·电路与控制·

基于 DOS 平台的多串口通讯的实现

宋刈非,张 蕊

(东北电子技术研究所,辽宁 锦州 121000)

摘 要:DOS 系统由于技术成熟等优点在嵌入式操作系统领域始终占据着重要的地位,因此大部分的嵌入式计算机的程序设计都是在 DOS 平台上完成的.介绍了如何通过直接对 INS8250 的内部寄存器操作来实现串行通讯,以及多串口通讯在 DOS 平台上的中断服务函数的程序设计,并解决了多串口通讯的高号中断问题.

关键词:多串口;串行通讯;中断

中图分类号:TP273^{+}.5 **文献标识码**:A **文章编号**:1673-1255(2009)01-0062-04

Implementation of Multi-serial Communication Based on DOS

SONG Yi-fei, ZHANG Rui

(*Northeast Research Institute of Electronics Technology*, *Jinzhou* 121000, *China*)

Abstract: For one of virtues is the mature technique, the DOS system takes an important place in the embedded operation system all the time. So design for most of the embedded computer programs are completed on the DOS platform. How to implement the multi-serial communication in the way of operating directly INS8250's register and the program design of IRQ function on the DOS platform are introduced. And the issue of high-order IRQ in the multi-serial communication is resolved.

Key words: multi-serial; serial communication; IRQ

目前,单片机和嵌入式计算机越来越广泛的应用于工业控制、自动化生产设备的数据采集等领域,但通常一般的 PC 机只配有 2 个串口,在控制领域有时候是不够的,而基于 PC/104 等结构的嵌入式系统可以根据实际的要求通过简单的搭积木的方法进行配置,当进行多串口控制时可以选用基于 PC/104 的多串口卡.

DOS 操作系统以其内核小、操作简单、技术成熟等优点使其在嵌入式操作系统领域占据着重要的地位,因此大部分的嵌入式计算机的程序设计都是在 DOS 平台上完成的.而在 DOS 操作系统下,不能像 Windows 操作系统一样有 API(应用编程接口),只能是直接操作硬件端口(异步通信适配器 8250)来实现编程.

1 PC 机异步通信适配器 8250 及其编程操作

1.1 INS8250 内部寄存器及其选择方式

PC 机主板上负责串行通信的核心器件为 8250(或器件冗余器件)异步通信适配器(UART).程序通过对 8250 内部的寄存器进行读写进而控制 8250.表 1 为 8250 内部寄存器的定义和描述.

从表 1 中可以看出,INS8250 借助线控制寄存器的最高位 $D7$,通过 3 条寄存器选通线 $A2A1A0$,实现对寄存器进行寻址.当 $D7$ 为 0 时,寄存器 0 和 1 分别为接收/发送数据寄存器和中断允许寄存器;当 $D7$ 为 1 时,则分别为低 8 位除数锁存器 LSB 和

收稿日期:2008-12-23

作者简介:宋刈非(1981-),男,辽宁锦州人,学士,主要研究方向为自动控制.

表 1　8250 内部寄存器

寄存器	缩写	偏移	COM1	COM2	*A*2*A*1*A*0	*D*7
接收数据寄存器	RXR	0	3F8H	2F8H	000	
发送数据寄存器	TXR	0	3F8H	2F8H	000	0
中断允许寄存器	IER	1	3F9H	2F9H	001	
中断标识寄存器	IIR	2	3FAH	2FAH	010	
线路控制寄存器	LCR	3	3FBH	2FBH	011	
MODEM 控制寄存器	MCR	4	3FCH	2FCH	100	×
线路状态寄存器	LSR	5	3FDH	2FDH	101	
MODEM 状态寄存器	MSR	6	3FEH	2FEH	110	
波特率除数锁存器低	LSB	0	3F8H	2F8H	000	1
波特率除数锁存器高	MSB	1	3F9H	2F9H	001	

高 8 位除数锁存器 MSB. *D*7 也通常被称为除数锁存器访问位 DLAB.

1.2　中断 I/O 通信方式相关设置

当采用中断 I/O 方式时,通过对中断允许寄存器的低 4 位操作,可产生 UART 支持的 4 类中断. 由前述内容已知,当线路控制寄存器 *D*7 位为 0 时,寄存器 1 即为中断允许寄存器,其各位定义如下:

*D*7～*D*4:保留,恒为 0;

*D*3:该位为 1 表示允许 MODEM 状态变化中断,即在任何一个计算机的 RS－232C 输入端改变状态时,产生一个中断;

*D*2:该位为 1 时,表示当接收到有错信息(包括奇偶校验错、超越错、帧格式错)或间断条件时,允许接收器产生中断;

*D*1:该位为 1 时,表示允许发送保持寄存器空中断,即当从发送器的保持寄存器中移一个字节到发送器的移位寄存器时,产生一个中断;

*D*0:该位为 1 时,表示允许接收器数据就绪中断,即当接收器的缓冲寄存器中有待读的有效字节时,产生一个中断.

