



国防特色教材·电子科学与技术

# 新一代计算机体系结构

XINYIDAI JISUANJI TIXI JIEGOU

李静梅 吴艳霞 主 编  
付 岩 张春生 副主编

北京航空航天大学出版社

北京理工大学出版社 哈尔滨工业大学出版社  
哈尔滨工程大学出版社 西北工业大学出版社



国防特色教材·电子科学与技术

# 新一代计算机体系结构

李静梅 吴艳霞 主 编  
付 岩 张春生 副主编

北京航空航天大学出版社

北京理工大学出版社 哈尔滨工业大学出版社  
哈尔滨工程大学出版社 西北工业大学出版社

## 内 容 简 介

本书作为“十一五”国防特色规划学科专业教材,在介绍计算机系统结构的基本概念、原理、结构和分析方法的基础上,着重阐述了计算机系统的并行化技术、片上多核技术,旨在帮助学生在建立计算机系统完整概念的基础上,充分掌握计算机系统结构的最新研发思想和技术,了解目前最新研发技术领域。

本书可作为高等院校计算机专业高年级本科生或研究生的教材,也可供从事计算机体系结构设计或嵌入式系统设计的工程技术人员参考。

## 图书在版编目(CIP)数据

新一代计算机体系结构 / 李静梅, 吴艳霞主编. --

北京: 北京航空航天大学出版社, 2010. 8

ISBN 978-7-5124-0172-3

I. ①新… II. ①李… ②吴… III. ①计算机体系结构—高等学校—教材 IV. ①TP303

中国版本图书馆 CIP 数据核字(2010)第 148557 号

版权所有, 侵权必究。

## 新一代计算机体系结构

李静梅 吴艳霞 主 编

付 岩 张春生 副主编

责任编辑 张少扬 孟 博

\*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号(邮编 100191) <http://www.buaapress.com.cn>

发行部电话:(010)82317024 传真:(010)82328026

读者信箱: bhpess@263.net 邮购电话:(010)82316936

涿州市新华印刷有限公司印装 各地书店经销

\*

开本: 787×960 1/16 印张: 17.5 字数: 392 千字

2010 年 8 月第 1 版 2010 年 8 月第 1 次印刷 印数: 3 000 册

ISBN 978-7-5124-0172-3 定价: 32.00 元

# 前 言

《新一代计算机体系结构》是“十一五”国防特色学科专业教材,主要作为高等学校计算机专业研究生及本科生“计算机系统结构”课程的通用教材。为了让本科生也易于接受,在编写时力求做到深入浅出、通俗易懂。

计算机硬件技术与计算机软件技术互相制约、互相影响,促进了两种技术的良性发展,从而使计算机系统结构呈现出迅速发展的态势。本书的内容主要涉及计算机体系结构的概念、结构、机制及发展,既介绍了计算机系统结构方面已经成熟的技术,也介绍了国内外最新的研究成果。

作者本着三个出发点来编写此书:

① 保证本书内容的先进性。在介绍计算机系统设计所必须掌握知识的基础上,以提高效率和系统优化为目标,介绍了现阶段计算机体系结构的先进技术方法,如指令级并行技术、同时多线程技术等。

② 实例详细,浅显易懂。各种代码、实例等具体示例的详细分析和说明有助于自学者阅读。

③ 清晰地介绍技术发展的脉络。本书在介绍原理或算法时,会介绍该技术的开发背景、应用情况以及后续发展,有利于开拓读者实践和创新的思维方式。最终向读者清晰而全面地展现现阶段计算机体系结构的原理和本质,紧密结合行业发展,促使计算机系统结构课程内容向实用化和新理念方向延伸。

本书内容分为7章。第1章论述了计算机系统的概念及其设计原则,通过对冯·诺依曼计算机模型指令集的分类,简要阐述计算机系统发展的脉络,最后简单介绍几种先进的微体系结构;第2章论述了流水线的基本概念、分类及性能计算方法,以DLX模型为实例,详细描述了流水线执行细节,最后分析了流水线中相关和冲突的问题;第3章介绍了指令级并行的概念,详细阐述了实现指令级并行的关键技术或算法;第4章为线程级并行技术,阐述了多线程技术的概念及分类,详细论述同时多线程技术,并以超线程技术为例,介绍其工作原理;第5章和第6章分别介绍了超流水、超标量以及超长指令字处理器,主要阐述了关键技术的主要原理,并且介绍了几种典型处理器的结构和特点;第7章为片上多核处理

