

第3波

计算机语言技术系列丛书(二)

Quick BASIC

编程技巧

林祚仪编著



学苑出版社

■ 以实例为引导, 易学易用

■ 内容详尽, 应用面广

■ 随书附学习盘

计算机语言技术系列丛书(二)

QuickBASIC 编程技巧

林祚仪 编著
王真华 改编
燕卫华 审校

学苑出版社

1994

(京)新登字 151 号

内 容 提 要

本书详细介绍了 QuickBASIC 与 Assembly 连接技术;M/G 单色卡与中文字型读取;扩充内存的使用技术;文件及磁盘的保护技术;各种图形文件格式转换技术;计算机与计算机间的通讯;DBF 文件的数据读取。每章都有详细的内容与实例。

本书面向从事计算机研究的广大用户,可供大专院校计算机专业师生参考,也适合于广大计算机用户阅读。

欲购本书的用户,请直接与北京 8721 信箱联系,邮政编码 100080,电话 2562329。

版 权 声 明

本书繁体字中文版原书名为《Quick BASIC 专题实作》,由第 3 波文化事业股份有限公司出版,版权归第 3 波公司所有。本书简体字中文版由第 3 波公司授权北京希望电脑公司和学苑出版社独家出版、发行。未经出版者书面许可,本书的任何部分均不得以任何形式或任何手段复制或传播。

计算机语言技术系列丛书(二)

QuickBASIC 编程技巧

编 著:林祚仪
改 编:王真华
审 校:燕卫华
责任编辑:甄国亮
出版发行:学苑出版社 邮政编码:100036
社 址:北京市海淀区万寿路西街 11 号
印 刷:兰空印刷厂
开 本:787×1092 1/16
印 张:9.125 字 数:207 千字
印 数:1~5000 册
版 次:1994 年 7 月北京第 1 版第 1 次
ISBN7-5077-0905-1/TP·29
本册定价:13.00 元

学苑版图书印、装错误可随时退换

目 录

第一章 QuickBASIC 与 Assembly 连接技术	(1)
1.1 QuickBASIC 与汇编语言连接	(1)
1.2 编写汇编语言子程序的重点	(2)
1.3 利用 CALL ABSOLUTE 调用汇编语言子程序	(3)
1.3.1 CALL ABSOLUTE 用法说明	(3)
1.3.2 使用 BLOAD 装入子程序及执行	(4)
1.3.3 使用 POKE、READ 装入子程序及执行	(7)
1.4 使用 CALL 来调用子程序	(9)
1.5 使用 CALL 来调用汇编语言子程序实例	(9)
1.5.1 调用实例一:两数相加	(9)
1.5.2 调用实例二:印出一字串	(11)
1.5.3 调用实例三:字符页存取	(13)
1.5.4 调用实例四:弹出式画面设计	(17)
第二章 M/G 单色卡与中文字型读取	(20)
2.1 M/G 单色卡	(20)
2.2 单色字符页的存取	(20)
2.2.1 屏幕的显示	(20)
2.2.2 字符视频页(VIDEO BUFFER)	(20)
2.2.3 屏幕上字符的存取	(22)
2.3 中文字型文件及中文内码	(25)
2.4 内码与中文转换程序	(26)
2.5 读取并显示中文字型	(28)
2.6 制作及读取自用字型文件	(33)
2.7 使用自用字型文件	(35)
2.7.1 打开字型文件模块	(35)
2.7.2 显示字型模块	(36)
2.7.3 关闭字型文件模块	(36)
2.7.4 三个模块使用重点范例	(37)
2.7.5 实际应用	(38)
第三章 扩充内存使用技术	(43)
3.1 扩展内存(Extended Memory)的了解	(43)
3.1.1 总线(Bus)	(43)
3.1.2 内存	(43)
3.1.3 80286 以上的寻址模式	(44)
3.2 BIOS 提供的扩展内存存取接口(INT 15h)	(44)

