Multichannel Digital Audio Connectivity

IN THIS DOCUMENT

Multichannel audio is pervasive in modern society. Sometimes it's as an integral part of the very activity itself, such as listening to music or a movie. Other times it's hidden away as an unseen (or perhaps we should say unheard?) technology enabling a better user experience, such as an intelligent speaker phone.

This article explores applications of multichannel audio, identifies important aspects in multichannel audio, and considers the requirements these place on the selection of system components in the design of a multichannel digital audio connectivity product.

1 What is multichannel audio?

The term multichannel audio may be interpreted in many ways:

- As a consumer, we typically associate multichannel audio with the playback of content; such as listening to music in stereo, or watching movies with 5.1 or 7.1 surround sound.
- A recording engineer uses multichannel audio to record a band session, using separate microphones for each vocalist and instrument. Later, a mixing engineer combines these multiple independent tracks into a final stereo mix.
- A DJ uses multichannel audio to blend multiple stereo audio sources together, to create cue feeds and a master output mix.
- A live event uses multichannel audio to capture an onstage performance and relay it to the audience, all in real-time.
- A public address system uses multichannel audio to route a mixture of live and pre-recorded audio content from one or more sources to one or more output zones, either independently or simultaneously.

Publication Date: 2016/3/14 Document Number: XM010299A

- A smart conferencing system might use embedded multichannel audio to capture the output from an array of directional microphones, for real-time processing in to a single audio feed.

All of these are valid interpretations and illustrate just some of the diverse applications of multichannel audio and its use for audio playback (output), audio recording (input), and combinations of both.

For the purposes of this article, we shall define multi-channel audio as two or more channels of audio. More specifically we shall say two or more channels of audio that are related in time and amplitude.

2 Preserving relationships

If multichannel audio means multiple channels of related audio, then the definition of a good multichannel audio system is one which that respects and preserves these relationships. But why is preserving these relationships important?

2.1 Phase delay

When capturing, processing and/or playing-back multichannel audio, keeping the audio channels correctly aligned in time is very important. Whilst this may sound obvious, for channels with similar content it is critical. For example, a timing misalignment between the channels of a stereo music track will introduce a phase shift, which in turn will cause content at certain frequencies to be attenuated, or removed completely. The plots below show examples of how destructive this effect can be. Figure 1 illustrates how a 2sample misalignment in a digital system sampling at 48kHz attenuates higher frequency content. Figure 2 illustrates how a 1ms delay manifests as an undesirable 'comb filter' effect.

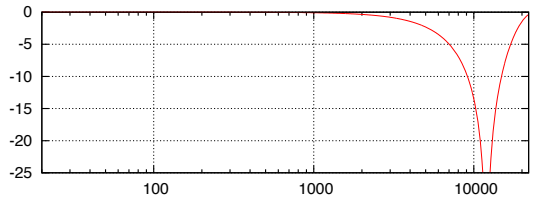

Figure 1: Effect of 2sample phase shift at 48kHz

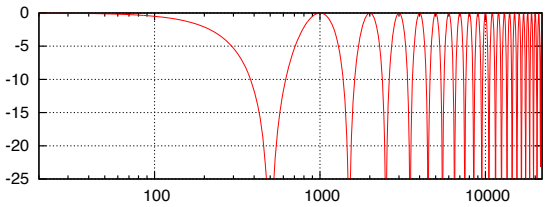

Figure 2: Effect of 1ms phase shift

2.2 Latency

Another important timing consideration is latency – this is the delay in the audio path as it passes through equipment.

Mismatched latency between disparate processing blocks in a multichannel audio path can cause the phase shift problems described above. In a multichannel audio system complementary delay stages may therefore be required to ensure all channels remain time aligned. This requires careful control to manage effects being switched in and out, and cope with effects delay varies. A simple example is shown in Figure 3.

