第 5 章　C51 的库函数

C51 运行时间库提供大量可用于 8051 系列 C 语言程序的预定义函数和宏。它大大简化了 8051 的 C 程序设计过程。

大部分库函数与 ANSI-C 兼容，其中部分函数为了能更好发挥 8051 结构的特性，做了少量的改动。例如，isdigit 函数返回的是位值，而不是整数。只要可能，函数的参数和返回值总是尽量采用体积最小的数据类型，总是尽量采用无符号数。这些对标准库的变动，提高了函数性能，也减小了程序代码体积。

所有函数的实现与函数选用的寄存器组无关。

5.1　C51 的库文件

C51 有 6 种编译时间库，支持绝大部分 ANSI-C 函数。它们分别适用于不同的应用存储模式，见表 5-1。

表 5-1　编译时间库

库 文 件	说　明
C51S.LIB	小模式，无浮点运算
C51FPS.LIB	小模式，有浮点运算
C51C.LIB	紧凑模式，无浮点运算
C51FPC.LIB	紧凑模式，有浮点运算
C51L.LIB	大模式，无浮点运算
C51FPL.LIB	大模式，有浮点运算

实现与硬件相关的低级流输入输出功能的函数，以源文件的形式提供，它们可以在 LIB 目录下找到。通过修改这些文件，可以替换库中相应的库程序，使得函数库能够适应目标系统中的流输入输出环境。

5.2　C51 的库函数分类

库函数分为几大类，基本上分属于不同的 H 文件。这些文件在 INC 目录下可以找到，其中包含了常数定义、宏定义、类型定义和原型函数。以下先按 H 文件，分别对各个库函数做简要说明。

5.2.1 ABSACC.H

ABSACC.H 中包含了允许直接访问 8051 不同区域存储器的宏。

1. CBYTE

允许访问 8051 程序存储器中的字节。例如：

```
rval = CBYTE [0x0002];
```

从程序存储器地址 0002h 读出内容。

2. CWORD

允许访问 8051 程序存储器中的字。例如：

```
rval = CWORD [0x0002];
```

从程序存储器地址 0004h 读出内容，地址计算：2*sizeof (unsigned int)。

3. DBYTE

允许访问 8051 片内 RAM 中的字节。例如：

```
rval = DBYTE [0x0002];
DBYTE [0x0002] = 5;
```

从片内 RAM 地址 02h 读出或写入内容。

4. DWORD

允许访问 8051 片内 RAM 中的字。例如：

```
rval = DWORD [0x0002];
DWORD [0x0002] = 57;
```

从片内 RAM 地址 0004h 读出和写入内容。

5. PBYTE

允许访问 8051 片外 RAM 页面中的字节。例如：

```
rval = PBYTE [0x0002];
PBYTE [0x0002] = 57;
```

从片外 RAM 页的相对地址 0002h 读出或写入内容。

6. PWORD

允许访问 8051 片外 RAM 页面中的字。例如：

```
rval = PWORD [0x0002];
PWORD [0x0002] = 57;
```

从片外 RAM 页的相对地址 0004h 读出和写入内容。

7. XBYTE

允许访问 8051 片外 RAM 页面中的字节。例如：

```
rval = PBYTE [0x0002];
PBYTE [0x0002] = 57;
```

从片外 RAM 地址 0002h 读出或写入内容。

8. XWORD

允许访问 8051 片外 RAM 中的字。例如：

```
rval = XWORD [0x0002];
XWORD [0x0002] = 57;
```

从片内 RAM 地址 0004h 读出和写入内容。

5.2.2　ASSERT.H

ASSERT.H 包含 assert 宏。assert 对程序生成测试条件。

5.2.3　CTYPE.H

CTYPE.H 中包含 ASCII 字符的分类和转换函数。

Isalnum	可重入，测试是否为字母数字。
isalpha	可重入，测试是否为字母。
iscntrl	可重入，测试是否为控制字符。
isdigit	可重入，测试是否为十进制数字。
isgraph	可重入，测试是否为可打印字符，不包括空格。
islower	可重入，测试是否为小写字母。
isprint	可重入，测试是否为可打印字符，包括空格。
ispunct	可重入，测试是否为标点符号。
isspace	可重入，测试是否为空白字符。
isupper	可重入，测试是否为大写字母。
isxdigit	可重入，测试是否为十六进制数字。
toascii	可重入，将字符转换成 7 位 ASCII 码。
toint	可重入，将十六进制数字转换成十进制数。
tolower	可重入，测试字符并将大写字母转换成小写字母。
_tolower	可重入，无条件将字符转换成小写。
toupper	可重入，测试字符并将小写字母转换成大写字母。
_toupper	可重入，无条件将字符转换成大写。

5.2.4　INTRINS.H

INTRINS.H 包含内部函数，编译时产生的是插入代码，而不是产生 ACALL 或 LCALL 指令去调用一个功能函数。因此代码量小，效率更高。

chkfloat	内部函数，检查浮点数状态，返回说明浮点数状态的无符号字符。
crol	内部函数，无符号字符左旋。
cror	内部函数，无符号字符右旋。
irol	内部函数，无符号整数左旋。
iror	内部函数，无符号整数右旋。
lrol	内部函数，无符号长整数左旋。
lror	内部函数，无符号长整数右旋。
nop	内部函数，在程序中插入 NOP 指令。
testbit	内部函数，在程序中插入 JBC 指令。

5.2.5 MATH.H

MATH.H 中包含算术运算函数，包括浮点运算。

abs	可重入，求整数的绝对值。
acos	计算反余弦。
asin	计算反正弦
atan	计算反正切。
atan2	计算分数的反正切。
cabs	可重入，求字符的绝对值。
ceil	求大于等于参数的最小整数。
cos	计算余弦。
cosh	计算双曲余弦。
exp	计算参数的指数函数。
fabs	可重入，求浮点数的绝对值。
floor	求小于等于浮点数的最大整数。
fmod	计算浮点数的余数。
labs	可重入，求长整数的绝对值。
log	计算参数的自然对数。
log10	计算参数的常用对数。
modf	分离参数的整数和分数部分。
pow	计算幂函数。
sin	计算正弦。
sinh	计算双曲正弦
sqrt	计算平方根。
tan	计算正切。
tanh	计算双曲正切。

