RISC C 程序使用注意事项

声明：

RISC C 使用请严格按照下述注意事项和方法操作，否则可能会导致程序运行混乱及无法预料的情况发生，若排除以下情况后编译或程序运行仍有异常请及时与我们技术支持取得联系。

1. 操作特别注意

1.1 C 相关文件命名规则

“.h”、“.c”文件的文件名必须由字母或数字或下划线组成

1.2. 变量定义及代码写法

1.2.1 编译器默认占用前七个寄存器地址，请勿重复定义
1.2.2 局部变量能正常使用，但是目前还不能查看变量的值，建议使用全局变量进行操作，考虑到编译器的优化问题，建议声明变量时，变量前面加”volatile”修饰符，这样所有对变量的操作都不会被编译器优化掉.
1.2.3 代码输入，同一行建议不要写多行代码

1.3 特殊寄存器操作

操作 SFR、SFR0、SFR1、INDF、INDF0、INDF1、INDF2、HIBYTE、PCL 等，操作数据指针寄存器，程序指针计数器，间接寻址寄存器，字操作高字节缓冲器时，需要关闭中断，否则可能导致程序运行异常。

1.4 乘除法/指针/结构体等操作

编译器对乘数法/指针/结构体支持不友好，程序里尽量不用乘法、除法、指针、结构体等，可用移位、加法、减法等来代替。

1.5 函数定义及调用

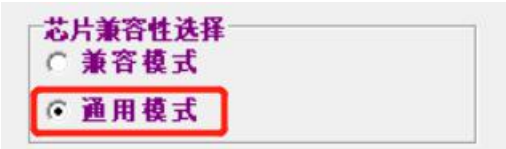
函数带参调用及返回值不能使用单字节以上的参数，请使用单字节（unsigned char，char）参数传递或用全局变量。

1.6 中断返回

中断函数中不能使用“return”等返回语句。

1.7 通用兼容模式选择

若芯片带通用和兼容模式选择，请务必在编译和烧录时保持选项一致。建议统一使用通用模式，除非有特殊需求。



2. 部分特殊操作方法

2.1 嵌入汇编代码的方法

嵌入汇编代码需要在函数内部完成；需要注意的是，调用汇编代码使用到寄存器时，需加“_”前缀，如“_TOCR”，如下调用 ASM 方法：

```
Void test(void)
{
__asm
        MOVAR _TOCR
__endasm;

}
```

2.2 位定义的方法

位定义的方法参考 C 语言位域操作，如下：

```
Typedef union
{
    unsigned char byte;
    struct
    {
        unsigned char bit0 : 1
        unsigned char bit1 : 1
        unsigned char bit2 : 1
        unsigned char bit3 : 1
        unsigned char bit4 : 1
        unsigned char bit5 : 1
        unsigned char bit6 : 1
        unsigned char bit7 : 1
    }bits;
}Flag;
volatile Flag Flag1;
#define Flag_Timer10Ms      Flag1.bits.bit0
```

2.3 指定地址变量定义方法

指定地址变量定义方法，如下：

指定 ram 地址：

```
__sfr __at 0x17 a0;

typedef union {
  struct {
       unsigned char a00:1;
       unsigned char a01:1;
       unsigned char a02:1;
       unsigned char a03:1;
       unsigned char a04:1;
       unsigned char a05:1;
       unsigned char a06:1;
       unsigned char a07:1;
  };
} __a0bits_t;
volatile __a0bits_t __at 0x17 a0bits;
#define a00       a0bits.a00      /* bit 0 */
#define a01       a0bits.a01      /* bit 1 */
#define a02       a0bits.a02      /* bit 2 */
#define a03       a0bits.a03      /* bit 3 */
#define a04       a0bits.a04      /* bit 4 */
#define a05       a0bits.a05      /* bit 5 */
#define a06       a0bits.a06      /* bit 6 */
#define a07       a0bits.a07      /* bit 7 */
```

也可以用 volatile unsigned char __at 0x17 a0;方法进行定义

2.4 指定 rom 地址的方法

```
const unsigned char __at 0x17 a0;
```

2.5 中断函数定义方法

中断函数定义(需要用户完成压栈、出栈)，下面例子是无压栈、出栈指令，利用变量保存现场；

```
volatile unsigned char ABuf;
volatile unsigned char StatusBuf;
void int_isr(void) __interrupt
{
```

```
__asm
movra   _ABuf    //__asm ….   __endasm 为中断保护现场固定写法，请勿修改。
swapar  _STATUS //STATUS 为状态字寄存器，根据规格书寄存器描述来写，有些是 PFLAG
movra   _StatusBuf
__endasm;
……
code
……
__asm
swapar  _StatusBuf//__asm ….   __endasm 为中断保护现场固定写法，请勿修改。
movra   _STATUS  //STATUS 为状态字寄存器，根据规格书寄存器描述来写，有些是 PFLAG
swapr   _ABuf
swapar  _ABuf
__endasm;
}
```

请将 ABuf、StatusBuf 定义成全局变量；另外，在中断函数中尽量使用不要过多层的调用函数，以防止堆栈溢出，尽量使用全局变量或者定义的位来操作，不能使用“return”等返回语句。

2.6 Ram 使用情况查看方法

Ram 使用情况查看已在输出信息框中列出，格式为：

```
RAM USAGE MAP ('X' = Used,  '-' = Unused)
0000 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXX----- ----------------
```

3. 数据类型描述

Type	Size(byte)	Scope
char	1	-128 ~ 127
unsigned char	1	0 ~ 255
short	2	-32768~32767
unsigned short	2	0 ~ 65535
int	2	-32768~32767
unsigned int	2	0 ~ 65535
long	4	-2147483648 ~ 2147483647
unsigned long	4	0 ~ 4294967295
long long	4	/
unsigned long long	4	/
float	4	/
double	4	/
long double	4	/