

Project	IEEE 802.16 Broadband Wireless Access Working Group < http://ieee802.org/16 >	
Title	LDPC coding for OFDMA PHY	
Date Submitted	2004-08-29	
Source(s)	Brian Classon Yufei Blankenship Motorola	brian.classon@motorola.com yufei.blankenship@motorola.com
	Jerry Kim Panyuh Joo Gyubum Kyung Hongsil Jeong Sang-Hyo Kim Hanju Kim DS Park Jaeho Jeon Samsung	kimjy@samsung.com panyuh@samsung.com gyubum.kyung@samsung.com hongsil.jeong@samsung.com sanghyo7.kim@samsung.com hanju.kim@samsung.com dspark@samsung.com jhjeon@samsung.com
	Eric Jacobsen Bo Xia Intel	eric.a.jacobsen@intel.com bo.xia@intel.com
	Brian Johnson Alek Purkovic Nina Burns Sergey Sukobok Nortel Networks	brjohnso@nortelnetworks.com apurkovi@nortelnetworks.com nburns@nortelnetworks.com ssukobok@nortelnetworks.com
	Dale Hocevar Anuj Batra Texas Instruments	hocevar@ti.com batra@ti.com
	Victor Stolpman Jianzhong (Charlie) Zhang Nico Van Waes Nokia	Victor.Stolpman@nokia.com Charlie.Zhang@nokia.com
	Masoud Olfat Nextel Communications	masoud.olfat@nextel.com
Re:	IEEE P802.16-REVe/D4-2004, ballot #14c	
Abstract	This contribution contains a harmonized LDPC code definition agreed to by 6 of the 8	

companies that provided LDPC proposals to an informal LDPC group. All of the specification material within contribution 373 is included in this contribution (i.e., 373 is a subset of this contribution).

Purpose Complete the LDPC specification text.

Notice This document has been prepared to assist IEEE 802.16. It is offered as a basis for discussion and is not binding on the contributing individual(s) or organization(s). The material in this document is subject to change in form and content after further study. The contributor(s) reserve(s) the right to add, amend or withdraw material contained herein.

Release The contributor grants a free, irrevocable license to the IEEE to incorporate material contained in this contribution, and any modifications thereof, in the creation of an IEEE Standards publication; to copyright in the IEEE's name any IEEE Standards publication even though it may include portions of this contribution; and at the IEEE's sole discretion to permit others to reproduce in whole or in part the resulting IEEE Standards publication. The contributor also acknowledges and accepts that this contribution may be made public by IEEE 802.16.

Patent Policy and Procedures The contributor is familiar with the IEEE 802.16 Patent Policy and Procedures <<http://ieee802.org/16/ipr/patents/policy.html>>, including the statement "IEEE standards may include the known use of patent(s), including patent applications, provided the IEEE receives assurance from the patent holder or applicant with respect to patents essential for compliance with both mandatory and optional portions of the standard." Early disclosure to the Working Group of patent information that might be relevant to the standard is essential to reduce the possibility for delays in the development process and increase the likelihood that the draft publication will be approved for publication. Please notify the Chair <<mailto:chair@wirelessman.org>> as early as possible, in written or electronic form, if patented technology (or technology under patent application) might be incorporated into a draft standard being developed within the IEEE 802.16 Working Group. The Chair will disclose this notification via the IEEE 802.16 web site <<http://ieee802.org/16/ipr/patents/notices>>.

Overview

An informal LDPC group has been working on the goal of achieving consensus on a proposed LDPC code design as an optional advanced code for the OFDMA PHY. Many excellent code designs have been submitted. The codes have been qualitatively and quantitatively characterized, and it is clear that a LDPC code with excellent flexibility and performance, as well as low encoding and decoding complexity, can be defined for 802.16e.

This contribution provides an LDPC code definition agreed to by 6 of the 8 companies that provided LDPC proposals (Intel, Motorola, Nokia, Nortel, Samsung, and TI) to the informal LDPC group. The defined code provides good performance and ensures an LDPC is defined for .16. Performance results are provided for frequency selective channels (Ch D and TU 30 kmph). The group will continue investigations towards improving code performance and flexibility, as well as characterizing edge of cell and high mobility performance. In particular, using shortening and puncturing in addition to scaling/expansion will be explored. The benefits of defining IR will also be investigated.

References

- [1] Bo Xia and Eric Jacobsen, "Intel LDPC Proposed for IEEE 802.16e," Intel submission to informal LDPC group, 13 August 2004.
- [2] Min-seok Oh, Kyuhyuk Chung, "Scalable LDPC coding scheme for OFDMA," LG submission to informal LDPC group, 13 August 2004.
- [3] Y. Blankenship, B. Classon, and K. Blankenship, "Motorola Harmonized Structured LDPC Proposal," Motorola submission to informal LDPC group, 13 August 2004.
- [4] V. Stolpman, J. Zhang, N. van Waes, "Irregular structured LDPC codes," Nokia submission to informal LDPC group, 13 August 2004.
- [5] N. Burns, A. Purkovic, S. Sukobok, B. Johnson, "Algebraic low-density parity check codes for OFDMA PHY layer," Nortel submission to informal LDPC group, 13 August 2004.
- [6] E. Shasha, S. Litsyn, and E. Sharon, "Multi-rate LDPC code for OFDMA PHY," Runcom submission to informal LDPC group, 13 August 2004.
- [7] Jerry Kim, Gyubum Kyung, Hongsil Jeong, Sanghyo Kim, Panyuh Joo and DS Park, "Samsung's Harmonized Structured LDPC Proposal," Samsung submission to informal LDPC group, r1, 17 August 2004.
- [8] D. Hocevar and A. Batra, "LDPC coding scheme for OFDMA," TI submission to informal LDPC group, 13 August 2004.

Features

The LDPC codes have excellent performance, and contain features that provide flexibility and low encoding/decoding complexity.

- **Structured block LDPC for low complexity decoding.** The entire matrix (i.e., both the sections that correspond to the information and the parity) is composed of the same style of blocks, which reduces decoder implementation complexity and allows structured decoding.
- **Low-complexity differential-style encoding.** The encoding can be performed in a structured, recursive manner, without hurting performance with multiple weight-1 columns.
- **Designed to match the OFDMA subchannel structure.** No puncturing or rate-matching operations are required to provide exact code rates for many different block sizes.
- **Compatible with hybrid ARQ (Chase or Incremental Redundancy).**

