中山大學
SUN YAT-SEN UNIVERSITY

量化设计与分析基础

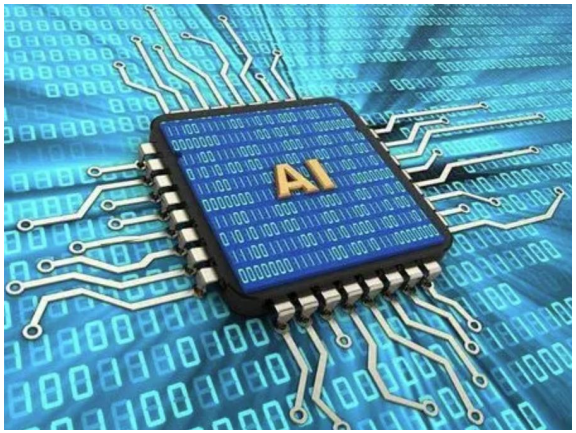
主讲教师：胡淼

中山大学 计算机学院

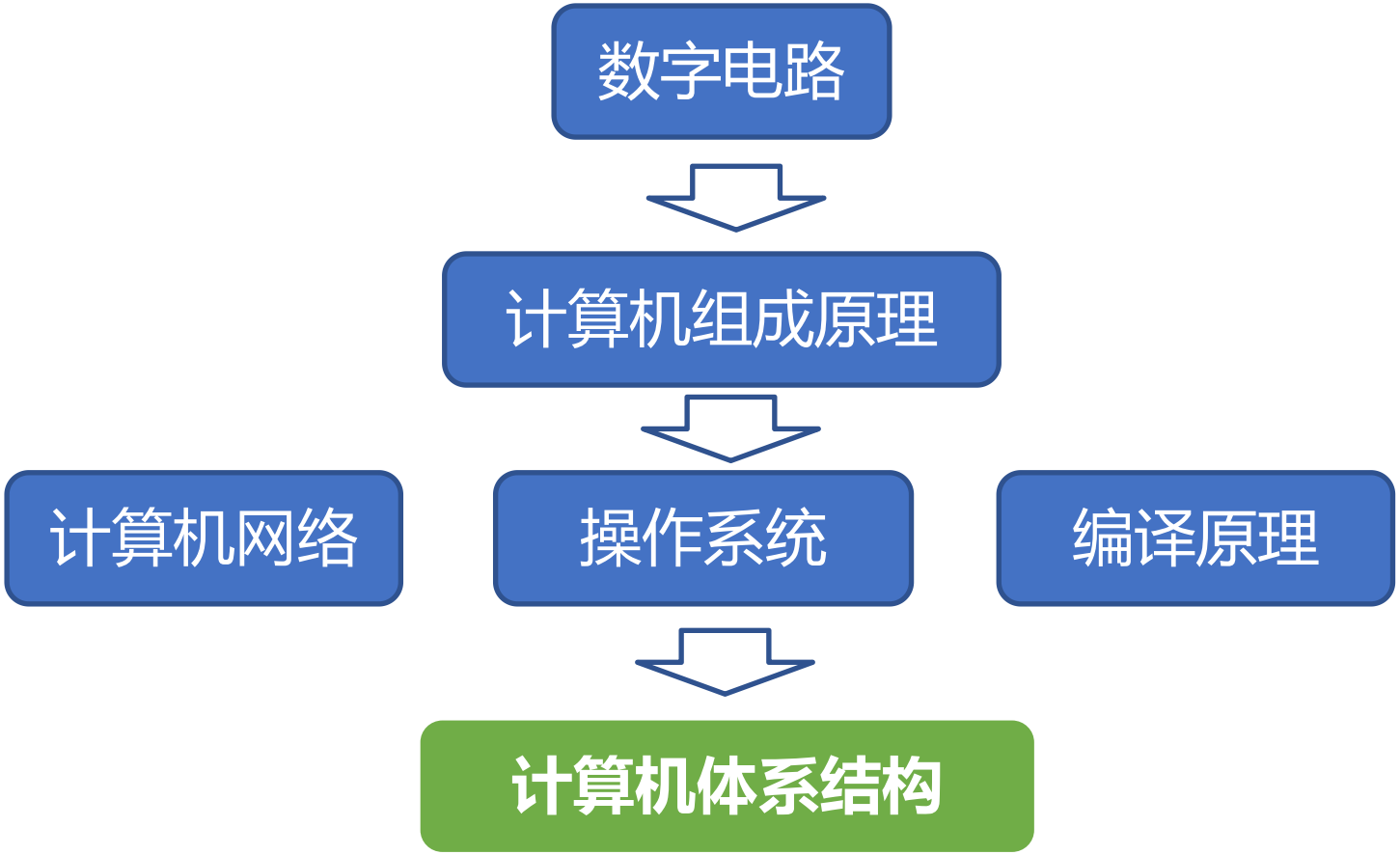
2024 年 秋季

哪些人应该选这门课?

- **高年级本科生或研究生**
  - 未来打算从事计算机体系结构相关方向研究
  - 计算机系统的设计者
    - AI智能计算系统
    - 云计算、边缘计算系统
    - 并行计算系统等
  - 对计算机系统有兴趣



课程先序关系

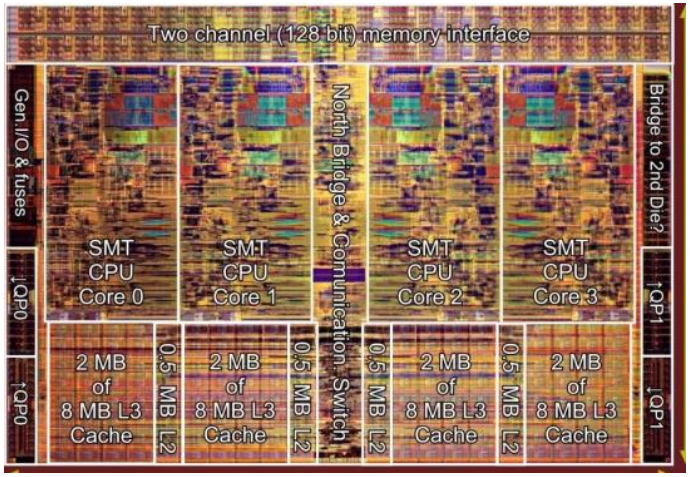


课程简介

- **教学目标：**
  - 了解计算机体系结构的基础和当代主流计算机系统
  - 熟悉**存储顺序性**、**Cache 一致性**、**流水线**和**片上网络**等内容，初步掌握高性能处理器的设计与分析方法
  - 熟悉计算机的体系架构，了解处理器**性能评测**基本手段和方法
  - 了解**异构计算体系结构**和**领域专用体系结构**

课程内容

- 高性能计算机的基本概念
- 量化设计与分析基础
- 并行程序的性能评价和基准评测集
- 高性能计算机的体系结构
- 高性能处理器的并行计算技术
- 高性能计算机的存储层次
- 高性能计算机的互连网络
- 异构计算体系结构
- 领域专用体系结构



教学内容

- 第 1 章 量化设计与分析基础
- 第 2 章 基准评测集
- 第 3 章 高性能计算机的体系结构
- 第 4 章 高性能处理器的并行计算技术
- 第 5 章 高性能计算机的存储层次
- 第 6 章 高性能计算机的互连网络
- 第 7 章 异构计算体系结构
- 第 8 章 领域专用体系结构

课程参考教材

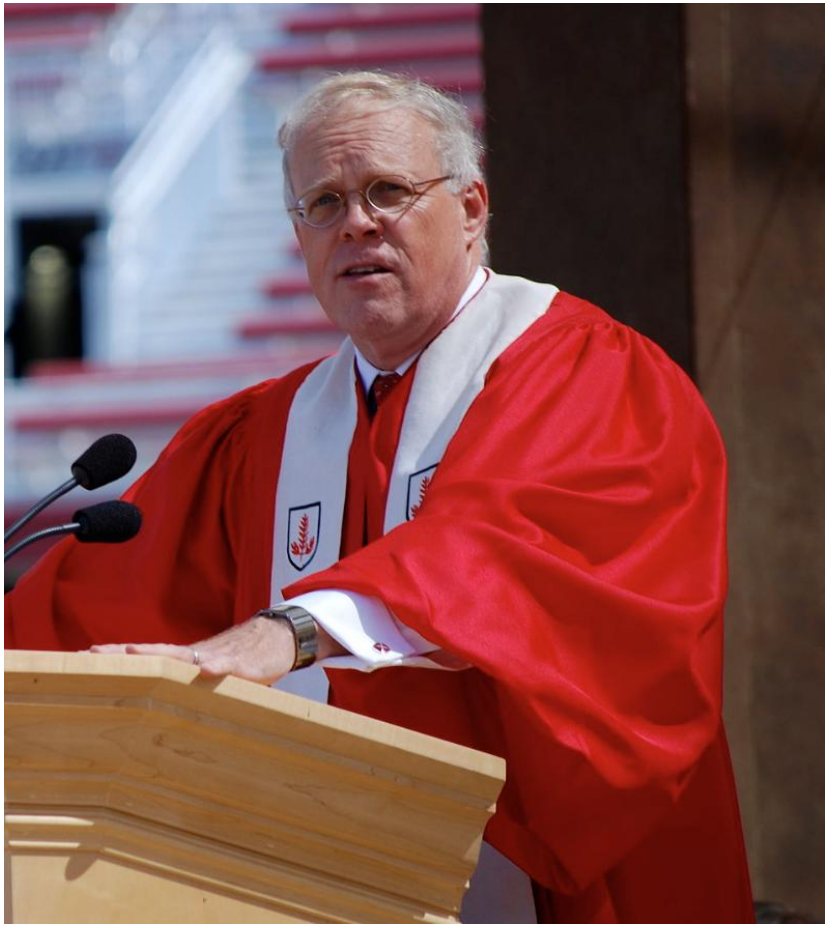
- **参考书目：**

1. John L. Hennessy, and David A. Patterson. **Computer Architecture: A Quantitative Approach**. Morgan Kaufmann; 6th edition, 2017
2. 陈国良，吴俊敏 等. **并行计算机体系结构** (第2版). 高等教育出版社, 2021
3. 陈云霁 等. **智能计算系统 从深度学习到大模型** (第2版). 机械工业出版社, 2024
4. 胡伟武 等. **计算机体系结构基础** (第3版). 机械工业出版社, 2021
5. Yan Solihin, **Fundamentals of Parallel Computer Architecture: Multichip and Multicore Systems**, Chapman & Hall/CRC Computational Science, 2015

- **计算机体系结构：量化研究方法（英文版·原书第6版）[Computer Architecture：A Quantitative Approach，Sixth Edition]**

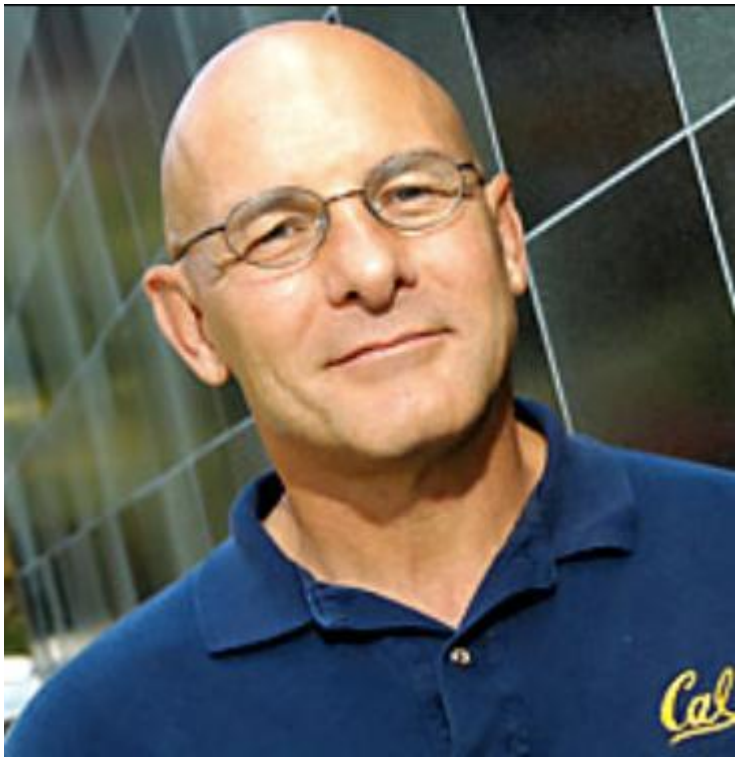
- **作者：**
  - **John L．Hennessy**
  - **David A．Patterson**

John L. Hennessy



- 2017年图灵奖获得者
- Google母公司Alphabet的董事长
- 曾任斯坦福大学第十任校长
- 美国三院院士
- MIPS公司创始人

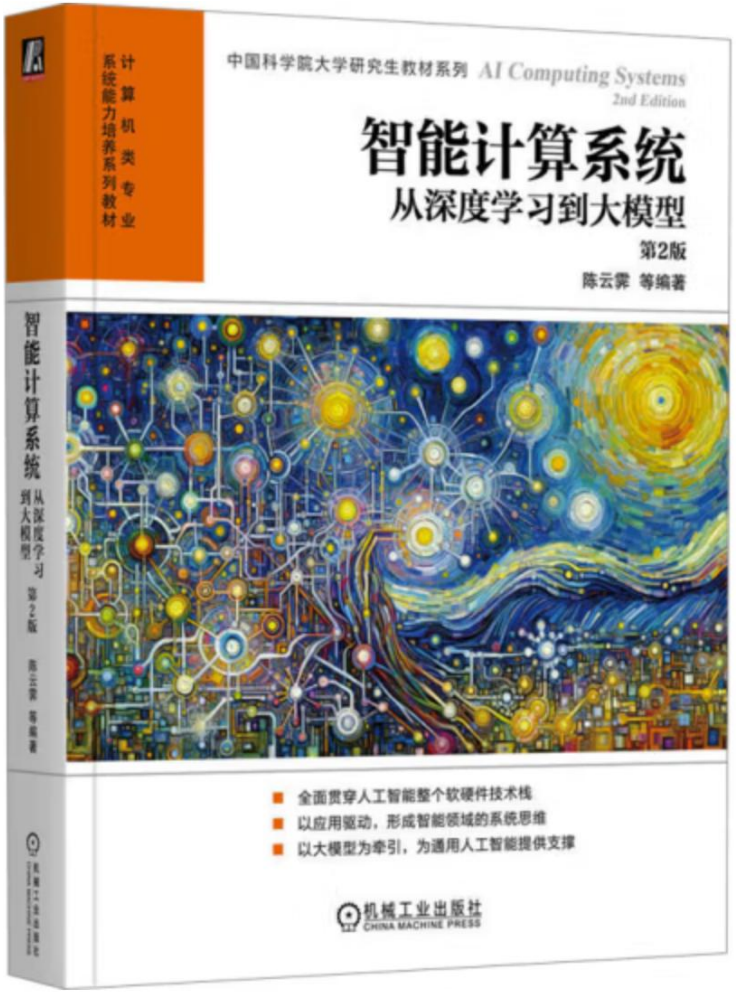
David A. Patterson

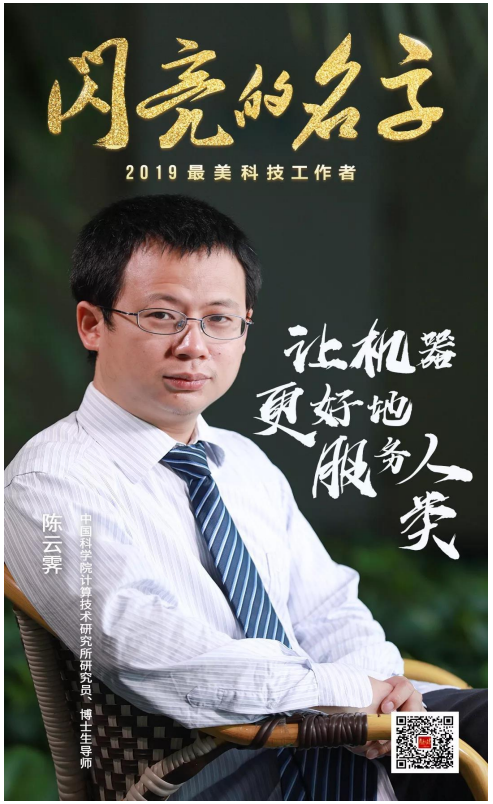


- 2017年度图灵奖获得者
- 加州大学伯克利分校教授
- 曾任ACM主席
- ACM和IEEE会士
- 美国三院院士
- 领导RISC、RAID等项目

陈国良 院士

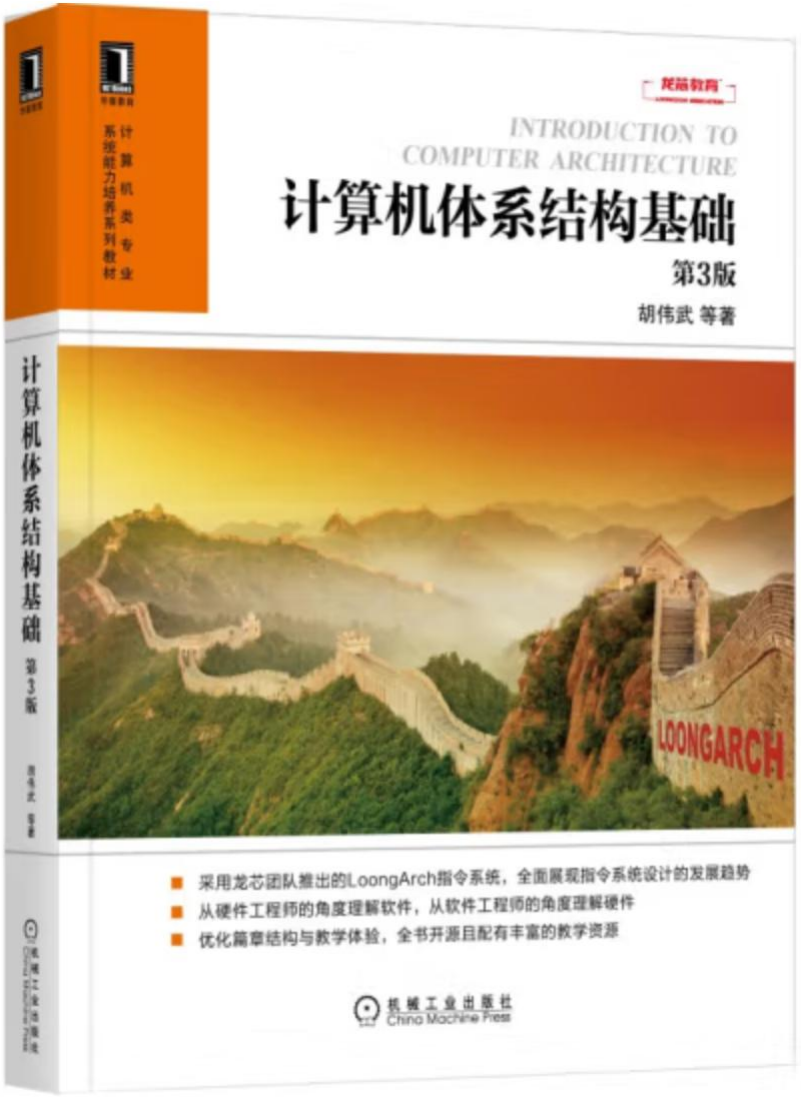
- 计算机专家，主要从事并行算法研究
- 1938年6月3日生于安徽颍上
- 1961年毕业于西安交通大学计算机专业
- 2003年当选为中国科学院院士





1983出生于江西南昌的 陈云霁

- 9岁上中学
- 14 岁进入中科大少年班
- 24 岁取得计算机博士学位
- 29岁晋升为研究员（Full Professor）
- 2017年国家自然科学奖二等奖；2019年科学探索奖；2020年中国青年五四奖章；2022年全国五一劳动奖章；2025年，中国科学院院士增选有效候选人