5.2.6 SETJMP.H

SETJMP.H 定义用于 setjmp 和 longjmp 程序的 jmp_buf 类型。

jmp_buf	用于 setjmp 和 longjmp 中保护和恢复程序环境。jmp_buf 类型定义如下：
	`#define _JBLEN 7`
	`typedef char jmp_buf[ _JBLEN];`

longjmp	长跳转。
setjmp	设置长跳转的返回点。

5.2.7 STDARG.H

STDARG.H 定义访问函数参数的宏。定义保持函数调用参数的 va_list 数据类型。

va_arg	读函数调用中的下一个参数。
va_end	结束读函数调用参数。
va_start	开始读函数调用参数。

5.2.8　STDDEF.H

STDDEF.H 定义 offsetof 宏。

Offsetof	计算结构成员的偏移量。

5.2.9　STDIO.H

STDIO.H 中包含流输入输出的原型函数，定义 EOF 常数。

getchar	可重入，用_getkey 和 putchar 读入和回应一个字符。
_getkey	用 8051 串行接口读一个字符。
gets	用 getchar 读入一个字符串。
printf	用 putchar 写格式化数据。
putchar	用 8051 串行接口写一个字符。
puts	可重入，用 putchar 写字符串和换行字符。
scanf	用 getchar 读格式化数据。
sprintf	写格式化数据到字符串。
sscanf	从字符串读格式化数据。
ungetchar	将一个字符返回到 getchar 输入缓存。
vprintf	用指针向流输出。
vsprintf	写格式化数据到字符串。

5.2.10　STDLIB.H

STDLIB.H 中包含数据类型转换和存储器定位函数。

atof	将字符串转换成浮点数。
atoi	将字符串转换成整数。
atol	将字符串转换成长整数。
calloc	为数组在存储池中定位。
free	释放用 calloc、malloc 或 realloc 定位的存储块。
init_mempool	初始化存储池的定位和体积。
malloc	从存储池中定位一个存储块。
rand	可重入，产生一个伪随机数。
realloc	从存储池中重定位一个存储块。
srand	初始化伪随机数发生器。
strtod	将字符串转换成浮点数。
strtol	将字符串转换成长整数。
strtoul	将字符串转换成无符号长整数。

5.2.11　STRING.H

STRING.H 中包含字符串和缓存操作函数，定义了 NULL 常数。

memccpy	从一个缓存向另一个复制，直至复制了指定字符或指定字符数。
memchr	可重入，返回指定字符在缓存中首次出现的位置指针。

memcmp	可重入，对两个缓存中给定数量字符做比较。
memcpy	可重入，将给定数量字符从一个缓存复制到另一个。
memmove	可重入，将给定数量字符从一个缓存移动到另一个。
memset	可重入，将缓存中指定字节初始化为指定值。
strcat	连接两个字符串。
strchr	可重入，返回指定字符在字符串中首次出现的位置指针。
strcmp	可重入，比较两个字符串。
strcpy	可重入，复制字符串。
strcspn	返回字符串中首字符与另一个字符串匹配的位置指针。
strlen	可重入，返回字符串的长度。
strncat	将字符串中指定字符连接到另一个字符串。
strncmp	比较两个字符串的指定数量字符。
strncpy	将字符串中指定数量字符复制到另一个字符串。
strpbrk	返回一个字符串中与另一个字符串匹配的第一个字符的位置指针。
strpos	可重入，返回字符串中指定字符首次出现的位置指针。
strrchr	可重入，返回字符串中指定字符最后出现的位置指针。
strrpbrk	返回字符串中最后一个与另一字符串中任意字符匹配的字符位置指针。
strrpos	可重入，返回字符串中指定字符最后出现的位置指针。
strspn	返回字符串中第一个与另一字符串中任意字符不匹配的字符位置指针。
strstr	返回字符串中与另一个字符串相同的子串的位置指针。

5.3 C51 库函数说明

以下是按字母顺序排列的库函数详细说明。

5.3.1 abs

函数原型	#include <math.h> int abs(int x);
参数	x：整型值。
功能说明	求 x 的绝对值。
返回值	x 的绝对值，整型。

5.3.2 acos

函数原型	#include <math.h> float acos(float x);
参数	x：在 [-1，+1] 范围内的浮点数。
功能说明	求 x 的反余弦主值，弧度。
返回值	x 的浮点反余弦值，在 [0，pi] 范围内。

5.3.3　asin

函数原型	#include <math.h> float asin(float x);
参数	x：在 [-1，+1] 范围内的浮点数。
功能说明	求 x 的反正弦，弧度。
返回值	x 的浮点反正弦值，在 [-pi/2，+pi/2] 范围内。

5.3.4　assert

函数原型	#include <assert.h> void assert(int expr);
参数	expr：被检查的表达式。
功能说明	检查表达式的宏。如果结果为假，输出到 printf 打印出错消息。
返回值	无。

5.3.5　atan

函数原型	#include <math.h> float atan(float x);
参数	x：浮点数。
功能说明	求 arg 的反正切值，弧度。
返回值	x 的浮点反正切值，在 [-pi/2，+pi/2] 范围内。

5.3.6　atan2

函数原型	#include <math.h> float atan2(float x, float y);
参数	x：浮点数。 y：浮点数。
功能说明	求 x/y 的反正切值，弧度，用参数符号确定返回值所在象限。
返回值	x/y 的浮点反正切值，在 [-pi/2，+pi/2] 范围内。

5.3.7　atof

函数原型	#include <stdlib.h> float atof(void *string);
参数	string：指向 ASCII 码数字字符串的指针。
功能说明	将 string 所指的 ASCII 字符串转换为浮点数。 转换时跳过空白字符，并在遇到不可识别的字符时结束。
返回值	字符串所表达的浮点数。
举例	“　-3K”转换成：-3.00 “.0006”转换成：0.0006 “ 1e-4”转换成：0.0001

