Visvesvaraya Technological University, Belagavi.

PROJECT REPORT

on

EFFICIENT QC-LDPC ENCODER FOR 5G NEW RADIO

Project Report submitted in partial fulfillment of the requirement for the award of the degree of

Bachelor of Engineering

in

Electronics and Communication Engineering

For the academic year 2019-20

Submitted by

USN	Name
1CR16EC015	ANTO ALEX B
1CR16EC048	HARSHITH B MUTHANNA
1CR16EC061	KHAJA WASEEM AKRAM
1CR16EC063	KONDURU SAI KISHORE

Under the guidance of

Monika Singh
Assistant Professor
Department of ECE
CMRIT, Bangalore

Department of Electronics and Communication Engineering

CMR Institute of Technology, Bengaluru – 560 037

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to Certify that the dissertation work **“EFFICIENT QC-LDPC ENCODER FOR 5G NEW RADIO”** carried out by ANTO ALEX B USN:1CR16EC015, HARSHITH B MUTHANNA USN:1CR16EC048, KHAJA WASEEM AKRAM USN: 1CR16EC061, KONDURU SAI KISHORE USN:1CR16EC063 bonafide students of **CMRIT** in partial fulfillment for the award of **Bachelor of Engineering** in **Electronics and Communication Engineering** of the **Visvesvaraya Technological University, Belagavi,** during the academic year **2019-20**. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said degree.

Signature of Guide	Signature of HOD	Signature of Principal
Monika Singh Assistant Professor, Dept. of ECE., CMRIT, Bengaluru.	Dr. R. Elumalai Head of the Department, Dept. of ECE., CMRIT, Bengaluru.	Dr. Sanjay Jain Principal, CMRIT, Bengaluru.

External Viva

Name of Examiners Signature & date

1.

2

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose consistent guidance and encouragement crowned our efforts with success.

We consider it as our privilege to express the gratitude to all those who guided in the completion of the project.

We express my gratitude to Principal, **Dr. Sanjay Jain,** for having provided me the golden opportunity to undertake this project work in their esteemed organization.

We sincerely thank Dr. R. Elumalai**,**HOD**,** Department of Electronics and Communication Engineering, CMR Institute of Technology for the immense support given to me.

We express my gratitude to our project guide Monika Singh, Assistant Professor, for their support, guidance and suggestions throughout the project work.

Last but not the least, heart full thanks to our parents and friends for their support.

Above all, we thank the Lord Almighty for His grace on us to succeed in this endeavor.

ABSTRACT

This Work presents a novel efficient encoding method and high-throughput low-complexity encoder architecture for quasi-cyclic low-density parity-check (QC-LDPC) codes for the 5th-generation (5G) New Radio (NR) standard.

By storing the quantized value of the permutation information for each sub matrix instead of the whole parity check matrix, the required memory storage size is considerably reduced. In addition, sharing techniques are employed to reduce the hardware complexity.

The encoding complexity of the proposed method was analyzed, and indicated a substantial reduction in the required area as well as memory storage when compared with existing state-of-the-art encoding approaches.

The proposed method requires only 61% gate area, and 11%ROM storage when compared with a similar LDPC encoder using the Richardson–Urbanke method.

TABLE OF CONTENTS

CHAPTER-1

CHAPTER-2

CHAPTER-3

CHAPTER-4

CHAPTER-5

CHAPTER-6

LIST OF FIGURES

Chapter 1

INTRODUCTION

Low-density parity-check (LDPC) codes, which were first proposed by Gallager in the early 1960s and rediscovered by MacKay and Neal in 1996, have attracted widespread attention thanks to their remarkable error correction capabilities near the Shannon limit, with advancements in very large-scale integration (VLSI). Moreover, LDPC codes are among the most widely used types of forward error correction (FEC) codes in several communications standards such as the wireless local area network (WLAN, IEEE 802.11n), wireless radio access network (WRAN, IEEE 802.22), digital video broadcast (DVB), and the Advanced Television System Committee (ATSC). Recently, the fifth generation (5G) communication has been a hotspot of research and development. More specially, LDPC codes play an important role in 5G communication and have been selected as the coding scheme for the 5G enhanced Mobile Broad Band (eMBB) data channels. To support compatible rate and scalable data transmission, 3rd Generation Partnership Project (3GPP) has agreed to consider two rate-compatible base graphs, BG1 and BG2, for the channel coding. Accordingly, several studies have been conducted on the 5G LDPC codes. Over recent years, research on LDPC codes has been focused on structured LDPC codes known as quasi-cyclic low-density parity-check (QC-LDPC)

Codes, which exhibit advantages over other types of LDPC codes with respect to the hardware implementations of encoding and decoding using simple shift registers and logic circuits. A low-complexity encoder can be realized by using QC-LDPC codes, due to the sparseness of the parity check matrix. However, it is not straightforward to encode with low complexity as LDPC codes are defined by their parity check matrix, and the generator matrix is generally unknown. Various approaches have been suggested to improve the hardware complexity of LDPC encoders. One of the most conventional approaches is systematic encoding, in which the generator matrix is derived from the parity check matrix by exploiting Gaussian elimination. The main drawback related to this method is that the storage overhead is dramatically increased for large block sizes, which limits its practical applicability. The Richardson–Urbanke (RU) algorithm is a widely-used LDPC codes encoding scheme developed by Richardson and Urbanke. The underlying principle of the method is the transformation of the parity check matrix into an approximate lower triangular (ALT) form by using only row and column permutations, which preserves the sparseness of the matrix. This method suffers from a long critical path, which could make the LDPC encoder unsuitable for high throughput applications. To overcome the limitations of the previous approaches, the design proposed in this paper, which is referred to as a low-complexity high-throughput LDPC encoder architecture for the 5G standard, requires significantly less area and memory storage while maintaining a high throughput. This work targets the design of low-complexity high-throughput QC-LDPC encoders for the 5G NR standard. In LDPC encoders, the memory and interconnecting blocks are considered as the major influencing factors of the overall area, delay, and power performance of the hardware design. Hence, the size of the read only memory (ROM) was decreased by storing the quantized value of the permutation information for each sub matrix instead of the entire parity check matrix H. The proposed architecture requires less matrix multiplications than the RU method, by exploiting the characteristics of the 5G NR base matrix. In addition, the proposed algorithm does not require the inverse of the component matrix, which presents a primary advantage over the RU method. Moreover, block-memories are not required to store the generator matrix G, and the number of required components is reduced. The ROM size of the proposed method is 98.2% and 88.9% lower than those of the G matrix method and RU method, respectively.

Chapter 2

METHODOLOGY

2.1 **Linear Block Codes** :

Error correcting codes attach extra bits to the transmitted data. The extra bits are the redundancy which is then used to detect and correct errors on the received data. In block coding, the transmitted data is segmented into blocks of fixed length of K bits. Linear block codes are a special class of block codes where each bit or symbol can be expressed as a linear combination of other bits or symbol in the transmitted data. The encoder then based on certain rules transforms the input segment into an output block of length N. (i.e.)N > M ,thus providing the redundancy needed for error correction and detection .The rate of the code is expressed as R=M/N .With M bits in an input message ,there are 2^M distinct input entries possible. Each output message of N bits associated with each input message is called a code word. With input message of M bits ,an arbitrary encoding would require the encoder to store a table of 2^M entries each of length N. Linear block code reduce the complexity of encoding by using a linear generator matrix to transform inputs to codeword . The generator matrix is made up of linearly independent row vectors g_1, g_2……g_R, such that it can be expressed as

$$G = \begin{pmatrix} g_1 \\ g_2 \\ . \\ . \\ . \\ . \\ g_R \end{pmatrix}$$

The encoder generates the code word by multiplying the input vector with the generator matrix, c=uG, where c is a codeword and u is the input vector of bits. The encoder has to store only G, thus reducing the space complexity from 2^M×N(arbitrary encoding) to M × N(size of G).From the generator matrix G, a parity check matrix H can be derived .The matrixes are related by GH^T =0.When the decoder receives a word, it checks using H if the word is indeed a codeword.

2.2 Low density parity check codes:

In informationtheory, a **low-density parity-check** (**LDPC**) **code** is a linear error correcting code, a method of transmitting a message over a noisy transmission channel. An LDPC is constructed using a sparse Tanner graph (subclass of the bipartite graph). LDPC codes are capacity-approaching codes, which means that practical constructions exist that allow the noise threshold to be set very close (or even *arbitrarily* close on the binary erasure channel) to the theoretical maximum (the Shannon limit) for a symmetric memoryless channel. The noise threshold defines an upper bound for the channel noise, up to which the probability of lost information can be made as small as desired. Using iterative belief propagation techniques, LDPC codes can be decoded in time linear to their block length.LDPC codes are finding increasing use in applications requiring reliable and highly efficient information transfer over bandwidth-constrained or return-channel-constrained links in the presence of corrupting noise. Implementation of LDPC codes has lagged behind that of other codes, notably turbo codes. The fundamental patent for Turbo Codes expired on August 29, 2013.LDPC codes are also known as **Gallager codes**, in honor of Robert G. Gallager, who developed the LDPC concept in his doctoral dissertation at the Massachusetts Institute of Technology in 1960.

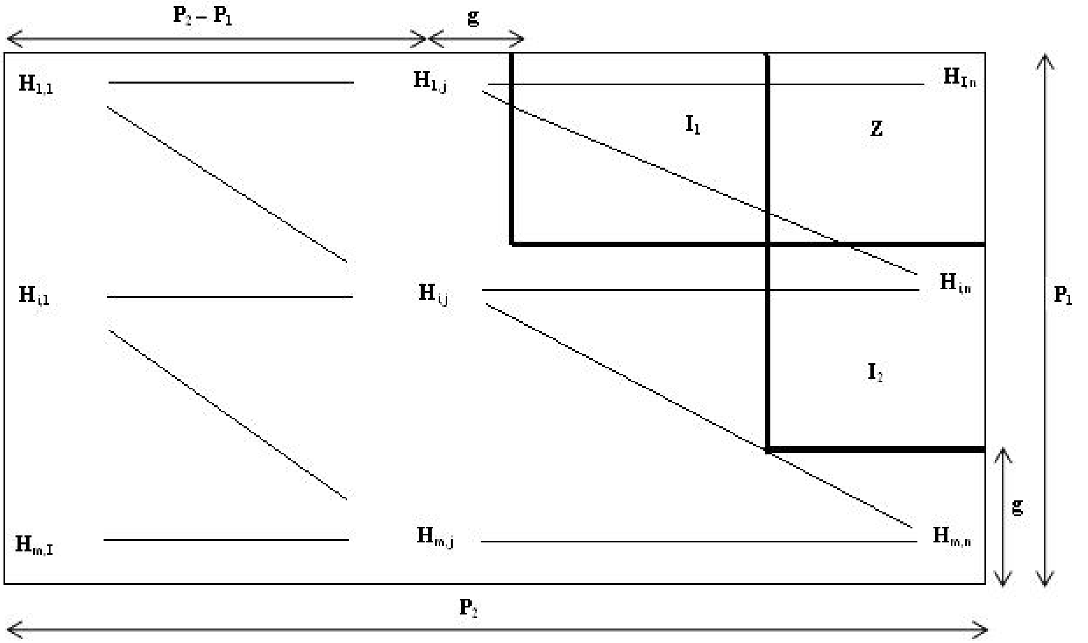


Figure : Parity check matrix structure

Here we are considering (3, 6) parity check matrix, where P_1= 6, P_2 =12 and g=2 where ‘g’ represents the gap value. The ‘H’ matrix chosen in this work is shown in Figure below.

