Constructions of Multiple Error Correcting WOM-Code

Hidetoshi USTUNOMIYA
School of Information Science,
Gifu University,
1-1 Yanagido, Gifu, 501-1193, JAPAN
Email: ustunomiya@kmb.info.gifu-u.ac.jp

Hiroshi KAMABE
Department of Information Science,
Gifu University,
1-1, Yanagido, Gifu, 501–1193, JAPAN
Email: kamabe@ieee.org

***Abstract*—WOM-codes are constrained codes for memory devices whose state transition is irreversible. If we use the code, n-bit data can be written more than once on write-once cells of length greater than n. Error correcting WOM-codes have also been proposed. In this paper, a method for constructing multiple error correcting WOM-codes by modifying and simplifying the previously proposed error correcting WOM-code is presented. Codes constructed with our method have better code rates for some cases than the previously proposed code for some cases and no restrictions on the number of correctable errors. A single-symbol and a two-symbols error correcting code are also proposed.**

I. Introduction

WOM-codes are constrained codes for memory devices whose state transition is irreversible. Typical examples of such devices are the CD-ROM and punch cards for computer programs. It was shown that even such devices can be used as a memory device that is rewritable more than once by using the WOM-code [1].

Flash memories are memory devices that can hold data but on which we can rewrite data only finitely many times. It is easy to change their states (the charge level) upward, but it is costly to decrease the levels. It has been shown that the WOM-code can be used as a modulation code for flash memories [2], [3].

There are several studies in the literature on error correcting WOM-codes since flash memories, like communication channels, are subject to some sources of errors, channels, e.g., programming errors and leakage of the electronic charge of a cell [4]–[6]. Yaakobi et al. proposed m-bit error correcting WOM-codes based on BCH codes, where m is $1, 2$, or 3 [6]. In this paper, a simple method for constructing multiple-bit error correcting WOM-codes and a one- or two-symbol error correcting WOM-code are proposed.

II. Background and Notations

In this paper, it is assumed that each cell can take only two different states, "0" and "1", that is, the memory device consists of single level cells (SLCs). Each cell can change its state from "0" to "1" easily, but it is costly to change its state from "1" to "0."

A memory state vector is a binary sequence of length n, and a data vector is a binary sequence of length k. A WOM-code is specified by an encoding map $\mathcal{E}_{C_W}$ and a decoding map $\mathcal{D}_{C_W}$. The decoding map $\mathcal{D}_{C_W} : \{0,1\}^n \to \{0,1\}^k$ obtains the data vector $\boldsymbol{v} \in \{0,1\}^k$ from the memory state vector $\boldsymbol{c} \in \{0,1\}^n$ so that $\boldsymbol{v} = \mathcal{D}_{C_W}(\boldsymbol{c})$. The encoding map $\mathcal{E}_{C_W} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n \bigcup \{E\}$ obtains the next cell state vector $\boldsymbol{c}' = \mathcal{E}_{\mathcal{C}_W}(\boldsymbol{v}, \boldsymbol{c}) \in \{0,1\}^n$ where $\boldsymbol{c}'$ should satisfy $\boldsymbol{v} = \mathcal{D}_{C_W}(\boldsymbol{c}') \in \{0,1\}^k$ and $c_i \leq c'_i$ for all $1 \leq i \leq n$ when $\boldsymbol{c}' \neq E$. The symbol E means that there are no next state vectors which are greater than the current state vector and which represent the given data vector.

The WOM-code rate R^t is given by

$$R^t = \frac{kt}{n} \tag{1}$$

where n is the length of the cell state vector, k the length of the data vector, and t the maximum number of rewrites by the WOM-code.

III. Single Bit Error Correcting WOM-Code

It is assumed that we have a WOM-code C_W with which we can write data on a WOM device t times [4].

Definition 1. By $WC[n, k, t]$, we mean a WOM-code C_W that is defined by two functions $\mathcal{E}_{C_W}$ and $\mathcal{D}_{C_W}$, and which ensures that any sequence of length t consisting of k-bit information vectors can be successively stored in a block of cells of length n.