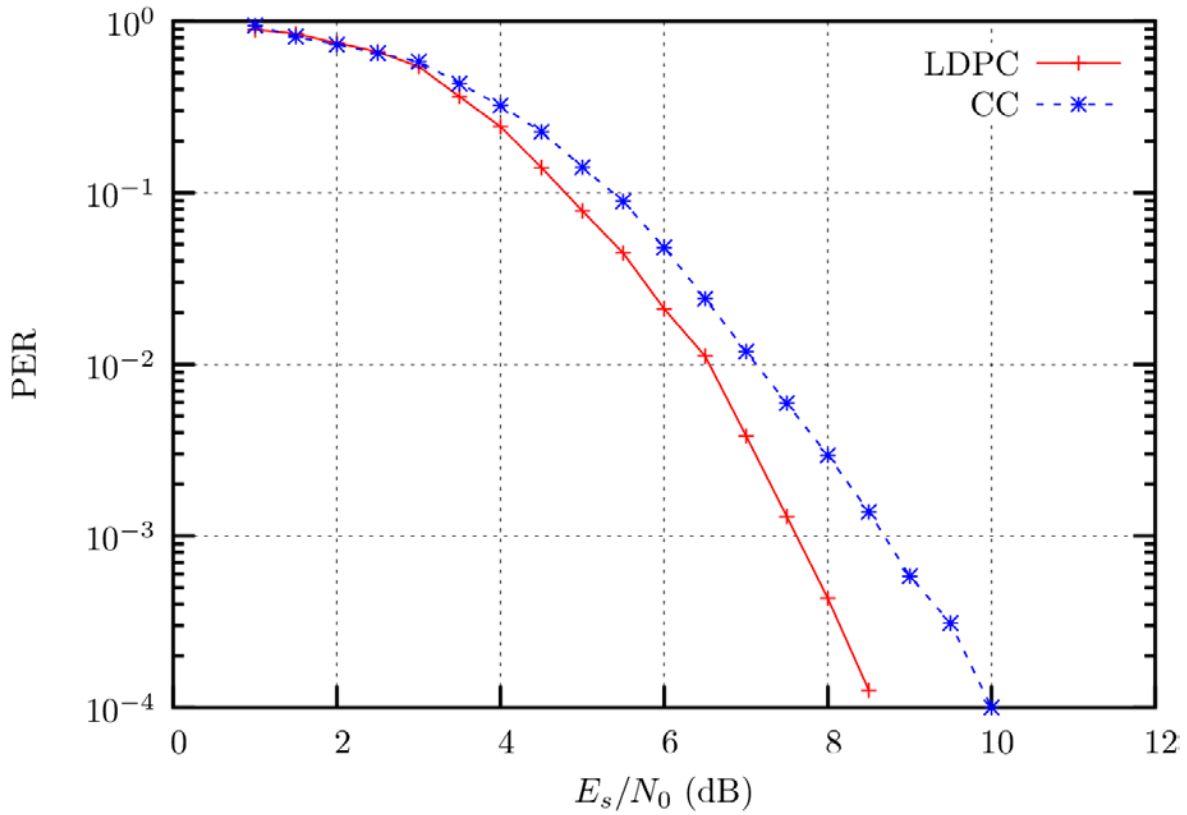
Simulation Results

Representative ODFM frequency selective fading simulation results are provided for code rates $\frac{1}{2}$ and $\frac{3}{4}$ and for block sizes 576 and 2304. The results indicate that in all cases the LDPC code significantly outperforms the tailbiting convolutional code (CC). In addition, the results indicate that the LDPC code (a block semi-regular

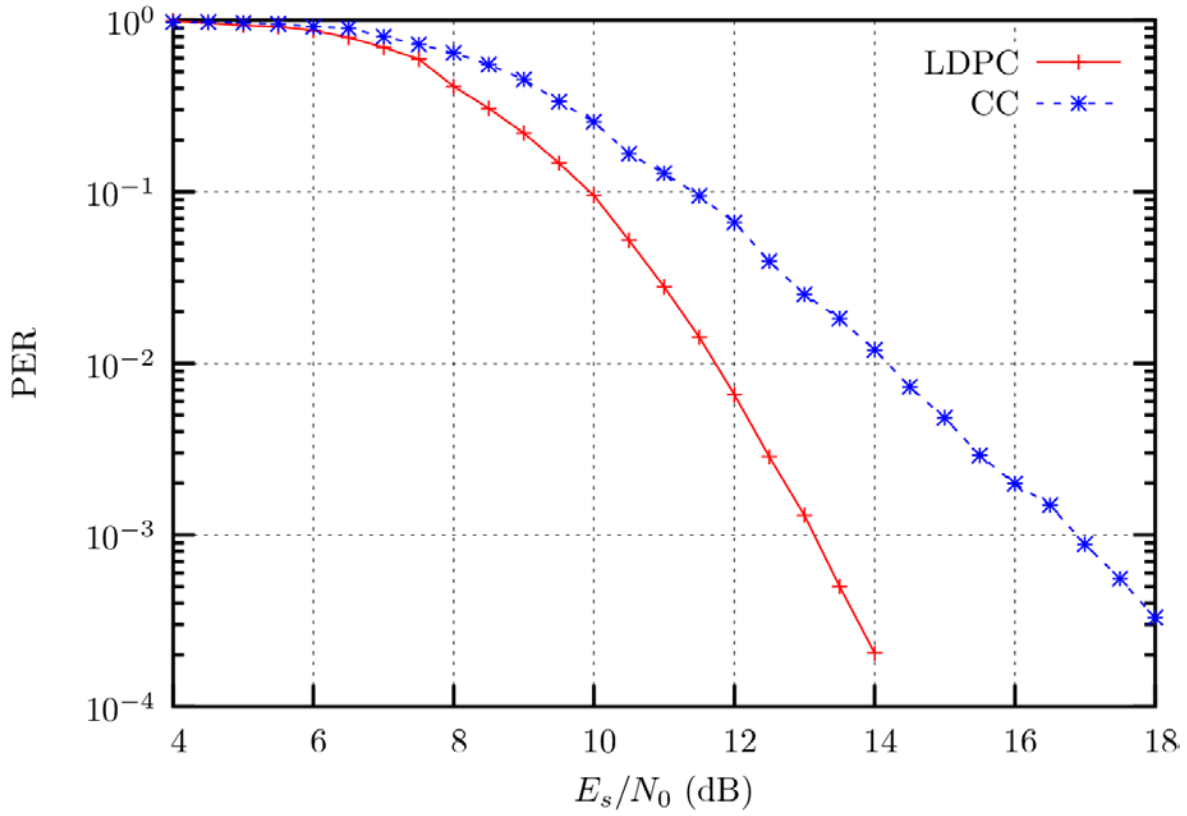
code) outperforms both a block regular code (i.e., a code without the modified dual-diagonal structure of the block semi-regular code) and a semi-regular code without a block structure. Generic floating-point belief propagation is used in all cases.

The first set of four plots is for 20 MHz bandwidth, QPSK, 50 iterations, and the GSM Typical Urban (TU) channel model at 30 kmph.

