|  |
| --- |
|  |
| ECSE 490 – Experiment 2 |
| Decoding a Signal in Noise |
|  |
| **Group 1: Isabel Helleur (260350713) and Loren Lugosch (260404057)** |
| **February 20th, 2013** |

|  |
| --- |
|  |

# Decoding a Signal in Noise

Some data is sent serially over a channel and when it has crossed this channel, it will have been affected by the channel itself as well as a possible addition of noise. Knowing this, we have decoded the original message of a noisy signal that was subject to some channel effects.

By examining an example signal, both before and after these effects, we were able to decide what methods to use to correct for the noise and channel effects.

## Noise Filter

Firstly, we compared the clean example to its noisy counterpart in time and found that they were similar to our own mystery signal.

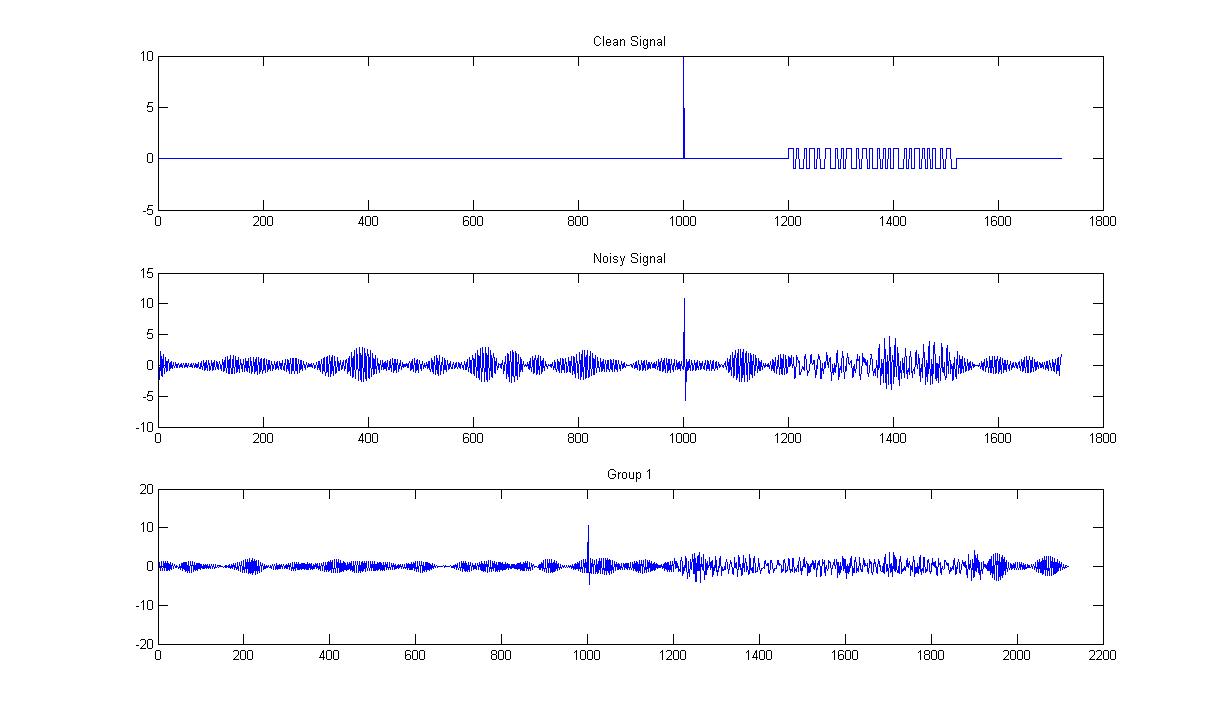


Figure 1 - Audio Signals

We examined the spectra of the group1 audio signal in the frequency domain and observed that there was a significant gap between the audio signal’s spectra and the noise, so we concluded it would be best to use a lowpass filter to cut off the noise components.

In Figure 2, the sample noisy signal and our group’s signal are scaled in the vertical axis to show the full magnitude of the noisy bands. Notice that there are bands containing useful information at around 1000 Hz in all three plots. The signal-to-noise ratio is roughly 3:25; hence the filter should attenuate the noise by at least a factor of 8.

