Re: Can someone explain how to correctly use Low level Digital Audio A

Source:
http://www.tech−archive.net/Archive/Development/microsoft.public.win32.programmer.mmedia/2006−04/msg00011.

- *From*: alanglloyd@xxxxxxx
- *Date*: 8 Apr 2006 11:25:19 −0700

You have to have more than one buffer and associated WaveHdr structure
and provide the memory for them. Here's an outline of waveIn − in
Delphi, but that may be a clearer to show what you have to do at a
higher level. Turn it into whatever language you favour (var declares
the variables, const are constants).

```
const
WaveFmtPCM : TWaveFormatEx = (wFormatTag : WAVE_FORMAT_PCM;
nChannels : 1;
nSamplesPerSec : 8000{11025};
nAvgBytesPerSec : 8000{11025};
nBlockAlign : 1;
wBitsPerSample : 8;
cbSize : 0);
BuffCount = 2;
BuffSize = 800; // 8000hz * 0.1sec
var
HndWaveIn : THandle;
HndWindow : THandle;
WaveBuffs : array[0..BuffCount−1] of PChar;
WaveHdrs : array[0..BuffCount−1] of PWaveHdr;

waveInOpen(@HndWaveIn, WAVE_MAPPER, @WaveFmtPCM,
HndWindow, 0, CALLBACK_WINDOW);
for i := 0 to BuffCount − 1 do begin
WaveBuffs[i] := AllocMem(BuffSize);
WaveHdrs[i] := AllocMem(SizeOf(TWaveHdr));
with WaveHdrs[i]^ do begin
lpData := WaveBuffs[i];
dwBufferLength := BuffSize;
dwUser := integer(Self);
end;
waveInPrepareHeader(HndWaveIn, WaveHdrs[i], SizeOf(TWaveHdr));
waveInAddBuffer(HndWaveIn, WaveHdrs[i], SizeOf(TWaveHdr));
inc(BuffersInUse);
end;
waveInStart(HndWaveIn);
```

All that starts the waveIn recording. It then sends a WIM_DATA message
to the window (handled in the waveInWndProc) with a pointer to the
WaveHdrcalls back to a function. On receipt of that message you must
clear the buffer (savng to a file), reset some flags and call
waveInAddBuffer with the pointer to the waveHdr.

```
procedure WaveInWndProc(var Msg : TMessage);
var
PtrWaveHdr : PWaveHdr;
PtrWaveBuff : PChar;
begin
case Msg.Msg of
WIM_OPEN : ; // 958d $3BE
WIM_CLOSE : ; // 959d $3BF
WIM_DATA : // 960d $3C0
begin
dec(BuffersinUse);
if WaveStopped then
Exit; // finished with level indicator
PtrWaveHdr := PWaveHdr(Msg.LParam);
PtrWaveBuff := PtrWaveHdr^.lpData;
// code to copy buffer to file here
{clear buffer ...}
FillChar(PtrWaveBuff^, BuffSize, #0);
{... set flags}
PtrWaveHdr^.dwFlags := PtrWaveHdr^.dwFlags and WHDR_PREPARED
and not WHDR_DONE;
waveInAddBuffer(HndWaveIn, PtrWaveHdr, SizeOf(TWaveHdr));
dec(BuffersinUse);
end;
end; {case Msg.Msg of}
```

You have to wait after calling waveInReset until all the buffers are
returned (BuffersInUse == 0) and then call waveInClose.

```
WaveStopped := true;
WaveInReset(hndWaveIn);
while BuffersInUse > 0 do
Application.ProcessMessages;;
for i := 0 to BuffCount − 1 do begin
waveInUnPrepareHeader(HndWaveIn, WaveHdrs[i], SizeOf(TWaveHdr));
FreeMem(WaveBuffs[i]);
FreeMem(WaveHdrs[i]);
end;
WaveInClose(hndWaveIn);
```

wavOut operates in a similar fashion except that you fill the buffers
from your file, and re−fill them when you get a WOM_DONE message. Call
waveOutWrite() to start playing and after re−filling a buffer.

Look in MSDN on the MS web−site under the function names to get more info.

Alan Lloyd

.