

Turbo C 2.0

HOPE

运行库函数 源程序与参考大全



中国科学院希望高级电脑技术公司

Turbo C 2.0 运行库函数源程序 与参考大全

萧 黎 尤晓东 金利群 编

中国科学院希望高级电脑技术公司

版权所有

不许翻印

违者必究

■ 北京市新闻出版局

准印证号：3154—90154

■ 订购单位：北京 8721 信箱资料部

■ 电 话：2562329

■ 电 传：01—2561057

■ 电 挂：0755

■ 地 址：海淀影剧院北侧

■ 乘 车：320、332、302路海淀黄庄下车

■ 办公地点：公司大楼 101 房间

Turbo C 2.0 运行库源程序与参考大全

《Turbo C 2.0 运行库源程序与参考大全》整理编译了 Turbo C 2.0 运行库中所有函数以及详细的函数使用说明。它不仅全部提供了《Turbo C 参考手册》中提供的函数，而且还提供了实现这些函数的所有低层支持函数。这使得 Turbo C 2.0 运行库成为真正的开放式结构，用户可以从多方面高效地使用库函数。

不仅如此，有了库函数源程序，用户就可以对库函数进行单步断点等符号调试，可以领会 Turbo C 2.0 运行库以及整个软件的设计策略和技巧，为将来开发语言系统提供实例模型。Turbo C 之所以能风靡当今 PC 世界，不仅因为它有良好的用户界面，强大的符号调试功能，还因为它有设计精良、运行高速、结构优化的运行库函数。运行库函数源程序让用户能目睹其真正风采，领略其库设计策略、技巧、程序设计风格、函数依赖关系等等。

有了库函数源程序，就可以无限制地把 Turbo C 与 Turbo Pascal 揉合起来，综合 C 与 Pascal 的优点编程，使程序设计“更上一层楼”。(关于具体的接口方法可与相榷，联系地址：北京航空航天大学 1-54 信箱李振格)。

对于高级程序员来说，通过修改库函数源程序，可更高效地、多方位地使用 Turbo C，对于一般的编程者和初学 C 的人，它又是一个很好的实例教材，是进行程序设计的良师益友。

齐全的运行库函数与低层支持函数，结合《Turbo C 2.0 用户手册》、《Turbo C 2.0 参考手册》，将给用户带来无穷无尽的新发现与高效率。

本书在许多同志的帮助下，经过认真整理而成，由于篇幅浩大，源程序很多，虽然经过认真校对但错误难免，欢迎指正。

编者
一九九零年八月

目录

前言

Turbo C 库函数源程序

abort	1	CheckOpenType	28
abs	1	chmod	30
absread2abswrite	3	chmod	30
access	3	chsize	31
acos	4	circle	33
ACosAsin	4	clear	87
allocmem	6	cleardevice	34
_AppendCtIZ	7	clearerr	34
arc	7	clearviewport	34
asctime	8	_close	34
asin	9	close	34
assert	9	closegraph	35
atan	9	clock	35
atan2	9	clreol	36
atexit	11	clrscr	36
atof	11	comtome	36
atoi	12	_control	37
atol	12	CopyIt	38
bar	13	CopyUpr	39
bar3d	14	Coreleft	39
bdos	14	cos	40
bdosptr	15	cosh	40
bios	15	country	42
bioscom	16	cprintf	42
biosdisk	17	cputs	42
biosquip	19	_creat	43
bioskey	20	creat	43
biosmemory	21	CreatFile	45
biostime	22	creatnew	45
_brk	23	_crtinit	46
_brk	23	cscanf	46
brk	23	ctime	47
bsearch	24	ctrlbrk	48
_c0crtinit	24	delay	48
cnbs	26	delline	53
calloc	26	detecthgraph	53
ceil	26	difftime	53
cgets	27	disable	53
chdir	27	div	54
CheckFile	28	Dispalcement	54

