**乐器数字接口(MIDI)简介**

# 1. 概述

乐器数字接口（Musical Instrument Digital Interface，MIDI），是20世纪80年代初为解决电声乐器之间的通信问题而提出的，是电子乐器制造商们建立的通信标准，是电子乐器之间以及电子乐器与电脑之间的统一交流协议。从一个MIDI设备转送到另一个MIDI设备上去的数据就是MIDI消息（MIDI Message）。

例如，在电子钢琴上按下某个音符对应的键（称为开音），则产生一个MIDI消息，该消息记录了所按的键、力度；当松开所按的键时（称为关音），便产生另一个MIDI消息用以表示所松开的键及松开的速度。当然这些信息都是易编辑的。

显然，MIDI消息不是音频波形的数字化，而是一种电子乐谱。MIDI已成为编曲界最广泛的音乐标准格式，可称其为“计算机能理解的乐谱”。

# 2. 音符数字化

现行的国际标准音高为1939年5月国际标准协会在伦敦通过的a1（小字一组的A）的为440Hz，亦称“第一国际音高”或“音乐会音高”。

十二平均律（十二等程律）是一种音乐定律方法。它将一个倍频率按几何平均分成十二等份形成八度，每等分称为半音（两个半音之间的频率为倍的关系）。

表1 小字组和小字一组音符音名对照表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MIDI note 音符编号 | 音名 (小字组) | 频率/Hz | MIDI note 音符编号 | 音名 (小字一组) | 频率/Hz |
| 48 | c | 130.8 | 60 | c1（中央C） | 261.6 |
| 49 | #c | 138.6 | 61 | #c1 | 277.2 |
| 50 | d | 146.8 | 62 | d1 | 293.7 |
| 51 | #d | 155.6 | 63 | #d1 | 311.1 |
| 52 | e | 164.8 | 64 | e1 | 329.6 |
| 53 | f | 174.6 | 65 | f1 | 349.2 |
| 54 | #f | 185.0 | 66 | #f1 | 370.0 |
| 55 | g | 196.0 | 67 | g1 | 392.0 |
| 56 | #g | 207.7 | 68 | #g1 | 415.3 |
| 57 | a | 220.0 | 69 | a1 | 440.0 |
| 58 | #a | 233.1 | 70 | #a1 | 466.2 |
| 59 | b | 246.9 | 71 | b1 | 493.9 |

MIDI采用7位二进制数表示音符（编号0~127，共128个），相邻两个音符为半音关系。这128个音符除包含了钢琴上的88个音外，还向下扩展21个音符、向上扩展19个音符。因此，编号为21对应钢琴上最左侧的键。中央C（小字一组的C，即c1，频率是261.6Hz）的编号为60=0x3C（参见表1，其他音符类推）。

# 3. 音量数字化

MIDI采用7位二进制数表示演奏某音符的音量（或演奏力度）。一般地，可参考图1所示的对应关系。



图1 音量（力度）数字化参考标准

# 4. MIDI音色与通道

不同的乐器所发出的乐音音色是不同的。MIDI采制了多种乐器演奏不同音符时的实际音频数据，形成音色库备用。MIDI消息传送给MIDI设备的音源驱动程序后，音源驱动程序首先解析MIDI消息，从音色库中找到相应的采样样本，根据音高、力度等参数，利用音频合成技术合成具体的音频，最后输出到扬声器。

这就好像“字库”一样，不同字体的字库安装到计算机系统中，可根据文本字符的编码在字库中找到指定字体的字库，再根据字号、颜色等参数生成具体的字符供显示或打印。

在MIDI规范制定之初，仅预留了128个音色。后来增加了音色库选择控制，使音色库的音色总数达16384×128=2097152种。基本的音色参见附录1。

MIDI定义了16个逻辑通道（channel），每个逻辑通道可指定一种乐器（音色）。这16个通道共享一条电缆传递MIDI消息。通道编号0~15，其中通道9被指定为打击乐通道，此时的音符编号为打击乐器，参见附录2。

# 5. MIDI编程

MMAPI（MultiMedia API）是Windows的底层多媒体编程接口，包括波形音频处理（44个函数）、MIDI函数（41个函数）、多媒体计时器管理（8个函数）等。下面介绍几个常用的MIDI输出函数。

使用MIDI函数需要包含mmsystem.h头文件，该头文件已在windows.h文件中包含了，故#include <windows.h>即可。

一般地，还需要设置连接库（winmm）或连接选项（-lwinmm）以连接winmm.lib库。Debug和Release两种配置需要分别设置。

以MinGW Developer Studio集成开发环境为例。在主菜单下依次点击Project | Settings...[Alt+F7] | Link | Libraries:下方的编辑栏中输入winmm（参见图2）。或者 Project | Settings...[Alt+F7] | Link | Extra linking options:下方编辑栏中输入 -lwinmm。

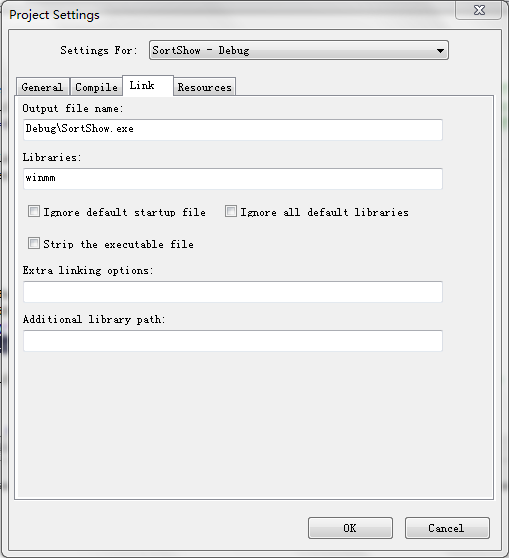


图2 连接winmm库设置

# 6. chen有关MIDI介绍

一、winmm.lib库

MIDI播放需要动态链接库WINMM.LIB或者WINMM.DLL。

Cannot open file winmm.lib Visual Studio 2017解决方案：

<https://learn.microsoft.com/en-us/archive/msdn-technet-forums/8ca8a5bb-7fca-48ce-b115-e9bc3eb79381>

I think the first thing you should do is verify that these libraries actually exist on disk. Assuming that Windows is installed on C, you should find these at:

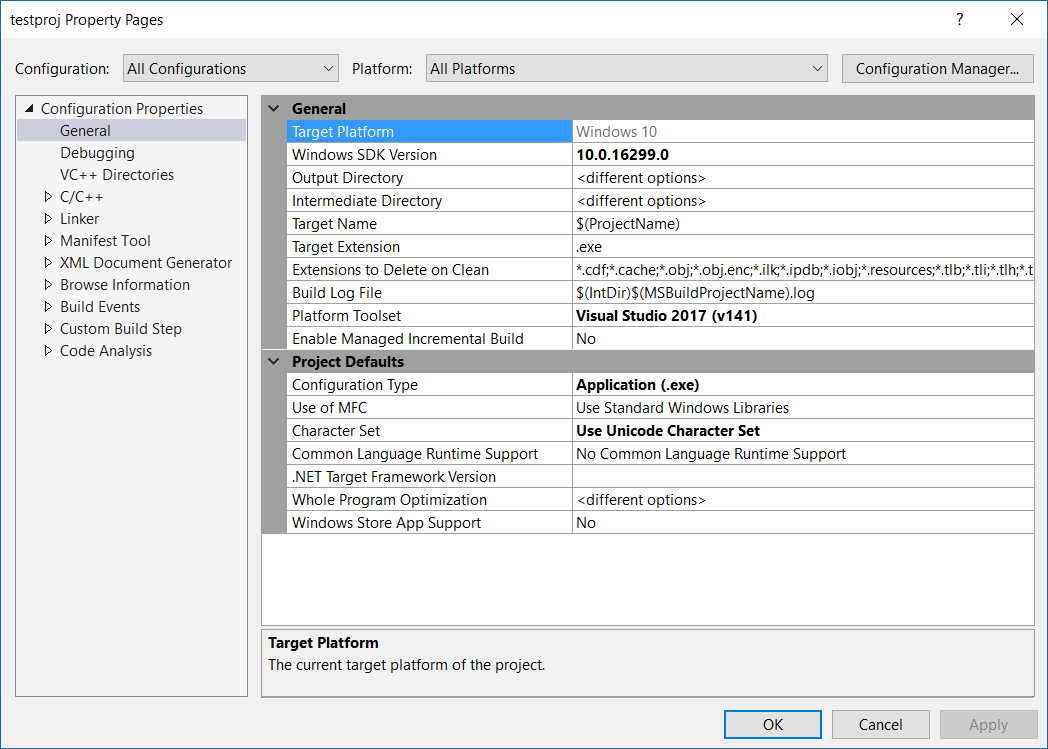
C:\Program Files (x86)\Windows Kits\8.1\Lib\winv6.3\um\platform>

or:

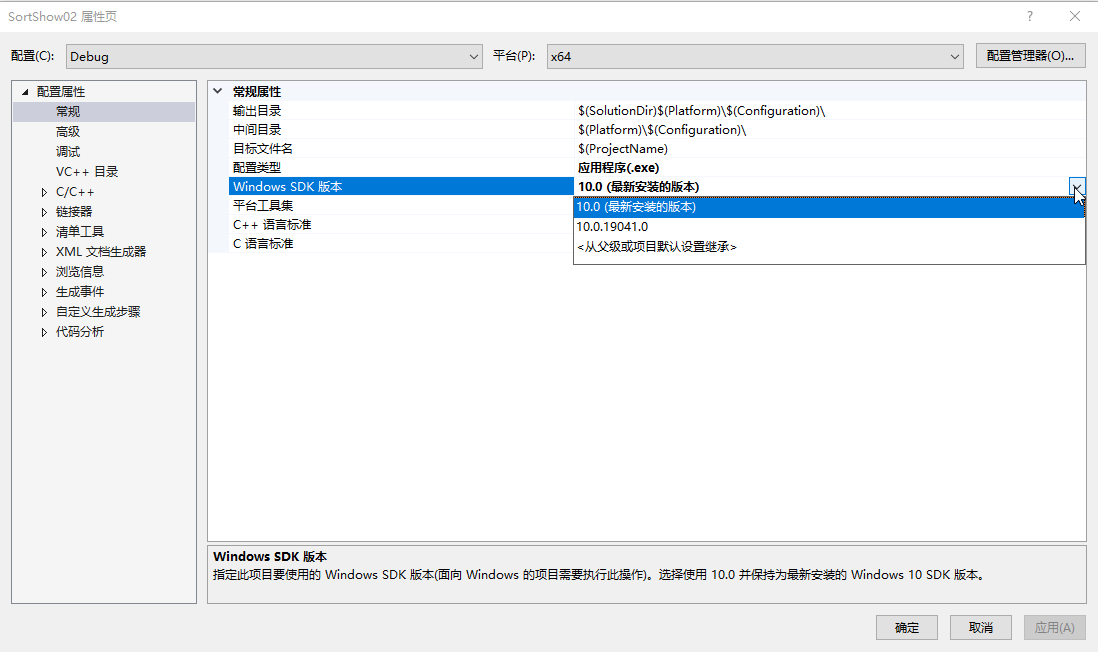
C:\Program Files (x86)\Windows Kits\10\Lib\<version>\um\platform>

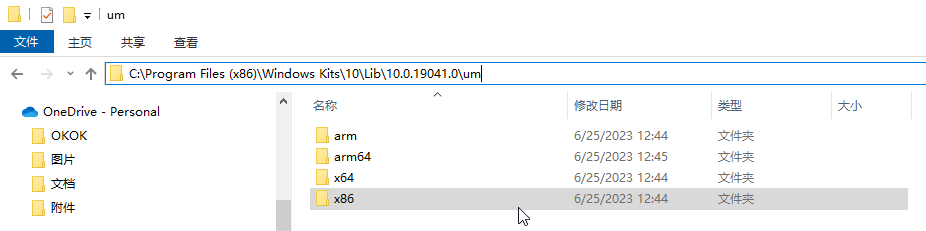
chen我的是C:\Program Files (x86)\Windows Kits\10\Lib\10.0.19041.0\um\x64

Depending on which Windows SDK you have installed. The <platform> will match the platform you selected in Visual Studio (it is the bit at the top that has Debug/Release and x86/x64/arm). The version in the Windows 10 SDK will match the version in the project properties.

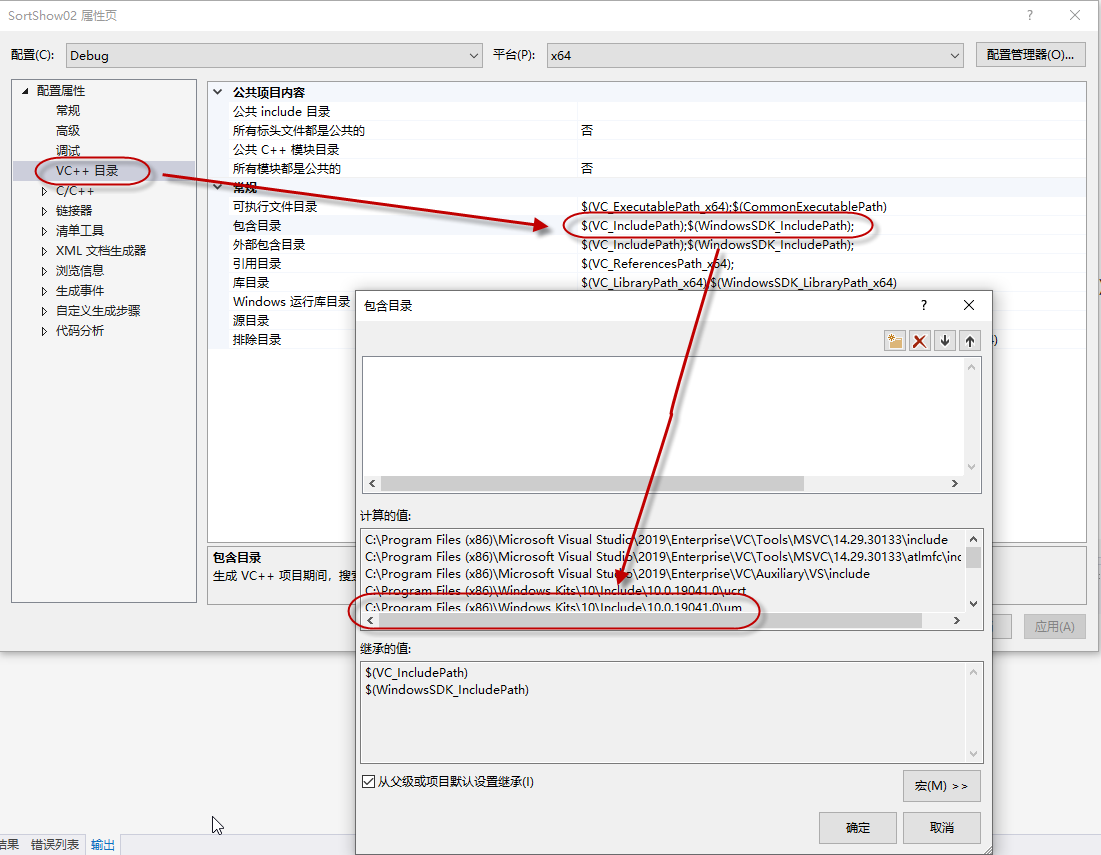


chen安装的是VS 2019，我的windows SDK是10.0.19041.1,所以，正常情况下，我的winmm.lib应该在这个目录下C:\Program Files (x86)\Windows Kits\10\Lib\10.0.19041.0\um\x64。



