Windows 3.1 下的串行通讯系统

戴志远 张雪松 (暨南大学)

摘要:本文通过计算机的串行通信口和 *MODEM*,在 *Windows 3.1* 下实现了串行通讯系统的设计,充分利用了 *Windows* 的图形界面优点。系统具有实时响应能力和多任务机制,并且功能齐全。

一、引 言

目前微机的通讯系统大体可分为两类: 一类是 DOS 下的单任务型通讯系统; 二是 Windows 及其他操作系统之下的多任务型通讯系统。前者在进行通讯时, 要独占计算机的所有资源,用户不能进行其他的操作。而且对通讯事件要有预知性, 也就是说通讯时必须先告知接收方,使其开启系统,然后才能进行通讯。显然这种系统使用起来极不方便。而系统的设计一般采用循环等待的方式处理,既不好理解又容易出错。比较好的是后者,现在 Windows 下的通讯系统已有不少产品。比如 Windows 自带的终端仿真程序,符合多任务的要求,但是使用起来却不方便,尤其对于一个非专业人员,它的参数设置,操作方法都甚为不便。另外实际中可能用到各种各样的通讯协议, 而通讯事件又具有极强的随机性和偶然性,所以我们开发了在 Windows 3.1 下的串行通讯系统。这个通讯系统的开发目标是:实用性与易用性;多任务机制;实时响应能力; 实际要求变化时的适应性和易于维护性。在 Windows 3.1 工作平台下, 该通讯系统与其他应用程序可以并发运行,并且能对通讯事件进行实时响应处理。在以下的叙述中,把通讯系统接收或发送的文件(数据)统一称为邮件。整个系统的结构如图 1 所示。

二、通信系统的功能

系统的主要功能包括:

1.文本文件服务功能。2.电话簿功能。3.系统参数配置功能。4.发送邮件功能。可自动或人工拨号发送。5.自动接收邮件功能.可同时接收多路邮件。6. 接收邮件记录查寻功能。

其中功能 1、2、3、6 是由系统管理层完成的,功能 4 和 5 是由通讯内核模块完成的; 管理模块在前台运行,通讯内核模块在后台独立运行,二者之间使用动态数据交换机制信息。在通讯内核模块中,安装了系统定时器,该模块具有错误时的重发功能,定时发送功能,以及错误时的自启动功能。

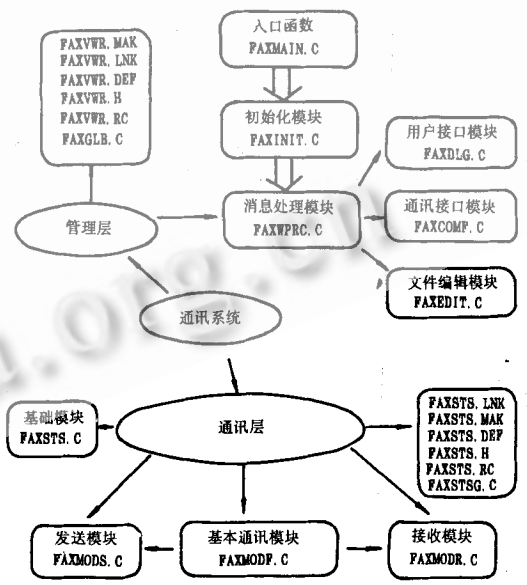

图 *1* 系统总体结构

本系统各部分功能的实现都是用 C 语言在 Windows 环境中按所有标准规范而实现的, 因而具备严格的 Windows 多任务机制。在处理任一功能模块时,都是在整个 Windows 系统的控制下进行,不会影响或干扰 Windows 系统的正常运行以及其它系统的运行。

1.系统的基本功能

文本文件服务功能,包括文本文件的打开、编辑、保

存、改名、关闭、新建等功能。电话簿功能、包括电话本的制作及存储,电话号码的输入与存取,电话本的改名,以及电话号码的编辑修改功能。

系统参数配置功能,当使用的硬件平台变化时,或者初始安装系统时,可利用参数配置功能为系统配置参数,以达到可移植的目的。主要有用户资料配置,MODEM 配置,FAX 配置。

2.通讯功能

包括接收、发送、查寻邮件功能。完成 MODEM 的初始化之后,只要选择好要发送对象的电话号码,以及待发邮件,则系统自动拨号,自动应答,自动发送。完成后进入等待状态。这就是发送邮件功能。也可以人工拨号,然后由系统自动发送。

