# SoundFont2 标准翻译

译者：cymheart

开始时间: 2019/9/8

结束时间:

### 1 Introduction

1.1 Scope and Intended Purpose of this Document

This document is the definitive source for the SoundFont 2 standard. This document should provide complete and accurate information to allow any user to correctly construct and interpret SoundFont 2 compatible banks. This document is not intended to provide any information on the design or implementation of music synthesizers.

### 1引言

1.1本文档的技术范围和期望达成的目的

本文档是”SoundFont 2标准”的权威文档，提供完整和准确的协议信息，以允许任何人正确构建和解析与SoundFont 2兼容的库。但本文档没有提供任何关于音乐合成器的设计或实现的信息。

1.2 Document Organization

This document is organized such that sections 1 and 2 give introductory information about the SoundFont 2 standard. Both new and seasoned musical engineers will get value from the review of terminology provided in section 2. Sections 3 through 8 provide increasingly detailed descriptions of the SoundFont 2 standard data structures. The sections will ultimately serve as reference, but can be scanned in order to provide sufficient detail for any level of understanding. Section 9 deals with the Synthesis model supported by the SoundFont standard, and will be of interest to anyone involved with the synthesis engine or bank creation. Section 10 specifies error handling when dealing with SoundFont compatible banks, and will be of interest primarily to programmers using the SoundFont standard. The alphabetical glossary in section 11 can be used as a reference for any unfamiliar or confusing terminology.

1.2文档结构组织方式

本文档的结构组织方式如下:第1节和第2节给出了关于SoundFont 2标准的介绍信息。新的和经验丰富的音乐工程师都将在第2节提供的术语说明中获得有用的东西。第3节到第8节逐步详细地描述了SoundFont 2标准的数据结构。这些部分最终将作为参考档案，可以随时翻阅，以便为任何层次的理解提供足够的细节上的支持。第9节讨论了SoundFont标准支持的合成器模型，任何涉及合成器引擎或库创建的人员都会感兴趣这部分。第10节指定了处理与SoundFont兼容库的错误处理，主要是使用SoundFont标准的程序员感兴趣这部分内容。第11节中的术语表可作为任何不熟悉或混淆的术语的参考。

1.3 SoundFont 2 Objectives

The SoundFont 2 standard is intended to provide an extensible, portable, universal interchange format for wavetable synthesizer “samples” and articulation data. The standard is made extensible largely by the use of enumerated “generators” and “modulators” so that additional function units can be added as requirements dictate. The standard is made portable and universal by the use of precisely defined and hardware independent parameters, as well as by specific practices designed to provide support to a broad range of technologies.

1.3 SoundFont 2 的设计目标宗旨

SoundFont 2标准的设计目标是提供一个可扩展的、可移植的,通用的交换协议格式来生成波表合成器的”采样”和发音方法数据。通过使用可枚举的“生成器”和“调制器”，使协议标准具有很大的可扩展性，从而可以根据要求添加附加的功能单元。通过使用精确定义的与硬件无关的参数，以及通过为广泛的技术提供支持而设计的具体实践，协议标准变得可移植和通用。

1.4 SoundFont 1.x

The SoundFont standard was originally released in its 1.0 embodiment with the Creative Labs AWE32 product using the EMU8000 music synthesis chip. This proprietary format proved very successful, but experience brought a number of refinements. These initially were performed in an upward compatible manner to revision 1.5.

However, due to increasing demand for a public downloadable sound interchange format, Creative Technology determined that a public disclosure of the SoundFont format would be in its best interest. Because there were still more improvements required, many of which could not be supported in a completely compatible manner, Creative decided to combine public disclosure with the step to a revised format. The result is the SoundFont 2 standard.

There are several key enhancements contained in the SoundFont 2 standard. The first is the use of relative parameters in the Preset level. This allows instruments to be adjusted without altering their self-consistency, providing easy and effective user editing of instruments. The second is an improvement in the data structures associated with the samples themselves, again providing key information which will allow the sound designer to re-use samples with a minimum of difficulty. An increased specificity in the rules for sample data produces enhanced portability across various sound engines. Finally, the addition of modulators produces a robust structure which can express all the typical function in current and future wavetable synthesizers.

1.4 SoundFont 1.x

SoundFont标准最初是在1.0版实施推出，并跟随Creative Labs AWE32产品使用的EMU8000音乐合成芯片一起发布。这种专有格式被证明是非常成功的，之后产品经验带来了许多改进。这些改进最初以向上兼容的方式，一直修订到1.5版。

但是，由于用户对公共可下载声音交换格式的需求不断增加，Creative Technology确定公开开源SoundFont格式，是符合其最佳利益的。但由于还需要进行更多改进，其中许多改进不能以完全兼容的方式得到支持，因此Creative决定将公开披露与修改后的格式相结合。结果诞生了SoundFont 2标准格式。

SoundFont 2标准中包含几个关键增强功能。第一个是在预设层级中使用相对参数。这样可以在不改变仪器自身一致性的情况下调整仪器，从而轻松有效地对仪器进行编辑。第二个是与采样本身相关的数据结构的改进，再次提供关键信息，使声音设计者能够以最小的难度重复使用采样数据。采样数据规则的特异性增加，可以增强各种声音引擎的可移植性。最后，调制器的加入，而产生了一个健壮的结构，达到可以表达当前和未来波表合成器的所有典型的功能。

1.5 Future Enhancements to the SoundFont 2 Standard

The SoundFont 2 standard is designed to allow for enhancements based on future wavetable synthesis technology capabilities by additional enumerations of generators and modulators. This will be done as required in an upwardly compatible manner. Suggestions for additions can be made via e-mail to soundfont@emu.com. In general, our policy for updating the specification will be based on consumer need, rather than technological idealism.

It is our expectation to maintain bi-directional compatibility within the SoundFont 2 standard for some years.

1.5 SoundFont 2标准的未来增强功能

SoundFont 2标准旨在通过增加额外的生成器和调制器枚举，实现基于未来波表合成技术功能的增强。 这将按照向上兼容的方式按要求完成。 有关添加的建议，请发送电子邮件至soundfont@emu.com。 一般而言，我们更新规范的政策将基于实际的用户需求，而不是技术理想主义。

我们期望在SoundFont 2标准中能够保持多年的双向兼容性。

### 2 Terms and Abbreviations

The following sections introduce terms used within this specification in a logical order. They are provided both as an introduction to readers unfamiliar with wavetable synthesis implementation details, as well as a review and reference for the expert. These and other terms and abbreviations can also be found arranged alphabetically for reference in the glossary at the end of this specification.

### 2术语和缩写

下面几节将按逻辑顺序介绍此规范中使用的术语。为不熟悉波形合成实现细节的读者提供了概念介绍，同时为这方面的专家提供了综述和参考。这些术语和缩写，已在本规范文档末尾的术语表中按字母顺序罗列，以方便查找参考。

2.1 Data Structure Terminology

bag - A SoundFont data structure element containing a list of preset zones or instrument zones

big endian - Refers to the organization in memory of bytes within a word such that the most significant byte occurs at the lowest address. Contrast “little endian.”

byte - A data structure element of eight bits without definition of meaning to those bits.

BYTE - A data structure element of eight bits which contains an unsigned value from 0 to 255.

case-insensitive - Indicates that an ASCII character or string treats alphabetic characters of upper or lower case as identical. Contrast “case-sensitive.”

case-sensitive - Indicates that an ASCII character or string treats alphabetic characters of upper or lower case as distinct. Contrast “case-insensitive.”

CHAR - A data structure of eight bits which contains a signed value from -128 to +127.

chunk - The top-level division of a RIFF file.

doubleword - A data structure element of 32 bits without definition of meaning to those bits.

DWORD - A data structure of 32 bits which contains an unsigned value from zero to 4,294,967,295.

enumerated - Said of a data element whose symbols correspond to particular assigned functions.

global - Refers to parameters which affect all associated structures. See “global zone”

global zone - A zone whose generators and modulators affect all other zones within the object.

header - A data structure element which describes several aspects of a SoundFont element.

hydra - A. A nine-headed mythical beast. B. The nine “pdta” sub-chunks which make up the SoundFont articulation data.

instrument - In the SoundFont standard, a collection of zones which represents the sound of a single musical instrument or sound effect set.

instrument zone - A subset of an instrument containing a sample reference and associated articulation data defined to play over certain key numbers and velocities.

layer - An obsolete SoundFont term, now called a Preset Zone.

level - In the SoundFont structure, this refers either to the preset and preset zones (the preset level) or the instrument and instrument zones (the instrument level).

little endian - A method of ordering bytes within larger words in memory in which the least significant byte is at the lowest address. Contrast “big endian.”

object - Either an instrument or a preset, depending on what level (preset or instrument) is being discussed.

orphan - Said of a data structure which under normal circumstances is referenced by a higher level, but in this particular instance is no longer linked. Specifically, it is an instrument which is not referenced by any preset zone, or a sample which is not referenced by any instrument zone.

preset - A keyboard full of sound. Typically the collection of samples and articulation data associated with a particular MIDI preset number.

preset zone - A subset of a preset containing an instrument reference and associated articulation data defined to play over certain key numbers and velocities.

record - A single instance of a data structure.

RIFF - Acronym for Resource Interchange File Format. The recommended form for interchange files such as SoundFont compatible files within Microsoft operating systems.

SHORT - A data structure element of sixteen bits which contains a signed value from -32,768 to +32,767.

split - An obsolete SoundFont term, now called an Instrument Zone.

sub-chunk - A division of a RIFF file below that of the chunk.

terminator - A data structure element indicating the final element in a sequence.

WORD - A data structure of 16 bits which contains an unsigned value from zero to 65,535.

word - A data structure element of 16 bits without definition of meaning to those bits.

zone - An object and associated articulation data defined to play over certain key numbers and velocities.

2.1数据结构术语

bag – 一个SoundFont数据结构元素，包含一个preset zones (预设区)列表或instrument zones(乐器区)列表

big endian - 大端模式，是指数据的高字节保存在内存的低地址中，而数据的低字节保存在内存的高地址中，这样的存储模式有点儿类似于把数据当作字符串顺序处理：地址由小向大增加，而数据从高位往低位放；这和我们的阅读习惯一致。对比“little endian。”

byte - 8位的数据结构元素，没有对这些位的意义进行定义。

BYTE - —个8位的数据结构元素，包含一个从0到255的无符号值。

case-insensitive - 表示ASCII字符或字符串将大小写字母字符视为相同。

对比“case-sensitive”。

case-sensitive - 表示ASCII字符或字符串将大小写字母字符视为不同的。

对比“case-insensitive。”

CHAR - 8位的数据结构，包含从-128到+127的带符号值。

chunk - RIFF文件的顶层分割区块。

doubleword - 一个32位的数据结构元素，没有对这些位的意义进行定义。

DWORD - 一个32位的数据结构，包含一个从0到4,294,967,295的无符号值。

enumerated - 表示其符号对应于特定指定功能的数据元素 。

global - 指影响所有相关结构的参数。参见“global zone”

global zone - 一个区域，其生成器和调制器影响对象内的所有其他区域。

header - 描述SoundFont元素的几个方面的数据结构元素。

hydra - A.一种传说中的九头兽。B.构成SoundFont 演奏方法（articulation）数据的九个“pdta”子块。

instrument - 在SoundFont标准中，表示单个乐器声音或声效集声音的instrument zone集合。

instrument zone - instrument 的一个子集，包含一个采样引用和相关的articulation（演奏方法）数据的定义为在某些指定键号和指定力度上进行演奏。

layer - 一个过时的SoundFont术语，现在用Preset Zone(预设区)代替。

level - 在SoundFont结构中，这指的是preset(预设)和preset zones(预设区)(预设层级)或instrument（乐器）和 instrument zones（乐器区）(乐器层级)。

little endian - 小端模式，是指数据的高字节保存在内存的高地址中，而数据的低字节保存在内存的低地址中，这种存储模式将地址的高低和数据位权有效地结合起来，高地址部分权值高，低地址部分权值低。对比“big endian”。

object - instrument(乐器)或者preset(预设)，取决于正在讨论的level(层级)

orphan - 表示在正常情况下由更高层级引用，但在此特定实例中不再链接的数据结构。具体而言，它是一种未被任何preset zone(预设区)引用的instrument(乐器)，或是未被任何instrument zone(乐器区)引用的采样数据。

preset - 全声键盘。通常是与特定MIDI预设编号相关联的采样数据和articulation（演奏方法）数据的集合。

preset zone - preset(预设)的一个子集，包含一个instrument(乐器)引用和相关的articulation（演奏方法）数据，这些数据被定义为在某些指定键号和指定力度上进行演奏。

record - 某个数据结构的单个实例。

RIFF -资源交换文件格式的缩写。用于交换文件的推荐格式，如Microsoft操作系统中的SoundFont兼容文件。

SHORT - 一种16位的数据结构元素，包含从-32,768到+32,767的带符号值。

split - 一个过时的SoundFont术语，现在用 Instrument Zone（乐器区）代替。

sub-chunk - 一个RIFF文件在块的下面的部分。

terminator - 指示序列中最后一个元素的数据结构元素。

WORD - 一个16位的数据结构，包含一个从0到65,535的无符号值。

word - 一个16位的数据结构元素，没有对这些位的意义进行定义。

zone - 一个object和相关演奏方法的数据，这些数据被定义为在某些指定键号和指定速度上进行演奏。

2.2 Synthesizer Terminology

articulation - The process of modulation of amplitude, pitch, and timbre to produce an expressive musical note.

artifact - A (typically undesirable) sonic event which is recognizable as not being present in the original sound.

attack - That phase of an envelope or sound during which the amplitude increases from zero to a peak value.

attenuation - A decrease in volume or amplitude of a signal.