器的相关知识,首先概述其体系结构特点,接下来叙述了芯片组和操作系统对其的支持,最后介绍了几种典型的片上多核处理器架构。

本书在编写过程中多次得到有关领导部门、兄弟院校及 Intel 公司的专家、教授和同志们的鼓励和支持,有的曾提出宝贵的建议,在此表示感谢。

本书承蒙郑纬民、顾国昌二位专家审阅,并对书稿提出了许多宝贵的修改意见,在此一并表示衷心的感谢。

最后对 Intel 公司在哈尔滨工程大学计算机体系结构精品课程建设方面的大力支持表示衷心的感谢。

由于作者的水平有限,书中难免有错误和不妥之处,恳请广大读者批评指正。

编 者

2010 年 3 月

# 目 录

|                          |    |
|--------------------------|----|
| 第 1 章 概 述                | 1  |
| 1.1 计算机系统结构的基本概念         | 1  |
| 1.1.1 多级层次结构             | 1  |
| 1.1.2 系统结构、组成和实现之间的关系    | 4  |
| 1.2 计算机系统结构的设计方法         | 6  |
| 1.2.1 计算机系统的设计原则         | 6  |
| 1.2.2 计算机系统的设计思路及步骤      | 10 |
| 1.3 软件、应用、器件的影响          | 12 |
| 1.4 计算机系统结构的分类           | 16 |
| 1.5 基于冯·诺依曼计算机模型的指令集分类   | 20 |
| 1.5.1 CISC 体系结构          | 21 |
| 1.5.2 RISC 体系结构          | 22 |
| 1.5.3 CISC 和 RISC 混合体系结构 | 24 |
| 1.5.4 EPIC 体系结构          | 25 |
| 1.6 先进的微体系结构             | 27 |
| 1.6.1 多核处理器              | 27 |
| 1.6.2 流处理器               | 33 |
| 1.6.3 PIM                | 39 |
| 1.6.4 可重构计算              | 41 |
| 习 题                      | 47 |
| 第 2 章 流水线技术              | 50 |
| 2.1 流水线的基本概念             | 50 |
| 2.1.1 什么是流水线             | 50 |
| 2.1.2 流水线的分类             | 51 |
| 2.2 流水线的性能指标             | 55 |
| 2.2.1 吞吐率                | 56 |
| 2.2.2 加速比                | 58 |
| 2.2.3 效 率                | 58 |
| 2.3 DLX 的基本流水线           | 59 |

|                                  |            |
|----------------------------------|------------|
| 2.3.1 DLX 指令集结构 .....            | 59         |
| 2.3.2 基本的 DLX 流水线 .....          | 61         |
| 2.3.3 DLX 流水线各级的操作 .....         | 63         |
| 2.3.4 DLX 流水线处理机的控制 .....        | 65         |
| 2.4 流水线的相关与冲突 .....              | 74         |
| 2.4.1 流水线相关 .....                | 74         |
| 2.4.2 流水线冲突 .....                | 77         |
| 习 题 .....                        | 87         |
| <b>第 3 章 指令级并行</b> .....         | <b>90</b>  |
| 3.1 指令级并行的概念 .....               | 90         |
| 3.2 循环展开 .....                   | 91         |
| 3.2.1 循环展开的原理 .....              | 91         |
| 3.2.2 循环展开的特点 .....              | 93         |
| 3.3 动态指令调度 .....                 | 94         |
| 3.3.1 静态指令调度与动态指令调度 .....        | 94         |
| 3.3.2 动态指令调度的基本思想 .....          | 95         |
| 3.3.3 动态指令调度算法：记分板 .....         | 96         |
| 3.3.4 动态指令调度算法：Tomasulo 算法 ..... | 108        |
| 3.4 动态分支预测 .....                 | 125        |
| 3.4.1 采用分支预测表 .....              | 126        |
| 3.4.2 采用分支目标缓冲器 .....            | 128        |
| 3.4.3 基于硬件的推断执行 .....            | 131        |
| 3.4.4 先进的分支预测技术 .....            | 137        |
| 习 题 .....                        | 139        |
| <b>第 4 章 线程级并行</b> .....         | <b>142</b> |
| 4.1 多线程技术发展背景 .....              | 142        |
| 4.2 线程概念 .....                   | 144        |
| 4.2.1 用户级线程 .....                | 144        |
| 4.2.2 内核级线程 .....                | 145        |
| 4.2.3 硬件线程 .....                 | 148        |
| 4.3 单线程处理器 .....                 | 148        |
| 4.4 多线程技术概述 .....                | 151        |
| 4.4.1 阻塞式多线程 .....               | 152        |
| 4.4.2 交错式多线程 .....               | 153        |

