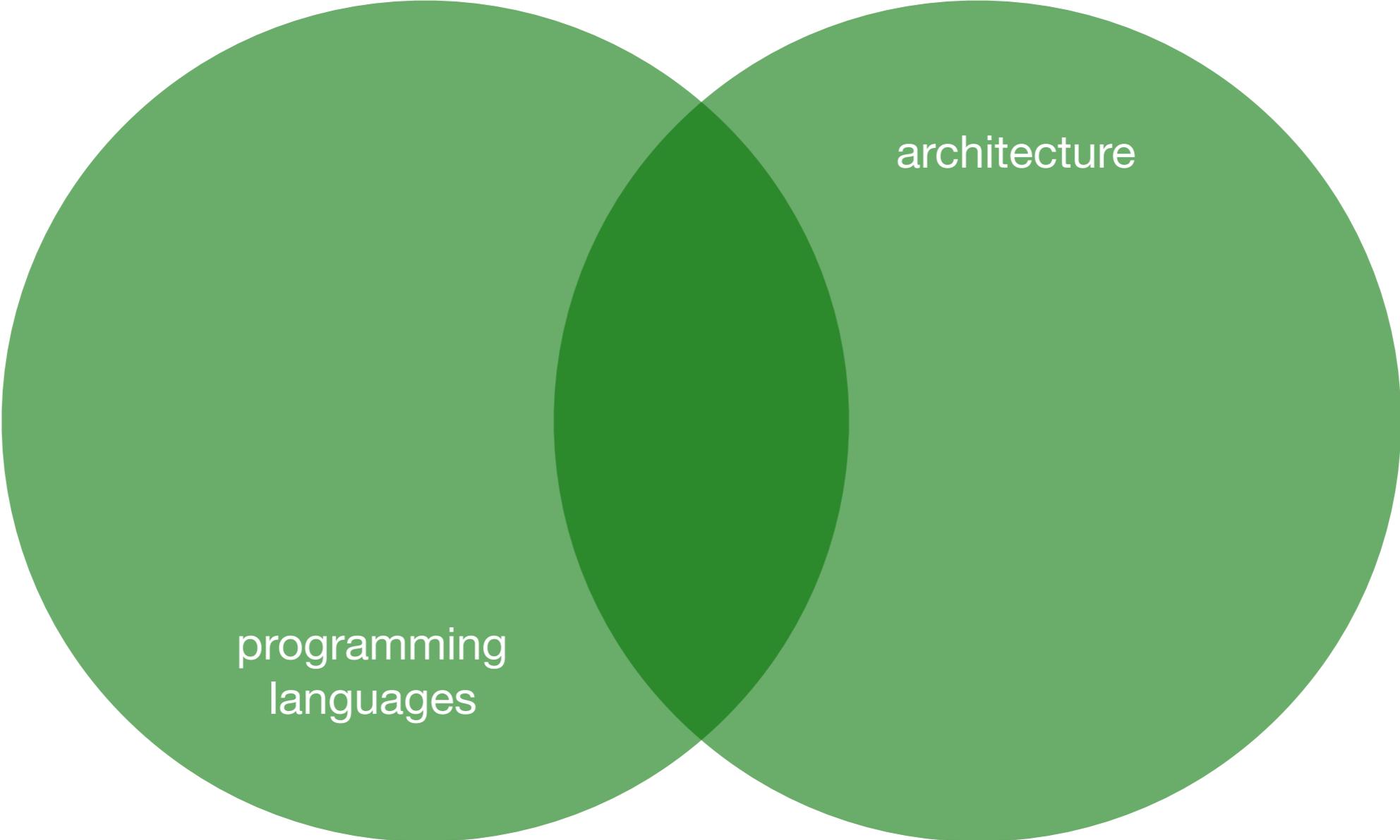
Manage topics

1,006 commits 7 branches 0 releases 1 environment 5 contributors MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

tedbauer and rachitnigam Add vim syntax highlighting support (#92) Latest commit 856f704 12 hours ago

.circleci	update circleci	23 days ago
buildbot	changes to buildbot and Dockerfile	23 days ago
docs	Create new docs website.	22 hours ago
examples	Stencil support files	a day ago
notes	rename docs/ to notes/	22 hours ago
paper	define fuse syntax and GeMM for sec 2	20 days ago
project	add project assembly dep	23 days ago
src	adding TSizedInt rule for consumeBanks (#88)	20 hours ago



programming
languages

architecture

