8086asm 用户手册

> 本手册最新版本地址: https://hyan23.org/tech/2018/09/21/8086asm-user-manual.html

目录

1 使用本函数库

要使用本函数库定义的函数, 有以下两种方式:

1. 将需要使用的函数库`*.inc`文件复制到个人工作目录下, 并在代码段空白区域插入一行

   ```
   include 文件名.inc
   ```

   以包含单个库文件;

2. 对照用户手册-函数列表, 在对应库文件中找到原函数定义, 复制从(包含)

   ```
   函数名 proc near
   ```

   到

```
函数名 endp
```

的所有内容到代码段空白区域中. 在复制时, 也要注意该函数是否调用了其他函数(这种情况极少, 每个文件外调用我都标注了被调函数的所属文件名), 并把被调函数一并复制到个人代码中.

> 注: 建议使用第二种方式. 这样, 在库函数接口发生变化时不会影响到已写好的程序.

2 编程规范

1. 一般, ax寄存器作为函数的参数和返回值来使用
2. 与地址有关的函数参数, 若无特别说明, 其段寄存器假定为ds, 即: 若函数参数声明为dx, 函数将从 ds:dx 取得数据
3. 用 dx 寄存器指向只读内存区域, bx寄存器指向可写内存区域
4. 字符串以'$'符号结尾
5. 32位寄存器组合: dx:ax, 64位寄存器组合: dx:ax:cx:bx
6. 在调用 file.inc 内的子程序时, 需同时include common.inc, 该文件包含被 file.inc 使用的常量定义
7. 还有一些返回值为布尔类型的函数, 它们将使用 common.inc 中定义的 @TRUE 和 @FALSE 常量
8. 另有一些对数组操作的函数, 它们使用 common.inc 中定义的 @WORD 常量

对 common.inc 里定义的常量, 现作说明如下:

常量名	值	含义
@WORD	2	字的大小
@TRUE	1	逻辑真
@FALSE	0	逻辑假

下面列出的这一组常量针对 file.inc 中的 `open` 函数, 作为第二个参数使用 ax 传入, 用来指示文件打开方式, @ORD代表只读, @OWR代表只写, @ORDWR代表可读可写

```
; file.inc.open
@ORD         equ              0
@OWR         equ              1
@ORDWR       equ              2
```

下面列出的这一组常量针对 file.inc 中的 `seek` 函数, 作为第二个参数使用 cx 传入, 用来指示移动模式, @SEEK_SET表示移动相对于文件首, @SEEK_CUR表示移动相对于当前位置, @SEEK_END表示移动相对于文件尾, 具体用法可参考c语言 `fseek` 函数

```
; file.inc.seek
@SEEK_SET    equ              0
@SEEK_CUR    equ              1
@SEEK_END    equ              2
```

3 函数列表

> 注: 函数名前带 '*' 标记表示该函数为函数库内部使用, 用户不应单独调用这些函数.

3.1 基本输入输出

源代码文件名: iopt.inc

函数名	功能	参数列表	返回值/结果	备注
clear	清屏			
cursor	设置光标位置	ah: 行号, al: 列号		参数说明: 行号, 列号的取值从0开始, 取值的范围取决于当前显示模式. 默认显示模式为: 25*80, ah的取值范围为0-24, al的取值范围为0-79
putc	以指定属性在光标位置打印一个字符	al: 待打印字符, bl: 属		

		性值		
space	在光标位置打印一个空格			
crlf	打印回车(CR, 0dh)换行(LF, 0ah), 这会导致光标跳到下一行行首			
getch	从键盘输入一个字符		ax: 输入的字符	1. 该函数无需输入确认, 即无需在输入完成后键入一个额外的'回车'或'空格'空白字符来结束输入, 也就是说, 空白字符也会被读入; 2. 返回值 ax 高8位总为零
gets	从键盘输入一个字符串, 以'回车'结束, 并追加'$'结束标记	bx: 缓冲区地址	bx: 输入的字符串	
iptpwd	从键盘输入一个密码字符串, 以'回车'结束, 并追加'$'结束标记. 与gets不同的是, 此函数无回显	bx: 缓冲区地址	bx: 输入的字符串	
pause	打印一行提示, 等待用户键入'回车'			
optch	在光标位置打印一个字符	al: 待打印字符		
iptch	从键盘输入一个字符, 需键入'回车'和'空格'空白字符确认		ax: 输入的	返回值 ax 高8位总为零

