N87-17954

An Algorithm to Design Finite Field Multipliers Using a Self-Dual Normal Basis

C. C. Wang

Communications Systems Research Section

Finite field multiplication is central in the implementation of some error-correcting coders. Massey and Omura [4] have presented a revolutionary design for multiplication in a finite field. In their design, a normal basis is utilized to represent the elements of the field. In this article, the concept of using a self-dual normal basis to design the Massey-Omura finite field multiplier is presented. The article first presents an algorithm to locate a self-dual normal basis for $GF(2^m)$ for odd m. Then a method to construct the product function for designing the Massey-Omura multiplier is developed. It is shown that the construction of the product function based on a self-dual basis is simpler than that based on an arbitrary normal basis.

I. Introduction

Finite field multiplication is central in the implementation of some error-correcting coders [1] [2] and authentication devices [3]. There is a need for good multiplication algorithms that can be easily realized. Massey and Omura [4] have developed a new algorithm for multiplication in a Galois field based on a normal basis representation. Using this normal basis, the design of the finite field multiplier is simple and regular [5]. The product components can be obtained by the same logical function operating on the cyclically shifted versions of the components of the multiplicand and multiplier. Hence, designing a Massey-Omura multiplier is essentially designing this product function. An architecture for implementing Massey-Omura multipliers in $GF(2^m)$ was presented in [5]. The normal basis used in the design of [5] is the linearly independent roots of a generating polynomial of $GF(2^m)$. However, it is very difficult to verify the linear independence of the roots of a polynomial. Wah and Wang [6] [7] have shown that if $m + 1$ is a prime and 2 is primitive mod $(m + 1)$, the all-one polynomial of degree m is irreducible and its roots constitute a normal basis. Pei, Wang and Omura [8] have also presented necessary and sufficient conditions for an element to generate a normal basis for the field $GF(2^m)$ for some particular m's. Recently a generalized algorithm to locate a normal basis in any field has been developed [9]. In [9], the concept of dual basis is used to design the product function of the Massey-Omura multiplier.

In this article, a self-dual normal basis is used to design the Massey-Omura multiplier. It is well known [1] that there exists a self-dual normal basis in $GF(2^m)$ if m is odd. This article will show that the construction of the product function for a self-dual normal basis is simpler than that for an arbitrary normal basis. It also presents an algorithm to locate a self-dual normal basis in $GF(2^m)$ for odd m. Finally, a method to construct the product function is developed.

II. Massey-Omura Finite Field Multiplier

The fundamental concept of Massey-Omura finite field multiplication [4] [5] [9] is based on the utilization of a normal basis of the form $\{\alpha, \alpha^2, \alpha^4, \cdots, \alpha^{2^{m-1}}\}$. Multiplication in the normal basis representation requires the same logic circuitry for any one product component as it does for any other product component. Adjacent product-component circuits differ only in their inputs, which are cyclically shifted versions of one another.

Let $\{\alpha, \alpha^2, \alpha^4, \cdots, \alpha^{2^{m-1}}\}$ be a normal basis for $GF(2^m)$. Any two elements y and z in $GF(2^m)$ can be expressed as

$$y = y_0\alpha + y_1\alpha^2 + y_2\alpha^{2^2} + \cdots + y_{m-1}\alpha^{2^{m-1}}$$

$$= \sum_{i=0}^{m-1} y_i\alpha^{2^i} \tag{1}$$

$$z = z_0\alpha + z_1\alpha^2 + z_2\alpha^{2^2} + \cdots + z_{m-1}\alpha^{2^{m-1}}$$

$$= \sum_{i=0}^{m-1} z_i\alpha^{2^i} \tag{2}$$

Let

$$\omega = y \cdot z$$

$$= \omega_0\alpha + \omega_1\alpha^2 + \omega_2\alpha^{2^2} + \cdots + \omega_{m-1}\alpha^{2^{m-1}}$$

$$= \sum_{k=0}^{m-1} \omega_k\alpha^{2^k} \tag{3}$$

Then, as stated in [4] [5] [9],

$$\begin{aligned}
\omega_{m-1} &= f(y_0, y_1, y_2, \cdots, y_{m-1}; \\
&\qquad z_0, z_1, z_2, \cdots, z_{m-1}) \\
\omega_{m-2} &= f(y_{m-1}, y_0, y_1, \cdots, y_{m-2}; \\
&\qquad z_{m-1}, z_0, z_1, \cdots, z_{m-2}) \\
&\vdots \\
\omega_1 &= f(y_2, y_3, \cdots, y_{m-1}, y_0, y_1; \\
&\qquad z_2, z_3, \cdots, z_{m-1}, z_0, z_1) \\
\omega_0 &= f(y_1, y_2, \cdots, y_{m-1}, y_0; \\
&\qquad z_1, z_2, \cdots, z_{m-1}, z_0)
\end{aligned} \tag{4}$$

where

$$f(a_0, a_1, \cdots, a_{m-1}; b_0, b_1, \cdots, b_{m-1})$$

$$= \sum_{i=0}^{m-1}\sum_{j=0}^{m-1} \rho_{ij}a_ib_j \tag{4a}$$

with $\rho_{ij} = 0$ or 1. Therefore, the central problem in designing a Massey-Omura multiplier is to construct the product function f given in (4a). A product function can be constructed in such a way that the coefficient ρ_{ij} of a_ib_j in (4a) is

$$\rho_{ij} = Tr\left(\alpha^{2^i} \cdot \alpha^{2^j} \cdot \gamma^{2^{m-1}}\right) \tag{5}$$

where $Tr(x)$ denotes the trace value of the element x in $GF(2^m)$ and $\{\gamma, \gamma^2, \gamma^{2^2}, \cdots, \gamma^{2^{m-1}}\}$ is the dual basis to the basis $\{\alpha, \alpha^2, \alpha^{2^2}, \cdots, \alpha^{2^{m-1}}\}$ [9].

A simple and equivalent way to represent the product function f is by means of a Boolean matrix

$$\bar{\Omega} = \left[\rho_{ij}\right]_{i,j=0}^{m-1} \tag{6}$$

where the i-j entry ρ_{ij} of $\bar{\Omega}$ is the coefficient of $a_i\,b_j$ given in (5).

III. Properties of Boolean Matrix Generated by a Self-Dual Normal Basis

A self-dual basis is a basis whose dual basis is itself. It is known [1] that, if m is odd, $GF(2^m)$ has a self-dual normal

C-2

basis. Let $\{\alpha, \alpha^2, \alpha^{2^2}, \cdots, \alpha^{2^{m-1}}\}$ be a self-dual normal basis, that is, $Tr(\alpha^{2^i} \cdot \alpha^{2^j}) = \delta_{ij}$ where $\delta_{ij} = 0$ for $i \neq j$ and 1 for $i = j$. From (5), the Boolean matrix associated with this self-dual normal basis can be written as

$$\bar{\Omega} = \left[\rho^*_{ij}\right]_{i,j=0}^{m-1} \tag{7}$$

where

$$\rho^*_{ij} = Tr\left(\alpha^{2^i} \cdot \alpha^{2^j} \cdot \alpha^{2^{m-1}}\right) \tag{7a}$$

Three properties of the Boolean matrix $\bar{\Omega}$ have been proved in [9]. They are

Property 1

$\bar{\Omega}$ is symmetric, that is, $\rho^*_{ij} = \rho^*_{ji}$.

