EE230B Final Project

**Design Implementation Considerations and Performance Analysis of 802.11n (EWC Proposal)**

*Gourav Khadge and Nathan Wong*

**1. INTRODUCTION**

In this report, we develop the end to end design of a commercial WLAN Physical layer, based on an 802.11n standards proposal proposed by the Enhanced Wireless Consortium (EWC). The implementation design space is discussed and tradeoffs are analyzed. The design performance is shown with analysis on the importance of each element of the design. The elements under consideration include: the Automatic Gain Control (AGC), Block Boundary Detection (BBD), channel estimation, and carrier frequency estimation. All of these design elements were simulated and analyzed in MATLAB.

For the purposes of this report, only Greenfield mode was considered under several limited modes. Namely:

* The following antenna configurations: 1x1, 1x2, and 2x2 spatial multiplexing
* Only QPSK, 16-QAM and 64-QAM modulations were studied
* Only the 20 MHz mode was implemented
* Only 0.25 KByte packet payloads were used
* No channel coding was implemented
* 20 MHz bandwidth mode
* 2.4 GHz carrier frequency

**2. BACKGROUND**

802.11n is WiFi standard for packet-based OFDM wireless communication. It can be set up in SISO, SIMO, MISO, and MIMO configurations with various numbers of transmit and receive antennas. The EWC proposal modulates OFDM using 56 subcarriers with 4 pilot tones and 52 data subcarriers spaced 312.5 KHz apart. Each of the data subcarriers can be modulated independently. In this report we study QPSK, 16-QAM, and 64-QAM for data subcarrier modulation. Each OFDM packet consists of a preamble including one Short Training Frame (STF) and one or more Long Training Fields (LTF). The first LTF is a special LTF that is repeated for two symbols. For our study, every packet was transmitted with a total of four LTFs. In general 802.11n requires a “Signal” field which includes metadata such as the length of the data in bytes. In our study, the signal field was omitted.

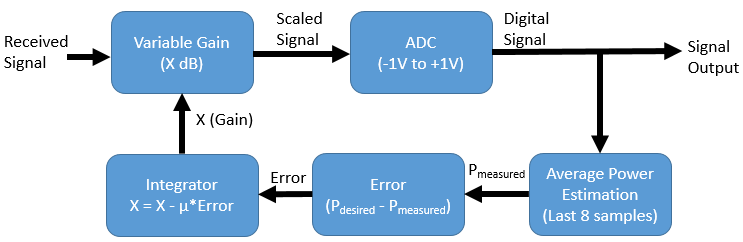
**3. DESIGN**

At the receiver, we implement an AGC to allow the receiver to handle a large dynamic range with a fixed Analog to Digital Convert (ADC). if a packet preamble is detected, the AGC is halted and locked after 3 microseconds from the estimated start of the packet. An ADC is simulated by clipping the AGC output at +/- 1 V. Using the Short Training Frame (STF) in the 802.11n preamble, a coarse estimate of the carrier frequency offset is obtained and a correction is applied to the signal. Block Boundary Detection (BBD) is then performed to more precisely identify the timing of the packet by identifying the boundary between the STF and the first Long Training Field (LTF) symbol. The LTF1 symbol is then used to perform a fine frequency correction. For our study, four LTFs are transmitted in all packets. After fine frequency correction, these are used to obtain an estimate of the channel, which is used to undo the effects of the channel. After all of this, the received OFDM symbol is demodulated and its data is extracted. We judge the performance of the system on the bit error rate at various SNRs.

**3.1 AGC**

The purpose of the AGC is to make the best usage of the dynamic range of the ADC. In this case, we assumed the ADC would operate between -1 and +1 Volts, and any input beyond this range would be clipped. The AGC adds a variable gain to the received signal, which may vary in power over a large dynamic range. The goal of the AGC is to scale the signal such that the signal voltages are well distributed across the full range of the ADC with minimal clipping. There is a trade-off between how much clipping is in your signal and how well the dynamic range of the ADC is utilized. This is especially true in OFDM which can suffer from large Peak to Average Power Ratio (PAPR). Large PAPR implies large peak voltage spikes in the signal, which are susceptible to clipping.

The structure of our AGC design is shown in Figure 1. The AGC is a first order loop. The received signal is amplified by a variable gain amplifier. The scaled signal is then received by the ADC. The average power is estimated using the last 8 samples. This measured power is then compared against a user-defined average power and the error is calculated. The error is scaled by an adaptation factor, μ, and accumulated in an integrator. The output of the integrator is used as the variable gain. We found it helpful to clip the value of the integrator to maximum and minimum values thus limiting the necessary dynamic range required by the AGC algorithm. The AGC was designed to settle within 0.1 dB of the desired power within 3 μs with a dynamic range of 80 dB. The desired power is specified by the user and should be chosen based on the PAPR statistics and tolerable amount of clipping. In our algorithm, we chose -6 dB as the desired power. The adaptation rate, μ, was chosen to be fixed at 0.1. We found these parameters to give satisfactory performance.



***Figure 1:*** *Our AGC design.*

There are several ways this design can be adjusted. Particularly, the way that average power is estimated, and the adaptation rate. The average power is estimated using the last 8 samples. This was a tradeoff. Using more samples would allow noise to average closer to zero, but it would increase the delay in the system. Additionally, if the power was changing rapidly, the estimated power would lag longer behind the actual instantaneous power. For real-time operation of the signal, it is important to accurately measure the instantaneous power. Additionally, the power could be measured before the ADC using analog hardware. However, the clipping of the ADC was actually beneficial for our purposes so the average power estimation wouldn’t be affected as strongly by very large powers. Particularly, before the preamble, the AGC is just amplifying noise at a high gain. When a high SNR preamble enters the system, the AGC will amplify the signal well into saturation for several samples before the algorithm has time to react.

