1 单片机 C51 编程规范－ 前言

为了提高源程序的质量和可维护性，从而最终提高软件产品生产力，特编写此规范。

2 单片机 C51 编程规范－范围

本标准规定了程序设计人员进行程序设计时必须遵循的规范。本规范主要针对 C51 编程语言和 keil 编译器而言，包括排版、注释、命名、变量使用、代码可测性、程序效率、质量保证等内容。

3 单片机 C51 编程规范－总则

格式清晰
注释简明扼要
命名规范易懂
函数模块化
程序易读易维护
功能准确实现
代码空间效率和时间效率高
适度的可扩展性

4 单片机 C51 编程规范－数据类型定义

编程时统一采用下述新类型名的方式定义数据类型。
建立一个 datatype.h 文件，在该文件中进行如下定义：

```
typedef bit BOOL; // 位变量 //
typedef unsigned char INT8U; // 无符号 8 位整型变量 //
typedef signed char INT8S; // 有符号 8 位整型变量 //
typedef unsigned int INT16U; // 无符号 16 位整型变量 //
typedef signed int INT16S; // 有符号 16 位整型变量 //
typedef unsigned long INT32U; // 无符号 32 位整型变量 //
typedef signed long INT32S; // 有符号 32 位整型变量 //
typedef float FP32; // 单精度浮点数(32 位长度) //
typedef double FP64; // 双精度浮点数(64 位长度) //
```

5 单片机 C51 编程规范－标识符命名

5.1 命名基本原则

命名要清晰明了，有明确含义，使用完整单词或约定俗成的缩写。通常，较短的单词可通过去掉元音字母形成缩写；较长的单词可取单词的头几个字母形成缩写。即"见名知意"。
命名风格要自始至终保持一致。
命名中若使用特殊约定或缩写，要有注释说明。
除了编译开关/头文件等特殊应用，应避免使用以下划线开始和／或结尾的定义。

同一软件产品内模块之间接口部分的标识符名称之前加上模块标识。

5.2 宏和常量命名

宏和常量用全部大写字母来命名，词与词之间用下划线分隔。对程序中用到的数字均应用有意义的枚举或宏来代替。

5.3 变量命名

变量名用小写字母命名，每个词的第一个字母大写。类型前缀（u8\s8 etc.）全局变量另加前缀 g_。

局部变量应简明扼要。局部循环体控制变量优先使用 i、j、k 等；局部长度变量优先使用 len、num 等；临时中间变量优先使用 temp、tmp 等。

5.4 函数命名

函数名用小写字母命名，每个词的第一个字母大写，并将模块标识加在最前面。

5.5 文件命名

一个文件包含一类功能或一个模块的所有函数，文件名称应清楚表明其功能或性质。

每个.c 文件应该有一个同名的.h 文件作为头文件。

6 单片机 C51 编程规范－注释

6.1 注释基本原则

有助于对程序的阅读理解，说明程序在"做什么"，解释代码的目的、功能和采用的方法。

一般情况源程序有效注释量在 30%左右。

注释语言必须准确、易懂、简洁。

边写代码边注释，修改代码同时修改相应的注释，不再有用的注释要删除。

6.2 文件注释

文件注释必须说明文件名、函数功能、创建人、创建日期、版本信息等相关信息。

修改文件代码时，应在文件注释中记录修改日期、修改人员，并简要说明此次修改的目的。所有修改记录必须保持完整。

文件注释放在文件顶端，用"/*……*/"格式包含。

注释文本每行缩进 4 个空格；每个注释文本分项名称应对齐。

```
/*************************************************************
文件名称：
作  者：
版  本：
说  明：
修改记录：
*************************************************************/
```

6.3 函数注释

6.3.1 函数头部注释

函数头部注释应包括函数名称、函数功能、入口参数、出口参数等内容。如有必要还可增加作者、创建日期、修改记录（备注）等相关项目。

函数头部注释放在每个函数的顶端，用"/*……*/"的格式包含。其中函数名称应简写为 FunctionName()，不加入、出口参数等信息。

```
/************************************************************
函数名称：
函数功能：
入口参数：
出口参数：
备  注：
************************************************************/
```

6.3.2 代码注释

代码注释应与被注释的代码紧邻，放在其上方或右方，不可放在下面。如放于上方则需与其上面的代码用空行隔开。一般少量注释应该添加在被注释语句的行尾，一个函数内的多个注释左对齐；较多注释则应加在上方且注释行与被注释的语句左对齐。

函数代码注释用"//…//"的格式。

通常，分支语句（条件分支、循环语句等）必须编写注释。其程序块结束行"}"的右方应加表明该程序块结束的标记"end of ……", 尤其在多重嵌套时。

6.4 变量、常量、宏的注释

同一类型的标识符应集中定义，并在定义之前一行对其共性加以统一注释。对单个标识符的注释加在定义语句的行尾。

全局变量一定要有详细的注释，包括其功能、取值范围、哪些函数或过程存取它以及存取时的注意事项等。

注释用"//…//"的格式。

7 单片机 C51 编程规范－函数

7.1 设计原则

函数的基本要求：

正确性：程序要实现设计要求的功能。

稳定性和安全性：程序运行稳定、可靠、安全。

可测试性：程序便于测试和评价。

规范／可读性：程序书写风格、命名规则等符合规范。

扩展性：代码为下一次升级扩展留有空间和接口。

全局效率：软件系统的整体效率高。

局部效率：某个模块／子模块/函数的本身效率高。

编制函数的基本原则：

单个函数的规模尽量限制在 200 行以内（不包括注释和空行）。一个函数只完成一个功能。

函数局部变量的数目一般不超过 5～10 个。
函数内部局部变量定义区和功能实现区（包含变量初始化）之间空一行。
函数名应准确描述函数的功能。通常使用动宾词组为执行某操作的函数命名。
函数的返回值要清楚明了，尤其是出错返回值的意义要准确无误。
不要把与函数返回值类型不同的变量，以编译系统默认的转换方式或强制的转换方式作为返回值返回。
减少函数本身或函数间的递归调用。
尽量不要将函数的参数作为工作变量。

