Microsoft® C 5.0

优化编译

附加库的安装及使用说明

（第八册）

中国科学院软件所
科海培训中心

Microsoft C 5.0 优化编译

附加库的安装使用说明

（第八册）

译 校 者 序 言

翻 译：宗丽萃 吴 倩 邦继明 鲁 倩 王 越 徐砥祥 陈 林 彭怀宇
王淑平 何 骏 郭瑞阳 萧燕林

Microsoft C 5.0优化编译语言系统是美国87年国家标准，具有效率高、功能强的特点。为了迎合广大热心者的要求，配合国内用戶高水平开发工作，由中国科学院386微机研究，开发组直接组织编译了此书共九册。虽然译校者大多是软件专业的研究生及工作多年的高资历工作人员，但由于印排仓促、审校时间紧迫，难免会有这样或那样的错误。望读者能给以谅解与指正。

目　录

第一部分 快速参考

常量

名字	文件	类型	功能
AT_INV	sc_head.h	常量	反转颜色属性
AT_REG	sc_head.h	常量	正规屏幕属性
BLACK	sc_head.h	常量	屏幕正文颜色
BLIB_CSTACK	blackstr.h	常量	C 栈
BLUE	sc_head.h	常量	屏幕正文颜色
BROWN	sc_head.h	常量	屏幕正文颜色
CYAN	sc_head.h	常量	屏幕正文颜色
DDY3F	ut_defs.h	常量	三字符目名标志
DMO3F	ut_defs.h	常量	三字符目名标志
DTNDF	ut_defs.h	常量	目名标志（不是#）
DTNMF	ut_defs.h	常量	目名标志（不是#）
DYR2F	ut_defs.h	常量	2 数字年标志
EPSONRX80	pr_head.h	常量	Epson
ERROR	blackstr.h	常量	-1
FALSE	blackstr.h	常量	0
GRAY	sc_head.h	常量	屏幕正文颜色
GREEN	sc_head.h	常量	屏幕正文颜色
EMSDOS	sy_head.h	常量	MSDOS 中断（21H）
ISCREEN	sy_head.h	常量	屏幕中断（10H）
IVIDEO	sy_head.h	常量	IBM BIOS 显示中断
（链值）	kb_head.h	常量	（见金属）80(sowrce)
LATTICE	blackstr.h	常量	用于 Lattice 编译器
LDATA	blackstr.h	常量	大数据模型
LPROC	blackstr.h	常量	大程序模型
MAGENTA	sc_beae.h	常量	屏幕正文颜色
MENU	tx_head.h	结构	菜单的结构
MESC	tx_head.h	结构	信息结构

MICROSOFT	blackstr.h	常量	用于 Microsoft 编译器
MO_BW80	sc_head.h	常量	黑白80列方式
MO_CO80	sc_head.h	常量	彩色80列方式
NEC8023	pr_head.h	常量	NEC8023打印机名
NOERROR	blackstr.h	常量	0
RDOT	gr_head.h	常量	IBM BIOS 读点命令
RED	sc_head.h	常量	屏幕正文颜色
TRUE	blackstr.h	常量	1
TSECF	ut_defs.h	常量	时间和百秒标志
T24HF	ut_defs.h	常量	24小时的时间标志
WHITE	sc_head.h	常量	屏幕正文颜色
WRDOT	gr_head.h	常量	IBM BIOS 写点命令

