

Appeared IN IEEE TRANS. ON Computers
Vol 42, N9, Sept 1993

A Design of Self-Diagnostic Boards by Multiple Signature Analysis *

MARK G. KARPOVSKY, *Fellow, IEEE*, SAEED M. CHAUDHRY, *Student Member, IEEE* †

Abstract—A new design approach, based on multiple signature analysis, for self-diagnostic boards is presented. For this approach, test responses from all chips on the board are compressed into space-time signatures using nonbinary multiple error-correcting codes and faulty chips are identified by analyzing relations between distortions in these signatures. This approach results in a considerable reduction of a hardware overhead, required for diagnostic, as compared to the straightforward approach where separate signatures are computed for each chip on the board. The presented diagnostic approach can also be used for identification of faulty boards in a system or for faulty processors in a multiprocessor environment.

Key Words—Built-in self-diagnostic, design for testability, multiple signature analysis, self-diagnostic board (system), space-time compression of test responses.

I. Introduction

In this paper, we will consider a design approach for a self-diagnostic board. We assume that the original board consists of several chips with or without any provision for built-in self-test [1], [2]. The proposed design approach requires an additional diagnostic chip. This chip is responsible for controlling and execution of a diagnostic algorithm (presented below) on the board. We further assume that all the chips on the board are functionally interconnected through a common bus, and this bus is

*This work was supported by the National Science Foundation under Grant MIP-8813748 and the NATO under Grant 910411.

†The authors are with the Department of Electrical, Computer and Systems Engineering, Boston University, Boston, Massachusetts 02215.

used for transfer of data between chips. This configuration is illustrated in Fig. 1. If chips on the board are not interconnected via a common bus, an additional test bus is required for transferring data between the chips and the diagnostic chip, for example, a test and diagnostic bus. For the case of IEEE Standard 1149.1 Test Access Port (TAP) and Boundary Scan Architecture [3] chips on the board are sequentially linked to the diagnostic chip, with a four (optionally five) signal test bus, forming one long scan chain and necessary control signals for TAPs will be provided by the on board diagnostic chip.

The diagnostic chip (see Fig. 1) consists of three components: (i) a test pattern generator implemented by a linear feed-back shift register [4]–[7] to generate pseudorandom test patterns; (ii) a multiple signature analyzer; and (iii) a timing and control module. On receiving an external command, board level diagnostic is initiated by the diagnostic chip. The on-chip test pattern generator generates pseudorandom input test patterns. These test patterns are applied to all chips on the board and results are sequentially shifted into the diagnostic chip. Random pattern testing, in some cases, may take much longer time to fully exercise a board, alternatively, one may use weighted random pattern testing [8] to provide necessary test coverages. (The problem of a test length for pseudorandom tests has been extensively studied see e.g. [2], [4]–[7]. In this paper we assume that a selected length T of the pseudorandom test is sufficient to provide for a required fault coverage for all chips on the board.)

Test results from all chips are compressed into space-time signatures by a space-time compressor based on a nonbinary multiple error-correcting code. In particular, we use l -error-correcting q -ary Reed-Solomon (RS) codes [13] for space compression and single-error-detecting RS codes for time compression where l is a maximum number of faulty chips on the board and $q = 2^m$ (m is the width of the system bus). The signature analyzer module compares the obtained signatures with the predetermined reference signatures and analyze relations between distortions in signatures to find upto l faulty chip locations. The special case of this approach for $l = 1$ was presented in [15], [16]. Application of single error-correcting RS codes for ROM testing was considered in [17], [18]. Analysis of aliasing probabilities for testing by signature analysis based on single and multiple q -ary error-correcting RS codes was presented in [19]–[21].

In Section II, we consider a straightforward approach for a self-diagnostic board based on generic signature analysis. This approach requires n (n is the total number of chips on the original board) reference signatures to be stored for diagnostic. This approach has $\Theta(nm)$ hardware complexity for a board which has n chips and m -bit system bus.

Implementation of the straightforward approach does incur a high overhead and, therefore, may only be viable for boards with a small number of chips. In Section III, we present an l -faulty-chip diagnostic algorithm which will require only $2l$ reference signatures to be stored for any number of chips on the original board. The proposed approach requires $\Theta(lm)$ hardware and results in considerable savings in a required overhead as compared to the straightforward approach.

In Section IV, we present a modular VLSI design of the signature analyzer module for the proposed diagnostic chip. This design can be easily expanded to accommodate any chip-fault multiplicity l . We will also compare the space and time complexities for this design to the straightforward design and present details of a prototype diagnostic chip implementation.

II. A Straightforward Self-Diagnostic Board Design Approach

For a straightforward self-diagnostic board design, we consider an approach based on generic signature analysis. For this approach, each chip on the original board is tested separately (Fig. 2). A linear feedback shift register (LFSR) is used as an on-board test pattern generator. Pseudorandom test patterns are applied onto the primary inputs of chip i , $0 \leq i \leq n - 1$, being tested. Test responses from chip i are transferred to the multiple input signature register (MISR). When all T (test length) test responses are transferred, signature s_i is obtained. The obtained m -bit signature s_i is then compared with the corresponding precomputed reference signature s_i^0 . Any mismatch between s_i and s_i^0 is detected by the m -bit comparator.

The signature analyzer module for the straightforward approach can be implemented by a m -bit MISR, a m -bit match detector, a m -bit $n \times 1$ MUX and n m -bit reference signature registers, where n and m are the number of chips and number of

outputs per chip on the board, respectively. The space complexity L_1 , in terms of equivalent two-input gates, for the signature analyzer module for the straightforward approach is

$$L_1 = 13nm + 15m + q(m) + n \lceil \log_2 n \rceil, \quad (1)$$

where $q(m)$, $1 \leq q(m) < m$, is the number of XOR gates in the feedback network for the m -bit MISR. (A flip-flop is assumed to have a complexity of 12 two-input equivalent gates.) For example, a board with $n = 100$ chips and $m = 32$ outputs per chip requires about 42,800 equivalent two-input gates for the signature analyzer module ($q(32) = 3$).

The time complexity P_1 for the straightforward approach (number of steps in the diagnostic procedure) is

$$P_1 = (n + 1)T, \quad (2)$$

where T is the length of the test.

III. A Multiple Signature Analysis Self-Diagnostic Board Design Approach

Since an implementation of the straightforward approach incurs a high overhead (see (1)), we present an alternative approach for the board-level self-diagnostic. This approach is based on multiple signature analysis (presented below) and the major advantage over the straightforward approach is that only $2l$ reference signatures are stored for any number of chips on the original board to identify upto l chip faults.

The diagnostic procedure for this approach requires: (i) application of pseudorandom test patterns onto the primary inputs of chips; (ii) space and time compression of test responses into $2l$ signatures and computation of distortions in signatures; and (iii) location of upto l faulty chips by analyzing the relations between distortions in the obtained signatures. A block diagram for this approach is given at Fig. 3.

Consider a board consisting of n chips. We assume here that every chip has the same number of output pins equal to m . (For the case of unequal numbers of output pins one can take m as a maximum number of output pins per chip and assign additional zero components to the outputs of chips with less than m output

pins.) Let $y(t) = (y_0(t), y_1(t), \dots, y_{n-1}(t))$ be a board test response at the moment t , $0 \leq t \leq T - 1$, where T is a number of test patterns applied. A component $y_i(t)$, $0 \leq i \leq n - 1$, of $y(t)$ is a test response (m -bit symbol) from the chip number i , $0 \leq i \leq n$, due to the t th test pattern.

Step (i) of the approach requires application of T test patterns to n chips on the board. As a result, T board responses $y(0), y(1), \dots, y(T - 1)$ are obtained. These board responses can be represented by a $(T \times n)$ matrix $Y = [y_i(t)]$ where $y_i(t) \in GF(q = 2^m)$, $0 \leq i \leq n - 1$, $0 \leq t \leq T - 1$ ($GF(q = 2^m)$ denotes the Galois field of order 2^m).

Step (ii) requires a two-step (space-time) computation of $2l$ m -bit signatures $s = (s_0, s_1, \dots, s_{2l-1})$ from Y . This multiple signature analysis scheme is based on space compression techniques [9]–[11] and on nonbinary RS codes [12], [13]. Space-time compression of test responses can be considered as a decoding procedure for a concatenated code [12], where the *inner* (space compression) code is $[n, n - 2l, 2l + 1]$ RS code and the *outer* (time compression) code is $[T, T - 1, 2]$ RS code.

The first step (space compression) is to compute space signatures based on a $[n, n - 2l, 2l + 1]$ l -error-correcting RS code over $GF(q)$. The codewords $v = (v_1, v_2, \dots, v_n)$ of this code are vectors in the n -dimensional space $GF(q)^n$ over $GF(q)$, $v_i \in GF(q)$. The l -error-correcting RS code $C \subset GF(q)^n$ over $GF(q)$ with block size n and number of redundant symbols $2l$ is the null space, $C = \{v | vH_s^{tr} = 0\}$, of check matrix H_s whose transpose is

$$H_s^{tr} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ \alpha & \alpha^2 & \dots & \alpha^{2l} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{n-1} & \alpha^{2(n-1)} & \dots & \alpha^{2l(n-1)} \end{bmatrix}, \quad (3)$$

where $n \leq q - 1$, and α is a primitive in $GF(q)$ (α is a primitive element if and only if $\alpha^i \neq \alpha^j$ for $i \neq j$, $i, j = 0, 1, \dots, 2^m - 2$) [14].

Space compression of Y based on the above code is defined as

$$Z = YH_s^{tr}, \quad (4)$$

where $Z = [z_j(t)]$ and $z_j(t) = y_0(t) \oplus y_1(t)\alpha^{j+1} \oplus \dots \oplus y_{n-1}(t)\alpha^{(j+1)(n-1)}$, $0 \leq j \leq 2l - 1$, $0 \leq t \leq T - 1$. (\oplus stands for the component-wise modulo two addition.)

Computation of Z is performed by $2l$ MISRs with feedback polynomials corresponding to $\alpha, \alpha^2, \dots, \alpha^{2l}$.

The second step (time compression) is the computation of space-time signatures from the intermediate space signatures $z_0(t), z_1(t), \dots, z_{2l-1}(t)$ using $[T, T-1, 2]$ RS code over $GF(q)$. To compute space-time signatures $s = (s_0, s_1, \dots, s_{2l-1})$, time compression of Z based on $[T, T-1, 2]$ RS codes over $GF(q)$ is performed as follows:

$$s = H_t Z, \quad (5)$$

where

$$H_t = [\alpha^{T-1} \alpha^{T-2} \dots 1], \quad (6)$$

and $s_j = \alpha^{T-1} z_j(0) \oplus \alpha^{T-2} z_j(1) \oplus \dots \oplus z_j(T-1)$, $0 \leq j \leq 2l-1$. This time compression requires $2l$ identical MISRs with the feedback polynomial corresponding to α (α is the root of the feedback polynomial).

Using (4) and (5), space-time compression of a board test response Y into $2l$ signatures is

$$s = H_t Y H_s^{tr}, \quad (7)$$

where H_s and H_t are called space and time compression matrices, respectively.

For fault detection, we need to compute distortions $\delta = (\delta_0, \delta_1, \dots, \delta_{2l-1})$, in space-time signatures and verify whether δ is equal to zero or not. The distortions $\delta = (\delta_0, \delta_1, \dots, \delta_{2l-1})$ are defined as follows:

$$\delta = s \oplus s^0, \quad (8)$$

where $\delta_j = s_j \oplus s_j^0$, $0 \leq j \leq 2l-1$ and $s^0 = (s_0^0, s_1^0, \dots, s_{2l-1}^0)$ are precomputed space-time reference signatures from the fault free board response Y^0 . If $\delta = \overbrace{(0, 0, \dots, 0)}^{2l}$, we conclude that the board is fault-free and the diagnostic procedure is completed.

Example: Consider the case of a board with $n = 5$ chips, $m = 3$ outputs per chip, double chip faults ($l = 2$) and $T = 6$ test patterns. We further assume without loss of generality that $y_i(t) = y_i^0(t) \oplus e_i(t)$ and $y_i^0(t) = 0$, $0 \leq i \leq 4$, $0 \leq t \leq 5$ where $y_i^0(t)$ and $e_i(t)$ are fault free response and error from the chip i at moment t . Therefore, $s^0 = 0$ and

$$\delta = s \oplus s^0 = H_t Y H_s^{tr}. \quad (9)$$

Let us consider first the construction of the finite field $GF(2^3)$ with 2^3 elements. To construct this field we choose a primitive polynomial as $p(x) = x^3 \oplus x \oplus 1$ over $GF(2)$. The nonzero elements of the field $GF(2^3) \{\alpha^i | i = 0, 1, \dots, 6\}$, can be generated by a MISR (see Fig. 4), that has a characteristic (feedback) polynomial $p(x)$, with initial state $(d_2, d_1, d_0) = (0, 0, 1)$ and input $(z_2, z_1, z_0) = (0, 0, 0)$. These seven nonzero elements of $GF(2^3)$ (internal states of MISR) are given in Table 1. The polynomial and exponential representations in terms of the primitive element α are given by columns three and four of Table 1. The primitive element α is a root of $p(x) = x^3 \oplus x \oplus 1$, i.e., $\alpha^3 \oplus \alpha \oplus 1 = 0$.

