Design Document

**Concatenated Codes in**

**Amateur Radio Satellite Telemetry**

Submitted To:

Professor Dennis Silage

Senior Design Project I and II

Temple University

College of Engineering

1947 North 12th Street

Philadelphia, Pennsylvania 19122

December 2, 2013



Prepared By:

B. Keith, B. Thibodeau, and C. Destin

Faculty Advisor(s): Dr. Dennis Silage

Programmable Communication Group

Temple University

College of Engineering

1947 North 12th Street

Philadelphia, Pennsylvania 19122

For further information, please contact Dr. Dennis Silage (email: silage@temple.edu).

|  |  |  |
| --- | --- | --- |
| **Team SD1-01** | **Programmable Communication Group** | **PCG** |
| **Team Members** | Brandon Keith, Brian Thibodeau, Cedric Destin | |
| **Advisor(s)** | Dennis Silage | |
| **Coordinator** | Thomas Sullivan | |
| **Department(s)** | Electrical and Computer Engineering | |
| **Project Title** | Concatenated Codes in Amateur Radio Satellite Telemetry | |
| **Abstract** | **TBD** | |
| **URL** | <https://sites.google.com/a/temple.edu/programmable-communication-group/> | |

**Executive Summary**

Table of Contents

[1. Problem 5](#_Toc372906561)

[1.1. Overall Objectives 5](#_Toc372906562)

[1.2. Historical and Economic Perspective 5](#_Toc372906563)

[1.3. Candidate Solutions 5](#_Toc372906564)

[1.3.1. Forward Error Correction: Block and Convolutional Codes 5](#_Toc372906565)

[1.3.2. Line Coding: Non Return Zero and Manchester 5](#_Toc372906566)

[1.3.3. BPSK Carrier Recovery 5](#_Toc372906567)

[1.3.4. Coherent and Non-Coherent BFSK Modulation 5](#_Toc372906568)

[1.3.5. Coherent and Non-Coherent BFSK Demodulation 5](#_Toc372906569)

[1.3.6. BPSK and BFSK Timing Recovery 5](#_Toc372906570)

[1.4. Proposed Solution Concept 5](#_Toc372906571)

[1.5. Major Design and Implementation Challenges 5](#_Toc372906572)

[1.6. Implications of Project Success 5](#_Toc372906573)

[2. DESIGN REQUIREMENTS 5](#_Toc372906574)

[2.1. Functional Design Constraints 5](#_Toc372906575)

[2.2. Non-Functional Design Constraints 5](#_Toc372906576)

[3. APPROACH 5](#_Toc372906577)

[3.1. Software Simulation Using Matlab/Simulink 5](#_Toc372906578)

[3.1.1. FSK Modem 5](#_Toc372906579)

[3.1.2. BPSK Modem 5](#_Toc372906580)

[3.1.3. Interleaving FEC 5](#_Toc372906581)

[3.2. Hardware Implementation using ISE Design Suite 5](#_Toc372906582)

[4. EVALUATION 5](#_Toc372906583)

[5. SUMMARY AND FUTURE WORK 6](#_Toc372906584)

[6. ACKNOWLEDGEMENTS 6](#_Toc372906585)

[7. REFERENCES 7](#_Toc372906586)

# Problem

## Overall Objectives

It has been shown that forward error correction dramatically improves bit error rate performance (BER) in amateur packet radio satellite telemetry links (Hsiao, et. al, 2000). Additionally, it has been shown that binary phase shift-keying (BPSK) modulation is more reliable and bandwidth-efficient than audio frequency shift keying (AFSK) modulation (Hsiao, et. al, 2000). Being that average amateur satellite telemetry benefits from neither of these facts, this senior design project aims to demonstrate the degree to which forward error correction and interleaving techniques with BPSK modulation can improve the reliability of the average amateur satellite telemetry link. Consequently, this senior design project advocates for improved robustness in amateur packet radio communication systems, specifically in those systems dealing with satellite telemetry.

Amateur packet radio satellite telemetry is often unidirectional (simplex) and does not benefit from automatic repeat request (ARQ) like in other bidirectional (duplex) amateur packet radio communications (Hsiao, et. al, 2000). In other words, if even one bit of an AX.25 telemetry packet is received in error, the entire packet is discarded and cannot be re-transmitted (Karn, 1994). This means that beacon signals from the amateur satellites must be transmitted with enough power to ensure that the embedded telemetry packet is received without error (de Milliano, et. al, 2010). BPSK modulation with forward error correction combined with interleaving can supersede AFSK, resulting in greatly improved network reliability and power-efficiency in amateur packet radio satellite telemetry. The enhanced network reliability could lower overall power consumption in amateur telemetry satellites (de Milliano, et. al, 2010), resulting in two benefits: 1) reduced cost of satellite construction, and 2) making amateur telemetry satellites more technologically and financially accessible to amateur satellite operators by reducing the size, cost, and complexity of ground station antennas (Karn, 2011).

Hence, the ultimate goal of this senior design project is to demonstrate the improved network reliability and power-efficiency that results from implementing forward error correction and interleaving with BPSK modulation in amateur packet radio telemetry satellites and ground stations.

## Historical and Economic Perspective

The standard digital modulation scheme used for amateur radio very-high frequency (VHF) and ultra-high frequency (UHF) operation is Bell 202 (Capitaine, et. al, 2010). Bell 202 provides AFSK modulation using 1200 Hz and 2200 Hz tones, with a resulting data rate of 1200 b/sec. It is typically used in the physical layer of the AX.25 data link layer protocol and this has been the case since the early 1980s (Karn, 1994). In 1984, when Bell 202 was a fairly new standard in the amateur radio community, Steve Goode, K9NG, performed an exhaustive bit error rate (BER) performance analysis of a standard Bell 202 modem (Goode, 1984). Goode found that at least 25 dB of FM receiver quieting (25 dBQ) was necessary for high communication reliability. In other words, 25 dBQ or greater was required to accurately receive 98% of incoming packets, which corresponded to a BER of 1.6e-5. Ralph Wallio, WORPK, figured out that with this BER, there is only a 1.603% chance of accurately receiving 117 consecutive 256-byte AX.25 packets (Wallio). Wallio concluded that “this is as Goode as it gets” and it is virtually impossible to get better results without error correction.

This poor reliability performance is not exclusive to amateur radio terrestrial communications. In 1995, it was demonstrated that error detection alone is not robust enough for amateur radio microsatellite communications (Hsiao, et. al, 2000). Particularly in simplex satellite communications, the harsh environmental conditions coupled with the microsatellite’s characteristically low transmitter power made for very unreliable telemetry data links (Hsiao, et. al, 2000). It has been demonstrated that forward error correction, specifically convolutional encoding and decoding, can generally correct up to 75 percent of errors (Hsiao, et. al, 2000). It was also demonstrated that 1200 b/sec BPSK provides much more reliable transmission quality than 1200 b/sec AFSK, irrespective to whether the VHF or UHF amateur bands are used. Moreover, it was demonstrated that BPSK occupies a considerably smaller frequency bandwidth than AFSK while possessing excellent anti-interference properties. And with a general tenfold BER performance increase for both 1200 b/sec AFSK and BPSK over 144 MHz VHF, implementing forward error correction for amateur satellite telemetry was clearly demonstrated to be better than not implementing forward error correction.

