On a Class of One-Step Majority-Logic Decodable Cyclic Codes
On a Class of One-Step Majority-Logic Decodable Cyclic Codes

Majority-logic decoding is attractive for three reasons: (1) It can be simply implemented; (2) the decoding delay is short; and (3) its performance, while suboptimal, is always superior to bounded distance decoding. For these reasons, majority-logic decodable cyclic codes are very suitable for error control in high-speed digital data transmission systems. Among the majority-logic decodable codes, we take the one-step majority-logic codes as our fist choice. In this paper we study a class of one-step majority-logic decodable cyclic codes. First, we describe these codes in a simple manner. Second, a way of finding the orthogonal polynomials for decoding these codes is presented. Third, we show that for a given error correction capability, the ratio of the number of parity digits to the code length goes to zero as the code length increases. For error correction capabilities of the form $2^m - 1$ or $2^m + 1$, we determine the dimensions of the codes exactly.

1. Introduction

Majority-logic decoding is attractive for three reasons: (1) It can be simply implemented; (2) the decoding delay is short; and (3) its performance, while suboptimal, is superior to bounded distance decoding [1]. For these reasons, majority-logic decodable cyclic codes are very suitable for error control in high-speed digital data transmission systems. Among the majority-logic decodable codes, the one-step decodable codes can be most easily implemented, since they employ a simple majority-logic gate [2]. In this paper, we study a class of one-step majority-logic decodable cyclic codes. This class of codes is a subclass of the generalized Euclidean geometry codes which are not in general one-step majority-logic decodable studied by Delsarte [4], Kasami and Lin [5], and Lin and Yiu [6]. First, we briefly describe the codes in a simple manner. Second, a method of finding the orthogonal polynomials (or orthogonal parity-sums) for decoding these codes is presented. Third, we show that for a given error correction capability, the ratio of the number of parity digits to the code length goes to zero as the code length increases. For error correction capabilities of the form $2^m - 1$ or $2^m + 1$, we determine the dimensions of the codes exactly. At the end, we present an example to illustrate the process of finding the orthogonal polynomials.

Since the decoding of these codes is based on the property that the coefficients, when extended by the addition of an overall parity-check digit, are invariant under the affine group of permutations, we give a brief discussion of this invariant property here.

Let $C$ be a binary cyclic code of length $n = 2^m - 1$, generated by the polynomial $g(x)$. Let $C_1$ be a code obtained from $C$ by appending an overall parity-check digit to every code vector in $C$, i.e., if $(u_0, u_1, u_2, \ldots, u_{n-1})$ is a vector in $C$, then

$$(u_0, u_1, u_2, \ldots, u_{n-1}, u_n)$$

is a vector in $C_1$, where $u_n$ is the overall parity-check digit and $u_n = u_0 \oplus u_1 \oplus u_2 \oplus \cdots \oplus u_{n-1}$, where $\oplus$ denotes the modulo-2 addition. Clearly, the length of $C_1$ is $2n$.

Let $GF(2^m)$ be the Galois field of $2^m$ elements. Let $a$ be a primitive element in $GF(2^m)$. Then the nonzero elements in $GF(2^m)$ can be expressed as powers of $a$, $a^0 = 1$, $a^1$, $a^2$, \ldots, $a^{2^m-2}$. The zero element $0$ in $GF(2^m)$ is sometimes represented by $a^m$. Now, we number the components of a vector $(u_0, u_1, u_2, \ldots, u_{n-1})$ in $C_1$ by the elements of $GF(2^m)$ as follows: The component $u_{i}$ is numbered $a^i$, the component $u_{i}$ is numbered $a^{i}$, and, for $0 \leq i < n$, the component $u_{i}$ is numbered $a^{i}$. These numbers are called location numbers. Let $T$ denote the location of a component in $(u_0, u_1, u_2, \ldots, u_{n-1})$. An affine permutation with parameters $\alpha$ and $\beta$ in $GF(2^m)$ and $\sigma \neq 0$ is a permutation that carries the component at location $T$ to the location $\alpha T + \beta$. The code $C_1$ is said to be invariant under the affine group of permutations if every affine permutation carries every code vector in $C_1$ into another code vector in $C_1$.

