Design of High-Performance and Area-Efficient Decoder for 5G LDPC Codes

Hangxuan Cui, Fakhreddine Ghaffari, *Member, IEEE*, Khoa Le, *Member, IEEE*, David Declercq, *Senior Member, IEEE*, Jun Lin, *Senior Member, IEEE*, and Zhongfeng Wang, *Fellow, IEEE*

***Abstract*— Low-density parity-check (LDPC) code as a very promising error-correction code has been adopted as the channel coding scheme in the fifth-generation (5G) new radio. However, it is very challenging to design a high-performance decoder for 5G LDPC codes because their inherent numerous degree-1 variable-nodes are very prone to be erroneous. In this article, the problem is solved gracefully by developing a low-complexity check-node update function, greatly improving the reliability of check-to-variable messages. By further incorporating the proposed column degree adaptation strategy, our decoder could offer a 0.4dB performance gain over the existing ones. In addition, this article presents an efficient 5G LDPC decoder architecture. Benefiting the specific structure of 5G LDPC codes, layer merging, split storage method, and selective-shift structure are introduced to facilitate a significant reduction of decoding delay and area consumption. Implementation result on 90-nm CMOS technology demonstrates that the proposed decoder architecture yields an impressive improvement in throughput-to-area ratio, achieving up to 173.3% compared to conventional design.**

***Index Terms*— Low-density parity-check codes, 5G LDPC decoder, high-performance, VLSI implementation.**

I. Introduction

LOW-DENSITY parity-check (LDPC) codes [1] have attracted considerable attention over the past several decades because of their remarkable error-correction performance and inherent parallelism for hardware implementation. LDPC codes also have been adopted in several industrial standards, including IEEE 802.11 [2], the second generation satellite digital video broadcast (DVB-S2) [3], and advanced television system committee (ATSC) [4]. Recently, LDPC codes have been chosen as the 5G new radio (NR) channel coding scheme in the enhanced mobile broadband (eMBB) scenario [5]. LDPC codes can perform close to the Shannon limit when paired with the belief propagation (BP) decoding algorithm [6]. However, the BP algorithm involves complex non-linear functions in check-node (CN) processing, leading to large implementation complexity. As an alternative, the min-sum (MS) algorithm [7] was proposed and became the primary solutions in practical applications. By approximating the non-linear functions with simple summation and comparison operations, the MS algorithm can get significant complexity reduction at the cost of obvious performance loss. By introducing the correction factor to decoding, the normalized MS (NMS) and offset MS (OMS) algorithms could offer a better balance between decoding complexity and performance [8].

This article targets the design of an area-efficient and high-performance 5G LDPC decoder. In general, 5G LDPC codes are built from a concatenation of a high-rate LDPC code and a low-density generator matrix (LDGM) code [9]. Since the variable-nodes (VNs) in the LDGM part are degree-1 VNs which can only receive one check-to-variable (CTV) message in each iteration, they are very sensitive to the reliability of the received CTV messages, and so to the choice of the correction factor. Therefore, in fixed-point implementations with low quantization bits where the precision of correction factor is limited, the OMS decoder suffers from severe performance degradation [10].

Many algorithms have been proposed in recent years [10]–[14] to improve the error-correction performance of 5G LDPC codes. By taking account the approximate-min* algorithm [15], the adjusted MS and generalized approximate-min* algorithms were proposed in [11] and [12], respectively. However, like the BP decoding, they suffer from relatively large implementation complexity due to the involved non-linear functions. In [13], the authors proposed a hybrid decoding algorithm in which the non-linear functions in the BP algorithm are simplified using the linear approximation. In [14], the offset and normalized factors are both introduced to decoding for a better calculation precision and their values vary during iterations which are optimized by machine learning. Despite the performance improvement, the main problem for these two methods is the numerous parameters, making the algorithm impractical in applications. Moreover, since all of the above algorithms are designed

Manuscript received July 28, 2020; revised October 20, 2020; accepted November 12, 2020. Date of publication December 1, 2020; date of current version January 12, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61604068, in part by the Fundamental Research Funds for the Central Universities under Grant 021014380065, in part by the Key Research Plan of Jiangsu Province of China under Grant BE2019003-4, and in part by the French ANR under Grant ANR-15-CE25-0006-01. This article was recommended by Associate Editor S. Gupta. *(Corresponding author: Zhongfeng Wang.)*

Hangxuan Cui, Jun Lin, and Zhongfeng Wang are with the School of Electronic Science and Engineering, Nanjing University, Nanjing 210008, China (e-mail: haxcui@163.com; jlin@nju.edu.cn; zfwang@nju.edu.cn).

Fakhreddine Ghaffari, Khoa Le, and David Declercq are with ETIS UMR 8051, CY Cergy Paris Université, ENSEA, CNRS, F-95000 Cergy, France (e-mail: fakhreddine.ghaffari@ensea.fr; khoa.letrung@ensea.fr; declercq@ensea.fr).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSI.2020.3038887.

Digital Object Identifier 10.1109/TCSI.2020.3038887

based on floating-point decoding, their performance cannot be guaranteed after being quantized.

Recently, the adapted MS (AMS) decoder [10] was proposed which targets fixed-point decoding. In 5G LDPC codes, the CNs connected to the degree-1 VNs are called extension checks and others are referred to core checks. Considering the fact that the degree-1 VNs are more likely to be erroneous when an imprecise offset factor is adopted, in the AMS decoder, the offset factor is only applied to core checks. Consequently, with low quantization bits, the AMS decoder could offer better performance than the MS and OMS decoders on 5G LDPC codes.

To further improve the performance of 5G LDPC decoders, this article introduces an improved AMS (IAMS) algorithm. Starting from reducing the error-probability of degree-1 VNs, a modified CN-update function is designed which considerably improves the reliability of CTV messages while maintaining the low-complexity property. Moreover, considering 5G LDPC codes are extremely irregular, a column degree adaptation strategy is proposed to manage the influence of the high-degree VNs on the decoding process. Simulation results on several 5G LDPC codes with different code rates and code lengths demonstrate that the proposed IAMS algorithm could offer an obvious performance improvement compared to existing ones, especially for codes with low to moderate code rates.

The implementation of LDPC decoders has been fully investigated [16]–[20]. In [17], the authors introduced a fully-parallel bit-parallel architecture with detailed optimizations for high-throughput applications. Since the complexity of the fully-parallel decoder is relatively high, the partially-parallel schedule, such as the layered schedule, has become the most popular one, which could use the up-to-date information from the current iteration, thereby doubling the speed of the decoding convergence. When the quasi-cyclic LDPC (QC-LDPC) codes are adopted, the CNs in the same block row of the base matrix are usually grouped into a single layer. In [18], an efficient reordered layered schedule was proposed to minimize the memory consumption. Moreover, to reduce the required number of iterations, the authors of [21] introduced a modified layered schedule for 5G LDPC codes, in which the processing order of layers is not sequential, but depends on the number of punctured edges and check-node degrees.

Though many works focus on the implementation of LDPC decoders, to the best of our knowledge, there is no prior work presenting the design for a whole 5G-LDPC decoder. In this article, we for the first time introduce an efficient 5G LDPC decoder architecture. In the proposed architecture, first, a layer merging technique benefitting from the orthogonal structure of 5G LDPC codes is proposed, which could reduce the number of clock cycles by 28.3%. By further incorporating with the proposed split storage method, the CTV memory consumption could be reduced by 39.6%. To alleviate the interconnection network overhead, we present the selective-shift structure and the message reordering methods, leading to obvious area and latency reduction. ASIC implementation results on 90-nm CMOS technology show that the proposed IAMS decoder could improve the throughput-to-area ratio (TAR) by 173.3% compared to the conventional design.

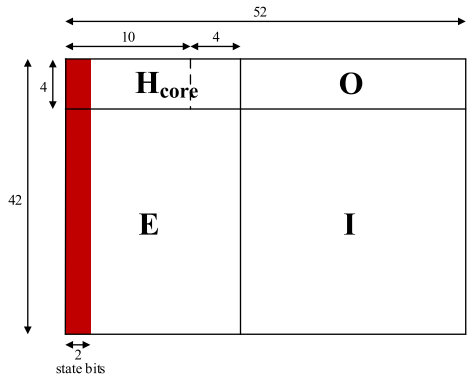


Fig. 1. The structure of base matrix BG2.

The remainder of this article is organized as follows. Section II gives some notations, as well as the preliminaries for 5G LDPC codes and fixed-point LDPC decodings. The proposed IAMS decoding algorithm is introduced in Section III. Numerical results and related discussions are provided in Section IV. Section V describes the proposed hardware architecture and Section VI presents the implementation results. Finally, Section VII concludes the paper.

II. Notations and Preliminaries

A. Notations

An LDPC code is specified by a sparse $M \times N$ parity check matrix $\mathbf{H}$, where M denotes the number of parity checks and N represents the number of code bits. The code rate $R = K/N = (N - M)/N$. LDPC codes can also be defined by bipartite Tanner graphs [22] which comprise a set of VNs and a set of CNs, corresponding to code bits and parity checks, respectively. Let $\mathcal{N}(m)$ denote the set of VNs that participate in the mth check. Similarly, the neighbors set of the nth VN is denoted by $\mathcal{M}(n)$. The number of neighbors connected with a VN is called column degree and with a CN is called row degree, denoted by d_v and d_c, respectively. An LDPC code is regular if the degrees of each set of nodes are the same, while degrees of an irregular LDPC code vary according to some degree distributions. QC-LDPC codes have a structured $\mathbf{H}$ matrix that can be generated from an $M_b \times N_b$ base matrix $\mathbf{H}_B$. Each nonzero entry of $\mathbf{H}_B$ can be expanded by circularly shifting a $Z \times Z$ identity matrix and each zero entry represents a $Z \times Z$ all-zero matrix, where Z denotes the expansion factor.