Now consider the following board response matrix Y where each element in i th column represents $y_i(t) = e_i(t)$ ($t = 0, 1, \dots, 5$), produced by the chip number i , $0 \leq i \leq 4$:

$$Y = \begin{bmatrix} 0 & \alpha^6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha^2 & 0 \\ 0 & \alpha & 0 & 0 & 0 \\ 0 & \alpha^5 & 0 & \alpha & 0 \\ 0 & 1 & 0 & \alpha^6 & 0 \end{bmatrix}. \quad (10)$$

From this response matrix we can see that chips 1 and 3 are faulty.

Since $n = 5$ and $l = 2$, we select a check matrix H_s of $[5, 1, 5]$ double-error-correcting RS code over $GF(2^3)$ for space compression by (3) as

$$H_s^{tr} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ \alpha & \alpha^2 & \alpha^3 & \alpha^4 \\ \alpha^2 & \alpha^4 & \alpha^6 & \alpha \\ \alpha^3 & \alpha^6 & \alpha^2 & \alpha^5 \\ \alpha^4 & \alpha & \alpha^5 & \alpha^2 \end{bmatrix}. \quad (11)$$

From (4), space compression of Y (10) based on the above H_s^{tr} can be represented as

$$Z = YH_s^{tr} = \begin{bmatrix} 1 & \alpha & \alpha^2 & \alpha^3 \\ 0 & 0 & 0 & 0 \\ \alpha^5 & \alpha & \alpha^4 & 1 \\ \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 \\ \alpha^3 & 0 & 1 & 1 \\ \alpha^4 & \alpha^3 & 1 & 0 \end{bmatrix}. \quad (12)$$

Since $T = 6$, we select a check matrix H_t of $[6, 5, 2]$ single-error-detecting RS for time compression by (6) as

$$H_t = [\alpha^5 \alpha^4 \alpha^3 \alpha^2 \alpha 1], \quad (13)$$

and by (5) we compute the space-time signatures $s = (s_0, s_1, s_2, s_3)$ as

$$s = H_t Z = (\alpha^3, \alpha^5, \alpha^4, \alpha). \quad (14)$$

Since $Y^0 = 0$ and $\delta = s$ is not equal to zero (from (9)), we conclude that the board is faulty. \square

Once the presence of a fault is detected, locations of faulty chips are computed by the decoder shown in Fig. 3 which analyzes relations between distortions $\delta_0, \dots, \delta_{2l-1}$ in the observed signatures. The function of the decoder is to perform the diagnostic algorithm (presented below).

Without loss of generality, let us assume that $y_i(t) = y_i^0(t) \oplus e_i(t)$ with $y_i^0(t) = 0$, $0 \leq i \leq n-1$, $0 \leq t \leq T-1$. From (3)–(8) we have for distortions $\delta_0, \dots, \delta_{2l-1}$ in signatures:

$$\delta_j = \sum_{t=0}^{T-1} \left(\sum_{i=0}^{n-1} e_i(t) \alpha^{(j+1)i} \right) \alpha^{T-1-t} = \sum_{i=0}^{n-1} e_i \alpha^{(j+1)i} \quad (j = 0, 1, \dots, 2l-1), \quad (15)$$

where

$$e_i = \sum_{t=0}^{T-1} e_i(t) \alpha^{T-1-t} \quad (i = 0, 1, \dots, n-1), \quad (16)$$

is the time compressed error vector. Thus, distortions are syndromes of a time compressed error vector $e = (e_0, e_1, \dots, e_{n-1})$ (e_i is given by (16)) for the space compression code generated by H_s and

$$\delta = eH_s^{tr}. \quad (17)$$

Since the minimum distance of the space compression $[n, n - 2l]$ RS code is $2l + 1$, up to $2l$ chip-faults can be detected, and up to l chip-faults can be located from distortions in signatures.

To locate faulty chips, we need to solve the following system of $2l$ equations:

$$\begin{aligned}
\delta_0 &= e_{i_1} \alpha^{i_1} \oplus e_{i_2} \alpha^{i_2} \oplus \dots \oplus e_{i_l} \alpha^{i_l} \\
\delta_1 &= e_{i_1} \alpha^{2i_1} \oplus e_{i_2} \alpha^{2i_2} \oplus \dots \oplus e_{i_l} \alpha^{2i_l} \\
&\vdots \\
\delta_{2l-1} &= e_{i_1} \alpha^{2li_1} \oplus e_{i_2} \alpha^{2li_2} \oplus \dots \oplus e_{i_l} \alpha^{2li_l}
\end{aligned} \tag{18}$$

where $i_1, \dots, i_l, e_{i_1}, \dots, e_{i_l}$ are unknowns. Our interest is only to find up to l locations i_1, \dots, i_l from $2l$ distortions in the obtained signatures.

It is worth mentioning here that for the case of single-faulty-chip [15], [16], this can be accomplished by a very simple decoder. Also, for the case of double faults ($l = 2$), efficient implementations are possible due to simple analytical solutions for the above system. However, for a chip-fault multiplicity l more than two analytical solutions are very cumbersome and require a complex decoding procedure. Below we present a diagnostic algorithm, which utilizes a modified form of the Euclidean algorithm [22], [23], as a recursive procedure for calculating faulty chip locations from distortions in the obtained signatures.

Let

$$\delta(x) = \sum_{k=0}^{2l-1} \delta_k x^{2l-1-k} = \delta_0 x^{2l-1} \oplus \delta_1 x^{2l-2} \oplus \dots \oplus \delta_{2l-1}, \tag{19}$$

be the polynomial representing distortions $\delta = (\delta_0, \dots, \delta_{2l-1})$ in signatures. The diagnostic algorithm to find the faulty chip locations is a two step procedure: (i) perform the modified Euclidean algorithm (described below) on x^{2l} and $\delta(x)$, to obtain the fault locator polynomial $\lambda(x) = \lambda_0 x^l \oplus \lambda_1 x^{l-1} \oplus \dots \oplus \lambda_l$; and (ii) find the faulty chip locations $X = \{x_j | 0 \leq j \leq l\}$, where $x_j = i$ if $\lambda(\alpha^i) = 0$ for $i = 0, 1, \dots, n - 1$.

Consider the two polynomials x^{2l} and $\delta(x)$. The modified Euclidean algorithm is a recursive procedure for finding the i th remainder $r_i(x)$ and the quantities $v_i(x)$ and $t_i(x)$ that satisfy

$$v_i(x) x^{2l} \oplus t_i(x) \delta(x) = r_i(x). \tag{20}$$

As soon as the degree of the i th remainder $r_i(x)$ is less than l , the algorithm stops.

The resulting $t_i(x)$ at the termination of the algorithm is the desired fault locator polynomial $\lambda(x) = t_i(x)$.

The modified Euclidean algorithm involves four sequences of polynomials $(r_i(x))$, $(q_i(x))$, $(t_i(x))$ and $(u_i(x))$ with initial conditions as

$$\begin{aligned} r_0(x) &= x^{2l}, & q_0(x) &= \delta(x), \\ t_0(x) &= 0, & u_0(x) &= 1. \end{aligned} \quad (21)$$

The rules to compute $r_{i+1}(x)$, $q_{i+1}(x)$, $t_{i+1}(x)$, and $u_{i+1}(x)$ are

$$\begin{aligned} r_{i+1}(x) &= [\sigma_i q_i r_i(x) \oplus \bar{\sigma}_i r_i q_i(x)] \oplus x^{|d_i|} [\sigma_i r_i q_i(x) \oplus \bar{\sigma}_i q_i r_i(x)], \\ q_{i+1}(x) &= \sigma_i q_i(x) \oplus \bar{\sigma}_i r_i(x), \\ t_{i+1}(x) &= [\sigma_i q_i t_i(x) \oplus \bar{\sigma}_i r_i u_i(x)] \oplus x^{|d_i|} [\sigma_i r_i u_i(x) \oplus \bar{\sigma}_i q_i t_i(x)], \\ u_{i+1}(x) &= \sigma_i u_i(x) \oplus \bar{\sigma}_i t_i(x), \end{aligned} \quad (22)$$

where r_i and q_i are the highest nonzero coefficients of $r_i(x)$ and $q_i(x)$, respectively, $d_i = \deg(r_i(x)) - \deg(q_i(x))$, $\sigma_i = 1$ if $d_i \geq 0$ and $\sigma_i = 0$ if $d_i < 0$ and $\bar{\sigma}_i$ is the negation of σ_i . The modified Euclidean algorithm stops as soon as the $\deg(r_i(x))$ is less than l . (Note that $u_i(x)$ is not used in the decoding algorithm.)

For step (ii), the simplest way to find the roots of $\lambda(x)$ which correspond to faulty chip locations, is by trial and error (Chien search [13], [24]). To check whether chip i , $0 \leq i \leq n-1$, is faulty or not, we simply compute $\lambda(\alpha^i)$ and check for zero. A block diagram for the above diagnostic algorithm for identification of upto l faulty chips is given in Fig. 5.

Example continued: Step (i): modified Euclidean algorithm is used to compute $\lambda(x)$. The polynomial $\delta(x)$ based on the previously computed (see (14)) distortions $\delta = (\alpha^3, \alpha^5, \alpha^4, \alpha)$ is

$$\delta(x) = \sum_{k=0}^3 \delta_k x^{3-k} = \alpha^3 x^3 \oplus \alpha^5 x^2 \oplus \alpha^4 x \oplus \alpha. \quad (23)$$

Since $\deg(r_0(x) = x^4) = 4$ and $\deg(q_0(x) = \delta(x)) = 3$, for $i = 0$ we have $d_0 = 4 - 3 = 1$ and, therefore $\sigma_0 = 1$. Hence by (22)

$$\begin{aligned} r_1(x) &= \alpha^3 r_0(x) \oplus x q_0(x), & q_1(x) &= q_0(x), \\ t_1(x) &= \alpha^3 t_0(x) \oplus x u_0(x), & u_1(x) &= u_0(x), \end{aligned}$$

where

$$\begin{aligned} r_0(x) &= x^4, & q_0(x) &= \delta(x) = \alpha^3 x^3 \oplus \alpha^5 x^2 \oplus \alpha^4 x \oplus \alpha, \\ t_0(x) &= 0, & u_0(x) &= 1. \end{aligned}$$

Thus,

$$\begin{aligned} r_1(x) &= \alpha^5 x^3 \oplus \alpha^4 x^2 \oplus \alpha x, & q_1(x) &= \alpha^3 x^3 \oplus \alpha^5 x^2 \oplus \alpha^4 x \oplus \alpha, \\ t_1(x) &= x, & u_1(x) &= 1. \end{aligned}$$

Computations required to construct fault locator polynomial $\lambda(x)$ for this example are presented in Table 2. The modified Euclidean algorithm terminates after step 4 because the degree of $r_4(x)$ is one, which is less than $l = 2$. At this moment, the computation for $\lambda(x) = \lambda_0 x^2 \oplus \lambda_1 x \oplus \lambda_2$ is complete, and the fault locator polynomial for this example is

$$\lambda(x) = t_4(x) = x^2 \oplus x \oplus \alpha^4. \quad (24)$$

Step (ii): Chien search is used to compute locations of faulty chips. Computations of Chien search are presented in Table 3. It is evident from the 5th column of Table 3 that α and α^3 are the roots of the fault locator polynomial $\lambda(x) = x^2 \oplus x \oplus \alpha^4$. Hence, we conclude, that chips 1 and 3 are faulty. \square

The above proposed approach requires only $\Theta(lm)$ overhead and this overhead does not depend on a number of chips in the board. For large m (say, $m \geq 32$), an additional field compression from $GF(2^m)$ into $GF(2^b)$, where $b < m$, will result in a reduction of an overhead to $\Theta(lb)$. A design of optimal $GF(2^m) \rightarrow GF(2^b)$ field compressor is presented below.

