IBM PC／AT

硬件和XENIX系统资料汇编之十一

FORTRAN语 言

白为民 译

陆拓实 校

孙玉方

北京科海总公司培训中心

中国科学院软件研究所

一九八七年四月

IBM PC／AT硬件和XENIX系统资料汇编

（之十一）

FORTRAN 语 言

白 为 民 译
陆 拓 实 校
孙 玉 方

北京科海总公司培训中心
中国科学院软件研究所
一九八七年四月

编 者 序

IBM PC已从PC、PC/XT推进到PC/AT。PC/AT以Intel 80286为主CPU，具有丰富的硬件资源。鉴于目前DOS系统基本上是一种单用户系统，许多硬件资源未得到充分利用，许多用户都要求在PC/AT上配备多用户多任务的XENIX系统。

XENIX系统是UNIX系统在以Intel为主CPU的微机上的实现，该系统由Microsoft公司开发。目前在PC/AT上运行的XENIX相当于UNIX的System Ⅲ或System Ⅴ。

为了更好地在国内推广PC/AT及其兼容机，中国科学院软件研究所在其雄厚的技术力量基础上，积多年研究、开发UNIX系统之经验，开发成功多种XENIX中西文信息处理系统并移植到几乎所有PC/AT的兼容机和部分386机上。为了更好地推广XENIX中英文信息处理系统，科海培训中心和中国科学院软件研究所组织了一批专家和技术人员，收集并编译整理了有关XENIX及IBM PC/AT的全部技术资料。

这些资料包括以下几类：

Ⅰ. IBM PC/AT硬件资料，包括硬件安装及组装手册、技术手册和维护手册。

Ⅱ. XENIX基本系统的安装、基本用户指南、命令参考手册、系统管理手册和直观shell手册。

Ⅲ. XENIX开发系统方面的软件开发手册、库函数程序员手册、系统调用和子程序手册。

Ⅳ. XENIX系统上运行的汇编语言和各种高级语言（C、Fortran、Cobol、Basic）的用户指南和参考手册。

Ⅴ. XENIX正文格式化处理手册。

Ⅵ. XENIX系统上配备的最新版本INFORMIX和UNIFY数据库管理系统用户及参考手册。

Ⅶ. 中西文兼容的C—XENIX系统安装和基本使用手册。

全套资料约400万字，分装成20本。

全书的主要译校任务由中国科学院软件研究所的专家、技术人员承担，科海培训中心负责编辑、印刷和发行工作。

由于时间仓促，本资料汇编中必有不少错漏之处，敬请读者批评指正，以便再版时更正。

主编 孙玉方

董洪皋

Microsoft FORTRAN
编译程序用户指南

白为民　译
陆拓实　孙玉方　校

目　录

第一章 引 言

1.1 有关本手册

Micosoft的FORTRAN编译程序用户指南的对象是熟悉XENIX操作系统并且具备一定的FORTRAN知识的程序员。本手册说明如何建立、编译、链接FORTRAN程序，也描述了如何在程序中结合进PASCAL,C和汇编语言模块，以及如何用XENIX的ar和ranlib命令创建你自己的库。

如果你对FORTRAN语言及它的扩展有疑问，请查阅《Microsoft FORTRAN 参考手册》，有关FORTRAN程序设计的进一步信息，见1.3节“FORTRAN的进一步学习”。

本手册分成以下几章：

第一章“引言”；

第二章“开始工作”解释了如何在你的XENIX系统中装入Microsoft的FORTRAN编译程序，并包括一小段编译和链接过程的演示。

第三章“开发一个程序”，讨论了程序的开发方法，包括编写大程序以及使用浮点算术运算的一些建议。

第四章“编译一个FORTRAN程序”，描述了如何编译FORTRAN语言源文件，如何产生可重定位目标文件，如何使用编译任选项以及如何产生源程序和反汇编清单。

第五章“链接目标模块指明了如何用cl命令链接先前编译好的目标文件，如何给可执行程序取一个除缺省名外的名字，如何链接程序与库中的函数。同时也解释了如何设置栈的大小以最有效地使用内存，如何创建一个装入得更快的程序，并通过从可执行文件中去掉符号表而使程序所占的磁盘空间更少。

