基于实时多任务操作系统 μCOS-Ⅱ的 C8051F系列单片机应用系统开发

黄亮亮，朱欣华
（东南大学 仪器科学与工程系，江苏 南京 210096）

摘要：介绍了 μCOS-Ⅱ的工作原理及 C8051F系列单片机的特点，讨论了将 μCOS-Ⅱ移植到 C8051F040单片机应用系统中的方法。应用系统实现了串口通信及 CAN通信等任务。
关键词：实时多任务操作系统；μCOS-Ⅱ；C8051F040；CAN；移植
中图分类号：TP316，TP368　　文献标识码：A　　文章编号：1000－8829(2005)09－0039－04

Application and Development of C8051F Microcontroller Family Based on RTOS μCOS-Ⅱ

HUANG Liang-liang，ZHU Xin-hua
(Department of Instrument Science and Technology，Southeast University，Nanjing 210096，China)

Abstract: The principle of μCOS-Ⅱ and the speciality of the C8051F microcontroller family are introduced. The method of porting the μCOS-Ⅱ to the application system using C8051F040 is also discussed. Several tasks，such as serial communication and CAN communication，are realized in this system.
Key words: RTOS；μCOS-Ⅱ；C8051F040；CAN；porting

随着微机系统的进一步复杂化和系统实时性需求的不断提高，以及应用软件向系统化方向的快速发展，应用实时多任务操作系统（RTOS）作为嵌入式设计的基础和开发平台已逐步成为嵌入式应用设计的主流。μCOS-Ⅱ是一个源码公开、免费的嵌入式操作系统，能成功地移植到各种 16位、32位单片机上，也能移植到 8位单片机应用系统中。本研究讨论将 μCOS-Ⅱ移植到以 Cygnal公司的高性能 8位单片机 C8051F040组成的应用系统中的方法及过程。在移植成功的基础上，编写了包括串口通信和 CAN通信功能的应用测试软件。从结果上看，CAN 通信在 μCOS-Ⅱ内核环境中，实时性得到了保证。

1 μCOS-Ⅱ及其工作原理

1.1 μCOS-Ⅱ简介

μCOS-Ⅱ是由 Jean J. Labrosse编写的一种源码公开的嵌入式实时操作系统。程序大部分用 C 语言编写，带有少量的汇编，适合小型控制系统，具有执行效率高、占用空间小、实时性能优良、可扩展性强等特点。最小内核可编译至 2 KB[1]。

μCOS-Ⅱ是基于优先级的抢占式实时多任务操作系统，其内核提供了任务管理、时间管理、资源和内存管理 4部分功能，没有文件系统、网络接口、输入输出界面。μCOS-Ⅱ共有 64个优先级，系统占用 8个，用户可创建 56个任务。任务的堆栈要占用较大的 RAM空间，空间的大小取决于任务的局部变量、缓冲区大小及可能的中断嵌套层数。应用程序的时间精度由系统时钟节拍决定，一般时钟节拍应在 10～100 Hz之间[2]。

1.2 μCOS-Ⅱ工作原理

μCOS-Ⅱ的工作核心原理是近似地让最高优先级的就绪任务处于运行状态。为了保证这个核心，它在调用系统 API函数、中断结束、定时中断结束时总是执行调度算法，原作者事先计算好数据，简化了运算量，设计就绪表结构，使得延时可预知。任务切换是通过模拟一次中断实现的。

任务切换的整个过程就是，用户任务程序调用系统 API函数，API调用 OSSched()，OSSched()调用软中断 OS_TASK_SW()，即 OSCtxSw()，返回地址（PC值）压栈，进入 OSCtxSw中断处理子程序内部。反之，切换程序调用 RETI返回紧邻 OS_TASK_SW()的下一条汇编指令的 PC地址，进而返回 OSSched()下一句，

