PAPER

On Multiple-Valued Separable Unordered Codes

Yasunori NAGATA[†] ***and*** **Masao MUKAIDONO**[††], ***Members***

SUMMARY In this paper, a new encoding/decoding scheme of multiple-valued separable balanced codes is presented. These codes have $2 \cdot m$ information digits and $m \cdot (R-2)$ check digits in radices $R \geqq 4$, $2 \cdot m - 1$ information digits and $m+1$ check digits in $R = 3$, where code-length $n = R \cdot m$. In actual use of code-lengths and radices, it is shown that the presented codes are relatively efficient in comparison with multiple-valued Berger codes which are known as optimal unordered codes. Meanwhile, the optimality of multiple-valued Berger codes is discussed.

key words: *balanced codes, Berger codes, constant weight codes, multiple-valued logic, uniderectional error detection*

1. Introduction

Multiple-valued error detecting codes are of great significance in improving the reliability of multiple-valued circuits and systems [1]. For instance, since multiple-valued unordered codes detect all unidirectional errors, the codes are used online concurrent error checking in systems [1],[4].

In an R-valued data word $(a_1 \cdots a_i \cdots a_n)$, both $a_i \rightarrow b_i$ $(a_i < b_i)$ and $a_i \rightarrow b_i$ $(a_i > b_i)$ errors are possible, but in any particular word all errors are of one type, and are called unidirectional errors. It has been found that these types of errors are predominant in VLSI circuits and memories in case of binary and some types of multiple-valued circuits [1]–[4].

The multiple-valued code C is called unordered when no codeword covers the other. Unordered codes detect all unidirectional errors of data words, and these have proven useful for designing failsafe, fault-secure, and self-checking logic circuits, namely, state assignments in fault tolerant and fail-safe sequential circuits [7]–[10].

Multiple-valued *separable* unordered codes may have applications not only to multiple-valued logic systems, but also to byte error control in binary systems [3].

In this paper, multiple-valued unordered codes, particularly, multiple-valued Berger codes and multiple-valued separable balanced codes are considered.

R-valued *balanced codes*, where each codeword contains equally many $0s, 1s \cdots, (R-1)'s$, are a class of multiple-valued constant weight codes. It can be pointed out that error checking circuits of these balanced codes are simpler than the circuits of Berger codes.

In Sect. 2, multiple-valued unordered codes: multiple-valued complemented duplication codes, multiple-valued Berger codes and multiple-valued constant weight codes, are outlined [1]–[6]. Continually, the code *efficiency* (optimality) of multiple-valued Berger codes is discussed. In Sect. 3, a class of multiple-valued constant weight codes, that is, $2 \cdot m$ information digits, $m \cdot (R-2)$ check digits R-valued separable balanced codes $(R \geqq 4)$ are presented. Similarly, it is shown that $2 \cdot m - 1$ information digits, $m+1$ check digits 3-valued separable balanced codes can be constructed. The proposed procedure modifies some digits of an information word and store the number of modified digits to its check word. Also, the check word compensate the codeword to be balanced. The proposed unordered codes are reasonably efficient and the encoding and decoding algorithms are quite simple.

2. On Multiple-Valued Unordered Codes

Initially, some background definitions are introduced, and classes of unordered codes; multiple-valued Berger codes, multiple-valued complemented duplication codes and multiple-valued constant weight codes are outlined. Later the optimality of multiple-valued Berger codes will be discussed.

2.1 Definitions

Suppose C_R is an R-valued code-sequence over $Z_R = \{0, 1, \cdots, R-1\}$. The following notations are used hereafter.

R	:	radix number
n	:	code length
k	:	number of information digits
$c = n - k$	:	number of check digits
$\lvert C_R \rvert$	:	number of codewords of C_R

Symbols of an R-valued code have totally ordered relations as $0 < 1 < \cdots < R-1$. When $X = (x_1 x_2 \cdots x_n)$ and $Y = (y_1 y_2 \cdots y_n)$ are two codewords $(x_i, y_i \in Z_R,\ i = 1, 2, \cdots, n)$, assume $N(X, Y)$ is the number of digit where $x_i > y_i$ $(i = 1, 2, \cdots, n)$ [1].

Manuscript received April 25, 1995.
Manuscript revised July 20, 1995.

[†]The author is with the Department of Electrical and Electronics Engineering, University of the Ryukyus, Okinawa-ken, 903-01 Japan.
[††]The author is with the Department of Computer Science, Meiji University, Kawasaki-shi, 214 Japan.

Example 1: On Z_4, if $X = (2131)$ and $Y = (2320)$, then $N(X,Y) = 2$ and $N(Y,X) = 1$.

When $N(Y,X) = 0$, then represent X covers Y as $X \geqq Y$. If neither $X \geqq Y$ nor $Y \geqq X$, then X and Y are said to be unordered.

If the information digits and the check digits are distinguishable, the codes are said to be *systematic codes*. Again, if the information blocks (words) are separately identified from the check blocks, they are called *separable codes*. The advantage of systematic codes is that error detection and data manipulation can be done in parallel. In many cases of practical use or circuit manufacture, separable codes are more desirable. Though Berger codes are separable, constant weight codes usually are neither systematic nor separable.

