单片机 C51 开发新技术的研究

石春和[1] 乔 宇[2] 王 江[2]

(1. 湘潭矿业学院信息与电气工程系,湖南湘潭,411201)
2. 天津大学自动化学院,天津,300072)

摘 要 从单片机的技术应用角度解决了当前 51 系列单片机 C51 与汇编语言开发的关键技术,并将 BL51、RTX51 及其对系统头文件配置等高级技术作了深入研究,以期开发出更高性能的单片机系统,在实际应用中取得了满意的效果. 图 1,参 4.

关键词 单片机 系统头文件 BL51 RTX51
分类号 TP15

第一作者简介 石春和 男 34 岁 硕士 工程师 计算机控制及应用

单片机进入中国十余年以来,以其体积小、功能强、扩展灵活、支持芯片多、使用方便的特点,逐渐渗透到各行业的工程实际应用中[1]. 其中 8 位单片机 8051 已成为从小型到中型应用的首选,是单片机领域事实上的标准. 但是,我国的 51 单片机软件开发在不少项目中仍停留在汇编代码编制、调试的水平上. 事实上,早在 1985 年就推出了 8051 的 C51 编译器[2],使用 C 语言进行 51 单片机的开发,可以很好的利用现成的大量 C 程序资源与丰富的库函数,生成的代码编译效率高、可移植性好、完全模块化,加快了项目的进度.

1 C51 程序员与汇编程序员协同开发

目前,单片机的开发中汇编程序员是大量的. 在项目中如何协同 C51 程序员与汇编程序员的开发,并非仅仅是个混合编程问题,而应各展所长,互补所短.

汇编语言的难点在于数据处理,由于汇编语言不直接支持浮点运算,而现在单片机开发日趋复杂,在许多地方须应用高精度的复杂算法,C51 直接支持单精度的浮点运算,对于大多数场合已经够用,并且可以通过算法扩展到双精度. 算法的设计上,已有大量的 C 程序可供选用,基本不须重新开发. 在这些模块中应用 C51 可以尽快解决问题[2].

然而,完成同样功能,C51 程序经编译链接生成的代码比汇编程序生成的代码稍长. 在需要实时响应的场合,开发者惟恐执行时间太长,所以这些模块一般用汇编代码实现. 在开发简单的位控制器时,汇编代码也比 C51 有效地多.

在此,笔者要说明的是,凡是汇编能实现的功能用 C51 都可实现,只是一个熟练程度的问题. 而且,据最新的资料,新版的 C51 编译器的编译效率可以达到 1.1. 在单片机项目中应用 C51,更重要的是开发周期可以大大缩短. 一般说来,一个熟练的 C51 程序员的开发速度是汇编程序员的两倍以上,两者的执行速度相差无几. 如果不是有特别苛刻的要求,程序整体用 C51 开发要更快.

收稿日期:1999-12-03 本文责任编辑 邓学良

接下来就是 C51 模块与汇编模块相互调用问题.

标准的 C51 与汇编程序的参数传递是通过寄存器 R1 - R7 进行[4],但这种方式会破坏寄存器中的内容,对于协同开发不太合适.笔者认为,通过开发者之间约定内存或外存的一块共享存储区来传递数据更有效.但这里要注意的是 C51 的数据格式要求是"高字节在低地址,低字节在高地址"(对于常用的 int 而言),而许多汇编算法是从低地址向高地址进行运算[3],因此要对数据进行相应的处理.

下面以汇编语言做主程序为例,说明一个相互调用的实例.

因为每一个 C51 函数在汇编语言中都对应一个 PUBLIC 声明,所以用 EXTRN CODE(…)引入即可.编译链接时采用了覆盖与共享技术,以节省空间,因此汇编程序中用到的存储区必须声明保留,否则会被 C51 程序使用并覆盖.

```
NAME REMAINW
?PR?MAIN?REMAINW SEGMENT CODE
;声明主程序代码段浮动定位,因为要在此段中调用 C51 函数,所以必须按照这种格式书写.
?BIT?DDBIT SEGMENT BIT
EXTRN CODE(FLOAT1,FLOAT2,PRINT1) ;声明 C51 的函数
EXTRN CODE(KEYIN,PRDIG,PRCHAR,ERROR,HASSIS,ZTOF) ;声明汇编的外部调用子程序……
EXTRN DATA(DATE1,DATE2,MCHNUM,MISSDAY,MISSMON,DANGCI,TAXM)
;声明汇编的外部引用数据段
……
WXL1 DATA 24H ;声明须保留的内部存储区
KCHLG BIT 00H
FLAGB BIT 01H
……
RSEG ?BIT?DDBIT ;声明须保留可重定位的内部存储区
THFLAG:DBIT 1
……
XSEG
ORG 0000H ;声明须保留的外部存储区
ZSY:DS 512
ORG 0340H
WXL:DS 984
ORG 0900H
ZW:DS 5888
COUNT EQU 39H ;声明符号
……
PUBLIC CHUSH2, SPCOM, KAOQINGSCD, CHUSH0, SMP2, BBFLAG, CHFLAG, SOFLAG,MXP2,TFLAG2
```

```
;声明公用子程序
CSEG AT 0000H
LJMP MAIN
RSEG ?PR?MAIN?REMAINW
MAIN:MOV SP,#50H ;定位堆栈
     ……
     LCALL FLOAT1 ;调用 C51 计算程序
     ……
     LCALL FLOAT2 ;调用 C51 计算程序
     ……
     LCALL PRINT1 ;调用 C51 输出打印程序
     ……
```