第六章“库管理”描述了如何创建及管理你自己的目标模块库。

第七章“使用汇编语言子程序”描述了如何将80286汇编语言子程序与Microsoft FORTRAN程序一起使用。

第八章“混合语言程序设计”解释了如何在FORTRAN程序中用PASCAL和C语言子程序。

第九章“用adb调试”解释了怎样用adb控制一个程序的执行，以及怎样检查和修改一个可执行文件。

本手册末尾的附录提供了有用的参考资料。附录A“命令小结”是有关编译、链接、建库命令的完整汇总。

附录B“Microsoft FORTRAN的改进”包含有关这个新版本的Microsoft FORTRAN的信息。

附录C“初始化与终止”解释了如何建立你自己的初始化与终止过程。

附录D“磁盘内容”列出了Microsoft FORTRAN磁盘产品上的所有文件。

附录E“80287浮点错误的处理”描述了如何处理由浮点错误造成的问题。

附录F“出错信息”列出了所有由编译程序运行时的系统、链接程序及ar和ranlib命令发出的出错信息，并描述了如何纠正这些错误。

词汇表定义了许多本手册中使用的术语。

1.2 符号约定

斜体字 斜体字用于命令行说明中，标志在该处有一个特定的词出现在命令中，例如：

cl *filename*

这里*filename*是斜体字，表示这是命令的一般形式，在cl命令实际使用时，用一个别的名字来取代*filename*。

斜体字也用于程序名、变量名、函数名及过程名。例如，*myprogf*程序名就是以斜体字出现的。

[括号] 括号把任选的命令行参数括起来。

省略号 跟在某一项后面的省略号表明有更多同样形式的项出现。如：

cl [*option*…] *filename*…

这里省略号说明你可以给出不止一个选择（当然你也可以不作任何选择，因为*option*是在括号内的）以及不止一个的文件名。

大写字 大写字母用于语句（如INTEGER X）及元命令中，如$NODEBG。这个约定仅适用于本文中，你不必用大写字母输入这些字符，Microsoft FORTRAN对大小写没有限止。

1.3 FORTRAN的进一步学习

如果你是初学FORTRAN或对进一步学习程序设计有兴趣，你可以阅读下面这些书：

Agelhoff, R.,and Mojena, Richard “Applied FORTRAN77, Featuring Structured Programming”

Friedman, F.,and E.Koffman,“Problem Solving and Structured Programming in FORTRAN”

Wagener,J.L. “FORTRAN 77: Principles of Programming”

第二章　开始工作

在建立你的可执行程序之前，必须将Microsoft FORTRAN编译程序装入你的硬盘。本章解释如何在你的XENIX系统中装入编译程序，装入过程大约需10分钟。

编译程序装入结束后，你就可以用2.2节练习部分中的例子来编译、链接和运行实际的FORTRAN程序，练习部分也向你表明如何产生源代码清单以及如何命名编译程序产生的可执行文件。

2.1 FORTRAN编译程序的装入

自动执行装入过程的Shell程序在第一块编译盘中，这一节解释如何把编译程序装入到硬盘中去。

为了装入Microsoft FORTRAN编译程序，你必须具有超级用户权限。按以下步骤装入软件：

1.用root注册。

2.用cd命令把你的当前目录改为/*tmp*：

```
cd /tmp
```

3.把第一张FORTRAN盘（标有1 of 4）插入顶上的软盘驱动器并关上驱动器门，用tar命令从盘上提出装入程序msinstall

```
tar xvf /dev/fdo48ds9 msnstall
```

注意：

下一步要执行的msinstall程序，要替换许多库、C包含文件和以下程序：

```
adb
ar
ld
ranlib
```

如果你对上述内容进行了修改，检查一下附录D“磁盘内容”的所有文件清单，确认没有毁掉任何你想保留的内容。msinstall程序将对附录D中列出的文件全部重写。

4.键入

```
/tmp/msinstall
```

并按下回车键，以执行装入程序。msinstall程序从盘上取出合适的文件把它们放在适当的目录下。

5.当你看到信息：

```
Are you ready to begin installation [y,n] ?
```

键入“Y”，按下回车键，下一个提示接着问：

```
first disk?(第一张盘吗?)
```

如果你已拿走了第一张盘，把它放回驱动器中，键入“Y”和回车键。

6.每一张盘拷贝回后，都会提示你换盘，当驱动器的灯灭了的时候，你看见提示信息：

```
Next disk(y, n)?
```

插入下一张盘，键入“Y”和回车键，对所有的软盘都进行这样的操作。当全部软盘装完后，在“Next disk”提示下，键入“N”和回车键，你可以看到这样的信息：

```
Installat,on Complete 装入完毕
go on to the next step 进行下一步
```