_DOScmd	55	fileellipse	87
dosCreat	55	fillpoly	87
DOSenv	56	findfirst	87
dosexterr	57	findnext	89
dosReadOne	57	floodfill	89
dosSeekFinalChar	58	floor	89
_DOSTimeToU	58	flushall	90
dostounix	60	FlushOutStreams	90
doswriteNone	61	fmod	91
DotFound	61	fnmerge	92
drawpoly	61	fnsplit	93
dup	62	fopen	94
ecvt	63	_fpreset	95
egainstalled	63	FP_OFF	95
ellipse	64	FP_SEG	96
_emit	64	fprintf	96
enable	64	_fputc	96
dof	64	fputc	96
Exchange	65	fputchar	97
exec...	66	_fputn	98
_exit	73	fputs	98
exit	73	fread	98
exp	73	free	99
fabs	75	freemem	101
farcalloc	76	freopen	101
farcoreleft	76	frexp	101
farfree	76	fscanf	102
farmalloc	78	fseek	102
farrealloc	80	fsetpos	103
fclose	81	fstat	103
fcloseall	81	ftell	105
fcvt	82	ftime	105
fdopen	82	fwrite	106
feof	82	gcvt	106
ferror	82	geninterrupt	106
fill	83	Get	107
flush	83	getarccoords	107
fgetc	84	getaspectratio	107
fgetchar	85	getbkcolor	107
_fgetn	85	getc	108
fgetpos	86	getchrk	110
fgets	86	getch	111
filelength	86	getchar	111
fileno	87	getche	111

getcolor	111	grapherrormsg	135
getcurdir	112	graphexit	135
getcwd	112	graphfreemem	135
getdate	113	graphfreemem	135
getdefaultpalette	114	graphresult	136
getdfree	114	harderr	137
getdisk	115	hardresume	138
getdrivename	115	hardretn	138
getdta	115	heaplen	138
getenv	116	hentry	139
getfat	117	Hex4	139
getfatd	118	highvideo	140
getfillpattern	119	hyperb	140
getfillsettings	119	hypotf	140
getfp	120	imagesize	141
getftime	120	initgraph	141
getgraphmode	121	inp	144
getimage	122	inport	144
GetIndex()	122	insline	144
getlinesettings	122	installuserdrever	145
getmaxcolor	123	installuserfont	145
getmaxx	124	Int0Catcher()	146
getmaxy	124	Int4Catcher()	146
getmodename	124	Int5Catcher()	146
getmoderange	124	Int6Catcher()	147
getpalette	124	int86	148
getpass	126	int86x	148
getpixel	126	intdos	150
getpsp	127	intdosx	150
gets	127	intr	151
getswitchar	128	ioctl	153
gettext	128	IOerror	154
gettextinfo	128	is...	156
gettextsettings	129	isatty	157
gettime	131	isDST	157
getvect	131	itoab	157
getverify	132	_KbdFlu	159
getviewsetting	132	kbhit	159
getw	133	keep	160
getx	133	labs	160
gety	133	ldexp	160
gmtime	133	ldiv	161
gotoxy	134	ldtrunc	162
graphdefaults	134	lfind	164

line	164	peekb	192
linere1	164	perror	193
lineto	164	pieslice	193
localtime	165	poke	194
lick	165	pokeb	194
log	166	poly	194
log10	166	pow	195
longjump	167	pow10	198
lowvdo	168	...printf	200
_lrotl	168	purc	205
_lrotr	169	putch	205
_lsearch	169	purchar	206
lsearch	170	putenv	206
lseek	170	putimage	208
ltoa	171	putpixel	208
malloc	171	puts	208
lsetmem	173	puttext	209
_matherr	174	putw	209
matherr	175	qsort	209
max	177	qSortHelp	210
mem...	177	raise	211
mkdir	180	rand	212
MK_FP	181	randbrd	212
_mkname	181	randbwr	213
mktemp	181	random	213
modf	182	randomize	214
movedata	183	_read	214
moverel	183	read	214
movetext	183	_realcvt	216
moveto	184	realloc	221
movemem	185	rectangle	221
normalize	186	registerbgidriver	221
normvideo	186	remove	222
nosound	186	rename	222
_open	187	restorecrtmode	223
open	187	rewind	223
_openfp	190	rmdir	223
outp	190	_rotl	223
outport	190	_rotr	224
outportb	191	_shrkb	224
outtext	191	_shrk	225
outtextxy	191	shrk	225
parsfnm	191	...scanf	225
peek	192	_scanner	231

