21世纪普通高等教育规划教材

微型计算机原理及应用

Weixing Jisuanji Yuanli Ji Yingyong

许立梓　何小敏
陈　玮　高明琴　编

机械工业出版社
CHINA MACHINE PRESS

21 世纪普通高等教育规划教材

微型计算机原理及应用

许立梓　何小敏　陈玮　高明琴　编
毛宗源　杨宜民　主审

机 械 工 业 出 版 社

本书是21世纪的高等院校计算机基础教材。以8086/8088为主线，全面、系统、深入地介绍了16位微型计算机的基本知识、基本组成和体系结构、指令系统、汇编语言及程序设计方法、主存储器的组成及设计、输入输出信息的控制方法、中断系统、可编程接口芯片、A/D、D/A转换器及接口技术、总线技术、微型计算机系统结构，并对现代微机为了提高其性能而采用的流水线技术、高速缓存技术、PCI总线技术等作了简要介绍。书中附有大量的例题，各章配有适当的习题，适合60～80学时教学使用。

本书注重理论联系实际，突出实用技术，融入作者多年的经验和体会，可作为高等院校非计算机专业学生微机原理课程的教材，也可以作为从事微机应用与开发的工程技术人员自学教材或参考书。

图书在版编目(CIP)数据

微型计算机原理及应用/许立梓等编．—北京：机械工业出版社，2003.1

21世纪普通高等教育规划教材

ISBN 7-111-11333-0

Ⅰ.微… Ⅱ.许… Ⅲ.微型计算机—高等学校—教材 Ⅳ.TP36

中国版本图书馆CIP数据核字(2002)第100352号

机械工业出版社(北京市百万庄大街22号 邮政编码100037)

责任编辑：贡克勤 版式设计：霍永明 责任校对：张 媛

封面设计：张 静 责任印制：路 琳

北京机工印刷厂印刷·新华书店北京发行所发行

2003年1月第1版·第1次印刷

787mm×1092mm 1/16·19印张·468千字

0 001—5 000册

定价：26.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

本社购书热线电话(010)68993821、88379646

封面无防伪标均为盗版

前　言

“微型计算机原理及应用”课程是高等院校电类专业的一门专业基础主干课程，学习和掌握与之相关内容是高等院校学生的需求和迫切愿望。为了使学生更好地掌握本门课程的核心内容，满足21世纪培养高素质人才和教学改革的需要，根据作者多年从事高校本门课程的教学实践和体会，编写了本书。

本书以IBM PC系列微机中的基础机型IBM PC/XT为主讲机型。现代许多高档微机为了保持对它的兼容性，把它的许多设计思想，芯片连接方法等作为参考对象。正是基于这种出发点并考虑了本课程性质，本书结合8086/8088系统，按CPU—存储器—I/O接口—总线及系统结构的轴线来讲述微机的CPU及其指令系统、汇编语言程序设计、存储器、中断系统、接口技术、总线技术及系统结构等。在此基础上为了与现代微机知识相连接，书中也介绍了提高CPU性能的新技术，存储系统的新技术，为增强系统性能的总线新技术等等。目的是使学生在学习了本课程之后，具有对微机应用系统的硬、软件的开发能力。

编写本书时，叙述力求深入浅出，尽量多举实例，每章之后附有习题，引导学生思考并掌握该章内容。在学习“微型计算机原理及应用”课程时，注意加强实践环节，完成教学大纲规定的实验学时。在完成了理论教学和实验教学之后，建议安排课程设计，培养学生系统设计能力。

本书第1章第1节，第2、6章，第9章1～5节及附录由许立梓编写，第1章第2、3节及第3章由高明琴编写，第4章、第8章第6～9节及第9章第6节由陈玮编写，第5、7章及第8章第1～5节由何小敏编写。许立梓负责全书内容的统稿和定稿。

本书由华南理工大学博士生导师毛宗源教授及广东工业大学博士生导师杨宜民教授主审，华南师范大学黄元梅副教授协审。在审稿过程中，他们提出了许多宝贵意见，在此致以衷心的感谢。

由于编者水平有限，书中难免存在错误或不妥之处，敬请读者不吝指正。

编　者

目　录

第 1 章　微型计算机的基础知识

本章主要介绍微型计算机的发展、基本组成原理以及微机的运算基础，是深入学习微机原理的一个必要的入门。

1.1　微型计算机系统概述

1.1.1　微处理器和微型计算机的发展及应用

微处理器对于微型计算机来说，就像心脏对于人体一样重要，因此，个人计算机的发展史，实质上就是微处理器从低级向高级、从简单向复杂发展的过程。下面，以 Intel 公司微处理器的主流产品发展为例，看看微机技术发展历程。

Intel 公司早期的 4004 是 4 位微处理器，1972 年推出的 8008 是 8 位微处理器，它们组成的 MCS-4 和 MCS-8 的微型计算机只能做基本的算术运算，主要采用机器语言和简单的汇编语言，是低档的微处理器和微型计算机。

1973 年，Intel 公司推出 8 位微处理器 8080，这是第一个现代的 8 位微处理器，8080 可寻址范围是 8008 的 4 倍（64KB），指令的执行速度也比 8008 快 10 倍。后来 Intel 又开发出 8080 的改进型芯片 8085，它兼容 8080 且性能更好，硬件接口更简单。软件除采用汇编语言外，还配有 BASIC 等高级语言及其相应的解释程序。

