CDMA - Uncovering the Secret Message

Karol Wadolowski

***Abstract*—This document serves to explain how Prof. Hoerning used code division multiple access (CDMA) techniques to encode a secret message and how said message was decoded by yours truly. In this process two major topics will be covered, Walsh codes and PN sequences.**

***Index Terms*—Walsh Codes, PN sequences, LFSR, Hysteria**

I. Briefing

THIS report covers how secret agent (SA) Hoerning's secret message, received 122050RMAR20, was decoded. First off, the encoding method of the message will be discussed. Afterwards, the method used to decode the message will be explained. Lastly, the message itself will be revealed along with some commentary about the quality of equipment used here at command.

II. Encoding Secrets

The following explains the agreed upon encoding scheme used by SA Hoerning and the members at the command base. To encode a text message, the characters are first converted into their 8-bit ASCII equivalent. This bit message will be called the data stream. The data stream is the sent into the CDMA encoding system seen in Figure 1.

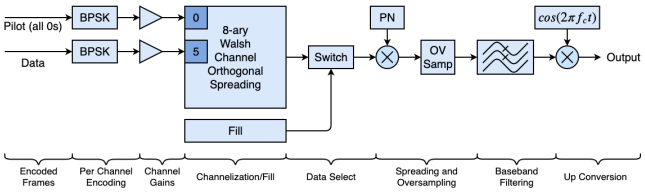

Figure 1. The CDMA encoder used to encode the secret message.

The encoder starts by taking the binary data stream and a bit stream of all 0s, the pilot stream, and mapping it to the BPSK constellation. The mapping is given by:

$$\begin{aligned} 0 &\rightarrow +1 \\ 1 &\rightarrow -1 \end{aligned} \tag{1}$$

After converting the pilot and data streams to BPSK, each of the two streams has an individual channel gain applied. For the specific case of SA Hoerning's encoding, the channel gains are set to 1. Afterwards, each stream goes into the 8-ary Walsh channel orthogonal spreader. The pilot stream corresponds to Walsh code 0 (w_0) and the data stream to Walsh code 5 (w_5). The 8-ary Walsh code matrix is:

$$W_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \end{bmatrix}$$

Lets go through an example on what the output of the Walsh orthogonal spreader looks like. Given the binary data stream $[0, 1]$ the corresponding BPSK stream is $[1, -1]$. The corresponding BPSK stream for the pilot is $[1, 1]$. Encoding the pilot stream with w_0 the output is:

$$\begin{aligned} p &= [1w_0, 1w_0] \\ &= [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1] \end{aligned} \tag{2}$$

Encoding the data stream with W_5 gives:

$$\begin{aligned} d &= [1w_5, -1w_5] \\ &= [1, -1, 1, -1, -1, 1, -1, 1, -1, 1, -1, 1, 1, -1, 1, -1] \end{aligned} \tag{3}$$

The output of the 8-ary Walsh channel orthogonal spreader is given by $p + d$. From this we can see that each BPSK value is spread over 8 values. If only one channel was being encoded then the Walsh spreader decreases the rate by a factor of 8. If there were 8 channels being encoded then the rate of the spreader would be 1. Just encoding the pilot and data streams decreases the overall rate by a factor of 4. The output of the spreader then goes into the switch, which along with the filler organizes the bits into frames as in Figure 2.

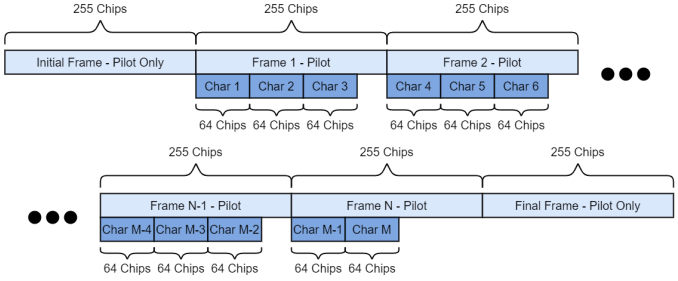

Figure 2. Frame structure.