Consider an optimal binary $[m, m - b, l_b + 1]$ linear error detecting code of length m with b redundant bits. The parity check matrix H_b of this code is $H_b = [h_{p,q}]$, where $h_{p,q} \in GF(2)$, $0 \leq p \leq b - 1$, $0 \leq q \leq m - 1$. Let us define a mapping $H_f : GF(2^m)^n \rightarrow GF(2^b)^n$ as follows:

$$H_f = \begin{bmatrix} H_b & & & 0 \\ & H_b & & \\ & & \ddots & \\ 0 & & & H_b \end{bmatrix}. \quad (25)$$

Field compression of Y based on H_f is defined as follows:

$$X = H_f Y, \quad (26)$$

where $X = [x_i(t)]$, $x_i(t) = H_b y_i(t) \in GF(2^b)$, $0 \leq i \leq n - 1$, $0 \leq t \leq T - 1$, and all multiplications and additions are performed in $GF(2)$. A field compressor based on above mapping is shown in Fig. 6. Computation of X (26) is implemented by b parity trees corresponding to b rows of check matrix H_b .

The effectiveness of a field compressor largely depends on its error propagating capability and actual errors caused by faults in chips. All faults in chips resulting in distortions of at most l_b bits at any given moment of time at the output of every chip will not be masked in the field compressor. The fraction of errors with a multiplicity larger than l_b which are not propagated (masked) by the field compressor is $(2^{m-b} - 1)/(2^m - 1) \simeq 2^{-b}$. Therefore, desired fault coverages can be achieved by selection of the corresponding field compression $[m, m - b]$ code.

IV. A VLSI Implementation of Diagnostic Chip

In this section we present a VLSI implementation for the above proposed diagnostic chip. We will also compare the space and time complexities of this implementation with implementations based on the straightforward approach presented in Section II.

For the approach presented in Section III a VLSI architecture for the signature analyzer module of the diagnostic chip consists of three blocks (see Fig 3): (i) data compressor block; (ii) reference comparator block; and (iii) fault-locator block. The VLSI logic structures associated with each of these block are described below.

The data compressor block computes $2l$ space-time signatures from board test responses using equation (7). This block (see Fig. 7) consists of $4l$ m -bit MISRs. $MISR_1, \dots, MISR_{2l}$ perform space compression of test responses $y_0(t), \dots, y_{n-1}(t)$, $i = 0, 1, \dots, n-1$, into space signatures $z_0(t), \dots, z_{2l-1}(t)$ (see (3), (4)). $MISR_{2l+1}, \dots, MISR_{4l}$ perform time compression of these intermediate space signatures $z_0(t), \dots, z_{2l-1}(t)$, $t = 0, 1, \dots, T - 1$ into the space-time signatures s_0, \dots, s_{2l-1} (see (5), (6)).

$MISR_k$ ($k = 1, \dots, 2l$) have its feedback connections setup to multiply its contents by α^k and add its input in $GF(q)$, $q = 2^m$. A block diagram for such a MISR is shown in Fig. 8. Test responses $y_i(t)$ ($i = n - 1, n - 2, \dots, 0$) from n chips at a moment t , $0 \leq t \leq T - 1$, are clocked into the $MISR_1, \dots, MISR_{2l}$; the contents of the $MISR_1, \dots, MISR_{2l}$ are $z_0(t), \dots, z_{2l-1}(t)$, respectively. (Note that the $MISR_1, \dots,$

MISR_{2l} are cleared prior to application of $y_i(t)$, $i = n - 1, n - 2, \dots, 0$.) Thus, after n transitions, n m -bit test responses $y_0(t), \dots, y_{n-1}(t)$ are compressed into $2l$ space compressed responses $z_0(t), \dots, z_{2l-1}(t)$ as

$$\begin{aligned}
z_0(t) &= (\dots(y_{n-1}(t)\alpha \oplus y_{n-2}(t))\alpha \oplus y_{n-3}(t)\dots)\alpha \oplus y_0(t) = \sum_{i=0}^{n-1} y_i(t)\alpha^i, \\
z_1(t) &= (\dots(y_{n-1}(t)\alpha^2 \oplus y_{n-2}(t))\alpha^2 \oplus y_{n-3}(t)\dots)\alpha^2 \oplus y_0(t) = \sum_{i=0}^{n-1} y_i(t)\alpha^{2i}, \\
&\vdots \\
z_{2l-1}(t) &= (\dots(y_{n-1}(t)\alpha^{2l} \oplus y_{n-2}(t))\alpha^{2l} \oplus y_{n-3}(t)\dots)\alpha^{2l} \oplus y_0(t) = \sum_{i=0}^{n-1} y_i(t)\alpha^{2li}.
\end{aligned} \tag{27}$$

At this moment, the contents of MISR₁, ..., MISR_{2l} are transferred to MISR_{2l+1}, ..., MISR_{4l}. Next, MISR₁, ..., MISR_{2l} are cleared, and the process of obtaining $z_0(t+1), \dots, z_{2l-1}(t+1)$ starts.

MISR_k ($k = 2l + 1, \dots, 4l$) have its feedback taps corresponding to the primitive polynomial $p(x) = x^m \oplus p_{m-1}x^{m-1} \oplus \dots \oplus 1$ where $p_j \in \{0, 1\}$ and $p(\alpha) = 0$ (MISR_k ($k = 2l + 1, \dots, 4l$) multiply its contents by α and add its input in $GF(q)$). A block diagram of MISR_k ($k = 2l + 1, \dots, 4l$) is given at Fig. 9. Space-time signatures $s = (s_0, s_2, \dots, s_{2l-1})$ are obtained by MISR_{2l+1}, ..., MISR_{4l}, respectively, as

$$\begin{aligned}
s_0 &= (\dots(z_0(0)\alpha \oplus z_0(1))\alpha \oplus z_0(2)\dots)\alpha \oplus z_0(T-1) = \sum_{t=0}^{T-1} z_0(t)\alpha^{T-1-t}, \\
s_1 &= (\dots(z_1(0)\alpha \oplus z_1(1))\alpha \oplus z_1(2)\dots)\alpha \oplus z_1(T-1) = \sum_{t=0}^{T-1} z_1(t)\alpha^{T-1-t}, \\
&\vdots \\
s_{2l-1} &= (\dots(z_{2l-1}(0)\alpha \oplus z_{2l-1}(1))\alpha \oplus z_{2l-1}(2)\dots)\alpha \oplus z_{2l-1}(T-1) = \sum_{t=0}^{T-1} z_{2l-1}(t)\alpha^{T-1-t}.
\end{aligned} \tag{28}$$

The reference comparator block (see Fig. 10) compares the obtained signatures s_0, \dots, s_{2l-1} with the prestored reference signatures s_0^0, \dots, s_{2l-1}^0 , in registers $\text{regs}_0^0, \dots, \text{regs}_{2l-1}^0$ respectively, to detect any fault in the board and it also computes distortions $\delta_0, \dots, \delta_{2l-1}$ in the signatures. This block is implemented by $2lm$ two-input XOR gates, $2l$ m -bit registers, and a $2lm$ -input OR gate. The outputs of the reference signature comparator are component-wise exclusive-ORs between obtained signatures s_0, \dots, s_{2l-1} and references s_0^0, \dots, s_{2l-1}^0 . If the distortions $\delta \neq \overbrace{(0, 0, \dots, 0)}^{2l}$, the "Fault

Detected" signal is equal to one and the fault-locator block is signaled to execute the diagnostic algorithm described in Section III.

The diagnostic algorithm is implemented by the fault-locator. The outputs of the fault-locator are faulty-chip locations. A VLSI implementation for the fault-locator block requires one Euclidean cell implementing (22), $6l + 2$ m -bit shift registers, l m -bit MISRs and a modulo n counter. A block diagram for the fault-locator block is given in Fig. 11. Coefficient registers r and q are $2lm$ -bits long, whereas registers t and u are $(l + 1)m$ -bits long.

The function of the fault-locator can be described as follows. First, as soon as the "Fault Detected" signal is equal to one, δ is loaded into the register q and registers r , t and u are initialized according to the initial conditions given in (21), then the contents of registers $r_i(x)$, $q_i(x)$, $t_i(x)$, and $u_i(x)$ are shifted into the Euclidean cell. The Euclidean cell computes the quantities $r_{i+1}(x)$, $q_{i+1}(x)$, $t_{i+1}(x)$, and $u_{i+1}(x)$ based on (22) and shifts them back into the corresponding registers. As soon as the Euclidean cell detects that the $\deg(r_{i+1}(x))$ is less than l it indicates the completion of the algorithm by making the "Stop" signal equal to one. At this moment, the contents of the register t are coefficients of the fault locator polynomial. In the worst case, the computation of a fault locator polynomial requires $2l$ steps or $4l^2m$ clocks.

As soon as the "Stop" signal is equal to one, contents of the coefficient-register t are clocked in MISR $_{4l+1}$, ..., MISR $_{5l}$. MISR $_{4l+1}$, ..., MISR $_{5l}$ have their feedback set up to multiply by $\alpha^l, \alpha^{l-1}, \dots, \alpha$, respectively. By clocking MISR $_{4l+1}$, ..., MISR $_{5l}$, we substitute $\alpha^0, \alpha, \dots, \alpha^{n-1}$ for x , into $\lambda(x)$, to find its roots.

In order to test α^0 , component-wise modulo two sum of the contents of MISR $_{4l+1}$, ..., MISR $_{5l}$ (see Fig. 11), which are the coefficients $\lambda_0, \dots, \lambda_{l-1}$, respectively, is compared with the coefficient λ_l . If the result is zero, then α^0 is a root of $\lambda(x)$ and chip 0 is faulty; otherwise it is not. Now, to test α , first, each coefficient λ_j , $0 \leq j \leq l-1$, is multiplied by α^{l-j} . This is accomplished by clocking MISR $_{4l+1}$, ..., MISR $_{5l}$. These new set of coefficients, $\lambda_0\alpha^l, \dots, \lambda_{l-1}\alpha$ (contents of MISR $_{4l+1}$, ..., MISR $_{5l}$ after one transition, respectively) are summed and if their result is equal to λ_l , then α is a root of $\lambda(x)$ and chip 1 is faulty.

In general, α^i is a root and chip i is faulty if and only if $\sum_{j=0}^{l-1} \lambda_j \alpha^{i(l-j)}$ is equal to λ_l . Thus, at each step, i ($i = 0, 1, \dots, n-1$) we only multiply the coefficients $\lambda_j \alpha^{i(l-j)}$,

$0 \leq j \leq l-1$, by α^{l-j} to obtain the next set of coefficients and this is accomplished by clocking MISR_{4*l*+1}, \dots, MISR_{5*l*}. Thus, in n clocks, faulty chip locations X are obtained from the modulo n counter. (Note that the clocking of the MISR_{4*l*+1}, \dots, MISR_{5*l*} and of the counter is implemented the same clock.)

Example continued: Let us now consider an implementation for the Example given in Section III. For this example, we require eight 3-bit MISRs for the data compressor block in order to compute four space-time signatures ($l = 2$). An implementation for the comparator block requires four 3-bit registers to store reference signatures, 12 2-input XOR gates for computing distortions δ , and a 12-input OR gate for fault detection. MISR_{1}, \dots, MISR_4} are required to multiply by α, \dots, α^4 , respectively. (A block diagram for a 3-bit MISR to multiply by α^2 is given in Fig. 12.)

In Table 4, we show the state transitions of MISR_{1}, \dots, MISR_4} to obtain the first row of Z (see (12)) from Y (see (10)) and in Table 5, computations of space-time signatures (see (14)) by MISR_{5}, \dots, MISR_8} are presented.

An implementation of the fault-locator block for this example requires 14 3-bit cyclic shift registers for four coefficient registers r, q, t , and u , and one Euclidean cell to compute error-locator polynomial $\lambda(x)$. Computation of faulty chip addresses X requires two 3-bit MISRs and a 3-bit modulo five counter. \square

In comparing the hardware complexities of the diagnostic chip for the straightforward approach and for the space-time diagnostic approach, we will estimate the complexities of the signature analyzer module only, since, the complexities of the test generation and control modules for both approaches have approximately the same order and in most cases much smaller than the complexities of signature analyzers.

