FX.25

Forward Error Correction Extension to AX.25 Link Protocol For Amateur Packet Radio

Version: 0.01 DRAFT
Date: 01 September 2006

Authors:
Jim McGuire, KB3MPL
Ivan Galysh, KD4HBO
Kevin Doherty
Hank Heidt, N4AFL
Dave Neimi

Revision History

v 0.01.06	Clarify Layer-2-ness of FX.25 frame.
v 0.01.05	Updates to body text. Fix place-holders in Correlation Tag table. Fix direction arrow errors on Gold Code drawing. Finalize FEC code/tag assignments. J. McGuire
v 0.01.04-	Initial document releases for review (changes not tracked … well.) J. McGuire

Overview

AX.25[1] packet radio communications exist for myriad application from remote weather monitoring to APRS position reporting. The AX.25 protocol is well established, but has poor performance in environments where the communications channel is error-prone. A single bit error will cause the AX.25 frame's 16-bit CRC to be invalid, resulting in the entire packet being discarded. Error detection and recovery mechanisms are implemented at higher levels of the AX.25 protocol, and rely on retransmission requests for damaged or missing packets. Retransmission requests in a noisy environment are problematic - the retransmitted packet may be errored as well. Retransmission is not feasible in communication environments that are fundamentally unidirectional – beacon telemetry and multicast structures are two examples.

The FX.25 extension to AX.25 implements a Forward Error Correction (FEC) "wrapper" around a standard AX.25 packet. The FX.25 error correction capability shifts a portion of the error correction process toward the bottom of Layer 2 (Data Link) in the OSI protocol stack[2], reducing the need for retransmission requests and increasing channel throughput in unidirectional environments.

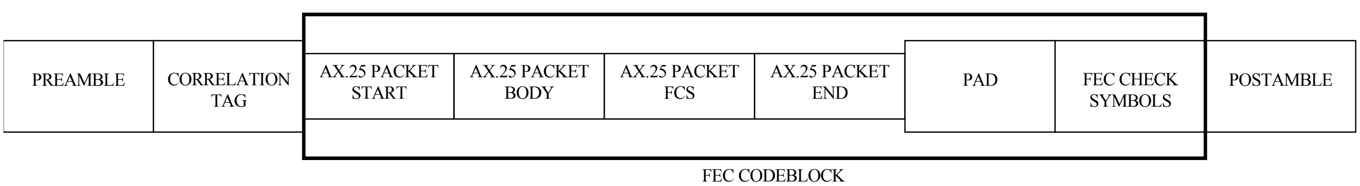
Interoperability with existing systems is a key requirement. The FX.25 wrapper is designed to supplement the existing AX.25 infrastructure without displacing it. The FX.25 signal structure allows reception with a standard AX.25 receiver, albeit without the benefit of the FEC correction functions. An AX.25 receiver interprets the additional FEC information as channel noise.

The FX.25 frame structure is compatible with both connectionless and connection-oriented networks.

Protocol Summary

The FX.25 frame structure encapsulates the AX.25 packet element, and does not duplicate services provided by the AX.25 protocol. Special consideration is given to the amateur radio environment, where communications channels are rarely "error free."

The basic structure of the FX.25 frame is shown below.



The Preamble block is a sequence of 0x7E bytes intended to allow the receiver to acquire the signal.

The Correlation Tag is an 8-byte (64-bit) fixed sequence designed to indicate the start of a packet. Different Correlation Tags have been selected to indicate which FEC algorithm is being applied.

The FEC Codeblock is the data space over which the FEC algorithm is applied. The Correlation Tag is outside this boundary because the Correlation Tag is designed to provide good correlation even in the presence of channel errors. The FEC Codeblock size is dependent on the block requirements of the FEC algorithm being implemented. Details of the FEC algorithms are located in the appropriate sections of this document.

