Efficient Method for Frozen Bits Encoding and Decoding of Polar Code

Yan-Haw Chen¹, Min-Yen Tsai², J.J. Wang³, Zhe-Wei Kang³ and Jian-Ting Pan³

¹Dept. of Computer Science and Information Engineering, I-Shou University
²Dept. of Computer Science and Information Engineering, I-Shou University
³Dept. of Computer Science and Information Engineering, I-Shou University

The corresponding author’s e-mail: wsx102218@gmail.com

Abstract. An efficient encoding and decoding of the Polar code utilizing algebraic method to find error patterns is presented in this study. The key idea behind the proposed decoding method is theoretically based on the existence of a one-to-one mapping between the frozen bits and correctable error patterns. This method would help reduce the decoding time for finding error patterns when decoding the ($N=8$, $K=4$) and ($N=16$, $K=8$) Polar Code. Furthermore, it would reduce the encoding by 91% with lookup table for $G_3$ and $G_4$ matrix and the encoding method can be used to the decoding of Polar Code with algebraic method. Ultimately, the proposed decoding algorithm for Polar codes can be made regular, simple, and suitable for software implementations.

1. Introduction
The encoding of Polar Code is a linear block coding method recently confirmed by Arikan [1], which can not only correct the error but approach the Shannon-Hartley theorem. This caused the researchers' interest, the decoding of this code architecture is not complicated, the reliability of the data in the performance of error correction is excellent. Due to the complexity of encoding and decoding, it grows logarithmically with the increase of code length. If using Successive Cancellation (SC) algorithm [2-4], the decoding complexity increases little. However, the SC decoding algorithm used in the long code decoding process, its error correction performance is worse than Turbo code and LDPC code. Recent research literature based on the lack of SC decoding algorithm for method improving [5-9]. For example, performance of algorithms such as List SC (SCL), Stack SC (SSC), Cyclic Redundancy Checker-assisted SCL (CRC-SCL). These methods are also implemented as hardware circuits [10], and these decoding technical documents only focus on the soft decision mode to discuss the error correction performance. Since the standard form of Polar Code is Non-systematic code [1], and in [11] it is mentioned that systematic code has a lower bit error rate (BER) than non-systematic codes. Therefore, the coding method in this paper, use the method of look-up table and algebra scheme to correct errors. The error correction algorithm of Polar Code will use the characteristics presented by the frozen bit. Since the frozen bit must be zero after decoding without error, a simple error correction rule can be established by using this feature and solving the data errors that may occur in real communications.

2. Experiment Method
There will be divided into three parts to illustrate: generation matrix, encoding, decoding and error correction.

2.1. Generation Matrix
The definition of a generation matrix is as follows:

\[ G = \begin{bmatrix}
1 & 1 \\
0 & 1
\end{bmatrix} \]  
(1)

The definition of expansion matrix of generation matrix is as follows, using Kronecker product to produce matrix of encoding and decoding:

\[ G \otimes k = \begin{bmatrix}
1 \times G^{\otimes k-1} & 1 \times G^{\otimes k-1} \\
0 \times G^{\otimes k-1} & 1 \times G^{\otimes k-1}
\end{bmatrix}, k \geq 2, k \in \mathbb{Z}^+ \]  
(2)

In this paper, it will use \( G^{\otimes 3} \) and \( G^{\otimes 4} \):

\[ G^{\otimes 3} = \begin{bmatrix}
1 \times G^{\otimes 2} & 1 \times G^{\otimes 2} \\
0 \times G^{\otimes 2} & 1 \times G^{\otimes 2}
\end{bmatrix} \]  
(3)

\[ G^{\otimes 4} = \begin{bmatrix}
1 \times G^{\otimes 3} & 1 \times G^{\otimes 3} \\
0 \times G^{\otimes 3} & 1 \times G^{\otimes 3}
\end{bmatrix} \]  
(4)

Since the generation matrix of the construction of Polar Code is related to the generation of Reed-Muller (RM) code, the generator matrix has the characteristics of the upper triangular, which also means that the encoding matrix is inverse matrix.