Yaakobi et al. proposed multiple bit error correcting WOM-codes based on parity check equations where the number of error bits should be less than 3 [6]. A single bit error correcting WOM-code proposed by them has the following structure. A code word of the code consists of three sub-code words, c_A, c_B and c_C where c_A corresponds to a block of cells for storing information bits by using C_W, c_B a block of cells for storing parity bits, and c_C a block of cells of length t for storing parity bits for detecting a single error bit. Although the parity check equation for c_C checks the even parity, the parity check equations for c_B consist of a linear combination of elements for a finite field. These three sub-code words are used at most t times. Let r be the number of cells to be used to store $\lceil \log_2(n+1) \rceil$ bits t times. This corresponds to a size of c_B. The length of c_C is t. Let C'_W be another $WC[r, \lceil \log_2(n+1) \rceil, t]$

WOM-code consisting of an encoding function $\mathcal{E}_{C'_W}$ and a decoding function $\mathcal{D}_{C'_W}$.

[Encoding]

1) Let $\boldsymbol{v}$ be an information vector to be stored, and let $\boldsymbol{c}$ be a current cell state vector. A new code word $\boldsymbol{c}'$ obtained by calculating $\mathcal{E}_{C_W}(\boldsymbol{v}, \boldsymbol{c})$ is stored as c'_A.
2) Calculate a syndrome
$$\boldsymbol{s} = \sum_{i=0}^{n-1} c'_i \alpha^i,$$
where $\alpha \in GF(2^{\lceil \log_2(n+1) \rceil})$ is a primitive element.
3) $\mathcal{E}_{C'_W}(\boldsymbol{s}, c_A)$ is stored in the c_B part.
4) Calculate the parity (even parity) of c_A. If the parity is different from the previous one, then we write a new "1" in the c_C part.

Yaakobi et al. also proposed m error correcting WOM-codes for cases $m = 2$ and $m = 3$ using parity check equations on finite fields [6].

IV. Single Bit Error Correcting WOM-Code by copying

We propose a simple error correcting WOM-code where "simple" means that our codes depend only on a copy operation in encoding and a counting argument in decoding rather than calculations of the encoding and decoding functions of the underling WOM-code. The WOM-code rate of our code is better than that of the code proposed by Yaakobi et al. for $n < r$.

As mentioned above, a code word of our code consists of three parts:
$$\boldsymbol{c} \cdot \boldsymbol{c} \cdot \boldsymbol{p}.$$
The second part is just a copy of the first part. $\boldsymbol{p}$ is a block for t parity bits.

[Encoding]

A.1 Let $\boldsymbol{v}$ be the information vector to be stored. Calculate $\boldsymbol{c}'$ by
$$\boldsymbol{c}' = \mathcal{E}_{C_W}(\boldsymbol{v}, \boldsymbol{c})$$
where $\boldsymbol{c}$ is the current code word (the current state vector).

A.2 Calculate the even parity of $\boldsymbol{c}' \cdot \boldsymbol{p}$. If the result is 0, then $\boldsymbol{p}' \leftarrow \boldsymbol{p}$. If the result is 1, then the leftmost 0 bit in $\boldsymbol{p}$ is flipped. For both cases, the even parity of $\boldsymbol{c}' \cdot \boldsymbol{p}'$ is 0.

Now we consider a decoding of our code. We assume that $\boldsymbol{c}_0 \cdot \boldsymbol{c}_1 \cdot \boldsymbol{p}_1$ is retrieved.

[Decoding]

A.3 Suppose that the existence of an error in $\boldsymbol{c}_1 \cdot \boldsymbol{p}_1$ is detected by checking the even parity of $\boldsymbol{c}_1 \cdot \boldsymbol{p}_1$. Then the decoding algorithm outputs $\boldsymbol{c}_0$ because $\boldsymbol{c}_0$ should contain no error under the assumption that the number of error bits is at most 1.

A.4 Suppose that no error is detected in $\boldsymbol{c}_1 \cdot \boldsymbol{p}_1$. Then the decoding algorithm outputs $\boldsymbol{c}_1$ because the only one bit error is in $\boldsymbol{c}_0$ but not in $\boldsymbol{c}_1$.

V. Double Bit Error Correcting WOM-Code by Copying

A code word of our code consists of 5 parts: $\boldsymbol{c} \cdot \boldsymbol{c} \cdot \boldsymbol{p} \cdot \boldsymbol{c} \cdot \boldsymbol{p}$. Assume that the current state of cells is $\boldsymbol{c} \cdot \boldsymbol{c} \cdot \boldsymbol{p} \cdot \boldsymbol{c} \cdot \boldsymbol{p}$.

