

FORTH

语 言

IBMPC 丛书

16

PC 丛书编辑部

## PC/FORTH+用户手册

## 目 录



PC/FORTH+ 概述	[ 1 ]
PC/FORTH+ 用户指南	[ 14 ]
PORTH 解释程序的一般描述	[ 14 ]
软盘的分配	[ 14 ]
PORTH 的启动	[ 15 ]
用PORTH 编写程序	[ 17 ]
PORTH 虚存	[ 18 ]
PC/M—86 操作系统的访问	[ 20 ]
保留 PORTH	[ 26 ]
用户的主要解释程序单元	[ 26 ]
在PC/PORTH+ 中CPU寄存器的使用	[ 27 ]
控制台输入和输出的注释	[ 27 ]
屏幕的自动装入	[ 28 ]
建立PORTH 应用程序	[ 28 ]
多任务	[ 29 ]
PC/FORTH+ 核 心 参 考	[ 31 ]
FORTH.SCR屏幕文件内容分配	[ 37 ]
PC/FORTH+ 全 屏 幕 编 辑 程 序	[ 40 ]
磁盘文件和记录接口	[ 43 ]
8086汇编程序	[ 46 ]
PC/FORTH+ 模 式 说 明	[ 58 ]
FORTH的 评 述	[ 68 ]
FORTH控制 结 构	[ 81 ]
PC/FORTH+ 词 汇 表	[ 88 ]
PC/FORTH+ 错 误 信 息	[ 124 ]
PC/GEN技术手册	[ 128 ]
PC/FORTH交互式符号纠错程序	[ 130 ]
屏幕文件列表	[ 134 ]
用于 C: debug.scr 屏幕文件的交叉索引	[ 160 ]



# PC/FORTH+ 概述

(本资料是MICROSYSTEMS公司实验室专利)

该手册安排这一节是为以前的PC/FORTH用户提供关于PC/FORTH+新功能的简单评述。对FORTH有了解的程序人员可略过该节而直接着手第二章用户指南。PC/FORTH+是与实验室微系统PC/FORTH向上兼容的版本,它的基础是32位的虚拟机。因此,从FORTH程序员的角度看,全部地址都是32位字长,并且数据和返回堆栈也是32位。为了节省词典空间,我们开发了一种新型编译技术,它还允许穿插使用16位和24位地址的线索代码。

## 要求机器的条件

PC/FORTH+要求一台运行PC-DOS 1.1或更高版本的IBM个人计算机和一个磁盘驱动器。PC/FORTH+也在PC-DOS 2.0和PC/XT上验证过。虽然,PC/FORTH+要求在64K字节的RAM上执行,但我们建议最低要采用128K字节。

## 关于速度的一般评述

通常,PC/FORTH+比基本的FORTH包运行慢,因为所有原始运算是32位数而不是16位。在“内部解释程序”中,因读出线索码的固定段时有额外的开销,而且在返回堆栈上进行的也是32位地址的“嵌套”和“非嵌套”操作。在执行中,速度下降的程度与应用程序类型和程序设计结构有关,大致的变化范围是在25~50%。

## 兼容性

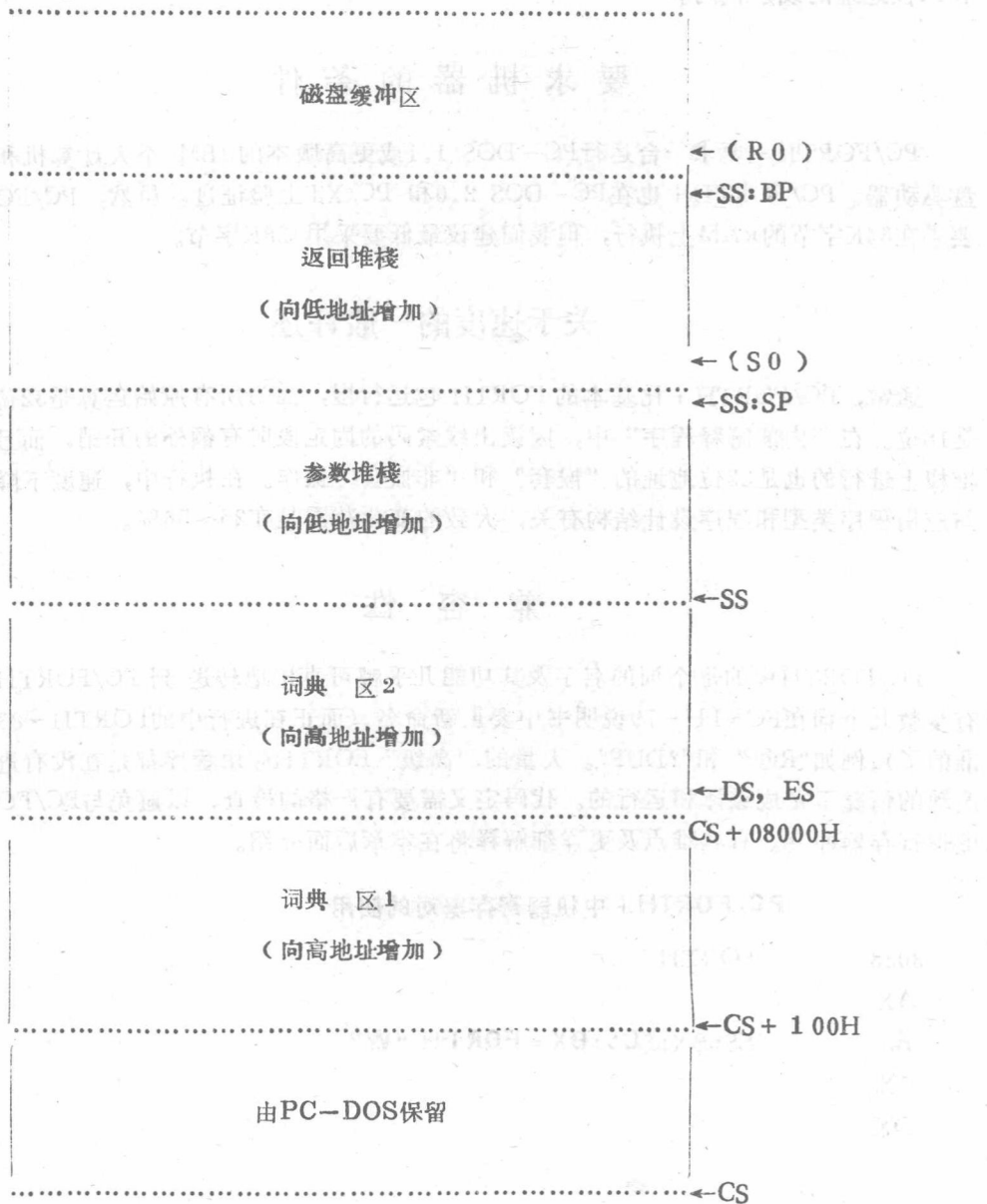
PC/FORTH中的每个词的名字及其功能几乎都可直接地转送到PC/FORTH+。只有少数几个词在FORTH-79说明书中要重新命名,(而正在进行中的FORTH-83就是标准的了),例如“R@”和“?DUP”。大量的“高级”FORTH应用程序都是在没有进行任何改动的情况下完成编译和运行的。代码定义需要有严格的检查,以避免与PC/FORTH+虚拟寄存器冲突。有些难点及更详细解释将在本章后面介绍。

