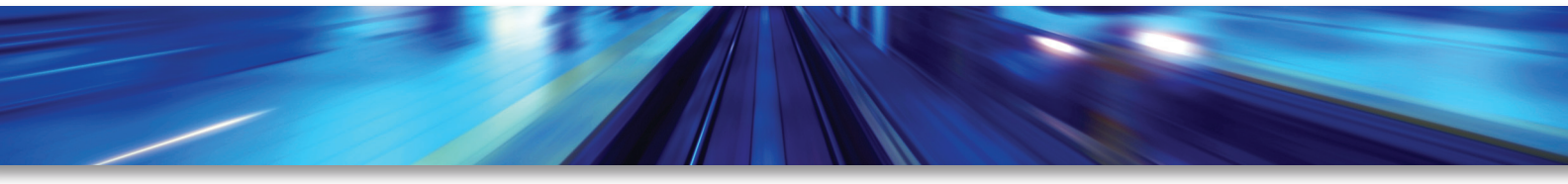
SYNOPSYS®
Predictable Success

Simple Ways to Manage Different Clock Frequencies of Audio Codecs

April 2011



Author
Carlos Azeredo-Leme
Analog Design,
Senior Staff,
Synopsys, Inc.

Overview

Audio processing is essential to many electronic applications such as mobile phones, MP3 players and a host of other products. Size and power consumption are often critical design criteria while the market demands high-quality high fidelity (Hi-Fi) audio capabilities. These trends have pushed the audio codec into an embedded core within modern systems-on-chip (SoCs).

The audio codec creates the interface between the digital host processor and the audio transducers, such as microphones and speakers. It is also responsible for several routine audio functions, thereby alleviating the workload on the host processor. When embedded in a SoC as an IP core, an audio codec appears as a digital block to the internal interfaces and transparently handles all the off-chip analog transducers and inputs/outputs (I/Os).

On the internal digital interface, it is important to understand the aspects relating audio sample rates and clocks. The clocks required by the data converters on an audio codec depend on the audio material sampling rates as well as on the clocks available on the host application and SoC. The combinations are quite complex due to the multitude of audio sample rate options and available host clocks. To further complicate matters, in audio-video (A/V) applications, the audio clocks need to also be synchronized with the video clocks required by the video data converters. Therefore, many designers are confronted with complex choices when deciding on trade-offs to minimize system costs related to clock generation and interfacing a multitude of sample rates.

The digital filters play an important role to help solve this problem because they process the digital samples between the digital audio interface and the audio data converters, and therefore, can perform sampling rate conversions.

This paper will review the functions of digital filters in audio codecs and will illustrate with several examples how they can be used to support interfacing in a multitude of sample-rates and clock environments.

Audio Processing

The core of an audio codec is composed of two types of data converters: a digital-to-analog converter (DAC) for playback and an analog-to-digital converter (ADC) for recording. For a stereo or multi-channel codec, these are replicated accordingly. Figure 1 shows a typical block diagram of an audio codec.

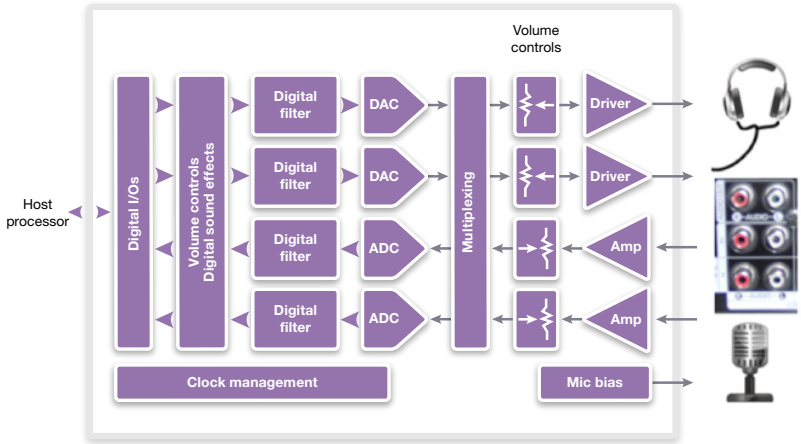


Figure 1: Functional block diagram of an audio codec

On the ADC inputs, there are amplifiers with volume controls to bring both the weak microphone levels and the large interconnect line levels to the input range of the ADC. On the DAC outputs, there are output drivers able to directly connect to earphones or to small speakers, each with its respective volume controls. It is possible to bypass input signals directly to the outputs, without passing through the ADC and DAC. There is also a low-noise power supply for microphone biasing.