[Encoding]

B.1 Let $\boldsymbol{v}$ be the information vector to be stored, and let $\boldsymbol{c}'$ be a vector given by $\boldsymbol{c}' = \mathcal{E}_{C_W}(\boldsymbol{v}, \boldsymbol{c})$.

B.2 $\boldsymbol{p}'$ is determined so that the even parity of $\boldsymbol{c}' \cdot \boldsymbol{p}'$ is 0 by flipping the leftmost '0' bit in $\boldsymbol{p}$ if necessary.

B.3 Store the vector $\boldsymbol{c}' \cdot \boldsymbol{c}' \cdot \boldsymbol{p}' \cdot \boldsymbol{c}' \cdot \boldsymbol{p}'$.

Suppose that $\boldsymbol{c}_0 \cdot \boldsymbol{c}_1 \cdot \boldsymbol{p}_1 \cdot \boldsymbol{c}_2 \cdot \boldsymbol{p}_2$ is retrieved.

[Decoding]

B.4 Compare $\boldsymbol{c}_1 \cdot \boldsymbol{p}_1$ and $\boldsymbol{c}_2 \cdot \boldsymbol{p}_2$. If there are two different bits, then the decoding algorithm outputs $\boldsymbol{c}_0$.

B.5 Suppose that the even parity of $\boldsymbol{c}_1 \cdot \boldsymbol{p}_1$ is not 0. Then there must be at most 1 error in $\boldsymbol{c}_0 \cdot \boldsymbol{c}_2 \cdot \boldsymbol{p}_2$. Apply the decoding algorithm for the single-bit error correcting WOM-code to $\boldsymbol{c}_0 \cdot \boldsymbol{c}_2 \cdot \boldsymbol{p}_2$.

VI. m Bit Error Correcting WOM-Code

Now we can describe an m-bit error correcting WOM-code. A code word of our code is represented by $\boldsymbol{c} \cdot (\boldsymbol{c} \cdot \boldsymbol{p})^m$, where $(\boldsymbol{c} \cdot \boldsymbol{p})^m$ is the m-th repetition of $\boldsymbol{c} \cdot \boldsymbol{p}$, and $\boldsymbol{p}$ is defined similarly as in the previous sections.

We only consider the decoding operation here. We assume that we are given $\boldsymbol{c}_0 \cdot \boldsymbol{c}_1 \cdot \boldsymbol{p}_1 \cdot \boldsymbol{c}_2 \cdot \boldsymbol{p}_2 \cdots \boldsymbol{c}_m \cdot \boldsymbol{p}_m$ as a retrieved code word.

[Decoding]

C.1 If $m = 1$ or $m = 2$, then apply the algorithms given in the previous sections.

C.2 If $\boldsymbol{c}_i \cdot \boldsymbol{p}_i$ has an error (the even parity of $\boldsymbol{c}_i \cdot \boldsymbol{p}_i$ is not 0), then apply this algorithm to $\boldsymbol{c}_0 \cdot \boldsymbol{c}_1 \cdot \boldsymbol{p}_1 \cdots \boldsymbol{c}_{i-1} \cdot \boldsymbol{p}_{i-1} \cdot \boldsymbol{c}_{i+1} \cdot \boldsymbol{p}_{i+1} \cdots \boldsymbol{c}_m \cdot \boldsymbol{p}_m$ with $m \leftarrow m-1$. Since the reduced code word contains at most $m-1$ errors, this algorithm is valid for this case.

C.3 Suppose that all even parities of $\boldsymbol{c}_i \cdot \boldsymbol{p}_i$, $i = 1, 2, \ldots, m$ are 0. If there are i and j such that $\boldsymbol{c}_i \cdot \boldsymbol{p}_i \neq \boldsymbol{c}_j \cdot \boldsymbol{p}_j$, then apply this algorithm to $\boldsymbol{c}_0 \cdot \boldsymbol{c}_1 \cdot \boldsymbol{p}_1 \cdots \boldsymbol{c}_{i-1} \cdot \boldsymbol{p}_{i-1} \cdot \boldsymbol{c}_{i+1} \cdot \boldsymbol{p}_{i+1} \cdots \boldsymbol{c}_{j-1} \cdot \boldsymbol{p}_{j-1} \cdot \boldsymbol{c}_{j+1} \cdot \boldsymbol{p}_{j+1} \cdots \boldsymbol{c}_m \cdot \boldsymbol{p}_m$ with $m \leftarrow m-2$. Since the reduced code word contains at most $m-2$ errors, this algorithm is valid for this case.

