高等学校计算机应用规划教材

计算机体系结构设计

- 计算机体系结构基本概念
- 数的表示与计算体系
- 指令系统设计
- 中央处理器体系结构设计
- 存储器体系结构设计
- I/O系统设计
- 并行处理与普适计算
- 生物计算机
- 光计算机
- 量子计算机

蔡政英 刘势 张上 肖明 编著

清华大学出版社

高等学校计算机应用规划教材

计算机体系结构设计

蔡政英　刘势　张上　肖明　编著

清华大学出版社

北　京

内 容 简 介

本书全面透彻地讲解经典计算机以及生物、光、量子等非经典计算机的体系结构设计方法，并融入大量新知识点，技术底蕴深厚。全书共分 10 章，清晰阐释重要概念，详述计算机体系结构的分析、设计和计算方法，注重培养读者的体系结构思维和创新能力，帮助读者建立起完整的知识体系。为方便读者自我检测，并扎实掌握所学的知识点，全书共列出 200 多道精选例题和习题。

本书提供的配套资源包含 PPT、例题等学习资源，便于读者学习、参考，读者可在 http://www.tupwk.com.cn/downpage 免费下载。任课教师可免费获取教学资源。

本书层次清晰、图文并茂、实例丰富、讲述详细，可作为高等院校计算机相关专业的本科生、研究生教材，可作为计算机资格考试培训机构用书，也可供计算机工程技术人员参考。

图书在版编目(CIP)数据

计算机体系结构设计 / 蔡政英　等编著. —北京：清华大学出版社，2018
(高等学校计算机应用规划教材)
ISBN 978-7-302-49078-4

Ⅰ. ①计…　Ⅱ. ①蔡…　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(2017)第 296174 号

责任编辑：刘金喜　韩宏志
封面设计：孔祥峰
版式设计：思创景点
责任校对：曹　阳
责任印制：沈　露

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编：**100084
社 总 机：010-62770175　　**邮　　购：**010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：三河市金元印装有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张：**24.25　　**字　　数：**636 千字
版　　次：2018 年 5 月第 1 版　　**印　　次：**2018 年 5 月第 1 次印刷
印　　数：1～2000
定　　价：59.00 元

产品编号：071993-01

前　言

本书可作为普通高等院校“计算机组成原理”“计算机组织与结构”“计算机系统结构”等相关课程的教材。面向高等院校的计算机、自动化以及电子工程等相关专业的本科生及研究生，并可作为参加研究生入学考试、全国计算机软件资格(水平)考试的备考书籍，也可供科研人员参考。

本书借鉴了国内外经典的相关教材和资料，汲取了它们各自的优点，并具有以下特点。

一是本书知识点全面，内容丰富，不同院校可以根据不同的教学目标适当选取教学内容，不同层次的读者也可以根据自己学习、考试、研究等的不同需要选择本书相关章节进行阅读。

二是本书避免过多地纠结于理论讲述和概念背诵，着重于对读者分析能力和设计能力的培养，全书各章节均穿插了大量图表、例题、习题，全书共计 200 多道例题及习题，均精选于研究生入学考试、全国计算机软件资格(水平)考试等中的典型分析题、设计题，主要参数均与市场主流产品的性能参数相当，尽量让读者学完了觉得“有用”。

三是本书包含大量的较新的知识点和题型，与新技术、实际应用的结合较为紧密，比如固态硬盘、脑波、移动计算、生物计算机、光计算机、量子计算机等，让读者在掌握经典计算机体系结构的基础上，对相关新技术与应用有所了解，引导和培养读者的体系结构思维和创新能力。

本书共分 10 章，全面系统地介绍计算机体系结构的相关知识。第 1 章简述计算机的发展和体系结构的演化；第 2 章主要介绍数的表示与计算体系，进而讨论运算器的设计；第 3 章主要讨论指令体系设计，从指令概念到指令体系设计与优化均做了详细介绍；第 4 章主要讨论中央处理器体系结构设计，包括组合逻辑控制器设计与微程序控制器设计，以及多核处理器和国产处理器的相关技术；第 5 章主要讨论存储器体系结构设计；第 6 章主要讨论输入/输出系统设计，以及常见外设与接口技术；第 7 章全面讨论在高速计算要求下的并行处理与普适计算技术；第 8 章、第 9 章、第 10 章分别介绍生物计算机与纳米机器人、光计算机、量子计算机的体系结构设计和发展演变。

本书内容通俗易懂，语言简练，深入浅出，图文并茂，共包含图表 420 多幅。本书按层次和模块化结构组织教学内容，授课教师可以根据需要对内容进行灵活的取舍。

本书主要由蔡政英、刘势、张上、肖明编写，蔡政英主要负责 1、5、10 章，刘势主要负责 2、6、8 章，张上主要负责 4、9 章，肖明主要负责 3、7 章。此外，张余、刘辉、卢晓燕、屈静、吴玥、胡绍齐等也参加了文字录入和绘图工作，在此表示感谢。

为说明问题，本书引用了大量原理、概念、定理、数据、公式、通用代码、试题等属于公有领域的非独创性内容，直接或间接引用了许多专家和学者已经发表的论文、文献或著作，在此向他们表示衷心的感谢，有兴趣的读者可以进一步查阅所附的主要参考文献。未能标出的参考文献，请作者通过出版社与本书编者联系。

本书配套的 PPT 讲稿可从 http://www.tupwk.com.cn/downpage 下载。

由于作者水平有限，书中难免有错误和不妥之处，敬请读者批评指正。

编　者

2018 年 1 月

目　录

第1章　绪　　论

内容提要：本章简要介绍计算机体系结构的基本概念、计算机的发展简史、计算机体系结构的分类、计算机系统的性能指标、计算机的应用。

本章重点：计算机体系结构的分类、计算机系统的性能指标。

1.1　计算机体系结构的基本概念

计算机体系结构(Computer Architecture)：又称为**计算机系统结构**，指机器语言程序员看到的传统机器级具有的属性结构，包括概念性结构和功能性结构两方面。为确保其所设计或生成的程序在机器上能正确运行，机器语言程序设计者或编译程序生成系统必须遵循这些计算机属性。对通用寄存器型机器来说，主要属性包括：数据表示、指令集、寻址规则、寄存器定义、存储系统、输入/输出结构、终端系统、信息保护等。

计算机组织(Computer Organization)：也称为**计算机组成**，指计算机体系结构的逻辑实现或逻辑结构，即物理机器级内各部件的功能及各部件的联系，各事件的控制方式与排序方式，包括物理机器级内数据流和控制流的组成及逻辑设计等。

计算机实现(Computer Implementation)：指计算机组成的物理实现或物理结构，即器件技术(占主导作用)和微组装技术，包括处理机、主存、外设等器件的物理结构与器件集成，信号传输技术，模块、插件、底板的划分与连接，电源、冷却及整机装配技术等。

上述三个术语具有不同的概念和不同的内容，但又互相紧密联系。具有相同计算机体系结构(如相同指令系统)的计算机根据速度要求不同等可采用不同的组成方式(逻辑结构)。与此类似，一种计算机组成也可采用不同的计算机实现(物理结构)。

计算机系统(Computer System)包括硬件和软件两大部分，两者缺一不可。硬件是计算机系统的物质基础，缺少硬件，再好的软件也无法运行；软件是计算机系统的灵魂，缺少软件，再好的硬件也毫无用处。硬件和软件密切配合，计算机才能正常工作和发挥作用。

计算机硬件(Computer Hardware)又称**硬件系统**，指计算机系统的实体部分，即计算机系统中电子、机械、光电等元件组成的各种物理器件的总称。计算机硬件的功能是输入并存储程序及数据，并将数据加工成可被利用的形式。各类物理硬件按系统结构的要求组成一个有机整体，为计算机软件提供运行的物质基础。中央处理器(Central Processing Unit，CPU)是计算机硬件的运算核心和控制核心，包括运算器(Arithmetic Unit，AU)、控制器(Control Unit，CU)和寄存器(Register)。运算器的核心是算术逻辑运算单元(Arithmetic Logic Unit，ALU)。

计算机软件(Computer Software)又称**软件系统**，指计算机系统中的软体部件，即计算机程序及其文档。程序是为了完成计算任务而组织起来的处理对象和处理规则的描述；文档是为了说明任务或程序而使用的文本式资料。程序只有装入机器内部才能工作；文档不一定装入机器内部，

一般用于给人看。计算机软件一般分为系统软件和应用软件两大类。计算机系统如图 1.1 所示。

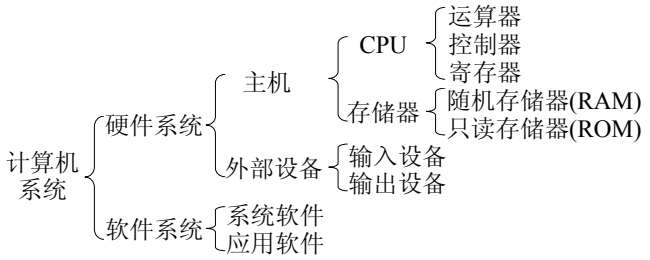

图 1.1　计算机系统

理论上，硬件和软件具有逻辑功能等效性，即计算机系统的某一具体功能既可以由硬件实现，也可以由软件实现。但两种实现方式具有不同的成本和速度。在设计一个计算机系统时，应当根据设计要求、当前的技术水平和现实条件，合理确定哪些功能由硬件实现或由软件实现，即硬件和软件的功能分配。而硬件与软件的交界面则称为计算机体系结构。

1.2　计算机的发展简史

1.2.1　机械式计算机的发展

经典计算机的发展经历了机械式计算机、机电式计算机、电子计算机几个阶段，之后又出现了生物计算机、光学计算机、量子计算机等超经典的计算机。

算盘是人类最早的计算工具，包括起源于中国的(穿)珠算盘、日本的十露盘、俄罗斯算盘、古希腊算板、演算沙盘等。在阿拉伯数字广泛使用之后，通过机械技术实现数字运算的机械式计算机开始出现。17 世纪时，一批欧洲数学家开始设计和制造数字计算机以实现数字形式基本运算。1642 年，法国数学家帕斯卡(Blaise Pascal)采用齿轮传动装置制作了最早的十进制加法器，也是世界公认的第一台机械式计算机，如图 1.2 所示。这台加法机利用类似钟表的齿轮传动原理，通过手工操作来完成加、减运算。

图 1.2　帕斯卡的加法机

1674 年，德国数学家和哲学家莱布尼茨(Gottfried Wilhelm Leibniz)设计了可手动进行完整四则运算的通用计算机，并提出了一个重要思想“用机械替代人完成繁杂重复的计算工作”。

1822 年，英国数学家巴贝奇(Charles Babbage)设计了一台能够代替人来编制数表的差分机。1834 年，他在差分机的基础上做了较大改进，设计了既能进行数字运算、又能进行逻辑运算的分析机。其设计思想已具备了现代计算机的概念，但当时的技术水平是无法实现的。