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Figure:Paritycheckmatrix‘H’

Normally LDPC codes are represented using bipartite and tanner graph (Ahmadi et al 2012). But by using the method of back substitution in this LDPC construction, the encoding complexity is reduced by transforming the parity check matrix into upper triangular form (Richardson and Urbanke 2001). The idea is to do as much of the transformation as possible by only row and column permutations to keep the 'H' as sparse. The 'H' matrix is sub divided into six blocks are shown below, using the blocks the messagesignalsareencoded.

$$H = \left[\begin{array}{c|c|c} A & B & C \\ \hline D & E & F \end{array}\right]$$

The different block sizes in 'H' are designed by the following procedure A= $[P_1\text{-}g] \times [P_2\text{-}P_1] = [6\text{-}2] \times [12\text{-}6] = 4 \times 6$ (4 rows 6 columns)

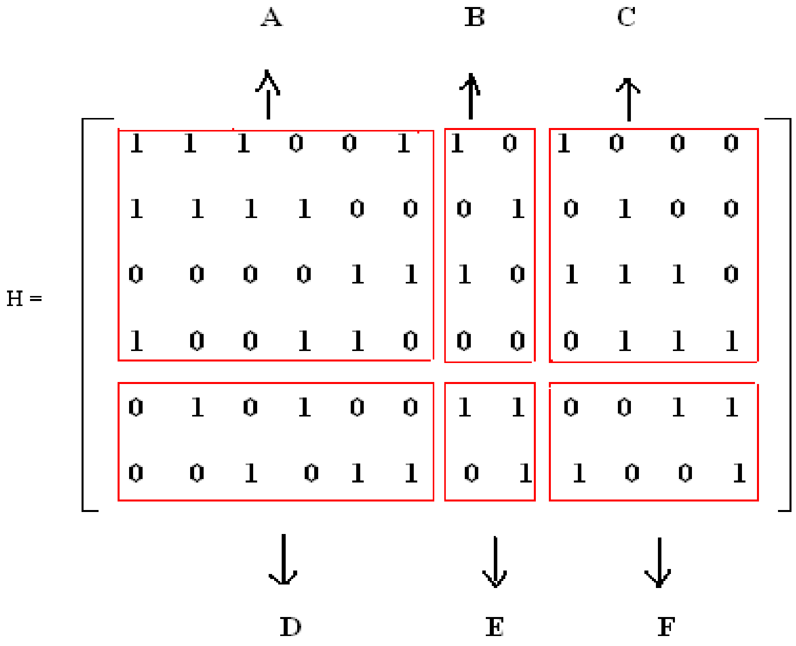
B = $[P_1\text{-}g] \times g = [6\text{-}2] \times 2 = 4 \times 2$ (4 rows 2 columns) C block is an upper triangular matrix where, $C = C^{-1}$

C = $[P_1\text{-}g] \times [P_1\text{-}g] = [6\text{-}2] \times [6\text{-}2] = 4 \times 4$ (4 rows 4 columns)

D = $g \times [P_2\text{-}P_1] = 2 \times [12\text{-}6] = 2 \times 6$ (2 rows 6 columns)

E = $g \times g = 2 \times 2$ (2 rows 2 columns)

$F = g \times [P_1\text{-}g] = 2 \times [6 - 2] = 2 \times 4$ (2 rows 4 columns)



$$H = \left[\begin{array}{cccccc|cc|cccc}
1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
\hline
0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1
\end{array}\right]$$

Figure:EncodedmessagebitsofLDPC

The generated code word is obtained by

$[S_1, R_1, R_2]$ S_1 Informationbit

R_1→ Parity bit, calculated using the formula

$R_1{=}FCAS_1^T$

R_2→ Parity bit, calculated using the formula

$R_2{=}[C\,(AS_1^T) + BR_1^T]$

Let the information bit be $S_1 = (1, 0, 0, 0, 0, 0)$

$R_1 = FCAS_1^T = (0, 1)$

$R2 = [C\,(AS_1^T) + BR_1^T] = (1, 0, 1, 0)$

$$\text{Encoded data} = [S_1, R_1, R_2]$$

$$= [\ 1, 0, 0, 0,\ \ 0,\ \ 0,\ \ 0,\ \underbrace{1,}\ 1,\ \ 0,\ \ 1,\ \ 0\ ]$$

$$S_1 \qquad\qquad R_1 \qquad\qquad R_2$$

2.3 QC-LDPC Codes:

The encoding of LDPC is based on parity check matrix 'H' where the QCLDPC codes can be constructed by applying circulant matrixes (Spagnol and Marnane 2009). The circulant matrix 'C_M' is shown in Figure 4.

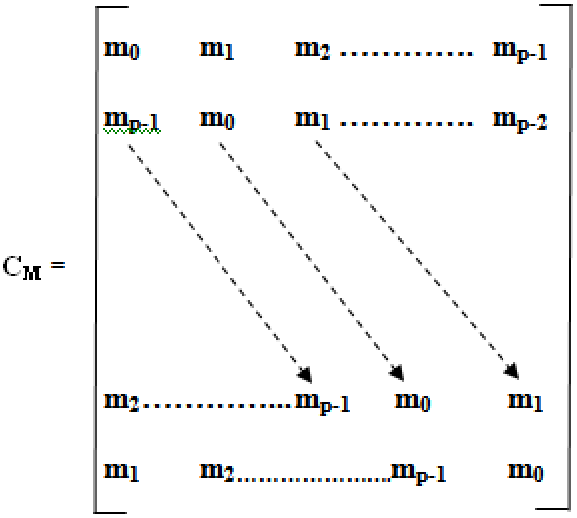

C_M =
m_0 m_1 m_2 .... m_{p-1}
m_{p-1} m_0 m_1 .... m_{p-2}
m_2............. m_{p-1} m_0 m_1
m_1 m_2................. m_{p-1} m_0

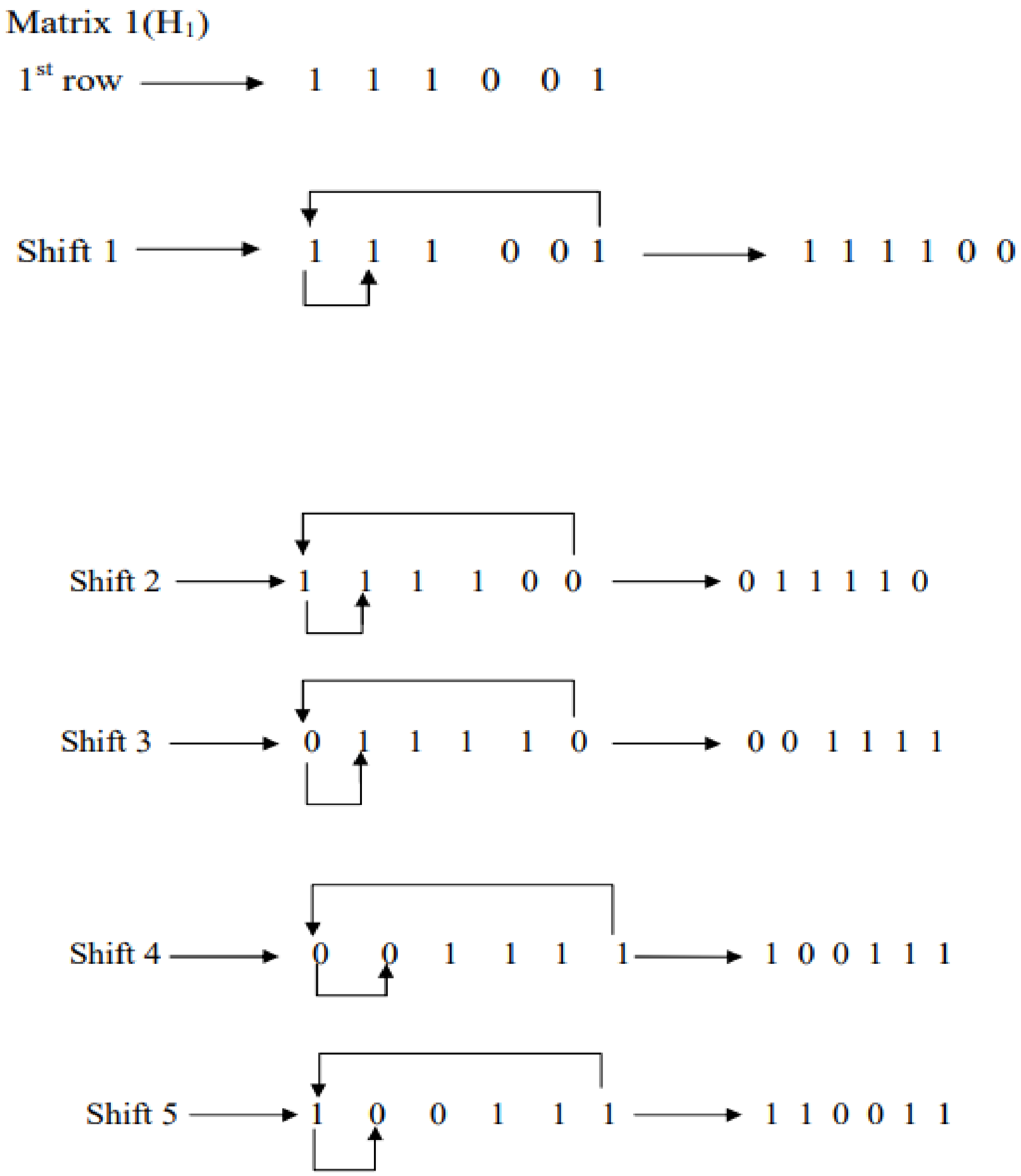
Figure : General circulant matrix structure

Based on the circulant matrix the parity check matrix 'H' is subdivided into two matrix maintaining the equal row and column length (Marc Fossorier,2004).The partitioned H matrix is shown in Figure.

