

并行编程原理与实践

2. 并行计算机体系结构

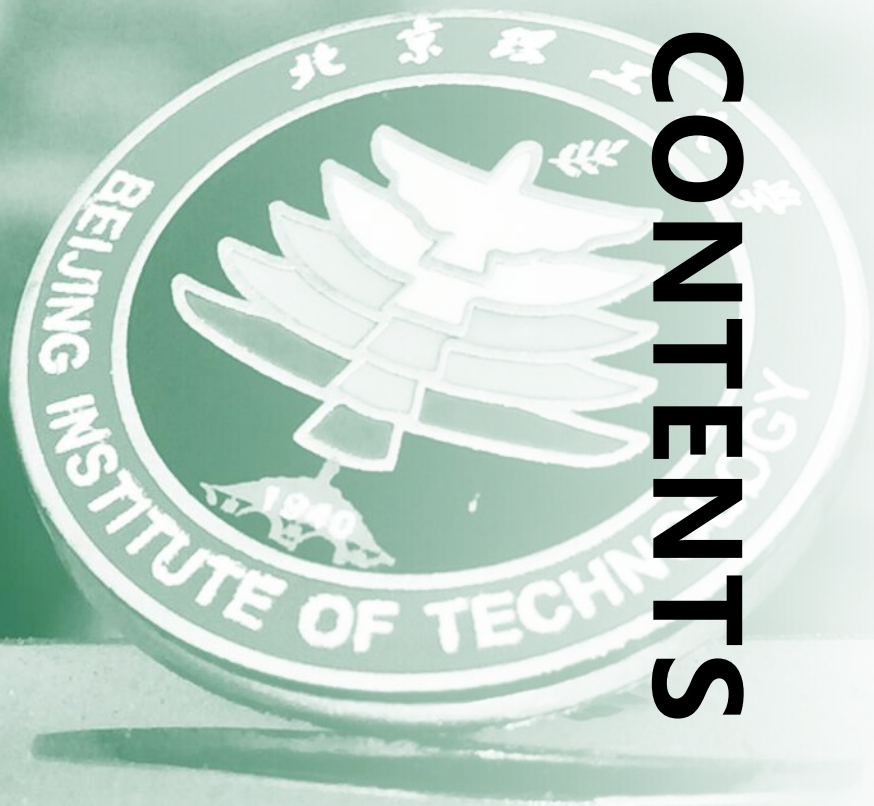
 王一拙、计卫星

 北京理工大学计算机学院

德以明理 学以精工

目录

CONTENTS



- 1 计算机系统结构的背景知识
- 2 计算机系统的分类
- 3 传统体系结构
- 4 现代体系结构

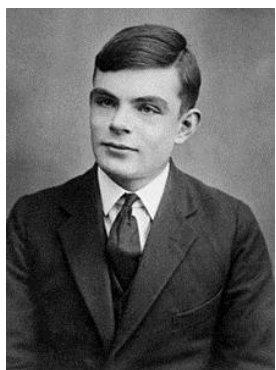


1 计算机系统结构的背景知识

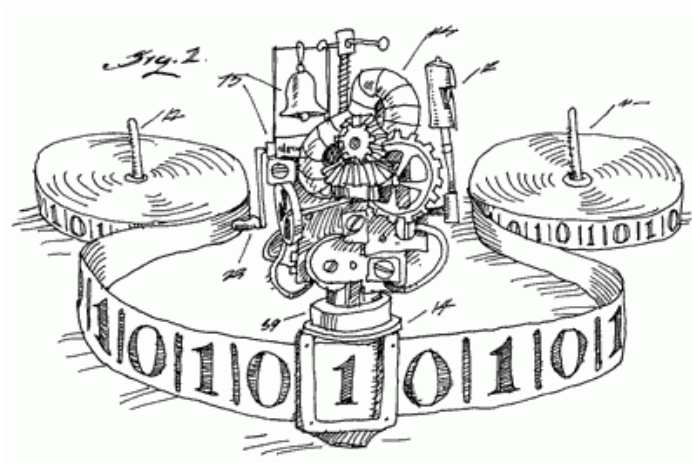
1 计算机系统结构的背景知识



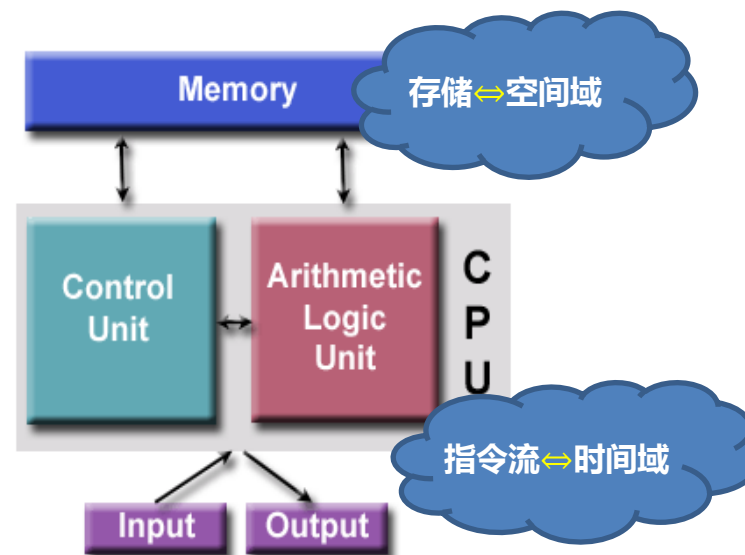
图灵机模型



图灵 (Alan Mathison Turing , 1912-1954年)



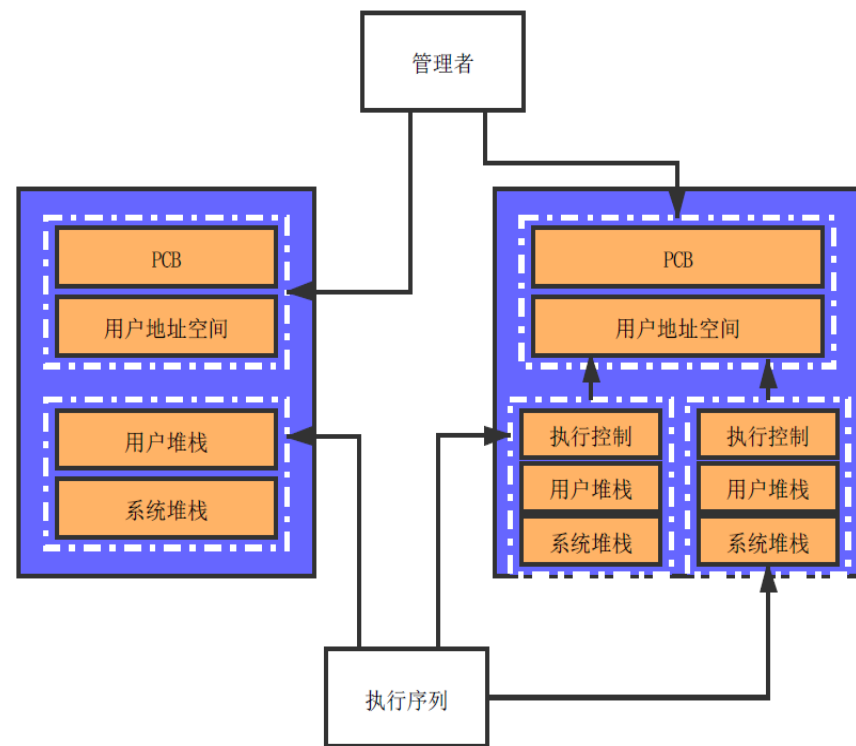
冯·诺依曼结构



冯·诺依曼 (John von Neumann , 1903-1957年)

■ 进程、线程和多任务OS

- 现代操作系统一般都是多用户、多任务OS。
- 进程是计算机中的程序关于某数据集合上的一次运行活动，是OS进行资源分配和调度的基本单位。
- 引入线程后，线程作为被处理器调度运行的最小单位。线程被包含在进程之中，一条线程就是进程中一个单一顺序的控制流。

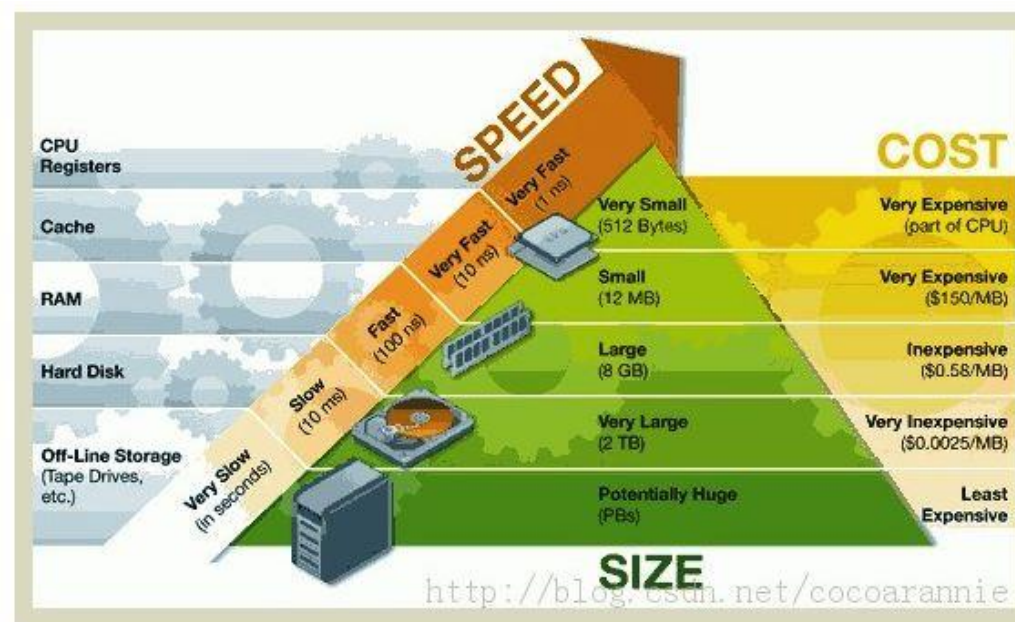


■ 存储系统

➤ 存储墙(Memory Wall)问题：长期以来存储器性能的提高跟不上处理器的发展，使得主存和处理器之间性能的差距日益扩大，就如同在处理器和存储器之间形成了一道墙。

