IEEE Standard for Information Technology—
Telecommunications and Information Exchange between Systems
Local and Metropolitan Area Networks—
Specific Requirements

Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications

IEEE Computer Society

Developed by the
LAN/MAN Standards Committee

IEEE Std 802.11™-2020
(Revision of IEEE Std 802.11-2016)

19.3.11.6 BCC coding and puncturing

When BCC encoding is used, the encoder parser output sequences $\{x_i^0\}$, and $\{x_i^1\}$ where applicable, are each encoded by the rate 1/2 convolutional encoder defined in 17.3.5.6. After encoding, the encoded data shall be punctured by the method defined in 17.3.5.7 to achieve the rate selected by the MCS.

If rate 5/6 coding is selected, the puncturing scheme is defined in Figure 19-11.

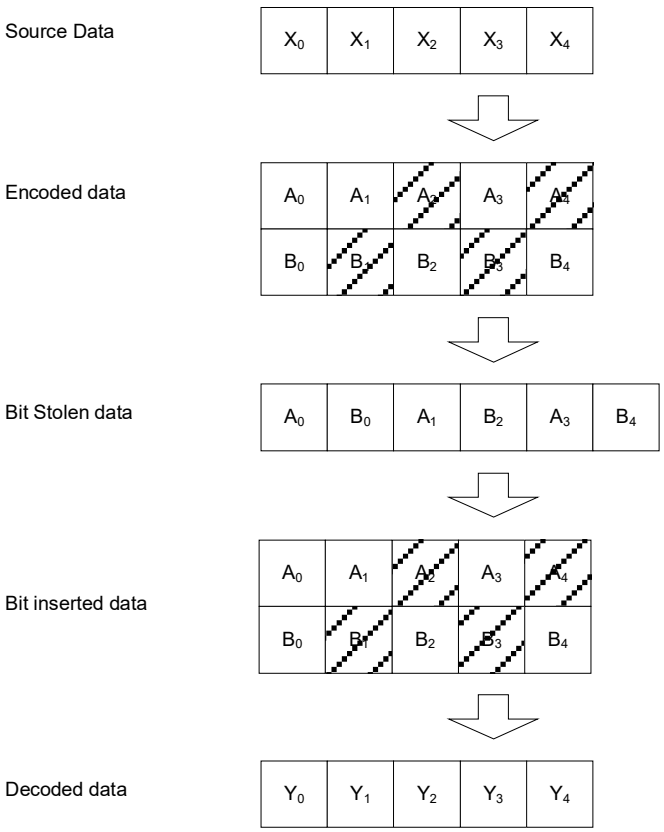

Figure 19-11—Puncturing at rate 5/6

19.3.11.7 LDPC codes

19.3.11.7.1 Introduction

HT LDPC codes are described in 19.3.11.7.2 to 19.3.11.7.6. These codes are optionally used in the HT system as a high-performance error correcting code instead of the convolutional code (19.3.11.6). The LDPC encoder shall use the rate-dependent parameters in Table 19-27 to Table 19-41, with the exception of the N_{ES} parameter.

Support for LDPC codes is optional.

19.3.11.7.2 LDPC coding rates and codeword block lengths

The supported coding rates, information block lengths, and codeword block lengths are described in Table 19-15.

Table 19-15—LDPC parameters

Coding rate (R)	LDPC information block length (bits)	LDPC codeword block length (bits)
1/2	972	1944
1/2	648	1296
1/2	324	648
2/3	1296	1944
2/3	864	1296
2/3	432	648
3/4	1458	1944
3/4	972	1296
3/4	486	648
5/6	1620	1944
5/6	1080	1296
5/6	540	648

19.3.11.7.3 LDPC encoder

For each of the three available codeword block lengths, the LDPC encoder supports rate 1/2, rate 2/3, rate 3/4, and rate 5/6 encoding. The LDPC encoder is systematic, i.e., it encodes an information block, $c=(i_0,i_1,\ldots, i_{(k-1)})$, of size k, into a codeword, **c**, of size n, $\mathbf{c}=(i_0,i_1,\ldots i_{(k-1)}, p_0, p_1,\ldots, p_{(n-k-1)})$, by adding $n-k$ parity bits obtained so that $\mathbf{H}\times\mathbf{c}^T = \mathbf{0}$, where **H** is an $(n-k)\times n$ parity-check matrix. The selection of the codeword block length (n) is achieved via the LDPC PPDU encoding process described in 19.3.11.7.5.