scanpop	244	setwritemode	268
scanrsit	244	signal	268
scantod	245	sin	270
scantol	251	sinh	270
_screenio	255	sleep	271
screenpos	255	sopen	272
_scroll	255	sound	272
_searchpath	256	spawn...	272
searchpath	257	sprintf	276
sector	257	sqrt	276
segread	258	srand	277
setactivepage	258	ssanf	277
setallpalette	259	stat	277
setaspectratio	259	_status87	280
setbkcolor	259	stime	280
setblock	259	_stklen	280
setbuf	260	stl_Digit	280
setcbkr	260	stpcpy	281
setcolor	261	Str...	282
setdate	261	_strerror	296
setdisk	261	strerror	296
setdta	261	strpatn	297
setfillpattern	262	strtoul	297
setfillstype	262	swab	298
setftime	262	system	298
setgraphbufsize	262	tan	299
setgraphmode	263	tanh	300
setjmp	263	tell	301
setlinestyle	264	textattr	301
setmem	264	textbackground	302
setmode	264	textcolor	303
setpalette	265	textheight	303
setrgbcolor	265	textmode	304
setgbpalette	265	textwidth	305
setswitchar	265	time	305
settextjustify	266	tmpfile	305
settextstyle	266	tmpnem	305
settime	266	toascii	306
setusercharsize	266	_tolower	306
setvbuf	267	tolower	306
setvect	267	_toupper	307
setverify	268	toupper	307
setviewport	268	trig	307
setvisualpage	268	_TrimCtlZ	308

TrimTrailing	308
tzset	309
ultoa	310
umask	310
UnGet	311
ungetc	311
ungetch	311
unixtodos	311
unlink	312
unlock	313
va...	313
va_arg	314
va_end	314
_validatexy	314
wva_start	314
vfprintf	315
vfprintf	315
_VideoInt	315
_vprinter	316
vprintf	330
vptr	331
_vram	331
vscanf	332
vsprintf	332
vsscanf	333
wherex	333
wwherey	333
window	334
_write	334
write	335
_xclose	336
_xcvt	336
_xfclose	340
_xflush	341
zapline	341

Turbo C 库函数源程序

本章按字母顺序列出 Turbo C 2.0 软件包中提供的各个库函数,介绍它们的功能、用法、相关函数用法、原型所在头文件、说明信息、返回值、可移植性、参阅部分和示例,并给出实现该函数的源程序,具体格式如下:

函数名 功能

用 法 本函数的用法

相关函数

用 法 本函数的相关函数的用法

原 型 在 本函数原型所在的头文件

说 明 本函数的使用信息说明

返 回 值 本函数的返回值

可移植性 本函数的适用系统

参 见 和本函数有关的其它函数名

示 例 本函数使用的程序例

源 程 序 实现本函数的源程序

abort 异常终止一进程

用 法 void abort(void);

原 型 在 stdlib.h, process.h

说 明 本函数写终止信息到 stderr 中,并通过_exit调用终止程序执行。

返 回 值 本函数不返回值

可移植性 适用于 UNIX 系统

参 见 assert, _exit, exec..., exit, spawn...

abs 绝对值

—abs.cas

用 法 int abs(int i);

相关函数 double cabs(struct complex znum);

用 法 double fabs(double x);

long labs(long n);

原 型 在 stdlib.h, math.h

说 明 本函数返回整数参数 i 的绝对值,如果包含了 stdlib.h 头文件并调用了 abs, abs 将作为可扩展为插入代码的宏看待。如果不包含 stdlib.h(或包含了但使用了#undef abs), abs 将作为函数看待而不是宏。

cabs 是一计算复数 znum 绝对值的宏。znum 是一具有类型 complex 的结构。此结构在 math.h 定义如下:

```
struct complex{
    double x,y;
};
```

调用 cabs 等价于调用具有 znum 的实部和虚部的 sqrt,如下式所示:

```
sqrt(znum.x*znum.x+znum.y*znum.y)
```