全局量和常量

名字	文件	类型	功能
breakf_	kb_defs.h	char	识别键盘中断(break)标志
chinf	kb_defs.h	char	字符输入标志
co_attrptrs_	co_defs.h	**char[ ]	控制台设备的属性值数目
co_colrpus	co_defs.h	**char[ ]	控制台设备的颜色代码数组
co_culpus	co_defs.h	**char[ ]	控制台设备的控制值数组
colen_	sc_defs.h	int	屏幕窗口结束列
colst_	sc_defs.h	int	屏幕窗口开始列
co_modepus_	co_defs.h	**char[ ]	控制台设备模式值数组
(conflg 值)	sy_defs.h	int	(见表sy-head)
co_type_	co_defs.h	shou	控制台类型数组
dateoptf_	ut_defs.h	char	控制日期格式选择标志
dry_	ut_defs.h	int	当前日
dy_names_	ut_defs.h	*char[ ]	日名
dy_3names_	ut_defs.h	*char[ ]	3字符日名
echof_	kb_defs.h	char	显示字符标志
gr_bcol_	gr_head.h	short	图形背景颜色值
gr_bcolr_	gr_defs.h	short	图形背景颜色值
gr_bpal_	gr_defs.h_	short	图形背景配色器
gr_buf_	gr_head.h	char*	图形缓冲区地址
gr_bytes_	gr_head.h	int	图形缓冲区大小
gr_cgap0	gr_head.h	char[ ]	CGA 模式－0图形配色器
gr_cgap1	gr_head.h	char[ ]	CGA 模式－1图形配色器
gr_colen_	gr_defs.h	int	图形窗口列数

续

名字	文件	类型	功能
gr_cols_	gr_defs.h	int	图形最大列数
gr_colst_	gr_defs.h	int	图形窗口列始
gr_egap_	gr_head.h	char[ ]	EGA 模式图形省缺配色器
gr_fcol_	gr_head.h	char	图形前景颜色索引
gr_fcolr_	gr_defs.h	shou	图形前景颜色值
gr_fpal_	gr_defs.h	int	图形前景配色器
gr_maxx_	gr_head.h	int	图形窗口在 X轴的最大值
gr_maxy_	gr_head.h	int	图形窗口在 Y 轴的最大值
gr_minx_	gr_head.h	int	图形窗口在X 轴的最小值
gr_miny_	gr_head.h	int	图形窗口在 Y 轴的最小值
gr_off_	gr_head.h	int	图形缓冲区地址位移
gr_mode_	gr_defs.h	int	图形模式值
gr_roweh_	gr_defs.h	int	图形窗口行尾
gr_towst_	gr_defs.h	int	图形最大行数
gr_towst_	gr_defs.h	int	图形窗口行始
grx_	gr_defs.h	int	图形 X 坐标
gr_xotf_	gr_defs.h	short	图形异式标志
gr_y_	gr_defs.h	int	图形 Y 坐标
hour_	ut_defs.h	int	当前小时值
hsec_	ut_defs.h	int	当前1/100分秒
ibmattr_	co_head.h	char*[ ]	ibm ansi 控制设备属性值
ibmcolrl_	co_head.h	char*[ ]	ibm ansi 控制设备颜色
ibmctrl_	co_head.h	char*[ ]	ibm ansi 控制设备控制值
ibmmode_	co_head.h	char*[ ]	ibm ansi 控制设备模式值
ibsmodef_	kb.defs.h	char	插入模式 联接断 开 (on/off) 标志
keybrkibl_	kb.defs.h	char*	键盘中断表(break)指针
keybrktbll_	kb.defs.h	char*	省缺键盘中断表
keyfibl_	kb.defs.h	char*	键盘功能表指针
keyftbll_	kb.defs.h	char*	省缺键盘功能表 功能键表
keystbl_	kb.defs.h	char*	键盘扫描表指针
keystbll_	kb.defs.h	char*	省缺键盘扫描表
keyxtbl_	kb.defs.h	char*	键盘变换表指针
keyxtbll_	kb.defs.h	char*	省缺键盘变换表