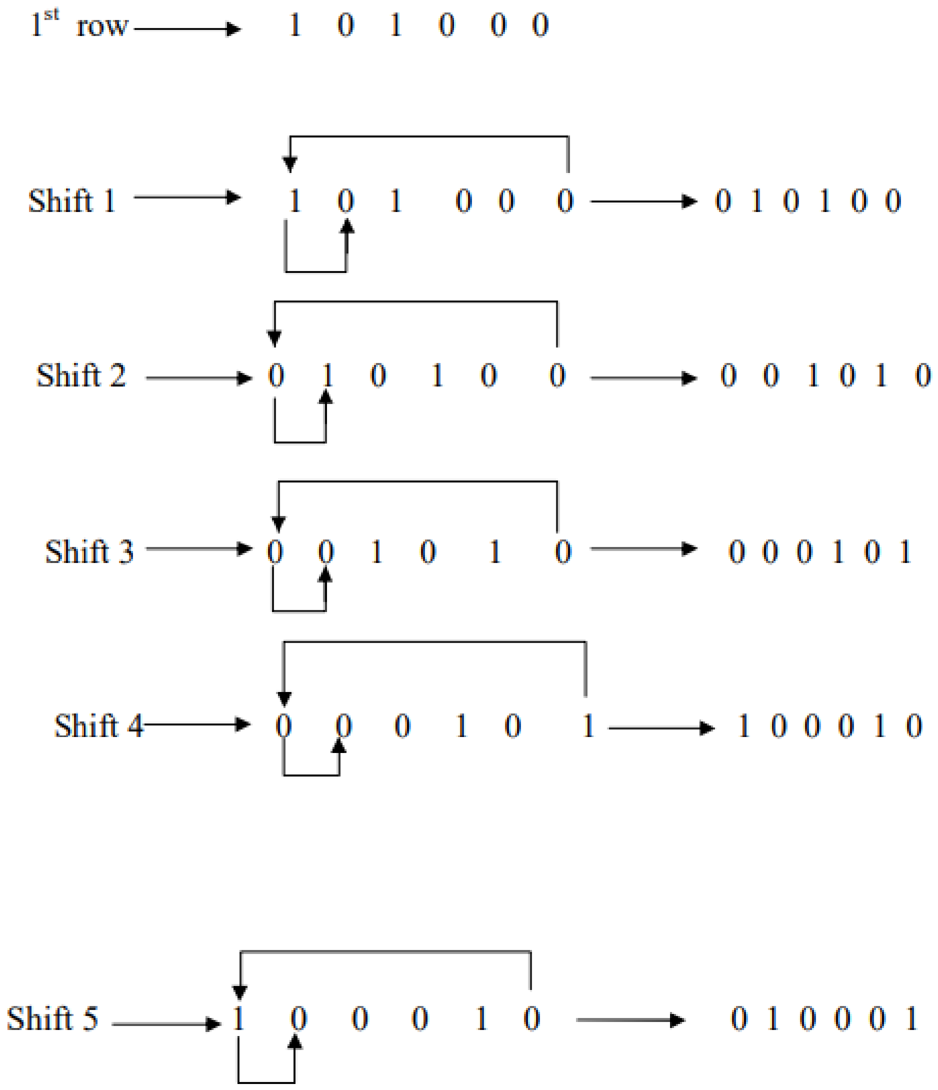
$$H = \overbrace{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}}^{1^{st}\ \text{Matrix }(H_1)} \overbrace{\begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}}^{2^{nd}\ \text{Matrix}(H_2)}$$

Figure 4.6 Partitioned 'H' matrix

The circulant concept in 'H' matrix defines that each row in the above two matrix is one time right cyclic shift of the previous one, since each column and row is a shift of the previous column and row, where the column and row weight is uniform for all the shift in the 'H' circulant matrix(Spagnol and Marnane 2009). The generator matrix is obtained while defining the 'H' matrix as $GH^T= 0$ .The bits are encoded through the generator matrix G and the codeword should obey the property $CH^T= 0$, where 'C' is a codeword. The explanation for the shifting is givenbelow.


Matrix 1(H1)
1st row → 1 1 1 0 0 1
Shift 1 → 1 1 1 0 0 1 → 1 1 1 1 0 0
Shift 2 → 1 1 1 1 0 0 → 0 1 1 1 1 0
Shift 3 → 0 1 1 1 1 0 → 0 0 1 1 1 1
Shift 4 → 0 0 1 1 1 1 → 1 0 0 1 1 1
Shift 5 → 1 0 0 1 1 1 → 1 1 0 0 1 1

Matrix 2 (H_2)


1st row → 1 0 1 0 0 0
Shift 1 → 1 0 1 0 0 0 → 0 1 0 1 0 0
Shift 2 → 0 1 0 1 0 0 → 0 0 1 0 1 0
Shift 3 → 0 0 1 0 1 0 → 0 0 0 1 0 1
Shift 4 → 0 0 0 1 0 1 → 1 0 0 0 1 0
Shift 5 → 1 0 0 0 1 0 → 0 1 0 0 0 1

The above operation is done for all the rows of matrix 1 and matrix 2. After the completion of circulant shift procedure, the obtained weightsofthematrixis

$$H = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$H = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

The circulant parity check matrix is sub divided into six block matrix namely A, B, C, D, E, and F. The size of the each block matrix gets varied. Except the QCLDPC circulant shift procedure all the encoding procedures (ie Block size calculation) are same as in the LDPC structure. The obtained encoded message bits of QCLDPC is shown in Figure

$$H = \left[\begin{array}{ccc} \overset{\overset{A}{\uparrow}}{\boxed{\begin{matrix}1&1&0&0&1&1\\1&1&1&0&0&1\\0&0&0&1&1&0\\0&0&1&1&0&1\end{matrix}}} & \overset{\overset{B}{\uparrow}}{\boxed{\begin{matrix}0&1\\1&0\\0&1\\0&0\end{matrix}}} & \overset{\overset{C}{\uparrow}}{\boxed{\begin{matrix}0&0&0&1\\1&0&0&0\\1&1&0&1\\1&1&1&0\end{matrix}}} \\ \underset{\underset{D}{\downarrow}}{\boxed{\begin{matrix}1&0&1&0&0&0\\0&1&0&1&1&0\end{matrix}}} & \underset{\underset{E}{\downarrow}}{\boxed{\begin{matrix}1&0\\1&1\end{matrix}}} & \underset{\underset{F}{\downarrow}}{\boxed{\begin{matrix}0&1&1&1\\0&0&1&0\end{matrix}}} \end{array}\right]$$

Let the information data S_1= (0, 1, 1, 0, 1, 0)

The parity bit R_1& R_2 are calculated using the formula R_1=

$[FCAS_1^T] = [0\ 0\ ]$

$R_2= [C\ (AS_1^T) + BR_1^T] = [\ 0\ 0\ 1\ 0\ ]$

The encoded data = $[S_1\ R_1\ R_2]$

$$= [0, 1, 1, 0, 1, 0, 0, \underbrace{0, 0,}\ 0, 1, 0\ ]$$

S_1 R_1 R_2

2.4 5G NR QC-LDPC Characteristics:

As mentioned above, QC-LDPC codes play an important role in 5G communications and havebeen accepted as the channel coding scheme for the 5G eMBB data channel in 3GPP standard meeting. Figure 1 illustrates the general structure of the NR QC-LDPC base graph. The columns are dividedinto three parts: information columns, core parity columns, and extension parity columns. The rowsare partitioned into two parts: core check rows and extension check rows. As shown in the figure, thebase matrix is composed of five submatrices, namely, A, B, O, C, and I [22]. Submatrix A correspondsto systematic bits. In addition, B corresponds to the first set of parity bits and is a square matrix witha dual-diagonal structure: its first column is of weight 3, whereas the submatrix composed of othercolumns after the first column has an upper dual-diagonal structure. Submatrix O is an all-zero matrix.

For the efficient support of incremental redundancy hybrid automatic repeat request (IR-HARQ),a single parity-check (SPC) based extension is used to support lower rates, as shown in Figure 1.Submatrix C corresponds to SPC rows, and I is an identity matrix that corresponds to the second set ofparity bits, i.e., the SPC extension. The combination of A and B is referred to as the kernel, and theother parts (O, C, and I) are referred to as extensions. This code structure is similar to the Raptor-likeextension, as described in [26].The 3GPP agreed to consider two rate-compatible base graphs, denoted by BG1 and BG2, for thechannel coding. Base graphs BG1 and BG2 have similar structures. However, BG1 is targeted forlarger block lengths (500 ≤ K ≤ 8448) and higher rates (1/3 ≤ R ≤ 8/9), whereas BG2 is targeted forsmaller block lengths (40 ≤ K ≤ 2560) and lower rates (1/5 ≤ R ≤ 2/3). The actual base graph usage and the definition of the two matrices are detailed in the NR standard specification TS 38.212 [27].The base graph that supports K_{max} should support the following set of shift sizes Z, where $Z = a \times 2^j$ for $a \in \{2, 3, 5, 7, 9, 11, 13, 15\}$ and $0 \leq j \leq 7$.

