System Modelling Lab
(EC_670) Lab Report

2nd Semester

Submitted by:
Gokaran Dev Gautam
2402039
Mtech VLSI & Embedded Systems

Department of Electronics and Communication Engineering,
Indian Institute of Information Technology Guwahati
Bongora, Assam, 781015

INTRODUCTION

In traditional wireless communication, sending high-speed data over a single carrier faced serious problems like *inter-symbol interference (ISI) caused by multipath fading*, where signals reflect off objects and arrive at different times.

As data rates increased, channels became more prone to distortion, making reliable transmission difficult. OFDM (Orthogonal Frequency Division Multiplexing) was developed to solve these problems.

It works by splitting the high-speed data stream into multiple parallel lower-rate streams, each transmitted over a separate, closely spaced subcarrier. These subcarriers are made mathematically orthogonal, meaning they do not interfere with each other even though they overlap in frequency.

This design greatly reduces ISI and improves resistance to fading and noise.

Another reason for developing OFDM was to send more data without needing a bigger frequency band. By splitting the data into many smaller streams and sending them close together, OFDM uses the available bandwidth more efficiently. This way, we can transmit a lot of information through a single channel without wasting space.

OFDM has become the foundation of many modern systems like Wi-Fi (802.11 standards), 4G LTE, 5G, and digital television.

Comparison of IEEE 802.11 Standards

Standard	Release Year	Frequency Band	Max Data Rate	Modulation	Channel Bandwidth
802.11a	1999	5 GHz	54 Mbps	OFDM	20 MHz
802.11g	2003	2.4 GHz	54 Mbps	OFDM	20 MHz
802.11n	2009	2.4 GHz and 5 GHz	600 Mbps	OFDM, MIMO	20/40 MHz
802.11ac	2013	5 GHz	6.93 Gbps	OFDM, 256-QAM	20/40/80/160 MHz
802.11ax	2019	2.4 GHz, 5 GHz, (6 GHz Wi-Fi 6E)	9.6 Gbps	OFDMA, 1024-QAM	20/40/80/160 MHz

Table 1: IEEE 802.11 comparison table

Objectives

1. **Model an OFDM System in MATLAB**
   Design and simulate an OFDM system in MATLAB, including key components such as modulation (64-QAM), cyclic prefix, and pilot symbol insertion,etc

2. **Generate HDL-Compatible Code Using MATLAB**
   Use MATLAB to generate HDL-compatible code for the OFDM system to ensure that the model is suitable for hardware implementation.

3. **Implement the System on an FPGA Board**
   Implement the generated HDL code on an FPGA board, ensuring the OFDM system operates as expected in a hardware environment.

OFDM Parameters for this project

Parameter	Value	Description
numOFDMSymbols	10	Number of OFDM symbols per frame.
fftSize	64	FFT size for the OFDM modulation.
modOrder	64	For 64-QAM modulation order.
bitsPerSymbol	log2(modOrder) = 6	Bits per symbol for 64-QAM.
dataSubcarriers	(fftSize / 2) - 1 = 31	Data subcarriers per OFDM symbol.
numBits	numOFDMSymbols * dataSubcarriers * bitsPerSymbol = 1860 bits	Total bits per frame.
CP_Length	fftSize / 4 = 16	Cyclic Prefix length.
pilotSymbol	1 + 0j	Pilot symbol for channel estimation.

Table 2: Paramters for my project

These parameters were used in the implementation of the OFDM project, which models an 802.11ac-like OFDM system.

METHODOLOGY

Software used is MATLAB 2020b

The methodology for this project involves the design, simulation, and testing of an OFDM system using MATLAB. The key objective is to model the system, generate HAL-compatible code, and eventually implement it on an FPGA board

