单片机原理与接口应用

侯紫达　曹焕林　编著

兵器工业出版社

单片机原理与接口应用

侯紫达　曹焕林　编著

兵器工业出版社

内　容　简　介

单片机是一种体积小、耗电少、工作可靠、成本低的微型计算机，只需极少的外围器件，便可组成多种智能控制器。本书介绍单片机工作原理与接口应用，共十二章，重点介绍8098系列单片机及其指令系统、8098汇编语言程序设计和单片机的接口扩展等。各章末均附有习题及思考题。

本书可作为大专院校单片机应用课程的教材、短训班的培训教材，也可供具有高中以上文化程度的微型机用户和广大爱好者自学使用。

图书在版编目（CIP）数据

单片机原理与接口应用/侯紫达，曹焕林编著．—北京：兵器工业出版社，1997

ISBN 7-80132-389-0

Ⅰ．单…　Ⅱ．①侯…　②曹…　Ⅲ．①单片微型计算机-基础理论②单片微型计算机-接口　Ⅳ．TP368.1

中国版本图书馆CIP数据核字（97）第23185号

单片机原理与接口应用

侯紫达　曹焕林　编著

*

兵器工业出版社出版发行

（邮编：100081　北京市海淀区车道沟10号）

各地新华书店经销

北京市京通印刷厂印装

*

开本：787×1092　1/16　印张：16.25　字数：393千字

1997年12月第1版　1997年12月第1次印刷

印数：1～2000　定价：29元

前　言

单片机是在单片芯片上集成了CPU（中央处理机）、RAM（随机存取存储器）、ROM（只读存储器）、定时/计数器、并行I/O、串行I/O、多级中断控制以及A/D转换等多种输入/输出接口的微型机。单片机具有体积小、耗电少、工作可靠、成本低、设计灵活、使用方便等突出的优点，只需极少的外围器件，就可把它组成多种智能控制器。因此，单片机在过程控制、智能仪器仪表以及家用电器的控制等方面的应用十分广泛。

本书介绍单片机工作原理与接口应用，可用作大专院校单片机应用课程的教材，或用作单片机应用短训班的培训教材，也可供具有高中以上文化程度的微型机用户和广大爱好者自学使用。

全书共分如下十二章：

第一章介绍计算机基础知识，供未掌握这方面基础知识的读者学习。

第二章为单片机概述。

第三章和第四章分别介绍8098系列单片机及其指令系统。

第五章介绍8098汇编语言程序设计。

第六章介绍中断系统。

第七、八、九、十章分别介绍定时器及其应用、高速I/O部件、A/D转换和脉冲调宽PWM、串行口SIO。

第十一章介绍单片机的接口扩展。

第十二章为单片机开发及单片机开发系统简介。

在前十一章中，各章章末均附有习题及思考题。

附录部分给出8098单片机指令表。

本书由侯紫达、曹焕林合编，其中第一、四、五、八、九、十一各章由侯紫达编写，第二、三、六、七、十、十二各章由曹焕林编写，最后全书由翁瑞琪统稿和审定。

限于水平，书中难免有疏漏和不足之处，欢迎广大读者批评指正。

作者

1997年8月

目　录

第一章　计算机基础知识

本章为便于一些不具备计算机基础知识的读者学习，简要介绍一些有关计算机的基础知识。主要有微型计算机系统的构成和工作原理、计算机中数的表示和编码以及一些常用的术语和主要技术指标。对于已有这方面基础知识的读者可跳过本章内容直接去学习下一章。

1.1　微机系统的组成和工作原理

1.1.1　微型计算机的系统组成

实用的计算机系统是由硬件系统和软件系统两大部分组成。

硬件系统是指组成计算机系统的实际物理实体，都是看得见摸得着的实际设备，可以是各种电子器件、导线、电源以及具有各种功能的外部设备等，也称为硬件。计算机的基本组成见图 1-1。

图 1-1 的硬件结构是基于冯·诺依曼的基本思想：即用二进制来表示数据和指令；将程序和数据事先存储；用运算器、控制器、存储器、输入和输出装置五大部件组成计算机。图中宽线代表数据流，窄线代表执行程序形成的控制信息，即指令流。数据流在指令流控制下存取和处理。计算机工作时，数据由输入设备送至运算器，再存入存储器中。运算时，数据又从存储器取至运算器进行处理，结果可存入存储器或从输出设备输出。指令从存储器取出送至控制器进行分析，按照指令的功能产生相应的一系列微命令，在严格的定时控制下完成各种各样的处理工作，从而有条不紊地使各部件协调一致工作。