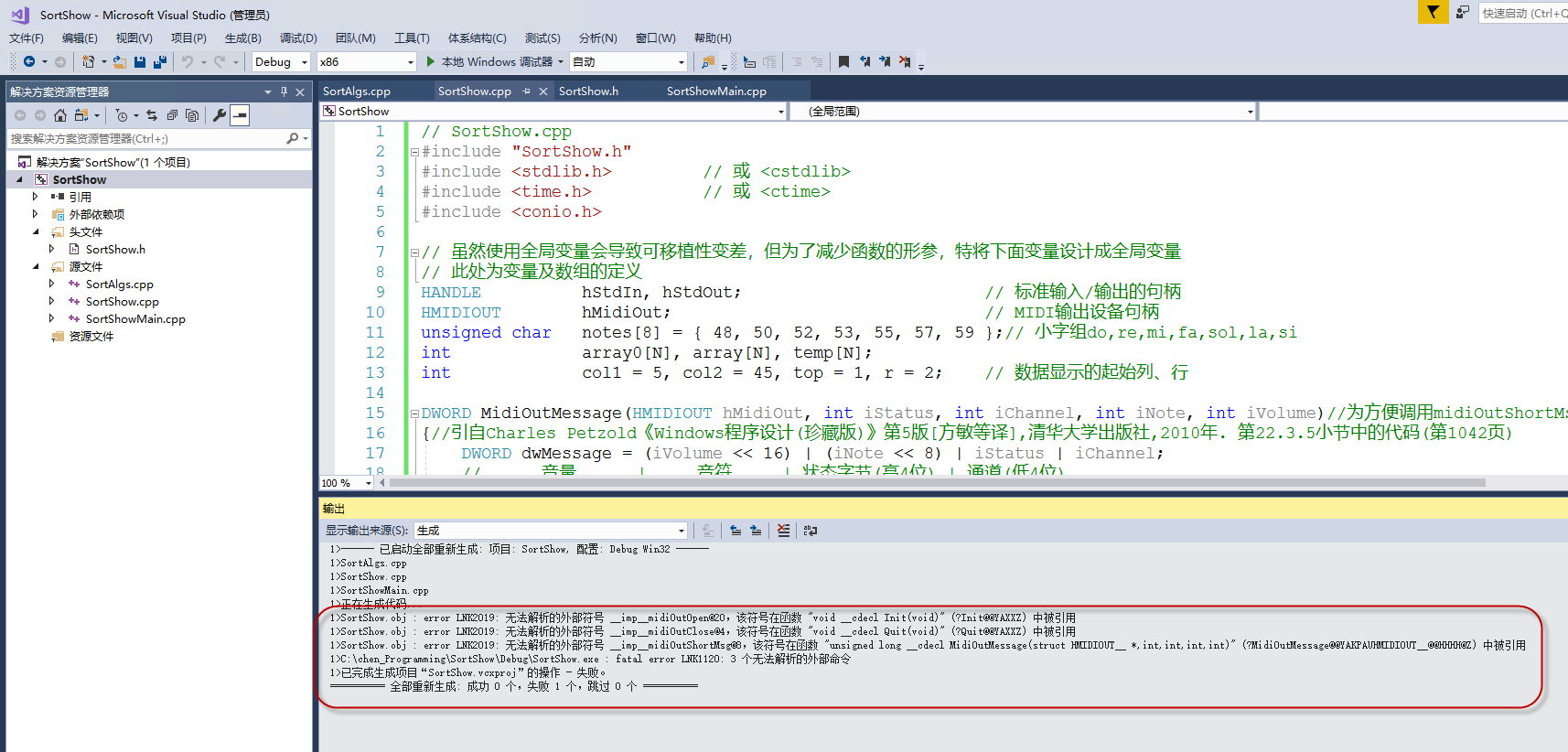


这个目录下，有x86、x64、arm等各个平台，看你自己选择的平台是什么。



chen上面这个要看看包含目录中是否有，否则会DWORD和HANDLE都不能识别。

编译中出现下面的错误提示的话：



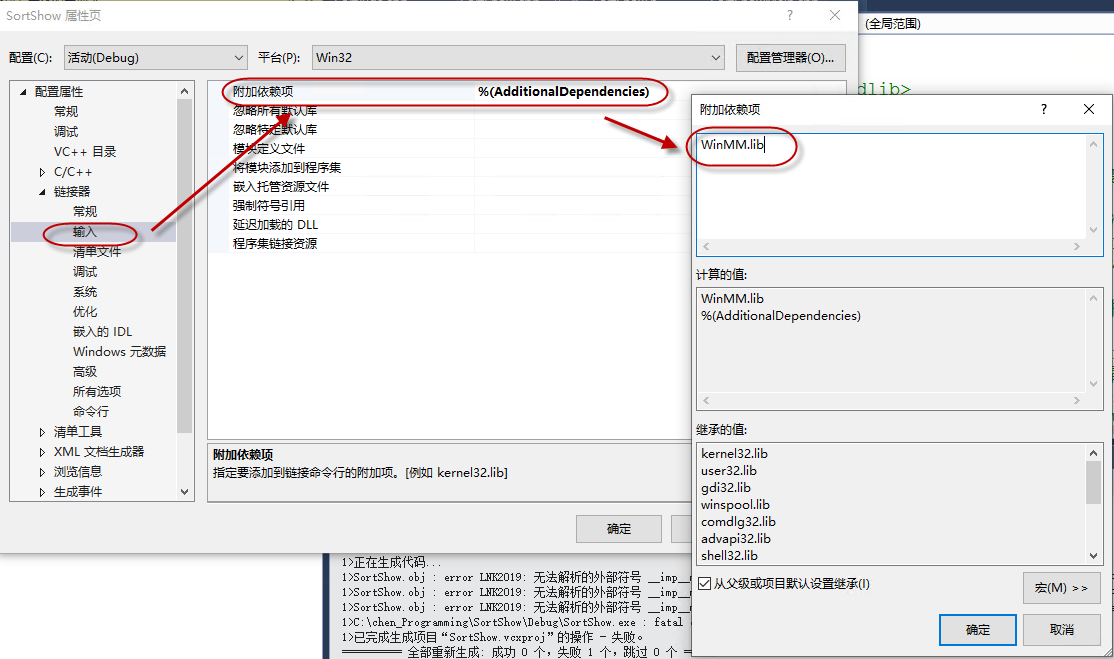
报错信息“无法解析的外部符号 \_\_imp\_\_midiOutOpen@20”表明链接器无法找到midiOutOpen函数的实现。midiOutOpen是Windows多媒体库中用于打开MIDI输出设备的函数。这个错误通常发生在使用了midiOutOpen函数的C或C++程序中，但是没有正确地链接多媒体库（winmm.lib）。

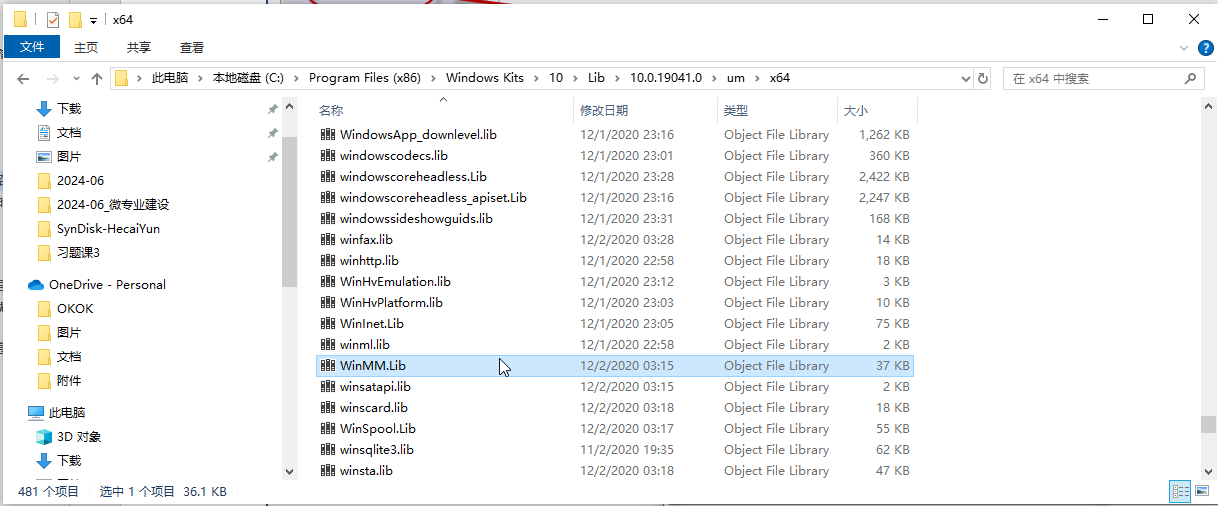
解决方法：

确保你的项目中包含了对winmm.lib的链接。在Visual Studio中，你可以通过以下步骤进行设置：

右击项目 -> 属性 -> 链接器 -> 输入 -> 附加依赖项，然后添加winmm.lib。

或者在代码中使用#pragma comment(lib, "winmm.lib")。





If you can't find winmm.lib in these directories then it would be best to do a repair install of Visual Studio 2017.

If you do find winmm.lib, then the next question would be, did you change any of the paths in Visual Studio. If you create a new, empty windows desktop console project, does:

|  |
| --- |
| #pragma comment(lib, "winmm.lib")  int main()  {  return 0;  } |

<https://www.python100.com/html/94699.html>

Requirements

\*\* Windows NT/2000/XP:\*\* Included in Windows NT 3.1 and later.

\*\* Windows 95/98/Me:\*\* Included in Windows 95 and later.

\*\* Header:\*\* Declared in Mmsystem.h; include Windows.h.

\*\* Library:\*\* Use Winmm.lib.

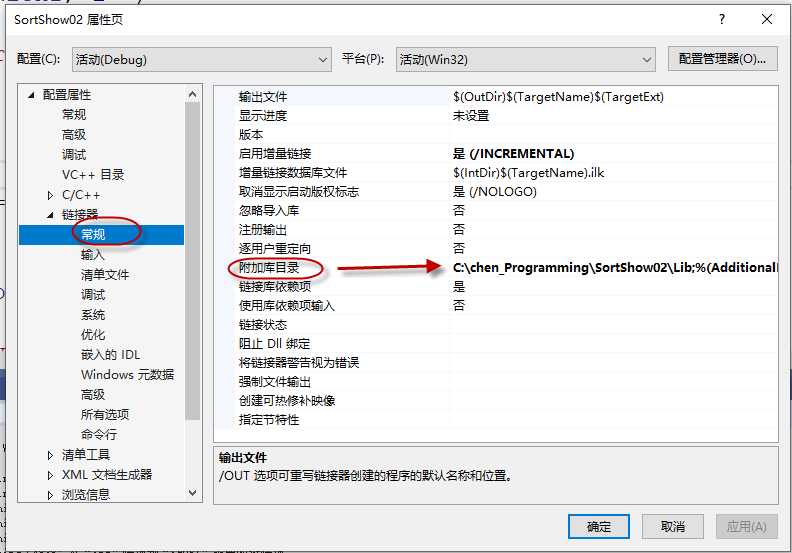
winmm.lib是Windows多媒体API的一个库文件，包含了Windows多媒体服务中的各种功能函数。它可以提供多媒体播放、录音、音视频格式转换和处理等功能。

当我们需要使用Windows多媒体API时，需要在源文件中添加#include <windows.h>和#include <mmsystem.h>头文件，并在链接的时候添加winmm.lib库文件。