19.3.11.7.4 Parity-check matrices

Each of the parity-check matrices is partitioned into square subblocks (submatrices) of size $Z \times Z$. These submatrices are either cyclic-permutations of the identity matrix or null submatrices.

The cyclic-permutation matrix P_i is obtained from the $Z \times Z$ identity matrix by cyclically shifting the columns to the right by i elements. The matrix P_0 is the $Z \times Z$ identity matrix. Figure 19-12 illustrates examples (for a subblock size of 8 × 8) of cyclic-permutation matrices P_i.

Table F-1 displays the "matrix prototypes" of parity-check matrices for all four coding rates at block length n=648 bits. The integer i denotes the cyclic-permutation matrix P_i, as illustrated in Figure 19-12. Vacant entries of the table denote null (zero) submatrices.

Table F-2 displays the matrix prototypes of parity-check matrices for block length n=1296 bits, in the same fashion.

Table F-3 displays the matrix prototypes of parity-check matrices for block length n=1944 bits, in the same fashion.

$$P_0 = \begin{bmatrix} 1&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \end{bmatrix}, P_1 = \begin{bmatrix} 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \\ 1&0&0&0&0&0&0&0 \end{bmatrix}, P_5 = \begin{bmatrix} 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \\ 1&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \end{bmatrix}$$

Figure 19-12—Examples of cyclic-permutation matrices with *Z*=8

19.3.11.7.5 LDPC PPDU encoding process

To encode an LDPC PPDU, step a) to step g) shall be performed in sequence:

a) Compute the number of available bits, N_{avbits}, in the minimum number of OFDM symbols in which the Data field of the packet may fit.

$$N_{pld} = length \times 8 + 16 \tag{19-35}$$

$$N_{avbits} = N_{CBPS} \times m_{STBC} \times \left\lceil \frac{N_{pld}}{N_{CBPS} \times R \times m_{STBC}} \right\rceil \tag{19-36}$$

where

m_{STBC} is 2 if STBC is used and 1 otherwise
length is the value of the HT Length field in the HT-SIG field defined in Table 19-11
N_{pld} is the number of bits in the PSDU and SERVICE field

b) Compute the integer number of LDPC codewords to be transmitted, N_{CW}, and the length of the codewords to be used, L_{LDPC} from Table 19-16.

Table 19-16—PPDU encoding parameters

Range of N_{avbits} (bits)	Number of LDPC codewords (N_{CW})	LDPC codeword length L_{LDPC} (bits)
$N_{avbits} \le 648$	1	1296, if $N_{avbits} \ge N_{pld} + 912 \times (1\text{-}R)$ 648, otherwise
$648 < N_{avbits} \le 1296$	1	1944, if $N_{avbits} \ge N_{pld} + 1464 \times (1\text{-}R)$ 1296, otherwise
$1296 < N_{avbits} \le 1944$	1	1944
$1944 < N_{avbits} \le 2592$	2	1944, if $N_{avbits} \ge N_{pld} + 2916 \times (1\text{-}R)$ 1296, otherwise
$2592 < N_{avbits}$	$\left\lceil \frac{N_{pld}}{1944 \cdot R} \right\rceil$	1944

Annex F

(normative)

HT LDPC matrix definitions

Table F-1 defines the matrix prototypes of the parity-check matrices for a codeword block length n = 648 bits, with a subblock size Z = 27 bits.

