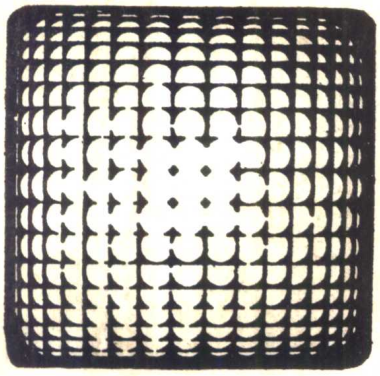

中央广播电视大学出版社

微机原理与应用自学指南

陶龙芳

微机原理与应用自学指南

陶 龙 芳

中央广播电视大学出版社

微机原理与应用自学指南

陶 龙 芳

*

中央广播电视大学出版社出版
新华书店北京发行所发行
国防科工委印刷厂印装

*

开本787×1092 1/16 印张8 千字 181
1988年8月第1版 1989年2月第2次印刷
印数：15,001～40,000

定价：1.75元

ISBN 7—304—00269—7/TP·19

前　言

“微机原理与应用”课程从应用的角度比较全面系统地分析和论述了微型计算机的硬件和软件，内容十分丰富。考虑到电大学员的学习条件及电视授课学时偏少的特点，为帮助学员自学，特编写了这本《微机原理与应用自学指南》。

《微机原理与应用自学指南》全书共分七章，每章内容包括三部分：第一部分是内容概要，根据教学大纲的要求，用简短的篇幅概括了必须掌握的基本内容；第二部分是自学重点，强化了教学内容中的重要环节，加深学员对主要问题的理解；第三部分是复习题，用以巩固所学的知识。

《微机原理与应用自学指南》一书采用概念叙述与典型例题相结合的讲授方法，便于自学，对帮助学员平时学习和期末复习具有指导意义。

《微机原理与应用自学指南》虽为电大学员编写，但就其内容来说，亦可供其它大专院校学生及从事计算机应用工作的工程技术人员参考。

由于编者水平有限，编写时间仓促，缺点错误难免，敬请批评指正。

编　者

1988年2月

目　录

第一章　微机系统导论

一、内 容 概 要

(一)微型计算机系统的组成

微型计算机系统由硬件系统和软件系统两大部分组成。

1．硬件系统是指由逻辑部件组成的微型计算机的物理实体。它的组成结构如图1-1所示，由微处理器、存贮器组成主机，主机通过接口电路与外部设备相连接。各个部件按同一形式挂在单总线上，单总线是一组用来进行信息传送的公共信号线、单总线包括若干条数据总线（Data Bus）、若干条地址总线（Address Bus）和若干条控制总线（Control Bus）。在单总线上有两股信息流在流动：数据信息流和控制信息流。

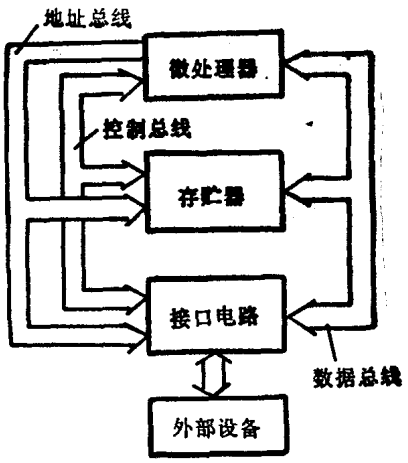

图1-1 微型计算机硬件系统结构示意图

微处理器由运算器和控制器构成，利用大规模集成电路技术集成在一个芯片上。运算器是执行算术运算和逻辑运算的部件。控制器控制输入设备的启动或停止，控制运算器按规定一步步地进行各种运算和处理，控制存贮器的读或写，控制输出设备输出结果等等。

存贮器通常可分为内存和外存两部分，微处理器中的存贮器是指内存而言。内存容量少，但存取速度快。微型机中大部分采用半导体存贮器。外存容量大，但存取速度慢，常用的有磁盘和磁带等。存贮器能存贮原始数据、中间结果及为了使机器有条不紊地自动进行运算而编制的各种命令。

输入设备常用的有键盘等，用来输入原始数据及命令。输出设备常用的有CRT(字符显示器)、电传打字机、行打印机等，用来输出计算结果。输入设备和输出设备合称外部设备或外围设备，简称外设。

2．为了运行、管理和维修微型计算机所编制的各种程序的集合称为软件。其中：

(1)由机器设计者提供的，用来使用和管理计算机的软件，统称为系统软件。系统软件包括：操作系统、监控管理程序；各种语言的汇编或解释、编译程序；编辑程序、调试程序、装配程序、故障检查和诊断程序等。

程序库：各种标准子程序的集合。

(2)程序设计语言：这是编写程序的工具，微型机常用汇编语言和各种高级语言。

(3)用户利用计算机以及它所提供的各种系统软件，编制的解决用户各种实际问题的程序称为应用软件。应用软件可以逐步标准化、模块化，形成解决各种典型问题应用程序的组合，称为软件包。