1978 年，Intel 公司推出 16 位 8086 微处理器，并在一年后推出准 16 位微处理器 8088，8086/8088 可寻址 1MB 存储器，指令的执行速度大大超过 8085。1982 年，IBM 选择 8088 作为其首台个人计算机 IBM-PC/XT 的 CPU。从此，微型计算机进入一个高速发展时代。

1982 年，Intel 发布 80286，随即它在 IBM-PC/AT 中找到了用武之地。80286 提供仍然是 16 位数据总线，但地址总线为 24 位，可寻址 16MB 内存。与 8088 相比，指令的执行速度增加了 8 倍。80286 通过实模式和保护模式两种操作模式实现了与以前处理器的兼容。同时，80286 建立了 CISC（Complex Instruction Set Computer）体系结构，为以后的 X86 系列 CPU 指出了可能发展的功能。

计算机的广泛应用要求微处理器的速度更高，存储器容量更大。Intel 公司顺应这种要求，于 1985 年，推出了 32 位 CPU80386 系列，其代表性的产品 386DX-33 的主频为 33MHz，执行单一指令需要两个时钟周期。芯片内外部数据总线及地址总线都是 32 位。80386 系列的主要优点是：由于新的内存使用方式和多任务性，使开发基于 PC 的图形用户界面 GUI 成为可能，也为运行 Windows3.x 提供了所需的处理能力。

1989 年，Intel 又推出了 80486CPU，486CPU 在 Intel 处理器中第一个采用一级高速缓存（8KB），它减少了访问 RAM 的次数，提供的突发模式总线结构加快了 RAM 与 CPU 之间的数据传输；80486 也是第一个真正采用流水线设计的 Intel X86 处理器，它提高了指令的吞吐率；同时采用浮点单元 FPU（Floating-Point Unit）实现对包含十进制小数的实数进行运算，因此改善了应用程序的性能。486CPU 的主频有 33MHz、40MHz、50MHz，执行单一指令需

要一个时钟周期，芯片内外部数据总线及地址总线也都是 32 位，但综合性能是 386 的 3 倍以上。1992 年以后，Intel 又相继推出了 486 系列的新成员：采用倍频技术的 486DX2 和 486DX4，其倍频为 2 倍或 3 倍。它有效地缓解了外部设备速度跟不上 CPU 主频速度的矛盾。

微型计算机的进一步发展就是提高其运行速度，1993 年，具有 64 位数据总线和 32 位地址总线的 Pentium（奔腾）处理器问世，其运算速度超过 100MIPS，Pentium 处理器中部分采用了 RISC（Reduced Instruction Set Computer）技术；具有超标量体系结构；采用双流水线结构及分支预测技术；在从内存读写时采用流水线突发模式；同时，具有两个 16KB 的独立的一级高速缓存分别用于存储数据和指令，提高了程序执行速度。

面对剧烈的竞争，Intel 在 1995 年底又推出了 Pentium Pro（高能奔腾），它主要在几个方面对 Pentium 进行了改进。首先，Pentium Pro 采用 RISC 技术，14 级流水线核心结构实现了动态执行。其次，Pentium Pro 在 Pentium 处理器 16KB 一级高速缓存的基础上，增加了内置的 256KB 二级高速缓存，并采用将二级高速缓存封装在芯片内的方式，使二级高速缓存能以 CPU 的时钟频率工作。其三，Pentium Pro 的地址总线增加到 36 位，使其寻址空间增加到 64GB。

1997 年 1 月，Intel 公司又推出 Pentium MMX（多能奔腾）芯片，它在 X86 系列指令集的基础上加入了 57 条多媒体指令。这些指令专门用来处理视频、音频和图像数据，使 CPU 在多媒体操作上具有更强大的处理能力。Pentium MMX 还使用了许多新技术，单指令多数据流 SIMD（Single Instruction Multiple Data）技术能够用一条指令并行处理多个数据，缩短了 CPU 在处理视频、音频、图形和动画时用于运算的时间；流水线从 5 级增加到 6 级，一级高速缓存扩充为两个 16KB：一个用于数据高速缓存，另一个用于指令高速缓存，因而速度大大加快；Pentium MMX 还吸收了其他 CPU 的优秀处理技术，如包含了一个原本用于 Pentium Pro 的分支预测单元和 Cyrix 6X86 具有的返回堆栈技术。

同年 5 月，Intel 公司又推出了 Pentium Ⅱ芯片，它采用 Slot 1 架构，通过单边插接卡（SEC）与主板相连，SEC 卡盒将 CPU 内核和二级高速缓存（256KB）封装在里面，处理器采用了与 Pentium Pro 相同的动态执行技术，可以加速软件的执行；通过双重独立总线与系统总线相连，可进行多重数据交换，提高系统性能；Pentium Ⅱ也包含 MMX 指令集。第一代 Pentium Ⅱ工作在 66MHz；1998 年推出的第二代 Pentium Ⅱ处理器主频有工作在 66MHz 外频下的 333MHz 和工作在 100MHz 外频下的 350MHz、400MHz 和 450MHz。