In 2003, the AAU-Cubesat was one of the first pico-satellites to be launched into space. Moreover, the miniaturized satellite harbored a communication subsystem that implemented both forward error correction and interleaving over 9600 b/sec Gaussian minimum shift-keying (GMSK) AX.25 (Alminde, et. al, 2002). The enhanced robustness and data rate was justified by the fact that it had to transmit approximately 1461 kilobytes (kB) of telemetry and picture data per day. This simply would not have been possible had the satellite not utilized error detection and correction. However, it operated at 437.9 MHz, meaning that it was particularly difficult for the average amateur radio operator with a 2-meter radio transceiver to receive its telemetry data. This would particularly bother Phil Karn, KA9Q, who is a strong proponent of making robust satellite telemetry links accessible to the average amateur radio operator (Karn, 2011). Karn asserts that robust telemetry links (using forward error correction) reduce the cost of satellite construction and simplify ground antennas, making amateur radio satellite telemetry much more technologically and financially accessible to amateur satellite operators (Karn, 2011).

As amateur satellite designers foresee the next generation of miniature satellites (de Milliano, et. al, 2010), and as the next generation of amateur satellites equipped with robust communication schemes continue to ascend into space, and as miniature satellites become increasingly more financially and technologically accessible to amateur satellite operators, it must be clearly demonstrated to the amateur radio community how these advancements trump the ubiquitous 1200 b/sec AFSK AX.25. Hence, to reiterate, this senior design project hopes to clearly demonstrate the performance advantages that yield from using forward error correction and interleaving schemes with BPSK modulation in amateur satellite telemetry.

## Candidate Solutions

Typical functions of a modem include forward error correction, source encoding, modulation, demodulation and source decoding. In the last century, many solutions have been proposed that trade performance in terms of bandwidth, transmission power and complexity. In this section we consider two common types of forward error correction - block codes and convolutional codes, two line codes – Non-Return to Zero (NRZ) and Manchester code, and finally coherent and non-coherent demodulation techniques used for BPSK and FSK. This includes solutions for carrier recovery and timing recovery.

### Forward Error Correction: Block and Convolutional Codes

Forward error correction (FEC) is a form of robust channel coding. It is used to correct errors that are injected into a digital communication link across a noisy propagation medium. FEC codes fall into two general categories: block codes and convolutional codes. It is important to note that at the time of writing this document, the Xilinx CORE Generator in Project Navigator ISE 14.6 only consists of one block coder/decoder pair and one convolutional coder/decoder pair. The block coding pair consists of a Reed-Solomon coder and decoder. The convolutional coding pair consists of a convolutional encoder and a Viterbi decoder. Hence, the FEC engine will be limited to using these channel code pairs.

In Section 1.3.5, we discussed that the satellite communication link is vulnerable to random errors and burst errors. Block codes are better suited for correcting burst errors while convolutional codes are better suited for correcting random errors (Viswanathan, 2013). A combination of block codes and convolutional codes, namely a two-level coding system, are used in many systems to provide robustness against both kinds of errors (see Figure 9). This two-level coding system consists of a coding chain and a decoding chain. The coding chain resides in the transmitter and consists of a Reed-Solomon encoder, followed by an interleaver, then a convolutional encoder. The decoding chain resides in the receiver and undoes what the coding chain did. Namely, the decoding chain consists of a Viterbi (convolutional) decoder, followed by a de-interleaver, then a Reed-Solomon decoder.



Figure 9. Top-level diagram for the FEC engine (System C) consisting of a block code pair, an interleaving pair, and a convolutional code pair.

### Line Coding: Non Return Zero and Manchester

### BPSK Carrier Recovery

The demodulator is responsible for providing either coherent or non-coherent demodulation. Coherent demodulators require phase synchronization between the received signal and the locally generated oscillator. Conversely, Non-coherent demodulation does not require synchronization and makes no attempt to estimate the phase of the received signal. The advantage of non-coherent modulation is that it does not require additional hardware like phase-locked loops which are used to lock onto the incoming carrier phase. However, the LEO-AMSAT’s we are interested in communicating with use BPSK for downlink and thus requires the design of a coherent demodulator.

The successful extraction of information from a received signal in a coherent demodulator requires both carrier and timing synchronization. Figure 1 illustrates the architecture of a typical coherent demodulator.



Figure 1. Received waveform takes two paths. First path extracts carrier for coherent demodulation and the second path recovers timing information. This architecture is based on the optimum binary receiver

The received signal from the transceiver is first processed by a band pass filter to remove as much noise as possible and then sent to the carrier recovery circuit. Recovering the carrier is done in one of two ways, the squaring loop or the Costas loop. Each method utilizes phase-lock concepts and has its own advantages and disadvantages in terms of complexity and performance.

**Carrier Recovery using Squaring Loop**

The squaring loop is a popular choice for coherent demodulation of BPSK waveforms. It’s mathematically easy to analyze and its hardware implementation is not as complex as the Costas loop. As the name implies, the received signal is squared to remove any phase offsets and then processed by a bandpass filter to remove as much noise as possible. After the band pass filter, the signal is fed to a phase-lock loop (PLL) for phase and frequency tracking. Once the output of the voltage controlled oscillator (VCO) is locked in phase and frequency with the received signal, its frequency is divided by two. The resulting carrier is fed back to the mixer where it is mixed with the received waveform and the timing can be recovered. The operation of the squaring is shown in Figure 2.



Figure 2. Squaring loop used for carrier recovery in the coherent demodulator. The Phase-Lock Loop utilizes feedback to track and lock onto in the received waveforms suppressed carrier

**Carrier Recovery using Costas Loop**

Another method for carrier recovery was proposed by John P. Costas in his 1957 paper, *Synchronous Communication*. Unlike the squaring loop whose only purpose is suppressed carrier reconstruction, the Costas loop is capable of synchronous data detection in addition to suppressed carrier reconstruction. One of its disadvantages is its mathematical complexity compared to the squaring loop, but in terms of hardware components needed for complete coherent demodulation, they both require approximately the same amount.



Figure 3. Costas loop used for suppressed carrier reconstruction as well as synchronous data detection.

Coherent modulation utilizing the Costas loop would require one band-pass filter, three low-pass filters, three multipliers and a VCO. Likewise, the squaring loop would also require one band-pass filter, three multipliers (including the squarer) and a VCO. Instead of three low-pass filters needed by the Costas, the squaring loop only requires two. Note also that the squaring loop requires a flip-flop for frequency division, but with today’s FPGA’s, a single flip-flop is negligible. The decision for implementing the squaring loop versus the Costas loop will ultimately be decided by their tracking and locking performance in the presence of noise and Doppler shifts (See section 1.5, Major Design and Implementation Challenges).

