电子报/2007 年/7 月/15 日/第 015 版
单片机应用

PIC 单片机 C 语言程序——实例 2

成都　立本

在上一讲中，笔者对 PIC 单片机 C 语言程序作了定性介绍，其目的是使读者对 C 语言有所了解。至于 C 语言的语法规则、程序设计方法、C 编译器等内容，笔者将通过更多的 C 程序实例逐一介绍。

二、C 语言的标识符和关键字

1.C 语言的标识符

在 C 语言中所用到的文件名、函数名、变量名、数组名……，都是由字符组成的。字符由英语字母和下划线组成，字符的组合称为 C 语言的标识符。标识符是由 C 程序设计者自定义的。

标识符的第一个字符通常是字母，其后可以是字母或下划线，下划线可作为标识符的分段。

不同的 C 编译器对标识符的长度有不同的要求，在编程时定义的标识符不得超过所用编译器规定的长度，以免编译时出现错误。在本文中，笔者介绍的 C 语言程序自定义的标识符不超过一个字节，在各种编译器的规范以内。上一讲的 C 程序实例 1 中，delay、main 均为 C 语言的标识符。

注意：同一字母的大小写，被视为不同的标识符，在设计 C 程序中不能混合使用。

2.C 语言的关键字

C 语言中的关键字（又称为保留字）是 C 系统定义了的特定标识符，用户自定义的 C 标识符不能与这些关键字的字符相同，否则编译不会成功。

C 语言中的关键字可分成三类：1）数据类关键字；2）程序控制类关键字；3）预处理类关键字。

C 语言的主要关键字如表 1 所示。

表1

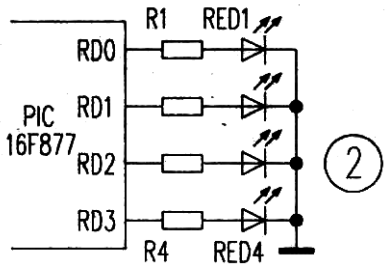
类	关键字	含　义	用　途
数据类	int	整形变量基本型	
	long	长与数据变量配合	
	short	短与数据变量配合	
	signed	有符号	数
	unsigned	无符号	据
	char	字符型(数据)	类
	const	常量	型
	float	单精度浮点(数据)	说
	double	双精度浮点(数据)	明
	struct	结构型(数据)	
	void	无类型数据	
	union	联合	
	enum	枚举	
	volatile	隐含改变(可变变量)	
	typedef	重新定义	
	register	CPU的寄存器	存储种类说明
	static	静态(变量)	存储种类说明
	extern	全局(变量)	存储种类说明
	auto	自动变量,局部变量为缺省	存储种类说明
	sizeof	字节	运算符
程序控制类	for	构成for循环结构	程序语句
	while	构成while和do while循环结构	
	if	构成if和else选择结构	
	switch	构成switch选择结构	
	return	函数返回	
	continue	转向下一次循环	
	goto	转移	
	default	Switch语句中失败选择项	
	case	Switch语句选择项	
	break	退出内层循环体	
预处理	include	包含	C程序文件
	define	定义	程序预处理定义
	⋮	⋮	⋮

说明:利用表1的关键字,结合C程序实例1中的注释,可观察出关键字在C程序中的用途,如:unsigned int i,j 代表i,j是无符号的整形变量;continue代表程序转向下一次循环;for,while循环语句……。可见,关键字是C语言的重要组成部分。

三、实例 2：PIC16F877 的 D 口（PORTD）位操作功能

C 语言程序中关键字十分重要。为使读者有较多的感性认识，笔者通过实例 2，用 C 语言（包含更多的关键字）编写 PIC 单片机任一口的位操作功能程序（实际上，在上一讲中，是用 C 语言编写 PIC 单片机任一口的字节操作功能）。

图 2 的示意电路是使 PIC16F877（用 PIC16F84A 的 B 口同样有效）的 D 口，即 RD0～RD3，外接的 LED 顺序（以秒为单位）循环点亮。如果读者有兴趣，可在以下的 C 程序中添加 RD4～RD7 外接 LED 的点亮功能。按所述的 C 语言源程序命名为 Pic02.C 其清单如下：




```
    ＃include＜pic.h＞？／／包含头文件 Pic.h
    ／／＝＝＝＝／／程序段分隔
    ＃define PORTDIT（add，bit）（(un－
signed)（＆add）＊8＋（bit)）／／位定义
    static bit PORT_0 @PORTDIT
（PORTD，0)；／／定义 PORTD 的 0 位
    static bit PORT_1 @PORTDIT
（PORTD，1)；／／定义 PORTD 的第 1 位
    static bit PORT_2 @PORTDIT
（PORTD，2)；／／定义 PORTD 的第 2 位
    static bit PORT_3 @PORTDIT
（PORTD，3)；／／定义 PORTD 的第 3 位
    void delay（）／／定义 delay 延时子函数
     {／／子函数开始
    Unsigned longint i；
／／定义 i 为无符号的整型变量
    for（i＝0；i＜＝45000；i＋＋）／／由 for 语句引
导，指定次数 1 秒延时
    continue；／／转移下次循环
    } ／／子函数结束
    ／／＝＝／／程序段分隔（可不用）
    main（）／／定义 main 主函数
     {／／主函数开始
    TRISD＝0x00；／／给 TRISD 赋值 0，设
D 口为输出
    INTCON＝0x00；／／给
INTCON 赋值 0，关中断
    PORTD＝0x00；／／给
PORTD 赋值 0，清 0D 口
    while（1）／／while 循环语句，这里是
无限循环
     {／／while 语句开始
```

PORT 0＝1；／／给 PIC16F877 的 RD0 位赋值 1，外接／／LED 亮。

delay（）；／／调延时子函数 delag

PORT 0＝1；／／给 RD1 位赋值 1，外接 LED 亮

delay（）；／／调延时子函数 delay

RD2＝1；／／RD2 位置 1，外接 LED 亮

delay（）；／／调延时子函数 delay

RD3＝1；／／RD3 位置 1，外接 LED 亮

delay（）；／／调延时子函数 delay

PORTD＝0；／／给 PORTD 赋值 0（字节），清 0D 口

delay（）；／／调延时子函数 delay

｝／／循环语句结束

｝／／主函数结束

说明：1）上述程序说明，在 C 语言程序中，要对 PIC 单片机 I／O 口的位进行操作（即某引脚），必须先用预处理关键字 define 进行定义（如程序的开始部分，见注释）之后，才能在功能函数中进行应用。应用时，口的位书写可以按定义中的书写格式，如 PIC16F877 的 D 口第 0 位，可写成 PORT_0；第 1 位写成 POTR_1……。也可直接写成 RD0、RD1、RD2、RD3……。两种方法是等效的。

2）在本连载的实例 1、实例 2 中，都有由 for 语句引导的延时语句，其书写略有差异，但都是等效的——延时 1 秒。从上述延时子函数看出，C 语言的延时比汇编语言中的延时程序更为简单。

笔者在今后的连载中，还会列举其他 C 语言的延时程序，同样十分简单。这在汇编语言是不能做到的。

3）在实例 1、实例 a 中，还涉及很多的物理量及运算符，笔者将在下一讲中作更详细的介绍，请读者关注。

电子报/2007 年/7 月/22 日/第 015 版
单片机应用

PIC 单片机 C 语言程序——实例 3

成都　立本

四、C 语言的数据类型

C 语言程序包括两方面内容：对数据的描述、对操作的描述（即算法）。所以，一些 C 语言有识之士，提出了一个公式，即

程序＝数据结构＋算法

算法是程序的灵魂，算法的对象是数据或加工对象。在今后的文章中，笔者将会逐步介绍 C 程序的各种操作语句，实际上就是 C 程序算法的具体体现，以帮助初学者学会编写 PIC 单片机 C 程序的方法。

C 语言的数据类型如下：

标准型是由 C 语言本身提供的基本数据类型，结构简单、使用频率十分高；枚举型是由用户自定义的数据类型；指针型是一种使用灵活的数据类型，在 C 语言中用于动态运算处理。这里，主要介绍 C 语言程序中的标准型数据，同样仍以 C 程序实例给予说明。

