Manipulating a Sound That Is Playing Changing sounds in progress

The **Sound Manager** provides a number of sound commands that allow you to manipulate sounds currently in progress. You can also pause or stop a sound currently in progress. For information on how to pause the processing of a sound channel see the section entitled **Pausing and Restarting Sound Channels**.

You can use the getRateCmd command to determine the rate at which a sampled sound is currently playing. If **SndDoImmediate** returns noErr when you pass it getRateCmd, the current rate of the channel is returned as a Fixed value in *param2* of the sound command. (As usual, the high bit of the value returned is not interpreted as a sign bit.)

To modify the pitch of the sampled sound currently playing, use the rateCmd command. The current pitch is set to the rate specified in the *param2* field of the sound command. The code example below illustrates how to halve the frequency of a sampled sound.

You can also use rateCmd and getRateCmd to pause a sampled sound that is currently playing. To do this, read the rate at which it is playing, issue a rateCmd command with a rate of 0, and then issue a rateCmd command with the previous rate when you want the sound to resume playing.

```
// Halving the frequency of a sampled sound


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

// used for fixed math operations
#include <FixMath.h>

// Prototype routine like this prior to calling
void HalveFreq (SndChannelPtr);

void HalveFreq (SndChannelPtr mySndChan)
{
    long myRate;
    SndCommand mySndCmd;
    OSErr myErr;

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

    // Initialize sound command record
    mySndCmd.cmd = getRateCmd;
    mySndCmd.param1 = 0;                        // unused
    mySndCmd.param2 = (long)&myRate;

    myErr = SndDoImmediate(mySndChan, &mySndCmd);
    if ( !myErr ) {
        mySndCmd.cmd = rateCmd;
        mySndCmd.param1 = 0;                // unused
        mySndCmd.param2 = FixDiv(myRate, 0x0002000);
        myErr = SndDoImmediate(mySndChan, &mySndCmd);
```

```
    }
    if ( myErr )
        DoError(myErr);
}
```

You can use the getAmpCmd command to determine the current amplitude of a sound in progress. The getAmpCmd command is similar to getRateCmd, except that the value returned is an integer. The value returned is in the range 0-255. Here's an example:

```
short   myAmp;

mySndCmd.cmd = getAmpCmd;
mySndCmd.param1 = 0;                    // unused
mySndCmd.param2 = &myAmp;
myErr = SndDoImmediate(&mySndChan, mySndCmd);
```

To change the amplitude of the sound in progress, issue the ampCmd command. If no sound is currently playing, ampCmd sets the amplitude of the next sound. The desired new amplitude is passed in the *param1* field of the sound command and should be a value in the range 0 to 255.

To modify the timbre of a sound played by the square-wave synthesizer, use the timbreCmd command. A sine wave is specified as 0 in *param1* and produces a very clear sound. A value of 255 in *param1* represents a modified square wave and produces a buzzing sound. You should change the timbre of the square-wave synthesizer before playing the sound. Only a Macintosh with the Apple Sound Chip allows this command to be sent while a sound is in progress.To cause a synthesizer to stop playing the sound in progress, send the quietCmd sound command. Here's an example:

```
mySndCmd.cmd = quietCmd;            //the command is quietCmd
mySndCmd.param1 = 0;                //unused
mySndCmd.param2 = 0;                //unused

//stop the sound now playing
myErr = SndDoImmediate(&mySndChan, mySndCmd);
```

To stop a sound that is currently playing on the specified sound channel, send a quietCmd command. To bypass the command queue, you should issue quietCmd by using **SndDoImmediate**. Any sound commands that are already in the sound channel remain there, however, and further sound commands can be queued in that channel.