

Nyquist Revisited: How Many Samples Are Really Necessary?

by Al Wegener, *Samplify Systems*

DSP newcomers and veterans alike have at least a passing familiarity with the Nyquist sampling theorem named after the Bell Labs physicist, Harry Nyquist, which states that a signal having bandwidth B Hz must be sampled (converted from the analog to the digital domain) at a rate of at least $2 \times B$ samples per second. Similarly, most engineers have at one time or another derived the *6 dB-per-bit* rule, verifying that every bit of resolution in an ADC or DAC represents 6 dB of dynamic range (DR), where DR is the difference (in dB) between the largest signal and the smallest signal that can be represented by the converter. The derivation is simple and intuitive (Eq. [1]), since each additional bit in a data converter increases the magnitude of each sample by a factor of 2:

$$20 \log (2.0) = 6.02 \text{ dB} \quad [1]$$

There's actually an additional 1.76 dB in the signal-to-noise (SNR) equation ($\text{SNR} = [n \times 6.02] + 1.76 \text{ dB}$). This additional term reflects the quantization error ($\frac{1}{2}$ LSB divided by $\sqrt{3}$), but engineers drop the 1.76 dB to annoy the mathematicians.

Let's consider the effect of these rules in practical, everyday language, using a typical high-speed ADC, the Analog Devices AD9233, which operates up to 125 Msample/s and has a resolution of 12 bits per sample. From the Nyquist theorem we conclude that the AD9233 can uniquely represent signals having a bandwidth of 62.5 MHz (125 divided by 2). Similarly, the 12-bit resolution of the AD9233 should allow us to represent signals having a dynamic range of 72 dB. The combined data rate of the AD9233 based on our two favorite DSP rules is thus 125 Msample/s \times 12 bit/sample = 1.5 Gbit/s (192 Mbyte/s). In other words, it appears that sampled data systems that use the AD9233 must size all of their signal processing interfaces to handle nearly 200 Mbyte/s of data. Fig. 1 illustrates a signal having a 60 MHz bandwidth and a 70 dB DR that should be adequately sampled by the AD9233 at its maximum sample rate of 125 Msample/s.

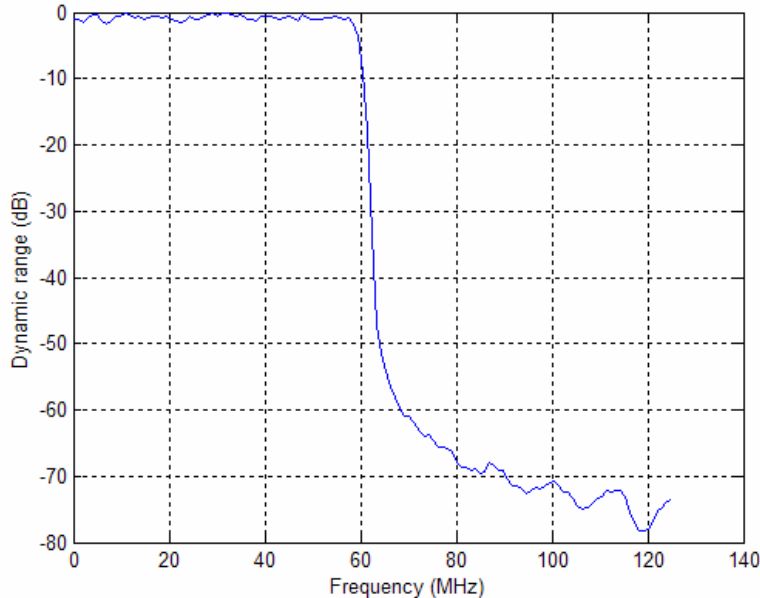


Fig. 1: Signal With 60 MHz Bandwidth And 70 dB DR

The 6-dB-per-Bit Rule Vs Measured SNR And ENOB

It often comes as a surprise to DSP newcomers that an n-bit ADC does not provide $6 \times n$ dB of SNR. Wasn't this the whole point of the 6 dB-per-bit rule? In fact, depending on the converter, an n-bit ADC may have an SNR as low as $(n - 2) \times 6$ dB.