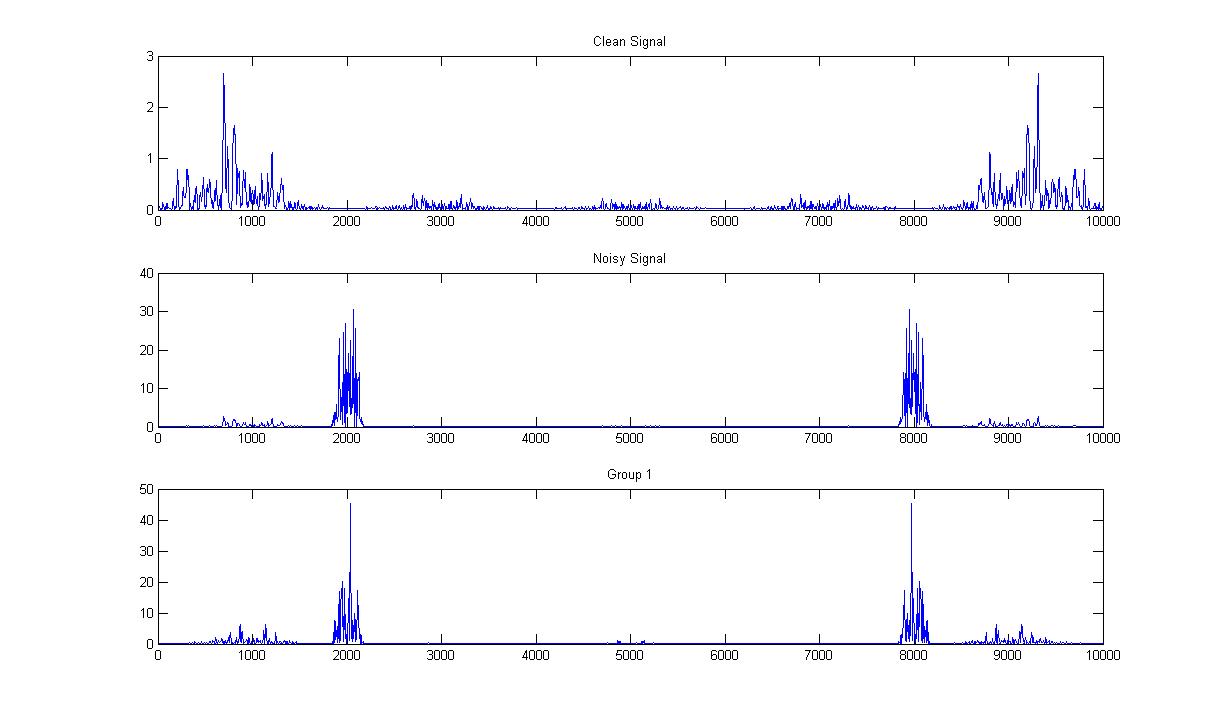


Figure 2 - Power spectra of the audio signals

Comparing the power spectrum for the audio signals to the theoretical power spectrum of a biphase-encoded signal (graph below), it can be seen that they both have the majority of their spectra centered around 1000 Hz and have a somewhat sinc-like shape. As for the noisy signals, they manifest another set of spectra at approximately 2000 Hz and since this is not seen in the theoretical power spectrum, we must conclude that these are all noise components.

