面向FoxBASE的汇编语言编程与应用

毕广吉 著

北京大学出版社

面向 FoxBASE 的汇编语言编程与应用

毕广吉 著

北京大学出版社
北 京

内 容 提 要

本书是 FoxBASE 编程者的高级读物，专门介绍面向 FoxBASE 的汇编语言编程方法，并给出大约 200 个实用的汇编语言模块。这些汇编语言程序极大地扩充了 FoxBASE 的功能，使 FoxBASE 应用程序升级到新的水平。书中的方法及程序同时也适用于 FoxPro。

本书供学习和使用 FoxBASE，FoxPro 的编程者阅读，也可供学习汇编语言的人员学习，还可作为大、中专院校师生的参考书。

与本书同时出版的还有配套的软件，包含了书中全部的. ASM，. PRG 源程序及. BIN 模块，需要的读者请与北京大学出版社软件部联系购买。

图书在版编目(CIP)数据

JS201/15

面向 FoxBASE 的汇编语言编程与应用/毕广吉著. —北京：北京大学出版社，1996. 9
ISBN 7-301-03170-X

Ⅰ. 面… Ⅱ. 毕… Ⅲ. 汇编语言 Ⅳ. TP312

书　　名：面向 FoxBASE 的汇编语言编程与应用
著作责任者：毕广吉
责 任 编 辑：沈承凤
标 准 书 号：ISBN 7-301-03170-X/TP・0305
出　版　者：北京大学出版社
地　　址：北京市海淀区中关村北京大学校内　100871
电　　话：出版部 62752015　发行部 62559712　编辑部 62752032
排　印　者：北京大学印刷厂印刷
发　行　者：北京大学出版社
经　销　者：新华书店
787×1092 毫米 16 开本　26.75 印张　666 千字
1996 年 10 月第一版 1996 年 10 月第一次印刷
定　　价：35.00 元

前　言

现在FoxBASE数据库系统非常流行，相当多的人学习过并且掌握了FoxBASE编程，但是其中为数不少的编程者却难以编写出可达商品化水平的应用程序。其主要原因是，FoxBASE的命令/函数集并不能满足我们全部的需要，必须利用汇编语言对其功能进行扩充。本书就是针对这样的读者编写的，专门介绍面向FoxBASE的汇编语言编程，是FoxBASE的高级读物。利用本书介绍的方法或直接调用本书的程序，可以做出图文并茂、声色俱佳的用户界面和应用程序，使软件增色不少。

尽管目前图书市场上关于FoxBASE的书很多，但其中关于汇编语言编程的叙述却极少，往往只是罗列一下调用规则而已，最多也不过举上一个十几行的例子，读者无法从中学习编程方法，更得不到实用的程序。另一方面，很多FoxBASE编程者对汇编语言了解不多，再加上汇编语言编程还要涉及BIOS，DOS，汉字系统、VGA显示卡、可编程控制器等软、硬件知识，因而本书专门介绍供FoxBASE调用的汇编语言编程是十分必要的。

本书首先分析了FoxBASE调用汇编语言子程序的规则、调用的过程、FoxBASE内存变量存储的格式、参数传递方法等必备的基础知识，然后分别具体介绍显示输出、键盘控制、时钟操作、磁盘/文件操作、存取屏幕/窗口、图形编程、前台/后台音乐、位图显示、统计图形、亮度控制、艺术清屏等编程原理和方法，并给出约200个实用程序。

编写本书时还特别考虑到了另外一些读者，这部分读者只想获得实用程序，不打算深入学习编程原理和技术细节，那么只要将本书中提供的程序应用于你的系统中，立即会使你的程序得到极大的改善，升级到专业化、商品化的水平。

本书介绍的方法和程序也适用于FoxPro。通过本书的学习，读者不但可以学习汇编语言编程方法，得到实用程序，而且还能学习到很多关于VGA卡、DOS，BIOS可编程控制器的知识。除此之外，本书还能帮助读者更深一步地了解FoxBASE本身。

全书所有程序均在FoxBASE　2.10，MASM　5.10，DOS　3.30，256K的VGA卡环境下调试通过，所有程序均与具体的汉字系统无关。

由于作者水平所限，谬误之处在所难免，敬请读者赐教指正。

毕　广　吉

1996年3月

目　录

第一章　面向 FoxBASE 的汇编语言编程基础

本章主要介绍面向 FoxBASE 的汇编语言编程必备的基础知识。由于篇幅所限，本书不介绍8088/8086汇编语言指令和伪指令的功能与用法，也不介绍汇编语言编程的一般方法，读者可以自行参考有关的书籍，例如书末所列的参考文献[1]等。