			字符	
optstr	在光标处打印一个以'$'符号结束的字符串	dx: 待打印字符串		
optstrq	在光标处打印一个以'$'符号结束的字符串, 并在两端添加引号	dx: 待打印字符串		
puts	在光标处打印一个以'$'符号结束的字符串, 并回车换行	dx: 待打印字符串		
putsq	在光标处打印一个以'$'符号结束的字符串, 并在两端添加引号, 并回车换行	dx: 待打印字符串		
optdec	在光标处以十进制打印一个数	ax: 待打印的数		范围: 0-65535
optdecdw	在光标处以十进制打印一个数	dx:ax 待打印的数		范围: 0-4294967295
iptdec	从键盘输入一个十进制数, 需键入'回车'和'空格'空白字符确认		ax: 输入的数字	
optdecs	在光标处打印一个带符号十进制数	ax: 待打印的数		范围: -32768-32767
iptdecs	从键盘输入一个带符号的十进制数, 需键入'回车'和'空格'空白字符确认		ax: 输入的数字	
*hex2ascii				内部使用
*opt0x				内部使用

opthex	在光标处以十六进制打印一个数	ax: 待打印的数		
opthexb	以十六进制打印一个字节	al: 待打印的字节		附带前导’0x’
opthexw	以十六进制打印一个字	ax: 待打印的字		附带前导’0x’
opthexdw	以十六进制打印一个双字	dx:ax 待打印的双字		附带前导’0x’
opthexqw	以十六进制打印一个四字	dx:ax:cx:bx 待打印的四字		附带前导’0x’
*ascii2hex				内部使用
ipthex	从键盘输入一个十六进制数		ax: 输入的数	
optbin	在光标处以二进制打印一个数	ax: 待打印的数		
iptbin	从键盘输入一个二进制数		ax: 输入的数	
optbcd	在光标处以压缩 BCD 码打印一个数	ax: 待打印的数		最高数位: 4位
iptbcd	从键盘以 BCD 码输入一个数, 存储为压缩 BCD 形式		ax: 输入的数	最高数位: 4位
printArray	在光标处以空格隔开打印一个无符号字数组	dx: 数组首地址, cx: 元素个数		

printArrayS	在光标处以空格隔开打印一个带符号字数组	dx: 数组首地址, cx: 元素个数		

3.2 字符串

源代码文件名: string.inc

> 注: C风格字符串, 即以数字 0 作为结束符的字符串; DOS风格字符串, 即以字符'$' 作为结束符的字符串, 本函数库中, 与文件操作有关的函数对参数的要求是 C 风格字符串, 其他函数则要求传入 DOS 风格字符串.

函数名	功能	参数列表	返回值/结果	备注
convert	把一个 C 风格字符串转换为 DOS 风格字符串	bx: 待转换的字符串	bx: 转换后的字符串	
convertC	把一个 DOS 风格字符串转换为 C 风格字符串	bx: 待转换的字符串	bx: 转换后的字符串	
lower	把给定字符串里的所有英文字母转为小写	bx: 待转换的字符串	bx: 转换后的字符串	
upper	把给定字符串里的所有英文字母转为大写	bx: 待转换的字符串	bx: 转换后的字符串	
strlen	计算字符串的长度	dx: 待计算的字符串	ax: 字符串长度	可应用于两种风格的字符串
strequ	求字符串是否相等	dx: 字符串1, bx: 字符串2	ax: 布尔类型结果	
strcmp	以 ASCII 顺序比较两个字符串的大小	dx: 字符串1, bx: 字符串2	ax: 有符号整	返回值说明: 0代表两字符串相等, -1代表字符串1小于字符串2, 1代表字符串1大于字符串2

