电脑电话集成系统设计中的多路控制技术

余华云　　蔡莲红
湖北省荆州教育学院计算机管理中心　清华大学计算机科学与技术系

[摘　要] 本文着重介绍了电脑电话集成系统设计中的两种多线路同时控制技术以及该技术在实际开发应用中应注意的一些问题。
[关键词] 信息；电脑电话集成系统；多路控制技术。

Technology of Multi-circuit Control In Computer Telephone Integration System Design

YuHuaYun　Cailianhong
Computer Control Centre of Education Academy HuBei Jingzhou　434001
Department of Computer Science and Technology Tsinghua University
Beijing　100084

[Abstract] the paper emphasize introduce both of technology of multi-circuit control at the same time in computer telephone integration system design and the technology should attentive some problems in practical develop apply.
[Key words] information ; computer telephone integration system ; technique of multi-circuit control.

一、问题的提出

随着计算机技术和通信技术的不断发展以及电话机、传真机在办公室和家庭中逐渐普及，一种计算机技术与通信技术的结合产物——电脑电话集成系统(CTI)也开始逐步走入市场。电脑电话的兴起不仅使通信成本下降，通信质量大为改善，而且为用户提供了丰富的语音服务，并很大程度地实现了办公室自动化和个人通讯的自动化，提高了个人的工作效率。用户可以利用电话，通过语音卡与计算机进行交互，从而方便地得到各种服务，如语音留言、信息咨询、资料查询等等，因此受到了人们的普遍欢迎，其应用也日益广泛。下面是一种电脑电话集成系统应用中的系统结构图：

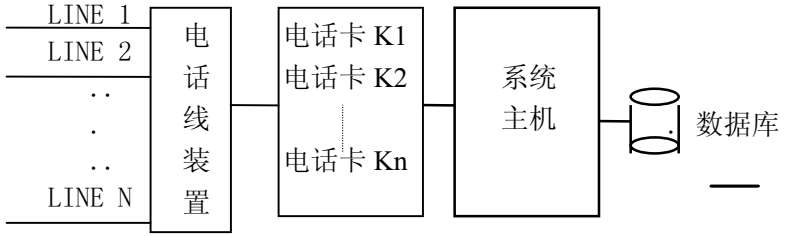

图 1

该系统可以实现语音信箱、交互式语音应答等多种功能。

从图中可以看出，在这个应用系统中，一个关键的问题就是如何同时控制多个信道。因为在系统的实际设计中，我们必须考虑有多个用户同时访问该系统的情况（实际生活中也确实存在这种情况），因此电脑电话集成应用系统应是多信道的（一般都应在二线以上），而且要求每个信道都能并行地为用户提供服务。

二、解决方法

解决上述提出的问题最简单的方法是在每一信道后附一个相应的执行程序，每一个程序负责管理一根线路。当有多个人同时访问该系统时，计算机启动多个相应的程序为不同的用户服务。但这种解决方法仅适合于只有几线的电话语音输出系统。即使是这样，这种方法对计算机内存的开销也是很大的，因为该方法需要将多个应用程序同时调入内存。对