其后，Intel 公司在 1999 年 2 月 28 日首推的 Pentium Ⅲ处理器，主频为 450MHz、800MHz，配备 133MHz 或 100MHz 系统总线，512KB 二级缓存以 CPU 主频全速工作，Pentium Ⅲ处理器结合了动态执行技术、双重独立总线技术和 MMX 多媒体增强技术，此外 Pentium Ⅲ增加了 70 条称为 SSE（Streaming SIMD Extension）的新指令，显著提高了数字图片处理、三维图形处理、实时视频/音频处理以及语音识别处理等应用的处理速度和质量。

Intel 的 Pentium 4 于 2000 年 11 月问世，首次登场的 Pentium 4，主频有 1.4GHz 及 1.5GHz 两种。它新增 144 条 SIMD Extensions 2（SSE2）指令，具有 256KB 的 L2Cache、20 级超流水线技术使 CPU 指令的运算速度成倍增长，另一方面，高级动态执行 ADE 使流水线中所能处理的指令比 Pentium Ⅲ多 3 倍以上，并合理地预测指令分支。随后，主频为 1.6GHz、1.7GHz、1.8GHz、1.9GHz、2.0GHz 的 Pentium 4 相继问世，近期内，主频还将进一步提高。

从微型计算机的发展过程看，一方面是迅速提高微处理器的性能，例如，提高大规模集

成电路的集成度和速度，采用流水线技术，高速缓冲存储器技术、虚拟存储器管理技术、并行处理技术以及精简指令集 RISC 等技术。另一方面是提高整个微型计算机系统的性能，例如采用多处理机并行处理技术，用微处理器组成的多微处理机系统和阵列处理机系统将多个微处理器组合起来进行“并行处理”，大大提高了处理速度，同时在提高系统可靠性、充分利用资源及进行分布处理等方面，也具有重要意义。

微型计算机的发展史也包括软件的发展，它给微机应用提供了强有力的支持。

微型计算机具有功能强、产量大、体积小、价格便宜、性能可靠、适应性强等特点，微型计算机的应用已渗透到国民经济的各个领域，主要是在科学与工程计算、数据处理和信息加工、实时控制、计算机辅助设计、智能模拟等方面。

1.1.2 微型计算机系统的组成

微型计算机（简称微机）系统由硬件系统和软件系统两大部分组成，微型计算机系统的组成如图 1-1 所示。

1. 硬件　所谓硬件系统，是指构成微机系统的物理实体。从图 1-1 可看到，微型计算机由微处理器、存储器、接口电路以及连接在这些部件上的总线组成（这些部件将在后面各章中讲述）。

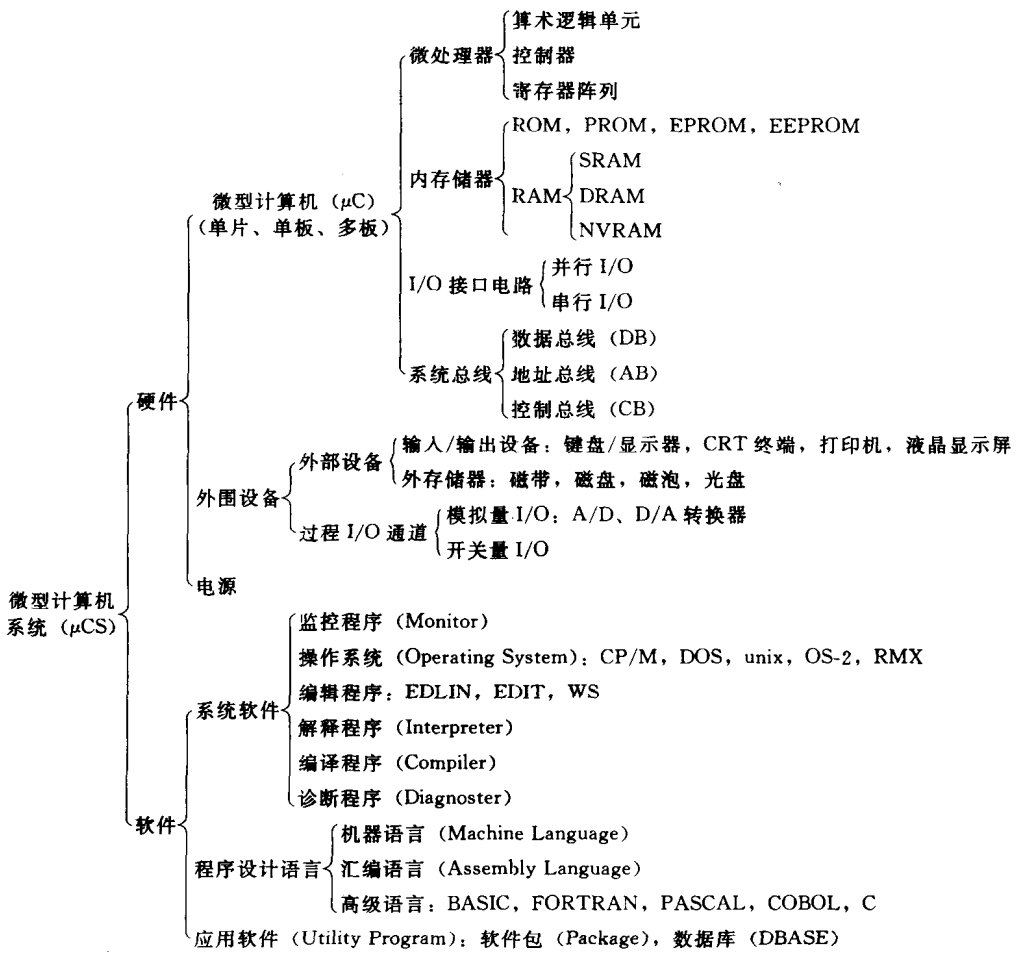

图 1-1　微型计算机系统的组成

如果把这些部件都集成在一个芯片上，由一个芯片构成一台计算机，称之为单片计算机；如果把上述各部件安装在一块印制电路板上而组成微型计算机，称之为单板计算机；如果将处理器、存储器以及I/O接口电路安装在不同的印制电路板上，由若干块这样的电路板组合而成的计算机，称之为多板计算机。