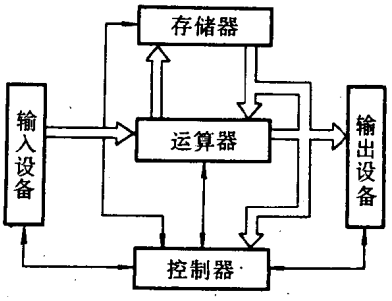

图 1-1　计算机的基本组成

一般来讲，无论大、中、小型计算机或者是微型计算机和单片机在本质上是没有多大区别的。微型计算机中的中央处理单元是把运算器、控制器集成在一片或几片大规模集成电路中，称之为微处理器 μP 或 MP（Microprocessor），以示区别，通常称为 MPU（Microprocessing Unit）。如果把微处理器（MPU）、程序存储器（ROM）、数据存储器（RAM）以及输入输出接口等集成在一片大规模或超大规模集成电路上，则称之为单片机。

微型计算机的硬件系统结构，通常是以微处理器为核心的单总线结构。单总线由数据总线 DB、地址总线 AB、控制总线 CB 组成。计算机中的存储器以及接口电路都挂接在这三条系统总线上，外部设备通过接口电路和总线相连。由于这种结构易于扩充，且结构简单，所以大多数微型机都采用了单总线结构。所谓单总线是指计算机中任何两个设备之间传输信息都必须通过这一组总线进行。因此，在同一时刻，总线中仅允许相同的一组信息流动，如图 1-2 所示。

CPU（MPU）内部包括有运算器、控制器和为数众多的寄存器等，内部数据也是通过总线结构来进行传输的。通常把 CPU 内的总线称为内部总线或 CPU 总线，把连接存储器和接

口电路的总线称为外部总线或系统总线。

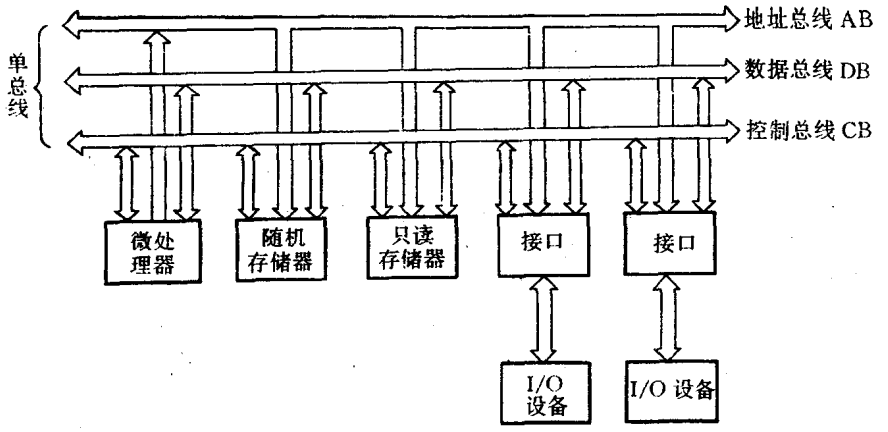

图 1-2 微型计算机硬件系统结构

数据总线用于传输数据（指令），不同类型的计算机由于处理数据的能力不同，数据总线的宽度也不同。

地址总线用于传送数据的地址，这些地址包括数据的源地址和目的地址。由于不同计算机存储器的容量不同，地址总线的宽度也不同。

控制总线用于传送控制信息，这些控制信息严格按计算机的定时关系发出，控制各部件的协调工作。

软件系统是指在计算机硬件系统基础上编制的各种程序的集合。它是用户和计算机硬件打交道的接口，体现为一些二进制信息，存放在存储媒体上，因此称为软件。通常分为系统软件和应用软件两大类。系统软件一般用户不能干予，例如：操作系统；用于汇编语言的汇编程序；高级语言的编译和解释程序；管理和服务的程序（如监控程序、调试程序、诊断程序、编辑程序等）。应用软件是针对解决各种实际问题而设计的程序，如企业管理程序、科学计算程序、数据处理程序、自动控制程序等等。

