文章编号:1008-0570(2009)06-2-0155-03

基于嵌入式 LINUX 下 CAN 设备驱动程序设计

Driver Design of CAN Device Under Embedded Linux

(1.桂林电子科技大学;2.广西工学院) 田小刚[1]　蔡启仲[2]　邬军伟[1]
TIAN Xiao-gang　CAI Qi-zhong　WU Jun-wei

摘要: 本文以嵌入式微处理器 S3C2410 为主控制器，通过 SPI 接口，采用 MCP2510 控制器扩展 CAN 总线接口。文章分析了 Linux 下设备驱动程序的结构和工作原理,详细论述了嵌入式 Linux 操作系统下 CAN 设备驱动程序的设计方法和具体实现。针对字符设备驱动程序的特点,采用中断驱动 I/O 口方式结合缓冲区的使用可将数据接收和 read 系统调用隔离开来,同时在系统调用函数中加入了休眠代码,确保设备在系统中的高效运行。

关键词: Linux; S3C2410; CAN 总线; 设备驱动; MCP2510
中图分类号: TP 311　　**文献标识码:** A

Abstract: Basing on s3c2410 microprocessor, through its SPI interface, the article extended CAN Bus through adopting MCP2510 CAN controller. The paper analyzes device driver of the structure and working principle under the Linux,The way of design and realization of CAN device driver under the embedded operating system LINNUX are discussed in detail. According to the characteristics of Character device driver, data receiving and system call can be seperated by the combination of interrupt-driven I/O and data buffer and the sleep code are added in the function of system call. To ensure the Equipment is operating efficiency in the system.
Key words: Linux; S3C2410; CAN Bus; Device Driver; MCP2510

技术创新

引言

目前在复杂的工业控制现场领域中,大量的传感器、执行器、控制器之间的数据传输大部分依赖于 RS232 和 CCITT V.24 通信标准。这种低速率和点对点的通信,无能力支持更高层次的现代化工业控制系统;同时,目前的局域网技术是以共享资源为主要目的,不适合工业测控数据的传输。CAN 是一种非常有效的支持实时控制和分布式控制的串行通信网络,并且具有传输速度快、可靠性高、抗干扰能力强、通信方式灵活、成本低廉等特点,因此在工业控制现场被广泛应用。目前 LINUX 操作系统中没有 CAN 设备的驱动程序，要实现 CAN 总线的通讯，必须针对所使用的嵌入式微处理器自己开发设备驱动程序。本文是以 S3C2410 微处理器为硬件平台和 Linux 操作系统为软件平台下开发 CAN 设备的驱动程序,在设计过程中,根据字符设备驱动程序的特点，运用中断驱动 I/O 口方式与缓冲区的使用相结合,将数据接收和系统调用分开,并在系统调用函数中加入了休眠代码,从而提高设备在系统中的运行效率。

1 CAN 总线硬件结构原理与设计

系统主控制器采用三星公司的 S3C2410 微处理器,CAN 控制器和收发器分别采用 Micorchip 公司的 MCP2510 控制器和 Philips 公司 TJA1050 收发器。S3C2410 是一款低功耗、性价比高的 32 位 RISC 嵌入式微处理器。同时拥有丰富的内部资源：如 SDRAM 控制器、3 个通道的 UART、117 位通用 I/O 口和 24 位的外部中断源以及 2 个 SPI 接口等等。SPI 是一个同步串行外设接口,通过 SPI 接口微处理器与外围设备可以以串行方式进行传输数据,此外还兼容 SPI 协议(ver.211),包含一个 8 位发送移位寄存器和 8 位接收移位寄存器，同时支持查询、中断、DMA 三种传输模式,传输速率可以达到 20Mb/s 以上。MCP2510 是一款独立的 CAN 控制器,其独特功能如下：(1)支持标准帧和扩展帧两种数据帧格式,0~8 字节的有效数据长度,支持远程帧;(2)2 个接收缓冲器,可以优先存储报文。6 个完全验收滤波器,2 个完全验收屏蔽滤波器;(3)3 个发送缓冲器,具有优先级设定以及发送终止功能;(4)支持回环模式(Loop back),便于测试。最大 1Mb/s 的可编程波特率;(5)支持 3~5V 宽电压范围供电。CAN 总线结构示意图如图 1 所示。

