Computer Organization and Performance – some visual communication

Lecture 2

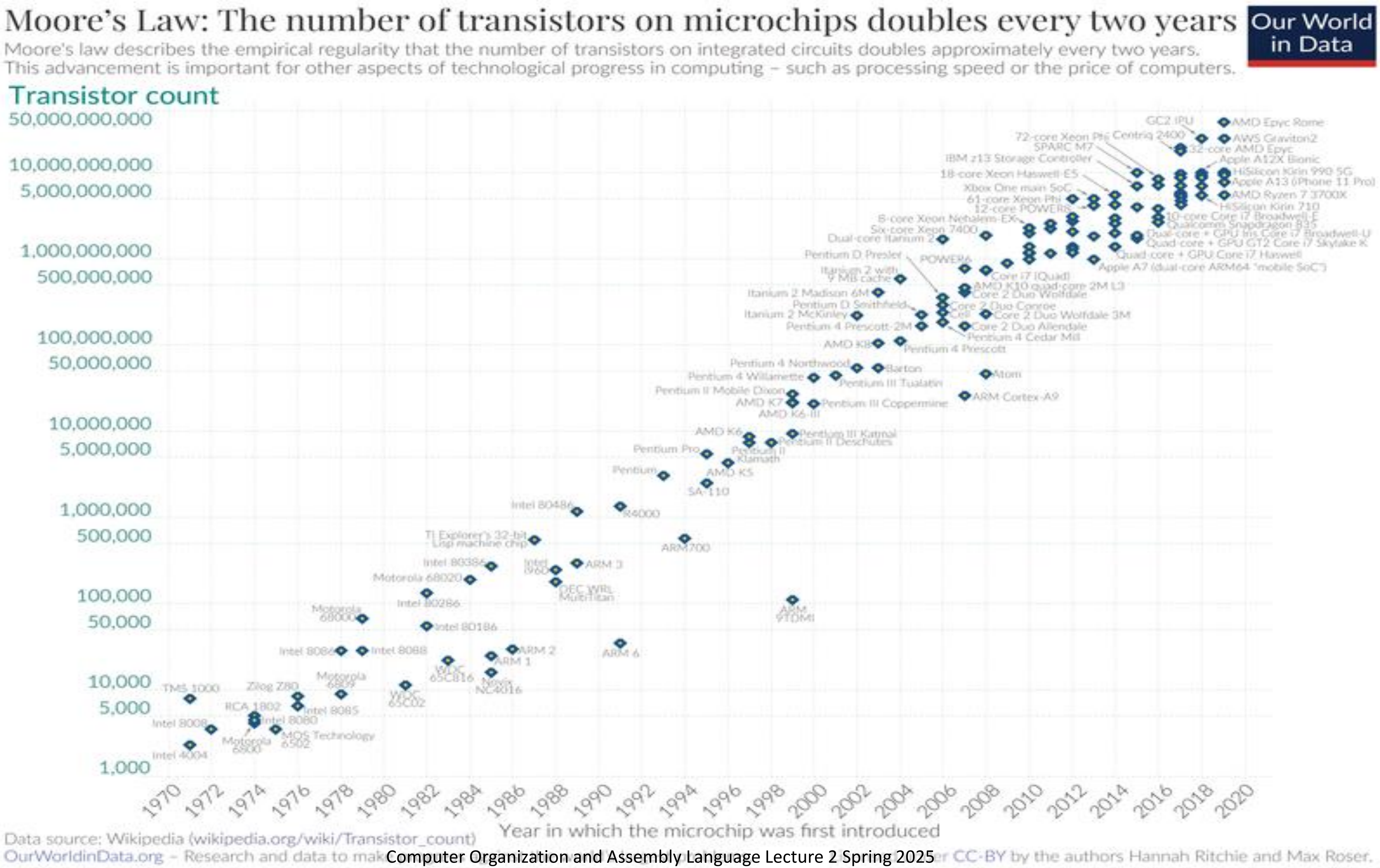
CS/EE 320 Computer Organization and Assembly Language

Dr. Shahid Masud

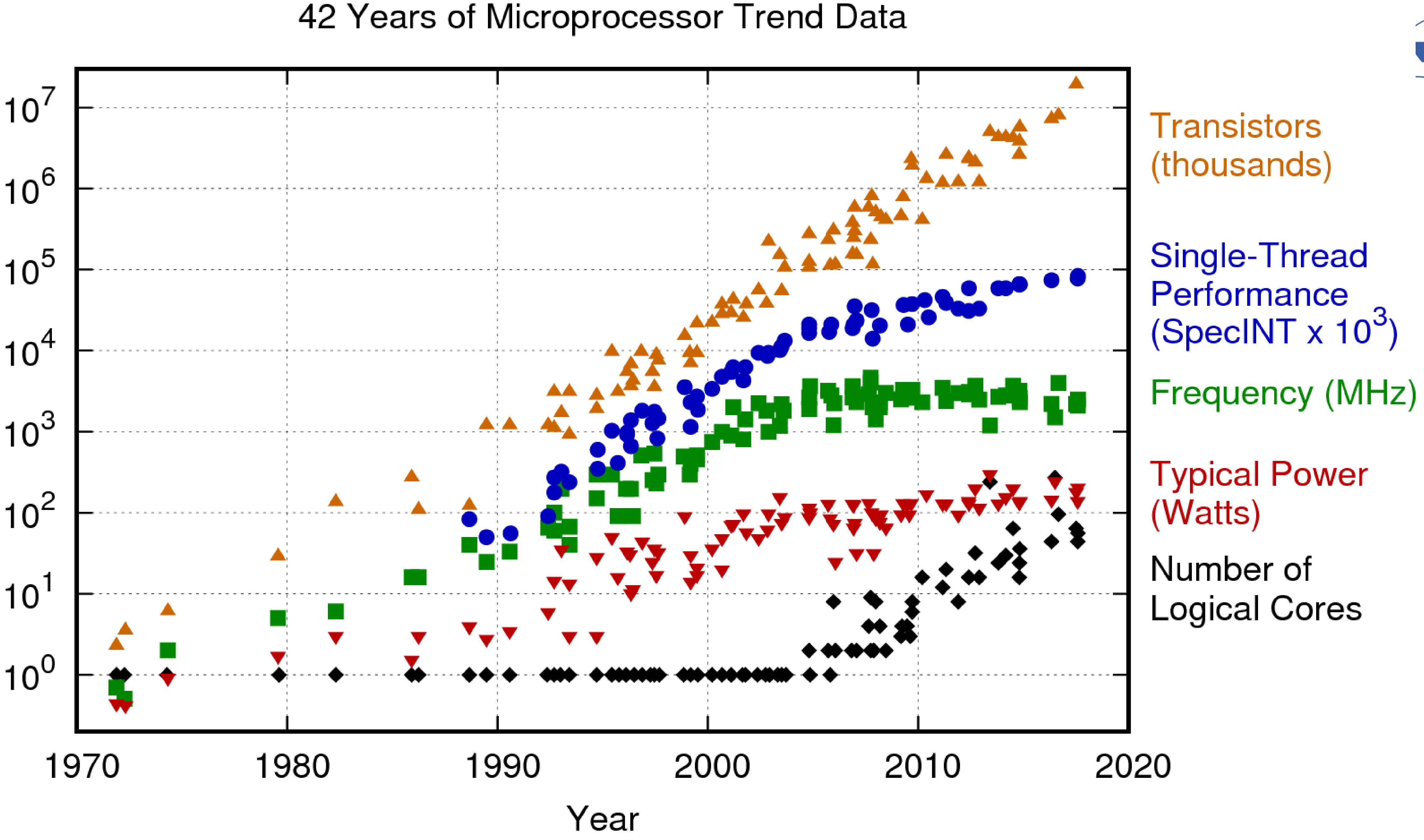
Topics

- Computer Performance – in light of
  - Power Wall
  - Complexity Wall
  - Moore's Law
  - Dennard Scaling
- Computing ideas in post-PC era
- What is meant by the Hardware / Software Interface
- Calculate Power dissipation
- Notion of Computer Performance