3.2.1	移动内存块(INT 15h,功能 87h)	(44)
3.2.2	返回扩展内存大小(INT 15h,功能 88h)	(46)
3.2.3	在保护模式下执行程序(INT 15h,功能 89h)	(47)
3.3	利用 QuickBASIC 调用扩展内存存取接口(INT 15h)	(47)
3.3.1	取得扩展内存的大小子程序	(47)
3.3.2	QuickBASIC 对 GDT(全域描述表)的声明	(47)
3.3.3	QuickBASIC 存取扩展内存子程序	(48)
3.3.4	存取扩展内存应用实例	(52)
3.4	XMS 扩展内存规格	(54)
3.4.1	XMS 接口的启动	(54)
3.4.2	QuickBASIC 使用 XMS 功能调用的困扰及解决	(55)
3.4.3	XMS 服务功能列表与实例	(58)
3.5	扩展内存(Expanded Memory)的了解	(68)
3.5.1	扩展内存的来源	(68)
3.5.2	扩展内存接口(EMM)的安装	(68)
3.5.3	INT 67h (EMM 应用程序接口 API)功能列表与实例	(68)
第四章	文件及磁盘保护技术	(79)
4.1	文件的保护	(79)
4.1.1	简单代换编码法	(79)
4.1.2	XOR 编码法	(85)
4.2	防止程序及文件被拷贝	(87)
4.2.1	文件防拷技术——BIOS 检查法	(87)
4.2.2	文件防拷技术——CMOS 检查法	(88)
4.2.3	磁盘防拷技术——超道防拷法	(91)
4.2.4	磁盘防拷技术——无缝锁	(98)
第五章	各种图形文件格式转换技术	(103)
5.1	QuickBASIC (BASIC) BSAVE 所存的图形文件	(103)
5.2	DVC 区块文件(.BLK)及图形文件(.DVC)格式	(105)
5.2.1	区块文件(.BLK)	(105)
5.2.2	图形文件(.DVC)	(106)
5.3	Paint_Brush 的 PCX 文件格式	(110)
5.4	抓图大师的. BPC 文件格式	(112)
5.5	Rambow PAINT 的. MAP 文件格式	(114)
5.6	变影 72 的. PUT 文件	(115)
第六章	通讯	(117)
6.1	数据传送方式	(117)
6.2	RS-232C	(118)
6.3	QuickBASIC 的控制串行端口指令	(118)
6.3.1	在 QuickBASIC 中串行端口中,是使用 OPEN 来控制的	(118)

6.3.2 其他的控制指令.....	(120)
6.4 接收与传送	(121)
6.5 通讯实例	(122)
6.5.1 两人联机会话.....	(122)
6.5.2 猜数字游戏.....	(125)
第七章 DBF 读取	(130)
7.1 DBF 文件格式跟踪	(130)
7.2 简单的 DBF 文件的读取	(133)
7.3 实例应用	(136)
附录 A EMS 返回的状态码及错误码	(139)
附录 B Q3.31 文件的内容	(140)

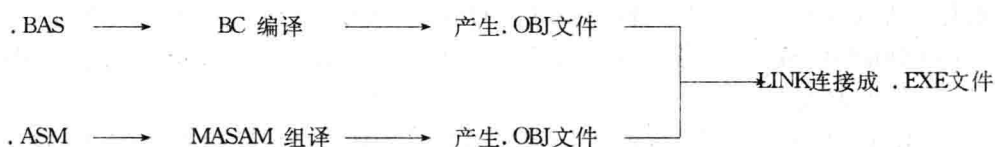
第一章 QuickBASIC 与 Assembly 连接技术

1.1 QuickBASIC 与汇编语言连接

由于 QuickBASIC 在循环处理方面与 ASSEMBLY、C、PASCAL 等语言比起来慢很多,当我们用 QuickBASIC 去编写大循环处理的子程序(如:移动或拷贝内存等)时,虽然是可以办到,但在速度上却非常慢,毫无效率可言;或许,你可以去换最新的 CPU,但速度会增快多少呢?!而如果利用 ASM 来编写大循环处理的子程序,再利用 QuickBASIC 与其连接的话,那可就不平凡了。

语言连接

汇编语言可以与 QuickBASIC 连接吗?答案是肯定的。虽然语言不同,但在 Microsoft 的系统中,大部分的语言所编成的 .OBJ 文件形式是相同的,LINK 程序并不会去区别 .OBJ 文件是以哪种语言编成的;所以,只要 .OBJ 文件格式相同,这些文件都可被连接在一起。



循环子程序比较:

比较项目	QB 编写与编译后执行	ASM 编写与 QB 连接后执行
子程序 编写	<pre> SUB LOOPTEST FOR I=1 TO 1000 FOR J=1 TO 1000 NEXT J NEXT I END SUB </pre>	<pre> .MODEL MEDIUM .CODE PUBLIC LOOPTEST LOOPTEST PROC push bp mov bp,sp mov cx,1000 lp1: push cx mov cx,1000 lp2: loop lp2 pop cx loop lp1 pop bp ret LOOPTEST ENDP END </pre>

