Project	**IEEE 802.16 Broadband Wireless Access Working Group <http://ieee802.org/16>**	
Title	**LDPC Coding for OFDMA PHY**	
Date Submitted	**2004-11-04**	
Source(s)	Min-seok Oh, Kyuhyuk Chung, Kihyoung Cho LG Electronics,Inc. 533,Hogye-1dong,Dongan-gu, Anyang-shi,Kyongki-do,Korea	Voice: 82-31-450-2945 Fax: 82-31-450-7912 [mailto:minoh@lge.com, kyuhyuk@lge.com, kihyoung@lge.com]
Re:	Response to Sponsor Ballot	
Abstract	Proposal for LDPC FEC scheme as an option for OFDMA	
Purpose	Complete the LDPC FEC 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>.	

Introduction

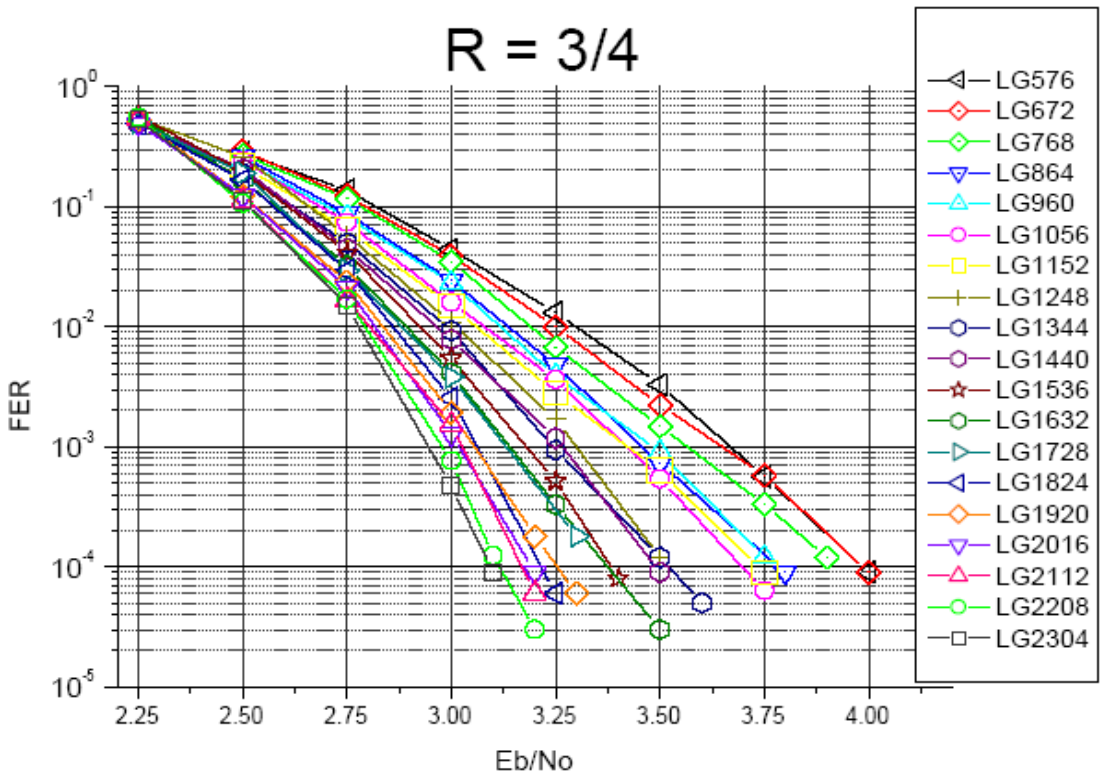
On the last session, the basic framework of LDPC codes for OFDMA PHY was adopted. So the completing of the LDPC code section is required. We propose low-complexity and high performance LDPC codes supporting all rates and codeword sizes with flexible and efficient rate adjustment. The proposed LDPC codes are based on the contribution IEEE C802.16e-04/373r1.

Features

- Low complexity encoding and decoding
- Low memory requirement
- Good performance over AWGN channel and fading channel environment
- Simple code description

Simulation Results

For rate 3/4, performance curves are provided for all codeword sizes. One base matrix with 12 by 48 above is used for the following simulations. To cover different codeword sizes at a given code rate, expansion is used.



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.

Each LDPC code in the set of LDPC codes is defined by a matrix **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 **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 **H** is expanded from a 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 permutation information in the base matrix with a z-by-z permutation matrix, and each -1 with a z-by-z zero matrix. The base matrix is an integer is an integer multiple of 24.

$\mathbf{H}_b$ 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 $H_b = \left[(H_{b1})_{m_b \times k_b} \mid (H_{b2})_{m_b \times m_b} \right]$.

Base matrix

To reduce complexity, the minimal number of base matrices is used. Two base matrices are used to cover all rates and codeword sizes. One base matrix with size 8 by 24 is for r=2/3. The other base matrix with 12 by 48 is for r=3/4 and r=1/2 with shortening. For r=2/3 base matrix, the maximum shift s_m is 96. For r=3/4 base matrix, s_m is 96 in order to cover all the codeword sizes at r=1/2. A shift s is scaled based on an expansion factor z as follows;

$$s' = \lfloor s \cdot z / s_m \rfloor$$

where $\lfloor \cdot \rfloor$ is the nearest integer to minus infinity.