### PC/FORTH+中机器寄存器对的使用

8088	PORTH
AX	
BX	ES:BX或CS:BX = FORTH "W"
CX	
DX	

- SI        DS:SI = PORTH "IP"
- DI
- BP        SS:BP = PORTH "RP"
- SP        SS:SP = FORTH "SP"
- CS        始终指向 FORTH 的基地址
- DS        用SI去构成FORTH "IP"
- ES        有时用BX构成FORTH "W"
- SS        始终指向RAM的顶部—64K字节

**PC/FORTH + 内存映像图**



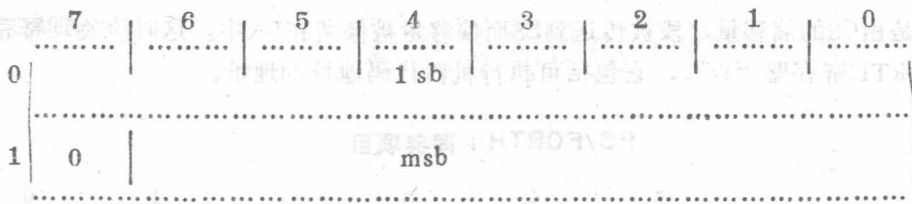


词典代码区 1 包括全部子编译的PC/FORTH+ 系统, 并从逻辑地址 00100H延伸到 07FFFH。对于剩下来的内存, 作为用户程序的编译都是可用的。当最高有效位为零时, 包含在这个区域中的所有涉及到的定义都被编译成固有的16位代码字段地址。以常规FORTH的方法建立在这个区域上的代码原语是包含参数字段的代码段。区 1 也包括内部解释程序、自举文字表 (boot—up literal)、磁盘和视频I/O驱动程序以及正文I/O缓冲区。

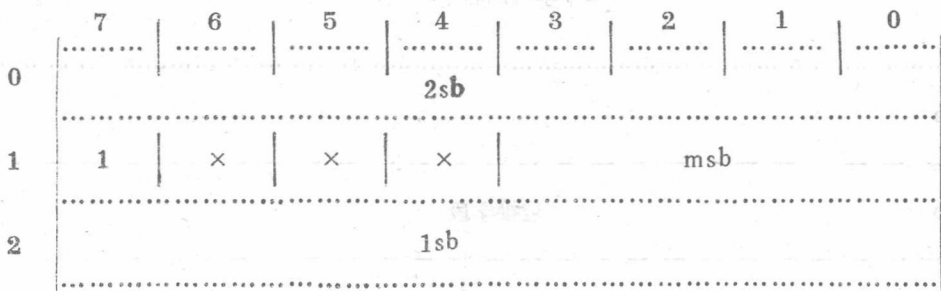
词典区 2 (逻辑地址空间为08000~FFFFFH) 对用户程序的编译也可用。该区上的线索代码是作为16位与24位地址混合编译的 (见下面线索代码格式说明)。虽然该区的定义通常是用一个单独的内部解释程序处理的。这对编程序的人员来说是不透明的并且区 2 简单被视为区域 1 的延伸。

词典区 2 上的代码原语是按一种指定的方式汇编的。代码字段指向区域 1 中的一个短处理程序, 在该区中存贮FORTH虚拟机状态并执行参数字段开始的“远调用”(这里PFA的地址是由“W” + 2 得到)。代码序列是用一个“远返回”结束, 并能附带恢复常规的CS内容, 同时传送控制到另一区 1 中的其它代码段, 重新装入FORTH其余寄存器内容。然后, 恢复正常线索代码的解释。确切地说, 涉及到的这个调用过程是不可少的, 因为, 就整体而言, FORTH解释程序/编译程序可把CS寄存器看作是不变的。