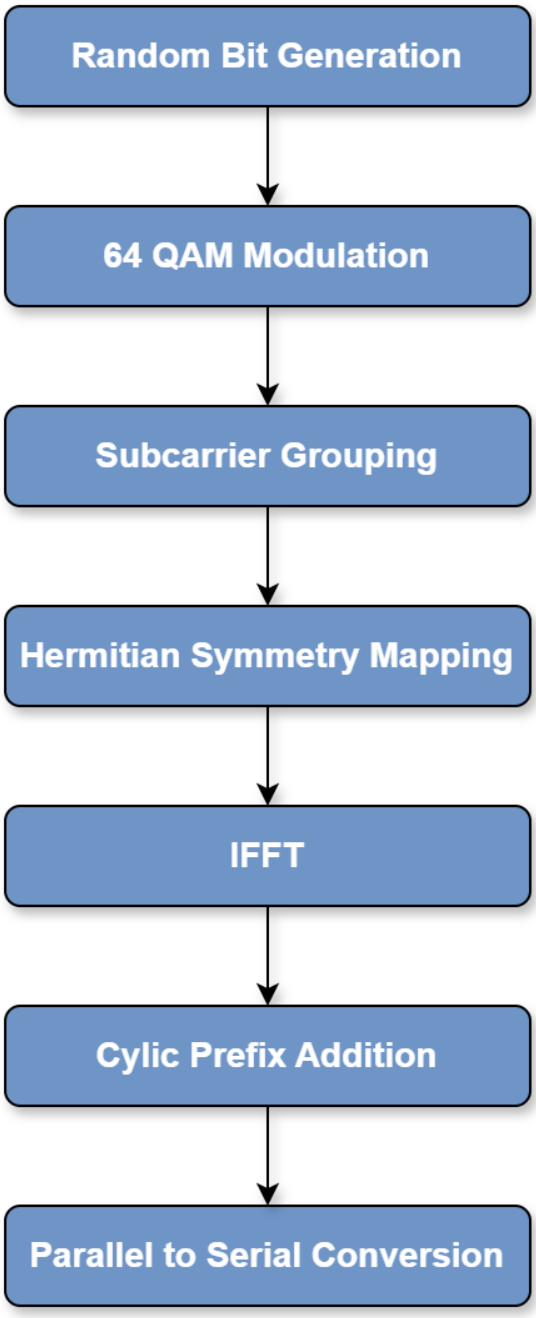


Figure 1: flowchart for Transmitter deisgn

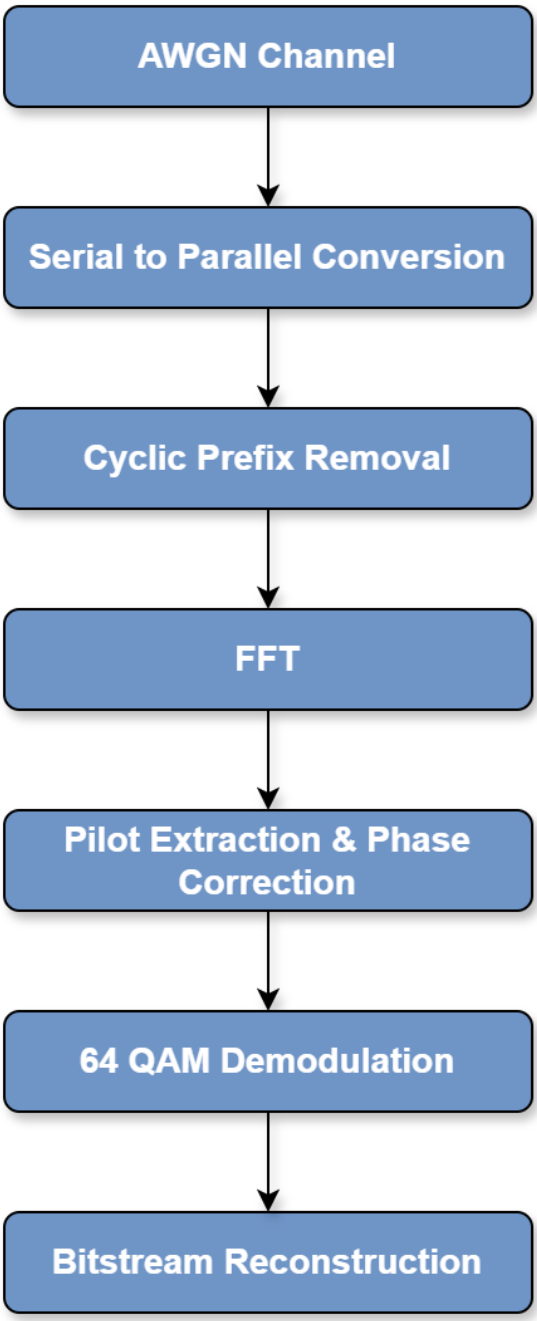


Figure 2: flowchart for Receiver design

The OFDM system was implemented using separate functions stored in *OFDM_Transmitter* and *OFDM_Receiver* folders. A main OFDM Module script called these functions to handle different stages of transmission and reception.

Flags were added to each stage, allowing users to enable or disable specific plots for easier debugging and visualization.

TRANSMITTER

1. **Random Bits Generation:**
   A total of 1860 random bits are generated. These bits represent the raw information we want to transmit through the OFDM system.

2. **64QAM Modulation:**
   These 1860 bits are grouped into sets of 6 bits each (since 64-QAM uses 6 bits per symbol).
   As a result, 1860÷6=310 QAM symbols are created.
   Each group of 6 bits is mapped onto a specific point in the 64-QAM constellation.

3. **Subcarrier Grouping:**
   The 310 QAM symbols are arranged onto the available data subcarriers.
   Since we have 31 data subcarriers per OFDM symbol, multiple 10-OFDM symbols are created. Pilot tones and guard subcarriers are also considered to ensure proper spectral shaping.

4. **Hermitian Symmetry Mapping:**
   To make the IFFT output real, Hermitian symmetry is applied.
   This means the second half of the subcarriers is a mirror (complex conjugate) of the first half, ensuring the time-domain signal is real-valued, as needed for physical transmission.

5. **IFFT Operation:**
   An IFFT is performed on the arranged subcarrier data.
   This transforms the 64-point frequency domain OFDM symbol into a 64-point time domain signal, ready to be transmitted.

6. **Cyclic Prefix Addition:**
   To combat ISI (Inter-Symbol Interference), a cyclic prefix of 16 samples (since CP length = 64/4 = 16) is added to each 64-sample OFDM symbol. This protects the signal in multipath fading environments.

7. **Parallel to Serial Conversion:**
   Finally, the matrix of OFDM symbols (each with a cyclic prefix) is converted into a single serial stream.
   This serial data stream is what will actually be sent over the channel.

RECEIVER

1. **Channel AWGN Addition:**
   The transmitted serial stream passes through a simulated AWGN channel.

2. **Serial to Parallel Conversion:**
   The received serial stream is reshaped back into parallel frames, where each frame contains one full OFDM symbol (64 + 16 = 80 samples, including cyclic prefix).

3. **Cyclic Prefix Removal:**
   The 16-sample cyclic prefix is removed from each frame, restoring the original 64-point time-domain OFDM symbol.

4. **FFT Operation:**
   A 64-point FFT is performed on each time-domain OFDM symbol to move the data back into the frequency domain.
   Now, the subcarriers contain the transmitted QAM symbols (distorted by the channel and noise).

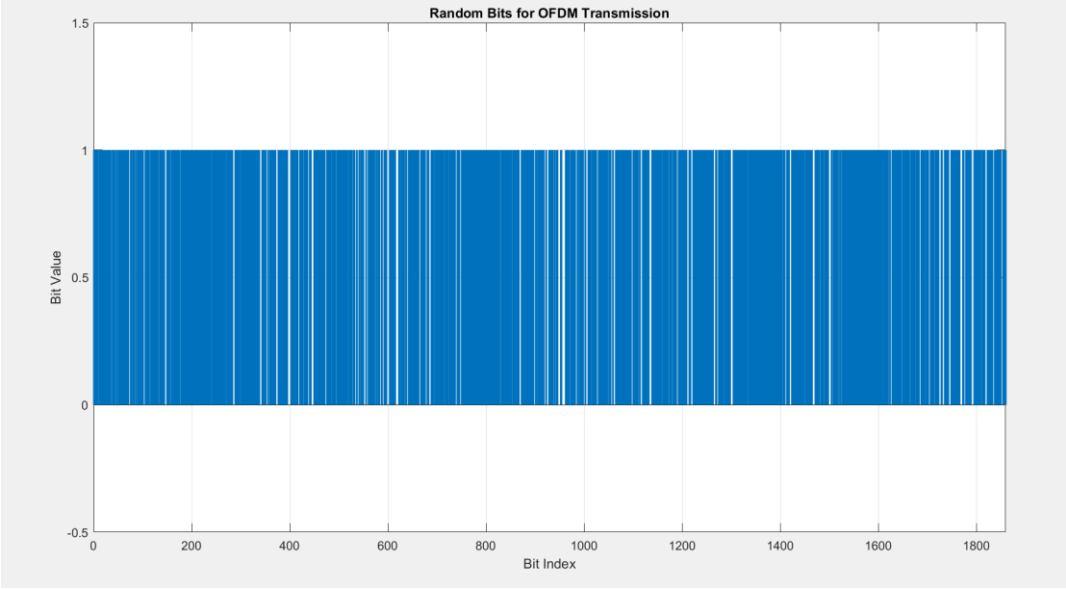
5. **Pilot Extraction and Phase Correction:**
   Pilot subcarriers are extracted and used to estimate the phase rotation caused by the channel.
   Using these pilots, phase correction is applied to all the data subcarriers, aligning them correctly in the constellation.