Property 2

$$\rho^*_{ii} = \begin{cases} 0 \text{ if } i \neq m-2 \\ 1 \text{ if } i = m-2 \end{cases}$$

Property 3

$$\sum_{i=0}^{m-1} \rho^*_{ij} = \begin{cases} 0, & j \neq m-1 \\ 1, & j = m-1 \end{cases}$$

In addition, there are two more properties for the Boolean matrix generated by a self-dual normal basis.

Property 4

$$\rho^*_{i,m-1} = \rho^*_{m-1,i} = \delta_{i0}$$

Proof:

$$\begin{aligned} \rho^*_{i,m-1} &= \rho^*_{m-1,i} \\ &= Tr\left(\alpha^{2^i} \cdot \alpha^{2^{m-1}} \cdot \alpha^{2^{m-1}}\right) \\ &= Tr\left(\alpha^{2^i} \cdot \alpha^{2^m}\right) \\ &= Tr\left(\alpha^{2^i} \cdot \alpha^{2^0}\right) = \delta_{i0}\,. \end{aligned}$$

Property 5

$$\rho^*_{ij} = \rho^*_{(m-1+i-j)(m-j-2)} = \rho^*_{(j-i-1)(m-i-2)}$$

for $i < j$ and $0 \leqslant i,j < m-1$

Proof:

Since $Tr(\alpha) = Tr(\alpha^2)$,

$$\begin{aligned} \rho^*_{ij} &= Tr\left(\alpha^{2^i}\alpha^{2^j}\alpha^{2^{m-1}}\right) \\ &= Tr\left(\alpha^{2^{[i+(m-1-i)]}} \cdot \alpha^{2^{[j+(m-1-i)]}} \right. \\ &\qquad \left. \cdot\, \alpha^{2^{[(m-1)+(m-1-i)]}}\right) \\ &= Tr\left(\alpha^{2^{m-1}} \cdot \alpha^{2^{(m-1-i+j)}} \cdot \alpha^{2^{(2m-2-i)}}\right) \\ &= Tr\left(\alpha^{2^{m-1}} \cdot \alpha^{2^{(j-i-1)}} \cdot \alpha^{2^{(m-2-i)}}\right) \\ &= \rho^*_{(j-i-1)(m-2-i)} \end{aligned}$$

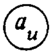
Also,

$$\begin{aligned} \rho^*_{ij} &= Tr\left(\alpha^{2^{[i+(m-1-j)]}} \cdot \alpha^{2^{[j+(m-1-j)]}} \right. \\ &\qquad \left. \cdot\, \alpha^{2^{[(m-1)+(m-1-j)]}}\right) \\ &= Tr\left(\alpha^{2^{(m-1+i-j)}} \cdot \alpha^{2^{m-1}} \cdot \alpha^{2^{(2m-2-j)}}\right) \\ &= Tr\left(\alpha^{2^{(m-1+i-j)}} \cdot \alpha^{2^{m-1}} \cdot \alpha^{2^{(m-2-j)}}\right) \\ &= \rho^*_{(m-1+i-j)(m-j-2)} \end{aligned}$$

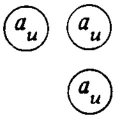
Property 4 implies that the components of the last column and the last row of $\bar{\Omega}$ are all zeros except the first component as shown in Fig. 1. Property 5 illustrates a triangular symmetric structure as shown in Fig.1. This triangular symmetry is described as follows.

Since $\bar{\Omega}$ is symmetric with respect to the diagonal (Property 1), it is sufficient to discuss only the upper-right triangle of $\bar{\Omega}$. Ignoring the main diagonal and the last column, the upper-right triangular portion of the matrix consists of $[m/3]$ *equilateral* triangles in the sense that the numbers of elements on all of the three sides of each triangle are the same. Here, $[x]$ denotes the greatest integer which is smaller than or equal to x. Let Δ_1 denote the outer-most (largest) triangle, and Δ_i the ith outer-most (ith largest) triangle. The triangular symmetric structure is such that the sequences of the vectors counting clockwise on three sides of the triangle Δ_i are identical. Define this identical sequence by $\underline{\nu}_i$. As shown in Fig. 1, $\underline{\nu}_i = (a_i, b_i, c_i, \ldots)$ where $a_i, b_i, c_i, \ldots \in GF(2)$. The dimension of $\underline{\nu}_i$ is $(m-3i)$. As the structure merges toward the inner-most (smallest) triangle, one of the following three possible patterns will happen.

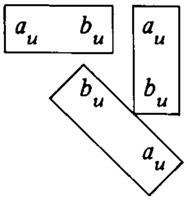
(i) If $m = 0 \bmod 3$, then



(ii) If $m = 1 \bmod 3$, then



(iii) If $m = 2 \bmod 3$, then



where $a_u, b_u \in GF(2)$, and $u = [m/3]$.

One advantage of using this particular Boolean matrix is that its construction requires fewer trace computations. For $GF(2^m)$, the number of trace computations required to construct this Boolean matrix is

$$\frac{m^2 - 3m + 2}{6}, \quad \text{if } m \neq 0 \bmod 3; \text{ and}$$

$$\frac{m^2 - 3m}{6} + 1, \quad \text{if } m = 0 \bmod 3,$$

which is less than one-third of that required for a Boolean matrix corresponding to an arbitrary normal basis as given in [9].

IV. Locating a Self-Dual Normal Basis in $GF(2^m)$ When m Is Odd

Theorem 26 of Chapter 4 of [1] shows that $GF(2^m)$ has a self-dual normal basis if m is odd. In this section, a method to locate a self-dual normal basis for $GF(2^m)$ when m is odd is presented. Let $\{\underline{\alpha}\} \triangleq \{\alpha, \alpha^2, \alpha^{2^2}, \ldots, \alpha^{2^{m-1}}\}$ be an arbitrary normal basis and $\{\underline{\beta}\} \triangleq \{\beta, \beta^2, \beta^{2^2}, \ldots, \beta^{2^{m-1}}\}$ be a self-dual normal basis in $GF(2^m)$. Then α can be expressed

$$\alpha = b_0\beta + b_1\beta^2 + b_2\beta^{2^2} + \ldots + b_{m-1}\beta^{2^{m-1}} \tag{8}$$

Due to the fact that $\beta^{2^m} = \beta$, one can obtain

$$\begin{bmatrix} \alpha \\ \alpha^2 \\ \alpha^{2^2} \\ \vdots \\ \alpha^{2^{m-1}} \end{bmatrix} = \bar{B} \begin{bmatrix} \beta \\ \beta^2 \\ \beta^{2^2} \\ \vdots \\ \beta^{2^{m-1}} \end{bmatrix} \tag{9}$$

where

$$\bar{B} = \begin{bmatrix} b_0 & b_1 & b_2 & \cdots & b_{m-1} \\ b_{m-1} & b_0 & b_1 & \cdots & b_{m-2} \\ b_{m-2} & b_{m-1} & b_0 & \cdots & b_{m-3} \\ \vdots & \vdots & \vdots & & \vdots \\ b_1 & b_2 & b_3 & \cdots & b_0 \end{bmatrix} \tag{9a}$$

is the transformation matrix from the basis $\{\underline{\beta}\}$ to the basis $\{\alpha\}$. Clearly, $\bar{B}$ is invertible. Taking the transpose of (9) results in

$$\left[\alpha, \alpha^2, \alpha^{2^2}, \ldots, \alpha^{2^{m-1}}\right] = \left[\beta, \beta^2, \beta^{2^2}, \ldots, \beta^{2^{m-1}}\right] \bar{B}^T \tag{10}$$