巴贝奇的思想提出之后的一百多年间，电子学、电磁学、电工学蓬勃发展。在元器件方面，相继发明了真空二极管和真空三极管；在系统技术方面，接连发明了无线电报、电视、雷达等，为现代计算机的发展准备了技术基础和物质条件。

同时，数学、物理也发展迅速。至 20 世纪 30 年代，物理学的所有领域都进入了定量化阶段，描述物理过程的各种数学方程层出不穷，很多方程已经很难用经典的分析方法解决。数值分析(Numerical Analysis)应运而生，奠定了现代计算机的数值算法基础，通过各种数值积分、数值微分、微分方程数值解法，从而把计算过程归结为基本的数值运算。

另外，现代计算机诞生的根本动力还是社会上对先进计算工具的迫切渴望。20 世纪以后，几乎各个领域都遇到了巨量计算困难，严重阻碍了科学的顺利发展。20 世纪上半叶两次世界大战的爆发，使得军事领域对高速计算工具的需求更为迫切，促使德、美、英等国几乎同时开始了机电式计算机和电子计算机的研究。

机电式计算机使用机械电子一体化技术实现数字运算，主要器件是机电式继电器，有时也称为继电器计算机。1938 年，德国科学家朱斯(Konrad Zuse)研制成功第一台机电式二进制可编程计算机 Z-1。1941 年，又研制成功机电式全自动计算机 Z-3，具备二进制运算、浮点记数、数字存储地址的指令等现代计算机的特征。1940~1947 年间，美国也相继制成了机电式计算机 MARK-I、MARK-Ⅱ、Model-1、Model-5 等。由于继电器的开关速度比较慢(大约为百分之一秒)，大大限制了计算机的运算速度。

1937 年，美国哈佛大学教授艾肯(Howard Aiken)在撰写博士论文时因需要求解非线性常微分方程，研究了巴贝奇的工作之后，提出了第一份自动计算机建议书(Proposed Automatic Calculating Machine)。经艾肯和 IBM 公司的合作与努力，在 1944 年制成了哈佛 MARK-I，IBM 将其命名为 ASCC(Automatic Sequence Controlled Calculator，自动时序控制计算机)，如图 1.3 所示。

MARK-I 是世界上最早的通用型自动机电式计算机之一，是计算机技术史上的一个重大突破。其核心是 72 个循环寄存器，每个可存放一个正或负的 23bit 数字，使用了 3000 多个继电器，加法速度是 300ms，乘法速度是 6s，除法速度是 11.4s。它长 15 米，高 2.4 米，有 15 万个元件，800 千米导线，重量达 5 吨。由穿孔卡片机实现数据和指令的输入，并由电传打字机实现数据输出。1947 年，艾肯又研制出机电式计算机 MARK-Ⅱ。1949 年，由于电子管技术的重大进步，艾肯研制的 MARK-Ⅲ是采用电子管的计算机。

图 1.3　哈佛 MARK-I

1.2.2　电子计算机硬件结构的发展

电子计算机的发展过程，经历了从器件制作到整机、从专用机到通用机、从外加式程序到存储程序的演变。一般将电子计算机的发展划分为五个时代。

1. 电子管时代(1946 ~ 1959 年)

在第一代电子管时代，电子计算机以电子管为基本逻辑单元，由汞延迟线、磁鼓等构成主存储器，用定点表示数据。图 1.4 和图 1.5 分别显示电子管和磁鼓存储器。

1938 年，美国爱荷华州立大学物理学家阿塔纳索夫(John Vincent Atanasoff)首先制成了电子计算机 ABC(Atanasoff-Berry Computer)，也是第一台能做加法和减法运算、有再生存储功能并以电子管为元件的数字计算机。

图 1.4　电子管

图 1.5　磁鼓存储器

二次大战期间，英国数学家、逻辑学家图灵(Alan Mathison Turing，见图 1.6)研制出一台译码计算机“图灵甜点”(Turing Bombe)。1944 年 2 月，巨人(Colossus Computer)计算机(如图 1.7 所示)正式启用，也是世界上最早的电子数字计算机，依靠该计算机英国在二战期间破解了大量德军通信密码。图灵也被誉为计算机科学之父、人工智能之父。

图 1.6　图灵

图 1.7　“巨人”计算机

1946 年 2 月 14 日，美国宾夕法尼亚大学莫尔学院研制的大型电子数值积分计算机 ENIAC (Electronic Numerical Integrator And Calculator)通过验收，这就是公认的世界上第一台电子计算机，如图 1.8 所示。该计算机完全采用电子线路实现算术、逻辑运算和信息存储，包含 18 000 多只电子管、1500 多个继电器、10 000 多只电容器、70 000 只电阻，运算速度比继电器计算机快 1000 倍。它占地 160 多平方米，重达 30 吨，功率 150 千瓦，有 5 种功能(每秒 5000 次加法运算，每秒 385 次乘法运算，平方和立方计算，sin 和 cos 函数数值运算，及其他更复杂的计算)。ENIAC 最初专门用于火炮弹道计算，后经多次改进而能用于各种科学计算，包括弹道计算、天气预报、原子能、热能点火、风洞试验设计等。但是，ENIAC 采用十进制计算和外加式程序，存储容量也太小，未完全具备现代计算机的主要特征。

新的重大突破是由美国普林斯顿大学教授冯 • 诺伊曼(John von Neumann，见图 1.9)的研究小组完成的。冯 • 诺依曼是 20 世纪最重要的科学家之一，在现代计算机、博弈论、核生化武器等多个领域内均有杰出建树的科学全才之一，1931 年成为美国普林斯顿大学的第一批终身教授时还不满 30 岁，被后人誉为计算机之父和博弈论之父。1945 年 3 月，该小组研制了一个全新的存储程序式通用电子计算机方案，即电子离散变量自动计算机(Electronic Discrete Variable Automatic Computer，EDVAC)，首次使用二进制而不是十进制。

本阶段的代表性机器有：冯 • 诺依曼的 IAS(1946 年)、UNIVAC 公司的 UNIVAC-1(1951 年)、IBM 公司的 IBM701(1953 年)和 IBM704(1956 年)。中国的有 103 机、104 机、119 机等。

图 1.8　ENIAC

图 1.9　冯 • 诺依曼

2. 晶体管时代(1959 ~ 1964 年)

在第二代晶体管时代，电子计算机主要以晶体管作为基本逻辑单元，由磁芯构成主存储器，引入浮点运算硬件提高科学计算能力。图 1.10 和图 1.11 显示了晶体管和磁芯存储器。

图 1.10　晶体管

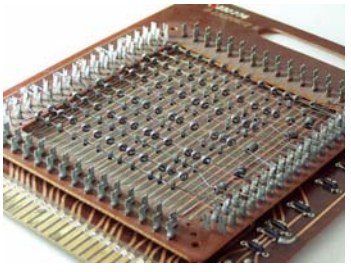

图 1.11　磁芯存储器

相比电子管计算机，晶体管计算机体积小、功耗低、速度快和可靠性高。代表性的机器有：IBM 公司的 IBM7090(1959 年)、IBM7094(1962 年)。中国的第一台晶体管计算机是 1965 年的 DJS-5 机，后继机有 DJS-121 机、DJS-108 机等。

3. 中、小规模集成电路时代(1964 ~ 1975 年)

在第三个发展时代，半导体技术的高速发展催生了集成电路和以集成电路器件为主要逻辑单元的电子计算机，进入了中、小规模集成电路(MSI、SSI)时代。主存储器也由半导体存储器构成(如图 1.12 所示)，出现了小型机，以及采用多处理器并行结构的大型机、巨型机。

代表性的计算机有：1964 年 IBM 公司的 IBM360 系列(图 1.13)、CDC 公司的 CDC6600(1964 年)、美国数字设备公司(Digital Equipment Corporation，DEC)的 PDP-8(1964 年)。中国的有 150 机(1973 年)、DJS-130 机(1974 年，100 系列机)、220 机(1973 ~ 1981 年，200 系列机)和 182 机(1976 年，180 系列机)。

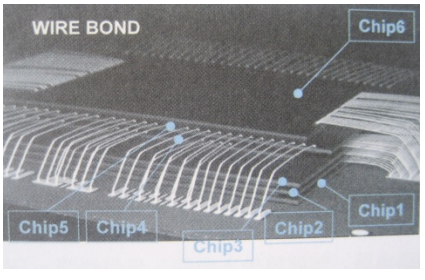

图 1.12　半导体存储器显微照片

图 1.13　IBM S/360-20

4. 大规模集成电路时代(1975 ~ 1990 年)

在第四个时代，集成电路的集成度不断提高，进入了大规模集成电路(LSI)时代。半导体存储器已完全取代了磁芯存储器，RISC 指令集出现。在该阶段，并行系统、多机系统、分布式系统、巨型向量机、阵列机等体系结构得到了发展，如美国的 Gray-1、中国的 HY-1 等，也出现了低档的微处理器。

1973 年，Intel 8080 微处理器的研制成功标志着 8 位微机的诞生，如 Apple Ⅱ、Z80 等。1978 年，采用 Intel 8086 微处理器构成的 16 位微机 IBM-PC/XT(如图 1.14 所示)面世，标志着台式个人计算机(Personal Computer，PC)开始走进办公室和家庭。诞生于 1977 年的苹果Ⅱ(如图 1.15 所示)是世界最早为个人用户量产的 PC 设备，同样也是最畅销的一款产品。

图 1.14　IBM PC

图 1.15　苹果Ⅱ

低端微机发展的另一方面是以单片机(Microcontrollers)为核心的工业控制、智能仪器仪表。计算机网络也由实验室走向了商业市场，推动并形成了信息技术(Information Technology，IT)产业。

5. 超大规模集成电路时代(1990 年至今)

第五个时代，出现了纳米科技和超大规模集成电路(VLSI)，大规模并行计算和高性能机群开始涌现，如 IBM 公司的“深蓝”RS/6000 SP2 是一台有 256 块处理芯片的超级并行计算机。中国的有 HY-III(1997 年，128 个 CPU 大规模并行处理)、HY-IV(机群技术)巨型机，而在 1999 年，“神威-I”超级并行处理计算机的成功研制使中国成为继美、日之后第三个具备高性能计算机研制能力的国家。

同一时期的微处理器技术也在高速发展，32 位、64 位的处理器芯片相继出现，如 Pentium IV、Itanium II 等。中国也推出了“龙芯”系列微处理器芯片。除了用作微机的主要处理部件外，微处理器芯片也可用作大规模巨型机的处理阵列。表 1.1 列出了电子计算机发展的五个时代。