Table F-1—Matrix prototypes for codeword block length *n* = 648 bits, subblock size is *Z* = 27 bits

(a) Coding rate R = 1/2.

0	–	–	–	0	0	–	–	0	–	–	0	1	0	–	–	–	–	–	–	–	–	–	–
22	0	–	–	17	–	0	0	12	–	–	–	–	0	0	–	–	–	–	–	–	–	–	–
6	–	0	–	10	–	–	–	24	–	0	–	–	–	0	0	–	–	–	–	–	–	–	–
2	–	–	0	20	–	–	–	25	0	–	–	–	–	–	0	0	–	–	–	–	–	–	–
23	–	–	–	3	–	–	–	0	–	9	11	–	–	–	–	0	0	–	–	–	–	–	–
24	–	23	1	17	–	3	–	10	–	–	–	–	–	–	–	–	0	0	–	–	–	–	–
25	–	–	–	8	–	–	–	7	18	–	–	0	–	–	–	–	–	0	0	–	–	–	–
13	24	–	–	0	–	8	–	6	–	–	–	–	–	–	–	–	–	–	0	0	–	–	–
7	20	–	16	22	10	–	–	23	–	–	–	–	–	–	–	–	–	–	–	0	0	–	–
11	–	–	–	19	–	–	–	13	–	3	17	–	–	–	–	–	–	–	–	–	0	0	–
25	–	8	–	23	18	–	14	9	–	–	–	–	–	–	–	–	–	–	–	–	–	0	0
3	–	–	–	16	–	–	2	25	5	–	–	1	–	–	–	–	–	–	–	–	–	–	0

(b) Coding rate R = 2/3.

25	26	14	–	20	–	2	–	4	–	–	8	–	16	–	18	1	0	–	–	–	–	–	–
10	9	15	11	–	0	–	1	–	–	18	–	8	–	10	–	–	0	0	–	–	–	–	–
16	2	20	26	21	–	6	–	1	26	–	7	–	–	–	–	–	–	0	0	–	–	–	–
10	13	5	0	–	3	–	7	–	–	26	–	–	13	–	16	–	–	–	0	0	–	–	–
23	14	24	–	12	–	19	–	17	–	–	–	20	–	21	–	0	–	–	–	0	0	–	–
6	22	9	20	–	25	–	17	–	8	–	14	–	18	–	–	–	–	–	–	–	0	0	–
14	23	21	11	20	–	24	–	18	–	19	–	–	–	–	22	–	–	–	–	–	–	0	0
17	11	11	20	–	21	–	26	–	3	–	–	18	–	26	–	1	–	–	–	–	–	–	0

(c) Coding rate R = 3/4.

16	17	22	24	9	3	14	–	4	2	7	–	26	–	2	–	21	–	1	0	–	–	–	–
25	12	12	3	3	26	6	21	–	15	22	–	15	–	4	–	–	16	–	0	0	–	–	–
25	18	26	16	22	23	9	–	0	–	4	–	4	–	8	23	11	–	–	–	0	0	–	–
9	7	0	1	17	–	–	7	3	–	3	23	–	16	–	–	21	–	0	–	–	0	0	–
24	5	26	7	1	–	–	15	24	15	–	8	–	13	–	13	–	11	–	–	–	–	0	0
2	2	19	14	24	1	15	19	–	21	–	2	–	24	–	3	–	2	1	–	–	–	0	

(d) Coding rate R = 5/6.

17	13	8	21	9	3	18	12	10	0	4	15	19	2	5	10	26	19	13	13	1	0	–	–
3	12	11	14	11	25	5	18	0	9	2	26	26	10	24	7	14	20	4	2	–	0	0	–
22	16	4	3	10	21	12	5	21	14	19	5	–	8	5	18	11	5	5	15	0	–	0	0
7	7	14	14	4	16	16	24	24	10	1	7	15	6	10	26	8	18	21	14	1	–	–	0

Table F-2 defines the matrix prototypes of the parity-check matrices for a codeword block length n = 1296 bits, with a subblock size Z = 54 bits.