An unordered code is *efficient* if there is a very simple way to encode and decode k-digit numbers. In other words, we want to find a one-to-one correspondence between the set of all k-digit information words and the set of all $(k+c)$-digit codewords such that, if X corresponds to Y, we can rapidly compute Y from X and vise versa with possibly lower costs of operation. Furthermore, by extending the concepts of literature [5], we want c to be very small compared with k so that the code is *efficient* in its use of space, that is hardware costs, as well as time.

2.2 Multiple-Valued Unordered Codes

(i) Multiple-Valued Berger Codes

Suppose $X = (x_1 x_2 \cdots x_k)$ is an information word (block) where $x_i \in Z_R$. Multiple-valued Berger codes are constructed as a concatenation of X and S where S is a check word (block). The check word S is an R-valued expression of ν which is derived from the following equation.

$$\nu = \sum_{i=1}^{k}((R-1) - x_i) = (R-1)\cdot k - \sum_{i=1}^{k} x_i \qquad (1)$$

Thus, the length of check word S is

$$c = \lceil \log_R((R-1)\cdot k + 1) \rceil . \qquad (2)$$

where $\lceil \gamma \rceil$ means the smallest integer over the real number γ. Note that the value ν derived from Eq. (1) is said to be the *check value* and the R-valued expression of ν is referred to as *check word* or *check block* henceforth.

Example 2: For an information word $X_1 = (2131)$ over Z_4, the check value $\nu = (3-2)+(3-1)+(3-3)+(3-1) = 5$ in decimal. The decimal 5 is 11 in radix-4 representation, and thus the check word is 11 with $c = 2$.

(ii) Multiple-Valued Complemented Duplication Codes

In a multiple-valued complemented duplication code, the values of each check digit takes the complement of each information digit respectively. Namely, for an information digit x_i, the value of check digit is $((R-1) - x_i)$, thus the number of check digit $c = k$. A multiple-valued unidirectional error cannot increase the value of a information digit with decreasing the value of the related check digit simultaneously. Although the complemented duplication code is interesting in fault tolerant technology, the code system can be regarded as a row of two-digit-sliced Berger codes ($n = 2, k = 1$), MV-complemented duplication codes are not mentioned hereafter.

(iii) Multiple-Valued Constant Weight Codes

In an R-valued constant weight code, the number of each symbol is constant where the symbol $x_i \in \{0, 1, \cdots, R-1\}$. In the binary case, the codes are M-out-of-N codes.

The R-valued constant weight codes, in which each codeword contains equally many $0s, 1s, \cdots, (R-1)'s$, are called R-valued *balanced codes*. It is well known that binary balanced codes (half-hot codes) are particularly suited to high speed optical fiber links and similar channels [1].

2.3 Considerations of Multiple-Valued Berger Codes

In binary Berger codes, an encoding/decoding scheme of 2^m information bits and m check bits codes are presented [1]. (The Berger codes with 2^m information bits generally require $m+1$ check bits.)

Moreover, if an information word $(00\cdots0)$ whose check value is 2^m is not used, a number of check bits $c = m$ is sufficient. Namely, $k = 2^m$, $c = m$ Berger codes which have $2^{2^m} - 1$ codewords can be obtained.

On the other hand, in R-valued Berger codes, since the maximum check value is $(R-1)\cdot k$ from Eq. (1), the check value of information word which causes an extension of one digit in the check block is

$$(R-1)\cdot k = R^m.$$

Thus, k is obtained as follows:

$$k = \frac{R^m}{R-1}$$

In the above equation, if R is an odd number then the denominator is even and the numerator is odd. Similary, if R is an even number then the denominator is odd and the numerator is even. Therefore, it is when R=2 that k becomes an integer. Thus, the number of information digits that the check digit increases by one digit is

$$k = \left\lceil \frac{R^m}{R-1} \right\rceil . \qquad (3)$$

Example 3: In R = 3,

$$k = \left\lceil \frac{3^m}{2} \right\rceil = \left\lceil \frac{3}{2} \right\rceil, \left\lceil \frac{9}{2} \right\rceil, \left\lceil \frac{27}{2} \right\rceil, \cdots = 2, 5, 14, \cdots$$

and the numbers of check digits are $c = 2, 3, 4, \cdots$.

In R = 4,

$$k = \left\lceil \frac{4^m}{3} \right\rceil = \left\lceil \frac{4}{3} \right\rceil, \left\lceil \frac{16}{3} \right\rceil, \left\lceil \frac{64}{3} \right\rceil, \cdots = 2, 6, 22, \cdots$$

$$c = 2, 3, 4, \cdots$$

are formed.

Consequently, if one symbol is lengthened in a check block, the most significant digit of the check block is used by a number of codewords as non-zero. For example, $n_c/R^m = 3/3^2, 6/3^5, 6/4^2, 28/4^6, \cdots$, where R^m is the number of total codewords in radix R, and n_c is the number of codewords of which the most significant digit of the check block is non-zero. Of course, the above examples are on inefficient cases relating to Eq. (3). Note that in the binary Berger code of which $k = 2^m$, the most significant digit of check word is used by just one information word : namely by the codeword $(00\cdots0)$ (i.e., $n_c/2^m = 1/2^4, 1/2^8, 1/2^{16}, \cdots$). Thus, we expect that the check digit reduction technique in [1] is hard to apply directly to $R \geqq 3$ cases.