Increasing μ allows it to adapt faster, but it will also make it harder to settle, and might overshoot. A possible improvement on the algorithm is to use a variable μ that is large when the error is large, but smaller as the error decreases for improved settling performance.

**3.2 Packet Detection**

Packet Detection determines whether or not the receiver is able to detect an incoming packet. In the 802.11 standard, a packet begins with what is known as a Short Training Field (STF). The STF is a known sequence that can be used by the receiver to assist in some necessary steps of decoding, including packet detection and coarse frequency estimation. There are many methods in which detection of the packet can occur.

Some methods are described in Lecture 11: Life of a Packet. One could hypothetically detect the energy of the incoming packet via a double sliding window energy detector, but this does not differentiate from other types of packets. Another option takes advantage of the structured repetition of the preamble and implements an autocorrelation method in comparison to a delayed period of the STF, which is of 0.8 microseconds. However, this method can prove disastrous given that any repetitive signal that meets such a qualification will be considered a packet. Finally, one can use a matched filter to the STF to obtain an accurate detection method.

Our design implements a combination of the latter two methods: autocorrelation and matched filter. In our experimentation of choosing the best packet detection method, we found that the signal was too noisy to solely rely on even a matched filter construct of one STF period. When we attempted to lower the threshold for matched filter approval, we encountered many false detections, as noise can falsely represent an STF period. To deter this, we added an autocorrelation check to verify that the signal was indeed repetitive. The autocorrelation is normalized according to the delayed signals energy. In our simulations, we found that this was sufficient for SNR values of around 10 dB and above in Rayleigh fading, but demonstrated difficulty when the SNR was lowered.

Our matched filter uses one STF period for maximum detection possibility. We choose an autocorrelation size of two STF periods to allow for a better comparison of the repetition of the signal. We found that the combination of these two filters allowed for the most accurate packet detection at the receiver. It is recognized that this creates a more complex system, but given the channel that the receiver must endure, we decided it necessary to implement some combination of techniques to succeed.

Given an opportunity to improve the packet detection, we would experiment on the size of the autocorrelation and matched filter. Additionally, we would vary the thresholds to experiment if different thresholds would improve accuracy at lower SNRs while retaining a small percentage of false packet detection.

**3.3 Coarse Frequency Estimation and Correction**

The algorithm used for coarse frequency estimation was adapted from Supplementary Handout 7: OFDM Synch. The STF is composed of 10 repetitions of a 0.8 microsecond symbol. Sampled at 20 MHz, these repetitions have a period of 16 samples. Thus, we use the algorithm below, where r is the received sampled signal, D is the period of repetition, and Ts is the sample period.



In Supl 7, the performance of this estimator is also given:



Supl 7 also details a frequency domain based method of estimating the phase offset, but shows that the performance is not better than this time domain method. The time domain method is preferable because it is less computationally intensive.

In our design, we run this algorithm using the samples that are 60 from the start to 120 from the start. The entire STF is 160 samples long. However, the AGC only locks after 3 microseconds (60 samples), and the packet detection’s estimate of the start of the packet can be off by as much as two sample periods (32 samples) or more. Thus, we found this 60 sample period to generally be well guaranteed to fall inside the range of the STF. This was found to have sufficient performance for our purposes. The coarse frequency detection only needs to be accurate enough to reduce the frequency offset to a range that the fine frequency detector can handle.   
  
Notably, the max frequency offset that this algorithm can detect is inversely proportional to D.



As will be discussed in 3.5, the fine frequency detector uses the same algorithm, but with a D of 64. Thus, the main job of the coarse frequency detector is to at least reduce the frequency offset to within the range of the fine frequency detector, though good coarse frequency correction will also aid BBD.

Once the frequency offset is detected. A correction signal is generated as a complex exponential with the negative of the detected frequency. This signal is then multiplied into the received signal to cancel out the offset.

**3.4 Block Boundary Detection**

With the structure of the packet, it is necessary to choose a point to determine the location of the following packet components. Given that packet detection immediately leads to coarse frequency estimation, the signal can be corrected sufficiently to determine the STF and Long Training Field (LTF) boundary.

The algorithm for block boundary detection is derived from Lecture 11: Life of a Packet. For block boundary detection, a matched filter is used. This matched filter contains the ending samples of the STF and the beginning samples of the LTF. We choose the length of each side to be 4 microseconds, or 80 samples, long to provide sufficient determination that a point is the block boundary. When a shorter amount was used, the correlation to the STF would rise, giving the notion that the HT-LTF boundary was located in the STF field. Using too many samples would prevent the autocorrelation from showing within the signal, due to noise.

In our simulations, we found that the matched filter was able to detect the starting point of the HT-LTF signal within a 3 sample range. This is sufficient for the rest of the receiver processes. Thus, we determined that a single matched filter containing both STF and LTF points was ideal for our simulations.

Alternatively, two matched filters could have been made, one with the STF and one with the LTF. By running the signal through both filters, one could find the location of the LTF where the peak begins to decrease in size within the STF and reaches its maximum in the LTF. We determined the two methods to be the same and found that the combined matched filter proved easier to implement, so we decided to continue using the said filter. Should the case arise where a more accurate HT-LTF start location need be implemented, we would look to see if the double window filter would prove more useful.

**3.5 Fine Frequency Estimation and Correction**

The algorithm used for fine frequency estimation was adapted from Supplementary Handout 7: OFDM Synch, and is the same one described in Section 3.3 of this report. The only difference is where it is applied. The first LTF, LTF1 is 8 microseconds long and is composed of two and a half repetitions of a 3.2 microsecond period, or 64 samples. Since BBD has already occurred, we are also able to use all 160 samples to compute our detected frequency. The correction signal is generated as a complex exponential and this correction signal is compounded into the coarse correction signal to give the total frequency correction to the signal.

