BF-PCI

系列电话语音/传真卡

用户手册

（For Windows 98/2000/XP）

深圳市博峰电子有限公司

ShenZhen BeauPhone Electronics CO., LTD.

目　　录

第一部分　BF-VOICE 电话语音产品概述

这一部分介绍 BF-VOICE 电话语音产品的类型、特点、提供给程序员开发使用的接口库及本手册的组织结构等内容。

第一章　BF-VOICE 电话语音产品类型

▲ **BF-PCI-V4**　4 路语音通道的电话语音卡
可提供 1~4 条模拟电话线接口或 1~4 个座席电话接口。

▲ **BF-PCI-R4**　4 路录音通道的电话录音卡
可提供 1~4 条录音线接口或 1 个放音音箱接口或 1 个远端查询接口。

▲ **BF-PCI-V8**　**8** 路语音通道的语音卡
可提供 1~8 条模拟电话线接口或 1~8 个座席电话接口

▲ **BF-PCI-R8**　8 路录音通道的电话录音卡
可提供 1~8 条录音线接口或 1 个放音音箱接口或 1 个远端查询接口。

▲ **BF-PCI-V4F4**　**4** 路语音通道 4 路传真通道的语音卡
可提供 1~4 条模拟电话线接口或 1~4 个座席电话接口，还提供 4 路传真资源。

▲ **BF-PCI-V8F4**　**8** 路语音通道 4 路传真通道的语音卡
可提供 1~8 条模拟电话线接口或 1~8 个座席电话接口，还提供 4 路传真资源。

▲ **BF-PCI-V8F8**　**8** 路语音通道 8 路传真通道的语音卡
可提供 1~8 条模拟电话线接口或 1~8 个座席电话接口，还提供 8 路传真资源。

▲ **BF-PCI-V4IP4**　**4** 路语音通道 4 路 VoIP 通道的语音卡
可提供 1~4 条模拟电话线接口或 1~4 个座席电话接口，还提供 4 路 VoIP 资源。提供标准的 H323 协议栈。

▲ **BF-PCI-V8IP4**　**8** 路语音通道 4 路 VoIP 通道的语音卡
可提供 1~8 条模拟电话线接口或 1~8 个座席电话接口，还提供 4 路 VoIP 资源。提供标准的 H323 协议栈。

▲ **BF-PCI-V8IP8**　**8** 路语音通道 8 路 VoIP 通道的语音卡
可提供 1~8 条模拟电话线接口或 1~8 个座席电话接口，还提供 8 路 VoIP 资源。提供标准的 H323 协议栈。

▲ **BF-PCI-V4F4IP4**　**4** 路语音通道 4 路传真通道 4 路 VoIP 通道的语音卡
可提供 1~4 条模拟电话线接口或 1~4 个座席电话接口，还提供 4 路传真资源或 4 路 VoIP 资源。提供标准的 H323 协议栈。

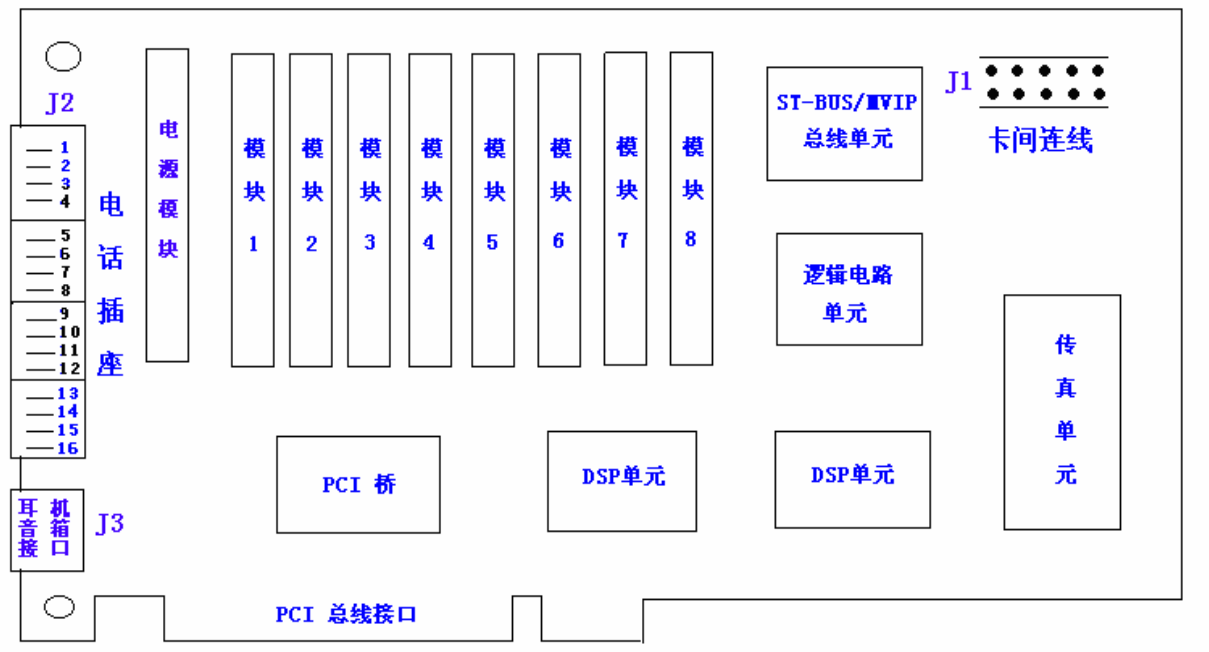
▲ **BF-PCI-V8F8IP8**　**8** 路语音通道 8 路传真通道 8 路 VoIP 通道的语音卡
可提供 1~8 条模拟电话线接口或 1~8 个座席电话接口，还提供 8 路传真资源或 8 路 VoIP 资源。提供标准的 H323 协议栈。

第二章　BF-VOICE 电话语音产品的功能特点

1.2.1　板卡功能特点

- ▲ 独特的 DSP 处理结构，能实时、高效地进行每线独立的数字化录音和放音。
- ▲ 独特的 DSP 处理结构，每线的数字化录音和放音可以同时并独立进行。
- ▲ 可以动态地选择不同的语音编码速率，并进行实时的语音压缩和解压，如 32kbps 的 ADPCM、 6 4 kbps 的 PCM。
- ▲ 可靠的双音多频按键（DTMF）检测，可在录音和放音的同时，立即响应 DTMF 按键。
- ▲ 独特的信号音产生(Generation)和识别(Detection)算法,可精确地识别任意频率的音频信号或产生任意频率的音频信号。
- ▲ 可靠的两线模拟中继的振铃检测。
- ▲ 带有极性反转和语音识别，能准确识别对方摘机。
- ▲ 带有主叫识别(Calling Identity Delivery)功能。
- ▲ 带有 ST-BUS 和 MVIP-BUS 总线，可实现同机内任意卡间的任意两个通道之间的交换。
- ▲ 卡上带有处理器和大容量的存贮器，不额外占用主机的资源。
- ▲ 能识别座席卡的摘机、挂机和拍插等动作。
- ▲ 驱动程序使用事件驱动结构，便于多通道编程。
- ▲ 卡上的传真芯片，使用 DSP 技术，提供与 Group 3 传真协议兼容，并可实现多通道传真。
- ▲ 不需要额外硬件，可以实现三方通话和会议功能
- ▲ 带有 TTS(Text to Speech)功能。
- ▲ 针对 VoIP，提供标准的 H323 协议栈。
- ▲ 多达 16 块卡在同一台计算机上，并共用 IRQ，不占用主机存贮器空间，使用户方便扩容。
- ▲ 扩容时，用户的应用程序基本不变。

1.2.2　板卡布局示意图



上图为八线卡的示意图，仅供参考，所有产品结构样式等均以实际板卡为准。

第三章　本手册的组织结构

BF-VOICE 电话语音卡用户手册(For Windows 98/2000/XP)为您在 WINDOWS 98/2000/XP 操作系统下使用和开发基于 BF-VOICE 电话语音卡的应用程序提供有益的信息，以及描述在 WINDOWS 98/2000/XP 下开发基于 BF-VOICE 电话语音卡的应用程序所使用的驱动程序和接口程序库。

本手册只适用于 BF-Voice 语音卡系列的以下型号产品：BF-PCI-V4、BF-PCI-R4、BF-PCI-V8、BF-PCI-R8 、 BF-PCI-V4F4 、 BF-PCI-V8F4 、 BF-PCI-V8F8 、 BF-PCI-V4IP4 、 BF-PCI-V8IP4 、 BF-PCI-V8IP8、BF-PCI-V4F4IP4、BF-PCI-V8F8IP8。

注意：在本手册中，如果没有特别指明，所有的 BF-VOICE 语音卡仅仅指为 BF-PCI-V4、BF-PCI-R4、BF-PCI-V8、BF-PCI-R8、BF-PCI-V4F4、BF-PCI-V8F4、BF-PCI-V8F8、BF-PCI-V4IP4、BF-PCI-V8IP4、BF-PCI-V8IP8、BF-PCI-V4F4IP4 和 BF-PCI-V8F8IP8 等几种型号的 PCI 接口的语音卡。

本手册安排了以下主要内容：

第一部分 BF-VOICE 电话语音产品概述，提供 BF-VOICE 电话语音系列产品的基本功能描述及本手册的组织结构。

第二部分 介绍了 BF-VOICE 系列产品的硬件特点以及硬件安装方法。

第三部分 介绍了一些编程必备知识，如电话信号音、DTMF、FSK 以及传真过程等。

第四部分 介绍了 BF-VOICE 语音卡的软件安装、编程接口函数以及编程实例等内容。

第五部分 介绍了 BF-VOICE 系列产品的应用工具，如语音编辑、信号音分析等内容。

第六部分 介绍了 BF-VOICE 语音系列产品的技术支持和售后服务信息。

第四章　BF-VOICE 语音卡应用平台

BF-VOICE 系列的语音产品可以安装在 CPU 为 80486 以上，内存 16M 以上，带有 PCI 扩展槽的 IBM-PC 兼容机上。

BF-VOICE 系列的语音产品提供 Windos 98、Windows 2000、以及 Windows XP 的驱动程序，故 BF-VOICE 系列的语音产品可以应用于 Windos 98、Windows 2000 以及 Windows XP 等操作系统平台上。

第五章　BF-VOICE 语音卡技术指标

接口方式：　两线模拟中继

接口阻抗：　600 欧姆

频率响应：　300Hz~3400Hz

中断向量：　可选择 2、3、5、6、7、10、11、12 中的任一个

片上缓存：　32K~256K Byte

数字语音：　A-率 PCM、ADPCM

数据速率：　64KBPS、32KBPS、16KBPS

DTMF 码： 0~9、*、#、A、B、C、D

交换功能： 256 通道直接交换

呼叫监测： 信号音、语音、极性反转

主叫号码： FSK、DTMF

FAX ： FAX Group 3

FSK 调制方式：

逻辑 0 ： 2200Hz±1%

逻辑 1 ： 1200Hz±1%

数据传送方式： 二进制异步串行方式

VoIP ： 标准的 H.323 协议栈。

第六章 BF-VOICE 语音卡提供的编程支持

对于 Windows 98/2000/XP 系统平台，BF-Voice 语音卡为程序员提供了动态连接库接口程序——BfVoice.DLL 和 BfVoice.H，以及 BF-VOICE 语音卡的各种编程语言的演示文件——Demo 程序。其中：

BfVoice.DLL 存放在操作系统目录，

BfVoice.H 存放在用户安装目录的 Demo 子目录下的各种编程语言子目录中。

BF-VOICE 语音卡的编程接口库提供 BF-VOICE 语音卡系列产品的编程接口函数，以及常量和数据结构的定义。用户可以选择一种熟悉的编程语言进行编写应用程序。我们将提供全方位的技术支持。

BF-VOICE 语音卡的编程设计采用流行的事件驱动编程结构，即应用程序不断的从 BF-VOICE 语音卡的系统程序中获取事件，然后根据事件内容，发出相应的处理命令。

BF-VOICE 语音卡的通道分为语音通道和资源通道两种，语音通道用于语音处理，资源通道必须绑定在语音通道上，以在语音通道实现传真和 VoIP 等功能。

无论每一个语音通道，抑或资源通道，都有物理通道和逻辑通道两种表示方法。

对于语音通道，BF-VOICE 语音卡的物理通道是相对于每一块语音卡的，是指母板左边的电话插座和母板上直插的模拟接口模块，其顺序从 1 开始，对于电话插座，1 通道对应于电话插座的最上面的一个插座，然后顺序数下，最后一个插座的物理通道为 4 或 8；而对于在母板上直插的模拟接口模块，1 通道对应于靠近挡板和电话插座的模块。然后顺序数下，最后一个远离挡板和电话插座的模拟接口模块的物理通道为 4 或 8；而逻辑通道则对应于插在同一台计算机中的所有卡而言，其顺序从 0 开始，其数量为所有板卡的所有通道的总和。逻辑通道的排列从第一块卡开始，先是排列物理通道的 1~4 或 8，接着再排列下一块卡，直至所有卡排列完毕；而每块卡的顺序由操作系统自动排列。

对于资源通道，其物理通道也是相对于每一块语音卡的，由板卡上的特定芯片提供，其顺序也是从 1 开始，直至该卡排完；而逻辑通道则对应于插在同一台计算机中的所有卡而言，其顺序也是从 0 开始，其数量为所有板卡的所有通道的总和。逻辑通道的排列也从第一块卡开始，直至最后一块卡结束。

在本手册中，如果没有特别说明，所有的 BF-VOICE 语音卡的通道均指逻辑通道。

用户在编写应用程序时，请参考第四部分第五章的相关小节。

第二部分　BF-VOICE 语音卡软硬件安装

BF-VOICE 系列语音卡为 PCI 接口，所有插于同一机器中的所有卡可以共享同一个中断，也可以与其他卡共享中断，其所需任何资源，均由系统自动分配，无须用户手工配置。

BF-VOICE 语音卡硬件采用模块化结构设计，模拟和数字分开。数字部分为一块独立的整板（简称母板），带有 DSP、CPLD、SRAM 和资源芯片等；模拟部分为一个个独立的模拟接口模块，共有三种模拟接口模块，即两线环路模拟中继模块（简称中继模块）、两线电话用户中继模块（简称用户模块）、两线环路模拟或两线语音录音模块（简称录音模块）。

第一章　BF-VOICE 语音卡产品包装

当您第一次打开 BF-VOICE 产品的包装时，您应该见到以下必备物品：

- ▲ BF-VOICE 语音卡母板一块
- ▲ 模拟接口模块 1~8 块（类型和数量视乎您的订购要求）
- ▲ CD-ROM 光盘一份，盘片内容为 Windows 98/2000/XP 驱动程序和应用程序
- ▲ 转接分配线若干条(视乎您的订购要求)

选购物品（视乎您的订购要求）：

- ▲ 铃流模块一个
- ▲ 交换电缆一条

第二章　BF-VOICE 语音卡的硬件安装

对于 Windows 98/2000/XP 系统平台，BF-VOICE 语音卡的硬件安装过程是一致的，请遵从 2.2.1——2.2..5 节的顺序进行安装。

2.2.1　安装模拟接口模块

一般情况下，模拟接口模块已经在出厂时被倒扣/安插在母板上，不需要用户再重新安装，但如果需要变动时，一定要注意模拟接口模块的倒扣/安插方向，不能插反。

2.2.2　插入 BF-VOICE 语音卡

关闭待安装计算机的电源，包括所有外设电源。一定将电源插头拔掉

拧下计算机盖的螺丝，小心打开计算机

选择空闲的 PCI 插槽，插入 BF-VOICE 语音卡

用螺丝固定好 BF-VOICE 语音卡

2.2.3　连接卡间连线

为了使插在同一台计算机中的所有卡的任意两个逻辑通道之间可以进行直接交换，每一块 BF-VOICE 卡上都设有 ST-BUS/MVIP-BUS/CT-BUS 总线的接口。如果您在同一台计算机中插入

多块语音卡，则必须连接所有卡的 ST-BUS/MVIP-BUS/CT-BUS 总线的接口。

卡间连线的接口为 J1，在连接卡间连线时，一定注意连线方向正确，即所有卡的连线方向一致。

2.2.4 连接电话机和/或模拟电话线

按照实际需要和模块类型，连接必要的模拟电话线和/或电话机，模拟电话线和/或电话机的接口为 J2，对于八线卡，每个 J2 插口含有两条线，并且每个插口的中间两芯为一组信号线，外侧两芯为一组信号线。

2.2.5 完成安装

装回计算机盖，并用螺丝固定好

将计算机放平放稳

打开计算机电源

至此，所有的硬件安装全部完毕，紧接着可以进入软件安装。

注意：一定要保证计算机的电源接好大地，并且做好防雷保护措施。

尽管 BF-VOICE 语音卡已经具有很好的防雷保护措施，但用户的机房或设备也应做好防雷保护措施，因接地或雷击等用户因素而导致的任何后果均由用户自己承担！

第三章 BF-VOICE 语音卡的软件安装

对于 Windows 系统平台，BF-VOICE 语音卡软件分为驱动程序和应用软件，对驱动程序而言，Windows 98、 Windows 2000 和 Windows XP 的安装是有区别的；而对于应用程序的安装，两者是一致的。用户必须先安装驱动程序，然后安装应用软件。下面详细叙述：

2.3.1 对于 Windows 98 驱动程序的安装过程如下：

▲ 安装好硬件，打开计算机电源，进入 Windows 操作系统。

▲ 将 BF-VOICE 语音卡安装盘（For Windows98/2000/XP）插入光盘驱动器中。

▲ 当系统提示找到新硬件并出现“此向导搜索下列设备的新驱动程序”，并指明“PCI Multimedia Device”或“PCI Bridge”提示对话框时，单击“下一步”。

▲ 在接下来的对话框中，选择“搜索设备的最新驱动程序（推荐）”，并单击“下一步”。

▲ 在接下来的对话框中，选择“指定位置”，依据 BF-VOICE 语音卡安装盘所插入的驱动器，在编辑框里键入相应的内容。如 BF-VOICE 安装盘在光盘驱动器 D：驱动器，则键入“D:\Drivers”。再单击“下一步”。

▲ 在接下来的对话框中，单击“下一步”。

▲ 在接下来的对话框中，单击“完成”。

至此，Windows 98 的驱动程序安装完毕。

2.3.2 对于 Windows 2000/XP 驱动程序的安装过程如下：

▲ 安装好硬件，打开计算机电源，进入 Windows 操作系统。
▲ 将 BF-VOICE 语音卡安装盘（For Windows98/2000/XP）插入光盘驱动器中。
▲ 当系统提示找到新硬件，并出现“此向导帮助您为硬件设备安装驱动程序”的对话框时，单击“下一步”。
▲ 在接下来的对话框中，选择“搜索适于我的设备的驱动程序（推荐）”，并单击“下一步”。
▲ 在接下来的对话框中，选择“指定位置”，并单击“下一步”。
▲ 在接下来的对话框中，依据 BF-VOICE 语音卡安装盘所插入的驱动器，在编辑框里键入相应的内容。如 BF-VOICE 安装盘在光盘驱动器 D：驱动器，则键入“D：\Drivers”。再单击“确定”。
▲ 在接下来的对话框中，单击“下一步”。
▲ 在接下来的对话框中，单击“完成”。