➤ 存储器层次结构

- 寄存器组
- 高速缓冲存储器 (Cache)
- 主存储器
- 辅助存储器
- 脱机存储器



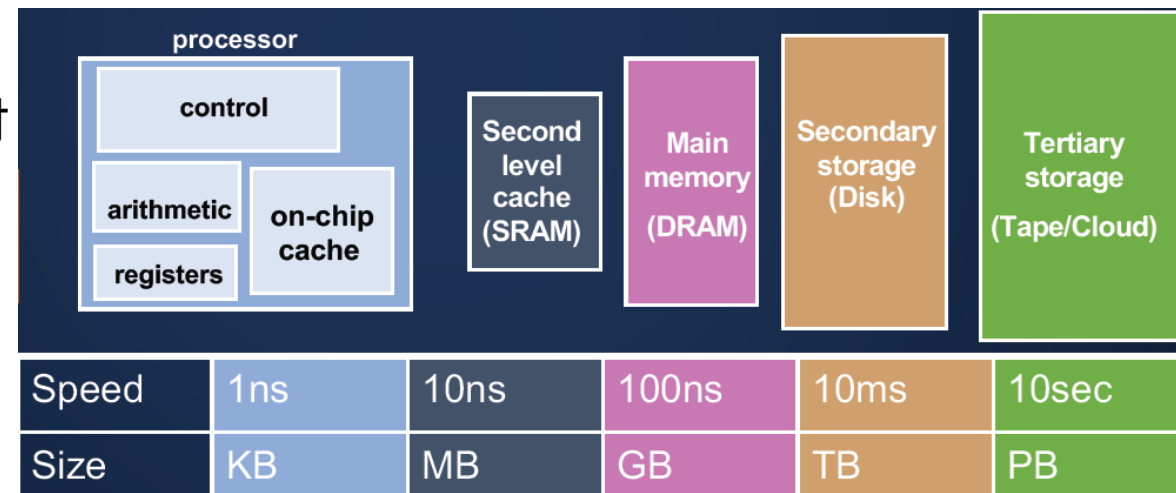
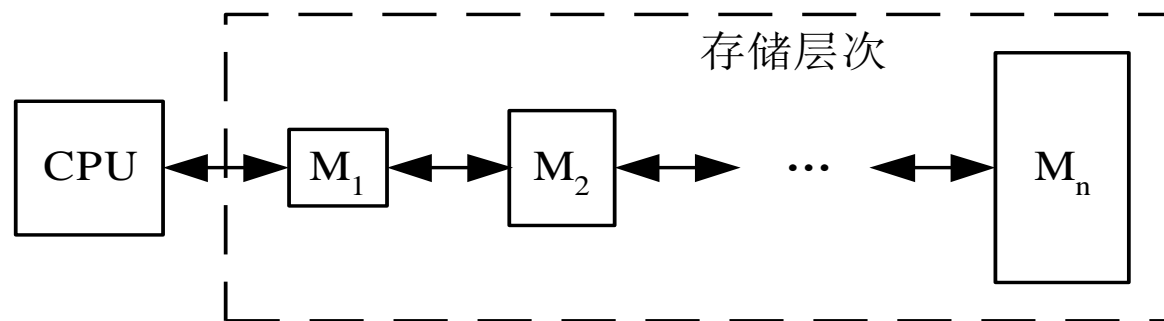
■ 存储系统

➤ 存储系统层次结构

- CPU \rightarrow M1 \rightarrow M2 \dots Mn
- 访问时间T : $T_1 < T_2 < \dots T_n$
- 容量S : $S_1 < S_2 < \dots S_n$
- 平均每位价格C : $C_1 > C_2 > \dots C_n$
- 整个系统的平均访问时间接近M1的访问时间，容量和平均每位价格接近Mn的。

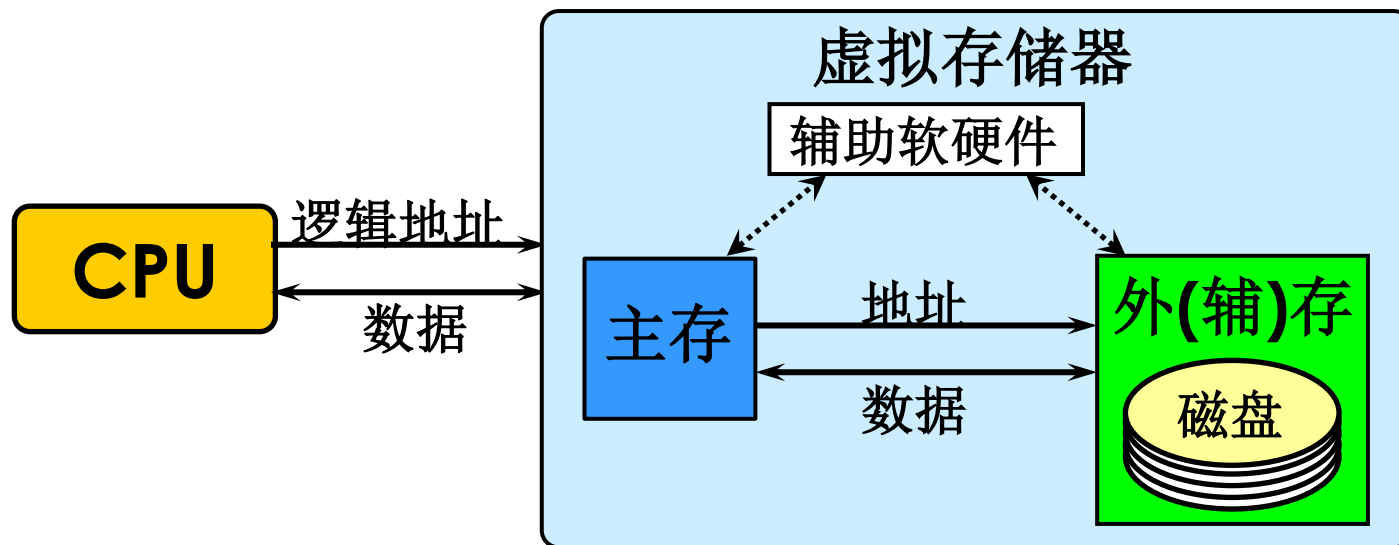
➤ 程序的局部性原理

- 时间局部性
- 空间局部性



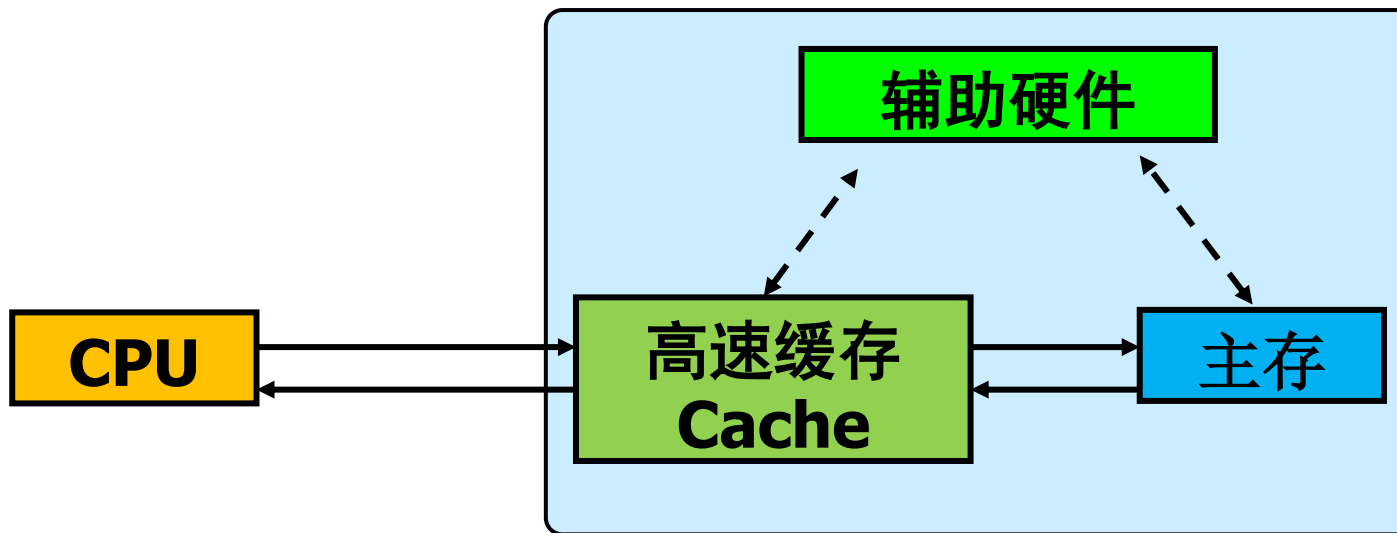
■ 虚拟存储系统

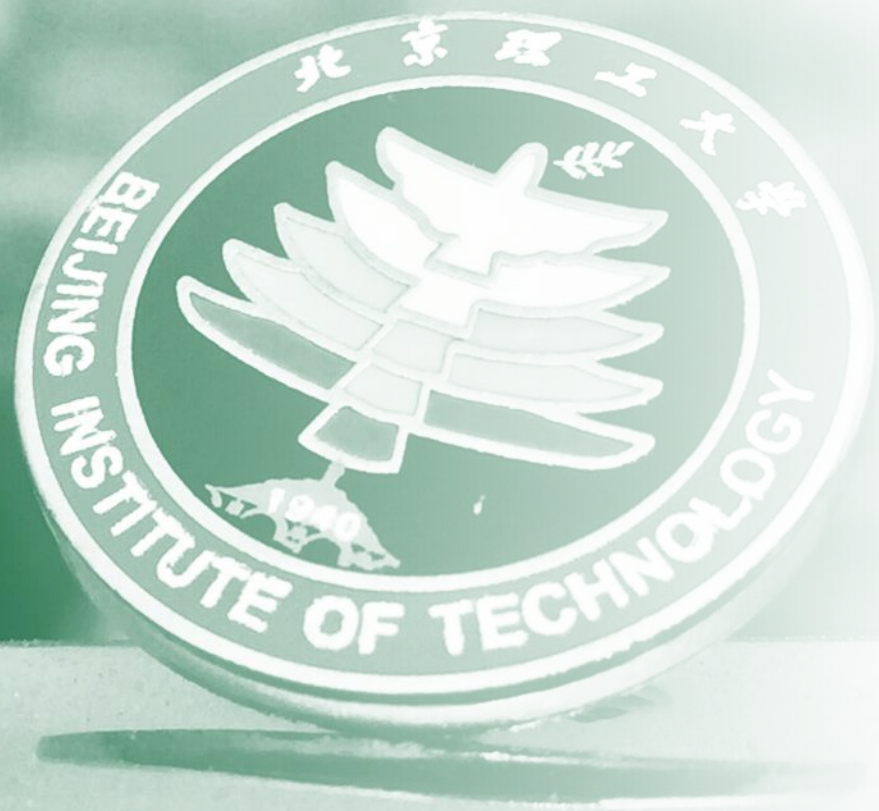
- 主存 - 辅存存储层次（二级存储层次）
- 主要解决容量问题：容量和价格是外(辅)存的，速度是主存的。



