Deterministic BIST for RNS Adders

Haridimos T. Vergos, *Member*, *IEEE*, Dimitris Nikolos, *Member*, *IEEE*,
Maciej Bellos, and Costas Efstathiou

Abstract—Modulo $2^n - 1$ adders as fast as n-bit 2's complement adders have been recently proposed in the open literature. This makes a Residue Number System (RNS) adder with channels based on the moduli 2^n, $2^n - 1$, and any other of the form $2^k - 1$, with $k < n$, faster than RNS adders based on other moduli. In this paper, we formally derive a parametric, with respect to the adder size, test set, for parallel testing of the channels of an RNS adder based on moduli of the form $2^n, 2^n - 1, 2^k - 1, 2^l - 1, \ldots,$ with $l < k < n$. The derived test set is reusable; it can be used for any value of $n, k, l, \ldots$, regardless of the implementation library used and is composed of $n^2 + 2$ test vectors. A test-per-clock BIST scheme is also proposed that applies the derived test vectors within $n^2 + 2n$ cycles. Static CMOS implementations reveal that the proposed BIST offers 100 percent postcompaction fault coverage and an attractive combination of test time and implementation area compared to ROM and FSM-based deterministic BIST or LFSR-based pseudorandom BIST.

Index Terms—Residue Number System, Built-In Self-Test, deterministic and pseudorandom tests, formal test sets.

1 Introduction

Built-in Self-Test (BIST) is an effective approach for testing contemporary integrated circuits (ICs) that reduces the need for external test since the circuit and its tester are implemented in the same chip, enabling the circuit to test itself. To meet the ever-shrinking time-to-market requirements of custom ICs, rapid test-pattern generation and test set embedding are essential. Test sets derived by deterministic test pattern generation, however, require large implementation areas if embedded as a finite state machine or by storing them in an ROM, while easily implementable pseudorandom test pattern generators require long test sequences. Besides the specific disadvantages mentioned previously, the above schemes, except the ROM-based one, suffer from lack of reusability. In the case of regular circuits, reusable formal test sets may be derived, which are parameterized with respect to the number of inputs of the circuit. If this test set is also independent of the implementation library, migration to new libraries can be done rapidly. In this paper, we derive a formal test set and propose an efficient BIST scheme for Residue Number System (RNS) adders.

The RNS has been widely investigated and used in Digital Signal Processing (DSP) applications [1], [2]. A set of L moduli, suppose $(m_1, m_2, \ldots, m_L)$, that are pairwise relative prime is used to define an RNS. Any integer X, with $0 \leq X < M$, where $M = m_1 \times m_2 \times \ldots \times m_L$, has a unique representation in the RNS given by the L-tuple of residues $X = (x_1, x_2, \ldots, x_L)$, where $x_i = X \bmod m_i$. A two operand RNS operation, suppose $*$, is defined as $(z_1, z_2, \ldots, z_L) = (x_1, x_2, \ldots, x_L) * (y_1, y_2, \ldots, y_L)$, where $z_i = (x_i * y_i) \bmod m_i$. In most RNS applications, $*$ is either addition, subtraction, or multiplication. According to the above, each residue can be computed independently of the others allowing fast data processing in L parallel independent channels.

The latency of an RNS operation depends on the latency of the slowest among the channels. The delay of an adder modulo m is greater when $m \neq 2^n$. Efficient designs for the RNS channels have been recently proposed in [3], [4], [5] for m of the form $2^n - 1$. The authors of [3], [4], respectively, propose Carry Look-Ahead (CLA) and parallel-prefix design methodologies for modulo $2^n - 1$ adders that lead to implementations that can operate as fast as modulo 2^n adders. Modulo $2^n - 1$ modified Booth multipliers that can operate as fast as the corresponding integer multipliers were introduced in [5]. According to the above, RNS adders based on moduli of the form $2^n, 2^n - 1, 2^k - 1, 2^l - 1, \ldots$, with $l < k < n$, lead to the fastest implementations. For this reason, we focus on RNS with moduli of this form. In most practical cases, the choice $L = 3$ and $k = n - 1$ is preferred. Examples of such systems using earlier design methodologies for the modulo $2^n - 1$ channel have been presented in [6], [7], [8]. Efficient residue to binary converters in the $L = 3$ and $k = n - 1$ case, have been proposed in [9], [10].