比较项目	QB 编写与编译后执行	ASM 编写与 QB 连接后执行
主程序~ 编写	A=TIMER CALL LOOPTEST B=TIMER PRINT "Time: ";B-A	A=TIMER CALL LOOPTEST B=TIMER PRINT "Time: ";B-A
执行机器	80486-33	80486-33
执行结果	Time:5.219188(秒)	Time:0.328125(秒)
文件大小	27218 bytes	26990 bytes

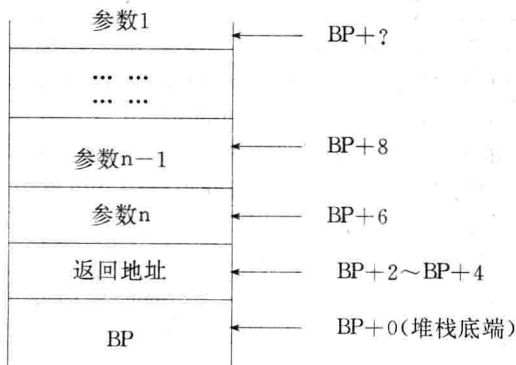
编写流程

1. 编写 ASM 子程序,需注意编写时的一些限制及重点。
2. 选定在主程序中调用执行子程序的方法:
 - (1) CALL ABSOLUTE 调用汇编语言子程序。
 - (2) CALL 来调用汇编语言子程序。

1.2 编写汇编语言子程序的重点

虽然 DOS 系统下各种语言编译后的 .OBJ 文件格式相同,但是语言与语言间的调用及参数传送规则却不一定。以下就是 ASM 与 QuickBASIC 相互调用的重点及规则:

- (1) QuickBASIC 的预设内存模式是中型模块(Medium),因此必须采用远程调用与返回的方式。
- (2) QuickBASIC 调用汇编语言子程序时,是以传址法来传递参数,所产生的堆栈规则为:



最后一个参数位于[BP+6],倒数第二个则位于[BP+8],以下类推。

- (3) QuickBASIC 执行汇编语言子程序要返回时,必须使用 RET n 返回并清除参数的方式(n 表示子程序参数占用堆栈的个数)。

例:要传递两个整型变量,因共占 4 bytes,所以返回时须使用 RET 4。

- (4) 程序实例

```

;Environment :Macro Assembly 5.1
.MODEL MEDIUM           ;设定中型模块
.CODE
public test
test PROC

    push bp
    mov bp,sp             ;指向堆栈顶端

    mov ax,[bp+6] ;取得第 2 个参数
    mov bx,[bp+8] ;取得第 1 个参数

    pop bp                ;返回原程序(QuickBASIC 程序)
    ret 4                 ;因为 2 个参数占 4 Bytes

test ENDP
end

```

1.3 利用 CALL ABSOLUTE 调用汇编语言子程序

1.3.1 CALL ABSOLUTE 用法说明

语 法

```

DEF SEG=Segment
CALL ABSOLUTE ([参数],[位移])

```

说 明

1. 此调用用于执行内存内某一段汇编语言子程序。
2. 执行此功能前,必须先利用 DEF SEG 将机器语言程序段地址定出来。
3. “位移”:程序段开头到子程序开头之间的位移地址。
4. “参数”:预备传给汇编语言子程序的变量。
5. 如要在 QB 编辑器中执行 CALL ABSOLUTE,必须连 QB.QLB 一起装入 QB.EXE 中。

执行例

```

DEF SEG=&H9000          '设定 Segment 为 9000:0000
CALL ABSOLUTE(&H100) '执行由 9000:0100 开始存放的汇编语言子程序

```

注意:子程序(ABSOLUTE())并不包含在 BCOM4x.LIB 中,而是存放在 QB.LIB 及 QB.QLB 内,因此程序中如有使用到 CALL ABSOLUTE()时,必须使用以下的方法才可以让程序顺利执行。

