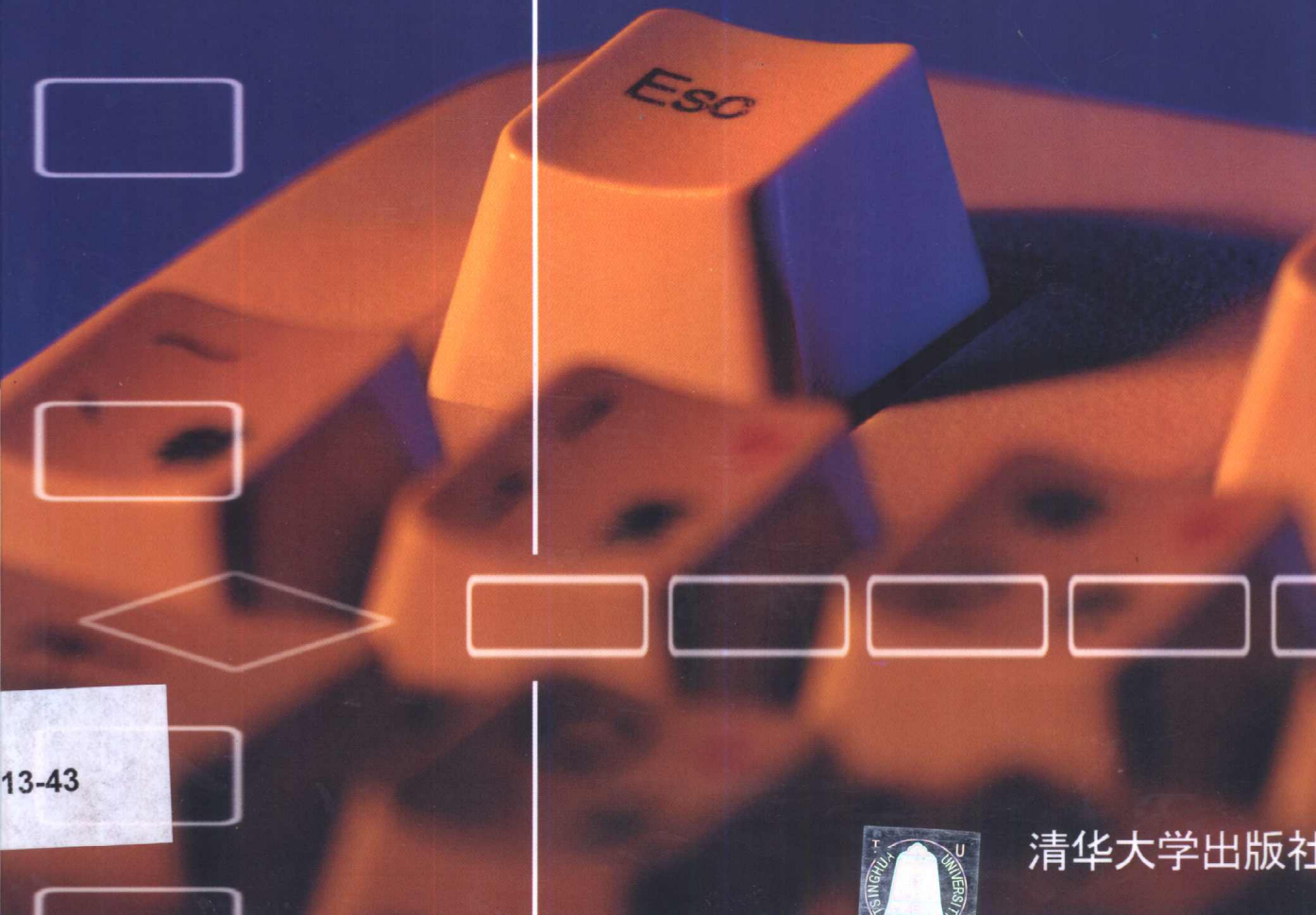
CASL

程序设计教程

刘克武 李冰 李冬梅 编著



13-43

清华大学出版社

158

TP 313-43
L72

CASL 程序设计教程

刘克武　李　冰　李冬梅　编著

清华大学出版社

(京)新登字 158 号

内 容 简 介

CASL 是建立在一种假想机上的汇编语言系统，汇集了当今主流 PC 机的指令结构和功能。本书以程序设计为纲，全面系统地介绍了 CASL 汇编语言。全书共分 9 章，分别讲述了 CASL 程序设计环境、伪指令和宏指令、数的存取和传送、算术运算和算术操作、逻辑运算和逻辑操作、比较与转移、数据栈与子程序、程序设计基础和例题分析等内容。本书每章后面都附有习题，供读者练习参考。

本书可作为高等院校汇编语言程序设计课程的教材，也可供参加“中国计算机软件专业技术资格和水平考试”的考生备考使用。

版权所有，翻印必究。

本书封面贴有清华大学出版社激光防伪标签，无标签者不得销售。

书　　名：CASL 程序设计教程

作　　者：刘克武　李冰　李冬梅　编著

出 版 者：清华大学出版社（北京清华大学学研大厦，邮编 100084）

http://www.tup.tsinghua.edu.cn

责任编辑：刘　彤

印 刷 者：北京昌平环球印刷厂

发 行 者：新华书店总店北京发行所

开　　本：787×1092 1/16　**印张**：13　**字数**：298 千字

版　　次：2002 年 7 月第 1 版　2002 年 7 月第 1 次印刷

书　　号：ISBN 7-302-05507-6/TP・3237

印　　数：0001～5000

定　　价：23.00 元

此为试读，需要完整PDF请访问：www.ertongbook.com

前 言

汇编语言是一种对计算机硬件有很强依赖性的低级语言，学习汇编语言可以深入到计算机内部了解计算机底层的部件功能及运作。掌握了汇编语言可以设计系统软件，实施工业过程控制自动化，汇编语言在计算机众多应用领域中是一种必不可少的，也是不可替代的计算机语言。

