北京大学
PEKING UNIVERSITY

智能硬件体系结构

第一讲：智能硬件体系结构简介

主讲：陶耀宇、李萌

2025年秋季

课程简介

北京大学 PEKING UNIVERSITY

- 培养学生初步理解**智能时代的硬件芯片的工作原理、设计原理与未来发展方向**

指标	课程信息
课程号	04632042
学分	2
课程体系	专业任选
地址	二教414
优秀率	无强制限制
考核方式	出勤（5%）、3次课后作业（10%+10%+10%） 简单硬件编程实验（Lab 1 15% + Lab 2 25% + Lab 3 25%）

扫描二维码：加入智能硬件体系结构（校内）群添加说明：年级-姓名

前置知识要求：无强制先修要求、建议具备最初步编程能力

编程技能：简单Python、Verilog（将通过课程逐步进行教学）

课程网站：

https://aiarchpku.com

推荐教科书：

- **逻辑电路方面**：
  - Digital Integrated Circuits: A Design Perspective - Anantha Chandrakasan
  - CMOS数字集成电路：分析与设计 - 康松默
- **智能计算架构方面**：
  - Computer Architecture: A Quantitative Approach - John L. Hennessy
  - 智能计算系统 - 陈云霁
  - 人工智能芯片设计 - 尹首一

课程简介

- 培养学生初步理解**智能时代的硬件芯片的工作原理、设计原理与未来发展方向**

主讲：陶耀宇

http://taoyaoyu.me/

人工智能研究院副研究员、博雅青年学者，国家优青（海外），华为未名青年学者，长期从事智能芯片体系结构、电路系统研究

主讲：李萌

http://mengli.me/

人工智能研究院研究员/助理教授、博雅青年学者，国家优青（海外），高效安全的多模态人工智能加速算法、跨层次协同优化

助教：詹喆

集成电路学院博士1年级

主要负责CLAB服务器平台、实验课、编程Lab等

助教：李中源

信息科学技术学院本科4年级

主要负责CLAB服务器平台、实验课、编程Lab等

助教：罗子翔

元培学院本科3年级

主要负责作业批改等

课程作业与实验

- **培养学生初步理解智能时代的硬件芯片的工作原理、设计原理与未来发展方向**
  - 分别从**逻辑电路与计算架构2**方面进行全栈式介绍

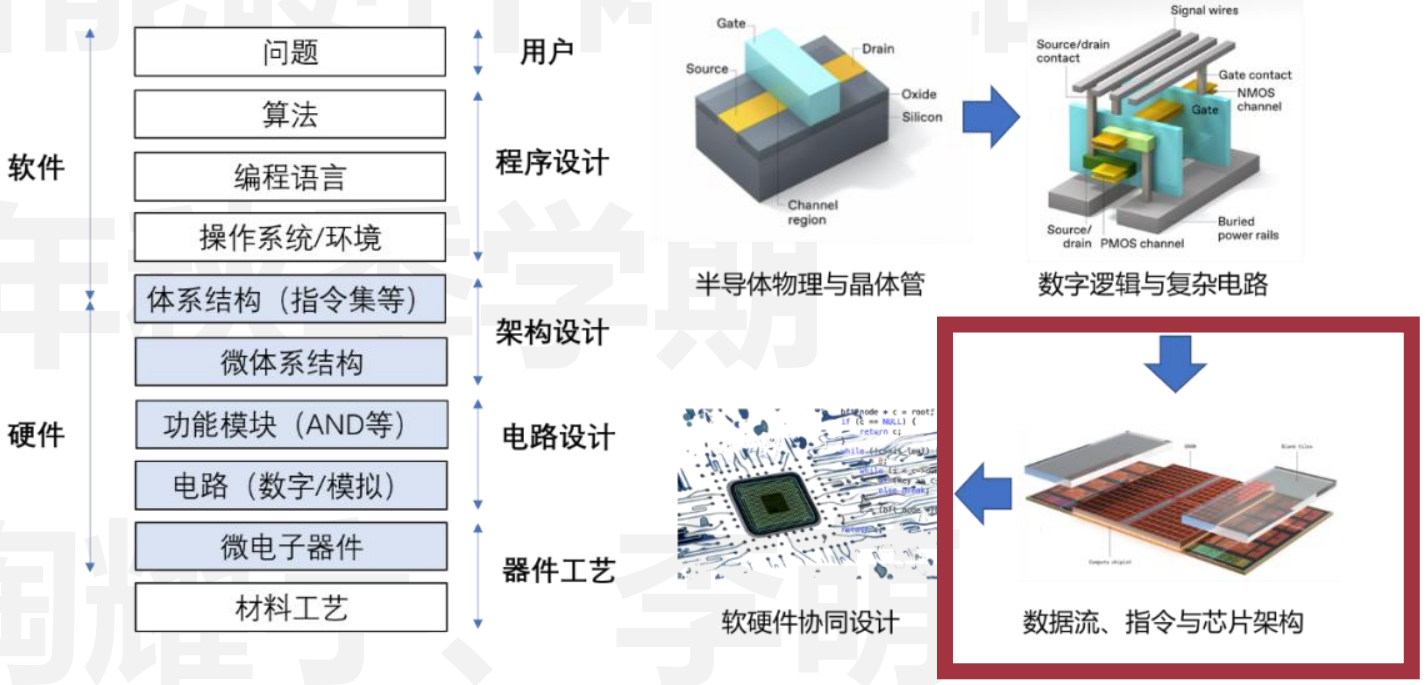
- 本课程开设在北京大学元培学院，**无强制先修课程，建议具备初步的编程技能（任何编程语言都可）。**欢迎各学院、各专业同学选课，了解人工智能硬件全栈知识。
- 特别适合以下同学：
  - **关注AI硬件的发展，想探索AI硬件相关的新方向**
  - **对人工智能硬件物理原理及简单电路设计感兴趣**
  - **想要全面理解智能芯片体系结构及相关全栈技术链**

课程核心内容

- **培养学生初步理解智能时代的硬件芯片的工作原理、设计原理与未来发展方向**
  - 分别从**逻辑电路与计算架构2**方面进行全栈式介绍

课程核心内容：

- **逻辑计算单元与数字电路设计：**学习如何利用晶体管构建计算逻辑、复杂计算单元，了解芯片大规模集成的验证、综合与物理版图
- **多级数据流与指令集计算架构：**学习指令集与流水线设计，数据/控制冲突及其处理机制，了解指令动态发射原理、分支预测与超标量，深入多级缓存微架构、缓存一致性与虚拟缓存等知识
- **AI芯片与未来计算架构新趋势：**学习传统GPU、FPGA、TPU等AI加速器架构原理与发展趋势，了解当前大模型加速芯片、权重量化、稀疏剪枝等技术，学习和了解AI辅助设计、存算一体/感存算一体、光计算架构等新型智能硬件技术



课程作业与实验

- **培养学生初步理解智能时代的硬件芯片的工作原理、设计原理与未来发展方向**
  - 分别从**逻辑电路与计算架构2**方面进行全栈式介绍

3次课后作业

（总计30%，每次占总成绩10%）

① 逻辑门级电路与复杂计算单元、指令集、多级流水线及其控制

② 超标量、乱序执行、存储系统微架构

③ AI加速器架构、未来计算架构

课程作业与实验

- **培养学生初步理解智能时代的硬件芯片的工作原理、设计原理与未来发展方向**
  - 分别从**逻辑电路与计算架构2**方面进行全栈式介绍

- **3次编程实验（总计65%，分别为15%+25%+25%）**
  ① （利用课程所学CMOS器件与逻辑电路知识，构建复杂计算单元（例如：如何在硬件上实现高效卷积运算或三角函数计算？如何在硬件上实现高效的大位宽乘法？），并且评估其硬件性能。所需编程语言：Python/Matlab、Verilog，累计代码编写量：100~150行左右。**（4周时间完成）**

  ② 利用课程所学流水线、指令集、AI芯片架构知识，构建简单芯片架构与电路，能够实际运行某一类计算任务，并且评估其硬件性能。所需编程语言： Linux脚本、Verilog，累计代码编写量：200行左右。**（6周时间完成）**

  ③ 利用课程所学软硬件协同设计知识，使用CUDA或者Triton设计量化和稀疏的GPU算子。实验平台使用Google的Colab，课程提供Colab平台使用CUDA的基础编译工具链示例。所需编程语言：Linux脚本、Python等，累计代码编写量：200行左右。**（6周时间完成）**

***课程将提供所需完整工程环境，无需使用个人电脑配置环境。**

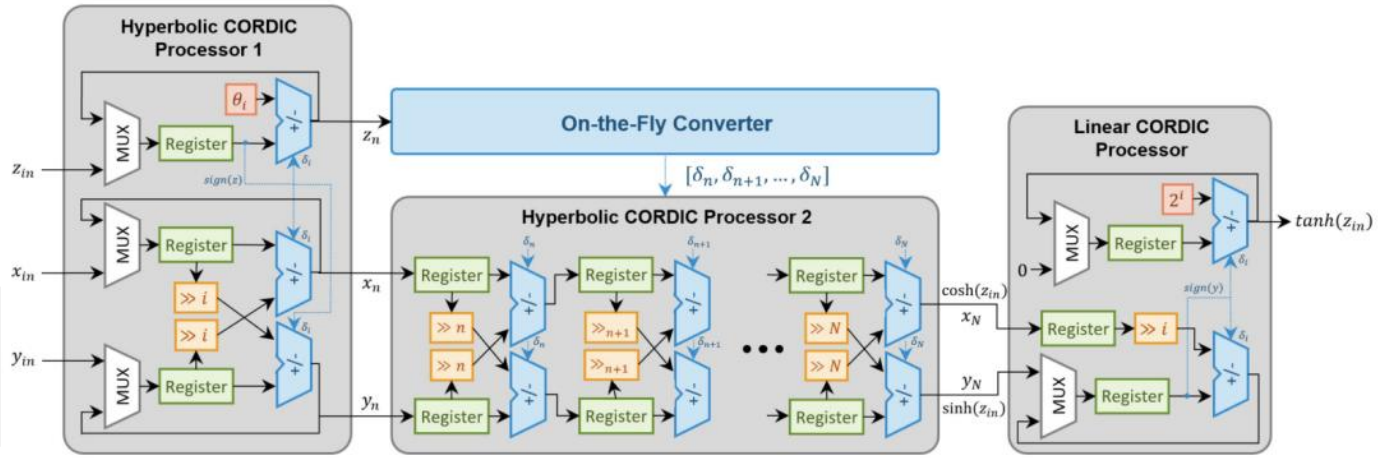
课程实验环境

- **培养学生初步理解智能时代的硬件芯片的工作原理、设计原理与未来发展方向**
  - 分别从**逻辑电路与计算架构2**方面进行全栈式介绍
    - 项目采用CLAB平台：https://clab.pku.edu.cn/、具体使用方式请参考： CLAB使用手册
    - 实验采用Linux环境进行开发
      - 所需软件环境已为各位同学安装好，无需自己配置环境
      - Linux运行Lab的说明请参考：Linux使用参考信息
    - 助教正在为各位同学建立CLAB账号，具体事宜请同学们联系助教詹喆同学
    - **第2周开始，为大家提供Verilog代码入门习题课，支持硬件0基础的同学们**
    - 如有任何问题，欢迎联系授课老师或助教！

课程作业与实验

- **培养学生初步理解智能时代的硬件芯片的工作原理、设计原理与未来发展方向**
  - 分别从**逻辑电路与计算架构2**方面进行全栈式介绍

- **3次编程实验（总计65%，分别为15%+25%+25%）**
  ① （利用课程所学CMOS器件与逻辑电路知识，构建复杂计算单元（例如：如何在硬件上实现高效卷积运算或三角函数计算？如何在硬件上实现高效的大位宽乘法？），并且评估其硬件性能。所需编程语言：Python/Matlab、Verilog，累计代码编写量：100~150行左右。**（4周时间完成）**

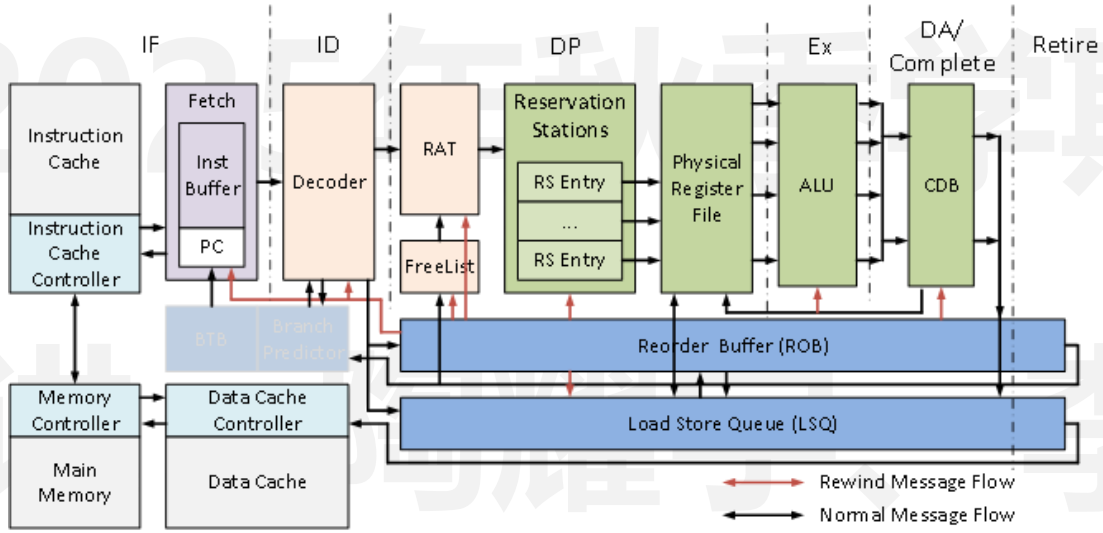


实验1示例：构建复杂计算单元电路（卷积核、CORDIC或大位宽乘法器等）
详情参考：https://aiarchpku.com/2025Spring/project/

课程作业与实验

- **培养学生初步理解智能时代的硬件芯片的工作原理、设计原理与未来发展方向**
  - 分别从**逻辑电路与计算架构2**方面进行全栈式介绍

- **3次编程实验（总计65%，分别为15%+25%+25%）**
  ① 利用课程所学流水线、指令集、AI芯片架构知识，构建简单芯片架构与电路，能够实际运行某一类计算任务，并且评估其硬件性能。所需编程语言： Linux脚本、Verilog，累计代码编写量：200行左右。 **（6周时间完成）**



实验2示例：构建基于指令集的流水线架构
详情参考：https://aiarchpku.com/2025Spring/project/

课程作业与实验

- **培养学生初步理解智能时代的硬件芯片的工作原理、设计原理与未来发展方向**
  - 分别从**逻辑电路与计算架构2**方面进行全栈式介绍

- **3次编程实验（总计65%，分别为15%+25%+25%）**
  ① 利用课程所学软硬件协同设计知识，使用CUDA或者Triton设计量化和稀疏的GPU算子。实验平台使用Google的Colab，课程提供Colab平台使用CUDA的基础编译工具链示例。所需编程语言：Linux脚本、Python等，累计代码编写量：200行左右。 **（6周时间完成）**

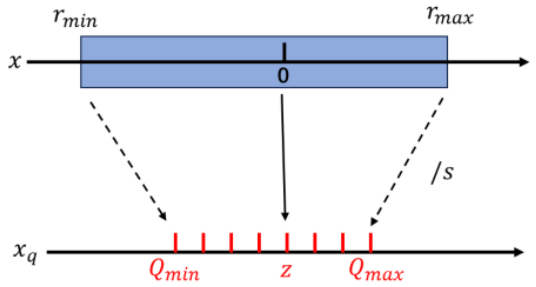


图 1 量化示意图

$$x_q = clamp\left(round\left(\frac{x}{s}\right) + z; Q_{min}, Q_{max}\right)$$

实验3示例：基于 CUDA/Triton 的 GPU 算子设计与量化
详情参考：https://aiarchpku.com/24Fall/project/

目 录

CONTENTS

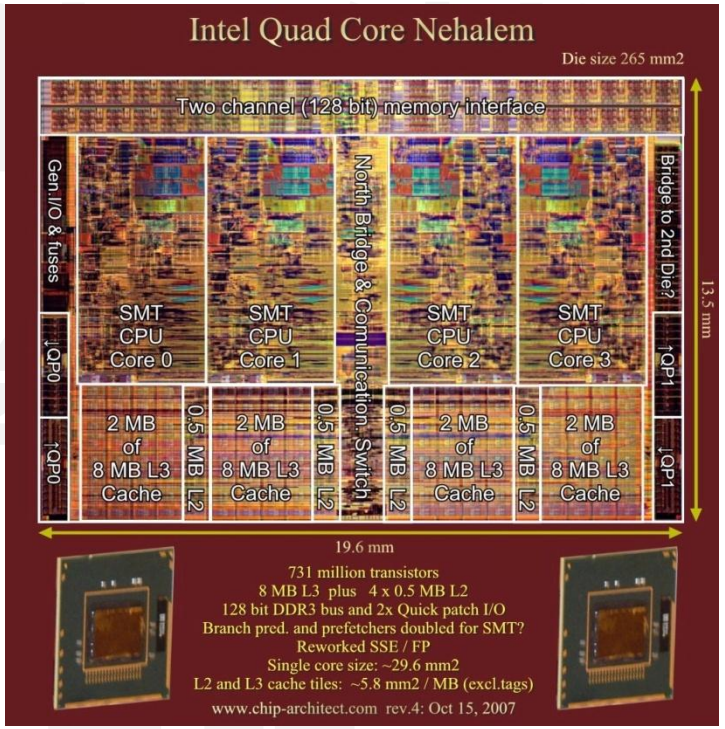
什么是硬件体系结构?

- 硬件体系结构这一概念的随着现代计算机的出现而出现，由Amdahl首次提出

"The term *architecture* is used here to describe the attributes of a system as seen by the programmer, i.e., the conceptual structure and functional behavior as distinct from the organization of the dataflow and controls, the logic design, and the physical implementation."

***Gene Amdahl*, IBM Journal of R&D, April 1964**

吉恩•阿姆达尔：IBM大型机之父

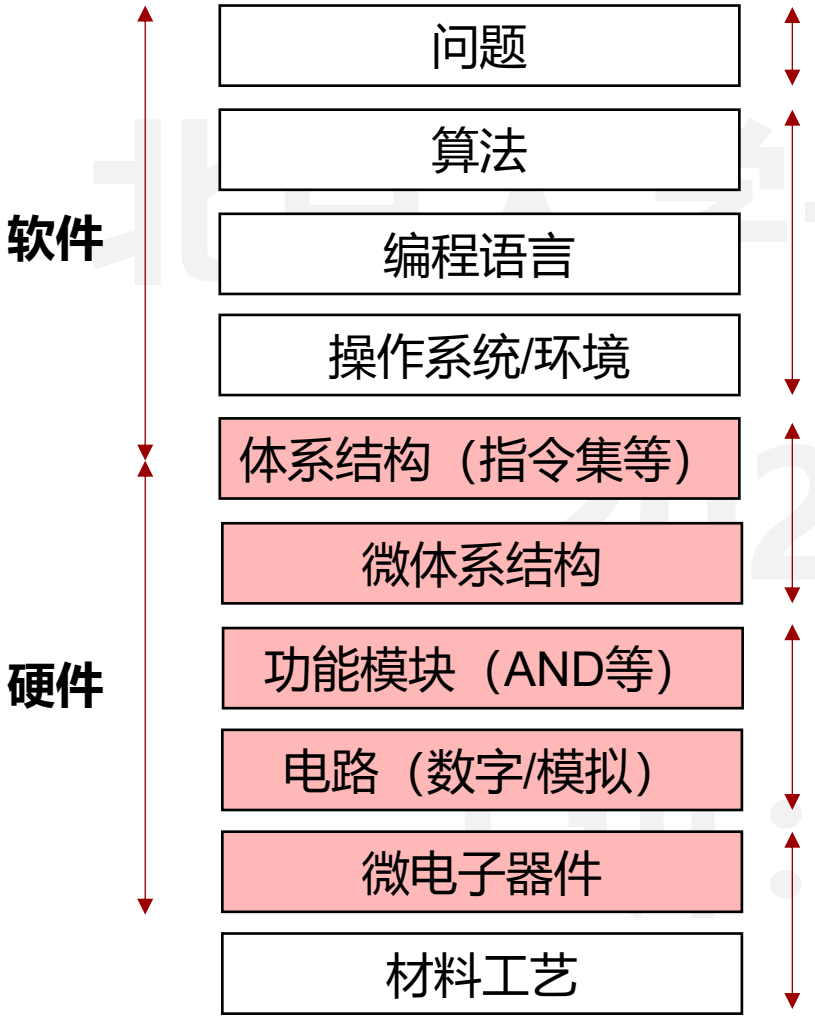


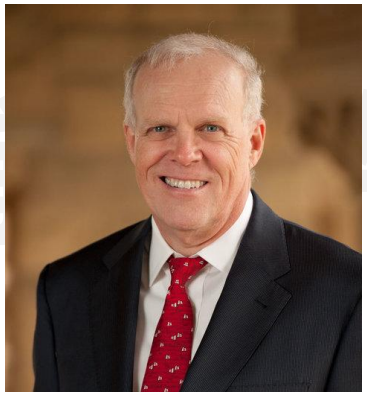
从1956年的达特茅斯会议开始，人工智能（Artificial Intelligence，AI）作为一个专门的研究领域出现

芯片体系结构作为一个独立研究领域的出现，甚至晚于人工智能

为什么要学习硬件体系结构?

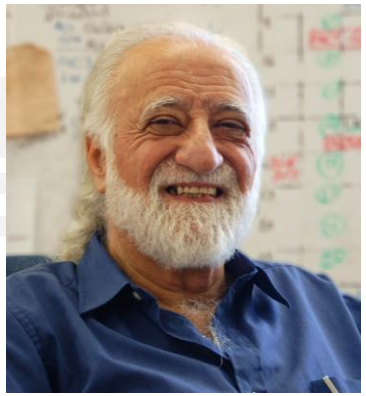
- 硬件体系结构是链接底层微电子器件电路与上层计算任务之间不可或缺的纽带





John L. Hennessy
Stony Brook/Stanford
美国科学院/
工程院院士
2017年图灵奖

David Patterson
UCLA/UC Berkeley
美国科学院/
工程院院士
2017年图灵奖



Yale Patt
UMich/UT Austin
美国工程院
院士
2016年富兰克林奖

上世纪80年代发明了RISC指令集架构
并通过MIPS/Sun等公司成功商业化

现代计算机广泛应用的
乱序并行/分支预测

推动现代硬件体系结构的先驱者

为什么要学习硬件体系结构?

- 体系结构是为了解决上世纪60年代出现的实际工程问题 – 如何链接多样算法与单一硬件?

IBM Compatibility Problem in Early 1960s

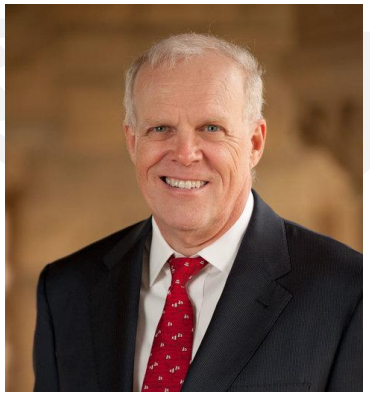
By early 1960's, *IBM had 4 incompatible lines of computers*

701 ➡ 7094
650 ➡ 7074
702 ➡ 7080
1401 ➡ 7010

Each system had its own:

- Instruction set architecture (ISA)
- I/O system and Secondary Storage:
  magnetic tapes, drums and disks
- Assemblers, compilers, libraries,...
- Market niche: business, scientific, real time, ...

