Putting John Doe to work

How to use aliasing in testing semiconductor devices

By Dan Bullard, Marketing Applications Engineer, Nextest Systems Corporation

In the electronics world, nothing is so feared as aliasing. Few understand it and many simply refuse to believe it exists. Often someone will misquote the Nyquist[1] criterion and say that you can't see anything beyond the Nyquist frequency just like you can't see behind a mirror. Some of us however, have seen behind the mirror, like Penny in Lost In Space's "The Magic Mirror" we have traversed the barrier and found a new world where few dare to tread. For those who risk it, the journey can be very rewarding indeed.

Wagonwheels and stroboscopes

Often when explaining aliasing I have invoked various visions of wagonwheels, automotive strobe lights, fans in front of televisions, etc. While these examples can be useful, they only explain the effects of aliasing as observed in time domain. While we believe that we experience the world in time domain, it just isn't so. Our ears work in frequency domain, and we don't seem to question the fact that girls voices are "higher" than men's voices. Think about what that means, *higher* does not mean that girls stand on stepladders when they speak. It's intuitive, girls voices use tones that vibrate faster than men's voices because their vocal chords are smaller. However, we don't perceive it as faster, we perceive it as higher in the spectrum of speech. Our ears send us signals about where in the audible spectrum a particular sound belongs, our ears are a pair of spectrum analyzers. Frequency domain is natural to us, but many still have trouble grasping the idea. Let's quickly review some sampling theory so we can wrap our brains around this concept and move forward to understand aliasing in it's native domain.

I'll see your N and raise you an M

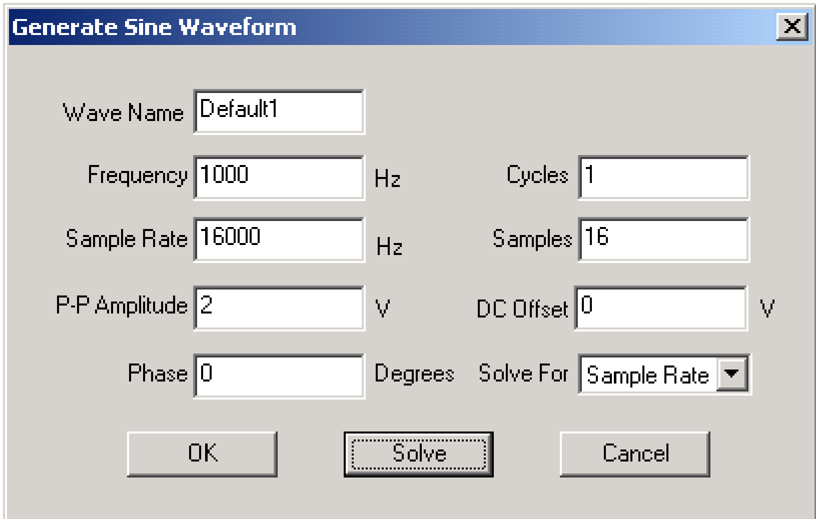
OK boys and girls, sampling is easy, the math never goes beyond what you can do on a one dollar calculator, so spend a buck and follow along.

When we sample, we capture a signal known as F_T (the Test frequency) along with other stuff (noise, harmonics, etc). We capture this signal by clocking an ADC at a frequency known as F_S, otherwise known as the sampling frequency. When we capture the signal we grab some finite number of samples, known as N and stuff them into a memory. N can be any integer, but we tend to use values of N that are powers of two (2, 4, 8, 16, 32, 64, 128, …4096, 8192, etc) because the algorithm for converting to frequency domain is faster if we use these values. You can capture any number of samples, even an odd number, but it will take longer to convert to frequency domain.

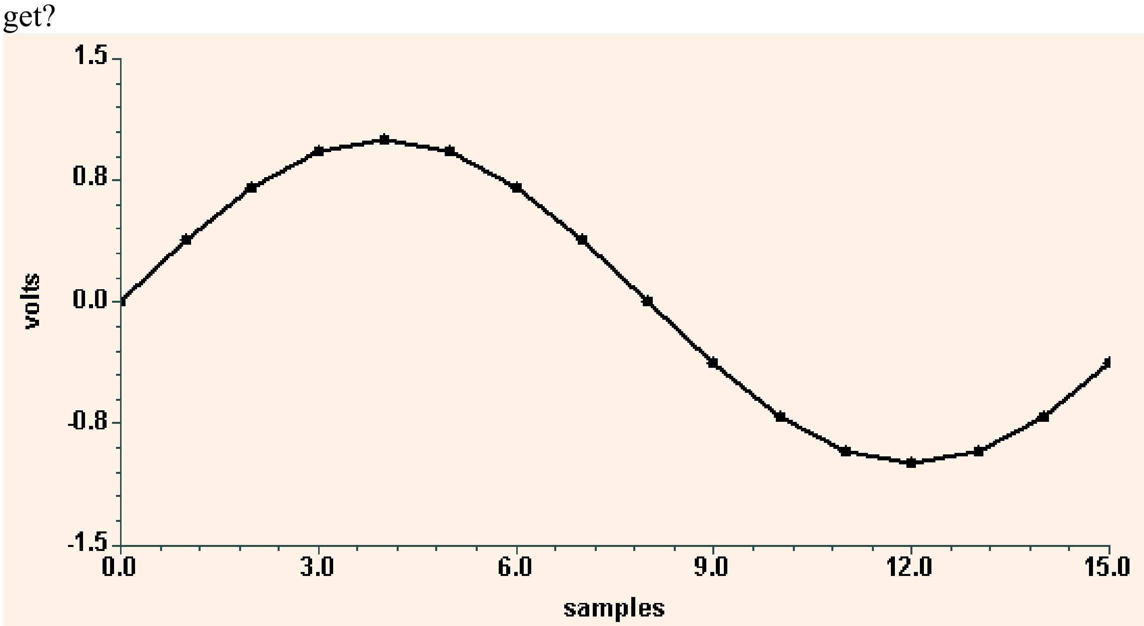
So, if you capture a signal called F_T by taking N measurements at a rate of F_S and stuff the N measurements into a memory, something funny happens. You end up with M cycles of the signal in your sample memory. Let's take a simple example, Here is the

[1] Harry Nyquist was a pioneer in telecommunications, whose name is actually an alias. His family name was Jonsson but Harry's father, Lars changed the family's last name because there was another Lars Jonsson just down the road and mail delivery became a real problem.

dialog box from Nextest's Mixed Signal Wave Tool (MSWT) that is used to generate a sine wave.



I've filled in the values to generate (or sample) a 1KHz sine wave (Frequency = 1000 Hz) sampled at 16KHz (Sample Rate = 16000 Hz). In this dialog, Frequency is F_T, and Sample Rate is F_S. On the right side of the dialog you can see that Samples is set to 16, that's N, and yes, it's a power of two. Above that is M, which is 1. M is usually an integer, but it doesn't have to be. So what happens if we hit the OK button, what do we get?



This is a single cycle of a sine wave with a frequency of 1KHz, consisting of 16 samples. The samples are separated in time by 1/16KHz (or 62.5uS), since they were captured at a rate of 16KHz. Why is there only one cycle? Because we captured 16 samples at a rate 16 times greater than F_T. How many cycles would you expect there be? If we sample every 62.5uS then 16 samples later we will have been capturing for 16*62.5uS or 1mS, which

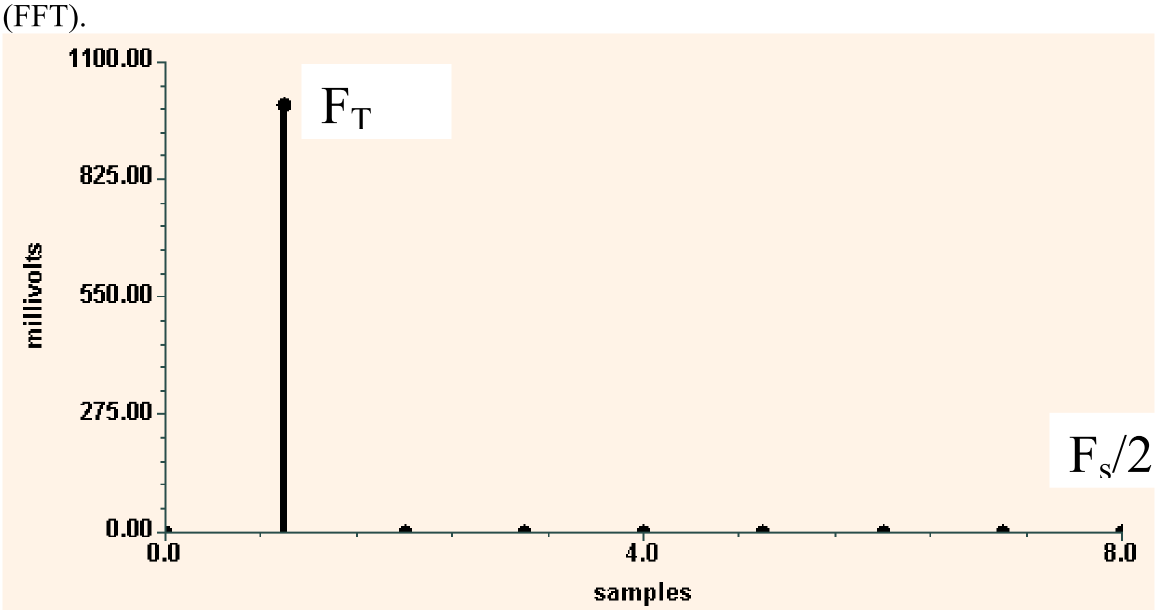
(not coincidentally) is how long it takes for 1 (M) cycle of 1KHz (F_T) to go by. It's pretty simple, isn't it?
OK, here's the math part, are you ready? First the magic formula

$$\frac{F_T}{F_S} = \frac{M}{N}$$

To solve for M, $M = N*F_T/F_S$
To solve for N, $N = M*F_S/F_T$
To solve for F_T, $F_T = F_S*M/N$
To solve for F_S, $F_S = F_T*N/M$
Not exactly rocket science.
Well, here is the rocket science part, let's convert to frequency domain.

Spectral images

A long time ago, a guy named Fourier (For-ee-a, it's French) who was smarter than you or me figured out how to move from time domain to frequency domain. Lots of arrogant people think you should understand how Fourier did it, but if you are only going to use his trick and not reinvent it do you really need to know the details? Probably not. So let's look at our sixteen sample sine wave after we run it through a Fast Fourier Transform (FFT).



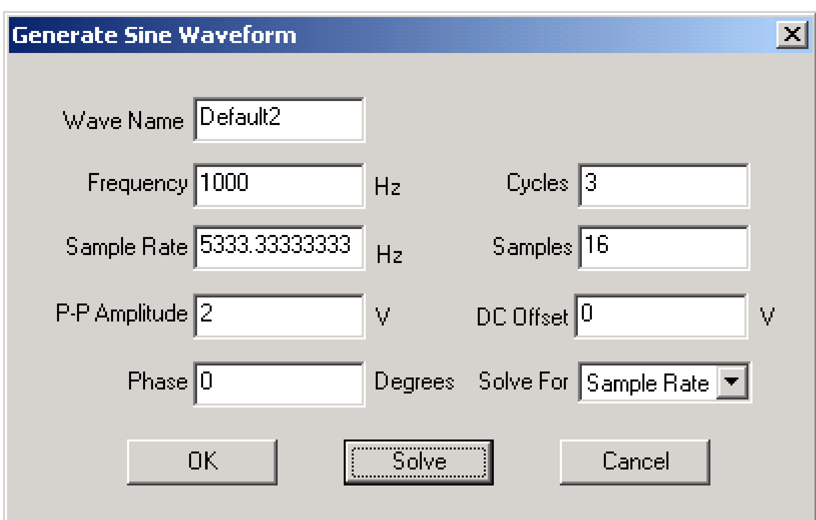
Note that there are nine samples (0 through 8) in this spectrum. Notice the highest sample (called a bin, since this is essentially a bin histogram) is numbered 8. That's N divided by 2 (16/2=8). Bin N/2 defines the upper bounds of the spectrum. Bin zero is where any DC voltage would appear, if we had any. In this case it appears there is none. Bin N/2 is where any signal at $F_S/2$ would appear, but it seems there is nothing there either. In fact the only place where there is any amplitude is in bin 1, which you might remember was the value of M in our formula. That's where F_T appears, and it looks like there is about 1V there. If you look back at the dialog box, you'll note that the amplitude was 2VPP, which is 1V peak. Now, here is the really important point. **F_T always appears in the M^{th} bin of a spectrum. Always.** That's why we bother with these formulas, if we sample a signal F_T we can predict what bin the signal is going to appear in our spectrum. This essentially gives us a voltmeter per bin (to borrow from Matt Mahoney).