至此，Windows 2000/XP 的驱动程序安装完毕。

2.3..3 对于应用程序的安装过程如下：

▲ 在安装完 BF-VOICE 的 Windows 98/2000/XP 的驱动程序后，启动计算机，并进入系统。
▲ 将 BF-VOICE 语音卡安装盘（For Windows98/2000/XP）插入光盘驱动器中。
▲ 在 Windows 的菜单“开始”——“运行”下，键入“驱动器名称：\Utility\Setup”，并回车，进入安装向导，您可以跟着安装向导，进行安装。
▲ 根据安装向导提示，首先选择安装语言（中文/英文）。
▲ 根据安装向导提示，键入语音卡的应用程序的安装路径，缺省为 C:\BFVOICE。
▲ 根据安装向导提示，键入语音卡的 WINDOWS/Program 的菜单名称，缺省为 BFVOICE。

至此，BF-VOICE 语音卡的应用程序安装完毕。

2.3.4 测试 BFVOCIE 语音卡：

当完成上述三节的过程后，您可以使用 BF-VOICE 语音卡的应用程序来测试语音卡，我们为您提供的应用程序有演示程序（在 Demo 目录）、信号音分析程序(在 Analyze 目录)、语音编辑程序(在 VoiceEdit 目录)等。

BF-VOICE 的语音卡的演示程序即提供了测试 BF-VOICE 语音卡的大部分功能，如按键测试、录音、放音、信号音产生、发送传真和接收传真等，也提供了一个编写基于 BF-VOICE 语音卡的应用程序的范例，仅供用户参考。

第三部分　BF-VOICE 语音卡编程准备知识

使用 BF-VOICE 语音卡编写应用程序需要了解通信方面的知识，这些也是必备的知识。下面介绍一些基本的知识。

第一章　DTMF

DTMF（Dual Tone Multi-Frequency），即双音多频信号，广泛用于电话拨号上，DTMF 由 CCITT 制定，并推荐作为按键式电话的标准。

每一个 DTMF 信号实际上是由两种音调的声音组合而成，一个是低频，另一个是高频。这样组成　DTMF 信号的频率就有两组，一组是低频信号 697Hz、770Hz、852Hz 和 941Hz，另一组是高频信号 1209Hz、1336Hz、1477Hz 和 1633Hz。

由四个低频信号和四个高频信号组合成 16 种 DTMF 信号——0~9 十个数字、*、#、A、B、C、D，这些信号的具体频率定义如下：

	1209Hz	1336Hz	1477Hz	1633Hz
697Hz	1	2	3	A
770Hz	4	5	6	B
852Hz	7	8	9	C
941Hz	*	0	#	D

在 BF-VOICE 语音卡上，DTMF 的识别和产生全部由 DSP 处理器实现。由于使用独特的算法，其识别准确度高于硬件识别。

第二章　信号音

在公共电话网上，线路的状态大都是通过信号音来表达的，如忙音、拨号音和回铃音等，下面介绍几种常用的信号音。

3.2.1　拨号音

公共电话网上的拨号音是在提机时，由交换机发出的一个 450Hz 的声音，一般它持续 6~10 秒，表示电话线路正常，并可以进行拨号呼叫。BF-VOICE 语音卡提供了 BF_StartDetectDialTone 函数来检测标准拨号音。

3.2.2 回铃音

公共电话网上的回铃音是在发出呼叫后，由交换机发出的一个 450Hz 的声音，表示被叫电话已经被接通，只要对方摘机，便可以进行对话。它是一个 1 秒有，4 秒无的有无间断的 450Hz 的声音。BF-VOICE 语音卡提供了 BF_StartDetectBackTone 函数来检测标准回铃音。

3.2.3 忙音

忙音有几种形式，如在呼叫之后由于对方线路正忙而无法接通时，由交换机发出给主叫的信号音；也有在呼叫双方通话完毕，因一方挂机，而交换机给另一方发出的信号音。虽然国际规定忙音为 700 毫秒有，700 毫秒无的有无间隔的声音，但各个交换机厂家也有自己的忙音标准，致使忙音种类很多，需要用户小心使用。

第三章　极性反转

极性反转是在拨号之后，由交换机送出的用于识别对方是否摘机的信号。

判断对方摘机的一般方法是:，当拨号之后，检测是否有回铃音，如果有回铃音再检测是否有语音，以判定对方是否摘机。但这种判断方法存在误差，而且无法分辨因对方占线或误拨等情况下由交换机送出的机器声音。

基于以上原因，使用极性反转来判断对方是否摘机，是最准确的，但在使用本功能时，必须首先向电信公司申请电话的极性反转功能，因为该项功能并不是电信业务的基本功能，所以必须特殊申请。

BF-VOICE 语音卡既支持由语音判断对方摘机的功能，也支持用极性反转来判定对方摘机的功能。

注　意：只有向电信局申请了极性反转功能，并且电信局的交换机支持极性反转功能，才能有效使用 BF-VOICE 语音卡为极性反转功能而提供的相应函数。

第四章　语音以及语音压缩

3.5.1 A-率 PCM 码

Pulse Code Modulation(PCM)是一种非均匀量化模拟信号的方法，该方法被广泛应用于电信业的语音传输和存储上。PCM 码是在模拟信号量化的基础上，再根据语音的特点，进行非均匀压缩。CCITT 的建议 G.711 规定了两种 PCM 的量化和压缩方法——A-律 PCM 码和 u-律 PCM 码。

u-律 PCM 为北美和日本所采用的语音压扩方法；而中国和欧洲采用 A-律 PCM 语音压扩方法。

经过A-律 PCM 编码后的模拟语音数据就变成了一个离散的数码流。一个 PCM 码的取样值对应 8 个比特，整个数码流的速率为 8kHz×8bit=64kbit/s。

BF-VOICE 语音产品在录音时，支持从模拟语音信号到 A-律 PCM 的数字化过程，并以 A-律 PCM 数字语音码存储语音文件；同时在放音时，支持从 A-律 PCM 的数字语音码到模拟语音信号的扩张过程。

3.5.2 ADPCM 码

ADPCM(Adaptive Differential Pulse Code Modulation)是 CCITT 推荐的语音压缩方法。它是根据语音的特点而制定的有损压缩方法，在 ADPCM 中，实际存储的不是语音的采样值，而是这一次采样与上一次采样的差值。由于语音信号的变化是连续、平滑而且很慢的，所以采样差值的位数减少，从而所需的存储空间也减少，达到压缩的目的。

BF-VOICE 语音卡不需要额外增加硬件，由板卡上 DSP 来实时处理 ADPCM 码的压扩。

BF-VOICE 语音卡提供的是 CCITT 建议的 32kbps 标准的 ADPCM 码的录音和放音。

第五章　主叫号码识别

主叫识别信息传送及显示(Calling Identity Delivery，简称 CID)业务是向被叫电话用户提供的一种新的服务项目。是指在被叫用户终端设备上显示主叫号码、主叫用户姓名、呼叫日期、时间等主叫识别信息，并进行存储，以供用户查阅的一种服务项目。

在模拟电话线路上，实现主叫识别信息的传送方法有两种——DTMF 和 FSK。

DTMF 方式是在第一声振铃之前使用 DTMF 音频信号传递，其特点是发送速度慢、无效验，而且仅仅发送主叫电话号码。

FSK 方式是在第一声振铃和第二声振铃之间使用 FSK 传递，特点是发送速度快、有效验，并且可以发送除主叫号码以外的信息，如主叫姓名、呼叫日期、时间等。但要注意，如果在第一声振铃时马上摘机，便无法收到主叫信息。

FSK(Frequency Shift Keying)，即频移键控码，在电信上，主要被用为主叫信息识别的信号。

使用 FSK 传送主叫信息的国际标准有两种——BELL 和 CCITT 的，其中：

BELL 建议标准：

逻辑 0 用 2200Hz±1%表示

逻辑 1 用 1200Hz±1%表示。

CCITT 建议标准：

逻辑 0 用 2100Hz±1%表示

逻辑 1 用 1300Hz±1%表示。

我国原邮电部规定的主叫信息传送采用 BELL 标准，即

逻辑 0 用 2200Hz±1%表示

逻辑 1 用 1200Hz±1%表示。

在 BF-VOICE 语音卡上，FSK 信号的识别由硬件完成，并且遵循我国原邮电部规定的标准。

注　意：只有向电信局申请了主叫识别信息功能，并且电信局的交换机支持主叫识别信息功能，才能有效使用 BF-VOICE 语音卡为主叫识别信息功能而提供的相应函数。

第四部分　BF-VOICE 语音卡软件编程

这一部分介绍 BF-VOICE 语音卡的编程接口函数以及编程常量和数据结构的定义。BF-VOICE 语音卡的编程接口函数分为语音通道操作接口函数和资源通道操作接口函数两大类。

第一章　BF-VOICE 语音卡语音通道编程接口函数

4.1.1　语音通道操作接口函数按功能索引表

初始化和关闭系统接口函数包括：

```
unsigned short WINAPI BF_InitializeSystem(void);
unsigned short WINAPI BF_CloseSystem(void);
```

获取系统参数接口函数

```
unsigned short WINAPI BF_GetTotalCard(void);
unsigned short WINAPI BF_GetCardSerialNo(unsigned short CardNo, unsigned
        char *SerialNo);
unsigned short WINAPI BF_SetCardUserID(unsigned short CardNo,unsigned char
        *Password,unsigned short PasseordLength,unsigned char *UserID);
unsigned short WINAPI BF_GetCardUserID(unsigned short CardNo,unsigned char
        *Password,unsigned short PasseordLength,unsigned char *UserID);
unsigned short WINAPI BF_GetTotalChannel(void);
unsigned short WINAPI BF_GetChannelType(unsigned short ChannelNo);
unsigned short WINAPI BF_GetChannelStatus(unsigned short ChannelNo);
```

打开关闭通道接口函数

```
unsigned short WINAPI BF_OpenChannel(unsigned short ChannelNo);
unsigned short WINAPI BF_CloseChannel(unsigned short ChannelNo);
```

获取事件接口函数

```
unsigned short WINAPI BF_CheckMessage(unsigned short WaitTime,
        PBF_MESSAGE_INFO pMessageBuffer);
unsigned short WINAPI BF_GetMessage(unsigned short WaitTime,
        PBF_MESSAGE_INFO pMessageBuffer);
```

侦听与连接通道接口函数

```
unsigned short WINAPI BF_ListenChannel(unsigned short ChannelNo, unsigned short
        SourceChannel);
unsigned short WINAPI BF_UnListenChannel(unsigned short ChannelNo);
unsigned short WINAPI BF_LinkChannel(unsigned short FirstChannelNo, unsigned
        short SecondChannelNo);
unsigned short WINAPI BF_UnLinkChannel(unsigned short FirstChannelNo, unsigned
        short SecondChannelNo);
```

两线模拟中继模块操作接口函数包括：

```
unsigned short WINAPI BF_TrunkHookOff(unsigned short ChannelNo);
unsigned short WINAPI BF_TrunkHookOn(unsigned short ChannelNo);
```

```
unsigned short WINAPI BF_StartDetectTrunkPolarity(unsigned short ChannelNo);
unsigned short WINAPI BF_StopDetectTrunkPolarity(unsigned short ChannelNo);
unsigned short WINAPI BF_StartTrunkPickUp(unsigned short ChannelNo,unsigned short
        TimeByMilliSecond);
unsigned short WINAPI BF_StopTrunkPickUp(unsigned short ChannelNo);
```

用户模块操作接口函数

```
unsigned short WINAPI BF_UserRingOn(unsigned short ChannelNo,unsigned short
        RingOnTime,unsigned short RingOffTime,unsigned short TotalTimes);
unsigned short WINAPI BF_UserRingOff(unsigned short ChannelNo);
unsigned short WINAPI BF_SetUserPickUpTime(unsigned short TimeByMilliSecond);
```

录音接口函数

```
unsigned short WINAPI BF_StartRecordFile(unsigned short ChannelNo,char
        *FileName,unsigned long Offset,unsigned long Length,char StopChar);
unsigned short WINAPI BF_StartRecordBuffer(unsigned short ChannelNo,char
        *Buffer,unsigned long Offset,unsigned long Length,char StopChar);
unsigned short WINAPI BF_StartRecordCircleBuffer(unsigned short ChannelNo,char
        *Buffer,unsigned long TotalLength,char StopChar);
unsigned short WINAPI BF_StopRecordVoice(unsigned short ChannelNo);
unsigned short WINAPI BF_GetRecordLength(unsigned short ChannelNo, unsigned long
        *RecordLength);
```

放音接口函数

```
unsigned short WINAPI BF_StartPlayFile(unsigned short ChannelNo,char
        *FileName,unsigned long Offset,unsigned long Length,char StopChar);
unsigned short WINAPI BF_StartPlayBuffer(unsigned short ChannelNo,char
        *Buffer,unsigned long Offset,unsigned long Length,char StopChar);
unsigned short WINAPI BF_InitPlayIndexFile(unsigned short ChannelNo);
unsigned short WINAPI BF_AddPlayIndexFile(unsigned short ChannelNo, char
        *FileName);
unsigned short WINAPI BF_StartPlayIndexFile(unsigned short ChannelNo,char
        StopChar);
unsigned short WINAPI BF_InitPlayIndexBuffer(void);
unsigned short WINAPI BF_AddPlayIndexBuffer(char *Buffer,unsigned long Length);
unsigned short WINAPI BF_StartPlayIndexBuffer(unsigned short ChannelNo,unsigned
        short *IndexTable,unsigned short IndexCount,char StopChar);
unsigned short WINAPI BF_StartPlayCircleBuffer(unsigned short ChannelNo,char
        *Buffer,unsigned long TotalLength,char StopChar);
unsigned short WINAPI BF_StopPlayVoice(unsigned short ChannelNo);
unsigned short WINAPI BF_GetPlayLength(unsigned short ChannelNo, unsigned long
        *PlayLength);
```

DTMF 按键产生和识别接口函数

```
unsigned short WINAPI BF_GenerateDTMFString(unsigned short ChannelNo, char
        *DTMFString);
unsigned short WINAPI BF_StartDetectDTMF(unsigned short ChannelNo, unsigned short
        SmartLevel);
```

```
    unsigned short WINAPI BF_StopDetectDTMF(unsigned short ChannelNo);
    unsigned short WINAPI BF_ClearDTMFBuffer(unsigned short ChannelNo);
    unsigned short WINAPI BF_GetDTMFKey(unsigned short ChannelNo,unsigned short
            DTMFCount,char *DTMFKey);
    unsigned short WINAPI BF_CheckHitDTMF(unsigned short ChannelNo);
```

信号音产生接口函数

```
    unsigned short WINAPI BF_StartGenerateSignal(unsigned short ChannelNo,unsigned
            short Frequency0,unsigned short Frequency1,unsigned short OnTime,
            unsigned short OffTime,unsigned short TotalTimes,char StopChar);
    unsigned short WINAPI BF_StopGenerateSignal(unsigned short ChannelNo);
```

标准信号音识别接口函数

```
    unsigned short WINAPI BF_SetDetectToneInfo(unsigned short ChannelNo,
            PBF_TONE_INFO ToneInfo);
    unsigned short WINAPI BF_SetDetectToneFrequency(unsigned short ChannelNo,
            unsigned short Frequency);
    unsigned short WINAPI BF_StartDetectDialTone(unsigned short ChannelNo);
    unsigned short WINAPI BF_StopDetectDialTone(unsigned short ChannelNo);
    unsigned short WINAPI BF_StartDetectBackTone(unsigned short ChannelNo);
    unsigned short WINAPI BF_StartDetectBackToneStop(unsigned short ChannelNo,
            unsigned short StopTimeByMs);
    unsigned short WINAPI BF_StopDetectBackToneStop(unsigned short ChannelNo);
    unsigned short WINAPI BF_StopDetectBackTone(unsigned short ChannelNo);
    unsigned short WINAPI BF_StartDetectBusyTone(unsigned short ChannelNo);
    unsigned short WINAPI BF_StopDetectBusyTone(unsigned short ChannelNo);
    unsigned short WINAPI BF_StartDetectToneData(unsigned short ChannelNo);
    unsigned short WINAPI BF_StopDetectToneData(unsigned short ChannelNo);
```

特殊信号音识别接口函数

```
    unsigned short WINAPI BF_SetSpecialSignalFrequency(unsigned short ChannelNo,
            unsigned short Frequency);
    unsigned short WINAPI BF_StartDetectSpecialSignal(unsigned short ChannelNo,
            unsigned short TimeByMilliSecond);
    unsigned short WINAPI BF_StopDetectSpecialSignal(unsigned short ChannelNo);
```

语音/静音识别接口函数

```
    unsigned short WINAPI BF_StartDetectHelloVoice(unsigned short ChannelNo);
    unsigned short WINAPI BF_StopDetectHelloVoice(unsigned short ChannelNo);
    unsigned short WINAPI BF_StartDetectSilence(unsigned short ChannelNo,unsigned
            short ThresholdValue);
    unsigned short WINAPI BF_StopDetectSilence(unsigned short ChannelNo);
```

超时设置接口函数

```
    unsigned short WINAPI BF_StartTimeOut(unsigned short ChannelNo, unsigned short
            TimeBySecond);
    unsigned short WINAPI BF_StopTimeOut(unsigned short ChannelNo);
    unsigned short WINAPI BF_SetTimeOutValue(unsigned short ChannelNo, unsigned short
            TimeBySecond);
```

主叫识别接口函数

```
unsigned short WINAPI BF_StartDetectCallerID(unsigned short ChannelNo);
unsigned short WINAPI BF_StopDetectCallerID(unsigned short ChannelNo);
unsigned short WINAPI BF_GetRawFskCallerID(unsigned short ChannelNo, char
        *CallerIDString);
unsigned short WINAPI BF_GetFskCallerID(unsigned short ChannelNo,char
        *DateTime,char *TelNo,char *Name);
```

发送/接收 FSK 接口函数

```
unsigned short WINAPI BF_StartSendFsk(unsigned short ChannelNo,unsigned short
        SeizureFlag,char *Buffer,unsigned short Count);
unsigned short WINAPI BF_StopSendFsk(unsigned short ChannelNo);
unsigned short WINAPI BF_StartDetectFsk(unsigned short ChannelNo);
unsigned short WINAPI BF_StopDetectFsk(unsigned short ChannelNo);
unsigned short WINAPI BF_GetFskData(unsigned short ChannelNo,char *Buffer,
        unsigned short Count);
```

自动拨号接口函数

```
unsigned short WINAPI BF_AutoDialOut(unsigned short ChannelNo,char *DTMFString);
```