硬件系统和软件系统共同构成了计算机系统，二者缺一不可。没有硬件系统作为基础，软件是不能发挥作用的；同样，没有软件系统的硬件系统仅仅是裸机系统，也是没有任何用处的。随着现代计算机技术的飞速发展，在硬件系统和软件系统之间已没有一个明显的分界线，二者已密不可分．现在常采用软件硬化的方式，即把软件功能固化于硬件中。因此，总的趋势是两者统一融合，互相促进。

1.1.2 计算机的工作过程

1. 中央处理器的结构

中央处理器 (Control Processing Unit)，常写为 CPU，微型机中为加以区别，称为 MPU。一个模型机的 CPU 主要组成部分如图 1-3 所示。

由图 1-3 可见，微处理器主要由运算部件 ALU、累加器 A、数据寄存器 DR、程序计数器 PC、地址寄存器 AR、指令寄存器 IR、指令译码器、ID、微操作控制线路及状态标志寄存器 F 组成。各部分功能简述如下：

运算部件 ALU (Arithmetic Logic Unit) 用于进行各种算术、逻辑运算。由图 1-3 中可看出，运算过程中两个操作数一个取自累加器 A，另一个取自数据寄存器 DR。运算的中间结果

通常保存于 A 中。

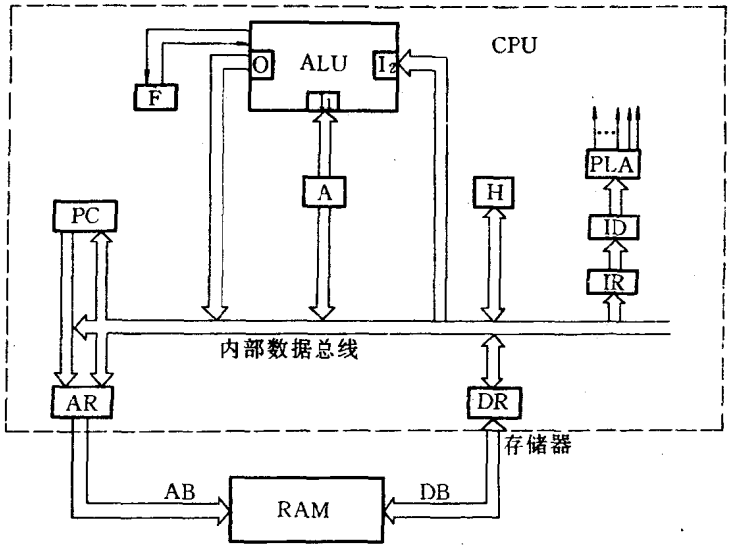

图 1-3　微处理器结构

累加器 A 运算前用于保存一个操作数，运算后用于保存结果。

数据寄存器 DR 用于暂存数据或指令。从存储器读出时，若读出的是数据，通过 CPU 的内部数据总线送至有关寄存器或运算器；如读出为指令，则通过内部数据总线送至指令寄存器 IR。

程序计数器 PC 存放执行指令的地址，根据 PC 的值从存储器相应单元中取出将要执行的指令，它具有自动加 1 的功能。

地址寄存器 AR 存放指令地址或操作数的地址。若为取指令操作，AR 接收 PC 的值执行取指令。若取的是操作数，操作数地址从内部数据总线将操作数地址写入 AR，从而从存储器中取出操作数。同样，若要向存储器写入数据，也要将写入地址送入 AR。

指令寄存器 IR 接收由存储器中取出的将要执行的指令。

指令译码器 ID 对指令寄存器 IR 中存放的指令进行译码，通过对指令的分析确定指令的相应操作。

微操作控制线路产生取指令和执行指令时的各种微操作控制信号。由于不同指令的操作过程不同，对应的微操作序列不同。因此，根据不同的指令对应控制信号的一种组合。

状态标志寄存器 F 由一些标志位组成，用于寄存执行指令后产生的状态标志信号。如运算过程中是否发生溢出、运算结果是否为零、有无进位发生等等，它是判断程序流向的重要依据。

2. 存储器

存储器用于存储程序和数据，程序（指令）、数据都是用二进制代码来表示。用八个二进制位表示一个字节，根据计算机处理数据能力的不同，一个字由一个或几个字节构成。当一个字用于表示数时称为数据字；表示一条指令时，称为指令字。不同的计算机存储器中包含的存储单元个数不同，通常存储器的容量取决于地址总线的宽度。例如：若地址总线宽度为 8 位，则存储器的最大容量为 2^8，即 256 个单元；若宽度为 16 位，则容量为 2^{16}，即存储单元的个数为 65536。为了能区分不同的存储单元，对每个存储单元赋以一个唯一的编号，这个编