外部编译时

```
C:\QB>BC/O filename
```

```
C:\QB > LINK /EX filename,,, QB.LIB
```

内部编辑环境使用时

```
C:\QB > QB/L QB.QLB filename
```

1.3.2 使用 BLOAD 装入子程序及执行

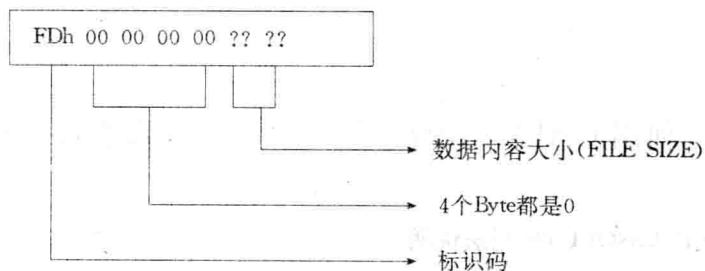
[步骤 1] 利用 DEF SEG 指定程序段

[步骤 2] 利用 BLOAD 将机器语言子程序装入程序段内

[步骤 3] 利用 CALL ABSOLUTE 执行程序段内的程序

如果你使用此方法调用子程序时,由于必须用到 BLOAD,因此你的子程序文件前面必须符合 BLOAD 规定的文件头,也就是说,前面一定要多加入 7 Bytes 的标识码,当你使用 BLOAD 读子程序时,才能顺利读入。

文件头结构(7 Bytes)



调用实例

使用 BLOAD 装入实例及使用 CALL ABSOLUTE 调用实例

QuickBASIC 主程序(行号可省略)

```
1 'PROGRAM ID:P0101.BAS
2 '
3 '外部编译本程序时: BC/O P0101.BAS;
4 '          LINK /EX P0101.BAS,,, QB.LIB
5 '
6 '要在内部执行时需装入 QB.QLB:QB/L QB.QLB P0101.BAS
7 '
8 '$INCLUDE :'QB.BI'
9 PRINT "This is a DEMO program"
10 FILES
11 DEF SEG=&H8000
12 BLOAD "P0101.BIN";0
13 CALL ABSOLUTE (0)
```

程序说明

行号 8 :将 QB.BI 含入程序文件中

行号 11 :利用 DEF SEG 将段设定到 8000:0000

行号 12 :利用 BLOAD 将汇编语言读到内存内

行号 13 :执行放置在内存 8000:0000 的汇编语言子程序

汇编语言编写的子程序

```
1      ;
2      ; PROGRAM-ID. P0101. ASM
3      ;
4      ;子程序功能:将屏幕上的字符左右颠倒
5      ;
6      CODE_SEG_A      SEFMENT
7                      assume cs:code_seg_a
8
9      JOKE PROC FAR
10
11      db 0fdh      ;标识码
12      db 0,0,0,0      ;4 Byte 的 0
13      dw PGMSIZE;      ;内容大小
14      push bp
15      mov bp,sp
16      push ds
17      push ax
18      push bx
19      push cx
20      push dx
21      push si
22      push es
23      mov ax , 09000h
24      mov es ,ax
25      mov di,0
26      mov si,0
27      mov ax,0b000h
28      mov ds,ax
29      mov cx,4000h
30      rep movsb
31      mov ax,0b000h
32      mov es,ax
33      mov ax,09000h
34      mov ds,ax
35      mov cx,158
36 LP3:
37      push cx
38      mov di,cx
```

```
39      mov si,158
40      sub si,cx
41      mov cx,24
42  LP4:
43      push cx
44      mov cx,1
45      rep movsb
46      sub di,1
47      sub si,1
48      add di,160
49      add si,160
50      pop cx
51      loop LP4
52      pop cx
53      loop LP3
54      mov di,cx
55      mov si,158
56      sub si,cx
57      mov cx,24
58  LP5:
59      push    cx
60      mov     cx,1
61      rep     movsb
62      sub     di,1
63      sub     si,1
64      add     di,160
65      add     si,160
66      pop     cx
67      loop    LP5
68      pop     es
69      pop     si
70      pop     dx
71      pop     cx
72      pop     bx
73      pop     ax
74      pop     ds
75      pop     bp
      ;
      ;并无参数传入及返回
      ;
76      ret 0
77  JOKE ENDP
78  PGMSIZE EQU $-JOKE-7      ;自动计算内容长度
```

```

79      code_seg' a ends
80      END JOKE

```

程序说明

行号 11~13: 模拟 BLOAD 文件头

```

db 0fdh      ; 标识码(固定)
db 0,0,0,0   ; 4 Byte 的 0
dw PGMSIZE   ; 内容大小自动由程序算出

