Concatenable Cellular Automata Register Design for Built-in Self-Test

Andrzej Hlawiczka, Michał Kopeć

Technical University of Gliwice
ul. Pstrowskiego 16, 44-100 Gliwice, POLAND

Abstract

This paper describes the principles of concatenations of Cellular Automata based LFSRs (MISR). It proves some theorems and lemmas which allow for quick determining if a certain Cellular Automata based register has the reducible polynomial of the form \(xg(x)\) or \((x+1)h(x)\). Further it answers the question how to find a minimal set of standard CAD system cells to permit to construct the largest number of various length and various primitive characteristic polynomials Cellular Automata based LFSRs (MISRs). The paper also list many examples of d-bit sliced Cellular Automata registers families which allow for building concatenations having primitive polynomials.

1: Introduction

The idea of testing digital networks with the use of an interface compatible to P1149.1 standard [1] which facilitates testing procedure is shown in Fig.1.1. The P1149.1 bus consisted of four lines TDI, TDO, TCK and TMS serves to transfer serially control and test data from/to the tester to/from the circuit under test (CUT).

The interface comprises a boundary scan path input register (BIR), boundary scan path output register (BOR) and a test access port (TAP). In normal applications, both BIR and BOR are reconfigured during various test modes in the normal, scan, external/internal, and sample modes of operation [1-2].

In RUNBIST mode [1] BIR may act as a built-in pseudo-random pattern generator while BOR acts as a built-in compactor, which converts the response matrix of the CUT into the form of a short word. Thus, the boundary scan path can assume the part of a built-in tester [1-2].

In both modes, BIRs and BORs are configured either in a linear feedback shift register (LFSR) which generates pseudorandom tests [2-3] or in a multi-input signature register (MISR) which performs a linear parallel compression [4-5]. Some methods of configuration of the linear feedback loop of such registers are known e.g. in the form of an internal Exclusive-OR [4], external Exclusive-OR [4] or top-bottom, bottom-top [4,6] feedback loop. Recently, a configuration based on one-dimensional Cellular-Automata (CA) using rules 150 and 90 [7-9] has been proposed. Fig.1.2a illustrates a block diagram of such n-cell CA-MISR with a null boundary condition. The circuit diagram of j-th cell of CA-MISR whose \(k_j = 1\) for the rule 150 (with feedback loop) and \(k_j = 0\) for the rule 90 (no feedback loop) is shown in Fig.1.2b. These cells will be referred as cell 150 and cell 90 respectively and denoted as 1 and 0.

The set of properties of CA- MISRs and CA-LFSRs presented in [7-9] makes these registers particularly suitable for incorporating the built-in self-test (BIST) with boundary scan path and causes the design process to be very susceptible to automation.

The layout of every 150/90 cell is always the same independently of the location in the register. This facilitates designing of the whole n-stage CA-LFSR or CA-MISR which may be created by advancing successively n cells having the same height. Too much application flexibility of 150/90 cells, however, may in many cases cause unnecessary chip surface losses (the surface intended to implement CA based registers) and labor efforts when
Fig. 1.2: A cellular automata based linear register (a), circuit diagram of \( j \)-th cell (b), concatenation of a four-bit register with a five-bit one (c)

designing CA-registers. By completing the library of 150/90 cells with standard ones in the form of concatenable \[11\] d-bits cellular automata linear registers but with "economically" designed layouts one can obtain also "economically" designed layouts of the whole n-stage linear registers and reduce their designing time. Implementation of concatenable d-bits CA registers makes easier the design automation of BIST and increases the yield. Thus, they may be applied for bit-slice implementation of CA-LFSRs and CA-MISRs having primitive polynomials. The example of such concatenation is illustrated in Fig. 1.2c. Some concatenations of bit-sliced CA-register \( a_0a_1a_2a_3 \) with bit-sliced CA register \( b_0b_1b_2b_3b_4 \) creating a 9-bit CA register having primitive characteristic polynomial are given below:

| 7+bi | 7+ei | 8+5o | 8+8i | 9+60 | 9+20 | 9+4i | 9+90 |

The sign "i" means \( k_j = 1 \) (cell rule 150) while sign "o" means \( k_j = 0 \) (cell rule 90). A hex sign denotes four bits e.g. \( 50 = \text{oioi} \).