Formal test sets for CLA integer adders have been presented in [11], [12]. In [11], the authors show that $2 \times (n + 1)$ vectors are sufficient for testing a simple n-bit CLA inclusive-OR adder. The author of [12] derives test sets for exclusive-OR CLA and block-CLA adders. Inclusive-OR CLA adders are faster than exclusive-OR CLA adders.

In this paper, we derive a formal test set for modulo 2^n inclusive-OR CLA and parallel prefix adders, consisting of $2 \times n$ vectors, and, for the first time in the open literature, a formal test set for modulo $2^n - 1$ full CLA and parallel-prefix inclusive-OR adders. We show that extracting from the latter test set a subset of k bit positions $(k < n)$ in order,

- *H.T. Vergos, D. Nikolos, and M. Bellos are with the Computer Engineering and Informatics Department, University of Patras, 26 500 Greece. H.T. Vergos and D. Nikolos are also with the Computer Technology Institute, 3 Kolokotroni St., 26 221 Patras, Greece. E-mail: {vergos, bellos}@ceid.upatras.gr, nikolosd@cti.gr.*
- *C. Efstathiou is with the Informatics Department, TEI of Athens, Ag. Spyridonos St., 12 210 Egaleo, Greece. E-mail: cefsta@teiath.gr.*

Manuscript received 15 Nov. 2001; accepted 17 Sept. 2002.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number 115377.

will form a reduced width test set for testing a modulo $2^k - 1$ or a modulo 2^k adder. Therefore, the test set of a modulo $2^n - 1$ adder is a superset of the test set of any modulo $2^k - 1, 2^l - 1, \ldots,$ adder with $l < k < n$. By merging the test sets derived for modulo 2^n and modulo $2^n - 1$ adders, we get a formal test set for an RNS adder based on the moduli $\langle 2^n, 2^n - 1, 2^k - 1, 2^l - 1, \ldots \rangle$, consisting of $n^2 + 2$ vectors. Our test sets, apart from being composed of a small $(O(n^2))$ number of vectors, also have the advantage that they are derived based on the adder's equations and are parameterized with respect to the size of the adder; therefore, they are reusable. A designer can use them, irrespective of the implementation library that he targets or the size of the operands.

We also present a novel Built-In Self-Test (BIST) scheme that can apply the formal test set in $O(n^2)$ cycles and compare it against other BIST alternatives. Static CMOS implementations are utilized for comparing the proposed BIST scheme against Linear Feedback Shift Registers (LFSR)-based pseudorandom BIST as well as against ROM or Finite State Machine (FSM)-based deterministic BIST solutions. The attained results reveal that the proposed BIST offers 100 percent postcompaction fault coverage and an attractive combination of test application time and implementation area.

2 Preliminaries

Let $A = a_{n-1}a_{n-2}\ldots a_1a_0$ and $B = b_{n-1}b_{n-2}\ldots b_1b_0$ be two n-bit numbers. For computing their sum, $S = s_{n-1}s_{n-2}\ldots s_1s_0$, one needs to compute the carries $c_{n-1}, c_{n-2}, \ldots, c_0$ since the sum at position i is given by the relation $s_i = a_i \oplus b_i \oplus c_{i-1}$. Note that c_{-1} is the input carry and c_{n-1} the output carry.

To speed up the addition operation, the carry computation time should be minimized. To this end, carry look-ahead (CLA) adders [13], [14] are used. Since the circuit required by the carry look-ahead logic grows rapidly with the operand width, it is quite profitable to divide the carry computation unit into smaller units. The carry outputs of these smaller units can then either ripple between them or be driven to a CLA unit of a subsequent level, leading to two or more level CLA adders. If carry computation is treated as a prefix-problem, a special form of multilevel CLA adders can be derived, which are well-known as parallel-prefix adders.

When dealing with CLA adders, two functions are commonly used for describing the equations for the carries:

- $g_i = a_i \cdot b_i$, the carry generate function, and
- p_i, the carry propagate function.