The space-time diagnostic approach requires $5l+1$ MISRs, $l+1$ m -bit cyclic registers, one Euclidean cell, and one modulo n counter (hardware required for computing space-time signatures and for determining the roots of a fault locator polynomial is shared). The space complexity L_2 for the l -faulty-chip approach in terms of equivalent two-input gates, is

$$L_2 = (119l + 202)m + (5q(m) + 20)l + 102\lceil \log_2 l \rceil + 13\lceil \log_2 n \rceil + 4q(m) + 138, \quad (29)$$

where $q(m)$, $1 \leq q(m) < m$, is the number of XOR gates in the feedback network for

a m -bit MISR and from (1) and (29) for $l/m \rightarrow 0$, $\log_2 n/m \rightarrow 0$, we have

$$\frac{L_1}{L_2} \simeq 0.11 \frac{n}{l}. \quad (30)$$

The total time complexity P_2 for space-time approach is

$$P_2 = (n + 1)T + 4l^2m, \quad (31)$$

and comparing (2) and (29)

$$\frac{P_1}{P_2} \simeq 1 - \frac{4l^2m}{(n + 1)T}. \quad (32)$$

In Table 6, we show a comparison of the space complexities for the straightforward approach and the proposed space-time approach for the case of a board with 32-bit system bus and a number of chips $20 \leq n \leq 200$, and for the chip fault multiplicity $l = 1, 2, \dots, 5$ ($q(32) = 3$). Since, the number of signatures required for the space-time approach is always $2l$, the advantage L_1/L_2 , of this approach over the straightforward approach is more apparent with an increasing n . In Fig. 13, we show the percentage savings $(1 - L_2/L_1) \times 100$ in hardware for boards with a 32-bit bus for the case of chip-fault multiplicity $1 \leq l \leq 5$, by using the proposed space-time diagnostic approach. One can see, for example, that for a 32-bit system bus board with 100 chips and for the 3-faulty chip diagnostic, the proposed approach offers about 57% savings of hardware.

Another alternative that can be used to implement the proposed space-time diagnostic algorithm is an assembly language program utilizing as efficiently as possible the internal microprocessor architecture [25], [26]. The execution of the proposed diagnostic algorithm on a microprocessor, assuming a m -bit processor and one clock cycle to multiply by α in $GF(2^m)$, will require additional $\Theta(l)$ time overhead as compared to the above application-specific VLSI implementation. Since, number of test patterns can be quite large for random testing this alternative may not be viable for boards with a large number of chips.

An application specific integrated circuit (ASIC) for the above VLSI implementation has been simulated at the gate level. The simulated diagnostic chip has a capability to handle 255 chips on board with a 32-bit bus architecture and can locate upto three faulty chips. A field compressor based on [32, 16] Reed-Muller code [13]

is used to compress chip responses from 32-bits to 16-bits. Some important characteristics of this prototype diagnostic chip, including performance and gate counts are summarized in Table 7.

Conclusions

In this paper we have proposed a board-level self-diagnostic technique by multiple signature analysis. This technique is based on space-time compression of test responses by nonbinary multiple error-correcting Reed-Solomon codes. Multiple faulty chips can be located by an analysis of distortions in the obtained signatures. As described in this paper, this technique requires only $2l$ (where l is the chip fault multiplicity) reference signatures for any number of chips on the original board and all chip faults with a multiplicity up to l will be correctly identified. For the case of a board with system bus architecture a modular VLSI design of the diagnostic chip is presented. The design of this diagnostic chip depends on three parameters, m —the number of output lines per chip, n —the number of chips on the original board, and l —the required chip-fault multiplicity. The proposed approach can be expanded to accommodate higher levels of integrations. We also note that the proposed approach is also capable of supporting a variety of board (systems) without requiring special customization.

Acknowledgements

The authors wish to thank Prof. Lev B. Levitin of the Boston University for many helpful discussions during the course of this research and the referees for their valuable comments.

References

- [1] H. T. Nagle, S. C. Roy, C. F. Hawkins, M. G. Macnamer, and R. R. Fritzscheier, "Design for Testability and Built-In Self Test: A Review," *IEEE Trans. on Industrial Electronics*, vol. 36, pp. 129–140, May 1989.

- [2] P. H. Bardell, W. H. McAnney and J. Savir, *Built-in Self Test for VLSI: Pseudorandom Techniques*, New York: Wiley Interscience, 1987.
- [3] IEEE Standard 1149.1-1990, *IEEE Standard Test Access Port and Boundary Scan Architecture*, New York: IEEE Standards Board, 1990.
- [4] J. Savir, G. S. Ditlow, and P. H. Bardell, "Random Patterns Testability," *IEEE Trans. on Computers*, vol. C-33, pp. 79-90, January 1984.
- [5] P. H. Bardell, W. H. McAnney, "Self Testing of Multichip logic Modules," *Proceedings International Test Conference*, pp. 200-204, 1982.
- [6] T. W. Williams, "Test-Length in a Self-Testing Environment," *IEEE Design and Test of Computers*, pp. 58-63, April 1985.
- [7] S. W. Golomb, *Shift Register Sequences*, Laguna Hills, California: Aegean Park Press, 1982.
- [8] J. A. Waicnkauski and E. Lindbloom, "Fault Detection Effectiveness of Weighted Random Patterns," *Proceedings International Test Conference*, pp. 245-255, 1988.
- [9] K. K. Saluja and M. G. Karpovsky, "Testing Computer Hardware Through Data Compression in Space and Time," *Proceedings International Test Conference*, pp. 83-88, 1983.
- [10] S. R. Reddy, K. K. Saluja and M. G. Karpovsky, "A Data Compression Technique for Built-in Self Test," *Proceedings Fault-Tolerant Computing Symposium*, pp. 294-299, 1985.
- [11] S. R. Reddy, K. K. Saluja and M. G. Karpovsky, "A Data Compression Technique for Test Responses," *IEEE Trans. on Computers*, vol. C-38, pp. 1151-1156, September 1988.
- [12] W. W. Peterson and E. J. Weldon, Jr., *Error-Correcting Codes*, Second Edition, Cambridge, Massachusetts: The MIT Press, 1972.
- [13] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, New York: North-Holland, 1977.

- [14] R. Lidl and H. Niederreiter, *Finite Fields, Encyclopedia of Mathematics*, vol. 20, Reading, Massachusetts: Addison-Wesley, 1983.
- [15] M. G. Karpovsky, "An Approach for Error Detection and Error Correction in Distributed Systems Computing Numerical Functions," *IEEE Trans. on Computers*, vol. C-30, pp. 947-953, December 1981.
- [16] M. G. Karpovsky and P. Nagvajara, "Design of Self-Diagnostic Boards by Signature Analysis," *IEEE Trans. on Industrial Electronics*, vol. 36, pp. 241-245, May 1989.
- [17] P. Nagvajara and M. G. Karpovsky, "Built-In Self-Diagnostic Read-Only-Memories," *Proceedings International Test Conference*, pp. 695-703, 1991.
- [18] K. Iwasaki, T. Fujiwara and T. Kasami, "A Defect-Tolerant Design for Mask ROMs," to appear in, *Proceedings IEEE VLSI Test Symposium*, 1992.
- [19] D. K. Pradhan, S. K. Gupta and M. G. Karpovsky, "Aliasing Probability for Multiple Input Signature Analyzer," *IEEE Trans. on Computers*, vol. 39, pp. 586-591, April 1990.
- [20] M. G. Karpovsky, S. K. Gupta and D. K. Pradhan, "Aliasing and Diagnosis Probability in MISR and STUMPS Using General Error Model," *Proceedings International Test Conference*, pp. 828-839, 1991.
- [21] M. G. Karpovsky and S. M. Chaudhry, "Multiple Signature Analysis: A Framework for Built-In Self-Diagnostic," *Proceedings Fault-Tolerant Computing Symposium*, pp. 112-119, 1992.
- [22] R. J. McEliece, *The Theory of Information and Coding, Encyclopedia of Mathematics and its Applications*, vol. 3, Reading, Massachusetts: Addison-Wesley, 1977.
- [23] H. M. Shao, T. K. Truong, L. J. Deutsch, J. H. Yuen and I. S. Reed, "A VLSI Design of a Pipeline Reed-Solomon Decoder", *IEEE Trans. on Computers*, vol. C-34., pp. 393-403, May 1985.

- [24] R. T. Chien, "Cyclic Decoding Procedures for Bose-Chaudhri-Hocquenghem Codes," *IEEE Trans. Information Theory*, vol. IT-10, pp. 357-334, October 1964.
- [25] R. L. Miller, T. K. Truong and I. S. Reed, "Efficient Program for Decoding the (255, 223) Reed-Solomon Code Over $GF(2^8)$ with Both Errors and Erasures, Using Transform Decoding," *IEEE Proceedings*, vol. 127, July 1980.
- [26] F. J. Garcia-Ugalde, "Coding and Decoding Algorithms of Reed-Solomon Codes Executed On a M68000 Microprocessor," *Proceedings 2nd International Colloquium*, Cachan-Paris, France, November 1986.

Figure 1: General Configuration of Self-Diagnostic Board.

Figure 2: Straightforward Approach–Signature Analyzer Module.

Figure 3: Multiple Signature Analysis Approach–Signature Analyzer Module.

Figure 4: Block Diagram of a 3-bit MISR–Example.

Figure 5: l Faulty-Chip Diagnostic Algorithm.

Figure 6: Field Compressor for a Space-Time Diagnostic Chip Design.

Figure 7: Logic Structure of Data Compressor Block.

Figure 8: Block Diagram of a m -bit MISR _{k} ($k = 1, \dots, 2l$).

Figure 9: Block Diagram of a m -bit MISR _{k} ($k = 2l + 1, \dots, 4l$).

Figure 10: Logic Structure of Reference Comparator Block.

Figure 11: Logic Structure of Fault-Locator Block.

Figure 12: Block Diagram of 3-bit MISR to Multiply by α^2 –Example.

Figure 13: Percentage Savings in Overhead for l -faulty-chip Diagnostic Over Straightforward Approach for 32-bit System Bus.