1.1　在 FoxBASE 中调用汇编语言子程序的方法

1.1.1　调用方法

在 FoxBASE 中使用汇编语言子程序有三个步骤：

第一步，用 LOAD 命令将汇编语言子程序模块从磁盘装入内存，准备供 CALL 命令调用。即

LOAD<二进制文件名>

装入的程序必须是二进制文件形式，默认的扩展名是.BIN，每个文件长度不能超过32K字节。FoxBASE 最多可装入16个二进制文件。文件装入后，文件名便成为模块名，扩展名不再使用。因此，如果新装入的文件名与原先装入的文件名相同，即使它们的扩展名不同，新装入的文件也将覆盖掉内存中原先装入的那个文件。如果在装入文件时不指出扩展名，LOAD 命令认为扩展名是默认的扩展名.BIN。

本书中将把二进制汇编语言模块简称为.BIN 模块，而不论实际的扩展名是不是.BIN，事实上也没有太充分的理由不使用.BIN 作为扩展名。

有一点应该特别指出，如果一个二进制文件被重复装入，FoxBASE 并不会去检查它是否已经装入过，而是再一次装入同一文件，其结果是将上次装入的同一个内容的文件覆盖掉。

通常看来，这样做法不仅浪费装载的时间而且对于一个较复杂的.BIN 模块来说，可能会造成灾难。例如，用 LOAD 命令将音乐程序 SOUND.BIN 装入内存(见第八章)，然后用 CALL 命令调用后台演奏音乐功能，一首悠扬的乐曲就开始了。如果此时再重复装入 SOUND.BIN 模块，就不再演奏乐曲并且有可能死机。

其原因很简单，在第一次装入程序并调用后台音乐功能时已经改变了1CH 号时钟中断向量，同时将表示乐曲的音符串填写在内存中，使音乐正常演奏。这时如果突然调入同一程序进行覆盖，纵然二者的程序代码完全相同，但却失去了已经填写好的重要数据，音乐演奏无法进行，更无法恢复原来的1CH 号中断向量，这就会导致死机。

因为 FoxBASE 并不负责检查.BIN 模块的重复装入，所以这一责任就只好由程序员自己承担了。

第二步，用 CALL 命令调用汇编语言程序。当 FoxBASE 用 LOAD 命令装入一个.BIN 程序时，并不立即运行它。要想调用.BIN 程序，就要使用 CALL 命令。CALL 命令的用法是：

CALL<二进制模块名>[WITH<字符串表达式>/<内存变量>]

其中 WITH 子句指明调用参数，当无输入输出参数时，该子句可以省略。传递参数可以用字符

串表达式，也可以用内存变量。内存变量可以是字符型C、数值型N、日期型D、逻辑型L中的任一种。关于参数的使用，将在1.3.3节中详细讨论。

因为LOAD命令在装入二进制模块时已经不再使用扩展名，所以CALL命令不需要指出扩展名。

第三步，释放.BIN模块。RELEASE MODULE命令用于从内存中释放二进制模块占用的内存，方法是：

RELEASE MODULE<二进制文件名>

使用该命令一方面可以使FoxBASE使用多于16个.BIN模块，另一方面，及时释放.BIN模块所占用的内存，能让FoxBASE有更多的内存空间用于其他工作。

RELEASE MODULE命令不支持通配符或ALL选项。当要释放多个.BIN模块时，必须逐个地释放。该命令也不需要指定扩展名，即使二进制文件不是使用默认的扩展名.BIN时也是如此。

CLEAR ALL或CLOSE ALL都不能使FoxBASE释放二进制模块。除RELEASE MODULE命令之外，唯一能让FoxBASE释放二进制模块的命令只有QUIT，QUIT在退出FoxBASE时同时释放二进制模块占用的内存。

使用RELEASE MODULE命令或QUIT命令释放.BIN模块时，必须先恢复由该.BIN模块改变了的重要的数据，例如中断向量表等，不然可能会使系统处于瘫痪。

1.1.2 FoxBASE对汇编语言子程序的要求

FoxBASE在调用汇编语言子程序时有严格的约定，在编写汇编语言模块时，必须遵守这些约定，只有这样，才能正确无误地调用.BIN模块，也只有这样，才能在调用之后顺利地返回FoxBASE。通常FoxBASE手册的说明有如下几点：