VII. WOM-Code Rate

WOM-code rates of the YSVW code and proposed code are given as follows. Note that the proposed code can be defined

	$m=1$	$m=2$	$m=3$
YSVW code	$\frac{kt}{n+r+t}$	$\frac{kt}{n+2r+2t}$	$\frac{kt}{n+3r+4t}$
Proposed code	$\frac{kt}{2n+t}$	$\frac{kt}{3n+2t}$	$\frac{kt}{4n+3t}$

TABLE I
WOM-Code rate

for any m. From this we can see that the WOM-code rate of the proposed code is better than that of the YSVW code for $n < r$.

VIII. Symbol Error Correcting WOM-Codes

The results given in the previous section can be extended to symbol error correcting WOM-codes in a straightforward manner. First a one symbol error correcting coding scheme is introduced as a straightforward extension of the one-bit error detection scheme. Then symbol error correcting WOM-codes are introduced based on the idea of syndromes but not of duplications. The schemes are rather similar to the methods of Yaakobi et al. [6].

In the following it is assumed that symbols are elements in $GF(2^d)$ and that "symbol" means d consecutive cells, not a symbol in binary data sequences. The $WC[n,k,t]$ WOM-code is again used to store k-bit data t times.

A. One symbol error detecting WOM-code

A code word (or a sequence of cells) of this method is

$$\boldsymbol{a}_0\boldsymbol{a}_1\cdots\boldsymbol{a}_{\nu-1}f_0f_1\cdots f_{q-1}$$

where n is equal to νd and q cells in the second part of the code word are used to write d bits of parity check data t times. A symbol $\boldsymbol{a}\in GF(2^d)$ is also regarded as a binary sequence of length d. Note that no symbol corresponds to any data bits of fixed length. A $WC[q,d,t]$ WOM-code is used in this scheme for the second part.

[Encoding]

- D.1 A k-bit data is written on the first part of a code by the $WC[n,k,t]$ WOM-code. The code word is divided into ν sub-blocks of length d, $\boldsymbol{a}_0,\boldsymbol{a}_1,\ldots,\boldsymbol{a}_{\nu-1}$.
- D.2 Calculate a syndrome $D=\boldsymbol{a}_0+\boldsymbol{a}_1+\cdots+\boldsymbol{a}_{\nu-1}$ where "+" means an addition in $GF(2^d)$.
- D.3 The syndrome D is written on the second half of the code word by using the $WC[q,d,t]$ WOM-code.

[Decoding]

- D.4 Calculate syndrome $D'=\boldsymbol{a}'_0+\boldsymbol{a}'_1+\cdots+\boldsymbol{a}'_{\nu-1}$, where $\boldsymbol{a}'_i$ means the i-th symbol in the first part of a retrieved data.
- D.5 Let D'' be a syndrome in the second half of the retrieved codeword. If $D'\neq D''$, then we conclude that there are some errors.

B. One symbol error correcting WOM-code

A code word (sequence of cells) of this scheme consists of the following four parts:

1) n cells for storing a k bit data vector t times. A $WC[n,k,t]$ WOM-code is used.
2) q cells for storing the first syndrome bits. A $WC[q,d,t]$ WOM-code is used.
3) q cells for storing the second syndrome bits. The $WC[q,d,t]$ WOM-code is also used.
4) q cells for storing the third syndrome bits D, where D is the syndrome of the previous two syndrome parts. The $WC[q,d,t]$ WOM-code is also used.

