



中国科学技术大学
University of Science and Technology of China

计算系统概论A
Introduction to Computing Systems
(CS1002A.03)

Introduction to Computing Systems

计算系统概论A

CS1002A.03

陈俊仕

2026 Fall

计算机科学与技术学院
School of Computer Science and Technology



中国科学技术大学
University of Science and Technology of China

计算系统概论A
Introduction to Computing Systems
(CS1002A.03)

Chapter 1 Course Introduction

计算机科学与技术学院
School of Computer Science and Technology

0 Course Crew

1 What' s the difference between “Big” and “Small” ?

2 Why Take This Course?

3 Great Ideas in Computing Systems

4 Course Implementation

0 **Course Crew**

1 **What' s the difference between “Big” and “Small” ?**

2 **Why Take This Course?**

3 **Great Ideas in Computing Systems**

4 **Course Implementation**



- **My primary specialty is High Performance Computing**
 - High Performance Computing (on Supercomputers)
 - Parallel Algorithm Design (Science and Engineering Softwares)
 - Parallel Programming Model and Runtime System
- **I am the member of the Advanced Computer System Architecture (ACSA) Lab**
 - To tackle the challenge of improving computational efficiency
 - To investigate advanced architectures for energy-efficient HPC + AI + Big Data convergence applications
 - To make parallel programming accessible to most programmers
- **I am the mentor of the SwanGeese team of USTC**
 - Funded by Prof. Hong An
 - Participated in 60+ student SC competitions with 80+ honors and awards.
- **My goal as a lecturer**
 - To make your experience in ICS as enjoyable & informative as possible!
 - Humor, enthusiasm & technology-in-the-news in lecture
 - Fun and challenging projects
 - Projects & exams will be as rigorous as every year

Course Crew of CS1002A.03



	Name	Email	Phone
Instructor	CHEN Jun-shi (陈俊仕)	cjuns@ustc.edu.cn	13721074819
TA	TAO Xi-wei (陶熙伟)	xwtao@mail.ustc.edu.cn	17730209091
TA	CHEN Yu-qiao (陈雨桥)	chenyq665@mail.ustc.edu.cn	15295736658
TA	YANG Bo-wen (杨博文)	ustc2023ybw@mail.ustc.edu.cn	18135731102
TA	ZHU Dong-sheng (朱东胜)	zy937@mail.ustc.edu.cn	19011344174

Office hours and Discussion

- Chaired by 4 TAs
- Face to face help
- See web page for times

Web of course: <http://acsa.ustc.edu.cn/ics/>

Office: 西区科技实验楼西楼14层



Outline



0 Course Crew

1 What' s the difference between “Big” and “Small” ?

2 Why Take This Course?

3 Great Ideas in Computing Systems

4 Course Implementation

Questions



- How powerful are today's computers? Why are they so powerful?

What can computers do?

- Is an abacus a computer? How to understand Turing's contribution to computers?

How are they done?

- What are the serious flaws in today's computers?

What can't computers do?

Today, Computer is in Everything !



Games



Pad



Refrigerators



Robots



Laptops



Routers



Smart Phones



Media Players



Set-top boxes



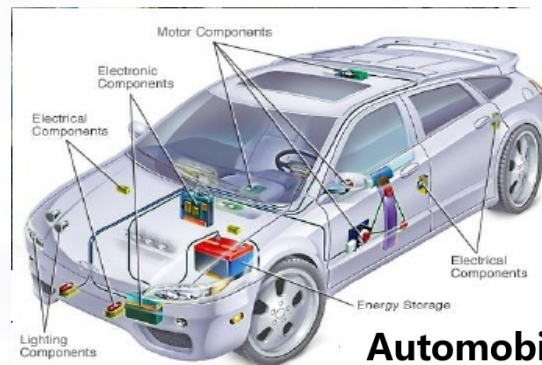
Cameras



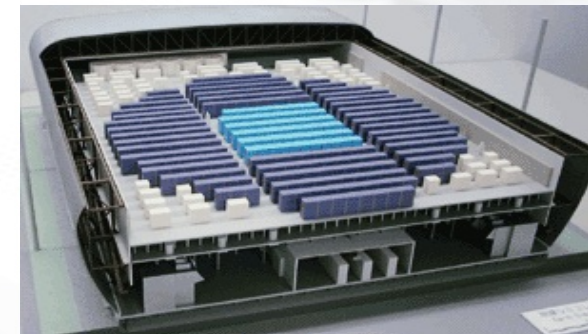
Servers



Sensor Nets



Automobiles



Supercomputers

Today, Computer is in Everything !



Vast infrastructure behind them: from the small to the big



Questions



- How powerful are today's computers? Why are they so powerful?

What can computers do?

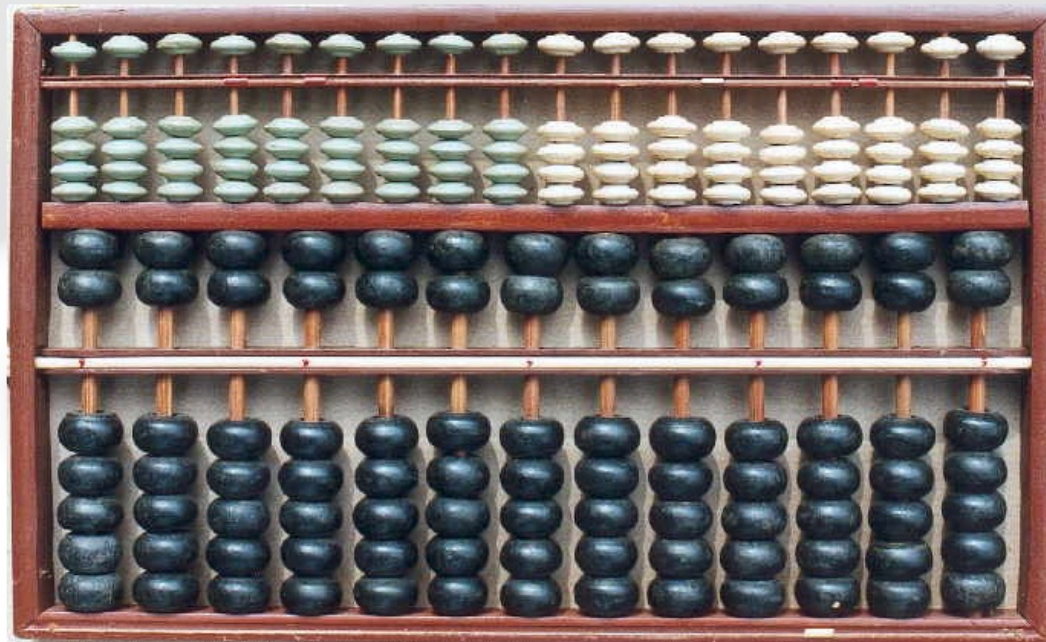
- Is an abacus a computer? How to understand Turing's contribution to computers?

How are they done?

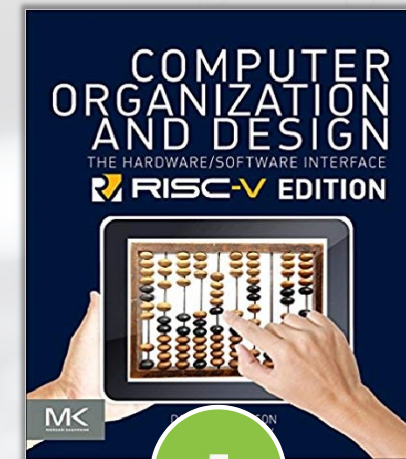
- What are the serious flaws in today's computers?

What can't computers do?

Is an abacus a computer?



Is an abacus a computer?



5

Computer Organization and Design: The Hardware/Software Interface,

David A Patterson,
John L. Hennessy,
5th edition. Morgan
Kaufmann
Publishers, Inc., 2017

Questions



- How powerful are today's computers? Why are they so powerful?

What can computers do?

- Is an abacus a computer? How to understand Turing's contribution to computers?

How are they done?

- What are the serious flaws in today's computers?

What can't computers do?

The Absolute Limits of Computing



■ The theoretical hard limit

- Computers cannot solve undecidable problems – problems for which no algorithm can ever guarantee a correct answer in finite time.

■ The Halting Problem (Turing, 1936)

- Can we write a program that checks whether any given program will halt or loop forever?
- Answer: NO. Proven mathematically impossible.

What Are the Serious Flaws in Today's Computers?



■ Four physical Walls

- **Power Wall:** chips overheat, frequency stalled
- **Memory Wall:** CPU starves waiting for data
- **Communication Wall:** Moving data costs more than computing
- **Manufacturing Wall:** Transistors hit atomic limits

■ Three system paradoxes

● Security Paradox

—To go fast, CPUs speculate (Spectre/Meltdown). Patching those holes forces cache flushes, slowing down performance

● AI Black-Box Paradox

—Deep learning works, but it's a black box. When it fails, we cannot debug it like code

● Software Bloat Paradox

—Wirth's Law: software slows down faster than hardware speeds up

What is Computer?

"Big " vs. "Small" Computer

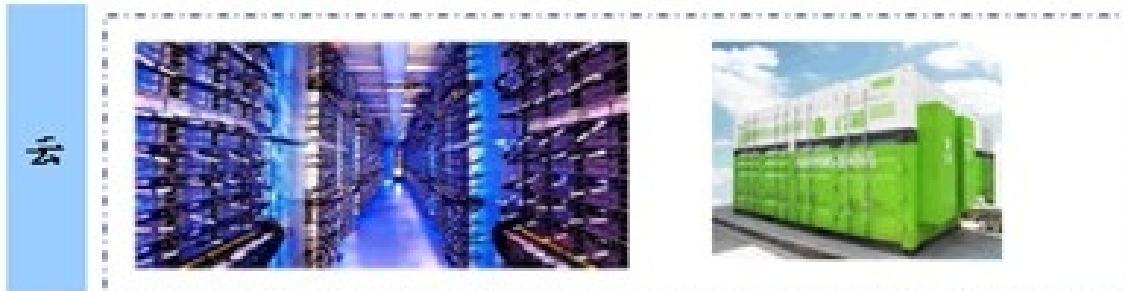


Personal mobile devices/smart terminal devices

Small

- High performance calculation
- High-performance communications and I/O
- Power consumption constraints
- The volume constraint

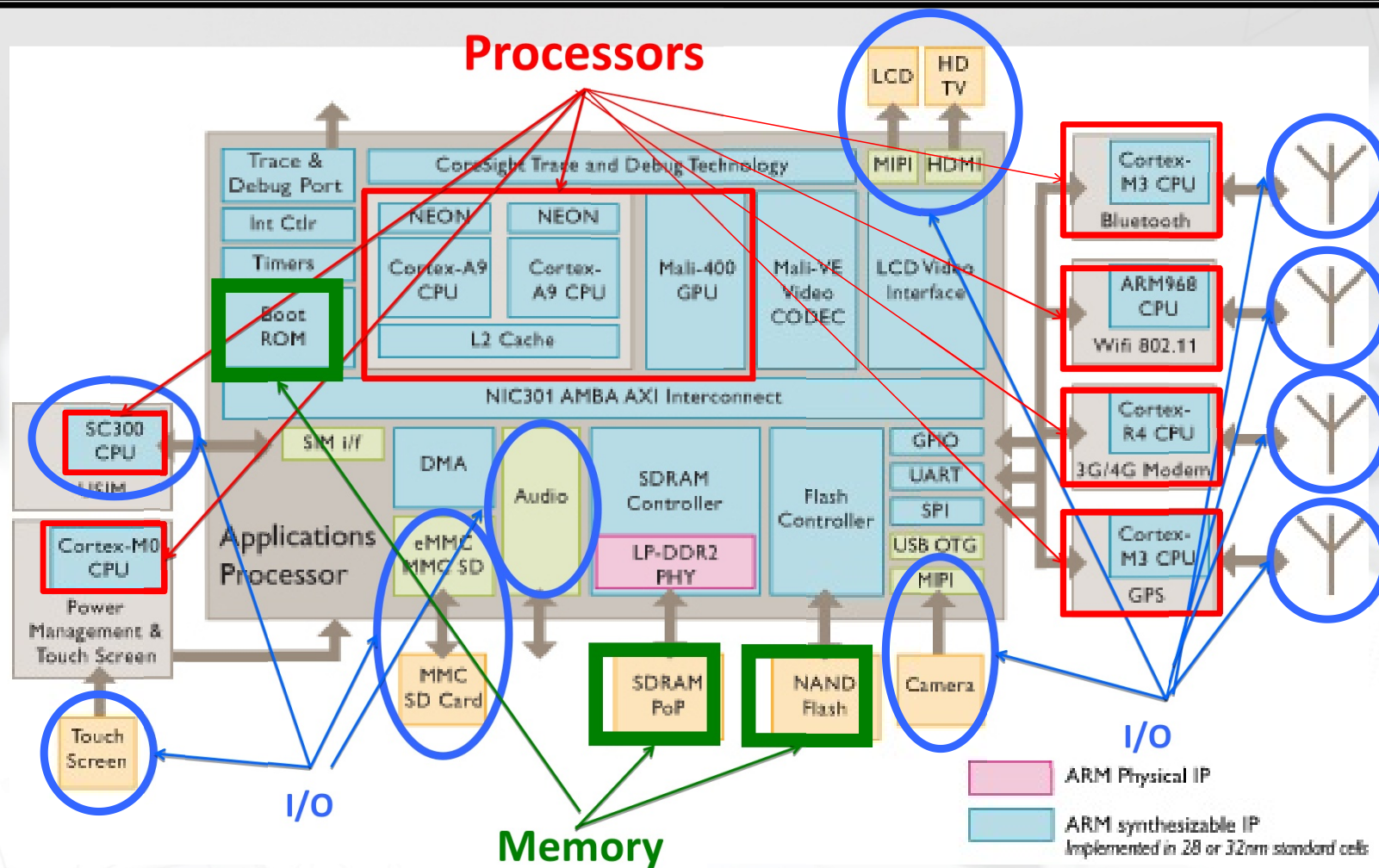
Supercomputing Center (computation) / Data Center (storage)



Big

- Capable of high-performance data processing
- More than demand
- Reliability requirement

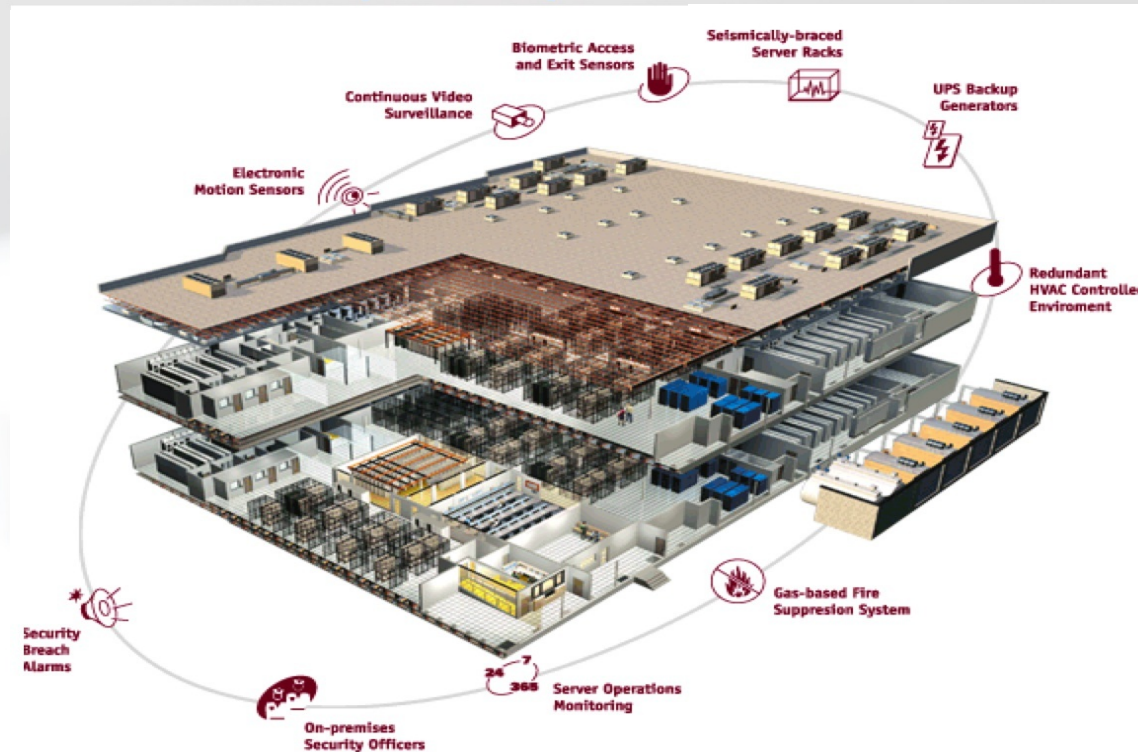
“Small” Computer Inside



"Big " Computer Inside



■ Data center: a "warehouse" supercomputer



Each data center covers an average of about 45,000 square meters and cost about \$600 million to build

The data center has 10,000 square meters



■ 占地超过1万平方米的数据中心



俄勒冈州康瑟尔布拉夫斯数据中心



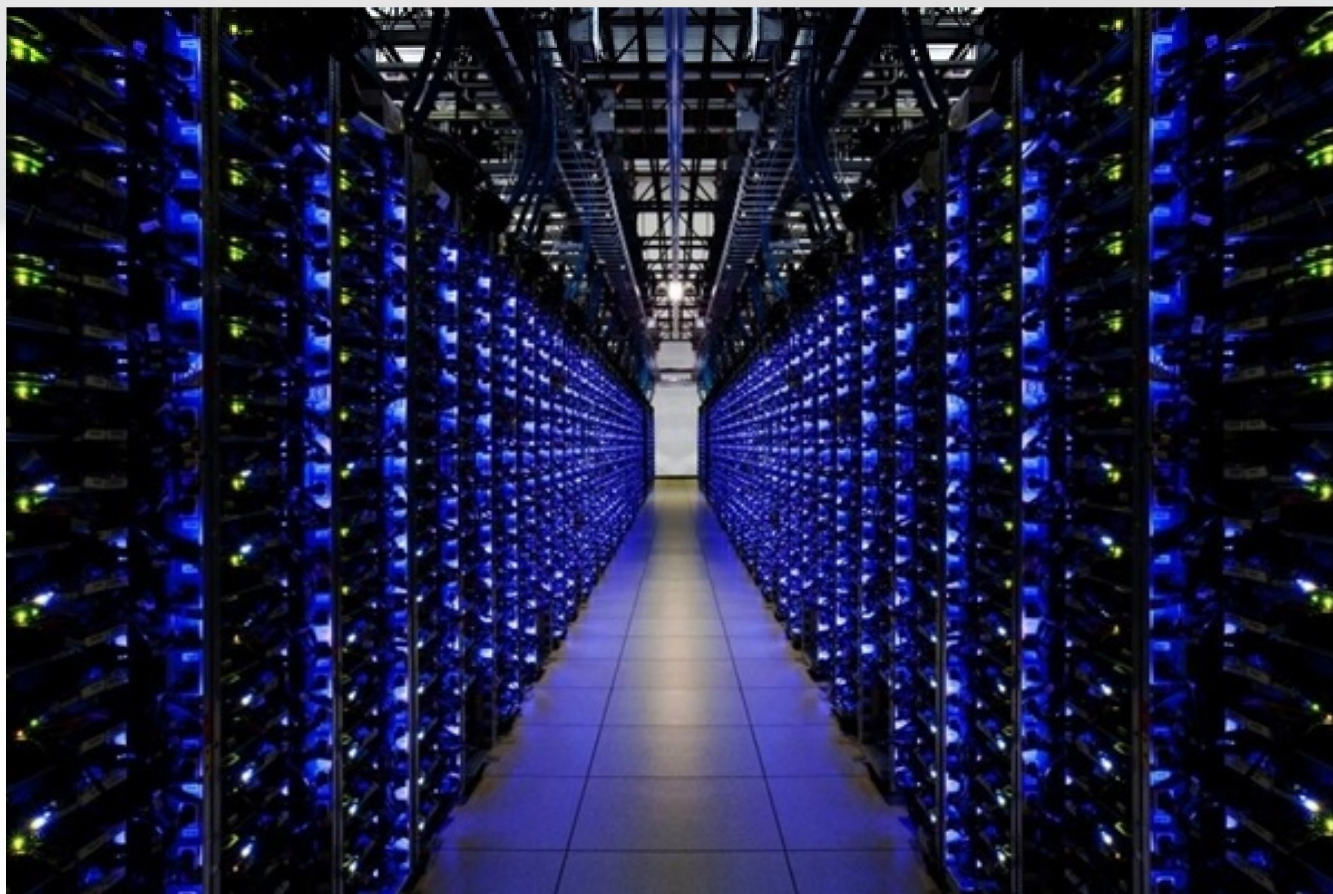
- 悬挂在俄勒冈州康瑟尔布拉夫斯数据中心上方的设备架, google数据中心的规模开始成形。巨大的钢梁不仅支撑整个结构还负责分配电力。



LEDs indicator on the server to monitor the normal operation of the system



- 蓝色LED灯显示服务器运转良好。使用LED因为它节能、长寿。



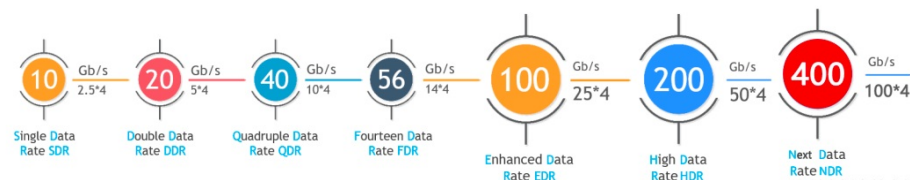
High-speed routers and switches built into the data center



- google园区内部的网络机房, 路由器和交换机让各个数据中心可以相互沟通。连接各中心的光纤网络速度比普通家庭的网速快200,000倍。这些光纤电缆沿天花板上的黄色电缆线架安装。



InfiniBand有源光缆 (Mellanox)

