Exploring Dither in Floating-Point Systems

By Nika Aldrich

February 13, 2005

Introduction

One topic that surfaces with regularity is that of floating-point versus fixed-point processing. Since most computer DAW systems currently use floating-point protocol for internal processing a flurry of articles and innuendo seems to have surfaced regarding the benefits of floating-point nomenclature. Indeed there are many benefits to a floating-point architecture. Floating-point systems can handle much larger and much smaller numbers than fixed-point systems, thereby giving them the ability to handle much wider dynamic ranges. Further, some mathematical processes used in digital signal processing can take advantage of the ability to represent (and then mathematically manipulate) large and small numerical values.

In any mathematical system, simple operations create larger numerical results. Multiplying a 3-digit number times a 3-digit number creates a 6-digit number, for example. These larger values (more bits, in the digital audio world) create the need for some form of bit-reduction. Floating-point systems, as well as fixed-point systems, need to be able to appropriately bit-reduce values. The mathematically correct methodology is the application of dither, as was first presented and proved decades ago. Dither is easily and "properly" applicable in fixed-point domains, but it is not so easily applied in floating-point systems. This paper will explore the issues having to do with dither and other bit-reduction schemes in floating-point systems.

Before we address this it is probably necessary to review a few concepts or definitions that relate.

Noise: Any signal added to another signal that is unrelated to it (uncorrelated to it) in all ways. As the initial waveform changes the added signal may or may not change, but this is entirely independent of the changes of the initial signal.

Distortion: Any signal added to another signal that is related to it (correlated to it) in some way. As the initial waveform changes the added signal changes in a predictable and definable way. The change of the added signal is directly related to some aspect of the initial signal.

Quantization Error: The rounding error that occurs from quantizing a waveform, either during the initial sampling process or during bit-reduction stages. Quantization error is a

waveform unto itself that is added to the initial waveform, the summing of which produces the shape of the quantized result.

Quantization Distortion: If the quantization error is correlated to the initial waveform then the quantization error (waveform) is distortion.

Quantization Noise: If the quantization error is uncorrelated to the initial waveform then the quantization error (waveform) is noise.

Dither: Noise added to the waveform prior to quantizing. If the noise is added with enough amplitude then when the signal is quantized or bit reduced (quantized to a less-fine statistical resolution) the quantization error becomes correlated to the noise instead of to the signal. This means that the quantization does not lead to distortion of the signal. Instead, it leads to quantization of the noise. If the noise that was added was random white noise then the quantization error is also random white noise.

It is important to remember this last point – that the purpose of dither is to force the quantization error to be quantized to it rather than the original waveform. If the noise added is random, white noise then the quantization error will be the same because distortion of white noise yields more white noise. If, however, the dither added is not random white noise then the quantization error will be a distortion of the frequency content of the dither added and will thus yield defined harmonic activity.

For example, one could choose to use a 60Hz sine wave as dither noise. While the harmonic content of this dither is easily definable, it may be completely unrelated to the initial frequency content of the waveform. As the initial waveform changes, the 60Hz sine wave does not change and therefore it is noise and not distortion, even though it is not random and is harmonically simple. If one were to use a 60Hz sine wave as their dither, not only would they then have 60Hz added into their material, but they would also have the resultant distortion from quantizing. The quantization error in this case would be correlated to the 60Hz sine wave and would thus produce distortion that is harmonic to the 60Hz. The result is that the 60Hz waveform and its subsequent distortion all get added to the original signal by means of quantizing. Clearly this would not produce the desired results.[1]

Fixed-point: A numerical nomenclature wherein the decimal "point" stays in a fixed location. This is akin to our normal method of counting in decimal notation: 1, 100, 1400, 13,250 are all numbers where in the decimal point is at the end. This is distinguished from:

---

[1] Dr. John Vanderkooy augments the provided requirements of dither by saying that it must be "statistically independent." This brings up the very valid point that not only must dither noise be independent of the signal itself (uncorrelated) but it must also be uncorrelated to dither used elsewhere in the system (after other processes or on other data streams – tracks).