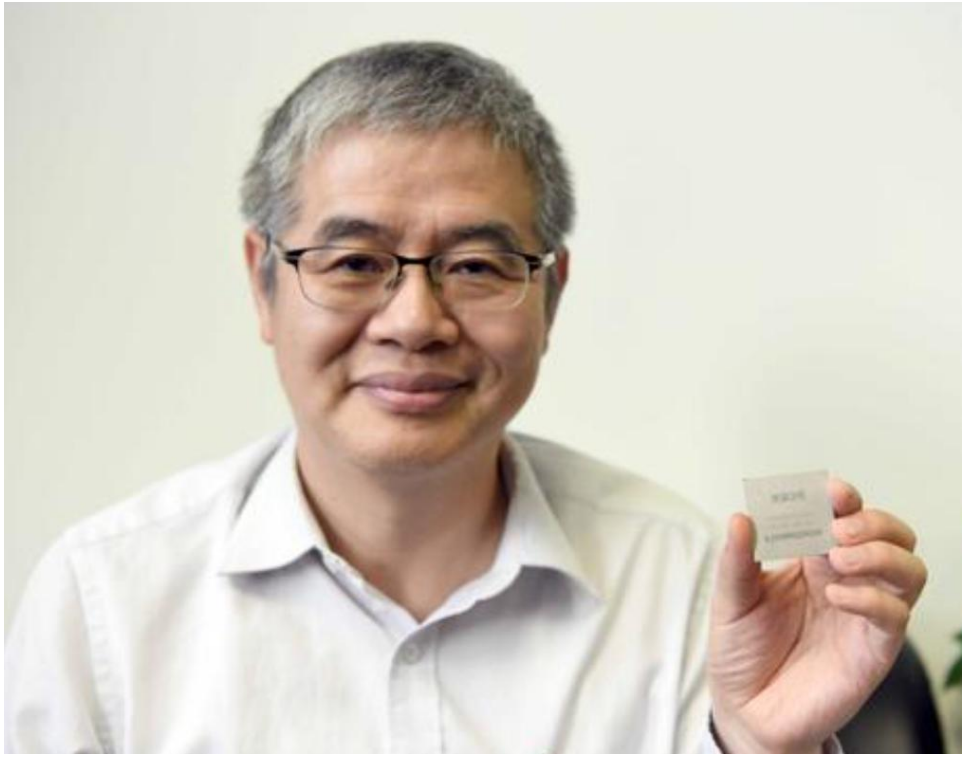


https://foxsen.github.io/archbase/

胡伟武（1968年11月 - ），男，汉族，浙江永康人，中国计算机科学家

现任龙芯中科技术有限公司董事长，中科院计算所总工程师，龙芯CPU首席科学家

20年龙芯CPU之路




LOONGSON
Cored By
龙芯
LOONGSON


龙芯中科
热烈庆祝龙芯中科技术股份有限公司
成功登陆上交所科创板
2022年6月24日 中国·北京
龙芯中科
LOONGSON TECHNOLOGY

课程考核

- **考核形式：**
  - 课堂出勤：**10%**
  - 平时作业：**40%**
  - 期末大作业：**50%**
- **作业提交形式：**
  - 通过超算习堂平台（easyhpc.net）在线提交

让我们开始吧！

教学内容

- **第 1 章 绪论**
- 第 2 章 基准评测集
- 第 3 章 高性能计算机的体系结构
- 第 4 章 高性能处理器的并行计算技术
- 第 5 章 高性能计算机的存储层次
- 第 6 章 高性能计算机的互连网络
- 第 7 章 异构计算体系结构
- 第 8 章 领域专用体系结构

第 1 章 绪论

课程内容

- **Computer Architecture, A Quantitative Approach (6th Ed.), 2017**
  - Chapter 1. Fundamentals of Quantitative Design and Analysis
- **并行计算机体系结构 (第2版), 2021**
  - 第一章 绪论

提 纲

- （一）量化设计与分析基础
- （二）并行计算与并行计算机
- （三）高性能计算机发展史

（一）量化设计与分析基础

1.1 Introduction

Computer Technology

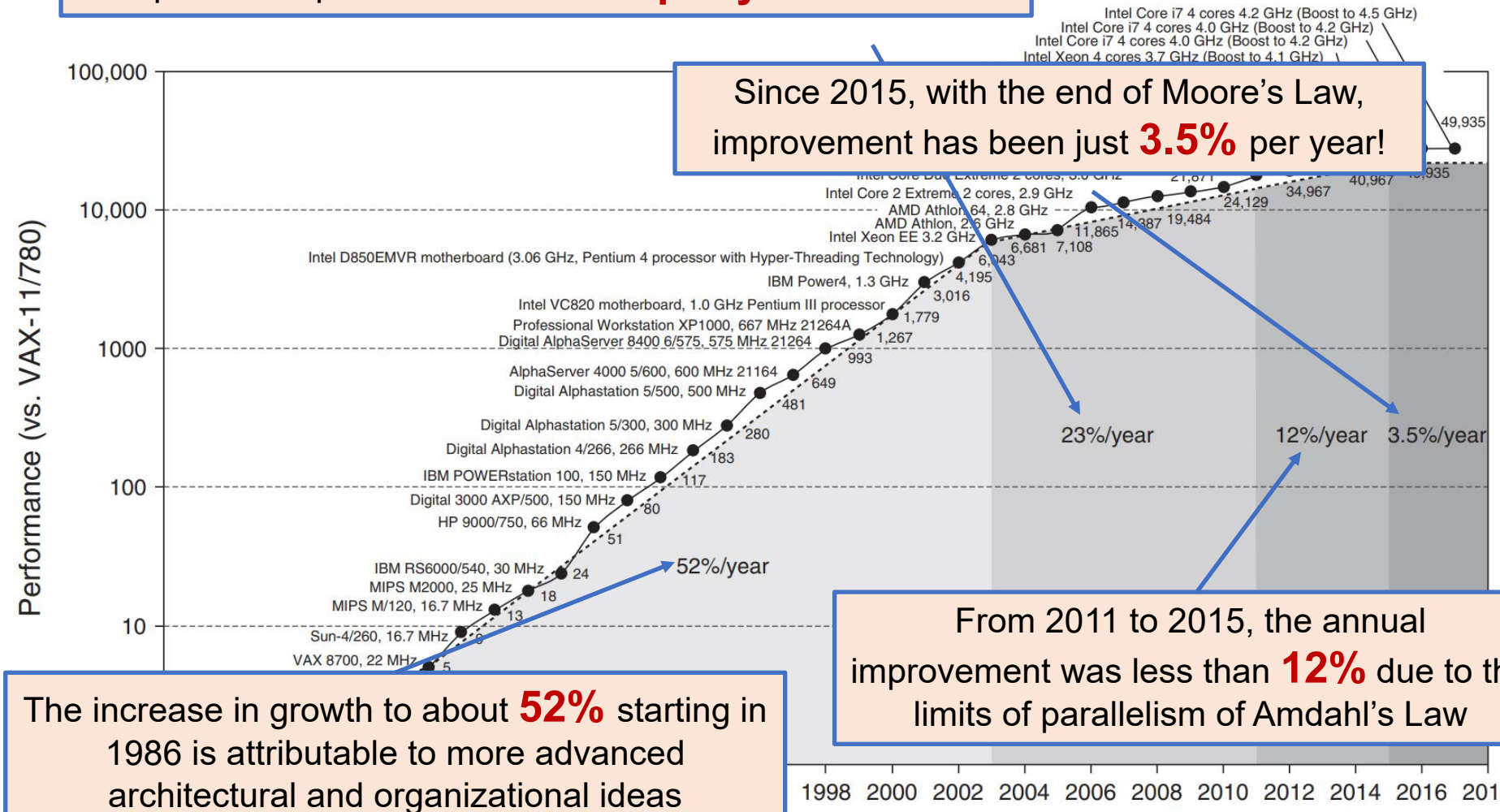
- **The rapid improvement in computer technology has come from advances in two aspects:**
  - **Semiconductor technology** used to build computers
    - integrated circuit, microprocessor
  - Innovations in **computer architecture**:
    - virtual elimination of **assembly language programming**
      - reduced the need for object-code compatibility
    - the creation of **standardized, vendor-independent OS**
      - such as UNIX, Linux
      - lowered the cost and risk of bringing out a new architecture

Computer Technology

- **These changes made it possible to develop a new set of architectures with simpler instructions**
  - called **RISC (Reduced Instruction Set Computer)**
- **RISC focused on two critical performance techniques**
  - the exploitation of **instruction-level parallelism**
    - pipelining -> multiple instruction issue
  - the use of **caches**
    - simple -> sophisticated organizations
- **RISC-based computers raised the performance bar**
  - ARM, becoming dominant

Single Processor Performance

In 2003 the limits of power and the available ILP slowed uniprocessor performance to **23% per year** until 2011

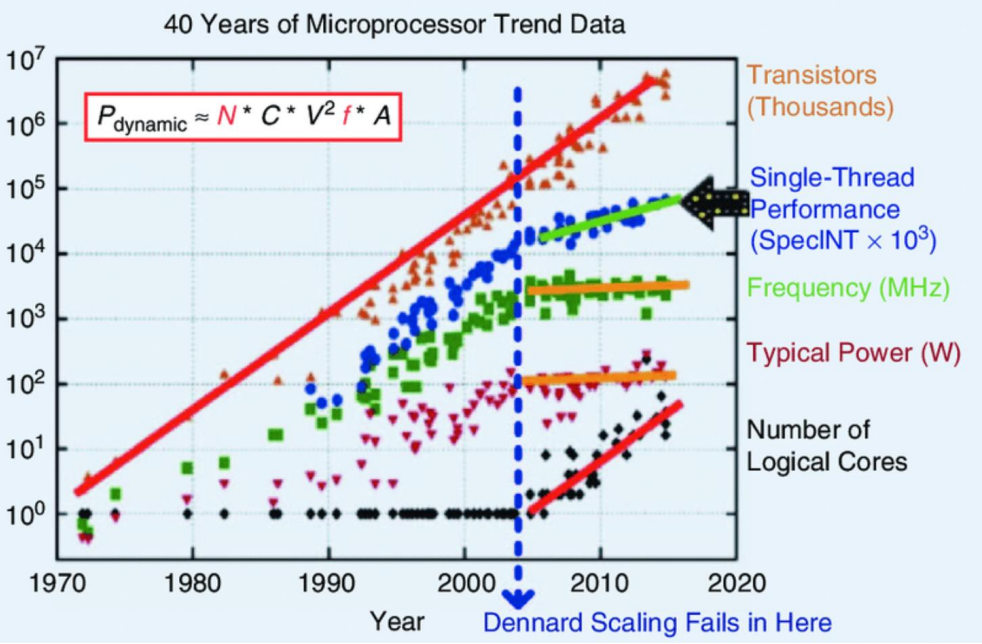


Effects of Dramatic Growth

- **Enhanced the capability available to computer users**
  - highest-performance microprocessors > supercomputer of 20 years earlier
- **Improvement in cost-performance led to new classes of computers**
  - PC, workstations, smartphones, tablets
- **Improvement of semiconductor manufacturing led to the dominance of microprocessor-based computers**
  - Minicomputers → servers made by using microprocessors
- **Allowed modern programmers to trade performance for productivity**
  - C and C++ → Java, Scala, JavaScript, Python
  - Software deployment is changing as well: Software as a Service (SaaS)
- **The nature of applications is also changing**
  - Speech, sound, images, and video are becoming increasingly important

Dennard Scaling Law

- **In 1974, Robert Dennard observed that power density was constant for a given area of silicon**
  - Even as you **increased the number of transistors** because of smaller dimensions of each transistor
  - As transistors get smaller, their power density stays constant, so that the **power use stays in proportion with area**



Dennard scaling ended by 2004

- current and voltage couldn't keep dropping and still maintain the dependability of integrated circuits

End of Dennard Scaling

- **This change forced the microprocessor industry to use multiple processors or cores**
- **In 2004, Intel canceled its high-performance uniprocessor projects**
  - declaring that the road to higher performance would be via multiple processors per chip rather than via faster uniprocessors

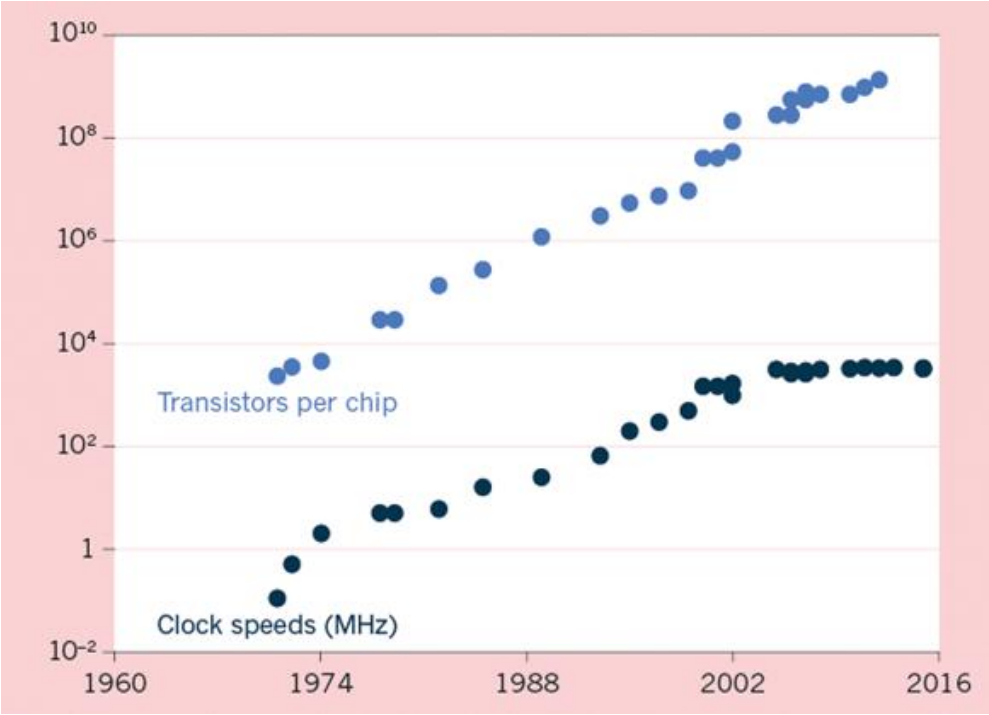
"We are dedicating all of our future product development to multicore designs...
This is a sea change in computing"
-- Paul Otellini, President, Intel (2004)

End of Dennard Scaling

- **This milestone signaled a historic switch: relying solely on ILP → DLP, TLP, RLP**
  - Data-level parallelism (DLP)
  - Thread-level parallelism (TLP)
  - Request-level parallelism (RLP)
- **Compiler and hardware can exploit ILP implicitly**
  - without the programmer's attention
- **DLP, TLP, and RLP are explicitly parallel**
  - requiring the restructuring of the application
  - a major new burden for programmers

摩尔定律

- **Intel创始人之一戈登摩尔（Gordon Moore）提出摩尔定律**
  - 他在1965年发表的一篇论文中指出：晶体管数量每18-24个月翻倍，而其成本减少一半



摩尔定律失效

- 随着时间的推移，摩尔定律的速度开始放缓
  - 晶体管的数量增长已经开始减缓，且成本下降的速度也减慢了
  - 在实际应用中，越来越难以实现更小、更快、更便宜的芯片

摩尔定律失效

- **物理极限**
  - 硅材料有着天然的物理极限，无法进一步缩小晶体管的大小
    - 当晶体管达到一定的大小时，电子将开始隧穿效应，电路的稳定性将受到影响
- **成本效益**
  - 随着晶体管数量的增加，制造芯片的成本也会随之增加
  - 此外，制造更小的晶体管需要更加复杂的制造工艺，这也会增加制造成本
- **散热问题**
  - 在高密度的芯片中，热量会积聚并导致芯片的性能下降

Performance Slow-down

- **Reasons:**
  - **transistors no longer getting much better**
    - because of the slowing of Moore's Law and the end of Dennard scaling
  - the **unchanging power budgets** for microprocessors
  - the **replacement** of the single power-hungry processor with several energy efficient processors
  - the **limits to multiprocessing** to achieve Amdahl's Law
- **The only path to improve energy-performance-cost is specialization**
  - Future microprocessors will include several domain-specific cores that perform only one class of computations well
  - but they do so remarkably better than general-purpose cores

1.3 Defining Computer Architecture

"Old" View of Computer Architecture

- **Old view: computer architecture generally referred to only Instruction Set Architecture (ISA) design**
- **ISA refers to the actual programmer-visible instruction set**
  - The ISA serves as the boundary between the software and hardware
  - Besides instructions, the ISA defines items in the computer that are available to a program
    - e.g., data types, registers, addressing modes, and memory
- **This view is incorrect**
  - The architect's or designer's job is much more than instruction set design

Quick Review of ISA

- **RISC-V is a modern RISC instruction set developed at UC Berkeley**
  - free and open, elegant example of the RISC architectures
  - more than 60 companies have joined the RISC-V foundation, including AMD, Google, HP Enterprise, IBM, Microsoft, Nvidia...
- **RISC-V inherits its ancestors' good ideas**
  - a large set of registers
  - easy-to-pipeline instructions
  - a lean set of operations
- **We use the integer core ISA of RISC-V as the example ISA**

(1) Class of ISA

- **Nearly all ISAs today are classified as general-purpose register architectures**
  - operands are either registers or memory locations
- **Two popular versions of this class:**
  - **register-memory ISAs** (e.g., 80x86)
    - access memory as part of many instructions
  - **load-store ISAs** (e.g., ARMv8, RISC-V)
    - access memory only with load or store instructions
    - All ISAs announced since 1985 are load-store

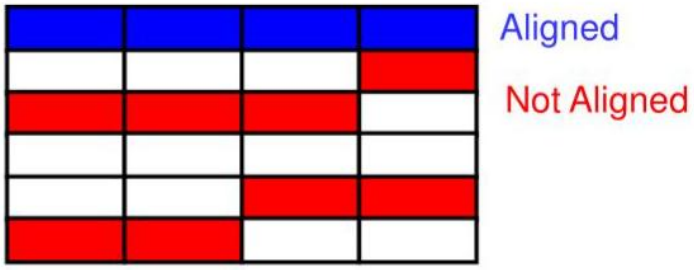
Instruction Set Architecture

- **RISC-V registers**
  - 32 g.p., 32 f.p.

Register	Name	Use	Saver
x0	zero	constant 0	n/a
x1	ra	return addr	caller
x2	sp	stack ptr	callee
x3	gp	gbl ptr	
x4	tp	thread ptr	
x5-x7	t0-t2	temporaries	caller
x8	s0/fp	saved/ frame ptr	callee
x9	s1	saved	callee
x10-x17	a0-a7	arguments	caller
x18-x27	s2-s11	saved	callee
x28-x31	t3-t6	temporaries	caller
f0-f7	ft0-ft7	FP temps	caller
f8-f9	fs0-fs1	FP saved	callee
f10-f17	fa0-fa7	FP arguments	callee
f18-f27	fs2-fs21	FP saved	callee
f28-f31	ft8-ft11	FP temps	caller

(2) Memory Addressing

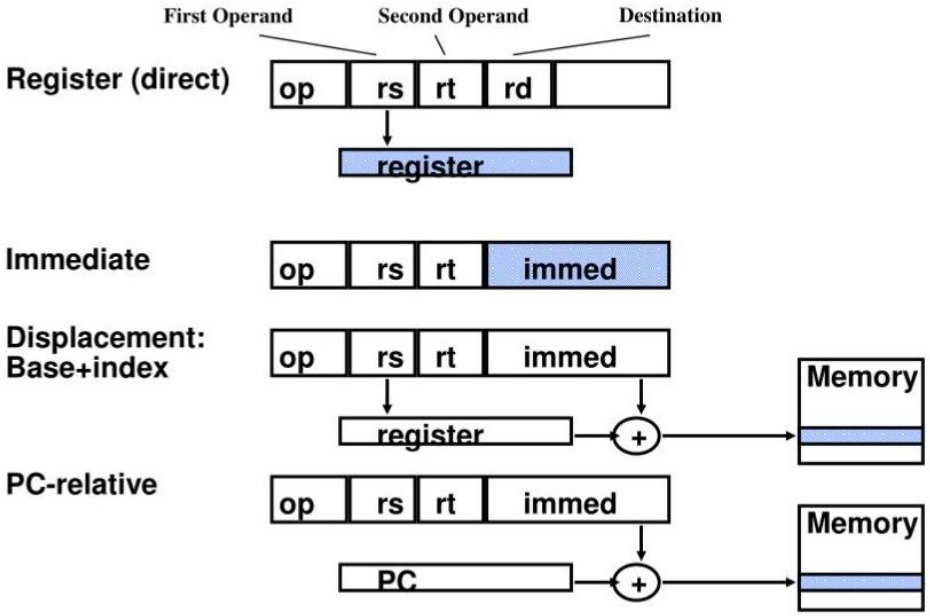
- **Virtually all desktop and server computers use byte addressing to access memory operands**
  - including 80x86, ARMv8, and RISC-V
- **Some architectures require that objects must be aligned**
  - Example: ARMv8
  - An access to an object of size s bytes at byte address A is aligned if A mod s=0
- **80x86 and RISC-V do not require alignment**
  - but accesses are generally faster if operands are aligned



(3) Addressing Modes

- **Addressing modes specify the address of a memory object**
- **RISC-V addressing modes:**
  - **Register**
  - **Immediate** (for constants)
  - **Displacement**:

    a constant offset is added to a register to form the memory address



(4) Types and sizes of operands

- **x86, ARMv8, and RISC-V support different operand sizes:**
  - 8-bit (ASCII character)
  - 16-bit (Unicode character or half word)
  - 32-bit (integer or word)
  - 64-bit (double word or long integer)
  - IEEE 754 floating point in 32-bit (single precision) and 64-bit (double precision).