Power Wall, and Complexity Wall

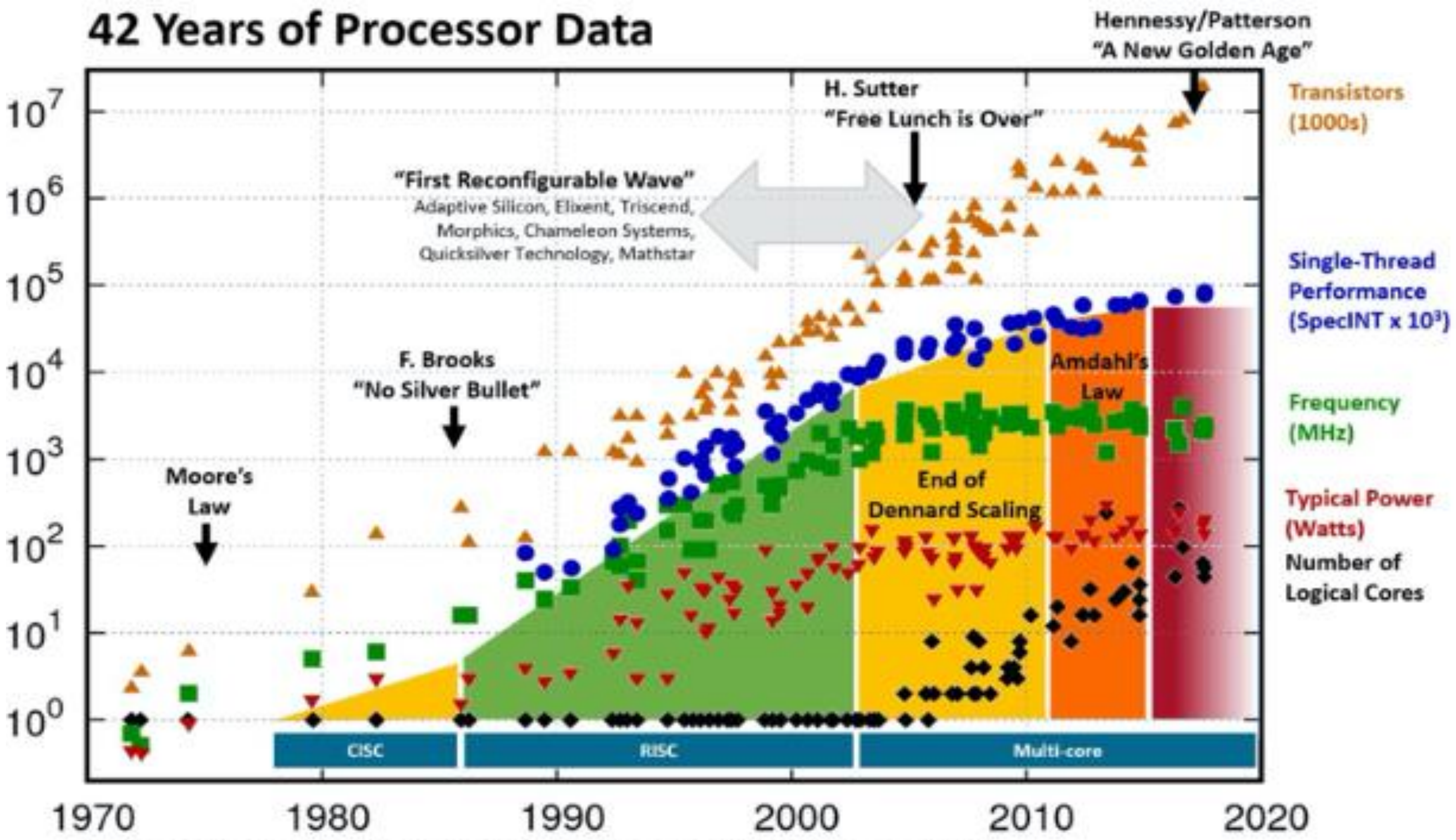


Moore's Law: The number of transistors on microchips doubles every two years
Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years.
This advancement is important for other aspects of technological progress in computing – such as processing speed or the price of computers.
Our World in Data
Transistor count
50,000,000,000
10,000,000,000
5,000,000,000
1,000,000,000
500,000,000
100,000,000
50,000,000
10,000,000
5,000,000
1,000,000
500,000
100,000
50,000
10,000
5,000
1,000
1970
1972
1974
1976
1978
1980
1982
1984
1986
1988
1990
1992
1994
1996
1998
2000
2002
2004
2006
2008
2010
2012
2014
2016
2018
2020
Year in which the microchip was first introduced
AMD Epyc Rome
AWS Graviton2
GC2 IPU
Centriq 2400
72-core Xeon Phi
SPARC M7
32-core AMD Epyc
IBM z13 Storage Controller
Apple A12X Bionic
18-core Xeon Haswell-E5
HiSilicon Kirin 990 5G
Apple A13 (iPhone 11 Pro)
Xbox One main SoC
AMD Ryzen 7 3700X
61-core Xeon Phi
HiSilicon Kirin 710
12-core POWER8
10-core Core i7 Broadwell-E
8-core Xeon Nehalem-EX
Qualcomm Snapdragon 835
Six-core Xeon 7400
Dual-core + GPU Iris Core i7 Broadwell-U
Dual-core Itanium 2
Quad-core + GPU GT2 Core i7 Skylake K
Pentium D Presler
POWER6
Quad-core + GPU Core i7 Haswell
Apple A7 (dual-core ARM64 "mobile SoC")
Itanium 2 with 9 MB cache
Core i7 (Quad)
AMD K10 quad-core 2M L3
Itanium 2 Madison 6M
Core 2 Duo Wolfdale
Pentium D Smithfield
Core 2 Duo Conroe
Itanium 2 McKinley
Cell
Core 2 Duo Wolfdale 3M
Pentium 4 Prescott-2M
Core 2 Duo Allendale
Pentium 4 Cedar Mill
AMD K8
Pentium 4 Prescott
Pentium 4 Northwood
Barton
Atom
Pentium 4 Willamette
Pentium III Tualatin
Pentium II Mobile Dixon
AMD K7
Pentium III Coppermine
ARM Cortex-A9
AMD K6-III
AMD K6
Pentium III Katmai
Pentium II Deschutes
Pentium Pro
Pentium II Klamath
Pentium
AMD K5
SA-110
Intel 80486
R4000
TI Explorer's 32-bit Lisp machine chip
ARM700
Intel 80386
Intel i960
ARM 3
Motorola 68020
DEC WRL MultiTitan
Intel 80286
ARM 9TDMI
Motorola 68000
Intel 80186
Intel 8086
Intel 8088
ARM 2
ARM 1
ARM 6
WDC 65C816
Novix NC4016
TMS 1000
Zilog Z80
Motorola 6809
WDC 65C02
RCA 1802
Intel 8085
Intel 8080
Intel 8008
Motorola 6800
MOS Technology 6502
Intel 4004
Data source: Wikipedia (wikipedia.org/wiki/Transistor_count)
OurWorldinData.org – Research and data to mak
er CC-BY by the authors Hannah Ritchie and Max Roser.

