

### 3/ Chapter 1

9. 调查资料, 说明对不同微处理器架构进行性能分析和对比的方法是什么? Dhrystone 和 CoreMark 等评分是如何测得的?

答: 对不同微处理器架构进行性能分析和对比的方法包括:

#### 1) 基准测试 benchmarking

通过运行一些标准化的测试程序来评估处理器性能的方法。

常见的基准测试程序包括: Dhrystone, CoreMark, SPEC 等。

这些基准测试程序可以模拟实际应用场景下的运行情况。

从而提供一个相对准确的性能评估。

#### 2) 模拟器 simulator

用来模拟不同处理器架构的性能, 并提供了对处理器行为的深入理解。模拟器通常可以用于执行指令集仿真、循环级仿真或者系统仿真等不同级别的仿真。通过在模拟器中运行相同的测试程序, 可以比较不同处理器的性能差异。

### 3) 实际应用测试 real-world application testing

通过运行实际的应用程序来测试处理器的性能。并且提供提供最真实的性能数据。但由于应用程序的复杂性和系统环境等其他因素，很难评估(区分)处理器的性能和应用程序自身的性能。

Dhrystone 是一种 <sup>非主要</sup> 浮点运算和 <sup>主要</sup> 整数运算性能测试。主要测试处理器的整数运算能力，包括算术运算、逻辑运算和比较运算。测试用到包括数组操作、循环操作等。通过测试程序执行的时间和每秒钟可以执行多少条指令来计算。单位： $\frac{MIPS}{MIPS}$  执行多次取平均值。

CoreMark 是一种多线程测试程序，它可以测试处理器性能的整数和浮点运算性能，以及内存和系统性能。列表处理、矩阵操作、状态机。这个多线程测试程序涵盖了多个领域，如压缩、加密、排序、字符串处理等，测试任务间会有相互竞争和协作。

评分过程：

- ① 在测试平台上编译并执行 CoreMark 测试程序，记录测试程序的执行时间。
- ② 根据测试任务的数量和复杂度，计算 <sup>工作量</sup> CoreMark 调整系数因子。
- ③ 根据执行时间和工作量调整系数，计算指标 CoreMarks/s。
- ④ 多次执行，取平均值。



10. 层次化是计算机体系结构中的重要概念, 简述现代计算机系统中有哪些地方体现了层次化的设计特点? 有怎样的实际意义?

答: 层次化结构包括: (自上而下)

1) 应用程序层: 包括各种应用软件, 如办公软件、游戏、浏览器等。这是用户直接与计算机系统交互的层次。

2) 操作系统层: 介于应用程序层和硬件层之间的软件层, 管理和协调硬件资源, 为应用程序提供各种服务和接口。包括了进程管理、文件管理、内存管理、网络管理等多个模块。

3) 应用程序框架层: 是一种软件开发模型, 它为应用程序开发提供了基础设施和开发工具。一般包括各种库、API 和工具集等。

4) 编译器层: 介于高级语言和汇编语言之间的软件层。它将高级语言编写的源代码转化为机器指令, 以便计算机系统执行。  
*Machine Language*

5) 汇编语言层:

汇编语言层是介于机器语言和高级语言之间的软件层。

使用汇编语言编写的代码可以直接转化为机器指令, 比高级语言更接近硬件层。

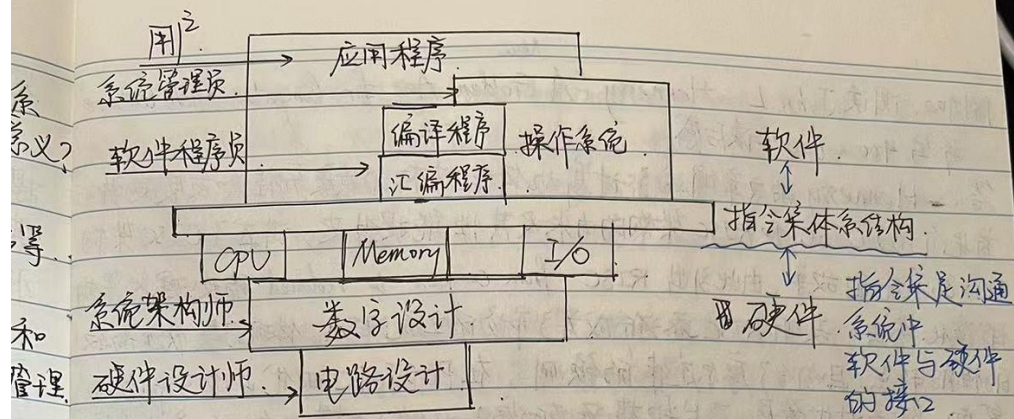
6) 指令集架构层:

定义了计算机系统的指令集和编程模型。规定了 CPU 的运算方式和操作方式, 包括寄存器、地址空间、指令集等...

