1. **概论**

一 汇编基本概念

1. 汇编语言
   1. 汇编语言的概念

汇编语言（Assembly Language）是任何一种用于电子计算机、微处理器、微控制器或其他可编程器件的低级语言，亦称为符号语言。通俗来讲，汇编语言是直接在硬件之上工作的编程语言。

* 1. 汇编语言的产生

早期的程序员们很快就发现了使用机器语言带来的麻烦，它是如此难于辨别和记忆，给整个产业的发展带来了障碍。于是汇编语言产生了。

汇编语言的主体是汇编指令。汇编指令和机器指令的差别在于指令的表示方法上。汇编指令是机器指令便于记忆的书写格式。

例如∶机器指令100010011011000表示把寄存器BX的内容送到AX中。汇编指令则写成mov ax，bx。这样的写法与人类语言接近，便于阅读和记忆。

操作∶寄存器BX的内容送到AX中

机器指令∶1000100111011000

汇编指令∶mov ax，bx

（寄存器，简单地讲是CPU中可以存储数据的器件，一个CPU中有多个寄存器。AX 是其中一个寄存器的代号，BX 是另一个寄存器的代号。更详细的内容我们在以后的课程中将会讲到。）

此后，程序员们就用汇编指令编写源程序。可是，计算机能读懂的只有机器指令，那么如何让计算机执行程序员用汇编指令编写的程序呢?这时，就需要有一个能够将汇编指令转换成机器指令的翻译程序，这样的程序我们称其为编译器。程序员用汇编语言写出源程序，再用汇编编译器将其编译为机器码，由计算机最终执行。图1.1描述了这个工作过程。

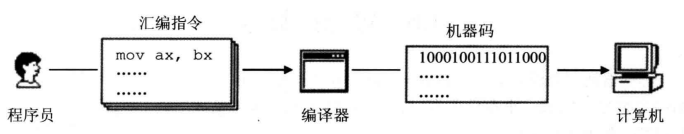


图1.1 用汇编语言编写程序的工作过程

* 1. 汇编语言的组成

汇编语言发展至今，有以下3类指令组成。

（1）汇编指令∶机器码的助记符，有对应的机器码。

（2）伪指令∶没有对应的机器码，由编译器执行，计算机并不执行。

（3）其他符号∶如+、一、\*、/等，由编译器识别，没有对应的机器码。

汇编语言的核心是汇编指令，它决定了汇编语言的特性。

* 1. 汇编语言的特点

汇编语言是面向机器的低级语言，通常是为特定的计算机或计算机系列专门设计的。

汇编语言保持了机器语言的优点，具有直接和简捷的特点。

汇编语言可有效地访问、控制计算机的各种硬件设备,如磁盘、存储器、CPU、I/O端口等。

汇编语言目标代码简短，占用内存少，执行速度快，是高效的程序设计语言。

汇编语言经常与高级语言配合使用，应用十分广泛。

（添加知识点讲解的ppt）

1. 机器语言
   1. 机器语言的概念

说到汇编语言，也要讲一下机器语言。机器语言是机器指令的集合。机器指令展开来讲就是一台机器可以正确执行的命令。电子计算机的机器指令是一列二进制数字。计算机将之转变为一列高低电平，以使计算机的电子器件受到驱动，进行运算。

上面所说的计算机指的是可以执行机器指令，进行运算的机器。这是早期计算机的概念。现在，在我们常用的PC机中，有一个芯片来完成上面所说的计算机的功能。这个芯片就是我们常说的CPU（Central Processing Unit，中央处理单元），CPU是一种微处理器。以后我们提到的计算机是指由CPU和其他受CPU直接或间接控制的芯片、器件、设备组成的计算机系统，比如我们最常见的PC机。

