**音频数据格式和数据范围**

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/audio-data-formats-and-data-ranges.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/audio-data-formats-and-data-ranges.md" \o "1个贡献者)

音频驱动程序使用[**KSDATAFORMAT**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ks/ns-ks-ksdataformat)和[**KSDATARANGE**](https://docs.microsoft.com/previous-versions/ff561658(v=vs.85))结构来指定音频流格式：

* KS数据流的数字格式由以KSDATAFORMAT结构开头的KS格式描述符指定。
* KS引脚可以支持的流格式范围由KS数据范围的数组指定；例如，每个数组元素都是一个以KSDATARANGE结构开头的范围描述符。

有关这两种结构的更多信息，请参见[KS数据格式和数据范围](https://docs.microsoft.com/windows-hardware/drivers/stream/ks-data-formats-and-data-ranges)。有关KS数据范围的更多信息，请参见[AVStream中的数据范围交叉点](https://docs.microsoft.com/windows-hardware/drivers/stream/data-range-intersections-in-avstream)。

本节的其余部分讨论以下主题：

[音频数据格式](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/audio-data-formats)

[音频数据范围](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/audio-data-ranges)

[可扩展波形格式描述符](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/extensible-wave-format-descriptors)

[家庭影院系统的多通道格式](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/multichannel-formats-for-home-theater-systems)

[音频数据格式和数据范围的示例](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/examples-of-audio-data-formats-and-data-ranges)

**音频数据格式**

* 2017/02/15
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/audio-data-formats.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/audio-data-formats.md" \o "1个贡献者)

为了指定波形音频流的数据格式，[**KSDATAFORMAT**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ks/ns-ks-ksdataformat)结构后紧接着是[**WAVEFORMATEX**](https://docs.microsoft.com/windows/desktop/api/mmreg/ns-mmreg-twaveformatex)或[**KSDSOUND\_BUFFERDESC**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdsound_bufferdesc)结构，并且[**KSDATAFORMAT**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdsound_bufferdesc)的**Specifier**成员相应地设置为以下两个值之一：

* KSDATAFORMAT\_SPECIFIER\_WAVEFORMATEX

指示数据格式属于waveIn或waveOut应用程序正在使用的wave流。在这种情况下，如果KSDATAFORMAT结构的*FormatSize*足够大，则跟随KSDATAFORMAT结构的数据格式说明符就是WAVEFORMATEX结构。

* KSDATAFORMAT\_SPECIFIER\_DSOUND

指示数据格式属于DirectSound应用程序正在使用的wave流。在这种情况下，遵循KSDATAFORMAT结构的数据格式说明符是KSDSOUND\_BUFFERDESC结构，其中包含嵌入式WAVEFORMATEX结构。

该[**KSDATAFORMAT\_WAVEFORMATEX**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdataformat_waveformatex)结构封装了一个KSDATAFORMAT结构和它后面的WAVEFORMATEX结构。同样，[**KSDATAFORMAT\_DSOUND**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdataformat_dsound)结构既封装了KSDATAFORMAT结构，又封装了其后的DSOUND\_BUFFERDESC结构。

对于KSDATAFORMAT\_WAVEFORMATEX或KSDATAFORMAT\_DSOUND，结构中的最后一项是嵌入式WAVEFORMATEX结构。对于KSDATAFORMAT\_DSOUND，WAVEFORMATEX结构包含在嵌入式DSOUND\_BUFFERDESC结构中。在任一情况下，该结构WAVEFORMATEX可能是一个的开始[**WAVEFORMATEXTENSIBLE**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-waveformatextensible)结构，在这种情况下**wFormatTag** WAVEFORMATEX的构件被设置为值WAVE\_FORMAT\_EXTENSIBLE。有关更多信息，请参见[可扩展波形格式描述符](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/extensible-wave-format-descriptors)。

要指定MIDI流或DirectMusic流的数据格式，KSDATAFORMAT结构就足够了。它后面没有任何其他信息。

有关wave和DirectSound数据格式的示例，请参见[PCM流数据格式](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/pcm-stream-data-format)和[DirectSound流数据格式](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/directsound-stream-data-format)。有关MIDI和DirectMusic数据格式的示例，请参见[MIDI流数据格式](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/midi-stream-data-format)和[DirectMusic流数据格式](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/directmusic-stream-data-format)。

**音频数据范围**

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/audio-data-ranges.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/audio-data-ranges.md" \o "1个贡献者)

KS过滤器上的每个引脚都声明其支持的数据格式。引脚工厂将这些信息公开为一系列数据范围。与前面描述的格式描述符不同，数据范围描述了一系列数据格式。例如，波形引脚的数据范围指定了该引脚支持的样本大小，频率和通道的范围。

当微型端口驱动程序实例化一个引脚时，它将引脚配置为处理具有从引脚数据范围中选择的特定数据格式的流。这项工作由微型端口驱动程序的数据交叉处理程序完成，该处理程序选择两个引脚共有的音频数据格式，以便可以连接它们。有关更多信息，请参见[数据交叉处理程序](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/data-intersection-handlers)。

有关使用属性请求查询音频引脚的数据范围和选择数据交集的信息，请参见[引脚数据范围和交集属性](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/pin-data-range-and-intersection-properties)。

要指定波形引脚的数据范围，请在[**KSDATARANGE**](https://docs.microsoft.com/previous-versions/ff561658(v=vs.85))结构后跟随描述该引脚支持的样本大小，频率和通道范围的信息。该信息（包括KSDATARANGE结构本身）被封装在[**KSDATARANGE\_AUDIO**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdatarange_audio)结构中。

要为MIDI或DirectMusic引脚指定数据范围，请在KSDATARANGE结构后跟随其他信息，包括可以同时播放的最大通道数和音符。此信息与KSDATARANGE结构本身一起封装在[**KSDATARANGE\_MUSIC**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdatarange_music)结构中。

本文档提供了使用KSDATARANGE\_AUDIO和KSDATARANGE\_MUSIC结构的数据范围的几个示例：

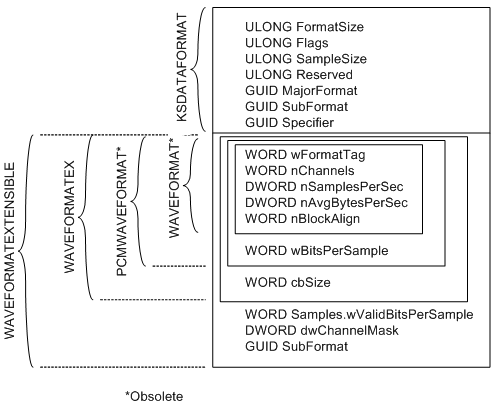
* 有关wave和DirectSound数据范围的声明的示例，请参见[PCM流数据范围](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/pcm-stream-data-range)和[DirectSound流数据范围](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/directsound-stream-data-range)。
* 有关MIDI和DirectMusic数据范围的声明的示例，请参见[MIDI流数据范围](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/midi-stream-data-range)和[DirectMusic流数据范围](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/directmusic-stream-data-range)。
* 有关非PCM格式的数据范围声明的示例，请参阅“ [指定AC-3数据范围”](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/specifying-ac-3-data-ranges)和“ [指定WMA Pro数据范围”](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/specifying-wma-pro-data-ranges)。