On the digital side, there are multiple blocks. The most important are the digital audio filters that convert the data rate to the oversampled clocks of the data converters and remove the high-frequency noise outside the audio band. Also important is a clock management block, which makes sure that all multi-rate blocks are synchronized with each other and supports all the required sampling rate combinations. There are additional blocks on the digital domain, such as volume control, which provide additional flexibility to allocate gain/attenuation between digital and analog blocks. In addition some audio codecs can include sound effects that are implemented digitally, for example bass boost, 3D audio effects, automatic volume controls and parametric equalization.

Digital Audio Filters

The data converters used nowadays in audio codecs operate at highly oversampled frequencies, meaning that their conversion frequency is much higher than the audio band, by a factor of 100 or more. For example, assuming an audio data rate of 44.1 kSamples/s (kS/s), such as from a Redbook CD player, the typical oversampling rate is 128X, leading to a conversion rate on the DACs of 5.6448 MSamples/s (MS/s).

The reason for such high conversion rates is related to the techniques used for data converter design, which provide an important advantage by enabling all required filtering to be implemented in the digital domain. It is much more efficient to implement filters digitally than in analog form. In a digital implementation, the transfer function is free of distortions and there is perfect matching between the left and right channels. And in modern semiconductor technologies, due to process scaling, the silicon area and power consumption of the digital filters have become negligible compared with their analog implementations.

Why are filters required on an audio codec?

The main reason filters are required on an audio codec is for removal of the aliasing or imaging bands. These are replicas of the signal band around the multiples of the audio sampling rate and are a result of the multi-rate operation. For example, an audio stream at 44.1 kS/s up sampled to 5.6448 MS/s has spectrum replicas around 88.2 kHz, 132.3 kHz, and so on. This is a result of the Nyquist Sampling theorem, as illustrated in Figure 2.

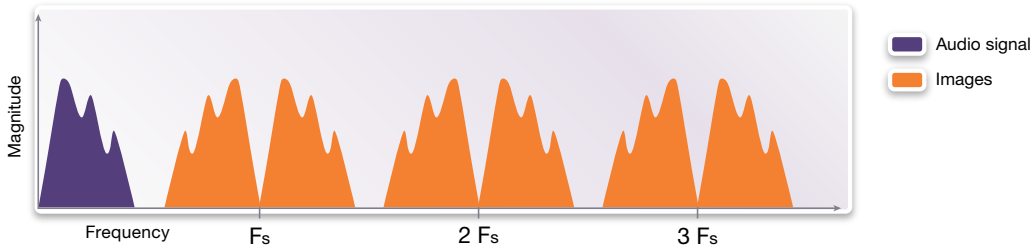


Figure 2: An audio signal sampled at a sampling frequency (F_s) has replicas of its spectrum at the multiples of the sampling frequency (in orange)

On a DAC, the image bands cause a stair-like waveform as shown in Figure 3. The filter smoothes the waveform and reduces the high-frequency energy. If this high-frequency energy were not removed, it would waste power and cause intermodulation distortion in the output drivers causing the loud-speakers to generate audible noises.

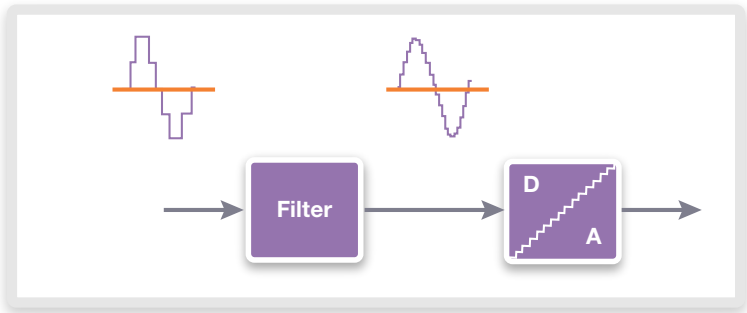


Figure 3: The digital filter up-samples and smoothes the signal waveform before application to the DAC

On an ADC, the filter removes any out-of-band noise picked up at the input or generated within the ADC, as shown in Figure 4. If not removed, when the signal is re-sampled at the standard audio rate (F_s), that noise would be folded down in-band due to aliasing and become audible.