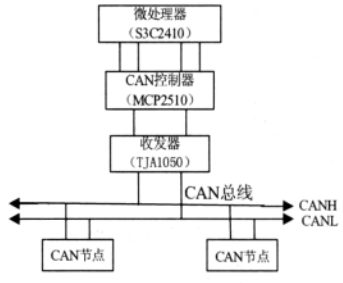

图 1 CAN 总线结构示意图

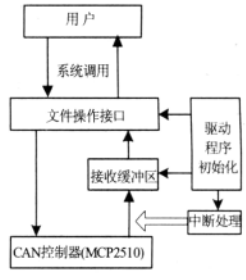

图 2 CAN 设备驱动程序结构图

2 CAN 设备驱动程序的设计

Linux 系统内核是通过驱动程序来驾驭外围硬件设备,设备驱动程序是 Linux 内核的一部分，其主要功能就是为用户屏蔽设备操作细节,对不同的设备提供统一的操作方式。一般来说在 Linux 系统中，将所有的硬件设备抽象成虚拟的设备文件系统，即可以像对待文件那样使用 write、read 等系统调用对虚

田小刚: 硕士研究生
基金项目: 基金申请人: 蔡启仲; 项目名称: 基于智能控制 CAN 现场总线的嵌入式微处理器控制系统的研究; 基金颁发部门: 广西自然科学基金资助项目 (桂科基 0448011)

拟设备文件进行文件操作。在 Linux 驱动程序支持三种类型设备:字符设备、块设备、网络设备 字符设备都是以顺序的方式写和读取的，通常都是以字节流的方式操作 ;MCP2510 就属于字符设备。在系统初始化完毕后,发送数据 :用户通过系统调用将一帧数据拷贝到内核层,发送数据处理函数将内核层的一帧数据写入 MCP2510 控制器 ;接收数据 :当一帧数据到来中断被触发,系统将跳到中断处理程序,在中断处理程序中系统调用接收数据处理函数从 MCP2510 控制器中读取数据到接收缓冲区,最后用户通过系统调用读到完整的一帧数据。其 CAN 设备驱动结构图如 2 所示：

2.1 MCP_device 结构体

在设备驱动程序运行时，系统通过接收函数将数据从 MCP2510 中读取到内核缓冲区中,用户通过系统调用 read 从缓冲区获得数据,在 read 函数中加入了休眠代码,从而提高设备在系统中的运行效率。这里我们定义 MCP_device 结构体用于记录缓冲区的各种运行状态。

```
struct MCP_device { volatile unsigned long RecvBuf;//缓冲区的入口指针
    volatile unsigned long RecvHead;//缓冲区的写入指针
    volatile unsigned long RecvTail;//缓冲区的读取指针
    volatile unsigned long RecvNum;//缓冲区的序号
    wait_queue_head_t inq;   // 全局变量
    struct semaphore sem;   //互斥信号量
}__attribute__((aligned(L1_CACHE_BYTES), packed));
```