C 语言中的数据可分为整型、实型、字符型三类。每一类数据中，又有常量与变量之分。

数据在 C 语言中的重要性，笔者将其分类按表格形式列出。初学者编写 C 程序时，可将这些表格作为资料查找，也可将表格与本文中的 C 程序实例 1、实例 2 中的注释对比学习，以加深理解常量与变量的功能。

C 语言中的常量和变量 C 语言的常量是指在程序运行过程中，其值始终保持不变的量，在编写 C 语言程序时，常用关键字 const 定义。常量的类型、表示方法及示例说明如表 2 所示。

表2

	数据类型	表示方法	举例或说明
整型常量	十进制	按十进制数直接书写	123、82、5
	八进制	以字母O开头	O12、O32
	十六进制	以字母Ox开头 或以H结尾	Ox24、24H、10H (表示十进制的16)
实型常量(浮点)	十进制 小数形式	用十进制表示的 有符号(+、-)的实数	1.5、12.3、25
	指数形式	小数前面是一位非0数 称为标准型;e之前应 有数、e之后应为整数	1.234e1 (即12.34) 1234e3
字符型常量	字符-- (包括 数字)	用一对单引号括起的 字符,引号仅起界定 作用。	′A′、′8′′#′。字符是按ASC Ⅱ码值存储的，一个字 符占用一个字节，如′A′ 对应65(十进制)
字符串常量	字符串 --(包括 数字)	用一对双引号括起来 的字符,引号仅起 界定作用。	"PIC18F452"。字符串常 量,在内存中存放时,会 自动在字符串的末尾加 结束标志′\0′,即ASCⅡ 码值0字符。字符串占n+ 1个字节。

C 语言的变量是指在程序运行过程中，其值可以改变的量，其中整型变量的基本类型是 int 型。在 int 前面加上关键字 short、unsigned 则分别表示有符号（如一）和无符号数。标准 C 语言的整型数据的取值范围如表 3 所示。

表3

数据类型	位数（字节）	取值范围（定义阈）
整型(int)	16 (2)	-32768~+32767
短整型(short int)	16 (2)	-32768~+32767
长整型(long int)	32 (4)	-2147483648~+2147483647
无符号整型(unsig ned int)	16 (2)	0~65535
无符号短整型(unsigned short int)	16 (2)	0~65535
无符号长整型(unsigned long int)	32 (4)	0~4294967295
有符号整型(signed int)	16 (2)	-32768~+32767
有符号长整型(signed long int)	32 (4)	-2147483648~+2147483647
字符型(char)	8 (1)	-128~+127
无符号字符型(unsigned char)	8 (1)	0~255
有符号字符型(signed char)	8 (1)	-128~+127
浮点型(fload)	32 (4)	1.175e-38~3.4e+38
双精度型(double)	64 (8)	1.7e-308~1.7e+308

从表 3 中可以看出，此整型变量数据取值范围在 PIC 单片机汇编语言中是无法实现的。例如，用无符号长整型定义的一个变量 i（unsigned long int i）的最大值可达 10 位数，因此，只需一条有限循环指令，延时便可达 24 小时以上（见后续文章），可见，C 语言程序是十分简洁的。

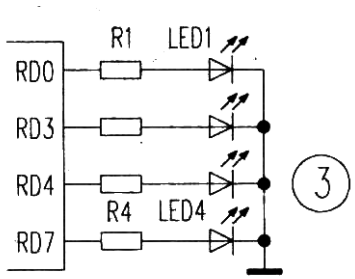


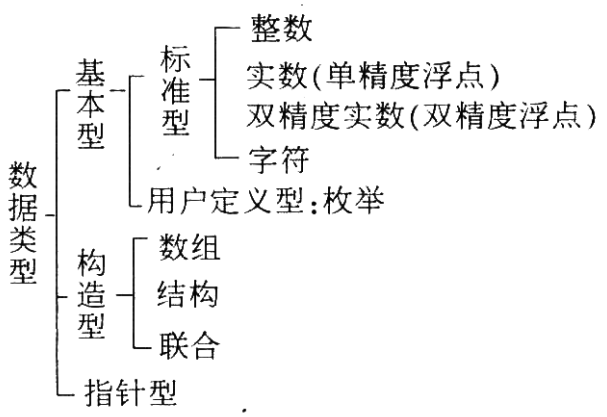
表 3 中给出的 C 语言所用到的各类型变量（变量名自定义，如 i、j、a、b 等），在编写 C 语言程序派上用途时，应加以说明或定义（这里的“ 定义 ”一词，不要与预处理类关键字 define 相混），如实例 1～3 中整型变量 i、j（见延时语句）的注释。

五、实例 3：PIC16F877 的 D 口长延时功能

鉴于在单片机中延时功能十分重要，这里以点灯（LED）为例，分别介绍 1 秒、1 分、1 小时的三种延时函数的编写实例，以供读者选用。

图 3 的示意电路在 PIC16F877（PIC16F84A 的 B 口同样有效）的 D 口 RD0、RD3、RD4、RD7 分别外接一只 LED，其 C 程序功能是电源接通后，RD0 高电平 1 秒钟点亮 LED1，之后灭，接着

RD7 高电平 1 分钟点亮 LED4，之后灭，接着 RD3、RD4 高电平 1 小时点亮 LED2、LED3，之后灭，然后又是 RD0 高电平点亮 LED0……，如此循环工作。



C 语言程序为 PIC03.C，其清单如下：

```
#include ＜pic.h＞
／／包含头文件 Pic.h
#define  PORTDIT（add，bit）((un－
signed)（&add）＊1＋（bit))
static bit PORT_0
@PORTDIT（PORTD，0）；
／／D 口定义
void delay1（）
／／定义延时子函数 de－
lay 1
{／／子函数开始
unsigned long int
i；／／字义 i 为无符号
长整型变量
for（i＝O；i＜39000；
i＋＋）／／由 for 语句引
导 1 秒的延时
continue；／／转移下次循环
} ／／子函数结束
void delay2（）
／／定义延时子函数 delay2
{／／子函数开始
unsigned long int d＝2220000；
／／定义 d 为无符号长整形，并赋值
while（－－d）
／／由 while 语句引导 1 分钟延时
{；／／while 语句开始和运行
} ／／ while 语句结束
} ／／子函数结束
```

```
    void delay3（）
    ／／定义延时子函数 delay3
     {／／子函数开始
    unsigned long int c＝133200000；
    ／／定义 C 无符号长整形，并赋值
    while（－－c）
    ／／由 while 语句引导 1 小时延时
     {；／／while 语句开始和运行
    } ／／while 语句结束
    } ／／子函数结束
    void light1（）
    ／／点灯子函数 light1
     {／／子函数开始
    RD0＝1；／／RD0 位置高电平
    delay1（）；／／调延时子函数 delay1
    RD0＝0；／／RD0 位置低电平
    } ／／子函数结束
    void light2（）／／点灯子函数 light2
     {／／
子函数开始
    PORTD＝0x80；／／
最高位置高电平
    delay2（）；／／调用延时子函数 delay2
    PORTD＝0x00／／D 口清 0
    } ／／子函数结束
    void light3（）／／点灯子函数 light3
     {／／子函数开始
    PORTD＝0x18；／／RD4、RD5 置 1
    delay3（）；／／调延时子函数
    PORTD＝0x00；／／RD4、RD5 置 0
    } ／／子函数结束
    void main（）／／定义 main 主函数
     {／／主函数开始
    TRISD＝0x00；
    ／／给 TRISD 赋值 0，设 D 口为输出
    INTCON＝0x00；
    ／／给 INTCON 赋值 0，关中断
    PORTD＝0x00；
    ／／给 PORTD 赋值 0，清 D 口
    while（1）／／无限循环
     {／／while 语句开始
    light1（）；／／调用点灯子函数 light1
    light2（）；／／调用点灯子函数 light2
    light3（）；／／调用点灯子函数 light3
```

} ／／while 语句结束
} ／／主函数结束

说明：上述 C 语言不同点灯时间的延时程序，较充分地说明了主函数 main 是如何调用各 LED 点灯子函数，以及点灯子函数如何调用各自的延时子函数（函数间相互调用）的工作过程，读者可以结合程序中注释进行查看。

