



- | | |
|---|---|
| <p>(51) International Patent Classification:
 <i>H03M 13/11</i> (2006.01) <i>H03M 13/29</i> (2006.01)</p> <p>(21) International Application Number:
 PCT/EP2017/071793</p> <p>(22) International Filing Date:
 30 August 2017 (30.08.2017)</p> <p>(25) Filing Language: English</p> <p>(26) Publication Language: English</p> <p>(71) Applicant: HUAWEI TECHNOLOGIES CO., LTD.
 [CN/CN]; Huawei Administration Building Bantian Long-gang District, Shenzhen, Guangdong 518129 (CN).</p> <p>(72) Inventor; and</p> <p>(71) Applicant (for US only): MANFREDI, Marco [IT/DE];
 c/o Huawei Technologies Duesseldorf GmbH, Riesstr. 25,
 80992 Munich (DE).</p> | <p>(72) Inventors: CANNALIRE, Giacomo; c/o Huawei Tech-nologies Duesseldorf GmbH, Riesstr. 25, 80992 Munich (DE). MAZZUCCO, Christian; c/o Huawei Technologies Duesseldorf GmbH, Riesstr. 25, 80992 Munich (DE).</p> <p>(74) Agent: KREUZ, Georg; Huawei Technologies Duessel-dorf GmbH, Riesstr. 8, 80992 Munich (DE).</p> <p>(81) Designated States (<i>unless otherwise indicated, for every kind of national protection available</i>): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.</p> |
|---|---|

(54) **Title:** DECODING OF LOW-DENSITY PARITY-CHECK CONVOLUTIONAL TURBO CODES

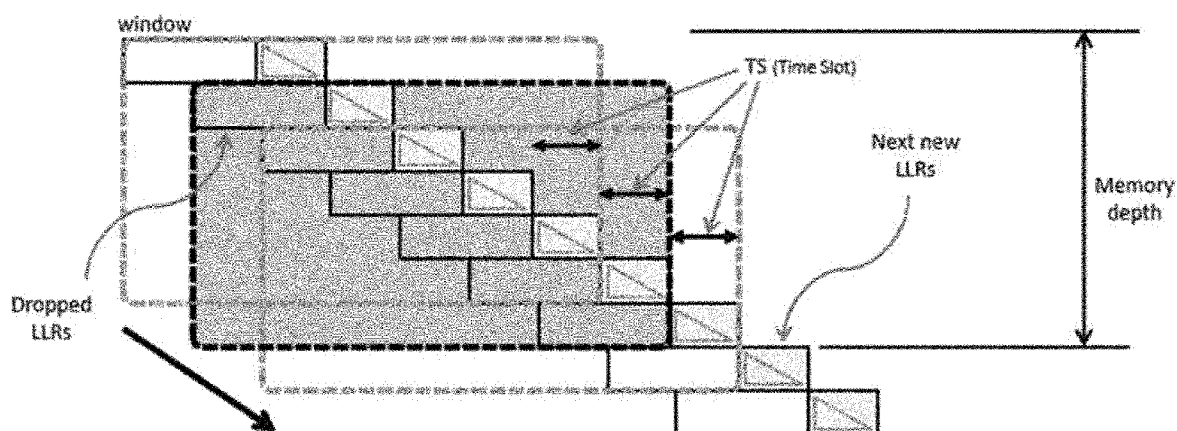


Fig. 9

(S7) Abstract: Provided is a procedure for decoding low-density parity-check convolutional turbo codes, LDPC-CTC, the LDPC-CTC being constructed from parallel LDPC-CC. The decoding procedure based on slots or windows (sliding window decoding) involves determining, based on a received signal, statistical values, e.g. LLRs, corresponding to information and parity bits of an encoded information bit sequence. If a number of determined statistical values equals or exceeds a threshold, a new decoding iteration is started, wherein a decoding iteration includes updating the determined statistical values based on actual determined statistical values and statistical values from a previous decoding iteration stored in a memory of the decoder. A stored statistical value stored in the memory is replaced with an updated statistical value, wherein the updated statistical value is kept in the memory for a number of succeeding decoding iterations. The information bits to which the updated stored statistical values correspond are decoded. If the latency/memory requirements are to be adapted according to circumstances/transmission requirement, the decoder can modify the memory depth.

(84) Designated States (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

DECODING OF LOW-DENSITY PARITY-CHECK CONVOLUTIONAL TURBO CODES

FIELD

- 5 The present disclosure relates to low-density parity-check convolutional turbo codes (LDPC-CTCs). In particular, the present disclosure relates to a procedure for decoding LDPC-CTCs.

BACKGROUND

LDPC-CCs suggested by A. J. Felstrom and K. Zigangirov, “*Time-varying periodic convolutional codes with low density parity-check matrix*”, IEEE TRANSACTIONS ON
10 INFORMATION THEORY, vol. 45, No. 6, pp. 2181-2191, Sept. 1999, are based on a binary parity-check matrix which has a column weight that is not equal to ‘1’. For this reason, it may be unfeasible to employ the parallel layered decoding algorithm already used for LDPC block codes (LDPC-BCs), as there can be conflicts regarding the reading and writing of the received signal vector.

15 Rather, decoding of LDPC-CCs suggested by A. J. Felstrom and K. Zigangirov may be carried-out based on:

- serial layered decoding, which decodes the same information sequence, and parallel layered decoding, which decodes different information sequences (this provides for good performance/fast convergence of the belief propagation algorithm, but suffers
20 from high decoding latency); or
- parallel layered decoding, which decodes the same information sequence (this provides for low decoding latency but does not achieve good performance - slow convergence of the belief propagation algorithm).

Accordingly, there is a demand for more advanced decoding procedures.

SUMMARY

In the following, an improved procedure for decoding LDPC-CTCs is described. To exploit the small structural latency of LDPC-CTC, the inner core of the decoder may be based on a simplified version of the standard belief propagation algorithm (BPA) (which is usually adopted in LDPC-BC), but the management of the IN-OUT memories may be modified.

E.g., if a time slot (TS) is the period to receive a block of wc (rate= b/c) bits (where w is the number of “parallel” LDPC-CCs on which the LDPC-CTC is based), a single decoding iteration may be performed starting from the last new log likelihood ratios (LLRs) acquired in the current TS and involve updating all LLRs up to the last LLRs memorized (e.g., LLRs of several code periods, depending on the memory depth). As the number of rows to iterate is fixed, the processing time is fixed. Old LLRs in the memory are iterated more times and may be dropped out from the memory according to latency/performance as required.

In this decoding procedure, which may be referred to as a “sliding-window” decoding procedure, the latency of the decoding process can be modified by the decoder in a way that is transparent to the encoder and the size of the memory used during decoding can be adapted by changing the number of iterations LLRs that are kept in the memory.

According to a first aspect of the present invention, there is provided a decoder for decoding a LDPC-CTC, the decoder being configured to determine, based on a received signal, statistical values corresponding to information and parity bits of an encoded information bit sequence, start a new decoding iteration if a number of determined statistical values equals or exceeds a threshold, wherein a decoding iteration includes updating the determined statistical values based on actual determined statistical values and statistical values from a previous decoding iteration stored in a memory of the decoder, replace a stored statistical value stored in the memory with an updated statistical value, wherein the updated statistical value is kept in the memory for a number of succeeding decoding iterations, and decode the information bits to which the updated stored statistical values correspond.

In this regard, it is noted that the term “decoder” as used throughout the description and claims in particular refers to hardware, or a combination of hardware and software which implements

a decoding procedure. In particular the decoder may include a processor or a processing circuit configured to perform the decoding operations or steps. Moreover, the term “received signal” as used throughout the description and claims in particular refers to a channel output, e.g., an electromagnetic signal such as a modulated carrier wave. Furthermore, the term “statistical value” as used throughout the description and claims in particular refers to reliability values corresponding to the received information and parity bits, wherein a reliability value (e.g., a likelihood, likelihood ratio, or log likelihood ratio) indicates the reliability of making a correct decision if mapping (a part of) the received signal to a particular bit value.

This “sliding-window” decoding procedure provides for low latency and reduces memory requirements. In fact, the latency of the decoding procedure can be instantly/independently modified by the decoder (during decoding) based on performance considerations, without a need to provide feedback/communicate a change of the encoding matrix to the encoder. Moreover, unlike known decoding procedures for LDPC-BCs, the size of the parity check matrix of the LDPC-CTC is substantially independent of code-word length. Furthermore, these benefits can be achieved without (substantially) increasing the hardware complexity of the decoder as compared to a decoder for LDPC BCs.

In a first possible implementation form of the decoder according to the first aspect, the decoder is further configured to remove/overwrite the updated stored statistical values from/in the memory of the decoder after the succeeding decoding iterations have been performed and the information bits corresponding to the updated stored statistical value have been decoded.

Hence, the overall need for memory space can be reduced by re-using memory space freed from updated stored statistical values that drop out of memory.

In a second possible implementation form of the decoder according to the first aspect, the decoder is further configured to adapt the number of succeeding decoding operations for which the updated statistical values are kept in the memory of the decoder.

Hence, the usage of memory space can be reduced by reducing the number of TSs for which statistical values are kept in the memory.

In a third possible implementation form of the decoder according to the first aspect, the decoder is further configured to decrease the number of succeeding decoding operations for which the updated statistical values are kept in the memory of the decoder in response to a request to reduce a latency of the decoding operation.

- 5 Thus, the latency of the decoding procedure can be reduced.

In a fourth possible implementation form of the decoder according to the first aspect, the decoder is further configured to increase the number of succeeding decoding operations for which the updated statistical values are kept in the memory of the decoder in response to a request to reduce an error rate of the decoding operation.

- 10 Thus, the performance of the decoding procedure can be improved.

In a fifth possible implementation form of the decoder according to the first aspect, the threshold equals the number of information and parity bits of one layer of the LDPC-CTC.

In this regard, it is noted that the term “layer” as used throughout the description and claims in particular refers to one row for each LDPC-CC matrix on which the LDPC-CTC is based.

- 15 In a sixth possible implementation form of the decoder according to the first aspect, updating the statistical values based on the statistical values and statistical values from a previous decoding iteration stored in a memory of the decoder includes parallelly updating different sets of statistical values assigned to different parity bits.

Hence, performance of the decoding procedure can be further improved.

- 20 Furthermore, the implementation forms of the decoder may include, but are not limited to, one or more processor, one or more application specific integrated circuit (ASIC) and/or one or more field programmable gate array (FPGA). Implementation forms of the decoder may also include other conventional and/or customized hardware such as software programmable processors.

According to a second aspect of the present invention, there is provided a method of decoding a LDPC-CTC the method comprising determining, based on a received signal, statistical values corresponding to information and parity bits of an encoded information bit sequence, starting a new decoding iteration if a number of determined statistical values equals or exceeds a threshold, wherein a decoding iteration includes updating the determined statistical values based on actual determined statistical values and statistical values from a previous decoding iteration, replacing a stored statistical value stored in the memory with an updated statistical value, wherein the updated statistical value is kept in the memory for a number of succeeding decoding iterations, and decoding the information bits to which the updated stored statistical values correspond.

Thus, as indicated above, the latency of the decoding procedure can be instantly/independently modified (during decoding) based on performance considerations, without a need to adapt the encoding procedure. Moreover, unlike known decoding procedures for LDPC-BCs, the size of the parity check matrix of the LDPC-CTC is substantially independent of code-word length.

In a first possible implementation form of the method according to the second aspect, the method further comprises removing/overwriting the updated stored statistical values from/in the memory after the succeeding decoding iterations have been performed and the information bits corresponding to the updated stored statistical value have been decoded.

Hence, the overall need for memory space can be reduced by re-using memory space freed from updated stored statistical values that drop out of memory.

In a second possible implementation form of the method according to the second aspect, the method further comprises adapting the number of succeeding decoding operations for which the updated statistical values are kept in the memory.

Hence, the usage of memory space can be reduced by reducing the number of decoding iterations for which statistical values are kept in the memory.

In a third possible implementation form of the method according to the second aspect, the method further comprises decreasing the number of succeeding decoding operations for which

the updated statistical values are kept in the memory in response to a request to reduce a latency of the decoding operation.

Thus, the latency introduced by decoding can be reduced.

5 In a fourth possible implementation form of the method according to the second aspect, the method further comprises increasing the number of succeeding decoding operations for which the updated statistical values are kept in the memory in response to a request to reduce an error rate of the decoding operation.

Thus, decoding performance can be improved.

10 In a fifth possible implementation form of the method according to the second aspect, the threshold equals the number of information and parity bits of one layer of the LDPC-CTC.

Thus, a decoding operation starts as soon as all bits of the first layer are received.

15 In a sixth possible implementation form of the method according to the second aspect, updating the statistical values based on the statistical values and statistical values from a previous decoding iteration stored in a memory of the decoder includes parallelly updating different sets of statistical values assigned to different parity bits.

Hence, decoding performance can be further improved.

Moreover, it will be appreciated that method steps and decoder features may be interchanged in many ways. In particular, the features of the disclosed decoder can be part of the method and the method steps can be implemented by the decoder.

20 According to a third aspect of the present invention, there is provided a machine-readable medium comprising machine-readable instructions which when carried-out by the machine cause the machine to perform the method according to a second aspect or any one of the implementation forms of the method according to a second aspect.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 illustrates horizontal scheduling on a simplex matrix.

Fig. 2 shows a block diagram of a calculus or decoding unit for a turbo decoding message passing procedure based on the normalized min sum algorithm.

5 Fig. 3 shows a block diagram of a digital communication system in which the decoding procedure of the present invention may be implemented.

Fig. 4 illustrates an exemplary encoding procedure which may be performed by an LDPC-CTC encoder of the digital communication system of Fig. 3.

Fig. 5 illustrates an example of the encoding procedure for an LDPC-CTC with rate
10 $R = k_i / n_c = 3/4$.

Fig. 6 shows an LDPC-CTC seed matrix portion according to the example of Fig. 5.

Fig. 7 illustrates a block-based decoding procedure.

Fig. 8 illustrates a slot-based decoding procedure.

Fig. 9 illustrates a “sliding-window” decoding procedure which may be performed by an LDPC-
15 CTC decoder of the digital communication system of Fig. 3.

Fig. 10 shows a flow chart of steps of the decoding procedure.

DETAILED DESCRIPTION

The following provides a non-limiting example of an encoding/decoding procedure for LDPC-CTC codes with is described with reference to Fig. 3 to Fig. 12.

To provide the stage for the following disclosure and introduce a notation coherent with the notation used in the prior art, the “sum-product” algorithm (SPA) proposed by D. J. C. MacKay, “*Good error-correcting codes based on very sparse matrices*,” IEEE TRANSACTIONS ON INFORMATION THEORY, vol. 45, No. 2, pp. 399-431, Mar. 1999, and its low-complexity version “normalized min sum” algorithm (NMS) is described in the following with reference to Fig. 1 and Fig. 2.

In the notation introduced by MacKay (which is based on Tanner graphs), q_{ij} denotes the message of the i -th bit node to the j -th check node and r_{ji} denotes the message of the j -th check node to the i -th bit node, wherein the number of the check nodes is equal to the number r of the rows of parity matrix H , while the number of the bit nodes is equals to the number of the columns n .

