AN ERROR-LOCATING SIGNATURE ANALYZER TO IDENTIFY FAULTY UNITS IN DIGITAL SYSTEMS

Vadim Geurkov*, Vladimir Dynkin**, and Reza Sedaghat*

*Electrical and Computer Engineering Department, Ryerson University, Toronto, Canada
**Department of System Identification, Institute for Control Science, Moscow, Russian Federation

ABSTRACT

In traditional signature analysis, faulty units are identified by mismatch between the actual and reference signatures. The amount of reference signatures can be quite large in complex systems, which complicates the error-detection procedure. We show how under certain restrictions this amount can be reduced preserving the diagnosis resolution and the aliasing rate. As a result, we construct a signature analyzer that slightly differs from the traditional one, but it is also capable of locating faulty units. If the analyzer is implemented in the external automated test equipment, the throughput of the latter may be significantly increased.

Index Terms— Built-in self-test, error-control codes, signature analysis, test data compression

1. INTRODUCTION

The density and complexity of integrated circuits have grown as modern digital technology has developed. These circuits are used in advanced applications whose down time due to failure can be critical. Therefore, it is very important to detect improper functions of these applications as efficiently as possible. Here efficiency means that (i) only a small number of faults can escape detection and (ii) the test hardware overhead is low.

One of the popular methods satisfying the above conditions has been signature analysis [1]. A sufficiently large number of test stimuli are applied to the inputs of a unit under test (UUT) in order to detect as many faults as possible. The output response is compacted into a short signature. If the actual signature does not match the expected error-free signature, the UUT is considered to be faulty.

A UUT is often partitioned into a certain number of replaceable units (RUs), each of which can be characterized by a unique signature. Hardware complexity of a diagnostic system based on signature analysis and being able to locate faulty RUs is proportional to the amount of RUs. This is particularly sensible for self-test systems built-in to complex UUTs. In such systems, test response compaction is conducted on the same chip as the UUT [2]. Any reduction of the number of reference signatures required to diagnose the given amount of RUs, without deterioration of the probability of error escape would increase efficiency in testing. In this paper we demonstrate how to solve this problem, provided that the maximum number of RUs, which may fail simultaneously, is known a priori.

The problem of extraction of diagnostic information from signatures has attracted attention of test researchers for a long time. In [3] authors use modular and burst error-correcting codes to identify erroneous bits within the output response of the UUT. In [4] and [5] a technique is developed to locate single errors in the output sequence, using a linear feedback shift register (LFSR) and multiple input signature registers, (MISRs) respectively. In [6] authors propose to use a multiple error-correcting BCH code for diagnostic purposes. The techniques developed in [3] - [6] are oriented to a relatively small number of errors, since the test hardware complexity rapidly increases with the growth of potential errors. Reference [7] considers an X-tolerant convolutional compactor with low aliasing rate, which is used to diagnose faulty cells of scan chains. This approach also deals with a small number of errors and the size of signatures is greater compared to the traditional (deconvolutional) approach. The diagnosis procedure is complex for built-in implementation; it is based on heuristics and a significant amount of time is spent on its completion.

The methods developed in [8] and [9] are aimed to identify any arbitrary number of errors in the output responses. They are based on a two-level procedure that uses two different classes of error-control codes. This procedure resembles a technique used in an error-control theory to construct an error-locating code (ELC) [10]. As we will see below, ELCs require less redundancy than considered in [8] and [9]. The encoding/decoding procedures for ELCs are quite simple as well. But their application to testing is not straightforward. In this paper we will consider how ELCs can be employed for the diagnosis of digital circuits. The approach can be used at the chip, board or system level.

2. PROBLRM STATEMENT

A general form of a UUT broken into \( n \times q \) replaceable units 
\( RU_{00}, \ldots, RU_{nq} \) is represented in Fig. 1. To prevent
propagation of errors through the feedback paths, the UUT comprises tri-state buffers $T$. While testing, all the feedback paths are broken. Thus, Test stimuli are applied to the UUT and the output responses coming from the RUs are shifted into an LFSR or MISR. It is important to probe RUs in the level order. A level of an RU is the maximum distance of its inputs from the inputs to the UUT. Testing starts from the lowest level RUs. If signatures of RUs in this level are erroneous, then these RUs will be repaired. Afterwards, testing resumes to the next level of RUs. This procedure ensures that errors in the RUs are independent. We will refer to the RUs of the level $i$ simply as a slice $i$. If the diagnosis procedure is automated, then the size of the memory to store error-free signatures for the entire UUT is $n'' \times q \times \sigma$ bits, where $n''$ is the number of RUs in a slice, $q$ is the number of slices, and $\sigma$ is the bit-length of the signature analyzer (coinciding with the number of outputs of an RU in case of using a MISR). These requirements are normally satisfied in scan chained UUTs, in this case the scan outputs are directly connected to the signature analyzer. Distinct portions of the data sequences obtained through those chains characterize distinct RUs.