AWE32 - The original Creative Technology Sound Blaster product which contained an EMU8000 wavetable synthesizer and supported the SoundFont standard.

balance - A form of stereo volume control in which both left and right channels are at maximum when the control is centered, and which attenuates only the opposite channel when taken to either extreme.

bank - A collection of presets. See also MIDI bank.

chorus - An effects processing algorithm which involves cyclically shifting the pitch of a signal and remixing it with itself to produce a time varying comb filter, giving a perception of motion and fullness to the resulting sound.

cutoff frequency - The frequency of a filter function at which the attenuation reaches a specified value.

data points - The individual values comprising a sample. Sometimes also called sample points. Contrast “sample.”

decay - The portion of an envelope or sound during which the amplitude declines from a peak to steady state value.

delay - The portion of an envelope or LFO function which elapses from a key-on event until the amplitude becomes nonzero.

DC gain - The degree of amplification or attenuation a system presents to a static, or zero frequency, signal.

digital audio - Audio represented as a sequence of quantized values spaced evenly over time. The values are called “sample data points.”

downloadable - Said of samples which are loaded from a file into RAM, in contrast to samples which are maintained in ROM.

dry - Refers to audio which has not received any effects processing such as reverb or chorus.

EMU8000 - A wavetable synthesizer chip designed by E-mu Systems for use in Creative Technology products.

envelope - A time varying signal which typically controls the pitch, volume, and/or filter cutoff frequency of a note, and comprises multiple phases including attack, decay, sustain, and release.

flat - A. Said of a tone that is lower in pitch than another reference tone. B. Said of a frequency response that does not deviate significantly from a single fixed gain over the audio range.

interpolator - A circuit or algorithm which computes intermediate points between existing sample data points. This is of particular use in the pitch shifting operation of a wavetable synthesizer, in which these intermediate points represent the output samples of the waveform at the desired pitch transposition.

key number - See MIDI key number.

LFO - Acronym for Low Frequency Oscillator. A slow periodic modulation source.

linear coding - The most common method of encoding amplitudes in digital audio in which each step is of equal size.

loop - In wavetable synthesis, a portion of a sample which is repeated many times to increase the duration of the resulting sound.

loop points - The sample data points at which a loop begins and ends.

lowpass - Said of a filter which attenuates high frequencies but does not attenuate low frequencies.

MIDI - Acronym for Musical Instrument Digital Interface. The standard protocol for sending performance information to a musical synthesizer.

MIDI bank - A group of up to 128 presets selected by a MIDI “change bank” command.

MIDI continuous controller - A construct in the MIDI protocol.

MIDI key number - A construct in the MIDI protocol which accompanies a MIDI key-on or key-off command and specifies the key of the musical instrument keyboard to which the command refers.

MIDI pitch bend - A special MIDI construct akin to the MIDI continuous controllers which controls the real-time value of the pitch of all notes played in a MIDI channel.

MIDI preset - A “preset” selected to be active in a particular MIDI channel by a MIDI “change preset” command.

MIDI velocity - A construct in the MIDI protocol which accompanies a MIDI key-on or key-off command and specifies the speed with which the key was pressed or released.

mono - Short for “monophonic.” Indicates a sound comprising only one channel or waveform. Contrast with “stereo.”

oscillator - In wavetable synthesis, the wavetable interpolator is considered an oscillator.

pan - Short for “panorama.” This is the control of the apparent azimuth of a sound source over 180 degrees from left to right. It is generally implemented by varying the volume at the left and right speakers.

pitch - The perceived value of frequency. Generally can be used interchangeably with frequency.

pitch shift - A change in pitch. Wavetable synthesis relies on interpolators to cause pitch shift in a sample to produce the notes of the scale.

pole - A mathematical term used in filter transform analysis. Traditionally in synthesis, a pole is equated with a rolloff of 6dB per octave, and the rolloff of a filter is specified in “poles.”

Preditor - E-mu Systems’ proprietary SoundFont 2.00 compatible bank editing software.

preset - A keyboard full of sound. Typically the collection of samples and articulation data associated with a particular MIDI preset number.

Q - A mathematical term used in filter transform analysis. Indicates the degree of resonance of the filter. In synthesis terminology, it is synonymous with resonance.

release - The portion of an envelope or sound during which the amplitude declines from a steady state to zero value or inaudibility.

resonance - Describes the aspect of a filter in which particular frequencies are given significantly more gain than others. The resonance can be measured in dB above the DC gain.

resonant frequency - The frequency at which resonance reaches its maximum.

reverb - Short for reverberation. In synthesis, a synthetic signal processor which adds artificial spaciousness and ambience to a sound.

sample - This term is often used both to indicate a “sample data point” and to indicate a collection of such points comprising a digital audio waveform. The latter meaning is exclusively used in this specification.

soft - The pedal on a piano, so named because it causes the damper to be lowered in such a way as to soften the timbre and loudness of the notes. In MIDI, continuous controller #66 which behaves in a similar manner. sostenuto - The pedal on a piano which causes the dampers on all keys depressed to be held until the pedal is released. In MIDI, continuous controller #67 which behaves in a similar manner.

2.2合成器术语

articulation - 演奏方法。 调制amplitude（振幅），音高(频率)和音色以产生富有表现力的音符的过程。

artifact - 一种（通常是不受欢迎的）声音事件，可辨别不存在原始声音中。

attack – 起音。声音达到最大幅度的时间，鼓有很快的起音，而弦乐器的起音比较慢。

包络或声音的相位，在此期间amplitude（振幅）从零增加到峰值。

attenuation - 信号的音量或amplitude（振幅）减小。

AWE32 - 最初的Creative Technology Sound Blaster产品，包含EMU8000波表合成器并支持SoundFont标准。

balance - 一种立体声音量控制形式，当控制器居中时，左右声道都处于最大值，当达到两侧极限值时（即最大值或最小值），它只会衰减相反的声道。

bank - （presets)预设的集合。另见MIDI库

chorus - 一种效果处理算法，它包括循环移动信号的音高并将其自身重新混合，以产生时变梳状滤波器，从而产生运动感和饱满感。

效果器的一种,称为合唱效果器.合唱效果是由多个完全相同的声音叠加而成的。

cutoff frequency - 滤波函数的频率，衰减达到指定值。

data points – 采样数据中的单个值集合。有时也称为采样点。对比“sample”。

decay - 包络线或声音中振幅从峰值下降到稳态值的部分。

delay - 包络线或LFO函数的一部分，从按键按下事件开始直到幅度变为非零。

混响延时，延迟。它是一种可控制时间参数的信号处理器。只有在预定的时间值之后才开始延迟，并不能改变事件的实际长度。

DC gain - 系统对静态或零频率信号的放大或衰减程度。

digital audio - 表示为随时间均匀间隔的量化值序列的音频。这些值称为“sample data points.”（采样数据点）。

downloadable - 说的samples是从一个文件加载到RAM，而不是在ROM中维护的样本。

dry - 指没有收到任何效果处理的音频，如混响或合唱。

EMU8000 - 由E-mu Systems设计的用于 Creative Technology 产品的波表合成器芯片。

envelope -包络；包络线。一种时变信号，通常控制音符的音高、音量和/或滤波器cutoff frequency（截止频率），一个包络线包括attack(起音), decay, sustain, and release等多个阶段。

包络是代表声音随时间变化的振幅（或其他参数）的曲线形状。包络线是随时间改变声音的，包络曲线决定了波形的外形走向，包括振幅、频率和音质，是音色编辑里最重要的途径之一。包络线是一种控制信号，它被应用到音色合成的各方面，如音高、滤波器截止频率和整个振幅。

flat - A.指音调比另一参考音低的音调。B.指在音频范围内不明显偏离单个固定增益的频率响应。

interpolator - 计算现有的sample data points（采样数据点）之间的中间点的电路或算法。这在波表合成器的音调移位操作中特别有用，其中这些中间点表示所需的音调变换的波形的输出采样。

key number - 见MIDI key number。

LFO - 低频振荡器的缩写。慢周期调制源。

低频滤波震荡器。通过对声波的低频调整和对振幅的调制，产生音乐表现所需要的振音、震音和颤音效果。它与包络发生器不同，低频震荡器是改变声音的周期特性，而包络发生器则改变声音的时间特性。

linear coding - 数字音频中最常见的编码振幅方法，其中每个步长具有相同的大小。

loop - 在波表合成技术中，一部分采样数据重复多次以增加结束声音的持续时间。

loop points - 循环开始和结束的采样数据点。

lowpass - 用于衰减高频而不衰减低频的滤波器。

MIDI - 乐器数字接口的缩写。用于将乐谱表演信息发送到音乐合成器的标准协议。

MIDI bank - 由MIDI“change bank”命令选择的一组最多128个preset(预设)。

MIDI continuous controller - MIDI协议中的结构。

MIDI key number - MIDI协议中的一个构造，它伴随着MIDI key-on 或 key-off命令，并指定该命令所指向的乐器键盘的key。

MIDI pitch bend - 一种特殊的MIDI结构，类似于MIDI continuous controller，可控制MIDI通道中所有音符的pitch(音高)的实时值。

MIDI preset - 通过MIDI“更改预设”命令选择在特定的MIDI通道中激活的“预设”。

MIDI velocity——MIDI协议中的一种构造，它伴随着MIDI key-on或key-off命令，并指定按下或释放键的速度。

mono - monophonic的缩写。表示只包含一个声道或波形的声音。对比“立体声”。

oscillator - 在波表合成中，波表插值器被认为是oscillator（振荡器）。

pan -“panorama”的缩写。“这是控制声源从左到右超过180度的视方位角。它通常是通过改变左右扬声器的音量来实现的。

称为声相.也叫平衡控制和定位控制，调整立体声声场中某一信号的左右位置,使音乐更增添立体感.

pitch - 音高，频率的直观感知值。一般可与频率互换使用。

pitch shift - 音高的变化。方波合成依赖内插器在样本中引起音高偏移来产生音符的比例缩放值。

pole - 滤波器变换分析中的数学术语。传统上，在合成中，一个pole相当于每倍频程6分贝的衰减，滤波器的衰减在“poles”中指定。

Preditor - E-mu系统的专有的SoundFont 2.00兼容库编辑软件。

preset – 全声键盘。通常是与特定MIDI预设号相关联的采样数据和演奏方法数据的集合。

Q - 用于滤波器变换分析的数学术语。表示滤波器的谐振程度。在合成器术语中，它是resonance（共振）的同义词。

release - 包络线或声音的一部分，在此期间振幅从稳定状态下降到零值或听不见。

释放、尾音，声音包络的最后部分，在其间振幅往往趋于0值。释音时间在包络设定中，释音时间指从波峰回到零点的时间。在动态处理器中，此参数设定处理器停止工作时的速度。

resonance - 描述滤波器的一个方面，其中特定频率的增益明显高于其他频率。共振可在DC gain上以分贝为单位测量。

resonant frequency - 共振达所能到最大的频率。

reverb - reverberation的缩写。在合成技术中，是一种合成信号处理器，它使声音具有人造的空间感和环境感。

混响。最常用信号处理效果之一，声音反射产生声音残留延续的效果。

sample - 这个术语经常用来表示一个“采样数据点”，也用来表示包含数字音频波形的这些点的集合。后一种含义仅在本规范文档中使用。

采样、样本。可将人和自然界的声音录制到音源内部，让键盘或音序器演奏。

soft - 钢琴上的踏板，之所以如此命名，是因为它使阻尼器降低，以软化音色和音量的音符。在MIDI中，(continuous controller)连续控制器#66 以类似的方式工作。（柔音踏板）使音变得柔和、弱、细

sostenuto - 钢琴上的踏板，它使所有按键上的阻尼器都被按住，直到踏板松开。在MIDI中，(continuous controller)连续控制器#67 以类似的方式工作。

三角钢琴（选择延音踏板）：这是一个特殊性能的踏板，需要某个特殊的音保持住延长时值用的踏板，使用机会极少2、立式钢琴（消音踏板）：能使音量大幅减少，用于夜晚练琴用，以此不打扰邻居。注：和左踏板soft不同的是，立式琴的中踏板并不是演奏技巧的一种，是为特定的使用环境和场所设计的

sustain - 钢琴上的踏板，当琴键上的减震器被踩下时，可以防止它们被释放。在MIDI中，以类似方式工作的( continuous controller)连续控制器#64。（延音踏板）：延长音的时值，也就是在手离开琴键的情况下使音保持住

