汇编语言程序设计（第3版）

杨宇博
软件工程系
QQ：376468511
（请备注班级姓名）

总学时：32学时(24理论＋8实验)

教材

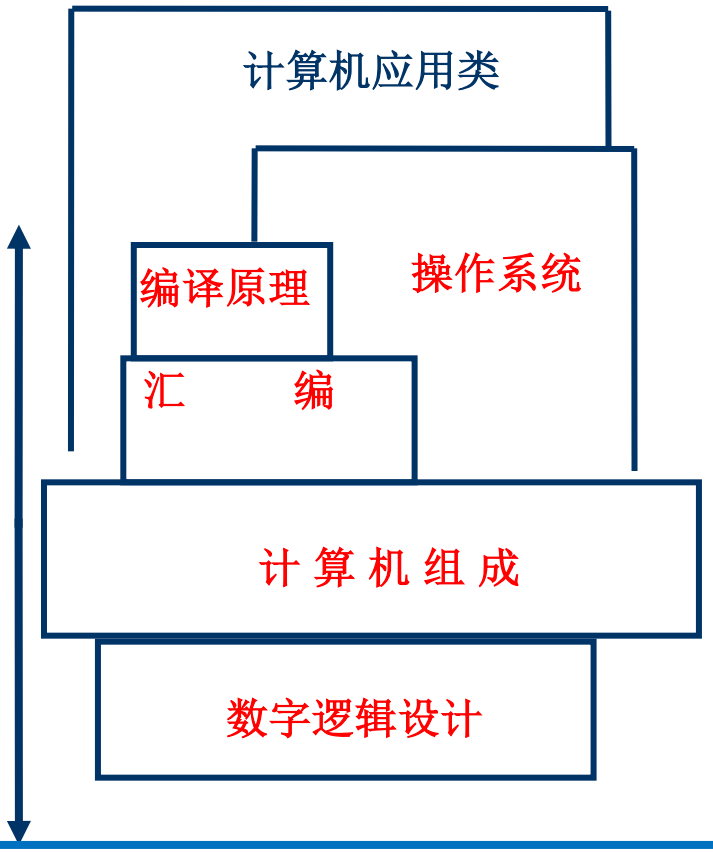
教材：

- 《汇编语言程序设计》（第3版），刘慧婷等，人民邮电出版社

课程定位

与计算机组成原理、编译原理、操作系统、数字逻辑设计等组成计算机系统类课程

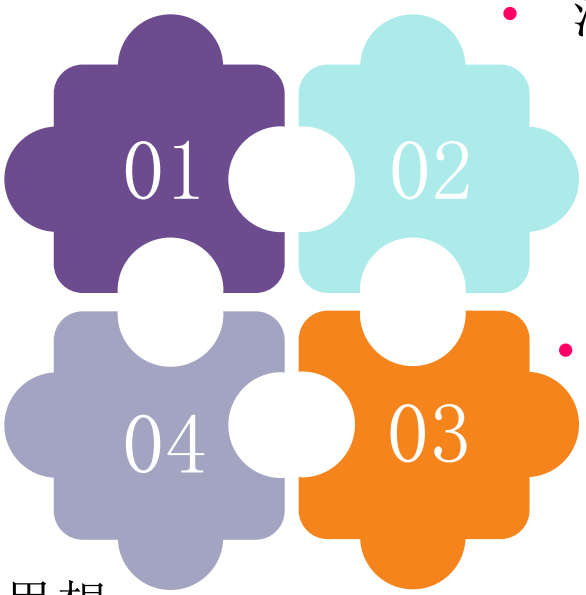
汇编语言程序设计与计算机组成原理作为软硬件界面起到“承上启下”的作用



为理解计算机系统打下指令集、汇编编程入门的基础。

课程特点： 内容抽象、概念性强、内容灵活

- 提前预习、认真听课、按时完成书面及上机作业
- 注意循序渐进：
  - ✓ 基本概念、基本思想、基本步骤、程序设计



- 注意先修课程的知识准备
  - ✓ 离散数学、C语言、数据结构
- 注意培养程序设计的能力
  - ✓ 理解所讲语法结构、对此多做思考：若问题要求不同，应如何设计有效的算法和程序。