**可扩展波形格式描述符**

* 2017年10月27日
* 7分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/extensible-wave-format-descriptors.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/extensible-wave-format-descriptors.md" \o "2位贡献者)

* + [[https://github.com/EliotSeattle.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/extensible-wave-format-descriptors.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/extensible-wave-format-descriptors.md" \o "2位贡献者)

下图显示了波形音频流的数据格式描述符。



如图所示，[**KSDATAFORMAT**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ks/ns-ks-ksdataformat)结构后面的附加格式信息量随数据格式而变化。

音频系统以几种方式使用这种类型的格式描述符：

* 如上图所示的格式描述符作为调用参数传递给微型端口驱动程序的**NewStream**方法（例如，请参见[**IMiniportWaveCyclic :: NewStream**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-iminiportwavecyclic-newstream)）。
* [**IMiniport :: DataRangeIntersection**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/portcls/nf-portcls-iminiport-datarangeintersection)方法的*ResultantFormat*参数指向一个缓冲区，该方法在该缓冲区中写入格式描述符，如上图所示。
* 所述[**KSPROPERTY\_PIN\_DATAINTERSECTION**](https://docs.microsoft.com/windows-hardware/drivers/stream/ksproperty-pin-dataintersection)获得属性请求检索的格式描述符等上图中所示。
* 所述[**KSPROPERTY\_PIN\_PROPOSEDATAFORMAT**](https://docs.microsoft.com/windows-hardware/drivers/stream/ksproperty-pin-proposedataformat)设置属性请求接受的格式描述符等上图中所示。
* [**KsCreatePin**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ks/nf-ks-kscreatepin)函数的*Connect*调用参数使用类似的格式。此参数指向也包含格式描述符的缓冲区开头的[**KSPIN\_CONNECT**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ks/ns-ks-kspin_connect)结构。紧随KSPIN\_CONNECT结构的格式描述符以KSDATAFORMAT结构开始，如上图所示。

遵循KSDATAFORMAT结构的格式信息应该是[**WAVEFORMATEXTENSIBLE**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-waveformatextensible)结构或[**WAVEFORMATEX**](https://docs.microsoft.com/windows/desktop/api/mmreg/ns-mmreg-twaveformatex)结构。WAVEFORMATEXTENSIBLE是WAVEFORMATEX的扩展版本，它可以描述比WAVEFORMATEX更广泛的格式。WAVEFORMATEX是WDM之前的WAVEFORMAT结构的扩展版本。WAVEFORMAT已过时，并且在任何版本的Microsoft Windows中，WDM音频子系统均不支持WAVEFORMAT。

同样，PCMWAVEFORMAT结构是WAVEFORMAT的扩展版本，已过时，但WDM音频子系统为此提供了有限的支持。

有关WAVEFORMAT和PCMWAVEFORMAT的信息，请参阅Microsoft Windows SDK文档。

四个波形格式结构-WAVEFORMAT，PCMWAVEFORMAT，WAVEFORMATEX和WAVEFORMATEXTENSIBLE-都以相同的五个成员开头，以**wFormatTag开头**。上图显示了这四个结构彼此叠加，以突出显示结构中相同的部分。PCMWAVEFORMAT和WAVEFORMATEX通过添加**wBitsPerSample**成员来扩展WAVEFORMAT ，但是WAVEFORMATEX也添加了**cbSize**成员。WAVEFORMATEXTENSIBLE通过添加三个成员（从**Samples** .wValidBitsPerSample 开始）扩展了WAVEFORMATEX 。（**Samples**是一个联合，其联盟成员**wValidSamplesPerBlock**代替了**wValidBitsPerSample**对于某些压缩格式）的**wFormatTag**构件，其紧跟在所述缓冲器中的KSDATAFORMAT结构的端部，指定什么样的格式信息如下KSDATAFORMAT。该[KMixer系统驱动程序](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/kernel-mode-wdm-audio-components#kmixer_system_driver)支持使用下表中所示的三种格式标签中的一个仅PCM格式。

| **wFormatTag值** | **含义** |
| --- | --- |
| WAVE\_FORMAT\_PCM | 由WAVEFORMATEX或PCMWAVEFORMAT指定的整数PCM数据格式。 |
| WAVE\_FORMAT\_IEEE\_FLOAT | WAVEFORMATEX指定的浮点PCM数据格式。 |
| WAVE\_FORMAT\_EXTENSIBLE | 由WAVEFORMATEXTENSIBLE指定的扩展数据格式。 |

实际上，KMixer仅支持可由这些标记值描述的PCM格式的子集（并且不支持非PCM格式）。USB音频设备（请参阅[USBAudio类系统驱动程序](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/kernel-mode-wdm-audio-components#usbaudio_class_system_driver)）仅限于此子集，因为所有PCM格式的USB音频流都通过KMixer传递。（某些非PCM USB音频流可以绕过KMixer；有关更多信息，请参见[USB音频对非PCM格式的支持](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/usb-audio-support-for-non-pcm-formats)。）但是，在Windows XP和更早的版本中，DirectSound应用程序可以通过直接连接到WaveCyclic上的硬件引脚来克服KMixer的限制。支持KMixer不支持的格式的WavePci设备。有关更多信息，请参见[WDM Audio中的DirectSound硬件加速](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/directsound-hardware-acceleration-in-wdm-audio)。

请注意上表中WAVE\_FORMAT\_PCM标记值含义的歧义-它可以指定WAVEFORMATEX或PCMWAVEFORMAT结构。但是，这两个结构几乎相同。唯一的区别是WAVEFORMATEX包含**cbSize**成员，而PCMWAVEFORMAT不包含。根据WAVEFORMATEX说明书中，**CBSIZE**是如果忽视**wFormatTag** = WAVE\_FORMAT\_PCM（因为**CBSIZE**是隐含零）; **cbSize**用于所有其他格式。因此，在PCM格式的情况下，PCMWAVEFORMAT和WAVEFORMATEX包含相同的信息，并且可以被相同地对待。

WAVEFORMATEX只能指定WAVEFORMATEXTENSIBLE可以指定的格式的子集。与WAVEFORMATEX不同，WAVEFORMATEXTENSIBLE可以执行以下操作：

1. 指定每个样本的位数，与样本容器的大小分开。例如，可以将一个20位的样本左对齐存储在一个三字节的容器中。WAVEFORMATEX无法将每个样本的数据位数与样本容器的大小区分开，因此无法明确描述这种格式。
2. 将特定的扬声器位置分配给多声道流中的音频声道。WAVEFORMATEX缺乏此功能，只能充分支持单声道和（两声道）立体声流。

由WAVEFORMATEX描述的任何格式也可以由WAVEFORMATEXTENSIBLE描述。有关将WAVEFORMATEX结构转换为WAVEFORMATEXTENSIBLE的信息，请参见[在格式标记和子格式GUID之间转换](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/converting-between-format-tags-and-subformat-guids)。

WAVEFORMATEX足以描述样本大小为8或16位的格式，但是WAVEFORMATEXTENSIBLE对于充分描述样本精度大于16位的格式是必需的。这是两个示例：