SoundFont - E-mu Systems, Inc .的注册商标，表示E-mu自家生产的符合SoundFont技术规范的文件、数据、合成器、硬件或软件。

SoundFont Compatible - 表示符合SoundFont技术规范的文件、数据、合成器、硬件或软件。

stereo - 字面上表示三个维度。在本规范中，该术语用于表示双声道立体声，表示声音由两个独立的音频通道组成，分别称为左通道和右通道。对比单声道。

synthesis engine - 与特定合成器的信号处理和调制路径相关的硬件和软件。

synthesizer - 能够产生理想的任意音乐声的设备。

tremolo - 声音振幅的周期性变化，通常是通过将低频振荡器应用于最终音量放大器而产生的。

triangular - 一个周期性波形，向上斜坡上升到一个正极限，然后在相反的斜率向下斜坡到对称的负极限。

unpitched - 表示没有特定感知频率的声音。这对于类似噪音的乐器，以及许多音效都是如此。

velocity - 在合成技术中，按下键盘键的速度，通常与弹奏者所发出力道影响成比例。参见 MIDI velocity。

力度。MIDI事件里代表下键力度，即音符的力度。它代表音符的音量特性。在

使用带有力度分层的音色时，一定的力度变化会造成音色的变化。

vibrato - 音高的周期性变化，通常通过将低频振荡器应用于振荡器音调而产生。

volume - 声音的响度或amplitude（振幅），或对该参数的控制。

音量；音量开关。声波振幅的大小；音响系统输出的分贝数。

wavetable - 一种音乐合成技术，其中音乐声音以数学方式记录或计算并存储在内存中，然后以可变速率播放以产生所需的音高。通常使用amplifiers（放大器），filters（滤波器）和效果处理（例如reverb混响和chorus合唱）对产生的声音进行额外的音色调节。

2.3 Parameter Terminologyabsolute - Describes a parameter which gives a definitive real-world value. Contrast to relative.  
additive - Describes a parameter which is to be numerically added to another parameter.  
attenuation - A decrease in volume or amplitude of a signal.  
bipolar - Said of a controller which has a minimum value of -1 and a maximum value of 1. Contrast “unipolar”  
cent - A unit of pitch ratio corresponding to the twelve hundredth root of two, or one hundredth of a semitone,  
approximately 1.000577790.  
centibel - A unit of amplitude ratio corresponding to the two hundredth root of ten, or one tenth of a decibel, approximately 1.011579454.

cutoff frequency - The frequency of a filter function at which the attenuation reaches a specified value.  
decibel - A unit of amplitude ratio corresponding to the twentieth root of ten, approximately 1.122018454.  
octave - A factor of two in ratio, typically applied to pitch or frequency.  
pitch - The perceived value of frequency. Generally can be used interchangeably with frequency.  
pitch shift - A change in pitch. Wavetable synthesis relies on interpolators to cause pitch shift in a sample to produce the  
notes of the scale.  
relative - Describes a parameter which merely indicates an offset from an otherwise established value. Contrast to absolute.  
resonance - Describes the aspect of a filter in which particular frequencies are given significantly more gain than others.  
The resonance can be measured in dB above the DC gain.  
sample rate - The frequency, in Hertz, at which sample data points are taken when recording a sample.  
semitone - A unit of pitch ratio corresponding to the twelfth root of two, or one twelfth of an octave, approximately  
1.059463094.  
sharp - Said of a tone that is higher in pitch than another reference tone.  
timecent - A unit of duration ratio corresponding to the twelve hundredth root of two, or one twelve hundredth of an octave,  
approximately 1.000577790.  
unipolar - Said of a controller which has a minimum value of 0 and a maximum value of 1. Contrast with “bipolar”

2.3参数术语

absolute - 绝对值。 解释为一个给出确定的真实世界值的参数。对比 relative。

additive - 解释为这个参数以数值方式增加到另一个参数值上。

attenuation - 信号音量或振幅的减小。

bipolar - 指最有最小值为-1，最大值为1的控制器。对比“unipolar”

cent - 音高比的单位， 相当于2的1200次平方根, or one hundredth of a semitone，约1.000577790。

centibel - 振幅比的单位，相当于10的200次平方根, or one tenth of a decibel，约1.011579454。

cutoff frequency - 截止频率，滤波函数的频率衰减截至值。

decibel - 分贝，振幅比的单位，相当于10的20次平方根，约为1.122018454。

octave - 八度， 比率为2的系数，通常用于音高或频率。

pitch - 音高, 频率的感知值。一般可与频率互换使用。

pitch shift - 音高的变化。波表合成依赖于内插器在采样数据中引起音高变化来产生

缩放了比例的音符。

relative - 相对值，描述仅指示与其它已建立值之间的偏移量。对比 absolute。

resonance - 谐振，描述滤波器的一个方面，其中特定频率的增益明显高于其他频率。

谐振可在直流增益上以dB为单位测量。

sample rate - 采样率，采样数据点在记录采样时的频率，单位为赫兹。

semitone - 半音, 音高比的单位，约等于2的12次平方根，或一个octave(八度)的12分之1 , 大约为 1.059463094。

sharp - 表示音的音调高于另一参考音的音调。

timecent - 一种持续时间比的单位，对应于2的1200次平方根， 或一个octave(八度)的1200分之1，大约1.000577790。

unipolar - 指最小值为0，最大值为1的控制器。对比“bipolar”

**附加电子音乐术语**

1 Velocity力度。MIDI事件里代表下键力度，即音符的力度。它代表音符的音量特性。在

使用带有力度分层的音色时，一定的力度变化会造成音色的变化。

2 Volume音量；音量开关。声波振幅的大小；音响系统输出的分贝数。

3 notes音符。MIDI事件之一，参数主要包括音符键位号（Key）即音高、（Velocity）力度、长度（Duration）即时值。

4 Tempo速度。每分钟演奏（演唱）的拍数。

5 meter节拍。在一些乐谱软件界面中，意为指定节拍，即乐曲的拍号。

6 Plug-in插件程序。一种可扩展软件功能的外挂式程序。

7 Chorus 效果器的一种,称为合唱效果器.合唱效果是由多个完全相同的声音叠加而成的。

8 Limiter 限幅器.是一种常规效果器.达到限制波的振幅的作用.它既可以处理低电平信号,而只限制所设置电平的信号,以防止发声失真.

9 Compressor 效果器的一种,称为压缩效果器.使用后是声音听上去更加结实,集中.它用来限制信号的电平,缩小音频信号的动态范围.,并自动调整为合适的水平.

10 Hardware 硬件.用于数据处理中的物理设备。

11 Mix 混音或者调音台.更好的处理好音频.将不同的音频信号组合在一起,采用单声道或立体声道输出,.

12 PAN 称为声相.也叫平衡控制和定位控制，调整立体声声场中某一信号的左右位置,使音乐更增添立体感.

13 Bypass:旁通。让信号不经过效果等处理装置而直接输出。

14 DFD：Direct From Disk。直接从硬盘读取的技术。

15 Session：录音项目文件，这种文件包含录音项目的详细资料，但不包含实际的波形文件。

16 Sequence：音序器。电脑音序软件或专业的硬件音序器，能够记录、编辑和播放midi数据序列。

17 Transport：传送、输送。控制栏区域。控制放音、录音等。

18 Sample：采样、样本。可将人和自然界的声音录制到音源内部，让键盘或音序器演奏。

19 Normalize:标准化。一种声音的数字处理功能，通常是增加声音的振幅到最大值。

20 Pattern：样板，节奏型，节奏方式。一般用以制作乐曲里的某一个多轨片断，多用于打击乐声部的制作上，播放时会循环反复。音序器的乐句片段样板最多可达几十小节或更多。

21 Transpose：移调。有整体移调和局部移调两种。整体移调一般出现在系统设定菜单里，它影响整个乐器的音高；局部移调被包含在音色或音序器的编辑菜单里，只影响被编辑的音色或声部。

22 Note limit: 键盘分区。将键盘划分为几个区，形成不同的力度、音色。

23 Pad:音色垫，背景音色。

24 Monitor：监听设备。

25 ADAT：多轨数码磁带录音机。ADAT的数字传输接口的规格。通常有8轨，可与外部MIDI

设备同步录音，有并轨、音轨复制等功能。

26 Program Change 音色变换，通过合成器从一种乐器声音变为另一种的MIDI信号。程序改变信息，对应于指定音色的一种MIDI信息。当合成器接收到一个程序改变信息，对应的音色就被选中。

27 Modulation 颤音，调制。此专业词为调节，改变的意思。在合成器中，为改变预制程序的时值。通常改变时值包括：滑轮调节，键盘的力度与触后调节等。

28 Release释放、尾音，声音包络的最后部分，在其间振幅往往趋于0值。释音时间在包络设定中，释音时间指从波峰回到零点的时间。在动态处理器中，此参数设定处理器停止工作时的速度。

29 Attack 起音。声音达到最大幅度的时间，鼓有很快的起音，而弦乐器的起音比较慢。在压缩器和噪声门当中，起音表示处理器开始影响增益的快慢。

30 Tone 构成音色的基本单位，它包含波形生成器、TVF、TVA、包络控制和低频振荡器。

31 Element因子，同“TONE”，在YAMAHA和KORG里称为Element。

32 Bounce:(=Bouncing) 并轨。音频软件里的常用语。将多个音轨的所有事件合并到或生成为一个单一的音频文件。

33 Import:导入，如从外部调入MIDI文件或音频文件等。34 Export:导出，如从软件内部生成出MIDI文件或音频文件等。

35 Input:输入.

36 Output:输出。

37 Event:MIDI事件。任何能被MIDI设备接收记录的消息。这些事件按时间先后排列，互相独立。

38 Quantize:(=Quant,Quantiz)量化。按最接近的参照值凑整或者修剪时间的参数值。在乐曲编辑的模式下，参照指定的时钟精度和量化百分比使MIDI事件就近对齐。

39 STEP 在MIDI制作中分步录音。可以不在节拍内或不用指定速度进行录音

40 EQ 均衡器。可对各段频率进行调整。

41 REVERB混响。最常用信号处理效果之一，声音反射产生声音残留延续的效果。

42 DELAY 混响延时，延迟。它是一种可控制时间参数的信号处理器。只有在预定的时间值之后才开始延迟，并不能改变事件的实际长度。

43 FLANGER镶边效果。使声音产生回旋转动的感觉。它是由多个同样的声音叠置而成的，这个术语来自早期的磁带录音技术。

44 Filter:滤波器。从声音的信号频谱中除去所选择的频率，从而调整音色。或者说，它是用来消除或减少音频信号中某些不必要频率的装置或电路。根据其截止频率可以分为低通、高通、带通和带阻四种。

45 Envelope：包络；包络线。包络是代表声音随时间变化的振幅（或其他参数）的曲线形状。包络线是随时间改变声音的，包络曲线决定了波形的外形走向，包括振幅、频率和音质，是音色编辑里最重要的途径之一。包络线是一种控制信号，它被应用到音色合成的各方面，如音高、滤波器截止频率和整个振幅。

46 Amplifier：功放；扩音机；放大器。一种用来增强信号强度（振幅）的电子管或半导体设备。

47 OSC：振荡器。一种产生循环波形的电子装置，是声音发生器的重要组成部分，是波形编辑的重要工具。震荡器的主要功能是调整音高。

48 LFO：低频滤波震荡器。通过对声波的低频调整和对振幅的调制，产生音乐表现所需要的振音、震音和颤音效果。它与包络发生器不同，低频震荡器是改变声音的周期特性，而包络发生器则改变声音的时间特性。

49 Glide：滑音。电子乐器中的一个特殊功能，使演奏的音从一个音高平稳滑到另一个音高。

50 Master：主要的，总的，标准的，主导装置。合成器或音源的面板按钮，用来设定系统状态，包括主调音、移调、输出控制、MIDI系统设定等。

51 VST：VST(Virtual Studio Technology)虚拟录音棚技术。由Steinberg为数字音频工作站(DAW)软件开发的效果处理引擎插件（plug-in）。可以使用于兼容VST的音频录音/编辑主软件，音频和MIDI效果处理，合成器和采样器模块，母带处理工具，实时多轨数字音频等应用。

52 ASIO：ASIO的全称是Audio Stream Input Output，就是音频流输入输出接口的意思。通常这是专业声卡或高档音频工作站才会具备的性能。它可以通过软件效果器，即时的进行效果的处理，但同时不破坏原有信号。采用ASIO技术可以减少系统对音频流信号的延迟，增强声卡硬件的处理能力

53 RTAS：RTAS (Real-Time AudioSuite)protools插件格式，以主机为依托的处理器，它主要利用电脑的处理功能来工作，主要在实时处理时占有优势，并且在系统允许的限定内，分配到多个音轨上使用。

54 MUTE：哑音，静音，在进行多轨的音频或midi编辑时，根据需要使某一轨或某几轨静音。

55 Channel：通道，midi用于发送和接收的16个离散数据通路之一，一个端口（port）有16个通道。

56 Effect：效果。常缩写为FX，利用回声或其他人工的方法来增加音响逼真或音场效果

57 Distortion :失真效果器。利用电子回路超过饱和量所产生的一种音效，通称用于电吉他的solo

58 ARPEGGIO:琶音器，这是MIDI编曲器的一种功能，只需按下琴键或按键，就可以照着预定的音符序列，产生出一串旋律或节奏。

59 GATE:门限，噪声门。门限以上的声音可以通过，门限以下的声音都被视为噪声而被切掉。

60 S-PDIF:S/PDIF的全称是Sony/PhilipsDigital Interface Format,民用数字音频格式标准，大量的消费类音频数字产品如民用CD机、DAT、MD机、计算机声卡数字口等都支持S/PDIF，在不少专业设备上也有该标准的接口,S/PDIF的普通物理连接媒质主要是采用捆紧式/光学波导连接设备。

61 Quick punch:快速切入切出录音方式，在这种常用的录音方式下，PROTOOLS工作站规定，实际可以用于同时录音的轨数是其开通的虚拟轨的一半。

62 Surround：环绕声

63 GAIN：增益。用来增加或减少信号的强度。

64 Preset：在插件内预先设定、设置的参数。

3 RIFF Structure  
**3.1 General RIFF File Structure**The RIFF (Resource Interchange File Format) is a tagged file structure developed for multimedia resource files, and is described in some detail in the Microsoft Windows SDK Multimedia Programmer’s Reference. The tagged-file structure is useful because it helps prevent compatibility problems which can occur as the file definition changes over time. Because each piece of data in the file is identified by a standard header, an application that does not recognize a given data element can skip over the unknown information.  
A RIFF file is constructed from a basic building block called a “chunk.” In ‘C’ syntax, a chunk is defined:  
typedef DWORD FOURCC; // Four-character code  
typedef struct {  
FOURCC ckID; // A chunk ID identifies the type of data within the chunk.  
DWORD ckSize; // The size of the chunk data in bytes, excluding any pad byte.  
BYTE ckDATA[ckSize]; // The actual data plus a pad byte if req’d to word align.  
};  
Two types of chunks, the “RIFF” and “LIST” chunks, may contain nested chunks called sub-chunks as their data.  
The ordering requirements of chunks and sub-chunks within a RIFF file is not well documented in the RIFF file format. In SoundFont 2.0, the order of the sub-chunks within the INFO chunk is arbitrary, but for consistency it is recommended that the sub-chunks be ordered as presented in this document. The order of the all other chunks and sub-chunks is strictly defined and must be maintained as presented in this document.