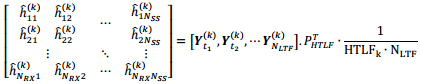
We found that so long the HT-LTF boundary was within an acceptable distance from the actual boundary, we were able to use fine frequency correction to our advantage.

Given the opportunity to further improve fine frequency estimation and correction, we would consider the effects of how to correct the frequency at each provided pilot signal instance. Currently, with such a large packet, we may experience large amounts of fast fading within those later payload values. By applying fine frequency correction using the pilot signals, we may be able to correct the frequency and assist correct detection even further.

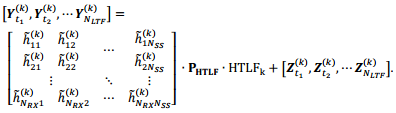
**3.6 Channel Estimation**

Our algorithm for channel estimation was based on the paper

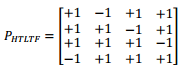
“IEEE 802.11n: On Performance of Channel Estimation Schemes over OFDM MIMO Spatially Correlated Frequency Selective Fading TGn Channels” authored by Roger Pierre and Fabris Hoefel. We chose to use this algorithm because it generalized easily to MIMO systems, which was not true of the method in Supl. 7. The algorithm uses the four LTFs in the preamble, and all measurements of them from each of the receive antennas. The 4 LTFs all contain the same symbol, but, one of the four is negated, depending on which spatial stream it belongs to, according to the polarity matrix P.



The algorithm is derived from this equation detailing how the measured received signal is related to the input acquired from the spatial streams. If there are more receive antennas than spatial streams, this equation has an invertible formulation (with the noise having an expected value of zero.



The polarity matrix is described in the WiFi standard and is copied below.



Using this algorithm, the least squares estimate of the channel matrix can be computed, using all information of LTFs from all receive antennas. There is one channel matrix for every subcarrier. Once this channel matrix is found, its inverse can be found (pseudo-inverse if there are more receive antennas than spatial streams). Since Y = H\*X, then X = inv(H)\*Y, where X and Y are the received signal and transmitted signals for each subcarrier.

We assume that the channel stays roughly constant for the duration of the packet. This is mostly true for our assumptions because the packet duration is very small (Less than 100 microseconds), while we assume Rayleigh fading has a maximum doppler frequency of 100 Hz.

To improve the estimate of the channel, and allow the channel estimation to adapt over time as fast fading changes, it would be possible to use the four pilot tones to help refine the estimate of the channel. This wouldn’t be perfect because the pilot tones are separated in frequency from the other subcarriers and may not necessarily be coherent with all others. However, the pilot tones are well spaced at k = -21,-7,+7,+21 and will likely be coherent with neighboring subcarriers. Another option may be to use a decision directed approach where the decision output of each symbol is used as a reference to estimate the channel distortion.

Each of these extended methods may improve performance, but at the expense of complexity.

**4. SIMULATION**

**4.1 SNR calculation**

Signal-to-Noise Ratio (SNR) is a comparison metric between signal and noise, usually defining a desired signal strength to obtain sufficient energy over the noise within the channel. Given a user defined SNR, there exists a method in which Noise can be randomly generated.

In our determination of SNR at the receiver, we determine that the power of the received signal should solely depend on the signal Path Loss, or the loss of signal strength across a distance d (we denote d = 50m in our simulations). Thus, we will ignore effects from Shadow Fading and Rayleigh Fading. From this, we calculate the average energy in our signal.

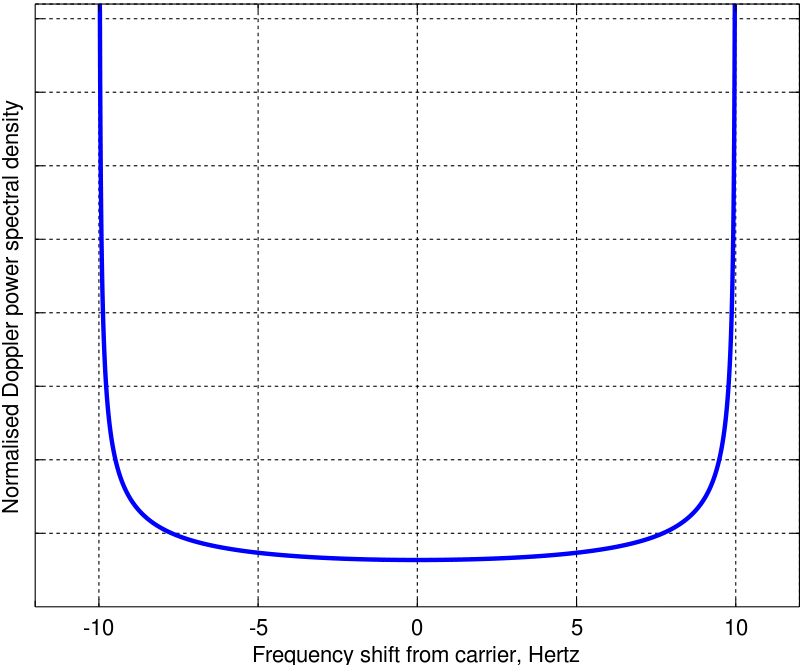
Then taking the SNR in the linear domain, we can divide the symbol energy by the SNR to obtain Noise energy, N0. This can be multiplied to a complex Gaussian random variable with zero mean and a variance of 1 (additionally divided by 2 as N0/2) to create the noise that matches the user defined SNR. Thus, we will use this method to generate noise throughout our simulations.