### Coherent and Non-Coherent BFSK Modulation

The BFSK modem abides by the Bell 202 standard which uses frequencies of 1200 Hz for Mark (*b0*) and 2200 Hz for Space (*b1*). Following this protocol, the phase of the signal can be implemented either coherently or non-coherently. A coherent modulation (continuous phase modulation) implies that the phases of the two tones representing the data are always the same, which inherently prevents discontinuous jumps between a Mark and Space. Conversely, non-coherent FSK modulates the two signal waveforms without any effort to match the two signals’ phase, hence the modulated signal may experience discontinuous jumps in phase.

Coherent FSK modulators tend to consist of several complicated components, and therefore are not commonly used to avoid unnecessary loss of power although they yield a better BER performance (Rao et. al, 1990). On the other hand Non-coherent FSK modulation is simpler to implement and is commonly used in several modulation despite its BER performance compared to the coherent modulation. However, with the technological development, coherent modulation can surely be implemented with as much efficiency as the non-coherent modulation.

**Non-coherent modulation**

As previously mentioned, coherent modulation requires continuous phase of the modulated signal, which can involve complicated hardware or algorithms. As a result, it is common to ignore the phase of the signals and directly modulate the two signals. This implies that the phase modulated signal will be subject of random variations. The non-coherent modulator can be implemented using the two sinusoidal wave generators or two sine functions and a multiplexer controlled by the input data *m(t)*. Switching between the frequencies will generate a BFSK waveform with a bit period equal to the periodicity of the switches.



Figure 4. BFSK modulator used in non-coherent modulators. The data *m(t)* controls the output of the multiplexer at data’s baud rate.

**VCO Coherent modulation**

Non-coherent waveforms originate from the fact that two totally different sources are used to modulate the data, therefore the phase resulting from the modulator varies as signal is altered through the two tones. Using a single source to modulate will maintain a continuous phase as expected. Voltage Controlled Oscillators are commonly used to provide a continuous phase, and generate a sinusoidal wave based on the input control signal.



Figure 5. Coherent modulator for BFSK. The data *m(t)* controls the output of the VCO through Eq. 3

### Coherent and Non-Coherent BFSK Demodulation

The Bell 202 Protocol is quite complex due to the frequency deviation and the ratio between the data rate and carrier frequency. The Bell 202 modem uses frequencies with a small frequency deviation from the carrier frequency to represent it binary data and where the tones selected are 1200 Hz and 2200 Hz. The frequencies selected to represented two symbols result in a signal space that is difficult to optimize since the frequencies are not orthogonal as the minimum frequency separation is denoted in equations (2) and (3) below (Nguyen, et. al, 2009).

**Coherent demodulation**

Similar to the modulator, the demodulator can be categorized into a coherent modulator and a non-coherent demodulator. In the coherent demodulator, the phase of the modulated signal is either known or is extracted prior to demodulation. Several methods are used to extract the phase of the modulated waveform, such as the phase-lock loops or more complicated systems as illustrated in Figure 6.



Figure 6. Coherent modulator for BFSK. The data *m(t)* controls the output of the VCO through Eq. 3

The coherent demodulator in Figure 6 uses two parallel branches for matching the Space and Mark unto the two orthonormal basis functionsand. Finally, using the appropriate threshold and decisions, the bits can be recovered using Maximum Likelihood. The correlation receivers or matched filters and are designed to be orthonormal to each other, and at a frequency corresponding to the Mark and Space signals. The process of using matching filter results in an optimum demodulator in terms of BER and can even be reduced to a single correlation receiver using the following relationship:

**Non-Coherent demodulation**

In the case of a signal with discontinuous phase non-coherent demodulation is regarded as the ideal demodulator in FSK modulation. The advantage of non-coherent demodulator come from their ability to ignore the phase change contained in the signal. Matched filters are still utilized however, an envelope detector is present in each branch after each tone’s matched filters.

The matched filters are configured with the same objective as the coherent receiver, and the use of the envelope detector removes the phase changes. The performance of the non-coherent demodulator results in performances that closely approach the performance of the optimum coherent receiver. (Linsey et. al, 1977)(Rao et. al, 1990)



Figure 7. Non-coherent demodulator for BFSK, where the Mark and Space filters are centered at 2200Hz and 1200Hz respectively.

**Non-coherent demodulation (PLL)**

****

Figure 8. Non-coherent demodulator for BFSK using PLL.

The use of a phase lock loop is also a valid method for demodulating FSK. The PLL has been integrated in several radio for demodulating FM and can also serve to demodulate FSK signals. In the case of non-coherent signals, the PLL acts as an estimator of the frequencies and phases (.) By rapidly matching the output of the VCO, the PLL is used to appropriately estimate the correlation between the signal and the output of the VCO.

### BPSK and BFSK Timing Recovery

## Proposed Solution Concept

This senior design project will determine the telemetry packet error rate (PER) performance and the coding gain due to implementing forward error correction in amateur radio satellite telemetry. These two parameters, respectively, will allow us to compare the reliability and power-efficiency between using and not using FEC in amateur radio satellite telemetry. This implies that we will be comparing several digital communication systems. In this senior design project, three digital communication systems will be developed for modeling amateur radio satellite telemetry. The first system (System A) will replicate most amateur radio satellite telemetry links that exist today – 1200 b/sec Bell 202 modulation without FEC. The second system (System B) will exploit the fact that BPSK modulation is better than AFSK in amateur VHF and UHF operations (Hsiao, et. al, 2000) – 1200 b/sec BPSK without FEC. The third system (System C) will be a robust version of system B – 1200 b/sec BPSK with FEC.

Each of the three digital communication systems will represent simplex amateur satellite telemetry over a transmission medium (i.e., AWGN). We have chosen to use the AWGN channel model to represent our transmission medium because , compared to other channel models, it provides maximum bit corruption and it is assumed that systems that perform well in AWGN perform well in real-world scenarios (Viswanathan, 2013). Additionally, AWGN is a good model for many satellite communication links including amateur radio satellite telemetry (Viswanathan, 2013). Hence, no other satellite telemetry phenomena, such as interference, distortion, fading, or Doppler shift, will be modeled in this senior design project. Each system will form a digital loopback for BER performance analysis. Each system will contain a modulator (at the transmitter), a demodulator (at the receiver), a simulated transmission medium (i.e., AWGN), and a bit error rate tester (BERT) in software via PC. Additionally, the third system (System C) will also implement forward error correction. The three systems are outlined below:

1. **System A**
   1. Transmitter
      1. AX.25 packet generator (BERT)
      2. Bell 202 modulator (1200 b/sec)
   2. AWGN Channel
   3. Receiver
      1. Bell 202 AFSK demodulator (1200 b/sec)
      2. AX.25 packet comparator (BERT)
2. **System B**
   1. Transmitter
      1. AX.25 packet generator (BERT)
      2. BPSK modulator (1200 b/sec)
   2. AWGN Channel
   3. Receiver
      1. BPSK demodulator (1200 b/sec)
      2. AX.25 frame comparator (BERT)
3. **System C**
   1. Transmitter
      1. AX.25 packet generator (BERT)
      2. FEC engine (send)
      3. BPSK modulator (1200 b/sec)
   2. AWGN Channel
   3. Receiver
      1. BPSK demodulator (1200 b/sec)
      2. FEC engine (receive)
      3. AX.25 packet comparator (BERT)

## Major Design and Implementation Challenges

The biggest design challenge associated with this project is the development of a carrier reconstruction circuit that is capable of mitigating the effects of Doppler shift. The relative motion of satellites in orbit around earth with respect to the ground station can cause the received frequency to appear 20 kHz above or below its nominal downlink frequency. In John A. Maglicane’s 1993 design, he derived a control signal from the carrier recovery circuit that simulated a person tuning the transceivers frequency control button. In our design, Doppler shift correction will be done autonomously through the use of a type II PLL.