号称为存储器单元的地址，它是由地址线上取不同的电平（0 或 1）决定的。在每一个存储单元中存放不同的二进制信息，这个信息就是存储单元的内容。因此，地址和内容是两个不同的概念。图 1-4 所示为模型机存储器的结构。

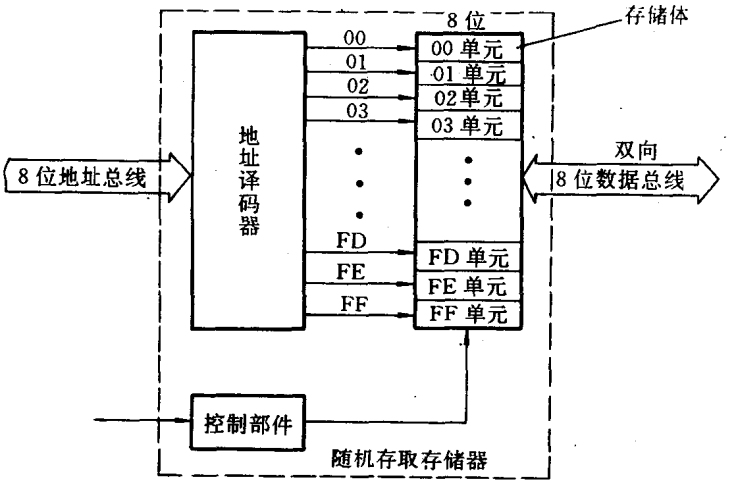

图 1-4　模型机存储器的结构

存储器的地址和内容常用十六进制数来表示，图中用两位十六进制数表示了存储器的地址和内容，若地址为十六位的，则可用四位十六进制数来表示。由图可以看出，存储器是由存储体、地址译码器和控制部件组成。在计算机工作过程中，由地址总线送来的地址，经过译码选中与此相对应的某个存储单元，在控制部件的控制下，完成了对存储器的读写工作。