The Fourier Frequency

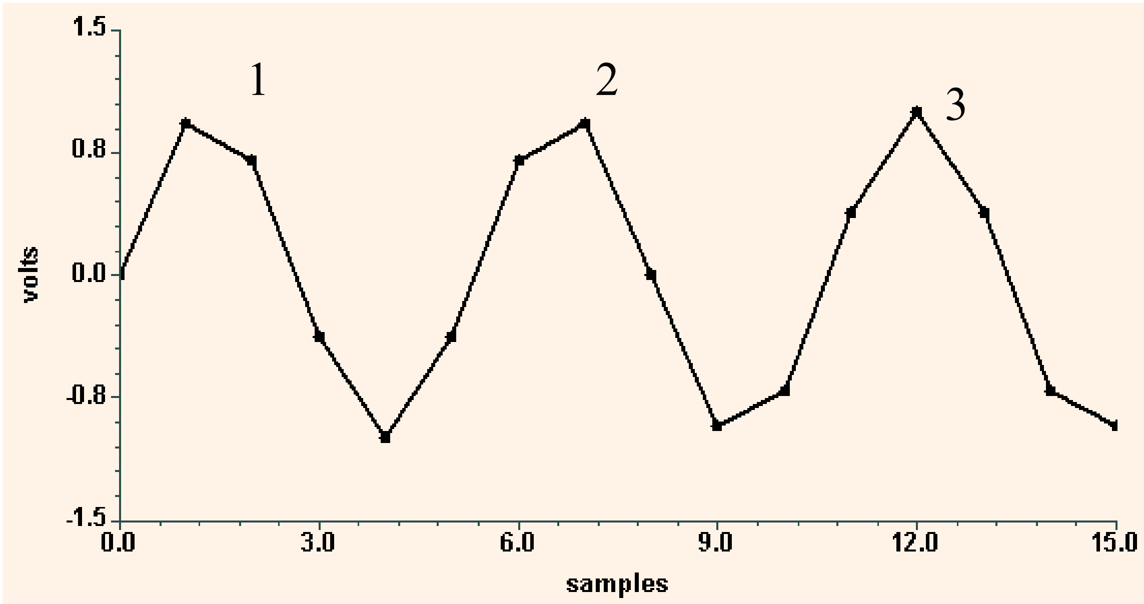
It should make sense that this spectrum has some resolution, some number of Hertz per bin. The formula for that value is $F_F = F_T/M$ (also $F_F = F_S/N$). In this example the Fourier Frequency is 1KHz/1 or 1KHz. That works, because $F_S/2$ holds the value for 8KHz ($F_S/2$) or eight times the Fourier frequency, which is also 8KHz.
Another amazing thing is that the time it takes to capture these samples is the reciprocal of the Fourier frequency, in this case 1mS. Of course that makes sense, the waveform consists on one cycle of 1KHz, that's one millisecond. This time is called the Unit Test Period or UTP. UTP is the reciprocal of F_F, the longer you sample, the finer the frequency resolution you will have. If you setup your numbers (M, N, F_S, F_T) such that the UTP is one second, your spectrum will have one Hertz resolution. Of course it's unlikely you really want to sample for one second, it's rather costly to let a piece of test equipment sit around for a whole second just capturing signals.
By the way, that's it for variables, just six terms, no square roots, no exponents, just six little variables that can easily be solved with a cheap calculator. Pretty easy right?

Changing the parameters

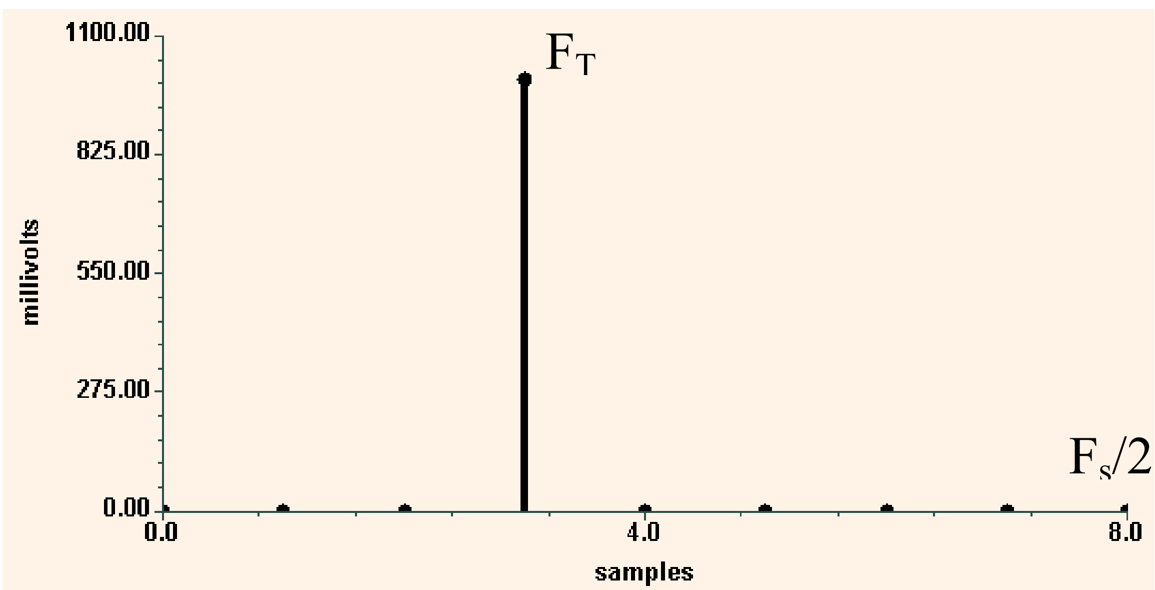
So what happens if we change one of the variables, say, F_S, what happens then?



Here we are back at the Generate Sine Waveform dialog box, but this time I changed the Sample Rate (F_S) from 16KHz to 5.333333333KHz. In other words F_S was reduced by a factor of three. You'll note that I kept F_T and N the same, so the only thing that could change is M. So now when we look at the waveform generated by this dialog, it should have 3 cycles.

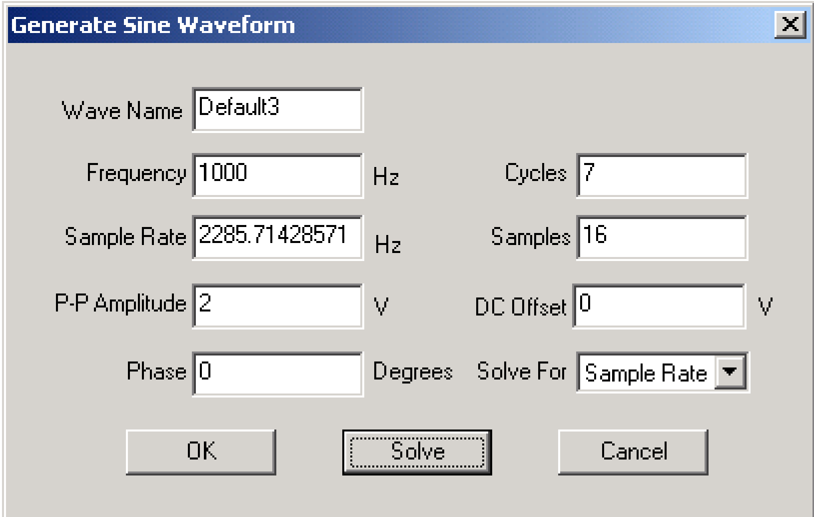


Sure enough, although it looks a little "bumpy", it approximates a sine wave with 3 cycles. The bumpiness has no impact at all on the spectrum, so don't let that bother you, to all intents and purposes this is a perfect single tone sine wave.
Now for the spectrum.

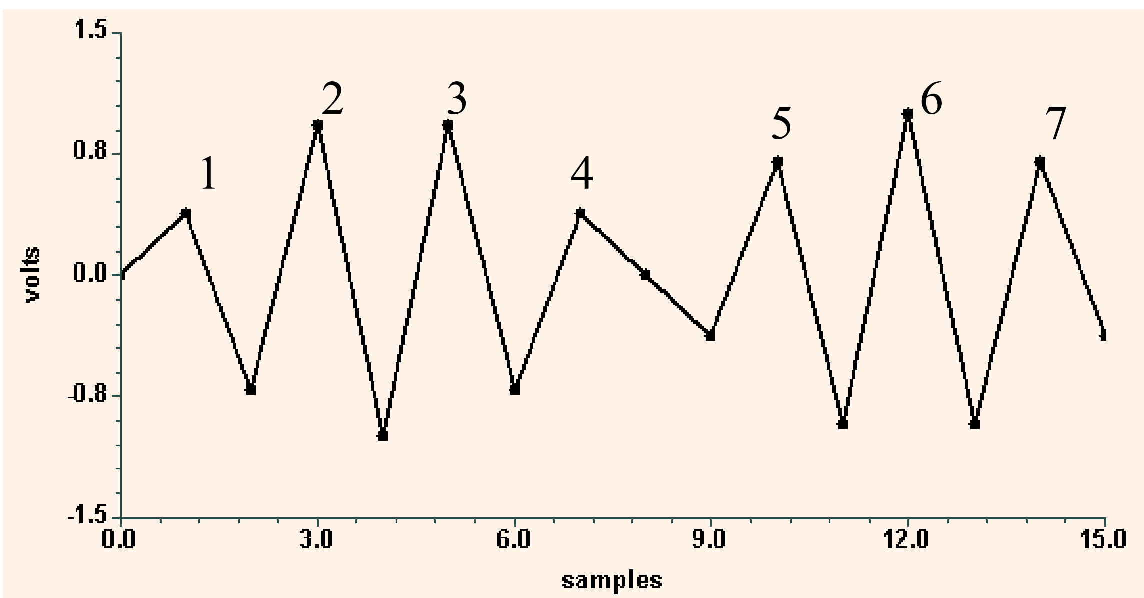


Just like the last spectrum we looked at, this one also has 9 bins, DC through $F_S/2$. But now because the F_S is lower (5.3333KHz), F_T (which is still 1KHz) has moved up towards the last bin. What bin is F_T in? Remember our rule, **F_T <u>always</u> appears in the M^{th} bin of a spectrum. <u>Always</u>.** So bin 3 contains the amplitude of the 1KHz signal we captured. Notice that nothing has happened to the signal, it's still the same signal as before, we just sampled it slower, so it moved up in the spectrum which is bounded by the two limits, DC on the low end and $F_S/2$ on the high end. The Fourier frequency changed too, it's F_T/M or 1KHz/3 or 333.333Hz. Nothing difficult here, just simple math.

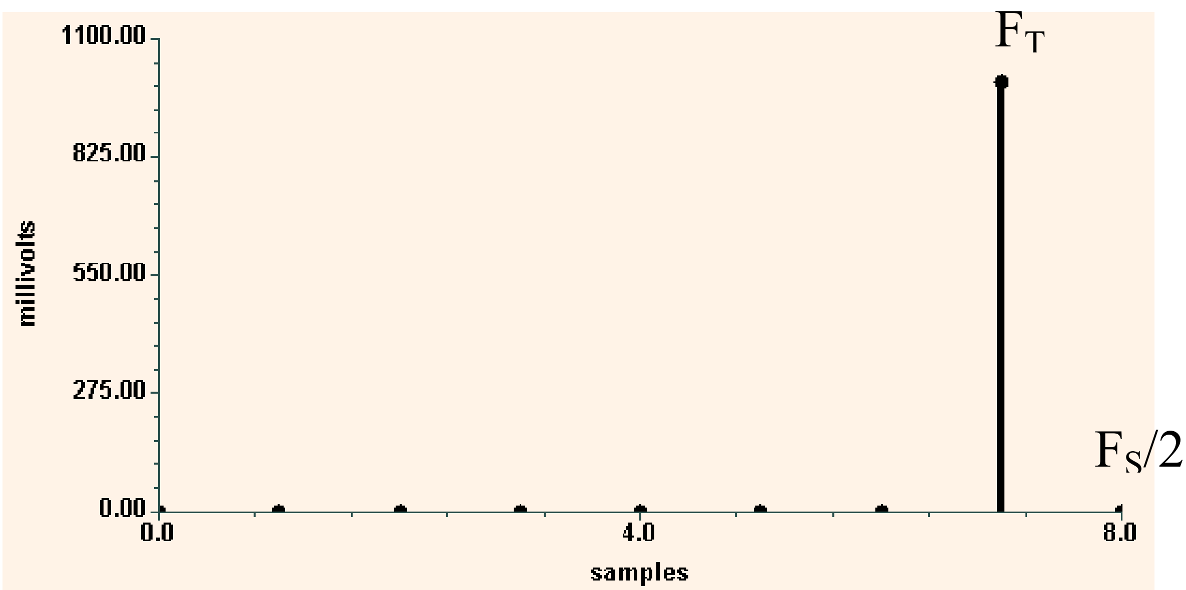
Now, what if we sample even slower, what's going to happen? Well, let's take a look.