若要禁止某类中断,只需将相应位置 0 即可. 对于采用 I/O 通信方式的寄存器,应将中断允许寄存器置 0.

如何来确定中断的优先级呢? 中断标识寄存器就是用来保持优先级最高的中断请求的标识码的.

在 CPU 处理这个特定的中断请求之前,不接受其他的中断请求. 因此,在中断服务程序内部必须通过读取中断标识寄存器才能识别出确切的中断源,然后转入相应的中断处理子程序. 中断标识寄存器是只读寄存器,*D*7～*D*3 恒为 0,其余各位如表 2.

表 2　中断标识寄存器

*D*2*D*1*D*0	优先级	中断源类型	中断源	寄存器复位
001			无	
110	1(最高)	接收有错或间断条件	奇偶校验错、超越错、帧格式错、间断条件	读线路状态寄存器
100	2	接收的数据就绪	接收数据就绪	读接收缓冲寄存器
010	3	发送保持寄存器空	发送数据就绪	写发送保护寄存器或读中断标识寄存器
000	4	MODEM 状态变化	输入状态变化	读 MODEM 状态寄存器

2　多串口通讯软件的设计

软件采用 C 语言编程,工作于 DOS6.22 平台系统下,可读性好,能够方便移植到其他平台. 程序流程见图 1.

此处省略了初始化串口的部分,介绍了一个可重用的 DOS 下多串口通信程序的中断服务函数.

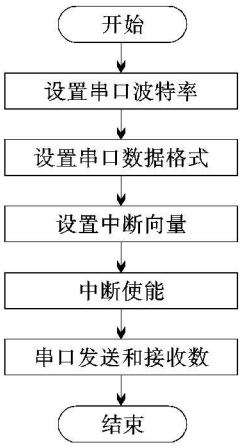

图 1 程序流程图

```
void interrupt ComIntServ-comport(...)
{
int temp;
disable();
temp = (inportb(comport1.PortBase + IIR)) & IIR-MASK;
//中断响应
switch(temp)
{  case 0x00: // modem status changed
     inportb(comport.PortBase + MSR);//read in useless char
     break;
  case 0x02: // Request To Send char
     if (comport.outhead ! = comport.outtail)//有需要发送
的送字符
     outportb(comport.PortBase + TXR,comport.outbuf[com-
port.outhead + + ]);//发送字符
     if (comport.outhead = = OBUF-LEN)
     comport.outhead = 0; //如果在缓冲区尾,重置指针
     break;
case 0x04:  //准备读入字符
     comport.inbuf[comport.inhead] = inportb(comport.
PortBase + RXR);//读入缓冲区
     comport.inhead + + ;
     if (comport.inhead = = IBUF-LEN) //如果在缓冲区尾
     comport.inhead = 0;                    //重置指针
     break;
case 0x06: //line status has changed
     inportb(comport.PortBase + LSR);  //read in useless char
     break;
default:
     break;
}
outportb(PIC8259-ICR, PIC8259-EOI); //中断结束
enable();          //中断使能
}
short int ReadChar-comport(void) //读缓冲区函数,为多串口
设计
{
unsigned char ch;
if (comport.inhead ! = comport.intail) //有字符
{
     disable();              //取字符时禁止中断
     ch = comport.inbuf[comport.intail + + ]; //从缓冲区
取字符
     if (comport.intail = = IBUF-LEN)  //如果在缓冲区尾
     comport.intail = 0;     // 重置指针
     enable();     // 中断使能
     return(ch);
}       // return the char
ch = 0x100;
return(ch);
}     // return nothing
```

在多串口通讯程序中,由于各串口接收数据时,处理代码会有所不同,会有各自存放数据的变量,这些都需要在自己的中断服务程序中完成.定义好各串口对象后,只要将中断服务函数的入口地址指针送到相应的中断号中即可.

3 高号中断 8259A 可编程中断控制器的控制

有时在处理多串口时,要对中断进行扩展,很典型的实例就是在 PC104 中对多串口的控制,PC104 与 IBMPC 是兼容的,但留了多余的中断号供编程者使用.

在 IBMPC 及其兼容机中,通过 CPU 的 NMI(非屏蔽中断)和 2 个 8259A 可编程中断控制器芯片为系统提供了 16 级中断,硬件中断结构如图 2 所示,2 片 8259A 构成主从式级联控制结构,与 CPU 相连的称为主片,下一层的称为从片,从片中断请求信号 INT 与主片的 IRQ2 相连.