7.键入

```
rm msinstall
```

删除装入程序

这样Microsoft FORTRAN编译程序就可以使用了。把最后一张盘从驱动器里取出，并把所有盘保存在一安全地方。如果你的硬盘被毁或文件被删，你还要用这些盘重新装入编译程序。

装入过程在/*tmp*中留下了一个叫做README的文件，这个文件包含有本FORTRAN版本的许多重要信息，建议你将这个文件移到一个永久位置，使用编译程序前先读一读它。

2.2 练习部分

在这一节中，你将用Microsoft FORTRAN编译程序来：

- 建立一个源文件*sample.f*；
- 建立一个可执行文件*a.aut*，这个文件是通过编译和链接*sample.f*得到；
- 运行*a.aut*；
- 产生*sample.f*源代码清单；
- 给可执行文件一个名字，而不用缺省名字*a.aut*。

2.2.1 建立FORTRAN源文件

首先键入一小段程序以进行编译。用正文编辑器，建立一个名为*sample.f*的文件，并键入以下程序：

```
program sample
write (*, *) 'Hello world!'
stop
end
```

这段程序，编译运行后，将在显示屏上显示：“Hello world!”。

2.2.2 编译和链接FORTRAN源文件

cl命令用一个命令编译和链接你的程序。为从刚建立的源文件得到一可执行程序，在系统提示符下键入：

```
cl sample.f
```

这就同时调用了Microsoft的FORTRAN编译程序和XENIX链接程序。编译分两遍进行，第一遍建立第二遍要读取的中间文件，第二遍建立可执行程序。

cl命令控制编译程序和链接程序。cl的变量允许你命名可执行文件或存取除FORTRAN标准库以外的其他库。这些选项在第四章“编译一个FORTRAN程序”和第五章链接“

目标模块”中讨论，并且在附录A“命令小结”中列出。有关每一遍的进一步信息，见4.1节“cl命令怎样工作”。

编译结束后，你的程序自动就与FORTRAN标准库链接，可执行程序放在你当前目录下的一个叫作*a.out*的文件中。

2.2.3 运行你的程序

编译与链接过程结束后，你就可运行你的程序，并且XENIX的系统提示符重新出现。要运行你的程序，键入：

```
a.out
```

按下回车键，下面的句子就出现在你的屏幕上。

```
Hello world!
```

第三章　开发一个程序

本章介绍程序开发并讨论两个特别问题：大程序下的工作和用浮点选择。

3.1　程序开发

程序开发包括四个步骤：写程序、编译、链接和运行，进行最后三个步骤的时间分别称为“编译时间”，“连接时间”，“运行时间”，这个顺序有时也称作“编辑—编译—执行”循环，在一个程序完成以前，通常要重复多次。

第一步，写源程序，可以用任何正文编辑器来完成，包含有Microsoft FORTRAN程序的文件叫做FORTRAN“源”文件。

程序开发的第二步是编译，Microsoft FORTRAN编译程序将你源文件中的FORTRAN语句翻译成微处理机能执行的指令，称为“机器代码”，如果编译成功，编译程序将建立一个称为“目标”文件的文件。目标文件中包含有可重新定位机器代码，也即这些目标码可与其它目标文件连接起来，形成可执行程序。

编译程序也执行以下功能：

1。在读入你的源文件时标出所有的语法错误。

2。对那些运行时可能出现的错误进行检查，只有在你的源文件中包含有 $DEBUG元命令时才进行这样的检查。在第一次编译时，用$DEBUG元命令，是一种好的程序设计风格，有关$DEBUG的进一步信息，见Microsoft FORTRAN参考手册。

3。查找变量与GOTO语句的目标，把它们与内存地址联系起来，这样在你的程序执行时，就能减少时间。

4。优化，Microsoft FORTRAN编译程序以“优化编译”而闻名，它通过剔除冗余，记录表达式，使用专用机器指令等把你的程序翻译成体积更小而执行速度更快的机器代码。

程序开发的第三步是链接，一旦程序编译完毕，目标代码必须与FORTRAN库中的运行与程序链接，这些子程序为你的程序提供诸如算术运算、字符操作及输入／输出等服务。在运行的时候，你也可以与其他独立编译的目标模快连接。

链接程序将所有你规定的程序链接起来，并产生可执行程序文件，可执行程序文件的缺省名字是*a.out*，要执行一个可执行程序文件，只要打入文件名即可。

程序开发的第四步是调试。这一过程的任何一点你都可能碰到错误或需要修改程序，不管你何时这样做，你又返回到第一步，编辑源文件并重复这个循环。在你成功地编译、链接和运行你的程序后，编辑你的源文件，将$DEBUG元命令改成 $NODEBUG 元命令，然后，重新编译、链接、运行，用$NODEBUG元命令将减少你的可执行程序的大小与运行时间。

