

《汇编语言程序设计》

教学辅导与上机实验辅导

罗万钧 吴方中 编



313

西安电子科技大学出版社

内 容 简 介

E

本书是配合《汇编语言程序设计》一书的参考教材。内容包括：《汇编语言程序设计》教学辅导，给出了西安电子科技大学出版社出版的《汇编语言程序设计》(罗万钧主编)中各章部分习题的解答，其中编程序的全部习题都给出了参考解答，并对每章的主要内容、要求、学习方法作了扼要的说明；上机实验辅导，给出了实验题目及实验程序的参考清单。

《汇编语言程序设计》 教学辅导与上机实验辅导

罗万钧 吴方中 编

责任编辑 徐德源

西安电子科技大学出版社出版发行

西安市高陵县印刷厂印刷

陕西省新华书店发行 新华书店经售

开本 787×1092 1/16 印张：9.125 字数：216 千字

1998 年 5 月第 1 版 1998 年 5 月第 1 次印刷 印数：1~4 000

ISBN 7 - 5606 - 0561 - 3/TP · 0279

定价：9.20 元

目 录

第 1 部分 教学辅导

第 1 章 计算机基础	3
一、计算机基本组成	3
二、存储器组织	3
三、计算机中的数制与码制	3
四、机器语言、汇编语言、高级语言	3
五、部分习题解答	4
第 2 章 8088/8086 系统硬件结构	6
一、8088 CPU 的内部寄存器	6
二、段基址、段内偏移量及物理地址	6
三、8088 CPU 内部两大功能部件	6
四、部分习题解答	7
第 3 章 8088/8086 指令系统	8
一、寻址方式	8
二、指令系统	11
三、部分习题解答	13
第 4 章 8088/8086 汇编语言	18
一、汇编语言程序结构	18
二、汇编语言语句	22
三、汇编语言伪指令	24
四、其它	25
五、部分习题解答	25
第 5 章 基本程序设计	30
一、程序的 4 种基本结构	30
二、顺序程序设计	30
三、分支程序设计	30
四、循环程序设计	31
五、子程序设计	32
六、部分习题解答	33
第 6 章 8088/8086 汇编程序开发过程	51

第 7 章 数值运算程序设计	52
第 8 章 非数值运算程序设计	62
第 9 章 输入输出程序设计	79
一、输入输出的基本指令	79
二、CPU 与外设间的数据传送方式	79
三、部分习题解答	80
第 10 章 中断程序设计	85
一、中断的概念	85
二、中断类型号与中断服务程序入口地址的关系	85
三、部分习题解答	86
第 11 章 系统调用程序设计	89
一、DOS 系统调用及 BIOS 功能调用的概念	89
二、应掌握的系统调用	90
三、应掌握的 BIOS 功能调用	90
四、部分习题解答	90
第 12 章 汇编语言与高级语言的接口	100
一、在 C 语言的语句行间嵌入汇编程序	100
二、编写被 C 语言调用的汇编子程序	103

第 2 部分 上机实验辅导

一、概述	109
二、实验一 利用 DEBUG 调试程序调试程序段	109
三、实验二 分支程序实验	110
四、实验三 循环程序实验	112
五、实验四 子程序实验(一)	114
六、实验五 子程序实验(二)	118
七、实验六 字符处理程序实验	122
八、实验七 输入输出实验	125
九、实验八 中断程序实验	130
十、实验九 系统调用程序实验	134
十一、实验十 语言接口实验	139

第 1 部分

教 学 辅 导

- | | |
|---|---|
| ☐ 计算机基础 | ☐ 8088/8086 系统硬件结构 |
| ☐ 8088/8086 指令系统 | ☐ 8088/8086 汇编语言 |
| ☐ 基本程序设计 | ☐ 8088/8086 汇编程序开发过程 |
| ☐ 数值运算程序设计 | ☐ 非数值运算程序设计 |
| ☐ 输入输出程序设计 | ☐ 中断程序设计 |
| ☐ 系统调用程序设计 | ☐ 汇编语言与高级语言的接口 |