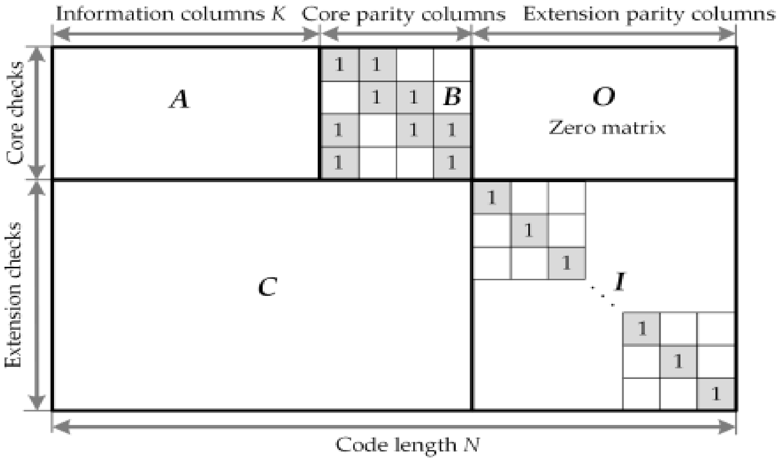
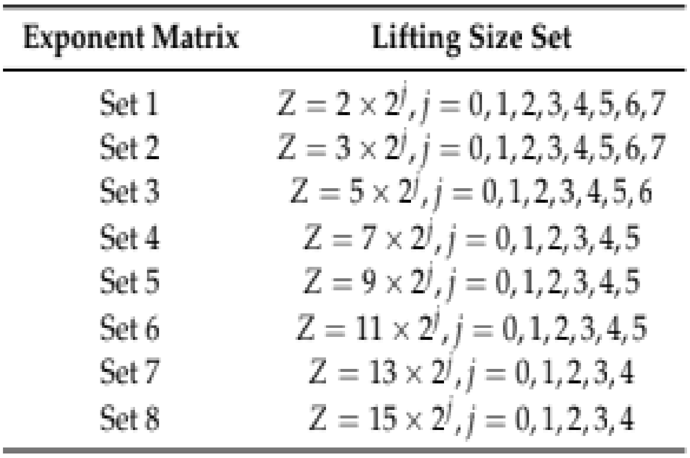


Figure 1. Sketch of base parity check structure for the 5G NR QC-LDPC codes.

For base graphs BG1 and BG2, the number of shift coefficient designs is 8. All lift sizes are divided into eight sets based on parameter a, where a is used for the definition of the lifting-size $a\times2^j$. The set of shift coefficients are listed in Table 1.

Table 1. Relationship between exponent matrices and sets of lifting size.



Exponent Matrix	Lifting Size Set
Set 1	$Z = 2 \times 2^j, j = 0,1,2,3,4,5,6,7$
Set 2	$Z = 3 \times 2^j, j = 0,1,2,3,4,5,6,7$
Set 3	$Z = 5 \times 2^j, j = 0,1,2,3,4,5,6$
Set 4	$Z = 7 \times 2^j, j = 0,1,2,3,4,5$
Set 5	$Z = 9 \times 2^j, j = 0,1,2,3,4,5$
Set 6	$Z = 11 \times 2^j, j = 0,1,2,3,4,5$
Set 7	$Z = 13 \times 2^j, j = 0,1,2,3,4$
Set 8	$Z = 15 \times 2^j, j = 0,1,2,3,4$

The shift value $P_{i,j}$ can be calculated using the function $P_{i,j} = f(V_{i,j}, Z)$, where $V_{i,j}$is the shift coefficient of the (i,j)-th element in the corresponding shift design. The function f is defined as

$$P_{i,j} = f(V_{i,j}, z) = \begin{cases} -1, & \text{if } V_{i,j} = -1, \\ mod(V_{i,j}, z), & else. \end{cases}$$

The following procedures are the steps of constructing the parity check matrix of the target (N, K)QC-LDPC code with a given information block size K and code rate R = K/N. For a base graph,

k_b denotes the number of information circulant columns; thus, if the lifting size is Z, K = Z × k_b nominally.

Step 1: Obtain the base graph BG1 or BG2 and determine the value of k_b for the given K and R.

– For BG1: $k_b = 22$.
– For BG2: $k_b = 10$ if $K > 640$; $k_b = 9$ if $560 < K \leq 640$; $k_b = 8$ if $192 < K \leq 560$;and $k_b = 6$ elsewhere.

Step 2: Determine Z by selecting the minimum Z value in Table 2, such that $k_b \times Z \geq K$.

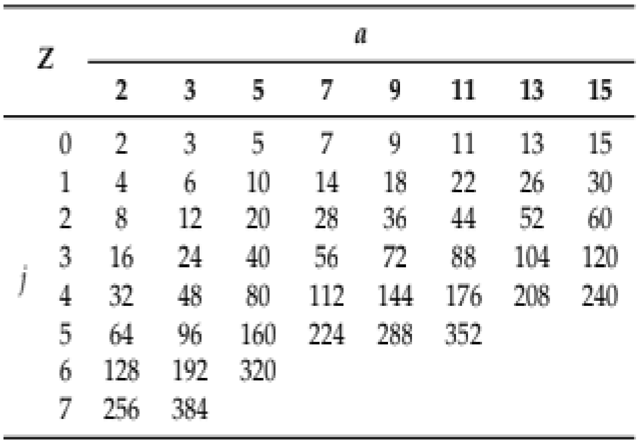
Step 3: After the lifting size Z is determined, the corresponding shift coefficient matrix is then selected from Table 1 {Set 1, Set 2,. . . , Set 8} according to set Z.

Step 4: Calculate the shifting coefficient value Pi,j by the modular Z operation, as discussed in Equation (4).

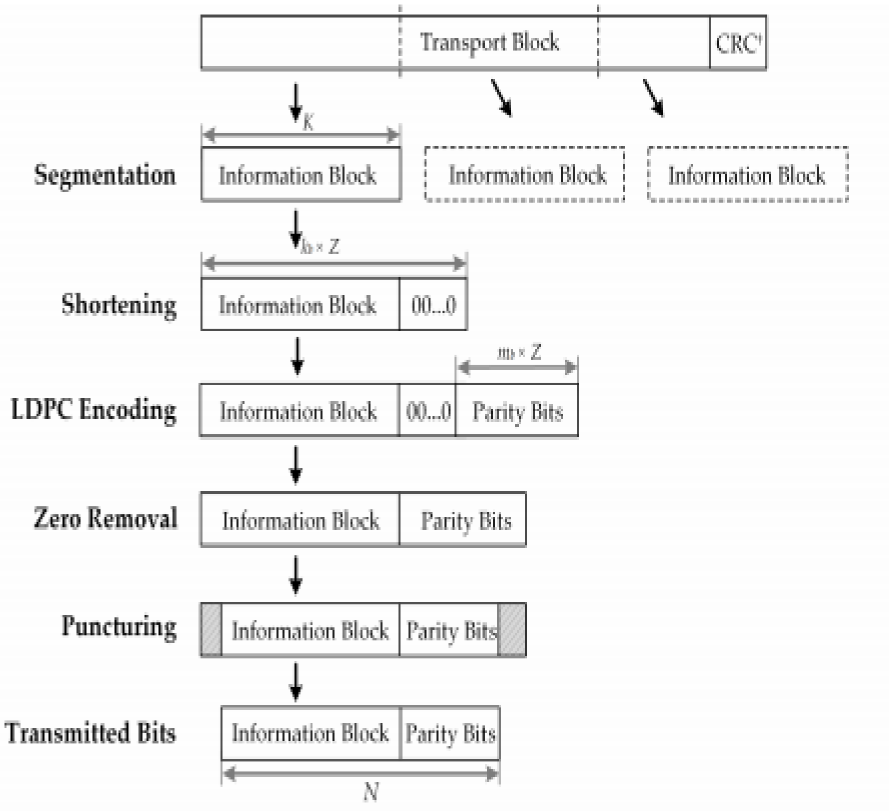
Step 5: Replace each entry in the final exponent matrix with the corresponding circulant permutation matrix or zero matrix of size Z × Z. The QC-LDPC code construction is completed and a parity check matrix H of size $m_bZ \times n_bZ$ is obtained. In 5G QC-LDPC codes, shortening and puncturing is carried out to obtain the desired information lengths and rate adaption.

Figure 2 presents an illustration of the encoding process of these codes

Table 2. Lifting size Z supported by standard 5G QC-LDPC codes.



Z		a							
		2	**3**	**5**	**7**	**9**	**11**	**13**	**15**
j	0	2	3	5	7	9	11	13	15
	1	4	6	10	14	18	22	26	30
	2	8	12	20	28	36	44	52	60
	3	16	24	40	56	72	88	104	120
	4	32	48	80	112	144	176	208	240
	5	64	96	160	224	288	352		
	6	128	192	320					
	7	256	384						



†CRC : Cyclic Redundancy Check

Figure 2. Shortening by zero padding and puncturing of standard 5G QC-LDPC codes.

Chapter 3

3.1 LDPC Encoding Algorithms:

Given a parity check matrix H, the objective of LDPC encoding is to solve parity equations:

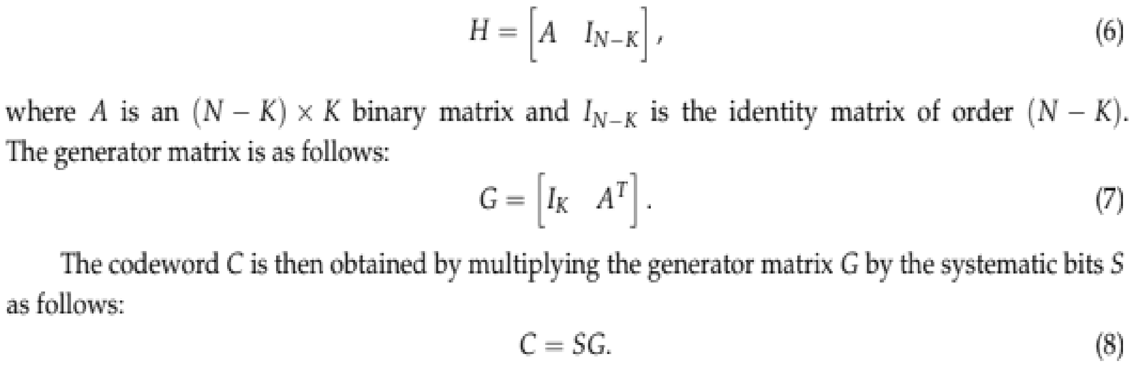
$$HC^T = 0^T$$

where C is the systematic codeword, which consists of the information bit vector S and parity code vector P.

This section presents a review on two generic encoding methodologies for the implementation of the LDPC encoder: the Gaussian elimination methodandtheRUmethod.