电子报/2007 年/7 月/29 日/第 015 版
单片机应用

PIC 单片机 C 语言程序——实例 4

成都　立本

因 C 语言是一种表达式语句，所以只需在任意一个表达式后面加上分号“ ;”，就构成了一个表达式语句。

C 语言中的运算符是完成某种特定运算的符号，而表达式则是由运算符及运算对象所组成的具有特定功能的运算表达式。这些式子就组成 C 语言程序的各种语句。

C 语言具有十分丰富的运算符，其应用很广，除了控制语句和输入输出外，几乎所有基本操作都是作为运算符来处理。有些运算符还与汇编语言相似，所以具有汇编语言的工程师，很易转用 C 语言编程。

六、C 语言中运算符分类

按照运算符在表达式中所起的作用，可把他们分为：算术运算符、关系运算符、逻辑运算符、赋值运算符、自增与自减量运算符、指针和地址运算符、位运算符、条件运算符、逗号运算符、强制转换运算符和 Sizeof 运算符等类型。运算符按其在表达式中与运算对象的关系，又可分为单目运算符、双目运算符和三目运算符等。单目运算符只有一个运算对象如“ !”、“ 0 ”，双目运算符有两个运算对象，如以下介绍的逻辑“ 与 ”、逻辑“ 非 ”等运算。

鉴于 C 语言中运算符的重要性，笔者将部分运算符列成表格，以便读者应用时查阅（其余运算符，请读者自行查找相关资料；笔者在今后的文章中也将有所介绍）。

表 4 列出了 C 语言中的算术运算符、关系和逻辑运算符以及位运算符的功能和说明实例。

表4

类型	符号	功　能	说明或实例
算术运算符	+	相加或表正数	如4+2,结果为6。
	-	相减或表负数	如8-5,结果为3。
	※	乘法运算	如3※6,结果为18。
	/	除法运算	Y=a/b,如两个整数相除,结果取整数,小数舍去。
	%	取模运算,只求余	参加运算的均为整数,仅取余数,如5%2为1。
	++	自增量运算	操作数变量自身+1
	---	自减量运算	操作数变量自身-1
关系运算符	>	大　于	设a=4、b=5,若a>b,返回值为0;b>a返回值为1。
	>==	大于等于	设a=5、b=6,若a>b,返回值为0;b>a返回值为1。
	~	等　于	设a=6、b=7,若a==b,返回值为0(为假)。
	<	小　于	设a=7、b=8,若a<b,返回值为1(为真)。
	<=	小于等于	设a=8、b=9,若a<=b,返回值为1,b<=a,返回值为0。
	! =	不等于	设a=9、b=10,若a! =b,返回值为1(为真)。
逻辑运算符	&&	逻辑"与"	设两值不为0,反回值为1(结果为真),否则为假。
	‖	逻辑"或"	设两值不为0,反回值为0(结果为假),否则为真。
	!	逻辑"非"	一元运算符! 0=1,! 1=0。
位运算符	&	AND(按位"与")	设a =0001,b=0010,若a&b=0000。
	|	OR(按位"或")	设a =0001,b=0010,若a | b=0011。
		XOR(按位"异或")	设a =0001,b=0010,若a∧b=0011。
	~	接位取反	设a =0001,~a=1110(单操作运算符)。
	>>	位右移(等效除2)	右边移出位舍去,正数和无符号数左边补0,反之补1。
	<<	位左移(等效乘2)	左边移出位舍去,右边补0。

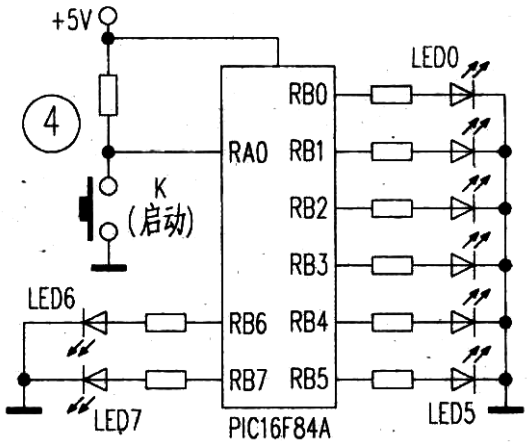
注:自增量运算符"++"和自减量运算将"--"是C语言最具特色的两种单目运算符。要求的操作对象,必须是整形变量,而不能为常量或表达式,其功能分别是使变量值加1和减1。此外,他们有两种表达方式——前缀运算符和后缀运算符,两者产生的结果是不同的,其使用方法如下:前缀运算符:++i和-i,操作i之前,使i的值加1和减1,再操作。后缀运算符:i++和i--,操作i之后,使i的值加1或减1。

举例:设y=5,则有X=++y ,运算过程是y先加1再赋值给x ,结果x=y=6。

X=y++, 运算过程是y先赋值给x,之后y+1,结果x=5,y-6

七、PIC 单片机 C 语言程序实例 4

这里用 PIC16F84A（其他 PIC 单片机同样等效）单片机，利用 C 语言中的逻辑非“ !”运算符（逻辑运算符）和按位取反“ ～”运算符（位运算符）编写一源程序，其硬件示意电路如图 4 所示，该 C 语言程序的功能是用 PORTA 的 RA0 位作启动信号（低电平有效），使 PORTB 的 RB1、RB3、RB5 和 RB7 同时点亮外接的 LED（序号 1、3、5、7）一秒钟，之后熄灭，然后由 RB0、RB2、RB4 和 RB6 同时点亮外接的 LED（序号 0、2、4、6）一秒钟，之后熄灭，然后由 RB0 位点亮外接的 LED0 一秒钟，之后熄灭，最后 RB7 位点亮外接的 LED7 一秒钟之后熄灭。熄灭之后又从上述的起始状态反复循环工作。该 C 源程序命名为 PIC04.C 其程序清单如下：




```
#include<pic.h>//包含头文件
#define PORTBIT(add,bit)((unsigned)(&add)*4+(bit))//位定义 static bit PORT_0
@PORTBIT(PORTB,0);//B 口的 0 位定义。
static bit PORT_7@PORTBIT(PORTB,7);//B 口的第 7 位定义。
#define PORTAIT(add,bit)((unsigned)(&add)*1+(bit))
static bit PORT_0 @PORTAIT(PORTA,0);//A 口的 0 位定义。
void delay() //定义函数名为 delay 的延时子函数。
{//子函数开始。
unsigned int i,j;//变量 i,j 为无符号的整形变量。
for(i=0;i<=90;i++) //由两级 for 语句引导指定次数 1 秒延时。
for(j=0;j<=1000;j++)
continue;//转移下次循环。
} //子函数结束。
main  0//定义函数名为 main 的主函数。
{//主函数开始。
TRISB=0x00;//给 TRISB 赋值 0,设 B 口为输出。
INTCON=0x00;//给 INTCON 赋值 0,关中断。
PORTB=0x00;//给 PORTB 赋值 0,清零 B 口。
while(RA0) //判 RA0 若为真(为 1),空操作循环。
{;//判 RA0 若为假,转以下 while 语句。
} //(RA0 位是按键启动信号)。
while(1) //由 while 引出的循环语句,这里是无限循环。
{//循环语句开始。
PORTB=0x55;//给 B 口赋值 B'10101010'。
PORTB=~PORTB;//B 口取反为 B'01010101'。
delay();//调用延时子函数(1 秒)。
PORTB=~PORTB;//B 口再取反为 B'10101010'。
delay();//调用延时,以便观察(1 秒)。
PORTB=0x00;//B 口清 0。
RB0=!RB0;//RB0 位取反(RB0=1)。
delay();//调用延时,以便观察(1 秒)。
RB0=0;//清零 RB0(即 RB0=0)。
```

