Modified Majority Logic Decoding of Reed-Muller Codes for Channel Coding

Ting-Ya Yang, Houshou Chen*, and Chen-Han Chuang

Dept. of Electrical Engineering and Graduate Institute of Communication Engineering
National Chung Hsing University, 250, Kuo Kuang Rd., Taichung 402, Taiwan

*Corresponding Author: houshou@dragon.nchu.edu.tw

This work was supported by Grant MOST-107-2221-E-005-015-MY2

Abstract

This paper proposes the modified majority-logic decoding using sum-product or min-sum decoding algorithms at the highest-order information bits in RM codes. Numerical results show that, compared with the traditional majority-logic decoding algorithm, the modified majority-logic decoding method proposed in this study can reduce the bit error rate (BER) considerably. This task is accomplished without incurring a significant increase in decoding complexity because soft-decision decoding is used only at the highest information bits of RM codes, whereas majority-logic decoding is still applied at the other orders of information bits.

1. Introduction

Digital communication techniques have enabled impressive progress in the technology industry in recent years. Transmission of images and multimedia requires a digital conversion process with almost no errors occur during the transmission and reading of digital signals. In this process, error-correcting codes (ECCs) [1] play an important role by performing special mathematical processing of digital signals and minimizing errors subject to various types of noise, distortion and interference during transmission [2], [3]. Many ECC techniques have been proposed for various communication systems, including deep-space communication, asymmetric digital subscriber lines, mobile communications and satellite communications [4], [5].

From the viewpoint of the construction of codes, ECC can be classified into three categories: codes on algebra,

codes on graph and codes on geometry [6], [7]. On the other hands, ECC decoding algorithms can be divided into hard-decision decoding, soft-decision decoding, and iterative-decision decoding algorithms [8], [9]. This paper examines binary Reed-Muller (RM) decoding algorithms [10] with low bit error rates and reduced decoding complexity.

Binary Reed-Muller decoding errors often occur in high-order information bits when traditional majority-logic decoding [11], [12] is applied. To address this problem, this paper proposes a modified majority-logic decoding algorithm that reduces the decoding error of binary Reed-Muller codes at the highest information bits without increasing the overall decoding complexity. Soft-decision decoding using sum-product or min-sum decoding algorithms [13][14] is applied in the highest-order information bits in an RM code whereas hard-decision decoding using traditional majority-logic decoding is used for other information bits. Numerical experiment results show that the proposed decoding method can reduce the bit error rate with almost no increase in decoding complexity.

The rest of the paper is organized as follows. In Section2, a basic description of error correcting codes is provided, and the construction of binary Reed-Muller codes and majority-logic decoding algorithms are reviewed. Section3 presents modified majority-logic decoding, i.e., factor graph decoding using sum-product and min-sum algorithms at the highest information bit in RM codes. In Section4, an additive white Gaussian noise (AWGN) channel is adopted to simulate the channel error rate. The simulation results show that compared with the traditional majority-logic decoding algorithm, the bit error rate of the proposed method can be reduced by a large margin. Section5 concludes this paper.

2. The Basic Theory of Reed-Muller Codes

2.1 Reed-Muller codes

We assume in what follows that the reader is familiar with the properties of linear block codes such as generator matrices or parity check matrices of linear block codes [5]. Let F_2^n be an n -dimensional vector space over the binary field $F_2 = \{0,1\}$. Given a binary vector $c = (c_0, \dots, c_{n-1}) \in F_2^n$, the support of c is the set of nonzero positions in c , which is denoted by $X(c) = \{i | c_i \neq 0\}$. The (Hamming) weight of c is defined as the size of $X(c)$, i.e., $|c| = |X(c)|$. For any subset $S \subset F_2^n$, the support of S is the union of the individual supports, $X(S) = \bigcup_{v \in S} X(v)$, and we will denote the minimum nonzero weight of vectors in $\omega(S)$.