(4)数据库及数据库管理系统。

3. 通常人使用程序设计语言编制应用程序，在系统软件的支持下操纵微型计算机的硬件系统，来完成各种工作任务。

(二)存贮器组织

存贮器用来存放数据和程序，数据和程序在微型机内部均以二进制代码表示，八位二进制数字组成一个字节，一至几个字节组成一个字（Z80规定一个字为两个字节，即16位二进制数）。用字表示的数据，称为数据字，用字表示的指令称为指令字。一个存贮器划分为很多单元，称为内存单元，单元的编号称为地址。要注意单元的地址和单元的内容是不同的。

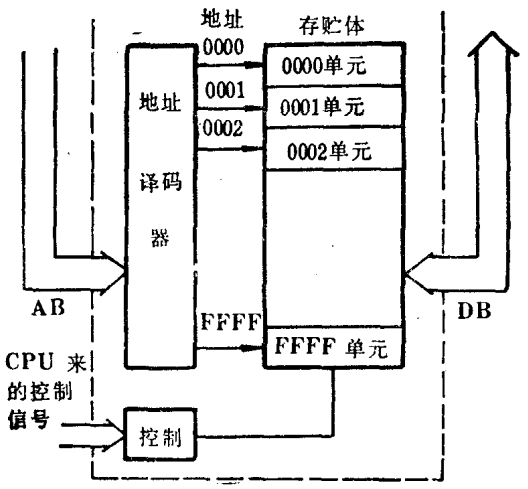

图1-2 存贮器结构图

存贮器的组织如图1-2所示，由存贮体、地址译码器和控制电路组成。

存贮器的读操作如图1-3所示。假设CPU（即中央处理单元，对微型机来说，就是指微处理器）的地址寄存器AR给出了地址1FE0H，通过地址总线AB送给存贮器，存贮器中的地址译码器对它进行译码，找到1FE0H内存单元。然后CPU发出读控制命令，于是1FE0H单元的内容80H读出送至数据总线DB，由它再送至数据寄存器DR。

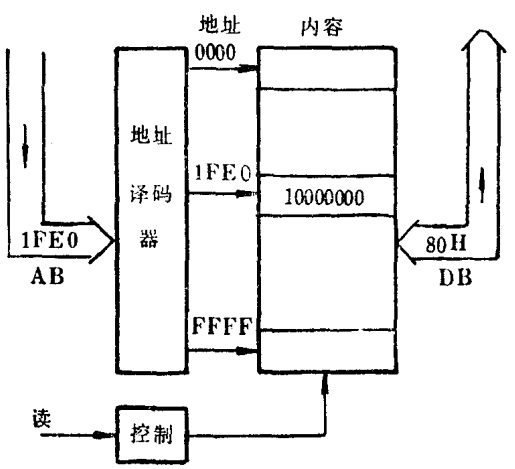

图1-3 存贮器读操作示意图

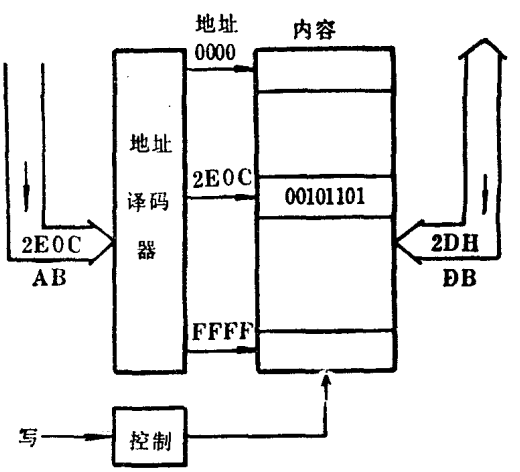

图1-4 存贮器写操作示意图

存贮器的写操作如图1-4所示。假设CPU的地址寄存器AR给出了地址2E0CH，通过地址总线AB送给存贮器，存贮器中的地址译码器对它进行译码，找到2E0CH内存单元。然后CPU发出写控制命令，于是数据总线DB上的信息2DH就可以写入到2E0CH内存单元了。

(三)微处理器的结构

以Z80微处理器为例说明，它的结构如图1-5所示。主要部分有：

1. CPU内部寄存器组：分为通用寄存器和专用寄存器两种(见下面(五)内容)。

2. 算术和逻辑单元ALU：ALU通过内部数据总线与内部寄存器和外部数据总线交换信息。ALU所能完成的功能有：加、减、与、或、异或、比较、左移、右移、循环、加1、减1、位操作等。

3. 指令寄存器、指令译码器和CPU控制：从存贮器中取出来的指令，送入到指令寄存器中，再由指令译码器译码，通过定时和控制线路，在规定的时刻，发出例如寄存器传送加或减或各种逻辑运算等所需要的CPU内部控制信号，以及发出所需要的CPU外部控制信号。