Table F-2—Matrix prototypes for codeword block length *n* = 1296 bits, subblock size is *Z* = 54 bits

(a) Coding rate R = 1/2.

40	–	–	–	22	–	49	23	43	–	–	–	1	0	–	–	–	–	–	–	–	–	–	–
50	1	–	–	48	35	–	–	13	–	30	–	–	0	0	–	–	–	–	–	–	–	–	–
39	50	–	–	4	–	2	–	–	–	–	49	–	–	0	0	–	–	–	–	–	–	–	–
33	–	–	38	37	–	–	4	1	–	–	–	–	–	–	0	0	–	–	–	–	–	–	–
45	–	–	–	0	22	–	–	20	42	–	–	–	–	–	–	0	0	–	–	–	–	–	–
51	–	–	48	35	–	–	–	44	–	18	–	–	–	–	–	–	0	0	–	–	–	–	–
47	11	–	–	–	17	–	–	51	–	–	–	0	–	–	–	–	–	0	0	–	–	–	–
5	–	25	–	6	–	45	–	13	40	–	–	–	–	–	–	–	–	–	0	0	–	–	–
33	–	–	34	24	–	–	–	23	–	–	46	–	–	–	–	–	–	–	–	0	0	–	–
1	–	27	–	1	–	–	–	38	–	44	–	–	–	–	–	–	–	–	–	–	0	0	–
–	18	–	–	23	–	–	8	0	35	–	–	–	–	–	–	–	–	–	–	–	–	0	0
49	–	17	–	30	–	–	–	34	–	–	19	1	–	–	–	–	–	–	–	–	–	–	0

(b) Coding rate R = 2/3.

39	31	22	43	–	40	4	–	11	–	–	50	–	–	–	6	1	0	–	–	–	–	–	–
25	52	41	2	6	–	14	–	34	–	–	–	24	–	37	–	–	0	0	–	–	–	–	–
43	31	29	0	21	–	28	–	–	2	–	–	7	–	17	–	–	–	0	0	–	–	–	–
20	33	48	–	4	13	–	26	–	–	22	–	–	46	42	–	–	–	–	0	0	–	–	–
45	7	18	51	12	25	–	–	–	50	–	–	5	–	–	–	0	–	–	–	0	0	–	–
35	40	32	16	5	–	–	18	–	–	43	51	–	32	–	–	–	–	–	–	–	0	0	–
9	24	13	22	28	–	–	37	–	–	25	–	–	52	–	13	–	–	–	–	–	–	0	0
32	22	4	21	16	–	–	–	27	28	–	38	–	–	–	8	1	–	–	–	–	–	–	0

(c) Coding rate R = 3/4.

39	40	51	41	3	29	8	36	–	14	–	6	–	33	–	11	–	4	1	0	–	–	–	–
48	21	47	9	48	35	51	–	38	–	28	–	34	–	50	–	50	–	–	0	0	–	–	–
30	39	28	42	50	39	5	17	–	6	–	18	–	20	–	15	–	40	–	–	0	0	–	–
29	0	1	43	36	30	47	–	49	–	47	–	3	–	35	–	34	–	0	–	–	0	0	–
1	32	11	23	10	44	12	7	–	48	–	4	–	9	–	17	–	16	–	–	–	–	0	0
13	7	15	47	23	16	47	–	43	–	29	–	52	–	2	–	53	–	1	–	–	–	–	0

(d) Coding rate R = 5/6.

48	29	37	52	2	16	6	14	53	31	34	5	18	42	53	31	45	–	46	52	1	0	–	–
17	4	30	7	43	11	24	6	14	21	6	39	17	40	47	7	15	41	19	–	–	0	0	–
7	2	51	31	46	23	16	11	53	40	10	7	46	53	33	35	–	25	35	38	0	–	0	0
19	48	41	1	10	7	36	47	5	29	52	52	31	10	26	6	3	2	–	51	1	–	–	0

Table F-3 defines the matrix prototypes of the parity-check matrices for a codeword block length n = 1944 bits, with a subblock size Z = 81 bits.

