Hardware Sharing Design Method of Rate Matching and Interleaving for Wireless Terminal in Industrial Internet of Things

Wei Chen, Kejia Huo, and Dake Liu, *Senior Member, IEEE*

***Abstract*—Reducing the power consumption of baseband chips in wireless terminals of the Industrial Internet of Things (IIoT) is a great challenge. Among them, the logic function of the rate matching and interleaving hardware module is very complex, which occupies a considerable portion of power consumption in the baseband chip, so it is of great significance to design a low-power rate matching and interleaving hardware. Due to differences in interleaving algorithms and throughput, the interleavers used in 4G long term evolution (LTE) and 5G NR/6G have not been merged into a single architecture. Switching between different standards provides new options for interleaver design, and by configuring the architecture of different interleavers, it is possible to use the same hardware for different standards to reduce hardware resources and power consumption. This article studies the block interleaving and rate matching of turbo codes, convolutional codes, polar codes, and low-density parity check (LDPC) codes used in 4G LTE and 5G NR/6G communication links. With regard to the different algorithms for these four types of encoding, shared design is carried out on the hardware structure. In this experiment, according to the proposed memory and interleaving sharing scheme for hardware design and hardware simulation, the area overhead of 0.1 μm^2 and power consumption of 4.31 mW are obtained by Synopsys synthesis at the SMIC 28-nm process and the frequency of 50 MHz. This achieves the maximum hardware reuse of four encoding schemes in the downlink communication link of 4G LTE and 5G NR/6G, and reduce power consumption.**

***Index Terms*—Block interleaving, hardware multiplexing, Industrial Internet of Things (IIoT), rate matching.**

I. Introduction

With the rapid development of the Industrial Internet of Things (IIoT), it is also facing some challenges: in the stage of interconnectivity, there may be strong electromagnetic interference. And during channel transmission, noise can damage the integrity and accuracy of information, potentially leading to errors in the information received at various levels [1]. In order to solve this problem, after encoding at the physical layer, rate matching and interleaving are adopted to better combat noise and improve the reliability of data transmission [2], and improve the stability of the IIoT wireless terminal system. Rate matching and interleaving hardware are indispensable components of IIoT wireless terminals [3]. This hardware is typically implemented using application-specific integrated circuit (ASIC) technology and are integrated into baseband chips, such as Vivek Karthick Perumal, presented a reconfigurable design for interleaver based on ASIC technology [4], Akshaya et al. [5] designed rate matching and interleaving hardware for turbo codes, which was implemented and synthesized in ASIC, and Lakshmi and Jayakumari [6] proposed a reduced complexity rate-matching and channel interleaver based on ASIC technology. The rate matching and interleaving hardware module accounts for approximately 20% of the total power consumption in 5G baseband chips [2], [6]. These hardware modules account for a considerable amount of the power consumption in baseband chips, to the extent, it impacts the performance of IIoT wireless terminal system [7]. Reducing the power consumption of baseband chips in IIoT wireless terminals is a great challenge to be solved. We hence focus on the design of low-power rate matching and interleaving hardware, and study the hardware fusion and multiplexing scheme to achieve the low-power rate matching and interleaving hardware under multistandards for IIoT wireless terminal, this not only contributes to improve the reliability and timeliness of information transmission, but also solves the challenge of reducing the power consumption of baseband chips in IIoT wireless terminals.

A. Object and Contribution

In Internet of Things (IoT) 6G environments, the power consumption of the interleaver proposed in [8] was 20 mW. In 5G NR standard, the power consumption and memory area of the rate matching and interleaving hardware proposed in [9] were 36 mW and 3.1 μm^2, respectively. The power consumption of the interleaver proposed in [10] was 28.62 mW. It can be seen that the state-of-art rate matching and interleaving hardware have high-power consumption and silicon area. At present, there is not yet a fully implementable low-power rate matching and interleaving hardware for 4G long term evolution (LTE) and 5G NR/6G communication links. For four encoding schemes under 4G LTE and 5G NR/6G communication link, we propose the hardware-friendly rate matching and parallel interleaving method for turbo code, convolutional code, polar

Received 26 July 2024; revised 17 September 2024 and 11 October 2024; accepted 2 November 2024. Date of publication 5 November 2024; date of current version 7 March 2025. This work was supported by the National Natural Science Foundation of China under Grant 62464005. *(Corresponding author: Dake Liu.)*

The authors are with the School of Information and Communication Engineering, Hainan University, Hainan 570228, China (e-mail: weichen@hainanu.edu.cn; kejiahuo@163.com; dakeliu6@163 com).

Digital Object Identifier 10.1109/JIOT.2024.3491976

code and low-density parity check (LDPC) code, find out the common part, and carry on the fusion design to the hardware architecture and circuits, which can achieve hardware module reuse to a certain extent. This design not only saves silicon area and power consumption in hardware implementation, but also realizes multiple standards through the same set of hardware and flexibly switches under multiple standards, which improves the flexibility and reconfiguration of the IIoT wireless terminal system to a certain extent. Moreover, we propose the parallel conflict-free addressing methods for parallel interleaving, the corresponding address computation circuits and the shared circuits for address computation unit of four codes, which can make the hardware system process multiple bits in one clock, improve the throughput of the system, reduce the encoding latency, and meet the requirements of IIoT wireless terminal system for higher user data rate to a certain extent. The main contributions of this article are as follows.

1) Proposing the hardware-friendly rate matching and parallel interleaving methods for four encoding schemes.
2) Proposing the parallel conflict-free addressing methods for parallel interleaving, and the shared circuits for the corresponding address computation unit of four codes.
3) Based on the proposed hardware fusion and multiplexing scheme, designing the hardware architecture and circuits for rate matching and interleaving of four codes involved in 4G LTE and 5G NR/6G communication link.

B. Organization

The organizational structure of this article is as follows. In Section II, the related work is presented. In Section III, rate matching and parallel interleaving methods of four coding schemes under 4G LTE and 5G NR/6G communication link are presented. In Section IV, implementation methods of hardware design and sharing are proposed. In Section V, experimental results and analysis are presented. In Section VI, summary is made.

II. Related Work

From the perspective of mobile communication development, rate matching in third generation partnership project (3GPP) standard was a process of getting the right number of bits out of full-rate channel encoded bits through repetition or perforation at the transmitter. The purpose of rate matching is to select a specific set of encoded bits for transmission by the process of puncturing and repetition, so as to support hybrid automatic repeat request (HARQ) operation. Hu et al. [11] proposed a rate matching algorithm based on the transfer function matching principle, which could enhance the performance of interleave-division multiple access (IDMA) scheme for 5G/6G. However, the hardware construction complexity of this method was relatively high. Suls et al. [12] proposed a rate matching scheme that could effectively reduce data transmission errors and retransmission requirements. However, the power consumption of this method was relatively high. Saha and Adrat [13] proposed a rate match method for polar codes based on novel downsizing type-selection parameter, which could optimize the utilization of transmission resources. However, this method increased the cost of hardware implementation. In summary, the aforementioned state-of-art rate matching methods [11], [12], [13] would cause relatively large hardware resource consumption and power consumption in the hardware implementation phase.

From the perspective of different interleaving methods, in IIoT. Son and Kim [14] proposed a bit interleaving coding scheme of MIMO systems, which regulated the interleaving indices to achieve that illegal eavesdropper could not retrieve any information. However, the implementation of this method required quite a few hardware resources, and increased the hardware cost of the system. Kong and Park [15] proposed an interleaving method that could divide interleaving patterns into P disjoint subpatterns, and achieved P-parallel processing without degrading error-rate performance noticeably. However, this method consumed a large amount of storage resources, and caused unnecessary overhead in the IIoT wireless terminal. A novel cyclic multilevel Tree interleaver (CM-Tree) theoretical model was proposed in [16], which could solve the problem of high latency in communications. A two-way time interleaving scheme was proposed in [17], which could further improve the reliability of data transmission. However, the power consumption of these interleaving method [16], [17] were relatively high. In summary, the aforementioned state-of-art interleaving methods [14], [15], [16], [17] had high-power consumption and silicon overhead.

From the perspective of different encoding schemes, 5G/6G adopted LDPC code as the channel encoding candidate for data channels [3]. Fang et al. [18] proposed a rate-matching and bit-interleaving functions for LDPC codes to overcome the problem of burst errors in the communication system. With the development of 5G/6G, polar codes have become a hot research topic in the field of communications. Wang et al. [19], Ahmed and Al-Raweshidy [20], and Kestel et al. [21] proposed different rate-matching and interleaving methods for concatenated polar codes to improve the error performance of finite-length polar codes. The 4G LTE communication link adopts turbo codes and convolutional codes. Sahnoune et al. [22] proposed an interleaving method for turbo codes using general symmetric unimodal maps, which could achieve better performance. Jihwan and Lee [23] proposed a block interleaving method for convolutional codes, which could improve the bit error rate (BER) performance of the convolutional code. The aforementioned state-of-art rate-matching and interleaving methods for different encoding schemes [18], [19], [20], [21], [22], [23] could only be implemented on the specialized hardware, not on the same hardware, which would increase additional hardware resource and power consumption of system, and reduce the flexibility of the system in different encoding schemes.