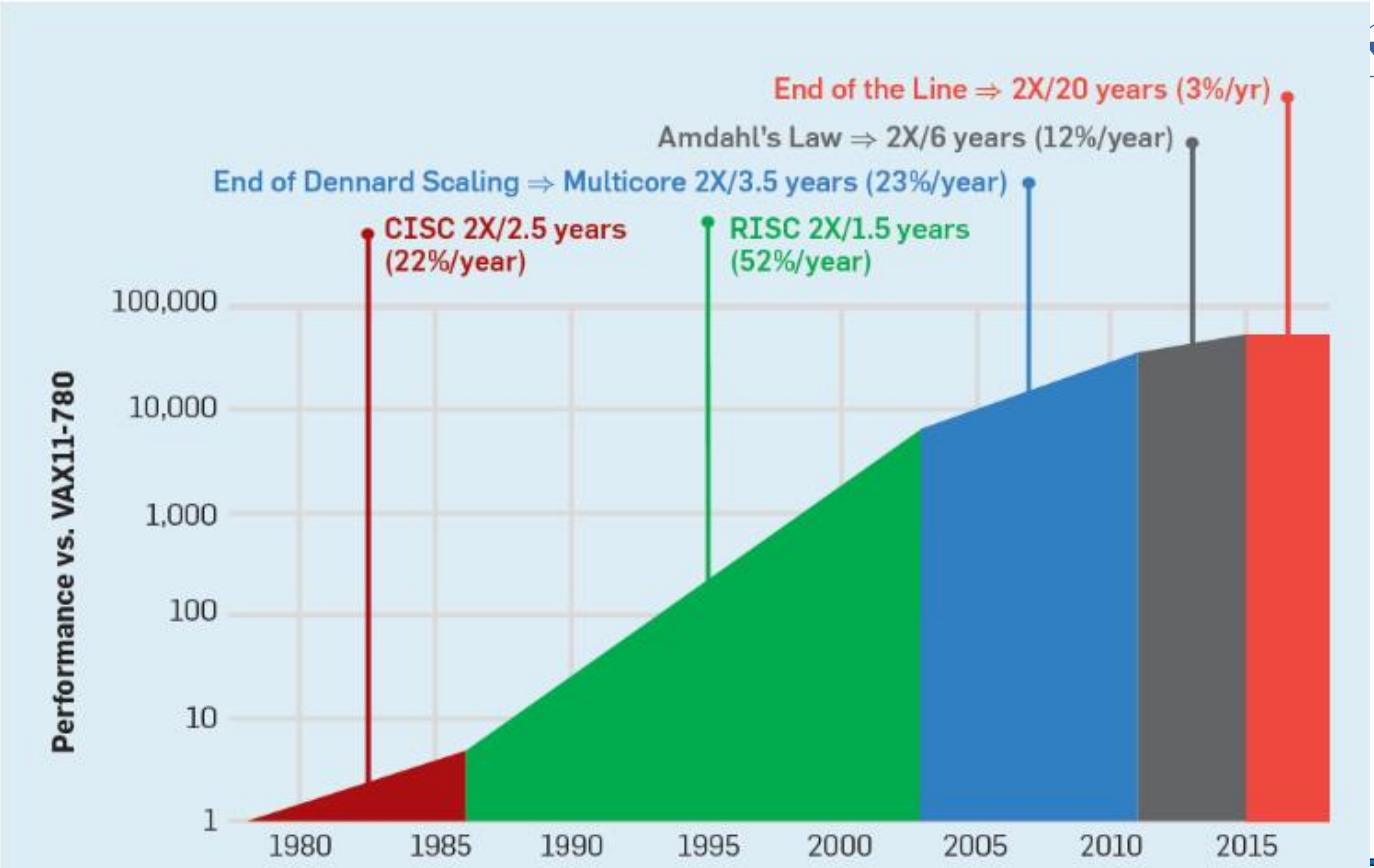


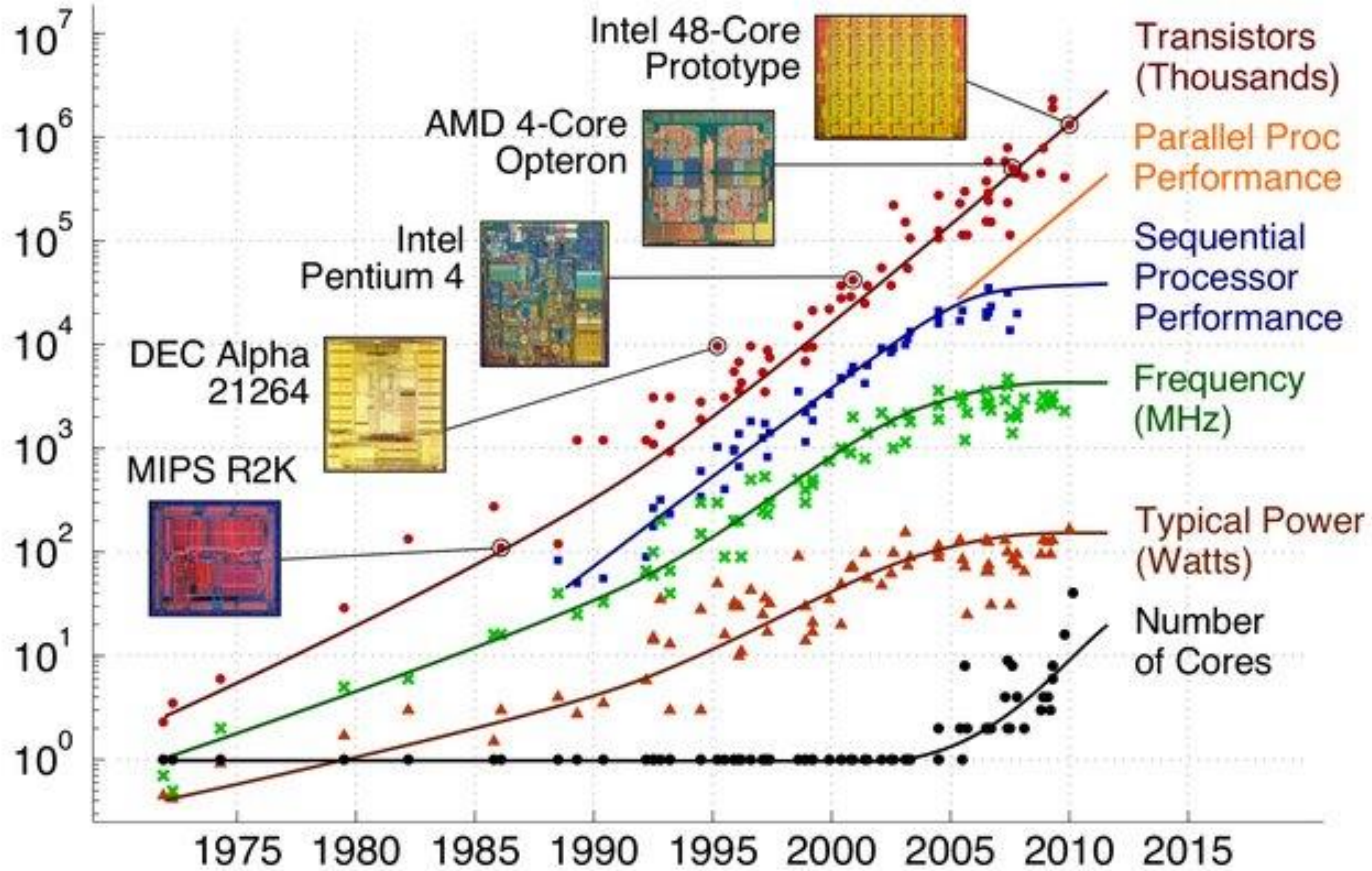
Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2017 by K. Rupp

42 Years of Processor Data



Hennessy and Patterson, Turing Lecture 2018, overlaid over "42 Years of Processors Data"
https://www.karlrupp.net/2018/02/42-years-of-microprocessor-trend-data/;"First Wave" added by Les Wilson, Frank Schirrmeister
Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2017 by K. Rupp


End of the Line ⇒ 2X/20 years (3%/yr)
Amdahl's Law ⇒ 2X/6 years (12%/year)
End of Dennard Scaling ⇒ Multicore 2X/3.5 years (23%/year)
CISC 2X/2.5 years (22%/year)
RISC 2X/1.5 years (52%/year)
Performance vs. VAX11-780
100,000
10,000
1,000
100
10
1
1980
1985
1990
1995
2000
2005
2010
2015



Intel 48-Core Prototype
AMD 4-Core Opteron
Intel Pentium 4
DEC Alpha 21264
MIPS R2K
Transistors (Thousands)
Parallel Proc Performance
Sequential Processor Performance
Frequency (MHz)
Typical Power (Watts)
Number of Cores
10^7
10^6
10^5
10^4
10^3
10^2
10^1
10^0
1975
1980
1985
1990
1995
2000
2005
2010
2015

CPU Architecture Today
Heat becoming un unmanageable problem

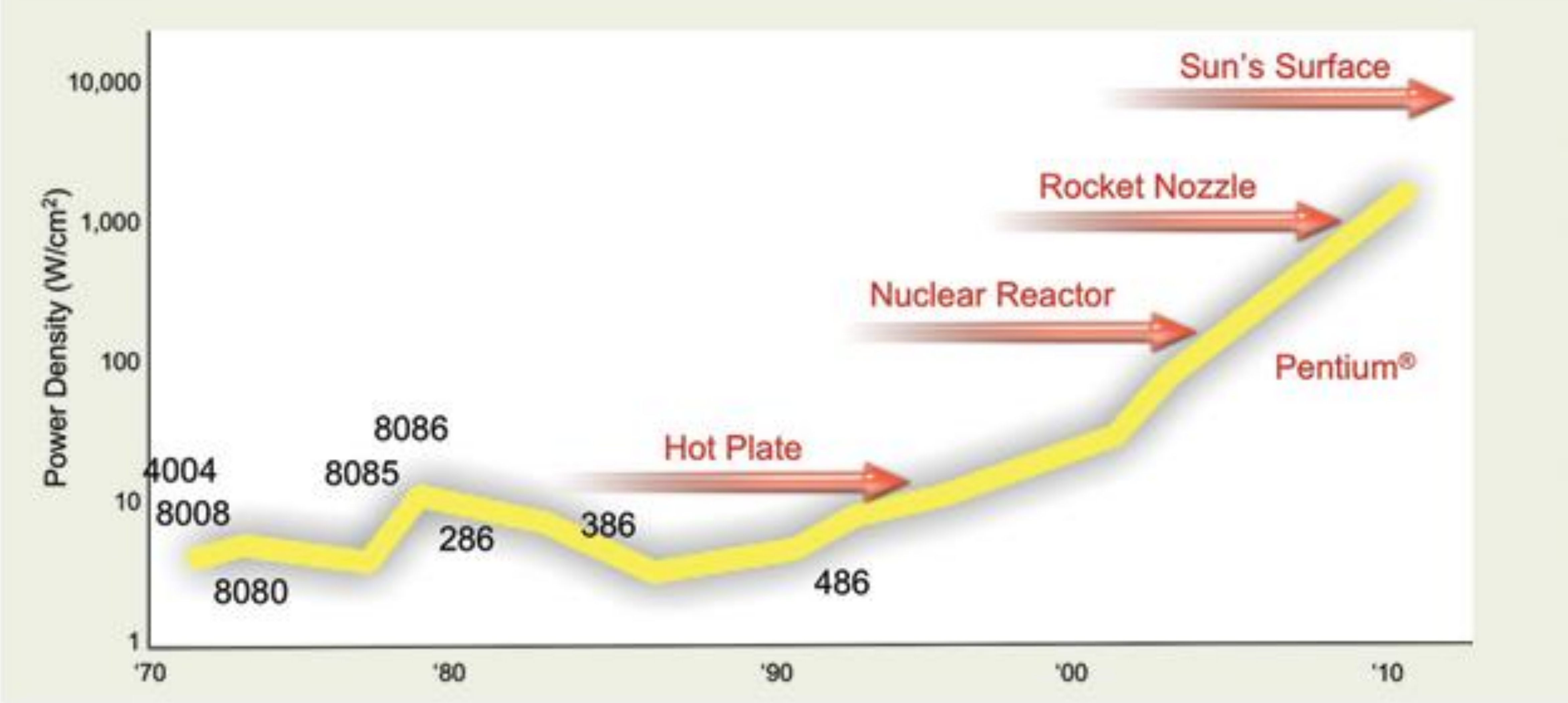


Figure 1. In CPU architecture today, heat is becoming an unmanageable problem. (Courtesy of Pat Gelsinger, Intel Developer Forum, Spring 2004)

CPU: From GHz to multi-core

Moore's Law:

- *~ the number of transistors on an IC doubles every two years.*
  - Less space, more complexity.
  - Shorter gates, higher clock rate.

Strategy of the 80s and 90's:

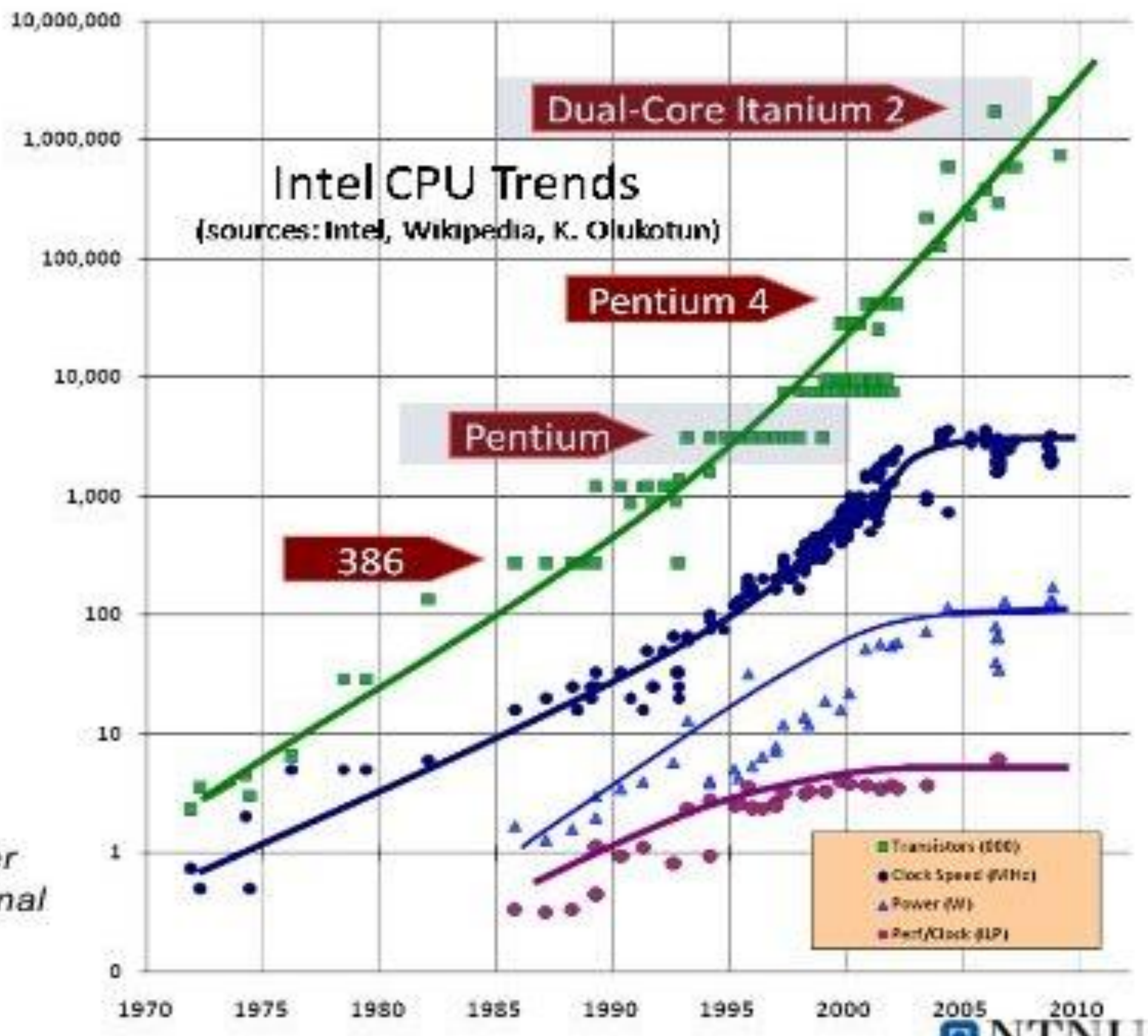
- Add more complexity!
- Increase the clock rate!

