维普资讯 http://www.cqvip.com

电 子 测 量 技 术
ELECTRONIC MEASUREMENT TECHNOLOGY
第29卷 第5期
2006年10月

Keil C51 中调用 A51 程序

梁 航 王剑钢 张 帅
(吉林大学电子科学与工程学院 长春 130012)

摘 要：单片机C语言编程的过程中，在某些对时序要求比较严格的情况下，直接使用汇编语言可以提供一种灵活高效的解决方法。文中详尽叙述了Keil C51调用A51程序编程的命名规则和参数传递等规则，并通过对具体实例的分析说明了混合调用的具体实现方法。

关键词：C语言；汇编程序；混合编程

On interfacing Keil C51 to A51

Liang Hang Wang Jiangang Zhang Shuai
(College of Electronic Science and Engineering, Jilin University, Changchun 130012)

Abstract: When we need to improve the speed of a function to meet the strict sequences requirement or to directly deal with CPU, interfacing C51 to A51 is a flexible and effective method. This article discussed the rule of mixed programming C51 and A51 in detail and explained the concrete implementation.

Keywords: C language; assemble programs; mixed programming

0 引 言

目前，C语言已经成为嵌入式系统软件开发的主流语言，相对于汇编语言来说虽然有很多优点，但它并不能完全代替汇编语言。在某些直接针对处理器的操作中和某些对时序要求比较严格的情况下，或者对已经有写好的汇编程序，不希望用C语言重写，合理使用汇编语言将提供一种更加灵活高效的解决方法，而且在程序优化时也离不开汇编语言。目前关于51单片机的书籍中都会提及混合调用的问题，但很少给予全面的讲解，也很少对实现的具体方法给出详细说明，而手册中相关的信息又分布在不同章节中不便于整体把握。本文中将对混合调用中设计的问题和注意事项给予尽量详细的说明。

1 调用的规则

要在C语言中成功调用汇编必须按照相关的规则来编写汇编程序，同时对C语言也有一定要求。因为C51编译器可以将已有的C程序编译成汇编的目标代码文件，通过对这一过程的分析也可以比较清楚地了解调用的规则。

1.1 函数命名对应规则

表1说明了C语言和汇编源文件中的函数命名规则的对应关系[1]。

需要注意的是C语言中大小写函数或参数命名是不同的，但是混合编程时汇编程序中不区分大小写，因此C语言命名中最好不要出现只有大小写不同的同名函数或参数。

表1 函数命名规则

C语言中的函数	汇编文件中的函数名(标号)	规则说明
void func(void)	FUNC	无参数传递或将不含寄存器参数的函数名直接转换成大写
void func(char)	_FUNC	带寄存器参数的函数转换成大写并加"_"前缀
void func (void) reentrant	_? FUNC	可重入函数转换成大写并加"_?"前缀，表明该函数包含栈内的参数传递

2.2 段命名规则和数据目标规则

C51编译器生成的目标代码中，不同的数据段的段名不同。段名是由两个"?"为界中间为储存器类型区分符的前缀组成(如表2所示)。

主函数与局部函数的代码段均声明为code段类型形式为：

? 储存器类型区分符? 函数名? 模块名(即当前的汇编文件名) SEGMENT 段类型

例：? PR? FUNCTION_NAME? MODULE_NAME SEGMENT CODE

全局数据段的声明形式为：? 储存器类型区分符? 模块名 SEGMENT 段类型

维普资讯 http://www.cqvip.com

例:? DT? MODULE_NAME SEGMENT DATA ;data 型全局变量(SMALL 存储器模式)

? PD? MODULE _ NAME SEGMENT XDATA INPAGE ;(COMPACT 存储器模式)

? XD? MODULE_NAME SEGMENT XDATA ;(LARGE 存储器模式)

? BI? MODULE_NAME SEGMENT BIT ;bit 型全局变量

局部数据段的声明形式为:? 储存器类型区分符? 函数名? 模块名 SEGMENT 段类型 OVERLAYABLE

其中储存器类型的确定与全局数据段中的规则相同。当汇编程序按照 C 编译器编译的代码规则编写,连接器将能对程序数据段进行覆盖分析。声明中的"OVERLAYABLE"标志就是用来打开段的覆盖属性,允许连接器进行覆盖分析。没有这个属性则段中的变量将一直占用这些空间,这样会降低储存器利用效率。