|                                 |            |
|---------------------------------|------------|
| 4.4.3 同时多线程 .....               | 154        |
| 4.5 同时多线程技术 .....               | 155        |
| 4.5.1 超级线程技术概述 .....            | 156        |
| 4.5.2 超线程技术概述 .....             | 157        |
| 4.6 超线程技术 .....                 | 159        |
| 4.6.1 超线程技术的工作原理 .....          | 159        |
| 4.6.2 实现超线程的前提条件 .....          | 160        |
| 4.6.3 Intel 的超线程技术 .....        | 161        |
| 4.7 同时多线程技术存在的挑战 .....          | 166        |
| 习 题 .....                       | 167        |
| <b>第 5 章 超流水、超标量处理器 .....</b>   | <b>169</b> |
| 5.1 超级流水线处理器 .....              | 169        |
| 5.1.1 指令执行时序 .....              | 170        |
| 5.1.2 MIPS R4000 超级流水线处理器 ..... | 170        |
| 5.1.3 超级流水线的弊端 .....            | 171        |
| 5.2 标量处理器 .....                 | 172        |
| 5.2.1 标量流水线性能上限 .....           | 173        |
| 5.2.2 性能损失 .....                | 173        |
| 5.3 超标量处理器 .....                | 174        |
| 5.3.1 超标量流水线典型结构 .....          | 174        |
| 5.3.2 指令执行时序 .....              | 175        |
| 5.3.3 超标量技术 .....               | 175        |
| 5.3.4 超标量处理器性能 .....            | 179        |
| 5.3.5 龙芯 2F 超标量处理器 .....        | 180        |
| 5.4 其他三种典型的超标量处理器 .....         | 187        |
| 5.4.1 MIPS R10000 .....         | 187        |
| 5.4.2 Alpha 21164 .....         | 189        |
| 5.4.3 AMD K5 .....              | 190        |
| 习 题 .....                       | 191        |
| <b>第 6 章 超长指令字处理器 .....</b>     | <b>192</b> |
| 6.1 概 述 .....                   | 192        |
| 6.1.1 引 言 .....                 | 192        |
| 6.1.2 基本概念 .....                | 195        |
| 6.1.3 传统方法的不足 .....             | 198        |

|                              |            |
|------------------------------|------------|
| 6.2 精确中断技术 .....             | 199        |
| 6.2.1 概 述 .....              | 199        |
| 6.2.2 RP 缓冲机制 .....          | 200        |
| 6.2.3 RRP 缓冲机制 .....         | 202        |
| 6.3 RFCC - VLIW 结构 .....     | 205        |
| 6.3.1 概 述 .....              | 205        |
| 6.3.2 寄存器堆结构 .....           | 205        |
| 6.3.3 代价分析 .....             | 206        |
| 6.3.4 性能分析 .....             | 207        |
| 6.3.5 THUASDSP2004 处理器 ..... | 209        |
| 6.4 MOSI 体系结构 .....          | 218        |
| 6.4.1 概 述 .....              | 218        |
| 6.4.2 MOSI 微体系结构 .....       | 219        |
| 6.4.3 性能分析 .....             | 222        |
| 6.5 基于 VLIW 的多核处理器 .....     | 226        |
| 6.5.1 华威处理器 .....            | 226        |
| 6.5.2 安腾处理器 .....            | 230        |
| 习 题 .....                    | 235        |
| <b>第 7 章 片上多核处理器 .....</b>   | <b>236</b> |
| 7.1 片上多核体系结构概述 .....         | 236        |
| 7.1.1 片上多核体系结构简介 .....       | 236        |
| 7.1.2 多核体系结构和超线程技术的区别 .....  | 239        |
| 7.1.3 多核多线程体系结构 .....        | 241        |
| 7.2 芯片组对多核的支持 .....          | 242        |
| 7.2.1 EFI 概述 .....           | 242        |
| 7.2.2 EFI 对多核处理器的初始化 .....   | 244        |
| 7.2.3 EFI 对多核操作系统的支持 .....   | 247        |
| 7.3 操作系统对多核的支持 .....         | 247        |
| 7.4 典型片上多核架构 .....           | 248        |
| 7.4.1 异构多核处理器 .....          | 248        |
| 7.4.2 同构多核处理器 .....          | 255        |
| 习 题 .....                    | 267        |
| <b>参考文献 .....</b>            | <b>268</b> |

