数字音频规范与程序设计：基于Visual C++开发

曹 强 编著

中国水利水电出版社
www.waterpub.com.cn

数字音频规范与程序设计：基于 Visual C++开发

曹　强　编著

中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

本书是作者根据多年的工作经验而总结出来的有关 Windows 平台下数字音频规范与程序设计的一手资料，由浅入深地介绍了当今多数主流音频相关的技术规范与编程实现，适合于对数字音频感兴趣、准备或正在从事数字音频相关工作和开发的用户。

本书共分 11 章，涵盖了 Windows 平台下数字音频相关规范与编程的许多方面，详细介绍了 Windows 系统下高层与底层的多媒体音频接口（API）；WAVE 音频的全面解析、播放与录音；MIDI 音乐的全面解析、播放与录制；USB MIDI 驱动设计；MP3 的格式与编解码；乐音的基频检测；音频混音器原理与编程；低延迟 ASIO 音频驱动开发；普通音频插件与 VST(i)插件的设计；游戏开发中经常使用的 DirectSound 播放与录制等。

本书最大的特色是不但针对多媒体音频应用程序的开发做了较全面的介绍，可以满足大部分常规音频编程用户的需求，而且对音频驱动的开发也做了较多的切入。其中一部分是作者近几年独自研究的成果，对于有这方面需求的读者来说都是很宝贵的参考资料。

本书结构清晰，逻辑严密，内容具体且涉及面广泛，不但是从事多媒体（音频）开发与应用的广大开发人员的技术指导书，同时也可作为各高等院校相关专业、非相关专业师生重要的参考读物。

本书中每章都提供了一个或多个经典的编程实例程序，附带在光盘中赠送，光盘中提供了所有章节中实例程序的 Visual C++源代码。

图书在版编目（CIP）数据

数字音频规范与程序设计 : 基于Visual C++开发 / 曹强编著. -- 北京 : 中国水利水电出版社, 2012.6
ISBN 978-7-5084-9819-5

Ⅰ. ①数… Ⅱ. ①曹… Ⅲ. ①数字音频技术－技术规范②数字音频技术－C语言－程序设计 Ⅳ. ①TN912.2②TP312

中国版本图书馆CIP数据核字(2012)第112161号

策划编辑：杨庆川　责任编辑：李　炎　加工编辑：李　刚　封面设计：李　佳

书　　名	数字音频规范与程序设计：基于 Visual C++开发
作　　者	曹　强　编著
出版发行	中国水利水电出版社 （北京市海淀区玉渊潭南路 1 号 D 座　100038） 网址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）68367658（发行部）、82562819（万水）
经　　售	北京科水图书销售中心（零售） 电话：（010）88383994、63202643、68545874 全国各地新华书店和相关出版物销售网点
排　　版	北京万水电子信息有限公司
印　　刷	三河市铭浩彩色印装有限公司
规　　格	184mm×260mm　16 开本　29.5 印张　715 千字
版　　次	2012 年 6 月第 1 版　2012 年 6 月第 1 次印刷
印　　数	0001—3000 册
定　　价	58.00 元（赠 1CD）

推荐序一

当得知要出版这本关于数字音频规范与程序设计的书，仔细浏览过目录后我感到非常意外和高兴。意想不到的是，在我们得理乐器公司研发部还有一名工程师不为个人利益，愿意牺牲自己的业余休息时间，用近三年的时间持之以恒完成了本书的撰写和CD中应用程序实例的设计。我知道，如今写一本关于技术方面的书籍实际上是没有多少经济收入的，写这类书的人也越来越少了，所以，我想作者的这种坚持不懈而又愿意分享的精神是非常值得肯定和称赞的。高兴的是，从书籍的目录可以看出，此书的内容非常专业，涉及面也很广泛，鉴于我国在数字音频与音乐技术研发领域还处于相对弱势的局面，我想此书的公开出版必将为音频领域的技术工作者提供一手宝贵的参考资料，同时为推动国内数字音频与音乐技术及应用领域的发展做出一定的贡献。