表 1.1　电子计算机发展的五个时代

时代	年份	硬件	软件	应用
一	1946 ~ 1959	电子管	机器语言 汇编语言	科学计算
二	1959 ~ 1964	晶体管	高级语言	数据处理
三	1964 ~ 1975	小规模集成电路(Small-Scale Integration，SSI) 中规模集成电路(Medium-Scale Integration，MSI)	操作系统	工业控制
四	1975 ~ 1990	大规模集成电路(Large-Scale Integration，LSI)	数据库 网络	商业领域
五	1990 年至今	超大规模集成电路(Very-Large-Scale Integration，VLSI) 特大规模集成电路(Ultra-Large-Scale Integration，ULSI)	人工智能	各个领域

1.2.3　微处理器的发展

随着半导体和集成电路技术的进步，各大芯片厂商开始用微米级制程、甚至纳米制程生产处理器芯片，体积大大缩小。1971 年 11 月 15 日，全球第一款微处理器 Intel 4004 在 Intel 公司诞生。

微处理器(Microprocessor，μP 或 MPU)：即微型 CPU，是由一片或多片大规模集成电路组成的可编程中央处理部件，包括运算器、控制器和寄存器。

微型计算机(Microcomputer)：即微机，以微处理器为核心，配置了内存储器、输入输出接口电路及辅助电路的具有独立功能的计算机系统。将一台计算机的主要功能部件集成到一块芯片上也称为单片机(Microcontroller)，将一台计算机的主要功能部件都集成在一块电路板上也称为单板机(mono-plate processor)。

微型计算机系统(Microcomputer System)：以微型计算机为核心，配置了相应的外围设备(如键盘、鼠标、显示器、硬盘等)和电源等硬件系统，并安装必要的软件系统。

按单次处理数据位宽的不同，微处理器的发展大致可分为 6 个阶段。

(1) 第一代微处理器(1971 ~ 1973 年)

4 位或 8 位的微处理器阶段，典型的有 Intel 4004(4 位)和 Intel 8008(8 位)微处理器。Intel 4004 微处理器可进行 4 位二进制的并行运算，有 45 条指令，速度 0.05MIPS(Million Instruction Per Second，每秒百万条指令)。Intel 8008 是世界上第一款 8 位的微处理器，存储器采用 PMOS(Positive-channel Metal Oxide Semiconductor，P 沟道金属氧化物半导体)工艺。

该阶段微处理器工作速度较慢，指令系统不完整，存储器容量很小(仅几百字节)，只有汇编语言，没有操作系统，主要用于工业仪表、过程控制。

(2) 第二代微处理器(1974 ~1977 年)

8 位微处理器阶段，比第一代集成度提高了 1~ 4 倍，运算速度提高了 10 ~15 倍，指令系统更加完善，具备中断、直接存储器存取等功能，形成了典型的计算机体系结构。

由于微处理器性能强大却价格便宜，各大半导体公司都开始生产微处理器芯片。Intel 公司生产了 8080 和增强型 8085，Zilog 公司生产了 8080 的增强型 Z80，Motorola 公司生产了 M6800。但这些芯片基本没有改变 8080 的基本特点，均采用 NMOS(Negative-channel Metal Oxide Semiconductor，N 沟道金属氧化物半导体)工艺，集成度约 9000 只晶体管，平均指令执行时间为 1μs ~ 2μs，采用汇编语言、BASIC、Fortran 编程，使用单用户操作系统。

(3) 第三代微处理器(1978 ~ 1984 年)

16 位微处理器阶段。1978 年，Intel 公司率先推出 16 位微处理器 8086(见图 1.16)以及准 16 位微处理器 8088。8086 和 8088 均采用 16 位数据传输，8086 每周期能接收或传送 16 位数据，而 8088 每周期只有 8 位。1981 年，美国 IBM 公司将 8088 芯片用于其研制的 IBM PC 机中。其他公司的同类产品，有 Zilog 公司的 Z8000 和 Motorola 公司的 M68000 等。

(4) 第四代微处理器(1985 ~ 1992 年)

32 位微处理器阶段。1985 年 10 月 17 日，Intel 正式发布了 80386DX，内含 27.5 万个晶体管，时钟频率为 12.5MHz，之后提高到 33MHz 至 40MHz。得益于 32 位微处理器的强大运算能力，PC 的应用领域迅速扩大到商业办公、工程设计、数据中心、个人娱乐等。1989 年，Intel 推出了 80486 芯片。

(5) 第五代微处理器(1993～2005 年)

准 64 位阶段，典型产品是 Intel 公司的奔腾(Pentium)系列芯片及 AMD 的 K6 系列微处理器芯片。内部采用了超标量指令流水线结构和多媒体扩展(Multi Media eXtension，MMX)，并具有相互独立的指令和数据高速缓存。奔腾使用扩展 64bit 内存技术(Extended Memory 64 Technology，EM64T)，这并不是说奔腾可以直接执行 64 位应用程序，其寄存器仍然是 32 位。

1997 年，AMD 推出了 32 位的处理器 K6，0.35 微米制程，拥有全新的 MMX 指令。2000 年，Intel 推出的 Pentium 4 处理器(见图 1.17)采用 0.18 微米制程，内建 4200 万个晶体管，初期版本主频 1.5GHz，次年 8 月达到 2GHz。2002 年又推出内含超线程技术(Hyper-Threading，HT)的 Pentium 4 处理器。

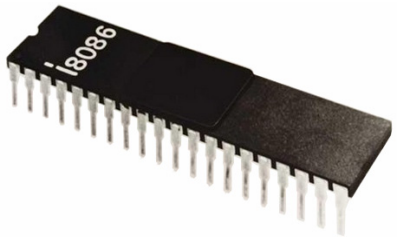

图 1.16　Intel 8086

图 1.17　Intel Pentium 4

(6) 第六代微处理器(2005 年至今)

64 位多核处理器阶段。2006 年 7 月 27 日，Intel 发布的酷睿 2(Core 2 Duo)是一个跨平台的构架体系，即基于 Core 微架构的产品体系的统称，包括服务器版(开发代号 Woodcrest)、桌面版(开发代号 Conroe)、移动版(开发代号 Merom)三大类，标志着进入 64 位双核技术时代。2008 年，Intel 推出 64 位四核处理器 Core i7(酷睿 i7，内核代号 Bloomfield)。在 2011 年初，Intel 发布沙桥(Sandy Bridge，SNB)处理器微架构，与处理器无缝融合的核芯显卡终结了集成显卡时代。AMD 也推出了 Athlon 系列 64 位多核微处理器，使用了超传输(Hyper Transport，HT)总线技术。

表 1.2 总结了微处理器发展的六个时代。

表 1.2　微处理器的六代

时代	年份	位数	典型处理器	典型制程
一	1971～1973	4 位	Intel 4004(4 位)、Intel 8008(8 位)等	10μm
二	1974～1977	8 位	Intel 8080/8085、Z80、M6800 等	6μm
三	1978～1984	16 位	Intel 8086、8088、Z8000、M68000 等	3μm
四	1985～1992	32 位	Intel 80386、80486 等	1.2μm
五	1993～2004	准 64 位	Intel Pentium 系列、AMD K6 等	0.18μm
六	2005 年至今	64 位多核	Intel 酷睿系列、AMD 速龙等	32nm

1.2.4　从模拟计算机到数字计算机

电子计算机还可以分为模拟式电子计算机和数字式电子计算机。

模拟式电子计算机(Analog Computer)，即模拟计算机，问世较早，是用电流、电压等连续变化的物理量直接进行运算的计算机。**模拟量**是在时间上或数值上都连续的物理量，**模拟信号**是表示连续的模拟量的信号，**模拟电路**是工作在模拟信号下的电子电路。

全电子化模拟计算机的研制开始于 20 世纪 30 年代。美国 Bell 实验室在二战期间研制出 M-9

火炮指挥仪。1947 年，以 M-9 火炮指挥仪中的运算放大器为基础，研制出全电子直流模拟计算机和高增益直流运算放大器。1948 年，第一台商品化模拟计算机研制成功。20 世纪 50 年代中后期，中国也研制出模拟计算机产品，如 M-2、M-6 等大型混合模拟计算机。

模拟计算机结构包括若干个加法器、乘法器、积分器、函数产生器等部件。根据待研究问题的数学模型，将一个(或几个)部件的输出端与另一个(或几个)部件的输入端互连起来，使整个计算机的输出量与输入量之间满足所研究问题的数学关系，在输出端得到问题的解。因而，模拟计算机部件之间的互连关系因问题而异。

模拟计算机可用于求解各种常微分方程和偏微分方程，也就是能模拟和仿真用这些方程描述的所有动力学物理系统。模拟计算机应用范围包括三个方面：作为计算工具、作为实物的数学模型和仿真设备、作为教学和训练工具。使用模拟计算机，其目的不是获得数学问题的精确解，更多的是提供实验研究的电子模型。

模拟电子计算机求解问题的精度不高，靠模拟电路来实现所有处理过程，电路结构复杂，可靠性极差。随着数字电路和处理技术的发展，逐渐被数字计算机所取代。

数字式电子计算机(Digital Electronic Computer)，即**数字计算机**，是现代电子计算机的主流，内部处理过程使用非连续的 0/1 数字信号或符号信号。**数字量**是时间上和数量上都离散的物理量，**数字信号**是表示离散的数字量的信号，**数字电路**是工作在数字信号下的电子电路。数字计算机的主要特征是离散性，即相邻两个符号之间不存在第三种符号，因此其结构和可靠性明显优于模拟计算机。

数字式电子计算机包括模数转换器和数模转换器。模数转换器(Analog-to-Digital Converter，ADC)是将模拟信号转换成数字信号的系统。模拟信号经带限滤波、采样保持电路转换成阶梯形状信号，再通过编码器将阶梯状信号中的电平变为二进制码，即数字信号。数模转换器(Digital -to-Analog Converter，DAC)是将数字信号转换为模拟信号的系统，一般用低通滤波器实现。数字信号先进行解码，把数字码转换成阶梯状的电平信号，然后通过低通滤波转换成模拟信号。

模拟计算机与数字计算机的比较如表 1.3 所示。

表 1.3　模拟计算机与数字计算机的比较

项目	模拟计算机	数字计算机
信号量	主要是模拟信号	主要是数字信号
电路	主要是模拟电路	主要是数字电路
精度	较低	较高
电路结构	较简单	较复杂
抗干扰性	较差，可靠性差	较强，可靠性好

1.2.5　计算机软件的发展

软件系统是计算机系统的重要组成部分，其发展与计算机硬件的发展密切相关，能在计算机硬件系统的基础上，更好地发挥计算机系统的性能。

1. 机器语言阶段

即早期的手编程序阶段。**机器语言**(machine language)又称**机器指令代码**(machine code)或**原**

生码(native code)，是计算机完全可以识别并执行的、由 0/1 串组成的低级语言。用这种机器语言来编写的程序称为目标程序。但是，机器语言非常难懂又容易出错，需要面向具体的机器，编写程序是非常困难，往往消耗大量的时间和人力，而且出错后也很难查找错误。

例如，在某种计算机上计算 2+6 的机器语言指令如下：

10110000 00000110

00000100 00000010

10100010 01010000

第一条指令功能是将“6”送到寄存器 AL 中；第二条指令功能是将“2”与寄存器 AL 中的内容“6”相加，结果仍在 AL 中；第三条指令功能是将 AL 中的内容送到地址为 5 的单元中。