(5) Operations

- **General categories of operations include:**
  - data transfer
  - arithmetic logical
  - control
  - floating point
- **RISC-V is a simple and easy-to-pipeline ISA**

Instruction type/opcode
Data transfers
lb, lbu, sb
lh, lhu, sh
lw, lwu, sw
ld, sd
flw, fld, fsw, fsd
fmv._.x, fmv.x._
csrrw, csrrwi, csrrs, csrrsi, csrrc, csrrci
Arithmetic/logical
add, addi, addw, addiw
sub, subw
mul, mulw, mulh, mulhsu, mulhu
div, divu, rem, remu
divw, divuw, remw, remuw
and, andi
or, ori, xor, xori

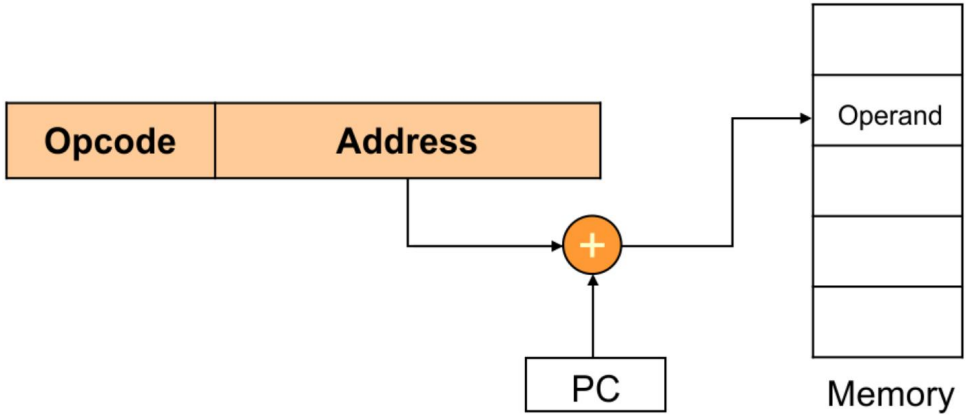
Integer instructions for RISC-V

Instruction type/opcode
Floating point
fadd.d, fadd.s
fsub.d, fsub.s
fmul.d, fmul.s
fmadd.d, fmadd.s, fnmadd.d, fnmadd.s
fmsub.d, fmsub.s, fnmsub.d, fnmsub.s
fdiv.d, fdiv.s
fsqrt.d, fsqrt.s
fmax.d, fmax.s, fmin.d, fmin.s
fcvt._._, fcvt._._u, fcvt._u._
feq._, flt._,fle._
fclass.d, fclass.s
fsgnj._, fsgnjn._, fsgnjx._

Floating point instructions for RISC-V

(6) Control flow instructions

- **Virtually all ISAs support:**
  - conditional branches
  - unconditional jumps
  - procedure calls
  - returns

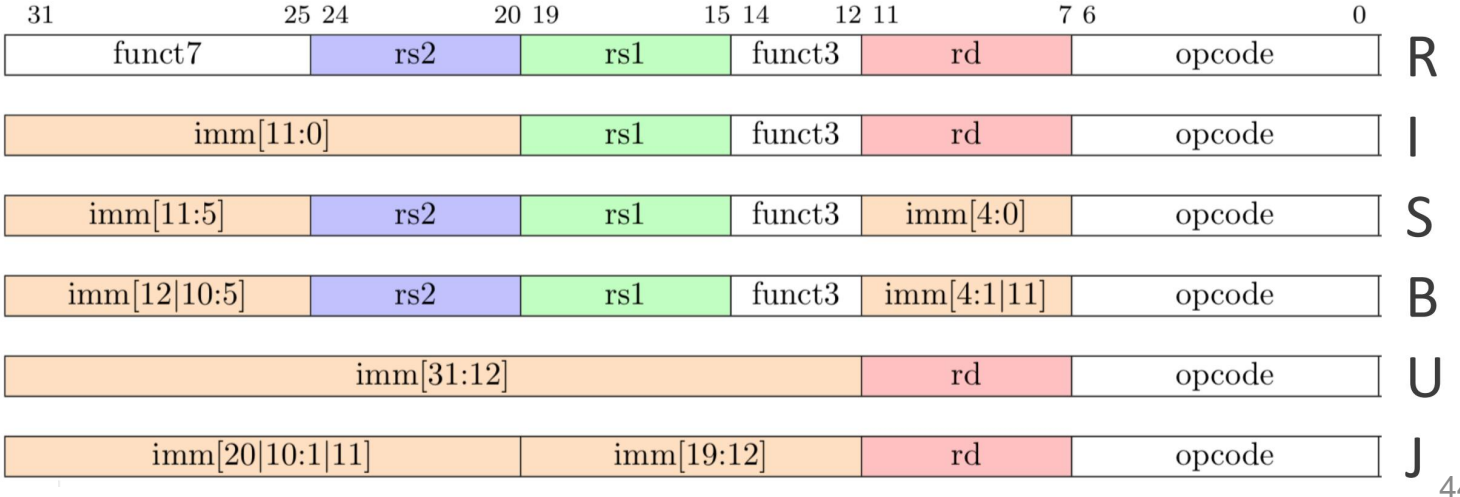


- **PC-relative addressing**
  - the branch address is specified by an address field that is added to the PC

(7) Encoding an ISA

- **There are two basic choices on encoding:**
  - **fixed length:**
    - ARMv8 and RISC-V instructions are 32 bits long
    - simplifies instruction decoding
  - **variable length:**
    - 80x86 encoding, ranging from 1 to 18 bytes
    - take less space than fixed-length instructions

RISC-V ISA formats



"Real" Computer Architecture

- **Designing the organization and hardware to meet goals and functional requirements**
  - Includes **ISA, microarchitecture, hardware**
- **Concerned with balancing the performance, efficiency, cost, and reliability of a computer system**
  - **instruction set architecture** can be used to illustrate the balance of these competing factors
    - longer and more complex instructions take longer for the processor to decode and can be more costly to implement effectively

"Real" Computer Architecture

- **The implementation of a computer has two components: *organization* and *hardware***
- ***Organization* includes the high-level aspects of a computer's design**
  - such as memory system, memory interconnect, and the design of the internal processor or CPU

“Real” Computer Architecture

- ***Microarchitecture:*** **a.k.a "computer organization"**
  - Describes how a particular processor will implement the ISA
  - E.g., two processors with the same ISA but different organizations are the AMD Opteron and the Intel Core i7

- **Both processors implement the 80x86 instruction set**

- **But they have very different pipeline and cache organizations!**

"Real" Computer Architecture

- ***Hardware*** **refers to the specifics of a computer**
  - including detailed *logic design* and *packaging* technology of the computer
- **Often a line of computers contains computers with identical ISA and very similar organizations, but they differ in the detailed hardware implementation**
  - For example, the Intel Core i7 and the Intel Xeon E7
  - nearly identical but offer different clock rates and different memory systems
  - making the Xeon E7 more effective for server computers

1.4 Trends in Technology

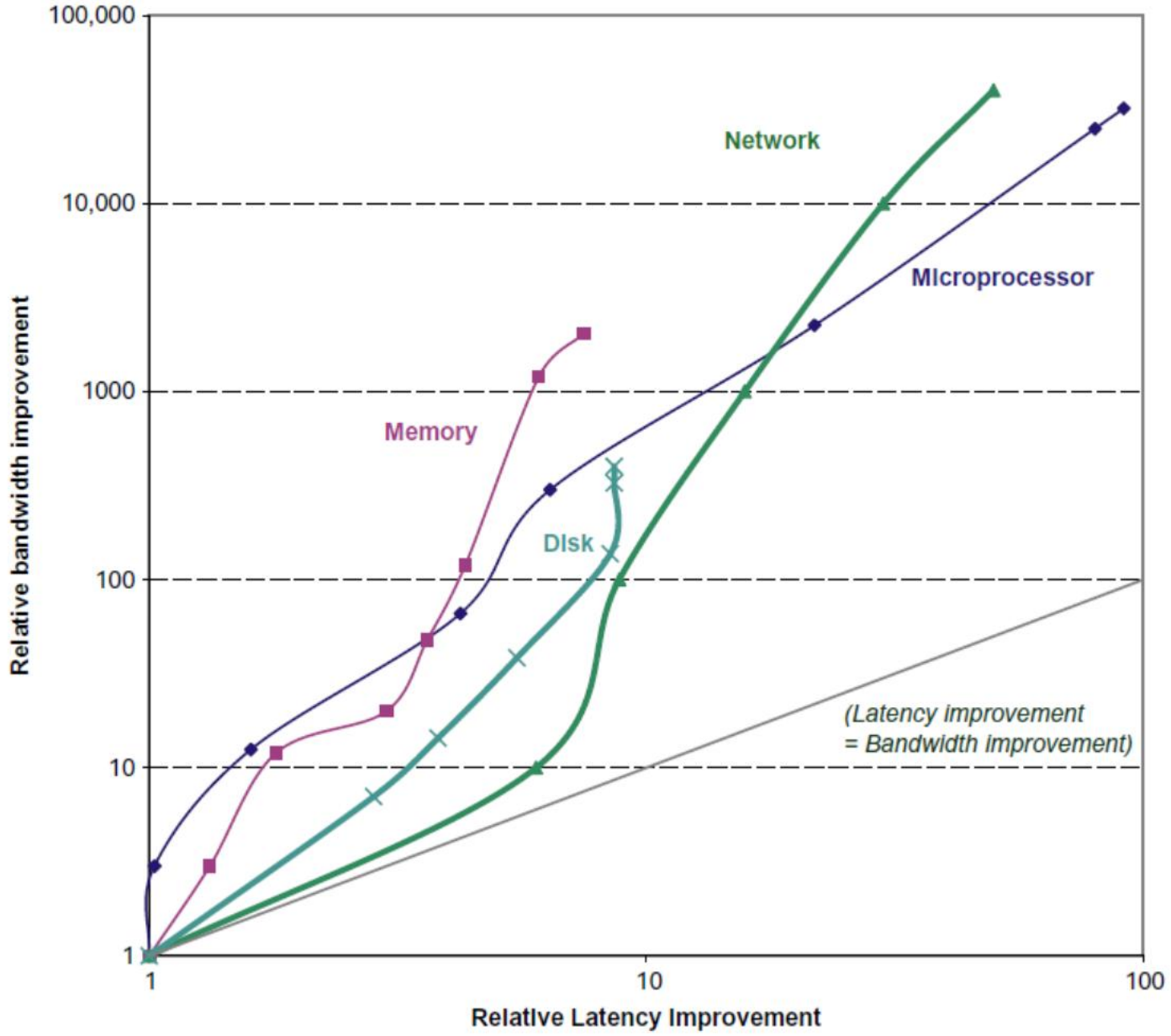
Trends in Technology

- **Integrated circuit technology (Moore's Law is no more)**
  - Transistor density: 35%/year
  - Die size: 10-20%/year
  - Integration overall: 40-55%/year
- **DRAM capacity: 25-40%/year (slowing)**
  - 8 Gb (2014), 16 Gb (2019) → 32 Gb
- **Flash capacity: 50-60%/year**
  - 8-10X cheaper/bit than DRAM
- **Magnetic disk capacity: recently slowed to 5%/year**
  - Density increases may no longer be possible, maybe increase from 7 to 9 platters
  - 8-10X cheaper/bit than Flash
  - 200-300X cheaper/bit than DRAM

Bandwidth and Latency

- **Bandwidth (or throughput)**
  - Total **amount of work** done in a given time
  - 32,000-40,000X improvement for processors
  - 300-1200X improvement for memory and disks

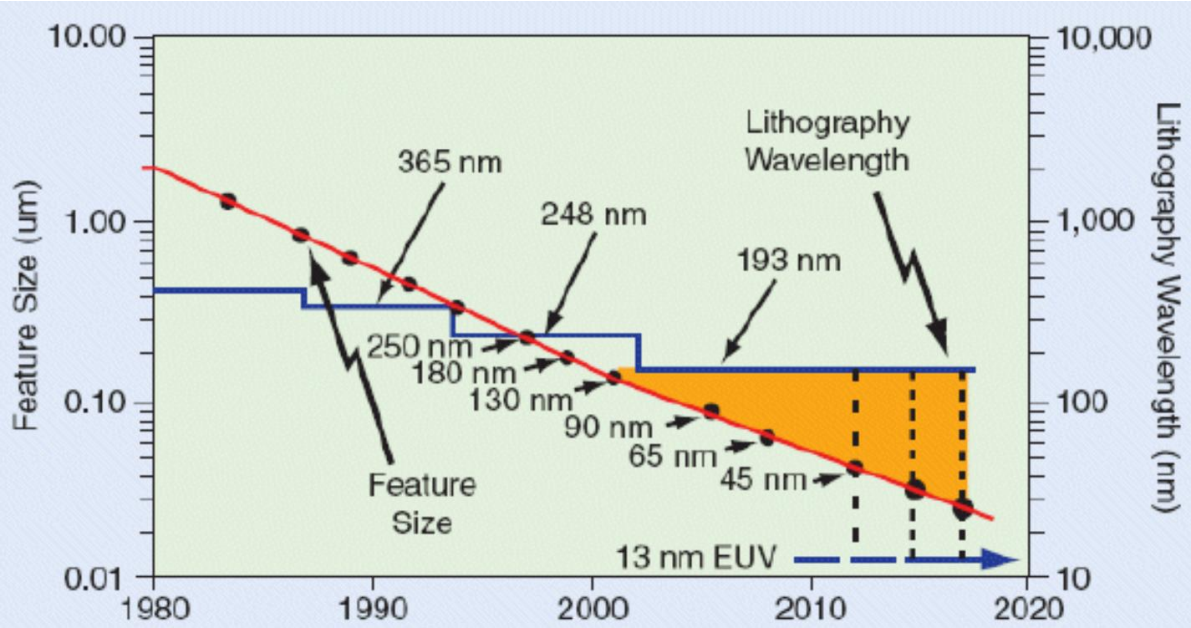
- **Latency (or response time)**
  - **Time** between start and completion of an event
  - 50-90X improvement for processors
  - 6-8X improvement for memory and disks



Relative improvement in bandwidth and latency for technology milestones for microprocessors, memory, networks, and disks

Scaling of Transistor Performance and Wires

- Integrated circuit processes are characterized by the **feature size**
- **Feature size** is the **minimum size of a transistor or a wire** in either the **x or y dimension**
  - Feature sizes decreased from 10 μm in 1971 to 0.016 μm (i.e., 16 nm) in 2017



Scaling of Transistor Performance and Wires

- **The density of transistors increases quadratically with a linear decrease in feature size**
- **But wire delay does not improve with feature size!**
  - Scales poorly compared to transistor performance
  - become a major design obstacle for large integrated circuits
- **Larger and larger fractions of the clock cycle have been consumed by the propagation delay of signals on wires**
  - but power now plays an even greater role than wire delay

1.5 Trends in Power and Energy in Integrated Circuits

Power and Energy: A Systems Perspective

- **From the viewpoint of a system designer, there are three primary concerns**
  - what is the maximum power a processor ever requires?
  - what is the sustained power consumption?
  - energy and energy efficiency

Thermal Design Power (TDP)

- **Sustained power consumption is widely called the *thermal design power* (TDP)**
  - actual average power < TDP < peak power (1.5x)
  - TDP determines the cooling requirement
  - A typical power supply for a system is typically sized to exceed the TDP
  - a cooling system is usually designed to match or exceed TDP

Energy vs. Power

- **Power is simply energy per unit time**
  - 1 watt = 1 joule per second
- **Which metric is the right one for comparing processors?**
  - In general, energy is always a better metric
    - because it is tied to a specific task and the time required for that task
  - The energy to complete a workload = the average power x the execution time for the workload

Energy vs. Power

- **When is power consumption a useful measure?**
- **The primary legitimate use is as a constraint**
  - E.g., an air-cooled chip might be limited to 100 W

Energy and Power within a Microprocessor

- **For CMOS chips, the primary energy consumption has been in switching transistors, also called *dynamic energy***
- **The energy required per transistor**
  - energy of pulse of the logic transition of 0→1→0, or 1 → 0→1:

$$\text{Energy}_{\text{dynamic}} \propto \text{Capacitive load} \times \text{Voltage}^2$$

- **The energy of a single transition (0→1 or 1 → 0):**

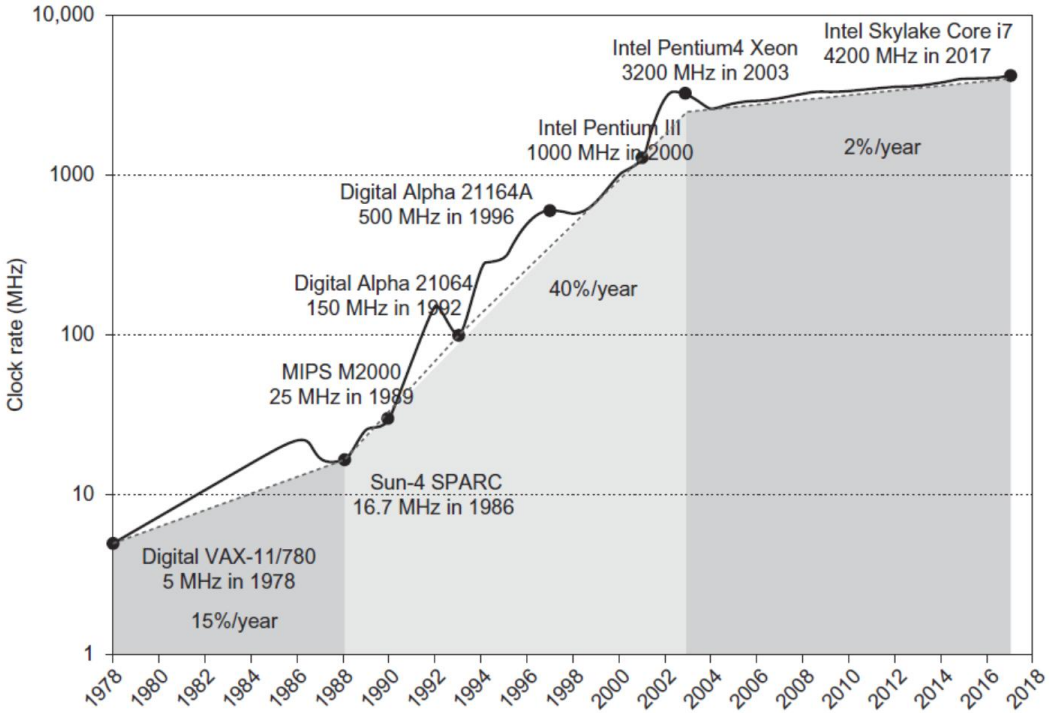
$$\text{Energy}_{\text{dynamic}} \propto 1/2 \times \text{Capacitive load} \times \text{Voltage}^2$$

- **The power required per transistor = energy of a transition x frequency of transitions:**

$$\text{Power}_{\text{dynamic}} \propto 1/2 \times \text{Capacitive load} \times \text{Voltage}^2 \times \text{Frequency switched}$$

Energy and Power within a Microprocessor

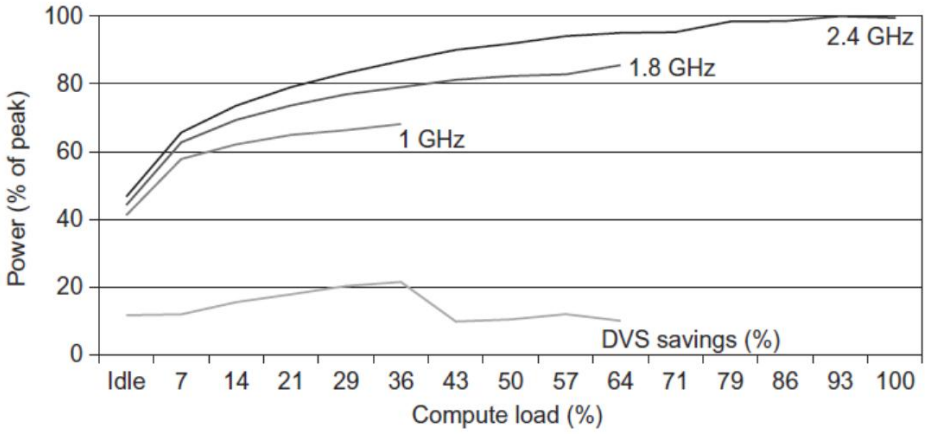
- **Dynamic power and energy are greatly reduced by lowering the voltage**
- **Clock frequency growth should be slow down if we cannot reduce voltage or increase power per chip**