据我了解，作者因完成此书的写作耗费了不少的时间和精力，一部分是用来研究新的技术和国内还较少人研究的技术，比如VST插件、软音源、ASIO音频驱动、基音检测等。此外，为了给读者提供易懂而实用的程序实例，他也花费了不少心思，光盘中好几十个的实例程序，每个都是经过他精心设计和完整测试的，实例程序中封装的一些模块，很多都是可以直接或稍作修改就可以移植到其他项目中的。可想而知，作者是怀着极大的热情和坚定的信念去写这本书的，先不去考虑书中内容的深浅和价值，技术的精粗，就作者的这种诚心写作的精神便值得鼓掌赞美。

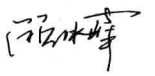
尽管我现在已在得理做了十多年的总经理，但我也是从研发工作做起的，有十多年的研发管理经验。作者也是我亲自面试进入得理研发中心的软件工程师，还记得作者刚进我公司的那段时间，正好有一个底层的音源驱动程序要进行平台的移植，对于一个以优异成绩从国防科学技术大学计算机学院毕业的高材生，我想将这个较有难度的项目交给他应该没有问题，事后证明我是正确的。我发现，那些能够进行持之以恒的钻研，肯在实践中锲而不舍地学习，总是能够不断地解决一个又一个的问题，当大部分人选择了放弃他还在坚持时，通常他就能够掌握核心技术，我认识的作者就是这样的人。而且他在业余时间还能坚持自己的兴趣爱好，为此他在短短的时间内，不知不觉学会了钢琴和吉他的基本演奏，所以我想，勤奋和坚持是成功必备的两个基础条件，否则成功就无从谈起。

这本书重点介绍了Windows平台下音频应用程序的开发，包括多种主流音频格式的解析和编解码实现，同时通过对USB MIDI用户层和核心层驱动程序、ASIO用户层音频驱动程序实现过程的讲解，使读者对Windows音频驱动程序的开发模式和过程也有一个较详细的了解。作者将他多年在实践中积累起来的经验整理成书，必定凝聚了不少的心血。我想，通过对本书的学习，后学者应该能够少走不少的弯路，尤其是那些正打算或正在从事与数字音频应用程序或驱动程序相关工作的编程者，我认为这本书是一个非常不错的选择。

事实上，作者那里已成文的文章不止本书中这些，但考虑到整体一致性以及出版的特定

要求，其中一些被筛选下来，比如基础乐理常识、乐谱显示 MusicXML、DirectMusic 等内容都没有出现在本书中。当时我给他的建议是，要么分为上下两册出版，上册为入门篇，下册为提高篇；要么进行缩减，把未列入书中的内容以其他形式出版，比如以论文的形式发表到相关的刊物上，或者通过博客、论坛等现代化方法以电子文档的形式进行分享。作者最终还是选择了后者，那些未在本书露面的文章内容，我想会在作者的博客或个人空间上与大家分享的。

此文送给《数字音频规范与程序设计：基于 Visual C++开发》的所有读者以及那些打算从事音频行业的工作者们，同时希望会有更多数字音乐领域内高水平复合型人才出版更多、更精彩的相关书籍，以共享、学习和交流的方式来进一步推动音频行业的向前发展。



2012 年 2 月 11 日于得理工业园

推荐序二

我早在国防科学技术大学读研究生的时候，便与他相识；他不仅是编程高手，而且是音乐爱好者；我们有幸一同供职于国内最大的电子乐器研发、生产、销售公司——得理集团。他就是本书的作者——曹强。