The AX.25 packet is specification-compliant, unmodified, and complete. For compatibility with legacy equipment, all necessary bit-stuffing and formatting is applied to the AX.25 packet elements as if the FX.25

wrapper was not being transmitted. If the FX.25 frame structure is removed from the AX.25 packet, the AX.25 elements shall present a valid and complete AX.25 packet to any receiving equipment. As bit-stuffing at the AX.25 level can expand the size of the AX.25 transmitted datastream beyond that of the raw data, consideration needs to be given to the size of the resultant packet relative to the FX.25 frame. Any AX.25 Segmentation and Reassembly (SAR) functions need to be aware of the post-AX.25-bit-stuffed packet data size, and not simply the raw information size. (This is a fundamental limitation of the AX.25 frame delineation process.)

The Pad block is necessary to build-out the FEC Codeblock to the number of bytes required by the FEC algorithm.

The FEC Check Symbols are applied at the end of the FEC Codeblock. The number of FEC Check Symbols is dependent on the FEC algorithm selected. All other symbols inside the FEC Codeblock area are Information Symbols.

The Postamble block is a series of 0x7E bytes intended to provide separation between the end of the FX.25 frame and the transmitter un-key event.

Physical Layer Considerations

AX.25 packets may be transmitted over a variety of physical links. The FX.25 frame structure attempts to interoperate with existing transmission formats. However, a number of legacy AX.25 structures complicate matters. The bit-stuffing requirement of AX.25 complicates mapping the legacy data into the FX.25 frame structure. The bit-stuffed AX.25 packet size is data dependent, and additional steps must be taken in order to ensure that the bit-stuffed AX.25 packet maps properly into the FX.25 frame structure.

AX.25 (and it's parent X.25[3]) is defined as a bit-serial protocol. The demarcation point between Layer 2 and Layer 1 is the bit-serial data stream. Serial bit data-rate and bit order-of-transmission are therefore promoted to Layer 2 entities.

All bytes/octets in the FX.25 frame are transmitted LSb first. This includes all bytes in the AX.25 packet. Since the AX.25 specification requires that the FCS bytes be transmitted MSb first, it is required that the AX.25 FCS bytes be mapped into the FX.25 frame in reverse-bit order. This requirement is in-place because the AX.25 bit-stuffing requirement creates a data-dependent mapping of the data from the AX.25 packet into the FX.25 frame space. The proper order of operations for mapping the AX.25 packet is:

- Create raw AX.25 packet
- Calculate the FCS value over the AX.25 packet data space
- Reorient (flip) the FCS bytes for proper order of transmission - all LSb-first
- Bit-stuff the AX.25 packet
- Perform SAR, if necessary
- Add Pad bits and Pad bytes to meet FEC algorithm data-size requirements
- Calculate FEC Check Symbol values
- Add Preamble, Correlation Tag, and Postamble bytes
- Hand off to Layer 1:
  - Add data scrambling, if required by transmission channel
  - NRZI encode the data
  - Modulate onto transport medium (i.e. AFSK for 1200bps; GFSK for 9600bps; etc.)

It should be noted that the AX.25 bit-stuffing algorithm is not appropriate to use at the FX.25 frame level. FEC is a block-based algorithm, and requires a fixed number of bits in the block. FEC can detect and correct relatively large numbers of errors, but only if the block size is correct and the FEC Check Symbols are in the correct locations in the frame. A single bit error that would cause the AX.25 bit-stuffing function to add or remove a single bit would invalidate the FEC correction capabilities. The AX.25 bit-stuffing method is mathematically "weak," and is not recommended for use outside the legacy AX.25 packet. If additional data transitions are desired at the Layer 1 level, frame-synchronous LFSR-based scramblers would be a better choice. However, Layer 1 physical transport issues are beyond the scope of this document.

If additional Physical Layer line-coding is added to the FX.25 frame, care should be taken to guarantee that the additional structures do not invalidate the error correction capabilities of the FEC algorithm.

Preamble Details

The Preamble is a series of 0x7E flag bytes. The Preamble serves to provide a valid signal while the receiving equipment can break squelch, perform RF AGC, threshold the data, and perform clock recovery. A minimum of four bytes should be transmitted. Additional Preamble bytes may be necessary depending on the expected receiver characteristics and the transmitted line rate. Layer 2 and Layer 1 entities should negotiate the Preamble length prior to transmission. Preamble bytes are useful for determining byte alignment when used in conjunction with the Correlation Tag, but may be discarded by the receiver.