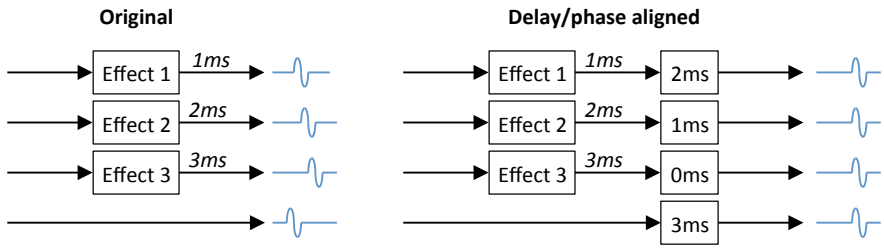

Figure 3: Effect of delay/phase alignment

Latency is also particularly important for 'live' audio systems; where the output of the audio system is available at the same time of the input to the audio system. For example, a singer wearing monitor earphones will hear both their own voice and the monitor feedback. A delay on the monitor feedback can be heard as an echo. Echo delay perceptibility varies between individuals, but delays of 10-15ms are considered noticeable and delays above 15-20ms distracting.

The effect of audio latency is also important in audio-video applications; to avoid 'lip sync error' – where audio content is not time aligned with onscreen activity. TV industry guidelines recommend that audio should lead video by no more than 15ms and audio should lag video by no more than 45ms. Expert viewing tests found the threshold for detectability to be 45ms lead to 125ms lag.

2.3 Gain

Maintaining the correct amplitude relationships between each channel is also important. In a recording studio, gain errors between channels will result in level mismatches between vocal and/or instrumental content. Mismatched channel levels in a stereo or surround sound system will corrupt positional presentation and lead to a distorted sound stage – just in the same way a stereo balance control shifts the left-to-right stereo imaging.

3 Requirements

We have shown why preserving the timing and amplitude relationships between audio channels is so important in a multichannel audio system. Getting this right is a core requirement to building robust, reliable and accurate multichannel audio systems.

Whilst all of the above applies to any multichannel audio system, for the rest of this article we shall look at digital implementations and in particular focus on multichannel digital audio connectivity.

First let's take a quick detour to explore what is meant by digital audio.

4 Digital audio

Audio data is commonly represented in the digital domain using Linear Pulse Code Modulation (PCM) encoding. PCM is a raw, uncompressed format where each audio channel is sampled at a fixed rate and bit-depth; 16bit stereo at 44.1kHz in the case of 'CD quality' as shown in Figure 4.

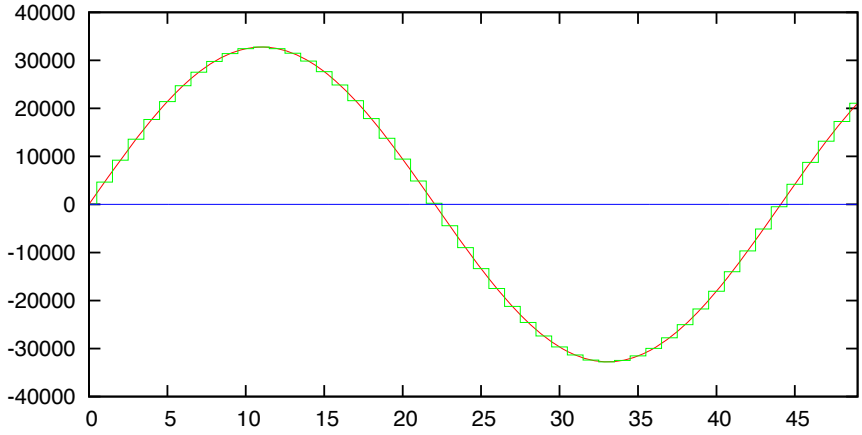

Figure 4: PCM sampling

PCM is a direct digital representation of the sampled analogue audio signal and produces a constant bitrate data stream. No processing is required to interpret PCM data, which provides low-latency access and makes it relatively easy to manipulate the audio signal in hardware and/or software.