课程考核

- 平时（作业+考勤+课堂表现） 30%
- 考试成绩 70%

- 期末作业未交或者3次考勤不到，无平时成绩！

- 学习通（1-4班）
- 所有人必入！关系考勤！

- QQ群（班长&学委加入）
- 进群暗号：汇编语言程序设计

- 学习通（5-8班）
- 所有人必入！关系考勤！

- QQ群（班长&学委加入）
- 进群暗号：汇编语言程序设计

- 学习通（1-2班）
- 所有人必入！关系考勤！

- QQ群（班长&学委加入）
- 进群暗号：汇编语言程序设计

目标
target

01 OPTION 认识汇编语言的意义

02 OPTION 掌握计算机中数据和字符的常用表示方法

03 OPTION 掌握补码的运算

第一章 汇编语言基础知识

- 1.1 汇编语言简介
- 1.2 计算机中数据的表示

1.1 汇编语言简介

- 1.1.1 机器语言与汇编语言
- 1.1.2 汇编语言的组成
- 1.1.3 为什么要学习汇编语言

1.1.1 机器语言、汇编语言、高级语言

- 机器指令：cpu能直接识别并遵照执行的指令，用二进制编码表示，由操作码，操作数组成，编码只含二进制0或1。

- 机器语言：用二进制编码组成的机器指令的集合和一组使用机器指令的规则。

- 汇编语言：对机器指令中的操作码用英文单词的缩写描述（助记符），对操作数用标号、变量、常量描述。

示例：

机器码：1010 0001 0000 0011 0000 0000

汇编指令：Mov AX, [3]

高级语言：面向人的语言，表达形式接近自然语言

优点：便于阅读，易学易用，不涉及硬件，具有通用性

缺点：目标代码冗长，占用内存多，从而执行时间长，
效率不高，不能对某些硬件进行操作

机器语言：依赖于机器的低级语言，书写格式为二进制代码

优点：执行速度快，效率高

缺点：表达的意义不直观，编写、阅读、调试较困难

汇编语言：是一种符号语言，与机器语言一一对应；它使用助记符表示相应的操作，并遵循一定的语法规则

优点：面向机器编程，在“时间”和“空间”上效率

缺点：涉及硬件细节，要求熟悉计算机系统的内部结构

- 用汇编语言编写的程序称为汇编源程序。

- 汇编语言是一种符号语言，比机器语言容易理解和掌握，也容易调试和维护。但是，汇编语言源程序要翻译成机器语言程序才可以由计算机执行。这个翻译的过程称为“汇编”，这种把汇编源程序翻译成目标程序的语言加工程序称为汇编程序。

1.1.2 汇编语言的组成

- 汇编语言有以下三类指令组成：
- 汇编指令：机器码的助记符，有对应的机器码。它是汇编语言的核心。
- 伪指令：没有对应的机器码，由编译器执行，计算机并不执行。
- 其他符号：如+、-、*、/等，由编译器识别，没有对应的机器码。

1.1.3 为什么要学习汇编语言

- 汇编语言程序是用符号指令写成的，本质上还是机器语言，与具体机机型的硬件密切相关，可以直接有效地控制计算机硬件，程序运行速度快，程序短小精悍，占用内存少，在某些特殊应用场合更能发挥作用。如：智能化仪表，家用电器，实时控制系统，嵌入式系统，单片机控制，病毒研究，硬件驱动程序等。

- 学习汇编语言是从根本上认识和理解计算机工作过程的最好方法。

1.2 计算机中数据的表示

- 1.2.1 不同进位计数制及其相互转换
- 1.2.2 二进制数和十六进制数的运算
- 1.2.3 带符号数的表示
- 1.2.4 补码的加法和减法
- 1.2.5 无符号数的表示
- 1.2.6 字符的表示
- 1.2.7 基本逻辑运算

1.2.1 不同进位计数制及其相互转换

1. 进位计数制

对于任意一个进位计数制，如果用R表示基数，那么任何一个数S均可用如下多项式表示：

$$S= k_nR^n+k_{n-1}R^{n-1}+\ldots+k_0R^0+k_{-1}R^{-1}+k_{-2}R^{-2}+\ldots+k_{-m}R^{-m}$$

- 十进制数:
  423.5=4×10^2+2×10^1+3×10^0+5×10^-1
  各位权值 10^k

- 二进制数: 101101(B) =
  1×2^5+1×2^3+1×2^2+1×2^0=45(D)
  各位权值 2^k

- 十六进制数: 5F(H)=5×16^1+15×16^0=95(D)
  各位权值 16^k

- 在书写不同进位计数制数时，常常在尾部用一个字母来表示该数是什么进位计数制的数。

- 结尾用B（2进制数Binary）、O（8进制数Octonary）、D（10进制数Decimalism）、H（16进制数Hexadecimal）。缺省为十进制数。例如712O、9198D、10010B、BE49H等等。