Cache存储系统

- Cache - 主存存储层次（二级存储层次）
- 主要解决速度问题：从CPU看，速度是Cache的，容量是主存的。





2 计算机系统的分类



■ Flynn分类

- 1966年，Michael J. Flynn提出
- 广泛使用的方法之一
- 按指令流和数据流的多倍性将计算机系统分为四类：
 - SISD系统 (单指令流，单数据流系统)
 - SIMD系统 (单指令流，多数据流系统)
 - MISD系统 (多指令流，单数据流系统)
 - MIMD系统 (多指令流，多数据流系统)

SISD Single Instruction stream Single Data stream	SIMD Single Instruction stream Multiple Data stream
MISD Multiple Instruction stream Single Data stream	MIMD Multiple Instruction stream Multiple Data stream

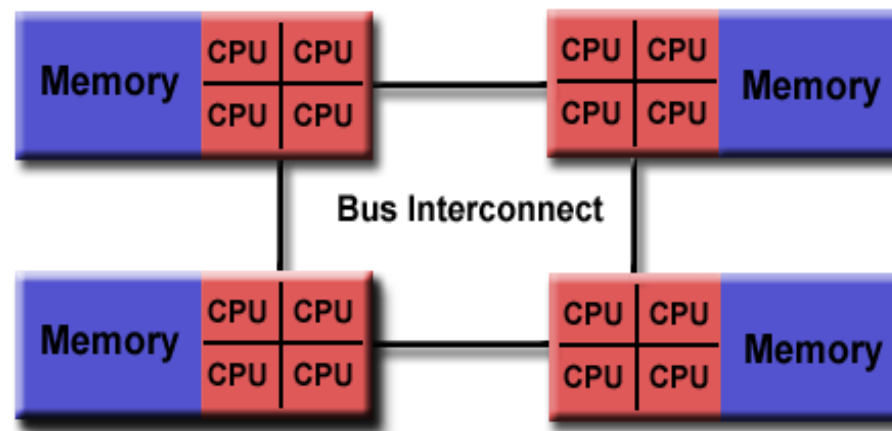
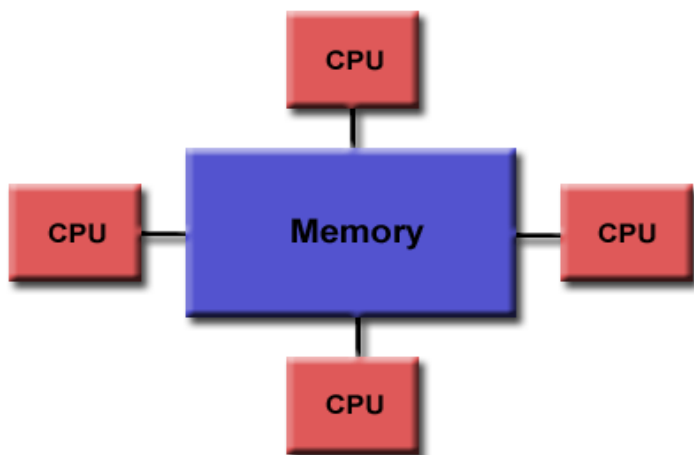


■ 从存储模型角度分类

- 共享存储计算机系统
- 分布式存储计算机系统
- 混合分布式-共享存储计算机系统

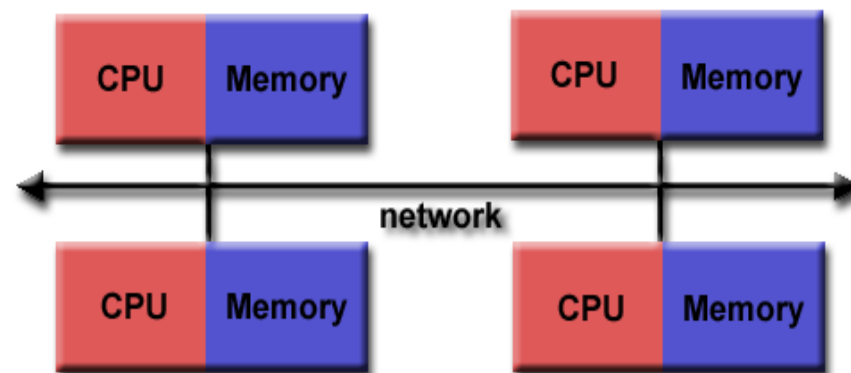
■ 共享存储计算机系统

- UMA (Uniform Memory Access)
- NUMA (Non-Uniform Memory Access)



■ 分布式存储计算机系统

- 多个节点通过网络连接在一起，每个节点的处理器有自己本地的存储
- 相比共享存储系统，分布式系统具有很好的可扩展性
- 分布式系统中处理器访问本地存储的速度很快，而且通常不用维护与其它节点的数据一致性
- 节点间的数据传递、任务同步，数据在系统中的分布等诸多细节都需要程序员自己实现
- 编程时要了解网络拓扑结构并小心处理网络拥塞等问题

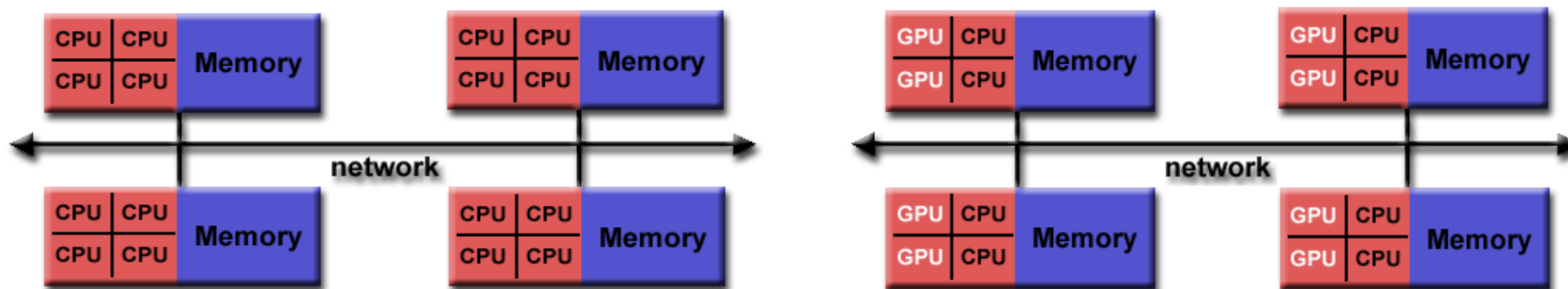


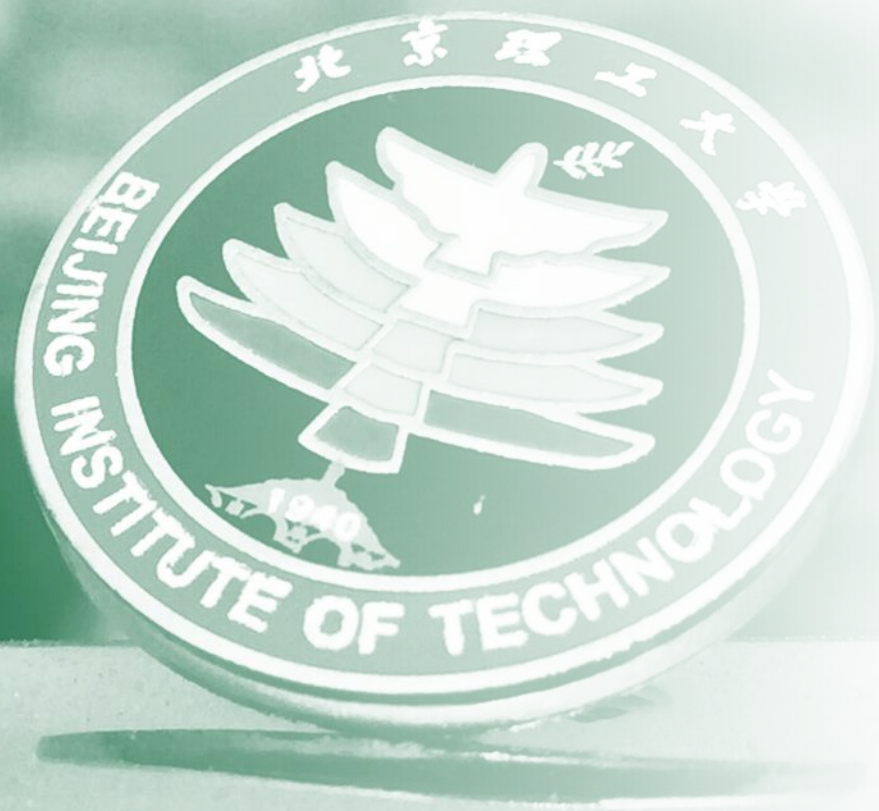
2 计算机系统的分类



■ 混合分布式-共享存储计算机系统

- 这种系统可分成两个层次：
 - 节点之间通过网络互连的层次
 - 节点内部多个处理器共享访问同一个存储器
- 系统可扩展性更加灵活，可以同时扩充分布式的互连节点或者节点上的处理器和存储器来提高系统性能。但系统复杂性的增加也使得编程更加困难。

