Windows CE 的 CAN 总线驱动程序设计※

■北京交通大学　彭少武　宋娟　王立德

摘　要　介绍一种基于 S3C2410 的 Windows CE 扩展 CAN 总线接口的方法。利用 SPI 接口扩展 CAN 总线接口，编写 CAN 流接口驱动程序；给出硬件接口设计原理图，重点阐述 WinCE 下独立 CAN 控制器 MCP2510 的驱动程序的实现，并给出软件处理流程。

关键词　CAN 总线　S3C2410　SPI 接口　流接口驱动程序　MCP2510　Windows CE

ARM 芯片 S3C2410 是 Samsung 公司生产的一种高集成度、高性价比的嵌入式处理芯片，已成功用于工控设备上。然而，其美中不足的是没有集成 CAN 控制器，使其在工控产品的应用中受到了一定的限制。

目前，国内广泛应用的 Philips 公司的独立 CAN 控制器 SJA1000 存在一定的缺陷，如地址、数据总线的分时复用常导致接口效率低下；接收、发送缓冲区的个数太少，导致数据吞吐率不高；帧屏蔽和过滤器的设置不够灵活，不能满足同时需要更多屏蔽和过滤条件的要求等。

Microchip 公司生产的 MCP2510 是一种带有 SPI 接口的独立 CAN 控制器，支持 CAN 技术规范V2.0A/B，能够发送和接收标准的和扩展的信息帧，同时具有接收滤波和信息管理的功能。MCP2510 通过 SPI 接口与 MCU 进行数据传输，最高数据传输速率可达 5 Mb/s。采用 MCP2510 给 S3C2410 扩展 CAN 接口，可以降低硬件电路的复杂性，保证 CAN 总线通信的稳定性，并提高效率。下面以自行开发的人机界面（HMI，Human Machine Interface）中 CAN 总线通信接口设计为例进行说明。

1　系统硬件结构

SPI 接口协议是 Motorola 公司推出的一种使用时钟和 2 根数据线传输数据的同步串行协议。MCP2510 将在一个 CS 的下降沿选用一种 SPI 模式。在 SPI 模式 0-0 时，命令和数据在 SCK 的上升沿通过 SI 脚送入 MCP2510，在 SCK 的下降沿通过 SO 脚送出数据。当执行这些操作时，CS 引脚应始终保持在低电平。CS 引脚被置低后送进的第一个字节的数据就是命令字，紧跟命令字后的是地址和数据。本系统的相关硬件部分为 S3C2410 和 CAN 控制器芯片 MCP2510 的连接与实现。接口的硬件连接如图 1 所示。MCP2510 作为 S3C2410 的一个从设备，可以将 S3C2410 的 SPI 接口直接接在 MCP2510 的 SPI 接口上。具体接法是：S3C2410 的接收引脚 MISO 接 MCP2510 的发送引脚 SO，S3C2410 的发送引脚 MOSI 接的是 MCP2510 的 SI 脚；同步时钟 SPICLK 由 S3C2410 提供；MCP2510 的片选信号 CS 由 S3C2410 的标准 I/O 口 GPG2 控制，GPG2 置低时选通 MCP2510，反之则不选通；中断信号由 MCP2510 发给 S3C2410；MCP2510 的复位引脚 RESET 与 S3C2410 相连，且接在上电复位和按键复位上；MCP2510 出来的