3 RIFF 结构

3.1通用RIFF文件结构

riff（resource interchange file format，资源交换文件格式）是为多媒体资源文件开发的一种标记文件结构，在Microsoft Windows SDK Multimedia Programmer’s Reference中有详细描述。标记文件结构是很有用，因为它有助于防止在文件定义随时间变化时可能出现的兼容性问题。由于文件中的每一条数据都由标准头标识，因此对于应用程序中一段无法识别的给定数据元素可以跳过这些未知信息。

RIFF文件是由称为“chunk”的基础构造数据块组成的。在“c”语法中，这个chunk定义如下：

typedef DWORD FOURCC; // 4字节编码

typedef struct {  
FOURCC ckID; // chunk ID 定义了此chunk的数据类型

DWORD ckSize; // chunk实际数据的字节尺寸，不包括任何pad辅助对齐字节。  
BYTE ckDATA[ckSize + pad]; // 如果需要字对齐，实际数据加上一个pad字节};

};

“RIFF”和“LIST” 是chunk（块）的两种类型，可能包含称为sub-chunks(子块)的嵌套块作为其数据。

RIFF文件中的块和子块的顺序没有很好地记录在RIFF文件中。在SoundFont2.0中，INFO chunk（信息块）中子块的顺序是任意的，但是为了保持一致性，建议按照本文档中的顺序排列子块。所有其他块和子块的顺序都是严格定义的，必须按照本文档中的说明进行维护。

**3.2 The SoundFont 2 Chunks and Sub-chunks**A SoundFont 2 compatible RIFF file comprises three chunks:

an INFO-list chunk containing a number of required and optional sub-chunks describing the file, its history, and its intended use, an sdta-list chunk comprising a single sub-chunk containing any referenced digital audio samples, and a pdta-list chunk containing nine sub-chunks which define the articulation of the digital audio data.  
The SoundFont 2 standard allows that the sub-chunks within the INFO-list chunk may appear in arbitrary order. However, the order of the three chunks, and the order of the sub-chunks within the pdta-list chunk, is fixed. The SoundFont 2 specification requires that implementations ignore unknown sub-chunks within the INFO-list chunk. Note, however, that until such sub-chunks become defined in the specification, inclusion of additional INFO-list sub-chunks will preclude the file from conforming to the SoundFont standard.  
A detailed description of the SoundFont 2 RIFF structure is provided in Section 4.

**3.2 SoundFont2的块和子块**

一个SOUNDFONT2兼容的RIFF文件包含三个块：

INFO-list chunk（信息列表块），其中包含了许多必需和可选的子块用来描述这个RIFF文件，

这些子块是一些历史记录和预期用途的。

一个sdta-list 列表块(SDTA)，其中包含一个子块，子块中包含任何已引用的数字音频样本；

以及一个pdta-list列表块，包含定义数字音频数据演奏方法的九个子块。

SoundFont2标准允许INFO-list chunk（信息列表块）中的子块以任意顺序出现。但是，这三个块的顺序以及pdta-list列表块中子块的顺序是固定的。SoundFont2规范要求实现忽略信息列表块中的未知子块。但是请注意，在规范中定义这些子块之前，包含其他信息列表子块将阻止文件符合SoundFont标准。

第4节提供了SoundFont 2 RIFF结构的详细说明。

**3.3 Redundancy and Error Handling in the RIFF structure**The RIFF file structure contains redundant information regarding the length of the file and the length of the chunks and subchunks. This fact enables any reader of a SoundFont compatible file to determine if the file has been damaged by loss of  
data.  
If any such loss is detected, the SoundFont compatible file is termed “structurally unsound” and in general should be  
rejected. SoundFont compatible software developers may produce utilities to recover data from structurally unsound files,  
producing with or without user assistance a corrected and structurally sound SoundFont 2compatible file

3.3 RIFF结构中的冗余和错误处理

RIFF文件结构包含有关文件长度以及块和子块的长度的冗余信息。 此事实使任何与SoundFont兼容的文件的读写器都可以确定文件是否已因数据丢失而损坏。

如果检测到任何此类丢失，则与SoundFont兼容的文件被称为“结构上不健全”，通常应

被拒绝。 与SoundFont兼容的软件开发人员可能会开发实用工具，以从结构不健全的文件中恢复数据，无论是否需要用户帮助，都可以生成经过校正且结构合理的SoundFont 2兼容文件。

**4 SoundFont 2 RIFF File Format**

**4** **SoundFont 2 RIFF文件格式**

**4.1 SoundFont 2 RIFF File Format Level 0**

<SFBK-form> -> RIFF (‘sfbk’ ; RIFF form header

{

<INFO-list> ; Supplemental Information

<sdta-list> ; The Sample Binary Data

<pdta-list> ; The Preset, Instrument, and Sample Header data

}

)

**4.2 SoundFont 2 RIFF File Format Level 1**<INFO-list> -> LIST (‘INFO’  
{  
<ifil-ck> ; Refers to the version of the Sound Font RIFF file  
SoundFont 2.01 Technical Specification - Page 13 - Printed 12/10/1996 5:57 PM  
<isng-ck> ; Refers to the target Sound Engine  
<INAM-ck> ; Refers to the Sound Font Bank Name  
[<irom-ck>] ; Refers to the Sound ROM Name  
[<iver-ck>] ; Refers to the Sound ROM Version  
[<ICRD-ck>] ; Refers to the Date of Creation of the Bank  
[<IENG-ck>] ; Sound Designers and Engineers for the Bank  
[<IPRD-ck>] ; Product for which the Bank was intended  
[<ICOP-ck>] ; Contains any Copyright message  
[<ICMT-ck>] ; Contains any Comments on the Bank  
[<ISFT-ck>] ; The SoundFont tools used to create and alter the bank  
}  
)

<sdta-ck> -> LIST (‘sdta’  
{  
[<smpl-ck>] ; The Digital Audio Samples for the upper 16 bits  
}  
{  
[<sm24-ck>] ; The Digital Audio Samples for the lower 8 bits  
}  
)

<pdta-ck> -> LIST (‘pdta’  
{  
<phdr-ck> ; The Preset Headers  
<pbag-ck> ; The Preset Index list  
<pmod-ck> ; The Preset Modulator list  
<pgen-ck> ; The Preset Generator list  
<inst-ck> ; The Instrument Names and Indices  
<ibag-ck> ; The Instrument Index list  
<imod-ck> ; The Instrument Modulator list  
<igen-ck> ; The Instrument Generator list  
<shdr-ck> ; The Sample Headers  
}  
)

**4.3 SoundFont 2 RIFF File Format Level 2**<ifil-ck> -> ifil(<iver-rec>) ; e.g. 2.01  
<isng-ck> -> isng(szSoundEngine:ZSTR) ; e.g. “EMU8000”  
<irom-ck> -> irom(szROM:ZSTR) ; e.g. “1MGM”  
<iver-ck> -> iver(<iver-rec>) ; e.g. 2.08  
<INAM-ck> -> INAM(szName:ZSTR) ; e.g. “General MIDI”  
<ICRD-ck> -> ICRD(szDate:ZSTR) ; e.g. “July 15, 1997”  
<IENG-ck> -> IENG(szName:ZSTR) ; e.g. “John Q. Sounddesigner”  
<IPRD-ck> -> IPRD(szProduct:ZSTR) ; e.g. “SBAWE64 Gold”  
<ICOP-ck> -> ICOP(szCopyright:ZSTR) ; e.g. “Copyright (c) 1997 E-mu Systems, Inc.”  
<ICMT-ck> -> ICMT(szComment:ZSTR) ; e.g. “This is a comment”  
<ISTF-ck> -> ISFT(szTools:ZSTR) ; e.g. “:Preditor 2.00a:Vienna SF Studio 2.0:”  
<smpl-ck> -> smpl(<sample:SHORT>) ; 16 bit Linearly Coded Digital Audio Data  
<phdr-ck> -> phdr(<phdr-rec>)  
<pbag-ck> -> pbag(<pbag-rec>)  
<pmod-ck> -> pmod(<pmod-rec>)  
<pgen-ck> -> pgen(<pgen-rec>)  
<inst-ck> -> inst (<inst -rec>)  
<ibag-ck> -> ibag(<ibag-rec>)  
<imod-ck> -> imod(<imod-rec>)  
<igen-ck> -> igen(<igen-rec>)  
<shdr-ck> -> shdr(<shdr-rec>)

**4.4 SoundFont 2 RIFF File Format Level 3**<iver-rec> -> struct sfVersionTag  
{  
WORD wMajor;  
WORD wMinor;  
};  
<phdr-rec> -> struct sfPresetHeader  
{  
CHAR achPresetName[20];  
WORD wPreset;  
WORD wBank;  
WORD wPresetBagNdx;  
DWORD dwLibrary;  
DWORD dwGenre;  
DWORD dwMorphology;  
};  
<pbag-rec> -> struct sfPresetBag  
{  
WORD wGenNdx;  
WORD wModNdx;  
};  
<pmod-rec> -> struct sfModList  
{  
SFModulator sfModSrcOper;  
SFGenerator sfModDestOper;  
SHORT modAmount;  
SFModulator sfModAmtSrcOper;  
SFTransform sfModTransOper;  
};  
<pgen-rec> -> struct sfGenList  
{  
SFGenerator sfGenOper;  
genAmountType genAmount;  
};  
<inst-rec> -> struct sfInst  
{  
CHAR achInstName[20];  
WORD wInstBagNdx;  
};  
  
<ibag-rec> -> struct sfInstBag  
{  
WORD wInstGenNdx;  
WORD wInstModNdx;  
};  
<imod-rec> -> struct sfInstModList  
{  
SFModulator sfModSrcOper;  
SFGenerator sfModDestOper;  
SHORT modAmount;  
SFModulator sfModAmtSrcOper;  
SFTransform sfModTransOper;  
};  
<igen-rec> -> struct sfInstGenList  
{  
SFGenerator sfGenOper;  
genAmountType genAmount;  
};  
<shdr-rec> -> struct sfSample  
{  
CHAR achSampleName[20];  
DWORD dwStart;  
DWORD dwEnd;  
DWORD dwStartloop;  
DWORD dwEndloop;  
DWORD dwSampleRate;  
BYTE byOriginalKey;  
CHAR chCorrection;  
WORD wSampleLink;  
SFSampleLink sfSampleType;  
};

**4.5 SoundFont 2 RIFF File Format Type Definitions**The sfModulator, sfGenerator, and sfTransform types are all enumeration types whose values are defined in subsequent  
sections.  
The genAmountType is a union which allows signed 16 bit, unsigned 16 bit, and two unsigned 8 bit fields:  
typedef struct  
{  
BYTE byLo;  
BYTE byHi;  
} rangesType;  
typedef union  
{  
rangesType ranges;  
SHORT shAmount;  
WORD wAmount;  
} genAmountType;

The SFSampleLink is an enumeration type which describes both the type of sample (mono, stereo left, etc.) and the whether the sample is located in RAM or ROM memory:  
typedef enum  
{  
monoSample = 1,  
rightSample = 2,  
leftSample = 4,  
linkedSample = 8,  
RomMonoSample = 0x8001,  
RomRightSample = 0x8002,  
RomLeftSample = 0x8004,  
RomLinkedSample = 0x8008  
} SFSampleLink;

**4.1 SoundFont 2 RIFF 文件格式层级0**

<SFBK-form> -> RIFF (‘sfbk’ ; RIFF form header  
{  
 <INFO-list> ; 补充信息

<sdta-list> ; 二进制采样数据

<pdta-list> ; 预设、乐器和采样数据头

}）

**4.1 SoundFont 2 RIFF 文件格式层级1**

<INFO-list> -> LIST (‘INFO’  
{  
<ifil-ck> ; 指SoundFont RIFF文件的版本  
<isng-ck> ; 指目标音频的引擎

<INAM-ck> ; 指soundfont音色库名称

[<irom-ck>] ; 指声音ROM名称  
[<iver-ck>] ; 指声音ROM版本

[<ICRD-ck>] ; 指音色库生成的日期  
[<IENG-ck>] ; 指音色库的音频的设计师或工程师名字

[<IPRD-ck>] ; 音色库的知识产权

[<ICOP-ck>] ; 包含任何版权相关信息

[<ICMT-ck>] ; 包含任何对音色库的注释信息

[<ISFT-ck>] ; 指用于创建和修改音色库SoundFont工具名称

}  
)