Let $h$ be a nonnegative integer less than $2^m$. The radius-2 expansion of $h$ is

$$h = h_0 + h_1 \cdot 2^1 + h_2 \cdot 2^2 + \cdots + h_{m-1} \cdot 2^{m-1},$$

where $h_j$ is either 0 or 1 for $0 \leq j < m$. Let $h$ be another nonnegative integer $< 2^m$ whose radius-2 expansion is

$$h = h_0 + h_1 \cdot 2^1 + h_2 \cdot 2^2 + \cdots + h_{m-1} \cdot 2^{m-1}.$$

The integer $h$ is said to be a descendant of $h$ if $h_0 \leq h_1$ for $0 \leq j < m$. We also write $h \preceq h$ meaning that $h$ is a descendant of $h$. Clearly, for all $h$, $0 \preceq h$. Let $\Delta(h)$ denote the set of all nonzero descendants of $h$. The following theorem characterizes the necessary and sufficient condition for the extension $C_1$ of a cyclic code $C$ to be invariant under the affine group of permutations.

- Theorem 1 (Kasami, Lin, and Peterson [7])
- Let $C$ be a cyclic code of length $2^m - 1$ generated by $g(x)$. Let $C_1$ be the extended code obtained from $C$ by appending an overall parity-check digit to every code vector in $C$, i.e., if $(u_0, u_1, u_2, \ldots, u_{n-1})$ is a vector in $C$, then

$$(u_0, u_1, u_2, \ldots, u_{n-1}, u_n)$$

is a vector in $C_1$, where $u_n$ is the overall parity-check digit and $u_n = u_0 \oplus u_1 \oplus u_2 \oplus \cdots \oplus u_{n-1}$, where $\oplus$ denotes the modulo-2 addition. Clearly, the length of $C_1$ is $2n$.

Let $GF(2^m)$ be the Galois field of $2^m$ elements. Let $a$ be a primitive element in $GF(2^m)$. Then the nonzero elements in $GF(2^m)$ can be expressed as powers of $a$, $a^0 = 1$, $a^1$, $a^2$, \ldots, $a^{2^m-2}$. The zero element $0$ in $GF(2^m)$ is sometimes represented by $a^m$. Now, we number the components of a vector $(u_0, u_1, u_2, \ldots, u_{n-1})$ in $C_1$ by the elements of $GF(2^m)$ as follows: The component $u_{i}$ is numbered $a^i$, the component $u_{i}$ is numbered $a^{i}$, and, for $0 \leq i < n$, the component $u_{i}$ is numbered $a^{i}$. These numbers are called location numbers. Let $T$ denote the location of a component in $(u_0, u_1, u_2, \ldots, u_{n-1})$. An affine permutation with parameters $\alpha$ and $\beta$ in $GF(2^m)$ and $\sigma \neq 0$ is a permutation that carries the component at location $T$ to the location $\alpha T + \beta$. The code $C_1$ is said to be invariant under the affine group of permutations if every affine permutation carries every code vector in $C_1$ into another code vector in $C_1$.

Copyright 1980 by International Business Machines Corporation. Copying is permitted without payment of royalty provided that (1) each reproduction is done without alteration and (2) the Journal reference and IBM copyright notice are included on the first page. The title and abstract may be used without further permission in computer-based and other information-service systems. Permission to republish other excerpts should be obtained from the Editor.
these vectors is $L$. Adding an overall parity-check digit to each of these vectors, we obtain $J$ vectors $u_0, u_1, \ldots, u_{J-1}$ of length $2L$. The vectors $u_0, u_1, \ldots, u_{J-1}$ are code vectors in $C$, the extension of $C$. Since $L$ is odd, the overall parity-check digit of each $u_j$ is 1. Thus, $u_0, u_1, \ldots, u_{J-1}$ have the following properties:
1. They all have 1 at location $\alpha$ (overall parity-check digit location);
2. One and only one vector has a 1 at location $\alpha$ for $j = 0, 1, 2, \ldots, 2L - 1$.

These vectors are said to be orthogonal on the digit at location $\alpha$ [1].