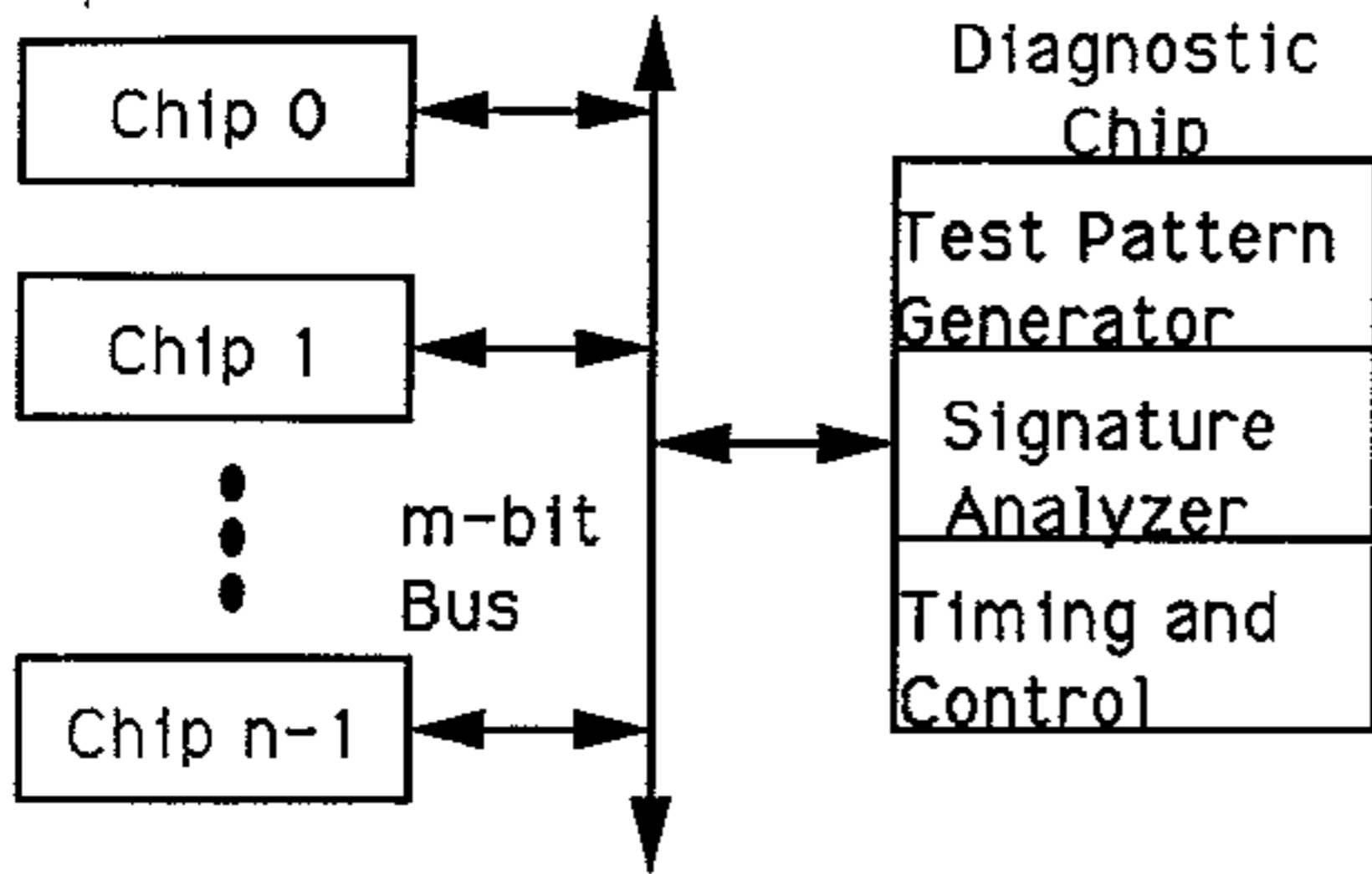


Figure 1

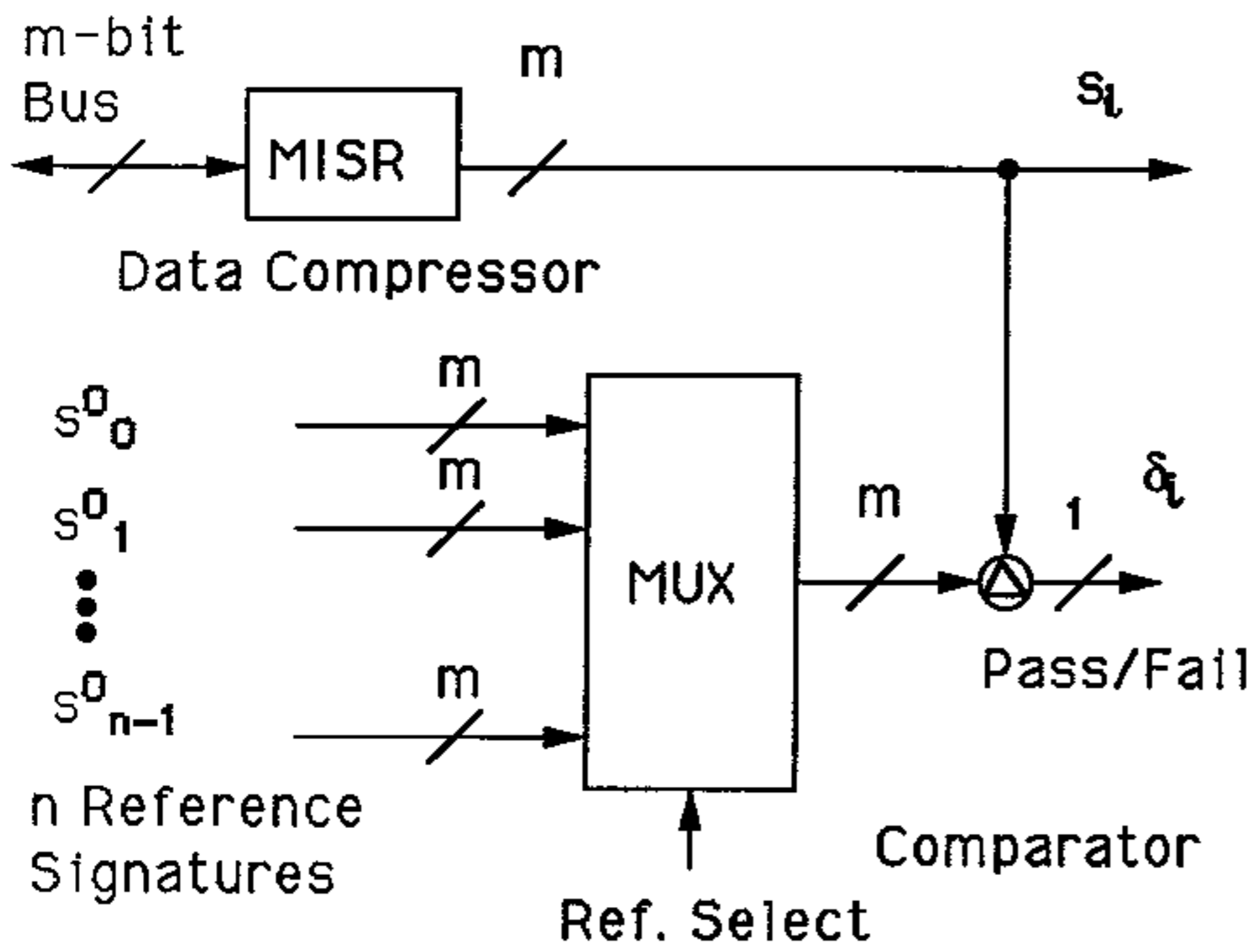


Figure 2

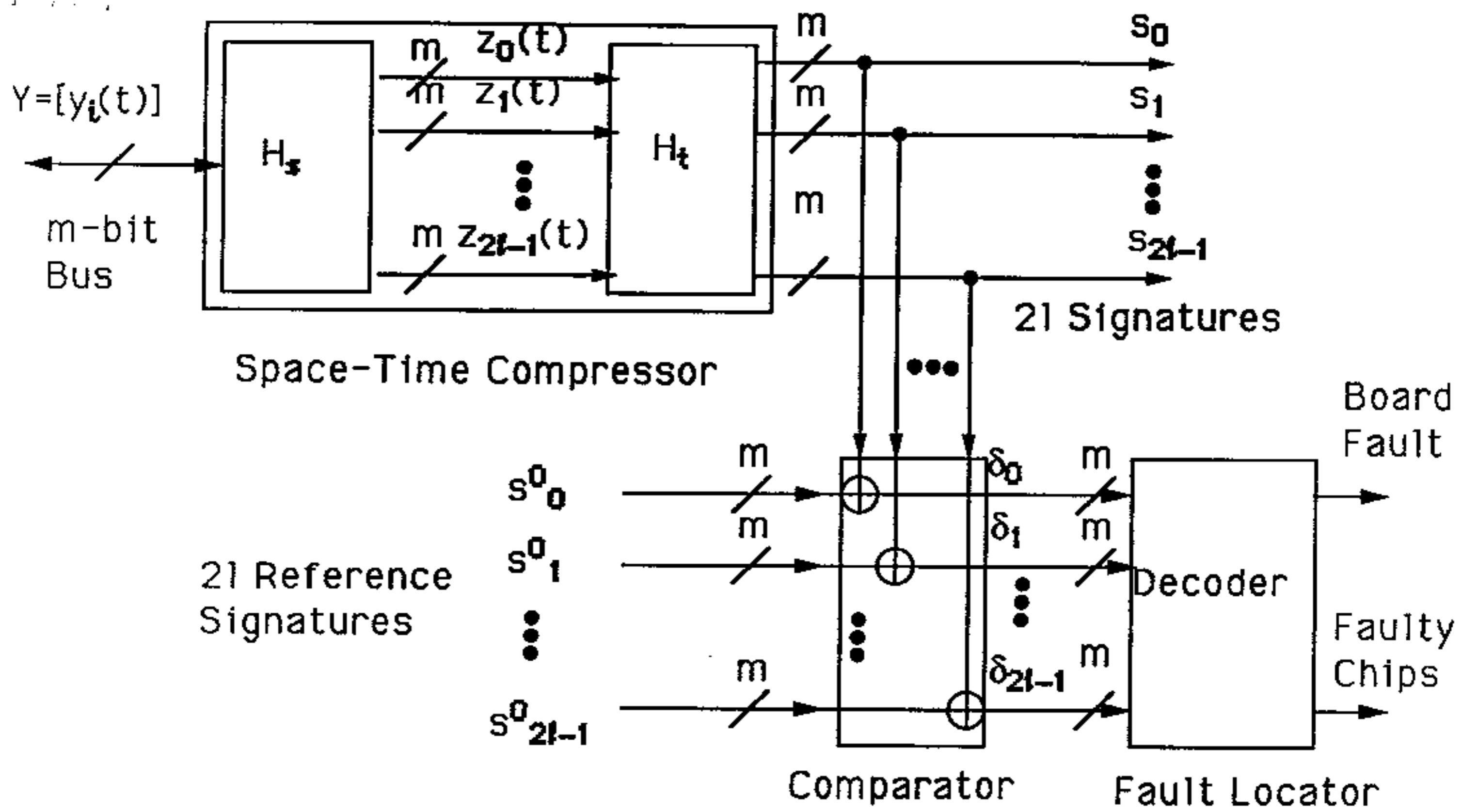


Figure 3

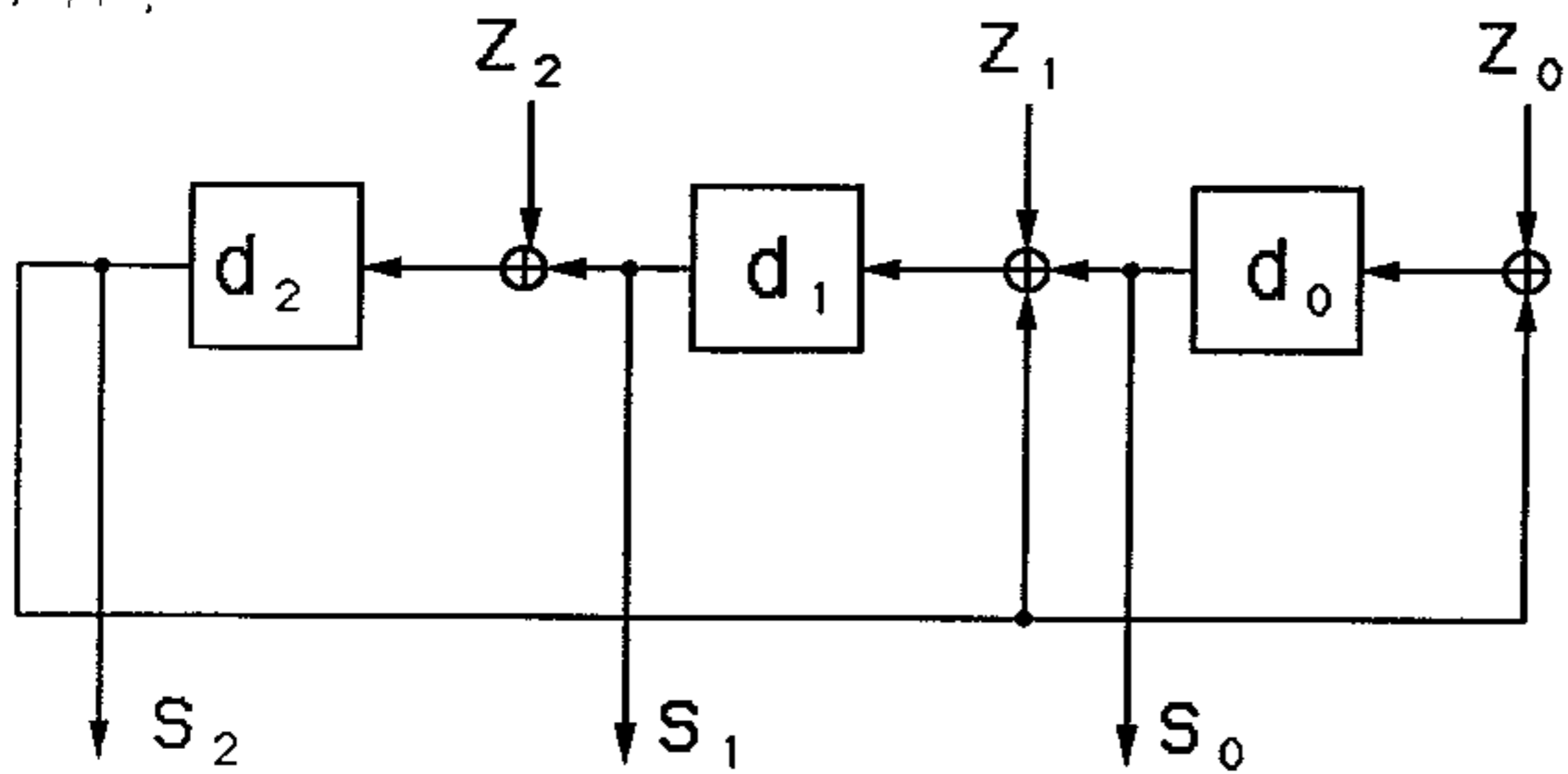


Figure 4

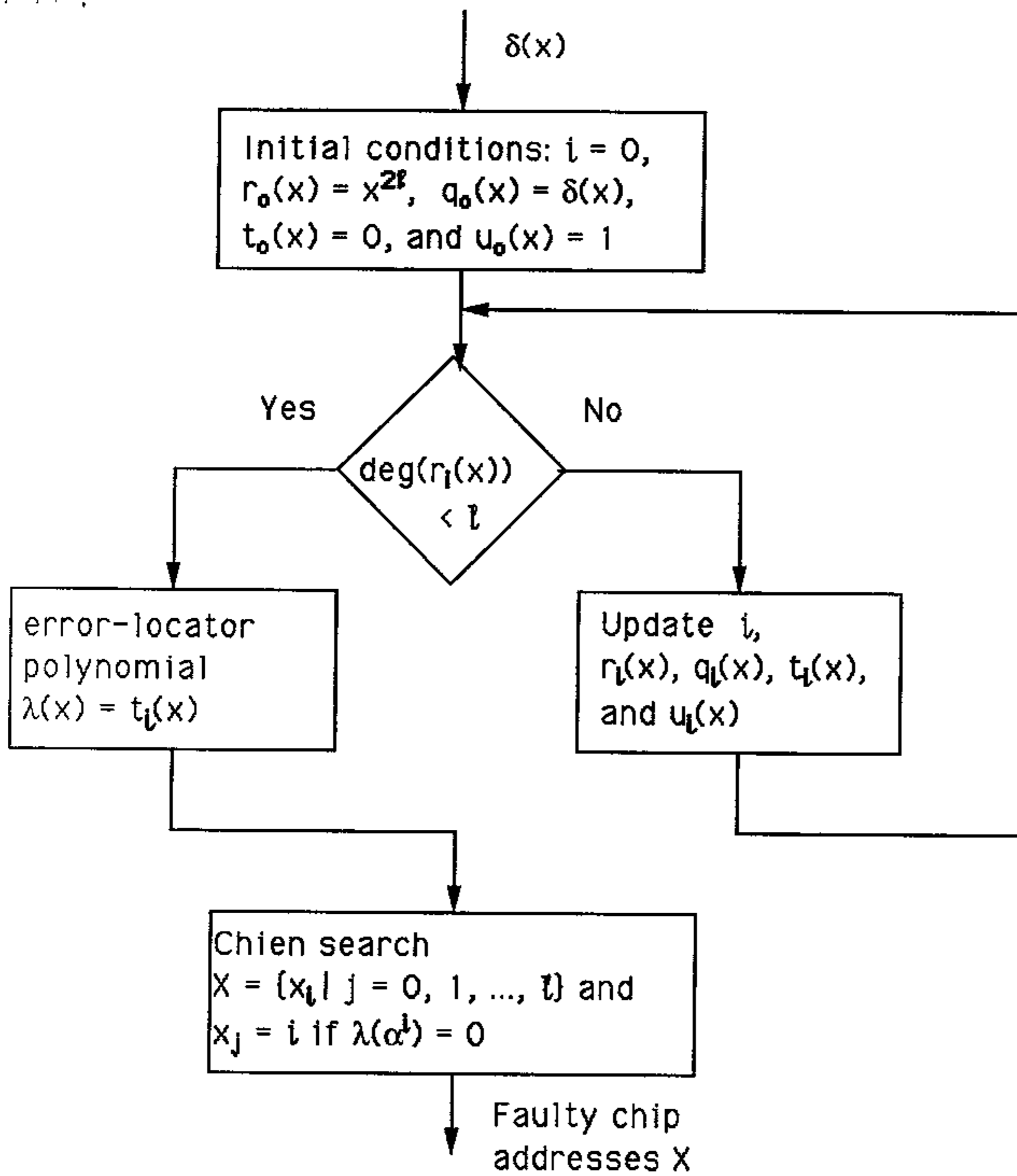