In summary, the research on interleaving and rate matching algorithms is relatively mature, but there is not yet a fully implementable low-power rate matching and interleaving circuit for 4G LTE and 5G NR/6G communication links. In order to solve this challenge, we propose the hardware-friendly rate matching and parallel interleaving algorithm, and propose the hardware fusion and multiplexing scheme to achieve the low-power rate matching and interleaving hardware under

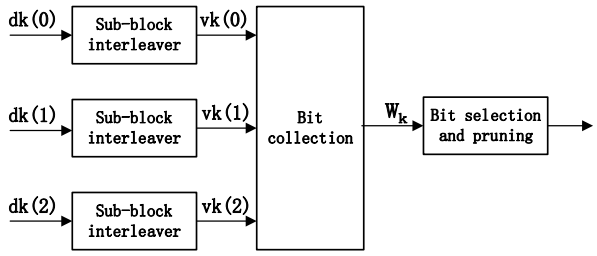

Fig. 1. Turbo code rate matching process. In figure, $dk(0)$, $dk(1)$, and $dk(2)$ represent the input bit stream of different sub-block interleaver, respectively. $vk(0)$, $vk(1)$, and $vk(2)$ represent the output bit stream of different sub-block interleaver, respectively. w_k represent the bit sequence stream after bit collection.

multistandards for IIoT wireless terminal. Moreover, some state-of-art parallel conflict-free addressing methods for parallel interleaving were proposed in [14], [15], [16], and [17], these methods introduced the misalignment technique when generating two addresses. But in these methods, the value of the misalignment factor was too large, when the number of conflicts was too high, these methods would use a large number of buffers to resolve conflicts. In order to reduce the cost of memory resources, the encoding latency, and improve system throughput, we propose the parallel conflict-free addressing methods for parallel interleaving, which can eliminate the conflict of most data blocks based on fewer storage resources.

III. Rate Matching and Interleaving of the Four Codes

After turbo code, convolutional code, LDPC code and polar code are encoded, the output bit stream must go through the rate matching process to select the information bits that match the channel resources for transmission. The rate matching process also involves the interleaving, which reorders the encoded bits to improve the anti-interference performance of the communication system. The rate matching process for each code is as follows.

A. Turbo Code Rate Matching

In the transmission channel of LTE system, there are two encoding schemes: 1) turbo code and 2) convolutional code, both of which have a bit rate of 1/3. According to 3GPP related protocols [24], the encoding output of turbo code consists of three-way bit streams $dk(0)$, $dk(1)$, and $dk(2)$. Rate matching is achieved by completing parallel sub-block interleaving of three-way bit streams, bit collection, bit selection and punching. The specific implementation process of rate matching is shown in Fig. 1.

The three-way bit streams generated by the encoder enter three sub-block interleavers with equal bit stream lengths in parallel. The length of each bit stream is D. The block interleaver is the row interleaving, and the number of columns C is 32. The number of rows R is determined according to

$$D \le R \times C. \tag{1}$$

TABLE I
Turbo Block Interleaved Column Permutation Mode

Columns	Intercolumn permutation pattern
$C^{TC}_{subblock}$	$< (\boldsymbol{p(0)}, \boldsymbol{p(1)}, \boldsymbol{p(2)}, \ldots, \boldsymbol{p(C^{TC}_{subblock}-1)} >$
32	< 0, 16, 8, 24, 4, 20, 12, 28, 2, 18, 10, 26, 6, 22, 14, 30, 1, 17, 9, 25, 5, 21, 13, 29, 3, 19, 11, 27, 7, 23, 15, 31 >

If $D = R \times C$, the three-way bit streams enter three sub-block interleavers directly from left to right in parallel. If $D < R \times C$, it is necessary to fill ND null bits and then perform intercolumn permutation, and the size of ND value is determined according to

$$\mathrm{ND} = R \times C - D. \tag{2}$$

The filled bits are 0, 1, 2, …, ND − 1, in the block interleaver, first filling the null bit, then filling the encoded bit stream, that is, $y_k = <null>, k = 0, 1, 2, \ldots, \mathrm{ND} - 1, y_{\mathrm{ND}+k} = dk(i), k = 0, 1, 2, \ldots, D-1, i = 0, 1, 2$. After filling, the bit sequence y_k is written into the $R \times C$ matrix by row

$$\begin{pmatrix} y_{P(0)} & y_{P(1)} & y_{P(2)} & \cdots & y_{P(C-1)} \\ y_{P(0)+C} & y_{P(1)+C} & y_{P(2)+C} & \cdots & y_{P(C-1)+C} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_{P(0)+(R-1)\times C} & y_{P(1)+(R-1)\times C} & y_{P(2)+(R-1)\times C} & \cdots & y_{P(C-1)+(R-1)\times C} \end{pmatrix}. \tag{3}$$

For $dk(0)$ and $dk(1)$, the bit stream written into the matrix is expanded by column for permutation, where $P(j)$ is the original column position of the jth permutation column, and the intercolumn permutation sequence is shown in Table I. After column permutation, the order of the $R \times C$ matrix is as follows:

$$\begin{pmatrix} y_{P(0)} & y_{P(1)} & y_{P(2)} & \cdots & y_{P(C-1)} \\ y_{P(0)+C} & y_{P(1)+C} & y_{P(2)+C} & \cdots & y_{P(C-1)+C} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_{P(0)+(R-1)\times C} & y_{P(1)+(R-1)\times C} & y_{P(2)+(R-1)\times C} & \cdots & y_{P(C-1)+(R-1)\times C} \end{pmatrix}. \tag{4}$$

The output bit stream of sub block interleaving is a sequence of bits outputted column by column from the inter column permutation matrix, and arranged sequentially. The bitstream output by block interleaving is represented as $vk(i)$, $k = 0, 1, 2, \ldots, R \times C - 1$, and $i = 0, 1$, where $v0(i)$ represents $y_{p(0)}$ and $v1(i)$ represents $y_{p(0)+C}$, which are output in columns in turn.

For $dk(2)$, the output sequence of block interleaving is $vk(2)$, $k = 0, 1, 2, \ldots$, and $R \times C - 1$. The positional correspondence between the encoded sequence written into the $R \times C$ matrix and the output sequence $vk(2)$ is shown as follows:

$$vk(2) = y_{\eth(k)} \tag{5}$$

wherein

$$\pi(k) = \left(P\left(\left\lfloor \frac{k}{R^{TC}_{\mathrm{subblock}}} \right\rfloor \right) + C^{TC}_{\mathrm{subblock}} \times \left(k \bmod R^{TC}_{\mathrm{subblock}} \right) + 1 \right) \bmod (R \times C). \tag{6}$$

where $C^{TC}_{\mathrm{subblock}}$ is the number of columns of the sub-block and $R^{TC}_{\mathrm{subblock}}$ is the number of rows of the sub-block. After

Algorithm 1 Turbo Code Bit Selection

1. set $k = 0$ and $j = 0$
2. while {k < E} do
3. if $w_{(k0+j) \bmod N_{cb}} \neq < null >$ then
4. $e_k = w_{(k0+j) \bmod N_{cb}}$
5. $k = k + 1$
6. end if
7. $j = j + 1$
8. end while

the three interleaved bitstreams are output, $vk(i), i = 0, 1, 2$ is put into the circular buffer in the order shown in 7, where w_k represents the bit sequence stream after bit collection

$$\begin{aligned} w_k &= vk(0) k = 0, 1, \ldots, R \times C - 1 \\ w_{R\times c+2k} &= vk(1) k = 0, 1, \ldots, R \times C - 1 \\ w_{R\times c+2k+1} &= vk(2) k = 0, 1, \ldots, R \times C - 1. \end{aligned} \tag{7}$$

After the bits enter the circular buffer, the bit selection process is completed by setting the relevant parameters according to the corresponding upstream and downstream transmission channels, and the bit selection is shown in Algorithm 1, where e_k is the bit selection output sequence, E is the bit length of e_k, and N_{cb} is the length of circular buffer.

B. Convolutional Code Rate Matching

In 4G LTE, in addition to the transmission channel (BCH, sidelink broadcast channel), the convolutional code encoding scheme is also used in some control information transmission (downlink control information (DCI), uplink control information, spatial channel information). The convolutional code rate matching process is the same as the process shown in Fig. 1, but the specific parallel sub-block interleaving and bit selection algorithms are different from those of turbo codes.

For the convolutional codes $dk(0)$, $dk(1)$, and $dk(2)$, it is assumed that each bit stream has D input information bits and the three outputs are of the same length. The $vk(0)$, $vk(1)$, and $vk(2)$ output in parallel by three sub-block interleavers are obtained by the following process.