A block code $C \in F_2^n$ of length n , size M , and minimum Hamming distance d is denoted by $C(n, M, d)$, where

$$d = d(C) = \min_{u \neq v \in C} |\{i: u_i \neq v_i, 0 \leq i \leq n-1\}|. \quad (1)$$

A linear code $C[n, k, d] = C(n, 2^k, d)$ over F_2 is a k -dimensional subspace of F_2^n such that the minimum distance of C is d . It is straightforward to show the minimum distance of a linear code C is equal to the nonzero minimum weight of C , i.e., $d(C) = w(c)$ [5]. The dual code of C , denoted by $C^\perp[n, n-k, d^\perp]$, is defined as

$$C^\perp = \{v \in F_2^n : u \cdot v = 0 \text{ for all } u \in C\},$$

where the multiplication denotes the inner product of u and v in F_2^n . The sum of two codes C_1 and C_2 is defined as

$$C = C_1 + C_2 = \{u + v : u \in C_1, v \in C_2\}.$$

If $C_1 \cap C_2 = \{0\}$, where 0 is the all-zero codeword, the resulting sum C is called the direct sum of C_1 and C_2 , denoted by $C = C_1 \oplus C_2$.

An encoding of $C[n, k]$ refers to any linear bijection from F_2^k to F_2^n . More precisely, a $k \times n$ matrix G that has a basis of the code C as its row vector is called a generator matrix of C . Usually, we denote a linear code with a generator matrix G by $C = \langle G \rangle$. Similarly a parity check matrix H of a linear code C is the counterpart for $C^\perp$, i.e., the row vectors of H form a basis of the dual code $C^\perp$. For any vector $e \in F_2^n$, the set $e + C = \{e + c : c \in C\}$ is called a coset of C or a translate of C . Each coset of $C[n, k]$ contains 2^k vectors and any two cosets are either disjoint or identical. Since $C[n, k]$ is a k -dimensional subspace of F_2^n , the set F_2^n can be partitioned into 2^{n-k} disjoint cosets of C .

2.2 Boolean polynomials and Reed-Muller codes

A Boolean function f of m variables $v_1, \dots, v_m$ is a function $f(v_1, \dots, v_m)$ from F_2^m to F_2 . Usually, a Boolean function $f(v_1, \dots, v_m)$ is specified by a truth table containing $m+1$ rows where the first m rows form an $m \times 2^m$ matrix whose columns are the 2^m binary m -tuples and last row gives the values of f at all of its 2^m arguments. If these 2^m arguments are listed in a fixed ordering, a length 2^m vector f can be associated with each Boolean function f ; indeed the vector f is simply the bottom row of the truth table for f . In this paper, the natural ordering of $v_1, \dots, v_m$, numbered from 0 to $2^m - 1$ in a base-2, is used to represent a vector of length 2^m . A Boolean function $p = v_{i_1} v_{i_2} \dots v_{i_s}$ that is a product of s distinct v_{i_l} , $1 \leq l \leq s$, is called a Boolean monomial of degree s . Its corresponding vector $p = v_{i_1} v_{i_2} \dots v_{i_s}$ of length 2^m is called a monomial vector, obtained from the componentwise product of those first-order monomial vector v_{i_l} , $1 \leq l \leq s$. The value of the Boolean monomial $p = v_{i_1} v_{i_2} \dots v_{i_s}$ is equal to 1 if and only if all s variables are equal to 1, i.e., $v_{i_1} = v_{i_2} = \dots = v_{i_s} = 1$. In other words, the Boolean monomial p is equal to 1 for all combinations of $(m-s)$ variables $v_{j_1}, v_{j_2}, \dots, v_{j_{m-s}}$, where $\{j_1, \dots, j_{m-s}\} \cup \{i_1, \dots, i_s\} = \{1, \dots, m\}$. As there are 2^{m-s} arguments for fixed values $v_{i_1} = v_{i_2} = \dots = v_{i_s} = 1$ and $x_{j_1} \in F_2, x_{j_2} \in F_2, \dots, x_{j_{m-s}} \in F_2$ in the truth table, the Hamming weight of a monomial vector p of length 2^m is thus equal to 2^{m-s} .

