Reproduced from

Gramophone, November 1999

© Haymarket Magazines Ltd

PACKING IT ALL IN

The ultra high resolution requirements of DVD-Audio and Super Audio CD have necessitated the development of lossless digital audio compression systems, as Keith Howard explains

Audio history, like history in general, does not always proceed in logical fashion. So it is that, with DVD-Audio and Super Audio CD almost upon us, we are belatedly being introduced to lossless digital audio compression systems following some years of prior exposure to their lossy relatives, as incorporated in the perceptual coding processes that underpin MiniDisc, Digital Radio and Dolby Digital.

There would have been less potential for confusion had matters developed logically, in reverse order. In the mind of dyed-in-the-wool audio enthusiasts 'compression' is already a dirty word because of its association with the lossy form: a term which smacks of compromise in a context where an uncompromising approach to sound quality is a central tenet. Little wonder in the circumstances that lossless compression – which, unlike the lossy alternative, involves no compromise of signal quality – has been alternatively termed lossless *packing*, in a deliberate attempt to distance it from those technologies which discard notionally inaudible signal components, such as ATRAC, MPEG and AC-3.

Compression is a potentially confusing term in any case because it has two distinct meanings in the audio context. In its old usage compression refers to a reduction in dynamic range, deliberate or otherwise. In digital audio, however, it is also used as a shortened form of 'data compression' and refers to methods of trimming back on the large amount of data required to represent the signal. If this reduction is achieved without any modification to the signal content – in other words, if the

decompressed signal is a bit-exact reconstruction of the input – then the compression is lossless; if output and input are not identical then the compression is lossy.

The latter form of compression has been more widely used to date because of the limitations in data capacity imposed by various means of music delivery. To compress a two-channel, full-spectrum, wide dynamic range audio signal on to MiniDisc, for example, and still retain CD-competitive playing time requires data compression of such an order that it cannot consistently be achieved without data loss. Likewise fitting 5.1 channels of high quality sound on to a film print or broadcasting two channels of Digital Radio from terrestrial transmitter sites. In all these cases the compression process generally involves loss of signal data, necessitated by limitations on data capacity. (The word 'generally' is appropriate here because lossy compression schemes usually incorporate lossless encoding techniques, which potentially means that simple signals will be encoded without data loss. In the PASC lossy compression system of Digital Compact Cassette, for example, half the 4:1 data compression was achieved by lossless encoding processes.)

Lossless compression is now making an appearance within DVD-A and SACD because, with their vastly increased data storage capability, these high-density media significantly reduce the compression requirement. Although data compression is still employed to provide the desired combination of sound quality, channel provision and playing time, the amount of data saving required has fallen sufficiently for lossless compression to suffice, guaranteeing the uncompromising hi-fi requirement that input and output be identical.

Reasons

To understand why compression is still required for DVD-A and SACD it's only necessary to perform some simple arithmetic. Let's take DVD-A as the example. On a single-sided disc the maximum data capacity is 4.7 gigabytes

(4.7GB). The maximum supported sampling rate is 192kHz (i.e. 192,000 samples per second) and the maximum supported resolution 24-bit. Using these figures we can calculate the maximum playing time for a two-channel audio signal stored at the highest available quality (ignoring, for the sake of convenience, the additional data required for error correction and other subcode purposes). Each channel of 24-bit/192kHz audio generates (24 x 192000 =) 4,608,000 bits per second, equivalent to 562.5 kilobytes (KB). For two channels the total data rate is therefore 1.1MB per second. At that rate the 4.7GB capacity of the disc is used up in 4,380 seconds or 73 minutes. For a two-channel signal that might suffice, but any multi-channel provision would clearly demand an unacceptable reduction in maximum playing time and/or sacrifice in either the signal's resolution and/or sampling rate. It's to offset this compromise between signal quality and playing time while maintaining signal integrity that DVD-A and SACD both incorporate lossless compression.

Unsurprisingly given that SACD uses 1-bit DSD coding while DVD-A uses linear PCM, the two utilize different compression schemes. SACD's goes by the name of DST (a potential confusion here with both DSD and DTS) and was developed by Philips. For DVD-A a competition was organised by Working Group 4 of the DVD Forum to assess the best compression technology, and interested parties invited to submit their offerings for independent testing. Four did so, the eventual winner being Meridian Lossless Packing, a technology developed in the UK principally by the late Michael Gerzon, Peter Craven of Algol Applications and Bob Stuart of Meridian. Although the other three competitors remain officially unidentified I understand them to have been Digital Theater Systems (DTS), JVC and Matsushita - information which was not, I should stress, given me by Meridian.