表 2 标准段名前缀表

段前缀	C51 储存器类型	汇编语言段类型	说明
? PR?	program	CODE	可执行的程序代码
? CO?	code	CODE	程序储存区的常数数据
? BI?	bit	BIT	内部数据区的为寻址数据
? BA?	bdata	BDATA	内部数据区的可为寻址数据
? DT?	data	DATA	内部数据区
? FD?	far		FAR 存储区(RAM 空间)
? FC?	const far		FAR 存储区(常数 ROM 空间)
? ID?	idata	IDATA	间接寻址内部数据区
? PD?	pdata	XDATA INPAGE	外部数据区的分页数据
? XD?	xdata	XDATA	XDATA 存储区(RAM 空间)
? XC?	const xdata		XDATA 存储区(常数 ROM 空间)

1.3 参数与返回值的传递规则

Keil c 在传递参数时一般使用当前的寄存器组,通过工作寄存器组最多可以传递 3 个参数,而其余的参数则要通过固定存储区传递(如表 3 所示)。函数的返回值根据其类型不同使用的寄存器的具体情况如表 4 所示。这样可以得到类似汇编程序的高效率。

表 3 参数传递寄存器选择[5]

参数	参数类型			
	char,1byte ptr	int,2byte ptr	long,float	generic ptr
第一个	R7	R6(M),R7(L)	R4(M),~R7(L)	R1,R2,R3
第二个	R5	R4(M),R5(L)	R4(M),~R7(L)	R1,R2,R3
第三个	R3	R2(M),R3(L)	R4(M),~R7(L)	R1,R2,R3

表 4 函数返回值使用的寄存器[5]

返回值类型	寄存器	说明
bit	C	进位标志
(unsigned)char	R7	
(unsigned)int	R6,R7	高位在 R6,低位在 R7
(unsigned)long	R4~R7	高位在 R4,低位在 R7
float	R4~R7	32 位 IEEE 格式,指数和符号位在 R7
指针	R1,R2,R3	指针类型在 R3,高位在 R2,低位在 R1

如果寄存器被占用,或在程序中用了 # pragma NOREGPARMS 指令则参数只能通过存储器中的默认段来传递。用做参数传递的固定存储区的地址空间与存储模式决定。SMALL 模式是最有效的,参数段用内部数据区;COMPACT 和 LARGE 模式用外部数据区。当通过寄存器传递参数时,如果无法用寄存器组传递全部参数,在这些段中仍然会给所有的参数分配空间,参数按每个段中的声明顺序保存。定义格式如下:

? 函数名? BYTE(BIT):

……

第 N 个变量名:DS(DBIT) 字节数(1) (无法在寄存器组中传递的参数)

(或)第 N 个变量名? 变量存储地址:DS(DBIT) 字节数(1)

1.4 参数与变量传递的其他方法介绍

变量的传递还可以利用 C 语言访问绝对存储地址的方法来达到目的。Keil C 访问绝对存储地址常用下面两种方法:

(1)绝对存储访问宏,用下面的宏直接访问 51 存储区

CBYTE DBYTE CWORD DWORD

PBYTE PWORD XBYTE XWORD

FCVAR FVAR FARRAY FCARRAY;

(2)在 C 程序中使用用_at_关键词

变量类型[分配尺寸]变量名_at_绝对地址。

在_at_后的绝对地址如步可用 Keil Cx51 编译器会报错。要注意的是:如用_at_关键词声明一个变量来访问一个 XDATA 外围设备,应使用 volatile 关键词以确保 C 编译器不会将必要的存储区访问优化掉;绝对变量不能在定义时初始化;bit 类型的函数和变量不能被定位到绝对