Pollack's Rule:

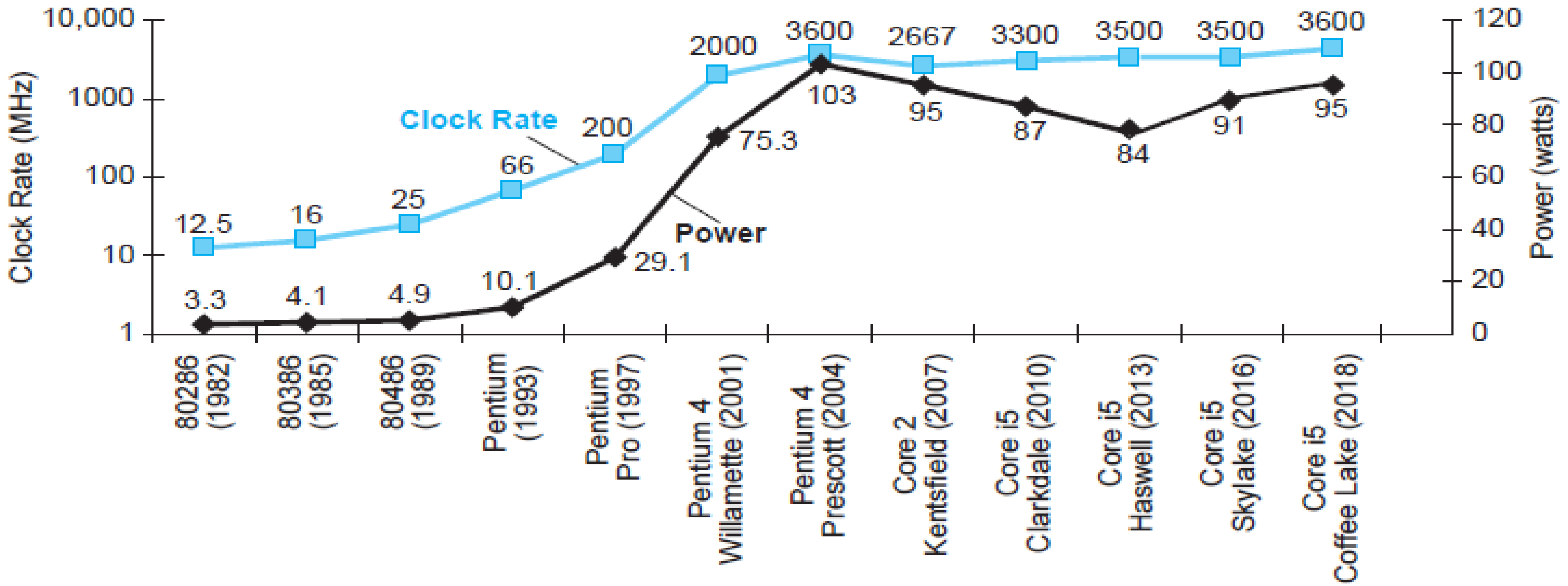
- *The performance increase is ~ square root of the increased complexity.* [Borkar 2007]

The Power Wall:

- *Increasing clock rate and transistor current leakage lead to excess power consumption, while RC delays in signal transmission grow as feature sizes shrink.* [Borkar et al. 2005]



CPU Power Trends



- In CMOS IC technology

$$\text{Power} = \text{Capacitive load} \times \text{Voltage}^2 \times \text{Frequency}$$

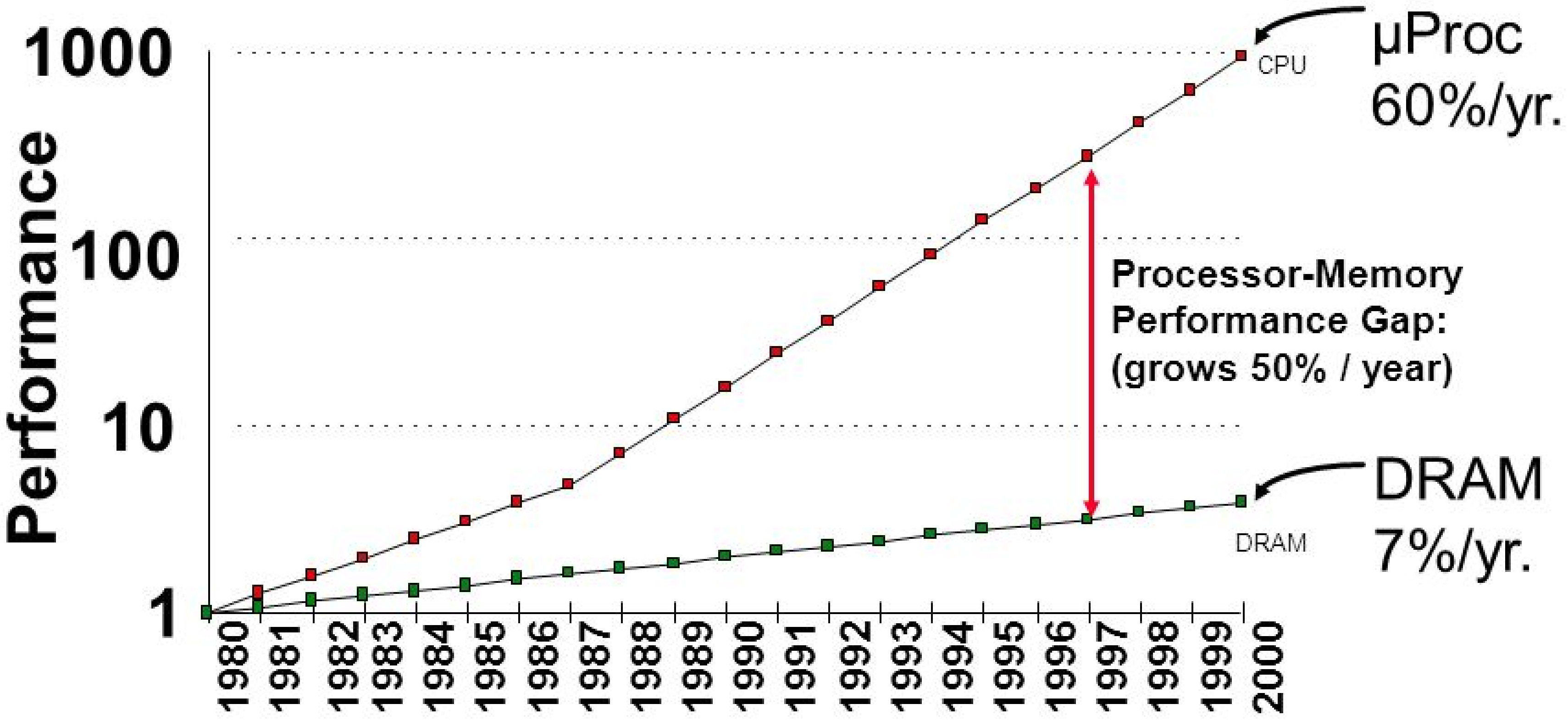
×30

5V → 1V

×1000

Memory Hierarchy: Motivation
Processor-Memory (DRAM) Performance Gap

(i.e. Gap between memory access time and CPU cycle time



EECC551 - Shaaban

(Review from 550)

Reducing Power

- Suppose a new CPU has
  - 85% of capacitive load of old CPU
  - 15% voltage and 15% frequency reduction

$$\frac{P_{new}}{P_{old}} = \frac{C_{old} \times 0.85 \times (V_{old} \times 0.85)^2 \times F_{old} \times 0.85}{C_{old} \times V_{old}^{\;2} \times F_{old}} = 0.85^4 = 0.52$$

- The power wall
  - We can't reduce voltage further
  - We can't remove more heat
- How else can we improve performance?