线索代码地址解释

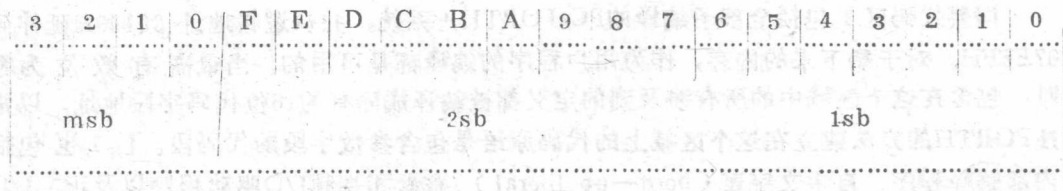


上图是一个被编译的线索代码地址, 它指向被设置在逻辑地址空间00000~07FFFH上的词典项的头。31位总为零, 这表明地址是从CS的偏移。所有的系统原语都是属于这种范畴。在此情况下, 物理寄存器对CS:BX形成FORTH寄存器“W”, 而且包含的是可执行的机器码地址的地址。

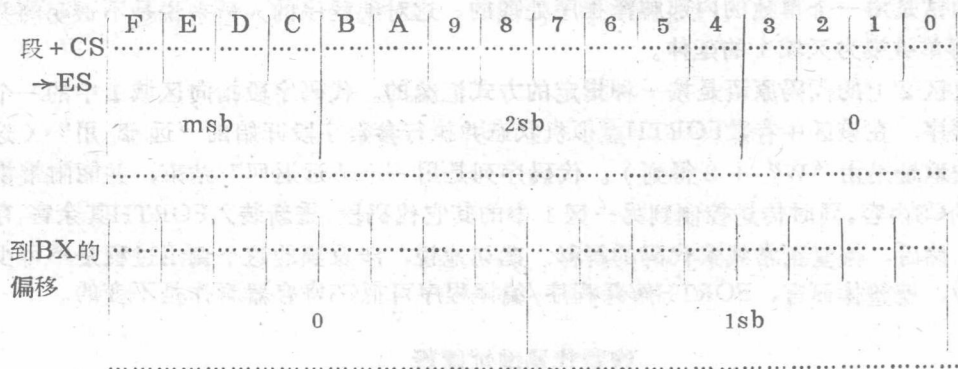


上图是被编译的线索代码地址, 它指向被设置在逻辑地址空间 08000~FFFFFH 的词典项的头。15位始终置 1, 它表明必须建立一个 20 位的地址, 并且是通过内部解释程序从CS当前内容的偏移。

汇编成三个字节的“长”线索代码地址如下: (包含特征位的# 1 字节的高 4 位被去掉了)

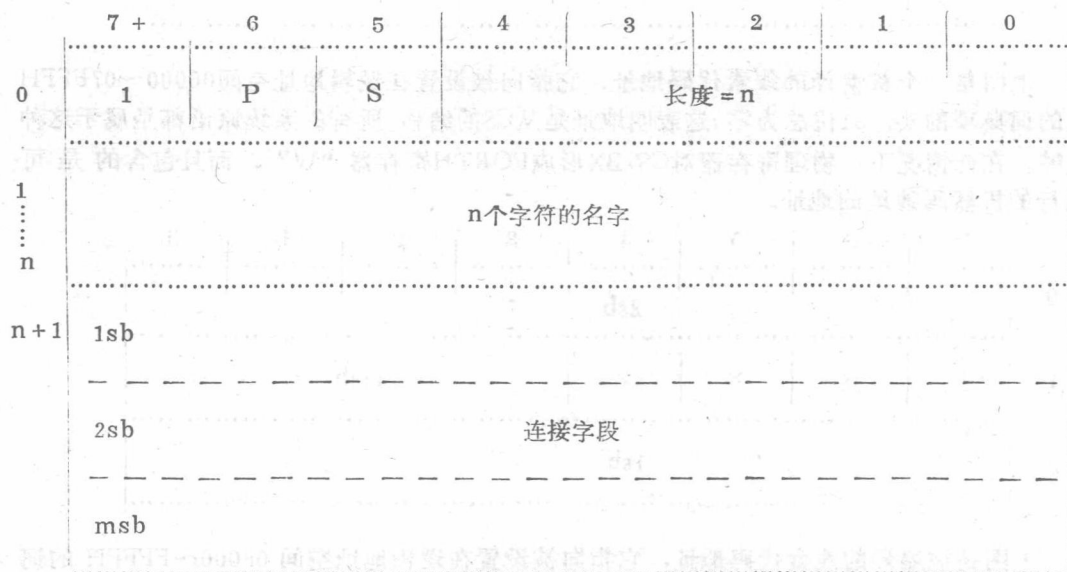


20位的逻辑地址通过一系列的移位被分解成一个程序段(segment)或者节段(paragraph)地址,再将其加到CS寄存器当前的内容上,而偏移始终在0000~00FFH范围内。这样确保偏移不会溢出或者在任何合理的程序设计状态下“兜圈”(“Wrap around”)。



后面是由CS的偏移量,段被传送到ES而偏移量被保留在BX中。这时该物理寄存器对构成FORTH寄存器“W”,它包括可执行机器代码地址的地址。

### PC/FORTH+ 词典项目



(接下页)



S = 抹去(Smudge)位, 在检索词典时阻止匹配  
 P = 优先或“立即”位

### PC/FORTH + 特有的词

(在基本PC/FORTH中没有)

