Episode 2.7 – The Effect of Sampling Rates on Digital Signals

Welcome to the Geek Author series on Computer Organization and Design Fundamentals. I'm David Tarnoff, and in this series we are working our way through the topics of Computer Organization, Computer Architecture, Digital Design, and Embedded System Design. If you're interested in the inner workings of a computer, then you're in the right place. The only background you'll need for this series is an understanding of integer math, and if possible, a little experience with a programming language such as Java.

Imagine you're passenger in a car driving under the street lights at night. Glancing over at the car riding in the lane next to you, you become mesmerized by its spinning wheels. Both the vehicle you're in and the one next to you are moving quite quickly, yet under the lights it looks as if the other car's wheel is spinning slowly, or perhaps it stops and maybe reverses direction. Your eyes are playing tricks on you, right? Actually, you're experiencing something called aliasing.

Before we get to aliasing, let's take a moment to discuss sampling. In order to reduce the infinite information of a continuous-time signal down to finite quantities storable by a computer, we need to take discrete measurements at a set time interval. Each measurement is referred to as a sample while the fixed rate at which the samples are taken is referred to as the sampling rate. The sampling rate has units of hertz, which in this case represents the number of samples captured per second. This sequence of samples can be stored to the computer as a list of values. If we know the original sampling rate, we can connect the dots in order to reproduce the original analog signal. Or can we?

Two consequences result from sampling an analog signal. First, since we are using a finite number of bits for our sample, each measurement won't be identical to the analog signal's level. It will be close, but depending on the number of bits we're using to represent the measurement, there will be anywhere from a minute to a notable difference between the original value and the stored measurement. Second, if we are not taking samples fast enough, we will miss important details happening between the measurements and potentially create new unwanted signal components. Let's begin with the second problem.

Except for certain analog signals such as the tone produced by a tuning fork, it is rare to find pure sinusoidal waves in the real world. Most analog signals are made up of the summation of many sine waves with changing levels at different frequencies. For example, picture, or rather hear, the difference between an A note played by a tuning fork and the same note coming from a guitar or maybe a clarinet. You can tell the difference because of the additional frequencies or overtones present in the same note played by the guitar or the clarinet. In the case of musical instruments, most of those additional frequencies are multiples of the fundamental frequency, in this case, the 440 hertz of the A note above middle C.

Sticking with sound for the time being, let's listen to a 440 hertz signal. Just like the sound of my voice, the soundwaves are just compressions and rarefactions passing through the air. Unlike my voice, however, if we were to plot the strength of the displacements of this tone over time, they would look like a pure sine wave. In the case of our A4, these pressure waves would have a frequency of about 440 hertz.

Now suppose that we take samples of this tone at exactly 440 samples per second. Every sample would occur at the exact same point on each period of the sine wave. According to the list of stored samples, the sound has vanished. The same would be true if we sampled at half the frequency or 220 hertz where the sample would occur at the same point on every other period of the sine wave. This is equivalent to the wheel of the car next to us appearing to have stopped under the streetlights at night.

If we were to sample just a bit faster than the frequency of the sine wave, then each measurement would occur just a bit earlier along the sine wave slowly backing up to the trough of the sine wave, then rising to its peak, and then coming back to its starting point. If the samples are plotted and connected like dot-to-dot, these measurements would produce a sine wave with a far slower frequency than that of the original signal.

Alternatively, if we were to sample just a bit slower than the frequency of the sine wave, then each measurement would occur a bit later along the sine wave slowly inching forward along the next period of the sine wave. Plotting the samples captured in this case would also produce a sine wave with a far slower frequency than that of the original signal. This is what is happening when the car's wheel appears to be moving slowly forwards or slowly backwards.

This phenomenon, when a sampling rate that is too low causes significant frequency shifts, is called aliasing. It turns out that all of the frequencies contained in an analog signal that are more than half the sampling rate will alias. At best, if the frequency is a multiple of the sampling rate it will disappear. At worst, new frequencies will be created in the list of samples that were not part of the original signal. Two things must be done to avoid frequency-related issues when it comes to sampling an analog signal. First, we must sample fast enough to capture everything we wish to store. How fast is fast enough? Well, a theorem called the Nyquist–Shannon Sampling Theorem states that an original analog waveform can be reconstructed if the waveform is sampled over twice as fast as highest frequency component contained in that signal.[1]

We can see this theorem applied to our everyday technology. For example, the typical sampling rate for digital audio is 44,100 samples/second. Dividing this number in half gives us the highest frequency that digital audio can capture, i.e., 22,050 hertz. This seems suitable as the frequency range a typical healthy young person can hear is about 20 hertz to 20,000 hertz.[2] The sampling rate for telephone audio is 8 kHz. Applying the Nyquist–Shannon Sampling Theorem shows us that the frequencies capable of being transmitted by telephone are capped at 4,000 hertz. This corresponds to research performed by Bell Telephone who determined that a frequency range from 200 to 3,200 hertz was sufficient for someone to recognize a caller's voice. At a 8,000 hertz sampling rate, all of those frequencies should be capable of being reconstructed.

So why wouldn't we want to increase the sampling rate for telephones in order to achieve better sound? There are two reasons, and both point back to cost. First, doubling the sampling rate would double the number of samples, and hence, double the amount of data to transmit or to store. That would cut in half the number of calls that could occupy a fixed bandwidth. Second, increasing the sampling rate would also require faster analog to digital converters, and while a sampling rate of 44,100 hertz is easily obtained with off the shelf parts, they aren't necessary if we are only taking 8,000 samples per second. The second thing that needs to be done to avoid frequency-related issues when it comes to sampling is to remove any higher frequencies that we don't wish to appear as aliased frequencies in stored data. Applying the Nyquist–Shannon Sampling Theorem in reverse, we see that for a fixed sampling rate, the analog signal cannot contain any frequencies higher than a half the sampling rate or aliasing will occur.

To prevent this, we use analog filtering to reduce the higher frequencies of the signal before we sample it. These analog filters are the domain of electrical engineers, and well, beyond what we're trying to present in this series.

That brings us to the end of another episode of the Geek Author series on Computer Organization. In our next episode, we will look at the second problem introduced when we read an infinite signal with a finite computer: quantization noise. This is the effect produced by converting the analog measurements to discrete digital values using a limited number of bits. If you're interested in getting a quick preview, search the Internet for 16-color images. It's not a pretty sight.

For transcripts, links, or other podcast notes, please check us out at intermation.com where you will also find links to our Instagram, Twitter, Facebook, and Pinterest pages. Until then remember that while the scope of what makes a computer is immense, it's all just ones and zeros.

References:

1 – Shannon, Claude E. (January 1949). "Communication in the presence of noise". Proceedings of the Institute of Radio Engineers. 37 (1): 10–21.
2 – Rosen, Stuart (2011). Signals and Systems for Speech and Hearing (2nd ed.). BRILL. p. 163.