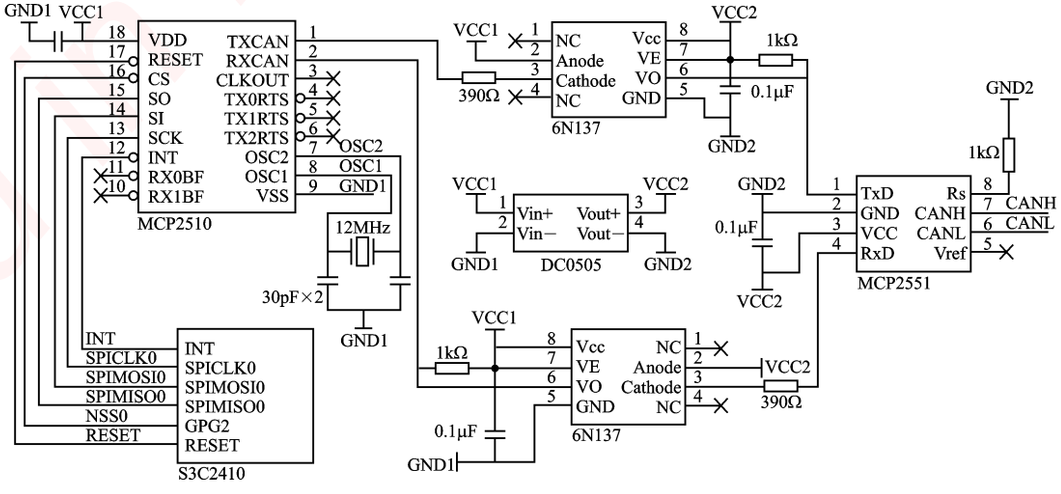

图 1　硬件电路原理

CAN 收与 CAN 发信号和 CAN 总线收发器 MCP2551 接在一起,构成一个完整的 CAN 总线收发模块。

DC0505 是一个 5 V —5 V 的隔离电源,提供给光耦 6N137 和 CAN 收发器 MCP2551,使系统更加稳定、可靠。

2 软件实现

Windows CE 把设备驱动程序分成 3 类[1]:本机设备驱动程序、总线驱动程序、流式接口驱动程序。

①本机设备驱动程序(native driver)也叫作"Build in 驱动程序",是硬件所必需的,通常由 OEM 设计硬件时完成,例如键盘驱动程序、触摸屏驱动程序以及音频驱动程序等等。该类型的驱动程序有时不支持通用的设备驱动程序接口,而且用户不能对其接口进行扩展,所以当有新版本的操作系统发行时,这些驱动程序就要被修改。

②总线驱动程序(bus driver)用来管理物理总线,它通过询问总线上的硬件设备来装载适当的驱动程序,如 PCI 总线。直接调用 ActiveDeviceEx 函数以加载设备驱动程序。但是所加载的设备驱动程序可能通过另一个设备驱动程序来间接管理硬件;同时,总线驱动程序决定哪些额外的驱动程序也要被装载进来,以及按照怎样的顺序来进行装载。

③流式接口驱动程序(stream interface driver)是指驱动程序把流式接口函数公开出来,而不考虑驱动所控制的设备类型。典型的流式接口驱动有:文件系统驱动(iostream、fstream),COM,LPT 等。

本系统采用流接口方式编写驱动程序,主要涉及 3 部分:本机设备驱动程序、导出流接口函数和修改注册表部分,如图 2 所示。

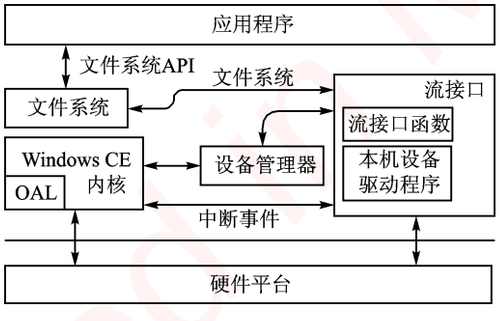

图 2 CAN 总线流接口驱动程序框图

2.1 软件处理流程

软件的实现,首先要确保 S3C2410 的 SPI 口可以正常收发,然后才可以利用 SPI 口对 MCP2510 的寄存器进行设置。所以第 1 步是对 S3C2410 的 SPI 口初始化,之后用 SPI 口对 MCP2510 进行初始化;第 2 步开始编写 CAN 控制器的收发程序;第 3 步开始写出 CAN 流接口函数形式;最后导出流接口,修改注册表和 CEC 文件。