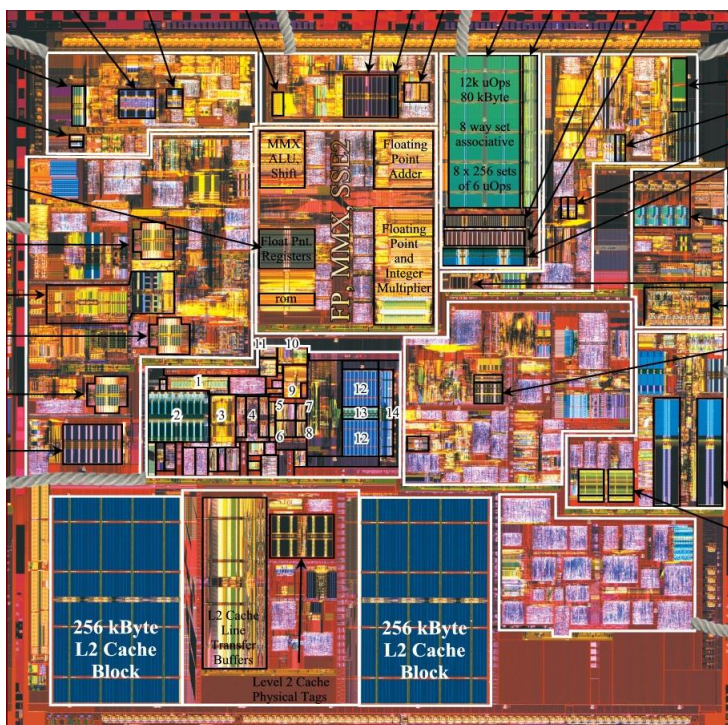




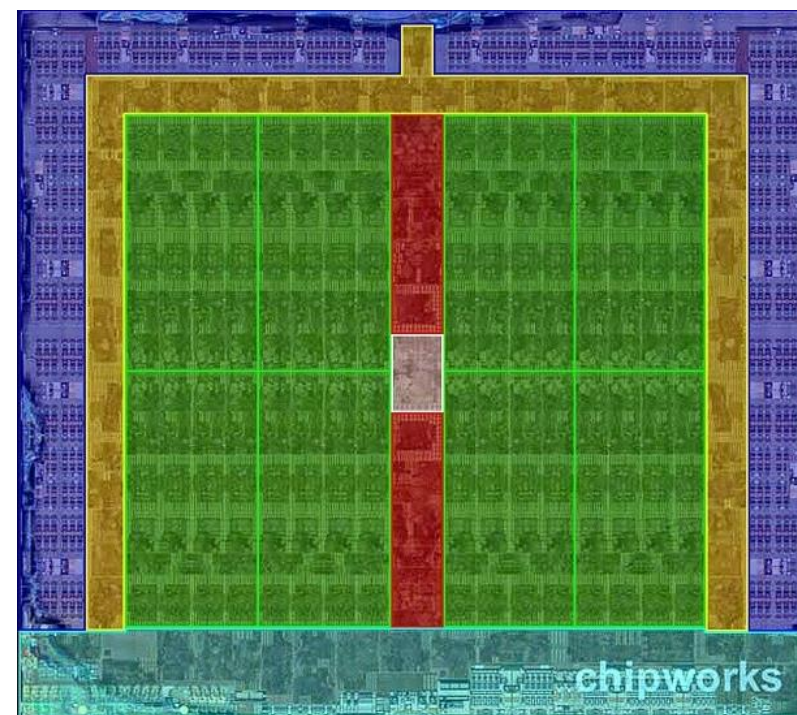
3

传统体系结构

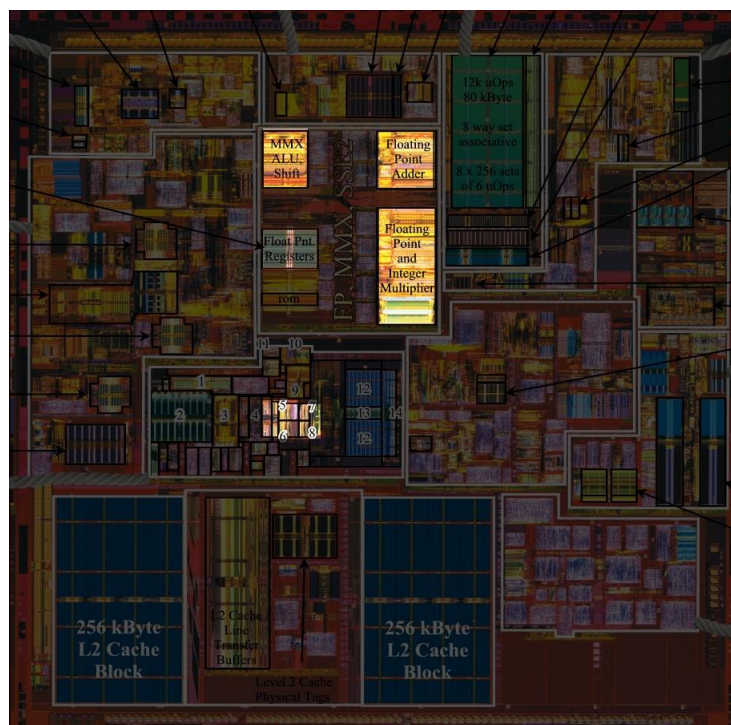
Pentium 4 “Northwood” (2002)



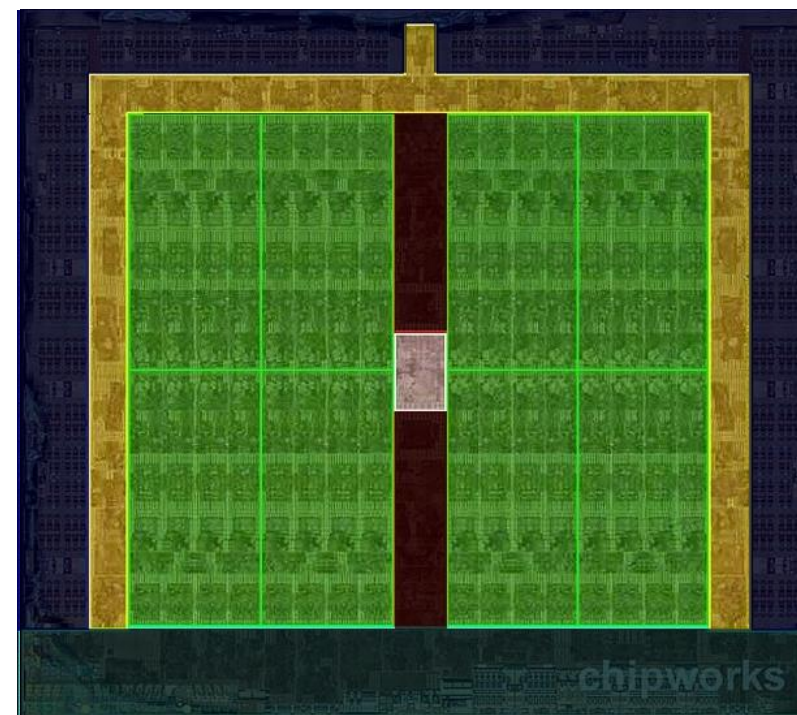
AMD Fiji (2015)



Pentium 4 “Northwood” (2002)



AMD Fiji (2015)

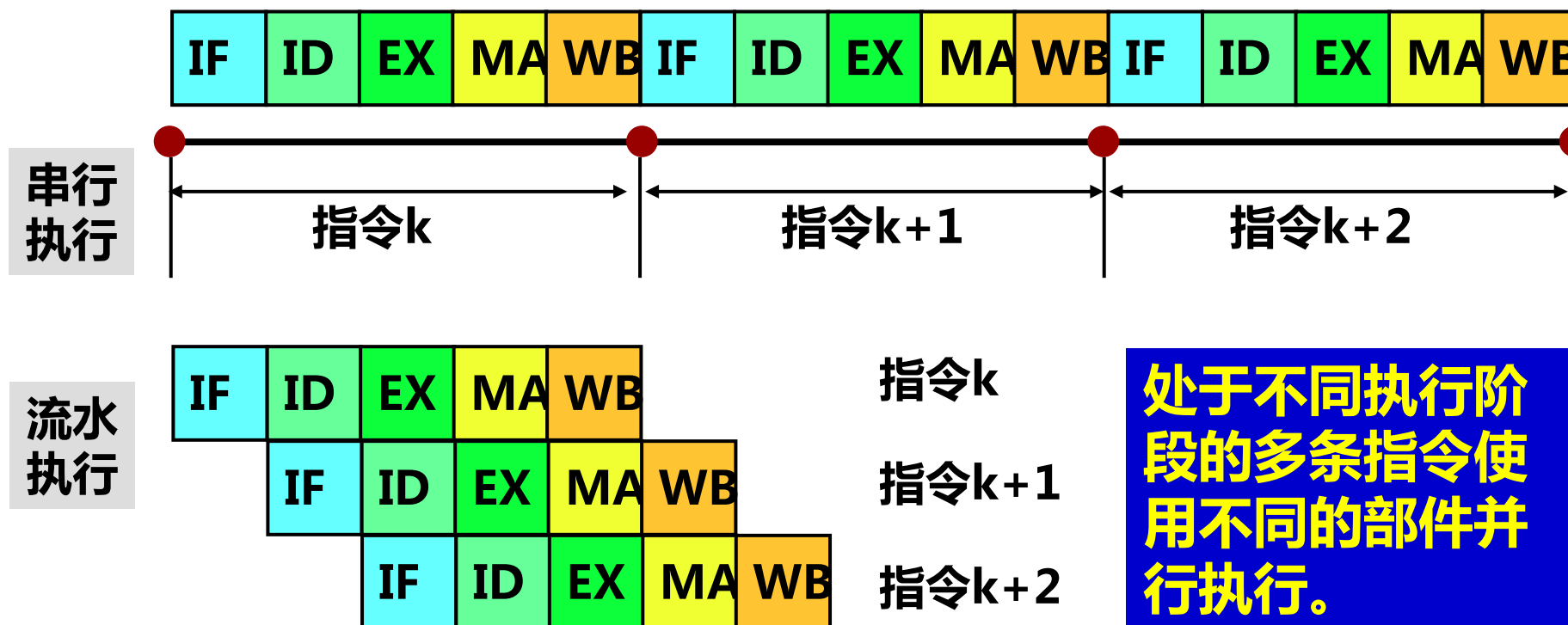




■ 发掘ILP (Instruction-Level Parallelism) 的技术

- 流水线 (Pipelining)
- 多发射 (Multi-issue)
- 超标量 (Superscalar)
- 乱序执行 (Out of order execution)
- 预测执行 (Speculative Execution)
- 超长指令字 (VLIW)
- 向量指令 (SIMD)
- 硬件多线程 (Hyperthreading)
-