Our objective is to locate erroneous RUs within a slice as efficiently as possible (i.e. with maximum fault coverage and minimum hardware overhead) provided that any number of errors can occur in faulty RUs and the number of RUs that can fail simultaneously, does not exceed $t$.

There are a few available classes of codes that can be exercised here, but none of them can be directly applied to attain the objective. Indeed, error-detecting codes (EDCs) only detect errors. Error-correcting codes (ECCs) can in addition, locate and correct errors in data sequences. However the complexity of the decoder readily becomes impractical for long word lengths and/or for large number of errors, that may occur in the UUT. In fact, error correction is not always necessary. It is often quite enough to identify faulty RUs which can be repaired afterwards. In communication, such situations are remedied by ELCs lying in-between EDCs and ECCs. ELCs can identify erroneous sub-words within a word. Their great advantage is that they possess less redundancy versus ECCs. Though, in the presence of multiple errors ELC’s redundancy may become quite large. We will show how to reduce the redundancy of ELCs to practical levels at the expense of some deterioration of the code capabilities. As we see below, this deterioration is fully acceptable for testing. Therefore, the most favourable choice toward achieving the main objective will be ELCs. The codec for an ELC will be able to locate faulty RUs and will be referred to as an error-locating signature analyzer.

### 3. Diagnostic Method

ELCs were introduced by Wolf and Elspas in [10]. Goethals has constructed a sub-class of ELCs that is equivalent to cyclic codes under certain permutation of coordinates [11]. Since cyclic codes tolerate simple practical realization, we will concentrate on encoding/decoding procedures for that particular class.

Existence of cyclic ELCs is stated in the following theorem [11].

Let $C'$ of block length $n'$ be a cyclic $d$-error-detecting code, generated by the product $g(x) = g_1(x) \ldots g_{r}(x)$ of $r$ distinct irreducible factors over $GF(q)$, and let $GF(q^{m'})$ be the smallest extension field of $GF(q)$ containing all roots of $g(x)$.

Let $C''$, of block length $n''$ relatively prime to $n'$, be a cyclic $t$-error-correcting code generated by the product $G(x) = G_1(x) \ldots G_{s}(x)$ of $s$ distinct irreducible factors over $GF(q^{m''})$, and let $GF(q^{m''})$ be the smallest extension field of $GF(q^{m''})$ containing the $n'' \cdot t$ roots of unity.

Then there exists an ELC $C$ of length $n = n' n''$ and having not more than $r s p m$ check digits over $GF(q)$, which locates up to $t$ erroneous sub-words, each sub-word having not more than $d$ errors. Furthermore, $C$ is equivalent to a cyclic code.

Let $\beta$ and $\gamma$ be any root of $g_1(x)$ and $G_1(x)$ respectively; $\beta$ and $\gamma$ be primitive roots of unity ({$n'$}-th and $n''$-th respectively). The following matrices may be taken as parity check matrices for the codes $C'$, $C''$, and $C$ respectively:

$$H' = \begin{bmatrix} \beta_0^0 & \ldots & \beta_0^{n'-1} \\ \vdots & \ddots & \vdots \\ \beta_s^0 & \ldots & \beta_s^{n'-1} \end{bmatrix} \quad H'' = \begin{bmatrix} \gamma_0^0 & \ldots & \gamma_0^{n''-1} \\ \vdots & \ddots & \vdots \\ \gamma_s^0 & \ldots & \gamma_s^{n''-1} \end{bmatrix} \quad H = \begin{bmatrix} \gamma_0^0 H_1 & \ldots & \gamma_0^{n''-1} H_1 \\ \vdots & \ddots & \vdots \\ \gamma_s^0 H_1 & \ldots & \gamma_s^{n''-1} H_1 \end{bmatrix} \quad (1)$$