5.3.8 atoi

函数原型	#include <stdlib.h> int atoi(void *string);
参数	string： 指向 ASCII 码数字字符串的指针。
功能说明	将 string 所指向的 ASCII 字符串转换为整型数。 转换时跳过空白字符，并在遇到不可识别的字符时结束。
返回值	字符串所表达的整型数。
举例	“ -3K”转换成：-3。 “149”转换成：149。

5.3.9 atol

函数原型	#include <stdlib.h> long atol(void *string);
参数	string： 指向 ASCII 码数字字符串的指针。
功能说明	将 string 所指向的 ASCII 字符串转换为长整型数。 转换时跳过空白并在遇到不可识别的字符时结束。
返回值	字符串所表达的长整型数。

5.3.10 cabs

函数原型	#include <math.h> char cabs(char x);
参数	x：字符型。
功能说明	求 x 的绝对值。
返回值	x 的绝对值，字符型。

5.3.11 fcalloc

函数原型	#include <stdlib.h> void *calloc(unsigned int num, unsigned int len);
参数	num：数组元素数，无符号整型。 len：数组元素体积，无符号整数。
功能说明	为给定大小的目标数组分配存储块，并初始化为 0。 空间体积的字节数为元素与元素体积的乘积。
返回值	成功：指针指向数组存储块的最低字节地址。 失败：如果没有数组所需存储块可分配时，为 0。

5.3.12 ceil

函数原型	#include <math.h> float ceil(float x);
参数	x：浮点数。

功能说明　求大于等于 x 的最小整数。
返回值　大于等于 x 的最小整数值，浮点数。

5.3.13　cos

函数原型　#include <math.h>
float cos(float x);
参数　x：浮点数。
功能说明　求 x 弧度的余弦。
返回值　x 的余弦值，浮点数。

5.3.14　cosh

函数原型　#include <math.h>
float cosh(float x);
参数　x：浮点数。
功能说明　求 arg 弧度的双曲余弦。
返回值　x 的浮点双曲余弦值。

5.3.15　exp

函数原型　#include <math.h>
float exp(float x);
参数　x：浮点数。
功能说明　求 x 的指数函数。
返回值　x 的指数函数值，浮点型。

5.3.16　fabs

函数原型　#include <math.h>
float fabs(float x);
参数　x：浮点数。
功能说明　求浮点数 x 的绝对值。
返回值　x 的浮点绝对值。

5.3.17　floor

函数原型　#include <math.h>
float floor(float x);
参数　x：浮点数。
功能说明　求小于等于 x 的最大整数。
返回值　含有小于等于 x 的最大整数的浮点数。

5.3.18　fmod

函数原型　#include <math.h>

float fmod(float x, float y);

参数　x：被除数，浮点数。
　　y：除数，浮点数。

功能说明　求 x/y 的余数，即：x – i * y。
　　其中，i 为结果与 x 同号，结果绝对值小于 y 绝对值的整数（y<>0）。

返回值　x 除以 y 得到的浮点余数。

5.3.19 free

函数原型　#include <stdlib.h>
　　viod free(void xdata *p);

参数　p：此前 malloc、calloc 或 realloc 分配的存储区指针。

功能说明　释放被 p 所指向的存储块。
　　p 必须在此以前已由 malloc、calloc 或 realloc 赋值。

返回值　无。

5.3.20 getchar

函数原型　#include <stdio.h>
　　char getchar(void);

参数　无。

功能说明　从标准输入流用_getkey 函数读一个字符。
　　读入字符传递给 putchar 函数用于回应。

返回值　来自标准输入流的下一个字符，整型，含有 ASCII 码值。

5.3.21 gets

函数原型　#include <stdio.h>
　　char *gets(char *string, int len);

参数　string：指向由标准输入接收到字符串的指针；
　　len：可读入的最大字符数。

功能说明　调用 getchar 函数读字符串行到 string，字符串行以换行符结束。读入后换行符改为空字符。len 功能说明可读的最大字符数，如遇到换行符前读入已达最大字符数，则停止读入，并以空字符作为读入字符的结束。

返回值　成功：指针，与 s 相同。
　　失败：null。

5.3.22 init_mempool

函数原型　#include <stdlib.h>
　　void init_mempool(void xdata *p, unsigned int size);

参数　p：存储池首地址。
　　size：存储池字节数。

功能说明　初始化存储池的定位和体积。p 可用于 free、calloc、malloc 和 realloc。
返回值　无。

5.3.23　isalnum

函数原型　#include <ctype.h>
bit isalnum(char c);
参数　c：表示检查字符的字符型数。
功能说明　测试字符是否为字母或数字。
返回值　如果 c 是字母或数字，则为 1，否则为 0。

5.3.24　isalpha

函数原型　#include <ctype.h>
bit isalpha(char c);
参数　c：表示检查字符的字符型数。
功能说明　测试字符是否为字母。
返回值　如果 c 是字母，则为 1，否则为 0。

5.3.25　iscntrl

函数原型　#include <ctype.h>
bit iscntrl(char c);
参数　c：表示检查字符的字符型数。
功能说明　测试字符是否为控制字符。
返回值　如果 c 是控制码，则为 1，否则为 0。

5.3.26　isdigit

函数原型　#include <ctype.h>
bit isdigit(char c);
参数　c：表示检查字符的字符型数。
功能说明　测试字符是否为 10 进数字。
返回值　如果 c 是数字，则为 1，否则为 0。

5.3.27　isgraph

函数原型　#include <ctype.h>
bit isgraph(char c);
参数　c：表示检查字符的字符型数。
功能说明　测试字符是否为非空白可打印字符。
返回值　如果 c 是非空白可打印字符，则为 1，否则为 0。

5.3.28　islower

函数原型　#include <ctype.h>

bit islower(char c);