In the binary case, Berger codes with $k = 2^m - 1$ information digits are most efficient. This is called the *maximal length Berger codes.* In R-valued Berger codes, since $R^c = (R-1)\cdot k+1$ from Eq. (2), the number of information digits, of which check block is most efficient, is

$$k = \frac{R^c - 1}{R - 1} . \tag{4}$$

For instance, $k = \frac{3^c-1}{2} = 1, 4, 13\cdots$ in $R = 3$, and $k = \frac{4^c-1}{3} = 1, 5, 21, \cdots$ in $R = 4$.

When $m = c$ in Eq. (3), we have

$$\frac{R^c}{R-1} \leqq \left\lceil \frac{R^c}{R-1} \right\rceil < \frac{R^c}{R-1} + 1. \tag{5}$$

Since $k = \frac{R^c-1}{R-1}$ is an integer number (Eq.(4)),

$$k < \left\lceil \frac{R^c}{R-1} \right\rceil < k + 1 + \frac{1}{R-1}. \tag{6}$$

Since $\frac{1}{R-1} \leqq 1$ where R is an integer number that is greater than or equal to 2,

$$\left\lceil \frac{R^c}{R-1} \right\rceil = k + 1. \tag{7}$$

Let $e_c = k_e - k_i$ where k_e is the most efficient k (the number of information digits) and k_i is the most inefficient k in fixed c (the number of check digits). From Eq. (4) and Eq. (7),

$$\begin{aligned} e_c &= \frac{R^c - 1}{R-1} - \frac{R^{c-1} - 1}{R-1} - 1 \\ &= R^{c-1} - 1 \end{aligned} \tag{8}$$

Besides, a larger e_c means a wider extent between k_e and k_i. Then, we obtain

$$\begin{aligned} \frac{\Delta e_c}{\Delta c} &= (R^c - 1) - (R^{c-1} - 1) \\ &= (R-1)\cdot R^{c-1} \end{aligned} \tag{9}$$

Naturally the e_c increases according to c in Eq. (9), e_c also has a close relation to R.

From the above discussion, in the higher radix R, multiple-valued Berger codes are expected to become relatively efficient in the practical range of k.

As a result, in a multiple-valued case, usage of the modified Berger codes [14] is a better way to reduce the length of a check block, a technique which uses not all 2^k information words. Incidentally, the check values of the modified Berger code are computed from information words as follows.

$$\nu' = \sum_{i=1}^{n} ((R-1) - x_i) - \delta \tag{10}$$

where δ is the smallest value of ν in Eq. (1) for the exploiting information words.

3. Multiple-Valued Separable Balanced Codes

In this section, the new scheme of separable balanced codes over Z_R are proposed. This coding/encoding scheme, in a sense, is an extention of the Bose's result in [2].

3.1 Definitions and Theorems

[**Definition 1**] Let $X = (x_1 x_2 \cdots x_k)$ be an information word of which the length is k. In the information word X, let $W_R(X) = (w_0, w_1, \cdots, w_{R-1})$ be an ordered set of weights where w_i is the weight of symbol i, that is, the number of i digits.

The notation $max(W_R(X))$, that is $max(w_0, w_1, \cdots, w_{R-1})$ indicates the maximum weight of an element in x.

[**Definition 2**] For a variable integer x, $\vec{x}$ represents a successor to x : $(x+1)$ mod R, and $\overleftarrow{x}$ represents a predecessor of x : $(x-1)$ mod R.

For instance, for input string $< 0, 1, 2, 3 >$ of 4-valued variable x, the outputs of the unary operations $\vec{x}$ and $\overleftarrow{x}$ are $< 1, 2, 3, 0 >$ and $< 3, 0, 1, 2 >$, respectively.

[**Definition 3**] Let $\sigma_j(X)$ be an ordered set of weights where the first j digits of X are applied as successors.

Example 4: For a code $X = (032012)$ over Z_4, $W_4(X) = (2, 1, 2, 1)$ and $\sigma_3(X) = W_4((103012)) = (2, 2, 1, 1)$.

[**Definition 4**] For a certain information block $X_I = (x_1 x_2 \cdots x_g)$, let $cyc_{+d}(X_I) = (x_{g-d+1} \cdots x_{g-d-1} x_{g-d})$ and $cyc_{-d}(X_I) = (x_{d+1} x_{d+2} \cdots x_d)$ represent *right cyclic shift* and *left cyclic shift*, respectively, where $0 < d < b_g$.

The following theorems and a lemma hold in multiple-valued balanced codes.

[**Theorem 1**] In an information word over Z_R, if there exists a symbol i where $w_i > \lfloor \frac{k}{2} \rfloor$, there is no other

symbol j where $w_j > \lfloor \frac{k}{2} \rfloor$.

proof: If there exist two symbols i, j where $w_i, w_j > \lfloor \frac{k}{2} \rfloor$, i.e. $w_i, w_j \geqq \lfloor \frac{k}{2} \rfloor + 1$, the length of information word l has a relation such that $l \geqq 2 \cdot (\lfloor \frac{k}{2} \rfloor + 1) > 2 \cdot \frac{k}{2}$, thus $l > k$. This is a contradiction. Q.E.D.