* 具有24位采样精度的流可以使用32位容器大小来进行有效处理，但是可以转换为使用24位容器来提高存储效率而不会丢失数据。
* 当使用24位样本数据处理流时，仅提供20位精度的渲染设备可以使用抖动来提高其输出信号的保真度。但是，抖动处理需要额外的处理时间，并且如果原始流的准确度仅为20位，则不需要额外的处理。

在这两个示例中，只有在样品精度和容器尺寸均已知的情况下，才能在处理和存储效率之间进行正确权衡的同时保持信号质量。

如果可以通过WAVEFORMATEX或WAVEFORMATEXTENSIBLE结构明确描述简单格式，则音频驱动程序可以选择选择任一结构来描述该格式。但是，音频驱动程序通常使用WAVEFORMATEX来指定具有8位或16位样本的单声道和（两通道）立体声PCM格式，并且某些较早的应用程序可能希望所有音频驱动程序都使用WAVEFORMATEX来指定这些格式。

如果驱动程序支持可以明确指定为WAVEFORMATEX或WAVEFORMATEXTENSIBLE结构的音频格式，则驱动程序应识别该格式，而不管客户端应用程序或组件用于指定结构的两种结构中的哪一种。例如，如果音频设备支持44.1kHz，16位立体声PCM格式，则微型端口驱动程序的KSPROPERTY\_PIN\_PROPOSEDATAFORMAT属性处理程序及其**NewStream**方法的实现应接受该格式，而不管该格式是指定为WAVEFORMATEX还是指定为WAVEFORMATEXTENSIBLE结构。

为了简化格式数据的处理，驱动程序通常使用WAVEFORMATEXTENSIBLE结构在内部表示格式。此方法可能需要将输入WAVEFORMATEX结构转换为内部WAVEFORMATEXTENSIBLE表示形式，或将内部WAVEFORMATEXTENSIBLE表示转换为输出WAVEFORMATEX结构。

将格式描述符从WAVEFORMATEX转换为WAVEFORMATEXTENSIBLE时，如果WAVEFORMATEX结构的**wFormatTag**成员是WAVE\_FORMAT\_PCM或WAVE\_FORMAT\_IEEE\_FLOAT，请将WAVEFORMATEXTENSIBLE结构的**dwChannelMask**成员设置为SPEAKER\_FRONT\_CENTER（对于单流| SPEAKER\_FRONT\_RIGHT（用于立体声流）。SPEAKER\_FRONT\_ *XXX*常量在头文件Ksmedia.h中定义。

在除Windows 98“ Gold”之外的所有Windows版本中，KMixer支持一系列WAVEFORMATEXTENSIBLE PCM格式，这些格式具有多个通道，每个样本最多32位。

KMixer支持的WAVEFORMATEX PCM格式的子集在Windows版本之间有所不同，如下表所示。

| **Windows版本** | **包装样品量** | **通道数** |
| --- | --- | --- |
| Windows 98“金” | 8、16、24和32位 | 多渠道 |
| Windows 98 SE | 仅8位和16位 | 仅单声道和立体声 |
| Windows 98 SE +修补程序 | 8、16、24和32位 | 仅单声道和立体声 |
| Windows 2000 | 仅8位和16位 | 仅单声道和立体声 |
| Windows Me | 8、16、24和32位 | 仅单声道和立体声 |
| Windows XP及更高版本 | 仅8位和16位 | 仅单声道和立体声 |

在WAVEFORMATEXTENSIBLE中，**dwBitsPerSample**是容器大小，而**wValidBitsPerSample**是每个样本的有效数据位数。容器始终在内存中按字节对齐，并且容器大小必须指定为八位的倍数。

在定义WAVEFORMATEXTENSIBLE结构之前，供应商必须向Microsoft注册每种新的wave格式，以便可以为该格式分配一个官方的16位格式标签。（格式标记包含在WAVEFORMATEX结构的**wFormatTag**成员中。）已注册的格式标记列表出现在公共头文件Mmreg.h中（例如WAVE\_FORMAT\_MPEG）。

使用WAVEFORMATEXTENSIBLE，不再需要注册格式。供应商可以根据需要将GUID独立分配给其新格式。（格式GUID包含在WAVEFORMATEXTENSIBLE 的**SubFormat**成员中。）但是，Microsoft在公共头文件Ksmedia.h中列出了一些更流行的格式GUID（例如KSDATAFORMAT\_SUBTYPE\_MPEG）。在定义新格式的GUID之前，供应商应检查Ksmedia.h中的KSDATAFORMAT\_SUBTYPE\_ *XXX*常量列表，以确定是否已为特定格式定义了适当的GUID。

使用WAVEFORMATEXTENSIBLE时，请将**wFormatTag**设置为WAVE\_FORMAT\_EXTENSIBLE，并将**SubFormat**设置为适当的格式GUID。对于整数PCM格式，请将**SubFormat**设置为KSDATAFORMAT\_SUBTYPE\_PCM。对于将样本值编码为浮点数的PCM格式，请将**SubFormat**设置为KSDATAFORMAT\_SUBTYPE\_IEEE\_FLOAT。对于这两种格式，请将**cbSize**设置为**sizeof**（WAVEFORMATEXTENSIBLE）**-sizeof**（WAVEFORMATEX）。有关使用WAVEFORMATEXTENSIBLE来描述非PCM数据格式的信息，请参阅[支持非PCM波形格式](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/supporting-non-pcm-wave-formats)。

**家庭影院系统的多通道格式**

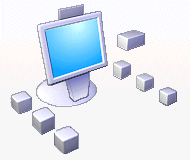
* 2017/04/20
* 4分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/multichannel-formats-for-home-theater-systems.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/multichannel-formats-for-home-theater-systems.md" \o "1个贡献者)

家庭影院系统的一种相对便宜的解决方案是将一组环绕立体声扬声器连接到运行Windows操作系统的计算机上的音频适配器。或者，用户可以在适配器输出和扬声器之间连接外部音频/视频（A / V）接收器。为了响应家庭影院系统的普及，已经为这些系统创作的5.1声道和7.1声道音频内容变得越来越可用。

为了在家庭影院系统上准确呈现多声道音频内容，需要一种音频格式描述符，该描述符可以将扬声器位置分配给多声道流中的音频通道。如前所述，[**WAVEFORMATEXTENSIBLE**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-waveformatextensible)结构可以指定此类扬声器分配。

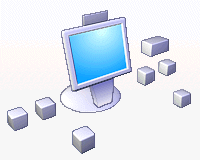
为了帮助为家庭影院系统提供音频驱动程序支持，Microsoft为Microsoft Windows XP和更高版本定义了新的7.1声道扬声器配置。Windows Vista，带有Service Pack 2（SP2）的Windows XP和带有Service Pack 1（SP1）的Windows Server 2003支持此配置。未安装Service Pack的Windows Server 2003，未安装Service Pack 1的Windows XP或未安装Service Pack的Windows XP不支持此功能。