2.2. Encoding
The vector of 8 bits original data is as follows:

\[ D = (d_0, d_1, d_2, d_3, d_4, d_5, d_6, d_7) \]  
(5)

The vector of 8 bits frozen bits and its elements must be zero is as follows:

\[ F = (f_0, f_1, f_2, f_3, f_4, f_5, f_6, f_7) = (0, 0, 0, 0, 0, 0, 0, 0) \]  
(6)

The vector of 16 bits data with \( D \) and \( F \) is as follows:

\[ M = (m_0, m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8, m_9, m_{10}, m_{11}, m_{12}, m_{13}, m_{14}, m_{15}) \]  
(7)

2.2.1. Original Method. The original encoding and decoding method is to use matrix operations to get the result. The operation is as follows:

\[ M' = G^{\otimes 4} M^T \]  
(8)

This will take 256 times AND operations and 240 times XOR operations. The lower triangular part of \( G^{\otimes 4} \) does not need to be calculated, so only 136 AND operations and 120 XOR operations are needed actually.

2.2.2. Look-up Table Method. Before discuss about the look-up table method of encoding and decoding of Polar Code, it need to establish the table \( T \) by the matrix multiplication of \( G^{\otimes 3} \) and \( X_j = (x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7) \). \( X_j \) is a binary combination of 0 to 255. \( X' \) is the result of \( G^{\otimes 3} X \).

\[ X' = G^{\otimes 3} X \]  
(9)

Splitting \( M \) into two parts \( M_0, M_1 \):

\[ M_0 = (m_0, m_1, m_2, m_3, m_4, m_5, m_6, m_7) \]  
(10)
and forming two new $8 \times 8$ matrix $P_0, P_1$ taken from the parts of $G^\otimes 4$, in fact, it is the result of Kronecker product of $1 \otimes G^\otimes 3 (G^\otimes 3 \text{Matrix})$ and $0 \otimes G^\otimes 3 (\text{Zero Matrix})$: \[
\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix},
\]
\[
P_1 = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]
\[ (11) \]

The original the encoding and the decoding formula of Polar code are as follows: \[
M' = (m'_{0}, m'_{1}, m'_{2}, m'_{3}, m'_{4}, m'_{5}, m'_{6}, m'_{7}, m'_{8}, m'_{9}, m'_{10}, m'_{11}, m'_{12}, m'_{13}, m'_{14}, m'_{15})^T
\]
\[ (12) \]

The above $M'$ matrix can be split into four sub-matrixes as follows:

\[
\begin{align*}
M'_A &= P_0 M_0^T \\
M'_B &= P_0 M_1^T \\
M'_C &= P_1 M_0^T \\
M'_D &= P_1 M_1^T
\end{align*}
\]
\[ (13) \]

Each row of $P_{od}, P_{id}$ can be presented array $P_{od}[][], P_{id}[][]$ with $(255, 85, 51, 17, 15, 5, 3, 1)$ and $(0, 0, 0, 0, 0, 0, 0, 0)$, respectively.

$M_0$, $M_1$ is presented value $m_{0-7}, m_{8-15}$ in decimal. And then using the table and look-up table mentioned earlier. The result of $m_{0-7}$ must be zero, and the result of $m_{8-15}$ is same as $m_{8-15}$. So neither of these two parts need to be calculated:

\[
\begin{align*}
m_{A0} &= T[P_{od}[0] & M_0] \\
m_{A1} &= T[P_{od}[1] & M_0] \\
m_{A2} &= T[P_{od}[2] & M_0] \\
m_{A3} &= T[P_{od}[3] & M_0] \\
m_{A4} &= T[P_{od}[4] & M_0] \\
m_{A5} &= T[P_{od}[5] & M_0] \\
m_{A6} &= T[P_{od}[6] & M_0] \\
m_{A7} &= T[P_{od}[7] & M_0]
\end{align*}
\]
\[ (14) \]