自动接收功能,在初始化完成之后,就处于自动接收状态。如果有邮件发送过来,对应串口的 MODEM 自动应答,自动接收,并将有关信息自动入档,供以后检索之用。若一台计算机配有多个 MODEM 卡,则各卡可互相独立而又互不干扰地同时接收各自的邮件。查寻邮件功能,在任何时候可以查看过去接收到的邮件的记录,按时间先后顺序,最近接收到的在最前面。提供的信息有接收的时间、文件名、发送方的标识(一般为电话号码)。

三、管理层设计

管理层的主要功能包括:标准的文本文件服务,标准的编辑功能,电话簿功能,以及串行通讯口的一些初始工作。它是一个标准的 Windows 窗口,在此窗口下,可以给通讯层发送初始化,发送邮件,或接受邮件的命令。并且可以对串行口和 MODEM 的参数进行设置。本系统管理部分由 FAXVWR.EXE 完成,限于篇幅只讨论其中的通讯模块。

通讯模块由 FAXCOMF.C 完成。它包括一些基本的通讯处理函数,主要用于和通讯内核模块交换信息。

打开通讯内核模块函数 OpenStsWnd()初始化通讯内核模块,把通讯内核模块的一个实例赋给某个指定的串口,以后该串口的所有工作都是在这一个实例下进行。通讯内核模块可以有多个实例,分别对应不同的串口。而这些实例是在 Windows 系统的统一管理下进行的,它们共用一个代码段,但有各自的数据段。它们之间互相独立地运行,完全符合 Windows 的编程要求和多任务机制。且可对各自的串口事件独立地作出实时响应。

初始化通讯口函数 InitCommPort()是初始化通讯口的入口,,之后调用函数 SendMessage()通知通讯内核模块作初始化的具体工作。

发送邮件控制函数 SendModemFile()是发送的入口,首先作一些必要的工作,之后用函数 SendMessage()通知相应的通讯内核模块作发送文件的具体工作。

FAXVWR.EXE 模块用 SendMessage()函数直接把消息邮送到 FAXSTS 模块的窗口处理函数,并等到处理完成后再返回;而 PostMessage()函数则把消息邮送到 FAXSTS.EXE 的消息队列中,然后立即返回。

该模块的部分程序原语可简单地表示如下:

```
if (已有对应的通讯内模块运行)
    {转到此实例下运行};
else
    { 初始化对应的通讯内核模块};
if(初始化通讯口)
    {作标记,通知通讯内核模块初始化};
if(发送邮件)
    {作标记,通知通讯内核模块发送邮件};
```

四、通讯层设计

通讯层的功能主要是电子邮件的接收与发送工作。通讯层由 FAXSTS.EXE 完成。

1.通讯层的基本模块

基本模块由 FAXSTS.C 完成。该模块负责通讯窗口的窗口类注册、产生及显示。并且从 Windows 系统读取消息,分发到自己的消息处理函数。这几项功能与一般 Windows 应用程序的主函数和初始化函数类似,详细情况可参见有关书籍。

FAXSTS.C 最主要的工作是接收管理层发来的命令并采取相应的措施(管理层发来的命令有 WM COM 标志)。它还是发送和接收邮件的管理调度部分。

有一个关键的细节要仔细解释一个。应用程序是如何与串行通讯口打交道的?首先应用程序要通过 Windows 系统函数打开串行口,设置串行口的参数,在串行口成功地打开之后,就可以与它对话了。这时如果应用程序想与它交换信息,要用函数 Enable Comm Notification()向 Windows 系统注册,当串行口事件发生时,Windows 系统会根据注册的情况向有关的窗口发送

通知消息 WM_COMMNOTIFY 以及附加的详细信息,这样应用程序就可以根据消息的不同作不同的处理。本程序把收到的串行口消息发送给主控制分流函数 ModemCommPort()作统一处理。

从管理层接收到的有关通讯的消息有:

(1)初始化命令 COM_INIT,此时 FAXSTS.C 先作初始化的标记,然后把接收到的参数传送给初始化函数 InitCommPort(),由它完成初始化的具体工作。