每一种微处理器，由于硬件设计和内部结构的不同，就需要用不同的电平脉冲来控制，使它工作。所以每一种微处理器都有自己的机器指令集，也就是机器语言。

早期的程序设计均使用机器语言。程序员们将用0、1数字编成的程序代码打在纸带或卡片上，1打孔，0不打孔，再将程序通过纸带机或卡片机输入计算机，进行运算。

* 1. 机器语言与汇编语言的区别

机器语言是机器能直接识别的程序语言或指令代码，是用二进制代码表示的语言；而汇编语言是面向机器的程序设计语言，用易于理解和记忆的名称和符号来表示机器指令中的操作码。他们的差别在于指令的表示方法上，汇编指令是机器指令便于记忆的书写格式。例如：机器指令1000100111011000表示把寄存器BX的内容送到AX中，汇编指令则写成mov ax,bx。这样的写法与人类语言接近，便于阅读和记忆。

（添加知识点讲解的ppt，和上一小节汇编语言的ppt可以合并成一个）

二 前置基础知识

1. 进制数
   1. 进制数的表示

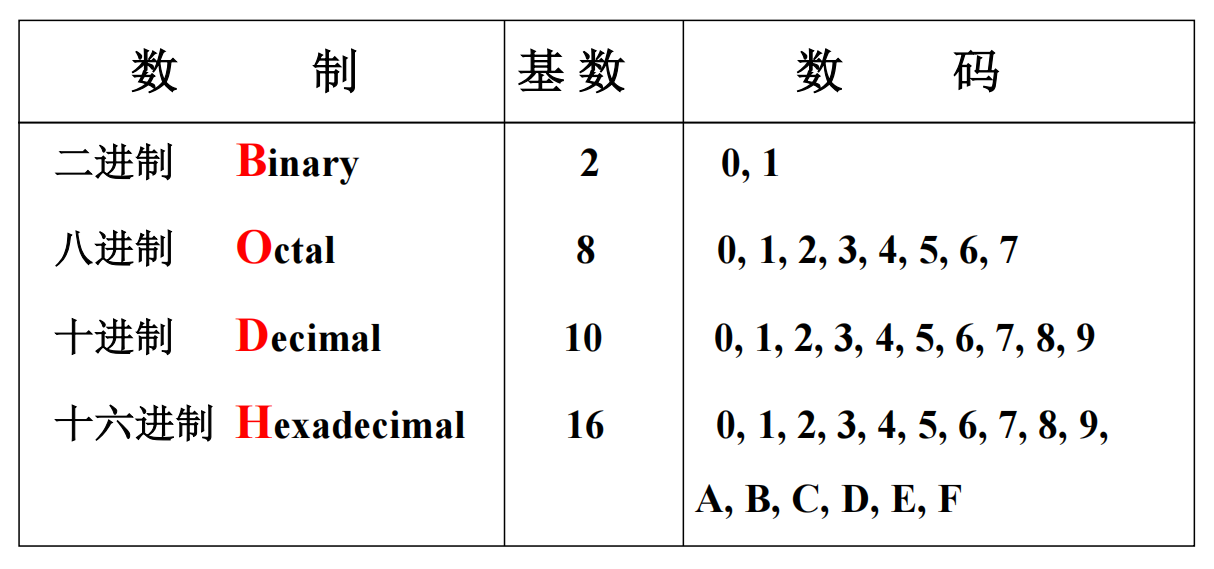
进位计数制是一种计数的方法,习惯上最常用的是十进制计数法。十进制数是人们最熟悉、最常用的一种数制，但它不是唯一的数制。例如，计时用得时、分、秒就是按60进制计数的。

计算机中为便于存储及计算的物理实现，采用了二进制数。二进制数的基数为2，只有0和1两个数码，并遵循逢二进一的规则，它的各位权是以表示的，因此二进制数的值是：

其中,为0,1两个数码中的一个。例如：

其中数的下角标表示该数的基数r，即二进制的101101与十进制的45等值。

为便于人们阅读与书写，经常使用八进制数或十六进制数来表示二进制数。它们的基数和数码如下图所示。



计算机中存储信息的基本单位为一个二进制位（Bit），它可以用来表示0和1两个数码。此外，由于计算机中常用的字符是采用由8位二进制数组成的一个字节（Byte）来表示的，因此字节也成为计算机中存储信息的单位。计算机的字长一般都选为字节的整数倍，如16位、32位。64位等。一个字节由8位组成，它可以用两个四位组（又称半字节）来表示，所以用十六进制数来表示二进制数是比较方便的。