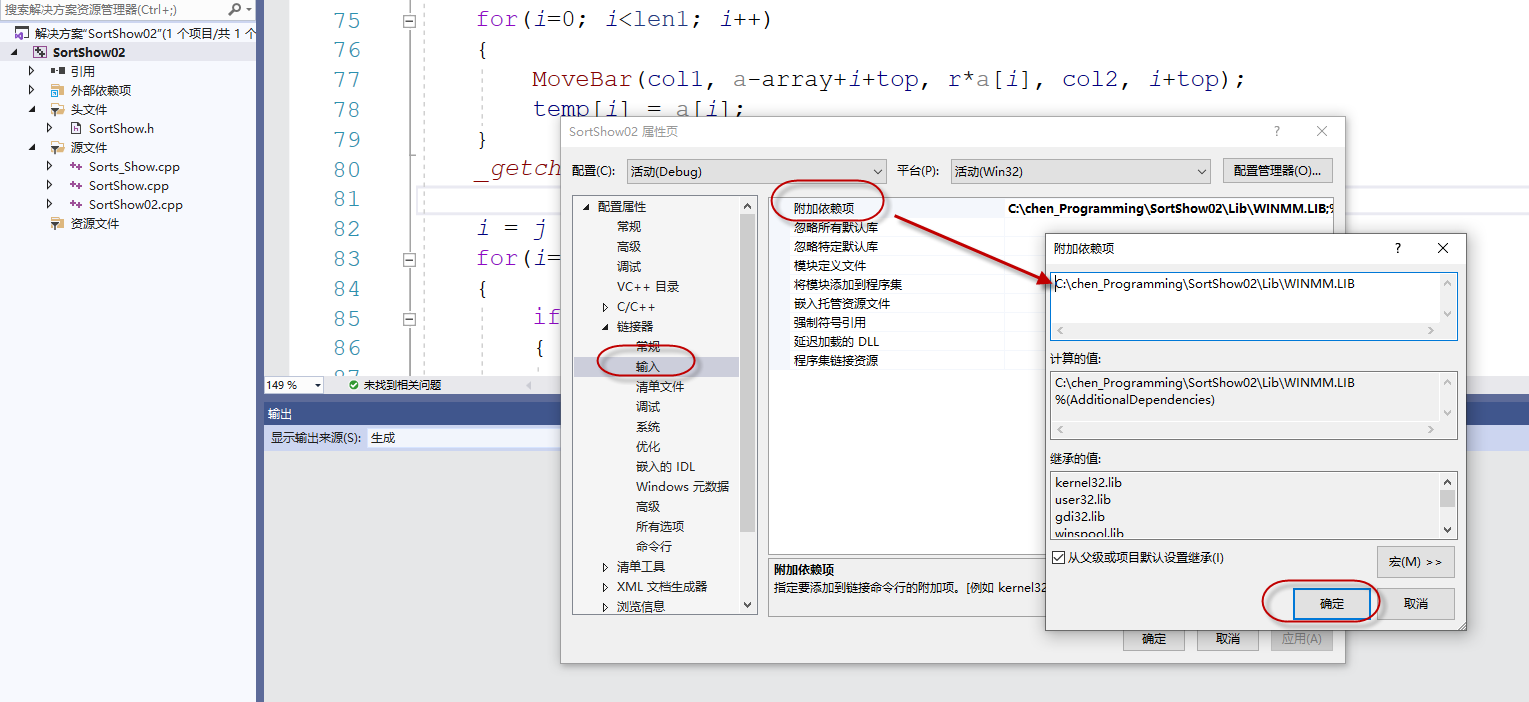
二、winmm.lib找不到

在链接时，如果出现类似“无法打开文件‘winmm.lib’”等错误提示，一般是因为编译器无法找到winmm.lib库文件。此时需要在编译器的链接器设置中添加winmm.lib的路径。具体方法如下：

在Visual Studio 2019中，可通过以下路径进行设置：**项目->属性->链接器->常规->附加库目录**。将winmm.lib所在的文件夹路径添加进去即可。

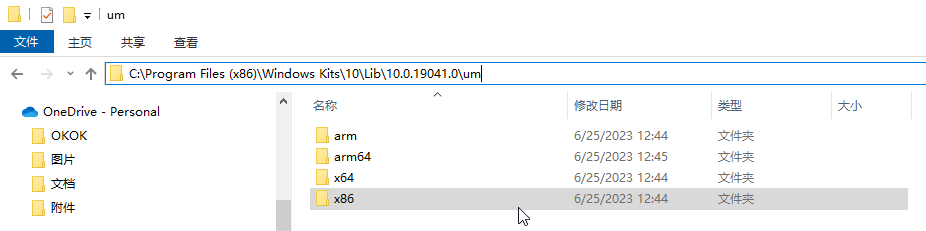


Chen上述添加还是编译通不过。继续：



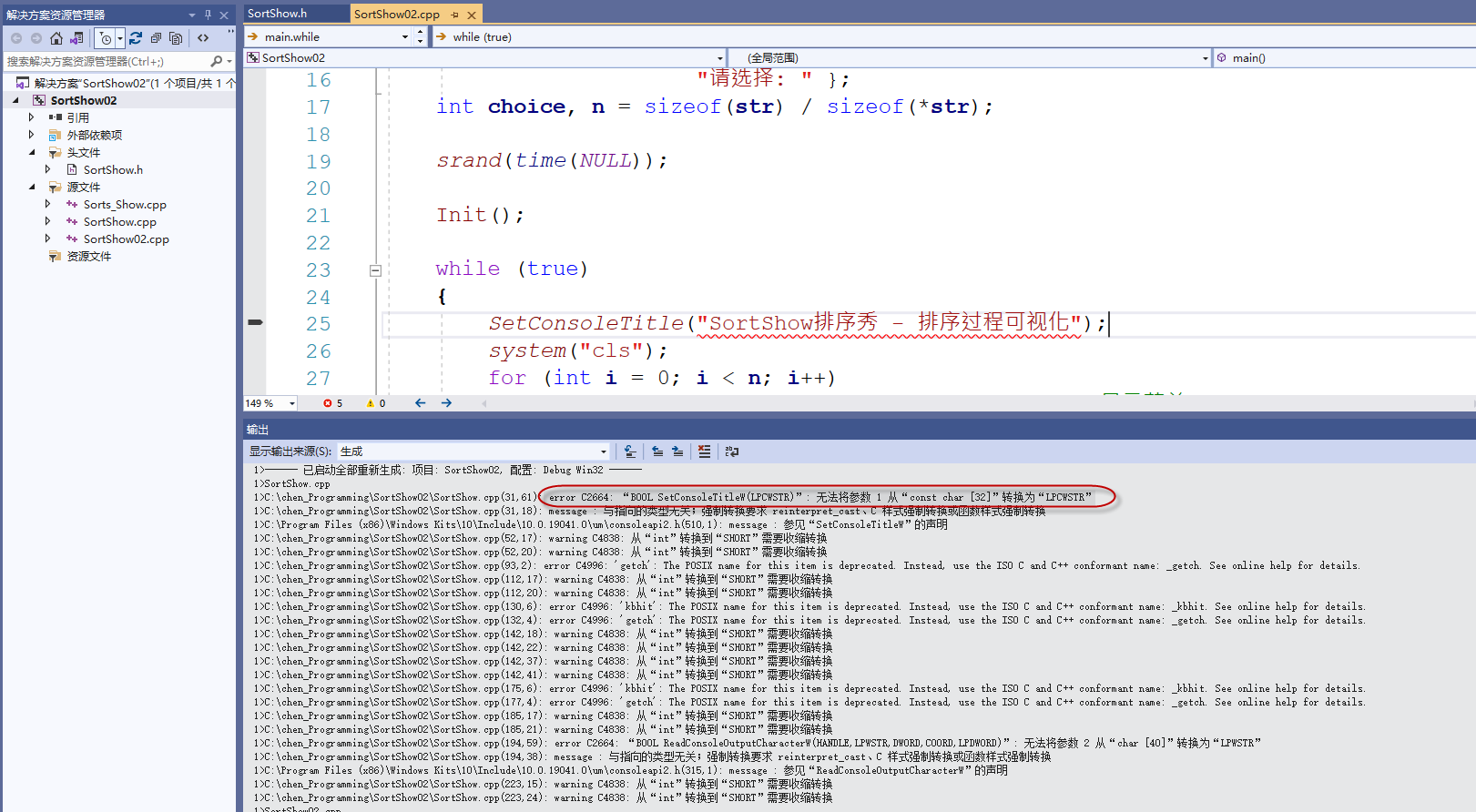
Chen：上面只要放lib文件名就可以了。前面的路径可以不需要。

三、winmm.lib路径

winmm.lib库文件位于Windows系统目录下的System32文件夹中。在链接时，编译器会自动查找该文件（chen：Windows 10里面没有这个lib，所以行不通）事实上，它在这里：