RB7=！RB7；／／RB7 位取（RB7＝1）。

delay（）；／／调用延时，以便观察（1 秒）。

RB7＝0；／／清 0 11137（RB7＝0）。

｝／／while（1）语句再循环。

｝／／主函数结束。

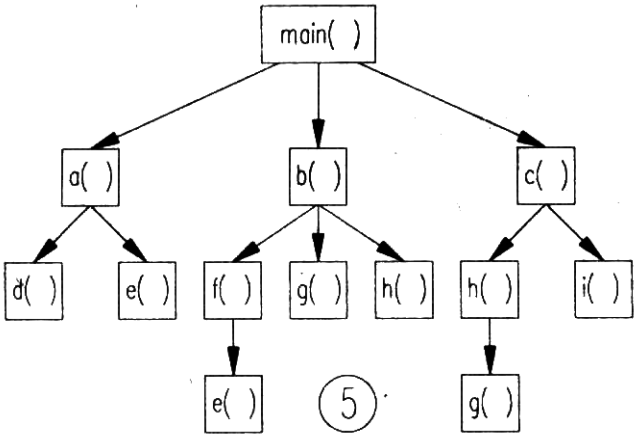
说明：上述的 PIC16F84A，图 4 电路的 LED 点亮功能的 C 程序，是采用了位运算符“～”（按位取反）和逻辑运算符“！”完成的。如果读者有兴趣，把该程序与实例（1、2、3）中的 LED 点亮 C 程序进行比较，可见上述方法所写的 C 语言程序更为简单。当然可用实例（1、2、3）赋值的方法编写图 4 功能的程序，但程序条数较多，这正好说明 C 语言的内容丰富，编程时，可操作的空间很大。

电子报/2007 年/8 月/5 日/第 015 版
单片机应用

PIC 单片机 C 语言程序——实例 5

成都　立本

前面已介绍了学习 PIC 单片机 C 语言程序的一些最基本内容，并以具有一定功能的 C 语言程序，说明其基本单位是函数，即一个 C 程序是由一个或多个函数组成的，其中只有一个主函数 main（）。主函数 main（）在程序中的位置可任意放置，但程序运行时一定是从主函数开始，再由主函数调用其他函数，其他函数也是互相调用的，如图 5 所示。



读者可以从程序实例的注释中，体会 C 语言程序的算法（即语法规则）与汇编语言指令的相似点和不同点。笔者介绍的 PIC 单片 C 语言程序，使用的时钟频率（指实验板上的晶振）都为 4MHz。

八、有参数函数

在《PIC 单片机 C 语言程序——实例 1》中已介绍：从函数的形式观察，C 语言函数可分为两种：无参数函数和有参数函数。这里再补充一点，无参数函数是指函数定义中没有形式参数，当然也没有形式参数的说明。在调用无参数函数时，主调函数没有数据传送给被调函数，其功能仅用来执行指定的一组操作，正如 C 程序实例中，主调的延时 dalay（）函数一样。无参数函数通常不返回函数值（少数可以返回）。

有参数函数的定义如下：

函数类型说明符　函数名（形式参数）

形式参数说明　语句

变量说明

其中，函数类型说明符、形式参数、形式参数说明和变量说明等都是可选项，所以在使用时应按设计功能自主选择。如果不需选用形式参数和形式参数说明，则该函数自然简化成无参数函数。

从以上的定义看出：在调用有参数函数时，在主调函数与被调函数间，存在着参数的传递，即主调函数要将给定的实际参数传给被调函数的形式参数，供给被调函数使用（这一点将在程序实例 5 中得以说明），被调函数执行后的结果，也可以带回供主函数使用。

请注意：用户为实现特定功能而编写子函数时，需按两个原则。

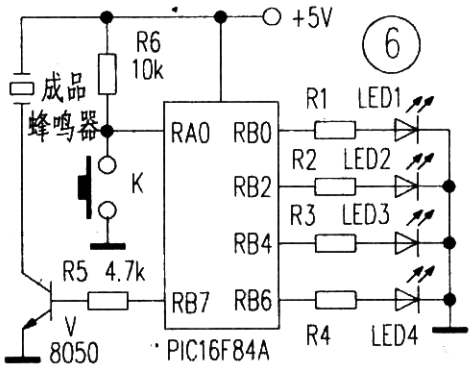
1.子函数与主调函数之间的界面应清晰，编写子函数的功能应鲜明，函数之间的数据传递越

少越好。

2.编写的子函数大小适中。若规模太大，即功能太复杂，常导致结构庞杂，影响阅读、分析和调试。一般情况下，即使要求的功能很多，函数的规模也应控制在几十行内，若不能达到要求，宁可增加新的子函数。

九、PIC 单片机 C 语言程序实例 5

这里用 PIC16F84A（其他型号的 PIC 单片机同样等效）单片机，编写一 C 语言源程序，其硬件电路见图 6。电路功能如下：用 RA0 位作程序启动信号，程序一旦启动，则 B 口的 RB0 位外接 LED 点亮一秒钟，接着 RB2 位、RB4 位、RB6 位外接的 LED 分别点亮 10 秒、1 分钟、2 分钟，循环工作。RB7 外接蜂鸣器，其用途是 RB 口的 RB0、RB2、RB4、RB6 位外接的 LED 完成给定时间的一个循环，蜂鸣器发出声响（一分钟），以便验证。



这里编写多个延时的 C 语言程序，仅用了一个带形式参数的延时子函数（delayCunsignad long int m），就完成了上述电路的功能，可见 C 语言的优点。

该 C 源程序命名为 PIC05.c，程序清单如下：

```
#include<pic.h>//包含头文件 pic.h
#define PORTBIT（add，bit）((unsigned)（&add）*5+（bit))
static bit PORT_0 @PORTBIT（PORTB，0）；
//B 口位定义
static bit PORT_2 @PORTBIT（PORTB，2）；
static bit PORT_4 @PORTBIT（PORTB，4）；
static bit PORT_6 @PORTBIT（PORTB，6）；
static bit PORT_7 @PORTBIT（PORTB，7）；
#define PORTAIT（add，bit）((unsigned)（&add）
*1+（bit))
static bit PORT_0 @PORTAIT （PORTA，0）；
//A 口位定义
void delay（unsigned long int m）
//定义有参数的延时子函数
{//dalay（延时）函数开始。
unsigned long mt i；
//无符号长整形变量 i。
for（i=0；i<=m；i++）
//带 m 参数的 for 循环。
continue；//转移下次循环。
```

```
    } ／／延时 dalay 子函数结束。
    main（）
／／定义函数名为 main（）
    ／／主函数。
     {／／主函数开始
    TRISB＝0x00；
／／初始化 PIC16F84A、B 口
    ／／A 口
    INTCON＝0x00；
    PORTB＝0x00；
    TRISA＝0x1F；
    while（RA0）／／判 RA0＝1，空
操作循环。
     {；
    } ／／RA0＝0 执行以下语句。
    while（1）／／循环语句，这是无限循环。
     {／／循环语句开始。
    RB0＝1；／／给 RBO 赋值 1（高电平）。
    delay（42000）； ／／RBO 外接 LED 点亮 1 秒钟。
    RB0＝0；／／RBO 外接 LED 灭。
    RB2＝1；／／给 RB2 赋值 1（高电平）。
    delay（490000）；／／RB2 外接 LED 点亮 10 秒钟。
    RB2＝0；／／RB2 外接 LED 灭。
    RB4＝1；／／给 RB4 赋值 1（高电平）。
    delay（2940000）；／／RB4 外接 LED 点亮 1 分钟。
    RB4＝0； ／／RB4 外接 LED 灭。
    RB6＝1；／／给 RB6 赋值 1（高电平）。
    delay（5890000）；／／RB6 外接 LED 点亮 2 分钟。
    RB6＝0；／／RB6 外接 LED 灭
    RB7＝1；／／给 RB7 赋值 1（高电平）。
    delay（3000000）；／／蜂鸣器响 1 分钟。
    RB7＝0；／／蜂鸣器停响。
    RB7＝0；
    } ／／while（1）一次循环结束。
    }
```

从以上程序中可以看出：主调函数 delay（给定的 m 值）要将给定的实际参数－42000、490000、2940000、5890000 和 3000000 等，传给被调函数 delay（unsignodlong int m）的形式参数，以决定 for 语句中的 m 值。