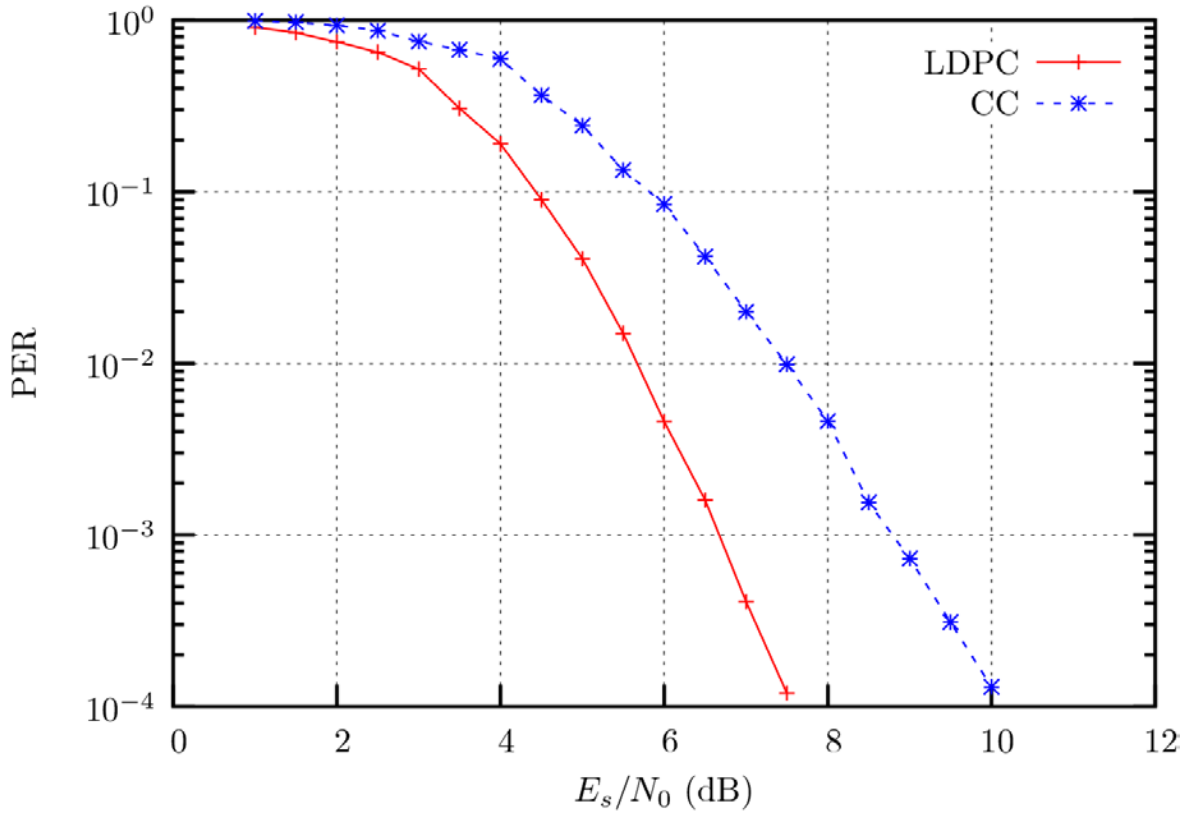
$$N = 576, r = 1/2$$

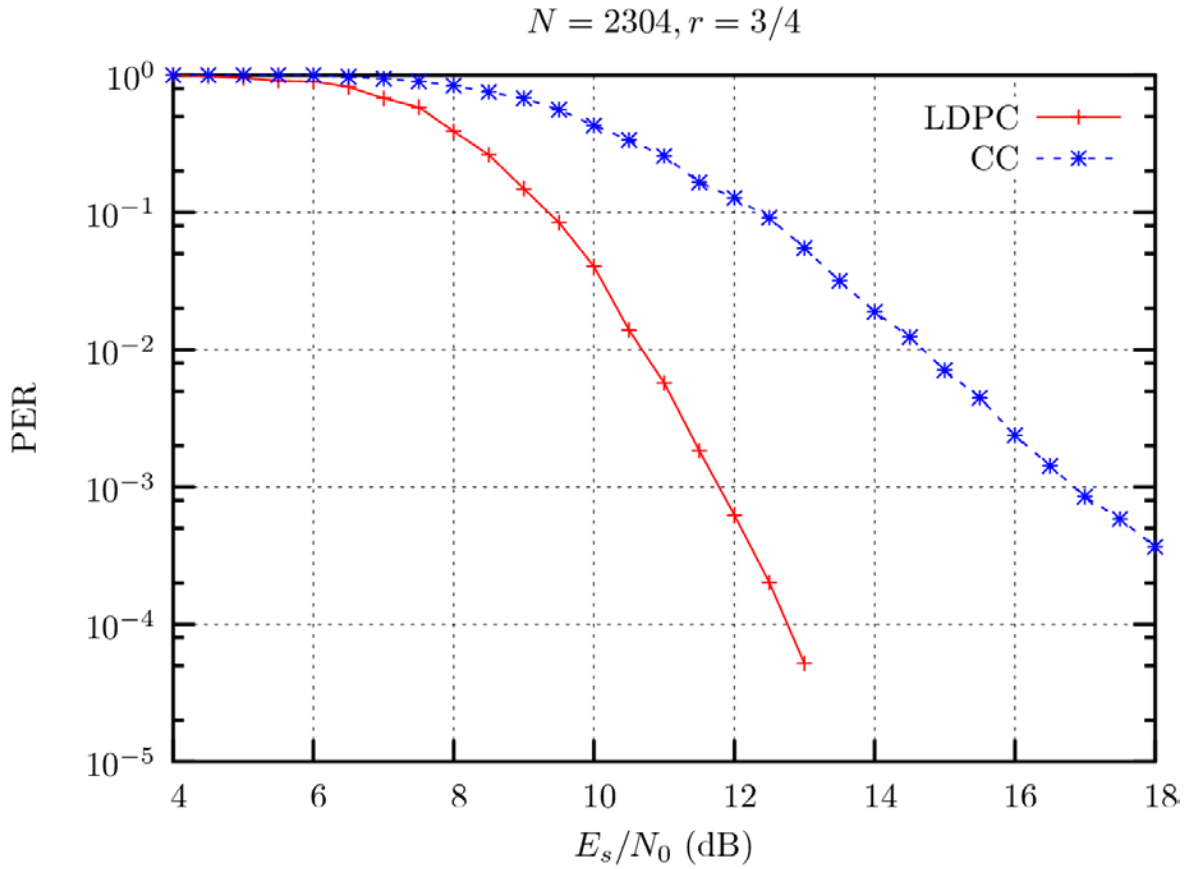


$N = 576, r = 3/4$



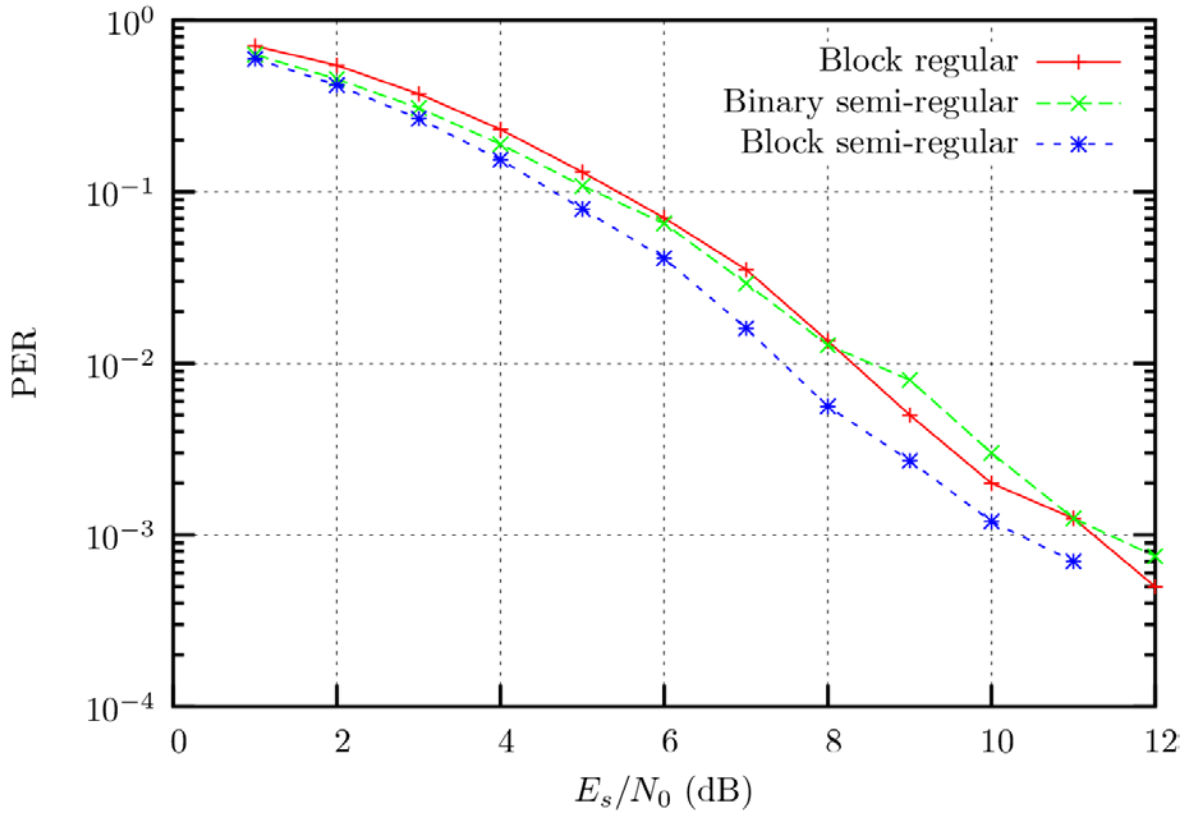
$N = 2304, r = 1/2$



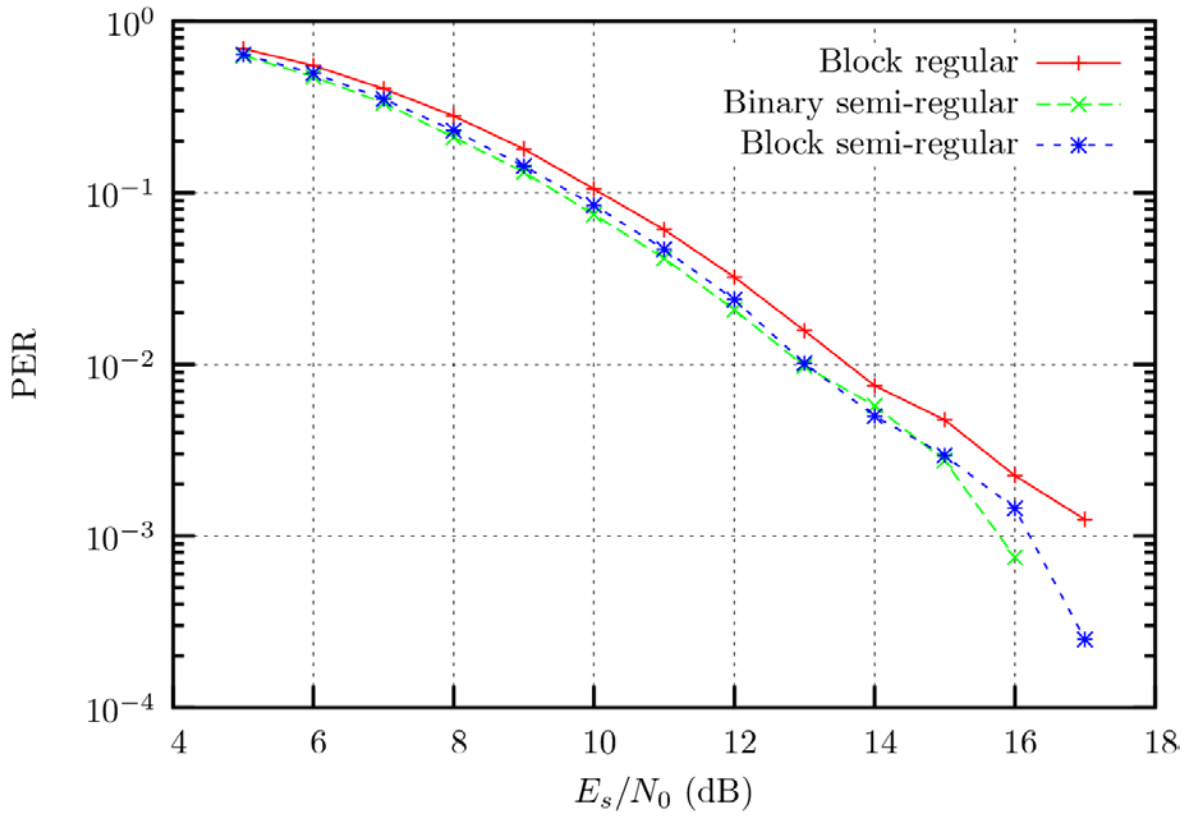


The second set of four plots is for a 40 MHz 802.11n OFDM system, BPSK, 20 iterations, and channel model D. The low PER performance is rough because only 20 packet errors (instead of 100 for the previous set of four plots) were run at each SNR point.