- Intel 80386 consumed ~ 2 W
- 3.3 GHz Intel Core i7 consumes 130 W
- Heat must be dissipated from 1.5 x 1.5 cm chip
- This is the limit of what can be cooled by air

Reducing Power

- **Modern microprocessors offer many techniques to improve energy efficiency:**
  - **1. Do nothing well (什么也不做)**
    - Turn off the clock of inactive modules to save energy
  - **2. Dynamic voltage-frequency scaling (DVFS)**
    - Offer a few clock frequencies and voltages in which to operate that use lower power and energy



The overall server power savings is about **10%–15%**

Reducing Power

- **Modern microprocessors offer many techniques to improve energy efficiency:**
  - **3. Design for the typical case**
    - Given that PMDs and laptops are often idle, **memory and storage** offer low power modes to save energy
    - must return to fully active mode to read or write
  - **4. Overclocking (超频)**
    - **Turbo mode （加速模式，或性能模式）**: the chip decides that it is safe to run at a higher clock rate for a short time until temperature starts to rise
    - E.g., turn off all cores but one and run it faster

Static Power

- **Static power is becoming an important issue, because leakage current （泄漏电流） flows even when a transistor is off**

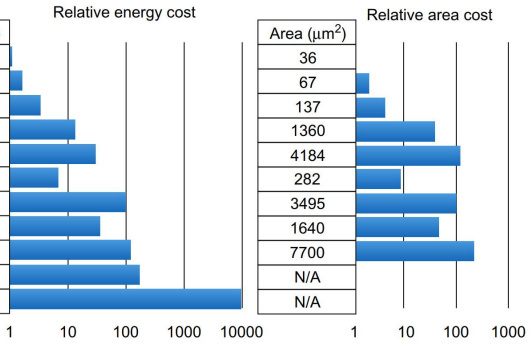
$$\text{Power}_{\text{static}} \propto \text{Current}_{\text{static}} \times \text{Voltage}$$

- **Leakage can be as high as 50% for high-performance chips**
  - large SRAM caches need power to maintain storage values
- **Power gating （电源门控）：**
  - turning off the power supply to inactive modules in order to control loss because of leakage
- **Race-to-halt （竞相暂停）：**
  - the processor operates at full speed during active periods, completing tasks as quickly as possible
  - when there is no immediate work to be done, the processor quickly transitions into a low-power state, effectively halting its activity

Energy Cost and Area Cost

- **Dark silicon: refers to the phenomenon where portions of a computer chip remain inactive due to power consumption and heat dissipation limitations**
  - As transistor density increases, power constraints restrict the number of active transistors that can operate simultaneously
  - As a result, certain areas of the chip remain "**dark**" or powered off at any given moment
- **Energy / area cost of building blocks of a modern computer**
  - A floating-point addition uses **30 times** as much energy as an 8-bit integer add
  - A small SRAM is **125 times** more energy-efficient than DRAM
  - demonstrates the importance of careful uses of **caches and memory buffers**

Operation:	Energy (pJ)	Area (μm^2)
8b Add	0.03	36
16b Add	0.05	67
32b Add	0.1	137
16b FB Add	0.4	1360
32b FB Add	0.9	4184
8b Mult	0.2	282
32b Mult	3.1	3495
16b FB Mult	1.1	1640
32b FB Mult	3.7	7700
32b SRAM Read (8KB)	5	N/A
32b DRAM Read	640	N/A



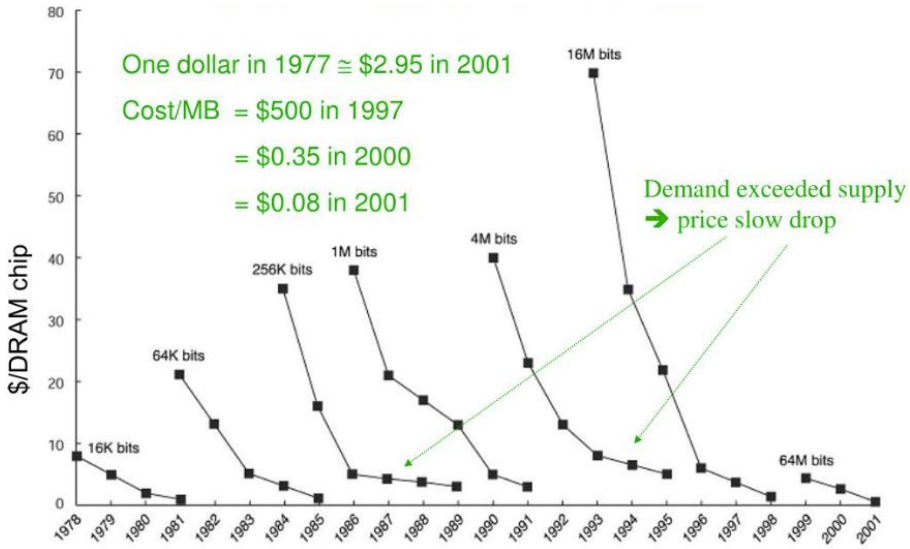
Shift in Computer Architecture

- **Domain-specific processors:**
  - save energy by reducing wide floating-point operations
  - deploying special-purpose memories to reduce accesses to DRAM
  - provide 10–100 more integer arithmetic units than a traditional processor
- **Such processors perform only a limited set of tasks**
  - But they perform them faster and more energy efficiently than a general-purpose processor

1.6 Trends in Cost

Impact of Time, Volume, and Commoditization

- **The underlying principle that drives costs down is learning curve**
  - manufacturing costs decrease over time
  - The learning curve itself is best measured by change in yield (良率)
- **Yield: the percentage of manufactured devices that survives the testing procedure**



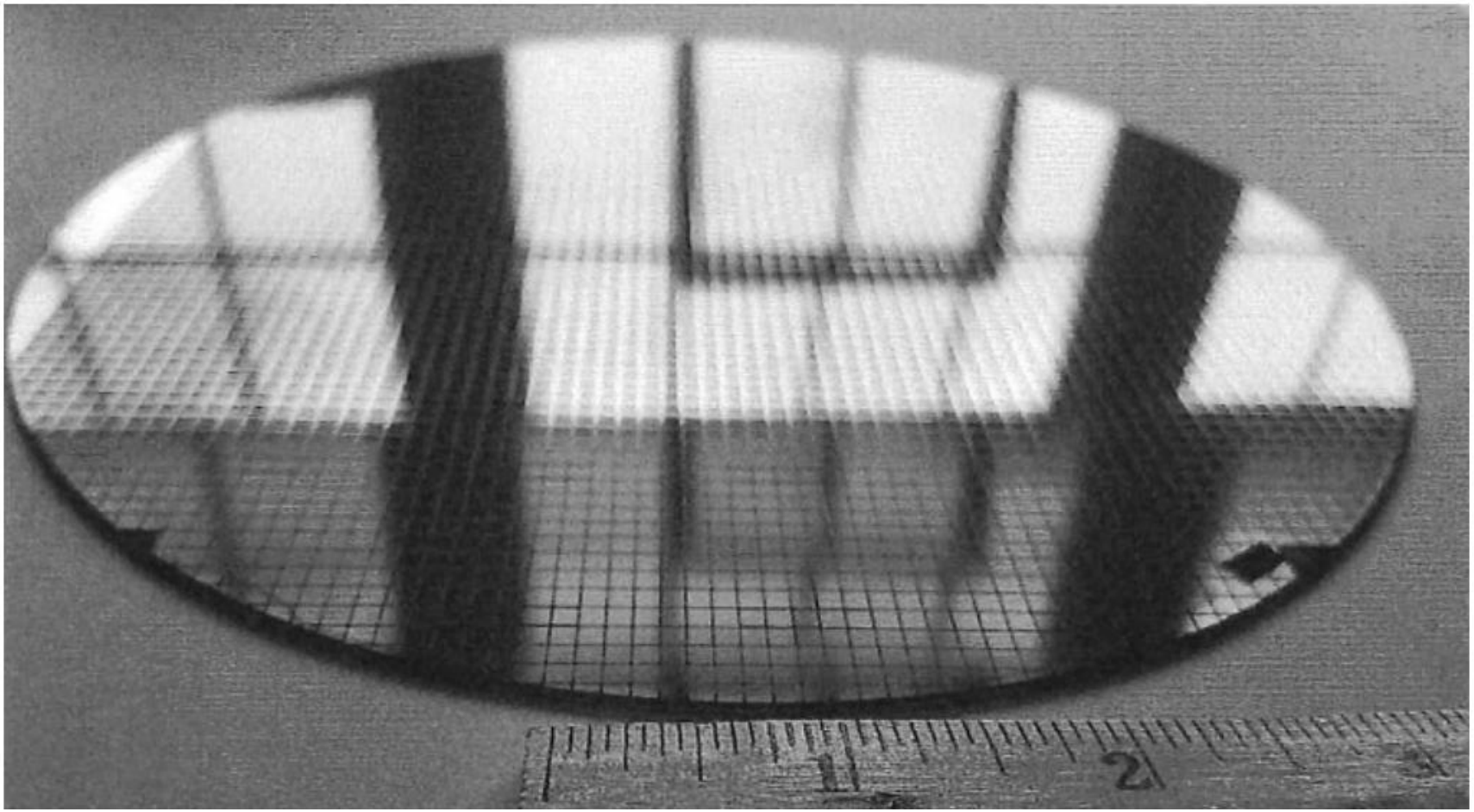
Example: the price per MB of DRAM

- dropped over the long term
- DRAMs tend to be priced in close relationship to cost

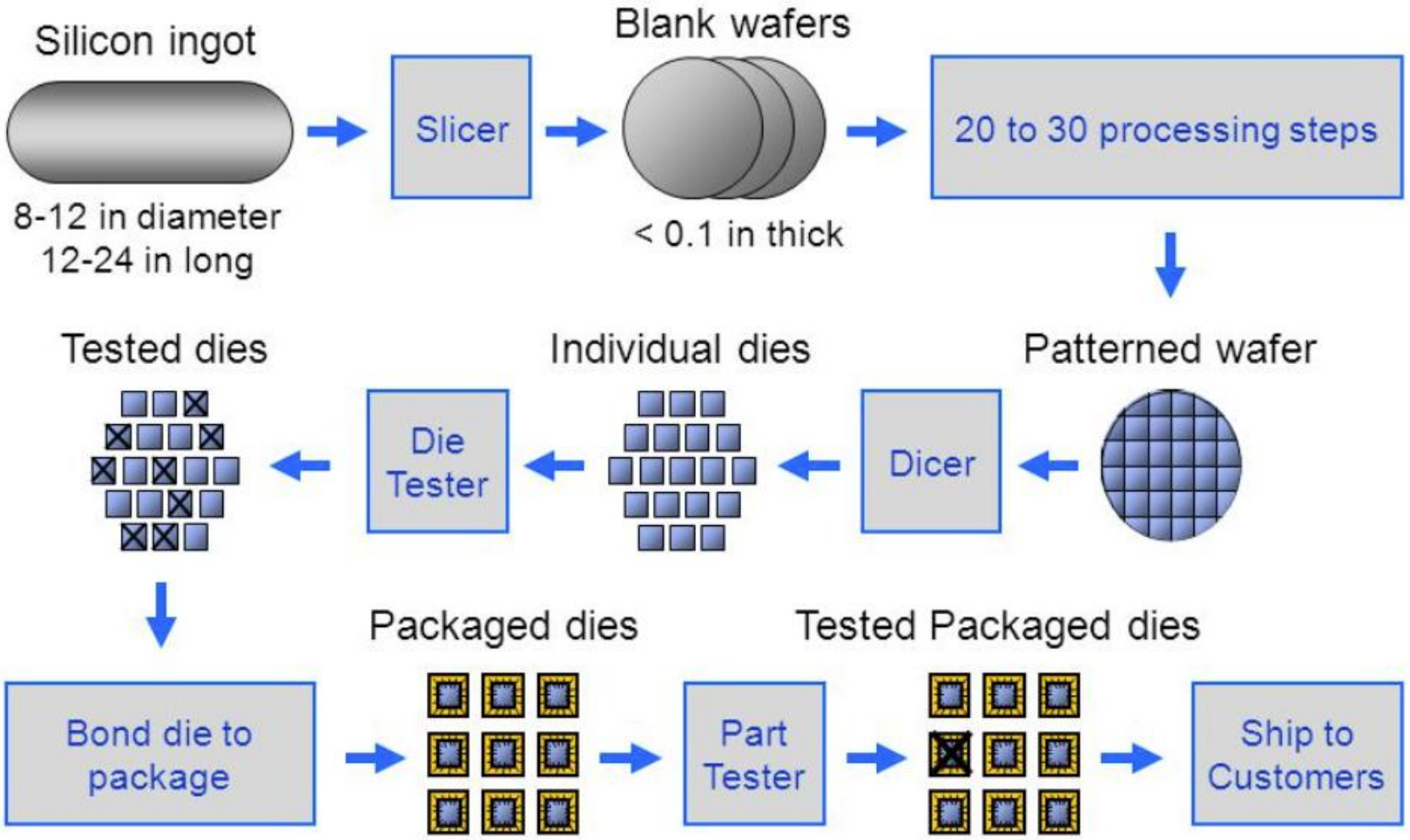
Impact of Time, Volume, and Commoditization

- **Volume is a second key factor in determining cost**
  - decrease the time needed to get through the learning curve
  - decrease cost because it increases purchasing and manufacturing efficiency
    - costs decrease about **10%** for each doubling of volume
  - decrease the amount of development costs
- **Competition decreases the gap between cost and selling price, but it also decreases cost**
  - competition among the suppliers of the components
  - volume efficiencies the suppliers can achieve

Cost of an Integrated Circuit



IC Manufacturing Process



Cost of an Integrated Circuit

- **Integrated circuit**

$$\text{Cost of integrated circuit} = \frac{\text{Cost of die} + \text{Cost of testing die} + \text{Cost of packaging and final test}}{\text{Final test yield}}$$

$$\text{Cost of die} = \frac{\text{Cost of wafer}}{\text{Dies per wafer} \times \text{Die yield}}$$

$$\text{Dies per wafer} = \frac{\pi \times (\text{Wafer diameter}/2)^2}{\text{Die area}} - \frac{\pi \times \text{Wafer diameter}}{\sqrt{2 \times \text{Die area}}}$$

$$\text{Die yield} = \text{Wafer yield} \times 1/(1 + \text{Defects per unit area} \times \text{Die area})^N$$

Defects per unit area = 0.016-0.057 defects per square cm (2010)

N = process-complexity factor = 11.5-15.5 (40 nm, 2010)

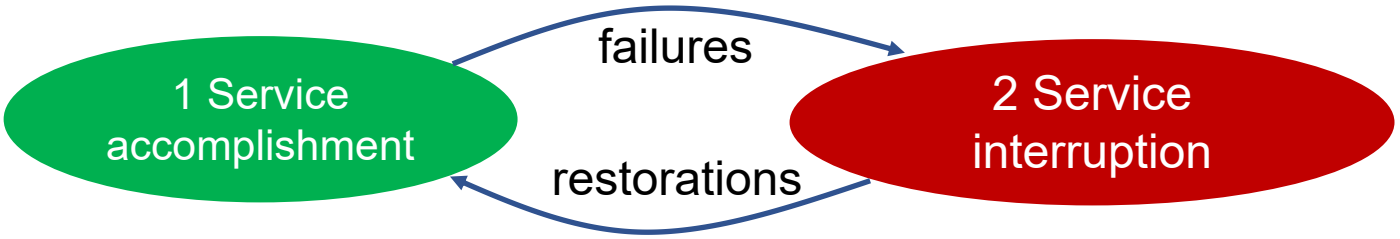
Cost of an Integrated Circuit

- **What should a computer designer remember about chip costs?**
  - The manufacturing process dictates the *wafer cost*, *wafer yield*, and *defects per unit area*
  - the **sole control** of the designer is **die area**
  - The **cost per die** grows roughly as the **square of the die area**
- **The computer designer affects die size, and thus cost**
  - by what functions are included on or excluded from the die
  - by the number of I/O pins

1.7 Dependability
(可信任度)

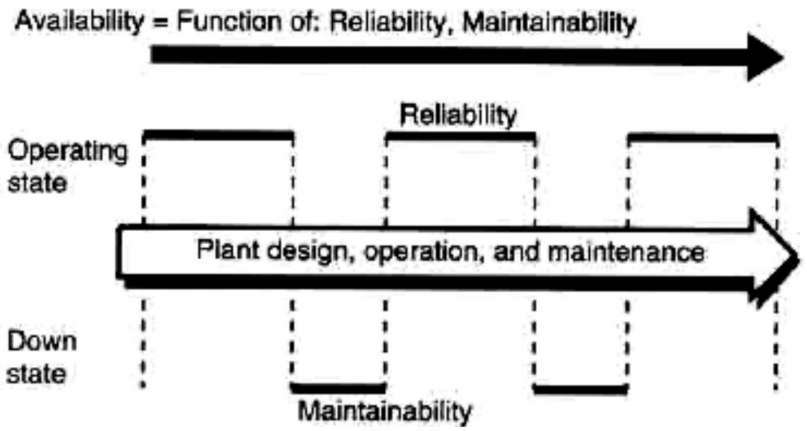
Dependability

- **Infrastructure providers started offering service level agreements (SLAs)**
  - an SLA could be used to decide whether the system was up or down
- **Systems alternate between two states of service with respect to an SLA:**
  - **1. Service accomplishment**, where the service is delivered as specified
  - **2. Service interruption**, where the delivered service is different from the SLA
- **Transitions between these two states are caused by failures (from state 1 to state 2) or restorations (2 to 1)**



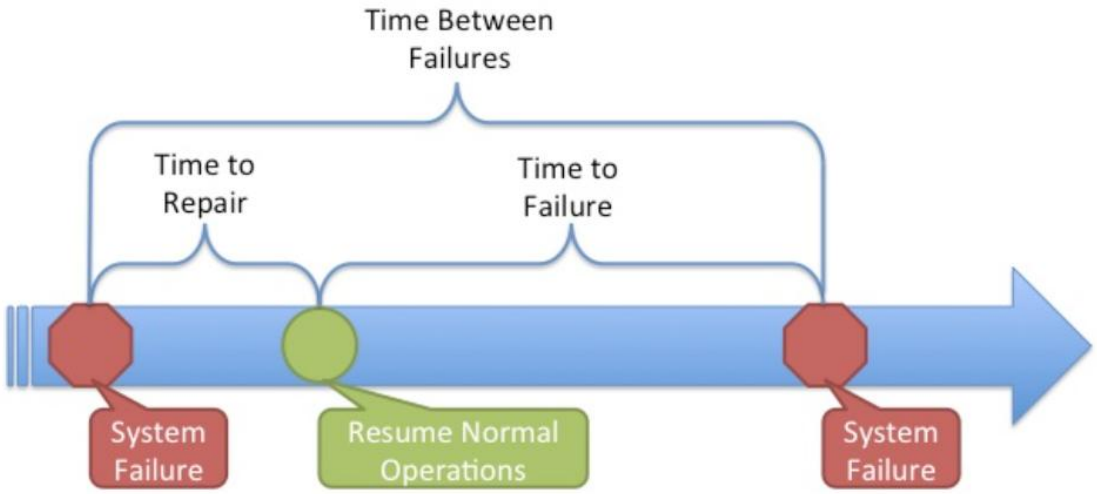
Dependability

- **Quantifying these transitions leads to two main measures of dependability:**
  - **Module reliability:** a measure of the continuous service accomplishment (or, equivalently, of the time to failure) from a reference initial instant
  - **Module availability:** a measure of the service accomplishment with respect to the alternation between the two states of accomplishment and interruption