6. **64QAM Demodulation:**
   The corrected frequency-domain symbols are demodulated using 64-QAM demapping. Each received QAM symbol is mapped back into the corresponding 6 bits.

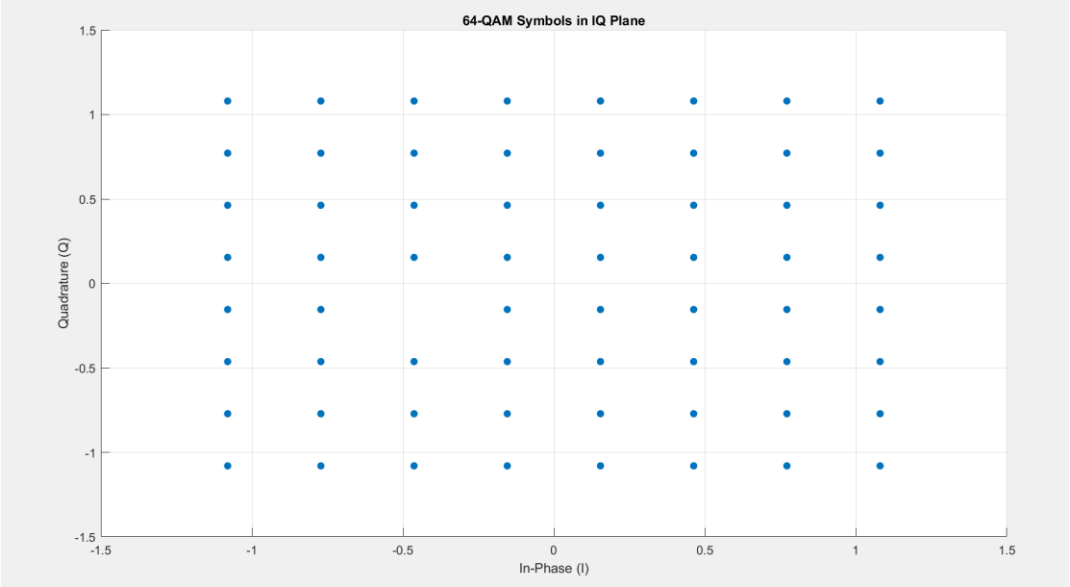
7. **Bitstream Reconstruction:**
   All demodulated bits are reassembled to form the full recovered bitstream.

8. **BER Calculation:**
   The Bit Error Rate (BER) is calculated by comparing the received bits to the original bits.

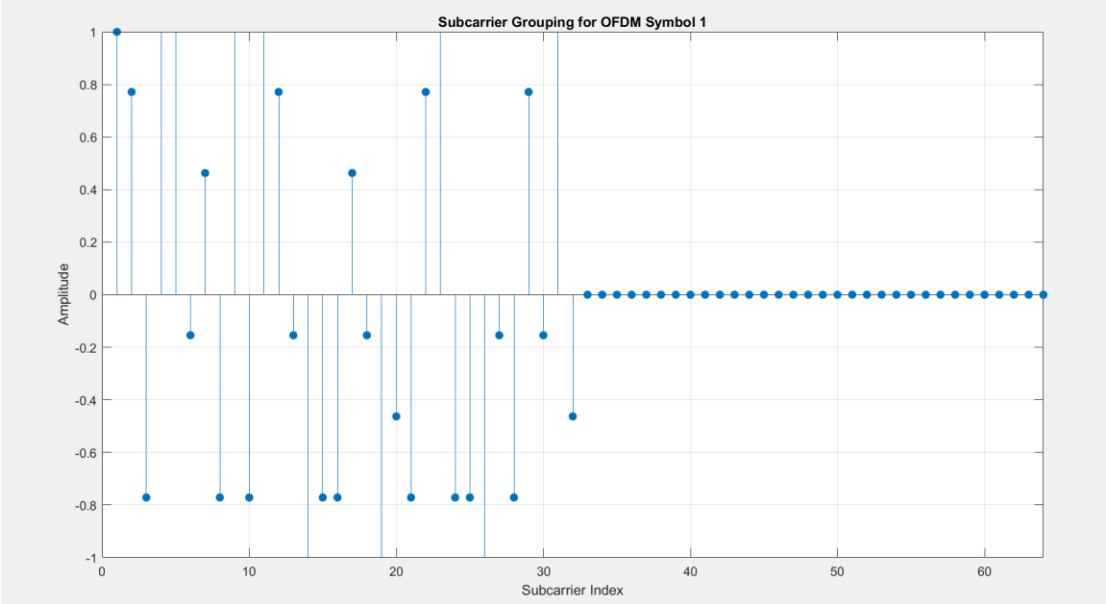
SIMULATION & RESULTS



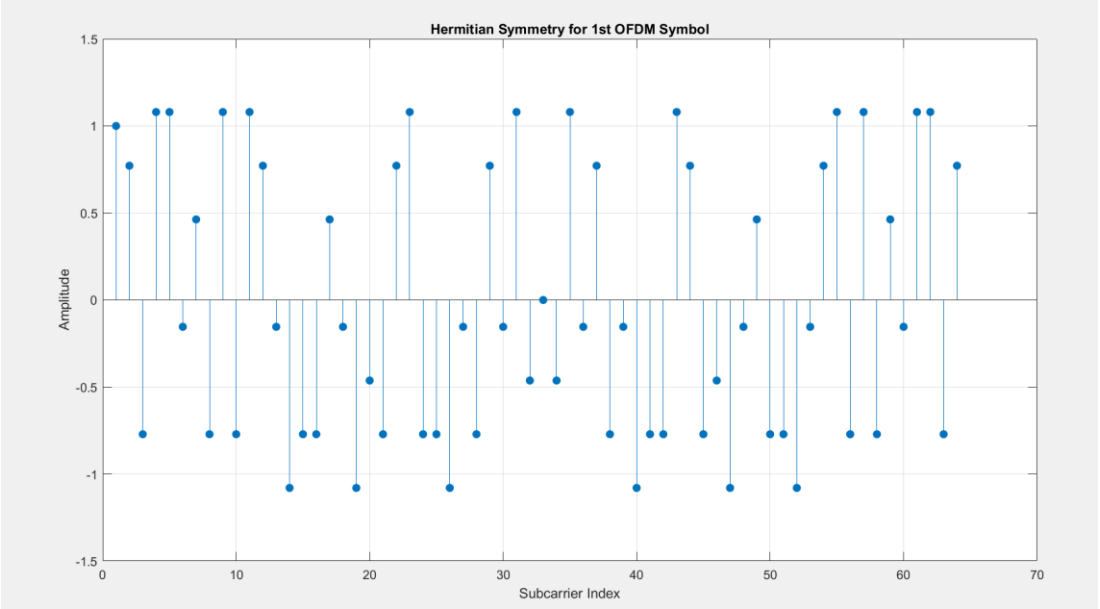
1806 bits data, randomly generated using the zeroes and randi function.