中断初始化编程时,当用主片中 *IRQ*0～*IRQ*7 时,只须在屏蔽寄存器中打开相应中断,在中断服务程序中,中断结束后,发一次中断结束命令 EOI;而涉及从片中 *IRQ*7～*IRQ*15 高号中断时,除在从片中的屏蔽寄存器中打开相对应的中断,还须打开主

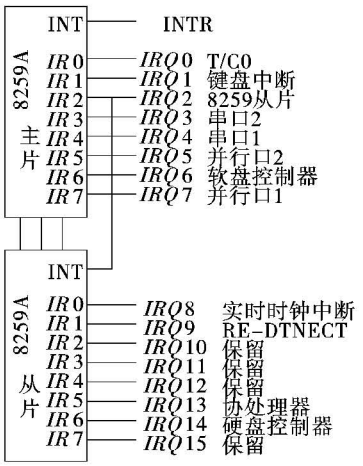

图 2　PC 机硬件中断结构

片中的 *IRQ*2,且在中断服务程序中中断结束时,要发 2 次 EOI 命令,分别使主片和从片执行中断结束命令.下面就 *IRQ*11 的初始化和中断服务程序处理给出 Turbo C 的源代码编程说明.

首先说明一个中断指针 oldvect 以保存原来的中断向量,在中断服务程序 ser-program()结束后,分别向主片和从片的中断控制寄存器 ICR 送中断结束信号 EOI.

```
    void interrupt( * oldvect)(...); //设置原中断向量保存指针
void interrupt ser-program(...)
{ {... ...} //中断服务程序代码
outportb(0xA0,0x20); //向从片 ICR 送 EOI 命令
outportb(0x20,0x20); } //向主片 ICR 送 EOI 命令
```

中断初始化时要先保存 *IRQ*11 对应的地址 73H 存储的原中断向量,然后将自己的中断服务程序入口地址装入,再分别打开主片 *IRQ*2 和从片 *IRQ*11.

```
    void Interrupt-Enable(void)
{int temp;
{... ...} //其他初始化代码
oldvect = getvect(0x73); //保存原中断向量
setvect(0x73,ser-program); //装入中断服务程序入口地址
temp = inportb(0x21) & 0xFB;//打开主片 IRQ2
outportb(0x21, temp);
temp = inportb(0xA1) & 0xF7;//打开从片 IRQ11
outportb(0xA1, temp); }
```

最后,不要忘记在程序关闭前关中断和恢复原中断向量.

```
    void Interrupt-Disable(void)
{ int temp;
setvect(0x73, oldvect); //恢复原中断向量
temp = inportb(0x21) | ~(0xFB);//关主片 IRQ2
outportb(0x21, temp);
temp = inportb(0xA1) | ~(0xF7);//关从片 IRQ11
outportb(0xA1, temp);}
```

4　应注意的问题

在实际应用时,应该考虑以下问题:

(1)设置中断向量,以保证 CPU 执行中断服务程序时有一个正确的入口地址.可使用 C 语言的库函数 setvect()和 getvect()对中断向量进行设置和读取.

(2)中断服务程序什么时候结束,CPU 无法知道,8259 中断控制器也无法知道,必须通过指令通知 8259 中断控制器.对于 PC 机,这条指令就是向端口地址 20H 写入一个命令字 20H.

(3)为保证通讯的可靠性以及不耽误 CPU 响应其他中断,中断服务程序应尽量短,一些费时的操作如数据存盘等应放在中断服务程序以外.

(4)在把一个数据写入发送保持寄存器(THR)之前,必须检查线路状态寄存器 LSR,以保证发送保持寄存器为空.

(5)初始化中断系统时,应该关闭中断,初始化完成后再开放.可使用 C 语言的库函数 disable()和 enable()来完成.

5　结　　论

较全面地介绍了 DOS 平台下多串口通讯的软、硬件的原理和过程,提出的方案确保了中断请求的无漏检测和服务,并有效解决了多个串口共享同一中断源时所造成的冲突和丢失等问题.由于已成功应用于实际项目中,表明方案是可行、有效的.

参考文献

[1]　徐海燕,付炎.嵌入式系统技术与应用[M].北京:机械工业出版社,2002:7－18.

[2]　龚建伟,熊光明.串口通信编程实践[M].北京:电子工业出版社,2005:169－195.

[3]　罗利. 基于中断方式 PC104 高速串行通讯软件[J].电子测量技术,2004(4):110－112.