续

名字	文件	类型	功能
mcol_	ms_defs.h	int	鼠标列位置
minute_	ut_deis.h	int	当前分
mo_days_	ut_defs.h	int[13]	每日无数
mo_names_	ut_defs.h	*char[ ]	日名
mo_3names_	ut_defs.h	*char[ ]	3字符日名
mouth_	ut_defs.h	int	当前日
mrow_	ms_defs.h	int	鼠标行位置
ms_mask	ms_defs.h	int*	鼠标掩码
uec23attr_	pr_defs.h	char[ ]	具有NEC8023打印机属性的打印机
uec23tab_	pr_defs.h	char[ ]	NEC8023打印机tab值
ulexpf_	sc_defs.h	int	换行扩展标志 (cl-lf)
打印机属性	pr_head.h	char*	(见pr-olefs.h)
rowst_	sc_defs.h	int	屏幕窗口超始行
rowen_	sc_defs.h	int	屏幕窗口超尾行
rx80attr_	pr_defs.h	char[ ]	EPSON打印机属性值
rx80tab_	pr_defs.h	char[ ]	EPSON打印机tab值
sc_adp_	sc_defs.h	char	屏幕建配器类型
sc_attr_	sc_defs.h	char	当前屏幕属性
sc_col_	sc_defs.h	int	当前光标列位置
sc_colr_	sc_head.h	char	当前屏幕颜色
sc_cols_	sc_head.h	int	屏幕最大列数
sc_ega__	sc_head.h	char	EGA卡的设置
sc_mem	sc_head.h	char	K字节的屏幕内贮值
sc_page_	sc_head.h	int	当前屏幕页
sc_mode_	sc_defs.h	int	当前屏幕模式
sc_row_	sc_defs.h	int	当前光标行位置
sc_rows	sc_head.h	int	屏幕的最大行数
sec	ut_defs.h	int	当前秒
timeoptf_	ut_defs.h	char	控制时间格式选择标志
tpratt_	pr_defs.h	char*[]	打印机属性类型数组
tprtabs_	pr_defsh	char*[]	打印机tab类型数组
wrapf_	sc_defs.h	int	屏幕行绕转求非绕转标志 (wrap on/off)
year_	ut_defs.h	int	当前年

库函数

ANSI控制函数（A = co - hancll. c）

文件	函数	功能
A	co_attr(attr)	设置控制台属性
A	co_clr()	请控制台屏幕
A	co_color(fcolr, bcolr)	设置屏幕的前景和背景颜色
A	co_cuget(col, row)	设置光标位置
A	co_curset(col, row)	设置光标的列和行的位置
A	co_eeol()	删除从光标当前位置列行尾的内容
A	co_getc()	从控制台读取字符
A	co_gets(str)	从控制台读取伪字符串
A	co_init(console)	对控制台初始化
A	co_mode(mode, fcolor, bcolor)	设置控制台模式
A	co_putc(c)	将字符送给控制台
A	co_puts(str)	将字符串送给控制台
A	co_sattr(attr)	返回控制台属性字符串
A	co_set(mod, val)	获得控制台 VALUE 控制字符串
A	co_type	设置控制台类型

文件函数（A = fl-paths. c, B = fl-tunc. c）

A	fl_files(fnames, fpath, ext_f)	将文件量加入数组中
A	fl-find(path)	寻找文件名
B	fl_getl(buff, fp)	从文件中读取一行
B	fl_getln(buf, fp)	获取一行并以 NUL 终止
B	fl_getr(buff, fp, len)	读取固定K记录
B	fl_getvr(buff, fp)	读取变定K记录
A	fl_mkpath(path,dr,dir,name,ext)	从字符串中建立一个路径名
A	fl_pathex(ext, path)	从路径中读取扩展
A	fl_pathnm(name,path)	从路径中读取名字
B	fl_putl(buff, fp)	将一行放入正文文件中
B	fl_putr(buff, fp, len)	将K为Len的记录于文件中
B	fl_putvr(buff, fp)	将变K记录放入文件中

图形函数（A = gr-handl. c, B = gr-drive. asm, C = gr-shape. c, D = gr-windo. c, E = gr-fill. c）

A	gr_bcolor(bcolr)	

