计算机组成原理
CH1_计算机抽象及相关技术

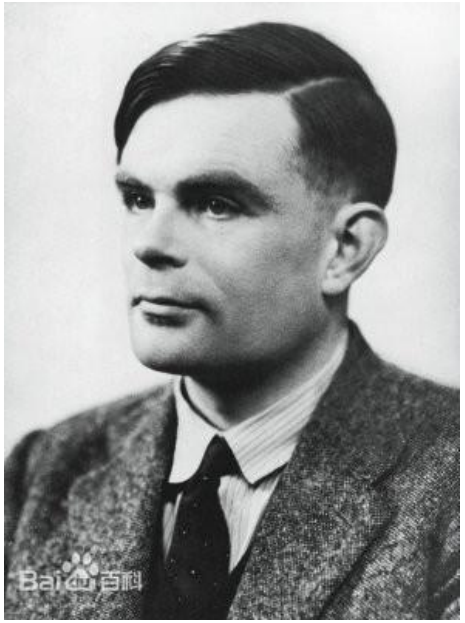
卢建良
lujl@ustc.edu.cn
2022年春季学期

提纲

- 引言
- 计算机体系结构中的8个伟大思想
- 程序表象之下
- 箱盖后的硬件
- 处理器和存储制造技术
- 性能
- 功耗墙
- 沧海巨变：从单处理器向多处理器转变
- 实例：测评Intel Core i7
- 谬误与陷阱
- 本章小结

1.1 引言

- 图灵：战争英雄，同性恋、计算机科学之父
  - 艾伦·麦席森·图灵（英语：Alan Mathison Turing，1912年6月23日—1954年6月7日），英国数学家、逻辑学家，被称为计算机科学之父，人工智能之父。1931年图灵进入剑桥大学国王学院，毕业后到美国普林斯顿大学攻读博士学位，第二次世界大战爆发后回到剑桥，后曾协助军方破解德国的著名密码系统Enigma，帮助盟军取得了二战的胜利。
  - 1952年，英国政府对图灵的同性恋取向定罪，随后图灵接受化学阉割（雌激素注射）。1954年6月7日，图灵吃下含有氰化物的**苹果**中毒身亡，享年41岁。2013年12月24日，在英国司法大臣克里斯·格雷灵的要求下，英国女王伊丽莎白二世向图灵颁发了皇家赦免。
  - 图灵对于人工智能的发展有诸多贡献，提出了一种用于判定机器是否具有智能的试验方法，即图灵试验，每年都有试验的比赛。此外，图灵提出的著名的图灵机模型为现代计算机的逻辑工作方式奠定了基础。
  - https://www.bilibili.com/video/BV1tx411V7yQ/?spm_id_from=333.788.recommend_more_video.3



1.1 引言

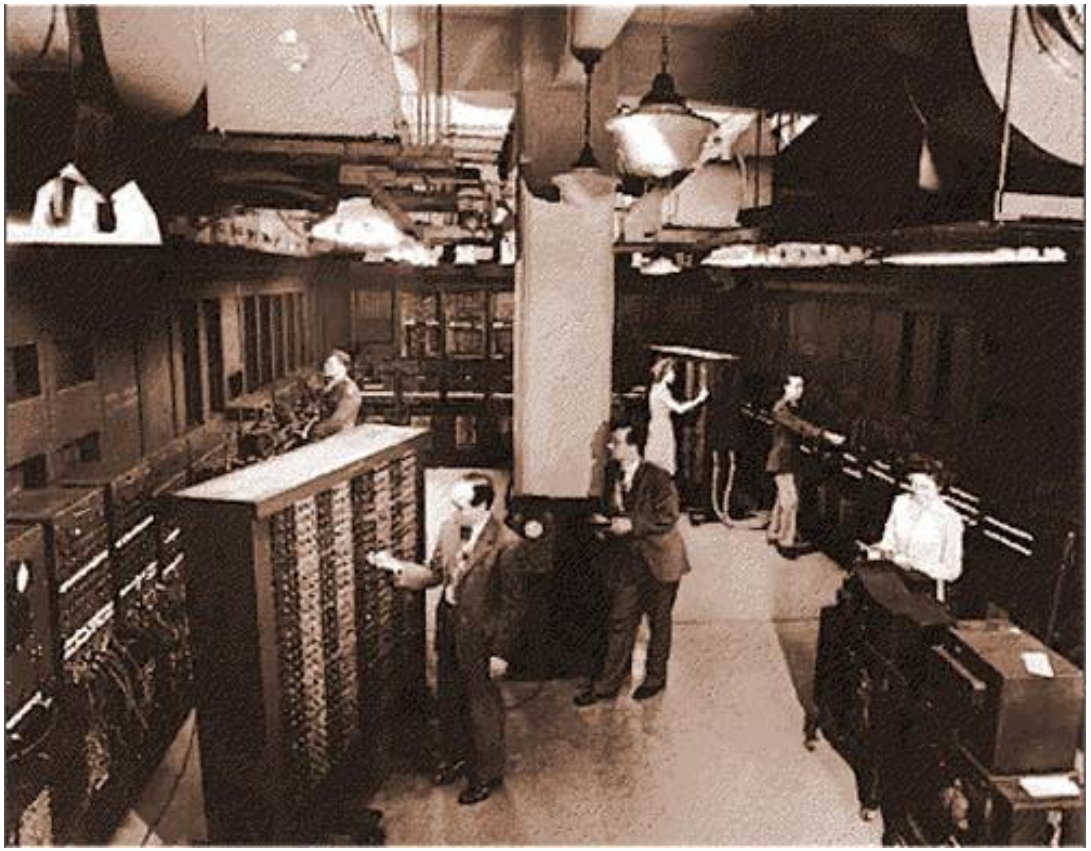
- 图灵机
  - 图灵机（Turing Machine）是图灵在1936年发表的 "On Computable Numbers, with an Application to the Entscheidungsproblem"（《论可计算数及其在判定性问题上的应用》）中提出的**数学模型**。既然是数学模型，它就并非一个实体概念，而是架空的一个想法。在文章中图灵描述了它是什么，并且证明了，**只要图灵机可以被实现，就可以用来解决任何可计算问题**
- 图灵完备
  - 图灵完备性（Turing Completeness）是针对一套数据操作规则而言的概念。数据操作规则可以是一门编程语言，也可以是计算机里具体实现了的指令集。当这套规则可以实现图灵机模型里的全部功能时，就称它具有图灵完备性。直白一点说，图灵完备性就是我给你一工具箱的东西，包括无限内存、if/else 控制流、while 循环……那么你现在图灵完备了吗?
- https://blog.csdn.net/a493823882/article/details/109149332
- https://www.bilibili.com/video/BV1br4y1N762/?spm_id_from=333.788.recommend_more_video.5

1.1 引言

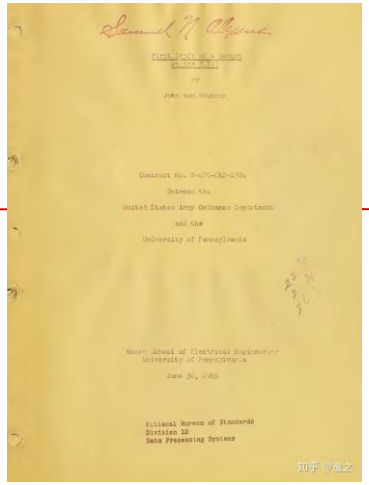
- ABC ,1942年，第一台电子计算机
  - Atanasoff–Berry computer
  - 阿塔纳索夫-贝里计算机
  - 爱荷华州立大学
  - 用于求解线性方程组
  - 最多支持29个方程
  - 非图灵完备

1.1 引言

- ENIAC ,1946年，第一台通用电子计算机
  - Electronic Numerical Integrator and Computer
  - 电子数字积分器和计算机
  - 宾夕法尼亚大学莫尔学院
  - 成本：100万美元
  - 功耗：150kw/h
  - 占地：170m^2
  - 运算速度：5000次/秒
  - 不可编程
  - 十进制并行计算
  - 无程序存储功能



1.1 引言

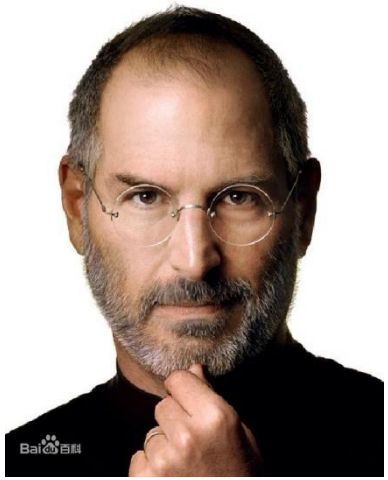


- EDVAC,1944~1952,
  - Electronic Discrete Variable Automatic Computer
  - 电子离散变量自动计算机
  - 1MHz
  - 二进制
  - 串行
  - 存储程序
  - 冯·诺依曼架构
  - EDVAC报告书的第一份草案

- ✓ 机器内部使用二进制表示数据；
- ✓ 像存储数据一样存储程序；
- ✓ 计算机由运算器、控制器、存储器、输入模块和输出模块5部分组成

1.1 引言

- iphone13 pro max,2021
  - 主频：3GHz
  - 尺寸：16cm*8cm*0.8cm
  - 功耗：1~4W
  - 成本：2000美元
  - 性能：每秒15.8万亿次计算
  - 集成度：150亿晶体管