[Encoding]

- E.1 The first part of the retrieved vector is obtained by encoding a k-bit data vector with the $WC[n,k,t]$ WOM-code. This part is divided into ν blocks of length d,
$$\boldsymbol{a}_0,\boldsymbol{a}_1,\ldots,\boldsymbol{a}_{\nu-1}$$
- E.2 The first syndrome is obtained by
$$\boldsymbol{s}_1=\sum_{i=0}^{\nu-1}\boldsymbol{a}_i\beta^i \tag{2}$$
where β is a primitive element of $GF(2^d)$. $\boldsymbol{s}_1$ is encoded by the $WC[q,d,t]$ WOM-code and stored in the second part of the code word.
- E.3 The second syndrome is obtained by
$$\boldsymbol{s}_2=\sum_{i=0}^{\nu-1}\boldsymbol{a}_i\beta^{2i}. \tag{3}$$
This part is also encoded by the $WC[q,d,t]$ WOM-code and stored in the third part of the code word.
- E.4 Let B be a code word consisting of the second and the third parts, and assume that B is divided into x parts of length d.
$$B=\boldsymbol{b}_0\boldsymbol{b}_1\cdots\boldsymbol{b}_{x-1}.$$
Calculate
$$\boldsymbol{s}_3=\sum_{i=0}^{x-1}\boldsymbol{b}_i. \tag{4}$$
- E.5 Encode $\boldsymbol{s}_3$ by the $WC[q,d,t]$ WOM-code and the resulting sequence should be the last part of the code word.

Suppose that it is known that there is one symbol error in the first part of a retrieved code word, $\boldsymbol{a}'_0\boldsymbol{a}'_1\cdots\boldsymbol{a}'_{\nu-1}$. Since there is no error in the remaining parts, syndromes $\boldsymbol{s}''_1$ and $\boldsymbol{s}''_2$ decoded from the remaining parts contain no error. Therefore $S'_1=\boldsymbol{s}'_1+\boldsymbol{s}''_1$ and $S'_2=\boldsymbol{s}'_2+\boldsymbol{s}''_2$ should be nonzero, where $\boldsymbol{s}'_1$ and $\boldsymbol{s}'_2$ are syndromes of $\boldsymbol{a}'_0\boldsymbol{a}'_1\cdots\boldsymbol{a}'_{\nu-1}$ calculated according to the parity check equations (2) and (3), respectively. Assume that there is one symbol error, $\boldsymbol{e}_i$, in the first part. Then

$$S'_1=\boldsymbol{e}_i\beta^i,\qquad S'_2=\boldsymbol{e}_i\beta^{2i}. \tag{5}$$

Because $S'_2/S'_1=\beta^i$, i can be resolved. Then $\boldsymbol{e}_i$ is given by

$$\boldsymbol{e}_i=\frac{S'_1}{\beta^i}.$$

Therefore, the following decoding procedure can find any one symbol error and correct it.

[Decoding]

- E.6 Check if there is one symbol error in the second, third or fourth parts by calculating syndromes of the second and third parts.
- E.7 If a one symbol error is detected in these three parts, then decode the first part of the retrieved code word

with the decoding function of the $WC[n,k,t]$ WOM-code and output the resulting data. Terminate this algorithm.

E.8 Suppose that there is no error in these three parts, that is,

$$\sum_{i=0}^{x-1} \boldsymbol{b}'_i = \boldsymbol{s}'_3$$

where $\boldsymbol{b}'_i$ are blocks in the second and third parts of the retrieved code word, and $\boldsymbol{s}'_3$ is the syndrome decoded from the fourth part of the code word. Calculate the following two syndromes from the first part of the code word:

$$\begin{aligned}\boldsymbol{s}'_1 &= \sum_{i=0}^{\nu-1} \boldsymbol{a}'_i \beta^i \\ \boldsymbol{s}'_2 &= \sum_{i=0}^{\nu-1} \boldsymbol{a}'_i \beta^{2i}.\end{aligned}$$

E.9 Let $\boldsymbol{s}''_1$ and $\boldsymbol{s}''_2$ be syndromes obtained from the first and second parts of the retrieved code word by decoding the $WC(q,d,t)$ WOM-code. Calculate S'_1 and S'_2 as follows:

$$\begin{aligned}S'_1 &= \boldsymbol{s}'_1 + \boldsymbol{s}''_1, \\ S'_2 &= \boldsymbol{s}'_2 + \boldsymbol{s}''_2.\end{aligned}$$

E.10 If $S'_1 = 0$ and $S'_2 = 0$, then there should be no error. Terminate this algorithm.