Figure 5

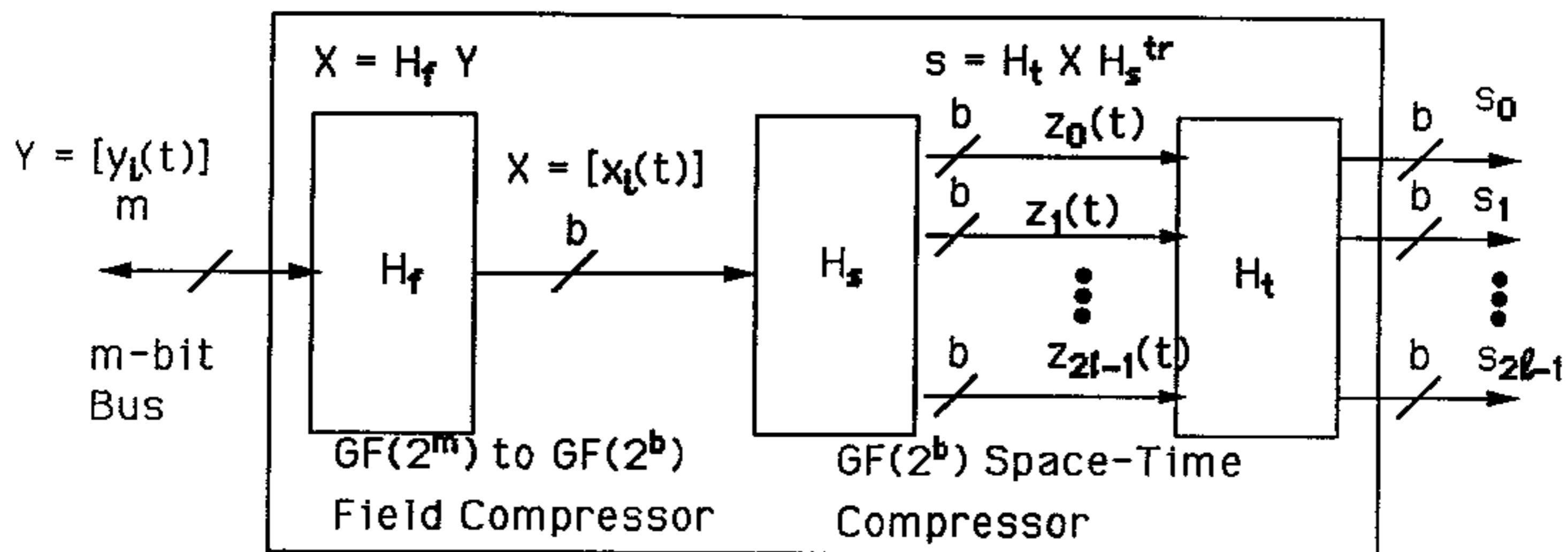
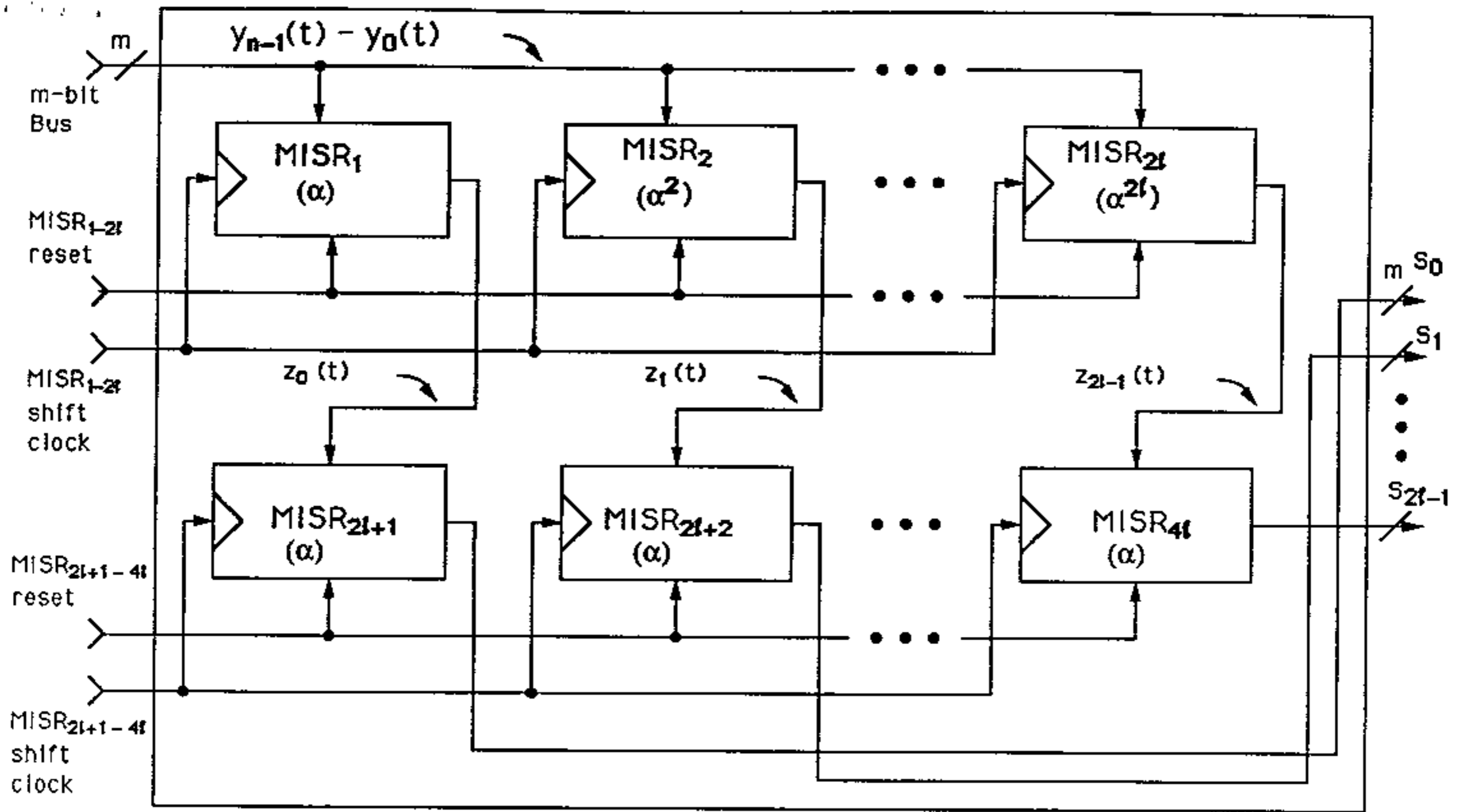


Figure 6



Legend :MISR _{k} $k = 1, \dots, 2l$ = m -bit MISR multiplying by α^k
 MISR _{k} $k = 2l+1, \dots, 4l$ = m -bit MISR multiplying by α