收稿日期：2004－12－10
作者简介：黄亮亮（1981—），女，张家港人，硕士研究生，主要研究方向为导航制导与控制及嵌入式系统应用。

再返回 API下一句,即用户程序断点。因此,如果任务从运行到就绪再到运行,它是从调度前的断点处运行。

当产生中断时,先关中断进行中断处理,在退出前必须进行任务调度,此时调用 OSIntCtxSw( ),它与 OSCtxSw( )类似,只是对堆栈指针做了简单调整,以保证所有挂起任务的栈结构看起来是相同的。μCOS-Ⅱ要求中断的堆栈结构符合规范,以便正确协调中断退出和任务切换。当任务切换发生在中断退出前,此时还没有返回中断断点,先保存现场到 TCB,等到恢复现场时再从切换函数返回原来的中断断点(由于中断和切换函数遵循共同的堆栈结构,所以退出操作相同,效果也相同)。用户编写的中断子程序必须按照 μCOS-Ⅱ规范书写。因此,任务调度发生在中断退出前是非常及时的,不会等到下一时间片才处理。

2 C8051F系列单片机的特点

2.1 C8051F系列单片机的特点

Cygnal公司推出的 C8051F系列单片机将 80C51系列单片机从 MCU推向了 SOC(片上系统)时代,其主要特点是[3]:

(1)指令运行速度大大提高。C8051F系列采用 CIP-51的 CPU模式,指令以时钟周期为运行单位。与 8051相比,相同的时钟下,单周期指令运行速度为原来的 12倍;全指令集平均运行速度为原来的 9.5倍。

(2)I/O从固定方式到交叉开关配置。C8051F系列采用开关网络以硬件方式实现 I/O端口的灵活配置。在这种通过交叉开关配置的 I/O端口系统中,单片机外部为通用 I/O口,内部有输入/输出的电路单元,通过相应的配置寄存器控制的交叉开关配置到所选择的端口上。

(3)为单片机提供了一个完善的时钟系统。当程序运行时,可实现内外时钟的动态切换。

(4)从传统的仿真调试到基于 JTAG接口的在系统调试。C8051F的 JTAG接口不仅支持 FLASH ROM的读/写操作及非侵入式在系统调试,而且它的 JTAG逻辑还为在系统测试提供了边界扫描功能。

(5)从引脚复位到多源复位。C8051F的多复位源提供了上电复位、掉电复位、外部引脚复位、软件复位、时钟检测复位、比较器 0复位、WDT复位和引脚配置复位。众多的复位源为保障系统的安全、操作的灵活性以及零功耗系统设计带来极大的好处。

2.2 C8051F040性能概述

C8051F040在一个芯片内集成了构成单片机数据采集或控制系统所需要的几乎所有的模拟和数字外设及其他功能部件。它的供电电压为 2.7~3.6V,带有内部可编程振荡器。

C8051F040的 A/D转换部分包括:一个 12位 ADC0子系统,可以测量 12路外部输入和 1路内部温度传感器输出,最高转换速度为 100 kS/s;一个 8位 ADC1子系统,可以测量 8路外部输入,最高转换速度为 500 kS/s。C8051F040的 ADC有单端输入和差分输入两种测量方式,另外它还集成了跟踪保持电路和可编程窗口检测器。同时,C8051F040有两个 12位电压方式 DAC。每个 DAC的输出摆幅均为 0V~VREF(对应的输入码范围是 0x000~0xFFF)。

C8051F040的内核采用流水线指令结构,70%的指令的执行时间为 1个或 2个系统时钟周期,它的最高时钟频率可达 25 MHz。同时,它具有 64 KB的 FLASH,可以进行在系统编程,扇区大小为 512 B,另外,它还有 64 KB的外部数据存储器接口,可以编程为复用和非复用方式。

C8051F040有 8 B宽的端口 I/O,所有口线均耐 5 V电压。它可以同时使用 SMBus、SPI以及 2个 UART串口。另外,它还有可编程的 16位计数器/定时器阵列,5个捕捉/比较模块,5个通用 16位计数器/定时器以及专用的看门狗定时器。