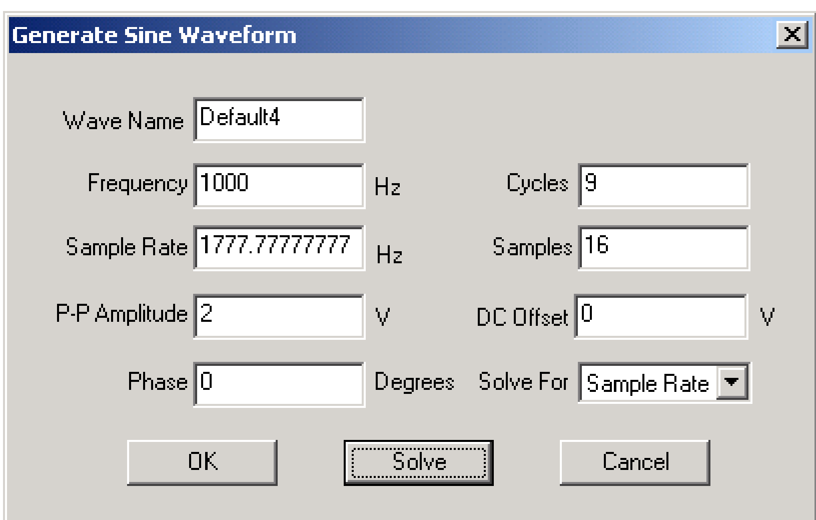
Here we have reduced the Sample Rate (F_S) to 2285.714Hz, again we kept F_T at 1KHz and N remains at 16, so again, the only other variable that can change is M. It's now 7. So now our sine wave should have seven cycles spread over 16 samples. Prepare yourself, it's going to look pretty bumpy.



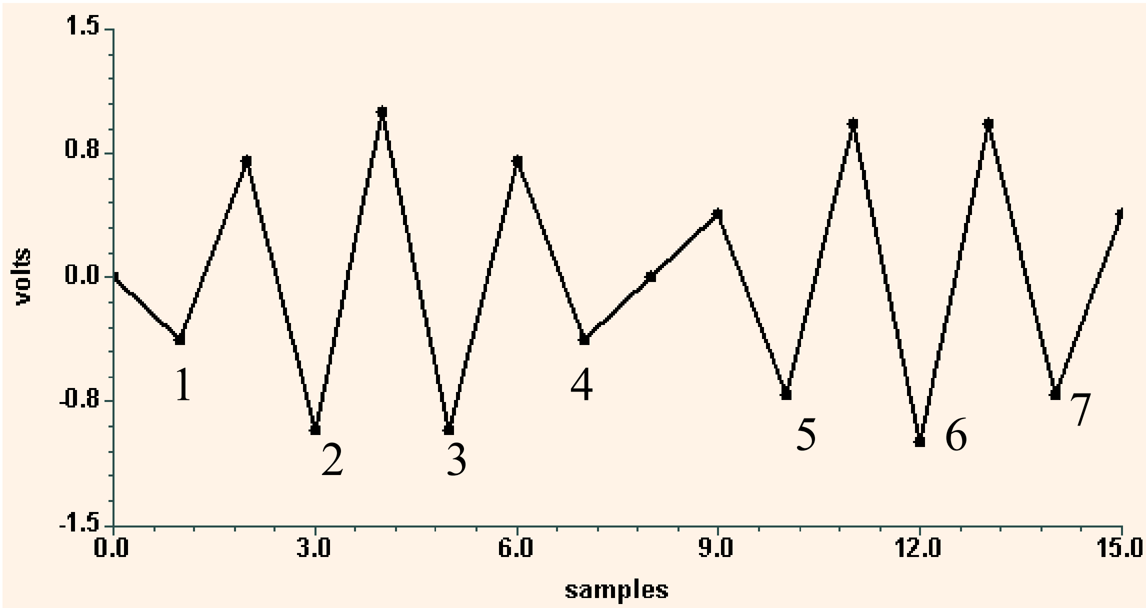
So here is the waveform, it indeed has seven cycles as noted above the waveform. Although it doesn't look much like seven cycles of a perfect sine wave, mathematically it is in fact, perfect. Just because it doesn't look so great in time domain, that's not what's important, frequency domain is what counts. Let's look.



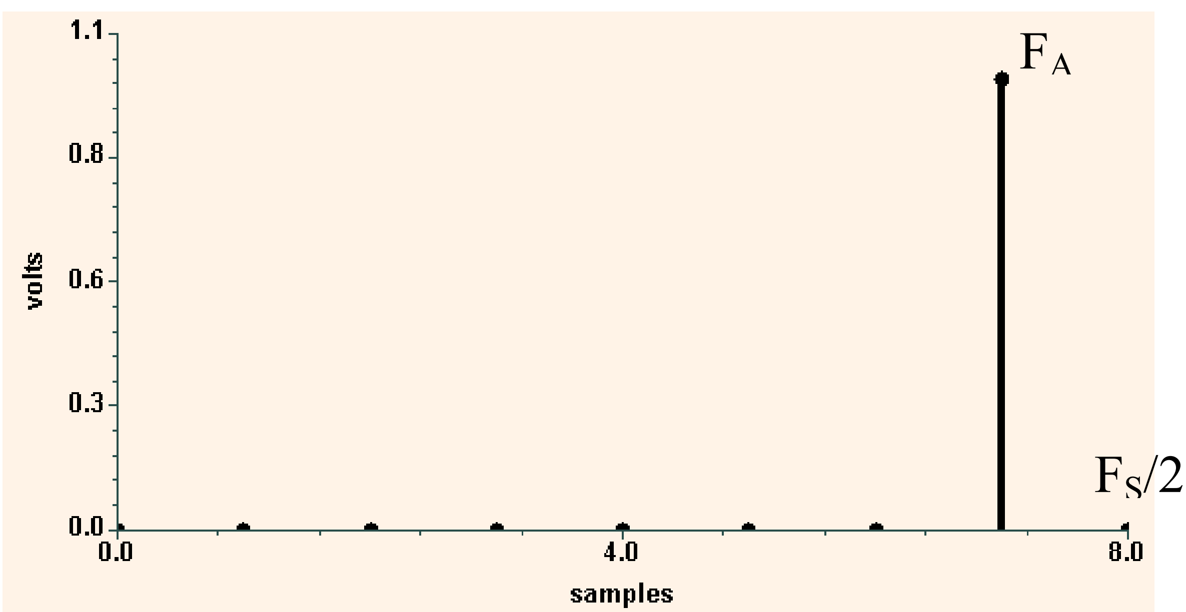
Now that the sample rate is down to 2.28571KHz, a 1KHz signal looms pretty high up in the spectrum, which shouldn't surprise you since $F_S/2$ is now 1.14285KHz.
If we were to reduce F_S any more the F_T has no place left to go, it's already in bin seven (the M^{th} bin). If we were to change the sampling frequency to something less than F_T*2 the frequency bin that contains F_T would go above $F_S/2$ and disappear forever. It would be just like Columbus sailing over the edge of the earth, never to return to Spain, never to have countries, cities, rivers or boat builders named after him.
But wait, Columbus did come back, and he did have a lot of stuff named after him, so does that mean there is hope for our signal? Let's see.



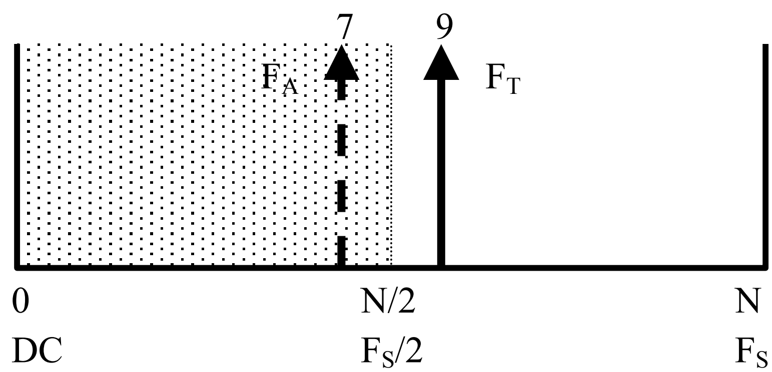
Now I've set the Sample Rate to 1777.777777Hz, which is lower than two times the test frequency F_T. Now M (the number of cycles) is 9, let's see if that really happened.



One of the first things you'll notice is that the waveform starts by going negative, not positive which it did in every waveform we saw previously. That is a clue to what just happened, but remember, we shouldn't trust what we see in time domain.
The other thing you'll notice is that there are not nine cycles, there appear to be only seven, as denoted by the numbers across the bottom (since the waveform started by going negative).
Now before we go on, it's important to remind you of the rule I established, specifically, **F_T <u>always</u> appears in the Mth bin of a spectrum. <u>Always</u>.** That does not mean that it can't appear in other bins as well.
Enough stalling, here is the spectrum.



In this example M was 9, but our spectrum doesn't have a bin 9, it only had bins 0 through 8. But, the time domain waveform gave you a clue, the signal will appear in bin 9, but it will also appear in bin 7.
We can see bin 7, but we can't see bin 9, that makes the signal in bin 7 an alias. It's an alias because while the amplitude in bin 7 is exactly the same as the signal amplitude in the invisible bin 9, bin 7 is not where this signal belongs, and if we didn't know better, we would think that some other frequency got into our capture. That frequency is labeled F_A, the alias frequency. The actual frequency of F_A can be calculated by multiplying the aliased bin by the Fourier frequency. The Fourier frequency in this case is 1000Hz/9 or 111.11111Hz. That makes the alias frequency 7*111.1111Hz or 777.777Hz. This frequency doesn't actually exist, but the spectrum makes it look like this signal is present. That's the problem with aliasing, it fools you into believing that you captured a signal that doesn't exist. But, it's important to remember that in testing you generally have control over the signal you are going to capture and since you know the frequency so you can't be easily fooled. Only when you are capturing a signal whose frequency is unknown can you be fooled by aliasing, and in those cases there is a simple technique you can use to figure out that frequency it is.
Here is what happened when our signal aliased.



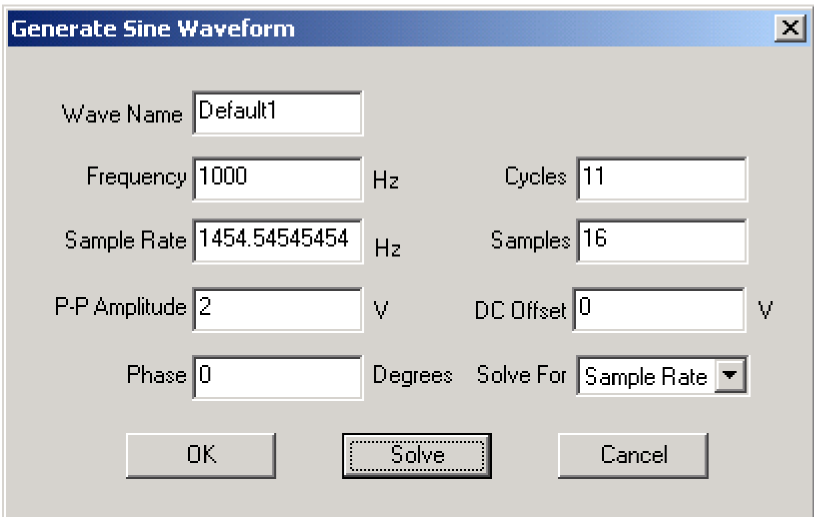
In the above diagram, we show the spectrum as it really exists. The sampled spectrum starts on the left at DC, or zero times the Fourier frequency (F_F) and goes to the right increasing in frequency to F_S. One of the problems we have in conceptualizing aliasing is that we can't see anything above $F_S/2$ (or bin N/2). The only thing you can see is the band that I show in gray, known as the Nyquist band. Everything you sample falls into this band, even if it doesn't belong there (such as aliases). The arrow at bin 9 shows the location of the real signal, F_T. But because the Nyquist frequency ($F_S/2$) acts like a mirror, the reflection of F_T bounces back into the Nyquist band and appears to us to be a signal in bin 7, which in this case is 777.777Hz. The real frequency, 1KHz is hidden from us, but the evidence of it's existence is the alias signal, F_A at 777.777Hz. In a test environment, we would already know that we had sent a 1KHz signal to the device under test (DUT) so we shouldn't be too surprised to see it come back looking like 777.777Hz. But, a person who doesn't understand aliasing would wonder how his DUT transformed a 1KHz signal into a 777.777Hz signal. He might conclude that he had created a "water into wine" chip, when in fact the DUT is putting out 1KHz, it's just aliasing back to a lower frequency.