The difference lies in practical, real-world difficulties of implementing ADCs and DACs. Data converters are physical devices whose SNR is always worse than the 6-dB-per-bit rule predicts. In fact, the SNR of a data converter is only approximated by the rule. To get the exact SNR, a converter must be carefully measured using a high-quality, full-scale sine wave input whose power is then compared to the sum of the power in all other spectral components (except the dc component). Data converter SNR is a measured value, while the 6-dB-per-bit rule is just an estimate. And measured SNR is what matters.

Let's compare the SNR specifications of a few ADCs to illustrate the difference between predicted SNR and measured SNR. National Semiconductor ADC14C105 (14 bit at 105 Msample/s) has a measured SNR of 74 dB, which is 10 dB less the theoretical $6 \times 14 = 84$ dB that the 6 dB-per-bit-rule would lead us to believe. In contrast, the aforementioned AD9233 (12 bit at 125 Msample/s) has a measured SNR of 69.5 dB, just 2.5 dB below the theoretical $6 \times 12 = 72$ dB. A third converter, the Linear LTC2207 (14 bit at 105 Msample/s) lies in between: its SNR of 77 dB is 7 dB below the 84 dB SNR predicted by the 6 dB-per-bit rule.

If the measured SNR of a ADC or DAC doesn't conform to the 6 dB-per-bit-rule that's based on the converter's resolution, how many bits per sample do data converters actually deliver? Converter manufacturers have developed a metric called the *effective number of bits* (ENOB) that relates the measured SNR (actually the converter's SINAD) to the *effective number of bits*. While SNR is the marquis ADC and DAC specification, ENOB is measured using a more stringent metric called signal-to-noise and distortion, or SINAD. While SNR excludes the first 5 harmonics (nonlinearities generated by the converter), SINAD includes the distortion of those harmonics and is thus a better predictor of converter performance than SNR. ENOB is calculated as shown in Eq. [2]:

$$\text{ENOB (bits per sample)} = (\text{SINAD} - 1.76) / 6.02 \quad [2]$$

Reference [1] is an excellent introduction to the importance of ENOB in test and measurement applications.

Nyquist Vs Oversampling

For many reasons, engineers regularly add margin -- sometimes significant margin -- over and above the Nyquist sample rate. Few systems ever sample at EXACTLY twice the signal bandwidth. The reasons for sampling faster than Nyquist are varied and interesting, and we'll consider a few of those reasons here.

First of all, engineers familiar with DSP know about a phenomenon called aliasing. If a signal is sampled at less than the Nyquist rate, unwanted frequencies above the Nyquist frequency will alias (fold into; interfere with) the desired signal components in the band of interest. Aliasing is to be avoided at all costs, because it can't be corrected. For this reason, ADCs should always be preceded by analog anti-alias filters, and DACs are always followed by reconstruction filters that remove aliasing components. Being analog components, anti-alias filters cost money and consume power and board space. As in most things in life, sloppy anti-alias filters are less expensive than good anti-alias filters. So if analog designers can use a sloppy anti-alias filter with a 12 dB per octave roll-off, they will save money when compared to a 24 dB per octave filter. Oversampling allows system designers to trade off sloppy (cheap) for good (expensive) anti-alias filters, just by oversampling the signal by 2x to 8x more than the Nyquist rate.

As the resolution of ADCs increases, the anti-aliasing stop band rejection requirements increase by 6 dB per bit to keep the power of the aliased image below the noise floor. For example, an 8-bit ADC requires 48 dB of rejection from its anti-aliasing filter at the frequency of the aliased image. As resolutions of high-speed ADCs increase to 12 bits, 14 bits, even 16 bits, the out-of-band rejection requirements increase to 72 dB, 84 dB, and 96 dB, respectively. This has obvious, negative implications for the complexity and cost of the anti-aliasing filter. To relax these requirements, system designers often choose to oversample even further, which decreases anti-alias filter component cost but creates a quadratic increase in the ADC output bit rate.

The delta-sigma converter folks take oversampling to the extreme, especially in audio applications where phrases like *128x oversampling* turn oversampling into a marketer's dream of "more (oversampling) is better." In fact, delta-sigma converters intentionally use oversampling to shape the quantization noise, pushing it into higher frequency bands where it is less noticeable (audible). More importantly, noise-shaping actually increases the DR of desired low-frequency signals.