3.2 大程序下的工作

在大程序下工作，你或许会有这些问题：

1．太多的代码（源代码、目标代码或可执行代码）；

2．太多的数据；

3．编译时内存不够。

第3.2.1与第3.2.2节将讨论如何解决这些问题。

3.2.1 代码和数据受限制下的工作

当你用XENIX和Microsoft FORTRAN的时候，对代码和数据有以下这些限制：

代码或数据	限制
源代码	32,767行
目标代码	每个源文件64K字节
数据和可执行代码	约1.3M字节

实际上，有32,767行长并生成64K字节目标码的源文件是很难编辑和管理的，把大程序分解为一些子程序、函数，并对它们分别编译是一种好的程序设计风格。因为你可以对你的源文件独立地进行编译，然后再链接，所以对代码大小的实际限制不是64K字节和32767行了，而是你的操作系统（1.3M字节）和计算机。

例如，你可以分别编译6个50K字节的不同的源文件，把它们链接起来产生一个总量达300K字节的程序。不过，编译不止一个源文件的程序会比同样的程序作为一个源文件进行编译产生更大的目标文件。

有几种减少你的程序大小的方法：

• 在你的源程序中用$NODEBUG元命令。编译一个包括$DEBUG元命令的源文件产生的代码最多可比不使用该命令多40%，一旦你的源文件能成功运行，用$NODEBUG替代$DEBUG，然后重新编译、链接，产生一个较小的可执行程序文件。

• 用$STORAGE：2元命令。

$STORAGE元命令规定INTEGER、LOGICAL变量是分配两个还是四个字节的内存，如缺省的话，就分配四个字节。建议可能的话就用$STORAGE：2，这同时也提高了你的可执行程序的速度。注意，这个源命令可能会对从大型机处理环境转来的程序造成问题，在编译时要用$DEBUG。有关元命令的进一步信息，见Microsoft FORTRAN参考手册。

• 使用简化的错误处理。

如果你不需要运行时的出错信息，你可以与*/lib/nule6.o*文件链接，这样如果发生运行时错误，你的程序就被终止，不返回错误标号、信息与位置。用*/lib/nule6.o*并不影响编译时的错误，有关出错处理的进一步信息，见附录F“出错信息”

• 把程序分解成可由XENIX exec系统调用引用的覆盖。

如果你的程序包含有不需共享内存数据的部分，则你可把程序分解成子程序，并写一个“驱动”程序来调用每个子程序，子程序可以连续运行或所有子程序同时运行。对有关exec和使用shell运行并发和顺序进程的，参阅XENIX文本。

3.2.2 编译时内存受限制下的工作

通常用完编译时内存的原因就是一个源文件中的名字过多，编译程序第一遍可处理大约1,000个名字，Microsoft FORTRAN为以下内容建立名字项：

• 源文件，

- 源文件中说明的子程序和函数，
- 源文件中引用的子程序和函数，
- COMMON块，
- COMMON变量，
- 形式参数，
- 局部变量。

最后三项中用到的名字，仅在包含它们的子程序或函数被编译时才需要，子程序或函数编译后，这些名字就被舍弃，它们所占的空间也可被其他名字使用。

如果你把你的程序分解成子程序和函数，则局部名的空间可被共享，你的程序也可更大些。把一个源文件分解成多个源文件通常也减少了每个源文件中名字的数量。

注意，这些策略并不总是行得通的，在某些情形下，程序各部分通信所需的名字，如子程序间的通信或由COMMON建立的数据项之间的通信，要占用比划分程序所得到的更多的空间。

其它可能用尽编译时内存的因素有：

- 非常复杂的语句或表达式（如，嵌套很深的语句或表达式）。
- 大量的出错信息。
- 大的说明语句块（特别是EQUIVALENCE语句）。
- 过长的表达式或参数表（特别是在与I/O语句连用时）。
- 大的“基本块”。

当编译程序在对代码进行优化时，它把代码分成基本块，一个基本块是一段有一个入口和一个出口的代码。例如，任何以标号开头的语句就可能是一入口点，在某一地方可能有一个GOTO语句指向它，这样，每一个标号行开始自己的基本块，同时也结束了前一个基本块。