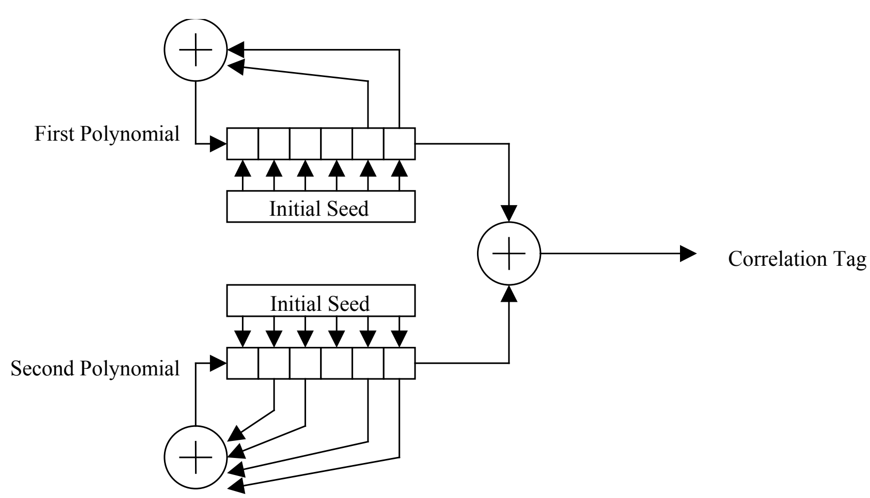
Correlation Tag Details

The Correlation Tag provides a reliable "start of frame" indication in a noisy environment. Desirable characteristics of a Correlation Tag are high auto-correlation and low cross-correlation. Gold Codes, developed by Robert Gold, possess pseudorandom properties ensuring favorable auto- and cross-correlation characteristics. Gold Codes are also simple to generate.

Eight bytes are allocated to the Correlation Tag. A 63-bit code provides good correlation characteristics while maintaining minimal bandwidth utilization desired in amateur radio applications. A leading zero is transmitted at the beginning of the Correlation Tag to bring the total bit count up to 64-bits.

Gold codes are derived from maximal-length PRBS sequences called m-sequences[4]. These sequences are generated by LFSR techniques, implemented in either hardware or software. Two different m-sequence polynomials are combined to create a Gold Code. The two sequences are defined over an 8-bit Galois Field GF(2^8) with the following polynomials[5]:

First Polynomial: $I(x) = x^6 + x^5$
Second Polynomial: $Q(x) = x^6 + x^5 + x^3 + x^2$



By fixing the initial seed of the first polynomial to 0x3F and varying the second polynomial seed from 0x01 through 0x3F, one obtains 2^r-1 = 63 distinct Gold Codes. Additionally, the two zero-seed codes (which reduce to either the first or second polynomial alone) generate valid Gold Codes. The two zero-seed codes are reserved for future expansion of this specification.

A family of Correlation Tags is thus defined, and is indexed by the initial seed of the second polynomial generator.

Tag_00	0x566ED2717946107E	<reserved, first seed = 0x3F, second seed = 0x00>
Tag_01	0xB74DB7DF8A532F3E	<first seed = 0x3F, second seed = 0x01>
Tag_02	0x26FF60A600CC8FDE	
…		
Tag_3E	0xF68FF114D7B50540	
Tag_3F	0x17AC94BA24A03A00	<first seed = 0x3F, second seed = 0x3F>
Tag_40	0x41C246CB5DE62A7E	<reserved, first seed = 0x00, second seed = 0x3F>

The above Correlation Tag values are represented in 64-bit notation, with the MSB at the left and the LSB at the right. Transmission order of the bytes for Tag_01 would be 0x3E 0x2F 0x53 0x8A 0xDF 0xB7 0x4D 0xB7.