汇编语言因机器不同而异，随着计算机更新换代，汇编语言也在不断地推陈出新，从而使得汇编语言在使用时效上较为短暂。如何延长汇编语言的使用时效，使其像当今的高级语言一样也具有普遍性和通用性呢？CASL 汇编语言的出现和推广在很大程度上解决了汇编语言时效短的问题。

CASL 是建立在一种假想机 COMET 的汇编语言系统，这种超脱使得它不存在因机器不同而异的问题；CASL 又汇集了当今主流 PC 的指令结构和功能，因此它又不失其实用性。学习 CASL 可以打下良好的汇编语言基础，掌握了 CASL 可以轻松、快捷地学会当今主流 PC 的汇编语言。

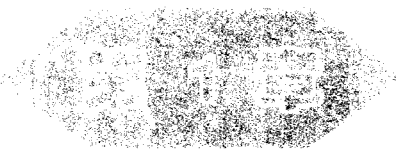
本书以程序设计为纲，全面、系统地介绍了 CASL 汇编语言。内容由浅入深，循序渐进，一章一个重点，一章一个学习周期。全书共分 9 章，每章都有习题。1 至 7 章及附录 1 由刘克武编写；8、9 两章由李冰编写；附录 2、3 由李冬梅编写，刘克武负责全书修正、定稿。

本书不仅可以作为高等院校“汇编语言程序设计”课程的教材，还可以供“计算机软件专业技术资格和水平考试”的考生备考使用。

编 者

2002 年 5 月

目 录



CASL 程序设计环境

CASL 是 Computer Assembly System Language 的简称，可译为计算机汇编语言系统。

由于汇编语言因计算机不同而异，所以当计算机的 CPU 更新换代时就会使得汇编语言随之而改变，因此造成汇编语言在使用时效上比高级语言短得多。CASL 汇编语言是建立在一个名为 COMET 假想计算机上的汇编语言系统，因此它在很大程度上克服了汇编语言使用时效短的问题。加之 CASL 集中了当今主流 PC 所使用的汇编系统的主要功能及指令形式，所以学习 CASL 与使用当今 PC 是紧密相关的。掌握了 CASL 程序设计，可以打下良好的汇编语言程序设计基础，具备了 CASL 基础，可以迅速学会当今建立在任何计算机上的汇编语言系统。因此可以说学习 CASL 即有广泛意义，又有现实意义。