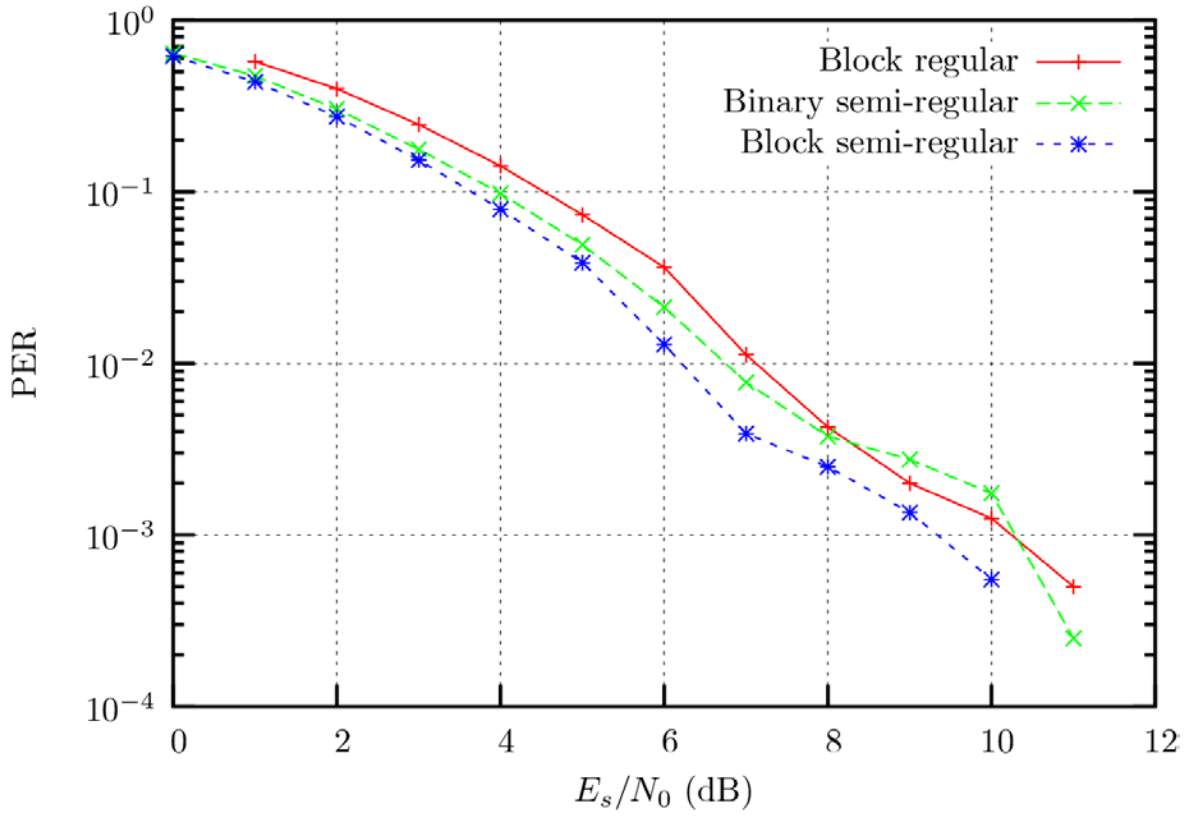
$N = 576, r = 1/2$



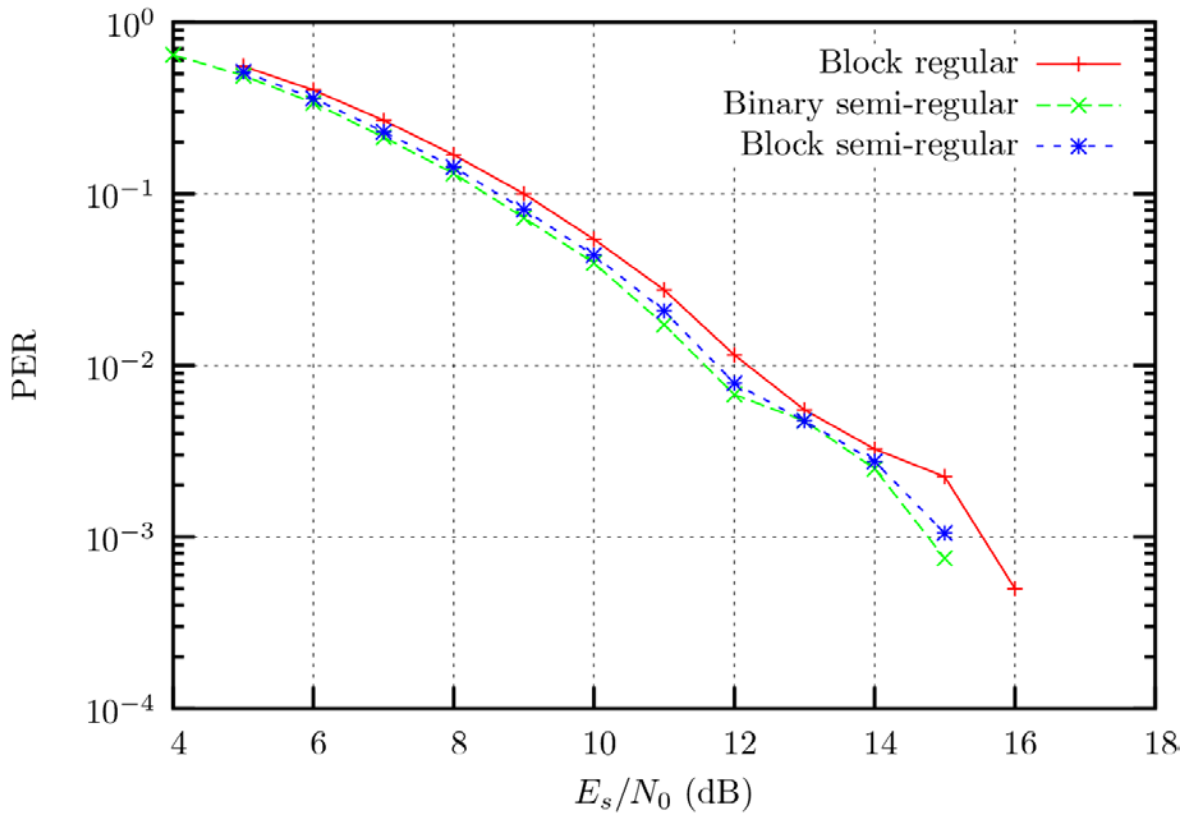
$N = 576, r = 3/4$



$N = 2304, r = 1/2$



$N = 2304, r = 3/4$



Recommended Text Changes:

Add/Modify the following text to 802.16e_D4, adjusting the numbering as required:

8.4.9.2 Encoding

<add text to the end of the ‘Concatenation’ paragraph starting at line 39>

, and for the LDPC encoding scheme (see 8.4.9.2.5) the concatenation rule is defined in 8.4.2.9.5.4.

8.4.9.2.5 Low Density Parity Check Code (optional)

8.4.9.2.5.1 Code Description

The LDPC code is based on a set of one or more fundamental LDPC codes. Each of the fundamental codes is a systematic linear block code. Using the described methods of scaling and shortening in 8.4.9.2.5.3 Code Rate and Block Size Adjustment, the fundamental codes can accommodate various code rates and packet sizes. ~~The code set can be applied to packets from [40] bytes up to ~200 bytes.~~

Each LDPC code in the set of LDPC codes is defined by a matrix \mathbf{H} of size m -by- n , where n is the length of the code and m is the number of parity check bits in the code. The number of systematic bits is $k=n-m$.

The matrix \mathbf{H} is defined as

$$\mathbf{H} = \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \cdots & \mathbf{P}_{0,n_b-2} & \mathbf{P}_{0,n_b-1} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \cdots & \mathbf{P}_{1,n_b-2} & \mathbf{P}_{1,n_b-1} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \cdots & \mathbf{P}_{2,n_b-2} & \mathbf{P}_{2,n_b-1} \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ \mathbf{P}_{m_b-1,0} & \mathbf{P}_{m_b-1,1} & \mathbf{P}_{m_b-1,2} & \cdots & \mathbf{P}_{m_b-1,n_b-2} & \mathbf{P}_{m_b-1,n_b-1} \end{bmatrix} = \mathbf{P}^{H_b}$$

where $\mathbf{P}_{i,j}$ is one of a set of z -by- z permutation matrices or a z -by- z zero matrix. The matrix \mathbf{H} is expanded from a binary base matrix \mathbf{H}_b of size m_b -by- n_b , where $n = z \cdot n_b$ and $m = z \cdot m_b$, with z an integer ≥ 1 . The base matrix is expanded by replacing each 1 in the base matrix with a z -by- z permutation matrix, and each 0 with a z -by- z zero matrix. The base matrix n_b is an integer is an integer multiple of 24.