2.各种数制间的相互转换

例如： 13 .8125 D = 1101.1101B = D.DH

- 二进制数转换为十进制数

  方法：各位二进制数码乘以对应的权之和

  例：**1.1**

  N=101101.1B =

  1×2^5+1×2^3+1×2^2+1×2^0+1×2^-1=45.5D

- 十六进制数转换为十进制数

  方法：各位十六进制数码乘以对应的权之和

  例： 1.2

  N=5FH = 5×16^1+15×16^0=80+15=95D

- 十进制数转换为二进制数

  （1）降幂法：先写出小于此数的各位二进制权 值，然后再求和。(适用于数值不大的数)

  例：1.3

  求N=13.5D的二进制数。小于此数的各位二进制权值为：8　4　2　1　0.5

- 13.5D=8+4+1+0.5 =1101.1B

1000
0100
0001
\+ 0.1
1101.1

- 十进制数转换为二进制数

  （2）除法：不断除以2，计下余数，直到商为0为止。(仅适用于整数部分)

  例：1.4

  求N=13D的二进制数。

  13/2=6　　余1　(b0)

  6/2=3　　余0　(b1)

  3/2=1　　余1　(b2)

  1/2=0　　余1　(b3)

  13D= b3b2b1b0 =1101B

- 对于十进制数的小数部分除了可以使用降幂法也可采用乘法，即不断乘2，并计下整数，而小数部分再乘2，直到结果为0为止。

- 并非所有的十进制小数都能用二进制完全表示，可按需要取一定精度即可。

- 例：1.5

  求N=0.625D的二进制数。

  0.625×2=1.25　　(b-1=1)

  0.25×2=0.5　　(b-2=0)

  0.5×2=1.0　　(b-3=1)

  N=0.625D=b-1b-2b-3=0.101B

课堂习题

把十进制数67转换为二进制数

1000011(B)

- 十进制数转换为十六进制数

（1）降幂法：先写出小于此数的各位十六进制权值，然后再求和。(适用于数值不大的数)

例：1.6

求N=95D的十六进制数。小于此数的各位十六进制权值为：

16　　1

显然应选16×5，再选1×F，所以

N=95D=80+15=16×5+1×F=5FH

- 十进制数转换为十六进制数

  （2）除法：不断除以16，计下余数，直到商为0为止。(仅适用于整数部分)

  例：1.7

  求N=95D的十六进制数。

  95/16=5　　余15 (h0)

  5/16=0　　余5 (h1)

  N=95D= h1h0=5FH

- 对于十进制数的小数部分除了可以使用降幂法也可采用乘法，即不断乘16，并计下整数，而小数部分再乘16，直到结果为0为止。

- 注意：并非所有的十进制小数都能用十六进制完全表示，可按需要取一定精度即可。

课堂习题

把十进制数67转换为十六进制数

43(H)

- 二进制数和十六进制数的相互转换

  直接转换，每四位一组，整数从低位开始，小数从高位开始，不足位补0。

  例：1.8

  N=1011111.11(B)=01011111.1100(B)=5F.C(H)

1.2.2 二进制数和十六进制数运算

- 二进制运算

  加法规则： 0+0 =0　1+0=1　0+1=1　1+1=0（进位1）

  乘法规则： 0×0=0　1×0=0　0×1=0　1×1=1

- 十六进制数运算

  原则：逢十六进一

1.2.2 二进制数和十六进制数运算

- 例1.9

```
    43A5
+   5A34
-------
    9DD9
```

- 例1.10

```
    5A34
-   43A5
-------
    168F
```

1.2.2 二进制数和十六进制数运算

- 例1.11

```
      2A34
×     0025
-----------
      D304
+    5468
-----------
     61984(H)
```

1.2.3 带符号数的表示

- 带符号数最高位是符号位。
- 正数的符号位为0，负数的符号位为1。
- 表示方法：原码、反码、补码。

原码

- 概念：
  - 原码是一种最简单的表示有符号数的方法。
  - 使用最高位（最左边的一位）表示符号：0表示正数，1表示负数。
  - 其余位表示数值的大小。
- 规则：
  - 对于正数：原码的表示与其二进制值相同。
  - 对于负数：将其绝对值的二进制表示取原码，并将最高位设置为1（表示负数）。