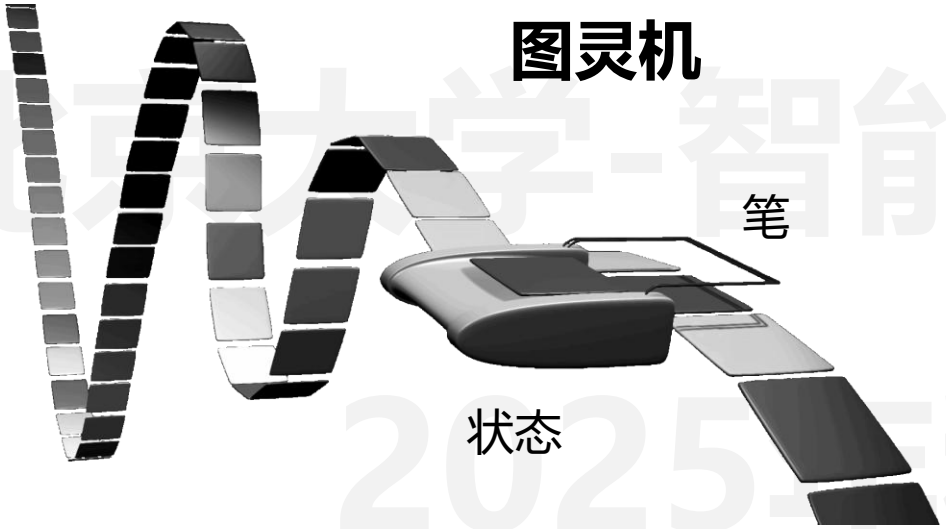
Stanford



John L. Hennessy
StonyBrook/Stanford
美国科学院/工程院
院士
2017年图灵奖

现代硬件体系结构的来源

- 图灵计算理论催生出以图灵机为理论支撑的现代智能芯片体系结构



运算法则

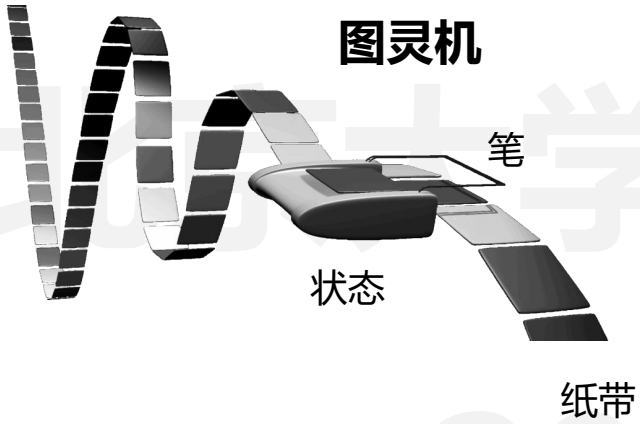
纸带符号	状态A			状态B		
	写	移动	状态	写	移动	状态
0	1	右	B	1	左	A
1	1	左	A	0	右	B

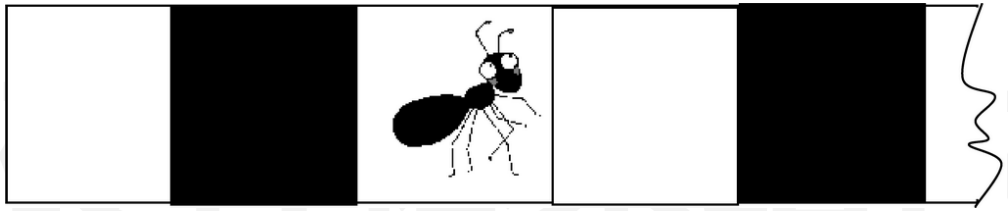
模拟人们用纸笔进行数学运算的过程

- **纸带：**一条无限长的纸带（**TAPE**），被划分为一个接一个的小格子，每个格子上包含一个来自有限字母表的符号
- **笔：**一个读写头（**HEAD**），可以在纸带上左右移动，能读出当前所指的格子上的符号，并能通过写操作改变它
- **运算法则：**一套规则（**TABLE**），根据当前状态及当前读写头所指格子上的符号来确定读写头下一步的动作
- **状态：**一个状态寄存器堆栈（**STATE**），保存图灵机当前的状态

图灵计算理论

- 图灵机的计算方式与工作流程 – **如何理解图灵机：小虫模型**





假设一只小虫在地上爬，那么我们应该怎样从信息处理的角度来建立一个小虫的模型呢？

假设它仅仅具有一个感觉器官：眼睛

小虫的输出动作就是在纸带上前进一个方格或者后退一个方格。

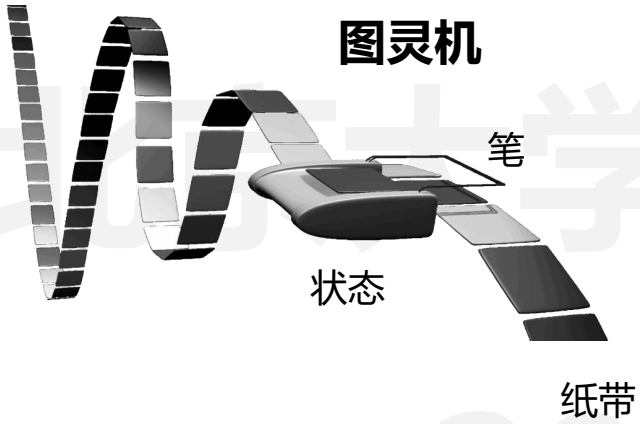
- 仅仅有了输入装置和输出装置，小虫还不能动起来，原因很简单，它并不知道该怎样在各种情况下选择它的输出动作。**于是我们就需要给它指定行动的规则，这就是程序。**
- 假设我们记小虫的输入信息集合为I={黑色，白色}，它的输出可能行动的集合是O={前移，后移}，那么程序就要告诉它在给定了输入（比如黑色）的情况下，它应该选择什么输出。

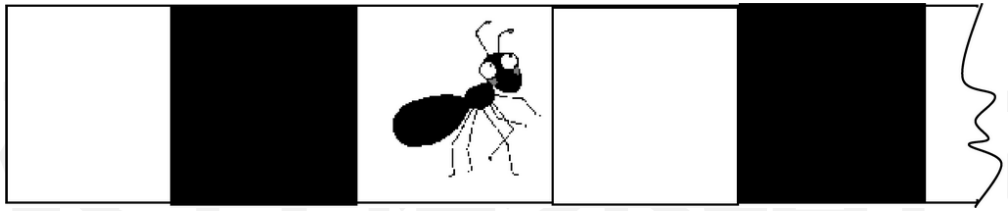
一个简单程序

输入	输出
黑	前移
白	后移

图灵计算理论

- 图灵机的计算方式与工作流程 – 如何理解图灵机：小虫模型





假设一只小虫在地上爬，那么我们应该怎样从信息处理的角度来建立一个小虫的模型呢？

小虫的世界

一个简单程序

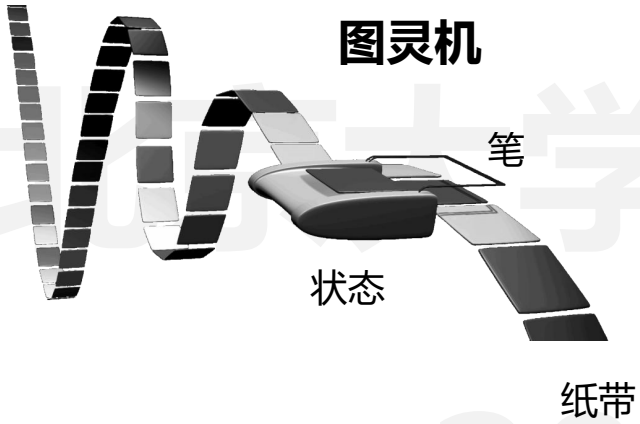
输入	输出
黑	前移
白	后移

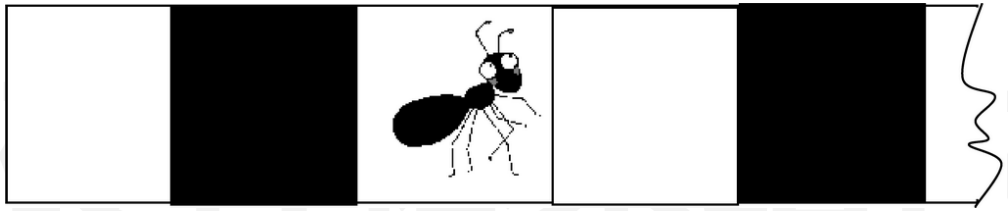
可以预见小虫会无限地循环下去。。。。

我们还需要改进这个最简单的模型

图灵计算理论

- 图灵机的计算方式与工作流程 – 如何理解图灵机：小虫模型





假设一只小虫在地上爬，那么我们应该怎样从信息处理的角度来建立一个小虫的模型呢？

小虫的世界

修改后的简单程序

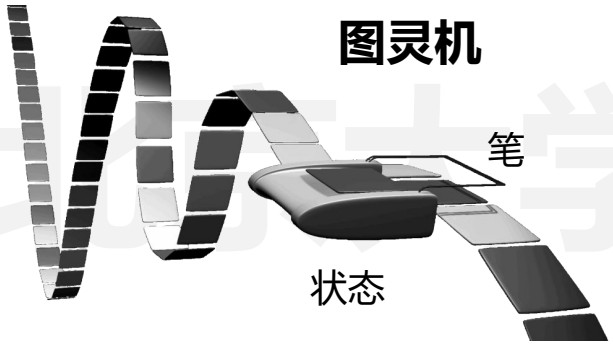
输入	输出
黑	前移
白	**涂黑**

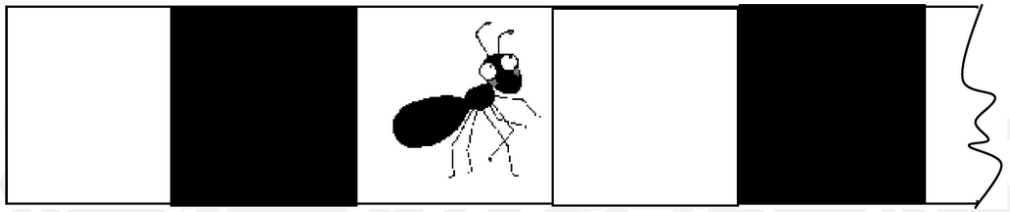
可以预见小虫会无限地循环下去。。。。

我们还需要改进这个最简单的模型

图灵计算理论

- 图灵机的计算方式与工作流程 – 如何理解图灵机：小虫模型

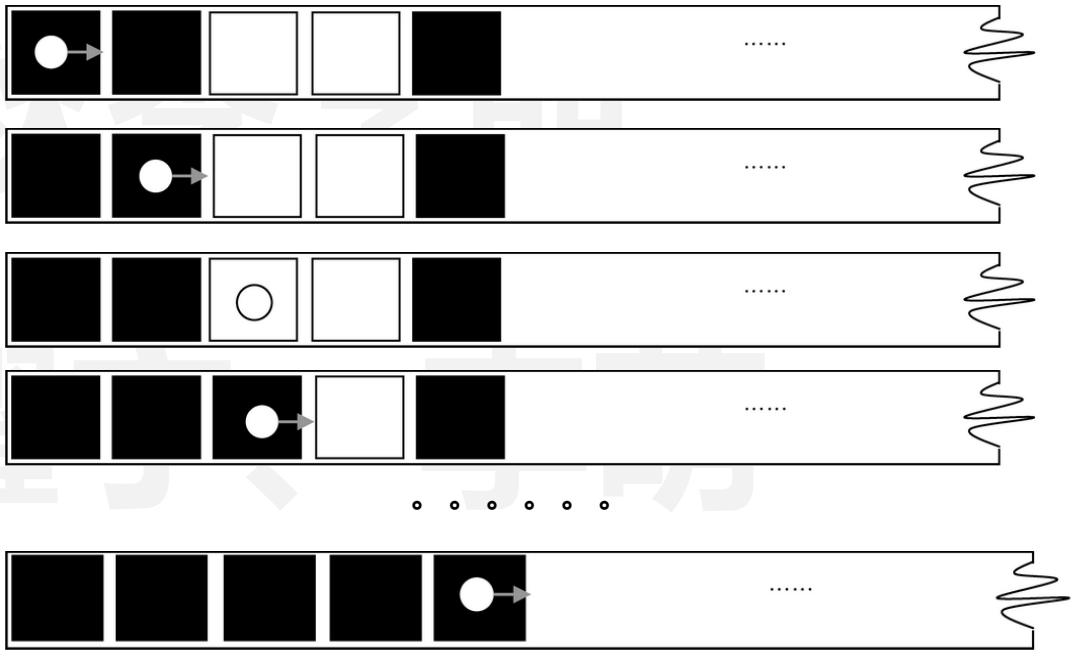




假设一只小虫在地上爬，那么我们应该怎样从信息处理的角度来建立一个小虫的模型呢？

修改后的简单程序

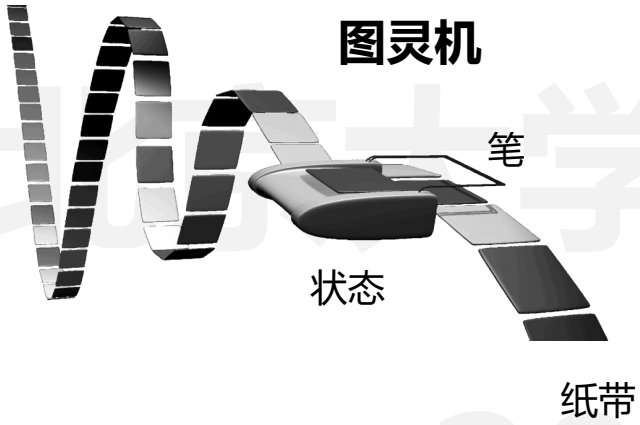
输入	输出
黑	前移
白	**涂黑**



纸带全部涂黑，一直前移

图灵计算理论

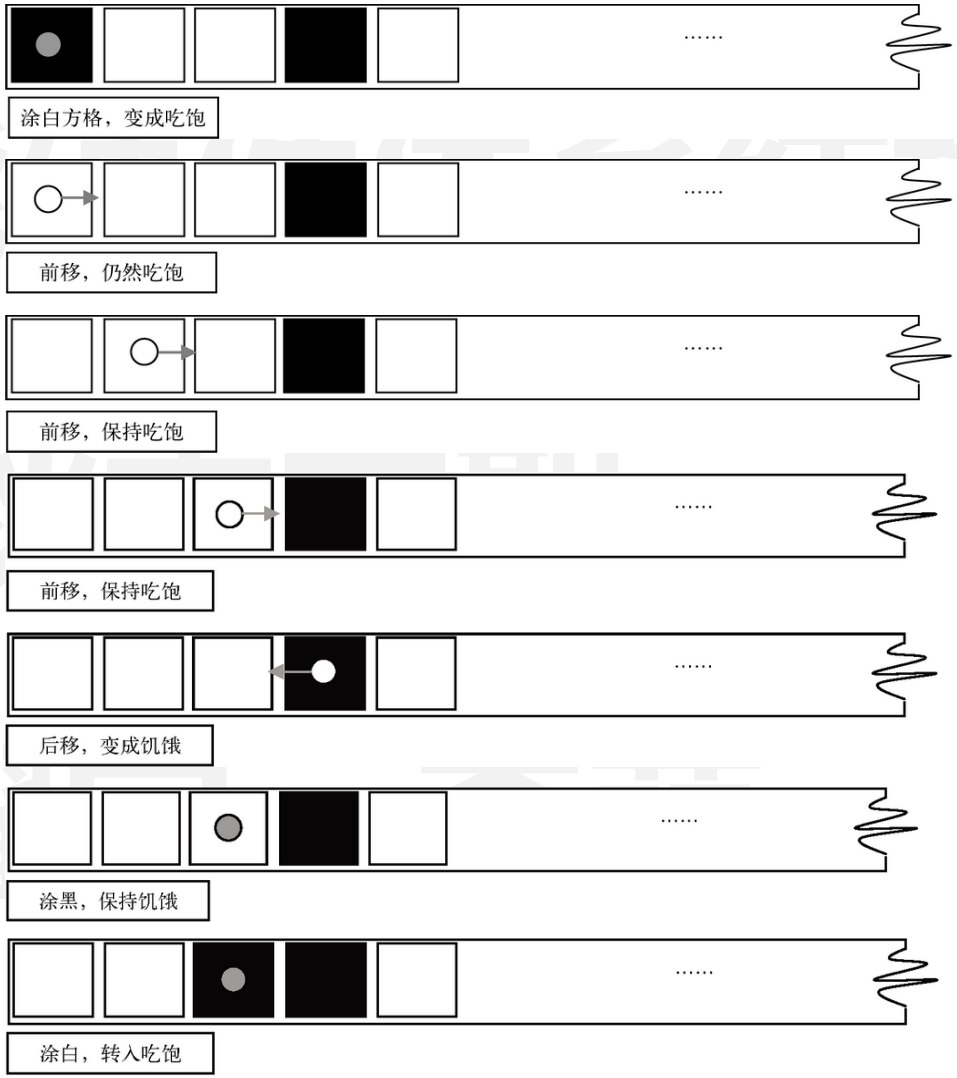
- 图灵机的计算方式与工作流程 – 如何理解图灵机：小虫模型



引入状态、下一个状态的复杂程序

输入	**当前状态**	输出	**下一时刻状态**
黑	**饥饿**	涂白	**吃饱**
黑	**吃饱**	后移	**饥饿**
白	**饥饿**	涂黑	**饥饿**
白	**吃饱**	前移	**吃饱**

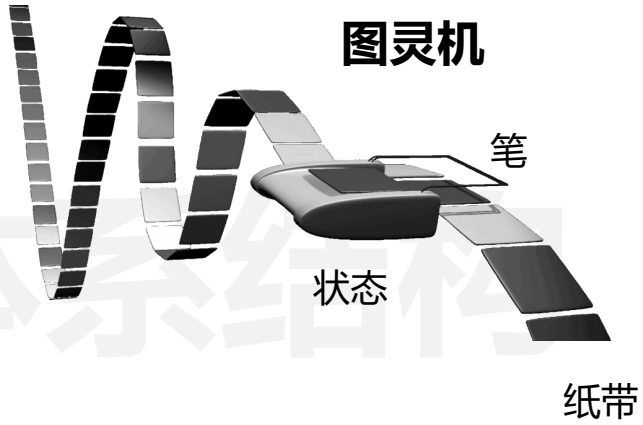
灰色的圆点表示饥饿的小虫，白色的圆点表示它吃饱了



图灵计算理论

- **图灵机的数学理论框架由一个七元有序组定义**

  一台图灵机可被定义为 $T = \{Q, \Sigma, \Gamma, q_0, q_{accept}, q_{reject}, \delta(q,s)\}$

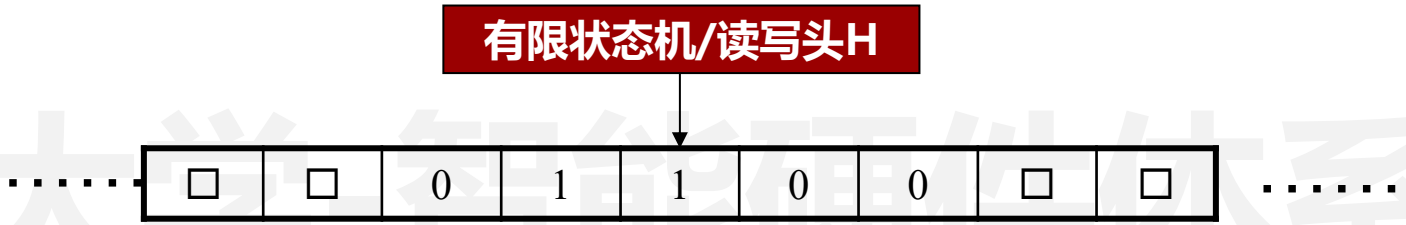


  - **Q**：是非空有限状态集合
  - **Σ**：非空有限输入符号表，其中特殊空白符 □ ∉ Σ
  - **Γ**：非空有限带符号且 Σ⊂ Γ，空白符 □ ∈ Γ- Σ，也是唯一允许出现无限次的字符
  - $q_0 \in Q$ 表示图灵机起始状态
  - $q_{accept} \in Q$ 表示接受状态
  - $q_{reject} \in Q$ 表示拒绝状态, 且$q_{reject} \neq q_{accept}$
  - **$\delta(q,s)$:** $Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ 是转移函数， 根据当前读入符号s和当前状态q决定下一个状态、写入的符号、纸带移动方向和距离， L, R表示读写头是向左移还是向右移, - 表示不移动

图灵计算理论

- 图灵机的计算方式与工作流程

$T = \{Q, \Sigma, \Gamma, q_0, q_{accept}, q_{reject}, \delta(q,s)\}$



- **初始状态：**将输入符号串 $\omega = \omega_0\omega_1 \ldots \omega_{n-1} \in \Sigma^*$ ⟹ 纸带第0,1, ..., n-1 号格子
  - 读写头H指向0号格子，T @ q_0状态
- **运行方式：**T按照转移函数所描述的规则进行计算
  - T@q 状态，H = x，设 $\delta(q, x) = (q', x', L)$
    - T→q' ，H→x'，读写头左移一格
    - 若某时刻H指向0号格子，但根据 $\delta(q, x)$ 将继续左移，则T原地不动
- **停机情况：** 1）若某时刻T @ q_{accept}或q_{reject}，T停机，并接受或拒接ω；
  2）δ(q,s)对某些q和s可能无定义，T停机

由图灵计算理论衍生出的冯诺依曼体系结构

- 图灵机计算范式中的元素可在冯诺依曼架构中找到对应

笔 & 状态 → 数据存储器

纸带 → 指令存储器

运算法则 → 控制逻辑&计算单元

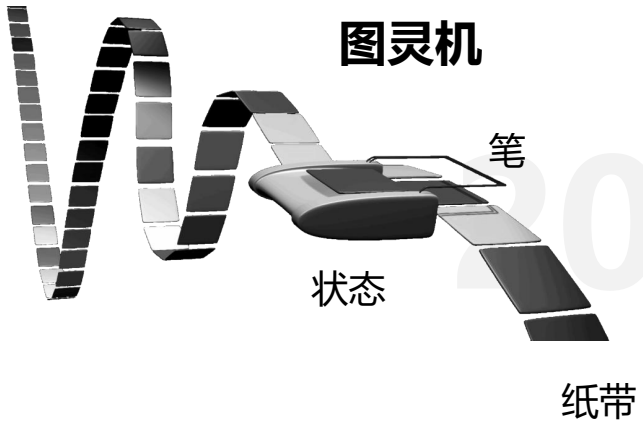
典型运算方式

简单元操作：加、减、乘、除、比较、逻辑门（与、或、非、异或等）

复杂操作：三角函数、幂函数、积分、微分、矩阵操作、卷积、数学变换（傅里叶变换、拉普拉斯变换等）

图灵计算理论

- **图灵机的计算方式与工作流程 – 实例：如何用图灵机计算非操作、加法操作**



假设纸带上写上了1 1 0，将1 1 0做一个非操作，即将1 1 0变成0 0 1

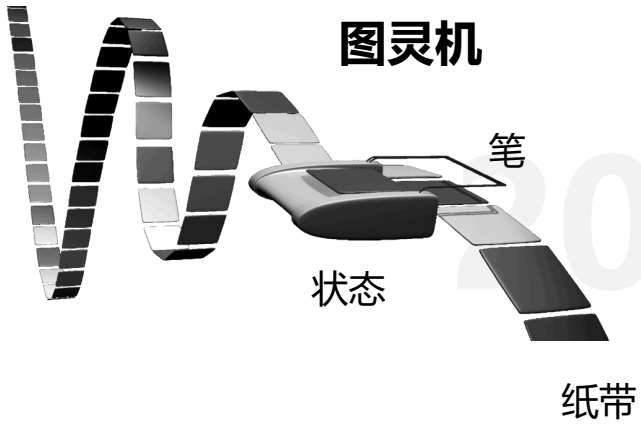
读到的符号	写入指令	移动指令
空	-	-
0	写入1	向右移动纸带
1	写入0	向右移动纸带

假设纸带上写上了111011，假设111 = 3、11=2

读到符号	写入指令	移动指令
空	-	向右移动纸带
0	写入“空”	停机
1	-	向右移动纸带

图灵计算理论

- 图灵机的计算方式与工作流程 – 实例：如何用图灵机计算非操作、加法操作



图灵可计算与不可计算性

对于一个函数，如果不存在一个图灵机能够根据其输入得到其对应的输出，那么这个函数就是不可计算函数

例如：

理发师悖论问题：村子里有个理发师，这个理发师有条原则是，只要村子里有人不自己刮胡子，理发师就给这个人刮胡子。如果这个人自己刮胡子，理发师就不给这个人刮胡子。**求解：理发师会自己刮胡子吗?**

