

Project	<b>IEEE 802.16 Broadband Wireless Access Working Group</b> < <a href="http://ieee802.org/16">http://ieee802.org/16</a> >	
Title	<b>High Girth LDPC Coding for OFDMA PHY</b>	
Date Submitted	<b>2005-01-10</b>	
Source(s)	Robert Xu, David Yuan, Li Zeng ZTE Corporation 3/F, Bldg.711, Pengji Industrial Park, Liantang, Shenzhen, 518004, P.R.China	Voice: +86 755 26773000 6574 Fax: +86 755 26773000 6616 mailto: <a href="mailto:xu.jun2@zte.com.cn">xu.jun2@zte.com.cn</a>  <a href="mailto:yuan.liuqing@zte.com.cn">yuan.liuqing@zte.com.cn</a>
Re:	Response to Sponsor Ballot on IEEE802.16e/D5a document	
Abstract	High girth LDPC codes design technologies are introduced into the contribution for the first time, to overcome the “error floor” phenomenon, and enhance the BER curve descending speed when SNR is high. Also it has a performance improvement over the LDPC codes proposed in the previous contribution “IEEE C802.16e-04/373r1”, which is supported by 6 of the 8 companies that provided LDPC proposals to an informal LDPC group.	
Purpose	To incorporate the text changes proposed in this contribution into the 802.16e/D6 draft.	
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 text 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	<p>The contributor is familiar with the IEEE 802.16 Patent Policy and Procedures (Version 1.0) &lt;<a href="http://ieee802.org/16/ipr/patents/policy.html">http://ieee802.org/16/ipr/patents/policy.html</a>&gt;, including the statement “IEEE standards may include the known use of patent(s), including patent applications, if there is technical justification in the opinion of the standards-developing committee and provided the IEEE receives assurance from the patent holder that it will license applicants under reasonable terms and conditions for the purpose of implementing the standard.”</p> <p>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 &lt;<a href="mailto:r.b.marks@ieee.org">mailto:r.b.marks@ieee.org</a>&gt; as early as possible, in written or electronic form, of any patents (granted or under application) that may cover technology that is under consideration by or has been approved by IEEE 802.16. The Chair will disclose this notification via the IEEE 802.16 web site &lt;<a href="http://ieee802.org/16/ipr/patents/notices">http://ieee802.org/16/ipr/patents/notices</a>&gt;.</p>	

# High Girth LDPC Coding for OFDMA PHY

*Robert Xu, David Yuan, Li Zeng*

**ZTE Corporation**

## Overview

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.

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. We would like to support their work, so we apply our technologies to improve the LDPC codes proposed in the contribution "IEEE C802.16e-04/373r1" produced by the group.

Based on the contribution "IEEE C802.16e-04/373r1", which contains a harmonized LDPC code definition agreed by 6 of the 8 companies that provided LDPC proposals to an informal LDPC group, we design high girth LDPC codes. The base matrix's dimension and the degree distribution of these LDPC codes is the same as those in "IEEE C802.16e-04/373r1". That is, the part of check matrix corresponding to systematic bits has a regular structure, row weight and column weight of the part are all 4, and the part of check matrix corresponding to parity bits has a dual-diagonal structure.

As we all know, code rate, codeword length and degree distribution decide the performance of LDPC codes. When Message Passing algorithm is used, the short cycles in the bipartite of LDPC codes will obviously degrade the performance of the LDPC codes, especially when SNR is high. Girth was defined as the length of the shortest cycle of the bipartite of LDPC codes, and it has become a criterion on the performance of LDPC codes. During the decoding iteration process, the extrinsic information from one variable node always returns to itself. Some variable nodes are dependent on each other. The higher the girth of one LDPC code is, the more iteration times that the extrinsic information from one variable node return to itself needs, and the less extrinsic information from one variable node returns to itself, so more independent the variable nodes will be. Thus it is very important to construct high girth LDPC codes to satisfy the requirement of Message passing algorithm.

High girth LDPC codes try to overcome the "error floor" phenomenon, and the BER curve of them will descend more steeply. However, normal LDPC codes have the "error floor" phenomenon, BER curve descends more and more slow. It is always difficult to arrive at the point  $BER = 10e-6$ , which efficient data communication needs. So high girth LDPC codes are suitable for the situation where low BER is needed.

By changing base matrix for each specific code rate to design high girth LDPC codes, performance improvement is obtained. Our method not only can be used to design regular LDPC codes, but also can be used to design irregular LDPC codes. In addition, any matrix structure with given code rate, codeword length and degree distribution can be constructed by our method, not only those given in [1], which largely increases the feasibilities of our method. It has become the solution of systematic shortcomings of normal LDPC codes design.