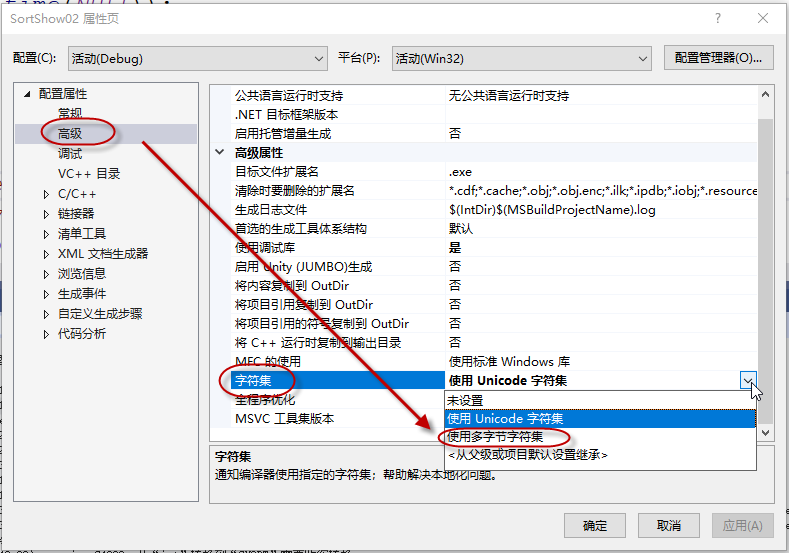
实际上，你也可以从网上找到并下载下来，在IDE添加到依赖项里。

# 四、编译过程中遇到的问题和解决方法：

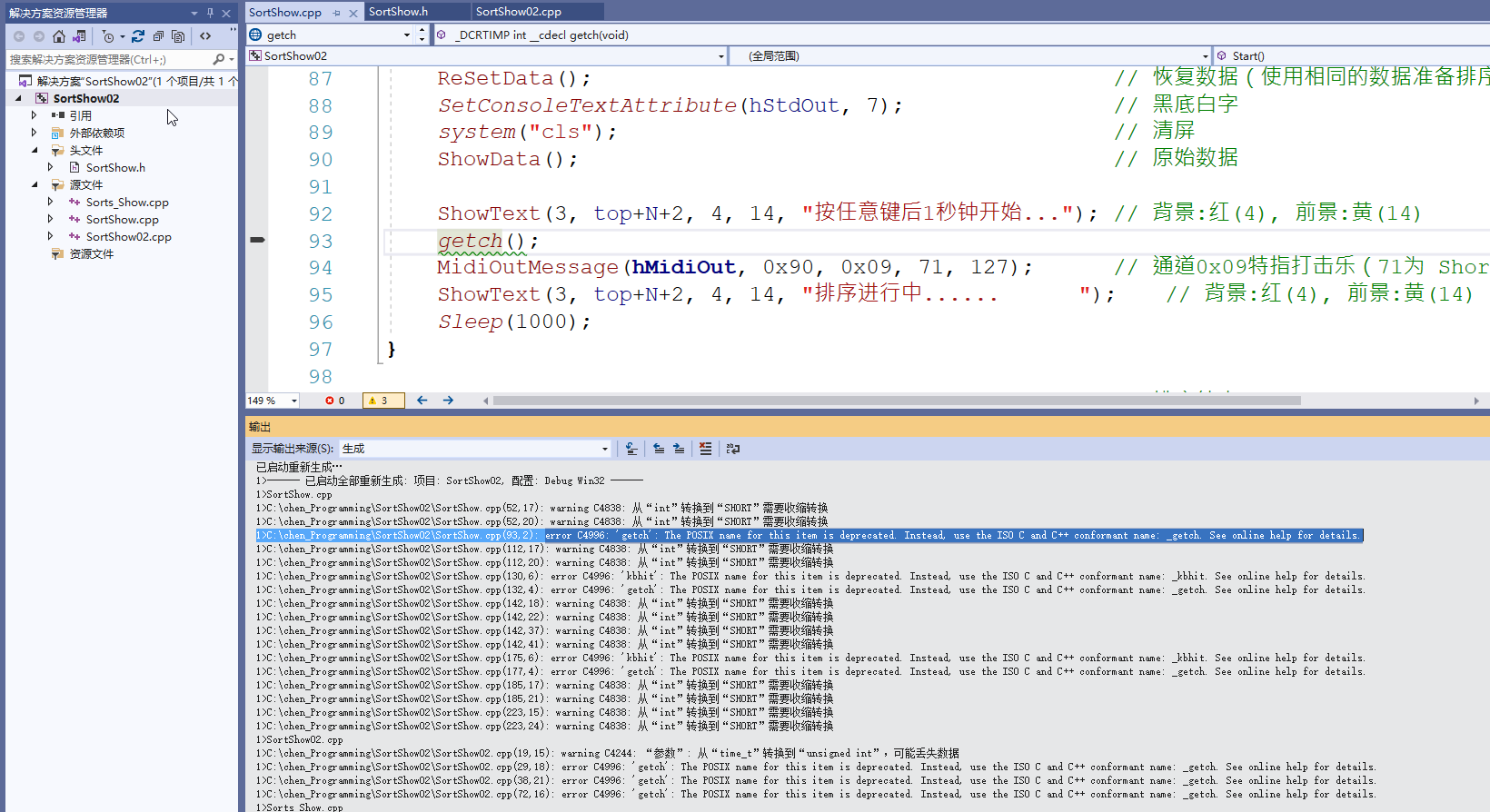


编译的时候，出现“error C2664: “BOOL SetConsoleTitleW(LPCWSTR)”: 无法将参数 1 从“const char [32]”转换为“LPCWSTR”，1>C:\chen\_Programming\SortShow02\SortShow.cpp(31,18): message : 与指向的类型无关；强制转换要求 reinterpret\_cast、C 样式强制转换或函数样式强制转换”

解决方法：如下：



遇到下面的错误提示：

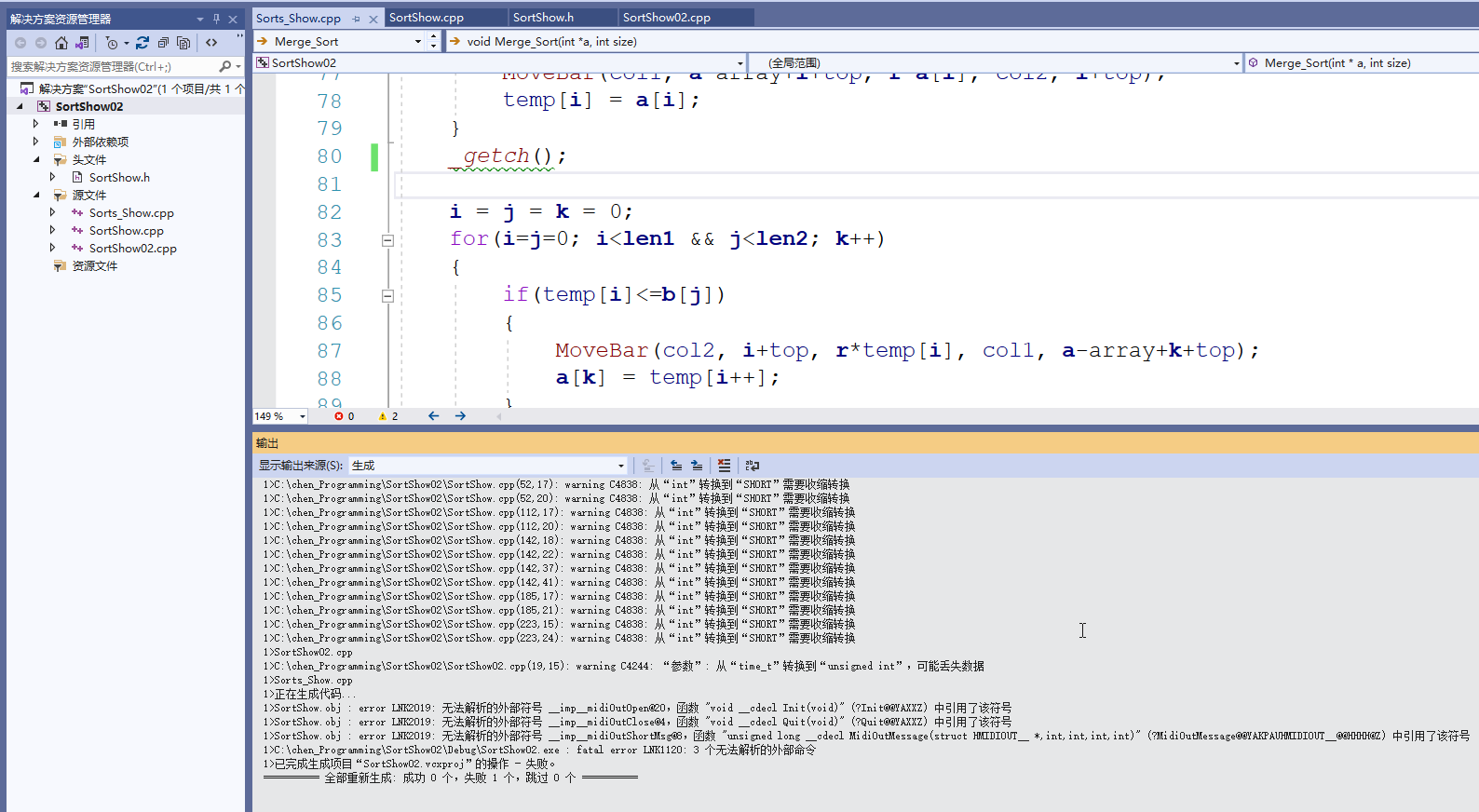


1> error C4996: 'getch': The POSIX name for this item is deprecated. Instead, use the ISO C and C++ conformant name: \_getch. See online help for details.