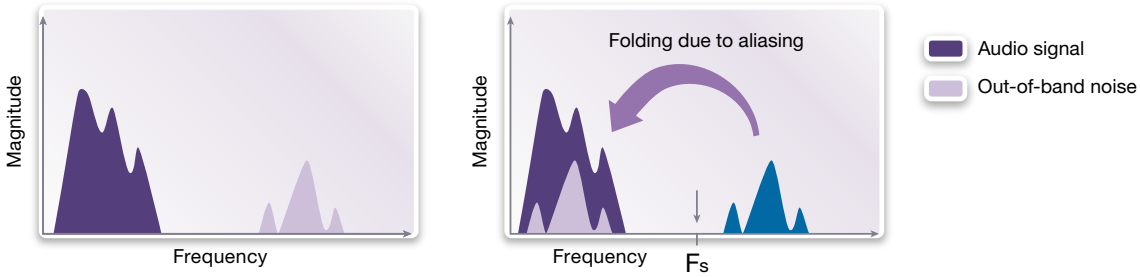


Figure 4: On an ADC, any out-of-band noises (plot on the left in lavender) would be folded down into the signal band when the sample rate is re-sampled to the standard audio rate (F_s) at the output (plot on the right)

Types of Filters and Phase Fidelity

There are two categories of digital filters:

- Infinite Impulse Response (IIR), also known as recursive filters because its output depends on both past input and past output values
- Finite Impulse Response (FIR), also known as non-recursive filters because its output depends only on past input values

IIR digital filters are similar to analog filters, and they have poles and zeros. The poles allow very efficient transfer function realization. So the order, or complexity, of the implementation is reduced. But the poles introduce phase response distortions.

FIR filters have no poles and they can be made such that all zeros are ideal notches (on the unit circle in Z-plane). These filters guarantee linear phase response. However, for achieving the same response specification, FIR filters must be of much higher order than an IIR equivalent. Even if the complexity for implementing a zero is lower than for a pole, the complexity of a FIR filter is normally higher than for the corresponding IIR, therefore, resulting in larger area and consumption.

There is much debate whether linear phase response is mandatory for consumer Hi-Fi equipment. There are studies that show that our ears cannot distinguish phase distortions unless it is very strong (see e.g. http://www.ocf.berkeley.edu/~ashon/audio/phase/phaseaud2.htm). A properly designed IIR filter compensated with phase equalization should sound perfect. However, the addition of the phase equalizers increases its complexity to nearly that of an equivalent FIR filter. The advantage is the much lower latency of the phase compensated IIR compared with the FIR and that is critical in some applications such as gaming.

Typically, mains plugged Hi-Fi equipment use the FIR type for ideal phase response, while consumer battery-powered equipment tend to use IIR filters that lead to much lower consumption and cost, with virtually inaudible phase distortions.

Clocks and Audio Sampling Rates

Digital audio signals are sampled at standard frequencies. Due to legacy from the old Redbook CD, many audio recordings use the standard 44.1 kS/s rate. This awkward number is derived from an early practice of reusing PAL videotape equipment for audio recordings. Modern audio systems, like the DVD format, use 48 kS/s and its multiples 96 kS/s and 192 kS/s. Voice applications, such as those in cell phones, use 8 kS/s and its multiples, 16 kS/s and 32 kS/s. Some applications may also use multiples of 44.1 kS/s, namely 88.2 kS/s and 176.4 kS/s. Since the data converters must operate at oversampled frequencies, typically 128X, or 256X, the required master clock frequencies to drive the data converters would be in the range of 5-12 MHz.

