'5

N84 13406

TDA Progress Report 42-75 July–September 1983

VLSI Architectures for Computing Multiplications and Inverses in $GF(2^m)$

C. C. Wang, T. K. Truong, H. M. Shao and L. J. Deutsch
Communications Systems Research Section

J. K. Omura
University of California, Los Angeles

I. S. Reed
University of Southern California

Finite field arithmetic logic is central in the implementation of Reed-Solomon coders and in some cryptographic algorithms. There is a need for good multiplication and inversion algorithms that can be easily realized on VLSI chips. Massey and Omura recently developed a new multiplication algorithm for Galois fields based on a normal basis representation. In this paper, a pipeline structure is developed to realize the Massey-Omura multiplier in the finite field $GF(2^m)$. With the simple squaring property of the normal-basis representation used together with this multiplier, a pipeline architecture is also developed for computing inverse elements in $GF(2^m)$. The designs developed for the Massey-Omura multiplier and the computation of inverse elements are regular, simple, expandable and, therefore, naturally suitable for VLSI implementation.

I. Introduction

Recently, Massey and Omura (Ref. 1) invented a multiplier which obtains the product of two elements in the finite field $GF(2^m)$. In their invention, they utilize a normal basis of form $\{\alpha, \alpha^2, \alpha^4, \cdots, \alpha^{2^{m-1}}\}$ to represent elements of the field where α is the root of an irreducible polynomial of degree m over $GF(2)$. In this basis each element in the field $GF(2^m)$ can be represented by m binary digits.

In the normal-basis representation the squaring of an element in $GF(2^m)$ is readily shown to be a simple cyclic shift of its binary digits. Multiplication in the normal basis representations requires for any one product digit the same logic circuitry as it does for any other product digit. Adjacent product-digit circuits differ only in their inputs which are cyclically shifted versions of one another. In this paper, a pipeline architecture suitable for VLSI design is developed for a Massey-Omura multiplier on $GF(2^m)$.

The conventional method for finding an inverse element in a finite field uses either table look-up or Euclid's algorithms. These methods are not easily realized in a VLSI circuit. However, using a Massey-Omura multiplier, a recursive, pipeline, inversion circuit is developed. This structure consists of four

sets of shift registers, one parallel-type Massey-Omura multiplier and two control signals. Such a design is regular, simple and expandable and, hence, naturally suitable for VLSI implementation.

II. Squaring and Multiplying in a Normal Basis Representation

In this section, the work originally described by Massey and Omura (Ref. 1) is reviewed. It is well known that there always exists a normal basis in the finite field $GF(2^m)$ (Ref. 2) for all positive integers, m. That is, one can find a field element α such that $N = \{\alpha, \alpha^2, \alpha^4, \cdots, \alpha^{2^{(m-1)}}\}$ is a basis set of $GF(2^m)$. Thus every field element $\beta \in GF(2^m)$ can be uniquely expressed as

$$\beta = b_0\alpha + b_1\alpha^2 + b_2\alpha^4 + \cdots + b_{m-1}\alpha^{2^{(m-1)}} \tag{1}$$

where $b_0, b_1, b_2, \cdots, b_{m-1}$ are binary digits and addition is mod-2 addition.

Three useful properties of a finite field $GF(2^m)$ are stated here without proof (for proofs see, for example, Ref. 2). These properties are:

(1) Squaring in $GF(2^m)$ is a linear operation. That is, given any two elements α and β in $GF(2^m)$,

$$(\alpha + \beta)^2 = \alpha^2 + \beta^2 \tag{2}$$

(2) For any element α of $GF(2^m)$,

$$\alpha^{2^m} = \alpha \tag{3}$$

(3) If α is a root of any irreducible polynomial $P(x)$ of degree m over $GF(2)$, the powers, $\alpha, \alpha^2, \alpha^4, \cdots, \alpha^{2^{(m-1)}}$, are in $GF(2^m)$ and constitute a complete set of roots of $P(x)$.

With regard to property (3), Peterson and Weldon (Ref. 3) list a set of irreducible polynomials of degree $m \leq 34$ over $GF(2)$ for which the roots $\{\alpha, \alpha^2, \alpha^4, \cdots, \alpha^{2^{(m-1)}}\}$ are linearly independent. These linear independent roots clearly form a normal basis of $GF(2^m)$.

Suppose that $\{\alpha, \alpha^2, \alpha^4 \cdots, \alpha^{2^{(m-1)}}\}$ is a normal basis of $GF(2^m)$. By (2) and (3) the square of (1) is

$$\begin{aligned}\beta^2 &= b_0\alpha^2 + b_1\alpha^4 + b_2\alpha^8 + \cdots + b_{m-2}\alpha^{2^{(m-1)}} + b_{m-1}\alpha^{2^m} \\ &= b_{m-1}\alpha + b_0\alpha^2 + b_1\alpha^4 + \cdots b_{m-2}\alpha^{2^{(m-1)}}\end{aligned} \tag{4}$$

Thus, if β is represented as a vector of components of the normal basis elements of $GF(2^m)$ in the form $\beta = [b_0, b_1, b_2, \cdots, b_{m-1}]$, then $\beta^2 = [b_{m-1}, b_0, b_1, \cdots, b_{m-2}]$. In the normal basis representation β^2 is a cyclic shift of β. Hence squaring in $GF(2^m)$ can be realized physically by logic circuitry which accomplishes cyclic shifts in a binary register. Such squaring circuitry is illustrated in block form in Fig. 1.

By (2) and (3) it is readily seen that $1 = \alpha + \alpha^2 + \alpha^4 + \cdots + \alpha^{2^{(m-1)}}$ for any element α in $GF(2^m)$. This implies that the normal basis representation of 1 is $(1, 1, 1, \cdots, 1)$.