由图灵计算理论衍生出的冯诺依曼体系结构

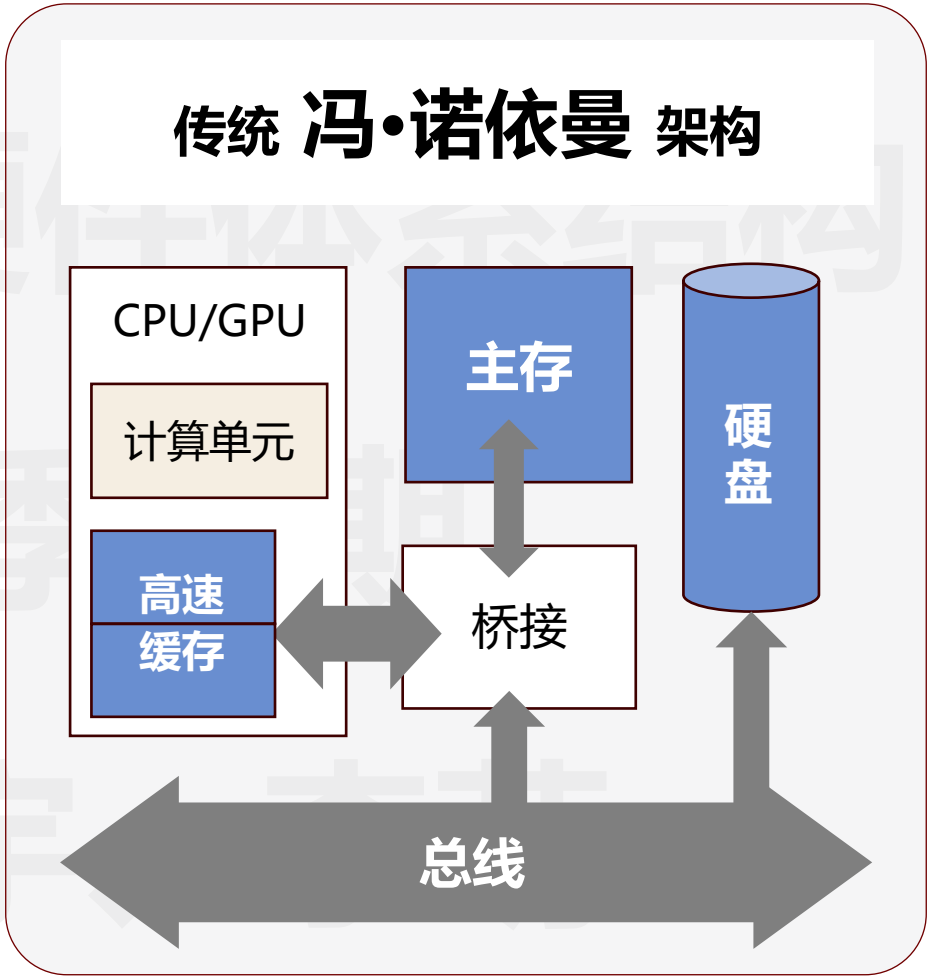
- 图灵机计算范式中的元素可在冯诺依曼架构中找到对应

图灵机

笔 & 状态 → 数据存储器

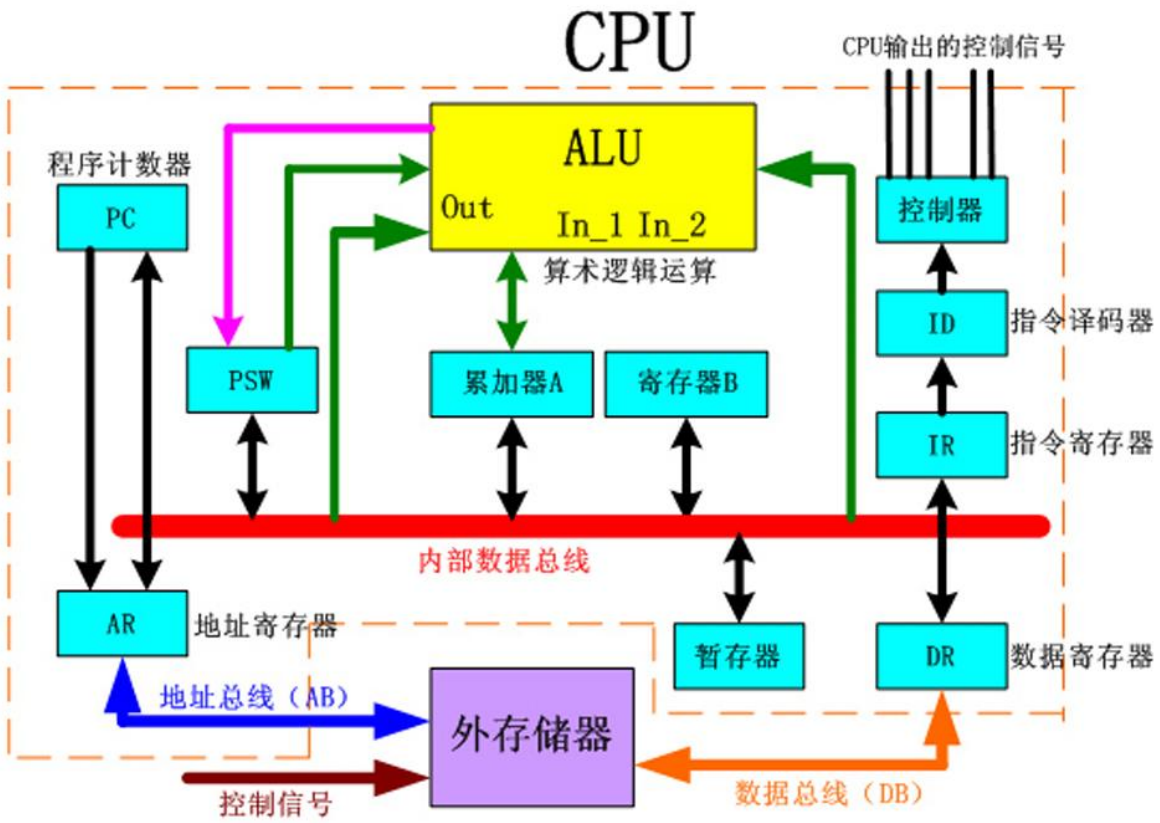
纸带 → 指令存储器

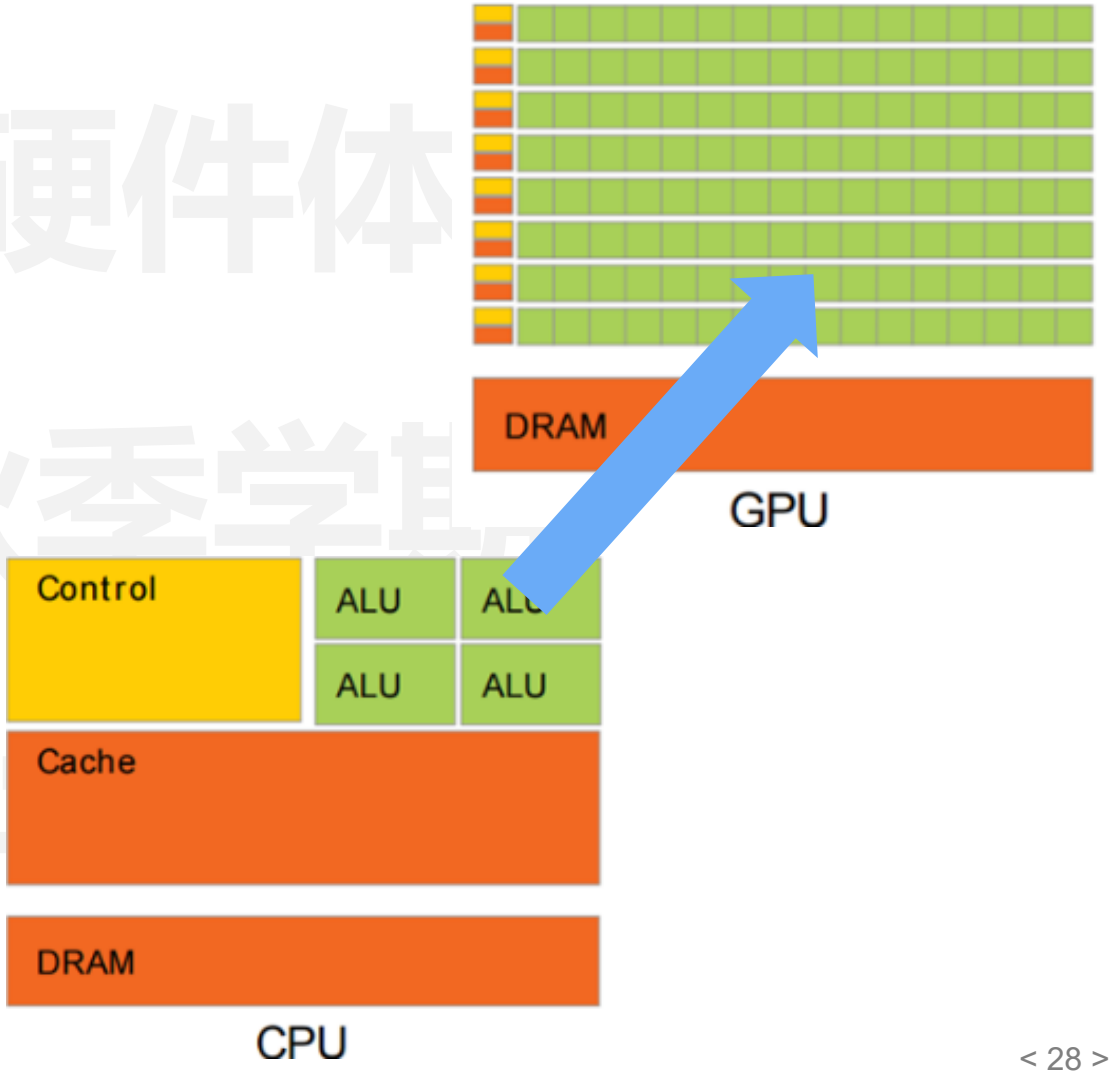
运算法则 → 控制逻辑&计算单元



典型智能芯片体系结构 – CPU/GPU

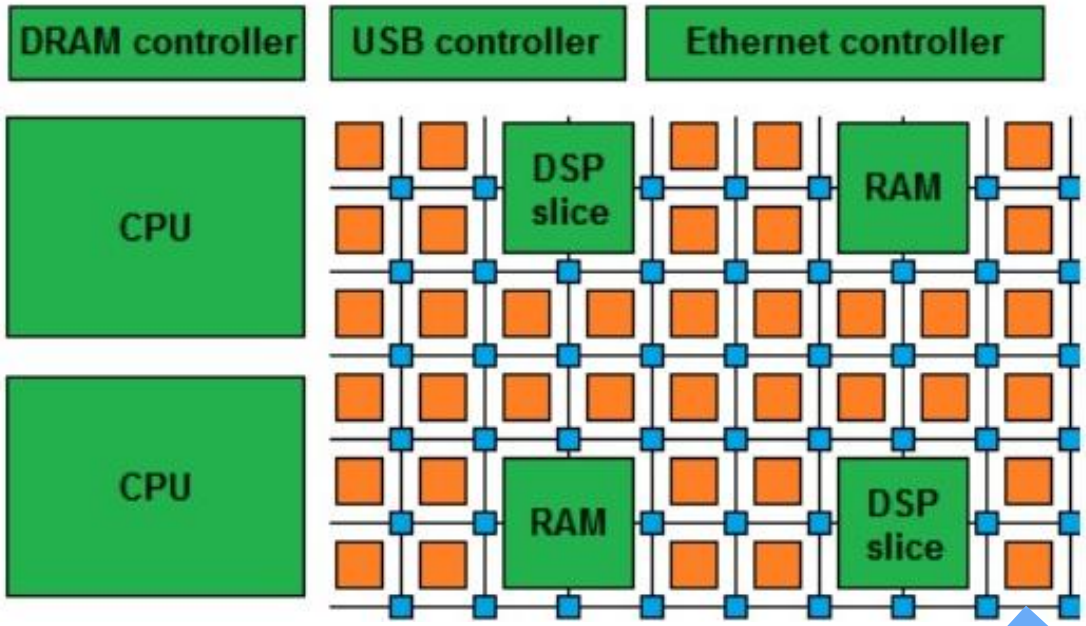
- 从偏向通用计算任务的CPU、GPU到偏向定制化设计的FPGA、ASIC





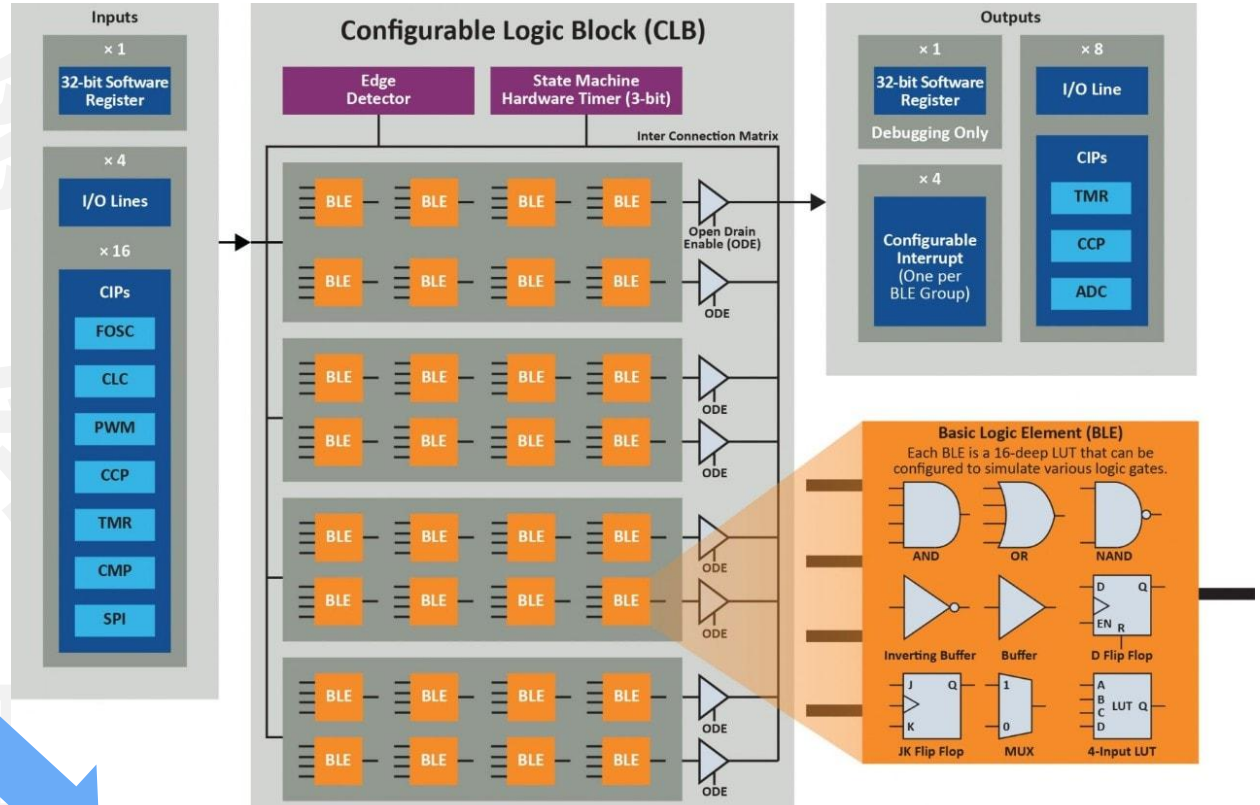
典型智能芯片体系结构 – FPGA

- 从偏向通用计算任务的CPU、GPU到偏向定制化设计的FPGA、ASIC



可编程FPGA

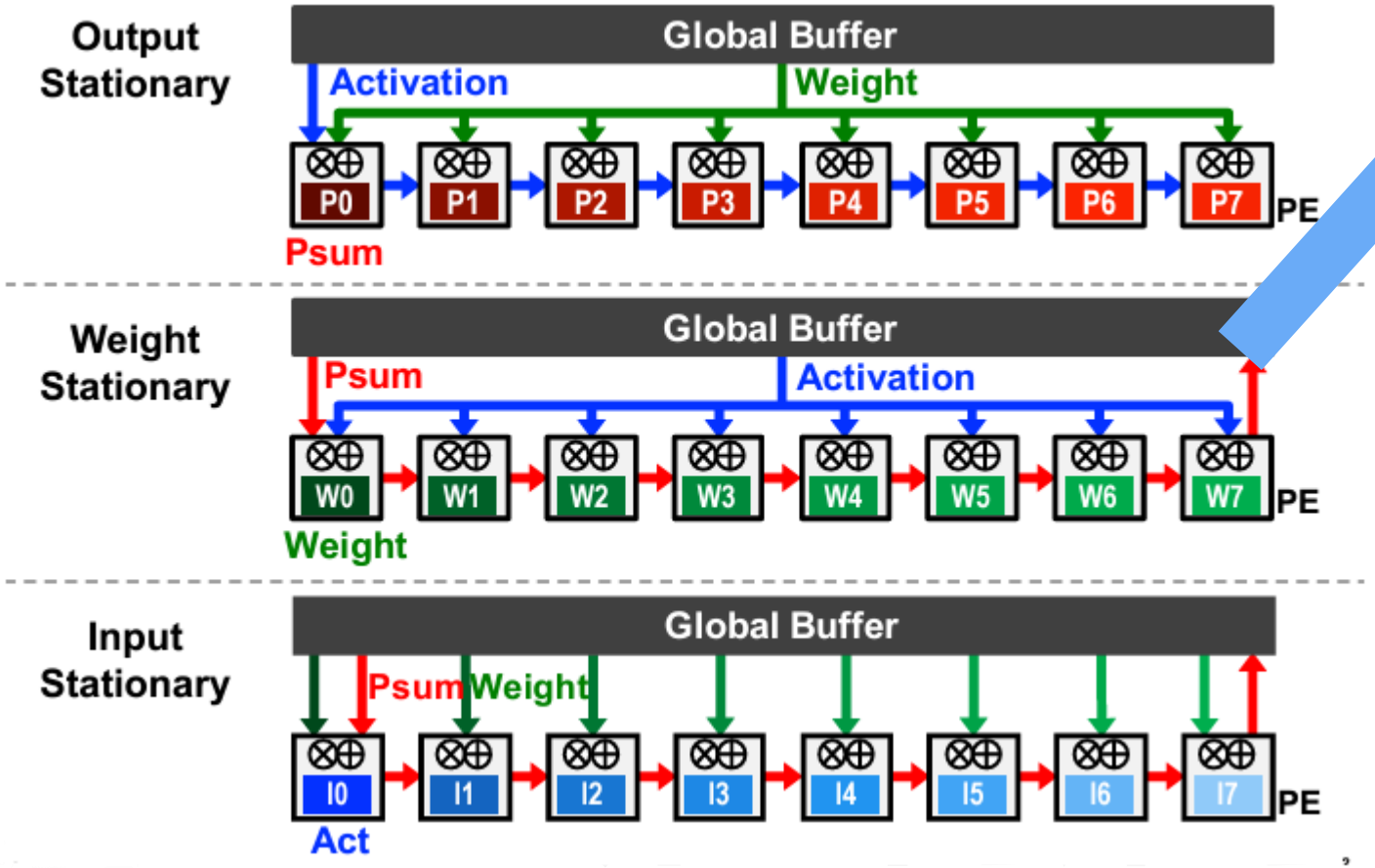
- 比CPU快
- 比GPU省功耗
- 比ASIC便宜流片周期短



可编程逻辑模块CLB

典型智能芯片体系结构 – ASIC

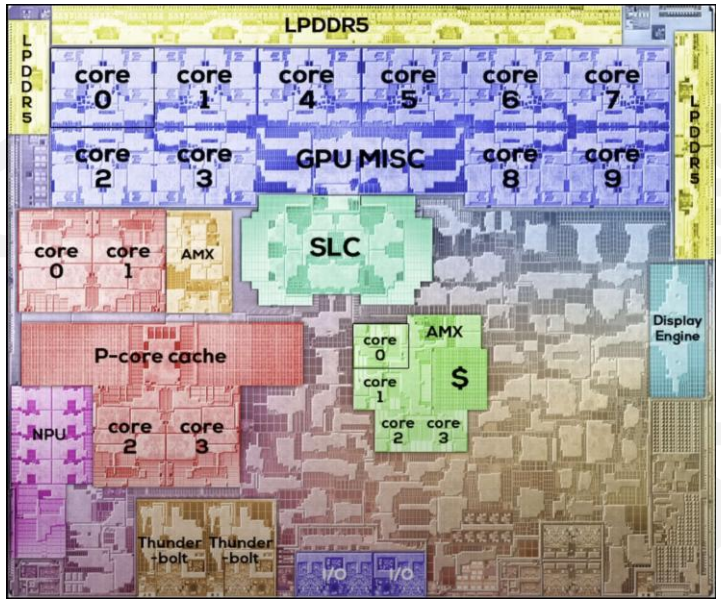
- 从偏向通用计算任务的CPU、GPU到偏向定制化设计的FPGA、ASIC

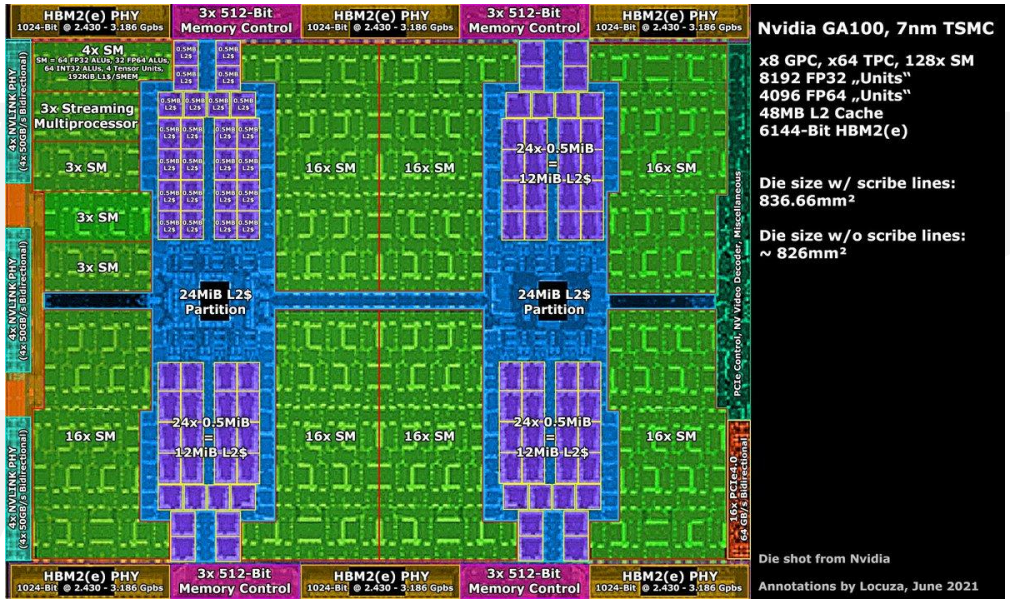


经典的DNN加速器ASIC体系结构

冯诺依曼体系结构

- 目前的成熟商用芯片基本均采用冯诺依曼体系结构





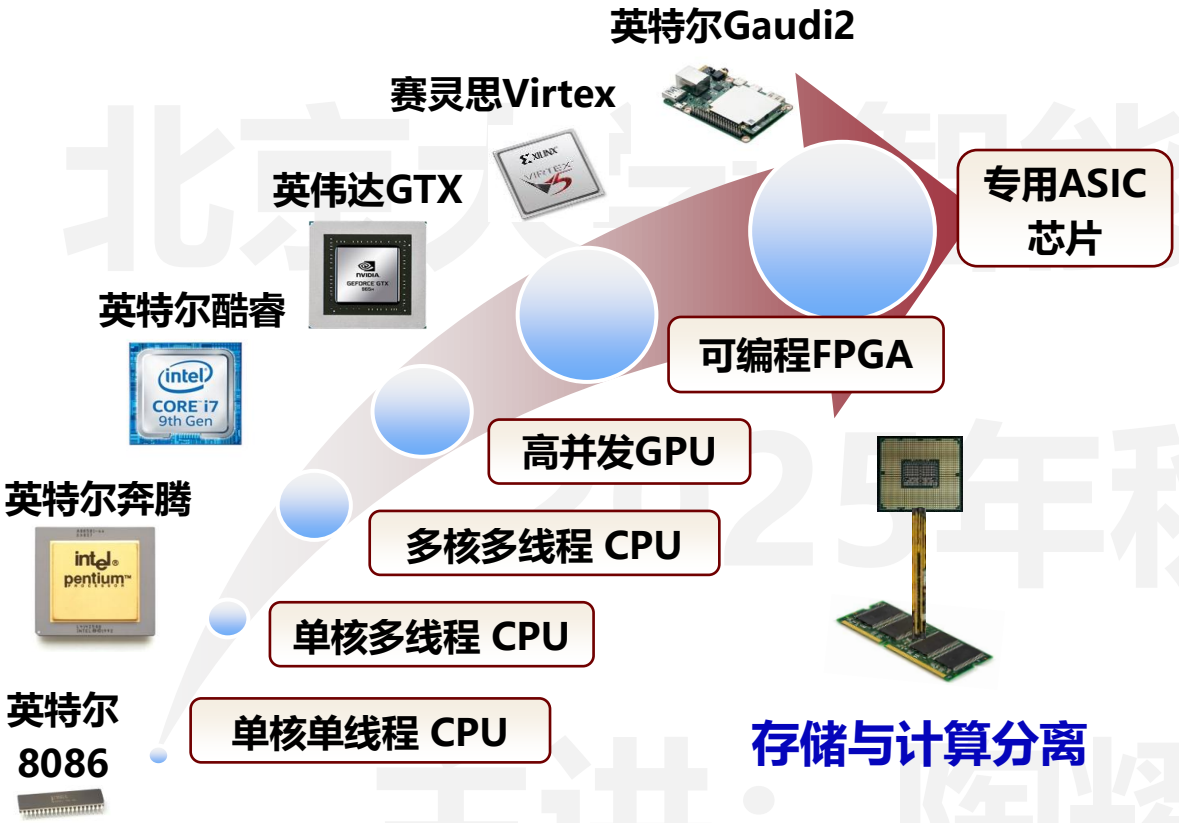
Apple M3 **NVIDIA H100**

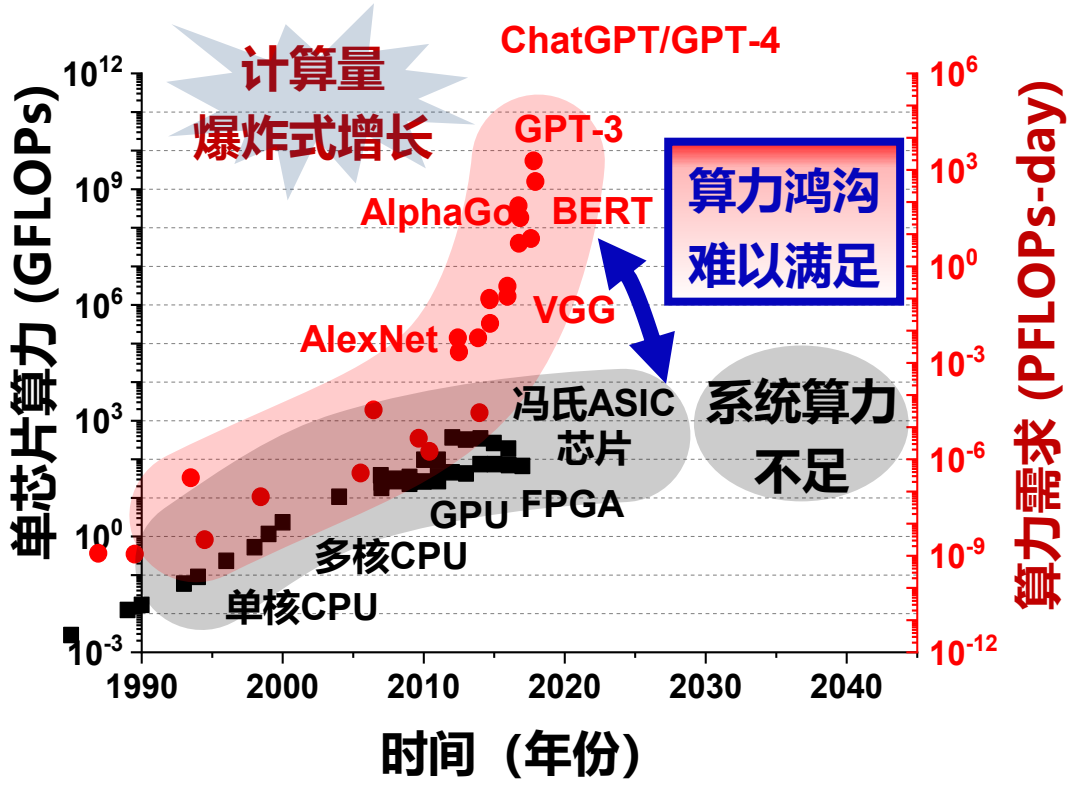
特点：存储与计算单元分离，依靠总线进行连接，

执行程序时需要来回搬运数据（读出→计算→写入）

冯诺依曼体系结构

- 从偏向通用计算任务的CPU、GPU到偏向定制化设计的FPGA、ASIC



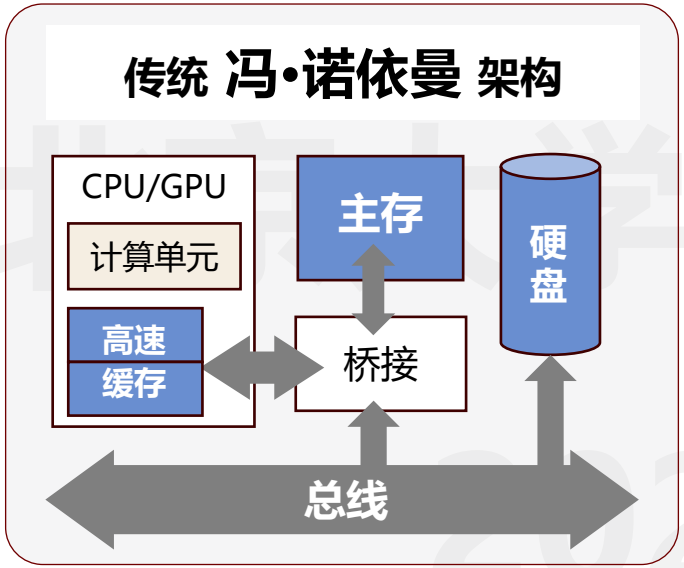


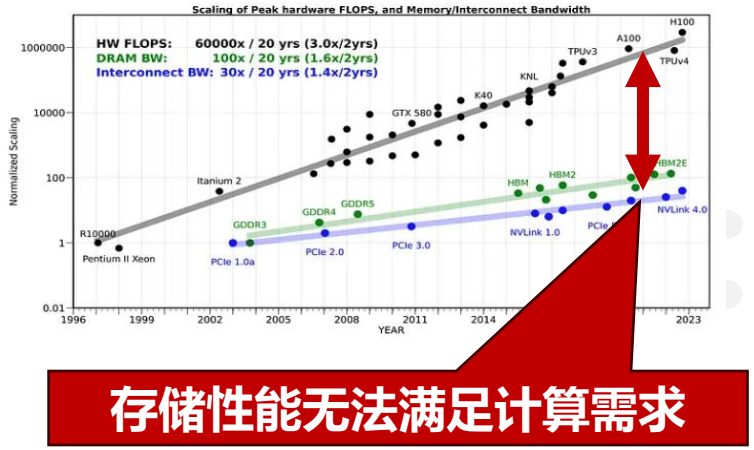
智能芯片体系结构演进图

新兴计算任务所需的计算量呈爆炸式增长

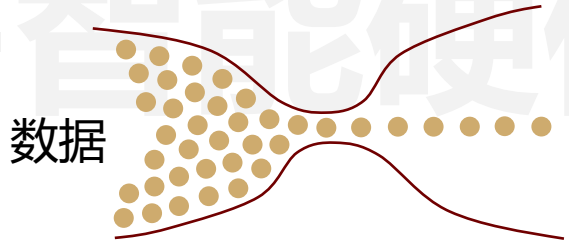
冯诺依曼体系结构

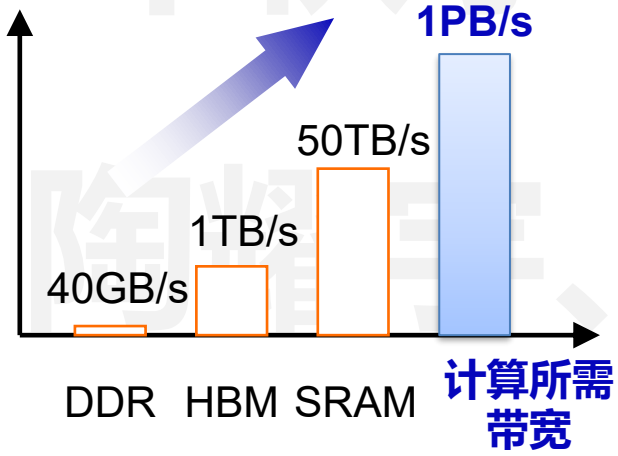
- 当前智能芯片体系结构的瓶颈



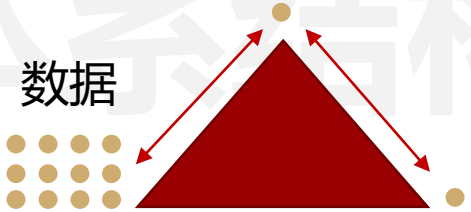


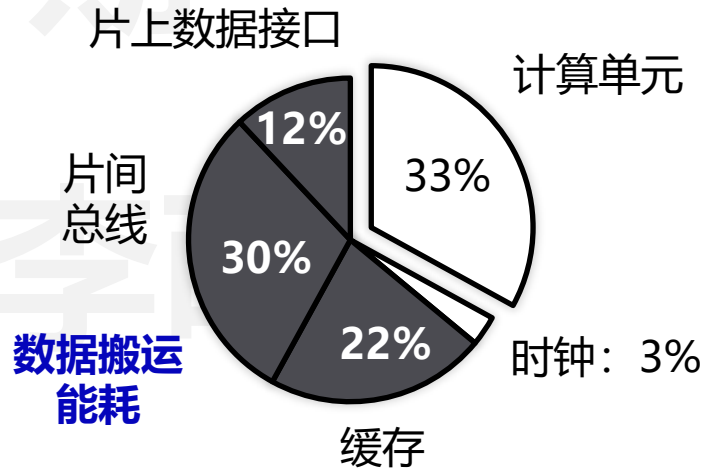
存储带宽限制算力





数据搬运限制能效

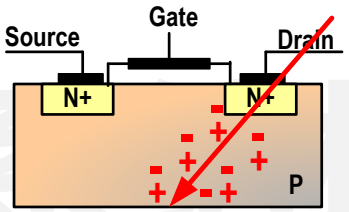




- 当前智能芯片体系结构的瓶颈

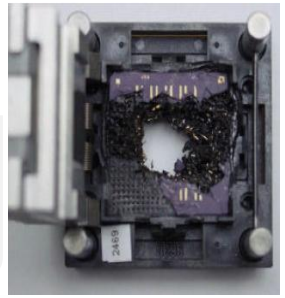
可靠性问题

Transient faults（瞬态故障）
- E.g, high-energy particle strikes



transients

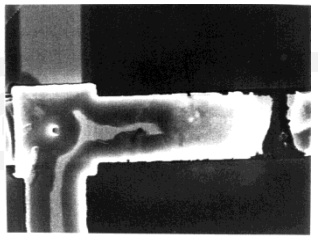
Manufacturing faults（制造缺陷）
- E.g., broken connections