如果没有包含 math.h 头文件,或包含了而又使用了(undef cabs), cabs 将作为一函数看待而

不是宏。

fabs 计算 double 类型 x 的绝对值。

labs 计算 long 整型 n 的绝对值。

返回值 **abs** 返回 0 到 32767 间的一个整数，有一个例外：当参数值为 -32768 时，返回 -32768。

cabs 返回 double 型的 znum 的绝对值，如果发生溢出，将返回 HUGE_VAL 并置 error 为

ERANGE 结果超出范围

可以通过 **matherr** 函数修改对 **cabs** 的错误处理。

fabs 返回 x 的绝对值，**labs** 返回 n 的绝对值，它们都不返回出错信息。

可移植性 适用于 UNIX 系统

参 见 **matherr** 源程序

```
#pragma inline
#include <stdlib.h>
#undef abs /* abs is defined as a macro in stdlib.h */
int abs(int i)
{
asm    mov    ax, i
asm    or     ax, ax
asm    jge    exit
asm    neg    ax
exit:
    return _AX;
}
```

absread 读数据

---absread.cas

用 法 int absread(int drive,int nsects,int sectno,
void *buffer);

相关函数

用 法 int abswrite(int drive,int nsects,int sectno,
void *buffer);

原 型 在 dos.h

说 明 这些函数读和写磁盘的指定扇区内容。它们忽略磁盘的逻辑结构，并不关心文件、FAT 和目录。

absread 通过 DOS 中断 0x25 读指定的磁盘扇区内容。

abswrite 通过 DOS 中断 0x26 写内容到指定的磁盘扇区。

drive=要读的磁盘驱动器号(0=A,1=B 等)

nsects=要读的扇区数

sectno=要读的起始扇区号

buffer=数据读或写的存储区地址

读的扇区数受段中 **buffer** 以上的存储空间量的限制。因此，在一次调用 **absread** 或 **abswrite** 过程中，最大的可读存储空间为 64k 字节。

返回值 如果调用成功，两函数都返回 0。

如果调用出错，都返回 -1，并置 **error** 为经系统调用后返回 **AX** 的寄存器的值。关于 **error** 的说明，请参阅 MS-DOS 使用手册。

可移植性 只适用于 MS-DOS

源 程 序

```
#pragma inline
#include <asmrules.h>
#include <dos.h>
#include <errno.h>
```

```

#pragma warn -use
int absread(int drive, int nsects, int lsect, void *buffer)
{
    register bogus1, bogus2;          /* force save of SI and DI */
asm    mov     al,drive
asm    mov     cx,nsects
asm    mov     dx,lsect
asm    pushDS
asm    LDS     bx,buffer
asm    int     25h
asm    pop     bx                      /* clear old flags */
asm    popDS
asm    jc      abserr
asm    return(0);
abserr:
#ifdef __HUGE__
asm    mov     bx,seg errno
asm    mov     es,bx
asm    mov     esi,errno,ax
#else
asm    mov     errno,ax
#endif
    return(-1);
}
#pragma warn .use

```

abswrite 写数据

—absread.cas

用 法 int abswrite(int drive,int nsects,int sectno,
void *buffer);

原 型 在 dos.h

说 明 见 absread

源 程 序

```

#pragma warn -use
int abswrite(int drive, int nsects, int lsect, void *buffer)
{
    register bogus1, bogus2;          /* force save of SI and DI */
asm    mov     al,drive
asm    mov     cx,nsects
asm    mov     dx,lsect
asm    pushDS
asm    LDS     bx,buffer
asm    int     26h
asm    pop     bx                      /* clear old flags */
asm    popDS
asm    jc      abserr
asm    return(0);
abserr:
#ifdef __HUGE__
asm    mov     bx,seg errno
asm    mov     es,bx
asm    mov     esi,errno,ax
#else
asm    mov     errno,ax
#endif
    return(-1);
}
#pragma warn .use

```

access 确定文件的存取权限

- access.c

用 法 int access (char *filename,int amode);

原 型 在 io.h

说明 本函数检查给定名的函数是否存在，它的可读、可写和可执行性。filename 是指向文件名字符串的指针。

包含在 amode 中的模式结构如下：

06 检查读/写允许
04 检查读允许
02 检查写允许
01 执行(被忽略)
00 检查文件的存在性

注意：在 MS-DOS 下，所有存在的文件具有读访问(amode=04)性，所以 00 和 04 有同样的结果。同理，06 和 02 是等价的，因为在 MS-DOS 下，写访问性隐含了读访问性。