在内核首先要建立接收缓冲区,通过 kmalloc()函数为缓冲区分配足够的内存空间，同时将缓冲区入口地址赋给 Device[i]->RecvHead(写入指针)和 Device[i]->RecvTail(读取指针)成员变量 ;当一帧数据到来,中断被激活,系统将跳到中断处理程序,通过接收函数,依次将数据接收并保存在缓冲区中,调用指针移动函数将写入指针(RecvHead)向下移动 10 个字节(CAN 控制器的接收缓冲器由 1 个字节的标识符,1 个字节的 RTR 和 DLC 位及 8 个字节的数据区组成，共 10 个字节) 系统将跳出中断处理程序,用户层通过调用 read 函数将数据从缓冲区读取。当用户层调用 read 函数从缓冲区读取数据时,在 read 操作中加入了休眠的代码，如果报文接收缓冲区的写入指针 RecvHead 和读取指针 RecvTail 值不等则说明有新的数据存入,即可以读出数据,在读取数据后调用指针移动函数将读取指针(RecvTail)向下移动 10 个字节。如果相等则说明此时没有接收到新的数据,于是进程阻塞，此时进程就要释放信号量 sem 置系统于休眠状态,或者把 CPU 的使用权交给其他进程去使用,从而提高设备在系统中的运行效率。

2.2 文件操作接口

Linux 的最显著的一个特点就是它抽象了对硬件设备的管理,为用户程序提供了一个统一的、抽象的、虚拟的文件系统界面 (即虚拟文件系统 VFS)。这个抽象界面的主体就是一个 file_operations 数据结构。结构中的成分几乎全是函数指针,每一个函数指针都指向了应用程序所调用函数的入口，编写驱动程序实质上就是填充该结构体中的每一个函数。每个文件系统都有自己的 file_operations 数据结构,同时根据设备的需要填写全部或部分的操作函数,没使用到的操作,相应的函数指针就设为 NULL。下面是 CAN 设备 file_operations 数据结构： struct file_operations Fops = { .open = device_open, //打开设备

```
    .release = device_release,   //关闭设备
    .read = device_read,         //读操作
    .write = device_write   }；  //写操作
```

这个结构的每一个成员都对应一个系统调用,其功能如下：

(1)device_open 负责对打开设备并做好准备工作,主要包括打开设备以及检查设备是否正常打开；判断 I/O 口是否可以被占用同时向系统注册此 I/O 口 ;此外向系统注册中断,安装中断处理程序 ;最后初始化硬件设备 MCP2510。

(2) device_release 主要负责关闭设备操作,主要包括释放缓冲区内存,释放 I/O,释放中断。

(3) device_read 负责从接收缓冲区读取数据，并且将数据传递给用户层。

(4) device_write 负责从用户层获得数据,并且将数据发送出去。

2.3 设备驱动程序的中断处理

在 Linux 系统中,中断处理是极为重要的一部分,并且中断信号线也是非常珍贵且有限的资源。因此在驱动程序使用中断前要先申请一个中断通道,在使用后释放该通道;CAN 设备驱动程序通过调用 request_irq 向系统请求一个中断通道。函数如下：

```
request_irq  (eint_irq,mcpcan0_handler,SA_INTERRUPT | SA_SHIRQ,MCP_name[dev->MinorNum],dev);
```

其中 eint_irq 是为设备驱动程序向系统申请的中断号(本设计为中断 5);SA_INTERRUPT |SA_SHIRQ 表示该中断处理程序是快速处理程序，并且中断处理程序运行时其他中断都被屏蔽,还表示该中断可以在设备之间共享;MCP_name[dev->MinorNum] 表示设备名;Dev 为申请中断通道时告诉系统的设备标识;mcpcan0 _handler 为向系统注册的中断处理例程，当控制器 MCP2510 接收缓冲器,装满数据后会置中断 5 为低电平,此时中断产生系统将跳转到此 mcpcan0_handler 中断处理函数内运行,在中断处理例程中接收报文数据到缓冲区中,系统将跳出中断处理程序,用户层通过调用 read 函数将数据从缓冲区读取。