汇编语言对计算机硬件有很大的依赖性，CASL 也不例外。因此学习 CASL 就需要了解 COMET 计算机，这就是 CASL 的硬件背景，而有关 CASL 的表述形式，指令格式，数的使用与表达等，这些内容则称为 CASL 的软件环境。CASL 的硬件背景与软件环境就是本章要介绍的 CASL 程序设计环境。

1.1 CASL 的硬件背景

1.1.1 COMET 计算机的结构

COMET 是一个 16 位机，每一个单元长度为两个字节。在 COMET 上可以进行 16 位的定点整数运算，逻辑运算，码制变换及字符处理等。

COMET 的组成有 CPU、内存、外部设备等 3 大部分，它与当今的个人计算机(Personal Computer)很相似，其结构示意图如图 1.1 所示。

1.1.2 COMET 的 CPU

CPU(Central Processing Unit)是中央处理器的简称。CPU 与 CASL 有关的部件有通用寄存器、指令计数器、标志寄存器、栈指针等。指令操作部件 OP 在 CASL 的描述中并不出现。

此为试读，需要完整PDF请访问：www.ertongbook.com

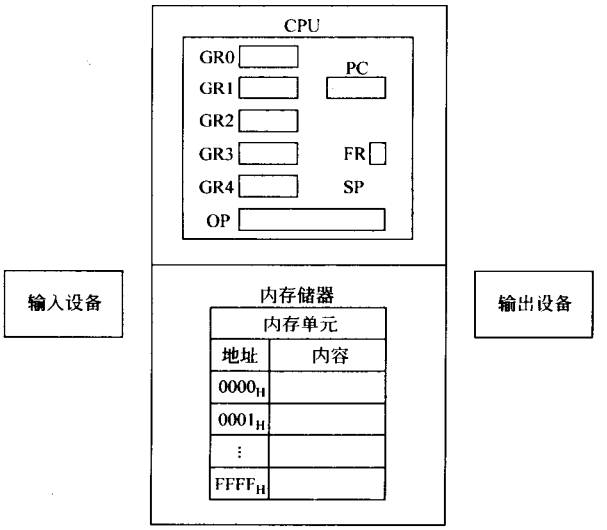


图 1.1 COMET 假想机结构

(1) 通用寄存器 GR(General Register)

COMET 的 CPU 设置了 5 个通用寄存器，它们分别是 GR0、RG1、GR2、GR3、GR4。这 5 个寄存器均在 CASL 指令中出现。

通用寄存器在 CASL 中既是一个存储空间，又是一个处理部件，使用它可以存储参加运算的数据，保留运算或处理的中间结果或最后结果。在运算或处理中，它可以作为累加器、移位运算器及个数计数器等。CASL 所有指令的描述几乎都与通用寄存器有关。

通用寄存器的空间长度为两个字节，16 位，它与内存单元的长度一样。因此，寄存器与单元相互交换数据十分方便。寄存器表示无符号数的范围为 0000_H～$FFFF_H$，这个值正好是内存单元的地址范围。

通用寄存器可以作为地址寄存器及变址寄存器，但 GR0 不允许用作变址寄存器。

(2) 指令计数器 PC(Program Counter)

指令计数器也叫程序计数器，程序计数器记录着当前被执行的指令的地址。由于 CASL 指令长度为 4 个字节，32 位，占两个内存单元，所以执行完一条指令，指令计数器的内容自动加 2，表示为(PC)+2→PC，而遇到转子、转移指令时除外。

(3) 标志寄存器 FR(Flag Register)