**4.2 Multipath Generation**

Multipath signals are generated using sinc interpolation for specified signal delays, as well as specified relative powers. Each multipath component is given the same path loss, but is randomly phased, and is simulated with independent shadow fading, rayleigh fading, and AWGN.

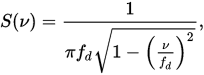
**4.3 Shadow Fading**

Shadow Fading is a slow fading value. Thus, we model this as a singular value multiplied to a multipath component. Each multipath should have a different shadow fading value, as the channel is different for each component. We calculate shadow fading from a log-normal distribution.

**4.4 Rayleigh Fading**

Rayleigh Fading is a fast fading value. Rayleigh Fading in its definition is a complex Gaussian. However, this complex Gaussian is dependent on a Doppler frequency fd that dictates how fast the fade is. To properly simulate, one must create a filter with respect to the power spectral density:

***Figure 2:*** *Power Spectral Density of Rayleigh Fading*



We further note that the complex Gaussian must pass through the filter in order to have a correct distribution.

There are alternative ways to create this distribution. One such popular distribution is Jake’s model. The idea derives from a large summation of sinusoids to create a pseudorandom sequence. Given an opportunity to better the fading model, we would attempt this method.

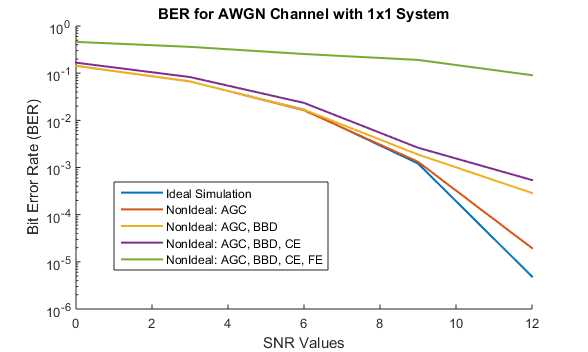
**5. PERFORMANCE**

A few calculation specifications must be made here. To calculate the Bit Error Rate (BER), we calculate all the bit errors that occur within the payload. Additionally, any missed packet due to packet detection will be considered a completely erroneous packet, and all bits will be counted against the BER.

Our ideal AGC assumes a set power and remains constant throughout the simulation. In Block Boundary Detection, we assume that having an ideal Block Boundary means that we know the starting location of the packet and HT-LTF. Thus, we do not go through packet detection and set the AGC to lock 3 microseconds after the start of the packet. Ideal channel estimation essentially takes in the packet without noise to estimate the channel, producing a perfect estimation. Ideal frequency estimation assumes that there is no frequency phase error.

In the following plots, we denote Automatic Gain Control as AGC, Boundary Block Detection as BBD, Channel Estimation and correction as CE, and Frequency Estimation and correction as FE.

**AWGN Channel with No Multipath, M = 4**

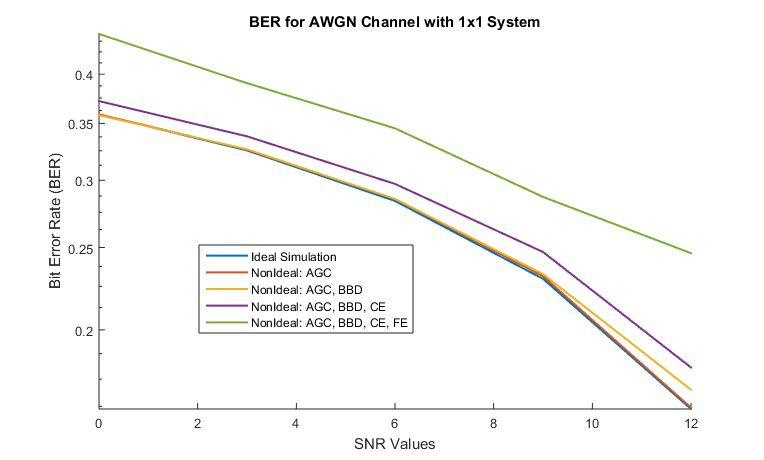
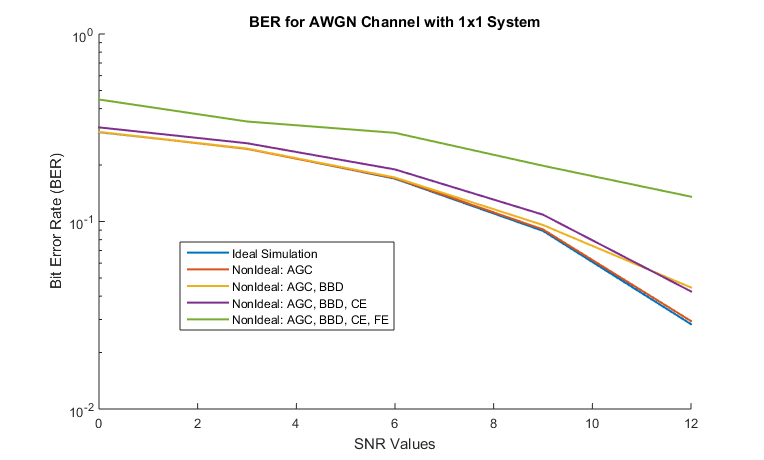


***Figure 3:*** *BER plot of AWGN Channel with No Multipath for M = 4*

Figure 3 demonstrates the effects of each non-ideal portion for an AWGN channel with no multipath and an MQAM of M = 4. We see that as more non-idealities get added onto the receiver, the worse the BER gets in general. This is especially true at higher SNR values, where the non-idealities begin to have a greater impact as compared to noise. In Figure 3, it appears the most significant contributor to degradation from ideal performance is the frequency estimation step. This makes sense as OFDM is famously susceptible to carrier frequency offset. The next largest contributor appears to be the block boundary detection. A major degradation in our simulations once ideal BBD is removed appears to be dropped packets. Namely, either our packet detection algorithm fails to detect the packet or the packet’s boundary is severely mis-estimated, causing a large number of bit errors at once. Thus, any efforts to improve the implementation developed here should prioritize focus on these two particular elements.