下图显示了新的7.1声道扬声器配置，该配置取自带有SP2的Windows XP中的Windows多媒体控制面板（Mmsys.cpl）。



Windows多媒体控制面板将名称“ 7.1家庭影院扬声器”分配给上图所示的新7.1声道宽扬声器配置。

下图显示了较旧的7.1通道配置，Windows 2000和更高版本以及Windows Me / 98支持该配置。



在带有SP2的Windows XP中，多媒体控制面板将名称“ 7.1宽配置扬声器”分配给上图所示的旧配置。

在带有SP2的Windows XP中，可以按照以下步骤找到前两个图中所示的两种配置：

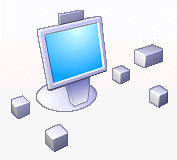
1. 在控制面板（类别视图）中，单击**声音，语音和音频设备**。
2. 在“ **声音，语音和音频设备”**窗格中的“ **选择任务”下**，单击“ **调整系统音量”**。
3. 在“ **声音和音频设备属性”**属性页的“ **音量”**选项卡上的“ **扬声器设置”下**，单击“ **高级”**。
4. 在“ **高级音频属性”**属性页的“ **扬声器设置”下**，打开下拉列表，然后选择两个7.1扬声器配置之一。

7.1宽配置扬声器图中显示的配置是Sony动态数字声音（SDDS）配置，该配置于1993年推出，用于电影院。但是，很少有家庭影院系统使用此配置。相反，8扬声器家庭影院系统可能会使用7.1家庭影院扬声器图中所示的新7.1配置。此外，已为SDDS配置编写了最少的家庭影院内容，并且用户可以期望将大多数可用的7.1声道内容格式化为新的7.1配置。

尽管新的“ 7.1家庭影院扬声器”配置在很大程度上取代了旧的“ 7.1宽配置扬声器”配置，但Windows将继续支持旧配置，以提供向后兼容性。

在Windows 2000和更高版本以及Windows Me / 98中，5.1声道扬声器配置分别将声道5和6分配给左后扬声器和右后扬声器。在Windows Vista，带有SP1的Windows Server 2003和带有SP2的Windows XP中，5.1通道配置保持不变。与7.1声道配置相反，这些Windows版本未定义新的5.1格式描述符，以区分5.1侧置扬声器配置和5.1后置扬声器配置。因为这两个配置是如此相似，所以定义两个5.1配置可能会导致用户在使用哪个配置以及是否播放在另一个配置上为一个配置创作的内容之间产生混淆。将扬声器放置在6扬声器家庭影院系统中时，用户倾向于不区分侧面扬声器和背面扬声器的位置。取而代之的是，扬声器的位置更有可能由房间的形状和家具在房间中的放置来确定。

下图显示了5.1声道环绕立体声扬声器的配置，该配置取自Windows XP SP2的Windows多媒体控制面板。



Windows多媒体控制面板将名称“ 5.1环绕声扬声器”分配给上图所示的5.1声道扬声器配置。

尽管5.1声道侧扬声器和后扬声器配置之间的差异可能对用户是透明的，但对于音频硬件供应商却不透明。如前所述，通常为辅助扬声器而不是后置扬声器创作5.1声道内容。因此，在通过“ 7.1家庭影院扬声器”配置播放5.1声道内容时，供应商应确保5.1声道流中的两个侧面扬声器声道通过侧面扬声器而不是后扬声器播放。类似地，当通过带有侧面扬声器的5.1扬声器配置播放针对“ 7.1家庭影院扬声器”配置创作的内容时，7.1侧面扬声器的通道最自然地映射到5.1配置中的侧面扬声器。

本节包括：

[通道遮罩](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/channel-mask)

[将流格式映射到扬声器配置](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/mapping-stream-formats-to-speaker-configurations)

[头文件更改](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/header-file-changes)

**通道遮罩**

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/channel-mask.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/channel-mask.md" \o "1个贡献者)

在Windows中，[**WAVEFORMATEXTENSIBLE**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-waveformatextensible)结构定义了多通道PCM音频流的数据格式。此结构指定参数，例如每个PCM样本的位数，流中的通道数以及通道掩码。通道掩码指定通道到扬声器的映射。下图显示了通道掩码中的各个位。

该图说明了通道掩码中的各个位

通道掩码中的每一位代表一个特定的扬声器位置。如果掩码将通道分配给特定的扬声器位置，则代表该位置的掩码位将设置为1；否则，该位将被设置为1。未分配扬声器位置的所有掩码位都设置为0。WAVEFORMATEXTENSIBLE结构定义了通道掩码中的其他位（未在上图中显示），但是这些位与正在讨论的家庭影院扬声器配置无关，因此在此处省略了简单。

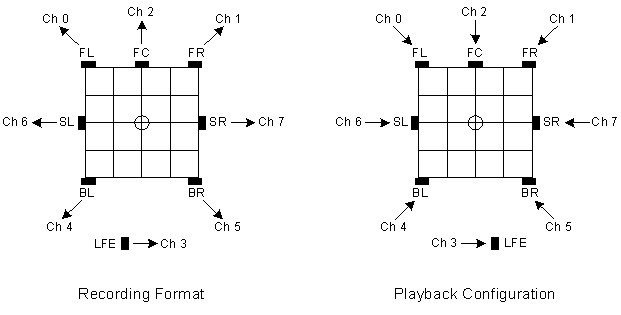
上图中通道掩码中扬声器位置的编码类似于用于[**KSPROPERTY\_AUDIO\_CHANNEL\_CONFIG**](https://docs.microsoft.com/windows-hardware/drivers/audio/ksproperty-audio-channel-config)属性请求的属性值的[**编码**](https://docs.microsoft.com/windows-hardware/drivers/audio/ksproperty-audio-channel-config)。有关更多信息，请参见[**KSAUDIO\_CHANNEL\_CONFIG**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksaudio_channel_config)。

下表显示了上图中每个掩码位的含义。

| **位数** | **演讲者位置** | **描述** |
| --- | --- | --- |
| 0 | FL | 左前 |
| 1个 | FR | 右前方 |
| 2 | 足球俱乐部 | 前中心 |
| 3 | LFE | 低频效果 |
| 4 | BL | 向左后退 |
| 5 | BR | 向右后退 |
| 6 | FLC | 中心左前 |
| 7 | FRC | 中心右前 |
| 8 | 公元前 | 背部中心 |
| 9 | SL | 左面 |
| 10 | SR | 右侧右侧 |

例如，**7.1家庭影院扬声器**配置由0x63F的通道掩码值描述，该值表示流中的八个通道已分配给以下扬声器位置（并按以下顺序）：FL，FR，FC，LFE ，BL，BR，SL和SR。再举一个例子，**7.1宽扬声器**配置由0xFF的通道掩码值描述，该值表示流中的八个通道已分配给以下扬声器位置：FL，FR，FC，LFE，BL，BR，FLC和FRC。

下图显示了通道掩码0x63F和**7.1家庭影院扬声器**配置之间的对应关系。



上图的左侧显示了将音频内容录制为**7.1家庭影院扬声器**流格式的过程。网格中心的小圆圈代表听众的位置。每个黑色小矩形代表一个麦克风。八个通道从0到7编号。FL麦克风记录到通道0，FR麦克风记录到通道1，依此类推。

上图的右侧显示了通过八扬声器环绕声配置播放的相同7.1声道流。在这种情况下，每个黑色的小矩形代表一个扬声器。七个扬声器映射到围绕收听者的网格上的位置。映射未将栅格位置分配给LFE扬声器（低音炮）。忽略是基于这样的假设，即这些扬声器通常仅产生非定向的低频声音。