Let $\beta = [b_0, b_1, \cdots, b_{m-1}]$ and $\gamma = [c_0, c_1, \cdots, c_{m-1}]$ be two elements of $GF(2^m)$ in a normal basis representation. Then the last term d_{m-1} of the product,

$$\delta = \beta \cdot \gamma = [d_0, d_1, \cdots, d_{m-1}], \tag{5}$$

is some binary function of the components of β and γ, i.e.,

$$d_{m-1} = f(b_0, b_1, \cdots, b_{m-1}; c_0, c_1, \cdots, c_{m-1}) \tag{6}$$

Since squaring means a cyclic shift of an element in a normal basis representation, one has

$$\left.\begin{aligned}\delta^2 &= \beta^2 \cdot \gamma^2 \\ &= [b_{m-1}, b_0, b_1, \cdots, b_{m-2}] \\ &\quad \cdot [c_{m-1}, c_0, c_1, \cdots, c_{m-2}] \\ &= [d_{m-1}, d_0, d_1, \cdots, d_{m-2}]\end{aligned}\right\} \tag{7}$$

Hence the last component d_{m-2} of δ^2 is obtained by the same function f in (6) operation on the components of β^2 and γ^2. That is, $d_{m-2} = f(b_{m-1}, b_0, b_1, \cdots, b_{m-2}; c_{m-1}, c_0, c_1, \cdots, c_{m-2})$. By squaring δ repeatedly, it is evident that

$$\left.\begin{aligned}d_{m-1} &= f(b_0, b_1, \cdots, b_{m-1}; c_0, c_1, \cdots, c_{m-1}) \\ d_{m-2} &= f(b_{m-1}, b_0, b_1, \cdots, b_{m-2}; \\ &\qquad c_{m-1}, c_0, c_1, \cdots, c_{m-2}) \\ &\vdots \\ d_0 &= f(b_1, b_2, \cdots, b_{m-1}, b_0; \\ &\qquad c_1, c_2, \cdots, c_{m-1}, c_0)\end{aligned}\right\} \tag{8}$$

The equations in (8) define the Massey-Omura multiplier. In the normal basis representation this multiplier has the property that the same logic function f which is used to find the last component of d_{m-1} of the product δ can be used to find sequentially the remaining components d_{m-2}, d_{m-3}, $\cdots$, d_0 of the product. This feature of the product operation requires only one logic function f of the $2m$ components of β and γ to sequentially compute the m components of the product.

Figure 2 illustrates the logic diagram of the above-described sequential-type Massey-Omura multiplier on $GF(2^m)$. Alternately, for parallel operation this feature permits the use of m identical logic functions, f, for calculating simultaneously all components of the product. In the latter case, the inputs to the m logic functions f are connected directly to the components of β and γ. The only difference in the connections to the components of β or γ to a function f is that they are cyclically shifted versions of one another. Figure 3 shows the structure of the parallel-type Massey-Omura multiplier for the simple case of $m = 4$. The extension of this type of structure to a general case of $GF(2^m)$ is straightforward.

III. A Pipeline Structure for Implementing Massey-Omura Multiplier

A detailed design of a Massey-Omura multiplier is now developed for the finite field $GF(2^4)$. As illustrated in Figs. 2 and 3, the design of either the sequential-type or parallel-type Massey-Omura multiplier must focus on the product function f.

The design of f begins with the selection of an irreducible polynomial $P(x) = x^4 + x^3 + 1$ of degree $m = 4$ over $GF(2)$. This particular polynomial function has linearly independent roots, namely, α, α^2, α^4 and α^8. Hence, the set of roots $\{\alpha, \alpha^2, \alpha^4, \alpha^8\}$ constitutes a normal basis of $GF(2^4)$. Any two elements β and γ in $GF(2^4)$ can be expressed as

$$\begin{aligned} \beta &= b_0\alpha + b_1\alpha^2 + b_2\alpha^4 + b_3\alpha^8 \\ \gamma &= c_0\alpha + c_1\alpha^2 + c_2\alpha^4 + c_3\alpha^8 \end{aligned} \tag{9}$$

By (9) the product of β and γ is

$$\left.\begin{aligned} \delta = \beta\cdot\gamma &= (b_0\alpha + b_1\alpha^2 + b_2\alpha^4 + b_3\alpha^8) \\ &\quad\cdot(c_0\alpha + c_1\alpha^2 + c_2\alpha^4 + c_3\alpha^8) \\ &= d_0\alpha + d_1\alpha^2 + d_2\alpha^4 + d_3\alpha^8 \end{aligned}\right\} \tag{10}$$

By (10) and the fact that $\alpha^4 = \alpha^3 + 1$, one obtains

$$\left.\begin{aligned} d_3 &= b_2c_2 + b_3c_2 + b_2c_3 + b_3c_1 + b_1c_3 \\ &\quad + b_3c_0 + b_0c_3 + b_1c_0 + b_0c_1 \\ d_2 &= b_1c_1 + b_2c_1 + b_1c_2 + b_2c_0 + b_0c_2 \\ &\quad + b_2c_3 + b_3c_2 + b_0c_3 + b_3c_0 \\ d_1 &= b_0c_0 + b_1c_0 + b_0c_1 + b_1c_3 + b_3c_1 \\ &\quad + b_1c_2 + b_2c_1 + b_3c_2 + b_2c_3 \\ d_0 &= b_2c_3 + b_0c_3 + b_3c_0 + b_0c_2 + b_2c_0 \\ &\quad + b_0c_1 + b_1c_0 + b_2c_1 + b_1c_2 \end{aligned}\right\} \tag{11}$$

Comparing (11) with (8), the function f is given by

$$\begin{aligned} f(b_0, b_1, b_2, b_3; c_0, c_1, c_2, c_3) &= b_2c_2 + b_3c_2 + b_2c_3 + b_3c_1 + b_1c_3 \\ &\quad + b_3c_0 + b_0c_3 + b_1c_0 + b_0c_1 \end{aligned} \tag{12}$$

Since the mod-2 sum in (12) can be implemented by the "exclusive or" operation (XOR), the structure of the product function f can be represented by the logic circuit in Fig. 4. This circuit consists of two portions; the left half is an AND plane which computes each term of (12), while the right half is XOR plane which computes the mod-2 sum. The inputs to the AND plane are the complements of the components of β and γ. This is due to the fact that the AND operation in the AND plane is obtained by the NOR operation on the complements of the two digits being ANDed, i.e., $xy = \overline{(\bar{x} + \bar{y})}$ where $\bar{x}$ is the complement of $\bar{x}$.