Floating-point: A numerical nomenclature wherein the decimal point "floats" like it does in scientific notation. Floating-point systems have an "exponent" and a "mantissa." The mantissa is the numerical data and the exponent indicates how many places to the left or the right the mantissa gets shifted. For example, Avagadro's Number is 6.022 x 10^{23}. This means that the mantissa (6.022) gets shifted 23 places to the left. The actual number is 602,200,000,000,000,000,000,000. Planck's Constant is 6.626176 x 10^{-34}, or .00000000000000000000000000000006626176.

Floating-point nomenclature provides two benefits: it gives us the ability to denote very large and very small numbers with less data, and it prevents us from committing to more precision than we actually have. 6.022 x 10^{23}, for example, implies that we only have four digits worth of accuracy and gives no indication of what the remaining 20 bits are. It is distinguished from 602,200,000,000,000,000,000,000, which implies that the remaining 20 digits are indeed all zeros.

The most common form of floating-point notation in the audio industry is 32-bit floating-point using the IEEE 754 protocol. In this system there are 8 bits for describing the exponent and 25 bits worth of mantissa (one of which is implied).

Dithering Fixed-Point Data

All audio must, at some point, return to a fixed-point state for converting back to analog. For this reason, all floating-point numbers must undergo bit-reduction, or "quantization." Further, after any calculation in a floating-point environment the mantissa, which will inherently have grown due to the calculation, must be returned to the permissible size, again requiring quantization. The goal is to perform this quantization without producing distortion. As we know, however, simply truncating off excess bits yields distortion, as this is the same as quantization. The error will inherently be correlated to the original waveform. We must apply dither in this situation in order to prevent the quantization error from being quantization distortion. This all brings us to the question of how to most effectively accomplish this.

Let us look at the first 20 words of a random 24-bit fixed-point signal.

Figure 1.1
24-bit Fixed-Point Data

We see the bold line after the 24th bit showing the end of the data. The pink boxes represent "valid" material. The white boxes to the left of the pink boxes represent zeros prior to the first "valid" bit.

Let us also then look at the same material in a 32-bit floating-point system.

Figure 1.2
32-bit Floating-Point Data

In this example we see the pink boxes again representing "valid" material, but each word has 24 valid bits (ignoring the implied one for the sake of explanation). While after the A/D conversion stage the bits behind the bold vertical line would be "0"s, they are still valid bits. If this signal is the result of some sort of processing or is digitally generated the bits behind the bold line would not only be "valid" but would also be "consequential" (filled with 1s and 0s).

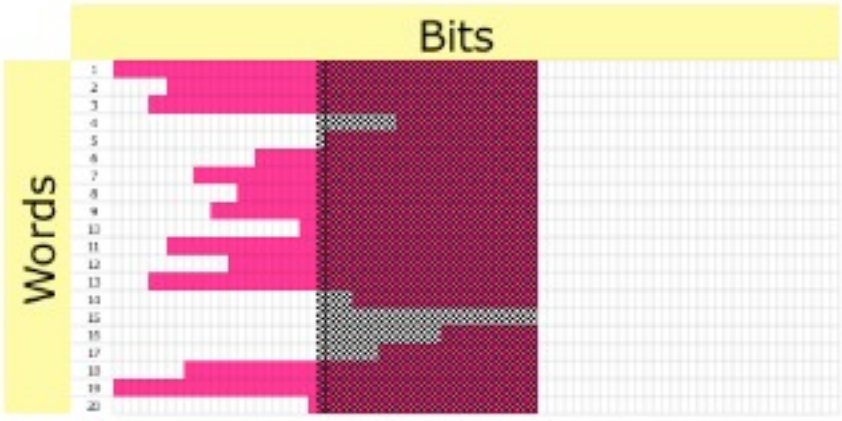
Now let us look at what happens when we process these signals. First we look at the 24-bit fixed-point example above after undergoing some form of processing.

