嵌入式系统C语言

《C语言程序设计》 高等教育出版社

《嵌入式系统高级C语言编程》北京航天航空大学出版社 （电子版）

C语言知识附录

关键词(32个：数据类型，控制语句，存储类型，其它)

关键字		意义
	char	声明字符型变量或函数
数	double	声明双精度变量或函数
据	enum	声明枚举类型
类	float	声明浮点型变量或函数
型	int	声明整型变量或函数
	long	声明长整型变量或函数
	short	声明短整型变量或函数
	signed	声明有符号类型变量或函数
	struct	声明结构体变量或函数
	union	声明共用体（联合）数据类型
	unsigned	声明无符号类型变量或函数
	void	声明函数无返回值或无参数，声明空类型指针

结构体和共用体的区别在于：结构体的各个成员会占用不同的内存，互相之间没有影响；而共用体的所有成员占用同一段内存，修改一个成员会影响其余所有成员。

关键字		意义
	for	循环语句
控	do	循环语句的循环体
制	while	循环语句的循环条件
语	break	跳出当前循环
句	continue	结束当前循环，开始下一轮循环
	if	条件语句
	else	条件语句否定分支（与if连用）
	goto	无条件跳转语句
	switch	开关语句
	case	开关语句分支
	default	开关语句中的其它分支
	return	返回语句

退出

关键字		意义
存	auto	声明自动变量，缺省时一般为“auto”
储	extern	声明变量是在其它文件中声明（也可以看做是引用变量）
类	register	声明寄存器变量
型	static	声明静态变量
	const	声明只读变量
其	sizeof	计算对象所占内存空间大小
它	typedef	给数据类型取别名
	volatile	说明变量在程序执行中

- 1. const
- 声明只读的变量，其值在编译时不能被使用，声明时必须为只读变量赋一个初始值。例如：const float Pi = 3.14;
- 为了防止函数的参数在函数体中被意外改变，用const修饰函数参数，如：
- void Fun(const int I); //const修饰参数，防止其在函数体中被意外改变。

- 2. typedef:给一个已经存在的数据类型取一个别名，定义一个类型的别名，而非定义一个新的数据类型。
- [格式] typedef 类型名 定义名

- （1）基本类型的自定义
- typedef float REAL ； //定义单精度型为REAL
- typedef char CHARACTER ； //定义字符型为CHARACTER
- typedef unsigned long INT32U ； //定义32位无符号整型为INT32U
- typedef unsigned int INT16U ； //定义16位无符号整型为INT16U
- typedef unsigned char INT8U ； //定义8位无符号整型为INT8U

则很明显的看出：uint8_t是用1个字节表示的；uint16_t是用2个字节表示的；uint32_t是用4个字节表示的。 比如：

```
typedef signed char             int8_t;
typedef short int               int16_t;
typedef int                     int32_t;

typedef unsigned char           uint8_t;
typedef unsigned short int      uint16_t;
typedef unsigned int            uint32_t;
```

- （2）结构体类型的自定义

- typedef struct
- {
- long num;
- char name[10];
- char sex;
- }STUDENT　　；//定义结构体类型为STUDENT

- int main()
- {
- STUDENT stu1, stu[10]　；//定义STUDENT类型的变量stu1和数组stu
- }

- 3. volatile: 用于描述一个对应于内存映射的IO端口或者硬件寄存器。用volatile关键字修饰变量，使得每次都重新读取这个变量的值，而不是使用保存在寄存器里的备份。
- volatile int flag = 0;
- void f()
- {
- while(1)
- {
- if(flag)
- {
- do_action();
- }
- }
- }
- void isr_f()
- {
- flag = 1;
- }

2. 运算符: 算术运算符(假设a=8，b=2)

运算符	说明	范例	执行结果
+	加	c=a+b；	c 等于10
—	减	d=a-b；	d 等于6
*	乘	e=a*b；	e 等于16
/	除	f=a/b；	f 等于4
%	取余数	g=a%b；	g 等于0
++	加1	c++；相当于c=c+1；	c 等于11
--	减1	d--；相当于d=d—1；	d 等于5
=	等于	a=8；	设置a等于8
+=	先相加在等于	e+=5；相当于e=e+5；	e 等于21
-=	先相减在等于	f-=5；相当于f=f-5；	f 等于-1
*=	先相乘在等于	b*=5；相当于b=b*5；	b 等于0
/=	先相除在等于	a/=5；相当于a=a/5；	a 等于1
%=	先取余数在等于	a%=5；相当于a=a%5；	a 等于3