A pipeline structure of a Massey-Omura multiplier for $GF(2^4)$ is shown in Fig. 5. This structure has a sequential type of operation. For each of the two inputs, corresponding to β and γ, to the f function, an inverter, two sets of shift registers, B and R, and 11 gate transistors are utilized. Note that registers B and R have an identical circuit structure.

In Fig. 5 during the first three clock cycles, when signal $LD = 0$, the complements of b_3, b_2, b_1 and c_3, c_2, c_1 are fed

sequentially into three buffer flip-flops B_k for (k = 1, 2, 3). At the fourth clock cycle, when Ld = 1, the values of $\bar{b}_3$, $\bar{b}_2$, $\bar{b}_1$ and $\bar{c}_3$, $\bar{c}_2$, $\bar{c}_1$, previously stored in buffer registers B_k and $\bar{b}_0$ and $\bar{c}_0$ are shifted into the second set of registers R_k for (k = 1, 2, 3, 4). Then the R-registers are cyclically shifted Such a cyclic-shift operation is needed to sequentially yield the product components d_3, d_2, d_1 and d_0 of δ. While the R-registers are cyclically shifting the components of β (or γ), the components of another element in $GF(2^4)$ following β (or γ) can be fed into the buffer B-registers. Therefore, the structure in Fig. 5 provides a pipeline operation in which no time is lost except for an initial fixed time delay. The VLSI layout of a Massey-Omura multiplier for $GF(2^4)$ is shown in Fig. 6.

Figure 7 illustrates a system structure of a pipelined Massey-Omura multiplier for $GF(2^m)$. For this general case over $GF(2^m)$, the buffer and the cyclic shift mechanism in Fig. 7 have m - 1 and m stages, respectively. Each stage consists of a shift register and a gate transistor. The product function f is a mod-2 sum of AND products of the components of the two inputs being multiplied. Such a circuit for function f consists of an AND programmed logic array (PLA) (Ref. 4) followed by an XOR sequential-PLA. In the XOR sequential-PLA there are several levels of XORs. At each level, the inputs, pair-by-pair, are fed sequentially one-by-one into an XOR as shown in Fig. 4.

Let $n(j)$ be the number of XOR circuits at the j-th level of the XOR sequential-PLA. Then $n(j + 1) = [n(j)/2]$ where $[x]$ is the smallest integer greater than x and where initially, $n(0)$ = total number of terms to be XORed in product function f. At the last level, there is only one XOR circuit and the output is the value of f. In general, if k denotes the number of levels required in the XOR sequential-PLA, $k = [\log_2 n(0)]$.

It should be noted that as m gets large, the number of mod-2 sums in the function f becomes large. In this case, more XORs and as a consequence more levels in the XOR sequential-PLA are required. To maximize the pipeline operation speed, shift registers are required between the XOR levels in order to store the XOR outputs of the intermediate levels.

Another approach to the realization of product function f is to use a standard AND-OR PLA (Ref. 4). This is possible since $x + y = \bar{x}y \vee x\bar{y}$ where $\vee$ denotes inclusive OR. In general, although the design of f by the use of such a PLA is tedious, the product function f can be accomplished in less than one clock cycle. One trade-off for such a design is the large chip area required. The required area for such a PLA increases dramatically with m. Hence, a design utilizing a standard AND-OR PLA to realize f is practical only for small m.

IV. A Pipeline Structure for Computing an Inverse Element in the Finite Field ($GF(2^m)$)

For any α in the finite field $GF(2^m)$, $\alpha^{2^m} = \alpha$. Hence the inverse of α is $\alpha^{-1} = \alpha^{2^m-2}$. Let $2^m - 2$ be decomposed as $2 + 2^2 + 2^3 + \cdots + 2^{m-1}$, then α^{-1} can be expressed as

$$\alpha^{-1} = (\alpha^2) \cdot (\alpha^{2^2}) \cdot (\alpha^{2^3}) \cdot \cdots \cdot (\alpha^{2^{m-1}}) \qquad (13)$$

As discussed in Section II, if α is represented in a normal basis, squaring can be realized by a cyclic shift operation. α^{2^j} is the j-th cyclical shift (CS) of α. Thus, the inverse element α^{-1} can be obtained by using successive cyclic-shift operations and a Massey-Omura multiplier. The algorithm for α^{-1} is the following:

(1) Obtain the cyclic shift of α, i.e., $\alpha^2 = CS(\alpha)$ where CS denotes the cylic shift function. Let $B = CS(\alpha)$ and C = 1. Let k = 0.

(2) Multiply B and C to obtain the product, $D = B \cdot C$. Set $k = k + 1$.

(3) If k = m - 1, $\alpha^{-1} = D$. Stop. If $k < m - 1$, let $B = CS(B)$ and $C = D$.

(4) Go back to (2).

Figure 8 shows a flow chart diagram of this procedure.

This recursive algorithm for computing an inverse element in $GF(2^4)$ can be realized using the circuit shown in Fig. 9. In this circuit the parallel-type Massey-Omura multiplier shown in Fig. 3 with the circuit for the product function f shown in Fig. 4 is utilized.

To illustrate, let Ld_1 and Ld_2 be two control signals with period of four clock signals as shown in Fig. 9. Also let the normal basis representation of α be (a_0, a_1, a_2, a_3). At the end of the third clock pulse, the values $\bar{a}_1$, $\bar{a}_2$, $\bar{a}_3$ are stored in the input buffer flip-flops B_1, B_2, B_3, respectively. During the four clock cycle, $\bar{a}_3$, $\bar{a}_0$, $\bar{a}_1$ and $\bar{a}_2$ are simultaneously shifted to R_1, R_2, R_3 and R_4, respectively. With the appropriate connections among the input buffer flip-flops B_k and flip-flops R_k, the cyclic shift of $\bar{\alpha} = (\bar{a}_0, \bar{a}_1, \bar{a}_2, \bar{a}_3)$, i.e., $\bar{\alpha}^2 = (\bar{a}_3, \bar{a}_0, a_1, a_2)$ is obtained in R. At the fourth clock pulse R_5, R_6, R_7, R_8 are also fed the value "0". These four complementary values of "1" introduce the element 1 in $GF(2^4)$.

