Implementing Software Defined Radio – a 16-bit QAM System using the USRP2 Board

Functional Description and Complete System Block Diagram

Patrick Ellis & Scott Jaris

Dr. In Soo Ahn & Dr. Yufeng Lu

October 16, 2010

Introduction

Radios exist in many facets of our daily life and Software Defined Radio has the promise to revolutionize the way radios function. Most radios today have limited flexibility in regards to cross-functionality and ease of modifications. Software Defined Radio takes what used to be ingrained physically within the radio and provides a variety of hardware and software alternatives that add more flexibility and functionality. We are looking to embody this definition by using the USPR2 board and show the benefits of Software Defined Radio by implementing a 16-bit QAM system.

Goal

The ultimate goal of this project is to implement a 16-bit QAM system onto a USPR2 board to demonstrate the power, flexibility, and advantage of doing so with Software Defined Radio (SDR). However, a variety of smaller goals must be met in order for the main one to be realized. These are listed below.

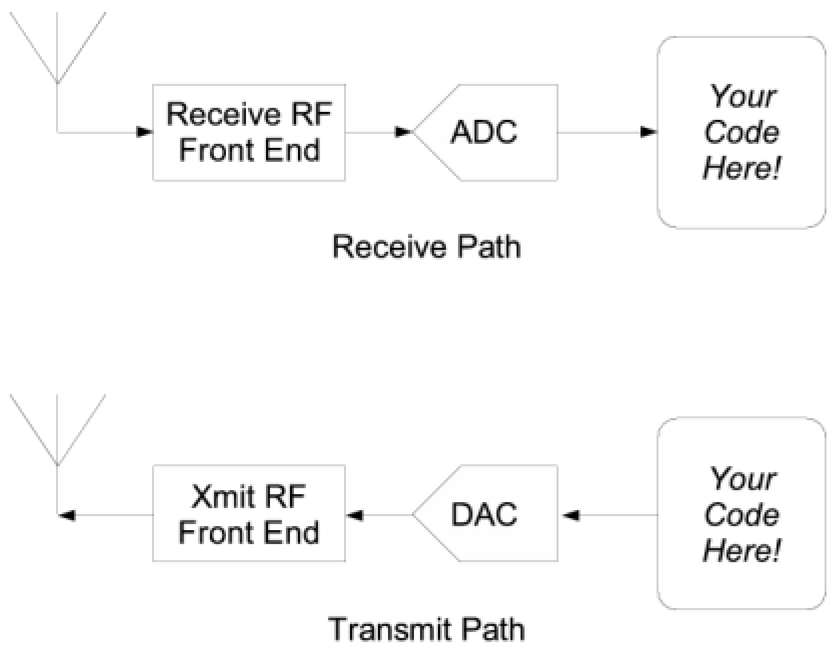
1. Design a 16-bit QAM system using blocks made available by the GNU Radio companion.
2. Design a 16-bit QAM system controlled completely by a Python program calling the block functions.
3. Design a 16-bit QAM system using blocks constructed from scratch and run with the wrapper language, Python.

Through these short and long-term goals it is believed that a clear justification can be made on the practicality and usability of Software Defined Radio by comparing the results of these systems and their ease of construction with previously completed projects without using Software Defined Radio.

High Level Block Diagram

Figure 1 shows a typical block diagram of software defined radio that demonstrates its flexibility by allowing the designer to change or update the function of the radio as easily as re-writing a few lines of code. The ADC/DAC are provided and carried out by the USRP2 board while the receiving and transmitting ends utilize the DBSRX(800 MHz to 2.4 GHz Receiver) or XCVR2450 (2.4-2.5 GHz and 4.9 to 5.85 GHz Dual-band Transceiver) daughter board respectively.