```

行号 14~15: 将 sp 值传给 bp, 以便使参数可通过 bp 加上固定位移来加以定值。

行号 16~74: 主要子程序

行号 75~76: 子程序结束, 将 bp 弹出, 并返回主程序

行号 78 : 自动计算文件长度

程序编写完后, 必须将 .ASM 编成 .BIN 文件, 其程序如下:

```

A>MASM P0101.ASM;      ←对原始程序进行组译,产生.OBJ 文件
A>link P0101;          ←对.OBJ 文件进行连接,产生.EXE 文件
A>exe2bin P0101        ←利用 EXE2BIN 将.EXE 转成.BIN 文件

```

1.3.3 使用 POKE、READ 装入子程序及执行

使用此法前, 必须先将子程序组译(Assemble)过后的机器码取出, 再一个个依次建立 DATA。



[步骤 1] 利用 DEF SEG 指定程序段

[步骤 2] 使用 READ 将 DATA 内的机器码读出后, 再用 POKE 将机器码一个个的填入指定的内存段内

[步骤 3] 利用 CALL ABSOLUTE 执行内存内的程序

调用实例

利用 POKE 及 CALL ABSOLUTE 的调用实例:

此法是将机器语言子程序内容和主程序合为单一文件, 做法就是将组译(Assemble)过后的机器码取出, 取出后再利用 DATA 将它定成主程序的数据, 主程序再利用 READ 去读取 DATA 里面的数据。读取之后, 最后利用 POKE 将程序码填入内存中, 如此便可达到和 BLOAD 相同的作用。

以下是 P0101. BIN 的程序码,其中前面 7 Bytes 是 BLOAD 标识用的文件头,使用此法时并不需要此 7 bytes,所以可除去。

此段不是程序码,可除去

```

FD,00,00,00,00,7B,00,55,8B,EC,1E,50,53,51,52,56
06,B8,00,90,8E,C0,BF,00,00,BE,00,00,B8,00,B0,8E
D8,B9,00,40,F3,A4,B8,00,B0,8E,C0,B8,00,90,8E,D8
B9,9E,00,51,8B,F9,BE,9E,00,2B,F1,B9,18,00,51,B9
01,00,F3,A4,83,EF,01,83,EE,01,81,C7,A0,00,81,C6
A0,00,59,E2,E9,59,E2,DB,8B,F9,BE,9E,00,2B,F1,B9
18,00,51,B9,01,00,F3,A4,83,EF,01,83,EE,01,81,C7
A0,00,81,C6,A0,00,59,E2,E9,07,5E,5A,59,5B,58,1F
5D,CB,3E

```

QuickBASIC 主程序

```

1  '
2  ' PROGRAM-ID. P0102. BAS
3  ' $ INCLUDE: 'QB. BI'
4  DEF SEG=&H8000
5  PRINT "This is a DEMO"
6  FILES
7  '
8  '将程序码填至内存 8000:0000~
9  '
10  FOR I=0 TO &H7A
11  READ VALUE$
12  POKE I,VAL("&H"+VALUE$)
13  NEXT I
14  CALL ABSOLUTE(0)
15  DATA 55,8B,EC,1E,50,53,51,52,56
16  DATA 06,B8,00,90,8E,C0,BF,00,00,BE,00,00,B8,00,B0,8E
17  DATA D8,B9,00,40,F3,A4,B8,00,B0,8E,C0,B8,00,90,8E,D8
18  DATA B9,9E,00,51,8B,F9,BE,9E,00,2B,F1,B9,18,00,51,B9
19  DATA 01,00,F3,A4,83,EF,01,83,EE,01,81,C7,A0,00,81,C6
20  DATA A0,00,59,E2,E9,59,E2,DB,8B,F9,BE,9E,00,2B,F1,B9
21  DATA 18,00,51,B9,01,00,F3,A4,83,EF,01,83,EE,01,81,C7
22  DATA A0,00,81,C6,A0,00,59,E2,E9,07,5E,5A,59,5B,58 1F
23  DATA 5D,CB

```

程序说明

行号 3 :将 QB. BI 含入程序中
 行号 4 :设定节地址为 8000

行号 10~13 :将程序码由内存 8000:0000 开始填入

行号 14 :执行内存 8000:0000 开始的程序

行号 15~23 :子程序的程序码(16 进制),此处由 P0101.BIN 取出

1.4 使用 CALL 来调用子程序

CALL 用法说明

语法 CALL 子程序名称([传给子程序的参数])