Figure 7

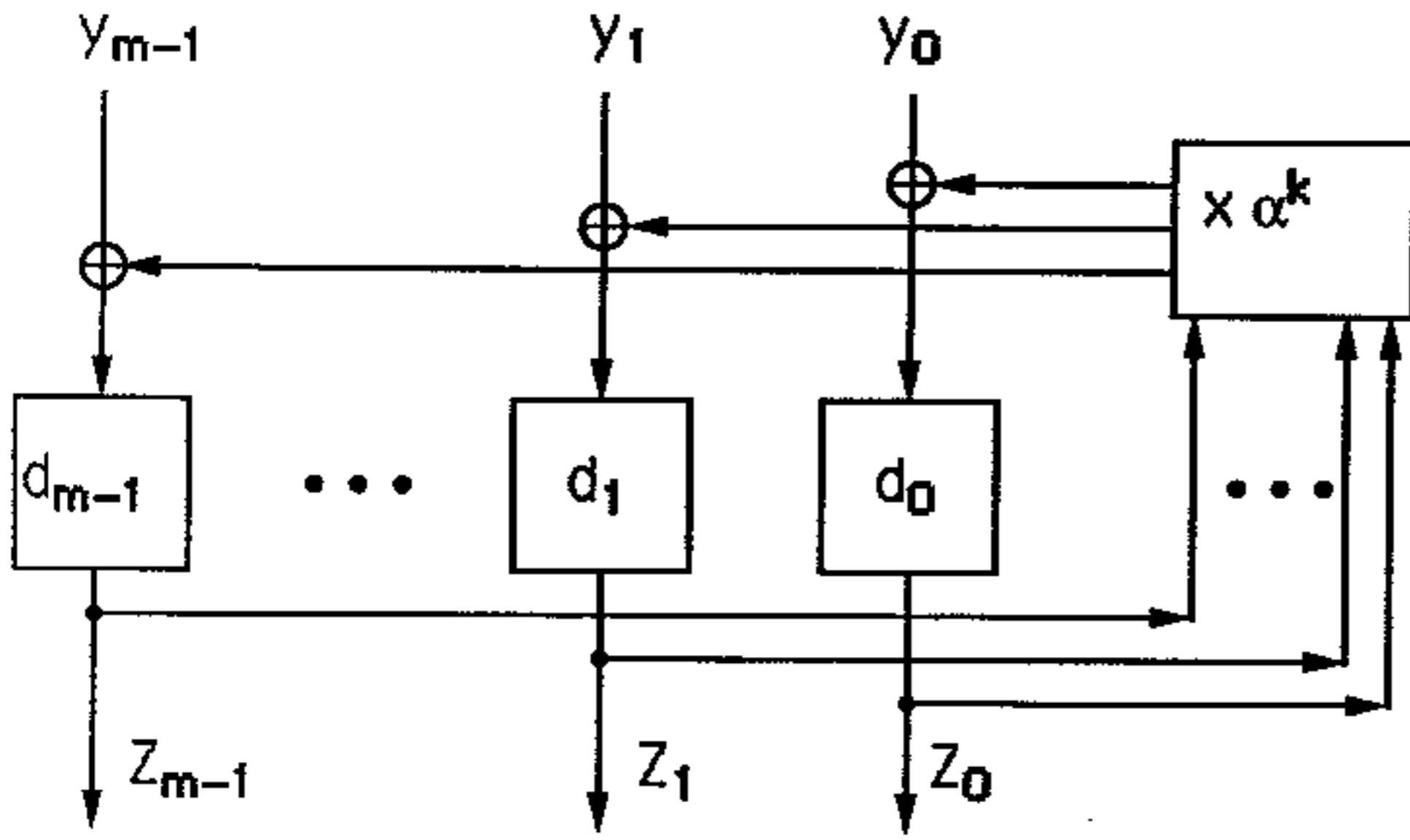


Figure 8

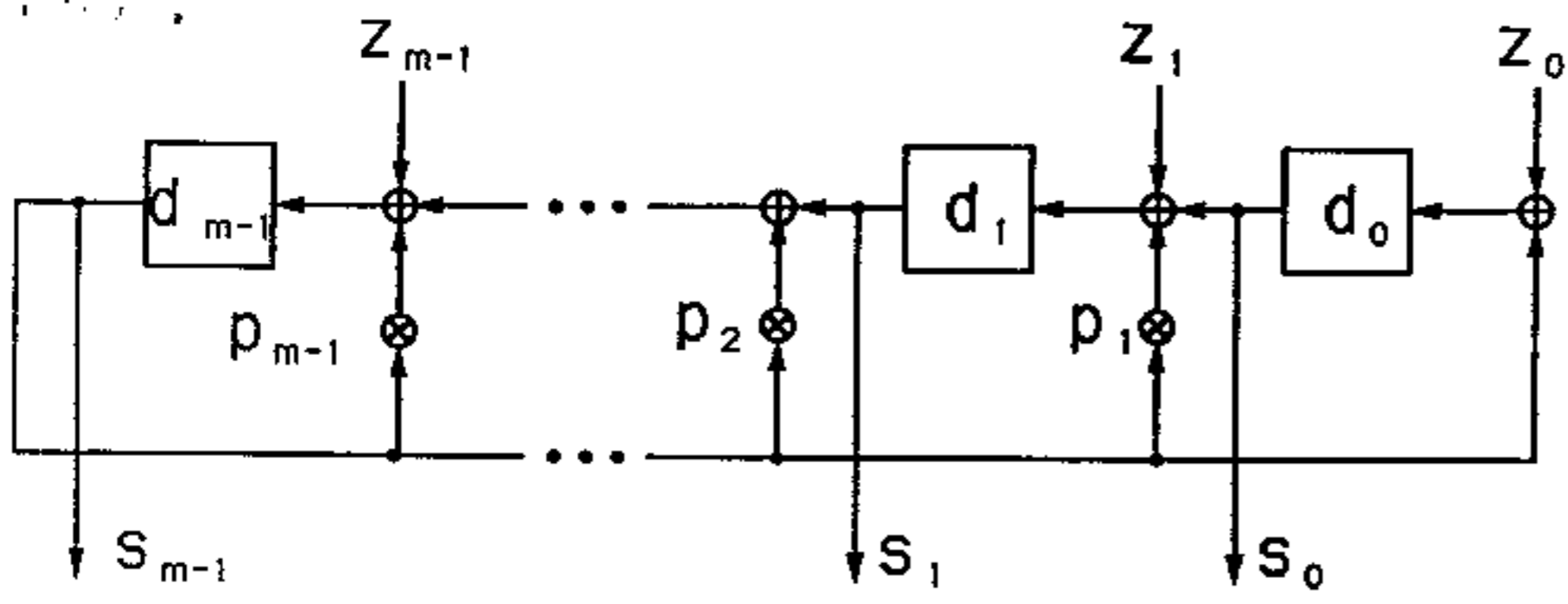
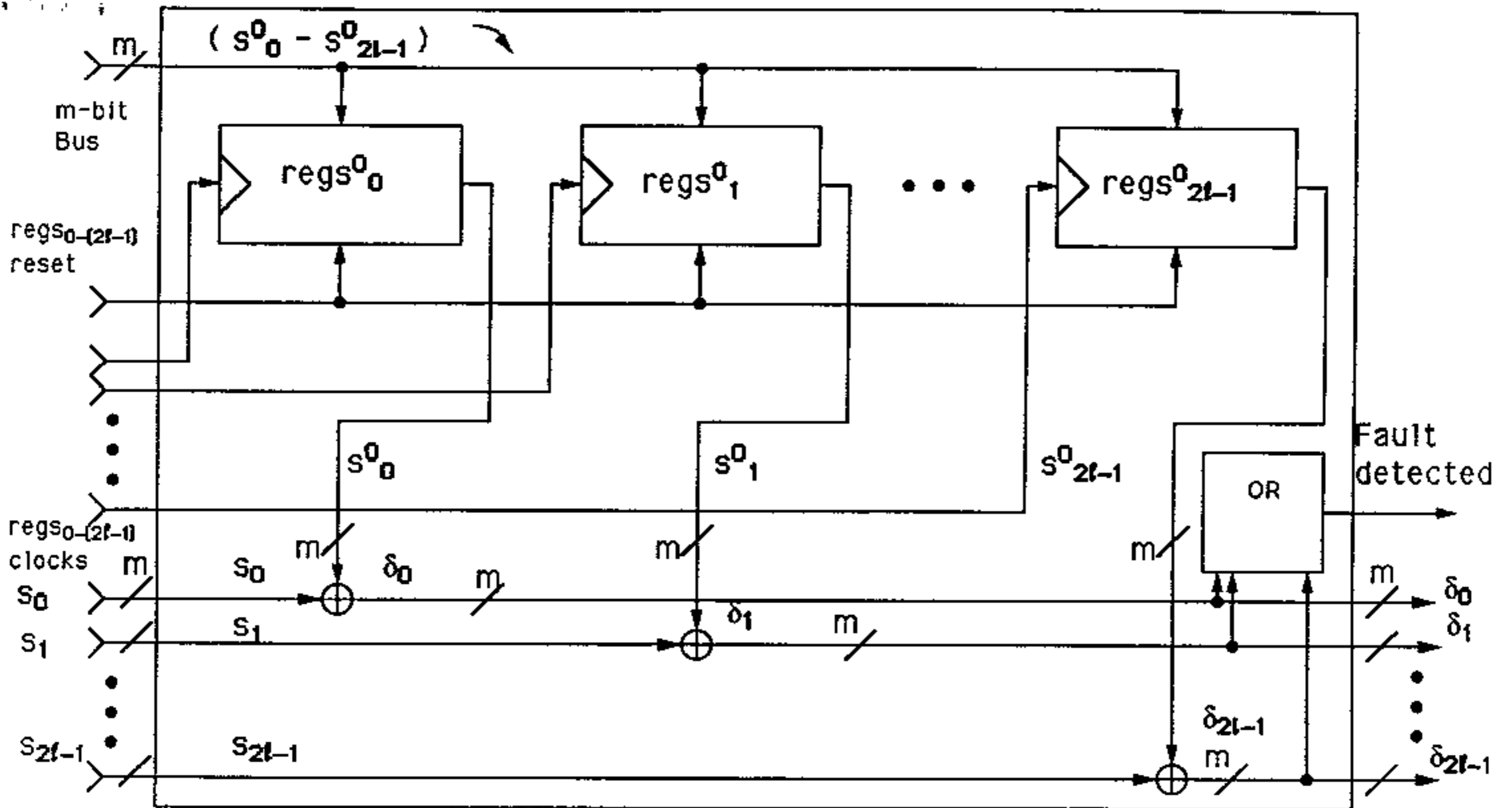
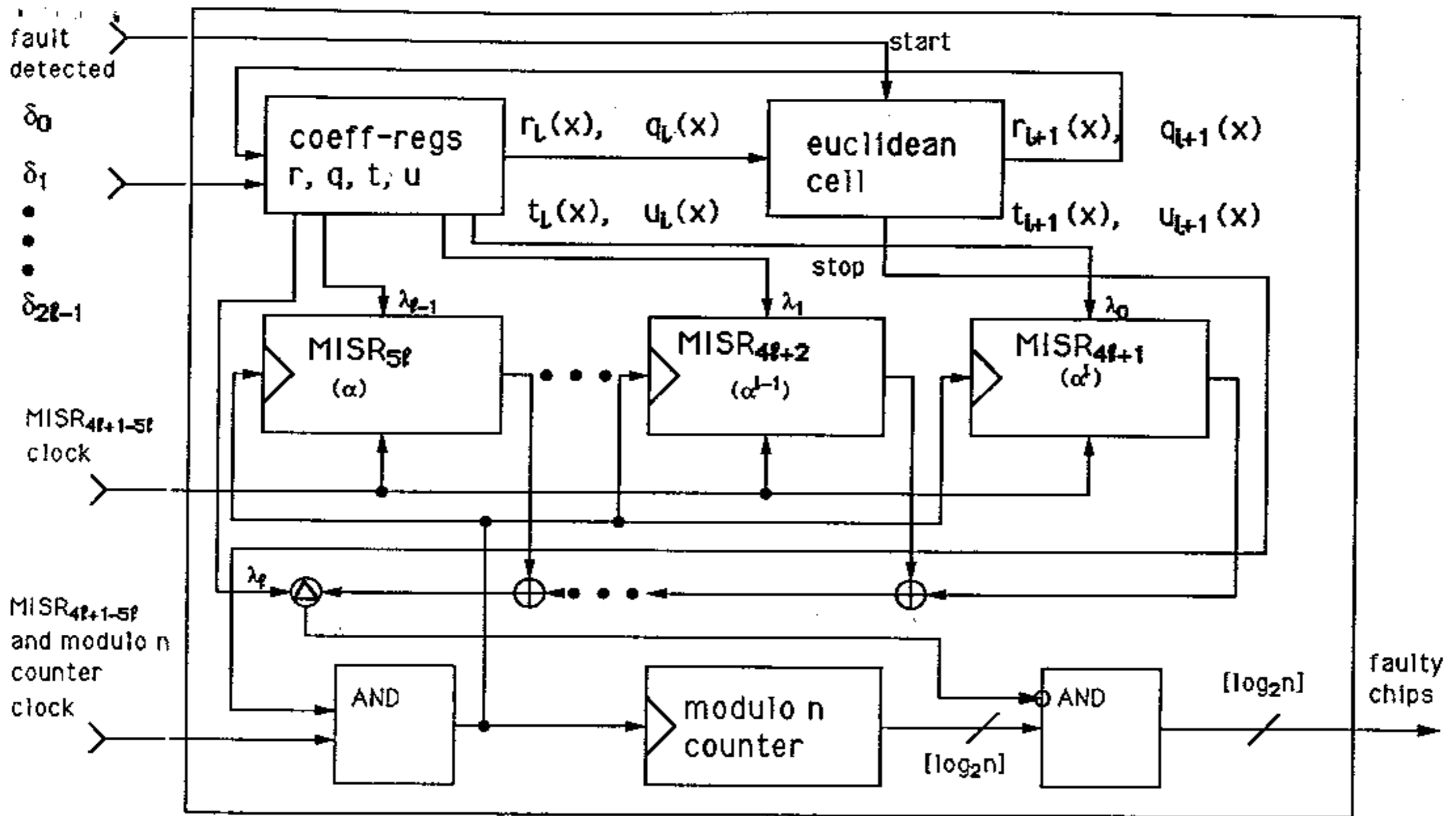


Figure 9



Legend : $regs^0_t, t = 0, \dots, 2l-1 = m$ -bit register
 \oplus = finite field adder (m 2-input XOR gates)

Figure 10



Legend : coeff-reg r, q = $2l$ m-bit cyclic shift register
 coeff-reg t, u = $l+1$ m-bit cyclic shift register
 MISR _{$l = 4l+1, \dots, 5l$} = m-bit MISR with feed back α^l

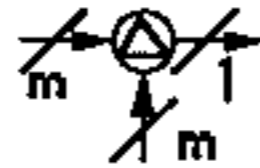
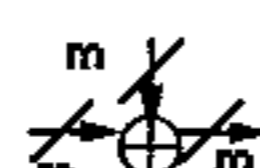
 = m-bit comparator
 = finite field adder