除上述功能外,C8051F040还集成了 BOTSH-CAN,它兼容 CAN技术规范 2.0A和 2.0B,主要由 CAN内核、消息 RAM(独立于 CIP-51的 RAM)、消息处理单元和控制寄存器组成。CAN内核由 CAN协议控制器和负责消息收发的串行/并行转换 RX/TX移位寄存器组成。消息 RAM用于存储消息目标和每个目标的仲裁掩码。这种 CAN处理器有 32个随意配置为发送和接收的消息目标,并且每一个消息目标都有它自己的识别掩码,所有的数据传输和接收滤波都是由 CAN控制器完成的,而不是由 CIP-51来完成。通常 CAN内核不能直接访问消息 RAM,而必须通过接口寄存器 IF1或 IF2来访问。另外,CIP-51的 SFR并不能直接访问 CAN内部寄存器的所有单元,其配置 CAN、消息目标、读取 CAN状态以及获取接收数据、传递发送数据都由 SFR中的 6个特殊寄存器来完成,其中 CAN0CN、CAN0TST和 CAN0STA 3个寄存器可直接获取或修改 CAN控制器中对应的寄存器,而 CAN0DATH、CAN0DATL、CAN0ADR 3个寄存器主要用来访问修改其他不能直接访问的 CAN内部寄存器,消息处理单元则用于根据寄存器中的信息来控制 CAN内核中移位寄存器和消息 RAM之间的数据传递,同时,它还可用来管理中断的产生。

3 移植及应用

由 2.2中所述的 C8051F040的性能可知,该系列单片机是一个性能强大的 SOC,因此,特别适用于任务繁重的小型化测控系统。当芯片具有的功能被较多地使用时,系统要处理的任务就较多,编程头绪也多。为了简化应用程序的编写思路,实现程序模块化,提高应用程序的实时性和可靠性,将 μCOS-Ⅱ移植到 C8051F040中就成为一件很有意义的事。

3.1 μCOS-Ⅱ的移植

要使 μCOS-Ⅱ正常运行,处理器必须满足以下要求:

①处理器的 C编译器能产生可重入代码;

②用 C语言就可以打开和关闭中断;

③处理器支持中断,并且能产生定时中断(通常在 10~100 Hz之间);

④处理器支持能够容纳一定量数据(可能是几千字节)的硬件堆栈;

⑤处理器有将堆栈指针和其他 CPU寄存器读出和存储到堆栈或内存中的指令。

C8051F040微控制器有 64 KB的程序空间, 4 KB的数据空间(满足 μCOS-Ⅱ对硬件堆栈的要求)和 256 B的片内 RAM区和 5个定时器/计数器, 20个中断矢量。程序的堆栈可以位于 256 B内部数据存储器中的任何位置,堆栈深度最大为 256 B,由于部分变量要存放在内部存储器,因此,每个任务占用的空间要小于 256 B,本研究设置每个任务的空间为 100 B。另外,可以使用 PUSH和 POP指令入栈和出栈。因此, C8051F040满足 μCOS-Ⅱ移植的处理器要求。

μCOS-Ⅱ的移植只与 4个文件相关:汇编文件(OS_CPU_A.ASM)、处理器相关 C文件(OS_CPU.H, OS_CPU_C.C)和配置文件(OS_CFG.H)。其中, OS_CPU.H声明了各种数据类型和宏; OS_CPU_A.ASM则主要是 4个函数: OSStarHighRdy()、OSCtxSw()、OSIntCtxSw()、OSTickISR(),完成时间中断、用户堆栈和系统堆栈之间的数据交换,中断程序一般在 OS_CPU_A.ASM中,也可以在 C中嵌汇编,为了提高系统的实时性,一般在中断程序中只是发送信号量,具体操作在任务中执行; OS_CPU_C.C包含 6个简单的 C函数: OSTaskStkInit()、OSTaskCreateHook()、OSTaskDelHook()、OSTaskSwHook()、OSTaskStaHook()、OSTimeTickHook(),惟一必要的函数是 OSTaskStkInit(),其他 5个函数必须声明但没必要包含代码。OSTaskCreate()和 OSTaskCreateExt()通过调用 OSTaskStkInt()来初始化任务的堆栈结构,因此,堆栈看起来就像刚发生过中断并将所有的寄存器保存到堆栈中的情形一样。