（1）存储器的读操作

存储器的读（取）操作过程如图 1-5 所示。

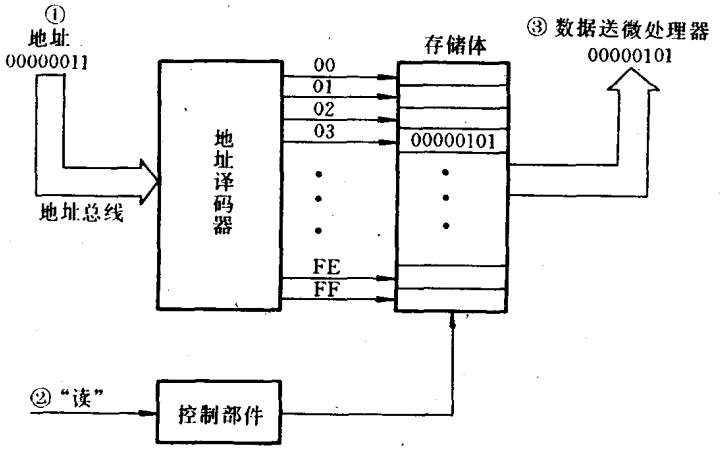

图 1-5　存储器的读操作

以下具体说明读操作的过程。设在 03H 单元中存放的内容为 05H，则分成以下几个步骤完成取数的过程：

1）CPU 的地址寄存器 AR 给出地址码 03H，通过地址总线送到存储器的地址译码器；

2）存储器的地址译码器经译码选中 03H 单元；

3）CPU 发出读命令，在控制部件的控制下，03H 单元的内容 05H 放到数据总线上并把它送数据寄存器 DR，完成了读操作。

在进行了读操作之后，03H 单元的内容仍为 05H，称为非破坏性读出。

（2）存储器的写操作

存储器的写（存）操作过程如图 1-6 所示。

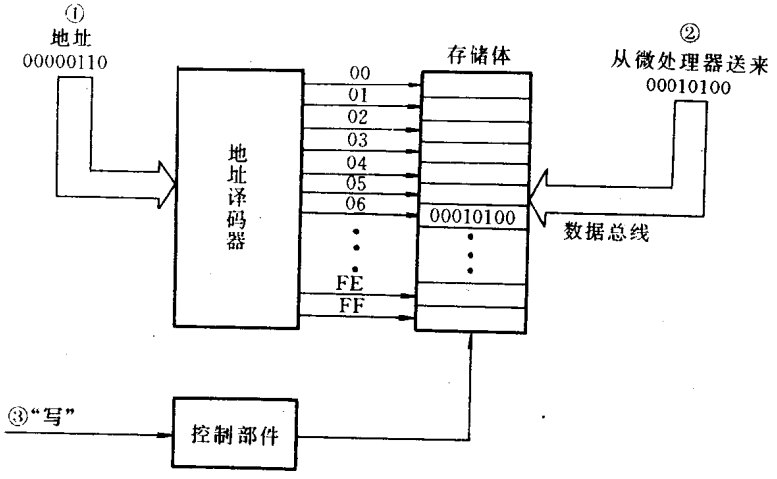

图 1-6　存储器的写操作

以下具体说明写操作的过程。设要给存储器中 06H 单元写入 14H，则分成以下几个步骤进行存数的操作：

1）CPU 的地址寄存器 AR 给出地址码 06H，通过地址总线送到存储器的地址译码器；

2）地址译码器经译码选中存储器的 06H 单元；

3）CPU 将数据寄存器 DR 的内容 14H 放到数据总线 DB 上，然后 CPU 发出写命令，在控制部件的控制下将 14H 写入 06H 单元，完成了写操作。

由于 06H 中原有的数据被新的数据所替代，因此存储器的写操作是破坏性的。

这种既能读出又能写入的存储器通常称为随机存储器，即 RAM（Random Access Memory）；只能读出，不能写入的存储器称为只读存储器，即 ROM（Read Only Memory）。

3. 工作过程

由前所述，已经知道数据的处理是依据指令的执行来进行的。因此，计算机的工作过程就是执行指令的过程。我们把要进行的工作用指令编成一段程序并把它存储在存储器中，顺序地一条一条执行指令就完成了所需要进行的工作。由此可见，指令序列的集合就构成了程序，而计算机的工作过程就是不断地取指令、执行指令的过程。

从计算机执行指令的过程来看，每一条指令的执行基本上可分成两个阶段，即取指阶段和执行阶段，不同的指令在取指阶段都具有相同的操作，称为公操作。而不同指令由于功能不同，在执行阶段执行的操作不一样，需要的时间也不一样，所以执行阶段对于不同的指令是不一样的。为了说明计算机的工作过程，现以图 1-3 所示的模型机为例来说明执行指令的过程。

假设要执行 5＋7 的操作，在计算机中完成这个工作，就需要编写程序去实现。现假设在模型机中已提供了如下三条指令，见表 1-1。

表 1-1 指令表

名　称	助记符	指令代码		注　释
立即数取至累加器	LD A，n	00111110 n	3E n	双字节指令，将第二个字节的立即数 n 送累加器 A
加立即数	ADD A，n	11000110 n	C6 n	双字节指令，累加器 A 和第二字节立即数 n 相加，结果在 A
暂　停	HALT	01110110	76	停止全部操作