In the logarithmic domain, the quantities of interest may be the following probability ratios:

- Reliability information/values derived from the channel output:

$$Lc_i = \log \frac{\Pr(b_i = 0 | s_i)}{\Pr(b_i = 1 | s_i)} \quad (1)$$

- Check node value:

$$Lr_{ji} = \log \frac{r_{ij}(0)}{r_{ij}(1)} \quad (2)$$

- Bit node value:

$$Lq_{ij} = \log \frac{q_{ij}(0)}{q_{ij}(1)} = \text{sgn}(Lq_{ij}) \cdot |Lq_{ij}| = \alpha_{ij} \cdot \beta_{ij} \quad (3)$$

With $R_j = \{j : H(i, j) = 1\}$, i.e., the set of the indexes of the columns of the parity check matrix H that, in the row i , have the value ‘1’, and with $C_i = \{i : H(i, j) = 1\}$ indicating the set of the indexes of the rows that, in the column j , have the value ‘1’, the SPA algorithm may be summarized as in the followings lines:

$$1. Lq_{ij} = Lc_i = LLR_i / \sigma_i^2, \quad \forall i, j : H(i, j) = 1$$

$$2. Lr_{ji} = \left(\prod_{i' \in R_{ji}} \text{sgn}(Lq_{i'j}) \right) \cdot \phi \left(\sum_{j' \in R_{ji}} \phi(|Lq_{i'j}|) \right)$$

$$3. Lq_{ij} = Lc_i + \sum_{j' \in C_{ij}} Lr_{ji}$$

$$4. LQ_i = Lc_i + \sum_{j \in C} Lr_{ji}$$

$$5. \hat{c}_i = \begin{cases} 1, & LQ_i < 0 \\ 0, & LQ_i > 0 \end{cases}$$

- 5
- i. if ($\hat{c}_i H = 0$ or iteration=MAX) $\rightarrow$ STOP
 - ii. else $\rightarrow$ go to step 1

with ϕ denoting $\phi(x) = -\log \tanh(x/2) = \log \frac{e^x + 1}{e^x - 1}$ and $\hat{c}_i H = 0$ being the parity check equation of the code. If a decoded word does not satisfy the parity check equation, it can be assumed that the decoded word contains errors. With

- 10
- wr as the check node degree, which is equivalent to the weight of the rows of the matrix H , i.e., the number of '1s' in the rows of H , and
 - wc as the bit node degree, which is equivalent to the weight of the columns of the matrix H , i.e., the number of '1s' in the column of H ,

the SPA requires the following memory structures:

- 15
- LQ of dimension $1 \times n$ (soft bit word);
 - Lq of dimension $wc \times n$ (soft bit word); and
 - Lr of dimension $r \times wr$ (soft bit word).

The soft bit words are words of nb bits and correspond to the likelihood messages, i.e., the probability ratios and may, for example, be up to 8 bits wide.

There may be three different types of scheduling (wherein each type of scheduling defines an order in which the messages memorized in the memories are processed): flooding, horizontal and vertical.

Flooding scheduling updates all bit nodes first and then proceeds to update the check nodes.

- 5 The phases, for each iteration, are shown in the following pseudo-code that comprises two phases:

Phase 1. for every row i:

- a) updating the probability equation related to the check node (reading from the memory Lq)

10

$$Lr_i = \prod_{n \in R_i} \text{sgn}(Lq_{in}) \times \phi \left(\sum_{n \in R_i} \phi(|Lq_{in}|) \right)$$

- b) marginalization of the probability equation related to the check node (writing on memory Lr)

$$\forall j, Lr_{ij} = \text{sgn}(Lr_i) \times \text{sgn}(Lq_{ij}) \times \phi[\phi(|Lr_i|) - \phi(|Lq_{ij}|)]$$

Phase 2. for every column j:

15

- a) updating the a-posteriori messages (updating the memory structure storing LQ)

$$Lq_j = \sum_{m \in R_j} Lr_{mj} + Lc_j$$

- b) updating the probability related to the bit node (updating the memory structure storing Lq)

20

$$\forall m, Lq_{mj} = Lq_j - Lr_{mj}$$

The above equations are equivalent to the SPA original ones, but make the implicit calculation of the extrinsic information in the previous formulation of $R_{j \setminus i}$ ($R_{j \setminus i}$ indicating the set of indexes where the message related to element that is under calculation is not considered)

explicit. This is an important aspect of the used decoding algorithm and is usually called marginalization. Also important is that this way of calculating the extrinsic information requires less operations.

The complexity of the SPA can be reduced by using the following approximation:

$$5 \quad \phi \left(\sum_{i \in R_{jN}} \phi(|Lq_{ji}|) \right) \cong \phi \left(\phi \left(\min_i (|Lq_{ji}|) \right) \right) = \min_i (|Lq_{ji}|), \text{ with } \phi(\phi(x)) = x$$

I.e., as the result is dominated by the smallest value in the sum, the result can be approximated by calculating two absolute minimums between the modulus of the Lq messages. The Lq messages are those related to the parity equation of the row of the parity check matrix H that is processed. The necessity to calculate two minimums (instead of only one) comes from the
 10 fact that the information is always extrinsic. If, for example, the sequence of Lq messages to process is $\{4.0, 5.1, 5.2, 3.1, 2.1, 3.3, 1.4, 6.0, 6.0\}$, where the absolute minimum is in the seventh position, the output sequence will be $\{1.4, 1.4, 1.4, 1.4, 1.4, 1.4, 2.1, 1.4, 1.4\}$. The value in the seventh position is now the second absolute minimum. This simplification reduces performance to an extent that can be (sometimes completely) recovered by multiplying the
 15 output sequence by a scaling factor γ of about 0.7.

In this regard, it is important to note that it is not needed to know the variance of the Gaussian noise related to each LLR coming from the demapper.

Another important choice is the scheduling type. Reasons for using horizontal scheduling are:

- the memory saving, since the Lq memory is not required;
- a faster convergence, as for a given performance, the iterations are halved; and
- that it is suitable for a parallel architecture.

In this case, the decoding procedure has only one phase which combines the two phases (Phase 1 and Phase 2) for calculating the values Lq . As described by M. Mansour and N. Shambhag,

“Memory-efficient turbo decoder architectures for LDPC codes”, IEEE WORKSHOP ON SIGNAL PROCESSING SYSTEMS, pp. 159-164, Oct.2002, the “two phase message passing” (TPMP) can be replaced by “turbo decoding message passing” (TDMP). In brief, the TDMP is characterized by:

- 5 a) the node bit phase is comprised in the check node phase:

$$Lr_{ji} = \left[\prod_{j'} \text{sgn}(LQ_{ji'} - Lr_{ji'}) \right] \times \left[\min_{j'} (|LQ_{ji'} - Lr_{ji'}|) \right]$$

- b) during the processing of the current check-node all the connected bit-nodes are updated; and
 c) the propagation of “always up-to-date” messages speeds up the convergence
 10 factor by two.

For example, looking at the matrix in Fig. 1, the processing of the third parity equation ($cn\ 2$) requires four bit-node messages where 3 of the 4 has been already updated in the processing of the two previous equations (i.e., $q0$ and $q7$ in the updating of row 1, $q6$ in the updating of row 2). With I_i denoting the subset of the column index R_i , the pseudo-code then becomes:

```

15 for each iteration
    for each row  $i$  of  $H$ 
        read  $Lr_i$  and  $LQ(i)$ 
         $Lq_i \leftarrow LQ(I_i) - Lr_i$ 
         $LR_i \leftarrow NMS(Lq_i)$ 
         $Lr_i \leftarrow LR_i$ 
         $LQ(I_i) \leftarrow Lq_i + LR_i$ 
    end
20 end

```

The terms LQ and Lr are vectors that are read from the memory structures LQ and Lr . If, for example, the weight of the i -th row is 22, LQ and Lr each comprises 22 elements (wherein each element is 8 bits wide), $LQ(I_i)$ indicates that 22 (not adjacent) values have to be selected from the memory LQ based on I_i as index (or address), which can be derived from
 25 row i of the parity check matrix H . With these 22 values, the Lq values are calculated which,

after being processed by a NMS function, give the new values to update the memory Lr . Adding the previously obtained Lq to these Lr , the locations of LQ indicated by I_i can be updated. The function that calculates the $Lr = NMS(Lq)$ is

$$Lr_{ij} = \gamma \cdot \prod_{n \in R_{ij}} \text{sgn}(Lq_{in}) \times \min_{n \in R_{ij}} (|Lq_{in}|).$$

- 5 For efficiency reason, and for the architecture chosen, the multiplication of all the signs may be calculated and finally multiplied to extract the extrinsic information (marginalization). This is the analogy to the calculus of the extrinsic information using two minimums. At the end, the modulus is scaled by the factor gamma (γ). To give an overview, the functional blocks of a calculus or “decoding unit” (DU) for TDMP based on the NMS algorithm are depicted in Fig.
- 10 2.

Fig. 3 shows a block diagram of a digital communication system in which the decoding procedure of the present invention may be implemented (e.g., at the decoder CH_DEC of the system). As shown in Fig. 3, a binary word A1 (information sequence) of k_i bits may be encoded by a channel encoder CH_ENC that adds $r = n_c - k_i$ parity bits (redundancy

15 sequence) to the information sequence to obtain a binary word A2 (encoded sequence) of n_c bits. The channel code-rate R is defined as the ratio between the number of information bits k_i and the number of encoded bits n_c , thus $R = k_i / n_c$.

The modulator MOD may transform the encoded vector A2 into a modulated signal vector CH_IN which is in turn transmitted through a channel CH which may, for example, correspond

20 to a wired and/or wireless transmission path. The encoder CH_ENC and the modulator MOD may be comprised in an electronic device that performs the encoding/modulation procedure using customized hardware and/or a processor executing instructions stored in a machine-readable medium. Since the channel CH is usually subject to noisy disturbance NS, the channel output CH_OUT may differ from the channel input CH_IN.

- 25 On the receiver side, the channel output CH_OUT may be processed by the demodulator DEM which may perform the MOD inverse operation (demodulation) and produce some likelihood ratio (soft bits). The channel decoder CH_DEC may use the redundancy in the received

sequence A3 to correct errors in the information part of the received sequence A3 and produce a decoded signal A4 which is an information sequence estimate.

The encoder/decoder CH_ENC, CH_DEC may perform the encoding/decoding procedure based on a convolutional code with a LDPC matrix as proposed by A. J. Felstrom and K. Zigangirov (see above). As a consequence, the n_c encoded bits output by the encoder CH_ENC depend not only on the k_i input bits received at the current iteration, but also on M previous input blocks (of k_i input bits each).

The transposed parity-check matrix, called syndrome matrix, of a periodical binary convolutional code with memory M , code-rate $R = k_i/n_c$, code-period $T = M - 1$ and code-period number p is:

$$H^T = \begin{bmatrix} H_0^{(0)} & H_1^{(1)} & H_2^{(2)} & H_3^{(3)} & \cdot & H_M^{(M)} & 0 & 0 & 0 & 0 & \cdot & 0 \\ 0 & H_1^{(0)} & H_2^{(1)} & H_3^{(2)} & \cdot & H_M^{(M-1)} & H_{M+1}^{(M)} & 0 & 0 & 0 & \cdot & 0 \\ 0 & 0 & H_2^{(0)} & H_3^{(1)} & \cdot & H_M^{(M-2)} & H_{M+1}^{(M-1)} & H_{M+2}^{(M)} & 0 & 0 & \cdot & 0 \\ 0 & 0 & 0 & H_3^{(0)} & \cdot & H_M^{(M-3)} & H_{M+1}^{(M-2)} & H_{M+2}^{(M-1)} & H_{M+3}^{(M)} & 0 & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 0 & \cdot & 0 & H_{[(M-1)p-1]}^{(0)} & H_{[(M-1)p]}^{(1)} & H_{[(M-1)p+1]}^{(2)} & \cdot & \cdot & H_{[M+2T-1]}^{(M)} \end{bmatrix} \quad (4)$$

where each element $H_t^{(m)}$ with $m = 0, 1, 2, \dots, M$ and $t = 0, 1, 2, \dots$ is a binary sub-matrix of the size $n_c \times (n_c - k_i)$.

For example, for a code-rate $R = k_i/n_c = 2/3$, a code-period $T = M - 1 = 7 - 1 = 6$ and a code-period number $p = 2$, each element of the transposed parity-check matrix $H_t^{(m)}$ is a parity matrix of size $n_c \times (n_c - k_i) = 3 \times 1$. Then, the syndrome matrix H^T of equation (4) becomes:

$$H^T = \begin{bmatrix} h_1^{(0)}(0) & h_1^{(1)}(1) & h_1^{(2)}(2) & . & h_1^{(7)}(7) & 0 & 0 & . & 0 & 0 & . & 0 \\ h_2^{(0)}(0) & h_2^{(1)}(1) & h_2^{(2)}(2) & . & h_2^{(7)}(7) & 0 & 0 & . & 0 & 0 & . & 0 \\ h_3^{(0)}(0) & h_3^{(1)}(1) & h_3^{(2)}(2) & . & h_3^{(7)}(7) & 0 & 0 & . & 0 & 0 & . & 0 \\ 0 & h_1^{(0)}(1) & h_1^{(1)}(2) & . & h_1^{(6)}(7) & h_1^{(7)}(8) & 0 & . & 0 & 0 & . & 0 \\ 0 & h_2^{(0)}(1) & h_2^{(1)}(2) & . & h_2^{(6)}(7) & h_2^{(7)}(8) & 0 & . & 0 & 0 & . & 0 \\ 0 & h_3^{(0)}(1) & h_3^{(1)}(2) & . & h_3^{(6)}(7) & h_3^{(7)}(8) & 0 & . & 0 & 0 & . & 0 \\ 0 & 0 & h_1^{(0)}(2) & . & h_1^{(5)}(7) & h_1^{(6)}(8) & h_1^{(7)}(9) & . & 0 & 0 & . & 0 \\ 0 & 0 & h_2^{(0)}(2) & . & h_2^{(5)}(7) & h_2^{(6)}(8) & h_2^{(7)}(9) & . & 0 & 0 & . & 0 \\ 0 & 0 & h_3^{(0)}(2) & . & h_3^{(5)}(7) & h_3^{(6)}(8) & h_3^{(7)}(9) & . & 0 & 0 & . & 0 \\ . & . & . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . & . & . \\ 0 & 0 & 0 & . & 0 & 0 & 0 & . & h_1^{(0)}(11) & h_1^{(1)}(12) & . & h_1^{(7)}(18) \\ 0 & 0 & 0 & . & 0 & 0 & 0 & . & h_2^{(0)}(11) & h_2^{(1)}(12) & . & h_2^{(7)}(18) \\ 0 & 0 & 0 & . & 0 & 0 & 0 & . & h_3^{(0)}(11) & h_3^{(1)}(12) & . & h_3^{(7)}(18) \end{bmatrix} \quad (5)$$

The low-density parity-check matrix may be characterized by the following parameters:

- M determines the code period $T = M - 1$;
- Ju is the row weight (number of elements equal to '1' in the subrow) corresponding to the information sequence;
- Jv is the row weight (number of elements equal to '1' in the subrow) corresponding to the parity bits; and
- K is the column weight (number of elements equal to '1' in the column) starting from column $[M \cdot (n_c - k_i) + 1] - th$.