这一阶段基本没有软件，更没有系统软件，编程使用机器语言，计算机只有专业人员才能操作。没有程序控制流的概念，每当插入一条新指令时，只能由编程人员手工移动数据及程序，操作相当困难。

2. 汇编语言阶段(20 世纪 50 年代)

即中级的半自动编程阶段。**汇编语言**(assembly language)又称**符号语言**(symbolic machine code)，用规定格式的数字、符号和文字来实现各种不同的指令，然后用这些特殊的符号指令(指令助记符)来编写程序，以提高编程效率。汇编语言程序代表了机器语言的第一层抽象，也是最早的软件设计抽象形式，从而实现了半自动化程序设计。

例如，用 ADD、SUB、MOV 分别表示加、减、移动数据的汇编语言指令，则计算 2+6 如下：

MOV　AL，6

ADD　AL，2

MOV　#5，AL

由于只有机器语言才可以在计算机上执行，因此需要将汇编语言源程序翻译成可执行的机器语言。其工作过程如图 1.18 所示。

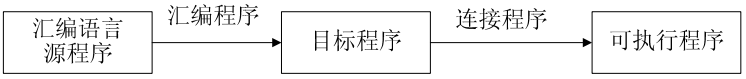

图 1.18　汇编语言工作过程

汇编程序(assembler，或**汇编器**)将汇编语言编制的程序(称为**源程序**，如.asm)翻译成机器语言程序(称为**目标程序**，如.obj)的工具，并经过**连接程序**(linker，**链接程序**或**链接器**)进一步连接(链接)成为**可执行程序**(如.exe)。符号语言仍然是低级语言，需要面对具体的机器硬件。一般把汇编程序和连接程序合称为**汇编系统**(assembly system)，

3. 高级语言阶段

即高级编程阶段。**高级语言**(high-level programming language)又称**算法语言**(algorithmic language)，包括一套规定的基本符号及其构成程序的规则。相对低级语言(机器语言和汇编语言)，高级语言的指令形式接近于自然语言和数学语言，易于学习和编程，也提高了程序的可读性，减少了出错的可能。例如，高级语言指令允许直接使用 2+6 进行计算。

1954 年，IBM 公司发明了第一个用于科学与工程计算的高级语言 FORTRAN(Formula Translation，公式翻译器)。20 世纪 50 年代出现了首批清晰定义的高级语言 ALGOL(ALGOrithmic Language，算法语言)。1958 年，美国麻省理工学院的麦卡锡(John Macarthy)发明了第一个用于人工智能的语言 LISP(LISt Processing，表处理)。1959 年，美国宾夕法尼亚大学的霍普(Grace Hopper)发明了第一个用于商业应用程序设计的 COBOL(Common Business Oriented Language，面向商业通用语言)。1962 年，美国哈佛大学的艾弗森(Kenneth E. Iverson)设计了 APL(A Programming Language，编程语言；或 Array Processing Language，阵列处理语言)。1964 年，美国达特茅斯学院的凯梅尼(John Kemeny)和卡茨(Thomas Kurtz)发明了 BASIC(Beginners' All-purpose Symbolic Instruction Code，初学者通用符号指令代码)。1967 年，英国剑桥大学的理察德(Martin Richards)设计了 BCPL(Basic Combined Programming Language，基本组合编程语言)。苏黎世工学院的沃斯(Nicklaus Wirth)于 1968 年创建了 Pascal 语言(以数学家 Pascal 命名)，并提出了“算法+数据结构=程序”(Algorithm+Data Structures=Programs)。1970 年，美国 Bell 实验室的汤普森(Ken Thompson)设计出 B 语言(取 BCPL 的首字母)，并用其写了第一个 UNIX 操作系统。1972 年，美国 Bell 实验室的里奇(Dennis M. Ritchie)设计出了 C 语言(取 BCPL 的第二个字母)。

在这一阶段出现了编译器、算法和数据结构，应用了控制流概念、数据类型、子程序、函数、模块等概念，并建立了子程序库和批处理的管理程序用于软件调度及管理。

通常采用以下两种语言处理程序把算法语言编写的源程序翻译为机器语言：

- **编译方式**

编译方式需要给计算机配置一套能把源程序翻译成目标程序的**编译程序**，一般由机器语言编写，然后由机器执行目标程序得出运算结果。编译方式包括词法分析、语法分析、语义分析、中间代码生成、代码优化和目标代码生成等过程，以及符号表管理和出错处理模块。但目标程序通常不能独立运行，还需要配置**连接程序**(linker，**链接程序**或**链接器**)来帮助。一般把编译程序和连接程序合称为**编译系统**(compiling system)，如图 1.19 所示。

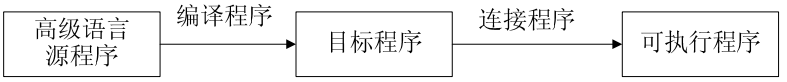

图 1.19　编译方式

编译型语言写的源程序需要专门的编译过程才能执行，将其翻译成目标程序(比如.obj)和机器语言的文件(比如.exe)，以后再运行时就不需重新翻译了，直接运行编译结果即可。由于只需要翻译一次，运行时不再需要翻译，因此编译型语言的程序执行效率高、执行速度快，因此常用于开发大型应用程序、操作系统、数据库系统等。C/C++、Pascal 等都是编译语言。

- **解释方式**

解释方式通过**解释系统**(interpretive system)对源程序的语句进行逐个解释并立即执行，不产生目标代码。**解释程序**(interpreter，或**解释器**)用于对源程序进行逐句分析，和源程序一起参加运行。若解释过程中没有错误，将该语句翻译成一条或多条机器语言指令并立即执行；若解释过程中发现错误，则会立即停止并向用户报错。解释方式在词法、语法和语义分析方面与编译方式基本相同，但在运行用户程序时，它直接执行源程序或源程序内部形式。工作过程如图 1.20 所示。

按照解释过程中产生中间代码的情况，解释程序又可分为三种方式(图 1.21)。

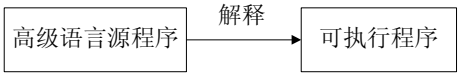

图 1.20　解释方式

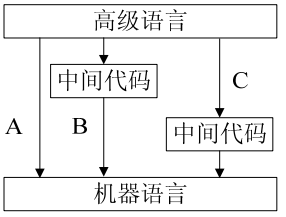

图 1.21　解释程序实现高级语言的三种方式

早期高级语言多采用这种方式，如 BASIC、dBASE、APL；需要兼容不同系统平台的网页脚本、服务器脚本及辅助开发接口也经常使用这种方式，如 Java、JavaScript、VBScript、Perl、Python、Ruby、美国 Mathworks 公司发布的 MATLAB(MATrix&LABoratory，即矩阵工厂/矩阵实验室)等。

编译和解释是高级语言处理的两种基本方式，两者的根本区别是：在编译方式下，源程序和编译程序都不参与目标程序的执行过程，编译程序将源程序翻译成独立的目标程序，机器上运行的是与源程序等价的目标程序；而在解释方式下，源程序(或其等价表示)和解释程序都要参与到程序的执行过程，翻译源程序时不产生独立的目标程序，运行程序的控制权在解释程序。

4. 操作系统阶段

操作系统(Operating System，OS)是直接运行在裸机上的最基本系统软件，用于管理和控制计算机硬件与软件资源。操作系统是计算机系统和用户的接口，也是硬件和其他软件的接口，其他任何软件都必须在操作系统的支持下才能运行。

1979 年，美国 Microsoft 公司为 IBM 个人计算机开发的磁盘操作系统 MS-DOS(Disk Operating System)是一个单用户单任务的操作系统。

1985 年，Microsoft 公司开发的 Microsoft Windows 采用了图形用户界面(Graphical User Interface，GUI)，交互方式更为人性化。

1969 年，美国 Bell 实验室的 B 语言发明者肯 • 汤普森(Ken Thompson)、C 语言之父丹尼斯 • 里奇(Dennis M. Ritchie)和计算机病毒发明者道格拉斯 • 麦克罗伊(Douglas Mcllroy)开发了 UNIX 操作系统，这是一个强大的多用户、多任务、分时操作系统，支持多种处理器架构。目前它的商标权由国际标准化组织(International Standards Organization，ISO)所拥有，只有符合单一 UNIX 规范的系统才能使用 UNIX 这个名称，否则只能称为类 UNIX(UNIX-like)。

1991 年 10 月正式公布的 Linux 操作系统是一套免费使用和自由传播的类 UNIX 的嵌入式操作系统，支持多用户、多任务、多线程和多 CPU，可运行于多种硬件平台上，包括 80x86、680x0、SPARC、Alpha 等。

Android(机器人)操作系统最初由美国计算机科学家安迪 • 鲁宾(Andy Rubin)开发，主要支持手机，2005 年 8 月由 Google 收购。Android 的系统架构是分层架构，从高到低分为四个层，分别是应用程序层、应用程序框架层、系统运行库层和 Linux 内核层。

2007 年 1 月 9 日，Apple 公司公布了移动操作系统 iOS，由 Apple 公司的 Mac OS X 核心演变而来，两者都属于类 UNIX 的商业操作系统，支持多任务处理。

该阶段还出现了数据库管理系统(Database Management System，DBMS)，比较流行模型有三种，即树结构的层次模型、图结构的网状模型和表结构的关系模型。1961 年，通用电气公司的查理斯 • 巴

克曼(Charles Bachman)成功地开发出世界上第一个 DBMS，采用网状结构模型的集成数据存储(Integrated Data Store，IDS)。dBASE 是第一个在微型计算机上被广泛使用的关系型 DBMS。数据库(Data base，DB)和 DBMS 一起，组成了数据库系统(Data base System，DBS)。常见的有 SYBASE、DB2、ORACLE、MySQL、ACCESS、Visual Foxpro、MS SQL Server、Informix、PostgreSQL。

5. Web 服务阶段

在这个阶段，形成了以 Web 应用服务为核心的多层开发体系架构，包括 J2EE(Java 2 Platform Enterprise Edition)编程技术规范和 Web Service 协议架构，出现了软件工程思想。