(1) 第一个可执行命令必须放在偏移量0处；

(2) 配置或使用的内存空间不可超过实际.BIN模块的长度，因为LOAD命令用文件的大小来确定需要分配的内存容量；

(3) .BIN程序不能增加或减少作为CALL…WITH…命令参数的内存变量的长度；

(4) 返回FoxBASE之前必须恢复CS和SS寄存器；

(5) .BIN程序结束时用远程返回指令RETF，将控制权返回给FoxBASE。

下面我们对这些约定做深入的分析和说明。

第(1)条约定让我们在程序开关处写一条伪指令ORG 0，以便程序从偏移量0开始。占据该存储单元的必须是一个可执行的指令，如果程序的开头打算作为数据区，那么此处应该用一个JMP指令跳转到程序开始处。不要指望汇编语言的END START之类的伪指令为我们安排程序开始的地址，因为FoxBASE总是从偏移量0处开始执行.BIN程序的。

第(2)条约定主要是指不要向.BIN模块之外的内存空间写数据，不然极有可能造成系统的错误或混乱。实际上，从.BIN模块空间之外读取数据总是安全的，而向.BIN模块空间之外写数据虽不一定安全，但又是必不可少的。在本书中，经常向显示存储区、BIOS数据区、甚至FoxBASE空间写数据，问题不是写不写，而是如何将数据写到安全处，绝不可以将FoxBASE搞乱。

第(3)条约定是至关重要的，.BIN程序不能改变作为调用参数的内存变量在FoxBASE

内存空间中的长度，不然会造成 FoxBASE 内存变量的混乱。关于这一点，在1.3.2节中还要仔细分析。

至于约定(4)，一般都可以自动满足。FoxBASE 用远调用 CALL FAR 方式调用.BIN 模块，而.BIN 模块用远返回 RETF 返回到 FoxBASE，所以 CS 寄存器总是自动恢复的。至于 SS 寄存器，除非很有必要，绝大多数的.BIN 程序并不切换堆栈段，而是使用 FoxBASE 的堆栈。既然 SS 未被改变过，所以也用不着保存和恢复。当然，使用 FoxBASE 的堆栈时要注意一点，那就是不要使堆栈溢出，这在编程时给予适当的注意就可以了。

最后，约定(5)当然是必须遵守的，就是说要将程序定义成一个远过程，并用 RETF 指令返回到 FoxBASE。

关于 FoxBASE 调用汇编语言子程序的问题在1.3.1至1.3.4节中还要进一步深入分析。

1.2 汇编语言的上机过程

1.2.1 汇编语言的上机过程

本节简单介绍汇编语言的上机过程，详细内容请读者自己参考有关书籍。

1. 编辑源程序

编辑源程序可以使用行编辑程序 EDLIN.COM 或字处理软件 WORDSTAR，CCED，WPS 等，使用字处理软件时应该用 N 方式，不要用 D 方式，以免 D 方式在源程序中插入软回车、分页符等，使汇编时出错。源程序文件扩展名通常是.ASM。

2. 汇编

汇编是把.ASM 源程序转换成供 LINK 使用的目标模块.OBJ，汇编用 MASM.EXE 完成。汇编时需一个输入文件.ASM，产生三个输出文件，即.OBJ，.LST 和.CRF。其中只有目标文件.OBJ 是最终生成.BIN 文件所必须的，其他两个文件可以帮助你调试程序，根据需要决定取舍。

3. 连接

连接程序 LINK.EXE 把汇编生成的目标文件.OBJ 转换成可执行文件，扩展名为.EXE。输入文件有两个，一个是.OBJ 文件，另一个是库文件.LIB，可以直接回车。输出文件也有两个，即.EXE 和.MAP，只有.EXE 是最终产生.BIN 文件所必需的。

连接时产生一个

```
LINK: warning L4021: no stack segment
```

(没有堆栈段)的报错信息，可不予理睬。

4. 转换成.BIN 文件

把.EXE 文件转换成.BIN 文件要用 EXE2BIN.EXE 程序，EXE2BIN 要求的输入文件是.EXE，输出文件默认的扩展名就是.BIN。

并不是所有的.EXE 文件都能转换为.BIN 文件。为了能转换成.BIN 文件，在编写源程序.ASM 时必须遵守一定的规则，详细的规定请读者参考汇编语言的有关书籍。本书中所有程序都是符合这些规定的。

1.2.2 第一个汇编语言程序

现在我们用实例说明汇编语言的上机过程。这个实例，就是我们的第一个汇编语言程序——SMALLCAP.ASM。这个程序的功能是将一个字符串中的所有小写字母转换成大写字母，对于非英文字母不做任何改变。该程序的功能类似于 FoxBASE 的 UPPER( )函数，但这是由.BIN 模块完成的。