Apple iPhone 13 Pro Max (A2644) 1TB 远峰蓝色 支持移动联通电信5G 双卡双待手机

自适应高刷新率，画面更流畅、响应更灵敏，电影效果模式随手拍大片！选购[快充套装]加99元得20W快充头！更多

京东价 ¥12999.00 降价通知

促销 赠品 ×1 ×1 ×1

¥9799.00
Apple iPhone 13 Pro Max (A2644) 256GB 远峰蓝色 支持移动联通电信5G 双卡双待

¥11399.00
Apple iPhone 13 Pro Max (A2644) 512GB 远峰蓝色 支持移动联通电信5G 双卡双待

¥219.00
京东超市 闪迪(SanDisk)256GB USB3.0 U盘 CZ73酷铄 银色 读速150MB/s 金属外

1.1 引言

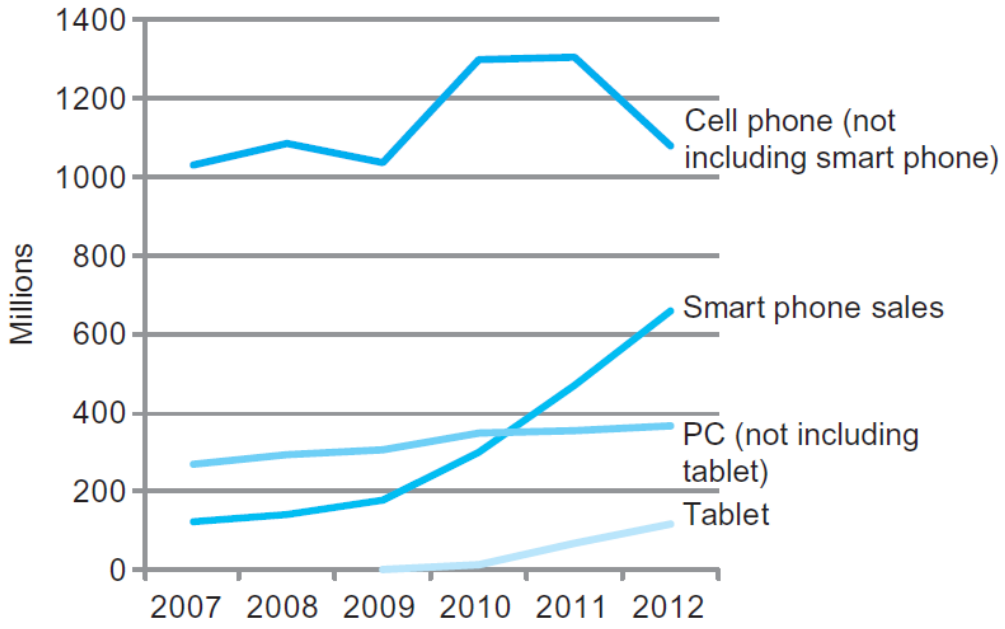
- 常用规格术语
  - 二进制 vs 十进制

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

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

1.1 引言

- 传统计算机分类
  - 个人计算机（PC）
  - 服务器
  - 嵌入式计算机
- 后PC时代
  - 个人移动设备
  - 云计算
    - 软件即服务 SaaS
    - Software as a Service



1.1 引言

- 您从本课程中学到什么
  - How programs are translated into the machine language
    - And how the hardware executes them
  - The hardware/software interface
  - What determines program performance
    - And how it can be improved
  - How hardware designers improve performance
  - What is parallel processing

1.1 引言

- Understanding Performance

Hardware or software component	How this component affects performance	Where is this topic covered?
Algorithm	Determines both the number of source-level statements and the number of I/O operations executed	Other books!
Programming language, compiler, and architecture	Determines the number of computer instructions for each source-level statement	Chapters 2 and 3
Processor and memory system	Determines how fast instructions can be executed	Chapters 4, 5, and 6
I/O system (hardware and operating system)	Determines how fast I/O operations may be executed	Chapters 4, 5, and 6

1.1 引言

Check Yourself

Check Yourself sections are designed to help readers assess whether they comprehend the major concepts introduced in a chapter and understand the implications of those concepts. Some *Check Yourself* questions have simple answers; others are for discussion among a group. Answers to the specific questions can be found at the end of the chapter. *Check Yourself* questions appear only at the end of a section, making it easy to skip them if you are sure you understand the material.

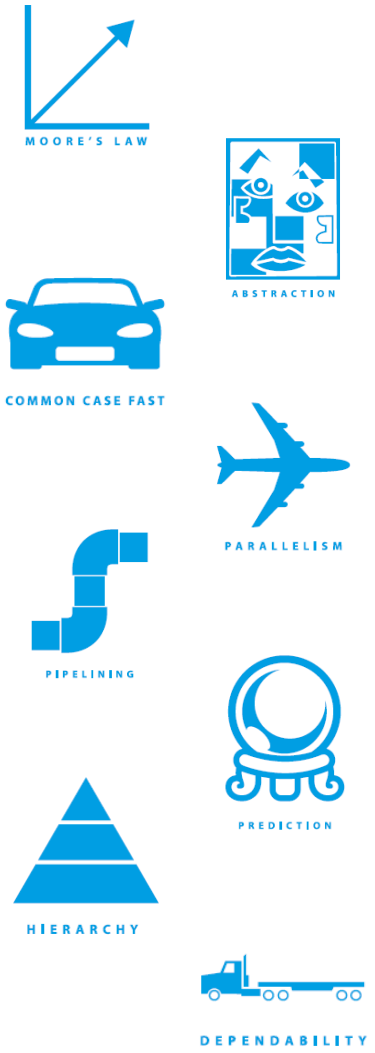
1. The number of embedded processors sold every year greatly outnumbers the number of PC and even post-PC processors. Can you confirm or deny this insight based on your own experience? Try to count the number of embedded processors in your home. How does it compare with the number of conventional computers in your home?

2. As mentioned earlier, both the software and hardware affect the performance of a program. Can you think of examples where each of the following is the right place to look for a performance bottleneck?

   - The algorithm chosen
   - The programming language or compiler
   - The operating system
   - The processor
   - The I/O system and devices

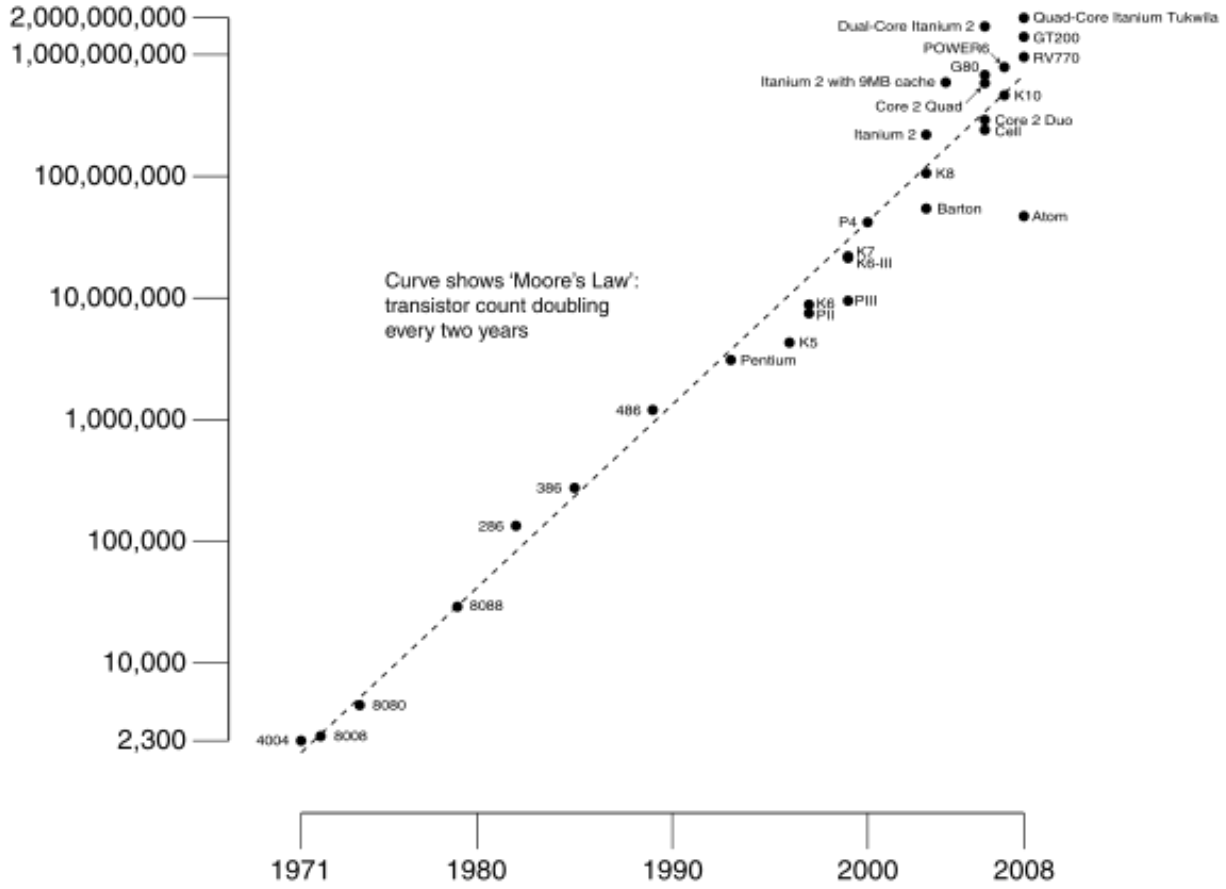
1.2 计算机体系结构中的8个伟大思想

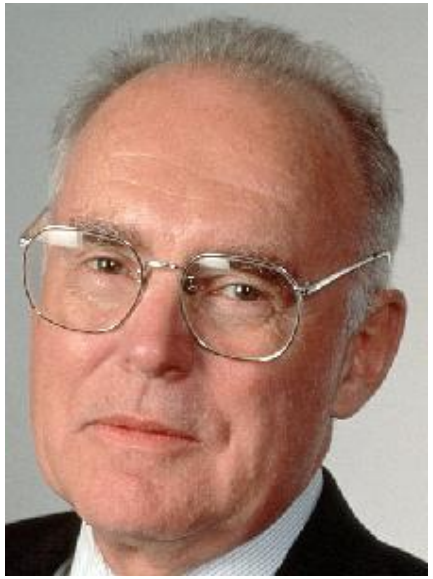
- Design for ***Moore's Law***
- Use ***abstraction*** to simplify design
- Make the ***common case fast***
- Performance *via* ***parallelism***
- Performance *via* ***pipelining***
- Performance *via* ***prediction***
- ***Hierarchy*** of memories
- ***Dependability*** *via* redundancy