该阶段还出现了面向对象程序设计(Object-Oriented Programming，OOP)和面向对象语言(Object-Oriented Language，OOL)。1967 年 5 月 20 日，挪威科学家奥利-约翰·达尔(Ole-Johan Dahl)和克利斯登·奈加特(Kristen Nygaard)发布了最早的面向对象的程序设计语言 Simula 67。20 世纪 70 年代初 Xerox PARC 的艾伦·凯(Alan Kay)、丹·英戈尔斯(Dan Ingalls)、特德·凯勒(Ted Kaehler)和阿黛勒·戈德堡(Adele Goldberg)等开发了 Smalltalk，被公认为历史上第二个面向对象的程序设计语言和第一个真正的集成开发环境(Integrated Development Environment，IDE)。1983 年，美国 Bell 实验室的 Bjame Sgoustrup 改良了 C 语言并命名为 C++。1985 年，ISE 公司专家贝特朗·梅耶(Bertrand Meyer)等人开发了 Eiffel 语言。1989 年，荷兰人吉多·范罗苏姆(Guido van Rossum)发明了 Python，并于 1991 年公开发行。1996 年 1 月，Sun 公司发布了 Java 的第一个开发工具包(JDK 1.0)。1998 年，美国国家标准学会(American National Standards Institute，ANSI)和 ISO 组织共同制订了 C++的 ANSI/ISO 标准，包括 Microsoft 的 Visual C++和 Borland 公司的 C++ Builder 都支持这个标准。2000 年 6 月，Microsoft 的安德斯·海尔斯伯格(Anders Hejlsberg)发布了第一个面向组件的编程语言 C#(C Sharp)。

该阶段还形成了以面向对象为基础的概念和模型，包括 CORBA(Common Object Request Broker Architecture，公共对象请求代理体系结构)、DLL(Dynamic Link Library，动态链接库)、JavaBean(咖啡豆)、ODBC(Open Database Connectivity，开放数据库连接)、OLE(Object Linking and Embedding，对象链接与嵌入)等。

1.3　计算机体系结构的分类

1.3.1　冯·诺依曼体系结构

冯·诺依曼体系结构也称**普林斯顿结构**，将指令存储器和数据存储器合并在一起，并使用同一个总线传输。1946 年 6 月，冯·诺依曼在研究 EDVAC 计算机时提出了存储程序的概念。现代计算机很多都是以冯·诺依曼的体系结构为基础的，如 Intel 8086、ARM7、MIPS 处理器等。

冯·诺依曼机：由一个中央处理器和一个存储器组成，将指令和数据都存储在同一个存储器中，指令和数据的宽度相同。存储器存有指令和数据，可根据所给的地址进行读或写。

冯·诺依曼思想的基本要点可归纳如下：

(1) 计算机由运算器、控制器、存储器、输入设备和输出设备五大部件组成。

图 1.22 所示为冯·诺依曼结构计算机的基本硬件组成。通常把运算器和控制器统称为**中央处理单元** CPU，把 CPU 与主存储器(内存)统称为**计算机主机**，而把输入设备、输出设备、外存储

器称为计算机的**外部设备**或 **I/O 设备**。

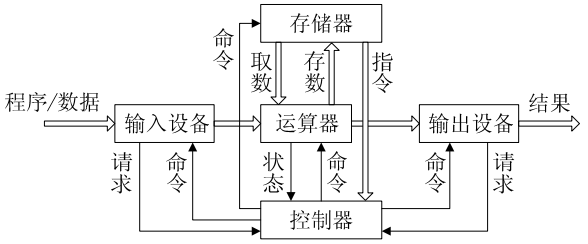

图 1.22　典型的冯 • 诺依曼计算机体系结构图

(2) 采用二进制形式表示数据和指令

程序是由完成一定功能的若干指令组成的有序集合。指令是程序的基本单位，包括操作码和地址码两部分，操作码说明操作的功能，地址码指出数据所在存储单元。冯 • 诺依曼机中，指令与数据均以二进制代码的形式共同存于存储器中按地址访问，两者地位相等。

(3) 采用存储程序方式

这是冯 • 诺依曼思想的核心，是计算机自动、高速运行的基础。**存储程序**是指预先编制好解题程序，并在运行前与所需的数据一起存入存储器中。在程序运行(解题)时，按照存储器中预先编制好的程序，控制器从存储器中自动、连续地依次取出指令并执行，直到求解出所要求的结果。

通常，完成一条指令需要 3 个步骤，即：取指令、指令译码(分析指令)和执行指令。冯 • 诺依曼结构的处理器对存储器进行读写操作的指令如图 1.23 所示：

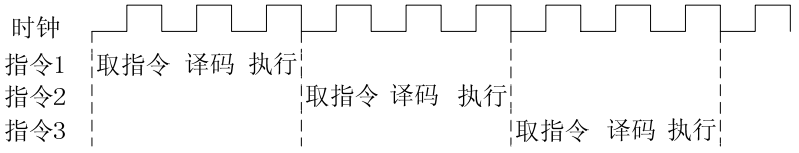

图 1.23　冯 • 诺依曼结构的处理器对存储器进行读写操作

在冯 • 诺依曼结构中，指令和数据从同一个存储空间存取，并使用同一总线传输，所以只能串行执行，而无法重叠执行。如图 1.23 所示，一条指令需要 3 个指令周期，3 条指令至少需要 9 个指令周期。因为 CPU 与内存之间的信息流量(资料传输率)比内存容量小太多，将 CPU 与内存分开会导致所谓的“**冯 • 诺伊曼瓶颈**”(von Neumann bottleneck)，如 CPU 需要对大量数据执行简单操作，信息流量将成为限制计算机性能的瓶颈。

1.3.2　哈佛体系结构

在数字信号处理(Digital Signal Processing，DSP)中，最大的工作量是与存储器交换数据，包括输入信号的采样数据、滤波器参数和程序指令。为此，哈佛大学提出了与普林斯顿体系结构完全不同的哈佛结构。

哈佛结构是将程序和数据存储在不同的存储空间中的并行体系结构，即程序存储器(Program Memory，PM)和数据存储器(Data Memory，DM)分开，两者独立编址、独立访问。相应使用 4 条系统总线，程序和数据各有一条地址总线与一条数据总线。程序和数据的存储器及总线全部分离的结构允许在一个机器周期内同时获得指令字和操作数，执行和取址能完全重叠，大大提高了执行速度

和数据吞吐率。典型处理器有 AVR、ARM9、ARM10、ARM11，以及众多的数字信号处理器(Digital Signal Processor，DSP)等。美国德州仪器公司在 1982 年成功推出了第一代 DSP 芯片 TMS32010。之后，Motorola 公司在 1986 年推出了定点 DSP 芯片 MC56001，1990 年推出了兼容 IEEE 浮点格式的浮点 DSP 芯片 MC96002。另外，ADI 公司也推出了 ADSP 系列 DSP。

哈佛机：为程序和数据提供了各自独立的存储器和总线，指令和数据可以有不同的数据宽度，为数字信号处理提供了较高的效率。程序计数器只指向程序存储器而不指向数据存储器，但在哈佛机上很难编写出一个自修改的程序。

哈佛结构的计算机由中央处理器、程序存储器和数据存储器组成，其结构如图 1.24 所示。

中央处理器首先到程序存储器中读取指令内容，译码后得到数据地址，再到相应的数据存储器中读取数据，并进行下一步的操作，如图 1.25 所示。可见，在处理相同的 3 条存取数指令的时候，各条指令可以重叠地执行，虽然每条指令仍然需要 3 个指令周期，但 3 条指令总共只需要 5 个指令周期，克服了数据流传输过程中的瓶颈，提高了系统性能。

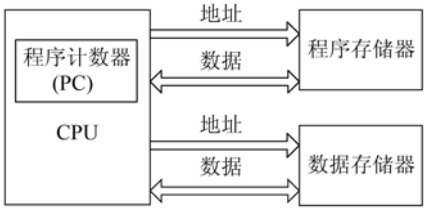

图 1.24　哈佛结构

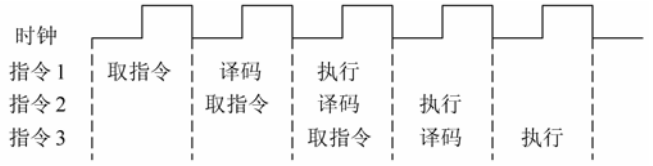

图 1.25　哈佛结构各条指令可以重叠地执行

普林斯顿结构处理器与哈佛结构处理器的对比如表 1.4 所示。

表 1.4　普林斯顿结构与哈佛结构对比

项目	普林斯顿结构	哈佛结构
编址方式	指令与数据统一编址	指令与数据独立编址
空间管理	程序空间与数据空间统一	程序空间与数据空间完全分开
总线	程序与数据总线不独立，速度较低	程序与数据总线独立，速度较高
代码位数	指令与数据宽度必须相同	指令与数据宽度可以不同
应用领域	主要应用于通用计算机领域	主要应用于嵌入式计算机领域
交互性	较强	较弱

超级哈佛结构(Super Harvard Architecture，SHARC)是美国 ADI 公司推出的 32 位浮点数字信号处理器系列产品，最早起源于浮点单指令单数据流(SISD)的 ADSP-21020，在标准哈佛架构基础增加指令缓存形成三总线架构。SHARC 处理器是一个不带嵌入式存储器或外设的独立计算内核，PM 和 DM 是通过连接 SRAM 的外部总线进行访问的，从新增加的指令缓存器能够执行所选的指令，同时进行数据和系数存取，提升了基于紧密循环的计算过程的吞吐性能。

1.3.3　Flynn 计算机体系结构的分类

1966 年，美国斯坦福大学教授弗林(Michael J. Flynn)提出根据指令流、数据流的多倍性(multiplicity)对计算机体系结构进行分类。

- **指令流**：计算机执行的指令序列。

- **数据流**：计算机指令调用的数据(包括输入数据、输出数据和中间结果)序列。
- **多倍性**：系统性能瓶颈部件上，处于同一执行阶段的指令或数据的最大可能个数。

根据不同的指令流-数据流组织方式，Flynn 把计算机系统分为四种。

(1) 单指令流单数据流(Single Instruction stream Single Data stream，SISD)

SISD 是顺序执行的传统单处理机，指令部件每次只译码一条指令，只对一个操作部件分配数据。

(2) 单指令流多数据流(Single Instruction stream Multiple Data stream，SIMD)

SIMD 以并行处理机为代表，由一个指令部件控制，包括多个重复的处理单元 $PU_1 \sim PU_n$，根据同一指令流为不同处理单元分配各自的数据。

(3) 多指令流单数据流(Multiple Instruction stream Single Data stream，MISD)

MISD 具有 n 个处理单元，根据 n 条不同指令的要求对同一数据流(包括其中间结果)进行不同处理，一个处理单元的输出又可作为另一个处理单元的输入。

(4) 多指令流多数据流(Multiple Instruction stream Multiple Data stream，MIMD)

MIMD 以多处理机为代表，能在作业、任务、指令等级别全面实施并行。

1.3.4　冯泽云分类法

1972 年，美籍华人冯泽云教授(Tse-yun Feng)提出用最大并行度对计算机体系结构进行分类。所谓**最大并行度**(degree of parallelism)P_m是指在单位时间内计算机系统能够处理的最大二进制位数。设每一个时钟周期 t_i 内处理的二进制位数为 P_i，则 T 个时钟周期内平均并行度为 $P_a=(\sum P_i)/T$(其中 i=1,2,…,T)。平均并行度 P_a 与应用程序无关，完全取决于系统，则系统在周期 T 内的平均利用率为 $\mu=P_a/P_m=(\sum P_i)/(T\times P_m)$。