参数　c：表示检查字符的字符型数。

功能说明　测试字符是否为小写字母。

返回值　如果 c 是小写字母，则为 1，否则为 0。

5.3.29 isprint

函数原型　#include <ctype.h>

bit isprint(char c);

参数　c：表示检查字符的字符型数。

功能说明　测试字符是否为包括空格在内的可打印字符。

返回值　如果 c 是可打印字符（包括空格），则为 1，否则为 0。

5.3.30 ispunct

函数原型　#include <ctype.h>

bit ispunct(char c);

参数　c：表示检查字符的字符型数。

功能说明　测试字符是否为除了空格、数字或字母之外的可打印字符。

返回值　如果 c 可打印且非空格、数字或字母，则为 1，否则为 0。

5.3.31 isspace

函数原型　#include <ctype.h>

bit isspace(char c);

参数　c：表示检查字符的字符型数。

功能说明　测试字符是否为空白字符，空白字符定义为下列字符之一：

空格：' '

换页：\f

换行：\n

回车：\r

水平制表符：\t

垂直制表符：\v

返回值　如果 c 是空白字符，则为 1，否则为 0。

5.3.32 isupper

函数原型　#include <ctype.h>

bit isupper(char c);

参数　c：表示检查字符的字符型数。

功能说明　测试字符是否为大写字母。

返回值　如果 c 是大写字母，则为 1，否则为 0。

5.3.33　isxdigit

函数原型	#include <ctypc.h> bit isxdigit(char c);
参数	c：表示检查字符的字符型数。
功能说明	测试字符是否为大写或小写十六进制数字，即 0~9、a~f 或 A~F 之一。
返回值	如果 c 是大写或小写数字，则为 1，否则为 0。

5.3.34　labs

函数原型	#include <math.h> long labs(long x);
参数	x：长整型值。
功能说明	求长整形数 x 的绝对值。
返回值	x 的绝对值，长整型。

5.3.35　log

函数原型	#include <math.h> float log(float x);
参数	x：浮点数。
功能说明	求 x 的自然对数。
返回值	x 的自然对数值，浮点数。

5.3.36　log10

函数原型	#include <math.h> float log10(float x);
参数	x：浮点数。
功能说明	求 x 的以 10 为底的常用对数值。
返回值	x 的以 10 为底的对数值，浮点数。

5.3.37　longjmp

函数原型	#include <setjmp.h> viod longjmp(jmp_buf env, int val);
参数	env：jmp_buf 结构类型，是由 setjmp 保存的环境。 val：返回到相应的 setjmp 处的整型值。
功能说明	恢复先前由 setjmp 保存的环境，使程序长跳转到 setjmp 处继续执行。用 val 作为函数的返回值。
返回值	无。

5.3.38　malloc

函数原型	#include <stdlib.h>

void *malloc(unsigned int size);

参数　size：缓存体积字节数。

功能说明　为说明体积的缓存分配存储区。

返回值　成功：指向缓存存储区最低字节地址的指针。

失败：如果没有缓存所需存储区可分配，为 0。

5.3.39 memccpy

函数原型　#include <string.h>

void *memccpy(void *s1, const void *s2, char c, int len);

参数　s1：指向目的缓存的指针。

s2：指向源缓存的指针。

c：结束复制的字符。

len：复制的最大字符数。

功能说明　将字符从源复制到目的，直至复制 len 个字符或复制字符 c。

返回值　完整复制为 s1。如果复制了指定的结束字符则为 0。

5.3.40 memchr

函数原型　#include <string.h>

void *memchr(void *s, char c, int len);

参数　s：指向缓存的指针。

c：表示要搜索字符的字符型数。

len：缓存最大长度，整型。

功能说明　在指针所指缓存前 len 字节内搜索首次出现的字符 c。

返回值　成功：指向 c 在缓存中首次出现的位置的指针 s。

失败：null。

5.3.41 memcmp

函数原型　#include <string.h>

char memcmp(void *s1, void *s2, int len);

参数　s1：指向第一个缓存的指针。

s2：指向第二个缓存的指针。

len：比较的最大字节数。

功能说明　比较两个缓存的前 len 个字符。

返回值　字符型，指示由 s1、s2 所指缓存前 n 个字符的比较结果。

返回值含义：　>0：　s1 > s2

=0：　s1 = s2

<0：　s1 < s2

5.3.42　memcpy

函数原型　#include <string.h>
　　void *memcpy(void *s1, const void *s2, int len);
参数　s1：指向目的缓存的指针。
　　s2：指向源缓存的指针。
　　len：复制的最大字符数。
功能说明　将指定数目的字符从源复制到目的，
　　如存储器重叠，则结果是不确定的。应用 memmove 函数替代。
返回值　s1。

5.3.43　memmove

函数原型　#include <string.h>
　　void *memmove(void *s1, void *s2, int len);
参数　s1：指向目的缓存的指针。
　　s2：指向源缓存的指针。
　　len：移动的最大字节数。
功能说明　将 len 个字符从源复制到目的。能保证复制时不发生存储器重叠。
返回值　s1。

5.3.44　memset

函数原型　#include <string.h>
　　void *memset(void *s, int c, int len);
参数　s：指向目的缓存的指针。
　　c：表示字符字符型数。
　　len：缓存大小。
功能说明　将缓存的前 n 字节设置为 c。
返回值　s。

5.3.45　modf

函数原型　#include <math.h>
　　float modf(float value, float *iptr);
参数　value：待分离的浮点数。
　　iptr：指向 value 的整数部分的指针，浮点数。
功能说明　分离 value 的小数和整数部分，两部分的符号与 value 相同。
返回值　value 的小数部分，有符号。

5.3.46　offsetof

函数原型　#include <stddef.h>
　　int offsetof (struc, mem);

参数　　struc：结构。

mem：结构的成员。

功能说明　求结构的成员的偏移量，实现结构成员的定位。

返回值　　结构成员对于结构开始地址的偏移量字节数。

5.3.47 pow

函数原型　#include <math.h>

float pow(float x, float y);