The length of the test generators and test compressors depends upon the CUT and thus may vary considerably. During the design modification, necessity may arose for a slight lengthening/shortening of the test generator and the compressor. It is also recommended to construct test generators and compressors with the use of a library that contains a slight set of concatenable d-bits CA-linear registers.

The aim of this paper is to solve the problem how to choose a minimal set of concatenable standard library cells in order to:

- obtain the possibility of creation of registers that vary in length,
- realize as many primitive polynomials given degree as possible.

Which set of concatenable standard cells is best applicable to design CA-LFSRs and CA-MISRs of different lengths and having the primitive polynomials at the same time? Answering "yes" we can considerably reduce the space of search. To obtain it, the properties of the sequence \( k_0k_1k_2...k_{n-2}k_{n-1} \) of the CA-linear registers are analyzed in this paper.

Unfortunately, the CA based linear register in question have an essential disadvantage i.e. a very complicated correlation between the sequence of rule 150/90 cells and the characteristic polynomial \( p(x) \) of the CA-MISR (CA-LFSR) constructed from such set of cells \[8\]. As a result handling of the bits of the sequence \( k_0k_1k_2...k_{n-2}k_{n-1} \) has almost unrecognizable influence on the characteristic polynomial of CA-MISRs and CA-LFSRs constructed from such sets of rule 150/90 cells. The recognition whether the characteristic polynomial is reducible, prime or primitive has an essential significance in determining the static and dynamic probability of error aliasing by the CA-MISR and CA-LFSR \[5-6\] as well as in determining the length of a pseudorandom test sequence generated by the CA-LFSR \[2-3\]. It is found that among all possible n-stage CA-LFSRs (CA-MISRs) more than a half has an infinitive characteristic polynomial \( p(x) = x^g(x) \) or even characteristic polynomial \( p(x) = (x+1)^h(x) \) to be not applicable in BIST \[10\] at all.

Therefore, this paper addresses the following additional problem:

- what is the influence of handling bits \( k_j \) of the sequence of rule 150/90 cells on the properties of the characteristic polynomial related to this sequence?

2: Properties of concatenation of bit-sliced CA-registers

Bhavsar \[11\] has designed a family of sliced structures containing linear feedback registers with such characteristic polynomials that it was possible to create n-stage primitive concatenations i.e. such chains of the a/m sliced structures
The attribute of the sequence $K_a$ is a pair $<a_{0n-1},a_{0n}> = C(K_a)$, where $a_{0n-1}$ and $a_{0n}$ are free terms of the polynomials $p_{a-1}(x)$ and $p_{a}(x)$ respectively.

The attribute of an empty sequence $K_0$ is $<1,0>$ (It satisfies formula 2.1).

Definition 2.3. A CA-register is referred to infirmitive, when $K_n = p_{a}(x) = x w(x)$

A CA-register is referred to even, when $K_n = p_{a}(x) = (x+1) w(x)$

Theorem 2.1. The module $K_n$ is infirmitive only when $C(K_a) =<1,0>$.

Proof: The module $K_n$ is infirmitive only when a free term of its characteristic polynomial is equal to 0. Let us see then how the attribute $K_n$ and the free term of its characteristic polynomial $a_{0n}$ change during adjoining of a successive cell $k_{n-1}$. From formula 2.1 we have:

$$a_{0n} = a_{0n-2} + k_{n-1} a_{0n-1}$$ (2.2)

It is easy to prove, that the attribute $<a_{0n-1},a_{0n}>$ never is equal to $<0,0>$.

Equation 2.2 may be expressed in the form of a graph as illustrated in Fig.2.1. where the node with the attribute $<1,0>$ means no free term in the polynomial $p_{a}(x) = K_0$ (infirmitive polynomial), whilst the nodes with the attribute $<0,1>$ mean non-infirmitive polynomials $p_{b}(x) = K_n$.

On the basis of the graph from Fig. 2.1 it is possible to determine quickly whether the given CA-register $K_n$ is infirmitive with no necessity for calculation of its characteristic polynomial as based on the iterative formula 2.1. The CA-register is infirmitive only when the sequence $K_n$ describes the path along the branch of the graph from the point START $<0,1>$ to the point $<1,0>$. Examples of infirmitive registers are also given on the graph.