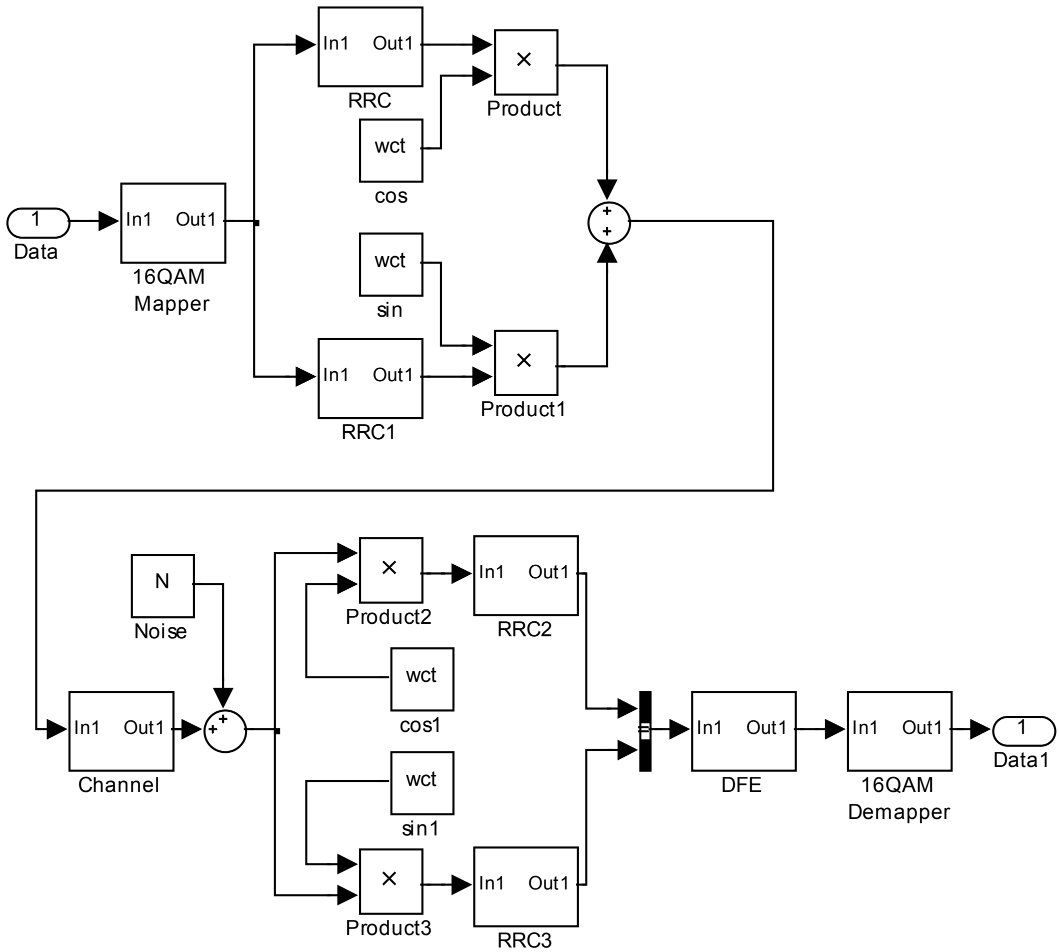
Figure 1. Software radio block diagram



System Level Flowchart

Figure 2 shows the flowchart of a general 16 bit QAM system which we hope to implement using SDR. The processes on both sides of the channel represent each of the “Your Code Here!” blocks in Figure 1. The RRC blocks represent root raised cosine filters and the DFE is a decision feedback equalizer. This shows how SDR allows the designer to implement the majority of the system in software, as close the RF ends as possible.

Figure 2. 16 bit QAM block diagram



<u>Conclusion</u>

This project requires a great amount of new, unfamiliar software which may prove to be one of its biggest challenges. Currently, we are at a point where the direction we are taking is not set in stone. As we continue to learn more about GNU radio, python, and the USRP2 board we will have a better idea of where the project is headed and be able to adjust and expand upon our goals. We look forward to learning a lot about the true power of software defined radio and its functions.

References

[1] http://www.gnu.org/software/gnuradio/doc/exploring-gnuradio.html

[2] http://www.steepestascent.com/content/mediaassets/pdf/example%2016qam.pdf