(以上程序用 FranklinC51、A51、L51 编译链接)

全部程序编译链接通过后,应仔细查看生成的 M51 文件,有无溢出或冲突的情况,数据存储区与程序存储区是否定位恰当.

C51 程序的代码段、参数段都是浮动定位的,当 C51 代码段、参数段与汇编程序冲突时,应用 L51 的 CODE(…)、XDATA(…)、DATA(…)来移动代码段、参数段以解决冲突.STACK 堆栈段的值应根据最后的 M51 文件所显示的总共占用的内存 RAM,而设为最大可用堆栈段,C51 的子程序调用最好不超过 3 级,以免堆栈溢出.

这些 C51 程序在主程序中与汇编语言做到了"无缝链接". 但在与 3 个以上汇编模块同时链接时,编译链接系统报警. 笔者采用了如下技术解决了这一问题.

```
……
PUBLIC PACKET
……
PACKET :LCALL FLOAT1
RET
……
```

通过给在主程序声明的 C51 函数外部做个"外壳",将 C51 函数完全转化为汇编语言公共子程序,可以与任意个汇编子模块相链接.

2 系统头文件的应用

20 世纪 80 年代中期,INTEL 公司将 8051 内核使用权以专利互换或出售的形式转给世界上众多著名 IC 制造厂商,如 PHILIPS、SIEMENS、ATMEL、LG、DALLAS、AMD、OKI 等,这些厂商在自身技术基础上又发展出了各具特色的 51 系列单片机. 在 8051 内核的基础上集成了多种功能,如多通道高精度的 A/D 转换器、显示驱动、CAN 总线等功能;或扩展了 51 的引脚、中断、定时器,如 80451;或删减了 51 的地址与数据总线,使之成为最小系统应用的专用型,如 891051、892051,使 8051 成为有上百个品种的大家族. 开发者必须会定制所用单片机的系统头文件,以充分发挥各种单片机的功能. 系统头文件 xxx.h 实际上定义的是各端口、各

功能寄存器所对应的物理地址.当单片机型号改变时,只须转换相应的头文件,即可实现 C51 程序的可移植性.通过将 C51 程序与头文件相分离,也正是 C51 程序移植性的魅力所在.这些头文件也可以从 INTERNET 上专业编译器厂商的网址上下载(如 KEIL 的网址 http://www.keil.com),但总不如自己定制来的得心应手.

头文件的定制还有其它用途.目前我国自行开发的 8051 单片机仿真器有不少不能真实仿真 P0、P2 口,只将它们映射到内存的某些地址上,此时应编辑一个对应此仿真器的系统头文件,就可以正常仿真 C51 程序了.开发成功后,再使用原头文件(一般为 REG51.H)编译即可脱机运行.另外,开发者可以将单片机系统的外设定义做成一个头文件,一旦外部接线改变,只须修改头文件即可.下面是笔者根据所用仿真器而修改的头文件:

```
/* BYTE Register */
sfr P0 = 0xd8;/* 依仿真器而修改 */
sfr P1 = 0x90;
sfr P2 = 0xc8;/* 依仿真器而修改 */
sfr P3 = 0xB0;
sfr PSW = 0xD0;
sfr ACC = 0xE0;
sfr B = 0xF0;
sfr SP = 0x81;
sfr DPL = 0x82;
sfr DPH = 0x83;
sfr PCON = 0x87;
……
(以下略)
```

当然,相应的位地址也应作相应修改.

3　BL51 技术

众所周知,8051 汇编程序的代码上限为 64KB,面对目前日益复杂的应用颇有力不从心之感.FranklinC51 工具包所带的 BL51 链接器,恰如其分的解决了这一问题.现在常用的 BL51V2.10 可以管理至 1M 代码,1 个公共区与 16 个代码组,每组 64KB 以下.最新版本的 BL51 可以管理 1 个公共区与 32 个代码组,多达 2M 代码.

应用 BL51 技术的关键在于对全部代码的合理划分.将常用的子程序或代码组之间调用的函数放于公共区,可以减少代码组的切换,提高程序的执行效率.