Computing Ideas in Post-PC Era

Current High-Performance Computing Architectures Post-Moore's-Law Era

Model Based Abstract Software Development for Complex Systems
Parallel Processing through Multicores
Instruction Level Parallelism through Pipelining
Speculation and Prediction in RISC Architecture
Performance Evaluation to make the 'Common Case Fast'
Use variety of memory hierarchies to improve performance
Cloud based Computing and Storage
Role of Network in Computer Performance

Upcoming Computing Architectures
Post-PC Era

- Mobile and Embedded Computing
- Domain Specific Architectures – E.g. Google Tensor Processor
- Domain Specific Languages and Compilers – E.g. Tensorflow
- Open Instruction Set Architectures – RISC – V
- Agile Computing through Reconfigurable Hardware and Customized Instructions
- Security in CPU Hardware

The Scale in Computing

Decimal term	Abbreviation	Value	Binary term	Abbreviation	Value	% Larger
kilobyte	KB	10^3	kibibyte	KiB	2^{10}	2%
megabyte	MB	10^6	mebibyte	MiB	2^{20}	5%
gigabyte	GB	10^9	gibibyte	GiB	2^{30}	7%
terabyte	TB	10^{12}	tebibyte	TiB	2^{40}	10%
petabyte	PB	10^{15}	pebibyte	PiB	2^{50}	13%
exabyte	EB	10^{18}	exbibyte	EiB	2^{60}	15%
zettabyte	ZB	10^{21}	zebibyte	ZiB	2^{70}	18%
yottabyte	YB	10^{24}	yobibyte	YiB	2^{80}	21%

FIGURE 1.1 The 2^X vs. 10^Y bytes ambiguity was resolved by adding a binary notation for all the common size terms. In the last column we note how much larger the binary term is than its corresponding decimal term, which is compounded as we head down the chart. These prefixes work for bits as well as bytes, so *gigabit* (Gb) is 10^9 bits while *gibibits* (Gib) is 2^{30} bits.

The Hardware / Software Interface

Interface of Hardware and Software

Hardware or software component	How this component affects performance
Algorithm	Determines both the number of source-level statements and the number of I/O operations executed
Programming language, compiler, and architecture	Determines the number of computer instructions for each source-level statement
Processor and memory system	Determines how fast instructions can be executed
I/O system (hardware and operating system)	Determines how fast I/O operations may be executed

Hardware / Software Interface

Software

Instruction Set Architecture

Hardware

C/C++/Python
↓
Assembly / Machine Code
↓
Registers, Datapath
↓
Gates and Logic
↓
Technology

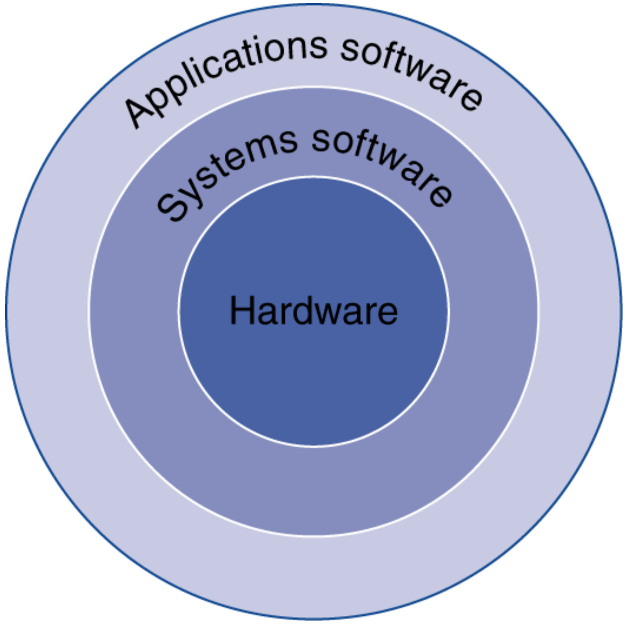
Programmer View

Why Abstraction?

- Delving into the depths reveals more information
- An abstraction omits unneeded detail, helps us cope with complexity

Below a Computer Program

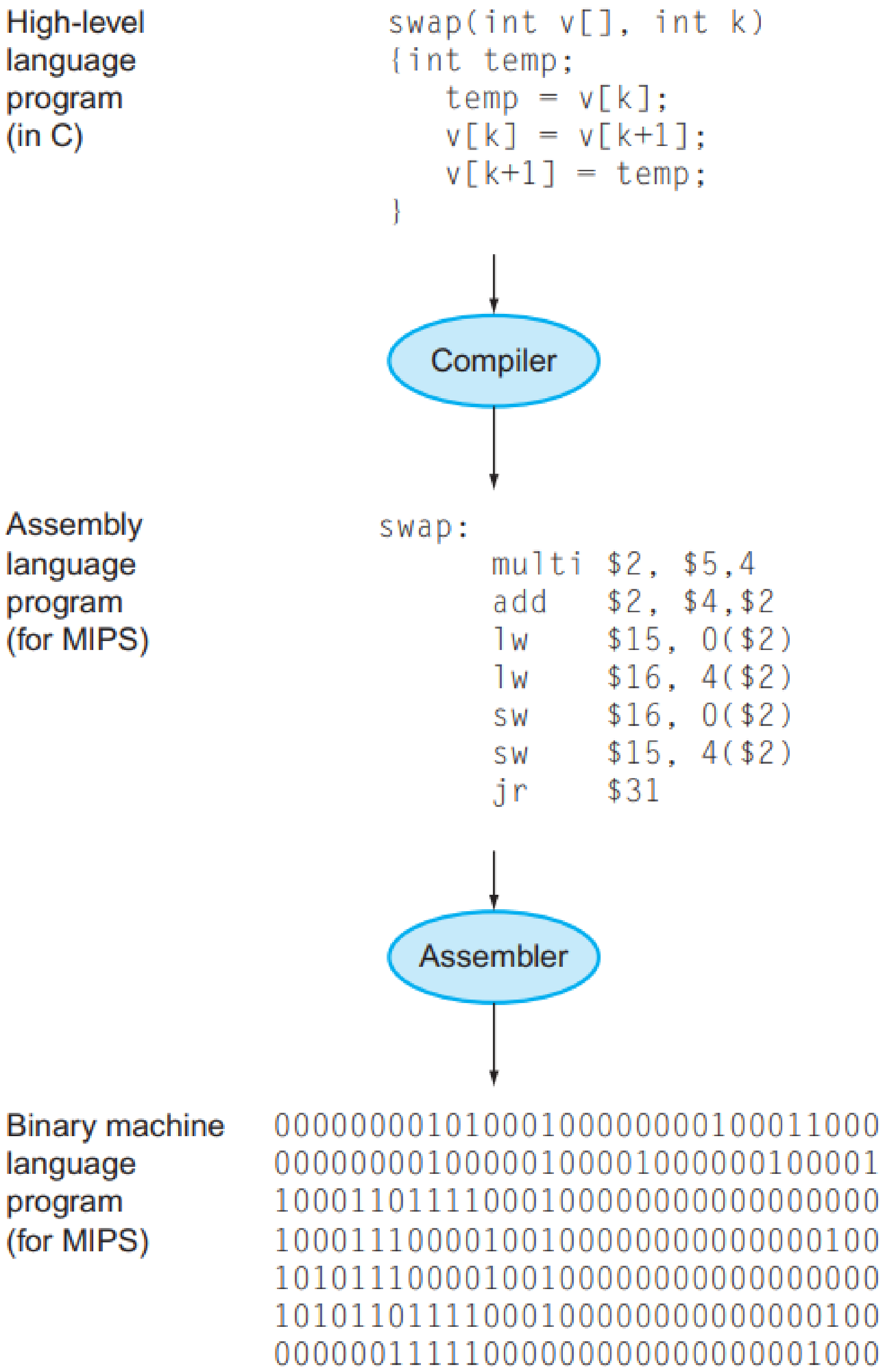
Below Your Program



- Application software
  - Written in high-level language
- System software
  - Compiler: translates HLL code to machine code
  - Operating System: service code
    - Handling input/output
    - Managing memory and storage
    - Scheduling tasks & sharing resources
- Hardware
  - Processor, memory, I/O controllers

Compiling a Program

Example of Software Abstraction

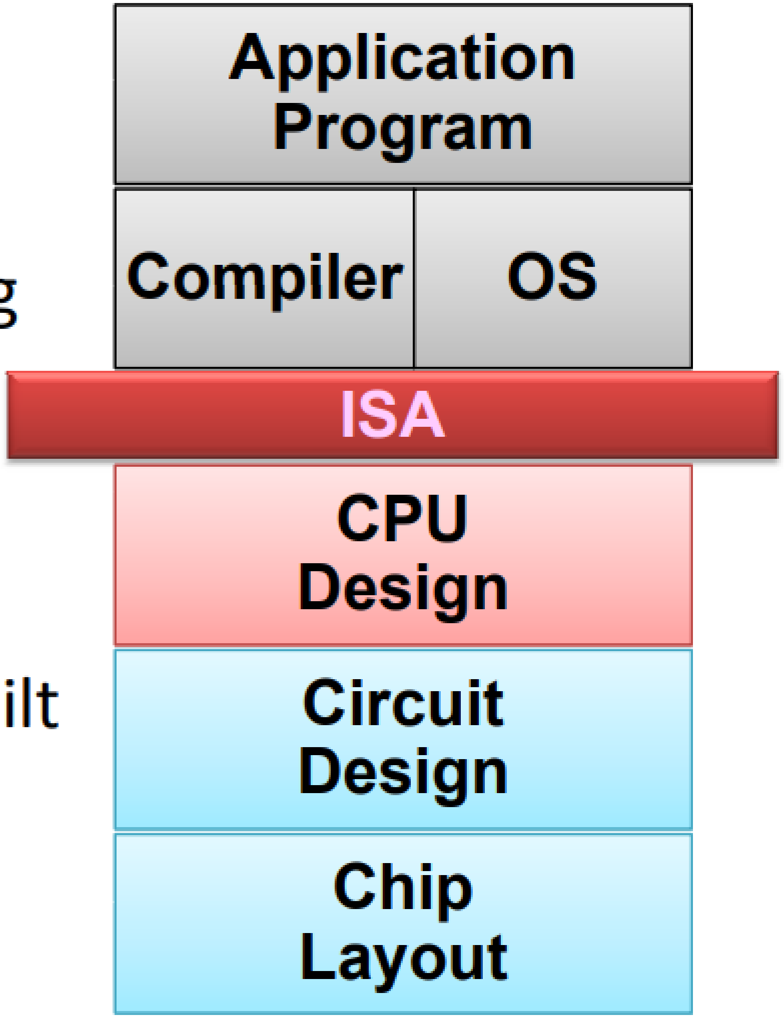


What is an Instruction Set Architecture ISA?