Multiplying (9) by (10), one has

$$\begin{bmatrix} \alpha \\ \alpha^2 \\ \alpha^{2^2} \\ \vdots \\ \alpha^{2^{m-1}} \end{bmatrix} \begin{bmatrix} \alpha\, \alpha^2\, \alpha^{2^2} \ldots \alpha^{2^{m-1}} \end{bmatrix}$$

$$= \bar{B} \begin{bmatrix} \beta \\ \beta^2 \\ \beta^{2^2} \\ \vdots \\ \beta^{2^{m-1}} \end{bmatrix} \begin{bmatrix} \beta\, \beta^2\, \beta^{2^2} \ldots \beta^{2^{m-1}} \end{bmatrix} \bar{B}^T \qquad (11)$$

Carrying out the multiplication of the column and row vectors, and then taking the trace function Tr on both sides of (11), it can be shown that

$$\begin{aligned} \bar{F}(\alpha) &\triangleq \left[F_{ij}\right]_{i,j=0}^{m-1} \\ &\triangleq \left[Tr\left(\alpha^{2^i+2^j}\right)\right]_{i,j=0}^{m-1} \\ &= \bar{B}\bar{B}^T \end{aligned} \qquad (12)$$

since $Tr(\beta^{2^i+2^j}) = \delta_{ij}$.

To locate the self-dual normal basis $\{\underline{\beta}\}$ from $\{\underline{\alpha}\}$, $\bar{B}$ of (9a) needs to be solved from (12). Since $F_{ij}(\alpha) = F_{\overline{i-1}\;\overline{j-1}}(\alpha)$ where $\overline{j-1} = (j-1) \bmod m$, in (12), it is sufficient to consider only the equality between the first row of $\bar{F}(\alpha)$ and the first row of the product of $\bar{B}\bar{B}^T$. Therefore,

$$\left.\begin{aligned} Tr(\alpha^2) &= b_0^2 + b_1^2 + b_2^2 + \ldots + b_{m-1}^2 \\ Tr(\alpha^3) &= b_0 b_{m-1} + b_1 b_0 + b_2 b_1 + \ldots + b_{m-1} b_{m-2} \\ &\vdots \\ Tr(\alpha^{2^j+1}) &= \sum_{k=0}^{m-1} b_k b_{m-j+k} \\ &\vdots \\ Tr(\alpha^{2^{m-1}+1}) &= b_0 b_1 + b_1 b_2 + b_2 b_3 + \ldots + b_{m-1} b_0 \end{aligned}\right\} \qquad (13)$$

Since $\{\underline{\alpha}\}$ is a normal basis, $Tr(\alpha^2)$ must be 1. Also, since $b_i \in GF(2)$, $b_i^2 = b_i$. Then the first equation of (13) becomes

$$1 = b_0 + b_1 + b_2 + \ldots + b_{m-1}.$$

This implies that the set of $\{b_0, b_1, b_2, \ldots, b_{m-1}\}$ must have an odd number of 1's.

Applying Lemma 11 of [9], that is, $Tr(\alpha^{2^j+1}) = Tr(\alpha^{2^{m-j}+1})$ for $1 \leqslant j < m/2$, it can be seen that, ignoring the first equation, the first half of the remaining equations in (13) is identical to the second half of the equations in a reverse order. This means that (13) has at most $(m-1)/2 + 1 = (m+1)/2$ linearly independent equations which are

$$\left.\begin{aligned} b_0 + b_1 + b_2 + \ldots + b_{m-1} &= F_{00}(\alpha) = 1 \\ \sum_{k=0}^{m-1} b_k b_{\overline{m-j+k}} &= F_{0j}(\alpha) \\ \text{for } j &= 1, 2, 3, \ldots, (m-1)/2 \end{aligned}\right\} \qquad (14)$$

But, in equation (14), m unknowns $\{b_0, b_1, b_2, \ldots, b_{m-1}\}$ need to be found. Therefore, the solution is not unique.

Now, an algorithm to find a solution to (14) is demonstrated by using a simple example of $m = 7$. In this case, (14) becomes

$$b_0 + b_1 + b_2 + b_3 + b_4 + b_5 + b_6 = 1 = F_{00} \qquad (15a)$$

$$b_0 b_1 + b_1 b_2 + b_2 b_3 + b_3 b_4 + b_4 b_5 + b_5 b_6 + b_6 b_0 = F_{01} \qquad (15b)$$

$$b_0b_2 + b_1b_3 + b_2b_4 + b_3b_5 + b_4b_6 + b_5b_0 + b_6b_1 = F_{02} \tag{15c}$$

$$b_0b_3 + b_1b_4 + b_2b_5 + b_3b_6 + b_4b_0 + b_5b_1 + b_6b_2 = F_{03} \tag{15d}$$

The purpose of the algorithm is to find a possible solution vector $\underline{b} \triangleq (b_0, b_1, b_2, b_3, b_4, b_5, b_6)$, for a given vector $\underline{t} \triangleq (F_{00}, F_{01}, F_{02}, F_{03})$ under the condition that $F_{00} = 1$. Notice that the left hand sides of equations (15b), (15c) and (15d) are the sums of all possible products $b_ib_{\overline{i+k}}$ ($i = 0, 1, 2, \ldots, 6$) for $k = 1, 2,$ and 3 respectively. Let n_0, n_1, n_2, and n_3 be the numbers of 1's to be added in (15a) (15b), (15c), and (15d), respectively. That is, n_0 is the number of 1's in $\underline{b}$, and n_k ($k = 1, 2, 3$) is the number of i's such that $b_ib_{\overline{i+k}} = 1$ where $i = 0, 1, 2, 3, 4, 5, 6$. Note that n_0 must be odd. Since $b_ib_{\overline{i+k}} \in GF(2)$, it is clear that, when $F_{0k} = 0, k = 1, 2, 3, n_k$ must be even (considering 0 is even too). On the other hand, when $F_{0k} = 1, n_k$ must be odd.

In this algorithm, b_0 is assumed to be always 1. Since the first element F_{00} of $\underline{t}$ must be 1, eight possible patterns of vector $\underline{t}$ need to be considered.

Case (i): $\underline{t} = (1, 0, 0, 0)$

As $F_{01} = F_{02} = F_{03} = 0, n_1, n_2$ and n_3 must be even. Recognize that the condition that $b_j = 0$ for all $j \neq 0$ can result in $n_1 = n_2 = n_3 = 0$, and, consequently, satisfy the equalities of (15b), (15c) and (15d). Hence, a possible solution $\underline{b}$ is (1, 0, 0, 0, 0, 0, 0).