标志寄存器也叫标志寄存器，它的作用是记录指令执行后的状态。在 CASL 指令系统中有 12 条指令执行后都会对标志寄存器产生影响，或者说标志寄存器对这 12 条指令执行后的状态都予以记录。

标志寄存器的长度为两位，两位所能表示的状态最多为 4 种，这 4 种状态为 00、01、10、11。11 这种状态不使用，所以只有 3 种状态用于指令执行后的状态标志，具体规定如表 1.1。此外，两位标志寄存器，左侧的一位表示符号，0 为正，1 为负。

(4) 栈指针 SP(Stack Pointer)

数据栈也称堆栈，是一种特定的内存空间。在这个空间中可以存储数据，也可以取出

存入的数据。存入数据叫数据进栈，取出数据叫数据出栈。数据栈的特定性能是“先进后出”或“后进先出”。这个性能通俗地说就是，最先进入数据栈的数据，最后才能出栈。数据栈的这种有序出、入数据，不但体现出栈的空间含义，而且包含了一个先、后的时序概念。

表 1.1　3 种用于指令执行后的状态标志规定

执行完运算或操作	执行完比较指令	FR 中的状态
① 运算结果为正数	比较结果为大于	00
② 运算结果为零	比较结果为等于	01
③ 运算结果为负数	比较结果为小于	10

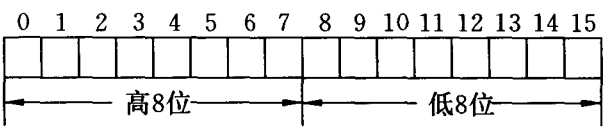
栈指针也叫栈顶指针，它在数据栈的操作过程中所描写的是数据栈的存数状况。数据进栈，栈顶上升，栈指针上浮；数据出栈，栈顶下降，栈指针下沉。但不管数据栈的数据怎样变化，栈指针永远指示着栈顶所在的单元地址。

在 CASL 中，栈指针规定用寄存器 GR4。用圆括号表示 GR4 的内容，则(GR4)表示栈顶地址。而数据进栈，则有(GR4)－1→GR4；数据出栈，栈指针变化为(GR4)＋1→GR4。

1.1.3　COMET 的内存储器

(1) COMET 的内存容量

COMET 的内存容量为 65536 个字，即 64K 字(1K＝1024)。一个字的长度，即字长为两个字节，16 位。16 位的编号如下：



在 16 位的一个字长中，第 0 位为最高位，在表示有符号数时为数的符号位，第 15 位为最低位。

用一个字 16 位存储无符号数时，表示范围为 0～$12^{16}-1$，用十六进制表示时为 0～FFFF。

用一个字 16 位存放有符号数的补码时，表示范围为-2^{15}～$2^{15}-1$，即－32768～32767。

用一个字 16 位存储一个字符的常数时，高 8 位为 0，低 8 位中的 7 位为美国信息交换标准代码 ASCII。

(2) 内存单元与地址

一个单位的存储空间称为一个单元，65536 个字，即 65536 个单元。由于单元编号，即地址从 0 开始，因此地址的表示范围为 0～65535，用十六进制表示这个地址范围为 0000_H～$FFFF_H$。在 CASL 中，内存单元地址使用标号地址，当表示连续的若干单元的地址时，只需要一个首地址。例如图 1.2 所示。

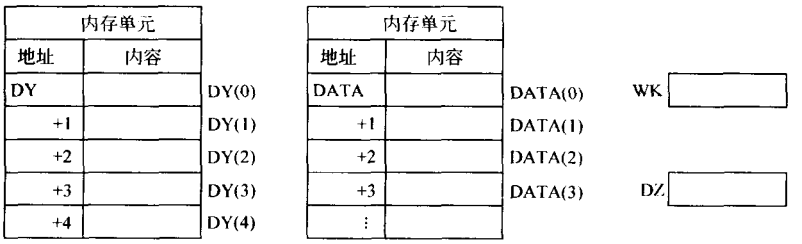


图 1.2 内存单元与地址

图 1.2 中 WK,DZ 是在 CASL 中使用的内存单元地址,由标号名表示,它很像高级语言中的变量名。而 DY,DATA 是在 CASL 中使用的连续单元的首地址,它们很像高级语言中的数组名。DY,DATA 为首地址的连续单元中的每一个单元地址很像数组元素名。怎样命名标号,如何定义内存单元将在第 2 章介绍。