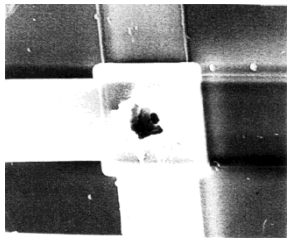
burn-in

Wearout faults（老化）
- E.g., Electromigration

interconnect



via

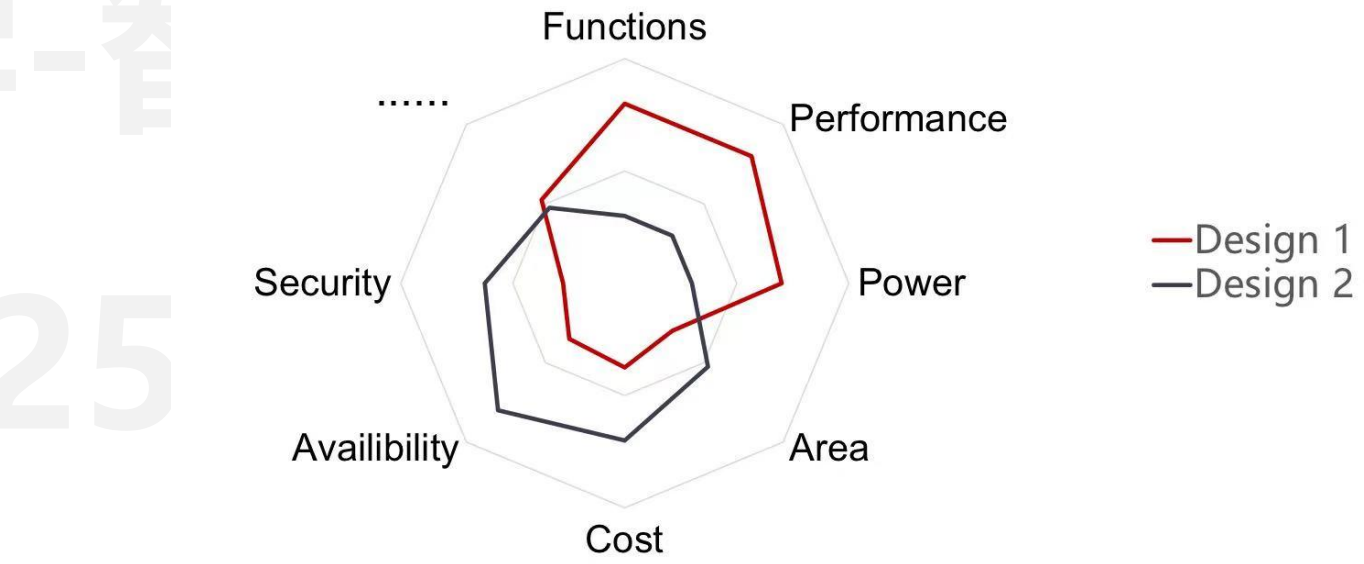


Device variability（器件涨落）
(not all transistors created equal)

体系结构基本概念

- 如何衡量一个芯片体系结构的性能

- **Functions**
- **Performance**
- **Power**
- **Area**
- **Cost**
- **Availability**
- **Security**
- **…**



体系结构基本概念 - 1

- **速度提升概念 – Amdahl Law**

加速比 = $time_{without\ enhancement}$ / $time_{with\ enhancement}$

Technique speeds up a fraction **f** of a task by a factor of **S**

$$time_{new} = time_{orig} \cdot ((1-f) + f/S)$$

$$S_{overall} = 1 / ((1-f) + f/S)$$

体系结构基本概念 - 2

- **并行处理的基本概念 - Parallelism law**

并行度 - the amount of independent sub-tasks available

Work=T_S - time to complete a computation on a sequential system（**完全串行需要的工作时间**）

Critical Path=T_∞ - time to complete the same computation on an infinitely-parallel system（**完全并行需要的工作时间**）

平均并行度 Average Parallelism

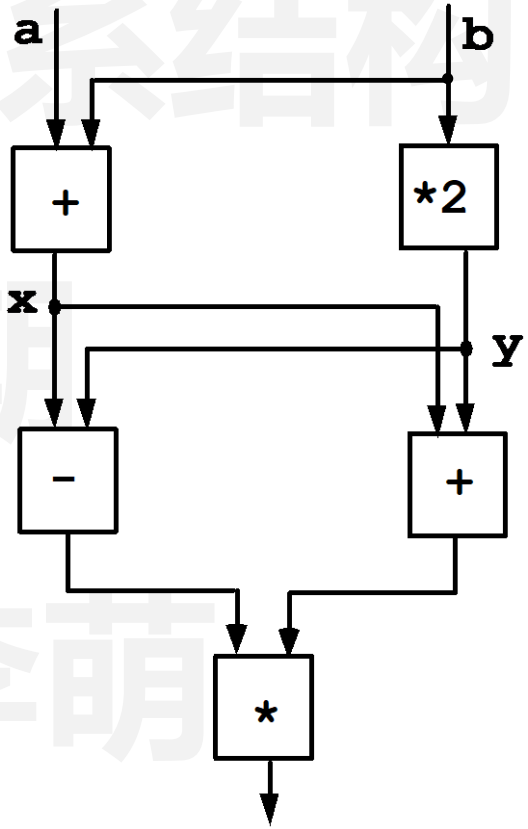
$$P_{avg} = T_S / T_\infty$$

For a p wide system（并行度P的硬件系统）

$$T_p \geq \max\{ T_S/p, T_\infty \}$$

$$P_{avg} >> p \Rightarrow T_p \approx T_S/p$$

```
x = a + b;
y = b * 2
z =(x-y) * (x+y)
```



体系结构基本概念 - 3

- **局部性原理的基本概念 – Locality law**

最近发生事件是近期未来发生时间最好的指标.

- **时域局部性（Temporal Locality）**: If you looked something up, it is very likely that you will look it up again soon
- **空间局部性（Spatial Locality）**： If you looked something up, it is very likely you will look up something nearby next

Locality == Patterns == Predictability

Converse:

Anti-locality : If you haven’t done something for a very long time, it is very likely you won’t do it in the near future either

体系结构基本概念 - 4

- 记忆与存储的基本概念 – Memoization law

Dual of temporal locality but for computation

如果某项计算很昂贵（硬件开销、时间、能耗等），那么最好的办法是记住答案一段时间，以备不时之需。

Why does memoization work??

Examples

- Trace caches

体系结构基本概念 - 5

- 处理开销平摊的基本概念 – Amortization law

  - overhead cost : one-time cost to set something up
  - per-unit cost : cost for per unit of operation

total cost = overhead + per-unit cost x N

如果处理可以分摊到大量单位上，通常高额overhead cost是可以接受

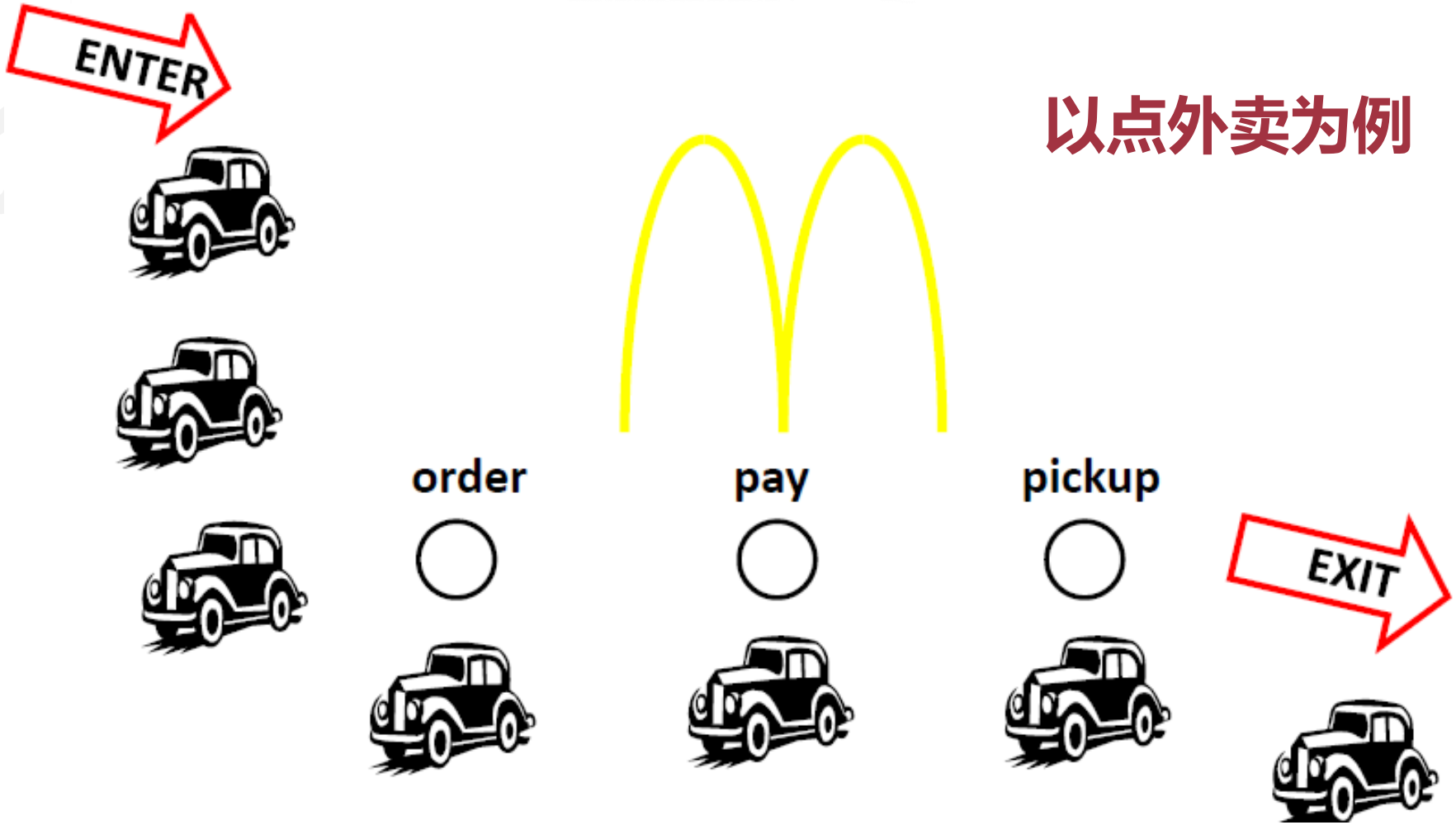
⇒ lower the *average cost*

average cost = total cost / N

= (overhead / N) + per-unit cost

体系结构基本概念 - 6

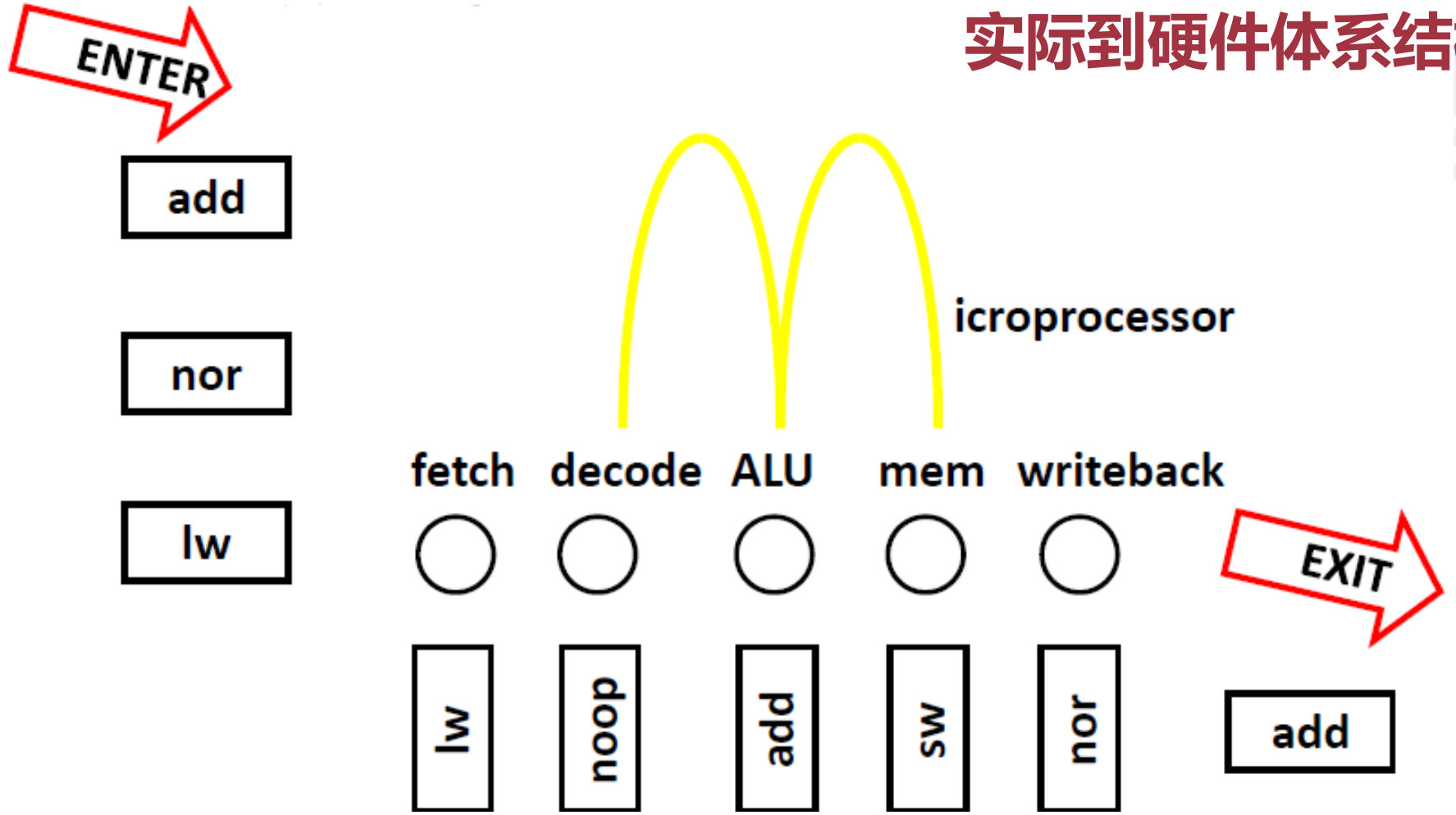
- 流水线处理的基本概念 – Pipeline law



以点外卖为例

体系结构基本概念 - 6

- 流水线处理的基本概念 – Pipeline law



体系结构基本概念 - 7

北京大学 PEKING UNIVERSITY

- 指令集的基本概念 – Instruction Set Architecture

- ISA (instruction set architecture)
  - 链接软件与硬件的接口
    - **Functional definition** of operations, modes, and storage locations supported by hardware
    - **Precise description** of how to invoke, and access them
  - 指令集并不确定内含的指令在硬件上的运行效率，只负责提供功能描述
    - Which operations are fast and which are slow and when
    - Which operations take more power and which take less

Type	Example Instruction
Arithmetic and logical	and, add
Data transfer	move, load
Control	branch, jump, call, return
System	trap, rett
Floating point	add, mul, div, sqrt
Decimal	addd, convert
String	move, compare

What operations are necessary?

What is the minimum complete ISA for a von Neuman machine?

Too little or too simple → not expressive enough

difficult to program (by hand)
programs tend to be bigger

Too much or too complex → most of it won't be used

too much "baggage" for implementation.
difficult choices during compiler optimization

体系结构基本概念 - 8

- 乱序并行的基本概念 – Out-of-Order Instruction Execution

```
code1:   r1 ← r2 + 1
         r3 ← r1 / 17
         r4 ← r0 - r3
```

```
code2:   r1 ← r2 + 1        (1)
         r3 ← r9 / 17       (2)
         r4 ← r0 - r10      (3)
         r11 ← r4*16        (4)
         r28 ← r11*r3       (5)
```

ILP: Instruction-Level Parallelism

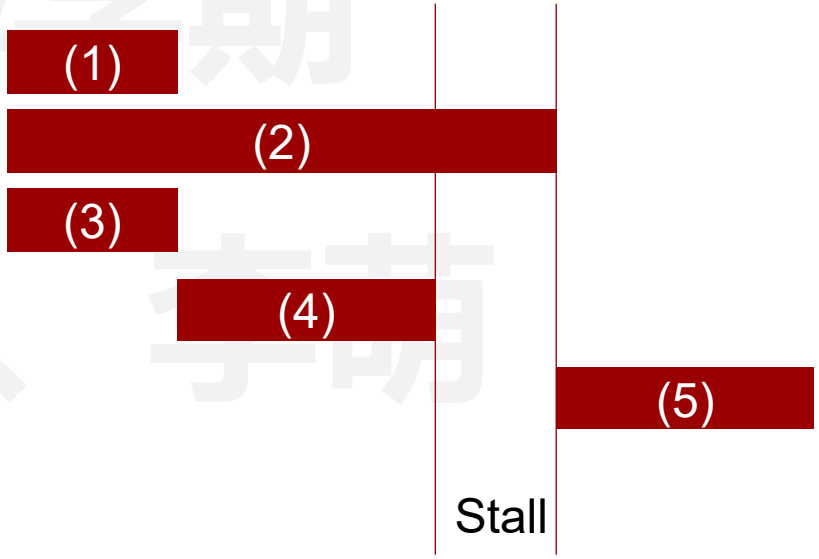
Average ILP = no. instruction / no. cyc required

code1: ILP = 1

i.e. must execute serially

code2: ILP = 3

i.e. can execute at the same time



体系结构基本概念 - 9

- 硬件性能指标的基本概念 – Iron law

Time (latency) 计算延时

- elapsed time vs. processor time

Rate (bandwidth or throughput) 计算速率

- performance = rate = work per time

$$\text{Processor Performance} = \frac{\text{Time}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Time}}{\text{Cycle}}$$

(code size) (CPI) (cycle time)

算法程序指标 体系结构指标 体系结构/电路

Architecture --> Implementation --> Realization

Compiler Designer Processor Designer Chip Designer

目 录

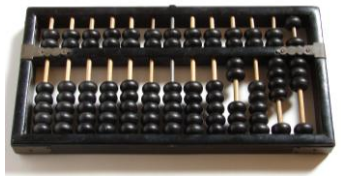
CONTENTS

智能芯片的计算能力是未来新的生产力

- 数据是新的生产资料，计算能力是新的生产力，是支撑科技发展的源动力

古代

基于人力的算力



算盘

18世纪以前

机械化时代

基于机械的算力

机械式计算机

18世纪-20世纪初

电气化时代

基于电气元件的算力

电子管计算机

20世纪上半叶

电子信息时代

基于集成电路的算力

晶体管计算机

1960 – 2030

大数据智能时代

云边端多样化算力

新器件、新架构计算机

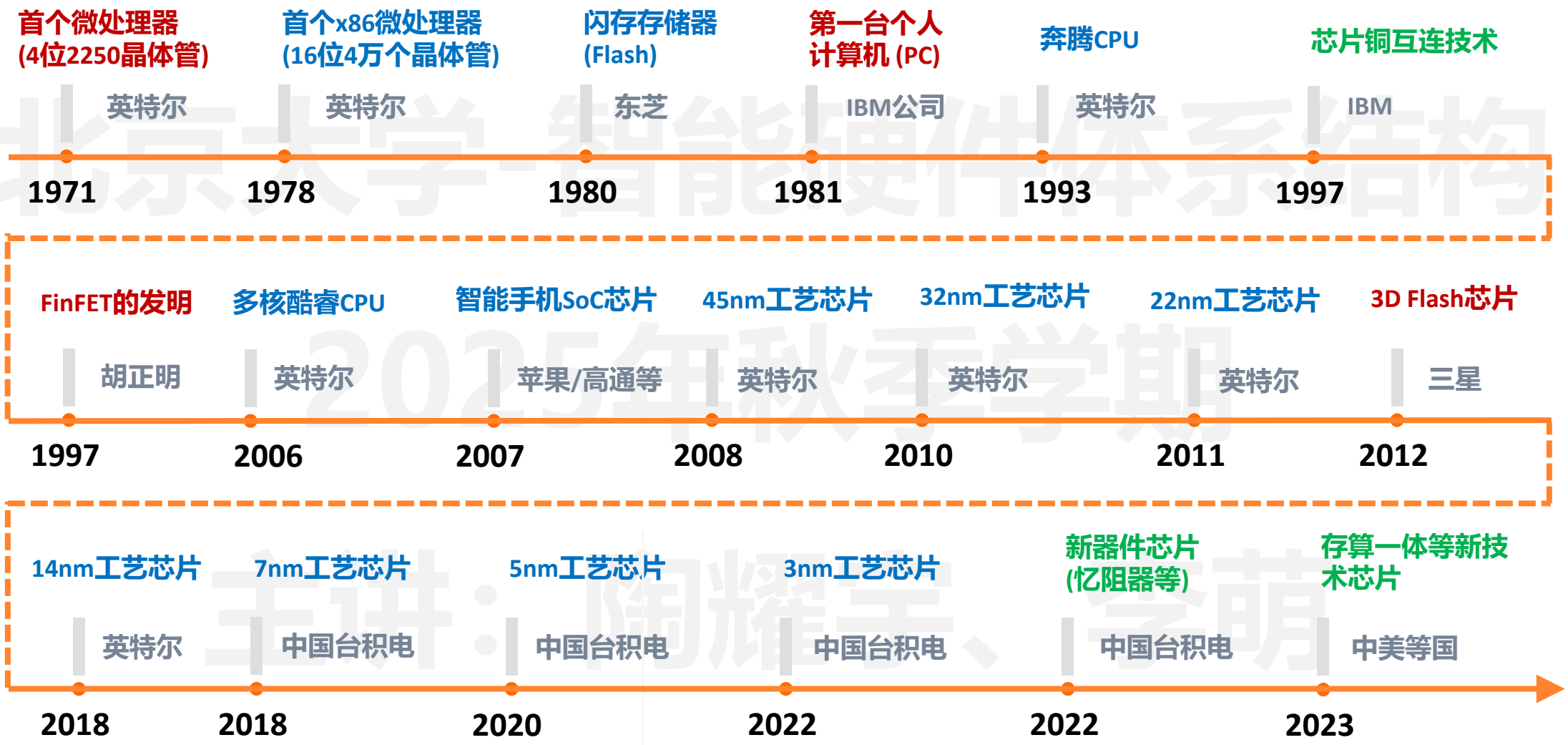
2030以后

波澜壮阔的智能芯片发展史

- 智能芯片的发展历史 (1833 - 1968)

波澜壮阔的智能芯片发展史

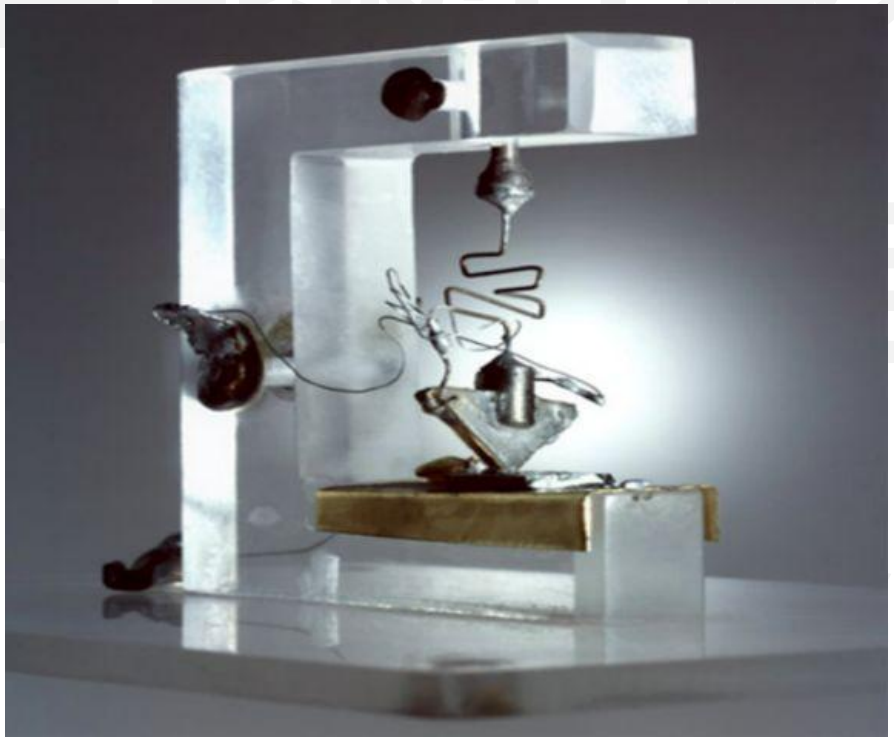
- 智能芯片的发展历史 (1968 - 2023)