<sdta-ck> -> LIST (‘sdta’  
{  
[<smpl-ck>] ; 16位以上的数字音频采样

}  
{  
[<sm24-ck>] ; 低8位的数字音频样本}  
)

<pdta-ck> -> LIST (‘pdta’  
{  
<phdr-ck> ; 预设头

<pbag-ck> ; 预设的 Index列表

<pmod-ck> ; 预设调制器列表  
<pgen-ck> ; 预设生成器列表  
<inst-ck> ; 乐器的名称和索引

<ibag-ck> ; 乐器的Index 列表  
<imod-ck> ; 乐器的调制器列表  
<igen-ck> ; 乐器的生成器列表  
<shdr-ck> ; 采样头

}  
)

**4.3 SoundFont 2 RIFF 文件格式层级2**

<ifil-ck> -> ifil(<iver-rec>) ; e.g. 2.01  
<isng-ck> -> isng(szSoundEngine:ZSTR) ; e.g. “EMU8000”  
<irom-ck> -> irom(szROM:ZSTR) ; e.g. “1MGM”  
<iver-ck> -> iver(<iver-rec>) ; e.g. 2.08  
<INAM-ck> -> INAM(szName:ZSTR) ; e.g. “General MIDI”  
<ICRD-ck> -> ICRD(szDate:ZSTR) ; e.g. “July 15, 1997”  
<IENG-ck> -> IENG(szName:ZSTR) ; e.g. “John Q. Sounddesigner”  
<IPRD-ck> -> IPRD(szProduct:ZSTR) ; e.g. “SBAWE64 Gold”  
<ICOP-ck> -> ICOP(szCopyright:ZSTR) ; e.g. “Copyright (c) 1997 E-mu Systems, Inc.”  
<ICMT-ck> -> ICMT(szComment:ZSTR) ; e.g. “This is a comment”  
<ISTF-ck> -> ISFT(szTools:ZSTR) ; e.g. “:Preditor 2.00a:Vienna SF Studio 2.0:”  
<smpl-ck> -> smpl(<sample:SHORT>) ; 16 bit Linearly Coded Digital Audio Data  
<phdr-ck> -> phdr(<phdr-rec>)  
<pbag-ck> -> pbag(<pbag-rec>)  
<pmod-ck> -> pmod(<pmod-rec>)  
<pgen-ck> -> pgen(<pgen-rec>)  
<inst-ck> -> inst (<inst -rec>)  
<ibag-ck> -> ibag(<ibag-rec>)  
<imod-ck> -> imod(<imod-rec>)  
<igen-ck> -> igen(<igen-rec>)  
<shdr-ck> -> shdr(<shdr-rec>)

**4.4 SoundFont 2 RIFF 文件格式层级3**

<iver-rec> -> struct sfVersionTag  
{  
WORD wMajor;  
WORD wMinor;  
};  
<phdr-rec> -> struct sfPresetHeader  
{  
CHAR achPresetName[20];  
WORD wPreset;  
WORD wBank;  
WORD wPresetBagNdx;  
DWORD dwLibrary;  
DWORD dwGenre;  
DWORD dwMorphology;  
};  
<pbag-rec> -> struct sfPresetBag  
{  
WORD wGenNdx;  
WORD wModNdx;  
};  
<pmod-rec> -> struct sfModList  
{  
SFModulator sfModSrcOper;  
SFGenerator sfModDestOper;  
SHORT modAmount;  
SFModulator sfModAmtSrcOper;  
SFTransform sfModTransOper;  
};  
<pgen-rec> -> struct sfGenList  
{  
SFGenerator sfGenOper;  
genAmountType genAmount;  
};  
<inst-rec> -> struct sfInst  
{  
CHAR achInstName[20];  
WORD wInstBagNdx;  
};  
<ibag-rec> -> struct sfInstBag  
{  
WORD wInstGenNdx;  
WORD wInstModNdx;  
};  
<imod-rec> -> struct sfInstModList  
{  
SFModulator sfModSrcOper;  
SFGenerator sfModDestOper;  
SHORT modAmount;  
SFModulator sfModAmtSrcOper;  
SFTransform sfModTransOper;  
};  
<igen-rec> -> struct sfInstGenList  
{  
SFGenerator sfGenOper;  
genAmountType genAmount;  
};  
<shdr-rec> -> struct sfSample  
{  
CHAR achSampleName[20];  
DWORD dwStart;  
DWORD dwEnd;  
DWORD dwStartloop;  
DWORD dwEndloop;  
DWORD dwSampleRate;  
BYTE byOriginalKey;  
CHAR chCorrection;  
WORD wSampleLink;  
SFSampleLink sfSampleType;  
};

**4.5 SoundFont 2 RIFF文件格式类型定义**

sfModulator，sfGenerator和sfTransform类型都是枚举类型，其值在后续各节中定义。  
genAmountType 是一个union类型，它允许有符号16位、无符号16位和两个无符号8位字段：

typedef struct  
{  
BYTE byLo;  
BYTE byHi;  
} rangesType;

typedef union  
{  
rangesType ranges;  
SHORT shAmount;  
WORD wAmount;  
} genAmountType;

SFSampleLink是一种枚举类型，它描述采样的类型（单声道，左立体声等）以及采样是位于RAM还是ROM内存中：  
typedef enum  
{  
monoSample = 1,  
rightSample = 2,  
leftSample = 4,  
linkedSample = 8,  
RomMonoSample = 0x8001,  
RomRightSample = 0x8002,  
RomLeftSample = 0x8004,  
RomLinkedSample = 0x8008  
} SFSampleLink;

**5 The INFO-list Chunk**The INFO-list chunk in a SoundFont 2 compatible file contains three mandatory and a variety of optional sub-chunks as defined below. The INFO-list chunk gives basic information about the SoundFont compatible bank that is contained in the file.

**5 The INFO-list Chunk**

SoundFont2兼容文件中的INFO-list Chunk(信息列表块)包含三个强制性的子块和多种可选子块，定义如下。INFO-list chunk信息列表块提供了相关SoundFont兼容库文件中数据的基本信息。

**5.1 The ifil Sub-chunk**The ifil sub-chunk is a mandatory sub-chunk identifying the SoundFont specification version level to which the file complies. It is always four bytes in length, and contains data according to the structure:  
struct sfVersionTag  
{  
WORD wMajor;  
WORD wMinor;  
};  
The WORD wMajor contains the value to the left of the decimal point in the SoundFont specification version, the WORD wMinor contains the value to the right of the decimal point. For example, version 2.11 would be implied if wMajor=2 and wMinor=11.  
These values can be used by applications which read SoundFont compatible files to determine if the format of the file is usable by the program. Within a fixed wMajor, the only changes to the format will be the addition of Generator, Source and Transform enumerators, and additional info sub-chunks. These are all defined as being ignored if unknown to the program.  
Consequently, many applications can be designed to be fully upward compatible within a given wMajor. In the case of editors or other programs in which all enumerators should be known, the value of wMinor may be of consequence.  
Generally the application program will either accept the file as usable (possibly with appropriate transparent translation), reject the file as unusable, or warn the user that there may be uneditable data in the file.  
If the ifil sub-chunk is missing, or its size is not four bytes, the file should be rejected as structurally unsound.

### 5.1 The ifil Sub-chunk（库文件版本 子块）

### ifil子块是一个强制性子块，用于标识文件所遵循的SoundFont规范版本级别。 它始终为四个字节，并依据结构包含数据：

struct sfVersionTag  
{  
WORD wMajor;  
WORD wMinor;  
};

在SoundFont规范版本中，字wMajor包含小数点左边的值，字wMinor包含小数点右边的值。 例如，如果wMajor = 2和wMinor = 11，则将隐含版本2.11。

这些值可供读取SoundFont兼容文件的应用程序使用，以确定程序是否可以使用该文件的格式。在固定的wMajor中，对格式的唯一更改是添加Generator(生成器)、Source(源)和Transform(转换) 这些枚举器以及附加的info sub-chunks(信息子块)。如果这些对程序是未知的，则这些定义都将被忽略掉。因此，许多应用程序可以设计为在给定的wMajor中完全向上兼容。在编辑器或其他程序中，所有枚举器都应该是已知的，因此wMinor 的值很重要的。通常，应用程序在文件可用时则接受，不可用时则拒绝，或者警告用户文件数据不可编辑。

如果文件缺少ifil sub-chunk子块，或者它的大小不是四个字节，那么应该拒绝该文件，因为它的结构不健全。

**5.2 The isng Sub-chunk**The isng sub-chunk is a mandatory sub-chunk identifying the wavetable sound engine for which the file was optimized. It contains an ASCII string of 256 or fewer bytes including one or two terminators of value zero, so as to make the total byte count even. The default isng field is the eight bytes representing “EMU8000” as seven ASCII characters followed by a zero byte.  
The ASCII should be treated as case-sensitive. In other words “emu8000” is not the same as “EMU8000.”  
The isng string can be optionally used by chip drivers to vary their synthesis algorithms to emulate the target sound engine.  
If the isng sub-chunk is missing, or is not terminated with a zero valued byte, or its contents are an unknown sound engine,  
the field should be ignored and EMU8000 assumed.

### 5.2 The isng Sub-chunk （音频引擎 子块）

isng子块是一个强制性子块，用于标识针对其文件进行了优化的波表声音引擎。 它包含一个256个或更少字节的ASCII字符串，包括一个或两个值为零的字节终止符，以使总字节数为偶数。 默认的isng字段是8个字节，代表“ EMU8000”，为七个ASCII字符，后跟一个零字节。

ASCII应区分大小写。 换句话说，“ emu8000”与“ EMU8000”不同。

芯片驱动程序可以选择使用isng字符串来更改其合成算法，以模拟目标音频引擎。

如果isng子块丢失，或者没有以零值字节终止，或者其内容是未知的声音引擎，

则该字段应被忽略，并假定为EMU8000。

**5.3 The INAM Sub-chunk**The INAM sub-chunk is a mandatory sub-chunk providing the name of the SoundFont compatible bank. It contains an ASCII string of 256 or fewer bytes including one or two terminators of value zero, so as to make the total byte count even.  
A typical INAM sub-chunk would be the fourteen bytes representing “General MIDI” as twelve ASCII characters followed by two zero bytes.  
The ASCII should be treated as case-sensitive. In other words “General MIDI” is not the same as “GENERAL MIDI.”  
The inam string is typically used for the identification of banks even if the file names are altered.  
If the inam sub-chunk is missing, or not terminated in a zero valued byte, the field should be ignored and the user supplied with an appropriate error message if the name is queried. If the file is re-written, a valid name should be placed in the INAM field.

**5.3 The INAM Sub-chunk （声音库名称 子块）**

INAM子块是提供SOUNDFONT兼容库名称的强制子块。它包含一个256个或更少字节的ASCII字符串，包括一个或两个值为零的终止符，以便使总字节数为偶数。典型的INAM子块是14个字节，表示“General MIDI”为12个ASCII字符，后跟2个零字节。

ASCII应区分大小写。换句话说，“General MIDI”与“GENERAL MIDI”不相同。

INAM字符串通常用于识别音源库，即使文件名已更改。

如果INAM子块丢失或未以零值字节终止，则应忽略该字段，如果查询该名称，则将向用户提供相应的错误消息。如果重新写入文件，则应在INAM字段中放置一个有效的名称。

**5.4 The irom Sub-chunk**The irom sub-chunk is an optional sub-chunk identifying a particular wavetable sound data ROM to which any ROM samples refer. It contains an ASCII string of 256 or fewer bytes including one or two terminators of value zero, so as to make the total byte count even. A typical irom field would be the six bytes representing “1MGM” as four ASCII characters followed by two zero bytes.  
The ASCII should be treated as case-sensitive. In other words “1mgm” is not the same as “1MGM.”  
The irom string is used by drivers to verify that the ROM data referenced by the file is available to the sound engine.  
If the irom sub-chunk is missing, not terminated in a zero valued byte, or its contents are an unknown ROM, the field should be ignored and the file assumed to reference no ROM samples. If ROM samples are accessed, any accesses to such intruments should be terminated and not sound. A file should not be written which attempts to access ROM samples without both irom and iver present and valid.

**5.4 The irom Sub-chunk**【音频ROM名称 子块】

 irom sub-chunk是一个可选的子块，用于标识任何ROM采样引用的特定波形声音数据ROM。它包含一个256个或更少字节的ASCII字符串，包括一个或两个值为零的终止符，以便使总字节数为偶数。一个典型的IROM字段是6个字节，表示“1MGM”为4个ASCII字符，后跟2个零字节。

ASCII应区分大小写。换句话说，“1mgm”与“1MGM”不相同。

驱动程序使用irom字符串来验证文件引用的ROM数据是否可用于声音引擎。

如果irom sub-chunk丢失，未以零值字节终止，或其内容是未知的ROM，则应忽略该字段，并假定该文件未引用任何ROM采样。如果ROM采样是可访问的，则应终止对此类乐器的任何访问且不发声。若没有有效的irom 和iver 存在，则文件不应编写试图访问ROM的采样。

**5.5 The iver Sub-chunk**The iver sub-chunk is an optional sub-chunk identifying the particular wavetable sound data ROM revision to which any ROM samples refer. It is always four bytes in length, and contains data according to the structure:  
struct sfVersionTag  
{  
WORD wMajor;  
WORD wMinor;  
};  
The WORD wMajor contains the value to the left of the decimal point in the ROM version. The WORD wMinor contains the value to the right of the decimal point. For example, version 1.36 would be implied if wMajor=1 and wMinor=36.  
The iver sub-chunk is used by drivers to verify that the ROM data referenced by the file is located in the exact locations specified by the sound headers.  
If the iver sub-chunk is missing, not four bytes in length, or its contents indicate an unknown or incorrect ROM, the field should be ignored and the file assumed to reference no ROM samples. If ROM samples are accessed, any accesses to such instruments should be terminated and not sound. Note that for ROM samples to function correctly, both iver and irom must be present and valid. A file should not be written which attempts to access ROM samples without both irom and iver present and valid.