说明 调用执行子程序功能

例 CALL NEWPRINT(X,Y,A\$) '执行子程序 NEWPRINT,并传送
'X;Y;A\$ 等参数给子程序使用。

注意事项

1. 此法不须用到 BLOAD 指令,所以在编写汇编语言程序时,不须加入 BLOAD 标识用的文件头。
2. 在汇编语言程序上,必须包括一个 PUBLIC 声明,以便当我们利用 LINK(连接程序)产生可执行文件时,LINK 能够根据 PUBLIC 所声明的名称,对应到 QuickBASIC 程序中的同一名称。
3. 如果要让子程序能够在 QuickBASIC 编辑器内执行的话,必须先将子程序编成 .LIB 文件及 .QLB 文件。两文件产生方式如下:
 - (1)产生 .LIB 文件
LIB MYLIB,xxxxx.OBJ;
xxxxx.OBJ 文件为子程序的 .OBJ 文件,执行完后会自动产生 MYLIB.LIB 文件,下次 QuickBASIC 要用到此子程序时,就可利用 LINK 直接与 MYLIB 连接即可。
 - (2)产生 .QLB 文件
LINK/QU xxxxx.OBJ,MYLIB.QLB,NUL,BQLB40.LIB(或 BQLB45.LIB)
xxxxx.OBJ 文件为子程序 .OBJ 文件,执行完后会自动产生 MYLIB.QLB 文件,如果要在 QB 编辑模式下执行此子程序的话,只需在进入 QB 编辑模式前,先将 MYLIB.QLB 装入即可。装入方法:QB/L MYLIB.QLB。
4. QuickBASIC 传送字符串参数给子程序时,须特别小心。因为 QuickBASIC 的字符串格式是由四个字节的描述字来描述字符串储存的方式,并不是字符串本身数据。

字符串长度 (2 bytes)	字符串地址 (2 bytes)
-----------------	-----------------

1.5 使用 CALL 来调用汇编语言子程序实例

1.5.1 调用实例一:两数相加

QuickBASIC 主程序

```
DECLARE SUB PLUS(A%,B%,VALUE%)
```

```
' PROGRAM-ID. P0103.BAS
```

```
' 外部编译时:
```

```
' C>LINK P0103.BAS+PLUS.OBJ
```

```
A% = 100
```

```
B% = 500
```

```
CALL PLUS(A%,B%,TOTAL%)
```

```
PRINT A%;"+";B%;"=";TOTAL%
```

汇编语言子程序

```
;
```

```
; PROGRAM-ID. PLUS.ASM
```

```
;
```

```
;产生.OBJ的方法:C>MASM PLUS;
```

```
;
```

```
.MODEL MEDIUM
```

```
.CODE
```

```
PUBLIC PLUS
```

```
;
```

```
;
```

```
; PLUS (A%, B%, VALUE%)
```

```
;
```

```
;
```

```
;
```

```
;
```

```
;VALUE%为 A%+B%后的结果
```

```
;
```

```
PLUS PROC
```

```
push bp
```

```
mov bp,sp
```

```
;
```

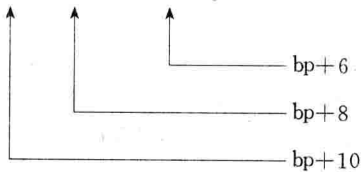
```
;取得第1个参数 A%
```

```
;
```

```
mov bx,[bp+10]
```

```
mov ax,[bx]
```

```
;
```



```

;取得第 2 个参数 B%
;
mov     bx,[bp+8]
mov     cx,[bx]
mov     bx,[bp+6]
;
;A% + B%
;
add     ax,cx
;
;将加完后的结果传给第 3 个参数 C%
;
mov     [bx],ax
pop     bp
;
; 返回
;
ret     6

PLUS ENDP

END

```

1.5.2 调用实例二:印出一字串

QuickBASIC 主程序

```

DECLARE SUB PRINTS (Y%,X%,Length%,Address%)
'
' PROGRAM-ID. P0104.BAS
'
' 外部编译时:
' C> LINK P0104.BAS+PRINTS.OBJ
'
'
X%=10
Y%=10
A$="Hello World !"

FOR Y%=1 TO 23
CALL PRINTS(Y%,1,LEN(A$),SADD(A$))
NEXT Y%

```

说明:1. 利用 LEN() 可取得字符串长度

2. 利用 SADD()可取得字符串真正的地址