参数　　x：浮点数。

y：浮点数。

功能说明　计算一个数的幂。如果 x<>0,y=0，返回值为 1。

如果 x=0,y<=0 或者 x<0,y 不是整数，返回 NaN（非数据错）。

返回值　　x 的 y 次幂。

5.3.48 printf

函数原型　#include <stdio.h>

int printf(const char *format, ...);

参数　　format：指向格式化字符串的指针。

...：在 format 控制下的待打印数据。

功能说明　将格式化数据用 putchar 函数写到标准输出流。format 是一个字符串，它包含字符、字符序列和格式说明。字符与字符序列按顺序复制到流。格式说明以百分符（%）开始，格式说明使跟随的相同序号的数据按格式说明转换和输出。如果数据的数量多于格式说明，多余的数据被忽略。如果格式说明多于数据，结果将不可预测。

格式说明为：

%_flags__width__.precision__{hlBIIfL}_type

type：说明参数是字符、字符串、数字或指针字符，见表 5-2。

表 5-2　数制转换

Type	结　果
D	有符号十进制
U	无符号十进制
O	无符号八进制
x	无符号十六进制，使用小写（0~9、a~f）
X	无符号十六进制，使用大写（0~9、A~F）
f	形式为 [-] ddd.ddd 的浮点数
e	形式为 [-] d ddde+dd 的浮点数
F	形式为 [-] d.dddE+dd 的浮点数

（续表）

Type	结　果
g	取 f 或 e 中用较合适形式的浮点数
G	取 f 或 E 中用较合适形式的双精度值
c	单字符常数
s	字符串常数
p	指针，格式：t:aaaa，其中：aaaa 为十六进制地址 t 为存储类型，c：代码；i：片内 RAM；x：片外 RAM；p：片外 RAM 页
n	无输出，但在下一参数所指整数中写入字符数
%	% 字符

b、B、l、L：加在 type 前，说明整型 d、i、u、o、x、X 的 char 或 long 转换。

flags：标志，功能如表 5-3 所示。

表 5-3　标志意义

flags	作　用
-	域左对齐
+	有符号，数值总是以正负号开始
空格	数值总是由负号或空格开始
#	变换形式：o、x、X，首数字为 0、0x、0X G、g、E、e、f：打印小数点
*	忽略

width：域宽。非负数，说明打印字符的最小数。如打印字符少用空格填充，在前面加负号表示域左对齐，加 0 表示用 0 填充。如打印字符多于域宽，仍输出全部字符。* 号表示后续的整数参数提供域宽，前面加 b，表示后续参数是无符号字符。

Precision：精度。对不同类型意义不同，会引起截尾或舍入，见表 5-4。

表 5-4　精度对不同数据类型的作用

数据类型	意　义
d、u、o、x、X	输出数字的最小数，如输出数字超出，不截尾，如超出在左面填 0
f、e、E	输出数字的小数位数，末位四舍五入
g、G	输出数字的有效位数
c、p	不影响
s	输出字符的最大字符数，超过部分将不输出

精度可以是 * 号，表示后续的参数是整型；前面加 b，表示参数是无符号字符。

返回值　所写的字符数。

5.3.49 putchar

函数原型　#include <stdio.h>
char putchar(char c);
参数　c：表示输出的字符，字符型。
功能说明　用 8051 的串行接口输出字符 c。
返回值　c。

5.3.50 puts

函数原型　#include <stdio.h>
int puts(const char *s);
参数　s：指向待写字符串的指针。
功能说明　用 putchar 函数将字符串及换行字符写到输出流。
返回值　成功：0。
失败：如果出错，返回 EOF。

5.3.51 rand

函数原型　#include <stdlib.h>
int rand(void);
参数　无。
功能说明　产生伪随机数，分布在[0，32767]范围内。
返回值　随机数序列的下一个数。

5.3.52 realloc

函数原型　#include <stdlib.h>
void * realloc(void xdata *p, unsigned int size);
参数　p：指向存储区起始地址的指针。
size：说明缓存大小的值，size_t 类型。
功能说明　改变原先由 malloc、calloc 或 realloc 分配的存储区体积。p 指向已分配的存储块，size 说明新的块体积。块中原有内容复制到新块，如新块体积较大，超过部分未做初始化。
返回值　成功：指向新存储区最低地址的指针。
失败：如果没有所需存储区可供分配，返回 null。

5.3.53 scanf

函数原型　#include <stdio.h>

int scanf(const char *format,...);

参数　format：指向格式字符串的指针。

...：指向接收数据变量的指针，可选。

功能说明　用 getchar 函数按格式化字符串及参数读入并保存到参数。每个参数必须是一个指向变量的指针，变量的类型在格式化字符串中定义，用于解释读入的数据。格式说明包含空白字符、非空白字符、格式说明符，定义如下：

- 空白字符（空格、制表符、换行符）使扫描输入时跳过；
- 除百分号（%）外的非空白字符产生扫描读入，但不保存。如果流输入的字符与说明的非空白字符不匹配时，扫描停止；
- 格式说明以百分号开始，产生扫描，按说明的类型转换读入字符，并存入参数表中的参数。

后跟百分号的字符不当做格式说明，只作为普通字符处理。例如，%% 与输入中的一个百分号匹配。

不是格式说明成分的字符必须与输入字符匹配，从流读入但是不保存。如果输入流的字符与格式说明字符串冲突，扫描输入停止。

格式字符串中的格式说明引用相应顺序的格式参数，并以格式说明转换输入字符并保存。如果格式参数超过格式说明符，多余参数被忽略。如果格式参数不足，结果不可预测。

输入流中的数据以空白字符分隔，输入时遇见空白字符是结束转换。遇见当前格式无法解释的字符时也结束转换。格式说明的格式如下：

% _*_ _width_ _{b | h | l}_ type

格式说明中的每个域用一个字符或数字来说明格式选项。

type：说明输入字符解释为字符、字符串或数字，见表 5-5。

表 5-5　类型字符意义

字　符	参 数 类 型	输 入 格 式
d	int *	有符号十进制整数
i	int *	有符号十、十六、八进制数整数
u	unsigned int *	无符号十进制数
o	unsigned int *	无符号八进制数
x	unsigned int *	无符号十六进制数
e	float *	浮点数
f	float *	浮点数
g	float *	浮点数
c	char *	字符
s	char *	字符串或以空白字符结束的字符序列