1) The number of columns of the block interleaver C is set to 32, that is, the columns of the matrix from left to right are 0, 1, 2, …, $C - 1$.
2) The number of matrix rows is set as R, and the size of R is also determined by 1. The rows of matrix from top to bottom are 0, 1, 2, …, $R - 1$.
3) If the multiplication value of the row value and the column value is equal to the number of encoded bits per way, the three-way bit streams enter three sub-block interleavers directly from left to right in parallel. If the multiplication value of the row value and the column value is greater than the number of encoded bits per way, ND null bits are added and the ND value is determined according to 2. Then bit y_0 is written in row order starting from 0th column and 0th row. The coded bit is written to the $R \times C$ matrix, as shown in 8, where

TABLE II
CONVOLUTIONAL BLOCK INTERLEAVED COLUMN PERMUTATION MODE

Columns	Intercolumn permutation pattern
$C^{TC}_{subblock}$	$< (\boldsymbol{p(0)}, \boldsymbol{p(1)}, \boldsymbol{p(2)}, \ldots, \boldsymbol{p(C^{TC}_{subblock} - 1)} >$
32	< 1, 17, 9, 25, 5, 21, 13, 29, 3, 19, 11, 27, 7, 23, 15, 31, 0, 16, 8, 24, 4, 20, 12, 28, 2, 18, 10, 26, 6, 22, 14, 30 >

Algorithm 2 Convolutional Code Bit Selection

1. set $k = 0$ and $j = 0$
2. while $\{k < E\}$ do
3. if $w_{j \bmod K_W} \neq < null >$ then
4. $e_k = w_{j \bmod K_W}$
5. $k = k + 1$
6. end if
7. $j = j + 1$
8. end while

$y_k =< null >, k = 0, 1, 2, \ldots, \mathrm{ND} - 1, y_{\mathrm{ND}} + k = dk(i), k = 0, 1, 2, \ldots, D - 1, i = 0, 1, 2$

$$\begin{pmatrix} y_0 & y_1 & y_2 & \cdots & y_{C-1} \\ y_C & y_{C+1} & y_{C+2} & \cdots & y_{2C-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_{(R-1)\times C} & y_{(R-1)\times C+1} & y_{(R-1)\times C+2} & \cdots & y_{(R\times C-1)} \end{pmatrix}. \tag{8}$$

After the coded bits are written to the matrix, the intercolumn interleaving is performed according to the intercolumn permutation pattern given in Table II, where $P(j)$ is the original column position of the jth permutation column. After the intercolumn permutation, the bit sequence is read out from the matrix $R \times C$ by column.

After block interleaving, the bit streams $vk(0)$, $vk(1)$, and $vk(2)$ enter the circular buffer of length $3 \times R \times C$ for bit collection in the order shown in

$$\begin{aligned} w_k &= vk(0) \text{ for } k = 0, 1, \ldots, R \times C - 1 \\ w_{R\times c+k} &= vk(1) \text{ for } k = 0, 1, \ldots, R \times C - 1 \\ w_{2\times R\times c+k} &= vk(2) \text{ for } k = 0, 1, \ldots, R \times C - 1. \end{aligned} \tag{9}$$

Bit selection is performed after the bit sequence is output from the circular buffer, and the bit selection process is shown in Algorithm 2, where K_W represents the sum of the data stream lengths of the three interleaved sub-blocks, the value of K_w is shown in 10. The bit sequence e_k with length E is output for transmission and the rate matching process of the convolutional code is completed

$$K_W = 3 \times R \times C. \tag{10}$$

C. Polar Code Rate Matching

Different transmission channels and control information use different coding schemes [25], [26]. In 5G NR/6G, polar code is used for BCH transmission channel and DCI.

The polar code rate matching is performed for each code block individually and consists of three parts: 1) sub-block interleaving; 2) bit selection; and 3) bit parallel interleaving.

Algorithm 3 Polar Code Sub-Block Interleaving

1. for $n = 0$ to $N - 1$ do
2. $i = \lfloor 32n/N \rfloor$
3. $J(n) = P(i) \times (N/32) + \mathrm{mod}(n, N/32)$
4. $y_n = d_{J(n)}$
5. end for

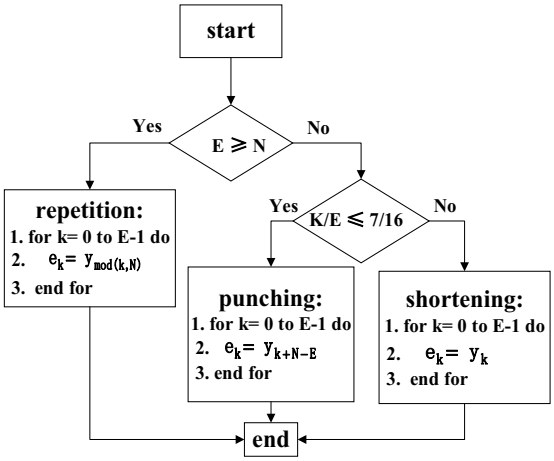

Fig. 2. Polar bit selection. In figure, E represents the length of bit selection output sequence, N represents the length of the post-encoding bitstream, K represents the length of pre-encoding bit stream, e_k represents the bit selection output sequence, and y_k represent the encoded bit sequence.

Algorithm 4 Polar Code Bit Parallel Interleaving

1. if $k = 0$
2. for $i = 0$ to $T - 1$ do
3. for j $= 0$ to $T - 1 - i$ do
4. if $k<E$
5. $v_{i,j} = e_k$
6. else
7. $v_{i,j} =< NULL >$
8. end if
9. k=k+1
10. end for
11. end for
12. k=0
13. for j=0 to T-1
14. for i=0 to T-1-j
15. if $v_{i,j} \neq< NULL >$
16. $f_k = v_{i,j}$
17. k=k+1
18. end if
19. end for
20. end for
21. else
22. for i=0 to E-1
23. $f_i = e_i$
24. end for
25. end if

1) Sub-Block Interleaving: The total length of the encoded output bit stream is $N(N = 2^n)$, and N bits enter 32 sub-blocks for sub-block interleaving. In the interleaving process, address mapping is performed according to Algorithm 3, where the interleaving output bit stream is y_k, and the sub-block interleaving sequence permutation order is shown in Table III. It can be seen from this table that after the bit sequence enters 32 sub-blocks, only the middle sequence needs to be replaced, and the positions of the sequences on both sides remain unchanged.

2) Bit Selection: Bit selection is performed on the interleaved output sequence, which is divided into repetition, punching and shortening. The bit selection process is shown in Fig. 2, where K is the pre-encoding bitstream length, N is the post-encoding bitstream length, and E is the rate matching output sequence length.

In the downlink transmission channel, the length of rate matching output sequence is 864, and the three address calculation formulas for repetition, punching and shortening are shown in Fig. 2. It can be seen from the bit selection formula that repetition is to add (E–N) repetition bits based on the bit sequence y_k; puncturing is to extract the last E bits from the original bit stream of length N; and shortening is to extract the first E bits from the original N bits

$$T(T + 1)/2 \geq E. \tag{11}$$

3) Bit Parallel Interleaving: Bit parallel interleaving is performed on the sequence of bit selection output. The bit selection output sequence is e_k, and the bit length of e_k is E. The output sequence through bit interleaving is f_k, and the length remains unchanged. The specific bit interleaving algorithm is determined based on the rate matching length parameter E and other parameters. When the value of E is not greater than 8192, the value of T is determined through 11, and the bit parallel interleaving process is completed according to Algorithm 4.

D. LDPC Code Rate Matching

In 5G NR/6G, LDPC codes are used for data channels. For transmission channels, uplink shared channel (UL-SCH), downlink shared channel (DL-SCH), and paging channel (PCH) also use LDPC codes. LDPC code rate matching consists of bit selection and bit parallel interleaving.

1) Bit Selection: For each encoding block, the encoded output bit stream d_k is written into a circular buffer of length N_{cb}. The value of N_{cb} should be determined based on parameters, such as the length of the encoded bitstream output, the maximum number of layers for each transmission block in the shared channel, and the number of code block segments. E is the encoding bit length for rate matching output. If the encoding block is not transmitted according to the code block group transmission information (CBGTI) specification, the rate matching output bit length is 0. Otherwise, the value of the E is determined according to Algorithm 5.

In Algorithm 5, N_L is the number of transmission layers for the transmission block mapping, Q_m is the modulation order, G is the total number of encoding bits available for block transmission. If CBGTI does not exist in the DCI of the scheduled transmission block, the value of C' is the number of segmented code blocks C. If CBGTI exists in the DCI of

TABLE III
POLAR CODE SUB-BLOCK INTERLEAVING PERMUTATION PATTERN

i	$P(i)$	i	$P(i)$	i	$P(i)$	i	$P(i)$	i	$P(i)$	i	$P(i)$	i	$P(i)$	i	$P(i)$
0	0	4	3	8	8	12	10	16	12	20	14	24	24	28	27
1	1	5	5	9	16	13	18	17	20	21	22	25	25	29	29
2	2	6	6	10	9	14	11	18	13	22	15	26	26	30	30
3	4	7	11	11	17	15	19	19	21	23	23	27	28	31	31

Algorithm 5 Determination of Output Length Value for LDPC Code Bit Selection

1. set $j = 0$
2. for r = 0 to C − 1
3. if j ≤ $C' - \mathrm{mod}(G/(N_L \cdot Q_m), C') - 1$
4. $E = N_L \cdot Q_m \cdot \left\lfloor \frac{G}{N_L \cdot Q_m \cdot C'} \right\rfloor$
5. else
6. $E = N_L \cdot Q_m \cdot \left\lceil \frac{G}{N_L \cdot Q_m \cdot C'} \right\rceil$
7. end if
8. $j = j + 1$
9. end for

Algorithm 6 LDPC Bit Selection

1. $k = 0$;
2. $j = 0$;
3. while $k < E$ do
4. if $d_{(k_0+j) \bmod N_{cb}} \neq < null >$
5. $e_k = d_{(k_0+j) \bmod N_{cb}}$
6. $k = k + 1$
7. end if
8. $j = j + 1$
9. end while

TABLE IV
STARTING POSITIONS OF DIFFERENT REDUNDANCY VERSIONS, K_0

rv_{id}	K_0	
	LDPC base graph 1	LDPC base graph 2
0	0	0
1	$\left\lfloor \frac{17N_{cb}}{66Z_c} \right\rfloor Z_c$	$\left\lfloor \frac{13N_{cb}}{50Z_c} \right\rfloor Z_c$
2	$\left\lfloor \frac{33N_{cb}}{66Z_c} \right\rfloor Z_c$	$\left\lfloor \frac{25N_{cb}}{50Z_c} \right\rfloor Z_c$
3	$\left\lfloor \frac{56N_{cb}}{66Z_c} \right\rfloor Z_c$	$\left\lfloor \frac{43N_{cb}}{50Z_c} \right\rfloor Z_c$

the scheduled transmission block, then the value of C' is the number of scheduling code blocks in the transmission block.