7.2 函数定义

函数若没有入口参数或者出口参数，应用 void 明确申明。
函数名称与出口参数类型定义间应该空一格且只空一格。
函数名称与括号()之间无空格。
函数形参必须给出明确的类型定义。
多个形参的函数，后一个形参与前一个形参的逗号分割符之间添加一个空格。
函数体的前后花括号"｛｝" 各独占一行。

7.3 局部变量定义

同一行内不要定义过多变量。
同一类的变量在同一行内定义，或者在相邻行定义。
先定义 data 型变量，再定义 idtata 型变量，再定义 xdata 型变量.
数组、指针等复杂类型的定义放在定义区的最后。
变量定义区不做较复杂的变量赋值。

7.4 功能实现区规范

一行只写一条语句。
注意运算符的优先级，并用括号明确表达式的操作顺序，避免使用默认优先级。

各程序段之间使用一个空行分隔，加以必要的注释。程序段指能完一个较具体的功能的一行或多行代码。程序段内的各行代码之间相互依赖性较强。
不要使用难懂的技巧性很高的语句。
源程序中关系较为紧密的代码应尽可能相邻。
完成简单功能、关系非常密切的一条或几条语句可编写为函数或定义为宏。

8 单片机 C51 编程规范－排版

8.1 缩进

代码的每一级均往右缩进 4 个空格的位置。

8.2 分行

过长的语句（超过 80 个字符）要分成多行书写；长表达式要在低优先级操作符处划分新行，操作符放在新行之首，划分出的新行要进适当的缩进，使排版整齐，语句可读。避免把注释插入分行中。

8.3 空行

文件注释区、头文件引用区、函数间应该有且只有一行空行。

相邻函数之间应该有且只有一行空行。

函数体内相对独立的程序块之间可以用一行空行或注释来分隔。

函数注释和对应的函数体之间不应该有空行。

文件末尾有且只有一行空行。

8.4 空格

函数语句尾部或者注释之后不能有空格。

括号内侧（即左括号后面和右括号前面）不加空格，多重括号间不加空格。

函数形参之间应该有且只有一个空格（形参逗号后面加空格）。

同一行中定义的多个变量间应该有且只有一个空格（变量逗号后面加空格）。

表达式中，若有多个操作符连写的情况，应使用空格对它们分隔：

在两个以上的关键字、变量、常量进行对等操作时，它们之间的操作符前后均加一个空格；在两个以上的关键字、变量、常量进行非对等操作时，其前后均不应加空格；

逗号只在后面加空格；

双目操作符，如比较操作符，赋值操作符"="、"+="，算术操作符"+"、"%"，逻辑操作符"&&"、"&"，位操作符"<<"、"^"等，前后均加一个空格；

单目操作符，如"!"、"~"、"++"、"-"、"&"（地址运算符）等，前后不加空格；

"->"、"."前后不加空格；

if、for、while、switch 等关键字与后面的括号间加一个空格；

8.5 花括号

if、else if、else、for、while 语句无论其执行体是一条语句还是多条语句都必须加花括号，且左右花括号各独占一行。

do{}while()结构中，"do"和"{"均各占一行，"}"和"while();"共同占用一行。

```
if ( ) do
{ {

} }while( );
else
{

}
```

8.6 switch 语句

每个 case 和其判据条件独占一行。

每个 case 程序块需用 break 结束。特殊情况下需要从一个 case 块顺序执行到下一个 case 块的时候除外，但需要在交界处明确注释如此操作的原因，以防止出错。

case 程序块之间空一行，且只空一行。

每个 case 程序块的执行语句保持 4 个空格的缩进。

一般情况下都应该包含 default 分支。

```
Switch ( )
{
case  x:

break;

case  x:

break;

default:

break;
}
```

9 单片机 C51 编程规范－程序结构

9.1 基本要求

有 main()函数的.c 文件应将 main()放在最前面，并明确用 void 声明参数和返回值。

对由多个.c 文件组成的模块程序或完整监控程序，建立公共引用头文件，将需要引用的库头文件、标准寄存器定义头文件、自定义的头文件、全局变量等均包含在内，供每个文件引用。通常，标准函数库头文件采用尖角号< >标志文件名，自定义头文件采用双撇号""标志文件名。

每个.c 文件有一个对应的.h 文件，.c 文件的注释之后首先定义一个唯一的文件标志宏，并在对应的.h 文件中解析该标志。

在.c 文件中：

```
#define  FILE_FLAG
```

在.h 文件中：

```
#ifdef  FILE_FLAG
#define  XXX
#else
#define  XXX  extern
#endif
```

对于确定只被某个.c 文件调用的定义可以单独列在一个头文件中、单独调用。

9.2 可重入函数

可重入函数中若使用了全局变量，应通过关中断、信号量等操作手段对其加以保护。

9.3 函数的形参

由函数调用者负责检查形参的合法性。

尽量避免将形参作为工作变量使用。

9.4 循环

尽量减少循环嵌套层数

在多重循环中，应将最忙的循环放在最内层

循环体内工作量最小

尽量避免循环体内含有判断语句