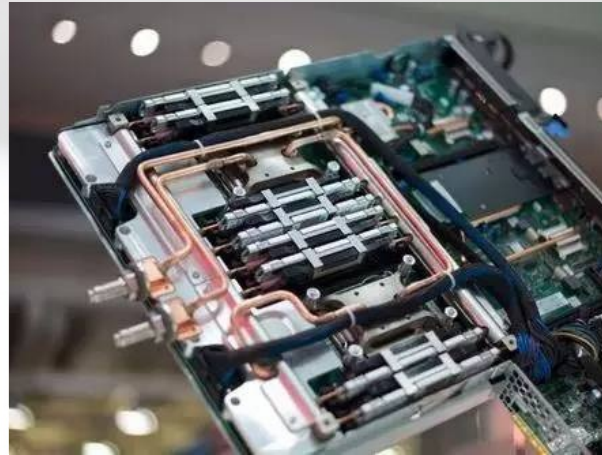


InfiniBand的网络带宽发展路线

A large air duct used to dissipate heat in a data center



风冷

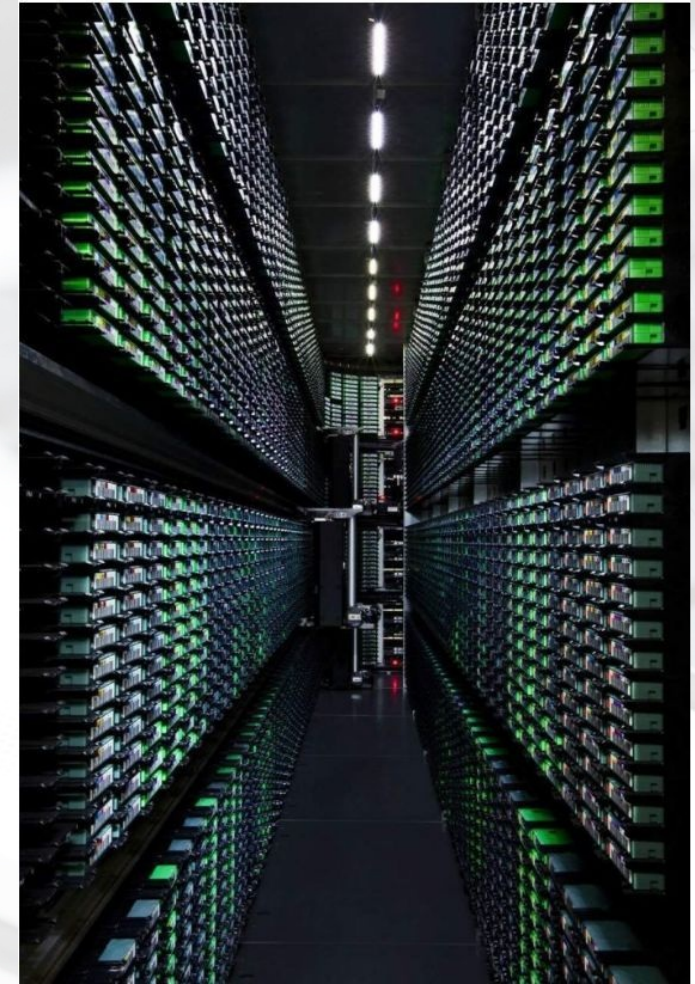
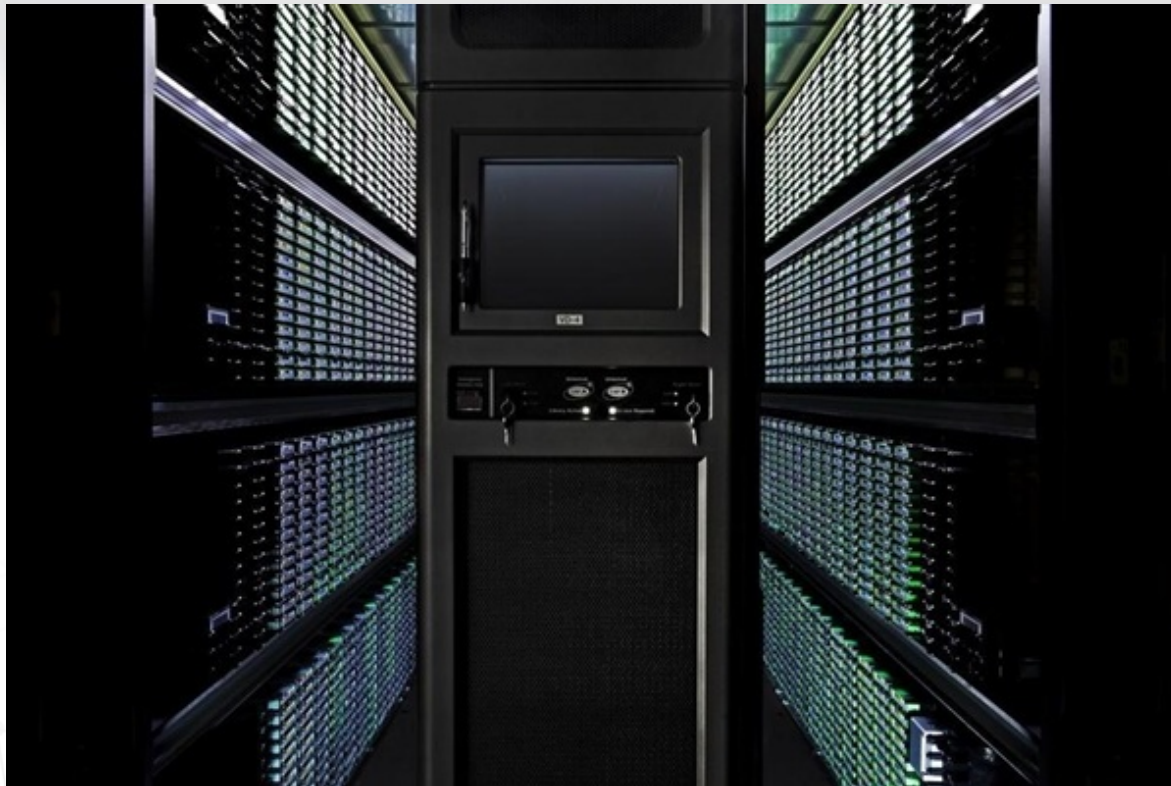


水冷



油冷

A robotic arm designed to help easily replacing the damaged hard disks



A piping system for dissipating heat in a data center

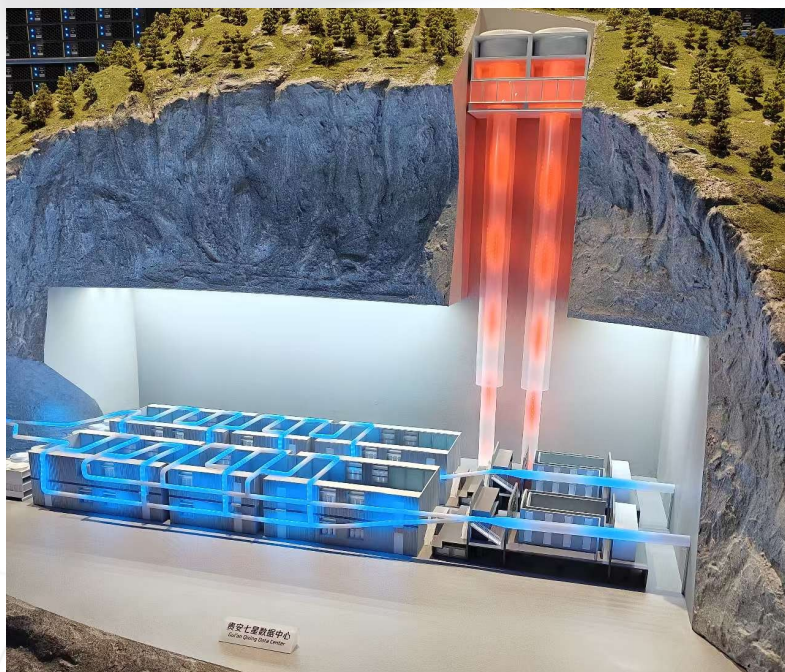


Gui'an Qixing Data Center (腾讯·贵安七星数据中心)



■世界首座洞库式数据中心（建在大山肚子里），国家级“数据保险箱”

- 天然“空调”：贵州地下百米深处岩温常年稳定在15-18摄氏度，为服务器提供了天然的冷却环境
- 成本节约：仅电费一项每年可节省约4000万元
- 长效设计：设计寿命 100年（常规仅50年）
- 荣获大奖：2024年，荣获中国土木工程最高奖——詹天佑奖



The TOP500 List (2016, since 1963)



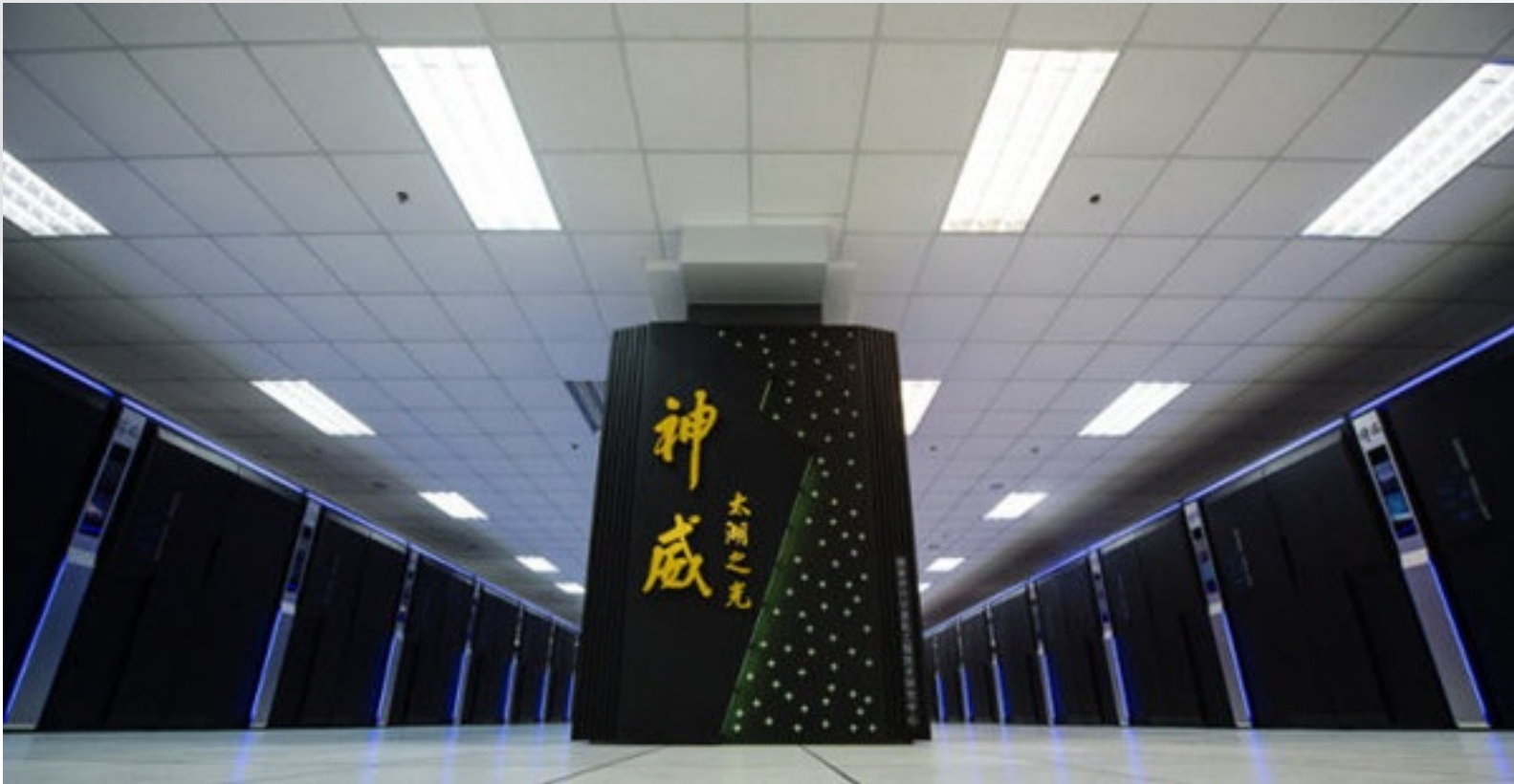
Top 10 supercomputers in the TOP500 List

Rank	System		Cores	Rmax (PFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	Sunway TaihuLight	CN, SW26010	10,649,600	93,014	125,435	15,371
2	Tianhe-2A	CN, Intel Xeon Phi 31S1P	3,120,000	33,862	54,902	17,808
3	Titan	US, Opteron 6274, NV K20x	560,640	17,590	27,112	8,209
4	Sequoia	US, BlueGene/Q, Power BQC 16C	1,572,864	17,173	20,132	7,890
5	Cori	US, Intel Xeon Phi 7250 68C	622,336	14,014	27,880	3,939
6	Oakforest-PACS	Japan, Intel Xeon Phi 7250 68C	556,104	13,554	24,913	2,719
7	K computer	Japan, SPARC64 VIIIfx	705,024	10,510	11,280	12,660
8	Piz Daint	Switzerland, Intel Xeon, NV P100	206,720	9,779	15,987	1,312
9	Mira	US, BlueGene/Q, Power BQC 16C	786,432	8,586	10,066	3,945
10	Trinity	US, Intel Xeon E5-2698v3	301,056	8,100	11,078	4,233

<https://www.top500.org/>

Sunway TaihuLight

- June 2016, National Supercomputing Wuxi Center



Sunway TaihuLight

- June 2016, National Supercomputing Wuxi Center



Table 1: Sunway TaihuLight System Summary

CPU	Shenwei-64
Developer	NRCPC
Chip Fab	CPU vendor is the Shanghai High Performance IC Design Center
Instruction set	Shenwei-64 Instruction Set (this is NOT related to the DEC Alpha instruction set)
Node Processor cores	256 CPEs (computing processing elements) plus 4 MPEs (management processing elements)
Node Peak Performance	3.06 TFlop/s
Clock Frequency	1.45 GHz
Process Technology	N/A
Power	15.371 MW (average for the HPL run)
Peak Performance of system	125.4 Pflop/s system in Wuxi
Targeted application	HPC
Nodes	40,960
Total memory	1.31 PB
Cabinets	40
Nodes per cabinet	1024 Nodes
Cores per node	260 cores
Total system core count	10,649,600

How are "very large" computer systems built? -Sunway TaihuLight



■运算节点板

- 1CPU, 260计算核, 系统基本构成单元, 众核处理器+存储器

■运算插件板

- 4 运算节点板高密组装, 4CPU。运算节点+网络接口板

■运算超节点

- 64 运算插件板, 256CPU, 超节点内部采用紧耦合弹性互连

■运算机仓 (Cabinet)

- 4运算超节点, 256运算插件板, 1024运算节点 (CPU)

■整机系统

- 40运算机仓, 160超节点, 40960节点, 10649, 600计算核, 1.31PB



The TOP500 List (June 2026, since 1963)



Rank	System		Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power (kW)
1	LineShine	CN, LingKun, LX2 304C, 1.55GHz	13,789,440	2,198.40	2,735.82	42,220
2	El Captain	US, AMD EPYC, AMD MI300A	11,340,000	1,809.00	2,821.10	29,685
3	Frontier	US, AMD EPYC, AMD MI250X	9,066,176	1,353.00	2,055.72	24,607
4	Aurora	US, Intel Xeon, Intel GPU Max	9,264,128	1,012.00	1,980.01	38,698

<https://www.top500.org/lists/top500/2026/06/>



LX2 Processor – Key Specifications (in the LineShine System)



■ CPU

- ARMv9 w/ SVE&SME
- 304 cores @ 1.5GHz
- 2 dies per CPU (152 cores/die)
- 4 NUMA domains per die
- 32GB HBM + 256GB DDR5

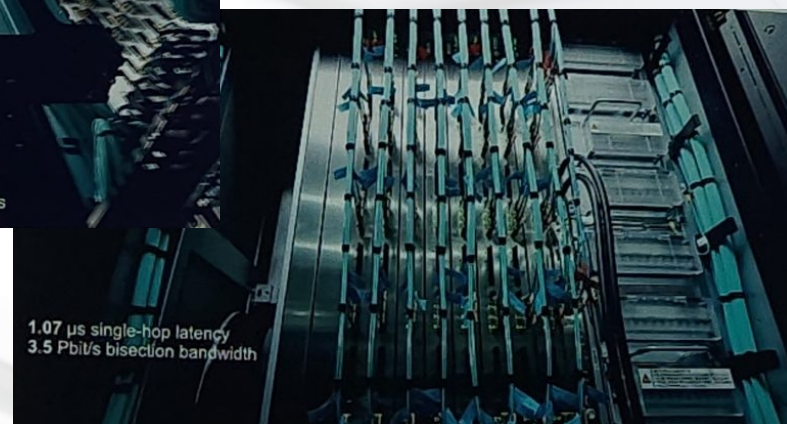


■ Performance

- FP64: 60.3 TFLOPS
- FP32: 120.6 TFLOPS
- FP16/INT8: sve/sme



380V DC power direct to blades



El Captain



■ US LLNL (@加州劳伦斯利弗莫尔国家实验室)



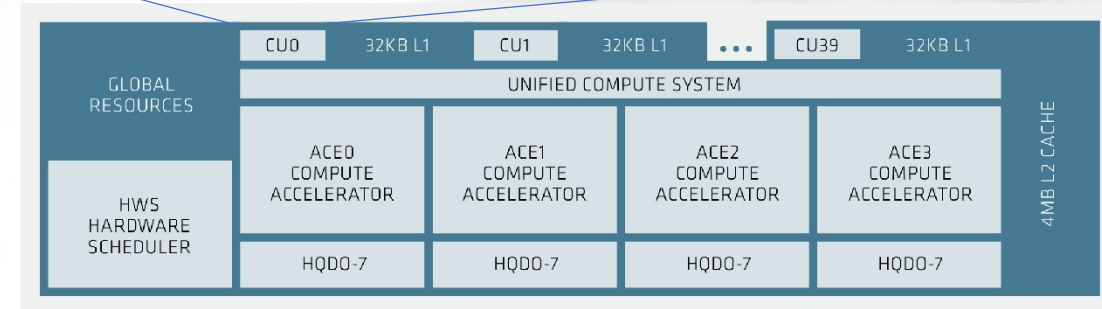
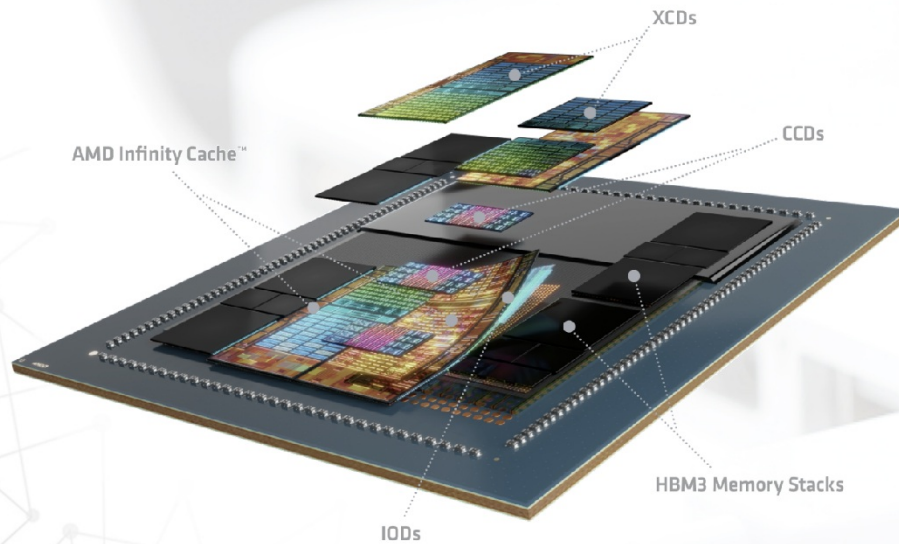
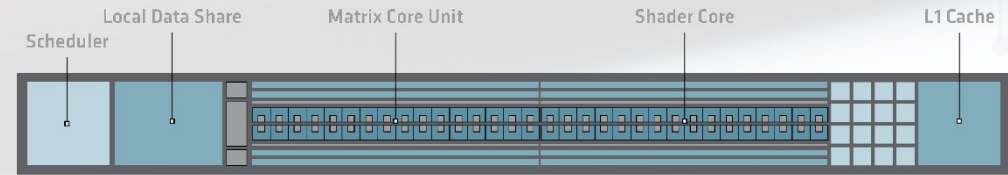
酋长岩，世界上第三台正式发布的E级超级计算机，2.746 ExaFLOPS的双精度浮点峰值性能（1 EFLOPS= 10^{18} FLOPS）。

El Captain – MI300X



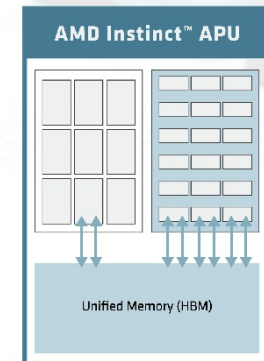
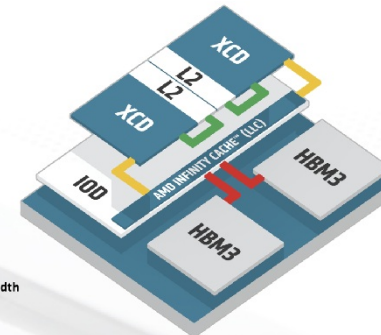
■ AMD Instinct MI300X Accelerator

- CCD (CPU Complex Die)
- XCD (Accelerator Complex Die)
- IOD (I/O Die)



LEGEND

- Green line is L2 to XCD 51.6 TB/s (Aggregate)
- Yellow line is LLC to XCD 17.2 TB/s (Aggregate)
- Red line is LLC to XCD Bandwidth 5.3 TB/s (Aggregate)



El Captain – MI300X



■ AMD Instinct MI300X Accelerator

AI PEAK THEORETICAL PERFORMANCE

		with sparsity
TF32 (TFLOPs)	653.7	1307.4
FP16 (TFLOPs)	1307.4	2614.9
BFLOAT16 (TFLOPs)	1307.4	2614.9
INT8 (TOPS)	2614.9	5229.8
FP8 (TFLOPs)	2614.9	5229.8

HPC PEAK THEORETICAL PERFORMANCE (TFLOPS)

FP64 vector	81.7
FP32 vector	163.4
FP64 matrix	163.4
FP32 matrix	163.4

DECODERS AND VIRTUALIZATION

Decoders [†]	4 groups for HEVC/H.265, AVC/H.264, V1, or AV1
JPEG/MJPEG CODEC	32 cores, 8 cores per group
Virtualization support	SR-IOV, up to 8 partitions

The MI200 Series does not support TF32, FP8, or sparsity

[†]Video codec acceleration (including at least the HEVC (H.265), H.264, VP9, and AV1 codecs) is subject to and not operable without inclusion/installation of compatible media players. GD-176

SPECIFICATIONS

Form factor	OAM module
Lithography	5nm FinFET
I/O Dies (IODs)	6nm FinFET
GPU compute units	304
Matrix cores	1216
Stream processors	19,456
Peak engine clock	2100 MHz
Memory capacity	Up to 192 GB HBM3
Memory bandwidth	5.3 TB/s max. peak theoretical
Memory interface	8192 bits
AMD Infinity Cache™ (last level)	256 MB
Memory clock	Up to 5.2 GT/s
Scale-up Infinity Fabric™ Links	7x 128 GB/s
I/O to host CPU	1 PCIe® Gen 5 x16 (128 GB/s)
Scale-out network bandwidth	PCIe Gen 5 x16 (128 GB/s)
RAS features	Full-chip ECC memory, page retirement, page avoidance
Maximum TBP	750W

Today' s Dominant Target Systems



■ Mobile (smartphone/tablet)

- >1 billion sold/year
- Market dominated by ARM-ISA-compatible general-purpose processor in system-on-a-chip (SoC)
- Plus sea of custom accelerators (radio, image, video, graphics, audio, motion, location, security, etc.)