**AWGN Channel with No Multipath, M = 16 and M = 64**

Figures 4 and 5 show the two plots in comparison, shown in Figures 4 and 5. The left plot is of M = 16 and the right plot is of M = 64.

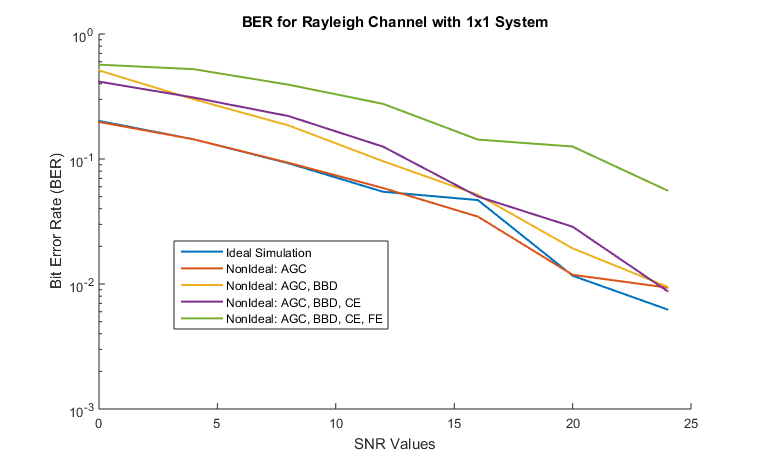


***Figure 4:*** *BER, AWGN, No Multipath, M = 16* ***Figure 5:*** *BER, AWGN, No Multipath, M = 64*

We first notice that the progression is the same. From the ideal, we marginally increase in error at non-ideal AGC and BBD. A small increase in BER after channel estimation is expected, with a much larger increase of BER at the frequency estimation side. Thus, we assume that this is consistent for all MQAM versions.

Second, we see that the BER increases as M increases. This makes sense, as higher M values in MQAM produce more crowded constellations. Thus, we can also conclude that this persists for all values of M.

**Rayleigh Channel with No Multipath, M = 4, Doppler Frequency = 10 Hz**

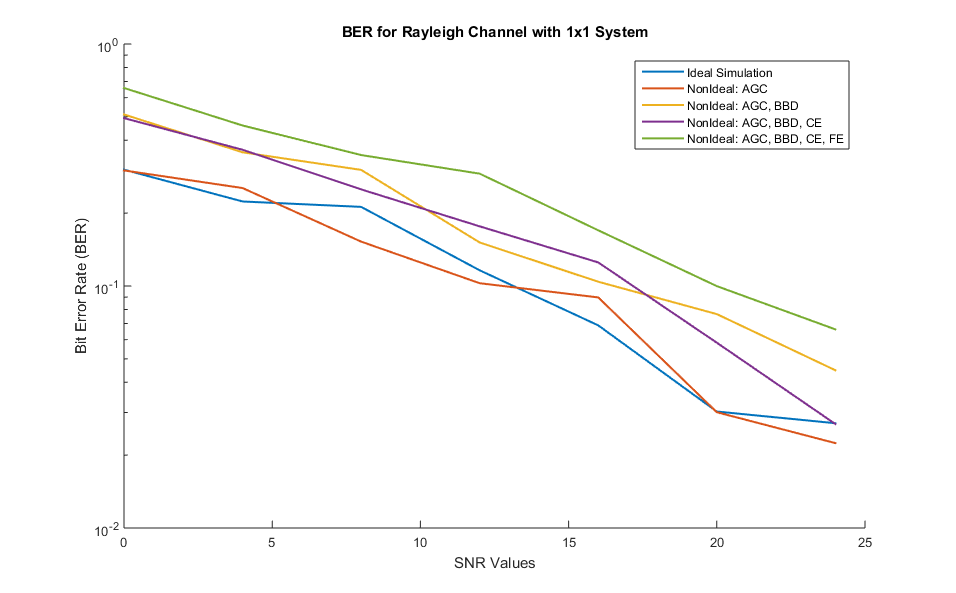
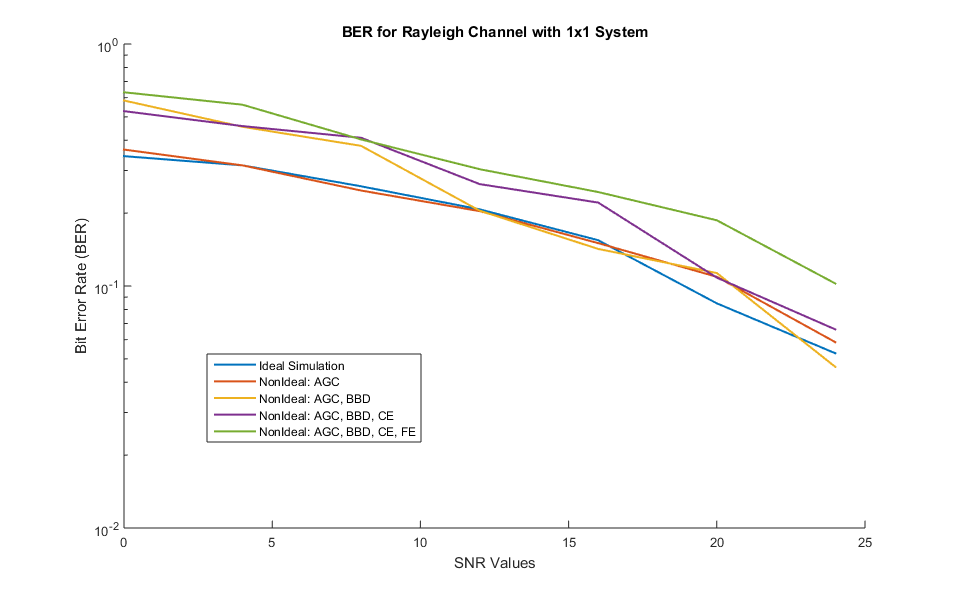