# 第 1 章 概 述

根据摩尔定律,集成电路芯片上所集成的晶体管和电阻器等的数目,每隔 18 个月就翻一番。如何更合理地利用新器件,最大限度地发挥其潜力,设计并构成综合性能指标最佳的计算机系统,单纯依靠器件的变革是不能解决的,还要靠计算机系统结构上的不断改进。计算机系统结构的发展不断改变人们对计算机的整体认识。计算机系统可以看成是按功能划分的多级层次结构。本章首先给出计算机系统结构的定义,并说明结构、组成、实现三者的含义和关系;同时,从计算机系统多级层次结构的角度有助于更好地认识计算机系统设计的基本原则与目标;接下来,通过对冯·诺依曼计算机模型指令集的分类,简要阐述计算机系统发展的脉络,最后简单介绍几种先进的微体系结构。

## 1.1 计算机系统结构的基本概念

计算机系统由紧密相关的硬件和软件组成,怎样从整体上来认识和分析它呢?一种观点是从使用语言的角度上将计算机系统看成是按功能划分的多级层次结构。

### 1.1.1 多级层次结构

随着计算机的发展,计算机语言经历了机器语言(二进制机器指令系统)、汇编语言、高级语言、应用语言这样一个从低级向高级的发展过程,后者均以前者为基础,又比前者的功能更强,使用更方便。从这个意义上讲,计算机语言可以分成若干层或若干级,最低层的语言功能最简单。对使用某一层语言编程的程序员来说,只要遵守该级语言的规定,所编写出的程序总是可以在机器上运行并获得结果,而不必考虑程序在机器中究竟是怎样执行的,就好像有了一台直接使用这种语言作为其机器语言的计算机一样。

而实际上,只有二进制机器指令是与机器硬件直接对应,并被其直接识别和执行的。然而使用机器语言既不方便也无法适应解题需要,更不利于计算机应用范围的扩大。汇编语言是一种符号式程序语言,给程序员编程提供了方便,尽管其每个语句基本上与机器指令对应,却不能被机器硬件直接识别和执行。

那么为什么汇编语言程序可以在机器上运行并获得结果,就好像对汇编语言程序员来说有了一台用汇编语言作为其机器语言的机器呢?可以把这想象成在使用二进制机器语言的实际机器级之上出现了用汇编语言作为机器语言的一级“虚拟”机器,如图 1-1 所示,这样从功能上计算机系统就被看成是一个由虚拟机器 M2 和实际机器 M1 构成的二级层次结构。汇编

语言程序员为了能正确编程,只需要熟悉面向它的虚拟机器 M2 即可,不用了解实际机器级 M1。在计算机系统上运行汇编语言(L2)源程序,应先将源程序完整地经汇编程序变换成等效的机器语言(L1)目标程序,而后再在实际机器上执行目标程序以获得结果。

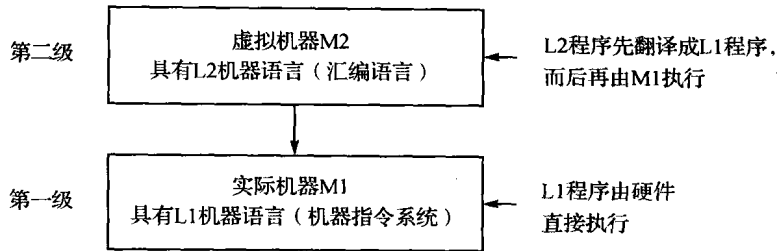


图 1-1 汇编虚拟机的实现

在这种变换模式的递推下,出现了面向题目和过程的高级语言。

这种层次概念还可以引伸到机器内部,对于采用微程序控制的机器,每条机器指令对应于一串微指令(一段微程序),每条微指令执行一些如按照各种数据传送通路之类的最基本操作。这样又可以把实际机器级分解成传统机器级和微程序机器级的二级层次结构。微程序机器级的机器语言是微指令系统,当执行到某条机器指令时控制转入执行相应的一串微指令,实现完这条机器指令后,再由程序内的下条机器指令控制转入实现它的下一串微指令。

翻译和解释是语言实现的两种基本技术。解释比翻译费时,但省存储空间。对于微程序控制的机器,在高级语言的实现过程中,先把高级语言源程序经编译程序翻译成传统的机器语言程序,而后再经微程序对每条机器指令进行解释来实现。

