Technical Specifications for KD5HIO Software

Version 0.2 12/12/2000
by Glen Hansen, KD5HIO

HamScope Forward Error Correction Algorithms

HamScope is a terminal program designed to support multi-mode digital Amateur radio communications using a computer soundcard. This software is designed to operate on computers equipped with Microsoft Windows 95, and offers many of the features that are available with similar programs available to an Amateur radio operator, such as MixW[1], Digipan[2], P31sbw[3], and WinPSK[4]. HamScope supports the reception and transmission of popular Amateur modes, including PSK31 (BPSK and QPSK), CW, RTTY, and ASCII [5]. Additionally, HamScope supports a STREAM compatible MFSK16 [9] mode.

HamScope further provides various forward error correction (FEC) capabilities, in the same vein as the Viterbi convolutional encoding approach used in the existing QPSK mode available in many PSK31 terminal programs. This document is intended to detail the FEC approaches implemented in HamScope, both to satisfy the pertinent FCC regulations (FCC Part 97, Subpart D, 97.309(a)(4)) prior to using these modes over the air, and to encourage further experimentation on FEC technology for digital communications.

HamScope was designed to provide an experimental tool for the study of four forward error correction protocols under adverse communications conditions. These FEC protcols are:

- An interleaved, (3, 1) redundant mode (3N).
- Interleaved, (24, 12) Golay encoding
- Interleaved, (216, 64) Turbo coding
- Interleaved Reed-Solomon encoding

With this stated purpose, existing communications channels were used with minor modifications to support the FEC protocols. The above protocols are supported fully using BPSK, QPSK, FSK16, and ASCII; with 3N and Reed-Solomon provided for RTTY modulation. The PSK FEC methods maintain the usual 31.25 Hz bandwidth of the base methods, along with sharing many of the signal and modulation characteristics. The nature of the bitstream modulating the carrier is different to support the FEC modes. The same is true for the FEC ASCII and RTTY modes. The FSK16 mode (HFSK16) uses a similar channel as the MFSK16 mode (sharing the Gray code basis and 16 tones separated by 15.625 Hz tone spacing. Due to interleaving and encoding operations to support the particular redundancy required by the desired mode, reception using ordinary means is generally not possible. It is necessary to stress that this redundancy and interleaving operation was performed to support quality communications under adverse conditions, not to obscure the meaning of the transmitted information. By implementing the details of these FEC approaches described herein, it is straightforward to develop means to demodulate any of these signals.

This document is structured to detail the signal channel of these modes, by describing how the transmitted bitstream is constructed.

FEC Mode Transmission

Prior to the transmission of the digital FEC data, four steps are required to prepare and encode the data:

- Block register preparation
- Encoding
- Interleaving
- Filling the channel

The block register contains the input to the FEC encoder, and may be viewed as a sequential bitstream or bit array. It's length is a function of both the block length of the encoder and the fundamental word size of the FEC mode. The encoders for each of the 8-bit native modes (BPSK, QPSK, HFSK16, and ASCII) are:

- 3N, a (3, 1) code with an input block length of 8 words
- Golay, a (24, 12) code with an input block length of 9 words
- Turbo, a (216, 64) code with an input block length of 8 words
- Reed-Solomon (RS), a (15, 8) code with an input block length of 4 words

The block register used for data input to the FEC encoder for the four modes, BPSK, QPSK, HFSK16, and ASCII, can also be viewed as a byte array. This array is *n* 8 bit bytes in length, where *n* is 8, 9, 8, and 4 for 3N, Golay, Turbo, and RS, respectively.

For these four modes, the block register is filled with the 7 bit ASCII character representation and an additional parity bit to create even parity symbols. The seven ASCII bits that represent the character are stored in the lower 7 bit positions of the byte, and the even parity bit is stored in the 8^{th} position. The resulting parity symbols are stored in the byte array that forms the block register, with *n* symbols stored in the register in a first-in-first-out (FIFO) format. The bits are stored in the block register bytes in a least-significant bit (LSB)-first format.