■ Warehouse-Scale Computers (WSCs)

- 10,000,000's cores per warehouse
- Market dominated by x86-compatible server chips
- Dedicated apps, plus cloud hosting of virtual machines
- Now seeing increasing use of GPUs, FPGAs, TPUs, custom hardware to accelerate workloads

■ Embedded Computing

- Wired/wireless network infrastructure, printers
- Consumer TV/Music/Games/Automotive/Camera/MP3
- Internet of Things!

AI Chips



厂商	代表产品	制程	片上/显存	峰值算力（厂商口径）	定位
NVIDIA	B200 (Blackwell)	4nm	192 GB HBM3e	~2.25 PFLOPS FP8 稠密 / 4.5 PFLOPS FP4 稠密	训练+推理
NVIDIA	GB200 NVL72	4nm	72×192 GB HBM3e	~1.44 EFLOPS FP4	超大规模训练/推理
NVIDIA	Rubin (2026 H2)	3nm	288 GB HBM4	待公开	下一代训练+推理
AMD	MI325X	5nm	256 GB HBM3e	~2.6 PFLOPS FP8	训练+推理
AMD	MI350X (2025)	3nm	288 GB HBM3e	宣称 4× MI300X	训练+推理
Google	TPU v6 Trillium	未公开	2× HBM	宣称 4.7× TPU v5e	训练+推理
Google	TPU v7 Ironwood	未公开	未公开	~2300 BF16 TFLOPS（单芯片）	下一代训练
AWS	Trainium2	未公开	96 GB HBM3/chip	~20.8 PFLOPS/节点（多芯片）	训练
AWS	Inferentia2	未公开	32 GB HBM2e	未公开	推理
Intel	Gaudi 3	5nm	128 GB HBM	~1835 BF16 TFLOPS	训练+推理

AI Chips



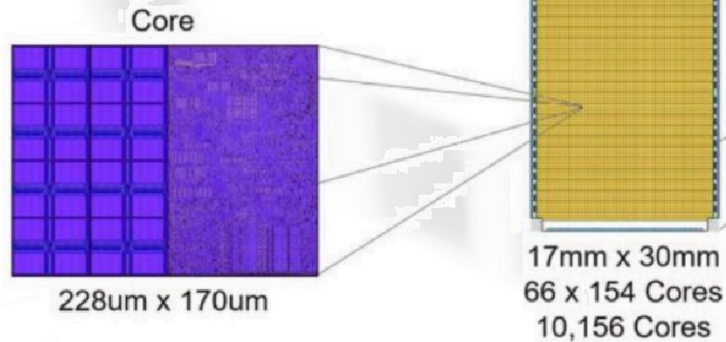
厂商	代表产品	制程	片上/显存	峰值算力（厂商口径）	定位
华为	昇腾 910C	7nm	96 GB HBM2e	~800 TFLOPS FP16	训练+推理
寒武纪	MLU590	7nm	未公开	800 TOPS/W（能效比）	训练+推理
Groq	LPU (TSP)	14nm→4nm 演进	230 MB SRAM/chip	确定性超低延迟	推理专用
Cerebras	WSE-3 (CS-3)	5nm	44 GB SRAM	~125 PFLOPS FP16	训练+推理
SambaNova	SN40L RDU	5nm	1.5 TB 三级存储	~640 BF16 TFLOPS	训练+推理
燧原科技	云燧 T21/T20	未公开	未公开	未公开	训练+推理
摩尔线程	MTT S4000	未公开	未公开	未公开	训练+推理/图形
壁仞科技	BR100/BR104	7nm Chiplet	未公开	未公开	训练+推理
海光	深算二号 DCU	7nm	512 GB（系统级）	未公开	训练

<https://www.ai-insight.org/reports/chip-landscape-2026>

2026-9-3

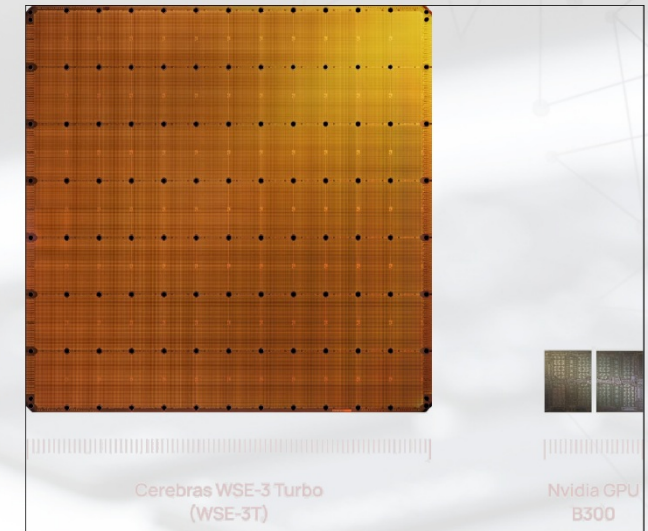
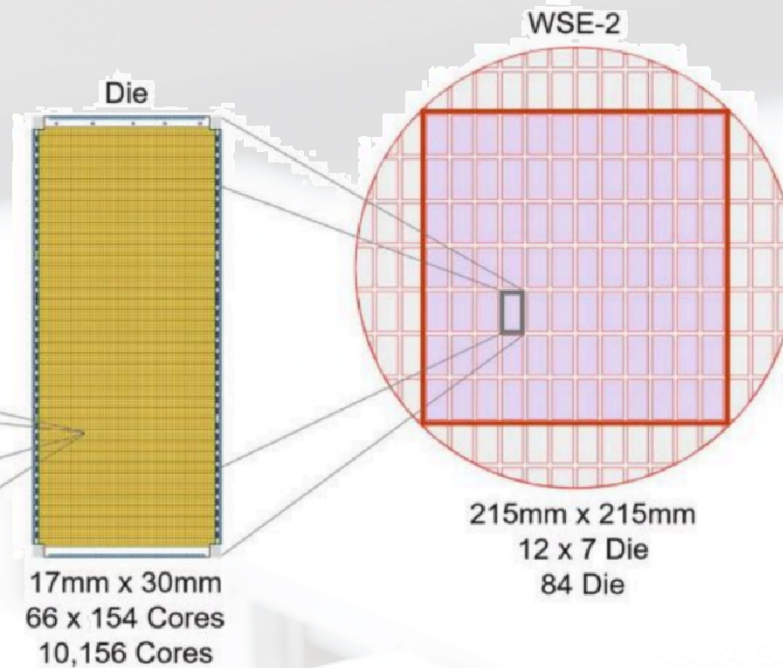
Cerebras Wafer Scale Engine (WSE-3)

- 4 trillion transistors
- 46225 mm² silicon
- 900000 cores
- 5nm TSMC
- 125 PFLOPS FP16

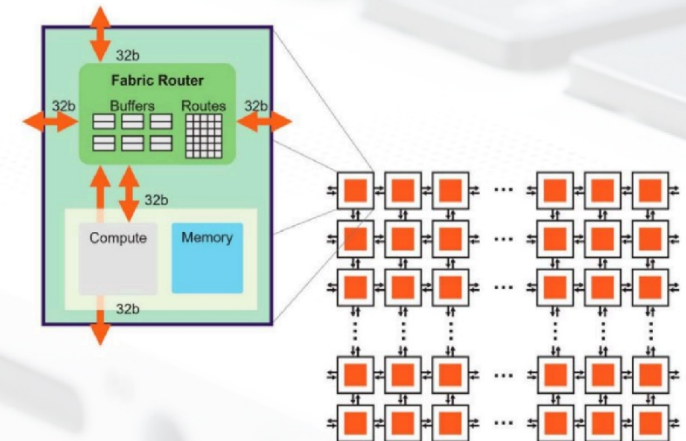


The Cerebras core physical design:
50% of the area is static random-access memory (SRAM) and 50% of the area is logic.

<https://cdn.sanity.io/files/e4qjo92p/production/2d7fa58e3b820715664bcf42097e86c05070c161.pdf>



WSE-3 vs. NV GPU B300



Huawei CloudMatrix384

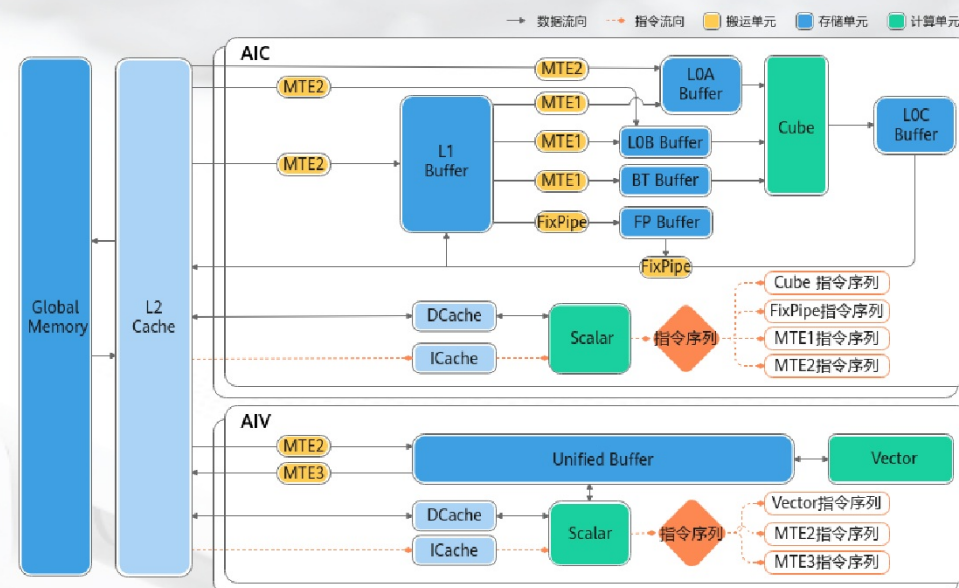
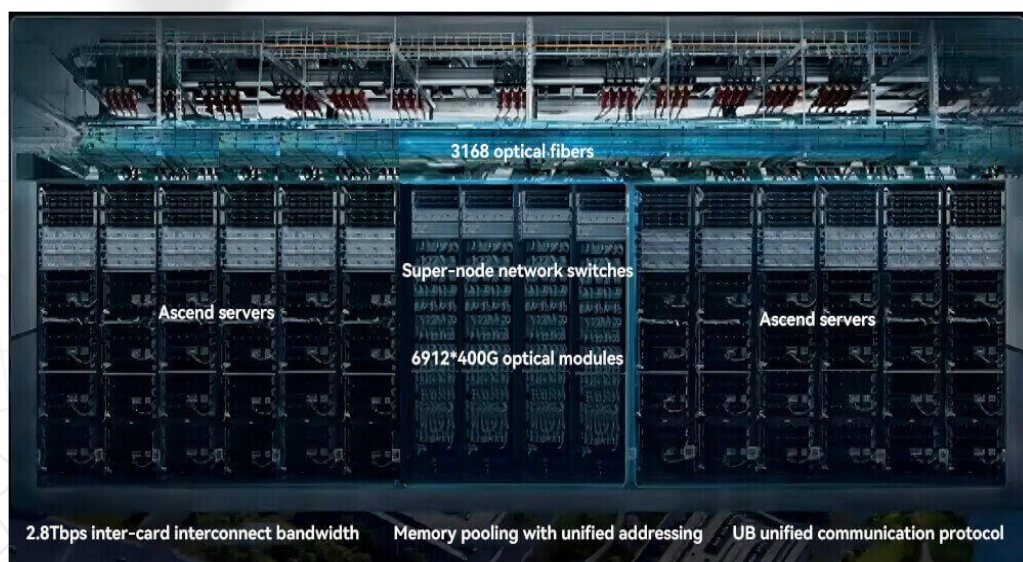


■ CloudMatrix384 supernode

- 384 Ascend 910C NPUs + 192 Kunpeng CPUs
- fully interconnected via unified bus (UB).

■ Key innovations:

- UB bus: 392 GB/s per NPU, 200 ns latency
- Dual-die packaging: 2 compute dies (752 TFLOPS FP16), 128 GB HBM shared



What's the difference between "Big" and "Small"?



**Smart
Phones**

VS

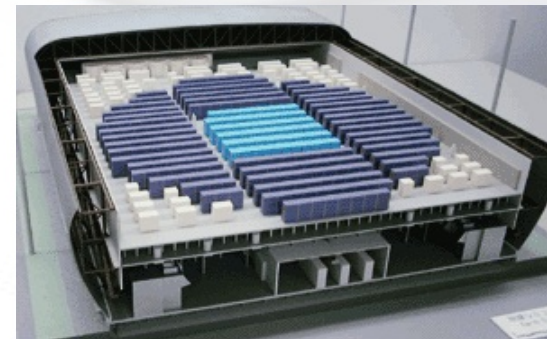


Laptops

VS



Data-Centers



Supercomputers

0 Course Crew

1 What' s the difference between “Big” and “Small” ?

2 Why Take This Course?

3 Great Ideas in Computing Systems

4 Course Implementation

Why Take This Course?



■ Foundational Goal

Deeply understand Intersects all aspects of computing system

■ Preparatory/Complementary

- Algorithm and data structures
- Programming Language
- Compilers and Interpreters
- Operating Systems
- Digital Systems Organization and Design
- Mathematical Foundations of CS

■ Fun!!!

- Who wouldn't want to understand the magic?

■ 存在问题

- 缺乏一门独立的能够贯穿整个计算机系统的基础课程
- 缺少大课时的硬课：无法给课程的深度提供空间
- 课程之间的衔接和关联不够：无法给课程的宽度提供空间
- 课程内容比较陈旧
- 课程体系缺乏对系统设计和应用能力培养的整体考虑

■ 世界一流计算机专业给我们的启示

- 用什么来衡量计算机教育做得好的高校？
- 计算机教育影响力最大的美国高校
- 院校排名 (2019年US News的最新排名)

当代计算机专业学生最重要的核心竞争力是什么？

四大专业基本能力		能力点数
计算思维能力		9
算法设计与分析能力		8
程序设计与实现能力		3
系统能力	系统认知	6
	系统设计	19
	系统开发	23
	系统应用	14

中国科大新一轮计算机专业教学改革：强调“系统观”



■ 计算机学科的发展趋势

- 计算x学：越来越多的跨学科应用，并行无处不在
- 人工智能+：无处不在
- 后硅时代，量子芯片和量子计算机越来越成为现实

■ 中国科大计算机教育的目标和理念、特色和优势

- 计算机系统
- 高性能计算
- 人工智能
- 量子计算

■ 改革思路

- 用“系统观”来指导计算机专业培养方案和课程体系的设计
- 增加一门计算机系统基础课程：《计算机系统概论》
- 重新组合的核心课程，给课程的深度留出空间
- 调整和更新方向课，拓展课程的宽度

2026-9-3



中国科学技术大学
University of Science and Technology of China

系统观指导下的计算机专业课程体系设计和课程改革

报告专家：安虹

教授，博士生导师，安徽省教学名师，现任计算机科学与技术学院先进计算机系统结构实验室主任，计算机系统结构专业学位负责人，院教学委员会和学术委员会委员，学生竞赛实践创新教育委员会主任，本科生系统结构类课程组组长，教育部基础学科拔尖学生培养试验计划研究课题负责人，教育部高等学校计算机专业类教学指导委员会计算机系统专委会委员，提出创办了计算机专业英才班“华夏班”，组织了第一届华夏班计算机系统结构教学体系和核心课程改革工作。新开设了《计算机系统概论（II）》（本科生英才班）、《计算机系统原型设计》（本科生）、《高性能处理器体系结构》（本硕）、《计算机系统性能评价与预测》（博士生）等一系列计算机系统结构相关课程，主持过10余项校级和省部级计算机教育研究项目，2012年组建了中科大超算鸿雁队，并担任主教练，先后组织了34支队伍182人次参加了国内外高性能计算相关赛事 SC-SCC、ISC-SCC、ASC、RMDA、PAC、CPC、HPC-AI，其中获得一等奖或第一名12次，二等奖或第二名7次，三等奖或第三名7次，特别奖7次。在SC16大会上获得的大学生超算竞赛中，中国科学技术大学代表队包揽了总分和最高LINPACK性能两项冠军，成为SC大学生超算竞赛自2007年举办十年以来首个双料冠军队。教学方面获得过中科院教育成果一等奖，安徽省教学成果一等奖，中国科大教学成果一等奖，中科院朱李月华优秀教师奖，宝钢优秀教师奖（两次），华为优秀教师奖，中国科大平凡基金会教育奖，中国科大西区精神教学成就奖，安徽省教学名师，安徽省模范教师等奖项和荣誉。

时间：2020年4月11日（周六）下午3:00
地点：Zoom线上会议（会议ID:2300271958）
网址：<https://zoom.com.cn/j/2300271958>

报告摘要：
“系统观”对于计算机大类专业人才培养是一个很重要的视角，不但影响专业课程的教学内容安排，也是专业课程体系设计的重要依据。正如图灵奖得主UC Berkeley EECS系David Patterson教授所言：“在异构计算的时代，程序员必须对于算法和硬件模型融会贯通，才能写出高质量的代码，因此，未来的程序员也必须懂硬件！”本报告在分析美国计算机专业排名前五的几所大学计算机专业的培养方案，以及“ACM / IEEE-CS Computing Curriculum 2020”的基础上，结合中国科大计算机大类新一轮培养方案的重新制定工作，以计算机学院计算机专业课程体系设计和课程改革为实例，讨论如何在新的时代背景下落实计算机教育的“系统观”，如何取舍课程数量、课时，复杂性、系统性等，如何开展消肿、更新和补缺的课程设计，形成长效、可扩展的人才培养方案。

主办单位：计算机科学与技术学院

欢迎全校师生参加！

47

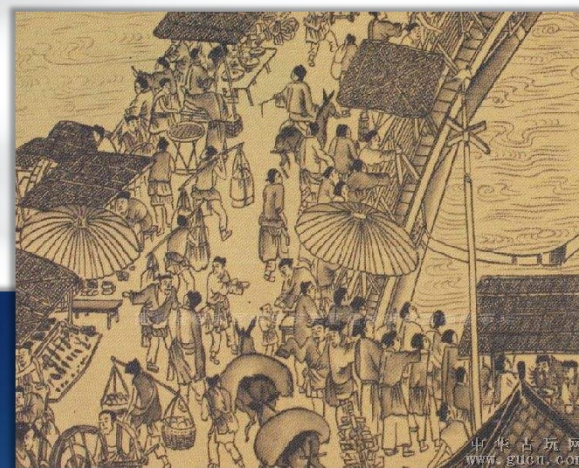
要从本科低年级开始逐步培养学生的**计算思维能力**



计算思维：利用包括网络在内的计算系统进行问题求解（自然问题、技术问题和社会问题**）的思维方式**



自然

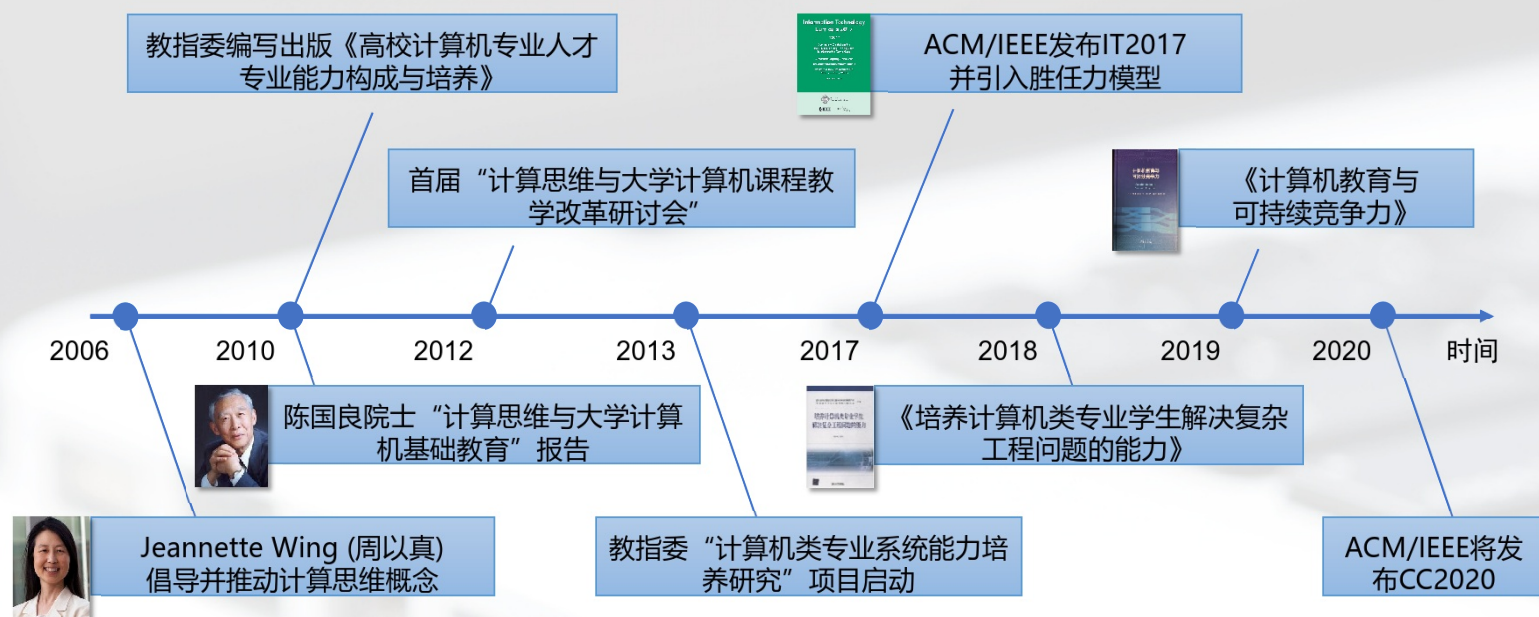


社会



技术

计算思维的发展历程(内涵与外延)



“在异构计算的时代程序员必须对于算法和硬件模型融汇贯通，才能写出高质量的代码。因此，未来的程序员也必须懂硬件！”

——图灵奖得主David Patterson

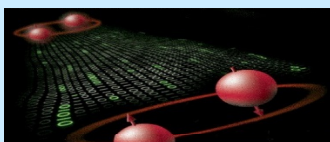
什么是计算思维能力?