The permutations used are circular right shifts, and the set of permutation matrices contains the $z \times z$ identity matrix and circular right shifted versions of the identity matrix. Because each permutation matrix is specified by a single circular right shift, the binary base matrix information and permutation replacement information can be combined into a single compact model matrix \mathbf{H}_{bm} . The model matrix \mathbf{H}_{bm} is the same size as the binary base matrix \mathbf{H}_b , with each binary entry (i,j) of the base matrix \mathbf{H}_b replaced to create the model matrix \mathbf{H}_{bm} . Each 0 in \mathbf{H}_b is replaced by a blank or negative value (e.g., by -1) to denote a $z \times z$ all-zero matrix, and each 1 in \mathbf{H}_b is replaced by a circular shift size $p(i,j) \geq 0$. The model matrix \mathbf{H}_{bm} can then be directly expanded to \mathbf{H} .

\mathbf{H}_b is partitioned into two sections, where \mathbf{H}_{b1} corresponds to the systematic bits and \mathbf{H}_{b2} corresponds to the parity-check bits, such that $\mathbf{H}_b = \left[\left(\mathbf{H}_{b1} \right)_{m_b \times k_b} \mid \left(\mathbf{H}_{b2} \right)_{m_b \times n_b} \right]$. Section \mathbf{H}_{b2} is further partitioned into two sections, where vector \mathbf{h}_b has odd weight, and \mathbf{H}'_{b2} has a dual-diagonal structure with matrix elements at row i , column j equal to 1 for $i=j$, 1 for $i=j+1$, and 0 elsewhere:


```

-1 -1 -1 -1 27 -1 -1 -1 -1 3 9 8 -1 -1 -1 36 27 -1 -1 24 -1 -1 -1 3 -1 16 -1 -1 -1 -1 2 -1 -1 31 -1 28
32 -1 30 17 9 -1 -1 7 -1 -1 -1 -1 18 -1 -1 -1 -1 -1 27 -1 -1 6 -1 -1 -1 -1 -1 34 5 24 -1 -1 4 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 1 -1 -1 -1 -1 22 -1 23 24 -1 23 29 -1 -1 13 -1 -1 12 -1 2 -1 -1 -1 22 -1 21 24
-1 22 -1 -1 11 -1 38 13 39 -1 30 -1 -1 28 -1 22 -1 -1 -1 -1 1 22 -1 -1 -1 -1 -1 -1 -1 -1 11 -1 19 -1 -1
-1 4 -1 -1 -1 21 -1 -1 18 -1 -1 25 25 -1 -1 -1 14 -1 24 -1 -1 -1 4 6 25 -1 -1 -1 -1 20 -1 -1 -1 -1 27 -1
2 -1 -1 2 -1 -1 27 -1 -1 37 -1 16 -1 -1 15 -1 -1 -1 -1 -1 12 34 -1 -1 -1 -1 -1 37 38 12 -1 -1 19 -1 -1
-1 8 -1 -1 -1 -1 36 7 25 -1 -1 -1 -1 24 -1 -1 -1 19 -1 -1 39 -1 -1 3 -1 -1 29 38 -1 -1 -1 9 -1 -1 -1 15
-1 -1 18 21 -1 25 -1 -1 -1 17 -1 -1 -1 -1 -1 9 -1 3 -1 36 26 -1 -1 -1 21 -1 6 -1 -1 -1 -1 4 -1 -1 15 -1
-1 -1 4 -1 -1 -1 -1 39 -1 -1 -1 -1 23 33 -1 -1 -1 29 32 -1 -1 -1 -1 -1 17 18 -1 -1 12 31 26 -1 -1 15 -1 -1
-1 0 27 -1 -1 -1 -1 -1 -1 -1 35 -1 18 -1 7 -1 -1 -1 5 -1 -1 -1 34 -1 -1 20 -1 29 -1 -1 -1 20 26 -1 -1 0
34 -1 -1 -1 17 23 -1 -1 -1 -1 11 -1 -1 34 -1 -1 -1 -1 -1 34 -1 12 29 -1 -1 11 -1 33 -1 -1 35 -1 -1 20 -1

```

$\mathbf{H}_{bm2} =$

```

0 0 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 0 0 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 0 0 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 0 0 -1 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 0 0 -1 -1 -1 -1 -1 -1
-1 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1
0 -1 -1 -1 -1 -1 0 0 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 0 0 -1 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 0 0 -1 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 0 0 -1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 0 0
0 -1 -1 -1 -1 -1 -1 -1 -1 -1 0

```

8.4.9.2.5.2 LDPC encoding

The code is flexible in that it can accommodate various code rates as well as packet sizes. ~~Since LDPCs are block-oriented codes, some restrictions are necessary on the combinations of available code rates and codeword sizes in order to control complexity.~~

The encoding of a packet at the transmitter generates parity-check bits $\mathbf{p}=(p_0, \dots, p_{m-1})$ based on an information block $\mathbf{s}=(s_0, \dots, s_{k-1})$, and transmits the parity-check bits along with the information block. Because the current symbol set to be encoded and transmitted is contained in the transmitted codeword, the information block is also known as systematic bits. The encoder receives the information block $\mathbf{s}=(s_0, \dots, s_{k-1})$ and uses the matrix \mathbf{H}_{bm} to determine the parity-check bits. The expanded matrix \mathbf{H} is determined from the model matrix \mathbf{H}_{bm} . Since the expanded matrix \mathbf{H} is a binary matrix, encoding of a packet can be performed with vector or matrix operations conducted over GF(2).

One method of encoding is to determine a generator matrix \mathbf{G} from \mathbf{H} such that $\mathbf{G} \mathbf{H}^T = \mathbf{0}$. A k -bit information block $\mathbf{s}_{1 \times k}$ can be encoded by the code generator matrix $\mathbf{G}_{k \times n}$ via the operation $\mathbf{x} = \mathbf{s} \mathbf{G}$ to become an n -bit codeword $\mathbf{x}_{1 \times n}$, with codeword $\mathbf{x}=[\mathbf{s} \mathbf{p}]=[s_0, s_1, \dots, s_{k-1}, p_0, p_1, \dots, p_{m-1}]$, where p_0, \dots, p_{m-1} are the parity-check bits; and s_0, \dots, s_{k-1} are the systematic bits.

Encoding an LDPC code from \mathbf{G} can be quite complex. The LDPC codes are defined such that very low complexity encoding directly from \mathbf{H} is possible.

Direct Encoding (Method 1)

Encoding is the process of determining the parity sequence \mathbf{p} given an information sequence \mathbf{s} . To encode, the information block \mathbf{s} is divided into $k_b = n_b - m_b$ groups of z bits. Let this grouped \mathbf{s} be denoted \mathbf{u} ,

$$\mathbf{u} = [\mathbf{u}(0) \quad \mathbf{u}(1) \quad \cdots \quad \mathbf{u}(k_b - 1)],$$

where each element of \mathbf{u} is a column vector as follows

$$\mathbf{u}(i) = [s_{iz} \quad s_{iz+1} \quad \cdots \quad s_{(i+1)z-1}]^T$$

Using the model matrix \mathbf{H}_{bm} , the parity sequence \mathbf{p} is determined in groups of z . Let the grouped parity sequence \mathbf{p} be denoted \mathbf{v} ,

$$\mathbf{v} = [\mathbf{v}(0) \quad \mathbf{v}(1) \quad \cdots \quad \mathbf{v}(m_b - 1)],$$

where each element of \mathbf{v} is a column vector as follows

$$\mathbf{v}(i) = [p_{iz} \quad p_{iz+1} \quad \cdots \quad p_{(i+1)z-1}]^T$$

Encoding proceeds in two steps, (a) initialization, which determines $\mathbf{v}(0)$, and (b) recursion, which determines $\mathbf{v}(i+1)$ from $\mathbf{v}(i)$, $0 \leq i \leq m_b - 2$.