An audio codec must therefore support a wide variety of audio sample rates and accommodate a range of master clock frequencies facilitating its integration in the application. It is not a straightforward objective due to the multitude of combinations and restrictions in the possible clock frequency ratios. For this reason, the digital filters must include programming of its sample rate conversion. For example, let's consider a practical case with an audio rate of 48 kS/s and the converters sampling frequency of 6.144 MS/s. The resulting sample rate conversion is 128X. To support practical crystal frequencies, a divide by two is used, leading to a master clock frequency of 12.288 MHz.

Now, for supporting 96 kS/s, the filters are reconfigured for a sample rate conversion of 64X. And for supporting 192 kS/s the filters are reconfigured for a sample rate conversion of 32X. The data converters sampling frequency stays constant at the same 6.144 MS/s because the audio band limit is fixed at 20 kHz. For the 44.1 kS/s audio rate family, the corresponding master clock would be 11.2896 kHz.

Phase-locked loop for the audio clock

Many applications cannot have dedicated crystal oscillators for the audio codec, for example where space and/or cost are the limitation. The audio codec must therefore be able to support the different audio rates from the available host master clock, which is often the USB clock of 12 MHz or a multiple of it. Then, a phase-locked loop (PLL) can be used to generate the required audio clocks. But this PLL is not a commodity type due to the very fine frequency resolution required for supporting all the frequency combinations, while at the same time providing a low-jitter output clock for performance. Other solutions not requiring a PLL would be preferable.

PLL-less technique

An alternative solution is the PLL-less technique of re-using the USB clock and avoiding adding a dedicated PLL for audio. USB is a very popular interface and almost universally present in any application. Either 12 MHz or 24 MHz clocks are used and have relatively low jitter, which is an important requirement for audio.

A USB clock of 24 MHz can support the 48 kS/s audio rate and its multiples, 96 kS/s and 192 kS/s, because it is an integer multiple (24000 = 125 x 192). To use it, the filters sample rate conversion needs to be reconfigured from the nominal 128X to 125X.

The 44.1 kS/s audio rate, however, can only be approximately generated. Using a sample rate conversion of 136, the audio clock can be approximated to 44.1176 kS/s, which is slightly different from the nominal. But the difference is quite small and hardly noticeable. In effect, it is just a 0.04% change in pitch. To put that in perspective, it is 100 times less than a semitone. Another way to appreciate the effect of the clock approximation is in the duration of a song: 0.04% faster playback corresponds to a 3-minute song completing 10 milliseconds (ms) earlier.

AV clocks

A/V multimedia equipment produces data streams that are both video and audio. Examples are DVD players/recorders and MPEG media. The sampling rates are independent for video and audio. Video uses 27 MHz clocks or multiples. The audio clocks must be derived from 27 MHz, and all standards must be supported, those based on 44.1 kS/s, 48 kS/s and 8 kS/s.

These audio clocks are best generated with a PLL having as reference a division of the 27 MHz video clock. Due to the synchronism between audio and video data, the PLL-less technique is not applicable because it would change the audio playback cadence relative to the video. This would cause the audio to get misaligned with the video image, causing loss of lip-sync.

Synchronization with the source clock

There are applications where the source of data is remote to the equipment. An example of this would be TV broadcasts (cable, terrestrial or satellite), HDMI cables and web streaming. In these cases, the audio and video clocks must be synchronized with the source clocks. To illustrate this situation, let's consider a digital TV transmission.

The digital television signal is primarily comprised of MPEG transport streams. Basically, a transport stream is a combination of digitally encoded video and audio data. After decoding, the digital signals are converted to analog in the audio DACs and the video DACs. The data transmitted by the TV station is synchronized to a reference 27 MHz clock at the transmitter site. Due to tolerances on the transmitter crystal oscillator and Doppler effects during propagation, the frequency received by the TV antenna at the receiver site may vary. In order to ensure synchronization of the data, a coded time stamp (CTS) is included in the transmitter MPEG data signal. The CTS allows synchronization of the video and audio packets for an accurate lip-sync during playback.

Since the received frequency varies over time, data may be arriving slower or faster, producing clock drifts relative to the receiver own 27 MHz clock. This variation in data flow will result in either too much or too little data arriving at the receiver to be processed.