第 1 章 计算机基础

本章共有 3 节内容，主要介绍计算机的基本结构及有关的技术名词，此外对计算机中采用的数制与编码做了全面扼要的介绍。

要求掌握的内容要点有：

一、计算机基本组成

计算机由硬件子系统和软件子系统组成。

计算机硬件子系统由运算器、控制器、存储器、输入输出设备等组成。

微型计算机硬件系统结构多为单总线的系统结构。总线是一簇用来进行信息传递的公共信号线，单总线由地址总线 AB、数据总线 DB 和控制总线 CB 组成。

二、存储器组织

对于初学计算机的人来讲，首先要了解计算机执行指令的过程，从弄清存储器组织入手是一种较好的捷径。

需掌握的技术名词有：二进制位 Bit，字节 Byte、字 Word 及双字 Double Word 及它们之间的相互关系；存储单元地址和存储单元的内容；存储器的读及存储器的写操作过程。

三、计算机中的数制与码制

常用的进位计数制有二进制、八进制、十进制、十六进制等。但在计算机中，信息的存储、处理都是以二进制代码来表示的。

要求掌握二进制数与十进制数之间的转换方法。

数在机器中的表示形式称为机器数。带符号数的机器数表示形式有原码、反码和补码 3 种表示法。要求熟练掌握求一个带符号数的原码、反码和补码的方法。

已知一个数 X 的补码 $[X]_{\text{补}}$ ，去求 $[-X]_{\text{补}}$ 的过程称为求补或取补操作。即将 $[X]_{\text{补}}$ 连同符号位一起求反加 1 得到 $[-X]_{\text{补}}$ 。 $[-X]_{\text{补}} = [[X]_{\text{补}}]_{\text{求补}}$ 。

BCD 码是将十进制数的每一位用 4 位二进制数表示的编码。常用的 BCD 码有 8421 BCD 码。

可以对各种字符进行编码，普遍采用的编码有 ASCII 码。

四、机器语言、汇编语言、高级语言

机器语言就是能被计算机直接识别并执行的语言，也即由二进制编码的机器指令表示的语言，是人们最早使用的一种程序设计语言。

指令是能被计算机识别并执行的二进制代码。

一台具体的计算机所能识别并执行的全部指令，就是该计算机的指令系统。

程序是计算机一系列指令的有序集合。

汇编语言由汇编指令、汇编伪指令和相应的语法规则所组成。它是一种符号化的程序设计语言。

在汇编指令中，操作码、数据、寄存器或数据的地址都用相应的符号表示。

由汇编语言编写的程序称为汇编语言程序，或者称为汇编语言源程序。

能实现将汇编语言源程序汇编(翻译)成为机器语言程序的专用软件称为汇编程序。

可以这样说：汇编程序能够正确地识别汇编语言源程序中的所有汇编指令及汇编伪指令并进行正确的汇编。而汇编语言则用事先约定好的汇编指令、汇编伪指令及语法规则，告诉汇编程序应该如何(实现)正确地进行汇编。汇编语言与汇编程序是相互对应的。

高级语言是脱离具体的计算机的面向过程的程序设计语言。

五、部分习题解答

5. 指出在计算机中对二进制、八进制、十进制及十六进制代码书写的形式。0FFH, 177Q, 1970D, 01101101B 等代码表示什么计数制的代码?

【解】 在计算机中，二进制代码书写时，结尾用字母 B 标志，八进制代码结尾用字母 Q 标志，十进制代码结尾用字母 D(或省略)标志，十六进制代码结尾用字母 H 标志。因此，0FFH 为十六进制数，177Q 为八进制数，1970D 为十进制数，01101101B 为二进制数代码表示形式。

7. 将下列十六进制数转换成二进制数，将十进制数表示为对应的二进制数及十六进制数：

1A2B3CH, 07FFH, 0ACBH, 77DEH, 4062, 2938, 3656。

【解】 $1A2B3CH = 000110100010101100111100B$

$07FFH = 0000011111111111B$

$0ACBH = 0000101011001011B$

$77DEH = 0111011111011110B$

$4062 = 111111011110B = 0FDEH$

$2938 = 101101111010B = 0B7AH$

$3656 = 111001001000B = 0E48H$

8. 当字长为 8 位时，写出下列各数的原码、补码和反码： $X = -00100101B$, $Y = -01011010B$ 。

【解】 $X = -00100101B$, $[X]_{\text{原}} = 10100101B$

$[X]_{\text{反}} = 11011010B$, $[X]_{\text{补}} = 11011011B$

$Y = -01011010B$, $[Y]_{\text{原}} = 11011010B$

$[Y]_{\text{反}} = 10100101B$, $[Y]_{\text{补}} = 10100110B$

9. 已知 X 的补码，求 X 的真值及 $-X$ 的补码：

$[X1]_{\text{补}} = 10100101B$

$[X2]_{\text{补}} = 11111111B$

【解】 $X1 = -1011011B$, $[-X1]_{\text{补}} = 01011011B$
 $X2 = -0000001B$, $[-X2]_{\text{补}} = 00000001B$

