Allocating Sound Channels Using low-level routines

To use most of the low-level **Sound Manager** routines, you must specify a sound channel that maintains a queue of commands destined for a particular synthesizer. Generally you do not need to worry about allocating memory for sound channels because the **SndNewChannel** function automatically allocates a sound-channel record in the application's heap if passed a pointer to a NULL sound channel. For example, the following lines of code request that the **Sound Manager** open a new sound channel and link it to the sampled sound synthesizer:

```
mySndChan = nil;
SndNewChannel = myErr(&mySndChan, sampledSynth, 0, nil);
```

If you are concerned with memory management, you can allocate your own channel memory and pass the address of that memory as the first parameter to **SndNewChannel**. The following code example illustrates one way to do this.

Creating a sound channel

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

// Prototype creation function like this prior to calling
SndChannelPtr CreateSndChannel(void);

SndChannelPtr CreateSndChannel()
{
    SndChannelPtr mySndChan;                // pointer to a sound channel
    OSErr myErr;

    // allocate a sound channel
    mySndChan = (SndChannelPtr)NewPtrClear(sizeof(SndChannel));
    if ( mySndChan != nil ) {
        mySndChan->qLength = stdQLength;    //128 sound commands
        myErr = SndNewChannel(&mySndChan, sampledSynth, initMono, nil);
    }
    return mySndChan;                       // return SndChannelPtr
}
```

Note that if you allocate your own channel memory, you must set the size of the sound channel (the number of sound commands that the channel can store). You should set the size by assigning a value to the qLength field of the sound channel you allocate. You can use the constant stdQLength, as illustrated in the Listing above, or provide a value of your own.

```
stdQLength          //default size of standard sound channel
```

Note: The number of sound commands in a channel should be an integer greater than 0. If you open a channel with a 0-length queue,

most of the **Sound Manager** routines will return a badChannel result code.

The second parameter in the **SndNewChannel** function is the resource ID of the play-back synthesizer that is to be linked to the new sound channel. Currently recognized playback synthesizer values are

squareWaveSynth	square-wave synthesizer
waveTableSynth	wave-table synthesizer
sampledSynth	sampled sound synthesizer

The third parameter in the **SndNewChannel** function specifies the initialization parameters to be associated with the new channel. These are discussed in the following section. The fourth parameter in the **SndNewChannel** function is a pointer to a callback procedure. If your application produces sounds asynchronously or needs to be alerted when a command has completed, you can specify a callback procedure by passing the address of that procedure in the fourth parameter. If you pass NULL as the fourth parameter, then no callback routine executes. See **Specifying Callback Routines** for more information on setting up and using callback procedures.