B. 5G LDPC Codes

To support a broad range of code lengths and rates, two rate-compatible base graphs, BG1 and BG2, are designed for 5G LDPC codes. These two base graphs have a similar structure while BG1 is targeted for larger information lengths ($500 \leq K \leq 8448$) and higher rates ($1/3 \leq R \leq 8/9$) and BG2 is targeted for smaller information lengths ($40 \leq K \leq 2560$) and lower rates ($1/5 \leq R \leq 2/3$). Fig. 1 shows the structure of base matrix BG2 which has 42 rows and 52 columns. The sub-matrix $\mathbf{H}_{\text{core}}$ is called the core part and the other three sub-matrices form the extension part. In both BG1 and BG2 matrices, $\mathbf{H}_{\text{core}}$ consists of the first four rows of the base

matrix and adopts a dual diagonal structure for parity bits to simplify the encoding process. The extension part has an equal amount of VNs and CNs, and all extension VNs are degree-1 nodes. **O** denotes an all-zero matrix and **I** denotes an identity matrix. The core checks usually have a higher row degree than the extension checks. The leftmost two columns of the base matrix correspond to the punctured bits, also known as the state bits. One important feature for 5G LDPC codes is that they are extremely irregular, which means there exists a significant difference in row degrees and column degrees. For instance, in base matrix BG2, d_v varies from 1 to 23 and d_c varies from 3 to 10.

C. Fixed-Point LDPC Decodings

Assume an LDPC codeword $\boldsymbol{c} = \{c_0, c_1, \ldots, c_{N-1}\}$ is transmitted over the additive white Gaussian noise (AWGN) channel using the binary phase shift keying (BPSK) modulation, the received vector $\boldsymbol{y}$ is

$$y_i = x_i + n_i, \quad n_i \sim \mathcal{N}(0, \sigma^2), \quad i = 0, 1, \ldots, N-1, \quad (1)$$

where $x_i = 1 - 2c_i$ and n_i is a Gaussian random variable with zero mean and variance σ^2. In fixed-point implementations, the quantized version of $\boldsymbol{y}$, denoted by $\boldsymbol{\gamma}$, is typically input to the decoders. Let Γ represent the input alphabet comprising of integers, and then we have $\gamma_i = [\mu \cdot y_i]_\Gamma$ where $\mu > 0$ is a constant referred to as the gain factor. $[x]_\Gamma$ returns the closest integer to x that belongs to Γ. Assume the input messages are expressed by q bits, we have $\Gamma = \{-Q, \ldots, -1, 0, 1, \ldots, +Q\}$ where $Q = 2^{q-1} - 1$. Actually, $\mu = 2^q$ means that all channel LLR values are shifted q bits to the left and then rounded to integers, which is the same as the usual quantization method when q fraction bits are preserved. Moreover, the introduced quantization method is more flexible because the values of μ could be optimized to other values besides 2^q for better decoding performance [23].

Let $\alpha_{m,n}$ and $\beta_{n,m}$ denote the messages passed from the mth CN to the nth VN and from the nth VN to the mth CN, respectively. $\tilde{\boldsymbol{\gamma}}$ denotes the a-posteriori-probability (APP) vector. The exchanged messages $\alpha_{m,n}$ and $\beta_{n,m}$ are quantized to q bits. Since the APP messages are generally larger than the input and exchanged messages, $\tilde{\gamma}_n$ is quantized to $\tilde{q}$ bits where $\tilde{q} > q$ to avoid clipping. $\mathcal{A} = \{-\tilde{Q}, \ldots, -1, 0, 1, \ldots, +\tilde{Q}\}$ denotes the alphabet for $\tilde{\boldsymbol{\gamma}}$ where $\tilde{Q} = 2^{\tilde{q}-1} - 1$.

The decoding process of the layered schedule is described as follows.

1) Initialization:

Assign the values of the input vector $\boldsymbol{\gamma}$ to the APP vector $\tilde{\boldsymbol{\gamma}}$. Moreover, all CTV messages $\alpha_{m,n}$ are initialized with zeros.

2) Iterative Process:

In the layered schedule, each iteration comprises several decoding layers. The decoding is executed layer by layer and each layer has three steps.

Step 1 (VN update): In the tth iteration, the variable-to-check (VTC) message $\beta_{n,m}^{(t)}$ is calculated by

$$\beta_{n,m}^{(t)} = [\tilde{\beta}_{n,m}^{(t)}]_\Gamma = [\tilde{\gamma}_n^{(t)} - \alpha_{m,n}^{(t-1)}]_\Gamma. \quad (2)$$

Step 2 (CN update): In the BP decoding, the CTV message is given by

$$\alpha_{m,n}^{(t)} = \tau_{m,n}^{(t)} \cdot \phi\left(\sum_{n' \in \mathcal{N}(m)\backslash n} \phi(|\beta_{n',m}^{(t)}|)\right), \quad (3)$$

where $\tau_{m,n}^{(t)} = \prod_{n' \in \mathcal{N}(m)\backslash n} \operatorname{sgn}(\beta_{n',m}^{(t)})$ and $\phi(x) = -\log[\tanh(x/2)]$. Considering $\phi^{-1}(x) = \phi(x)$ and the magnitude of $\alpha_{m,n}^{(t)}$ is dominated by the minimum input $|\beta_{n',m}^{(t)}|$ [24], the MS simplifies (3) according to

$$\begin{aligned} \alpha_{m,n}^{(t)} &\simeq \tau_{m,n}^{(t)} \cdot \phi\left(\phi\Big(\min_{n' \in \mathcal{N}(m)\backslash n} |\beta_{n',m}^{(t)}|\Big)\right) \\ &= \tau_{m,n}^{(t)} \cdot \min_{n' \in \mathcal{N}(m)\backslash n} |\beta_{n',m}^{(t)}|. \end{aligned} \quad (4)$$

Since $\phi(|\beta_{n',m}^{(t)}|) > 0$, we have $\phi(\min_{n' \in \mathcal{N}(m)\backslash n} |\beta_{n',m}^{(t)}|) < \sum_{n' \in \mathcal{N}(m)\backslash n} \phi(|\beta_{n',m}^{(t)}|)$. Moreover, because $\phi(x)$ is a decreasing function, it can be deduced from (3) and (4) that the MS decoding overestimates the magnitudes of CTV messages compared to the BP decoding, leading to the performance degradation [11]. To alleviate the overestimation, an offset factor is included in the OMS decoding, as shown in (5).

$$\alpha_{m,n}^{(t)} = \tau_{m,n}^{(t)} \cdot \max\left(\min_{n' \in \mathcal{N}(m)\backslash n} |\beta_{n',m}^{(t)}| - \lambda, 0\right), \quad (5)$$

where λ denotes the offset factor. In fixed-point implementations, λ is typically fixed to 1, which is the least significant bit (LSB) under the integer representation [20].

The only difference between the AMS and OMS algorithms is the CN processing procedure. To reduce the error probability of degree-1 VN, the AMS decoder processes the core checks and extension checks differently using different offset factors [10], as shown in (6). For the core checks, λ is set to 1 to obtain the gain from the offset principle while λ is set to 0 for the extension checks to reduce the offset effect on these VNs.

$$\alpha_{m,n}^{(t)} = \begin{cases} \text{applying (5)}, & \text{for the core checks} \\ \text{applying (4)}, & \text{for the extension checks.} \end{cases} \quad (6)$$

Step 3 (APP update): In order to achieve better precision, $\tilde{\beta}_{n,m}^{(t)}$ in (2) is used to update APP values according to

$$\tilde{\gamma}_n^{(t)} = [\alpha_{m,n}^{(t)} + \tilde{\beta}_{n,m}^{(t)}]_\mathcal{A}, \quad (7)$$

where function $[\cdot]_\mathcal{A}$ is applied to ensure the updated APP values are taken from alphabet $\mathcal{A}$.

After all layers have been processed, the tentative codeword $\hat{\boldsymbol{c}}^{(t)}$ can be obtained by applying the hard-decision to vector $\tilde{\boldsymbol{\gamma}}^{(t)}$ according to

$$\hat{c}_n^{(t)} = HD(\tilde{\gamma}_n^{(t)}) = \begin{cases} 0, & \tilde{\gamma}_n^{(t)} \geq 0 \\ 1, & \tilde{\gamma}_n^{(t)} < 0. \end{cases}$$

The decoding stops when all parity check equations are satisfied or the maximum number of iterations It_{max} is reached.

III. The Proposed IAMS Decoding Algorithm

A. The Modified CN-Update Function

As mentioned above, all extension VNs in 5G LDPC codes are with degree-1 and each is connected to a unique CN. Consequently, these VNs only receive one CTV message in each iteration so they are sensitive to the reliability of CTV messages and the choice of offset factor. In fixed-point implementations, the offset factor is generally not optimal so the reliability of CTV messages is limited due to the limited bit representation of messages, which is the main reason for the severe performance degradation appearing in fixed-point OMS decoder. In order to improve the performance of 5G LDPC decoders, we propose a new CN-update function in this subsection to improve the reliability of CTV messages, and thus efficiently benefits the performance improvement of 5G LDPC decoders.