***Figure 6:*** *BER plot of Rayleigh Channel with No Multipath for M = 4*

The first thing we notice is that the BER plot diagnostic of the Rayleigh Channel differs slightly. A non-ideal AGC does not differ much from the ideal, similar to the AWGN case. However, we see a larger difference when non-ideal BBD is included. From the way we have defined ideal BBD and BER, the extremely high error rates, especially ones at lower SNR values, are due to the packet loss. Non-ideal channel estimation and Frequency estimation provide similar impact to BER as AWGN.

One interesting result is that at times the non-ideal AGC does better than ideal. By the way we have defined an ideal AGC, sometimes a fixed adjustment can do worse than an adapting one.

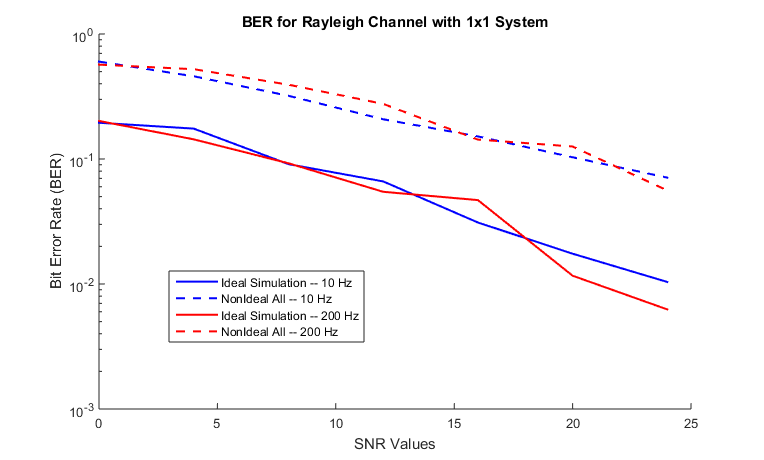
**Rayleigh Channel with No Multipath, M = 16 & 64, Doppler Frequency = 10 Hz**



***Figure 7:*** *BER, Rayleigh, No Multipath, M = 16* ***Figure 8:*** *BER, Rayleigh, No Multipath, M = 64*

Here, in Figures 7 and 8, we demonstrate that the different values of M for MQAM perform similarly to the M = 4 case. We additionally see that higher values of M perform worse in similar reasoning to that of the AWGN channel.

**Rayleigh Channel with No Multipath, M = 4, Doppler Frequency = 10 Hz vs 200 Hz**



***Figure 9:*** *BER plot comparison of Doppler Frequencies of 10 Hz and 200 Hz*

Figure 9 above compares two different types of Rayleigh fading, with other parameters essentially identical. By the definition of Doppler frequency, a higher frequency denotes a faster Rayleigh fading.

We find that the 10 Hz and 200 Hz fading are very similar with ideal receiver components. This makes sense, as perfect estimation nearly negates all effects of fast fading. Additionally, the packet is not of an extremely large size. This is intentional such that Wi-Fi packets are not as effected by fast fading.

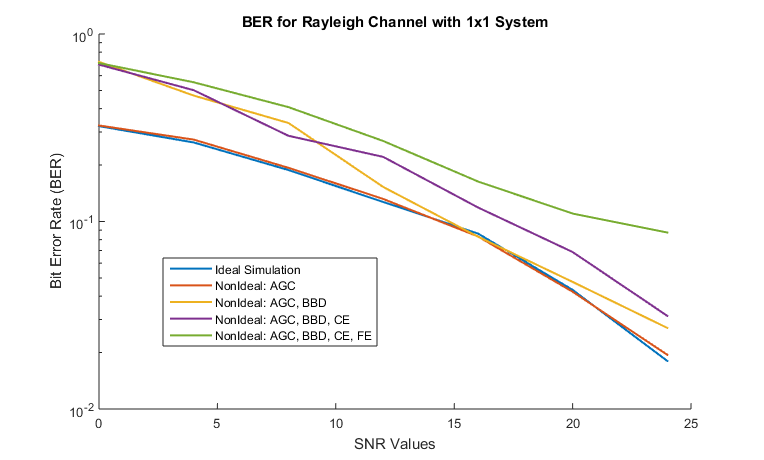
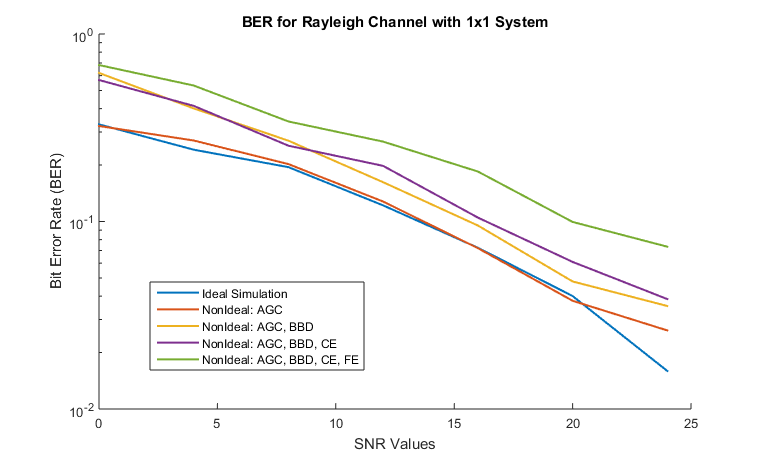
However, with non-ideal components, the 200 Hz BER performs slightly worse than then that of the 10 Hz. Given that Block Boundary Detection, Channel Estimation, and Frequency Estimation are all dependent on the state of the channel, a channel that fades within can throw off many calculations. The multiple stages of frequency detection and adjustment attempt to compensate for this, but it is still not perfect.