很早便与他讨论过一个问题，那就是偌大的中国，很难买到一本令人满意的有关数字音频程序设计方面的书籍，要么是外文原版，要么就是外文翻译而来。而在“拿来”的过程中，一些重要的专业术语有的不够准确，甚至是错误的。探其原因，一方面是中国数字音频行业起步较晚，在数字音频和数字音乐的专业技术上还是较为落后；另一方面是撰写一本专业书籍往往要花费作者大量的精力和时间，而这往往与作者所得到的回报不成正比，所以很多这方面的专家没有时间或不愿花时间把自己所掌握的专业知识整理成书。也鉴于此，他当时表示要写一本真正实用的数字音频程序设计的书。现在呈现在大家面前的，便是他在工作之余，花费了近三年的时间完成的这本优秀的数字音频程序设计著作。

对于一个软件工程师而言，可能经常被编程中的种种问题所烦恼，但驾驭的欲望和乐趣也在其中。在编程的过程中，你就是导演，并且拥有绝对的控制权。在编程中，脑海里思考的自然是怎样设计编排才能完成预定的逻辑功能，于是你必须划定不同的功能模块来充当不同的角色。当一段完整的代码运行之后，如果程序完全按照你的意图工作，这真是很鼓舞人、更是一件很奇妙的事情。

本书中有大量的实例程序，作为演示程序的编排者，作者认真地对待每个应用程序，在达到演示作用的同时还尽量做到界面友好，充分体现了人机交互性和表现力，特别是一些看似无关重要的参数和效果，他都用可见的图形方式更直观地显示出来，便于读者理解。对作者的这种科学而严谨的态度，我是非常赞赏和敬佩的。

相信不管是因为爱好音乐或是编程需要，不管是从事上层应用程序开发还是进行底层驱动开发，不管使用的是C、C++还是Java，读者都能够从这本书中获得帮助和启发。希望这本书能够成为每一位关注数字音频程序设计的读者的良师益友。

2012年2月18日于得理工业园

前　言

音频是多媒体技术中非常重要的一部分，它往往与视频结合在一起广泛应用于影视娱乐、多媒体系统、电子乐器、游戏设计、广告设计等诸多行业。近几年来，随着多媒体技术的迅速发展，音频所涉及的领域也越来越广泛，越来越深入人们日常生活的方方面面，多媒体开发也是近年来开发人员所关注的重点。

本书包含了 Windows 平台下音频开发的许多方面，主要由如下几部分组成：Windows 多媒体音频 API；多种主流音频格式及其播放与录制，包括 WAVE、MIDI、MP3 等；数字音乐合成；音频插件；基频检测；MP3 编解码等。为了使读者能够抓住重点，把握解决问题的方法，书中每章都安排了由 Visual C++实现的一个或多个实例程序，力求做到理论联系实践，所见即所得。

因本书涉及的概念和专业术语比较多，并且在实际应用中所需要的原理也比较广泛，建议读者在阅读本书时可以先略过不懂的概念或词汇，因为篇幅的限制也许对它们没在书中做出详细的解释，不过有关乐理的常识大部分在本人的博客上可以找到详细的解释和说明，有此方面需求的读者请关注我的博客空间：http://blog.sina.com.cn/consonance。

虽然本书的所有实例程序均基于 Visual C++，但为了能够使不同开发平台下的开发人员也能很好地理解其实现原理与过程，一些核心代码在书中都有详细的解释，并且大部分关键代码都已通过类的形式封装好，因此要移植到其他的开发平台也不是很困难的事。当然，本书的第 1 章对 Visual C++的入门做了最基本的介绍，目的就是希望每位读者都能够熟悉 Visual C++的一些基本使用方法，会使用并看懂书中的所有实例程序。

本书的所有内容都是作者在实际工作中总结出来的宝贵而实用的经验，有些还是个人多年来专心研究的成果，比如 VST(i)插件、ASIO 音频驱动等。书中所有程序实例也是本人精心挑选设计而成，我想对不少读者来说应该都是很有参考价值的，某些代码甚至可以直接应用在你的项目开发中。CD 中的所有源代码都通过了 Visual C++ 6.0 和 Visual Studio 2005 的编译。