Denote the first and second minimum magnitudes of the input VTC messages in a CN by min_1 and min_2, respectively. In order to maintain the low computation complexity, we only use these two values which are available in conventional MS decoder to design a new CN-update function. Let idx_1 and idx_2 be the indices of VNs corresponding to min_1 and min_2, respectively. $I(m)$ is defined as $I(m) = \{idx_1, idx_2\}$ and $\bar{I}(m) = \mathcal{N}(m) \setminus I(m)$. Observing (3) we notice that, for $n \in \bar{I}(m)$, both min_1 and min_2 are extrinsic VTC messages that are used to calculate the CTV message $\alpha_{m,n}^{(t)}$. Since the magnitude of $\alpha_{m,n}^{(t)}$ is dominated by the minimum magnitude of extrinsic VTC messages, a sufficient precision can be achieved if the first and second minimum magnitudes of the extrinsic VTC messages are both employed to approximate the CN-update function of the BP algorithm. Therefore, for $n \in \bar{I}(m)$, we approximate the CN-update function shown in (3) to

$$\alpha_{m,n}^{(t)} = \tau_{m,n}^{(t)} \cdot \phi\Big(\phi(min_1) + \phi(min_2)\Big). \tag{8}$$

It can be seen that the overestimation of the CTV messages appearing in the MS algorithm could be alleviated by using (8) since more extrinsic VTC messages are included. Based on the approximate-min* algorithm proposed in [15], (8) can also be written as

$$\alpha_{m,n}^{(t)} = \tau_{m,n}^{(t)} \cdot (min_1 \boxplus min_2), \tag{9}$$

where $x \boxplus y = \min(|x|, |y|) - \log \frac{1+e^{-|x-y|}}{1+e^{-|x+y|}}$. In fact, (9) can be viewed as the MS decoding with an offset factor which is inherently optimized by the BP decoding. For simplicity, let a represent min_1 so $min_2 = a + \Delta$. Therefore, the offset factor λ in (9) is

$$\lambda = \log \frac{1 + e^{-\Delta}}{1 + e^{-(2a+\Delta)}}.$$

Since min_1 and min_2 are both non-negative integers in fixed-point implementations, a and Δ are also non-negative integers. Therefore, we can conclude that $\lambda \geq 0$ so the quantized version of λ is

$$\lambda = \lfloor \log \frac{1 + e^{-\Delta}}{1 + e^{-(2a+\Delta)}} + \frac{1}{2} \rfloor.$$

Property: The offset factor λ will be 1 only when min_1 and min_2 are both strictly positive and equal. Otherwise, $\lambda = 0$.

Proof: To prove this property, we consider three cases.

Case 1: $min_1 = 0$. In this case, $a = 0$. Then,

$$\lambda = \lfloor \log \frac{1 + e^{-\Delta}}{1 + e^{-\Delta}} + \frac{1}{2} \rfloor = 0.$$

Case 2: $min_1 > 0$ and $min_1 \neq min_2$. In this case, $a \geq 1$ and $\Delta \geq 1$. Therefore,

$$\begin{aligned} &\log \frac{1 + e^{-\Delta}}{1 + e^{-(2a+\Delta)}} \\ &\leq \log \frac{1 + e^{-1}}{1 + e^{-(2a+1)}} < \log(1 + e^{-1}) \\ &\Rightarrow \log \frac{1 + e^{-\Delta}}{1 + e^{-(2a+\Delta)}} + \frac{1}{2} < \log(1 + e^{-1}) + \frac{1}{2} < 0.7133 \\ &\Rightarrow \lambda = 0. \end{aligned}$$

Case 3: $min_1 > 0$ and $min_1 = min_2$. In this case, $a \geq 1$ and $\Delta = 0$. Let $\Delta = 0$ and we have

$$\begin{aligned} \log \frac{1 + e^{-\Delta}}{1 + e^{-(2a+\Delta)}} &= \log \frac{2}{1 + e^{-2a}} < \log 2 \\ &\Rightarrow \log \frac{1 + e^{-\Delta}}{1 + e^{-(2a+\Delta)}} + \frac{1}{2} < 1.1931 \\ &\Rightarrow \lambda < 2. \end{aligned}$$

Also,

$$\begin{aligned} \log \frac{1 + e^{-\Delta}}{1 + e^{-(2a+\Delta)}} &= \log \frac{2}{1 + e^{-2a}} \geq \frac{2}{1 + e^{-2}} \\ &\Rightarrow \log \frac{1 + e^{-\Delta}}{1 + e^{-(2a+\Delta)}} + \frac{1}{2} \geq 1.0662 \\ &\Rightarrow \lambda \geq 1. \end{aligned}$$

Therefore, we have $\lambda = 1$. ■

Based on this property, the offset factor for $n \in \bar{I}(m)$ can be determined according to min_1 and min_2. For $n \in I(m)$, we cannot obtain a more precise correction factor only based on min_1 and min_2. Since MS decoder performs better than OMS decoder on 5G LDPC codes in fixed-point implementations [10], λ is set to 0 for $n \in I(m)$. The proposed CN-update function is shown in (10), which still remains the low-complexity property.

$$\alpha_{m,n}^{(t)} = \begin{cases} \tau_{m,n}^{(t)} \cdot min_2, & n = idx_1 \\ \tau_{m,n}^{(t)} \cdot min_1, & n = idx_2 \\ \tau_{m,n}^{(t)} \cdot \max(min_1 - 1, 0), & n \in \bar{I}(m) \ \& \ \Delta = 0 \\ \tau_{m,n}^{(t)} \cdot min_1, & n \in \bar{I}(m) \ \& \ \Delta \neq 0. \end{cases} \tag{10}$$

To demonstrate the effectiveness of the proposed CN-update function, the mismatch probabilities of different CN-update functions are shown in Fig. 2, where the exchanged messages are quantized to 4 bits, *i.e.,* $q = 4$. Therefore, the values of $|\beta_{n,m}|$ can only be $0 \sim 7$ so the total number of combinations of the received messages in a degree-d_c CN is 8^{d_c} ($2^{(q-1)\cdot d_c}$). For each case, if the CTV value calculated by the tested decoder is not equal to the CTV value calculated by the 4-bit quantized BP decoder, we consider this case as a mismatch case. The mismatch probability is obtained by testing all 8^{d_c} cases and then calculating the proportion of mismatch cases.

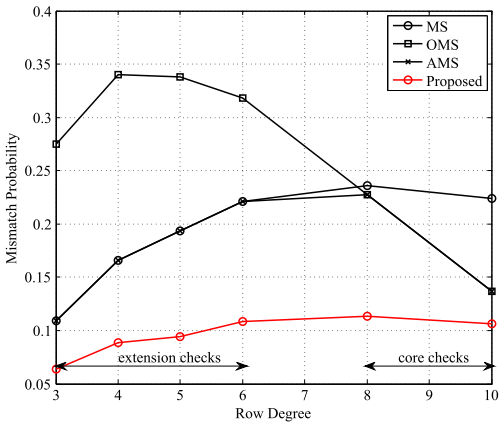


Fig. 2. The mismatch probabilities of different CN-update functions under the row degree region for BG2 codes.

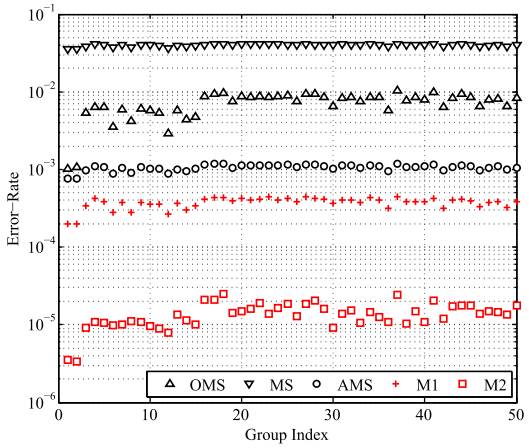


Fig. 3. The error-rate of each group when $E_b/N_0 = 2.0$dB.

From Fig. 2 we can see that, compared to the MS, OMS, and AMS decodings, the proposed CN-update function shows a much lower mismatch probability in the simulated row degree region, which is also the row degree region for BG2 codes. Therefore, the reliability of CTV messages is significantly improved, especially for the extension checks. It can also be seen that the OMS decoder shows a higher mismatch probability for the extension checks while a lower probability for the core checks compared with the MS decoder. The AMS decoder [10] combines the advantages of the MS and OMS decoders, which explains its performance improvement.

B. Column Degree Adaptation

As stated before, 5G LDPC codes are extremely irregular and there exists a wide variation in column degrees. In base matrix BG2, the column degree varies from 1 to 23 and from 1 to 30 in BG1. With more neighbor CNs, the high degree VNs usually have larger APP magnitudes, which are called strong messages. These strong messages can be helpful or harmful to the decoding process, depending on whether they are correct or not. In the waterfall region where many bits are received incorrectly, the incorrect strong messages tend to negatively influence the correction of the received bits. In the error-floor region where the channel conditions are good and trapping-sets dominate the decoding performance [25], the correct strong messages can overcome the incorrect messages in trapping-sets and thus contribute to improving the decoding performance [26]. Therefore, the requirement of strong messages is different in different SNR regions.

In order to manage the influence of strong messages on the decoding process, we propose a column degree adaptation strategy in which the CTV messages passed to different VNs from a CN are computed non-uniformly. Observing (5) and (10) we can conclude that the magnitudes of CTV messages computed by the OMS decoding are generally smaller than those by the proposed CN-update function. To limit the magnitudes' growth of strong messages, the CTV messages transmitted to the VNs whose degrees are larger than threshold D is computed using the CN-update function of the OMS decoding rather than the proposed CN-update function. To avoid over-correction to strong messages, the column degree adaptation is only applied to core checks, whose CTV messages show a lower mismatch probability than those of extension checks when applying the OMS decoding, as shown in Fig. 2. Consequently, the influence of strong messages to the decoding process could be managed to some extent by adjusting parameter D and the decoding performance could get a better balance in the waterfall and error-floor regions. Moreover, one can select a proper D to get the best performance in the required SNR region.

The effectiveness of the column degree adaptation is illustrated in Fig. 3. In this work, we divide the codeword into N_b groups and each group corresponds to a column in base matrix $\mathbf{H}_B$. Consequently, a group consists of Z bits and the bits in each group have the same column degree. In simulations, a group is considered as erroneous if there exists an error bit in the group. Since 5G LDPC codes are extremely irregular, the degrees of bits are very different, so the bits with different column degrees may perform differently under the same decoding algorithm. Considering the bits in different groups may have different column degrees, Fig. 3 shows the error-rate of each group when $E_b/N_0 = 2.0$dB. The $R = 1/5$, $Z = 52$, $N = 2600$ 5G LDPC code defined by BG2 is applied and all decodings are quantized with parameters $(q, \tilde{q}) = (4, 6)$. For each decoding, at least 1000 error frames are collected. We denote the decoding where only the proposed CN-update function is applied as $M1$ and the decoding where both the proposed CN-update function and column degree adaptation are applied as $M2$, namely the IAMS algorithm. The parameter D is selected by traversing all row degrees of the code to find the value which shows the optimal performance through Monte-Carlo simulations. For the selected code, $D = 6$.