以(*)号开始的格式说明产生输入扫描、但不存储。

width：非负数，说明从输入流读入字符的最大数。超过 width 的字符不再读入与转换。但不包括其中的空白字符和不可识别字符。可选字符 b、h 和 l 加在 type 之前，分别说明 d、i、u、o 和 x 类型是 char、int 或 long。

返回值　成功：输入成功的字段数量。
失败：返回 EOF。

5.3.54 setjmp

函数原型　#include <setjmp.h>
int setjmp(jmp_buf env);
参数　env：jmp_buf 类型缓存，当前环境。
功能说明　保存当前 CPU 状态，供此后调用的 longjmp 函数用。
setjmp 与 longjmp 一起使用，提供了非局部转移的方法。调用 setjmp 保护当前指令地址和 CPU 寄存器等状态，此后调用 longjmp 时恢复状态，使程序从调用 setjmp 后处继续执行。
返回值　CPU 状态复制到 env 后，返回 0。
执行 longjmp 函数，返回非 0 值。

5.3.55 sin

函数原型　#include <math.h>
float sin(float x);
参数　x：浮点弧度值。必须在-65535~+65535 之间。
功能说明　求 x 弧度的正弦。
返回值　x 的浮点正弦值。如 x 数值越界，返回 NaN 错。

5.3.56 sinh

函数原型　#include <math.h>
float sinh(float x);
参数　x：浮点弧度值。必须在-65535~+65535 之间。
功能说明　求 x 弧度的双曲正弦值。
返回值　x 的浮点双曲正弦值。如 x 数值越界，返回 NaN 错。

5.3.57 sprintf

函数原型　#include <stdio.h>
int sprintf(char *s, const char *format,...);
参数　s：指向接收格式化数据的字符串的指针。
format：指向格式化字符串的指针。
...：在 format 控制下打印的可选值。
功能说明　除了直接输出到字符串外，操作和 printf 完全相同，详见 printf。

返回值　输出的字符数。

5.3.58　sqrt

函数原型　#include <math.h>
float sqrt(float x);
参数　x：浮点数。
功能说明　求 x 的平方根。
返回值　x 的浮点平方根。

5.3.59　srand

函数原型　#include <stdlib.h>
void srand(unsigned int seed);
参数　seed：确定特定随机数序列的无符号整型值。
功能说明　伪随机数发生器初始化。对于给定的 seed，产生的随机数序列相同。
返回值　无。

5.3.60　sscanf

函数原型　#include <stdio.h>
int sscanf(const char *s, const char *format, ...);
参数　s：指向含有数据的字符串的指针。
format：指向格式化字符串的指针。
...：指向将接收数据的变量的指针，可选。
功能说明　除了从字符串读取外，操作与 scanf 完全一样。详见 scanf。
返回值　成功：输入成功的字段数量。
失败：返回 EOF。

5.3.61　strcat

函数原型　#include <string.h>
char *strcat(char *s1, const char *s2);
参数　s1：指向第一字符串的指针。
s2：指向第二字符串的指针。
功能说明　将第二字符串复制后添加到第一字符串的末尾。
第二字符串的首字符覆盖第一字符串的结束字符（null）。
返回值　s1。

5.3.62　strchr

函数原型　#include <string.h>
char *strchr(const char *s, int c);
参数　s：指向字符串的指针。
c：表示字符的整数值。

功能说明　在字符串中搜索指定字符的首次出现位置，结束字符 null 也视为字符串的一部分。

返回值　成功：指向 c 在 s 所指字符串中首先出现的位置的指针。
失败：找不到 c，返回 null。

5.3.63 strcmp

函数原型　#include <string.h>
int strcmp(const char *s1, const char *s2);

参数　s1：指向第一字符串的指针。
s2：指向第二字符串的指针。

功能说明　比较 2 个字符串。

返回值　比较 2 个字符串的结果，整型数，意义如下：

>0	s1 > s2
=0	s1 = s2
<0	s1 < s2

5.3.64 strcpy

函数原型　#include <string.h>
char *strcpy(char *s1, const char *s2);

参数　s1：指向目的字符串的指针。
s2：指向源字符串的指针。

功能说明　将源字符串复制到目的字符串，以 null 结束。

返回值　s1。

5.3.65 strcspn

函数原型　#include <string.h>
int strcspn(char *s1, char *s2);

参数　s1：指向主题字符串的指针。
s2：指向对象字符串的指针。

功能说明　查找主题字符串中不包含对象字符串中字符的最大起始片段。

返回值　s1 字符串中最大片段，其中的字符全不在 s2 字符串中。s1 中第一个字符在 s2 中，返回 0；s1 中字符全不在 s2 中，返回 s1 长度。

5.3.66 strlen

函数原型　#include <string.h>
int strlen(char *s);

参数　s：指向字符串的指针。

功能说明　求字符串中的字符数，但不包括结束字符（null）。

返回值　字符串长度。

5.3.67　strncat

函数原型　#include <string.h>

char *strncat(char *s1, const char *s2, int len);

参数　s1：指向目的字符串的指针。

s2：指向源字符串的指针。

len：用于连接的源字符串中的字符数。

功能说明　将源字符串中不超过 n 个的起始字符连接到目的字符串的末尾。

返回值　s1。

5.3.68　strncmp

函数原型　#include <string.h>

char strncmp(char *s1, char *s2, int len);