Measures of Module Reliability

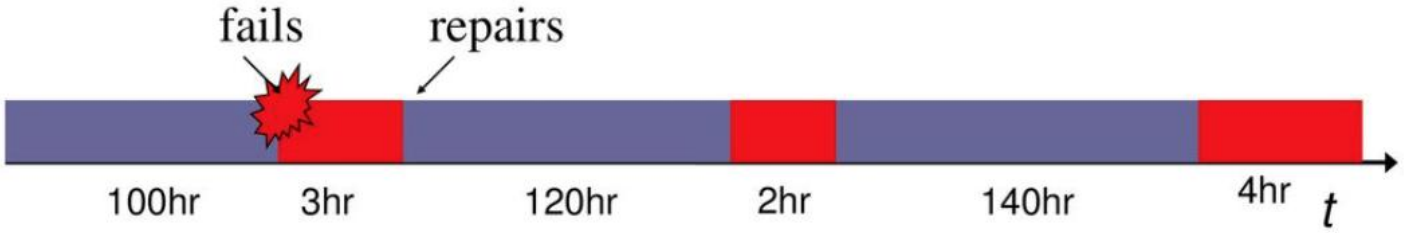
- **Mean time to failure (MTTF) is a reliability measure**
- **The reciprocal of MTTF is a rate of failures**
  - generally reported as **failures per billion hours of operation**, or **FIT** (for failures in time).
  - an MTTF of 1,000,000 hours equals 1000 FIT.
- **Service interruption is measured as mean time to repair (MTTR)**
- **Mean time between failures (MTBF) is the sum of MTTF+MTTR**



Measures of Module Availability

- **For non-redundant systems with repair, module availability is:**

$$\text{Module availability} = \frac{\text{MTTF}}{(\text{MTTF} + \text{MTTR})}$$



$$A = MTTF / (MTTF+MTTR)$$

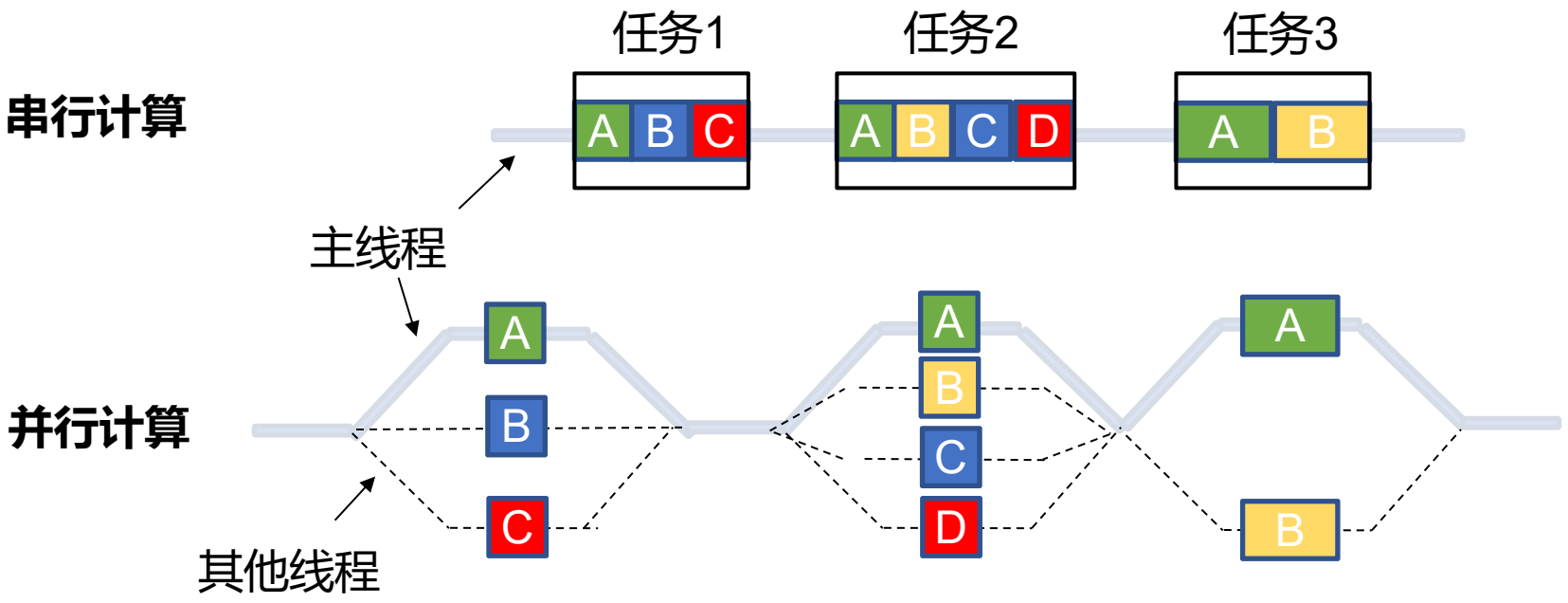
MTTF = (100 + 120 + 140) / 3 = 120 (hr/case)
MTTR = (3 + 2 + 4) / 3 = 3 (hr/case)

A = 120 / 120+3
= 120/123 = 0.975

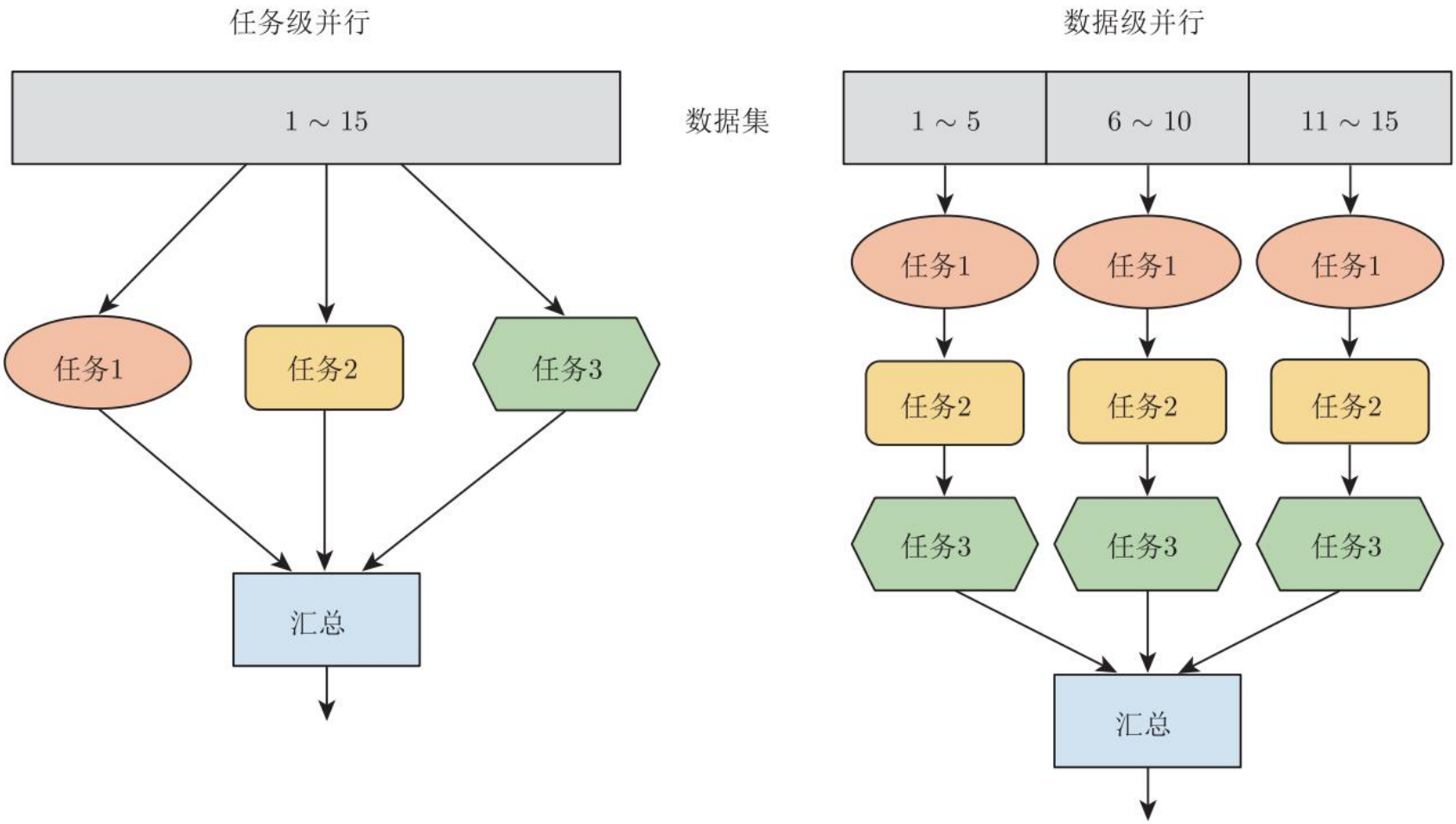
（二）并行计算与并行计算机

并行计算

- 并行计算通常是指把一个大规模的计算问题划分为若干规模较小的部分，然后可以在多个互连的处理器上同时执行求解的计算模式

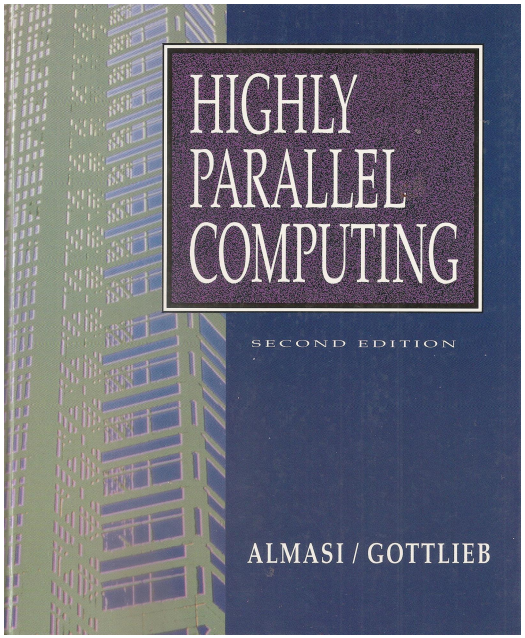


并行计算


任务级并行
1 ~ 15
任务1
任务2
任务3
汇总
数据集
数据级并行
1 ~ 5
6 ~ 10
11 ~ 15
任务1
任务1
任务1
任务2
任务2
任务2
任务3
任务3
任务3
汇总

并行计算机

- ***"A parallel computer is a collection of processing elements that can communicate and cooperate to solve a large problem fast."***



-- Almasi & Gottlieb

Highly Parallel Computing

1989

并行计算机

- ***A parallel computer is a collection of processing elements that can communicate and cooperate to solve a large problem fast. [Almasi&Gottlieb]***

- **"collection of processing elements (PEs)"**
  - How many? How powerful each? Scalability?
  - Few very powerful (e.g., Earth Simulator) vs. many small ones (e.g., BlueGene)

并行计算机

- ***A parallel computer is a collection of processing elements that can communicate and cooperate to solve a large problem fast. [Almasi&Gottlieb]***

- **"that can communicate"**
  - How do PEs communicate? (shared memory vs. msg passing)
  - Interconnection network (bus, multistage, crossbar, …)
  - Evaluation criteria: cost, latency, throughput, scalability, and fault tolerance

并行计算机

- ***A parallel computer is a collection of processing elements that can communicate and cooperate to solve a large problem fast. [Almasi&Gottlieb]***

- **“and cooperate”**
  - **Synchronization** allows sequencing of operations to ensure correctness
  - **Granularity** up => parallelism down, communication down, overhead down
  - **Autonomy**
    - SIMD vs. MIMD (single/multiple instruction streams)

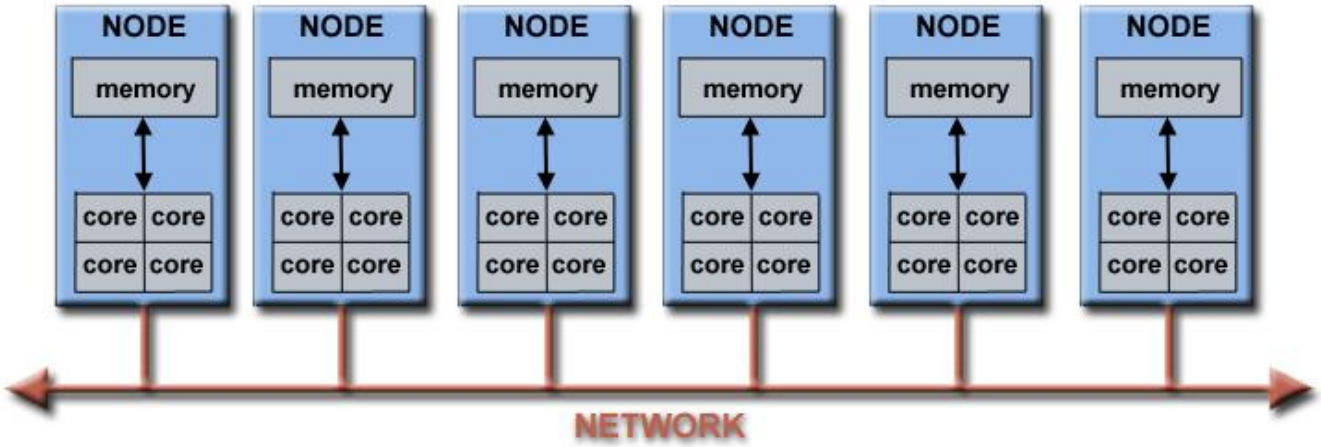
并行计算机

- ***A parallel computer is a collection of processing elements that can communicate and cooperate to solve a large problem fast. [Almasi&Gottlieb]***

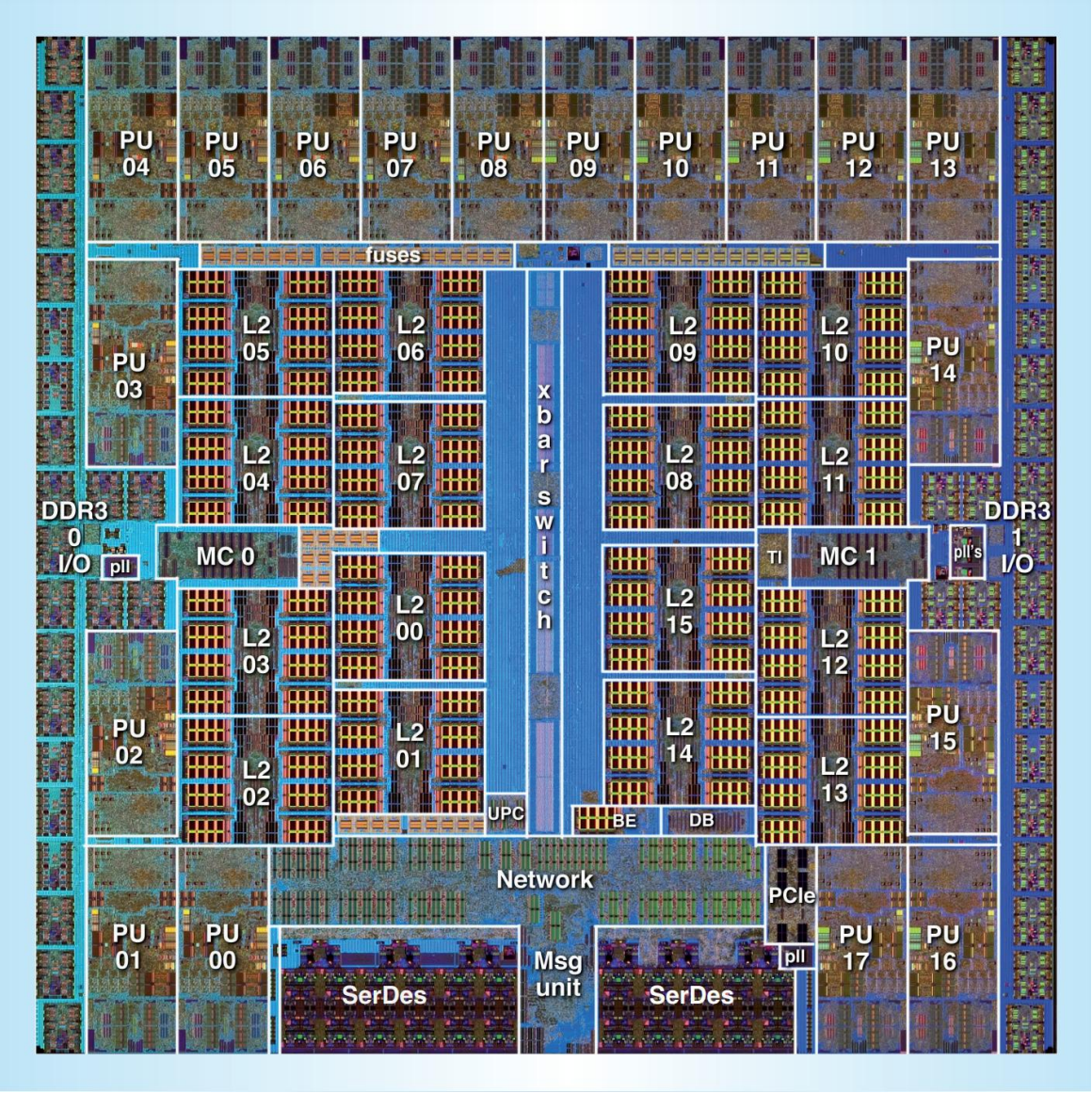
- **“solve a large problem fast”**
  - General vs. special purpose machine?
    - Any machine can solve certain problems well

并行计算机

- **Virtually all stand-alone computers today are parallel from a hardware perspective:**
  - Multiple functional units (L1 cache, L2 cache, branch, prefetch, decode, floating-point, graphics processing (GPU), integer, etc.)
  - Multiple execution units/cores
  - Multiple hardware threads
- **Networks connect multiple stand-alone computers (nodes) to make larger parallel computer clusters**



IBM BG/Q Compute Chip with 18 cores (PU) and 16 L2



https://www.cs.umd.edu/class/fall2019/cmsc714/readings/Haring-BGQ-chip.pdf

Typical Prallel Computer Cluster


compute node


infiniband switch


management hardware


login / remote partition server node


gateway node

并行计算机体系结构

- **并行计算机就是由多个处理单元组成的计算机系统，这些处理单元相互通信和协作，能快速，高效地求解大型复杂问题。**
- **涉及到的一些问题:**
  - 资源分配:
    - 并行计算机系统有多大?
    - 处理单元的性能如何?
    - 购买价格如何?
  - 数据存取, 通信 以及 同步
    - 各处理单元如何协作及通信?
    - 数据如何在各处理机间传送?
    - 如何同步?
  - 性能与规模
    - 如何将之全部转化为高性能?
    - 如何对其进行评测?