Table F-3—Matrix prototypes for codeword block length *n* = 1944 bits, subblock size is *Z* = 81 bits

(a) Coding rate R = 1/2.

57	-	-	-	50	-	11	-	50	-	79	-	1	0	-	-	-	-	-	-	-	-	-	-
3	-	28	-	0	-	-	-	55	7	-	-	-	0	0	-	-	-	-	-	-	-	-	-
30	-	-	-	24	37	-	-	56	14	-	-	-	-	0	0	-	-	-	-	-	-	-	-
62	53	-	-	53	-	-	3	35	-	-	-	-	-	-	0	0	-	-	-	-	-	-	-
40	-	-	20	66	-	-	22	28	-	-	-	-	-	-	-	0	0	-	-	-	-	-	-
0	-	-	-	8	-	42	-	50	-	-	8	-	-	-	-	-	0	0	-	-	-	-	-
69	79	79	-	-	-	56	-	52	-	-	-	0	-	-	-	-	-	0	0	-	-	-	-
65	-	-	-	38	57	-	-	72	-	27	-	-	-	-	-	-	-	-	0	0	-	-	-
64	-	-	-	14	52	-	-	30	-	-	32	-	-	-	-	-	-	-	-	0	0	-	-
-	45	-	70	0	-	-	-	77	9	-	-	-	-	-	-	-	-	-	-	-	0	0	-
2	56	-	57	35	-	-	-	-	-	12	-	-	-	-	-	-	-	-	-	-	-	0	0
24	-	61	-	60	-	-	27	51	-	-	16	1	-	-	-	-	-	-	-	-	-	-	0

(b) Coding rate R = 2/3.

61	75	4	63	56	-	-	-	-	-	-	8	-	2	17	25	1	0	-	-	-	-	-	-
56	74	77	20	-	-	-	64	24	4	67	-	7	-	-	-	-	0	0	-	-	-	-	-
28	21	68	10	7	14	65	-	-	-	23	-	-	-	75	-	-	-	0	0	-	-	-	-
48	38	43	78	76	-	-	-	-	5	36	-	15	72	-	-	-	-	-	0	0	-	-	-
40	2	53	25	-	52	62	-	20	-	-	44	-	-	-	-	0	-	-	-	0	0	-	-
69	23	64	10	22	-	21	-	-	-	-	-	68	23	29	-	-	-	-	-	-	0	0	-
12	0	68	20	55	61	-	40	-	-	-	52	-	-	-	44	-	-	-	-	-	-	0	0
58	8	34	64	78	-	-	11	78	24	-	-	-	-	-	58	1	-	-	-	-	-	-	0

(c) Coding rate R = 3/4.

48	29	28	39	9	61	-	-	-	63	45	80	-	-	-	37	32	22	1	0	-	-	-	-
4	49	42	48	11	30	-	-	-	49	17	41	37	15	-	54	-	-	-	0	0	-	-	-
35	76	78	51	37	35	21	-	17	64	-	-	-	59	7	-	-	32	-	-	0	0	-	-
9	65	44	9	54	56	73	34	42	-	-	-	35	-	-	-	46	39	0	-	-	0	0	-
3	62	7	80	68	26	-	80	55	-	36	-	26	-	9	-	72	-	-	-	-	-	0	0
26	75	33	21	69	59	3	38	-	-	-	35	-	62	36	26	-	-	1	-	-	-	-	0

(d) Coding rate R = 5/6.