4.1 Audio resolution

PCM is flexible and may be implemented at any sample rate and bit-depth. Common multichannel audio sample rates are 44.1kHz (CD and MP3 audio), 48kHz (broadcast and consumer recording), 96kHz (studio recording) through to 192kHz and even 384kHz (high-resolution audio recording). Common audio bit-depths vary

from 16bits (consumer audio), to 24bits (recording, processing and high-resolution audio) up to 32bits (processing and high-resolution audio).

4.2 Channel count

Multichannel audio channel count varies greatly across its many applications. From a simple 2 output channel system for listening to music in stereo, to a 6 input and 4 output channel DJ mixing deck, through to a prosumer recording studio where channel counts of 16 or 32 input and outputs are common, right up to a professional recording studio complex with perhaps hundreds of channels.

4.3 Data-rate

In a digital audio system, the combination of channel count and audio resolution defines the audio data-rate.

$$DataRate(bits/s) = ChannelCount \times SampleRate(Hz) \times BitDepth(bits)$$

In a system with limited data bandwidth the above equation illustrates how channel count and audio resolution may be traded to offer maximum user flexibility. For example: the user could be presented with the option of 40channels at 48kHz/24bits, or 20channels at 96kHz/24bits, or 10 channels at 192kHz/24bits.

The equation also clearly shows how the trend towards higher resolution audio leads to higher data rates. For example: stereo audio sampled at 384kHz/32bits produces a 24.6Mbit/s data stream.

A robust digital multichannel audio connectivity solution needs to maintain channel timing, amplitude accuracy and control system latency irrespective of the channel count and audio resolution used.

4.4 Clocking

A PCM encoded digital audio stream contains only raw sample data; it does not include any time information. The use of an accurate, stable, clean and jitter-free clock to align the sampled data against is therefore vital to ensure accurate representation of the underlying audio signal(s).

What is jitter? A clock of frequency F(Hz) should produce a pulse exactly every $1/F$(seconds). Any variations in the actual timing of when the pulse occurs is known as jitter. Jitter on a clock degrades audio quality by causing audio samples to be misplaced in time. Jitter can be introduced when a clock is generated (especially if recovered or derived from another signal source) or distributed around a system.

In a multichannel digital audio system there may be many sources and sinks of digital audio content. Ideally these will all be configured to operate from a single master clock, and at the same sample rate. Where this is not possible specialist sample-rate conversion techniques must be used to accurately resample the data from one time domain to another.

To illustrate why this is necessary, consider a multichannel digital audio system with two separate clock sources, both 24.576MHz. Let's say that due to manufacturing tolerances, temperature and ageing effects these clock sources have an accuracy of ±100ppm. At worst case, one of these clocks could be running ~5kHz faster than the other. This will cause the audio data to drift apart. If these clocks were sampling audio at 96kHz, this drift will cause 1sample slip every 52seconds. In the short term this could cause the phase shift effects described earlier. In the longer term this will cause audio samples to be either dropped or repeated.

5 Multichannel digital audio connectivity

A multichannel digital audio connectivity solution enables the transfer of multiple audio channels over a standardised interface. This might be an audio specific interface; such as S/PDIF, ADAT or MADI; or the encapsulation of audio data on to a more general purpose interface; such as USB, Firewire or Ethernet.

A relatively straightforward connectivity solution might be a PCM stereo S/PDIF decoder, as shown in Figure 5. This device needs to recover clock timing information and provide a bit-accurate transfer of audio sample data. Typically a Phase Locked Loop (PLL) is used to recover a low-jitter version of the underlying clock signal. Since S/PDIF does not define the data-rate, the system needs to be flexible to accommodate a range of incoming audio sample rates.