The subsequent operation is completed as follows:

\[
\begin{align*}
m'_0 &= m_{A0} \oplus m_{B0} \\
m'_1 &= m_{A1} \oplus m_{B1} \\
m'_2 &= m_{A2} \oplus m_{B2} \\
m'_3 &= m_{A3} \oplus m_{B3} \\
m'_4 &= m_{A4} \oplus m_{B4} \\
m'_5 &= m_{A5} \oplus m_{B5} \\
m'_6 &= m_{A6} \oplus m_{B6} \\
m'_7 &= m_{A7} \oplus m_{B7}
\end{align*}
\]
\[ (15) \]

The look-up table method will take 16 times AND operations and 8 times XOR operations. In addition to 16 times the look-up table.

2.3. Decoding and Error Correction

Decoding and encoding use the same matrix for multiplication, so the same can be through the look-up table solution. Similarly, each bit must also be calculated as a decimal number in order to look up the table. Assumed that the encoded $M'$ codeword is transmitted with a single bit error or not with error, and then the decoded codeword will be presented by $M'_e = M' + e_i$, $i$ represents the location into which a single-bit error is injected.

\[
\begin{align*}
M'_e &= (m'_0, m'_1, m'_2, m'_3, m'_4, m'_5, m'_6, m'_7) \\
M'_e &= (m'_8, m'_9, m'_{10}, m'_{11}, m'_{12}, m'_{13}, m'_{14}, m'_{15})
\end{align*}
\]
\[ (16) \]
Through program verification of all codewords generated when a single bit error is injected, one-to-one property of frozen bit and single bit error can be found. Before correcting error, it need to decode $M_e$.

$$M_{eA} = P_0 M_0^T = (m_{e0}, m_{e1}, m_{e2}, m_{e3}, m_{e4}, m_{e5}, m_{e6}, m_{e7})^T$$

$$M_{eB} = P_0 M_1^T = (m_{e8}, m_{e9}, m_{e10}, m_{e11}, m_{e12}, m_{e13}, m_{e14}, m_{e15})^T$$

(17)

Where $P_0$ is same as $1 \ast G^{\otimes 3}$ ($G^{\otimes 3}$ Matrix).

Next, $M'_{eA}$ is obtained the same procedure, the table and rules of a single bit error correction is as follows:

1. If $m_{e0} = 1$ Then codeword has a single bit error and it can find the error location through the following Table 1.
   
   Else If All frozen bits $= 0$ Then codeword has no error.
   
   Else $m_{e0} = 0$, but other frozen bits $\neq 0$ Then codeword has least two bits error.

2. Table 1. Frozen bit for a single bit error table. Through a specific frozen bit position, this table can tell which bit was wrong in encoding.

**Table 1. Frozen bit for a single bit error table.**

| Frozen location | $m_{e0}$ | $m_{e1}$ | $m_{e2}$ | $m_{e3}$ | $m_{e4}$ | $m_{e5}$ | $m_{e6}$ | $m_{e7}$ |
|-----------------|---------|---------|---------|---------|---------|---------|---------|---------|
| Error location  | $m_0$   | 1       | 1       | 1       | 1       | 1       | 1       | 1       |
|                 | $m_1$   | 1       | 0       | 1       | 1       | 1       | 1       | 0       |
|                 | $m_2$   | 1       | 1       | 0       | 1       | 1       | 1       | 0       |
|                 | $m_3$   | 1       | 0       | 0       | 1       | 1       | 1       | 0       |
|                 | $m_4$   | 1       | 1       | 1       | 0       | 1       | 1       | 0       |
|                 | $m_5$   | 1       | 0       | 1       | 0       | 1       | 1       | 0       |
|                 | $m_6$   | 1       | 1       | 0       | 0       | 1       | 1       | 0       |
|                 | $m_7$   | 1       | 0       | 0       | 0       | 1       | 1       | 0       |

However, it is not worthwhile to handle only one bit error. Therefore, this study aims at the adjustment of data bit and frozen bit, and proposes a modified look-up table method.