第 2 章 8088/8086 系统硬件结构

本章共两节。第一节为 CPU 的功能结构，它是深入理解指令功能的基础，必须熟练掌握。第二节内容只要求一般掌握和了解。

要求掌握的内容要点有：8088 CPU 的内部寄存器，段基址、段内偏移量及物理地址，8088 CPU 内部两大功能部件等内容。

一、8088 CPU 的内部寄存器

8088 CPU 内部共有 14 个 16 位的寄存器：其中 AX、BX、CX 及 DX 为数据寄存器，它们又可以作为 8 个独立的 8 位寄存器使用，即分为 AH、AL、BH、BL、CH、CL、DH 和 DL 8 个 8 位的寄存器；两个 16 位的指针寄存器 BP 及 SP，两个 16 位的变址寄存器 SI 及 DI，它们与 4 个数据寄存器组成通用寄存器组；两个控制寄存器 IP 及 F，IP 为指令指针寄存器，F 为标志寄存器，F 包含 9 个标志位，即 OF、DF、IF、TF、SF、ZF、PF、AF 和 CF；另有 4 个段基址寄存器 CS、DS、ES 及 SS。以上内部寄存器必须熟练地掌握。

二、段基址、段内偏移量及物理地址

8088 CPU 有 20 根地址线，可以寻址 1 MB 的内存空间，20 位的地址代码从 00000H 至 FFFFFH，称之为物理地址。

真正的物理地址是由 16 位的段基址及 16 位的段内偏移量(也称为有效地址 EA)按一定的算法形成的。即物理地址等于相应的段基址乘 16(相当于左移 4 位二进制位)的积与 16 位的段内偏移量之和。

在 8088 汇编指令中，只显式地给出操作数的段内偏移量，即有效地址 EA，而将相对来说来并不经常变化的段基址以隐含的方式给出。它们分别存放在段寄存器 DS、CS、ES 及 SS 中。一般情况下，规定通用数据的读写从 DS 所指定的段中存取；规定指令代码的读出从代码段寄存器 CS 所规定的段中读出；对堆栈段的读写则指定从堆栈段寄存器 SS 所指定的段中读写；在串指令执行时，源串数据从 DS 所指定的数据段中读出，目的串数据则向附加数据段 ES 所指定的段中写入。除非指令中有特殊的段超越说明，否则均按以上规则对有关的段进行操作。

三、8088 CPU 内部两大功能部件

8088 CPU 由两大功能部件组成，即由总线接口单元 BIU 和指令执行单元 EU 组成。

EU 负责指令的执行，BIU 用于实现 EU 的所有总线操作。

四、部分习题解答

6. 已知 $CS=3500H$, $IP=2600H$, 问物理地址为多少?

【解】 $CS:IP$ 用于确定待执行的下一条指令的地址。计算物理地址算法为:

$$\begin{array}{r} 35000H \\ +) 2600H \\ \hline 37600H \end{array}$$

所以, 物理地址为 $37600H$ 。

7. 已知物理地址为 $FFFF0H$, 且段内偏移量为 $0A000H$, 放在 BX 中, 若对应的段基址值放在 DS 中, 问 $DS=?$

【解】 因物理地址由 $DS:BX$ 所形成, 所以求出相应段地址的算法为:

$$\begin{array}{r} FFFF0H \\ -) A000H \\ \hline F5FF0H \end{array}$$

所以 DS 的内容应为 $F5FFH$ (16 位)。

8. 8088 中对段寄存器使用的约定, 在哪些情况下是不允许违例的?