Now, we apply the affine permutation
\[ Z = a + x \cdot d \]
to $u_0, u_1, \ldots, u_{J-1}$. This permutation carries the set of $J$ vectors $u_0, u_1, \ldots, u_{J-1}$ into another set of $J$ vectors $u'_{0}, u'_{1}, \ldots, u'_{J-1}$ in $C$. Note that the permutation carries the component $u_j$ at location $\alpha$ to location $\alpha$' $\alpha'$. Thus, these vectors $u_0, u_1, \ldots, u_{J-1}$ have the following properties:
1. All the vectors have a 1 at location $\alpha$; and
2. One and only one vector has a 1 at location $\alpha$ for $j = 0, 1, 2, \ldots, 2L - 1$.

Hence, $u_0, u_1, \ldots, u_{J-1}$ are orthogonal on the digit at location $\alpha'$ $\alpha'$. Deleting the digit at location $\alpha'$ $\alpha'$ from $u_0, u_1, \ldots, u_{J-1}$, we obtain $J$ vectors $x_0, x_1, \ldots, x_{J-1}$ of length $2L - 1$ which are code vectors in code $C$. These vectors are still orthogonal to the digit at location $\alpha$ and will be used for decoding the code $C$ generated by $G(d, e)$.

Suppose a vector $c$ in $C$ is transmitted and a vector $r = (r_0, r_1, \ldots, r_{2L-1})$ is received. For decoding, we form the following inner products:
\[ A_j = r_j \cdot x_j \quad (j = 0, 1, 2, \ldots, J-1) \]
\[ A_j = r_j \cdot x_j \quad (j = 0, 1, 2, \ldots, J-1) \]
\[ A_j = r_j \cdot x_j \quad (j = 0, 1, 2, \ldots, J-1) \]
\[ A_j = r_j \cdot x_j \quad (j = 0, 1, 2, \ldots, J-1) \]

where $r_j$ denotes the inner product of $r$ and $x_j$ and denotes the $j$th component of $c$. These inner products are called $J$-input majority-logic sums [1]. Since $c$ and $x_j$ are code words in $C$, it must be a sum of the transmitted code word $x$ and an unknown error vector $e = (e_0, e_1, \ldots, e_{2L-1})$, i.e.,
\[ r = x + e \]
\[ r = x + e \]
\[ r = x + e \]
\[ r = x + e \]

Since $x_j = 0$ for $j = 0, 1, 2, \ldots, J - 1$, we obtain, from (3), the following relations between the parity-check sums $A_0, A_1, \ldots, A_{J-1}$ and the error digits:
\[ A_j = r_j \cdot x_j = r_j + x_j + e_j \quad (j = 0, 1, 2, \ldots, J-1) \]
\[ A_j = r_j \cdot x_j = r_j + x_j + e_j \quad (j = 0, 1, 2, \ldots, J-1) \]
\[ A_j = r_j \cdot x_j = r_j + x_j + e_j \quad (j = 0, 1, 2, \ldots, J-1) \]
\[ A_j = r_j \cdot x_j = r_j + x_j + e_j \quad (j = 0, 1, 2, \ldots, J-1) \]

(4)

Suppose there is $j$ such that $x_j = 1$. Then we will show that the error digit $e_j$ can be correctly determined from the parity-check sums. If $e_j = 1$, then the nonzero error digit $e_j$ can be found by using the sum of all the nonzero error digits $e_j$ and will be used for decoding the code $C$ generated by $G(x)$.

In the above notation, $\alpha$ is the parity-check digit location.

**4. Numerical parameters**

For the codes described above, we have that the number of the parity-check digits is $2L - 1 = \deg H(x)$, while the total length of the code is $2L - 1$. We will see that
\[ \deg H(x) = 2L - 1 \]
\[ \deg H(x) = 2L - 1 \]
\[ \deg H(x) = 2L - 1 \]
\[ \deg H(x) = 2L - 1 \]

and, for $j < 2L - 1$, give the exact formulas for $\deg H(x)$.

**Lemma 2**

Let $J \geq 2$ be any odd integer. There exists a positive integer $\delta$ such that for all $\delta, 0 \leq \delta \leq J - 1$.

**Proof**

Since $\delta = 1$, the number of units of $2 \delta$ is $\delta$, and has an order $\delta$. We claim that for all positive integers $\delta$, we claim that $\delta$ is an odd number.