The challenge is designing a stable control loop that minimizes time to lock and inter-symbol interference but still has a narrow enough bandwidth to reduce noise and the bit error rate. Since the PLL is an inherently non-linear system, it must be linearized in terms of the phase of the received signal. This problem becomes more challenging if the Costas loop is implemented because the arm filters much be matched perfectly. However, the advantage of an all digital Costas loop is that designing two identical filters is much easier than if it were done with analog components.

## Implications of Project Success

It was hinted in Section 1.1 (Overall Objective) and Section 1.2 (Historical Perspective) that this senior design team has identified a problem within the amateur radio community. According to amateur radio operator Jeff Davis, KE9V, amateur radio has somewhat of a *lost future* (Davis, 2010). In the earlier half of the 20th century, amateur radio operators led the forefront of “discovery and experimentation” in the industries of electronics and communications. This was the case because many amateur radio operators were in fact professional electronics technicians and electronics engineers that designed and implemented the next wave of commercial and military communications. Oftentimes, the budding amateur radio operator, a *neophyte* if you will, would go on to become the next electronics repairman or electronics engineer. However, Davis highlights the fact that at some point in the past, the amateur radio community reached somewhat of a crossroads. Up to that point in time, the amateur radio community had pioneered Frequency Modulation (FM) communications over ultra-high frequency (UHF) and very-high frequency (VHF) operations, stationed repeaters throughout the land for long-distance over-air communications, and launched amateur radio satellites into the heavens which led to improved methods for space communications in addition to low-cost spacecraft manufacturing and launch. Davis highlights the fact that although the non-amateur world would go on to produce cellular technology, drastically improved over-air communications, and intelligent military digital communications, the amateur radio community as a whole decided to dwell in the past as the future marched ahead without it.

This senior design team identified one amateur radio operator and notable electrical engineer, Phil Karn, KA9Q, in his efforts to secure the future of amateur radio. Like Jeff Davis, Phil Karn is also aware of amateur radio’s *lost future*. In a modem design article (Karn, 2011), Karn hints that making amateur radio communications more accessible to prospective amateur radio operators is one solution for securing the future of amateur radio. Specifically, in the design article, Karn identifies the fact that amateur radio satellite communications is mostly inaccessible to amateur radio operators because the equipment involved is too expensive and esoteric. Karn’s philosophy is that by making amateur radio satellite communication accessible to all amateur radio operators, school demonstrations will be more commonplace and consequently more kids will want to become amateur radio operators. It is implied that if more kids become amateur radio operators, or *hams*, amateur radio in general cannot have a *lost future*.

Hence, according to Phil Karn, one solution to securing the future of amateur radio is to make amateur radio satellite communications more accessible to kids. In order to make amateur radio satellite communications more accessible to kids, the amateur radio equipment involved in said communications must be less expensive and esoteric. By expensive and esoteric, Karn is referring to software-defined radio systems and bulky antennas. This kind of equipment is regarded as being too inaccessible for the typical school demonstration of amateur radio satellite communications. Instead, Karn emphasizes the fact that a standard 2-meter single sideband (SSB) transceiver and an inexpensive antenna system should be all that is required at these school demonstrations. Satellite communications in general requires for relatively high-powered transmission of signals to overcome the high fading (energy loss) that results from an electromagnetic wave propagating through space (Sklar, 2001). In fact, free space attenuates an electromagnetic wave more than any other form of power attenuation along a satellite communication link. Hence, it is often the case that transmitted signals between amateur packet radio satellites and ground stations either deal with high transmission power to acquire a digital communication link with high data reliability or lower transmission power and low data reliability and link efficiency. It is understood that if you increase the reliability of a communication link, you can consequently get away with communicating at a lower signal-to-noise (SNR) ratio (Sklar, 2001). This results in lower transmitted power between an amateur radio satellite and ground station. Being that the power amplifier of the transmitter utilizes the most power of an amateur radio satellite, the lower power requirement could result in cutting the cost of satellite construction and simplify the ground antennas (Karn, 2011). Consequently, amateur radio satellite communications would become more *accessible* to prospective amateur satellite operators.

In a similar fashion as Phil Karn, KA9Q, and others (Hsiao, 2000), this senior design project aims to demonstrate that there are much more power-efficient digital communication schemes than are currently employed in most amateur radio satellites today. The intention of this senior design project is to provide concrete evidence that BPSK modulation and concatenated error-correcting codes can make amateur radio satellite communications more power-efficient and hence, more *accessible* to prospective amateur satellite operators. Perhaps a simple BER performance analysis of popular and prospective communication schemes, like showcased in this senior design project, would further persuade an amateur satellite designer to employ more power-efficient communication schemes in the increasing fleet of miniaturized amateur radio satellites.

# DESIGN REQUIREMENTS

## Functional Design Constraints

|  |  |
| --- | --- |
| **Name** | **Description** |
| Bit Rate | Provides 1200 b/sec data rate to meet LEO-AMSAT telemetry requirements |
| FEC |  |
| Modulation Type | Supports BPSK and FSK modulation and Demodulation |
| Operating Frequencies | The modulator and demodulator will provide operation between 1200 Hz and 2400 Hz in accordance with the Bell 202 standard |
| Interface |  |
| Signal to Noise (SNR) |  |
| Bit Error Rate |  |

Table 2. Functional design constraints for the GADGET system.

## Non-Functional Design Constraints

|  |  |  |
| --- | --- | --- |
| **Type** | **Name** | **Description** |
| Economic | Cost |  |
| Environmental | Temperature |  |
| Environmental | Power Consumption |  |
| Manufacturability | Dimensions |  |
| Manufacturability | Weight |  |

Table 3. Non-functional design constraints for the GADGET system.