用平面直角坐标系中的一个点表示一个计算机系统，横坐标表示一个字中同时处理的二进制位数，即字宽(N 位)；纵坐标表示在一个位片中能同时处理的字数，即位片宽度(M 位)，则最大并行度 $P_m=N\times M$。由此得出四种不同的计算机结构：

(1) 字串行、位串行(Word-Serial and Bit-Serial，WSBS)，其中 N=1，M=1。

(2) 字并行、位串行(Word-Parallel and Bit-Serial，WPBS)，其中 N=1，M>1。

(3) 字串行、位并行(Word-Serial and Bit-Parallel，WSBP)，其中 N>1，M=1。

(4) 字并行、位并行(Word Parallel and Bit Parallel，WPBP)，其中 N>1，M>1。

1.3.5　计算机的语言层次结构

从计算机语言的角度出发，把计算机系统划分成 5 个层次级别，每一级能以一种不同的语言进行程序设计。计算机系统的语言层次结构如图 1.26 所示。

各层次之间关系紧密，下层是上层的基础，上层是下层功能的扩展。计算机系统中 5 个层次级别

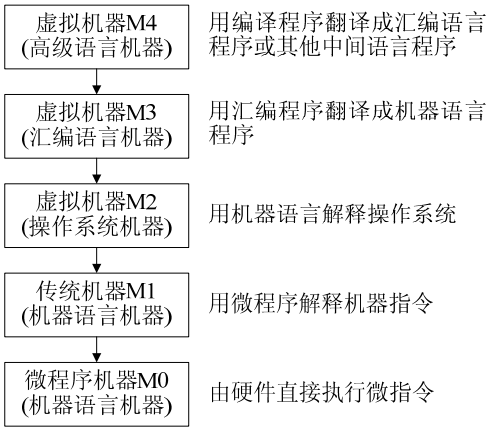

图 1.26　计算机系统的语言层次结构

的特点如表 1.5 所示。

表 1.5　计算机系统语言层次结构的特点

第 M4 级	高级语言级	由各种高级语言编译程序支持和执行	软件级	符号语言
第 M3 级	汇编语言级	由汇编程序支持和执行		
第 M2 级	操作系统级	由操作系统程序实现	混合级	二进制语言
第 M1 级	机器语言级	由微程序解释机器指令系统	硬件级	
第 M0 级	微程序设计级	由机器硬件直接执行微指令		

1.3.6　计算机的总线组织结构

总线是一组可被多个功能部件共享的信息传送公共线路。总线结构可灵活地组织、修改与扩充系统、连接一台计算机的不同部件(或多台不同的计算机)，大大减少传输线数，并简化硬件结构，如图 1.27 所示。

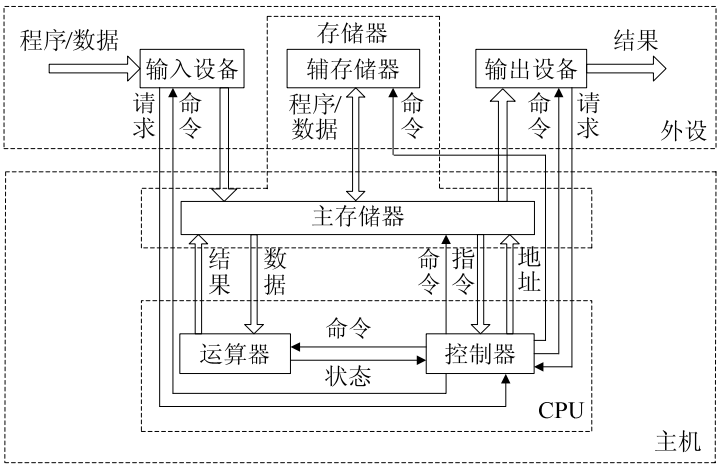

图 1.27　现代计算机总线组织结构

为防止总线上信息冲突，各个部件共享总线时必须分时使用总线，保证总线上传送的信息总是唯一的。但是总线上的信息可被各个部件同时接收。

按其任务，可把总线分为下面几种类型。

- **CPU 内部总线**：这一级属于数据总线，即(芯)片内总线。用于连接 CPU 内部算术逻辑单元和各寄存器。
- **部件内总线**：这一级属于芯片间的总线，即功能模块(板)内总线。计算机中经常将不同功能模块制作成插件，并用总线结构连接插件上的相关芯片。
- **系统总线**：这一级属于部件间或功能模块(板)间总线，是连接整机系统的基础，用于连接系统内各大部件(如 CPU、主存、I/O 设备等)。系统总线包括地址线(Address Bus，AB)、控制/状态信号线(Control Bus，CB)、数据线(Data Bus，DB)。
- **外总线**：这一级属于计算机间总线，用于计算机系统之间或与其他系统之间的连接。

根据不同的类型的总线组织架构，可将计算机的体系结构做如下分类。

(1) 单总线结构机器

单总线结构机器，通过一组系统总线(AB、CB、DB)把 CPU、主存及各种 I/O 接口连接起来，是微、小型机的典型结构。有时用接口泛指系统总线与外围设备间连接的逻辑部件。

在图 1.28 的单总线结构机器中，CPU 与存储器之间，CPU 与各 I/O 设备之间、I/O 设备与存储器之间、各 I/O 设备之间，都通过一条单总线交换信息。所以各 I/O 设备的寄存器与存储器可以统一编址，统称为总线地址。优点是 CPU 可以像访问存储器一样访问 I/O 设备的寄存器，控制简单，易于扩充。但单总线结构机器同一时刻只能在两个部件之间传送信息，系统速度受到限制。

(2) 双总线结构机器

双总线机器在 CPU 与存储器之间增加了一组存储总线，CPU 通过存储总线直接访存，如图 1.29 所示。这种结构保持了单总线结构控制简单、易于扩充的优点，也提高了 CPU 的访存速度和系统性能。

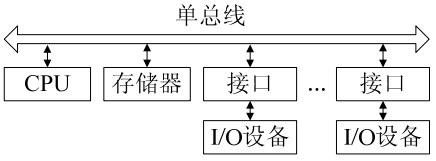

图 1.28　计算机的单总线结构

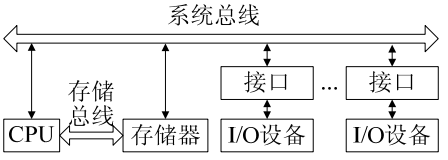

图 1.29　面向存储器的双总线结构

此外，还有三总线结构机器，以及以 CPU 为中心的双总线结构，如图 1.30 所示。图 1.30 中有两组总线，一组为存储总线，用于 CPU 与存储器之间的信息传送；另一组为 I/O 总线，用于 CPU 与 I/O 设备之间的信息交换。其优点是比较简单，但存储器与 I/O 设备间的信息交换都必须经过 CPU，降低了 CPU 的工作效率。

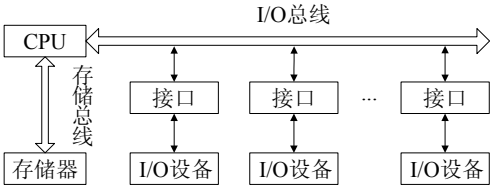

图 1.30　以 CPU 为中心的双总线结构

(3) 通道结构机器

通道是一种专门用来管理 I/O 操作的具有处理机功能的控制部件。通道结构机器通常采用主机、通道、I/O 控制器、外设四级连接方式，具有较强的扩充能力，是大、中型机的典型结构。

较小的系统可将 CPU 与通道合并一起，并将 I/O 控制器与外设合并一起。较大的系统可单独设置通道，如图 1.31 所示。对于更大的系统，通常使用专门的 I/O 处理机代替通道，甚至使用功能更强的前端机。

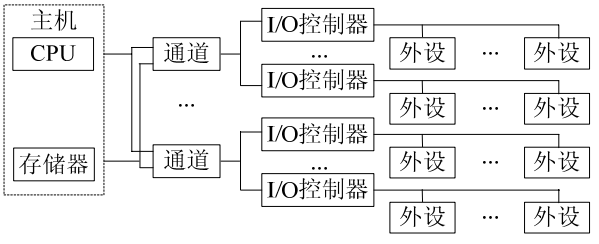

图 1.31　大、中型计算机系统的通道结构

【例 1.1】　图 1.32 为主机框图，根据要求回答存数指令的信息流程。

解：取指令：PC→MAR→M→MDR→IR

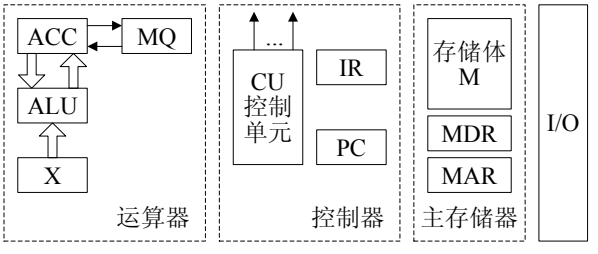

图 1.32　某主机框图

分析指令：OP(IR)→CU

执行指令：Ad(IR)→MAR→M

ACC→MDR→M

1.3.7　计算机的软件系统

在计算机系统中，基本的软件系统包括系统软件与应用软件两大类。

(1) 系统软件

系统软件是用于保证计算机系统正确、高效运行的基础软件，以系统资源的形式提供给用户使用。包括操作系统(如 DOS、Windows、UNIX、Android、iOS 等)、操作系统的补丁程序、硬件驱动程序、语言处理程序、DBMS、分布式软件、网络管理系统等。

(2) 应用软件

应用软件是为解决某个应用领域中的问题而编制的程序。目前应用软件正向标准化、集成化方向发展，很多通用的应用程序都可组成不同的应用软件包以共享使用。常见应用软件包括各类工具软件(科学计算类程序、工程设计类程序、数据统计与处理类程序、情报检索程序等)、应用管理软件(企业管理系统、购物支付系统、教学系统等)、游戏软件等。

【例 1.2】 环路复杂度可用于定量描述程序的拓扑结构复杂度或逻辑复杂度，常用 McCabe 方法度量。McCabe 方法最早由美国麻省理工学院的麦凯(Warren L. McCabe)和蒂勒(Ernest Thiele)于 1925 年提出。在程序控制流程图 G 中，程序代码的最小单元用节点表示，节点间的程序流用边表示。设流程图 G 有 n 个节点和 e 条边，其环路复杂度为：

$$V(G)=e-n+2 \tag{1.1}$$

环路复杂度值 $V(G)$越大，表示程序的控制路径越复杂。一般程序模块的环路复杂度为 10。试用 McCabe 度量法计算图 1.33 所示程序图的环路复杂度。

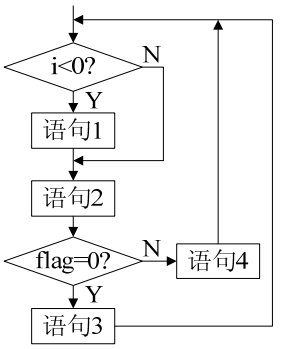

图 1.33　某程序图

解：根据图 1.33 可知，e=8，n=6，根据式(1.1)，有：$V(G)=e-n+2=8-6+2=4$。

1.4　计算机系统的性能指标

1.4.1　摩尔定律

1964 年，Intel 公司创始人戈登 • 摩尔(Gordon Moore)在一篇短论文里提到：集成电路的性能每 18 个月将提高一倍，且价格降低一半。这就是半导体发展史上意义深远的**摩尔定律**。摩尔定律还有其他的表述，如计算机系统性能每过 10 年将会增加 100 倍，通信带宽也提高 100 倍，且成本不会增加。