- Lowest level of Computer Architecture that is visible to a programmer
- ISA comprises instructions in Assembly Language and Corresponding Binary Machine Code
- It is the language of the CPU machine
- Primitive syntax compared to a high-level language
- It is easily interpreted and understood by the CPU
- Designed to maximize performance
- Designed to minimize cost and design time

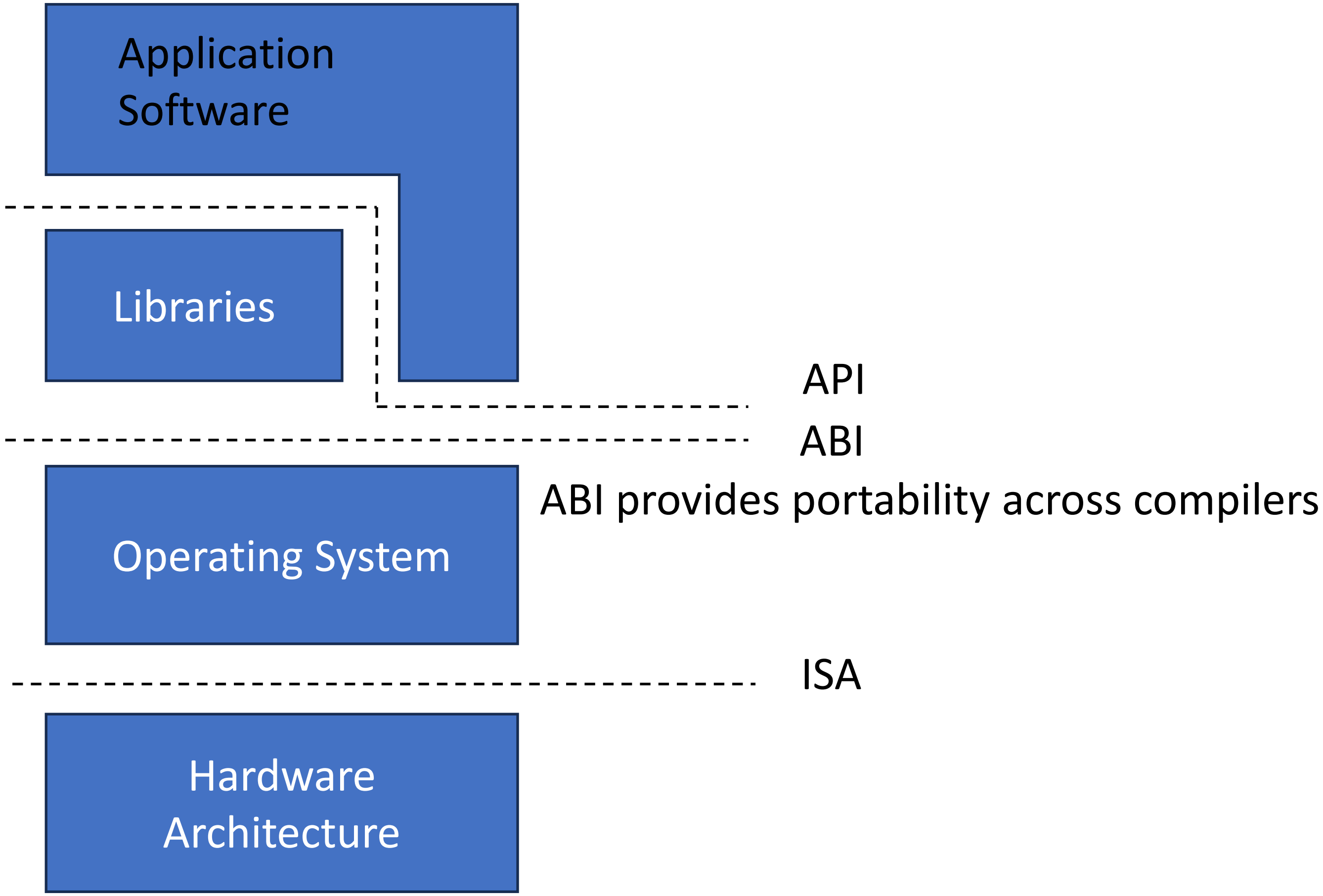
Role of ISA within a Computer

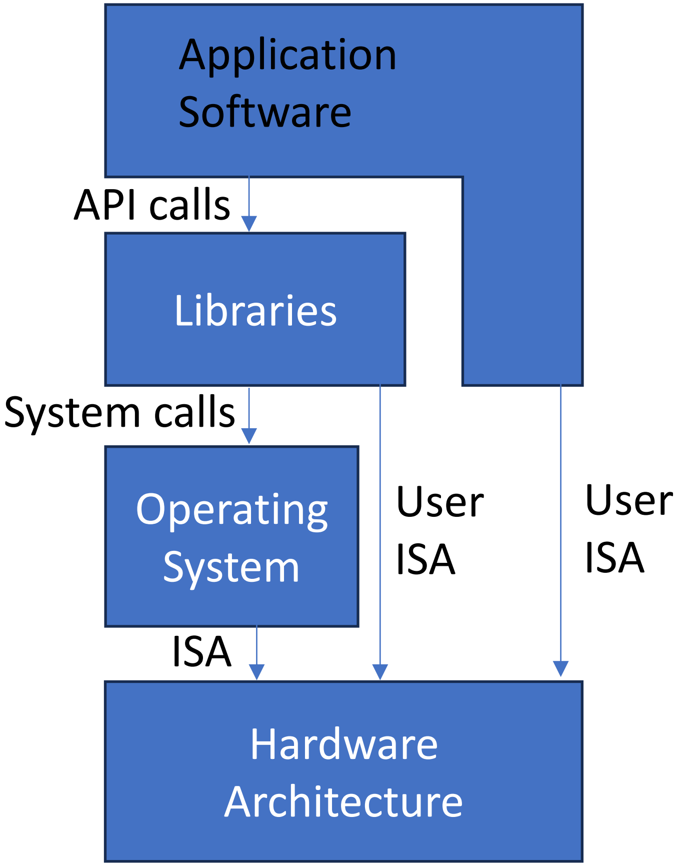
- Assembly Language View
  - Processor state (RF, mem)
  - Instruction set and encoding
- Layer of Abstraction
  - Above: how to program machine - HLL, OS
  - Below: what needs to be built - tricks to make it run fast



Key:
RF = Register File
Mem = Memory
HLL = High Level Language
OS = Operating System

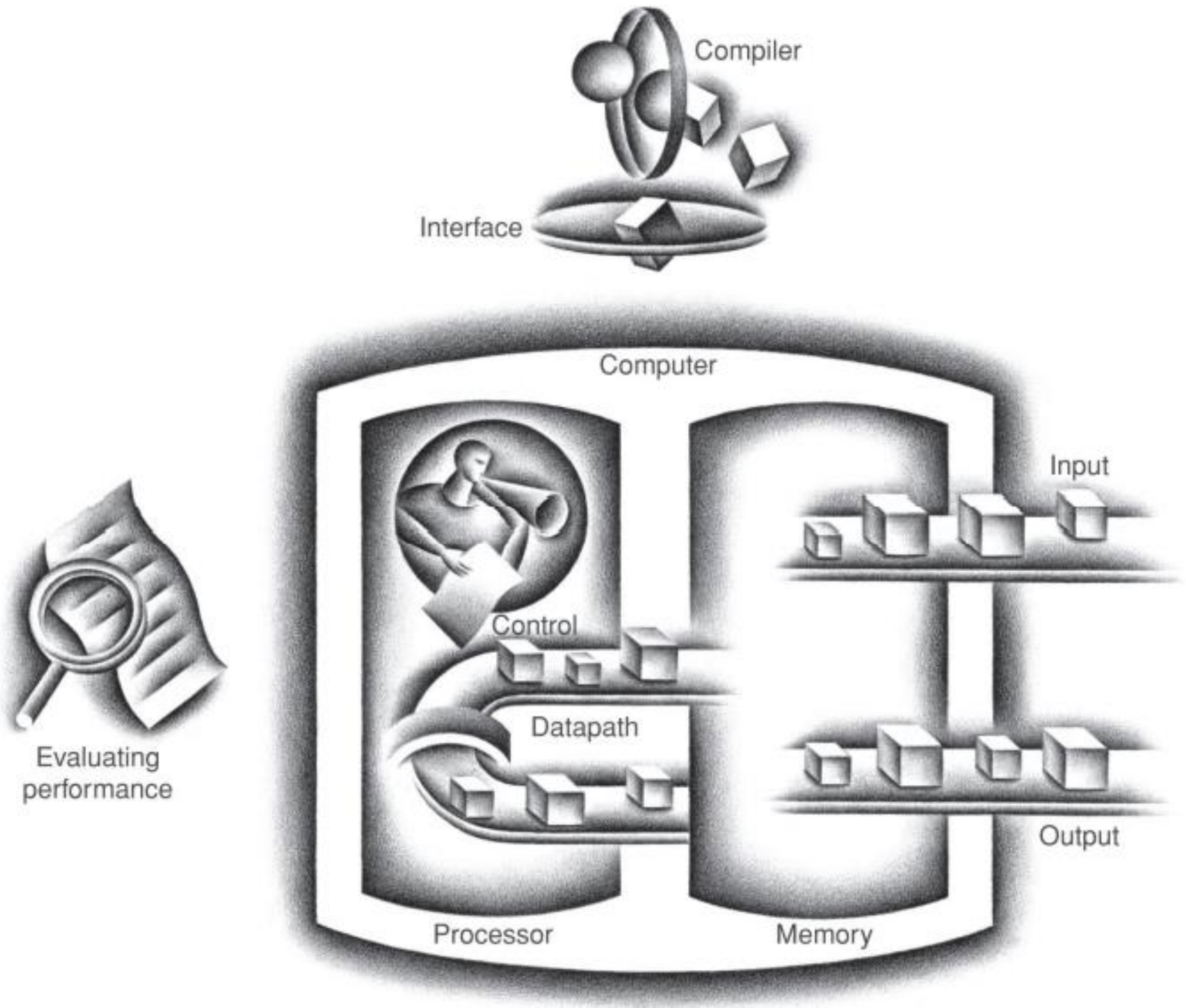
Operating System Interface Software / Hardware