芯片发展的重要历史节点：半导体锗晶体管的发明 – 1947年

- 半导体晶体管被誉为“21世纪最伟大的发明”，深刻的改变了人类历史发展进程

肖克利（前）、巴丁（后一）、布拉顿（后二），因为晶体管的发明，共同获得了1956年的诺贝尔物理学奖



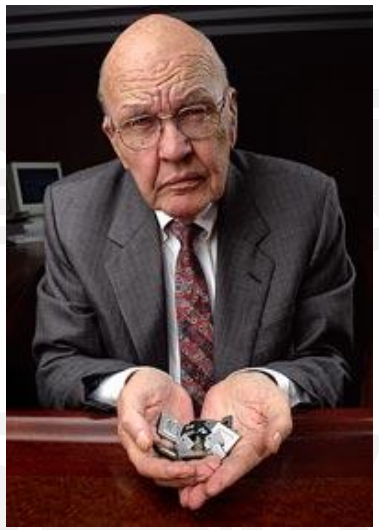
点接触式晶体管：把间距为50 μm的两个金电极压在锗半导体上，微小的电信号由一个金电极 (发射极) 进入锗半导体 (基极) 并被显著放大，然后通过另一个金电极 (集电极) 输出，这个器件在1kHz的增益为4.5

重要历史节点：集成电路的发明 – 1958年/1959年

- 德州仪器公司的工程师基尔比 (Jack Kilby) 发明了第一块集成电路

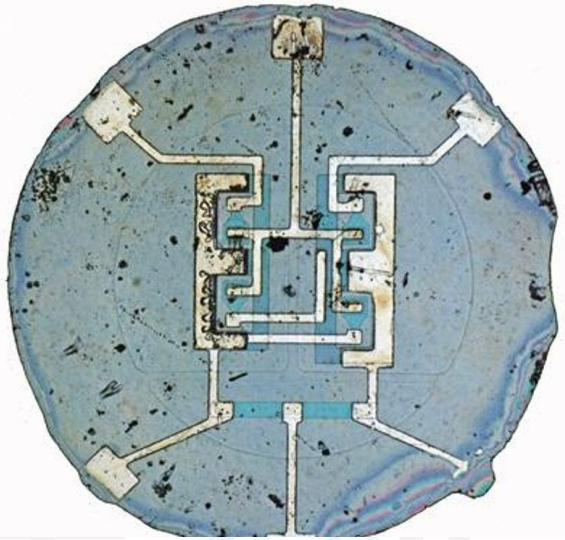
1958年8月28日世界第一块集成电路

尺寸7/16×1/16英寸



基尔比获2000年

诺贝尔奖



罗伯特-诺伊斯于1959年8月发明第一块

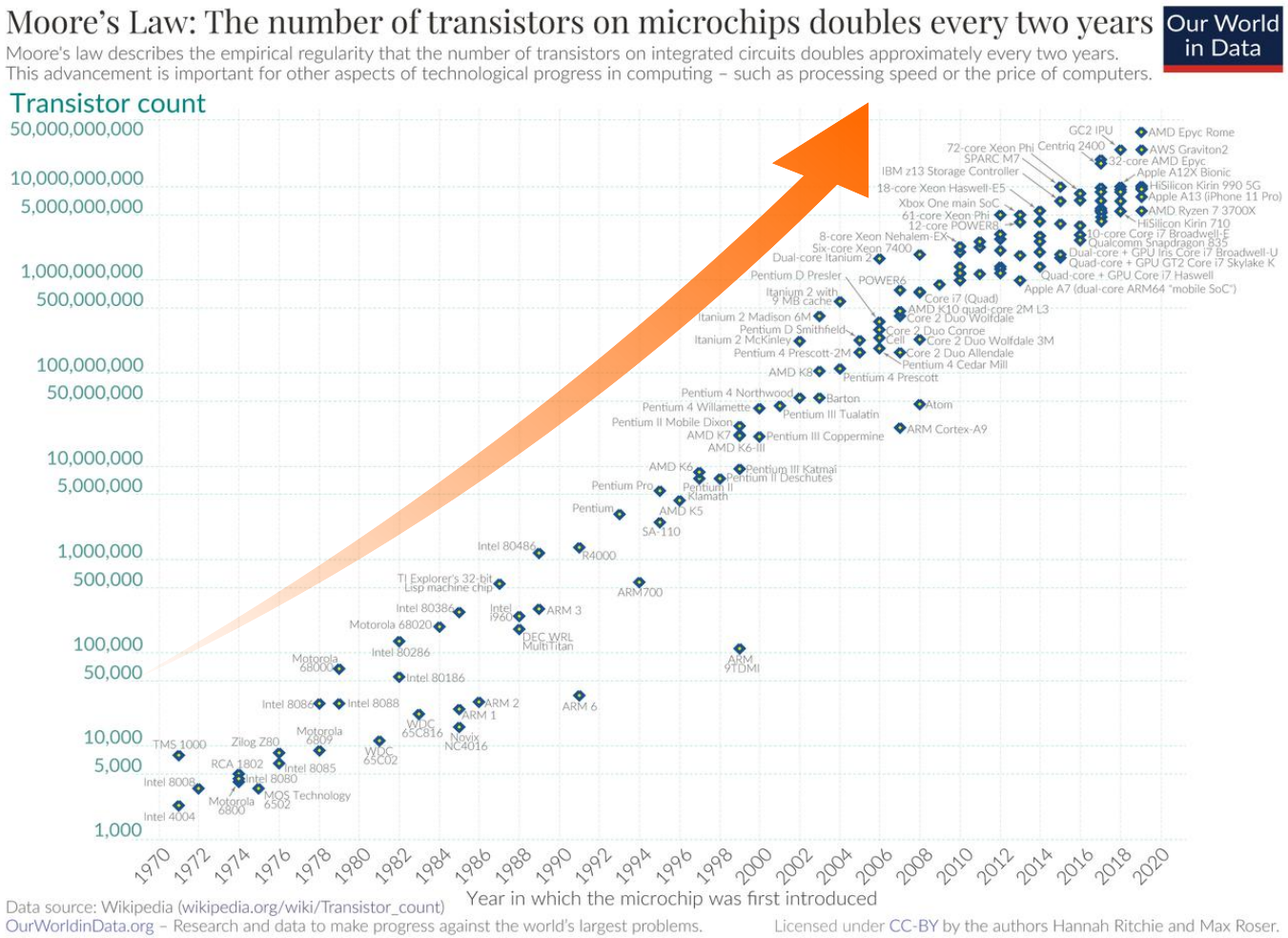
硅集成电路

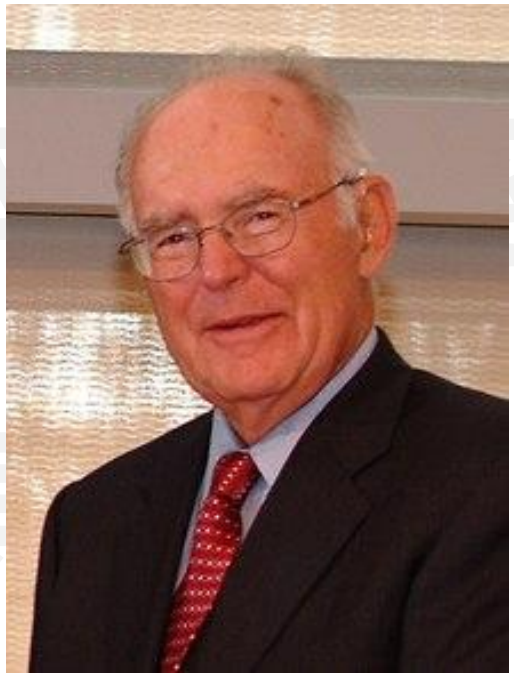
将包括锗晶体管在内的五个元器件集成在一起，基于锗材料制作了一个叫做相移振荡器的简易集成电路

参与创立仙童半导体 (Fairchild) 和英特尔 (Intel) 公司，奠定了硅谷的基石

重要历史节点：摩尔定律的提出 – 1964年

- 仙童半导体/英特尔的联合创始人戈登摩尔提出了著名的“摩尔定律”





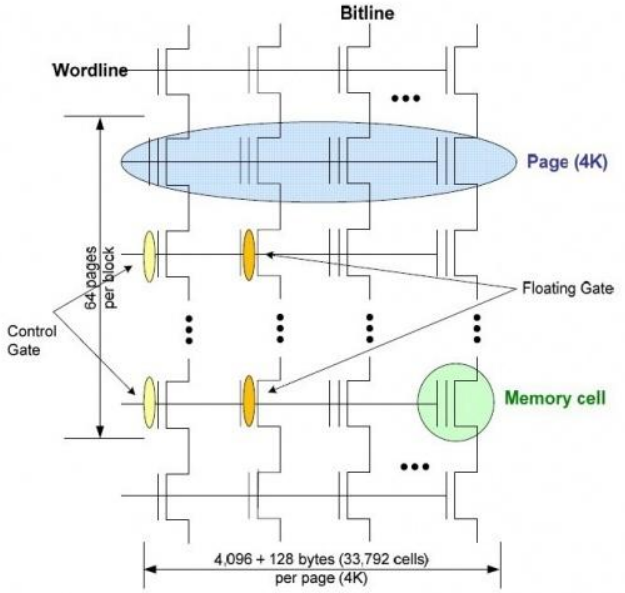
戈登·摩尔

集成电路上可容纳的晶体管数目，

每隔两年便会增加一倍

重要历史节点：非易失性存储器Flash的发明 – 1967年

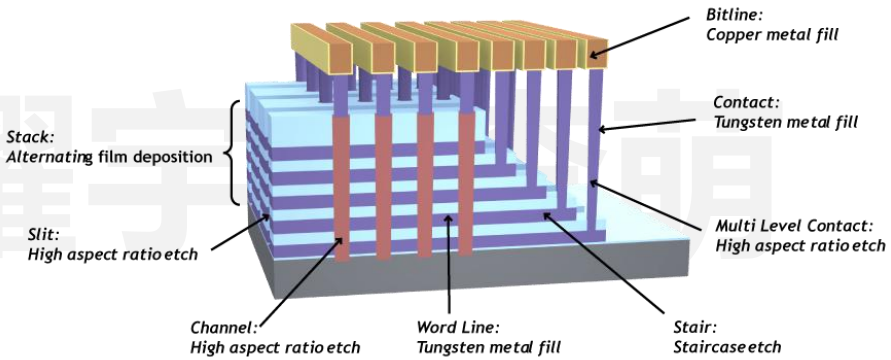
- 除了计算场景之外，存储也是占据智能芯片重要份额的典型应用场景



传统平面型NAND Flash非易失性存储器

Dawon Kahng (韩) 和 Simon Sze (华裔) 在贝尔实验室发明了非易失性存储器浮动门（Floating Gate ）

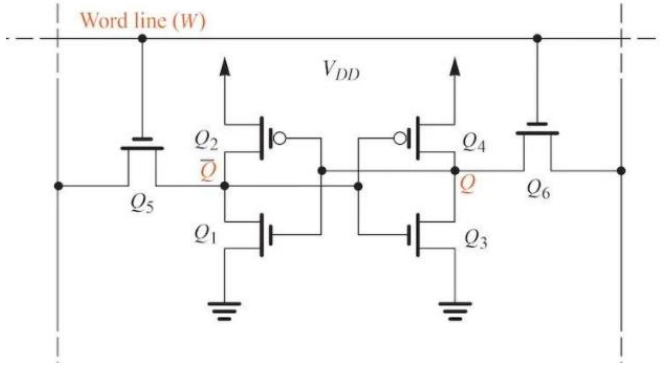
本文发表为“A Floating Gate and Its Application to Memory Devices” （贝尔系统技术期刊）

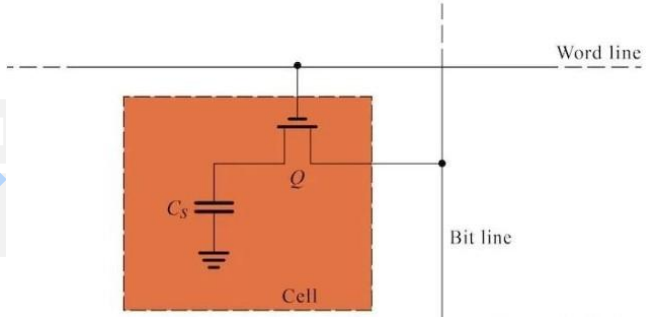


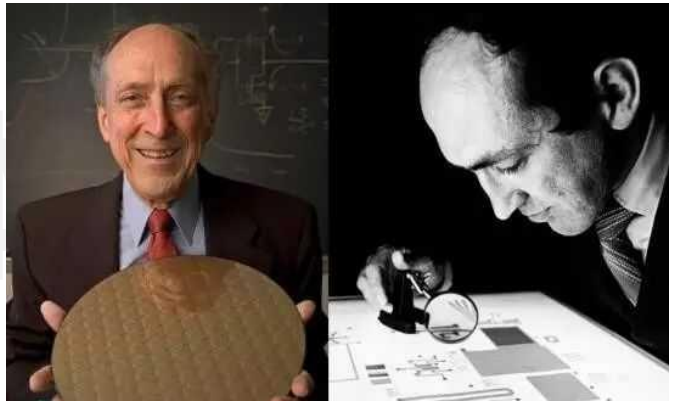
现代三维堆叠 NAND Flash

重要历史节点：易失性存储器DRAM的发明 – 1968年

- SRAM/DRAM是两种最常用的易失性存储器件，广泛应用于现代智能芯片中







SRAM需要6个CMOS晶体管来存储数据

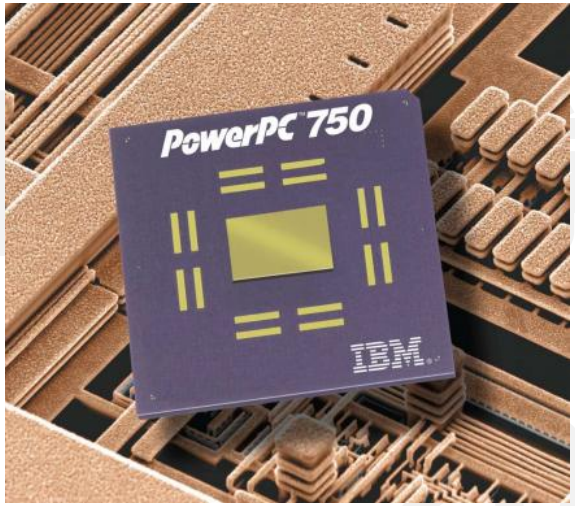
罗伯特·丹纳德发明了DRAM (**动态随机存取存储器**) 存储器

SRAM（**静态随机存取存储器**）的优点是它的速度快，它的存取速度比DRAM（动态随机存取存储器）快得多，因为它不需要每次访问数据都要重新刷新电容。

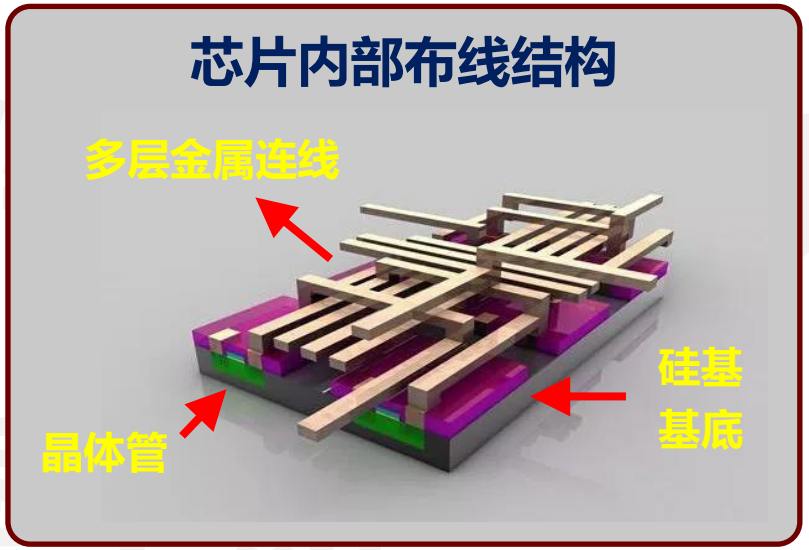
与SRAM相比，DRAM的优势在于结构简单—**每比特都只需一个电容跟一个晶体管来处理**，相比之下在SRAM上一个比特通常需要六个晶体管。 正因这缘故，**DRAM拥有非常高的密度，单位体积的容量较高因此成本较低**。 但相反的，DRAM也有访问速度较慢，耗电量较大的缺点。

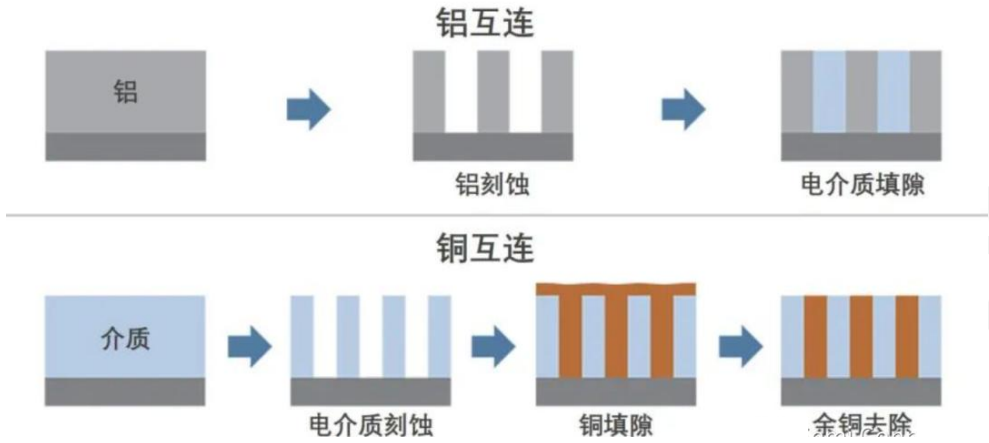
重要历史节点：智能芯片的铜互连技术 – 1997年

- IBM率先从铝互连转向铜互连，并推出了第一个铜基微处理器 IBM PowerPC 750



IBM PowerPC 750 最初是采用铝设计的，其工作频率高达300 MHz，采用铜互连之后，同一芯片的速度至少能达到400MHz，提高了33%

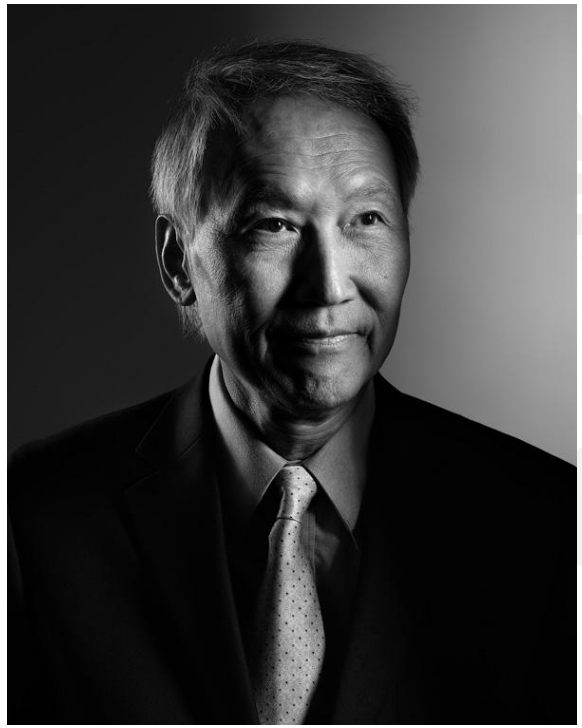




集成电路金属互连线制造工艺达到纳米级后，因为超高纯铜具有更佳的电阻率和抗电迁徙能力，很快高纯铜就替代超高纯铝合金成为金属互连线的主要材料

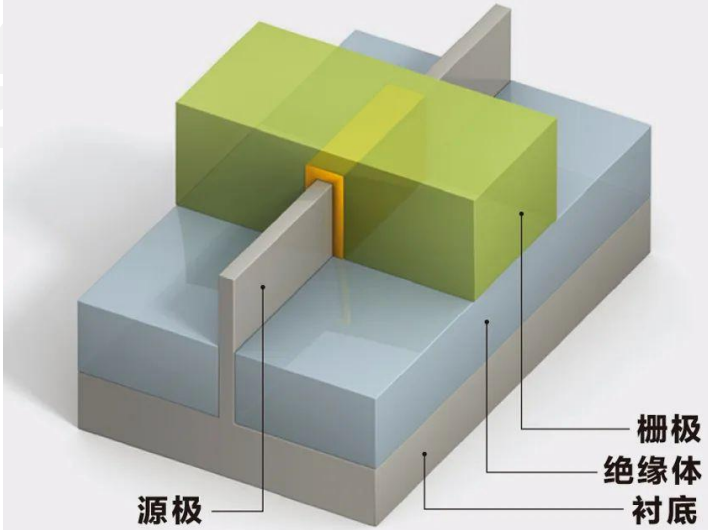
拯救摩尔定律的发明：鳍式三维晶体管FinFET – 1999年

- 原本预计2010年后，传统CMOS工艺技术在20nm走到尽头，胡正明的发明拯救了摩尔定律

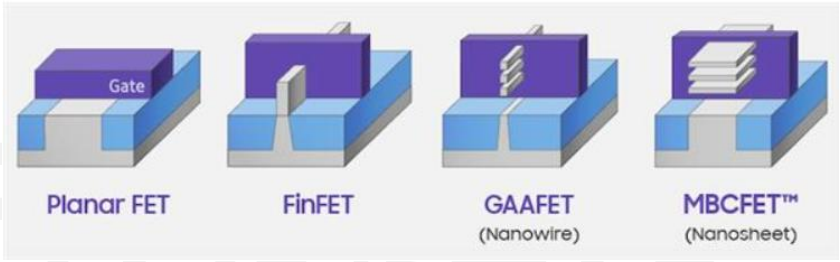


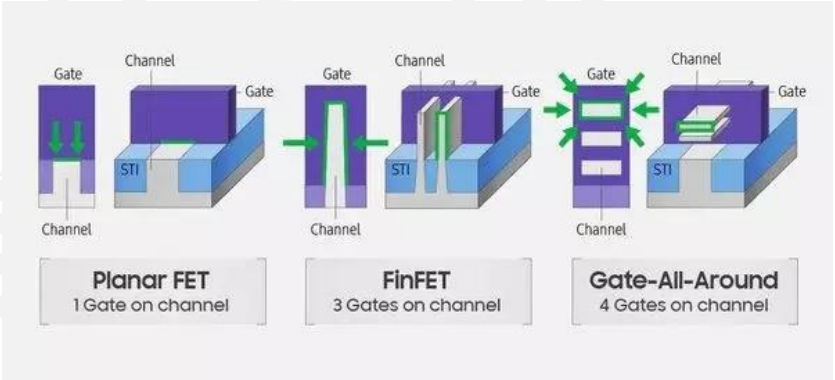
加州大学伯克利分校的胡正明教授

(IEEE Fellow，美国工程院院士，中国科学院外籍院士)



FinFET的特点：每个晶体管包含一个源极、一个漏极、一个连接两者的导电通道和一个控制电流沿通道流动的栅极。在FinFET中，将通道抬高，如同鲨鱼鳍高出芯片表面，使得栅极三面环绕，从而给予栅极更大的控制力。

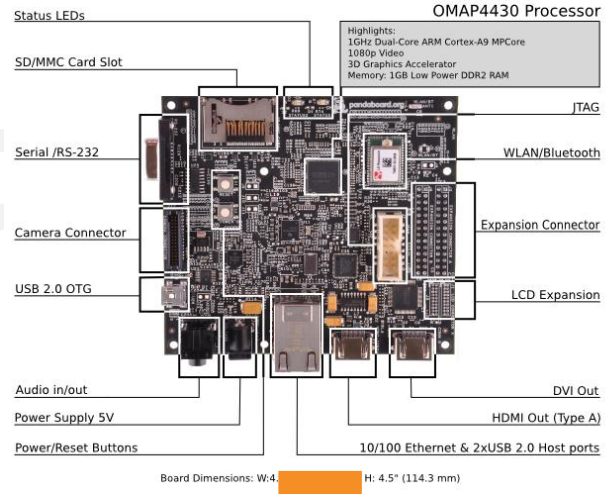




由FinFET演化出多种三维晶体管构型，推动制程向3nm/1nm演进

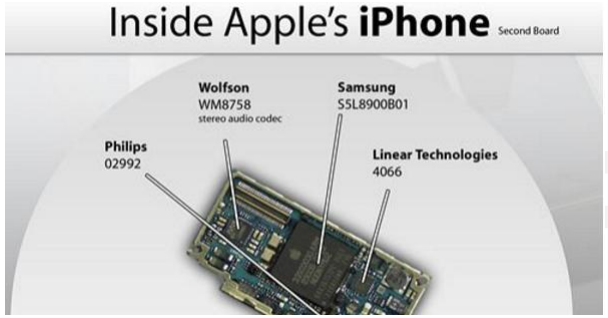
推动移动互联网飞速发展：移动SOC芯片 – 2007年至今

- 移动电话SOC芯片成为推动移动互联网飞速发展的算力基石，引领过去十几年的技术革命



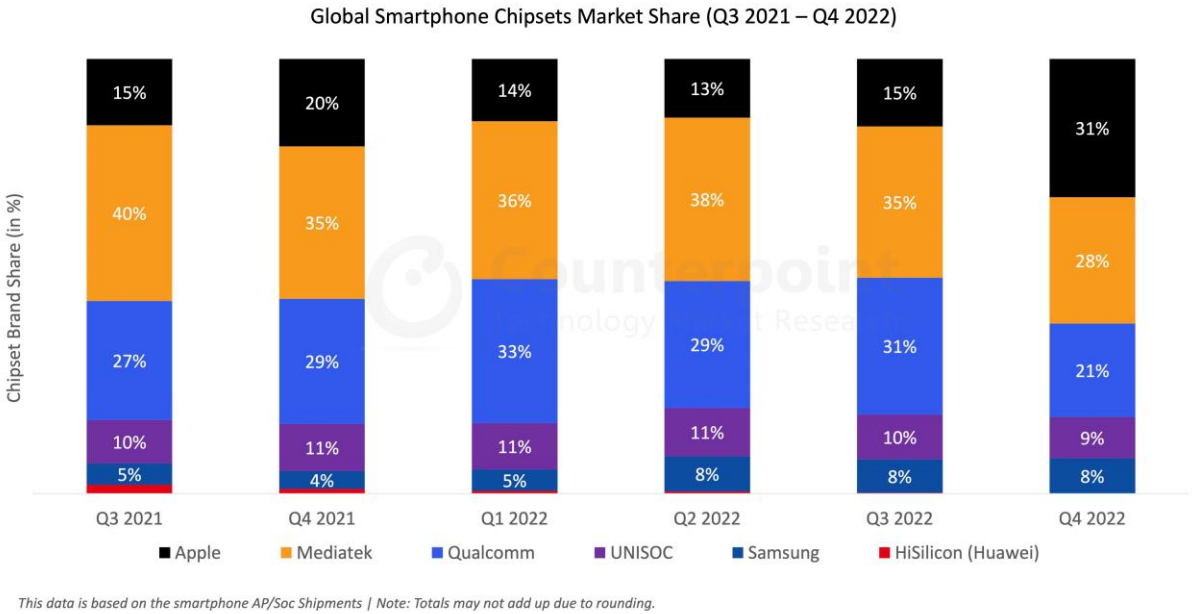
德州仪器OMAP手机芯片

诺基亚 6630、 6680、6681、E50、E60、E61、E62、E65、E70、N70、N71、N72、N73、N80、N90、N91和N92 等



三星S5L8900 SOC芯片

2007年乔布斯发布了第一代iPhone采用90nm制程三星SOC芯片



苹果、高通、联发科、三星、紫光展锐、华为海思占据移动SOC市场的前列

推动制程不断向前发展：中国台湾台积电/英特尔/三星 – 2008年至今

北京大学 PEKING UNIVERSITY

- 过去十几年，中国台湾积体电路公司、英特尔公司、三星公司是推动芯片制程发展的主要力量

全球主要晶圆厂制程节点技术路线图

晶圆代工厂	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024
台积电	28nm			20nm	16nm		10nm	7nm	7nm +	5nm 6nm		3nm		2nm
三星		28nm		22nm	14nm		10nm	8nm	7nm EUV 6nm	5nm	3nm			
英特尔	22nm			14nm		14nm +	14nm ++		10nm	10nm +	7nm 10nm ++	7nm +	7nm ++	
格罗方德			28nm		14nm		12nm							
联电				28nm			14nm							
中芯国际	40nm				28nm				14nm					