（三）高性能计算机发展史

并行计算机 vs 高性能计算机

- 为了实现高性能，仅靠改进电路工艺来提高单机器件速度是很有限的，而使用并行计算机的方法则更为普遍和有效

- 现今，并行计算机可视作高性能计算机的**同义词**
  - 虽不严格，但已普遍认可

高性能计算机（HPC）= 超级计算机（SC）

- ***H**igh **P**erformance **C**omputer*
  高性能计算机

- 衡量性能的指标是什么？

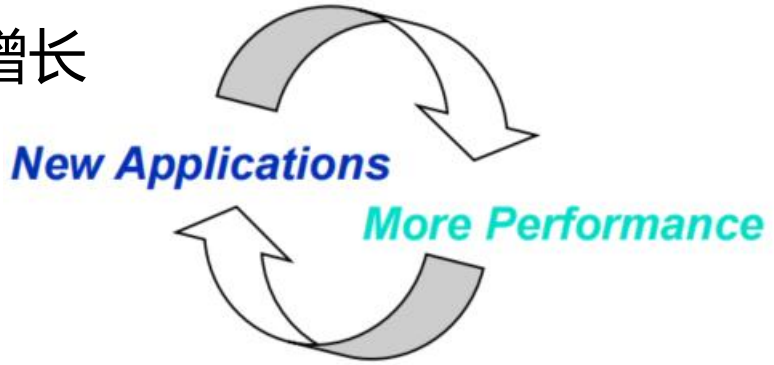
  - 时间
  - 操作次数

高性能计算是不可避免的

- **应用需求：人类对运算速度的追求永无止境**
- **技术进展**
- **结构趋势**
- **经济发展的需要**
- **当前趋势：**
  - 当今的微处理器可以为多处理机系统提供支持
  - 服务器以及工作站向着多处理器方向发展
  - 今后的微处理器将成为多处理器

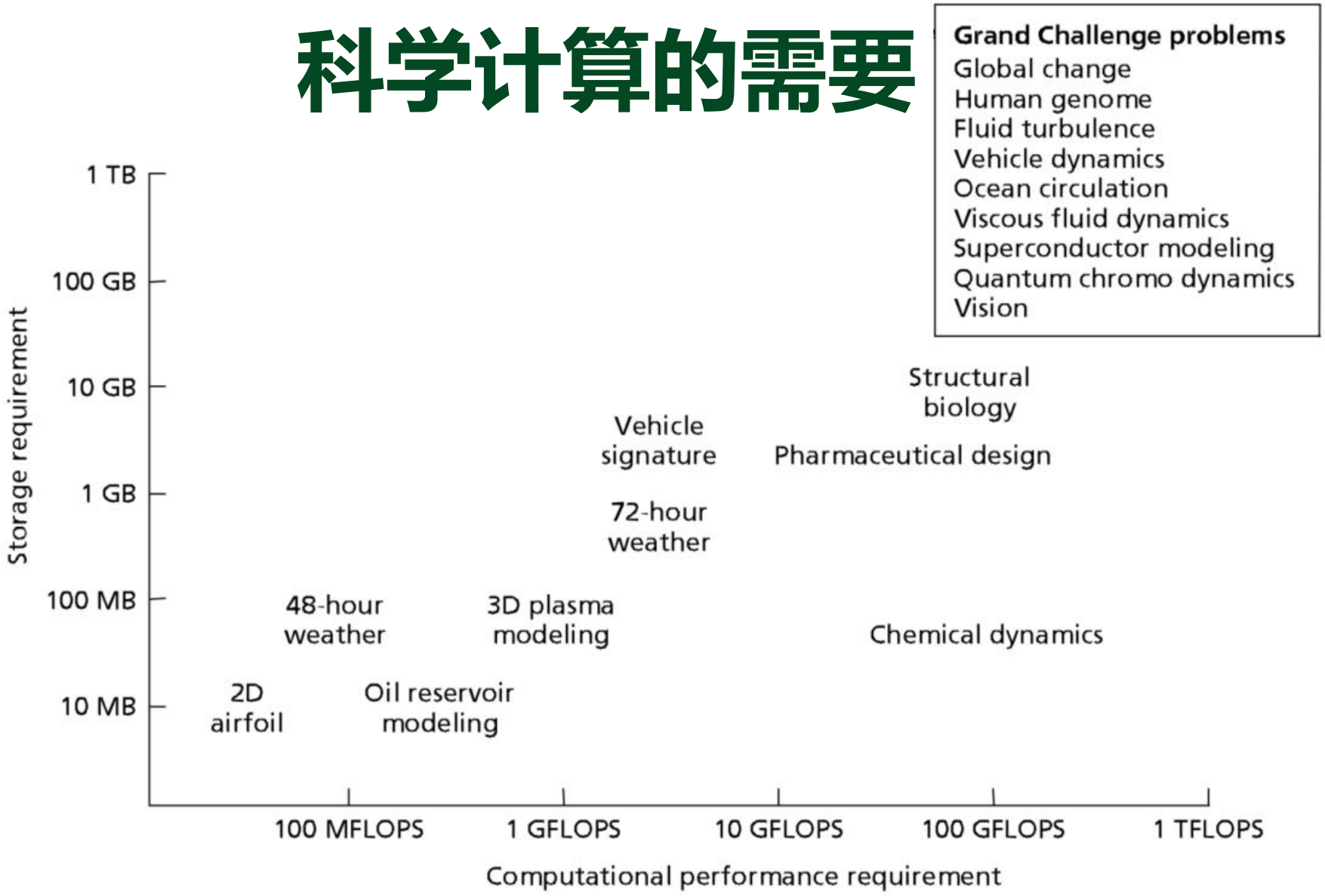
应用需求

- **实际应用中需要性能更强大的硬件系统，而这也使一些新的应用需求得以满足**
  - 微处理器的运算速度呈几何级数增长
  - 并行结构硬件
    - 绝大部分应用需求



- **对性能要求的范围**
  - 随着花费的日益增多需要知道系统的性能范围

科学计算的需要



美国HPCC计划重大挑战性课题的需求

FLOPS

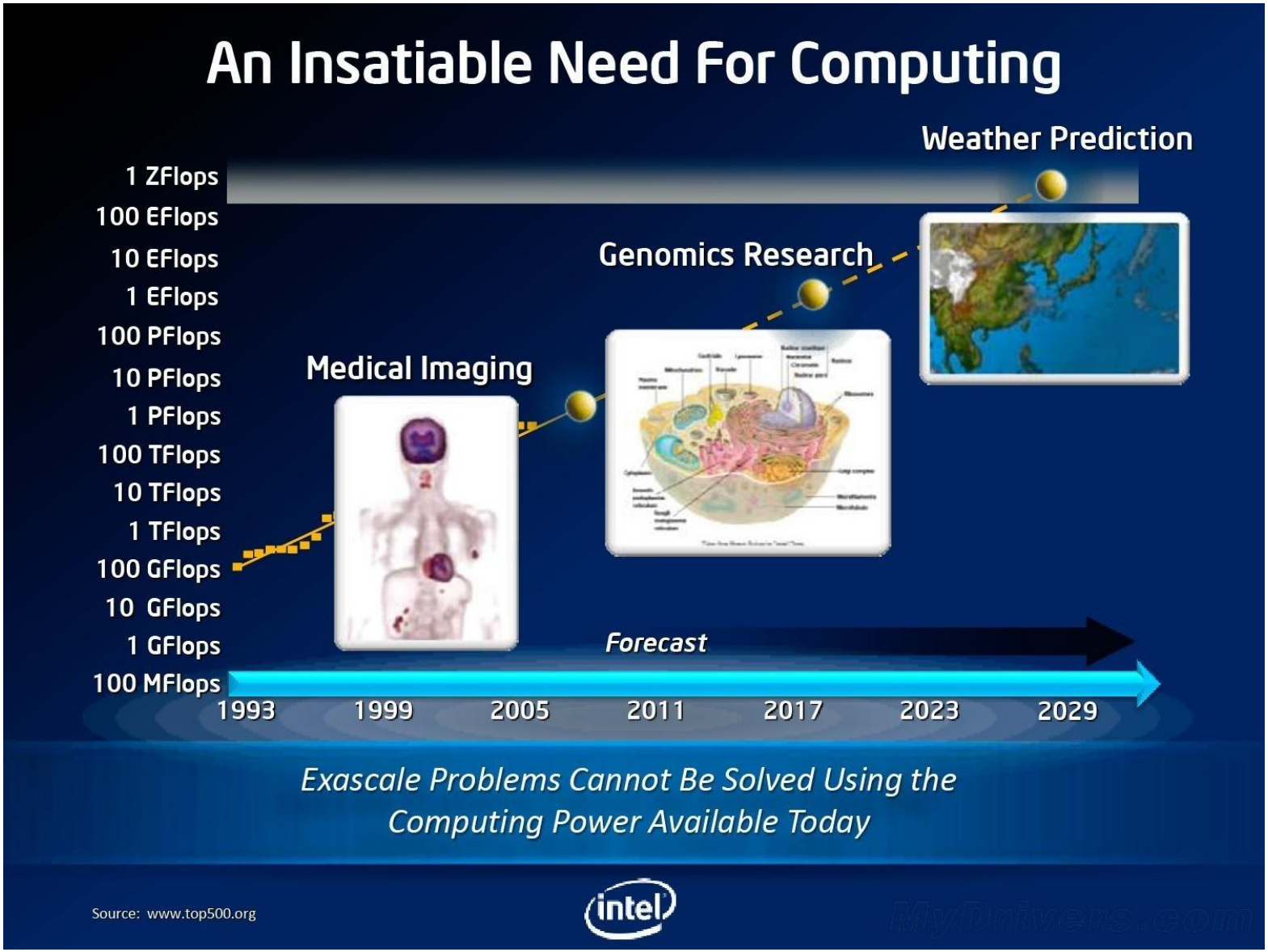
Floating-point Operations Per Second

每秒钟进行的浮点计算操作次数

Meta → Giga → Tera → Peta → Exa → Zetta

10^6 → 10^9 → 10^{12} → 10^{15} → 10^{18} → 10^{21}

科学计算的需要



科学计算的需要

Time-constrained scaling example

Real-time 3D graphics: more compute power allows for rendering of much more complex scene
Problem size metrics: number of polygons, texels sampled, shader length, etc.

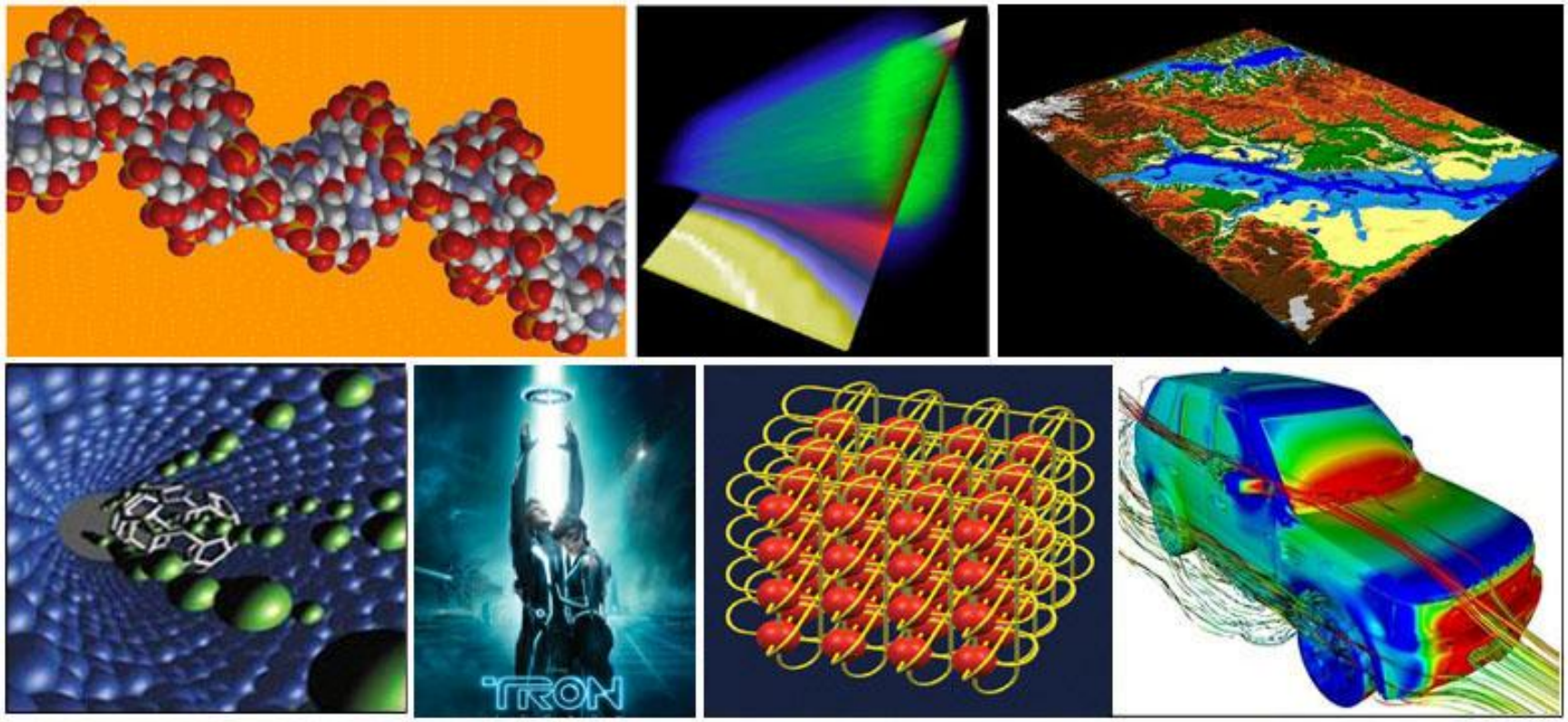
Half-Life (1998)

Assassin's Creed Unity (2014)

工程计算的需要

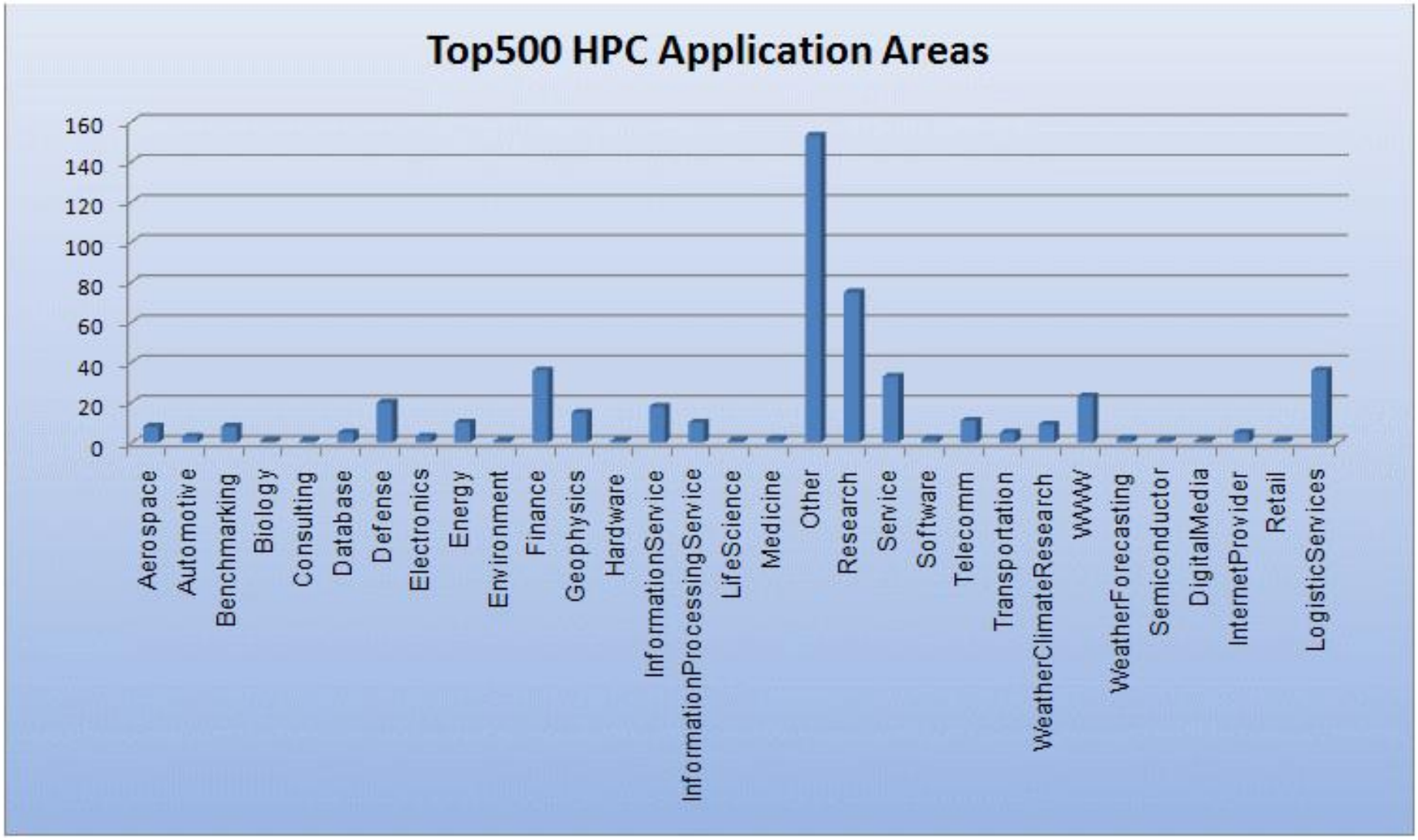
- **大型并行计算机是许多工业部门的支柱**
  - 石油行业 (油藏分析)
  - 汽车行业 (碰撞模拟, 拖拉模拟, 燃料的燃烧效率的计算)
  - 航空行业 (气流分析, 发动机效率, 结构机械)
  - 计算机辅助设计
  - 制药行业 (分子模型)
  - 视觉要求下的应用
    - 应用于上面的全部领域
    - 娱乐业
    - 建筑行业 (预排工作以及绘制立体视图)
  - 经济学建模 (投资收益及相关分析)
  - …

工程计算的需要



商业计算的需要

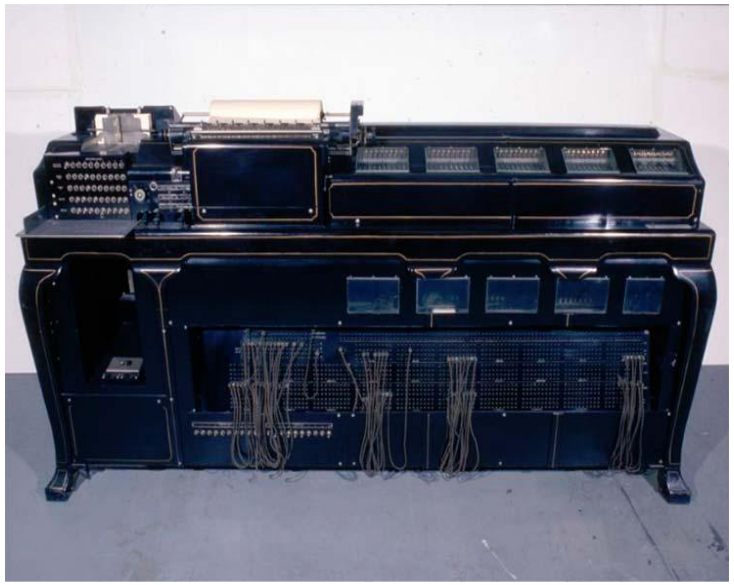
- **高质量要求依赖于并行计算机**
  - 计算能力决定了可以掌控的交易规模
- **数据库, 在线处理, 决策支持, 数据挖掘, 数据仓库**
- **TPC 基准 (TPC-C order entry, TPC-D 决策支持)**
  - 提供了明确的测量标准
  - 企业的规模决定着系统的规模
  - 随着处理器数目的增加，问题的规模也并不是不变的
  - 吞吐率作为性能标准 (transactions per minute, tpm)



科学、工程以及商业计算的需要促使了向并行计算的转变

超级计算

- “超级计算”（supercomputing）一词于1929 年在《纽约世界报》首次使用
  - 指的是 IBM 为哥伦比亚大学制造的大型定制制表机（tabulator）



http://www.columbia.edu/cu/computinghistory/packard.html

超级计算机

- 超级计算机（Supercomputer）指能够执行一般个人电脑无法处理的高速运算的计算机
  - 通常而言，超级计算机的处理能力是目前最快的笔记本电脑或台式机的**一百万倍**以上
  - 现有的超级计算机运算速度大都可以达到每秒一万亿次以上

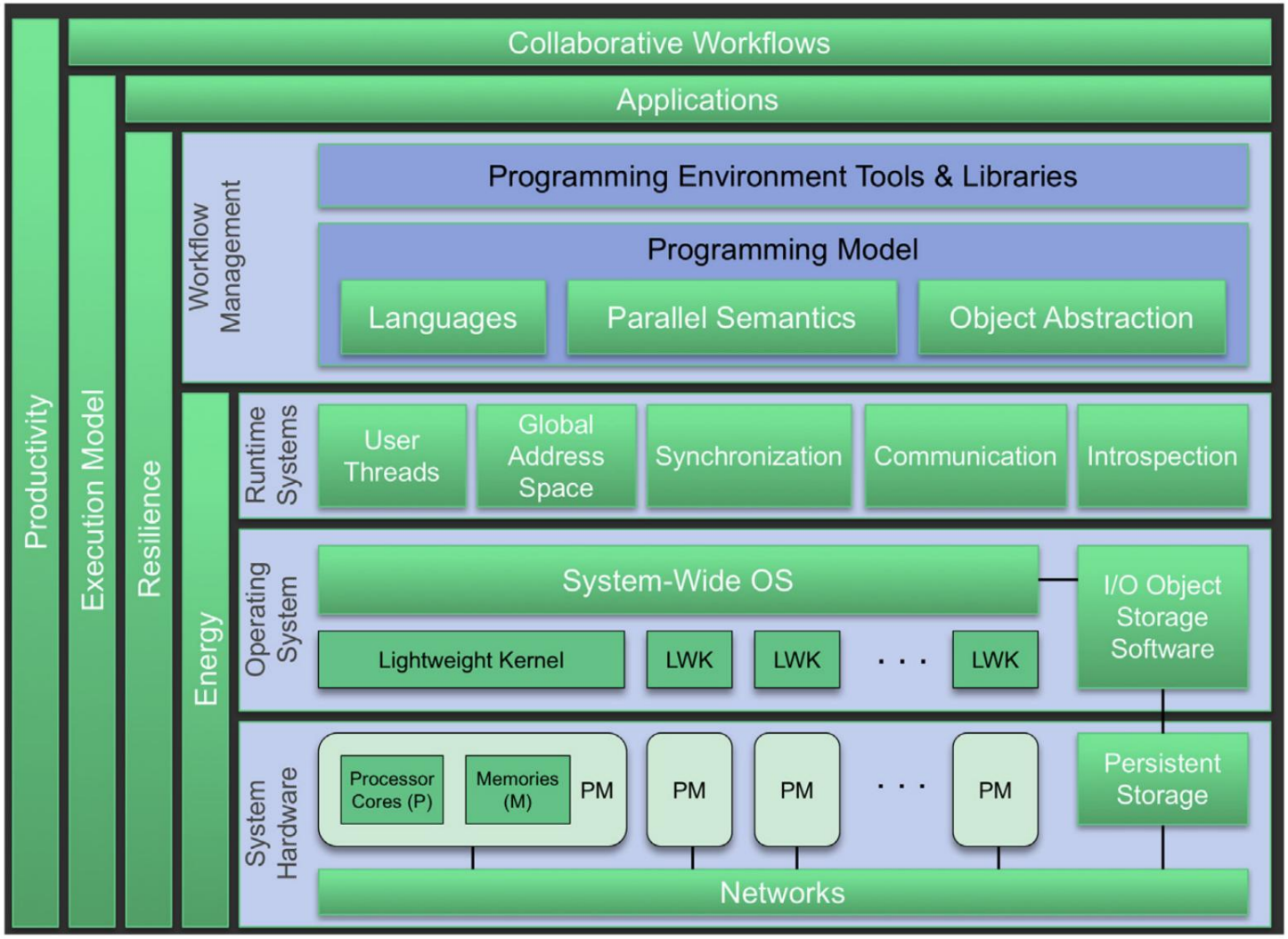
超级计算机的特征

某一时代性能最高的系统，服务于国家战略目标

- 运算速度超级快
- 存储容量超级大
- 占地面积超级大　　100-1000 m^2
- 能耗超级高　　~10 MW
- 造价超级贵　　~Billion RMB