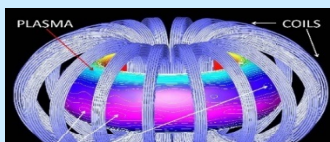
不同的历史时期，人类关注的重大问题不同

- **20世纪**：战争、人口增长、饥饿等
- **21世纪**：自然资源消耗过快、**环境污染**、气候异常、**健康医疗**、人口老龄化、贫富差距过大、城市交通、非传统**安全问题**（非典型传染病爆发、金融危机、恐怖主义、网络攻击等）

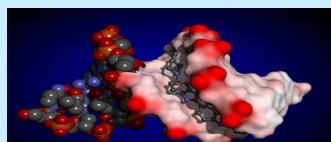
科学与工程计算



量子计算模拟



可控核聚变模拟



蛋白质与药物设计

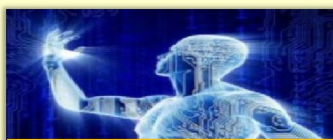


大气环境监测

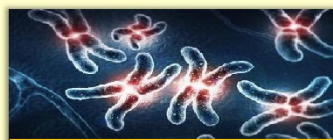
大数据，人工智能和云计算



健康大数据



人工智能



基因分析



金融分析

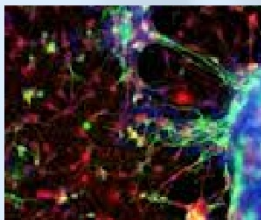
Artificial intelligence is everywhere



Supercomputers



Business
analytics



Drug design

Data Centers

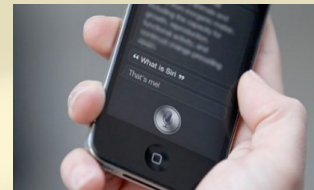


Ad prediction



Automatic
translation

Smartphones



Audio recognition



Image
analysis

Embedded Devices

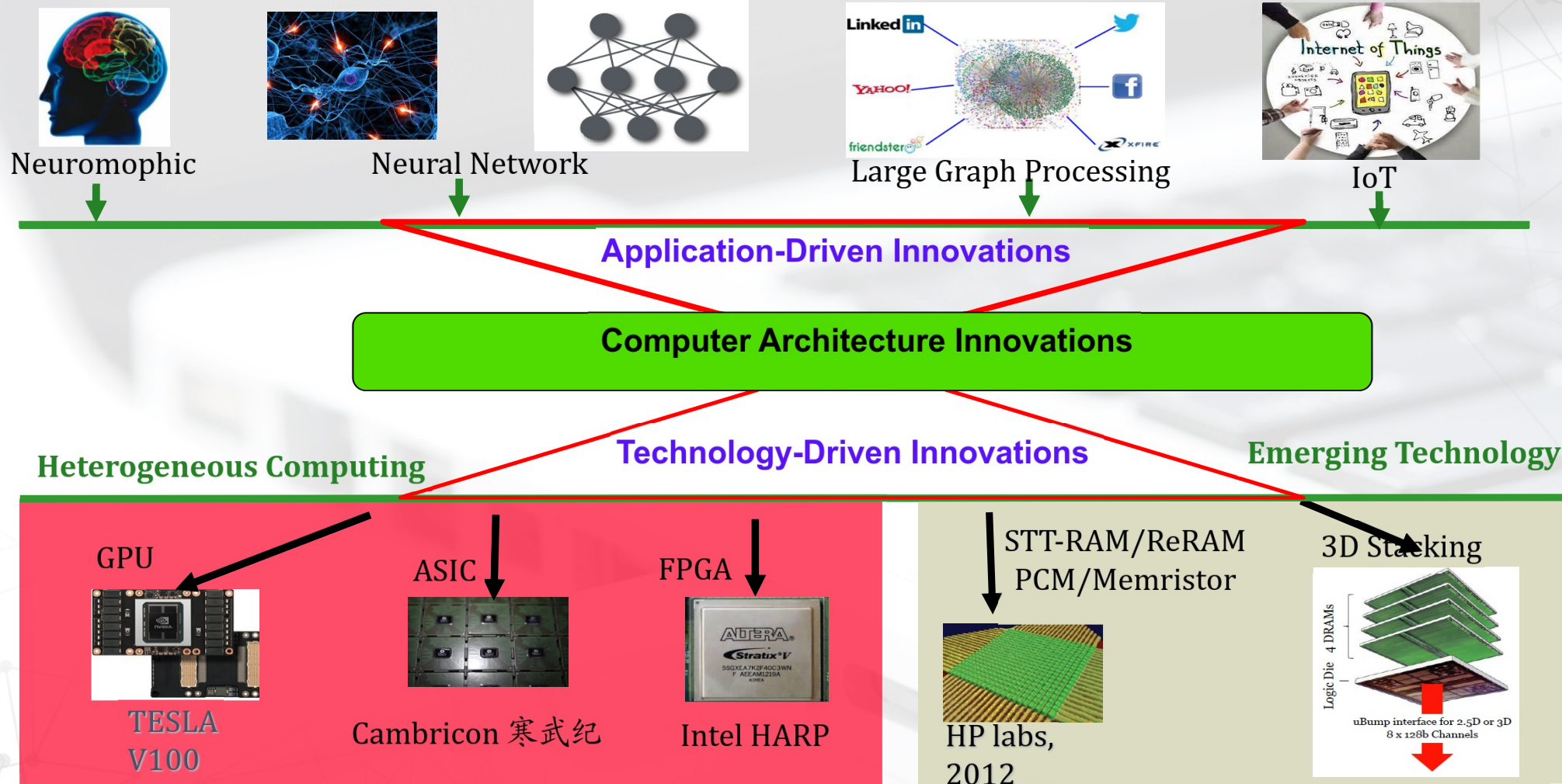


Robotics



Consumer
electronics

New applications demand new structures all the time



人类如何实现从物理设备到问题求解的?



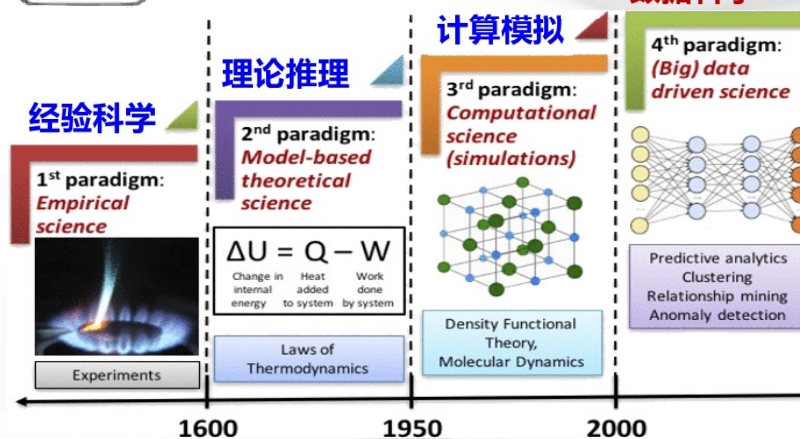
Application



差距太大,
一步无法跨越!

也有例外,
例如罗盘

Physics



0 Course Crew

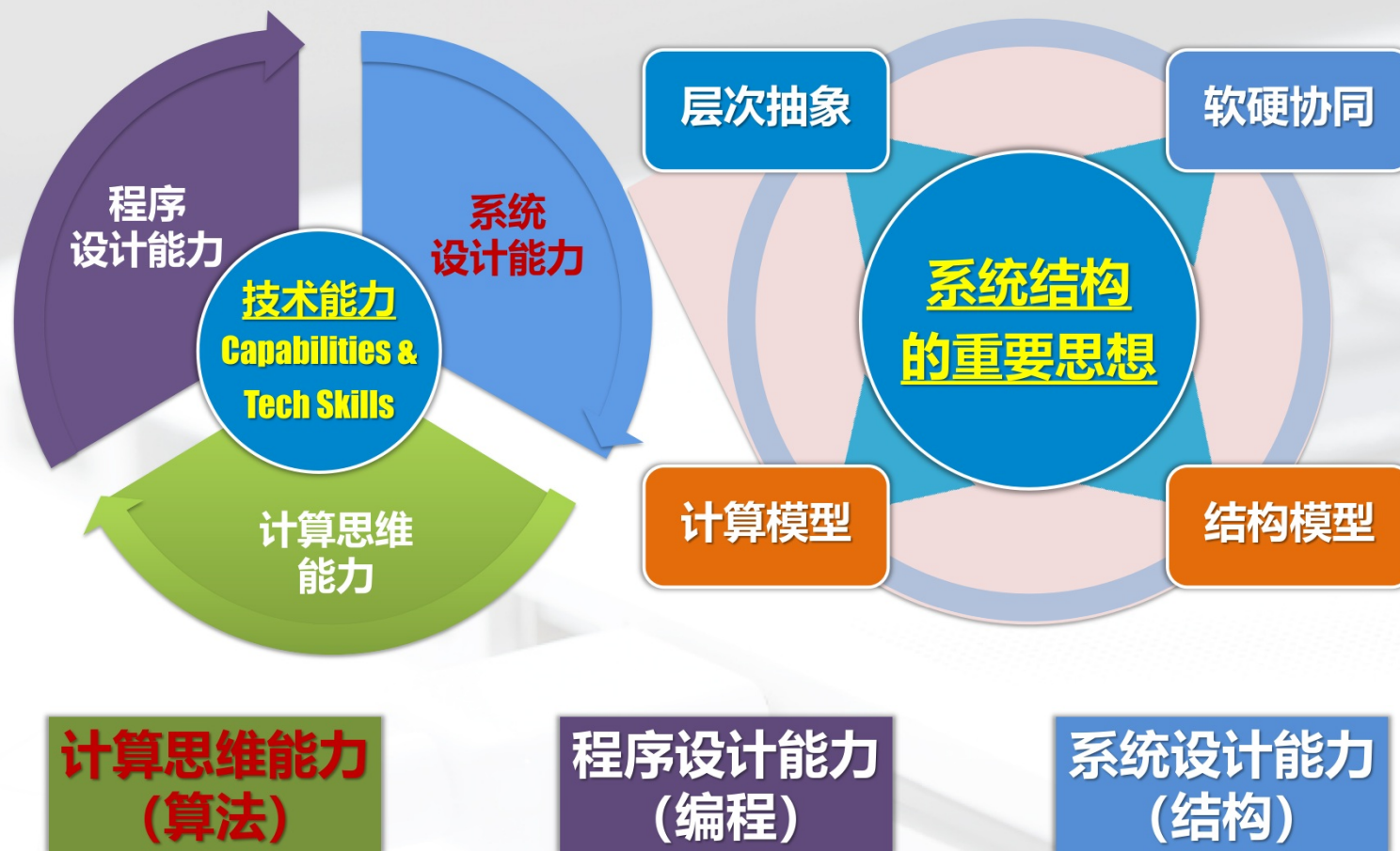
1 What' s the difference between “Big” and “Small” ?

2 Why Take This Course?

3 Great Ideas in Computing Systems

4 Course Implementation

课程目标：理解计算机系统结构的重要思想



Great Ideas in Computing Systems in This Courses



- **Great Idea #0: Great Idea from Ancient Chinese Philosophy(Bits and Bytes)**
- **Great Idea #1: Computer is an Universal Computing Device(Turing Machine Model)**
- **Great Idea #2: Stored program computer(Von Neumann Model)**
- **Great Idea #3: Abstraction Helps Us Manage Complexity(Layers of Representation/Interpretation)**
- **Great Idea #4: Software and Hardware Co-design**

Great Idea from Ancient Chinese Philosophy



All things come into being, all things come into nothing

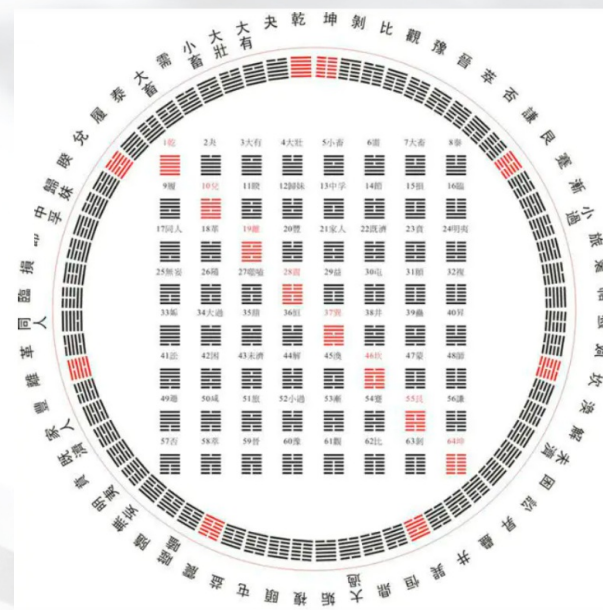
天下万物生于有, 有生于无

《老子·四十章》

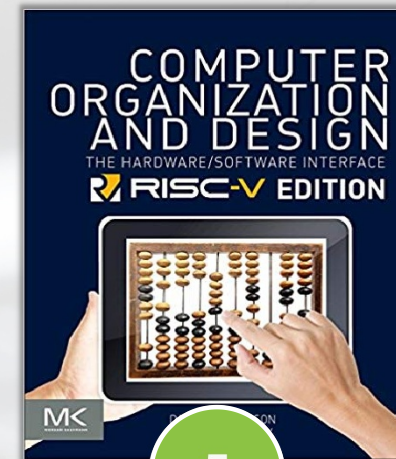
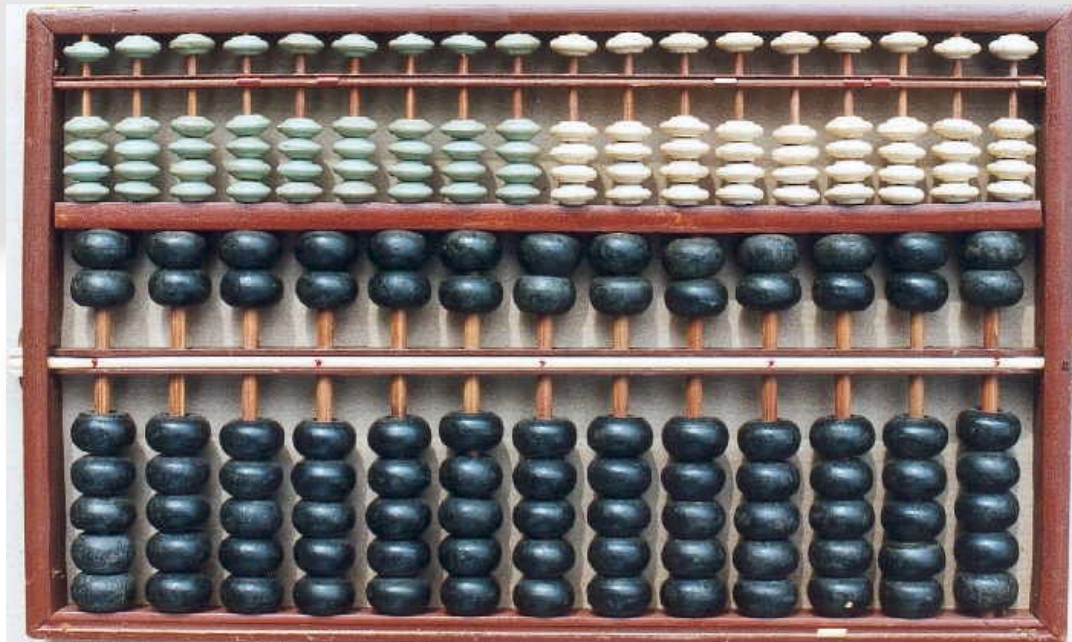


《易经》

太极生两仪,
两仪生四象,
四象生八卦,
八卦演万物。



Great Idea from Ancient Chinese Philosophy



5

Computer Organization and Design: The Hardware/Software Interface,

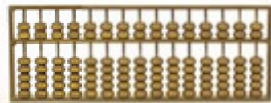
David A. Patterson,
John L. Hennessy,
5th edition. Morgan
Kaufmann
Publishers, Inc., 2017

A computing tool that does not use electricity



■ Abacus (公元前500年, 中国)

Abacus
China
c. 1970
Loan of Gwen and Gordon
Bell, B1643.01



Soroban
Japan
c. 1960
Loan of Gwen and Gordon
Bell, B1659.01



Schoty
Russia
Early 20th century
Gift of Warren Yogi, 1020



**Table abacus
(reproduction) and
jetons**
Germany
17th century
Loan of Michael R.
Williams, L2003.3.2

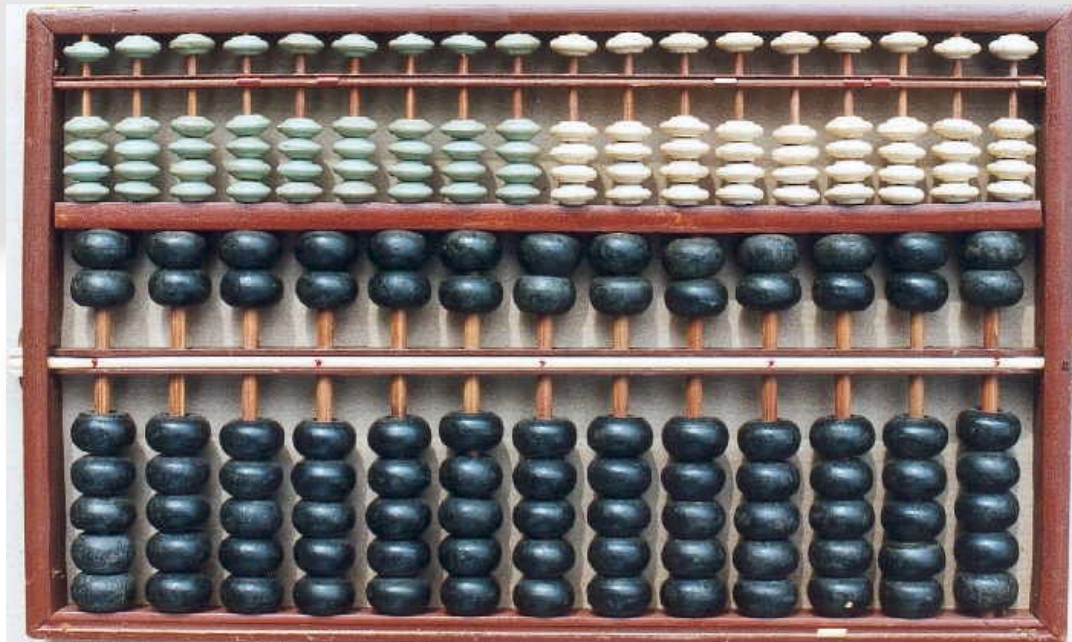


Counting Frame
Early 20th century
Gift of Gwen and
Gordon Bell, B141.80





Great Idea from Ancient Chinese Philosophy



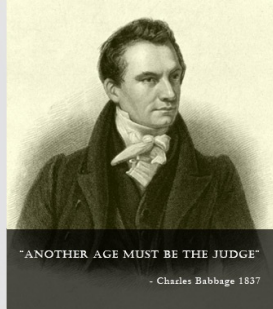
5

Computer Organization and Design: The Hardware/Software Interface,

David A. Patterson,
John L. Hennessy,
5th edition. Morgan
Kaufmann
Publishers, Inc., 2017

Is an abacus a computer?