1.2 计算机体系结构中的8个伟大思想

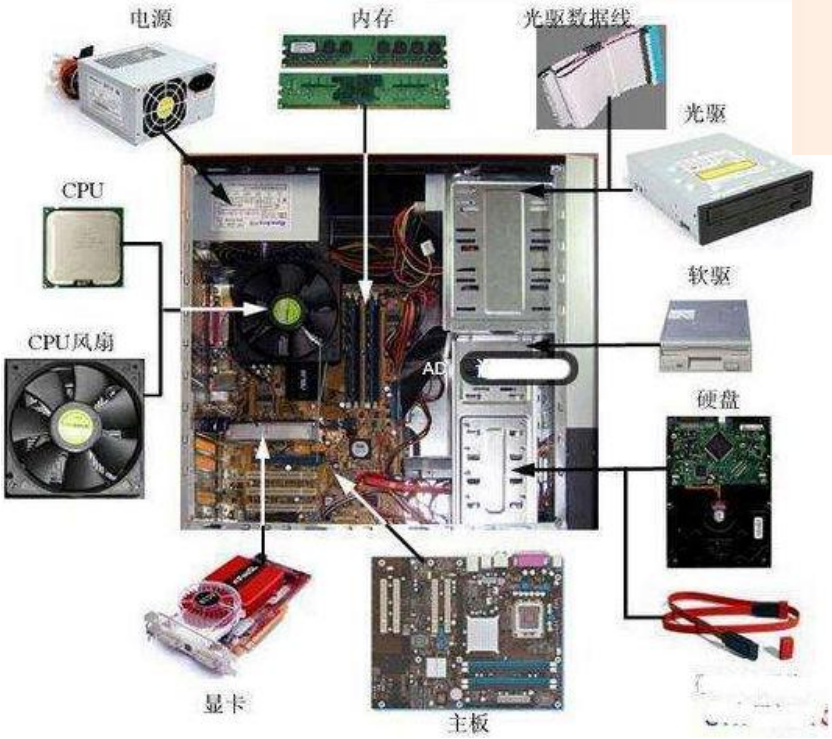
- 1 面向摩尔定律
  - 英特尔创始人之一戈登·摩尔的经验之谈，其核心内容为：集成电路上可以容纳的晶体管数目在大约每经过18个月便会增加一倍

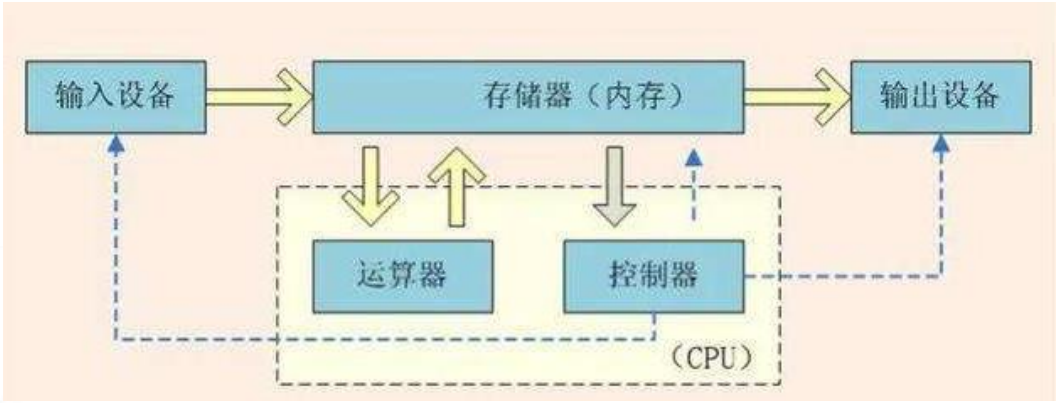




1.2 计算机体系结构中的8个伟大思想

- 2 使用抽象简化设计
  - 隐藏低层次细节，为高层次提供更简单的模型





1.2 计算机体系结构中的8个伟大思想

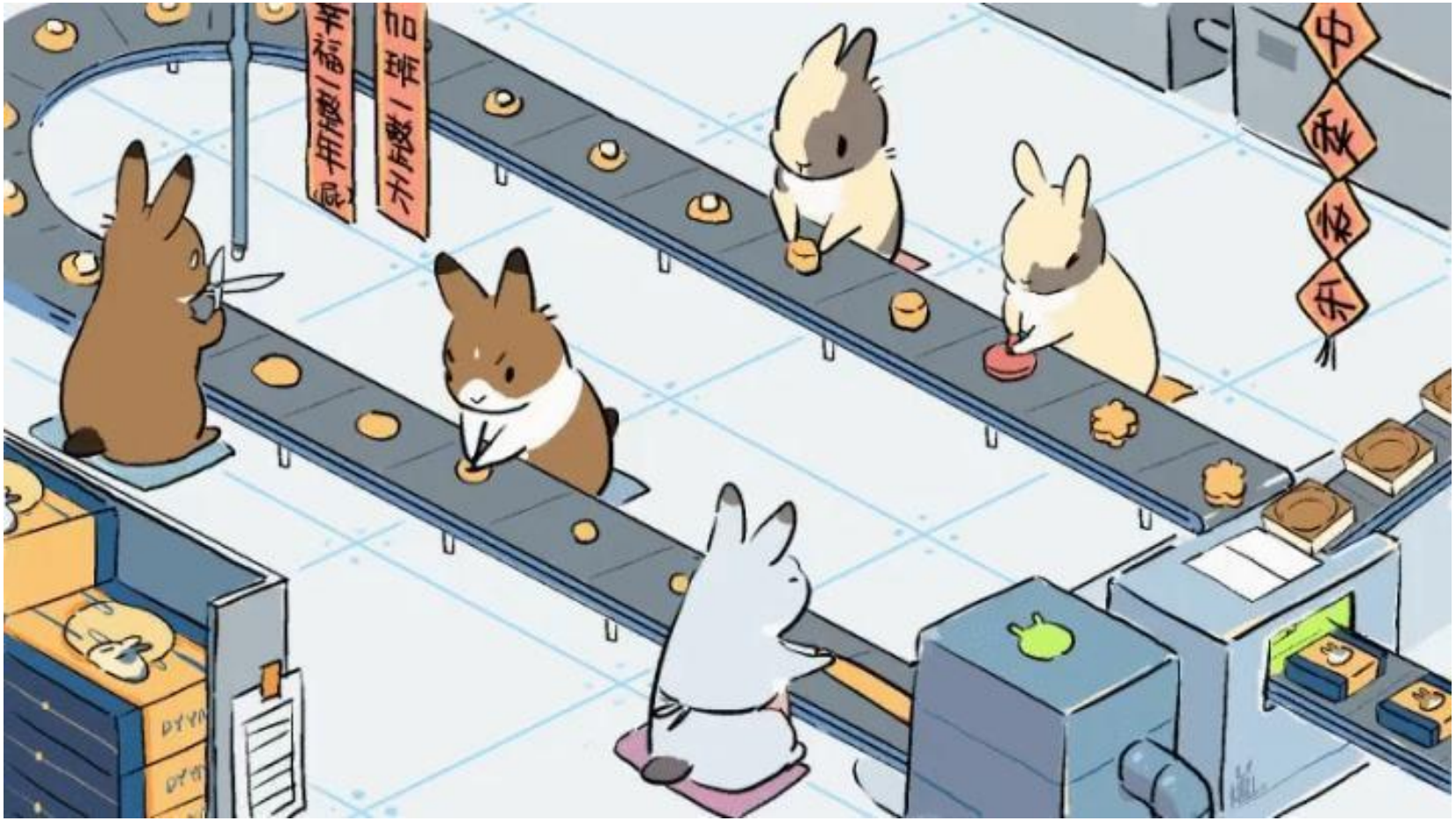
- 3 加速经常性事件
  - 通过实验及测量确定经常性事件
  - 宰我问：“三年之丧，期已久矣！君子三年不为礼，礼必坏；三年不为乐，乐必崩。旧谷既没，新谷既升，钻燧改火，期可已矣。”子曰：“食夫稻，衣夫锦，于女安乎？”曰：“安！”“女安则为之！夫君子之居丧，食旨不甘，闻乐不乐，居处不安，故不为也。今女安，则为之！”
  - 宰我出，子曰：“予之不仁也！子生三年，然后免于父母之怀。夫三年之丧，天下之通丧也，予也有三年之爱于其父母乎！”

1.2 计算机体系结构中的8个伟大思想

- 4 通过并行提高性能

1.2 计算机体系结构中的8个伟大思想

- 5 通过流水线提高性能



1.2 计算机体系结构中的8个伟大思想

- 6 通过预测提高性能

1.2 计算机体系结构中的8个伟大思想

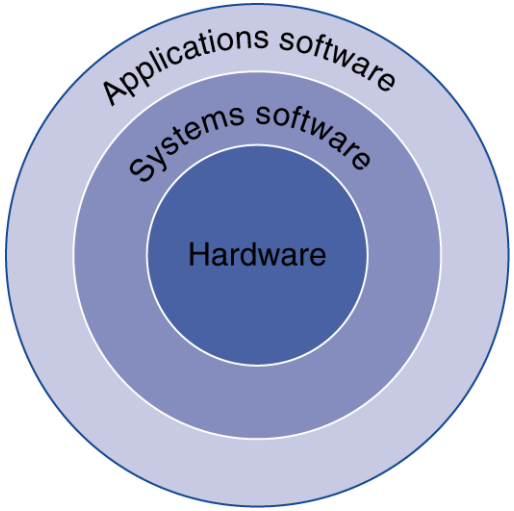
- 7 存储层次

1.2 计算机体系结构中的8个伟大思想

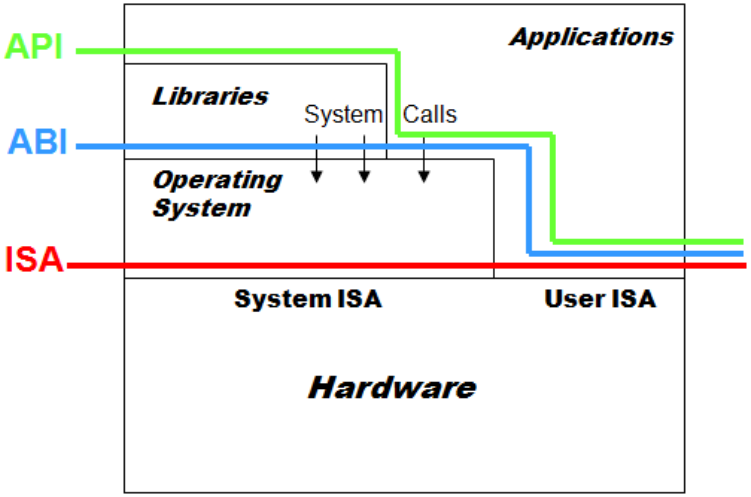
- 8 通过冗余提高可靠性

1.3 程序表象之下

- 应用软件
- 系统软件
- 硬件



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



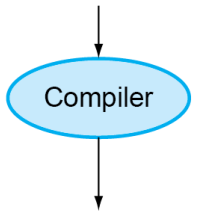
- API – application programming interface
- ABI – application binary interface
- ISA – instruction set architecture