13	48	80	66	4	74	7	30	76	52	37	60	-	49	73	31	74	73	23	-	1	0	-	-
69	63	74	56	64	77	57	65	6	16	51	-	64	-	68	9	48	62	54	27	-	0	0	-
51	15	0	80	24	25	42	54	44	71	71	9	67	35	-	58	-	29	-	53	0	-	0	0
16	29	36	41	44	56	59	37	50	24	-	65	4	65	52	-	4	-	73	52	1	-	-	0

Annex G

(informative)

LDPC direct encoding

The LDPC 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 $\mathbf{p}=(p_0, \ldots, p_{m-1})$ based on an information block $\mathbf{s}=(s_0, \ldots, s_{k-1})$, 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 $\mathbf{s} = (s_0, \ldots, s_{k-1})$ and uses the matrix $\mathbf{H}_{\text{bm}}$ to determine the parity-check bits. The expanded matrix $\mathbf{H}$ is determined from the model matrix $\mathbf{H}_{\text{bm}}$. 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 $\mathbf{H}$ so that $\mathbf{G}\,\mathbf{H}^{\mathrm{T}} = \mathbf{0}$. 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 $\mathbf{x}=[\mathbf{s}\ \mathbf{p}]=[s_0, s_1, \ldots, s_{k-1}, p_0, p_1, \ldots, p_{m-1}]$, where $p_0, \ldots p_{m-1}$ are the parity-check bits; and $s_0, \ldots s_{k-1}$ are the systematic bits.

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

The following informative subclauses show two such methods.

G.1 Method 1a

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 $\mathbf{u}$,

$$u = \left[u(0)\ u(1)\ \ldots\ u(k_b - 1)\right],$$

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

$$u(i) = \left[S_{iz}\ S_{iz+1}\ \ldots\ S_{(i+1)z-1}\right]^T$$

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

$$v = \left[v(0)\ v(1)\ \ldots\ v(m_b - 1)\right],$$

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

$$v(i) = \left[p_{iz}\ p_{iz+1}\ \ldots\ p_{(i+1)z-1}\right]^T$$

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

An expression for **v**(0) can be derived by summing over the rows of $\mathbf{H}_{\text{bm}}$ to obtain Equation (G.1).

$$P_{p(x,k_b)}v(0) = \sum_{j=0}^{k_b-1}\sum_{i=0}^{m_b-1} P_{p(i,j)}u(j) \tag{G.1}$$

where x, $1 \le x \le m_b - 2$, is the row index of $\mathbf{h}_{\text{bm}}$ where the entry is nonnegative and unpaired, and $\mathbf{P}_i$ represents the zxz identity matrix circularly right shifted by size i.

Equation (G.2) is solved for $\boldsymbol{v}(0)$ by multiplying by $P^{-1}{}_{p(x,k_b)}$, and $P^{-1}{}_{p(x,k_b)} = P_{z-p(x,k_b)}$ since $p(x,k_b)$ represents a circular shift.

Considering the structure of $\mathbf{H}'_{\text{b2}}$, the recursion can be derived as follows in Equation (G.2) and Equation (G.3).

$$v(1) = \sum_{j=0}^{k_b-1} P_{p(i,j)}u(j) + P_{p(i,k_b)})v(0), i = 0, \tag{G.2}$$

$$v(i+1) = v(i) + \sum_{j=0}^{k_b-1} P_{p(i,j)}u(j) + P_{p(i,k_b)}v(0), i = 1, \ldots, m_b - 2 \tag{G.3}$$

where

$$P_{-1} \equiv 0_{z\times z}$$

Thus all parity bits not in $v(0)$ are determined by evaluation Equation (G.3) for $0 \le i \le m_b - 2$. Equation (G.1), Equation (G.2), and Equation (G.3) completely describe the encoding algorithm. These equations also have a straightforward interpretation in terms of standard digital logic architectures. Since the nonzero elements $p(i,j)$ of $\mathbf{H}_{\text{bm}}$ represent circular shift sizes of a vector, all products of the form $\mathbf{P}_{p(i,j)}\boldsymbol{u(j)}$ can be implemented by a size-z barrel shifter.