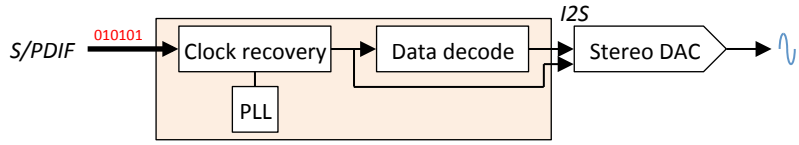

Figure 5: Example PCM stereo S/PDIF decoder

MADI, otherwise known as AES10, defines an interface that can carry multiple (up to 64) channels of digital audio over a single coaxial cable, or fibre-optic line for longer distances. The example of a MADI encoder, shown in Figure 6, ingests multiple audio channels, multiplexes them together, packs that data into frames, and then encodes the serial stream for subsequent transmission and distribution. The system needs to maintain phase alignment of all input channels, deliver bit-accurate audio transfer and provide consistent behaviour across variations in channel count and sample rate.

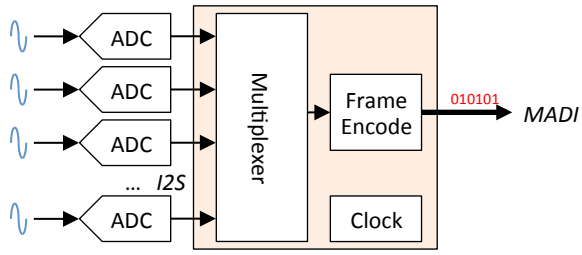

Figure 6: Example MADI (AES10) interface

The next example Figure 7 introduces bidirectional multichannel audio and encapsulation of the audio data in to a higher-level non-audio specific protocol, in this case Ethernet. Whilst the external interface has changed, the overall system requirements remain the same: to preserve the timing and amplitude relationship of the audio channels and to provide bit-accurate transfer of the audio data. Clearly a key factor in determining overall system performance, especially latency, is the choice of audio-over-Ethernet protocol (the discussion of which is beyond the scope of this article). For example, packetisation of the audio data for transmission over an Ethernet link will require local buffering of audio data between packets, and the Ethernet link quality-of-service will determine how many packets then need to be buffered.

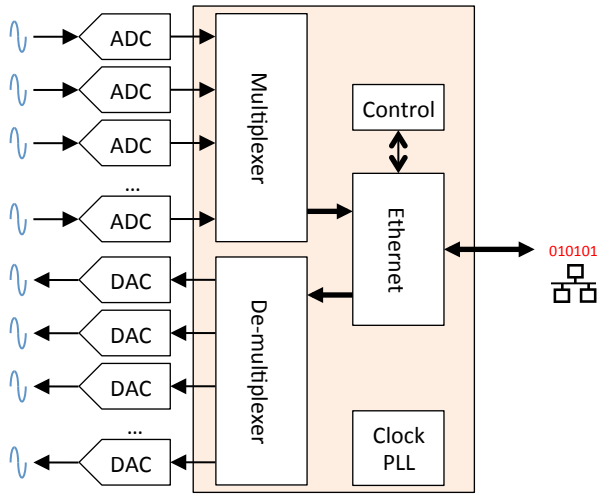

Figure 7: Bi-directional multichannel Ethernet interface

When implemented correctly, all of the previous examples should be audibly transparent; each system should transfer a bit-accurate, time-correct digital representation of the audio data.

In other multichannel digital audio applications, manipulation of the audio data is actually a requirement. For example: gain adjustment, frequency filtering, delay effects, channel mixing, or other complex DSP algorithms.

Figure 8 is an example of a mixer with a USB interface, connected to a host PC running Digital Audio Workstation (DAW) software.

