16 QAM Communications Waveform

Emerson Kim, Kevin Sadykhov
University of Washington – Department of Electrical Engineering

Introduction

The goal of this project is to develop a 16 QAM communications waveform to work on an X310 Universal Software Radio Peripheral (USRP) device. 16 QAM is a form of digital modulation. Modulation is the process through which information is added to a periodic carrier signal to be transmitted over a telecommunications medium. Information can be added to a periodic waveform by altering its properties such as the amplitude, frequency and phase of the waveform. QAM modifies the amplitude and phase of the carrier signal. The number 16 represents the total number of symbols (groups of bits) that can be transmitted. For 16 QAM, each symbol represents 4 bits. A common way to represent the symbols that may be transmitted in a given modulation scheme is a constellation diagram. A constellation diagram is a representation of a modulated signal that displays all possible symbols that may be selected. For this project, we relied on the use of constellation diagrams in order to summarize some of our results.

Python Simulation

- Initially used Python for prototyping the waveform
- Implemented a QAM modulator and demodulator
- 16, 4-bit binary values from 0000 to 1111
- Additive White Gaussian Noise (AWGN) was included to show the effects of transmitting data over a medium
- Probability of symbol error was calculated based on SNR

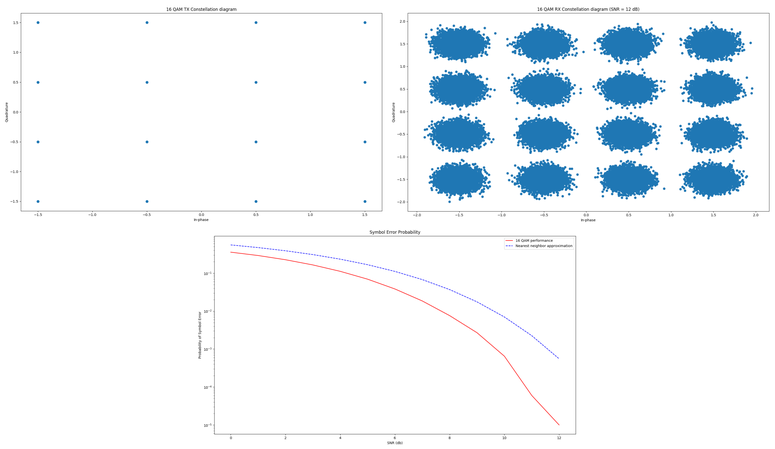

Figure 1. Python code simulated Tx (top left), Rx (top right), and symbol error probability curve (bottom).

Contact Us

Emerson Kim – ekim1221@uw.edu
Kevin Sadykhov – sadykhov@uw.edu

GNU Radio Simulation

GNU Radio is an open source software toolkit that allows the user to combine blocks or make their own blocks for signal processing. GNU Radio can also be used to interact with the USRP by pushing the design to the USRP hardware to run. We used GNU Radio to simulate our design as well as test our design on the hardware.

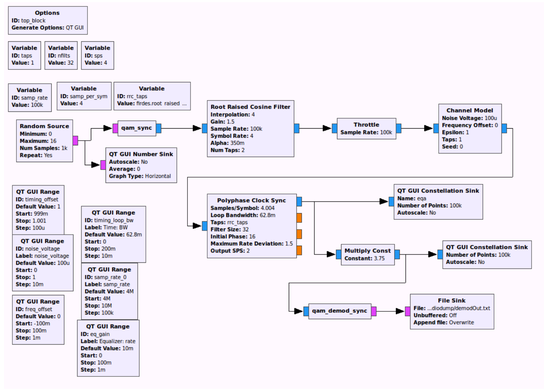

Figure 2. The flowgraph of our 16 QAM system.

- Extended our Python code to custom GNU Radio blocks
- Created a custom QAM modulator and demodulator block
- Accepts a byte stream as input and classifies each point as a 4-bit value
- After completion, the transmitted and received data can be compared and checked for accuracy

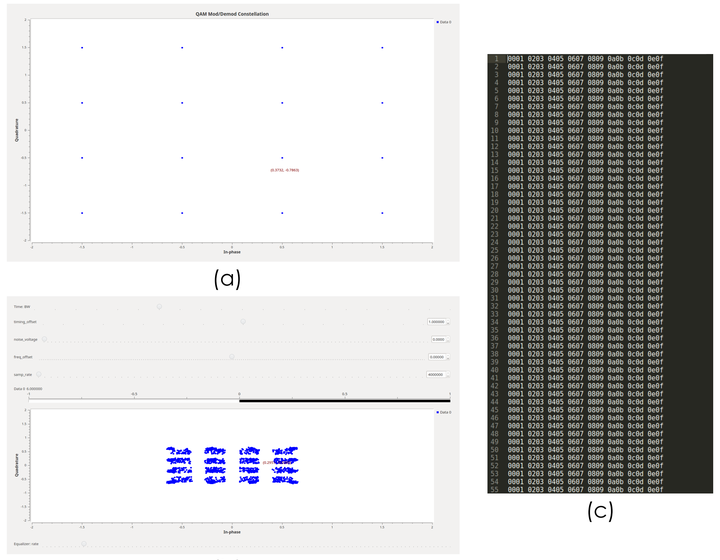

Figure 3. (a) The constellation diagram from the modulator, (b) constellation diagram after transmission, and (c) the data being sent.

Results and Findings

- A known vector of data (values 0 through 15) are repeatedly transmitted, modulated, and recovered by the demodulator (Figure 3c)
- During real transmission, an ideal constellation cannot be preserved due to noise interference
- Constellation values decreased as seen in Figure 3b
- In order for the demodulator to correctly read the data, recovery on the waveform must be performed
- A Polyphase Clock Sync block was used to automatically compensate for clock offsets
- A Root Raised Cosine Filter block was used to shape the data

Future Work

- Implementing our GNU Radio blocks to work with the Ettus USRP X310
- Transmitting over a real medium makes it difficult to control parameters such as clock offsets and noise
- Many unpredictable issues occur when hardware is invoked in place of a simulation
- When the recovery issue is dealt with, we should be able to transmit data using both wired and wireless connections

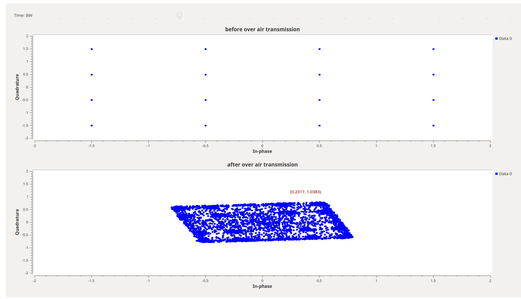

Figure 4. The constellation diagram produced with phase offset and incorrect clock and frequency recovery.

Acknowledgements

We would like to acknowledge and thank the following individuals who helped us during this project:

- Nicholas Morello, Millennium Space Systems: Our industry mentor who has given us guidelines and guidance throughout the project.
- Professor Sumit Roy: Our faculty advisor who has helped us set and achieve realistic and manageable goals.
- Professor Payman Arabshahi: Our capstone professor who has helped us with various details of the project.
- Anish Ashok and Kyeong Su Shin, Graduate Students: TAs who have helped us extensively by assisting with debugging and offering their knowledge on various issues.