### 5.5  iver Sub-chunk 【ROM版本 子块】

iver sub-chunk是一个可选的子块，用于标识任何ROM采样所引用的特定可波形声音数据ROM版本。它的长度始终为四个字节，并根据结构包含数据：

struct sfVersionTag  
{  
WORD wMajor;  
WORD wMinor;  
};

WORD wMajor 包含ROM版本中小数点左边的值。 WORD wMinor 包含小数点右边的值。例如，如果wMajor=1且wMinor=36，则示意版本1.36。

驱动程序使用iver sub-chunk子块来验证文件引用的ROM数据是否位于声音头指定的确切位置。如果缺少iver sub-chunk子块，长度不是4个字节，或者其内容指示未知或不正确的ROM，则应忽略该字段，并假定该文件未引用任何ROM采样。如果ROM采样被访问，则应终止对此类仪器的任何访问且不可发声。请注意，要使ROM样本正常工作，必须同时存在有效的iver 和irom。若没有有效的irom 和iver 存在，则不应编写试图访问ROM采样的文件。

**5.6 The ICRD Sub-chunk**

The ICRD sub-chunk is an optional sub-chunk identifying the creation date of the SoundFont compatible bank. It contains an ASCII string of 256 or fewer bytes including one or two terminators of value zero, so as to make the total byte count even. A typical ICRD field would be the twelve bytes representing “May 1, 1995” as eleven ASCII characters followed by a zero byte.

Conventionally, the format of the string is “Month Day, Year” where Month is initially capitalized and is the conventional full English spelling of the month, Day is the date in decimal followed by a comma, and Year is the full decimal year. Thus the field should conventionally never be longer than 32 bytes.

The ICRD string is provided for library management purposes.

If the ICRD sub-chunk is missing, not terminated in a zero valued byte, or for some reason incapable of being faithfully copied as an ASCII string, the field should be ignored and if re-written, should not be copied. If the field’s contents are not seemingly meaningful but can faithfully reproduced, this should be done.

**5.6 The ICRD Sub-chunk （库创建日期 子块）**

icrd子块是一个可选的子块，用于标识soundfont兼容库的创建日期。它包含256个或更少字节的ascii字符串，包括一个或两个值为零的终止符，以便使总字节数为偶数。典型的icrd字段是12个字节，表示“1995年5月1日”为11个ascii字符，后跟一个0字节。

通常，字符串的格式是“Month Day, Year”，其中Month最初是大写的，是常规完整英文拼写；Day 是小数点后接逗号的日期；Year 是完整的十进制年份。因此，字段通常不应超过32个字节。

ICRD字符串是为库管理目的而提供的。

如果ICRD子块丢失，未以零值字节终止，或由于某些原因无法作为ASCII字符串如实复制，则应忽略该字段，如果重新写入，则不应复制该字段。如果该字段的内容看起来没有意义，但可以如实复制，那可以这么做。

5.7 The IENG Sub-chunk

The IENG sub-chunk is an optional sub-chunk identifying the names of any sound designers or engineers responsible for the SoundFont compatible bank. It contains an ASCII string of 256 or fewer bytes including one or two terminators of value zero, so as to make the total byte count even. A typical IENG field would be the twelve bytes representing “Tim Swartz” as ten ASCII characters followed by two zero bytes.

The IENG string is provided for library management purposes.

If the IENG sub-chunk is missing, not terminated in a zero valued byte, or for some reason incapable of being faithfully copied as an ASCII string, the field should be ignored and if re-written, should not be copied. If the field’s contents are not seemingly meaningful but can faithfully reproduced, this should be done.

5.7 The IENG Sub-chunk （设计师或工程师的名字 子块）

ieng子块是一个可选的子块，用于标识负责SoundFont兼容库的任何声音设计师或工程师的名称。它包含一个256个或更少字节的ASCII字符串，包括一个或两个值为零的终止符，以便使总字节数为偶数。典型的IENG字段是12个字节，表示“Tim Swartz”为10个ASCII字符，后跟2个零字节。

IENG 字符串是为库管理目的而提供的。

如果缺少IENG sub-chunk子块，未以零值字节终止，或由于某种原因无法被忠实地复制为ASCII字符串，则应忽略该字段，如果重新写入，则不应复制该字段。如果该字段的内容看起来没有意义，但可以如实复制，则可以如实复制。

5.8 The IPRD Sub-chunk

The IPRD sub-chunk is an optional sub-chunk identifying any specific product for which the SoundFont compatible bank is intended. It contains an ASCII string of 256 or fewer bytes including one or two terminators of value zero, so as to make the total byte count even. A typical IPRD field would be the eight bytes representing “SBAWE32” as seven ASCII characters followed by a zero byte.

The ASCII should be treated as case-sensitive. In other words “sbawe32” is not the same as “SBAWE32.”

The IPRD string is provided for library management purposes.

If the IPRD sub-chunk is missing, not terminated in a zero valued byte, or for some reason incapable of being faithfully copied as an ASCII string, the field should be ignored and if re-written, should not be copied. If the field’s contents are not seemingly meaningful but can faithfully reproduced, this should be done.

5.8 The IPRD Sub-chunk （知识产权 子块）

IPRD sub-chunk子块是一个可选的子块，用于标识SoundFont兼容库所针对的任何特定产品。它包含一个256个或更少字节的ASCII字符串，包括一个或两个值为零的终止符，以便使总字节数为偶数。典型的IPRD字段是8个字节，将“SBAWE32”表示为7个ASCII字符，后跟一个零字节。

IPRD字符串是为库管理目的而提供的。

如果缺少IPRD sub-chunk子块，未以零值字节终止，或者由于某些原因无法作为ASCII字符串如实复制，则应忽略该字段，如果重新写入，则不应复制该字段。如果该字段的内容看起来没有意义，但可以如实复制，则可以如实地复制。

5.9 The ICOP Sub-chunk

The ICOP sub-chunk is an optional sub-chunk containing any copyright assertion string associated with the SoundFont compatible bank. It contains an ASCII string of 256 or fewer bytes including one or two terminators of value zero, so as to make the total byte count even. A typical ICOP field would be the 40 bytes representing “Copyright (c) 1995 E-mu Systems, Inc.” as 38 ASCII characters followed by two zero bytes.

The ICOP string is provided for intellectual property protection and management purposes.

If the ICOP sub-chunk is missing, not terminated in a zero valued byte, or for some reason incapable of being faithfully copied as an ASCII string, the field should be ignored and if re-written, should not be copied. If the field’s contents are not seemingly meaningful but can faithfully reproduced, this should be done.

5.9 The ICOP Sub-chunk (版权 子块)

ICOP sub-chunk子块是一个可选的子块，包含与SoundFont兼容库关联的所有版权声明字符串。它包含一个256个或更少字节的ASCII字符串，包括一个或两个值为零的终止符，以便使总字节数为偶数。典型的ICOP字段是40个字节，代表“Copyright (c) 1995 E-mu Systems, Inc.”，即38个ASCII字符，后跟两个零字节。

ICOP字符串是为知识产权保护和管理目的而提供的。

如果缺少ICOP sub-chunk子块，未以零值字节终止，或由于某些原因无法作为ASCII字符串如实复制，则应忽略该字段，如果重新写入，则不应复制该字段。如果该字段的内容看起来没有意义，但可以如实复制，则可以如实地复制。

5.10 The ICMT Sub-chunk

The ICMT sub-chunk is an optional sub-chunk containing any comments associated with the SoundFont compatible bank. It contains an ASCII string of 65,536 or fewer bytes including one or two terminators of value zero, so as to make the total byte count even. A typical ICMT field would be the 40 bytes representing “This space unintentionally left blank.” as 38 ASCII characters followed by two zero bytes.

The ICMT string is provided for any non-scatological uses.

If the ICMT sub-chunk is missing, not terminated in a zero valued byte, or for some reason incapable of being faithfully copied as an ASCII string, the field should be ignored and if re-written, should not be copied. If the field’s contents are not seemingly meaningful but can faithfully reproduced, this should be done.

5.10 The ICMT Sub-chunk （库注释 子块）

ICMT sub-chunk子块是一个可选的子块，包含与SoundFont兼容库相关的所有注释。它包含一个小于等于65536字节的ASCII字符串，包括一个或两个值为零的终止符，以便使总字节数为偶数。一个典型的icmt字段是40个字节，代表“This space unintentionally left blank”作为38个ASCII字符，后跟两个零字节。

ICMT 字符串用于任何非污秽低俗内容用途。

如果ICMT sub-chunk子块丢失，没有以零值字节终止，或者由于某些原因不能作为ASCII字符串忠实地复制，则应忽略该字段，如果重新写入，则不应复制该字段。如果该字段的内容看起来没有意义，但可以如实复制，则可以如实地复制。

5.11 The ISFT Sub-chunk

The ISFT sub-chunk is an optional sub-chunk identifying the SoundFont compatible tools used to create and most recently modify the SoundFont compatible bank. It contains an ASCII string of 256 or fewer bytes including one or two terminators of value zero, so as to make the total byte count even. A typical ISFT field would be the thirty bytes representing “Preditor 2.00a:Preditor 2.00a” as twenty-nine ASCII characters followed by a zero byte.

The ASCII should be treated as case-sensitive. In other words “Preditor” is not the same as “PREDITOR.”

Conventionally, the tool name and revision control number are included first for the creating tool and then for the most recent modifying tool. The two strings are separated by a colon. The string should be produced by the creating program with a null modifying tool field (e.g. “Preditor 2.00a:), and each time a tool modifies the bank, it should replace the modifying tool field with its own name and revision control number.

The ISFT string is provided primarily for error tracing purposes.

If the ISFT sub-chunk is missing, not terminated in a zero valued byte, or for some reason incapable of being faithfully copied as an ASCII string, the field should be ignored and if re-written, should not be copied. If the field’s contents are not seemingly meaningful but can faithfully reproduced, this should be done.

5.11 The ISFT Sub-chunk （音源创建修改工具 子块）

ISFT sub-chunk子块是一个可选的子块，用于标识用于创建和最近修改SoundFont兼容库的工具。它包含一个256个或更少字节的ASCII字符串，包括一个或两个值为零的终止符，以便使总字节数为偶数。典型的ISFT字段是表示“Preditor 2.00a:Preditor 2.00a”的30个字节，即29个ASCII字符，后跟一个零字节。

ASCII应视为区分大小写。换句话说，“Preditor”与“PREDITOR”不同。

通常，首先为创建工具包含工具名和修订控制号，然后为最新的修改工具包含工具名和修订控制号。这两个字符串由冒号分隔。字符串应由具有空的修改工具字段（例如“Preditor 2.00a:”）的创建程序生成，并且在工具每次修改库时，应使用其自己的名称和修订控制号替换修改工具字段。

ISFT字符串主要用于错误跟踪。

如果缺少ISFT sub-chunk子块，未以零值字节终止，或由于某些原因无法作为ASCII字符串如实复制，则应忽略该字段，如果重新写入，则不应复制该字段。如果该字段的内容看起来没有意义，但可以如实复制，则可以如实地复制。

**6 The sdta-list Chunk**

The sdta-list chunk in a SoundFont 2 compatible file contains a single optional smpl sub-chunk which contains all the RAM based sound data associated with the SoundFont compatible bank. The smpl sub-chunk is of arbitrary length, and contains an even number of bytes. The sm24 sub-chunk, if present, is exactly ½ the size of the smpl sub-chunk, plus 1 byte if necessary to meet the RIFF 16-bit alignment specification.

**6 The sdta-list Chunk**

SoundFont2兼容文件中的SDTA列表块包含一个可选的smpl 子块，该子块包含与SoundFont兼容库关联的所有基于RAM的声音数据。smpl 子块的长度是任意的，并且包含偶数个字节。如果存在sm24 子块，则其尺寸正好是smpl 子块的½ ，如果需要，则加上1个字节，以满足RIFF 16位对齐规范。

6.1 Sample Data Format in the smpl Sub-chunk

The smpl sub-chunk, if present, contains one or more “samples” of digital audio information in the form of linearly coded sixteen bit, signed, little endian (least significant byte first) words. Each sample is followed by a minimum of forty-six zero valued sample data points. These zero valued data points are necessary to guarantee that any reasonable upward pitch shift using any reasonable interpolator can loop on zero data at the end of the sound.

1. SMPL子块中的采样数据格式

SMPL子块（如果存在）包含一个或多个数字音频信息的“采样”，其形式为线性编码的16位有符号小尾数（最低有效字节优先）字。每个采样后面至少有46个零值样本数据点。这些零值数据点是必要的，以保证插值器进行合理地音高偏移都可以在声音结束时对零数据进行循环。

6.2 Sample Data Format in the sm24 Sub-chunk

The sm24 sub-chunk, if present, contains the least significant byte counterparts to each sample data point contained in the smpl chunk. Note this means for every two bytes in the [smpl] sub-chunk there is a 1-byte counterpart in [sm24] sub-chunk.