A single digital clock domain is used across the entire system. This can be selected from one several sources:

- Local crystal oscillator
- Derived from a digital audio (S/PDIF) input
- Slaved from an external master reference clock source

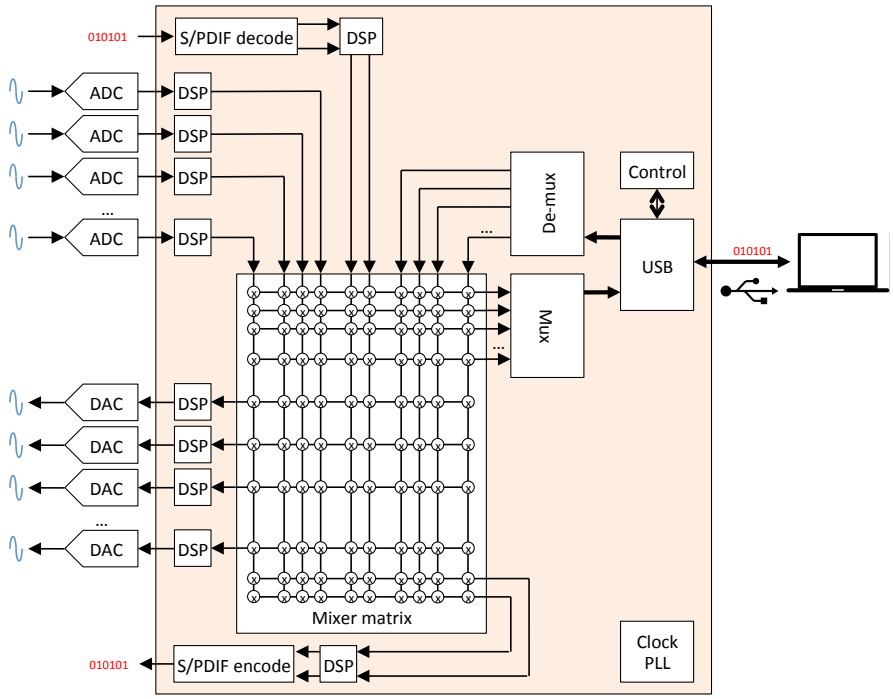

Figure 8: Mixer with USB interface, connected to host PC running Digital Audio Workstation software.

- Synchronised with a host PC system over the USB link.

To maintain accurate phase alignment all the input channel DSP blocks should implement the same delay, irrespective of whether the DSP is active or not (unless adding delay is a deliberate part of the DSP function!). Similarly, all the output channel DSP blocks, and all paths through the mixer, should implement consistent delays.

In contrast to the previous examples, the DSP and mixing functions may alter the audio data. To ensure bit-accurate processing and prevent loss of audio precision, wider internal data-paths providing processing headroom are required.

Like the previous Ethernet example, audio data is packetised for transmission over USB to a host PC system. Again, such packetization requires buffering, which introduces latency. USB Audio Class uses isochronous transfers to guarantee bandwidth availability and ensure quality-of-service allowing this latency to be kept as low as 500µs in each direction. However, it should be noted that other latencies will likely exist in the host; for example at the driver and DAW application software layers.

Revisiting a previous end application, consider an artist's microphone connected to an input channel, and their monitor earphones connected to an output channel.

A local feedback loop can be constructed through the mixer matrix. Also, a remote feedback loop can be constructed via the USB interface and DAW software running on the host PC. Clearly this remote loop will have a higher latency. If using this path then to avoid the artist experiencing distracting echo feedback, a low latency USB audio implementation is a critical requirement.

This example also illustrates the need for user control; how to configure and control all the DSP, routing and mixing functions? A real product might provide buttons, knobs and dials for user 'hands-on' control, or a software GUI on the host PC with control over the USB link, or combination of both.

For a more in-depth discussion of audio over USB, see the XMOS paper: *Why do you need USB Audio Class 2?*[1]

6 Pulling it all together

Our brief tour through a few multichannel digital audio connectivity examples has highlighted some important considerations in the selection of system components when designing a multichannel audio product.