**2.4. Modified Look-up Table Method**

Modifying data vector $D$ ,frozen bit vector $F$ and 16 bits data vector $M$ :

$$D = (d_0, d_1, d_2, d_3, d_4, d_5)$$

$$F = (f_0, f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9) = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$$

(18)

$$M = (f_0, f_1, f_2, d_0, f_3, f_4, d_3, f_5, f_6, f_7, f_8, d_4, f_9, d_5, d_6, d_7)$$

$$= (m_0, m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8, m_9, m_{10}, m_{11}, m_{12}, m_{13}, m_{14}, m_{15})$$

The frozen bit of 2 bits error frozen bit is calculated as an integer number $F_z$, it will be used as an index to establish the table $T_f$, and correspond the error correction value.

$$T_f[F_z] = \text{error correction value}$$

(19)

When the $F_z$ value is 1, 2, 3, 9, 18, 27, 65, 73, 97, 130, 146, 195, 219, there are a problem of the same frozen bits at different 2-bit error position. Therefore, if the value of $F_z$ has the value mentioned above, it will need to check 2 times.

**3. Result and Conclusion**

3.1. Original method v.s. Look-up table
Table 2. Origin method v.s. Look-up table.

| Operation | Origin method | Look-up table |
|-----------|---------------|---------------|
| AND       | 272 times     | 53 times      |
| XOR       | 240 times     | 16 times      |
| OR        | -             | 32 times      |
| Shift     | -             | 37 times      |
| Look-up   | -             | 32 times      |

The results from the previous mathematical operation are shown in Table 2. The look-up table method saves a lot of AND and XOR operations, which is still cost-effective to the original matrix operation coding, although it needs to be built before use. This will be conducive to the current Internet of Things (IoT) trends and needs.

3.2. Error Correction and Detection

It is possible to correct a 1-bit error and detect at least two bit errors by frozen bit. About error correction rules can refer to 2.3. Section. If the decoding needs to correct two bits error, it will to decrease k/n rate.

References

[1] E. Arıkan, “Channel combining and splitting for cutoff rate improvement,” *IEEE Trans. Inform. Theory*, vol. IT-52, pp. 628–639, Feb. 2006.
[2] E. Arıkan, “Channel polarization: A method for constructing capacity achieving codes for symmetric binary-input memoryless channels,” *IEEE Trans. Inform. Theory*, vol. 55, pp. 3051–3073, 2009.
[3] E. Arıkan, “Source polarization,” in Proc. IEEE Inform. Theory, Austin, TX, USA, June 2010.
[4] E. Arıkan and E. Telatar, “On the rate of channel polarization,” in Proc. IEEE Symp. Inform. Theory, Seoul, South Korea, pp. 1493–1495, 2009.
[5] N. Hussami, S. B. Korada, and R. Urbanke, “Performance of polar codes for channel and source coding,” in Proc. IEEE Int. Symp. Inform. Theory, pp. 1488–1492, June 2009.
[6] S. B. Korada, “Polar codes for channel and source coding,” Ph.D. thesis, EPFL, Lausanne, 2009.
[7] E. Abbe and E. Telatar, “MAC polar codes and matroids,” in Proc. Inform. Theory and App. Workshop, pp. 1-8, Jan. 2010.
[8] E. Hof, I. Sason, and S. Shamai, “Polar coding for reliable communications over parallel channels,” in Proc. IEEE Inform. Theory Workshop, Aug. 2010.
[9] H. Mahdavifar and A. Vardy, “Achieving the secrecy capacity of wiretap channels using polar codes,” *IEEE Trans. Inform. Theory*, vol. 57, no. 10, pp.6428-6443, October 2011..
[10] C. Leroux, I. Tal, A. Vardy, and W. J. Gross, “Hardware architectures for successive cancellation decoding of polar codes,” ICASSP 2011, May 2011.
[11] Erdal Arıkan, “Systematic Polar Coding,” *IEEE Commun. Lett.*, vol. 15, no. 8, pp.860-862, August 2011.