第一步，用编辑软件(如 WS 等)将程序 SMALLCAP.ASM 输入并存盘。

```
;
;   程序名：SMALLCAP.ASM
;   用   途：将 FoxBASE 传入的参数变为大写后传回 FoxBASE
;   形   式：.BIN
;   日   期：1995.08
;
CODE        SEGMENT     BYTE PUBLIC          ;定义代码段
            ASSUME      CS:CODE              ;说明代码段
            ORG         0                    ;程序起始偏移量
START:
            PUSH        DS
            POP         ES                   ;ES 与 DS 同段
            MOV         SI,BX                ;初始化 SI
            MOV         DI,BX                ;初始化 DI
SMALLCAP1:
            LODSB                            ;取一字符
            CMP         AL,0                 ;是否结束符
            JZ          SMALLCAP3            ;是结束符，程序结束
            CMP         AL,"a"               ;是否<"a"
            JB          SMALLCAP2            ;小于，小写字母
            CMP         AL,"z"               ;是否>"z"
            JA          SMALLCAP2            ;大于，非小写字母
            AND         AL,5FH               ;小写转大写
SMALLCAP2:
            STOSB                            ;送回
            JMP         SHORT SMALLCAP1      ;下一个字符
SMALLCAP3:
            RETF                             ;返回 FoxBASE
CODE        ENDS                             ;代码段结束
            END         START                ;程序结束
```

第二步，汇编，做法是：

```
C:\>MASM
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

Source filename [.ASM]:SMALLCAP
Object filename [SMALLCAP.OBJ]:
Source listing [NUL.LST]:
Cross-reference [NUL.CRF]:

  49786 + 307041 Bytes symbol space free

    0 Warning Errors
    0 Severe Errors
```

汇编时共提问四个文件名，第一个是源文件名，回答 SMALLCAP，扩展名.ASM 可以省略。第

二个是目标文件名，系统已经给出了一个缺省的文件名 SMALLCAP.OBJ，所以直接回车即可。第三、四个文件名分别为列表文件和交叉引用文件名，默认的是空文件 NUL.LST 和 NUL.CRF，即不要这两个文件，所以通常是直接回车。然后开始进行汇编，并显示信息。

也可以用命令行方式进行汇编，方法是：

```
C:\>MASM SMALLCAP;
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

  49786 + 307041 Bytes symbol space free

    0 Warning Errors
    0 Severe Errors
```

其中 MASM 后的第一个参数是源文件名，紧跟着一个分号表示其余三个参数(即.OBJ 文件、.LST 文件和.CRF 文件)取缺省值。

如果汇编过程没有发生错误，则继续进行第三步——连接；若出现报错信息，则返回到第一步，根据报错信息修改源程序 SMALLCAP.ASM，然后重新汇编。

第三步，汇编正确之后就可以进行连接，由目标文件.OBJ 生成可执行文件.EXE，方法是：

```
C:\>LINK

Microsoft (R) Overlay Linker Version 3.64
Copyright (C) Microsoft Corp 1983-1988. All rights reserved.

Object Modules [.OBJ]: SMALLCAP
Run File [SMALLCAP.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
LINK : warning L4021: no stack segment
```

连接时首先提问目标文件名，输入 SMALLCAP，扩展名.OBJ 可以省略。接着提问可执行文件名，系统给出的默值值是 SMALLCAP.EXE，故可直接回车。提问的第三个文件名是连接程序列表文件，缺省值是空文件 NUL.MAP，即不要该文件，通常是直接回车取缺省值。第四提问库文件名，本书中并不使用库文件，所以直接回车。回答完文件名后开始进行连接工作，生成可执行文件 SMALLCAP.EXE。

连接程序也可以用命令行方式运行，方法是：

```
C:\>LINK SMALLCAP;
Microsoft (R) Overlay Linker Version 3.64
Copyright (C) Microsoft Corp 1983-1988. All rights reserved.

LINK : warning L4021: no stack segment
```

其中命令行中的第一个参数为目标文件名，分号表示其余的三个参数取缺省值。

通常连接产生的.EXE 文件可以在 DOS 提示符下键入并运行。但我们的第一个汇编语言程序 SMALLCAP.ASM 生成的可执行文件 SMALLCAP.EXE 却不能在 DOS 下正确运行，这是因为我们的源程序 SMALLCAP.ASM 是按.BIN 文件的要求编写的，而不是按.EXE 文件的要求编写的。所以对于我们来说，.EXE 文件只是一个中间产物，最终我们要的是.BIN 文件，因此还要进行第四步，转换成.BIN 文件。关于.EXE 文件和.BIN 文件的不同之处，在第