语音数据格式转换接口函数

```
unsigned short WINAPI BF_AlawToWave(char *AlawFileName, char *WaveFileName);
unsigned short WINAPI BF_WaveToAlaw(char *WaveFileName, char *AlawFileName);
unsigned short WINAPI BF_AlawToAdpcm(char *AlawFileName, char *AdpcmFileName);
unsigned short WINAPI BF_AdpcmToAlaw(char *AdpcmFileName, char *AlawFileName);
```

停止通道所有操作接口函数

```
unsigned short WINAPI BF_StopChannelOperate(unsigned short ChannelNo);
```

会议接口函数

```
unsigned short WINAPI BF_ConferenceAddMember(unsigned short ChannelNo,unsigned
        short Mode,unsigned short Group);
unsigned short WINAPI BF_ConferenceDelMember(unsigned short ChannelNo,unsigned
        short Mode,unsigned short Group);
```

TTS(Text To Speech)接口函数

```
unsigned short WINAPI BF_InitTTS(void);
unsigned short WINAPI BF_CloseTTS(void);
unsigned short WINAPI BF_TTSFileToFile(char *TextFileName, char *VoiceFileName);
unsigned short WINAPI BF_TTSBufferToBuffer(char *TextBuffer,unsigned long
        TextCount,char *VoiceBuffer,unsigned long *VoiceCount);
unsigned short WINAPI BF_TTSBufferToFile(char *TextBuffer,unsigned long
        TextCount,char *VoiceFileName);
```

4.1.2 语音通道操作接口函数详细说明

4.1.2.1 初始化和关闭函数

▲ BF_InitializeSystem

函数原型： unsigned short WINAPI BF_InitializeSystem(void);

功　　能： 该函数初始化 BF-VOICE 语音卡的驱动程序参数，所以在使用任何 BF-VOICE

提供的接口函数前，一定要调用该函数

参　　数：　无

返　　回：　如果函数成功，返回 BF_OK；否则返回错误码。错误码如下：

代号	数值	含义
BF_OK	0x0000	没有错误
BF_ERROR	0xFFFF	一般性错误

注　　释：　只有该函数返回 BF_OK，语音卡才能正常操作。如果该函数调用失败，请检查语音卡的硬件和软件设置

▲ BF_CloseSystem

函数原型：　unsigned short WINAPI BF_CloseSystem(void);

功　　能：　该函数释放 BF-VOICE 语音卡的驱动程序所申请的系统资源，所以当停止 BF-VOICE 语音卡操作时，应该调用该函数。

参　　数：　无

返　　回：　如果函数成功，返回 BF_OK；否则返回错误码。

注　　释：　如果调用过该函数之后，需要再次使用语音卡，需要再次调用函数 BF_InitializeSystem 来初始化语音卡

4.1.2.2 获取系统参数接口函数

▲ BF_GetTotalCard

函数原型：　unsigned short WINAPI BF_GetTotalCard(void);

功　　能：　该函数取得插入计算机中的 BF-VOICE 语音卡的数量

参　　数：　无

返　　回：　函数返回插入计算机中的 BF-VOICE 语音卡的数量

注　　释：　如果函数成功，返回 BF_OK

▲ BF_GetCardSerialNo

函数原型：　unsigned short WINAPI BF_GetCardSerialNo(unsigned short CardNo,
unsigned char *SerialNo);

功　　能：　该函数取得插入计算机中的指定语音卡的系列号。对于任一 BF-VOICE 语音卡，都有一个三十字节的系列号，用以识别 BF-VOICE 语音卡

参　　数：　CardNo　　卡的逻辑号码，从 0 开始
SerialNo　　返回的卡的系列号的指针，该缓存区由用户提供

返　　回：　如果函数成功，返回 BF_OK

注　　释：　无

▲ BF_SetCarduserID

函数原型：　unsigned short WINAPI BF_SetCardUserID(unsigned short CardNo,
unsigned char *Password,unsigned short PasseordLength,
unsigned char *UserID);

功　　能：　该函数设置指定语音卡的用户 ID 和密码。

参　　数：　CardNo　　卡的逻辑号码，从 0 开始
Password　　用户密码，长度 1-7
PasswordLength　用户密码长度，取值 1--7

UserID　　用户 ID，长度 1-7

返　回：　如果函数成功，返回 BF_OK

注　释：　为了保护用户的应用软件，BF-VOICE 语音卡为用户应用程序在每快板卡上提供了一个 8 个字节的加密区域，该区域允许用户依据自己的喜好，写入 ID 与 Password 长度共为 8 字节的加密数据，因为板卡依据 ID 给 Password 数据加密，所以 ID 与 Paaword 长度均不能为 0

▲ BF_GetCarduserID

函数原型：　unsigned short WINAPI BF_GetCardUserID(unsigned short CardNo,
unsigned char *Password,unsigned short PasseordLength,
unsigned char *UserID);

功　能：　该函数设置指定语音卡的用户 ID 和密码。

参　数：　CardNo　　卡的逻辑号码，从 0 开始
Password　　用户密码，长度 1-7
PasswordLength　用户密码长度，取值 1--7
UserID　　用户 ID，长度 1-7

返　回：　如果函数成功，返回 BF_OK

注　释：　为了保护用户的应用软件，BF-VOICE 语音卡为用户应用程序在每快板卡上提供了一个 8 个字节的加密区域，该区域允许用户依据自己的喜好，写入 ID 与 Password 长度共为 8 字节的加密数据，因为板卡依据 ID 给 Password 数据加密，所以 ID 与 Paaword 长度均不能为 0。为了保护用户数据起见，板卡并不记录设置时的密码长度，所以此函数必须输入正确的密码长度。

▲ BF_GetTotalChannel

函数原型：　unsigned short WINAPI BF_GetTotalChannel(void);

功　能：　该函数取得插入计算机中的 BF-VOICE 语音卡的语音逻辑通道数量

参　数：　无

返　回：　函数返回插入计算机中的 BF-VOICE 语音卡的语音逻辑通道数量

注　释：　逻辑通道数量仅仅包括语音通道的

▲ BF_GetChannelType

函数原型：　unsigned short WINAPI BF_GetChannelType(unsigned short ChannelNo)

功　能：　该函数获取语音逻辑通道号码为 ChannelNo 的通道所对应的物理通道类型

参　数：　ChannelNo　　语音逻辑通道号码

返　回：　函数返回语音逻辑通道号码为 ChannelNo 的通道所对应的物理通道类型
物理通道类型常量的定义如下：

代号	数值	含义
CHANNEL_TYPE_NONE	0	逻辑通道所对应的物理通道没有任何模块
CHANNEL_TYPE_USER	1	逻辑通道所对应的物理通道为用户模块
CHANNEL_TYPE_TRUNK	2	逻辑通道所对应的物理通道为中继模块
CHANNEL_TYPE_RECORD	3	逻辑通道所对应的物理通道为录音模块

注　释：　ChannelNo 不能超过最大逻辑通道数量减一

▲ BF_GetChannelStatus

函数原型： unsigned short WINAPI BF_GetChannelStatus(unsigned short ChannelNo)

功　　能： 该函数获取语音逻辑通道号码为 ChannelNo 的通道的当前状态

参　　数： ChannelNo　　语音逻辑通道号码，取值从 0 开始。

返　　回： 函数返回语音逻辑通道号码为 ChannelNo 的通道所对应的通道状态。
通道状态往往是各种状态的组合，即下面的几个通道状态常量的或值。
通道状态常量的定义如下：

代号	数值	含义
CHANNEL_STATUS_IDLE	0x0001	通道空闲，没有打开
CHANNEL_STATUS_OPEN	0x0002	通道已经打开
CHANNEL_STATUS_PLAY	0x0004	通道正在放音
CHANNEL_STATUS_RECORD	0x0008	通道正在录音
CHANNEL_STATUS_GENERATE_DTMF	0x0010	通道正在产生 DTMF
CHANNEL_STATUS_SEND_FSK	0x0020	通道正在产生 FSK
CHANNEL_STATUS_GENERATE_SIGNAL	0x0040	通道正在产生信号音
CHANNEL_STATUS_DETECT_SIGNAL	0x0080	通道正在识别信号音
CHANNEL_STATUS_SEND_FAX	0x0100	通道正在发送传真
CHANNEL_STATUS_RECEIVE_FAX	0x0200	通道正在接收传真
CHANNEL_STATUS_VOIP	0x0400	通道正在 VoIP 操作
CHANNEL_STATUS_AUTO_DIAL_OUT	0x0800	通道正在自动拨号

注　　释： ChannelNo 不能超过最大逻辑通道数量减一。
通道的状态可以是以上几种状态的与。

4.1.2.3 打开关闭通道接口函数

▲ BF_OpenChannel

函数原型： unsigned short WINAPI BF_OpenChannel(unsigned short ChannelNo)

功　　能： 打开语音通道，本函数仅仅适用于语音通道

参　　数： ChannelNo　要打开通道的逻辑号码

返　　回： 如果打开成功，返回 BF_OK

注　　释： 对于每一个逻辑通道，只有被打开之后才能正常工作。

对于内线模块，通道打开后，BF-VOICE 驱动程序给它所对应的物理通道供电，并实时检测内线模块的摘机、挂机和拍插动作，并在内线模块有动作时，传递相应事件给应用程序。

对于中继模块，通道打开后，BF-VOICE 驱动程序实时检测中继模块的振铃和极性信息，并在中继模块有振铃时，传递振铃事件；在模块摘机后，开始检测极性是否反转，在有极性反转时，发送极性反转事件，而在模块挂机后，停止极性反转检测，如果需要特殊的极性反转检测，请使用 BF_StartDetectTrunkPolarity 和 BF_StopDetectTrunkPolarity 函数。

对于录音模块，通道打开后，BF-VOICE 驱动程序实时检测与录音模块并接的话机的摘机和挂机动作，并在有动作时，传递相应事件给应用程序。

▲ BF_CloseChannel

函数原型： unsigned short WINAPI BF_CloseChannel(unsigned short ChannelNo)

功　　能： 关闭语音通道，本函数仅仅适用于语音通道

参　　数：　ChannelNo　　要关闭通道的逻辑号码

返　　回：　如果打开成功，返回 BF_OK

注　　释：　当不再使用已经打开的某些通道时，为减少占用语音卡的板上资源，可以关闭不再使用的通道。如果要使用某个已经被关闭的通道，必须再调用函数 BF_OpenChannel 重新打开。

4.1.2.4　获取事件接口函数

▲　BF_GetMessage

函数原型：　unsigned short WINAPI BF_GetMessage(unsigned short WaitTime,
PBF_MESSAGE_INFO pMessageBuffer);

功　　能：　获取 BF-VOICE 语音卡的事件，该函数将事件从事件队列中取出。

参　　数：　WaitTime　　指示如果没有事件，该函数可以被挂起的时间，单位毫秒。
WaiTime 的取值可以从 0 至 0xFFFF， 建议取值为 50。
如果 WaiTime 的取值为 0，则不管是否有事件，都立即返回；
如果 WaiTime 的取值为 0xFFFF，则挂起该调用，直到有事件才返回；
如果 WaiTime 的取值为 0x0001 至 0xFFFE 之间的任意一个值，则为挂起的时间数量，其单位为毫秒。

pMessageBuffer　指向 PBF_MESSAGE_INFO 结构的地址指针，用于存放返回的事件，PBF_MESSAGE_INFO 的结构定义如下：

```
typedef struct _BF_MESSAGE_INFO{
    USHORT      MessageCode;
    USHORT      ChannelNo;
    USHORT      Parameter[6];
} BF_MESSAGE_INFO, *PBF_MESSAGE_INFO;
```

如果该函数返回 BF_OK，则表示有事件发生，事件类型以及其它指示存于上面结构中。PBF_MESSAGE_INFO 结构中的各项含义如下：

MessageCode 指示事件的类型，取值见《第四部分——第四章——BF-VOICE 语音卡编程接口事件类型说明》

ChannelNo　指示发生事件的逻辑通道号码。

Parameter　与相应事件相关联的参数，不同的事件附带不同的参数，关于事件类型及其它参数的详细解释，请参考事件类型的相关章节。

返　　回：　如果有事件，返回 BF_OK，相应的事件内容存于 pMessageBuffer 中。

注　　释：　一定在本函数返回 BF_OK 时，pMessageBuffer 中的内容才有意义。
用户应用程序需要不断的循环调用该函数，取得各个通道的事件，并进行处理或根据实际需要发送命令，才能保证 BF-VOICE 语音卡系统的正常运行。一般情况下，在初始化 BF-VOICE 语音卡和打开通道之后，便开始循环调用该函数。

▲　BF_CheckMessage

函数原型：　unsigned short WINAPI BF_CheckMessage(unsigned short WaitTime,
PBF_MESSAGE_INFO pMessageBuffer);

功　　能：　检测 BF-VOICE 语音卡的是否有事件，该函数并不将事件从事件队列中取出。

参　　数：　WaitTime　　指示如果没有事件，该函数可以被挂起的时间，单位毫秒。
WaiTime 的取值可以从 0 至 0xFFFF，建议取值为 50。
如果 WaiTime 的取值为 0，则不管是否有事件，都立即返回；
如果 WaiTime 的取值为 0xFFFF，则挂起该调用，直到有事件才返回；
如果 WaiTime 的取值为 0x0001 至 0xFFFE 之间的任意一个值，则为挂起的时间数量，其单位为毫秒。

pMessageBuffer　指向 PBF_MESSAGE_INFO 结构的地址指针，用于存放返回的事件，PBF_MESSAGE_INFO 的结构定义如下：

```
typedef struct _BF_MESSAGE_INFO{
    USHORT      MessageCode;
    USHORT      ChannelNo;
    USHORT      Parameter[6];
} BF_MESSAGE_INFO, *PBF_MESSAGE_INFO;
```

如果该函数返回 BF_OK，则表示有事件发生，事件类型以及其它指示存于上面结构中。PBF_MESSAGE_INFO 结构中的各项含义如下：

MessageCode 指示事件的类型，取值见《第四部分——第四章——BF-VOICE 语音卡编程接口事件类型说明》

ChannelNo　指示发生事件的逻辑通道号码。

Parameter　与相应事件相关联的参数，不同的事件附带不同的参数，关于事件类型及其它参数的详细解释，请参考事件类型的相关章节。

返　　回：　如果有事件，返回 BF_OK，相应的事件内容存于 pMessageBuffer 中。

注　　释：　一定在本函数返回 BF_OK 时，pMessageBuffer 中的内容才有意义。
为确保下一次能取到新的事件，用户应用程序必须在调用该函数后，再调用函数 BF_GetMessageForWindows 将事件队列中的事件取出。

4.1.2.5 侦听和连接通道接口函数

▲ BF_ListenChannel

函数原型：　unsigned short WINAPI BF_ListenChannel(unsigned short ChannelNo, unsigned short SourceChannel);

功　　能：　侦听通道，即逻辑通道号码为 ChannelNo 的通道侦听逻辑通道号码为 SourceChannel 的通道。

参　　数：　ChannelNo　　目的通道的逻辑号码
SourceChannel　　源通道的逻辑号码

返　　回：　如果调用成功，返回 BF_OK

注　　释：　当函数成功调用后，目的通道的信息来源于被侦听的通道，而在这种情况下发送放音命令时，将无法正确放音。

▲ BF_UnListenChannel

函数原型：　unsigned short WINAPI BF_UnListenChannel(unsigned short ChannelNo);

功　　能：　停止侦听，即停止逻辑通道号码为 ChannelNo 的通道侦听其它通道。

参　　数：　ChannelNo　　目的通道的逻辑号码

返　　回：　如果调用成功，返回 BF_OK

注　　释：

▲ BF_LinkChannel

函数原型：　unsigned short WINAPI BF_LinkChannel(unsigned short FirstChannelNo, unsigned short SecondChannelNo);

功　　能：　连接两个通道，即 FirstChannelNo 的通道侦听 SecondChannelNo 的通道，SecondChannelNo 的通道侦听 FirstChannelNo 的通道

参　　数：　FirstChannelNo　　通道的逻辑号码

SecondChannelNo　　通道的逻辑号码

返　　回：　如果调用成功，返回 BF_OK

注　　释：

▲ BF_LinkUnChannel

函数原型：　unsigned short WINAPI BF_UnLinkChannel(unsigned short FirstChannelNo,unsigned short SecondChannelNo);;

功　　能：　拆除两个通道之间的连接

参　　数：　FirstChannelNo　　通道的逻辑号码

SecondChannelNo　　通道的逻辑号码

返　　回：　如果调用成功，返回 BF_OK

注　　释：

4.1.2.6　两线模拟中继模块接口函数

▲ BF_TrunkHookOff

函数原型：　unsigned short WINAPI BF_TrunkHookOff(unsigned short ChannelNo)

功　　能：　设置中继模块摘机。

参　　数：　ChannelNo　逻辑通道号码。

返　　回：　如果调用成功，返回 BF_OK。

注　　释：　对于中继模块通道，当有振铃事件或需要向外拨号时，必须首先摘机，才能进行其它操作。如在有振铃事件时，先摘机，然后进行放音、录音或 DTMF 识别等操作。

▲ BF_TrunkHookOn

函数原型：　unsigned short WINAPI BF_TrunkHookOn(unsigned short ChannelNo)

功　　能：　设置中继模块挂机。

参　　数：　ChannelNo　逻辑通道号码。

返　　回：　如果调用成功，返回 BF_OK

注　　释：　对于中继模块通道，当双方通话完毕或检测到对方挂机时，应该挂机。

▲ BF_StartDetectTrunkPolarity

函数原型：　unsigned short BF_StartDetectTrunkPolarity(unsigned short ChannelNo)

功　　能：　开始检测中继模块的极性反转，该函数主要用于在特殊情况下，需要检测极性反转时，使用该函数如果检测到有极性反转，则送出事件 MESSAGE_TRUNK_HAVE_POLE，并停止检测极性反转；如果需要停止检测，可以调用 BF_StopDetectTrunkPolarity 函数。

当用于呼出时，判定对方是否摘机的极性反转，系统自动检测，无须调用该函数。

参　　数：　ChannelNo　逻辑通道号码。

返　　回：　如果调用成功，返回 BF_OK

注　　释：　在使用本功能时，必须首先向电信公司申请极性反转功能。

▲ BF_StopDetectTrunkPolarity

函数原型：　unsigned short BF_StopDetectTrunkPolarity(unsigned short ChannelNo)

功　　能：　停止检测中继模块的极性反转。

参　　数：　ChannelNo　逻辑通道号码。

返　　回：　如果调用成功，返回 BF_OK

注　　释：

▲ BF_StartTrunkPickUp

函数原型：　unsigned short BF_StartTrunkPickUp(unsigned short ChannelNo,unsigned short TimeByMilliSecond)

功　　能：　启动中继模块的拍插动作。该函数实现中继模块挂机，然后等待指定时间（TimeByMilliSecond）后再摘机，同时发送拍插完毕事件。

参　　数：　ChannelNo　　逻辑通道号码。

TimeByMilliSecond　拍插时间，以毫秒为单位

返　　回：　如果调用成功，返回 BF_OK

注　　释：

▲ BF_StopTrunkPickUp

函数原型：　unsigned short BF_StopTrunkPickUp(unsigned short ChannelNo)