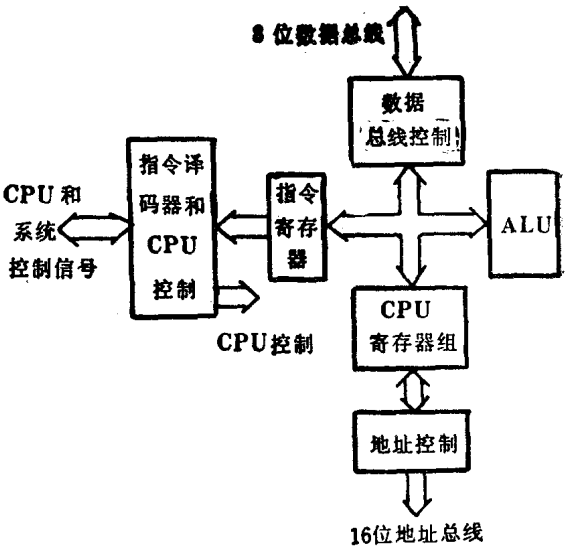

图1-5 Z80-CPU的内部结构示意图

(四)微机的工作过程

微机的工作过程是执行程序的过程。程序由指令序列组成，因此，执行程序的过程就是执行指令序列的过程。执行指令序列就是周而复始地取指令、执行指令，所以微机的工作过程，也就是不断地取指令、执行指令的过程。

例如计算两个数5和7相加的程序如下：

```
LD      A, 5        ; 将数5送累加器A
ADD     A, 7        ; 将累加器A中内容5
                    ; 和数7相加，结果仍
                    ; 放A中
HALT                ; 停止操作
```