由于把16位和24位混合线索地址编译成词典、三字节的连接字段及其它指定格式的数据, 所以PC/FORTH+又包含若干用于取数, 编译或存贮地址项的新词。因为这种词是苛刻的, 其词典的物理结构对基本PC/FORTH包向下是不兼容的, 所以在应用软件中应避免使用它们。注意, 堆栈上的数据始终为32位字长, 不管内存字段的源程序或目的程序的字长是多少。

< CMOVE                      地址1   地址2   长度   - - -

“反向词<CMOVE”, 其功能像CMOVE一样, 只是字符串的传送是从右到左, 而不是从左到右。传送的最大长度为OFFFHH(64K)字节。

?VPAGE                      - - -   n

返回活动的视频页数。只有显示方式2和3是合法的。

ADDR>S&O                      逻辑地址 - - -段 + 偏移

接收一个范围在00000~FFFFFH之间的相对于FORTH基址的32位逻辑地址, 并返回一个32位的字, 其高16位为物理段, 而低16位为偏移量。

CFA!                              数据   地址 - - -编译字节

用适当的格式为PC/FORTH+线索代码在给定的地址上存贮数据。如15—31位全为零时, 存贮两个字节, 并且在堆栈上返回两个数。如果15—19位的任一位非零时, 那么就存贮三个

字节，并且在堆栈上返回三个数。在后一种情况下，把数据重新组合成第15位为1的特定的格式（关于线索代码的说明请见这本资料的有关章节）。

**CFA,** 数据 ---

按HERE将线索代码地址编译到词典上，再根据数据编译成两个或三个字节。

**CFA@** 地址 --- 内容 字节数

从地址中取出并解释可为两个或三个字节的线索代码地址，重新组织并扩展符号，使它成为最基本FORTH的32位相对逻辑地址的一般格式。同时也在堆栈顶上返回经过译码的字节数（2或3）。

**CLIT** --- 数据

把一个8位（二进制）在行的字母装入到堆栈上，并用零将它扩展成32位。注意，这是一个不带符号的扩展！

**LINK!** 数据 地址 ---

按照词典上3字节连接字段的要求格式，在给定的地址上存入给定的数据，就是说，依次将3个字节的最有效的低字节存入低位的内存地址上。提供数据的高8位（二进制）被舍掉。

**LINK,** 数据 ---

把一个三字节的连接字段编译到词典上。在堆栈上提供数据的高8位（二进制）被舍掉；在丢失有效位的情况下也不给出错误信息。

**LINK@** 地址 --- 数据

从地址中取出三个字节的连接字段的内容，并用零把它扩展成32位字长。

**S&O>ADDR** 段 + 偏移 --- 逻辑地址

接受一个32位的物理地址，其中高16位为段，低16位为偏移，并返回一个在00000—FFFFFFH范围内的相对于PC/FORTH基址的32位逻辑地址。这种方法通常用来把一个地址转换成FORTH程序可用的格式。这个地址是从返回堆栈中由R@取出的。

**VPAGE** n ---

建立活动的视频页。只有显示方式2和3是合法的。



**W!**

**数据 地址 - - -**

按通常的内部格式，在给定的地址上存入给定数据的低16位（最有效位被存到低位内存地址上）。堆栈上数据的高16位（二进制）被舍掉。

**W,**

**数据 - - -**

把16位数据编译到词典上。舍掉在堆栈上提供数据字的高8位。在丢失有效位的情况下，也不给出报告信息。

**W@**

**地址 数据 - - -**

从地址取出16位数据，并带符号扩展到32位。

**WLIT**

**- - - 数据**

将一个十六位（二进制）在行的字母装入到堆栈上，并带符号扩展到32位。

## PC/FORTH + 各种功能的词

FORTH的标准传送操作符〔如@ ! , 〕全都是存取32位的数值。

FORTH标准的数或逻辑操作符〔+ - \* / AND OR XOR等〕都采取32位自变量，并且返回的结果也是32位。

**名字**

**注 释**

**+ LOOP**

增量值被限定为16位带符号整数。

**BLANKS**

初始区域长度被限定为64K字节。

**CMOVE**

串传送的长度被限定为64K字节。

**COMPILE**

存入两个或3个字节，取决于地址。

**CREATE**

三字节连接字段。

**DLITERAL**

编译DLIT和8个字节为一行的值。

**FRASE**

初始区长度限定为64K字节。

**FILL**

初始区长度限定为64K字节。

**LITERAL**

根据字母值的大小用8位、16位或32位值分别在行上编译CLIT、WLIT或LIT。

**LFA**

pfa→lfa = 5 - （相同的逻辑结果）。

**NFA**

pfa→nfa = 6 - - 1 TRAVERSE（相同的逻辑结果）

**PFA**

nfa→pfa = 1 TRAVERSE 6 + （相同的逻辑结果）

**S =**

比较字符串的长度被限定为64K字节。

**SAVE**

在写.COM文件前，自动地建立自举文字表。

**set-dta**

接收一个32位逻辑地址（在很少的PC/FORTH版本中使用）。

## PC/FORTH + 不能使用的词

在基本的PC/FORTH包中有下述的词,PC/FORTH+ 不能使用: + BUFF, B/SCR, BUFFER, FIRST, LIMIT, R/W, REC, SEC-READ, SEC-WRITE, SEC/BLK, SET-IO, USER和USE。

## 使用新名字的词

? DUP	为 -DUP
PALETTE	为 PALLETTE
R@	为 R

## FORTH虚拟机的注释

### 冷启动

提供给PC/FORTH+的像“.COM”或“.CMD”文件,在装配时要求你不必再定位或者连接。装入PC/FORTH时,内存单元也不是在它的控制之下,而是取决于操作系统的版本和当前外部设备驱动器的类型等因素。虽然PC/FORTH允许连续重复地使用,但不另外从磁盘重新装入它的情况下,由操作系统重新输入FORTH在目前是没有保证的。