1.4.2　性能测试程序

计算机性能评价指标通常用峰值性能和持续性能。**峰值性能**(peak performance)是计算机系统

在理想情形下可获得的最高理论性能，但不一定是系统的实际性能。**持续性能**(sustained performance)又称**实际性能**，指计算机系统连续工作时的性能。由于实际运行时，程序会受到计算机系统硬件结构、操作系统、算法设计等因素的影响，持续性能往往只有峰值性能的 5%～35%。

核心测试程序(kernel benchmark)：由实际程序中抽取的关键程序框架代码，其运行对程序总的执行时间有直接影响。如 Livermore Loops 和 Linpack。

基准测试程序集(benchmark)：也称为测试程序组件(benchmark suites)，将一组有代表性的、类型不同的应用程序集中起来，以准确测试计算机系统处理各种程序的性能。如 SPEC、TPC。

SPEC(Standard Performance Evaluation Corporation，标准性能评估组织)是一个开放性的非营利组织，1988 年由 DEC、HP、MIPS、SUN 共同发起，是最成功的性能测试标准化组织。SPEC CPU2006 是 SPEC 组织推出的 CPU 子系统评估软件，有 CINT2006 和 CFP2006 两个子项目，共 12 个整数基准测试程序集和 17 个浮点数基准测试程序集。CINT2006 包括 C 编译程序、量子计算机仿真、象棋程序等；CFP2006 包括有限元模型结构化网格法、分子动力学质点法、流体动力学稀疏线性代数法等。

由于服务器(Server)有不同的服务功能，因此有多类基准(benchmark)：

- SPECrate——多处理器的处理速率

面向 Server 的处理器吞吐量的基准，通过运行多个 SPEC CPU 基准测试程序副本来获取多处理机的处理速率(SPECrate)。

- SPECSFS——文件服务器基准

测试网络文件系统的性能，包括处理器、磁盘、I/O 的性能。

- SPECWeb——Web 服务器基准

模拟多个用户请求 Web 服务器的静态和动态页面以测试性能。

1.4.3　基本性能指标

1. 处理机字长(机器字长)

处理机字长(机器字长)指处理机一次能够完成二进制运算的位数，标志着运算精度，如 32 位、64 位；机器字长与系统数据总线宽度有一定的相关性，但并非完全相同。

当 i 位十进制数与 j 位二进制数进行比较时，有下列等式：

$$10^i = 2^j$$

两边取对数，可得：

$$\frac{j}{i} = \frac{\ln 10}{\ln 2} = 3.3$$

因此，若要保证 i 位十进制数的精度，至少要 3.3 倍 i 位二进制数的位数才能满足要求。

2. 主频/周期

CPU 的主时钟频率为主频(主振频率)，其倒数为 CPU 的时钟周期(T 周期)。

- **时钟周期**(clock cycle，也称为**振荡周期**)

时钟周期是计算机中最基本的、最小的时间单位，定义为时钟脉冲的倒数，通常是单片机外接晶振的倒数。例如外接 40MHz 的晶振，则时钟周期为 $1/2^{22}$μs。

时钟脉冲是计算机的基本工作脉冲，控制着系统的工作节奏，CPU 在一个时钟周期内仅完

成一个最基本的动作。不同的计算机所需的时钟周期也不相同；对于同一机型计算机，时钟频率越高则其工作速度就越快。

- **机器周期**(machine cycle，也称为 **CPU 周期**)

机器周期是指令周期中的某一工作阶段所需的时间，一般包括若干个**时钟周期(振荡周期)**。通常把一条指令的执行过程划分为若干工作阶段，每阶段完成一项基本操作或工作。例如，取指令、译码、执行指令等。机器周期就是完成一个基本操作所需的时间。

- **指令周期**(instruction cycle)

指令周期是执行完一条指令所需的时间，一般包括若干个**机器周期(CPU 周期)**。不同指令需要的机器周期数有所不同。如某些单字节指令，指令在取指令阶段被取出到指令寄存器后，立即进行译码执行，不需要其他机器周期，称为单周期指令。而比较复杂的指令(如转移指令、乘法指令)往往需要两个以上的机器周期，称为双周期指令或多周期指令。

- **总线周期**(bus cycle)

总线周期是进行一个访问存储器或 I/O 设备所需的操作时间，一般包括若干个时钟周期。一个指令周期往往由若干个总线周期组成。一般情况下，一个总线周期包含 4 个 T 周期(时钟周期)：CPU 从内存读取指令、对内存存取数据、对外设读写数据、执行总线周期。

3. CPU 的运算速度

CPU 执行时间：CPU 执行一段程序所占用的 CPU 时间。

CPI(Cycle Per Instruction)：执行一条指令所需的平均时钟周期数。

MIPS(Million Instructions Per Second)：每秒百万条指令数，即单位时间内执行的指令条数，用于评估标量机(通常执行一条标量指令只得到一个运算结果)的平均指令执行速度，不适于评估向量机。CPU 性能评估一般采用合成测试程序，常用的有用于测试整数计算能力的 Dhrystone(计算单位 DMIPS)和测试浮点计算能力的 Whetstone(计算单位 MFLOPS)两种。

$$\text{MIPS}=\frac{\text{指令条数}}{\text{执行时间}}\times 10^{-6} \tag{1.2}$$

MFLOPS(Million FLoating-point Operations Per Second)：每秒百万次浮点操作数，即单位时间内浮点操作的次数，适于评估向量机(通常执行一条向量指令可得到多个运算结果)的平均操作执行速度。

$$\text{MFLOPS}=\frac{\text{浮点运算次数}}{\text{执行时间}}\times 10^{-6} \tag{1.3}$$

4. 一个程序的 CPU 时间

$$\text{CPU 时间}=\text{一个程序的 CPU 时钟周期数}\times\text{时钟周期长度} \tag{1.4}$$

或者：

$$\text{CPU时间}=\frac{\text{一个程序的CPU时钟周期数}}{\text{时钟频率}}$$

除了计算程序所需的时钟周期数，经常还需要计算程序执行的指令数(Instruction Count，IC)，可得到执行一条指令所需的平均时钟周期数(Cycles Per Instruction，CPI)：

$$\text{CPI}=\frac{\text{一个程序的时钟周期数}}{\text{IC}} \tag{1.5}$$

CPI 可直观地评估不同的指令集或不同实现方式的性能。在式(1.5)中，一个程序的时钟周期数可定义为 IC×CPI，则在 CPU 时间公式(1.4)中可使用 CPI 表示为：

$$\text{CPU 时间}=\text{IC}\times\text{CPI}\times\text{时钟周期的长度} \tag{1.6}$$

将式(1.4)展开成度量单位后，可得出不同指标的组合方式：

$$\frac{\text{指令}}{\text{程序}}\times\frac{\text{时钟周期}}{\text{指令}}\times\frac{\text{秒}}{\text{时钟周期}}=\frac{\text{秒}}{\text{程序}}=\text{CPU时间} \tag{1.7}$$

可见，CPU 时间与程序的指令数、执行每条指令所需的时钟周期数、时钟周期的长度均有关。而且这三个因素对 CPU 时间的影响程度是等同的，其中任何一个改进 10%，则 CPU 时间就改进 10%。实际上，很难孤立地改变一个因素，因为各因素存在相互关联：

- 指令数 IC 取决于指令集的结构和编译器的设计；
- 平均时钟周期数 CPI 取决于计算机组成结构和指令集结构；
- 时钟周期的长度取决于硬件技术和软件技术。

5. 吞吐量

吞吐量(throughput)指计算机在某一时间间隔内所处理的信息量多少，可根据测试周期内完成的事务数量计算得出，即每秒执行的事务数量(Transaction Per Second，TPS)。

6. 响应时间

响应时间指系统从输入有效到产生响应之间所需的时间，常用时间单位表示。

7. 利用率

利用率指在给定的时间范围内，实际使用系统的时间所占比率，用百分比表示。

8. 总线宽度

即主存储器带宽，指运算器与存储器之间的数据总线宽度，常用单位时间内从主存储器读出的二进制信息量来衡量，单位为字节数/秒。

9. 主存储器容量

主存储器容量指主存储器所能容纳最大信息量或所有主存储器单元的总数目，常用二进制数据的字节数表示。主存容量大，则可容纳更多信息，或运行更大、更复杂的程序。

【例 1.3】 某机 CPU 主频为 1.862GHz，平均指令执行速度为 200MIPS，若每个机器周期平均包含 4 个时钟周期，试问：(1) 时钟周期是多少？平均指令周期是多少？(2)平均指令周期包含多少个机器周期？(3)若改用时钟周期为 0.2ns 的 CPU，则平均指令执行速度又是多少？(4)若要达到 4 亿条/s 的平均指令执行速度，CPU 主频应为多少？

解：(1) 时钟周期=1÷1.862GHz≈0.5ns。

平均指令周期=1÷200MIPS=5ns。

(2) 由于每个机器周期平均包含 4 个时钟周期，机器周期=0.5ns×4=2.0ns。

平均指令周期包含的机器周期数=5ns÷2.0ns=2.5。

(3) 改用时钟周期为 0.2ns 的 CPU 芯片后，主频为 1/0.2ns≈4.657GHz，平均指令执行速度为(200MIPS×4.657GHz)/1.862GHz≈501MIPS。

(4) 若要达到 4 亿条/s 的指令执行速度=400MIPS，平均指令周期 1/400MIPS=2.5ns，时钟周期为 2.5ns/(4×2.5)=0.25ns，所以，主频=1/0.25ns≈3.725GHz。

【例 1.4】 主频 2GHz 的某机需要运行 500 000 条指令组成的程序，指令分为四种类型，程序中各类指令混合比及 CPI 值如表 1.6 所示。

表 1.6　四类指令的指令混合比及每类指令的 CPI 值

指令类型	指令混合比	CPI
算术和逻辑	55%	1
高速缓存命中的加载/存储	21%	2
转移	15%	4
高速缓存未命中时的存储器访问	9%	8

(1) 试计算执行该程序的平均 CPI；

(2) 计算相应的平均指令执行速度及程序的 CPU 时间。

解：

(1) $\mathrm{CPI}=\sum_{i=1}^{n}\left(\mathrm{CPI}_i+\frac{\mathrm{I}_i}{\mathrm{IC}}\right)=1\times0.55+2\times0.21+4\times0.15+8\times0.09=2.29\mathrm{CPI}$