Since each character corresponds to an 8 bit ASCII code, and since each bit will correspond to 8 values, after the Walsh spreader, then each character corresponds to 64 chips (samples). Each frame corresponds to 255 chips. The first 192

are composed of the output of the Walsh spreader and the last 63 come from the fill, which just gives binary 0s or in BPSK -1. It should also be noted that the first and last frame do not contain any characters and the second to last frame can contain 1 or 2 characters. The output stream from the switch, as seen in Figure 2, is then mixed with the PN sequence.

A PN sequence is a psuedorandom bit sequence. This means that actual sequence is not actually random but has properties that satisfy several tests for randomness. Given a PN sequence and a message, the two can be mixed together such that to outside observer the message is just gibberish. However, if the PN sequence is known by the transmitter (SA Hoerning) and the receiver (Karol Wadolowski at command base), then the mixing of the message and the PN sequence can be undone. This stems from the fact the multiplying the PN sequence (in BPSK form) by itself will eliminate the effect as $(\pm 1)^2 = 1$.

PN sequences can be generated using linear feedback shift registers (LFSR) as explained in [1]. The LFSR used for this CDMA system can be seen below.

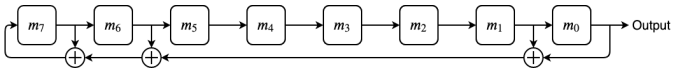

Figure 3. LFSR used in the CDMA system.

In this figure the m values denote the binary value stored in each register. If all the m values are not initially zero, then this LFSR produces an m-sequence of length 255.

After the signal has been mixed with the PN sequence, it is interpolated by a factor of 4. The resulting signal is then passed through a root raised cosine filter with coefficients given by the MATLAB function rcosdesign(.75,6,4,'sqrt'). The signal is then mixed with a carrier and sent. The channel over which the secret message was sent just added white gaussian noise, known as AWGN in the business.

III. Undoing the Protections

In order to decode the message, each step in the encoding has to be undone in reverse order. In a real scenario down converting the signal to baseband would be the first step. However, Karol Wadolowski at command base, charged with decoding the message, received it already at baseband. Taking the baseband signal, the first step is to filter it with the same root raised cosine filter and then decimate the signal by 4. The samples at this point can be seen in Figure 4.

In order to undo the PN sequence it is necessary to find the initial state that was used to generate the PN sequence at the encoder side. This can be done by exploiting the initial frame. The initial frame should ideally contain 255 copies of the value 1 since only the pilot is being sent. For an incorrect initialization of the LFSR, the PN sequence will not be correctly undone leading to an initial frame that looks like Figure 5. As can be seen, there are two main clusters meaning that the initialization was incorrect. The fact that the two clusters are not centered around ± 1 indicates there is a phase offset as well.

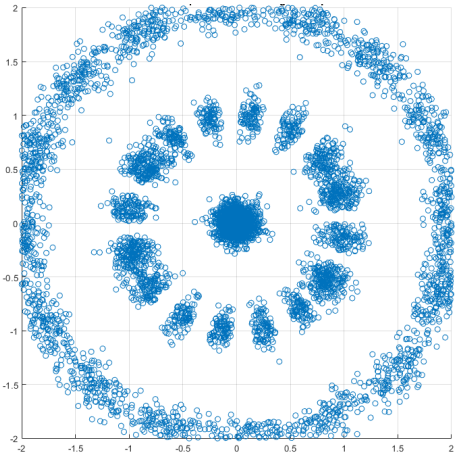

Figure 4. The received signal after filtering with the root raised cosine filter but before undoing the PN sequence.

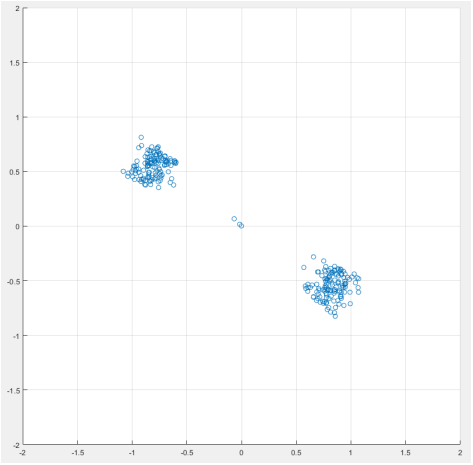

Figure 5. Initial frame decoded using the incorrect initialization of the LFSR.