The information sequence $u_{[0,n]}$ and the encoded sequence $v_{[0,n]}$ can be expressed by equation (6) and equation (7) as:

$$u_{[0,n]} = [u_{01}, u_{02}, u_{11}, u_{12}, u_{21}, u_{22}, u_{31}, u_{32}, \dots, u_{n1}, u_{n2}] \quad (6)$$

$$v_{[0,n]} = [v_{01}, v_{02}, v_{11}, v_{12}, v_{21}, v_{22}, v_{31}, v_{32}, \dots, v_{n1}, v_{n2}] \quad (7)$$

The received sequence will be correct, if equation (8) is satisfied:

$$\hat{v}_{[0,n]} \cdot H^T = 0 \quad (8)$$

For all information bits to be protected by corresponding parity bits, it is necessary to impose the following condition:

$$5 \quad \begin{cases} h_1^{(0)}(n) = 1 \\ h_2^{(0)}(n) = 1 \\ h_3^{(0)}(n) = 1 \end{cases} \quad (9)$$

where $n = 0, 1, 2, \dots$

A systematic LDPC-CC encoder may carry-out a procedure corresponding to the following equations:

$$\begin{cases} v_{n1} = u_{n1}, v_{n2} = u_{n2} \\ v_{n3} = \sum_{i=0}^M h_1^{(i)}(n) \cdot u_{(n-i)1} + \sum_{i=0}^M h_2^{(i)}(n) \cdot u_{(n-i)2} + \sum_{i=1}^M h_3^{(i)}(n) \cdot u_{(n-i)3} \end{cases} \quad (10)$$

10 As shown by T. Kishigami, Y. Murakami, and I. Yoshii, “*LDPC Convolutional Codes for IEEE 802.16m FEC Scheme*”, IEEE 802.16 Broadband Wireless Access Working Group, LDPC-CCs have advantages in terms of encoder complexity and decoder latency as compared to conventional FEC classes such as Convolutional Turbo Codes (CTC) and LDPC Block Codes (LDPC-BC).

15 A binary parity-check matrix built as suggested by A. J. Felstrom and K. Zigangirov (see above) for code-rates $R_1 = k_{i1}/n_{c1}$, code-period $T = 5$, and code-period number $p = 3$, may be composed of three code periods, wherein the first two code periods are complete and the last code period is incomplete. The first code period and the second code period can be repeated

several times (to allow for variable code-word lengths) due to which the parity-check matrix may be called semi-infinite.

5 In order to improve the LDPC-CC, it can be feasible to apply a “tail-biting” operation that masks the parity-check matrix columns with lower weight which correspond to the last code period. In this last code period, information bits are encoded equal to ‘0’ (which are not transmitted), to generate the corresponding parity bits, which are transmitted.

The binary parity-check matrix construction for LDPC-CC as suggested by A. J. Felstrom and K. Zigangirov (see above):

- has no column weight equal to ‘1’; and
- 10 • the row layered decoding, already used in LDPC-BC, cannot be used.

Thus:

- a serial layered decoding, which decodes the same information sequence, achieves good performance (fast convergence of the belief propagation algorithm) but leads to high latency;
- 15 • a parallel layered decoding, which decodes the same information sequence, leads to weak performance (slow convergence of the belief propagation algorithm) but achieves low latency; and
- a parallel layered decoding, which decodes a different information sequence, achieves good performance (fast convergence of the belief propagation algorithm) but leads to
- 20 high latency.

Besides, the binary parity-check matrix construction for LDPC-CC as suggested by A. J. Felstrom and K. Zigangirov (see above) may suffer from cycles of length 4 which lead to performance degradation of the LDPC-CC. The proposed encoding/decoding procedure for LDPC-CTC codes overcomes the aforesaid drawbacks as it constructs the seed matrix for

25 LDPC-CTC by interlacing (two or more) binary parity-check matrixes of LDPC-CCs, wherein

each LDPC-CC may be based on the construction procedure proposed by A. J. Felstrom and K. Zigangirov (see above), with (or without) concatenated parity part.

A seed matrix for a specific rate R masked with a full exponent matrix E_{exp} for the rate R and expanded with a spreading matrix (cyclic permutation matrix) may produce the binary parity-check matrix H_{rxn} for the LDPC-CTC which leads to significant performance gains and allows carrying-out parallel layered decoding with good performance (fast convergence of the BPA) and low decoding latency. For example, the seed matrix may be of the form:

r/c	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	1	1	0	0
2	1	1	1	1	1	1	1	1	0
3	0	0	0	1	1	1	0	0	1
4	1	1	0	0	0	1	0	0	1
5	0	1	1	0	1	0	0	0	1
6	0	0	1	1	0	1	0	0	1
7	1	0	1	0	0	0	0	0	0
8	0	1	1	0	1	0	0	0	0
9	0	1	0	1	0	0	0	0	0
10	1	0	0	0	1	0	1	0	1
11	1	0	0	0	0	1	0	0	1
12	0	0	0	1	0	0	1	0	0
13	0	0	0	0	0	0	1	0	0
14	0	0	0	0	0	0	0	1	0
15	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0

29	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0

r/c	10	11	12	13	14	15	16	17	18
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	1	0	0	0
4	1	1	1	1	1	1	1	0	0
5	0	0	0	0	0	0	0	1	1
6	0	0	1	0	0	1	0	1	1
7	0	1	1	0	0	0	0	0	1
8	0	1	0	1	0	0	0	0	1
9	1	1	0	1	0	0	0	0	0
10	0	0	0	1	1	0	0	1	0
11	1	0	0	0	1	0	0	0	1
12	1	1	1	0	0	0	0	0	0
13	0	0	1	1	1	0	0	1	0
14	1	0	0	0	1	1	0	0	0
15	0	0	0	0	0	1	0	1	0
16	0	0	0	0	0	0	1	1	1
17	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0

r/c	19	20	21	22	23	24	25	26	27
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	1	1	1	1	1	0	0	0	0
6	1	1	1	1	1	1	0	0	0
7	0	1	1	0	0	0	1	1	1
8	0	1	1	0	0	0	1	1	1
9	1	0	0	0	0	0	1	1	0
10	0	0	0	1	1	0	1	0	0
11	0	0	0	1	0	0	1	0	1
12	1	0	1	1	0	0	0	1	0
13	0	0	1	0	0	0	0	0	1
14	1	1	0	0	1	0	0	0	1
15	1	1	0	1	0	0	0	0	0
16	0	0	0	0	0	0	1	1	1
17	0	0	0	0	1	0	0	1	0
18	0	0	0	0	0	1	0	0	0
19	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0

r/c	28	29	30	31	32	33	34	35	36
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0

7	1	1	1	1	0	0	0	0	0
8	1	1	1	1	1	0	0	0	0
9	1	1	0	0	0	1	1	1	1
10	0	1	0	0	0	1	1	1	1
11	0	0	0	0	0	1	0	1	0
12	0	1	1	1	0	0	0	0	1
13	0	1	0	0	0	0	1	0	0
14	1	0	0	1	0	0	0	1	0
15	1	0	1	0	0	1	1	0	0
16	0	0	0	0	0	0	1	0	0
17	0	0	1	0	0	0	0	0	1
18	1	0	1	0	0	1	1	1	0
19	0	0	0	1	0	0	0	1	1
20	0	0	0	0	1	1	0	0	1
21	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0

r/c	37	38	39	40	41	42	43	44	45
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	1	1	1	0	0	0	0	0	0
10	1	1	1	1	0	0	0	0	0
11	0	1	0	0	1	1	1	1	1
12	1	1	0	0	1	1	1	1	1
13	1	0	0	0	0	0	0	1	1

14	0	1	1	0	1	1	0	0	0
15	0	0	0	0	0	1	1	0	1
16	0	0	1	0	0	0	1	1	0
17	1	1	0	0	1	0	1	0	0
18	1	0	0	0	0	1	1	0	1
19	0	0	0	0	0	1	0	1	0
20	0	0	0	0	1	0	0	0	1
21	0	0	1	0	1	0	0	0	0
22	0	0	0	1	0	0	0	1	0
23	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0

r/c	46	47	48	49	50	51	52	53	54
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0
11	1	1	0	0	0	0	0	0	0
12	1	1	1	0	0	0	0	0	0
13	1	0	0	1	1	1	1	1	1
14	1	0	0	1	1	1	1	1	1
15	0	0	0	1	0	0	0	0	0
16	1	0	0	1	0	0	1	0	0
17	0	0	0	0	0	1	1	0	0
18	0	0	0	0	0	1	0	1	0
19	0	0	0	0	1	1	0	1	0
20	0	1	0	1	0	0	0	1	1

21	1	0	0	1	1	0	0	0	1
22	0	1	0	0	1	1	1	0	0
23	0	1	0	0	0	0	1	1	1
24	0	0	1	0	1	0	0	0	1
25	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0

r/c	55	56	57	58	59	60	61	62	63
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0
13	1	0	0	0	0	0	0	0	0
14	1	1	0	0	0	0	0	0	0
15	0	0	1	1	1	1	1	1	1
16	1	0	1	1	1	1	1	1	1
17	0	0	0	1	0	1	1	0	0
18	0	0	0	1	0	1	1	0	0
19	0	0	0	0	1	0	0	0	0
20	0	0	1	0	0	0	0	1	1
21	0	0	0	1	0	0	0	1	0
22	0	0	0	0	1	0	1	1	0
23	0	0	1	0	0	0	1	0	0
24	1	0	0	0	1	1	0	0	1
25	1	0	1	0	1	1	0	1	0
26	0	1	1	1	0	0	0	0	0
27	0	0	0	0	0	0	0	0	1

28	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0

r/c	64	65	66	67	68	69	70	71	72
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0
16	1	0	0	0	0	0	0	0	0
17	0	1	1	1	1	1	1	1	0
18	0	1	1	1	1	1	1	1	1
19	0	1	1	0	1	1	0	0	0
20	0	1	0	0	0	1	0	0	0
21	0	1	0	1	0	0	0	0	0
22	0	0	1	0	0	1	1	1	0
23	0	0	0	1	0	1	0	0	0
24	0	0	0	1	1	0	0	1	0
25	0	0	0	0	1	0	1	0	0
26	0	1	1	1	0	0	0	0	0
27	0	0	1	0	0	0	1	0	0
28	1	0	0	0	1	0	1	0	0
29	0	0	0	0	0	0	0	1	0
30	0	0	0	0	0	0	0	0	1
31	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0

r/c	73	74	75	76	77	78	79	80	81
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0
19	1	1	1	1	1	1	1	0	0
20	1	1	1	1	1	1	1	1	0
21	1	0	1	0	0	1	0	0	1
22	0	0	0	1	1	1	0	0	1
23	0	1	0	0	1	0	0	0	0
24	0	0	1	0	0	1	1	0	1
25	1	1	0	0	0	0	0	0	0
26	0	1	0	0	0	0	1	0	0
27	0	0	0	1	1	1	0	0	1
28	1	1	1	0	1	0	0	0	0
29	0	0	1	1	0	0	0	0	0
30	1	0	0	1	0	0	0	0	1
31	0	0	0	0	0	0	1	0	1
32	0	0	0	0	0	0	0	1	0

r/c	82	83	84	85	86	87	88	89	90
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0

7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0
21	1	1	1	1	1	1	0	0	0
22	1	1	1	1	1	1	1	0	0
23	0	0	1	1	1	0	0	1	1
24	1	0	0	0	1	0	0	1	1
25	1	1	0	1	0	0	0	1	0
26	0	1	1	0	1	0	0	1	0
27	0	1	0	0	0	0	0	0	0
28	1	1	0	1	0	0	0	0	0
29	1	0	1	0	0	0	0	0	1
30	0	0	0	1	0	1	0	1	0
31	0	0	0	0	1	0	0	1	1
32	0	0	1	0	0	1	0	0	1

r/c	91	92	93	94	95	96	97	98	99
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0

14	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0
23	1	1	1	1	1	0	0	0	0
24	1	1	1	1	1	1	0	0	0
25	0	0	0	0	0	0	1	1	1
26	0	1	0	0	1	0	1	1	1
27	1	1	0	0	0	0	0	1	0
28	1	0	1	0	0	0	0	1	0
29	1	0	1	0	0	0	0	0	1
30	0	0	1	1	0	0	1	0	0
31	0	0	0	1	0	0	0	1	0
32	1	1	0	0	0	0	0	0	1

r/c	100	101	102	103	104	105	106	107	108
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0

21	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0
25	1	1	1	1	0	0	0	0	0
26	1	1	1	1	1	0	0	0	0
27	1	1	0	0	0	1	1	1	1
28	1	1	0	0	0	1	1	1	1
29	0	0	0	0	0	1	1	0	1
30	0	0	1	1	0	1	0	0	0
31	0	0	1	0	0	1	0	1	0
32	0	1	1	0	0	0	1	0	0

r/c	109	110	111	112	113	114	115	116	117
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0
27	1	1	1	0	0	0	0	0	0

28	1	1	1	1	0	0	0	0	0
29	1	0	0	0	1	1	1	1	1
30	1	0	0	0	1	1	1	1	1
31	0	0	0	0	1	0	1	0	0
32	1	1	1	0	0	0	0	1	1

r/c	118	119	120	121	122	123	124	125	126
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0
29	1	1	0	0	0	0	0	0	0
30	1	1	1	0	0	0	0	0	0
31	1	0	0	1	1	1	1	1	1
32	1	0	0	1	1	1	1	1	1

r/c	127	128
1	0	0
2	0	0
3	0	0
4	0	0
5	0	0
6	0	0
7	0	0
8	0	0
9	0	0
10	0	0
11	0	0
12	0	0
13	0	0
14	0	0
15	0	0
16	0	0
17	0	0
18	0	0
19	0	0
20	0	0
21	0	0
22	0	0
23	0	0
24	0	0
25	0	0
26	0	0
27	0	0
28	0	0
29	0	0
30	0	0
31	1	0
32	1	1

Table 1

In this regard, it is noted that the LDPC-CTC does not require an interleaver circuit, because this operation can be performed by spreading matrixes with different exponent value for each parallel path (two or more). Moreover, there exists a tradeoff between the desired performance and the number of parallel concatenated LDPC-CCs (hardware complexity).

The spreading matrix is a square binary matrix (circulant permutation matrix) of weight equal to '1' and size $Z_f \times Z_f$ being Z_f the spreading factor which is tied to the code-word length of the carried-out LDPC code. For example, if the spreading factor is $Z_f = 48$ then the full exponent matrix E_{exp} will have elements (exponents) between 0 and $(Z_f - 1) = 47$.

- 5 The exponent matrix elements with values between 0 and $(Z_f - 1)$ may thus represent cyclic permutation matrixes of the identity matrix (with weight equal to '1' for each row and for each column) with size $Z_f \times Z_f$. The value of the exponent matrix element may indicate the cyclic shift to the right of the identity matrix, as shown in the matrixes $I_{Z_f}^{[0]}, I_{Z_f}^{[1]}, \dots, I_{Z_f}^{[Z_f-1]}$ below. All non-negative exponent matrix elements may correspond to circulant permutation matrices, for example the zero exponent permutation matrix may correspond to the identity matrix:
- 10

$$I_{Z_f}^{[0]} = \begin{bmatrix} 1 & 0 & 0 & 0 & . & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & . & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & . & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & . & 0 & 0 & 0 \\ . & . & . & . & . & . & . & . \\ 0 & 0 & 0 & 0 & . & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & . & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & . & 0 & 0 & 1 \end{bmatrix}$$

$$I_{Z_f}^{[1]} = \begin{bmatrix} 0 & 1 & 0 & 0 & . & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & . & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & . & 0 & 0 & 0 \\ . & . & . & . & . & . & . & . \\ 0 & 0 & 0 & 0 & . & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & . & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & . & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & . & 0 & 0 & 0 \end{bmatrix}$$

$$I_{Z_f}^{[Z_f-1]} = \begin{bmatrix} 0 & 0 & 0 & 0 & . & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & . & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & . & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & . & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & . & 0 & 0 & 0 \\ . & . & . & . & . & . & . & . \\ 0 & 0 & 0 & 0 & . & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & . & 0 & 1 & 0 \end{bmatrix}$$

The same benefits may be obtained if the cyclic shift is a cyclic shift to the left instead of to the right.