By the way, this trend towards higher oversampling can also be observed in the wireless infrastructure domain. While 3G (W-CDMA) basestations designed in 2002 used converters at 30.72 Msample/s (I & Q), wireless infrastructure designers now (2007) select converters that operate at 122.88 Msample/s, despite the fact that the four-carrier, 20 MHz W-CDMA bandwidth hasn't increased. Oversampling is used in wireless infrastructure for several reasons, one of which is "because they can." After all, aren't more samples always better? Read on, and consider the alternatives.

Oversampling is also often used in test and measurement applications to improve the accuracy of important measurements. For instance, when a Tektronix or Agilent high-speed scopes measure rise time, fall time, eye diagram, and jitter specifications, the limited sample rates of scope ADCs (40 Gsample/s at 8 bit/sample) may only capture a few samples per rising and falling edge. For example, if a 40 Gsample/s scope digitizes a 6.25 Gbit/s SerDes link, the resulting sampled waveform only has two or three samples on each rising and falling edge. If the low voltage of the SerDes waveform represents the 0% level and the high voltage represents 100%, each of the three edge samples represents just a 30% rise/fall time resolution. In order to measure the 10%-to-90% rise time and the zero-crossing time with the required 5% to 10% resolution, the captured samples must first be interpolated (oversampled) before the measurement can be performed. Reference [2] describes this process in more detail, suggesting that 3 - 5 samples per edge before interpolation will suffice.

In summary, all real-world DSP systems fail to sample at Nyquist rate and instead intentionally oversample by modest to excessive factors, for very practical reasons.

What About Compression?

What effect does compression have on the Nyquist criteria and the 6-dB-per-bit rule? To explore these concepts, let's consider the effects of MP3 compression on the bit rate of audio signals -- the bit rate predicted by our good friends Nyquist and 6-dB-per-bit.

If Nyquist and 6-dB-per-bit were the only factors in play, audio signals would all be represented at a bit rate that only preserves the properties of human hearing. After all, why sample parts of audio signals that people can't hear anyway? Psychoacoustic studies in the 1940s determined that humans can distinguish frequencies up to about 20 kHz, which drops to 15 kHz or less as we age. Similarly, human hearing has a DR of 130 dB, a number that varies with frequency. The dynamic range of hearing also reduces with age, depending on the aggregate number of rock concerts we've attended? The 20 kHz bandwidth of human hearing is what drove the original CD sampling rate of 44.1 ksample/s, which is a little higher than the Nyquist criteria of 2×20 kHz. 16-bit samples were chosen for CDs because, well, 16-bit converters at 44.1 ksample/s could be built at reasonable cost in the 1980s when CD players were first sold. The 96 dB of (theoretical) DR from 16-bit sampling was good enough for CD audio, despite the current trend that claims 24-bit audio, which is simply a marketing fantasy. For instance, one of the best audio ADCs, the Cirrus Logic CS5381, has just 120 dB SNR, about 20 bit ENOB. There simply are no 24-bit ADCs or DACs operating at 44.1 ksample/s or higher that have measured ENOB anywhere close to 24×6 dB. So it is certainly feasible to represent high-quality audio (good enough for all but golden-eared audiophiles) using $44.1 \text{ ksample/s} \times 16 \text{ bit/sample} = 705.6 \text{ kbit/s}$ for a mono audio CD channel, and twice that rate for stereo.

Now compare that raw audio rate of 1411 kbit/s with MP3 rates, where users can select the desired, compressed bit rate from a palate of choices: 192 kbit/s, 128 kbit/s, 96 kbit/s,

and 64 kbit/s. Don't the MP3 folks realize that at these low rates, they are violating both the Nyquist criteria and the 6-dB-per-bit rule – simultaneously, by factors of 10x to 20x? What were these audio compression gals and guys thinking, violating our two sacred DSP rules?

Actually, the audio compression community was thinking very creatively, and very cost-effectively. Anyone who's used an MP3 player or a digital camera or a DVD should thank compression researchers for their creativity and inventiveness. The compression community has done nothing less than figure out how to obey the Nyquist criteria and the 6-dB-per-bit rule, but using a lot fewer bits. It now appears (and this article makes the general claim) that the Nyquist criteria and the 6-dB-per-bit rules are just guidelines – starting points – that can be bypassed by clever, research-backed compression techniques, perhaps regardless of sample rate and bits per sample.