A single base matrix  $H_0$  is designed for the longest code ( $N_0 = 2304$ ) of each code rate (1/2, 2/3 and 3/4), and the base matrices for shorter codes ( $N = 576$  to 2208) are derived from  $H_0$ .

For a shorter code size  $N$  corresponding to expansion factor  $z_N$ , the cyclic shift value  $s$  in the base matrix  $H_0$  is also proportionally adjusted according to  
(For our contribution)

$$s = s_0 \bmod z_N$$

(For Motorola's contribution)

$$s = \text{Floor}(s_0 * N / N_0)$$

where  $s_0$  is cyclic shift entry in  $H_0$  and  $\bmod$  is the module operation. Our calculation is more efficient.

## References

The following documents contain background material and source material from which the group is working. Modifications to this material are being considered as well as new material from Motorola etc, in order to achieve harmonization on the best possible code for 802.16e.

- 1 C80216e-04\_373r1 LDPC coding for OFDMA PHY, Brian Classon etc.
- 2 C80216e-04\_141r2 LDPC coding for OFDMA PHY, Eric Jacobsen etc., Intel Corporation
- 3 C80216e-04\_374 LDPC for the OFDMA PHY, Brian Classon etc., Motorola Corporation
- 4 ACHIEVING HIGH DATA RATE USING LDPC, Yufei W. Blankenship (Motorola Co.), Bo Xia (Intel Co.), Informal document(not final version)

## Features

- **Simple encoding and decoding**
- **Less average iteration numbers**
- **Good performance**
- **Eliminate error floor**

## Simulation Results

Simulation results for ZTE high girth codes of the rate 1/2, 2/3, 3/4 code families are shown in Figure 1-3. For these three rates, code sizes considered are all 576-2304. The simulation conditions are: AWGN channel, BPSK modulation, max iterations times 50, using generic floating-point belief propagation. Our high girth code has the same degree distribution (column weight, row weight) as the 373r1 code in [1], base matrices of our code have the same dimension to 373r1 code in [1]. From the simulation results we can find that our codes overcome the "error floor" phenomenon, and the BER curve of them will descend more steeply. When SNR is high, our high girth method obviously obtained an improved performance.

Performance of the ZTE design for 802.16e in AWGN channel is shown in Figure 1, 2, and 3 for rate 1/2, 2/3, 3/4. The block sizes  $n$  range from 576 to 2304 for all three code rates. The expansion factor  $z$  ranges from 24 to 96, as shown in Figures 1-3. The block size and the expansion factor are related by  $n = 24 * z$ .

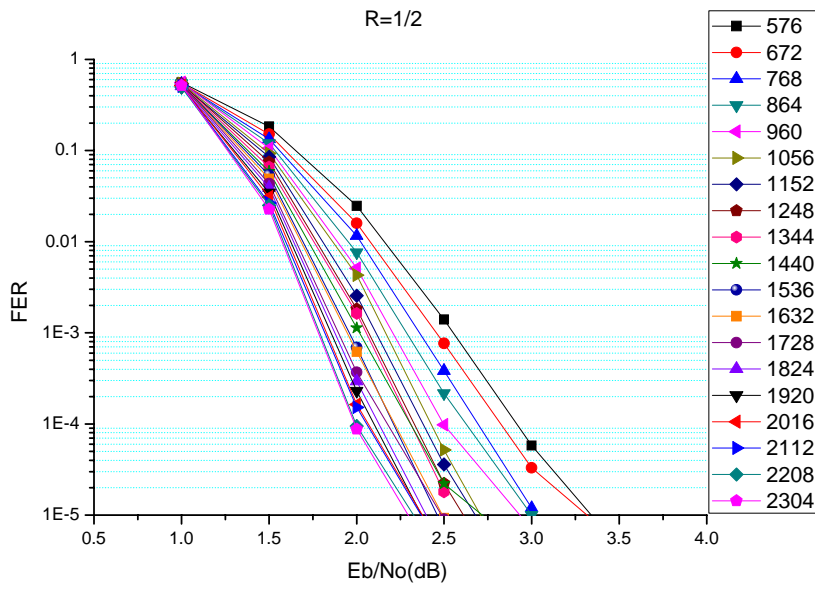


Figure 1. FER vs.  $E_b/N_0$  (dB), BPSK, rate 1/2, AWGN channel.