Considering the degrees of bits in the first two groups are much larger than others, these bits have more chances to be corrected so the first two groups show the best performance, especially for the OMS and $M2$ decoders. Since Fig. 3 shows the simulation results in the low SNR region where many bits are received incorrectly, the propagation of incorrect strong messages has larger negative influence to decoding than the imprecise offset factor. Therefore, the OMS decoder performs better than the MS decoder. However, they both perform worse than the AMS decoder [10], which is the state-of-the-art one

for 5G LDPC codes in fixed-point. Moreover, it can be seen that for all groups, the $M1$ and $M2$ decoders exhibit better performance than the AMS decoder. With the help of the proposed column degree adaptation, $M2$ significantly improves the decoding performance of the $M1$ decoder, proving the effectiveness of the proposed column degree adaptation.

The detailed decoding process of the proposed IAMS algorithm is shown in Alg. 1, where the layered schedule is adopted and each layer corresponds to one row of the base matrix. When $l < 4$, the core checks are processed with the proposed column degree adaption applied. The number of layers is denoted by L and $\mathcal{L}_l$ denotes the indices set of rows in the l-th layer.

Algorithm 1: The Proposed IAMS Decoding Algorithm

input : $\boldsymbol{\gamma} = (\gamma_0, \gamma_1, \cdots, \gamma_{N-1})$
initialize: $\forall m \in [0, M), n \in [0, N) : \alpha_{m,n}^{(0)} = 0$,
$\forall n \in [0, N) : \tilde{\gamma}_n^{(0)} = \gamma_n$
for $t = 1$ **to** It_{max} **do**
  for $l = 0$ **to** $L - 1$ **do**
    for $m \in \mathcal{L}_l$ *and* $n \in \mathcal{N}_m$ **do**
      $\beta_{n,m}^{(t)} = [\tilde{\beta}_{n,m}^{(t)}]_\Gamma = [\tilde{\gamma}_n^{(t-1)} - \alpha_{m,n}^{(t-1)}]_\Gamma$
    for $m \in \mathcal{L}_l$ *and* $n \in \mathcal{N}_m$ **do**
      if $l < 4$ *and* $d_v^n \geq D$ **then**
        Calculate $\alpha_{m,n}^{(t)}$ using (5)
      else
        Calculate $\alpha_{m,n}^{(t)}$ using (10)
    for $m \in \mathcal{L}_l$ *and* $n \in \mathcal{N}_m$ **do**
      $\tilde{\gamma}_n^{(t)} = [\alpha_{m,n}^{(t)} + \tilde{\beta}_{n,m}^{(t)}]_\mathcal{A}$
  for $n = 0$ **to** $N - 1$ **do**
    $\hat{c}_n^{(t)} = HD(\tilde{\gamma}_n^{(t)})$
  if $\hat{\boldsymbol{c}}^{(t)} \cdot \boldsymbol{H}^T = \boldsymbol{0}$ **then**
    break
output : $\hat{\boldsymbol{c}}^{(t)}$

IV. Numerical Results

In this section, the decoding performance of the proposed IAMS algorithm is illustrated and compared to the MS, OMS, and AMS decodings. All decodings take the layered schedule. In practical applications, the number of quantization bits used in LDPC decoders is usually no more than 6 in order to reduce the area and power consumption. Therefore, the quantization parameters are set to $(q, \tilde{q}) = (4, 6)$ in this work. Moreover, the performance of the floating-point MS and OMS algorithms are shown for reference, which also take the layered schedule. The offset value for the floating-point OMS decodings is set to 0.2. The simulation results are obtained through Monte-Carlo simulations that generate at least 100 error frames for each plotted point. Because the fraction of degree-1 bits is very small in high code rate 5G LDPC codes while our approach targets for improving the performance regarding the degree-1 bits, the proposed decoder is more suitable for the low to moderate code rates 5G LDPC codes. Therefore, we consider two 5G LDPC codes with different rates and lengths: a $R = 1/5$, $Z = 52$, $N = 2600$ BG2 code and a $R = 2/3$, $Z = 104$, $N = 3432$ BG1 code. For simplicity, it is assumed that the codeword is sent only once without using any hybrid automatic repeat request (HARQ) scheme.

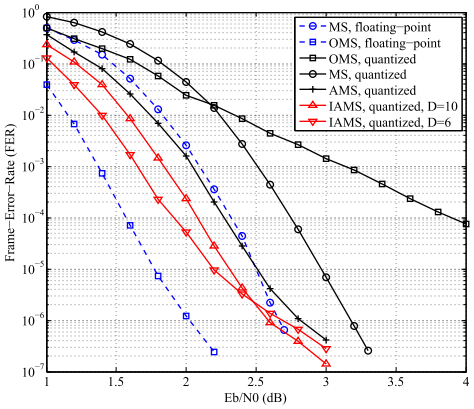


Fig. 4. Simulation results on the $R = 1/5$, $Z = 52$, $N = 2600$ BG2 code when $It_{max} = 15$.

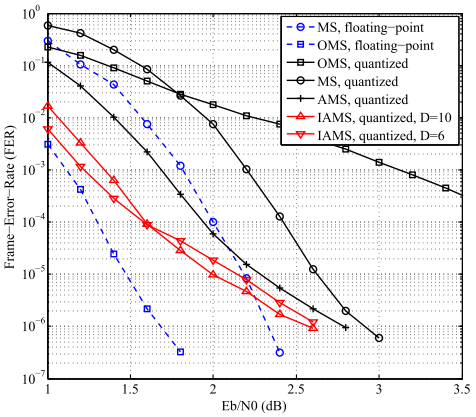


Fig. 5. Simulation results on the $R = 1/5$, $Z = 52$, $N = 2600$ BG2 code when $It_{max} = 100$.

A. Performance Comparisons

Since the maximum number of iterations is typically less than 20 in practical implementations considering the throughput requirement while the decoders need about 100 iterations to be saturated, Fig. 4 and Fig. 5 show the simulation results on the $R = 1/5$, $Z = 52$, $N = 2600$ BG2 code when $It_{max} = 15$ and $It_{max} = 100$, respectively. For a fair comparison, the channel gain factors for each decoding are fixed and optimized by simulations to find the value which performs best when $FER = 10^{-7}$, where the test step is set to 0.05. The optimal values for the OMS, MS, AMS, and IAMS decoders are 1.3, 1.1, 0.85, and 0.8, respectively. Due to the imprecise offset factor, the OMS decoder suffers from severe performance degradation under (4,6) quantization, which could be compensated by increasing one bit of quantization length. Compared to the AMS decoding, the proposed IAMS decoding shows a much better performance. When the threshold D is well-selected, the performance gain could be 0.4dB in the

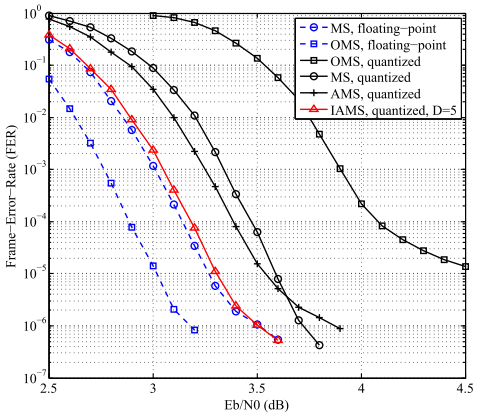


Fig. 6. Simulation results on the $R = 2/3$, $Z = 104$, $N = 3432$ BG1 code when $It_{max} = 15$.

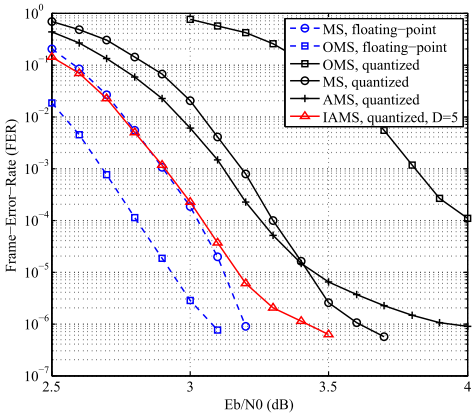


Fig. 7. Simulation results on the $R = 2/3$, $Z = 104$, $N = 3432$ BG1 code when $It_{max} = 100$.

waterfall region and 0.2dB in the error-floor region. However, a limitation of the IAMS decoding is that the error floor starts around $FER = 10^{-5}$. It can be explained by the fact that due to the quantization, the dynamic range of messages is limited so they are hard to escape from trapping-sets [26]. By increasing the number of quantization bits, the error-floor phenomenon can be overcome to some extent. It should be noted that though the IAMS decoding suffers from the error-floor phenomenon, its performance is still better than those of other fixed-point decodings in high-SNR region.

To further verify the comparison results, Fig. 6 and Fig. 7 show the simulation results on the $R = 2/3$, $Z = 104$, $N = 3432$ BG1 code, where the threshold D is set to 5. The optimal values of the channel gain factor for the OMS, MS, AMS, and IAMS decoders are 3.1, 2.6, 2.8, and 2.55, respectively. As can be seen, the IAMS algorithm shows the best error-correction performance among all fixed-point decodings. Compared to the AMS decoding, the IAMS decoding could offer 0.4dB to 0.6dB performance gain. Therefore, we can conclude that for 5G LDPC codes with low to moderate rates, there are many extension bits that could benefit from the proposed CN-update function so the proposed IAMS decoding could offer a much-improved error-correction performance compared to the existing fixed-point decodings.