For video data, this drift is not significant and can be corrected by dropping or repeating frames. While the video is decoded, the receiver is constantly comparing the time stamp received from the transmitter with that of the decoded video. When too much data accumulates in the decoder (i.e., the receivers 27 MHz clock is slower than the transmitter clock), a frame is dropped. When too little data is available for processing, a frame is repeated. Since frames are either dropped or repeated so infrequently, the effect is not noticeable to the viewer.

Audio data must also be synchronized with the transmitter clock. The audio clocks must be derived from the same 27 MHz video clock, and all standards must be supported. But dropping/repeating samples are unacceptable in audio because it would be highly perceptible to our ear. The solution is to have a PLL to generate the audio clock, and use the time stamp information to lock the PLL to the received sampling data, as illustrated in Figure 5.

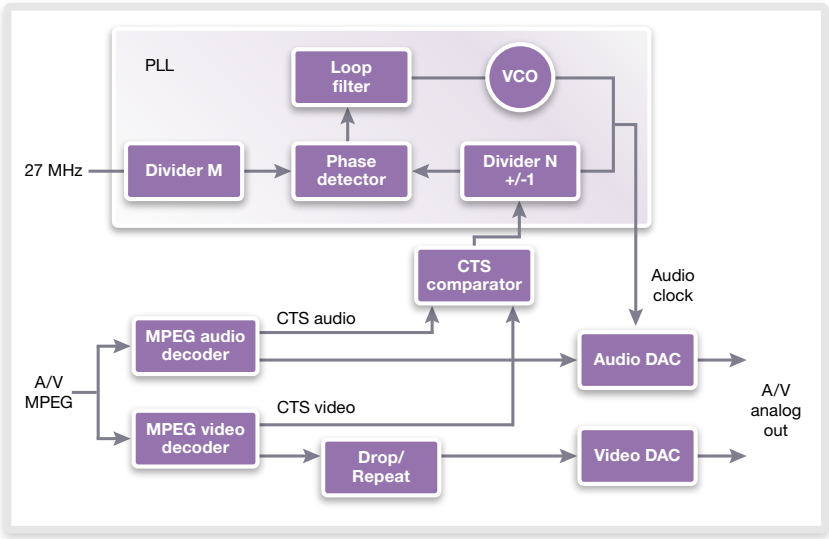


Figure 5: Block diagram of a solution with a PLL that uses time stamp information from the video decoder to synchronize the audio clock

Assuming, for instance, 256X oversampling on a 48 kS/s audio stream, the PLL will need to generate 12.288 MHz. The greatest common divisor of 27 MHz and 12.288 MHz is 24 kHz, so the 27 MHz clock must be first divided down by M=1125 and then multiplied in the PLL by N=512.

To synchronize the audio clock, the CTS comparator compares the time stamp for audio and video, incrementing or decrementing the PLL multiplier ratio N accordingly. When audio decoding is advancing CTS too quickly, N is decremented by 1 to 511. When audio decoding is delaying CTS too slowly, N is incremented by 1 to 513.

Summary

Audio codecs are used in a wide variety of audio and video applications with a multitude of standards and audio rates. This often creates complex situations to support with the limited clock generation capabilities of the host application. Making sense of it all is the task of the digital filters and clock management features of modern audio codecs.

By proper reconfiguration of the digital filters, sample rate conversion and flexible clock frequency choices, it is possible to support a wide range of audio-video applications with modern audio codecs. Examples were given with solutions for these applications, ranging from the PLL-less technique to support USB clocks, to multimedia systems with synchronization between audio and video clocks that help designers, to understanding the trade-off decisions to minimize the costs of their SoCs.

By making available silicon-proven, fully featured audio codecs, Synopsys' DesignWare® IP enables SoC designers to achieve the best trade-offs between cost and performance while allowing them to focus on product differentiation. For more information on Synopsys' DesignWare Audio IP solutions, please visit www.synopsys.com/audio.

SYNOPSYS®
Predictable Success
Synopsys, Inc. • 700 East Middlefield Road • Mountain View, CA 94043 • www.synopsys.com