InterPARES 2 Project
International Research on Permanent Authentic Records in Electronic Systems

Focus 1

A Brief Introduction to Digital Music File Formats

Brent Lee
The University of British Columbia

May 2000

Introduction

Data formats designed to represent musical scores, recordings, and other miscellaneous aspects of musical composition (compositional algorithms, synthesizer patches, etc.) have proliferated over the last several decades. While some have (at least for a period of time) been recognized as industry standards, archival records have been generated in a bewildering array of formats specific to particular operating systems (and generations of these systems) and software, in addition to a variety of file interchange formats. This article is intended as an introduction to the state of musical data representation.

The digital representation of music can be broken down into three broad categories. The first category includes file formats that represent actual sound (digital recordings), while the second includes formats that represent musical scores (notation files). A third category includes formats that neither represent a score or recording, but serve to control computer operations that could then generate a score or recording. Each of these categories will be discussed separately.

Digital Recording (Audio) Formats

Audio files have one thing in common: they all contain a stream of numbers that represent changes in the amplitude of sound waves (volume) over time. When a digital recording is made, a recording device measures the amplitude of the sound wave thousands of times each second. Each of these measurements is called a sample. The frequency at which samples are measured is called the sampling rate. The sampling rate is described in samples/second; thus, a sampling rate of 44.1K (used for CDs) means that the sound was recorded 44,100 times each second. Clearly, the higher the sample rate, the better the sound quality, as it gives a more accurate picture of the sound. (Imagine a curve being represented by the numbers 1, 2, 5, 9, 11, 13, 16, 13, 10, 8, 6, 4, 3, 2, 1. If you plotted these numbers on graph paper and connected the dots, you could reconstruct the curve. You could represent the same curve with the numbers 1, 5, 11, 16, 10, 6, 3, 1, but this reconstruction would not be as accurate as the one with more numbers or samples.)

Sampling rate is only one of several variables in audio files. Others include the sample size, the number of channels, the encoding algorithm, the type of compression used (if any), and possibly commands and/or information useful to the operating system for which the file format was developed.

In addition to the sampling rate, the fidelity of a digital sound recording is also dependent on the sample size; the larger the sample size, the more precise the measurement. 8-bit (a scale of 0 to 255) and 16-bit (a scale of 0 to 65535) samples are most common. (Imagine marking an undergraduate term paper out of 3 or out of 100. The larger number allows for a much finer distinction.)

Many audio file formats allow for a variable number of channels. Thus a file could be mono (1), stereo (2), or any number of discreet channels.

Audio files can be encoded in different ways. Most encoding schemes are linear (like PCM), while some are logarithmic (like U-law and A-law). Encoding schemes also vary in their use of signed or unsigned integers. In addition, some file formats (like mp3) use a compression scheme to greatly reduce the size of an audio file. (Consider again the series of numbers I used in the sampling example: 1, 2, 5, 9, 11, 13, 16, 13, 10, 8, 6, 4, 3, 2, 1. This same series could be represented by these numbers which measure the change from one measurement to the next: +1, +3, +4, +2, +3, -3, -3, -2, -2, -2, -1, -1, -1. This second set of numbers is much smaller, and can thus be stored in a smaller file.)

The last variables in an audio file are the system-specific commands or information. File formats that make use of these variables are only useful on certain computers; these variables thus account for the wide array of file formats developed for use within different operating systems, including AIFF (Macintosh), WAV (Windows), U-law (Sun and NeXT), SND (Amiga), and AVR (Atari). Most systems have evolved so that they can easily convert files from one format to another. The differences in file format are normally encoded in a header at the beginning of the file that describes the status of all of the above-mentioned variables.

To some extent, the file format used can help to determine the chronology of files in an archive. Some formats have becoming obsolete as technology improved, and some operating systems have decreased in popularity. For example, if one file was recorded in 8-bit 32K mono, and a second similar file was recorded in 16-bit, 44.1K stereo, the first most likely older than the second.

A more complete description of the plethora of file formats and their technical specifications can be found at the Audio Formats FAQ.[1]

[1] Available at http://home.sprynet.com/~cbagwell/AudioFormats.txt.

Notation formats

File formats that are used to represent the notation of music are graphical in nature, typically using sets of music-character fonts to draw music on a screen and then to print music. Some aspects of music notation (such as phrase markings, beams, and layout) must be calculated by the program in much the same manner as conventional graphics software. Nearly all music notation programs allow for file playback via MIDI.