Filling the block register for the RTTY FEC modes that are supported in HamScope is accomplished in a similar manner. The modes supported for RTTY modulation are:

- 3N, a (3, 1) code with an input block length of 8, 5 bit Baudot symbols
- Reed-Solomon, a (31, 16) code with an input block length of 16, 5 bit Baudot symbols

In the case of RTTY, the block register holds *n* 5 bit Baudot symbols, where *n* is 8 and 16 for 3N and RS modes, respectively. Again, the Baudot bits are stored in the block register in LSB-first format.

Bit Interleaving

HamScope is designed to employ bit interleaving to support the randomization requirements for Turbo encoding, and to provide a temporal information spread to minimize the effects of burst noise error. The interleaving form is a straightforward (*k*, *n*) operation, where a data buffer of length *k* bits is stored in a two-dimensional bit array composed of *n* columns and *k*/*n* rows of data. The interleaving operation stores the data into the array by filling each row of data, and reads the data from the array by traversing down the columns. This operation is reversed to deinterleave the data on the receiving end of the channel.

3N Interleaved FEC

For the ASCII, HFSK16, BPSK, and QPSK modes, the 3N method is based on lengthening the block input register by a factor of 3 through the creation of redundant parity information. In this case, the block register information (the 8, 8 bit ASCII words with parity), is duplicated twice and postpended to the end of the current block register data. For example, immediately following the 8 data bytes in the original data stream would be a duplicate of these 8 bytes in identical order. Immediately following these duplicated bytes, would be the final set of 8 bytes, again an exact duplicate of the original information bytes. This operation results in a block register of length 8 * 3, or 24 bytes long. This register is then (216, 8) interleaved in place. Following the interleaving process, a pair of sync bytes are added to the block register to allow window detection by the receiving process. A sync byte of 0x00 is prepended to the block register, with a sync byte of 0xFF postpended to the register. These sync marks may be treated as a byte "space" and "mark" symbol by the receiving process to allow detection and block deinterleaving.

(24, 12) Golay FEC

Golay encoding for the ASCII, HFSK16, BPSK, and QPSK modes is based on 12 bit segments of the block register. This algorithm selects data windows consisting of 12 bit data segments, from the beginning to the end of the 9, 8 bit word register, processing a total of 6, 12 bit segments. Each segment is copied from the block register to temporary storage, and Golay encoded [6] to form a 23 bit codeword. The 24th bit of the codeword is an even parity bit based on the previous 23 bits. This process is repeated to form a set of 24 bit codewords (6 of them). This set is then recombined, resulting in a new 144 bit block register. This block register is then (144, 8) interleaved, with a sync byte of 0x00 is prepended and a sync byte of 0xFF postpended to form the final complete block register.

(216, 64) Turbo FEC

Turbo encoding for the ASCII, HFSK16, BPSK, and QPSK modes is based on combining the 8 original ASCII data bytes with two parity streams derived by encoding the original 8 data bytes. A new block register is formed by copying the original 8 data bytes to the beginning of a new bit array. To this result, a null byte is added to form a 9 byte data stream. The first stream of parity bits is formed by Turbo encoding [7] the original 8 byte data stream and postpending the encoded stream at the end of the original data in the new block register. This encoded parity stream may be as long as 9 bytes due to the requirement of suitably terminating the first Turbo encoder step. The unused portion of the 9th byte is filled will null data. At this point, the block register consists of 18 total bytes of data, the original data followed by the encoded Turbo parity information. To complete the encoding, the original data stream is (72, 8) interleaved (a null byte is again postpended to the data to form a 9 byte data stream), and run through the Turbo encoder with the termination state padding function disabled. This result is postpended to the block register to form an encoded 27 byte data stream. This stream has a sync byte of 0x00 prepended, and a sync byte of 0xFF postpended, to form the final block register.