如果 filename 指向一目录，access 就只确定该目录是否存在。

返回值 如果要求的存取是允许的，返回 0；否则返回-1，并置 error 为以下值之一：

ENOENT 路径或文件名没找到
EACCES 无此权限

可移植性 适用于 UNIX 系统

参见 chmod

源程序

```
#include <io.h>
#include <dos.h>
#include <errno.h>
int access(const char *filename, int amode)
{
    register int attrib;
    attrib = _chmod(filename, 0);
    if (attrib == -1)
        return(attrib);
    if ((amode & 0x00e2) != 0 || (attrib & FA_RDONLY) == 0)
        return(0);
    errno = EACCES;
    return(-1);
}
```

acos 三角函数 - acosasin.cas

用法 double acos(double x);

原型在 math.h

说明 trig

源程序

```
double acos (double x)
{
    return AcosAsin ("acos", 0xFF, &x);
}
```

AcosAsin 计算反正弦或反余弦。 - acosasin.cas

用法 double AcosAsin (char *whoS, bits16 flags, double *xP),

说明 计算由 xP 指向的数的反正弦或反余弦值。

返回值 返回 xP 的反正弦或反余弦值。

源程序

```
#pragma inline
#include <asmrules.h>
#include <rules.h>
#include <math.h>
```

```

#include <_math.h> #include <errno.h>
static unsigned short NANINVTTRIG [4] = {0,0,0x0440, 0x7FF0};
#pragma warn -rvl
#pragma warn -use
static double AcosAsin (char *whoS, bits16 flags, double *xP)
{
    bits16 status;
    int temp;
register
    SI, DI; /* prevent the compiler making its own usage */
asm mov dx, flags
asm mov si, W0 (xP)
asm mov cx, SS [si+6]
asm and BY0 (SS [si+7]), 07Fh /* absolute value */
asm fld DOUBLE (SS [si])
asm shl cx, 1
asm rcr dh, 1 /* DH = sign bit */
asm jcxz acs_zero /* biased exponent of 10 */
asm cmp cx, 7FE0h
asm jnb acs_extreme
/*
    Use the trig identity asin (x) = atan (x / sqrt (1 - x^2)).
*/
asm FLDI
asm FLD ST(1) /* duplicate X */
asm FMUL ST(0), ST(0) /* square it */
asm FSUBP ST(1), ST /* 1 - X^2 */
asm FSQRT
/*
    We now have tangent = ST(1)/ST. In order to use the FPATAN instruction
    we must order them so that ST(1) < ST. The special case of equating
    (angle pi/4) must be treated separately.
*/
asm FCOM
asm FSTSW W0 (status)
asm mov ah, 41h /* C3, C0 are the important bits */
asm FWAIT
/*
    At this stage note that acos (x) = atan (sqrt (1-x^2) / x), which is
    the inverse of the tangent ratio used for asin (x). That is why
    DL, the inversion flag, started as 0FF for acos, and 0 for asin.
*/
asm and ah, BY1 (status)
asm jz acs_atan
asm add ah, ah
asm js acs_pi_4
asm FXCH
asm not dl /* remember the exchanged order */ acs_atan:
asm FPATAN /* arctan (ST(1) / ST) */
asm or dl, dl /* should ratio be inverted ? */
asm jz acs_sign
asm mov W0 (temp), -1
asm FILD W0 (temp)
asm FLDPI
asm FSCALE /* pi/2 */
asm FSTP ST(1)
asm FSUBRP st(1), st /* ST = pi/2 - ST */
acs_sign:
asm or dh, dh
asm jns acs_end
asm FCHS
asm cmp BY0 (flags), 0FFh /* are we doing acos (x) ? */
asm jne acs_end
asm FLDPI
asm FADDP ST(1), ST
acs_end:
    return;
/*
    Special cases.
*/