LDPC code rate matching is carried out independently according to each code block, then encoding the output bitstream with length N, and outputting the bit sequence $e_k$with length E after bit selection. The bit selection process is shown in Algorithm 6. Among them, the parameter k_0 must be determined according to parameters, such as redundancy versions and factor maps. Table IV shows different redundancy versions and starting positions, as well as the corresponding values of k_0.

Algorithm 7 LDPC Code Parallel Interleaving Algorithm Under Different Modulation Orders

1. for i 0 to $\frac{E}{32} - 1$ do
2. for j 0 to 31 do
3. if $Q_m == 2$ then
4. $\mathrm{t}_{i \cdot 32+j} = \mathrm{e}_{\frac{E}{2} \times \mathrm{mod}(j,2) + \left\lfloor \frac{j}{2} \right\rfloor + 16 \times i}$
5. else if $Q_m == 4$ then
6. $\mathrm{t}_{i \cdot 32+j} = \mathrm{e}_{\frac{E}{4} \times \mathrm{mod}(j,4) + \left\lfloor \frac{j}{4} \right\rfloor + 8 \times i}$
7. else
8. $\mathrm{t}_{i \cdot 32+j} = \mathrm{e}_{\frac{E}{8} \times \mathrm{mod}(j,8) + \left\lfloor \frac{j}{8} \right\rfloor + 4 \times i}$
9. end if
10. end for
11. end for

2) Bit Parallel Interleaving: The bit stream is output through bit selection and bit interleaving with bit length unchanged. Bit parallel interleaving process as shown in Algorithm 7, where Q_m is the modulation order and the number of rows of block interleaving, E is the output bit length after rate matching, e is the preinterleaving sequence, and t is the post-interleaving sequence. In order to be consistent with the interleaving output order in the standard, when the modulation order Q_m is 1, that is, the number of rows in the block interleaver is 1, the E bits are divided into several groups, each group of 32 bits, and are written into the data memory in sequence, and read out sequentially. When the modulation orders Q_m are 2, 4 and 8, in order to keep the original output bit order unchanged, we transform the position of the encoded sequence. Position transformation is completed according to different modulation orders.

E. Data Flow Diagram

We research the parallel interleaving algorithms of four different codes, and summarize their similarities, and list the parallel interleaving implementation processes for the four codes through a general data flow graph, and use the flow graph to share the four codes. According to the processing order of the data flow, the data flow diagram is divided into two parts: 1) precalculation part and 2) execution part. The specific implementation is shown in Figs. 3 and 4.

In Figs. 3 and 4, for the precalculation stage, the input data is processed first, because all four types of encodings involve block interleaving. In the precalculation stage, it is

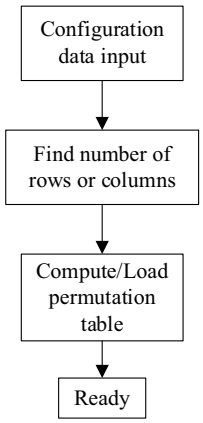

Fig. 3. Precalculation stage.

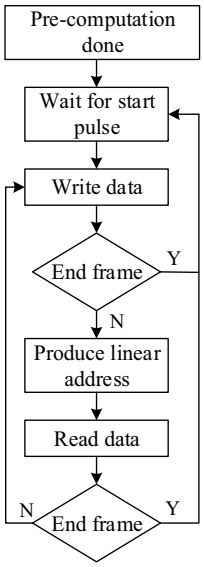

Fig. 4. Execution phase.

necessary to determine the number of rows and columns of the interleaved input data block, and load the required column permutation table/sub-block permutation table or address mapping formula according to different encodings. After the preparation is complete, it enters the execution stage, waits for the clock pulse, the bit stream is processed in sequence. When the rising edge of the clock arrives, the input data is written into the corresponding memory according to the address, and is read out according to the generated address. The data is processed sequentially according to the clock until the data processing is completed.

In this section, we propose the hardware-friendly rate matching and parallel interleaving method for four encoding schemes, which can improve the performance of rate matching and interleaving with low-hardware overhead. In addition, we research the similarities between the four rate matching processes to find the same operating units that they shared.

IV. Hardware Sharing Circuit Design Scheme

For rate matching and interleaving of the above four encodings, we present a scheme of memory sharing. Memory sharing is to use a certain number of SRAM block memories to achieve parallel block interleaving for turbo code, parallel block interleaving for convolutional code, bit parallel interleaving for polar code and LDPC code, respectively. In addition, we propose the parallel conflict-free addressing method for four encodings and their corresponding address computation circuits. Moreover, we design the shared circuits for four codes address computation unit. We also design the hardware architecture and specific implementation circuits for the rate matching and parallel interleaving of the four encodings.

A. Memory Sharing Design

According to the maximum data stream, the data storage is divided into eight submemory systems. Each memory size is 99×32 bits and can be up to 32-bits parallel.

For turbo codes, the encoded bit stream is cyclically shifted, bits are written to each submemory module in rows, and each row stores 32 bits of data. Intercolumn permutation is performed according to the column permutation rule and address generation formula. According to the (12), (13), and (14), the address is generated by the address computing unit to complete the address mapping.

For convolution codes, the data is put into the submemory block, and the column permutation table is preloaded, then the intercolumn permutation is performed according to Table II, and the address operation is performed according to (15), (16), and (17).

For polar codes, the encoded bit sequence is divided into 32 parts, in the storage design, each part is placed into the submemory module by column, and 32 sub-blocks correspond to 32 columns of the submemory module. According to Algorithm 3, the data reading process is equivalent to reading one bit at the same position of each part, in the implementation process, the data hence can be output by line.

For LDPC codes, block interleaving is to write the output bits of the bit selection to the eight submemory modules. Because of the parallel storage, it can be written in eight modules at the same time, and read in parallel.

In summary, we divide the memory into multiple identical submemories, which can write and read multiple memory modules at the same time. Previously, one bit was processed in one clock, but now it can be increased to 32 bits, which can increase the throughput of the system, and reduce the encoding latency.

B. Parallel Conflict-Free Addressing and Address Computation Unit Multiplexing

In order to support parallel interleaving of four codes, we propose the parallel conflict-free addressing methods for parallel interleaving and the corresponding address computation unit.

For turbo codes, the bit stream enters the block interleaver, the index is i, block interleaver output three-way bit stream, and enter the bit collection area, output bit index is j. The relation between input index i and output index j is shown in (12), (13), and (14). These formulas are used for parallel conflict-free addressing of data blocks.

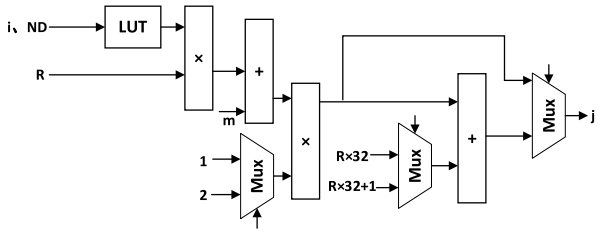

Fig. 5. Turbo code interleaved address calculation circuit. In figure, i represents the row index, ND represents the number of null bits, R represents the number of rows of the interleaver, m is $\lfloor (i+\mathrm{ND})/32 \rfloor$, LUT represents LUT module, Mux represents multiplexer module, and j represents the column index.

Sub-Block 0:

$$j = P[(i+\mathrm{ND})\&31] \times RTC + \left\lfloor \frac{i+\mathrm{ND}}{32} \right\rfloor. \tag{12}$$

Sub-Block 1:

$$j = 2 \times \left(P[(i+\mathrm{ND})\&31] \times RTC + \left\lfloor \frac{i+\mathrm{ND}}{32} \right\rfloor \right) + RTC \times 32. \tag{13}$$

Sub-Block 2:

$$j = 2 \times \left(P[(i+\mathrm{ND})\&31] \times RTC + \left\lfloor \frac{i+\mathrm{ND}}{32} \right\rfloor \right) + RTC \times 32 + 1. \tag{14}$$

In (12), (13), and (14), ND denotes the number of filled null bits, & denotes the bit and operation, *RTC* is the number of rows per sub-block, and $\lfloor\ \rfloor$ denotes the round down operation. $P[\ ]$ is the permutation sequence of the turbo code sub-block interleaving. The value of $P[\ ]$ is determined according to the intercolumn permutation pattern table of turbo codes in Table I. For turbo encoding outputs of any code length, when the output bit stream length is not an integer multiple of 32, ND null bits need to be added at the beginning of the bit segment to fill an entire line of the bit stream. In the pre calculation stage, the value of ND and the number of rows *RTC* in the block interleaver are determined. Based on the two parameters of row number *RTC* and the column number 32 in the block interleaver, the size of the block interleaver can be determined. After the block interleaving output, the ND value is set from 0 and calculated according to (12), (13), and (14), the index position of the padded null bits in the total bit stream after interleaving can be obtained, thereby eliminating the null bits.