Each element of the exponent matrix E_{exp} of value equal to '-1' may represent a null square matrix of size $Z_f \times Z_f$ as shown in matrix $I_{Z_f}^{[-1]}$:

$$I_{Z_f}^{[-1]} = \begin{bmatrix} 0 & 0 & 0 & 0 & . & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & . & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & . & 0 & 0 & 0 \\ . & . & . & . & . & . & . & . \\ 0 & 0 & 0 & 0 & . & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & . & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & . & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & . & 0 & 0 & 0 \end{bmatrix}$$

For example, the exponent matrix E_{exp} may be of the form:

r/c	1	2	3	4	5	6	7	8	9
1	33	6	15	36	30	24	0	-1	-1
2	36	22	40	42	17	5	13	0	-1
3	39	38	12	0	4	39	26	33	35
4	42	1	37	1	44	20	39	13	40
5	45	17	9	7	31	1	4	46	45
6	0	33	34	13	18	35	12	26	2
7	3	1	6	19	5	16	25	6	2
8	1	12	31	25	45	2	38	39	7
9	4	28	3	31	32	31	3	19	12
10	7	44	28	37	19	12	11	4	17
11	10	7	0	43	6	46	24	32	22
12	13	23	25	1	46	27	37	12	27
13	16	39	2	2	33	8	2	45	32
14	19	2	22	8	20	42	10	25	37
15	-1	-1	-1	-1	-1	-1	-1	-1	42
16	-1	-1	-1	-1	-1	-1	-1	-1	47
17	-1	-1	-1	-1	-1	-1	-1	-1	-1
18	-1	-1	-1	-1	-1	-1	-1	-1	-1
19	-1	-1	-1	-1	-1	-1	-1	-1	-1
20	-1	-1	-1	-1	-1	-1	-1	-1	-1
21	-1	-1	-1	-1	-1	-1	-1	-1	-1
22	-1	-1	-1	-1	-1	-1	-1	-1	-1

23	-1	-1	-1	-1	-1	-1	-1	-1	-1
24	-1	-1	-1	-1	-1	-1	-1	-1	-1
25	-1	-1	-1	-1	-1	-1	-1	-1	-1
26	-1	-1	-1	-1	-1	-1	-1	-1	-1
27	-1	-1	-1	-1	-1	-1	-1	-1	-1
28	-1	-1	-1	-1	-1	-1	-1	-1	-1
29	-1	-1	-1	-1	-1	-1	-1	-1	-1
30	-1	-1	-1	-1	-1	-1	-1	-1	-1
31	-1	-1	-1	-1	-1	-1	-1	-1	-1
32	-1	-1	-1	-1	-1	-1	-1	-1	-1

r/c	10	11	12	13	14	15	16	17	18
1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1
3	20	1	25	39	9	0	-1	-1	-1
4	2	12	7	0	28	21	0	-1	-1
5	27	23	42	4	47	42	37	31	37
6	4	34	24	13	13	10	21	32	44
7	34	45	6	22	32	31	5	33	3
8	11	3	41	31	3	4	42	34	5
9	41	14	23	40	17	20	26	35	12
10	18	25	5	1	36	41	10	36	19
11	0	36	40	5	2	9	47	37	26
12	25	47	22	14	21	30	31	38	33
13	2	5	4	23	40	3	15	39	40
14	32	16	39	32	6	19	4	40	47
15	9	27	21	41	25	40	36	41	1
16	39	38	3	2	44	8	20	42	8
17	-1	-1	-1	-1	-1	-1	-1	43	15
18	-1	-1	-1	-1	-1	-1	-1	44	22
19	-1	-1	-1	-1	-1	-1	-1	-1	-1
20	-1	-1	-1	-1	-1	-1	-1	-1	-1
21	-1	-1	-1	-1	-1	-1	-1	-1	-1
22	-1	-1	-1	-1	-1	-1	-1	-1	-1
23	-1	-1	-1	-1	-1	-1	-1	-1	-1
24	-1	-1	-1	-1	-1	-1	-1	-1	-1
25	-1	-1	-1	-1	-1	-1	-1	-1	-1
26	-1	-1	-1	-1	-1	-1	-1	-1	-1
27	-1	-1	-1	-1	-1	-1	-1	-1	-1
28	-1	-1	-1	-1	-1	-1	-1	-1	-1
29	-1	-1	-1	-1	-1	-1	-1	-1	-1

30	-1	-1	-1	-1	-1	-1	-1	-1	-1
31	-1	-1	-1	-1	-1	-1	-1	-1	-1
32	-1	-1	-1	-1	-1	-1	-1	-1	-1

r/c	19	20	21	22	23	24	25	26	27
1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	2	13	21	5	0	-1	-1	-1	-1
6	14	36	4	20	20	0	-1	-1	-1
7	26	6	30	35	40	26	40	18	8
8	38	29	8	2	7	4	2	46	26
9	2	4	39	12	27	25	7	21	44
10	9	22	17	27	47	3	17	1	9
11	21	45	0	42	14	24	27	24	27
12	33	15	26	4	34	2	37	4	45
13	45	38	4	19	1	23	47	27	10
14	4	8	35	34	21	1	4	2	28
15	16	31	13	1	41	22	14	30	46
16	28	1	44	11	8	0	24	5	11
17	40	24	22	26	28	21	34	33	29
18	4	47	0	41	0	47	44	8	47
19	-1	-1	-1	-1	-1	-1	1	36	12
20	-1	-1	-1	-1	-1	-1	11	11	30
21	-1	-1	-1	-1	-1	-1	-1	-1	-1
22	-1	-1	-1	-1	-1	-1	-1	-1	-1
23	-1	-1	-1	-1	-1	-1	-1	-1	-1
24	-1	-1	-1	-1	-1	-1	-1	-1	-1
25	-1	-1	-1	-1	-1	-1	-1	-1	-1
26	-1	-1	-1	-1	-1	-1	-1	-1	-1
27	-1	-1	-1	-1	-1	-1	-1	-1	-1
28	-1	-1	-1	-1	-1	-1	-1	-1	-1
29	-1	-1	-1	-1	-1	-1	-1	-1	-1
30	-1	-1	-1	-1	-1	-1	-1	-1	-1
31	-1	-1	-1	-1	-1	-1	-1	-1	-1
32	-1	-1	-1	-1	-1	-1	-1	-1	-1

r/c	28	29	30	31	32	33	34	35	36
1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	-1	-1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	-1	-1
7	26	32	7	0	-1	-1	-1	-1	-1
8	9	34	24	22	0	-1	-1	-1	-1
9	45	36	41	44	38	38	14	4	22
10	28	38	5	13	23	46	38	18	1
11	11	40	22	35	8	1	9	32	33
12	47	42	39	4	46	9	33	46	12
13	30	44	3	26	31	17	4	7	44
14	13	46	20	0	16	25	28	21	23
15	1	0	37	17	1	33	4	35	2
16	32	2	1	39	39	41	23	1	34
17	15	4	18	8	24	1	47	10	13
18	3	1	35	30	9	4	18	24	45
19	34	3	4	4	47	12	42	38	24
20	17	5	16	11	32	20	13	4	3
21	-1	-1	-1	-1	-1	28	37	13	35
22	-1	-1	-1	-1	-1	36	8	27	14
23	-1	-1	-1	-1	-1	-1	-1	-1	-1
24	-1	-1	-1	-1	-1	-1	-1	-1	-1
25	-1	-1	-1	-1	-1	-1	-1	-1	-1
26	-1	-1	-1	-1	-1	-1	-1	-1	-1
27	-1	-1	-1	-1	-1	-1	-1	-1	-1
28	-1	-1	-1	-1	-1	-1	-1	-1	-1
29	-1	-1	-1	-1	-1	-1	-1	-1	-1
30	-1	-1	-1	-1	-1	-1	-1	-1	-1
31	-1	-1	-1	-1	-1	-1	-1	-1	-1
32	-1	-1	-1	-1	-1	-1	-1	-1	-1

r/c	37	38	39	40	41	42	43	44	45
1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	-1	-1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	-1	-1

7	-1	-1	-1	-1	-1	-1	-1	-1	-1
8	-1	-1	-1	-1	-1	-1	-1	-1	-1
9	17	34	0	-1	-1	-1	-1	-1	-1
10	44	38	29	0	-1	-1	-1	-1	-1
11	18	42	5	39	33	6	15	36	30
12	45	46	34	25	36	22	40	42	17
13	19	2	10	11	39	38	12	0	4
14	46	1	39	2	42	1	37	1	44
15	20	5	15	36	45	17	9	7	31
16	47	9	44	22	0	33	34	13	18
17	21	13	20	8	3	1	6	19	5
18	0	17	1	47	1	12	31	25	45
19	22	21	25	33	4	28	3	31	32
20	1	25	1	19	7	44	28	37	19
21	23	29	30	5	10	7	0	43	6
22	2	33	6	44	13	23	25	1	46
23	-1	-1	-1	-1	16	39	2	2	33
24	-1	-1	-1	-1	19	2	22	8	20
25	-1	-1	-1	-1	-1	-1	-1	-1	-1
26	-1	-1	-1	-1	-1	-1	-1	-1	-1
27	-1	-1	-1	-1	-1	-1	-1	-1	-1
28	-1	-1	-1	-1	-1	-1	-1	-1	-1
29	-1	-1	-1	-1	-1	-1	-1	-1	-1
30	-1	-1	-1	-1	-1	-1	-1	-1	-1
31	-1	-1	-1	-1	-1	-1	-1	-1	-1
32	-1	-1	-1	-1	-1	-1	-1	-1	-1

r/c	46	47	48	49	50	51	52	53	54
1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	-1	-1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	-1	-1
7	-1	-1	-1	-1	-1	-1	-1	-1	-1
8	-1	-1	-1	-1	-1	-1	-1	-1	-1
9	-1	-1	-1	-1	-1	-1	-1	-1	-1
10	-1	-1	-1	-1	-1	-1	-1	-1	-1
11	24	0	-1	-1	-1	-1	-1	-1	-1
12	5	13	0	-1	-1	-1	-1	-1	-1
13	39	26	33	35	20	1	25	39	9

14	20	39	13	40	2	12	7	0	28
15	1	4	46	45	27	23	42	4	47
16	35	12	26	2	4	34	24	13	13
17	16	25	6	2	34	45	6	22	32
18	2	38	39	7	11	3	41	31	3
19	31	3	19	12	41	14	23	40	17
20	12	11	4	17	18	25	5	1	36
21	46	24	32	22	0	36	40	5	2
22	27	37	12	27	25	47	22	14	21
23	8	2	45	32	2	5	4	23	40
24	42	10	25	37	32	16	39	32	6
25	-1	-1	-1	42	9	27	21	41	25
26	-1	-1	-1	47	39	38	3	2	44
27	-1	-1	-1	-1	-1	-1	-1	-1	-1
28	-1	-1	-1	-1	-1	-1	-1	-1	-1
29	-1	-1	-1	-1	-1	-1	-1	-1	-1
30	-1	-1	-1	-1	-1	-1	-1	-1	-1
31	-1	-1	-1	-1	-1	-1	-1	-1	-1
32	-1	-1	-1	-1	-1	-1	-1	-1	-1

r/c	55	56	57	58	59	60	61	62	63
1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	-1	-1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	-1	-1
7	-1	-1	-1	-1	-1	-1	-1	-1	-1
8	-1	-1	-1	-1	-1	-1	-1	-1	-1
9	-1	-1	-1	-1	-1	-1	-1	-1	-1
10	-1	-1	-1	-1	-1	-1	-1	-1	-1
11	-1	-1	-1	-1	-1	-1	-1	-1	-1
12	-1	-1	-1	-1	-1	-1	-1	-1	-1
13	0	-1	-1	-1	-1	-1	-1	-1	-1
14	21	0	-1	-1	-1	-1	-1	-1	-1
15	42	37	31	37	2	13	21	5	0
16	10	21	32	44	14	36	4	20	20
17	31	5	33	3	26	6	30	35	40
18	4	42	34	5	38	29	8	2	7
19	20	26	35	12	2	4	39	12	27
20	41	10	36	19	9	22	17	27	47

21	9	47	37	26	21	45	0	42	14
22	30	31	38	33	33	15	26	4	34
23	3	15	39	40	45	38	4	19	1
24	19	4	40	47	4	8	35	34	21
25	40	36	41	1	16	31	13	1	41
26	8	20	42	8	28	1	44	11	8
27	-1	-1	43	15	40	24	22	26	28
28	-1	-1	44	22	4	47	0	41	0
29	-1	-1	-1	-1	-1	-1	-1	-1	-1
30	-1	-1	-1	-1	-1	-1	-1	-1	-1
31	-1	-1	-1	-1	-1	-1	-1	-1	-1
32	-1	-1	-1	-1	-1	-1	-1	-1	-1

r/c	64	65	66	67	68	69	70	71	72
1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	-1	-1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	-1	-1
7	-1	-1	-1	-1	-1	-1	-1	-1	-1
8	-1	-1	-1	-1	-1	-1	-1	-1	-1
9	-1	-1	-1	-1	-1	-1	-1	-1	-1
10	-1	-1	-1	-1	-1	-1	-1	-1	-1
11	-1	-1	-1	-1	-1	-1	-1	-1	-1
12	-1	-1	-1	-1	-1	-1	-1	-1	-1
13	-1	-1	-1	-1	-1	-1	-1	-1	-1
14	-1	-1	-1	-1	-1	-1	-1	-1	-1
15	-1	-1	-1	-1	-1	-1	-1	-1	-1
16	0	-1	-1	-1	-1	-1	-1	-1	-1
17	26	40	18	8	26	32	7	0	-1
18	4	2	46	26	9	34	24	22	0
19	25	7	21	44	45	36	41	44	38
20	3	17	1	9	28	38	5	13	23
21	24	27	24	27	11	40	22	35	8
22	2	37	4	45	47	42	39	4	46
23	23	47	27	10	30	44	3	26	31
24	1	4	2	28	13	46	20	0	16
25	22	14	30	46	1	0	37	17	1
26	0	24	5	11	32	2	1	39	39
27	21	34	33	29	15	4	18	8	24