虽然微型计算机已经能够独立完成算术和逻辑运算，但由于没接必要的输入/输出设备，原始数据和程序无法输入，运算结果无法输出，因此，无法完成正常的计算机功能。在微计算机基础上加上各种外围设备就构成了微机系统的硬件部分。这是完成正常计算机功能的物质基础。

2. 软件　软件系统是微机所使用的各种程序的集合。它包括系统软件、各种程序设计语言、应用软件和数据库等。由于使用场合的不同，微机配上的软件规模也不同。

系统软件是由计算机供应商提供的，是指为了方便用户和充分发挥计算机效能的一系列程序，包括监控程序、操作系统、汇编语言、解释程序、编译程序、诊断程序等。系统软件中最为典型的是操作系统，它起着管理整个微机、提供人机接口以及充分发挥机器效率的作用。有了操作系统的微机系统，其所有资源都将由操作系统统一管理起来，用户不必过问各个部分资源的使用情况，而只是通过使用它的一些命令就行了。操作系统通常放在磁盘中。

程序设计语言是指用来编写程序的语言，通常分为机器语言、汇编语言和高级语言。机器语言是一种唯一能够直接被计算机识别和执行的语言，用二进制代码编写，不同的微机由于指令系统不同，使用的机器语言也不相同。机器语言由于通用性差、实际应用不方便，很少直接采用。

汇编语言是一种用助记符（指令的英文字或缩写）表示机器指令的语言，用它编写的程序可读性强，便于交流但计算机不能识别，所以汇编语言程序（称源程序）必须经汇编程序翻译加工（称自动汇编）成对应的机器语言程序（称目标程序），计算机才能执行。汇编语言程序也可人工翻译（称手编）。机器语言和汇编语言都是面向机器的初级程序设计语言，使用它便于利用计算机的所有硬件特性，是能直接控制硬件的语言。高级语言是面向用户的算法语言。使用它，用户不必了解计算机的指令系统，其所用的语句也与实际问题更接近，编写的程序通用性强，常用的高级语言有C语言、PASCAL语言、BASIC语言及FORTRAN语言等。用高级语言编写的程序也必须经过各种相应的解释程序或编译程序把它翻译成机器语言表示的程序，计算机才能识别和执行。

应用程序是系统的用户为解决自己的特定问题的需要而设计的程序或购买的程序，应用程序可按功能组成不同的程序包，以减少重复的编程工作。

在数据处理系统中，微机需要处理大量的数据、检索和建立大量的各种各样的表格，对这些数据和表格按一定的形式和规律加以组织，这就是建立了数据库。数据库管理系统则是为了便于用户建立数据库，并方便对库中内容进行询问、显示、修改以及输出打印各种表格的软件。例如FOXPRO就是广泛采用的数据库管理软件。

由此看来，硬件和软件是组成微机系统不可缺少的组成部分。组成计算机的硬件可以影响计算机的功能，但是，没有软件的计算机（称裸机）是不能提供使用的，相反，丰富的软件是对硬件功能的强有力的扩充。

1.1.3 微型计算机的典型结构及工作原理

我们先从一个以实际系统为基础经过简化的8位微机系统的典型模型开始，讨论其基本

结构和工作原理，然后再过渡到实际的16位微机系统。

1. 微机硬件系统连接　一种典型的微机硬件系统结构如图1-2所示。微机的核心部件CPU是靠三组总线将存储器、I/O接口连接起来的。总线是一组用来进行信息传递的公共信号线，它由地址总线、数据总线和控制总线组成。微处理器、存储器和所有I/O设备之间的信息交换都通过总线进行。由于各部件均以同一形式挂在总线上，结构显得简单，它易于扩充，所以目前绝大多数微机硬件系统均采用这种结构。

(1) 数据总线DB (Data Bus)　数据总线用于传送数据信息，其位数与处理器字长相等，例如8位微机的数据总线有8条，16位微机的数据总线有16条。在计算机中“数据”有比较广的涵义，数据总线上所传送的可以是真正的数据，也可以是指令代码、某些状态信息等。数据总线是双向的，它既可供处理器送出数据，也可供其他部件将数据送至微处理器内部。8位微机的8根数据线分别表示为D_7～D_0，D_0为最低位。