According to (12), (13), and (14), the address generating circuit is shown in Fig. 5. First, based on the parameters i and ND, the current number of columns is obtained through bit-AND operation and lookup table (LUT). Under the control of the port, the input port of the data selector is selected, and the output address index of 12, 13, and 14 is completed in turn. In Fig. 5, m value is $\lfloor (i+\mathrm{ND})/32 \rfloor$.

For convolution codes, the bit index of each sub-block is i, by precalculation, calculating $i \ mod\ 32 = X$, finding $P(n) = X$ by looking up Table II, and obtaining n value. Parallel conflict-free interleaved address mapping for convolutional code sub-block is shown in (15), (16), and (17).

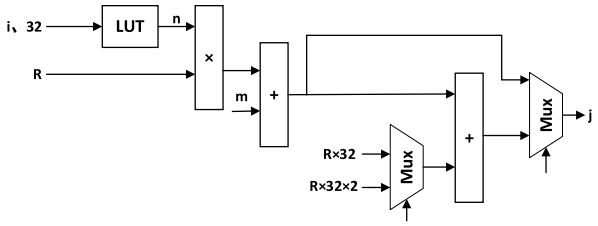

Fig. 6. Convolutional code address calculation circuit. In figure, n represents the column index in the sub-block.

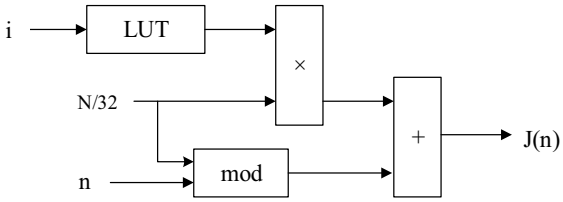

Fig. 7. Polar code sub-block interleaving address circuits. In figure, mod represents calculating remainder module and $J(n)$ represents the polar code sub-block interleaving address.

Sub-Block 0:

$$j = n \times R + \left\lfloor \frac{i+\mathrm{ND}}{32} \right\rfloor. \tag{15}$$

Sub-Block 1:

$$j = n \times R + \left\lfloor \frac{i+\mathrm{ND}}{32} \right\rfloor + R \times 32. \tag{16}$$

Sub-Block 2:

$$j = n \times R + \left\lfloor \frac{i+\mathrm{ND}}{32} \right\rfloor + R \times 32 \times 2. \tag{17}$$

In (15), (16), and (17), j is the index of a single bit in the total bitstream when entering the cyclic circular buffer, ND is the number of bits filled in the sub-block, R is the number of lines in each sub-block.

According to (15), (16), and (17), the address calculation circuit is shown in Fig. 6. First, bit-AND operation is performed on the two parameters i and 32, and the value of n is obtained through a LUT. Under the action of the control port, the input port of the data selector is selected, and the address calculation process of 15, 16, and 17 can be completed in turn.

In Fig. 6, m value is $\lfloor (i+\mathrm{ND})/32 \rfloor$.

For polar codes, according to Algorithms 3 and 5, the address calculation circuit that supports parallel conflict-free addressing is shown in Fig. 7.

For LDPC codes, rate matching includes bit selection and bit parallel interleaving. Bit selection is implemented by mod operation. Fig. 8 shows the parallel conflict-free addressing process involved in bit parallel interleaving for LDPC codes with different modulation orders. Bit parallel interleaving involves data inputting in row order and data outputting in column order, the specific hardware implementation circuit is relatively similar to that of LDPC codes.

For the address calculation of the above four codes, this article proposes an address calculation block diagram of rate matching and interleaved sharing, as shown in Fig. 9.

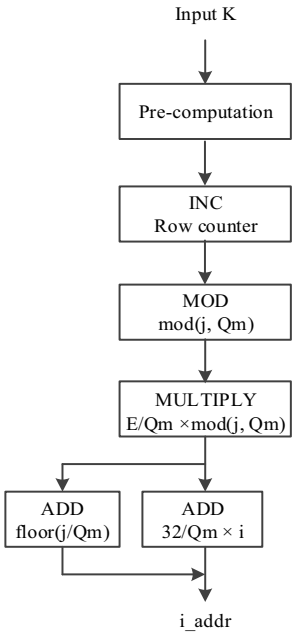

Fig. 8. Parallel conflict-free addressing scheme for LDPC code. In figure, K represents the length of pre-encoding bit stream, Q_m represents the modulation order, E represents the length of bit selection output sequence, INC represents increment-by-1 operation, ADD represents addition operation, and i_addr represents the LDPC code sub-block interleaving address.

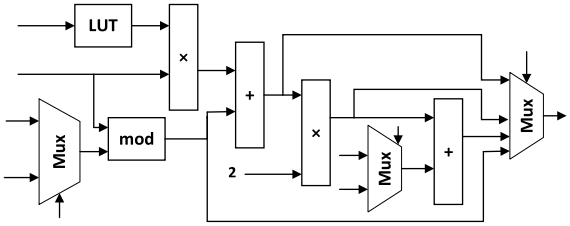

Fig. 9. Shared circuits for interleaved address calculation unit of turbo code, convolutional code, polar code, and LDPC code.

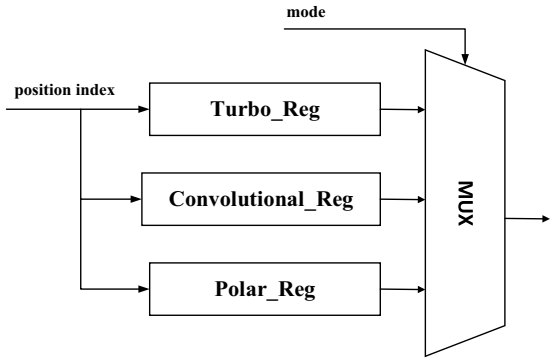

Fig. 10. Hardware structure of LUT module. In figure, Turbo_Reg is a register for storing turbo block interleaved column permutation mode, Convolutional_Reg is a register for storing convolutional block interleaved column permutation mode, and Polar_Reg is a register for storing polar code sub-block column permutation mode.

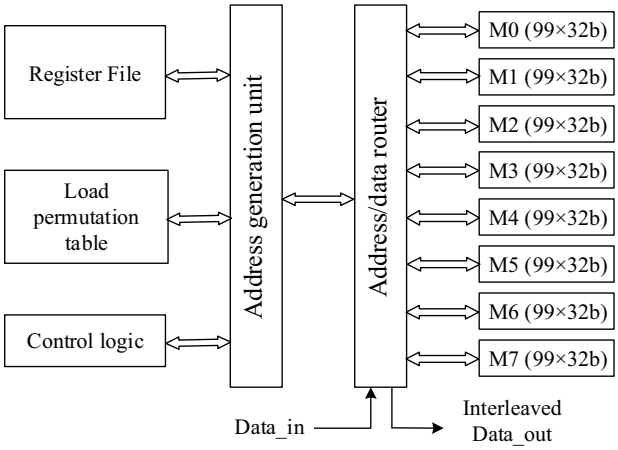

Fig. 11. Top-level hardware architecture of rate matching and interleaving for four encoding schemes. In figure, M0–M7 represent eight identical storage units.

In Fig. 9, for the LUT module, it can be utilized to determine parameters for turbo codes, convolutional codes, and polar codes by initializing different contents. For LDPC codes, the mod unit can be entered through MUX unit, and the result is calculated through the mod operation. The hardware struct of LUT module is shown in Fig. 10, where Turbo_Reg is a register for storing turbo block interleaved column permutation mode, Convolutional_Reg is a register for storing convolutional block interleaved column permutation mode, and Polar_Reg is a register for storing polar code sub-block column permutation mode. The position index and MUX unit is used to select parameters based on interleaved column permutation modes with different codes.

C. Hardware Sharing Architecture Design

For the rate matching and parallel interleaving of the four encodings, we propose a shared hardware architecture design for the four codes, as shown in Fig. 11. The register group module is used to implement the shift register, and column permutation is realized by cyclically shifting the bits of each row. Load permutation table (LUT) is used to select parameters from permutation modes of different encodings. The control unit contains some control signals for data selectors. The address generation unit generates the write address of the data according to the corresponding encoding scheme, this unit includes computation-intensive components involved in interleaving, namely, a turbo code block interleaving calculation formula, a convolutional code block calculation interleaving formula, a polar code sub-block interleaving address mapping formula, and an LDPC code bit selection address calculation formula. The address/data router is used to selected which storage unit are the address and data stored in. The storage module is mainly consisted of eight identical storage units, each with a size of 99 × 32b, which can be used to store the largest data stream.