Automatic computing equipment: from mechanical computer to electronic computer



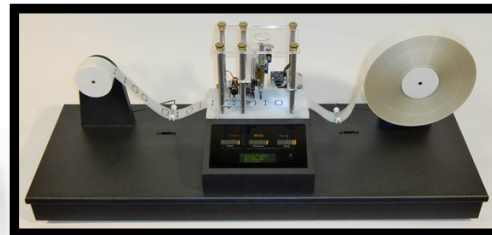
**Charles Babbage,
1791 – 1871, England**



**1832, 2002, 2008
The Babbage Difference
Engine, 17 years, 25,000
parts, 5ton, cost: £17,470**



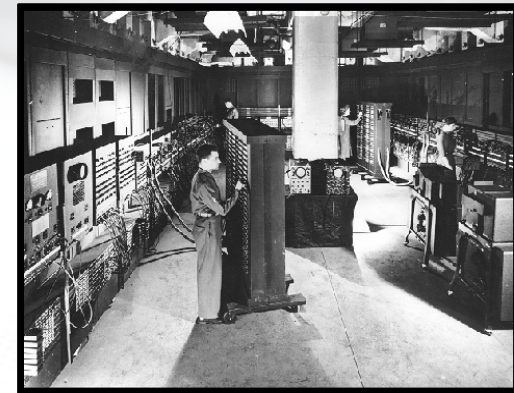
**Alan
Turing(24)**



**Turing Machine,
1936**



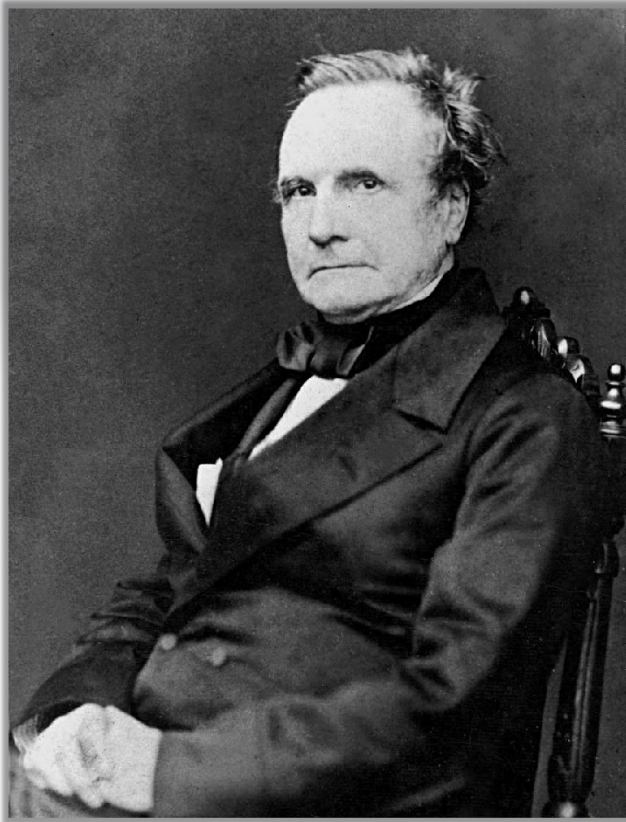
Eckert(24) and Mauchly(36)



ENIAC

1946

Charles Babbage (1791-1871): A Fallen Hero!



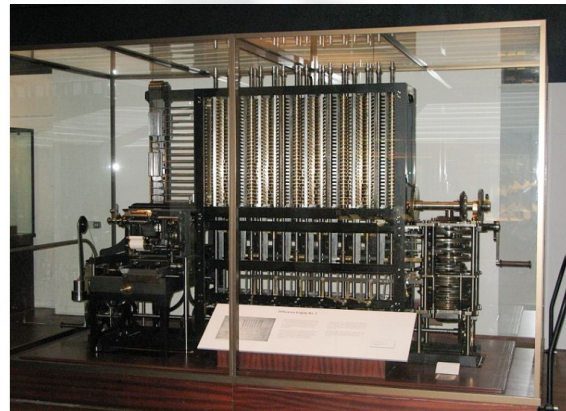
[Copyright expired and in public domain.
Image obtained from Wikimedia Commons.]

- Lucasian Professor of Mathematics, Cambridge University, 1828-1839
- A true “polymath” with interests in many areas
- Frustrated by errors in printed tables, wanted to build machines to evaluate and print accurate tables
- Inspired by earlier work organizing human “computers” to methodically calculate tables by hand

$$F(x) = 10x^2 + 2x + 4, x \in \mathbb{N}$$

x	0	1	2	3	4	5	6
F(x)	4	16	48	100	172	264	376
$\Delta F^1(x)$	12	32	52	72	92	112	
$\Delta F^2(x)$	20	20	20	20	20		

Babbage difference engine: the first mechanical computer (1832)



- 2002, 2008
- The Babbage
- Difference Engine
- 17 years,
- 25,000 parts, 5ton
- cost: £17,470



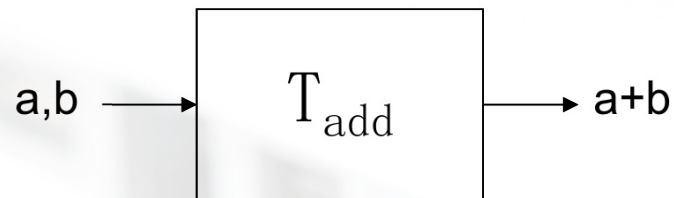
Turing Machine



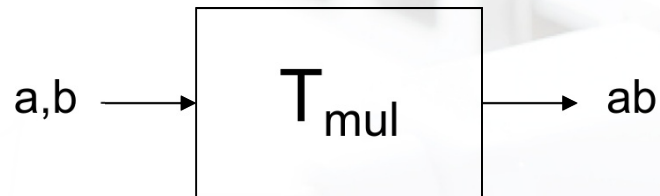
■ Mathematical model of a device that can perform any computation – Alan Turing (1937)

- ability to read/write symbols on an infinite “tape”
- state transitions, based on current state and symbol

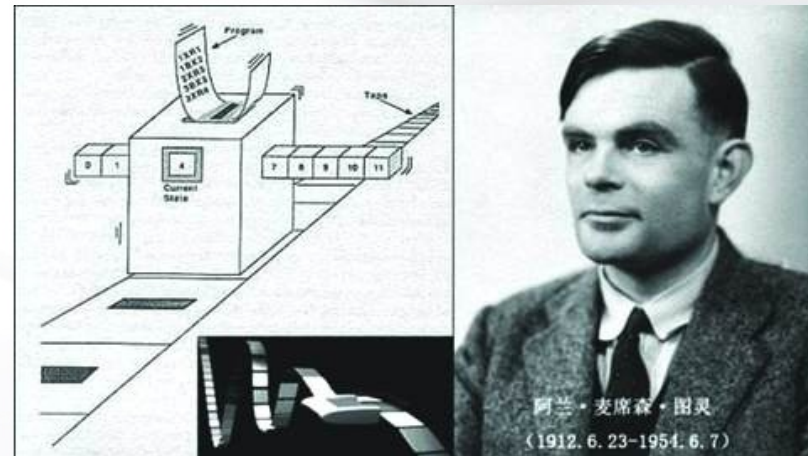
■ Every computation can be performed by some Turing machine. (*Turing's thesis*)



Turing machine that adds



Turing machine that multiplies

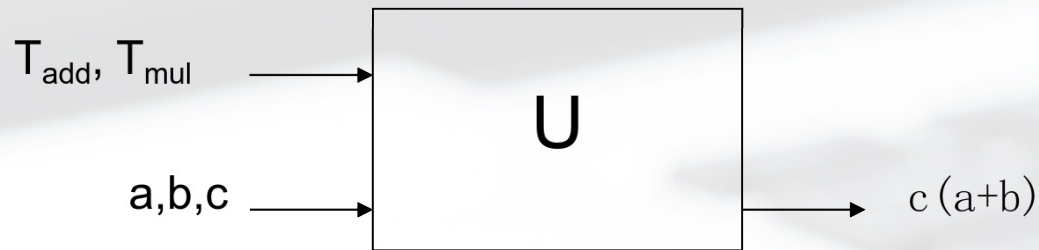


Universal Turing Machine



■ Turing described a Turing machine that could implement all other Turing machines.

- inputs: data, plus a description of computation (Turing machine)



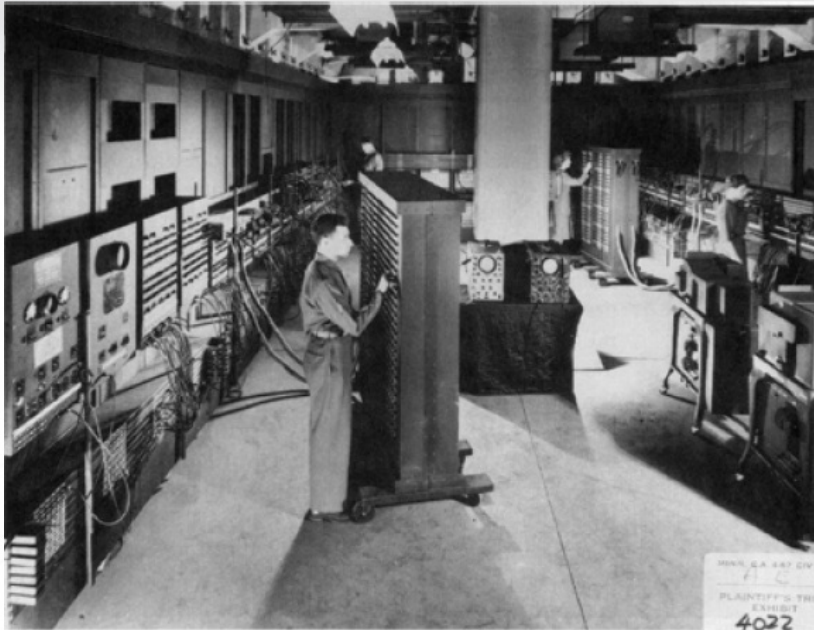
Universal Turing Machine

■ U is programmable – so is a computer!

- instructions are part of the input data
- a computer can emulate a Universal Turing Machine, and vice versa

■ *Therefore, a computer is a universal computing device!*

ENIAC - The first electronic computer ,1946



ENIAC(Electrical Numerical Integrator And Calculator)

- 17,468 vacuum tubes (真空管)
- Power 150kW
- Weighed 30 tons
- Occupied 1800 sq ft
 - 80 feet long
 - 8.5 feet high
- Clock: 100kHz, About 5000 additions per second
- RAM: ~230bytes, Could store 20 numbers
Could store 20 numbers in main memory
- IO: punched card
- Cost about \$500,000



1904, The world's first electron tube was born at the hands of the British physicist Fleming

ENIAC (1946)



■ First electronic general-purpose computer

- Construction started in secret at UPenn Moore School of Electrical Engineering during WW-II to calculate firing tables for US Army, designed by **Eckert** and **Mauchly**
- Twelve 10-decimal-digit accumulators
- Had a conditional branch!

■ Programmed by plugboard and switches, time consuming!

■ Purely electronic instruction fetch and execution, so fast

- 10-digit x 10-digit multiply in 2.8ms (2000x faster than Mark-1)

■ As a result of speed, it was almost entirely I/O bound

■ As a result of large number of tubes, it was often broken (**5 days was longest time between failures**)

Great Idea #1: Computer is an **Universal Computing Device**



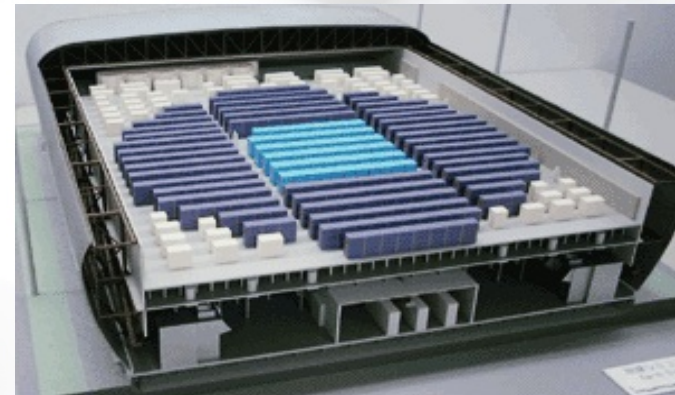
- All computers, **given enough time and memory**, are capable of computing exactly the same things.



=



=



Supercomputers

From Theory to Practice



■ In theory, computer can *compute* anything

■ that' s possible to compute

- given enough *memory* and *time*

■ In practice, *solving problems* involves computing under constraints.

- time

- weather forecast, next frame of animation, ...

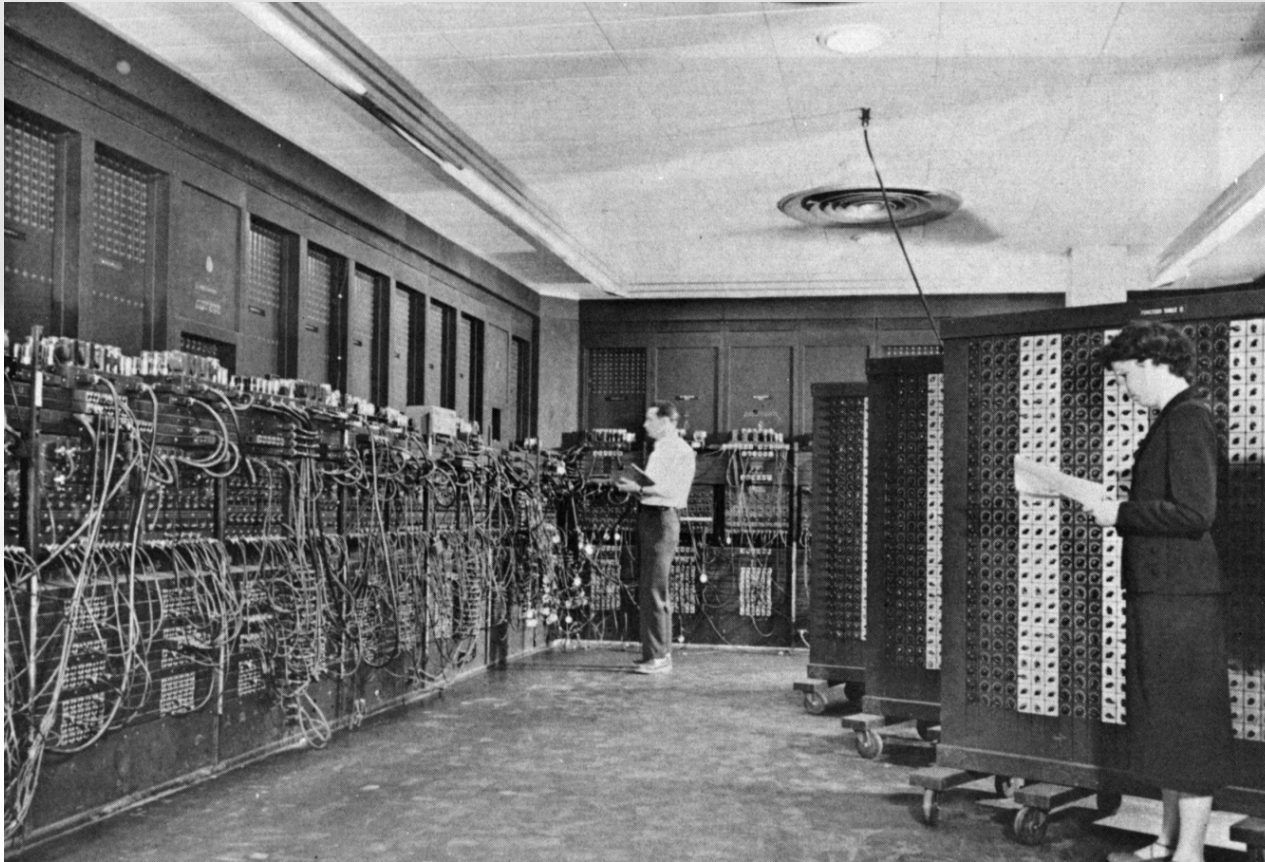
- cost

- cell phone, automotive engine controller, ...

- power

- cell phone, handheld video game, ...

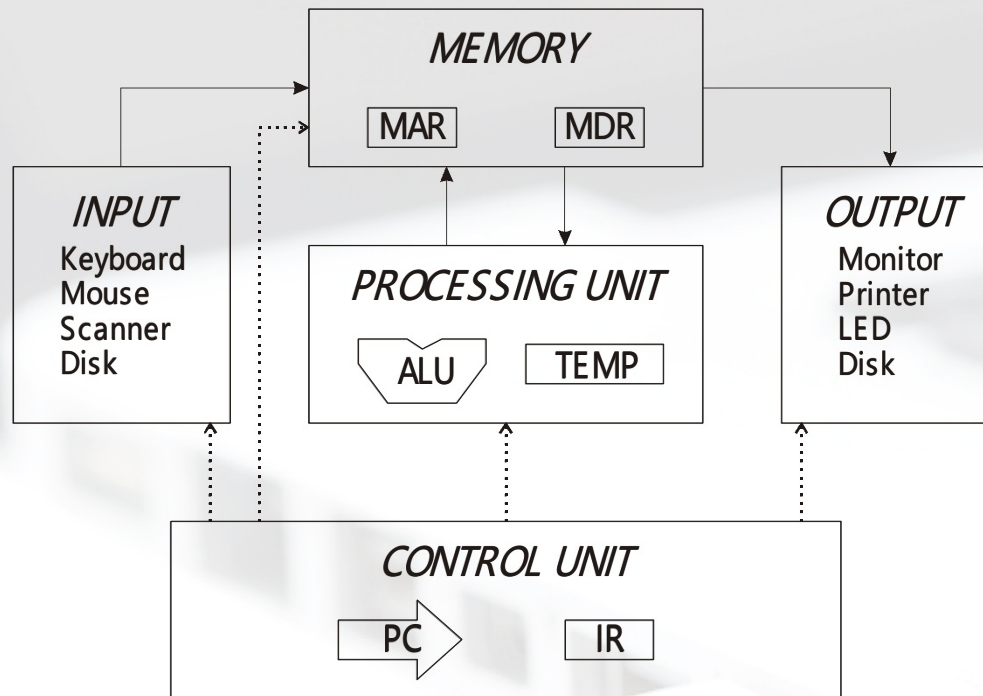
ENIAC



Changing the program could take days!

Great Idea #2

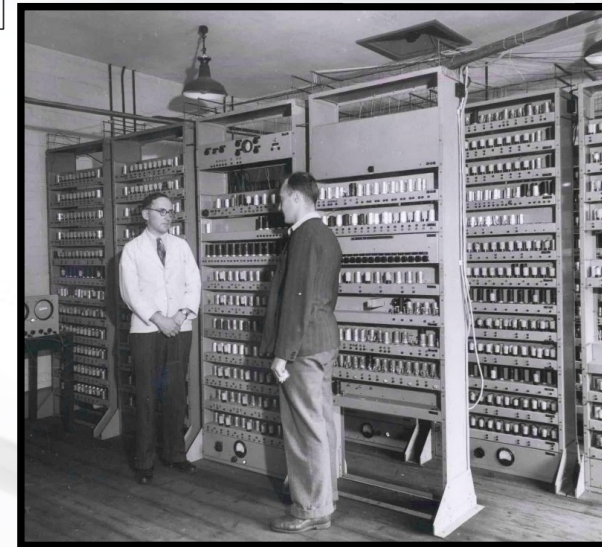
Von Neumann Model(存储程序计算机)



Von Neumann



Maurice
Vincent Wilkes
Turing Award @1967



EDSAC, University of
Cambridge, UK, 1949

我们的祖上“挺阔的”



- 中国科大计算机专业创建于1958年, 当时隶属于应用数学和计算技术系

- 应用数学和计算技术系首任系主任: 华罗庚

- 计算机专业首任教研室主任: 夏培肃

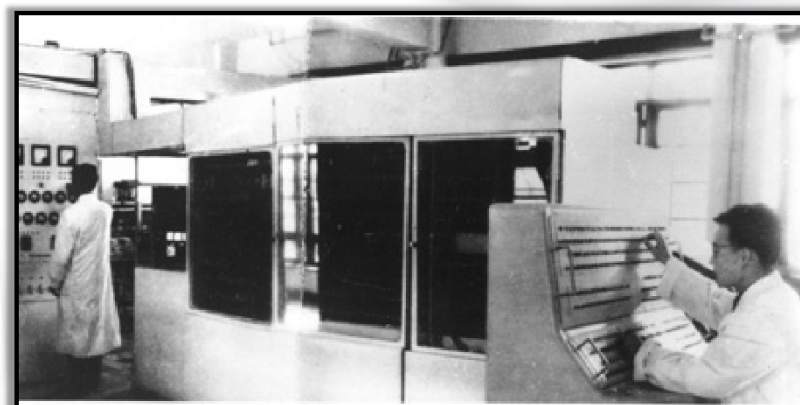
- 中国第一个计算机三人小组

- 1952年, 华罗庚教授会见了三位年轻科学家, 讨论的是一个前沿话题: 研制中国的计算机。由此中国第一个计算机三人小组成立。他们是夏培肃、闵乃大, 王传英。



我校首台计算机：107计算机

- 夏培肃先生主持研制
- 中国第一台自主设计的通用电子计算机
- 中国第一台自主设计的冯·诺依曼结构计算机
 - EDVAC, 美国, 1945-1952
 - EDSAC, 英国, 1945-1949
 - 107机, 中国, 1953-1959
- 1960年在中国科大投入使用 (命名为KD-1)
- 1970年随科大下迁至合肥
- 1974年被拆除



首批教师：参与107机研制的教师



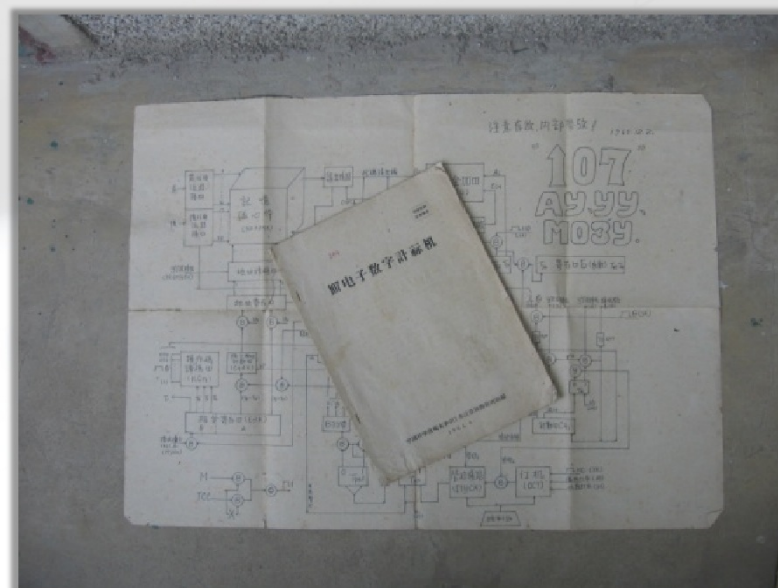
夏培肃院士（左六）与我校计算机学科早期建设者、107机研制者、原计算机系统结构教研室主任郑世荣教授、钟津立、周行仁、赵鼎文、杨学良、王武良等在玉泉路校园合影。