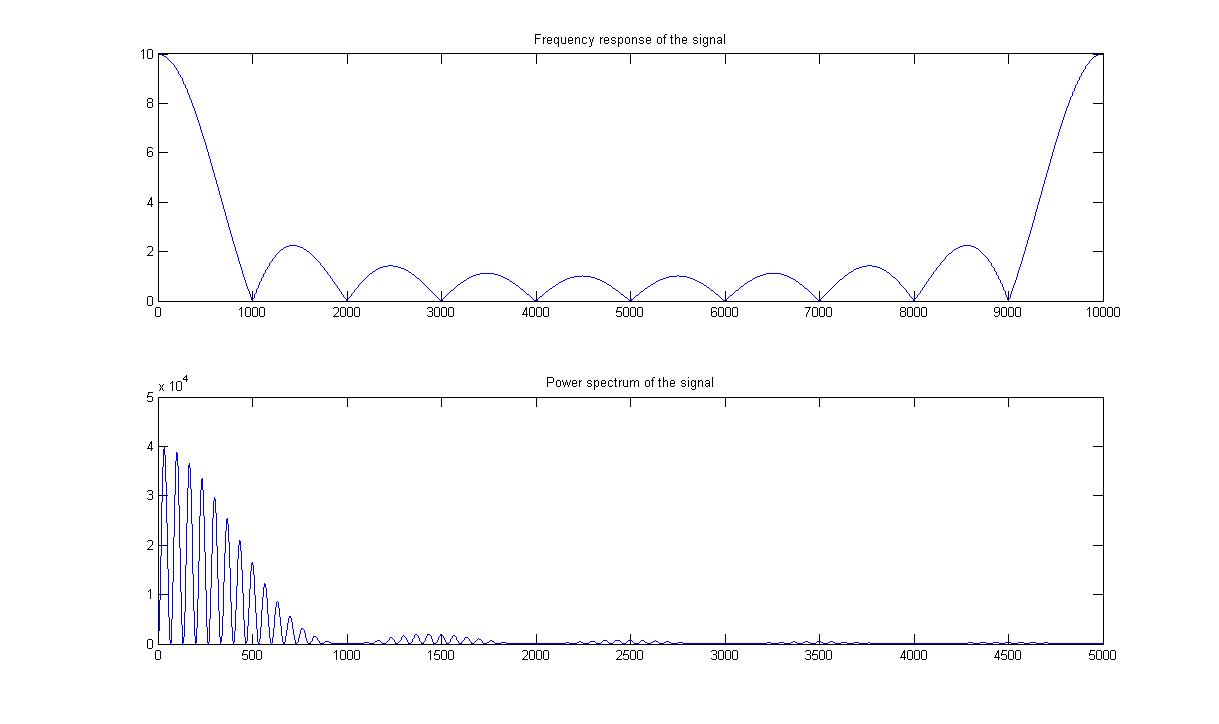


Figure 3 - Biphase power spectrum

We designed the necessary lowpass FIR filter using the SPTool with the following specifications:

Fpass = 1550; % Passband Frequency

Fstop = 1650; % Stopband Frequency

Dpass = 0.057501127785; % Passband Ripple

Dstop = 0.0001; % Stopband Attenuation

dens = 20; % Density Factor

order 253

3-dB point: 1566 Hz

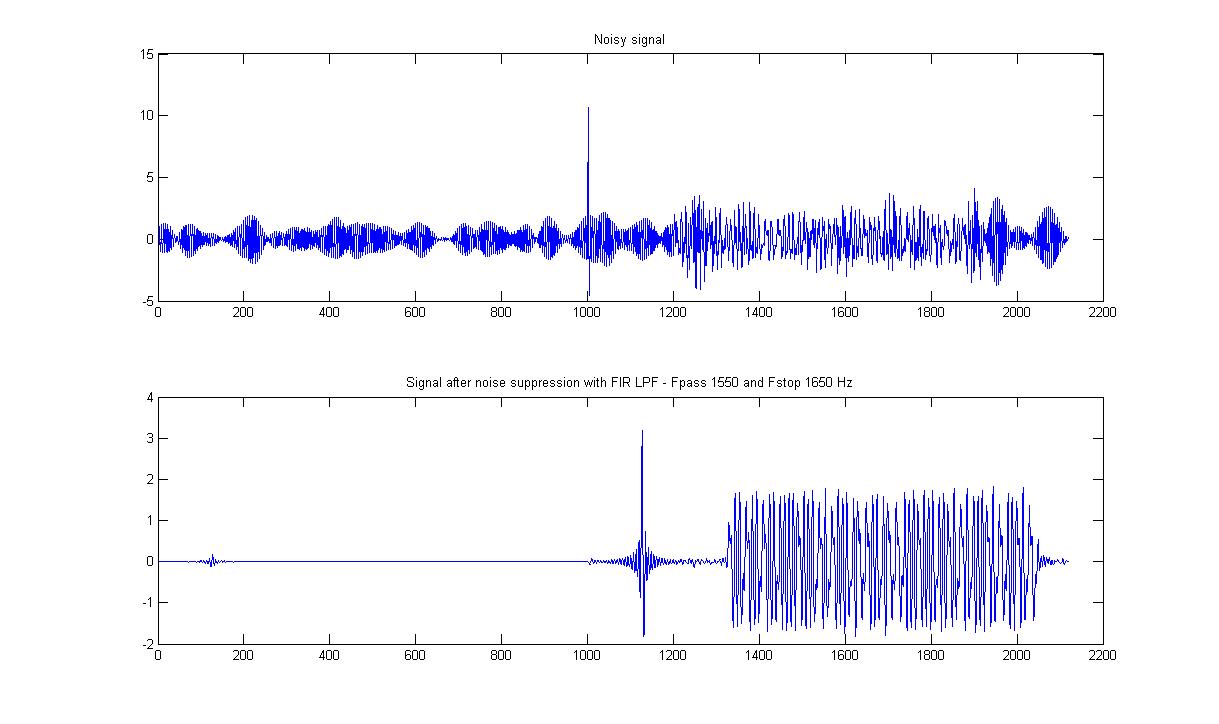


Figure 4 - FIR noise suppression filter

We then designed an IIR lowpass filter with the following specifications:

For the IIR:

Fpass = 1450; % Passband Frequency

Fstop = 1650; % Stopband Frequency

Apass = 1; % Passband Ripple (dB)

Astop = 80; % Stopband Attenuation (dB)

match = 'passband'; % Band to match exactly

order = 65

3-dB point : 1456

We applied the filter to the noisy signal:

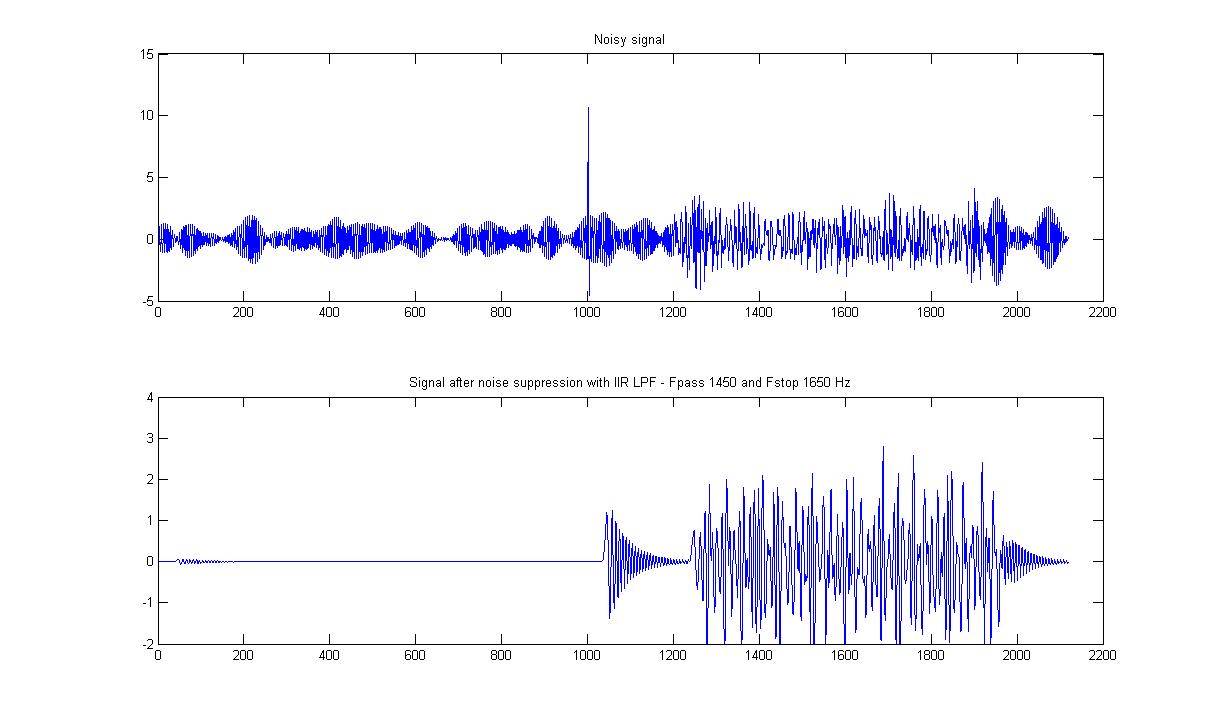


Figure 5 – IIR noise suppression filter

We tried a few different thresholds for the IIR filter in attempt to get a better cancellation of the noise, but a cutoff at 1450 to 1650Hz gave the best results and was still completely inferior to the FIR filter, so it was clear that we should use the FIR lowpass filter for this receiver system.