Finding the alias

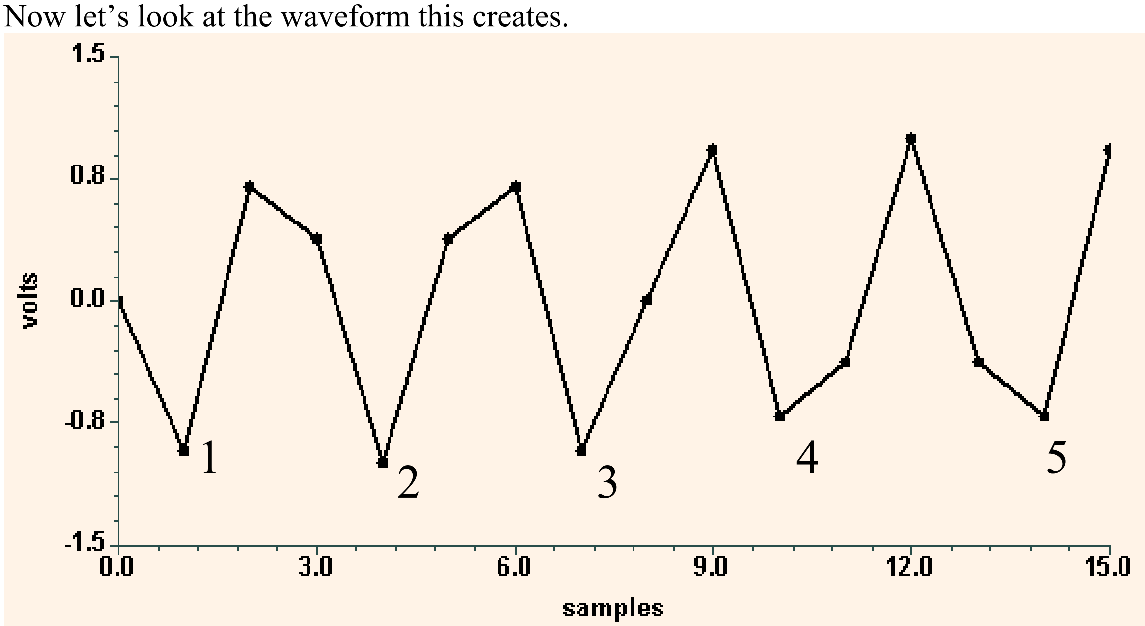
There is a simple way to figure out where something is going to alias. If you know the bin that F_T should appear in (the M^{th} bin, remember?) then you can easily figure out where the alias will fall in the Nyquist band. I call it Dan's Aliasing Rule Number Two.

If Bin > N/2 then Bin = N-Bin

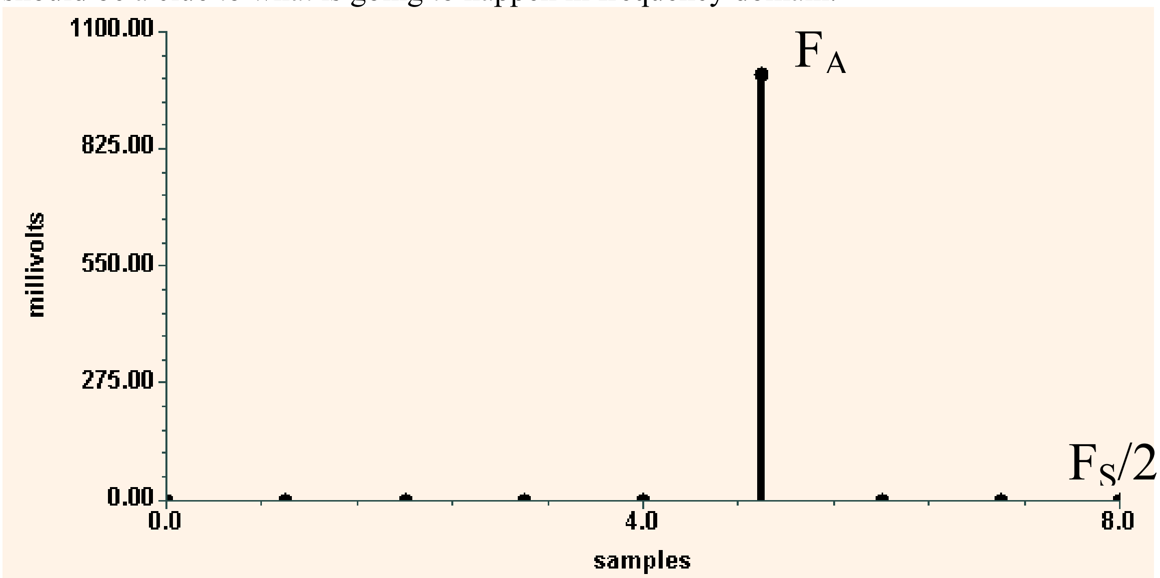
If the bin value, in our case, 9 (since M was 9) is greater than N/2, or 8 in our case, then the aliased bin will be 16-9 or 7.
Let's give it a try on another example.



Again I fire up the Generate Sine Waveform dialog and set the Sample Rate (F_S) to 1454.545454Hz, which, keeping N and F_T constant results in an M of 11.
Now let's look at the waveform this creates.



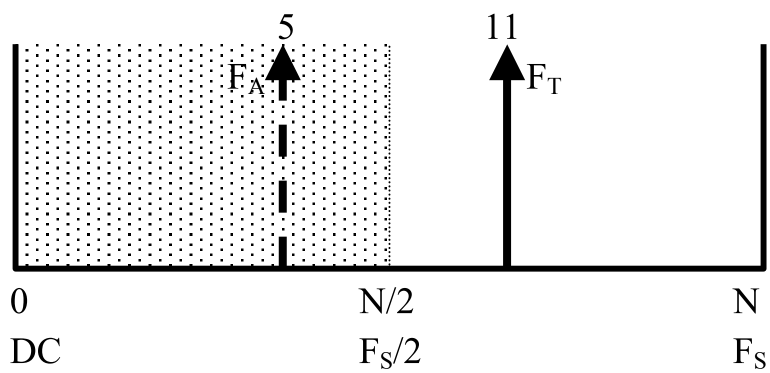
Note right away that we don't see 11 cycles, and the waveform starts by going negative despite a zero degree phase setting in our dialog box. We actually only get 5 cycles. That should be a clue to what is going to happen in frequency domain.



In frequency domain we see a 1V peak amplitude in bin 5, and remember, we saw 5 cycles in the captured time domain waveform, despite the fact that we had M set to 11. Let's try Dan's Aliasing Rule Number Two (does it bother you that we haven't seen rule one yet?).

If Bin > N/2 then Bin = N-Bin

If the bin value, in our case, 11 (since M was 11) is greater than N/2, or 8, then the aliased bin will be 16-11 or 5.
Here is what happened.



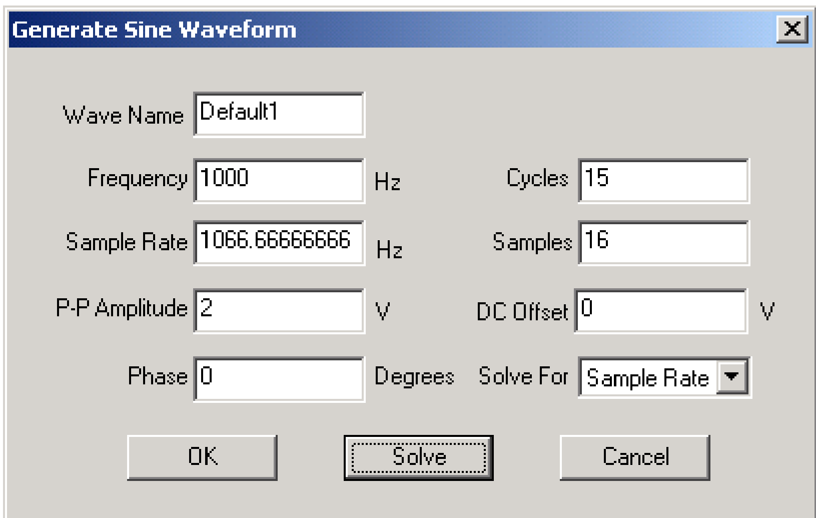
Now that F_T (which is still 1KHz) went way above $F_S/2$ (which is now 727.27Hz) it's gone outside the Nyquist band, but it's image bounced off $F_S/2$ and came back as an alias. Notice in this diagram how we show $F_S/2$ equating to N/2, and F_S equal to bin N. This is true, remember that our magic sampling formula equates M to F_T and N to F_S?

$$\frac{F_T}{F_S} = \frac{M}{N}$$

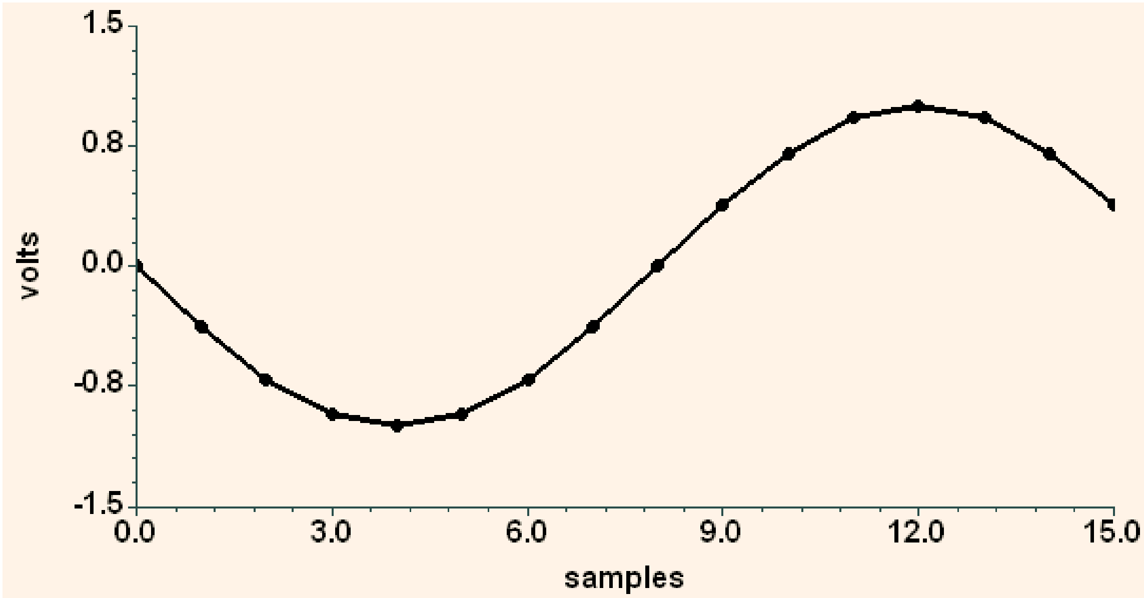
This means that bins equate to frequencies, and frequencies equate to bins. The bin number is not limited to N/2, nor is it limited to N, so theoretically you can sample frequencies not only over $F_S/2$, but also beyond F_S itself. But let's not get ahead of ourselves. Let's look at one more example before we plunge off the deep end.

Undersampling

The term undersampling is often used to describe aliasing, but it only seems appropriate in a few cases. Those cases are where you are sampling just over, or just under F_T. What we have done so far can also be described as undersampling, although some would argue the point with you. Let's look at an example of true undersampling.