Now, we see that
\[ U = \sum_{\delta = \delta}^\infty \frac{M_{\delta} x_{\delta}}{\delta} \]
\[ U = \sum_{\delta = \delta}^\infty \frac{M_{\delta} x_{\delta}}{\delta} \]
\[ U = \sum_{\delta = \delta}^\infty \frac{M_{\delta} x_{\delta}}{\delta} \]
\[ U = \sum_{\delta = \delta}^\infty \frac{M_{\delta} x_{\delta}}{\delta} \]

Thus,
\[ U = \sum_{\delta = \delta}^\infty \frac{M_{\delta} x_{\delta}}{\delta} \]
\[ U = \sum_{\delta = \delta}^\infty \frac{M_{\delta} x_{\delta}}{\delta} \]
\[ U = \sum_{\delta = \delta}^\infty \frac{M_{\delta} x_{\delta}}{\delta} \]
\[ U = \sum_{\delta = \delta}^\infty \frac{M_{\delta} x_{\delta}}{\delta} \]

which goes to 0 as $\delta \rightarrow \infty$.

We now give some additional information about $\gamma$.

**Corollary 4**

In the above notation,
\[ \lim_{\delta \rightarrow \infty} \frac{M_{\delta} x_{\delta}}{\delta} = 1, \]
\[ \lim_{\delta \rightarrow \infty} \frac{M_{\delta} x_{\delta}}{\delta} = 1, \]
\[ \lim_{\delta \rightarrow \infty} \frac{M_{\delta} x_{\delta}}{\delta} = 1, \]
\[ \lim_{\delta \rightarrow \infty} \frac{M_{\delta} x_{\delta}}{\delta} = 1, \]

**Proof**

We claim that $M_{\delta} x_{\delta} = 1$ for all $\delta, 0 \leq \delta \leq J - 1$. We see that this means that $M_{\delta} x_{\delta} = \delta x_{\delta}$. If all the $\delta x_{\delta} \rightarrow 1$, then we have the number of units having a 1 in $\delta x_{\delta}$ and $\delta x_{\delta}$ having 0 in the same number of digits.

Thus, we claim that $\gamma = \delta x_{\delta}$. We see that this means that $\gamma = \delta x_{\delta}$. If all the $\delta x_{\delta} \rightarrow 1$, then we have the number of units having a 1 in $\delta x_{\delta}$ and $\delta x_{\delta}$ having 0 in the same number of digits.

Moreover, we claim that $\gamma = \delta x_{\delta}$. We see that this means that $\gamma = \delta x_{\delta}$. If all the $\delta x_{\delta} \rightarrow 1$, then we have the number of units having a 1 in $\delta x_{\delta}$ and $\delta x_{\delta}$ having 0 in the same number of digits.

Note that $\gamma = 2^\delta$, where $\delta$ is the minimum number of digits in any of the numbers $x_0, x_1, x_2$.
terminating this quantity in general is probably best carried out by direct calculation. We now turn our attention to the $J$'s of the form $2^p - 1$ or $2^p + 1$.

**Theorem 5**

Let $J = 2^p - 1$ for some $k \geq 2$. Then $m = 2k$ and degree $U(J) = (2^p - 1)^2$.

**Proof.**

Clearly the $8$ of Lemma 2 is $k$ and $L_k = 1$. Thus $m = 2k$ for some integer $k$ and

$$L_k = \sum_{i=1}^{\infty} 2^{i-1} a_i .$$

We note that because of the structure of $L_k(L_k = 1)$, for each integer $p_i$

$$p_i = \sum_{i=1}^{\infty} a_i = \sum_{i=1}^{\infty} a_i .$$

Thus, to calculate $U$ it is enough to work with the quantities $L_k, L_k, L_k, \ldots, L_k, L_k$, since they are the minimal elements with respect to $\preceq$. A bit of reflection shows that we can adapt the inclusion-exclusion formula of Theorem 3 to read as follows:

$$U = \sum_{i=1}^{\infty} M_{\preceq} L_k = \sum_{i=1}^{\infty} M_{\preceq} L_k .$$

Furthermore, note that $M_{\preceq} L_k = \ldots = 2^{2k-3}$, so we get that

$$deg U(J) = 2^p - 1 + \sum_{i=1}^{\infty} a_i 2^{i-1} \preceq (2^p - 1)^2 .$$

$$= (2^p - 1)^2 .$$

**Theorem 6**

Let $J = 2^p + 1$ for some $k \geq 1$. Then $m = 2k$ for some $p \geq 1$ and $deg U(J) = (2^p + 1)^2$.