1.4.3节中还要加以说明。

第四步，转换成.BIN 文件，方法是：

```
C:\>EXE2BIN SMALLCAP
```

命令行中的第一个参数是.EXE 文件名，扩展名可以省略；第二个参数是转换后的文件名，缺省时转换后的文件名主名与.EXE 文件主名相同，扩展名为.BIN，我们就用这种缺省的方式。

由于我们的源程序 SMALLCAP.ASM 是按照.BIN 文件的格式编写的，所以转换不会出错，不然可能会报告一些出错信息。

至此，产生.BIN 文件的全部过程已经完成，看一看目录：

```
C:\>DIR SMALLCAP.*
 Volume in drive C is 12345678901
 Directory of C:\

SMALLCAP BIN        25     3-26-96    6:12a
SMALLCAP OBJ        86     3-26-96    6:12a
SMALLCAP ASM      1264     3-26-96    6:08a
SMALLCAP EXE       537     3-26-96    6:12a
        4 File(s)     5218304 bytes free
```

其中 SMALLCAP.OBJ，SMALLCAP.EXE 是中间产物，可以删去。

1.2.3 .BIN 程序的调用方法

调用.BIN 程序要在 FoxBASE 中进行，以 SMALLCAP.BIN 为例，方法如下：

```
C:\>MFOXPLUS
. LOAD SMALLCAP
. A="123ABCmnop;$"
. CALL SMALLCAP WITH A
. ? A
123ABCMNOP;$
RELEASE MODULE SMALLCAP
. QUIT
C:\>
```

可见 SMALLCAP.BIN 达到了预定的设计目的，将小写字母转换为大写字母，对其他字符没做任何变换。

SMALLCAP.BIN 是一个带有返回参数的.BIN 程序，如果把它作为一个函数，则调用起来更加方便。下面是自定义函数 SMALLCAP( )：

```
* 自定义函数 SMALLCAP.PRG
* 调用前先 LOAD SMALLCAP
PARAMETERS X
CALL SMALLCAP WITH X
RETURN X
```

调用方法是：

```
? SMALLCAP("123ABCmnop;$")
或
A="123ABCmnop;$"
B=SMALLCAP(A)
? B
```

123ABCMNOP;$

有些读者可能会产生这样的疑问:既然SMALLCAP.BIN与函数UPPER( )功能相同,又何必编这样一个程序呢?至少有两个理由,其一,这是我们的第一个汇编语言程序,着重用以说明汇编语言上机过程,程序本身的功能当然不能太复杂。其二,相当重要的,UPPER( )是FoxBASE系统提供的内部函数,你无法改变它的功能,而SMALLCAP.BIN是我们自己开发的,它可以完成我们希望的功能。例如,如果我们愿意,可以输入一个"1",返回一个"男",输入一个"2",返回一个"女",输入一个"bjdx"返回一个"北京大学"等等,可以说是"心想事成"。无论FoxBASE的命令/函数有多少,总不会满足程序员的所有要求,要扩充FoxBASE的功能,就靠.BIN模块。本书的目的就是向读者介绍如何进行.BIN格式的汇编语言编程。

1.2.4 用批文件完成汇编

汇编、连接、转换工作总是反复进行的,特别是源程序有错误时,就要多次地编辑、汇编、连接、转换、试运行。汇编、连接、转换工作可以用一个批文件来完成:

```
@ ECHO OFF
REM ASMTOBIN.BAT
IF %1!==! GOTO NOFILENAME
IF NOT EXIST %1.ASM GOTO NOASM
MASM/Z %1;
IF NOT EXIST %1.OBJ GOTO NOOBJ
LINK %1;
IF NOT EXIST %1.EXE GOTO NOEXE
EXE2BIN %1
IF EXIST %1.BAK DEL %1.BAK
DEL %1.OBJ
DEL %1.EXE
DIR %1.*
GOTO END
:NOFILENAME
ECHO NO ASM FILENAME
ECHO Usage: ASMTOBIN FILENAME
GOTO END
:NOASM
ECHO %1.ASM NOT EXIST
GOTO END
:NOOBJ
ECHO %1.OBJ NOT EXIST (MASM ERROR?)
GOTO END
:NOEXE
ECHO %1.EXE NOT EXIST (LINK ERROR?)
:END
```

在批文件ASMTOBIN.BAT的第5行MASM/Z %1;中的开关/Z告诉汇编程序MASM.EXE当汇编出错时,不但显示报错信息,而且显示错误所在源程序行和行号。

批文件ASMTOBIN.BAT的用法是(以SMALLCAP.ASM为例):

```
ASMTOBIN SMALLCAP
```

其中命令行中的参数必须省略扩展名.ASM。该批文件将顺序依次完成汇编、连接、转换、删除中间文件的工作,并具有错误处理能力。