1.3 程序表象之下

- 从高级语言到硬件语言
  - High-level language
    - Level of abstraction closer to problem domain
    - Provides for productivity and portability
  - Assembly language
    - Textual representation of instructions
  - Hardware representation
    - Binary digits (bits)
    - Encoded instructions and data

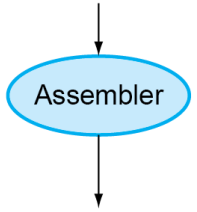
High-level language program (in C)

```
swap(int v[], int k)
{int temp;
   temp = v[k];
   v[k] = v[k+1];
   v[k+1] = temp;
}
```



Assembly language program (for RISC-V)

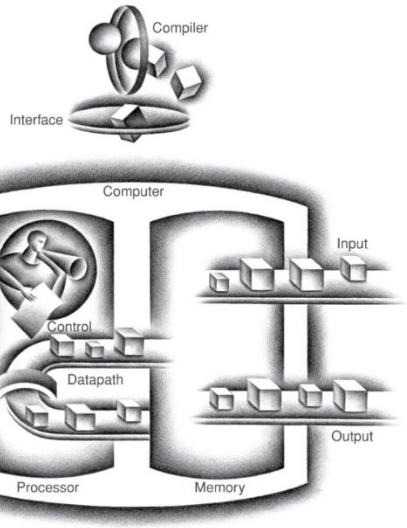
```
swap:
        slli x6, x11, 3
        add  x6, x10, x6
        ld   x5, 0(x6)
        ld   x7, 8(x6)
        sd   x7, 0(x6)
        sd   x5, 8(x6)
        jalr x0, 0(x1)
```



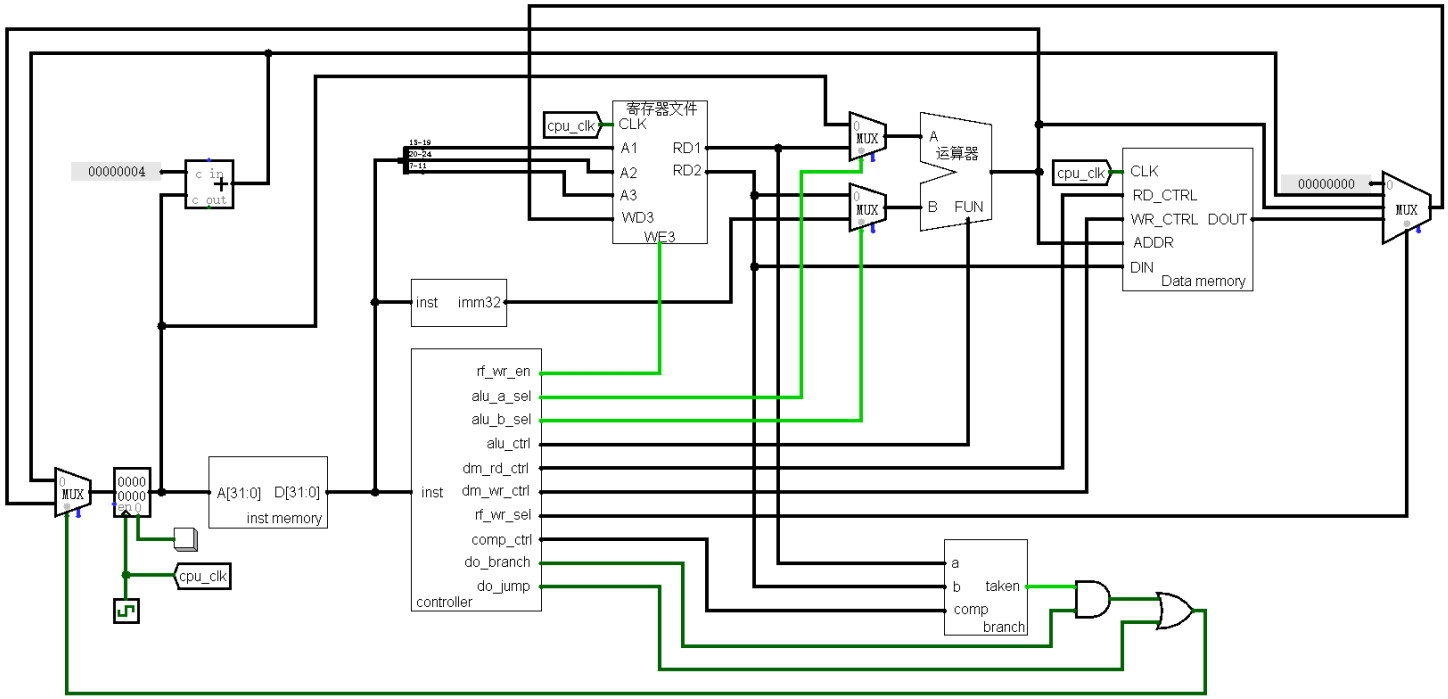
Binary machine language program (for RISC-V)

```
00000000001101011001001100010011
00000000011001010000001100110011
00000000000000011001100101000011
00000000100000110011001110000011
00000000011100110011000000100011
00000000010100110011010000100011
00000000000000001000000001100111
```

1.4 箱盖后的硬件

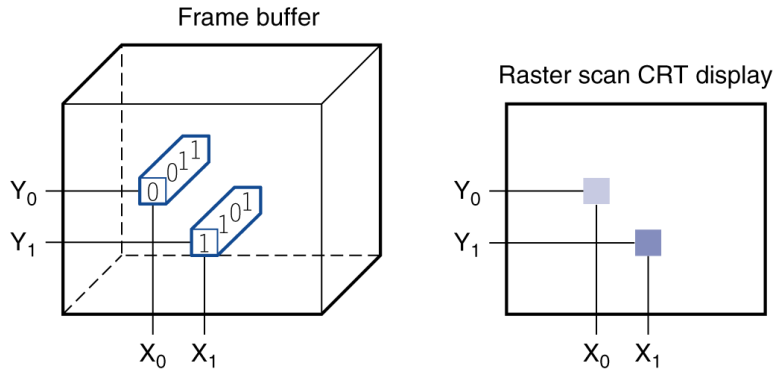


- 输入设备：鼠标、键盘
- 输出设备：显示器
- 存储器：硬盘、内存
- 控制器：处理器的一部分
- 运算器：处理器的另一部分



1.4 箱盖后的硬件

- 显示器
  - LCD：Liquid Crystal Display
  - 显示器的工作原理，2’43”
  - https://www.bilibili.com/video/BV12s411a76M?from=search&seid=12997120343234421829&spm_id_from=333.337.0.0
  - LCD屏幕的工作原理是怎样的，7’20”
  - https://www.bilibili.com/video/BV1Sb411W7Zf?from=search&seid=12997120343234421829&spm_id_from=333.337.0.0
  - 用高速摄像和微距镜头探索电视工作原理，11’38”
  - https://www.bilibili.com/video/BV11W411H7Bo/?spm_id_from=autoNext