2.4 驱动程序初始化

驱动程序在加载到内核中时,首先会运行驱动程序的初始化函数,然后等待系统调用。在 file_operations 数据结构中定义的相关函数，实现对设备的操作 ;CAN 设备的初始化函数主要负责创建 CAN 设备节点文件,声明和初始化信号量。初始化程序片段编写如下：

```
static int __init mcpcan_init_module(void)
 { .....
res=register_chrdev(MCP_major,"MCPCAN",&Fops);
if(res < 0) { printk("device register failed with %d.\n",res);
return res;}
if(MCP_major == 0) MCP_major = res;
MCP_device_init();.....
return 0;}
```

在 Linux 系统中,通过调用函数 register_chrdev 向系统注册字符设备设备驱动程序 ;函数原型如下：

```
register_chrdev (unsigned int major,const char *name,struct file_operation *fops);
```

参数 MCP_major 为主设备号,在内核中一个驱动程序对应一个主设备号;name 是设备文件名或者驱动程序的名。Fops 就是文件操作接口 file_operation 结构体的操作指针 如果注册成功则此函数返回 0 ;根据这些参数,我们将要在/dev 目录下创建

设备文件,可用命令 mknod /dev/can c 98 0 创建,这里面的 c 表示字符设备,98 表示主设备号,0 表示次设备号。

2.5 设备驱动程序的编译

编写完了驱动程序代码后，接下来就是编译设备驱动程序。在 Linux 系统中我们采用交叉编译工具来编译驱动程序,同时也完全支持内核驱动程序的两种加载方式：一种是静态编译到内核,一种是先编译成模块然后动态加载到内核中。对于没有 MMU 的处理器,只能采用静态加载方式。对于本设计所采用的 S3C2410 微处理器,可以采用两种方式加载,静态加载就是将驱动程序加载到内核中和内核一起编译,每次调试修改驱动程序,都要进行重新编译内核,大大降低了调试效率。而本文采取动态加载,将驱动程序编译成模块,通过 insmod 命令将驱动程序模块加载到内核中去运行，编译一个驱动程序的时间远远小于编译一个内核的时间,从而大大提高调试效率。

编译驱动程序需要 Make 工具，同时需要 Makefile 文件来管理。简单的说,Makefile 告诉 Make 工具应该使用什么样的工具(通常是指编译器),使用那些选项(如头文件、库文件路径、优化级别等),同时还可以告诉 Make 希望编译的结果。本驱动程序代码需要 arm-linux-gcc 和 arm-linux-ld 编译器来进行编译,因此需要在 Makefile 中要添加如下命令 CC:=/usr/local / arm/ 2.95.3/bin/arm -linux -gcc 和 LD: =/usr/local/arm/2.95.3/bin/arm -linux-ld；同时还要告诉编译器头文件的路径，命令如:INCLUDE:=-I/home/cvtech/jx2410/linux/include-I../ include 最后就是将三个源文件(mcpcan.c、spi_cmd.c 和 spi.c)合并编译连接成一个可执行文件即驱动模块，命令如下 CFLAGS:=-O2 -DMODULE -D__KERNEL__ $ {INCLUDE},$ (EXEC): $(OBJS), $(LD) $(LDFLAGS) -o $@ $(OBJS) 在 root 权限下执行 Make 命令编译驱动程序，就可以得到可以动态加载或者卸载的 CAN 驱动程序模块 candrv.o

3 结论

本文详细地介绍了在嵌入式 Linux 操作系统下 CAN 控制器 MCP2510 驱动程序的开发设计过程。该驱动程序根据字符设备驱动程序的特点,合理的设计了数据缓冲区,将数据的接收与 read 系统调用分离开来，提高系统的整体性能，并结合 LINUX 下驱动程序的编写一般规则,编写了相关操作代码。为在嵌入式系统中扩展其它硬件设备的驱动程序提供了很好的参考案例,此外 Linux 源代码是开放的,开发者可以利用这一特点开发出各种相关产品。