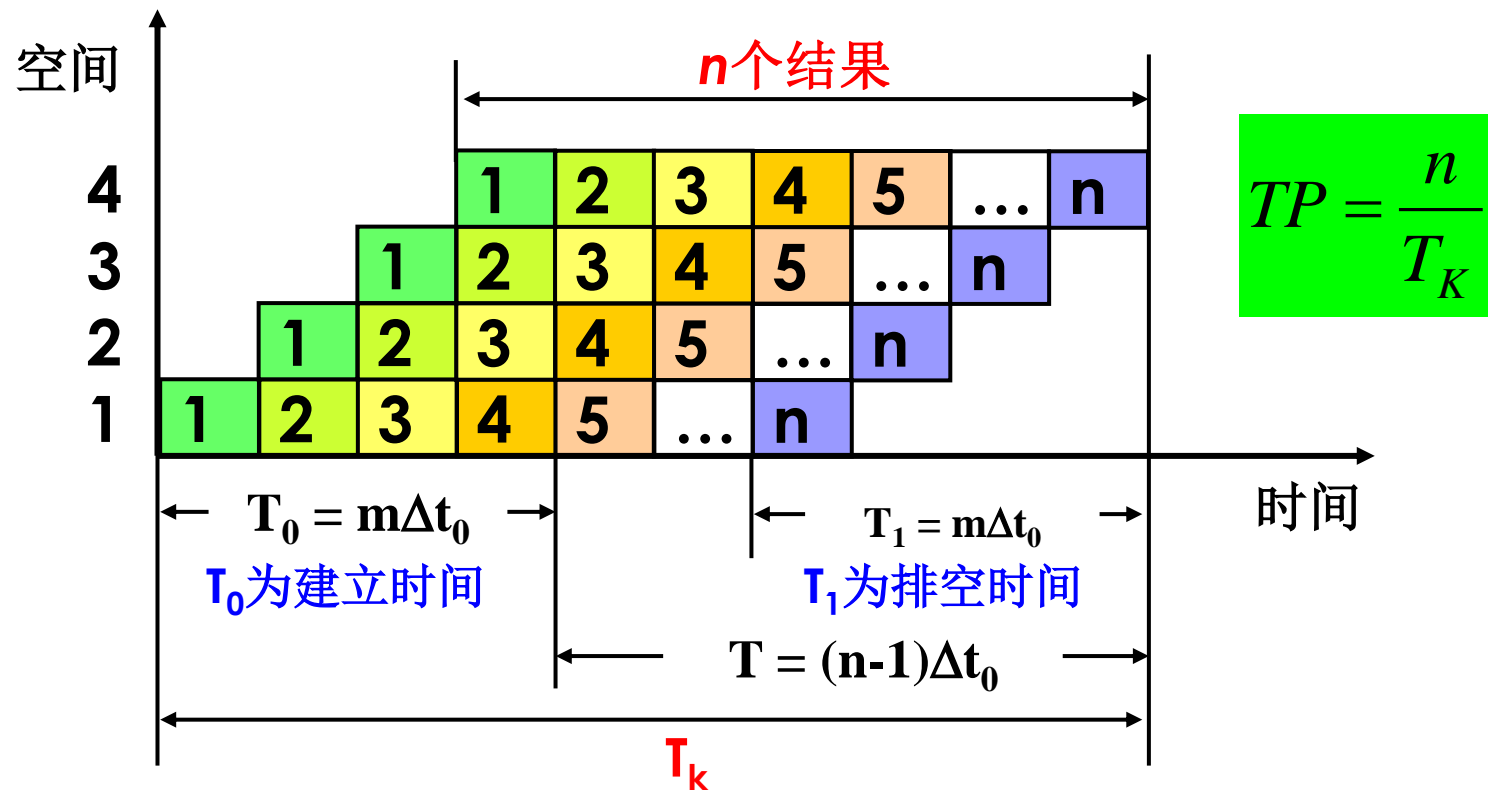
■ 流水线 (Pipelining)



假设每个子过程的处理时间均为 Δt ，则：串行执行完成时间 $= 5 \times \Delta t \times 3 = 15\Delta t$ ；流水执行完成时间 $= 7\Delta t$

流水线的性能指标

- 吞吐率：流水线单位时间内能处理的指令条数或能输出的结果数。
- 加速比：指流水线速度与等效的非流水线速度之比。



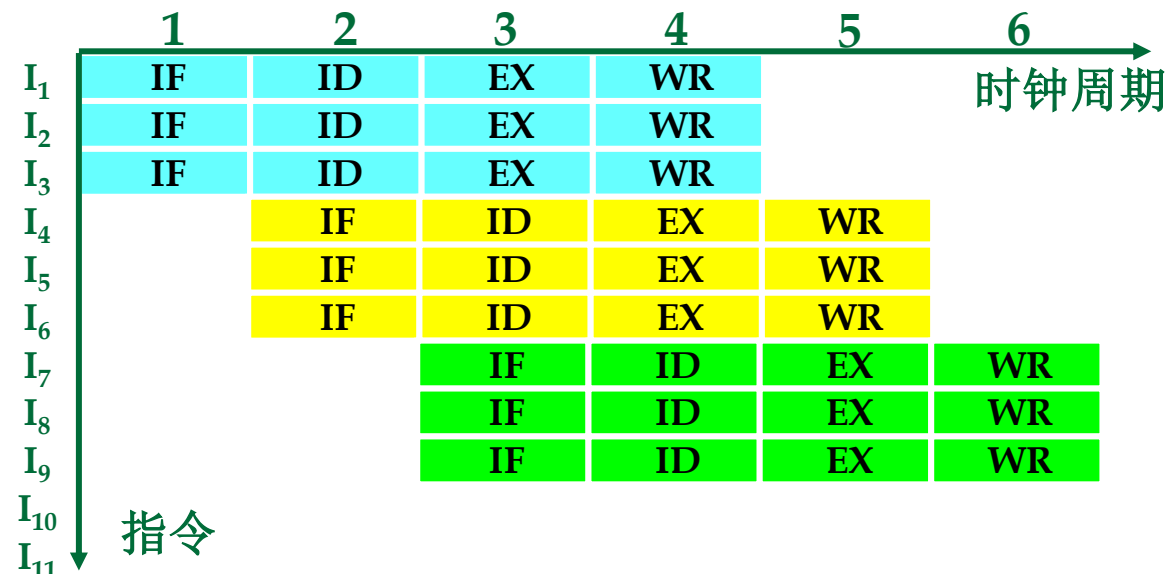
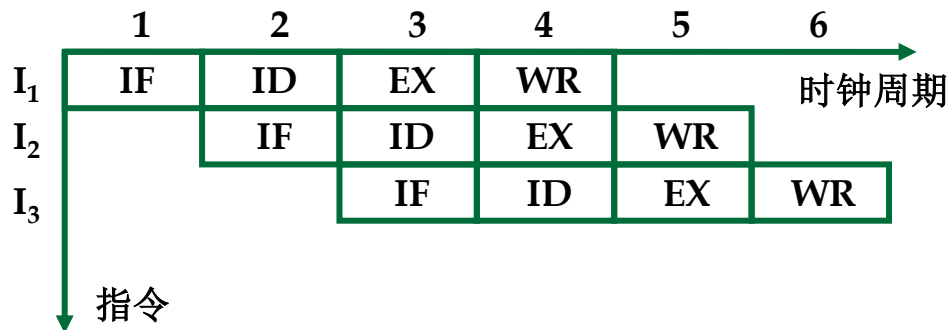


■ 多发射 (Multi-issue)

- 单发射处理机：每个周期只取一条指令、只译码一条指令，只执行一条指令，只写回一个运算结果。
- 多发射处理机：每个周期同时取多条指令、同时译码多条指令，同时执行多条指令，同时写回多个运算结果。
- 多发射处理机的实现技术可分为两类：
 - 静态多发射 — 向量指令 (SIMD)、超长指令字 (VLIW)
 - 动态多发射 — 超标量 (Superscalar)

■ 超标量 (Superscalar)


- 标量处理机一个时钟周期最多完成一条指令的执行
- 超标量处理机通过同时发射多条指令（多发射），设置多个指令执行部件，实现一个时钟周期执行多条指令





■ 乱序执行 (Out of order execution)

- 按照书写顺序执行称为按序执行 (In-Order)。按序执行是限制流水线性能的主要因素之一。
- 乱序执行 (Out-of-Order) 允许将多条指令不按程序规定的顺序分开发送给各相应电路单元处理。根据各个电路单元的状态和各指令能否提前执行的具体情况分析后，将能提前执行的指令立即执行。



ld	r1, 0(r2) // load r1 from memory at r2
add	r2, r1, r3 // r2 := r1 + r3
add	r4, r3, r5 // r4 := r3 + r5

■ 乱序执行 (Out of order execution)

Pipelined Processing (486)

Cycle	1	2	3	4	5	6	7	8	9
Instr ₁	Fetch	Decode	Execute	Write					
Instr ₂		Fetch	Decode	Execute	Write				
Instr ₃			Fetch	Decode	Execute	Write			
Instr ₄				Fetch	Decode	Execute	Write		
Instr ₅					Fetch	Decode	Execute	Write	
Instr ₆						Fetch	Decode	Execute	Write

■ 乱序执行 (Out of order execution)

In-Order Pipeline (486)

Cycle	1	2	3	4	5	6	7	8	9
Instr ₁	Fetch	Decode	Execute			Write			
Instr ₂		Fetch	Decode	Wait		Execute	Write		
Instr ₃			Fetch	Decode	Wait		Execute	Write	
Instr ₄				Fetch	Decode	Wait		Execute	Write
Instr ₅					Fetch	Decode	Wait		Execute
Instr ₆						Fetch	Decode	Wait	



■ 乱序执行 (Out of order execution)

Out-of-Order Execution (Pentium II)

Cycle	1	2	3	4	5	6	7	8	9
Instr ₁	Fetch	Decode	Execute			Write			
Instr ₂		Fetch	Decode	Wait		Execute	Write		
Instr ₃			Fetch	Decode	Execute	Write			
Instr ₄				Fetch	Decode	Wait	Execute	Write	
Instr ₅					Fetch	Decode	Execute	Write	
Instr ₆						Fetch	Decode	Execute	Write