Figure 1.3
24-bit Fixed-Point Data after Processing

The actual number of bits present is dependent upon the size of the accumulator in the digital signal processor. If the accumulator is a 48-bit accumulator the total number of bits across would be 48.

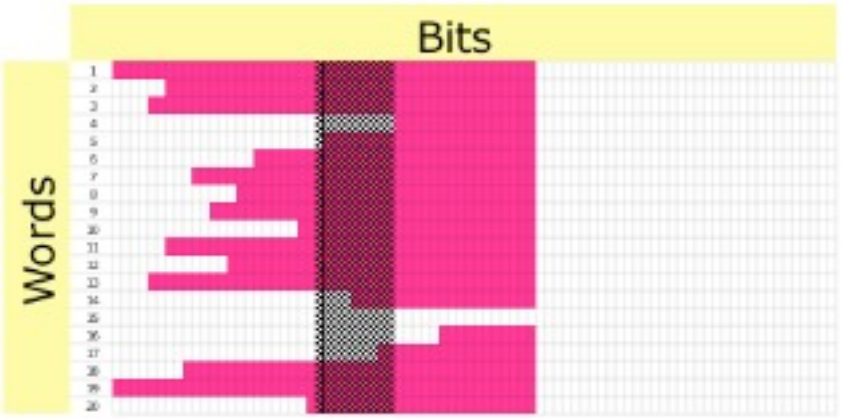
Whenever this material is reduced to 24-bits, whether after each, individual process, or only at the end of multiple processes, the method is the same. Dither is added in the form of a random noise signal. The amount of dither is enough to incorporate all valid bits, the total amplitude being the size of two of the remaining quantization steps (2 LSBs). The grey represents the dither added to the signal in the graphic below.

Figure 1.4
24-bit Fixed-Point Data with Dither



We notice that the bit-depth of the dither is equivalent to the number of bits to be re-quantized away. We also notice that the dither extends to the left of the bold line by one bit (in reality, two quantization steps, which is between one bit and two bits). One might ask what would happen if we did not add this much dither. What if, for example, dither was added that was only 8 bits “deep?” An example of this is shown below.

Figure 1.5
24-bit Fixed-Point Data with Inadequate Dither



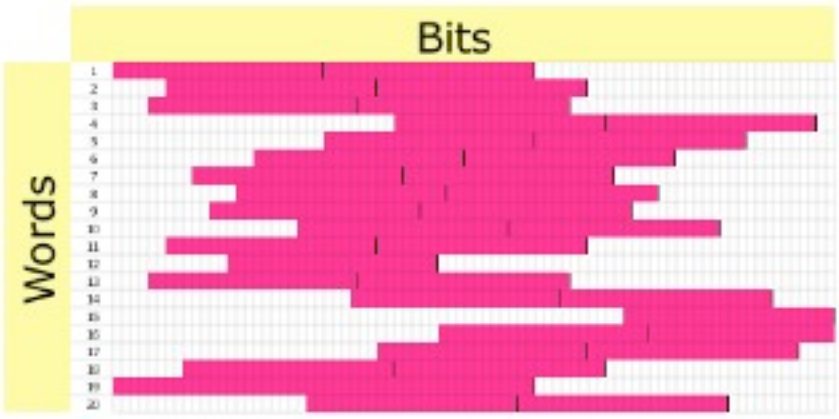
This does add dither and helps to reduce quantization distortion, but it does not completely remove distortion from the results as 16 bits worth of materially are effectively quantized in this process prior to dither being applied. This method would certainly be better than the complete absence of dither, but it is not capable of completely preventing distortion. With this solution we would still have quantization distortion but at a level 8 bits (48dB) below the noise floor of the 24-bit re-quantized data.

