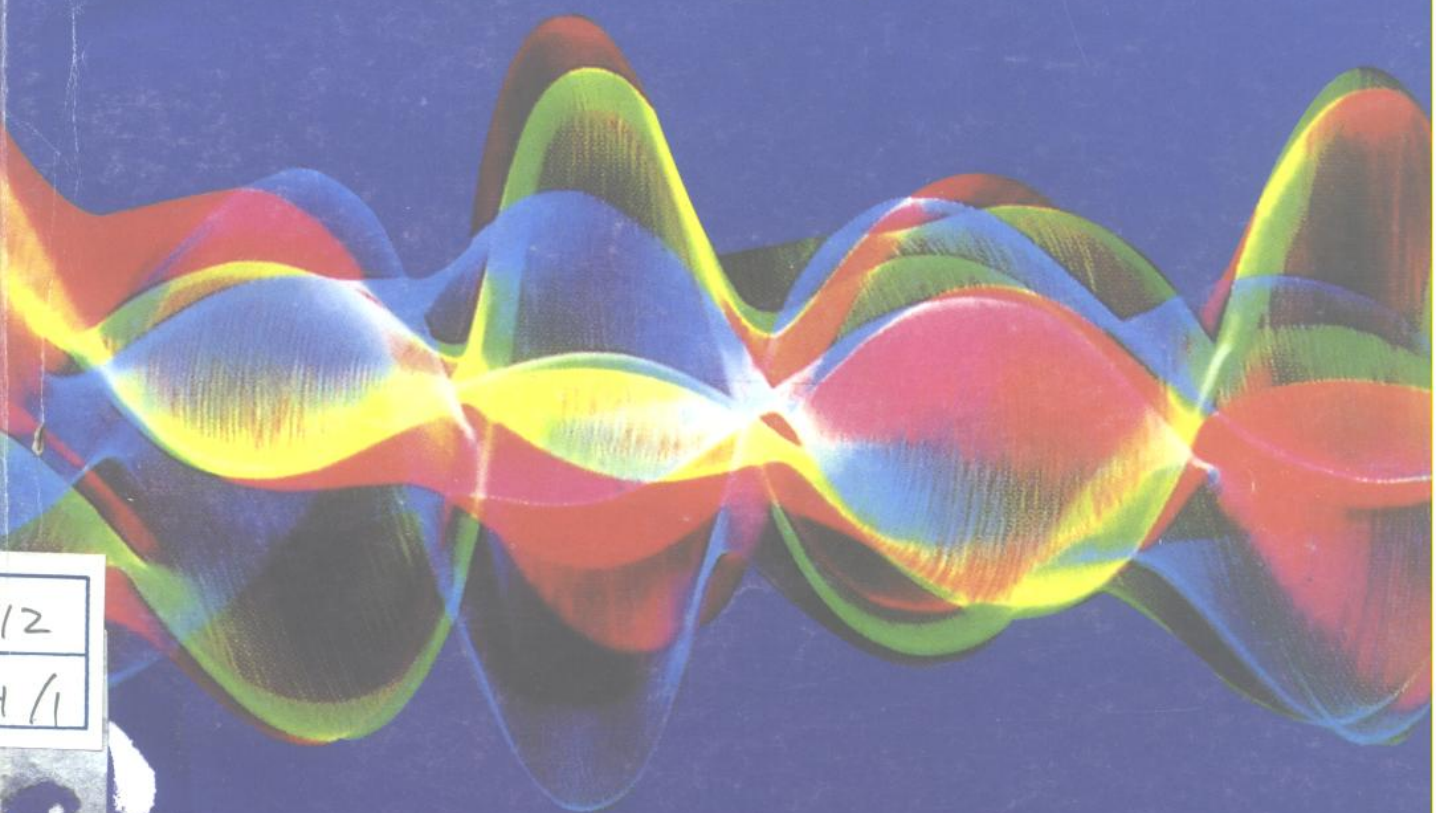
MS—FORTRAN 5.1

MS—FORTRAN 图形程序设计

游世辉 卿启湘 卿 钧 等编著



湖南大学出版社

MS-FORTRAN 图形程序设计

游世辉　卿启湘　卿　钧　等编著

湖南大学出版社
1996·长沙

内 容 简 介

本书全面介绍了 FORTRAN 语言目前最高版本的 Microsoft-FORTRAN5.1 的数据类型、I/O 控制、控制结构、数组与字符处理、子程序等与 FORTRAN77 完全兼容的升级部分语言；集成开发系统；MS-FORTRAN5.1 绘图的基本方法、图形库函数与子程序及应用。其次介绍了工程基本图形、二、三维图形技术、弹出式窗口与屏幕界面、西文图形下汉字显示等的应用图形程序设计。最后介绍了作者用 MS-FORTRAN5.1 开发的计算机画法几何学辅助教学的基本算法。本书全部程序都通过了 MS-FORTRAN5.1 集成环境下的编译、调试、运行。

本书可以作为高等院校计算机绘图的教科书，也可作为数值计算与工程仿真、计算机图形学与辅助设计、工程图学、FORTRAN77 的后续课程等相关学科的研究生、本科生和工程技术人员等的参考书。

MS-FORTRAN 图形程序设计

MS-FORTRAN Tuxing Chengxu Sheji

游世辉　卿启湘　卿　钧　等编著

责任编辑　张高明　卢　宇

☆

湖南大学出版社出版发行

（长沙岳麓山　邮政编码　410082）

湖南省新华书店经销　　湖南大学印刷厂印装

☆

787×1092　16 开　15 印张　347 千字

1996 年 9 月第 1 版　1996 年 9 月第 1 次印装

印数：1—5 000

INBN 7—81053—040—2/TP・5

定价：20.00 元

前　言

