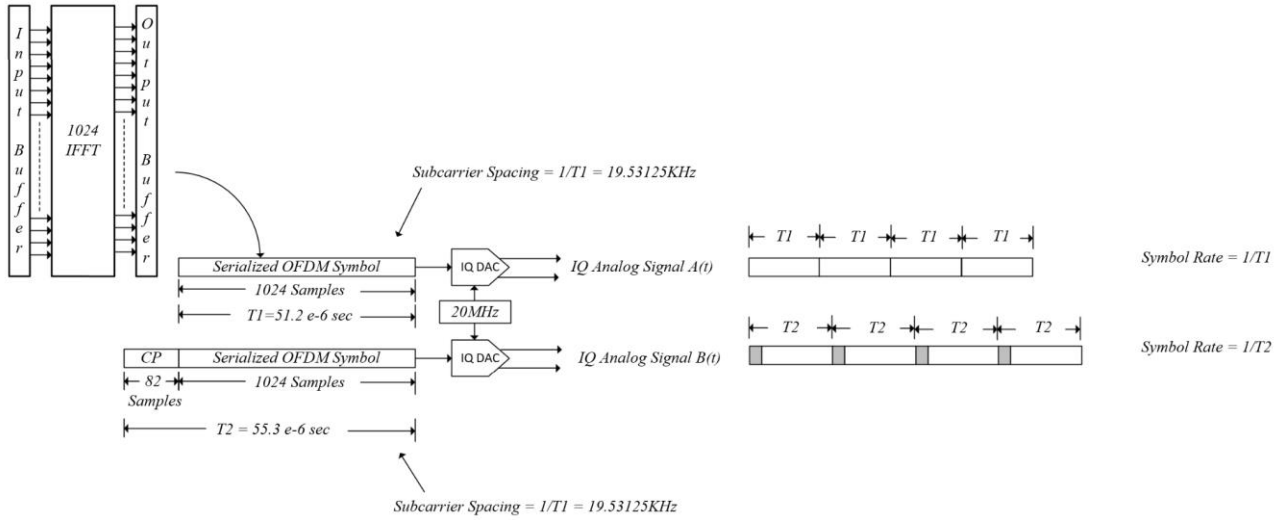

Hi Michelle,

Unfortunately, in our slack chat I couldn't get my point across, so I have drawn a picture that explains the difference between out viewpoints. Please take a look at the figure above. We will use YOUR definition of what an OFDM symbol is. It is the output of the IFFT, thus 1024 samples for our application.

By definition, the subcarrier spacing is equal to the clock rate at which the OFDM symbol is read out of the DAC divided by the length of the symbol which is 1024.

$$Subcarrier\ Spacing = \frac{20MHz}{1024} = \frac{1}{T1} = 19.53125KHz \qquad (1)$$

Now please look at signal $A(t)$ and $B(t)$. In signal A(t), I have concatenated the OFDM symbols directly. One OFDM symbol appears every T1 second and the OFDM symbol rate is thus as follows.

$$OFDM\ Symbol\ Rate = \frac{1}{T1} = 19.53125KHz = Subcarrier\ Spacing$$

Obviously, that is not what an OFDM modem does. It puts a CP in front of the IFFT output, as seen in signal $B(t)$. The period of this combo is T2 = 55.3 microseconds. And therefore, the rate at which OFDM symbols appear is not 1/T1.

$$OFDM\ Symbol\ Rate = \frac{1}{T2} = 22.12KHz \neq Subcarrier\ Spacing$$

If you try to compress the 82+1024 = 1106 samples so that their period becomes T1, rather than T2, then you need to increase the sample rate from 20MHz to 21.0601MHz. But if you do that, you have violated equation 1, and your subcarrier spacing is no longer what you thought it was.

The following is the equation for the IDFT.

$$x[n] = \sum_{m=0}^{N-1} X[m] \cdot e^{j2\pi nm/N}$$

The normalized carrier spacing of $x[n]$ is equal to $1/N$ Hz. The actual carrier spacing is equal to the normalized carrier spacing times the sample rate at which x[n] is used. Thus 20MHz/N, or 20MHz / 1024.

Therefore, the carrier spacing is a parameter of the DFT/IDFT equation, it is **independent** of the application of the Fourier transform in OFDM.

Let me reiterate what my motivation with FlexLink is:

1. I want to design a modem standard that runs circles around the modem that DJI uses in its drones.
2. I want the scope of the standard to be such that a small team of engineers can build it and not an army of engineers that work at Qualcomm.
3. I want to build a set of Python scripts that will completely simulate both transmitter and receiver and prove out all performance criteria that I have set out.

I am confident that I can achieve bullet 1, I know I can do bullet 3 and I hope I can achieve bullet 2.

FlexLink is the standard that Leonard and I work on. Neptune is your project, that I am happy to advise. There is another team in Estonia that we are also advising. They are implementing FlexLink on a signal processor, not an FGPA. And they are much, much further along than Neptune.

When you read about suffixes, HARQ, latency improvements and other interesting sounding concepts and you want to incorporate them into Neptune, then this is completely fine with me. But you are deviating from the FlexLink specification and at one point your team will need our Python scripts to produce test vectors. It is kind of pointless if your HDL and our Python scripts don't do the same thing.

Please know, that this subcarrier spacing issue and the OFDM modulation / demodulation process is something that competent engineers will understand even if it takes a bit. It was a bit of an effort for me as well. In time, both you and Leonard will have a thorough grasp of it. However, there are plenty of algorithms in the receiver that are FAR more difficult than OFDM Mod/Demod. With these things, you will have to trust me blindly, because you can not learn and understand concepts in 6 months that it took me 15 years to completely understand. Just like I can not pretend that I will learn and understand FPGAs and VHDL within 6 months to the level that Leanord understands it, or you understand it. If you have questions about FlexLink, you need to send me an email …. not go searching on the internet. Otherwise, Neptune will not see the light of day. It just takes too long. If you sincerely believe that a particular concept that you have encountered should be in FlexLink, then I will not ignore it. But it can't be something that you read in passing in some publication, or it is something that a grad student on your team heard about. It has to be a concept that you understand completely, that you can quantify and demonstrate in code completely and have a complete understanding of how its inclusion will affect the effort in both the physical layer and MAC layer of the standard. Otherwise, these potentially interesting concepts are … well, not so useful.

If Neptune is going to work, we need to put some faith in the competences of each participant.