由于 C8051F040有 4 KB的数据存储器,通过 MOVX指令访问,所以将系统源文件中的大部分指针和变量放到该存储器。值得注意的是, C8051F040的外部数据存储器也是通过 MOVX访问,所以必须确定 EMI0CF寄存器中的模式选择。

系统的时钟节拍通过定时器 0来控制,设置每 20 ms产生一次定时中断,因此时钟节拍是 50 Hz,必须注意的是,定时器 0的中断必须在调用 OSStart()之后才能打开,否则系统容易崩溃,因此中断必须在第一个任务中打开。

由于 μCOS-Ⅱ的源文件中有许多函数是在临界区处理的,函数内部先关中断,处理完之后再开中断。如果在调用之前,中断已经关闭,则调用函数后有可能使中断打开,引起内部混乱。因此,在调用某些临界区函数时要确定中断。

另外,多个任务同时调用的函数必须具有可重入性,因此,自己编写的函数一般都需带有可重入性,在函数定义后面加 reentrant关键字即可。有些 C语言通用函数也不支持重入,例如,本研究用的 KEIL 51编译器,它的 printf库代码就不支持重入,因此,在任务中发送串口数据不能用 printf()函数,只能自己编写发送函数。

3.2 基于 μCOS-Ⅱ的 C8051F040应用系统开发

移植了 μCOS-Ⅱ的 C8051F040的每个功能都可以作为一个独立的任务在系统中运行,每个任务都有自己的堆栈空间,可以被其他任务和中断服务程序挂起。

测试应用程序实现了 4个任务: Task1是每 1 s发送 CAN数据包, Task2是每 4 s发送 CAN数据包, Task3是处理 CAN接收到的数据, Task4是每 2 s发送串口数据,波特率为 9 600 kb/s。程序中设置了两个不同 ID的 message object用来发送 CAN数据包, CAN总线接收采用中断方式,其优先级高于其他任务,为了保证系统的实时性,在中断程序中不处理数据,只是发送一个信号量,在 Task3中处理 CAN数据。发送串口数据采用的是查询方式,按字节发送。程序中设置 4个任务的优先级依次为 10, 11, 8, 12,因此, Task3的优先级最高; Task1发送的 CAN数据包的 ID为 0X001, Task2发送的 CAN数据包的 ID为 0X002,因此,在 CAN传输中 Task1发送的数据包的优先级高。

整个应用程序的结构如图 1所示。

在主程序中,首先初始化 C8051F040和 CAN,并进行系统初始化,调用 OsInit()函数,此函数中创建了一个空闲任务;然后调用 API函数,创建 4个任务(不包括空闲任务);之后创建一个信号量 CAN_EVENT,为中断与 Task3通信所用,最后调用 OSStart()函数, OS系统开始运行优先级最高的任务。Task3的优先级

最高,但是在没收到 CAN_EVENT之前,任务一直处于休眠状态,当 CAN接收器收到数据包后,Task3进入就绪态,在中断返回时,进行任务切换,执行优先级最高的 Task3。在 Task3还未收到信号量之前,Task1、Task2和 Task4根据时间延时和优先级的不同各自独立运行。从测试结果可以看出,当执行 2次 Task1后,Task4执行 1次,当执行 4次 Task1后,Task2执行 1次,这说明程序的任务调度和实时性都得到了很好的保证。