C:\chen\_Programming\SortShow02\SortShow.cpp(52,20): warning C4838: 从“int”转换到“SHORT”需要收缩转换

解决方法：按错误提示，将这几个方法签名添加一个下划线。比如：\_getch

遇到下面的提示，表示找不到WINMM.LIB文件：



1>正在生成代码...

1>SortShow.obj : error LNK2019: 无法解析的外部符号 \_\_imp\_\_midiOutOpen@20，函数 "void \_\_cdecl Init(void)" (?Init@@YAXXZ) 中引用了该符号

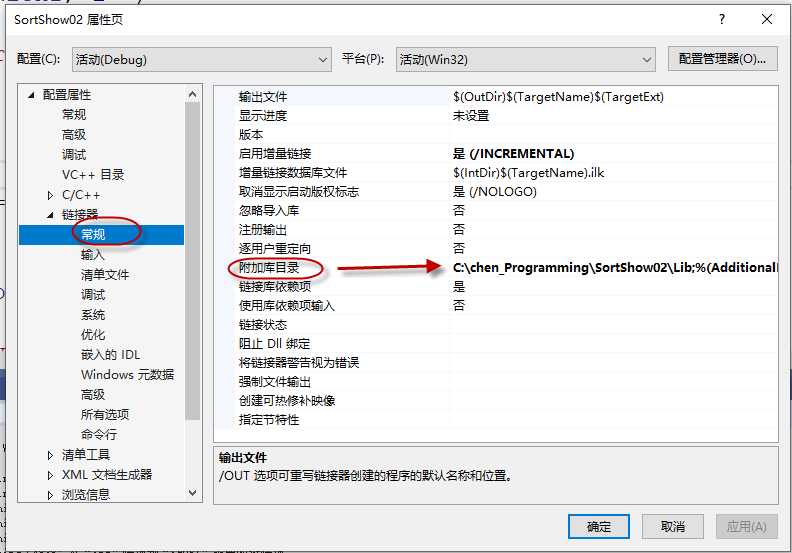
1>SortShow.obj : error LNK2019: 无法解析的外部符号 \_\_imp\_\_midiOutClose@4，函数 "void \_\_cdecl Quit(void)" (?Quit@@YAXXZ) 中引用了该符号

1>SortShow.obj : error LNK2019: 无法解析的外部符号 \_\_imp\_\_midiOutShortMsg@8，函数 "unsigned long \_\_cdecl MidiOutMessage(struct HMIDIOUT\_\_ \*,int,int,int,int)" (?MidiOutMessage@@YAKPAUHMIDIOUT\_\_@@HHHH@Z) 中引用了该符号

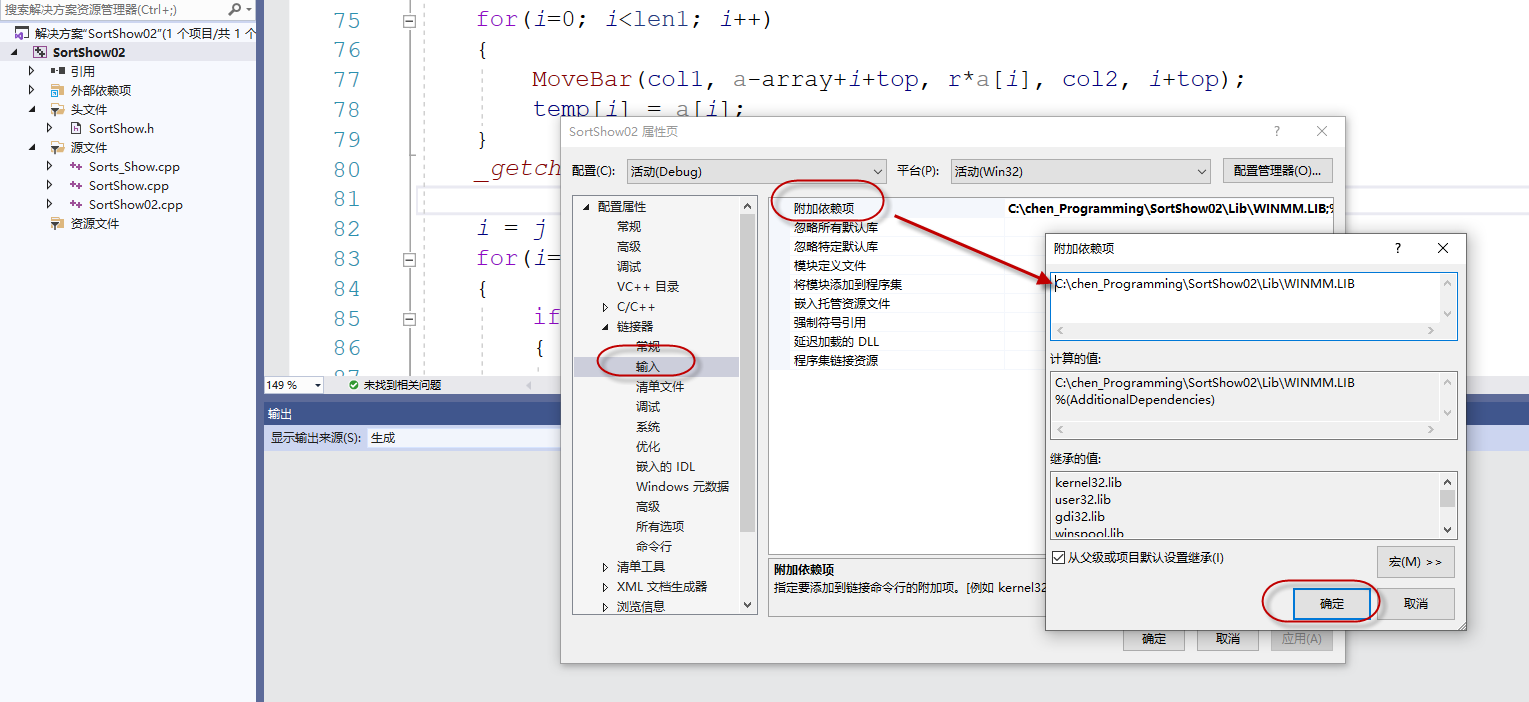
1>C:\chen\_Programming\SortShow02\Debug\SortShow02.exe : fatal error LNK1120: 3 个无法解析的外部命令

解决方法：

在Visual Studio 2019中，可通过以下路径进行设置：**项目->属性->链接器->常规->附加库目录**。将winmm.lib所在的文件夹路径添加进去即可。