(15, 8) Reed-Solomon FEC

RS encoding for the ASCII, HFSK16, BPSK, and QPSK modes is based on splitting the 4, 8 bit data bytes into 8, 4 bit segments. Each of these 4 bit segments, in turn, is fed into the (15, 8) Reed-Solomon [8] encoder as a RS symbol. The RS encoder returns a 15 symbol (4 bits per symbol) data stream, which is used to fill a new block register; the first 8, 4 bit symbols is the original data derived from splitting the input ASCII (with parity) data. The remaining 7, 4 bit symbols is RS parity information. To this block register, a final null symbol 4 bits long is added, resulting in a block register of 16, 4 bit symbols (or 8, 8 bit symbols). This register is then (64, 8) interleaved. Again a sync byte of 0x00 is prepended, with a sync byte of 0xFF postpended, to form the final block register.

Transmission and Modulation

Given the final form of the block register, with the appropriate prepended and postpended sync bytes, it is possible to pass the block register as a bit stream to traditional BPSK, QPSK, and ASCII modulators.

For BPSK and QPSK modes, modulation is straightforward. The prepended and postpended sync bytes are used to isolate the interleaved block segments upon reception, and also serve to sync the detector to the current state of the bitstream. For BPSK mode, serial bit by bit transmission is used, starting at the beginning sync byte that leads the block register. The block register is divided into 8 bit symbols, and each symbol is sent to the BPSK modulator most-significant bit (MSB)-first. The modulator converts the boolean state of each bit to a phase change value, with a zero (0) sent as an 180 degree phase change and a one (1) results in no phase change in the modulator. The signal results from a traditional BPSK modulation algorithm, with only the information modulating stream being changed through the encoding process. As

such, the traditional BPSK ramps during phase changes are preserved, along with the 31.25 Hz bandwidth. Indeed, a BPSK signal modulated with one of the above FEC strategies is difficult to tell, qualitatively, from a traditional BPSK signal. The interleaving and encoding process employed in the chosen FEC mode rearranges the transmitted bits, however, so it is not possible to directly extract the information stream without employing the de-interleaving process to the detected data.

The QPSK algorithm is very similar, but each 8 bit symbol of the block register is further subdivided into 4, 2 bit symbols. These 4, 2 bit symbols are sent to the modulator MSB first. With QPSK modulation, there are four possible symbol bit pairs and resulting states:

- Bit pair 0x00 (00), shift signal phase 180 degrees
- Bit pair 0x01 (01), shift signal phase +90 degrees
- Bit pair 0x02 (10), shift signal phase –90 degrees
- Bit pair 0x03 (11), do not change phase

Upon reception, the prepended and postpended sync bytes are used to synchronize the block deinterleaver and set the QPSK detector to the appropriate state. Pairwise detection is used to convert from the observed phase change to bit pairs to reconstruct the interleaved data stream. Again, the signal characteristics of a given QPSK FEC mode are very similar to native QPSK, possessing similar bandwidth and modulation qualities.

For ASCII mode modulation, the block register is split into *n* 8 bit segments. These segments have a space symbol (0) prepended to the data, sent LSB first, with a mark symbol (1) postpending the data stream. The mark symbol is double the duration of the space and data symbols, as is found in traditional ASCII modulation [5]. Frequency shift keying is used to send the mark and space bits, along with the data bits from the block register. In this case, the receive detector is synchronized using the mark and space bits, and the sync bytes are used solely to synchronize the block deinterleaver. The modulation characteristics are again very similar to traditional ASCII digital data transmission; only the information being sent on the data stream is different due to the encoding and interleaving processes. At a given baud rate, the process of FEC corrected ASCII and traditional ASCII communication effectively differs only due to the detection of the signal information being impossible without deinterleaving and decoding of the transmitted data.

For HFSK16 mode modulation, the block register is split into 2*n* 4 bit segments. Each of these 4 bit segments are converted to a tone frequency via a Gray code lookup (as detailed in [9]), and transmitted accordingly as an MFSK16 tone sequence. Again, the prepended 0x00 and postpended 0xFF bytes to the data stream provide for interleaver sync on receive (following the reverse-Gray code detection to a stream of 2*n* * 4 bit symbols).