D. Specific Circuit Design

Based on the proposed hardware architecture for rate matching and parallel interleaving of four encoding schemes, we design specific implementation circuits. The designed circuit is mainly divided into address generation module, data processing module and data storage module. The address generation module can generate read addresses and write

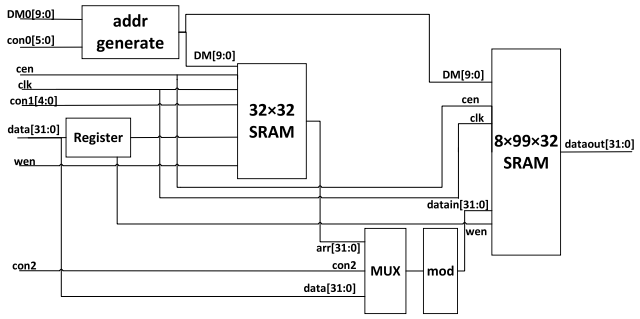

Fig. 12. Circuit block diagram. In figure, addr generate represents address generation module and SRAM represents the static random access memory, which is used as storage module.

addresses according to different control signals, the register is used to cyclically shift input bits, the multiplexer (MUX) is used to select input data, and the mod unit is used to perform mod operation. During data writing, the addresses generated by the address generation module are sent to the data processing module and the storage module, respectively. The data processing module is used to realize row and column replacement, and the storage module stores the processed bits according to the addresses. During data reading, the address generation unit generates different read addresses based on the control signal, and reads the corresponding data from the storage module in turn. The specific circuit block diagram is shown in Fig. 12.

Through the analysis of the existing references, we find that there is no complete hardware sharing design scheme for the rate matching and interleaving of turbo code, convolutional code, polar code and LDPC code involved in 4G LTE and 5G NR/6G communication link. In this section, we hence propose the scheme of memory sharing, the parallel conflict-free addressing method for four encoding schemes and their corresponding address computation circuits. Moreover, in order to further reduce hardware overhead and power consumption, we design the shared circuits for address computation unit of four codes, the hardware sharing architecture and specific implementation circuits for the rate matching and parallel interleaving of four codes.

V. Experiment

The hardware design proposed in this article is first simulated on MATLAB, then the corresponding RTL code is written based on Quartus prime, and the hardware circuit is described and debugged with Verilog, waveform simulation and function verification are completed by Modelsim. Under the SMIC process of 28 nm and the frequency of 50 MHz, the area cost and corresponding power consumption of the hardware are obtained through Synopsys synthesis.

A. Experimental Verification

First, the proposed method is verified by MATLAB. For turbo coding, assuming that the input of each sub-block is 32 bits, there is no need to fill null bits, the intercolumn permutation is completed in turn, and the bit collection is completed according to the corresponding rules. There are 96

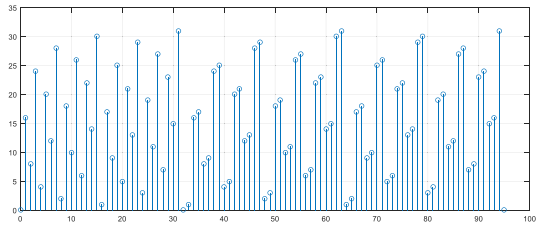

Fig. 13. Turbo code block interleaving output bit stream without padding null bit. In figure, the horizontal axis represents the position of the bit collection area and the vertical axis represents the original position index of the bit.

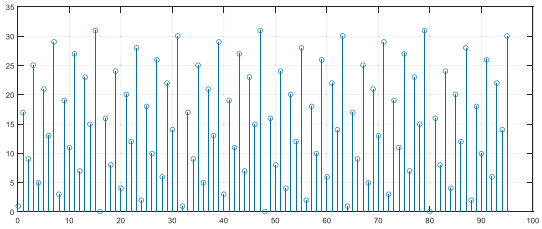

Fig. 14. Convolutional code block interleaving output bit stream without padding null bit.

bits in the bit collection area, the position index is defined for each bit according to the position when each bit initially enters the sub-block interleaver. Through MATLAB simulation, the bit position index of the bit collection area is obtained. In Fig. 13, the horizontal axis represents the position of the bit collection area, and the vertical axis represents the original position index of the bit at that position. From Fig. 13, it can be seen that at the position 0 to 31, the first sub-block outputs in column permutation order. Starting from the position 32, the output bits of the second sub-block are arranged in a cross order with those of the third sub-block. According to Table I, (12), (13), and (14), as can be seen from Fig. 13, the results are consistent with the standard.

For convolutional coding, it is also assumed that the input of each sub-block is 32 bits, and each output bit index is represented by a discrete sequence diagram, as shown in Fig. 14. It can be seen that each sub-block is output sequentially in the bit collection area. According to Table II, (15), (16), and (17), as can be seen from Fig. 14, the results are consistent with the standard.

Figs. 13 and 14 show that when the length of the encoded output bitstream is a multiple of 32 integers, it is not necessary to fill in null bits. However, in most cases, when the encoded output bits enter the block interleaver, the corresponding null bits need to be filled. We arbitrarily assume that for turbo codes and convolutional codes, each sub-block has 46 input data bits. First, 18 null bits need to be filled in the first part of the first row, then replaced between columns, and output by column. Assuming that the filled bits are all 0, in the bit collection area, the discrete sequence diagram is used to represent the bit stream of turbo code and convolutional code that enter the cyclic buffer after block interleaving. The total length of the three-way bit streams is 192. Among them, Fig. 15 represents the interleaved output sequence of turbo code bit stream blocks and Fig. 16 represents the interleaved

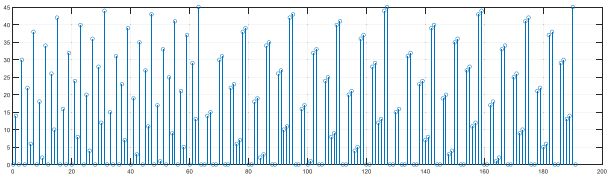

Fig. 15. Turbo code padding bit block interleaving diagram.

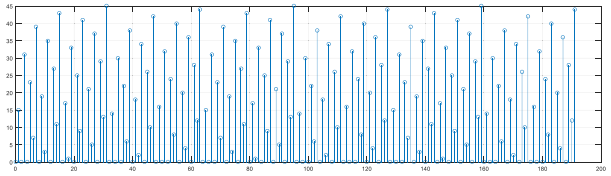

Fig. 16. Convolutional code padding bit block interleaving diagram.

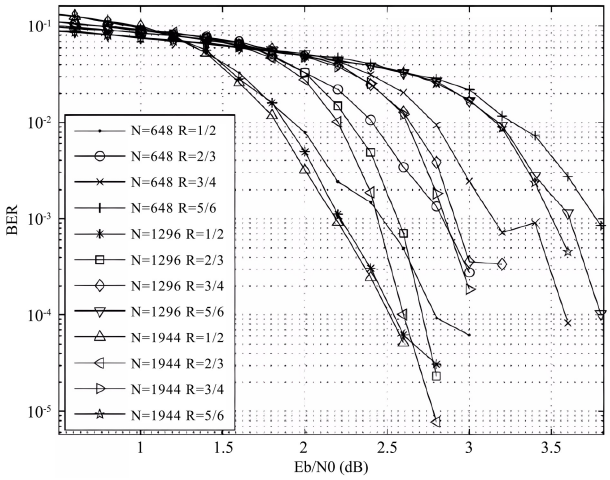

Fig. 17. Performance evaluation of LDPC coding. In figure, N is the code length, R is the code rate, and BER is bit error ratio.

output sequence of convolutional code bit stream blocks, and the sequence in the figure is filled bits on the horizontal axis, except for the first bit.

Fig. 17 shows the BER curve of LDPC coding. Since we are using a 5 bits fixed-point quantization, the error floor is approximately 10^{-6}. Different K values are selected to measure the BER, and all LDPC codes of code length converge to 0 within Eb/N0 of 4 dB.

Through Synopsys synthesis, the layout is generated, as shown in Fig. 18, and the layout related parameters are shown in Table V. The technology directly affects SRAM size, there are an inverse proportional relationship between them. Voltage and frequency indirectly affect SRAM size, where voltage has a positive proportional relationship with SRAM size, and frequency has an inverse proportional relationship with SRAM size. The technology, voltage, frequency, and SRAM size all directly influence power consumption, there is an inverse proportional relationship between the technology and power consumption, while voltage, frequency, and SRAM size all have positive proportional relationships with power consumption.

For the algorithm and hardware circuit proposed in this article, the hardware simulation is verified by Modelsim.

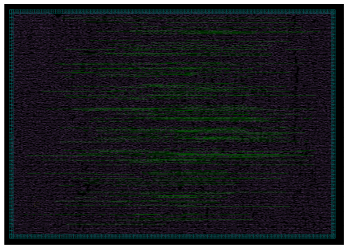

Fig. 18. Chip layout based on 28-nm technology.

TABLE V
LAYOUT PARAMETERS

Parameters	Value
Technology	28nm
Voltage	0.9V
Frequency	50MHz
SRAM size	0.1μm^2
Power consumption	4.31mW

The results show that the address calculation formula can realize the correct output of the bit stream after interleaving. According to the waveform diagram, when the parallelism degree is increased in the interleaver, the output waveform can be observed, and the corresponding bit sequence can be correctly output at the rising edge of the clock.