数据计算与工程仿真、计算机图形处理与辅助设计、工程图学的计算机辅助教学等学科都需要计算机绘图，为此，目前大多数高等院校都开设了计算机绘图课程。但目前在计算机绘图教学和实践中使用BASIC语言的还很多。由于BASIC语言的局限性，同时，由于FORTRAN语言广泛地使用在工科教学与上述工程应用中，所以，把FORTRAN语言用于计算机绘图的教学之中势在必行。

FORTRAN语言是1954年推出的第一个高级语言，以适用于数值计算而拥有广大的用户和丰富的应用软件。1966年美国标准化协会(ANSI)通过标准FORTRAN(3.9-1966)或称FORTRAN IV，1978年ANSI公布了FORTRAN77。1990年推出标准FORTRAN90，至此，FORTRAN语言仍然没有绘图功能。1990年MS公司推出了第一个带有图形库的MS-FORTRAN5.0编译器，但没有集成开发环境系统，使用起来很不方便。目前，MS-FORTRAN5.1是带有集成开发环境系统和图形库的与FORTRAN77完全兼容的最新的FORTRAN升级版本。

MS-FORTTAN5.1具有以下特点：

(1)强大的图形库，可支持屏幕和内存图形的操作；

(2)混合语言编程，可与C语言、PASCAL语言、BASIC等语言混合编程；

(3)带有集成开发环境系统，无需另写编译与连接程序；

(4)支持OS/2性能，可在DOS，Windows下运行；

(5)更丰富的数据类型、I/O控制及控制结构；

(6)更好的可移植性；

(7)与FORTRAN77完全兼容，结构化程序设计。

1995年在湖南大学自然科学基金的支持下，我室消化MS-FORTRAN5.1整个软件，将其用于开发计算机绘图与画法几何学计算机辅助教学的有关算法和程序，在总结其成果的基础上，写成了本书。

本书的全部程序都通过了运行，完整地奉献给读者。

本书可以作为高等院校计算机绘图的教科书，也可作为数值计算与工程仿真、计算机图形学与辅助设计、工程图学、FORTRAN77的后续课程等相关学科的研究生、本科生和工程技术人员等的参考书。