(2)发送命令 COM_SEND,此时 FAXSTS.C 先作发送的标记,然后把接收到的参数传送给发送函数,由它完成发送的具体工作。

(3)接收命令 COM_RECV,此时 FAXSTS.C 先作接收的标记,然后把接收到的参数传送给接收函数,由它完成接收的具体工作。

(4)标题赋予命令 COM_TITLE.此时 FAXSTS.C 把该实例赋给指定的串行口,改变窗口的标题,以后就由该实例负责这个串行口的所有工作。

(5)关闭命令 COM_SHUTDOWN,此时 FAXSTS.C 关闭串行口。若想再使用串行口就必须重新初始化。

另外当用户选择关闭(CLOSE)窗口时,FAXSTS.C 会提示用户是否确实想关闭,如果用户回答关闭,则关闭串行口,然后销毁窗口。否则对关闭命令不予理采。

2.通讯层的通讯模块

该通讯模块由 FAXMODF.C,FAXMODR. C,FAXMODS. C 构成。下面分别讨论。

(1)FAXMODF.C

该模块包括通讯层的一些基本功能模块,比如通讯口的初始化,以及发送接收邮件前后的一些通用操作。

①通讯通知消息分流功能。ModemComPort()函数完成通知消息的分流功能。它根据过程标志(idNotify)来判断通讯口所处的过程(初始化,发送,接收),从而转向不同过程的处理函数。见下面的程序原语和流程图。

本程序中采用的是 Windows 系统的消息通知及处理机制,程序只在收到通知消息时才取得系统的控制权而进行处理,处理完成之后立刻把控制权交还给系统,因而完全符合 Windows 系统的多任务机制,而且可以对串行口事件作出实时响应,这也就达到了本文的目的。分流程序的流程图如图 2(A)。

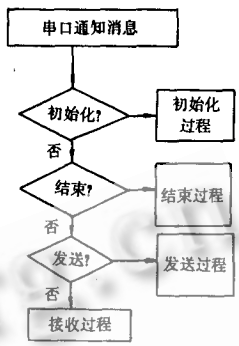

图 2 (A)分流程序的流程

②初始化功能。初始化通讯口函数 Init Comm Port()负责初始化串行口和 MODEM 的初始例程以及引导工作。根据给它的参数(哪一个串行口),先到一个初始化文件中取出这个串行口的所有参数,包括波特率、奇偶校验、数据位、停止位,然后用 OpenComm() 函数打开串行口。

打开串行口之后,用 FlushComm()函数刷新其发送和接收队列,用 SendToStatus() 函数显示信息,表明开始初始化。用 BukldCommDCB()函数建立数据结构,供系统使用,用函数 Set Comm State()真正设置串行口的参数。再用几个注册函数 Clear CommBreak(), Escape Comm Function()设置通讯口的状态,用 Enable Comm Notification()函数向系统注册,要求系统在串行口接收到数据时,给应用程序发送通知消息。然后作初始化的标记(idNotify)给串行口发第一个命令后,进入响应等待状态。

当串行口接收到 MODEM 的响应后,Windows 系统就给应用程序发送 WM_COMMNOTIFY 消息。经过 ModemCommPort()函数的分流,进入初始化过程函数。InitModemProc(),它判断上一个命令的执行情况:成功则执行下一个命令直到初始化过程全部完成,错误则重复执行直到超过设定的次数或时间而退出。初始化成功之后,就自动设置接收标记而进入接收状态。并且给管理层发送初始化成功的消息。参见图 2(B)。

(2)FAXMODR.C

该模块是通讯系统的接收模块,它负责接收邮件的主要工作。以下主要讨论接收流程和控制分流部分。

当设置了接收标记时,若有通讯口事件发生,则接收

控制分流函数 RecvModemControl()从主控制分流部分取得控制权。然后根据 idEvent 的标记分流到具体的某一个执行函数,每一个执行函数完成特定阶段的一些任务,并且设置相应阶段的标志,任务完成后就设置下一个阶段的标志。这样就可以保证各个阶段的自动衔接。

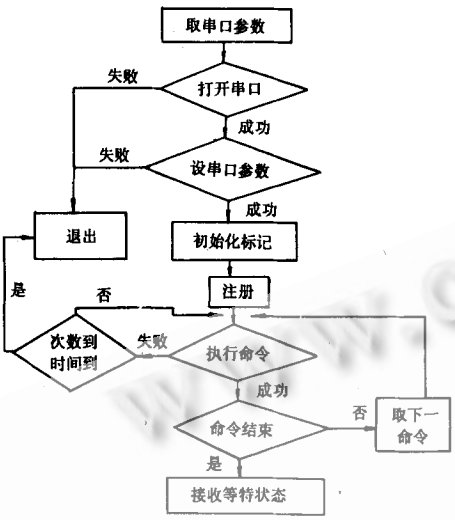

图 2 (B)初始化串行口的流程

idEvent 可设置的标记有:

①COM_RING。当 MODEM 处于等待接收状态时,设置这个标记。它使通讯内核模块检测电话铃声并应答。当和另一个 MODEM 立连接之后,设置下一阶段标记 COM_CONNECT。

② COM_CONNECT。当 MODEM 和另一 MODEM 进行协议交换时,设置此标记。当协议交换完毕可以交换数据时,设置下一阶段标记 COM_FAX。

③COM_FAX。当 MODEM 和另一 MODEM 进行数据交换时,设置此标记。当数据交换完毕时,设置下一阶段标记 COM_CONFIRM。

④ COM_CONFIRM。当数据交换完毕,两个 MODEM 进行结束过程的协商。主要是判断接收的数据是否有错或者是否接收完毕,以便采取相应的措施。

该模块程序原语如下:

```
switch(idEvent 事件标志){
  case COM_RING:
    检测振铃;应答;
    if (成功){idEvent=COM_CONNECT;};
    else{错误处理};
    break;
  case COM_CONNECT;
    协议交换;
    if(成功{idEvent=COM_FAX;};
    else{错误处理}
    break;
  case COM_FAX;
    数据交换;
    if(成功地交换完毕){idEvent=COM_CONFIRM;}
    else{错误处理};
    break;
  case COM_CONFIRM;
    结束过程;
    if(成功){挂机;初始化通讯口;等待接收;}
    else{错误处理};
};
```

(3)FAXMODS.C

该模块是通讯系统的发送模块,它负责发送邮件的主要工作。

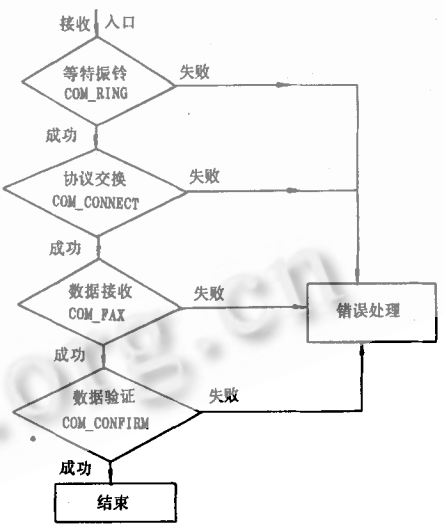

图 3 接收控制流程

当设置了发送标记时,若有通讯口事件发生,则发送控制分流函数 SendModemControl()从主控制分流部分流部分取得控制权。然后根据 idEvent 的标记分流到具体的某一个执行函数,每一个执行函数完成特定阶段的一些任务,并且设置相应阶段的标志,任务完成后就设置下一个阶段的标志。这样就可以保证各个阶段的自动衔接。控制流程图,程序原语,以及 idEvent 可设置的标记都与接收部分的类似,可参见有关部分。

另外还可以选择手动发送命令,进行手工拨号。这

在接收方使用分机的情况下特别有用。在选择要发送的邮件之后,手工拨号,拨通之后,选择相应状态窗口中的 Answer 菜单,则 Modem 自动进行下面的发送工作。

3.定时功能

定时功能对于一个通讯系统来说是极为重要的。为了确保系统安全有效地运行,在线路故障,通讯错误以及系统出现致命错误时,都要使用定时功能来进行错误处理。定时功能是在系统中安装一个定时器,并对定时器消息和通讯口消息进行综合判断处理。

每当成功地向通讯口写入数据之后,就打开定时器。若超时(一般为 3 秒)没有响应,则重发数据。三次重发之后没有响应,就认为系统出了致命错误,由第三次定时器消息重新启动系统。

每当从通讯口读取数据之前,就关闭定时器,以免定时消息误导系统的运行。所以通讯口消息的优先级高于定时消息的优先级,因而确保前者不会被后者屏蔽而误导系统。出现致命错误导致系统关闭后,打开定时器,在设定的时间(如 30 秒)以后再启动系统,这就是系统的自启动功能。同样道理可以进行延时发送。

定时功能的程序原语如下:

```
if (通讯口事件){
    if(成功发送数据){打开 3 秒定时器};
    else{错误处理};
    if (有数据读){关闭定时器,读数据};
    else{空闲};
}
if(定时器消息)}
    if(超时等于三次){关闭系统,打开自启动定时器};
    else{重复上次操作;超时累加};
}
```

结束语:本文的串行通讯系统,结构复杂,功能强大,可应用于实际。若针对不同的要求略作修改,则可以建立合乎要求的通讯系统。下文就是一个应用的实例。