E.11 Find the index i such that

$$\frac{S'_2}{S'_1} = \beta^i.$$

Then the value of the symbol error is given by

$$\boldsymbol{e}_i = S'_1/\beta^i.$$

E.13 Correct the i-th symbol by

$$\boldsymbol{a}'_i + \boldsymbol{e}_i$$

and output the resulting code word.

C. *Two symbol error correcting WOM-code*

A code word of this coding scheme is divided into 7 parts.

- Part 1: n cells for storing k-bit data vector t times. This part is divided into ν blocks of length d,

  $$\boldsymbol{a}_0 \boldsymbol{a}_1 \cdots \boldsymbol{a}_{\nu-1}.$$

  A $WC[n,k,t]$ WOM-code is used for this part.
- Part 2: q cells for storing d-bit data vector t times. The d-bit vector is given by the parity check equation (2). This part is used to store the d-bit data vector t times with the $WC[q,d,t]$ WOM-code.
- Part 3: q cells for storing d-bit data vectors t times. The d-bit data vector is given by (3).
- Part 4: Suppose that a concatenation of the second and third parts is divided into x blocks of length d:

  $$\boldsymbol{b}_0 \boldsymbol{b}_1 \cdots \boldsymbol{b}_{x-1}.$$

  A syndrome is calculated according to (4) and the syndrome is written in this section of the code word t times.
- Part 5: q cells for storing d-bit data vector t times. The d-bit data vector is given by

  $$\boldsymbol{s}_3 = \sum_{i=0}^{\nu-1} \boldsymbol{a}_i \beta^{3i}. \tag{6}$$
- Part 6: q cells for storing d-bit data vector t times. The d-bit data vector is given by

  $$\boldsymbol{s}_4 = \sum_{i=0}^{\nu-1} \boldsymbol{a}_i \beta^{4i}. \tag{7}$$
- Part 7: Suppose that a concatenation of the fifth and sixth parts is divided into x blocks of length d,

  $$\overline{\boldsymbol{b}}_0 \overline{\boldsymbol{b}}_1 \cdots \overline{\boldsymbol{b}}_{x-1}.$$

  A syndrome is calculated according to (4) and the syndrome is written in this section t times.

Since the above definitions of the 7 parts of the code word also describe an encoding procedure, the procedure is described simply as follows.

[Encoding]

F.1 A-k bit data vector is stored in the first part by using the $WC[n,k,t]$ WOM-code.

F.2 Syndromes for the 2nd, 3rd, 5th, and 6th parts are calculated and stored in these parts by using the $WC[q,d,t]$ WOM-code.

F.3 A syndrome for a concatenation of the 2nd and 3rd parts is calculated and stored in the 4th part by using the $WC[q,d,t]$ WOM-code. Similarly, a syndrome for the 5th and 6th parts is calculated and stored in the 7th part.

Let $\boldsymbol{r}'_1 \boldsymbol{r}'_2 \cdots \boldsymbol{r}'_7$ be a retrieved code word, where $\boldsymbol{r}'_i$ corresponds to the i-th part of the code word. Let $\boldsymbol{a}'_0 \boldsymbol{a}'_1 \cdots \boldsymbol{a}'_{\nu-1}$ be the first part of a retrieved sequence, that is, $\boldsymbol{r}'_1 = \boldsymbol{a}'_1 \boldsymbol{a}'_2 \cdots \boldsymbol{a}'_{\nu-1}$. Syndromes $\boldsymbol{s}'_1, \boldsymbol{s}'_2, \boldsymbol{s}'_3$, and $\boldsymbol{s}'_4$ are calculated from $\boldsymbol{r}'_1$ according to parity check equations (2), (3), (4), and (7), respectively. Let $\boldsymbol{s}''_1, \boldsymbol{s}''_2, \boldsymbol{s}''_3$, and $\boldsymbol{s}''_4$ be syndromes obtained from the 2nd, 3rd, 5th, and 6th parts, respectively, and let S_1, S_2, S_3, and S_4 be vectors given by

$$\begin{aligned}S_1 &= \boldsymbol{s}'_1 + \boldsymbol{s}''_1, \\ S_2 &= \boldsymbol{s}'_2 + \boldsymbol{s}''_2, \\ S_3 &= \boldsymbol{s}'_3 + \boldsymbol{s}''_3, \\ S_4 &= \boldsymbol{s}'_4 + \boldsymbol{s}''_4.\end{aligned}$$

From the above encoding procedure the following propositions can be proved.