比较运算符(假设a=8)

运算符	说明	范例	执行结果
==	等于	a==5	F
！=	不等于	a!=5	T
<	小于	a<5	F
>	大于	a>5	T
<=	小于等于	a<=5	F
>=	大于等于	a>=5	T

逻辑运算符(假设a=8)

运算符	说明	范例	执行结果
&&	AND	(a>5)&&(a<10)	T
||	OR	(a<5)||(a>10)	F
!	NOT	!(a>10)	T

位逻辑运算符(假设a=8)

运算符	说明	范例	执行结果
&	AND	a=a&0x01	a等于0x08
|	OR	a=a|0x80	a等于0x08
~	NOT	a=~a	a等于0xF7
^	XOR	a=a^0xFF	a等于0xF7
<<	左移	a=a<<1	a等于0x10
>>	右移	a=a>>1	a等于0x04

3. 预处理

预处理命令	意义
#define	宏定义
#undef	撤销已定义过的宏名
#include	使编译程序将另一源文件嵌入到带有#include的源文件中
#if	#if 的一般含义是：如果#if 后面的常量表达式为true，则编译它与#endif之间的代码，否则跳过这些代码。
#else	#endif 标识一个#if 块的结束。
#elif	#else 在# if 失败的情况下建立另一选择。
#endif	#elif 命令意义与else if 相同，它形成一个if else-if 阶梯状语句，可进行多种编译选择。
#ifdef #ifndef	用#ifdef 与#ifndef 命令分别表示“如果有定义”及“如果无定义”，是条件编译的另一种方法
#line	改变当前行数和文件名称，它们是在编译程序中预先定义的标识符，命令的基本形式为：#line number["filename"]
#error	编译程序时，#error 就会生成一个编译错误提示消息，并停止编译
#pragma	它允许向编译程序传送各种指令，当编译器遇到这条指令时就在编译输

- (1) 简单的宏定义:
- 如： #define PI 3.1415926
- 将PI定义为3.1415926，以后的程序中遇到PI都将用3.1415926替换。

- 注意：在简单宏定义的使用中，当替换文本所表示的字符串为一个表达式时，容易引起误用。

- (2) 带参数的宏定义
- #define <宏名> (<参数表>) <宏体>
- 如： #define A(x) x
- 将A(x)定义为x，程序中A(x)将被替换为x，如A(5)将被替换为5。

- 在带参数的宏定义的使用中，如果缺少括号，极易引起误解。例如我们需要做个宏替换能求任何数的平方，这就需要使用参数，以便在程序中用实际参数来替换宏定义中的参数。

- 2). 条件编译

- (1) #if / #endif/ #else/ #elif指令
- #include <stdio.h>
- #define DEBUG　　//此时#ifdef DEBUG为真
- //#define DEBUG 0 //此时为假
- int main()
- {
- #ifdef DEBUG
- printf("Debugging\n");
- #else
- printf("Not debugging\n");
- #endif
- printf("Running\n");
- system("pause");
- return 0;
- }

- #include <stdio.h>
- #define TWO
- int main()
- {
- #ifdef ONE
- printf("1\n");
- #elif defined TWO
- printf("2\n");
- #else
- printf("3\n");
- #endif
- system("pause");
- return 0;
- }

- 3). 文件包含#include：将已存在文件的内容嵌入到当前文件中。
- (1)#include <filename>
- 其中 filename 为要包含的文件名称，尖括号括起来，也称为头文件。表示预处理到系统规定的路径中去获得这个文件（即C编译系统所提供的并存放在指定的子目录下的头文件）。找到文件后，用文件内容替换该语句。
- (2)#include “filename”
- 其中 filename 为要包含的文件名称，双引号表示预处理在当前目录中查找文件名为 filename 的文件，若没有找到，则按系统指定的路径信息，搜索其它目录。找到文件后，用文件内容替换该语句。
- 注意：#include支持相对路径，.代表当前目录，..代表上层目录，以此类推，如#include “..filename”表示在上层目录中查找名为filename的文件。

