Playing Sampled Sounds From Files Controlling continuous playback

There are three functions that you can use to initiate and control a continuous playback of sampled sounds stored in files-**SndStartFilePlay**, **SndPauseFilePlay**, and **SndStopFilePlay**. You use **SndStartFilePlay** to initiate the playing of the sound. You use **SndPauseFilePlay** to temporarily suspend a sound from playing. If a sound is playing and you call **SndPauseFilePlay**, then the sound is paused. If the sound is paused and you call **SndPauseFilePlay** again, then the sound resumes playing. Hence, the **SndPauseFilePlay** routine acts like a pause button on a tape player, which toggles the tape between playing and pausing. (You can determine the current state of a play from disk by using the **SndChannelStatus** function. See complete details ,**Obtaining Information About a Single Sound Channel** under the section, **Obtaining Information About Sound Features**.) Finally, you can use **SndStopFilePlay** to stop the file from playing.

SndStartFilePlay can play sampled sounds stored in 'snd ' resources (either format 1 or format 2) or in files that conform to the AIFF or AIFF-C format. In addition, you can specify whether the play from disk should be asynchronous or synchronous. The **SndStartFilePlay** function is a high-level **Sound Manager** routine, like **SndPlay**. If you specify NULL as the sound channel, then **SndStartFilePlay** allocates memory for a channel internally. However, since you must specify a sound-channel pointer when calling either **SndPauseFilePlay** or **SndStopFilePlay**, you must allocate a sound channel yourself and call **SndStartFilePlay** asynchronously if you want to be able to pause or stop the sound prior to its natural ending point.

Playing an 'snd ' Resource From Disk

To play a sampled sound that is contained in an 'snd ' resource, you need to pass **SndStartFilePlay** the resource ID number of the resource to play. The following code example illustrates how to play an 'snd ' resource synchronously from disk.

Notice that the second parameter passed to **SndStartFilePlay** here is set to 0. That parameter is used only when playing files from disk.

```
// Playing an 'snd ' resource from disk


// Assuming inclusion of MacHeaders
#include <Sound.h>

// constants used in routine
#define kTotalSize 16*1024
#define kAsync TRUE          // play sound asynchronously
#define kQuietNow TRUE       // quiet channel now

// Prototype routine like this prior to calling it
void SyncStartFilePlay(short);

void SyncStartFilePlay (short myResNum)
{
    OSErr myErr;
    SndChannelPtr mySndChan;

    // Prototype for error handling routine
    void DoError(OSErr);

    // allocate a sound channel
    mySndChan = nil;
    myErr = SndNewChannel (&mySndChan, sampledSynth, initMono, nil);
    if ( myErr )
        DoError(myErr);

    // play the 'snd ' resource
    myErr = SndStartFilePlay(mySndChan, 0, myResNum, kTotalSize, nil,
                             nil, nil, !kAsync);

    // if error occured handle it
    if ( myErr )
        DoError(myErr);

    // dispose of the channel, if soundchannel was allocated
    if ( mySndChan )
        myErr = SndDisposeChannel(mySndChan, !kQuietNow);

    // if error occured, handle it
    if ( myErr )
        DoError(myErr);
}
```

Playing a File From Disk

To play a sampled sound that is contained in a file, you need to pass **SndStartFilePlay** the file reference number of the file to play. The sample should be stored in either AIFF or AIFF-C format. If the sample is compressed, then it will be automatically expanded during playback.

For example, to play a sampled sound that is found in a file whose file reference number is stored in the variable myfRefNum, you could write

```
myErr = SndStartFilePlay (mySndChan, myfRefNum, 0, kTotalSize, NULL,
NULL,NULL, FALSE);
```

Notice that the third parameter passed to **SndStartFilePlay** here is set to 0. That parameter is used only when playing resources from disk.

Playing Selections

The sixth parameter passed to **SndStartFilePlay** is a pointer to an **AudioSelection** record, which allows you to specify that only part of the sound be played. If that parameter has a value different from NULL, then **SndStartFilePlay** plays only a specified selection of the entire sound. You indicate which part of the entire sound to play by giving two offsets from the beginning of the sound, a time at which to start the selection and a time at which to end the selection. Currently, both time offsets must be specified in seconds.