原码示例

- 8位原码表示
  - +5：00000101
  - -5：10000101

问题

使用原码计算5 + （-5）

```
  0000 0101  // 5的原码
+ 1000 0101  // -5的原码
---------------
  1000 1010  // -10 错误！
```

反码（Ones' Complement）

- 概念：
  - 反码是通过对原码的数值部分逐位取反（0变1，1变0）得到的。
  - 正数的反码和原码相同。
  - 负数的反码是其原码除符号位外所有位取反。
- 规则：
  - 对于正数，反码与原码相同。
  - 对于负数，反码是将原码中所有位（除了符号位）取反，即将每个二进制位从 0 改为 1，或者从 1 改为 0。

反码示例

- 8位反码表示
  - +5：00000101
  - -5：11111010

使用反码计算5 + （-5）

```
  0000 0101  // 5的反码
+ 1111 1010  // -5的反码
---------------
  1111 1111  // -0的反码
```

问题

使用反码计算5 - 3

```
  0000 0101  // 5的反码
+ 1111 1100  // -3的反码
---------------
  0000 0001  // 1 错误!
```

问题

使用原码计算5 － 3

```
  0000 0101  // 5的原码
+ 1000 0011  // -3的原码
---------------
  1000 1000  // -8 错误！
```

补码（Two's Complement）

- 概念：
  - 补码是反码加1得到的。
  - 正数的补码和原码相同。。
  - 负数的补码是其反码加1。
- 规则：
  - 对于正数，补码与原码相同。
  - 对于负数，补码是将反码加 1。

补码示例

- 8位补码表示
  - +5：00000101
  - -5： 11111011

使用补码计算5 － 3

```
  0000 0101  // 5的补码
+ 1111 1101  // -3的补码
---------------
  0000 0010  // 2
```

- 例1.12

  用8位二进制来表示，求[-3]补。

  先写出+3：　　0000 0011

  各位取反为：　1111 1100

  最低位加1为：　1111 1101

  [-3]补=1111 1101，或用十六进制表示，[-3]补=FDH

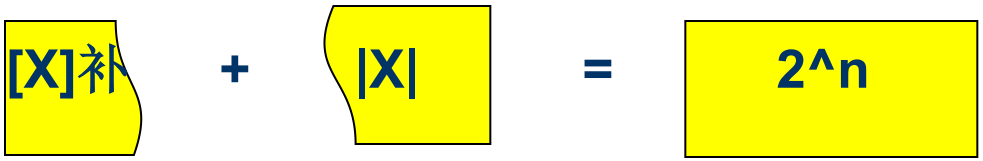
数的补码表示

定义:

（X>=0时） [X]补=符号+|X| ---(1)

（X<0时） [X]补= 2^n - |X| =(2^n - 1-|X|) + 1 ---(2)

即X<0时：



数的补码具体操作是：

正数不变，负数则用绝对值取反+1

- 例1.13

  依据补码定义写出以下各数的补码，以8位二进制表示。

  [-1]补= 256 －1 = 1 0000 0000-1= 1111 1111，直接由（2）式得到。

  [-127]补=2^8－127=（ 256 －1－127 ）+ 1

  =（1111 1111-0111 1111）+ 1

  = 1000 0000 + 1

  = 1000 0001

- 例1.14　识别以下各数的十进制值。

  [a]补=1111 1111，求补后为0000 0001 = [1]补，所以，a= -1

  [b]补=1000 0000，求补后为1000 0000 = [128]补，所以，b= -128

  [c]补=1000 0001，求补后为0111 1111 = [127]补，所以，C= -127

1.2.4 补码的加减法

- 加法规则：[X+Y]补=[X]补+[Y]补
- 减法规则：[X-Y]补=[X]补+[-Y]补
- 例:1.15

8位补码的加法运算

十进制	二进制
25	0001 1001
+（－32）	+ 1110 0000
－7	1111 1001
32	0010 0000
+（－25）	+ 1110 0111
7	0000 0111
	1↙

1.2.5 无符号数的表示

- 对于正数，不保留符号位，把符号位也作为数值，这样的数叫无符号数。

1.2.6 字符的表示

- ASCII码
- 扩充的ASCII码
- 表1-2 P8

  回车:0dh

  换行:0ah

  空格:20h

  0~9:30h~39h

  A~Z:41h~5ah

  a~z:61h~7ah

1.2.7 基本逻辑运算

- 逻辑运算按位操作
- 与运算 **AND**
- 或运算 **OR**
- 异或运算 **XOR**
- 非运算 **NOT**