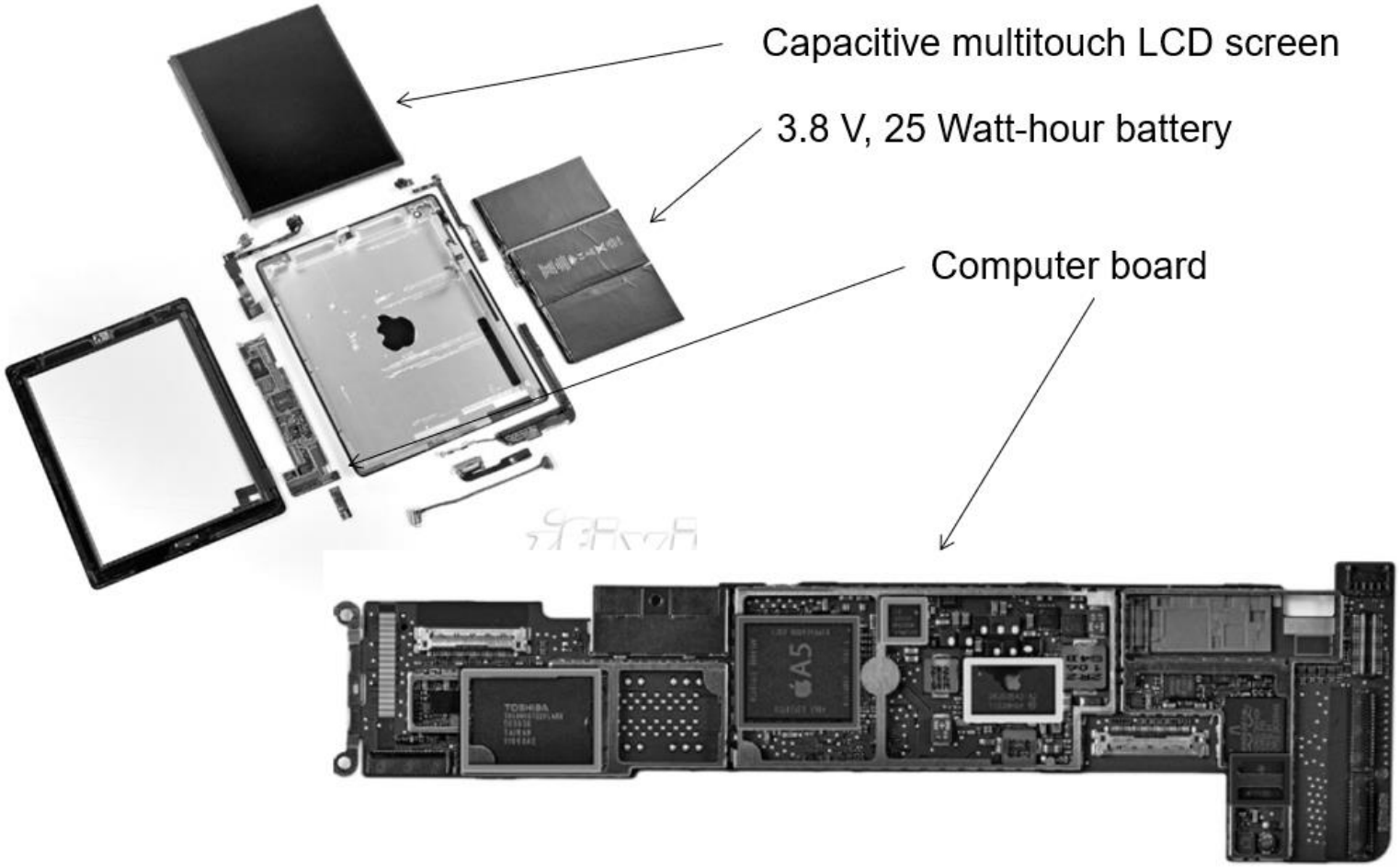


1.4 箱盖后的硬件

- 触摸屏
  - 触摸屏原理 手机电容式触摸屏显示器是如何工作的,8’ 36”
  - https://www.bilibili.com/video/BV1ub4y1n77d?from=search&seid=25256951038393402 89&spm_id_from=333.337.0.0
- 多点触控
  - 触摸屏多点触控原理，5’ 57”
  - https://www.bilibili.com/video/BV14T4y127A1/?spm_id_from=333.788.recommend_more_video.-1

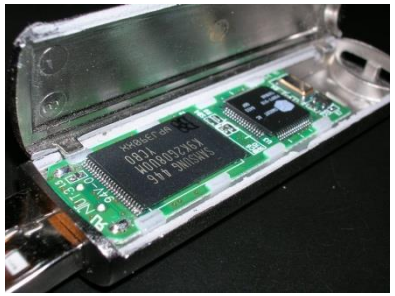
1.4 箱盖后的硬件

- 打开机箱（机壳）



1.4 箱盖后的硬件

- 数据安全
  - 易失性存储器：掉电后数据丢失
    - 寄存器、缓存、内存等
  - 非易失性存储器：掉电后数据不丢失
    - 硬盘、Flash、光盘等



1.4 箱盖后的硬件

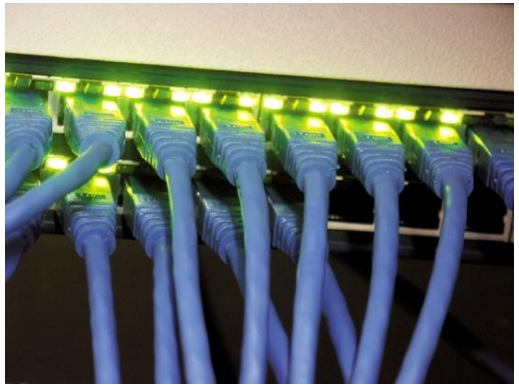
Check Yourself

- Semiconductor DRAM memory, flash memory, and disk storage differ significantly. For each technology, list its volatility, approximate relative access time, and approximate relative cost compared to DRAM.

1.4 箱盖后的硬件

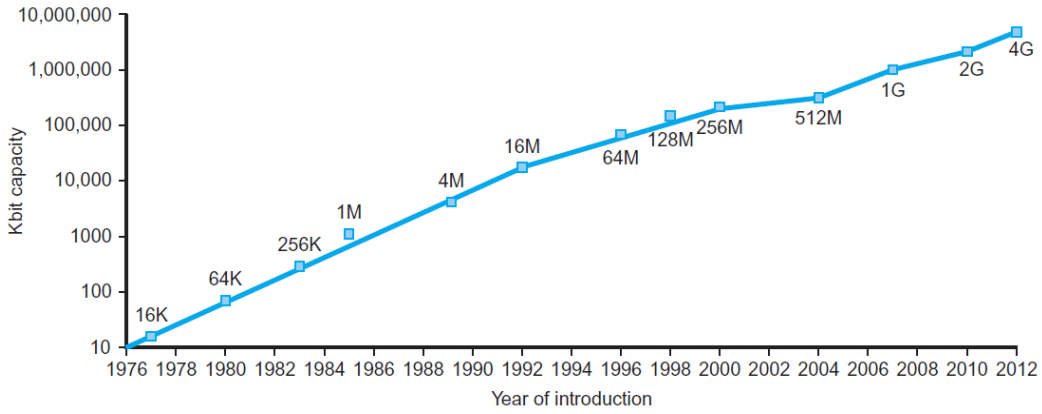
- 网络
  - Communication, resource sharing, nonlocal access
  - Local area network (LAN): Ethernet
  - Wide area network (WAN): the Internet
  - Wireless network: WiFi, Bluetooth

- 带宽
  - 永远不要忽略一辆载满磁带的在高速公路上飞驰的卡车的带宽
  - https://www.zhihu.com/question/20548494/answer/989693899?utm_source=qq



1.5 处理器和存储制造技术

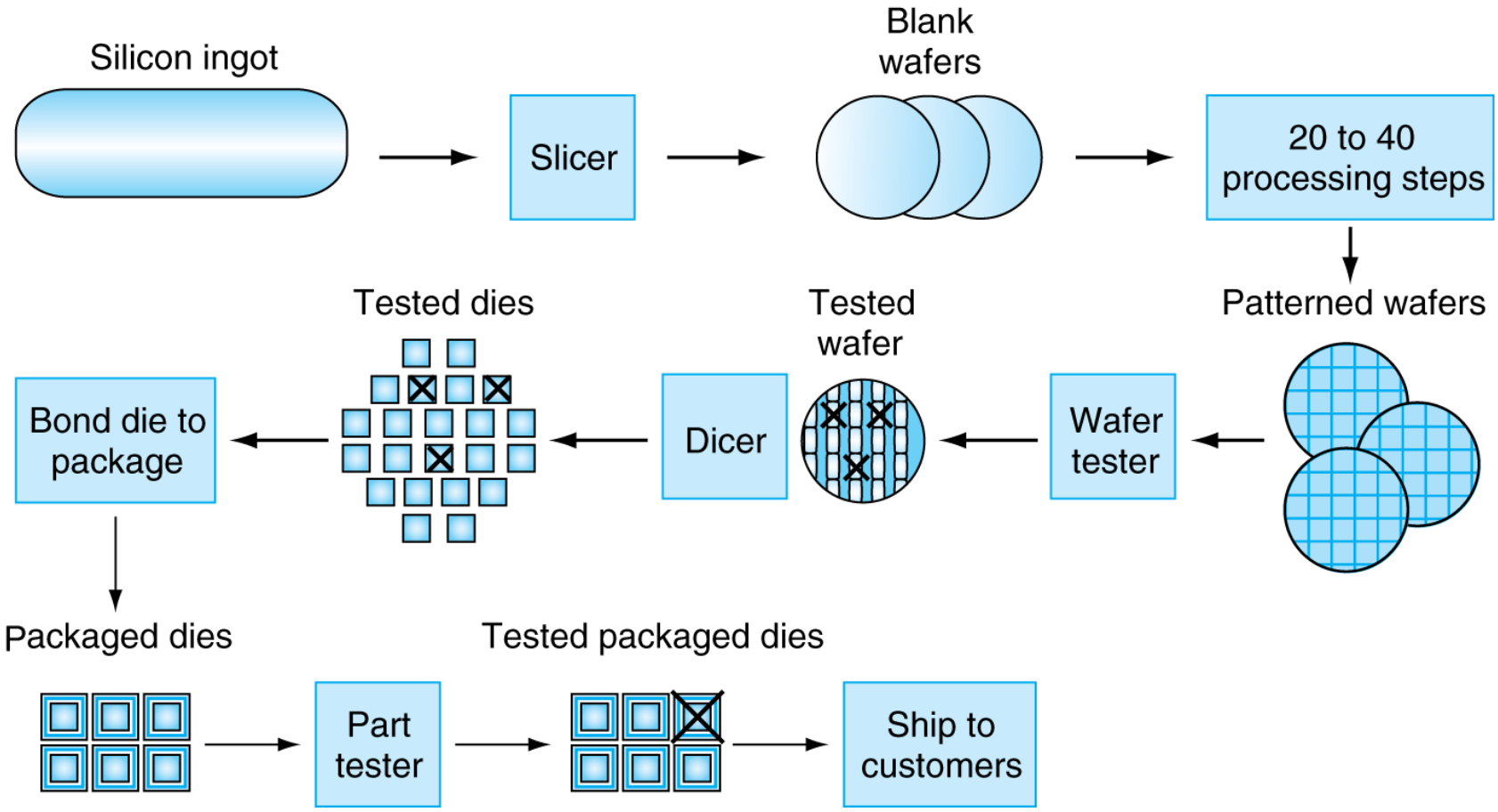
- 硅：Silicon，化学符号Si
- 是一种半导体
- 通过掺杂，可以转变为
  - 导体
  - 绝缘体
  - 开关（可控的导通与截止）