AX.25 Packet Requirements

The AX.25 packet is required to meet certain minimum specifications when mapped into the FX.25 frame.
- The maximum length of the bit-stuffed AX.25 packet is determined by the FEC algorithm selected.
- A minimum of one 0x7E flag byte must be included at the start of the AX.25 packet
- A minimum of one 0x7E flag byte must be included at the end of the AX.25 packet

The AX.25 packet is constructed prior to mapping into the payload space of the FX.25 frame, and is done so without regard to the FX.25 byte boundaries. Bit-to-byte alignment is handled by the Pad block.

The FX.25 frame is designed to be compatible with legacy AX.25 packets. Additional 0x7E flag bytes may be included before or after the minimum AX.25 packet data-set, as long as the maximum packet length restriction is met. Any additional flag bytes are added in compliance with the AX.25 specification, and are only constrained by the FX.25 FEC data space.

Pad Requirements

The Pad block serves two functions: a) provide bit-to-byte alignment of data after the AX.25 packet; and b) fill-out the FEC Codeblock to the number of bytes required by the FEC algorithm. Most FEC algorithms require a fixed size data block. The Pad size will be different for each FX.25 frame transmitted. The Pad field carries no data, and is discarded at the receive end of the link.

The recommended Pad byte value is 0x7E. Consecutive 0x7E bytes constitute "null" frames per the AX.25 specification, and are discarded without generating errors. This particular value also allows some flexibility with regard to the Start and End flags within the AX.25 packet.

In situations where the AX.25 packet occupies a non-integral number of bytes within the FX.25 frame, the Pad block shall insert a fraction of the 0x7E byte into the remainder of the byte-position that terminates the final AX.25 Packet End Flag. The fractional portion of the Pad byte shall be aligned with the MSb position of the FX.25 frame byte boundaries (i.e. truncate the LSb portions.) This fractional truncation requirement maintains the integrity of the final byte in the AX.25 least-significant-bit-first transmission architecture.

FEC Algorithms

For the initial release of this document, Reed Solomon (RS) FEC coding has been selected. RS codes have been used in terrestrial and radio telecommunications environments for many years with good results. The SCAMP[6] and RDFT[7] transfer protocols use RS algorithms, as do spacecraft compliant with the CDSDS Space Telemetry Coding[8] standard.

Tag	Correlation Tag Value	FEC coding algorithm, number of information bytes available
Tag_00	0x566ED2717946107E	Reserved
Tag_01	0xB74DB7DF8A532F3E	RS(255, 239) 16-byte check value, 239 information bytes
Tag_02	0x26FF60A600CC8FDE	RS(144,128) - shortened RS(255, 239), 128 info bytes
Tag_03	0xC7DC0508F3D9B09E	RS(80,64) - shortened RS(255, 239), 64 info bytes
Tag_04	0x8F056EB4369660EE	RS(48,32) - shortened RS(255, 239), 32 info bytes
Tag_05	0x6E260B1AC5835FAE	RS(255, 223) 32-byte check value, 223 information bytes
Tag_06	0xFF94DC634F1CFF4E	RS(160,128) - shortened RS(255, 223), 128 info bytes
Tag_07	0x1EB7B9CDBC09C00E	RS(96,64) - shortened RS(255, 223), 64 info bytes
Tag_08	0xDBF869BD2DBB1776	RS(64,32) - shortened RS(255, 223), 32 info bytes
Tag_09	0x3ADB0C13DEAE2836	RS(255, 191) 64-byte check value, 191 information bytes
Tag_0A	0xAB69DB6A543188D6	RS(192, 128) - shortened RS(255, 191), 128 info bytes
Tag_0B	0x4A4ABEC4A724B796	RS(128, 64) - shortened RS(255, 191), 64 info bytes
Tag_0C	0x0293D578626B67E6	Undefined
Tag_0D	0xE3B0B0D6917E58A6	Undefined
Tag_0E	0x720267AF1BE1F846	Undefined
Tag_0F	0x93210201E8F4C706	Undefined
Tag_10 thru Tag_3F		Undefined
Tag_40	0x41C246CB5DE62A7E	Reserved