本书既可以作为广大音频行业工作者的参考书，也可以供从事多媒体相关研究和设计的院校师生使用。

应该说本书的创作过程是艰难的，能够以这样全面和完整的方式与读者见面，我想离不开我的家人、朋友和同事以及中国水利水电出版社的全力支持。

感谢顾冰峰先生和郭润博先生为本书写的序，使我的作品蓬荜生辉，特别是顾总在本书的构架和篇章的安排上给了我很多非常好的建议，从而加速了此书的出版。

感谢公司董事长郑荃文，因为得理公司给了我很好的学习环境和生活环境，使我有更多的学习机会，在这样的环境中我才能静下心来完成此书。

感谢中国乐器协会电鸣乐器分会会长盛子斐先生对本书的特别指导和宣传。

感谢同事谢奇彬和陈乐在乐理方面、谭晓文和王建军在编程方面给予我的帮助。

感谢中国水利水电出版社杨庆川老师的耐心指导和帮助，她请业内专业的老师看了本书的内容简介和目录后提出了几点非常中肯的意见，比如第 1 章的内容就是我后来添加进来的。

感谢中国水利水电出版社编辑李炎老师的信任和支持，将书中的错误降至最低，使本书更加完整。

感谢我的父母（曹子云、王梦科）和岳父岳母（刘国建、田双华），谢谢他们这几年的所有辛勤付出和源源不断的精神鼓励。

最后，特别感谢我的妻子刘君，因为写作占用了我很多休息时间，确切地说是三年的业余时间，很多时候都忽略了她的感受，还要承担家里的家务活，在此我要对她说一声辛苦。

希望本书对读者有所帮助，也希望能够为音频行业的发展做出一点点贡献。尽管在编写本书时尽了最大的努力，但由于工作繁忙和限于作者的水平，书中难免会出现一些不足甚至错误之处，还望广大读者给予批评和指正。

曹强

2012 年 3 月 24 日

目　录

第1章 Windows音频体系与Visual C++开发基础

导 读

作为本书的开篇之章，本章意在向读者介绍微软的Windows平台下音频体系结构是如何构成的，以及在这个体系架构下，针对不同用途的音频应用程序或驱动程序，我们应该通过何种有效的方式来进行快速而高效的开发。同时，本章也提到了Windows平台下有关声音延迟的问题及产生的原因，并提供了解决问题的方法。总之，在实际中，我们应该根据系统的复杂性、应用场合以及软件实时性要求的不同来选择最恰当的开发途径。

此外，本章还简单介绍了Visual C++开发的相关基础知识，包括Windows应用程序的驱动模型、MFC框架的简介和开发环境的概述等，并通过一个简单的实例程序详细介绍了开发一个基于对话框的应用程序的全部过程。当然，在后续的章节中还会有一些关于Visual C++使用的方法和步骤，例如动态链接库（DLL）文件的制作和使用，特殊控件的使用等。

本书的主要目的是介绍数字音频相关的规范以及程序设计与实现，并通过Visual C++开发平台进行结果演示。事实上，无论何种开发平台，其实现思想和过程多数都大同小异，只是希望通过本章的讲述能给那些没有相关开发经验或者未曾使用过Visual C++的读者一个基本的指导，为学习后面的其他章节做一定的准备。

1.1 认识声音与音频

多媒体技术是利用计算机对文字、图像、图形、动画、音频、视频等多种信息进行综合处理，建立逻辑关系和人机交互作用的产物。

Microsoft Windows对多媒体的支持起源于1991年的所谓Microsoft Windows多媒体扩展功能（Multimedia Extensions to Microsoft Windows）。1992年，Windows 3.1的发布使其对多媒体的支持成为一类独特的API。从20世纪90年代后期开始，声卡、CD/DVD-ROM驱动器、游戏、多媒体应用程序开始慢慢普及，至今很多已成为了PC电脑的标配。可以说多媒体技术的发展大大推进了Windows可视化图形界面的进程，从此，计算机因为多媒体元素的加入而更富有生命力和影响力，已不再是单纯处理文字和数字的传统机器。