These sample waveform points are to be combined with the sample waveform points in the smpl sub-chunk, to collectively create a single sample data pool with up to 24 bits of resolution.

If the smpl Sub-chunk is not present, the sm24 sub-chunk should be ignored. If the ifil version of the format is less than that which represents 2.04, the sm24 sub-chunk should be ignored. If the size of the sm24 chunk is not exactly equal to the ½ the size of the smpl chunk (+ 1 byte in the case that ½ the size of smpl chunk is an odd value), the sm24 sub-chunk should be ignored.

In any and all cases where the sm24 sub-chunk is ignored, the synthesizer should render only those samples contained within the smpl sub-chunk.

2. SM24子块中的采样数据格式

sm24 子块（如果存在）与smpl 块中包含的每个采样数据点最低有效字节相对应。注意，这意味着对于[smpl]子块中的每两个字节，[sm24]子块中都有一个1字节的对应项。

这些采样波形点将与smpl 子块中的采样波形点组合，以共同创建一个分辨率高达24位的单个采样数据池。

如果smpl子块不存在，则应忽略sm24子块。如果该格式的ifil字段版本小于表示2.04的版本，则应忽略sm24子块。如果sm24块的大小不完全等于smpl块的½大小（如果smpl块的½大小为奇数字节，则+1字节），则应忽略sm24子块。

在忽略sm24子块的任何的情况下，合成器应该只渲染包含在smpl子块中的那些采样。

6.3 Sample Data Looping Rules

Within each sample, one or more loop point pairs may exist. The locations of these points are defined within the pdta-list chunk, but the sample data points themselves must comply with certain practices in order for the loop to be compatible across multiple platforms.

The loops are defined by “equivalent points” in the sample. This means that there are two sample data points which are logically equivalent, and a loop occurs when these points are spliced atop one another. In concept, the loop end point is

never actually played during looping; instead the loop start point follows the point just prior to the loop end point. Because of the bandlimited nature of digital audio sampling, an artifact free loop will exhibit virtually identical data surrounding the equivalent points.

In actuality, because of the various interpolation algorithms used by wavetable synthesizers, the data surrounding both the loop start and end points may affect the sound of the loop. Hence both the loop start and end points must be surrounded by continuous audio data. For example, even if the sound is programmed to continue to loop throughout the decay, sample data points must be provided beyond the loop end point. This data will typically be identical to the data at the start of the loop. A minimum of eight valid data points are required to be present before the loop start and after the loop end.

The eight data points (four on each side) surrounding the two equivalent loop points should also be forced to be identical. By forcing the data to be identical, all interpolation algorithms are guaranteed to properly reproduce an artifact-free loop.

6.3 采样数据循环规则

在每个采样中，可能存在一个或多个循环点对。这些点的位置是在pdta-list chunk列表块中定义的，但是采样数据点本身必须符合某些规则，以便循环在多个平台上兼容。

循环由样本中的“等效点”来定义。这意味着有两个逻辑上相等的采样数据点，当这些点相互重叠时，就会出现循环。在概念上，循环终点是循环过程中从未实际播放过；相反，循环开始点位于循环结束点之前。由于数字音频采样的带限特性，无缝环路将在等效点周围显示几乎相同的数据。

实际上，由于波形表合成器使用的各种插值算法，环路起点和终点周围的数据都会影响环路的声音。因此，循环的起点和终点都必须被连续的音频数据包围。例如，即使声音被编程为在整个衰减过程中继续循环，也必须在循环终点之外提供采样数据点。这些数据通常与循环开始时的数据相同。在循环开始之前和循环结束之后，至少需要存在八个有效数据点。

围绕两个等效循环点的八个数据点（每侧四个）也应强制相同。通过强制数据相同，所有的插值算法都可以保证正确地重现一个无缝的循环。

**7 The pdta-list Chunk**

7.1 The HYDRA Data Structure

The articulation data within a SoundFont 2 compatible file is contained in nine mandatory sub-chunks. This data is named “hydra” after the mythical nine-headed beast. The structure has been designed for interchange purposes; it is not optimized for either run-time synthesis or for on-the-fly editing. It is reasonable and proper for SoundFont compatible client programs to translate to and from the hydra structure as they read and write SoundFont compatible files.

7.1 Hydra数据结构

SoundFont 2兼容文件中的发音方法数据包含在九个子块中。这个数据以神话中的九头兽命名为“hydra(九头蛇)”。该结构是为格式交换而设计的；它没有针对运行时合成或动态编辑进行优化。SoundFont兼容的客户端程序在读取和写入SoundFont兼容文件时，可以合理地互相转换到Hydra数据结构。

7.2 The PHDR Sub-chunk

The PHDR sub-chunk is a required sub-chunk listing all presets within the SoundFont compatible file. It is always a multiple of thirty-eight bytes in length, and contains a minimum of two records, one record for each preset and one for a terminal record according to the structure:

struct sfPresetHeader

{

CHAR achPresetName[20];

WORD wPreset;

WORD wBank;

WORD wPresetBagNdx;

DWORD dwLibrary;

DWORD dwGenre;

DWORD dwMorphology;

};

The ASCII character field achPresetName contains the name of the preset expressed in ASCII, with unused terminal characters filled with zero valued bytes. Preset names are case sensitive. A unique name should always be assigned to each preset in the SoundFont compatible bank to enable identification. However, if a bank is read containing the erroneous state of presets with identical names, the presets should not be discarded. They should either be preserved as read or preferably uniquely renamed.

The WORD wPreset contains the MIDI Preset Number and the WORD wBank contains the MIDI Bank Number which apply to this preset. Note that the presets are not ordered within the SoundFont compatible bank. Presets should have a unique set of wPreset and wBank numbers. However, if two presets have identical values of both wPreset and wBank, the first occurring preset in the PHDR chunk is the active preset, but any others with the same wBank and wPreset values should be maintained so that they can be renumbered and used at a later time. The special case of a General MIDI percussion bank is handled conventionally by a wBank value of 128. If the value in either field is not a valid MIDI value of zero through 127, or 128 for wBank, the preset cannot be played but should be maintained.

The WORD wPresetBagNdx is an index to the preset’s zone list in the PBAG sub-chunk. Because the preset zone list is in the same order as the preset header list, the preset bag indices will be monotonically increasing with increasing preset headers. The size of the PBAG sub-chunk in bytes will be equal to four times the terminal preset’s wPresetBagNdx plus four. If the preset bag indices are non-monotonic or if the terminal preset’s wPresetBagNdx does not match the PBAG subchunk size, the file is structurally defective and should be rejected at load time. All presets except the terminal preset must have at least one zone; any preset with no zones should be ignored.

The DWORDs dwLibrary, dwGenre and dwMorphology are reserved for future implementation in a preset library management function and should be preserved as read, and created as zero.

The terminal sfPresetHeader record should never be accessed, and exists only to provide a terminal wPresetBagNdx with which to determine the number of zones in the last preset. All other values are conventionally zero, with the exception of achPresetName, which can optionally be “EOP” indicating end of presets.

If the PHDR sub-chunk is missing, or contains fewer than two records, or its size is not a multiple of 38 bytes, the file should be rejected as structurally unsound.

7.2 The PHDR Sub-chunk (预设头列表 子块)

phdr子块是必需的子块，是列出了SoundFont兼容文件中的所有预设的一个列表。它始终是38字节长度的倍数，并且至少包含两个记录（每条记录38个字节），即包括每个预设作为一个记录和一个终止记录的至少这两条记录，预设记录的结构如下：

struct sfPresetHeader

{

CHAR achPresetName[20];

WORD wPreset;

WORD wBank;

WORD wPresetBagNdx;

DWORD dwLibrary;

DWORD dwGenre;

DWORD dwMorphology;

};

ASCII字符字段achPresetName包含以ASCII表示的预设名称，其中未使用的终止字符用零值字节填充。预设名称区分大小写。应始终为SoundFont兼容库中的每个预设指定一个唯一的名称，以启用标识。但是，若读取的库包含具有相同预设名称的错误状态时，不应丢弃这些预设。它们应该保留为已读，或者最好是重新指定唯一的重命名。

WORD wPreset包含MIDI预设号，WORD wBank包含适用于此预设的MIDI库号。请注意，预设在SoundFont兼容库中是没有顺序的。预设应该有一组唯一的wPreset和wBank编号。但是，如果两个预设具有相同的wPreset和wBank值，则PHDR块中第一个出现的预设是激活预设，但应保留具有相同wBank和wPreset值的任何其他预设，以便在以后重新编号和使用。通常wBank的值为128时的通用MIDI打击乐库需要进行特殊处理。若wBank和wPreset任一字段中的值不是0到127的有效MIDI值，或者wBank值为128，则预设应该不被播放，但应该被保留。

WORD wPresetBagNdx是PBAG sub-chunk子块中预设区域列表的索引。由于预设区域列表(PBAG sub-chunk list)与预设头列表(PHDR Sub-chunk list)的顺序相同，预设包indices将随着预设头的增加而依次增加。PBAG子块的大小（以字节为单位）将等于预设头中终止预设这条记录的wPresetBagNdx值乘以4 + 4个字节数。如果预设包indices是非依次的，或者预设头中终止预设记录的wPresetBagNdx与PBAG subchunk子块大小不匹配，则认为该文件在结构上有缺陷，应在加载时拒绝。除终止预设外，所有预设必须至少有一个区域；任何没有区域的预设都应被忽略。

DWORDs dwLibrary、dwGenre和dwMorphology在预设的库管理功能中为将来的实现而保留，并应保留为可读，默认值为零。

终止sfPresetHeader记录不应被访问，它的存在只是为了提供一个终止的wPresetBagNdx，用于确定上一个预设中的区域数。除AchPresetName之外，所有其他值通常为零，可以选择“EOP”来表示预设结束。

如果PHDR sub-chunk子块丢失，或者包含的记录少于两个，或者它的大小不是38字节的倍数，那么应该拒绝该文件，因为它的结构不健全。

7.3 The PBAG Sub-chunk

The PBAG sub-chunk is a required sub-chunk listing all preset zones within the SoundFont compatible file. It is always a multiple of four bytes in length, and contains one record for each preset zone plus one record for a terminal zone according to the structure:

struct sfPresetBag

{

WORD wGenNdx;

WORD wModNdx;

};

The first zone in a given preset is located at that preset’s wPresetBagNdx. The number of zones in the preset is determined by the difference between the next preset’s wPresetBagNdx and the current wPresetBagNdx.

The WORD wGenNdx is an index to the preset’s zone list of generators in the PGEN sub-chunk, and the wModNdx is an index to its list of modulators in the PMOD sub-chunk. Because both the generator and modulator lists are in the same order as the preset header and zone lists, these indices will be monotonically increasing with increasing preset zones. The size of the PMOD sub-chunk in bytes will be equal to ten times the terminal preset’s wModNdx plus ten and the size of the PGEN sub-chunk in bytes will be equal to four times the terminal preset’s wGenNdx plus four. If the generator or modulator indices are non-monotonic or do not match the size of the respective PGEN or PMOD sub-chunks, the file is structurally defective and should be rejected at load time.

If a preset has more than one zone, the first zone may be a global zone. A global zone is determined by the fact that the last generator in the list is not an Instrument generator. All generator lists must contain at least one generator with one exception - if a global zone exists for which there are no generators but only modulators. The modulator lists can contain zero or more modulators.

If a zone other than the first zone lacks an Instrument generator as its last generator, that zone should be ignored. A global zone with no modulators and no generators should also be ignored.

If the PBAG sub-chunk is missing, or its size is not a multiple of four bytes, the file should be rejected as structurally unsound.

7.3 The PBAG Sub-chunk （预设索引列表 子块）

pbag子块是必需的子块，是列出了SoundFont兼容文件中的所有预设区域一个列表。它始终是4字节长度的倍数，并且至少包含两个记录（每条记录4个字节），即包括每个预设区域作为一个记录和一个终止区域记录的至少这两条记录，记录的结构如下：

struct sfPresetBag

{

WORD wGenNdx;

WORD wModNdx;

};

给定预设中的第一个区域位于该预设的wPresetBagNdx。预设中的区域数由下一个预设的wPresetBagNdx和当前wPresetBagNdx之间的差值决定。

WORD wGenNdx是PGEN sub-chunk子块中预设生成器列表的索引，wModNdx是PMOD sub-chunk子块中其调制器列表的索引。由于生成器和调制器列表与预设头和区域列表的顺序相同，因此这些索引将随着预设区域的增加而依次增加。以字节为单位的PMOD sub-chunk子块的大小将等于终止预设区域记录的wModNdx的10倍加上10，以字节为单位的PGEN sub-chunk子块的大小将等于终止预设区域记录的wGenNdx的4倍加上4倍。如果生成器或调制器索引是非依次的，或者与各自的PGEN或PMOD sub-chunks子块的大小不匹配，则该文件在结构上有缺陷，应在加载时被拒绝。

如果预设有多个区域，则第一个区域可能是全局区域。全局区域由列表中的最后一个生成器不是乐器生成器这一事实决定。所有生成器列表必须至少包含一个生成器，但有一个例外-如果存在一个全局区域，而该区域没有生成器，但只有调制器。调制器列表可以包含零个或多个调制器。

如果第一个区域以外的区域缺少乐器生成器作为最后一个生成器，则应忽略该区域。一个没有调制器和生成器的全局区域也应该被忽略。

如果缺少PBAG sub-chunk子块，或者它的大小不是四个字节的倍数，那么应该拒绝该文件，因为它的结构不健全。

7.4 The PMOD Sub-chunk

The PMOD sub-chunk is a required sub-chunk listing all preset zone modulators within the SoundFont compatible file. It is always a multiple of ten bytes in length, and contains zero or more modulators plus a terminal record according to the structure:

struct sfModList

{

SFModulator sfModSrcOper;

SFGenerator sfModDestOper;

SHORT modAmount;

SFModulator sfModAmtSrcOper;

SFTransform sfModTransOper;

};

The preset zone’s wModNdx points to the first modulator for that preset zone, and the number of modulators present for a preset zone is determined by the difference between the next higher preset zone’s wModNdx and the current preset’s wModNdx. A difference of zero indicates there are no modulators in this preset zone.