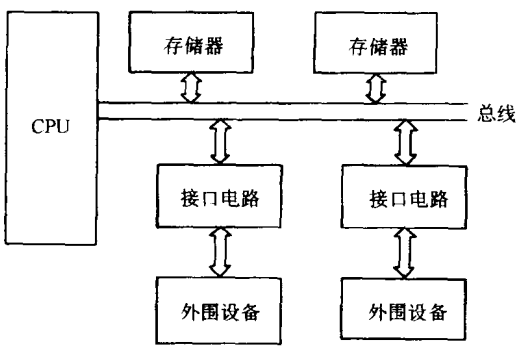

图1-2　微机硬件系统结构

(2) 地址总线AB (Address Bus)　地址总线是传送地址信息的一组线，是微处理器用来寻址存储器单元或I/O接口用的总线。其总线宽度(位数)将决定微处理器当前可寻址的内存储器容量范围，例如8位微处理器有16条地址线（分别用A_{15}～A_0表示，A_0为最低位)，可以寻找$2^{16}=65535$个不同地址，用十六进制数表示的地址范围为：0000H～FFFFH。

(3) 控制总线CB (Control Bus)　控制总线是系统中控制信号的传输线，其中有微处理器送往存储器和外围设备的输出控制信号，如读、写、访问请求信号等，也有外设通过接口反馈给微处理器的输入控制信号，如中断信号、总线请求信号、等待信号等。

上述地址线、数据线和控制线是计算机系统内各功能模块（如CPU、内存、I/O接口等）之间相互连接的总线，称系统总线，又称板间总线或内总线。

关于总线的进一步讨论，详见第9章。

2. 存储器组织　存储器用来存放数据和程序。程序是计算机操作的依据，数据是计算机操作的对象，它们都用二进制代码的形式表示，存放在存储器中。存储器由存储体、地址寄存器、地址译码器、数据缓冲器以及控制电路组成，存储器结构如图1-3所示。

存储体由很多存储单元组成，每个存储单元存放一个数据或一条指令，每个存储单元有一个唯一的编址，它可以按字编址，也可以按字节编址。如果按字编址，每个存储单元宽度(二进制位)就是CPU的字长；如果按字节编址，每个存储单元的宽度就是一个字节即8位。微机通常是按字节编址。

如图1-3所示，当CPU要对存储器进行读操作时，CPU通过16条地址总线将存储单元的地址送入地址寄存器，再经地址译码器译码后，寻找到指定的存储单元。CPU再发来读信号RD，将其中存放的8位数据读出到数据缓冲器，然后通过数据总线输入到CPU中。

当CPU要对存储器进行写操作时，除了和读操作一样，由CPU发来16个地址信号从而寻找到指定的存储单元外，CPU还通过数据总线把要写入的8位数据送到数据缓冲器，在写

信号 WR 作用下，把数据写入到所指定的存储单元。要注意的是，写入操作破坏了该单元原存内容，即由新内容代替了原存内容。

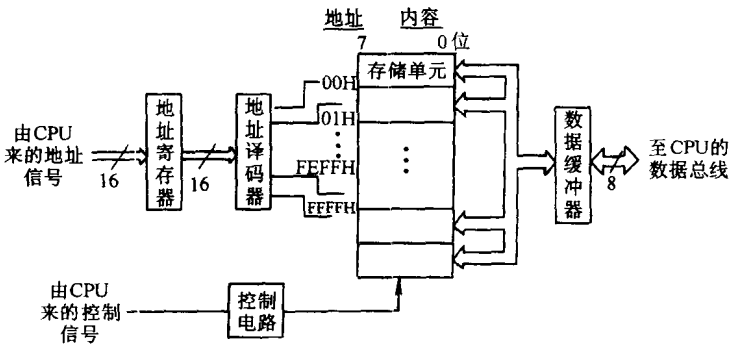

图 1-3　存储器结构

3. 微处理器组织　一个典型的 8 位微处理器的内部结构如图 1-4 所示。它主要由寄存器组、运算器、控制部件以及数据和地址缓冲器组成，各部件由片内总线相连接。

(1) 寄存器组　包括通用寄存器 B、C、D、E、H、L 和程序计数器 PC（专用寄存器）等。每一个通用寄存器都由 8 位触发器组成，它们可以暂存参加运算的操作数或中间结果，避免中间结果在 CPU 与存储器之间来回传送，以减少访问存储器的次数，提高运算速度。

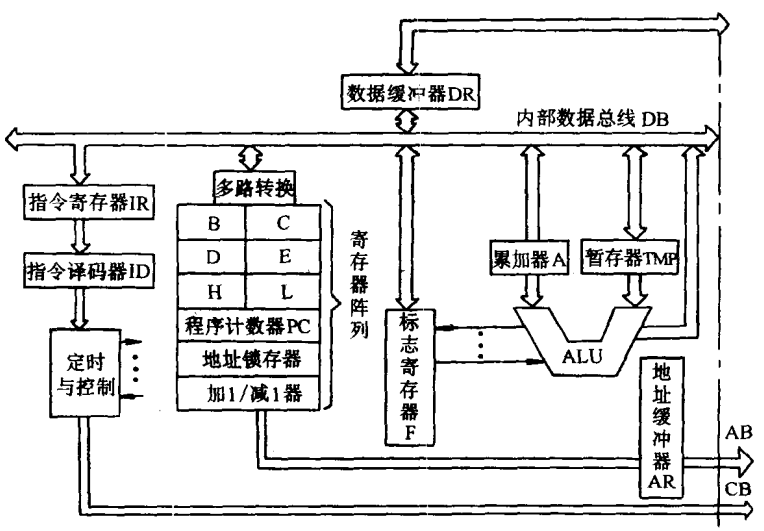

图 1-4　8 位微处理器的内部结构

程序计数器 PC（Program Counter）是一个 16 位寄存器，是专门设置来存放现行指令的 16 位地址的。每当存储器按 PC 指定地址取出一条单字节（或多字节指令的第一字节）指令后，PC 就自动加 1，指向下一条指令（或多字节指令的下一个字节）的地址，如此就可实现按先后顺序读取和执行指令了。仅当执行转移指令时，PC 的内容才由转移地址取代，从而改