图 3 只举例说明所有模块初始化部分。整个模块分为 3 部分初始化。

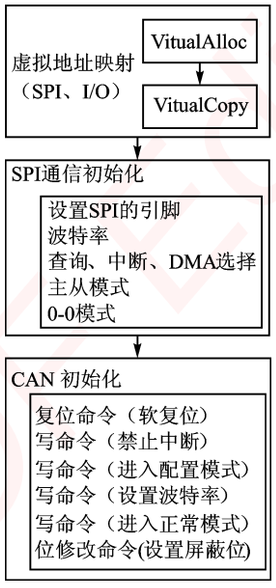

图 3 模块初始化

2.2 本机设备驱动程序

2.2.1 虚拟地址映射

由于 Windows CE 的特点,在操作系统之上要想让应用程序通过 OEM 层到达实际的硬件进行控制,必须进行虚拟内存分配和虚拟地址映射,这与单片机形式的直接控制硬件有很大的不同。管理虚拟内存的硬件是内存管理单元 MMU(Memory Management Unit)。MMU 负责把虚拟地址映射到物理地址,并且提供一定的内存保护。所以,整个驱动的首要工作是解决如何让操作系统之上的软件接触到底层硬件。

首先,调用 VitualAlloc 函数(最底层的分配虚拟地址空间的函数),如 v_pSSPregs = (volatile SSPreg *) VirtualAlloc(0, sizeof(SSPreg), MEM_RESERVE, PAGE_NOACCESS)。在 Samsung 公司提供的 BSP 的头文件 S2410.H 中,SSPreg 表示 SPI 寄存器的结构体。此语句分配一块 SSPreg 大小的空间,并将首地址返回给 v_pSSPregs,供后续调用。

之后,用 VitualCopy 函数,将 S3C2410 中的 SPI 寄存器结构体基地址 SSP_BASE 映射到刚刚分配的虚拟地址空间中,如 VirtualCopy((PVOID) v_pSSPregs,(PVOID) SSP_BASE,sizeof(SSPreg),PAGE_READWRITE| PAGE_NOCACHE))。此时,便可给 v_pSSPregs 赋值,以修改 SPI 寄存器,正常启动 SPI 通信。

2.2.2 SPI 初始化

虚拟地址映射成功后,开始设置与 SPI 功能相关的 S3C2410 的 4 个多功能复用引脚,分别为 SPISCK(SPI 同步时钟)、SPIMOSI(主入从出)、SPIMISO(主出从入)和 GPG2(片选 CS);接下来设置 SPI 波特率、主模式、查询方式、提供时钟、0-0 模式。

SPI 正常工作后,就可通过 SPI 对 MCP2510 的寄存器进行设置。归结起来,对 SPI 的操作仅需两条基本命令就可以组合出来,就是读命令和写命令。写命令为:

```
unsigned char Spi_Write(unsigned char Data){
    v_pSSPregs->rSPTDAT0 = Data;
      //向 SPI 发送缓冲器写入待发内容
```

```
    while(!(v_pSSPregs->rSPSTA0 & 1));
        //查询判断 SPI 状态,是否发完
    return (UINT8)v_pSSPregs->rSPRDAT0;
        //返回接收的数据
}
```

写命令中包含了读取 SPI 总线接收缓冲器,所以 SPI 读命令内容,只需用写命令向要查询的寄存器写入 0xFF(SPI 正常模式),就可以读出该寄存器的内容。

2.2.3 CAN 初始化

根据厂家 MCP2510 芯片的技术规定,采用以下操作步骤对芯片进行初始化。先发复位命令(软复位),接下来分别是写命令(禁止中断)、写命令(进入配置模式)、写命令(设置波特率)、写命令(进入正常模式)和位修改命令(设置屏蔽位)。