**Example 1** [11]. Let $C'$ be a cyclic (7,1) - code over $GF(2)$, which detects six errors and is generated by the polynomial $g(x) = g_1(x) g_2(x) = (1+x+x^3)(1+x^2+x^3)$, with roots in $GF(2^3)$. Let $C''$ be a cyclic (9,5) - code over $GF(2^5)$, which corrects two errors and is generated by the polynomial $G(x) = G_1(x) G_2(x) = (1+x+x^3)(1+\beta x+x^2)$ with roots in $GF(2^6)$. Let $\beta$ and $\gamma$ be primitive elements of the fields $GF(2^3)$ and $GF(2^6)$ respectively. Then:

$$H' = \begin{bmatrix} \beta_0^0 & \ldots & \beta_0^6 \\ \beta_1^0 & \ldots & \beta_1^6 \end{bmatrix} \quad H'' = \begin{bmatrix} \gamma_0^0 & \gamma_0^6 \\ \gamma_1^0 & \gamma_1^6 \end{bmatrix} \quad H = \begin{bmatrix} \gamma_0^0 H_1 & \gamma_0^6 H_1 \\ \gamma_1^0 H_1 & \gamma_1^6 H_1 \end{bmatrix}$$
Redundancy of the code \( C \) is \( 6 \times 4 = 24 \) bits. \( C \) is a \((63,39)\) – code over \( GF(2) \), which locates 2 out of 9 sub-blocks of length 7 distorted by not more than 6 errors each. A conventional analyzer would require \( 9 \times 6 = 54 \) bits.

Encoding procedure for this code uses the following generator polynomial: \((1+x+x^2+x^4+x^6) (1+x^2+x^4+x^5+x^6) (1+x+x^3+x^5+x^6) \) \((1+x+x^2+x^3+x^4) \) \((1+x+x^2) \). The procedure is simple; however the bits coming from distinct sub-words must be permuted. This does not cause a problem for communication applications, but is inconvenient for diagnostic purposes: there must be a multiplexer in the test system, which works in compliance with the permutation table.

Another problem that appears when the ELCs are applied to testing is that convenient ELCs deal with relatively small numbers of errors in sub-words.

The decoding procedure coincides with the known decoding procedure for the \( t \)-error-correcting code \( C' \).

In order to alleviate the second problem of encoding mentioned above, we will extend the length of code \( C \) keeping its redundancy at the same level. If the parameters of the original and extended codes are \((n, k)\) and \((N', K')\) respectively, where \( n' - K' = N' - K = r' \), the extended parity-check matrix will be \([\text{HH...H}]\), where the standard matrix \( \text{H} \) are repeated \( N'/n \) times. Such a manipulation makes the resulting ELC non-optimal. And some errors may now escape detection. If any combination of independent errors is likely to occur and \( N' \) is quite large, the probability of undetected error \( P_{nd} \) (or the aliasing probability \( P_{al} \)) can be estimated as \( P_{nd} = P_{al} \leq \frac{1}{2^{2r'}} \). For example, \( r' = 16 \), then \( P_{nd} = 0.000015 \). A minor increase of \( P_{nd} \) versus \( 0 \) is fully justified by reduction of hardware for the decoder.

To solve the second problem of encoding we will slightly modify the encoding procedure. Let

\[
A = (A_0, A_1, ..., A_{n-1})
\]

be a code vector of a code \( C \), where

\[
A_l = (c_0^{(l)}, c_1^{(l)}, ..., c_{n-1}^{(l)}), \quad l = 0, r_z - 1
\]

or

\[
A_l = (a_0^{(l)}, a_1^{(l)}, ..., a_{n-1}^{(l)}), \quad l = r_z, n_z - 1
\]

Here \( c_0^{(l)} \) and \( a_0^{(l)} \) are respectively the \( i - \text{th} \) check and data characters of the \( l - \text{th} \) sub-block.