Dithering Floating-Point Data

This brings us to 32-bit floating-point systems. While with our 24-bit fixed-point example we were only able to utilize 48 bits worth of accuracy, in a 32-bit floating-point system we may have as many as 281 bits in use (8 bits of exponent represents the ability to shift the data as many as 256 bits to the right or the left, plus the 25 bits of the mantissa). For the ease of graphic demonstration our chart will not show 281 bits, but will show indeed more bits in use than in our 48-bit accumulator used in the fixed-point processing.

Below is the result of the same processing we did on the 24-bit fixed-point signal on the 32-bit floating-point signal. The bold lines in the middle of each word denote the 24th bit for the respective words.

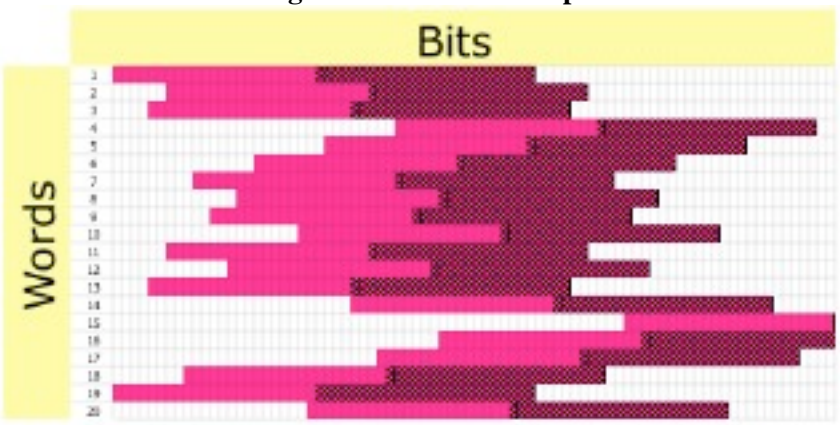
Figure 2.1
32-bit Floating-Point Data



We notice in this example that clearly there are more bits in use than in our fixed-point system – the bits all the way to the right are in use. We also notice that, just like in fixed-point processing, we created more bits through the act of processing. In this example we created an extra 24 bits for each word, so that each word now apparently has a 48-bit mantissa. Just like in a fixed-point system, however, we must reduce this after intermediate processing. After this process we may have to reduce it to 24-bits again. We may create more than 48 bits worth of data yet have to reduce it back to 48 bits of data. Either way, excess bits will be created through the act of processing that then have to be reduced. How do we do this? We will attempt to reduce the signal above to 24-bits of mantissa data again. In order to do this we must remove the last 24-bits of every word.

As we remember from our fixed-point system, we have to add dither that is of the same depth as the bits being reduced. One solution might be as follows.

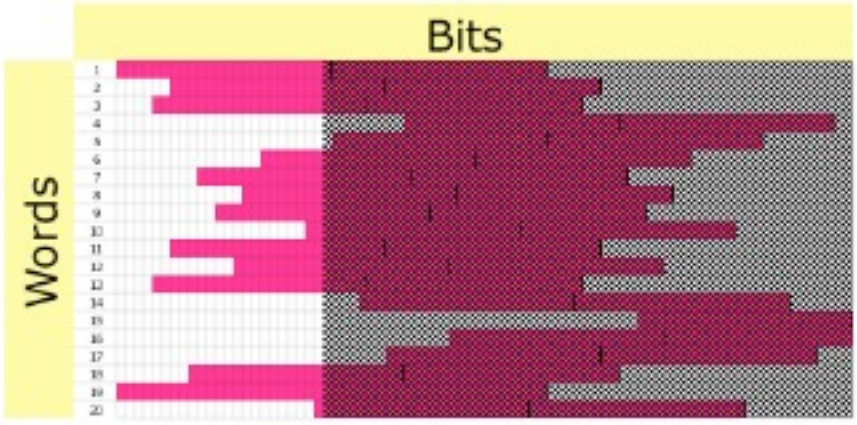
Figure 2.2
32-bit Floating-Point Data with Proposed Dither