本文作者创新点: 采用中断驱动 I/O 口方式结合缓冲区的使用可将数据接收和 read 系统调用隔离开来,从而提高设备在系统中的运行效率。

参考文献

[1]饶运涛,邹继军,王进宏,郑勇芸.现场总线 CAN 原理与技术应用[M].北京航空航天大学出版社,2008.7
[2]刘淼.嵌入式系统接口设计与 Linux 驱动程序开发[M].北京航空航天大学出版社,2006.5 [3]徐德民.操作系统原理 LINUX 篇[M].国防工业出版社,2004.1
[4]RUBINT A,CORBET J.LINUX 设备驱动程序 (第三版) [M].魏永明,耿岳,钟书毅.中国电力出版社,2007.7
[5]黄捷锋,蔡启仲,郭毅锋,田小刚.CAN 总线在嵌入式 LINUX 下的实现[J].微计算机信息,2008,1-2: 45-47

作者简介: 田小刚(1982-)男,宁夏人,汉族,硕士研究生,研究方向:智能控制、嵌入式系统设计; 蔡启仲(1956-),男,湖南人,教授, 硕士,硕士导师,研究方向: 智能控制、嵌入式系统设计；邬军伟(1982-)男,陕西,汉族, 硕士,研究方向: 智能控制、嵌入式系统设计。

Biography: TIAN Xiao-gang, Male (1982-), Born in ningxia province, the Han nationality, Master, Main research field: Intelligent Control、Embedded Systems Design.

(541004 广西桂林 桂林电子科技大学) 田小刚 邬军伟
(545006 广西柳州 广西工学院) 蔡启仲
(Guilin University of Electronic Technology, Guilin 541004) TIAN Xiao-gang WU Jun-wei
(Guangxi University of technology, Liu Zhou,545006) CAI Qi-zhong
通讯地址:(545006 广西柳州广西工学院电控系) 蔡启仲 转 田小刚

(收稿日期:2009.05.03)(修稿日期:2009.06.05)

(上接第 121 页)

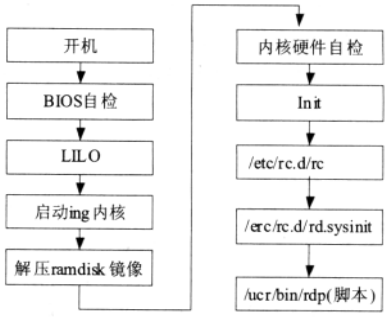

图 5 RDP 启动流程图

本文作者创新点:本文从 Linux 体系结构的角度提出了网络计算机操作系统层次结构。根据提出的嵌入式 Linux 网络计算机操作系统层次结构，研究了基于嵌入式 Linux 网络计算机的操作系统的实现方法。

参考文献

[1]林涛.嵌入式操作系统 WindowsCE 的研究[J].微计算机信息,2006,6-2:91-93
[2]吴明晖.基于 ARM 的嵌入式系统开发与应用.人民邮电出版社,2004,6.
[3]张绮文等.ARM 嵌入式常用模块与综合系统设计. 电子工业出版社,2007,1.
[4]姜波.Windows CE.Net 程序设计.机械工业出版,2007,1.

作者简介: 杨丽萍(1976-), 女,汉族,出生于吉林省松原市,硕士研究生,讲师, 主要研究方向:计算机应用、人工智能与模式识别方面研究。王自力(1966-),男,汉族,吉林省长春市人,硕士研究生,主要研究方向 数据库与神经网络。

Biography: YANG Li-ping (1976-), Female, Han Nationality, Born in Songyuan City Jilin Province, Master degree, Lecturer, Main Research Area: Computer application, artificial intelligence, and pattern recognition.

(130022 吉林长春 长春大学计算机科学技术学院) 杨丽萍
(130012 吉林长春 吉林大学中日联谊医院信息中心) 王自力
通讯地址: (130022 吉林长春市卫星路 6543 号长春大学计算机科学技术学院) 杨丽萍

(收稿日期:2009.05.03)(修稿日期:2009.06.05)

技术创新