参加本书编写的人员有：游世辉、卿启湘、卿钧、刘泽玉、黄红武、周水庭、张爱军、黄素华、熊逸珍、戴立铃、李莉莉、尹小波。全书由本室集体审核。插图工作由卿钧教授承担。

本书的出版工作得到了湖南大学出版社的大力支持，在此表示衷心的感谢。

由于作者水平有限，书中不妥之处，恳请读者批评指正。

编　者

1996年9月

目 次

1 MS-FORTRAN 语言的基本知识

1.1 MS-FORTRAN 基本特征和元素

1.1.1 MS-FORTRAN 语言的主要特点

虽然 FORTRAN77 在数值计算领域比其它语言具备较多的优势，但是其图形处理功能较差，满足不了当今工程界的需要。由于工程计算和图形的紧密联系，特别是 CAD 有限元的图形前后置处理等需要，国内外许多大学、研究单位以及计算机厂商，为了满足各自的特殊需要，建立了许多 FORTRAN 图形库。但这些图形库数量少、功能弱、通用性和兼容性差，难以使用和推广，因此，美国 MS 公司以吸收众家之长为我所用的原则，于 1990 年推出了第一个带有图形库的 MS-FORTRAN5.0 编译器。MS-FORTRAN5.0 在已经普及的 MS-FORTRAN4.0 语言的基础上进行了许多新的改进，增加了一些重要的新特征，它与 FORTRAN77 完全兼容。

1991 年，MS 公司又在 MS-FORTRAN5.0 的基础上推出了 MS-FORTRAN5.1，增加了少量的内部函数和图形函数，并建立了集成开发环境系统。它集 FORTRAN 语言的编辑、编译、连接和运行于一身，创立了良好的用户界面，为初学者使用 FORTRAN 提供了方便。

目前正在酝酿新的 FORTRAN 标准，功能将有更大的扩充，成为能适应现代程序设计思想的结构化语言。

为了对 MS-FOTRTAN 语言有一个概括性了解，以下简述 MS-FORTRAN5.0 的主要特点。

MS-FORTRAN5.0 是运行在各种微机上的较新 FORTRAN 版本。它吸取了其它语言的优点，克服了早期 FORTRAN 语言数据结构少，语句格式呆板、控制结构不灵活、编译器灵活性差和不支持图形等缺点，在已普及的 MS-FORTRAN(V4.0)语言的基础上增加了若干重要的功能：

(1)第一次提供了 FORTRAN 图形库，库中包括丰富的绘图函数和字体输出函数；

(2)支持所有 IBMSAA 扩展，包括许多 VAX 扩展，SAA 和 VAX 上的 FORTRAN 应用程序可以方便地移植到 MS—DOS 支持的微机上；

(3)支持 OS/2 系统，包括动态连接库和多流执行。程序可以在 DOS 和 OS/2 下运行，具有很好的灵活性；

(4)允许对源程序进行条件编译；

(5)提供了丰富的用户接口。可以方便地进行 C，PASCAL，BASIC 等混合语言编程；

(6)提供了某些新的数据结构，如复合数据类型(结构)；

(7)提供了若干新的控制结构，如 SELECT CASE 分支语句和 DO WHILE 等；

(8)面向窗口的调试工具 CODE VIEW，不仅适用于代码程序的调试，而且适用于 FORTRAN 源程序的调试；

(9)增加了许多新的函数和过程,如地址操作函数、位操作函数、时间和日期过程等。

目前,MS-FORTRAN5.1 是 MS-FORTRAN 的最新版本。它除了拥有 MS-FORTRAN5.0 的主要特点之外,它的突出优点就是建立了 MS-FORTRAN5.1 集成开发环境系统,创立了良好的用户界面,为用户带来了便利。

1.1.2 MS-FORTRAN 语言的书写格式

MS-FORTRAN 语言的书写格式基本上与 FORTRAN77 相同,但还有以下几点区别:

(1)MS-FORTRAN 可以以感叹号开头,不在第 6 列且不在字符常量中的感叹号可以作为注释行的开头。注释行可出现在一行的语句后,计算编译时,对该区的内容不作加工。

(2)在FORTRAN 程序行中,字符的位置是很重要的。FORTRAN 有四种很重要的行,即注释行、起始行、续行和结束行。MS-FORTRAN 除上面所列的四种行外,还有元命令行。

a)元命令行:第一列是书写 $ 字符的行,其后紧跟元命令,元命令控制 MS-FORTRAN 编译程序的操作,如 $DO66 即元命令行。元命令行不允许有续行,且元命令中不允许有空格。

b)元命令:以美元符号 $ 开头,在源程序中必须单独占一行,且从第 1 列开始,元命令行中不允许有其它字符或在元命令中加插空格。

编译元命令的使用是引导编译系统以某种特定方式处理 FORTRAN 源文件。它广泛用于程序的调试,跟踪或增强运行文件的某些功能,因此,正确使用编译元命令可以有效地缩短复杂程序的调试周期,提高运行程序的质量。FORTRAN77 没有此项。