C	gr_box(x1,y1,x2,y2)	用两个对角点画一长方形（绝对的）
A	gr_clip(x,y)	剔除一点
D	gr_clr()	清除图形窗口
A	gr_color(fcolr, bcolr)	设置前景和背景颜色
C	gr_draw(shape)	绘制一个图形
E	gr_fill(seedx, seedy)	对有界区填充颜色
E	gr_fnb(x,y)	寻找负边界
E	gr_fpb(x,y)	寻找正边界
E	gr_fseed(x1,x2,y1)	寻找新种点
C	gr_getp(pic)	获取图形到缓冲区中
A	gr_getpt(x,y)	获得点X，Y的颜色
A	gr_getwin(x1,y1,x2,y2)	读取图形窗口的象元颜色值到缓冲区中
A	gr_init(mode,fcolr,bcolr,pallet)	设置图形模式
A	gr_line(x1,y1,x2,y2)	从X_1，Y_1，到X_2,Y_2画一条直线
A	gr_lineto(x,y)	到(X,Y)点的直线
A	gr_pallet(pallet)	设置图形适配器
A	gr_pen(xorf,color)	设置图形笔
A	gr_poly(shape)	用线段画一图形
A	gr_pos(x,y)	设置图形笔点为X,Y
C	gr_putp(pic)	将图形显示在屏幕上
A	gl_putpt(x,y,color)	在点X,Y设置颜色
A	gr_putwin(fuf_x1,y1,x2,y2)	将缓冲区的内容放到图形窗口中
C	gr_scale(shape,factor)	按比例因子增大或缩小一图形
C	gr_scan(shape)	扫描屏幕上的图形
A	gr_setpt(x,y)	设置点X,Y
D	gr_wfill(color)	使用颜色填充窗口
D	gr_wfull()	将图形窗口设置为全屏幕
D	gr_wgetp(buff)	将图形放入缓冲区中
D	gr_windo(minx,miny,maxx,maxy)	设置图形窗口
D	gr_wpop()	恢复以前的图形窗口
D	gr_wpush()	存贮当前图形窗口
D	gr_wputp(buff)	将图形放入窗口中
A	gr_xorp(x,y)	对点(X,Y)进行异或操作

中断函数（A=in-handl.c）

文件	函　数	功　能
A	in_temint(int no)	清除以前设置的中断
A	in_setint(vector.function.stack.type)	设置一个中断

键盘函数(A = kb-field.c, C、B = kb-drive.asm, C = kb-handl.c, D = co-drive.asm)

C	kb_clr()	清键盘
C	kb_getc()	原始键盘输入
C	kb_getch()	使用 breakyneclo 获取字符
C	kb_getchar()	从键盘中获取字符
A	kb_getf(str,n)	获取域宽为 n 的字符串
C	kb_gets(str)	从键盘读取一个字符串
B	kb_hit()	查看是否在键盘上已按了键
C	kb_init(ftbl,xtbl,btbl)	初始化键盘
C	kb_inkey()	如果有一键输入时返回一键
A	kb_input(ntsg,str)	用信息提示字符输入
A	kb_inyom(prompt)	输入 yes 或 no
A	kb_isbrki()	看是否 c 是 break 键
A	kb_istunt(c)	看是否 c 在功能表中
A	kb_isky(c)	看是否 c 内一有效键
C	kb_pause()	暂停直到按键盘为止
D	kb_scanf("format string",vari. .va mrn)	格式化的键盘输入例程
C	kb_tpause(time)	暂停到键盘被击键或 100 秒过
A	kb_ungetc(c)	返回以前读取的字符

存贮函数 (A = me-alph.c)

A	me_cmpu(bufl,buf2,n)	用大写比较缓冲区的内容
A	me_lowet(buf,n)	把缓冲区中的字符转换为小写
A	me_squ(buf,c,n)	从缓冲区 buf 中压出字符c
A	me_upper(buf,n)	把缓冲区中的字符转换为大写

鼠标函数 (A = ms-handl.c)

A	ms_cget()	获得鼠标器光标的位置
A	ms_cset(col,row)	把鼠标器光标设置到行、列位置
A	ms_csof()	取消鼠标器光标
A	ms_cson()	恢复鼠标器光标
A	ms_field(icol l,rowl,co12,row2)	设置鼠标器域窗口
A	ms_init()	初始化鼠标器
A	ms_intr(mask,routine)	建立鼠标器中断服务例程
A	ms_motion()	使鼠标器运动
A	ms_press()	按下按钮