变程序执行的正常次序，实现程序的转移。

(2) 运算器　由累加器A、暂存器TMP、算术逻辑单元ALU、标志寄存器F等部件组成。累加器A (Accumulator) 由8位触发器组成。它有两个作用：运算前寄存第一个操作数，是ALU的一个输入端；运算后存放ALU的运算结果。

暂存器TMP (Temporarty) 是一个8位的内部寄存器，用来暂存从内部数据总线送来的来自寄存器或存储单元的另一操作数，是ALU的另一个输入端。它不能由使用者用程序控制。

算术逻辑单元ALU (Arithmetic Logic Unit) 的功能是完成各种算术逻辑运算。它以累加器A的内容作为第一个操作数，以暂存器TMP内容作为第二操作数，有时还包括由标志寄存器F送来的进位C。操作结果S送回累加器A，同时也把表示操作结果的一些特征送标志寄存器F。

标志寄存器F (Flag) 用于指示CPU执行当前指令的结果特征和运行状态，如运算结果是否为零，运算结果有无进位等，可以用来作为控制程序转移的条件。

(3) 控制部件　包括指令（操作码）寄存器IR、指令译码器ID、时序及控制电路。CPU根据PC指定的地址，把指令的操作码从存储器取出经数据总线DB送到IR中寄存，ID对操作码进行译码产生相应操作的控制电位，再与定时信号组合形成相应的控制信号，称微操作，它按时间的先后顺序控制适当的部件，完成适当的操作，从而完成指令规定的任务。

(4) 数据和地址缓冲器　是总线缓冲器，用来隔离微处理器的内部总线和外部总线并增加总线驱动能力。数据缓冲器DR是8位双向三态缓冲器，地址总线缓冲器AR是16位单向三态缓冲器。

4. 微机工作过程　为了说明微机的工作过程，我们举一个简单程序的实例。

若要求计算机求解3+5=?。为解决这个问题，需要编一个程序，执行所编程序，就可以求得3+5=8。

要编程序，就需知道计算机能提供哪些指令。每类CPU都有自己的指令系统，它包括计算机所能识别和执行的全部指令，通常有几十到上百条。一般，指令由操作码和操作数或操作数地址两部分组成，操作码指明指令要完成什么类型操作（例如，数的传送、加等），操作数即是操作对象，为了帮助记忆，用助记符来代表操作码，如Z80CPU中，数的传送用LD，加法用ADD，暂停用HALT。

现在，我们根据使用的CPU指令系统，用其中合适的指令，编写出完成上述题目的汇编语言程序如下：

```
LD      A，3        ；将数3送入累加器A
ADD     A，5        ；将累加数A中的3和5相加，和数在A中
HALT                ；停机
```

微机不能直接理解指令助记符和十进制数，因此，程序应转换成二进制表示：

第一条指令：00111110；传送指令LD A，n的操作码

　　0000C011；操作数3

第二条指令：1100C110；加法指令ADD A，n的操作码

　　0000C101；操作数5

第三条指令：0111C110；停机指令的操作码

用这三条指令所编程序共用 5 个字节，若将它们按顺序放在 0000 号开始的存储单元中，存储器中程序存放情况如图 1-5 所示。

假设这段程序已存入存储器，第一条指令执行过程如图 1-6 所示。在执行时，首先给程序计数器 PC 赋以第一条指令的地址 0000，接着开始取第一条指令，执行第一条指令；再取第二条指令，执行第二条指令；…，直至遇到暂停指令为止。

存储地址	存储单元内容
0000	0011 1110
0001	0000 0011
0010	1100 0110
0011	0000 0101
0100	0111 0110