Table 1 - FEC Algorithms and Correlation Tag Value Assignments

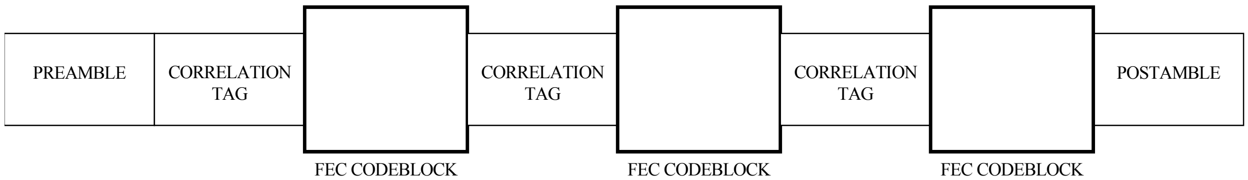
Selection of the “proper” FEC code to apply to a particular packet stream is highly dependent on the characteristics of the transmission channel.

Postamble Requirements

The Postamble is a series of 0x7E flag bytes. The Postamble serves to provide a valid signal separating the final FX.25 frame bytes from the transmitter un-key event. A minimum of two bytes should be transmitted. Additional Postamble bytes may be necessary depending on the expected receiver characteristics and the transmitted line rate. Layer 2 and Layer 1 entities should negotiate the Postamble length prior to transmission. Postamble bytes are discarded at the receiver.

Multi-Frame Blocks

Situations exist where it is desirable to transmit multiple AX.25 packets in a single burst - large payload information blocks, inter-repeater back-haul links, etc. In these cases, it is inefficient (though valid) to transmit each FX.25 frame as an individual entity. FX.25 frames may be concatenated as a single transmission unit, delineated by a single correlation tag separator.



Within a Mulit-Frame block, the correlation tag value should remain constant.

Future Developments

This document has a strong legacy-support characteristic. However the Correlation Tag feature provides an indexed table into undefined payload mappings for the FX.25 frame structure.

Future implementations of the FX.25 specification may consider eliminating the bit-stuffing and FCS-transmission-reversal liabilities of the AX.25 packet protocol. Replacement with control-escape octet values and a frame-synchronous scrambler may be a viable path to pursue, but would do so at the expense of interoperability with existing infrastructure. Such a development is beyond the scope of this document at this time.

Glossary

ITU International Telecommunication Union
ITU-R International Telecommunication Union, Radio Communication
ITU-T International Telecommunication Union, Telecom Standardization

LFSR Linear Feedback Shift Register
LSb Least Significant Bit
LSB Least Significant Byte

MSb Most Significant Bit
MSB Most Significant Byte

OSI Open Systems Interconnection

PRBS Pseudo Random Bit Sequence

SAR Segmentation and Reassembly

References

[1] AX.25 Link Access Protocol for Amateur Packet Radio, Version 2.2, Revision July 1998
http://www.tapr.org/pdf/AX25.2.2.pdf

[2] OSI Model, from Wikipedia
http://en.wikipedia.org/wiki/OSI_model

[3] ITU-T Recommendation X.25 "INTERFACE BETWEEN DATA TERMINAL EQUIPMENT (DTE) AND DATA CIRCUIT TERMINATING EQUIPMENT (DCE) FOR TERMINALS OPERATING IN THE PACKET MODE AND CONNECTED TO PUBLIC DATA NETWORKS BY DEDICATED CIRCUIT"
International Telecommunication Union, 03/93

[4] Tapani Ristaniemi, "Spread Spectrum Techniques, Chapter 2: Pseudorandom Sequences"
http://www.cs.tut.fi/kurssit/TLT-5606/lect2.pdf

[5] Esmael H. Dinan and Bijan Jabbari, "Spreading Codes for Direct Sequence CDMA and Wideband CDMA Cellular Networks"
IEEE Communications Magazine, September 1998

[6] Muething, "SCAMP Protocol Specification … An Amateur Message Protocol based on RDFT"
Draft Specification Rev Q: Feb 1, 2005

[7] Sanderson, Wyman, et al "Redundant File Transfer Protocol"
Multiple presentations at Dayton Hamvention
http://www.svs.net/wyman/examples/hdsstv/index.html

[8] Consultative Committee for Space Data Systems, "Telemetry Channel Coding"
CCSDS 101.0-B-6, BLUE BOOK, October 2002