**将流格式映射到扬声器配置**

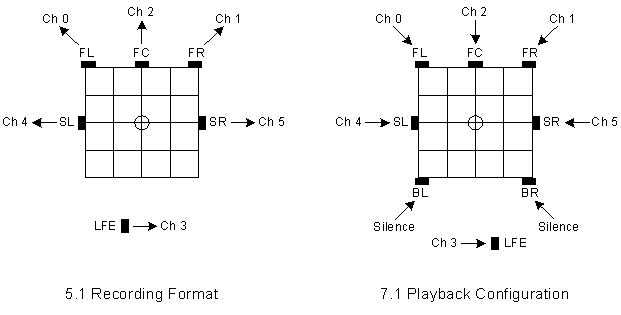
当要求播放与音频设备的扬声器配置不匹配的流格式时，音频驱动程序有几个选项：

* 拒绝播放流。
* 通过执行单个通道到扬声器的一对一映射来播放流。如果将一个通道映射到每个扬声器后还有任何通道遗留，请丢弃剩余的通道。相反，如果在将所有声道分配给扬声器后仍剩有扬声器，请通过剩余的扬声器播放静音。
* 通过混合原始流中的声道来播放流，以精确生成扬声器配置所需的声道数。如果原始流中的频道多于扬声器，则此方法可避免由于简单地丢弃多余的频道而造成的内容损失。混合和格式转换可以在软件或硬件中执行。

关于第三个选项，驱动程序应避免直接执行软件混合。相反，硬件供应商应安装全局效果（GFX）软件模块以在流到达音频设备之前对其进行处理。在Windows Vista中，全局效果被实现为GFX音频处理对象（APO）。在Windows Server 2003和Windows XP中，全局效果被实现为GFX筛选器。

**在7.1扬声器配置上播放5.1声道流**

下图显示了为5.1环绕声扬声器配置（左）录制的流，但通过7.1家庭影院扬声器配置（右）播放的流。



在上图中，录制的5.1格式不包含7.1扬声器配置中BL和BR扬声器位置的通道信息。因此，这两个扬声器保持沉默。（另一个更困难的选择是，音频设备通过混合录音中原始六个声道的内容来为BL和BR扬声器合成两个附加声道。）

根据通道掩码位的定义，用于记录上图左侧所示的5.1流的通道掩码应为0x60F，它将六个通道分配给以下扬声器位置：FL，FR，FC，LFE ，SL和SR。（这是前面讨论过的侧面扬声器5.1配置。）实际上，由于先前提到的原因，5.1流的通道掩码为0x3F而不是0x60F，现在将对其进行详细说明。

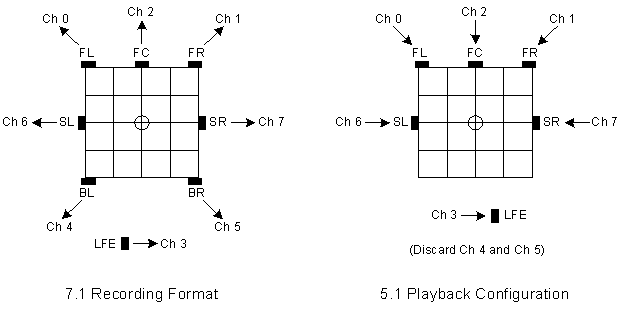
在Windows的早期版本（Windows Server 2003，带有SP1，Windows XP和Windows Me / 98的Windows XP）中，通道掩码0x3F的解释是它将5.1格式的六个通道分配给以下扬声器位置：FL ，FR，FC，LFE，BL和BR。（这是后扬声器5.1配置。）但是，在Windows Vista，带有SP1的Windows Server 2003和带有SP2的Windows XP中的解释是不同的：按照惯例，通道掩码为0x3F的5.1格式被解释为表示*一侧。 -speaker* 5.1配置，而不是*后置扬声器* 5.1配置。

以这种方式解释通道掩码消除了引入第二个5.1通道格式描述符的需求，以将侧扬声器5.1配置与后扬声器5.1配置区分开。这两个配置是如此相似，以至于普通用户可能难以区分它们。尽管只有一个5.1声道格式描述符可以避免使用户感到困惑，但确实需要硬件供应商记住将0x3F声道掩码解释为意味着将声道5和6分配给SL和SR扬声器位置，而不是BL和BR位置。 。作为必须记住对5.1流的通道掩码的这种特殊情况的解释，供应商可以使用户免于区分两个非常相似的5.1通道格式描述符的困难。

相信至少某些用户可能希望区分侧扬声器5.1配置和后扬声器5.1配置的供应商可以选择为此提供用户界面（UI）程序。通过UI，用户可以选择5.1声道流中的声道4和5是否应该驱动后置扬声器，而不是7.1家庭影院扬声器配置中的侧面扬声器。

**在5.1扬声器配置上播放7.1声道流**

下图显示了通过5.1环绕声扬声器配置（右）播放的7.1家庭影院扬声器配置（左）记录的流。7.1通道流的通道掩码为0x63F。



在此示例中，包含7.1配置中侧面扬声器位置数据的通道6和7播放5.1配置中的侧面扬声器位置。当音频设备在5.1配置中播放流时，音频设备仅丢弃包含第7.1配置中后扬声器位置数据的通道4和5。如前所述，另一种替代方案（在上图中未显示）是该设备尝试通过将它们与通道6和7混合，然后再通过5.1配置的侧扬声器播放它们来保留通道4和5中的内容。

### 系统混合器行为

在Windows Server 2003，Windows XP，Windows 2000和Windows Me / 98中，音频设备播放的多通道音频流通常由软件系统混合器Kmixer.sys生成。在开始播放流之前，系统混合器和音频驱动程序必须协商混合器和驱动程序都可以处理的流格式。

当要求播放格式与音频设备的扬声器配置不匹配的多声道流时，音频驱动程序可以拒绝该请求，在这种情况下，协商将继续。

系统混合器可以将内容从5.1声道输入流转换为7.1声道输出流（到音频设备），反之亦然，尽管它更喜欢避免这种转换以保持输入流的质量，但它仍然可以。因此，系统混合器通过要求驾驶员接受与系统混合器的最高质量输入流相同格式的流来开始协商。通常，这意味着如果系统混音器具有5.1或7.1通道格式的输入流，它将要求驱动程序接受相同格式的流。如果驱动程序拒绝此格式，则系统混合器将继续询问驱动程序是否可以处理其他流格式。

例如，如果具有5.1扬声器配置的音频设备的驱动程序拒绝了系统混音器播放7.1声道流的请求，则系统混音器将通过将7.1声道流转换为5.1声道来继续协商。流。如果驱动程序接受此格式，则系统混合器将为驱动程序执行流转换。

在设计音频驱动程序时，驱动程序编写者必须决定是处理自己的格式转换还是依靠系统混合器进行转换。在以下两种情况下，驱动程序可能需要处理转换：

* 如果驱动程序要求以与系统混合器执行的转换不同的方式执行转换。
* 如果驱动程序必须播放绕过系统混音器的流。

在第二种情况下，如果流是从Microsoft DirectSound硬件加速缓冲区直接播放到音频设备上的硬件混合引脚的，则流可以绕过系统混合器。此外，某些“专业音频”应用程序会将其流直接发送到音频设备，以避免系统混音器的等待时间，或者防止混音过程更改原始音频流中的数字采样值。