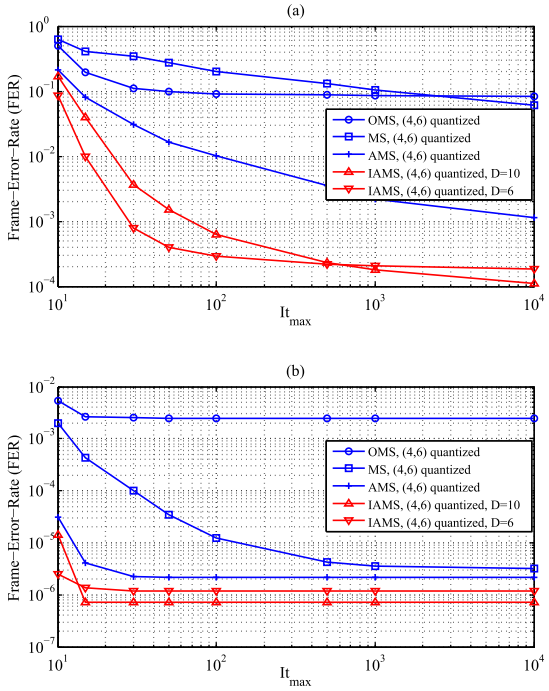


Fig. 8. FER performance under different iteration limits at (a): $E_b/N_0 = 2.0$dB, (b): $E_b/N_0 = 2.6$dB.

B. Decoding Performance Analysis

In Fig. 8, we explore the effect of limiting the maximum number of iterations on different decodings, where It_{max} increases from 10 to 10K. The $R = 1/5$, $Z = 52$, $N = 2600$ BG2 code is adopted and two E_b/N_0 points, 2.0dB and 2.6dB, are considered. Fig. 8(a) shows the performance in the waterfall region where the random-like errors are main causes of decoding failures [26]. As shown in Fig. 8(a), by increasing the maximum number of iterations, most of these errors can be corrected and the decoding performance is improved. Since a smaller D has better capability to limit the magnitudes growth of strong random-like errors, when It_{max} is not sufficient, the IAMS decoding paired with $D = 6$ has a better performance than with $D = 10$. Also, because the overestimation of CTV messages encourages the magnitude growth of errors, the MS decoding shows poor performance and converges slowly at this point. When E_b/N_0 is sufficiently large, the decoding performance is dominated by trapping-sets, which are main reasons for the error-floor phenomenon [27]. Fig. 8(b) shows the performance under different iteration limits in the error-floor region. As can be seen, except the MS decoding, almost no performance gain can be further obtained by increasing the maximum number of iterations when $It_{max} > 30$. As for the MS decoding, the saturation starts from $It_{max} = 500$. Moreover, the IAMS decoding paired with $D = 10$ can surpass that with $D = 6$ in performance within a smaller number of iterations.

Since the degree-1 VNs in 5G LDPC codes are prone to be erroneous, these VNs are easily to form a trapping-set. To analyze the performance behavior of IAMS decoding in the error-floor region, we collect a typical set of trapping-sets for

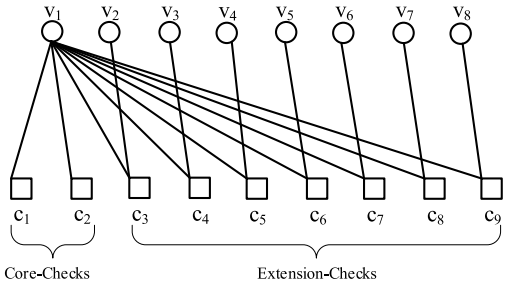


Fig. 9. Subgraph induced by the collected (8,2) trapping-set.

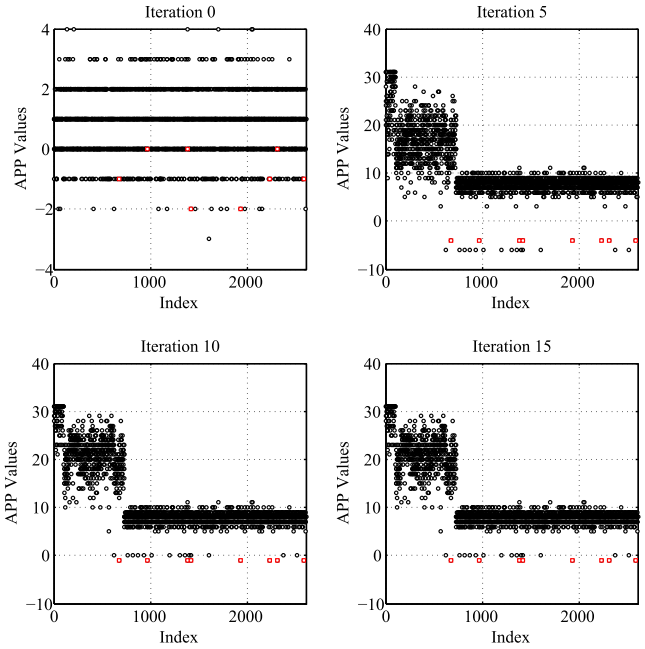


Fig. 10. Soft-decision values evolution along with iterations.

the selected code, as shown in Fig. 9. In order to facilitate the analysis, we assume only eight received bits are erroneous and all fall into this trapping-set. In this case, v_2 to v_8 are falsely estimated so v_1 receives seven wrong messages from c_3 to c_9, which are all extension checks. Moreover, v_1 receives two correct messages from c_1 and c_2, which belong to core checks. When the summation of the two correct messages is smaller than that of seven wrong messages, v_1 will be erroneous and the remaining seven bits cannot be corrected. Consequently, the decoder will be trapped in the trapping-set.

The trapping process can be shown with a practical example. Fig. 10 shows the APP value evolution when the IAMS algorithm with $D = 6$ is applied. The bits belonging to the collected trapping-set are marked with red squares and others with black circles. Assume the all-zero codeword is transmitted using the BPSK modulation. Accordingly, non-negative APP values are interpreted as correct and negative values denote faults. As can be seen, the decoder cannot escape from the trapping-set once it is captured. By increasing the value of D, more core checks could be processed by the proposed CN-update function rather than the OMS decoding. Therefore, the magnitudes of CTV messages generated from

TABLE I

THE VALUES OF RECEIVED CTV MESSAGES FOR A VN BELONGING TO TRAPPING-SET.

	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9
$D = 6$	0	6	0	0	−2	−2	−1	0	−1
$D = 10$	0	7	0	0	−2	−2	−1	0	−1

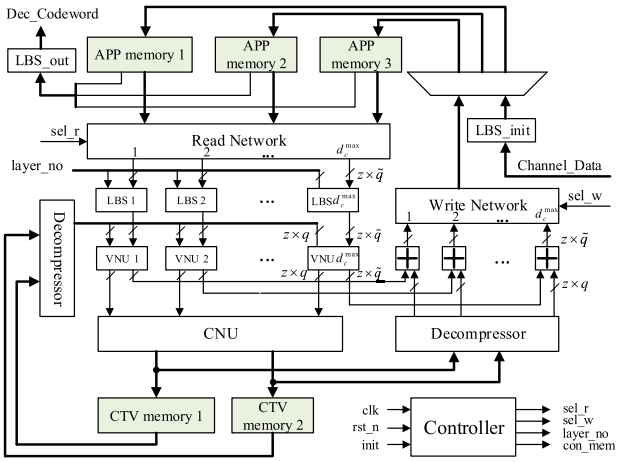


Fig. 11. Top-level architecture of the proposed 5G LDPC decoder.

the core checks (c_1 and c_2) can be increased to some extent. In that case, the probability that v_1 could be corrected is increased so the decoder could have a larger probability to escape from trapping-sets.

To illustrate the above discussion intuitively, Table I shows the values of received CTV messages of a VN that belongs to trapping-set in the 6th iteration. The channel message for this VN equals -1. Compared to the case when $D = 6$, the second connected CN c_2 sends a slightly larger correct CTV message to this VN when $D = 10$ and thus, the corresponding bit could be correctly recovered. Consequently, this codeword can be successfully decoded when $D = 10$. This explains why the IAMS decoding could perform better when paired with $D = 10$ in the error-floor region. However, in order to balance the decoding performance in the waterfall region, D is not the larger the better. Also, the correction of random-like errors in early iterations will be damaged by an excessive D.

V. HARDWARE ARCHITECTURE FOR 5G LDPC DECODERS

In this section, we propose an efficient architecture to implement 5G LDPC decoders. In order to design a high-throughput and area-efficient decoder, several optimization methods are developed, as shown in the following subsections.

A. Top-Level Architecture

The overall architecture of the proposed 5G LDPC decoder is shown in Fig. 11, which is implemented using the layered schedule. For convenience, we assume that the quantization version of channel LLR messages is available in the input port of the decoder. However, it should be noted that the method to quantize channel LLR messages should be compatible with the quantization method used in the decoder. The proposed architecture is not limited to a specific quantization method

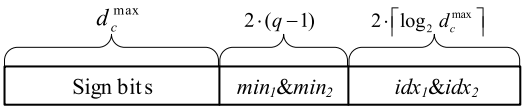


Fig. 12. Compressed format of CTV messages.

so one can easily modify the proposed decoder architecture to support different quantization schemes when the number of quantization bits remains unchanged.

For a QC-LDPC code defined by an $M_b \times N_b$ base matrix $\mathbf{H}_B$, the number of decoding layers L usually equals M_b. Therefore, the parallelism degree of decoder equals the expansion factor Z. As shown in Fig. 11, the input and exchanged messages are quantized to q bits and the APP values are quantized to $\tilde{q}$ bits. All control signals are generated by the *Controller*. Two memory blocks, namely APP memory and CTV memory, are used to store the APP and CTV messages, respectively. The CTV memory is implemented with the dual-port random access memory (DP-RAM) to support simultaneous read and write operations. In order to allow massively parallel read, write, and initiate operations, the APP memory is implemented with registers. In the proposed architecture, the APP memory is divided into three parts and the CTV memory consists of two parts. The reason for this configuration will be presented in the following subsections.