由于篇幅关系,现仅对最常用的写命令进行解释。如向 MCP2510 指定地址写入一个字节为:

```
void 2510_Write(unsigned char address,unsigned char value ) {
    2510_CS_L;
    Spi_Write(2510INSTRU_WRITE);
    Spi_Write(address);
    Spi_Write(value);
    2510_CS_H;
}
```

2510_CS_L 和 2510_CS_H 分别为 CS 引脚的置低和置高,这是对 MCP2510 进行操作的开始和结束的必不可少的环节。参考 Microchip 公司的 MCP2510 芯片手册,对其寄存器操作必须先写入指令操作码(操作方式),然后写入寄存器地址和内容。所以这里用 SPI 向 MCP2510 先写入指令写操作码 MCP2510INSTRU_WRITE(头文件中定义为 0X02),然后是地址 address 和内容 value,这样一个写命令就完成了。其他命令类似。

2.3 CAN 收发实现

针对 CAN 总线协议[2]和 MCP2510[3]的芯片特性 CAN 的收发函数都要包括 ID 和数据。下面对 CAN 总线发送数据函数举例展开:

```
void 2510_Can_Write( UINT8 nbuf, int IsExt, UINT32 id,int rxRTR, UINT8 * data,UINT8 dlc ) {
    UINT8 mcp_addr = (nbuf<<4) + 0x31;
    2510_Swrite(mcp_addr+5,data,dlc);
    2510_Write_ID(mcp_addr, id, IsExt);
    if (rxRTR)    dlc| = RTR_MASK;
    2510_Write((mcp_addr+4),dlc);
}
```

MCP2510 有 3 个 TXBUF(发送缓冲区),参数 nbuf 为使用第几个发送缓冲区;IsExt 表示是否是扩展总线(29 位 ID),id 为写入的 ID 值;rxRTR 表示是否远程帧;data 表示要写入的数据;dlc 表示数据长度。

编者注:驱动程序源代码见本刊网站 www.mesnet.com.cn。

2.4 导出流接口函数

流接口驱动程序实现 CAN_Init()、CAN_IOControl()以及 CAN_PowerUp()等一组标准的函数,用来完成标准的文件 I/O 函数和电源管理等,要生成一个 DLL。

在对设备进行读操作之前,首先要通过执行 CreatFile()函数来调用 CAN_Open()打开设备。CAN_Open()所需的第 1 个参数是应用程序初始化时由 CAN_Init()返回的设备句柄;而 CAN_Read()需要的第 1 个参数是 CreatFile()执行成功后返回的驱动引用实例句柄。第 2 个和第 3 个参数分别是用于从驱动中读数据的缓冲区地址和长度。应用程序通过 ReadFile()函数来调用 CAN_Read()。

最后建立一个"MCP2510.def"的文件,将 DLL 中的函数输出,并将此文件添加到流接口驱动程序的工程里面。

2.5 修改注册表

具体的流接口驱动程序跟注册表是分不开的,写一个注册表文件通过添加组件的方式添加到 Windows CE 内核中。下面为注册表修改处内容:[HKEY_LOCAL_MACHINE\Drivers\BuildIn\MCP2510]

```
"Index" = dword:1
"Prefix" = "CAN"
"Dll" = ?'CanDriver.dll"
"Order" = dword:0
```

3 结 论

本设计采用 ARM 芯片 S3C2410 集成的 SPI 接口扩展 CAN 总线接口,在不改变 CAN 总线自身的特点的前提下,使得系统硬件设计简单,增强了系统的可靠性。实现扩展的 CAN 总线接口通过回环测试和联网测试,效果良好,证明了该方案的正确性和可靠性。

参考文献

[1] 周毓林. Windows CE.NET 内核定制及应用开发[M]. 北京:电子工业出版社,2005.

[2] 邬宽明. CAN 总线原理和应用系统设计[M]. 北京:北京航空航天大学出版社,2001.

[3] Microsoft 公司. Microsoft Windows 驱动程序模型设计[M]. 北京:北京大学出版社,2000.

(收修改稿日期:2007-06-20)