十六进制数的基数为16，共有16个数码，它们是0，1，2，3，4，5，6，7，8，9，A，B，C，D，E，F，其中A表示十进制的10，余类推。它们与二进制和十进制数的对应关系如下：

|  |  |  |
| --- | --- | --- |
| 二进制数 | 十进制数 | 十六进制数 |
| 0000 | 0 | 0 |
| 0001 | 1 | 1 |
| 0010 | 2 | 2 |
| 0011 | 3 | 3 |
| 0100 | 4 | 4 |
| 0101 | 5 | 5 |
| 0110 | 6 | 6 |
| 0111 | 7 | 7 |
| 1000 | 8 | 8 |
| 1001 | 9 | 9 |
| 1010 | 10 | A |
| 1011 | 11 | B |
| 1100 | 12 | C |
| 1101 | 13 | D |
| 1110 | 14 | E |
| 1111 | 15 | F |

* 1. 不同进制数的转换
     1. 二进制数与十进制数之间的转换

各位二进制数码乘以与其对应的权之和即为与该二进制数相对应的十进制数。例如：

而十进制数转换为二进制数的方法很多，这里只说明比较简单的降幂法及除法两种。

1. 降幂法

首先写出要转换的十进制数，其次写出所有小于此数的各位二进制权值，然后用要转换的十进制数减去与它最相近的二进制权值，如够减用减去并在相应位记以1；如不够减则在相应位记以0并跳过此位；如此不断反复，直到该数为0为止。

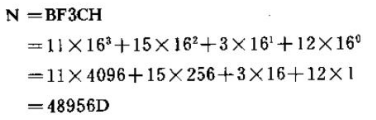
1. 除法

把要转换的十进制数的整数部分不断除以2，并记下余数，直到商为0为止。

而对于被转换的十进制数的小数部分则应不断乘以2，并记下其整数部分，直到结果小数部分为0为止。

* + 1. 十进制数与十六进制数之间的转换

各位十六进制数与其对应权值的乘积之和即为与此十六进制数相对应的十进制数。例如：



十进制数转换为十六进制数也可使用降幂法和除法。

降幂法：首先写出要转换的十进制数，其次写出小于该数的十六进制权值，然后找出该数中包含多少个最接近它的权值的倍数，这一倍数即对应位的值，用原数减去此倍数与相应位权值的乘积得到一个差值，再用此差值去找低一位的权值的倍数，如此反复直到差值为0为止。

除法：把要转换的十进制数的整数部分不断除以16，并记下余数，直到商为0为止。对于要转换的十进制数的小数部分，则应不断乘以16，并记下其整数部分，直到结果的小数部分为0为止。

* + 1. 二进制数与十六进制数之间的转换

由于十六进制数的基数是2的幂，所以这两种数制之间的转换是十分容易的。一个二进制数，只要把它从低位到高位每4位组成一组，直接用十六进制数来表示就可以了。例如：

0011 0101 1011 1111

3 5 B 0

亦即001101011011111B=35BFH

反之，把十六进制数中的每一位用4位二进制数表示，就形成相应的二进制数了。

A 1 9 C

1010 0001 1001 1100

亦即A19CH=1010000110011100

* 1. 进制数的运算
     1. 二进制数的运算

加法规则： 乘法规则：

0+0=0 0 X 0=0

0+1=1 0 X 1=0

1+0=1 1 X 0=0

1+1=0 (进位1) 1 X 1=1

* + 1. 十六进制数的运算

十六进制数的运算可以采用先把该十六进制数转换为十进制数，经过计算后再把结果转换为十六进制数的方法，但这样做比较繁琐。其实，只要按照逢十六进一的规则，直接用十六进制数来计算也是很方便的。