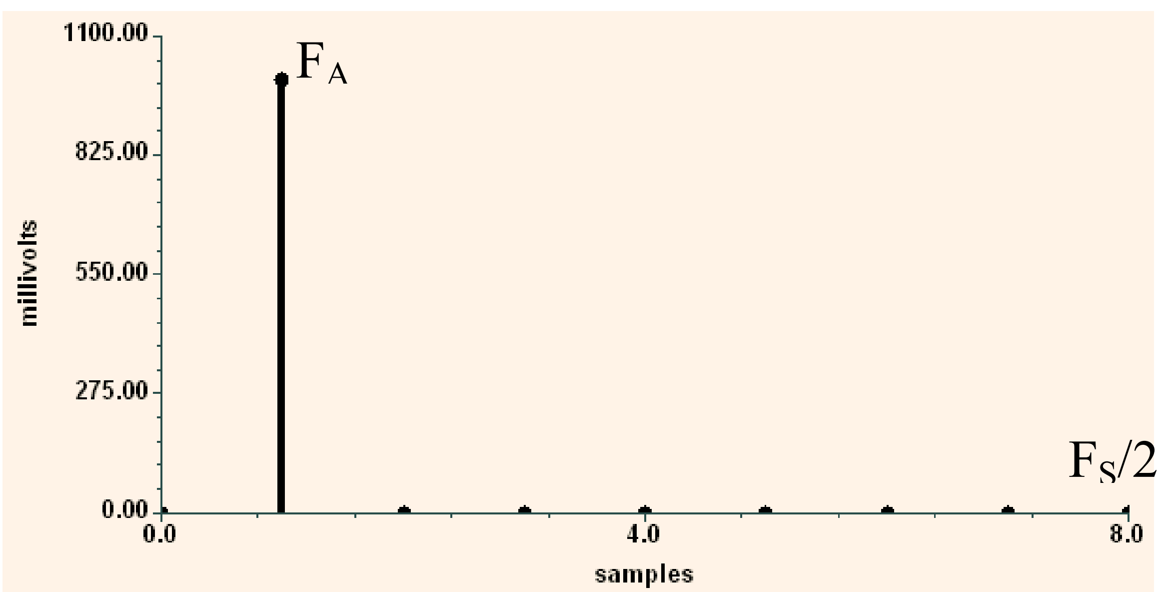


Again we use the Generate Sine Waveform dialog to create the waveform, this time F_S (Sample Rate) is 1066.66666Hz, just above F_T of 1000 Hz. We are going to be capturing 15 cycles of the waveform into 16 samples. Are you ready for the time domain representation of the waveform? I bet you're not.



Did you really expect to see 15 cycles when you only had 16 samples? It can't happen, you would need at least two samples (a positive peak and a negative peak) to describe a single cycle of any kind of waveform, so you would have needed at least 30 samples to describe 15 cycles, right? Well, that's why they say you must have at least two samples per cycle, but that only applies to creating the waveform (or showing it on a display). That doesn't apply when you capture, if you can handle the aliasing.

Notice that just like the other aliased waveforms, this one starts by going negative rather than positive like the unaliased waveforms. The reason for this is that M is greater than N/2, but less than N. Any signal in this region appears backwards when observed in time domain. Now for the spectrum.

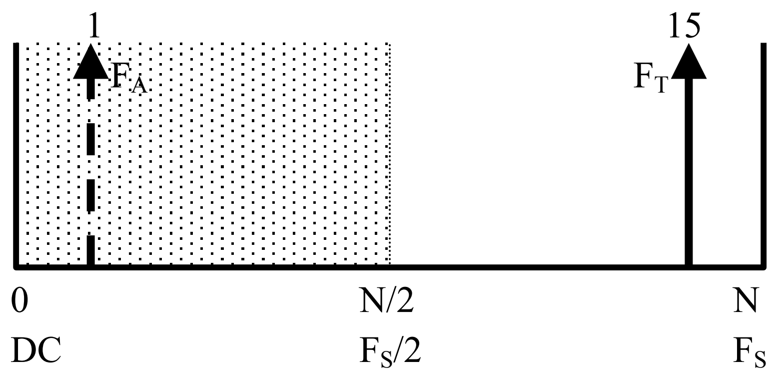


Since the time domain represents one (backward) cycle, the alias ends up in bin 1.

Note that we are only looking at magnitude here, the FFT polar data also includes a phase element which would indicate the reverse nature of our waveform, but the magnitude portion of the spectrum doesn't know about the phase and doesn't need to.
Let try applying Dan's aliasing rule number two to this problem.

If Bin > N/2 then Bin = N-Bin

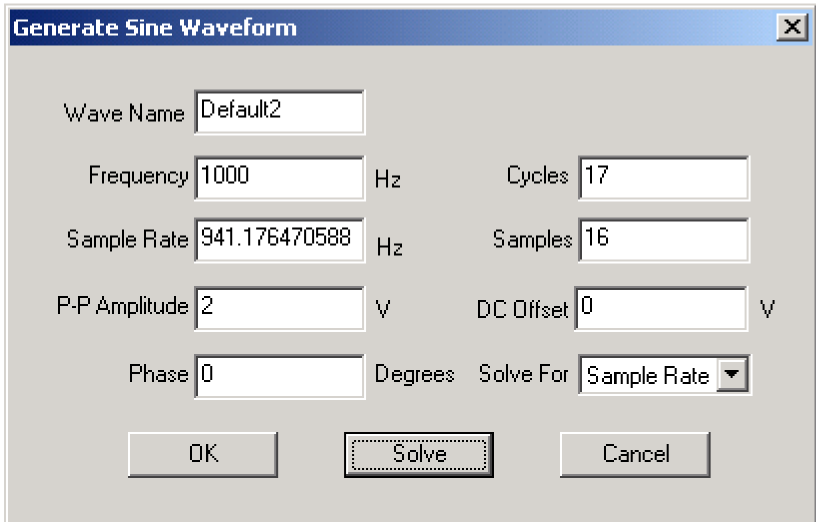
If the bin value, in our case, 15 (since M was 15) is greater than N/2, or 8, then the aliased bin will be 16-15 or 1.
Here is what happened.



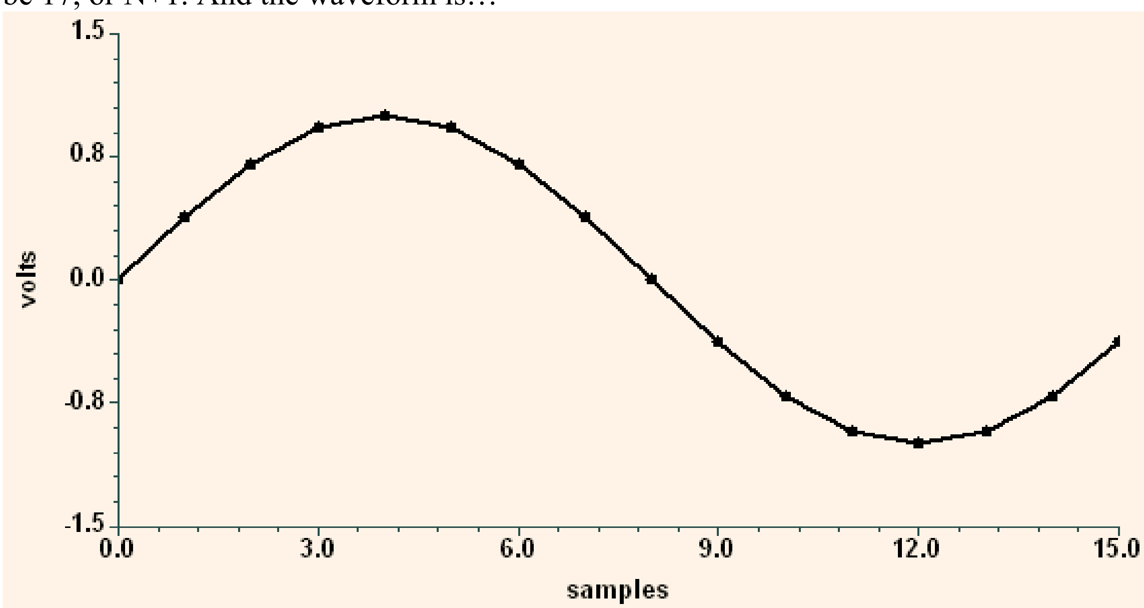
Since F_T falls into bin 15, or just one bin **below** F_S, F_A falls into bin 1, or one bin **above** DC. That's the way aliases work, when F_T falls in the band between $F_S/2$ and F_S, it's just like $F_S/2$ was a mirror, with F_A being the reflection of F_T. If F_T gets closer to the mirror, so does F_A (the reflection of F_T). If F_T gets farther from the mirror, so does F_A. Stand in front of a mirror and try it sometime.
The most useful thing about aliasing is that while the phase of the signal might be wrong, the amplitude of F_A is exactly the same as the amplitude of F_T. You don't need to be able to see F_T (good thing because you can't if it's aliasing) to know what it's amplitude is, just look at the amplitude of F_A, it's exactly the same. Yeah, sure, the phase is 180 out, big deal, nobody cares, and if they did, guess what? Just invert the phase!
Just one more thing, some people call this type of undersampling equivalent time sampling, because you get only one cycle of the waveform (backwards or not) it looks like you are sampling with an M of one. Remember that when M is one, you get one cycle. Well, set M to N-1 and you get the same thing. Now, set M to N+1 and guess what happens.

Going above F_S

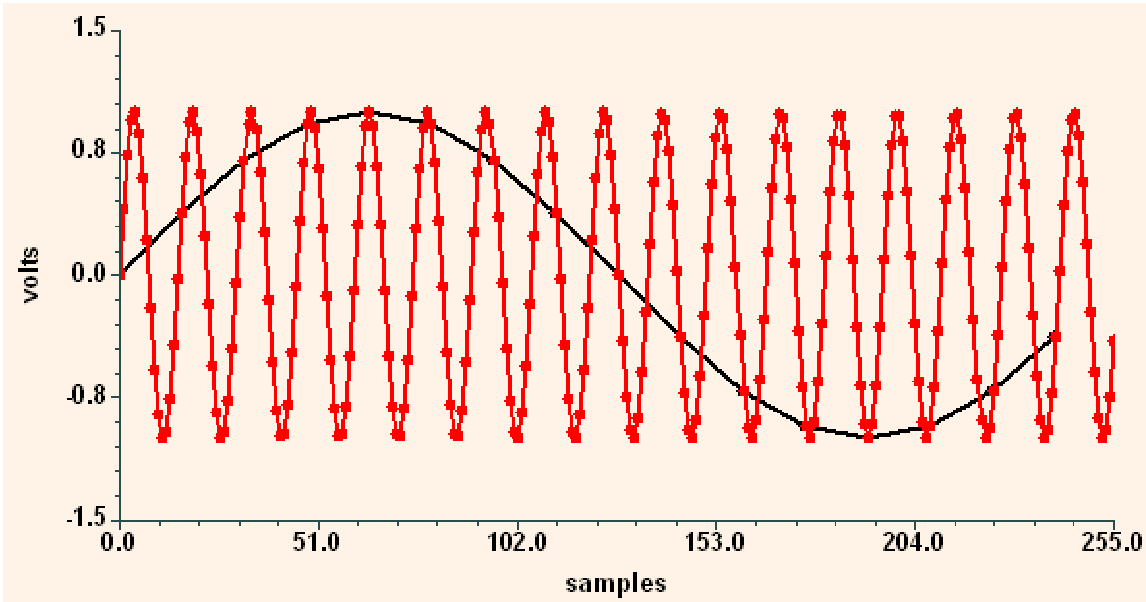
Let's go back and make a new waveform, this time with 17 cycles in 16 samples.



Now I've set the F_S to 941.176Hz, lower than F_T itself (still 1000Hz). This causes M to be 17, or N+1. And the waveform is…

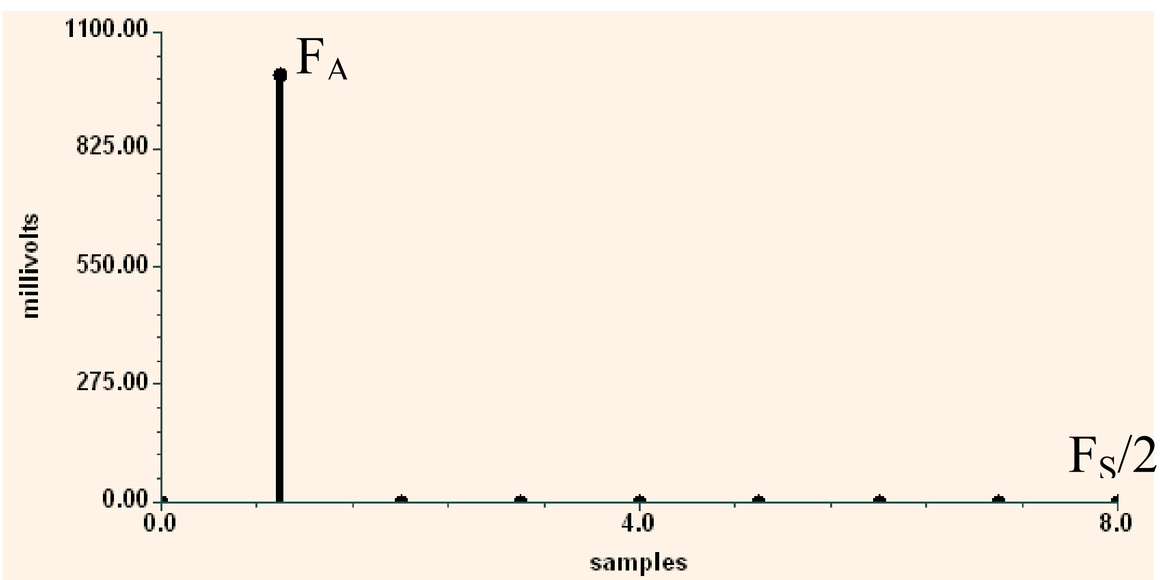


That's right, now F_T is aliasing in the forward direction, so it looks like one cycle of the waveform, but since it's above F_S, it now appears normal, with no phase shift at all. Now, I don't want you to get used to this, but let me show you what is going on in time domain.