As it was discussed in Section II, a parallel-type $GF(2^4)$ Massey-Omura multiplier simultaneously yields four product components d_0, d_1, d_2, d_3. Therefore, during the next three clocks three successive multiplications, i.e., $\beta_1 = 1 \cdot \alpha^2$, $\beta_2 = \beta_1 \cdot \alpha^4$ and $\beta_3 = \beta_2 \cdot \alpha^8$ are performed for the inversion. When the third multiplication is completed, Ld_2 = 1. Thus

the output product digits, which together represent the inverse element α^{-1}, are fed into the output buffer flip-flops B_k. Finally these are sequentially shifted from the inversion circuit.

The above technique for computing the inverse of an element in $GF(2^4)$ takes four clock cycles. During these four clock cycles, the circuit in Fig. 9 allows the bits of the next element (following α) to be fed into it and the bits of the previous element to be shifted out of it, simultaneously. This type of circuit provides a full pipeline capability. A VLSI layout of the pipeline inversion circuitry for $GF(2^4)$ is presented in Fig. 10. Figure 11 shows the system structure of an inversion circuit for the general finite field $FG(2^m)$.

References

1. Massey, J. L., and Omura, J. K., Patent Application of *Computational Method and Apparatus for Finite Field Arithmetic*, submitted in 1981.
2. MacWilliams, F. J., and Sloane, N. J. A., *The Theory of Error-Correcting Codes*, North-Holland Publishing, New York, 1977.
3. Peterson, W. W., and Weldon, E. J., Jr., *Error-Correcting Codes*, MIT Press, Cambridge, 1972.
4. Mead, C., and Conway, L., *Introduction to VLSI Systems*, Addison-Wesley, Reading, 1980.

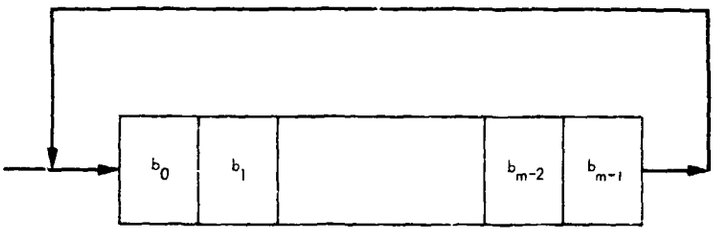

Fig. 1. The squaring operation for a normal-basis representation over $GF(2^m)$

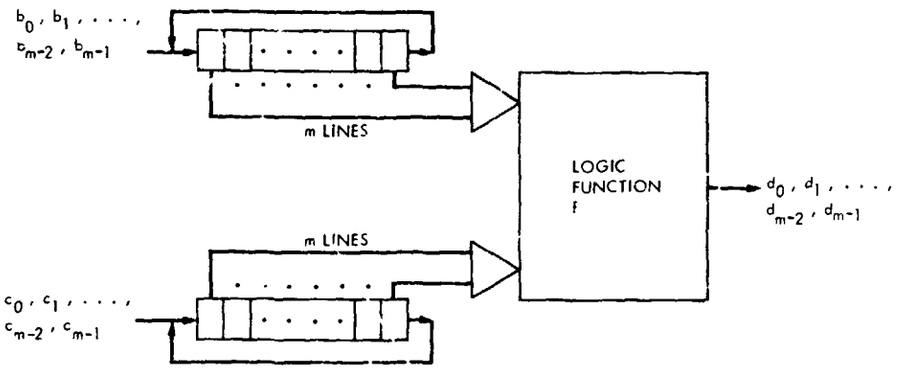

Fig. 2. System-logic diagram of a sequential-type Massey-Omura multiplier over $GF(2^m)$