功　　能：　停止中继模块的拍插动作。该函数将立即将相应的中继模块摘机。

参　　数：　ChannelNo　逻辑通道号码。

返　　回：　如果调用成功，返回 BF_OK

注　　释：

4.1.2.7 用户模块接口函数

▲ BF_UserRingOn

函数原型：　unsigned short BF_UserRingOn(unsigned short ChannelNo,unsigned short RingOnTime, unsigned short RingOffTime, unsigned short TotalTimes)

功　　能：　设置用户模块振铃。振铃信号是周期性的，每个周期由两段组成，即有无，每一段时间均可设定，总的振铃周期数量也可以设定。在振铃完毕时，会返回振铃结束(MESSAGE_USER_RING_END)事件，如果在振铃期间发送停止振铃命令，则立即停止振铃，并且没有任何事件返回。

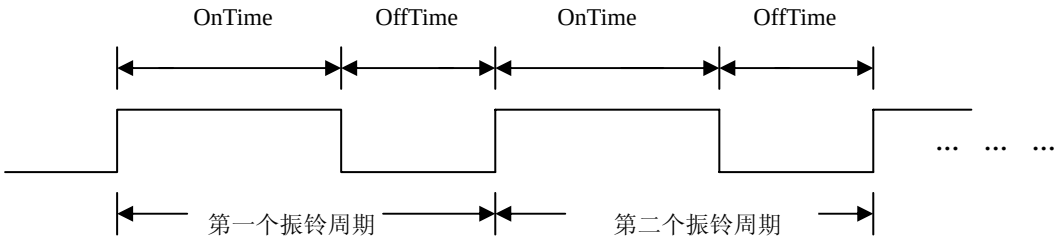
参　　数：　ChannelNo　逻辑通道号码

RingOnTime　代表振铃时间，单位毫秒，取值 0x0000—0xFFFF。

RingOffTime 代表不振铃时间，单位毫秒，取值 0x0000—0xFFFF。

TotalTimes　代表总的振铃周期数量，取值 0x0001--0xFFFF。

送出的振铃信号波形如下：



返　　回：　如果调用成功，返回 BF_OK
注　　释：

▲ BF_UserRingOff

函数原型：　unsigned short BF_UserRingOff(unsigned short ChannelNo)
功　　能：　停止用户模块振铃。
参　　数：　ChannelNo　逻辑通道号码。
返　　回：　如果调用成功，返回 BF_OK
注　　释：

▲ BF_SetUserPickUpTime

函数原型：　BOOL unsigned short BF_SetUserPickUpTime(unsigned short TimeByMilliSecond)
功　　能：　设置用户模块拍插时间。
参　　数：　TimeByMilliSecond　以毫秒为单位的时间，该缺省值为 500ms。
返　　回：　如果调用成功，返回 BF_OK
注　　释：　该函数对整个系统的所有用户模块均发生作用，当用户模块有拍插动作时，BF-VOICE 语音卡发送 MESSAGE_USER_PICK_UP 事件。
所谓的拍插动作，就是当用户模块由摘机状态转换到挂机状态，然后再回到摘机状态的总共时间小于一定时间的动作。

4.1.2.8 录音接口函数

▲ BF_StartRecordFile

函数原型：　unsigned short BF_StartRecordFile(unsigned short ChannelNo,char *FileName,unsigned long Offset,unsigned long Length,char StopChar)
功　　能：　开始文件录音，将录音数据存于文件。可以设置录音长度，或设置在收到某个或某些 DTMF 按键时，停止录音。当录音数据达到要求时，停止录音，并送出事件 MESSAGE_RECORD_VOICE_END；当收到相应的 DTMF 按键时，停止录音，并送出事件 MESSAGE_RECORD_VOICE_EXIT。
参　　数：　ChannelNo　通道的逻辑号码。
FileName　录音文件名称，含有文件存放的路径，最大长度为 MAX_FILE_NAME_NUM-1。
Offset　如果录音文件存在，存放语音的起始位置，如果录音文件不存在，必须设置为 0。
Length　需要录音数据的字节长度，如果设为 0xFFFFFFFF，则一直录音，直到发送录音停止命令，或有指定的 DTMF 收到。

StopChar 停止录音的 DTMF，当收到相应的 DTMF 按键时，停止录音，有效的取值为：

0x00 表示收到任何 DTMF 按键都不停止

0x01 表示收到任一 DTMF 按键就停止

0—9,*,#,A,B,C,D 表示收到某个 DTMF 按键就停止

其他值 没有意义

返　回：如果调用成功，返回 BF_OK

注　释：如果要使 StopChar 生效，必须首先调用识别 DTMF 函数 BF_StartDetectDTMF

▲ BF_StartRecordBuffer

函数原型：unsigned short BF_StartRecordBuffer(unsigned short ChannelNo,char *Buffer,unsigned long Offset,unsigned long Length,char StopChar)

功　能：开始缓存区录音，将录音数据存于缓存区。必须设置录音长度，可以设置在收到某个或某些 DTMF 按键时，停止录音。当录音数据达到要求时，停止录音，并送出事件 MESSAGE_RECORD_VOICE_END；当收到相应的 DTMF 按键时，停止录音，并送出事件 MESSAGE_RECORD_VOICE_EXIT。

参　数：ChannelNo 通道的逻辑号码。

Buffer 录音缓存区，

Offset 存放语音的起始位置

Length 需要录音数据的字节长度，如果设为 0xFFFFFFFF，则一直录音，直到发送录音停止命令，或有指定的 DTMF 收到。

StopChar 停止录音的 DTMF，当收到相应的 DTMF 按键时，停止录音，有效的取值为：

0x00 表示收到任何 DTMF 按键都不停止

0x01 表示收到任一 DTMF 按键就停止

0—9,*,#,A,B,C,D 表示收到某个 DTMF 按键就停止

其他值 没有意义

返　回：如果调用成功，返回 BF_OK

注　释：如果要使 StopChar 生效，必须首先调用识别 DTMF 函数 BF_StartDetectDTMF

▲ BF_StartRecordCircleBuffer

函数原型：unsigned short BF_StartRecordCircleBuffer(unsigned short ChannelNo, char *Buffer,unsigned long TotalLength,char StopChar)

功　能：开始循环缓存区录音

参　数：ChannelNo 通道的逻辑号码

Buffer 缓存区指针

TotalLength 循环缓存区的大小，单位为字节，其最小取值为 0x400,必须成倍递增,如 0x400,0x800,0x1000 等，但不能为 0xC00。

StopChar 指定如果通道收到某个或某些 DTMF 按键，停止产生信号音，有效取值如下：

0x00 表示收到任何 DTMF 按键都不停止

0x01 表示收到任一 DTMF 按键就停止

0—9,*,#,A,B,C,D 表示收到某个 DTMF 按键就停止

其他值 没有意义

返　　回：　如果调用成功，返回 BF_OK

注　　释：　循环缓存区录音是指 BF-VOICE 系统在给定的缓存区中，循环录音，在录音数据达到至整个缓存区后，再从缓存区的开始之处录音，并且在数据达到整个缓存区或整个缓存区一半的时候，分别发送事件 MESSAGE_RECORD_CIRCLE_SECOND_END 和 MESSAGE_RECORD_CIRCLE_FIRST_END 。

▲ BF_StopRecordVoice

函数原型：　unsigned short BF_StopRecordVoice(unsigned short ChannelNo)

功　　能：　停止录音，停止某个通道的录音操作 。

参　　数：　ChannelNo　通道的逻辑号码。

返　　回：　如果调用成功，返回 BF_OK

注　　释：

▲ BF_GetRecordLength

函数原型：　unsigned short BF_GetRecordLength(unsigned short ChannelNo, unsigned long *RecordLength)

功　　能：　取得当前录音长度。

参　　数：　ChannelNo　　通道的逻辑号码。

RecordLength　存放当前录音长度

返　　回：　如果调用成功，返回 BF_OK

注　　释：

4.1.2.9 放音接口函数

BF-VOICE 系列语音卡一系列功能强大的放音函数，既包括单一的文件或缓存区放音，也包括多文件或多缓存区的索引放音。

对于任何一种放音，如果需要中途停止放音，调用函数 BF_StopPlayVoice 即可。

对于文件索引放音：

BF-VOICE 系统为每一个逻辑通道独立建立一个最大长度为 64 的索引文件表，使用 BF_InitPlayIndexFile 初始化这个索引文件表，即将索引文件表清空；使用 BF_AddPlayIndexFile 增加项目到索引文件表；调用 BF_StartPlayIndexFile 函数进行索引文件放音，该函数从指定通道的索引文件表的第一个文件开始放音，当第一个文件结束时，自动放音下一个，直至索引表中的所有文件放音完毕，发送 MESSAGE_PLAY_VOICE_END 事件。

对于缓存区索引放音：

BF-VOICE 系统仅仅建立一个最大数量为 256 的缓存区索引表，所有通道的索引缓存区放音都使用相同的这个索引表，调用 BF_InitPlayIndexBuffer 函数来初始化这个索引表，即将索引表清空；调用 BF_AddPlayIndexBuffer 函数增加表中的项目，每个缓存区的加入是顺序进行的；调用 BF_StartPlayIndexBuffer 函数启动索引缓存区放音。

对于索引文件放音的应用，举例如下：

如有 0、1、2、年、月、日的语音文件，文件名分别为：Zero.tel、One.tel、Two.tel、Year.tel、Month.tel、Day.tel，那么要在第八通道放音“2001 年 2 月 1 日”这段话，只需如此操作：

```
BF_InitPlayIndexFile(8);
```

```
BF_AddPlayIndexFile(8,"Two.tel");
BF_AddPlayIndexFile(8,"Zero.tel");
BF_AddPlayIndexFile(8,"Zero.tel");
BF_AddPlayIndexFile(8,"One.tel");
BF_AddPlayIndexFile(8,"Year.tel");
BF_AddPlayIndexFile(8,"Two.tel");
BF_AddPlayIndexFile(8,"Month.tel");
BF_AddPlayIndexFile(8,"One.tel");
BF_AddPlayIndexFile(8,"Day.tel");
BF_StartPlayIndexFile(8, StopChar);
```

对于索引文件放音的应用，举例如下：

应用系统首先申请足够数量的缓存区，然后将频繁使用的语音读入相应的缓存区，再建立索引表，即可使用索引缓存区放音。

如有 0、1、2、年、月、日的语音，缓存区的指针分别为：Zero、One、Two、Year、Month、Day，那么可以如下建立缓存区索引表：

```
BF_InitPlayIndexBuffer();
BF_AddPlayIndexBuffer(Zero, xxxx);
BF_AddPlayIndexBuffer(One, xxxx);
BF_AddPlayIndexBuffer(Two, xxxx);
BF_AddPlayIndexBuffer(Year, xxxx);
BF_AddPlayIndexBuffer(Month, xxxx);
BF_AddPlayIndexBuffer(Day, xxxx);
```

其中 xxxx 代表相应缓存区字节长度

那么，无论以后哪个通道进行索引缓存区放音，均使用这个索引表，如要在第八通道放音“2001 年 2 月 1 日”这段话，只需如此操作：

```
unsigned short Buffertable[9]={2,0,0,1,3,2,4,1,5};
BF_StartPlayIndexBuffer(8,BufferTable,9,StopChar)
```

其中 BufferTable 项的值，就是各个语音缓存区在缓存区索引表中的顺序号。

对于循环缓存区放音，是指 BF-VOICE 系统在给定的缓存区中，循环放音，在放音数据达到至整个缓存区后，再从缓存区的开始处放音，并且在数据达到整个缓存区或整个缓存区一半的时候，分别发送事件 MESSAGE_PLAY_CIRCLE_SECOND_END 和 MESSAGE_PLAY_CIRCLE_FIRST_END 。

下面详细叙述各种放音相关函数；

▲ BF_StartPlayFile

函数原型： unsigned short BF_StartPlayFile(unsigned short ChannelNo,char *FileName,unsigned long Offset,unsigned long Length,char StopChar)

功　　能： 开始文件放音，放音数据存于文件。当发送停止放音命令，或收到某个或某些设定的停止 DTMF 按键时，停止放音。当所有要求的放音数据放完时，停止放音，并送出事件 MESSAGE_PLAY_END；当收到相应的 DTMF 按键时，停止放音，并送出事件 MESSAGE_PLAY_EXIT。

参　　数：

ChannelNo	通道的逻辑号码。
FileName	如果放音数据在文件中，则指定文件名称，含有文件存放的路径，最大长度为 MAX_FILE_NAME_NUM-1
Ofsset	指定放音文件的字节偏移长度

Length　　指定放音文件的字节长度，如果设为 0xFFFFFFFF，则表示一直到文件结尾

StopChar　　停止录音的 DTMF，当收到相应的 DTMF 按键时，停止录音，有效的取值为：

0x00　　表示收到任何 DTMF 按键都不停止

0x01　　表示收到任一 DTMF 按键就停止

0—9,*,#,A,B,C,D　表示收到某个 DTMF 按键就停止

其他值　　没有意义

返　　回：　如果调用成功，返回 BF_OK

注　　释：　如果要使 StopChar 生效，必须首先调用识别 DTMF 函数 BF_StartDetectDTMF

▲ BF_StartPlayBuffer

函数原型：　unsigned short BF_StartPlayFile(unsigned short ChannelNo,char *Buffer,unsigned long Offset,unsigned long Length,char StopChar)

功　　能：　开始缓存区放音，放音数据存于缓存区。当发送停止放音命令，或收到某个或某些设定的停止 DTMF 按键时，停止放音。当所有要求的放音数据放完时，停止放音，并送出事件 MESSAGE_PLAY_END；当收到相应的 DTMF 按键时，停止放音，并送出事件 MESSAGE_PLAY_EXIT。

参　　数：　ChannelNo　　通道的逻辑号码。

Buffer　　放音数据缓存区。

Ofsset　　指定放音文件的字节偏移长度

Length　　指定放音文件的字节长度，一定要设置，以防越界

StopChar　　停止录音的 DTMF，当收到相应的 DTMF 按键时，停止录音，有效的取值为：

0x00　　表示收到任何 DTMF 按键都不停止

0x01　　表示收到任一 DTMF 按键就停止

0—9,*,#,A,B,C,D　表示收到某个 DTMF 按键就停止

其他值　　没有意义

返　　回：　如果调用成功，返回 BF_OK

注　　释：　如果要使 StopChar 生效，必须首先调用识别 DTMF 函数 BF_StartDetectDTMF

▲ BF_InitPlayIndexFile

函数原型：　unsigned short BF_InitPlayIndexFile(unsigned short ChannelNo)

功　　能：　初始化索引放音文件，该函数清空指定通道的索引放音。

参　　数：　ChannelNo　　通道的逻辑号码。

返　　回：　如果调用成功，返回 BF_OK

注　　释：

▲ BF_AddPlayIndexFile

函数原型：　unsigned short BF_AddPlayIndexFile(unsigned short ChannelNo, char *FileName)

功　　能：　向指定通道内增加索引文件

参　　数：　ChannelNo　　通道的逻辑号码。

FileName　　放音文件名

返　　回：　如果调用成功，返回 BF_OK

注　　释：

▲ BF_StartPlayIndexFile

函数原型：　unsigned short BF_StartPlayIndexFile(unsigned short ChannelNo, char StopChar)

功　　能：　开始索引文件放音

参　　数：　ChannelNo　通道的逻辑号码。

StopChar　停止录音的 DTMF，当收到相应的 DTMF 按键时，停止录音，有效的取值为：

0x00　表示收到任何 DTMF 按键都不停止

0x01　表示收到任一 DTMF 按键就停止

0—9,*,#,A,B,C,D　表示收到某个 DTMF 按键就停止

其他值　没有意义

返　　回：　如果调用成功，返回 BF_OK

注　　释：　如果需要停止索引文件放音，调用 BF_StopPlayVoice 函数

▲ BF_InitPlayIndexBuffer

函数原型：　unsigned short BF_InitPlayIndexBuffer(void)

功　　能：　初始化缓存区索引，即将索引放音的缓存区清空

参　　数：　无

返　　回：　如果调用成功，返回 BF_OK

注　　释：

▲ BF_AddPlayIndexBuffer

函数原型：　unsigned short BF_AddPlayIndexBuffer(char *Buffer,unsigned long Length)

功　　能：　添加索引缓存区

参　　数：　Buffer　缓存区

Length　缓存区长度

返　　回：　如果调用成功，返回 BF_OK

注　　释：

▲ BF_StartPlayIndexBuffer

函数原型：　unsigned short BF_StartPlayIndexBuffer(unsigned short ChannelNo, unsigned short *IndexTable, unsigned short IndexCount, char StopChar)

功　　能：　开始索引缓存区放音

参　　数：　ChannelNo　通道的逻辑号码。

IndexTable　索引表，该索引表实际上是一个数组，该数组的每一个项指代缓存区索引表中的相应缓存区的顺序号。

IndexCount　索引表中的项数

StopChar　停止录音的 DTMF，当收到相应的 DTMF 按键时，停止录音，有效的取值为：

0x00　　表示收到任何 DTMF 按键都不停止
0x01　　表示收到任一 DTMF 按键就停止
0—9,*,#,A,B,C,D 表示收到某个 DTMF 按键就停止
其他值　　没有意义

返　　回：　如果调用成功，返回 BF_OK
注　　释：　如果需要停止索引文件放音，调用 BF_StopPlayVoice 函数

▲ BF_StartPlayCircleBuffer

函数原型：　unsigned short BF_StartPlayCircleBuffer(unsigned short ChannelNo, char *Buffer,unsigned long TotalLength,char StopChar)
功　　能：　开始循环缓存区放音
参　　数：　ChannelNo　通道的逻辑号码
Buffer　缓存区指针
TotalLength　循环缓存区的大小，单位为字节，其最小取值为 0x400, 必须成倍递增,如 0x400,0x800,0x1000 等，但不能为 0xC00。
StopChar　指定如果通道收到某个或某些 DTMF 按键，停止产生信号音，有效取值如下：
0x00　　表示收到任何 DTMF 按键都不停止
0x01　　表示收到任一 DTMF 按键就停止
0—9,*,#,A,B,C,D 表示收到某个 DTMF 按键就停止
其他值　　没有意义
返　　回：　如果调用成功，返回 BF_OK
注　　释：

▲ BF_StopPlayVoice

函数原型：　unsigned short BF_StopPlayVoice(unsigned short ChannelNo)
功　　能：　停止放音操作。
参　　数：　ChannelNo　通道的逻辑号码。
返　　回：　如果调用成功，返回 BF_OK
注　　释：　所有的放音（如文件、缓存区、索引文件和索引缓存区）停止，都调用该函数。

▲ BF_GetPlayLength

函数原型：　unsigned short BF_GetPlayLength(unsigned short ChannelNo, unsigned long *PlayLength)
功　　能：　取得当前实际放音长度。
参　　数：　ChannelNo　　通道的逻辑号码。
PlayLength　　存放实际放音长度
返　　回：　如果调用成功，返回 BF_OK
注　　释：