What the audio compression community realized in the late 1980s is that not all audio frequencies are equally important, and that not every bit of every sample must be preserved. What audio compression researchers focused on was preserving *simply the bits that matter* – the parts of the audio signal that humans can hear. If a compressed audio stream can be represented using fewer bits than those legislated by the Nyquist criteria and the 6-dB-per-bit rule, why not do so, especially since compression saves bandwidth and memory? Why not use the best research about human hearing to literally throw away features of the original linearly sampled data stream, creating a lower bit rate at equivalent quality? Why not give users a *knob* to determine the tradeoff between how many songs they can store in their iPods, Vs the audio quality of each song? The audio market has decided that for its listening habits, violating the Nyquist criteria and the 6-dB-per-bit rule is not only acceptable, it's preferable.

Nyquist And 6-dB-per-Bit Are Overkill For *Equivalent Results*

Perhaps the benefits of compression are limited to signals that humans consume: speech, audio, images, and video. After all, these signals may be unique because engineers can determine, through laborious and costly subjective human testing, what *equivalent results* (transparent audio and video coding) mean for compressed audible and visible signals. But what would the equivalent results criteria look like, if they weren't measured by human ears and eyeballs?

Let's consider a radical question: can compression possibly allow DSP engineers to significantly decrease the bit rate determined by the Nyquist and 6-dB-per-bit rules in the general case? Can a concept of equivalent results using compression be extended to signals other than audible and visible signals: signals having orders of magnitude higher bandwidths and equal or slightly reduced dynamic range requirements? If compression techniques have allowed speech, audio, and video signals to sidestep the bit rate determined by the Nyquist criteria and the 6-dB-per-bit rule, is it possible that other DSP systems – all DSP systems – could similarly sidestep the bit rate in other applications?

Recent compression research [3], [4] indicates that it is now possible to apply compression in real time at very high sample rates, from 20 Msample/s to 40 Gsample/s, and for any bit widths between 4 and 24 bits per sample. Furthermore, this new research shows [5] that a concept called *equivalent results* allows engineers to evaluate the amount of compression that their own signals make available to them. Engineers already have many system metrics they try to meet – bit error rate (BER), spectral masks, adjacent channel leakage ratios, rise and fall times, jitter measurements, etc. Why not use those system metrics to determine the effects of high-speed compression? Why not crank up the compression knob (lower system costs, increase system bandwidths) until the system metrics are just within specification, gaining additional bandwidth and storage as a result?

This is the idea behind equivalent results – using high-speed compression to attain higher bandwidths, lower costs, lower pin counts, smaller board areas, lower power, less memory, smaller disk drives – while still meeting all system metrics.

How could this be possible? What signal redundancies does a high-speed compression algorithm exploit? Let's go back to some fundamental characteristics of many signals: not just audio and video signals, but most real-world signals.

First, remember the oversampling discussion? Oversampling always introduces redundancies, redundancies that can be removed by compression. Second, remember the difference between a converter's resolution and its ENOB (the bits that matter)? Compression can borrow a trick or two from audio noise reduction research, because clean (high SNR) signals will compress better than noisy (low SNR) signals. One should aim to remove as much noise as possible before compressing high-speed signals. Third, many of today's popular modulation types – for instance, those utilizing spread spectrum (CDMA) techniques, or orthogonal frequency division multiplexing (OFDM) – have a very high peak-to-average ratio. High peak-to-average ratios matter for compression, because a high peak-to-average ratio implies that the signal needs the full DR of the data converter just a small percentage of the time. If a system using signals with high peak-to-average ratios need a 12-bit converter most of the time (98%) for the average signal, but only occasionally (2%) need the dynamic range of a 14-bit converter for the peaks, shouldn't the effective bit rate from the converter be $(0.98 \times 12) + (0.02 \times 14) = 12.04$ bits per sample? Could high-speed compression algorithm discover a signal's peak-to-average characteristics to reduce the bit rate of data converters? These are the three fundamental ideas that this new compression algorithm exploits. It appears that these techniques are indeed generally applicable to many sampled data signals in defense, aerospace, test and measurement, medical imaging, automated test, and data conversion applications.

Table 1 provides signal-specific compression results and lists typical equivalent results metrics for a few of these applications.