(2) $\mathrm{MIPS}=\frac{f}{\mathrm{CPI}\times10^6}=\frac{2.0\mathrm{G}}{2.29\times10^6}\approx937.766$

$$\text{CPU 时间}=\frac{\sum_{i=1}^{n}(\mathrm{CPI}_i\times\mathrm{IC})}{\text{时钟频率}}=500\,000\times\frac{1\times0.55+2\times0.21+4\times0.15+8\times0.09}{2.0\mathrm{G}}\approx533.18\mu\mathrm{s}$$

1.4.4　Amdahl 定律

1967 年，IBM360 机的主要设计者吉恩 • 迈伦 • 阿姆达尔(Gene Myron Amdahl)提出了 Amdahl 定律(Amdahl's law，或 Amdahl's argument)，是计算机系统设计的重要定量原理之一。该定律指出：系统中某一部件采用更快执行方式所获得的总体系统性能改进程度(加速比)，取决于该执行方式使用的频率或占总执行时间的比例(可改进比例)。因此，通过提高某部件性能来获得系统加速会受速度慢的部件限制。所以，Amdahl 定律表达了性能改进的递减规则：若仅改进某一部件性能，改进的越多，则提升系统总体性能就越有限。

Amdahl 定律定义了采取增强(加速)某部件的措施后，整个系统获得的性能改进程度或执行时间的**加速比**，记为 S_n。该加速比与两个因素有关。一个是在改进前的系统中，可改进部分的执行时间占总执行时间的比例，称为**可改进比例**，记为 F_e，F_e 总是小于 1 的。另一个是在改进后的系统中，可改进部件改进以后性能提高的倍数，称为**部件加速比**，记为 S_e，S_e 一般大于 1。假设改进前整个任务的执行时间为 T_0，改进后整个任务的执行时间为 T_n，则系统加速比 S_n 为：

$$S_n=\frac{T_0}{T_n}=\frac{1}{(1-F_e)+F_e/S_e} \tag{1.8}$$

其中，(1-F_e)表示不可改进部分。当 F_e=0 时，即没有可改进部分，S_n=1；当 F_e=1 时，即所有部分均可改进，S_n=S_e。当 S_e→∞时，则加速比不超过 S_n=1/(1−F_e)。假设改进前是单核处理器，改进后是 4 核处理器，且 F_e=1，则 S_n=S_e=4。

【例 1.5】 某机使用浮点运算部件后，浮点运算速度提高为 15 倍，运行某一程序时总体性能提高为 5 倍，试计算该程序中浮点运算的比例。

解： S_e=15，S_n=5，根据 Amdahl 定律：

$$S_n = \frac{T_0}{T_n} = \frac{1}{(1-F_e) + F_e / S_e} = 5$$

可知，F_e≈85.71%。

【例 1.6】 某系统中有三个部件可以改进，且部件 1、2、3 的加速比分别为 10、5、2；(1)若系统加速比为 2，部件 1 和部件 2 的可改进比例均为 20%，求部件 3 的可改进比例为多少？(2)若部件 1、2、3 的可改进比例分别为 40%、25%、15%，三个部件同时改进，求改进后不可加速部分的执行时间占总执行时间的比例是多少？

解：(1) 当多个部件可同时改进时，Amdahl 定律扩展为：

$$S_n = \frac{T_0}{T_n} = \frac{1}{\left(1 - \sum_i F_i\right) + \sum_i \frac{F_i}{S_i}}$$

已知 S_1=10，S_2=5，S_3=2，S_n=2，F_1=0.2，F_2=0.2，得：

$$2 = \frac{1}{[1-(0.2+0.2+F_3)] + \left(\frac{0.2}{10} + \frac{0.2}{5} + \frac{F_3}{2}\right)}$$

得 F_3=0.32，即部件 3 的可改进比例为 32%。

(2) 设改进前系统的执行时间为 T，F_1=0.4，F_2=0.25，F_3=0.15，则改进前 3 个部件的执行时间为：(0.4+0.25+0.15)T = 0.8T，不可改进部分的执行时间为 0.2T。

改进后，S_1=10，S_2=5，S_3=2，则 3 个部件执行时间为：$T_n' = \frac{0.4T}{10} + \frac{0.25T}{5} + \frac{0.15T}{2} = 0.165T$

改进后，整个系统的执行时间为：T_n=0.165T+0.2T=0.365T

则改进后，系统中不可改进部分的执行时间占总执行时间的比例是：$\frac{0.2T}{0.365T} \approx 0.5479$

1.5　计算机的应用

现代计算机的应用领域已经渗透到人类社会活动的方方面面，大致可划分为以下几类。

(1) 科学计算(scientific computing)

计算机起源于计算问题，科学计算是计算机应用最早且最重要的领域，主要利用计算机解决科学研究和工程技术中的各类数学问题。科学计算的特点是求解的问题复杂，计算量大，难以由人工完成，如空间探索、原子能、新材料新结构、基因编码等的计算和设计等。

(2) 数据处理(data processing)

数据处理(信息处理)是计算机应用最广泛的领域，主要是利用计算机处理各类非工程数据

的采集、计算、管理等工作。数据处理的特点是数据量大，但计算比较简单(如逻辑运算与判断)，常用表格和数据库形式。如管理信息系统、办公自动化系统、银行储蓄系统、证券交易系统等。

(3) 自动控制(automatic control)

自动控制是相对人工控制(manual control)而言，主要是利用计算机设备或装置控制机器、设备或生产过程自动按照预定的规律运行。自动控制的特点是不需要人直接参与。以微处理器为核心的嵌入式系统是当今最热门的应用之一，能够对设备、设施和系统运行进行控制、监视、辅助等，有助于将人类从复杂、危险、繁重的工作环境中解放出来，并大大提高工作效率。如数控机床、无人机、智能机器人、导弹、卫星、物联网、无线传感器网络等。

(4) 计算机辅助设计(Computer Aided Design，CAD)

计算机辅助设计主要是利用计算机帮助设计人员完成工程、产品、服务等设计工作的过程和技术。20 世纪 60 年代，美国麻省理工学院提出的交互式图形学研究计划是最早的 CAD。CAD 技术的特点是，能够提高设计的自动化水平和设计质量，减轻设计人员的劳动强度，缩短设计周期。如飞机、汽车、轮船等的设计。CAD 技术的发展，也带动了计算机辅助制造(Computer Aided Manufacturing，CAM)、计算机辅助工程(Computer Aided Engineering，CAE)、计算机辅助教学(Computer Aided Instruction，CAI)等的进步。

(5) 人工智能(Artificial Intelligence，AI；或 Machine Intelligence，MI)

人工智能是相对自然智能(Natural Intelligence，NI)来说的，指利用计算机来模拟、延伸和扩展自然智能的技术。1956 年夏罗切斯特(Nathaniel Rochester)和香农(Claude Elwood Shannon)等人发起了达特茅斯会议，会上，美国工程院院士艾伦 • 纽厄尔(Allen Newell)、1978 年诺贝尔经济学奖得主赫伯特 • 西蒙(Herbert Alexander Simon)、斯坦福大学教授约翰 • 麦卡锡(John McCarthy)、哈佛大学的马文 • 明斯基(Marvin Lee Minsky)、IBM 公司的亚瑟 • 塞缪尔(Arthur Lee Samuel)等为首的一批科学家首次提出了“人工智能”这一术语，成为公认的人工智能之父。人工智能技术主要应用于模式识别、机器学习、自然语言翻译、博弈、虚拟仿真等方面。1997 年 5 月 12 日，IBM 公司的“深蓝”超级并行计算机战胜了国际象棋特级大师俄罗斯名将卡斯帕罗夫(Garry Kasparov)，实现了人机对决的首场胜利。2016 年 3 月，Google 旗下 DeepMind 的阿尔法围棋(AlphaGo)战胜了围棋世界冠军李世石，其主要工作原理是“深度学习”。2017 年 11 月 16 日，波士顿动力(Boston Dynamics)发布的双足机器人 SpotMini 已经能够顺利完成旋转、跳跃、后空翻等动作。

(6) 网络应用

Internet 网络前身是 20 世纪 60 年代美国的 ARPANET，但直到 20 世纪 80 年代中期才进入中国，以 1987 年通过中国学术网 CANET 向世界发出第一封 E-mail 为标志。经多年发展，Internet 网络的应用不断拓展，包括电子邮件、移动支付、各类 Web 应用、远程教学、社交应用、多媒体音视频点播等。

习　题　1

1.1　试用 McCabe 度量法计算图 1.34 中的程序流程图的环路复杂度。

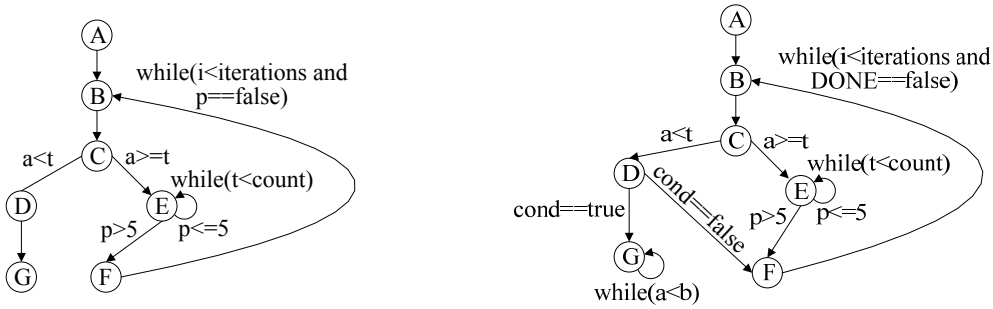

图 1.34　程序流程图

1.2　设某机主频为 2.0Hz，平均指令周期含 2.5 个机器周期，平均每个机器周期含 2 个时钟周期，试求：(1) 该机的 MIPS 为多少？(2) 若主频不变，平均指令周期含 5 个机器周期，但平均每个机器周期含 4 个时钟周期，则该机的 MIPS 又是多少？(3) 由此可得出什么结论？

1.3　某机的主频为 2.0GHz，不同指令的平均指令执行时间和使用频率如表 1.7 所示，试计算该机的 MIPS？若主频升为 4.0GHz，试计算该机的 MIPS？

表 1.7　各类指令的平均执行时间和使用频率

指令类别	存取	加、减、比较、转移	乘除	其他
平均指令执行时间	1.2ns	1.6ns	20ns	2.8ns
使用频率	40%	45%	5%	10%

1.4　在某机中，某功能部件的执行时间占整个系统执行时间的 60%。试求该功能部件的处理速度应提高多少倍，才将整个系统的性能提高 2.0 倍？

1.5　在某系统中，某一功能的处理时间占整个系统运行时间的 30%，若将该功能的处理速度加快 10 倍，问整个系统的性能提高多少？