Playing Sampled Sounds Using **SndPlay**

You can play a sampled sound by calling the **SndPlay** function and passing it a handle to an 'snd ' resource that contains a sampled sound header. To gain greater control over the sound output, you can use a number of sound commands, including bufferCmd, soundCmd, rateCmd, and getRateCmd. Both bufferCmd and soundCmd specify a pointer to a sampled sound header that is locked into memory. The soundCmd command is used to install a sampled sound into a channel as a voice and is discussed in the section entitled **Installing Voices Into Channels**.

To play a sampled sound in one-shot mode (without any looping), use the bufferCmd command. The pointer in the *param2* field of the sound command is the location of a sampled sound header. A bufferCmd command is queued in the channel until the preceding commands have been processed. If the bufferCmd command is contained within an 'snd ' resource, the high bit of the command must be set. If the sound was loaded in from an 'snd ' resource, your application is expected to unlock this resource and allow it to be purged after using it.

You can use the bufferCmd command to handle compressed sound samples in addition to sounds that are not compressed. To expand and play back a buffer of compressed samples, you pass the sampled sound synthesizer a bufferCmd command where *param2* points to a compressed sound header.

> **Note:** Using the bufferCmd command to play several consecutive compressed samples on the Macintosh Plus and the Macintosh SE is not guaranteed to work without an audible pause or click.

To play sampled sounds that are not compressed, pass bufferCmd a standard or extended sound header. The extended sound header is used for stereo sampled sounds. The standard sampled sound header is used for all other noncompressed sampled sounds.

You can divide large sampled sounds into multiple buffers and then issue successive bufferCmd commands to play that sound. In this case, each buffer must contain a sampled sound header. Except as noted just above, the sound will play smoothly, without audible gaps. It is usually much easier, however, to play large sampled sounds from disk by using the play-from-disk routines or the **SndPlayDoubleBuffer** function. See the section entitled **Playing Sampled Sounds From Files** for complete details.

> **Note:** If a sound is playing and you send a bufferCmd command by using **SndDoImmediate**, the sound specified in the bufferCmd command will not play. You should send all bufferCmd commands by using **SndDoCommand**, or else you should first stop the sound that is playing by sending a quietCmd command with **SndDoImmediate**. Alternatively, you can call the **SndChannelStatus** function repeatedly until the scChannelBusy flag of the sound-channel status record turns FALSE.

The **Sound Manager** can play sounds sampled at any rate up to 64 kHz. The Table below lists approximate values for the most common sample rates. When you specify a value in the sampleRate field of a sound header, you should use the values in the third column of the Table.

Table Sample Rates

Rate (kHz)	Rate (Hz)	sampleRate value (Fixed)
5 kHz	5563.6363	0x15BBA2E8
7 kHz	7418.1818	0x1CFA2E8B
11 kHz	11127.2727	0x2B7745D1
22 kHz	22254.5454	0x56EE8BA3
44 kHz	44100.0000	0xAC440000

To store and operate on compressed audio data, you use a structure called the compressed sound header, a record of type **CmpSoundHeader** that is a logical extension of the sound header.

The following code example illustrates the structure of an 'snd ' resource that contains compressed sound data.

Listing: An 'snd ' resource containing compressed sound data

```
data 'snd ' (9004, "Sample 1", purgeable) {
    0x"0001" //format type
    0x"0001" //number of synthesizers
    0x"0005" //resource ID of first synthesizer
    0x"00000380"    //initialization option: initMACE3 + initMono
    0x"0001" //number of sound commands that follow (1)
    0x"8051" //cmd: bufferCmd
    0x"0000" //param1: unused
    0x"00000014"    //param2: offset to sound header (20 bytes)
            //compressed sound header follows:
    0x"00000000"    //pointer to data (it follows immediately)
    0x"00000001"    //number of channels in sample
    0x"56EE8BA3"    //sampling rate of this sound (22 kHz)
    0x"00000000"    //starting of the sample's loop point; not used
    0x"00000000"    //ending of the sample's loop point; not used
    0x"FE"      //compressed sample encoding
    0x"00"          //baseFrequency; not used
    0x"00006590"    //number of frames in sample (26,000)
    0x"400DADDD1745D145826B"          //AIFFSampleRate (22 kHz in
//extended type)
    0x"00000000"    //markerChunk; NULL for 'snd ' resource
    0x"00000000"    //futureUse1; NULL for 'snd ' resource
    0x"00000000"    //futureUse2; NULL for 'snd ' resource
    0x"00000000"    //stateVars; NULL for 'snd ' resource
    0x"00000000"    //leftOverBlockPtr; not used here
    0x"0003" //compressionID, 3 means 3:1
    0x"0010" //packetSize, packetSize for 3:1 is 16 bits
    0x"000B" //snthID is 11
    0x"0008" //sampleSize, sound was 8-bit before processing
    0x"2F 85 81 32 64 87 33 86"           //the compressed sound data
    0x"6F 48 6D 65 72 6B 82 88"
    0x"91 FE 8D 8E 86 4E 7C E9"
```

```
        0x"6F 6D 71 70 7E 79 4F 83"
        0x"59 8F 8F 65" //rest of data omitted in this example
};
```

This resource has the same general structure as the 'snd ' resource illustrated in **The Format 1 'snd ' Resource** in the section entitled **Sound Resources**. The principal difference is that the standard sound header is replaced by the compressed sound header. This example resource specifies a monophonic sound compressed by using the 3:1 compression algorithm.