G.2 Method 1b

Equivalently, Method 1 can be implemented in a parallel fashion where almost all parity check parity bits are generated simultaneously. The initialization and the recursion steps of Method 1 become

a) *Initialization.* The parity check bit vector $v(0)$ are computed by Equation (G.4).

$$P_{p(x,k_b)}v(0) = \sum_{j=0}^{k_b-1}\left(\sum_{q=0}^{m_b-1} P_{p(q,j)}\right)u(j) \tag{G.4}$$

b) *Parallel computation.* The parity check bit vectors $v(1) \sim v(m_b - 1)$ are concurrently computed by Equation (G.5).

$$v(i) = \sum_{j=0}^{k_b-1}\left(\sum_{q=i}^{m_b-1} P_{p(q,j)}\right)u(j) + \sum_{q=i}^{m_b-1} P_{p(q,k_b)}v(0)\ i = 1, \ldots, m_b - 1 \tag{G.5}$$

The parallel encoding method may significantly reduced the latency at the expense of extra storage for the sum

$$\left(\sum_{q=i}^{m_b - 1} P_{p(q,j)} \right)$$

G.3 Method 2

For efficient encoding of LDPC, **H** are divided into the form of Equation (G.6).

$$H = \begin{bmatrix} A & B & T \\ C & D & E \end{bmatrix} \tag{G.6}$$

where A is $(m-z) \times k\alpha$, B is $(m-z) \times z$, T is $(m-z) \times (m-z)$, C is $z \times k$, D is $z \times z$, and finally E is $z \times (m-z)$. $\begin{pmatrix} B \\ D \end{pmatrix}$ and **D** correspond to the expanded $\mathbf{h}_b$ and $\mathbf{h}_b(m_b - 1)$, respectively.

Let $v = (u, p_1, p_2)$ where u denotes the systematic part, $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ combined denote the parity part, $\boldsymbol{p}_1$ has length **z**, and $\boldsymbol{p}_2$ has length $(m-z)$. The definition equation $(H \cdot v^t) = 0$ splits into two equations, as in Equation (G.7) and Equation (G.8).

$$Au^T + Bp_1{}^T + Tp_2{}^T = 0 \tag{G.7}$$

and

$$(ET^{-1}A + C)u^T + (ET^{-1}B + D)p_1{}^T = 0 \tag{G.8}$$

Define $\phi := (ET^{-1}B + D)$ and with the parity check matrix as indicated $\phi = I$. Then from Equation (G.8), it can be concluded that

$$P_1{}^T = (ET^{-1}A + C)u^T \tag{G.9}$$

and

$$P_2{}^T = T^{-1}(Au^T + Bp_1{}^T) \tag{G.10}$$

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

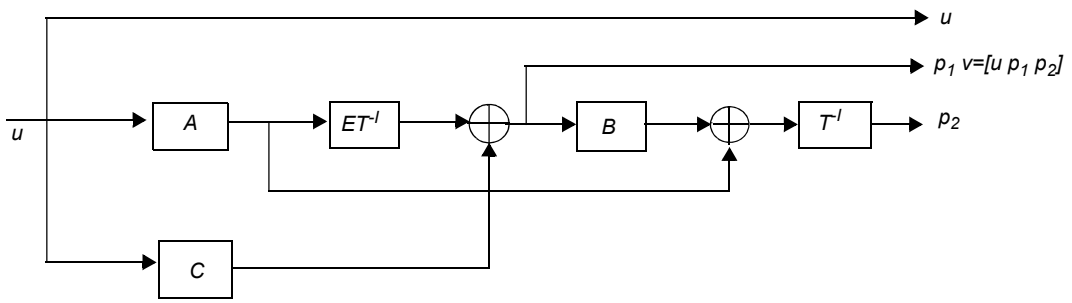

Figure G-1—Block diagram of the encoder architecture for the block LDPC code

G.4 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 $Tp_2{}^T = Au^T + Bp_1{}^T$