The zero order Boolean monomial is the all-one vector $1 \dots 1$, denoted by v_0 . There are m first-order monomials $v_1, v_2, \dots, v_m$. To form a second-order monomial, two randomly chosen first-order monomials can be multiplied by each other to obtain $v_i \cdot v_j$, $1 < i < j < m$. We can thus form a s -order monomial, by randomly multiplying s first-order monomials and there are $\binom{m}{s}$ s -order Boolean monomials of degree s in m variables. In total, there are $\binom{m}{0} + \binom{m}{1} + \binom{m}{2} + \dots + \binom{m}{m} = 2^m$ monomials in m variables. Any linear combination of these Boolean monomials is called a Boolean polynomial. Since these 2^m monomials are linearly independent, it follows that the vectors with which they are associated are also linearly independent. In other words, there are 2^{2^m} Boolean polynomials f in m variables and any vector f of length 2^m associated to the Boolean polynomial f can be written as the linear combinations of these 2^m monomial vectors as follows,

$$f = a_0 1 + a_1 v_1 + \dots + a_m v_m + a_{1,2} v_1 v_2 + \dots + a_{1,2,\dots,m} v_1 v_2 \dots v_m.$$

2.3 Majority-logic decoding of Reed-Muller codes

The concept of majority-logic decoding in RM codes, also known as Reed decoding, involves forming multiple check-sums for any information bit at each decoding step. In general, decoding algorithms using majority-logic decoding have low decoding complexity with suboptimal performance.

Consider the r -th order RM code, $RM(r, m)$. Let

$$a = (a_0, a_1, \dots, a_i, \dots, a_{1,2}, \dots, a_{i_1, \dots, i_r}, \dots, a_{m-r+1, \dots, m})$$

be the information bits to be encoded, where a_0 is the coefficient for the zero-order monomial vector v_0 , a_i are the coefficients for the first-order monomial vectors v_i , $1 \leq i \leq m$, and $a_{i_1, \dots, i_r}$ are the coefficients for the r -order monomial vectors $v_{i_1} v_{i_2} \dots v_{i_r}$, $1 \leq i_1 < i_2 < \dots < i_r \leq m$. Any codeword $c = (c_0, c_1, \dots, c_{n-1})$ of $RM(r, m)$ can be represented by

$$c = a_0 v_0 + \sum_{1 \leq i_1 \leq m} a_{i_1} v_{i_1} + \sum_{1 \leq i_1 < i_2 \leq m} a_{i_1, i_2} v_{i_1} v_{i_2} + \dots + \sum_{1 \leq i_1 < i_2 < \dots < i_r \leq m} a_{i_1, i_2, \dots, i_r} v_{i_1} v_{i_2} \dots v_{i_r}. \quad (2)$$

Let the received vector be $x = (x_0, x_1, \dots, x_{n-1})$. Decoding of RM code consists of $r+1$ steps and is conducted step-by-step, from the highest order of information bits to the lowest order. First, the information bits $a_{i_1, i_2, \dots, i_r}$ corresponding to the monomial vectors $v_{i_1} v_{i_2} \dots v_{i_r}$ of degree r are decoded, based on their check-sums formed from the received bits in $x^{(0)} = (x_0^{(0)}, x_1^{(0)}, \dots, x_{n-1}^{(0)}) = x$. Then, the decoded information

bits $a_{i_1, i_2, \dots, i_r}$ from the first step of decoding are used to modify $x^{(0)}$, thus producing the next modified received vector $x^{(1)} = (x_0^{(1)}, x_1^{(1)}, \dots, x_{n-1}^{(1)})$ which is then used for the information bits $a_{i_1, i_2, \dots, i_{r-1}}$ corresponding to the monomial vectors $v_{i_1} v_{i_2} \dots v_{i_{r-1}}$ of degree $r-1$ in the second step of decoding. This step-by-step decoding process, called $(r+1)$ -step majority-logic decoding, continues until the last information bit a_0 corresponding to the zero-order all-one vector v_0 is decoded.