Case (ii): $\underline{t} = (1, 0, 0, 1)$

As $F_{03} = 1, n_3$ must be odd. Let $n_3 = 1$. From (15d), $b_3 = 1$ can at least satisfy the condition of $n_3 = 1$ (since $b_0 = 1$). Then, a pattern of $\underline{b} = (\underline{1, X, X, 1}, X, X, X)$, where "X" indicates an undecided value, can be temporarily set up. Since n_0 must be odd, there must be at least one, but not an even number of, j's for $j \neq 0$ or 3 such that $b_j = 1$. Let n_0 be the minimum, that is, let there be only one $j (j \neq 0$ or 3) such that $b_j = 1$. Since $F_{01} = F_{02} = 0$, this j must be chosen so that n_1 and n_2 are both even and $n_3 = 1$. In order to satisfy this condition, this particular j must satisfy the condition that $b_{\overline{j-k}}\, b_j = b_jb_{\overline{j+k}}$ for all k. The only solution for this is that this j is located at the center of a segment which is composed of odd consecutive X's. Hence, $b_5 = 1$, that is, $\underline{b} = (1, X, X, 1, X, \underline{1}, X)$. Now, letting X = 0 satisfies the condition that $F_{01} = F_{02} = 0$ and $F_{03} = 1$. Therefore, a solution to (15) is $\underline{b} = (1, 0, 0, 1, 0, 1, 0)$.

Case (iii): $\underline{t} = (1, 0, 1, 0)$

Following the same rules discussed in Case (ii), a solution $\underline{b}$ to (15) can be sequentially decided as

(1) $\underline{b} = (\underline{1, X, 1}, X, X, X, X)$ because $F_{02} = 1$;

(2) $\underline{b} = (1, \underline{1}, 1, X, X, X, X)$ because, as stated in Case (ii), $\underline{b}$ requires an additional "1" to satisfy the condition of n_0 being odd, and, this additional "1" must be located at the center of a segment of odd consecutive X's;

(3) Letting X = 0, $\underline{b} = (1, 1, 1, 0, 0, 0, 0)$.

Case (iv): $\underline{t} = (1, 1, 0, 0)$

Again, a solution $\underline{b}$ can be sequentially decided as

(1) $\underline{b} = (\underline{1, 1}, X, X, X, X, X)$ because $F_{01} = 1$;

(2) $\underline{b} = (1, 1, X, X, \underline{1}, X, X)$ because of the same reasons stated in step (2) of Case (iii);

(3) Letting X = 0, $\underline{b} = (1, 1, 0, 0, 1, 0, 0)$.

Case (v): $\underline{t} = (1, 0, 1, 1)$

As in Case (ii), the first step is to recognize that $F_{03} = 1$. This gives a pattern of $\underline{b} = (\underline{1, X, X, 1}, X, X, X)$. Since the number of elements in $\underline{b}$ is odd, the locations of 1's must divide the present $\underline{b}$ pattern into two segments of consecutive X's. One segment has an even number of X's, while the other has an odd number of X's. The second step is to recognize that $F_{02} = 1$. As in Case (iii), a pattern of $\underline{1, X, 1}$ should exist in $\underline{b}$. Notice that pattern $\underline{1, X, 1}$ has an *odd* number of bits. In order not to affect the equalities given in (15b), (15c) and (15d), this pattern $\underline{1, X, 1}$ must be placed at the center of a segment in $\underline{b}$ which has an *odd* number of consecutive X's. Therefore $\underline{b} = (1, X, X, 1, \underline{1, X, 1})$. Since n_0 must be odd, letting n_0 be the minimum, the third step is to add an additional "1" in $\underline{b}$. Again, following the same argument described in Case (ii), this additional "1" must be placed at the center of a segment with odd consecutive X's, resulting in $\underline{b} = (1, X, X, 1, 1, \underline{1}, 1)$. Finally, letting X = 0, a solution $\underline{b} = (1, 0, 0, 1, 1, 1, 1)$ to (15) can be obtained.

Casé (vi): $\underline{t} = (1, 1, 0, 1)$

First, since $F_{03} = 1, \underline{b} = (\underline{1, X, X, 1}, X, X, X)$. Next, since $F_{01} = 1$, a pattern of $\underline{1, 1}$ should exist in $\underline{b}$. Similar to what was described in Case (v), since there are an even number of bits in pattern $\underline{1, 1}$, this pattern should be placed at the center of a segment of even consecutive X's in $\underline{b}$. Therefore, $\underline{b}$ becomes $(1, \underline{1, 1}, 1, X, X, X)$. Finally, since n_0 is odd, letting n_0 be the minimum results in $\underline{b} = (1, 1, 1, 1, 0, \underline{1}, 0)$.

Case (vii): $\underline{t} = (1, 1, 1, 0)$

Using the arguments given in Case (v) and Case (vi), $\underline{b}$ can be sequentially decided as

(1) $\underline{b} = (\underline{1, X, 1}, X, X, X, X)$ because $F_{02} = 1$;

(2) $\underline{b} = (1, X, 1, X, \underline{1, 1}, X)$ because $F_{01} = 1$;

(3) $\underline{b} = (1, \underline{1}, 1, X, 1, 1, X)$ because n_0 is odd;

(4) Letting X = 0, $\underline{b} = (1, 1, 1, 0, 1, 1, 0)$, finally.

Case (viii): $\underline{t} = (1, 1, 1, 1)$

Again, $\underline{b}$ can be sequentially decided as

(1) $\underline{b} = (\underline{1, X, X, 1}, X, X, X)$ because $F_{03} = 1$;

(2) $\underline{b} = (1, X, X, 1, \underline{1, X, 1})$ because $F_{02} = 1$;

(3) $\underline{b} = (1, \underline{1, 1}, 1, 1, X, 1)$ because $F_{01} = 1$;

(4) $\underline{b} = (1, 1, 1, 1, 1, \underline{1}, 1)$, because n_0 is odd.

For an arbitrary odd number m, the algorithm of solving $\underline{b} \triangleq (b_0, b_1, b_2, \cdots, b_{m-1})$ to equation (14) for a given $\underline{t} \triangleq (F_{00}, F_{01}, \cdots, F_{0(m-1)/2})$ can be described as follows

```
(i) Set $b_j = 0$ for $j = 0, 1, \cdots, m - 1$.
    Set $k = (m - 1)/2$.
    Set ISTART = 0.
(ii) If $F_{0k} = 0$, go to (iii).
    Else,
    (A) if $b_0$ =. 0, set $b_0 = 1, b_k = 1$ and ISTART = $k$;
    (B) if $b_0 = 1$,
        (a) if ISTART = odd,
            (1) if $k$ = odd,
                $b_{(\mathrm{ISTART}-k)/2} = 1$ and
                $b_{(\mathrm{ISTART}+k)/2} = 1$;
            (2) if $k$ = even,
                $b_{\mathrm{ISTART}+(m-\mathrm{ISTART}-k)/2} = 1$ and
                $b_{m-(m-\mathrm{ISTART}-k)/2} = 1$;
        (b) if ISTART = even,
            (1) if $k$ = odd,
                $b_{\mathrm{ISTART}+(m-\mathrm{ISTART}-k)/2} = 1$ and
                $b_{m-(m-\mathrm{ISTART}-k)/2} = 1$;
            (2) if $k$ = even,
                $b_{(\mathrm{ISTART}-k)/2} = 1$ and
                $b_{(\mathrm{ISTART}+k)/2} = 1$.
(iii) Set $k = k - 1$
(iv) If $k \neq 0$, go to (ii).
(v) If ISTART = 0, $b_0 = 1$.
    Else, (A) if ISTART = even, $b_{(\mathrm{ISTART}+1)/2} = 1$;
          (B) if ISTART = odd, $b_{(m+\mathrm{ISTART})/2} = 1$.
(vi) End.
```

Figure 2 illustrates a flow chart of this algorithm to solve equation (14) when m is odd. It should be pointed out that this algorithm is not the only algorithm for solving $\underline{b}$. However, this algorithm is the optimum in the sense that the number of 1's in $\underline{b}$ is minimum.