Exploring Computer Organization

Typical Computer Organization



Computer
-
Main Components

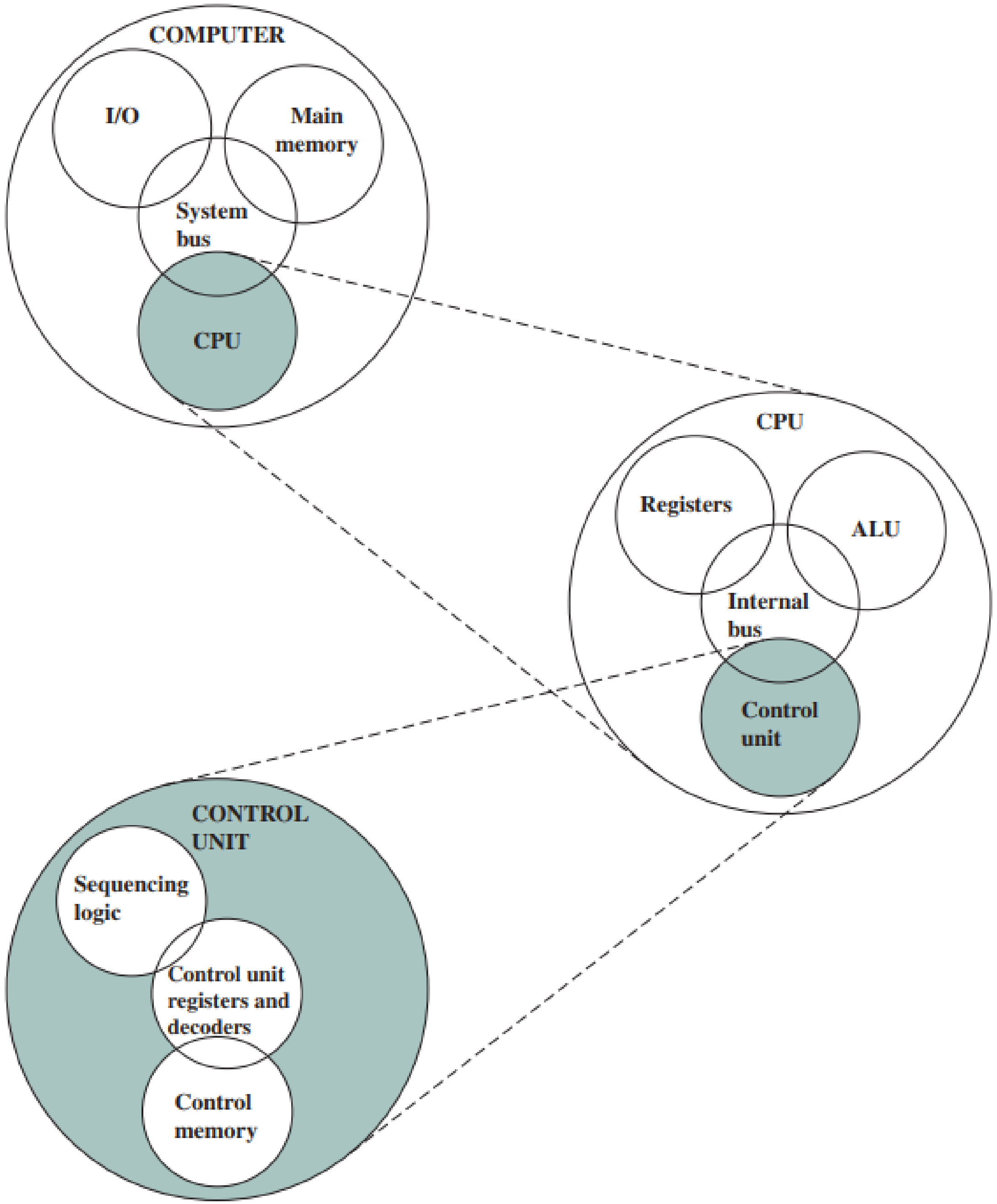


Figure 1.1 The Computer: Top-Level Structure

A Multi-core Computer

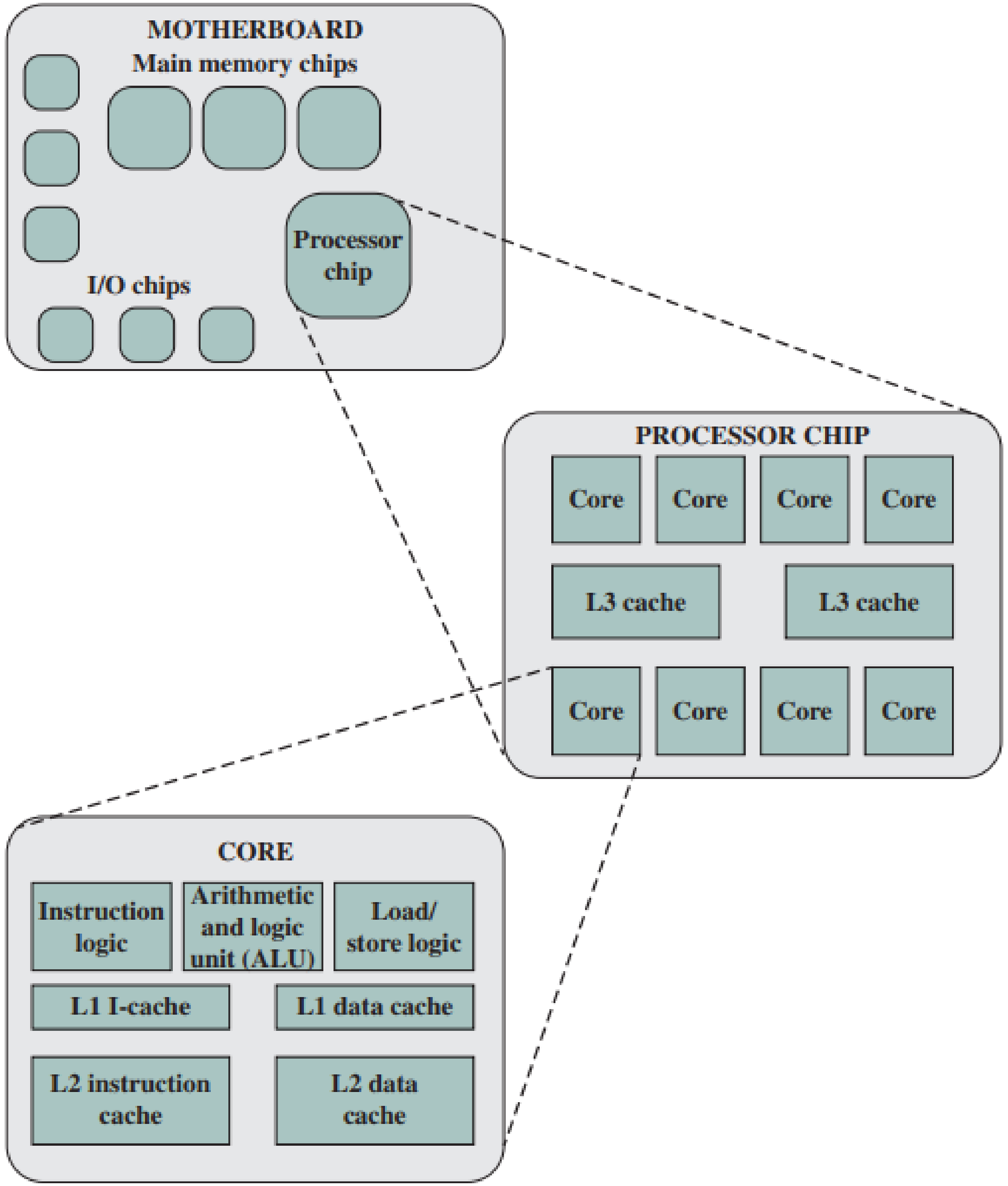
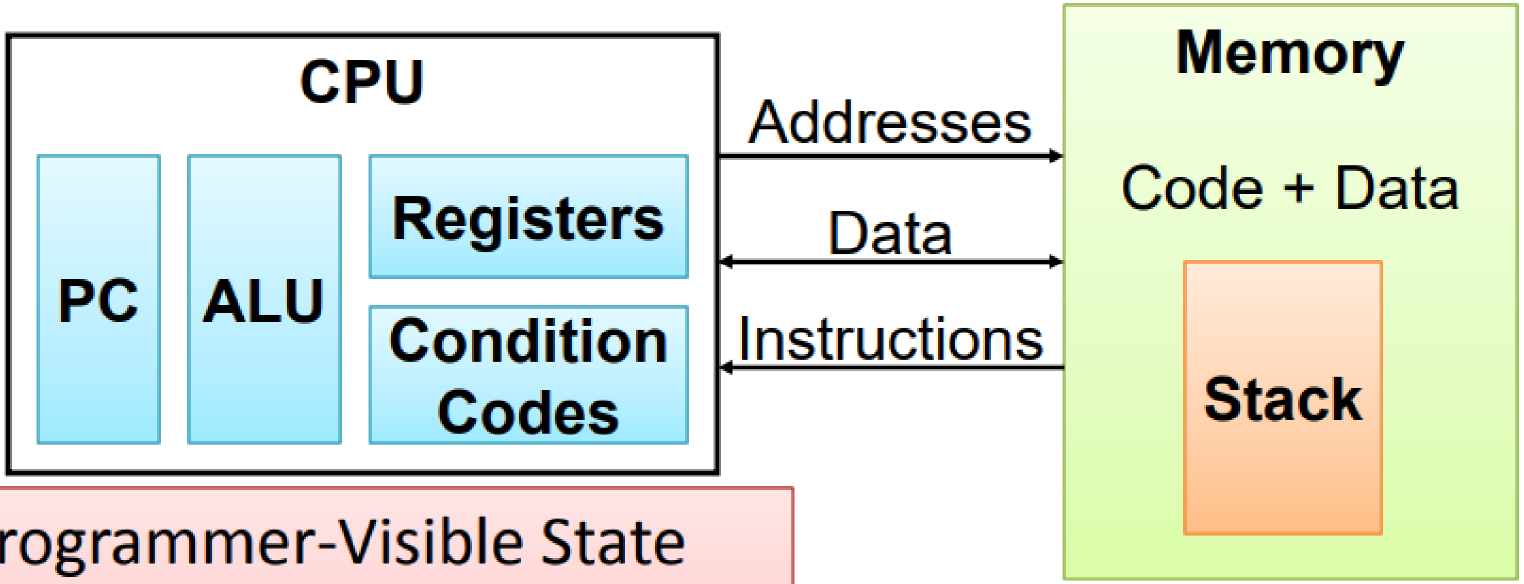