An expression for $\mathbf{v}(0)$ can be derived by summing over the rows of \mathbf{H}_{bm} to obtain

$$\mathbf{P}_{p(x,k_b)} \mathbf{v}(0) = \sum_{j=0}^{k_b-1} \sum_{i=0}^{m_b-1} \mathbf{P}_{p(i,j)} \mathbf{u}(j) \quad (1)$$

where x , $1 \leq x \leq m_b - 2$, is the row index of \mathbf{h}_{bm} where the entry is nonnegative and unpaired, and \mathbf{P}_i represents the $z \times z$ identity matrix circularly right shifted by size i . Equation (1) is solved for $\mathbf{v}(0)$ by multiplying by $\mathbf{P}_{p(x,k_b)}^{-1}$, and $\mathbf{P}_{p(x,k_b)}^{-1} = \mathbf{P}_{z-p(x,k_b)}$ since $p(x,k_b)$ represents a circular shift.

The recursion expressed in Equation (2) can be derived by considering the structure of \mathbf{H}'_{b2} ,

$$\mathbf{v}(1) = \sum_{j=0}^{k_b-1} \mathbf{P}_{p(i,j)} \mathbf{u}(j) + \mathbf{P}_{p(i,k_b)} \mathbf{v}(0), \quad i = 0, \quad (2)$$

$$\mathbf{v}(i+1) = \mathbf{v}(i) + \sum_{j=0}^{k_b-1} \mathbf{P}_{p(i,j)} \mathbf{u}(j) + \mathbf{P}_{p(i,k_b)} \mathbf{v}(0), \quad i = 1, \dots, m_b - 2 \quad (3)$$

where

$$\mathbf{P}_{-1} \equiv \mathbf{0}_{z \times z}.$$

Thus all parity bits not in $\mathbf{v}(0)$ are determined by evaluating Equation (2) for $0 \leq i \leq m_b - 2$.

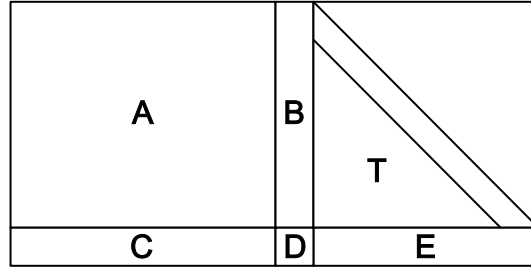
Equations (1) and (2) completely describe the encoding algorithm. These equations also have a straightforward interpretation in terms of standard digital logic architectures. Since the non-zero elements $p(i,j)$ of \mathbf{H}_{bm} represent circular shift sizes of a vector, all products of the form $\mathbf{P}_{p(i,j)} \mathbf{u}(j)$ can be implemented by a size- z barrel shifter.

Direct Encoding (Method 2)

For efficient encoding of LDPC, \mathbf{H} are divided into the form

$$\mathbf{H} = \begin{pmatrix} \mathbf{A} & \mathbf{B} & \mathbf{T} \\ \mathbf{C} & \mathbf{D} & \mathbf{E} \end{pmatrix} \quad (1)$$

where A is $(N_p - g) \times N_k$, B is $(N_p - g) \times g$, T is $(N_p - g) \times (N_p - g)$, C is $g \times N_k$, D is $g \times g$, and finally, E is $g \times (N_p - g)$. The basic structure of the H matrix is



Further, all these matrices are sparse and T is lower triangular with ones along the diagonal. B and D part have the column degree 3 and D has shift value of 1. B is with the first entry of 1 and shift value 0 in the middle of the column. This other entry is non-zero.

Let $v=(u, p_1, p_2)$ that u denotes the systematic part, p_1 and p_2 combined denote the parity part, p_1 has length g , and p_2 has length $(N_p - g)$. The definition equation $H \cdot v^t = 0$ splits into two equations, as in equation 3 and 4 namely

$$Au^t + Bp_1^t + Tp_2^t = 0 \tag{2}$$

and

$$(-ET^{-1}A + C)u^t + (-ET^{-1}B + D)p_1^t = 0 \tag{3}$$

Define $\phi := -ET^{-1}B + D$ and when we use the parity check matrix as indicated appendix we can get $\phi = I$. Then from (4) we conclude that

$$p_1^t = (-ET^{-1}A + C)u^t \tag{5}$$

and

$$p_2^t = T^{-1}(Au^t + Bp_1^t). \tag{6}$$

As a result, the encoding procedures and the corresponding operations can be summarized below and illustrated in Fig. 1.

Encoding procedure

- Step 1) Compute Au^t and Cu^t .
- Step 2) Compute $ET^{-1}(Au^t)$.
- Step 3) Compute p_1^t by $p_1^t = ET^{-1}(Au^t) + Cu^t$.
- Step 4) Compute p_2^t by $Tp_2^t = Au^t + Bp_1^t$.

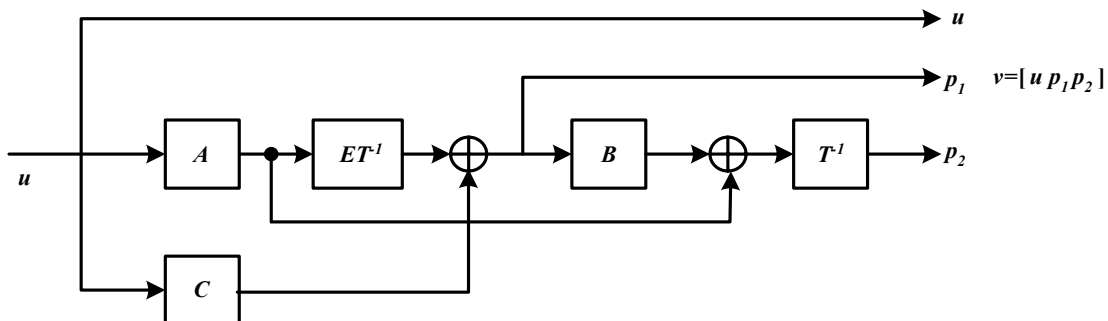


Fig. 2 Block diagram of the encoder architecture for the block LDPC code.

~~TBD description of packet encoding.~~

8.4.9.2.5.3 Code Rate and Block Size Adjustment

The code design will be flexible to support a range of code rates and block sizes through code rate and block size adjustment of the one or more H matrices of the fundamental code set. ~~The exact methods for supporting code rate and block size adjustment will depend on the final design.~~ For each supported rate and block size, ~~there will be~~ some combination of matrix selection, shortening, repetition, matrix expansion, and/or concatenation will be used.