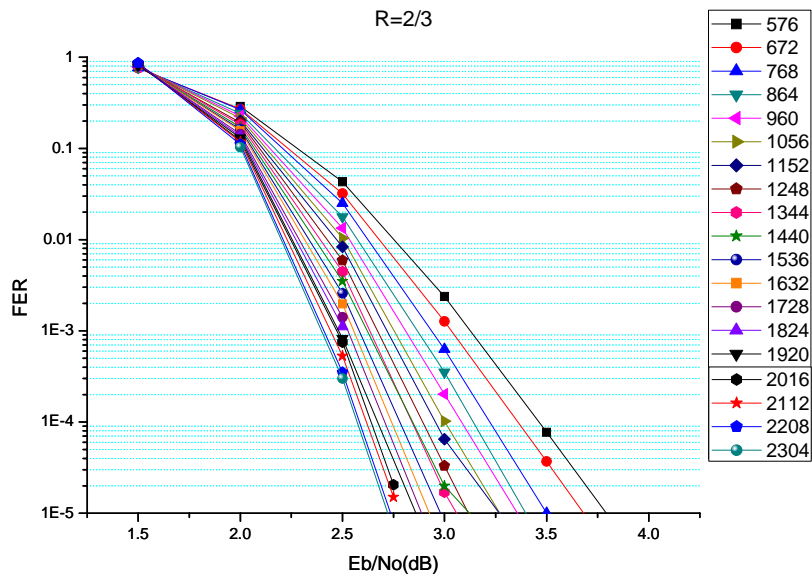


Figure 2. FER vs.  $E_b/N_0$  (dB), BPSK, rate 2/3, AWGN channel.

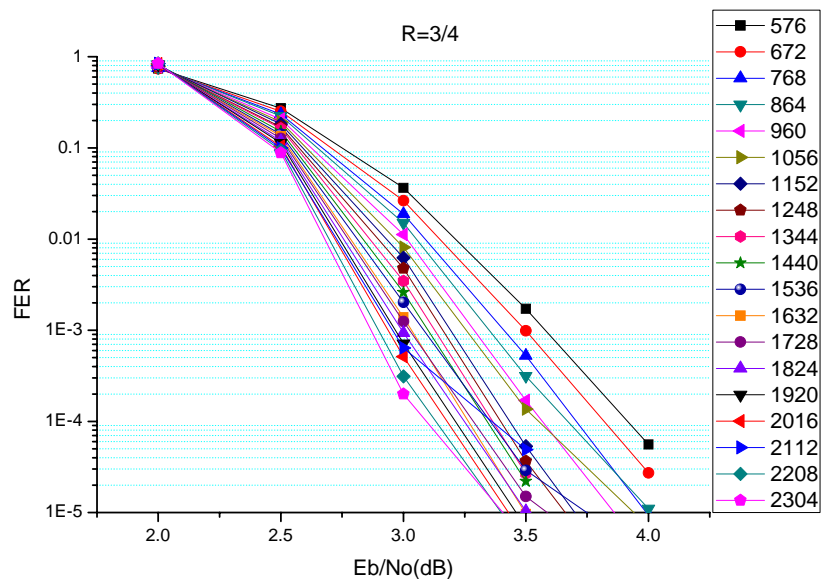


Figure 3. FER vs.  $E_b/N_0$  (dB), BPSK, rate 3/4, AWGN channel.

Simulation showed that normal LDPC codes' BER curves descend more and more slow. It is always difficult to arrive at the point  $BER = 10e-6$ , which efficient data communication needs. But high girth LDPC codes can arrive at the point easily. So, high girth LDPC codes are suitable for situations where low BER are needed.

## Recommended Text Changes:

[Add/Modify the following text to 802.16e\_D6, adjusting the numbering as required]

### 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  $\mathbf{H}$  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:

$$\mathbf{H} = \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \dots & \mathbf{P}_{0,n_b \angle 2} & \mathbf{P}_{0,n_b \angle 1} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \dots & \mathbf{P}_{1,n_b \angle 2} & \mathbf{P}_{1,n_b \angle 1} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \dots & \mathbf{P}_{2,n_b \angle 2} & \mathbf{P}_{0,n_b \angle 1} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \mathbf{P}_{m_b \angle 1,0} & \mathbf{P}_{m_b \angle 1,1} & \mathbf{P}_{m_b \angle 1,2} & \dots & \mathbf{P}_{m_b \angle 1,n_b \angle 2} & \mathbf{P}_{m_b \angle 1,n_b \angle 1} \end{bmatrix} = \mathbf{P}^{\mathbf{H}_b}$$