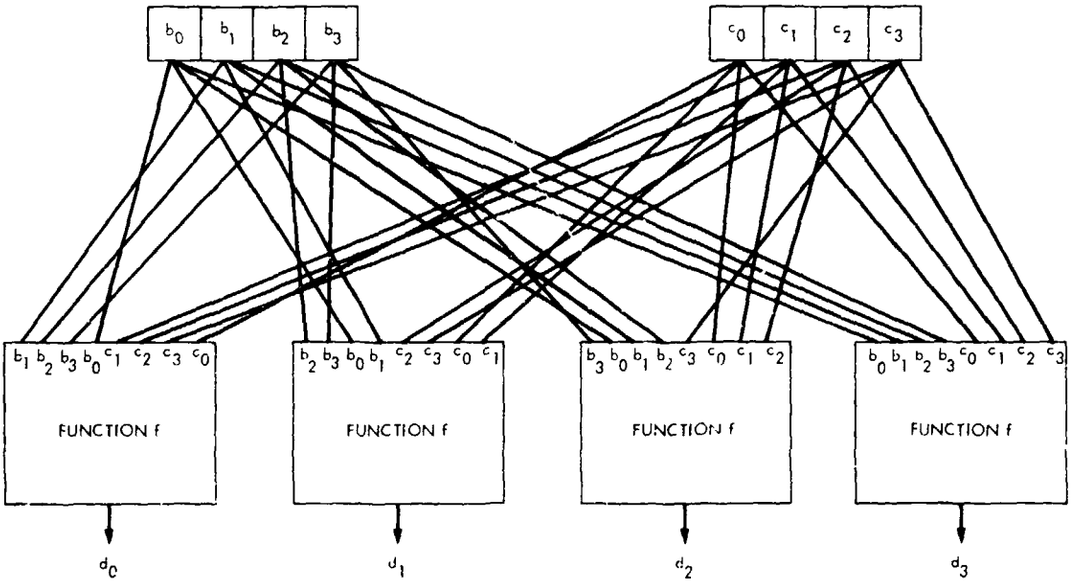

Fig. 3. Architecture of parallel-type Massey-Omura multiplier over $GF(2^4)$

ORIGINAL PAGE IS
OF POOR QUALITY

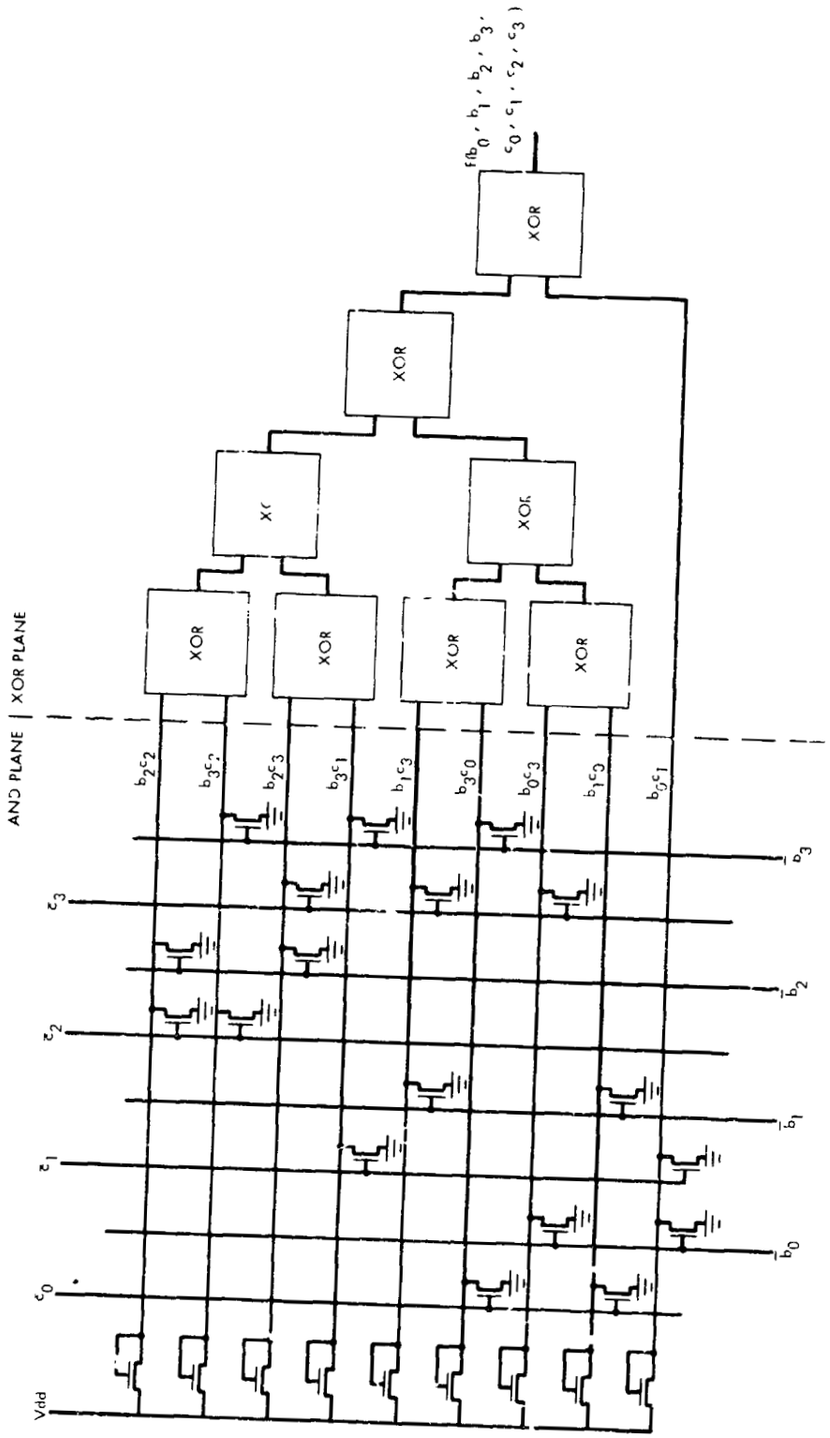

Fig. 4. Circuit diagram of product function f on GF(2⁴)