Year	Technology	Relative performance/cost
1951	Vacuum tube	1
1965	Transistor	35
1975	Integrated circuit (IC)	900
1995	Very large scale IC (VLSI)	2,400,000
2013	Ultra large scale IC	250,000,000,000

1.5 处理器和存储制造技术

- ingot → wafer → die → chip
- 良率：yield，合格芯片数占总芯片数的百分比



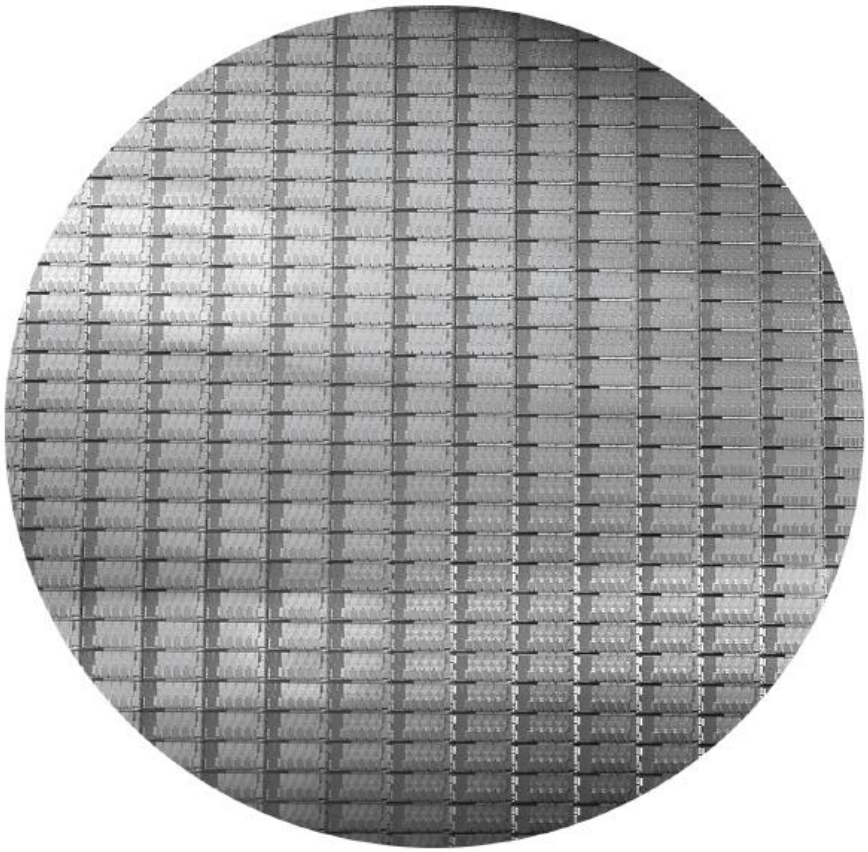
1.5 处理器和存储制造技术

- 碳基 vs 硅基

	ⅠA																0
	1																18
1	1 H 氢 1.008	ⅡA 2											ⅢA 13	ⅣA 14	ⅤA 15	ⅥA 16	ⅦA 17	2 He 氦 4.003
2	3 Li 锂 6.941	4 Be 铍 9.012											5 B 硼 10.81	6 C 碳 12.01	7 N 氮 14.01	8 O 氧 16.00	9 F 氟 19.00	10 Ne 氖 20.18
3	11 Na 钠 22.99	12 Mg 镁 24	ⅢB 3	ⅣB 4	ⅤB 5	ⅥB 6	ⅦB 7	Ⅷ 8	Ⅷ 9	Ⅷ 10	ⅠB 11	ⅡB 12	13 Al 铝 26.98	14 Si 硅 28.09	15 P 磷 30.97	16 S 硫 32.06	17 Cl 氯 35.45	18 Ar 氩 39.95
4	19 K 钾 39.10	20 Ca 钙 40,08	21 Sc 钪 44.96	22 Ti 钛 47.87	23 V 钒 50.94	24 Cr 铬 52.00	25 Mn 锰 54.94	26 Fe 铁 55.85	27 Co 钴 58.93	28 Ni 镍 58.69	29 Cu 铜 63.55	30 Zn 锌 65.41	31 Ga 镓 69.72	32 Ge 锗 72.64	33 As 砷 74.92	34 Se 硒 78.96	35 Br 溴 79.90	36 Kr 氪 83.80
5	37 Rb 铷 85.47	38 Sr 锶 87.62	39 Y 钇 88.91	40 Zr 锆 91.2	41 Nb 铌 92.91	42 Mo 钼 95.94	43 Tc 锝 98	44 Ru 钌 101.1	45 Rh 铑 102.9	46 Pd 钯 106.4	47 Ag 银 107.9	48 Cd 镉 112.4	49 In 铟 114.8	50 Sn 锡 118.7	51 Sb 锑 121.8	52 Te 碲 127.6	53 I 碘 126.9	54 Xe 氙 131.3
6	55 Cs 铯 132.9	56 Ba 钡 137.3	57-71 La-Lu 镧系	72 Hf 铪 178.	73 Ta 钽 180.9	74 W 钨 183.8	75 Re 铼 186.2	76 Os 锇 190.2	77 Ir 铱 192.2	78 Pt 铂 195.1	79 Au 金 197.0	80 Hg 汞 200.6	81 Tl 铊 204.4	82 Pb 铅 207.2	83 Bi 铋 209.0	84 Po 钋 209	85 At 砹 210	86 Rn 氡 222

1.5 处理器和存储制造技术

- Intel Core i7晶圆
  - 300mm晶圆，280chip，32nm工艺
  - chip尺寸：20.7 * 10.5mm



1.5 处理器和存储制造技术

- 集成电路的成本
  - 与面积和缺陷率并不呈线性关系
- 公式1：直接导出
- 公式2：近似
- 公式3：经验公式

$$\text{Cost per die} = \frac{\text{Cost per wafer}}{\text{Dies per wafer} \times \text{Yield}}$$

$$\text{Dies per wafer} \approx \text{Wafer area} / \text{Die area}$$

$$\text{Yield} = \frac{1}{(1 + (\text{Defects per area} \times \text{Die area}/2))^2}$$

1.5 处理器和存储制造技术

Check Yourself

A key factor in determining the cost of an integrated circuit is volume. Which of the following are reasons why a chip made in high volume should cost less?

1. With high volumes, the manufacturing process can be tuned to a particular design, increasing the yield.
2. It is less work to design a high-volume part than a low-volume part.
3. The masks used to make the chip are expensive, so the cost per chip is lower for higher volumes.
4. Engineering development costs are high and largely independent of volume; thus, the development cost per die is lower with high-volume parts.
5. High-volume parts usually have smaller die sizes than low-volume parts and therefore, have higher yield per wafer.

1.6 性能

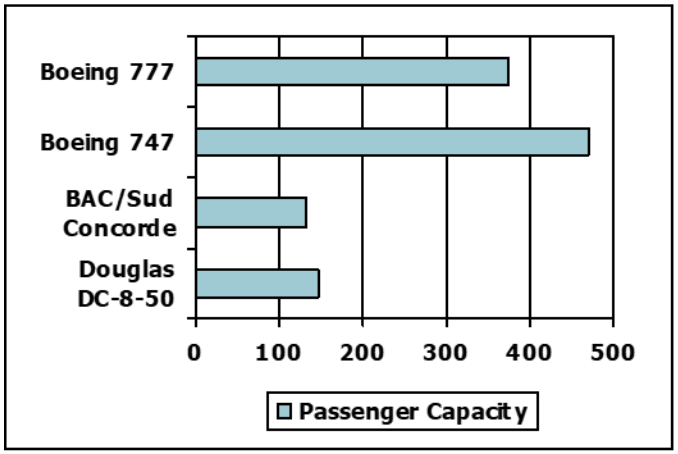
- 什么是性能，如何比较？
  - 以飞机为例，从载客量、航程、航速、乘客吞吐率等方面比较

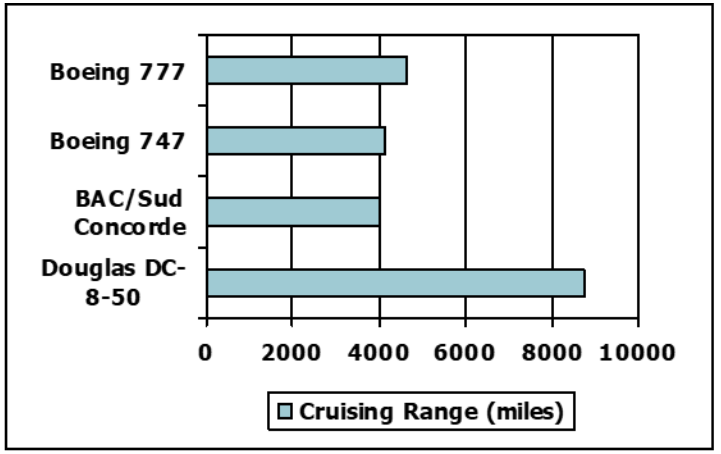
Airplane	Passenger capacity	Cruising range (miles)	Cruising speed (m.p.h.)	Passenger throughput (passengers × m.p.h.)
Boeing 777	375	4630	610	228,750
Boeing 747	470	4150	610	286,700
BAC/Sud Concorde	132	4000	1350	178,200
Douglas DC-8-50	146	8720	544	79,424

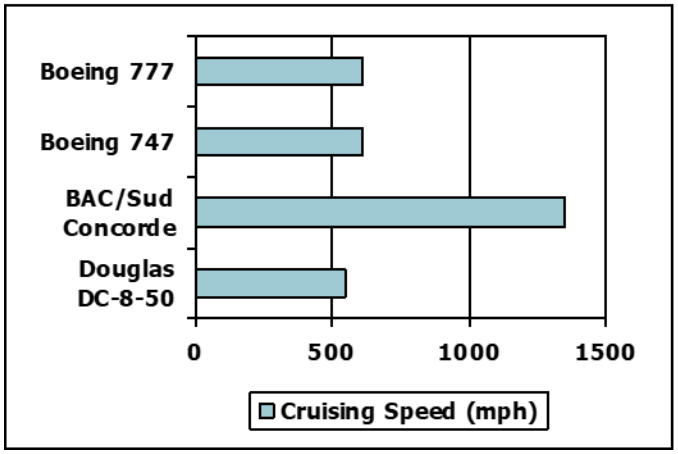
FIGURE 1.14 The capacity, range, and speed for a number of commercial airplanes. The last column shows the rate at which the airplane transports passengers, which is the capacity times the cruising speed (ignoring range and takeoff and landing times).