HFSK16 also supports a raw communications option where FEC is not used. In this mode, each 7 bit ASCII character is placed into an 8 bit buffer. The (unused) high 8^{th} bit of this buffer is then zeroed. The buffer is subsequently split into two, 4 bit segments, then transmitted through the Gray code mechanism described above. Upon receive, the knowledge that every other 4 bit symbol always begins with a zero can be used to sync a simple ASCII extraction mechanism to reform the original 7 bit ASCII character. Due to the lack of FEC and this weak synchronization scheme, this particular mode is effective only under good conditions.

RTTY FEC Modes

HamScope supports a subset of FEC protocols for RTTY transmission. The modes supported are 5 bit versions of 3N and Reed-Solomon encoding.

For 3N interleaving, the 5 bit Baudot block register is fed into a similar redundancy generating algorithm. At the end of the 8, 5 bit Baudot characters contained in the register, two sets of redundant data identical to the transmitted characters are postpended. This block register, now 24, 5 bit words in length, is interleaved

using the (120, 5) algorithm. To this result, a sync "byte" consisting of the last 5 bits of 0x00 is prepended, and a sync "byte" of the last 5 bits of 0xFF is postpended to the block register.

For (31, 16) Reed-Solomon encoding, 16, 5 bit symbols are passed to the RS encoding algorithm. The encoder postpends 15, 5 bit parity symbols to the end of the data stream. Additionally, the last symbol is a null symbol, resulting in a new block register of 32, 5 bit symbols. This block register is then (160, 5) interleaved, with the above sync bytes prepended and postpended to form the final block register.

These two FEC modes pass the block register to the RTTY modulator, which forms a transmitted symbol by beginning with a start bit (0), the 5 bit data word sent MSB first, then a stop bit (1) that is approximately 1.41 times the duration of the start and data bits. Again, this modulation strategy is similar to standard RTTY [5].

References

[1] *MixW* by UT2UZ, Nick Fedoseev, http://tav.kiev.ua/~nick/mixw/mixw.htm or http://users.nais.com/~jaffejim/mixwpage.htm

[2] *Digipan* by KH6TY, Howard Teller, http://members.home.com/hteller/digipan

[3] *P31sbw* by G3PLX, Peter Martinez, http://www.kender.es/~edu/psk31.html

[4] *WinPSK* by AE4JY, Moe Wheatley, http://www.qsl.net/ae4jy

Other information on PSK31 and digital modes can be found at http://www.kender.es/~edu/psk31.html or http://www.qsl.net/kk7lk/psk31.htm

[5] ARRL *Technical Descriptions*, ARRL *HF Digital Handbook*, and ARRL *1999 Handbook for Radio Amateurs*, Chapter 12. The American Radio Relay League, Newington, CT 06111 USA

[6] Golay encoding is discussed in Section 4.3 of Michelson and Levesque, *Error-Control Techniques for Digital Communication*, John Wiley & Sons, New York, NY, USA. Sample code can be viewed at http://imailab-www.iis.u-tokyo.ac.jp/~robert/golay23.c

[7] Turbo encoding is discussed in Chapter 8 of Schlegel, *Trellis Coding*, IEEE Press, New York, NY, USA. W.E. Ryan composed a paper on Turbo Codes that is the basis for the Turbo mode in HamScope, see http://telsat.nmsu.edu/~wryan/turbo2c.ps Sample code can be viewed at http://imailab-www.iis.u-tokyo.ac.jp/~robert/codes.html, file BCJRturbo.tar.

[8] Reed-Solomon encoding is discussed in Secion 6.4 of Michelson and Levesque, *Error-Control Techniques for Digital Communication*, John Wiley & Sons, New York, NY, USA. Theory is also presented at http://drake.ee.washington.edu/~adina/rsc/slide/slide.html Sample code can be viewed at http://people.qualcomm.com/karn/code/fec

[9] Technical details and a summary of the MFSK16 mode are presented at http://www.qsl.net/zl1bpu/MFSK.