如果在你的程序里，大的基本块已成为问题，你可能从编译程序得到如下的出错信息：

```
Staetment too complicated (188)
```

语句太复杂

```
Expression too complex (902)
```

表达式太复杂

```
Compiler out of memory (未标号)
```

编译时内存不够

同样也有可能从XENIX得到“Segmentation violation”出错信息。

为了使基本块更小一些，你可以对更多的行进行标号，或对大的IF块进行分解。当基本块较小的时候，优化的机会也就较少了。

3.3 浮点运算

对于许多应用程序，你可以选择你程序的浮点运算怎样实现。浮点支持是以下三个元素的结合：

元　素	选　择
数学程序库	从你的XENIX系统，替换数学程序库或十进制数学程序库得到的数学上

的支持。

指令类型	库中执行浮点运算的子程序调用，或是程序中直接插入指令，以使软件在运行时产生中断。
数字格式	IEEE标准浮点格式或二进制码十进制格式。

每种选择在3.3.1至3.3.4节中都有详细描述。结合这些选择，产生四种浮点支持选择，如表3.1所列，选择1是缺省的：

表 3.1 浮点支持

选择#	库	指令	数字格式
1	XENIX	子程序	IEEE
2	XENIX	直接插入	IEEE
3	替换	子程序	IEEE
4	十进制	子程序	十进制

你可以通过在cl命令行中设定来选择这些任选项。表3.2表明如何挑选表3.1中列出的四种选择，如，要选选项3，你可以在cl命令行中用－FPa选项。

表 3.2 设定浮点选项

选择#	cl选项
1	—FPc
2	—FPi
3	—FPa
4	—FPd

表3.3和3.4根据你的应用目标，总结了一些最好的选择。例如，你没有80287协处理器，但你又想使程序更紧凑些，你就应该选择XENIX数学支持，直接插入指令和IEEE格式数字。这就是表中的选择2，查表3.2，你就知道你必须在cl命令行中设定-FPi选项。3.3.1至3.3.5节讨论每--种选择。

表 3.3 带80287协处理器的浮点选择

目标	选择#
速度	2
紧凑	2
可移植	3
精确的小数点	4

表 3.4 不带80287协处理器的浮点选择

目标	选择#
速度	3
紧凑	2
可移植	3
精确的小数点	4

对大多情形，缺省的（选择1）并不都是最好的选择，但它是最灵活的选择，用选择1产生的目标文件可以与替换数学程序库链接，也可与XENIX数学程序库链接，这就使得你可以选择一个合适的程序库而不用重新编译你的程序。

3.3.1 80287协处理器和仿真器

如果你用XENIX数学程序库(缺省的)并且有80287协处理器，则协处理器也起作用，如果没有80287，则XENIX仿真协处理器。协处理器和仿真器不与替换数学程序库和十进制数学程序库一起使用。

注意：

如果你使用80287协处理器或仿真器，则所有的例外都将被屏蔽。有关进一步的信息，见附录E“80287浮点错误的处理”

用80287协处理器与用80287仿真器做的基本算术运算结果上没有差别，然而对一些超越函数结果可能会有点不一致，在某些情况下用仿真器和协处理器运算，一个对最后一位舍去，一个则进位，这样在最后一位上可能不一致。

3.3.2 选择数学程序库

XENIX数学程序库是缺省的，这是唯一一个可以用80287协处理器或协处理器仿真器的程序库，也是唯一一个允许你用直接插入指令的程序库，见3.3.3节，“选择子程序调用或直接插入指令”。XENIX数学程序库产生的代码总是很紧凑的，如果你有80287协处理器，它产生代码的运算速度也是最快的，XENIX数学程序库用IEEE标准格式数字。

替换数学程序库是用于提高可移植性和速度的。如果你没有80287协处理器，这是你最快的选择。因为替换数学程序库不依赖于一特别的XENIX版本，这样对于用该程序库的程序在任何XENIX系统上都能给出同样的结果，替换数学程序库用IEEE标准格式数字，但为了速度和简洁牺牲了一些精确性，要用替换数学程序库，在cl命令行中用—FPa选项编译。

在二进制编码的十进制格式中，十进制浮点数最多到14位数字，并且在一有限的指数范围内，能精确地表示。假如对十进制数字操作结果在允许范围内，它能精确地表达。这个选项在商业和财务应用中非常有用，因为这些地方精确的结果非常重要。注意，十进制格式并不比IEEE二进制格式提供更高的精确度，它仅仅防止以二进制表示小数部分时出现的错误，这个格式与Microsoft的BASIC十进制格式兼容。要用十进制格式，在用