从PC-DOS或CP/M-86在CS:0100H上进入PC/FORTH+,并包括了一个跳转到“低层”虚拟机冷启动的倒行程序。PC-DOS下的第一个动作是置CAPS-LOCK为TRUE并置NUM-LOCK为FALSE,然后接受上面的“控制台中断”的中断处理程序(1NT 23H和1BH)。初始化段寄存器ES和DS使之与CS入口时的值一样,然后调用PC-DOS的内存尺寸的操作(1NT 12H)或者是测试CP/M-86的BIOS内存段表,从而找出RAM的顶地址。置堆栈段寄存器SS为小于64或72K字节的RAM顶的节段(Paragraph)地址。根据机器中内存的数量来决定该段是否能与代码段重合。注意:如果PC装备的RAM小于128K字节的话,那么PC-DOS的COMMAND.COM的暂存部分要受到破坏。

由常数#BUFFS来确定磁盘缓冲区,并计算在堆栈段中,返回堆栈与数据堆栈对数据基址的偏移量,同时将它们放入BP和SP中。修改变量R0与S0的值,使它们分别逻辑等价于SS:BP和SS:SP。最后,把FORTH机IP的物理寄存器组DS:SI初始化,以指向含有高级定义ENTRY的代码区段地址的一个字,并把控制传送给内部解释程序“NEXT”,并从此处执行“高级”FORTH代码。

词ENTRY检查CS:0080H上缺省的缓冲区以确定在命令行的尾部是否有指定的文件名,如果没有,就把变量DEFAULT-FILENAME中的文件说明串传送到CS:005CH上的缺省文件控制块中。然后,ENTRY准备打开指定的或缺省的文件。如果文件没有设置时,CP/M-86系统将执行一个直接的出口,而PC-DOS系统将打开“NUL”设备作为当前的屏幕文件——然后使用操作符DIR和USING来打开合法的屏幕文件。在PC-DOS控制下建立文件时,缺省 fcb (在偏移006AH上)的记录长度参数为1024字节。最

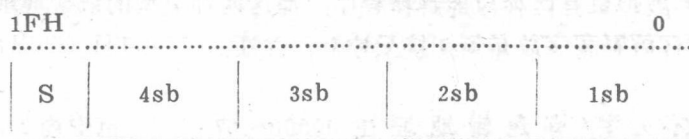
后，清除屏幕，并且当颜色/图形板是使用当前视频方式的情况下置为 2（80×25 黑白的正文）然后控制传送到 COLD。

在 COLD 中，置 STATE, IN, OUT, BLK, SCR 和 DPL 变量的值为零。初始化 TIB，使其指向代码段底部缺省的正文输入缓冲区。FENCE, DP, VOC—LINK 值和 FORTH 词汇表定义的连接字段是由自举文字表写入的。磁盘缓冲区清零，空出磁盘块表，指定控制为 ABORT。

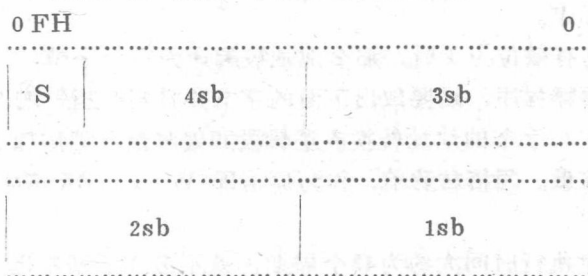
在 ABORT 情况下，强迫系统数据基址为 DECIMAL，初始化参数堆栈指针，并显示开工 (Sign-on) 信息，它包括有版本号和执行文件的建立日期。将 CURRENT 和 CONTEXT 词汇表置为 FORTH。然后引用词 QUIT，清除返回堆栈，并由键盘编制解释命令。

### 数据类型

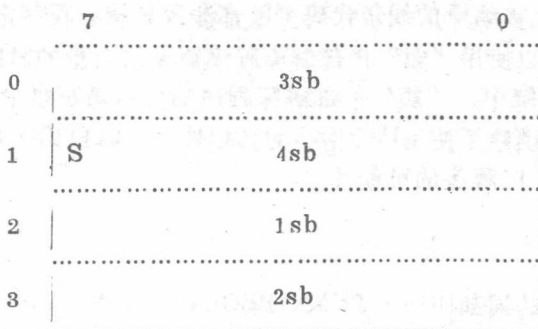
PC/FORTH 基本上有两种数据类型：一种为 32 位整数，而另一种是用零扩展到 32 位的 8 位字符。数据和参数堆栈均为 32 位字长。该方法的基本 FORTH 操作符 @ 和 ! 在指定的地址上进行取或存的也是 32 位字长的数值，而所有的算术逻辑操作符，如 ( + - \* / AND OR XOR ) 的自变量及其返回的结果均为 32 位字长的数值。同样，CONSTANT 和 VARIABLE 也分别返回 32 位的常数或 32 位的变量地址。如上所述，FORTH 的“双精度”运算，如 D+，D•，2DUP 和 2SWAP 是在堆栈的两个单元上进行的，实际上的操作是 64 位数值运算。