**Proof.**

Since $2^p - 1$ and $2^p + 1$ are both in $L_k$, we can compute $deg U(J)$ directly. Suppose $\theta_i \preceq \theta_i$ for some integers $\theta_i \geq 2^p$ and $\theta_i$. Write $\bigvee_{\theta_i} L_k$ as $a_i \mid a_i \mid \cdots \mid a_i$, where each $a_i$ is a string of $k$ digits and where $\preceq a_i$ and $\preceq a_i$ for $i = 1, \ldots, k$. We note that the above argument shows that $L_k, L_k, L_k, \ldots, L_k, L_k$ are all minimal with respect to $\preceq$.

Rather than using the inclusion-exclusion formula of Theorem 2 to calculate $U(J)$, we calculate it directly. Suppose $\theta_i \preceq \theta_i$ for some integers $\theta_i \geq 2^p$ and $\theta_i$. Write $\bigvee_{\theta_i} L_k$ as $a_i \mid a_i \mid \cdots \mid a_i$, where each $a_i$ is a string of $k$ digits and where $\preceq a_i$ and $\preceq a_i$ for $i = 1, \ldots, k$. We note that the above argument shows that $L_k, L_k, L_k, \ldots, L_k, L_k$ are all minimal with respect to $\preceq$.

Thus, we have two cases: either $2^p - 1$ or $2^p + 1$.

$$\theta_i \preceq \theta_i \preceq \theta_i \preceq \theta_i \preceq \theta_i \preceq \theta_i \preceq \theta_i \preceq \theta_i .$$

Next, we form the polynomial $H(X)$. The polynomial $H(X)$ has $\alpha_i$ as a root if and only if $\alpha_i$ is a root of $H(X)$ and, for every nonzero descendant $h_i$, $h_i$ is also a root of $H(X)$. For example $\alpha_i$ is a root of $H(X)$. The nonzero descendants of $\alpha_i$ are $h_i$ and $h_i$, and both $\alpha_i$ and $\alpha_i$ are roots of $H(X)$.

$$H(X) = \alpha_i \alpha_i \alpha_i \alpha_i \alpha_i \alpha_i \alpha_i \alpha_i$$

and we see that $deg H(X) = 2^p$ as predicted by Theorem 5. The roots $\alpha_i, \alpha_i, \alpha_i, \alpha_i, \alpha_i, \alpha_i, \alpha_i, \alpha_i$ are conjugates, and they have the same minimal polynomial

$$m_i(X) = (X + \alpha_i)(X + \alpha_i)(X + \alpha_i)(X + \alpha_i)(X + \alpha_i)(X + \alpha_i)(X + \alpha_i) .$$

Using Table 1, we obtain

$$m_i(X) = 1 + X + X^2 .$$

We note that the number whose binary expansion has a 1 in the $j$th place if each $b_j$ has a 1 in the $j$th place of its binary expansion. Assume $g$ has a 1 in its representation. Then each $b_j$ must have at least those $1$s. In order for $\alpha_i \preceq \alpha_i$, we must pick the remaining digits of the $b_j$ so that some $b_j$ has a 0 in each of the $(k - j)$ positions where $g$ has a 0. If we focus our attention on a particular position and try to fill it in all $b_j$'s simultaneously, we see that this can be done in $2^p - 1$ ways. Thus the $b_j$'s can be picked in $(2^p - 1)^{2^p}$ ways. Each $a_i$ must contain 1, but there are no restrictions on the remaining bits. Thus we see that $|A| = (2^p - 1)^2 (2^p - 1)^2$ and, since there are $(k/2)$ strings $g$ having $1$ for $i = 1, \ldots, k$, we see that

$$U = |A| = \sum_{i=1}^{k/2} |B| (2^p - 1)^2 (2^p - 1)^2.$$
in \( C \), are shown in Table 2. Adding an overall parity-check digit to each of these vectors, we obtain the vectors in Table 3. These are code vectors in \( \bar{C} \) (the extension of \( C \)). Now, we apply the affine permutation

\[
Z = \alpha Y + \alpha^3
\]

permutation to the components of \( u', u, u_2 \). The resultant vectors are given in Table 4. Deleting the overall parity-check digit from the above vectors, we obtain the vectors in Table 3, which are vectors in \( C \). We see that these vectors are orthogonal on the digit at location \( \alpha^3 \). Let

\[
r = (r_1, r_2, r_3, r_4, r_5, r_6, r_7, r_8)
\]

be the received vector. Then the parity-check sums orthogonal on error digit \( r_8 \) are

\[
A_k = r_8 \oplus r_4 \oplus r_3 \oplus r_2 \oplus r_1 \oplus r_6 \oplus r_5
\]