28	47	44	8	47	3	1	35	30	9
29	-1	1	36	12	34	3	4	4	47
30	-1	11	11	30	17	5	16	11	32
31	-1	-1	-1	-1	-1	-1	-1	-1	-1
32	-1	-1	-1	-1	-1	-1	-1	-1	-1

r/c	73	74	75	76	77	78	79	80	81
1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	-1	-1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	-1	-1
7	-1	-1	-1	-1	-1	-1	-1	-1	-1
8	-1	-1	-1	-1	-1	-1	-1	-1	-1
9	-1	-1	-1	-1	-1	-1	-1	-1	-1
10	-1	-1	-1	-1	-1	-1	-1	-1	-1
11	-1	-1	-1	-1	-1	-1	-1	-1	-1
12	-1	-1	-1	-1	-1	-1	-1	-1	-1
13	-1	-1	-1	-1	-1	-1	-1	-1	-1
14	-1	-1	-1	-1	-1	-1	-1	-1	-1
15	-1	-1	-1	-1	-1	-1	-1	-1	-1
16	-1	-1	-1	-1	-1	-1	-1	-1	-1
17	-1	-1	-1	-1	-1	-1	-1	-1	-1
18	-1	-1	-1	-1	-1	-1	-1	-1	-1
19	38	14	4	22	17	34	0	-1	-1
20	46	38	18	1	44	38	29	0	-1
21	1	9	32	33	18	42	5	39	33
22	9	33	46	12	45	46	34	25	36
23	17	4	7	44	19	2	10	11	39
24	25	28	21	23	46	1	39	2	42
25	33	4	35	2	20	5	15	36	45
26	41	23	1	34	47	9	44	22	0
27	1	47	10	13	21	13	20	8	3
28	4	18	24	45	0	17	1	47	1
29	12	42	38	24	22	21	25	33	4
30	20	13	4	3	1	25	1	19	7
31	28	37	13	35	23	29	30	5	10
32	36	8	27	14	2	33	6	44	13

r/c	82	83	84	85	86	87	88	89	90
1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	-1	-1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	-1	-1
7	-1	-1	-1	-1	-1	-1	-1	-1	-1
8	-1	-1	-1	-1	-1	-1	-1	-1	-1
9	-1	-1	-1	-1	-1	-1	-1	-1	-1
10	-1	-1	-1	-1	-1	-1	-1	-1	-1
11	-1	-1	-1	-1	-1	-1	-1	-1	-1
12	-1	-1	-1	-1	-1	-1	-1	-1	-1
13	-1	-1	-1	-1	-1	-1	-1	-1	-1
14	-1	-1	-1	-1	-1	-1	-1	-1	-1
15	-1	-1	-1	-1	-1	-1	-1	-1	-1
16	-1	-1	-1	-1	-1	-1	-1	-1	-1
17	-1	-1	-1	-1	-1	-1	-1	-1	-1
18	-1	-1	-1	-1	-1	-1	-1	-1	-1
19	-1	-1	-1	-1	-1	-1	-1	-1	-1
20	-1	-1	-1	-1	-1	-1	-1	-1	-1
21	6	15	36	30	24	0	-1	-1	-1
22	22	40	42	17	5	13	0	-1	-1
23	38	12	0	4	39	26	33	35	20
24	1	37	1	44	20	39	13	40	2
25	17	9	7	31	1	4	46	45	27
26	33	34	13	18	35	12	26	2	4
27	1	6	19	5	16	25	6	2	34
28	12	31	25	45	2	38	39	7	11
29	28	3	31	32	31	3	19	12	41
30	44	28	37	19	12	11	4	17	18
31	7	0	43	6	46	24	32	22	0
32	23	25	1	46	27	37	12	27	25

r/c	91	92	93	94	95	96	97	98	99
1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	-1	-1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	-1	-1

7	-1	-1	-1	-1	-1	-1	-1	-1	-1
8	-1	-1	-1	-1	-1	-1	-1	-1	-1
9	-1	-1	-1	-1	-1	-1	-1	-1	-1
10	-1	-1	-1	-1	-1	-1	-1	-1	-1
11	-1	-1	-1	-1	-1	-1	-1	-1	-1
12	-1	-1	-1	-1	-1	-1	-1	-1	-1
13	-1	-1	-1	-1	-1	-1	-1	-1	-1
14	-1	-1	-1	-1	-1	-1	-1	-1	-1
15	-1	-1	-1	-1	-1	-1	-1	-1	-1
16	-1	-1	-1	-1	-1	-1	-1	-1	-1
17	-1	-1	-1	-1	-1	-1	-1	-1	-1
18	-1	-1	-1	-1	-1	-1	-1	-1	-1
19	-1	-1	-1	-1	-1	-1	-1	-1	-1
20	-1	-1	-1	-1	-1	-1	-1	-1	-1
21	-1	-1	-1	-1	-1	-1	-1	-1	-1
22	-1	-1	-1	-1	-1	-1	-1	-1	-1
23	1	25	39	9	0	-1	-1	-1	-1
24	12	7	0	28	21	0	-1	-1	-1
25	23	42	4	47	42	37	31	37	2
26	34	24	13	13	10	21	32	44	14
27	45	6	22	32	31	5	33	3	26
28	3	41	31	3	4	42	34	5	38
29	14	23	40	17	20	26	35	12	2
30	25	5	1	36	41	10	36	19	9
31	36	40	5	2	9	47	37	26	21
32	47	22	14	21	30	31	38	33	33

r/c	100	101	102	103	104	105	106	107	108
1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	-1	-1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	-1	-1
7	-1	-1	-1	-1	-1	-1	-1	-1	-1
8	-1	-1	-1	-1	-1	-1	-1	-1	-1
9	-1	-1	-1	-1	-1	-1	-1	-1	-1
10	-1	-1	-1	-1	-1	-1	-1	-1	-1
11	-1	-1	-1	-1	-1	-1	-1	-1	-1
12	-1	-1	-1	-1	-1	-1	-1	-1	-1
13	-1	-1	-1	-1	-1	-1	-1	-1	-1

14	-1	-1	-1	-1	-1	-1	-1	-1	-1
15	-1	-1	-1	-1	-1	-1	-1	-1	-1
16	-1	-1	-1	-1	-1	-1	-1	-1	-1
17	-1	-1	-1	-1	-1	-1	-1	-1	-1
18	-1	-1	-1	-1	-1	-1	-1	-1	-1
19	-1	-1	-1	-1	-1	-1	-1	-1	-1
20	-1	-1	-1	-1	-1	-1	-1	-1	-1
21	-1	-1	-1	-1	-1	-1	-1	-1	-1
22	-1	-1	-1	-1	-1	-1	-1	-1	-1
23	-1	-1	-1	-1	-1	-1	-1	-1	-1
24	-1	-1	-1	-1	-1	-1	-1	-1	-1
25	13	21	5	0	-1	-1	-1	-1	-1
26	36	4	20	20	0	-1	-1	-1	-1
27	6	30	35	40	26	40	18	8	26
28	29	8	2	7	4	2	46	26	9
29	4	39	12	27	25	7	21	44	45
30	22	17	27	47	3	17	1	9	28
31	45	0	42	14	24	27	24	27	11
32	15	26	4	34	2	37	4	45	47

r/c	109	110	111	112	113	114	115	116	117
1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	-1	-1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	-1	-1
7	-1	-1	-1	-1	-1	-1	-1	-1	-1
8	-1	-1	-1	-1	-1	-1	-1	-1	-1
9	-1	-1	-1	-1	-1	-1	-1	-1	-1
10	-1	-1	-1	-1	-1	-1	-1	-1	-1
11	-1	-1	-1	-1	-1	-1	-1	-1	-1
12	-1	-1	-1	-1	-1	-1	-1	-1	-1
13	-1	-1	-1	-1	-1	-1	-1	-1	-1
14	-1	-1	-1	-1	-1	-1	-1	-1	-1
15	-1	-1	-1	-1	-1	-1	-1	-1	-1
16	-1	-1	-1	-1	-1	-1	-1	-1	-1
17	-1	-1	-1	-1	-1	-1	-1	-1	-1
18	-1	-1	-1	-1	-1	-1	-1	-1	-1
19	-1	-1	-1	-1	-1	-1	-1	-1	-1
20	-1	-1	-1	-1	-1	-1	-1	-1	-1

21	-1	-1	-1	-1	-1	-1	-1	-1	-1
22	-1	-1	-1	-1	-1	-1	-1	-1	-1
23	-1	-1	-1	-1	-1	-1	-1	-1	-1
24	-1	-1	-1	-1	-1	-1	-1	-1	-1
25	-1	-1	-1	-1	-1	-1	-1	-1	-1
26	-1	-1	-1	-1	-1	-1	-1	-1	-1
27	32	7	0	-1	-1	-1	-1	-1	-1
28	34	24	22	0	-1	-1	-1	-1	-1
29	36	41	44	38	38	14	4	22	17
30	38	5	13	23	46	38	18	1	44
31	40	22	35	8	1	9	32	33	18
32	42	39	4	46	9	33	46	12	45

r/c	118	119	01-dic	121	122	123	124	125	126
1	-1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	-1	-1	-1	-1	-1
3	-1	-1	-1	-1	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	-1	-1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	-1	-1
7	-1	-1	-1	-1	-1	-1	-1	-1	-1
8	-1	-1	-1	-1	-1	-1	-1	-1	-1
9	-1	-1	-1	-1	-1	-1	-1	-1	-1
10	-1	-1	-1	-1	-1	-1	-1	-1	-1
11	-1	-1	-1	-1	-1	-1	-1	-1	-1
12	-1	-1	-1	-1	-1	-1	-1	-1	-1
13	-1	-1	-1	-1	-1	-1	-1	-1	-1
14	-1	-1	-1	-1	-1	-1	-1	-1	-1
15	-1	-1	-1	-1	-1	-1	-1	-1	-1
16	-1	-1	-1	-1	-1	-1	-1	-1	-1
17	-1	-1	-1	-1	-1	-1	-1	-1	-1
18	-1	-1	-1	-1	-1	-1	-1	-1	-1
19	-1	-1	-1	-1	-1	-1	-1	-1	-1
20	-1	-1	-1	-1	-1	-1	-1	-1	-1
21	-1	-1	-1	-1	-1	-1	-1	-1	-1
22	-1	-1	-1	-1	-1	-1	-1	-1	-1
23	-1	-1	-1	-1	-1	-1	-1	-1	-1
24	-1	-1	-1	-1	-1	-1	-1	-1	-1
25	-1	-1	-1	-1	-1	-1	-1	-1	-1
26	-1	-1	-1	-1	-1	-1	-1	-1	-1
27	-1	-1	-1	-1	-1	-1	-1	-1	-1

28	-1	-1	-1	-1	-1	-1	-1	-1	-1
29	34	0	-1	-1	-1	-1	-1	-1	-1
30	38	29	0	-1	-1	-1	-1	-1	-1
31	42	5	39	33	6	15	36	30	24
32	46	34	25	36	22	40	42	17	5

r/c	127	128
1	-1	-1
2	-1	-1
3	-1	-1
4	-1	-1
5	-1	-1
6	-1	-1
7	-1	-1
8	-1	-1
9	-1	-1
10	-1	-1
11	-1	-1
12	-1	-1
13	-1	-1
14	-1	-1
15	-1	-1
16	-1	-1
17	-1	-1
18	-1	-1
19	-1	-1
20	-1	-1
21	-1	-1
22	-1	-1
23	-1	-1
24	-1	-1
25	-1	-1
26	-1	-1
27	-1	-1
28	-1	-1
29	-1	-1
30	-1	-1
31	0	-1
32	13	0

Table 2

Fig. 4 illustrates an example of an encoding procedure which may be performed by the encoder CH_ENC. The rate of the LDPC-CTC is given by:

$$R = R_1 \cdot R_2 \cdot R_3 \cdot \dots \cdot R_{(w-1)} \cdot R_w = \frac{k_b}{k_b + v_1 + v_2 + \dots + v_{(w-1)} + v_w} \quad (11)$$

- 5 The LDPC-CTC with rate $R = k_i / n_c$ is constituted by w parallel LDPC-CC codes. The single LDPC-CCs may be designed with the following rates:

$$R_\beta = \frac{(w \cdot k_i - 1) + \beta}{w \cdot (n_c - 1) + \beta} \quad \text{with} \quad \beta = 1, 2, 3, \dots, w \quad (12)$$

- Considering, for example, a LDPC-CTC with rate $R = k_i / n_c = 3 / 4$ and constituted by $w=4$ parallel LDPC-CCs, the single LDPC-CCs have rates R_1, R_2, R_3, R_4 which may be
10 computed using equation (12) as:

$$R_\beta = \frac{(w \cdot k_i - 1) + \beta}{w \cdot (n_c - 1) + \beta} = \frac{11 + \beta}{12 + \beta} \quad \text{with} \quad \beta = 1, 2, 3, \quad w=4 \quad (13)$$

$$R_1 = \frac{12}{13}, R_2 = \frac{13}{14}, R_3 = \frac{14}{15}, R_4 = \frac{15}{16}$$

- In the implementation example of Fig. 5, a LDPC-CTC with rate $R = k_i / n_c = 3 / 4$ is used, which is based on two ($w=2$) parallel LDPC-CCs, a first LDPC-CC with rate $R_1 = k_{i1} / n_{c1} = 6 / 7$ and a second LDPC-CC with rate $R_2 = k_{i2} / n_{c2} = 7 / 8$. As
15 indicated above, the number of parallel LDPC-CCs may more than two and be chosen while taking into account the tradeoff between desired performance and hardware complexity.

To build a LDPC-CTC seed matrix as shown in Table 1, the first LDPC-CC seed matrix with rate $R_1 = 6/7$ may be arranged to interlace with the second LDPC-CC seed matrix with rate $R_2 = 7/8$. Arranging the first LDPC-CC seed matrix may comprise adding a (zero) column between consecutive sets of 7 columns of the matrix, wherein the added columns are dedicated to (and will be filled with one or more entries of) the parity sequences of the second LDPC-CC seed matrix with rate $R_2 = 7/8$.

The entries of the rows of the first “arranged” LDPC-CC seed matrix with rate $R_1 = 6/7$ may be copied to the odd rows of the LDPC-CTC seed matrix of Table 1 while the entries of the rows of the second LDPC-CC seed matrix with rate $R_2 = 7/8$ may be copied to the even rows of the LDPC-CTC seed matrix of Table 1.

Fig. 6 shows a portion (consisting of 18 rows and 24 columns) of the LDPC-CTC seed matrix of Table 1, wherein the rows filled with entries of the first LDPC-CC seed matrix with rate $R_1 = 6/7$ and the rows filled with entries of the second LDPC-CC seed matrix with rate $R_2 = 7/8$ have been highlighted with using different shades of gray. The following correspondence exists between the rows of the first/second LDPC-CC seed matrix and the rows of the LDPC-CTC seed matrix:

First LDPC-CC

row 1 →
 “ 2 →
 “ 3 →
 “ 4 →
 “ 5 →
 “ 6 →

LDPC-CTC

row 1
 “ 3
 “ 5
 “ 7
 “ 9
 “ 11 and so on.

Second LDPC-CC

row 1 →
 “ 2 →
 “ 3 →
 “ 4 →
 “ 5 →

LDPC-CTC)

row 2
 “ 4
 “ 6
 “ 8
 “ 10

“ 6 → “ 12 and so on.

Therefore, although the LDPC-CTC may be constructed from several LDPC-CCs placed in parallel form, the LDPC-CTC can be represented by a single seed matrix, a single exponent parity-check matrix and a single binary parity-check matrix, which may be used in the encoding/decoding procedures.

As for LDPC-BCs carried-out with semi-random technique, the exponent parity-check matrix of the LDPC-CTC may be used in the encoding procedure because it takes advantage of the cyclic permutation matrix features. In the decoding process, the binary parity-check matrix of the LDPC-CTC may be used.