程序在存贮器中的存放示意图如图1-6所示。

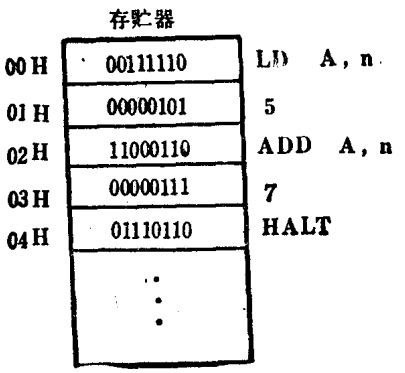

图1-6 程序存贮示意图

说明程序执行的过程：

1. 当机器进入运行状态时，首先把第一条指令所在的地址00H赋给程序计数器PC，然后就进入第一条指令的取指阶段。见图1-7，步骤为：

(1)PC的内容00H送至地址寄存器AR。

(2)当PC的内容已经可靠地送至AR后，PC的内容自动加1，变为01H。

(3)AR通过地址总线AB，把地址00H送至存贮器，经译码后，找到相应的内存单元。

(4)CPU发出“读”命令。

(5)选中的存贮单元的内容3EH读出至数据总线DB。

(6)读出的内容3EH通过数据总线送至数据寄存器DR。

(7)因是取指阶段，读出的是指令，故DR把它送至指令寄存器IR，经过指令译码器ID译码后，发出执行指令的各种控制信息。

2. 转入执行第一条指令的阶段。通过对指令的译码知道，这是一条把操作数送累加器

A的指令，所以执行这条指令，就是将操作数从指令的第2字节中取出来送累加器A。见图

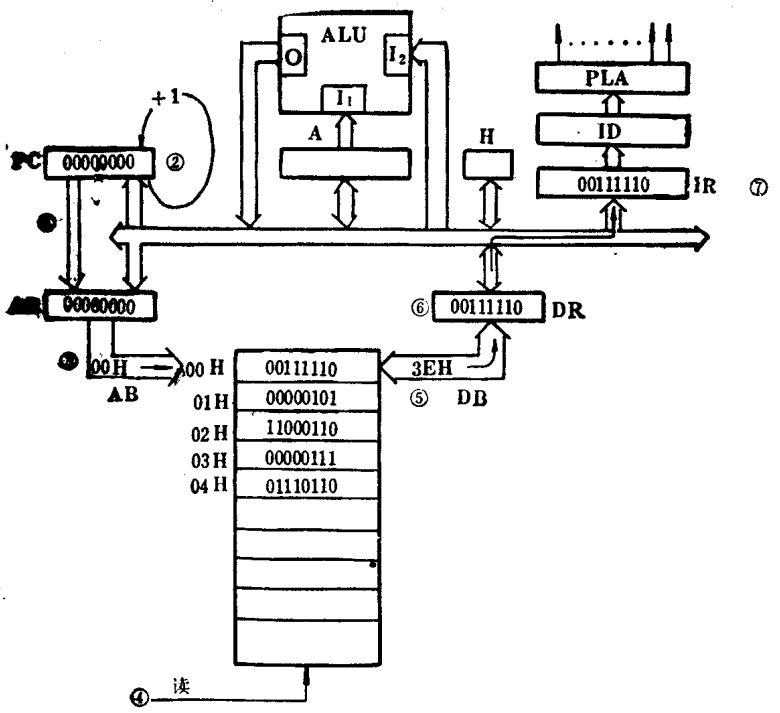

图1-7 取第一条指令的操作示意图

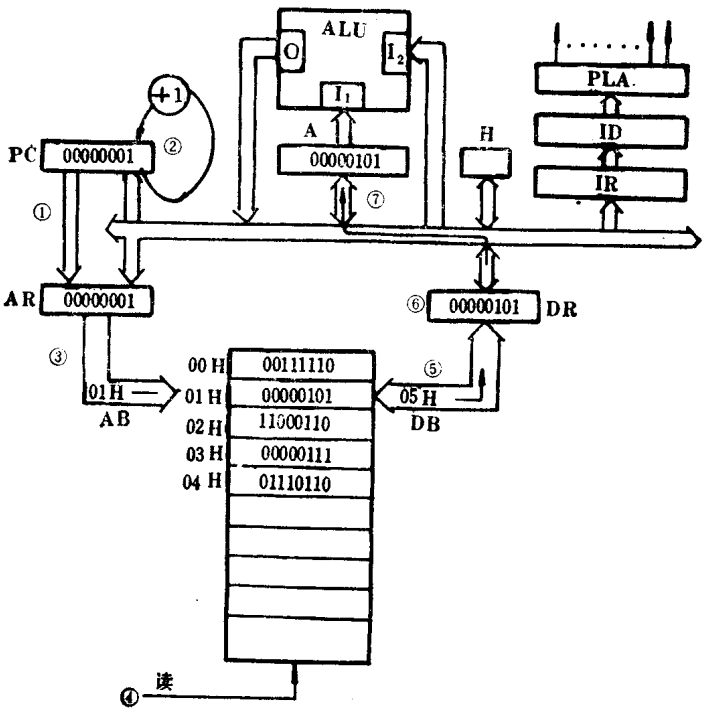

图1-8 执行第一条指令的操作示意图

1-8，步骤为：

（1）PC的内容01H送地址寄存器AR。

（2）当PC的内容已经可靠地送至AR后，PC的内容自动加1，变为02H。

（3）AR通过地址总线AB，把地址01H送至存贮器，经译码后，找到相应的内存单元。

（4）CPU发出“读”命令。

（5）选中的存贮单元的内容05H读出至数据总线DB。

（6）读出的内容05H通过数据总线送至数据寄存器DR。

（7）因已知读出的是操作数，且指令要求把它送至累加器A，故数据由DR通过内部数据总线送至累加器A。

3. 第一条指令执行完毕，进入第二条指令的取指阶段。见图1-9，步骤为：

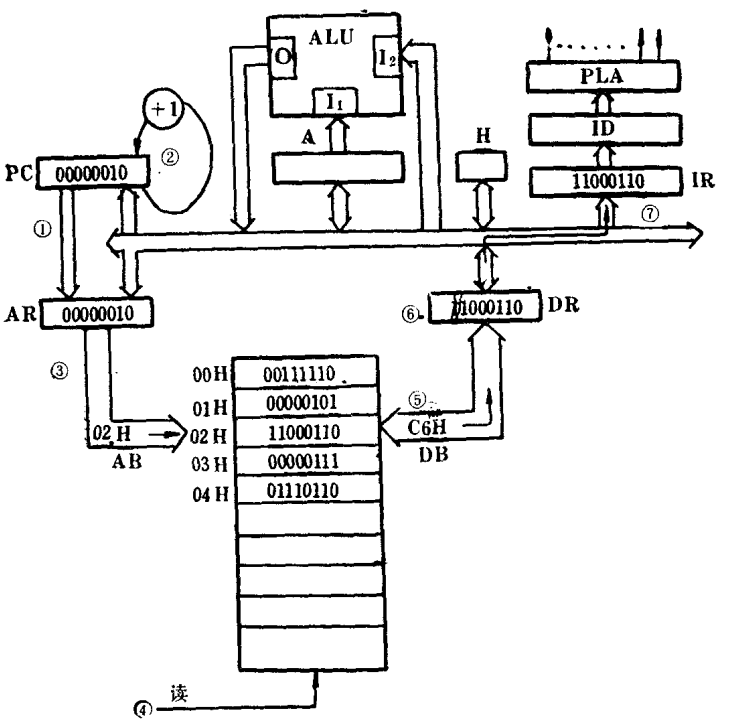

图1-9 取第二条指令操作示意图

（1）PC的内容02H送至地址寄存器AR。

（2）当PC的内容已经可靠地送至AR后，PC的内容自动加1，变为03H。

（3）AR通过地址总线AB，把地址02H送至存贮器，经译码后，找到相应的内存单元。

（4）CPU发出“读”命令。

（5）选中的存贮单元的内容C6H读出至数据总线DB。

（6）读出的内容C6H通过数据总线送至数据寄存器DR。

（7）因是取指阶段，读出的是指令，故DR把它送至指令寄存器IR，经过指令译码器ID译码后，发出执行指令的各种控制信息。

4. 转入执行第二条指令的阶段。经过对指令的译码知道，这是一条加法指令，以累加器A中的内容为一个操作数，另一个操作数在指令的第二字节中，所以，执行这条指令，

就是取出指令的第二字节，与A中的内容相加。见图1-10，步骤为：

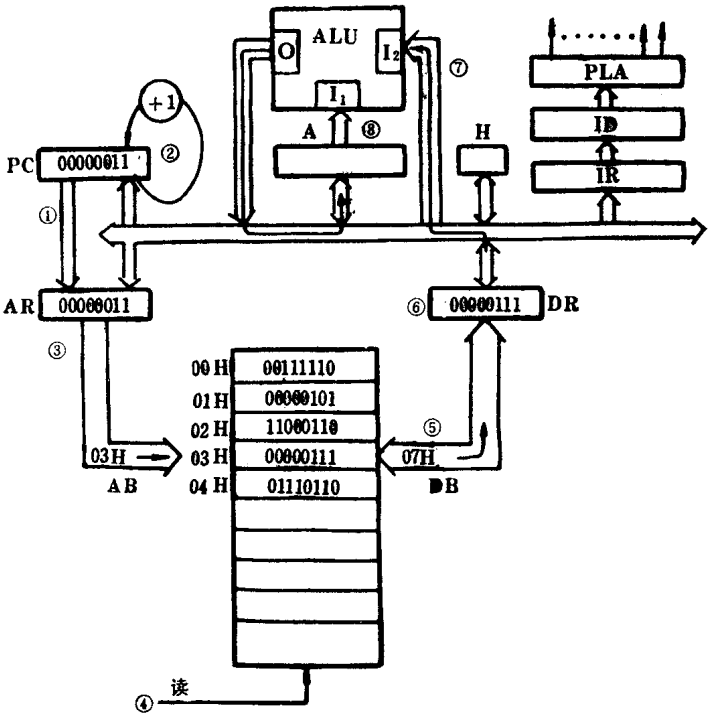