This solution indeed adds dither of the appropriate depth to each word, with the intent of dithering those bits off. The problem is that the noise in this example is not random. The noise is directly correlated to the signal itself. The amplitude of the noise directly parallels the amplitude of the signal. As the signal uses the bits more to the left, the noise must as well. This is not random noise and will not properly dither the results. In effect, this system adds harmonic distortion (the dither noise is effectively harmonic distortion) prior to the removal of the excess bits. Because the dither signal is correlated to the signal itself it is not actually noise and is actually distortion. Clearly this does not accomplish the goals of reducing the bits through a linear process. This method will yield quantization distortion by mere results of the fact that distortion was added *prior to quantization*.

Perhaps another system would be like that below.

Figure 2.3
32-bit Floating-Point Data with Alternate Proposed Dither



In this example we clearly did add dither that is uncorrelated noise. This solution will provide the desired results. The problem with this solution is multi-fold. First, we must remember that this graphic is simplified for the benefit of graphical simplicity. In actuality this would have 281 bits worth of data, meaning that as many as 256 bits worth

of dither would have to be generated (if the data extended all the way to the “right”). A 32-bit floating-point system cannot deal with 256-bit-long numbers, as this is well beyond the capacity of the system. The system would have to generate a number 256 bits long – effectively a signal with a 256-bit mantissa. If it could handle 256-bit-long numbers the system would instead be reverted to a fixed-point system, as anything accomplishable within the floating-point nomenclature could be done in this large fixed-point system. The second reason this is inadequate is that the noise is now so loud that we can effectively truncate all of the last bits off, leaving only a 24-bit fixed-point signal. This reduces any benefit of a floating-point system.

This brings up a possible third implementation of dither.

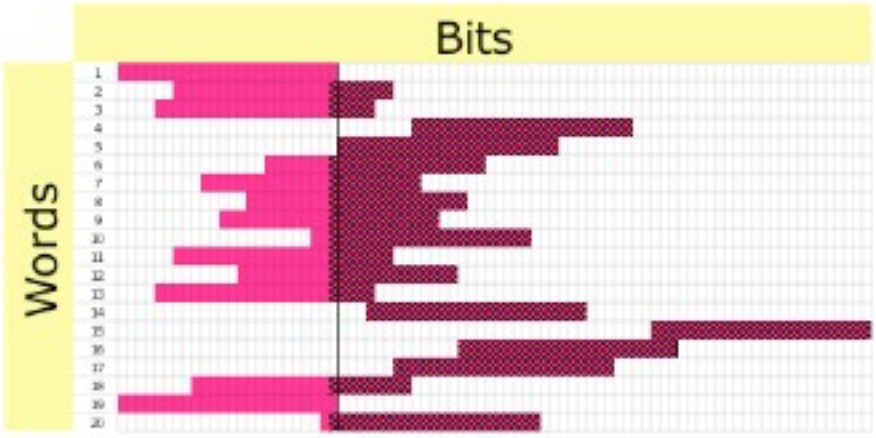
Figure 2.4
32-bit Floating-Point Data with Yet Another Proposed Dither

The problem with this implementation is that, while dither is added at a constant level, uncorrelated to the signal itself, and while it uses few enough bits to be handled by the processor, it does not adequately dither the signal. Many of these words will effectively be truncated without dither being added at all. Many of these words will therefore experience just as much quantization distortion as if dither were not added.

The sheer nature of a floating-point system negates the ability to add dither that is at a constant amplitude and thus uncorrelated to the amplitude of the signal and of the same bit depth as all bits to be truncated.

We should also consider the case that at the end of the system’s processing the signal must be reverted to a (likely 24-bit) fixed-point signal. This raises the question of how we can dither a 32-bit floating-point signal into a 24-bit fixed-point system. Again we have some examples.