那么操作系统应处在这个层次结构中的什么位置呢?从实质上来看,操作系统是传统机器的引伸,它要提供传统机器所没有,但为汇编语言和高级语言的使用和实现所需的某些基本操作和数据结构,它在许多机器上是经机器语言程序解释实现的。另外在高级语言机器级之上还可以有应用语言虚拟机器,这种虚拟机所用的语言是面向某种应用环境的应用语言。综上所述,一个现代的计算机系统可以从功能上看成是如图 1-2 所示形式的多级层次结构。

虚拟机器不一定全部由软件实现,有些操作也可以用固件或硬件实现,如操作系统中的某些命令可以由比它低两级的微程序解释,甚至可以设想直接用微程序或硬件来实现高级语言机器。采用何种实现方式,要从整个计算机系统的效率、速度、造价、资源的使用状况等方面全面考虑,而且要软件、硬件(包括固件)综合平衡。实际上,软件和硬件在逻辑功能上是等效的。具有相同功能的计算机系统,其软硬件的功能分配可以在很宽的范围内变化。这种分配比例是随不同时期及同一时期的不同机器而动态变化的。由于软硬件紧密相关,有时软硬件界面是很模糊的,一个功能很难说哪些是由硬件完成,哪些是由软件完成的。

例如,在计算机中实现十进制乘法这一功能,既可以用硬件来实现,也可以用软件来完成。硬件实现方法:设计十进制乘法机器指令,用硬件电路来实现该指令,其特点是完成这一功能

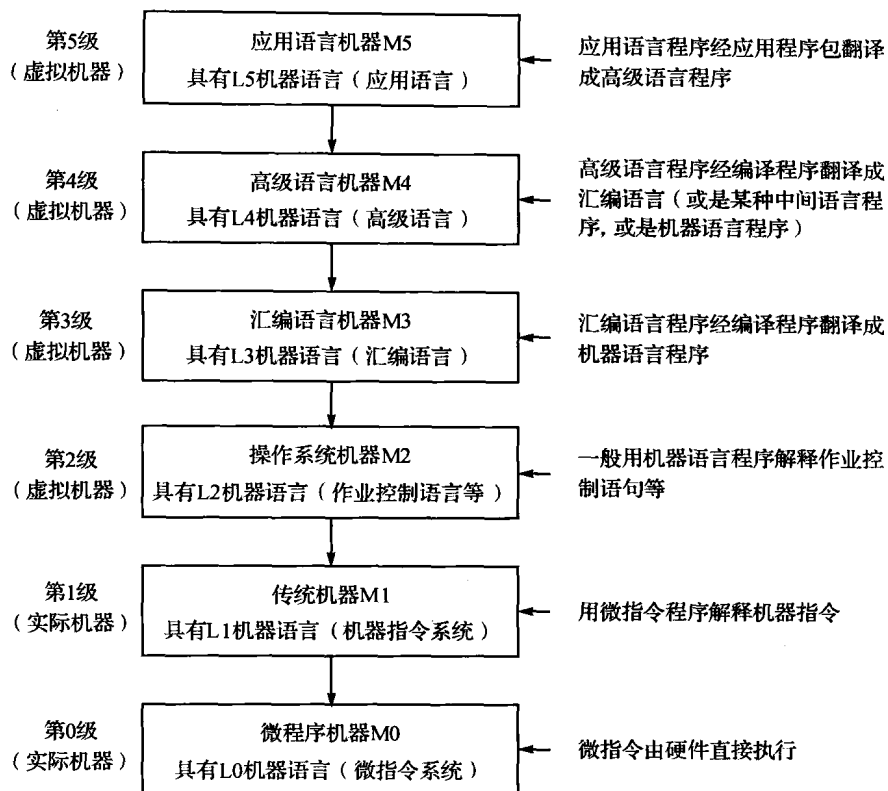


图 1-2 计算机系统的多级层次结构

的速度快,但需要更多的硬件。软件实现方法:通过编程采用加法、移位等指令来实现,其特点是实现的速度慢,但不需增加硬件。

最后,从概念和功能上把一个复杂的计算机系统看成是由多级构成的层次结构有很多优点:

① 有利于正确理解软件、硬件、固件在计算机中的地位和作用,也有助于理解各种语言的实质和实现途径。

② 直接或间接地推动了计算机系统结构的发展,发展了所谓的高级语言机器,操作系统计算机器。

③ 发展了多处理机系统、分布处理系统、嵌入式系统和计算机网络等系统结构。

## 1.1.2 系统结构、组成和实现之间的关系

### 1. 计算机系统结构

计算机系统结构也称为计算机体系结构,这个名词从 20 世纪 70 年代开始被广泛采用,但由于器件技术的发展及计算机软硬件界面的动态变化,使得对它的概念的理解不尽一致。