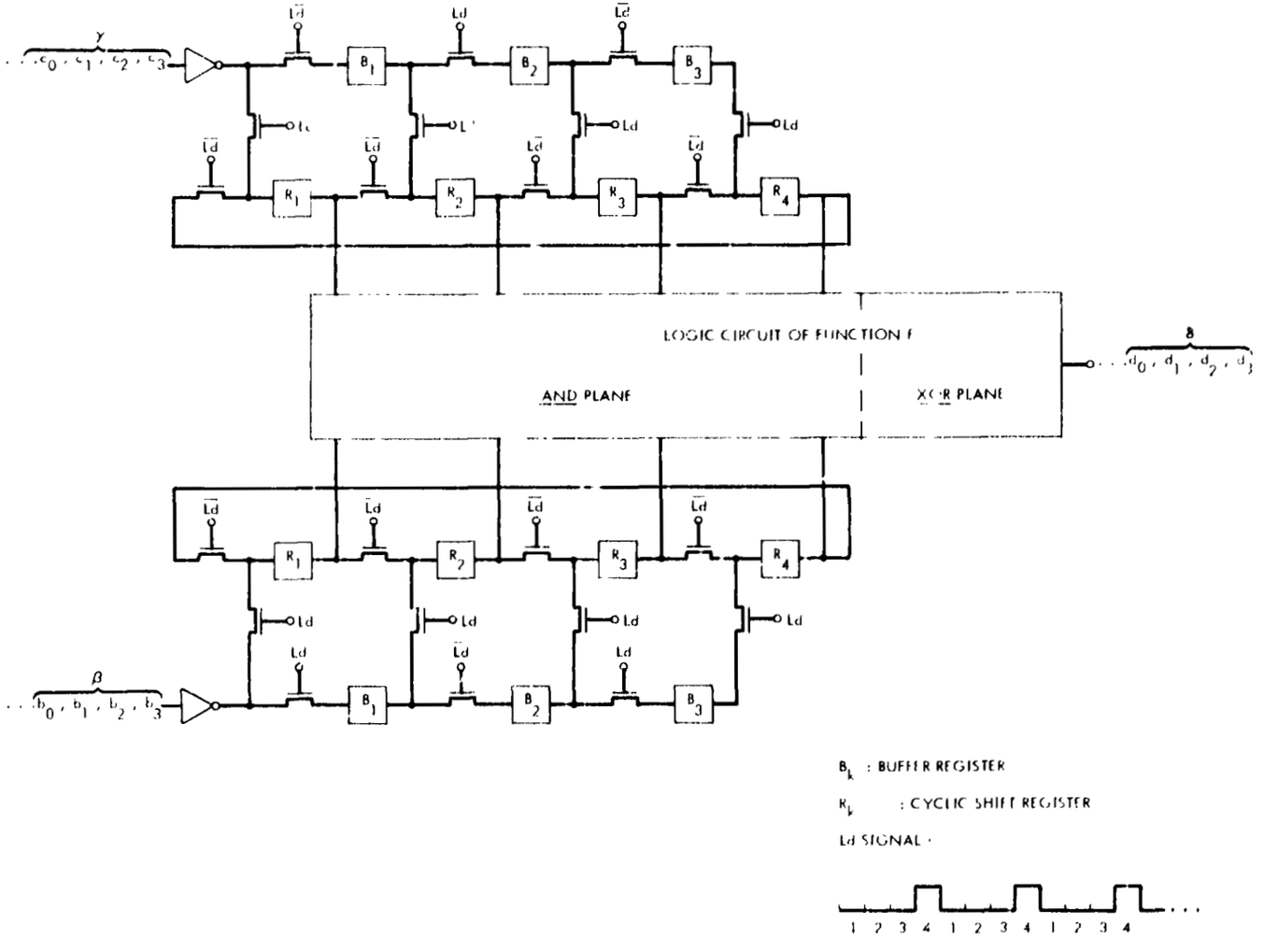

Fig. 5. A pipeline Massey-Omura multiplier for $GF(2^4)$

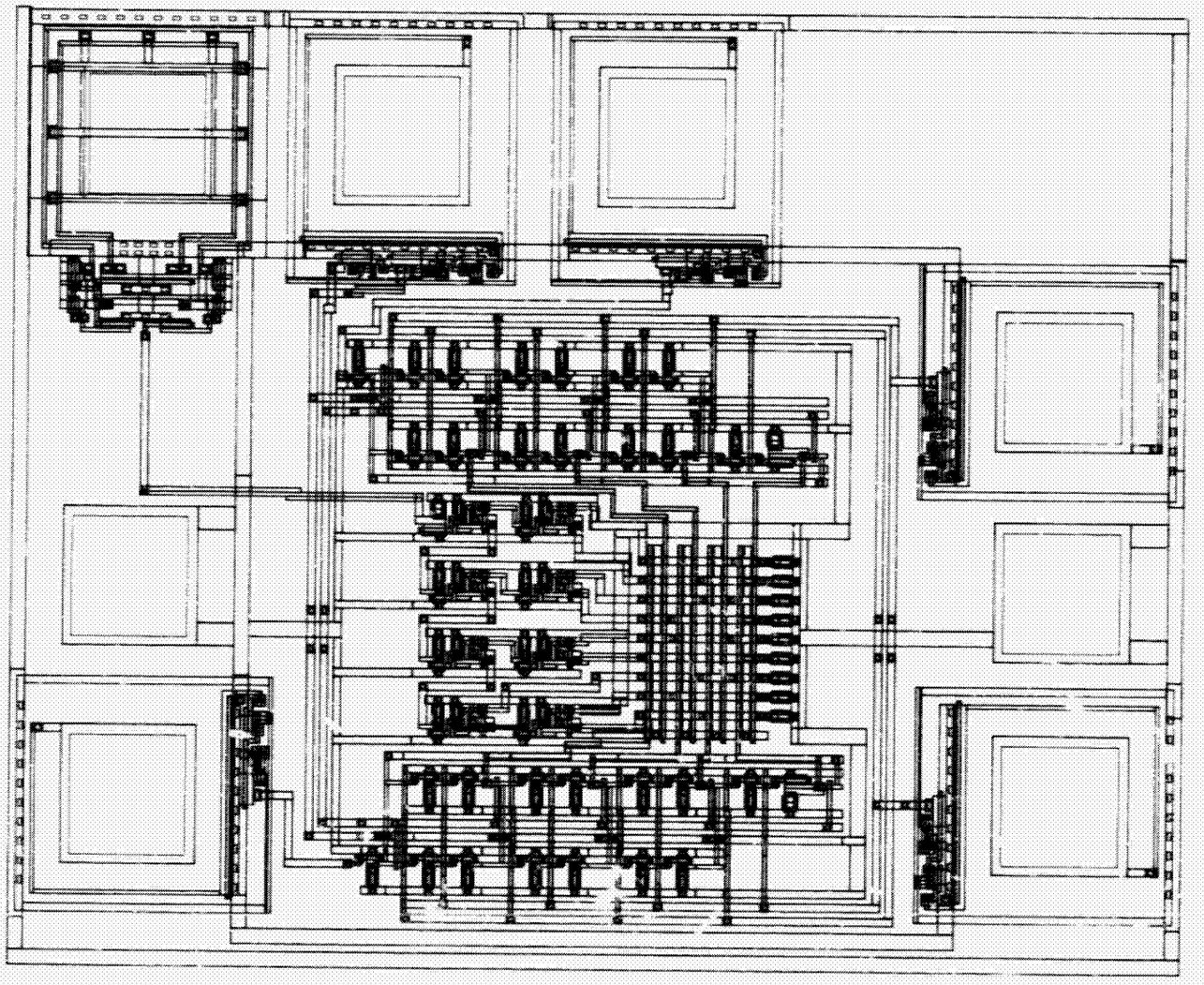

Fig. 6. Laycut of a Massey-Omura multiplier for $GF(2^4)$

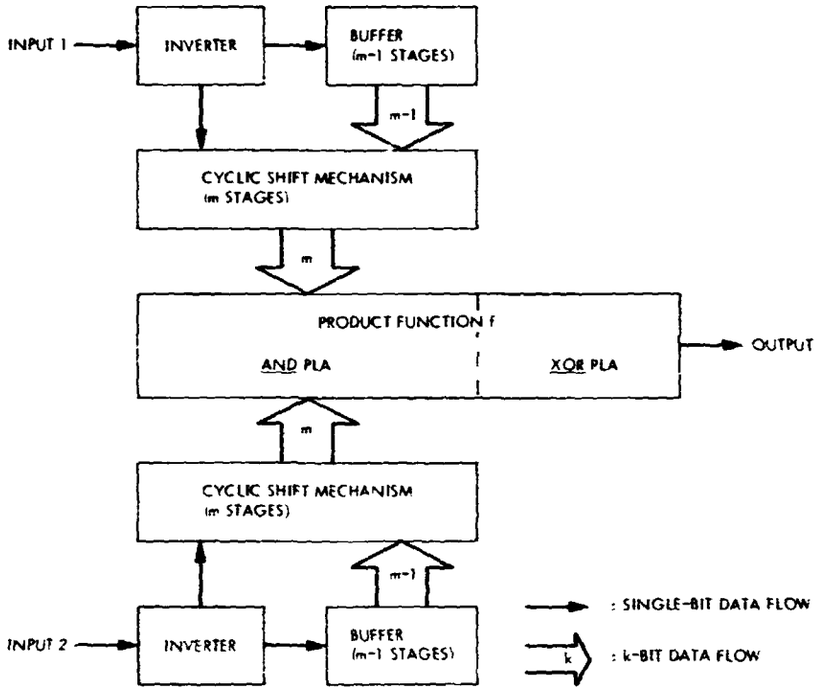