Since the matrix $\overline{B}$ in (9) is formed by $\underline{b}$ which depends on only α according to the algorithm described, a self-dual normal basis $(\beta, \beta^2, \beta^{2^2}, \cdots, \beta^{2^{m-1}})$ can be constructed from another normal basis $(\alpha, \alpha^2, \alpha^{2^2}, \cdots, \alpha^{2m-1})$ in such a way that

$$\begin{bmatrix} \beta \\ \beta^2 \\ \beta^{2^2} \\ \cdot \\ \cdot \\ \cdot \\ \beta^{2^{m-1}} \end{bmatrix} = \overline{B}^{-1} \begin{bmatrix} \alpha \\ \alpha^2 \\ \alpha^{2^2} \\ \cdot \\ \cdot \\ \cdot \\ \alpha^{2^{m-1}} \end{bmatrix} \qquad (16)$$

where $\overline{B}^{-1}$ is the inverse of $\overline{B}$

V. Construction of a Boolean Matrix from a Self-Dual Normal Basis When *m* is Odd

For an arbitrary element θ in $GF(2^m)$ such that $Tr(\theta) = 1$. $F_{0j} \triangleq Tr(\theta^{2^j+1})$ for $j = 1, 2, \ldots, m - 1$, can be calculated. Following the algorithm described in the last section, a solution $\underline{b} = (b_0, b_1, b_2, \cdots, b_{m-1})$ to the equation (14) can be obtained. Then, a matrix $\overline{B}$ can be constructed in the form of (9a).

Theorem 1

If $\overline{B}$ is invertible, $\{\theta, \theta^2, \theta^{2^2}, \cdots, \theta^{2^{m-1}}\}$ is a normal basis.

Proof:

Since $\overline{B}$ is invertible, $\overline{B}^{-1}$ exists. It can be easily shown that $\overline{B}^{-1}$ must be of form that the row vectors in $\overline{B}^{-1}$ are the cyclically shifted versions of one another which is the form of $\overline{B}$. Then, $\overline{B}^{-1}$ can be expressed as

$$\overline{B}^{-1} = \begin{bmatrix} b'_0 & b'_1 & b'_2 & \cdots & b'_{m-1} \\ b'_{m-1} & b'_0 & b'_1 & \cdots & b'_{m-2} \\ b'_{m-2} & b'_{m-1} & b'_0 & \cdots & b'_{m-3} \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ b'_1 & b'_2 & b'_3 & \cdots & b'_0 \end{bmatrix}$$

Let

$$\beta = \sum_{i=0}^{m-1} b'_i \theta^{2^i}$$

and

$$\underline{\beta} = [\beta, \beta^2, \beta^{2^2}, \beta^{2^3}, \cdots, \beta^{2^{m-1}}].$$

Then

$$\underline{\beta}^T = \overline{B}^{-1}\, \underline{\theta}^T \tag{17}$$

where

$$\underline{\theta} = [\theta, \theta^2, \theta^{2^2}, \theta^{2^3}, \cdots, \theta^{2^{m-1}}].$$

Let us first prove that $\{\beta, \beta^2, \beta^{2^2}, \cdots, \beta^{2^{m-1}}\}$ is a normal basis. A contradiction proof is used here.

Suppose that $\{\beta, \beta^2, \beta^{2^2}, \cdots, \beta^{2^{m-1}}\}$ are linearly dependent, there must exist a non-zero vector $\underline{C} \triangleq [C_0, C_1, C_2, \cdots, C_{m-1}]$ such that

$$\underline{C}\,\underline{\beta}^T = 0$$

From (17),

$$\underline{C}\,\overline{B}^{-1}\,\underline{\theta}^T = 0$$

Since $\underline{\theta}$ is a normal basis, vector $\underline{C}\overline{B}^{-1}$ must be an all-zero vector. This implies that $\{\underline{b}'_0, \underline{b}'_1, \underline{b}'_2, \cdots, \underline{b}'_{m-1}\}$ are linearly dependent where $\underline{b}'_k$ is the kth row vector of $\overline{B}^{-1}$. It contradicts the fact that $\overline{B}^{-1}$ is invertible. Therefore, $\{\beta, \beta^2, \beta^{2^2}, \ldots, \beta^{2^{m-1}}\}$ is a normal basis. Since $\underline{\theta}^T = \overline{B}\,\underline{\beta}^T$ and $\overline{B}$ is invertible, it is clear that $\{\theta, \theta^2, \theta^{2^2}, \cdots, \theta^{2^{m-1}}\}$ is a normal basis.

Theorem 2

If $\overline{B}$ is invertible, $\{\beta, \beta^2, \beta^{2^2}, \cdots, \beta^{2^{m-1}}\}$ which is constructed by (17) is a self-dual normal basis.

Proof:

From Theorem 1, $\{\theta, \theta^2, \theta^{2^2}, \cdots, \theta^{2^{m-1}}\}$ and $\{\beta, \beta^2, \beta^{2^2}, \cdots, \beta^{2^{m-1}}\}$ are both normal bases. Following the same procedure of (9)(10) and (11) by replacing α by θ arrives to

$$\overline{F}(\theta) = \overline{B}\,\overline{F}(\beta)\,\overline{B}^T$$

where $\overline{F}(x)$ is an $m \times m$ matrix with entry $F_{ij}(x) = Tr(x^{2^i+2^j})$, $i,j = 0, 1, \ldots, m-1$. Then,

$$\overline{B}^{-1}\,\overline{F}(\theta)\,(\overline{B}^{-1})^T = \overline{F}(\beta)$$

Since $\underline{b}$ is a solution of $\overline{F}(\theta) = \overline{B}\,\overline{B}^{\mathrm{T}}$, $\overline{B}^{-1}\,\overline{F}(\theta)\,(\overline{B}^{-1})^T = \mathrm{I}$. Therefore, $\overline{F}(\beta) = \mathrm{I}$, that is $\{\beta, \beta^2, \beta^{2^2}, \cdots, \beta^{2m-1}\}$ is a self-dual normal basis.

Now, an algorithm of constructing a Boolean matrix for a self-dual normal basis for $GF(2^m)$ when m is odd can be described as follows.