那些要求几十乃至几千线的电话语音输出系统，这种方法就根本不可能采用了。因此，在实际应用中，常采用以下几种方法：

第一、在DOS 系统中，对2 至16线的电话语音系统可以采用“状态机编程”的方法。

第二、在 UNIX 操作系统中，利用 UNIX 是一种分时操作系统`、允许多进程同时工作的特点来解决这一问题。

第三、在 WINDOWS NT 和 WINDOWS 95 操作系统中，利用 WINDOWS NT 和 95支持多任务、多线程的特点来解决这一问题。

但作为普及性，本文以应用广泛的 WINDOWS95 为操作系统平台、以美国的DIALOGIC公司的D/41D语音卡为例，介绍解决这一问题的两种方法。

1、同步程序模式中的解决方法

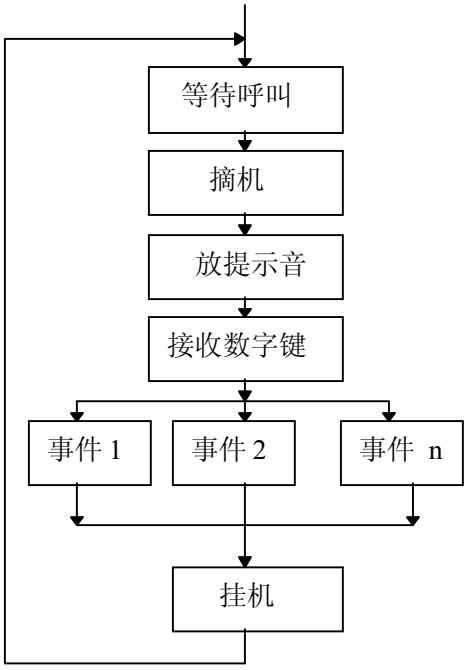

图 2

美国DIALOGIC公司为其语音卡提供了多种开发平台，如：DOS、UNIX 、WINDOWS95 、WINDOWS NT 、OS/2 。在 WINDOWS95 中，它提供了两种编程模式，即同步程序模式和异步程序模式。同步程序模式使用模块函数，允许一个独立的进程或线程控制一个信道。即该模式要求每一个访问（进程）都独占一个信道，而每一个信道都对应着一条独立的线路。这种模式能使你将不同的应用实时动态地分配给不同的信道。尽管可以将不同的进程分配到不同的信道中去处理，但在同步程序模式下，CPU 实时处理每一个任务，因此，如不加以适当的处理，某一访问一旦开始，则程序必须保证该访问全部执行完成为止，而后才允许其他的用户访问。其工作方式如图2 所示。因此，这样的系统就不可能为多个用户同时服务。如何解决这一问题呢？事实上，在这种形式下的某一个访问过程中，CPU 有很多时间是空闲的，如：在向用户播音的过程中，CPU 一直处于等待状态。因此，我们可以将每一个访问（进程）分为若干状态（如图3 所示），每个状态表示该进程执行到了一定的阶段，

然后用时间片轮询的方法让每个进程依次执行一小段。由于轮询的时间片很短，一般为毫秒级，每一个进程切换的延时很短，只要总的延时不超过 0.1 秒（总的延时与CPU的速度、硬盘存取速度、所接的线数等有关），用户是觉察不出来的。这样既充分利用了CPU 的资源,也在一定程度上满足了为多个用户同时提供服务的要求。

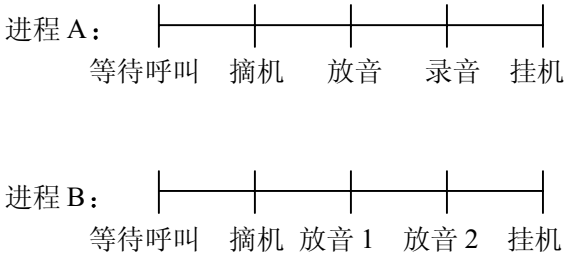

图 3

当然，为了在每次轮询中能正确无误地执行指定的操作，必须定义一个结构以存储信道的相应状态。程序根据各信道的当前状态标志，确定每个信道的任务执行阶段。

上述这种方法实际上是在同步模式下引入了异步机制。另一种方法是在同步模式下，充分利用 WINDOWS95 能够在一个进程中允许多个线程的特点，在程序的设计中建立多个线程，让每一个线程去管理相应的一根电话线路。这种方法看上去与前面所说的“在每一信道后附一个相应的执行程序，每一个程序负责管理一根线路”方法没有什么两样。事实上，这两种方法存在着很大的差别。这不仅仅是一个是多个程序而另一个是一个程序多个线程的差别。

这两者之间最根本的差别在于对计算机资源占有的不同。实际上，线程是进程的一个执行单元， Windows95 和 Windows NT 操作系统以轮转的方式向线程提供时间片，因此线程对系统资源的开销较小而适合于多线路控制技术。

2、异步程序模式中的解决方法

异步程序模式能在一个单进程中控制多个声音信道。这种异步程序模式支持得到事件和回电事件管理。在实际开发中，我们可以利用WINDOWS95 的多任务机制，通过一个或多个线程来控制各信道的状态消息。然后，通过状态消息的传递实现各信道不同进程所需的操作。这种状态消息传递机制（如图4 所示）完全类似于WINDOWS 的消息传递机制。