随着多媒体技术的高速发展，如今，在影视娱乐业、多媒体办公系统、医疗影像与诊断系统、平面/3D广告设计、电子图书、教育培训等行业都可以看到它的身影。

Windows对多媒体提供了非常广泛的支持，涉及的内容也很广泛，但本书主要介绍的是Windows平台下与音频相关的一些标准与程序设计。在多媒体应用领域内，声音是携带信息的极其重要的媒体，大约占到百分之二十左右。下面先看看声音是如何定义的，以及

它与音频是什么样的关系。

声是通过空气传播的一种连续的波，叫声波。音是人类的一种认知能力，即音可以被人所认知和理解。声音的基本定义决定了声波的产生必须具备两个必要的条件，即物体的振动和传输声波的媒质。

我们在初中物理就已经学过，声波具有频率、周期、波长、声速、波数等状态参量。此外，声波跟光波一样也具有反射、折射和衍射现象。声波的基本属性包括：传播速度（声速）、强度、频率（或频谱）和波长。

声音信号是由许多频率不同的分量信号组成的复合信号，复合信号的频率范围称为带宽。频率和振幅均为常数的声音称之为纯音。

音频，由英文 Audio 翻译而来，它是指人类所能听到的声音（声音频率范围为 20Hz～20kHz），因此它的研究范围不包括次声波和超声波。

音频一般分为如下三类：语音、乐音和其他音效（包括自然声和环境声）。

语音是人的说话声，它是一种特殊的媒体，也是一种波形声音，更重要的是它包含有丰富的语言内涵，可以经过抽象，提取特定成分，理解其意义，语音的频率范围一般为 80Hz～3400Hz。语音方面的研究，如今最为热门的当属语音识别，语音识别又是一门交叉学科，它涉及的领域比较广泛，包括信号处理、模式识别、概率论和信息论、发声机理和听觉机理、人工智能等。虽然本书第 8 章讲述的时域和频域中的基频检测方法与语音识别有一定的关系，因为基频特征提取是语音识别中一个非常重要的步骤，但语音识别不在本书的研究范围之内。

本书所讲述的内容主要是关于音乐方面的，比如 WAVE、MIDI、MP3、基频检测等。其中也涉及一部分音效的内容，比如音频插件（VST）、DirectSound 中的音效处理等。音乐中的音又分为乐音与噪音。乐音是指形式更加规范，符号化了的声音，这是我们研究的主要对象。一般的，振动规则且听起来有明显高低且发声悦耳的声音就叫做乐音，反之则为噪音。虽然音乐中使用的音以乐音为主，但并不是说噪音不重要，比如在乐队中，军鼓、锣、钹等打击乐器都是很常见的，虽然它们没有明确的音高，被划为在噪音范畴，但它们的节奏性在音乐进行和氛围渲染方面同样有着特别重要的意义。

1.2 Windows 音频体系层次结构

图 1.1 展示了 Windows 音频体系的层次结构。从图中可以看出，多媒体应用程序可以通过不同的途径与音频设备进行通信，从而实现音频的播放与录制。

因为本书的多数内容都是基于 Windows 音频系统的，因此，在阅读其他章节内容之前，建议读者先对 Windows 音频体系的总体框架有所了解。

1.2.1 用户模式与内核模式

从 Intel 80386 开始，出于安全性和稳定性方面的考虑，Windows 把 CPU 运行权限的级别从高到低分为了四个等级：Ring0～Ring3。运行于较低级别的代码不能随意调用高级别的代码和访问较高级别的数据，且只有运行在 Ring0 层的代码才可以直接对物理硬件进行访问。由于 Windows NT 是一个支持多平台的操作系统，为了与其他平台兼容，它只利用