(3)续行不能带语句标号。MS-FORTRAN V4.0 版本以下的编译系统对续行数有限制,一个语句最多有 19 个续行,即一个语句最多占 20 行。在一般情况下,MS-FORTRAN V4.0 版本以上的编译器对此没有限制。

注释行无续行。一行写不完时,可以写在第二行,但第二行第一列必须有注释行的标志。

(4)元命令 $FREEFORM 使 FORTRAN 程序源代码使用自由格式,$NOFREEFORM 说明采用标准 FORTRAN 格式的源程序,其缺省值也采用标准 FORTRAN 格式。

1.1.3 MS-FORTRAN 语言的语句顺序

语句是用来描述、指定、分类程序的元素以及程序将采取的动作,在 FORTRAN 程序中,各种语句先后出现次序都有一定要求,正确理顺各种语句之间的关系至关重要。这点务必引起我们的重视:

每个程序单元都必须将 END 语句作为最后一行。

注释行可以出现 END 语句之前的任何地方。

Block,data,function,interface to,program 和 subroutine 语句必须在所有其它语句之前,这些语句不一定要位于元命令之前。

说明语句必须在 data 语句、语句函数语句和可执行语句之前。

除了 program 语句之外,implicit 语句必须位于其它说明语句之前。

当说明语句定义一个将用于 parameter 语句中的常量的类型时,parameter 语句必须紧跟在该说明语句之后。parameter 语句必须在所有使用它所定义的符号常量的说明语句之前。

Interface to 语句必须位于所定义的子程序的引用之前。

如果 $DO66，$FLOATCALL，$NOFLOATCALL，$FREEFORM，$NOFREEFORM 和 $NOTLARGE 元命令出现，必须位于其它任何语句或元命令之前；当 $LARGE 和 $NOTLARGE 元命令不带参量时，不能在可执行语句部分出现，当带参量时，必须在其说明部分出现。其它元命令可出现在任何地方。

数据块子程序不能含有语句函数、FORMAT 语句或可执行语句。

1.1.4 MS-FORTRAN 的基本元素

MS-FORTRAN 程序的基本元素：FORTRAN 标识符的命名与作用域；数据类型及使用规则；常数、变量、属性及结构；各种运算符和表达式的运算规则及计算结果的数据类型转换等。

MS-FORTRAN5.1 的元素与 FORTRAN77 基本相同，但还有下面的一些区别。

1.1.4.1 结构类型

结构类型是 MS-FORTRAN5.0 以上版本允许用户定义的又一新的数据类型。它的使用使得在 FORTRAN 中的信息组织和表示更加方便、灵活。

一个结构是两个或多个有关数据项的汇集。而每个数据项可以具有不同的类型，我们可以使用一个结构存贮一个人的有关所有信息：名字、婚姻状况、年龄、出生日期等等。为了处理这些不同类型的量，MS-FORTRAN5.0 提供了几种确立构造数据类型的途径。

结构：它是一种在同一名字下各种不同类型变量的组合。结构中的每个项称为“结构元素”。

记录：是一个结构变量，用来访问结构中结构元素。

映象：指定一个或多个变量在内存中连续存放。

联合：它使得两个或两个以上不同变量类型共有一内存块。

(1)结构

在FORTRAN5.1 语言中，结构是一种被命名为一个标识符的各种变量的集合。这种方法为将各种信息汇集在一起提供了非常有利的途径。结构定义(structure definition)确立了结构变量的格式。结构中的所有元素通常彼此有逻辑关系。如下面所举的关于人的有关信息的例子。

结构定义的一般形式为：

```
STRUCTURE/结构名/
类型          变量名
类型          变量名
 ⋮             ⋮
类型          变量名
END STRUCTURE
```