在带有SP1的Windows Server 2003和带有SP2的Windows XP中，如果系统混合器生成5.1通道的输出流，则混合器始终将流的通道掩码设置为0x3F。即使系统混音器接收到带有0x60F通道掩码的5.1通道输入流，它也将以这种方式运行。通过这种行为，音频驱动程序永远不会从混音器接收带有0x60F通道掩码的5.1通道流。

如果系统混音器接收到通道屏蔽为0x63F的7.1通道输入流并产生5.1通道输出流（通道屏蔽为0x3F），则混频器会将输入流中的通道6和7复制到通道4和5在输出流中。混音器从7.1声道输入流中丢弃声道4和5（用于两个后扬声器）。此行为可确保在7.1声道流中包含两个侧面扬声器内容的声道通过5.1扬声器配置中的侧面扬声器播放。

# 头文件更改

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/header-file-changes.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/header-file-changes.md" \o "1个贡献者)

Windows驱动程序工具包（WDK）包含两个头文件，这些头文件定义Windows多媒体控制面板支持的扬声器配置：

* Ksmedia.h定义了由[**KSPROPERTY\_AUDIO\_CHANNEL\_CONFIG**](https://docs.microsoft.com/windows-hardware/drivers/audio/ksproperty-audio-channel-config)属性请求使用的[**KSAUDIO\_CHANNEL\_CONFIG**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksaudio_channel_config)结构的通道掩码。
* Dsound.h定义了可以提交给**IDirectSound :: SetSpeakerConfig**方法的扬声器配置标识符的列表。有关此方法的更多信息，请参见Windows SDK文档。

在Windows Server 2003，带有SP1，Windows 2000和Windows Me / 98的Windows XP中，Ksmedia.h定义了下表中显示的5.1和7.1通道流的通道掩码。

| **参数名称** | **通道遮罩** | **演讲者职位** |
| --- | --- | --- |
| KSAUDIO\_SPEAKER\_5POINT1 | 0x3F | FL，FR，FC，LFE，BL，BR |
| KSAUDIO\_SPEAKER\_7POINT1 | 0xFF | FL，FR，FC，LFE，BL，BR，FLC，FRC |

上表中的两个通道掩码分别代表5.1扬声器配置和7.1扬声器配置。为了标识相同的两个扬声器配置，Dsound.h定义了以下扬声器配置ID：

C ++复制

#define DSSPEAKER\_5POINT1 0x00000006

#define DSSPEAKER\_7POINT1 0x00000007

在Windows XP SP2和更高版本的Windows中，Ksmedia.h定义了下表中针对5.1和7.1通道流的通道掩码。

参数名称 通道遮罩 演讲者职位

KSAUDIO\_SPEAKER\_5POINT1 0x3F FL，FR，FC，LFE，BL，BR

KSAUDIO\_SPEAKER\_7POINT1\_SURROUND 0x63F FL，FR，FC，LFE，BL，BR，SL，SR

通过比较前面的两个表，可以明显看出以下几点：

* 即使在Windows SP2和更高版本的Windows中，即使将KSAUDIO\_SPEAKER\_5POINT1解释为使用SL和SR扬声器而不是BL和BR，第一个表中通道掩码0x3F的含义在第二个表中也没有更改。
* 支持值为0x63F的新通道掩码。此通道掩码代表7.1家庭影院扬声器配置。
* **注意**    在Windows Vista和更高版本的Windows中，不再支持KSAUDIO\_SPEAKER\_7POINT1扬声器配置。因此，它不是控制面板中的可用选项。

为了表示同一组扬声器配置，Dsound.h定义了以下扬声器配置ID：

C ++

复制

#define DSSPEAKER\_5POINT1 0x00000006

#define DSSPEAKER\_7POINT1 0x00000007

#define DSSPEAKER\_7POINT1\_SURROUND 0x00000008

#define DSSPEAKER\_7POINT1\_WIDE DSSPEAKER\_7POINT1

DSSPEAKER\_7POINT1\_SURROUND表示控制面板中新的7.1家庭影院扬声器配置。DSSPEAKER\_7POINT1和DSSPEAKER\_7POINT1\_WIDE都是相同的7.1宽配置扬声器配置的名称。

有关DirectSound扬声器配置的更多信息，请参阅[DirectSound扬声器配置设置](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/directsound-speaker-configuration-settings)。

# 音频数据格式和数据范围的示例

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/examples-of-audio-data-formats-and-data-ranges.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/examples-of-audio-data-formats-and-data-ranges.md" \o "1个贡献者)

下面的示例演示如何使用[**KSDATAFORMAT**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ks/ns-ks-ksdataformat)和[**KSDATARANGE**](https://docs.microsoft.com/previous-versions/ff561658(v=vs.85))结构来描述一些更常见的音频流格式：

[模拟音频流数据范围](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/analog-audio-stream-data-range)

[DirectMusic流数据格式](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/directmusic-stream-data-format)

[DirectMusic流数据范围](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/directmusic-stream-data-range)

[DirectSound流数据格式](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/directsound-stream-data-format)

[DirectSound流数据范围](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/directsound-stream-data-range)

[MIDI流数据格式](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/midi-stream-data-format)

[MIDI流数据范围](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/midi-stream-data-range)

[PCM流数据格式](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/pcm-stream-data-format)

[PCM流数据范围](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/pcm-stream-data-range)

[PCM多通道流数据格式](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/pcm-multichannel-stream-data-format)

[PCM多通道流数据范围](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/pcm-multichannel-stream-data-range)

[PCM高位深度流数据格式](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/pcm-high-bitdepth-stream-data-format)

[PCM高位深度流数据范围](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/pcm-high-bitdepth-stream-data-range)

# 模拟音频流数据范围

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/analog-audio-stream-data-range.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/analog-audio-stream-data-range.md" \o "1个贡献者)

本示例使用[**KSDATARANGE**](https://docs.microsoft.com/previous-versions/ff561658(v=vs.85))结构描述模拟音频流的数据范围。

C ++复制

DataRange.FormatSize = sizeof(KSDATARANGE);

DataRange.Flags = 0;

DataRange.SampleSize = 0;

DataRange.Reserved = 0;

DataRange.MajorFormat = STATICGUIDOF(KSDATAFORMAT\_TYPE\_AUDIO);

DataRange.SubFormat = STATICGUIDOF(KSDATAFORMAT\_SUBTYPE\_ANALOG);

DataRange.Specifier = STATICGUIDOF(KSDATAFORMAT\_SPECIFIER\_NONE);

通常，微型端口驱动程序使用这种类型的数据范围来描述通过桥接引脚的模拟信号，该桥接引脚表示音频适配器卡上的硬连线连接。有关桥脚的更多信息，请参见[音频滤波器图](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/audio-filter-graphs)。另外，请参见[暴露过滤器拓扑中](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/exposing-filter-topology)的代码示例

# DirectMusic流数据格式

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/directmusic-stream-data-format.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/directmusic-stream-data-format.md" \o "1个贡献者)

本示例使用[**KSDATAFORMAT**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ks/ns-ks-ksdataformat)结构来描述DirectMusic流的数据格式。

C ++复制

DataFormat.FormatSize = sizeof(KSDATAFORMAT);