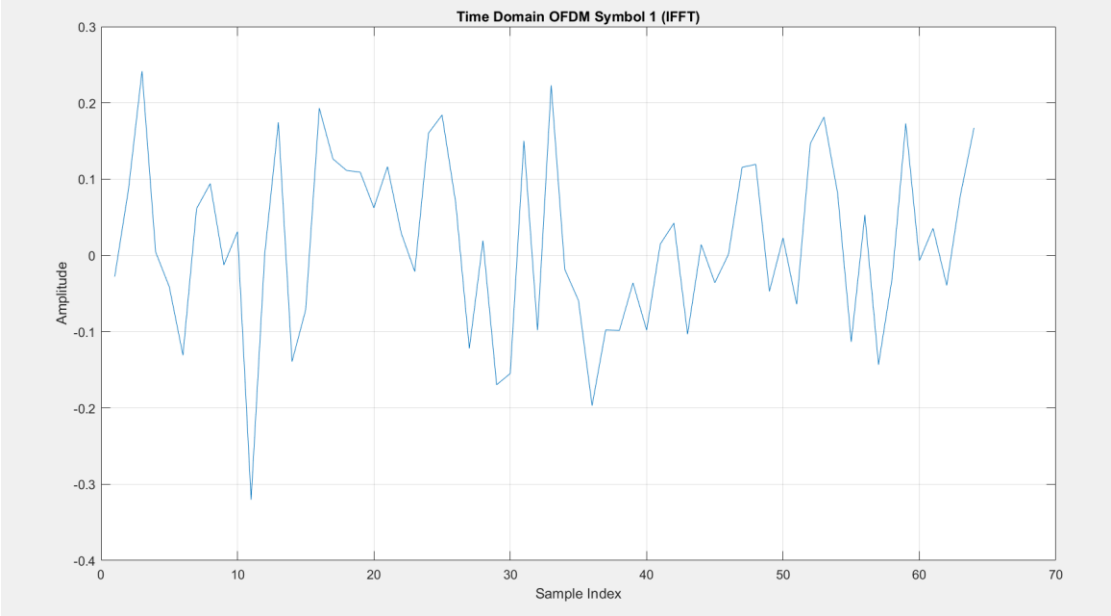


The random data generated earlier was modulated using 64 QAM, which formed 310 QAM symbols.

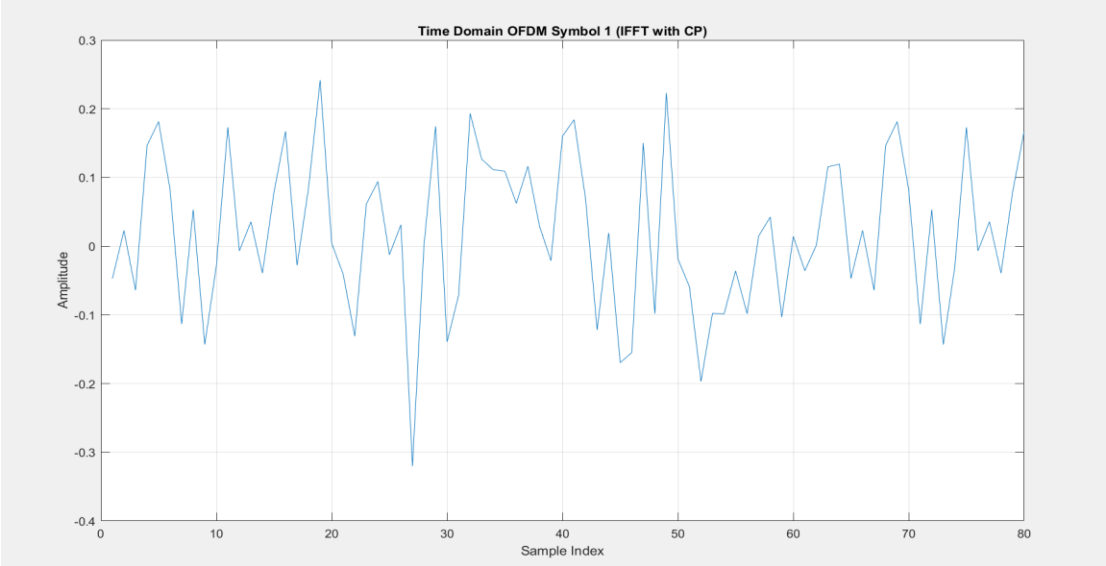


We have taken 31 subchannels for transmission of 310 QAM symbols. Each takes 64 bits with it. First 32 are the actual data, later 32 are zeros. These zeroes are used as guard rails.