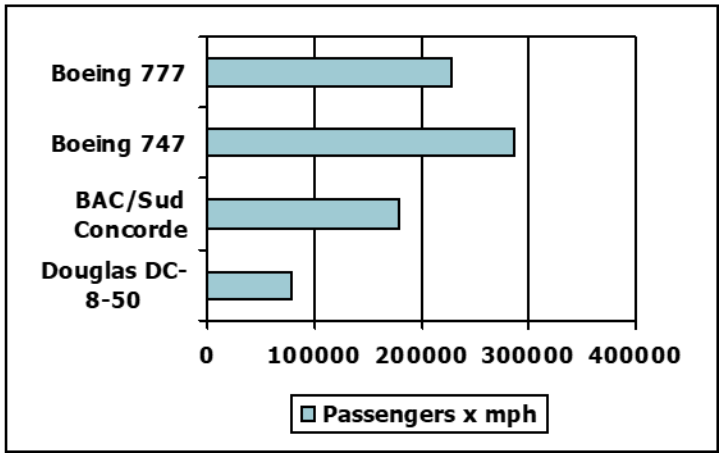
1.6 性能

- 飞机性能比较


Boeing 777
Boeing 747
BAC/Sud Concorde
Douglas DC-8-50
0
100
200
300
400
500
Passenger Capacity


Boeing 777
Boeing 747
BAC/Sud Concorde
Douglas DC-8-50
0
2000
4000
6000
8000
10000
Cruising Range (miles)


Boeing 777
Boeing 747
BAC/Sud Concorde
Douglas DC-8-50
0
500
1000
1500
Cruising Speed (mph)


Boeing 777
Boeing 747
BAC/Sud Concorde
Douglas DC-8-50
0
100000
200000
300000
400000
Passengers x mph

1.6 性能

- 响应时间
  - How long it takes to do a task
- 吞吐率（带宽）
  - Total work done per unit time
- 影响响应时间和吞吐率的因素
  - Replacing the processor with a faster version?
  - Adding more processors?

- 个人电脑：更关注响应时间
- 服务器：更关注吞吐率

1.6 性能

- 性能 = 1/执行时间

- X的执行速度是Y的n倍
  - 性能$_X$ / 性能$_Y$ = 执行时间$_Y$ / 执行时间$_X$ = n
- 示例：程序运行时间
  - 计算机A运行某程序，需要10秒
  - 计算机B运行同一程序，需要15秒
  - 性能$_A$ / 性能$_B$ = 执行时间$_A$ / 执行时间$_B$ = 15s/10s = 1.5

  - 计算机A的性能是B的1.5倍

1.6 性能

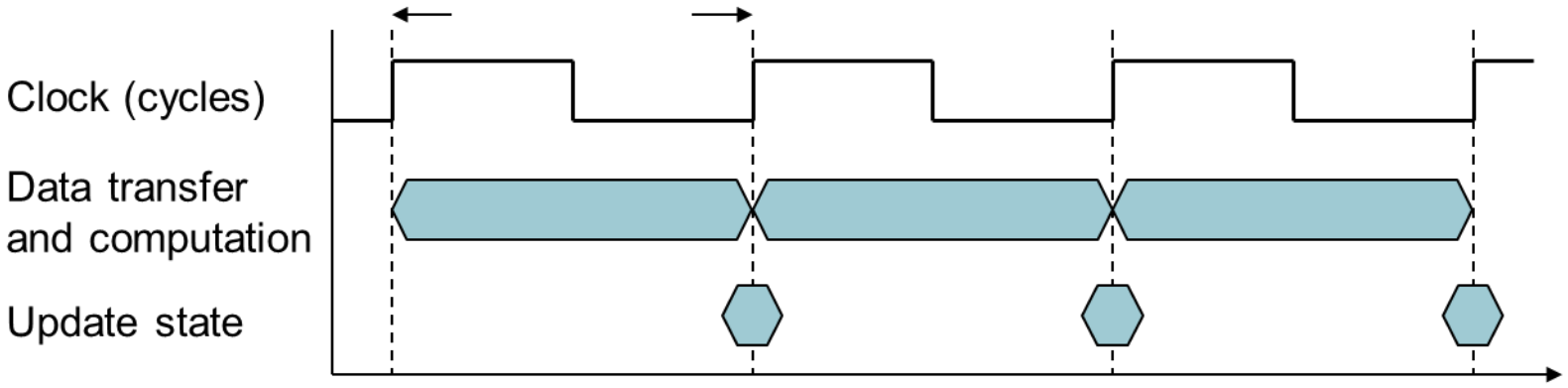
- 性能的度量：时间

- 运行时间：Elapsed time
  - 又称挂钟时间、响应时间
- CPU时间(或CPU执行时间)
  - 执行某一任务，在CPU上所花费的时间
  - 不包括IO时间、运行其它程序的时间
  - CPU时间 = 用户CPU时间 + 系统CPU时间
  - 用户CPU时间：程序本身所花费的CPU时间
  - 系统CPU时间：为执行程序而花费在操作系统上的时间

- 不同程序受CPU性能和系统性能的影响不同

1.6 性能

- CPU时钟/ CPU clock
  - 时钟周期，如：10ns
  - 时钟频率，如：4GHz



1.6 性能

- 影响CPU时间的相关参数
  - 减少时钟数
  - 增加时钟频率
  - 硬件设计者需在时钟频率和时钟周期数之间权衡

$$\text{CPU Time} = \text{CPU Clock Cycles} \times \text{Clock Cycle Time}$$

$$= \frac{\text{CPU Clock Cycles}}{\text{Clock Rate}}$$

1.6 性能

- CPU时间举例
- 某程序在时钟频率为2GHz的计算机A上运行需要10秒，现尝试设计计算机B，将运行时间缩短为6秒，但由于频率的提高，时钟周期数变为计算机A的1.2倍，试计算B的时钟频率

$$\text{Clock Rate}_B = \frac{\text{Clock Cycles}_B}{\text{CPU Time}_B} = \frac{1.2 \times \text{Clock Cycles}_A}{6\text{s}}$$

$$\text{Clock Cycles}_A = \text{CPU Time}_A \times \text{Clock Rate}_A$$

$$= 10\text{s} \times 2\text{GHz} = 20 \times 10^9$$

$$\text{Clock Rate}_B = \frac{1.2 \times 20 \times 10^9}{6\text{s}} = \frac{24 \times 10^9}{6\text{s}} = 4\text{GHz}$$

1.6 性能

- CPI：Clock Pre Instruction
- 程序指令数（Instruction Count）的影响因素
  - 程序、ISA、编译器
- 指令的平均执行周期
  - 由CPU的硬件结构决定
  - 如果不同指令有不同的CPI，则需要考虑平均CPI

$$\text{Clock Cycles} = \text{Instruction Count} \times \text{Cycles per Instruction}$$

$$\text{CPU Time} = \text{Instruction Count} \times \text{CPI} \times \text{Clock Cycle Time}$$

$$= \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}$$

1.6 性能

- 基于权重的平均CPI

$$\text{Clock Cycles} = \sum_{i=1}^{n} (\text{CPI}_i \times \text{Instructio n Count}_i)$$

$$\text{CPI} = \frac{\text{Clock Cycles}}{\text{Instruction Count}} = \sum_{i=1}^{n} \left(\text{CPI}_i \times \underbrace{\frac{\text{Instruction Count}_i}{\text{Instruction Count}}}_{\text{Relative frequency}} \right)$$

1.6 性能

- CPI举例：
- 计算机 A：Cycle Time = 250ps, CPI = 2.0
- 计算机 B：Cycle Time = 500ps, CPI = 1.2
- 相同的ISA
- 用I表示总指令数，哪台计算机更快？快多少？

$$\text{CPU Time}_A = \text{Instruction Count} \times \text{CPI}_A \times \text{Cycle Time}_A$$

$$= I \times 2.0 \times 250\text{ps} = I \times 500\text{ps}$$ ← A is faster…

$$\text{CPU Time}_B = \text{Instruction Count} \times \text{CPI}_B \times \text{Cycle Time}_B$$

$$= I \times 1.2 \times 500\text{ps} = I \times 600\text{ps}$$

$$\frac{\text{CPU Time}_B}{\text{CPU Time}_A} = \frac{I \times 600\text{ps}}{I \times 500\text{ps}} = 1.2$$ ← …by this much

1.6 性能

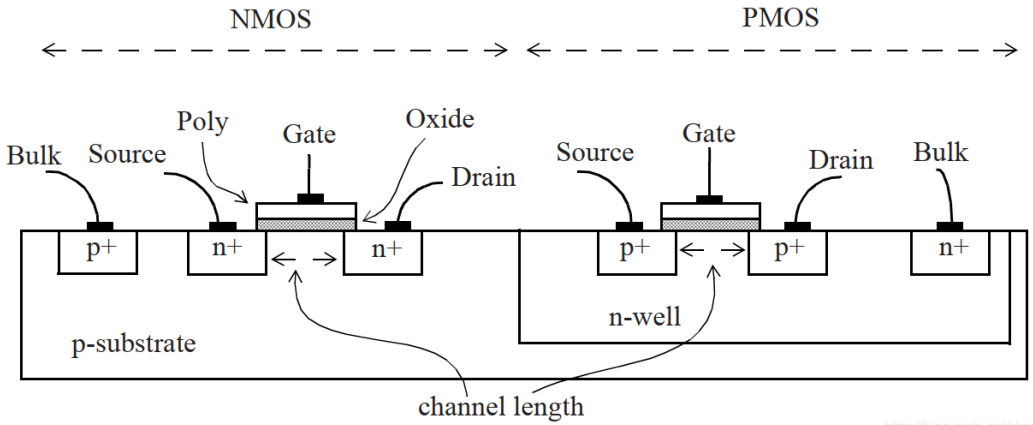
- CPI相关例题，请参考教材

$$\text{CPU Time} = \frac{\text{Instructio ns}}{\text{Program}} \times \frac{\text{Clock cycles}}{\text{Instructio n}} \times \frac{\text{Seconds}}{\text{Clock cycle}}$$