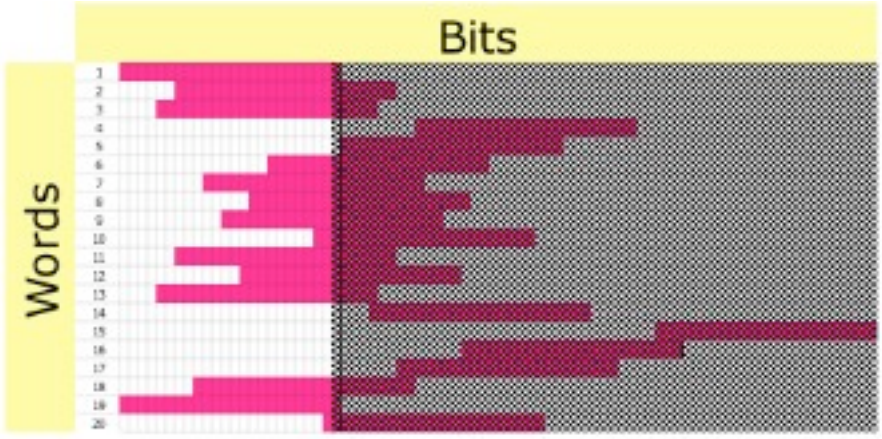
Figure 2.5
32-bit Floating-Point Data with Proposed Dither
for Reduction to 24-bit Fixed-Point



The bold vertical line again represents the 24 bits that will remain. The idea with this approach was to add dither to all bits that will be truncated. Again, however, this signal is correlated to the signal itself and is therefore not noise and will not result in quantization noise, but will instead result in quantization distortion.

Another solution is shown below.

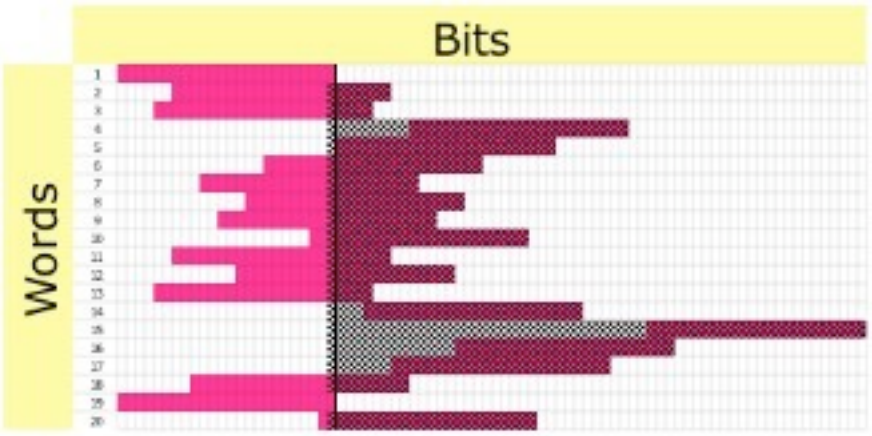
Figure 2.6
32-bit Floating-Point Data with Alternate Proposed Dither
for Reduction to 24-bit Fixed-Point



This solution again has the problems addressed in the segment above – the floating-point system cannot produce dither of enough bits to accomplish this.

Another possible solution is given below.

Figure 2.7
32-bit Floating-Point Data with Yet Another Proposed Dither
for Reduction to 24-bit Fixed-Point



This solution is not actually different than that given above. If the valid bits extend all the way to the right the system must generate more bits of dither than the system can provide. Further, the noise is not completely random as when the signal is higher in amplitude the precision of the noise is confined accordingly.