DataFormat.Flags = 0;

DataFormat.SampleSize = 0;

DataFormat.Reserved = 0;

DataFormat.MajorFormat = STATICGUIDOF(KSDATAFORMAT\_TYPE\_MUSIC);

DataFormat.SubFormat = STATICGUIDOF(KSDATAFORMAT\_SUBTYPE\_DIRECTMUSIC);

DataFormat.Specifier = STATICGUIDOF(KSDATAFORMAT\_SPECIFIER\_NONE);

# DirectMusic流数据范围

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/directmusic-stream-data-range.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/directmusic-stream-data-range.md" \o "1个贡献者)

本示例使用[**KSDATARANGE\_MUSIC**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdatarange_music)结构来描述DirectMusic流的数据范围。

C ++复制

DataRange.FormatSize = sizeof(KSDATARANGE\_MUSIC);

DataRange.Flags = 0;

DataRange.SampleSize = 0;

DataRange.Reserved = 0;

DataRange.MajorFormat = STATICGUIDOF(KSDATAFORMAT\_TYPE\_MUSIC);

DataRange.SubFormat = STATICGUIDOF(KSDATAFORMAT\_SUBTYPE\_DIRECTMUSIC);

DataRange.Specifier = STATICGUIDOF(KSDATAFORMAT\_SPECIFIER\_NONE);

Technology = STATICGUIDOF(KSMUSIC\_TECHNOLOGY\_WAVETABLE);

Channels = 0;

Notes = 0;

ChannelMask = 0xFFFF;

# DirectSound流数据格式

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/directsound-stream-data-format.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/directsound-stream-data-format.md" \o "1个贡献者)

本示例使用[**KSDATAFORMAT\_DSOUND**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdataformat_dsound)结构来描述DirectSound流的数据格式。

C ++复制

DataFormat.FormatSize = sizeof(KSDATAFORMAT\_DSOUND);

DataFormat.Flags = 0;

DataFormat.SampleSize = 0;

DataFormat.Reserved = 0;

DataFormat.MajorFormat = STATICGUIDOF(KSDATAFORMAT\_TYPE\_AUDIO);

DataFormat.SubFormat = STATICGUIDOF(KSDATAFORMAT\_SUBTYPE\_PCM);

DataFormat.Specifier = STATICGUIDOF(KSDATAFORMAT\_SPECIFIER\_DSOUND);

BufferDesc.Flags = KSDSOUND\_BUFFER\_LOCHARDWARE;

BufferDesc.Control = KSDSOUND\_BUFFER\_CTRL\_3D;

BufferDesc.WaveFormatEx.wFormatTag = WAVE\_FORMAT\_PCM;

BufferDesc.WaveFormatEx.nChannels = 2;

BufferDesc.WaveFormatEx.nSamplesPerSec = 22050;

BufferDesc.WaveFormatEx.nAvgBytesPerSec = 88200;

BufferDesc.WaveFormatEx.nBlockAlign = 4;

BufferDesc.WaveFormatEx.wBitsPerSample = 16;

BufferDesc.WaveFormatEx.cbSize = 0;

# DirectSound流数据范围

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/directsound-stream-data-range.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/directsound-stream-data-range.md" \o "1个贡献者)

本示例使用[**KSDATARANGE\_AUDIO**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdatarange_audio)结构描述DirectSound流的数据范围。

C ++复制

DataRange.FormatSize = sizeof(KSDATARANGE\_AUDIO);

DataRange.Flags = 0;

DataRange.SampleSize = 0;

DataRange.Reserved = 0;

DataRange.MajorFormat = STATICGUIDOF(KSDATAFORMAT\_TYPE\_AUDIO);

DataRange.SubFormat = STATICGUIDOF(KSDATAFORMAT\_SUBTYPE\_PCM);

DataRange.Specifier = STATICGUIDOF(KSDATAFORMAT\_SPECIFIER\_DSOUND);

MaximumChannels = 4; // max number of channels, or -1 for unlimited

MinimumBitsPerSample = 2;

MaximumBitsPerSample = 16; // 16, 24, 32, etc.

MinimumSampleFrequency = 5000;

MaximumSampleFrequency = 48000;

此示例中的成员值与[PCM多通道流数据范围](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/pcm-multichannel-stream-data-range)示例中的成员值类似，但**MaximumBitsPerSample**值除外。此值设置为样本容器的大小，并且应为8的倍数。例如，如果设备在24位容器中支持20位有效音频数据，则**MaximumBitsPerSample**的值应设置为24。

# MIDI流数据格式

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/midi-stream-data-format.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/midi-stream-data-format.md" \o "1个贡献者)

本示例使用[**KSDATAFORMAT**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ks/ns-ks-ksdataformat)结构来描述MIDI流的数据格式。

C ++复制

DataFormat.FormatSize = sizeof(KSDATAFORMAT);

DataFormat.Flags = 0;

DataFormat.SampleSize = 0;

DataFormat.Reserved = 0;

DataFormat.MajorFormat = STATICGUIDOF(KSDATAFORMAT\_TYPE\_MUSIC);

DataFormat.SubFormat = STATICGUIDOF(KSDATAFORMAT\_SUBTYPE\_MIDI);

DataFormat.Specifier = STATICGUIDOF(KSDATAFORMAT\_SPECIFIER\_NONE);

# MIDI流数据范围

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/midi-stream-data-range.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/midi-stream-data-range.md" \o "1个贡献者)

本示例使用[**KSDATARANGE\_MUSIC**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdatarange_music)结构来描述MIDI流的数据范围。

C ++复制

DataRange.FormatSize = sizeof(KSDATARANGE\_MUSIC);

DataRange.Flags = 0;

DataRange.SampleSize = 0;

DataRange.Reserved = 0;

DataRange.MajorFormat = STATICGUIDOF(KSDATAFORMAT\_TYPE\_MUSIC);

DataRange.SubFormat = STATICGUIDOF(KSDATAFORMAT\_SUBTYPE\_MIDI);

DataRange.Specifier = STATICGUIDOF(KSDATAFORMAT\_SPECIFIER\_NONE);

Technology = STATICGUIDOF(KSMUSIC\_TECHNOLOGY\_PORT);

Channels = 0;

Notes = 0;

ChannelMask = 0xFFFF;

# PCM流数据格式

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/pcm-stream-data-format.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/pcm-stream-data-format.md" \o "1个贡献者)

本示例使用[**KSDATAFORMAT\_WAVEFORMATEX**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdataformat_waveformatex)结构来描述PCM流的数据格式。

C ++复制

DataFormat.FormatSize = sizeof(KSDATAFORMAT\_WAVEFORMATEX);

DataFormat.Flags = 0;

DataFormat.SampleSize = 0;

DataFormat.Reserved = 0;

DataFormat.MajorFormat = STATICGUIDOF(KSDATAFORMAT\_TYPE\_AUDIO);

DataFormat.SubFormat = STATICGUIDOF(KSDATAFORMAT\_SUBTYPE\_PCM);

DataFormat.Specifier = STATICGUIDOF(KSDATAFORMAT\_SPECIFIER\_WAVEFORMATEX);

Format.wFormatTag = WAVE\_FORMAT\_PCM;

Format.nChannels = 2;

Format.nSamplesPerSec = 44100;