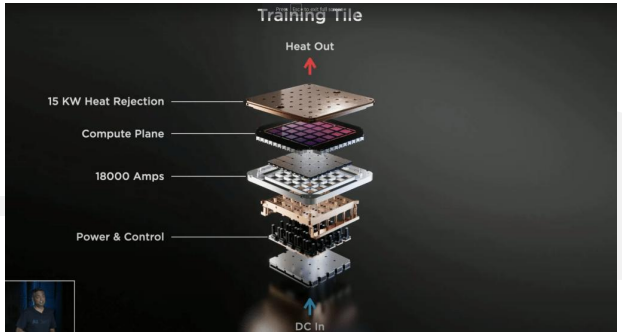
备注：以上信息整理自网络，如有错漏欢迎指正。

中国台湾积体电路公司后来追上，超越英特尔与三星

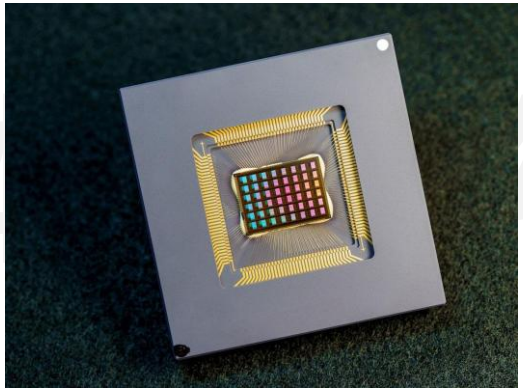
推动智能时代飞速发展：AI芯片 – 2014/2015年至今

- 高性能AI芯片成为推动智能时代发展的算力基石，将引领未来十几年的技术革命

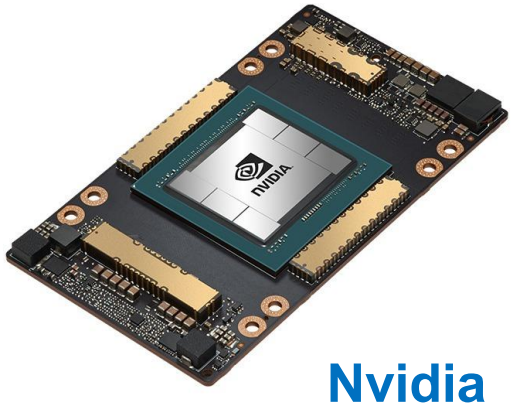
Google
TPU



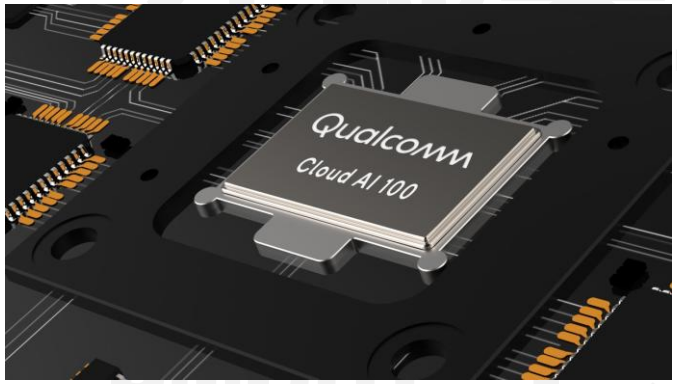
Tesla Dojo



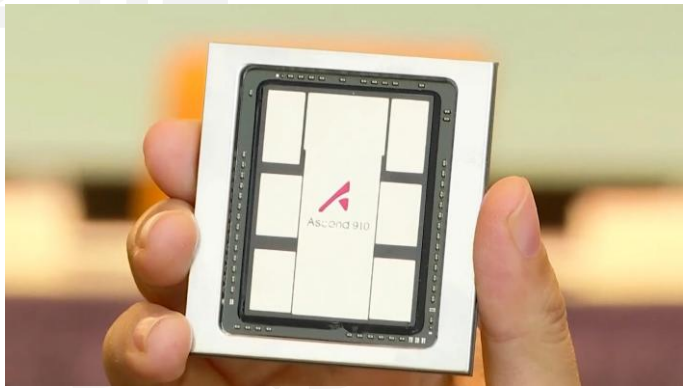
新器件AI芯片



Nvidia
GPU



Qualcomm Cloud AI



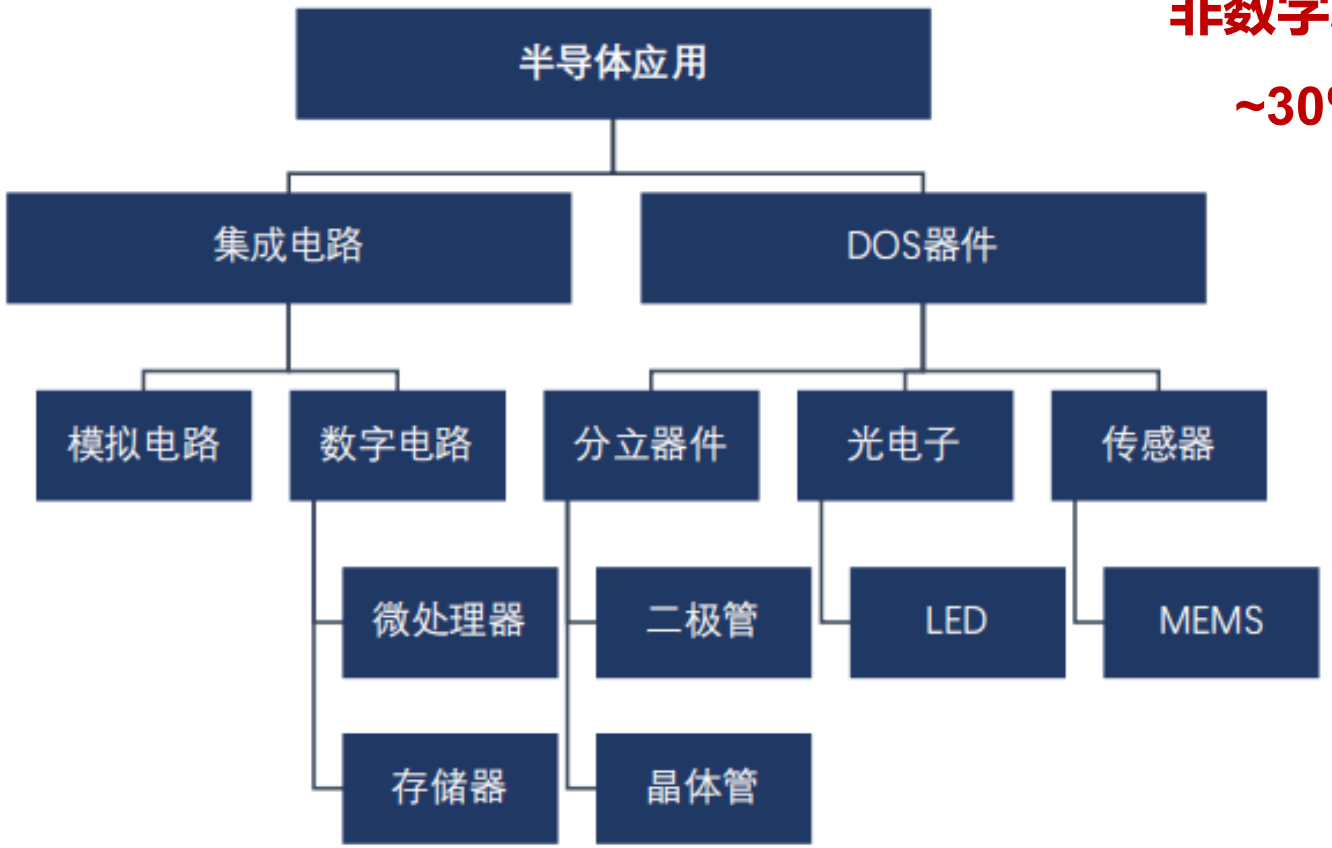
华为昇腾

目 录

CONTENTS

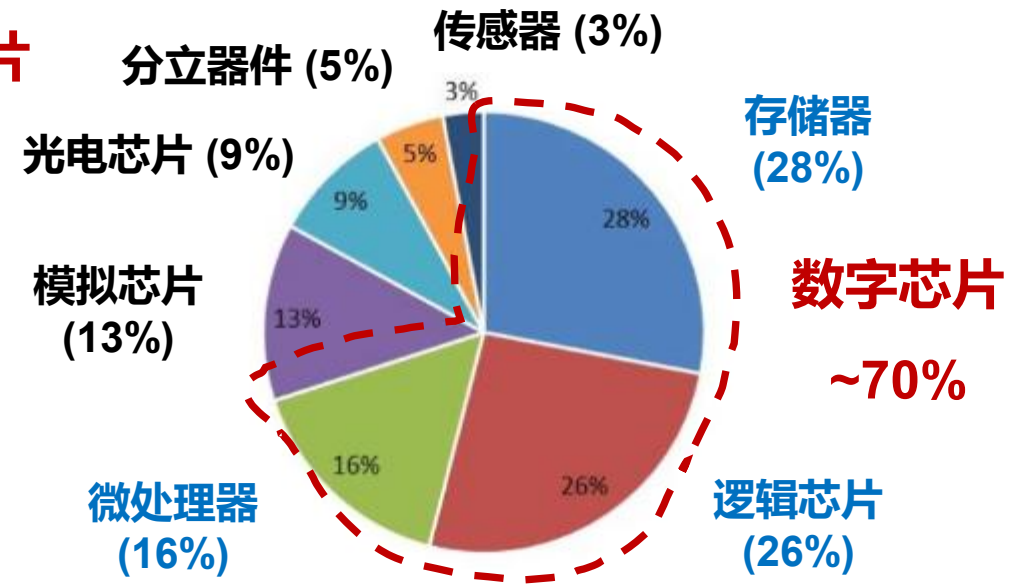
智能芯片产业按半导体应用分类

- 集成电路可分为模拟电路和数字电路，DOS器件分为光电子、传感器等

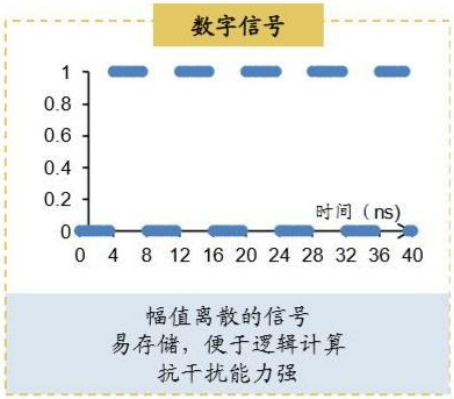


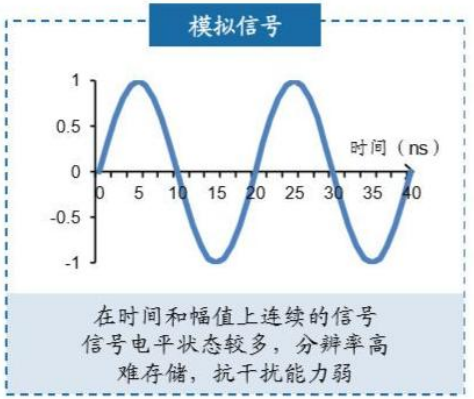
非数字芯片

~30%



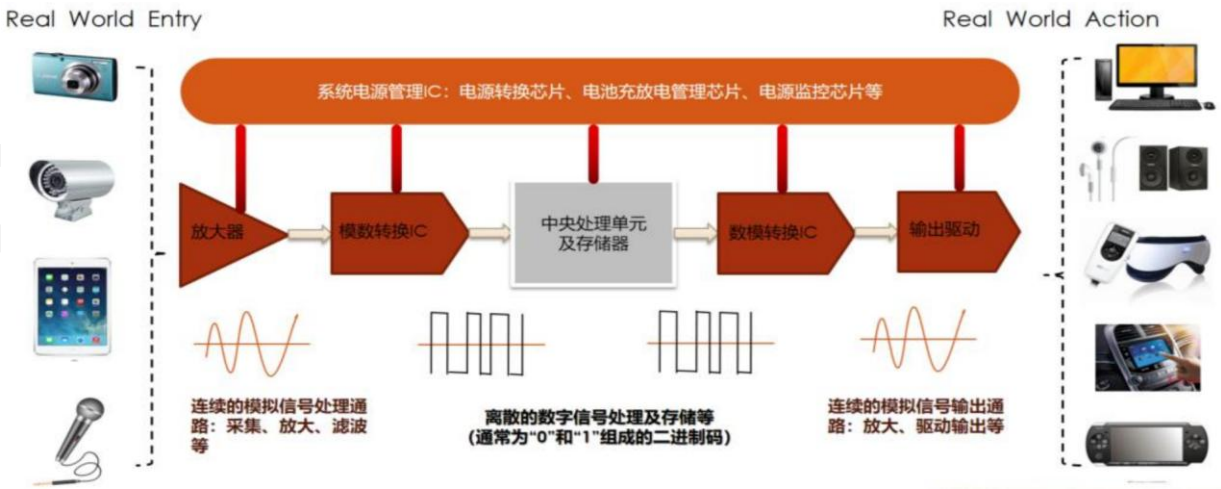
数字电路的市场份额占70%



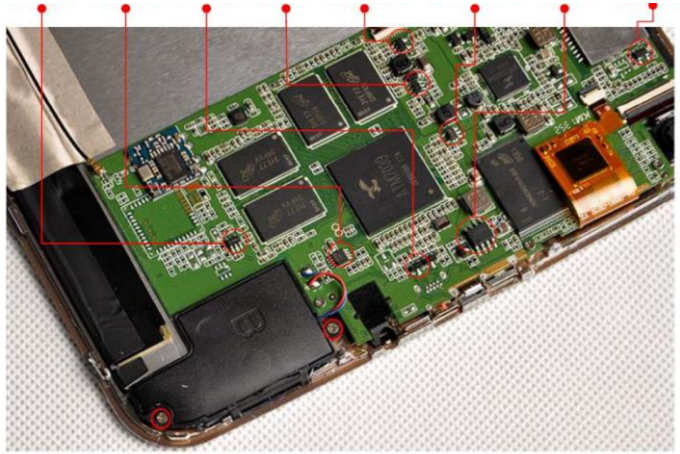


智能芯片产业按半导体应用分类

- 集成电路可分为模拟电路和数字电路，DOS器件分为光电子、传感器等

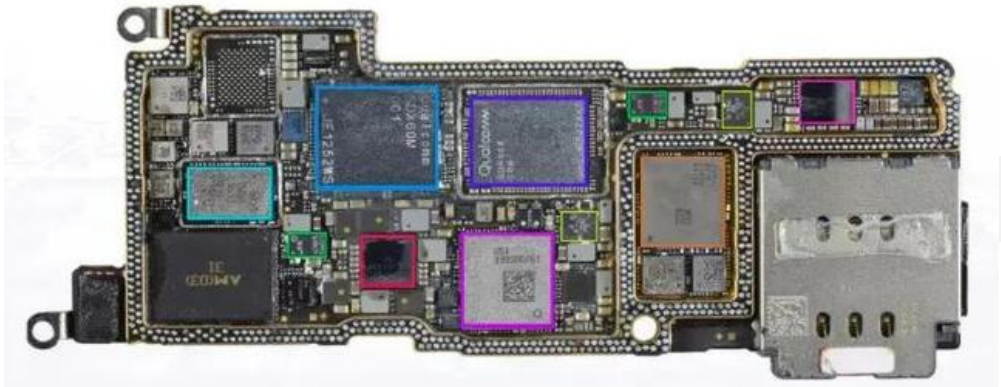


功率放大、电源管理、时钟生成、比较器、射频滤波、接口、数模转换、线性稳压等

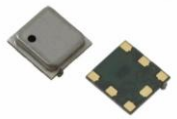


模拟芯片应用实例

CPU、GPU、存储器芯片、可编程逻辑芯片、MCU、DSP、NPU等



数字芯片应用实例



传感芯片 声、光、电、热、磁、压力、气体、震动、速度、湿度、惯性、流量、电磁波等

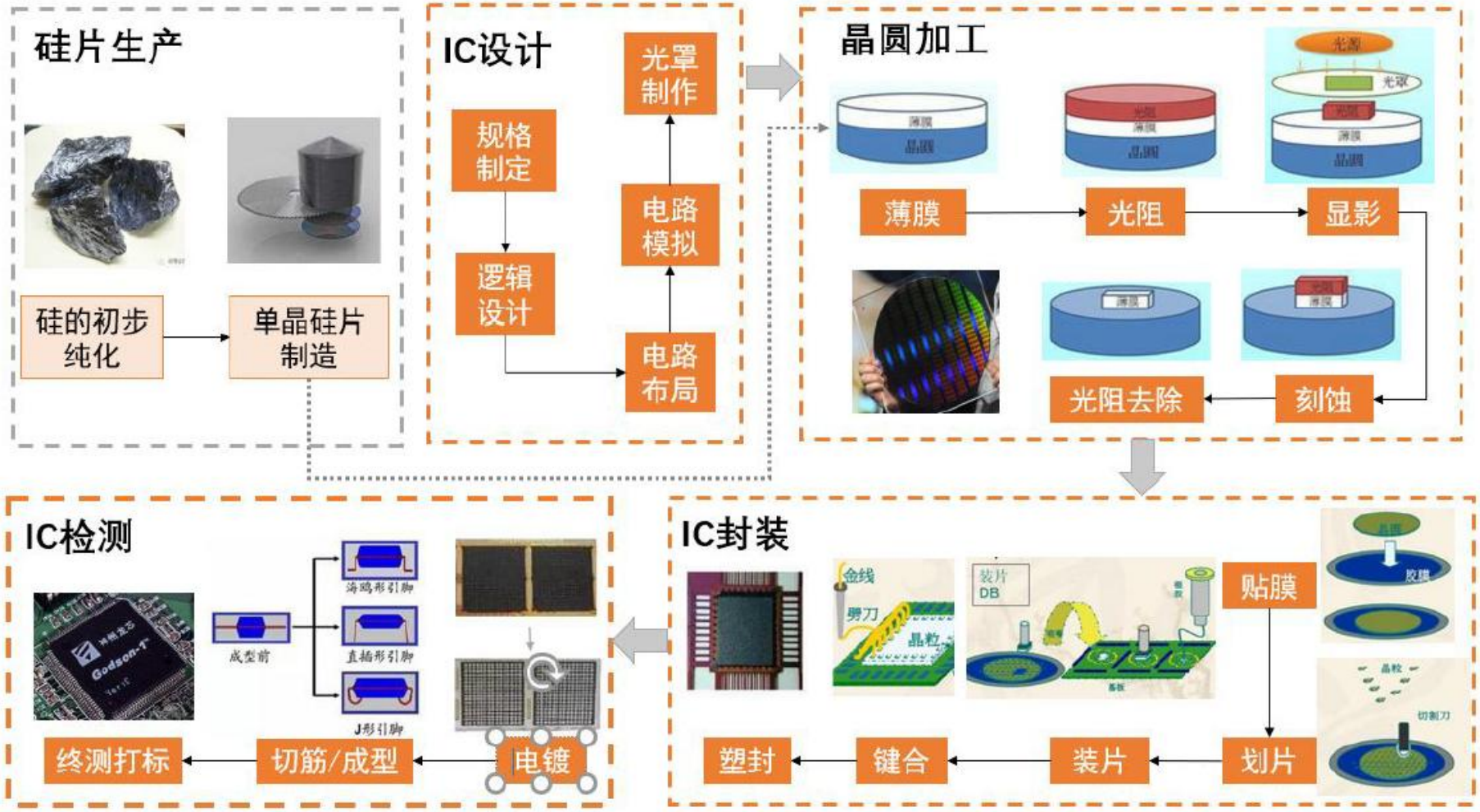
光电芯片 激光器芯片、半导体发光芯片等

分立器件 电阻、电容、电感、振荡器、晶体管、功率器件等

智能芯片产业现状 – 产业链极长、关联几乎所有工业门类

- 国际分工合作的庞大产业链生态

中国与世界先进水平差距较大



硅片生产企业
- 信越化学 (日本)
- 三菱住友 (日本)
- 环球晶圆 (台湾)

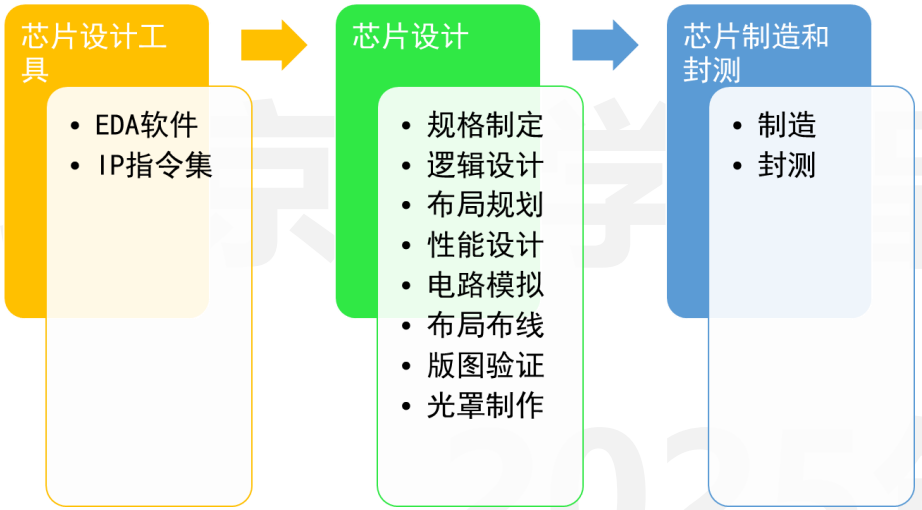
晶圆加工企业
- 台积电 (台湾)
- 三星 (韩国)
- 格芯 (美国)

芯片设计企业
- Intel (美国)
- Qualcomm (美国)
- 海思半导体 (中国)

芯片封测企业
- 日月光 (台湾)
- 安靠 (美国)
- 长电 (中国)

智能芯片产业现状 – 产业链极长、关联几乎所有工业门类

- 国际分工合作的庞大产业链生态



芯片设计的EDA软件工具目前由美国公司所垄断

芯片设计工具	芯片设计	芯片制造和封测
• 美国英特尔	• 美国苹果	• 美国英特尔
• 英国ARM	• 韩国三星	• 韩国三星
• 英国Imagination	• 英国Imagination	• 美国高通
• 美国IBM	• 海思	• 美国英伟达
• 美国Mips	• 紫光展锐（展讯+锐迪科）	• 台积电
• Synopsys	• 中兴微电子	• 中芯国际
• 美国Cadence	• 华大半导体	• 华虹半导体
• 美国Mentor Graphics	• 汇顶科技	• 长江存储
• 日本图研	• 兆易创新	• 合肥长鑫
• 华大九天	• 寒武纪	• 福建晋华
• 杭州中天微		• 紫光国微
• 苏州国芯		• 长电科技
• 寒武纪		• 华天科技
		• 通富微电

智能芯片产业的三种运作模式

- IDM (垂直整合)、Fabless (纯设计) 和 Foundry (晶圆加工)

	IDM模式	Fabless模式	Foundry模式
典型厂商	SAMSUNG intel	QUALCOMM BROADCOM	tsmc SMIC
基本特点	集芯片设计、制造、封测于一身。早期多数集成电路企业采用的模式，目前仅有极少数企业能够维持	只负责芯片的电路设计与销售；将生产、测试、封装等环节外包	不负责芯片设计，只负责制造或封测；可以同时为多家设计公司提供服务，但受制于公司间的竞争关系
主要优势	设计、制造等环节协同优化，有助于充分发掘技术潜力；能有条件率先实验并推行新的半导体技术	资产较轻，初始投资规模小，创业难度相对较小；企业运行费用较低，转型相对灵活	不承担由于市场调研不准、产品设计缺陷等决策风险。
主要劣势	公司规模庞大，管理成本较高；运营费用较高，资本回报率偏低	与IDM相比无法与工艺协同优化，因此难以完成指标严苛的设计；与Foundry相比需要承担各种市场风险	投资规模较大，维持生产线正常运作费用较高；需要持续投入维持工艺水平，一旦落后追赶难度较大。

早期企业都是IDM运营模式（垂直整合），这种模式涵盖设计、制造、封测等整个芯片生产流程，这类企业一般具有规模庞大、技术全面、积累深厚的特点，如Intel、三星等

随着专注于晶圆加工的台积电的出现，演化出Fabless和Foundry模式，专攻设计或者制造，各司其职

目 录

CONTENTS

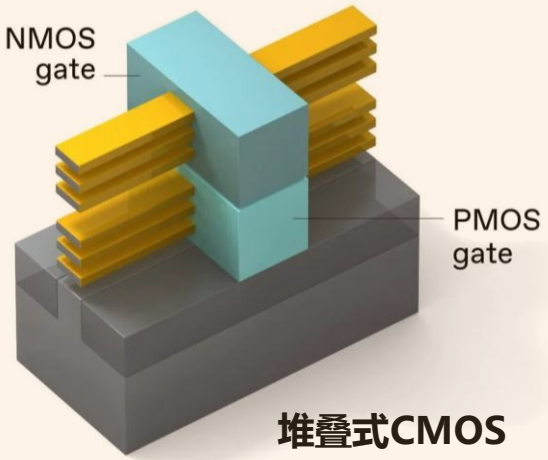
融合新器件、新架构、新计算是后摩尔时代体系结构的发展趋势

北京大学 PEKING UNIVERSITY

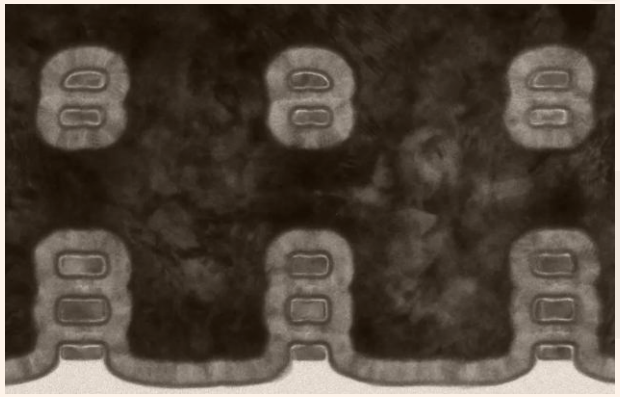
- 融合新器件、新架构、新计算是突破后摩尔时代大算力、高能效瓶颈的重大关键技术领域