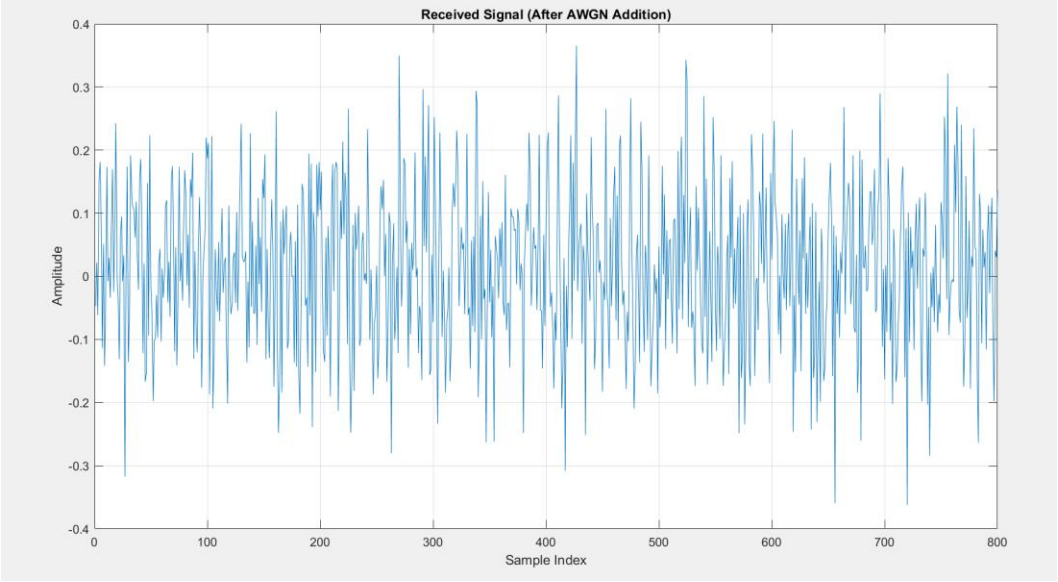




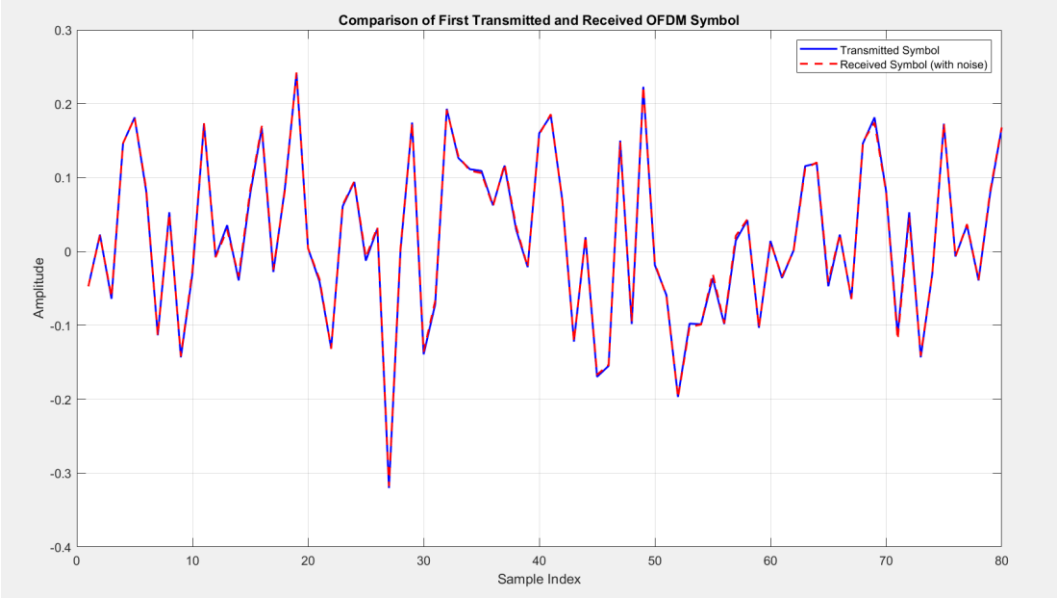
After using hermitian code, IFFT was perfomed.



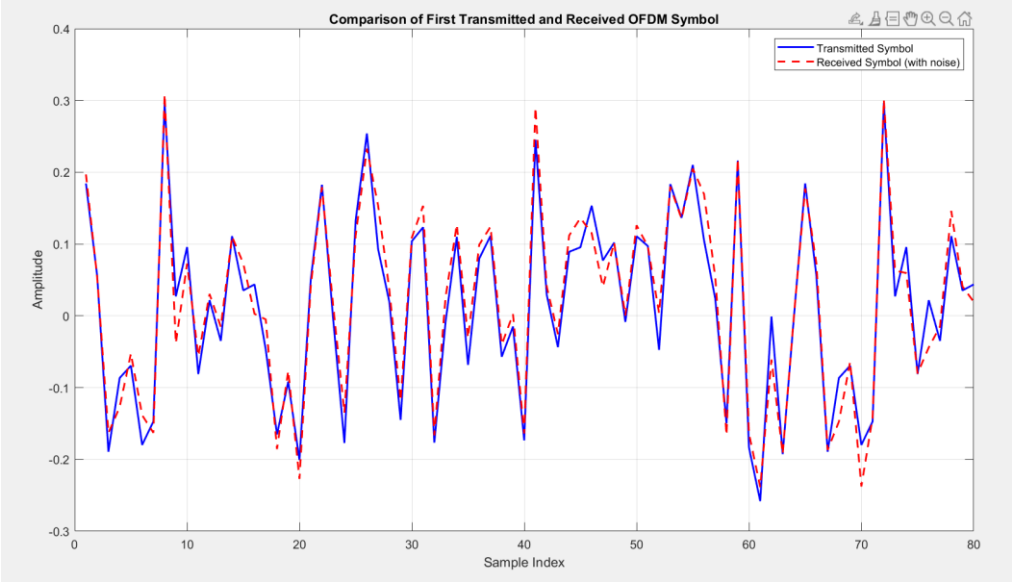
Cyclic Prefix was added to prevent ISI in channel.



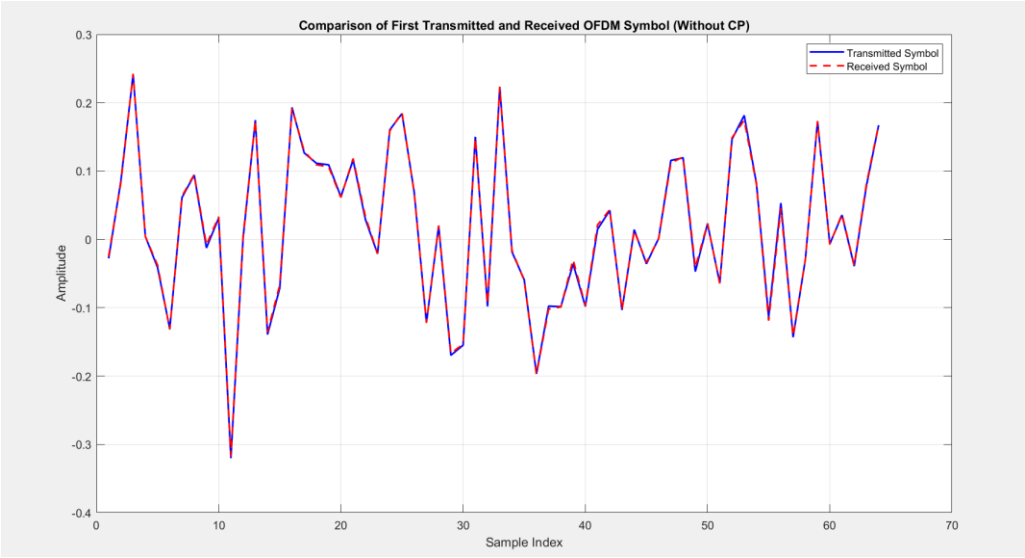
Signal after being passed through the channel.