The sfModSrcOper is a value of one of the SFModulator enumeration type values. Unknown or undefined values are ignored. Modulators with sfModAmtSrcOper set to ‘link’ which have no other modulator linked to it are ignored. This value indicates the source of data for the modulator. Note that this enumeration is two bytes in length.

The sfModDestOper indicates the destination of the modulator. The destination is either a value of one of the SFGenerator enumeration type values or a link to the sfModSrcOper of another modulator block. The latter is indicated by the top bit of the sfModDestOper field being set, the other 15 bits designates the index value of the modulator whose source should be the output of the current modulator RELATIVE TO the first modulator in the instrument zone. Unknown or undefined values are ignored. Modulators with links that point to a modulator index that exceeds the total number of modulators for a given zone are ignored. Linked modulators that are part of circular links are ignored. Note that this enumeration is two bytes in length.

The SHORT modAmount is a signed value indicating the degree to which the source modulates the destination. A zero value indicates there is no fixed amount.

The sfModAmtSrcOper is a value of one of the SFModulator enumeration type values. Unknown or undefined values are ignored. Modulators with sfModAmtSrcOper set to ‘link’ are ignored. This value indicates the degree to which the source modulates the destination is to be controlled by the specified modulation source. Note that this enumeration is two bytes in length.

The sfModTransOper is a value of one of the SFTransform enumeration type values. Unknown or undefined values are ignored. This value indicates that a transform of the specified type will be applied to the modulation source before application to the modulator. Note that this enumeration is two bytes in length.

The terminal record conventionally contains zero in all fields, and is always ignored.

A modulator is defined by its sfModSrcOper, its sfModDestOper, and its sfModSrcAmtOper. All modulators within a zone must have a unique set of these three enumerators. If a second modulator is encountered with the same three enumerators as a previous modulator with the same zone, the first modulator will be ignored.

Modulators in the PMOD sub-chunk act as additively relative modulators with respect to those in the IMOD sub-chunk. In other words, a PMOD modulator can increase or decrease the amount of an IMOD modulator.

In SoundFont 2.00, no modulators have yet been defined, and the PMOD sub-chunk will always consist of ten zero valued bytes.

If the PMOD sub-chunk is missing, or its size is not a multiple of ten bytes, the file should be rejected as structurally unsound

7.4 The PMOD Sub-chunk （预设调制器列表 子块）

pmod子块是必需的子块，是列出soundfont兼容文件中的所有预设区域调制器的列表。它的长度总是10字节的倍数，并且根据结构包含零个或多个调制器和一个终止记录：

struct sfModList

{

SFModulator sfModSrcOper;

SFGenerator sfModDestOper;

SHORT modAmount;

SFModulator sfModAmtSrcOper;

SFTransform sfModTransOper;

};

预设区域的wModNdx指向该预设区域的第一个调制器，而当前预设区域的调制器数量由下一个预设区域的较高wModNdx值和当前预设的wModNdx值之间的差值决定。差为零表示此预设区域中没有调制器。

sfModSrcOper是SFModulator枚举类型值之一的值。忽略未知或未定义的值。将sfModAmtSrcOper设置为“link”且没有其他调制器链接的调制器将被忽略。该值表示调制器的数据源。请注意，此枚举的长度为两个字节。

sfModDestOper指示调制器的目标。目标要么是一个SFGenerator枚举类型值的值，要么是到另一个调制器块的sfModSrcOper的链接。后者由设置的sfModDestOper字段的最高位表示，其他15位表示调制器的索引值，其源应该是当前调制器的输出相对于乐器区中的第一个调制器。忽略未知或未定义的值。调制器的链接指向调制器索引超过给定区域的调制器总数将被忽略。作为循环链接一部分的链接调制器也将被忽略。 请注意，此枚举的长度为两个字节。

SHORT modAmount 是一个有符号的值，表示源对目标进行调制的程度。零值表示没有固定数值。

sfModAmtSrcOper是SFModulator枚举类型值之一的值。忽略未知或未定义的值。将sfModAmtSrcOper设置为“link”的调制器将被忽略。该值表示源对目标调制的程度，由指定的调制源控制。请注意，此枚举的长度为两个字节。

sfModTransOper是SFTransform枚举类型值之一的值。忽略未知或未定义的值。此值表示在应用到调制器之前，将对调制源应用指定类型的转换。请注意，此枚举的长度为两个字节。

终止记录通常在所有字段中都为零，并且总是被忽略。

调制器由它的sfModSrcOper、sfModDestOper和sfModSrcAmtOper定义。一个区域中的所有调制器必须具有这三个枚举器的唯一集合。如果在相同乐器区域中第二个调制器与前一个调制器使用相同的三个枚举器，则第一个调制器将被忽略。

与IMOD子块中的调制器相比，PMOD子块中的调制器充当额外的相对调制器。换句话说，PMOD调制器可以增加或减少IMOD调制器的数量。

在SoundFont 2.00中，还没有定义任何调制器，PMOD子块将始终由10个零值字节组成。

如果PMOD子块缺失，或者其大小不是十个字节的倍数，则应该拒绝该文件在结构上的不合理。

7.5 The PGEN Sub-chunk

The PGEN chunk is a required chunk containing a list of preset zone generators for each preset zone within the SoundFont compatible file. It is always a multiple of four bytes in length, and contains one or more generators for each preset zone (except a global zone containing only modulators) plus a terminal record according to the structure:

struct sfGenList

{

SFGenerator sfGenOper;

genAmountType genAmount;

};

where the types are defined:

typedef struct

{

BYTE byLo;

BYTE byHi;

} rangesType;

typedef union

{

rangesType ranges;

SHORT shAmount;

WORD wAmount;

} genAmountType;

The sfGenOper is a value of one of the SFGenerator enumeration type values. Unknown or undefined values are ignored. This value indicates the type of generator being indicated. Note that this enumeration is two bytes in length.

The genAmount is the value to be assigned to the specified generator. Note that this can be of three formats. Certain generators specify a range of MIDI key numbers of MIDI velocities, with a minimum and maximum value. Other generators specify an unsigned WORD value. Most generators, however, specify a signed 16 bit SHORT value.

The preset zone’s wGenNdx points to the first generator for that preset zone. Unless the zone is a global zone, the last generator in the list is an “Instrument” generator, whose value is a pointer to the instrument associated with that zone. If a “key range” generator exists for the preset zone, it is always the first generator in the list for that preset zone. If a “velocity range” generator exists for the preset zone, it will only be preceded by a key range generator. If any generators follow an Instrument generator, they will be ignored.

A generator is defined by its sfGenOper. All generators within a zone must have a unique sfGenOper enumerator. If a second generator is encountered with the same sfGenOper enumerator as a previous generator with the same zone, the first generator will be ignored.

Generators in the PGEN sub-chunk are applied relative to generators in the IGEN sub-chunk in an additive manner. In other words, PGEN generators increase or decrease the value of an IGEN generator.

If the PGEN sub-chunk is missing, or its size is not a multiple of four bytes, the file should be rejected as structurally unsound. If a key range generator is present and not the first generator, it should be ignored. If a velocity range generator is present, and is preceded by a generator other than a key range generator, it should be ignored. If a non-global list does not end in an instrument generator, zone should be ignored. If the instrument generator value is equal to or greater than the terminal instrument, the file should be rejected as structurally unsound.

7.5 The PGEN Sub-chunk（预设生成器列表 子块 ）

PGEN块是一个必需的块，包含音源兼容文件中每个预设区域的预设区域生成器列表。它始终是四个字节长度的倍数，并且每个预设区域包含一个或多个生成器（除了仅包含调制器的全局区域）以及加一个终止记录，结构如下：

struct sfGenList

{

SFGenerator sfGenOper;

genAmountType genAmount;

};

where the types are defined:

typedef struct

{

BYTE byLo;

BYTE byHi;

} rangesType;

typedef union

{

rangesType ranges;

SHORT shAmount;

WORD wAmount;

} genAmountType;

sfGenOper是SFGenerator枚举类型值之一的值。忽略未知或未定义的值。该值表示所指示的生成器类型。请注意，此枚举的长度为两个字节。

genAmount是要分配给指定生成器的值。请注意，它有三种格式。某些生成器指定MIDI力度的，MIDI键的范围，具有最小值和最大值。其他生成器指定无符号word值。但是，大多数生成器指定一个有符号的16位SHORT值。

预设区域的wGenNdx指向该预设区域的第一个生成器。除非区域是全局区域，否则列表中的最后一个生成器是“乐器”生成器，其值是一个指针指向与该区域关联的乐器。如果预设区域存在“键范围”生成器，则它始终是该预设区域列表中的第一个生成器。如果预设区域存在“力度范围”生成器，则其前面只会有一个键范围生成器。如果任何生成器在乐器生成器之后则它们将被忽略掉。

生成器由其sfGenOper定义。区域中的所有生成器都必须具有唯一的sfGenOper枚举器。如果在相同的区域中，第二个生成器与前一个生成器具有相同的sfGenOper枚举器，则将忽略第一个生成器。

PGEN sub-chunk子块中的生成器相对于IGEN sub-chunk子块中的生成器以添加的方式应用。换句话说，PGEN生成器增加或减少IGEN生成器的值。

如果PGEN sub-chunk子块缺少，或者它的大小不是四个字节的倍数，那么应该拒绝该文件，因为它的结构不健全。如果一个键范围生成器不是第一个生成器，则应忽略它。如果存在力度范围生成器，并且该生成器前面有键范围生成器之外的生成器，则应忽略该生成器。如果非全局列表不以乐器生成器结尾，则应忽略区域。如果乐器生成器值等于或大于终止乐器记录，则文件应被视为结构不健全而拒绝。

7.6 The INST Sub-chunk

The inst sub-chunk is a required sub-chunk listing all instruments within the SoundFont compatible file. It is always a multiple of twenty-two bytes in length, and contains a minimum of two records, one record for each instrument and one for a terminal record according to the structure:

struct sfInst

{

CHAR achInstName[20];

WORD wInstBagNdx;

};

The ASCII character field achInstName contains the name of the instrument expressed in ASCII, with unused terminal characters filled with zero valued bytes. Instrument names are case-sensitive. A unique name should always be assigned to each instrument in the SoundFont compatible bank to enable identification. However, if a bank is read containing the erroneous state of instruments with identical names, the instruments should not be discarded. They should either be preserved as read or preferably uniquely renamed.

The WORD wInstBagNdx is an index to the instrument’s zone list in the IBAG sub-chunk. Because the instrument zone list is in the same order as the instrument list, the instrument bag indices will be monotonically increasing with increasing instruments. The size of the IBAG sub-chunk in bytes will be four greater than four times the terminal (EOI) instrument’s wInstBagNdx. If the instrument bag indices are non-monotonic or if the terminal instrument’s wInstBagNdx does not match the IBAG sub-chunk size, the file is structurally defective and should be rejected at load time. All instruments except the terminal instrument must have at least one zone; any preset with no zones should be ignored.

The terminal sfInst record should never be accessed, and exists only to provide a terminal wInstBagNdx with which to determine the number of zones in the last instrument. All other values are conventionally zero, with the exception of achInstName, which should be “EOI” indicating end of instruments.

If the INST sub-chunk is missing, contains fewer than two records, or its size is not a multiple of 22 bytes, the file should be rejected as structurally unsound. All instruments present in the inst sub-chunk are typically referenced by a preset zone. However, a file containing any “orphaned” instruments need not be rejected. SoundFont compatible applications can optionally ignore or filter out these orphaned instruments based on user preference.

7.6 The INST Sub-chunk (乐器 子块)

inst sub-chunk子块是必需的子块，列出了soundfont兼容文件中的所有乐器。它始终是22字节长度的倍数，并且至少包含两个记录，每个仪器一个记录，根据结构，终止记录一个记录：

struct sfInst

{

CHAR achInstName[20];

WORD wInstBagNdx;

};

ASCII字符字段achInstName包含用ASCII表示的乐器名称，未使用的终端字符用零值字节填充。乐器名称区分大小写。应始终为soundfont兼容库中的每个乐器指定一个唯一的名称，以便进行识别。但是，如果一库被读取时包含同名乐器的错误状态，则不应丢弃这些乐器。它们应该保留为已读，或最好是重命名唯一名称。

WORD wInstBagNdx是IBAG sub-chunk子块中乐器区域列表的索引。由于乐器区域列表与乐器列表的顺序相同，乐器包指数将随着仪表数量的增加而依次增加。以字节为单位的IBAG sub-chunk子块的大小将是终止（EOI）乐器记录的wInstBagNdx值的四倍以上。如果乐器包索引不依次，或者终止（EOI）乐器记录的wInstBagNdx与IBAG sub-chunk子块大小不匹配，则该文件在结构上有缺陷，应在加载时拒绝。除终止（EOI）乐器记录外的所有乐器必须至少有一个区域；任何没有区域的预设都应忽略。