Format.nAvgBytesPerSec = 176400;

Format.nBlockAlign = 4;

Format.wBitsPerSample = 16;

Format.cbSize = 0;

# PCM流数据范围

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/pcm-stream-data-range.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/pcm-stream-data-range.md" \o "1个贡献者)

本示例使用[**KSDATARANGE\_AUDIO**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdatarange_audio)结构来描述PCM流的数据范围。

C ++复制

DataRange.FormatSize = sizeof(KSDATARANGE\_AUDIO);

DataRange.Flags = 0;

DataRange.SampleSize = 0;

DataRange.Reserved = 0;

DataRange.MajorFormat = STATICGUIDOF(KSDATAFORMAT\_TYPE\_AUDIO);

DataRange.SubFormat = STATICGUIDOF(KSDATAFORMAT\_SUBTYPE\_PCM);

DataRange.Specifier = STATICGUIDOF(KSDATAFORMAT\_SPECIFIER\_WAVEFORMATEX);

MaximumChannels = 2;

MinimumBitsPerSample = 2;

MaximumBitsPerSample = 16;

MinimumSampleFrequency = 5000;

MaximumSampleFrequency = 48000;

# PCM多通道流数据格式

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/pcm-multichannel-stream-data-format.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/pcm-multichannel-stream-data-format.md" \o "1个贡献者)

本示例使用[**KSDATAFORMAT\_WAVEFORMATEX**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdataformat_waveformatex)结构的扩展版本来描述PCM多通道流的数据格式。

C ++复制

DataFormat.FormatSize = sizeof(KSDATAFORMAT) + sizeof(WAVEFORMATEXTENSIBLE);

DataFormat.Flags = 0;

DataFormat.SampleSize = 0;

DataFormat.Reserved = 0;

DataFormat.MajorFormat = STATICGUIDOF(KSDATAFORMAT\_TYPE\_AUDIO);

DataFormat.SubFormat = STATICGUIDOF(KSDATAFORMAT\_SUBTYPE\_PCM);

DataFormat.Specifier = STATICGUIDOF(KSDATAFORMAT\_SPECIFIER\_WAVEFORMATEX);

Format.wFormatTag = WAVE\_FORMAT\_EXTENSIBLE;

Format.nChannels = 4;

Format.nSamplesPerSec = 44100;

Format.nAvgBytesPerSec = 352800;

Format.nBlockAlign = 8;

Format.wBitsPerSample = 16;

Format.cbSize = sizeof(WAVEFORMATEXTENSIBLE) - sizeof(WAVEFORMATEX);

Format.wValidBitsPerSample = 16;

Format.dwChannelMask = KSAUDIO\_SPEAKER\_SURROUND;

Format.SubFormat = KSDATAFORMAT\_SUBTYPE\_PCM;

# PCM多通道流数据范围

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/pcm-multichannel-stream-data-range.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/pcm-multichannel-stream-data-range.md" \o "1个贡献者)

本示例使用[**KSDATARANGE\_AUDIO**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdatarange_audio)结构来描述PCM多通道流的数据范围。

C ++复制

DataRange.FormatSize = sizeof(KSDATARANGE\_AUDIO);

DataRange.Flags = 0;

DataRange.SampleSize = 0;

DataRange.Reserved = 0;

DataRange.MajorFormat = STATICGUIDOF(KSDATAFORMAT\_TYPE\_AUDIO);

DataRange.SubFormat = STATICGUIDOF(KSDATAFORMAT\_SUBTYPE\_PCM);

DataRange.Specifier = STATICGUIDOF(KSDATAFORMAT\_SPECIFIER\_WAVEFORMATEX);

MaximumChannels = 4; // max number of channels, or -1 for unlimited

MinimumBitsPerSample = 2;

MaximumBitsPerSample = 16;

MinimumSampleFrequency = 5000;

MaximumSampleFrequency = 48000;

# PCM高位深度流数据格式

* 2017/04/20
* 2分钟阅读

本示例使用[**KSDATAFORMAT\_WAVEFORMATEX**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdataformat_waveformatex)结构的扩展版本来描述PCM高位深度流的数据格式。这是类似于PCM多信道例如，与用于值的异常Format.wBitsPerSample和Format.wValidBitsPerSample下面显示。

C ++复制

DataFormat.FormatSize = sizeof(KSDATAFORMAT) + sizeof(WAVEFORMATEXTENSIBLE);

DataFormat.Flags = 0;

DataFormat.SampleSize = 0;

DataFormat.Reserved = 0;

DataFormat.MajorFormat = STATICGUIDOF(KSDATAFORMAT\_TYPE\_AUDIO);

DataFormat.SubFormat = STATICGUIDOF(KSDATAFORMAT\_SUBTYPE\_PCM);

DataFormat.Specifier = STATICGUIDOF(KSDATAFORMAT\_SPECIFIER\_WAVEFORMATEX);

Format.wFormatTag = WAVE\_FORMAT\_EXTENSIBLE;

Format.nChannels = 4;

Format.nSamplesPerSec = 44100;

Format.nAvgBytesPerSec = 529200;

Format.nBlockAlign = 8;

Format.wBitsPerSample = 24;

Format.cbSize = sizeof(WAVEFORMATEXTENSIBLE) - sizeof(WAVEFORMATEX);

Format.wValidBitsPerSample = 20;

Format.dwChannelMask = KSAUDIO\_SPEAKER\_SURROUND;

Format.SubFormat = KSDATAFORMAT\_SUBTYPE\_PCM;

# PCM高位深度流数据范围

* 2017/04/20
* 2分钟阅读
  + [[https://github.com/DOMARS.png?size=32](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/pcm-high-bitdepth-stream-data-range.md)](https://github.com/MicrosoftDocs/windows-driver-docs/blob/staging/windows-driver-docs-pr/audio/pcm-high-bitdepth-stream-data-range.md" \o "1个贡献者)

本示例使用[**KSDATARANGE\_AUDIO**](https://docs.microsoft.com/windows-hardware/drivers/ddi/content/ksmedia/ns-ksmedia-ksdatarange_audio)结构来描述PCM高位深度流的数据范围。

C ++复制

DataRange.FormatSize = sizeof(KSDATARANGE\_AUDIO);

DataRange.Flags = 0;

DataRange.SampleSize = 0;

DataRange.Reserved = 0;

DataRange.MajorFormat = STATICGUIDOF(KSDATAFORMAT\_TYPE\_AUDIO);

DataRange.SubFormat = STATICGUIDOF(KSDATAFORMAT\_SUBTYPE\_PCM);

DataRange.Specifier = STATICGUIDOF(KSDATAFORMAT\_SPECIFIER\_WAVEFORMATEX);

MaximumChannels = 4; // max number of channels, or -1 for unlimited

MinimumBitsPerSample = 2;

MaximumBitsPerSample = 24; // 24, 32, etc.

MinimumSampleFrequency = 5000;

MaximumSampleFrequency = 48000;

此示例中的成员值与[PCM多通道流数据范围](https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/pcm-multichannel-stream-data-range)示例中的成员值相似，但该MaximumBitsPerSample值大于16。该值设置为支持的最大有效位数。例如，如果设备在24位容器中支持20位有效音频数据，MaximumBitsPerSample则应将的值设置为20。