我校首套计算机教材



- 夏培肃先生主持编写的我校第一套《计算机原理》教材。也是国内最早的《计算机原理》教材。

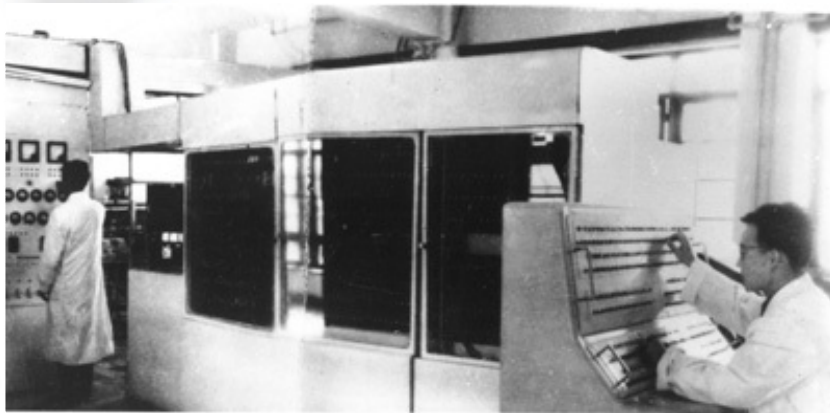


The USTCers' outstanding contributions to the Chinese computer

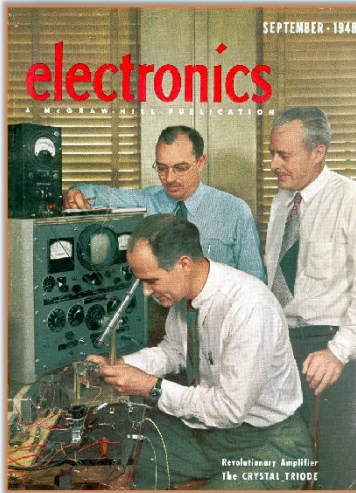
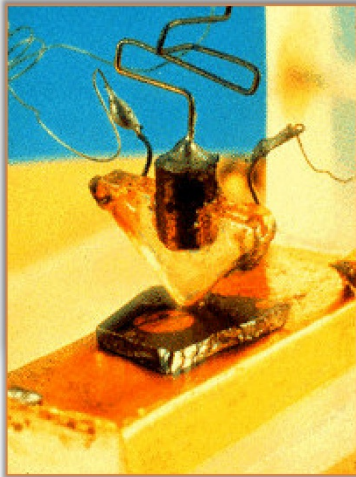


- 1959年, 107计算机
- 中国第一台自主设计的通用计算机
- 主设计师: 夏培肃

- 2002年, 龙芯1号
- 中国第一颗自主设计的通用微处理器芯片
- 主设计师: 胡伟武(86本)



The invention of the transistor



Bell Labs lays the groundwork:

- 1945: Bell sets up lab in the hopes of developing “solid state” components to replace existing electromechanical systems. William Shockley, John Bardeen, Walter Brattain: all solid-state physicists. Focus on Si and Ge.
- 1951: Shockley develops junction transistor which can be manufactured in quantity.
- 1954: The first transistor radio! Also, TI makes first silicon transistor (price \$2.50)
- 1956: Bardeen, Shockley, Brattain receive Nobel Prize.

Two major inventions of the microprocessor chip



Stored program + Transistor technology



Change the program so that you can do all kinds of tasks **on the same hardware**



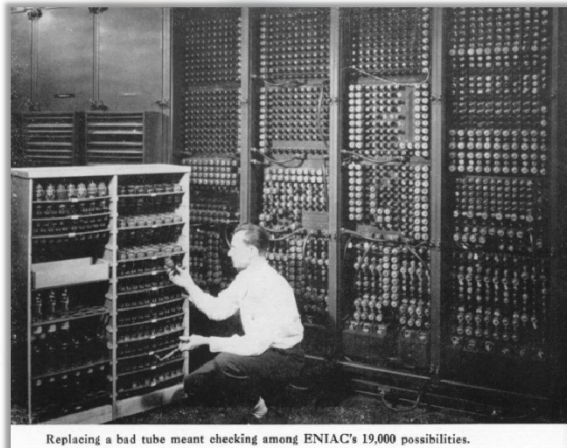
The device is **smaller** and **faster** than a vacuum tube

First computer vs. First microprocessor chip



1946, ENIAC(Electrical Numerical Integrator And Calculator)

- 18000 vacuum tubes
- 1500 relays
- 174 KW
- 30 tons
- 1800 sq. ft. footprint
- Clock: 100kHz
- RAM: ~230bytes
- IO: punched card

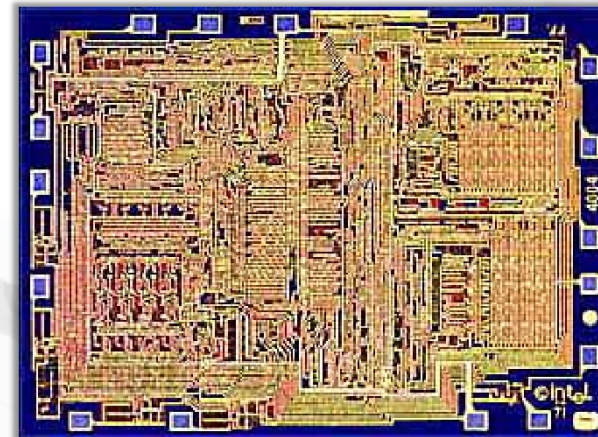


Replacing a bad tube meant checking among ENIAC's 19,000 possibilities.

1971, Intel 4004

- 10 micron process, NMOS-Only Logic
- 2,300 transistors
- 3cmx4cm die
- 4-bit bus
- Performance < 0.1 MIPS
- 640 bytes of addressable Memory
- 740 KHz

After 25 years



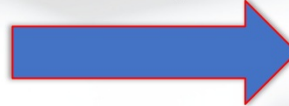
Thirty years after the first microprocessor chip was born



1971, Intel 4004

- 10 micron process
- **2,300 transistors**
- 3x4 mm die
- 4-bit bus
- 640 bytes of addressable Memory
- **750 KHz**

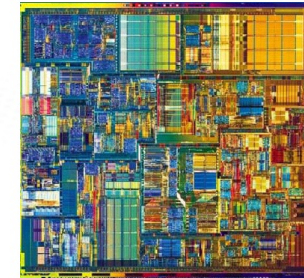
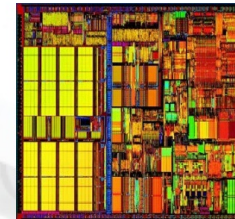
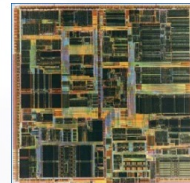
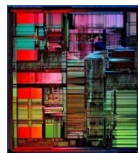
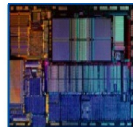
After 30 years



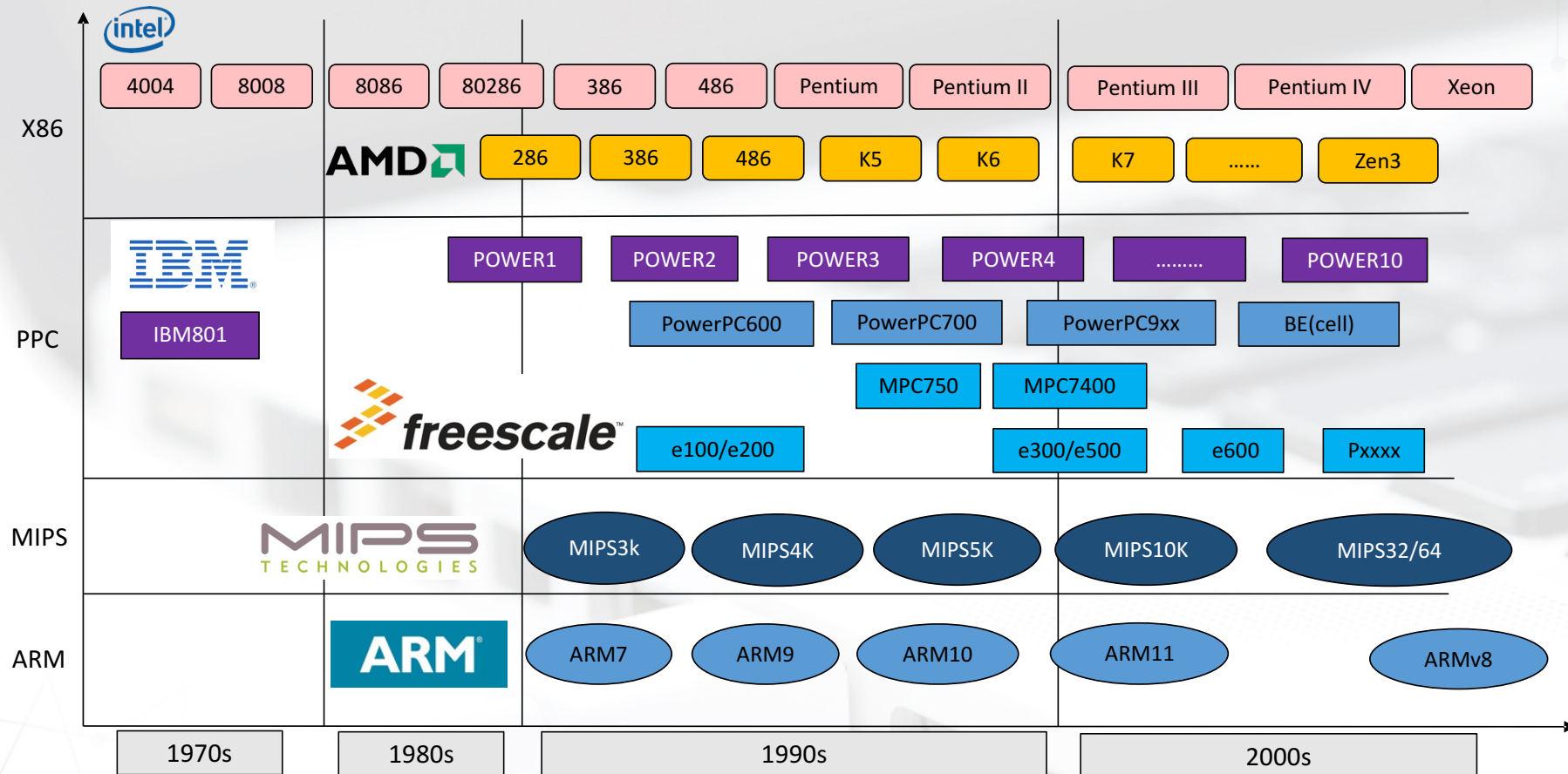
2000, Intel Pentium IV

- Issues up to 5 uOPs per cycle
- MMX, SSE, and SSE2
- 0.18 micron process
- **42 million transistors**
- 217 mm die
- 64-bit bus
- 8KB D-cache, 12KB op trace cache (I-cache), 256KB L2 cache
- **1.4 GHz**

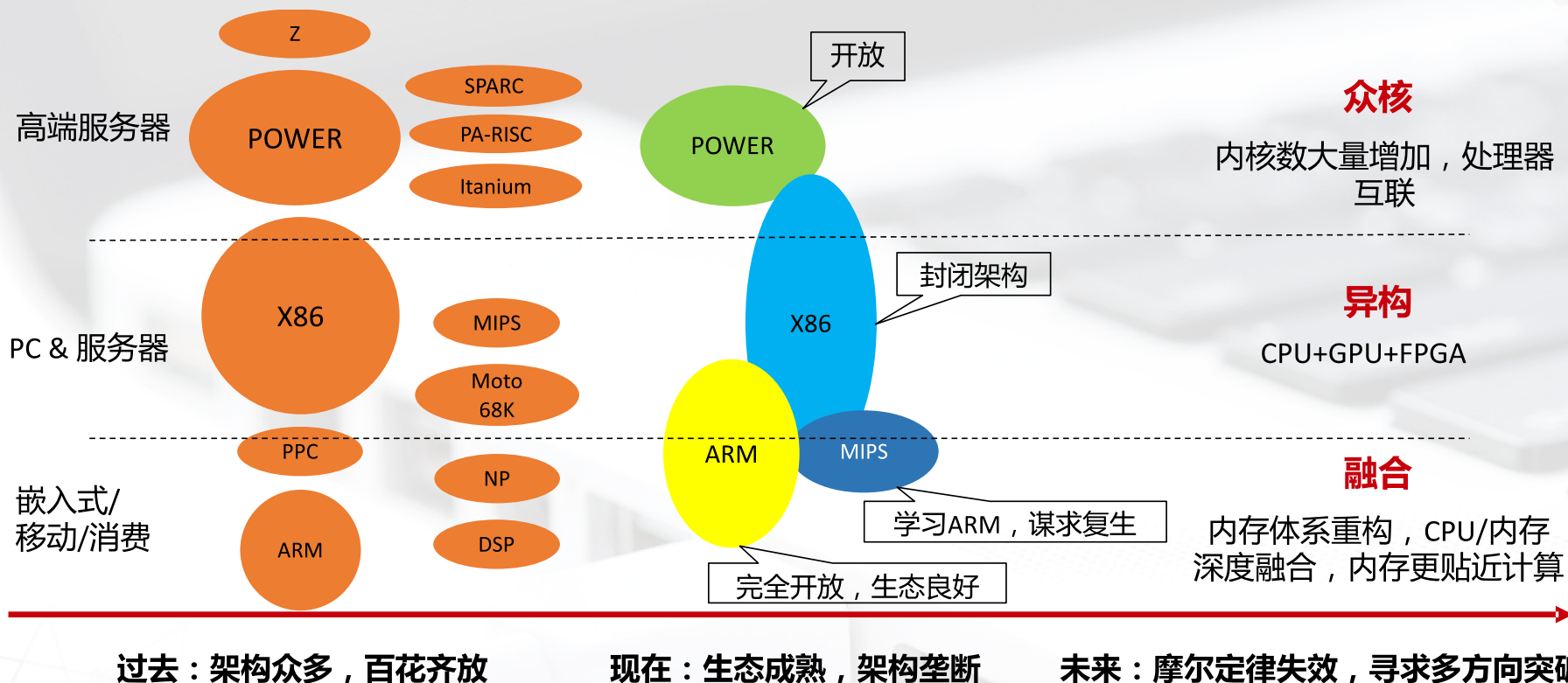
**Performance improved 5000x:
smaller, faster, cheaper**



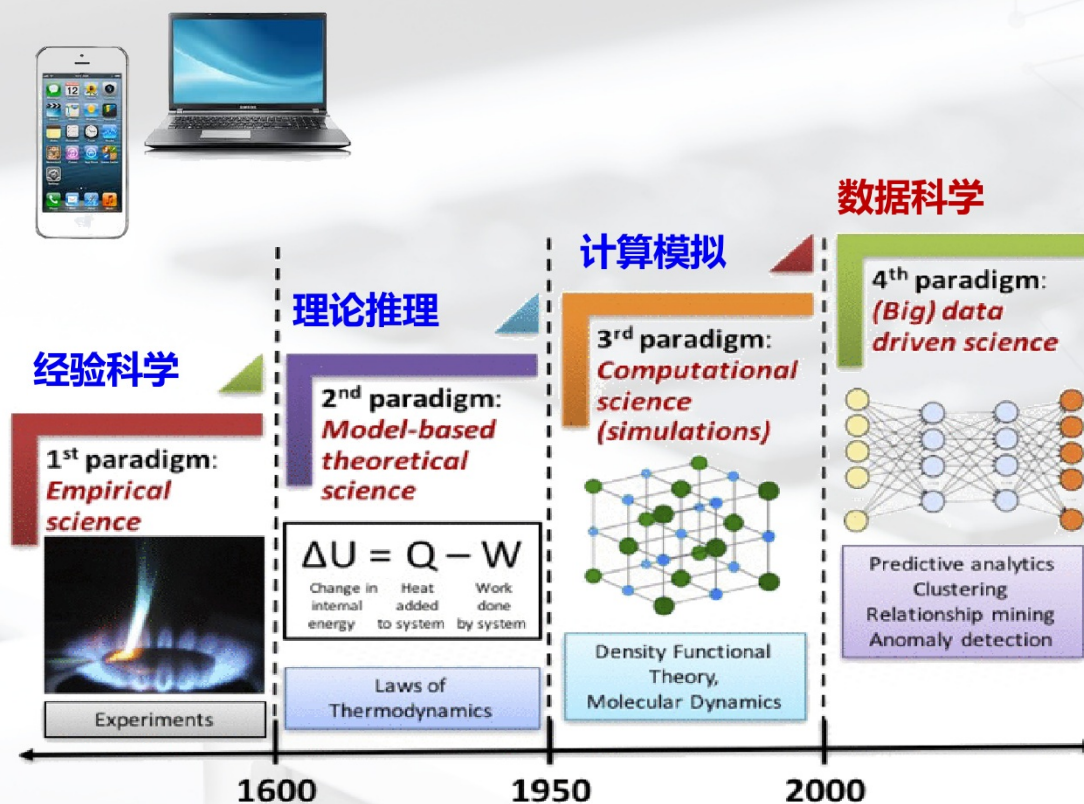
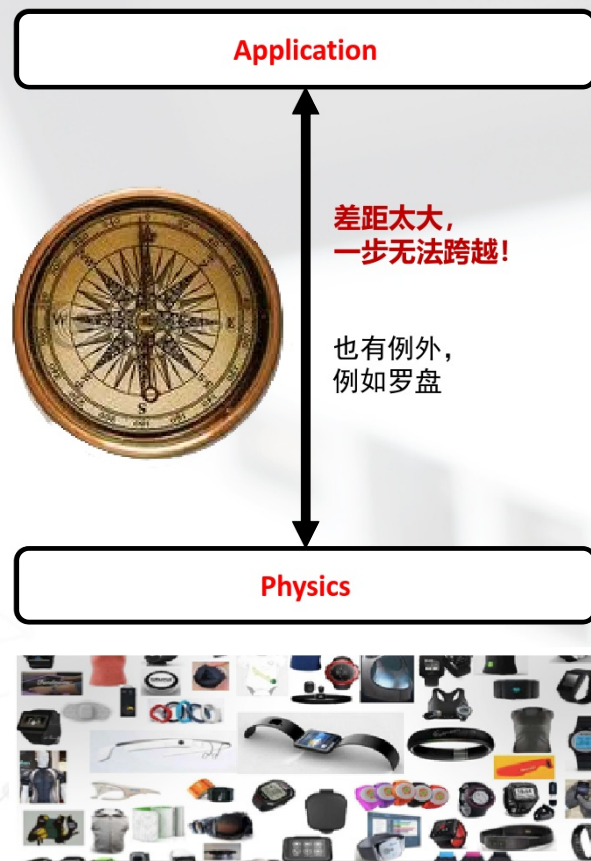
Thirty years after the first microprocessor chip was born



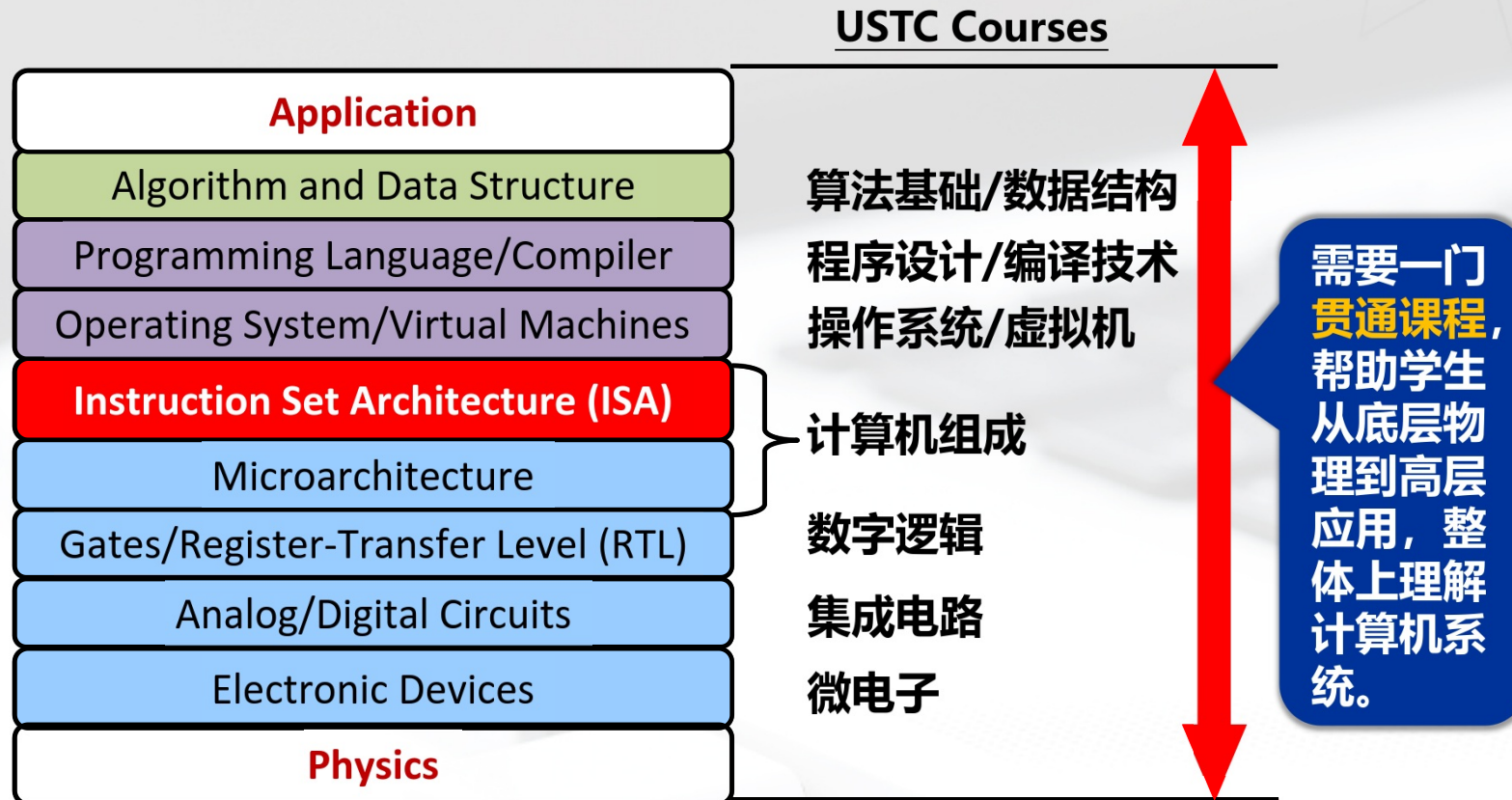
Thirty years after the first microprocessor chip was born



人类如何实现从物理设备到问题求解的?