For the first LDPC-CC, equation (7), which governs the encoding procedure, remains unchanged and, for a code-rate $R_1 = 6/7$, can be expressed as:

$$\begin{cases} v_{n1}^{cc1} = u_{n1}, v_{n2}^{cc1} = u_{n2}, v_{n3}^{cc1} = u_{n3}, v_{n4}^{cc1} = u_{n4}, v_{n5}^{cc1} = u_{n5}, v_{n6}^{cc1} = u_{n6} \\ v_{n7}^{cc1} = \sum_{i=0}^M h_1^{(i)}(n) \cdot u_{(n-i)1} + \sum_{i=0}^M h_2^{(i)}(n) \cdot u_{(n-i)2} + \dots \sum_{i=0}^M h_6^{(i)}(n) \cdot u_{(n-i)6} + \sum_{i=1}^M h_7^{(i)}(n) \cdot v_{(n-i)7} \end{cases} \quad (15)$$

For the second LDPC-CC with code rate $R_2 = 7/8$, equation (7) becomes:

$$\begin{cases} v_{n1}^{cc2} = u_{n1}, v_{n2}^{cc2} = u_{n2}, v_{n3}^{cc2} = u_{n3}, \dots, v_{n6}^{cc2} = u_{n6}, v_{n7}^{cc2} = v_{n7}^{cc1} \\ v_{n8}^{cc2} = \sum_{i=0}^M \gamma_1^{(i)}(n) \cdot u_{(n-i)1} + \sum_{i=0}^M \gamma_2^{(i)}(n) \cdot u_{(n-i)2} + \dots \sum_{i=0}^M \gamma_7^{(i)}(n) \cdot v_{(n-i)7}^{cc1} + \sum_{i=1}^M \gamma_8^{(i)}(n) \cdot v_{(n-i)8}^{cc2} \end{cases} \quad (16)$$

To carry-out the LDPC-CTC encoding procedure, equation (15) and equation (16) may be rearranged as:

$$\begin{cases}
v_{n1}^{cc2} = v_{n1}^{cc1} = u_{n1}, v_{n2}^{cc2} = v_{n2}^{cc1} = u_{n2}, \dots, v_{n5}^{cc2} = v_{n5}^{cc1} = u_{n5}, v_{n6}^{cc2} = v_{n6}^{cc1} = u_{n6} \\
v_{n7}^{cc1} = \sum_{i=0}^M h_1^{(i)}(n) \cdot u_{(n-i)1} + \sum_{i=0}^M h_2^{(i)}(n) \cdot u_{(n-i)2} + \dots + \sum_{i=0}^M h_6^{(i)}(n) \cdot u_{(n-i)6} + \sum_{i=1}^M h_7^{(i)}(n) \cdot v_{(n-i)7}^{cc1} \\
v_{n8}^{cc2} = \sum_{i=0}^M \gamma_1^{(i)}(n) \cdot u_{(n-i)1} + \sum_{i=0}^M \gamma_2^{(i)}(n) \cdot u_{(n-i)2} + \dots + \sum_{i=0}^M \gamma_7^{(i)}(n) \cdot v_{(n-i)7} + \sum_{i=1}^M \gamma_8^{(i)}(n) \cdot v_{(n-i)8}^{cc2}
\end{cases} \quad (17)$$

Equation (17) allows to carry-out the LDPC-CTC encoding procedure with a code-rate $R = R_1 \cdot R_2 = (6/7) \cdot (7/8) = 3/4$.

Thus, the LDPC-CTC seed matrix in Table 1 may be masked with the full exponent matrix E_{exp} in Table 2 for the rate R and expanded with a spreading matrix (cyclic permutation matrix) to produce a binary parity-check matrix H_{rxn} of the LDPC-CTC which allows parallel layered decoding with good performance (fast convergence of the belief propagation algorithm) and low decoding latency.

In this regard, it is noted that the code-rate of a LDPC-CTC for tail-biting (TB-LDPC-CTC) is tied to the rate of the code $R = k_i / n_c$, the parallel LDPC-CC number w , the code period length T , the code period number p and the parity element number $r = n_c - k_i$.

The LDPC-CTC seed matrix of Table 1 masked with the full exponent matrix E_{exp} of Table 2 has the following characteristics:

$$p \cdot w \cdot n_c \cdot T + w \cdot n_c = (p \cdot T + 1) \cdot w \cdot n_c \Rightarrow \text{Exponent parity-check matrix column number}$$

$$(p-1) \cdot w \cdot k_i \cdot T \Rightarrow \text{Information column number for tail-biting}$$

$$p \cdot w \cdot r \cdot T + w \cdot r = (p \cdot T + 1) \cdot w \cdot r \Rightarrow \text{Exponent parity-check matrix row number}$$

$H_{\text{exp}}((p \cdot T + 1) \cdot w \cdot r, (p \cdot T + 1) \cdot w \cdot n_c) \Rightarrow$ Exponent parity-check matrix size

The rate for the TB-LPCC-CTC becomes:

$$R_{TB}^{CTC} = \frac{(p-1) \cdot w \cdot k_i \cdot T}{(p-1) \cdot w \cdot k_i \cdot T + (p \cdot T + 1) \cdot w \cdot r} = \frac{(p-1) \cdot w \cdot k_i \cdot T}{(p \cdot n_c - k_i) \cdot w \cdot T + (n_c - k_i) \cdot w} \quad (18)$$

The LDPC-CTC seed matrix of Table 1 masked with the full exponent matrix E_{exp} of Table 2 is periodic for rows and for columns. The periodicity of the rows is equal to the product $w \cdot r \cdot T = w \cdot (n_c - k_i) \cdot T = 10$ while the periodicity of the columns is equal to the product $w \cdot n_c \cdot T = 40$. Thus, it is not necessary to store the exponent parity-check matrix H_{exp} and the binary parity-check matrix H_{rxn} in their real size, but it suffices to store $w \cdot r \cdot T = 10$ rows for the exponent matrix and $w \cdot r \cdot T \cdot Z_f = 10 \cdot 48 = 480$ rows for the binary matrix.

In order to construct the exponent matrix, using well-known construction rules, it may be operated on columns because this allows using 40 incremental steps (unlike the same operation for rows which would only allow using 10 incremental steps). By operating on columns, it may hence be possible to reduce the number of cycles with length 4 in the exponent parity matrix and consequently to improve the code performance. I.e., the increase of the number of incremental steps in the exponent parity matrix construction may allow to achieve an overlap factor (the scalar product between any two columns of the binary parity-check matrix H_{rxn}) less than or equal to '1' (≤ 1).

An overlap factor ≤ 1 makes the LDPC-CC robust, i.e. avoids cycles of length 4 which may degrade performance of the iterative decoding algorithm which reduces code performance.

In the encoding procedure, the exponent parity-check matrix H_{exp} may be used to take advantage of the following properties:

- the product between two permutation matrixes of the identity matrix with exponents α and β produces always a permutation matrix of the identity matrix with exponent $(\alpha + \beta)$; and
- the product between a cyclic permutation matrix of the identity matrix with exponent α and a vector of size Z_f produces the same vector of size Z_f but with a cyclic shift of its elements equal to α .

In the decoding procedure, the binary parity-check matrix may be used, because the iterative algorithm updates the single reliability value which corresponds to the single bit.

With the block structured information sequence being denoted by $U_{[0,n]}$ and the block structured encoded sequence being denoted by $V_{[0,n]}$:

$$U_{[0,n]} = [U_{01}, U_{02}, \dots, U_{06}, U_{11}, U_{12}, \dots, U_{16}, \dots, U_{n1}, U_{n2}, \dots, U_{n6}] \quad (19)$$

$$V_{[0,n]} = [V_{01}, V_{02}, \dots, V_{06}, V_{07}, V_{08}, V_{11}, V_{12}, \dots, V_{16}, V_{17}, V_{18}, \dots, V_{n1}, V_{n2}, \dots, V_{n6}, V_{n7}, V_{n8}] \quad (20)$$

the single elements $\{U_{ij}, V_{ij}\}$ are vectors of size equal to spreading factor $Z_f = 48$ and the received sequence will be correct, if the following equation will be satisfied:

$$\hat{V}_{[0,n]} \cdot H_{\text{exp}}^T = 0 \quad (21)$$

The systematic block structured LDPC-CTC encoding procedure derived from equation (17) may be carried-out based on:

$$\left\{ \begin{array}{l} V_{n1} = U_{n1} \quad , \quad V_{n2} = U_{n2} \quad , \quad \dots \quad , \quad V_{n5} = U_{n5} \quad , \quad V_{n6} = U_{n6} \\ V_{n7} = \sum_{i=0}^M H_1^{(i)}(n) \cdot U_{(n-i)1} + \sum_{i=0}^M H_2^{(i)}(n) \cdot U_{(n-i)2} + \dots + \sum_{i=0}^M H_6^{(i)}(n) \cdot U_{(n-i)6} + \sum_{i=1}^M H_7^{(i)}(n) \cdot V_{(n-i)7} \\ V_{n8} = \sum_{i=0}^M \Gamma_1^{(i)}(n) \cdot U_{(n-i)1} + \sum_{i=0}^M \Gamma_2^{(i)}(n) \cdot U_{(n-i)2} + \dots + \sum_{i=0}^M \Gamma_7^{(i)}(n) \cdot V_{n7} + \sum_{i=1}^M \Gamma_8^{(i)}(n) \cdot V_{(n-i)8} \end{array} \right. \quad (22)$$

In the example, the exponent $H_1^{(i)}(n), H_2^{(i)}(n), \dots, H_6^{(i)}(n), H_7^{(i)}(n)$ and $\Gamma_1^{(i)}(n), \Gamma_2^{(i)}(n), \dots, \Gamma_7^{(i)}(n), \Gamma_8^{(i)}(n)$, of the LDPC-CTC parity-check matrix are cyclic permutation matrices of size $Z_f = 48$. The coefficients $H_1^{(i)}(n), H_2^{(i)}(n), \dots, H_6^{(i)}(n)$ and $\Gamma_1^{(i)}(n), \Gamma_2^{(i)}(n), \dots, \Gamma_7^{(i)}(n)$ are tied to the information sequence while the coefficients $H_7^{(i)}(n)$ and $\Gamma_8^{(i)}(n)$ are tied to the parity sequence, respectively, of the first LDPC-CC and the second LDPC-CC. Each time slot $n = 0, 1, 2, 3, \dots$ corresponds to a block of $w \cdot n_c \cdot Z_f = 384$ bits (with $w \cdot k_i \cdot Z_f = 288$ information bits and $w \cdot r \cdot Z_f = 96$ parity bits).

The coefficients $H_j^{(i)}(n), \Gamma_j^{(i)}(n)$ are always comprised between '0' and $(Z_f - 1) = 47$, but the coefficients $H_7^{(0)}(n), \Gamma_8^{(0)}(n)$, which correspond to the parity columns of the exponent matrix, may be set equal to '0' to minimize the hardware complexity of the LDPC-CTC encoding process. Hence, the coefficient $H_7^{(0)}(n), \Gamma_8^{(0)}(n)$ may correspond to the identity matrix.

In other words, to reduce the number of arithmetic operations in the LDPC-CTC encoding procedure, the exponent parity-check matrix may be constructed by putting to '0' (cyclic permutation matrix equal to the identity matrix) the last exponent (rightmost element) of each row ($H_7^{(0)}(n), \Gamma_8^{(0)}(n)$ for all time slots $n = 0, 1, 2, 3, \dots$) corresponding to the encoded parity sequence.

As communication services may require different degrees of latency, according to circumstances/transmission requirements like real-time broadcasting, time sensitive service and fast responsiveness of networks for the tactile internet, the internet of things (IoT), etc. and the

encoding/decoding procedures may represent the predominant reason for latency introduced to the system, a possibility to adapt the latency introduced by the encoding/decoding procedures may be beneficial.

5 Current systems usually support different requirements regarding latency and a dynamic change of latency by exchanging information between the decoder CH_DEC and the encoder CH_ENC. This exchange of control messages can be time consuming and produce overhead in the network which leads to low performance of the procedure.

10 The control messages may include information regarding a modified encoding matrix, i.e., an encoding matrix of another size that permits the encoder to reduce the length n' of the code-word, where $n' \neq n$ to obtain a (substantial) change in the latency. In case of block codes, the consecutive blocks are encoded independently of each other and decoding cannot start before the complete frame of n bits is available at the input of the decoder. This approach, which is illustrated in Fig. 7, may be referred to as block-based decoding.

15 As with LDPC-CCs, the encoding procedure depends on several input blocks in accordance with the memory of the code. The input block length for convolutional codes may be, for example, in the order of a few information bits. The decoding can be performed as soon as the first parity bit is available at the decoder CH_DEC. This approach, which is illustrated in Fig. 8, may be referred to as slot-based decoding.

20 To perform a slot-based decoding procedure and to exploit the small structural latency of LDPC-CC, a modified decoding procedure is used. The inner core of the decoding procedure is based on a simplified version of the standard BPA but the IN-OUT memories are managed in a different manner.

25 As illustrated in Fig. 9, a single iteration is performed starting from the last new LLRs acquired in the current TS and decoding involves updating the statistical values up to the last LLRs memorized (wherein the memorized LLRs may correspond to several code periods depending on the memory depth). Old LLRs in the memory of the decoder CH_DEC are iterated several times and can be dropped out from the memory according to the desired latency/performance.

The latency involved in carrying-out of the decoding procedure can be modified by the decoder CH_DEC without a need to inform the encoder CH_ENC, as no change of the encoding matrix is required and the size of the parity-check matrix of the LDPC-CTC is independent of the code-word length.

- 5 Fig. 10 shows a flow chart of steps of the decoding procedure. After having received the channel output CH_OUT, the demodulator DEM (which may be integrated into the decoder CH_DEC or into separate device) may demodulate the channel output CH_OUT and the decoding procedure may start at the decoder CH_DEC by determining LLRs corresponding to the information and parity bits of the binary word A2, as indicated at step 10 of the flowchart shown
10 in Fig. 10. The length of the binary word A2 may be equal to the length of a TS.

- Once one or more LLRs for the parity bits of the binary word A2 have been determined, a new decoding iteration may be started as indicated in step 12 of the flowchart shown in Fig. 10. As shown in Fig. 9, the LLRs which are currently kept in the memory of the decoder CH_DEC may be updated (e.g., based on a turbo decoding procedure) and once the decoding operation
15 has been completed, the sliding window may be advanced by one layer and the LLRs corresponding to the last layer that is dropped may be removed from the memory to provide space for LLRs of the new layer.

- The LLRs of the to-be-dropped columns may be used to make a hard decision on the received information bits as indicated at step 16. If the latency/memory requirements are to be adapted,
20 the memory depth may be modified as indicated in Fig. 9.

CLAIMS

1. A decoder for decoding a low-density parity-check convolutional code, LDPC-CTC, the decoder being configured to:

determine, based on a received signal, statistical values corresponding to information
5 and parity bits of an encoded information bit sequence;

start a new decoding iteration if a number of determined statistical values equals or exceeds a threshold, wherein a decoding iteration includes updating the determined statistical values based on actual determined statistical values and statistical values from a previous decoding iteration stored in a memory of the decoder;

10 replace a stored statistical value stored in the memory with an updated statistical value, wherein the updated statistical value is kept in the memory for a number of succeeding decoding iterations; and

decode the information bits to which the updated stored statistical values correspond.

2. The decoder of claim 1, wherein the decoder is further configured to:

15 remove/overwrite the updated stored statistical values from/in the memory of the decoder after the succeeding decoding iterations have been performed and the information bits corresponding to the updated stored statistical value have been decoded.

3. The decoder of claim 1 or 2, wherein the decoder is further configured to:

20 adapt the number of succeeding decoding operations for which the updated statistical values are kept in the memory of the decoder.

4. The decoder of claim 3, wherein the decoder is further configured to:

decrease the number of succeeding decoding operations for which the updated statistical values are kept in the memory of the decoder in response to a request to reduce a latency of the decoding operation.

25 5. The decoder of claim 3, wherein the decoder is further configured to:

increase the number of succeeding decoding operations for which the updated statistical values are kept in the memory of the decoder in response to a request to reduce an error rate of the decoding operation.

6. The decoder of any one of claims 1 to 5, wherein the threshold equals the number of information and parity bits of one layer of the LDPC-CTC.

7. The decoder of any one of claims 1 to 6, wherein updating the statistical values based on the statistical values and statistical values from a previous decoding iteration stored in a memory of the decoder includes parallelly updating different sets of statistical values assigned to different parity bits.