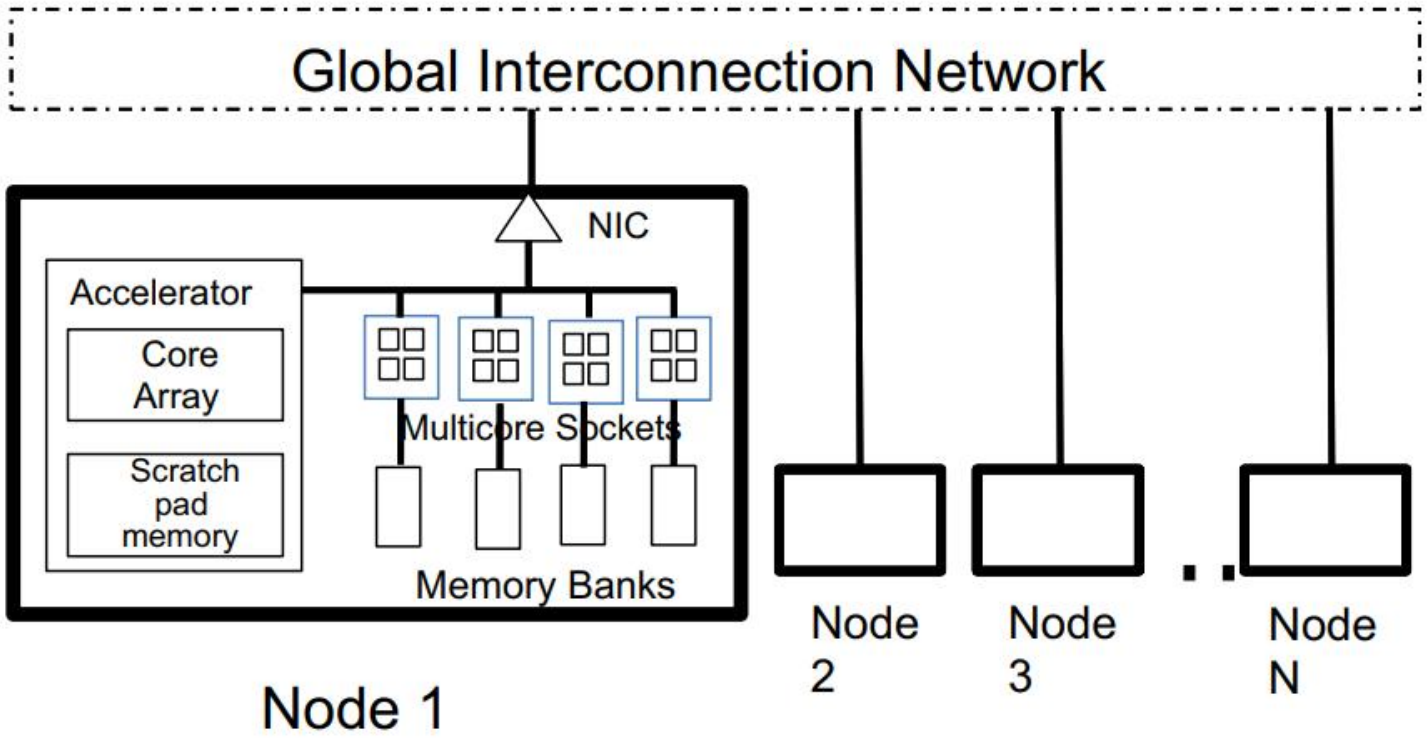
其关键技术迅速辐射到其它应用领域，推动国民经济建设、科学技术进步与人类社会发展

超级计算机剖析


Collaborative Workflows
Applications
Productivity
Execution Model
Resilience
Energy
Workflow Management
Programming Environment Tools & Libraries
Programming Model
Languages
Parallel Semantics
Object Abstraction
Runtime Systems
User Threads
Global Address Space
Synchronization
Communication
Introspection
Operating System
System-Wide OS
I/O Object Storage Software
Lightweight Kernel
LWK
LWK
LWK
System Hardware
Processor Cores (P)
Memories (M)
PM
PM
PM
PM
Persistent Storage
Networks

超级计算机 vs 传统计算机

- 区别在于组件资源的组织、互连和规模，以及支撑软件在该规模上管理系统运行的能力



高性能计算机的演进

- 高性能计算机的发展历史大致可以分成四个阶段：
  - “初生”时代
  - “克雷”时代
  - “多核”时代
  - “异构”时代

高性能计算机的演进

- **崭露头角：“初生”时代**
  - 世界上第一台真正意义上的高性能计算机诞生于 1964 年，是由美国计算机科学家西摩·克雷（Seymour Cray）和他的同事一起设计的
  - 由于是通过美国数据控制公司（Control Data Corporation，CDC）制造并推向市场，所以命名为 CDC 6600

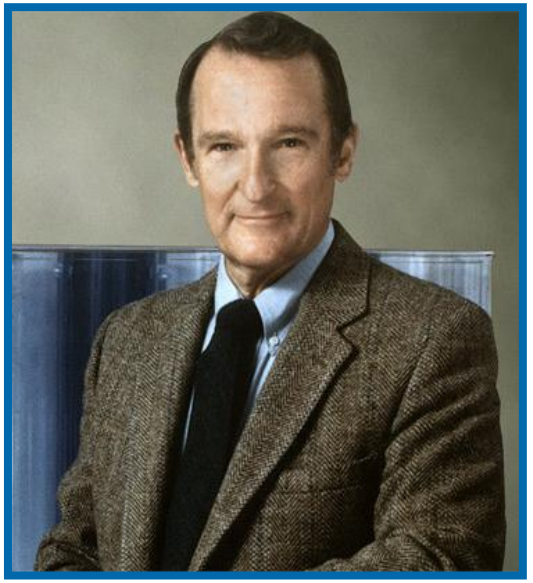
高性能计算机的演进

- **崭露头角：“初生”时代**

CDC 6600，1964年

- 约 3 MFLOPS
- 性能是当时最快的计算机IBM 7030的10倍
- 价格为800万美元

超算之父 – Seymour Cray



- 超算之父 1925~1996
- 创立了克雷公司，多次研制了世界上最快的超级计算机
- 美国国防部称他为“美国民族的智多星”

"Anyone can build a fast CPU. The trick is to build a fast system."

"Computers should obey a square law — when the price doubles, you should get at least four times as much speed.

高性能计算机的演进

- **崭露头角：“初生”时代**
  - CDC 6600 主要被用于高能核物理方面的研究工作，其 CPU 时钟**主频为 10 MHz**，峰值运算速度达到 3 MFLOPS，从 1964 年到 1969 年，一直保持世界排名第一的位置
  - CDC 6600 采用了一系列新技术，包括**液冷技术、硅晶体管技术、精简指令集技术**等
  - CDC 6600 的性能比同时期其他计算机性能高出约 10 倍，并重新定义了高性能计算机，因此我们将其作为高性能计算机的历史开端

高性能计算机的演进

- **独领风骚：“克雷”时代**
  - 作为超算之父的西摩·克雷，于 1972 年离开 CDC 公司，创立了克雷公司（Cray Research）
  - 1976 年，克雷公司发布了超级计算机 Cray-1，Cray-1 的推出在高性能计算的发展史上具有里程碑的意义
  - Cray-1的时钟频率达到 80 MHz，峰值性能为 136 MFLOPS，被部署到美国洛斯·阿拉莫斯国家实验室，用于**核武器的研究**
  - Cray-1 高性能计算机创造了**多个首次**
    - 首次使用集成电路对 CPU 芯片进行封装
    - 首次使用基于向量的处理器架构
    - 首次使用链式结构来大幅减少 CPU 和内存的数据交换频率

高性能计算机的演进

- 独领风骚：“克雷”时代

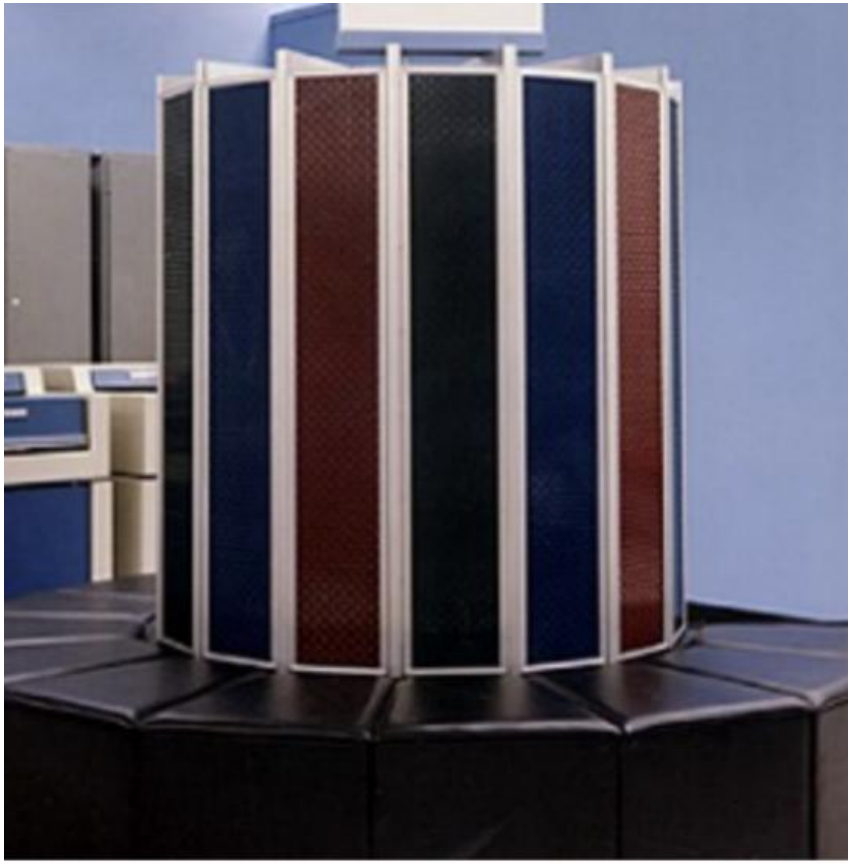


图 1-13　Cray-1 超级计算机

高性能计算机的演进

- **并行协作：“多核”时代**
  - 1980 年代中后期，自从 Cray-2 开创了高性能计算的一个全新发展方向—“**多核并行**”之后，大量拥有数千个处理器的高性能计算机如雨后春笋般快速出现
    - 日本电气株式会社公司于 1989 年发布了 SX-3 / 44R 高性能计算机，并以 23.2 GFLOPS 计算速度获得世界第一
    - 与此同时，另外一个传奇公司——英特尔也加入到高性能计算机领域的竞争当中
      - Paragon 系列超级计算机

高性能计算机的演进

- **并行协作：“多核”时代**
  - 英特尔公司在 1997 年推出的 ASCI Red 系列高性能计算机，首次突破了“T”级计算大关

高性能计算机的演进

- **百花齐放：“异构”时代**
  - 2002 年，日本 NEC 公司发布了 “地球模拟器（Earth Simulator）” 超级计算机，并借此重新夺回了 TOP500 的桂冠，该计算机拥有 35 TFLOPS 浮点运算性能
  - 但不久之后，该计算机就被美国 IBM 公司发布的 Blue Gene/L 系列超级计算机超越，Blue Gene 系列计算机也连续 4 年占据 TOP500 榜首

Top500排行榜 （2023年6月）

Rank	System	Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power (kW)
1	**Frontier** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE DOE/SC/Oak Ridge National Laboratory United States	8,699,904	1,194.00	1,679.82	22,703
7	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway, NRCPC National Supercomputing Center in Wuxi China	10,649,600	93.01	125.44	15,371
8	**Perlmutter** - HPE Cray EX235n, AMD EPYC 7763 64C 2.45GHz, NVIDIA A100 SXM4 40 GB, Slingshot-10, HPE DOE/SC/LBNL/NERSC United States	761,856	70.87	93.75	2,589
9	**Selene** - NVIDIA DGX A100, AMD EPYC 7742 64C 2.25GHz, NVIDIA A100, Mellanox HDR Infiniband, Nvidia NVIDIA Corporation United States	555,520	63.46	79.22	2,646
10	**Tianhe-2A** - TH-IVB-FEP Cluster, Intel Xeon E5-2692v2 12C 2.2GHz, TH Express-2, Matrix-2000, NUDT National Super Computer Center in Guangzhou China	4,981,760	61.44	100.68	18,482

美国产品重回榜首，竞争日益激烈

高性能计算机的演进

– 美国花费 6 亿美元研发的Frontier 超级计算机的设计计算性能为 1.5 EFLOPS，在 2022 年达到完全负载工作状态，实测计算性能为 1.102 EFLOPS

Oak Ridge National Laboratory (ORNL)

国产高性能计算机的发展历程

- 起步阶段（1956-1995 年）
  - 1958 年 8 月 1 日，中国科学院计算技术研究所，成功研制了中国第一台小型电子管通用数字计算机—**103 型计算机**，该机器运算速度为每秒 1800 次，主要用于科学计算
  - 1959 年，中国科学院计算技术研究所成功研制了我国第一台自行设计的大型电子管数字计算机—**104 型计算机**，该机器每秒运算次数为 1万次
  - 1964 年，我国成功研制了第一台国产大型电子管通用数字计算机—**119 型机**，该机器每秒运算 5 万次
  - 以 103 机、104 型机、119 型机为代表的我国第一代通用数字计算机均基于电子管技术

国产高性能计算机的发展历程

- 起步阶段（1956-1995 年）

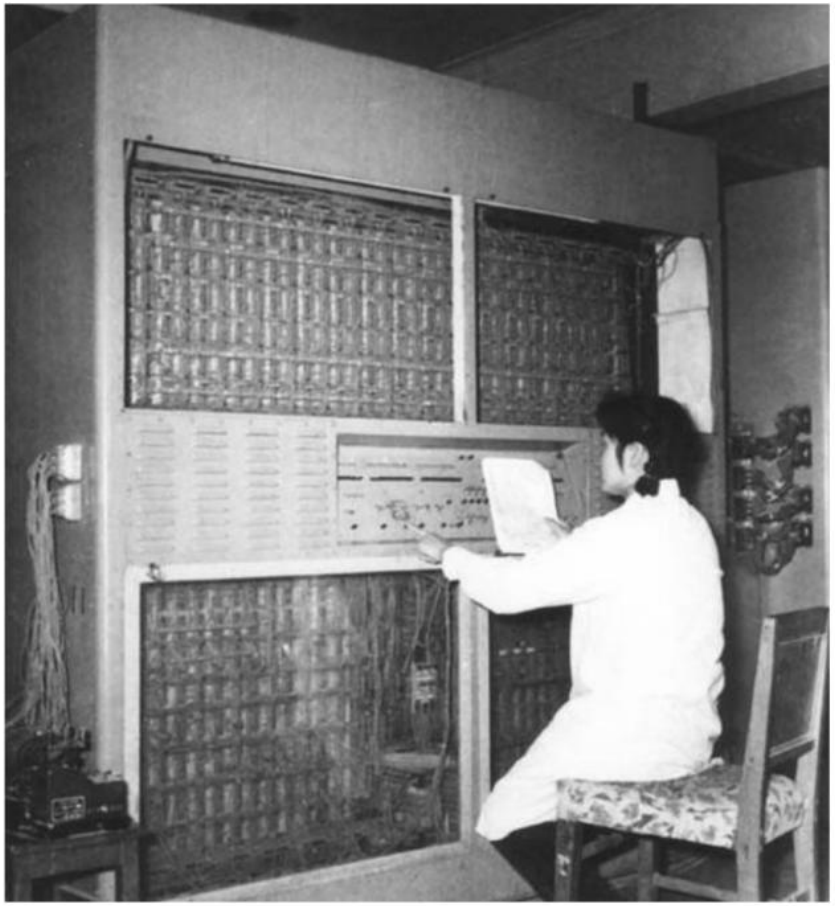


图 1-16　中国科学院计算技术研究所研制的 103 型计算机

国产高性能计算机的发展历程

- 起步阶段（1956-1995 年）
  - 为了服务“两弹一星”等国家重大战略项目，从上世纪五十年代开始，我国开始研制专门的高性能计算机
  - 1958 年 9 月 8 日，中国第一台电子管专用数字计算机 901 机研制成功，由中国人民解放军军事工程学院（哈军工，现国防科技大学）**慈云桂教授**带领团队研制
    - 以 901 机为代表的我国第一代专用数字计算机也是基于电子管技术

国产高性能计算机的发展历程

- 起步阶段（1956-1995 年）
  - 美苏技术封锁
    - 中国只能走自主设计、自主生产的发展路线
  - 1965 年 4 月，中国人民解放军军事工程学院（哈军工，现国防科技大学）成功研制出了中国第一台晶体管数字计算机（441-B 计算机）
    - 相对于美国第一台全晶体管计算机 RCA501 晚了6年

国产高性能计算机的发展历程

- 起步阶段（1956-1995 年）
  - 1983 年 12 月 22 日，国防科技大学慈云桂教授牵头的科研团队，成功研制了**“银河 I 号”巨型计算机**，运算速度达每秒 1 亿次
  - **“银河 I 号”的研制成功，标志着中国打破西方国家封锁，成为能够独立研发亿次高性能计算机的国家，我国真正地迈入了超算的行列！**

中国巨型计算机之父 -慈云桂

- 慈云桂(1917-1990)，中国科学院院士，中国计算机界老前辈中的举旗人
- 主持研制成功我国首台亿次级巨型计算机“银河-I”

银河颂

银河疑是九天来，妙算神机费剪裁。
跃马横刀多壮士，披星戴月育雄才。
精雕岂为人称誉，细刻缘求玉琢材。
极目远穷千里外，琼楼更上不徘徊。

国产高性能计算机的发展历程

- 起步阶段（1956-1995 年）
  - 上世纪80年代，在“造不如买、买不如租” “市场换技术”等政策指导下，半导体产业发展受到严重影响
  - 到80年代末，中国凭自己的技术已经很难生产出一台计算机了
  - 在中国丧失独立自主研发制造计算机的能力后，美国政府开始严格限制对中国出口高性能计算机

国产高性能计算机的发展历程

- 起步阶段（1956-1995 年）
  - “玻璃房子”
    - 原石油工业部地球物理勘探局曾花费巨资从国外购买了一台大型机，没想到，对方对后期的集成和维护费用开价百亿以上
    - 不仅如此，对方还提出了一个让中国IT人铭记了十几年的屈辱条件——在设备使用过程中，为防止机器核心技术外泄，设备机房采用全透明的玻璃墙壁，对方要时刻监控我国工作人员的一举一动，甚至连高性能计算机的启动密码和机房钥匙都要由对方控制
  - 为了彻底拆除“玻璃房子”，研发出完全拥有自主知识产权的国产高性能计算机，我国的科学家们走上了一条异常艰苦的突围之路

国产高性能计算机的发展历程

- 追赶阶段（1996-2009 年)
  - 进入二十一世纪，我国的高性能计算产业也随着世界高性能计算技术的突破而快速发展
  - 以“**天河**”、“**曙光**”、“**神威**”系列为代表的国产超级计算机不断迭代更新
  - 于 2004 年、2008 年和 2009 年分别突破十万亿次、百万亿次和千万亿次计算大关，不断缩小与世界领先技术的差距