以下面一个雇员的信息结构为例，说明结构的一般定义形式：

```
STRUCTURE/employee _data/
CHARACTER * 20          street _name
CHARACTER * 2           street _number
```

```
CHARACTER * 2          apt__number
CHARACTER * 20         city
CHARACTER * 2          state
INTEGER * 4            zip
INTEGER * 4            telephone
INTEGER * 2            date__of__birth
INTEGER * 2            date__of__hire
INTEGER * 2            social__security(3)
LOGICAL * 2            married
INTEGER * 2            dependents
END STRUCTURE
```

结构定义并没有说明实际的变量。它仅仅是一种结构的形式定义，说明组成这些变量的形式和特性。我们定义一个结构时，实质上是将结构元素的各种复杂的变量形式结合在一起。

结构说明不是变量，而是变量类型。结构变量用 RECORD 语句来定义。一个结构可以具有一个或多个结构变量。结构变量定义的一般形式为：

RECORD/结构名/结构变量名

例如：下面的记录语句定义了保存雇员数据的结构变量数组：

```
RECORD/employee-data/employees(100)
```

employees(100)是由 RECORD 语句定义的属于结构 employee-data 的一个结构变量数组。

(2)访问结构元素

对结构的使用是通过对其元素引用来实现的，不能把结构作为过程变量在程序单元间访问，也不能象简单变量赋值那样把一个结构的内容直接赋给另一个结构(即不能把结构作为一个整体来存取)，而只能对其元素的个别引用。这种引用是通过操作符圆点“.”来实现的。例如，第 88 个雇员的抚养人数 dependents 由 employees(88).dependents 来指定，这个雇员的电话号码由 employees(88).telephone 来指定。

结构定义可以包含结构说明。

(3)映象与联合

在 MS-FORTRAN5.0 中，映象是对 structrue 语句中的一组变量类型说明进行分界。它指定一个或多个变量在内存中的连续存放。变量可为任何常规类型，包括结构变量。例如：

```
MAP
    INTEGER * 4   many,moe,jack
    CHARACTER * 21   myname
END MAP
```

在例子中，4 字节的整数 many，moe 和 jack 是连续存储的，以它们在 MAP 语句中的次序排列，后面直接跟着 21 个字符的变量 myname，每个变量的串地址是由 $PACK 元

命令或/ZP 命令行选项决定的。

下面是映象的另一例子。

```
MAP
    CHARACTER * 7   yourname
    REAL * 4   meg,joe,amy,zelda
END MAP
```

映象即 MAP…END MAP 块必须出现在联合即 UNION…END UNION 块中。联合是把两个或更多的映象赋以相同的内存，不象 FORTRAN 中的 EQUIVALENCE 语句，它只覆盖一变量或数组。一个联合中至少有两个映象。例如：

```
UNION
  MAP
    INTEGER * 4 many,moe,jack
    CHARACTER * 21 myname
  END MAP
  MAP
    CHARACTER * 7 yourname
    REAL * 4 meg,joe,amy,zelda
  END MAP
END UNION
```

假定 $PACK：1 有效，7 字节的 yourname 将覆盖 4 字节的 many，以及 moe 的前三个字节，meg 的第一个字节将覆盖 moe 的最后一个字节，等等。其它情况依此类推。有关 $PACK 更多的信息请看第二章有关内容。

1.1.4.2 属 性

属性是 MS-FORTRAN5.0 特有的，是 FORTRAN77 的一种扩展。它指定了变量、变量类型、子程序和子程序形参的附加信息。

如属性允许 FORTRAN 程序使用 Microsoft C 和 PASCAL 的命名约定和调用约定，通过值或引用参量，使用分段的或不分段的地址，指定了一个形参能够跨越多个段，或者指定一个子程序或公共块外部名。属性也可以用在子程序、函数定义、类型说明中，还能与 INTERFACE TO 和 ENTRY 语句一起使用。有关详细信息请见下面的讨论。

属性的一般定义格式：

符号名［属性］

属性紧跟在它们所引用的对象之后，若指定多个属性，则属性之间必须用逗号隔开。例：

```
INTEGER intvar[REFERENCE,NEAR]
```

(1)ALLAS 属性

ALLAS 属性用于字符数的说明。我们知道,每种语言能识别不同数量的字符,MS-FORTRAN 能识别任何名字的前 31 个字符,但若使用元命令 $ TRUNCATE 则仅能识别前 6 个字符;PASCAL 能识别前 8 个字符,而 BASIC 能识别前 40 个字符,也就是说,若某个符号名的字符超过了每种语言可以识别的最大长度,则超出的字符将被忽略。还有 FORTRAN,PASCAL 和 BASIC 使用了大致相同的命令约定,每个字符均被换为大写,但 C 语言不同,C 编译器不将任何字母转换为大写,只是在每个子程序的符号名开头插入一条下划线。基于以上两点,FORTRAN 使用了 ALLAS 属性,增强了 FORTRAN 的综合编程能力和灵活性。