3.1.1 LDPC Encoding with Gaussian Elimination:

The Gaussian elimination is the most conventional method of encoding LDPC codes, which iscarried out by the multiplication of the generator matrix G, and contains a complexity quadratic in the block length [19]. The unknown generator matrix G can be derived from the parity check matrix H.A generator matrix for code with a parity check matrix H can be obtained by carrying out Gauss–Jordan elimination on H in the following form:



$$H = \begin{bmatrix} A & I_{N-K} \end{bmatrix}, \quad (6)$$

where A is an $(N-K) \times K$ binary matrix and I_{N-K} is the identity matrix of order $(N-K)$. The generator matrix is as follows:

$$G = \begin{bmatrix} I_K & A^T \end{bmatrix}. \quad (7)$$

The codeword C is then obtained by multiplying the generator matrix G by the systematic bits S as follows:

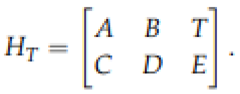
$$C = SG. \quad (8)$$

The sequential LDPC encoder based on the multiplication of the G matrix requires a ROM to store the generator matrix used to compute the codeword C. The main drawback of this approach is that, unlike parity check matrix H, the corresponding generator matrix G will most likely not be sparse. The complexity of this straightforward encoding algorithm is O(N2), where N is the number of bits in a codeword. Therefore, the implementation of the matrix multiplication at the encoder results in a very high complexity. For an arbitrary parity check matrix, the construction of

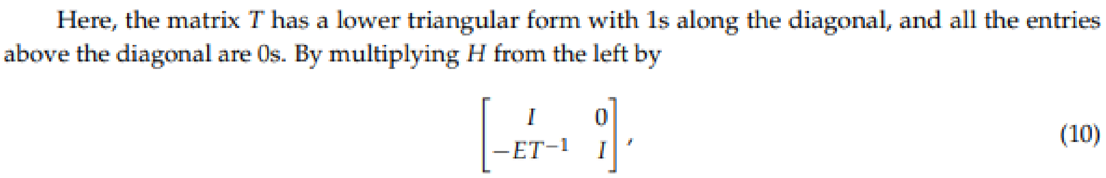
G should be avoided and encoding should be carried out using back substitutionwithH.

3.1.2 LDPC Encoding with the RU Method:

Instead of determining a generator matrix for H, an LDPC code can be directly encoded using the parity check matrix by transforming it into a lower triangular form and applying back substitution. The RU encoding method, which was proposed by Richardson and Urbanke, is a linear time encoding method for sparse parity check matrices. The underlying principle is transformation using only row and column permutations, to reformulate a parity check matrix H into a sparse matrix. Therefore, this approach can reduce the complexity more than the G matrix multiplication method. The RU algorithm consists of two steps: a pre-processing step and actual encoding step. First, in the pre-processing step, the parity check matrix H is converted into the approximate lower triangular (ALT) form, as shown in Figure 3. The parity check matrix H is given by the M × N matrix, where N is the block length of the code and M is the number of parity check equations. Given that the matrix transformation is realized solely by row and column permutations, the H matrix remains a sparse matrix:

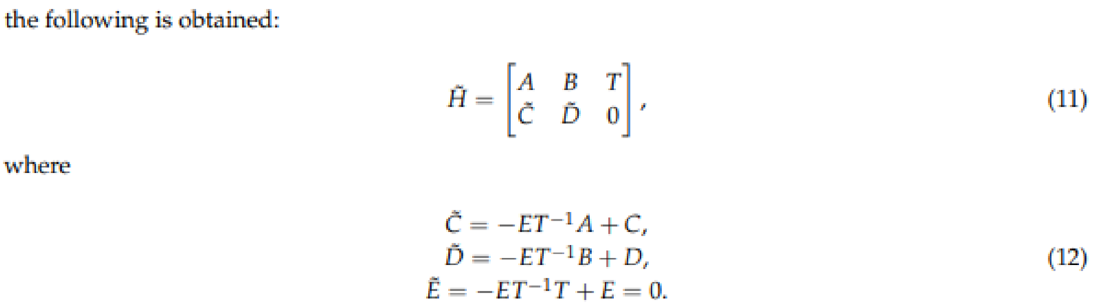


$$H_T = \begin{bmatrix} A & B & T \\ C & D & E \end{bmatrix}.$$



Here, the matrix T has a lower triangular form with 1s along the diagonal, and all the entries above the diagonal are 0s. By multiplying H from the left by

$$\begin{bmatrix} I & 0 \\ -ET^{-1} & I \end{bmatrix}, \qquad (10)$$



the following is obtained:

$$\tilde{H} = \begin{bmatrix} A & B & T \\ \tilde{C} & \tilde{D} & 0 \end{bmatrix}, \qquad (11)$$

where

$$\begin{aligned} \tilde{C} &= -ET^{-1}A + C, \\ \tilde{D} &= -ET^{-1}B + D, \\ \tilde{E} &= -ET^{-1}T + E = 0. \end{aligned} \qquad (12)$$

The actual encoding step is performed by matrix-multiplication, forward-substitution and vector addition operations. Let the codeword C = [s p_1 p_2] where s represents the information bits, p_1 denotes the first G parity check bits, and p_2 contains the remaining (M − G) parity check bits. The codeword C must satisfy the parity check equation $HC^T = 0^T$. The two equations are then expressed by:

$$As^T + Bp_1^T + Tp_2^T = 0^T,$$
$$\tilde{C}s^T + \tilde{D}p_1^T + 0p_2^T = 0^T.$$

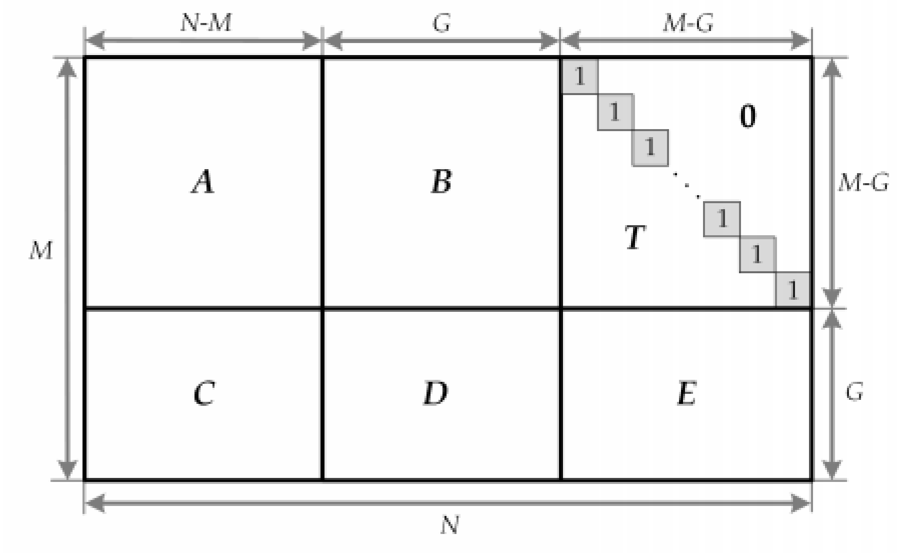


Figure 3. The parity check matrix H in approximate lower triangular form.

Using the RU method, the calculation of the parity bits in the first parity portion p_1 is only dependent on the information bits, given that E was cleared. Hence, it can be calculated independently of the parity bits in p_2. If D˜ is non singular, then p T 1 can be obtained fromEquation:

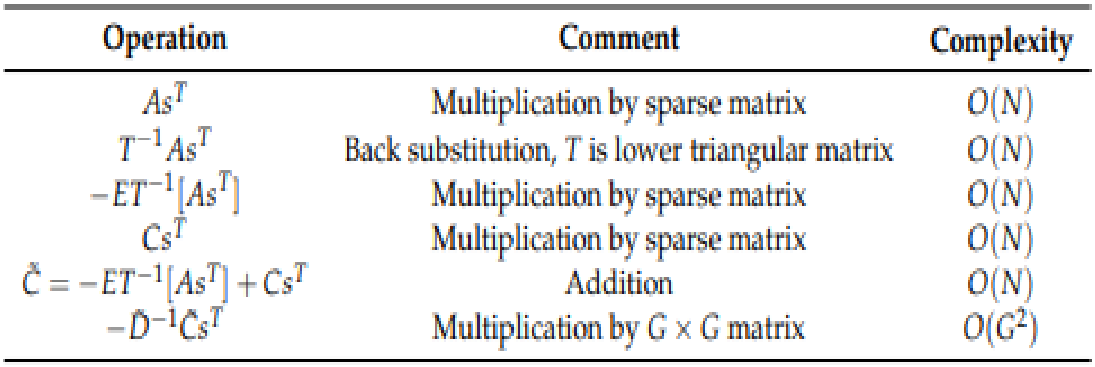
$$p_1^T = \tilde{D}^{-1}\tilde{C}s^T. \tag{14}$$

If $\tilde{D}$ is singular in GF(2), then it is necessary to further permute the columns of $\tilde{H}$ to eliminate this singularity. Once p_1 is known, p_2 can be determined using Equation (13):

$$p_2^T = -T^{-1}\left(As^T + Bp_1^T\right). \tag{15}$$

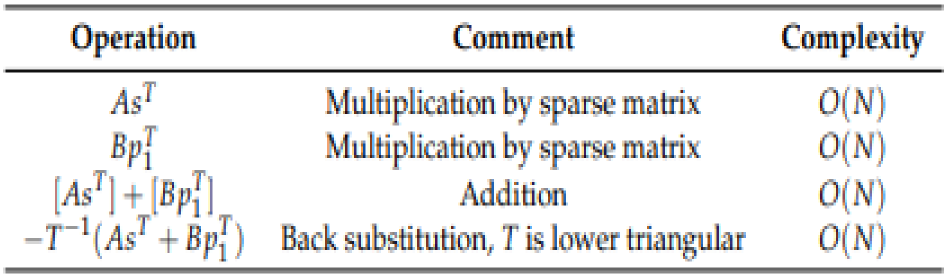
Given that T is the lower triangular form, p_2 can be found using back substitution. The complexity of this encoding procedure can be kept low since A, B and T are sparse. Tables 3 and 4 present the complexity of calculation of p_1^T and p_2^T, respectively. The complexity of the RU algorithm is given by $O(N + G^2)$, where N is the block length and G is the gap to linear encoding. The gap is actuallythenumber of rows of the parity check matrix that cannot be set into a triangular form using only row and column permutations. With a small gap G, the lower encoding complexity for the code is achieved.

Table 3. Complexity analysis of p_1^T calculation.



Operation	Comment	Complexity
As^T	Multiplication by sparse matrix	$O(N)$
$T^{-1}As^T$	Back substitution, T is lower triangular matrix	$O(N)$
$-ET^{-1}[As^T]$	Multiplication by sparse matrix	$O(N)$
Cs^T	Multiplication by sparse matrix	$O(N)$
$\tilde{C} = -ET^{-1}[As^T] + Cs^T$	Addition	$O(N)$
$-\tilde{D}^{-1}\tilde{C}s^T$	Multiplication by $G \times G$ matrix	$O(G^2)$

Table 4. Complexity analysis of p_2^T calculation.