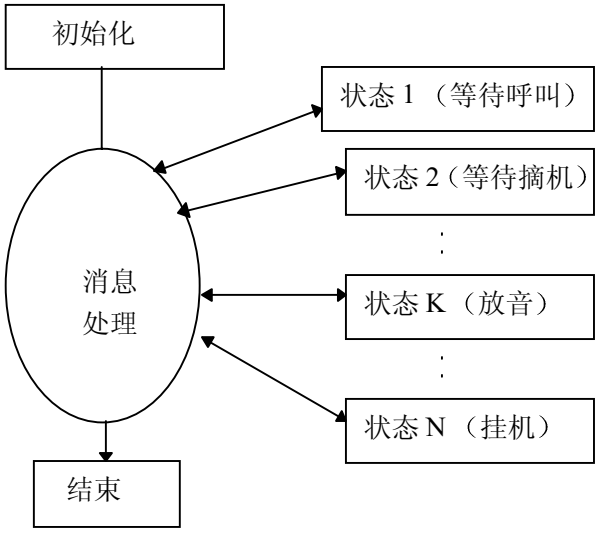

图 4

形式上，上图中的状态是流程图中的一个方框；本质上，状态意味着一定时间的操作。一个状态内操作的完成用事件来指示，即事件能使程序从一个状态进入到下一个状态。给定

状态下可能产生该事件，转移至相应的新事件。如：

状态 1　　等待呼叫

事件 = 检测到振铃：

开始等待摘机，并进入状态 2

状态 2　　等待摘机
事件 = 摘机
开始放提示音，进入状态 3
状态 3　　放提示音
事件 = 放音结束
开始等待按键数字音，进入状态 4
状态 4 = 取用户的按键数字
.....

如此传递下去，直到挂机，后又回到状态 1 （等待呼叫）。

由于程序在处理每一个事件（一个事件对应着一个状态消息）后，马上又回到等待处理下一个相同事件的状态，因而程序能够处理多个信道的不同访问。例如：在图4 中，当程序处理完第一个信道的取数（状态4）这一事件后，马上又回到等待取数状态，只要有相应的事件激活它，如第二个信道此时从状态3 进入到状态 4（取数事件），程序马上给第二个信道进行取数；之后，又回到等待取数状态。因此，该模式允许在一条线路中某一访问（进程）还没有结束，另一访问（进程）可以随即开始。这种编程模式非常适合于解决多线路同时控制的问题。它也是DIALOGIC 公司所推荐的一种编程模式。附1 简单地给出了异步程序模式下的算法描述。

三、注意的问题

在实际应用中，使用这种多线路的电脑电话语音集成系统还要注意以下几个问题：

第一、数据库的使用问题。尽管上面所述的方法可以解决多人同时访问的问题，但是如果有多个人在访问该系统时要求同时访问同一个数据库，这时就会引起系统延时，有时还会引起系统错误乃至整个系统崩溃。因此，要想系统能正常运行必须很好地解决这一问题。一般可以使用以下两种方法：

（1）保证让每一个信道访问不同的数据库。这样就意味着同样的数据库被复制成多个

数据库并取不同的名字以备各信道使用。这种方法不仅浪费很多磁盘空间，而且对那些要求几十线以上的电话语音输出系统也就无法使用了。

（2）让数据库共享。

第二、语音输出问题。如果直接使用录音重放的方法播放语音文件，对于那些只需要输出少量词汇的电脑语音集成系统还是很好的。但对于需要输出较大词汇量的系统则这种语音输出方式就有很多问题了。这是因为这种录音重放的方法有以下几个不利因素:

1.不利于系统的扩充与更改。因为每句话都靠录音形成，录好音后自然就不便更改了。所以任何更动与增减都需要重新录音，这显然是不方便的。

2.占用的磁盘空间大。 因为存储是的数字化了的音频信号，占用空间是比较大的。一个汉字的音频信号约为 3K 左右，所以输出一分钟的语音(按 200 个汉字计)，就约占空间 600K 。而且随着系统规模的扩大，磁盘空间的要求也随之大大的增加。

3.难以将数据库中的汉字比如用户姓名、工作单位等，用录音重放的方法一一拼接回放。

4 无法满足文本不能事先确定的应用场合。如：电子邮件阅读和传真阅读等。

为此，我们采用清华大学的文语转换系统（TTS）。清华大学计算机系研制的文语转换系统是一个无限词汇的合成系统，即按规则可生成任何语句而不必句句录音。它能将提供的文本文件转换为语音输出且系统音质清晰，语句自然度高。虽然存储文语转换系统的音库需要占用一些磁盘空间，但随着系统规模的扩大，音库的大小是不变的，增加的只是文本汉字数，显然增加的幅度是非常小的。因此以上的几个问题可以得到很好的解决。

第三、线路释放问题。在电话语音系统执行过程中，若用户在任何时候挂机，系统应能检测到这种状态并能做出相应的挂断，以便尽早结束通话状态，释放通讯线路。

Dialogic 语音卡并不提供直接的通话时忙音检测，但它能准确地检测各种音频信号。因此，由于忙音信号是 350 ms 一次通断(即 350 ms 高，350 ms 低)的方波，因此可构造相应的波形模板，并在通话状态时使能音频信号检测，这样即可达到目的 。