- 4). 指针

指针本质就是一个变量,而这个变量永远只能存储一个内存地址(编号)所以此变量对应的专业术语叫指针变量，通过指针变量保存的地址就可以对这块内存区域任意访问(读查看,写修改)，而指针指向的内存区域可以保存一个数字,而这个数字有数据类型

- 4). 指针
- (1) 指向数组元素的指针
- 定义一个整型数组和一个指向整型的指针变量:
- int a[10];
- int *p = NULL; //定义指针时要初始化
- p = a; //数组名a为数组第0个元素的地址,
- //因此p指针指向数组a中第0个元素
- //与 p = &a[0] 等价
- 根据地址运算规则，a+1 表示 a[1] 的地址，a+i 为 a[i] 的地址，则可以用指针表示数组元素的地址和内容:
- (a) p+i和a+i 均表示a[i]的地址，即指向数组第i号元素a[i]。
- (b) *(p+i)和*(a+i)都表示 p+i 和 a+i 所指对象的内容，即为a[i]。

- (2)字符指针
- 字符串常量是由双引号括起来的字符序列，如 “Hello World!”，存储字符串常量的方法有两种：
- (a) 把字符串常量存放在一个字符数组中，
- 如：char s[] = “Hello World!”
- (b) 用字符指针指向字符串，
- 如：char *cp =“Hello World!”
- 然后通过指针访问字符串的存储区；但是不可以通过指针来修改字符串常量，如：将“H”改为“A”，*cp=“A”是错误的。
- 字符指针cp只占用一个可以存放地址的内存单元，存储字符串字符的地址，而不是将字符串放到字符指针变量中

- (3)指针函数
- 指针函数一定有“函数返回值”，而且，在主调函数中，函数返回值必须赋给同类型的指针变量。指针函数的定义格式为：
- 类型名 *函数名(函数参数列表);
- 其中，后缀运算符“()”表示是一个函数，其前缀运算符“*”表示此函数为指针型函数，其函数值为指针，即它带回来的值的类型为指针。当调用这个函数后，将得到一个指针（地址），类型名表示函数返回的指针指向的类型。例如：
- int *pfun(int,int);
- 由于“*”的优先级低于“()”的优先级，因而pfun首先和后面的“()”结合，说明pfun是一个函数。接着再和前面的“*”结合，说明这个函数的返回值是一个指针。
- 最后，前面的int表明pfun是一个返回值为整型指针的函数。返回类型可以是任何基本类型和复合类型。

https://wenku.baidu.com/view/8e2f0bccbbf67c1cfad6195f312b3169a451eab9.html

- 5). 结构体
- (1). 结构体说明和结构变量定义

结构体是由基本数据类型构成的，并用一个标识符来命名的各种变量的组合。由于结构体也是一种数据类型，因此在使用结构体时要先定义，其定义格式一般为：

struct 结构名

{

- 类型 变量名;
- 类型 变量名;
- ...

} 结构变量;

构成结构的每一个类型变量称为结构成员，它象数组的元素一样，但数组中元素是以下标来访问的，而结构是按变量名字来访问成员的。

还可以用已说明的结构名定义结构变量。如:

```
struct string
{
char name[8];
int age;
char sex[4];
char depart[20];
float wage1,wage2,wage3;
  };
 struct string person;
```

如果需要定义多个具有相同形式的结构变量时用这种方法比较方便，它先作结构说明，再用结构名来定义变量。如:

```
struct string Tianyr, Liuqi, ...;
```

- (2). 结构体变量的使用：结构体成员的表示方式为：

结构变量.成员名

```c
struct students
{
    char name[50];
    char sex[50];
    int age;
    float score;
};

int main(void)
 {
    struct students student;
    printf("Name: %s\t",student.name[0]);
    printf("Sex: %s\t", student.sex);
    printf("Age: %d\t", student.age);
    printf("Score: %f\r\n", student.score);
    return 0;
 }
```

https://blog.csdn.net/XYF736162986/article/details/122602983

作业

复习和学习C语言

重点Typedef，struct，预处理，指针等

写一页word，提交到“学在浙大”

《C语言程序设计》 高等教育出版社

《嵌入式系统高级C语言编程》北京航天航空大学出版社 （电子版）