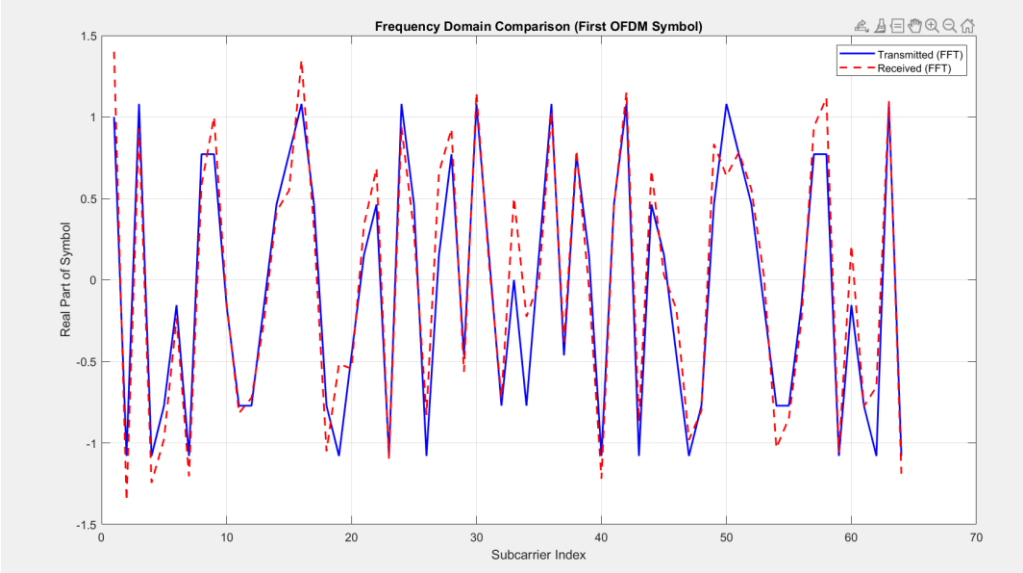
Comparison of First Transmitter and Received OFDM symbol at 40 SNDR



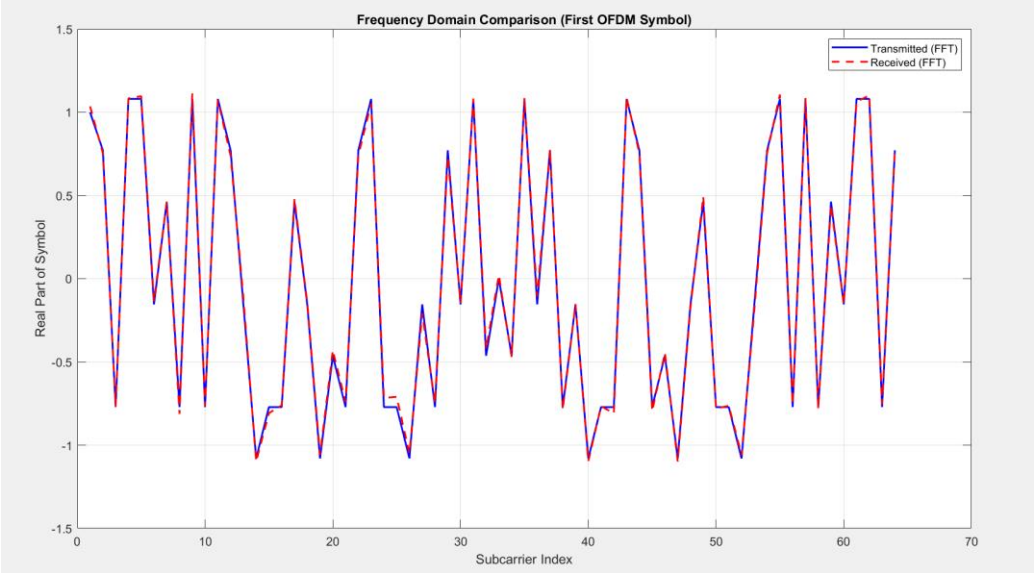
Comparison of first transmitted and received signal at 10 SNDR



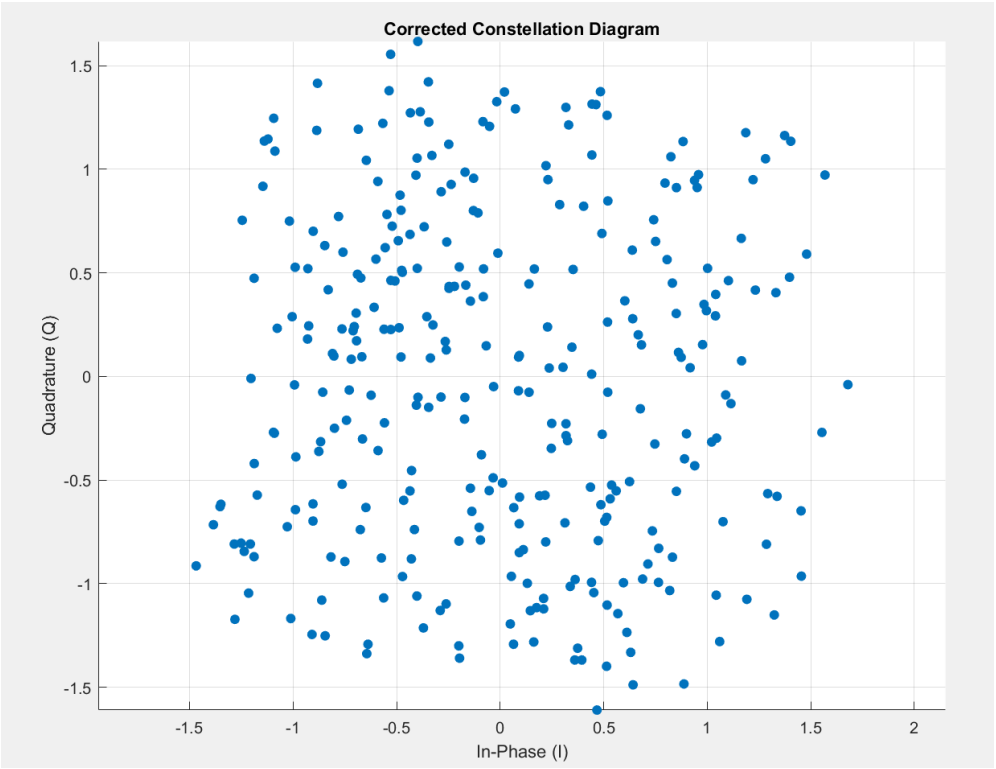
Comparison of First and Transmitted OFDM Symbol without CP.



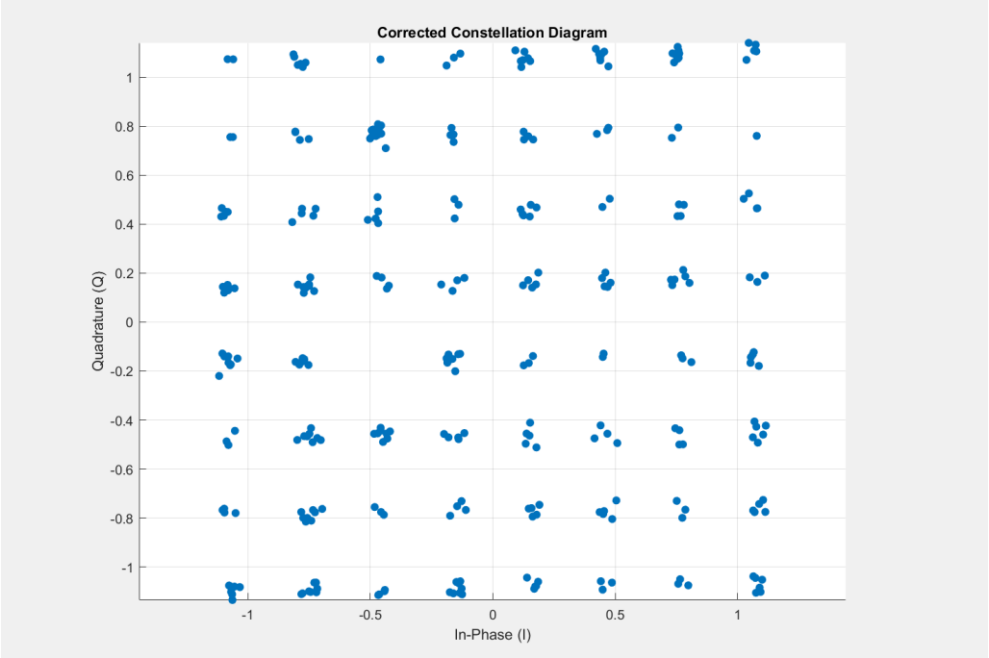
First OFDM symbol comparison at 10 SNDR , after doing FFT