For a certain X, if each value of w_i is e_i, then

$$\sigma_0(X) = (w_0, w_1, \cdots, w_{R-1}) = (e_0, e_1, \cdots, e_{R-1}) \tag{11}$$

$$\sigma_k(X) = (e_{R-1}, e_0, e_1, \cdots, e_{R-2}) \tag{12}$$

When

$$\sigma_{j-1}(X) = (w_0, \cdots, w_i, \cdots, w_{R-1}) \tag{13}$$

and the j-th digit of X is $x_j = i$,

$$\sigma_j(X) = (w_0, \cdots, w_i - 1, w_{(i+1) \bmod R} + 1, \cdots, w_{R-1}) \tag{14}$$

is formed. Thus the $w_{(i+1) \bmod R}$ is a random walk from $e_{(i+1) \bmod R}$ to e_i. From this and Theorem 1, therefore, a word of which $max(W_R(X)) \leqq \lfloor \frac{k}{2} \rfloor$ can be obtained from a word of which $max(W_R(X)) > \lfloor \frac{k}{2} \rfloor$.

Example 5: An information word $X = (1222)$ over Z_4 can be changed to (2332) by taking successors of the first 3 digits as follows:

$$X = (1222) W_4(X) = (0, 1, 3, 0)$$
$$max(W_4(X)) = 3 > \left\lfloor \frac{k}{2} \right\rfloor = 2$$
$$(\vec{1}222) = (2222)\sigma_1(X) = (0, 0, 4, 0)$$
$$max(W_4(X)) = 4 > \left\lfloor \frac{k}{2} \right\rfloor$$
$$(\vec{1}\vec{2}22) = (2322)\sigma_2(X) = (0, 0, 3, 1)$$
$$max(W_4(X)) = 3 > \left\lfloor \frac{k}{2} \right\rfloor$$
$$(\vec{1}\vec{2}\vec{2}2) = (2332)\sigma_3(X) = (0, 0, 2, 2)$$
$$max(W_4(X)) = 2 \leqq \left\lfloor \frac{k}{2} \right\rfloor$$

[**Lemma 1**] The number of codeword of the balanced codes over Z_R is given by

$$|C_R| = \frac{(R \cdot m)!}{(m!)^R} \tag{15}$$

where code length $n = R \cdot m$ $(m = 1, 2, \cdots)$.

proof: The number of codewords of constant weight codes over Z_R is

$$|C_R| = \frac{n!}{w_0! w_1! \cdots w_{R-1}!} \tag{16}$$

where $(w_0, w_1, \cdots, w_{R-1})$ is an ordered set of weights of code symbols. If $w_0 = w_1 = \cdots = w_{R-1} = m$, then

$$|C_R| = \frac{n!}{(m!)^R} = \frac{(R \cdot m)!}{(m!)^R}. \qquad \text{Q.E.D.}$$

Note that since $\sum_{i=0}^{R-1} w_i = n$ and the denominator of Eq. (16) is minimal when $w_0 = w_1 = \cdots = w_{R-1} = m$, the number of codewords of a balanced code is maximal in constant w eight codes.

[**Theorem 2**] In Eq. (15), for the set of codewords that $max(w_i) > \frac{k}{2}$, the function f is defined as follows:

Let f be a conversion from the words C_g where $max(w_i) > \frac{k}{2}$ to the words C'_ℓ where $max(w_i) \leqq \frac{k}{2}$. Namely,

$$f : C_g \longrightarrow C'_\ell$$

where

$$f(X) = f((x_1 x_2 \cdots x_k)) = (\vec{x_1}\vec{x_2} \cdots \vec{x_j} x_{j+1} \cdots x_k) = X' \tag{17}$$
$$max(W_R(X)) > \frac{k}{2}, \quad max(W_R(X')) \leqq \frac{k}{2}$$

and for $s < j$,

$$max(W_R((\vec{x_1}\vec{x_2} \cdots \vec{x_s} x_{s+1} \cdots x_k))) \not\leqq \frac{k}{2}.$$

Then the function f has a one-to-one correspondence for words C_g where $\sum_{i=0}^{R-1} W_R(C_g) = g > \frac{k}{2}$.

proof: It is sufficient for the proof that different words, but with the same $\sum_{i=0}^{R-1} w_i$, never project onto same word with appropriate successors.

Suppose two words X and $Y \in C_g$ where $X \neq Y$, but $f(X) = f(Y) = Z$. Let Z be the result of applying successors to X and Y by s and t digits, respectively. Namely $f(X) = f(Y) = Z$ and those two numbers s and t are minimal numbers as required by the function f. When $s \leqq t$,

$$X = (x_1 \cdots x_s \cdots x_t \cdots x_k)$$
$$Y = (y_1 \cdots y_s \cdots y_t \cdots y_k)$$
$$Z = (z_1 \cdots z_s : \cdots z_t \cdots z_k)$$

Thus

$$Z = (\vec{x_1} \cdots \vec{x_s}\ x_{s+1} \cdots x_t \cdots x_k) \tag{18}$$
$$= (\vec{y_1} \cdots \vec{y_s} \cdots \vec{y_t}\ y_{t+1} \cdots y_k) \tag{19}$$

In X let $B_1 = (x_1 x_2 \cdots x_s)$, $B_2 = (x_{s+1} \cdots x_t)$ and $B_3 = (x_{t+1} \cdots x_k)$. Similarly, in Y, $B'_1 = (y_1 y_2 \cdots y_s)$, $B'_2 = (y_{s+1} \cdots y_t)$ and $B'_3 = (y_{t+1} \cdots y_k)$. Then

$$X = B_1 @ B_2 @ B_3$$
$$Y = B'_1 @ B'_2 @ B'_3$$

where @ indicates concatination.