那么计算机系统结构到底指的是什么? 是一台计算机的外表,还是指一台计算机内部的一块块板卡安放结构? 1964 年, G. M. Amdahl 在介绍 IBM 360 系统时提出: 计算机系统结构就是计算机的机器语言程序员或编译程序编写者所看到的外特性,即程序员编写能在机器上正确运行的程序时必须了解到的概念性结构和功能特性。所谓外特性,就是计算机的概念性结构和功能特性。用一个不甚恰当的比喻,动物的“系统结构”是指什么呢? 它的概念性结构和功能特性就相当于动物的器官组成及其功能特性,比如鸡有胃,胃可以消化食物。至于鸡的胃是什么形状的、鸡的胃部由什么组成就不是“系统结构”研究的问题了。

计算机系统结构的外特性,一般应包括以下几个方面:

- ① 指令系统: 包括机器指令的操作类型和格式,指令间的排序方式和控制机构等;
- ② 数据表示: 包括硬件能直接识别和处理的数据类型和格式等;
- ③ 操作数的寻址方式: 包括最小寻址单位,寻址方式的种类、表示和地址计算等;
- ④ 寄存器的构成定义: 包括操作数寄存器、变址寄存器、控制寄存器及某些专用寄存器的定义、数量和使用约定;
- ⑤ 中断机构和例外条件: 包括中断的类型、中断分级、中断处理程序的功能和入口地址等;
- ⑥ 存储体系和管理: 包括最小编址单位、编址方式、主存容量和最大可编址空间等;
- ⑦ I/O 结构: 包括 I/O 的联结方式,设备的访问方式,数据的“源”、“目的”及数据传送量,操作的结束与出错指示等;
- ⑧ 机器工作状态定义和切换: 如管态、目态等的定义和切换;
- ⑨ 信息保护: 包括保护方式、硬件对信息保护的支持等。

关于计算机系统的多层次结构,可以用人与计算机系统作对比,这种联系不是很恰当,但可以给大家一个更直观的了解,如表 1-1 所示。

表 1-1 计算机系统结构的直观了解

| 计算机系统  | 人      |
|--------|--------|
| 应用语言级  | 为人民服务级 |
| 高级语言级  | 读书、学习级 |
| 汇编语言级  | 语言、思维级 |
| 操作系统级  | 生理功能级  |
| 传统机器级  | 人体器官级  |
| 微程序机器级 | 细胞组织级  |
| 电子线路级  | 分子级    |

## 2. 计算机组成与实现

对计算机组成与实现定义之前,要先介绍一下计算机系统结构的内特性,计算机系统结构的内特性就是将那些外特性加以“逻辑实现”的基本属性。这里说的“逻辑实现”就是在逻辑上如何实现这种功能,比如“设计者”给鸡设计了一个一定大小的胃,这个胃的功能是消化食物,这就是鸡系统的某一外特性,那么怎么消化呢,就要通过鸡吃进食物和砂石,再通过胃的蠕动,依靠砂石的研磨来消化食物,这里的吃和蠕动等操作就是内特性。

计算机实现,也就是指计算机组成的物理实现。它主要着眼于器件技术和微组装技术。拿上面的例子来说,这个胃由哪些组织组成,几条肌肉和神经来促使它运动就是“鸡实现”。

据此可以分清计算机系统的外特性、内特性以及物理实现之间的关系。在所有系统结构的特性中,指令系统的外特性是最关键的。因此,计算机系统结构有时就简称为指令集系统结构。

下面介绍计算机组成与实现的定义:

- 计算机组成是指计算机系统结构的逻辑实现,包括机器级内数据流的组成以及逻辑设计等。它着眼于机器级内各事件的排序方式与控制机构、各部件的功能以及各部件间的联系。
- 计算机实现是指计算机组成的物理实现,包括处理机,主存等部件的物理结构,器件的集成度和速度、器件、模块、插件、底板的划分与连接,专用器件的设计,微组装技术,信号传输,电源,冷却及整机装配技术等。它着眼于器件技术和微组装技术,其中,器件技术在实现技术中起着主导作用。

一般,计算机组成设计要确定以下几个方面:

- ① 数据通路宽度:数据总线上一次并行传送的信息位数。
- ② 专用部件的设置:设置哪些专用部件,如乘法专用部件、浮点运算部件、字符处理部件、地址运算部件,以及每种专用部件的个数等。这些与计算机要求达到的速度、专用部件的使用频率以及成本等因素相关。
- ③ 各种操作对部件的共享程度:如果共享程度过高,即使这些操作在逻辑上互不相关,也只能分时使用,这样就限制了速度,可以设置多个部件来降低共享程度,用提高操作并行度来提高速度,但成本也将提高。
- ④ 功能部件的并行度:功能部件的控制和处理方式是采用顺序串行,还是采用重叠流水或分布处理。
- ⑤ 控制机构的组成方式:事件、操作的排序机构是采用硬联控制还是微程序控制,是采用单机处理还是多机处理或功能分布处理。
- ⑥ 缓冲和排队技术:在不同部件之间怎样设置及设置多大容量的缓冲器来弥补它们的速度差异;采用什么次序来安排等待处理事件的先后次序,可以是随机、先进先出、后进先出、

优先级、循环队列等多种方式。

⑦ 预估、预判技术：采用什么原则来预测未来的行为，从而达到优化性能和优化处理的目的。

⑧ 可靠性技术：采用什么样的冗余技术和容错技术来提高可靠性。

下面的例子进一步说明了计算机系统结构、计算机组成和计算机实现的概念。

例如：指令系统的确定属于计算机系统结构，指令的实现，如取指令、取操作数、运算、送结果等的具体操作及其排序方式属计算机组成，而实现这些指令的具体电路，器件的设计及装配技术等则属计算机实现。

又如：确定是否要有乘法指令属计算机系统结构，乘法指令是用专门的乘法器实现，还是经加法器用重复的相加和右移操作来实现属计算机组成，而乘法器、加法器的物理实现，如器件的选定（包括器件集成度、类型、数量等的确定）及采用的微组装技术则属计算机的实现。

总之，计算机系统结构设计任务是进行软硬件功能的分配，确定传统机器级的软硬件界面。作为“计算机系统结构”这门学科，它实际包括了系统结构和组成两方面的内容，因此它研究的是软硬件的功能如何分配以及如何最佳、最合理地实现分配给硬件的功能。

## 1.2 计算机系统结构的设计方法

一般来说，在设计中提高硬件功能的比例可以提高解题速度，减少所需的存储容量，但会提高硬件成本，降低硬件的利用率和计算机系统的灵活性和适应性；而提高软件功能的比例，可以降低硬件的造价，提高系统的灵活性、适应性，但解题的速度会相对下降，软件设计费用和所需的存储器容量要增加。因此，确定软硬分配比例的一个主要依据，是在已有硬件和器件（主要是逻辑和存储器件）条件下，如何使系统具有较高的性价比。

### 1.2.1 计算机系统的设计原则

#### 1. 局部性原理

程序的局部性原理(principle of locality)是程序最重要的特征。它是指程序总是趋向于使用最近使用过的数据和指令，也就是说程序执行时所访问的存储器地址分布不是随机的，而是相对地簇集；这种簇集包括指令和数据两部分。程序局部性包括程序的时间局部性和程序的空间局部性：

① 程序的时间局部性：是指程序即将用到的信息可能就是目前正在使用的信息。若一条指令被执行，则在不久后可能再被执行。

② 程序的空间局部性：是指程序即将用到的信息可能与目前正在使用的信息在空间上

相邻或者临近。一旦一个存储单元被访问,那它附近的单元也将很快被访问。

程序的局部性原理是计算机体系结构设计的基础之一。利用程序的局部性原理,根据程序最近的访问情况来比较准确地预测将要访问的指令和数据。程序的局部性原理是虚拟存储技术引入的前提。虚拟存储的实现原理是当进程要求运行时,不是将它全部装入内存,而是将其一部分装入内存,另一部分暂不装入内存。

## 2. 关注经常性事件原则

要关注经常性事件,并以经常性事件为重点。在计算机系统设计中,经常需要在多种不同的设计方法之间折中,那么对经常性事件进行优化可以得到更多总体上的改进优化,且效果非常明显,所以在计算机设计中,最重要且应用最广泛的准则就是提高经常性事件的执行速度。在设计上必须有所取舍时,一定要优先考虑经常性事件。

经常出现的事件一般比不经常出现的事件简单,所以提高经常性事件的性能会相对容易些。在计算机设计中应用此原则时,首先要确定哪些是经常性事件,然后分析提高这种情况下的运行速度对计算机的整体性能提高的程度有多少,下面用两个例子来介绍该原则。

例如:处理器中的取指和译码单元要比乘法单元使用得更加频繁,那么这两个单元就是我们前面所说的“经常性事件”,所以在计算机设计中要先优化取指和译码单元。如 Pentium M 处理器为了降低系统功耗且同时提高计算机性能,在其译码单元引入 micro-op Fusion 概念,把原有的两个 micro-op(microinstructions)合成为一个进行操作,提升了传输速度。