**2-Ray Multipath and 4-Ray Multipath**

For the following comparisons, we will solely compare systems using M = 16 for the MQAM and a Doppler frequency of 10 Hz. We deduce that the effects of changing M and Doppler will have similar effects on both systems. Additionally, we will only compare the ideal case and the case with all non-idealities, as the above plots demonstrate the effects of each non-ideal portion.

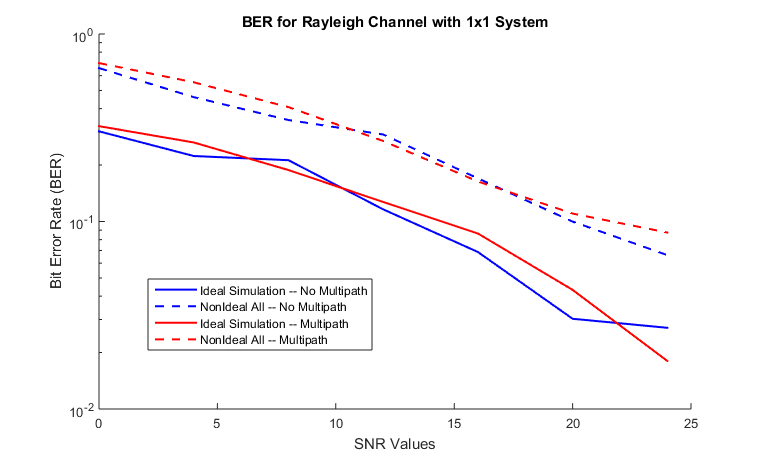
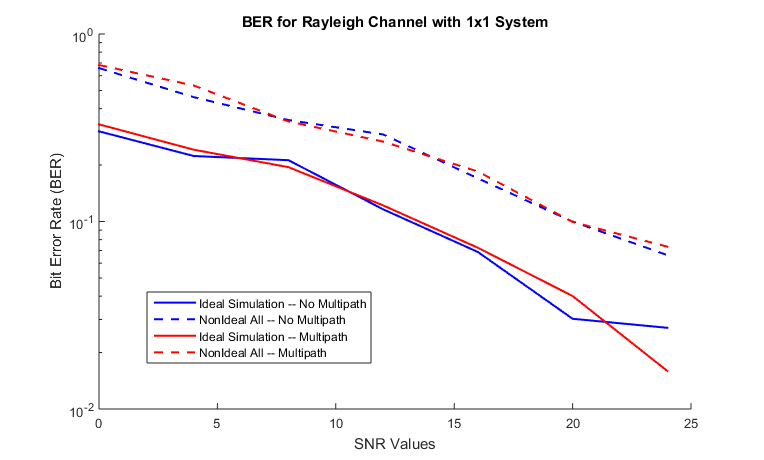
All the above plots did not consider multipath situations. In the following two scenarios, we will consider how multipath affects the receiver’s ability to decode.

Consider a 2-ray model with the multipath components having equal power contributions at 0 ns and 50 ns. Additionally, consider a 4-ray model with the multipath components of relative power of 0, -3, -6, and -10 dB at 0 ns, 70 ns, 150 ns, and 200 ns respectively. Plotted below, the 2-ray model is shown on the left and the 4-ray model is shown on the right:



***Figure 10:*** *BER, Rayleigh, 2-Ray Multipath, M = 16* ***Figure 11:*** *BER, Rayleigh, 4-Ray Multipath, M = 16*

We note that the patterns that occur between the non-idealities are similar to the simulation with no multipath in Figures 10 and 11. We then proceed to compare the results to the Rayleigh Channel with no multipath, again with the 2-ray model on the left and the 4-ray model on the right:

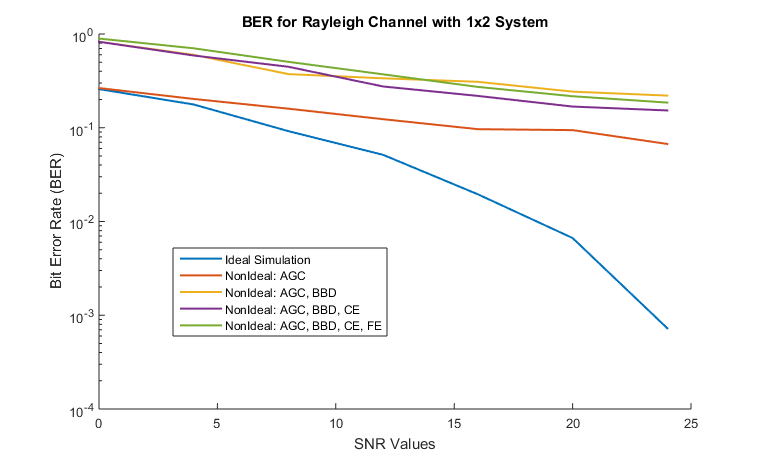


***Figure 12:*** *Compare, 2-Ray Multipath, M = 16* ***Figure 13:*** *Compare, 4-Ray Multipath, M = 16*

Observing Figures 12 and 13 shows that the receiver can handle multipath very well. As the multipath spread increases, it seems that the performance can vary at a larger amount, but it still responds closely to the no multipath result. Thus, we conclude that the receiver works well despite multipath components in a Rayleigh channel.