电子报/2007 年/9 月/2 日/第 015 版
单片机应用

PIC 单片机 C 语言程序——实例 7

成都　立本

十三、C 语言中的数组

在实例 6 中，介绍了用 PIC 单片机的输出口，外接 LED 数码管，采用 C 语言动态扫描方式的十进制计数显示。在程序中，用了 LED 数码管 7 段码组成的数组知识，这里对 C 语言的数组作一补充介绍。

数组是一种由同种类型变量组成的集合。使用这些变量时，可自定义一标识符，如用 SEG7 代表数码管的 7 段码。数组可以是一维的，也可是多维的。这里主要介绍一维数组。

1.一维数组的表达形式

类型说明符数组名［常量］

上式中的类型说明符是指变量的类型；数组名，前已所述是自定义的；方括号中的常量是指数组的元素数量，或称为下标。

例如：int x［10］；是自定义 x 整形变量，且有 10 个元素。如果要指明 x 的 10 个元素的具体内容，称为一维数码的初始化。

例如：int x［10］＝0、1、2、3、4、5、6、8、9；一维数组一旦有了表达式和初始化后，则该数组便已完整，并可进入程序。

又例如：程序实例 6 中，定义了一种带形参数的显示子函数 void display（unsigned int x），其子函数中的说明是一维数组 LED 数码管 0～9 的 7 段码；un－signed char SEG7［10］＝｛0xco，0xfq，0x04，xb0，0x99，0x92，0x82，0xf8，0x80，0x99｝；上式中的一维数组类型说明符是无符号的字符型（char）；数组名是自定义的 SEG7（7 段码）；［10］的下标 10，说明数组有 10 个元素，因该表达式已初始化，即表达式右边花括号内的 LED 数码管（共阳）的字段码为 0～9 的十个字符（这十个字符对熟悉 PIC 单片机汇编语言者是十分熟悉的字段码）。

2.一维数组部分赋值方式

一维数组可以部分赋值，如 int x［10］＝0，1，2，3，4；该一维数组只对数组的前 5 个元素 x［10］～x［4］赋值，对没有赋值的 x［5］～x［9］，默认其初始化值为 0。

3.一维数组的全赋值方式

一维数组如果对其全部元素赋值，可以省去其表达式中的元素内容即下标，例如：

int　　x［　　］＝0，1，2，3，4，5，6，7，8，9；

4.PIC 单片机口的数显位数表示法

PIC 单片机某一口，外接多位 LED 数码管增量计数时，常采用动态扫描方式。多位数码管之间的计数进位时，汇编语言和 C 语言有不同的取位（个、十、百、千……）方式。

在 C 语言中，PICC C 编译器能识别和取出 PIC 单片机某个口的个位、十位、百位、千位……值，并送某口外接 LED 数码管显示，其字符有几种表示法，这里取用最简单便于记忆的一种如下：

个位：x% 10；10 位：x／10%10；100 位；x／100%10；1000 位：x／1000%10

上式中的 x 因是自定义的，所以不同的设计者，会采用不同自定义字符。在 C 程序中，使用上述“位”字符时，对个位、10 位、100 位、1000……位还应自定义，再将上式的各位字符给自定义的每位赋值，如自定义个位为：unit－bit＝% 10；10 位为：ten－bit＝x／10% 10；100 位为：ten－bit＝x／100% 10；1000 位为：ten－bit＝x／1000%10。

以上各式应在子函数的“ 说明 ”中给予说明，请见程序实例 6 中的应用。

十四、PIC 单片机中断服务函数

单片机的“ 中断 ”是单片机一重要功能，不同型号的 PIC 单片机，有不同类型的中断源，其中 RBO 位的外部中断是最常用的中断源，也是入门学习中断方式的最佳途径。

PICC C 编译器支持 C 语言程序中直接编写中断服务函数程序。在用汇编语言编写中断服务程序时，会对堆栈出栈的保护问题难于下手，而在 C 语言程序中编写中断服务函数，是比较简单的。在 PICC C 编译器中，扩展有一关键字 interrupt（中断），该关键字 interrupt 是函数定义时的选项，利用该选项，即可定义中断服务函数（见以下实例 7）。

在编写 PIC 单片机中断服务函数时，应遵循以下原则：

1.中断服务函数不能进行参数传递，为此在编辑的中断函数中，若包含任何参数声明或传递延时，都会导致编译出错。

2.在汇编语言中，有一条中断返回指令，可使中断服务程序执行之后返回断点。在 C 语言的中断服务函数无返回标识符，乍看起来中断服务函数难于返回断点，但是，在编辑 C 程序时，经恰当安排中断服务函数，即可完成自动返回断点功能，为此，定义中断服务函数为 void 类型。

3.在任何情况下，都不能直接调用中断服务函数，否则会编译出错！对 RBO 外中断的中断服务函数，只能直接由 RBO 外接硬件开关启动中断服务函数工作，一旦服务程序执行完成即可返回断点（见以下程序实例 7）。

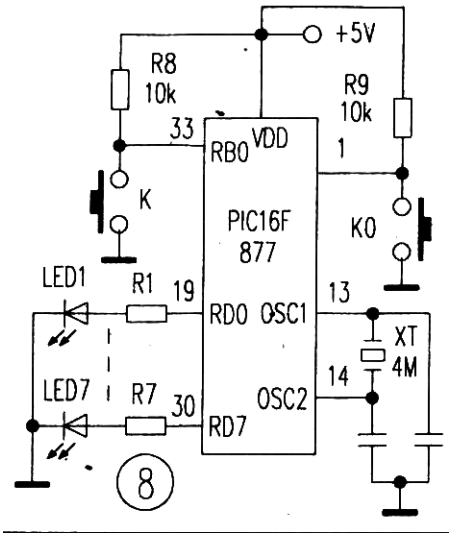
十五、PIC16F877 单片机中断程序实例 7

利用 PIC16F877 单片机，编辑－C 语言中断服务函数，硬件电路如图 7 所示，其功能是程序一旦运行，RD 口外接的 LED 闪亮约 1.6 秒，之后 RD 口低四位的 LED 又闪亮 1.6 秒，再之后 RD 口的 RD4、RD5 的 LED 又亮 1.6 秒，如此反复循环工作。一旦 RB0 外接微动开关 K 瞬时按下，即发生外中断，则程序执行中断服务——RD0 位外接 LED 点亮一定时间（该时间决定中断返回时间），即服务程序完成，立刻返回断点 LED 又按原循环方式工作。

上述的电路功能，可使读者完全理解 RB0 位外中断的全过程。为此编辑的 C 语言中断程序定义为 PIC07.C，其清单如下：

```
＃include＜pic.h＞／／包含头文件。
void delay（）／／定义 delay 为延时函数
 {
unsigned long int i；／／说明 i 为无符号长整型变量。
for（i＝0；i＜＝80000；i＋＋）／／for 引导延时约 1.6 秒。
 {；
}
}
void light1（）／／
定义点灯函数 1。
 {
PORTD＝0xAA；
／／D 口外接 LED 闪亮。
delay（）；／／LED
闪亮延时约 1.6 秒。
}
void light2（）／／
定义点灯函数 2。
 {
```

```
PORTD＝0x0F；／／D 口的外接 LED 低 4 位亮。
delay（）；／／LED 低 4 位亮延时约 1.6 秒。
}
void light3（）／／定义点灯函数 3。
 {
PORTD＝0x30；／／D 口的高位外接 LED 亮。
delay（）；／／高位 LED 亮延时约 1.6 秒。
}
void main0／／定义 main 为主函数。
 {／／主函数开始。
TRISB＝0XFF；／／设 PIC16F877 B 口为输入。
PSPMODE＝0；
TRISD＝0X00；／／设 D 口为输出。
PORTD＝0x00；／／清 0D 口。
INTCON＝0X00；／／关中断寄存器
INTEDG＝0；／／RBO／INT 下降沿触发。
RBPU＝1；／／B 口弱上拉不使能。
GIE=1；／／开总中断。
INTE=1；／／启动外部中断。
INTF＝0；／／清零中断标志位。
while（1）／／while 无限循环。
 {
lightl（）；／／调用点灯子函数 1。
light2（）；／／调用点灯子函数 2。
light3（）；／／调用点灯子函数 3。
}
}
void interrupt service（）／／定义 void 类型中断服务函数。
 {
if （INTF＝＝.1）／／判 RB 0 中断否！
 {／／中断函数开始。
INTF＝0；／／若发生了中断清除中断标志。
}
PORTD＝0X01；／／中断服务 RB0＝1 完成。
} ／／自动返回断点。
```