用助记符表示的汇编语言程序如下：

LD A，5　；立即数 5 送累加器 A

ADD A，7；A 和立即数 5 相加，结果在 A

HALT　　；停止

由于计算机中仅能识别二进制数（或 16 进制数），汇编语言编制的程序必须转换成二进制数表示的机器码（或十六进制数）。表示如下：

第一条指令：00111110　操作码

　　00000101　操作数

第二条指令：11000110　操作码

　　00000111　操作数

第三条指令：01110110　操作码

这三条指令，占了五个字节（用八位二进制代码表示一个字节），假定存放在存储器中 00H 至 04H 单元，如图 1-7 所示。

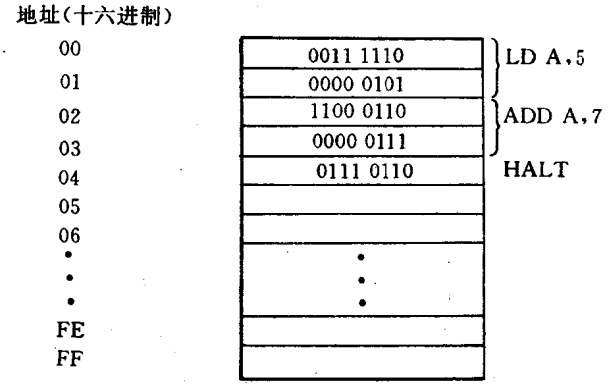

图 1-7　模型机存储器中的指令

执行上述程序，在程序计数器 PC 中赋于第一条指令的地址 00H；计算机开始执行第一条指令；

取指阶段：

1）PC 的内容 00H 送地址寄存器 AR。

2）PC 内容送至地址寄存器 AR 后，自动加 1，变为 02H，指向下一个单元。

3）地址寄存器 AR 的内容 00H 通过地址总线送至存储器地址译码器，经译码选中 00H 单元。

4）CPU 发出读操作命令。

5）00H 单元的内容 3EH 放到数据总线 DB 上，并经数据总线送到数据寄存器 DR。

6）因是取指阶段，读出为指令，把 DR 内容 3EH 送指令寄存器 IR。

7）经指令译码器译码，微操作控制部件发出该条指令的微操作序列控制命令。

以上是第一条指令的取指阶段。

执行阶段：

指令译码器经译码后，已知该条指令的功能为取出第二个字节的操作数并把它送至累加器 A，执行的操作为：

1）PC 的内容 01H 送地址寄存器 AR。

2）PC 的内容自动加 1，指向 02H。

3）地址寄存器 AR 的内容经地址总线送存储器的地址译码器，经译码选中 01H 单元。

4）CPU 发出读操作命令。

5）01H 单元的内容 05H 放到数据总线 DB 上，并经数据总线送数据寄存器 DR。

6）因读出的为操作数，故 05H 经 CPU 内部数据总线送累加器 A。

至此，第一条指令全部执行完毕。第二条指令的执行可参照第一条指令的执行过程写出相应的步骤。在取指阶段是相同的，如果一条指令的操作码不止一个字节，则取指过程将按指令的操作码所占字节数递增。在执行阶段，由于指令的功能为作加法，取出的操作数不是送累加器 A，而是送算术逻辑运算部件 ALU 的另一个输入端，执行和累加器 A 中的数相加，并把结果保存到 A 中。第二条指令执行完后，在累加器 A 中为两数的和 12（十进制数），用十六进制数表示为 0CH。