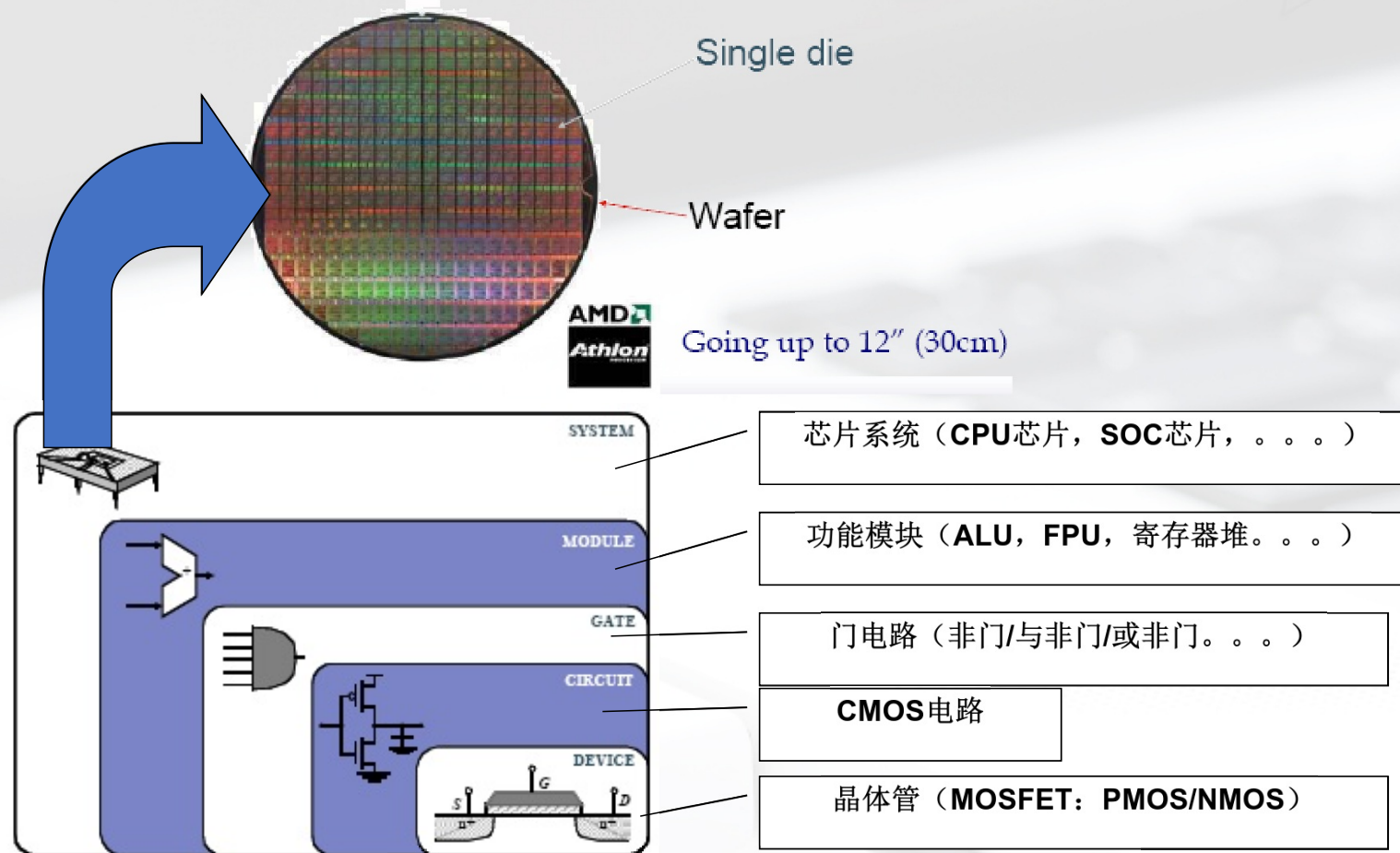


Great Idea #3: Abstraction helps us Manage Complexity



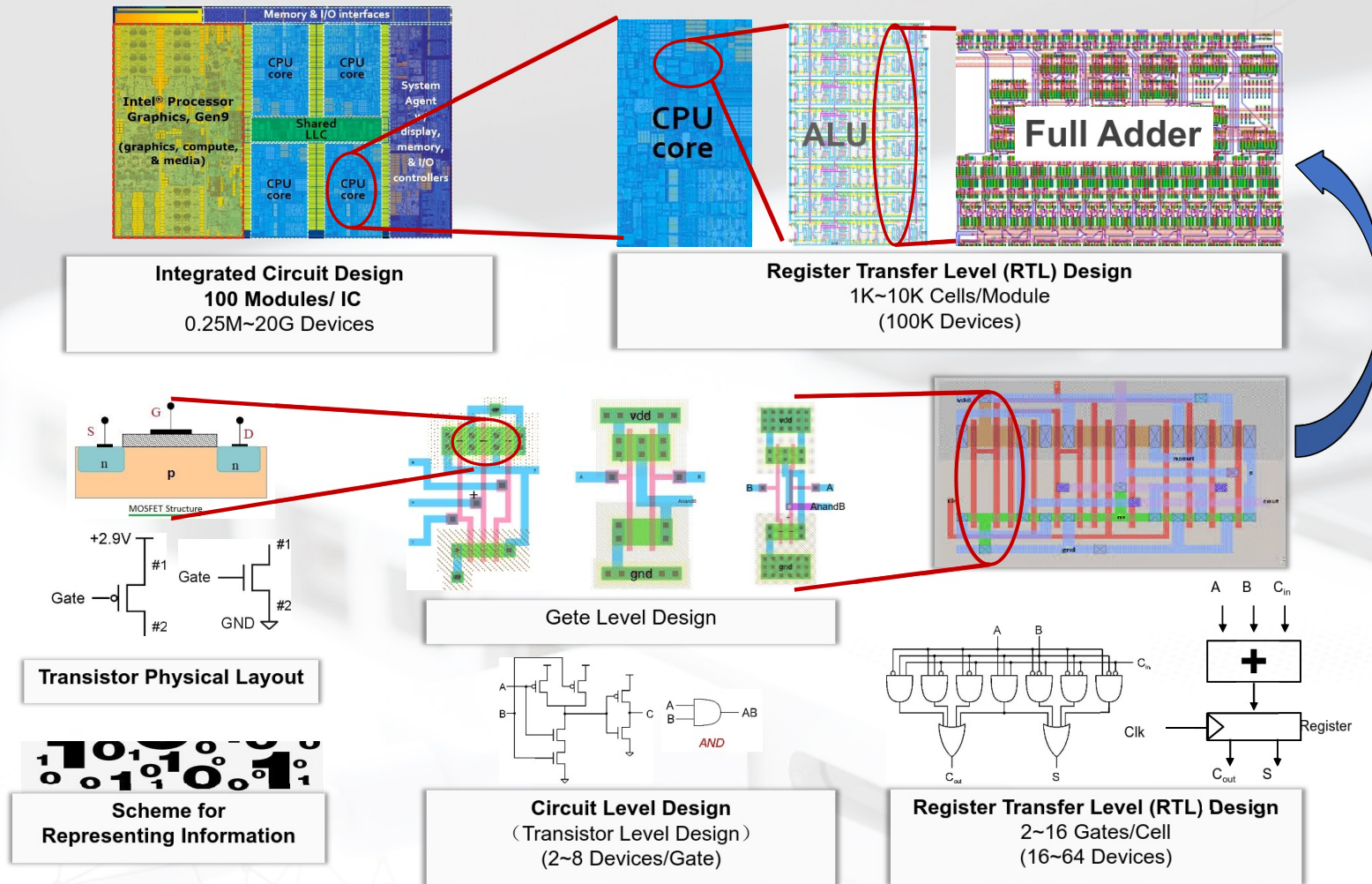
从广义上讲，**计算机系统结构**是抽象层次的设计，它允许我们使用可用的制造技术有效地实现信息处理**应用程序**。

Abstraction to Simplify Hardware Design



How do we put the devices into system?

Abstraction to Simplify System Design



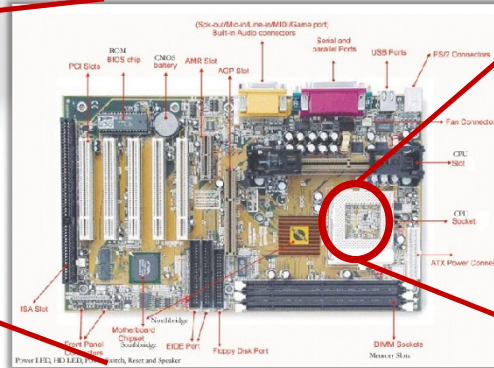
Abstraction to Simplify System Design



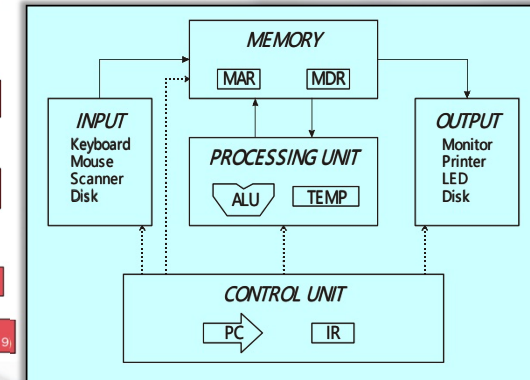
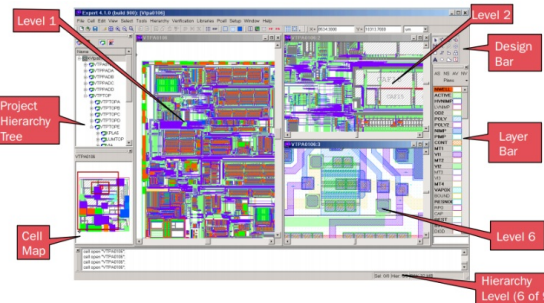
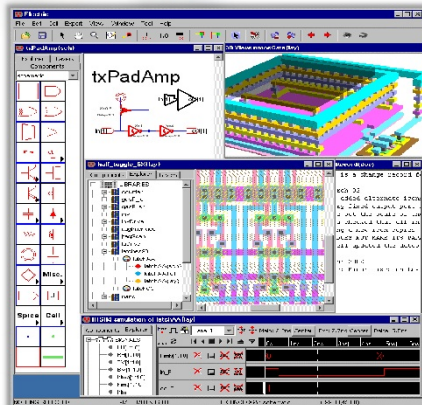
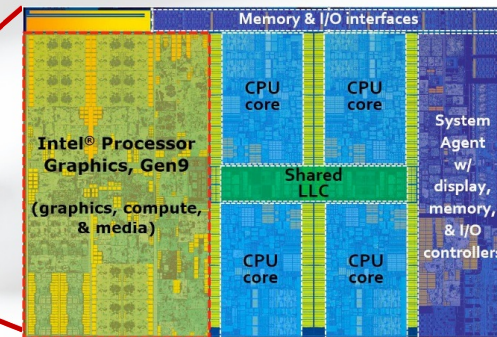
Personal Computer:
Hardware & Software Design
1~10PCBs/System



Motherboard Circuit Design
10 ICs/ PCB
1~50G Devices

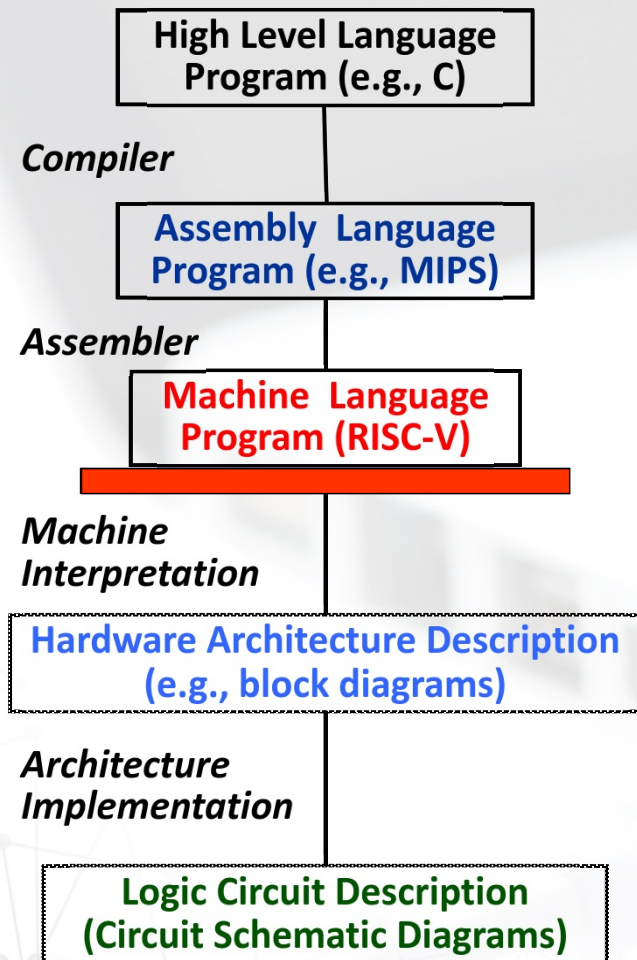


Integrated Circuit Design
100 Modules/ IC
0.25M~20G Devices



Electronic System Level (ESL) Design

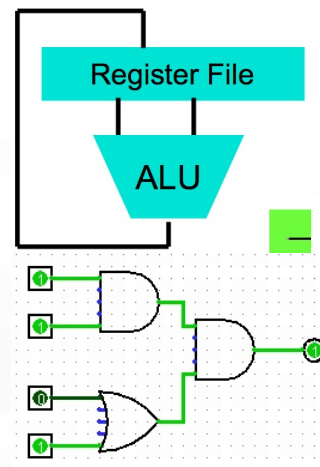
How do we get the electrons to do the work?



```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
```

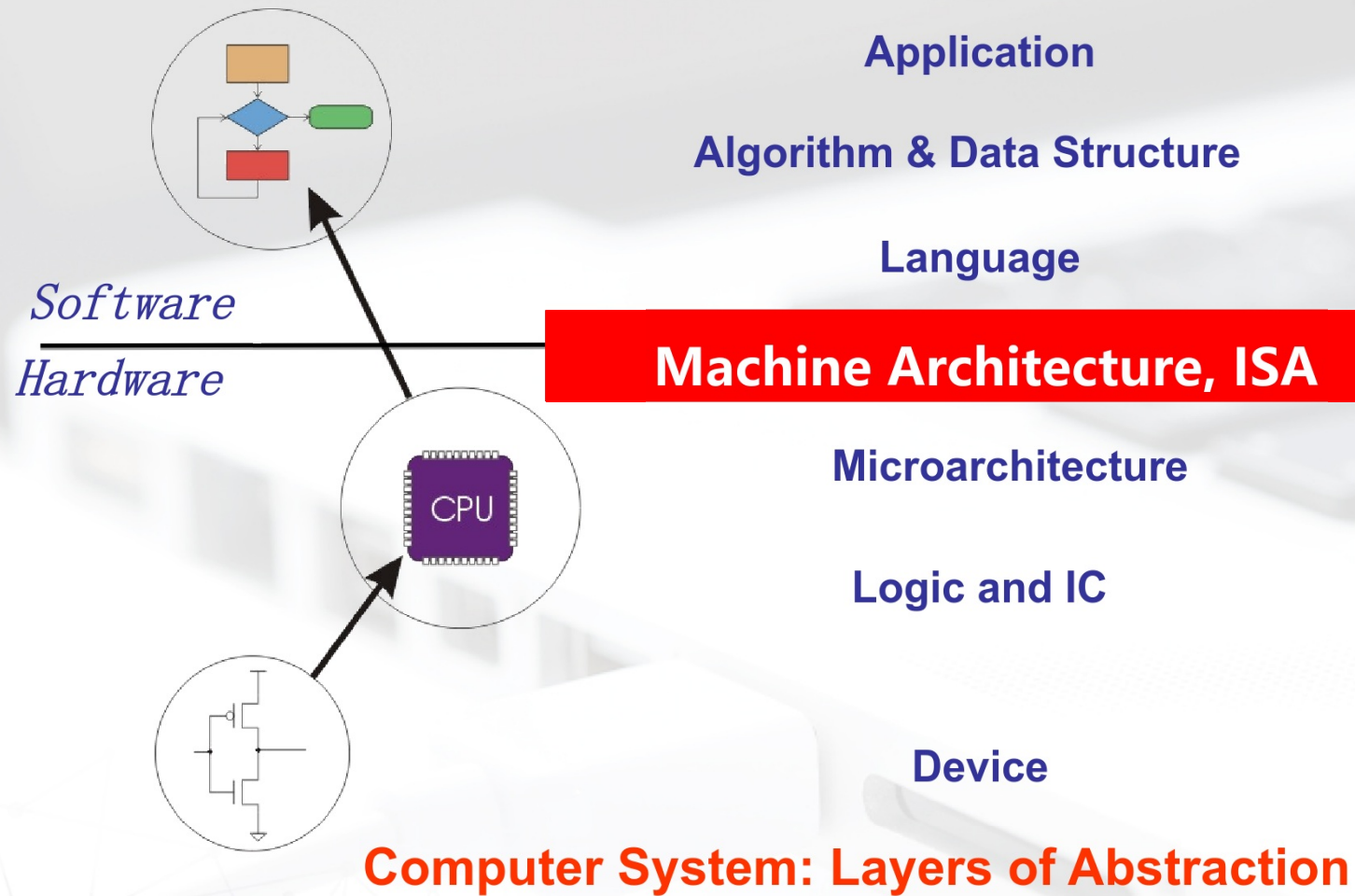
```
lw    $t0, 0($2)
lw    $t1, 4($2)
sw    $t1, 0($2)
sw    $t0, 4($2)
```

```
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111
```