图 1-5　存储器中程序存放情况

第一条指令取指过程：

1）CPU 将 PC 的内容 0000 送至地址缓冲器 AR。

2）PC 的内容自动加 1，变为 0001。

3）AR 中的内容 0000 经地址总线送至存储器，经地址译码器译码，选中 0000 单元。

4）CPU 经控制总线发出读命令到存储器。

5）在读命令控制下，所选中的 0000 单元的内容 00111110 读到数据总线 DB 上。

6）经数据总线送至数据缓冲器 DR。

7）因是取指阶段，读出的内容必是操作码，故 DR 将它送至指令寄存器 IR。

8）经指令译码器译码，时序与控制电路发出执行这条指令所需的各种控制命令。

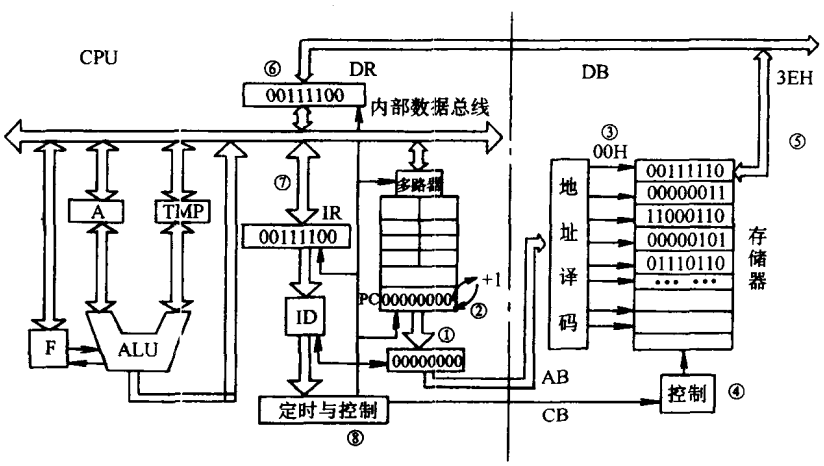

图 1-6　第一条指令执行过程

第一条指令的执行指令过程：

指令译码后，CPU 判定是一条 LD A，n 指令，操作数放在第二个字节，因而执行第一条指令，必须是取出第二字节中的数并放到 A 中。具体的执行过程如下：

1）CPU 把 PC 的内容 0001 送至 AR。

2）PC 的内容自动加 1，指向下一单元。

3）AR 的内容 0001 经地址总线送到存储器，经译码器译码，选中 0001 单元。

4）CPU 经控制总线发出读命令到存储器；

5）在读命令控制下，将所选中的 0001 单元内容 00000011 读到数据总线 DB 上。

6）经数据总线 DB，把读出数据 00000011 送到数据缓冲器 DR。

7）现在是执行指令阶段，读出的是操作数应由 DR 通过内部数据总线送入 A 中。

至此，第一条指令执行完毕，数据 00000011 在累加器 A 中。

同样，第二条指令在取指阶段取出加法指令操作数；在执行阶段取出操作数 00000101 送至暂存器 TMP 中。ALU 完成加法运算，和数 00001000 放在累加器中。

取出第三条指令，经译码 CPU 知是停机指令。执行这条指令，机器停止全部操作。

可以看到，微机的工作过程就是执行程序的过程。执行程序的过程，就是周而复始地从存储器取出指令、分析指令（译码）和执行指令的过程。

1.2 计算机中的数及其编码

1.2.1 机器数和真值

在计算机中，为了表示正数和负数，用最高位作为符号位。当该位为“0”时，表示正数；当该位为 1 时，表示负数。这就是说，数的符号在计算机中也数码化了。我们把一个数在机器（计算机）中的表示形式称为机器数，而把这个数本身，即用“+”、“−”号表示的数，称为数的真值。

真值可以用二进制表示，也可用十进制表示。例如

N1＝＋1101001B＝＋105 ，N2＝－1101001B＝－105

N1 和 N2 对应的机器数为

N1：01101001，N2：11101001

1.2.2 带符号数、无符号数

用 0 表示正数、用 1 表示负数的符号，这种表示数的方法，称为带符号数的表示方法。所表示的数，叫带符号数。带符号数的最高位为符号位。

如果把全部有效位都用以表示数的大小，这种表示数的方法，称为无符号数的表示方法。所表示的数，叫无符号数。

机器数 11001011 若看作带符号数，则其真值为－75；若看作无符号数，则其真值为 203。

1.2.3 原码、反码和补码

一个数的数值和符号都用二进制数码来表示，那么计算机对数据进行运算时，符号位应如何处理呢？是否也同数值位一道参加运算呢？为了妥善地处理好这个问题，就产生了把符号位和数值位一起进行编码的各种方法，这就是原码、反码和补码。

1. 原码　上述以最高位为 0 表示正数，为 1 表示负数，后面各位为其二进制数值，这样表示的机器数，称为原码。例如：

X_1＝＋67＝＋1000011B　　$[X_1]_{原}$＝01000011B

X_2＝－67＝－1000011B　　$[X_2]_{原}$＝11000011B

即一个数的原码，就是数值部分不变，而用最高为“0”和“1”分别来表示数的符号“+”和“−”的机器数。

在原码表示法中，根据定义，数 0 的原码有两种不同形式（设字长为 8 位）：

$[+0]_{原}$＝00000000B，　　$[-0]_{原}$＝10000000B

原码表示简单易懂，而且与真值的转换方便。但原码表示的数不便于计算机进行加减法运算，因为两个原码表示的数运算时，首先要判断它们的符号，然后再决定用加法还是用减法，因而使机器的结构相应地复杂化或增加机器的运算时间。为解决上述弊病，引入反码和

补码表示法。

2. 反码　正数的反码就是它的原码，负数的反码就是它的原码除符号位外各位取反。例如：

$X_1=83=+1010011B$　　$[X_1]_{反}=01010011B$

$X_2=-83=-1010011B$　　$[X_2]_{反}=10101100B$

在反码表示法中，根据定义，数 0 的反码也有两种不同形式（设字长为 8 位）：

$[+0]_{反}=00000000B$　　$[-0]_{反}=11111111B$

3. 补码　引入补码的目的是在于将加、减法运算简化为单纯的加法运算。

(1) 模的概念和性质　我们把一个计量器的容量或一个计量单位，称为模或模数，记为模 M。例如，一个 n 位二进制计数器，它的容量为 2^n，则它的模为 $M=2^n$。一个字长为 n 的计算机就是一个这样的有限计量系统，若 $n=8$，则 $M=2^8=256$。