In this picture there are seventeen cycles in the red (real) waveform, and one cycle in the aliased waveform. Because we are sampling so slowly, we only capture one sample from each cycle, and the sample point "drifts" across the waveform to slowly capture a single cycle of the waveform. This is the crux of equivalent time sampling. While it's pretty easy to understand aliasing in the way, it's much more useful to think of aliasing in frequency domain, because measuring signal amplitude is more accurate in frequency domain. This is because the FFT puts each frequency in it's place so you can ignore frequencies you don't care about and measure the amplitude of just the ones you do care about.

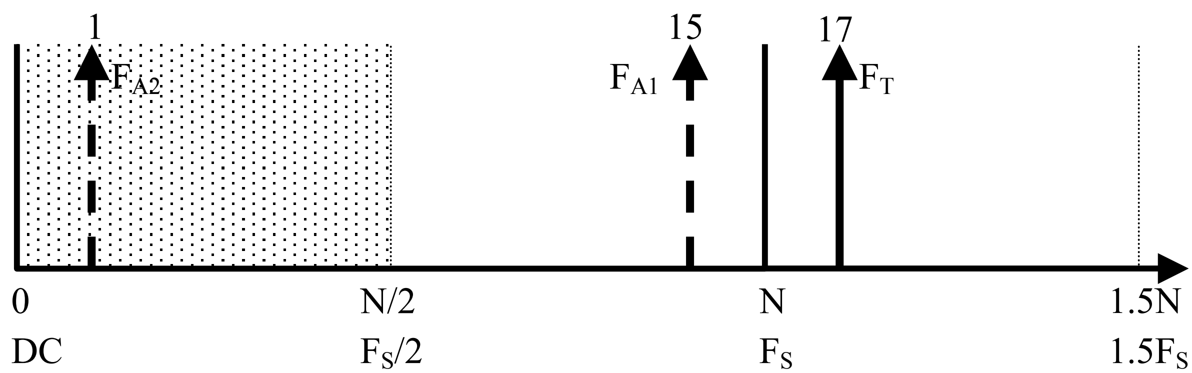
Speaking of FFTs, it's time to look at the spectrum of the undersampled waveform.



You shouldn't be too surprised that F_A ended up in bin one. Since F_T is in the otherwise invisible bin 17, that puts it one bin above bin N. This gives us an opportunity to introduce Dan's Aliasing Rule Number One (finally).

If Bin > N then Bin = Bin mod N

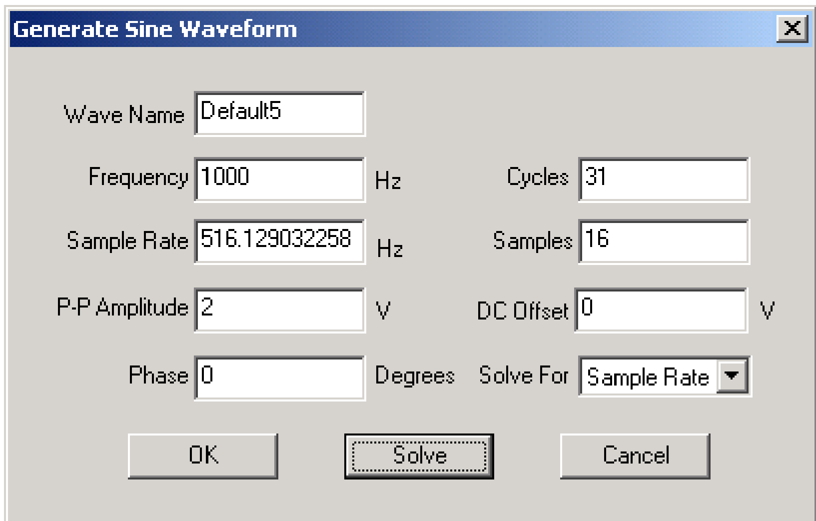
If the bin value, in our case, 17 is greater than N, or 16, then the aliased bin will be bin **modulo** N. The modulo operator is just looking at the remainder of a division. In this case, 17 divided by 16 is 1 with a remainder of 1. While Excel and C both have modulo operators, your calculator doesn't, so it can be a little odd to do yourself. There are two ways to do modulo on your calculator. One way is to subtract N from the bin number (repetitively if needed) until the bin number is less than N. The other way is to divide the bin number by N, then subtract out the whole portion (digits above the decimal point) then multiply the leftover value by N. For example, 17/16=1.0625. 1.0625-1.0=0.0625. 0.0625*16=1.
Now for the explanation of what happened.



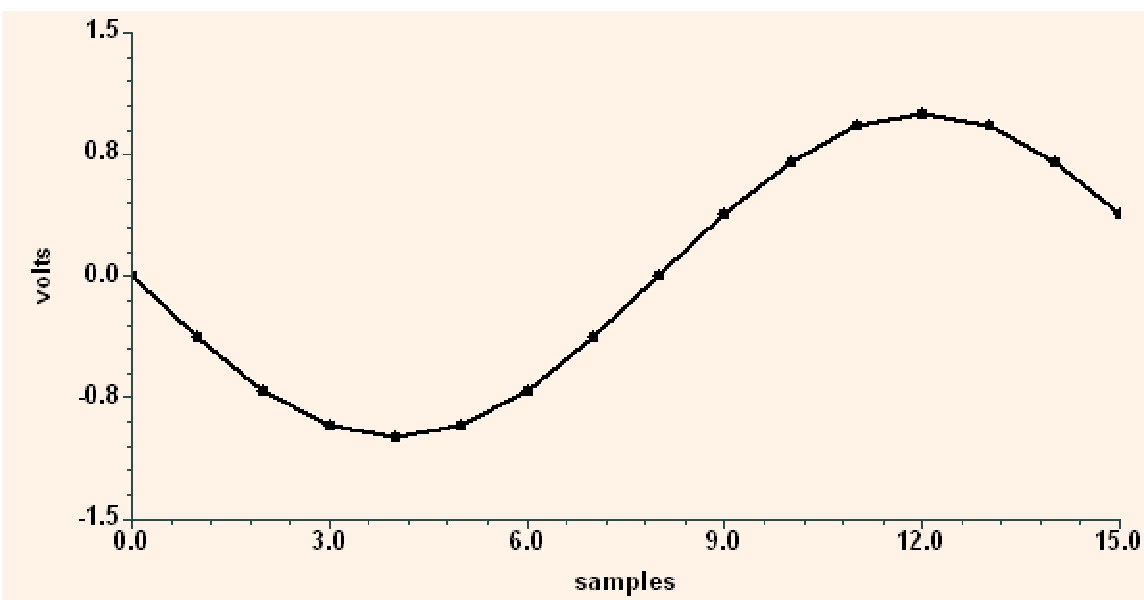
Now that F_T has gone above F_S, it creates an image at F_{A1}, and at F_{A2}. You can't see the image at F_{A1}, but you can see the image at F_{A2}, and it's an alias of F_T. Because F_T happened in the non-reversed band from F_S to $1.5F_S$, the signal appears in forward time, that is, it's phase is undisturbed. However, if F_T is forced into the band of $1.5F_S$ to $2F_S$, the phase will be reversed again. The arrow pointing to the right implies that this set of frequency bands continues. How far does it go? To infinity. Theoretically, you could capture light with a sampling frequency of 1Hz, but the Nyquist band would be pretty crowded, because any energy from DC to light would alias back into the Nyquist band. Luckily a low pass filter on the input of the ADC will prevent you from capturing all that garbage into your sampler, otherwise your Nyquist band would be full of all kinds of junk. By the way, a low pass filter on the input to an ADC is called an **anti-aliasing filter**. It doesn't prevent aliasing, nothing can stop that, but what it can do is prevent high frequencies from sneaking into your ADC, which prevents pollution of your Nyquist band with aliases of frequencies you don't care about.

Rules Rule

After applying Dan's Aliasing Rule Number One, don't forget to apply Dan's Aliasing Rule Number Two. While the result in the last example didn't need Aliasing Rule Number Two (because 1 is not greater than N/2), you might end up in a situation like this:

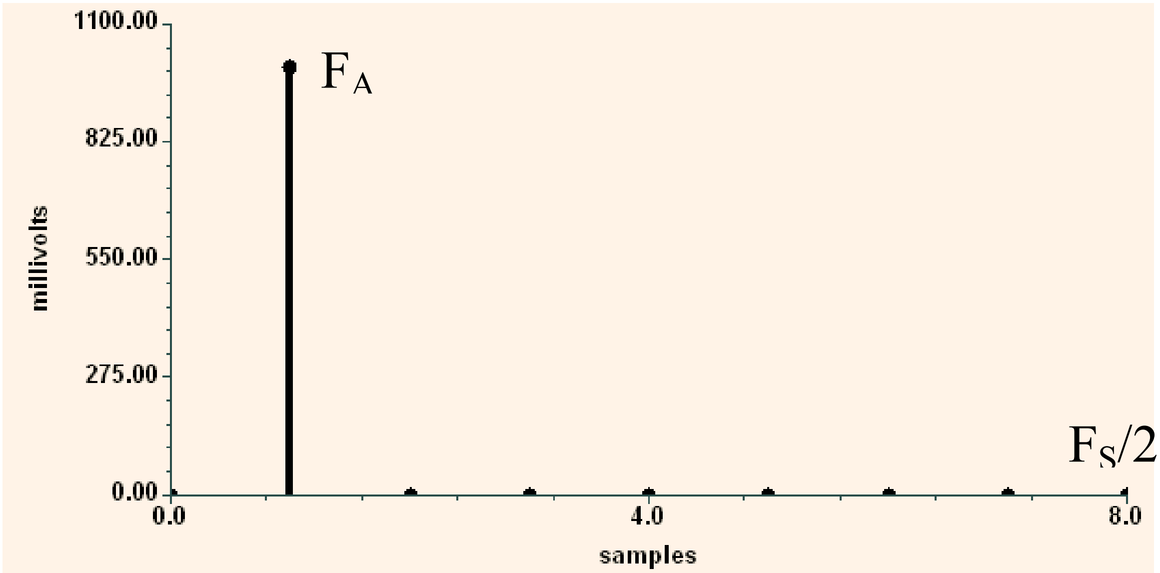


Here the sample frequency is 516Hz, and now M is 31. Looking at the time domain waveform:

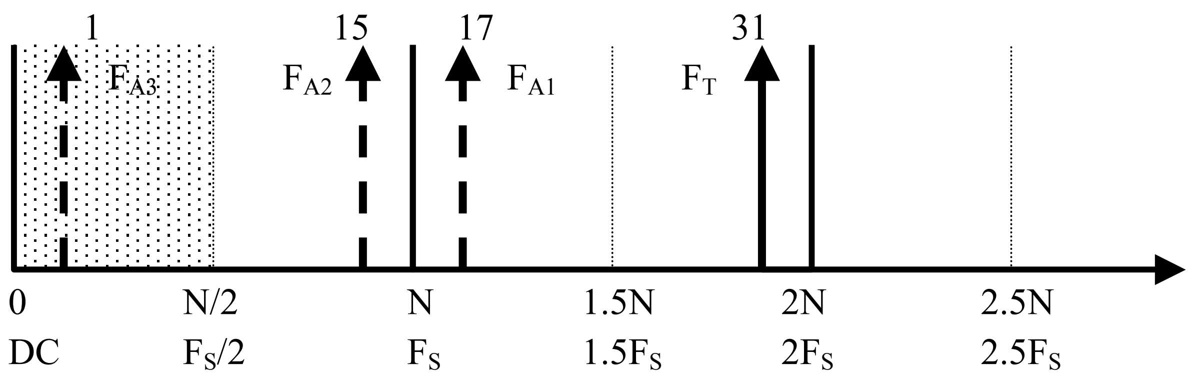


Remember we talked about equivalent time sampling, well here is another case. This happened because M is N*2-1. In fact, if M is N times any number, minus 1, the result looks like this. Examples in this case are 31, 47, 63, 79, 95, 111, 127, etc. If M is equal to N times any number, plus 1, the captured waveform will look like one cycle, but it will be forward. Examples of this case are 33, 49, 65, 81, 97, 113, 129, etc.