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  $m_b$  and  $n_b$  are integers, with  $z$  an integer  $\geq 1$ . 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 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  $\mathbf{H}_b = \left[ \begin{array}{c|c} (\mathbf{H}_{b1})_{m_b \times k_b} & (\mathbf{H}_{b2})_{m_b \times m_b} \end{array} \right]$ . 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:

$$\mathbf{H}_{b2} = \left[ \mathbf{h}_b \mid \mathbf{H}'_{b2} \right]$$

$$= \left[ \begin{array}{c|cccc} h_b(0) & 1 & & & \\ h_b(1) & 1 & 1 & & \mathbf{0} \\ \cdot & & 1 & \ddots & \\ \cdot & & & \ddots & 1 \\ \cdot & & \mathbf{0} & & 1 & 1 \\ h_b(m_b-1) & & & & & 1 \end{array} \right]$$

The base matrix has  $h_b(0)=0$ ,  $h_b(m_b-1)=0$ , and a third value  $h_b(j)$ ,  $0 < j < (m_b-1)$  equal to 1. The base matrix structure avoids having multiple weight-1 columns in the expanded matrix.

In particular, the non-zero submatrices are circularly right shifted by a particular circular shift value. Each 1 in  $\mathbf{H}'_{b2}$  is assigned a shift size of 0, and is replaced by a  $z \times z$  identity matrix when expanding to  $\mathbf{H}$ . The two 1s located at the top and the bottom of  $\mathbf{h}_b$  are assigned equal shift sizes, and the third 1 in the middle of  $\mathbf{h}_b$  is given an unpaired shift size.

### Model Matrix Set

Three block semi-regular model matrices  $\mathbf{H}_{bm}$  are defined, one each for rates 1/2, 2/3, and 3/4. For each code rate,  $\mathbf{H}_{bm}$  will be different.

A single base matrix  $H_0$  is designed for the longest code ( $N_0 = 2304$ ) of each code rate (1/2, 2/3 and 3/4), and the base matrices for shorter codes ( $N = 576$  to 2208) are derived from  $H_0$ . For a shorter code size  $N$  corresponding to expansion factor  $z_N$ , the cyclic shift value  $s$  in the base matrix  $H_0$  is also proportionally adjusted according to

$$s = s_0 \bmod z_N$$

where  $s_0$  is cyclic shift entry in  $H_0$  and mod is the module operation.

For rate 1/2, girths of the designed LDPC codes are all 8. As for rate 2/3 and 3/4, girths of the designed LDPC codes' are all 6.

### Rate 1/2:

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

```

38 -1 39 -1 76 -1 -1 -1 -1 -1 56 -1 0 0 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 90 -1 63 -1 -1 74 -1 54 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1 -1 -1 -1 -1
32 -1 -1 -1 80 31 -1 -1 -1 39 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 82 70 -1 -1 -1 -1 15 -1 72 -1 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1 -1 -1 -1
55 -1 -1 -1 -1 5 54 9 -1 -1 -1 -1 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1 -1 -1
-1 -1 62 74 -1 -1 -1 -1 -1 -1 66 26 -1 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1 -1
-1 72 -1 -1 -1 17 -1 14 -1 -1 -1 8 17 -1 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1
10 -1 -1 39 -1 -1 -1 -1 11 42 -1 -1 -1 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1
-1 57 -1 -1 70 -1 54 -1 -1 -1 44 -1 -1 -1 -1 -1 -1 -1 -1 -1 0 0 -1 -1
-1 -1 2 -1 -1 4 -1 -1 12 -1 31 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 0 0 -1
-1 -1 -1 60 65 -1 -1 12 -1 -1 -1 9 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 0 0
-1 -1 -1 -1 -1 -1 45 -1 33 47 -1 1 0 -1 -1 -1 -1 -1 -1 -1 -1 -1 0

```

**Rate 2/3:** $\mathbf{H}_{bm} =$ 

```

38 -1 39 -1 76 -1 -1 -1 -1 -1 56 -1 0 0 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 90 -1 63 -1 -1 74 -1 54 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1 -1 -1 -1 -1
32 -1 -1 -1 80 31 -1 -1 -1 39 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 82 70 -1 -1 -1 -1 15 -1 72 -1 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1 -1 -1
55 -1 -1 -1 -1 5 54 9 -1 -1 -1 -1 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1 -1 -1
-1 -1 62 74 -1 -1 -1 -1 -1 -1 66 26 -1 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1
-1 72 -1 -1 -1 17 -1 14 -1 -1 -1 8 17 -1 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1
10 -1 -1 39 -1 -1 -1 -1 11 42 -1 -1 -1 -1 -1 -1 -1 0 0 -1 -1 -1 -1 -1
-1 57 -1 -1 70 -1 54 -1 -1 -1 44 -1 -1 -1 -1 -1 -1 -1 -1 -1 0 0 -1 -1
-1 -1 2 -1 -1 4 -1 -1 12 -1 31 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 0 0 -1
-1 -1 -1 60 65 -1 -1 12 -1 -1 -1 9 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 0 0
-1 -1 -1 -1 -1 -1 45 -1 33 47 -1 1 0 -1 -1 -1 -1 -1 -1 -1 -1 -1 0