# APPROACH

**Here, will be an introduction and summary of our design approach from high-level Simulink blocks to lower level hardware realizable Xilinx blocks, and then finally delving into hardware implementation. The approach is divided into TWO major sub-sections, *Software Simulation Using Matlab/Simulink* and *Hardware Implementation Using Xilinx ISE Design Suite*. The first subsection WILL be completed by DEC 2. The second sub-section will be completed in SDII.**

## Software Simulation Using Matlab/Simulink

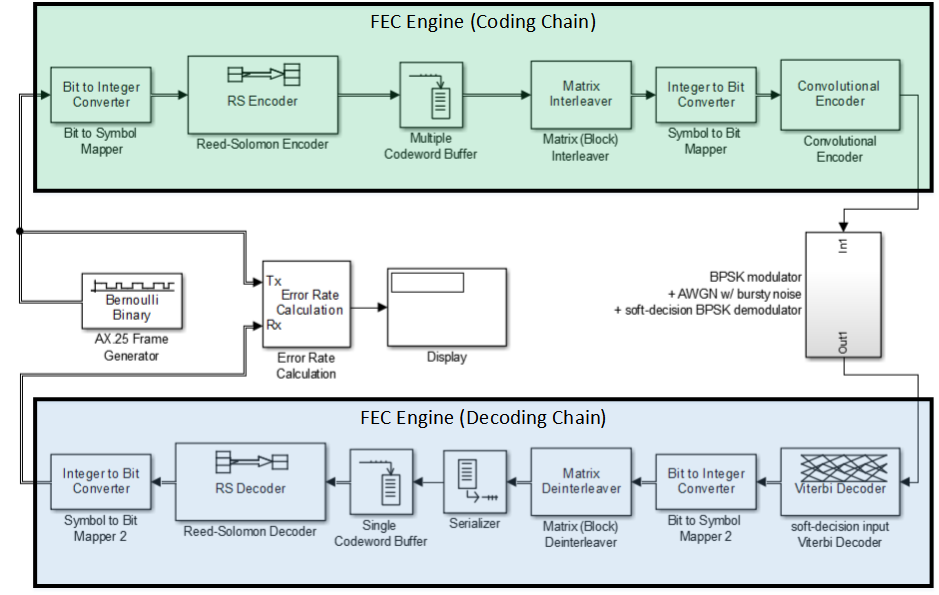
### FSK Modem

* Evaluating the different methods for modulations
  + Evaluating the Coherent and the non-coherent modulators
    - How many components do they need
    - How likely are they to be distorted
* Demodulation
  + Testing the Non-coherent FSK
    - Add noise
      * Change threshold
    - Find time response of filters
    - The number of components
    - BER
  + Test the PLL FSK demodulator
    - Commonly done for tone detectors
    - Add noise
    - Obtained Tuning parameters
    - BER
* Clocking recovery

### BPSK Modem

### Concatenated FEC codes

The third digital communication system to be analyzed is System C – which comprises everything in System B with the addition of a forward error correction (FEC) engine and a modification to the BPSK modem. Specifically, the modification includes making the BPSK demodulator implement soft-decision decoding instead of hard-decision decoding. This change will prove to be beneficial to the overall SNR of the digital communication system. An explanation of this will appear shortly as we describe the FEC engine in a clockwise fashion (see Figure 3.1.3.1) starting with AX.25 frame generation (top left) and looping back around to packet error rate calculation (bottom left).

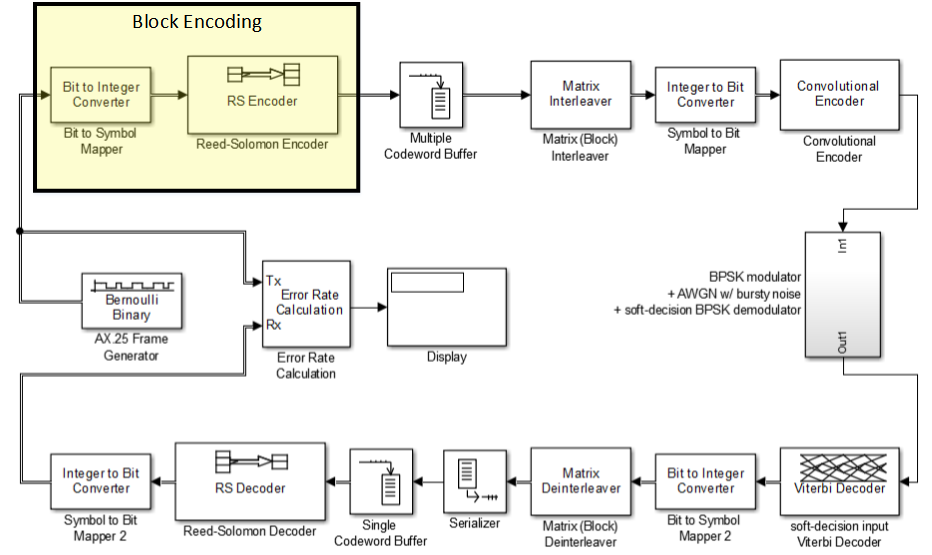


**Figure 3.1.3.1.** A system-level diagram depicting System C with the FEC engine. The engine is comprised of a coding chain and a decoding chain. Besides from the addition of the FEC engine, System C differs from System B in that the BPSK demodulator implements soft-decision decoding instead of hard-decision decoding.

The FEC engine comprises a concatenation of two forward error correcting codes. As discussed in Section 1, one of the codes will be a block code (correcting burst errors) while the other code will be a convolutional code (correcting random errors). The next text segments will elucidate the reasoning for this concatenation of FEC codes.

**Block Encoding**

One of the general categories of forward error correcting codes is *block codes*. Reed-Solomon (RS) codes, one form of block codes, perform exceptionally well in correcting burst errors in a received signal. This senior design project elected to incorporate an RS code into the FEC engine for the sole purpose of correcting burst errors that seep into the received bit stream. Figure 3.1.3.2 highlights the section of the FEC engine dedicated to block *encoding*.



**Figure 3.1.3.2.** Highlights the block encoding unit of the FEC engine.

At this point, one may ask themselves how does block *encoding* work to correct burst errors. To answer such a question, we must understand that once block *encoding* is done, it must be undone at some point – this is block *decoding*. Block *decoding* will be discussed shortly, but let us first examine how block *encoding* works and how we use Simulink to simulate its functionality. It should be noted that this will be a high-level explanation that circumvents the detailed implementation of block coding. Specifically, a thorough description of Reed-Solomon codes requires an involvement of abstract algebra, specifically Galois fields (Sklar, 2001). In this regard, the curious reader is recommended to visit the excellent mathematical treatment provided in the digital communications textbook entitled *Digital Communications: Fundamentals and Applications (2nd Edition)* by Bernard Sklar*.* Additionally, an RS code can be realized using a linear feedback shift register (LSFR), but in the interest of time, this senior design project has elected to utilize intellectual property cores in order to bypass this design step.