Correctly undoing the PN sequence should lead to samples clustered around a single main point. The center of this cluster should also lie around one unit from the origin. With this knowledge, the distance of the mean of the samples for incorrect initializations should be close to zero. For the correct initialization the distance of the mean of the samples should be close to one. So trying all initializations leads to a graph like Figure 6. This graph has a single spike corresponding to the correct initialization of the LFSR. Using the correct initialization and undoing the PN sequence for the initial frame we get Figure 7. Undoing the PN sequence for the entire signal gives us Figure 8. It seems that undoing the PN sequence has done nothing but this is because the phase offset is not constant. It changes by a constant amount between frames. This would occur if there was a frequency offset between the transmitter and receiver. It would also explain why both Figures 4 and 8 have circles present. To undo the phase offset present in each frame it is necessary to recover the pilot for each frame. To do this the 192 samples corresponding to the characters of a frame must be separated into groups of 8

samples. Then take an inner product of the 8 samples and w_0, followed by a division by 8, to retrieve the pilot stream. Taking the complex argument of the mean of the pilot stream gives the phase offset θ for the frame. Multiplying the frame samples by $e^{-j\theta}$ will get rid of this phase offset. The phase offset for each frame can be seen in Figure 9.

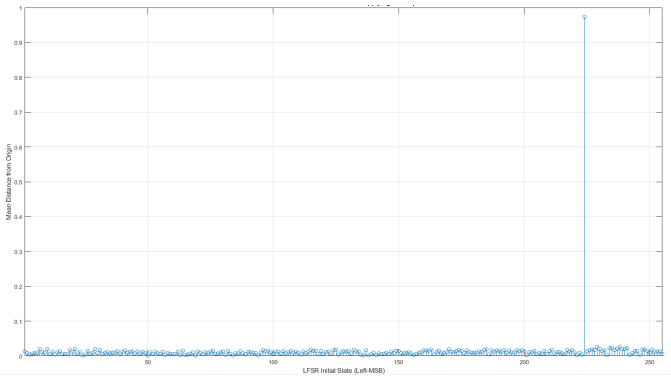

Figure 6. The distance of the mean of the samples for a given initial state.

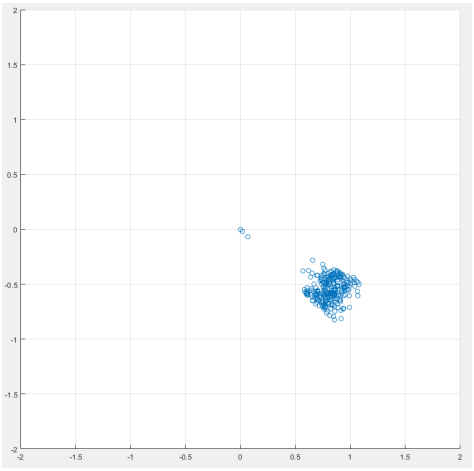

Figure 7. Initial frame decoded using the correct initialization of the LFSR.

Like what was done previously to get the pilot phase, the 192 samples can be taken and separated into groups of 8. This time we can take an inner product once with w_0 and separately with w_5 as well, followed again by a division by 8. This retrieves the pilot and data streams which can be seen in Figures 10 and 11. The two recovered streams can then be decoded from BPSK to binary based on their real component. The last step is to convert the data stream back into characters.

IV. The Message Revealed

After going through that entire hassle that was decoding, we can finally see the message that was hidden from us. The message is "It's such a magical mysteria, When you get that feelin', better start believin", which are lyrics from Def Leppard's Hysteria. This choice of message, knowing SA Hoerning is a huge music guy, probably conveys the extent

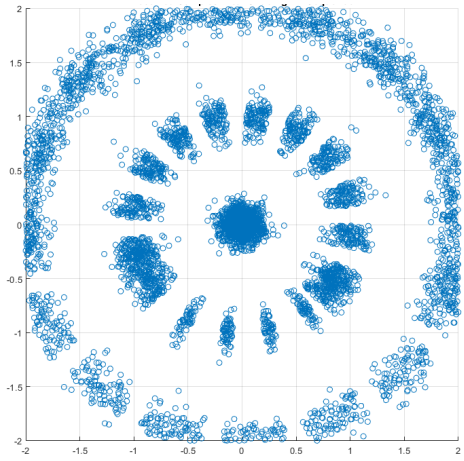

Figure 8. The received signal after correctly undoing the PN sequence.