Matrix descriptions

r=2/3

1	-1	4	-1	-1	3	7	-1	-1	17	-1	46	21	-1	-1	65	1	0	-1	-1	-1	-1	-1	-1
-1	19	-1	90	86	-1	-1	49	9	-1	91	-1	-1	32	76	-1	-1	0	0	-1	-1	-1	-1	-1
-1	16	-1	24	26	-1	-1	0	-1	28	-1	11	-1	52	-1	94	-1	-1	0	0	-1	-1	-1	-1
39	-1	61	-1	-1	92	53	-1	58	-1	38	-1	88	-1	66	-1	-1	-1	-1	0	0	-1	-1	-1
14	-1	-1	75	60	-1	-1	67	-1	89	-1	33	-1	40	-1	35	0	-1	-1	-1	0	0	-1	-1
-1	31	78	-1	-1	36	30	-1	54	-1	64	-1	48	-1	57	-1	-1	-1	-1	-1	-1	0	0	-1
-1	42	-1	93	-1	-1	77	74	-1	-1	68	-1	41	23	-1	-1	-1	-1	-1	-1	-1	-1	0	0
34	-1	84	-1	82	6	-1	-1	72	5	-1	87	-1	-1	15	37	1	-1	-1	-1	-1	-1	-1	0

r=3/4

-1	20	-1	75	-1	-1	-1	-1	70	-1	18	-1	-1	6	-1	-1	65	-1	-1	21	-1	85	-1	-1	-1	65	-1	-1	25	-1	8	-1	-1	-1	-1	29	1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
90	-1	-1	-1	50	-1	-1	63	-1	-1	-1	53	-1	-1	85	76	-1	-1	-1	-1	18	-1	-1	48	-1	-1	39	35	-1	-1	-1	-1	35	-1	35	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	28	-1	-1	69	52	-1	-1	23	-1	-1	47	-1	-1	-1	-1	20	84	-1	-1	-1	46	-1	95	-1	-1	-1	-1	33	-1	86	-1	48	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	78	-1	11	-1	-1	6	-1	-1	-1	85	79	-1	-1	-1	70	-1	27	-1	-1	-1	-1	75	-1	3	-1	-1	-1	90	-1	-1	30	44	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
72	-1	-1	36	-1	-1	-1	-1	47	45	-1	-1	-1	38	-1	-1	-1	27	-1	-1	34	82	-1	-1	-1	-1	89	-1	68	-1	34	-1	-1	-1	73	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	8	-1	-1	-1	87	45	-1	-1	-1	91	-1	-1	-1	63	71	-1	-1	-1	71	-1	-1	15	-1	66	-1	-1	55	-1	-1	-1	23	-1	-1	-1	30	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	67	39	-1	-1	-1	83	-1	-1	-1	58	2	-1	-1	-1	19	-1	-1	-1	84	-1	-1	82	-1	-1	31	22	-1	-1	-1	5	-1	-1	-1	58	0	-1	-1	-1	-1	-1	0	0	-1	-1	-1
46	-1	-1	-1	35	-1	35	-1	-1	-1	22	-1	-1	-1	18	-1	-1	32	-1	45	-1	64	-1	-1	-1	7	-1	-1	84	-1	-1	-1	47	13	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
-1	40	-1	-1	-1	63	-1	-1	32	79	-1	-1	-1	17	-1	35	-1	-1	66	-1	-1	-1	14	-1	26	-1	-1	-1	-1	73	55	-1	-1	-1	74	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	67	-1	-1	-1	69	-1	30	-1	-1	-1	32	85	28	-1	-1	-1	-1	8	11	-1	-1	-1	-1	76	-1	40	-1	95	-1	-1	42	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	-1	-1	63	9	-1	-1	-1	92	-1	62	-1	-1	-1	-1	69	-1	89	-1	-1	-1	94	63	-1	-1	-1	-1	77	-1	-1	72	-1	-1	9	60	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
34	-1	47	-1	-1	-1	43	-1	-1	31	-1	-1	-1	-1	75	-1	24	-1	-1	-1	42	-1	-1	13	-1	87	-1	-1	-1	88	-1	-1	42	-1	-1	10	1	-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.

The encoding of a packet at the transmitter generates parity-check bits $p = (p_0, \cdots, p_{m-1})$ based on an information block $s = (s_0, \cdots, s_{k-1})$, 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 and uses the matrix $\mathbf{H}_{bm}$ to determine the parity-check bits. The expanded matrix **H** is determined from the model matrix $\mathbf{H}_{bm}$. Since the expanded matrix **H** is a binary matrix, encoding of a packet can be performed with vector of matrix operations conducted over GF(2).

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

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

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 Adjustment of the one or more H matrices of the fundamental code set. For each supported rate and block size. Some combinations 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. 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 thecodeword. The number of bit corresponding to the reduced number of subchannels is equal to the number of shortened bits *L*. The matrix H is designed such that excellent performance is achieved under shortening, withdifferent column weights interlaced between the first **L** columns of H_1 and the rest of 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, $s' = (s_0, \cdots, s_{k-L-1})$. When encoding, the encoder first prepends *L* zeros to s' of length *(k-L)*. Then the zero-padded information vector $s = [0_L \; s']$ is encoded using H as if unshortened to generate parity bit vector p (length m). After removing the prepended zeros, the code bit vector $x = [s' p]$ is transmitted over the channel. This encoding procedure is equivalent to encoding using the last *(n-L)* columns of the matrix H to determine the parity-check vector 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.

8.4.9.2.5.4 Packet Encoding

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 dose 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	floor(N_{sch}/ *j*)
M	N_{sch} mod *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 isequal 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

Control information and packets that result in a codeword size n of less than 576 bits are encoded using convolutional coding (CC) with appropriate code rates and modulation orders, as described in section 8.4.9.2.1.