Let us first begin the explanation by imagining a bit stream. The block encoder deals with correcting *symbol* errors*,* not single bit errors. For instance, a group of 3 bits could be abstracted to one of eight symbols (0 through 7). This functionality is represented by the *Bit to Symbol Mapper* in Figure 3.1.3.2. The symbol is then operated on by the Reed-Solomon encoding process. This is represented by the *Reed-Solomon Encoder* block in Figure 3.1.3.2. Essentially, the Reed-Solomon encoder attaches a set of parity (or redundancy) symbols to the end of a collection of symbols (known as a message word) (Viswanathan, 2013). A RS code converts *k* symbols (the message word) into a codeword, or *block*, consisting of *k symbol*s. The RS encoder essentially extends the message word with *n-k* parity symbols. This is known as an (*n*, *k*) RS code. The following depicts a (7, 3) RS code which converts a 3-symbol message word into a 7-symbol codeword (*block)*:

🡪 [

The inputted bit stream representation of this would look like the following:

The (*n, k*) RS code has an error-correcting capability (*t*) expressed as (Viswanathan, 2013):

Put differently, the (*n, k*) RS code can correct up to *t* symbol errors in a given codeword. For instance, let us polute the previous codeword example with symbol errors. The first and second of the matrices below show one and two symbol errors, respectively, that are correctable by a (7, 3) RS code. However, the third matrix shows three symbol errors which is greater than the error-correcting capability of a (7, 3) RS code. Consequently, the code fails to correct the symbol errors in the third matrix.

[ correctable

[ correctable

[ not correctable

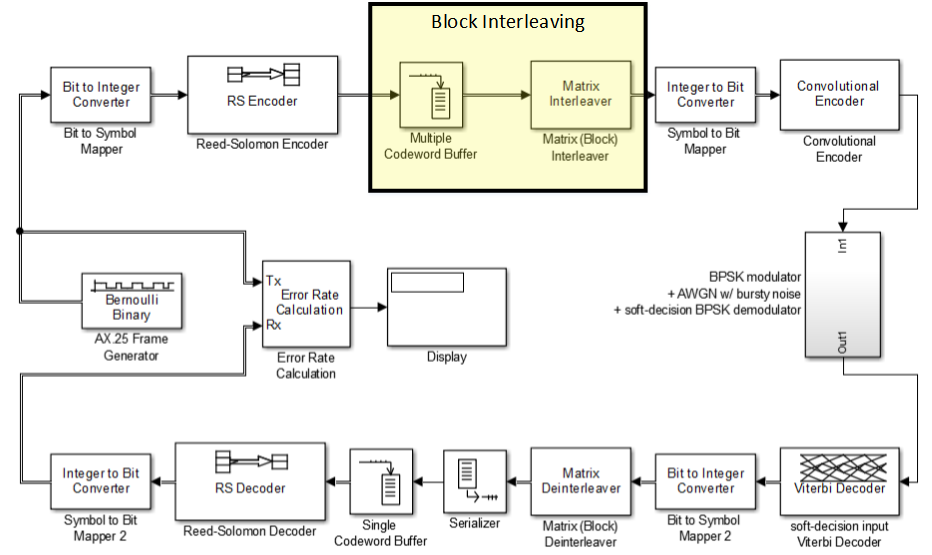
We see that regardless of if one or two symbols were received in error, the (7, 3) RS code could correct the symbol errors. One can imagine a lengthier RS code, such as the (255, 235) RS code, possesses an even more lenient error-correcting capability. In other words, the (255, 235) RS code can correct up to 5 symbol errors within a codeword. If the (255, 235) RS code deals with 8-bit symbols, this means that up to five contiguous symbol errors (up to 40 contiguous bits) are correctable. This elucidates the power of RS codes in correcting for long strings of received symbol errors (burst errors) caused by bursty noise in a propagation medium.

The performance of RS codes are a function of their symbol size (in bits), redundancy, and code rate (Sklar, 2001). One can easily imagine an RS code to be more successful at correcting errors the larger the codeword is, which means that a give burst error is relatively smaller (and hence more correctable). Hence, the larger the symbol size of a RS code, the larger the codeword is, and consequently the better the RS code performs. The code rate of an RS code is the ratio of symbols that comprise a message word and a codeword. Hence, the code rate is expressed as (Viswanathan, 2013):

When the code rate is high, the number of symbols that comprise a message word (*k*) and a codeword (*n*) are fairly close in value. The number of added redundancy symbols is fairly low. Contrarily, when the code rate is low, the number of added redundancy symbols is fairly high. This high number of redundancy symbols equates to a high computational complexity of the RS code and a higher bandwidth requirement (Sklar, 2001). However, a large number of redundancy symbols results in better error-correcting performance. Consequently, this senior design project will aim to optimize the symbol size, redundancy, and code rate for the purpose of increasing error correction capabilities without putting too much demand on hardware or bandwidth resources.

**Block Interleaving**

One can imagine that there are instances where a burst error is too extensive for a given RS code to correct. To increase the chances of the RS code receiving a sufficiently short burst error, we can essentially mix up the codewords from the RS encoder and then transmit the mixed information. This way, when an overly extensive burst error occurs over the propagation medium, the receiver can put the mixed information stream back into un-mixed sequence, which essentially splits the extensive burst error into a disjointed series of smaller, correctable burst errors. This technique is known as *block interleaving*. Figure 3.1.3.3 shows the Simulink blocks of System C responsible for block interleaving.

****

**Figure 3.1.3.3** Highlights the block interleaving unit of the FEC engine.

Block interleaving simply consists of a single (*d* x *n*) matrix permutation. The matrix consists of *d* rows (the *interleaver depth*) of the *n*-symbol wide codewords (blocks) generated by the RS encoding process. Hence, the matrix consists of *n* columns. The sole purpose of the *Multiple Codeword Buffer* in Figure 3.1.3.3 is to accumulate codewords (blocks) and provide the block interleaver with a matrix of codewords (blocks). Let the following table represent a codeword (block) matrix:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| 29 | 30 | 31 | 32 | 33 | 34 | 35 |

**Table 1.** Depicting a block matrix with five blocks (codewords). Each block comprises a single row and are colored differently to illustrate this point. There are seven columns to illustrate the point that we are dealing with seven-symbol wide blocks (codewords). The *interleaver depth* of this block interleaver is obviously 5 because there are five rows.

Each block is written into the block matrix *row-by-row* (e.g. from the top to bottom). The magic of block interleaving consists of the fact that the matrix is transmitted by reading the matrix *column-by-column* (e.g. from the left to right). For instance, the matrix of Table 1 may be filled as follows:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 |

The matrix will be *block interleaved* into the following sequence and transmitted further down the FEC encoding chain:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 8 | 15 | 22 | 29 | 2 | 9 | 16 | 23 | 30 | 3 | 10 | 17 | 24 | 31 | 4 | 11 | 18 | 25 | 32 | 5 | 12 | 19 | 26 | 33 | 6 | 13 | 20 | 27 | 34 | 7 | 14 | 21 | 28 | 35 |

Let us now demonstrate the power of block interleaving in augmenting the Reed-Solomon error-correcting capabilities. Let us assume that (7, 3) RS code generated the codewords in the matrix of Table 1. We know that the (7, 3) RS code can correct up to two symbol errors in a given codeword. Let us imagine that we elected to not use a block interleaver at all and just transmit the first of first of two matrices shown above. By a stroke of pure bad luck, let us assume that the *entire* third codeword gets corrupted by bursty noise. The result is shown in the following matrix (the red values represent burst error):

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | **15** | **16** | **17** | **18** | **19** | **20** | **21** | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 |