Proposition 1. *If one of the following conditions holds, then there is no error in* $\boldsymbol{r}_1$.

1) $S_1 = S_2 = 0$.

2) $S_3 = S_4 = 0$.

Proof:

Suppose that there is one symbol error in $\boldsymbol{r}_1$ at i-th position with error symbol $\boldsymbol{e}_i$. Then it is clear that all of S_1, S_2, S_3, and S_4 are nonzero because

$$S_\ell = (a_i + a_i')\beta^{\ell i} = \boldsymbol{e}_i \beta^{\ell i} \neq 0, \qquad \ell = 1, 2, 3, 4,$$

where β is the primitive element used in the encoding procedure. Therefore, none of two conditions of this proposition holds.

Suppose that there two symbol errors, $\boldsymbol{e}_i$ and $\boldsymbol{e}_j, i \neq j$ in $\boldsymbol{r}_1$. Then

$$\begin{aligned} S_1 &= \boldsymbol{e}_i\beta^i + \boldsymbol{e}_j\beta^j = \beta^{i+x} + \beta^{j+y}, \\ S_2 &= \boldsymbol{e}_i\beta^{2i} + \boldsymbol{e}_j\beta^{2j} = \beta^{2i+x} + \beta^{3j+y}, \\ S_3 &= \boldsymbol{e}_i\beta^{3i} + \boldsymbol{e}_j\beta^{3j} = \beta^{3i+x} + \beta^{3j+y}, \\ S_4 &= \boldsymbol{e}_i\beta^{4i} + \boldsymbol{e}_j\beta^{4j} = \beta^{4i+x} + \beta^{4j+y}, \end{aligned}$$

where x and y are defined so that $\boldsymbol{e}_i = \beta^x$ and $\boldsymbol{e}_j = \beta^y$ hold, respectively. If $S_1 = \beta^{i+x} + \beta^{j+y} = S_2 = \beta^{2i+x} + \beta^{3j+y} = 0$, then both of the following equalities should hold:

$$\begin{aligned} i + x &= j + y, \\ 2i + x &= 2j + y, \end{aligned}$$

because the characteristic of the field is 2 and β is the primitive element. From these equalities it follows $i = j$ but this contradicts the assumeption $i \neq j$. Therefore we can conclude that $S_1 = S_2 = 0$ does not hold.

The second condition is proved similarly. ■

If there is no error in $\boldsymbol{r}_1$ but there are some errors in syndrome blocks $S_1 = S_2 = 0$ or $S_3 = S_4 = 0$ may happen.

Proposition 2. *Suppose that there are one or two errors in $\boldsymbol{r}_1$. The following conditions are equivalent:*

1) $S_2/S_1 = S_3/S_2 = S_4/S_3$;
2) *there is only one symbol error in $\boldsymbol{r}_1$.*

Proof: It is clear that the second condition means the first condition.

Suppose that there are two symbol errors $\boldsymbol{e}_i$ and $\boldsymbol{e}_j, i \neq j$ in $\boldsymbol{r}_1$ and the first condition holds. Let

$$\begin{aligned} S_1 &= \boldsymbol{e}_i\beta^i + \boldsymbol{e}_j\beta^j, \\ S_2 &= \boldsymbol{e}_i\beta^{2i} + \boldsymbol{e}_j\beta^{2j}, \\ S_3 &= \boldsymbol{e}_i\beta^{3i} + \boldsymbol{e}_j\beta^{3j}, \\ S_4 &= \boldsymbol{e}_i\beta^{4i} + \boldsymbol{e}_j\beta^{4j}. \end{aligned}$$

By simple calculations,

$$\begin{aligned} S_1S_3 &= \boldsymbol{e}_i^2\beta^{4i} + \boldsymbol{e}_j^2\beta^{4j} + \boldsymbol{e}_i\boldsymbol{e}_j(\beta^{i+3j} + \beta^{3i+j}), \\ S_2^2 &= \boldsymbol{e}_i^2\beta^{4i} + \boldsymbol{e}_j^2\beta^{4j}. \end{aligned}$$

If the first condition holds, then it should follow that $S_1S_3 = S_2^2$ and that $\beta^{i+3j} = \beta^{3i+j}$. However this is impossible when $i \neq j$.