B. Experimental Comparison

The design proposed in this article maximizes hardware sharing of rate matching and interleaving of the four codes under 4G LTE and 5G NR/6G. We compare our design with the state-of-the-art rate matching and interleaving designs [27], [28], [29], [30], [31]. We implement our design and the state-of-the-art designs in a 28-nm CMOS technology. In the experiments, the clock frequency is 50 MHz, the voltage is 0.9 V, the code length is 4608, the code rate is 1/3 and the number of punching bits is 864. In the same communication standards, clock frequency, voltage, code length, code rate and number of punching bits, the experimental comparison results are shown in Table VI. The state-of-the-art rate matching and interleaving designs [27], [28], [29], [30], [31] and our design have the same final error performance, the final error rate is about 10^{-5}. Compared with the rate matching and interleaving designed in article [27], the experimental results show that our design reduces memory area and power consumption by about 47% and 44%, respectively, and improves the throughput of turbo code, LDPC code, convolutional code and polar code by about 37%, 35%, 38%, and 40%, respectively. Compared with the rate matching and interleaving designed in article [28], our design reduces memory area and power consumption by about 23% and 27%, respectively, and improves the throughput of turbo code, LDPC code, convolutional code and polar code by about 41%, 39%, 42%, and 41%, respectively. Compared with the rate matching and interleaving designed in article [29], our design reduces memory area and power consumption by 38% and 31%, respectively, and improves the throughput of turbo code, LDPC code, convolutional code and polar code by about

TABLE VI
COMPARISON OF RATE MATCHING AND INTERLEAVING DESIGN PROPOSED IN THIS ARTICLE AND THE STATE-OF-THE-ART RATE MATCHING AND INTERLEAVING DESIGNS

Methods	Memory area	Power consumption	Throughput
[27]	0.19μm^2	7.63mW	730Mbps for turbo code, 1560Mbps for LDPC code, 726Mbps for convolutional code, 709Mbps for polar code
[28]	0.13μm^2	5.91mW	710Mbps for turbo code, 1510Mbps for LDPC code, 703Mbps for convolutional code, 700Mbps for polar code
[29]	0.16μm^2	6.29mW	760Mbps for turbo code, 1620Mbps for LDPC code, 780Mbps for convolutional code, 746Mbps for polar code
[30]	0.2μm^2	8.01mW	850Mbps for turbo code, 1730Mbps for LDPC code, 839Mbps for convolutional code, 803Mbps for polar code
[31]	0.17μm^2	6.45mW	800Mbps for turbo code, 1660Mbps for LDPC code, 810Mbps for convolutional code, 765Mbps for polar code
Ours	**0.1μm^2**	**4.31mW**	**1000Mbps for turbo code, 2100Mbps for LDPC code, 1000Mbps for convolutional code, 990Mbps for polar code**

32%, 30%, 28%, and 33%, respectively. Compared with the rate matching and interleaving designed in article [30], our design reduces memory area and power consumption by 50% and 46%, respectively, and improves the throughput of turbo code, LDPC code, convolutional code and polar code by about 18%, 21%, 19%, and 23%, respectively. Compared with the rate matching and interleaving designed in article [31], our design reduces memory area and power consumption by 41% and 33%, respectively, and improves the throughput of turbo code, LDPC code, convolutional code and polar code by about 25%, 27%, 23%, and 29%, respectively. Experimental results show that the total area and power consumption of our design are relatively small, the system throughput is relatively large, and our design can be switched between multiple interleaving, to a certain extent, it improves the flexibility of the system. The experimental results indicate that our design shares the address computing unit, and can switch back and forth between the four codes, which improves the flexibility of the communication system. In addition, our design realizes the sharing of memory in four codes, which can effectively reduce the silicon area and power consumption.

In addition, based on our proposed parallel interleaving method and the corresponding parallel conflict-free addressing method, other use case of our designed hardware is that it can be applied to IIoT wireless terminals to further reduce the encoding latency. We compare the encoding latency in IIoT wireless terminal based on different rate matching and interleaving hardware. We also implement our designed hardware and the state-of-the-art rate matching and interleaving hardware [32], [33], [34] in a 28-nm CMOS technology. In the experiments, the clock frequency is 50 MHz, the voltage is 0.9 V, the code length is 4608, the code rate is 1/3 and the number of punching bits is 864. In the same clock frequency, voltage, code length, code rate, number of punching bits, encoding algorithm of turbo code, LDPC code, convolutional code and polar code, the experimental comparison results are shown in Fig. 19. Compared with the state-of-the-art rate matching and interleaving hardware designed in [32], the experimental results show that our designed hardware reduces the encoding latency of turbo code, LDPC code, convolutional code and polar code by about 28%, 30%, 26%, and 31%, respectively. Compared with the state-of-the-art rate matching and interleaving hardware designed in [33], the experimental results show that our designed hardware reduces the encoding latency of turbo code, LDPC code, convolutional code and polar code by about 23%, 25%, 22%, and 23%, respectively. Compared with the state-of-the-art rate matching and interleaving hardware designed in [34], the experimental results show that our designed hardware reduces the encoding latency of turbo code, LDPC code, convolutional code and polar code by about 20%, 23%, 21%, and 20%, respectively. Experimental results show that our designed rate matching and interleaving hardware can effectively reduce the encoding latency.

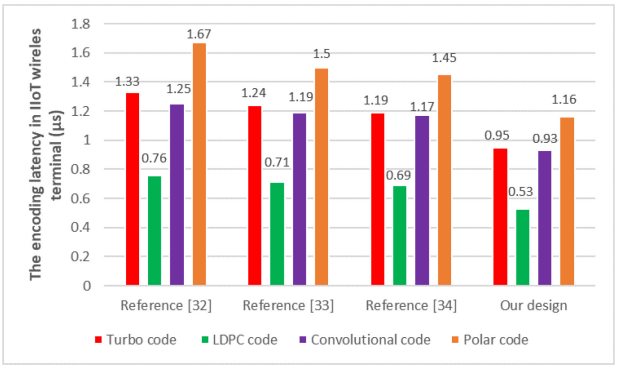

Fig. 19. Comparison of the encoding latency in IIoT wireless terminal based on different rate matching and interleaving hardware.

The data and control channels in 5G NR employ different channel interleavers to reduce the BER. However, the independent implementation results in a substantial increase in silicon cost. Technology merging is pivotal for optimizing resource utilization and reducing latency caused by frequent switching between multiple hardware [6]. To address this challenge, a shared-resource channel interleaver was designed to merge two interleaving techniques [6]. Upadhyaya et al. [10] designed a MIMO WLAN interleaver to accommodate more modulation schemes. These works merged various technologies to reduce hardware overhead, power consumption and

latency that caused by frequent switching between multiple hardware [6], [10]. Our designed hardware also has the same goals. For example, a customer specific use case of our designed hardware is that our design can facilitate reliable interconnection and communication of multiple industrial equipment with low latency and low overhead. Industrial equipment sometimes requires low-latency communication and low-latency data transmission in IoT scenarios, our hardware can achieve rate matching and interleaving for convolutional codes, which can quickly and flexibly adjust the code rate to ensure the data transmission rate according to channel conditions and data size. Industrial equipment sometimes requires highly reliable data transmission in IoT scenarios, our hardware can achieve rate matching and interleaving for turbo codes, which can improve the ability to resist against noise, thereby enhancing the reliability of data transmission. Industrial equipment sometimes requires high-data transmission quality and high-data transmission rate in IoT scenarios, our hardware can achieve rate matching and interleaving for polar codes, which can achieve high-data transmission rate while maintaining high-error correction capabilities, thereby achieving both high-data transmission quality and high-data transmission rate. Industrial equipment sometimes needs to transmit large amounts of data and has requirements for enhanced mobile broadband (eMBB) in IoT scenarios, our hardware can achieve rate matching and interleaving for LDPC codes, which can improve data transmission efficiency, thereby improving data throughput. Our design uses the same set of hardware to achieve multistandards rate matching and interleaving, which can provide services that combine ultralow latency, lower power consumption, high-data transmission rate, and highly reliable massive data transmission, it promotes seamless and highly reliable interconnection and communication of multiple industrial equipment to enhance industrial automation system. If industrial equipment uses different rate matching and interleaving hardware, which will cause waste of hardware resources, increase power consumption, and increase latency due to frequent switching between different rate matching and interleaving hardware. It cannot meet the requirements of quickly establishing persistent interconnection of multiple industrial equipment with limited power.

VI. Conclusion

For the IIoT wireless terminal, this article presents the hardware fusion and multiplexing design of rate matching and interleaving for the information processing of four kinds of encoding in the communication downlink, this hardware implementation saves silicon area and reduces power consumption. Moreover, we use the same set of hardware to achieve multistandards rate matching and interleaving, and flexible switching under multistandards. To a certain extent, our design improves the flexibility and reconfiguration of the IIoT wireless terminal. At the same time, a parallel algorithm of interleaving is proposed to improve the degree of interleaving parallelism, so as to reduce the delay and improve the throughput of the system. In the future, with the update of mobile communication standards, we will integrate rate matching and interleaving algorithms based on the future communication standards into our designed hardware to better serve the future IIoT.

References

[1] A. K. Mandal and S. De, "Analysis of wireless communication over electromagnetic impulse noise channel," *IEEE Trans. Wireless Commun.*, vol. 22, no. 2, pp. 1187–1200, Feb. 2023.

[2] K. C. Behera, "An efficient low-latency algorithm and implementation for rate-matching and bit-interleaving in 5G NR," in *Proc. IEEE 3rd 5G World Forum (5GWF)*, 2020, pp. 565–571.