十六进制加法：当两个一位数之和S小于16时，与十进制数同样处理；如果两个一位数之和大于或等于16时，则应该用S-16及进位1来取代S。

十六进制减法：与十进制数类似，够减时可直接相减，不够减时服从向高位借1为16的规则。

十六进制乘法：可以用十进制数的乘法规则来计算，但结果必须用十六进制数来表示。

十六进制除法：可以根据其乘法和减法的规则处理，需要时读者可自行处理，这里不再赘述。

（添加对进制数相关知识进行讲解的ppt，并且利用视频演示进制数的转换和运算）

1. 数和字符的表示
   1. 数的表示
      1. 数的补码表示

计算机中的数是用二进制来表示的，数的符号也是用二进制表示的，把一个数连同其符号在内在机器中的表示加以数值化，这样的数称为机器数）一般用最高有效值来表示数的符号，正数用0表示，负数用1表示。机器数可以用不同的码制来表示，常用的有原码、补码和反码表示法。由于多数机器的整数采用补码表示法，80x86机也是这样，所以我们在这里只介绍补码表示法。

补码表示法中，正数采用符号-绝对值表示，即数的最高有效位为0表示符号为正，数的其余部分则表示数的绝对值。例如，假设机器字长为8位，则[＋1] =00000001，[＋127] =01111111，[+0] =000000000。

当用补码表示法来表示负数时则要麻烦一些。负数X用来表示，其中n为机器的字长。当n=8时，[-1]补 ==11111111，而[-127]补 = =10000001，显然，最高有效位为1表示该数的符号为负。应该注意，[-0]补 = =00000000，所以在补码表示法中0只有一种表示，即00000000。对于10000000这个数，在补码表示法中被定义为-128。这样，8位补码能表示数的范围为-128～＋127。

我们可以用一种比较简单的办法来写出一个负数的补码表示：先写出与该负数相对应的正数的补码表示（用符号-绝对值法），然后将其按位求反（即0变为1，1变为0），最后在末位（最低位）加1，就可以得到该负数的补码表示了。

* + 1. 数的补码运算

我们知道，对一个正数的补码表示按位求反后再在末位加1，可以得到与此正数相应的负数的补码表示。我们把这种对一个二进制数按位求反后在末位加1的运算称为求补运算，可以证明补码表示的数具有以下特性：



在这里，只用例子来说明。由下例可见：

[117]补 =0075H

[-117]补 =FF8BH

现对 [-117]补作求补运算：

[-117]补为 1111 1111 1000 1011

按位求反后得 0000 0000 0111 0100

末位加1后得 0000 0000 0111 0101

此数正是[＋117]补=0075H。

这一特性在补码的加、减法运算中很有用。

补码的加法规则是：

[X+Y]补 =[X]补 +[Y]补

补码的减法规则是：

[X-Y]补 =[X]补 +[-Y]补

其中的[-Y]补只要对[Y]补求补就可得到。对于这两个规则我们只用例子来说明。读者可以从下面的例子中认识到由于用补码表示数，使计算机中的加、减法运算十分简便，它不必判断数的正负，只要符号位参加运算，便能自动地得到正确的结果。