1.2 CASL 的软件环境

1.2.1 CASL 的字符集

描写 CASL 的字符全体称为 CASL 的字符集,或者说,在 CASL 中允许使用的字符全体称为 CASL 的字符集。CASL 字符集如表 1.2 所示。

表 1.2 CASL 汇编字符集

高位 / 低位	2	3	4	5
0	间隔	0	@	P
1	!	1	A	Q
2	"	2	B	R
3	#	3	C	S
4	$	4	D	T
5	%	5	E	U
6	&	6	F	V
7	'	7	G	W
8	(	8	H	X
9	)	9	I	Y
A	*	:	J	Z
B	+	;	K	[
C	,	<	L	\
D	−	=	M	]
E	.	>	N	^
F	/	?	O	_

由 CASL 字符集可以看出字符的组成如下：

① 数字字符 0～9 共 10 个。

② 大写英文字母 A～Z 共 26 个。

③ 算术运算符、关系运算符、连接符及其他符号共 24 个。

全体字符合计共 60 个。

在 CASL 字符集中，列出了字符所对应的内码，所谓内码就是字符在计算机的代码。例如：

字母 A 在计算机中的内码为 $41_H=1000001_2=65_{10}$

字母 B 在计算机中的内码为 $42_H=1000010_2=66_{10}$

数字 0 在计算机中的内码为 $30_H=0110000_2=48_{10}$

数字 1 在计算机中的内码为 $31_H=0110001_2=49_{10}$

有了字符的内码，就可以在计算机中查询任何一个字符。借助内码我们还可以对字符信息进行筛选、变换等处理。

由字符集提供的内码可以看出，内码是一种 7 位码，它采用的是美国信息交换标准代码，即 ASCII(American Standard Code for Information Interchange)。

1.2.2　CASL 指令及结构

(1) CASL 指令种类

CASL 中有 3 类指令，即基本指令、伪指令和宏指令。

① 基本指令，是 CASL 汇编语言基本功能的体现，共 23 条。其中 21 条的写法有两种形式，一种形式为直接操作形式，另一种形式为间接操作形式。

② 伪指令，是对汇编编译程序进行说明的指令，这种指令在编译中并不产生目标代码。伪指令共有 4 条，所谓伪指令，从字面上说是“假指令”，其含义为非基本指令。

③ 宏指令，是一条指令可以启动一个程序段的指令。宏指令共有 3 条，宏可以理解为“大”，由于输入、输出等操作涉及到的部件较多，用一条基本指令概括不了，所以使用一个程序段来描述，为了使用上的方便，设定一条指令来启动这个程序段，担任启动的指令即为宏指令。

(2) CASL 指令结构

CASL 的基本指令，每条指令由下列 4 部分组成：

CASL 指令结构			
①	②	③	④
[标号名]␣	操作码␣	操作数␣	;注释
QS	LD	GR1,DY	;(DY)→GR1

①、②、③是指令的主体，它们之间在书写上要留出一个以上的空格：␣。注释是使用者所附加的指令功能说明，以备阅读。表中所列出的“QS LD GR1,DY”是一条取数指令，它的功能是取内存单元 DY 中的内容，放在寄存器 GR1 中。

在汇编语言表述中，为了简捷、方便，通常使用方括号、圆括号来表达、描述某个事项，并给它们赋予特定的含义，例如：

[ ]，方括号表示括号中的内容可以省略，也可以不省略，究竟是取是舍由使用者根据需要而定。

( )，圆括号表示其中的内容，如(DY)表示单元 DY 中的内容，(GR1)表示寄存器 GR1 中的内容，如果用两对括号，则表示“内容的内容”。例如，((GR1))表示 GR1 中内容的内容，同理((DY))表示 DY 单元中内容的内容。

