Compressing and Expanding Sounds Making Specific Requests

Some of the capabilities provided by MACE are transparently available to your application. For example, if you pass the **SndPlay** function a handle to an 'snd ' resource that contains a compressed sampled sound, the sampled sound synthesizer automatically expands the sound data for playback in real time. Your application does not need to know whether the 'snd ' resource contains compressed or noncompressed samples when it calls **SndPlay**. This is because sufficient information is in the resource itself to allow the synthesizer to determine whether it should expand the data samples.

However, aside from expansion playback, all of the MACE capabilities need to be specifically requested by your application. For example, you can use the procedures **Comp3to1** or **Comp6to1** if you want to compress a sampled sound (for example, to create an 'snd ' resource containing compressed audio data). And you can use the procedures **Exp1to3** and **Exp1to6** to expand compressed audio data.

All of these procedures require you to specify both an input and an output buffer, from and to which the sampled sound data to be converted is read and written. Your application must allocate the appropriate amount of storage for each buffer. For example, if you want to expand a buffer of compressed sampled sound data by using **Exp1to6**, the output buffer must be at least six times the size of the input buffer.

When calling these routines, you must also specify addresses of two small buffers (128 bytes each) that the **Sound Manager** uses to maintain state information about the compression or expansion process. When you first call a MACE routine, the state buffers should be filled with zeros to initialize the state information. You can pass NULL for both buffers if you do not want to save state information across calls to the MACE routines. The code example below illustrates the use of the **Comp3to1** procedure.

Because the numChannels and the whichChannel parameters are both set to 1, **Comp3to1** compresses monophonic audio data.

```
//  Compressing audio data


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

// Prototype routine like this prior to calling it
void CompressBy3(Ptr,Ptr,long);

void CompressBy3 (Ptr inBuf, Ptr outBuf, long numSamp)
{
    Ptr myInState;   // input state buffer
    Ptr myOutState;  // output state buffer

    // Allocate both state buffers
    myInState = NewPtrClear(128);
    myOutState = NewPtrClear(128);

    // Check that both buffers are non-nil i.e. non-zero
    if ( myInState  && myOutState )
```

```
        // Compress audio data
        Comp3to1(inBuf, outBuf, numSamp, myInState, myOutState, 1, 1);
}
```

THINK Reference © 1991-1992 Symantec Corporation