Starting with an arbitrary element θ in $GF(2^m)$, one first computes $\{F_{00}(\theta), F_{01}(\theta), F_{02}(\theta), \cdots, F_{0,(m-1)/2}(\theta)\}$. Going through the procedure described in the last section, one can obtain a solution $\underline{b} = (b_0, b_1, b_2, \cdots, b_{m-1})$ to equation (14). After forming a matrix $\overline{B}$ as shown in (9a), one checks whether $\overline{B}$ is invertible. If it is, $(\theta, \theta^2, \theta^{2^2}, \cdots, \theta^{2^{m-1}})$ is a normal basis. If it is not, try another θ until the corresponding matrix $\overline{B}$ is invertible. From the normal basis $(\theta, \theta^2, \theta^{2^2}, \cdots, \theta^{2^{m-1}})$, a self-dual normal basis $(\beta, \beta^2, \beta^{2^2}, \cdots, \beta^{2^{m-1}})$ can be formed by (17). Finally applying Property 1, 2, 4 and 5 in section 3, one can compute $\rho^*_{ij} = Tr(\beta^{2^i} \cdot \beta^{2^j} \cdot \beta^{2^{m-1}})$ for $i = 0, 1, \cdots, [m/3] - 1$ and $j = 2i+1, 2i+2, \cdots, m-3-i$, and then set up the Boolean matrix $\overline{\Omega} = [\rho^*_{ij}]_{i,j=0}^{m-1}$ of structure given in Fig. 1.

Figure 3 illustrates a flow chart of setting up the Boolean matrix. Our initial goal is to determine whether $(\theta, \theta^2, \theta^{2^2}, \cdots, \theta^{2^{m-1}})$ is a normal basis. Theorems 7 and 8 of [9] show two quick check rules to do this before solving the $\underline{b}$. Notice that Theorem 9 of [9] is not applicable here because m is odd. Figures 4(a), (b) and (c) illustrate the Boolean matrices obtained by using this algorithm for $m = 9$, 17 and 31, respectively. [7] also presents the Boolean matrix for $m = 127$. Figure 5 shows the CPU time required on VAX-11/750 to construct the Boolean matrix based on a self-dual normal basis of $GF(2^m)$. Compared to Fig. 5 of [9], it can be seen that, for large m, the computation time required for a self-dual normal basis is reduced to about 1/3 of that for a regular normal basis. For example, for $m = 127$, it takes only 16 minutes to construct a. Boolean matrix versus 40 minutes indicated in Fig. 5 of [9]. This is due to the fact that the number of trace computations required in this algorithm is less than 1/3 of that required in the algorithm stated in [9]. When m is small, the pre-matrix computation which includes the program initial setup and locating the normal basis dominates the computer time. Therefore, Fig. 5 doesn't show significant reduction on computer time when m is small.

In [5], it has been shown that the complexity of the VLSI design of Massey-Omura multipliers depends on the numbers of 1's in Boolean matrix $\overline{\Omega}$. A matrix with fewer 1's is more desirable. Comparing our computer results in this article with those in [9], it is observed that the number of 1's in the Boolean matrix generated by a self-dual normal basis is less than that generated by an arbitrary normal basis.

VI. Conclusion

In this article, it has been shown that the Boolean matrix obtained by a self-dual normal basis maintains a special symmetric structure so that the time required to construct it can be reduced to 1/3 that required for an arbitrary normal basis. To locate a self-dual normal basis in the field $GF(2^m)$ has been a challenging problem. This article has presented an algorithm to locate a self-dual normal basis and then to construct a Boolean matrix when m is odd.

References

1. MacWilliams, F. J., and Sloane, N. J. A., *The Theory of Error-Correcting Codes*, North-Holland Publishing, New York, 1977.

2. Peterson, W. W., and Weldon, Jr., E. J., *Error-Correcting Codes*, MIT Press, Cambridge, 1972.

3. Berkovits, S., Kowalchuk, J., and Schanning, B., "Implementing Public Key Scheme," *IEEE Communications Magazine*, Vol. 17, pp. 2-3, May 1979.

4. Massey, J. L., and Omura, J. K., Patent Application of "Computational Method and Apparatus for Finite Field Arithmetic," submitted in 1981.

5. Wang, C. C., et al., "VLSI Architectures for Computing Multiplications and Inverses in $GF(2^m)$," *IEEE Transactions on Computers*, Vol. C-34, No. 8, pp. 709-717, August 1985.

6. Wah, P. K. S., and Wang, M. Z., "Realization and Application of the Massey-Omura Lock," *Proceedings of International Zurich Seminar*, IEEE Press, pp. 175-182, March 1984.

7. Wang, C. C., *Exponentiation in Finite Fields*, Ph.D. Dissertation, School of Engineering and Applied Science, UCLA, June 1985.

8. Pie, D. Y., Wang, C. C., and Omura, J. K., "Normal Basis of Finite Field $GF(2^m)$," *IEEE Transactions on Information Theory*, Vol. II-32, No. 2, pp. 285-287, March 1986.

9. Wang, C. C., "A Generalized Algorithm to Design Finite Field Normal Basis Multipliers," *TDA Progress Report 42-87*, Jet Propulsion Laboratory, Pasadena, Calif., November 15, 1986, pp. 125-139.