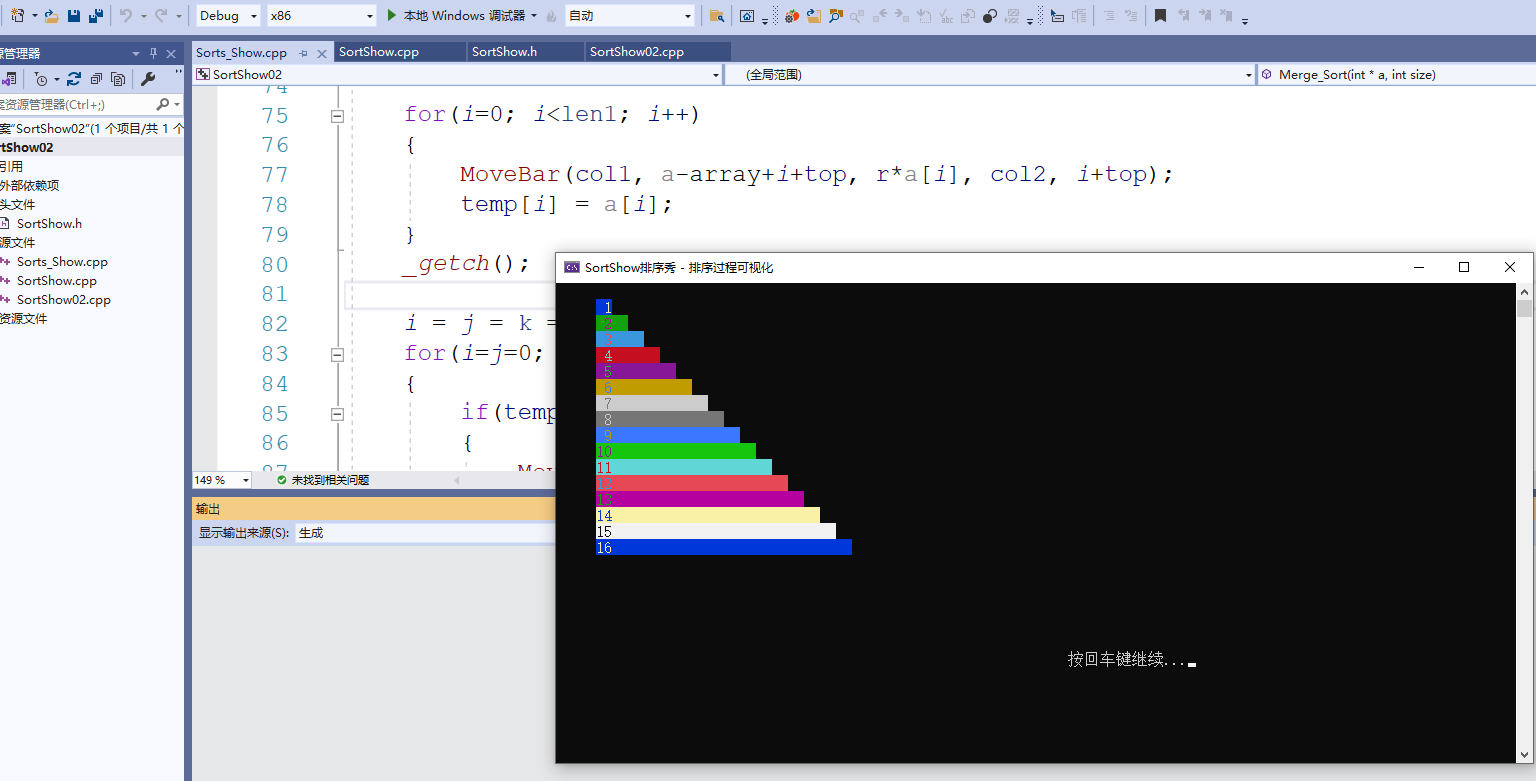
Chen上述添加还是编译通不过。继续：



OK，成功！

[Index of /sourceware/pthreads-win32/prebuilt-dll-2-9-1-release/lib/ | 清华大学开源软件镜像站 | Tsinghua Open Source Mirror](https://mirrors.tuna.tsinghua.edu.cn/sourceware/pthreads-win32/prebuilt-dll-2-9-1-release/lib/)

<https://mirrors.tuna.tsinghua.edu.cn/sourceware/pthreads-win32/prebuilt-dll-2-9-1-release/lib/>



请参见网页<https://docs.microsoft.com/en-us/previous-versions/ms712038(v%3dvs.85)>以了解MIDI函数的详细说明。常用的函数主要有如下3个。

**(1) 打开MIDI输出设备**midiOutOpen

参见<https://docs.microsoft.com/en-us/previous-versions/ms711632(v%3dvs.85)>

**(2) 关闭MIDI输出设备**midiOutClose

参见[https://docs.microsoft.com/en-us/previous-versions/ms711620(v%3dvs.85)](https://docs.microsoft.com/en-us/previous-versions/ms711620%28v%3dvs.85%29)

**(3) 发送消息到MIDI输出设备**midiOutShortMsg

参见<https://docs.microsoft.com/en-us/previous-versions/ms711640(v%3dvs.85)>

MIDI通道消息状态、参数及功能参见附录3。

建议使用参考文献[2]§22.3节“MIDI和音乐”中设计的函数，以方便使用midiOutShortMsg函数。参见下面的函数（这里DWORD为Double WORD，双字。已经有宏定义）**typedef unsigned long** DWORD; // 即32位无符号长整型

**DWORD** MidiOutMessage(HMIDIOUT hMidiOut, **int** iStatus, **int** iChannel, **int** iNote, **int** iVolume)

{ // 音量 | 音符 | 状态 | 通道

DWORD dwMessage = (iVolume<<16) | (iNote<<8) | iStatus | iChannel;

**return** midiOutShortMsg(hMidiOut, dwMessage);

}

# 6. 多媒体计时器

音乐中的时间控制，如延音时长、各种音色演奏时同步等都是十分重要的。有两种方式控制时间。比较简单的是利用Sleep函数延时（参见排序秀程序SortShow）。另一种方式是创建多媒体计时器。

Windows的底层多媒体编程接口（MMAPI）提供了多媒体计时器函数。参见如下网页。

<https://docs.microsoft.com/zh-cn/windows/desktop/Multimedia/multimedia-timer-functions>

请根据如下网页上所提供的样例代码，学习如何使用多媒体计时器。网址及其超链接如下：<https://docs.microsoft.com/zh-cn/windows/desktop/Multimedia/using-multimedia-timers>

|  |
| --- |
| Using Multimedia Timers This section contains examples demonstrating how to perform the following tasks:   * [Obtaining and Setting Timer Resolution](https://docs.microsoft.com/zh-cn/windows/desktop/Multimedia/obtaining-and-setting-timer-resolution) * [Starting a Single Timer Event](https://docs.microsoft.com/zh-cn/windows/desktop/Multimedia/starting-a-single-timer) * [Writing a Timer Callback Function](https://docs.microsoft.com/zh-cn/windows/desktop/Multimedia/writing-a-timer-callback-function) * [Canceling a Timer Event](https://docs.microsoft.com/zh-cn/windows/desktop/Multimedia/canceling-a-timer-event) |

# 附录1. 基本音色表

|  |  |  |
| --- | --- | --- |
| 编号 | 英文名 | 中文名 |
| Piano 钢琴系列 | | |
| 0 | Acoustic Grand Piano | 大钢琴(声学钢琴) |
| 1 | Bright Acoustic Piano | 明亮的钢琴 |
| 2 | Electric Grand Piano | 电钢琴 |
| 3 | Honky-tonk Piano | 酒吧钢琴 |
| 4 | Rhodes Piano | 柔和的电钢琴 |
| 5 | Chorused Piano | 加合唱效果的电钢琴 |
| 6 | Harpsichord | 羽管键琴(拨弦古钢琴) |
| 7 | Clavinet | 科拉维科特琴(击弦古钢琴) |
| Chromatic Percussion 半音阶打击乐 | | |
| 8 | Celesta | 钢片琴 |
| 9 | Glockenspiel | 钟琴 |
| 10 | Music Box | 八音盒 |
| 11 | Vibraphone | 颤音琴 |
| 12 | Marimba | 马林巴(非洲式木琴) |
| 13 | Xylophone | 木琴 |
| 14 | Tubular Bells | 管钟 |
| 15 | Dulcimer | 扬琴 |
| Organ 风琴 | | |
| 16 | Hammond Organ | 拉杆式管风琴 |
| 17 | Percussive Organ | 敲击式管风琴 |
| 18 | Rock Organ | 摇滚管风琴 |
| 19 | Church Organ | 教堂管风琴 |
| 20 | Reed Organ | 簧管风琴 |
| 21 | Accordian | 手风琴 |
| 22 | Harmonica | 口琴 |
| 23 | Tango Accordian | 探戈手风琴 |
| Guitar 吉他 | | |
| 24 | Acoustic Guitar (nylon) | 尼龙弦吉他 |
| 25 | Acoustic Guitar (steel) | 钢弦吉他 |
| 26 | Electric Guitar (jazz) | 爵士电吉他 |
| 27 | Electric Guitar (clean) | 清音电吉他 |
| 28 | Electric Guitar (muted) | 闷音电吉他 |
| 29 | Overdriven Guitar | 加驱动效果的电吉他 |
| 30 | Distortion Guitar | 加失真效果的电吉他 |
| 31 | Guitar Harmonics | 吉他和声 |
| Bass 贝司 | | |
| 32 | Acoustic Bass | 大贝司(声学贝司) |
| 33 | Electric Bass (finger) | 电贝司(指弹) |
| 34 | Electric Bass (pick) | 电贝司(拨片) |
| 35 | Fretless Bass | 无品贝司 |
| 36 | Slap Bass 1 | 击弦贝司1 |
| 37 | Slap Bass 2 | 击弦贝司2 |
| 38 | Synth Bass 1 | 电子合成贝司1 |
| 39 | Synth Bass 2 | 电子合成贝司2 |
| Strings 弦乐 | | |
| 40 | Violin | 小提琴 |
| 41 | Viola | 中提琴 |
| 42 | Cello | 大提琴 |
| 43 | Contrabass | 低音提琴 |
| 44 | Tremolo Strings | 弦乐震音 |
| 45 | Pizzicato Strings | 弦乐拨奏 |
| 46 | Orchestral Harp | 竖琴 |
| 47 | Timpani | 定音鼓 |
| Ensemble 合唱、合奏 | | |
| 48 | String Ensemble 1 | 弦乐合奏1 |
| 49 | String Ensemble 2 | 弦乐合奏2 |
| 50 | Synth Strings 1 | 合成弦乐合奏1 |
| 51 | Synth Strings 2 | 合成弦乐合奏2 |
| 52 | Choir Aahs | 人声合唱“啊” |
| 53 | Voice Oohs | 人声 “嘟” |
| 54 | Synth Voice | 合成人声 |
| 55 | Orchestra Hit | 管弦乐敲击齐奏 |
| Brass 铜管乐器 | | |
| 56 | Trumpet | 小号 |
| 57 | Trombone | 长号 |
| 58 | Tuba | 大号 |
| 59 | Muted Trumpet | 加弱音器小号 |
| 60 | French Horn | 法国号(圆号) |
| 61 | Brass Section | 铜管组(铜管乐器合奏) |
| 62 | Synth Brass 1 | 合成铜管1 |
| 63 | Synth Brass 2 | 合成铜管2 |
| Reed 簧片乐器 | | |
| 64 | Soprano Sax | 高音萨克斯 |
| 65 | Alto Sax | 中音萨克斯 |
| 66 | Tenor Sax | 次中音萨克斯 |
| 67 | Baritone Sax | 上低音萨克斯 |
| 68 | Oboe | 双簧管 |
| 69 | English Horn | 英国管 |
| 70 | Bassoon | 巴松(大管) |
| 71 | Clarinet | 单簧管(黑管) |
| Pipe 吹管乐器 | | |
| 72 | Piccolo | 短笛 |
| 73 | Flute | 长笛 |
| 74 | Recorder | 竖笛 |
| 75 | Pan Flute | 排箫 |
| 76 | Bottle Blow | 瓶木管(吹瓶口) |
| 77 | Shakuhachi | 尺八 |
| 78 | Whistle | 口哨 |
| 79 | Ocarina | 陶笛(洋埙) |
| Synth Lead 合成主音 | | |
| 80 | Lead 1 (square) | 合成主音1(方波) |
| 81 | Lead 2 (sawtooth) | 合成主音2(锯齿波) |
| 82 | Lead 3 (caliope lead) | 合成主音3(汽笛风琴) |
| 83 | Lead 4 (chiff lead) | 合成主音4(吹管) |
| 84 | Lead 5 (charang) | 合成主音5(吉他) |
| 85 | Lead 6 (voice) | 合成主音6(人声) |
| 86 | Lead 7 (fifths) | 合成主音7(平行五度) |
| 87 | Lead 8 (brass+lead) | 合成主音8(贝司加主音) |
| Synth Pad 合成柔音 | | |
| 88 | Pad 1 (new age) | 合成柔音1(新时代) |
| 89 | Pad 2 (warm) | 合成柔音(暖音) |
| 90 | Pad 3 (polysynth) | 合成柔音3(复合成) |
| 91 | Pad 4 (choir) | 合成柔音4(合唱) |
| 92 | Pad 5 (bowed) | 合成柔音5(弓弦) |
| 93 | Pad 6 (metallic) | 合成柔音6(金属声) |
| 94 | Pad 7 (halo) | 合成柔音7(光环) |
| 95 | Pad 8 (sweep) | 合成柔音8(扫弦) |
| Synth Effects 合成特效 | | |
| 96 | FX 1 (rain) | 合成效果1(雨声) |
| 97 | FX 2 (soundtrack) | 合成特效2(音轨) |
| 98 | FX 3 (crystal) | 合成特效3(水晶) |
| 99 | FX 4 (atmosphere) | 合成特效4(大气) |
| 100 | FX 5 (brightness) | 合成效果5(明亮) |
| 101 | FX 6 (goblins) | 合成效果6(哥布林、精灵) |
| 102 | FX 7 (echoes) | 合成特效7(回声) |
| 103 | FX 8 (sci-fi) | 合成特效8(科幻) |
| Ethnic 民族乐器 | | |
| 104 | Sitar | 西塔尔 |
| 105 | Banjo | 班卓 |
| 106 | Shamisen | 三味线 |
| 107 | Koto | 筝 |
| 108 | Kalimba | 卡林巴 |
| 109 | Bagpipe | 风笛 |
| 110 | Fiddle | 古提琴 |
| 111 | Shanai | 唢呐 |
| Percussive 打击乐器 | | |
| 112 | Tinkle Bell | 铃铛 |
| 113 | Agogo | 拉丁打铃 |
| 114 | Steel Drums | 钢鼓 |
| 115 | Woodblock | 木鱼 |
| 116 | Taiko Drum | 太鼓 |
| 117 | Melodic Tom | 嗵鼓 |
| 118 | Synth Drum | 合成鼓 |
| 119 | Reverse Cymbal | 铜钹(反向铙钹) |
| Sound Effects 声音特效 | | |
| 120 | Guitar Fret Noise | 磨弦声 |
| 121 | Breath Noise | 呼吸声 |
| 122 | Seashore | 海浪声 |
| 123 | Bird Tweet | 鸟鸣声 |
| 124 | Telephone Ring | 电话铃声 |
| 125 | Helicopter | 直升机声 |
| 126 | Applause | 鼓掌声 |
| 127 | Gunshot | 枪声 |

