Name: Justin Ngo
PSU ID: jvn5439
Professor: Mark Mahon
Class: CompEn 462
Date: 03.01.2025
Project : 1

Abstract:

This report presents the implementation and analysis of an Orthogonal Frequency Division Multiplexing (OFDM) system and covers several key components:

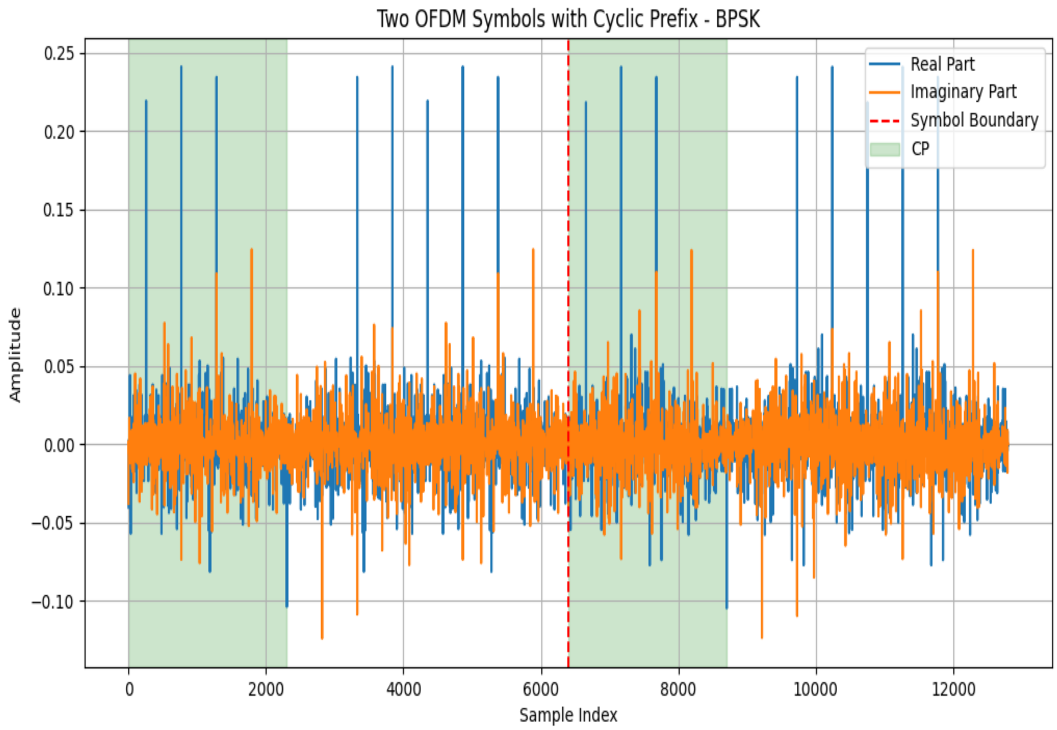
1. **Modulation Schemes**: Different modulation schemes such as BPSK, $\pi/2$-BPSK, QPSK, and 64-QAM are implemented in the `Modulators.py` file. These schemes are used to convert the input bit stream based off the 8-bit ASCII conversion of the string ”WirelessCommunicationSystemsandSecurityJustinNgo” into suitable symbols ready for transmission.