Application	Signal Type	Sample Rate	Bits per Sample	Equivalent Results Metric	Compression ratio
Wireless infrastructure	W-CDMA test model	122.88 Msample/s	16	ACLR, cEVM	2.1:1
Medical imaging	CT scanner	3 ksamples/s	20	Lossless	1.8:1
Test & measurement	6.25 Gbit/s SerDes	40 Gsample/s	8	Rise/fall time, jitter	6:1

Table 1: How Compression Reduces Nyquist/6-dB-per-bit Requirements

Why not decimate an oversampled system back to the Nyquist rate? Can we trade off the complexity of compression and just implement decimation to achieve the same equivalent results? Obviously a simple two-tap moving average would enable decimation by two, but the $\sin(x)/x$ roll-off creates distortion in the Nyquist band. Consequently more complex decimation filters are required, particularly as the bit width increases.

What about IF sampling? In cases where the signal of interest's center frequency is not known a priori, decimation and or digital down-conversion techniques fail. In contrast, the aforementioned high-speed compression techniques are agnostic to the center frequency, or bandwidth, of the signal. In contrast to filter-based approaches that discard information that might have been of interest, compression preserves the entire signal bandwidth. Filtering and compression solve different problems, and both are appropriate for different applications. These recently-introduced high-speed compression techniques can become an additional tool in the DSP designer's toolbox, techniques that offer benefits that simple filtering can't offer.

How Well Does High-Speed Compression Work For My Signals?

If this concept of equivalent results is going to have legs, high-speed compression has to offer a variety of compression modes, because the performance required to achieve equivalent results will vary from user to user and from application to application. Sometimes lossless compression will be required, where the decompressed samples are bit-for-bit identical to the original samples. Sometimes a fixed data rate is needed, but as with MP3 audio compression, a fixed bit rate creates signal quality that varies a little over time. And sometimes users will want a guaranteed quality (dB SNR or distortion level), but they can tolerate some variation in the compressed bit rate over time.

Samplify high-speed compression offerings support all three of these compression modes. Users can experiment with these modes using software evaluation tools (Samplify for Windows and Samplify for MATLAB). The compression results from these software tools are bit-for-bit identical in all compression modes (lossless, fixed rate, fixed quality) to those that Samplify FPGA netlist (Reference [6]) delivers.

Conclusion

High-speed compression provides a way for DSP engineers to significantly reduce the system bit rate predicted by Nyquist and the 6-dB-per-bit rule. By removing various types of redundancies caused by oversampling and by monitoring varying signal parameters, the Simplify algorithm enables FPGA users to deliver equivalent results in their systems while also reducing bandwidth, storage, power, pin count, and area requirements. By enabling compression solutions at sample rates up to 40 Gsample/s, high-speed compression unlocks new bandwidth and storage for DSP users in many applications where compression was previously not a possibility.

References

- [1] Tim Ludy, *Overall Accuracy = ENOB (Effective Number of Bits)*, Data Translation application note <http://www.datx.com/resources/dataacquisition/enob.pdf>
- [2] TDSJIT3 v2 *Jitter Analysis Application Online Help*, Tektronix, 077-0023-01, p 38
- [3] Al Wegener, *Simplify: Real-time Compression for Electronic Measurements*, Proceedings of the Data Compression Conference, Snowbird, UT, March 2007
- [4] Lee Goldberg, *Simplify Launches Ultra-high-speed FPGA-based Data Compression Technology*, programmablelogicZONE, April 2007 http://www.en-genius.net/site/zones/programmablelogicZONE/product_reviews/plp_041607
- [5] Al Wegener, *Equivalent Results: A methodology to measure the effects of high-speed compression*, PLD DesignLine article <http://www.pldesignline.com/showArticle.jhtml;jsessionid=NL1W1R5EJQULGQSNDRCKH0CJUNN2JVN?articleID=199601247>
- [6] Simplify FPGA netlist data sheet: http://www.simplify.com/images/SimplifyFPGA_040307.pdf

About The Author

Al Wegener is Chief Technology Officer and founder of Simplify Systems, a venture-funded Silicon Valley start-up whose patented compression solutions reduce bandwidth and storage bottlenecks in sampled data systems. Mr Wegener is a DSP engineer, technical manager, and inventor with more than 25 years of experience in defense electronics, professional and consumer audio, and wireless applications. Prior to founding Simplify Systems, Mr Wegener was technical manager of Texas Instruments Palo Alto ASIC design center (formerly Graychip). Al holds a BSEE from Bucknell University and an MSCS from Stanford University. He can be reached at awegener@simplify.com

as published in . . .