			数类型结果	
strcpy	字符串拷贝	dx: 源字符串, bx: 目的字符串	bx: 拷贝后的字符串	可应用于两种风格的字符串
concat	字符串连接	dx: 源字符串, bx: 目的字符串	bx: 连接后的字符串	可应用于两种风格的字符串, 也可混用, 但最终产生的字符串的风格取决于源字符串
index	字符串查找	dx: 源字符串, bx: 待查找字符串	ax: 有符号整数类型结果	返回值说明: -1代表在源字符串中未找到待查找字符串, 正数代表待查找字符串在源字符串中第一次出现的下标位置
rands	生成一个随机字符串	bx: 缓冲区地址, cx: 要生成的字符串长度	bx: 随机字符串	

3.3 算数

源代码文件名: arith.inc

> 注: 在8086体系结构中, 执行 div 或 idiv 指令, 若操作数(除数)为一个字节, 被除数被认为是一个存放在 ax 寄存器中的字, 指令执行完后, 商存放在 al 中, 为一个字节, 余数存放在 ah 中, 为一个字节. 然而对一个字做除法运算, 其商很可能大于一个字节, 特别是在除数较小的时候, 考虑 `1000 / 2`, 此时, al 对于这个"大商"来说位数是不够的, 这就发生了所谓的"除法溢出". 操作数(除数)为一个字也有类似情况.

函数名	功能	参数列表	返回值/结果	备注
bcd2dec	把4数位压缩BCD码转为8421码	ax: 待转换的4数位BCD码	ax: 转换后的数据	
mul32	进行32位无符号乘法运算, 乘数和被乘数均为32位	dx:ax 被乘数, cx:bx 乘数	dx:ax:cx:bx 积	
div32	进行32位无溢出的无符号除法运算	dx:ax 被除数, cx: 除数	dx:ax 商, bx: 余数	

3.4 文件

源代码文件名: file.inc

> 1. 该文件内定义的函数接收的字符串类型参数均要求是 C 风格字符串, 什么是 C 风格字符串?
> 2. 可使用相对路径

函数名	功能	参数列表	返回值/结果	备注
mkdir	创建文件夹	dx: 带路径的文件夹名	ax: 布尔类型结果	
rmdir	删除文件夹	dx: 带路径的文件夹名	ax: 布尔类型结果	
mkfile	创建一个空文件	dx: 带路径的文件名	ax: 布尔类型结果	
rmfile	删除一个文件	dx: 带路径的文件名	ax: 布尔类型结果	
exists	判断一个文件是否存在	dx: 带路径的文件名	ax: 布尔类型结果	
open	打开一个文件	dx: 带路径的文件名, ax: 模式	ax: 文件句柄	1. 模式: @ORD表示以只读模式打开, @OWR表示以只写模式打开, @ORDWR表示以读写方式打开. 常量定义位置; 2. 返回值说明: 若文件句柄为-1, 表示文件打开失败
close	关闭文件	ax: 文件句柄	ax: 布尔类型结果	
properties	获取文件属性	ax: 文件句柄	ax: 文件属性	文件属性: 位0 - 只读位, 位1 - 隐含位, 位2 - 系统位, 位3 - 卷标号, 位5 - 归档, 其他位保留不用, 设置为1
read	从一个已打开的文件读取若干字节, 存入指定的缓冲区中	ax: 文件句柄, cx: 要读取的字节数, bx: 缓冲区地址	ax: 布尔类型结果, cx: 实际读取的字节数, bx: 读取的数据	若文件剩余字节数小于 cx 指定的字节数, cx更新为实际读取字节数, ax更新为@FALSE(未验证)
write	向一个已打	ax: 文件句	ax: 布尔类	(失败情况未验证)