We see that there are seven symbol errors within a codeword and the (7, 3) RS code cannot correct for this many symbol errors. Let us block interleave the symbol stream this time around:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 8 | 15 | 22 | 29 | 2 | 9 | 16 | 23 | 30 | 3 | 10 | 17 | 24 | **31** | **4** | **11** | **18** | **25** | **32** | **5** | 12 | 19 | 26 | 33 | 6 | 13 | 20 | 27 | 34 | 7 | 14 | 21 | 28 | 35 |

Let us now de-interleave the symbol stream:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | **4** | **5** | 6 | 7 | 8 | 9 | 10 | **11** | 12 | 13 | 14 | 15 | 16 | 17 | **18** | 19 | 20 | 21 | 22 | 23 | **24** | **25** | 26 | 27 | 28 | 29 | 30 | **31** | **32** | 33 | 34 | 35 |

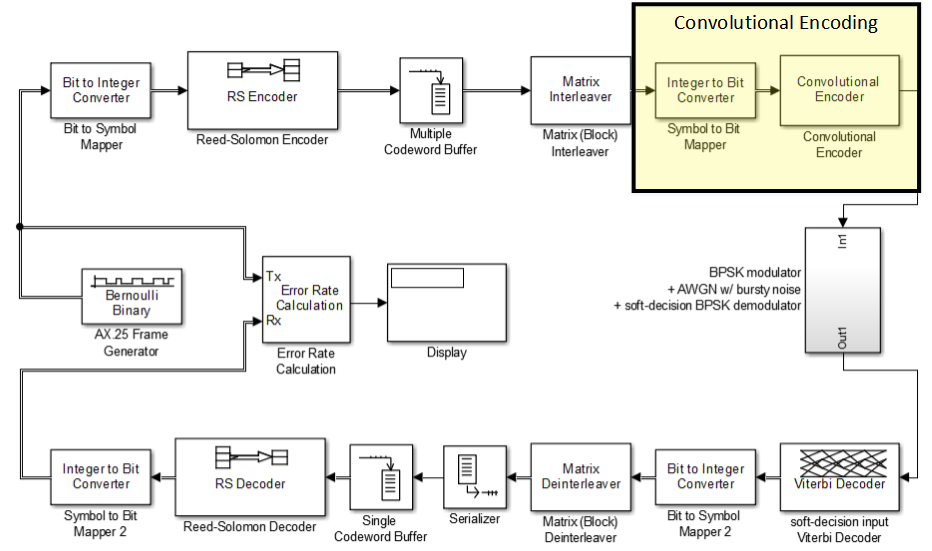
We can see now that no single codeword consists of more than two symbol errors. Hence, the (7, 3) RS code would succeed in correcting for the entire burst error. One can easily imagine how increasing the interleaver depth can spread burst errors apart even further. In general, if a propagation channel causes *b* symbol errors in contiguous fashion, then the interleaver depth (*d*) is calculated as follows:

where *t* is the maximum number of symbol errors within a single codeword that a given RS code can correct.

However, the disadvantage of increasing the interleaving depth is that besides from using slightly more hardware resources, the time required by *Multiple Codeword Buffer* for filling the block matrix increases as well. The higher the interleaving depth, the higher the delay in the digital communication system. This senior design team must be cognizant of this during the design of System C.

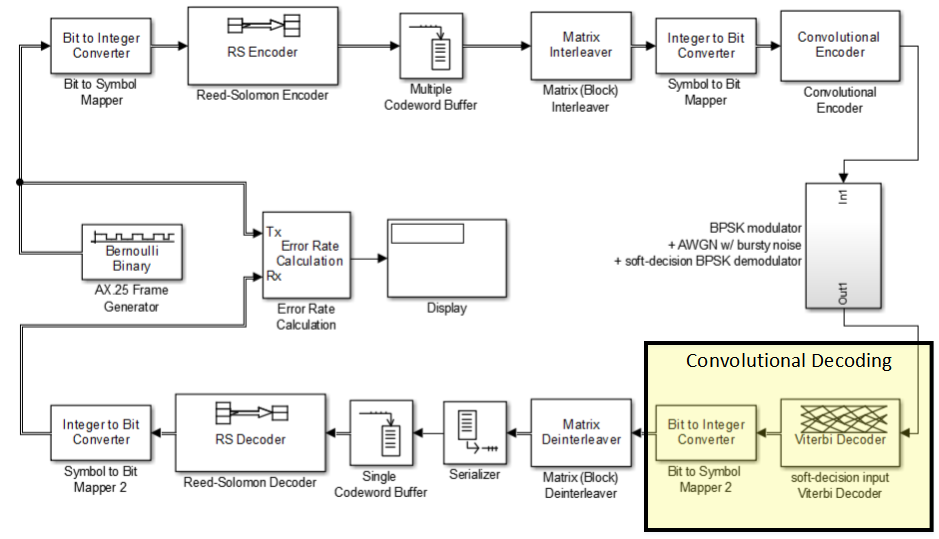
**Convolutional Encoding**

**[TODO (TBC by 12/02/13): A thorough, high-level explanation of the general (n, k, L) convolutional code will be discussed here.]**

****

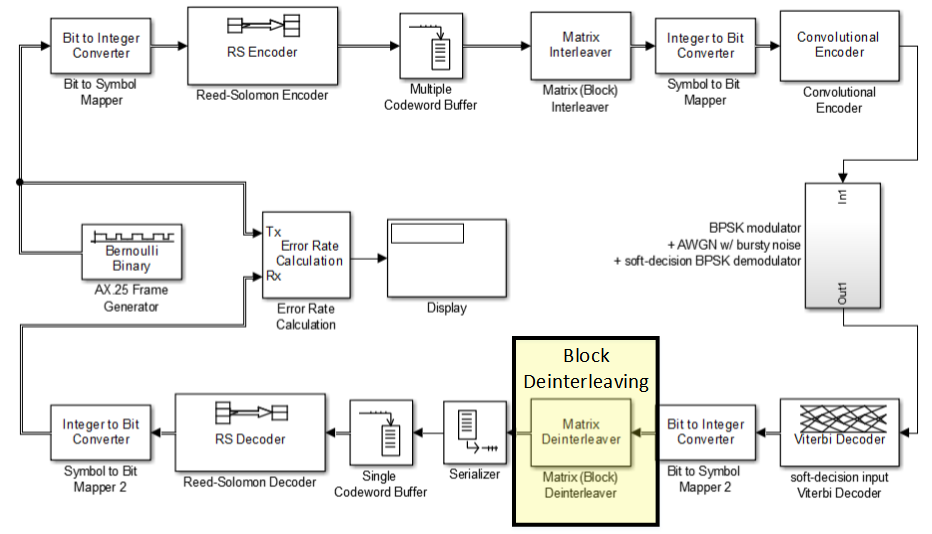
**Convolutional Decoding**

**[TODO (TBC by 12/02/13): Since the discussion of the FEC engine functionality is limited to high-level descriptions only, this section will emphasize the inverse functionality provided in the Convolutional Encoding section. However, it is necessary to provide reasoning for using soft-decision decoding instead of hard-decision decoding in the BPSK demodulator. That reasoning will be discussed in this section.]**