Then Eqs. (18) and (19) can be rewriten as

$$f(B_1 @ B_2 @ B_3) = f(B'_1 @ B'_2 @ B'_3) = Z$$
$$\text{thus, } (\vec{B_1} @ B_2 @ B_3) = (\vec{B'_1} @ \vec{B'_2} @ B'_3) = Z \tag{20}$$

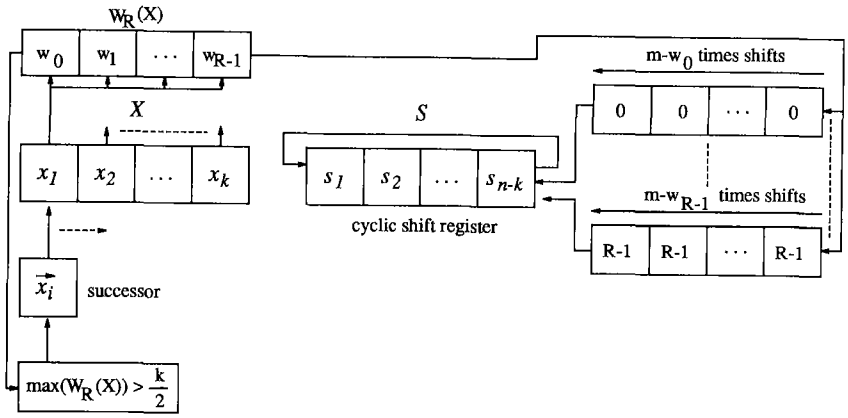


Fig. 1 Encoding to MV-separable balanced codes.

Note that $\vec{B_1} = (\vec{x_1}\vec{x_2}\cdots\vec{x_s})$, $\vec{B_1'} = (\vec{y_1}\vec{y_2}\cdots\vec{y_s})$, $\vec{B_2} = (\vec{x_{s+1}}\cdots\vec{x_t})$, and so forth.

From Eq. (20),

$$\vec{B_1} = \vec{B_1'} \text{ thus, } B_1 = B_1' \tag{21a}$$

$$B_3 = B_3' \tag{21b}$$

$$B_2 = \vec{B_2'} \tag{21c}$$

In Eq. (21c), since $W_R(X) = W_R(Y)$ and from Eqs. (21a) and (21b),

$$W_R(B_2) = W_R(B_2').$$

Therefore, these two blocks B_2 and B_2' are balanced: i.e. each entry $0, 1, \cdots, R-1$ are of equal cardinality in B_2 and B_2' (refer to Eqs. (11) and (12)). Meanwhile, s and t are the minimal numbers of digits that $max(W_R(X)) > \frac{k}{2}$ and $max(W_R(Y)) > \frac{k}{2}$ into $max(W_R(X)) \leqq \frac{k}{2}$ and $max(W_R(Y)) \leqq \frac{k}{2}$, respectively. Therefore, $W_R(B_2) \neq W_R(B_2')$ is required. Then $s = t$, thus $X = Y$ is concluded. This contradicts $X \neq Y$. Consequently, $f(X) \neq f(Y)$. Q.E.D.

3.2 Construction of Multiple-Valued Separable Balanced Codes

The encoding/decoding algorithms for multiple-valued separable balanced codes where $k = 2 \cdot m$ $(m \geqq 1)$ information digits and $c = m \cdot (R-2)$ are presented below. At present, no efficient algorithm of multiple-valued separable balanced codes can be found.

Encoding:

For R^k information words X where $k = 2m$,

Case 1: if $max(W_R(X)) \leqq \frac{k}{2}$, then concatenate X and check block S of which length is $m \cdot (R-2)$ and in which the number of each check symbol i is $m - w_i$ $(i = 0, 1, \cdots, R-1)$. Sort the symbols of the check block in an increasing order.

Case 2: if $max(W_R(X)) > \frac{k}{2}$, apply f to X for $max(W_R(X)) \leqq \frac{k}{2}$. Subsequently to apply f, construct check block S in a similar way as Case 1, and apply right cyclic shifts to S by j digits as $cyc_{+j}(S)$.

In Case 1, $max(W_R(X)) \leqq \frac{k}{2} = m$ as $k = 2m$. Then, for check block S, the number of each check symbol is $m - w_i$. Thus,

$$\begin{aligned} W_R(X@S) &= (w_0 + m - w_0, \cdots, \\ &\quad w_{R-1} + m - w_{R-1}) \\ &= (m, \cdots, m). \end{aligned}$$

Therefore, $X@S$ is a balanced code. Similarly, in Case 2, balanced codes can be obtained with f. This encoding scheme is illustrated in Fig. 1. In the Fig. 1, the encoding is carried out by weight counter (upper left), symbol "0", "1", ⋯ "R-1" generator with length m (right), $max(W_R(X))$ comparator with $\frac{k}{2}$ (bottom left), and successor (second from the bottom, left). In the sequel, encoded codeword can be extracted from rectangles X and S.