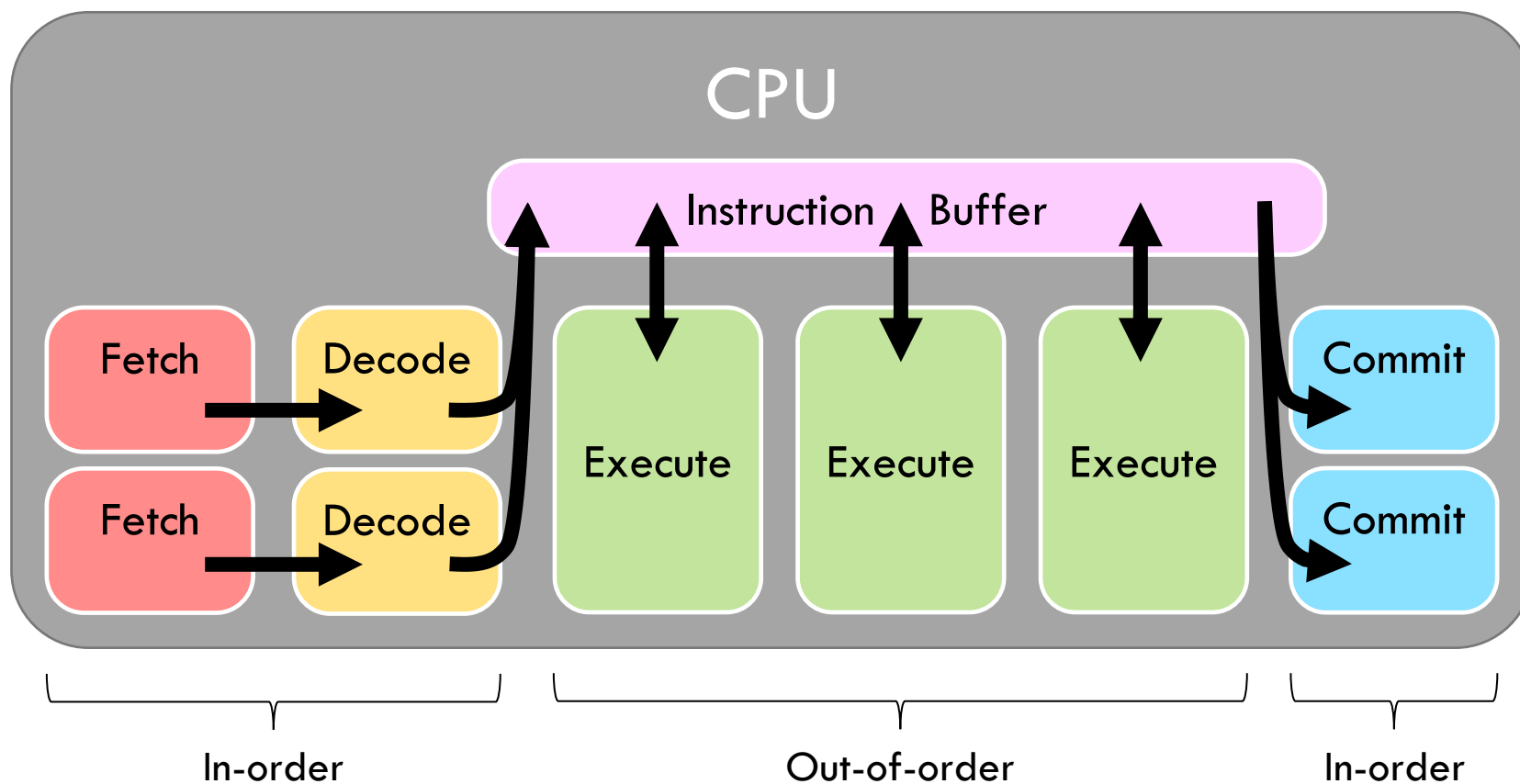


■ 乱序执行 (Out of order execution)

Superscalar Issue (Pentium)

Cycle	1	2	3	4	5	6	7	8	9
Instr ₁	Fetch	Decode	Execute			Write			
Instr ₂	Fetch	Decode	Wait			Execute	Write		
Instr ₃		Fetch	Decode	Execute	Write				
Instr ₄		Fetch	Decode	Wait			Execute	Write	
Instr ₅			Fetch	Decode	Execute	Write			
Instr ₆			Fetch	Decode	Execute	Write			
Instr ₇				Fetch	Decode	Execute	Write		
Instr ₈				Fetch	Decode	Execute	Write		

■ 超标量乱序执行 (Superscalar OoO) 微体系架构





■ 预测执行 (Speculative Execution)

- 预测执行 (投机执行) 是计算机系统设计中常用的一种优化技术，通过预先执行一些尚未确定是否要执行的任务来减少系统“卡顿”，从而提高系统高性能。
 - 分支预测 (Branch prediction) — 指令级的预测执行技术，优化指令流水线
 - Thread Level Speculation (TLS)，即Speculative Multithreading (SpMT) — 线程级的预测执行技术，优化多线程并行执行过程或软件流水线
- 分支预测：预测分支跳转并投机执行分支指令之后的指令
 - 现代分支预测器有很高的准确率 (>95%)，预测失败开销显著减少
 - 需要硬件支持: 分支历史记录表，分支目标缓冲区 (BTB) 等



■ 预测执行 (Speculative Execution)

➤ 预测失败怎么办？

- 顺序执行的机器: 清空流水线中分支指令之后的指令
- 乱序执行的机器: 回退到顺序执行状态进行处理；或利用检查点技术，在分支执行前建立检查点（相关寄存器和内存），预测失败恢复检查点

➤ 预测执行在现代处理器中被广泛应用，对性能有重要影响

➤ 2018年1月公布的CPU安全漏洞Meltdown（熔断）和Spectre（幽灵）利用了乱序和预测执行

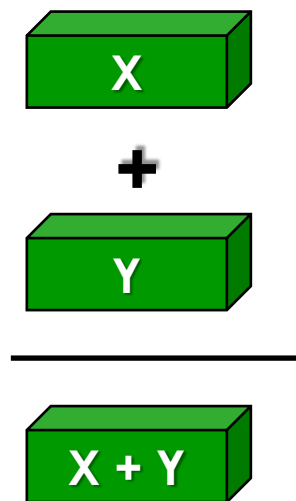




■ 向量指令 (SIMD)

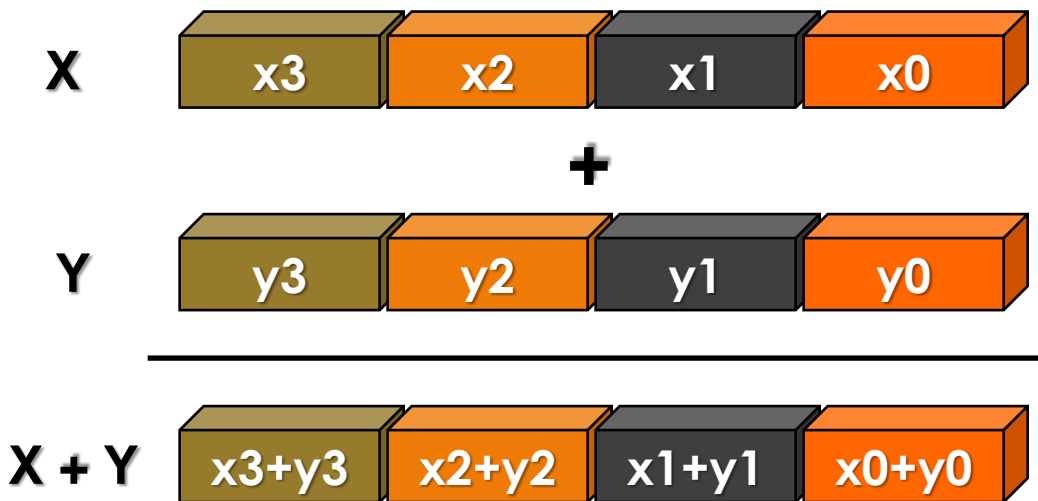
Scalar processing

- traditional mode



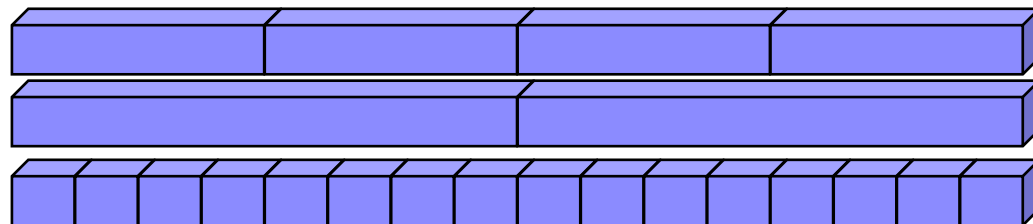
SIMD processing: vectors

- Sandy Bridge: AVX (256 bit)
- Haswell: AVX2 (256 bit w/ FMA)
- KNL: AVX-512 (512 bit)



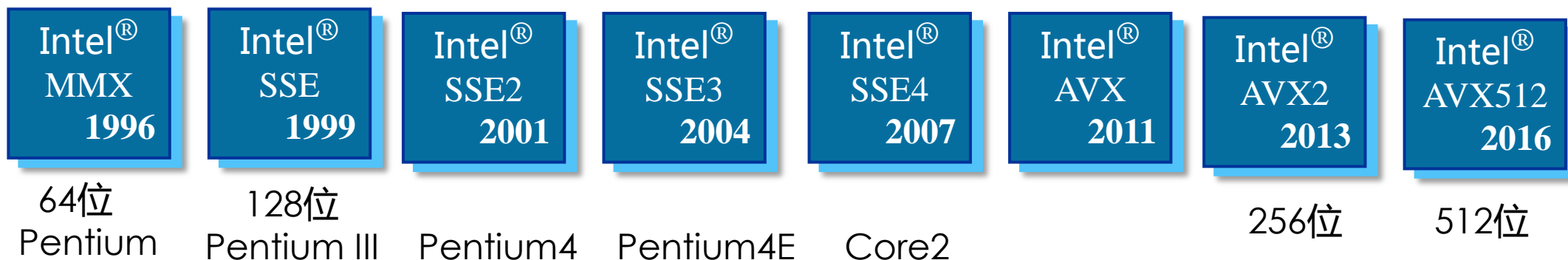
■ Intel 向量指令

- SSE2 指令数据类型: 任意总长为16 bytes 的数据



4x floats
2x doubles
16x bytes

- 硬件处理数据类型的不同，进行并行运算
- 需要合理组织数据，例如，数据应连续，并且缓存行对齐（cache aligned）

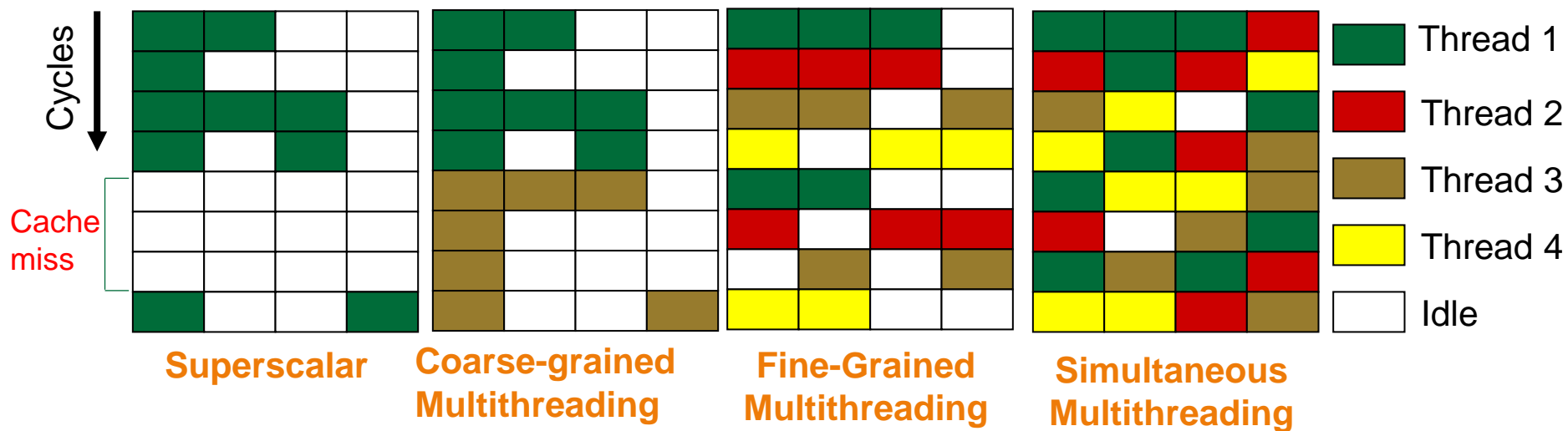




■ 硬件多线程 (Hardware Multithreading)