Fig. 7. System structure of a pipeline Massey-Omura multiplier for GF(2^m)

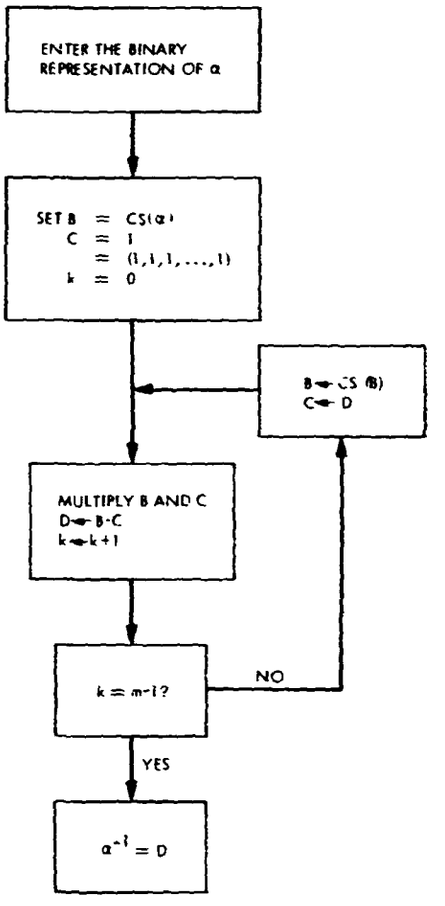

Fig. 8. Flow chart diagram of computing the inverse

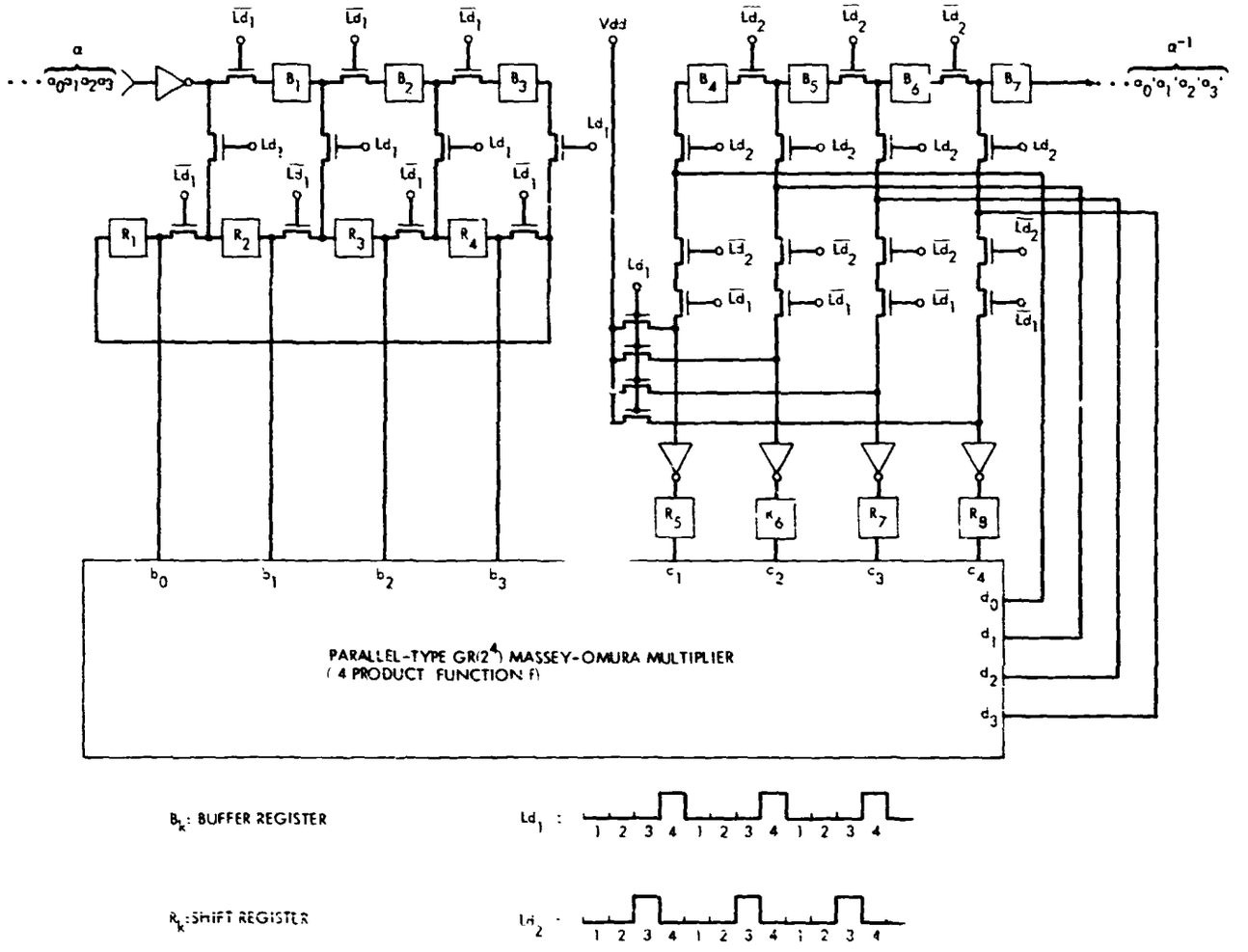

Fig. 9. Pipeline structure of computing the inverse element in GF(2⁴)