【解】 对段寄存器的使用, 凡取指令都是从 CS 段中取出的, 凡堆栈的压入和弹出, 都是对 SS 段进行的。在这种情况下, 是不允许对约定违例的。

第 3 章 8088/8086 指令系统

本章是该教材的重点之一，是进行汇编语言程序设计的基础。其内容包括寻址方式和指令系统两大部分。该章内容较多，在短时期内很难达到全面掌握的目的。在本章对 8088/8086 指令系统进行系统的介绍，要求有全面系统的了解。通过以后章节的学习及上机实习，应能熟练准确地使用它进行程序设计。

弄清以下 4 个问题，才算掌握了一条指令：

第一，指令助记符是什么？第二，指令所允许的操作数类型及寻址方式有哪些？第三，指令所完成的具体操作是什么？第四，执行该指令有哪些特殊的约定，执行后对标志位有哪些影响？

要求掌握的内容要点有：

一、寻址方式

1. 指令中操作数的类型

8088 指令系统中的操作数类型有 3 种，即立即数操作数、寄存器操作数和存储器操作数。

2. 寻址方式的概念

所谓寻址方式，就是寻找指令中操作数所在地址的方式。

3. 8088 指令系统寻址方式的类型

8088 指令系统的寻址方式有以下 8 类：

立即数寻址

寄存器寻址

直接寻址

寄存器间接寻址

变址寻址

基址变址寻址

串操作数寻址

输入输出指令的端口寻址

除了以上 8 类寻址方式外，某些教材中还有隐含寻址方式及相对寻址方式。隐含寻址是指某些指令执行时，其操作数是事先约定好的，在汇编指令中则被隐含掉。例如乘法指令、除法指令及十进制数调整指令，均指定被乘数、被除数或被调整的对象在 DX、AX 或 AL 中。相对寻址方式用在控制转移类指令中，其目标地址是当前的 IP 值与一个相对位移量的和。对这些寻址方式在介绍这些指令时予以解释。

4. 立即数操作数与立即寻址

立即数操作数是作为指令代码的一部分出现在双操作数指令中的源操作数。有时也将立即数操作数称为指令操作数。将操作数直接包含在指令中提供的寻址方式称为立即寻址方式。立即数可以是 8 位的，也可以是 16 位的。立即数只能作为源操作数出现，用于目的操作数是没有意义的。例如：

```
MOV AX, 0FFH      ; 字立即数
MOV AL, 64H       ; 字节立即数
```

5. 寄存器操作数及寄存器寻址

存放于寄存器中的操作数叫寄存器操作数。将操作数存于任意一个通用寄存器中的寻址方式称为寄存器寻址。例如：

```
MOV DS, AX
```

6. 存储器操作数及有关的寻址方式

存放在存储器中的操作数叫存储器操作数。要从存储器中得到一个操作数，或者将一个操作数存放到存储器中，都必须对存储单元进行访问，都必须找到相应的存储单元的段基址及段内偏移量，以形成有效地址。

为了确定有效地址，相关的寻址方式有直接寻址、寄存器间接寻址、变址寻址及基址变址寻址。

7. 寄存器间接寻址

操作数所在的存储单元的有效地址存放在某个间址寄存器 DI、SI、BX 或 BP 中。其中 DI、SI、BX 间接寻址时，操作数在数据段 DS 中，而 BP 间接寻址时，操作数在堆栈段 SS 中，除非指明段超越。例如：

```
MOV AL, [BX]      ; 段寄存器为 DS
MOV AL, [BP]      ; 段寄存器为 SS
MOV AL, ES:[BX]   ; 段寄存器为 ES
```

8. 直接寻址

这种寻址方式将操作数地址的 16 位偏移量直接包含在指令代码中。对直接寻址，操作数一般隐含在数据段 DS 中。例如：

```
MOV AL, [1000H]   ; 用方括号内的内容指出段内偏移地址
```

方括号内为数字值，表示是直接寻址，而不是间接寻址。间接寻址时，方括号内为寄存器名字。阅读和使用时应将它们加以区分。

9. 变址寻址

变址寻址与寄存器间址的区别在于：变址寻址是以某个寄存器中的值作为基地址，然后再加上一个 8 位或 16 位的直接位移量，例如：

```
MOV AX, DAT[SI]
```

DAT 即为 8 位或 16 位的直接位移量，也可表示为 $MOV\ AX, [SI+DAT]$ ，因为它的有效地址 $EA = (SI) + DAT$ 。

能够作为变址寄存器使用的有 SI、DI、BX 及 BP，同样，当使用 SI、DI 及 BX 时，操作数在数据段 DS 中，当使用 BP 时，操作数在堆栈段 SS 中。

10. 基址变址寻址

顾名思义,这种寻址方式由基址寄存器(BX 或 BP)的内容加上一个变址寄存器(SI 或 DI)的内容,再加上指令中给出的一个 8 位或 16 位的直接位移量而构成有效地址。例如:

```
MOV AX, DAT [BX][SI]
```

