Some notes on Sound Digitizers since QuickTime 1.0

New Selectors

New selectors have been added to support the new sound input configuration dialogs supplied by the Sequence Grabber. Your sound input driver should support these new selectors if you want the appropriate dialog support.

'sour' — Input Source Selector

If your hardware allows recording from more than one sound source, this selector allows you to get and set the source. The input source is specified by a 2-byte integer ranging from 1 to n, where 1 is the first input source and n is the last.

In response to a _Status call, the driver should return the current input source value. If the driver only supports one input source, it should return an siUnknownInfoType error.

In response to a _Control call, the driver should accept a value specifying the input source to use for all further recording. If the value is less than 1 or greater than the maximum number of input sources, the driver should return paramErr. If the driver only supports one input source, it should return an siUnknownInfoType error.

'snam' — Input Source Names:

If your hardware allows recording from more than one sound source, this selector allows you to get a list of names for the input sources.

In response to a _Status call, the driver should create a handle containing the names of the input sources and return the handle. It is then the responsibility of the caller to dispose of the handle. If the driver is returning a resource handle, be sure to detach it first, so the caller can throw it away.

The format of the handle should be identical to a 'STR#' resource, where the first 2-bytes contain the number of strings followed by the strings themselves. The strings should be packed in the same order as the input sources, so the first string corresponds to input source 1, and so on. If the driver only supports one input source, it should return an siUnknownInfoType error.

'cnam' — Compression Names

If your driver supports compression, this selector allows you to get a list of names for the compression types.

In response to a _Status call, the driver should create a handle containing the names of the compression types and return the handle. It is then the responsibility of the caller to dispose of the handle. If the driver is returning a resource handle, be sure to detach it first, so the caller can throw it away.

The format of the handle should be identical to a 'STR#' resource, where the first 2-bytes contain the number of strings followed by the strings themselves. The strings should be packed in the

same order as the compression types returned by the 'cmav' selector, so the

first string corresponds to compression type 1, and so on. If the driver does not support compression, it should return an siUnknownInfoType error.

'gain' — Input Gain

If your hardware allows adjustment of recording gain, this selector allows you to get and set the gain. The gain is specified by a 4-byte Fixed value ranging from 0.0 to 1.0, where 1.0 is maximum gain.

In response to a _Status call, the driver should return the current gain setting.

In response to a _Control call, the driver should accept a value specifying the gain setting to use for all further recording

Stereo Recording

Many hardware devices support stereo input, so it is important that all the proper selectors are implemented in order for the Sequence Grabber to function correctly. The following selectors should be supported for stereo devices:

'chan' — Number of channels

This selector determines the format of the data stream output by the driver. If 'chan' is set to 1, mono data must be output by the driver in response to a _Read call. If 'chan' is set to 2, interleaved stereo data must be output.

'chac' — Active Channels

This selector determines which stereo channels are active during recording using a bitmap. In combination with 'chan', this determines when stereo data needs to be mixed into mono. Valid settings are:

0x01 — Only record from the left input channel
0x02 — Only record from the right input channel
0x03 — Record from both left and right input channels

'lmac' — Active Level Meters

This selector returns a list of level meter values, one for each channel. The number of values returned is determined by the 'chan' selector. If 'chan' is set to 2, an array of 2 level meter values are returned, one for left and one for right.

Here are some examples:

'chan' = 1
'chac' = 0x03

A mono output stream is produced where the left and right channels of the source are mixed together to form a mono result. This should be the default behavior for mono output if the 'chac' setting is never specified.

'chan' = 1
'chac' = 0x01

A mono output stream is produced containing only samples from the left input channel.

'chan' = 1
'chac' = 0x02

A mono output stream is produced containing only samples from the right input channel.

'chan' = 2
'chac' = 0x03

A stereo output stream is produced containing interleaved samples from the left and right input channels. This should be the default behavior for stereo output if the 'chac' setting is never specified.

'chan' = 2
'chac' = 0x01 or 0x02

This operation is undefined. Just pretend that 'chac' = 0x03 in this case.

Stereo MACE

Many hardware devices support MACE compression, which gets pretty complicated for stereo cases. The big thing to remember is that the packet size for MACE 3:1 is 2-bytes, while the packet size for MACE 6:1 is one byte. This becomes important when interleaving a stereo 3:1 output stream, since you must interleave words, not bytes.

For example, to compress a stereo source using MACE 3:1, you would need to do the following:

```
// compress the left samples to a separate buffer

Comp3to1( source, compressionBuffer, sampleCount,
          leftState, leftState, 2, 1);

// interleave the left compressed samples into the dest

InterleaveWords(compressionBuffer, dest, sampleCount / 6);

// compress the right samples to a separate buffer

Comp3to1( source, compressionBuffer, sampleCount,
          rightState, rightState, 2, 2);
```

```
// interleave the right compressed samples into the dest

InterleaveWords(compressionBuffer, dest+2, sampleCount / 6);
```

The InterleaveWords routine copies every word of the source into every other word of the destination. Notice you must call the compression routine twice, once for each channel, and that you must compress into a separate buffer. This is because the compression routines accept stereo source but produce a mono result, so you must interleave the compressed data into your destination buffer as a separate step. Notice also that the source samples to compress must be a multiple of 6, or the compressor will not work correctly.

The MACE 6:1 steps are similar, but you must interleave bytes instead of words.

16-Bit Sound

Hardware devices that support 16-bit sound should always return samples in twos-complement (signed) format, never offset-binary. The 'twos' selector should always return true.