A	ms_reis()	释放按钮
A	ms_shape(hhot,vhot,masks)	设置光标形状
A	ms_stat()	获得鼠标器状态

打印机函数 (A = pr - handl.c)

A	Pr_attr (type)	设置打印机属性
A	Pr_init (preype)	初始化打印机
A	Pr_pause ( )	暂停打印机
A	Pr_prtsc ( )	打印当前屏幕
A	Pr_pute (c)	打印原字符
A	Pr_putch (c)	在标准打印机上打印一个字符
A	Pr_putff ( )	打印一换表字符
A	Pr_putl (buff)	打印一行正文
A	Pr_puts (str)	打印一字符串
A	Pr_sattr (type)	返回打印机属性字符串的指针
A	Pr_signal (on)	建立打印机中断服务例程
A	Pr_stat ( )	打印机准备例程
A	Pr_tabs ( )	返回打印机 tab 字符串

屏幕函数 (A = sc_handl.c, B = co_drive.asm, C = sc_field.c, D = sc_windo.c, E = sc_box.c)

A	sc_attr(atype)	设置显示属性
E	sc_box(x1,y1,x2,y2)	画一个 box
A	sc_clip(&col,&row)	剪取光标的位置
A	sc_ch()	清当前的窗口
C	sc_clrf(n)	清几个字节的屏幕域
A	sc_cirwin(col l,row1,co12,row2)	清一个屏幕窗口
A	sc_color(foolr,bcolr)	设置屏幕颜色
A	sc_cursor(flag)	设置光标/取消光标
A	sc_eeol()	擦除到行尾
A	sc_font(font)	装入新字形
A	sc_getch(col,row)	读取屏幕在col，row处的字符
A	sc_getcol()	获得光标所在的列
A	sc_getow()	读取光标所在的行
A	sc_getwin(col l,row1,co12,row2)	读取一个正文窗口的内容
E	sc_hline(row,c1,o2)	在C_1列到C_2列处显示一条水平线
D	sc_home()	置光标于起始位置

A	sc_init(mode,fcolr,bcolr)	初始化屏幕
A	sc_movcur(dcol,drow)	在屏幕上移动光标
B	sc_printf(format,var...)	printf 的另一种定义法
A	sc_putc(c)	在屏幕上显示字符并且移动光标
A	sc_putch(c)	在屏幕上显示字符并滚动
C	sc_putfi(i,n)	在屏幕上输出长为几个字节域的整数
C	sc_putifc(i,n)	将整数显示在屏幕域的中央
C	sc_putifr(i,n)	把整数按右对齐显示在屏幕域中
A	sc_putl(str)	在屏幕上输出一行
C	sc_putlf(line,n)	把一行信息显示在 n 个字节的屏幕域中
A	sc_puts(str)	在屏幕上输出字符串
C	sc_pustf(str,n)	把字符串显示在 n 个字节的屏幕域中
C	putsfc(str,n)	不清除地将一字符串显示在屏幕域的中央
C	sc_putsfl(str,n)	清除并显示字符串于屏幕域中
C	sc_putsflc(str,n)	清除且显示字符串于屏幕域的中央
C	sc_putsflr(str,n)	清除屏幕域，然后按右对齐将字符串显示在屏幕域中
C	sc_putsfr(str,n)	按右对齐在屏幕域中显示字符串
A	sc_putsv(str)	显示垂直方向的字符串
A	sc_putwin(buff,col 1,row1,col2,row2)	把正文缓冲区中的内容显示在窗口中
C	sc_scrld(row,nrows)	卷下屏幕 nrows 行
C	sc_scrlu(row,ntows)	卷上屏幕 nrows 行
A	sc_scroll(nrows,col 1,row1,col2,row2)	卷一部分屏幕
A	sc_setcut(col,row)	设置光标的绝对位置
E	sc_vline(col,r1,r2)	在 col 列的r1行到 r2行处显示一条垂线
A	sc_windo(col 1,row1,col2,row2)	设置一屏幕窗口
D	sc_winfull()	设置成物理屏幕
D	sc_winpop()	恢复屏幕参数
D	sc_winpush()	保存屏幕参数