图1-10 执行第二条指令的操作示意图

(1)PC 的内容03H送至地址寄存器 AR。

(2)当PC的内容已经可靠地送至AR后，PC的内容自动加1，变为04H。

(3)AR 通过地址总线 AB，把地址03H送至存贮器，经译码后，找到相应的内存单元。

(4)CPU发出“读”命令。

(5)选中的存贮单元的内容07H读出至数据总线DB。

(6)读出的内容07H通过数据总线送至数据寄存器DR。

(7)因已知读出的是操作数，且要与累加器A中的内容相加，故数据由 DR 通过内部数据总线送至ALU的一个输入端。

(8)A中的内容送ALU的另一输入端，并执行加法操作。

(9)相加的结果由ALU输出至累加器A中。

5. 第二条指令执行完毕，进入第三条指令的取指和执行阶段。步骤与第一、二条指令的取指和执行阶段类似：

(1)PC 的内容 04H送至地址寄存器 AR 。

(2)当PC的内容已经可靠地送至AR后，PC的内容自动加1，变为05H。

(3)AR通过地址总线AB，把地址04H送至存贮器，经译码后，找到相应的内存单元。

(4)CPU发出“读”命令。

(5)选中的存贮单元的内容76H读出至数据总线DB。

(6)读出的内容76H通过数据总线送至数据寄存器DR。

(7)因是取指阶段，读出的是指令，故DR把它送至指令寄存器IR，经过指令译码器ID译码后，停机。

(五)Z80程序设计模型

对于程序设计者来说，了解程序设计模型是十分必要的。Z80程序设计模型是进行Z80汇编语言程序设计必须要具备的基础知识。Z80程序设计模型包括计算机内部的寄存器阵列和存贮器，只要了解了寄存器的设置和功能，就能够讨论指令系统、程序设计方法和程序运行情况，为此，我们介绍Z80的寄存器阵列。

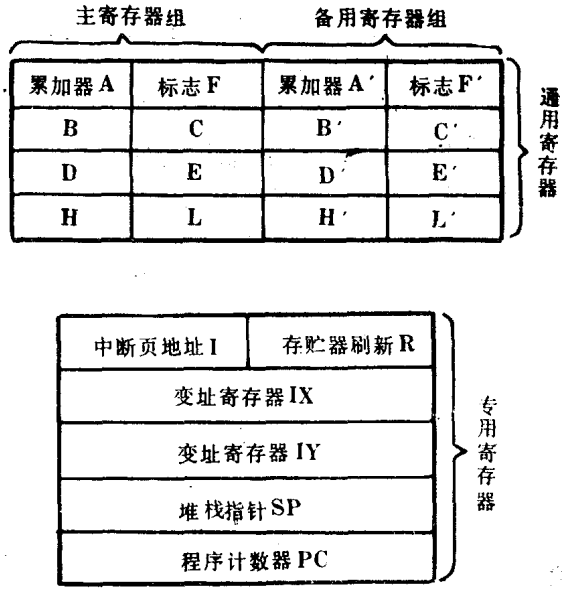

图1-11 Z80-CPU内部寄存器

见图1-11，Z80的寄存器阵列包括通用寄存器和专用寄存器。通用寄存器包括：

1. 累加器A。在8位单操作数或双操作数指令中，累加器A作为主要源或目标寄存器。在算术和逻辑操作指令中，累加器A中的内容必为一个操作数，且操作的结果放在累加器A中。

2. 状态标志寄存器F。算术和逻辑操作结果的一些特征寄存在F中。D_5和D_3两位不用，其余6个标志位的意义如下：

F寄存器

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S	Z	/	H	/	P/V	N	C

S：符号标志。运算结果的$D_7=1$，即符号为负，则S＝1；反之，$D_7=0$，即符号为正，则S＝0。

Z：零标志。运算结果为0，则Z＝1；反之，运算结果不为0，则Z＝0。

H：半进位标志。在进行（加法或减法运算中，若从D_3产生一个进位或从D_4产生一个借位，则H＝1；反之，即不从D_3产生进位或不从D_4产生借位，则H＝0。

C：进位标志。在进行无符号数运算时，若做加法D_7有进位，或做减法D_7有借位，则C＝1；反之，即加法D_7无进位，而减法D_7无借位，则C＝0。移位操作时，C存放移入的数码。