第三条指令无操作数，取出后为停机指令，执行后将停止操作。

以上指令的执行过程可参照图 1-8 进行理解。

实际的计算机中，由于用途和功能不同，在结构上要复杂的多，并有所差别，但基本原理是一样的。不同计算机指令的助记符和操作码也不一样。

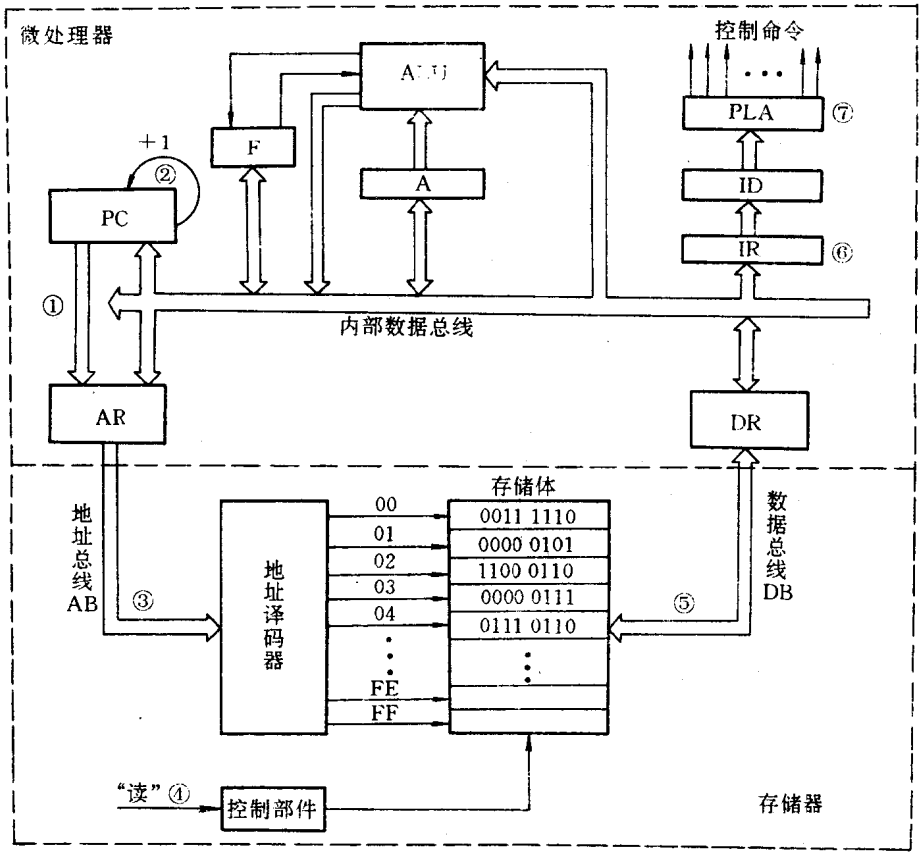

图 1-8　指令执行过程示意图

1.2 计算机中数的表示和编码

计算机中无论数据还是指令，都是用二进制来表示。也就是说，计算机仅能识别二进制代码。而人们习惯于十进制数，这就存在不同进位计数制之间的转换和实际的数如何在计算机中表示的问题。此外，人们习惯使用的字母和各种常用的符号在计算机中如何表示呢？这也是我们需要讨论的问题。

1.2.1 数的表示

1. 进位计数制和不同计数制间的转换

(1) 不同计数制和十进制的转换

现实生活中存在着各种各样的计数制，最常用的是十进制，此外还有二进制、八进制、十六进制、十二进制、六十进制等。但无论什么样的进制数，都有以下两个特点：

1) 基数：所谓基数体现了该种计数制的进位单位。例如十进制是逢十进一；八进制是逢八进一，而基数在该种计数制中是不出现的。设在任意计数制中用 R（正整数）表示基数，则在每位中可能出现的数字是 0～$R-1$。

2) 位数：它体现了不同计数制中处在不同位时数值的大小。例如，十进制中个位的位权为 1(10^0)、十位的位权为 10(10^1)・・・；同样八进制数中的位权为 1(8^0)、8(8^1)・・・。

由以上两点，可以写出任意进制数转换成十进制的通式：

$$N=\pm\ (K_{n-1}\times R^{n-1}+K_{n-2}\times R^{n-2}+\cdots+K_1R+K_0R^0+K_{-1}\times R^{-1}+\cdots+K_{-m}\times R^{-m})$$

$$=\pm\sum_{i=-m}^{n-1}K_iR^i$$

式中 N 为任意进制数，其基数为 R，整数有 n 位，小数有 m 位。

例：$(325.14)_8=3\times8^2+2\times8^1+5\times8^0+1\times8^{-1}+4\times8^{-2}=213.1875$

$(25.4)_{16}=2\times16^1+5\times16^0+4\times16^{-1}=37.25$

$(10111.11)_2=1\times2^4+0\times2^3+1\times2^2+1\times2^1+1\times2^0+1\times2^{-1}+1\times2^{-2}=23.75$

在十六进制数中，A 表示 10，B 表示 11，…F 表示 15，因此十六进制数中每位的取值范围为 0～9、A、B、C、D、E、F。

例如：$(FF)_{16}=15\times16^1+15\times16^0=255$

(2) 十进制数转换成其它计数制

十进制数转换成其它进制数，需要按整数部分和小数部分分别转换，再用小数点连接到一起，在这里我们只介绍方法。

十进制整数转换成其它进制数的方法为：除以该进制的基数，取余数，直至商为 0，逆序排列。以十进制转换二进制就是除 2 取余，直至商为 0，逆序排列。

例：将 325 转换成二进制数

商	余数
2 | 325	1
2 | 162	0
2 | 81	1
2 | 40	0
2 | 20	0
2 | 10	0
2 | 5	1
2 | 2	0
2 | 1	1
0	

所以 $325=(101000101)_2$

十进制小数部分转换成其它进制的方法为：乘以该进制的基数，取整数，直至小数部分为 0 或达到精度要求为止，正序排列。十进制小数转换成二进制就是乘 2 取整，直至小数为

0 或达到要求为止，正序排列。

例：将 0.625 转换成二进制数

×2	小数部分	整数部分
1.25	0.25	1
0.5	0.5	0
1.0	0	1

所以 $0.625=(0.101)_2$

在十进制数转换成其它进制过程中，整数部分可精确转换，小数部分不一定能精确表示成二进制的纯小数。例如，十进制数 0.725 转换成二进制数为：

×2	小数部分	整数部分
1.45	0.45	1
0.9	0.9	0
1.8	0.8	1
1.6	0.6	1
1.2	0.2	1
0.4	0.4	0
0.8	0.8	0
·	·	·
·	·	·
·	·	·

