Key Results and Conclusion*

Jaimeet Gulati

This work is produced by OpenStax-CNX and licensed under the
Creative Commons Attribution License 2.0†

Abstract

This section lists the key results of our projects and some final conclusions.

1 Key Results

The first implementation of our system involved sending an impulse over the sewing thread channel. The impulse wave was created by recording a quick flick on the microphone and then using the Sound Recorder on the computer to make a wave file out of it. This wave file can easily be plotted in MATLAB to see how the impulse looks like and if it actually produces a very high amplitude compared to the noise, in as little time as possible.

Once the impulse was received at the receiver cup, the microphone captured this sound and again this was saved as a wave file on the computer. This wave file can easily be read into MATLAB as a vector using the wavread function. The next step is to take the FFT of this impulse response to figure out the transfer function of the channel. The following plot shows you how the transfer function (frequency domain representation of the impulse response) looked like.

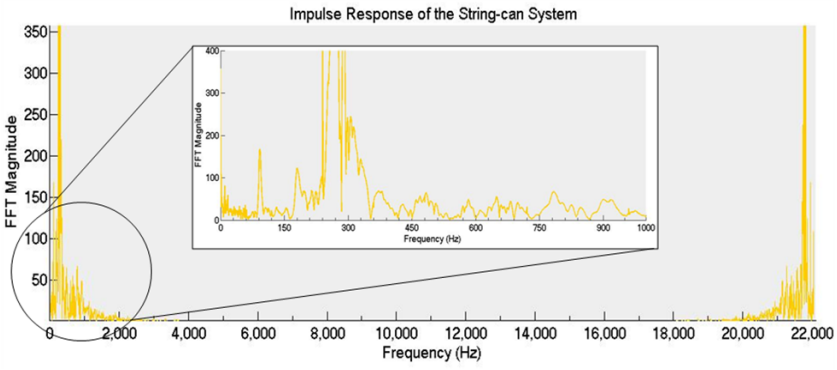

Figure 1

*Version 1.1: Dec 17, 2008 1:04 pm -0600
†http://creativecommons.org/licenses/by/2.0/

As you can see in the figure above, there is a major pass band at about 250Hz. This is exactly what we chose to send our data stream over the channel. A closer look would reveal that there are actually several other smaller peaks in the transfer function too, which could possibly be used to modulate data at different frequencies, and hence utilize the Frequency Division Multiplexing scheme.

The actual data stream that we sent over the channel involved modulating at 250Hz with a sine wave of an appropriate period. The modulated sine wave is shown below. Each point on the wave represent one data bit that we were sending over the channel (0s, 1s and -1s).

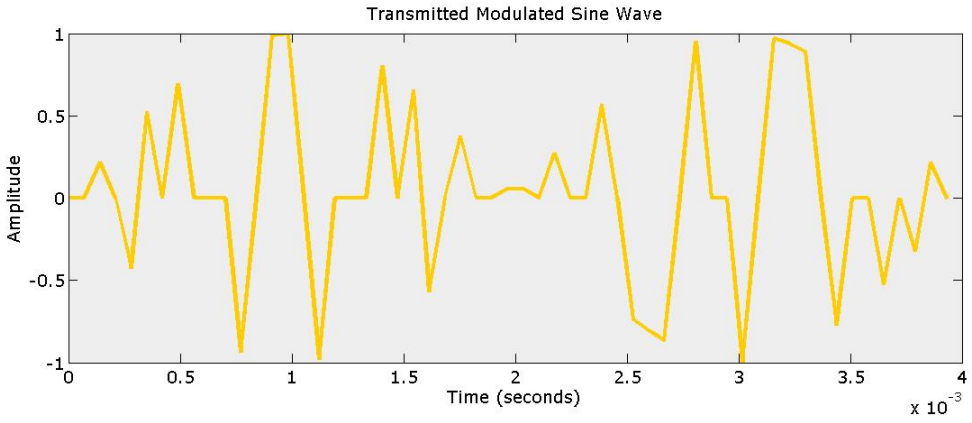

Figure 2

Also, to reduce the amount of bit errors at the receiving end, we sent multiple replications of the data stream one after the other. This allowed us to compare multiple strings of data at the receiver and then optimize the choice of the string that best represented the data at the transmitting end.

The spectrogram below will help you visualize the signal we sent over the channel.

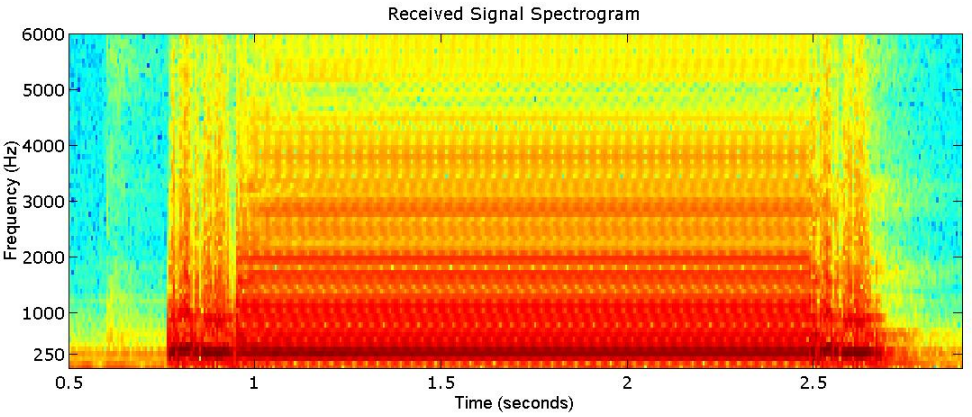

Figure 3

The figure above helps us prove and reiterate that the data we sent over the channel was in fact modulated at 250Hz – the horizontal dark red line at 250Hz suggests the major presence of that particular frequency in the transmitted data. Besides that, the multiple vertical sections that you see in the center of the spectrogram represent the data bits that are being sent over the channel at that particular instance. The presence of multiple sections indicates replication of the data bits, as I mentioned before.

Another important feature of this spectrogram is the presence of two impulse signals before and after the data bit stream. It is clearly visible that impulses of the same kind have been sent along with the actual data to help synchronize the receiver with the transmitter (as explained in the Channel Specifications module).

2 Conclusions

Here are some of the important conclusions and take aways from our project:

- DMT modulation allows us to transmit the word "Hello" over a channel as inconsistent as a string by taking different channel characteristics such as channel capacity, frequency pass bands and noise introduction into account before transmitting data
- Efficient data transmission fully exploits the bandwidth of the channel by using frequency division multiplexing by sending small, parallel blocks of data on orthogonal carriers

- Synchronization is one of the major challenges of using DMT modulation. One method of synchronizing the transmitted and received signals is to send loud bursts before and after the relevant data. This allows us to mark the start and end of the data in the received signal