ALLAS 属性的作用和应用范围如下:

a)FORTRAN 中的接口语句即 INTERFACE TO 语句可使用 ALLAS 属性,使各种语言的命名约定兼容;

b)ALLAS 属性覆盖 C 属性;

c)若使用了 $ TRUNCATE 元命令的 FORTRAN 程序中又使用了 ALLAS 语句,元命令 $ TRUNCATE 的作用失效;

d)ALLAS 不能用于形参。

(2)EXTERN 属性

EXTERN 属性用于变量说明,它表示该变量将位于另一源文件中。当访问其它语言所说明的变量时,必须使用 EXTERN。

EXTERN 也不能用于形参。

(3)ALLOCATABLE 属性

ALLOCATABLE 属性用于数组说明,指定一个数组是可分配的,即数组的每维大小是在运行时刻动态地建立的,而不是在编译期间建立的。其应用规则如下:

a)既可用于类型说明,又可用在 DIMENSION 语句;

b)可分配数组的每维由一个冒号说明,如一个可分配的三维数组 dynamic 说明如下:

```
REAL * 8 dynamic[ALLOCATABLE](:,:,:)
```

c)不能用于形参,也不能用于 NEAR 属性的参量。当用户预料一个可分配的数组大小将超过 65536 个字节,则必须将数组说明为 HUGE 属性,以便产生正确的地址。

存储器是计算机系统的重要资源之一,因为任何程序代码数据项都必须占用一定的存储空间,因此,存储的管理直接影响到系统的性能。

存储器由内存(Primary Storage)和外存(Secondary Storage)组成。内存由顺序编址的块组成,每块包含相同的物理单元,内存要通过启动相应的输入/输出设备后才能使外存与内存交换信息。以下主要讨论内存模式的选择。

通过指定程序的内存模式,可以更好地控制程序使用内存的方式。内存模式是一个预定义的规则集,编译器按照这些规则将程序代码和数据转换成内存中的 64k 段(内存块)。内存模式定义了编译器将代码和数据组织进段的方法,同时还规定了访问每段的代码和数据寻址方法(近程、远程或巨型)。近地址是个 16 位的偏移量(相对于段的头),它最多能

访问 64k 的内存；远地址或巨型地址是一个 32 位地址，它可以访问所有可用内存。

选择内存的模式，即是告诉编译器，它可以对程序的特性作某些假定，并产生相应的机器代码。例如，当选择大内存模式时(FORTRAN 的缺省模式)，编译器将允许程序带有 64k 以上代码和 64k 以上数据；当选择中模式时，编译器只能产生 64k 以下的数据，并产生访问数据项的地址。把一个程序装入内存时，其编译后的可执行语句即代码程序和所使用的信息(即数据项)被分别置于独立的存贮区域中，处理器把保存在内存中的代码当作将执行的操作序列，通过调用子程序或函数，一个程序可以对其它代码进行引用。存贮的数据是程序所需要的值，内存中为这些数据保留了存贮空间，包括变量名、数组或公共数据块。所有处理器均可区分代码和数据，代码和数据分别占据了自己的段，并通过自己的段寄存器进行访问。但程序的代码和数据的存贮与访问则取决于所用的处理的结构和寻址方式。

8086 处理器和相关的其它处理器都是 16 位机器。通常，使用 16 位地址的机器只能访问 64k 的内存，为了扩大程序能寻址的内存范围，8086 的物理地址被分成若干段，每段 64k 长，内存中每端的起点用一个 16 位地址表示。

a)近地址。8086 保留了四个寄存器来存贮段的基地址，这四个寄存器是：CS(代码段)，DS(数据段)，SS(栈段)和 ES(附加段)。但是段地址只是指向段的起始地址即基地址，为了引用段内的某个项，还必须给出该项的段内地址，这就需要一个 16 位偏移地址。它给出了一个项相对于段的开头的偏移量，16 位偏移量就称为“近地址”。