- Audio flexibility. Every application is unique - with different input and output channel counts and a range of operating sample rates and bit-depths. A common and flexible architecture is required; one with the capability to accommodate all sample rates and bit-depths, and flexibility to scale up in channel count to span across entire product ranges.
- Interface flexibility. An audio connectivity solution is only as good as the interfaces it can support. A solution needs to support standard audio interfaces as well as other audio-capable interfaces, preferably with the flexibility for customisation, or even to implement one's own proprietary interface.
- Data integrity. Bit-accurate transfer of audio data is a mandatory requirement. Dedicated audio data paths are necessary to ensure continuous, timely and robust flow of the audio data, without any interruptions that may lead to delayed or lost audio samples.
- Latency. In real-time applications, minimising latency is critical. Parallel or concurrent processing of audio data on a sample-by-sample basis will be advantageous to a buffering and batch processing approach.
- Audio precision. In simple audio interface applications a data path the same width as the sample data will be sufficient. However applications that process audio require greater bit depth to provide processing headroom and so preserve signal integrity. For example: summing two channels requires an additional data bit to maintain accuracy, and multiplying two channels requires double the number of data bits to maintain accuracy.
- System control. All but the very simplest of applications require some level of system configuration and control. Often some form of user input and feedback is also required. Intuitive, smart controls build value-add and differentiate one

[1] http://www.xmos.com/whyuac2

product from another. Some form of programmable general purpose compute with flexible I/O is required.

- Upgradability. Products rarely stand still. A solution that can provide an upgrade path allows for a fast refresh of a product line – to add new features or adapt to evolving market trends and standards. Such capability can also be used externally to update or resolve issues with deployed product in the field.

7 The XMOS solution

A scalable platform that provides two-to-many channels is an absolute requirement of any multichannel audio product. The platform must be flexible so you can create a range of products and maximise your IP, while future-proofing yourself against new standards and ever-increasing channel count and audio resolution requirements.

XMOS xCORE multicore microcontrollers offer programmable flexibility and low-latency deterministic performance making them ideally suited for multichannel audio connectivity applications. On-chip Hardware Response ports allow complex timing critical interfaces to be defined in software, and powerful multicore processing always ensures on-time, bit-perfect processing and transfer of up to 32 channels of audio data from one interface to another.

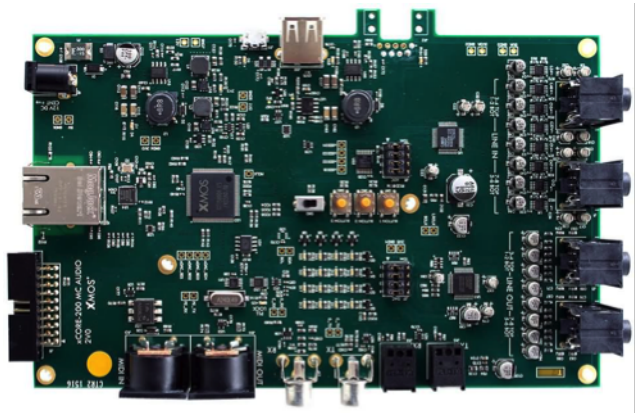

Figure 9: xCORE-200 Multichannel Audio Platform

Software libraries of standard audio interfaces (I2S, TDM, S/SPDIF, ADAT) together with USB and networking audio software components, all using common APIs, let you create applications that deliver the audio connectivity you require. Need to add some DSP? The 32bit data path with full 64bit multiply-accumulate precision and single cycle instruction execution, together with a sample-by-sample pipeline approach to data processing, enables predictable low-latency audio DSP – ideal for live applications.

The xTIMEcomposer Studio development tools make software development fast and easy, allowing you to quickly build applications from reference libraries as well

as develop additional application specific features; such as monitoring and driving your user controls and displays.

All of the above libraries are complemented by a family of hardware and software audio connectivity development platforms that provide complete reference implementations of USB Audio Class 2.0 and Ethernet AVB Audio.

The performance and configurability of xCORE multicore microcontrollers coupled with reference software delivers support for very high bi-directional channel counts, scalable audio resolutions and formats, and flexible audio interfaces. xCORE provides the ideal future-proofed solution for your multichannel audio products.

About Author

Laurence Strong[2], Technical Marketing and Applications Specialist

[2] http://uk.linkedin.com/in/laurencestrong