Although they specify the compression figure in different ways, DST and MLP appear to achieve broadly similar orders of data saving. In the case of DST the typical compression ratio is quoted as 2.3-2.6 to 1

– ie the signal data is reduced to 38-43 per cent its original size. Figures for MLP are quoted in terms of bit reduction per sample per channel and vary according to the sampling rate of the input signal. At 48kHz the average reduction is 5-11 bits, rising to 9-13 bits at 96kHz and 9-14 bits at DVD-A's maximum permitted sampling rate of 192kHz. A 12-bit saving per sample on a 24-bit input signal corresponds to a compression ratio of 2 to 1.

How is this order data reduction achieved without any compromise to signal content? Before exploring this using MLP as the example, first a terminological aside about the use of the word entropy in this context. If you know something of thermodynamics you'll understand entropy to be a measure of disorder, the entropy of a gas, for example, being higher than that of a solid because of its lack of an organised structure. In communication theory the term is used similarly, as a measure of the disordered nature of a signal. Disorder and the transmission of information might intuitively seem incompatible (disorder suggests noise), but for information to be conveyed disorder is essential. The steady, unvarying carrier wave of a radio transmitter, for example, conveys no information other than that the transmitter is active: only if the carrier wave is interrupted (e.g. Morse code) or modulated (e.g. AM or FM radio) can it convey information. In what follows, then, 'entropy' and 'information' can be regarded as synonyms.

Essentials

All lossless compression systems incorporate three key functional elements: a framing process (which divides up the incoming signal into appropriately sized chunks for processing), a predictor and an entropy encoder (Figure 1). The predictor is alternatively called a decorrelator but for the general reader the former term gives a more ready insight into its function. It and the entropy encoder operate in series to reduce the signal's data requirement at two distinct levels. First the predictor reduces the amount of data required to describe the signal waveform itself, then the entropy

encoder reduces the data required to represent the output of the predictor. Essentially the entropy encoder is analogous to the 'zipping' software used to compress files on computer. Although the algorithms used in the audio and general data contexts may differ, the process is essentially the same. What distinguishes dedicated audio compression from zipping is the former's signal predictor element, without which the amount of compression that can be achieved is much lower – as anyone who has tried zipping computer sound files will know. Whereas dedicated real-time audio compression systems can achieve compression ratios in excess of 2:1, general purpose compression algorithms typically perform only half as well, despite the inherent advantage of processing off-line.

As its name suggests, the role of the predictor is to estimate what the signal will do next. To do this it analyses the signal using a suite of digital filters; in the case of MLP a suite of both FIR (finite impulse response) and IIR (infinite impulse response) filters are available, of up to eighth-order. Having made its estimate, the predictor then generates an error signal which represents the difference between its prediction and the actual signal waveform. These two pieces of information – prediction and error – almost but not quite constitute the predictor's output; not quite because if they did there would be no data saving. Instead the predictor outputs the error signal plus the *rules* it used to generate the prediction, which the decoder can later employ to rebuild the predicted signal. In this way a significant data saving can be achieved.

The output of the predictor then enters the second stage of the compression process, the entropy encoder. What this does is look for patterns in the predictor output which can be exploited to reduce the data requirement still further. Various methods of doing this are provided within MLP, a proprietary algorithm first examining the data to decide which of them – Huffman coding, run time coding etc – will provide

the most effective data reduction in each instance.

Entropy encoding is a subject in itself but a simple example suffices to illustrate the basic concept. Imagine you have to code the English alphabet digitally. As there are 26 available letters (ignoring upper/lower case distinctions) you would in the normal way require a 5-bit digital word to identify each uniquely. For example, you could arrange for 'a' to be represented as 00001, 'b' as 00010, 'c' as 00011, etc. But in any sufficiently large average English text we know 'e' will be the most frequently occurring letter, which means 00101 will appear more times than any other data sequence. If we code this most common sequence as, say, 1, and the next most common letter ('t') as 10, etc then we will only have to use a full 5-bit word to represent the most infrequently occurring letters. In this way we can potentially save a lot of data without losing any information. This is an example of Huffman coding, which is ideally suited to any input, like language, which is highly variable from sample to sample but conforms to a statistical pattern overall.

Another possible pattern type is temporal: for example, the multiple repetitions of pixel colour that commonly occur in a raster (bitmap) image, much of which may comprise sky or sea or other large areas of consistent colour. In this case run time coding is likely to be the most efficient method of compressing the data. Instead of sending multiple repeats of a particular code sequence you simply send it once, appending an instruction to the decoder as to how many times to repeat it. Still other methods of entropy coding are particularly well suited to other situations, depending on the nature of the patterns within the data.