The decoding circuit is shown in Fig. 1. The code is capable of correcting any single error over the span of 15 digits. The code \( C \) has minimum distance 4 (the generator polynomial has weight 4). Thus, the code is capable of correcting single errors and detecting any double errors.

5. Summary

In this paper we have investigated a class of one-step majority-logic decodable codes. A method of decoding these codes has been presented. Combinatorial expressions for determining the dimensions of these codes have been derived. These codes are effective compared with other majority-logic decodable codes [3, pp. 176-177]. Most im-

Table 4 Resulting vectors after permutation.

<table>
<thead>
<tr>
<th>( \alpha' )</th>
<th>( a' )</th>
<th>( a' )</th>
<th>( a' )</th>
<th>( a' )</th>
<th>( a' )</th>
<th>( a' )</th>
<th>( a' )</th>
<th>( a' )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( u' )</td>
<td>( 1 )</td>
<td>( 0 )</td>
<td>( 0 )</td>
<td>( 1 )</td>
<td>( 0 )</td>
<td>( 1 )</td>
<td>( 0 )</td>
<td>( 0 )</td>
</tr>
<tr>
<td>( u )</td>
<td>( 1 )</td>
<td>( 0 )</td>
<td>( 0 )</td>
<td>( 1 )</td>
<td>( 0 )</td>
<td>( 0 )</td>
<td>( 1 )</td>
<td>( 0 )</td>
</tr>
<tr>
<td>( u_2 )</td>
<td>( 0 )</td>
<td>( 0 )</td>
<td>( 1 )</td>
<td>( 0 )</td>
<td>( 1 )</td>
<td>( 0 )</td>
<td>( 0 )</td>
<td>( 1 )</td>
</tr>
</tbody>
</table>

Table 5 Vectors after deletion of parity-check digit.

<table>
<thead>
<tr>
<th>( \alpha' )</th>
<th>( a' )</th>
<th>( a' )</th>
<th>( a' )</th>
<th>( a' )</th>
<th>( a' )</th>
<th>( a' )</th>
<th>( a' )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( r_8 )</td>
<td>( 1 )</td>
<td>( 0 )</td>
<td>( 0 )</td>
<td>( 1 )</td>
<td>( 0 )</td>
<td>( 1 )</td>
<td>( 0 )</td>
</tr>
<tr>
<td>( r_7 )</td>
<td>( 0 )</td>
<td>( 0 )</td>
<td>( 0 )</td>
<td>( 1 )</td>
<td>( 0 )</td>
<td>( 1 )</td>
<td>( 0 )</td>
</tr>
<tr>
<td>( r_6 )</td>
<td>( 1 )</td>
<td>( 0 )</td>
<td>( 0 )</td>
<td>( 0 )</td>
<td>( 1 )</td>
<td>( 0 )</td>
<td>( 1 )</td>
</tr>
</tbody>
</table>

Table 6 Some one-step majority-logic decodable codes.

<table>
<thead>
<tr>
<th>( n )</th>
<th>( k )</th>
<th>( r_{\text{ML}} )</th>
<th>( n )</th>
<th>( k )</th>
<th>( r_{\text{ML}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>8</td>
<td>1</td>
<td>2047</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>2</td>
<td>15</td>
<td>4</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>375</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>2</td>
<td>375</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>255</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>255</td>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>255</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>255</td>
<td>5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>255</td>
<td>6</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>255</td>
<td>7</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>255</td>
<td>8</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>255</td>
<td>9</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>255</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>255</td>
<td>11</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>255</td>
<td>12</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>255</td>
<td>13</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>255</td>
<td>14</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>255</td>
<td>15</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>255</td>
<td>16</td>
<td>16</td>
<td></td>
</tr>
</tbody>
</table>

References