① 标号名，从形式上看，它很像高级语言中语句的行号，但它并不表示指令的顺序，也不要求每条指令都加上标号名。只当该指令被其他指令调用时，才需要加上标号名。标号名实际上是该指令的标号地址，它描写的是该指令所在的位置。未加标号名的指令在程序中只体现出先、后的顺序位置。标号名的命名规定是，以字母开头，其后为字母或数字，长度为 6 个字母、数字之内。标号名中不允许用字母、数字以外的符号。

② 操作码，是 CASL 中规定的指令功能码，大都使用英文缩写，其目的是为了便于记忆。该功能码经汇编编译系统翻译后，在计算机中对应一条机器指令。例如 LD 是 LOAD 的缩写。

③ 操作数，从字面上讲，操作数往往被人们理解为“参加运算或操作的数据”。实际上，在汇编语言指令中，操作数这一项并不直接给出操作数本身，而是描述出操作数的来龙去脉，如与操作数有关的内存单元地址及存放操作数的寄存器等。例如在“LD GR，DY”指令中，操作数这一项所给出的是寄存器 GR1，作为操作数的临时寄存空间；DY 是内存单元地址，作为操作数的源发地。

④ 注释，它不属于指令的固有成分，注释是使用者附加的指令说明。按照 CASL 的规定，为指令加注释时，注释内容之前必须用分号，“；”用以与指令隔开。

下面再看几条指令书写的例子：

标号	操作码	操作数	注　释	功　能
[省]	LEA	GR1,6	;6→GR1	传送。将 6 传到寄存器 GR1。
[省]	ST	GR1,DY	;(GR1)→DY	存数。将 GR1 中的内容存于 DY 单元中。
[省]	ADD	GR1,DY	;(GR1)+(DY)→GR1	加法。GR1 中内容与单元 DY 中内容相加后送 GR1。

1.2.3 CASL 中的数

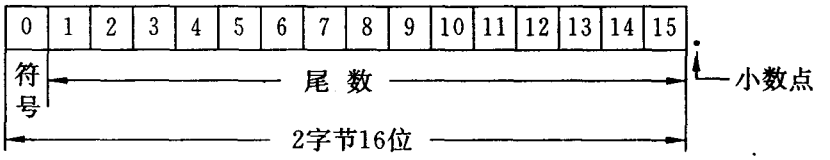
按照 CASL 的规定，在程序的指令中可以使用的数有 5 种，即有符号的十进制数、无符号的十进制数、十六进制数、字符常数和标号地址常数。

(1) 有符号的十进制数

在指令中直接写出十进制数，正数不加符号，负数加上符号，均为整数。

这种数在计算机中用二进制补码表示，一个数在 16 位的单元中，高位(第 0 位)为数

的符号，其他位(1～15 位)为数的尾数，小数点隐含在尾数的最低位之后。形式为：



从数的形式可以得出，指令中的十进制数为整数，在计算机中为补码，故数的范围为-2^{15}～$2^{15}-1$，即－32768～32767。例如：

指令中的数为＋1，则在计算机中为 0000000000000001；

指令中的数为－1，则在计算机中为 1111111111111111；如用十六进制表示计算机中的＋1，则为 0001_H；表示－1，则为 $FFFF_H$。

同理，若在计算机中得到的数为 $FFFE_H$，则该数为十进制的－2。“有符号的十进制数在计算机中为补码”在使用中要特别注意，这是汇编语言的一个难点。

(2) 无符号的十进制数

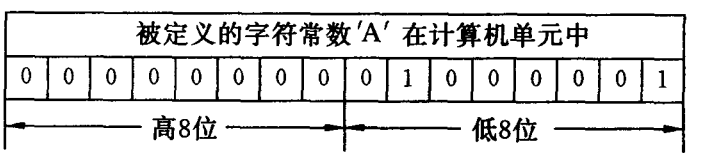
无符号数也称逻辑数，由于这种数在计算机 16 位的一个单元中全为数值，没有符号，所以数的最小值为 16 位全 0，即 0；最大值为 16 位全 1。因此数的表示范围为 0～65535，用十六进制表示数的范围为 0000_H～$FFFF_H$。在指令中使用这种无符号数时，注意不要越界。