Therefore when the second condition does not hold, the first condition also does not hold. ■

From these results, a decoding procedure is given as follows [Decoding]

- F.4 The algorithm for one symbol error detection in subsection III is applied to $\boldsymbol{r}_2'\boldsymbol{r}_3'\boldsymbol{r}_4'$. The algorithm is also applied to $\boldsymbol{r}_5'\boldsymbol{r}_6'\boldsymbol{r}_7'$.
- F.5 Suppose that two symbol errors are found (one symbol error in $\boldsymbol{r}_2'\boldsymbol{r}_3'\boldsymbol{r}_4'$ and another symbol error in $\boldsymbol{r}_5'\boldsymbol{r}_6'\boldsymbol{r}_7'$). Because there is no error in the first part of the retrieved code word, apply the decoding procedure of $WC[n, k, t]$ WOM-code to the first part and output the decoding result. Terminate this procedure.
- F.6 Suppose that a symbol error is detected in $\boldsymbol{r}_2'\boldsymbol{r}_3'\boldsymbol{r}_4'$ but no symbol error is detected in $\boldsymbol{r}_5'\boldsymbol{r}_6'\boldsymbol{r}_7'$. This is the case that no or one symbol error occurs in $\boldsymbol{r}_1'\boldsymbol{r}_5'\boldsymbol{r}_6'\boldsymbol{r}_7'$. Therefore, apply the algorithm for one-symbol error correction in subsection III to $\boldsymbol{r}_1'\boldsymbol{r}_5'\boldsymbol{r}_6'\boldsymbol{r}_7'$. and output a result. Terminate this procedure.
- F.7 Suppose that a symbol error is detected in $\boldsymbol{r}_5'\boldsymbol{r}_6'\boldsymbol{r}_7'$ but no symbol error is detected in $\boldsymbol{r}_2'\boldsymbol{r}_3'\boldsymbol{r}_4'$. Apply the algorithm for one-symbol error correcting in subsection III and output a result. Terminate this procedure.
- F.8 At this point, it is known that no one-symbol error is detected in $\boldsymbol{r}_2'\boldsymbol{r}_3'\boldsymbol{r}_4'$ and $\boldsymbol{r}_5'\boldsymbol{r}_6'\boldsymbol{r}_7'$. If $S_1 = S_2 = 0$ or $S_3 = S_4 = 0$, then from Proposition 1 it is known that there is no symbol error in $\boldsymbol{r}_1'$. Output a decoding result of $WC[n, k, t]$ WOM-code to $\boldsymbol{r}_1'$ and terminate this procedure.
- F.9 If $S_2/S_1 = S_3/S_2 = S_4/S_3$, then apply the decoding algorithm in subsection III to $\boldsymbol{r}_1'\boldsymbol{r}_2'\boldsymbol{r}_3'\boldsymbol{r}_4'$ and output a decoding result because there is only one symbol error from Proposition 2. Terminate this procedure.

ACKNOWLEDGEMENT

This work was supported by JSPS KAKENHI Grant Number 23560439.

REFERENCES

[1] R. L. Rivest and A. Shamir,"How to reuse a write-once memory,"in Information and Control, vol. 55, pp. 1-19, 1982.

[2] A. Jiang, V. Bohossian, and J. Bruck, "Floating codes for joint information storage in write asymmetric memories," in Proc. IEEE International Symposium on Information Theory(ISIT), Nice, France, 2007, pp.1166-1170.

[3] V. Bohossian, A. Jiang, and J. Bruck, "Buffer coding for asymmetric multi-mevel memory," in Proc. IEEE International Symposium on Information Theory(ISIT), Nice, France, 2007, pp.1186-1190.

[4] G. D. Cohen, P. Godlewski, and F. Merkx,"Linear binary code for write-once memories,"IEEE Trans. Inf. Theory, vol. IT–32, no. 5, pp. 697-700, Sep. 1986.

[5] A. Jiang, " On The generalization of error-correcting WOM-codes," " in Proc. IEEE International Symposium on Information Theory(ISIT), Nice, France, 2007, pp.1191-1395.

[6] E. Yaakobi, P. H. Siegel, A. Vardy and J. K. Wolf, "Multiple error-correcting WOM-codes," IEEE Trans. Inf. Theory, vol. IT-58, no. 4, pp. 2220–2230.