Different block sizes and code rates are supported through using a variable z expansion factor. The z expansion factors for coded block sizes n corresponding to integer numbers of subchannels are provided below. In each case, the number of information bits is equal to the code rate times the coded block size n . In addition to matrix expansion, shortening is used and puncturing may be used to support some coded block sizes and code rates.

n (bits)	n (bytes)	k (bytes)			Number of subchannels		
		R=1/2	R=2/3	R=3/4	QPSK	16QAM	64QAM
576	72	36	48	54	6	3	2
672	84	42	56	63	7		
768	96	48	64	72	8	4	
864	108	54	72	81	9		3
960	120	60	80	90	10	5	
1056	132	66	88	99	11		
1152	144	72	96	108	12	6	4
1248	156	78	104	117	13		
1344	168	84	112	126	14	7	
1440	180	90	120	135	15		5
1536	192	96	128	144	16	8	
1632	204	102	136	153	17		
1728	216	108	144	162	18	9	6
1824	228	114	152	171	19		
1920	240	120	160	180	20	10	
2016	252	126	168	189	21		7
2112	264	132	176	198	22	11	
2208	276	138	184	207	23		
2304	288	144	192	216	24	12	8

Shortening may be applied to any expanded H matrix by reducing the number of subchannels available for the codeword. The number of bit corresponding to the reduced number of subchannels is equal to the number of shortened bits L . The matrix \mathbf{H} is designed such that excellent performance is achieved under shortening, with different column weights interlaced between the first L columns of \mathbf{H}_1 and the rest of \mathbf{H}_1 . Encoding with shortening is similar to encoding without shortening, except that the current symbol set has only $k-L$ systematic

bits in the information block, $\mathbf{s}'=(s_0, \dots, s_{k-L-1})$. When encoding, the encoder first prepends L zeros to \mathbf{s}' of length $(k-L)$. Then the zero-padded information vector $\mathbf{s}=[\mathbf{0}_L \ \mathbf{s}']$ is encoded using \mathbf{H} as if unshortened to generate parity bit vector \mathbf{p} (length m). After removing the prepended zeros, the code bit vector $\mathbf{x}=[\mathbf{s}' \ \mathbf{p}]$ is transmitted over the channel. This encoding procedure is equivalent to encoding \mathbf{s}' using the last $(n-L)$ columns of matrix \mathbf{H} to determine the parity-check vector \mathbf{p} .

The z expansion factors are determined by the target block size n and the base matrix size n_b . Examples of the z expansion factors are given in the tables below. The base matrix n_b is an integer is an integer multiple of 24.

Table

n (bits)	n (bytes)	z expansion factor			Number of subchannels		
		R=1/2	R=2/3	R=3/4	QPSK	16QAM	64QAM
96	12	4	4	2	1		
192	24	8	8	4	2	1	
288	36	12	12	6	3		1
384	48	16	16	8	4	2	
480	60	20	20	10	5		
576	72	24	24	12	6	3	2
672	84	28	28	14	7		
768	96	32	32	16	8	4	
864	108	36	36	18	9		3
960	120	40	40	20	10	5	
1056	132	44	44	22	11		
1152	144	48	48	24	12	6	4
1248	156	52	52	26	13		
1344	168	56	56	28	14	7	
1440	180	60	60	30	15		5
1536	192	64	64	32	16	8	
1632	204	68	68	34	17		
1728	216	72	72	36	18	9	6
1824	228	76	76	38	19		
1920	240	80	80	40	20	10	
2016	252	84	84	42	21		7
2112	264	88	88	44	22	11	

2208	276	92	92	46	23		
2304	288	96	96	48	24	12	8
base n-k		12	8	12			
base n		24	24	48			

TBD description of code adjustment.

8.4.9.2.5.4 Packet Encoding

~~After harmonization, this section will be replaced with something equivalent to what is in section 8.2.1.2.4.1 which is the concatenation/packet encoding scheme for the CTC.~~

~~Since transported data packets can be any size from typically about 40 bytes up to 12000 bits and larger, the system must be able to encode variable length packets in a consistent manner. This consistency is required to ensure that the receiver always knows how to reconstruct the information field from the encoded transmitted data.~~

The encoding block size k shall depend on the number of subchannels allocated and the modulation specified for the current transmission. Concatenation of a number of subchannels shall be performed in order to make larger blocks of coding where it is possible, with the limitation of not passing the largest block under the same coding rate (the block defined by the 64-QAM modulation). The table below specifies the concatenation of subchannels for different allocations and modulations. The concatenation rule follows the subchannel concatenation rule for CC (Table 315) except that for LDPC the concatenation does not depend on the code rate.

For any modulation and FEC rate, given an allocation of N_{sch} subchannels, we define the following parameters:

- j parameter dependent on the modulation and FEC rate
- N_{sch} number of allocated subchannels
- F $\text{floor}(N_{\text{sch}}/j)$
- M $N_{\text{sch}} \bmod j$

The subchannel concatenation rule for CC in Table 315 is applied, noting that in Table 315 the parameter n is equal to N_{sch} , the parameter k is equal to F , and the parameter m is equal to M . The parameter j for LDPC is determined as shown in the table below.

Modulation	j
QPSK	$j=24$
16-QAM	$j=12$
64-QAM	$j=8$

~~Each packet is encoded as an entity. In other words, the data boundary of a packet is respected by the encoder. Control information and packets that result in a codeword size n of less than 576 bits smaller than 40 bytes are encoded using convolutional coding (CC) with appropriate code rates and modulation orders, as described in section 8.4.9.2.1.~~

The length and required rate of the packet that is to be encoded is all that is needed to encode or decode the packet using the following rules:

If $\text{Length} \leq N_i$ bits, then TBD.

If $\text{Length} > N_i$ bits and $\leq 2 N_i$ bits, then TBD

If $\text{Length} > 2 N_i$ bits, then compute $N_r = \text{modulo}(\text{Length}, N_i)$ (in bits), then TBD.

Concatenation when TBD

Combination of shortening and concatenation when TBD

The intent of the above rule set is to provide a means for data transmission without the need for additional information beyond the packet field length. This scheme does so with a simple rule set that reduces the rate of the last codewords in order to reduce the number of iterations (and therefore the latency) that must be performed on the last portion of the data. The length and position of the shortened codewords and erased bits are deterministic when the above rules are followed.

For all packets the codeword bits can be indexed using the corresponding column indices of the H matrix. Using this convention the systematic codeword bits comprise the leftmost bits starting at bit location zero, and fill the codeword to bit $k-1$. The remaining $N-k$ bits of the codeword, from indices k to $N-1$ are the parity bits. The codeword systematic bits are filled in an order consistent with the indices, so that the first bits of the packet fill the codeword from the lowest indices linearly to the highest indices. The codeword is then transmitted in a linear fashion starting from the lowest indices so that the systematic bits are transmitted first, followed by the parity bits. For shortened codewords the zeros are padded in the low order bits, so that the final codeword starting at the lowest indices contains first zero padded bits and then the systematic data bits followed by the parity bits. The zero padded bits are not typically sent over the channel.