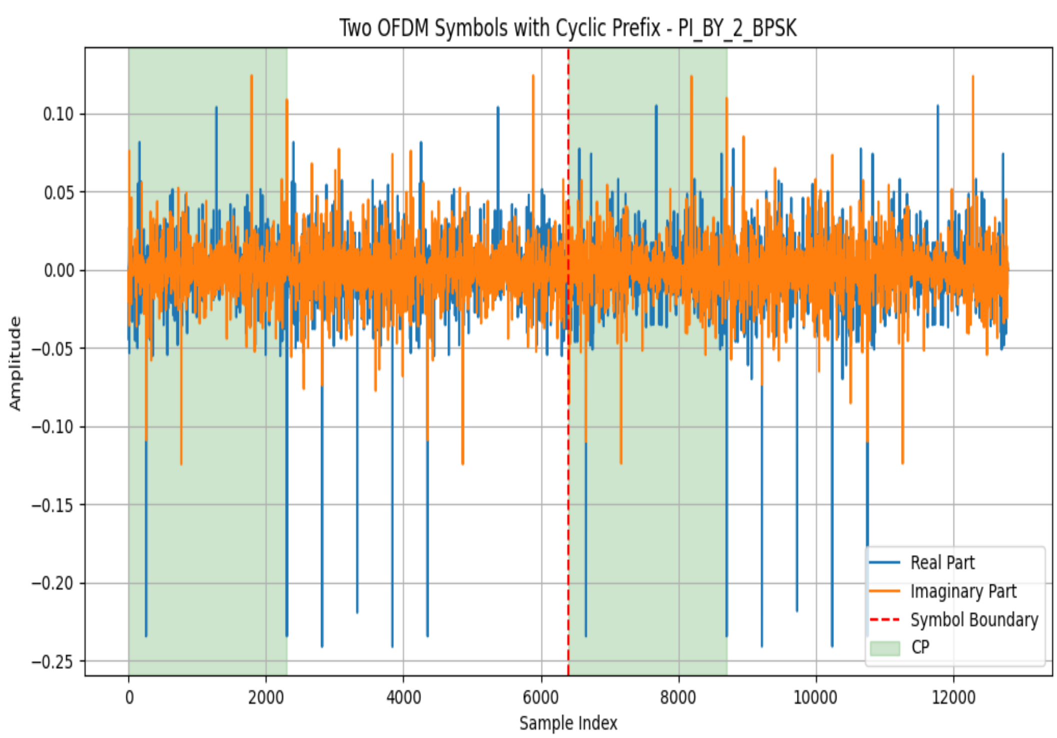
2. **OFDM Processing**: The `OFDM.py` file houses all the relevant functions used to produce an OFDM output, and handles the bulk of the processing. The OFDM function is responsible for running the serial-to-parallel conversion, Inverse Fast Fourier Transform (IFFT), cyclic prefix insertion, and also graphs the output.

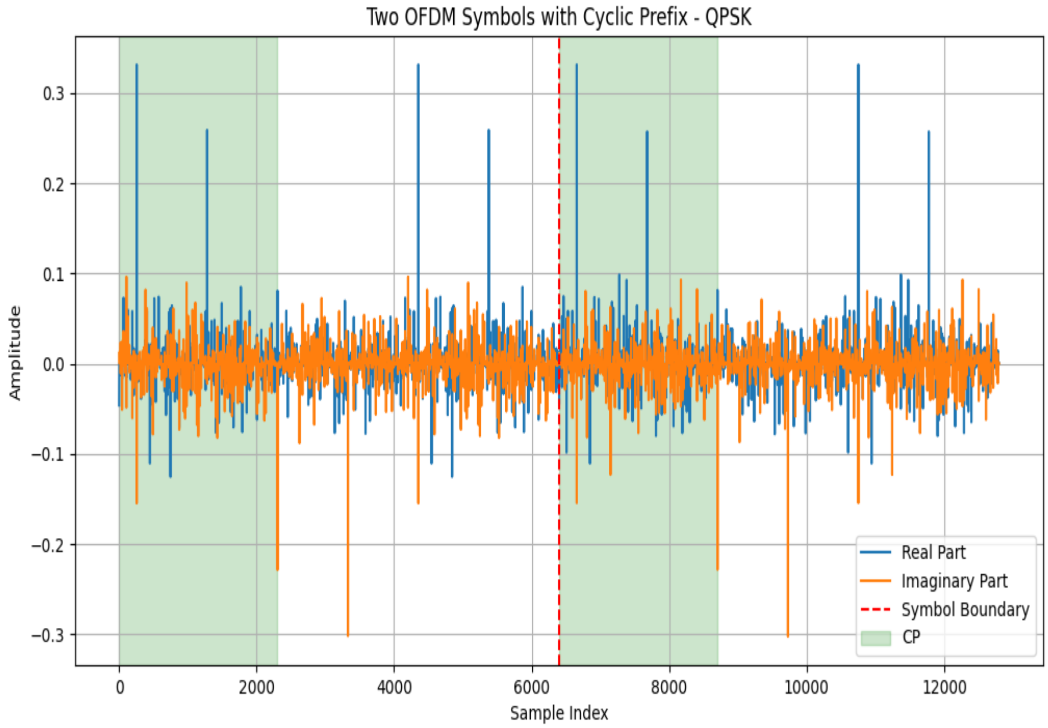
3. **Main Execution**: The `main.py` file generates the bit stream used for processing from the aforementioned phrase and is where the OFDM function is called to run the system for each modulation scheme.