It is necessary to know how to form the check-sums for decoding at each step, and to do so, parity checks are required at each order. For the information bit $a_{i_1, i_2, \dots, i_r}$ of degree r , the set S is defined as follows

$$S = \{c_{i_1-1} 2^{i_1-1} + c_{i_2-1} 2^{i_2-1} + \dots + c_{i_r-1} 2^{i_r-1}\}$$

where $1 \leq i_1 \leq i_2 \leq \dots \leq i_r \leq m$. In the above, $c_{i_j-1} \in \{0, 1\}$. Subsequently, the set E of integers is defined as

$$E = \{0, 1, \dots, m-1\} \setminus \{i_1-1, i_2-1, \dots, i_r-1\} \\ = \{j_1, j_2, \dots, j_{m-r}\}$$

and the complement of set S is defined as

$$S^c = d_{j_1} 2^{j_1} + d_{j_2} 2^{j_2} + \dots + d_{j_{m-r}} 2^{j_{m-r}}$$

Where $d_{j_t} \in \{0, 1\}$. Note that there are 2^{m-r} nonnegative integers in S^c , and $S \cap S^c = \{0\}$. For each integer $q_i \in S^c$, we form the following set of integers

$$B_i = q_i + S = \{q_i + s : s \in S\}.$$

Then, the checksum of the received sequence can be calculated by

$$A_i^{(0)} = \sum_{t \in B_i} x_t^{(0)}$$

for each B_i . Because there are 2^{m-r} integers q_i in S^c , we can form 2^{m-r} check-sums for decoding each information bit $a_{i_1, i_2, \dots, i_r}$ of degree r by majority votes.

There are $\binom{m}{r}$ information bits of r -th order in Reed-Muller codes. Assume the decoded information bits of degree r are $a_{i_1, i_2, \dots, i_r}^*$. We then form the received vector

$$x^{(1)} = x^{(0)} - \sum_{1 \leq i_1 < i_2 < \dots < i_r \leq m} a_{i_1, i_2, \dots, i_r}^* v_{i_1} \dots v_{i_r}. \quad (3)$$

We continue this process on the modified received vector $x^{(1)}$ for information bits $a_{i_1, i_2, \dots, i_{r-1}}$ of degree $r-1$ with the same procedure and different S and S^c until to the information bit a_0 . Readers can refer to [5, Chapter 4] for more information about the constructions and the majority-logic decoding of Reed Muller codes.

3. Modified Majority-Logic Decoding of Reed-Muller Codes

In this section, the sum-product and min-sum algorithms on the factor graph are briefly reviewed. Then the $RM(2, 3)$ code is used to demonstrate how the check-sums of the highest information bits of an arbitrary $RM(r, m)$ codes can be transformed into a fact graph. Finally, a modified majority-logic decoding using sum-product and min-sum algorithms at the highest information bit in RM codes is presented.

3.1 Modified soft decoding for highest order information bits

We use the $RM(2, 3)$ code to demonstrate how the check-sums of the highest information bits of an arbitrary $RM(r, m)$ code can be transformed into a fact graph. There are 7 information bits $a_0, a_1, a_2, a_3, a_{1,2}, a_{1,3}, a_{2,3}$ in $RM(2, 3)$ which is encoded into a codeword

$$(c_0, \dots, c_7) = a_0 v_0 + \sum_{1 \leq i_1 < 3} a_{i_1} v_{i_1} + \sum_{1 \leq i_1, i_2 < 3} a_{i_1, i_2} v_{i_1} v_{i_2}.$$