The transition along the given branch corresponds to adjoining (disjoining) one cell to (from) the module $K_n$.

\[ Fig.2.1. \text{Attribute changes of the sequence } K_{n-1}\]

Examples of infirmitive registers:

$0, 11, 101, 1001, 10001, 10\ldots 01, 0-0, 0-11, 0-101, 11-0, 101-0, 11-1, 101-101, 11-11; *-$ means either 0 or 1. Further, the designations $S=<0,1>, N=<1,1>$ and $U=<1,0>$.
Now, let us define a module characteristic being the generalization of the concept of attribute.

Definition 2.4. The characteristic of a bit-sliced CA-register \( K_d \) is a function \( \chi_{K_d}(x) \) satisfying the following conditions:

1. \( \chi_{K_d}(x) \) is a function \( x \in \{S,N,U\} \);
2. \( \chi_{K_d}(x) = C(K_d) \);
3. \( K_d \) is an empty module \( \Leftrightarrow \chi_{K_d}(x) = x \);
4. \( K_d = k_i = 0 \Leftrightarrow \chi_{K_d}(S) = U; \chi_{K_d}(N) = N; \chi_{K_d}(U) = S; \)
5. \( K_d = k_i = 1 \Leftrightarrow \chi_{K_d}(S) = N; \chi_{K_d}(N) = U; \chi_{K_d}(U) = S; \)
6. \( \chi(x) = \chi(\chi(x)) = \chi_2(x) \chi_0(x) \chi(x) \) (O - the sign of concatenation of the function)

Definition 2.5. The type of a module \( K_d \) is an ordered ternary:

\[ T(K_d) = \langle \chi_{K_d}(S), \chi_{K_d}(N), \chi_{K_d}(U) \rangle \]

The type of an empty module is an identity element.

Table 1 shows all possible types of modules \( K_d \) with accompanying examples which have been found by implementing definition 2.4 into a simple program.

<table>
<thead>
<tr>
<th>Module type ( K_d )</th>
<th>All modules ( K_d )</th>
<th>Examples for ( d &gt; 4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( S,N,U )</td>
<td>( o_0 )</td>
<td>( a_5 )</td>
</tr>
<tr>
<td>( S,U,N )</td>
<td>( o_i )</td>
<td>( b_4 )</td>
</tr>
<tr>
<td>( S,U,N )</td>
<td>( o_0 )</td>
<td>( d_2 )</td>
</tr>
<tr>
<td>( U,U,S )</td>
<td>( o_i )</td>
<td>( c_3 )</td>
</tr>
<tr>
<td>( U,N,S )</td>
<td>( o_i )</td>
<td>( o_c )</td>
</tr>
<tr>
<td>( U,N,S )</td>
<td>( o_i )</td>
<td>( d_2 )</td>
</tr>
<tr>
<td>( U,N,S )</td>
<td>( o_0 )</td>
<td>( o_c )</td>
</tr>
</tbody>
</table>

Denote further the set of all types of modules \( \{S,N,U\}, \{S,U,N\}, \{S,U,N\}, \{N,U,S\}, \{U,N,S\} \) as \( T \).  

Lemma 2.1. Every module of type \( S,N,U \) \( S,U,N \) \( N,S,U \) is non-informative.

Proof: It follows from Defintions 2.4, 2.5, and theorem 2.1.

Let \( A_d = a_0a_1a_2...a_d-1 \) and \( B_d = b_0b_1b_2...b_d-1 \). Also assume that modules \( A_d \) and \( B_d \) can be of any type from the set \( T \).

Lemma 2.2. The types of concatenation

\[ A_d + B_d = a_0a_1a_2...a_d-1b_0b_1b_2...b_d-1 \]

are Table 2 are true.

Proof: The proof results directly from the definition of characteristic 2.4.

The algebraic system consisted from the set \( T \) along with the operation called concatenation of module types creates a group.

Proof: The proof results from the fact that the operations in the form of module concatenation is joinable, the type of an empty module is an identity element, there is existed a type reciprocal to every type of the set \( T \).  

Theorem 2.3. Every concatenation of modules of types \( \{S,N,U\}, \{S,U,N\} \) or \( \{S,U,N\}, \{N,S,U\} \) is non-informative.

Proof: The proof results from Table 2. Only the subsets of types \( \{S,N,U\}, \{S,U,N\} \) or \( \{S,U,N\}, \{N,S,U\} \) along with concatenation of these types create subgroups.

A pair of modules \( \{a,b\} \) permits for creation of non-informative concatenations \( k^4 \) cells long.