模具有这样的性质，当模为 2^n 时，2^n 和 0 在 N 位计数器中的表示形式是相同的。例如，一个 n 位二进制计数器，它可以有从 0 到 2^n-1 的 2^n 个数。当它已经达到最大数 2^n-1 时，如果再加 1，计数器在最高位将溢出并变成全 0，即 n 位二进制计数器不能表示 2^n。或者说，2^n 和 0 在以 2^n 为模时，在计数器中表示形式是相同的。

(2) 补码的概念　现以时钟对时为例，假定时针正指 10 点，而此时标准时间是 6 点整。为了校准，可将时钟反拨 4 小时，使 $10-4=6$；也可将时钟顺拨 8 小时，使 $10+8=12+6$，由于表盘上 12 点和 0 点重合，可以把 12 看成 0，自动丢失了一个数字 12，使时钟指到 6 点。时钟实际上是一个模为 12 的计时系统，所以 $10+8$ 和 $10-4$（或 $10+(-4)$）是等价的。顺时针拨的数 8 与逆时针拨的数 4 对模 12 而言成互补关系。这就是说，在模 12 的意义下，负数（如 -4）就可以转化为正数（如 $+8$）。因此，可以说 -4 的补码为 8，或者说 -4 和 $+8$ 对模 12 来说互为补码。

我们知道，某一正数加一个负数，实际上是做一次减法。但引入补码概念之后，可以将此正数加上该负数的补码，得到的结果同样是正确的。因此在计算机中，有了补码，就可以将减法转化为加法来进行。

(3) 补码的求法　带符号数 X 的补码求法如下：正数的补码等于它的原码，负数的补码就是它的反码加 1。例如：

$X_1=+67=+1000011B$，则 $[X_1]_{补}=[X_1]_{原}=01000011B$

$X_2=-67=-1000011B$，则 $[X_2]_{补}=[X_2]_{反}+1=10111101$

根据定义，0 的补码仅有一种形式：$[+0]_{补}=[-0]_{补}=00000000$

如果将 $[X]_{补}$ 再求补一次，即将 $[X]_{补}$ 除符号位以外各位取反加 1 就得到 $[X]_{原}$。

如果已知 $[X]_{补}$，那么对 $[X]_{补}$ 的每一位（包括符号位）都按位求反，然后加 1，结果即为 $[-X]_{补}$。由 $[X]_{补}$ 求 $[-X]_{补}$，通常称为变补。

8 位二进制数的原码、反码和补码的对应关系见表 1-1。

从表 1-1 可见：

1) 对于无符号数，8 位二进制数的范围为 0～255，n 位二进制数其表示的范围为 0～2^n-1。对带符号数，原码、反码和补码所能表示的数值范围是不完全相同的。8 位二进制数原码和反码的表示范围都为 -127～$+127$，补码数的表示范围为 -128～$+127$。n 位二进制数带符

号数，其原码和反码的表示范围：$-2^{n-1}+1\sim+2^{n-1}-1$；补码的表示范围：$-2^{n-1}\sim2^{n-1}-1$。

2）原码和反码对于 0 有两种表示方法，补码只有一种表示方法，即在补码中数 0 的表示形式是唯一的。

3）带符号数，其最高位表示符号位。对于正数，三种编码都是一样的，即 $[X]_{原}=[X]_{反}=[X]_{补}$；对于负数，三种编码就不同了。原码、反码和补码的实质是用来解决负数在计算机中表示的三种不同编码的方法。

表 1-1　8 位二进制数的原码、反码和补码

8 位二进制	无符号数	原码	反码	补码
00000000	0	+0	+0	+0
00000001	1	+1	+1	+1
00000010	2	+2	+2	+2
⋮	⋮	⋮	⋮	⋮
01111101	125	+125	+125	+125
01111110	126	+126	+126	+126
01111111	127	+127	+127	+127
10000000	128	−0	−127	−128
10000001	129	−1	−126	−127
10000010	130	−2	−125	−126
⋮	⋮	⋮	⋮	⋮
11111101	253	−125	−2	−3
11111110	254	−126	−1	−2
111111111	255	−127	−0	−1

1.2.4　数的定点和浮点表示方法

在计算机中，数有两种表示方法，即定点表示法和浮点表示法。所谓定点表示法，就是小数点在数中的位置是固定不变的；所谓浮点表示法，就是小数点在数中的位置是浮动的。

通常，对于任意一个二进制数总可以表示为一个纯整数（或纯小数）和一个 2 的整数次幂的乘积形式 $N=2^P\times S$。其中，S 称为数 N 的尾数，P 称为数 N 的阶码，2 称为阶码的底。此处 P、S 都是用二进制表示的数。尾数 S 表示了数 N 的全部有效数字，阶码 P 指明了小数点的位置。

1. 定点表示法　当阶码为固定值时，这样的数，称为定点数。一般来说，小数点位置固定在哪个位置上并无限制，但为了简便起见，在计算机中有两种定点数是最常用的。

（1）定点纯整数　当 P=0，且尾数 S 为纯整数时，只能表示纯整数，称为定点纯整数。其格式为：

符号位	尾数 S（小数点）.

此时，把小数点固定在最低数值位右边，最高位为符号位，小数点本身不占位。

（2）定点纯小数　当 P=0，且尾数 S 为纯小数时，只能表示纯小数，称为定点纯小数。其格式为：

符号位	.（小数点）尾数 S

此时，把小数点固定在最高数值位左边，在小数点左边设有一位符号位，小数点本身不