****

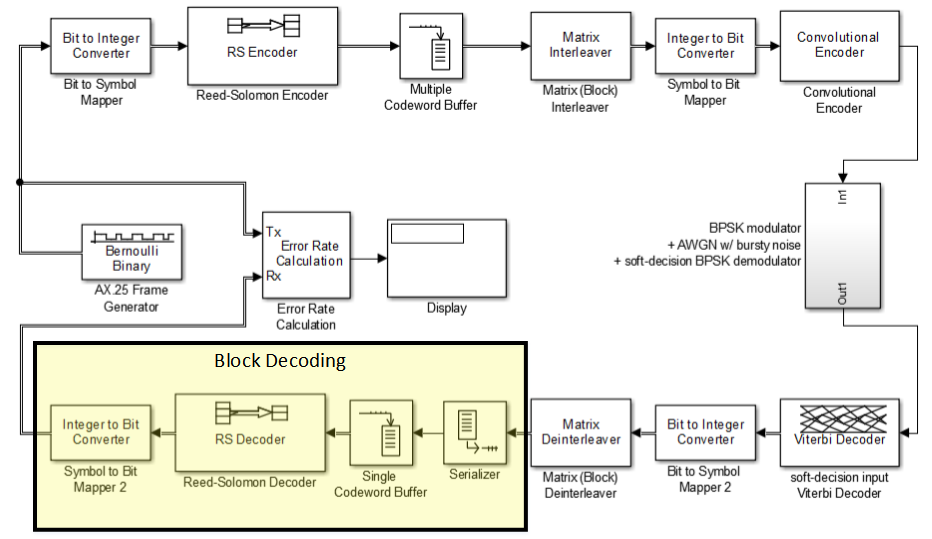
**Block De-interleaving**

**[TODO (TBC by 12/02/13): This section will literally emphasize the fact that the block de-interleaver performs the inverse functionality of the block interleaver. In fact, the examples provided in the Block Interleaving section already hinted as to how block de-interleaving works.]**

****

**Block Decoding**

**[TODO (TBC by 12/02/13): This section will emphasize high-level inverse functionality of the Block Coding section. The Serializer and Single Codeword Buffer will be briefly explained.]**

****

## Hardware Implementation using ISE Design Suite

# EVALUATION

Deep space and satellite communication links are riddled with random errors across a very wide bandwidth (Nguyen, et. al, 2009). In addition to random errors in the satellite link, bursts of noise can corrupt an entire segment of a link resulting in burst errors (Murphy, et. al, 1994). These channel imperfections are common in satellite communications and are modeled very well by the additive white Gaussian noise (AWGN) channel (Viswanathan, 2013). The AWGN channel is a random noise channel that makes a communication link vulnerable to random bit errors and burst errors. In general, it is understood that AWGN provides maximum bit corruption and compared to other channel models, systems that perform the best in AWGN perform the best in real-life situations (Viswanathan, 2013). Hence, this senior design project will rely solely on the AWGN channel (see Section 1.3.6) to represent the propagation medium for our three amateur radio satellite telemetry systems.

We implement the bit error rate tester (BERT) in software. The BERT consists of an AX.25 packet generation program written by us, a custom AX.25 packet comparison program written by us, and an available virtual serial terminal interface (with data logging capabilities). The BERT provides several performance metrics based off of bit error rate (BER) and packet error rate (PER). Please refer to Section 3 (Approach) for the implementation of this BERT and how it interfaces with the external FPGA board.

# SUMMARY AND FUTURE WORK

# ACKNOWLEDGEMENTS

# REFERENCES

Alminde, L., Bisgaard, M., Vinther, D., Viscor, T., & Ostergaard, K. Z. (2002). *Robustness of radio link between AAU-cubesat and ground station.*Unpublished manuscript.

Capitaine, T., Barrandon, L., Bourny, V., Senlis, J., Le Mortellec, A., Astier, R., et al. (2010). Robust satellite AX25 frames demodulation. Paper presented at the *The Small Satellites Systems and Services (4S) Symposium, , 31.*

Davis, J. (2010). Amateur radio's lost future. Message posted to <http://www.amateurradio.com/ham-radios-lost-future/>

de Milliano, M., & Verhoeven, C. (2010). Towards the next generation of nanosatellite communication systems.*Acta Astronautica, 66*(9–10), 1425-1433. doi:<http://dx.doi.org.libproxy.temple.edu/10.1016/j.actaastro.2009.10.034>

Goode, S. (1984). BER performance of TAPR TNC modem.*Packet Status Register,*(11), 14-15.

Hsiao, F., Liu, H., & Hsieh, S. (2000). Using forward error correction technique for microsatellite data broadcasting system.*Acta Astronautica, 46*(2–6), 203-211. doi:<http://dx.doi.org.libproxy.temple.edu/10.1016/S0094-5765(99)00224-6>

Karn, P. (1994). *Toward new link layer protocols.* Retrieved 11/01, 2013, from <http://www.qsl.net/n9zia/newlinkpaper.html>

Karn, P. (2011). *The BPSK1000 telemetry modem for ArriSSat-1.* Retrieved 11/01, 2013, from <http://www.ka9q.net/bpsk1000.html>

Lindsey, W., & Simon, M. K. (1977). Detection of digital FSK and PSK using a first-order phase-locked loop.*Communications, IEEE Transactions on, 25*(2), 200-214. doi:10.1109/TCOM.1977.1093804

Murphy, J., Chow, E., & Markley, R. (1994). ATM service-based selective retransmission over DSN satellite links. Paper presented at the *15th AIAA International Communication Satellite Systems Conference, San Diego, USA, , 350.*

Nguyen, H., & Shwedyk, E. (2009). *A first course in digital communications* (1st ed.). Cambridge, UK: Cambridge University Press.

Rao, K. H. S., Jamadagni, K. S., Von Allmen, L. A., & Shah, A. V. (1990). All-digital pseudo-coherent (PC) FSK modems. Paper presented at the *Digital Communications, 1990. Electronic Circuits and Systems for Communications. Proceedings, 1990 International Zurich Seminar on,*463-473. doi:10.1109/DIGCOM.1990.129390

Roelofs, E. (1987). a fsk modem for the map protocol.*Report of the Practical,*

Sklar, B. (2001). *Digital communications: Fundamentals and applications* (2nd ed.). New Jersey: Prentice Hall.

Viswanathan, M. (2013). In Mathuranathan V. (Ed.), *Simulation of digital communication systems using matlab* (2nd ed.). GaussianWaves: Viswanathan, Mathuranathan;.

Wallio, R. *AX.25 packet radio AFSK on FM Bit, packet and data set error rates.* Retrieved 11/07, 2013, from <http://showcase.netins.net/web/wallio/BER_Packetradiobiterrorrate.html>

1. Product SPECIFICATION
2. SOME INTERESTING RELEVANT DERIVATION