8. A method of decoding a low-density parity-check convolutional code, LDPC-CTC, the method comprising:

determining, based on a received signal, statistical values corresponding to information and parity bits of an encoded information bit sequence;

starting a new decoding iteration if a number of determined statistical values equals or exceeds a threshold, wherein a decoding iteration includes updating the determined statistical values based on actual determined statistical values and statistical values from a previous decoding iteration;

replacing a stored statistical value stored in the memory with an updated statistical value, wherein the updated statistical value is kept in the memory for a number of succeeding decoding iterations; and

decoding the information bits to which the updated stored statistical values correspond.

9. The method of claim 8, further comprising:

removing/overwriting the updated stored statistical values from/in the memory after the succeeding decoding iterations have been performed and the information bits corresponding to the updated stored statistical value have been decoded.

10. The method of claim 8 or 9, further comprising:

adapting the number of succeeding decoding operations for which the updated statistical values are kept in the memory.

11. The method of claim 10, further comprising:

5 decreasing the number of succeeding decoding operations for which the updated statistical values are kept in the memory in response to a request to reduce a latency of the decoding operation.

12. The method of claim 10, further comprising:

10 increasing the number of succeeding decoding operations for which the updated statistical values are kept in the memory in response to a request to reduce an error rate of the decoding operation.

13. The method of any one of claims 8 to 12, wherein the threshold equals the number of information and parity bits of one layer of the LDPC-CTC.

14. The method of any one of claims 8 to 13, wherein updating the statistical values based
15 on the statistical values and statistical values from a previous decoding iteration stored in a memory of the decoder includes parallelly updating different sets of statistical values assigned to different parity bits.

15. A machine-readable medium comprising machine-readable instructions which when
20 carried-out by the machine cause the machine to perform the method of any one of claims 8 to 14.

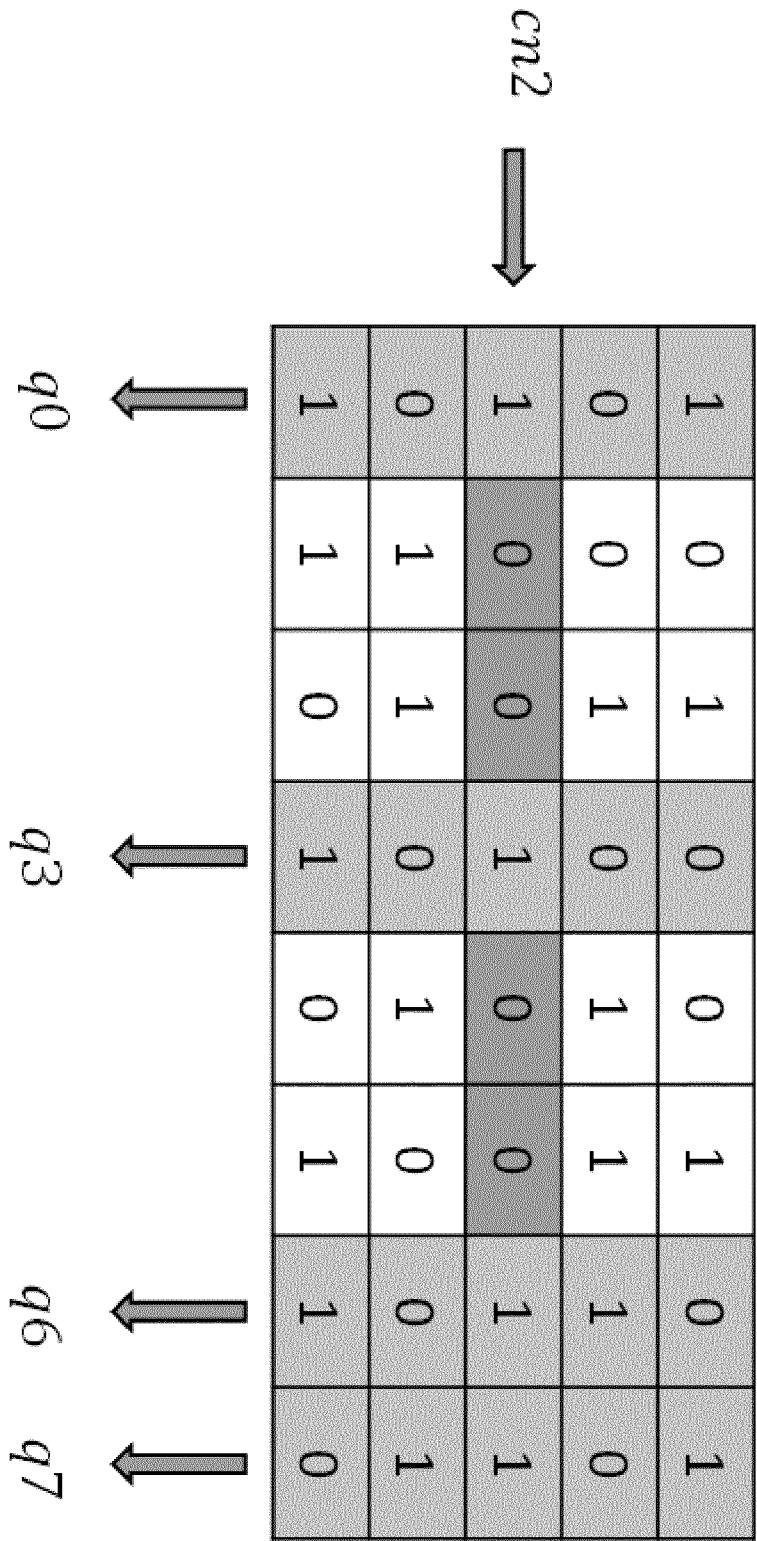


Fig. 1

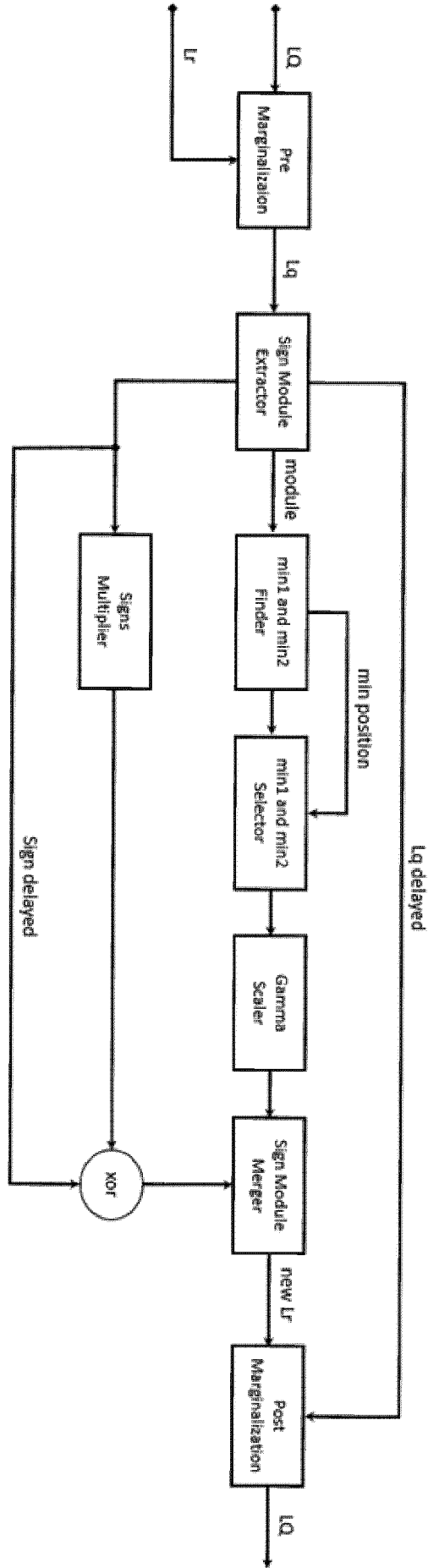


Fig. 2

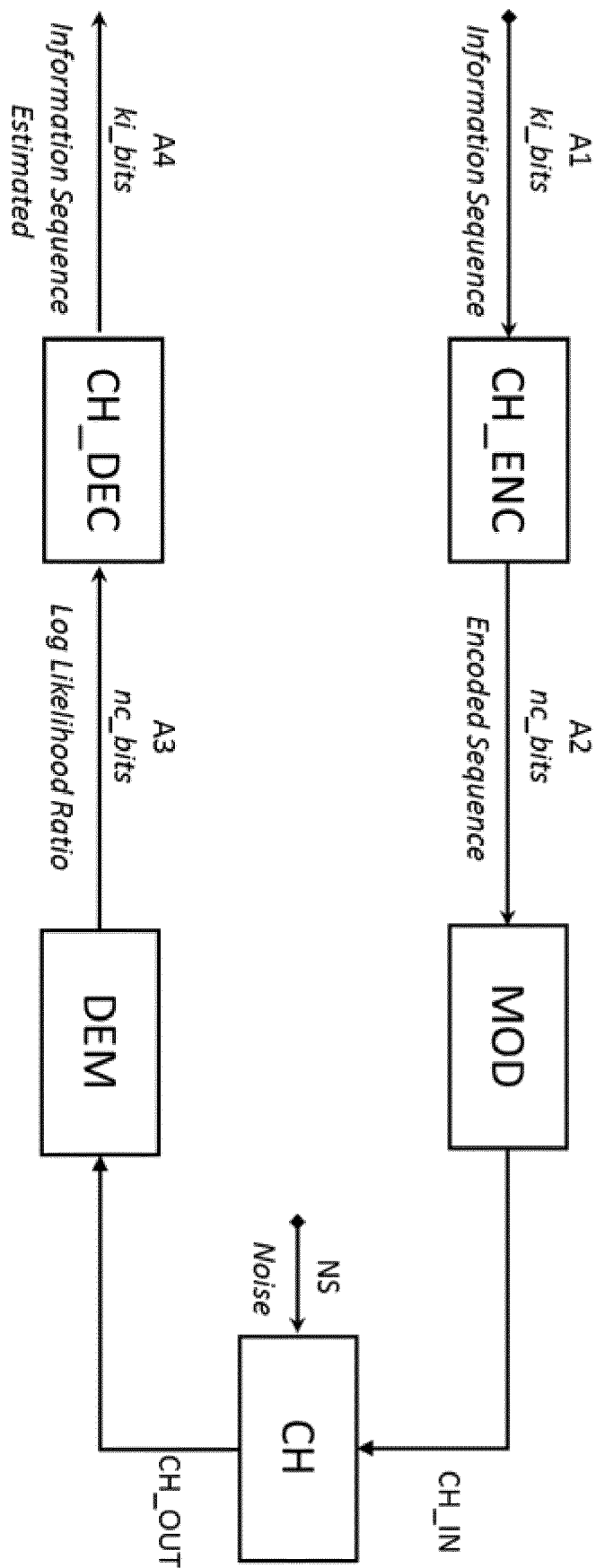


Fig. 3

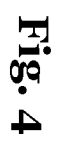
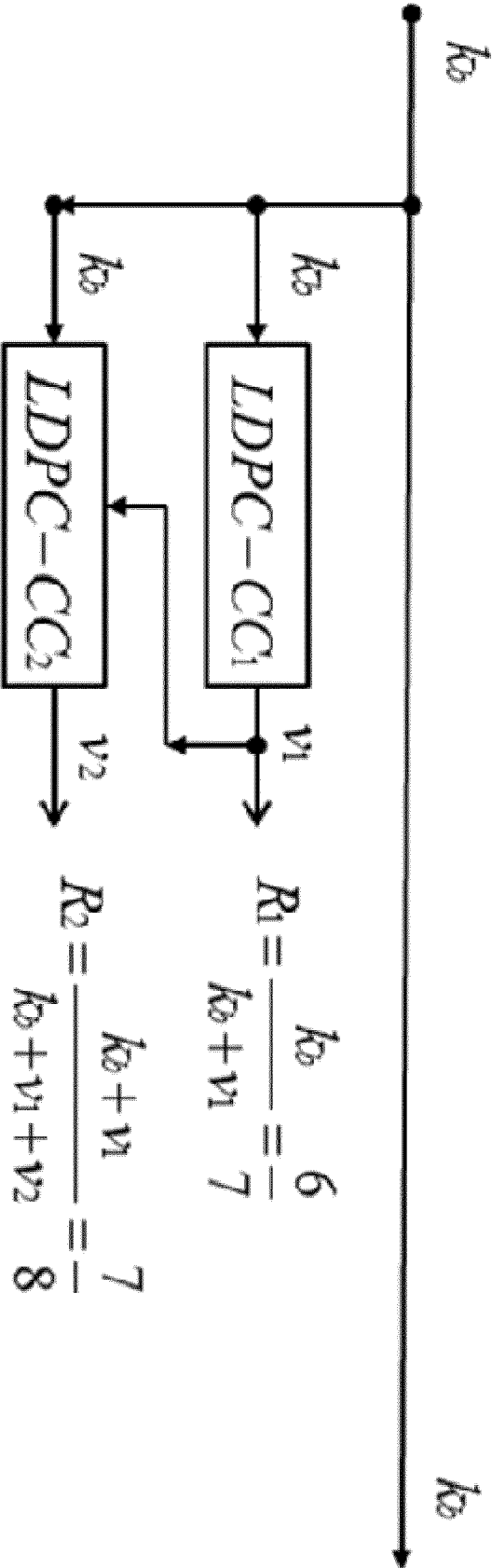