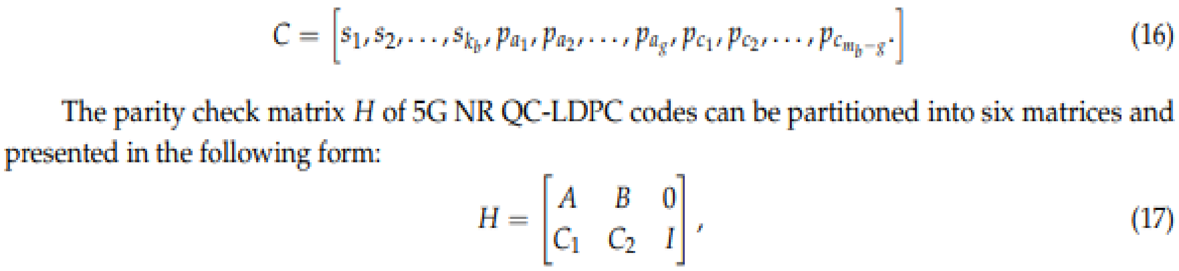


Operation	Comment	Complexity
As^T	Multiplication by sparse matrix	$O(N)$
Bp_1^T	Multiplication by sparse matrix	$O(N)$
$[As^T] + [Bp_1^T]$	Addition	$O(N)$
$-T^{-1}(As^T + Bp_1^T)$	Back substitution, T is lower triangular	$O(N)$

The disadvantage of encoding using the RU method is that there is no exact programmable step-by-step algorithm. The multiple matrix calculations in this algorithm significantly limit the development of a rapid flexible encoder. In addition, the RU method is subjected to a long critical path and odd constraints, which could render the LDPC encoder non-systematic.

3.2 Proposed QC-LDPC Encoding Algorithm:

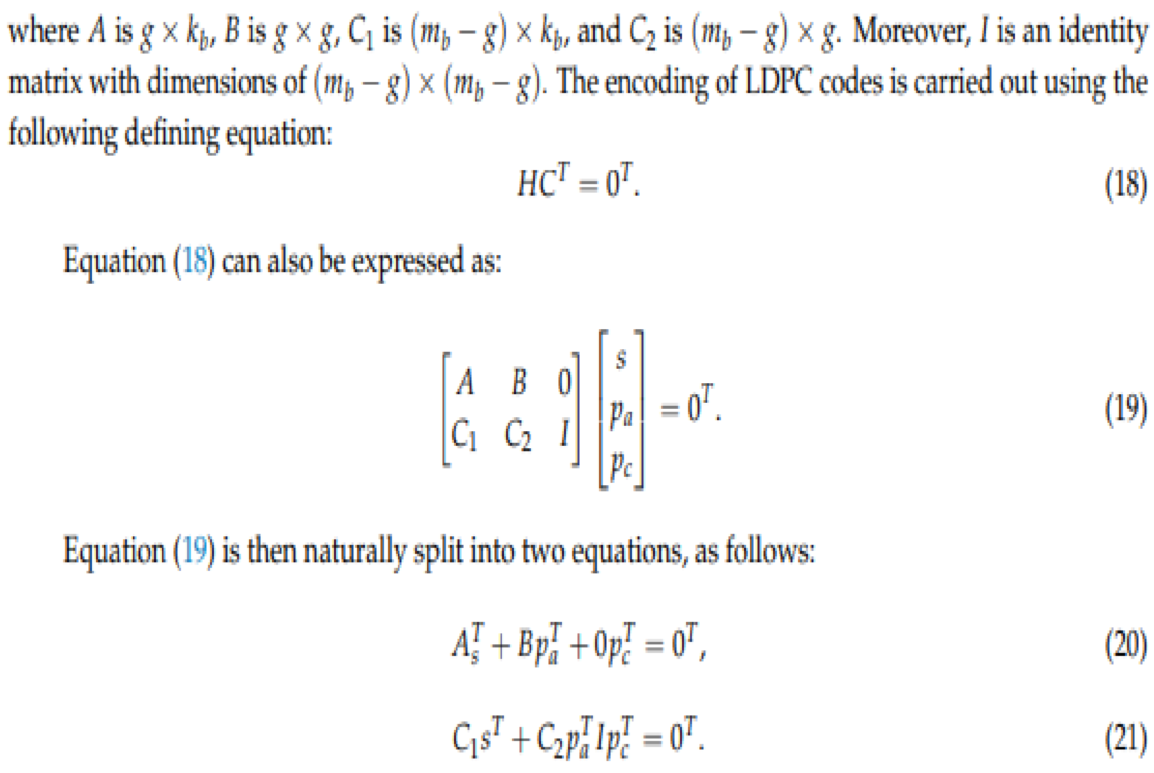
This section presents an efficient scheme developed in this study for the construction of efficient encoders for 5G NR QC-LDPC codes. The proposed encoding method is based on the special characteristics of 5G NR QC-LDPC codes, which are presented in Figure 1. The proposed architectures target low-complexity, while ensuring high-throughput. As reported in the literature review, base graphs BG1 and BG2 have similar structures. In this paper, we focus our description on BG1 with a size of $m_b \times n_b$ ($m_b = 46$, and $n_b = 68$), which is the main 5G NR high rate base graph. Let the codeword $C = [s\ p_a\ p_c]$, where s denotes the systematic portion, which is divided into 22 groups of Z bits, since the base graph BG1 has $k_b = n_b - m_b = 22$ information bit columns. Moreover, $s = [s_1, s_2, \ldots, s_{kb}]$, where each element of s is a vector of length Z. The information messages received by the encoder are stored in registers that are organized by kb blocks, denoted by s_i (i = 1, 2, . . . , k_b), which correspond to the systematic blocks, where each consists of Z bits. Given that the encoder was designed to read Z bits per clock cycle, it requires k_b cycles to store all the information blocks. Moreover, the parity sequence can be grouped into sets of Z bits. Suppose that the parity portion of each message p is split into two sub-components as follows: the first g = 4 parity bits pa = [pa_1 , pa_2 , . . . , pa_g], and the remaining (m_b − g) = 42 parity bits pc = [pc_1 , pc_2 , . . . , pc_{mb}−g] . More precisely, the encodedcodewordcanbeexpressedas:



$$C = \left[s_1, s_2, \ldots, s_{k_b}, p_{a_1}, p_{a_2}, \ldots, p_{a_g}, p_{c_1}, p_{c_2}, \ldots, p_{c_{m_b-g}}.\right] \quad (16)$$

The parity check matrix H of 5G NR QC-LDPC codes can be partitioned into six matrices and presented in the following form:

$$H = \begin{bmatrix} A & B & 0 \\ C_1 & C_2 & I \end{bmatrix}, \quad (17)$$



where A is $g \times k_b$, B is $g \times g$, C_1 is $(m_b - g) \times k_b$, and C_2 is $(m_b - g) \times g$. Moreover, I is an identity matrix with dimensions of $(m_b - g) \times (m_b - g)$. The encoding of LDPC codes is carried out using the following defining equation:

$$HC^T = 0^T. \tag{18}$$

Equation (18) can also be expressed as:

$$\begin{bmatrix} A & B & 0 \\ C_1 & C_2 & I \end{bmatrix} \begin{bmatrix} s \\ p_a \\ p_c \end{bmatrix} = 0^T. \tag{19}$$

Equation (19) is then naturally split into two equations, as follows:

$$A_s^T + Bp_a^T + 0p_c^T = 0^T, \tag{20}$$

$$C_1 s^T + C_2 p_a^T I p_c^T = 0^T. \tag{21}$$

The proposed algorithm is performed in two steps. In the initial step, the parity bits in the first portion pa are computed by solving Equation (20). The second step in the encoding process includes the computation of the pc parity portions using Equation (21). The first step in the encoder implementation is the determination of the p_a part. Initially, Equation (20) is re-written in block form as follows:

$$\begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,k_b} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,k_b} \\ a_{3,1} & a_{3,2} & \ddots & a_{3,k_b} \\ a_{4,1} & a_{4,2} & \cdots & a_{4,k_b} \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_{k_b} \end{bmatrix} + \begin{bmatrix} 1 & 0 & -1 & -1 \\ 0 & 0 & 0 & -1 \\ -1 & -1 & 0 & 0 \\ 1 & -1 & -1 & 0 \end{bmatrix} \begin{bmatrix} p_{a_1} \\ p_{a_2} \\ p_{a_3} \\ p_{a_4} \end{bmatrix} = 0. \tag{22}$$

This can then be expanded into the following set of equations:

$$\sum_{j=1}^{k_b} a_{1,j} s_j + p_{a_1}^{(1)} + p_{a_2} = 0, \tag{23}$$

$$\sum_{j=1}^{k_b} a_{2,j} s_j + p_{a_1} + p_{a_2} + p_{a_3} = 0, \tag{24}$$

$$\sum_{j=1}^{k_b} a_{3,j} s_j + p_{a_3} + p_{a_4} = 0, \tag{25}$$

$$\sum_{j=1}^{k_b} a_{4,j} s_j + p_{a_1}^{(1)} + p_{a_4} = 0, \tag{26}$$

where $p_{a_1}^{(\alpha)}$ denotes the α^{th} (right) cyclic shifted version of p_{a_1} for $0 \le \alpha \le Z$. By adding up all the above equations, the following is obtained:

$$p_{a_1} = \sum_{i=1}^{4} \sum_{j=1}^{k_b} a_{i,j} s_j. \tag{27}$$

It should be noted that a straightforward implementation of $a_{i,j}s_j$ can be done with the use of Z-bit cyclic shifters. Since $a_{i,j}s_j$ is a circular right shift of s_j with the shift coefficient defined by $a_{i,j}$, the hardware complexity is trivial. Based on the definition below,

$$\lambda_i = \sum_{j=0}^{k_b} a_{i,j} s_j \text{ for } i = 1,2,3,4, \tag{28}$$

the following can be obtained:

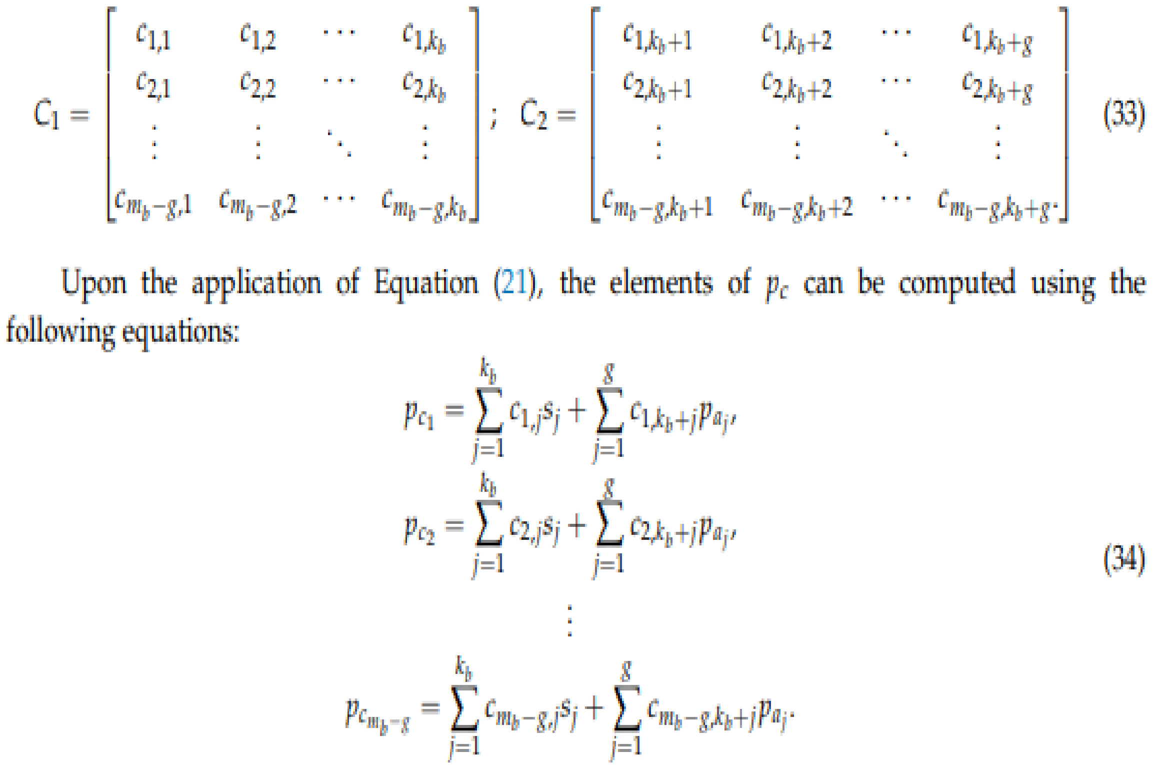
$$p_{a_1} = \sum_{i=1}^{4} \lambda_i, \tag{29}$$

$$p_{a_2} = \lambda_1 + p_{a_1}^{(1)}, \tag{30}$$

$$p_{a_3} = \lambda_3 + p_{a_4}, \tag{31}$$

$$p_{a_4} = \lambda_4 + p_{a_1}^{(1)}. \tag{32}$$

From Equation (28), each λ_i value is computed by accumulating all the $a_{i,j}s_j$ values. In Modulo 2, λ_i is obtained by carrying out XOR operations on all the elements of $a_{i,j}s_j$. The λ_i values can be estimated per clock cycle in g = 4 cycles. The first block of the parity bits pa1 is then calculated by accumulating all the λ_i values. The remaining parity bits pa_i can be obtained using a method that can be easily derived from Equations (30)–(32). This process can be done in two clock cycles since there is dependency between pa3 and pa4.All the parity bits pa in the first parity portion are stored in registers. In a second step, the pc portion can be easily determined based on Equation (21), where matrices C1 and C2 are givenby



$$C_1 = \begin{bmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,k_b} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,k_b} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m_b-g,1} & c_{m_b-g,2} & \cdots & c_{m_b-g,k_b} \end{bmatrix}; \; C_2 = \begin{bmatrix} c_{1,k_b+1} & c_{1,k_b+2} & \cdots & c_{1,k_b+g} \\ c_{2,k_b+1} & c_{2,k_b+2} & \cdots & c_{2,k_b+g} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m_b-g,k_b+1} & c_{m_b-g,k_b+2} & \cdots & c_{m_b-g,k_b+g}. \end{bmatrix} \tag{33}$$

Upon the application of Equation (21), the elements of p_c can be computed using the following equations:

$$\begin{aligned} p_{c_1} &= \sum_{j=1}^{k_b} c_{1,j}s_j + \sum_{j=1}^{g} c_{1,k_b+j}p_{a_j}, \\ p_{c_2} &= \sum_{j=1}^{k_b} c_{2,j}s_j + \sum_{j=1}^{g} c_{2,k_b+j}p_{a_j}, \\ &\vdots \\ p_{c_{m_b-g}} &= \sum_{j=1}^{k_b} c_{m_b-g,j}s_j + \sum_{j=1}^{g} c_{m_b-g,k_b+j}p_{a_j}. \end{aligned} \tag{34}$$

Similarly, $c_{i,j}s_j$ represents a circular shift of sj with the shift coefficient defined by $c_{i,j}$, and $c_i,k_{b+j}pa_j$ represents a circular shift of pa_j with the shift coefficient defined by c_i,k_{b+j}. As soon as $c_{i,j}s_j$ and $c_i,k_{b+j}pa_j$ have been obtained, they can be used to determine the value of the corresponding parity bits in the second parity portion pc. This step can be performed in a single clock cycle. Hence, all the pc parity bits can be acquired in (m_b − g) clock cycles. The encoded codeword is then a combination of the original message s and the two calculated parity portions pa and pc.

Chapter 4

Proposed QC-LDPC Encoder Architecture:

Figure below details the overall block diagram for the proposed low complexity 5G NR QC-LDPC code encoder. The hardware architectures were designed to conduct the encoding process through steps defined in Equations (29)–(32) and (34). In the first step, the computation of the parity bits in the first portion pa is carried out. From Equation (28), each λ_i value is computed by accumulating all the cyclic shift results of s_j . Since the information message s consists of k_b blocks of Z bits, a total of k_b = 22 barrel shifters of size Z, which are denoted by CS_j , j = 1, 2, ..., k_b , are required for the circular shift process. The vector addition of all the λ_i components is then carried out by the XOR trees. Each intermediate λ_i value corresponding to Equation (28) can be estimated per clock cycle and stored in the λ_memory to be used later. Thus, the value of pa1 can be obtained in g = 4 clock cycles when all λi values are obtained and stored in memory. The remaining parity bits of pa can be obtained in 2 clock cycles with the use of XOR gates. The objective of the second step is the calculation of the parity bits in the second portion pc. According to (34), the parity blocks pc_i can be achieved by the vector addition of $c_{i,j}s_j$ and $c_i{,}k_{b+j}pa_j$. The value of $c_{i,j}s_j$ is also computed by accumulating all the cyclic shift results of s_j . In this step, the overall hardware complexity can be further decreased by exploiting the sharing technique. More specifically, the barrel shifters and XOR trees are reused for the computation of pc in this step. Control signals are generated by the controller block. The value of c_i,$k_{b+j}pa_j$ is estimated by accumulating all the cyclic shift results of pa_j . The required number of Z–bit barrel shifters is g = 4. The main blocks of the proposed architecture can be described as follows.

(1) Input/ Output Buffer: the input buffer, which is implemented as a number of serial input parallel output shift registers, is exploited to store the input systematic bits s_i received by the encoder. The output buffer is used to store the encoded codeword.

(2) Memory Blocks: two memory blocks are utilized, namely, one for the submatrix permutation values, and the others for the accumulated values λ that correspond to matrix A. In Figure 4, the AROM, C1ROM,and C2ROM correspond to the ROMs that store the coefficients of matrix A, matrix C1, and matrix C2, respectively. Under the assumption that $q^{˜} = d\log_2 Z_e$ bits represent the required word length to store the permutation information for each sub matrix: q_{gk} ˜ b , $q^{˜}(m_b - g)k_b$, and $q^{˜}(m_b - g)g$ bits are required to store matrix A, C1, and C2, respectively. A significant portion of the hardware complexity of the LDPC encoder consists of the memory required to store the parity check matrix. Unlike the

RU method, the proposed algorithm does not require the inverse of the component matrix, which reflects its primary advantage over the RU method. Compared with the Gaussian method, the proposed architecture does not require for block-memories to store the generator matrix G, which further decreases the number of required components. The λ_memory is implemented as a dual port random access memory (RAM) for storing λi messages (i = 1, 2, . . . , g). Each memory word λ_i consists of Z bits, corresponding to one accumulated message of matrix A. Moreover, a total of (g × Z) bits of λ_memory are required for the proposed encoder.

(3) Barrel Shifters: barrel shifters are used to implement the cyclic shift permutations, according to the shift values provided by the cyclic shifter controllers. It should be noted that the number of cyclic shifters is equal to the number of message blocks, and the size of the barrel shifters is equal to sub matrix size Z.

(4) XOR Trees: In Modulo 2, the addition implementation is obtained by carrying out an XOR operation on all the elements.

(5) Controller: this block generates control signals, such as data_sel to indicate the step being processed; and mem_en, to enable write access to the λ_memory.

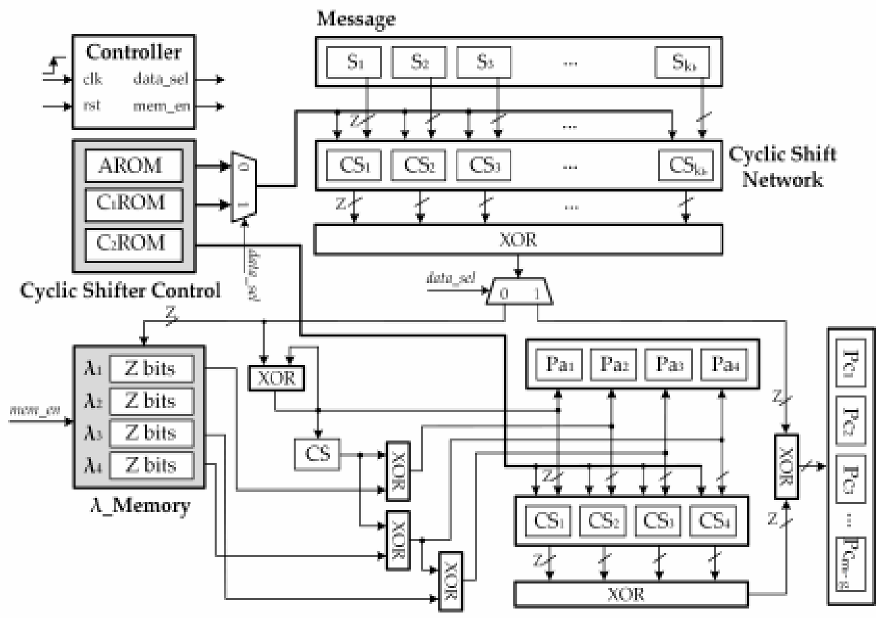


Figure 4. Low-complexity high-throughput encoder architecture for 5G NR QC-LDPC code.