Numerous programs for the notation of music have been developed for personal computers over the last twenty years. Until recently, file formats were software-specific, although a handful of unsuccessful attempts were made to create a standard interchange format. With the advent of music scanning software and the WWW, a number of new initiatives have appeared in the last decade to establish an accepted file exchange format. There are currently several such formats proposed, the most prominent being:

- NIFF (Notation Interchange File Format, based on Microsoft's RIFF)
- GUIDO (not an acronym, uses ASCII characters in a human-readable way)
- SMDL (Standard Music Description Language, based on SGML [Standard Generalized Markup Language])

Control formats

The third broad category of music-related file formats involves those used and created by various types of music software in the process of creating a recording or score. (Files in this category would nearly all be considered records, while scores and recordings would be considered digital objects.)

The most ubiquitous music file type is the MIDI (Musical Instrument Digital Interface) file. MIDI was developed in the early '80s by synthesizer manufacturers interested in allowing one digital synthesizer to control (play) the synthesized sounds stored in another synthesizer. MIDI can be used in performance situations without the creation of MIDI files; software programs that record and play back performance MIDI data are called sequencers, and the individual MIDI files created are called sequences.

MIDI sequences generally contain less information about a piece of music than a notation file; MIDI sequences usually include only the pitch to be played, its duration (by implication), and its volume. MIDI can also be used to instruct synthesizers to switch from one sound (patch) to another, to add vibrato, sustain pedal, etc. Ultimately, the way a MIDI sequences sounds is entirely dependent on the synthesizer (hardware or software) that receives the MIDI instructions. The same MIDI sequence will thus sound different when played back through different configurations. As such, composers use MIDI as a tool for playing back compositions in progress, or as a component in an audio recording. A very small percentage of composers use MIDI as a medium in itself.

Other control formats are software-specific. They generally fall into four categories: software synthesis, algorithmic composition files, synthesizer patches and samples, and audio editing files.

Software synthesis is the use of a computer's processing power to create digital audio files based on mathematical synthesis methods. (Examples include FM synthesis, additive synthesis, granular synthesis, and FOF synthesis.) A number of higher level programs (CSound, CLM [Common Lisp Music], Cmix) allow the user to specify the synthesis method to use. In each case, a certain number of variables must be defined by the composer; these variables are stored in either text files or software-specific files. (For example, in the creation of a CSound file the composer will specify synthesis variables in a .orc file and event information (similar to MIDI) in a .sco file. Other files may also be used in the synthesis such as samples, filter descriptions, and spectral analyses, each of which is contained in a separate file.)

Algorithmic composition allows a computer to make compositional decisions based on rules predetermined by the composer or by input received during the running of the software. Programmers have attempted to create rules bases for compositions in the style of Palestrina, Bach, Mozart, Bartók, and other composers; other programmer/composers create an individual rules base for each of their own compositions. Once again, these programs require that algorithms and variables be describe in a text file or software-specific file; the output of this type of software can be audio (if coupled with a synthesis program), MIDI, or notation files. Commercial software (such as Band-in-a-Box) incorporate accompaniment styles (saved in a software-specific file format) and user input (meter, chords, tempo, etc., saved in another software-specific file format) in the creation of MIDI files. Algorithmic compositions may pose the most profound problems for archivists, as the programs may rely on formatted audio or MIDI input as well as specific hardware to function.

Synthesizer patches are files that describe the variables needed to recreate a sound on a particular synthesizer. Composers create these patches as part of an electroacoustic composition, and generally use them in conjunction with MIDI files. Obviously, synthesizer patches are useless without the synthesizer for which they were created.

Samples are short audio files used in synthesis, MIDI playback, or as system sounds. Like other audio files, samples comprise a header with information about sample rate, bit-depth, etc. followed by raw audio data. Unlike the digital recordings described above, samples are most often by-products of the compositional process rather than an end in themselves.

In the last category of control files are the numerous files created in the process of editing digital audio files. Most editing programs allow for non-destructive editing of the original audio file; instead, numerous small files describing each edit are created so that the original remains unaltered. These edits might include splices, fade-ins, audio processing (reverb, chorus, etc.). Once again, these files are not products in themselves but files used and created in the process of making a recording.

Conclusions

Composers have different attitudes towards their sketch materials. Some keep everything, some destroy everything, and some keep things as long as they might be of some practical use in the creation of new works. Digital records are in a particular peril, as their utility is inevitably compromised by their rapid obsolescence.