四、结束语

本文介绍的多线路同时控制技术是电脑电话集成（CTI）应用系统的关键技术，它可以广泛适用于各种不同的应用领域。如：语音信箱、高考放榜、电话银行、客运信息、气象服务、商情查询、自动传呼、168 信息台、200 长途接拨、 114 电话号码查询等等。据统计，97年我国电话装机总数已超过2千万台，随着我国邮电事业的蓬勃发展，电信网能力的不断提高，电话装机数将进一步扩大，这将给电脑电话集成系统带来更为广阔地应用前景。因此，
本文介绍的多线路同时控制技术有着很大的经济实用性。

附 1：算法描述

```
typedef struct
{
       short    nActiveChannelNo;    //活动信道号
       int    CurrentState;             // 活动信道号的当前状态
       int    EventType;              //事件类型
       short  nBreakEvent;           //引起某一状态终止事件
       char     *UserInDTMFString; //用户通过电话输入的字串
       WORD     wBreakMask;         //用户通过电话输入的终止掩码
} ChannelState;
char * filenamep;                    //文件名
int  nKeyNumber;                  //要求用户拨的键数
WORD  wKeyMask,wEndKey;          // 键的掩码与终止键


    begin
       while ( TRUE )
       {
      sr_waitevt(-1) ;        //等待事件
      nActiveChannelNo = (short)sr_getevtdev(0);//获取活动信道
      ChannelState.EventType = sr_getevttype(0);//获取活动信道的事件类型
         switch ( ChannelState.EventType )
        {
              case TDX_CST:                     //状态传输事件
            if (ChannelState.CurrentState == CS_RINGS) // 响铃状态
            dx_sethook(nActiveChannelNo,DX_OFFHOOK,EV_ASYNC);
            ChannelState.CurrentState = DX_OFFHOOK;//  摘机状态
            break;
         case TDX_SETHOOK:             //挂钩事件
            if  (ChannelState.CurrentState == DX_OFFHOOK)
            palySound( nActiveChannelNo,filenamep,nBreakEvent,wBreakMask);
            ChannelState.CurrentState = CS_PLAY; //放音状态
                     if  (ChannelState.CurrentState == DX_ONHOOK)//挂机状态
              dx_stopch(nActiveChannelNo, EV_ASYNC);
              ChannelState.CurrentState = CS_IDLE;//信道空闲状态
             break;
        case TDX_PLAY:                 //放音事件
```

```
            GetDTMFDigit(nActiveChannelNo,nKeyNumber,wKeyMask,wEndKey);
               ChannelState.CurrentState = CS_GTDIG;//取数状态
                       break;
         case TDX_GETDIG:          // 取数事件
             switch(ChannelState.UserInDTMFString)
        {
                  case '1':      //放音
              palySound( nActiveChannelNo,filenamep,nBreakEvent,wBreakMask);
              ChannelState.CurrentState = CS_PLAY;
              break;
              case '2':         //录音
              RecordSound( nActiveChannelNo,filenamep,nBreakEvent,wBreakMask);
                       ChannelState.CurrentState = CS_RECD; //录音状态
                break;
              default:
              dx_stopch(nActiveChannelNo, EV_ASYNC);
              dx_sethook(nActiveChannelNo,DX_ONHOOK,EV_ASYNC);
              ChannelState.CurrentState = DX_ONHOOK;//挂机状态
               break;
             }
                case TDX_RECORD:          //录音事件
              _closefile(filenamep);
              palySound( nActiveChannelNo,filenamep,nBreakEvent,wBreakMask);
              ChannelState.CurrentState = CS_PLAY;
              break;
            default:
              dx_stopch(nActiveChannelNo, EV_ASYNC);
              dx_sethook(nActiveChannelNo,DX_ONHOOK,EV_ASYNC);
              ChannelState.CurrentState = CS_IDLE;
}
    end
```

参考文献

1. Dialogic "System Release Software Installation Reference for windows95"， April， 1995
2. Dialogic "Voice Hardware Installation Guide"， April， 1994
3. Dialogic "Voice Software Reference for Windows 95", December,1995
4. "Dialogic Products And Services Guide" April,1997
5. 张延平 林博文 《计算机语音集成技术和应用》北京邮电大学 1997.7
6. 许军 "汉语文语转换系统的研究与实现 "， 北方交通大学信息科学研究所，1992.4