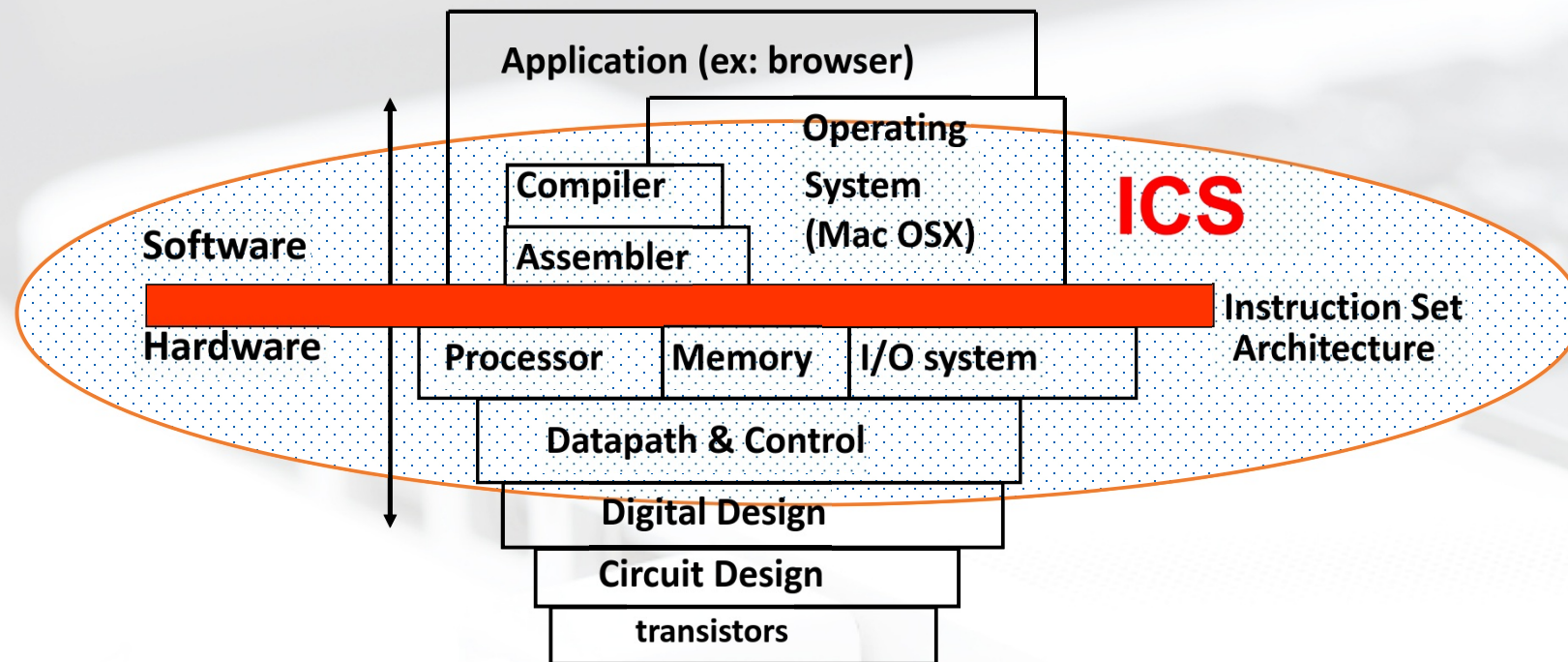


Anything can be represented
as a *number*,
i.e., data or instructions

Great Idea #4: Software and Hardware Co-design



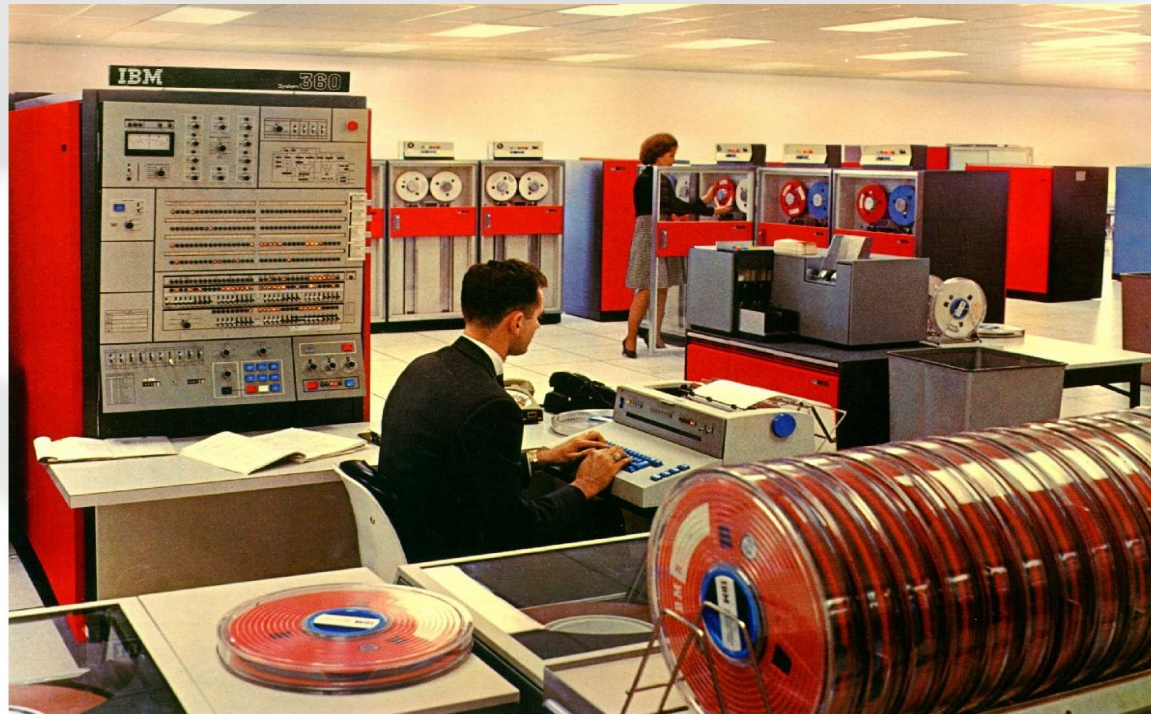
Old Machine Structures



Old Machine Structures



■ Mainframe: IBM System/360

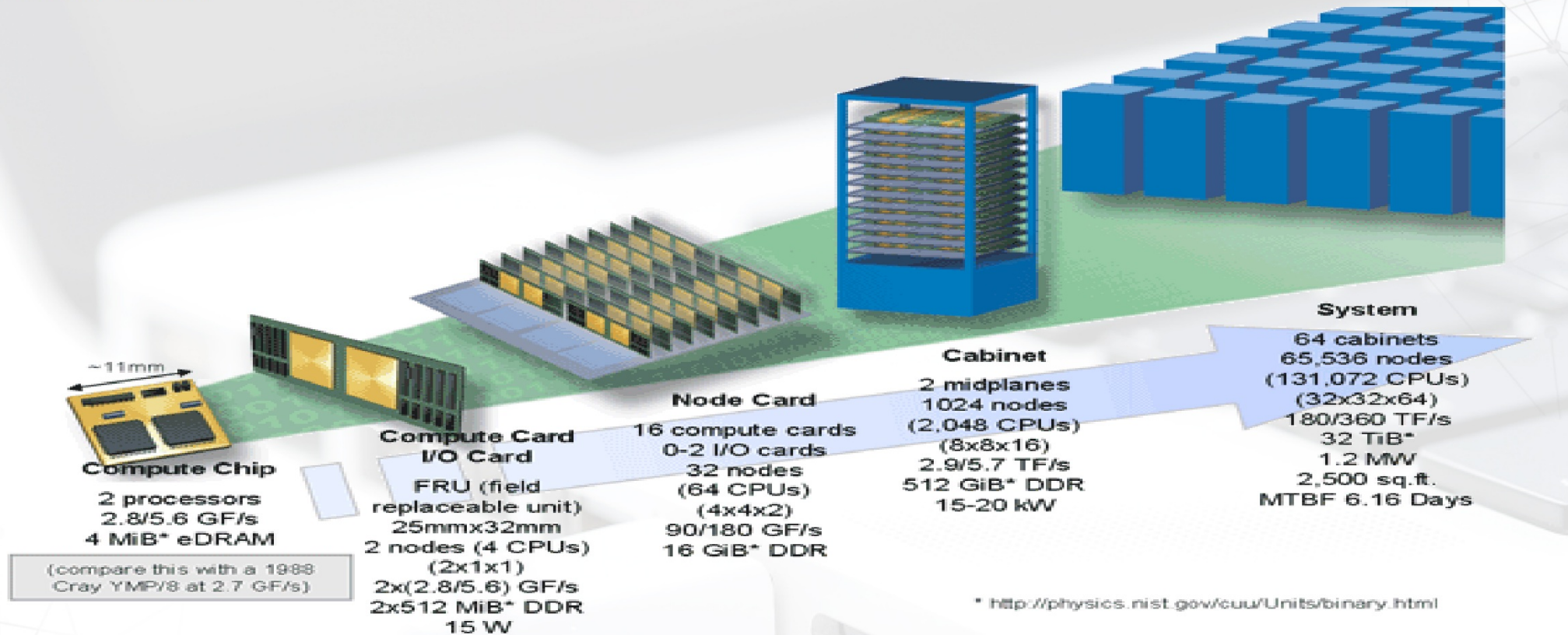


Computer Architecture: the structure of a computer that a machine language programmer must understand to write a correct (timing independent) program for that machine.

New Machine Structures



■ IBM Blue Gene



New Machine Structures: From the Gate to the Cloud



Software

Hardware

■ Parallel Requests

Assigned to computer
e.g., Search “Katz”

■ Parallel Threads

Assigned to core
e.g., Lookup Ads

■ Parallel

>1 instruct
e.g., 5 pipe

■ Parallel

>1 data item
e.g., Add o

■ Hardware Descriptions

All gates functioning in parallel at same time

■ Programming Languages

Warehouse
Scale Computer



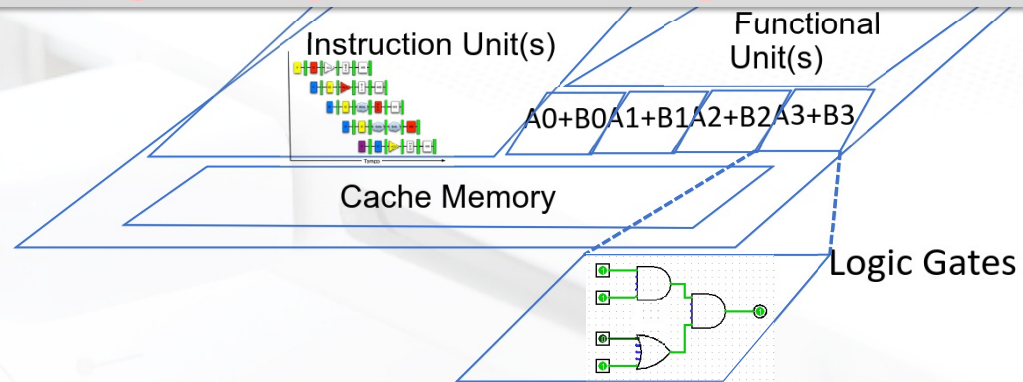
Smart
Phone



Leverage Parallelism &
Achieve High Performance

All computers are parallel!

**All computer engineers and scientists
require the knowledge of parallel computation**



Course Objectives



■ Exposure to...

- Machine organization
- Assembly language programming
- C programming

■ Understand how to build entire (slow) computing system

- Hardware and software
- You' ll get a chance in complementary courses

■ Be distinguished from mere programmers

- E.g. Matrix Multiply

计算机系统（晶体管器件、数字逻辑、组成原理、高级语言的编译与汇编、高级语言的硬件实现、操作系统）核心概念和思想的最小集

“Any sufficiently advanced technology is indistinguishable from magic.”

Arthur C. Clarke, "Profiles of The Future" (Clarke's 3rd law)

■ **No magic:** Computers should not be magic to computer scientists!

■ **Bottom UP:** Start with what they “know”

- Computing systems from transistors on up
- The transistor as light switch
- Not quantum mechanics

■ **Choose a computer model that is simple**

- Not about “**design**”, but about “**insight**” into all computers
- As the genius said: **simple**, but still **rich**
- Continually build on what you know
- Continually raising the level of abstraction
- **Memorizing as little as absolutely necessary**
- Trying very hard to not introduce magic

You take, You enjoy!!!

Outline



0 Course Crew

1 What' s the difference between “Big” and “Small” ?

2 Why Take This Course?

3 Great Ideas in Computing Systems

4 Course Implementation

Text Book



Assume already familiar with programming (C)

■ Part 1: Hardware (Chapter 1-5)

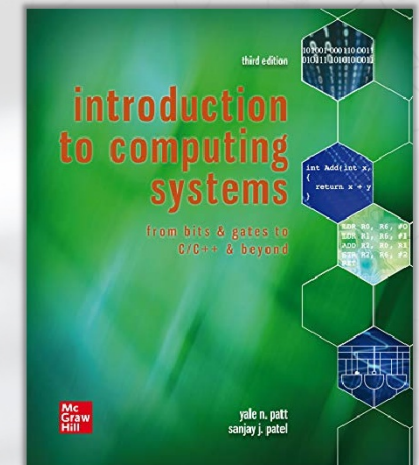
- Representing data, transistors, gates, digital logic structures
- von Neumann machine model
- *Instructions and its datapath*

■ Part 2: Software: Assembly language (Chapter 6-10) **No Chapter 6 & 10**

- (Structured) Programming
- Data structure
- Input/output
- *Software relationship to hardware*

■ Part 3: Software: C/C++ programming (Chapter 11-20), **No Chapter 15 & 20**

- Syntax, operators, control structures, functions, pointers, recursion, data structures
- *C/C++ relationship to assembly language*



Introduction to Computing Systems: from bits and gates to C/C++ and beyond (3rd edition),

Yale N. Patt and Sanjay J. Patel, June 2019, McGraw-Hill Higher Education

Background/Prerequisites



■ Requirement

- background in programming (C)
- Assume you can program/debug in C

This Course Focus on



■ Chapt 2 Bits, Data Types, and Operations

- How do we represent information using electrical signals?

■ Chapt 3 Digital Logic Structures

- How do we build circuits to process information?

■ Chapt 4, 5 Computer Machine Model, Processor and Instruction Set

- How do we build a processor out of logic elements?
- What operations (instructions) will we implement?

■ Chapt 6,7 Assembly Language Programming

- How do we use processor instructions to implement algorithms?
- How do we write modular, reusable code? (subroutines)

■ Chapt 8 Data Structures

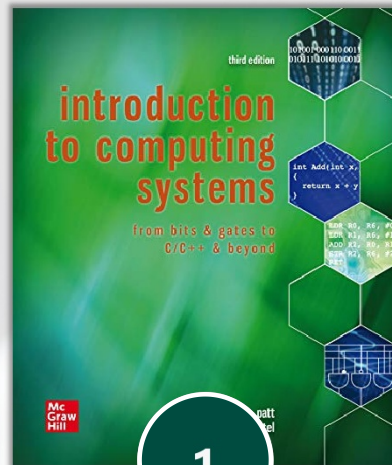
■ Chapt 9 I/O, Traps, and Interrupts

- How does processor communicate with outside world?

■ Chapt 10 Put It All Together: A Calculator

■ Chapt 11, C Implementation Related to Hardware

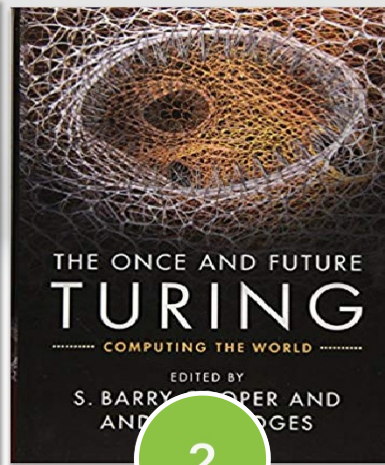
Other Reference Text Books



1

Introduction to Computing Systems: from bits and gates to C/C++ and beyond(3rd edition),

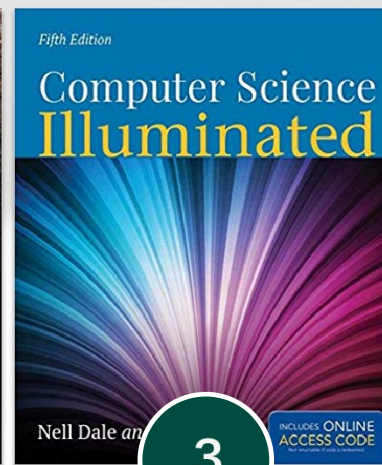
Yale N. Patt and Sanjay J. Patel, June 2019, McGraw-Hill Higher Education



2

The Once and Future Turing: Computing the World (1st Edition) ,

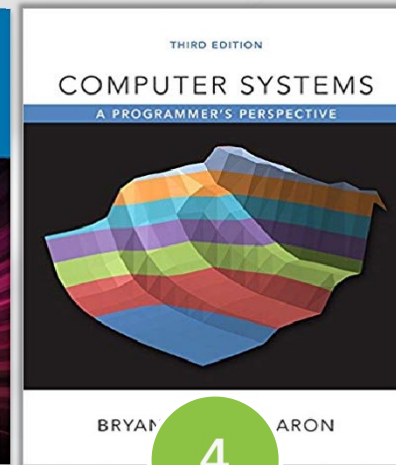
S. Barry Cooper, Andrew Hodges, Cambridge University Press, 2016



3

Computer Science Illuminated(5th Edition) ,

D. M. Harris, S. L. Harris, Morgan Kaufmann, San Francisco, 2007

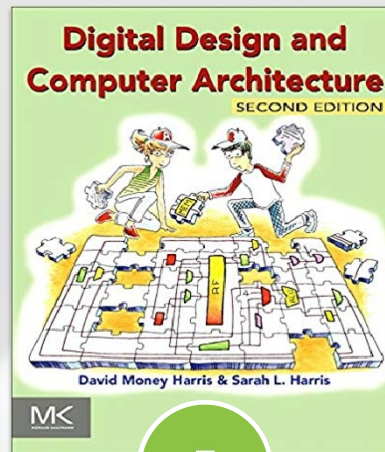


4

Computer Systems: A Programmer's Perspective (3rd Edition) ,

Randal E. Bryant, David R. O' Hallaron, Pearson Education Inc., 2016

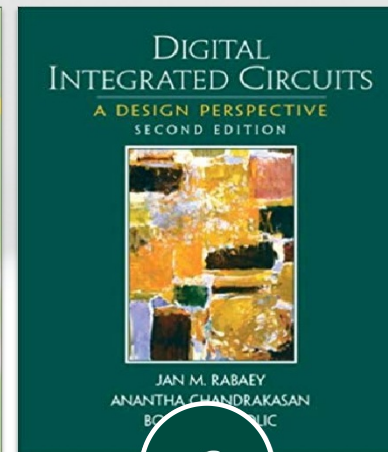
Other Reference Text Books



5

Digital Design and Computer Architecture (2nd Edition) ,

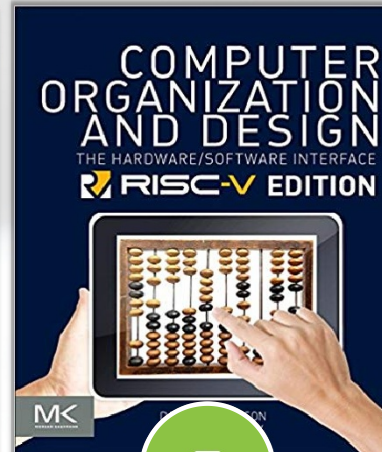
David Harris Sarah Harris, p712, Morgan Kaufmann, 24th July 2012



6

Digital Integrated Circuits: A Design Perspective (2nd Edition),

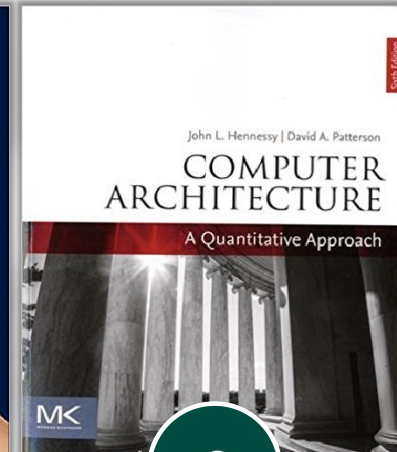
Jan M. Rabaey, Anantha Chandrakasan, Borivoje, Prentice-Hall, Inc, Nikolic, Jan 3, 2003



7

Computer Organization and Design: The Hardware/Software Interface,

David A Patterson, John L. Hennessy, 5th edition. Morgan Kaufmann Publishers, Inc., 2017

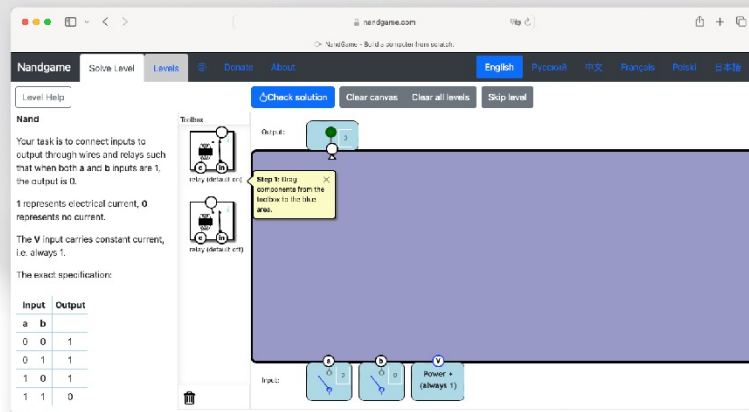


8

Computer Architecture: A Quantitative Approach,

John L. Hennessy and David A. Patterson, The Morgan Kaufmann, Dec 7, 2017

Game



NandGame
<https://www.nandgame.com>



图灵完备

Class Organization in 2011



■ 2011年,暑期小学期

- 全校160人选修,10级计算机英才班必修
- 主讲: **Yale Patt**, 中国科大客座教授, 安虹教授
- 助教: 张海博, 王涛, 汤旭龙, 吴石磊, 孙公瑾, 孙荪, 冷鹏, 魏学超



2026-9-3



105

Class Organization



- Office hours

- Problems Sets: ~8

- Labs: 6~7

- All labs start after 4 weeks

- A Calculator Design or **LC-3 Simulator/Assembler Design**

- 1 Midterm + 1 Final

Homework Assignments



■ Problem Sets: 6 sets

- Problem solving
- Complete *before* each due date
- Can work ahead
- Great exam preparation!

Labs Assignments



■ Simple Programming Assignments

- Assignments1: Programming In machine language
- Assignments2~5: Programming In LC3 assembly language that cover all LC-3 instruction.
- Assignments6: Programming In C
- Assignments7: **Design**
 - A Calculator Design
 - LC-3 Simulator/Assembler Design (Encouraged! Extra 5 points)

■ See schedule for each lab due dates

Exams & Grades



■ Middle Exam: 15%

■ Final Exam: 30%

■ Assignments : 50%

- Problem Sets for every chapter
- 6 Programming Assignments
- LC3 Simulator/Assembler Design (Honor class required)

■ EPA: Effort, Participation and Altruism in discussion section: 5%

- If you do not attend discussion, you can not participate Exam; 5% → 0%
- If you don't do the problems, you can not participate well; 5% → 0%

Policy on Assignments and Independent Work



■ ALL PROJECTS WILL BE DONE AND SUBMITTED **INDIVIDUALLY**

- With the exception of laboratories and assignments that explicitly permit you to work in groups, all homework and projects are to be YOUR work and your work ALONE.

■ You are encouraged to **discuss** your assignments with other students, and extra credit will be assigned to students who help others, particularly by **answering questions on Piazza**, but we expect that what you hand in is yours.

■ It is **NOT** acceptable to **copy** (or even "start with") solutions from other students or the Web

Tips on How to Get a Good Grade



- The lecture material is not the most challenging part of the course. You should be able to **understand everything** as we go along.
- **DO NOT fall behind** in lecture and tell yourself you “will figure it out later from the notes or books” .
- PPTs will be online after the lecture (usually the night). Do assigned reading before the lecture.
- **Ask questions in class** and stay involved in the class - that will help you understand.
- Come to office hours to check your understanding or to ask questions.
- **Complete all the homework problems** - even the difficult ones. The exams will test your depth of knowledge.

Tips on How to Get a Good Grade



■ You need to understand the material well enough to apply it in new situations. You need to **enroll in both the lab and the course.**

- Take the labs very seriously. They are an integral part of the course.
- Choose your partner carefully. Your best friend may not be the best choice!
- Most important : Be well organized and neat with homework, labs, project. In lab, add complexity a little bit at a time - always have a working design.



Do not post your work on public repositories like github (private o.k.) --negative points

■The rule is simple

- Claiming another's work as your own *will ruin your life*
- See syllabus for details and examples

■Who will know?

- We will (inspection, similarity detectors, exams)
- Your friends will... your parents will...
- You will

**NO AI to complete your
all assignments and experiments!**

■Remember

- If you need to cheat now, you've got much bigger problems
- **Cheating is like going 150 MPH over speed limit while drunk!**

Acknowledgements



■ This course is partly inspired by previous MIT 6.823 and Berkeley CS252 computer architecture courses created by my collaborators and colleagues:

- Yale Patt (University of Texas at Austin)
- Arvind (MIT)
- Joel Emer (Intel/MIT)
- James Hoe (CMU)
- John Kubiatowicz (UCB)
- David Patterson (UCB)
- Krste Asanovic (UCB)