串行函数（A = si-hanoll.c）

A	si_getc(pott)	从端口读取字符
A	si_init(port,parms)	初始化一个串行端口
A	si_mp(port)	从串行端口输入字符
A	si_out(port,c)	输出字符到串行端口上
A	si_putc(port,c)	输出字节至串行端口上

A	si_stat(port)	返回串行端口的状态

字符串函数（A = st_alph.c，B = st_iust.c，C = st_sub.c，D = st_conv.c）

文件	函　数	功　能
A	st_bcpy(buff,str)	将一字符串复制到缓冲区中
A	st_bllz(str)	以空格取代前导零
A	st_ccat(s1,s2,n)	在字符串 s_1 上附加几个字节的字符串 s_2
A	st_ccmpu(s1,s2,n)	以大写格式比较二个 n 个字节的字符串
B	st_cntr(str,n)	将字符串置于几个字节的屏幕域中央
A	st_delc(str,c)	在字符串中找定界符c
A	st_mbuf(str,buff,n)	查看字符串是否在缓冲区中
A	st_instr(str2,str1)	查看 str1 是否在 str2中
A	st_mstri(str2,str1)	不管大小写，查看 str1 是否在 str2中
B	st_iust(str,n)	把字符串调整到几个字节域的右边
C	st_left(str1,str2,n)	从字符串中抽取几个最左的字符
D	st_lower(str)	将字符串转换为小写
B	st_movl(str,n)	把字符串向右移动几个字节
B	st_movt(str,n)	把字符串向右移动几个字节
A	st_ncmpi(s1,s2,n)	不管大小写，比较几个字节的字符串
A	st_patma(str1,str2)	在str2 中，做 str1 的固定模式匹配
A	st_patmai(str1,str2)	固定模式匹配，不管大小写
A	st_ptrs(buff,pus,cnt)	获得指向缓冲区中字符串的指针
C	st_right(str1,str2,n)	从字符串中抽取几个最右的字符
A	st_squ(str,c)	挤出字符串中的所有c字符
C	st_subc(str,c,ch)	替换字符串中的字符c
C	st_substr(str1,str2,pos,n)	在 str2中，从pos处开始抽出n个 字符置于 str1中
A	st_tabr(str)	在字符串中扩展 tab
D	st_tof(str)	转换成浮点数的字符串
D	st_toi(str)	转换为整型的字符串
D	st_tol(str)	转换为长整型的字符串
D	st_uppet(str)	把字符串的小写字母转换为大写字母
A	st_wild(str)	查看字符串是否有无字符
A	st_zlbl(str)	以零代替头空格

实用函数（A = ut_crypt.c，B = ut_dtadd.c，C = ut_date.c，D = ut_dtstr.c E = ut_day.c，F = ut_stack.c，G = ut_qsort.c，H = ut_dtstr.c，I = ut_xlate.c）