国产高性能计算机的发展历程

- 追赶阶段（1996-2009 年）
  - 2004 年，由中科院计算所、曙光公司、上海超级计算中心三方共同研发制造的**“曙光 4000A”**高性能计算机实现了 10 TFLOPS 的运算速度，成为我国第一个跨入十万亿级计算的超级计算机，也标志着我国成为继美国和日本第三个有独立设计十万亿级高性能计算的国家
  - “曙光 4000A”计算机也是我国第一个进入TOP500前十的超级计算机

国产高性能计算机的发展历程

- 追赶阶段（1996-2009 年）
  - 2009 年 9 月，国防科技大学成功研制了我国首台千万亿次超级计算机—“天河一号”，部署在国家超级计算天津中心
  - “天河一号”峰值速度为 1.2 PFLOPS 且 LINPACK 实测性能为 563.1 TFLOPS，使中国成为世界上第二个（继美国之后）能够研制千万亿次超级计算机的国家
  - 至此，国产高性能计算机离登顶只有一步之遥

国产高性能计算机的发展历程

- 超越阶段（2010 年至今）
  - 2010 年 11 月 16 日是我国高性能计算机发展史中一个值得铭记的时刻，全球超级计算机 TOP500 排行榜在美国新奥尔良市揭晓，升级后的“天河一号”二期系统（“天河-1A”）以每秒 4700 万亿次的峰值性能、每秒 2507 万亿次的 LINPACK 实测值持续性能，超越美国橡树岭国家实验室的“美洲虎”超级计算机，成为当时世界上最快的超级计算机
  - 这是我国自主研发的高性能计算机首次登顶超级计算机 TOP500 排行榜，**这标志着我国国产高性能计算机正式进入世界一流梯队**

国产高性能计算机的发展历程

- 超越阶段（2010 年至今）
  - 2013 年底，国防科技大学研制的“天河二号”超级计算机正式验收并部署到国家超算广州中心，自2013 年 6 月至 2016 年 6 月，“天河二号”连续 6 次排名世界上最快的超级计算机，理论峰值达 54.90 PFLOPS，实际峰值速度达 33.86 PFLOPS

天河二号超级计算机

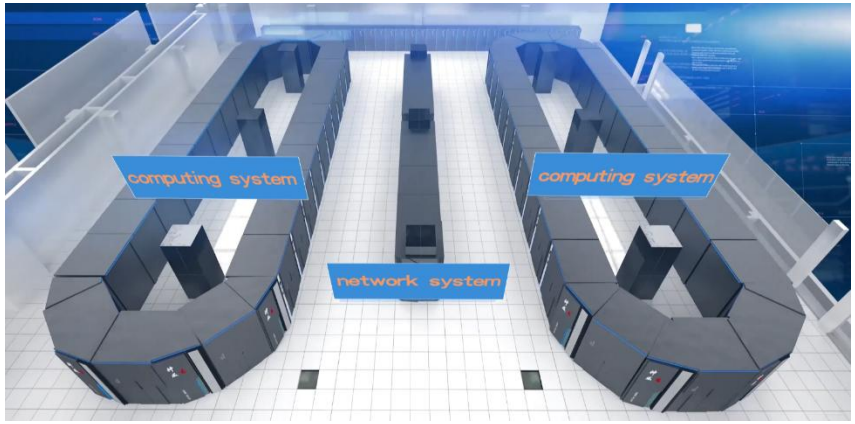
“天河二号”峰值计算性能为 100,678.7TFLOPS，持续计算性能为 61,444.5TFLOPS

国产高性能计算机的发展历程

- 超越阶段（2010 年至今）
  - 2016 年 6 月 20 日，“神威·太湖之光”超级计算机在 LINPACK 计算性能测试中以 93 PFLOPS 的评测结果超越“天河二号”，成为当时世界上最快的超级计算机
  - “神威·太湖之光”也是**中国首度自行设计并使用国产核心芯片登上 TOP500 第一名**的超级计算机
  - “神威·太湖之光”一直保持世界领先的计算性能，直到 2018 年 6 月被美国的 Summit 超级计算机所超越

“神威•太湖之光”超级计算机

- **神威太湖之光**
  - 采用**国产申威众核处理器**
  - 全系统4万个结点，1000万核
  - 峰值性能12.5亿亿次 / 秒，Linpack 9.3亿亿次 / 秒
  - 能效比 6GF/W
  - 装备国家超算无锡中心



- **世界超级计算机Top500排行榜 “三连冠”**

  2016.6~2018.6

中国超算发展的重要经验

- **国家科技计划与地方、应用部门的发展计划相结合**
  - 多渠道筹资研制高效能计算机
  - 科技部和地方政府共同支持建立国家超算中心
- **产学研用的结合**
  - 高性能计算中心在高效能计算机研制中发挥作用
    - 遴选研制团队 / 提出系统指标
  - 企业参与国家科技计划，提高自身水平，促进研制工作
    - 浪潮、曙光、联想参与千万亿次和亿亿次计算机的研制
  - 应用部门牵头发展应用软件
- **机器、环境、应用三者均衡发展，相互促进**

超级计算机的发展

Cray-1
1976 160Mflops

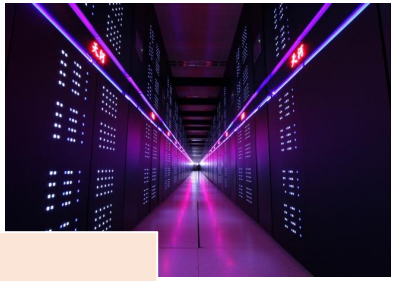
Intel ASCI Red
1997年，1万亿次

IBM BlueGene/L
2004年，70Tflops
2007年，596Tflops

中国 NUDT TH-1A
2010. 4.7Pflops

美国 Cray Titan
2012. 17.6Pflops

Cray-Y
1988年



DT 天河2号
2013. 33.86Pflops

40年系统性能增长5.8亿倍

CrayT3D
1993, 19Gflops

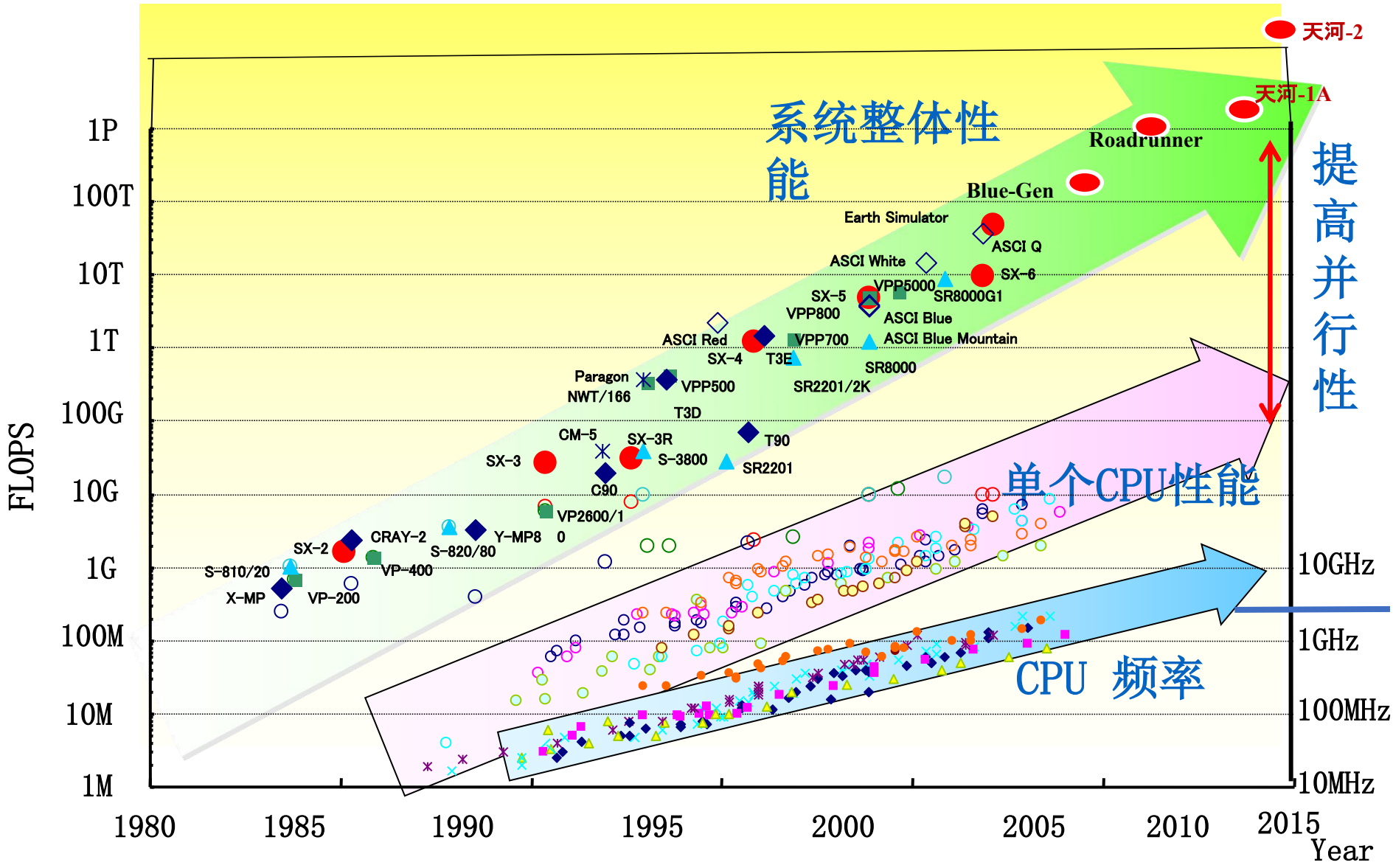
日本地球模拟器
2002年，40Tflops

Cray Jaguar
2009,1.76Pflops

美国 IBM Sequoia
2012. 16.32Pflops

中国 太湖之光
2016. 93Pflops

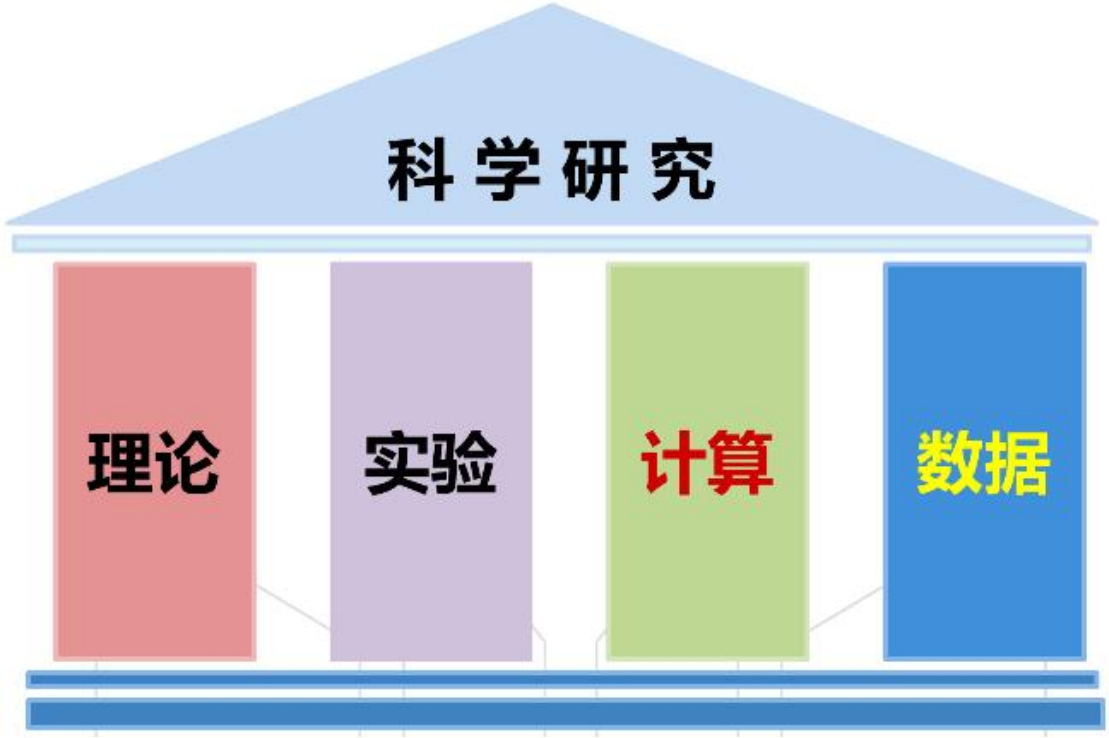
“并行”是通往超级计算的唯一途径



超级计算机能干什么?

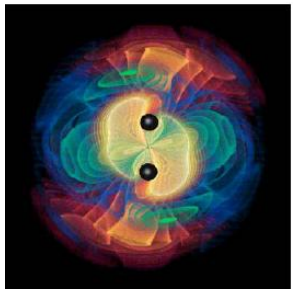
超级计算机能干什么?

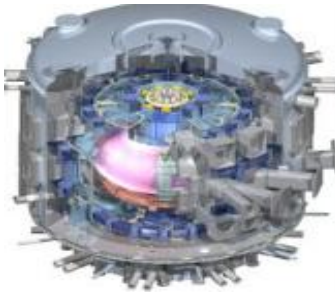
- 超级计算机对科学发现、技术创新、产业革命的重要作用
  - 高性能计算：是科学研究的三大手段之一
  - 大数据处理：正成为科学研究的第四范式

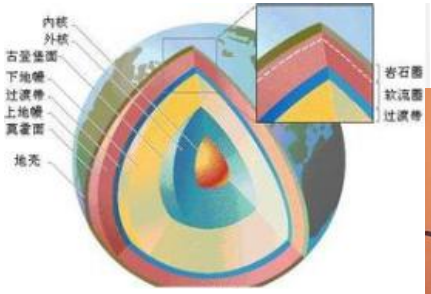


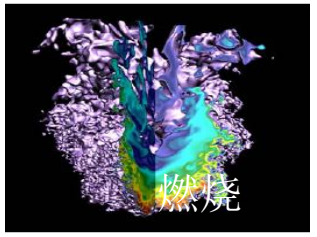
超级计算机能干什么?

- **超级计算可以帮助人们解决一系列重要问题**
  - 尺度超大 Too big
  - 尺度超小 Too small
  - 时变超快 Too fast
  - 时变超慢 Too slow
  - 过程超昂贵 Too expensive
  - 过程超危险 Too dangerous



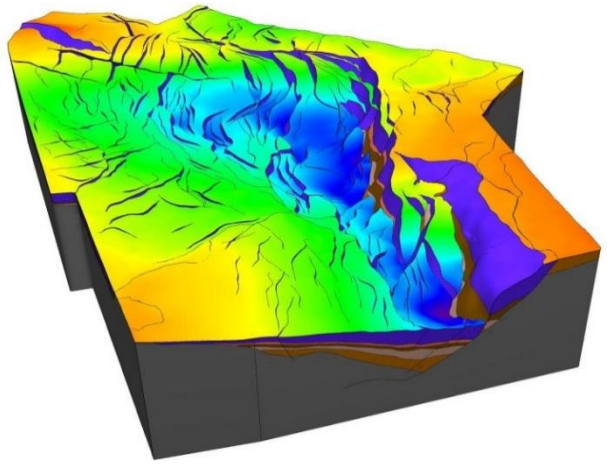


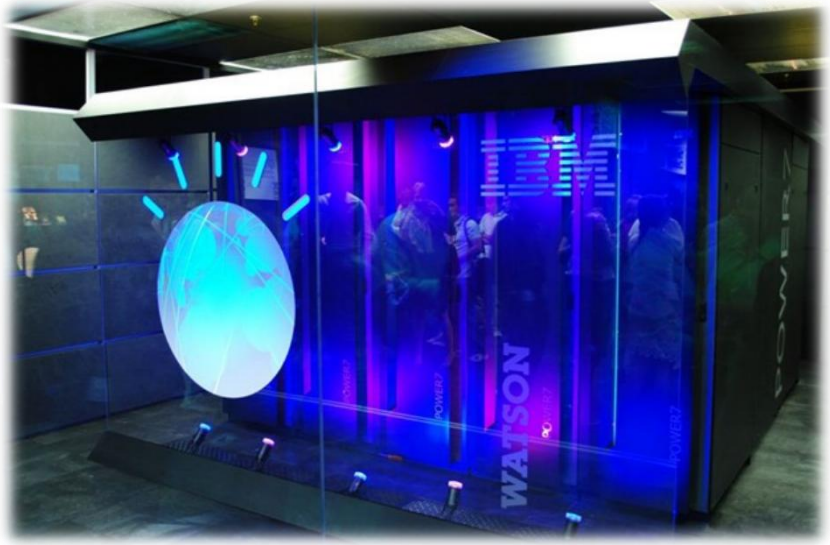




微观　宏观　极端条件

算天、算地、算人、算命





代表性超算问题

领域	问题
金融服务	欺诈与异常检测、回溯测试、量化交易、风险分析
石油天然气	地震处理、油田建模
制造业	材质模拟、结构仿真、空气动力学仿真、热力学仿真
生命科学	分子动力学、药物筛选、基因组测序
地球科学	大气层建模、气象模拟预测、冰川建模

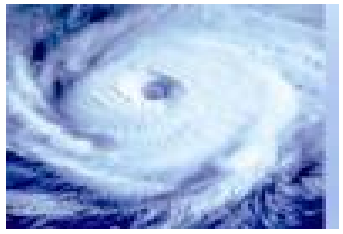
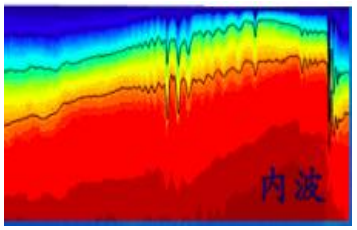
超级计算机的应用领域

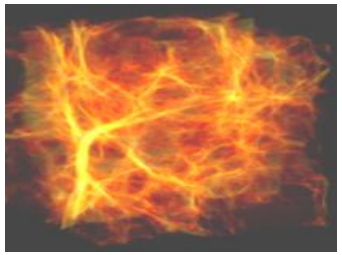
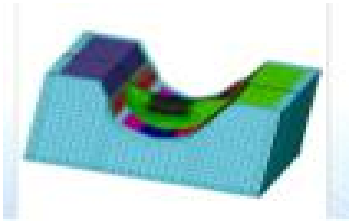
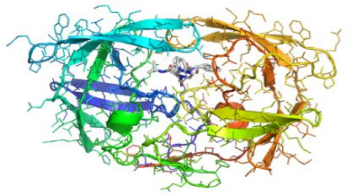
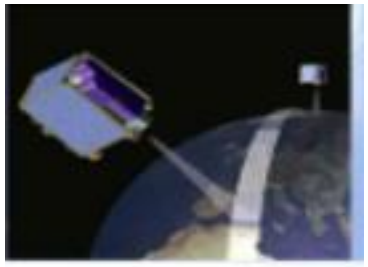
- 气候环境
- 天文物理
- 新型能源
- 材料化工
- 航空航天
- 工程设计
- 石油勘探
- 生命科学

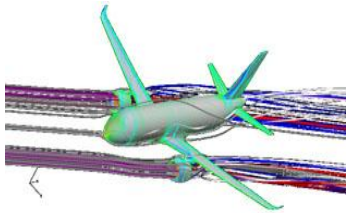
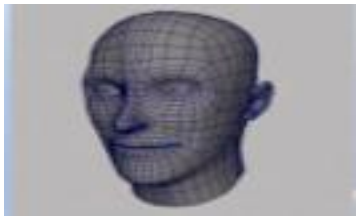
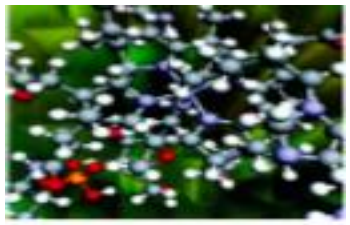
……

- 智慧城市
- 金融分析
- 互/物联网
- 智能交通
- 动漫渲染
- 人工智能
- 测绘分析
- 精准医疗

……

高性能计算机的未来发展趋势

- 从 E 级计算到 Z 级计算
  - 简单地堆叠硬件难以继续实现性能指标提升，我们还需要解决能耗、可扩展性、可靠性、应用编程、应用效率等诸多极具挑战性的难题
- 超算与人工智能、大数据的融合应用
  - 在未来的发展中，人工智能还需要更复杂的硬件结构才能完成。智能超级计算机的研发也获得了业界与科学家们的广泛关注
- 非冯·诺依曼架构的新型计算系统
  - 由于硅基半导体工艺已逐步逼近其物理极限，经典的冯·诺依曼架构暴露出了愈发明显的瓶颈

小结

- 课程简介
- 量化设计与分析基础
- 并行计算与并行计算机
- 高性能计算机发展史