[3] X. Zhao, W. Chen, and H. V. Poor, "Achieving extremely low-latency in Industrial Internet of Things: Joint finite blocklength coding, resource block matching, and performance analysis," *IEEE Trans. Commun.*, vol. 69, no. 10, pp. 6529–6544, Oct. 2021.

[4] V. K. Perumal, K. Mahalingam, and D. P. Raja, "FPGA implementation of reconfigurable address generator for various standard interleaver," in *Proc. Int. Conf. Intell. Innov. Technol. Comput., Electr. Electron. (IITCEE)*, 2023, pp. 472–477.

[5] V. Akshaya, K. N. Sreehari, and A. Chalil, "VLSI implementation of turbo coder for LTE using verilog HDL," in *Proc. 4th Int. Conf. Comput. Methodol. Commun. (ICCMC)*, 2020, pp. 275–279.

[6] J. L. Lakshmi and J. Jayakumari. "A reduced complexity rate-matching and channel interleaver/de-interleaver for 5G NR," *Eng. Res. Exp.*, vol. 6, no. 2, Apr. 2024, Art. no. 25301.

[7] M. Zhan, Z. Pang, D. Dzung, K. Yu, and M. Xiao, "Short-packet interleaver against impulse interference in practical industrial environments," *IEEE Trans. Wireless Commun.*, vol. 21, no. 12, pp. 10257–10270, Dec. 2022.

[8] S. Dixit, V. Shukla, M. K. Misra, J. M. Jimenez, and J. Lloret, "Progressive pattern interleaver with multicarrier modulation schemes and iterative multiuser detection in IoT 6G environments with multipath channels," *Sensors*, vol. 24, no. 11, p. 3648, Jun. 2024.

[9] X Xiong et al., "Hardware sharing for channel interleavers in 5G NR standard," *Secur. Commun. Netw.*, vol. 2021, pp. 1–13, Jan. 2021.

[10] B. K. Upadhyaya, P. K. D. Pramanik, and S. K. Sanyal, "High throughput resource efficient reconfigurable interleaver for MIMO WLAN application," *PeerJ Comput. Sci.*, vol. 7, p. e581, Jun. 2021.

[11] Y. Hu, C. Liang, L. Liu, C. Yan, Y. Yuan, and L. Ping, "Interleave-division multiple access in high rate applications," *IEEE Wireless Commun. Lett.*, vol. 8, no. 2, pp. 476–479, Apr. 2019.

[12] A. Suls, Y. Lefevre, J. Van Hecke, M. Guenach, and M. Moeneclaey, "Error performance prediction of randomly shortened and punctured LDPC codes," *IEEE Commun. Lett.*, vol. 23, no. 4, pp. 560–563, Apr. 2019.

[13] S. Saha and M. Adrat, "Novel multiparameter based rate-matching of polar codes," in *Proc. Int. Conf. Mil. Commun. Inf. Syst. (ICMCIS)*, 2021, pp. 1–8.

[14] T. Son and S. Kim, "An efficient PLS scheme for bit interleaved coded MIMO systems," in *Proc. IEEE Int. Conf. Consum. Electron. Asia (ICCE-Asia)*, 2022, pp. 1–3.

[15] B. Y. Kong and I.-C. Park, "Parallel IDMA architecture based on interleaving with replicated subpatterns," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Shanghai, China, 2019, pp. 1–6.

[16] Z. Deng, Y. Zhang, N. Li, and Q. Du, "A novel multilevel interleaver model in interleave division multiple access," in *Proc. IEEE 19th Int. Conf. Commun. Technol. (ICCT)*, Xi'an, China, 2019, pp. 30–33.

[17] Y. Feng, H. Deng, Q. Fan, R. Zhang, P. Bikkina, and J. Chen, "A 6-b 20-GS/s 2-way time-interleaved flash ADC with automatic comparator offset calibration in 28-nm FDSOI," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Seville, Spain, 2020, pp. 1–4.

[18] Y. Fang, Y. Bu, P. Chen, F. C. M. Lau, and S. A. Otaibi, "Irregular-mapped protograph LDPC-coded modulation: A band-width-efficient solution for 6G-enabled mobile networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 2060–2073, Feb. 2023.

[19] Q. Wang, P. Fu, and S. Zhang, "A comparison of concatenated polar codes with different interleaving and decoding schemes," in *Proc. 5th Int. Conf. Comput. Commun. Syst. (ICCCS)*, Shanghai, China, 2020, pp. 570–574.

[20] A. K. Ahmed and H. S. Al-Raweshidy, "Deep learning polar convolutional parallel concatenated (DL-PCPC) channel de-coding for 6G communications," in *Proc. Int. Conf. Comput., Inf. Telecommun. Syst. (CITS)*, Genoa, Italy, 2023, pp. 1–5.

[21] C. Kestel, M. Geiselhart, L. Johannsen, S. Ten Brink, and N. Wehn, "Automorphism ensemble polar code decoders for 6G URLLC," in *Proc. 26th Int. ITG Workshop Smart Antennas 13th Conf. Syst., Commun., Coding*, Braunschweig, Germany, 2023, pp. 1–6.
[22] A. Sahnoune, D. Berkani, and E. Zeraoulia, "Secure turbo code design based on chaotic interleavers," in *Proc. 2nd Int. Conf. Electron., Energy Meas. (IC2EM)*, Medea, Algeria, 2023, pp. 1–5.
[23] S. Jihwan and H.-K. Lee, "Burst Error Correction for Convolutional Code Concatenated with Hamming code with a block interleaver," in *Proc. Int. Conf. Artif. Intell. Inf. Commun. (ICAIIC)*, Fukuoka, Japan, 2020, pp. 531–533.
[24] *Multiplexing and Channel Coding, Version 15.4.0*, 3GPP Standard TS 36.212, 2018.
[25] J. H. Bae, A. Abotabl, H.-P. Lin, K.-B. Song, and J. Lee, "An overview of channel coding for 5G NR cellular communications," *APSIPA Trans. Signal Inf. Process.*, vol. 8, p. e17, Jun. 2019.
[26] W. Chen and D. Liu, "Conflict-free parallel data access technology for matrix calculation in memory system of ASIP of 5G/6G macro base stations," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 43, no. 1, pp. 99–112, Jan. 2024.
[27] R. Liu, Z. Zhong, X. Sun, and H. Liu, "Interlace type recognition analysis under noncooperative conditions," in *Proc. Int. Conf. Commun., Inf. Syst. Comput. Eng. (CISCE)*, Haikou, China, 2019, pp. 144–147.
[28] R. D. Cideciyan, S. Furrer, and M. A. Lantz, "Performance of interleaved block codes with burst errors," *IEEE Trans. Magn.*, vol. 55, no. 3, pp. 1–5, Mar. 2019.
[29] R. Gogia, B. Lall, and S. D. Joshi, "Fast algorithm for blind deinterleaving of a block interleaver using binary and nonbinary Block codes in a telecommunication system," in *Proc. IEEE Int. Conf. Adv. Netw. Telecommun. Syst. (ANTS)*, 2018, pp. 1–6.
[30] T. Aktas and P. Sen, "Interleaved block coding for achieving Gaussian random access channel capacity," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Los Angeles, CA, USA, 2020, pp. 3007–3012.
[31] K. Huo, Z. Hu, and D. Liu, "Design and implementation of shared memory for turbo and LDPC code interleaver," *Wireless Commun. Mobile Comput.*, vol. 2022, no. 1, Feb. 2022, Art. no. 5782199.
[32] K. Lai, L. Wen, J. Lei, G. Chen, P. Xiao, and A. Maaref, "Secure transmission with interleaver for uplink sparse code multiple access system," *IEEE Wireless Commun. Lett.*, vol. 8, no. 2, pp. 336–339, Apr. 2019.
[33] V. K. Perumal, V. Rampandian, and P. V. A. Devikamatchi, "FPGA implementation of reconfigurable address generator for various standard interleaver in Xilinx FPGA," in *Proc. 2nd Int. Conf. Intell. Data Commun. Technol. Internet Things (IDCIoT)*, 2024, pp. 236–241.
[34] J. Ding, J. Su, C. Li, G. Ren, and H. Wang, "Random interleaving multiplexing based IRSA random access system for satellite-enabled Internet-of-Things," *IEEE Access*, vol. 9, pp. 143093–143103, 2021.

Wei Chen received the B.Eng. and M.S. degrees from East China Normal University, Shanghai, China, in 2014 and 2017, respectively, and the Ph.D. degree from Hainan University, Haikou, China, in 2023.

He is currently a Lecturer with the School of Information and communication Engineering, Hainan University. His current research interests include application specific instruction-set processors design, integrated circuit design, CAD, and Internet of Things.

Kejia Huo received the B.Eng. degree from Zhengzhou University, Zhengzhou, China, in 2019, and the M.Eng. degree from Hainan University, Haikou, China, in 2022.

Her current research interests include integrated circuit design, signal, and information processing.

Dake Liu (Senior Member, IEEE) received the Tech.Dr. degree from Linköping University, Linköping, Sweden, in 1995.

He is a Professor with the School of Information and Communication Engineering, Hainan University, Haikou, China, and a Professor with Beijing Institute of Technology, Beijing, China, and also a Professor with the Computer Engineering Division, Department of Electrical Engineering, Linköping University. His current research interests include integration of on-chip multiprocessors for communications and integrated circuit design.