电子报/2007 年/9 月/23 日/第 015 版
单片机应用

PIC 单片机 C 语言程序——实例 8

成都　立本

十六、再谈 C 语言函数

鉴于 C 语言中函数的重要性，这里有必要作进一步介绍，并以实例 8 加以说明。

在 PIC 单片机的 C 语言程序设计中，常将一些电路的特定功能按模块方式编写成函数。所以，在 C 语言中，模块的功能是用函数来实现的，即使功能十分简单，如对单片机某一位取反或对某个口位的左、右移，均可用 C 语言中的位运算符编写两种子函数，供主函数调用执行。

前面已介绍过，C 函数定义的一般形式为：

类型说明符 函数名（形式参数）

形式参数说明

{

说明部分

语句

}

上述函数的类型是在定义函数时指定的，例如：

Int　num（x，y）

Char letter（c1，c2）

Float max（x，y，z）

分别指定函数类型为整型、字符型和浮点型。

函数定义中的形式参数是可选项。有形式参数的函数比无形式参数函数的执行功能强很多。如笔者已介绍的 PIC 单片机 C 语言程序中，多种延时量，通过带形式参数的延时子函数 delay（unsighedint k），就以一个延时函数 delay（k），k 是延时变量，完成多种延时的功能，使程序大大简化。

C 函数有两种：一种是标准函数，即库函数。库函数是由 C 语言系统提供的，用户无需自定义，即可直接调用；另一种是用户自定义的，即用户按需要编写的用来实现特定功能的函数，用户自定义函数符合上述 C 函数的一般定义形式。

此外，在 C 语言程序的设计中，还允许有“空函数”存在，其定义形式为：

函数类型说明符 函数名（ ）

{；

}

空函数的作用是先在程序中占好一个位置，留待以后的程序补其功能，或者用作等待功能而主函数调用空函数时，什么工作也不做。例如：PIC 单片机执行程序前，先在某口的一位上外接微动开关，当该开关按下时，程序才运行，不按时，处于等待状态，这种等待状态就是由空函数完成的。本文的实例 8 中便有空函数，参见其注释。

十七、PIC 单片机 C 语言中的运算符

C 语言提供了非常丰富的运算符，运算符的范围很广，它常把很多的基本操作作为运算符处理。C 语言中的运算符包括算术运算符（如＋、－、＊、／，加、减、乘、除）、关系运算符（如＝＝、！＝＞、＞＝－－等于、不等于、大于、大于等于）、逻辑运算符（如＆＆、‖、！－－与、或、非）、自增／自减运算符（如＋＋、－－、－－变量增 1、变量减 1）、字位运算符（如∽、《、》！

－－按位取反、左移、右移、位非)、赋值运算符（如变量＝表达式－－表达式值赋给变量)、逗号运算符（如“，”－－多个表达式的表示法）等。这里列举了 C 语言运算符的部分，其目的是希望 C 语言学习者高度重视 C 语言中的运算符在 C 语言中的重要性。事实上，在本报的 PIC 单片机 C 语言程序的实例 1～实例 8 中，上述的一些运算符已多次使用过，如赋值运算符“＝”、逗号运算符、自增、自减运算符、按位取反、位非运算符等。

在上述的运算符中，有些运算符与汇编语言相似，其运算对象与硬件指令相似，如字节位取反、位非，它们能对特定的物理地址进行访问，可见 C 语言还具有汇编语言的特点。以下将挑选几个运算符来编辑 PIC 单片机具有一定功能的 C 程序。

十八、PIC 单片机 C 语言程序实例 8

这里用 PIC16F84A，利用运算符“＞＞”右移、“＜＜”左移、“～”字节位取反、“！”位反（非）等 4 种运算符，分别编辑 5 种具有一定功能的子函数，并由主函数 main （ ）调用它们，而这些子函数又调用不同延时值的子函数，以达到预定的功能。

硬件电路功能是 16F84A 的 RAO 位一旦起动，则 B 口上 8 支 LED 全亮，然后右移 1 位、右移 2 位、右移 3 位，最后再右移一位，以验证运算符“＞＞”右移，且按指定的位数进行（后者在汇编语言中是不能实现的)。之后 B 口的 8 支 LED 全亮，又“＜＜”左移……等等。其余功能请见 C 程序实例 8 的注释。

编写该程序目的：学会 C 语言中的运算符使用方法；观察 C 语言程序中 5 种函数之间的调用过程；学会一个延时函数，实现多种延时方法。

注意：C 语言中，同一功能的程序有多种编辑方法，但仅有少数几种最便于记忆，学会最佳的 C 语言程序编写方法，会为大型程序的编写打下牢固基础。

C 程序实例 8，命名为 PICO8.C，其程序清单如下：

```
＃include  ＜pic.h＞／／包含头文件
＃define PORTAIT（add，bit）((unsigned)（＆add）＊1＋（bit))
static bit PORT_0 @PORTAIT（POR－TA，O)；／／定义 A 口的位
＃define PORTBFF（add，bit）((unsigned)（＆add）＊4＋（bit))
static  bit PORT_A  @PORTBIT（PORTB，O)；
static bit port_@PORTBFF （PORTB，7)；／／定义 B 口的位
void delay（unsigned int k）／／带形式参数的延时子函数
 {
unsigned int i，j；／／i、j 为无符号整型变量
for（i＝0；i＜＝81；i＋＋）
for （j＝0；j＜＝k；j＋＋）／／形参数 K，可改变延时量
continue；
}
void display 1（）／／定义无类型函数 display 1
 { ／／函数开始。
PORTB＞＞＝1；／／B 口右移 1 位
delay（3000)；／／原位补 0 延时约 3 秒种
PORTB＞＞＝2；／／B 口右移 2 位，原位补 0
delay（4000)；／／延时约 4 秒
PORTB＞＞＝3；／／B 口右移 3 位，原位补 0
delay（5000)；／／延时约 5 秒
PORTB＞＞＝1；／／B 口右移 1 位，原位补 0
delay（6000)；／／延时约 6 秒
```

```
PORTB＝OxFF；／／给 B 口赋值 QXFF
delay（500）；／／延时约 0.5 秒
｝／／子函数结束
void display2（）／／定义无类型函数 display2
 {
PORTB＜＜＝1；／／B 口左移 1 位，原位补 0
delay（3000）；／／延时约 3 秒
PORTB＜＜＝2；／／左移 2 位
delay（4000）；
PORTB＜＜＝3；／／左移 3 位
delay（5000）；
PORTB＜＜＝1；／／左移 1 位
delay（6000）；／／延时约 6 秒
PORTB＝0x55；／／给 B 口赋值 QX55
delay（500）；／／延时约 0.5 秒
}
void display3（）／／定义无类型函数 display3
 {
PORTB；／／B 口位取反
delay（2000）；／／延时约 2 秒
PORTB＝～PORTB；／／B 口位再取反
delay（4000）；／／延时约 4 秒
}
void display4（）／／定义无类型函数 display4
 {
PORTB＝0x00；／／ B 口清 0
delay（200）；
RBO=！ RBO；／／RBO 位取反
delay（3000）；／／延时约 3 秒
RBO=！ RBO；／／RBO 位再取反
detay（300）；／／延时约 0.3 秒
}
void display5（）／／定义无类型函数 display5
 {
PORTB＝0x00；　　／／B 口清 0
delay（200）；／／延时约 0.2 秒
RB7=！ RB7；　　／／RB7 位取反
delay（3000）；／／延时约 3 秒
RB7=！RB7；　　／／RB7 位再取反
delay（300）；／／延时约 0.3 秒
PORTB＝OxFF；／／给 B 口赋值 QXFF
delay（1000）；／／延时约 1 秒
}
void main（）／／定义主函数 main
```

```
{／／主函数开始
TRISA＝OxEF；／／pie 16F84 的口初始化
TRISB＝0x00；／／设 B 口为输出
INTCON＝0x00；　／／关中断
PORTB＝0x00；／／B 口清 0
while （RAO）／／RAO 位外接按钮作启动信号
{；／／RAO＝1 空函数，RAO＝O
}　／／启动
PORTB＝OxFF；／／给 B 口赋值 OxFF
delay（1000）；
while（1）　／／while 引导无限循环
{
display 1（ ）；／／调用子函数 display 1
display2（ ）；／／调用 display2
display3（ ）；／／调用 display3
display4（）；／／调用 display4
display5（）；／／调用 display5
}
}　／／主函数结束
```