These code bits $(c_0, \dots, c_7)$ and highest-order

information bits $a_{1,2}, a_{1,3}, a_{2,3}$ form the eight code-bit nodes and three information-bit nodes in Fig. 1. The two check-sums for the information bit $a_{1,2}$ are

$$c_0 + c_1 + c_2 + c_3, \quad c_4 + c_5 + c_6 + c_7,$$

Similarly, the two check-sums for $a_{1,3}$ and $a_{2,3}$ are

$$c_0 + c_1 + c_4 + c_5, \quad c_2 + c_3 + c_6 + c_7,$$

and

$$c_0 + c_2 + c_4 + c_6, \quad c_1 + c_3 + c_5 + c_7,$$

which describe the connection between the information-bit node, the code-bit nodes c , and these two check nodes. The connections of these nodes form the upper part of the factor graph in Fig. 1.

Fig. 1. A factor graph of three highest information bits in RM(2,3).

These eight check-sums describe the connection between the information-bit nodes $a_{1,3}, a_{2,3}$, the code-bit nodes c , and these six check nodes. The connections of these nodes form the bottom part of the factor graph in Fig. 1. Based on the factor graph in Fig. 1, we can form a parity check matrix whose rows are consisting Fig. 2.

$$H_{RM} = \begin{array}{c|ccc} & \text{code-bit nodes} & \text{information-bit nodes} & \\ & c_0 & c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & a_{1,2} & a_{1,3} & a_{2,3} \\ \hline 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}$$

Fig. 2. A parity check matrix for the high information bits in RM(2,3).

For an arbitrary RM(r, m) code, there are $\binom{m}{r}$ highest-order information bits, 2^m code bits, and 2^{m-r} check-sums for each highest-order information bit. In

other words, we have $\binom{m}{r} \cdot 2^{m-r}$ check-sums in total. Just as the case of the RM(1,3) code with a factor graph in Fig.

1, we can then form a factor graph for the RM(r, m) code with $L = \binom{m}{r}$ information-bit nodes, $N = 2^m$ code-bit nodes, and $M = \binom{m}{r} \cdot 2^{m-r}$ check nodes, where the edges

connecting between the check nodes and bit nodes, including information-bit nodes and code-bit nodes, follow

the rules of these $\binom{m}{r} \cdot 2^{m-r}$ check-sums. In other words,

a factor graph of decoding the highest-order information bits of RM(r, m) is a bipartite graph that consists of two kinds

of nodes: bit nodes and check nodes, where bit nodes include both information-bit nodes and code-bit nodes. Equivalently, just as the RM(2,3), we can form a $M \times (N + L)$ parity check matrix whose rows are consisting of M check sums for the highest information bits in RM(r, m).

In general, the errors often occur at the highest order information bits in the Reed-Muller code decoding process when an majority-logic algorithm is applied. Therefore, our goal is to upgrade decoding accuracy at the highest-order in order to improve the performance of RM decoding algorithms. As shown in Fig. 1, the information transmission process includes three types of nodes, namely, information-bit nodes, code-bit nodes, and check nodes, and the key to decoding lies in the computation of information transmission between these nodes. Fig. 1 is applied to explain how the conventional majority-decision decoding algorithm can exhibit the relations between the check nodes and bit nodes. This factor graph for the highest information bits of RM(2,3) codes clearly describes how decoding can be accomplished for any RM(r, m) codes using message passing algorithms, such as sum-product or min-sum algorithms.

Next, we introduce the proposed modified majority logic algorithm, including the following five steps.

(1) Initialization: All messages passing from the bit node to the check nodes are initialized with the LLR vector from the channel model.

(2) Bit nodes to check nodes: The message from the code-bit node n , $0 \leq n \leq N - 1$ to the check node m , $0 \leq m \leq M - 1$ is obtained by summing over all the check nodes except the m -th one (cf. also Fig.2).

$$\lambda_{n \rightarrow m} = L_c \cdot x_n + \sum_{m' \in N_c(n) \setminus \{m\}} \Lambda_{m' \rightarrow n}. \quad (4)$$

Fig. 3. Code-bit node to check node.