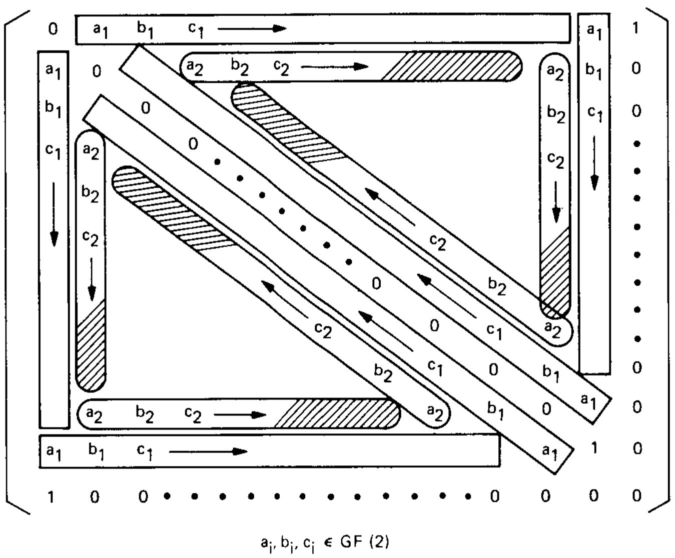
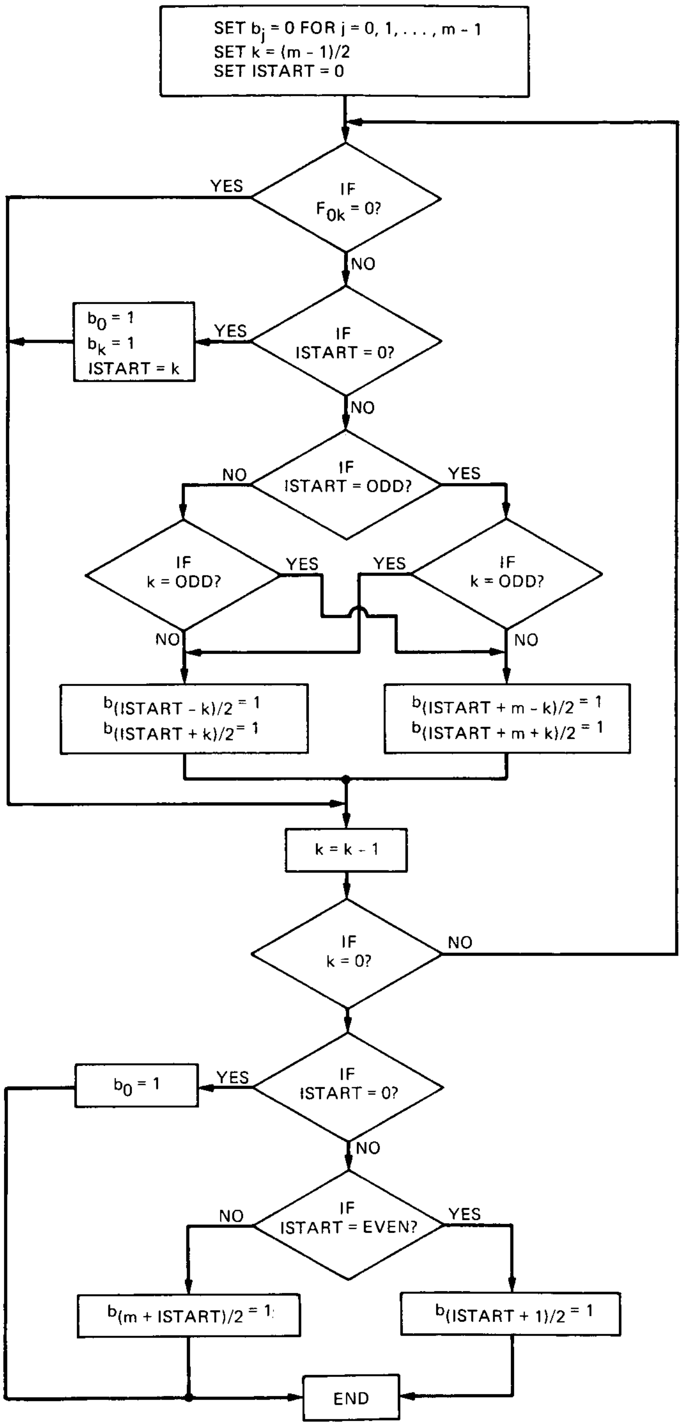


Fig. 1. Structure of a Boolean matrix corresponding to a self-dual normal basis



SET $b_j = 0$ FOR $j = 0, 1, \ldots, m - 1$
SET $k = (m - 1)/2$
SET ISTART = 0
IF $F_{0k} = 0$?
YES
NO
IF ISTART = 0?
$b_0 = 1$
$b_k = 1$
ISTART = k
IF ISTART = ODD?
IF k = ODD?
IF k = ODD?
$b_{(\text{ISTART} - k)/2} = 1$
$b_{(\text{ISTART} + k)/2} = 1$
$b_{(\text{ISTART} + m - k)/2} = 1$
$b_{(\text{ISTART} + m + k)/2} = 1$
$k = k - 1$
IF k = 0?
IF ISTART = 0?
$b_0 = 1$
IF ISTART = EVEN?
$b_{(m + \text{ISTART})/2} = 1$
$b_{(\text{ISTART} + 1)/2} = 1$
END

Fig. 2. Flow chart of computing the vector $\underline{b}$

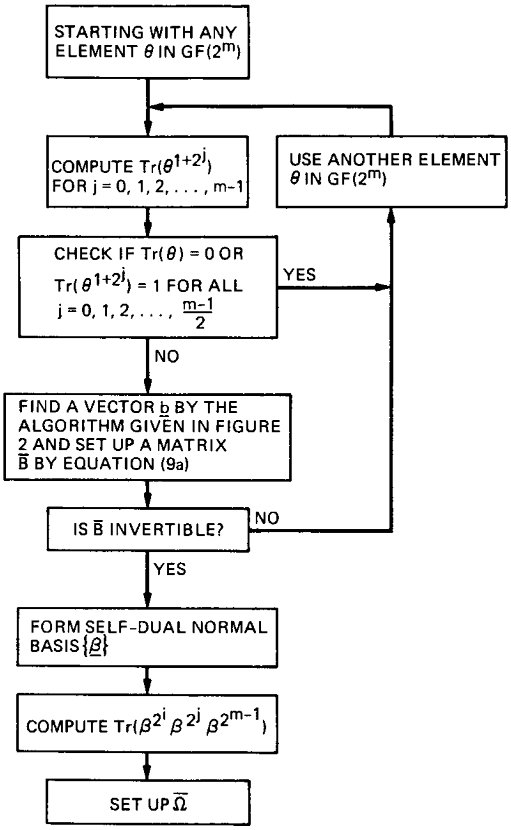

STARTING WITH ANY ELEMENT θ IN GF(2^m)
COMPUTE Tr(θ^{1+2^j}) FOR j = 0, 1, 2, . . . , m-1
USE ANOTHER ELEMENT θ IN GF(2^m)
CHECK IF Tr(θ) = 0 OR Tr(θ^{1+2^j}) = 1 FOR ALL j = 0, 1, 2, . . . , $\frac{m-1}{2}$
YES
NO
FIND A VECTOR $\underline{b}$ BY THE ALGORITHM GIVEN IN FIGURE 2 AND SET UP A MATRIX $\overline{B}$ BY EQUATION (9a)
IS $\overline{B}$ INVERTIBLE?
NO
YES
FORM SELF-DUAL NORMAL BASIS{β}
COMPUTE Tr($\beta^{2^i}\beta^{2^j}\beta^{2^{m-1}}$)
SET UP $\overline{\Omega}$

Fig. 3. Algorithm of constructing the Boolean matrix for a self-dual normal basis

(a)

	0	1	2	3	4	5	6	7	8
0	0	0	0	0	1	1	1	0	1
1	0	0	1	0	1	0	0	0	0
2	0	1	0	1	0	1	1	0	0
3	0	0	1	0	1	1	0	1	0
4	1	1	0	1	0	0	0	1	0
5	1	0	1	1	0	0	0	1	0
6	1	0	1	0	0	0	0	0	0
7	0	0	0	1	1	1	0	1	0
8	1	0	0	0	0	0	0	0	0

NUMBER OF 1's IN BOOLEAN MATRIX = 29

(b)

	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1
	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6
0	0	1	0	1	0	1	1	0	1	0	1	0	0	0	0	1	1
1	1	0	0	0	1	1	1	1	0	0	1	0	0	0	0	0	0
2	0	0	0	0	0	1	0	0	0	1	0	0	1	1	1	1	0
3	1	0	0	0	0	0	1	0	1	1	0	1	0	0	1	0	0
4	0	1	0	0	0	0	0	0	1	1	1	1	1	0	1	1	0
5	1	1	1	0	0	0	1	1	0	0	1	1	1	0	1	1	0
6	1	1	0	1	0	1	0	0	0	1	1	1	0	1	0	0	0
7	0	1	0	0	0	1	0	0	1	0	0	1	1	0	0	1	0
8	1	0	0	1	1	0	0	1	0	0	1	0	0	0	1	0	0
9	0	0	1	1	1	0	1	0	0	0	1	1	0	1	0	1	0
10	1	1	0	0	1	1	1	0	1	1	0	1	1	1	0	0	0
11	0	0	0	1	1	1	1	1	0	1	1	0	0	1	0	0	0
12	0	0	1	0	1	1	0	1	0	0	1	0	0	1	0	0	0
13	0	0	1	0	0	0	1	0	0	1	1	1	1	0	0	0	0
14	0	0	1	1	1	1	0	0	1	0	0	0	0	0	0	1	0
15	1	0	1	0	1	1	0	1	0	1	0	0	0	0	1	1	0
16	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