```

**Rate 3/4:** $\mathbf{H}_{bm} =$ 

```

34 74 -1 9 -1 66 59 59 -1 35 18 -1 5 25 83 -1 -1 84 0 0 -1 -1 -1 -1
91 -1 82 82 11 49 -1 5 12 -1 -1 41 13 87 45 61 -1 -1 -1 0 0 -1 -1 -1
79 57 16 -1 -1 12 92 -1 61 72 82 -1 54 -1 82 41 95 -1 -1 -1 0 0 -1 -1
-1 -1 49 18 85 -1 80 71 -1 52 13 62 -1 -1 1 66 41 95 0 -1 -1 0 0 -1
29 81 -1 21 73 -1 50 -1 69 63 -1 81 37 11 -1 -1 74 61 -1 -1 -1 -1 0 0
-1 12 37 -1 15 8 -1 5 61 -1 17 8 -1 20 -1 79 0 45 0 -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  $\mathbf{p}=(p_0, \dots, p_{m-1})$  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).

**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  $\mathbf{H}'_{b2}$ ,

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

where

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

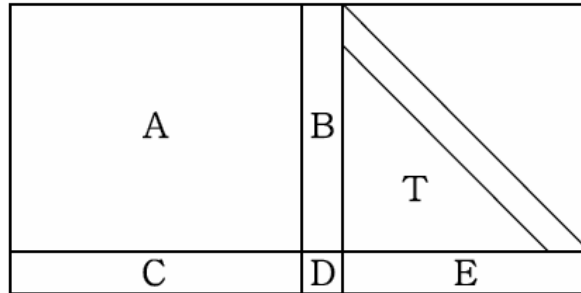
Thus all parity bits not in  $\mathbf{v}(0)$  are determined by evaluating Equation (2) for  $0 \leq i \leq m_b - 2$ . Equations (1) and (2) completely describe the encoding algorithm. These equations also have a straightforward interpretation in terms of standard digital logic architectures. Since the 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  $\mathbf{A}$  is  $(m-g) \times (n-m)$ ,  $\mathbf{B}$  is  $(m-g) \times g$ ,  $\mathbf{T}$  is  $(m-g) \times (m-g)$ ,  $\mathbf{C}$  is  $g \times (n-m)$ ,  $\mathbf{D}$  is  $g \times g$ . The basic structure of the  $\mathbf{H}$  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

$$Au^T + Bp_1^T + Tp_2^T = 0 \quad (2)$$

and

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

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

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

and

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

As a result, the encoding procedures and the corresponding operations can be summarized below and illustrated in Fig. 2.

### 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  $Tp_2^T = Au^T + Bp_1^T$ .

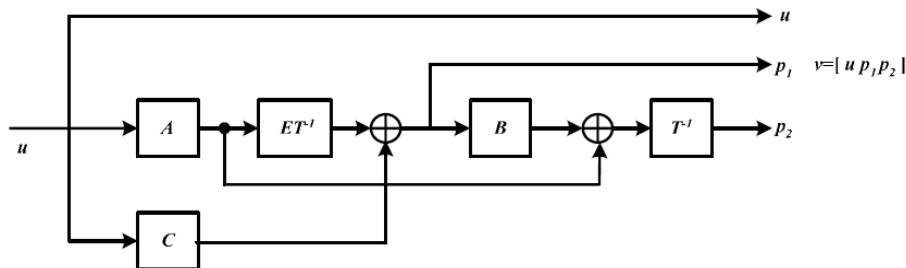


Fig. 2 Block diagram of the encoder architecture for the block LDPC code.

### 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. 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. 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.

Table 316b—LDPC 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  $\mathbf{H}$  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  $nb$ . Examples of the  $z$  expansion factors are given in the tables below. The base matrix  $nb$  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 does not depend on the code rate.

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

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

The subchannel concatenation rule for CC in Table 315 is applied, noting that in Table 315 the parameter  $n$  is equal to  $N_{\text{sch}}$ , 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.

<b>Modulation</b>	<b><math>j</math></b>
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.