了 CPU 的 Ring0 和 Ring3 两个级别，即我们通常所说的内核模式和用户模式。

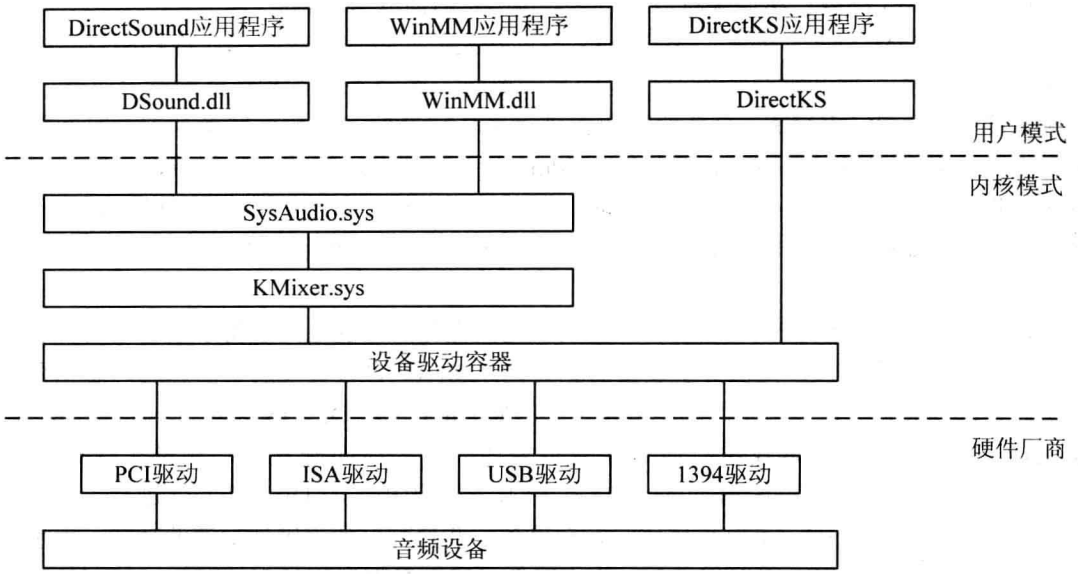


图 1.1 Windows 音频体系层次结构图

所有的应用程序和操作系统的用户接口部分（Win32 API）都是在用户模式下运行的，而操作系统的核心部分，如 I/O 管理器、设备驱动程序等运行在内核模式。

因为在用户模式下，我们不能获得系统数据的存取权，因此，应用程序在需要读取或存储设备数据时，就需要调用 Win32 API 函数。如图 1.1 所示，在 Windows 音频体系中，应用程序可使用的 API 有：Multi Media API 和 DirectSound API，它们分别封装在 WinMM.dll 和 DSound.dll 中，本书的第 2 章和第 11 章对它们分别进行了详细的介绍。此外，应用程序也可以通过 DirectKS 直接与音频驱动进行通信，使用 DirectKS 的目的在于获得声音的低延迟处理。

在内核模式下，音频驱动 sysAudio.sys 决定进入它的音频流是否需要进行格式的转换和采样率的转换（重采样），然后交由音频混合器驱动程序 KMixer.sys 执行真正的转换操作，经过数据格式重整的音频流随后交给相关的设备底层驱动进行特定的处理。

所有的 Windows 音频驱动都是基于 WDM 驱动模型的。WDM 是 Windows Driver Model 的缩写，意为 Windows 驱动模型，也是一个跨平台的驱动模型。它在 Windows 系统中构建了一些实例化的概念和思路，特别是一些结构或半结构的东西，我们通过对这些结构进行适当的调整和处理，可以较快地开发出很好的驱动程序。WDM 驱动的核心是 IRP，即输入/输出请求包（I/O Request Packet）处理，它通过 IRP 与音频设备或它的下一级驱动进行通信，完成特定的功能。不同的音频设备由生产厂商提供不同的设备驱动。