(3) 十六进制数

在 CASL 汇编程序中可以使用十六进制数，使用形式为数的前面加上“＃”号，后跟 4 位十六进制数。例如＃0001，＃01AB，＃FFFF，＃1234 等都是符合要求的十六进制数。但是，这种十六进制数不能在基本指令中出现，只能定义在伪指令中。

(4) 字符常数

在 CASL 中的字符常数，其形式是由单引号将字符括起来而构成的。例如'A'，'ABC'，'1234'，'A1'，'B2'等都是符合规定的字符常数。字符常数只能在伪指令中定义，不能出现在基本指令中。被定义的字符常数在计算机中，一个字符占一个单元，每个单元的高 8 位为 0，低 8 位从最低位开始存储字符所对应的 7 位 ASCII，形式如下：



由存储形式可以看出，常数'A'在计算机的一个单元中只占用了低 7 位存放字符所对应的 ASCII，其余位均为 0。

(5) 标号地址常数

标号地址常数，即标号名。它是以字母打头的字母、数字串。标号地址常数的用途为，一是作内存单元地址的代号，二是作指令的逻辑地址。标号常数只当汇编语言源程序转为目标程序时，它才对应实际的内存地址。

1.2.4 一个完整的 CASL 程序

一个完整的 CASL 程序大都包括基本指令、伪指令、宏指令。下面是实现 1+2=? 的 CASL 汇编源程序。

	计算 1+2=? 的 CASL 程序			
	标号	操作码	操作数	注　　释
①	JJ12	START		;开始
②		LD	GR3,S1	;(S1)→GR3。
③		ADD	GR3,S2	;(GR3)+(S2)→GR3
④		ST	GR3,JG	;(GR3)→JG,
⑤		EXIT		;程序执行结束。
⑥	S1	DC	1	;定义十进制常数 1
⑦	S2	DC	2	;定义十进制常数 2
⑧	JG	DS	1	;定义存结果的内存单元
⑨		END		;结尾

程序中,②、③、④为基本指令,②LD 为取数指令,执行该指令可以取出 S1 所定义的常数 1 放于寄存器 GR3 中。③ADD 为加法指令,执行该指令可将 GR3 中的当前内容 1 与 S2 中的内容 2 相加,结果又存在 GR3 中。④ST 为存数指令,执行该指令可将 GR3 中的相加结果存于内存单元 JG 中。⑤EXIT 为宏指令,执行该指令,程序执行终止。指令①、⑥、⑦、⑧、⑨为伪指令,⑥、⑦为定义常数指令,⑧为定义内存单元指令。由于⑥、⑦、⑧分别为②、③、④指令引用,所以在指令中加上了标号名。指令①为程序开头指令,其标号为程序名。指令⑨为程序结尾指令。每个程序必须有①、⑨指令,用以说明本程序的上、下界。

习　题　1

1. 解答 CASL 与硬件背景有关的问题。
   (1) 在 CASL 中使用的寄存器有哪几个?
   (2) 通用寄存器在 CASL 程序中的作用是什么?
   (3) 标志寄存器有什么功能?
   (4) 寄存器 GR0 与其他寄存器在使用上有什么不同?
   (5) 程序计数器如何跟踪程序的运行?
   (6) 栈指针用什么来描述? 怎样描述数据进、出栈?
   (7) COMET 是一种什么样的计算机?
   (8) COMET 计算机内存有多大? 内存的地址范围是多少?

此为试读,需要完整PDF请访问: www.ertongbook.com