- Performance depends on
  - Algorithm: affects IC, possibly CPI
  - Programming language: affects IC, CPI
  - Compiler: affects IC, CPI
  - Instruction set architecture: affects IC, CPI, T_c

1.7 功耗墙

- CMOS工艺集成电路的功耗
  - 静态功耗：漏电流，与工艺相关



  - 动态功耗：与负载、电压、频率相关

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

×30　　5V → 1V　　×1000

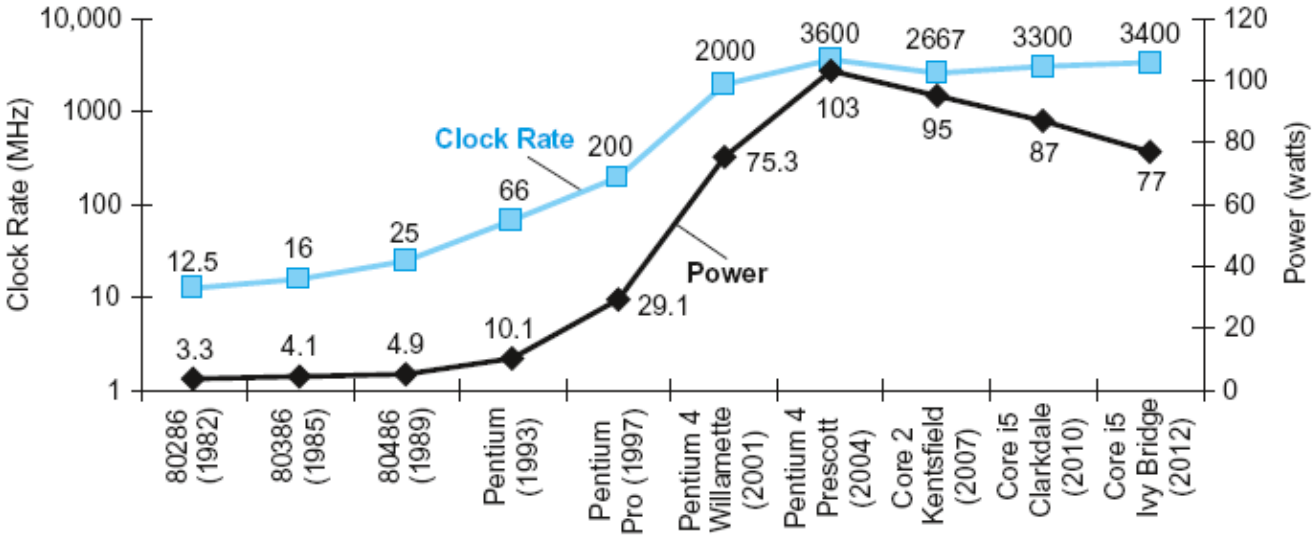
1.7 功耗墙

- 如何降低功耗?
- 假设一个新的CPU
  - 负载电容降为原来的85%
  - 电压和频率也都降为原来的85%

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

- 难题
  - 无法继续降低电压，电压降低将导致漏电流增大
  - 缺乏有效的降温手段
- 如何进一步提高计算机性能?

1.7 功耗墙



谁最早提出多核处理器？谁制造了第一个多核处理

我来答 分享 举报

1个回答 #热议# 医生收受红包、抢着给领导买单构成受贿罪吗?

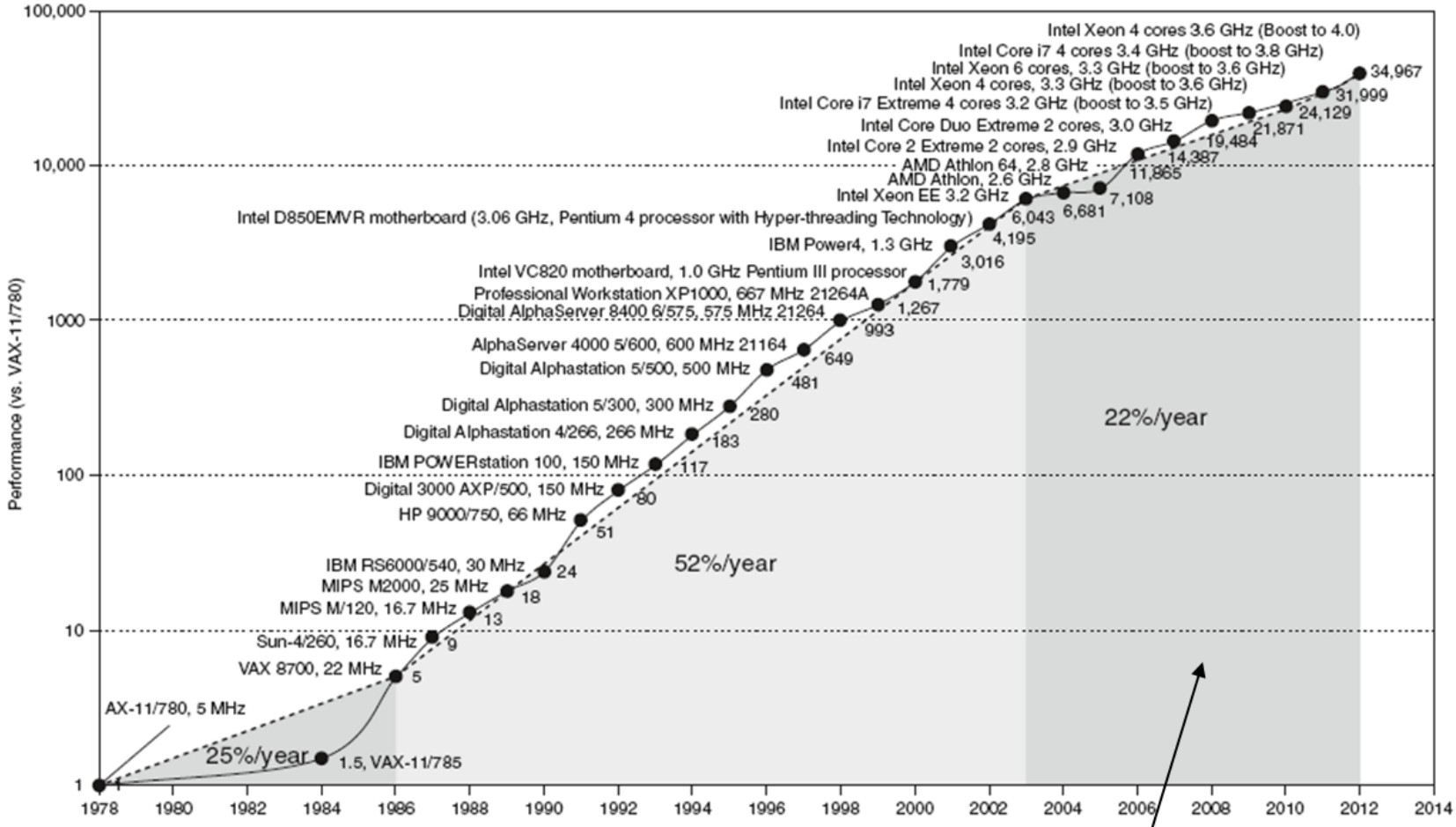
绫波的老公t

2014-07-10 · 超过72用户采纳过TA的回答

关注

AMD最早提出多核，2004提出的，结束了误入歧途的高频之争。 第一款AMD双核应该是速龙X2 3600+，intel第一款应该是奔腾D820(我的第一款电脑就是这款功耗很大的双核）。 如果严格来说，ARM11，9处理器其实就是可以多路并联的CPU，比如诺基亚N96手机就采用了ARM9双核处理器，不过由于CPU太老了，性能一般，那个时候手机是不关注CPU的。真正把手机带到双核的ARM A9的NVIDIA tegra 2处理器。

1.8 沧海巨变：从单处理器到多处理器



Constrained by power, instruction-level parallelism, memory latency

1.8 沧海巨变：从单处理器到多处理器

- 多处理器
  - 每个芯片有多个处理器
- 需要显式并行编程
  - 与指令级并行性比较
    - 硬件一次执行多条指令
    - 对程序员隐藏
- 难点
  - 高性能编程
  - 负载平衡
  - 优化通信和同步

1.9 实例：测评Intel Core i7

- 略

1.10 谬误与陷阱

- 谬误1：低利用率的计算机具有更低功耗
- 谬误2：面向性能的设计和面向能效的设计具有不相关的目标
- 陷阱1：在改进计算机的某个方面时，期望总性能的提高与改进大小成正比
- 陷阱2：用性能公式的一个子集去度量性能

- Amdahl定律：

$$T_{improved} = \frac{T_{affected}}{improvement\ factor} + T_{unaffected}$$

- MIPS：每秒百万条指令数

$$MIPS = \frac{Instruction\ count}{Execution\ time \times 10^6} = \frac{Instruction\ count}{\frac{Instruction\ count \times CPI}{Clock\ rate} \times 10^6} = \frac{Clock\ rate}{CPI \times 10^6}$$

1.11 本章小结

- 8个伟大思想
- ISA
- 性能评价：以真实程序的执行时间为尺度
- CPI
- MIPS
- CPU时间 = 用户CPU时间 + 系统CPU时间
- CPU时间 = 指令数 * CPI * 时钟周期长度
- Amdahl定律