4.1.2.10　DTMF 识别和产生接口函数

▲ BF_StartDetectDTMF

函数原型：　unsigned short BF_StartDetectDTMF(unsigned short ChannelNo, unsigned

short SmartLevel)

功　　能：开始识别 DTMF 按键，在识别到 DTMF 时，发送事件(MESSAGE_HAVE_DTMF)，并继续识别，直到发送停止识别命令。应用程序可以随时调用函数 BF_GetDTMFKey 来获取已经识别到的 DTMF 按键。

参　　数：ChannelNo　通道的逻辑号码。

SmartLevel　指定识别 DTMF 的灵敏度，取值为 0-2，数字越大越灵敏，但应注意，灵敏度越大，识别的准确率可能降低。通常取 0

返　　回：如果调用成功，返回 BF_OK

注　　释：一般情况下，灵敏度取缺省值 0

▲ BF_StopDetectDTMF

函数原型：unsigned short BF_StopDetectDTMF(unsigned short ChannelNo)

功　　能：停止检测 DTMF 按键。

参　　数：ChannelNo　通道的逻辑号码。

返　　回：如果调用成功，返回 BF_OK

注　　释：

▲ BF_ClearDTMFBuffer

函数原型：unsigned short BF_ClearDTMFBuffer(unsigned short ChannelNo)

功　　能：清除某一通道已经识别出来的但仍然没有被取走的 DTMF 按键。

参　　数：ChannelNo　通道的逻辑号码。

返　　回：如果调用成功，返回 BF_OK

注　　释：该函数多数用在已经调用开始识别 DTMF 按键后，在获取新的按键之前，清除通道内残留的没有用的 DTMF 按键。

▲ BF_GetDTMFKey

函数原型：unsigned short BF_GetDTMFKey(unsigned short ChannelNo, unsigned short DTMFCount,char *DTMFKey)

功　　能：获取 DTMF 按键，从 ChannelNo 通道内提取数量为 DTMFCount 个按键。如果含有足够数量的 DTMF 按键，则将按键依次添入 DTMFKey 所指的缓存区中，并返回 BF_OK；否则返回其他错误码。所有 DTMF 按键均为 ASCII 码，字母为大写。

参　　数：ChannelNo　通道的逻辑号码。

DTMFKey　存放取回的 DTMF 按键，数量为 DTMFCount。

DTMFCount　要取回的 DTMF 按键数量，取值为 1——(MAX_DTMF_NUM-1)。

返　　回：如果含有所需数目的 DTMF 按键，返回 BF_OK，按键存于 DTMFKey 中

注　　释：

▲ BF_GenerateDTMFString

函数原型：unsigned short BF_GenerateDTMFString(unsigned short ChannelNo, char *DTMFString)

功　　能：产生 DTMF 按键。

参　　数：ChannelNo　通道的逻辑号码。

DTMFString　要产生的 DTMF 按键的字符串，以 0x00 结尾，数量小于 MAX_DTMF_NUM-1。有效的 ASCII 字符为："0"—"9"、"*"、"#"、"A"、

“a”、”B”、”b”、”C”、”c”、”D”、”d” 、” ，”

返　　回：　如果调用成功，返回 BF_OK

注　　释：　符号“,”表示延时 500ms

4.1.2.11　信号音产生接口函数

▲　BF_StartGenerateSignal

函数原型：　unsigned short BF_StartGenerateSignal(unsigned short ChannelNo, unsigned short Frequency0, unsigned short Frequency1, unsigned short OnTime, unsigned short OffTime, unsigned short TotalTimes, char StopChar)

功　　能：　产生 300Hz—3400Hz 之间的一个或两个频率的有无间断的周期性声音，每一个周期分为两段，第一段产生信号音，第二段不产生信号音。并在产生完毕时，发送事件（MESSAGE_GENERATE_SIGNAL_END）；如果有指定的停止发送信号音的 DTMF 按键，则在收到指定的停止发送信号音的 DTMF 按键时，发送事件（MESSAGE_GENERATE_SIGNAL_EXIT），然后发送有 DTMF 事件；如果在产生信号音中间发送停止发送信号音命令，则立即停止发送信号音，并且没有任何事件产生。

参　　数：

ChannelNo	通道的逻辑号码。
Frequency0	指定第一个频率的频率值，单位赫兹（Hz）。
Frequency1	指定第二个频率的频率值，单位赫兹（Hz）。
OnTime	指定产生信号音的时间，单位毫秒，取值 0x0000-0x7FFF。
OffTime	指定不产生信号音的时间，单位毫秒，取值 0x0000-0x7FFF。
TotalTimes	指定产生信号音的周期次数。
StopChar	指定如果通道收到某个或某些 DTMF 按键，停止产生信号音，有效取值如下： 0x00　表示收到任何 DTMF 按键都不停止 0x01　表示收到任一 DTMF 按键就停止 0—9,*,#,A,B,C,D　表示收到某个 DTMF 按键就停止 其他值　没有意义

返　　回：　如果调用成功，返回 BF_OK

注　　释：　如果要停止产生信号音，必须调用函数 BF_StopGenerateSignal。

如果要使 StopChar 生效，必须首先调用识别 DTMF 函数 BF_StartDetectDTMF

▲　BF_StopGenerateSignal

函数原型：　unsigned short BF_StopGenerateSignal(unsigned short ChannelNo)

功　　能：　停止产生信号音。

参　　数：　ChannelNo　通道的逻辑号码。

返　　回：　如果调用成功，返回 BF_OK

注　　释：

4.1.2.12　标准信号音识别接口函数

▲　BF_SetToneInfo

函数原型：　unsigned short BF_SetToneInfo(unsigned short ChannelNo,PBF_TONE_INFO ToneInfo)

功　　能：　设置标准信号音的参数。一般情况下，不须调用该函数

参　　数：　ChannelNo　通道的逻辑号码。

ToneInfo　指向 PBF_TONE_INFO 结构的地址指针，用于存放标准信号音的各种参数，PBF_TONE_INFO 的结构定义如下：

```
typedef struct _BF_TONE_INFO{
        unsigned short      DialToneMinTotalTime;
        unsigned short      BusyToneMaxTotalTime;
        unsigned short      BusyToneMinTotalTime;
        unsigned short      BusyToneRatio;
        unsigned short      BusyToneErrorBetweenTwo;
        unsigned short      BusyToneTotalTimes;
        unsigned short      BackToneMaxTotalTime;
        unsigned short      BackToneMinTotalTime;
        unsigned short      BackToneRatio;
        unsigned short      BackToneErrorBetweenTwo;
        unsigned short      BackToneTotalTimes;
    }BF_TONE_INFO, *PBF_TONE_INFO;
```

PBF_TONE_INFO 结构中的各项含义如下：

代号	缺省值	含义
DialToneMinTotalTime	拨号音持续最小时间(毫秒)	2500
BusyToneMaxTotalTime	最大忙音周期时间(毫秒)	1500
BusyToneMinTotalTime	最小忙音周期时间(毫秒)	500
BusyToneRatio	忙音周期占空比	50
BusyToneErrorBetweenTwo	两个忙音周期的误差比	10
BusyToneTotalTimes	忙音周期数量	3
BackToneMaxTotalTime	最大回铃音周期时间(毫秒)	6000
BackToneMinTotalTime	最小回铃音周期时间(毫秒)	4000
BackToneRatio	回铃音周期占空比	20
BackToneErrorBetweenTwo	两个回铃音周期的误差比	10
BackToneTotalTimes	回铃音周期数量	3

返　　回：　如果调用成功，返回 BF_OK

注　　释：

▲ BF_SetToneFrequency

函数原型：　unsigned short BF_SetToneFrequency(unsigned short ChannelNo,
unsigned short Frequency)

功　　能：　设置信号音的频率。在调用该函数后，对于指定的通道，调用函数 BF_StatartDetectDialTone、BF_StartDetectBusyTone、BF_StartDetectback 以及 BF_StartDetectToneData 所识别的频率，均为这个函数设定的频率。没有调用该函数时的缺省频率为 450 赫兹。

参　　数：　ChannelNo　通道的逻辑号码。

Frequency　频率值，单位为赫兹，取值范围为 300--3500

返　　回：　如果调用成功，返回 BF_OK

注　　释：

▲ BF_StartDetectDialTone

函数原型： unsigned short BF_StartDetectDialTone(unsigned short ChannelNo)

功　　能： 开始识别拨号音，当识别到拨号音时，发送 MESSAGE_HAVE_DIAL_TONE(有拨号音)事件给应用程序。该信号音的频率可以调用函数 BF_SetToneFrequency 设定，缺省值为 450Hz。

参　　数： ChannelNo　通道的逻辑号码

返　　回： 如果调用成功，返回 BF_OK

注　　释： 该函数仅仅识别标准的拨号音，这个函数常常在中继模块的摘机之后，而在拨号之前调用，以验证线路是否可以拨号。在检测到标准拨号音后，BF-VOICE 语音卡继续检测标准拨号音，并不停止；如果要停止检测，必须调用函数 BF_StopDetectDialTone。

▲ BF_StopDetectDialTone

函数原型： unsigned short BF_StopDetectDialTone(unsigned short ChannelNo)

功　　能： 停止检测标准拨号音。

参　　数： ChannelNo　通道的逻辑号码。

返　　回： 如果调用成功，返回 BF_OK

注　　释：

▲ BF_StartDetectBackTone

函数原型： unsigned short BF_StartDetectBackTone(unsigned short ChannelNo)

功　　能： 开始识别标准的回铃音，当识别到回铃音时，发送 MESSAGE_HAVE_BACK_TONE（有回铃音）事件给应用程序。该信号音的频率可以调用函数 BF_SetToneFrequency 设定，缺省值为 450Hz。

参　　数： ChannelNo　通道的逻辑号码。

返　　回： 如果调用成功，返回 BF_OK。

注　　释： 该函数仅仅识别标准的回铃音，这个函数常常在中继模块的拨号之后调用，以验证线路是否接通。

在检测到标准回铃音后，BF-VOICE 语音卡继续检测标准回铃音，并不停止；如果要停止检测，必须调用函数 BF_StopDetectBackTone。

▲ BF_StopDetectBackTone

函数原型： unsigned short BF_StopDetectBackTone(unsigned short ChannelNo)

功　　能： 停止检测标准回铃音。

参　　数： ChannelNo　通道的逻辑号码。

返　　回： 如果调用成功，返回 BF_OK

注　　释：

▲ BF_StartDetectBusyTone

函数原型： unsigned short BF_StartDetectBusyTone(unsigned short ChannelNo)

功　　能： 开始识别标准的忙音，当识别到忙音时，发送 MESSAGE_HAVE_BUSY_TONE(有忙音)事件给应用程序。该信号音的频率可以调用函数 BF_SetToneFrequency 设定，缺省值为 450Hz。

参　　数：　ChannelNo　通道的逻辑号码。
返　　回：　如果调用成功，返回 BF_OK
注　　释：　该函数仅仅识别标准的忙音，这个函数常常在中继模块拨号之后调用，以验证线路是否接通；或在双方通话之中调用，以检测对方是否挂机。
在检测到标准忙音后，BF-VOICE 语音卡继续检测标准忙音，并不停止；如果要停止检测，必须调用函数 BF_StopDetectBusyTone。

▲ BF_StopDetectBusyTone

函数原型：　unsigned short BF_StopDetectBusyTone(unsigned short ChannelNo)
功　　能：　停止检测标准忙音。
参　　数：　ChannelNo　通道的逻辑号码。
返　　回：　如果调用成功，返回 BF_OK
注　　释：

▲ BF_StartDetectToneData

函数原型：　unsigned short BF_StartDetectToneData(unsigned short ChannelNo)
功　　能：　识别标准频率的声音，定时送出识别的结果数据事件（MESSAGE_HAVE_TONE_DATA)。识别结果数据由 16 位的一个字表示，这个字的每一位的 1 或 0 分别代表有要识别的语音和没有要识别的语音。该信号音的频率可以调用函数 BF_SetToneFrequency 设定，缺省值为 450Hz。
参　　数：　ChannelNo　通道的逻辑号码。
返　　回：　如果调用成功，返回 BF_OK
注　　释：　要停止识别，必须调用函数 BF_StopDetectToneData。

▲ BF_StopDetectToneData

函数原型：　unsigned short BF_StopDetectToneData(unsigned short ChannelNo)
功　　能：　停止识别信号音数据
参　　数：　ChannelNo　通道的逻辑号码
返　　回：　如果调用成功，返回 BF_OK
注　　释：

4.1.2.13 特殊信号音识别接口函数

▲ BF_SetSpecialSignalFrequency

函数原型：　unsigned short BF_SetSpecialSignalFrequency(unsigned short ChannelNo, unsigned short Frequency);
功　　能：　设置指定通道的特殊信号音，以便启动函数 BF_StartDetectSpecialSignal 开始检测。
参　　数：　ChannelNo　通道的逻辑号码
Frequency　特别频率，单位赫兹
返　　回：　如果调用成功，返回 BF_OK
注　　释：　该功能对检测传真等信号特别有用

▲ BF_StartDetectSpecialSignal

函数原型：　unsigned short BF_StartDetectSpecialSignal(unsigned short

ChannelNo,unsigned short TimeByMilliSecond);

功　　能：该函数识别指定通道是否有特殊的信号音持续一段时间，特殊的信号音的频率有函数 BF_SetSpecialSignalFrequency 设定。

参　　数：ChannelNo　　通道的逻辑号码
TimeByMilliSecond　持续时间，单位为毫秒

返　　回：如果调用成功，返回 BF_OK

注　　释：在检测到该特殊信号音后，BF-VOICE 语音卡发送事件 MESSAGE_HAVE_SPECIAL_SIGNAL，并停止继续检测。

▲ BF_StopDetectSpecialSignal

函数原型：unsigned short BF_StopDetectSpecialSignal(unsigned short ChannelNo);

功　　能：该函数停止检测指定通道的特别信号音。

参　　数：ChannelNo　　通道的逻辑号码

返　　回：如果调用成功，返回 BF_OK

注　　释：

4.1.2.14 话音/静音识别接口函数

▲ BF_StartDetectHelloVoice

函数原型：unsigned short BF_StartDetectHelloVoice(unsigned short ChannelNo)

功　　能：开始识别话音，当识别到有话音时，送出事件 MESSAGE_HAVE_HELLO_VOICE，并停止识别。如果需要连续识别，需要在收到事件时，再次调用该函数。

参　　数：ChannelNo　通道的逻辑号码。

返　　回：如果调用成功，返回 BF_OK

注　　释：该函数主要用于检测对方摘机，当中继模块拨号完毕，可调用该函数，检测是否有话音，如果有，则表示对方摘机。

▲ BF_StopDetectHelloVoice

函数原型：unsigned short BF_StopDetectHelloVoice(unsigned short ChannelNo)

功　　能：停止识别话音。

参　　数：ChannelNo　通道的逻辑号码。

返　　回：如果调用成功，返回 BF_OK

注　　释：

▲ BF_StartDetectSilence

函数原型：unsigned short BF_StartDetectSilence(unsigned short ChannelNo,
unsigned short ThresholdValue)

功　　能：开始识别静音，该函数每 12.75 毫秒产生一位数据，该位数据表示在这 12.75 毫秒内是否是静音，当该位数据值为 0 时，表示静音，否则不是静音。
当 BF-VOICE 系统检测完 16 个 12.75 毫秒，即有 16 位数据时，送出事件 MESSAGE_HAVE_SILENCE_DATA，并且继续识别，直至应用程序发出停止命令。16 位数据存放在事件的 Parameter[0]字中，并且该字中高位是先检测的数据。

参　　数：ChannelNo　　通道的逻辑号码。
ThresholdValue　语音信号的阀值，即当语音信号小于该值时，才认为是静

音，建议取值为 0x12

返　　回：　如果调用成功，返回 BF_OK

注　　释：　该函数主要用于录音时检测静音。

▲ BF_StopDetectSilence

函数原型：　unsigned short BF_StopDetectSilence(unsigned short ChannelNo)

功　　能：　停止识别静音。

参　　数：　ChannelNo　通道的逻辑号码。

返　　回：　如果调用成功，返回 BF_OK

注　　释：

4.1.2.15 超时设置接口函数

▲ BF_StartTimeOut

函数原型：　unsigned short BF_StartTimeOut (unsigned short ChannelNo,
unsigned short TimeBySecond)

功　　能：　启动某一通道的超时计数，当相应的超时时间到达时，送出事件 MESSAGE_TIME_OUT，然后再重新计时，当相应的超时时间到达时，再送出事件 MESSAGE_TIME_OUT，这样周期性计时，直到停止。

参　　数：　ChannelNo　　通道的逻辑号码。
TimeBySecond　　超时时间，单位秒。

返　　回：　如果调用成功，返回 BF_OK

注　　释：　该函数适用于包括中继模块通道、用户模块通道、录音模块通道。

▲ BF_StopTimeOutForWindows

函数原型：　unsigned short BF_StopTimeOut (unsigned short ChannelNo)

功　　能：　停止超时计时。

参　　数：　ChannelNo　通道的逻辑号码。

返　　回：　如果调用成功，返回 BF_OK

注　　释：　该函数适用于包括中继模块通道、用户模块通道、录音模块通道

▲ BF_SetTimeOutValueForWondows

函数原型：　unsigned short BF_SetTimeOutValue (unsigned short ChannelNo,
unsigned short TimeBySecond)

功　　能：　重新设置超时时间，当已经启动超时计数后，需要改变超时时间时，调用该函数。

参　　数：　ChannelNo　　通道的逻辑号码。
TimeBySecond　　新的超时时间，单位秒。

返　　回：　如果调用成功，返回 BF_OK

注　　释：　该函数适用于包括中继模块通道、用户模块通道、录音模块通道

4.1.2.16 主叫识别接口函数

▲ BF_StartDetectCallerID

函数原型：　unsigned short BF_StartDetectCallerID(unsigned short ChannelNo)

功　　能：　开始主叫识别，当识别到有主叫时，发送 MESSAGE_HAVE_FSK_CID 事件，并停

止识别。如果需要继续识别，必须重新调用该函数。关于主叫信息需要调用 BF_GetFskCallerID 函数取得。

参　　数：　ChannelNo　　通道的逻辑号码。

返　　回：　如果调用成功，返回 BF_OK

注　　释：　该函数适用于包括中继模块通道、用户模块通道、录音模块通道

▲ BF_StopDetectCallerID

函数原型：　unsigned short BF_StopDetectCallerID(unsigned short ChannelNo)

功　　能：　停止主叫识别。

参　　数：　ChannelNo　　通道的逻辑号码。

返　　回：　如果调用成功，返回 BF_OK

注　　释：　该函数适用于包括中继模块通道、用户模块通道、录音模块通道

▲ BF_GetRawFskCallerID

函数原型：　unsigned short BF_GetRawFskCallerID(unsigned short ChannelNo, char *CallerIDString)

功　　能：　获取 FSK 主叫信息，只有在收到 MESSAGE_HAVE_FSK_CID 时，调用该函数，其返回值才有意义。该函数所返回的主叫信息是未经过处理的原始信息。