也可以写成:

```
MOV AX, [BX+SI+DAT]
```

```
MOV AX, [BX] [SI+DAT]
```

因为其有效地址为 $EA = (BX) + (SI) + DAT$ 。

当以 BX 为基址寄存器时,可与变址寄存器 SI 或 DI 组合,在 DS 段中寻址存储单元。
当以 BP 为基址寄存器时,也可以与变址寄存器 SI 或 DI 组合,在 SS 段中寻址存储单元。

11. 串操作数寻址

这种寻址方式除使用隐含变址寄存器 SI 或 DI 的方法寻址操作数以外,在指令执行后,还具有自动修改变址寄存器值的功能。规定源串操作数偏移量由 SI 指定,段基址由 DS 指定;目的串操作数偏移量由 DI 指定,段基址由 ES 指定。例如:

```
MOVSB
```

REP MOVSB ; 带有重复前缀 REP 的串指令,重复次数由 CX 内容决定;修改变址;
; 值的增量或减量方向由 DF 决定;增减幅度为 1 或 2 由操作数是字
; 节或字决定;对 MOVSB,增减幅度为 1

12. 端口寻址及隐含寻址

对端口寻址及隐含寻址要求一般掌握即可。

端口寻址,是指在执行专用的输入输出指令,如 IN 或 OUT 指令时,操作数的地址(对前者为目的操作数,对后者为源操作数)指定为 AL 或 AX 的内容。对前者的源操作数及后者的目的操作数地址,一般用端口地址给出。端口地址空间是一个独立的寻址空间。端口地址可为 8 位或 16 位。为 16 位时,端口地址必须先送入 DX 中。例如:

```
IN AL, PORT8 ; PORT8 为 8 位端口地址
```

```
MOV DX, PORT16
```

```
IN AL, DX ; DX 中为 16 位的端口地址
```

此处的 PORT8 可看作 8 位的直接寻址,而 DX 则应看作为 16 位的间址寄存器,然而书写格式却易使人产生误解。例如下面指令应理解为

```
IN AL, PORT8 ; (PORT8)→(AL)
```

```
IN AL, DX ; (DX)→(AL)
```

隐含寻址,由于某些指令事先指定了(或约定了)某个寄存器参与操作,但在汇编指令格式中又隐含掉这些寄存器。例如:

```
MUL CX ; (AX) * (CX)→(DX)(AX)
```

```
MUL CL ; (AL) * (CL)→(AX)
```

```
DIV CX ; (DX)(AX)/(CX)→(AX), 余数送(DX)
```

```
DIV CL ; (AX)/(CL)→(AL), 余数送(AH)
```

MUL 及 DIV 指令中,看不到 AX 或 DX 或 AL,因为它们被隐含了。

二、指令系统

1. 指令分类

按指令功能不同,可分为 9 种类型指令:

数据传送类

算术运算类

逻辑运算类

移位类

转移控制类

字符串操作类

处理器控制类

输入/输出类

中断类

2. 数据传送类指令

数据传送类指令是使用最多,也是最基本的指令。它包括数据传送、堆栈操作、数据交换、换码(查表)、标志传送和地址传送 6 组指令,应该全部熟练掌握,尤其是前 3 组。

此处特别要求掌握 MOV 指令。

MOV OPRD1, OPRD2; OPRD1→OPRD2

OPRD1 为源操作数,OPRD2 为目的操作数。

OPRD1 可为寄存器、存储器操作数。寄存器可为通用寄存器 AX、BX、CX、DX、BP、SP、DI、SI 以及段寄存器 CS、DS、ES、SS。

OPRD2 可为寄存器、存储器操作数及立即数。

在同一条 MOV 指令中,允许通用寄存器之间相互传送;允许通用寄存器与段寄存器之间相互传送;允许通用寄存器与存储器之间相互传送;允许存储器与段寄存器之间相互传送;允许立即数传送至通用寄存器或存储器中。对应的例子如下:

MOV AX, BX

MOV DS, AX

MOV AX, CS

MOV AL, [SI]

MOV [DI], AL

MOV DS, [BX]

MOV [SI], ES

MOV AX, 0FFEh

MOV BYTE PTR[DI], 0FH

使用 MOV 指令时应当注意以下几点:

- (1) 目的操作数不允许使用立即数和 CS 寄存器。
- (2) 两个存储单元间和两个段寄存器间不允许直接传送。
- (3) 立即数不允许直接送段寄存器。
- (4) 立即数送存储器时,二者类型一定要匹配。

3. 算术运算类指令

对该类指令要求熟练掌握以下内容：

- (1) 加法指令 ADD 及 ADC
- (2) 减法指令 SUB 及 SBB
- (3) 乘除法指令 MUL 及 DIV
- (4) 增量减量指令 INC 及 DEC
- (5) 调整指令 DAA 及 DAS
- (6) 其它 CMP NEG CBW

弄清 SUB 与 CMP 指令的区别。

4. 逻辑运算指令

逻辑类指令中要求熟练掌握以下指令：

AND OR NOT XOR TEST

弄清 AND 与 TEST 指令的区别。

5. 移位类指令

要求熟练掌握以下指令：

SAL SAR RCL RCR

掌握下列指令书写格式之间的关系：

SAL AL, 1

SAL AL, CL

32 位操作数左移一位或右移一位的实现方法。

6. 转移类指令

该类指令要求熟练掌握以下指令：

JMP

JC/JNC JE/JNE JS/JNS JO/JNO JP/JNP JZ/JNZ JA JB JG JL

LOOP LOOPZ/LOOPNZ

CALL RET

弄清段内转移与段间转移的区别，弄清段内调用与段间调用的区别，弄清 RET n 指令的意义。

7. 字符串操作类指令

要求熟练掌握以下指令：

MOVSB CMPSB SCASB LODSB STOSB

REP REPZ REPNZ

8. 处理器控制类指令

该类指令应熟练掌握以下内容：

CLC/STC CLD/STD LIC/STI HLT NOP

9. 输入输出类指令

要求熟练掌握下列指令：

IN AL n OUT n, AL

IN AL DX OUT DX, AL

IN AX n OUT n, AX
IN AX DX OUT DX, AX

10. 中断类指令

只要求熟练掌握软中断指令 INT n。

三、部分习题解答

5. 下列每条指令间有什么不同?

- (1) MOV BX, OFFSET TABLE+4
- (2) MOV BX, TABLE+4
- (3) LEA BX, TABLE+4

【解】 (1)与(3)两条指令的功能相同,都是将变量 TABLE+4 的有效地址(或段内偏移量)送 BX 寄存器。(3)直接用指令来实现,(1)中使用了属性返回操作数 OFFSET。

(2)与(1)及(3)的功能不同,它将变量 TABLE+4 处的连续两个存储单元的内容(存储器字)送 BX。高地址单元的内容送 BH,低地址单元的内容送 BL 中。

6. 指出下列指令中,源操作数及目的操作数的寻址方式。

直接写出解答如下:

- (1) SUB AX, [BP+6] ; 源为变址寻址,目为寄存器
- (2) MOV [DI+BX], AX ; 源为寄存器,目为基址变址
- (3) MUL CL ; 源为寄存器,目为 AX 隐含
- (4) IN AL, 22H ; 源为端口,目为寄存器
- (5) ADD AX, 25[SI] ; 源为变址,目为寄存器
- (6) MOV AX, 100 ; 源为立即数,目为寄存器
- (7) JMP BX ; 目为寄存器间接
- (8) SCASB ; 串操作指令寻址
- (9) MOV AL, [2000H] ; 源为直接寻址,目为寄存器
- (10) INC BYTE PTR[BX+SI] ; 源目为基址变址寻址

7. 判别下列指令是否有错并说明理由。

直接写出解答如下:

- (1) MOV [DI],[SI] ; 错,两个存储单元间不能直接传送
- (2) MOV CS, AX ; 错,CS 不能作为目的操作数
- (3) MUL -25 ; 错,MUL 为不带符号数的乘法指令,-25 为符号数
- (4) MOV 128, CL ; 错,立即数不能作目的数
- (5) MOV DS, CS ; 错,段寄存器间不能直接进行传送
- (6) ADD AX, [SI][DI] ; 错,两个变址寄存器不能形成基址变址寻址
- (7) PUSH FLAG ; 错,无此指令,可用 PUSHF
- (8) MOV AX, BYTE PTR[SI] ; 错,类型不匹配。源为字节型,目为字类型
- (9) MOV BL, SI+2 ; 错,类型不匹配。BL 为 8 位;SI+2 为 16 位
- (10) MOV CL, AX ; 错,类型不匹配