The FIR filter was of order 253. If the filter is implemented in cascade form (a delay followed by a multiplication + a delay followed by a multiplication + …), a computer will have to perform 253 multiplications + 253 additions = 506 operations to compute the value of the filter for some frequency.

## Channel Compensation (Equalization)

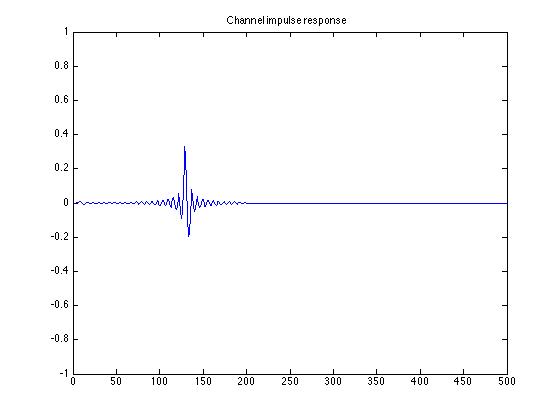
The original signal s[n] contains an amplified impulse followed by 199 zeros. Since we assume that the channel is a linear, time-invariant system, and the impulse response corresponding to such a system is the output when the input is an impulse, we can find the channel’s impulse response by taking values 1000-1200 of the channeled signal (remembering to divide them by the amplification factor, which was 10).

Figure - Impulse response of channel

We obtain the transfer function of the channel by simply taking the Fourier transform of the channel impulse response. Since we assume that the channel is LTI, we say that the channel is a rational function C(z) and its inverse is 1/C(z). The plots of their magnitudes are shown below.

**Note on minimum-phase systems:** If the channel response’s z-transform has zeros outside the unit circle, its inverse will have poles outside the unit circle, meaning that no region of convergence can be found which includes the unit circle, which in turn implies that any inverse channel is either unstable (if it lies outside the unit circle pointing outward) or anticausal (if it includes the unit circle pointing inward). Such a filter can be found, but cannot be realized in the time-domain. The workaround is to make the system minimum phase by introducing an allpass filter to move the zeroes of the C(z) outside the unit circle: C’(z) = C allpass (z) C minimumphase (z).

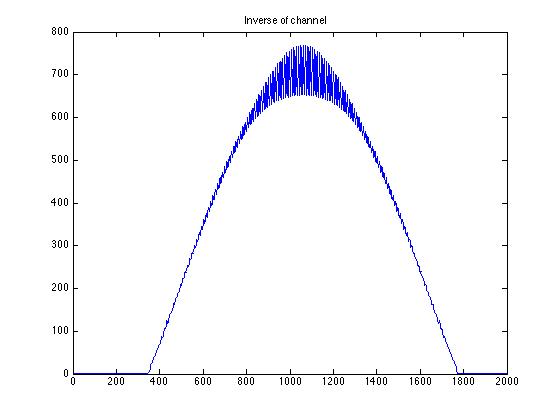
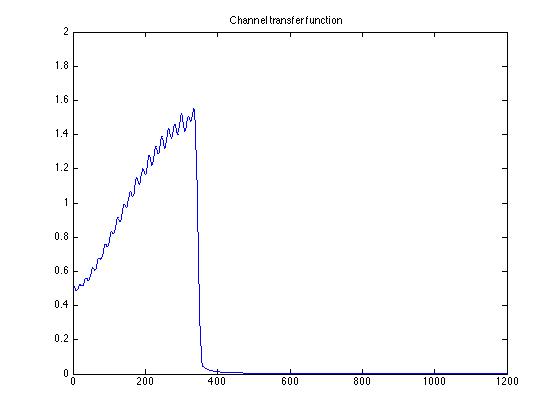


Figure 7 - Channel Transfer Function

Figure 8 - Inverse of Channel (obtained through element-wise inversion)

With the inverse channel transfer function, one can recover s[n] by either 1) multiplying the channeled signal in the frequency domain by the inverse channel and then taking the inverse Fourier transform or 2) taking the inverse Fourier transform of the inverse channel and convolving that with the channeled signal (if the dechanneling filter is anticausal, 1 is the only option). Since our impulse response is causal, we chose to perform 2. (If this were a real-time system, our filter could handle inputs in the time-domain using 2; i.e. the filter is realizable in the time-domain because it is causal). The recovered signal, s-hat-[n], is shown below. A simple low-pass filter blurs square waves because it attenuates their high-frequency components; hence s-hat-[n] is not a perfect reconstruction