参　　数：　ChannelNo　　通道的逻辑号码。

CallerIDString　用户提供的存放主叫信息的缓存区，最大为 MAX_CID_NUM

返　　回：　如果调用成功，返回 BF_OK

注　　释：　该函数适用于包括中继模块通道、用户模块通道、录音模块通道

▲ BF_GetFskCallerID

函数原型：　unsigned short BF_GetFskCallerID(unsigned short ChannelNo,char *DateTime,char *TelNo,char *Name)

功　　能：　获取 FSK 主叫信息，只有在收到 MESSAGE_HAVE_FSK_CID 时，调用该函数，其返回值才有意义。该函数所返回的主叫信息是经过处理的信息，即已经分解成主叫日期时间、主叫号码和主叫名称等。

参　　数：　ChannelNo　　通道的逻辑号码。

DateTime　　主叫时间，字符串型，如果没有主叫时间，则返回空串。

TelNo　　主叫号码，字符串型，如果没有主叫号码，则返回空串。

Name　　主叫名称，字符串型，如果没有主叫名称，则返回空串。

返　　回：　如果调用成功，返回 BF_OK

注　　释：　该函数适用于包括中继模块通道、用户模块通道、录音模块通道

对于 DateTime、Telno 和 Name，用户提供的缓存区长度一定要大于相应的最大可显位数，建议最好为 MAX_CID_NUM。

4.1.2.17　发送/接收 FSK 接口函数

▲ BF_StarSendFSK

函数原型：　unsigned short BF_StartSendFsk(unsigned short ChannelNo, unsigned short SeizureFlag,char *Buffer, unsigned short Count)

功　　能：　开始发送 FSK。

参　　数：　ChannelNo　通道的逻辑号码

SeizureFlag　指示是否发送信道占用信号，当为 0 时，不发送信道占用信号；为其他值时，发送信道占用信号。

Buffer　FSK 字符串，最大长度 MAX_FSK_NUM 个字节，字符数量由 Count 参数指定，可以包含 0x00——0xFF 在内的任意字符。

Count　FSK 字符串的字符数量，最大值为 MAX_FSK_NUM

返　回：调用成功，返回 BF_OK。发送完毕，返回发送 FSK 完毕事件。

注　释：

▲ BF_StopSendFSK

函数原型：unsigned short BF_StopSendFsk(unsigned short ChannelNo)

功　能：停止发送 FSK。

参　数：ChannelNo　通道的逻辑号码

返　回：如果调用成功，返回 BF_OK

注　释：

▲ BF_StarDetectFSK

函数原型：unsigned short BF_StartDetectFsk(unsigned short ChannelNo)

功　能：开始识别 FSK。

参　数：ChannelNo　通道的逻辑号码

返　回：调用成功，返回 BF_OK

注　释：

▲ BF_StopDetectFSK

函数原型：unsigned short BF_StopDetectFsk(unsigned short ChannelNo)

功　能：停止识别 FSK。

参　数：ChannelNo　通道的逻辑号码

返　回：如果调用成功，返回 BF_OK

注　释：

▲ BF_GetFSKData

函数原型：unsigned short BF_GetFskData(unsigned short ChannelNo,char *Buffer,unsigned short Count)

功　能：获取识别到的 FSK 数据。

参　数：ChannelNo　通道的逻辑号码

Buffer　用户提供的、用于存放 FSK 数据的缓存区

Count　要获取 FSK 数据的缓存区的字节大小

返　回：如果调用成功，返回实际获取的 FSK 数据的数量，否则返回 BF_OK

注　释：该函数尽量在收到 MESSAGE_HAVE_FSK_DATA 后调用。

4.1.2.18　自动拨号接口函数

▲ BF_AutoDialOut

函数原型：unsigned short BF_AutoDialOut(unsigned short ChannelNo,char *DTMFString)

功　能：该函数完成摘机、检测拨号音、拨号、检测忙音、回铃音、对方摘机等功能，

并返回相应的事件。

参　　数：ChannelNo　通道的逻辑号码

DTMFString　电话号码

返　　回：如果调用成功，返回 BF_OK

注　　释：该函数所返回的事件及含义如下：

事件	含义
MESSAGE_AUTO_DIAL_NO_DIAL_TONE	没有拨号音
MESSAGE_AUTO_DIAL_NONE_LISTEN	对方没人接听
MESSAGE_AUTO_DIAL_HAVE_BUSY	忙音
MESSAGE_AUTO_DIAL_HOOK_OFF	对方摘机

4.1.2.19 语音数据格式转换接口函数

▲ BF_AlawToWave

函数原型：unsigned short WINAPI BF_AlawToWave(char *AlawFileName, char *WaveFileName);

功　　能：将 A-law 格式的纯语音文件转换成 16bit 线性的 Wave 格式的语音文件

参　　数：AlawFileName　A-law 语音文件名称

WaveFileName　Wave 语音文件名称

返　　回：如果调用成功，返回 BF_OK

注　　释：

▲ BF_WaveToAlaw

函数原型：unsigned short WINAPI BF_WaveToAlaw(char *WaveFileName, char *AlawFileName);

功　　能：将 Wave 格式的语音文件转换成 16bit 线性的 A-law 格式的纯语音文件

参　　数：WaveFileName　Wave 语音文件名称

AlawFileName　A-law 语音文件名称

返　　回：如果调用成功，返回 BF_OK

注　　释：

▲ BF_AlawToAdpcm

函数原型：unsigned short WINAPI BF_AlawToAdpcm(char *AlawFileName, char *AdpcmFileName);

功　　能：将 A-law 格式的纯语音文件转换成

参　　数：AlawFileName　A-law 语音文件名称

AdpcmFileName　Adpcm 语音文件名称

返　　回：如果调用成功，返回 BF_OK

注　　释：

▲ BF_AdpcmToAlaw

函数原型：unsigned short WINAPI BF_AdpcmToAlaw(char *AdpcmFileName, char *AlawFileName);

功　　能：将 32Kbit/s 的 Adpcm 格式的语音文件转换成 A-law 格式的纯语音文件

参　　数：AdpcmFileName　Adpcm 语音文件名称

AlawFileName　　A-law 语音文件名称

返　　回：　如果调用成功，返回 BF_OK

注　　释：

4.1.2.20　停止通道所有操作接口函数

▲　BF_StopChannelOperate

函数原型：　unsigned short BF_StopChannelOperate(unsigned short ChannelNo)

功　　能：　停止通道的一切操作，仅指语音通道，不包括资源通道，使通道恢复打开时的状态。

参　　数：　ChannelNo　通道的逻辑号码

返　　回：　如果调用成功，返回 BF_OK

注　　释：

4.1.2.21　会议接口函数

▲　BF_ConferenceAddMember

函数原型：　unsigned short BF_ConferenceAddMember(unsigned short ChannelNo, unsigned short Mode, unsigned short Group)

功　　能：　将指定通道添加到某组会议中。

参　　数：　ChannelNo　通道的逻辑号码

Mode　　指示添加模式，取值如下：

1——只听模式

2——讲话模式

3——录音模式

Group　　会议的组别，最大为 MAX_CONFERENCE_GROUP_NUM

返　　回：　调用成功，返回 BF_OK

注　　释：　每增加一个讲话/录音模式，占用一个会议资源，所有只听模式共用一个资源

▲　BF_ConferenceDelMember

函数原型：　unsigned short BF_ConferenceDelMember(unsigned short ChannelNo, unsigned short Mode, unsigned short Group)

功　　能：　将指定通道从某组会议中删除。

参　　数：　ChannelNo　通道的逻辑号码

Mode　　指示添加模式，取值如下：

1——只听模式

2——讲话模式

3——录音模式

Group　　会议的组别，最大为 MAX_CONFERENCE_GROUP_NUM

返　　回：　调用成功，返回 BF_OK

注　　释：

4.1.2.22　TTS(Text To Speech)接口函数

BF-VOICE 系统含有一套免费的简单的 TTS 系统，能将文本转换成语音。如果用户的要求严格，BF-VOICE 语音卡可以提供第三方的语音合成系统，以满足客户的需求。

▲　BF_InitTTS

函数原型： unsigned short BF_InitTTS(void)
功　　能： 初始化 TTS 系统，该函数仅仅调用一次，只有返回成功时，TTS 系统才能正常工作。
参　　数： 无
返　　回： 如果调用成功，返回 BF_OK
注　　释：

▲ BF_CloseTTS

函数原型： unsigned short BF_CloseTTS(void)
功　　能： 关闭 TTS 系统，在不使用 TTS 系统时，调用该函数，以释放系统资源。
参　　数： 无
返　　回： 如果调用成功，返回 BF_OK
注　　释：

▲ BF_TTSFileToFile

函数原型： unsigned short BF_TTSFileToFile(char *TextFileName, char *VoiceFileName)
功　　能： 将文本文件转换成语音文件。
参　　数： TextFileName　　文本文件名称
VoiceFileName　　语音文件名称
返　　回： 如果调用成功，返回 BF_OK
注　　释：

▲ BF_TTSBufferToBuffer

函数原型： unsigned short BF_TTSBufferToBuffer(char *TextBuffer,unsigned long TextCount, char *VoiceBuffer,unsigned long *VoiceCount)
功　　能： 将文本缓存区转换成语音缓存区。
参　　数： TextBuffer　　文本缓存区指针
TextCount　　文本的字节数量
VoiceBuffer　　语音缓存区指针
VoiceCount　　语音缓存区数据数量，单位字节
返　　回： 如果调用成功，返回 BF_OK
注　　释：

▲ BF_TTSBufferToFile

函数原型： unsigned short BF_TTSBufferToFile(char *TextBuffer,unsigned long TextCount,char *VoiceFileName)
功　　能： 将文本缓存区转换成语音文件。
参　　数： TextBuffer　　文本缓存区指针
TextCount　　文本的字节数量
VoiceFileName　　语音文件名称
返　　回： 如果调用成功，返回 BF_OK
注　　释：

第二章　BF-VOICE 语音卡资源通道编程接口函数

4.2.1　资源通道操作接口函数按功能索引表

获取系统参数接口函数

```
unsigned short WINAPI BF_GetTotalResource(void);
unsigned short WINAPI BF_GetResourceType(unsigned short ResourceNo);
```

打开/关闭资源接口函数

```
unsigned short WINAPI BF_OpenResource(unsigned short ResourceNo,unsigned short
        ResourceType);
unsigned short WINAPI BF_CloseResource(unsigned short FaxChannelNo);
```

绑定资源接口函数

```
unsigned short WINAPI BF_BindResource(unsigned short ChannelNo, unsigned short
        ResourceNo);
unsigned short WINAPI BF_FreeResource(unsigned short ChannelNo, unsigned short
        ResourceNo);
```

传真资源操作接口函数

```
unsigned short WINAPI BF_InitFaxFunction(void);
unsigned short WINAPI BF_CloseFaxFunction(void);
unsigned short WINAPI BF_SetFaxLocalTelNo(unsigned short ChannelNo,unsigned char
        *LocalTelNo);
unsigned short WINAPI BF_GetFaxRemoteTelNo(unsigned short ChannelNo,unsigned char
        *RemoteTelNo);
unsigned short WINAPI BF_GetFaxInformation(unsigned short ChannelNo,
        PBF_FAX_INFORMATION FaxInformation);
unsigned short WINAPI BF_GetFaxPageNo(unsigned short ChannelNo);
unsigned short WINAPI BF_SendFaxTxtBuffer(unsigned short ChannelNo, char
        *TxtBuffer, unsigned short BufferCount);
unsigned short WINAPI BF_SendFaxTxtFile(unsigned short ChannelNo,char *FileName);
unsigned short WINAPI BF_SendBitFaxFile(unsigned short ChannelNo,char *FileName);
unsigned short WINAPI BF_SendTiffFaxFile(unsigned short ChannelNo, char
        *FileName);
unsigned short WINAPI BF_ReceiveBitFaxFile(unsigned short ChannelNo, char
        *FileName);
unsigned short WINAPI BF_ReceiveTiffFaxFile(unsigned short ChannelNo, char
        *FileName);
unsigned short WINAPI BF_StopFaxOperate(unsigned short ChannelNo);
```

VoIP 资源操作接口函数

```
unsigned short WINAPI BF_StartVoipOperate(unsigned short ChannelNo);
unsigned short WINAPI BF_StopVoipOperate(unsigned short ChannelNo);
```

4.2.2　资源通道操作接口函数的详细说明

BF-VOICE 语音卡包含资源通道，以提供传真、VoIP 和语音压缩等功能。

BF-VOICE 语音卡所有的资源通道在打开操作之后，必须与相应的语音通道绑定，然后在语

音通道上，操作资源的功能。下面具体介绍资源的操作接口函数。

4.2.2.1 获取系统参数接口函数

▲ BF_GetTotalresource

函数原型： unsigned short WINAPI BF_GetTotalResource(void);
功　　能： 获取系统的资源总的数量
参　　数： 无
返　　回： 如果调用成功，返回系统资源总的数量
注　　释： 无

▲ BF_GetResourceType

函数原型： unsigned short WINAPI BF_GetResourceType(unsigned short ResourceNo);
功　　能： 获取资源的类型
参　　数： ResourceNo　　资源的逻辑通道号码
返　　回： 如果调用成功，返回资源的类型
注　　释：

4.2.2.2 打开/关闭资源接口函数

▲ BF_Openresource

函数原型： unsigned short WINAPI BF_OpenResource(unsigned short ResourceNo,
unsigned short ResourceType);
功　　能： 打开资源通道。在使用资源以前，必须按需求打开资源通道
参　　数： ResourceNo　　资源的逻辑通道号码
ResourceType　　打开的类型，某个资源通道可能适于多种类型，如即适于传真，又适于 VoIP，但只能按一种类型打开
该参数的取值如下：

RESOURCE_TYPE_FAX	0x0001	传真资源
RESOURCE_TYPE_VOIP	0x0002	VoIP 资源
RESOURCE_TYPE_CODEC	0x0004	编码/解码资源

返　　回： 如果调用成功，返回 BF_OK
注　　释：

▲ BF_CloseResource

函数原型： unsigned short WINAPI BF_CloseResource(unsigned short FaxChannelNo);
功　　能： 关闭打开的资源
参　　数： ResourceNo　　资源的逻辑通道号码
返　　回： 如果调用成功，返回 BF_OK
注　　释：

4.2.2.3 绑定资源接口函数

▲ BF_BindResource

函数原型： unsigned short WINAPI BF_BindResource(unsigned short ChannelNo,
unsigned shortResourceNo);
功　　能： 绑定资源。任何打开的资源，必须与相应的语音通道绑定之后才能体现其功

能，在绑定之后，可以直接在语音通道上，操作被绑定的资源。

参　　数：　ChannelNo　　语音通道的逻辑通道号码
　　ResourceNo　　资源通道的逻辑通道号码

返　　回：　如果调用成功，返回 BF_OK

注　　释：

▲ BF_FreeResource

函数原型：　unsigned short WINAPI BF_FreeResource(unsigned short ChannelNo,
　　unsigned short ResourceNo);

功　　能：　释放被绑定的资源。当语音通道不需要资源操作时，必须释放被绑定的资源，BF_StopChannelOperate 函数并不能释放被绑定的资源。

参　　数：　ChannelNo　　语音通道的逻辑通道号码
　　ResourceNo　　资源通道的逻辑通道号码

返　　回：　如果调用成功，返回 BF_OK

注　　释：　BF_StopChannelOperate 函数并不能释放被绑定的资源。

4.2.2.4 传真资源操作接口函数

BF-VOICE 系列语音卡的某些板卡带有传真资源，这些传真资源可以被插入同一台机器上的指定 BF-VOICE 系列语音卡的语音通道使用，使用方法如下：

1、 调用函数 BF_InitFaxFunction，初始化传真功能
2、 调用函数 BF_OpenResource 以 RESOURCE_TYPE_FAX 打开资源通道
3、 调用函数 BF_BindResource 绑定资源通道与语音通道
4、 调用函数 BF_SetLocalTelNo 设置相应语音通道的本地传真号码
5、 调用相关函数 BF_SendFaxTxtBuffer、BF_SendFaxTxtFile、BF_SendBitFaxFile、BF_SendTiffFaxFile、BF_ReceiveBitFaxFile 和 BF_ReceiveTiffFaxFile 进行发送或接收传真
6、 如果收到事件 MESSAGE_FAX_HAVE_REMOTE_TEL_NO，可以调用函数 BF_GetFaxRemoteTelNo 取得对方传真号码
7、 直到收到相应事件，如 MESSAGE_FAX_SEND_OK、MESSAGE_FAX_SEND_ERROR、MESSAGE_FAX_RECEIVE_OK、MESSAGE_FAX_RECEIVE_ERROR 再进行下一步操作
8、 调用函数 BF_FreeResource 释放资源
9、 如果还有传真操作，从步骤 3 开始执行
10、当系统退出时，调用函数 BF_CloseFaxFunction 释放系统资源

▲ BF_InitFaxFunction

函数原型：　unsigned short BF_InitFaxFunction(void)

功　　能：　初始化传真系统，该函数仅仅调用一次，只有返回成功时，传真系统才能正常工作。

参　　数：　无

返　　回：　如果调用成功，返回 BF_OK

注　　释：

▲ BF_CloseFaxFunction

函数原型：　unsigned short BF_CloseFaxFunction(void)

功　　能：　关闭传真系统。
参　　数：　无
返　　回：　如果调用成功，返回 BF_OK
注　　释：

▲ BF_SetFaxLocalTelNo

函数原型：　unsigned short BF_SetFaxLocalTelNo(unsigned short ChannelNo,
　　unsigned char *LocalTelNo)
功　　能：　设置指定通道电话号码，该号码将发送给对方传真机。
参　　数：　ChannelNo　　语音通道号码，从 0 开始
　　LocalTelNo　　电话号码
返　　回：　如果调用成功，返回 BF_OK
注　　释：

▲ BF_GetFaxRemoteTelNo

函数原型：　unsigned short BF_GetFaxRemoteTelNo(unsigned short ChannelNo,
　　unsigned char *RemoteTelNo)
功　　能：　获取对方传真机电话号码。
参　　数：　ChannelNo　　语音通道号码，从 0 开始
　　RemoteTelNo　　对方传真电话号码
返　　回：　如果调用成功，返回 BF_OK
注　　释：　该函数只有在收到 MESSAGE_FAX_HAVE_REMOTE_TEL_NO 事件后调用，才有意义

▲ BF_GetFaxInformation

函数原型：　unsigned short BF_GetFaxInformation(unsigned short ChannelNo,
　　PBF_FAX_INFORMATION FaxInformation)
功　　能：　取得传真操作信息。
参　　数：　ChannelNo　　语音通道号码，从 0 开始
　　FaxInformation　传真信息参数，是一个指向 PBF_FAX_INFORMATION 结构的指针，PBF_FAX_INFORMATION 的结构定义如下：

```
typedef struct _BF_FAX_INFORMATION{
    unsigned short      SignalRate;
    unsigned short      Resolution;
    unsigned short      TwoDimensional;
    unsigned short      RecordWidth;
    unsigned short      RecordLength;
}BF_FAX_INFORMATION, *PBF_FAX_INFORMATION;
```