Example 6: For an information word $X = (3222)$, $max(W_R(X)) = 3 > \frac{k}{2} = 2$ $(m = 2)$. Then,

$$\begin{aligned} &(\vec{x_1}x_2x_3x_4) = (0222) \\ &(\vec{x_1}\vec{x_2}x_3x_4) = (0322) \\ &\text{thus, } W_R(f(X)) = (1, 0, 2, 1). \end{aligned} \tag{22}$$

The weights of check block S is provided as

$$\begin{aligned} W_R(S) &= (m-1, m-0, m-2, m-1) \\ &= (1, 2, 0, 1), \end{aligned}$$

then $S = (0113)$ and $cyc_{+2}(S) = (1301)$. Therefore, $X@cyc_{+2}(S) = (0322\ 1301)$ is obtained.

Decoding:

In a codeword, the check block $(m \cdot (R-2)$ digits) has an increasing order, the first k digits of the codeword are the information word directly.

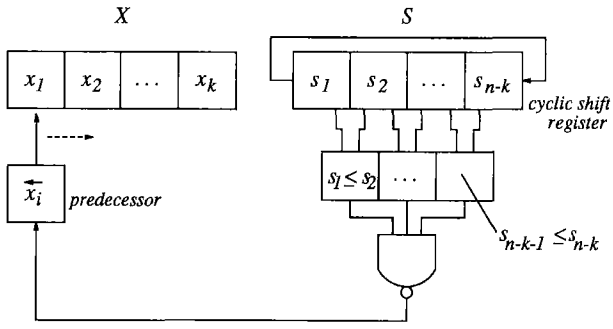


Fig. 2 Decoding of MV-separable balanced codes.

While, if the check block is not of an increasing order, apply the left cyclic shift until the check block becomes increasing ordered. Simultaneously, apply predecessors from the first digit of the information block by the same time (refer to Fig. 2). In the Fig. 2, the blocks squared off the $s_i \leqq s_{i+1}$ before a binary NAND gate operates as $\begin{cases} 1: & s_i \leqq s_{i+1} \\ 0: & s_i > s_{i+1} \end{cases}$ where 0/1 are binary output values. Predecessor (bottom left) decrement and cyclic shift in S are carried out simultaneously until all neighbors in S have $s_i \leqq s_{i+1}$ $(i = 1 \sim n-k)$ relations.

[**Theorem 3**] In the above encoding and decoding scheme, for j which indicates the time of (left/right) shift, successor, or predecessor, $j \leqq k-1$ is formed.

proof: When $max(w_i) > \frac{k}{2}$ for an information word, k times successor results $w_j > \frac{k}{2}$ where $j = (i+1) \bmod R$. Q.E.D.

[**Theorem 4**] In a check word $S = (s_1 s_2 \cdots s_c)$ where the code length is c, if $s_1 \leqq s_2 \leqq \cdots \leqq s_c$ and $s_1 < s_c$ then the check word S consists of 2 or more different kinds of elements and c different codes can be obtained by $0 \sim c-1$ times cyclic shifts.

proof: The $0 \sim c-1$ times cyclic shifts to S is illustrated below.

$$\begin{aligned} &s_1 \leqq s_2 \leqq \cdots \leqq s_c && : 0 \text{ time} \\ &s_2 \leqq s_3 \leqq \cdots \leqq s_c \leqq s_1 && : 1 \text{ time} \\ &\quad\vdots \\ &s_{i+1} \leqq s_{i+2} \leqq \cdots \leqq s_c < s_1 \leqq \cdots \leqq s_i && : i \text{ times} \\ &\quad\vdots \\ &s_c < s_1 \leqq s_2 \leqq \cdots \leqq s_{c-1} && : c-1 \text{ times} \end{aligned}$$

Note that s_1 and s_c are the minimum and the maximum values among $s_1 \sim s_c$, respectively. Actually, if information block X (or successored X') satisfies $X(\text{or } X') \leqq \frac{k}{2}$ and the check digits for which $X@S$ becomes balanced are sorted in an increasing order, $s_1 \leqq s_2 \leqq \cdots \leqq s_c$ and $s_1 < s_c$ are always formed. Thus the row $s_c s_1$ is the only presence in the code during the above cyclic shifts. On the other hand, evidently, the row $s_c s_1$ takes different positions in the codes with the $0 \sim c-1$ times shiftings. Therefore, these codes obtained $0 \sim c-1$ times shifts are all different from each other.

[**Theorem 5**] The mappings with the above Case 1 and Case 2 encoding schemes have a one to one correspondence.

proof: Let X and Y be two information words with $\sum_{i=0}^{R-1} w_i^{(X)} \neq \sum_{i=0}^{R-1} w_i^{(Y)}$, respectively where $w_i^{(X)} \in W_R(X), w_i^{(Y)} \in W_R(Y)$. Then, three cases

(a) $max(W_R(X)), max(W_R(Y)) \leqq \frac{k}{2} : s = t = 0$ and evidently $X \neq Y$

(b) $max(W_R(X)) \leqq \frac{k}{2}$ and $max(W_R(Y)) > \frac{k}{2} : s = 0, t > 0$

(c) $max(W_R(X)), max(W_R(Y)) > \frac{k}{2} : s \neq t$ $(s, t > 0)$

can be considered, where s and t are the numbers of successored digits.