再例如:当处理器执行两个数的加法运算时,一般情况下两数相加操作是不会发生溢出的,也就是说不溢出的情况是常见的,溢出情况是不常见的,所以不溢出相加的操作是“经常性事件”,那么在计算机设计中就要对不溢出相加的操作进行优化以达到提高机器性能的目的。当然,对不溢出的相加操作优化后,在处理溢出的相加操作时机器速度就会相对降低,但溢出的情况非常少,总体上机器的性能还是明显提高了。

## 3. Amdahl 定律

Amdahl 定律的内容是:通过使用某种较快的执行方式所获得的性能提高,受限于该部件占用系统执行时间的百分比。通过改进计算机的某一部分,所得到的性能提升程度可以通过 Amdahl 定律定量地反映出来。该定律将“关注经常性事件原则”进行了量化。

系统性能提升的程度可以用“加速比”这个概念来定量形象地对其进行描述。加速比反映了机器改进前后速度的提升程度,即机器改进前后系统性能提高的程度。如果对某个部件进行了优化改进,那么系统加速比表达式如下:

$$\text{加速比} = \frac{\text{优化后的系统性能}}{\text{优化前的系统性能}} = \frac{\text{优化前的总执行时间}}{\text{优化后的总执行时间}}$$

从加速比的表达式中我们可以看出加速比主要取决于两个因素：

① 优化前的系统中,可优化部分的执行时间在总的执行时间中占的比例。例如:一个程序需要运行 60 s,其中 20 s 的执行部分可以被优化,那么可优化部分所占的比例就是 20/60。我们将这个值称为“可优化比例”,可优化比例总是小于 1。可优化比例越大,越接近于 1,说明可优化的部分越多,相对得到的加速比就会越大,总体系统性能提高也就越明显。

② 优化后的系统和优化前的比较,系统性能提高的倍数。我们将可优化部分系统性能提高的倍数称为“优化加速比”。如果一个系统优化后,可优化的部分优化后的执行时间是 2 s,而优化前其执行时间是 5 s,那么优化加速比就是 5/2。显然,优化加速比是大于 1 的,优化加速比越大,系统性能提高程度越大。

优化前的系统包括可优化部分和不可优化部分,相对地,优化后的系统包括被优化部分和未被优化部分。优化前系统的执行时间等于可优化部分的执行时间和不可优化部分的执行时间之和。优化后系统的执行时间等于未优化部分的执行时间和优化部分的执行时间之和。

优化后总执行时间 =  $t_{\text{未优化部分}} + t_{\text{优化部分}}$

$$\begin{aligned} &= (1 - \text{可优化比例}) \times \text{优化前总执行时间} + \frac{\text{可优化比例} \times \text{优化前总执行时间}}{\text{优化加速比}} \\ &= \text{优化前总执行时间} \times \left[ (1 - \text{可优化比例}) + \frac{\text{可优化比例}}{\text{优化加速比}} \right] \end{aligned}$$

此时系统加速比的表达式如下：

$$\text{系统加速比} = \frac{\text{优化前总执行时间}}{\text{优化后总执行时间}} = \frac{1}{(1 - \text{可优化比例}) + \frac{\text{可优化比例}}{\text{优化加速比}}}$$

从以上的介绍中的我们可以分析出 Amdahl 定律反映的是收益增减的规律。当我们只优化系统的一部分计算性能时,优化的部分越多,即可优化比例越大,系统总体性能的提升就越大。也就是说,系统加速比随着可优化比例的增大而增大。若优化加速比趋近于无穷大的时候,系统加速比表达式中分母的(可优化比例/优化加速比)部分趋近于 0,那么系统加速比就趋近于  $1/(1 - \text{可优化比例})$ 。

有的时候,我们很难测量改进后可优化部分的运行时间,也就很难直接得到优化加速比。下面我们介绍一种基于公式的方法来计算和比较系统的性能。

#### 4. CPU 性能公式

基于处理器的性能来计算和比较系统的性能时,将 CPU 的执行时间分成时钟周期时间、CPI(每条指令的平均时钟周期数)、IC(指令条数)三个独立的分量。如果知道一种方案如何影响这三个分量,就能确定这种方案的总体性能效果,可以在设计硬件之前用仿真来测试这些分量。

我们知道,计算机存在一个基于恒定速率的时钟,通常用时钟周期或者时钟频率来描述一