代表性智能芯片新兴技术– 新器件：高密度的逻辑器件

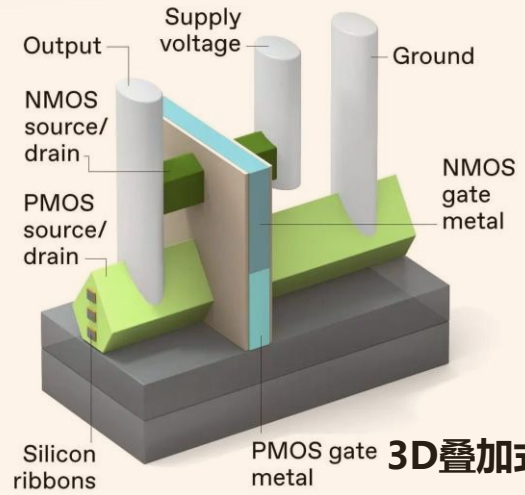
- 未来三维堆叠式晶体管与碳管器件



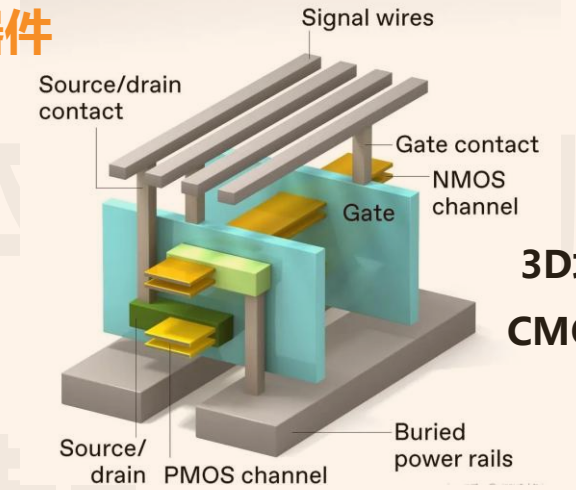
堆叠式CMOS



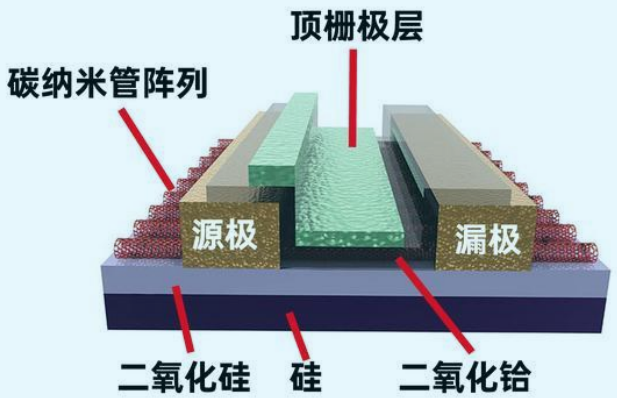
新兴硅基器件

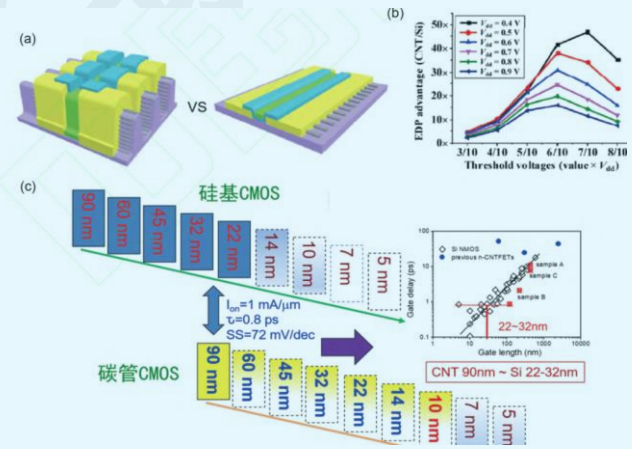


3D叠加式非门



3D堆叠的CMOS连接

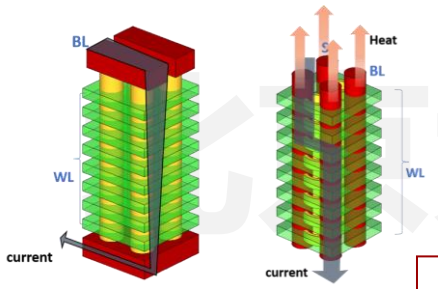




新兴碳基器件

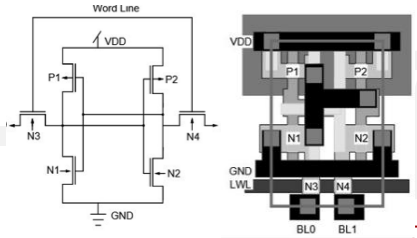
代表性智能芯片新兴技术– 新器件：存储-计算融合器件

- 未来存储器介质材料的创新



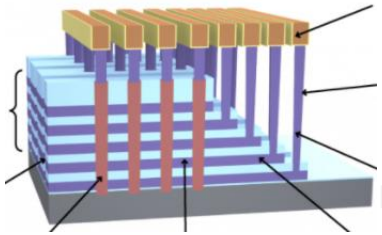
DRAM

优点：工艺成熟、密度高
缺点：速度低、刷新、只近存
非易失性：否
适合场景：冯氏架构过渡



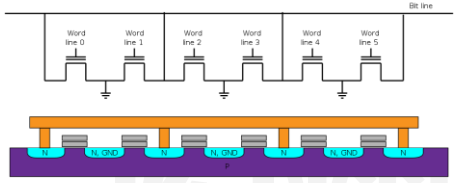
SRAM

优点：工艺成熟、IP化应用
缺点：能效低、密度低
非易失性：否
适合场景：端侧、边缘中小算力



SSD/Nand Flash

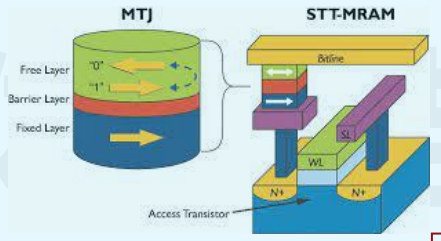
优点：工艺成熟、容量大、成本低
缺点：速度低、只能近存
非易失性：是
适合场景：云端大容量



Nor Flash

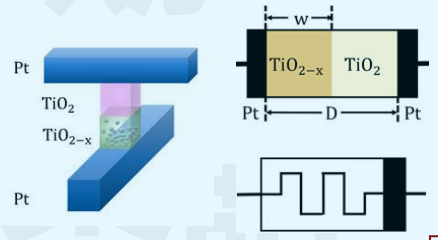
优点：工艺成熟、密度高、成本低
缺点：对PVT变化敏感、能效低
非易失性：是
适合场景：端侧、边缘低成本

新兴器件



磁器件 (MRAM)

优点：能效、速度、密度高
缺点：与CMOS大规模集成难
非易失性：是
适合场景：端侧、边缘中小算力

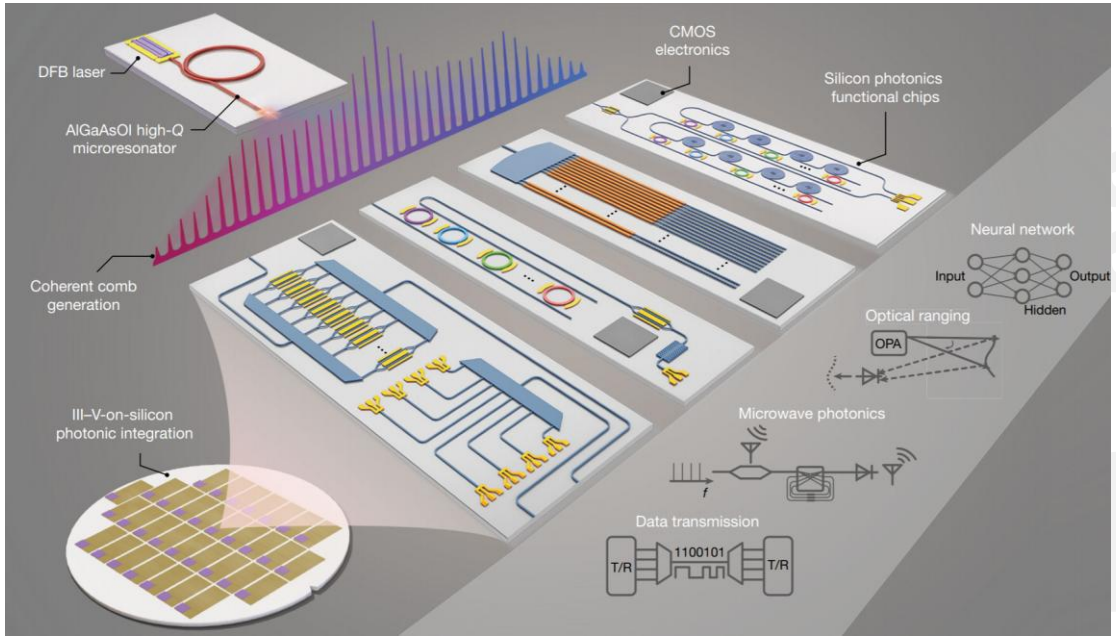


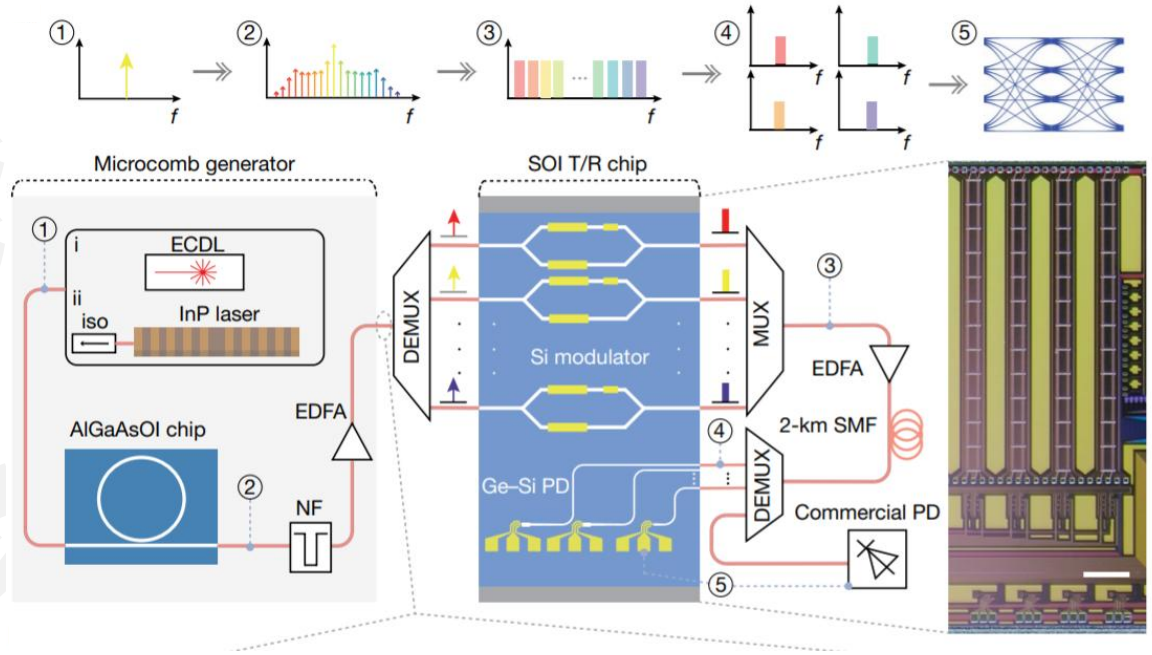
忆阻器 (RRAM/PCM)

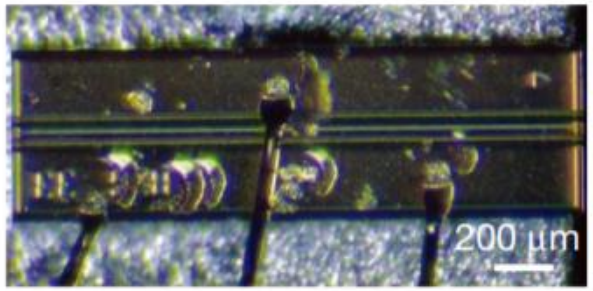
优点：算力、能效、密度高
缺点：工艺爬坡成熟中
非易失性：是
适合场景：云边端大算力

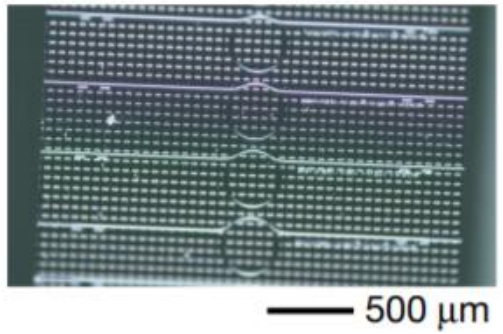
代表性新兴技术 – 新器件：光器件与光计算、光通信技术

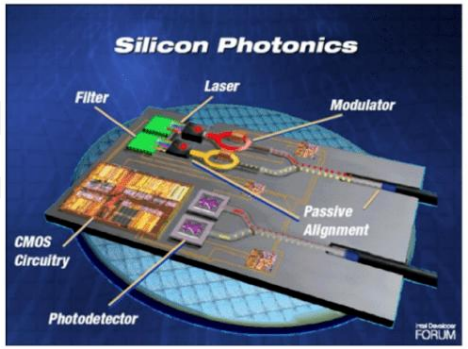
- 片上光计算、光通信有望突破信号传递延时的瓶颈，打破电芯片物理上限

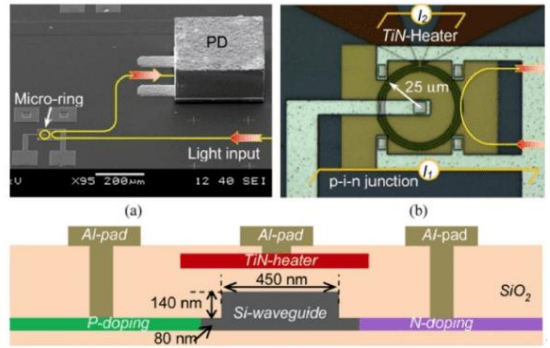






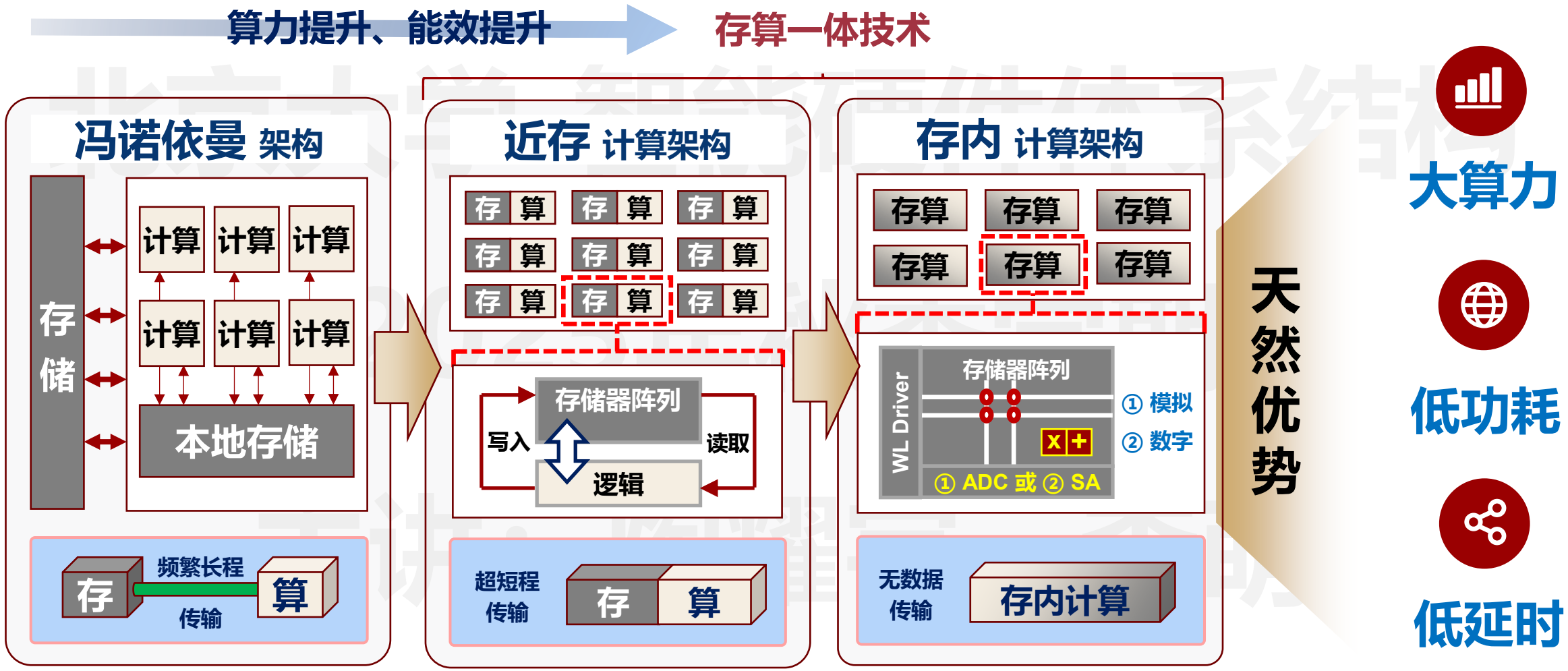






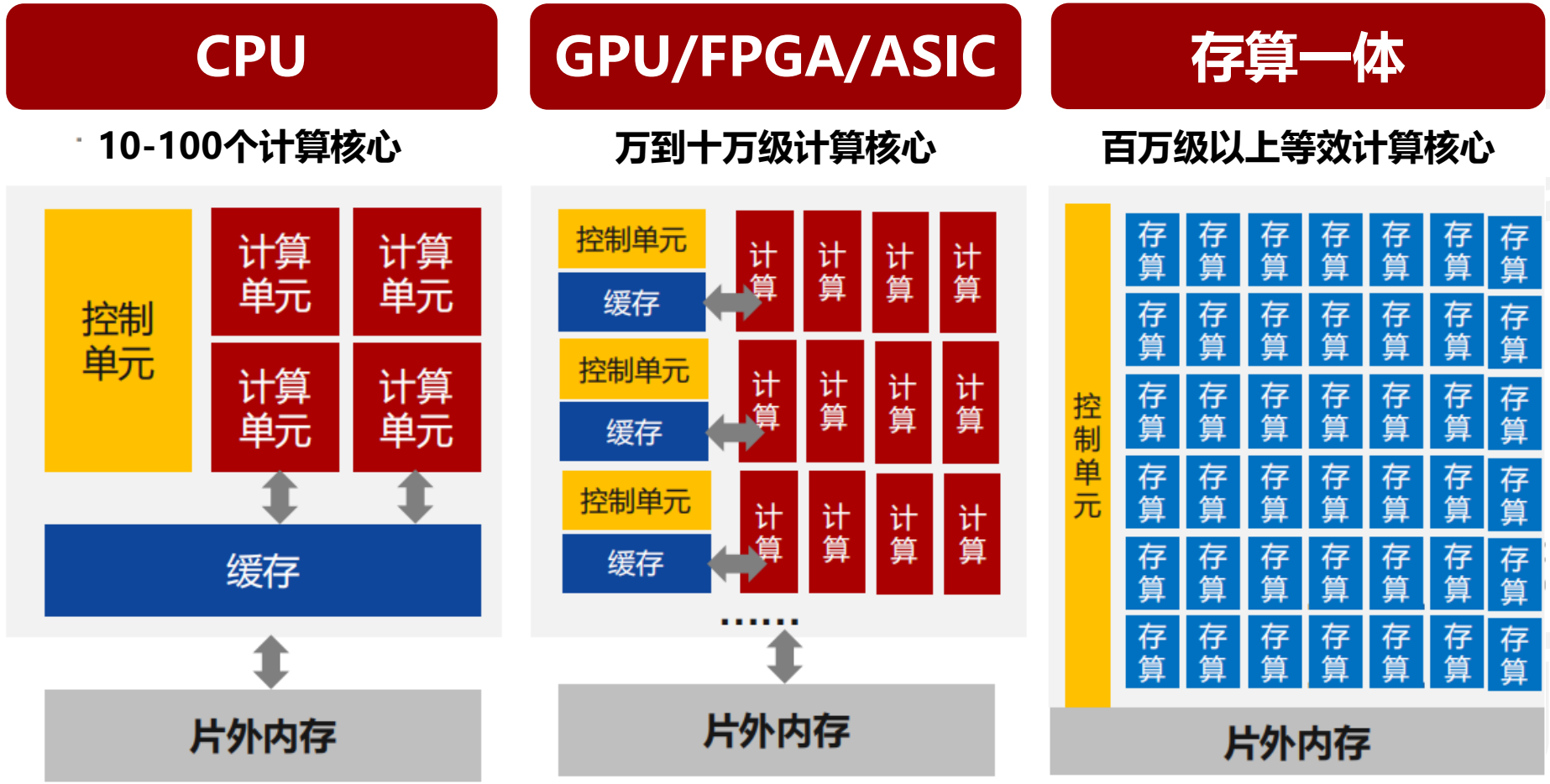
代表性智能芯片新兴技术 – 新架构：存算一体

- 存算一体技术成为后摩尔时代打破算力瓶颈的重要路径



存算一体成为打破AI大模型推理算力极具潜力的技术路径

- 存算一体提供比GPU等冯氏芯片高多个数量级的并发度，有效支撑AI大模型推理

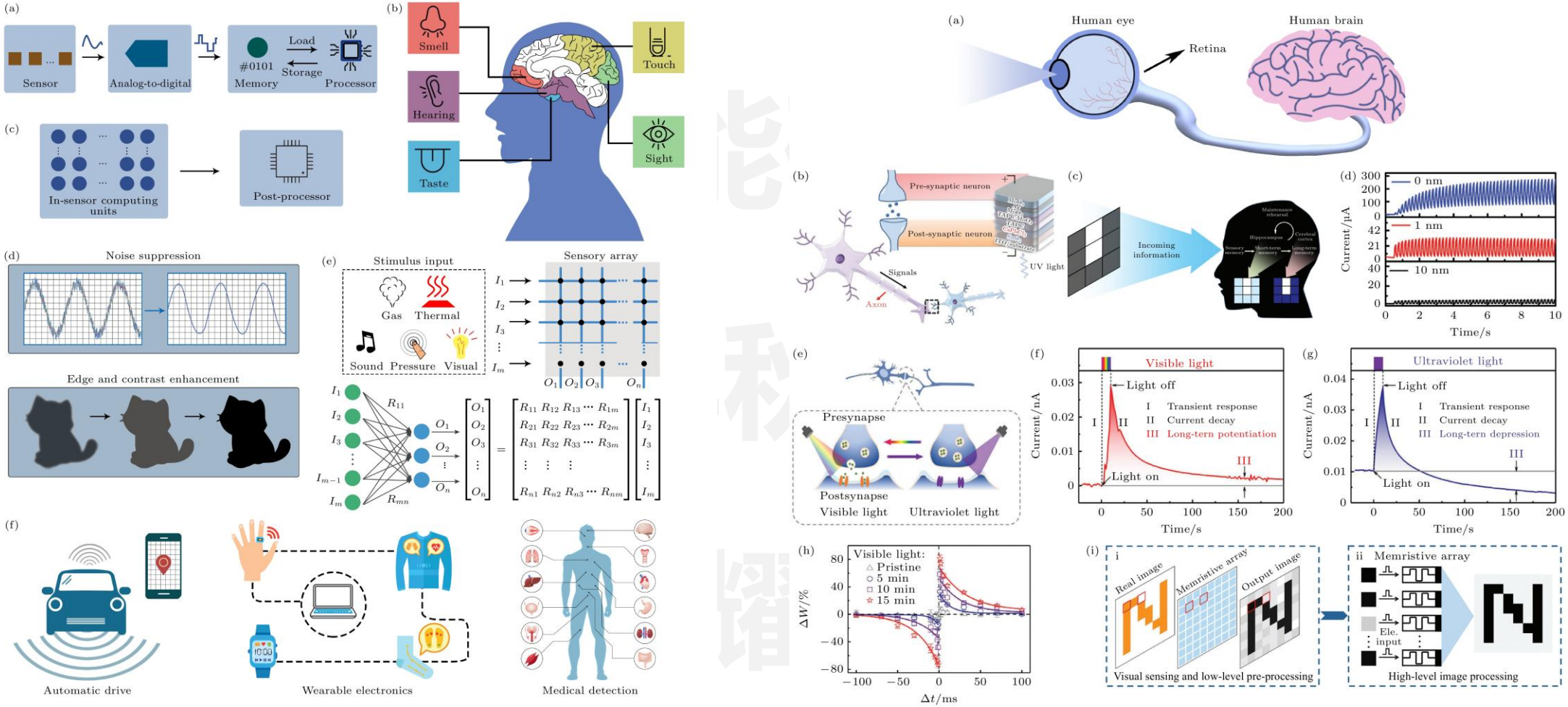


存算一体如何对AI大模型进行有效支持?