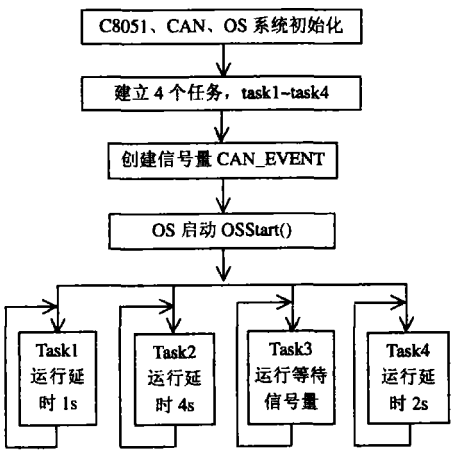

图 1　各任务在 μCOS-Ⅱ中运行的结构图

另外,为了提高代码效率,CAN中断服务程序是用汇编语言编写的:

```
        CSEG AT 009BH; CAN 中断
        LJMP CanISR
        RSEG ? PR? _? CanISR? OS_CPU_A
CanISR:
        USING    0
        CLR      EA; 关中断
        PUSHALL
        MOV      R0, #LOW(OSIntNesting)
        INC      @R0; 置标志位,表明在执行中断程序
        MOV      R0, #LOW(CAN_EVENT)
        MOV      AR3, @R0
        INC      R0
        MOV      A, @R0
        MOV      R2, A
        INC      R0
        MOV      A, @R0
        MOV      R1, A
        LCALL    _?OSSemPost; 发送 CANEVENT信号量
        LCALL    _?OSIntExit; 中断结束前进行任务切换
        POPALL
        RETI
```

4　结束语

将编写的测试程序下载到 C8051F040应用系统中进行了实际的运行测试,测试表明,基于 μCOS-Ⅱ的 C8051F040应用系统中的各任务工作正常。这些工作为正在开发的基于 μCOS-Ⅱ的 C8051F040应用系统用于各测控模块小型化的基于 CAN总线的军用车辆底盘综合电子系统的开发打下了坚实的基础。

参考文献:

[1] 邵贝贝. μCOSⅡ源码公开的实时嵌入式操作系统[M]. 北京:中国电力出版社,2001.
[2] 王劲松,等. 嵌入式操作系统 μCOS-Ⅱ的内核实现[J]. 现代电子技术,2003,(8):48-50.
[3] 潘琢金,等. C8051FXXX高速 SOC单片机原理及应用[M]. 北京:航空航天大学出版社,2002.

(上接第 35页)

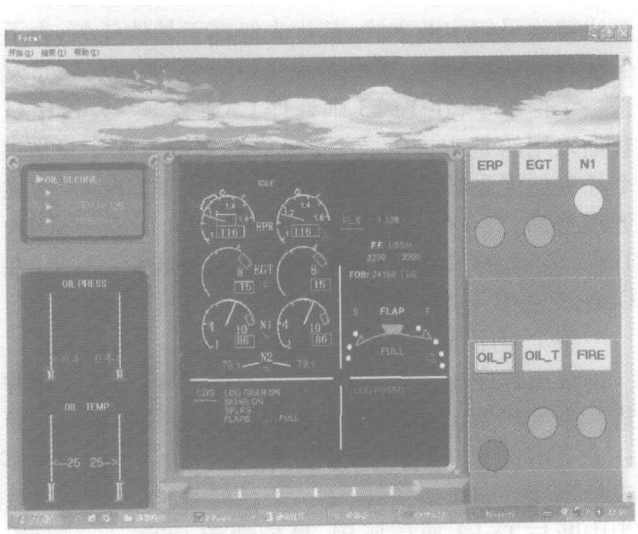

图 8　EICAS综合显示面板图

参考文献:

[1] 杜建勋. 发动机指示和机组警告原理及应用[M]. 北京:国防工业出版社,1994
[2] 盛乐山. 航空电器[M]. 北京:科学出版社,1994
[3] 孙宝元,杨宝清. 传感器及其应用手册[M]. 北京:机械工业出版社,2004
[4] 何希才. 传感器及其应用[M]. 北京:国防工业出版社,2001.
[5] 赵继文. 传感器与应用电路设计[M]. 北京:科学出版社,2002.
[6] 孙江宏,李良玉. Protel 99电路设计与应用[M]. 北京:机械工业出版社,2002
[7] 余家春. Protel99 SE原理图与 PCB设计[M]. 北京:中国铁道出版社,2003
[8] 赵负图. 传感器集成电路手册[M]. 北京:化学工业出版社,2002