- 粗粒度多线程 (**Course-grained multithreading**) : 在遇到比较耗时的“卡顿”(costly stalls)时切换线程 (例如, 2级cache miss)
 - 优点: 调度开销少, 不会拖慢单独一个线程的执行
 - 缺点: 无法隐藏短时间的stall
- 细粒度多线程 (**Fine-grained multithreading**) : 每个时钟周期都可以切换线程
 - 优点: 能够隐藏任意时间长短的stall
 - 缺点: 调度开销大, 会拖慢单独线程执行; 没有完全发挥多发射架构的能力
- 同时多线程 (**Simultaneous Multithreading, SMT**) : 一个时钟周期可同时发射执行不同线程的指令, 同时发挥了线程级并行性 (TLP) 和指令级并行性 (ILP) , 最大化了指令执行部件利用率

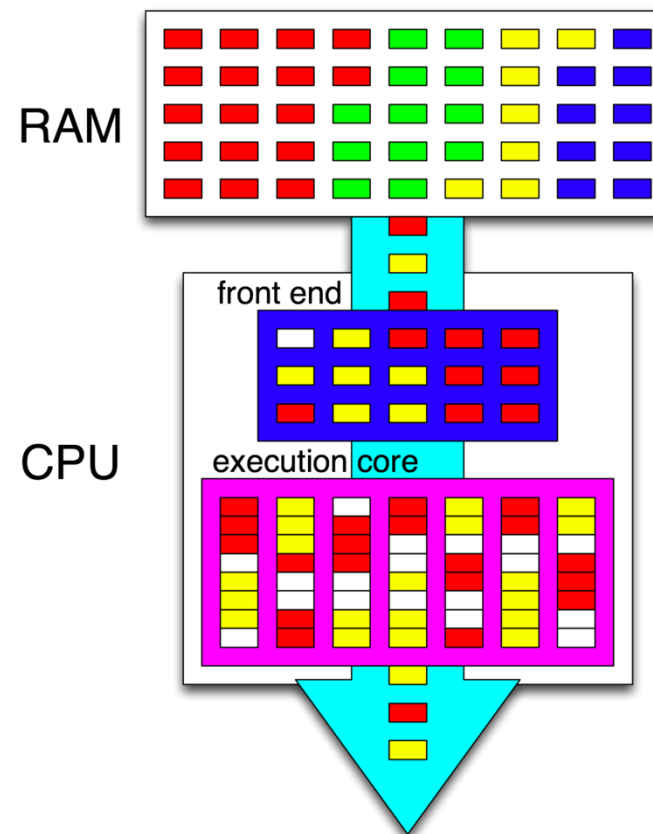
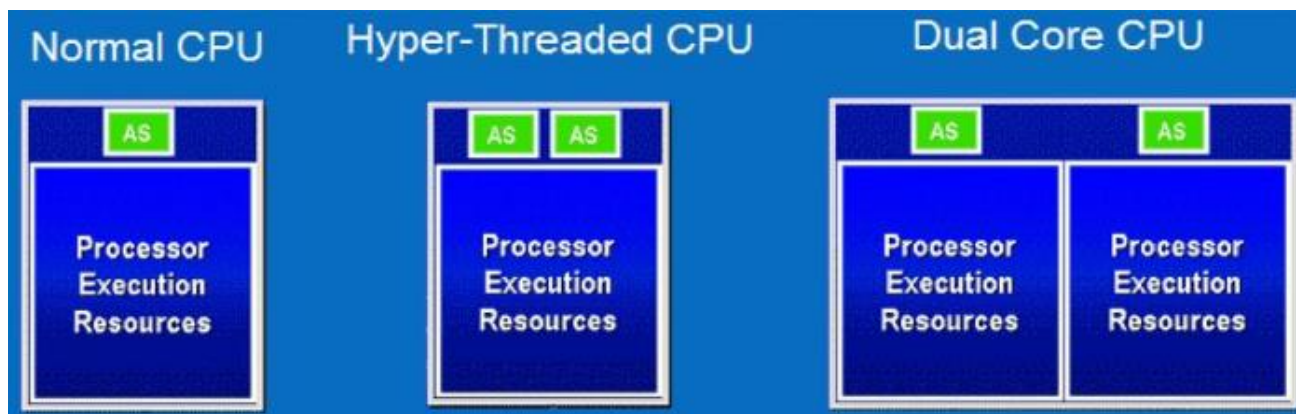
■ 硬件多线程 (Hardware Multithreading)



■ 硬件多线程 (Hardware Multithreading)

➤ Intel超线程技术 (Hyper-Threading, HT)

- 2002年开始应用于Pentium 4 HT，通过此技术，英特尔实现在一个实体CPU中，提供两个逻辑线程。
- 复制资源：寄存器等
- 共享资源：Caches (trace, L1, L2...) 和执行单元





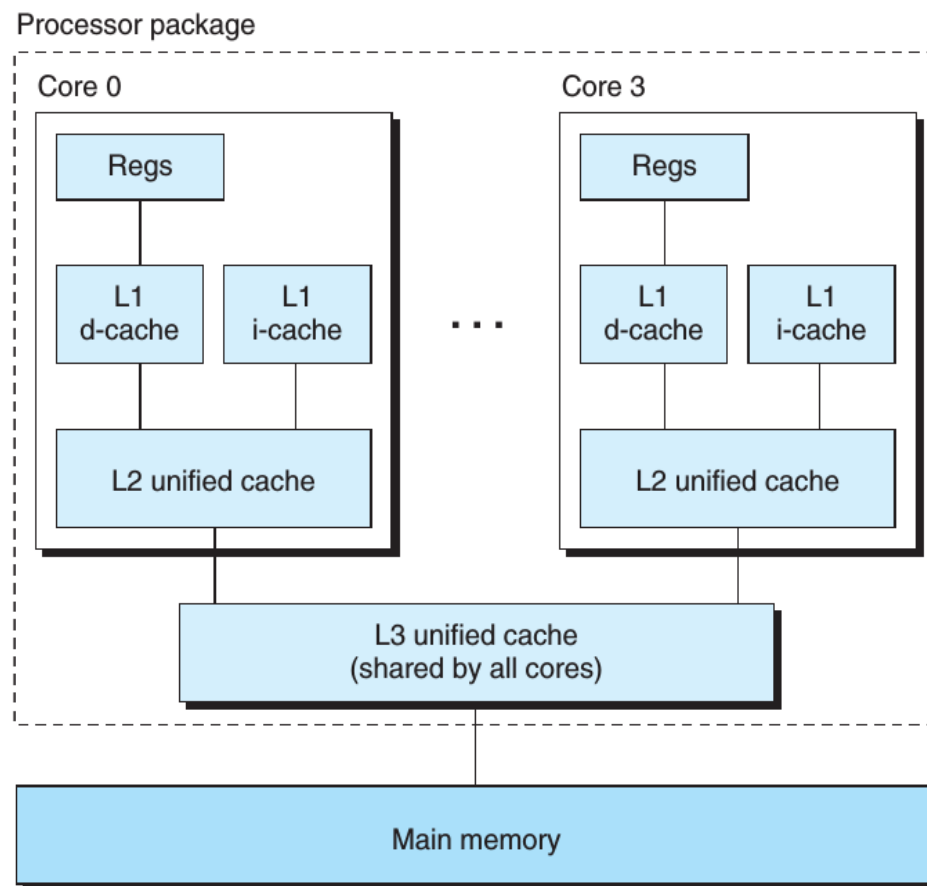
4

现代体系结构



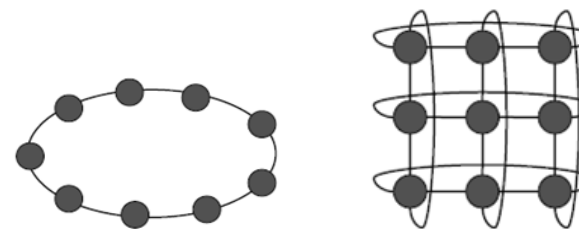
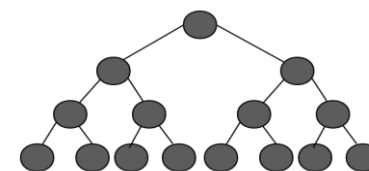
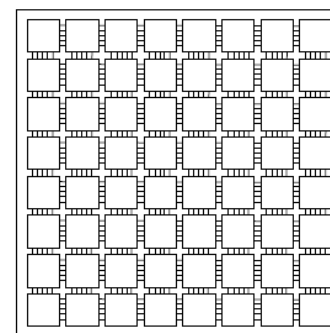
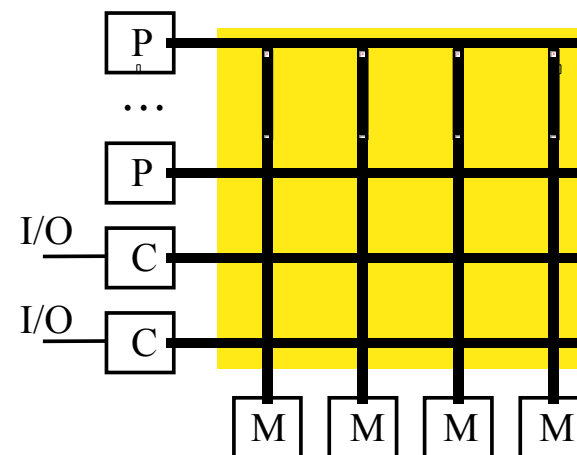
■ 多核处理器

- 推动微处理器性能不断提高的因素主要有两个：半导体工艺技术的飞速进步和体系结构的不断发展
- 单核主频的提高带来功耗的提高，功耗问题限制了单核处理器的发展
- 多核处理器把多个CPU（核心）集成到单个处理器芯片中，通过“横向扩展”（而非“纵向扩充”）提高性能



■ 多核处理器 — 核间通信机制

- 基于总线共享的Cache结构：多个内核拥有共享的二级或三级Cache，通过连接核心的总线进行通信。优点是结构简单，通信速度快，缺点是基于总线的结构可扩展性较差。
- 基于片上互连的结构：每个内核具有独立的处理单元和Cache，各个内核通过交叉开关或片上网络等方式连接在一起。各个内核间通过消息通信。这种结构的优点是可扩展性好，数据带宽有保证；缺点是硬件结构复杂，且软件改动较大。