1.2.2 关于声音的延迟问题及解决方法

在本书的第 2 章，对 Multi Media API（Windows 的底层多媒体编程接口）进行了较详细的介绍，包括波形音频处理、MIDI、多媒体计时器、多媒体文件 IO 和游戏操作杆控制等多个方面。在应用程序中使用 WinMM API 相对而言比较简单，可以使用回调机制，通

过调用 waveOutOpen、waveOutWrite、waveInOpen、midiOutOpen、midiOutShort、midiInOpen、midiInStart 等 API 函数，从而实现波形音频或 MIDI 的播放与录音。

虽然 WinMM、MCI 实现了 Windows 与设备无关的设计思想，不允许程序直接操作多媒体硬件设备，但却没有达到高性能的另一要求。因为不管音频格式是什么，也不管音频采样率是多少，KMixer 总是要将数据进行采样转换的，而数据转换是需要时间的。事实上，音频数据经过 KMixer 至少存在 50ms 以上的延迟，在整个过程中，音频数据从音频软件到音频设备发出声音，延迟时间可能会在 100ms 以上，有时甚至可达到 200ms 以上。因此，那些对延迟较为敏感且实时性要求很高的音频应用程序，WinMM 显然不是理想的选择。

之后 Windows 开发出了 DirectX，在设备无关性与高性能之间取得了某种平衡，它既允许程序直接操作硬件，又保持了设备的无关性。Direct 含有直接的字面意思，意指可直接访问多媒体硬件，X 是指不同的组成部分，例如 Direct3D、DirectSound、DirectMusic 等。DirectX 是微软为创建游戏和其他高性能多媒体应用程序而开发的一种基于 COM 的 Windows 底层 API。在本书的第 11 章对 DirectX 中与音频相关的 DirectSound 进行了详细的介绍。使用 DirectSound，可以直接访问音频设备，并且可以使用硬件加速的功能，更重要的是，由于它允许直接操作硬件设备，大大降低了音频数据中间处理的时间，从而使延迟降低到 50ms 左右。因此，那些对延迟不是特别敏感，实时性要求不是特别高的应用程序，使用 DirectSound 也许是个不错的选择。

虽然 DirectSound 和 DirectMusic 在高性能和设备无关之间取得了某种平衡且大大降低了声音延迟的时间，但人耳对声音的实时性是非常敏感的，一般只要声音的延迟超过 10ms，经过训练的音乐专业人士或耳朵特别灵敏的人就可以感觉出来。在某些特定情况下，如图 1.2 所示，演奏者演奏乐器并通过电脑上的音频软件实时回放时，延迟时间（从演奏动作开始到人耳听到声音时为止所经历的时间）不应超过 10ms，否则就会明显感觉到声音带有滞后，从而影响到演奏者的水平发挥。也就是说，若要求延迟时间不超过 10ms，即使是使用 DirectSound 也无法达到这个要求。

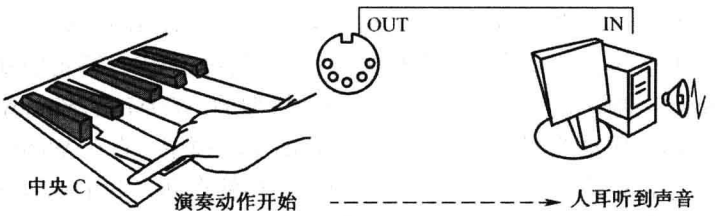


图 1.2　声音延迟示意图

那么，有没有办法解决延迟的问题呢，或者说有没有可能在 Windows 下实现“零延迟”呢？答案是肯定的，使用 DirectKS 可以实现比 DirectSound 更低的延迟。注意，我们平常所说的零延迟实际上是指声音延迟低于 10ms 的延迟，实际中不可能存在真正的零延迟，因为不管是声音信号在电缆中的传输，还是电子乐器自身对音符的处理或电脑声音驱动程序对音频流的处理，都会消耗一定的时间。