MLP extras

While framing, prediction and entropy encoding are common features of any lossless compression system, individual realisations will differ both in the details of these processes and in the provision of other processing elements which may be added to enhance performance. If we look at a block diagram of the MLP encoder

(Figure 2) we see the expected predictor (decorrelator) and entropy encoder stages, but there are other processing elements too. Preceding the predictor stage are channel remap, shift and lossless matrix stages, while after the entropy encoder there is an optional output buffer stage (not illustrated). The first three assist the data compression or expansion processes while the third tackles another important issue, that of data *rate*.

Channel remapping, the first of the additional elements, has the capability to subdivide incoming channels into two or more data substreams. This allows the compressed signal to be recovered using a simpler decoder architecture, thereby saving on cost. A shift process is then applied to each data channel to recover any unused bit depth capacity, which occurs either when the input data is of less than 24-bit precision or when the channel is not fully modulated, as is the case for much of the time with typical audio content. Lastly before passing to the predictor stage the data channels are processed by a loss matrix which exploits any correlation between the signal content of different channels to cut the data requirement still further. In a conventional stereo recording, for example, correlation is typically high between the two channels as a result of central images being represented by signals of similar amplitude and phase in either channel. Similar correlations usually exist in multi-channel recordings also.

An additional path from the lossless matrix, labelled 'LSB bypass' in the diagram, is provided to route the least significant bits of the signal around both the predictor and entropy encoder stages. The signal at these low levels typically comprises noise (often deliberately added dither noise), a high-entropy signal component that can advantageously bypass the data compression process.

It's a feature of lossless compression that the output data rate is variable. Whereas in a lossy compression system more or less information can be discarded in order to keep the output data rate constant, in a lossless process the amount of data in the

output necessarily reflects the entropy of the input signal. When the amount of information in the signal (its entropy) is low, so is the output data rate, but when the signal entropy is dense the output data rate must increase to reflect this.

In the case of a transmission channel or storage medium with no limit on data rate capability, this characteristic of lossless compression is of academic interest only. It becomes very important, though, if the channel or medium has a data rate limit sufficient to accommodate the *average* requirement (as of course it must) but which is less than the *maximum* that the lossless coder might generate on certain high-entropy signals. This is the case with DVD-A which has a maximum data rate of 9.6Mbps (megabits per second) but is specified to carry up to six channels of 24-bit/96kHz data, which potentially demand a peak data rate of 13.824Mbps.

This where the provision of buffering becomes important. If the data rate from the entropy encoder exceeds the maximum allowable, the excess data is temporarily diverted to a FIFO (first in, first out) buffer memory and only read out again once the data rate has fallen sufficiently. Figure 3 shows an example of buffering at work in MLP, the signal in question being a 30-second excerpt from a six-channel 24-bit/96kHz recording which features closely-miked cymbals in all six channels. Because of the virtually random nature of this signal its entropy is unusually high and the underlying compressed data rate reaches 12.03Mbps. As soon as the output of the entropy encoder exceeds 9.2Mbps, however - just below the maximum 9.6Mbps data rate supported by DVD-A - data begins to accumulate in the buffer, awaiting sufficient fall in the entropy of the input signal. When this occurs the buffer is progressively emptied again. In the example the required buffer memory is around 85kB and the graph scale goes up to 256kB, but Meridian declines to identify just how large a buffer MLP incorporates for DVD-A. In the extreme case of the data rate requirement exceeding the buffer provision, MLP offers the recording engineer various options for

reducing data within the source signal, by trimming back the sampling rate or reducing the bit depth on a channel by channel basis. This provision also allows a producer to increase playing time if required.

A block diagram of the MLP decoder (Figure 4) reveals, as you would expect, a mirror image of the encoder structure. What isn't apparent from the diagram is the decoder's relative simplicity – a key practical requirement since decoder complexity determines the cost of implementation in the end product. Meridian says that the computing power required to decode a two-channel data stream at 192kHz sampling rate is 27MIPs (millions of instructions per second), while six channels at 96kHz requires 40MIPs. These figures are well within the capability of inexpensive modern DSP chips.

Dolby Laboratories is handling the licensing of MLP and will provide technical support in the same manner as for its own products. To date ten semiconductor manufacturers have expressed an interest in developing and selling MLP decoders, two of whom – Motorola and Cirrus Logic (Crystal Semiconductor) – have publicly announced that they will do so. With DVD-Audio set for launch next year, it isn't long before the first chips will be needed.