■ 多核处理器 — Cache Coherency

- Cache对多核处理器性能的关键作用：减少平均数据访问时间；减少核间通信的带宽需求。
- 各内核私有Cache带来的问题：一个变量在多个内核中有多个副本；一个内核对变量的修改可能不被其它内核察觉。 \Rightarrow Cache coherence 问题
- 如何解决Cache coherence 问题？
 - Cache一致性协议：MESI协议中每个Cache line有4个状态M(Modified)、E(Exclusive)、S(Shared)、I(Invalid)
 - Cache一致性协议的硬件实现：基于侦听（Snooping）和基于目录（Directory-based）

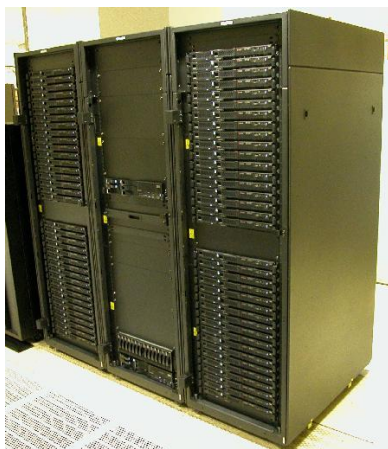


■ 多核处理器 — Memory Consistency

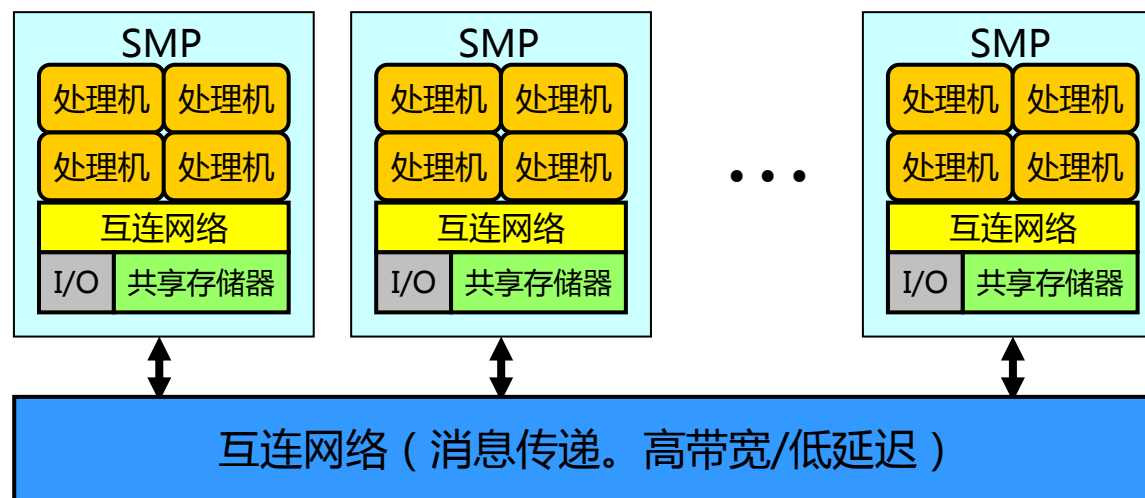
- 如果在每次读取某一数据项时都会返回该数据项的**最新写入值**，则称这个存储系统是一致的。
- 但是，什么是“**最新写入值**”？
 - 单线程可以由程序顺序确定；
 - 多个线程所在多个内核写同一个数据时，哪个是最新写入值？
- 要保证访存操作按照确定的顺序进行，也就是各个内核看到的访存顺序是一致的
 - Coherence保证对同一个内存地址写的顺序在各个内核来看都一样
 - Consistency保证对不同内存地址写的顺序在各个内核来看都一样

■ 多核集群

- 每个节点为共享存储器结构的SMP系统；
- 各SMP系统借助互连网络、通过消息传递机制相互通信；
- 是目前大规模并行处理（MPP）等系统普遍采用的结构。

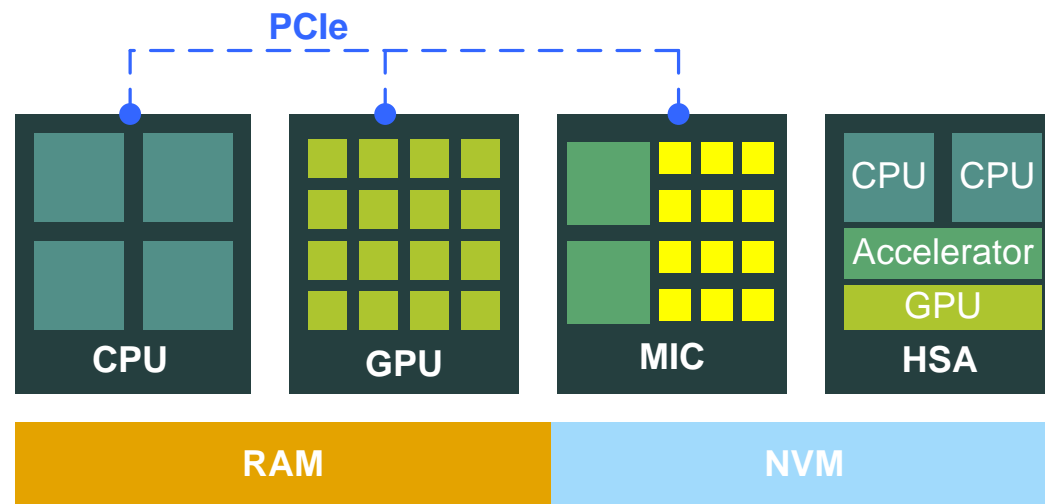


IBM Cluster 1350



■ 异构多核系统

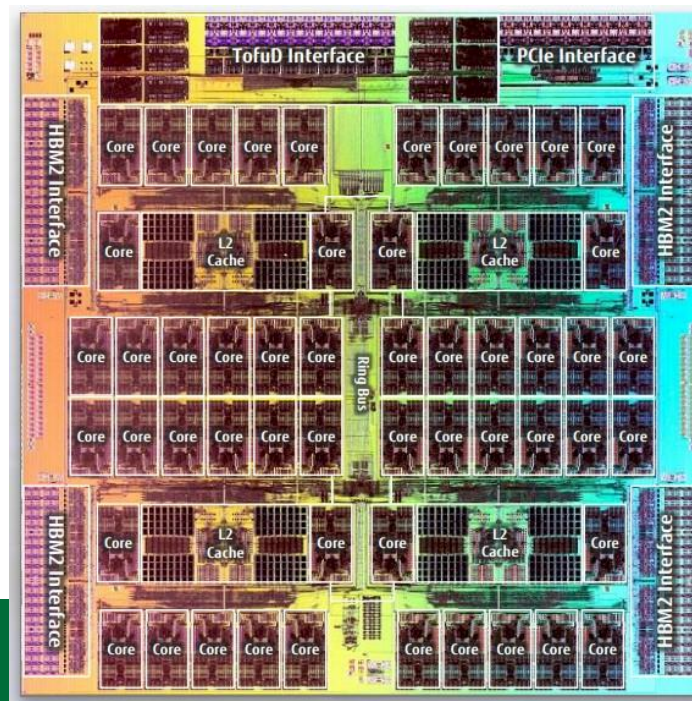
- 多核CPU与独立GPU/MIC等加速器通过PCIe总线连接
- CPU与GPU或其它加速器融合在同一个芯片内的系统，也就是HSA架构，如AMD的APU处理器
- 传统DRAM与新型NVM（Non-Volatile Memory）的异构存储



■ 超算系统

➤ 富岳 (Fugaku)

- 制造商：富士通
- 所在位置：日本理化学研究所
- 核数：7,630,848
- 内存：5,087,232 GB
- 处理器：A64FX 48C 2.2GHz



■ 超算系统

➤ Summit

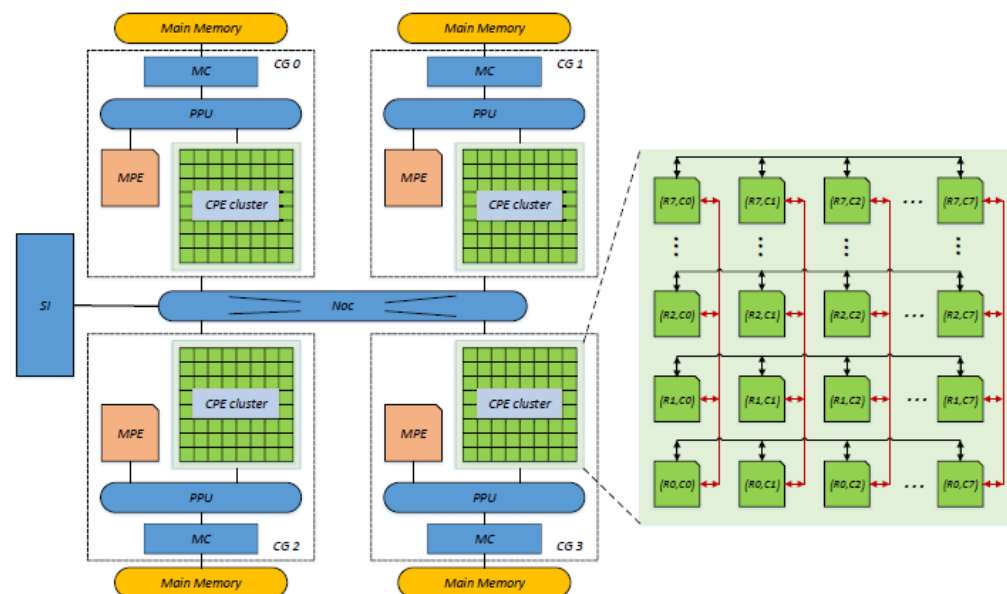
- 制造商：IBM
- 所在位置：美国能源部橡树岭国家实验室
- 核数：2,414,5928
- 内存：2,801,664 GB
- 处理器：IBM POWER9 22C 3.07GHz
- 加速器：NVIDIA VOLTA GV100
- GPU数量：27648



■ 超算系统

➤ 神威·太湖之光 (Sunway TaihuLight)

- 制造商：国家并行计算机工程技术研究中心
- 所在位置：国家超级计算无锡中心
- 核数：10,649,600
- 内存：1,310,720 GB
- 处理器：Sunway SW26010 1.45GHz
- 申威26010处理器核组数：163840
- 计算节点数：40960



■ 超算系统

➤ 天河二号A (TIANHE-2A)

- 制造商：国防科学技术大学
- 所在位置：国家超级计算广州中心
- 核数：4,981,760
- 内存：2,277,376 GB
- 处理器：Intel Xeon E5-2692 2.2GHz
- 加速器：Matrix-2000



Compute blade = Xeon part + Matrix-2000 part

4 Intel Xeon CPUs 4 FT Matrix-2000 2 Compute Nodes





北京理工大学
BEIJING INSTITUTE OF TECHNOLOGY

谢谢!

德以明理 学以精工