_BF_FAX_INFORMATION 结构中的各项含义如下：

代号	含义
SignalRate	传真速率
Resolution	传真分辨率 0——3.58/mm 1——7.71/mm
TwoDimensional	是否二维编码

		0——一维编码
		1——二维编码
	RecordWidth	每行的点阵数量
	RecordLength	每页的长度，单位毫米，取值如下：
		0xFFFF ——长度不限
		其他 ——单位为毫米的长度

返　　回：　如果调用成功，返回 BF_OK

注　　释：　该函数只有在得到事件 MESSAGE_FAX_HAVE_INFORMATION 后调用才有意义

▲ BF_GetFaxPageNo

函数原型：　unsigned short BF_GetFaxPageNo(unsigned short ChannelNo);

功　　能：　获取当前正在进行的发送或接收传真的页数。

参　　数：　ChannelNo　　语音通道号码，从 0 开始

返　　回：　函数返回值为传真页数

注　　释：

▲ BF_SendFaxTxtBuffer

函数原型：　unsigned short BF_SendFaxTxtBuffer(unsigned short ChannelNo,
char *TxtBuffer, unsigned short BufferCount)

功　　能：　发送格式为文本格式的缓存区传真。

参　　数：　ChannelNo　　语音通道号码，从 0 开始
TxtBuffer　　文本格式的缓存区指针
BufferCount　　文本字节数量

返　　回：　如果调用成功，返回 BF_OK

注　　释：　在传真通道上返回事件 MESSAGE_SEND_FAX_OK 或 MESSAGE_SEND_FAX_ERROR
文本每行最大字节长度为 106 字节，并以 0x0A 为行结束。

▲ BF_SendFaxTxtFile

函数原型：　unsigned short BF_SendFaxTxtFile(unsigned short ChannelNo,
char *FileName)

功　　能：　发送格式为文本文件的传真。

参　　数：　ChannelNo　　语音通道号码，从 0 开始
FileName　　文本文件名称

返　　回：　如果调用成功，返回 BF_OK

注　　释：　在传真通道上返回事件 MESSAGE_SEND_FAX_OK 或 MESSAGE_SEND_FAX_ERROR
文本每行最大字节长度为 106 字节，并以 0x0A 为行结束。

▲ BF_SendBitFaxFile

函数原型：　unsigned short BF_SendBitFaxFile(unsigned short ChannelNo,
char *FileName)

功　　能：　发送格式为 BitFax 文件的传真。

参　　数：　ChannelNo　　语音通道号码，从 0 开始
FileName　　BitFax 文件名称

返　　回：　如果调用成功，返回 BF_OK

注　　释：　在传真通道上返回事件 MESSAGE_SEND_FAX_OK 或 MESSAGE_SEND_FAX_ERROR

▲ BF_SendTiffFaxFile

函数原型：　unsigned short BF_SendTiffFaxFile(unsigned short ChannelNo, char *FileName)

功　　能：　发送格式为 BitFax 文件的传真。

参　　数：　ChannelNo　　语音通道号码，从 0 开始

FileName　　Tiff Fax 格式文件名称

返　　回：　如果调用成功，返回 BF_OK

注　　释：　在传真通道上返回事件 MESSAGE_SEND_FAX_OK 或 MESSAGE_SEND_FAX_ERROR

▲ BF_ReceiveBitFaxFile

函数原型：　unsigned short BF_ReceiveBitFaxFile(unsigned short ChannelNo, char *FileName)

功　　能：　接收传真，传真格式为 BitFax 格式。

参　　数：　ChannelNo　　语音通道号码，从 0 开始

FileName　　存放 BitFax 格式传真数据的文件名称

返　　回：　如果调用成功，返回 BF_OK

注　　释：　在传真通道上返回事件 MESSAGE_RECEIVE_FAX_OK 或 MESSAGE_RECEIVE_FAX_ERROR

▲ BF_ReceiveTiffFaxFile

函数原型：　unsigned short BF_ReceiveTiffFaxFile(unsigned short ChannelNo, char *FileName)

功　　能：　接收传真，传真格式为 BitFax 格式。

参　　数：　ChannelNo　　语音通道号码，从 0 开始

FileName　　存放 TiffFax 格式传真数据的文件名称

返　　回：　如果调用成功，返回 BF_OK

注　　释：　在传真通道上返回事件 MESSAGE_RECEIVE_FAX_OK 或 MESSAGE_RECEIVE_FAX_ERROR

▲ BF_StopFaxOperate

函数原型：　unsigned short BF_StopFaxOperate(unsigned short FaxChannelNo)

功　　能：　停止传真操作。

参　　数：　FaxChannelNo　　传真通道号码，从 0 开始

返　　回：　如果调用成功，返回 BF_OK；但必须等待传真通道返回事件，才能表示传真结束。

注　　释：　该函数仅仅停止相应传真通道的传真操作，并不释放与之关联的语音通道，如果需要释放连接的语音通道，必须调用函数 BF_FaxUnLinkChannel

4.2.2.5　VoIP 资源操作接口函数

BF-VOICE 语音卡的 VoIP 功能主要配合 BF-VOICE 的 H.323 协议栈使用，以实现 VoIP 的功能。

在这里，仅仅介绍相应函数的一般用法，详细说明请参阅 BF-H.323 协议栈操作手册。

▲ BF_StartVoipOperate

函数原型： unsigned short WINAPI BF_StartVoipOperate(unsigned short ChannelNo);
功　　能： 启动 VoIP 操作。该函数启动了与 ChannelNo 语音通道绑定的资源通道的 VoIP 操作。
参　　数： ChannelNo　　语音通道号码，从 0 开始
返　　回： 如果调用成功，返回 BF_OK;
注　　释：

▲ BF_StopVoipOperate

函数原型： unsigned short WINAPI BF_StopVoipOperate(unsigned short ChannelNo);
功　　能： 停止 VoIP 操作。该函数停止了与 ChannelNo 语音通道绑定的资源通道的 VoIP 操作。
参　　数： ChannelNo　　语音通道号码，从 0 开始
返　　回： 如果调用成功，返回 BF_OK;
注　　释：

第三章　BF-VOICE 语音卡编程接口数据结构及常量定义

以下介绍的常量或数据结构均包含在 BFVoice.H 文件中。

4.3.1　基本常量定义

```
#define     MAX_CARD_NUM                    0x0008
```
定义了插入同一台计算机中 BF-VOICE 语音卡的最大数量
```
#define     MAX_CHANNEL_NUM                 0x0040
```
定义了插入同一台计算机中 BF-VOICE 语音卡的所有语音通道的最大数量
```
#define     MAX_RESOURCE_NUM                0x0040
```
定义了插入同一台计算机中 BF-VOICE 语音卡的所有资源通道的最大数量
```
#define     MAX_SERIAL_NUM                  0x0020
```
定义了 BF-VOICE 语音卡的系列号的最大字节数量
```
#define     MAX_DTMF_NUM                    0x0040
```
定义了所有涉及 DTMF 按键个数的字符串中的最大字符数量，常常只允许 MAX_DTMF_NUM-1 个
```
#define     MAX_CID_NUM                     0x0080
```
定义了接收 FSK 主叫的字符串最大字符数量
```
#define     MAX_FSK_NUM                     0x0400
```
定义了发送 FSK 的字符串中的最大字符数量
```
#define     MAX_FILE_NAME_NUM               0x0080
```
定义了所有涉及文件名称的字符串中的最大字符数量，常常只允许 MAX_FILE_NAME_NUM -1 个
```
#define     MAX_INDEX_BUFFER_NUM            0x0100
```
定义了最大索引缓存区放音的数量
```
#define     MAX_INDEX_FILE_NUM              0x0040
```
定义了最大索引文件放音的数量

```
#define     MAX_CONFERENCE_GROUP_NUM            0x0010
```
定义了最大会议组的数量
```
#define     MAX_CONFERENCE_RESOURCE_NUM         0x0020
```
定义了最大会议资源的数量
```
#define     MAX_CONFERENCE_GROUP_RESOURCE_NUM   0x0010
```
定义了每个通道最大会议资源的数量
```
#define     BF_OK                               0x0000
```
当调用 BF-VOICE 函数时，如果返回该值，表示调用成功，否则意味出错
```
#define     BF_ERROR                            0xFFFF
```
当调用 BF-VOICE 函数时，如果返回该值，意味出错
```
#define     CHANNEL_TYPE_NONE                   0x0000
```
该值为语音通道的模块类型，表示相应的语音通道没有模块
```
#define     CHANNEL_TYPE_USER                   0x0001
```
该值为语音通道的模块类型，表示相应的语音通道的模块为用户模块
```
#define     CHANNEL_TYPE_TRUNK                  0x0002
```
该值为语音通道的模块类型，表示相应的语音通道的模块为中继模块
```
#define     CHANNEL_TYPE_RECORD                 0x0003
```
该值为语音通道的模块类型，表示相应的语音通道的模块为录音（高阻）模块
```
#define     RESOURCE_TYPE_NONE                  0x0000
```
该值为资源通道的类型，表示相应的资源通道的不存在
```
#define     RESOURCE_TYPE_FAX                   0x0001
```
该值为资源通道的类型，表示相应的资源通道的类型为传真
```
#define     RESOURCE_TYPE_VOIP                  0x0002
```
该值为资源通道的类型，表示相应的资源通道的类型为 VoIP
```
#define     RESOURCE_TYPE_CODEC                 0x0004
```
该值为资源通道的类型，表示相应的资源通道的类型为编码/解码
```
#define     RESOURCE_TYPE_FAX_VOIP              0x0003
```
该值为资源通道的类型，表示相应的资源通道的类型为传真和 VoIP
```
#define     RESOURCE_TYPE_FAX_CODEC             0x0005
```
该值为资源通道的类型，表示相应的资源通道的类型为传真和编码/解码
```
#define     RESOURCE_TYPE_VOIP_CODEC        0x0006
```
该值为资源通道的类型，表示相应的资源通道的类型为 VoIP 和编码/解码
```
#define     RESOURCE_TYPE_FAX_VOIP_CODEC    0x0007
```
该值为资源通道的类型，表示相应的资源通道的类型为传真、VoIP 和编码/解码

4.3.2 数据结构定义

4.3.2.1 事件数据结构

```
typedef struct _BF_MESSAGE_INFO{
    USHORT      MessageCode;
    USHORT      ChannelNo;
    USHORT      Parameter[6];
} BF_MESSAGE_INFO, *PBF_MESSAGE_INFO;
```
定义了事件数据结构，即每次取回事件的内容，具体意义如下：

MessageCode 占 1 个字，事件号码。

ChannelNo　占 1 个字，发生事件的通道。

Parameter　占 6 个字，事件参数，对于不同的事件，其内容不同，具体参见相关的事件类型说明章节。

4.3.2.2 标准信号音参数数据结构

```
typedef struct _BF_TONE_INFO{
    unsigned short      DialToneMinTotalTime;
    unsigned short      BusyToneMaxTotalTime;
    unsigned short      BusyToneMinTotalTime;
    unsigned short      BusyToneRatio;
    unsigned short      BusyToneErrorBetweenTwo;
    unsigned short      BusyToneTotalTimes;
    unsigned short      BackToneMaxTotalTime;
    unsigned short      BackToneMinTotalTime;
    unsigned short      BackToneRatio;
    unsigned short      BackToneErrorBetweenTwo;
    unsigned short      BackToneTotalTimes;
}BF_TONE_INFO, *PBF_TONE_INFO;
```

_BF_TONE_INFO 结构中的各项含义如下：

代号	缺省值	含义
DialToneMinTotalTime	拨号音持续最小时间(毫秒)	2500
BusyToneMaxTotalTime	最大忙音周期时间(毫秒)	1500
BusyToneMinTotalTime	最小忙音周期时间(毫秒)	500
BusyToneRatio	忙音周期占空比	50
BusyToneErrorBetweenTwo	两个忙音周期的误差比	10
BusyToneTotalTimes	忙音周期数量	3
BackToneMaxTotalTime	最大回铃音周期时间(毫秒)	6000
BackToneMinTotalTime	最小回铃音周期时间(毫秒)	4000
BackToneRatio	回铃音周期占空比	20
BackToneErrorBetweenTwo	两个回铃音周期的误差比	10
BackToneTotalTimes	回铃音周期数量	3

4.3.2.3 传真信息参数数据结构

```
typedef struct _BF_FAX_INFORMATION{
    unsigned short      SignalRate;
    unsigned short      Resolution;
    unsigned short      TwoDimensional;
    unsigned short      RecordWidth;
    unsigned short      RecordLength;
}BF_FAX_INFORMATION, *PBF_FAX_INFORMATION;
```

_BF_FAX_INFORMATION 结构中的各项含义如下：

代号	含义
SignalRate	传真速率，即波特率
Resolution	传真分辨率

	0——3.58 行/毫米
	1——7.71 行/毫米
TwoDimensional	是否二维编码
	0——一维编码
	1——二维编码
RecordWidth	每行的点阵数量
RecordLength	每页的长度，单位毫米，取值如下：
	0xFFFF ——长度不限
	其他 ——单位为毫米的长度

第四章 BF-VOICE 语音卡编程接口事件类型说明

本章介绍的结构 BF_MESSAGE_INFO 中 MessageCode 的种类，以及与 MessageCode 相对应的参数 Parameter 的意义。

#define MESSAGE_HAVE_CID_FSK 0x0000

有 FSK 方式的主叫事件

#define MESSAGE_HAVE_CID_DTMF 0x0001

有 DTMF 方式的主叫事件

#define MESSAGE_HAVE_DTMF 0x0002

当识别到 DTMF 按键时，BF-VOICE 语音卡发送该事件，该事件的参数 Parameter 没有意义。如果要想得到该事件，必须先调用函数 BF_StartDetectDTMF。

#define MESSAGE_PLAY_VOICE_END 0x0003

放音结束事件，当全部放音数据放完时，BF-VOICE 语音卡发送该事件，并停止放音，该事件的参数 Parameter 没有意义。

#define MESSAGE_PLAY_VOICE_EXIT 0x0004

放音中断事件，当打开或读取放音文件出错，或收到放音停止的 DTMF 时，BF-VOICE 语音卡发送该事件，并停止放音，该事件的参数 Parameter 没有意义。

#define MESSAGE_RECORD_VOICE_END 0x0005

录音结束事件，当所需数量的录音数据录完时，BF-VOICE 语音卡发送该事件，并停止录音，该事件的参数 Parameter 没有意义。

#define MESSAGE_RECORD_VOICE_EXIT 0x0006

录音中断事件，当创建或写录音文件出错，或收到录音停止的 DTMF 时，BF-VOICE 语音卡发送该事件，并停止录音，该事件的参数 Parameter 没有意义。

#define MESSAGE_GENERATE_SIGNAL_END 0x0007

发送信号音结束事件，当信号音发送完成时，BF-VOICE 语音卡发送该事件，并停止发送信号音，该事件的参数 Parameter 没有意义。

#define MESSAGE_GENERATE_SIGNAL_EXIT 0x0008

发送信号音中断事件，当收到中断发送信号音的 DTMF 时，BF-VOICE 语音卡发送该事件，并停止发送信号音，该事件的参数 Parameter 没有意义。

#define MESSAGE_GENERATE_DTMF_END 0x0009

当发送完所有需要发送的 DTMF 按键时，BF-VOICE 语音卡发送该事件，该事件的参数。Parameter 没有意义。没有命令可以中断发送 DTMF 命令。

#define MESSAGE_SEND_FSK_END 0x000A

当发送 FSK 完毕，BF-VOICE 语音卡发送该事件。

#define MESSAGE_HAVE_DIAL_TONE 0x000B

当模块有标准拨号音时，BF-VOICE 语音卡发送该事件，该事件的参数 Parameter 没有意义。如果要想得到该事件，必须先调用函数 BF_StartDetectDialTone。

#define MESSAGE_HAVE_BUSY_TONE 0x000C

当模块有标准忙音时，BF-VOICE 语音卡发送该事件，该事件的参数 Parameter 没有意义。如果要想得到该事件，必须先调用函数 BF_StartDetectBusyTone。

#define MESSAGE_HAVE_BACK_TONE 0x000D

当模块有标准回铃音时，BF-VOICE 语音卡发送该事件，该事件的参数 Parameter 没有意义。如果要想得到该事件，必须先调用函数 BF_StartDetectBackTone。

#define MESSAGE_HAVE_TONE_DATA 0x000E

当模块有信号音数据时，BF-VOICE 语音卡发送该事件，该事件的参数 Parameter 的第一个字为信号音数据。

#define MESSAGE_HAVE_HELLO_VOICE 0x000F

当识别到话音时，BF-VOICE 语音卡发送该事件，该事件的参数 Parameter 没有意义。如果要想得到该事件，必须先调用函数 BF_StartDetectHelloVoice。

#define MESSAGE_HAVE_SILENCE_DATA 0x0010

当启动静音识别时，每当 BF-VOICE 语音卡完成给定时间内的识别时发送该事件，该事件的参数 Parameter 的第一个字表示静音数据。如果要想得到该事件，必须先调用函数 BF_StartDetectSilence。

#define MESSAGE_HAVE_BACK_TONE_HALT 0x0011

表示回铃音已经停止

#define MESSAGE_HAVE_SPECIAL_SIGNAL 0x0012

在识别特殊信号音时，表示有符合持续条件的信号音

#define MESSAGE_HAVE_FSK_DATA 0x0013

当识别到 FSK 信息时，发送该事件

#define MESSAGE_HAVE_TIME_OUT 0x0014

当设定的超时到达时，BF-VOICE 语音卡发送该事件，该事件的参数 Parameter 没有意义。如果要想得到该事件，必须先调用函数 BF_StartTimeOut。

#define MESSAGE_PLAY_CIRCLE_FIRST_END 0x0015

在循环缓存区放音时，表示上一半缓存区已经放音完毕

#define MESSAGE_PLAY_CIRCLE_SECOND_END 0x0016

在循环缓存区放音时，表示下一半缓存区已经放音完毕

#define MESSAGE_RECORD_CIRCLE_FIRST_END 0x0017

在循环缓存区录音时，表示上一半缓存区已经录音完毕

#define MESSAGE_RECORD_CIRCLE_SECOND_END 0x0018

在循环缓存区录音时，表示下一半缓存区已经录音完毕

#define MESSAGE_AUTO_DIAL_NO_DIAL_TONE 0x0019

在自动拨号时，表示没有拨号音

#define MESSAGE_AUTO_DIAL_NONE_LISTEN 0x001A

在自动拨号时，表示对方没人接听

#define MESSAGE_AUTO_DIAL_HAVE_BUSY 0x001B

在自动拨号时，表示对方占线

#define MESSAGE_AUTO_DIAL_HOOK_OFF 0x001C

在自动拨号时，表示对方摘机

#define MESSAGE_USER_HOOK_OFF 0x1001

当用户模块有摘机动作时，BF-VOICE 语音卡发送该事件，该事件的参数 Parameter 没有意义。

#define MESSAGE_USER_HOOK_ON 0x1002

当用户模块有挂机动作时，BF-VOICE 语音卡发送该事件，该事件的参数 Parameter 没有意义。

#define MESSAGE_USER_PICK_UP 0x1003

当用户模块有拍插动作时，BF-VOICE 语音卡发送该事件，该事件的参数 Parameter 没有意义，所谓的拍插动作，就是当用户模块由摘机状态转换到挂机状态，然后再回到摘机状态的总共时间小于一定时间的动作，该时间缺省值是 500 毫秒，该值可由函数 BF_SetUserPickUpTime()重新设置。