In each decoding layer, APP messages are read from the APP memory first and then passed to the *Read Network*, which rearranges and selects these messages according to the current processing layer to ensure they will be processed by the proper left barrel shifters (LBSs) and VN unit (VNUs). Similarly, the *Write Network* is used to rearrange the updated APP messages to ensure they can be stored in the correct addresses of APP memory. Let d_c^{max} denote the maximum row degree of the code. In the proposed architecture, d_c^{max} pairs of LBSs and VNUs are applied. Messages output from the *Read Network* should be left rotated first by LBSs according to the corresponding shift factors and then passed to VNUs to calculate the VTC messages. By adopting the method to generate the shift factor presented in [20], the data write-back barrel shifters can be eliminated.

After being saturated to q bits, the VTC messages are sent to the CN unit (CNU) which generates CTV messages. The CNU is implemented using the area-efficient architecture proposed in [28]. In the IAMS decoder, idx_2 should also be calculated and stored, which is the main difference with other decoders. As shown in Fig. 12, CTV messages are stored in a compressed format to reduce the memory cost. Therefore, the width of the CTV memory is $z \times (d_c^{max} + 2 \cdot (q - 1 + \lceil \log_2 d_c^{max} \rceil))$. Since the CTV messages in all layers need to be stored, the depth of the CTV memory is L. In order to convert the CTV messages from the compressed format to the uncompressed format, two *De-compressors* are inserted into the decoder which generate the final CTV messages for the following calculations. Then, the APP values can be updated. After writing them back to the APP memory, one decoding layer is finished.

To minimize the number of clock cycles, no pipeline is inserted into the proposed architecture. Therefore, one decoding layer takes one clock cycle and the total number of clock cycles is $L \times It_{max}$. The throughput θ is computed as

$$\theta = \frac{f \times N}{L \times It_{max}}, \tag{11}$$

where f denotes the frequency of the decoder.

B. Memory and Clock Cycles Reduction

By observing the structure of BG1 and BG2 matrices, it can be found that part of them has the orthogonality property, meaning no VN is connected to two consecutive layers. For instance, the 21st to 46th rows of the BG1 matrix are orthogonal. Similarly, the 21st to 42nd rows of the BG2 matrix are orthogonal. In two orthogonal layers, the APP messages updated in the previous layer will not be used in the next layer. Therefore, the decoding processes in such two layers are independent. Based on this feature of 5G LDPC codes, we propose a layer merging method to reduce the number of clock cycles. A similar idea was also applied in [29] to optimize a pipelined decoder for IEEE 802.11ad standard. However, the configurations in these two architectures are different.

In the proposed architecture, two consecutive layers in the orthogonal part are processed simultaneously. Therefore, the number of decoding layers in the orthogonal part is reduced by half, which leads to fewer clock cycles. For BG1 and BG2 codes, the number of clock cycles could be reduced by 28.3% and 26.2%, respectively. Because row degrees in the orthogonal part are all less than $d_c^{max}/2$, no additional LBS or VNU is needed and the APP memory remains unchanged. However, since two layers are processed in one clock cycle, the CNU and CTV memory should be modified to make generating and storing two sets of CTV messages at the same time feasible. Fig. 13 shows the architecture of CNU, which is divided into two subunits. When two orthogonal layers are processed simultaneously, two sets of VTC messages are input to $\text{CNU}^{1\text{st}}$ and $\text{CNU}^{2\text{nd}}$, respectively. In this case, the Compare & Select unit is disabled so two sets of CTV messages are output from the CNU. Let d_c^o denote the maximum row degree in the orthogonal part. In order to store two sets of CTV messages in the same address, the width of CTV memory is set to

$$W = \max\{z \times (d_c^{max} + 2 \cdot (q - 1 + \lceil \log_2 d_c^{max} \rceil)), \\ 2 \times z \times (d_c^o + 2 \cdot (q - 1 + \lceil \log_2 d_c^o \rceil))\}. \tag{12}$$

Table II shows the size of CTV memory when $q = 4$. As can be seen, though the width of CTV memory is slightly increased after applying the layer merging, the depth is reduced due to less number of layers. Therefore, besides reducing the number of clock cycles, the proposed layer merging method could reduce the size of CTV memory by 26.2% and 13.9% for BG1 and BG2 codes, respectively.

Considering the 5G LDPC codes are extremely irregular and the degrees of some layers are relatively small, setting the width of CTV memory according to (12) will lead to a great waste of memory resource. To further reduce the memory cost, we present a split storage method. As mentioned

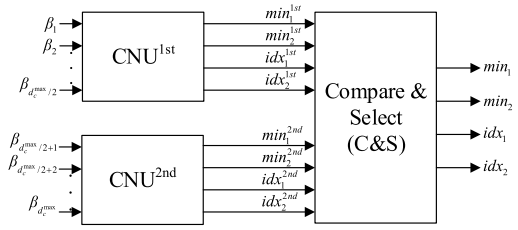


Fig. 13. The architecture of CNU.

TABLE II
THE SIZE OF CTV MEMORY.

	Before			After		
	Width	Depth	Total	Width	Depth	Total
BG1	$35z$	46	$1610z$	$36z$	33	$1188z$
BG2	$24z$	42	$1008z$	$28z$	31	$868z$

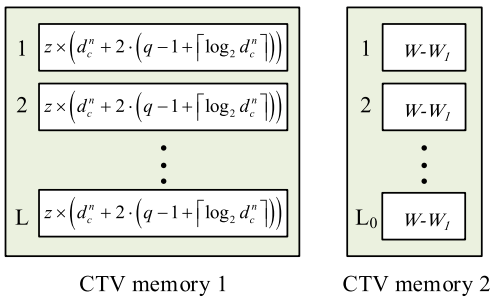


Fig. 14. The detailed structure of CTV memory.

in Section V-A, the CTV memory is divided into two parts. Fig. 14 shows the detailed structure of these two sub-memories, where d_c^n denotes the maximum row degree of layers except the core and orthogonal parts and $W_1 = z \times (d_c^n + 2 \cdot (q - 1 + \lceil \log_2 d_c^n \rceil))$. Since the width of the CTV messages generated in the core and orthogonal parts are larger than W_1, only the first W_1 bits of messages are stored in CTV memory 1 while the remaining bits are stored in CTV memory 2. For other layers, the CTV messages are totally stored in CTV memory 1. Because CTV memory 2 is specifically used for the layers in the core and orthogonal parts, its depth L_0 is less than L. Thanks to the spilt storage method, the size of CTV memory can be further reduced by 16.6% for BG1 codes and 18.4% for BG2 codes. Combining the layer merging method, 39.6% of CTV memory can be saved for BG1 codes and 29.8% can be saved for BG2 codes in total. Since the CTV memory occupies a large proportion of the decoder in area consumption, these modifications greatly benefit the total area reduction.

C. Interconnection Network Optimizations

Besides the memory block, the interconnection block is another important part that dictates the overall hardware overhead. For a QC-LDPC code generated from an $M_b \times N_b$ base matrix, N_b sets of APP messages are fed into the *Read Network* which outputs d_c^{max} sets of messages. Similarity, the *Write Network* selects d_c^{max} sets of APP messages from

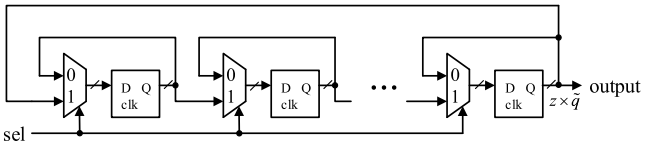


Fig. 15. The structure of the APP memory for extension bits.

N_b sets and replaces them by the updated APP messages. Thus, two complex interconnection networks are required which induce a long critical path and large area consumption. To minimize the hardware overhead of the interconnection block, two optimization methods are introduced.

First, the selective-shift structure is applied in the APP memory to minimize N_{io}, which represents the number of inputs to the *Read Network* and also the number of outputs from the *Write Network*. Considering the diagonal property of identity matrix $\mathbf{I}$ in the base matrix, only one set of APP messages corresponding to the extension bits is needed in each decoding layer. Therefore, rather than sending all APP messages corresponding to the extension bits to the *Read Network*, in this work, these APP messages are fed into the *Read Network* one set by one set in sequence order. Consequently, N_{io} and hence the area and critical path of the interconnection block can be significantly reduced. For this reason, we store the APP meesages corresponding to the extension bits in an individual memory which is implemented by applying the selective-shift structure, as shown in Fig. 15. The control signal *sel* decides whether the data should be rotated or not, which is only enabled when the extension part is processed. By cyclically shifting the APP messages, the address of the required APP messages is fixed during decoding so the required APP messages could be easily obtained.

Since the layer merging method is applied, two sets of APP messages corresponding to the orthogonal part are needed at the same clock cycle. In order to obtain them simultaneously, two memories are applied to store the APP values corresponding to extension bits. Therefore, all APP messages of the core bits and two sets of APP messages of the extension bits are input to the *Read Network* in each clock cycle. Hence, N_{io} could be reduced from 68 to 28 for BG1 codes and from 52 to 16 for BG2 codes. The APP memory for extension bits can also be implemented with the DP-RAM which consumes less area and power. However, to support massively parallel initialization, the shift register set is adopted in the proposed design.