For a code vector \( S(A) = AH^T = 0 \), or from (1) and (2),

\[
S(A) = (\sum_{j=0}^{n-1} A_j^c H_1^T, ..., \sum_{j=0}^{n-1} A_j^c H_{n_z}^T) = (A_0 H_1^T, ..., A_{n_z-1} H_{n_z}^T) H_z^T = 0
\]

where

\[
\begin{align*}
A_l H_1^T &= (\beta_0^{(l)}, ..., \beta_{n_z}^{(l)}), \quad l = 0, n_z - 1 \\
\sum_{j=0}^{n_z-1} c_j^{(l)} \beta_j + \sum_{j=0}^{n_z-1} a_j^{(l)} \beta_j &= 0, l = 0, r_z - 1, i = \overline{1,u} \\
\beta_0^{(l)} &= \sum_{j=0}^{n_z-1} a_j^{(l)} \beta_j, \quad l = r_z, n_z - 1, i = \overline{1,u}
\end{align*}
\]

It follows from (3) and (4) that:

\[
S(A) = (\sum_{j=0}^{n_z-1} \beta_j^{(0)} \gamma_j, ..., \sum_{j=0}^{n_z-1} \beta_j^{(n_z-1)} \gamma_j) = 0
\]

or

\[
\sum_{j=0}^{n_z-1} \beta_j^{(l)} \gamma_j + \left[ \sum_{j=0}^{n_z-1} \beta_j^{(l')} x \right] \mod g(x) \mid_{x=\overline{1,v}} = 0, \quad i = \overline{1,u}, \quad j = \overline{1,v}
\]

Let \( \left( \sum_{j=0}^{n_z-1} \beta_j^{(l')} x \right) \mod g(x) \mid_{x=\overline{1,v}} = \sum_{j=0}^{n_z-1} \beta_j^{(l')} \gamma_j \). Then

\[
\sum_{j=0}^{n_z-1} \beta_j^{(l')} \gamma_j = -\sum_{j=0}^{n_z-1} \beta_j^{(l')} \gamma_j
\]

Let us equate the coefficients for the same powers of \( \gamma_j \):

\[
\beta_j^{(l')} = \beta_j^{(l')}, \quad l = 0, r_z - 1, i = \overline{1,u}
\]

From (7) and (5) we get

\[
\sum_{j=0}^{n_z-1} c_j^{(l)} \beta_j = -\beta_j^{(l')} - \beta_j^{(l')}, \quad l = 0, r_z - 1, i = \overline{1,u}
\]

The check characters can be found by solving (8).

According to the above, the standard encoding procedure, which requires check bits to be a part of sub-words, can be modified. First, we compute modulo \( g(x) \) remainders \( S'_i, i = 0, ..., n - l, \) of all output responses of the fault-free RUs. Next, we compute modulo \( g(x) \) remainder \( S_{ref} \) of the polynomial with the factors \( S_0, ..., S_{n-1} \). We will refer to \( S_{ref} \) as generalized signature (of the slice). The procedure is irrespective of the type of the signature analyzer employed (whether it is LFSR or MISR).

When decoding, we perform the similar procedure with actual output responses. If for a given slice, the actual generalized signature \( S_a \) equals to \( S_{ref} \), we assume the RUs in the slice are fault-free. Otherwise, we use the code \( C' \) decoding procedure to locate faulty RUs.

The relative size of the memory for good signatures is:

\[
\sigma \times n \times p \times q / \sigma \times n \times q = v \times p / n
\]

If a UUT contains e.g. \( q = 30 \) slices with \( n = 9 \) RUs in each slice, the required aliasing rate is \( 2^8 \), and only two RUs may fail simultaneously, then the ratio will be \( 4/9 = 0.44 \). Instead of \( 6 \times 270 = 1620 \) bits we require \( 0.44 \times 1620 = 720 \) bits.

**Example 2**. Let us apply 5100 test stimuli to a UUT with 257 RUs in a slice. Only a single RU may fail at a time, although the number of faults that may occur in the RU is not limited. We construct the ELC on the basis of the two codes: an EDC \( C' \) generated by the \( g(x) = x^8 + x^5 + x^3 + x + 1 \), and a \((257,255)\) - Hamming code \( C \) correcting single \( 2^8 \)-ary errors and generated by the \( G(x) = x^2 + \beta x + 1 \) over \( GF(2^8) \), where \( \beta \) is a primitive element in \( GF(2^8) \). The actual error-locating signature analyzer will consist of the two LFSRs described by \( g(x) \) and \( G(x) \) respectively. The latter one is the decoder of \((257,255)\)-Hamming code over \( GF(2^8) \). It is represented in Fig. 2. The size of the memory of error-free signatures will reduce by \((257 \times 8) / (2 \times 8) = 128.5 \) per slice. If there are 100 slices, the memory size will change from \( 257 \times 8 \times 100 = 205600 \) bits to \( 2 \times 8 \times 100 = 1600 \) bits. The aliasing rate will then be \( 1/2^8 = 0.004 \). Hardware complexity
of the code
After shifting all digits and obtaining the sum
forward until its left stage,
the system.
proportional to the number of replaceable units residing in
diagnostic resolution and aliasing rate. The gain is
compression schemes) is reduced, while preserving the
error-locating codes (compared with conventional
simultaneously is small, the hardware of a tester based on
diagnosis of digital systems that have been partitioned into
We have adapted error-locating codes for the off-line
showed how to use it to identify faulty units in digital