所以 $0.725\approx(0.10111)_2$，取五位小数

对于十进制数转换成其它进制数，同样可采用上面的方法去进行，只不过计算机中常用的八进制、十六进制和二进制转换起来非常方便，故以二进制数举例，读者亦可用除八取余（整数）、乘 8 取整（小数）将十进制数转换成八进制数。

（3）二进制和八进制、十六进制数之间的转换

由于用二进制数表示十进制数，位数较长，读写都不方便，计算机中常采用八进制和十六进制数来表示二进制数。

二进制数转换成八进制数的方法是：从小数点出发，分左右两路，每三位二进制数为一节，整数高位不够三位直接转换，小数末位不够三位用零补足三位，每三位二进制数用一位八进制数代替。

例：将 $(101101101.1101)_2$ 转换成八进制数。

$$(101101101.1101)_2=(\underline{101}\ \underline{101}\ \underline{101}\cdot\underline{110}\ \underline{100})_2=(555.64)_8$$

八进制数转换成二进制数的方法是：把每位八进制数按三位二进制数展开，小数点保持不动，最高位的零和最低位的零（后面再无有效数字）可以不写。

例：将 $(374.74)_8$ 转换成二进制数。

$$(374.74)_8=(\underline{011}\ \underline{111}\ \underline{100}\cdot\underline{111}\ \underline{100})_2=(11111100.1111)_2$$

十六进制数和二进制数之间的转换和八进制和二进制数之间的转换区别，仅在于由十六进制转换成二进制时，每一位十六进制数按四位二进制数展开，二进制数转换成十六进制数

时，只需四位二进制数为一节，其余完全相同。

例：将 $(3FB \cdot E6)_{16}$ 转换成二进制数。

$$(3FB \cdot E6)_{16} = (\underline{11}\ \underline{1111}\ \underline{1011} \cdot \underline{1110}\ \underline{011})_2$$

例：将 $(101101101 \cdot 11011)_2$ 转换成十六进制数。

$$(\underline{1}\ \underline{0110}\ \underline{1101} \cdot \underline{1101}\ \underline{1})_2 = (16D \cdot D8)_{16}$$

2. 计算机中数的表示

(1) 真值和机器数

计算机中数的表示和我们日常生活中一样，也分做正数和负数。那么，在计算机中如何表示正和负呢？计算机中的一个二进制位有两种状态，即 0 和 1。在计算机中对符号位的处理常用 0 表示正，1 表示负，这样做到了“符号数码化”。因此，在计算机中使用的连同符号数码化的数称为机器数。而包含正、负号在内的数来表示原值称为真值。

设：$N1 = +0101011$

$N2 = -0101011$

这样的数称为真值，若对符号进行数码化则：

$N1 = 00101011$

$N2 = 10101011$

此时，$N1$、$N2$ 为机器数。

由于对符号进行了数码化，计算机中表示的数有了正、负之分，用这样的方法表示的数也称为带符号数。如果约定，机器的全部有效位全部用来表示数值的大小，而没有符号的区分，这样的数实际上相当于数的绝对值的大小，把这样的数称为无符号数。

例如，设 $N1 = 00101011$，作为带符号数表示 $+43$，用无符号数表示为 43，实际上二者的值是相等的。而对 $N2 = 10101011$，作为带符号数表示 -43，而作为无符号数，由于最高位不再是符号，而表示数值的大小。因此，它表示 171，二者是不相等的。

在带符号数中，对于数值部分的表示在计算机中有三种不同的方法，分别称为原码、反码、补码。

(2) 原码、反码、补码

在介绍这三种编码之前，先引入模的概念。

某一计量器的容量，称为该计量器的模，记为模 M 或 mod M。

一个 n 位二进制数的计数器，表示的最小数为全 0（n 个 0），表示的最大数为全 1（n 个 1），一共能表示 2^n 个不同的计数值，容量为 2^n，故在 n 位计数器中，模记为 mod 2^n。

模的性质是模和 0 等价。在 n 位计数器中的最大值是 2^n-1，若在此基础上再加 1，则数值变成 2^n，在机器中表示应为 $1\underbrace{0\cdots0}_{n个0}$，但由于仅能用 n 位表示数值，1 是丢掉的。因而，模和 0 等价。

1) 原码

在原码中，用最高位表示符号，0 表示正数，1 表示负数，数值部分用二进制的绝对值表示。

例如：$X = +1101011$　　$[X]_{原} = 01101011$

$X = -1010111$　　$[X]_{原} = 11010111$