N：加、减标志。执行的操作为减法，则N＝1；执行的操作为加法，则N＝0。

P/V：奇偶或溢出标志。若逻辑操作的结果中1的个数为偶数，则P＝1；1的个数为奇数，则P＝0。若算术操作的结果有溢出，则V＝1；结果无溢出，则V＝0。

关于F寄存器，在第二章数的运算方法中再进一步讨论。

3. 6个8位寄存器B、C、D、E、H、L。它们也可连成3个16位的寄存器对BC、DE、HL。

以上的寄存器合在一起构成了主寄存器组，在通用寄存器中，除了主寄存器组外，还有一组备用寄存器，在工作时，只有一组寄存器参与操作，可以用一简单的交换指令调换另一组寄存器工作。

4. 专用寄存器

(1)中断页地址寄存器I：8位。在Z80中断方式2中，I寄存器中的内容，是中断服务程序入口地址表的页面地址。

(2)存贮器刷新寄存器R：为防止MOS动态存贮器的信息泄漏，要定期对存贮器刷新，每一次刷新内存中的一行，每次刷新的地址由R寄存器内容提供。

(3)程序计数器PC：PC中的内容是下一条要执行的指令的16位地址。

(4)堆栈指针SP：堆栈是在存贮器中的一个按照先进后出原则组织的存贮区域，SP的内容是16位地址，始终指向堆栈的顶部。

堆栈的作用以后再讨论。

(5)变址寄存器IX和IY：各自包含着一个16位的基地址，由它加上指令中给定的偏移量形成操作数的有效地址。

(六)微机主要技术指标

这是评价微型计算机技术性能的重要依据。

1. 字长：指二进制位数的长短，字长标志着计算机的精度，微机字长一般1～32位，8位机为多。

2. 内存容量：以字节或字长为单位计算，微型计算机的内存容量一般为4K～64K字节（每1024字节称为1K字节）。

3. 存取周期：前面已经介绍过存贮器的读操作和写操作。完成一次读与写操作所需的全部时间称为存取周期。微型机常用的MOS半导体存贮器的存取周期约为几十ns～1μs。

4. 运算速度：用每秒所能执行的指令条数表示，微型机通常以给出机器的主频及执行每条指令所需的周期数或执行指令的平均时间来表示。Z80主频为2.5MHz，Z80A主频为4MHz，执行指令的平均时间为1.6μs，执行每条指令所需的周期数见Z80指令表。

5. 外部设备的配置能力。

6. 软件的配备状况。

二、自 学 重 点

微机的工作过程是本章的自学重点，目的在于使学生建立起初步的整机概念。前面已通过具体例子进行了讨论。为了加深对这一问题的认识，我们再举一个稍微复杂的例子说明：

仍是计算两个数5和7相加的和，但数5已存在存贮器中，要求相加的和也存在存贮器中，则程序如下：

```
LD      A,(M1)      ；将数5送累加器A
ADD     A,7         ；将累加器A中的内容5和数7相加，结果；仍放A中
LD      (M2),A      ；相加的和存入存贮器
HALT                ；停止操作
```