Fig. 2. The decoder of (257,255)-Hamming code over
$GF(2^8)$
of the ELC decoder only slightly exceeds the one for the
conventional LFSR.

Let the encoded word of the ELC be of the form $A = (A_0,\ldots,A_{256})$, where $A_i=(11000100\ldots\overline{0}1)$, $l=0,2,3,5$; $A_i=(01111010\ldots\overline{0}1)$, $l=4$; $A_j=(00\ldots\overline{0}00)$, $l=1,6,7,255$. Since $(1 + x + x^2 + x^3 + x^4 + x^5 + x^{1023} + x^{1275} + x^{5100}) \mod g(x)|_{x=\beta} = \beta^8$ and $(x + x^2 +
\beta x^4 + x^5 + x^6 + x^{1023} + x^{1275} + x^{5100}) \mod g(x)|_{x=\beta} = \beta^6$, then
$S_0 = S_2 = S_3 = S_5 = \ldots = S_{256} = 0$. And since $(\beta^8 + \beta^9 x^2 + \beta^8 x^3 + \beta^9 x^4 + \beta^8 x^5) \mod G(x) = 0 x^2 + \beta x^3 + \beta^9 x^4 + \beta^8 x^5 |_{x=\gamma}$, where $\gamma$ is a primitive element of the
field $GF(2^{16})$, then $S_{ref} = (0, \beta^8) = (00000000, 01101010)$. It occupies only two bytes.

Now let the second SU be faulty and the output
response changes to $(00\ldots\overline{0})$. Then $\overline{S}_2 = 0$ and $(\beta^8 + \beta^9 x^2 +
\beta^8 x^3 + \beta^9 x^4 + \beta^8 x^5) \mod G(x) = \beta^8 x^0 + 0 x^1 |_{x=\gamma}$. Therefore, $S_0 = (\beta^8, 0) = (11010100, 00000000)$. Since $S_0 \oplus S_{ref} \neq 0$, the decoder of the code $C^-$ (see Fig. 2) starts locating the erroneous RU.
After shifting all digits and obtaining the sum $S_0 \oplus S_{ref} = (\beta^8, \beta^8) = (11010100, 01101010)$, the decoder will shift
forward until its left stage, $R_0$ holds all zeroes. This occurs at the
255-th shift. Therefore, the number of the faulty RU is
257-255 = 2.

4. CONCLUSION

We have adapted error-locating codes for the off-line
diagnosis of digital systems that have been partitioned into
replaceable units. If the number of units that may fail
simultaneously is small, the hardware of a tester based on
error-locating codes (compared with conventional
compression schemes) is reduced, while preserving the
diagnostic resolution and aliasing rate. The gain is
proportional to the number of replaceable units residing in
the system.

We developed encoding/decoding procedures for error-
locating codes, which have simple practical implementation.
We introduced an error-locating signature analyzer and
showed how to use it to identify faulty units in digital
systems.

The method considered can use any conventional
compacting scheme. The type of the scheme determines the
overall aliasing rate.

5. REFERENCES

Journal, Publisher, Location, pp. 1-10, Date.

Publisher, Location, Date.

codes for diagnosis of digital devices”, in Proc. 8th Rep. Conf.,

[4] W. McAnney and J. Savir, “There is information in faulty


detection and identification via signature analysis,” J. Electron.


IEEE Trans. CAD of Integr. Circ. & Syst., vol. 18, no. 2, pp. 203-211,


[12] W. Peterson and E. Weldon, Error-correcting codes. MIT