从图 1.1 中我们可以看出，DirectKS 是直接与音频设备驱动通信的，其延迟时间基本

上就是数据拷贝的时间以及硬件本身产生的延迟时间。当然，前提条件是进入 DirectKS 之前的音频数据必须是 WAVE 格式的，有时，声音源数据本来就是 WAVE 格式的，音频软件不必进行任何格式的转换就可以通过 DirectKS 直接送往音频设备驱动。本书的第 9 章在介绍用户层 ASIO 驱动编程时，详细介绍了如何通过 DirectKS 实现一个虚拟的 ASIO 驱动。

我们通过 DirectKS 技术确实可以达到真正的低延迟要求（10ms 以下），不过有点遗憾的是，DirectKS 是一项微软未公开的技术，虽然微软提供了全套的源代码（包括一个名为 kssample 的实例程序，我们可从微软的官网上下载），但它们都是未被文档化的，我们在 MSDN 里面也找不到它们的相关资料。也就是说，即使微软在未来的某个时候改变其接口或内部实现，或者在未来的操作系统中不再使用它们，微软也不必通知我们。事实上，从 Windows Vista 开始的音频驱动 WaveRT（Wave Real-Times 实时波形）已经可以替代 Windows XP 下面的 ASIO 和 DirectKS，使用 WaveRT 的音频软件可以直接与音频接口的缓冲区和采样点位置进行通信，而不必像 ASIO 那样需要进行用户模式到核心模式的转换，所以，从这个原理上讲，WaveRT 比 WDM 和 ASIO 都要高效，但事实上并没有我们想象中的那么理想。当然，DirectKS 在相当长的一段时间内都将是稳定的，我们完全可以使用它来开发低延迟的多媒体应用程序（至少目前的 Windows 7 系统都还是支持 DirectKS 的。实际上，当今十分流行的千千静听播放器和 ASIO4ALL 虚拟 ASIO 驱动都使用了这门技术）。关于 DirectKS 更多的信息在本书的第 9 章有更为详细的介绍，对低延迟感兴趣的读者可参考这一章。

1.3 Visual C++开发基础

本书中所有的实例程序都是基于 Visual C++开发的，因此有必要对于那些刚开始或正想使用 Visual C++进行开发的人员进行一个基本的指导，以降低开发门槛，快速上手并能看懂 CD 中的源程序。注意，CD 中的源码都是通过 Visual C++ 6.0 和 Visual Studio 2005 编译的，因此不存在向下兼容的问题，如果使用的是更高的版本，请自行进行转换。源码中有不少非常实用的代码都以类的形式进行了封装，可以直接（或稍加修改）移植到项目或其他工程中。

基于篇幅的限制，这里只对 Visual C++进行入门级的介绍，并且假定读者对 C/C++语言有一定的基础，更多关于 Visual C++的高级开发请参考其他书籍和资料。当然，如果有相当的 Visual C++开发经验，那么可以跳过这部分的内容。

1.3.1 区分 C、C++和 Visual C++三者的关系

C 语言是一种结构化语言，它主要侧重于算法和数据结构，因此 C 程序设计是一种面向过程的开发方式，它强调的更多的是解决问题的算法，并通过适当的数据结构将相应的过程转化为可执行的代码。

随着计算机应用领域越来越广泛，许多软件变得越来越复杂，代码量也越来越庞大，开发团队人员不断增加，这时候面向过程的开发模式就显得有些吃力了，于是一种称为面向对象的软件开发方法应运而生，这就是后来的 C++，它在 C 语言的基础上增加了面向对象的功能，故从某种程度上讲，C 语言是 C++的子集。

但需要注意的是，虽然 C++是在 C 语言的基础上发展起来的，但并不是说 C++比 C 语