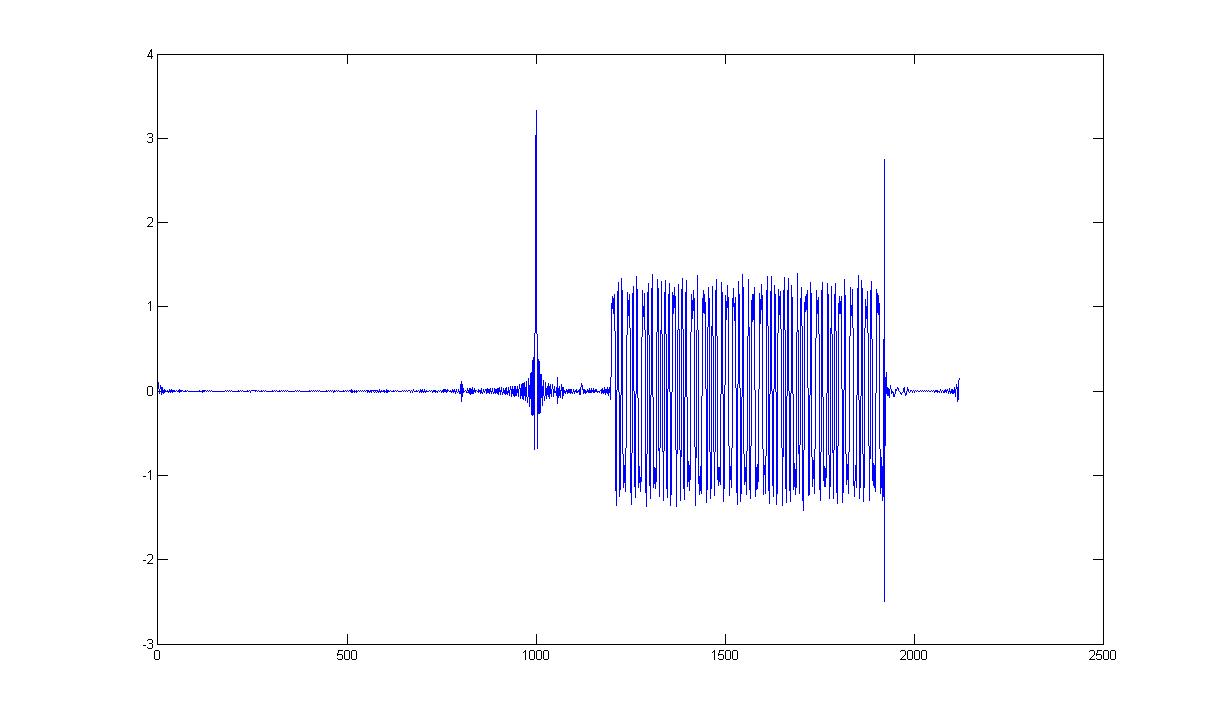


Figure 9 - Dechanneled signal

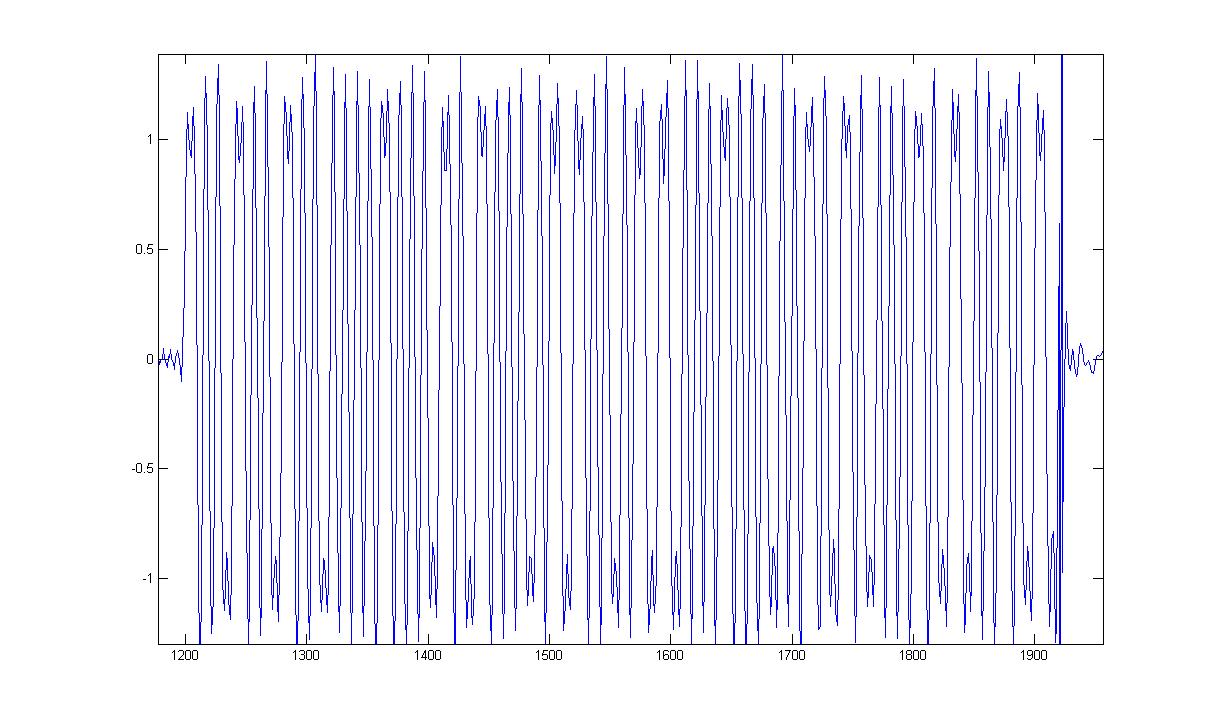


Figure 10 – Close up of information in dechanneled signal

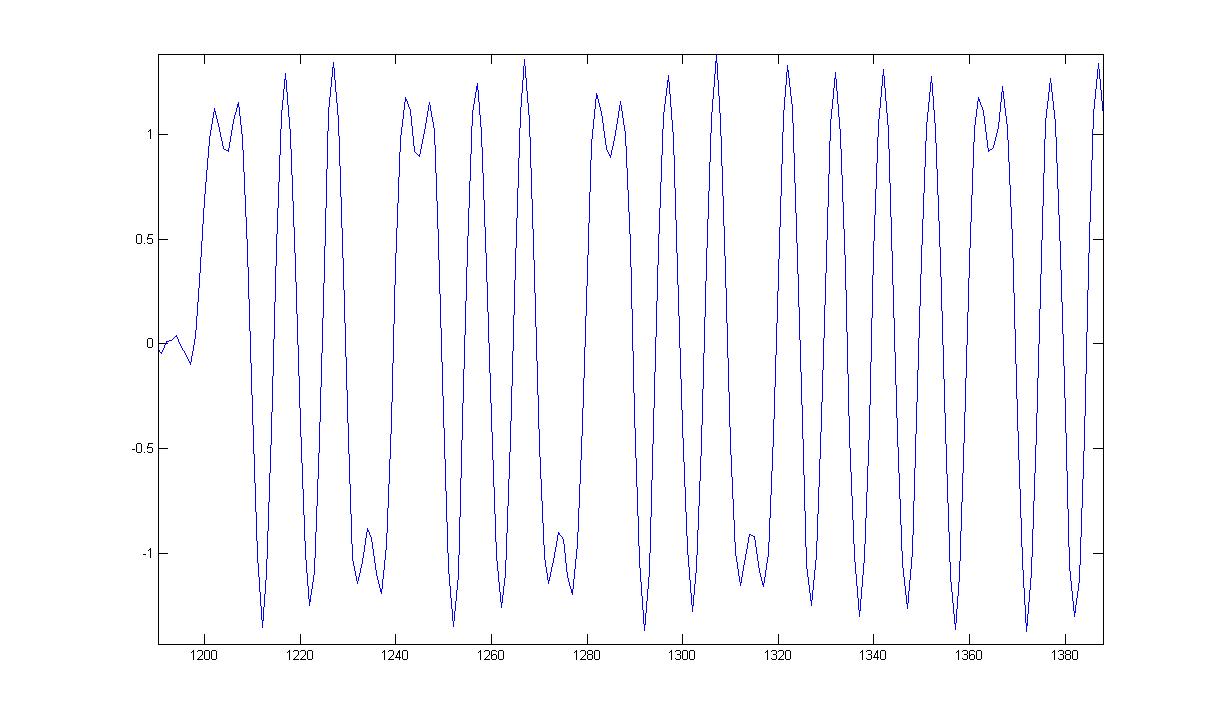


Figure 11 - Biphase encoded data of dechanneled signal

It was then possible to visually inspect the signal for the changes in state over time intervals to decode the message; the nine-letter word “forthwith”.