Now let's look at this in frequency domain.



It's in bin one again. Remember that the phase is going to be reversed (because M is N*X-**1**), but again, that doesn't matter.
Here is what happened.



In this diagram we see F_T falling in bin M (31) but you can't see bin 31, you only get bins 0 through N/2. The image of F_T shows up in bin 17 as F_{A1}, then aliases into bin 15 as F_{A2}, and finally falls back into bin 1 as F_{A3}. Let's try applying the rules first, let's apply Rule Number One.

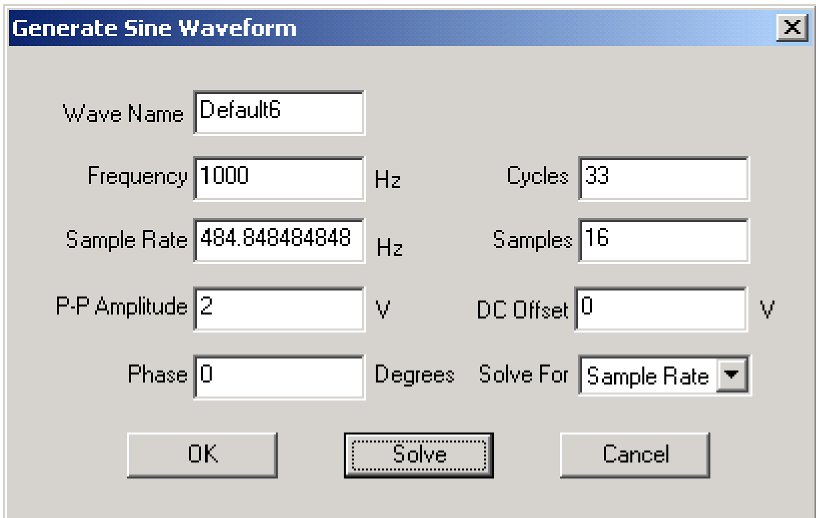
If Bin > N then Bin = Bin mod N

If the bin value, in our case, 31 is greater than N, or 16, then the aliased bin will be 31 **modulo** 16. Take 31 and subtract 16. The answer, 15 is less than 16 so you can stop. If the answer was greater than 16 you would subtract 16 again until the result was less than 16. Are we done? No, because the answer so far, 15 is still not in the Nyquist band. We now have to apply Rule Number Two:

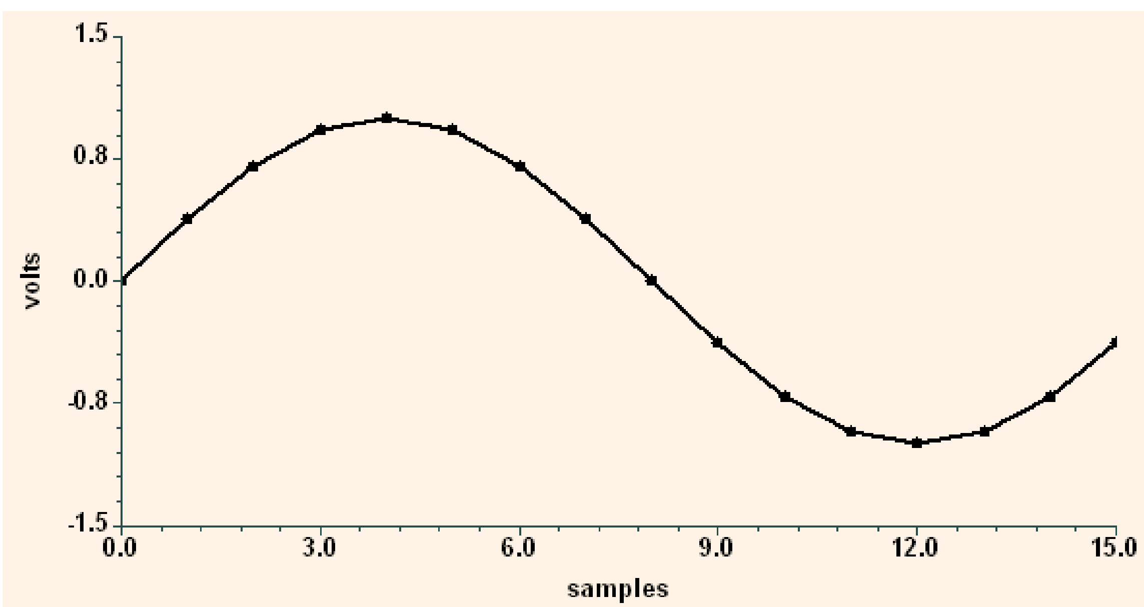
If Bin > N/2 then Bin = N-Bin

If the bin value, in our case, 15 is greater than N/2, or 8, then the aliased bin will be 16-15 or 1.

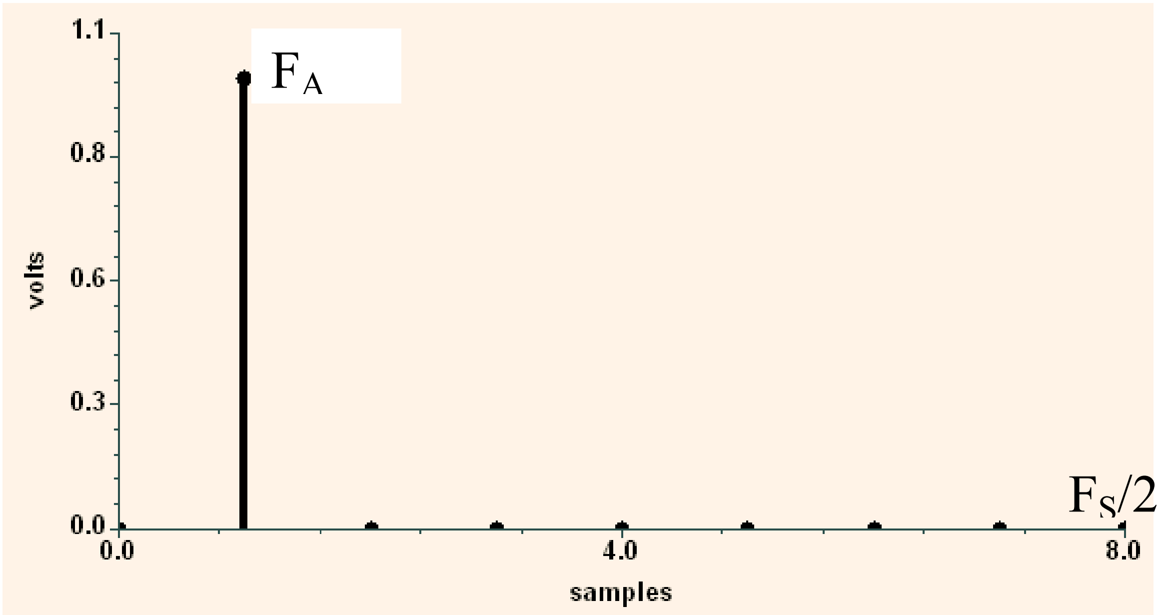
Let's try one more, just to make sure you've got it.



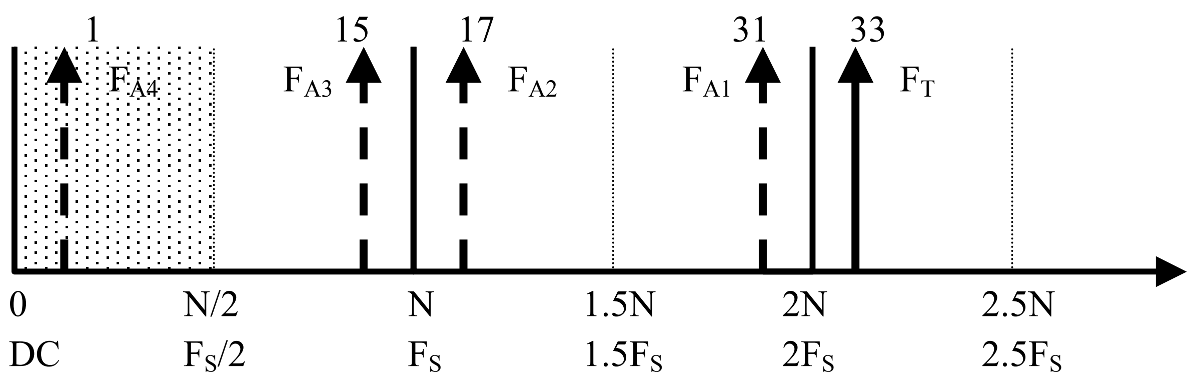
Here the sample rate is less than half the test frequency. You obviously are not going to see 33 cycles of a waveform with only 16 samples, but that doesn't mean it isn't happening.



What a surprise, a single cycle. If you were surprised you should note that an M of 33 is just 2*N+1. This waveform is equivalent time sampled.
Now for frequency domain.



Here you can clearly see that the signal aliased back to bin 1. Again, it should not be a surprise.
And here is what happened.



Now F_T is above two times F_S (or M is over two times N, same difference) so the waveform aliases back to the Nyquist bin, but is not reversed in time.
How far can this go on? Forever. Sampling is very similar to heterodyning, and in communications now, 2.4GHz carriers are being heterodyned down to a baseband frequency of, say 40MHz, then re-heterodyned (sampled) into an ADC and decoded. There is no limit, other than what your electronic front end can handle.

Putting it all to work

Now that you know how to deal with aliases, let's put this new knowledge to work. Say that you have to test a DAC with a 10MHz sine wave. This means that you will be driving digital data at some rate, say 80MHz into the DAC, digitizing the DAC output and looking at SNR and THD. Imagine that the THD spec requires that you look at the first five harmonics, that is, 20MHz through 60MHz. Now imagine that you are using the Lightning ACI to do the capture which has a maximum sample rate of 65MHz. That

would put your Nyquist frequency at 32.5MHz meaning that the fourth, fifth and sixth harmonics are going to alias. Can you do the test?
Of course you can, and you probably won't even sample as high as 65MHz. Let's do the numbers, remember our magic formula?

$$\frac{F_T}{F_S} = \frac{M}{N}$$

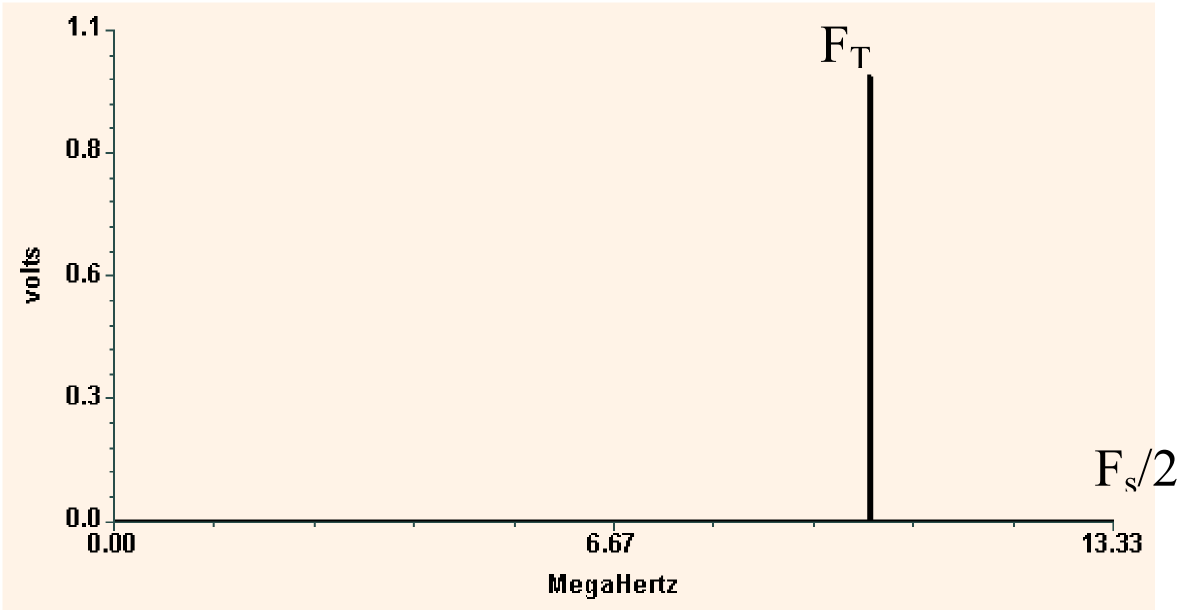
So applying the numbers to it and picking a reasonably large value of N on the source side we get…