Figure 1.2 Simplified View of Major Elements of a Multicore Computer

Computer Abstraction

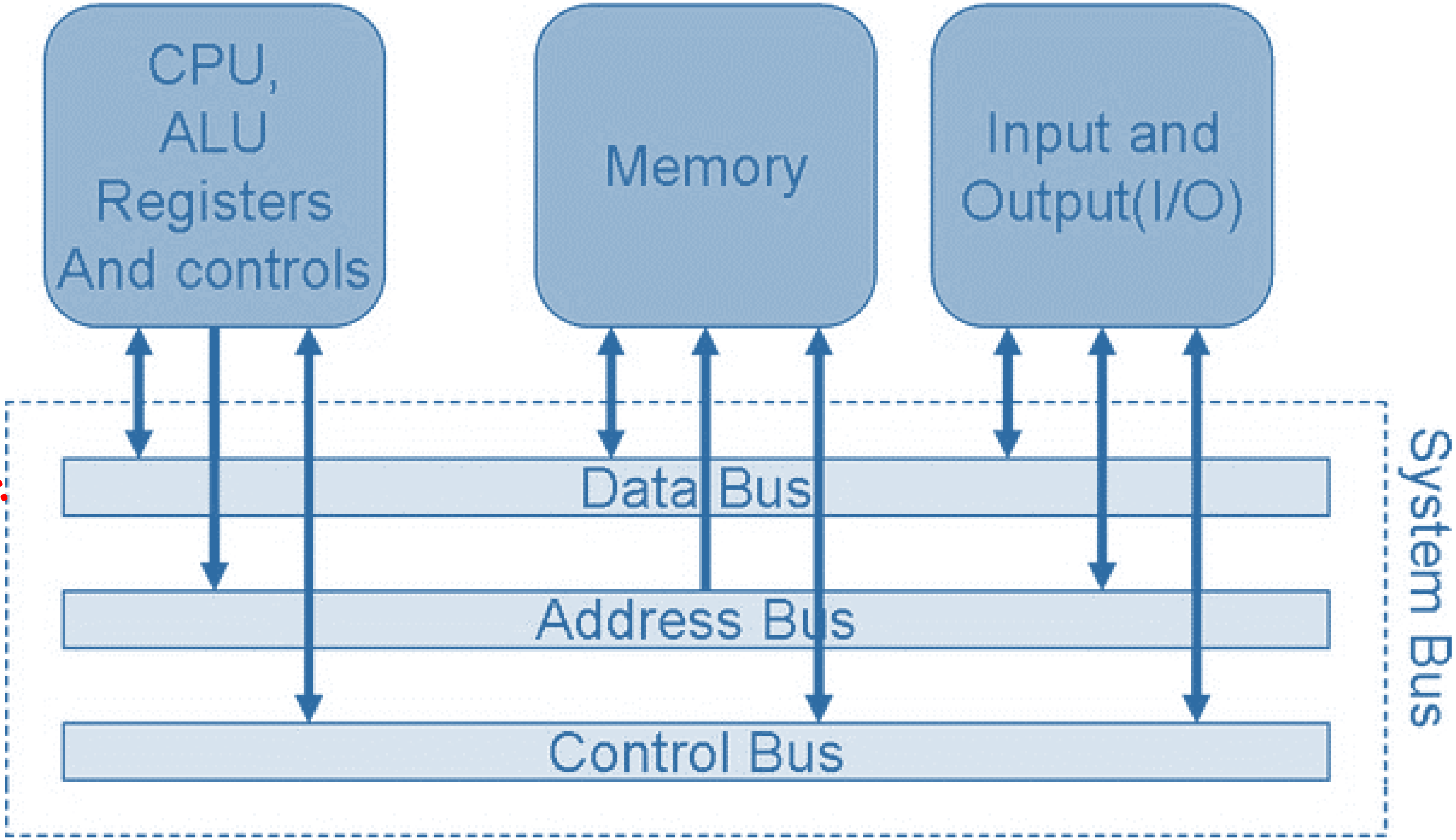


- Programmer-Visible State
  - PC Program Counter
  - Register File
    - Heavily used data
  - Condition Codes

- Memory
  - Byte array
  - Code + data
  - stack

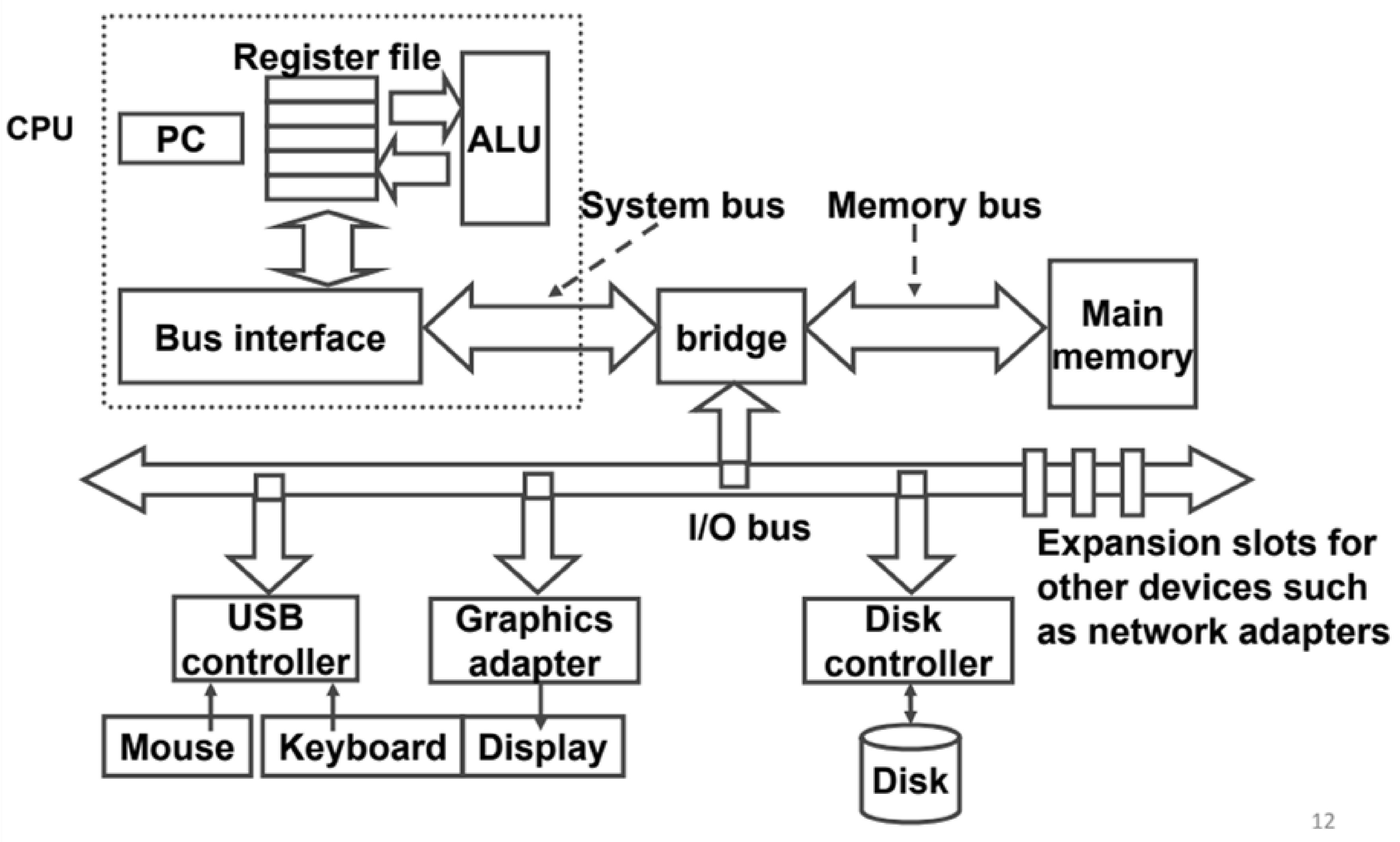
23

The Von-Neumann Architecture



Three Buses in Von-Neumann:
1. Data Bus
2. Address Bus
3. Control Bus

Hardware Abstraction Example



Computer Performance

Simple Computer Performance

- Time to execute a program – in seconds
- Clock Speed of Computer – MHz or GHz
- In terms of the width of Data Bus (16 bit, 32 bit, 64 bit)
- In terms of the size of different memory (8GB / 256GB)
  - Cache size
  - RAM
  - Hard disk
  - SSD, etc.
- In terms of multiple cores and multiple I/O available
- Software
  - Algorithm
  - Language / Compiler
  - Optimization – Parallel, Vector, etc.

What is MIPS?

- **MIPS is a leading example of RISC Architecture**
- **Used in many household products such as Nintendo, Sony, Mobile phones, Routers, etc.**
- **Simple and Small Instruction Set**

MIPS = Microprocessor without Interlock Pipeline Stages

Vs

MIPS = Million Instructions Per Second

What is MFLOPS

FLOPS = Floating Point Linear Operations Per Second

$$\text{MFLOPS}\ rate = \frac{Number\ of\ executed\ floating\ point\ operations\ in\ a\ program}{Execution\ time\ \times 10^6}$$

Modern notion of computer performance

user

<u>Software Performance</u>
Programming Languages
Operating System
Compilers
Algorithms
Parallelism / Threads

Software optimization

<u>Hardware Performance</u>
Processor Architecture
Vector Processing
Memory Hierarchy
Multicores
Network support

Hardware optimization

Make Common Case FAST