**1x2 Rayleigh Channel with No Multipath**

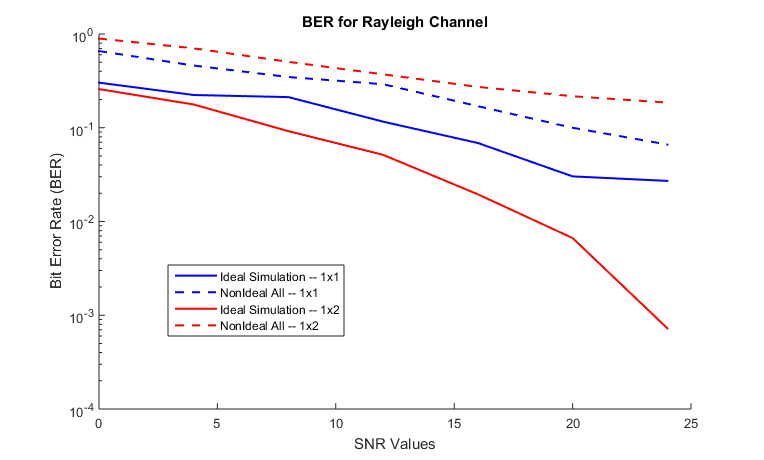
Figure 14 shows a plot of a 1x2 Rayleigh channel with No Multipath at M = 16.



***Figure 14:*** *BER plot of 1x2 System in Rayleigh Channel with No Multipath for M = 16*

With ideal receiver components, we see that the BER by a significant amount, especially at lower SNR values. However, introducing non-ideal components causes a dramatic worsening of the BER especially at larger SNR values. We deduce this to be due to a combination of packet loss complications at the receiver combining and the inability for estimation to work well with instabilities.

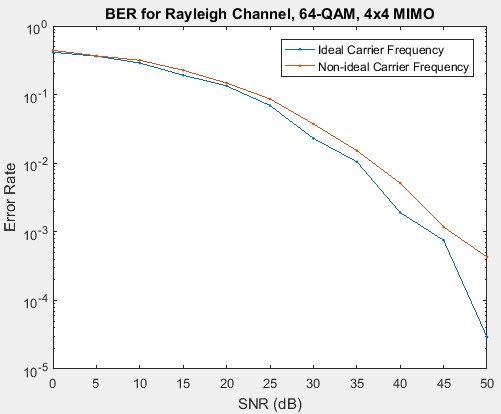
Figure 15 compares a 1x1 with a 1x2 antenna system.



***Figure 15:*** *BER plot comparison of 1x1 and 1x2 antenna systems, M = 16, Doppler 10 Hz*

We see that our assumptions made above were true. Under ideal circumstances, a 1x2 system should outperform a 1x1 system. However, non-ideal conditions worsen the channel enough to make things worse in a 1x2 system. Given a robust packet detection algorithm, perhaps the system could be improved.

**MIMO 4x4 Rayleigh Channel with No Multipath**



***Figure 16:*** *BER of a 64-QAM 4x4 system in Rayleigh with Doppler 10 Hz*

Additionally, to test the limits of the the 802.11n standard, we simulated a 4x4 MIMO system with 64-QAM to push the maximum data rate possible through the channel. To simplify simulation, we used an ideal AGC, ideal BBD, and the non-ideal channel estimator. The simulation was run with and without ideal carrier frequency estimation, with the results shown in Figure 16. This set-up requires much higher SNR than the other studied systems, but it offers the highest possible data rates offered by this 802.11n standard.

**6. DISCUSSION**

For all tests, we used a payload of 0.25 kB. However, there was a variety of modulation schemes and each of them have different numbers of bits/symbol. Thus the packet lengths were not the same when we used different modulation schemes. Longer packets are more affected by Rayleigh fading as fade changes with time and we did not implement dynamic channel estimation. Thus, a truer “apples to apples” comparison would have our simulations modulate with the same number of symbols and thus the same amount of time.

We recognize that due to our definition of a perfect AGC, we may experience scenarios where a non-ideal AGC will perform better than the perfect one. A more robust implementation would perhaps consider an ideal AGC that considers all of the inputs and adjusts them sample by sample.

It was seen that the carrier offset estimation is a dominating non-ideality in the simulated BER. One method of reducing sensitivity to carrier offset is windowing. This is in the 802.11n standard, and could be implemented in our simulation to reduce inter-carrier interference.

Aside from debugging code, one of the most difficult challenges in completing this project was integrating all the receiver components together. Given the structure of a receiver, there was ample modularity such that both contributors were able to create functions separately. However, when putting the receiver together, many modules were actually found incomplete due to the specific format needed to perform in the 802.11n receiver correctly. When creating a component, it is pertinent that the designer recognizes the scenario at which the said component will be running in. Accounting for all cases will produce the best results.

**7. CONCLUSION**

In this project, we have successfully implemented and simulated an end to end 802.11n system including various non-idealities in the receiver components as well as up to 4x4 MIMO. Our analysis has highlighted the dominant sources of non-idealities in our implementation, namely primarily the frequency estimation/correction, and also the block boundary detection. The performance has been demonstrated and compared across various variables including various data subcarrier modulations, and various noise environments. This report highlights the implementation considerations and tradeoffs used to examine the design space and each section details various possible improvements or possible areas for exploration.