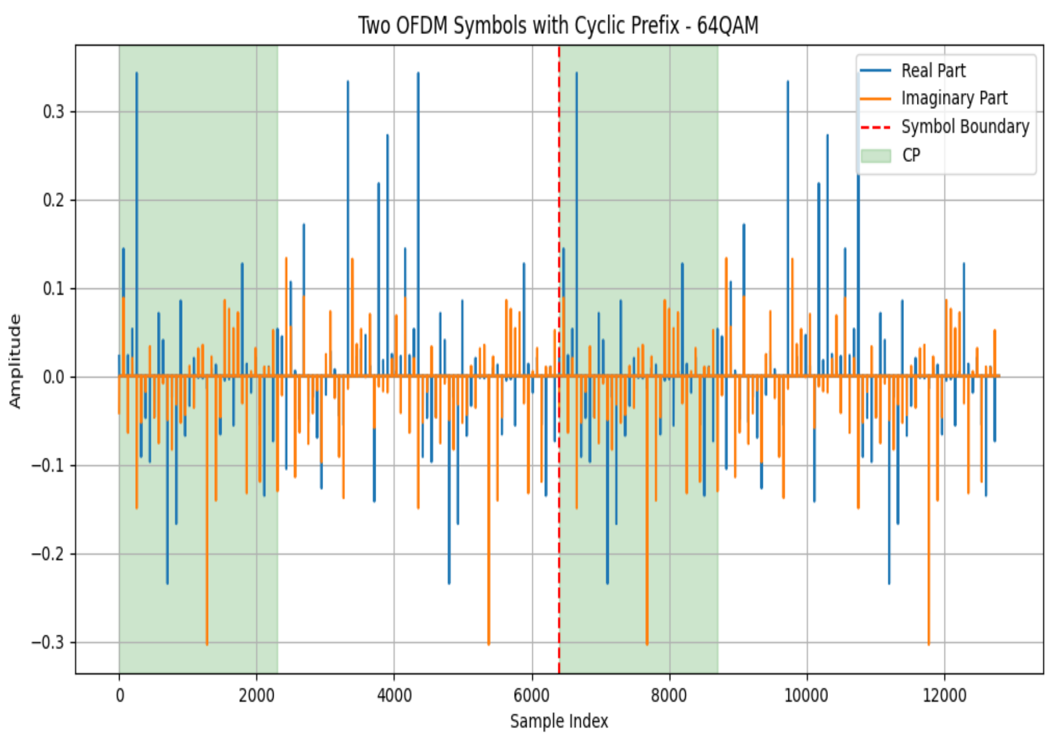
4. **Visualization**: A sample of 2 symbols generated from the OFDM system is plotted for visualization and highlights the cyclic prefix and symbol boundaries.

Generated Waveforms:






Two OFDM Symbols with Cyclic Prefix - QPSK
Amplitude
Sample Index
Real Part
Imaginary Part
Symbol Boundary
CP
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
0
2000
4000
6000
8000
10000
12000


Two OFDM Symbols with Cyclic Prefix - 64QAM
Amplitude
Sample Index
Real Part
Imaginary Part
Symbol Boundary
CP
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
0
2000
4000
6000
8000
10000
12000

Flow Diagram:

1. **Bit Stream Generation**: The input bit stream is generated from the ASCII conversion of the phrase "WirelessCommunicationSystemsandSecurityJustinNgo" and is repeated to ensure a minimum length of 1 slot using the `64QAM` modulation scheme, since that requires the most bits per symbol.

2. **Modulation**: Depending on the modulation scheme passed into the `OFDM` function, the input bits are mapped to a constellation based off of figure 5.2 from the book, that was provided in the lecture slides. The formulas I used were approximations of what were provided on the 3GPP TS 38.211 pdf.

3. **Serial to Parallel Conversion**: After modulation, the bits are passed into the `s2p` function which converts the serial input into a parallel output before running the IFFT

4. **IFFT**: The `IFFT` is run immediately after the bits are reorganized within the `s2p` function, and is the final output from `s2p`

5. **Cyclic Prefix Insertion**: The 2d array that's returned as the IFFT from `s2p` is then passed into `cyc_pref`, where each symbol is prepended with the last $2034T_c$ of it's output

6. **Steps That Weren't Modeled**:

   (a) **Parallel to Serial Conversion**: The parallel output of the Prefix Insertions would then be flattened into a serialized output for processing at the DAC

   (b) **DAC**: The DAC would convert the digital signal into an analog signal for transmission

   (c) **RF Modulation**: The analog signal would then be modulated with the carrier frequency to boost the signal before being broadcasted by the antenna

Technical Problems:

1. **Modulation Schemes**: Had a bit of an issue understanding the math behind the modulation schemes and figuring out how to implement them accurately , but wasn't too bad afterwards

2. **Cyclic Prefix**: I made an error when I first coded the cyclic prefix insertion, and prematurely flattened the output from the IFFT which caused the code to throw an fault and say that the array was inaccessable. Fixed the issue after going over the diagrams again, and only flattened the array after performing the cyclic prefix insertion