(3) Check nodes to bit nodes by sum-product: The message out of check node is then computed as the product of all messages from the information-bit nodes and code-bit nodes except the one into the check node.

The message from the check node to the information-bit node:

$$\begin{aligned} \Lambda_{m \rightarrow a_l} &= 2 \tanh^{-1} [\prod_{n' \in N_e(m)} \tanh(\frac{\lambda_{n' \rightarrow m}}{2})]. \end{aligned} \quad (5)$$

The message from the check node to the code-bit node:

$$\begin{aligned} \Lambda_{m \rightarrow n} &= 2 \tanh^{-1} [(\prod_{n' \in N_e(m) \setminus \{n\}} \tanh(\frac{\lambda_{n' \rightarrow m}}{2})) \\ &\quad \cdot (\tanh(\frac{\lambda_{a_l \rightarrow m}}{2}))]. \end{aligned} \quad (6)$$

The message passing procedure is shown in Fig. 4

Fig. 4. Message from check node to information-bit or code-bit node.

(4) Check nodes to bit nodes by min-sum: The message from the check node to information-bit node:

$$\begin{aligned} \Lambda_{m \rightarrow a_l} &= [\prod_{n' \in N_e(m)} \text{sign}(\lambda_{n' \rightarrow m}) \text{sign}(\lambda_{a_l \rightarrow m}) \\ &\quad \times \min[(\min_{n' \in N_e(m)} |\lambda_{n' \rightarrow m}|), |\lambda_{a_l \rightarrow m}|]]. \end{aligned} \quad (7)$$

The message from the check node to the code-bit node:

$$\begin{aligned} \Lambda_{m \rightarrow n} &= [\prod_{n' \in N_e(m)} \text{sign}(\lambda_{n' \rightarrow m}) \text{sign}(\lambda_{a_l \rightarrow m}) \\ &\quad \times \min[(\min_{n' \in N_e(m)} |\lambda_{n' \rightarrow m}|), |\lambda_{a_l \rightarrow m}|]]. \end{aligned} \quad (8)$$

(5) Termination: The information at the information-bit node is then computed as the sum of the message form all

check nodes: (cf. also Fig. 4)

$$a_l = \Lambda_{0 \rightarrow a_l} + \Lambda_{1 \rightarrow a_l} + \dots + \Lambda_{m \rightarrow a_l} + \dots + \Lambda_{(M-1) \rightarrow a_l}. \quad (9)$$

If this output value is greater than 0, the information bit

Fig. 5. Message at the information-bit node.

$\hat{a}_l$ is determined to be 0; otherwise, it is set as 1. After the highest-order information bits are decoded, these bits are used to update the receiver sequence and then the rest information bits are decoded by the conventional majority-logic decoding.

4. Simulation Results

We evaluate and compare the bit error rate (BER) performance in the AWGN channel between the conventional majority-logic decoding and the proposed modified majority-logic decoding. The encoded codeword from a RM code is first mapped to a signal constellation point, which is then transmitted through an AWGN channel with variance $\sigma^2 = N_0/2$. The simulation result in Fig. 6 shows that at $\text{BER}=10^{-5}$, the signal-to-noise ratio (SNR) of the modified min-sum or sum-product decoding is reduced by about 1.5 dB over the conventional majority-logic decoding for RM(1,4). Fig. 7 shows the BER performance for RM(3,7) and RM(4,7) which have the same code length. In Fig. 7, the solid line and the dotted line represent the BER numerical results of RM(4,7) and RM(3,7) respectively. It is seen from Fig. 7 that when the fixed bit error rate is 10^{-5} , the SNR of the proposed modified majority-logic decoding can be reduced by more than 1 dB over the conventional majority-logic decoding. Fig. 7 shows the BER performance for RM(2,5) and RM(3,7) which have the same rate 0.5. The numerical results show that the coding gain of the proposed modified majority-logic decoding over the conventional majority-logic decoding is more than 1 dB at $\text{BER}=10^{-5}$.