上图：在“逻辑”堆栈上理解的 32 位数



上图：在“物理”堆栈上理解的 32 位数



上图：存储到内存中的 32 位数

在PC/FORTH+中有些词的操作是16位的数值(比如W@和W!),但这些词被指定为由编译程序内部使用,而在应用程序中应避免使用。

### 内存寻址

虽然由FORTH程序员使用的逻辑地址是在00000—FFFFFH范围内,实际上它们是针对FORTH始地址的,而不是相对于机器的实际物理零地址的。因为逻辑零地址实际上是指向操作系统的顶部,真正可用的地址空间始终小于0FFFFFH字节。

在运行时,访问内存的操作符(例如@或!)给出的都是逻辑地址,通过移位分解将它转换成物理段和偏移量,并加到CS寄存器的内容上,如果被查看的物理地址已落到FORTH底部段内的情况下,那么去检查高位有效地址部分时,用一个CS段来替代,这样使用这些原语一般说是“灵活”的。同时也有效地改变了执行的速度。

通过运行的例行程序“:”,存储在返回堆栈上的地址是以物理段和偏移量的形式出现的,而不是一个逻辑地址。这样就大大地减少了嵌套线索代码的开销。如果FORTH程序员希望在运行时检查返回堆栈上的一个地址(通常用“R@”把它拷贝到参数堆栈上),就必须用词“S&0>ADDR”将它转换成32位逻辑地址。例如:

```
: (. ) R@ S&0>ADDR COUNT DUP 1+ R> + >R TYPE ;
```

### 内部解释程序

实际上FORTH虚拟机有两种内部解释程序,而这两种类型的线索地址都是可以编译的。“短”内部解释程序总是最初接受控制,并通过FORTH IP完成16位间接取的。

如果最高有效位为零(即逻辑地址在00000~07FFFH范围内),那么基本PC/FORTH包能精确地解释这个地址。在间接转移前作为另一个原语W=CS:BX的一个信号清除这个进位标志。

如果取出字的最高有效位为1时,那么地址被编译为三个字节。“短”内部解释程序转换成“长”内部解释程序,还要取出下面的字节并且指令指针增加。通过一系列的移位舍掉标志位并将三个字节的地址转换成逻辑段和偏移量,然后加上当前CS的内容后,成为物理段和偏移量。间接转移前,作为原语即W=ES:BX的信号,进位标志置1。

“短”内部解释程序执行时间大约为43个周期,差不多大于基本PC/FORTH的20%。如果引用“长”内部解释程序的话,通过IP整个间接转移时间大约为87个周期。因为在较高的FORTH逻辑地址上被编译的线索代码多数都涉及到核心程序定义、控制词以及由2个字节组成的项,所以使用“短”内部解释程序需要用大量的时间。

实际的PC/FORTH+系统中,“短”内部解释程序的代码是在每个代码原语的结尾按行进行编译的。这样就消除了使用JMP指令的必要性,同时也确保在需要时按照BIU指令队列予取大量NEXT程序的可能性。

### 控制结构

编译程序所有关于结构的词如IF……THEN, BEGIN……UNTIL,和DO……LOOP,都编译成一个在行上16位的偏移量。运行时是通过BRANCH或0BRANCH将它加到SI的

当前内容上。因为偏移量被限制为16位,所以BRANCH和0 BRANCH就不能直接地被编译,而且也不能使用线索码解释程序把控制传送到FORTH地址空间的任何地方,在溢出的情况下,SI只能在当前段上兜圈(arap azound)运行时,SI的内容不可预测,因为这要依赖装入的地址和其它因素。

目前,+LOOP只处理带符号的16位增量值,当然,输入到DO LOOP的32位参数全部是有效的。

用原始的PC/FORTH包提供的CASE语句使用PC/FORTH+将不能正确地编译出代码,但在分配的磁盘上又提供了一个正确的版本。同时还预编译成分配的FORTH.COM文件。

### 算术运算

为了提高执行速度,乘和除的例行程序都要检查输入的32位自变量数值的大小。如果高16位是无用的,那么就把控制传送给短的数学例行程序,并做单精度的数学运算(如基本的PC/FORTH程序包),而得出的是简单符号扩展的结果。如果输入的自变量已超过16位,那么就进行全双精度的数学运算。注意,当强行通过含有多重除和循环的长码序列时,使用U/操作(64位被除数和32位除数)是比较慢的。

### 磁盘I/O

为了改进PC/FORTH+屏幕文件接口的性能,这里的程序完全是用机器码重新编写的。它是在FORTH内存段底部的表中记住块地址的,而不是在缓冲区本身头上的字中记住的(如上述情况)。关于fig—FORTH的有些常数和组成的块,全成为多余的了,并且已将它们删掉,其中包括:SEC/BLK, B/SCR, FIRST, LIMIT, USE, REC, SET—IO, SEC—READ, SEC—WRITE, R/W, BUFFER, 和+BUF。

适合描述和访问屏幕文件的其它的词有: B/BUF, # BUFF, PREV, DISK—ERROR, BOOF—SCREEN, SCREEN—FCB, DEFAULT—FILE, BLOCK, UPDATE, FLUSH和EMPTY—BUFFERS。

### 视频I/O

为了最快速地显示,PC/FORTH+没使用PC—DOS或IBM ROM BIOS的调用,而是直接输出内存映象的。所以,如果机器在标准内存的单元中没就有单色或彩色/图示板的话,视频输出就不能正确地工作。于是,在自举文字表中包含提供一个指定的单元,使用它可作为支持用户驱动器设备的向量输出。