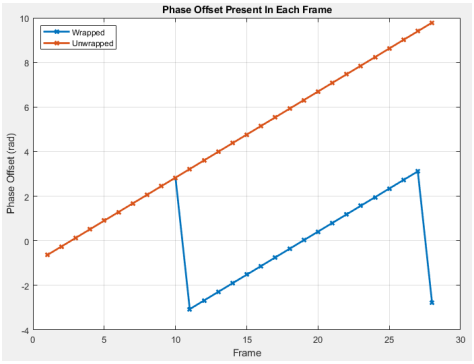

Figure 9. The wrapped and unwrapped phase offset for each frame.

to which he likes this song. He may also have some great memories associated with it. Whose to say? It must mean something to him if he decided that its better not to agree on an initial state for the LFSR.

While decoding there was also the issue of the frequency offset (mismatched LOs are always annoying to deal with). To get the frequency offset all we need to do is use the relation $\frac{d\theta}{dt} = 2\pi f$ and the fact that the chip rate is 1 MHz. We can use the chip rate to get a frame rate by dividing by the number of chips in a frame. This gives a frame rate of roughly 3.92 kHz. We can use the frame rate and the unwrapped phase graph to get $\frac{d\theta}{dt}$. Dividing this by 2π leads to a frequency offset of approximately 240 Hz. We really need to convince the head of command to buy better LOs.

V. Final Remarks

To summarize, SA Hoerning encoded deeply personal song lyrics using a simple CDMA system. He then decided to make things difficult by not agreeing about the initial LFSR state. But this is what it takes to prevent outside forces from intercepting the message. However, as the intended recipient of the message, the encoding scheme was known and thus a decoding was made less cumbersome. The message was decoded and left the members of this agency and myself to

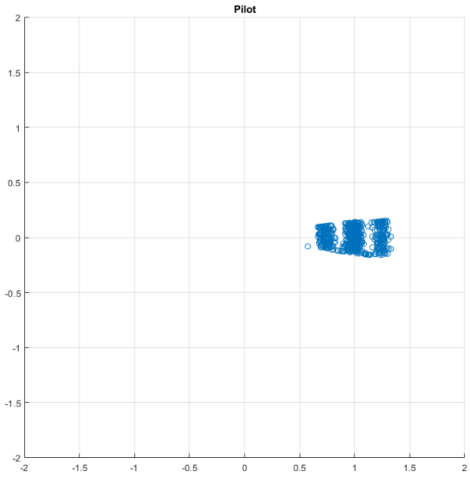

Figure 10. Recovered pilot signal.

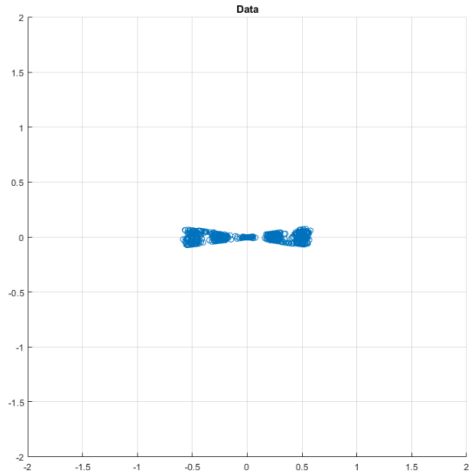

Figure 11. Recovered data signal.

ponder what the reference meant. Maybe we weren't meant to know, maybe we were meant to sit back and relax as Hysteria played in the background, whose to say? Also we need better LOs. A 240 Hz offset! Like seriously that is not acceptable.

REFERENCES

[1] "Linear feedback shift registers," http://in.ncu.edu.tw/ncume_ee/digilogi/prbs.htm.