To further reduce the hardware overhead of interconnection networks, the APP messages processed in the *Read Network* are reordered. For simplicity, we take the $R = 1/5$, $Z = 52$, $N = 2600$ BG2 code as an example. As stated before, 16 sets of messages are fed into the *Read Network* in each iteration, which are denoted as s_1, s_2, …, and s_{16}, respectively. Therefore, each output is selected from 16 inputs, leading to large hardware overhead. In the proposed design, the APP messages are reordered. Fig. 16 shows the mapping between the input and output messages in the *Read Network*, in which the last row indicates the number of inputs to generate the

	OUT1	OUT2	OUT3	OUT4	OUT5	OUT6	OUT7	OUT8	OUT9	OUT10
Layer 1	s1	s2	s3	s4	s11	s7	\	s12	\	s10
Layer 2	s1	s8	s5	s4	s6	s7	s9	s12	s13	s10
Layer 3	s1	s2	s5	s4	s11	\	s9	\	s13	s14
Layer 4	s14	s2	s3	s5	s11	s7	s9	s6	s8	s10
Layer 5	s1	s2	s15	\	\	\	\	s12	\	\
Layer 6	s1	s2	s15	\	s6	\	\	s12	s8	\
Layer 7	s1	s8	s15	\	s6	\	\	s12	\	s10
Layer 8	\	s2	s15	\	s6	\	\	s12	s8	s14
Layer 9	s1	s2	s15	\	\	\	\	\	s13	\
Layer 10	\	s2	s15	\	s11	\	s9	s12	\	\
Layer 11	s1	s2	s15	\	\	s7	\	\	s8	\
Layer 12	s1	s8	s15	\	\	\	\	\	s10	s14
Layer 13	\	s2	s15	s4	\	\	\	s12	\	\
Layer 14	s1	s2	s15	\	\	\	s9	\	\	s14
Layer 15	\	s2	s15	\	\	s7	\	s12	\	s14
Layer 16	s1	\	s15	\	s11	\	\	s12	\	\
Layer 17	\	s2	s15	\	\	\	\	s12	s13	s10
Layer 18	\	s2	s15	\	s6	\	\	s12	s13	\
Layer 19	s1	s8	s15	\	\	s7	\	\	\	\
Layer 20	s1	s2	s15	\	s11	\	\	\	\	\
Layer 21	s1	s2	\	s5	s15	s9	s16	s12	\	s14
Layer 22	s1	s2	s3	s4	s15	\	s16	s6	\	\
Layer 23	s1	s2	s3	\	s15	\	s16	s6	\	s10
Layer 24	s1	s8	s3	\	s15	s7	s16	\	s13	s14
Layer 25	s1	s2	s3	s5	s15	\	s16	s6	\	\
Layer 26	s14	s2	s3	\	s15	\	s16	s6	s8	s10
Layer 27	s1	s8	s3	s11	s15	\	s16	s6	s13	\
Layer 28	s1	s2	\	\	s15	s12	s16	s6	s13	s14
Layer 29	s1	s8	s3	s11	s15	\	s16	\	\	s14
Layer 30	s1	s2	\	\	s15	s12	s16	s6	s8	s13
NO. inpus	2	2	3	3	3	3	2	2	3	3

Fig. 16. Mapping relationship between the input and output messages in the *Read Network*.

corresponding output. It can be seen that by applying the message reordering, all outputs can be generated by no more than four inputs. Therefore, the critical path of the *Read Network* is only two multiplexers and the number of required multiplexers is significantly reduced. Since the *Write Network* has similar structure as the *Write Network*, the mapping for the *Write Network* can be easily deduced from Fig. 16.

VI. Implementation Results

The implementation results, as well as the corresponding comparisons, are reported in this section. The decoder architecture is implemented in RTL and synthesized under the TSMC 90-nm CMOS technology using the Synopsys Design Compiler. The Synopsys Prime Time PX tool is used for power estimation. We generate the VCD file to read the switching activity first and then estimate the power consumption of the decoder using time-based power analysis. Consider the $R = 1/5$, $Z = 52$, $N = 2600$ BG2 code to implement the proposed architecture. By applying the proposed methods presented in Section V, we can conclude that for the selected code, L could be reduced from 40 to 30 so the number of clock cycles could be reduced by 25%. Moreover, 29.2% of the CTV memory can be saved and the hardware overhead of the interconnection networks could be significantly reduced.

Since the AMS and OMS decoders have similar hardware complexity, we compare the implementation results of the proposed IAMS decoder to that of the OMS decoder. The IAMS and OMS decoders are implemented according to the structure presented in Section V-A. Table III shows the ASIC synthesis results on 90-nm CMOS technology of the OMS and IAMS decoders which are quantized with parameters $(q, \tilde{q}) = (4, 6)$. Due to the additional storage and calculations for applying idx_2, the IAMS decoder has a slightly larger area and lower throughput than the OMS decoder. However, this overhead is negligible considering its much-improved

TABLE III

ASIC Synthesis Results on 90-nm CMOS Technology.

Decoder	OMS(4,6)		IAMS(4,6)	
Variant	original	modified	original	modified
Frequency (MHz)	120.9	163.9 +35.6%	113.9	158.2 +38.9%
Iterations	15	15	15	15
Throughput (Mbps)	523.9	947.0 +80.8%	493.6	914.0 +84.1%
Area (mm^2)	1.889	1.271 -32.6%	1.997	1.353 -32.3%
XOR Gate Count	334644	225163	353777	239690
Power (mW)	72.0	73.6 +2.22%	73.0	76.4 +4.66%
TAR ($Mbps/mm^2$)	277.3	745.1 +168.7%	247.2	675.5 +173.3%
NTAR ($Mbps/mm^2$/iter)	4160	11177	3708	10133

TABLE IV

The Area of Each Block.

Decoder	OMS(4,6)		IAMS(4,6)	
Variant	original	modified	original	modified
Total area (mm^2)	1.889	1.271 -32.6%	1.997	1.353 -32.3%
Interconnection (mm^2)	0.848	0.343 -59.6%	0.848	0.343 -59.6%
APP Memory (mm^2)	0.334	0.374 +12.0%	0.334	0.374 +12.0%
CTV Memory (mm^2)	0.358	0.273 -23.7%	0.431	0.323 -25.1%
Others (mm^2)	0.349	0.281 -19.5%	0.384	0.313 -18.5%

decoding performance. Since there are no published architectures or synthesis results of 5G LDPC decoders, in order to evaluate the effectiveness of our optimization methods, Table III lists the synthesis results of decoders with and without applying the optimizations proposed in Section V-B and Section V-C. It can be seen that after being modified, the area of the IAMS decoder is reduced by 32.3% and the frequency is improved by 38.9%. Considering the decoding cycles is decreased by 25%, the throughput could be improved by up to 84.1%, reaching 914Mbps, which could meet the 5G requirements in terms of throughput on rate-1/5 codes [9]. In order to make easier comparisons with other works that use different technologies, the area in gate equivalents of each decoder is also reported, which is computed by dividing the total area by that of an XOR gate. Moreover, the power consumption results are reported. As can be seen, though the modified decoders are synthesized at a higher frequency, they nearly consume the same power as the original ones due to less resource usage. To keep the throughput comparison on an equal basis, we further define the TAR metric and normalized TAR (NTAR) metric. TAR = throughput/area and NTAR = TAR $\times$ Iterations. Table III shows that the TAR of the IAMS decoder is increased from 247.2Mbps/mm^2 to 675.5Mbps/mm^2, increasing by 173.3%. The similar conclusion can be drawn for the OMS decoder, for which the TAR is improved by 168.7%. Therefore, the effectiveness of the proposed optimizations can be proved.

Table IV lists the area of each block in the whole decoder. For the IAMS decoder, the area of the interconnection blocks (*Read Network*, *Write Network*, and *LBSs*) is decreased from

$0.848mm^2$ to $0.343mm^2$ after applying the proposed modifications, decreased by up to 59.6%. Moreover, the area of CTV memory is reduced by 25.1%, which is less than the theoretical analysis (29.2%). This result mainly comes from the reason that the area of DP-RAM is not fully decided by data size, so the reduction of the total area is not strictly equal to that of the data. We also notice that the area of the APP memory is slightly increased, which comes from applying the selective-shift structure. However, considering it greatly benefits the interconnection blocks, this overhead is acceptable.

VII. Conclusion

In this article, we propose a high-performance decoding algorithm, named the improved adapted min-sum algorithm, for fixed-point decoding of 5G LDPC codes. To reduce the error-probability of degree-1 VNs, a new CN-update function is designed, and the column degree adaptation is proposed to alleviate the excessive growth of posterior probability in high-degree VNs. As a result, the proposed decoder could outperform the state-of-the-art AMS decoder by 0.4dB in FER performance. We also present an efficient architecture for 5G LDPC decoders. First, the layer merging technique is applied based on the orthogonality property of the base matrix. Then, the split storage method is adopted to further reduce CTV memory cost. Finally, the interconnection blocks are optimized by using the selective-shift structure and message reordering method. Implementation results demonstrate that the proposed architecture can improve the throughput-to-area ratio by 173.3%.

References

[1] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inf. Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[2] *IEEE 802.11n Wireless LAN Medium Access Control MAC and Physical Layer PHY Specifications*, Standard IEEE 802.11n-D2.0, 2007.

[3] *Second Generation Framing Structure, Channel Coding and Modulation Systems for Broadcasting, Interactive Services, News Gathering and Other Broadband Satellite Applications (DVB-S2)*, ETSI, Sophia Antipolis, France, 2009.

[4] *Standard: Synchronization Standard for Distributed Transmission*, ATSC, Boston, MA, USA, 2007.

[5] *Multiplexing and Channel Coding*, document TS 38.212 V15.0.0, 3GPP, Dec. 2017.

[6] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 599–618, Feb. 2001.

[7] M. P. C. Fossorier, M. Mihaljevic, and H. Imai, "Reduced complexity iterative decoding of low-density parity check codes based on belief propagation," *IEEE Trans. Commun.*, vol. 47, no. 5, pp. 673–680, May 1999.

[8] J. Chen, A. Dholakia, E. Eleftheriou, M. P. C. Fossorier, and X.-Y. Hu, "Reduced-complexity decoding of LDPC codes," *IEEE Trans. Commun.*, vol. 53, no. 8, pp. 1288–1299, Aug. 2005.

[9] T. Richardson and S. Kudekar, "Design of low-density parity check codes for 5G new radio," *IEEE Commun. Mag.*, vol. 56, no. 3, pp. 28–34, Mar. 2018.

[10] K. Le Trung, F. Ghaffari, and D. Declercq, "An adaptation of min-sum decoder for 5G low-density parity-check codes," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Sapporo, Japan, May 2019, pp. 1–5.