E.g. \( a, ab, aaa, aaab... \) Further examples of another non-informative pairs of modules whose concatenations are non-informative are \( \{5,4\}, \{45,ef\}, \{50,02\}, \{ba,10\} \) etc.  

Lemmas assume that the complemented sequence \( K'_n \), we will denote with \( K'_n = k'_0k'_1k'_2...k'_n2k'_{n+1} \).

Lemma 2.3. \( [10] \)

\[ kokk_2...k_2k_1 = p_a(x) \Leftrightarrow k'_0k'_1k'_2...k'_n2k'_{n+1} = p_a(x+1) \]

Theorem 2.3. If the module \( K_a \) is informative then the module \( K_a \) is even and vice versa.

Proof: On the basis of lemma 2.3. \( K'_n = p_a(x + 1) \)

If \( p_a(x) \times g(x) \) then \( p_a(x + 1) = (x + 1) g(x + 1) \). This implication can also be reversed.

The following sets \( \{a,b\} \), \( \{5,4\}, \{45,ef\}, \{ba,10\} \) are examples of the module pairs having non-even module concatenations.

Conclusion 1. The common part of a set of modules with non-informative concatenations and a set of modules with non-even concatenations is the reduced set of modules whose every concatenation is non-informative and non-even.

Pairs \( \{a,b\} \) and \( \{45,ef\} \) are an example of such set whilst pairs \( \{b,d\} \) and \( \{46,84\} \) not.

Every pair \( \{a,b\} \) \( \{45,ef\} \) may create the base to find concatenations with primitive characteristic polynomials. Such concatenations will be referred further as primitive concatenations. The probability of finding a primitive concatenation for each pair of modules appears to be greater than for the remaining cases. It is confirmed by Table 3.
which illustrates all primitive concatenations constructed from the pairs of modules \{a, b\} and \{b, d\} respectively. The second set of modules is decisively worse. It is impossible to realize a concatenation 8, 12 and 24 bits-long having a primitive polynomial whilst the quantity of different realizable polynomials is far much lower.

Table 3. All primitive concatenations constructed from modules \{a, b\} and \{b, d\} respectively

<table>
<thead>
<tr>
<th>k</th>
<th>4</th>
<th>8</th>
<th>12</th>
<th>16</th>
<th>20</th>
<th>24</th>
</tr>
</thead>
<tbody>
<tr>
<td>{a, b}</td>
<td>a ab aab aabba aaaa</td>
<td>b baabaab bababab</td>
<td>baab babababababa</td>
<td>babb babbabbabbabab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>{b, d}</td>
<td>b bbb dbbb --</td>
<td>d dd bddd</td>
<td>dbbdd dbdd bddd</td>
<td>bdd bdd bdd</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3: Conclusions

On the basis of theoretical results presented herein above a special software package called CALIGEN (Cellular Automata Linear feedback shift register GENerator), has been developed to permit for:

- determining properties of characteristic polynomials on the basis of a given set of modules,
- finding, for a given set of \(d\)-bit sliced cellular automata linear registers, all \(k \times d\) bit-long primitive concatenations \((k \times d \leq 320)\).

This software package permitted to find all primitive concatenations mentioned previously as well as primitive concatenations of any reasonable length \(k \times 8\) bits, constructed from the module families \(\{45, ef\}\) and illustrated in Table 4. They enable the layouts of CA-MISRs and CA-LFSRs to be automatically designed. In the Table 4 a concatenation of \(n\) modules \(x\) is denoted as \(x^n\).

Calculations have been made which have proven the usability of theorems from Chapter 2 for searching module families that permit to create primitive concatenations. These calculations permit to evaluate quickly which set of modules is better. Because of elimination of concatenations having reducible polynomials these theorems give a relatively high probability of obtaining many different concatenations having a primitive polynomial. The result is to choose such primitive concatenation which will be best in the signature analysis [12]. The authors have also found a quad of modules \{a, b, c, d\} which permits to construct primitive concatenations of any length \(n \geq 11\). The results obtained enable also CA-MISRs (CA-LFSRs) to be redesigned by adjoining or disjoining 150/90 cells in order to achieve that the final CA-register has not a reducible informative or even polynomial. An open problem remains, in the opinion of the authors, how to develop such a graph which would permit to indicate CA-registers with primitive polynomials.

5: References