维普资讯 http://www.cqvip.com

地址。

2 C51 调用 A51 的程序实例

下面给出一个 μVision2 中 C 语言调用汇编程序的例子：

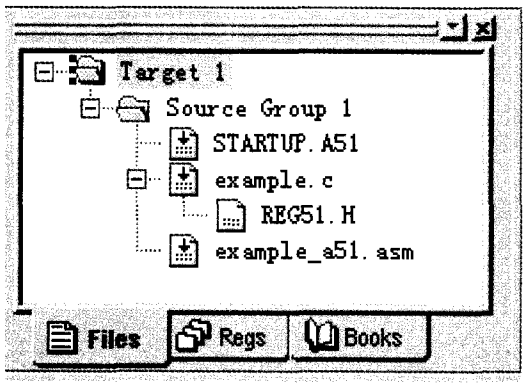

图 1　项目管理窗口

如图 1 所示，其中 example.c 为主函数所在文件，main()主函数中调用由汇编程序编写的 test_func(char, int, long)函数。example_a51.asm 是 test_func()函数所在的汇编文件。通过对 test_func 函数的调用可以说明在调用过程中对寄存器的使用情况。各函数功能描述如下：

(1)文件名：example.c；

函数：main()

功能：调用 test_func()函数，传递给函数 3 个参数，分别为 char、int、long 类型；接收返回值为 int 类型并打印。

(2)文件名：example_a51.asm

函数：test_func()

功能：接受 char、int、long 类型的 3 个参数，并将其中 int 类型的变量返回。

程序流程简图如图 2 所示：

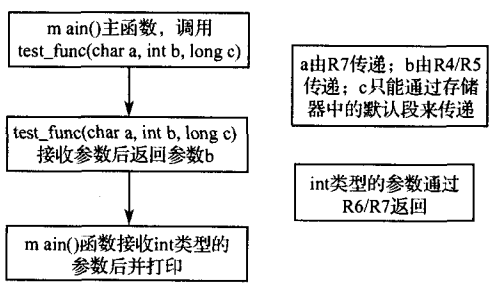

图 2　程序流程简图

另外，用控制命令 #pragma asm/endasm 也可在 C 程序中嵌入汇编代码，例如：

```
void test(char sig)          //子函数定义
{   #pragma asm
      ……汇编语言
    #pragma endasm
    return;
}
```

需要注意的是，在使用这个命令的时候要先设置选项。具体如下：

右键点击 PROJECT 管理窗口中文件表中的文件，选择 Options for…… 打开选项，选属性页，使能 Generate Assembler SRC file 和 Assemble SRC file 选项。如图 3 所示。

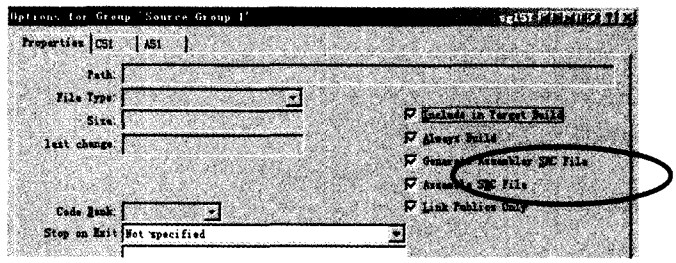

图 3　预先设置选项示意图

3 结　论

综上，在单片机的开发过程中，可以利用本文给出的混合编程的各种规则，合理地应用 C51 与 A51 的混合编程方法，这样既能充分发挥 C 语言的结构性强、易维护和开发效率高等优点，又可以灵活地用汇编语言来弥补 C 语言的局限，得到高效与精确的应用程序。

参 考 文 献

[1] Cx51 Compiler User's Guide[Z]:164-165.

[2] 靳达. 单片机应用系统开发实例导航[M]. 北京：人民邮电出版社，2003.

[3] 赵亮，侯国锐. 单片机 C 语言编程与实例[M]. 北京：人民邮电出版社，2003.

[4] 张培仁. C 语言编程 MSC-51 单片机原理与应用[M]. 北京：清华大学出版社，2002.

[5] 马忠梅. 单片机的 C 语言应用程序设计[M]. 北京：北京航空航天大学出版社，1999.

[6] 彭超，李晓白. 温度测控系统[J]. 电子测量技术，2005(5):61-62.