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| 6 | | | | 6 | | | | 6 | | | | F | | | | 7 | | | | 2 | | | | 7 | | | | 4 | | | | 6 | | | | 8 | | | |
| f | | | | | | | | o | | | | | | | | r | | | | | | | | t | | | | | | | | h | | | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| 7 | | | | 7 | | | | 6 | | | | 9 | | | | 7 | | | | 4 | | | | 6 | | | | 8 | | | |
| w | | | | | | | | i | | | | | | | | t | | | | | | | | h | | | | | | | |

## Biphase coding

Biphase encoding depends on whether a change of state occurs over one clock cycle in order to determine if the encoded value is a ‘1’ or a ‘0’. As can be seen in the diagram below, if a value remains at +1 or -1 over a single clock period, the value is a ‘0’ and if it changes from +1 to -1 or from -1 to +1, then the value is a ‘1’.

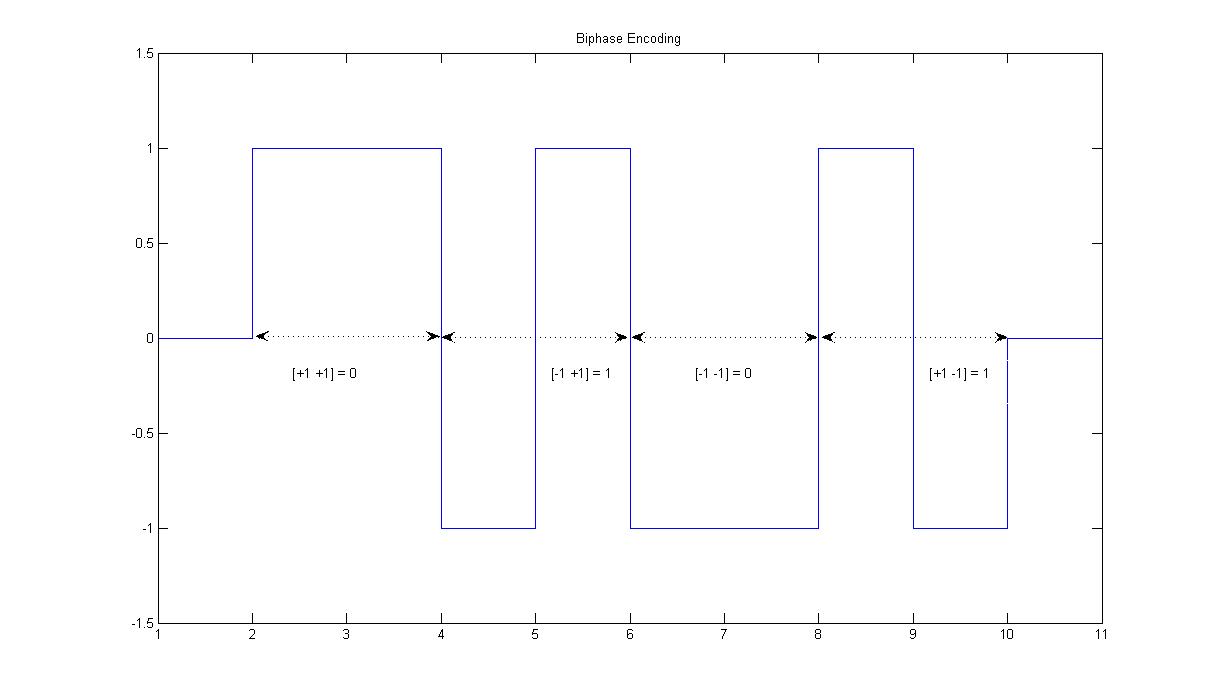


Figure 12 - Biphase coding scheme

|  |  |  |
| --- | --- | --- |
| Current State | Next State | Output |
| +1 | +1 | 0 |
| -1 | +1 | 1 |
| -1 | -1 | 0 |
| +1 | -1 | 1 |