#define MESSAGE_USER_RING_END 0x1004

当用户模块的振铃完成时，BF-VOICE 语音卡发送该事件，该事件的参数 Parameter 没有意义。

#define MESSAGE_TRUNK_RING 0x2000

当中继模块有振铃时，BF-VOICE 语音卡发送该事件，该事件的参数 Parameter 没有意义。

#define MESSAGE_TRUNK_POLE 0x2001

当中继模块有极性反转时,，BF-VOICE 语音卡发送该事件，该事件参数 Parameter 没有意义。

#define MESSAGE_TRUNK_PICK_UP_END 0x2002

当中继模块完成拍插动作后，BF-VOICE 语音卡发送该事件，该事件的参数 Parameter 没有意义。

#define MESSAGE_RECORD_OPEN_CIRCLE 0x3001

当录音模块没有接电话线时，BF-VOICE 语音卡发送该事件。

#define MESSAGE_RECORD_NO_OPEN_CIRCLE 0x3002

当录音模块从没有接电话线状态变为有接电话线状态时时，BF-VOICE 语音卡发送该事件。

#define MESSAGE_RECORD_HOOK_OFF 0x3004

当录音模块有摘机动作时，BF-VOICE 语音卡发送该事件，该事件参数 Parameter 没有意义。

#define MESSAGE_RECORD_HOOK_ON 0x3005

当录音模块有挂机动作时，BF-VOICE 语音卡发送该事件，该事件参数 Parameter 没有意义。

#define MESSAGE_RECORD_RING 0x3006

当录音模块有振铃时，BF-VOICE 语音卡发送该事件，该事件参数 Parameter 没有意义。

#define MESSAGE_SEND_FAX_OK 0x5000

当发送传真正常完成时,，BF-VOICE 语音卡发送该事件。

#define MESSAGE_SEND_FAX_ERROR 0x5001

当发送传真出错时,，BF-VOICE 语音卡发送该事件。

#define MESSAGE_RECEIVE_FAX_OK 0x5002

当接收正常完成时,，BF-VOICE 语音卡发送该事件。

#define MESSAGE_RECEIVE_FAX_ERROR 0x5003

当接收传真出错时,，BF-VOICE 语音卡发送该事件。

#define MESSAGE_FAX_HAVE_REMOTE_TEL_NO 0x5004

在接收或发送传真过程中，收到对方传真号码时，发送该消息，应用程序可以调用函数 BF_GetRemoteTelNo 来取得对方传真号码

#define MESSAGE_FAX_HAVE_INFORMATION 0x5005

在接收或发送传真过程中，确认双方传真信息时，发送该消息，应用程序可以调用函数 BF_GetFaxInformation 来取得传真信息

第五章 BF-VOICE 语音卡各种语言编程范例说明

BF-VOICE 系列语音卡为 Windows 系统平台上的用户提供了各种编程语言的演示程序，这些程序都以源代码方式分发，用户在编程时，可以参考。

BF-VOICE 系列语音卡为 Windows 系统平台上的用户提供的编程接口为动态连接库 BFVoice.dll，该文件存放在相应的 Windows 目录中。在实际编程中，用户可以根据实际情况进行操作。

4.5.1 VC 平台编程范例

4.5.1.1 C 接口简介

BFVoice.LIB 是 BFVoice.dll 的 C 语言输入库，由 Microsoft Visual C 6.0 生成。在 VC 环境下开发，可以直接将 BFVoice.lib 加入到工程文件中。

4.5.1.2 C 语言编程简介

在 Windows 98/2000/XP 下用 C 语言对 BFVoice 系列语音卡进行编程时，一般包括三部分。

一、初始化部分，包括初始化卡、得到通道数和通道类型等等，这部分在运行时调用一次即可。

二、结束部分，包括关闭语音卡，这部分在退出时调用。

三、处理部分，这部分对实时的情况进行处理，是一个循环过程。

4.5.1.3 C 语言示范程序

C 语言示范程序在 VC 6.0 的环境下开发。

安装在安装目录下的 Demo\Vc6 子目录下。

4.5.2 VB 平台编程范例

4.5.2.1 VB 接口简介

在 Visual Basic 中， 对动态库 BFVoice.DLL 的调用是通过 BFVoice.BAS 来声明的，请参见 BFVoice.bas。

在本质上，VB 也是通过调用 BFVoice.dll 来实现对 BFVoice 系列语音卡硬件的控制。

4.5.2.2 VB 对动态库的调用

在 VB 中，需要首先对 DLL 中的函数进行声明，然后才能对该函数进行调用。关于如何声明 DLL 中的函数，请参考 VB 中的 HELP 文件“Declare statement”。

4.5.2.3 VB 编程简介

同在 98/NT/2000 下用 C 语言编程一样，VB 对录音卡的操作，也包括三部分：

一、初始化部分，包括初始化卡、得到通道数和通道类型等等，在 Form_Load 中运行。

二、结束部分，包括关闭语音卡，在 Form_Unload 中运行。

三、处理部分，这部分在 TIMER 中运行，或放在 Form_Activate 函数里，具体用法请参见示范程序。

4.5.2.4 VB 编程的注意事项

A> 所有参数和返回值的字节长度要相等。

B> 在 VC 中（32 位环境），BOOL 为 32 位，WORD 为 16 位，DWORD 为 32 位；
在 VB 中，相应的使用 Long（32 位）, Integer（16 位）, Long（32 位）

C> 在 VB 6.0 中，变量类型 Integer 仍然是 16 位的。其范围是-32767～32767。VC 中的 int, long, DWORD, BOOL 在 VB5.0 中需用 32 位的 long 表示；而 VC 中的 WORD 在 VB5.0 中需用 Integer 表示。

D> 如果在 C 语言中某函数的参数为字符串，并且在函数调用完毕后，有内容通过该参数返回，当 VB 使用该函数时，带入的变量必须是已经分配空间的。

E> 如果在 Form_Activate 中处理 BF-VOICE 语音卡的事件，Form_Activate 应该是一个死循环，要调用 DoEvents，以使得其他 98/NT/2000 程序能正常工作。

4.5.2.5 VB 演示程序

VB 语言示范程序在 VB 6.0 的环境下开发。

安装在安装目录下的 Demo\Vb6 子目录下。

4.5.3 C++ BUILDER 平台编程范例

4.5.3.1 C++ Builder 接口简介

在 C++ Builder 中，对动态库 BFVoice.dll 的调用的声明可以直接使用 VC 语言的 BFVoice.h，但是由于 BFVoice.lib 是在 VC 环境下生成的，该 LIB 无法直接用于 C++ BUILDER 工程文件的连接，C++ BUILDER 提供了一个从 DLL 中提取输入库 LIB 的工具软件 IMPLIB.EXE，你可用该软件生成一个 LIB，用于开发。

4.5.3.2 C++ Builder 对动态库的调用

Implib 使用方法如下：

```
implib  BFVoice.lib  BFVoice.dll
```

对于 IMPLIB.EXE 的详细使用方法和功能，请参考其自身的帮助。

将生成的 BFVoice.lib 拷贝到开发程序需要的地方，并加入到应用程序的工程文件中即可。

4.5.3.3 C++ Builder 下的示范程序

C++ Builder 语言示范程序在 C++ Builder 5.5 的环境下开发。

安装在安装目录下的 Demo\BC5 子目录下。

4.5.4 Delphi 平台编程范例

4.5.4.1 Delphi 接口简介

在 Delphi 中， 对动态库 BFVoice.dll 的调用是通过 BFVoice.PAS 来声明的。

4.5.4.2 Delphi 对动态库的调用

同在 VB 中一样，在 Delphi 中也需要首先对 DLL 中的函数进行声明，然后才能对该函数进行调用。关于如何声明 DLL 中的函数，请参考 Delphi 中的 HELP 文件“external declarations。

对于 BFVoice.dll 的使用，用户可以参考声明文件 BFVoice.PAS。

4.5.4.3 注意事项

在 C 语言中的 char * 类型，要声明为 pchar。

4.5.4.4 Delphi 下的示范程序

Delphi 语言示范程序在 Delphi 6.0 的环境下开发。

安装在安装目录下的 Demo\Delphi6 子目录下。

4.5.5 PB 平台编程范例

4.5.5.1 PB 接口简介

在 Power Builder 中， 对动态库 BFVoice.dll 的调用是通过在“Decalre”菜单中的“Global External Function”来声明的。

4.5.5.2 PB 对动态库的调用

对于 BFVoice.DLL 的使用，用户可以参考演示程序。

4.5.5.3 PB 下的示范程序

PB 语言示范程序在 Power Builder 7.0 的环境下开发。

安装在安装目录下的 Demo\Pb7 子目录下。

4.5.6 其他编程语言的调用方法

在 98/NT/2000 的 32 位编程环境中，许多编程语言都可以用各种方式来调用 DLL 动态库。这样，也就可以通过对动态库 BFVoice.dll 的调用来控制 BFVoice 系列语音卡的硬件。

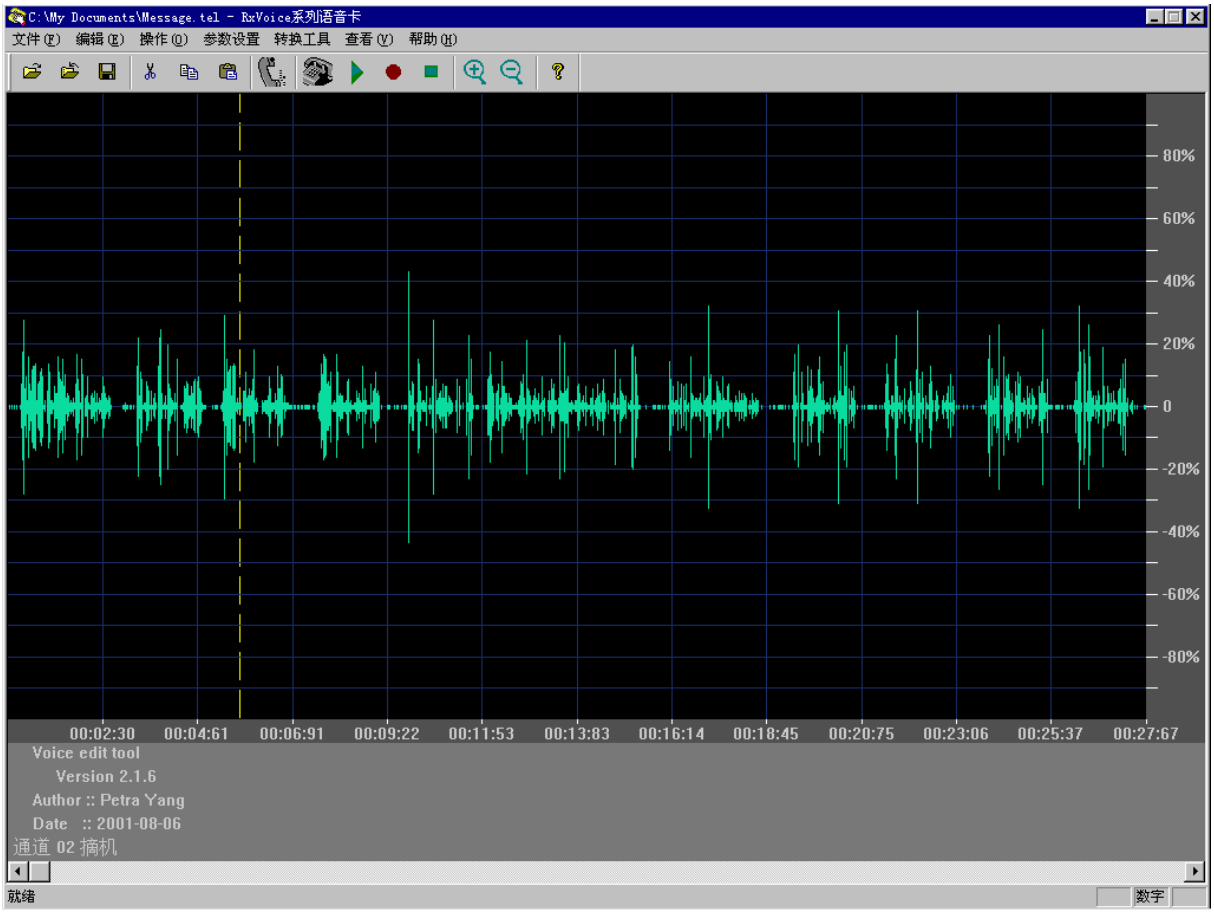
由于各种编程语言的方法不尽相同，本说明书无法一一描述。请用户仔细阅读你所用的编程语言的说明。

第五部分　BF-VOICE 系列产品的应用工具

BF-VOICE 系列语音卡为用户提供了诸如语音编辑、信号音分析等应用工具，下面详细介绍各个工具软件。

5.1　语音编辑工具——VoiceEdit

这个工具安装在用户安装目录的 Tools\VoiceEdit 子目录，运行界面如下：



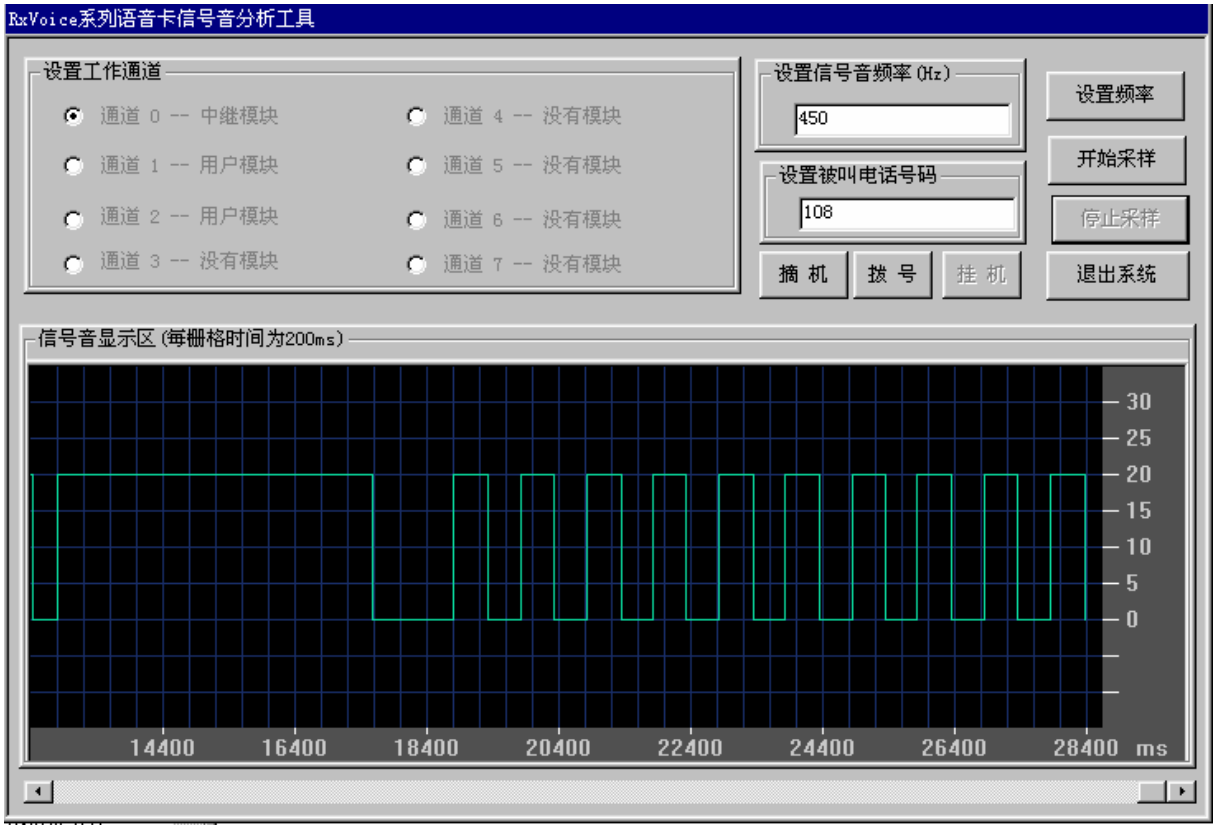
这个工具界面分为菜单条、工具条、语音数据显示区、标尺区以及状态显示区。标尺区显示语音数据的幅度和时间长度，状态显示区显示各个通道变化的状态。

利用菜单或工具条，用户可以转换工作通道、对中继通道进行摘机和挂机、同时对某个通道录音和放音等操作；也可以对语音数据显示区的图象进行放大和缩小。

总之，用户可以使用这个工具进行需要的语音编辑工作。

5.2 信号因分析工具——Analyze

这个工具安装在用户安装目录的 Tools\Analyze 子目录，运行界面如下：



这个工具用于确定中继通道的忙音、回铃音以及拨号音的频率和时长。

对于标准的信号音，其参数如下：

拨号音： 450HZ，持续通 5000 毫秒以上
占线忙音： 450HZ ，通 350 毫秒，断 350 毫秒
挂机忙音： 450HZ，通 700 毫秒，断 700 毫秒
回铃音： 450HZ，通 1000 毫秒，断 4000 毫秒

如果用户的中继线的信号音频率或时长不符合上述标准，需要使用以上工具进行确认，并在编程中使用相应函数设置，以使监控呼出过程或对方挂机检测精确。

5.3 专用语音处理工具——CoolEdit Pro

CoolEdit Pro 是一个第三方的专用的语音处理软件，该软件的安装版本附在用户安装目录下的 Tools\CoolEditPro 子目录，用户可参阅该软件的使用说明，使用该软件，这里不再叙述。

第六部分　BF-VOICE 系列产品的技术支持

深圳市博峰电子有限公司为 BF-VOICE 系列语音卡提供全面完善的售后服务和技术支持。

如果您在使用 BF-VOICE 系列语音卡时，遇到困难，可以遵循下列步骤处理：

1、仔细阅读相关产品用户手册的相关内容

2、检查您的源程序代码

3、参考 BF-VOICE 系列语音卡的 DEMO 例程

4、来电深圳市博峰电子有限公司询问

深圳市博峰电子有限公司将努力改进本公司的产品和用户手册，以使您的工作更省时省力。同时，我们也欢迎您对我们的产品和工作提出宝贵的意见，您可以使用您所喜欢的任何方式，在您方便的时候与我们联系，我们的联系方法是：

公司地址：深圳市福田区彩田路彩虹新都彩荟阁 10E

邮政编码：518033

联系电话：0755-82915501

联系传真：0755-82916502

E-MAIL　：support@szbofeng.com

http://www.szbofeng.com