若数据放在程序之后，则程序和数据在存贮器中的存放示意图如图1-12所示。

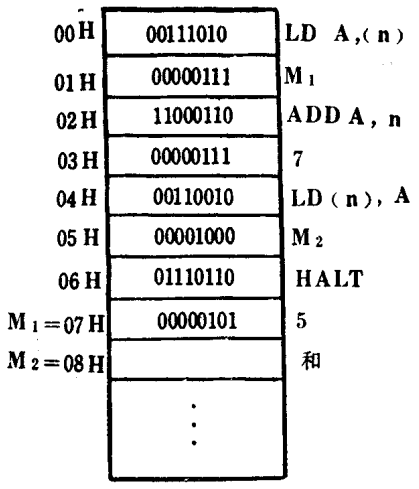

图1-12 程序存贮示意图

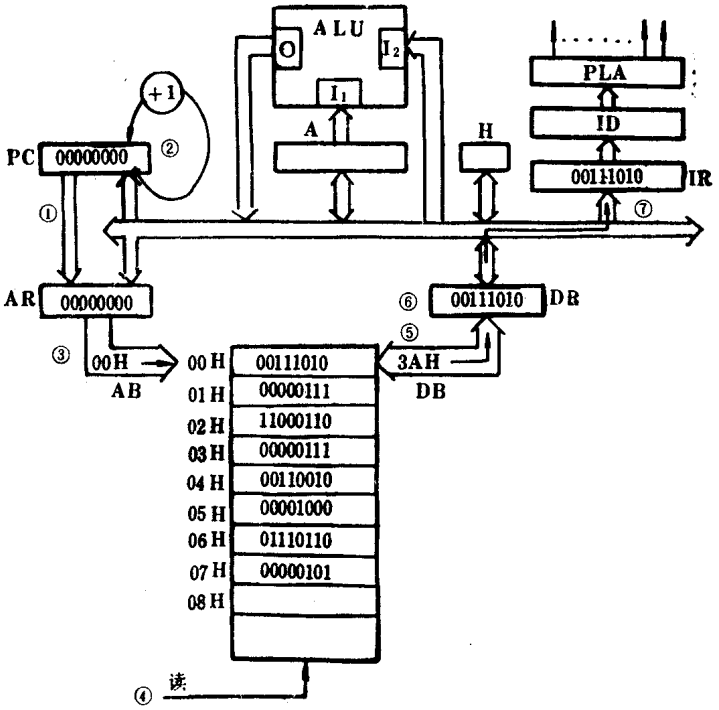

图1-13 取第一条指令的操作示意图

首先把第一条指令所在的地址 00 H赋给程序计数器PC，然后就进入第一条指令的取指阶段。见图1-13，步骤为：

（1）PC的内容00H送至地址寄存器AR。

（2）当PC的内容已经可靠地送至AR后，PC的内容自动加1，变为01H。

（3）AR 通过地址总线 AB，把地址 00 H送至存贮器，经译码后，找到相应的内存单元。

（4）CPU发出“读”命令。

（5）选中的存贮单元的内容3AH读出至数据总线DB。

（6）读出的内容3AH通过数据总线送至数据寄存器DR。

（7）因是取指阶段，读出的是指令，故DR把它送至指令寄存器 IR，经过指令译码器ID译码后，发出执行指令的各种控制信息。

接着，转入执行第一条指令的阶段。这个阶段又分两步：第一步要把操作数的地址从指令的第二字节中取出来，第二步从这个地址取出操作数送累加器A。取操作数地址的过程如图1-14所示，步骤为：