Fig. 6. BER performance of RM(1,4).

Fig. 7. BER performance of RM(3,7) and RM(4,7).

Fig. 8. BER performance of RM(2,5) and RM(3,7).

5. Conclusions

We propose to modify the majority-logic decision by using graphic min-sum and sum-product algorithms at the highest order in this paper. This method does not significantly increase the computational complexity, but the BER is lower than that of other majority-logic algorithms. We evaluate and compare the BER performance in the

AWGN channel between the conventional majority-logic decoding and the proposed modified majority-logic decoding. The encoded codeword from an RM code is first mapped to a signal constellation point, which is then transmitted through an AWGN channel with variance of $\sigma^2 = \frac{\sigma^2}{N_0/2}$. The numerical results show that the coding gain of the proposed modified majority-logic decoding over the conventional majority-logic decoding is more than 1 dB at BER=10⁻⁵.

References

- (1) MacWilliams F. J. and Sloane N. J.: 'The Theory of Error-Correcting Codes', (North-Holland, New York, 1978)
- (2) Wicker S. B.: 'Error Control Systems for Digital Communication and Storage', (Englewood Cliffs, NJ: Prentice Hall, 1995)
- (3) Lin S. and Costello D. J.: 'Error Control Coding: Fundamentals and Applications', (Englewood Cliffs, NJ: Prentice-Hall, 2004)
- (4) Berlekamp E. R., Peile R. E. and Pope S. P.: 'The application of error-control to communications', IEEE Commun. Mag, 1987, 25, (4), pp. 44-57
- (5) Costello D. J., Hagenauer J., Imai H. and Wicker S.B.: 'Applications of error-control coding', IEEE Trans. Inform. Theory, 1998, 44, (6), pp. 2531-2560
- (6) Richardson T. J., Shokrollahi M. A. and Urbanke R.L.: 'Design of capacity-approaching irregular low-density parity-check codes', IEEE Trans. Inform. Theory, 2001, 47, (2), pp. 619-639
- (7) Worthen A. P. and Stark W. E.: 'Unified design of iterative receivers using factor graphs', IEEE Trans. Inform. Theory, 2001, 47, (2), pp. 843-849
- (8) Kaneko T., Nishijima T. and Hirasawa S.: 'An improvement of soft-decision maximum-likelihood decoding algorithm using hard-decision bounded-distance decoding', IEEE Trans. Inform. Theory, 1997, 43, (4), pp. 1314-1319
- (9) Kschischang F. R., Frey B. J. and Loeliger H. A.: 'Factor graphs and the sum-product algorithm', IEEE Trans. Inform. Theory, 2001, 47, (2), pp. 569-584
- (10) Paterson K. G. and Jones A. E.: 'Efficient decoding algorithms for generalized Reed-Muller codes', IEEE Trans. Commun., 2000, 48, (8), pp. 1272-1285
- (11) Muller D. E.: 'Application of boolean algebra to switching circuits design and to error detection', IRE

Trans., 1954, 3, (3), pp. 6-12

- (12) Reed I. S.: 'A class of multiple-error-correcting codes and the decoding scheme', IEEE Trans. Inform. Theory, 1954, 4, (4), pp. 38-49
- (13) Chung S., Richardson T. and Urbanke R.: 'Analysis of sum-product decoding of low-density parity-check codes using a gaussian approximation', IEEE Trans. Inform. Theory, 2001, 47, (2), pp. 657-670
- (14) Loeliger H. A.: 'An introduction to factor graphs', IEEE Signal Process. Mag., 2004, 21, (1), pp. 28-41
- (15) Y. Ting-Ya and C. Houshou.: 'Modified majority logic decoding of Reed-Muller codes using factor graphs', IET Commun., vol.12, no.9, pp.1-4, Jun. 2018.