```

```

acs_extreme:
asm    ja      acs_domain
asm    mov     ax, SS_ [si+4]          /* check for an exact value +- 1.0 */
asm    or      ax, SS_ [si+2]
asm    or      ax, SS_ [si]
asm    jnz     acs_domain
asm    jmp     short acs_one
acs_zero:
asm    mov     dh, 0                  /* make zero unsigned. */
asm    FSTP    ST(0)                  /* pop stack */
asm    cmp     BY0 (flags), 0FFh      /* are we doing acos () ? */
asm    je      acs_resultP2
acs_resultZ:
asm    FLDZ
asm    jmp     short acs_sign
acs_one:
asm    FSTP    ST(0)                  /* pop stack */
asm    cmp     BY0 (flags), 0FFh      /* are we doing acos () ? */
asm    je      acs_resultZ
acs_resultP2:
asm    mov     W0 (temp), -1 asm      FILD    W0 (temp)
asm    FLDPI
asm    FSCALE
asm    FSTP    ST(1)                  /* pi/2 */
asm    jmp     short acs_sign
acs_pi_4:
asm    FCOMPP                          /* pop two items off the stack */
asm    mov     W0 (temp), -2
asm    FILD    W0 (temp)
asm    FLDPI
asm    FSCALE                          /* pi/4 */
asm    FSTP    ST(1)
asm    jmp     short acs_sign
/*
If the argument was outside [-1,+1] then it is a domain error.
*/
acs_domain:
asm    or      BY0 (SS_ [si+7]), dh    /* put the sign back */
asm    FSTP    ST (0)                  /* pop x from stack */
#pragma warn -ret
return _matherr (DOMAIN, whoS, xP, NULL, *((double *) NANINVRIG));
#pragma warn .ret
|
#pragma warn .rvl
#pragma warn .use

```

allocmem 分配 DOS 存储段

— allocmem.cas

用 法 int allocmem(unsigned size,unsigned *seg);

相关函数 int freemem(unsigned seg);

用 法 int setblock(int seg,int newsiz);

原 型 在 dos.h

说 明 本函数使用 MS-DOS 系统调用 0x48 来分配自由存储块，并返回分配块的段地址。

size 是段中要求的存储空间大小。seg 是一指针，它指向将被赋值的新分配块的段地址的地址字。如果没有足够的空间可供分配，将不对 seg 所指的地址字赋值。

所有被分配的块是用指针连在一起的段。

freemem 释放以前用 allocmem 调用所分配的存储块。seg 是该块的段地址。

setblock 修改存储段的大小。seg 是以前调用 allocmem 后返回的段地址。newsiz 是新的、要求的段中存储空间大小。

返 回 值 如果调用成功，allocmem 返回-1。若出错，将返回-1数(为最大可用块大小)。

如果调用成功，freemem 返回 0，否则返回-1，并置 error 为：

ENOMEM 无足够存储空间

如果调用成功, setblock 返回值-1, 否则返回最大可用块的大小。

allocmem 或 setblock 返回的错误码将置-doserrno 和全局变量 error 为:

ENOMEM 无足够的存储器

可移植性 只适用于 MS-DOS

参 见 malloc

源 程 序

```
#pragma inline
#include <asmrules.h>
#include <dos.h>
#include <io.h>
int allocmem(unsigned size, unsigned *sepp)
{
    asm    mov     ah, 48h
    asm    mov     bx, size
    asm    int     21h
    asm    jc      allocmemFailed
    asm    LES     di, sepp
    asm    mov     ES, [di], ax
    return(-1);
allocmemFailed:
    asm    push    bx
    asm    _IOerror(_AX);
    asm    pop     ax
    return (_AX);
}
```

函 数 名 __AppendCtlZ - 给文件添加 Ctrl-Z。

zappctrlz.cas

用 法 void pascal __AppendCtlZ (int fildes)

原型文件 _io.h

说 明 给文件添加一个 Ctrl-Z。

源 程 序

```
void pascal __AppendCtlZ (int fildes)
{
    char c = _ctlZ;
    dosSeekFinalChar (fildes);
    if (dosReadOne (fildes) != _ctlZ)
    {
        pushDS
        asm    mov     bx, fildes
        asm    mov     cx, 1
#ifdef LDATA
        asm    push    SS
        asm    pop     DS
#endif
        asm    lea     dx, c
        asm    mov     ah, 40h
        asm    int     21h /* dosWrite (fildes, DS:DX, CX) */
        popDS
    }
}
```

arc 画一条圆弧

用 法 #include <graphics.h>

void far arc(int x,int y,int stangle,int endangle,

int radius);

相关函数 void far circle(int x,int y,int radius);