参数　s1：指向第一个字符串的指针。

s2：指向第二个字符串的指针。

len：用于比较的源字符串的字符数。

功能说明　比较 2 个字符串中不超过 n 个的起始字符。

返回值　2 个字符串中不超过 n 个的起始字符比较的整型结果，意义如下：

>0	s1 > s2
=0	s2 = s2
<0	s1 < s2

5.3.69　strncpy

函数原型　#include <string.h>

char *strncpy(char *s1, char *s2, int len);

参数　s1：指向目的字符串的指针。

s2：指向源字符串的指针。

len：要复制的源字符串中的字符数。

功能说明　将来自源字符中不超过 n 个的起始字符复制到目的字符串。

返回值　s1。

5.3.70　strpbrk

函数原型　#include <string.h>

char *strpbrk(char *s1, char *s2);

参数　s1：指向主题字符串的指针。

s2：指向对象字符串的指针。

功能说明　在一个字符串中搜索第二字符串中任意字符的出现位置，不包括 null。

返回值　成功：指针指向 s1 字符串中，首次出现 s2 字符串中任意字符的位置。

失败：未发现，返回 null。

5.3.71 strpos

函数原型　#include <string.h>
　　int strpos(const char *s, char c);
参数　s：指向字符串的指针。
　　c：搜索字符。
功能说明　在一个字符串中搜索字符 c 首次出现的位置，包括 null。
返回值　成功：s 字符串中首次出现 c 字符的位置，第一个字符为 0。
　　失败：未发现，返回-1。

5.3.72 strrchr

函数原型　#include <string.h>
　　char *strrchr(const char *s, char c);
参数　s：指向字符串的指针。
　　c：表示字符的整数。
功能说明　在字符串中搜索字符 c 最后出现的位置，包括 null。
返回值　成功：指针指向 s 所指字符串中 c 最后出现的位置。
　　失败：未发现，返回 null。

5.3.73 strrpbrk

函数原型　#include <string.h>
　　char *strrpbrk(char * s1, char *s2);
参数　s1：指向主题字符串的指针。
　　s2：指向对象字符串的指针。
功能说明　在字符串中搜索另一字符串中任意字符最后出现的位置，不包括 null。
返回值：　成功：指针指向 s1 字符串中，最后出现 s2 字符串中任意字符的位置。
　　失败：未发现，返回 null。

5.3.74 strrpos

函数原型　#include <string.h>
　　int strrpos(const char *s, char c);
参数　s：指向字符串的指针。
　　c：搜索字符。
功能说明　在字符串中搜索字符 c 最后出现的位置，包括 null。
返回值　成功：s 字符串中最后出现字符 c 的位置，第一个字符为 0。
　　失败：未发现，返回-1。

5.3.75 strspn

函数原型　#include <string.h>
　　int strspn(char *s1, char *s2);

参数　　s1：指向主题字符串的指针。
　　s2：指向对象字符串的指针。
功能说明　在主题字符串中寻找只含有对象字符串中字符的最大起始片段。
返回值　　s1 字符串中只含有 s2 所指字符串中字符的最大起始片段的长度。

5.3.76　strstr

函数原型　#include <string.h>
　　char *strstr(const char *s1, char *s2);
参数　　s1：指向主题字符串的指针。
　　s2：指向对象字符串的指针。
功能说明　在一个字符串中搜索第二个字符串出现的位置。
返回值　　成功：在 s1 字符串中首次出现 s2 字符串的位置，不包括 null；
　　失败：如果字符串未找到，返回 null。

5.3.77　strtod

函数原型　#include <stdlib.h>
　　float strtod(const char *s, char **ptr);
参数　　s：指向字符串的指针。
　　ptr：指向转换后字符的指针。
功能说明　将字符串转换成浮点数，抛弃前导空白字符。
返回值　　成功：s 所指字符串表示的浮点数，ptr 指向转换常数后第一个字符。
　　失败：0，ptr 指向第一个非空白字符。

5.3.78　strtol

函数原型　#include <stdlib.h>
　　long strtol(const char *s, char **ptr, unsigned char base);
参数　　s：指向字符串的指针。
　　ptr：指向转换后字符的指针。
　　base：基数。
功能说明　将字符串转换为长整型值，去除前导空白字符。如基数为 0，结果是十、十六、八进制整数，基数必须在 2～36 之间。字母 [a, z] 和 [A, Z] 表示值 10～35，如果基数为 16，那么十六进制整数的 0x 部分允许作为起始序列。
返回值　　成功：s 所指字符串表示的整型，ptr 指向转换常数后第一个字符。
　　失败：0，ptr 指向第一个非空白字符。

5.3.79　strtoul

函数原型　#include <stdlib.h>
　　unsigned long strtoul(const char *s, char **ptr, unsigned char base);

参数　　s：指向字符串的指针。
ptr：指向转换后字符的指针。
base：基数。

功能说明　　将字符串转换为无符号长整型值，去除前导空白字符。如基数为 0，结果是十、十六、八进制整数，基数必须在 2～36 之间。字母 [a, z] 和 [A, Z] 表示值 10～35。如果基数为 16，那么十六进制整数的 0x 部分允许作为开始序列。

返回值　　成功：s 所指字符串表示的整型，ptr 指向转换常数后第一个字符。
失败：0，ptr 指向第一个非空白字符。

5.3.80　tan

函数原型　　#include <math.h>
float tan(float x);

参数　　x：浮点弧度值，必须在 –65535 ~ +65535 之间。

功能说明　　求 x 弧度的正切值。

返回值　　x 的浮点正切值。x 值越界，产生 NaN 错。

5.3.81　tanh

函数原型　　#include <math.h>
float tanh(float x);

参数　　x：浮点弧度值。

功能说明　　求 x 弧度的双曲正切值。

返回值　　x 的浮点双曲正切值。

5.3.82　toascii

函数原型　　#include <ctype.h>
char toascii(char c);

参数　　c：字符。

功能说明　　将字符转换成 7 位 ASCII 码，即保留最低 7 位。

返回值　　c 的 7 位 ASCII 码。

5.3.83　toint

函数原型　　#include <ctype.h>
char toint(char c);