(i) In case (a), clearly, $X@S_X \neq Y@S_Y$, where S_X and S_Y are the check blocks of X and Y, respectively.

(ii) In cases (b) and (c), since $S_X \neq S_Y$ $(s \neq t)$ from Theorem 5, $X@S_X \neq Y@S_Y$.

From (i), (ii) and Theorem 3, all k-length information words are mapped to different codewords with the encoding methods of Case 1 and Case 2. Q.E.D.

[**Theorem 6**] In the proposed encoding/decoding scheme of multiple-valued separable balanced codes, $k = 2 \cdot m$ information word and $c = m \cdot (R-2)$ check word for $R \geqq 4$, and $k = 2m-1$ and $c = m+1$ for $R = 3$ are sufficient.

proof: When $k = 2 \cdot m$, since the maximum weight of an information word X can be changed to $max(W_R(X')) \leqq \frac{k}{2}$ with successors (refer to Theorem 4), $c \geqq k$ $(= 2m)$ is formed for check word. On the other hand, when the maximum weight of information word is $\frac{k}{2}$ in an R-valued code, $c = m \cdot (R-2)$ must be stated for $X@S$ as balanced. For the above reason, $c = m \cdot (R-2) \geqq k = 2m$ thus, $R \geqq 4$.

From Theorem 5, two or more kinds of elements are needed containing in the check word for an information word where $max(W_R(X)) \geqq \frac{k}{2}$. When $R = 3$, to satisfy this condition, the information word should be decreased by one digit and increased by one digit in check block, sufficiently. Then,

$$k = 2 \cdot m - 1$$

$$c = m \cdot (R-2) + 1 = m + 1$$

for the ternary separable balanced codes. Q.E.D.

Example 7:

Encoding: When $R = 4$ and $k = 2$, an example of encoding for separable balanced codes is shown in Table 1 where for example, the information word (00) is changed to (10) by taking a successor at the underlined digit. Then check word (32) which is obtained by a right cyclic shift for (23) is concatenated to the information word.

Decoding: Apply left cyclic shift to check word until the check word becomes increasingly ordered. Simul-

taneously, apply predecessors to the information word. In the Table 1, for instance, (10 32) (00 23) is obtained.

Example 8: For a ternary information word (22222) where $m = 3$ is changed to (00222) by function f, and as $j = 2$, check word (1101) is obtained by 2 times right cyclic shifts of (0111). Then a ternary separable balanced code is obtained.

3.3 The Comparison of Multiple-Valued Unordered Codes

The number of check digits in multiple-valued Berger codes is $c = \lceil \log_R((R-1) \cdot k + 1) \rceil$. On the other hand, in multiple-valued separable balanced codes, the number of check digits is generally $k \cdot (R-1)$ in the worst case [4]. In the proposed separable balanced codes, $c = \frac{k}{2}(R-2)$ in $R \geqq 4$ and $c = \frac{k}{2} + 1$ in $R = 3$.

The number of check digits and the number of codewords where $R = 3$ and 4 in Berger codes and separable balanced codes are shown in Table 2, respectively.

In the Table 2, for the practical range of $k = 1 \sim 16$, and $R = 3$ or 4, we considered that the efficiency of the proposed separable balanced codes are fairly close to the Berger codes known as optimal unordered codes.

Let evaluate computational complexities of the multiple-valued Berger codes and the proposed separable balanced codes. For k information digits in multiple-valued Berger codes, encoding procedure needs cost k for weight detection, cost k for differences between $R-1$ and each weight of digits, and cost $k-1$ for sum of the differences (assume R-valued summations). Thus the total computational cost is $3k-1$. On the other hand, in the case of proposed separable balanced codes, encoding procedure needs cost k for weight detection, cost from 0 to $k-1$ for majority weight detection and increment operations, cost R for compensations with the check block in order that the codeword be balanced. (computation of $m - w_i$'s where $i = 0 \sim R-1$), and cost 1 for cyclic shift operation. Therefore, from $k + R + 1$ to $2k + R$ computational cost is required. It seems reasonable to suppose that if the length of the information block is greater than the radix R, the computational cost of proposed separable balanced code is smaller than that of multiple-valued Berger code. Moreover, since the Berger code is systematic code, an information word can be earmed from codewords directly. On the contrary, proposed separable balanced code requires from 0 to $k-1$ cost to obtain an information word in decoding procedure. Besides this, while Berger code frequently exploits arithmetic operations, proposed separable balanced code encoding/decoding operations might be implemented by gate-level components (max, shift, e.t.c.).

For unidirectional error detection, unordered codes are applied as $4 \sim 16$ digit slice constructions in general. Therefore, we expect that the comparison in Table 2 shows the validity of encoding/decoding scheme of the proposed separable balanced codes in actual applications.

Table 1 Encoding of a 4-valued separable balanced code where $k = 2$.

Information words		Code words	
0 0	→	1 0	3 2
0 1	→	0 1	2 3
0 2	→	0 2	1 3
0 3	→	0 3	1 2
1 0	→	1 0	2 3
1 1	→	2 1	3 0
1 2	→	1 2	0 3
1 3	→	1 3	0 2
2 0	→	2 0	1 3
2 1	→	2 1	0 3
2 2	→	3 2	1 0
2 3	→	2 3	0 1
3 0	→	3 0	1 2
3 1	→	3 1	0 2
3 2	→	3 2	0 1
3 3	→	0 3	2 1

4. Conclusions

Multiple-valued unordered codes are considered. Particularly, the efficiency of Berger codes are cleared to be more optimal in multiple-valued cases in whole ranges of k.