### 自举文字表

由于PC/FORTH+在内存寻址和缓冲区分配有变化,所以在基本PC/FORTH包中的自举文字表地址不再是正确的了。

地 址	名 字	含 意
0 0 1 0 3	INIT—FORTH	FORTH词汇表最顶项的字
0 0 1 0 7	INIT—FENCE	FENCE的初始值



0 0 1 0 B	INIT-DP	初始的词典顶项
0 0 1 0 F	INIT-VOC-LINK	最后定义的词汇表
0 0 1 1 3	RAM-SIZE	机器RAM的尺寸, 以字节计算
0 0 1 1 7	CODE-SEGMENT	FORTH段基址
0 0 1 1 B	UABORT	用户ABORT处理程序
0 0 1 1 F	UQUIT	用户QUIT例行程序
0 0 1 2 3	UEMIT	用户屏幕输出驱动程序
0 0 1 2 7	UKEY	用户键盘输入驱动程序
0 0 1 2 B	ULIST	用户打印机输出驱动程序
0 0 1 2 F	U?TERMINAL	用户键盘登记驱动程序
0 0 1 3 3	UTYPE	用户字符串输出驱动程序

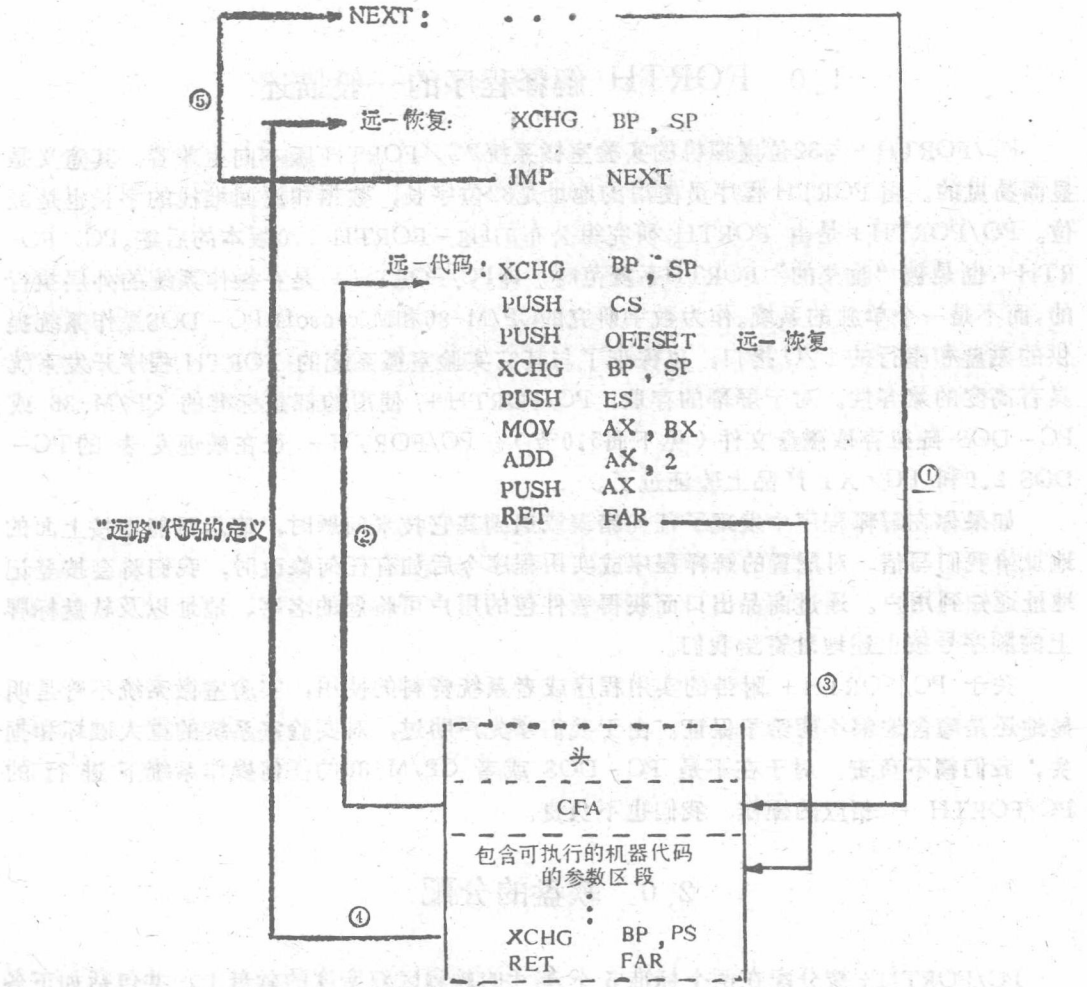
### 带注释的内部解释程序

在入口处, 假设DS:SI为FORTH IP

1 2	NEXT:	LODS	WORD	; 得到线索地址
3		OR	AX, AX	; 如果最高位已置1, 那么
4, 16		JS	LONG-NEXT	; 它是24位地址, 附带
				; 保留OR, 并清除进位标志
2		MOV	BX, AX,	; 通过CS: BX = W间接转移;
2		SEG	CS	;
20		JMP	[BX]	;
	LONG-NEXT:			; 24位地址跳转至此
2		MOV	CX, CS	; 从线索地址得
2		SHL	AX, 1	; 出FORTH段基址
2		SHL	AX, 1	; 段 偏移 = msb, 2sb * 16
2		SHL	AX, 1	;
2		SHL	AX, 1	;
3		ADD	AX, CX	; 由FORTH基址对真正
2		MOV	ES, AX	; 的物理段, 再定位
12		LODS	BYTE	; 得到线索地址的低字
2		MOV	BL, AL	; 节, 并放入BX, 然后
3		SUB	BH, BH	; 清除高8位, 用长信号
2		STC		
2		SEG	ES	; NEXT替代DS, 通过ES:
20		JMP	[BX]	; BX = W间接跳转。