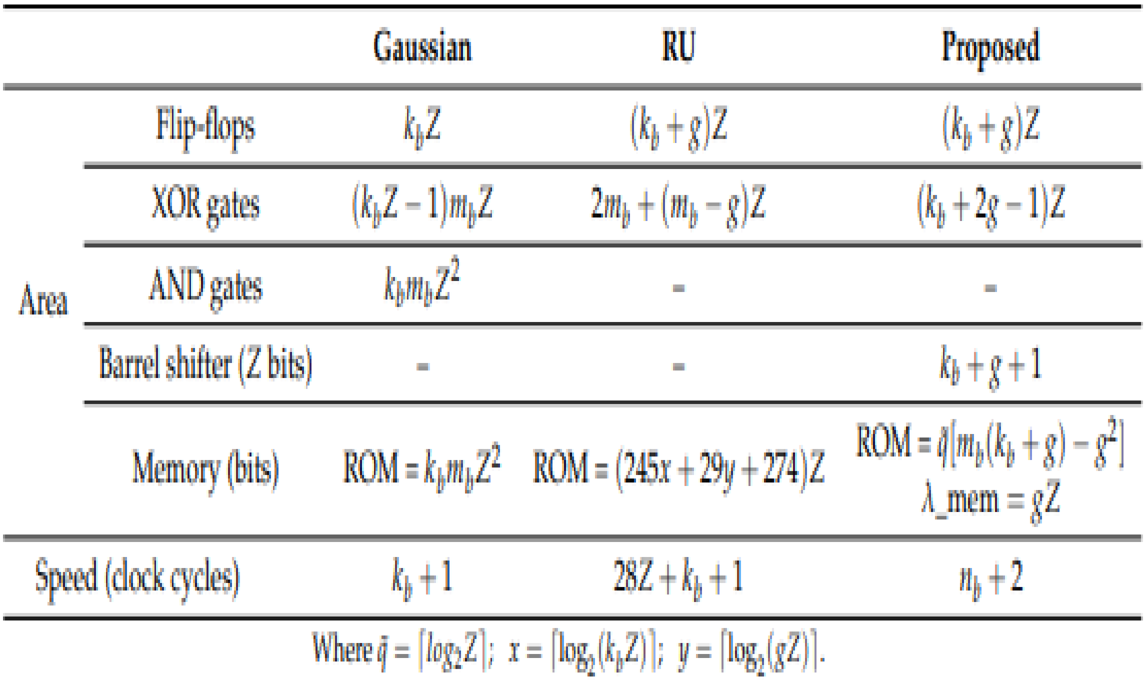
Chapter 5

Performance Analysis and Comparison:

This section reports the implementation results of the proposed LDPC encoder architecture as well as a detailed comparison between the proposed method and other encoder implementations in terms of area and speed for 5G NR standard. First, the design characteristics of different LDPC code encoders are discussed. Thereafter, an analysis of the proposed LDPC encoder, with respect to its implementation on ASIC, is presented. Table 5 presents a comparison between the area and speed of the proposed encoding method and those of other state-of-the-art approaches. As shown in Table 5, the matrix size was utilized to determine the

ROM storage, and the Hamming weights of the matrices were used in computing the gate count. Since all the systematic bits and parity check bits in the first parity portion are stored in registers, the number of flip-flops required was estimated by the bit sizes of K and pa. In Table 5, the time interval between input frames was exploited in order to compare the processing speed of different encoding methods. The time between two consecutive input frames is based on the total number of clock cycles between the arrivals of the first Z bits of a frame up to the cycle wherein the encoder is ready to receive another frame. To make it clear, a target LDPC code with a base graph BG1 and submatrix size Z = 16 was considered in Table 6. As can be observed from Table 6, the proposed encoder gains a significant reduction in the storage overhead. In the Gaussian elimination method, the entire generator matrix G is stored in the memory. In the RU method, the location of the edges (ones) of each row is stored, with an extra bit indicating the end of a row. By only storing the values of shift coefficients for each submatrix, the proposed method dramatically reduces the ROM size by 98.2% and 88.9% when compared with the G matrix method and RU method, respectively. Moreover, the proposed encoder reduces the number of XOR logic gate counts by 1.65 times compared with the RU method. This leads to a significant reduction in the hardware complexity for the proposed encoder as these components are the main contributors of logic resources in the encoder architecture. Hence, the proposed encoding structure shows a significant advantage over other LDPC encoding methods with respect to hardware complexity. As can be seen from Table 6, the Gaussian elimination approach requires only 23 clock cycles to generate the encoded codeword for a given LDPC code. However, this method suffers from a significant storage overhead which makes it less of an idea for implementation. From the analysis of the RU design, it was found to require 471 clock cycles per codeword. This is significantly higher than that of the proposed encoder, which only requires 70 clock cycles. From Table 6, it can be observedthat the number of clock cycles required per codeword for the encoding of the proposed encoder design decreased to 14.8% of that of RUmethod.

Table 5. Comparison between Gaussian method, RU method, and proposed method.



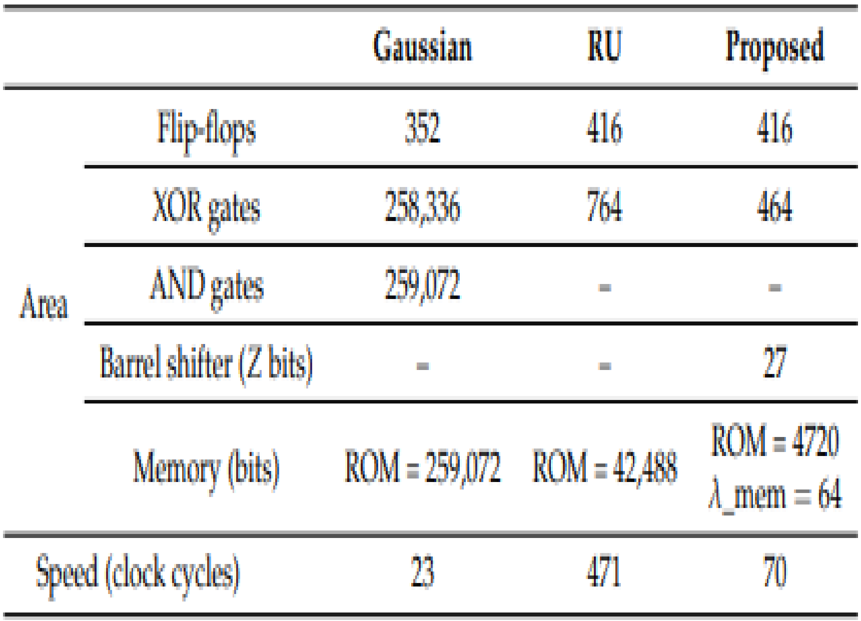
		Gaussian	RU	Proposed
Area	Flip-flops	$k_b Z$	$(k_b + g)Z$	$(k_b + g)Z$
	XOR gates	$(k_b Z - 1)m_b Z$	$2m_b + (m_b - g)Z$	$(k_b + 2g - 1)Z$
	AND gates	$k_b m_b Z^2$	-	-
	Barrel shifter (Z bits)	-	-	$k_b + g + 1$

	Memory (bits)	ROM = $k_b m_b Z^2$	ROM = $(245x + 29y + 274)Z$	ROM = $\bar{q}[m_b(k_b + g) - g^2]$
				λ_mem = gZ

Speed (clock cycles)		$k_b + 1$	$28Z + k_b + 1$	$n_b + 2$

Where $\bar{q} = \lceil log_2 Z \rceil$; $x = \lceil \log_2(k_b Z) \rceil$; $y = \lceil \log_2(gZ) \rceil$.

Table 6. Comparison between Gaussian method, RU method, and proposed method for submatrix size $Z = 16$.



		Gaussian	RU	Proposed
Area	Flip-flops	352	416	416
	XOR gates	258,336	764	464
	AND gates	259,072	-	-
	Barrel shifter (Z bits)	-	-	27

	Memory (bits)	ROM = 259,072	ROM = 42,488	ROM = 4720
				λ_mem = 64

Speed (clock cycles)		23	471	70

Chapter 6

Conclusion:

In this proposed work, a novel low-complexity high-throughput encoder approach for the 5G NR standard is proposed. Based on the proposed encoding algorithm.The derived architecture exhibits a significantly lower hardware complexity, as it decreases the memory and logic component requirements. The proposed design demonstrates a superior performance to the alternative methods. Moreover, the synthesis results revealed that the proposed design is appropriate for the high throughput 5G standard.

References:

I. Gallager, R.G. Low-Density Parity-Check Codes; MIT Press: Cambridge, MA, USA, 1963.

II. Richardson, T.; Kudekar, S. Design of Low-Density Parity Check Codes for 5G NewRadio.

III. Tang, H.; Xu, J.; Kou, Y.; Lin, S.; Abdel-Ghaffar, K. On Algebraic Construction of GallagerandCirculantLow-DensityParity-CheckCodes.

IV. Huang, Q.; Tang, L.; He, S.; Xiong, Z.; Wang, Z. Low-Complexity Encoding of Quasi-Cyclic Codes Based on Galois Fourier Transform.

V. Jung, Y.; Chung, C.; Jung, Y.; Kim, J. 7.7 Gbps Encoder Design for IEEE 802.11ac QC-LDPC Codes.

VI. Li, H.; Bai, B.; Mu, X.; Zhang, J.; Xu, H. Algebra-Assisted Construction of Quasi- QC- LDPC Codes for 5G New Radio.

VII. PAPR REDUCTION USING QUASI CYCLIC LDPC CODES from https://shodhganga.inflibnet.ac.in/.

VIII. Idea is taken from Efficient QC-LDPC Encoder for 5G New Radio
Tram Thi Bao Nguyen and Tuy Nguyen Tan and Hanho Lee.

IX. Construction of Quasi-Cyclic LDPC Codes Based on Fundamental Theorem of Arithmetic Hai Zhu, Liqun Pu, Hengzhou Xu , and Bo Zhang

X. Chen, T.; Vakilinia, K.; Divsalar, D.; Wesel, R.D. Protograph Based Raptor-like LDPC Codes.