只要有可能，编译器总是产生近地址来访问代码和数据项。近地址要比远地址的效率要高得多，这是因为它们只需要较少的空间，且只占用较少的时间来计算地址。

b)远地址。用近地址来访问代码和数据项远远不够，因为许多程序包含了超过 64k 的代码和数据，这就需要引入远地址的概念。8086/286/386 上的全地址要求 32 位，16 位段地址和 16 位的偏移量，全 32 位地址称为远地址。

当程序代码超过 64k 时，它要占用一个以上的代码段。由于代码的地址必须包括段地址和偏移量，因此对于子程序或函数的调用都要求编译器使用远地址，这样就增加了程序的长度，降低了效率，但允许程序突破 64k 的限制。

类似地，带有超过 64k 数据的程序必须占用一个以上段的数据，编译器也必须产生全部 32 位远地址。

当程序含有多个数据段时，某些数据项放在缺省数据段(内存限制为 64k 的段)，由 DS 寄存器寻址，编译器只需要产生 16 位偏移地址来访问这些项(因为段地址由 DS 决定)，但是对缺省数据段之外的数据项，编译器必须产生全部 32 位远地址。远地址允许程序访问大量数据。

c)巨地址。当程序有非常大的数据项时，即程序可能会包含一个超过 64k 的单个数据项(数组或公共数据块)，为了在超过 64k 的数据项内访问元素，编译器必须用 32 位来计算地址。在 MS-FORTRAN 中，大于 64k 的单个数据项被称为“巨型”数据项。这种项的地址为“巨型”地址，即访问数据项中的个别元素需要 32 位偏移量。

只有数据项才具有巨型地址。每个模块的代码被限制在 64k 以内，所以它永远不会超过 64k 内存块。巨型地址允许程序访问很大的数据项。

NEAR,FAR 和 HUGE 是 MS-FORTRAN 程序中使用的关键字,它们用以覆盖缺省寻址约定。

(4)NEAR 属性

NEAR 属性指定实参存储在缺省数据段,并只将其偏移量传送给子程序,此属性可用于公共块,有 NEAR 属性的公共块被映射到缺省数据段中。

语法:common/公共块名[NEAR]/…

注意,若公共块名缺省,则所有无名公共块都放入缺省数据段。

传递到近形参的实参必须在缺省数据段中,下列数据均不能传递给近形参:

(a)公共块中未说明为 NEAR 的数据;

(b)定义为 HUGE 的数组;

(c)$LARGE 元命令有效时定义的数组;

(d)在 $LARGE 元命令中命名的变量。

(5)FAR 属性

当用于形参时,FAR 属性指定该参量使用分段地址,即远地址来传递;当用于变量时,它说明该变量位于远程数据区。

(6)HUGE 属性

HUGE 属性是指定一个形参或一个可分配的数组可能跨越一个段的简便方法,它与 $LARGE 元命令作用相同。例如:

```
FUNCTION FUNC(a [HUGE])
DIMENSION a(200)
    或
$LARGE : a
FUNCTION FUNC(a)
DIMENSION a(200)
```

NEAR,FAR 和 HUGE 属性与选择一种标准内存模式相比,它使程序具有更大的灵活性。例如:当程序中的大小可调整或大小假定的数组所占的内存实际大小超过 64k 时,如果将这些内存说明为 HUGE 属性,那么程序可以避免从大模式切换到巨模式。另一方面,即使程序要求巨模式,通过把频繁访问的数据项放在缺省数据段,用以改进程序的效率。

当程序有许多适合于放在缺省数据段的数据项时,也应该使用 FAR 属性来说明这些数据项。访问频率较低的数据项可以说明为 FAR 属性,以便将它们移出缺省数据段,从而为使用频率高的数据项保留更大的空间。

(7)C 属性

C 属性可用于子程序、公共块和类型说明语句中。当用于一个子程序时,C 属性定义该子程序具有与 Microsoft C 过程相同的调用约定。有关 FORTRAN 调用约定和 C 调用约定的详细信息请参阅 1.6 有关内容。