应用 BL51 技术需要对硬件进行扩展,占用 P1 或 P3 口的几条连续的口线.至于占哪一个口、哪一些口线,都可由开发者在 L51_BANK.A51 中定义.下举一实例,将其分成 4 组:(参数使用缺省值,即用 8051 的 I/O 口 P1 作分组地址扩展,使用 P1.3、P1.4 作为地址扩展位.)

BL51 CO {MAIN.OBJ},B0 {BANK0.OBJ},B1 {BANK1.OBJ},B2 {BANK2.OBJ},B3 {BANK3.OBJ}

将代码分为 4 组(BANK0、BANK1、BANK2、BANK3),公共区代码 MAIN 复制到每个代码组,所有组的内容在公共区的地址范围都是一样的,每一组的地址范围缺省定义为 0000H—0FFFFH. 将生成的绝对目标文件用 OC51 分组转换,最后形成的存储器地址分配如图 1(阴影部分为代码区).

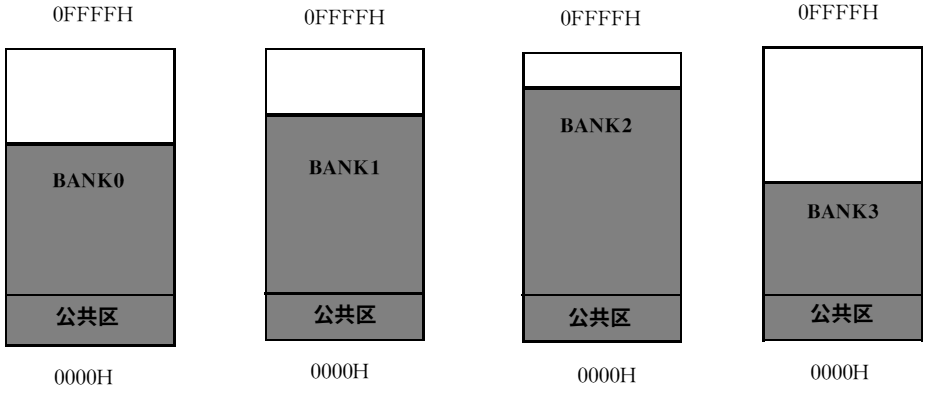

图 1 存储器地址分配图

Fig. 1 Storage address allocation

4 RTX51 技术

RTX51 技术面向复杂、对实时响应要求比较高的应用,是一种片上型的 RTOS(Real Time Operating System,实时多任务操作系统). 它使用时间片轮转算法,缺省占用 TIMER0. KEIL 的 RTOS 有 RTX51 FULL 与 RTX51 TINY 两种,RTX51 FULL 需另外付费购买 .RTX51 TINY 则在 KEIL 的工具包提供,可以应付一般的多任务操作 .

应用实例如下:

```
#include <rtx51tny.h> /* RTX_51 tiny functions & defines */
……
job0 ()_task_0 ;{
os_create_task (1);/* 标志任务 1 已就绪,启动 */
……}
job1 () _task_1 {
……}
```

RTX51 程序无 MAIN()函数,从_ TASK _ 0 开始执行,一段时间后切换至_ TASK _ 1,而后_ TASK _ 1 执行一段时间后切换回_ TASK _ 0. 两个任务就这样反复循环 . 任务之间要达成同步可通过 RTX51 TINY 的库函数来实现,如 OS _ WAIT().

链接格式为:

BL51 MAIN.OBJ RTX51TINY

即可生成多任务操作 .

将以上的单片机、汇编语言与 C51 开发技术相结合,可以高效地完成复杂的单片机开发

任务,通过对实际系统的开发使用,达到满意的效果,开发者并可从这种新的开发技术中体会到它的乐趣.

参考文献

1 徐爱钧,彭秀华.单片机高级语言 C51 应用程序设计.北京:电子工业出版社,1998.35~67

2 Corp K. 8051 Demo kit User's Guide. New York: MIT Publications, 1998. 5~62

3 Richard H B. MCS-51 微控制器系列用户指南.吴玉平译.北京:电子工业出版社,1995.25~40,101~111

4 丁元杰.单片微机原理及应用.北京:机械工业出版社,1994.39~110

STUDY ON NEW DEVELOP TECHNOLOGY OF MICROCONTROLLER C51

Shi Chunhe　Qiao Yu　Wang Jiang

(Dept. of Automation Eng. of Xiangtan Mining Institute, Xiangtan, Hunan, China, 411201)

ABSTRACT　In this paper, we solved main technology on application of mircocontroller, based on 51 series microcontroller C51 and assemble language currently, we lucubrated technology on BL51, RTX51 and how to configure their system header files, to develop higher performance microcontroller system, application in actual system, we obtain a satisfied effect. 1fig., 4refs.

Key words　microcontroller, system header file, BL51, RTX51

Synopsis of the first author　Shi Chunhe, male, born in 1966, M. E., engineer, computer control and application.