经过短NEXT的总周期为43

经过长NEXT的总周期为87



通过PC/FORTH+ 汇编程序连接到超出FORTH底部的内存段的代码定义:

- 1、当涉及代码定义时，FORTH寄存器W装有代码区段的地址，并在底层段上依次地装有称之为“远-代码”的分段机器语言地址。
- 2、“远-代码”例行程序将FORTH的代码段地址和“远-恢复”例行程序的偏移量被压到返回堆栈上。这样，代码定义就能访问FORTH参数堆栈上的数据了。然后，“远-代码”将涉及到代码定义的段和偏移量再压到数据堆栈上，并执行远返回。该返回具有远（中间段）跳转到要求的参数区段的功能。
- 3、执行代码定义内容，并由汇编程序放置那里的远返回所终止。
- 4、由反向远返回返回到分段代码远恢复，并附带恢复代码段寄存器的内容，使之指向FORTH的基址。
- 5、“远-恢复”保证FORTH虚拟机的环境不受破坏，并反向转移到NEXT，以恢复通常线索代码的执行。

# PC/FORTH+ 用户指南

## 1.0 FORTH 解释程序的一般描述

PC/FORTH+ 与32位虚拟机的实验室微系统PC/FORTH版本向上兼容,其意义是显而易见的。由FORTH程序员使用的地址是32位字长,数据和返回堆栈的字长也是32位。PC/FORTH+是由FORTH研究组公布的fig-FORTH 1.0版本的后继。PC/FORTH+也是被“收集的”FORTH系统范畴,即PC/FORTH+是在操作系统的外层执行的,而不是一个单独的系统。作为数字研究的CP/M-86和Microsoft PC-DOS操作系统提供的磁盘和串行的I/O接口,也保证了与其它实验室微系统的FORTH程序开发系统具有高度的兼容性。对于屏幕的存取,PC/FORTH+使用的都是标准的CP/M-86或PC-DOS随机存取磁盘文件(见下面5.0节)。PC/FORTH+已在最近发表的PC-DOS 2.0和PC/XT产品上验证过了。

如果你在解释程序中发现了任何错误或遇到其它技术问题时,请尽可能地按上面的地址给我们写信。对配置的解释程序或实用程序今后如有任何修改时,我们将会按登记地址通知到用户。通过商品出口而获得软件包的用户可将您的名字、地址以及软盘标牌上的顺序号按上述地址寄给我们。

关于PC/FORTH+附带的实用程序或者系统资料的使用,实验室微系统不管是明显地还是隐含地都不能给予保证。由于我们事先声明过,对实验室系统的重大破坏和损失,我们概不负责。对于在不是PC-DOS或者CP/M-86的任何操作系统下进行的PC/FORTH+相应的操作,我们也不负责。

## 2.0 软盘的分配

PC/FORTH+被分配在两个标准5 $\frac{1}{4}$ 英寸的软扇区双密度的软盘上,共包括如下各文件:

forth.C??	可执行的FORTH解释程序
pcfplus.C??	带予编译的编辑和汇编程序的可执行FORTH解释程序
scopy.C??	屏幕传输实用程序
memsize.com	PC/DOS内存尺寸测定程序
load.com	PC/DOS的Big COM文件装入程序
forth.scr	存取屏幕的FORTH实用程序
asm86.scr	8086汇编程序源程序正文
editor.scr	屏幕编辑程序源程序正文
latest.doc	用户注释★

此外,你买了浮点选件的话,你的磁盘还能包括下述全部或一部分文件:

hfpbase.C??	硬件辅助核心程序
-------------	----------

hfpforth. C??	硬件辅助解释程序
hfpplus. C??	带预编译的编辑程序和汇编程序硬件辅助解释程序
sfpbase. C??	浮点软件核心程序
sfpforth. C??	浮点软件解释程序
sfpplus. C??	带预编译的编辑程序和汇编程序的浮点软件解释程序
float.scr	浮点软件源程序
i 8087.scr	Intel 8087 支持的源程序代码

扩展的“C??”在CP/M-86下为“.CMD”，在PC-DOS下为“.COM”。注意，根据你使用的是IBM CP/M-86或是Digital Researcn CP/M-86，还是Concurrent CP/M-86所要求FORTH.COM的版本也略有不同。如果不能接收合理的命令文件时，那么就将你得到的磁盘退给我们，并带来一份有关操作系统的注释说明。

★该文件包括一些目前补充或修改的内容，有关这些文件的资料都是最近印刷的。

### 3.0 FORTH 的启动

将FORTH.COM, PCFPLUS.COM和FORTH.SCR文件传送到驱动器A的一个磁盘上（大约需要120K字节作为文件的自由空间）。然后，用下面的办法把FORTH引出来：

```
A>FORTH
```

此时将解释程序装入内存，然后印出识别信息（根据所提供的系统类型和内存数量的不同，印出的信息与下例也有所不同）：

```
PC/FORTH+ Version 1
(C) 1982 Laboratory Microsystems
Using file : a : forth.scr
Dictionary space available : 18532 bytes
```

```
OK
```

此时表明已作好使用FORTH的准备。为了检查屏幕文件的内容，可使用INDEX功能来显示每个屏幕的第一行，其用法如下：

```
1 50 INDEX <return>
```

INDEX功能的一种简单类型打入下面的命令可以看到四种格式：

```
QX <return>
```

检查一个给定屏幕的正文，可以先打入屏幕号，后面是词“LIST”。例如，要读