$$\frac{10.078125\text{MHz}}{80.0\text{MHz}} = \frac{129}{1024}$$

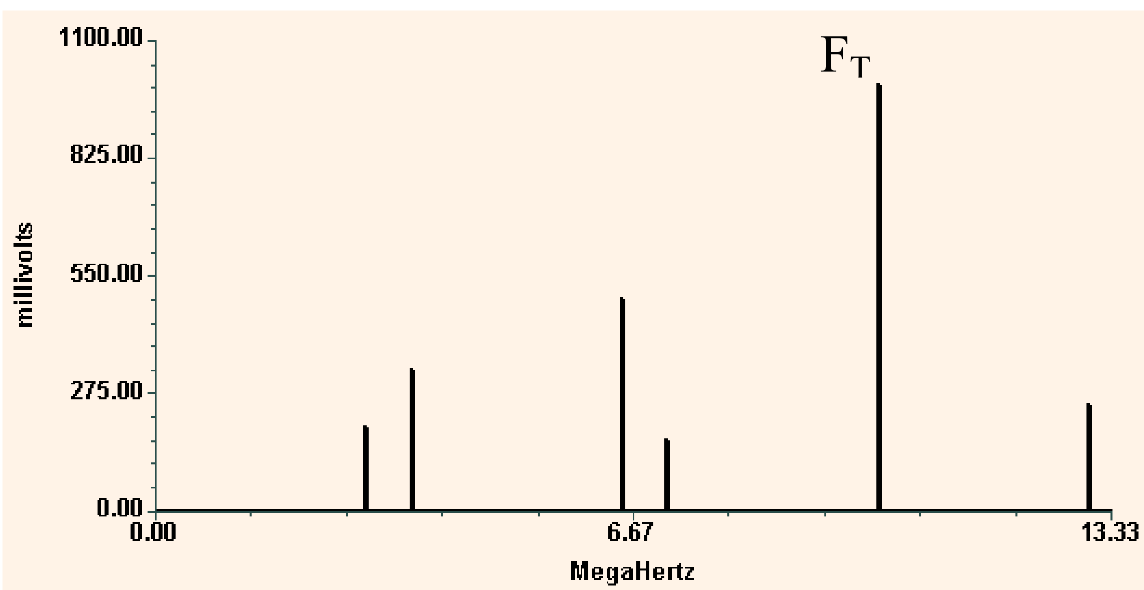
OK, remember that while we can source data and clocks to a DAC up to 133MHz, we can only capture analog signals at rates up to 65MHz, hence the need to use aliasing. The simplest thing to do is sample at a frequency that is some fraction of the source sample rate, say 26.666666MHz (80MHz/3). So the numbers on the capture side will look like this:

$$\frac{10.078125\text{MHz}}{26.666\text{MHz}} = \frac{387}{1024}$$

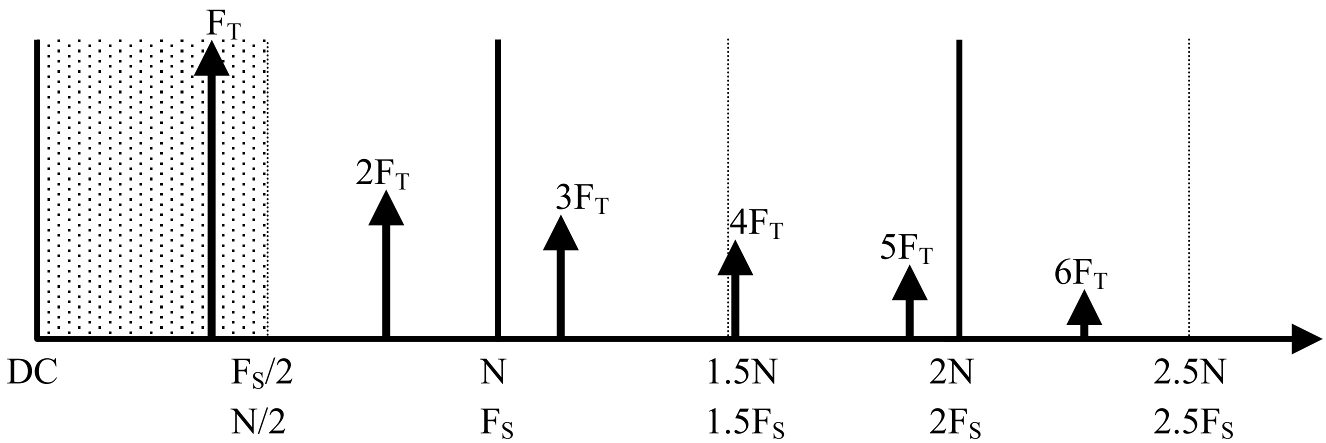
OK, that was easy, M for the capture is three times the M of the drive waveform, F_T and N stayed the same. By the way, you might wonder why I picked 129 for M in the source equation, it's because we want to keep M and N from having any common factors if we can avoid it. Some bad things can happen when they have common factors, for example, a harmonic could fall into either the DC bin or the Nyquist bin and then it really is lost forever. So, as a general rule, it's better to make sure M and N don't have any common divisors.
So, the fundamental (F_T, 10.078125MHz) will fall in bin 387 in a 513 bin spectrum. What's that going to look like?



You can see here that $F_S/2$ is going to be 13.333MHz, and if you remember that the spectrum is half the size of N then it should make sense that F_T in bin 387 is going to loom pretty high up in the spectrum, leaving no room for harmonics, at least unaliased harmonics. You should remember that what you see here is an artificial signal, so it doesn't have any harmonics, but it's not hard to create them and simulate a bad device. Just like this:



Here is the spectrum of another artificial signal, but this one contains additional frequency components, specifically the fundamental plus the second through the sixth harmonics. To make them easy to identify I made the amplitudes inversely proportional to the harmonic number, so the second biggest signal in this spectrum is the second harmonic, the third biggest is the third harmonic, and the smallest signal here is the sixth harmonic. You might notice however that the harmonic bins don't look like harmonic bins because they seem to be randomly spaced in the spectrum. Why is that? Because they are aliasing. Think about it, the fundamental is 10.078125MHz, the Nyquist frequency is 13.333MHz, where else do the harmonics have to go? They obviously have gone above the Nyquist frequency ($F_S/2$) and so the seemingly randomly scattered bins are the aliases of the harmonics that fell well above the Nyquist frequency. Let's look at another schematic of the spectrum to see what is happening.



Just looking quickly at the above diagram reveals how the aliases of the harmonics are going to be scattered throughout the Nyquist band. Especially interesting is the fourth harmonic, which appears just above $1.5F_S$, it will alias down to be just under $F_S/2$, and if you look back at the actual spectrum you can see a signal just below the Nyquist frequency, that is the alias of the fourth harmonic. Now let's do the numbers.
The fundamental will show up in bin 387, and that won't be an alias because it's less than 512 (N/2). The second harmonic will appear in the invisible bin, 774 (2*387). Remembering our rules, since 774 is not more than N (1024) then the first aliasing rule is not invoked, but 774 is greater than N/2 (512) so the second aliasing rule applies.

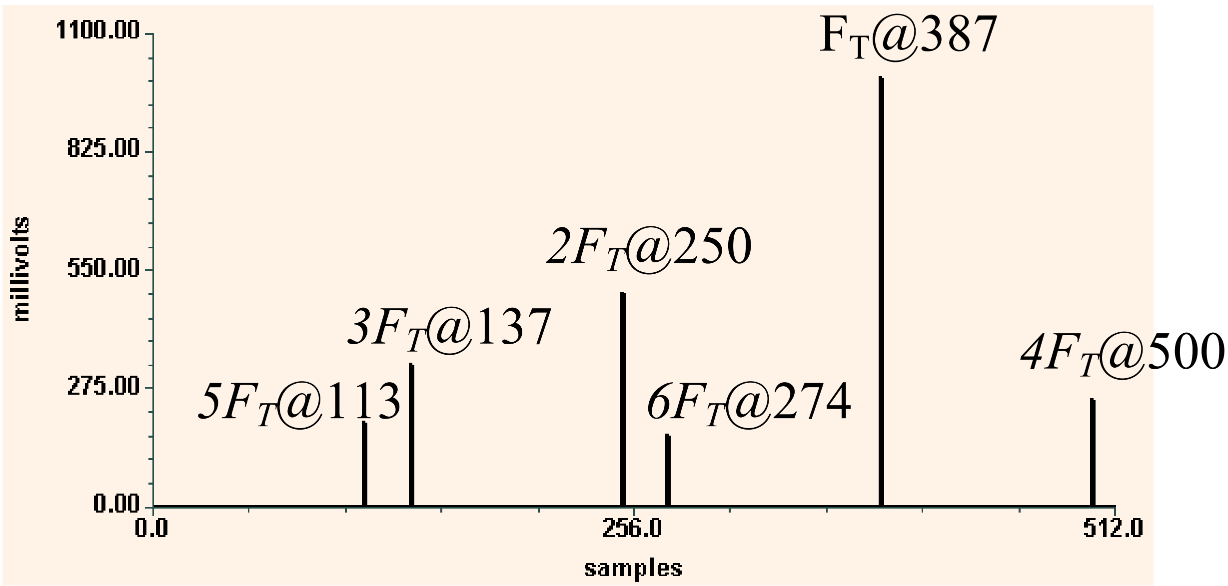
If Bin > N/2 then Bin = N-Bin

So the second harmonic aliases back to bin 1024-774 or bin 250.
The third harmonic has gone above F_S, so rule one will apply but not rule two. Specifically, the third falls in bin 1161, which is also invisible. If we remember aliasing rule one:

If Bin > N then Bin = Bin mod N

Since 1161 is greater than N, we perform the modulo operation making the result 137. Since this is less than N/2, rule two doesn't apply as I pointed out earlier.
Next is the fourth harmonic which falls very close to $1.5F_S$, and since it's just above, both aliasing rules will apply. The fourth harmonic falls in bin 1548, that's greater than N, so we perform the mod operation and get a result of 524, which is greater than 512, rule two applies, making the result 1024-524 or 500.

The fifth harmonic also falls between $1.5F_S$ and $2F_S$, so both rules will apply. Five times 387 is 1935, perform modulo 1024 and you get 911, which is greater than 512, so 1024-911 is 113.
Finally, the sixth harmonic falls above $2F_S$, but lower than $2.5F_S$, so only rule one will apply. Six times 387 is 2322, take the mod and it's now 274.
Looking at the real spectrum again with the bin numbers should make it clear what we captured. Note that the alias names are in *italics* so you don't forget the fact that these are not actually the harmonics, they are the aliases of the harmonics.



So while the spacing between the bins looks random it is not. Note that the distance between the alias of the fifth harmonic and the alias of the fourth harmonic is exactly the same as the distance between DC and the fundamental. The reason that's true is because there were no boundaries crossed between those two frequencies, they both ended up in the band from $1.5F_S$ to $2F_S$. It's harder to measure the gaps between the others, because they all crossed a boundary, either a $0.5F_S$ boundary or an F_S boundary, it doesn't matter, the boundary acts like a mirror and makes it more difficult to measure the distance between the two. Difficult but not impossible. If you know the mirror is there, you can account for it. That's what the aliasing rules are for, to help you deal with the boundaries.

Actual testing

Does this mean that if you plan on using aliasing you are going to have to deal with the rules yourself? Probably not, any good mixed signal tester should account for aliasing in it's DSP library routines that measure THD, SNR, SNDR, etc. The rules should be coded into the routine so that if the harmonics alias, they will be found in the Nyquist band and dealt with appropriately. This doesn't mean however that you can ignore them. It's important when designing a test (like I did in the example above) that you ensure that your harmonic aliases don't fall on top of each other, or on other things that might be in your spectrum. You should map out where they will fall to make sure you don't run into trouble and one pretty simple way to do that is to write an Excel spreadsheet that will calculate where the aliases will fall. It’s not too difficult to do, and will probably help to reinforce your knowledge of the rules.

If you end up coding your own kind of test, for example an Intermodulation Distortion (IMD) test and some of the products you are interested in do alias, then you will have to code the rules into your program to make sure you can extract the amplitudes from the frequency domain data you get back from the FFT. Occasionally you will find yourself out there, alone, then it’s up to you to figure out where the aliases will fall.

Conclusion

Aliasing is not something to be feared, it happens all the time, it can be easily understood, and the places where the aliases fall are easily calculated with two simple rules. If you ever doubt this, stand in front of a mirror with a tape measure and verify that measurements can be made by looking either at the tape measure directly, or by looking at the reflection in the mirror. Using aliasing is the same, while the view in the mirror may be reversed, the distances are just as valid.