Fig. 4



$$R_1 = \frac{k_0}{k_0 + v_1} = \frac{6}{7}$$

$$R_2 = \frac{k_0 + v_1}{k_0 + v_1 + v_2} = \frac{7}{8}$$

$$R = \frac{k_0}{k_0 + v_1 + v_2} = R_1 \cdot R_2 = \frac{3}{4}$$

Fig. 5

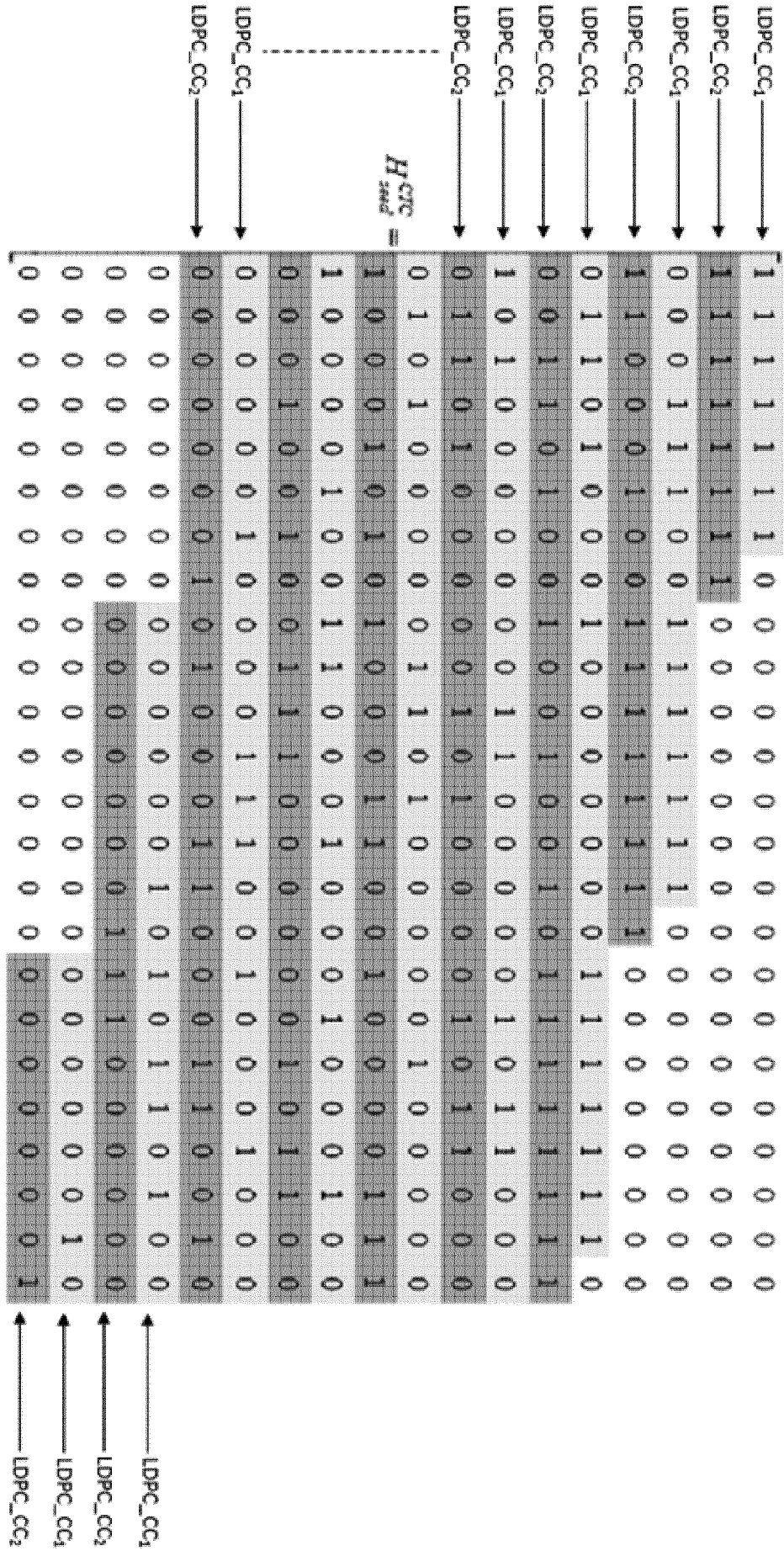


Fig. 6

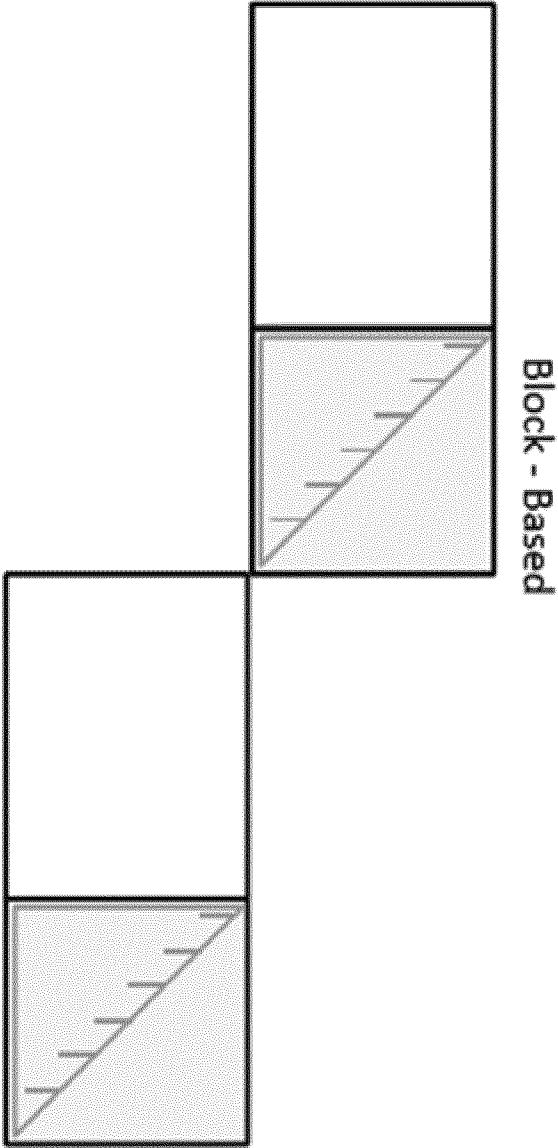
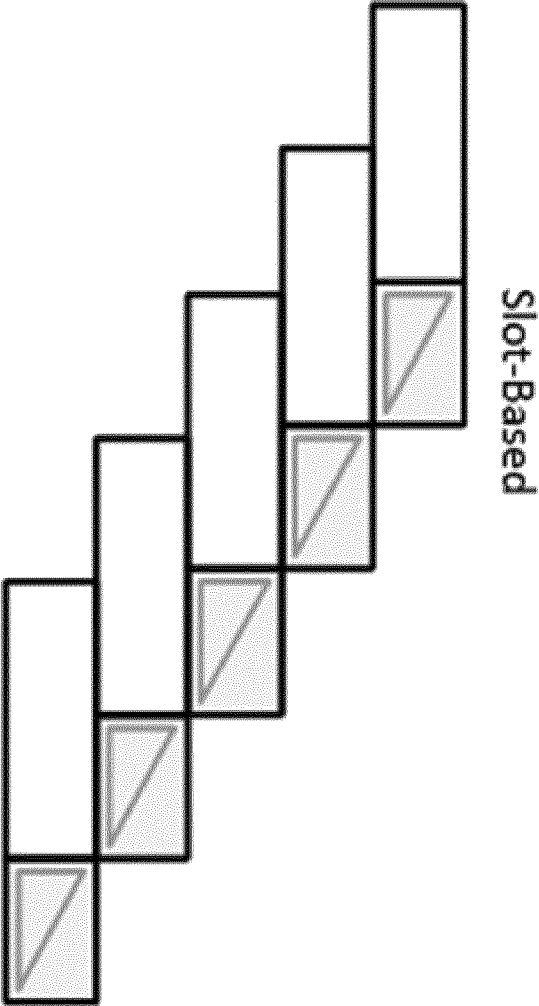


Fig. 7

Fig. 8



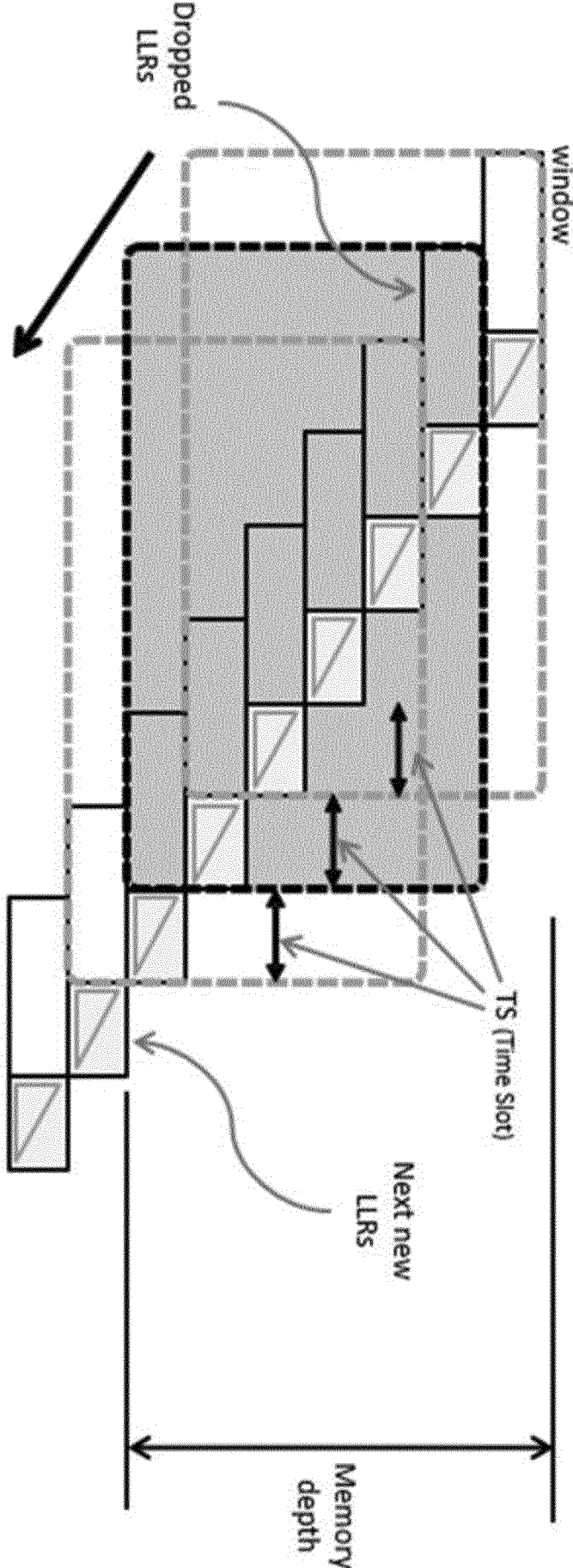
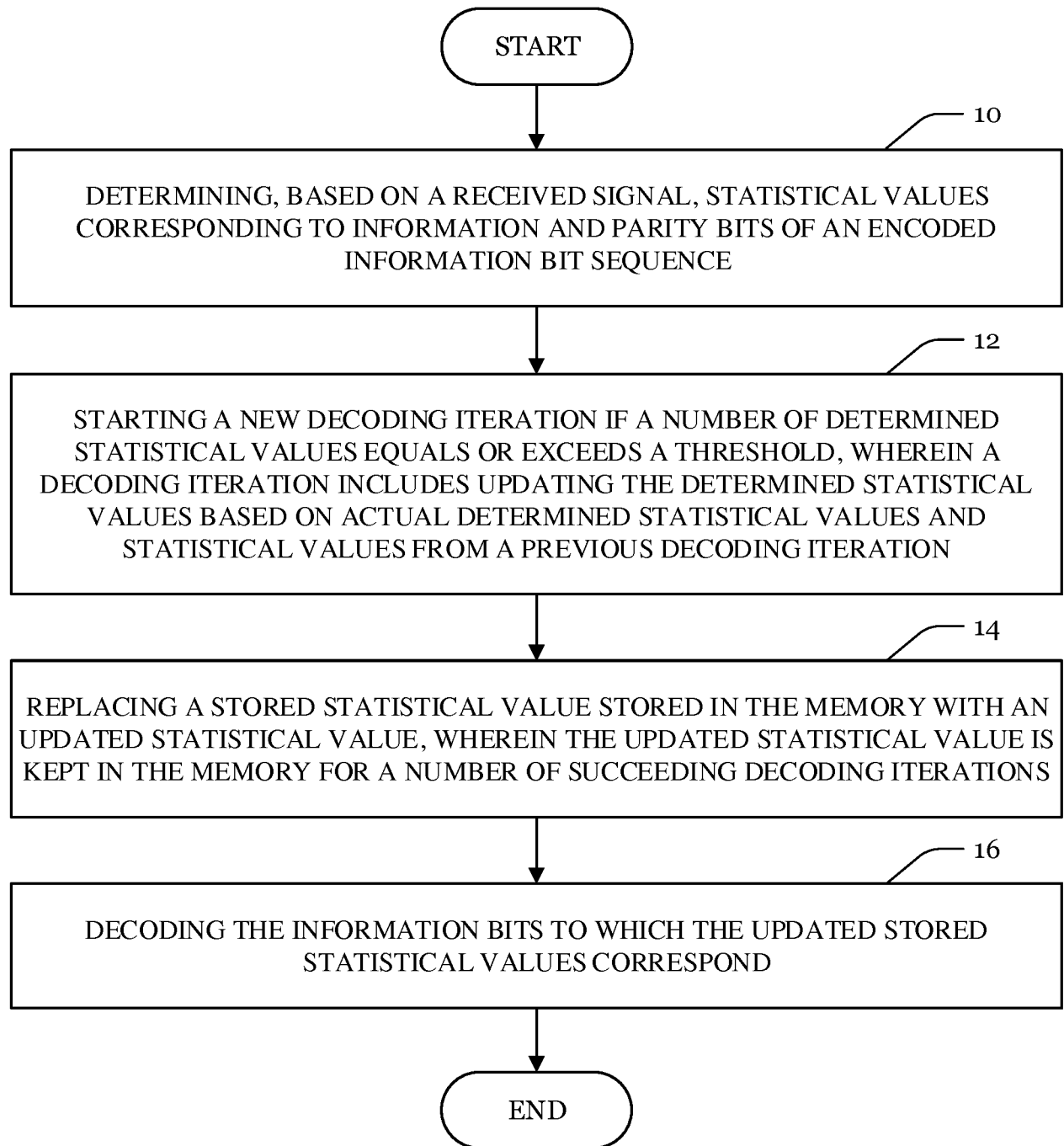


Fig. 9

**Fig. 10**

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2017/071793

A. CLASSIFICATION OF SUBJECT MATTER
INV. H03M13/11 H03M13/29
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H03M

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EP0-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	CORAZZA G E ET AL: "Latency constrained protograph-based LDPC convolutional codes", PROC., 6TH IEEE INTERNATIONAL SYMPOSIUM ON TURBO CODES AND ITERATIVE INFORMATION PROCESSING, ISTC 2010, 6 September 2010 (2010-09-06), pages 6-10, XP031783822, ISBN: 978-1-4244-6744-0 page 1 - page 3	1-15
X	US 2013/254633 A1 (OBATA HARUKA [JP] ET AL) 26 September 2013 (2013-09-26) paragraph [0033] - paragraph [0103]; figures 2,3,5 ----- -/--	1-15



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

8 May 2018

Date of mailing of the international search report

17/05/2018

Name and mailing address of the ISA/

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040,
Fax: (+31-70) 340-3016

Authorized officer

Offer, Elke

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2017/071793

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	PUSANE A E ET AL: "Implementation aspects of LDPC convolutional codes", IEEE TRANSACTIONS ON COMMUNICATIONS, IEEE SERVICE CENTER, PISCATAWAY, NJ. USA, vol. 56, no. 7, 1 July 2008 (2008-07-01), pages 1060-1069, XP011231533, ISSN: 0090-6778, DOI: 10.1109/TCOMM.2008.050519 page 1061 - page 1063 -----	1,2,7-9, 14,15
A	ZHENGANG CHEN ET AL: "High Throughput Parallel Decoder Design for LDPC Convolutional Codes", PROC., 4TH IEEE INTERNATIONAL CONFERENCE ON CIRCUITS AND SYSTEMS FOR COMMUNICATIONS, ICCSC 2008, 26 May 2008 (2008-05-26), pages 35-39, XP031268649, ISBN: 978-1-4244-1707-0 the whole document -----	1-15
A	Eran Pisek ET AL: "Capacity-Approaching TQC-LDPC Convolutional Codes Enabling Power-Efficient Decoders", IEEE Transactions on Communications, 10 October 2016 (2016-10-10), pages 1-13, XP055449345, DOI: 10.1109/TCOMM.2016.2616142 Retrieved from the Internet: URL:http://ieeexplore.ieee.org/ielx7/26/7818052/07587400.pdf?tp=&arnumber=7587400&isnumber=7818052 [retrieved on 2018-02-08] the whole document -----	1-15
A	PISEK ERAN ET AL: "Trellis-Based QC-LDPC Convolutional Codes Enabling Low Power Decoders", IEEE TRANSACTIONS ON COMMUNICATIONS, IEEE SERVICE CENTER, PISCATAWAY, NJ. USA, vol. 63, no. 6, 1 June 2015 (2015-06-01), pages 1939-1951, XP011584363, ISSN: 0090-6778, DOI: 10.1109/TCOMM.2015.2424434 [retrieved on 2015-06-12] the whole document -----	1-15

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/EP2017/071793

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2013254633 A1	26-09-2013	JP 2013198017 A	30-09-2013
		US 2013254633 A1	26-09-2013