# 附录2. 通道9音符编号对应的打击乐

|  |  |  |
| --- | --- | --- |
| 27 | Hi Q | 激光枪声 |
| 28 | Synth Slap | 合成拍音 |
| 29 | Scratch 2 | 高音刷音 |
| 30 | Scratch 1 | 低音刷音 |
| 31 | Sticks | 鼓槌 |
| 32 | Square Click | 敲方板 |
| 33 | Metronome Click | 节拍器 |
| 34 | Metronome Bell | 节拍器重音 |
| 35 | Acoustic Bass Drum | 低音大鼓 |
| 36 | Bass Drum 1 | 高音大鼓 |
| 37 | Side Stick | 鼓边 |
| 38 | Acoustic Snare | 小鼓 |
| 39 | Hand Clap | 拍手声 |
| 40 | Electric Snare | 电子小鼓 |
| 41 | Low Floor Tom | 低音落地嗵鼓 |
| 42 | Closed Hi-Hat | 合音踩镲 |
| 43 | High Floor Tom | 高音落地嗵鼓 |
| 44 | Pedal Hi-Hat | 踏音踩镲 |
| 45 | Low Tom | 低音嗵鼓 |
| 46 | Open Hi-Hat | 开音踩镲 |
| 47 | Low-Mid Tom | 中低音嗵鼓 |
| 48 | Hi-Mid Tom | 中高音嗵鼓 |
| 49 | Crash Cymbal 1 | 低砸音镲 |
| 50 | High Tom | 高音嗵鼓 |
| 51 | Ride Cymbal 1 | 低浮音镲 |
| 52 | Chinese Cymbal | 中国镲 |
| 53 | Ride Bell | 浮音镲碗 |
| 54 | Tambourine | 铃鼓 |
| 55 | Splash Cymbal | 溅音镲 |
| 56 | Cowbell | 牛铃 |
| 57 | Crash Cymbal 2 | 高砸音镲 |
| 58 | Vibraslap | 颤音叉 |
| 59 | Ride Cymbal 2 | 高浮音镲 |
| 60 | Hi Bongo | 高音邦戈 |
| 61 | Low Bongo | 低音邦戈 |
| 62 | Mute Hi Conga | 弱音康加 |
| 63 | Open Hi Conga | 高音康加 |
| 64 | Low Conga | 低音康加 |
| 65 | High Timbale | 高音铜鼓 |
| 66 | Low Timbale | 低音铜鼓 |
| 67 | High Agogo | 高音拉丁打铃 |
| 68 | Low Agogo | 低音拉丁打铃 |
| 69 | Cabasa | 沙锤 |
| 70 | Maracas | 响葫芦 |
| 71 | Short Whistle | 短哨 |
| 72 | Long Whistle | 长哨 |
| 73 | Short Guiro | 短锯琴 |
| 74 | Long Guiro | 长锯琴 |
| 75 | Claves | 击杆 |
| 76 | Hi Wood Block | 高音木块 |
| 77 | Low Wood Block | 低音木块 |
| 78 | Mute Cuica | 弱音吉加 |
| 79 | Open Cuica | 开音吉加 |
| 80 | Mute Triangle | 弱音三角铁 |
| 81 | Open Triangle | 开音三角铁 |
| 82 | Shaker | 沙锤（比69沙锤高） |
| 83 | Jingle Bell | 铃铛 |
| 84 | Bell Tree | 铃树 |
| 85 | Castanets | 响板 |
| 86 | Mute Surdo | 弱音瑟多 |
| 87 | Open Surdo | 开音瑟多 |

# 【注】表中未出现的编号未知。

# 附录3. MIDI通道消息及功能

MIDI通道消息按状态（status）、通道（channel）及参数组成实现一定的功能。主要的MIDI通道消息如下。状态字节占高4位，最高位一定为1，通道号按低4位拼入状态字节中，即：DWORD dwMessage = (iVolume<<16) | (iNote<<8) | iStatus | iChannel;

用二进制数表示MIDI通道消息

**(1) 选择音色（2字节）**

1100cccc 0ppppppp

其中：状态1100（即高4位为0xC=12）表示选择音色（乐器）；cccc表示通道（0000~1111，即0~15）；ppppppp表示音色对应的编号（00000000~01111111，即0~127）。

**(2) 开音码（3字节）**

1001cccc 0nnnnnnn 0vvvvvvv

其中：状态1001（即高4位为0x9=9）表示开始发声；nnnnnnn表示音符（00000000~  
01111111，即0~127）；vvvvvvv表示力度或音量（00000000~01111111，即0~127）。

**(3) 关音码（3字节）**

1000cccc 0nnnnnnn 0vvvvvvv

其中：状态1000（即高4位为0x8=8）表示停止发声；vvvvvvv表示关音的速度（127最快）。

【注】其他控制码参见文献[1]。

# 参考文献

[1] 曹强. 数字音频规范与程序设计：基于Visual C++开发. 北京：中国水利水电出版社，2012年6月（第一版）

[2] Charles Petzold. Windows程序设计(第5版 珍藏版) [方敏等译]. 北京：清华大学出版社，2010年9月（第5版）

[3] 百度百科MIDI [<https://baike.baidu.com/item/MIDI/217824?fr=aladdin>]

[4] MIDI函数[<https://docs.microsoft.com/en-us/previous-versions/ms712038(v%3dvs.85)>]

[5] 多媒体计时器  
[<https://docs.microsoft.com/zh-cn/windows/desktop/Multimedia/multimedia-timer-functions>]