文件	函　数	功　能
A	ut_crypt(buff.key.nbytes)	用键字来加密 n 个字节的缓冲区

B	ut_dateadd(sdate1,sdate2)	将 sdate1 与 sdate2 相加
B	ut_datecmp(sdate1,sdate2)	比较两个 date 结构，sdate1 和 sdate2
B	ut_datedif(sdate1,sdate2)	求两个日期的差
C	ut_datel()	读取长格式日期
C	ut_date(sdate)	获得 date 结构
B	ut-pdatesub(sdate1,sdate2)	从 date 结构 sdate1 中减去date结构sdate2
E	ut_daystr(day)	获得指向日字符的指针
C	ut_dump(memptr,nboytes)	将内存输出到 stdout 文件中去
C	ut_gdate()	获得日期
C	ut_gtime()	获得时间
C	ut_gtod()	获得每天的时间（日期和时间）
C	ut_hmtimestr(str,hr,mn)	将小时、分钟转换为字符串
D	ut_isleap(yt)	判断是否为闰年
B	ut_ldateadd(ldate1,ldate2)	将两个长日期相加
B	ut_ldatedif(ldate1,ldate2)	求两个长日期的差
C	ut_ldates(sdatel,date)	将长格式日期转换为结构格式日期
C	ut_ldatestr(str,ldate)	将长日期转换为字符串
B	ut_ldatestrb(ldate1,ldate2)	将两个长格式日期相减
E	ut_ldaystr(ldate)	从长日期返回一个周内日子的字符串
B	ut_ltimeadd(ltime1,ltime2)	将两个长时间相加
C	ut_ltimes(stime,ltime)	将长时间转换为结构时间
C	ut_ltimestr(str,ltime)	将长时间转换为字符串
B	ut_ltimesub(ltime1,ltime2)	从 ltime1 中减去 ltime2
E	ut_mostr(mo)	获得指向月名字符串的指针
C	ut_pause(time)	暂停百分之一秒
F	ut_pop()	从本地 C 栈中获值
F	ut_push(val)	将一个值压入栈
G	ut_qsott(buf,nel,width,compar)	在内存排序
C	ut_sdateatr(str,sdate)	将结构日期转换为字符串
E	ut_sday(sdate)	获得结构中的周内日值
E	ut_sjday(sdate)	从结构日期中求出与之相应的一年中的第几天
H	ut_stdatel(str)	将字符串日期转换为长日期
H	ut_stdates(sdate,str)	将字符串转换为日期结构
C	ut_stimestr(str,stime)	将结构时间转换为字符串时间
H	ut_sttimel(str)	将字符串时间转换为长时间
H	ut_sttimes(stime,str)	将字符串转换为时间结构
G	ut_swap(buf1,but12,n)	交换缓冲区中的两个元素
E	ut_thismo()	获得这个月的字符串

B	ut_timeadd(time1,time2)	将 time1 和 time2 相加
B	ut_umecmp(time1,time2)	比较两个时间
C	ut_time1()	获得时间
C	ut_times(stime)	获得一个结构时间
B	ut_timesub(time1,time2)	从 time1 中减去 time2
E	ut_today()	获得今天的周内日期整数
E	ut_todaystt()	返回指向今天周内日期名的指针
I	ut_xlateb(byte,table)	转换表内的字节
I	ut_xlprox(str,table)	在表中查询近似的值
I	ut_xlstr(str,table)	翻译表中的字符串
I	ut_xtable(str,table)	查询表中的字符串

系统函数（A = sy_handl.c）

A	sy_abort()	异常结束
A	sy_beep()	发出蜂鸣声
A	sy_doffi(data)	返回数据地址的偏移量
A	sy_dseg(data)	返回 date 的段地址
A	sy_fdirt()	寻找下一个出现的目录
A	sy_ffdir(path,attr)	寻找第一次出现的目录
A	sy_gdate()	获得当前日期
A	sy_getintv(vector.address)	获得中断向量
A	sy_getsys()	获得系统参数
A	sy_gtime()	获得当前时间
A	sy_indosf(ptr)	获得临界标志地址
A	sy_isdrive(number)	查看是否存在驱动号
A	sy_poff(function)	返回 function 地址的偏移量
A	sy_pseg(function)	返回 function 地址的段
A	sy_setintf(vector,function)	为 C 函数设置的中断矢量
A	sy_setintv(vector,address)	设置中断矢量地址

正文函数（A = tx_menu.c, B = tx_field.c）

A	tx_menu(menu)	返回菜单选择号
A	tx_menuclr(menu)	清除屏幕上的清单
A	tx_menudsp(menu)	显示菜单
A	tx_menuf(menu,item,attr)	显示一个菜单的条目
A	tx_menusel(menu)	获得初选择的菜单
A	tx_menust(menu)	返回被选择的字符串指针
B	tx_mesg(mesg)	一次输出一屏幕页信息
B	tx_putf(fp)	将文件输出到屏幕上
B	tx_putfp(fp,flpos)	将文件页输出到屏幕上