Conclusion

For these reasons it is simply impossible to adequately dither a floating-point system such that quantization error becomes exclusively quantization noise, such as can be accomplished in a fixed-point system. Having said this, it is necessary to keep in mind two things:

1. There are some methods that work better than others. Different DSP processors may give options of what to do with excess bits that may incorporate some sort of random behavior. This can greatly reduce the quantization distortion, and may even be possible to mask the quantization distortion beneath the noise that is added. It is possible, however, that distortion masked well below a given noise level may be exposed through further processing.
2. While fixed-point protocols are capable of completely removing quantization distortion in favor of quantization noise, and while fixed-point systems are therefore capable of maintaining an infinite dynamic range (though much of it is maintained below the noise floor), and while floating-point systems are not capable of either of these feats, the necessary precision of these systems is limited by the fact that there is an ultimate receiver that has its own limitations. The human ear is the eventual receiver of this material. The human ear has limitations of how low it can hear below a noise floor, as well as how low it can hear in absolute terms. The quantization distortion only need be lowered to a degree that it cannot be perceived by the human ear. Obviously, the concern is that the distortion that is not audible after one process may be exposed throughout additional processes.

A consequential amount of research is currently being conducted on the subject of "pseudo-dithering" floating-point data. The goal is finding a way in which the quantization distortion can be pushed low enough that it can be deemed inconsequential (and incapable of becoming consequential through further, normal processing) with respect to the human ear.

Clearly, the most effective method of preventing quantization distortion during intermediate processing stages is merely to use a mantissa of sufficiently larger depth than the resultant signal. One conjecture stated that eight extra bits of mantissa throughout the system and its processes would *probably* be enough to ensure that quantization distortion from intermediate processing did not have an effect on the results. With this paradigm, floating-point math would be done sans dither for all intermediate processing and would only be dithered for the final step to fixed-point for delivery. Some digital audio systems indeed use internal processing depths of as much as 64 bits (mantissa depth) for some of their intermediate processing. These larger-bit-depth systems would logically be preferable.

If the intermediate processing depth only has a 24-bit mantissa then it is questionable whether this system actually has any mathematical benefits in real-world applications

over 24-bit fixed-point systems in digital audio, specifically because of the inability to maintain the “resolution” or “mathematical precision” that a dithered 24 bit system can guarantee.

Because of the issues surrounding effective dither both mid-stream and also during the final conversion stage to fixed-point processing it is probably appropriate for users to do careful listening tests between both their available dithering options and not using dither at all. Because there is not a simple, absolute “goal” with respect to dither on floating-point systems (one cannot completely eliminate the distortion) and because there is not a simple mathematical formula for obtaining the best results (such as exists in the fixed-point domain), it is indeed possible that the available dither options will yield less desirable results than removing dither altogether. This is especially true as programmers try to independently find the most effective solution, or as they errantly apply the principles of dither in fixed-point systems to the floating-point domain.

As with many other areas of digital audio, it is probably best that the audio engineer, armed with sufficient knowledge of what is “supposed to be,” actually listen to verify that what is “supposed to be” or what is “purported to be” actually plays out in practice with their equipment.

Acknowledgement

Special thanks are due to Dr. Stanley Lipshitz, and John Vanderkooy of the Audio Research Group at the University of Waterloo, Glenn Zelniker of Z-Systems Audio Engineering, and Paul Frindle of Sony Corporation's Oxford Division for their comments and editing of this paper.

Suggested Further Reading

Aldrich, Nika. "An Explanation and Proof of the Benefit of Dither for the Audio Engineer." Cadenza Recording, 2002. http://www.cadenzarecording.com/papers.

Aldrich, Nika. "Digital Audio Explained: For the Audio Engineer." Sweetwater Press. Fort Wayne, IN, 2004. http://www.cadenzarecording.com.

Katz, Bob. "The Secrets of Dither." Digido, 2003. http://www.digido.com.

Lipshitz, Stanley P., and John Vanderkooy. "Dither Myths and Facts." Audio Engineering Society Convention Paper 6279. (2004)

All text and pictures are
Copyright© 2005 Nika Aldrich - Cadenza Recording
Not reproducible without written permission.
Links to this site are encouraged, and written permission will be readily granted with proper credit given.