电子报/2007 年/11 月/4 日/第 015 版
单片机应用

PIC 单片机 C 语言程序——实例 9

成都　立本

十九、C 语言中的指针

指针在 C 语言中有十分重要的地位，因为 C 语言的指针可以指向各种类型的变量，也可指向数组、指向函数、指向某种结构类型的变量……。可见 C 语言中的指针内容是十分丰富的。所以有人说：不掌握 C 语言中的指针，等于没有掌握 C 语言的精华，这句话是很有道理的。

为了说明指针的基本概念，先了解一下 C 程序中的变量在 C 编译系统中所占有的地位。一般说来，程序中的变量，经编译系统处理后，都对应着内存中的一个地址，即编译系统根据变量的类型，为该地址分配相应的内存单元，以便存放变量的内容。对不同类型的变量，所分配的内存单元的长度（即字节数）是不相同的，如字符型变量占一个字节，整型变量占两个字节……。而变量的存取是通过变量的地址进行的，这种按变量地址存取变量值的方法，称为“ 直接访问 ”（寻址）。

在 C 语言中，除了定义整型、字符型等变量外，还定义了另外一种类型的变量，该变量专用于存放其他变量在内存中所分配的存储单元的首地址。

假设 fp 变量是用于存放字符型（char）变量 M 所占用的存储单元的首地址，再假设已用某种方式将字符型（char）变量 M 所占用的内存单元的首地址赋值给了 fp 变量，那么要想通过变量 fp 取得字符变量 M 中的内容，可按照如下两个步骤操作：

1.根据变量 M 所占用内存单元的首地址，读取其中的数据，该数据就是字符变量 M 所占用的内存单元的首地址。

2.根据第一步读取的地址及字符型变量 M 所占用的存储单元的长度（字节数），读取字符变量 M 的值（内容）。

上述存取 M 变量的方式称为“ 间接访问 ”（寻址）方式。

在 C 语言中，借助于指针这一概念，能方便地达到间接访问（寻址）的目的。所谓指针，就是某个对象（如变量、数组和函数等）所占用存储单元的首地址。这时定义：专门用来存放某种类型变量的首地址的变量为该种类型的指针变量，而其首地址则为指针值。

通过上述关于 C 语言指针的概念的描述，可以看出：C 语言中的指针与 PIC 单片机汇编语言中的间址寄存器 FSR 和 INDF 十分类似，FSR 存放寄存器地址，INDF 存放与 FSR 地址对应的寄存器的内容。然而，C 语言指针变量的功能远比 FSR、 INDF 强得多。

二十、指针变量的定义和操作

1.指针变量的定义

数据类型说明符 ＊指针变量名 1，＊指针变量名 2。

其中，类型说明符，表示该指针变量所指向的变量类型［如整型（int）、字符型（char）、单精度浮点型、数组……」;“ ＊ ”，表示定义指针变量（自定义名）。指针变量名前面的“ ＊ ”，只指明该变量为指针变量，而指针变量名中并不包含“ ＊ ”。这一点与以前所介绍的定义变量是不相同的。再有：一个指针变量只能指向同一种类型的变量。

2.与指针有关的两个运算符

＆：取地址运算符

＊：指针运算符

例如：int　a，／／定义整型变量 a。

int ＊Pl；／／定义指向整型变量的指针变量 P1

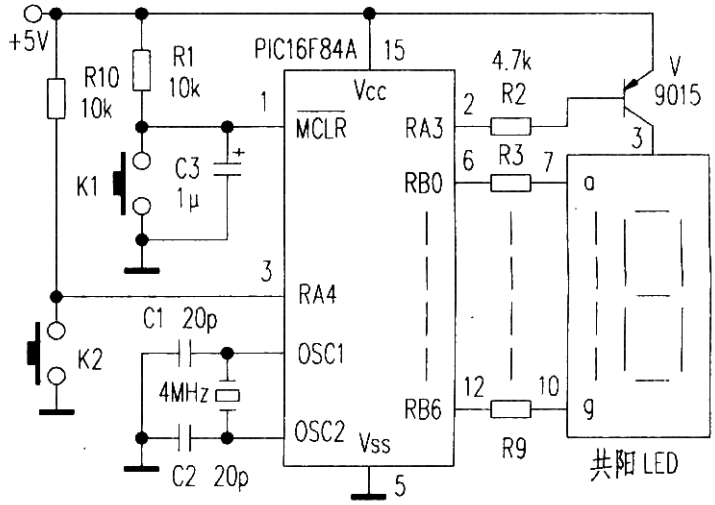
P1＝＆ a；／／变量 a 的地址，赋值给 P1

＊P1 表示 P1 指向的变量，即变量 a（即内容）

这样就在指针和它对应的变量之间建立了对应的关系。

二十一、实例 9

这里用 PIC16F84A 编辑－C 语言有关指针的 C 程序,其硬件电路如附图所示。电路功能如下：接上＋5V 电源，按动微动开关 K2，电路启动，数码管 LED 从 0～9 显示，显示完成后，接着显示 5、 7、 2 的指定三个数，然后循环工作。该 C 程序旨在用指针显示 0～9 的数，另外用指针显示数码管指定的数，是指针指向数组的应用实例。该 C 程序名为 PIC09.C，其清单如下：



＃include ＜pic.h＞／／包含头文件。

unsigned char SEG7 [10] ＝ {0xc0，0xf9，0xa4，0xb0，

0x99，0x92，0x82，0xf8，0x80，0x90}；

＃define PORTAIT（add，bit）((unsigned)（＆add）＆5＋（bit))

static bit PORT__3 ＠PORTAFF（POR－TA，3）；

／／PORTA 的位定义

static bit PORT__4 ＠PORTAIT（POR－TA，4）；

unsigned char ＊P，k＝0；

／／定义无符号字符型指针 P

void delay（unsigned long int M）

／／延时函数带形参数 M

{

unsigned long int i；

／／无符号长整形变量 1

for（i＝0；i＜＝M；i＋＋）

／／改变 M 可改变延时量

{；}

}

void main（）／／主函数 main（）

{ ／／主函数开始

```
TRISB＝0x00；
／／令 PIC16F84A B 口为输出
PORTB＝0xc0；
／／B 口 LED 数码管显示 0
TRISA＝0x10；
／／ A 口 RA4 为输入，其余位为输出
PORTA＝0x1F；
／／关 B 口指定的数码管 LED
while（RA4）／／判 RA4 位为 0 否（微动开关实现）
 {；／／若为 0，程序启动，执行以下程序
}
RA3＝0；／／开 B 口 LED 数码管的指定位
while（1）
／／while 引导无限循环
 {
P＝＆SEG7［0］；
／／ LED 数码管字符段起始地址赋值给指针 P
while（k＜＝9）
／／while 循环限制为 9 次，以便显示 0～9 数
 {
PORTB＝＊P；
／／指针起始地址内容送 B 口
delay（100000）；
／／延时约 3 秒
PORTB＝0xc0；
／／LED 数码管显示 0
delay（100）；  ／／短暂延时
P＋＋；  ／／指针 P＋＋
k＋＋；          ／／K＋＋
}
P＝＆SEG7［5］；
PORTB＝＊P；
delay（65000）；
PORTB＝0xc0；
delay（100）；
P＝＆SEG7［8］；
PORTB＝＊P；
delay（40000）；
PORTB＝0xc0；
delay（100）；
P＝＆SEG7［2］；
PORTB＝＊P；
delay（130000）；
PORTB＝0xc0；
```

```
delay（100）；
k＝0；
}
}
k＝0；        ／／K＝＝9 时，给 K 赋值 0
}
} ／／主程序结束
```