参数　　c：数字。

功能说明　　将数字 c 按十六进制转换成数值，如 c 不是十六进制数字，转换失败。

返回值　　成功：数字 c 的值。
失败：-1。

5.3.84　tolower

函数原型　#include <ctype.h>
　　char tolower(char c);
参数　c：字符。
功能说明　将字符转换为小写，如不是字母，不产生影响。
返回值　c 所表示字符的小写。

5.3.85　toupper

函数原型　#include <ctype.h>
　　char toupper(char c);
参数　c：字符。
功能说明　将字符转换为大写，如不是字母，不产生影响。
返回值　c 所表示字符的大写。

5.3.86　ungetchar

函数原型　#include <stdio.h>
　　char ungetchar(char c);
参数　c：字符。
功能说明　将字符 c 存回输入流中，使后续的 getchar 调用和其他流输入函数能返回 c，在两次 getchar 调用之间只能使用一次 ungetchar。
返回值　成功：c。
　　失败：如果在函数调用之间用了多次 ungetchar，返回 EOF。

5.3.87　va_arg

函数原型　#include <stdarg.h>
　　type va_arg(argptr, type);
参数　argptr：可选的参数列表。
　　type：下一个参数的类型。
功能说明　返回函数调用时下一参数的类型和值。用 va_start 初始化，由 type 说明变量。va_arg 使 argptr 能依次递送参数。
返回值　说明的参数类型的值。

5.3.88　va_end

函数原型　#include <stdarg.h>
　　void va_end(argptr);
参数　argptr：可选的参数列表。
功能说明　结束用 va_start 初始化的 argptr 指针的作用。
返回值　无。

5.3.89 va_start

函数原型　#include <stdarg.h>
　　void va_start(argptr, prevparm);
参数　argptr：可选的参数列表。
　　prevparm：可选参数的前一个参数。
功能说明　初始化 va_arg 和 va_end 所用的可变参数列表指针 argptr。
返回值　无。

5.3.90 vprintf

函数原型　#include <stdio.h>
　　void vprintf(const char *fmtstr, char *argptr);
参数　fmystr：格式化字符串指针。
　　argptr：参数列表指针。
功能说明　格式化字符串和数字，产生用 putchar 函数写到输出流的字符串。与 printf 类似，但是用参数列表指针而不用参数列表。
返回值　写入输出流的字符数。

5.3.91 vsprintf

函数原型　#include <stdio.h>
　　void vsprintf(char *buffer, const char *fmtstr, char *argptr);
参数　buffer：缓存指针。
　　fmtstr：格式化字符串指针。
　　argptr：参数列表指针。
功能说明　格式化字符串和数字，产生用 putchar 函数写到输出流的字符串存储在缓存中。与 sprintf 类似，但是用参数列表指针而不用参数列表。
返回值　写入输出流的字符数。

5.3.92 _chkfloat_

函数原型　#include <intrins.h>
　　unsigned char _chkfloat_(float val);
参数　val：数字。
功能说明　检查浮点数状态。
返回值　表示状态的字符。见表 5-6。

表 5-6　浮点数状态

字　符	意　义
0	标准浮点数
1	浮点数 0
2	正溢出（+INF）

（续表）

字　符	意　义
3	负溢出（-INF）
4	非数字错（NaN）

5.3.93　_crol_

函数原型　#include <intrins.h>
unsigned char _crol_(unsigned char c, unsigned char b);
参数　c：字符。
b：旋转位数。
功能说明　将字符 c 左旋 b 位。
返回值　旋转后的 c。

5.3.94　_cror_

函数原型　#include <intrins.h>
unsigned char _cror_(unsigned char c, unsigned char b);
参数　c：字符。
b：旋转位数。
功能说明　将字符 c 右旋 b 位。
返回值　旋转后的 c。

5.3.95　_getkey

函数原型　#include <stdio.h>
char _getkey(void);
参数　无。
功能说明　等待从 8051 串行通信接口读入一个字符。
返回值　接收到的字符。

5.3.96　_irol_

函数原型　#include <intrins.h>
unsigned int _irol_(unsigned int i, unsigned char b);
参数　i：整数。
b：旋转位数。
功能说明　将整数 i 左旋 b 位。
返回值　旋转后的 i。

5.3.97　_iror_

函数原型　#include <intrins.h>

unsigned int _iror_(unsigned int i, unsigned char b);

参数　　i：整数。

b：旋转位数。

功能说明　　将整数 i 右旋 b 位。

返回值　　旋转后的 i。

5.3.98 _lrol_

函数原型　　#include <intrins.h>

unsigned long _lrol_(unsigned long l, unsigned char b);

参数　　l：长整数。

b：旋转位数。

功能说明　　将长整数 l 左旋 b 位。

返回值　　旋转后的 l。

5.3.99 _lror_

函数原型　　#include <intrins.h>

unsigned long _lror_(unsigned long l, unsigned char b);

参数　　l：长整数。

b：旋转位数。

功能说明　　将长整数 l 右旋 b 位。

返回值　　旋转后的 l。

5.3.100 _nop_

函数原型　　#include <intrins.h>

void _nop_(void);

参数　　无。

功能说明　　插入 NOP 指令，用于延迟。

返回值　　无。

5.3.101 _testbit_

函数原型　　#include <intrins.h>

bit _testbit_(bit b);

参数　　b：测试和清除的位。

功能说明　　产生 JBC 指令，只能用于位寻址变量，任何形式表达式均无效。

返回值　　b。

5.3.102 _tolower

函数原型　　#include <ctype.h>

char _tolower(char c);

参数　　c：字符。

功能说明　　将字符转换为小写，用于已知是大写字母时。
返回值　　c 所表示的小写字符。

5.3.103　_toupper

函数原型　　#include <ctype.h>
　　char _toupper(char c);
参数　　c：字符。
功能说明　　将字符转换为大写，用于已知是小写字母时。
返回值　　c 所表示的大写字符。