Figure 11

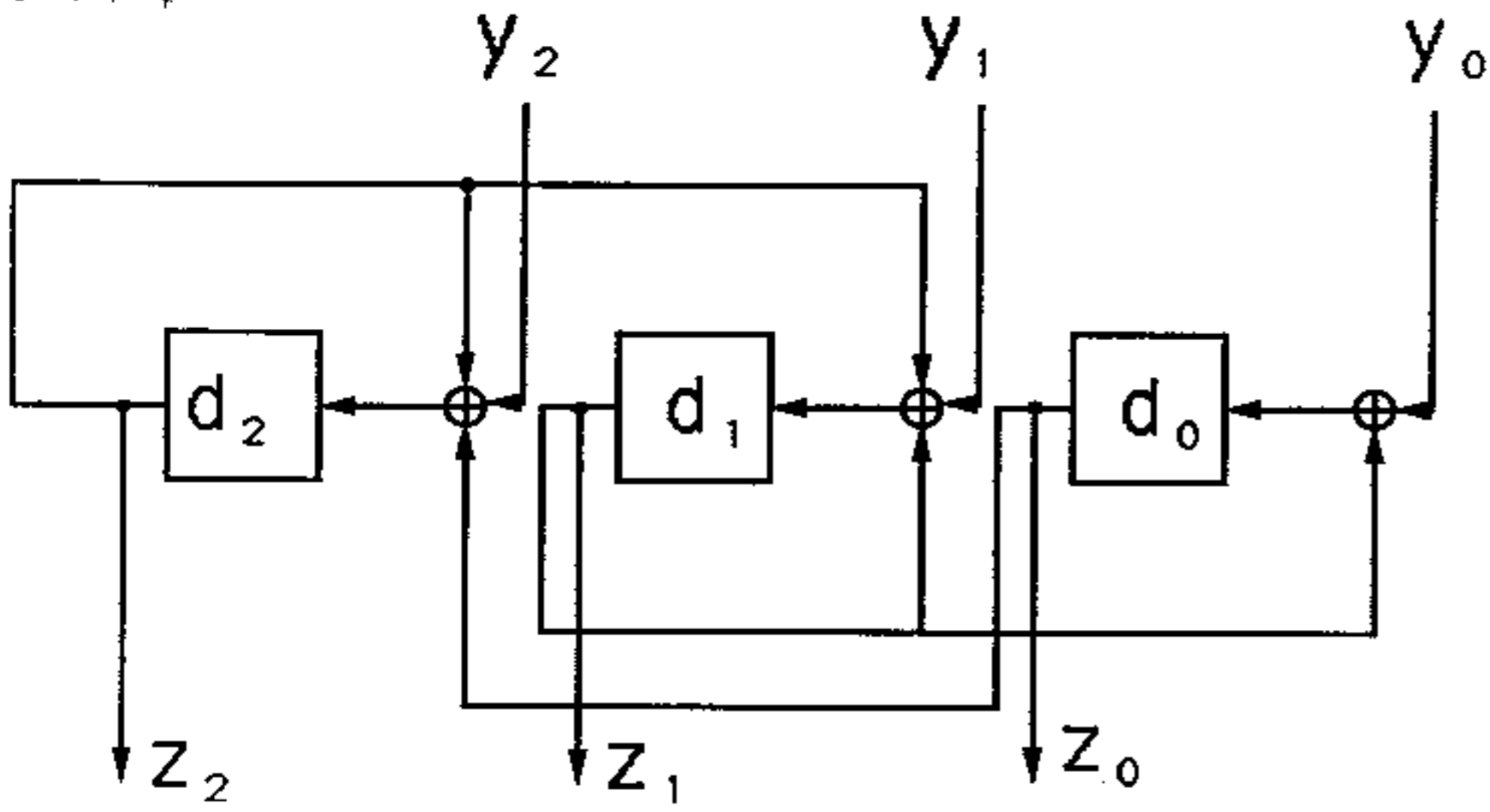


Figure 12

Savings

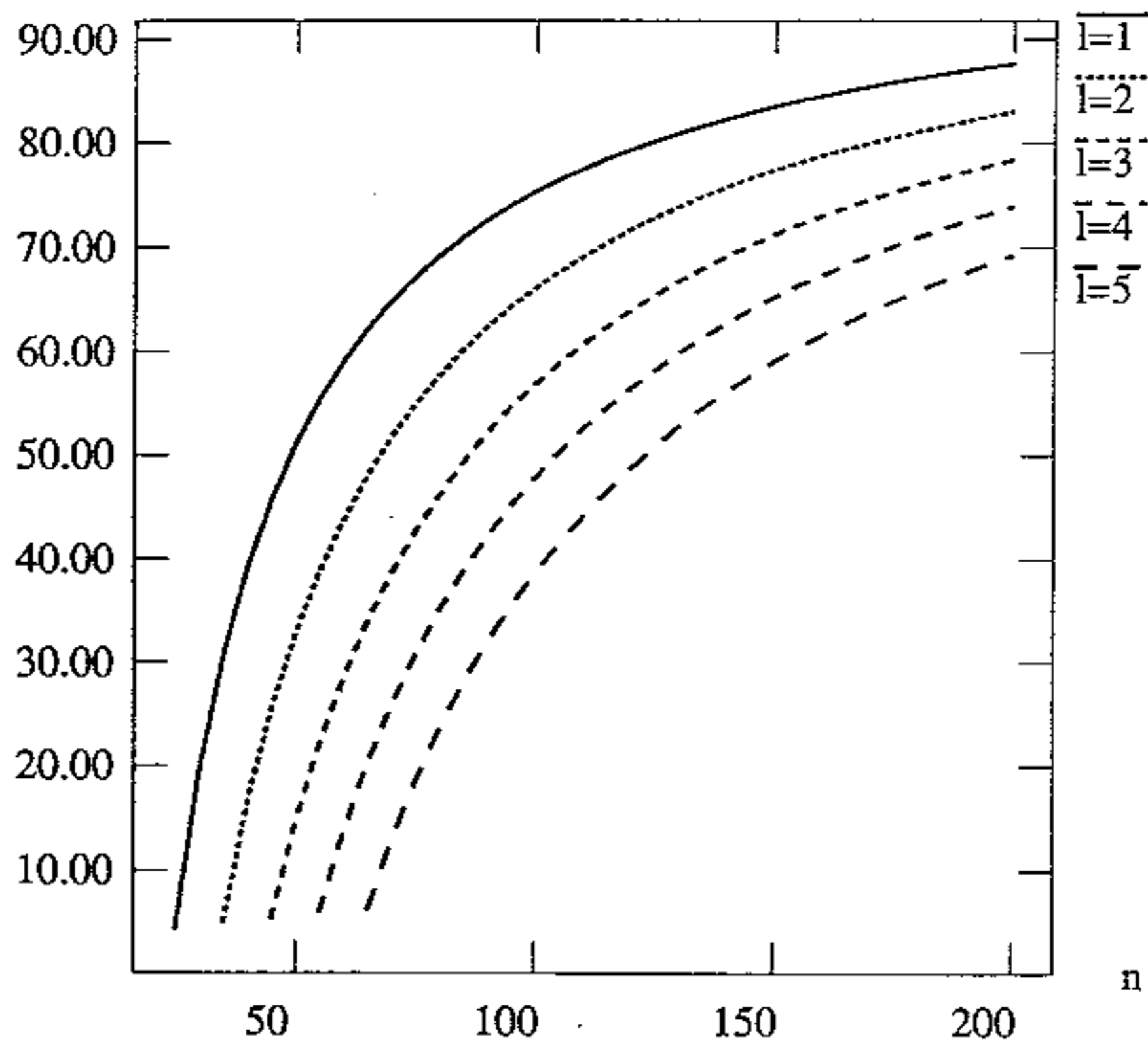


Figure 13

Table 1: Representation of the Nonzero Elements of $GF(2^3)$, Generated by $p(\alpha) = \alpha^3 \oplus \alpha \oplus 1 = 0$.

Table 2: Step (i): Modified Euclidean Algorithm-Example.

Table 3: Step (ii): Chien Search Algorithm-Example.

Table 4: State Transitions of $MISR_1, \dots, MISR_4$ performing Space Compression of Test Responses-Example.

Table 5: State Transitions of $MISR_5, \dots, MISR_8$ performing Time Compression of the Space Compressed Test Responses-Example.

Table 6: Hardware Overheads (Gate Counts) for Different Numbers of Chips n On a Board with a 32-bit System Bus.

Table 7: Prototype Diagnostic Chip Data.

Table 1: Representation of the Nonzero Elements of $GF(2^3)$, Generated by $p(\alpha) = \alpha^3 \oplus \alpha \oplus 1 = 0$.

t	$d_2d_1d_0$	α^2	α	α^0	α^t
0	0 0 1			1	1
1	0 1 0		α		α
2	1 0 0	α^2			α^2
3	0 1 1		$\alpha \oplus 1$		α^3
4	1 1 0	$\alpha^2 \oplus \alpha$			α^4
5	1 1 1	$\alpha^2 \oplus \alpha \oplus 1$			α^5
6	1 0 1	α^2		$\oplus 1$	α^6

Table 2: Step (i): Modified Euclidean Algorithm-Example.

i	$r_i(x)$	$q_i(x)$	$t_i(x)$	$u_i(x)$
0	x^4	$\alpha^3x^3 \oplus \alpha^5x^2 \oplus \alpha^4x \oplus \alpha$	0	1
1	$\alpha^5x^3 \oplus \alpha^4x^2 \oplus \alpha x$	$\alpha^3x^3 \oplus \alpha^5x^2 \oplus \alpha^4x \oplus \alpha$	x	1
2	$\alpha x^2 \oplus \alpha x \oplus \alpha^6$	$\alpha^3x^3 \oplus \alpha^5x^2 \oplus \alpha^4x \oplus \alpha$	$\alpha^3x \oplus \alpha^5$	1
3	$\alpha^3x^2 \oplus \alpha^3x \oplus \alpha^2$	$\alpha x^2 \oplus \alpha x \oplus \alpha^6$	$\alpha^6x^2 \oplus \alpha x \oplus \alpha$	$\alpha^3x \oplus \alpha^5$
4	α^5	$\alpha x^2 \oplus \alpha x \oplus \alpha^6$	$x^2 \oplus x \oplus \alpha^4$	$\alpha^3x \oplus \alpha^5$

Table 3: Step (ii): Chien Search Algorithm-Example.

i	$\lambda_0\alpha^{2i}$	$\lambda_1\alpha^i$	λ_2	$\lambda(\alpha^i) = \lambda_0\alpha^{2i} \oplus \lambda_1\alpha^i \oplus \lambda_2$	X
0	1	1	α^4	α^4	-
1	α^2	α	α^4	0	$x_1 = 1$
2	α^4	α^2	α^4	α^2	-
3	α^6	α^3	α^4	0	$x_2 = 3$
4	α	α^4	α^4	α	-

Table 6: Hardware Overheads (Gate Counts) for Different Numbers of Chips n On a Board with a 32-bit System Bus.

n	Straightforward Approach	Universal Diagnostic Approach				
		$l = 1$	$l = 2$	$l = 3$	$l = 4$	$l = 5$
		20	8,903	10,522	14,467	18,412
40	17,363	10,535	14,480	18,425	22,268	26,213
60	25,803	10,535	14,480	18,438	22,268	26,213
80	34,323	10,548	14,493	18,438	22,281	26,226
100	42,783	10,548	14,493	18,438	22,281	26,226
200	85,283	10,561	14,506	18,451	22,294	26,239

Table 7: Prototype Diagnostic Chip Data.

Implementation:

Diagnostic Chip Block	Number of Equivalent 2-Input Gates
Control	781
Data Compressor	2,520
Comparator	1,549
Fault Locator	
(i) Euclidean Cell and Registers	9,478
(ii) Chien Search	859
Total	15,187

Operating Conditions (100 ns Clock):

Parameter	Time	Unit
Single Fault Location	109,400	ns
Triple Fault Location	317,200	ns

Mark G. Karpovsky

Mark Karpovsky is a professor of computer engineering at Boston University and the Director of the university's Research Laboratory for Design and Testing of Computer Systems. His research interests include testing, fault-tolerant computing, design for testability, diagnostics of computer hardware, and error correcting codes. He published over 100 papers and several books in these areas. He has also been a consultant for international companies, including IBM, Digital, Honeywell, and AT&T. He is an IEEE fellow.

Saeed M. Chaudhry

Saeed Chaudhry is a PhD candidate in the Department of Electrical, Computer and Systems Engineering at the Boston University. His research interests are in testing and error-correcting codes. Saeed received his BE degree in avionics engineering, with distinction, from NED University, Karachi, Pakistan in 1979 and the MS degree in computer engineering from the Boston University in 1989. He is a student member of the IEEE Computer Society.