Moreover, multiple-valued separable balanced codes are proposed. These codes have $2 \cdot m$ informa-

Table 2 A comparison between MV-Berger codes and MV-separable balanced codes on the number of check digit: c and the code length: n over Z_3 and Z_4.

$\lvert c \rvert / \lvert n \rvert$	MV-Berger codes	MV-balanced block codes (proposed)	MV-balanced block codes (worst case)
$R = 3$ $k = 1$	1 / 2	2 / 3	2 / 3
3	2 / 5	3 / 6	6 / 9
5	3 / 8	4 / 9	10 /15
7	3 /10	5 /12	14 /21
15	4 /19	9 /24	30 /45
$R = 4$ $k = 2$	2 / 4	2 / 4	6 / 8
4	2 / 6	4 / 8	12 /16
6	3 / 9	6 /12	18 /24
8	3 /11	8 /16	24 /32
16	3 /19	16 /32	48 /64

tion digits and $m \cdot (R-2)$ check digits in radices $R \geqq 4$, and $2 \cdot m - 1$ information digits and $m + 1$ check digits in $R = 3$ where code-length $n = R \cdot m$. It is shown that the efficiency of multiple-valued separable balanced codes are quite close to the multiple-valued Berger codes in actual use.

References

[1] D. Wessels and J.C. Muzio, "Concurrent checking and unidirectional errors in multiple-valued circuits," Proc. 22nd Int. Symposium on Multiple-Valued Logic, pp.166–173, May 1992.

[2] B. Bose, "On unordered codes," IEEE Trans. Comput., vol.40, pp.125–131, Feb. 1991.

[3] B. Bose and D.K. Pradhan, "Optimal unidirectional error detecting/correcting codes," IEEE Trans. Comput., vol.c-31, pp.564–568, June 1982.

[4] G.P. Mak, J.A. Abraham, and E.S. Davidson, "The design of PLAs with concurrent error detection," Proc. 12th Int. Symposium on Fault-Tolerant Computing, pp.303–310, 1982.

[5] D.E. Knuth, "Efficient balanced codes," IEEE Trans. Inform. Theory, vol.IT-32, no.1, pp.51–53, Jan. 1986.

[6] J.E. Smith, "On separable unordered codes," IEEE Trans. Comput., vol.c-33, pp.741–743, Aug. 1984.

[7] D.J. Rosenkrantz, "Half-hot state assignment for finite state machines," IEEE Trans. Comput., vol.39, no.5, pp.700–702, March 1990.

[8] Y. Tohma, Y. Ohyama, and R. Sakai, "Realization of fail-safe sequential machines by using k-out-of-n code," IEEE Trans. Comput., vol.c-20, no.11, pp.1270–1275, Nov. 1971.

[9] Y. Nagata and C. Afuso, "A state assignment for p-valued sequential machines," SYSTEM and COMPUTERS in JAPAN, vol.21, no.3, pp.93–100, 1990.

[10] Y. Nagata, T. Zukeran, and C. Afuso, "On an algorithm for input encoding with half-weight codes," IEICE technical report on Fault Tolerant Systems, vol.91, no.122, FTS91-19, 1991 (in Japanese).

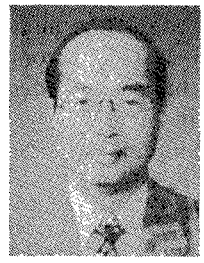
[11] Y. Nagata and M. Mukaidono, "A fault model for multiple-valued PLA's and its equivalences," IEICE Trans. Fundamentals, vol.E77-A, no.9, pp.1527–1534, Sept. 1994.

[12] Y. Nagata and C. Afuso, "A method of test pattern generation for multiple-valued PLA's," Proc. 23rd International Symposium on Multiple-Valued Logic, pp.87–91, May 1993.

[13] M. Mukaidono ed., "Fault Tolerant Computing," Maruzen, Sept. 1989.

[14] P.K. Lala, "Fault Tolerant & Fault Testable Hardware Design," Ohm ltd, Sept. 1988.

Yasunori Nagata was born in Japan in 1960. He received his B.S. and M.S. degrees in electronics and information engineering from University of the Ryukyus, Okinawa, in 1984 and 1987, respectively. He is currently an Assistant Professor in the Department of Electrical and Electronics Engineering, University of the Ryukyus. His main reserch interests are in multiple-valued logic, fault-tolerant systems and computer-aided logic design. He is a member of IEEE Computer, Circuits and Systems, and Information Theory Societies.



Masao Mukaidono was born in Japan in 1942. He received his B.S., M.S., and Ph.D. degrees in electrical engineering from Meiji University, Kawasaki, in 1965, 1968, and 1973, respectively. He is currently a Professor in the Department of Computer Science, Faculty of Science and Technology, Meiji University. His main research interests are in multiple-valued logic, fuzzy set theory and its applications, fault- tolerant computing systems, fail-safe systems, and computer-aided logic design. Dr. Mukaidono is a member of the IEEE Computer Society and Information Processing Society of Japan, and president of Japan Society for Fuzzy Theory and Systems.