电子报/2007 年/11 月/11 日/第 015 版
单片机应用

PIC 单片机 C 语言程序——实例 10

成都　立本

学习 PIC 单片机 C 语言程序，应当具有汇编语言的基础，如果没有汇编语言知识，就很难直接进入 PIC 单片机 C 语言程序的学习。下面的例子，充分说明以上的相关结论。

二十二、PIG 单片机 A／D 转换模块

在电子技术中，常需把模拟量转换成数字量，这种转换称为 A／D 转换。PIC16F87×系列单片机内集成有 A／D 转换部件，常称为 A／D 转换模块。在芯片内部具有多个 A／D 转换输入通道，如 PIC16F873 有 5 个 A／D 输入通道，16F877 有 8 个 A／D 输入通道。

要利用 PIC 单片机的 A／D 转换模块进行给定的模拟－数字变换，无论用汇编语言，还是 C 语言编辑程序，都需要用到 PIC 单片机中管理 A／D 转换模块的硬件资源，即专用寄存器，包括：

1.A／D 控制寄存器 AD-CON0，其作用是控制 MD 转换器的操作；2.A／D 控制寄存器 ADCON1，其作用是选择 A／D 引脚的功能；3.A／D 结果高字节寄存器 ADRESH，用于存放 A／D 转换结果的高字节：4.A／D 结果低字节寄存器 ADRESL，用于存放 A／D 转换结果的低字节；此外，还会用到常用的外中断接口使能寄存器 PIE1 和 IPIE2。

当 A／D 转换完成后，10bit(位)A／D 转换结果分别存放在 ADRESH(高字节)和 ADRESL(低字节)中，存放方式由 AD-CON1 特定位(ADFM)的设置而定。

以上 A／D 转换专用寄存器，都是用位功能对 A／D 转换进行管理，读者可查看相关书籍。下面以 ADCON0 寄存器为例，说明它们位的管理功能。

ADCON0 寄存器的位功能：

ADCS1：ADCS0 为 A／D 转换时钟选择位，定义如下：00=fosc／2，01=fosc／8，10=fosc／32，11=fosc(用 RC 振荡器驱动的时钟)。

CHS2：CHS1：CHS0 为 A／D 模拟通道选择位。0 0 0 选择通道 0(RA0／AN0)；0 0 1 选择通道 1(RA1／AN1)；0 10 选择通道 2(RA2／AN2)；0 1 1 选择通道 3(RA3／AN3)；……。

GO／DONE 为 A／D 转换状态位。当 ADON=1 时，GO／DONE=1 启动转换：GO／DONE=0 未进行 A／D 转换(若 A／D 转换已完成，该位自动清 0)。

bit 1 未使用，读作 0。

ADON 为 A／D 转换允许位。ADON=1 打开 A／D 转换，ADON=0 关闭 A／D 转换。

其余的 A／D 转换专用寄存器的位功能，可按 ADCON0 的位功能进行。

二十三、A／D 转换操作步骤

对于 PIC 单片机 A／D 转换的操作，汇编语言与 C 语言程序十分相似。其区别是，汇编语言利用指令，按 A／D 转换专用寄存器的位功能编辑汇编语言程序；C 语言利用 C 的语法规则，按 A／D 转换专用寄存器的位功能编辑 C 语言程序。所以具有 PIC 单片机汇编语言的基础的工程师。很易进入 C 语言环境编辑 C 程序。它们共同的操作步骤如下：

1.设置 A／D 转换模块

(1)对模拟引脚／数字 I／O(用 ADCON1 位功能完成)进行设置；(2)选择 A／D 输入通道、转换时钟和打开 A／D 转换模块(用 ADCON0 位功能完成)。

2.如需要 A／D 中断功能，例如利用中断把 A／D 结果送 D 口显示，此时设置中断子函数完成。

(1)A／D 转换完成标志位 ADIF 清 0(即 ADIF=0)；(2)对 A／D 转换中断允许位 ADIE 置 1；(3)

对全局中断允许位 GIE 置 1。

3.等待模拟量采样完成。

4.启动 A／D 转换，即 G0／DONE=1(或 ADG0=1)。

5.等待 A／D 转换完成——可用软件查询或 A／D 中断完成。

6.读 A／D 结果寄存器 ADRESH 和 ADRESL。

7.如果需要进行下一次 A／D 转换，转入步骤 1 或 2。A／D 转换所需时间定义为 TAD。

下面介绍的 PIC16F877 单片机 A／D 转换 C 语言程序，正是按上述步骤编辑的。

二十四、C 语言中的联合

C 语言中的联合(又称共用体)是将不同的数据项组织成一整体，其特点是：它们在内存中占用同一的内存区间，所以称为共用体，所占内存的长度等于最长的成员长度。这样几种不同变量共同占用相同存储区间的“联合体”类型结构可以提高内存的利用效率。

联合体函数定义为：

union 联合体名

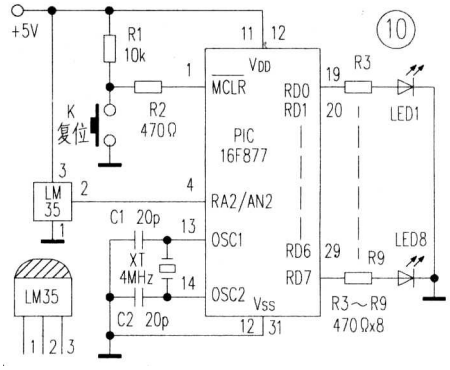
{成员列表

}变量列表

以下的 PIC16F877A／D 转换的 C 程序是利用了联合体而编辑的。

二十五、实例 10

硬件电路如图 10 所示。在 RA2 上接有温度传感器 LM35，A／D 转换目的是将 LM35 输出的模拟量转换成数字量。并用 D 口外接的 LED 显示出来。



该 C 程序是实验性质的，主要说明编辑 A／D 转换 C 程序的方法。实际上，对于 A／D 转换后的数字还需作进一步处理(如数字显示)，才能达到实际应用。

该 A，D 转换 C 程序名为 PIC10.C，其清单如下：

```
#include<pic.h>／／包含头文件
union adres／／联合体
{intvy1；／／成员列表：整型 Y1
unsigned char adre[2]；／／无符号字符数组 adre[2]
}adresult；／／变量列表
unsigned int j；／／无符号整形变量 j
void initial()／／I／O 初始化子函数
{  ／／函数开始
TRISD=0x00；／／设 D 口为输出
}  ／／子函数结束
void adinitial()／／A／D 转换初始化子函数
{  ／／函数开始
```

```
ADCON0=0x51；／／选择 A／D 通道 RA2
／／打开 A／D 转换，转换时钟为 8tosc
ADCON1=0x80；／／A／D 转换右移，ADRESH 高 6 位为 0
／／并把 RA2 位设为模拟输入方式
PIE1=Ox00：
PIE2=0x00；
ADIE=1；／／A／D 转换允许中断
PEIE=1；／／允许外围中断
TRISA2=1；／／设置 RA2 为输入
}
void delav()／／延时子函数
{
for(j=20000；－－j；)
continue；
}
void interrupt adint(void)
／／定义中断子函数
{
ADIF=O；／／清中断标志位
adresult.adre[O]=ADRESL；／／读取并存储 A／D 结果
adresult.adre[1]=ADRESH；／／通过联合体形式存放在变量 Y1 中
PORTD=ADRESL；／／A／D 转换低字节送 D 口
delay()；／／延时，以便观察
ADG0=1；／／启动下一次 A／D 转换
}
main()／／主函数
{
adinitial()；／／调 A／D 转换初始化函数
initi8l()；／／调 I／O 口初始化函数
ei()；／／允许总中断
ADG0=1；／／启动 A／D 转换
while(1)／／由 while 引导空操作
{；／／以等待 A／D 转换完成中断发生，执行中断程序
}
```

注意：因室温条件下，低 8 位已足够用了，所以在上述 C 程序中，LM35 的 A／D 转换时的高两位(在 ADREH 中)未送 PORTD 显示。