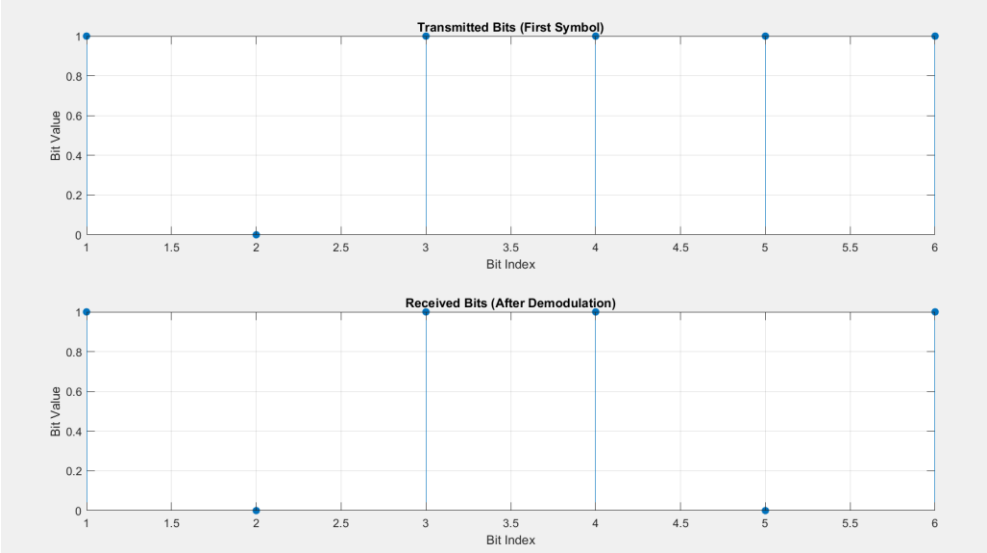
After doing FFT to get the original Time domain signal, 40 SNDR



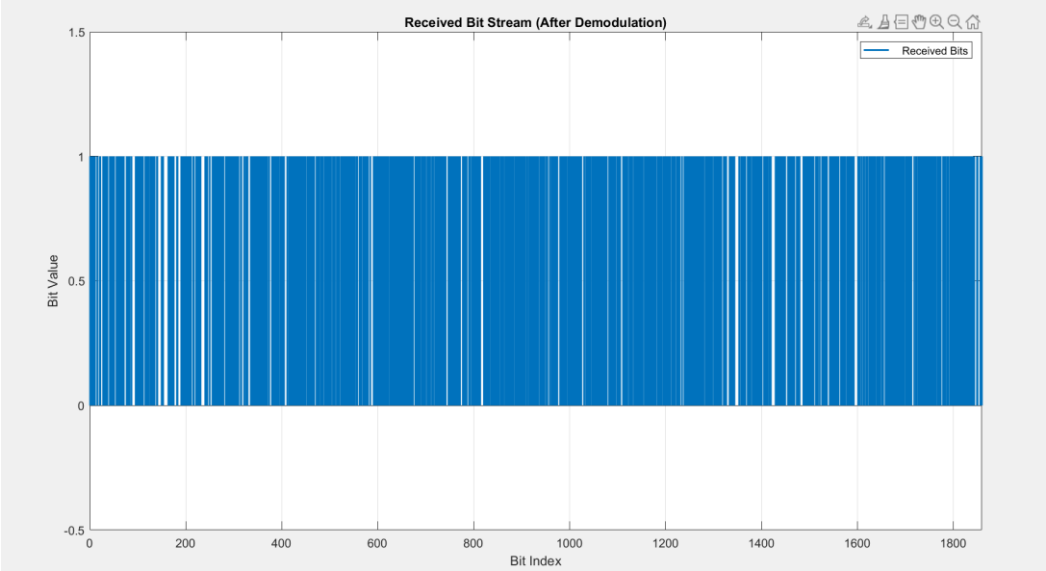
Constellation of received QAM Symbols at 10 SNDR



Constellation diagram of QAM Symbols before they were demodulated.

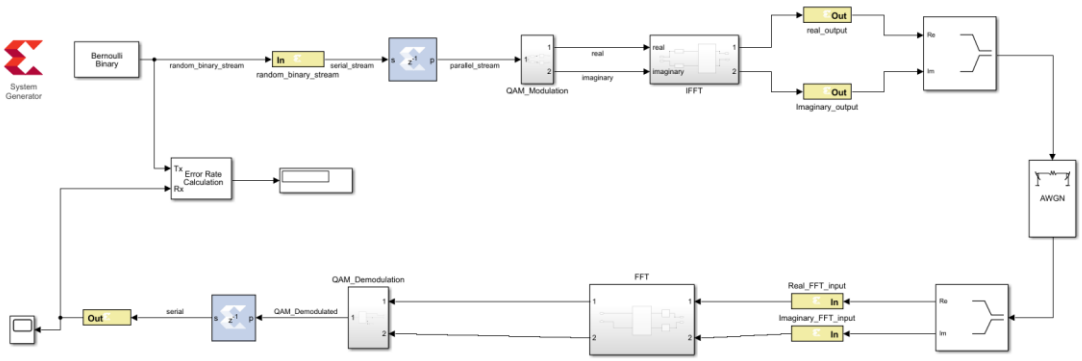


Reconstruction of Data.

