Sound Synthesizers Sound-producing code

A **synthesizer** is the code responsible for interpreting sound commands and using the available hardware to produce sounds. The **Sound Manager** insulates an application from the underlying hardware primarily by selecting the sound hardware drivers (or synthesizers) that are best suited to the available audio hardware.

Synthesizers that drive the built-in audio hardware on Macintosh computers are provided by Apple and are stored in the System file as resources of type 'snth'. Depending on the type of sound you wish to produce, the **Sound Manager** uses either the square-wave synthesizer, the wave-table synthesizer, or the sampled sound synthesizer to interpret commands and data sent to it by your application. Sometimes you do not need to specify which synthesizer you wish to use because some sound commands can be interpreted by all three.

The Square-Wave Synthesizer

The **square-wave synthesizer** is the simplest of the playback synthesizers supplied with the **Sound Manager**. You can use it to generate a sound based on a square wave, and it is functionally equivalent to the square-wave synthesizer contained in the old **Sound Driver**. Your application can use the square-wave synthesizer to play a simple sequence of sounds in which each sound is described completely by three factors: its frequency or pitch, its amplitude (or volume), and its duration.

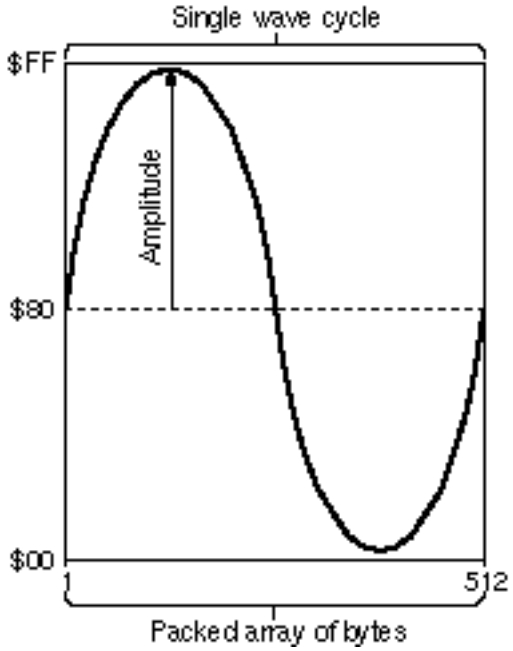
The square-wave synthesizer can play only one frequency at a time through a sound channel, and it cannot play back a waveform sound description or a recorded sound. The square-wave synthesizer requires very little CPU time, however, and it is very easy for your application to use.

The Wave-Table Synthesizer

To produce more complex sounds than are possible using the square-wave synthesizer, your applications can use the **wave-table synthesizer.** As the name indicates, the wave-table synthesizer can produce sounds that are based on a description of a single wave cycle. This cycle is called a *wave table* and is represented as an array of bytes that describe the timbre (or tone) of a sound at any point in the cycle.

Your application can use any number of bytes to represent the wave, but 512 is the recommended number because the **Sound Manager** resizes a wave table to 512 bytes if the table is not exactly that long. Your application can compute the wave table at run time or load it from a resource. You can open multiple wave-table channels (up to a maximum of four).

A **wave table** is a sequence of wave amplitudes measured at fixed intervals. For instance, a sine wave can be converted into a wave table by taking the value of the wave's amplitude at every $1/512$ interval of the wave (see the Figure below).



A graph of a wave table

A wave table is represented as a packed array of bytes. Each byte contains a value in the range 0x00-0xFF. These values are interpreted as offset values, where 0x80 represents an amplitude of 0. The largest negative amplitude is 0x00 and the largest positive amplitude is 0xFF. When playing a wave-table description of a sound, the wave-table synthesizer loops through the wave table for the duration of the sound.

The Sampled Sound Synthesizer

You can use the **sampled sound synthesizer** to play back sounds that have been digitally recorded (that is, sampled) as well as sounds that are computed, possibly at run time. The sampled sound synthesizer is the most widely used of all the available synthesizers, primarily because it is relatively easy to generate a sampled sound and because such samples can contain a wide variety of sounds. The sampled sound synthesizer is typically used to play back pre-recorded sounds such as speech or special sound effects.

You can use the **Sound Manager** to store sounds in one of two ways, either as resources of type 'snd ' or as AIFF or AIFF-C format files. The structure of resources of type 'snd ' is given in **Sound Resources**, and the structure of AIFF and AIFF-C files is given in **Sound Files**. If you simply want to play short prerecorded sampled sounds, you should probably include the sound data in an 'snd ' resource. If you want to allow the user to transfer recorded sound data from one application to another (or from one operating system to another), you should probably store the sound data in an AIFF or AIFF-C file. In certain cases, you need to store sampled sounds in files and not in resources. For example, some sampled sounds might be too large to store as resources; in those cases, you should store the sounds in an AIFF or AIFF-C file.

Regardless of how you store a sampled sound, you can use **Sound Manager** routines to play that sound using the sampled sound synthesizer. If you choose to store sampled sounds in files of type AIFF or AIFF-C, you can play those sounds by calling the **SndStartFilePlay** function. If you store sampled

sounds in resources, your application can play those sounds by passing the **Sound Manager** function **SndPlay** a handle to a resource of type 'snd ' that contains a sampled sound header. (The **SndStartFilePlay** function can also play 'snd ' resources directly from disk.) There are three types of sampled sound headers: the standard sound header (used for monophonic sampled sounds), the extended sound header (used for stereo samples), and the compressed sound header (used for compressed samples, whether monophonic or stereo). The sampled sound header contains information about the sample (such as the original sampling rate, the length of the sample, and so forth), together with an indication of where the sample data is to be found. The data can be stored in a buffer separate from the sound resource or as part of the sound resource at the end of the sound header.

> **Note:** The terminology *sampled sound header* can be confusing because in most cases the sound header (and hence the 'snd ' resource) contains the sound data as well as information describing the data.

You can play a sampled sound at its original rate or play it at some other rate to change its pitch. Thus, once you install a sampled sound header into a channel, it can be played at varying rates to provide a number of pitches and hence can be used as a voice to play a series of sounds.