[11] *LDPC Decoding With Adjusted Min-Sum*, document R1-1610140, TSG RAN WG1 #86bis, 3GPP, Qualcomm Incorporated, Lisbon, Portugal, Oct. 2016.

[12] W. Zhou and M. Lentmaier, "Generalized two-magnitude check node updating with self correction for 5G LDPC codes decoding," in *Proc. 12th Int. ITG Conf. Syst., Commun. Coding*, Rostock, Germany, Mar. 2019, pp. 1–6.

[13] K. Sun and M. Jiang, "A hybrid decoding algorithm for low-rate LDPC codes in 5G," in *Proc. 10th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Hangzhou, China, Oct. 2018, pp. 1–5.

[14] X. Wu, M. Jiang, and C. Zhao, "Decoding optimization for 5G LDPC codes by machine learning," *IEEE Access*, vol. 6, pp. 50179–50186, 2018.

[15] C. Jones, E. Valles, M. Smith, and J. Villasenor, "Approximate-MIN* constraint node updating for LDPC code decoding," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Boston, MA, USA, Oct. 2003, pp. 157–162.

[16] K. Zhang, X. Huang, and Z. Wang, "A high-throughput LDPC decoder architecture with rate compatibility," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 58, no. 4, pp. 839–847, Apr. 2011.

[17] C.-C. Cheng, J.-D. Yang, H.-C. Lee, C.-H. Yang, and Y.-L. Ueng, "A fully parallel LDPC decoder architecture using probabilistic min-sum algorithm for high-throughput applications," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 9, pp. 2738–2746, Sep. 2014.

[18] H.-C. Lee, M.-R. Li, J.-K. Hu, P.-C. Chou, and Y.-L. Ueng, "Optimization techniques for the efficient implementation of high-rate layered QC-LDPC decoders," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 64, no. 2, pp. 457–470, Feb. 2017.

[19] I. Tsatsaragkos and V. Paliouras, "A reconfigurable LDPC decoder optimized for 802.11n/AC applications," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 26, no. 1, pp. 182–195, Jan. 2018.

[20] T. T. Nguyen-Ly, V. Savin, K. Le, D. Declercq, F. Ghaffari, and O. Boncalo, "Analysis and design of cost-effective, high-throughput LDPC decoders," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 26, no. 3, pp. 508–521, Mar. 2018.

[21] C.-Y. Liang, M.-R. Li, H.-C. Lee, H.-Y. Lee, and Y.-L. Ueng, "Hardware-friendly LDPC decoding scheduling for 5G HARQ applications," in *Proc. ICASSP-IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Brighton, U.K., May 2019, pp. 1418–1422.

[22] R. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inf. Theory*, vol. IT-27, no. 5, pp. 533–547, Sep. 1981.

[23] Z. Mheich, T.-T. Nguyen-Ly, V. Savin, and D. Declercq, "Code-aware quantizer design for finite-precision min-sum decoders," in *Proc. IEEE Int. Black Sea Conf. Commun. Netw. (BlackSeaCom)*, Varna, Bulgaria, Jun. 2016, pp. 1–5.

[24] W. E. Ryan, "An introduction to LDPC codes," in *CRC Handbook for Coding and Signal Processing for Magnetic Recording Systems*, B. Vasic, Ed. Boca Raton, FL, USA: CRC Press, 2004, ch. 36.

[25] T. Richardson, "Error floors of LDPC codes," in *Proc. 41st Annu. Allerton Conf. Commun., Control, Comput.*, Oct. 2003, pp. 1426–1435.

[26] X. Zhang and P. H. Siegel, "Quantized iterative message passing decoders with low error floor for LDPC codes," *IEEE Trans. Commun.*, vol. 62, no. 1, pp. 1–14, Jan. 2014.

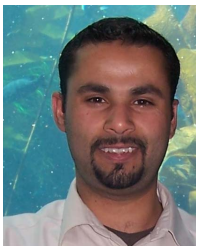
[27] F. Angarita, J. Valls, V. Almenar, and V. Torres, "Reduced-complexity min-sum algorithm for decoding LDPC codes with low error-floor," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 7, pp. 2150–2158, Jul. 2014.

[28] C. Zhang, S. Weng, X. You, and Z. Wang, "Area-efficient check node unit architecture for single block-row quasi-cyclic LDPC codes," in *Proc. IEEE Asia Pacific Conf. Circuits Syst. (APCCAS)*, Ishigaki, Japan, Nov. 2014, pp. 17–20.

[29] M. Weiner, B. Nikolic, and Z. Zhang, "LDPC decoder architecture for high-data rate personal-area networks," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Janeiro, Brazil, May 2011, pp. 1784–1787.

Hangxuan Cui received the B.S. degree in underwater acoustic engineering from Northwestern Polytechnical University, Xi'an, China, in 2017. He is currently pursuing the Ph.D. degree with Nanjing University.

His research interests include channel coding algorithms and low-power and high-throughput VLSI systems for digital signal processing.

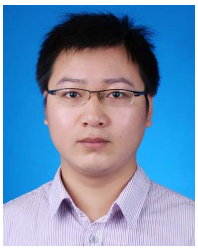


Fakhreddine Ghaffari (Member, IEEE) received the degree in electrical engineering and master's degree from the National School of Electrical Engineering (ENIS), Tunisia, in 2001 and 2002, respectively, and the Ph.D. degree in electronics and electrical engineering from the University of Sophia Antipolis, France, in 2006.

He is currently an Associate Professor with the Université de Cergy-Pontoise, France. His research interests include VLSI design and implementation of reliable digital architectures for wireless communication applications in ASIC/FPGA platform and the study of mitigating transient faults from algorithmic and implementation perspectives for high-throughput applications.

Khoa Le (Member, IEEE) received the bachelor's and M.Sc. degrees in electronics engineering from the Ho Chi Minh City University of Technology (HCMUT), Vietnam, in 2010 and 2012, respectively, and the Ph.D. degree from the Université de Cergy-Pontoise, France, in 2017. He is currently a Post-Doctoral Researcher with the ETIS Laboratory, ENSEA, France. His research interest includes error correcting code algorithms, analysis, and their implementations in FPGA/ASIC.

David Declercq (Senior Member, IEEE) was born in June 1971. He received the Ph.D. degree in statistical signal processing from the Université de Cergy-Pontoise, France, in 1998. From 2009 to 2014, he held the junior position with the Institut Universitaire de France. He is currently a Full Professor with ENSEA, Cergy. He is also the General Secretary of the National GRETSI Association. He worked several years on the particular family of LDPC codes, both from the code and decoder design aspects. Since 2003, he has been developing a strong expertise on non-binary LDPC codes and decoders in high order Galois fields GF(q). A large part of his research projects are related to non-binary LDPC codes. He mainly investigated two aspects the design of GF(q) LDPC codes for short and moderate lengths and the simplification of the iterative decoders for GF(q) LDPC codes with complexity/performance tradeoff constraints. He published more than 40 articles in major journals [the IEEE TRANSACTIONS ON COMMUNICATIONS, the IEEE TRANSACTIONS ON INFORMATION THEORY, the IEEE COMMUNICATONS LETTERS, and *EURASIP Journal on Wireless Communications and Networking* (JWCN)], and more than 120 articles in major conferences in information theory and signal processing. His research interests include digital communications and error-correction coding theory.



Jun Lin (Senior Member, IEEE) received the B.S. degree in physics and the M.S. degree in microelectronics from Nanjing University, Nanjing, China, in 2007 and 2010, respectively, and the Ph.D. degree in electrical engineering from Lehigh University, Bethlehem, in 2015. From 2010 to 2011, he was an ASIC Design Engineer with AMD. In summer 2013, he was an Intern with Qualcomm Research, Bridgewater, NJ, USA. In June 2015, he joined the School of Electronic Science and Engineering, Nanjing University, where he is currently an Associate Professor. He was a member of the Design and Implementation of Signal Processing Systems (DISPS) Technical Committee of the IEEE Signal Processing Society. His current research interests include low-power high-speed VLSI design for digital signal processing and deep learning, hardware acceleration for big data processing, and emerging computer architectures. He was a co-recipient of the Merit Student Paper Award at the IEEE Asia Pacific Conference on Circuits and Systems in 2008, the Best Paper Award at the IEEE Computer Society Annual Symposium on VLSI (ISVLSI) in 2019, and the Best Paper Award (The First Place) at the IEEE International Signal Processing Systems (SiPS) in 2019. He was a recipient of the 2014 IEEE Circuits & Systems Society (CAS) Student Travel Award.

Zhongfeng Wang (Fellow, IEEE) received the B.E. and M.S. degrees from the Department of Automation, Tsinghua University, Beijing, China, in 1988 and 1990, respectively, and the Ph.D. degree from the University of Minnesota, Minneapolis, in 2000. He was with Oregon State University and National Semiconductor Corporation. From 2007 to 2016, he was a Leading VLSI Architect with Broadcom Corporation, CA, USA. Since 2016, he has been a Distinguished Professor with Nanjing University, China.

He is a world-recognized expert on Low-Power High-Speed VLSI Design for Signal Processing Systems. He has published more than 200 technical articles with multiple best paper awards received from the IEEE technical societies, among which is the VLSI Transactions Best Paper Award of 2007. He has edited one book VLSI and held more than 20 U.S. and China patents. In the current record, he has had many articles ranking among top 25 most (annually) downloaded manuscripts in the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) PER STYLE SYSTEMS. His current research interests include optimized VLSI design for digital communications and deep learning. He has also served as a TPC member and various chairs for tens of international conferences. Moreover, he has contributed significantly to the industrial standards. So far, his technical proposals have been adopted by more than 15 international networking standards. In 2015, he was elevated to the Fellow of IEEE for contributions to VLSI design and implementation of FEC coding. In the past, he has served as an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS I: REGULAR PAPERS, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS II: REGULAR PAPERS, and the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS PER STYLE for many terms.