（1）PC的内容01H送地址寄存器AR。

（2）当PC的内容已经可靠地送至AR后，PC的内容自动加1，变为02H。

（3）AR 通过地址总线 AB，把地址 01H 送至存贮器，经译码后，找到相应的内存单元。

（4）CPU发出“读”命令。

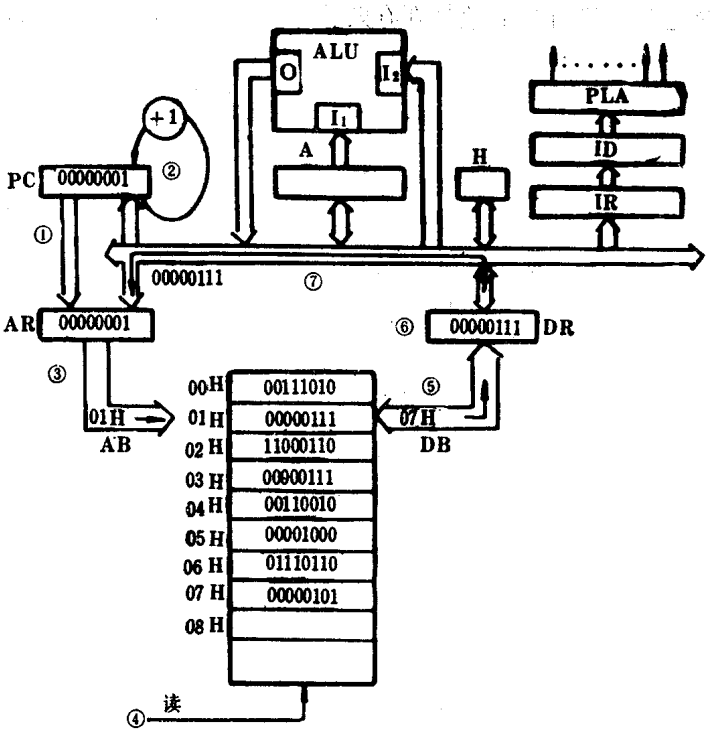

图1-14　取操作数地址的操作示意图

（5）选中的存贮单元的内容07H读出至数据总线DB。

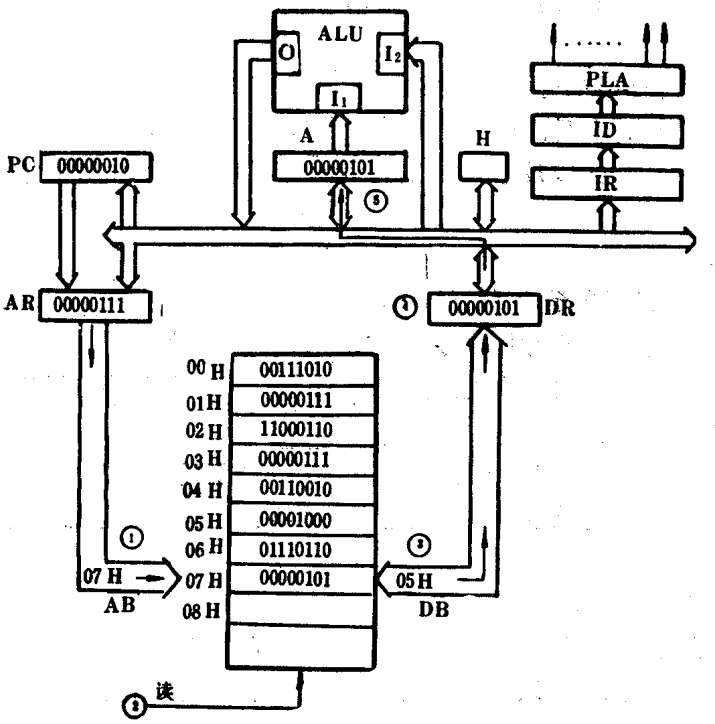

图1-15　取操作数的操作示意图

（6）读出的内容07H通过数据总线送至数据寄存器DR。

（7）由于读出的是操作数的地址，故由DR经内部数据总线送至AR。

然后进入执行指令的第二步，如图1-15所示，步骤为：

（1）AR通过地址总线把地址07H送至存贮器，经译码后，找到相应的内存单元。

（2）CPU发出“读”命令。

（3）选中的存贮单元的内容05H读出至数据总线DB。

（4）读出的内容05H通过数据总线送至数据寄存器DR。

(5)05H由DR经内部数据总线送至累加器A。

至此，第一条指令执行完毕，数据05H被放到累加器A中。

第二条指令的取指阶段取出加法指令。执行阶段取出操作数07H，送ALU的输入端I，操作数07H和累加器A中的05H相加，和数0CH放到累加器A中。也就是说，第二条指令执行完毕时，累加器A中为和数0CH。

第三条指令的取指阶段取出送数指令。执行阶段分为两步：第一步确定地址08H，第二步将累加器A中的和数0CH写入08H内存单元。

第四条指令取出的是机器操作暂停指令，执行这条指令，机器停止全部操作。

这里要说明两点：

(1)程序可以存放在内存的其它位置，也就是说，PC的初值不一定是00H，放在哪里依要求而定。同样地，数据也可以放在内存的其它位置，即数据不一定紧挨着程序存放，放在哪里也依要求而定。

(2)在我们的例题中，内存单元地址是单字节的，但实际的Z80机器，内存单元地址是双字节的，因而上述同样的加法程序在内存中的存放情况不同，如图1-16所示。程序的执行过程显然也要复杂一些，这里不赘述。

通过对以上简单程序执行过程的分析，了解了微机的大概工作过程，初步建立了整机的概念。

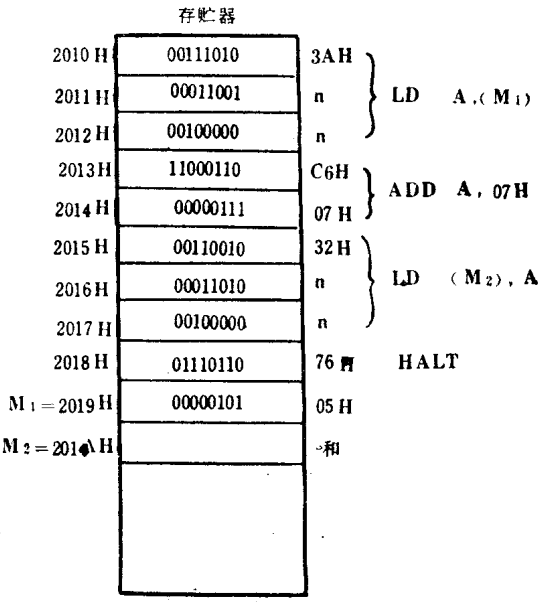

图1-16 程序存贮示意图

三、复 习 题

1. 微机系统由哪两大部分组成？硬件系统结构是怎样的？
2. 举例说明微机的工作过程。
3. Z80的程序设计模型是怎样的？寄存器阵列中包括哪些寄存器？各自功能如何？

第二章　微机运算基础

一、内　容　概　要

(一)计数方法

1．什么是进位计数制？什么是权和基数？

按进位的方法进行计数，称为进位计数制，简称进位制。十进制就是经常用到的一种进位制。

十进制的特点是有十个不同的数字符号，即0、1、2、3、4、5、6、7、8、9；且逢十进位。例如十进制数238，右数第一位8代表个位，它的值就是8，右数第二位3代表十位，它的值是3×10＝30，右数第三位2代表百位，它的值是2×100＝200，这“个、十、百、……”在数学上叫“权”，每一位上的数码与该位权的乘积表示该位数值的大小。“10”称为十进制的“底数”或“基数”。

与十进制类似，二进制数的特点是：有两个不同的数字符号，即0、1；且逢二进位。例如二进制数1111，右数第一位的值是1×1＝1，右数第二位的值是1×2＝2，右数第三位的值是1×4＝4，右数第四位的值是1×8＝8，这“1、2、4、8、……”叫二进制各位的“权”，“2”称为二进制的“基数”。

八进制的特点是：有八个不同的数字符号，即0、1、2、3、4、5、6、7；且逢八进位。“1、8、64、512、……”是八进制各位的“权”，“8”是八进制的“基数”。

十六进制的特点是：有十六个不同的数字符号，即0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F；且逢十六进位。“1、16、256、……”是十六进制各位的“权”，“16”是十六进制的“基数”。

对于任意进位计数制，基数可用正整数R来表示。这时，数N可表示为

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

式中m、n均为正整数。

(1)R是“基数”，表示R进制，逢R进1；

(2)R^i是第i位的“位权”，

(3)K_i是0至(R－1)中的任意一个；

(4)K_iR^i是第i位数值。

在计算机中，常用的是二进制、八进制、十六进制，其中二进制用得最多。

2．不同进位制间的转换

(1)二进制数转换为十进制数

方法很简单，只需将二进制数每一位的数码与该位的权相乘，便得到该位的数值，再将各位的数值加在一起就得到了相应的十进制数。例如：