7) 微体系结构层:

微体系结构层是硬件层次中最底层的一层。它定义了 CPU 内部的电路设计 (包括逻辑门、寄存器、ALU 等组件), 以及它们之间的连接方式和时序控制。





层次化的实际意义:

1) 分工合作:

计算机体系结构中的各个层次之间通过接口和规范相互连接, 实现了分工合作, 各层次只需要关注自己的功能和任务, 从而提高了系统的可维护性和可扩展性。

2) 性能提升:

通过在各个层次的优化, 可以提高系统整体的性能, 从而实现更高的运算速度和更好的用户体验。

3) 技术进步:

在每个层次中, 设计者可以采用不同的技术方案和算法, 从而使得计算机系统不断得到改进和升级, 满足不同领域的需求。

4) 标准化:

计算机体系结构的层次化设计使得不同厂家的计算机硬件和软件可以相互兼容和交换。各个层次之间的接口和规范都经过标准化, 从而使得不同厂家生产的计算机之间可以相互通信和共享数据。



附加2. 阅读 John L. Hennessy "A Golden Age for Computer Architecture"

并写 400 ~ 600 字读后感.

答: Hennessy 的文章阐述了计算机体系结构的发展历程和发展趋势. 首先介绍了 CISC 和 RISC 架构的由来及其性能提升史, 讲述了 CISC 架构的 x86 系统的故事, 由此引出 RISC from complex to reduced 的处理器架构的演化缘由, 表明 RISC 逐渐成为了市场的主流 ISA, 体现了其快速高效的性能特点, 且吻合了摩尔定律的预测, 在 POST-PC 时代占领了市场. 然而, 随着时代发展, 芯片规模逐渐缩小, 工艺上对其性能的提升终究达到瓶颈, 摩尔定律不再准确吻合, Dennard scaling (晶体管 density  $\uparrow$ , 每个晶体管功耗  $\downarrow$ , 每  $\text{mm}^2$  的能量消耗一定) 的定律同样不适用. 进入后摩尔时代, 目光逐渐转向了架构和指令集的升级优化以获得更优秀的性能. (架构优化是综合的考量, 要看全局的效益)

首先出现的是 ILP (指令集并行) 的想法, 但由于 ~~错误~~ 预测 misprediction 的可能会造成更严重的低效浪费而淘汰; 再发展到 ~~multicore~~ <sup>multicore</sup> 也即多核架构, 根据 Amdahl's law 知道这同样被比例  $F$  限制, 而且, 增加内核数量虽有利于提高工作负载, 但是由于拥有多个核而增加的活动开关元件仍会使整体功耗增加, 导致浪费产生. 在此之后, 自然想到可以通过控制工作区域和时间来降低非必要功耗, 进入 "暗硅" 时代, 分配能量, 以此获得高效性能表现. 不过, 在规模缩小, 性能提升, 数据海量, 云服务的背景下, 安全问题逐渐显露, 解决方案在之后会提及. (开源)

着眼于计算机体系结构的未来发展方向, 一个是减少高级语言, 改用 SIMD (single instruction multiple data), 单指令操作多数据来提升性能. 另一个是专业化, 发展 DSA (Domain Specific Architecture), 类似 "专人专事" 的想法, 在每一领域使性能最优化, 从而取得最佳的全局收益.



are: DSA<sup>同时</sup>还能结合 VLIW (very long instruction word) 和 ILP, 达到更优越的并行效果, 大幅提高运行效率; 并且, DSA 充分利用~~在~~层次化存储结构 (memory hierarchy), 避免 cache 的强局部性和低利用率的特点。DSA 的基础是 ~~DSA~~ DSL (Domain Specific Language), 可高效执行专业化操作。

此外, 人工智能相关领域 (DNN 等) 将会成为计算机体系结构的重要驱动因素之一, AI 将大幅提升其性能和效率。GPU, TPU, FPGA, CPU with SIMD 等新兴架构思维日趋成熟, 会大有作为。

最后是对开源架构 RISC-V 的赞美, 这种开源架构造福社区众多科技公司, 每个人都以专业化自己的 RISC-V 以达到目标效果, 同时也得到不断的维护和更新, 保证信息公开, 降低了安全风险, 以其 ISA 的简洁有效和格式的灵活广受好评欢迎, 是未来大热点之一, 敏捷硬件开发同样有很好的前景。

总而言之, John L. Hennessy 这篇文章提供了对计算机体系结构的发展历史和未来趋势的深入思考和分析, 观点和预测都与时代接轨, 很有前瞻性, 这让我对 computer architecture 的未来充满期待, 期待 Golden Age 的到来。