ORIGINAL PAGE IS
OF POOR QUALITY

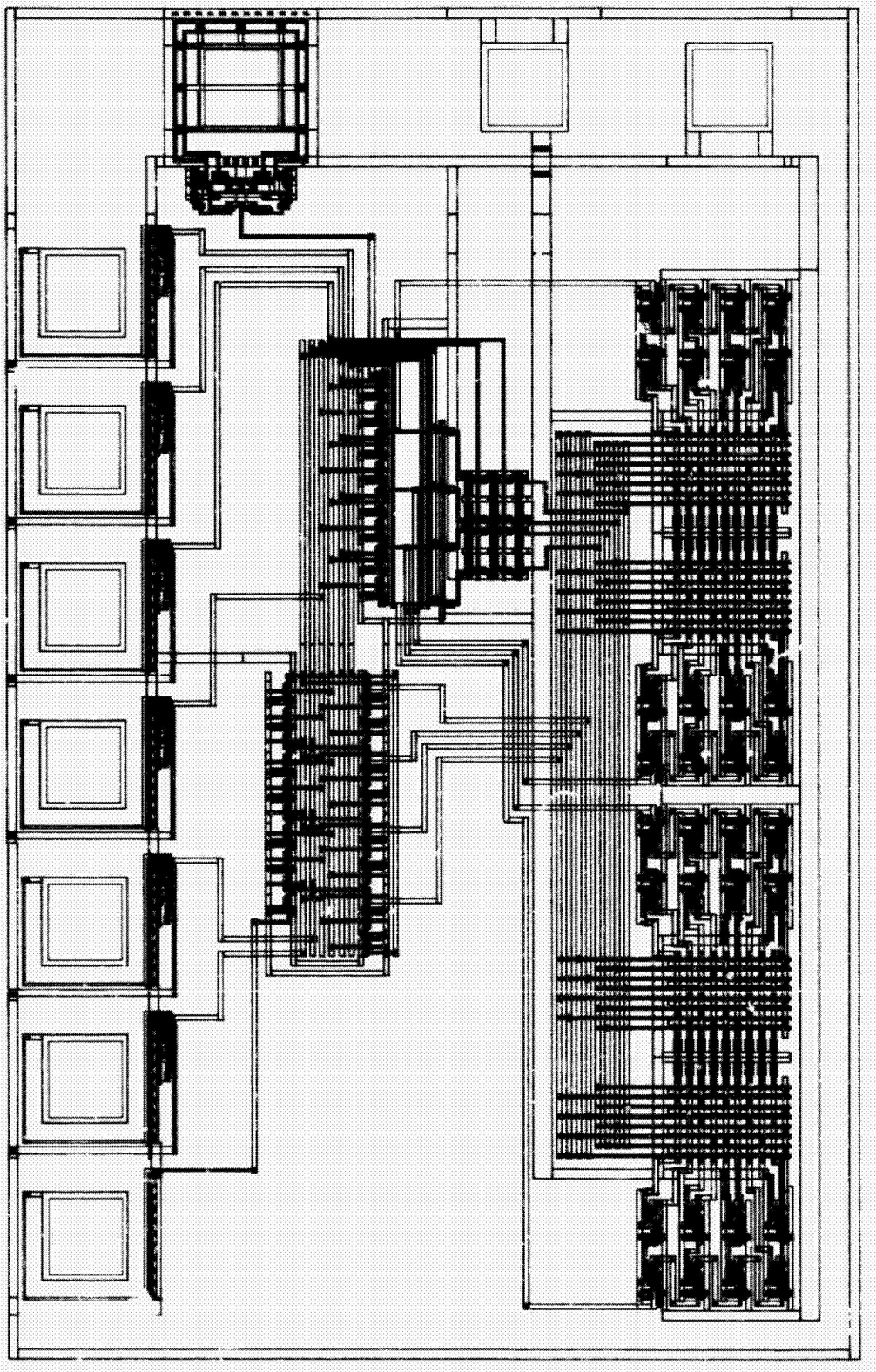

Fig. 10. Layout of the inversion circuit for $GF(2^4)$

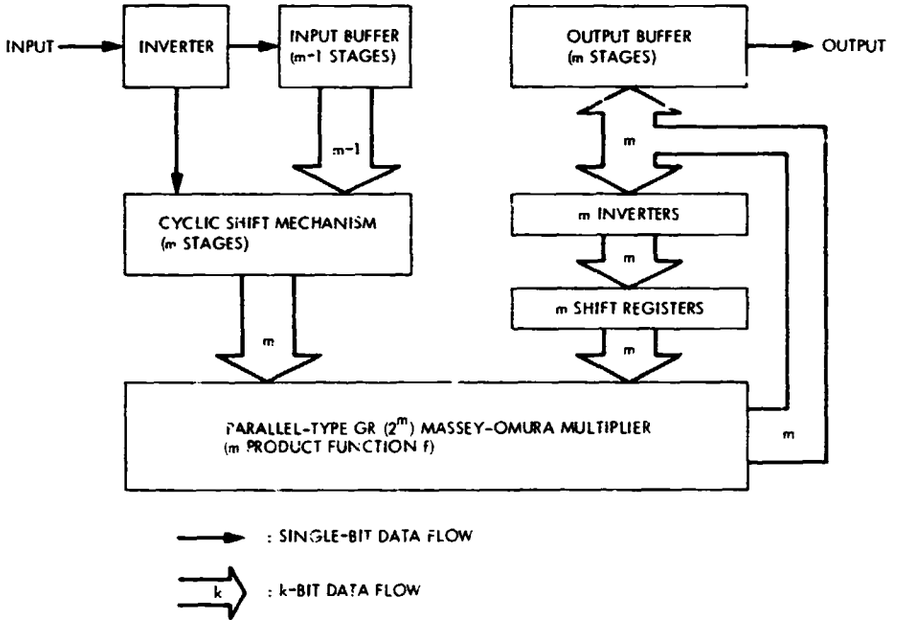

Fig. 11. System structure of a pipeline inversion circuitry for $GF(2^m)$