现有AI大模型推理基本上基于GPU/FPGA/ASIC等冯氏芯片

代表性智能芯片新兴技术 – 新架构：感存算一体

- 将传感、计算、存储融为一体，大幅降低系统功耗和计算延时，应用前景广阔

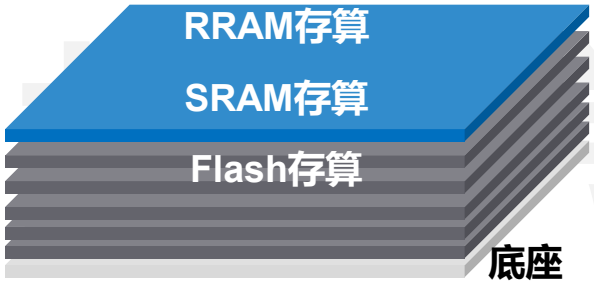


视觉感存算一体芯片与硬件系统

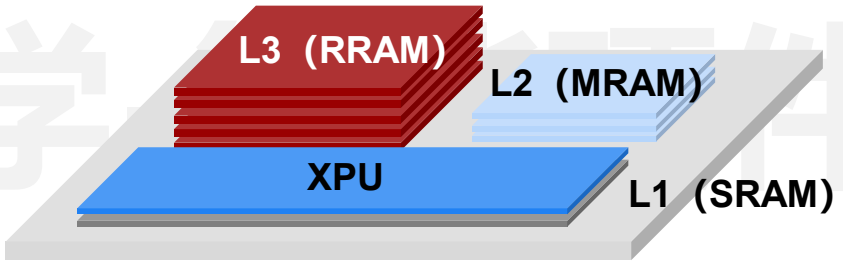
代表性智能芯片新兴技术 – 新架构：三维异质集成

- 协同先进封装技术，实现多种芯片方案相结合

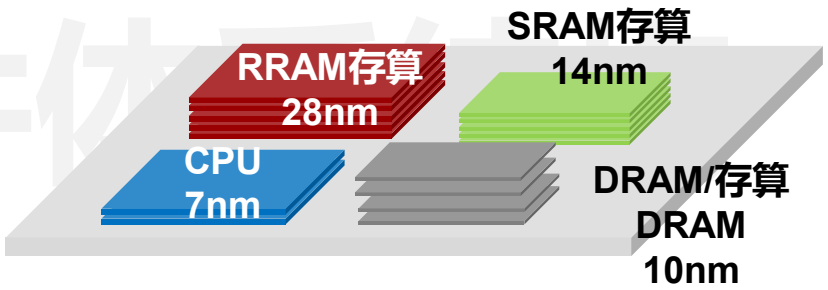
先进三维集成芯片示例图



三维集成

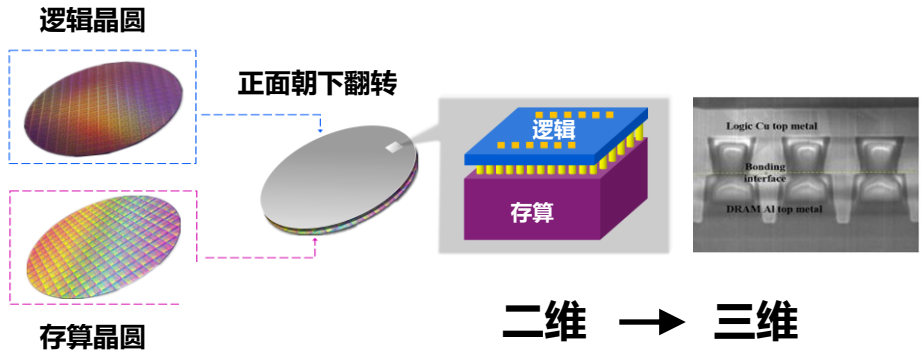


多级存储器堆叠SoC



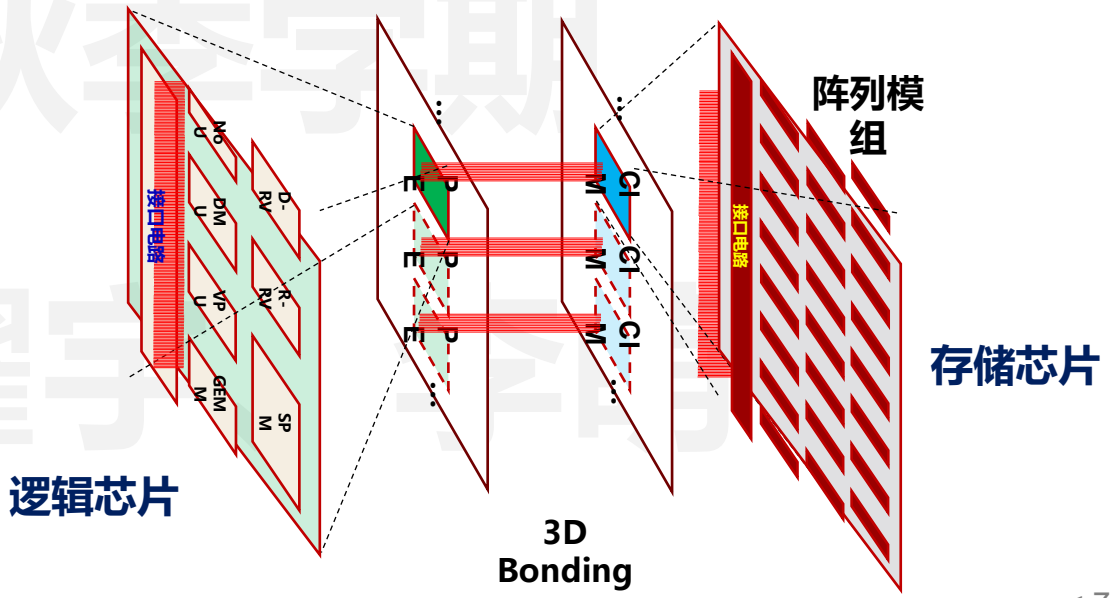
异构小芯粒封装

混合键合异质三维集成



二维 → 三维

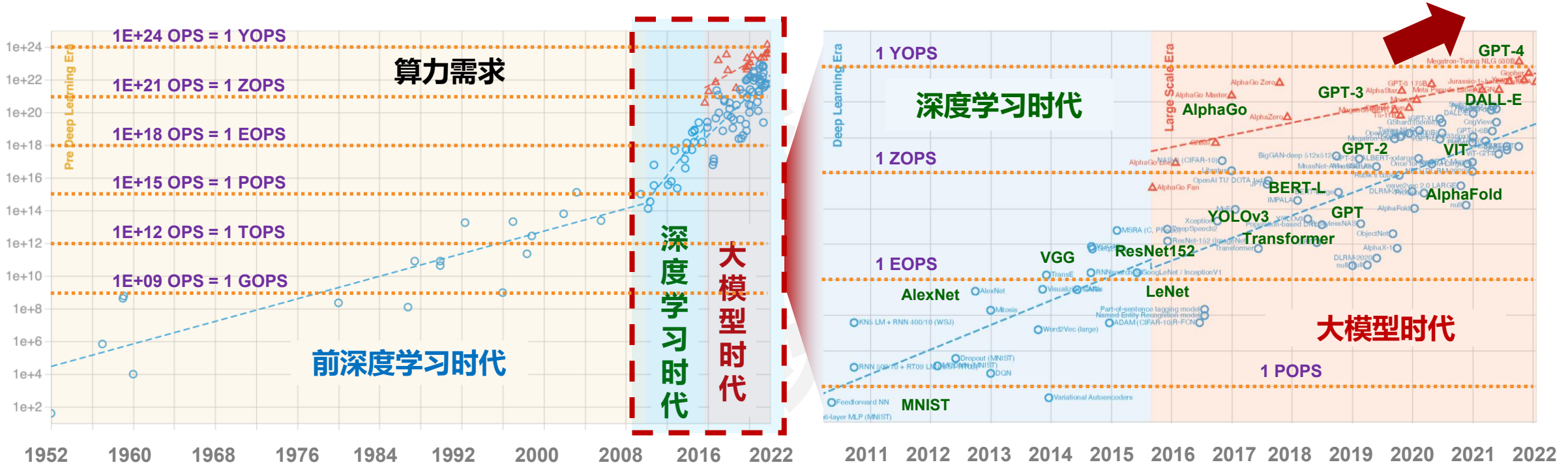
解决线路拥塞、突破面积约束、兼容不同制程、发挥各自优势



代表性智能芯片新兴技术 – 新计算：AI大模型

北京大学 PEKING UNIVERSITY

- 以AI大模型为代表的新一代人工智能系统对高性能AI芯片提出了新的要求



历史时期	算力需求	翻倍间隔
前深度学习时代 1952 – 2010	30 KOPS – 200 TOPS	21.3月
深度学习时代 2010 – 2022	700 TOPS – 2 EOPS	5.7月
大模型时代 2016 – 2022	1 ZOPS – 1 YOPS	9.9月

代表性AI大模型	参数量	算力需求
GPT-4	~1.5万亿个	~2.7 YOPS
GPT-3	~1746亿个	~314 ZOPS
GPT-3 Small	~1.25亿个	~224 EOPS

芯片性能成为支撑智能系统从量变产生质变的基石

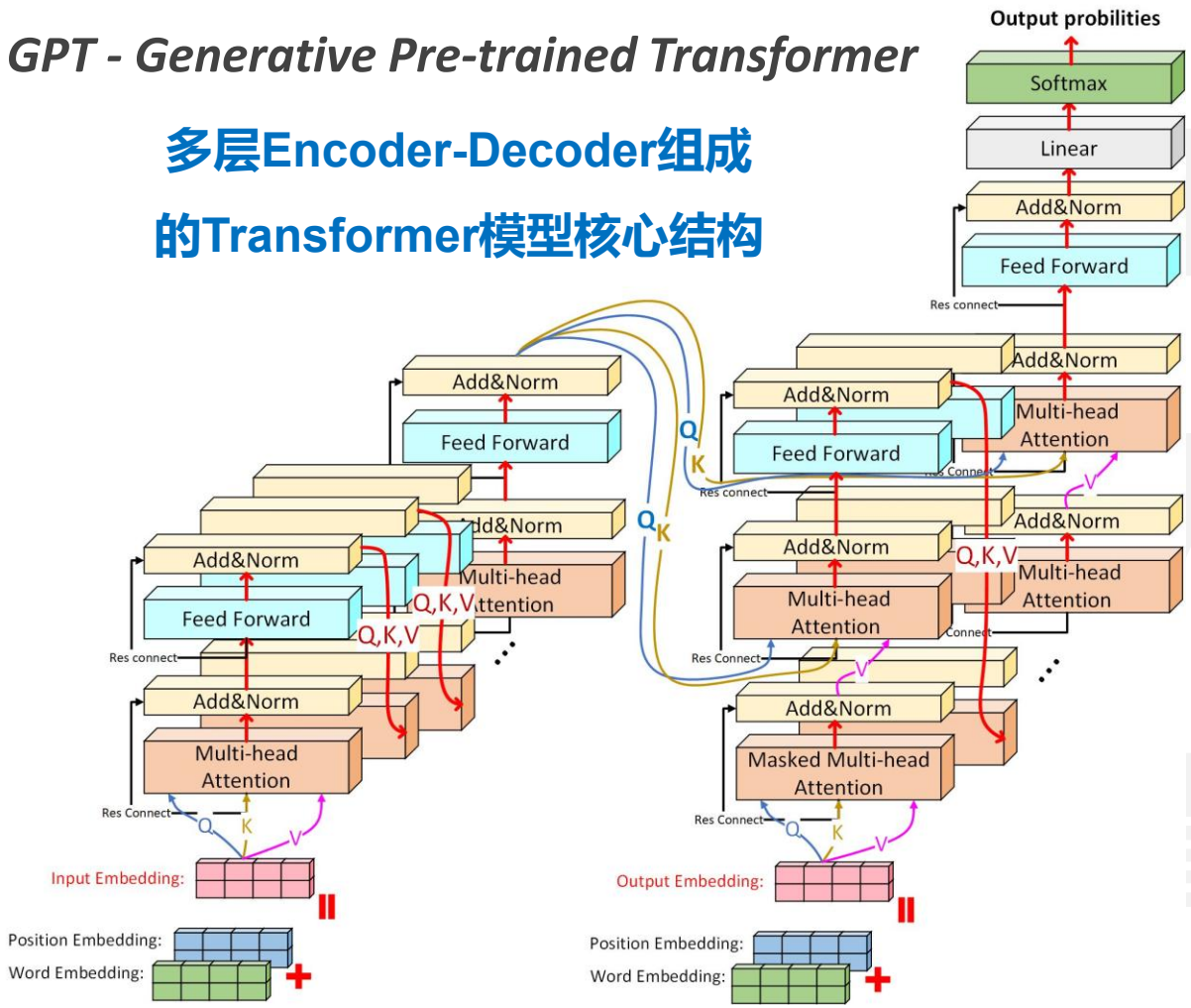
当前AI大模型以Transformer为基干网络 (以GPT为例)

- Decoder-Encoder层数、Token数量、掩码Mask尺寸、特征矩阵尺寸急剧增大

GPT - Generative Pre-trained Transformer

多层Encoder-Decoder组成

的Transformer模型核心结构



输入输出层

Input / Output Embedding

Positional Encoding

自注意力层

Multi-Head Attention

Masked Multi-Head Attention

归一化&前馈层

Add & Norm

Feed Forward

概率输出层

Linear Norm

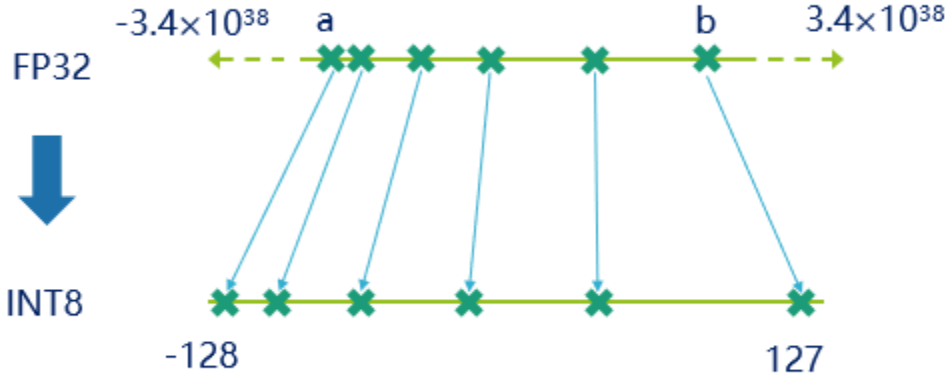
Softmax

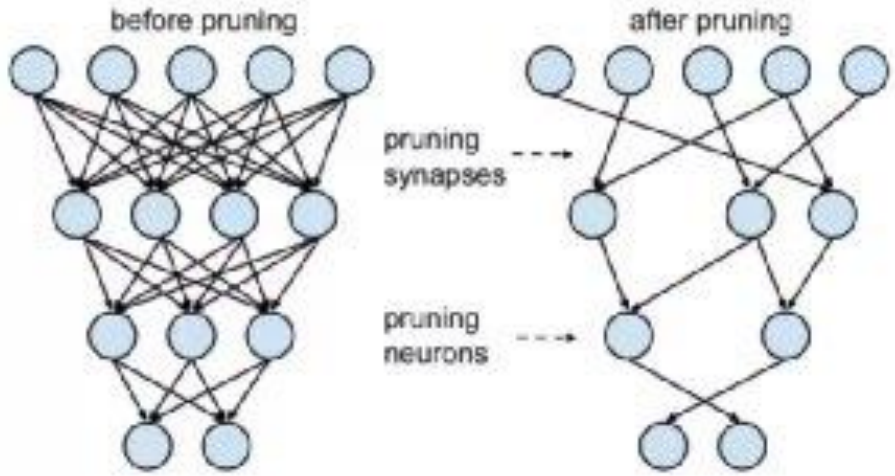
微软/OpenAI提出了**LongNet**，将Transformer的

Token数提高到了10亿级别，并持续提升

软硬件协同设计

- 面对复杂应用，单纯的硬件设计已经不足以支持性能需求，需要软硬件协同设计



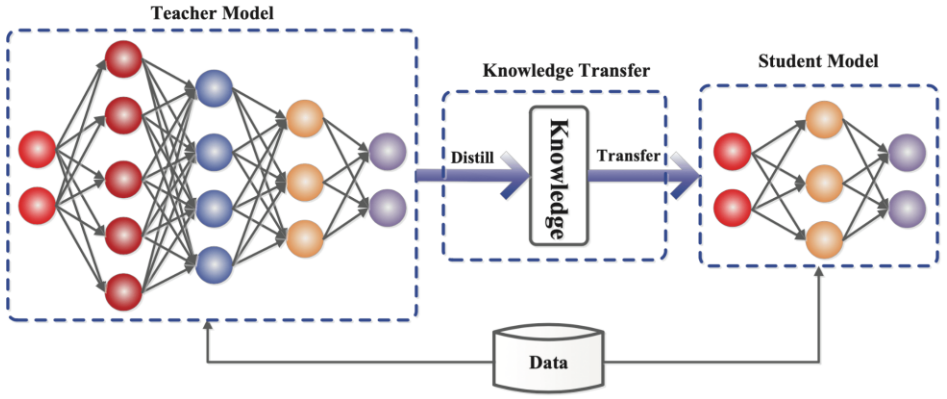


模型量化：将高精度的权重量化为低精度的权重，以一定的精度损失为代价换取更小的存储和计算开销

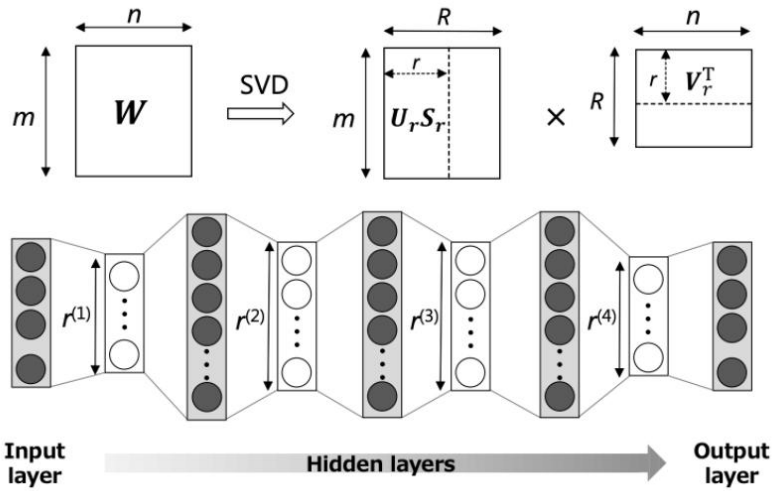
模型剪枝：将神经网络中重要性较小的神经元和权重删除，减少计算量，加速神经网络推理

软硬件协同设计

- 面对复杂应用，单纯的硬件设计已经不足以支持性能需求，需要软硬件协同设计



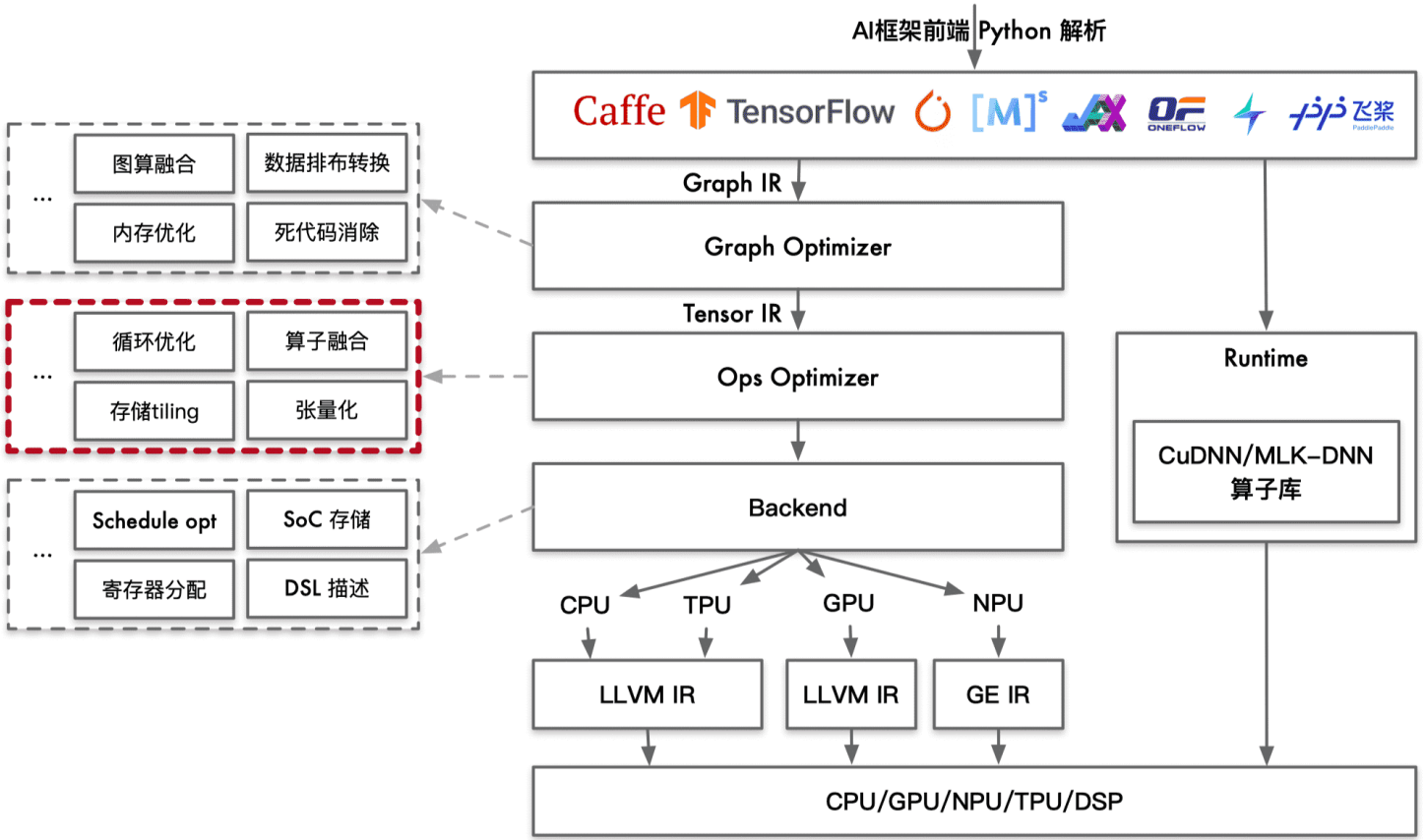
知识蒸馏：将规模较大的模型作为teacher model训练一个较小的student model，在尽可能保证性能的情况下减小模型规模



低秩分解：将大规模权重分解为两个小规模的权重矩阵相乘（SVD），减小矩阵向量乘的计算量

软硬件协同设计

- 编译层面优化



在程序编译过程中

对算子、存储tiling

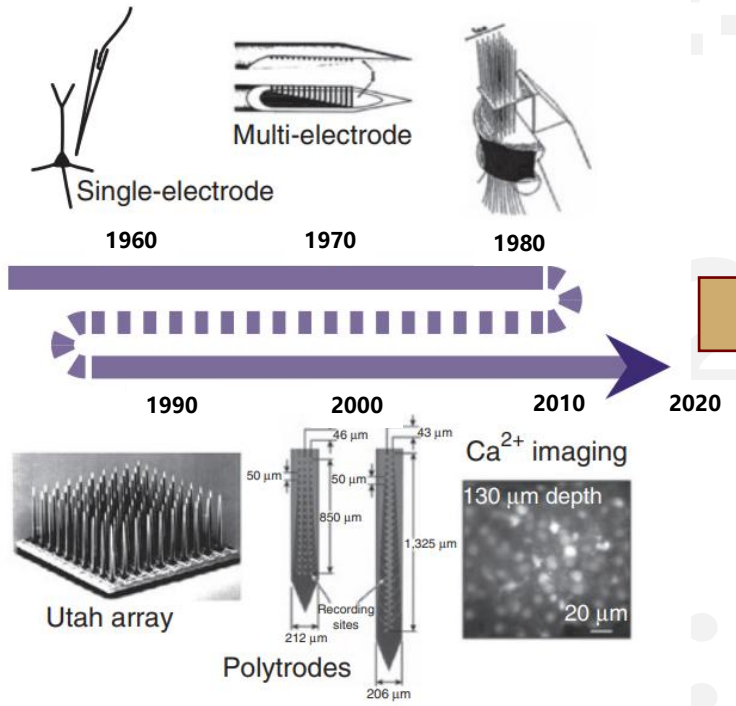
和寄存器分配等等

方面进行优化

代表性新兴技术 – 新计算：脑机接口芯片与系统

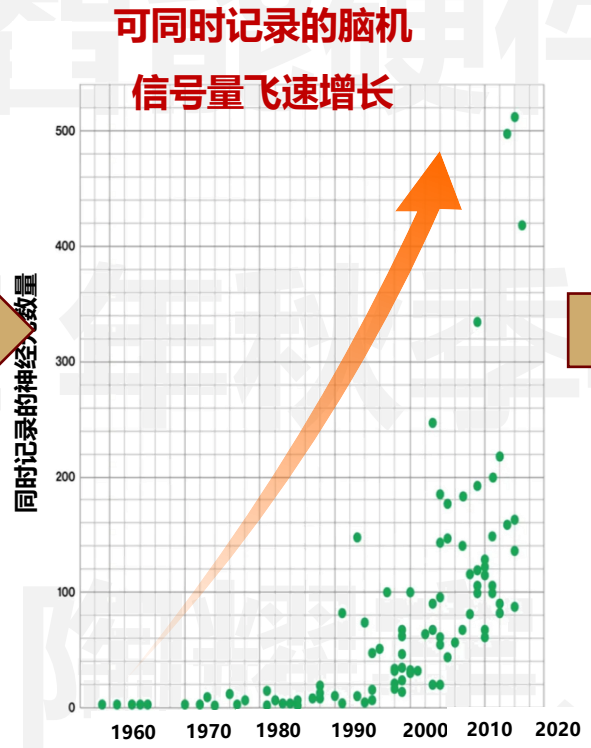
- 为脑机接口服务的芯片与系统将在未来数十年成为人类发展的方向之一

脑机数据采集工具的发展



Stevenson, I., et al. Nat Neuroscience

脑机数据量的“摩尔定律”

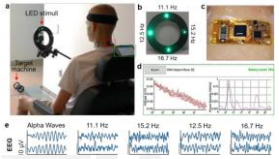


Stevenson, I., et al. Nat Neuroscience

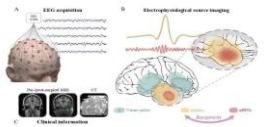
脑机打字

脑控无人机

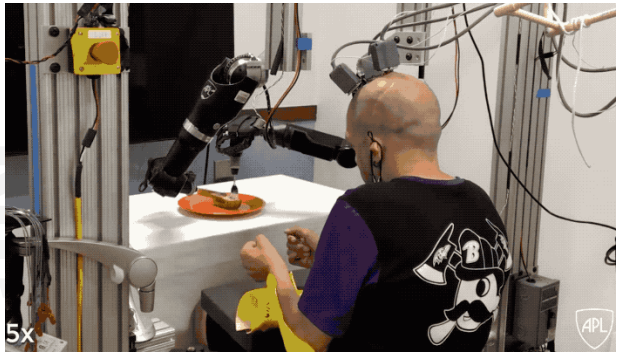
脑控座椅



脑机癫痫监测



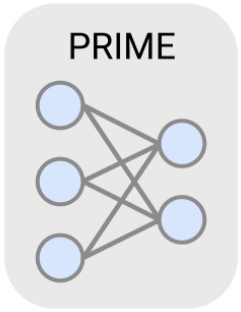
脑控机械臂



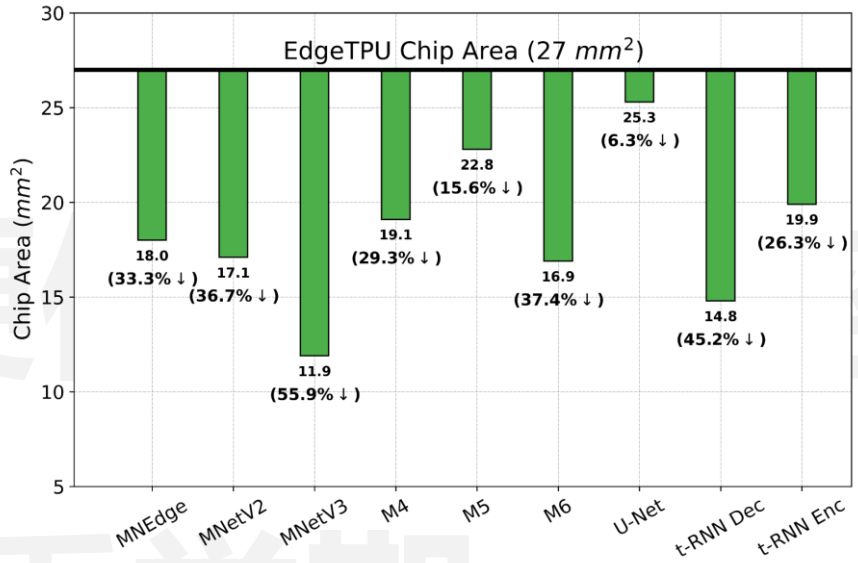
代表性新兴技术 – 新方法：AI设计AI芯片

- 设计AI芯片架构 -> 利用AI设计AI芯片架构

参数化硬件单元库　　针对某类任务的最优芯片设计



Optimizer



RL、大模型等方式

设计AI芯片的AI模型

AI芯片