（添加对数的表示的知识点进行讲解的ppt，并且用视频演示补码的运算）

* 1. 字符的表示

计算机中处理的信息并不全是数，有时需要处理字符或字符串，例如从键盘输入的信息或打印输出的信息都是字符方式输入输出的，因此，计算机必须能表示字符。字符包括：

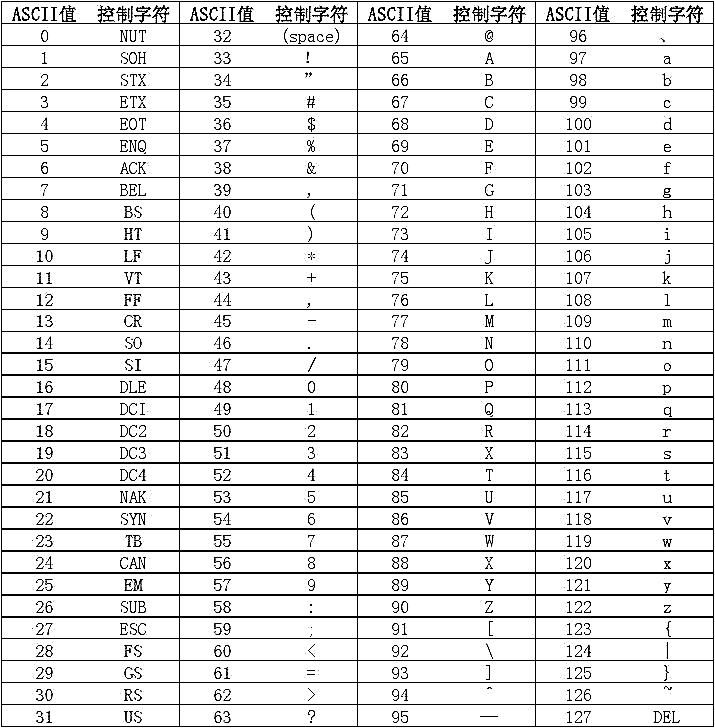
字母A、B、…、Z，a、b、…、Z；

数字∶0、1、…、9；

专用字符∶＋、-、\*、/、半、SP（space空格）····

非打印字符∶BEL（Bell响铃）、LF（Line Feed换行），CR（Carriage Return回车）、……

这些字符在机器里必须用二进制数来表示。IBM PC机采用目前最常用的美国信息交换标准代码ASCII（American Standard Code for Information Interchange）来表示。这种代码用一个字节（8位二进制码）来表示一个字符，其中低7位为字符的ASCII值，最高位一般用作校验位。下表列出了用十六进制数表示的部分常用字符的ASCII值。



（用ppt讲解字符的表示相关知识，可以和数的表示合并为一个ppt）

1. 基本逻辑运算
   1. “与”运算（AND）

“与运算”又称逻辑运算，可用符号•或∧来表示。如有A、B两个逻辑变量（每个变量只能有0或1两种取值），可能有的取值情况只有4种，在各种取值的条件下得到的“与”运算结果如下表所示：

|  |  |  |
| --- | --- | --- |
| *A* | *B* |  |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

* 1. “或”运算（OR）

“或”运算又称逻辑加，可用符号+或∨来表示。“或”运算规则可用下表表示：

|  |  |  |
| --- | --- | --- |
| *A* | *B* |  |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

* 1. “非”运算（NOT）

如变量为A，则它的“非”运算的结果用表示。“非”运算规则如下：

|  |  |  |
| --- | --- | --- |
| *A* | *B* |  |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

* 1. “异或”运算（XOR Exclusive-OR）

“异或”运算可以用符号来表示。其运算规则如下表所示：即当两个变量相异时，它们的运算结果为1。

|  |  |  |
| --- | --- | --- |
| *A* | *B* |  |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

（用ppt讲解基本逻辑运算相关知识）

**习题**：

1 对于不同数制之间关系的描述，正确的描述为 A 。

A任意的二进制有限小数，必定也是十进制有限小数。

B任意的八进制有限小数，未必也是二进制有限小数。

C任意的十六进制有限小数，不一定是十进制有限小数。

D任意的十进制有限小数，必然也是八进制有限小数。

2十六进制数FFF.CH相当十进制数 D 。

A 4096.3 B 4096.25 C 4096.75 D 4095.75

3 （2004）10+（32）16的结果是 D

A （2036）10 B （2054）16 C（4006）10 D（100000000110）2

4 十进制29的原码是 C

A 11100010 B 10101111 C 00011101 D 00001111

5 在计算机内部，信息的存储和处理都采用二进制，最主要的原因是 D

A 便于存储 B 数据输入方便 C 可以增加计算机存储容量 D 易于用电子元件实现

6 +127(10)的八位基二码补码是 A

A 01111111B B 11111110B C 10000000B D 00000010B

7 简述汇编语言的优缺点

优点 1 可有效的访问控制计算机各种硬件设备，如磁盘、存储器、CPU、I/0端口等。

2 目标代码简短，占用内存少，执行速度快，是高效的程序设计语言。

3 可与高级语言配合使用，应用广泛。

缺点 1 通用性和可移植性较差。

2 相对于高级语言来说较为繁琐、易出错、不够直观。

8 指令中的逻辑地址由段基址和偏移量组成。

9 已知[x]补=11111111，则x对应的真值为 -1 。