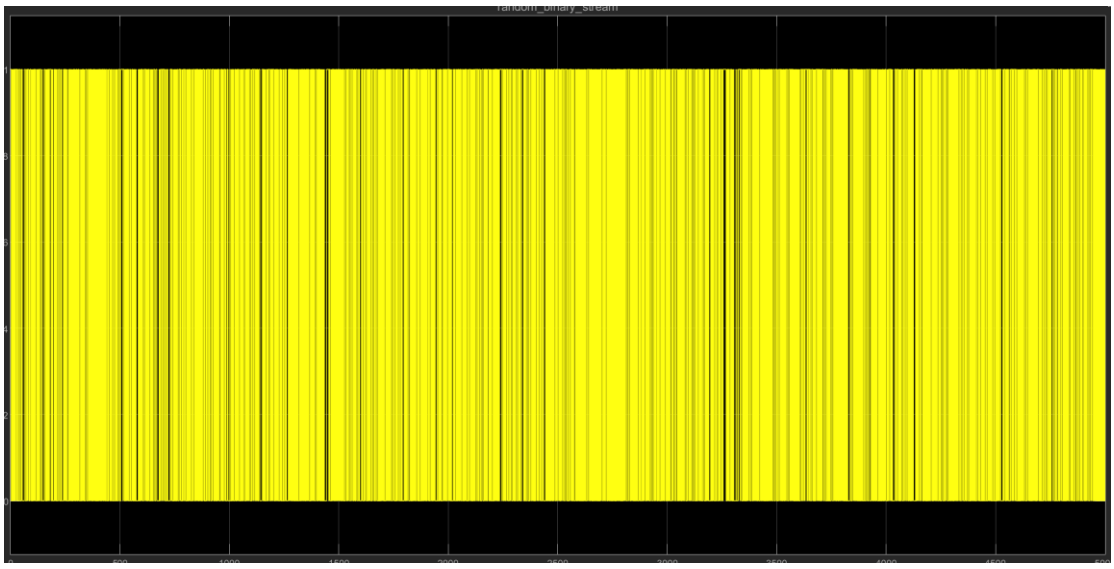


Final received bitstream in its entirety.

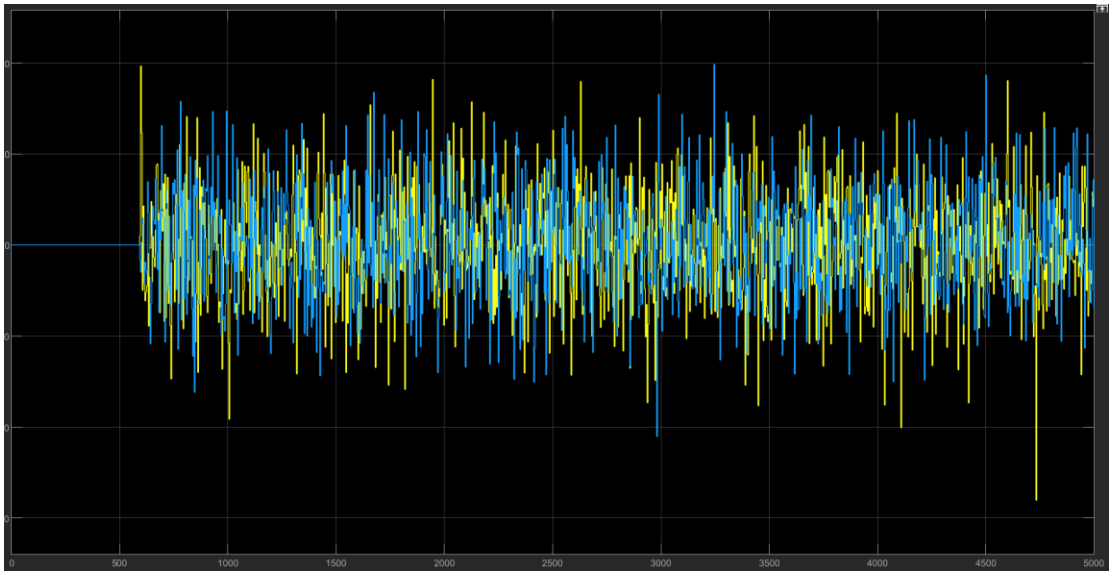
For my second objective, I have implemented a 16QAM system using Vitis Model Composer of Vivado.



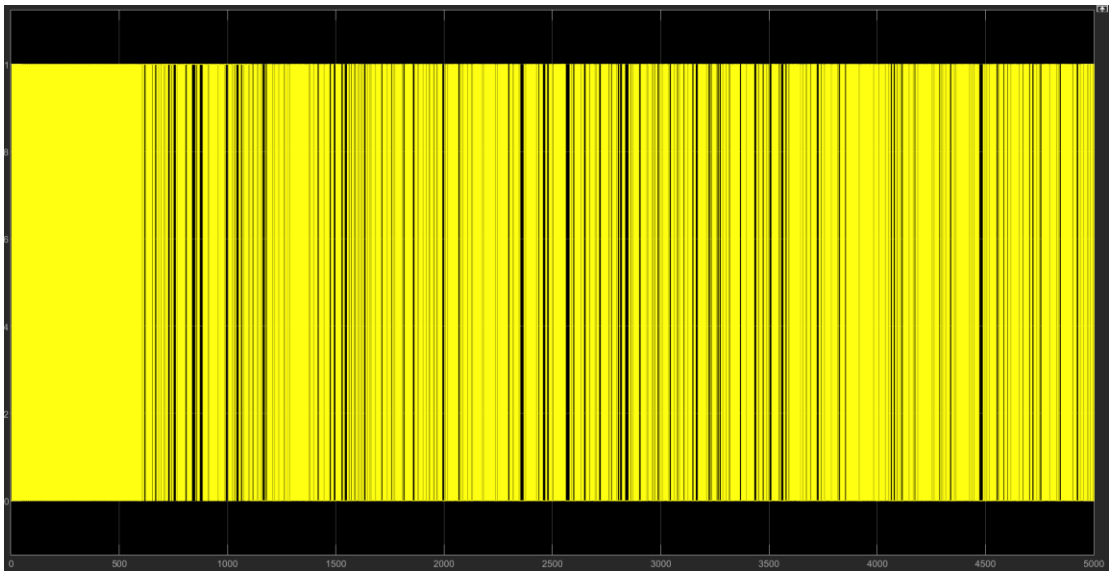
This is the entire OFDM system.



Transmitted bitstream.



This is the OFDM Modulated signal.



Received bistream.

For model composer, I made my own 16 QAM Demodulation module. HDL Code succesfully generated.

CONCLUSION

In this project, a complete OFDM system was developed and simulated using MATLAB. A 64-QAM modulation scheme was used in the MATLAB model to verify the transmitter and receiver operation. HDL code was generated for a simpler 16-QAM system to facilitate FPGA implementation. For real-world deployment, implementing the OFDM system on an FPGA board requires setting up appropriate communication protocols, either analog or digital. If a digital protocol is used, integration with XADC (for analog-to-digital conversion) and DAC (for digital-to-analog conversion) modules becomes necessary. Moving forward, further work will focus on learning these communication interfaces and completing the hardware implementation.