	开的文件写入若干字节数据	柄, cx: 要写入的字节数, dx: 要写入的数据首地址	型结果, cx: 实际写入的字节数	
seek	移动文件指针	ax: 文件句柄, cx: 模式, dx: 有符号的位移量	ax: 布尔类型结果	模式: @SEEK_SET表示偏移量是相对于文件首, @SEEK_CUR表示偏移量是相对于当前位置, @SEEK_END表示偏移量是相对于文件尾. 常量定义位置
rewind	把文件指针倒回到文件首	ax: 文件句柄	ax: 布尔类型结果	
tell	获取文件指针位置			未实现
mvfile	移动文件	dx: 带路径的源文件名, bx: 带路径的目标文件名	ax: 布尔类型结果	若目标文件已存在, 函数将静默覆盖此文件
mvdir	移动文件夹			未实现
cpfile	拷贝文件	dx: 带路径的源文件名, bx: 带路径的目标文件名	ax: 布尔类型结果	若目标文件已存在, 函数将静默覆盖此文件
cpdir	拷贝文件夹			未实现

3.5 系统

源代码文件名: sys.inc

函数名	功能	参数列表	返回值/结果	备注
exit	退出用户程序, 将控制权转交给	al: 可设置状态码		

	DOS 系统			
reside	以驻留方式退出用户程序	dx		我也不太懂
memory	获取内存容量		ax: 系统内存容量(以K为单位)	
date	获取系统日期		dx: 年, ah: 月, al: 日	
time	获取系统时间		dx: 时, ah: 分, al: 秒	
get_interrupt	获取一个中断向量	cx: 中断号	dx:ax 段地址:偏移地址	
set_interrupt	安装一个中断向量	cx: 中断号, dx:ax 段地址:偏移地址		
pspseg	获取当前程序 PSP 段		ax: PSP段地址	
cmdlen	获取命令行参数的长度		ax: 命令行参数的长度	
cmdline	获取命令行参数	bx: 缓冲区地址	bx: 命令行参数	
arg	依次获取每个以空	dx: 命令行参数地址,	ax: 处理过后	1. 由于返回值 ax 保存的是处理后的 dx, 应将 ax 传送给 dx 再进行一次调用以取得

	格隔开的参数	bx: 用于存放参数的缓冲区地址	的dx	下一条参数; 2. 调用本函数后, 如果[ax]='$' 或 strlen(dx)=0 则表示命令行参数已被处理完毕
drive	获取程序当前工作路径的驱动器号		ax:驱动器号	
pwd	获取程序当前工作路径	bx: 缓冲区地址	bx: 当前工作路径	
alloc	分配内存			未实现
free	释放内存			未实现
memcpy	内存拷贝	dx: 源地址, bx: 目的地址, cx: 要拷贝的字节数	bx: 拷贝后的数据	
dump	打印 CPU 当前时刻状态			“当前时刻”指的是调用 dump 函数的 call 指令正式执行之前的时刻, ip 应指向 call dump 的下一条语句

3.6 实用例程

源代码文件名: util.inc

函数名	功能	参数列表	返回值/结果	备注
srand	初始化随机数种子			
rand	产生一个随机数		ax:随机数	
sort	对一个有符号字数组进行排序, 升序	bx: 数组首地址, cx: 数组元素个数	bx: 排好序的数组	
*swap				内部使用
*partition				内部使用
*quick_sort				内部使用

qsort	对一个有符号字数组进行快速排序, 升序	bx: 数组首地址, cx: 数组元素个数	bx: 排好序的数组	
reverse	反转一个字数组	bx: 数组首地址, cx: 数组元素个数	bx: 反转后的数组	此函数可以跟 sort, qsort 搭配使用, 以改变数组次序
shuffle	打乱一个字数组的顺序	bx: 数组首地址, cx: 数组元素个数	bx: 打乱后的数组	

3.7 宏函数库

源代码文件名: macros.inc

1. “宏”是以宏替换和宏展开为基础的类函数结构, 源代码编译期间, 汇编器用展开后的宏函数体替换调用语句. 宏函数的传参方式比函数更灵活, 通常也比函数更快.
2. 使用说明: 复制宏函数体或插入 include macros.inc 到**用户代码之前**, 汇编器必须在展开一个宏之前看到它的定义
3. 宏函数的调用方式(以 WriteString 为例):

```
WriteString 'hello world'
```

宏函数名	功能	参数列表	返回值/结果	备注
WriteString	在光标位置打印一个字符串	1: 一个字符串		

4 常用汇编指令及其介绍

助记符	功能	用法	备注
mov			