NUMBER OF 1's IN BOOLEAN MATRIX = 117

(c)

	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2	2	2	3
	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0
0	0	1	1	0	1	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	0	1	0	0	1	1	1
1	1	0	1	0	0	1	1	1	0	1	0	1	1	0	1	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0
2	1	1	0	0	0	1	1	0	1	0	1	1	0	0	0	1	0	1	0	0	1	0	0	1	1	0	1	1	0	0	0
3	0	0	0	0	0	0	1	1	0	0	0	1	1	1	0	1	0	0	1	1	0	1	1	1	1	0	1	1	1	1	0
4	1	0	0	0	0	1	1	0	0	0	0	1	0	1	1	0	1	0	1	0	1	1	1	0	0	0	0	1	0	0	0
5	0	1	1	0	1	0	0	0	1	1	0	1	1	0	1	0	0	0	1	0	0	0	0	1	1	0	0	1	1	1	0
6	1	1	1	1	1	0	0	1	1	1	1	1	1	1	0	0	1	0	1	1	0	0	0	1	1	0	0	0	0	1	0
7	1	1	0	1	0	0	1	0	1	1	0	1	1	0	0	0	1	0	1	0	0	1	0	0	0	1	1	1	1	1	0
8	1	0	1	0	0	1	1	1	0	1	1	0	1	1	0	0	1	1	0	0	1	1	1	0	1	0	1	1	0	1	0
9	1	1	0	0	0	1	1	1	1	0	0	1	1	0	0	0	0	0	1	0	0	0	0	1	0	1	1	0	1	1	0
10	1	0	1	0	0	0	1	0	1	0	0	0	1	0	1	1	0	1	0	0	0	0	1	0	0	1	0	0	1	1	0
11	1	1	1	1	0	1	1	1	0	1	0	0	0	0	0	1	0	1	1	1	0	1	0	1	0	0	1	0	0	1	0
12	1	1	0	1	1	1	1	1	1	1	1	0	0	0	1	1	0	1	0	0	0	1	0	1	1	1	0	1	1	1	0
13	1	0	0	1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0
14	0	1	0	0	1	1	0	0	0	0	1	0	1	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1	1	0	0
15	0	0	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	1	0	0	1	1	0	0	0	0	1	0	1	0	0
16	0	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	1	0	0	0	0	0
17	0	1	1	0	1	0	0	0	1	0	1	1	1	0	1	1	1	0	1	1	0	1	1	1	1	1	1	1	1	0	0
18	0	0	0	1	0	1	1	1	0	1	0	1	0	0	1	0	0	1	0	1	1	1	1	0	1	1	1	0	1	0	0
19	0	1	0	1	1	0	1	0	0	0	0	1	0	0	1	0	0	1	1	0	1	0	1	0	0	0	1	0	1	0	0
20	0	1	1	0	0	0	0	0	1	0	0	0	0	1	0	1	1	0	1	1	0	1	1	0	0	0	1	1	1	1	0
21	1	1	0	1	1	0	0	1	1	0	0	1	1	1	0	1	0	1	1	0	1	0	1	0	1	0	0	1	1	1	0
22	1	1	0	1	1	0	0	0	1	0	1	0	0	1	0	0	0	1	1	1	1	1	0	1	1	0	1	0	0	1	0
23	1	1	1	1	1	1	1	0	0	1	0	1	1	0	0	0	1	1	0	0	0	0	1	0	1	1	1	1	1	0	0
24	0	0	1	1	0	1	1	0	1	0	0	0	1	0	0	0	0	1	1	0	0	1	1	1	0	0	1	1	0	1	0
25	1	1	0	0	0	0	0	1	0	1	1	0	1	0	1	0	1	1	1	0	0	0	0	1	0	0	1	0	0	0	0
26	0	0	1	1	0	0	0	1	1	1	0	1	0	0	1	1	0	1	1	1	1	0	1	1	1	1	0	0	0	0	0
27	0	0	1	1	0	1	0	1	1	0	0	0	1	0	1	0	0	1	0	0	1	1	0	1	1	0	0	0	1	1	0
28	1	0	0	1	1	1	0	1	0	1	1	0	1	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	1	0
29	1	1	0	1	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	0	1	0	0	1	1	1	0
30	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

NUMBER OF 1's IN BOOLEAN MATRIX = 453

Fig. 4. Boolean matrix (a) for $GF(2^9)$, (b) for $GF(2^{17})$, and (c) for $GF(2^{31})$

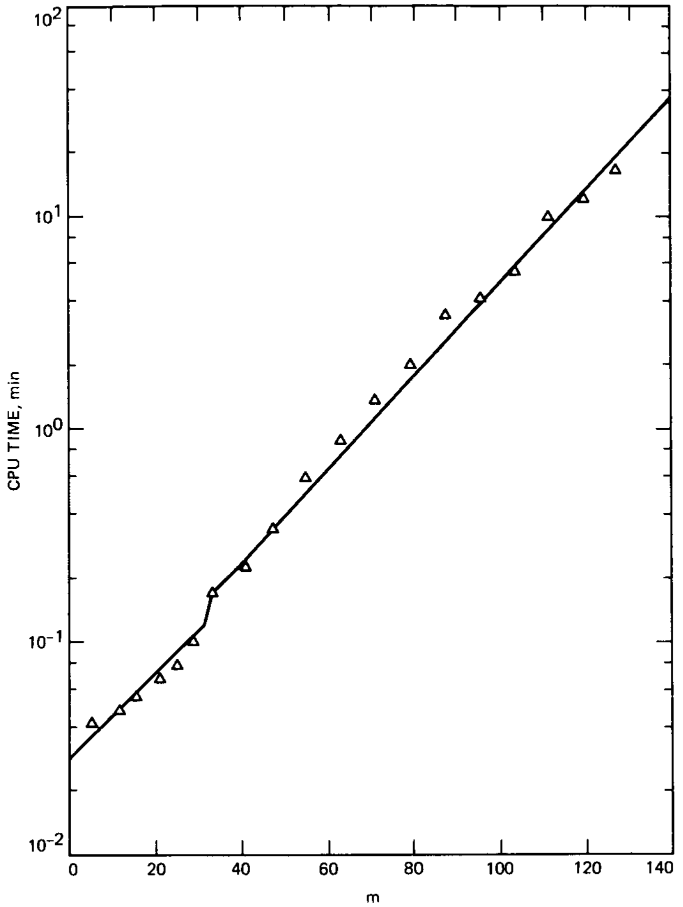


Fig. 5. CPU time required to construct Boolean matrix for $GF(2^m)$