The propagate function can be defined in two alternative ways. Its definition affects the adder's testability [11], [12]. In inclusive-OR CLA adders, p_i is defined as $p_i = a_i + b_i$ and another function, suppose $h_i = a_i \oplus b_i$, is commonly used to denote the half-sum at bit position i. Obviously, $s_i = h_i \oplus c_{i-1}$. In exclusive-OR CLA adders, $p_i = a_i \oplus b_i$ and $s_i = p_i \oplus c_{i-1}$. Since, in current CMOS technology, an OR gate is faster than an XOR gate, inclusive-OR CLA adders are faster than the corresponding exclusive-OR ones. Therefore, in this paper, we consider inclusive-OR CLA adders. In an integer adder, the carry at bit position i is given by:

$$c_i = g_i + \sum_{j=0}^{i-1} \left(\prod_{k=j+1}^{i} p_k \right) g_j + \left(\prod_{k=0}^{i} p_k \right) c_{-1}, \text{ for } \leq i \leq n-1.$$

Several fault models have been proposed for representing the actual faults of CMOS integrated circuits. The most common, however, is still the single stuck-at fault model because of its effectiveness and its simplicity. In this paper, for every cell, we exercise the logical paths from their inputs to their outputs and propagate the fault effects to the primary outputs of the circuit, which is equivalent to stuck-at fault test generation [12]. A library containing just the most primitive gates: inverters, 2 input AND, NAND, OR, and NOR gates is assumed. More complex gates are replaced by equivalent circuits built by the library elements. For example, although three tests are adequate for covering a 2-input XOR gate, by the above assumptions, the exhaustive four vector test is required. Note also that, by the above assumptions, no changes are required to the extracted test patterns if a library that includes multiple-input versions of the primitive elements is assumed [11].

For describing our test patterns, we use the notation introduced in [12] which utilizes carry propagation and generation across multiple adjacent bit positions. The notation is restated here for the sake of completeness:

G^l is a test that generates a carry out of l adjacent pairs of bits.

P^l is a test that permits carry propagation across l adjacent pairs of bits without generating a carry out.

O^l denotes a test that neither permits carry propagation nor carry generation out of l adjacent pairs of bits.

X^l denotes "don't care."

$\overline{P^l}$ denotes not P^l (equivalently, O^l for the inclusive-OR adder case).

$\overline{G^l}$ denotes not G^l (equivalently, P^l or O^l).

A complete test vector is written as a product list with the most significant bits at the left. For example, $(a_3, b_3, a_2, b_2, a_1, b_1, a_0, b_0,) = (1, 0, 1, 1, 0, 1, 0, 1)$ is denoted as $[P \cdot G \cdot P \cdot P]$ or, equivalently, $[P \cdot G \cdot P^2]$. The notation P_a and P_b is used for distinguishing between the cases $(a, b) = (1, 0)$ and (0, 1), respectively. The existence of carry-ins and carry-outs in test vectors, wherever they are needed, are handled by the + and – symbols as follows:

T_+ is a test that requires a carry-in.

T_- is a test that requires not to have a carry-in.

^+T is a test that produces a carry-out.

^-T is a test that does not produce a carry-out.

A test set consisting of k vectors is denoted by:

$$\left| \begin{matrix} T_1 \\ T_2 \\ \ldots \\ T_k \end{matrix} \right|.$$

3 A Formal Test Set for RNS Adders

3.1 Modulo 2^n Inclusive-OR Adder Test Set Derivation

We consider single-level CLA or parallel-prefix adder implementations [13], [14]. A modulo 2^n adder differs from an integer adder at the following:

1. A carry input signal is not present, hence p_0 is not required.
2. A carry output signal and the logic required for its production is not required, hence p_{n-1} and g_{n-1} are not required.

Since a modulo 2^n adder does not have a carry-in signal, the test sets derived for inclusive-OR integer adders in [11], [12] cannot be straightforwardly applied. However, we can follow a similar procedure in order to derive a new test set for modulo 2^n adders. In a modulo 2^n adder, the carry at bit position i is given by

$$c_i = g_i + \sum_{j=0}^{i-1} \left(\prod_{k=j+1}^{i} p_k \right) g_j,$$

for $0 \leq i \leq n-2$.

For testing the half sum functions, since they consist of XOR gates, we need to apply all four input combinations for every pair of bits. The propagation of possible fault effects on them, to the adder's outputs, is not a problem since they are connected to the XOR gates at the adder's outputs and the output of an XOR gate changes whenever either of its inputs change. According to this, the test set below should be applied for every pair of bits:

$$\left| \begin{matrix} O \\ P_a \\ P_b \\ G \end{matrix} \right|. \tag{1}$$

The generate functions are implemented by AND gates, which require that, at each pair of input bits, the following tests are applied:

$$\left| \begin{matrix} P_a \\ P_b \\ G \end{matrix} \right|.$$

Fault effects from these tests must be propagated through the carry computation functions. One way to achieve this is to propagate a fault effect on g_i to c_i, by forcing the other product terms in the function of c_i to zero. This, however, cannot be achieved by controlling p_i, but by ensuring that there is no carry-in at the ith bit position. For example, the equation for c_2 is given by $c_2 = g_2 + p_2 \cdot g_1 + p_2 \cdot p_1 \cdot g_0$. For propagating fault effects of g_2 to c_2 (equivalently to s_3), the terms $p_2 \cdot g_1$ and $p_2 \cdot p_1 \cdot g_0$ should be at zero. Since p_2 cannot be set to zero independently of g_2, we require that both g_1 and $p_1 \cdot g_0$ are at zero. Note that, since $g_0 = c_0$, fault effects at g_0 do not cause any propagation problems. In our notation, the above mean that each bit position $1 \leq i \leq (n-2)$ needs the tests (the tests for bit position 0 have already been considered in (1)):

$$\left| \begin{matrix} P_{a-} \\ P_{b-} \\ G_- \end{matrix} \right|. \tag{2}$$

In a similar way, the p_i functions require the tests $[O]$, $[P_a]$, and $[P_b]$, but, for propagating fault effects on them to c_i, a carry-in is required. So, the following tests for each bit position i, $1 \leq i \leq (n-2)$, are required:

$$\left| \begin{matrix} O_+ \\ P_{a+} \\ P_{b+} \end{matrix} \right|. \tag{3}$$

For showing how the required tests for the carry functions can be derived, we assume a modulo 32 adder. The most significant carry function in this adder is given by $c_3 = g_3 + p_3 \cdot g_2 + p_3 \cdot p_2 \cdot g_1 + p_3 \cdot p_2 \cdot p_1 \cdot g_0$. For testing c_3, we need to test each product term and propagate possible faults on them. We present below the required tests for this in terms of the inputs at the four least significant bit pairs of the adder:

- g_3 requires the application of $[G \cdot X^3]$ and $\left[\overline{G} \cdot X^3\right]$. In order to propagate the fault effects to c_3, the other terms should be set to 0 by the vectors $\left[\overline{P} \cdot X^3\right]$ or $[P \cdot {}^{-}X^3]$. Merging both the sensitization and the propagation conditions yields the complete test for the g_3 term:

$$\left| \begin{matrix} G \cdot {}^{-}X^3 \\ [O \cdot X^3] \; or \; [P \cdot {}^{-}X^3] \end{matrix} \right|,$$

- The term $p_3 \cdot g_2$ requires the application of $[P \cdot G \cdot X^2]$, $\left[P \cdot \overline{G} \cdot X^2\right]$, and $\left[\overline{P} \cdot G \cdot X^2\right]$. In order to propagate the fault effects, the other terms should be 0. Setting g_3 to 0 requires $\left[\overline{G} \cdot X^3\right]$, while setting $p_3 \cdot p_2 \cdot g_1$ and $p_3 \cdot p_2 \cdot p_1 \cdot g_0$ to 0 requires $\left[\overline{P^2} \cdot X^3\right]$ or $[P^2 \cdot {}^{-}X^2]$. So, a complete test for the term $p_3 \cdot g_2$ is:

$$\left| \begin{matrix} [P \cdot G \cdot {}^{-}X^2] \\ [P \cdot O \cdot X^2] \; or \; [P^2 \cdot {}^{-}X^2] \\ [O \cdot G \cdot X^2] \end{matrix} \right|,$$

- Working the same way, a complete test for the term $p_3 \cdot p_2 \cdot g_1$ is:

$$\left| \begin{matrix} [P \cdot P \cdot G \cdot \overline{G}] \\ [P \cdot P \cdot P \cdot \overline{G}] \; or \; [P \cdot P \cdot O \cdot X] \\ [P \cdot O \cdot G \cdot X] \\ [O \cdot P \cdot G \cdot X] \end{matrix} \right|,$$

and

- A complete test for the last term is:

$$\left| \begin{matrix} [P \cdot P \cdot P \cdot G] \\ [P \cdot P \cdot P \cdot \overline{G}] \\ [P \cdot P \cdot O \cdot G] \\ [P \cdot O \cdot P \cdot G] \; or \; [P \cdot O \cdot G \cdot G] \\ [O \cdot P \cdot P \cdot G] \end{matrix} \right|.$$

For finding a complete test set for c_3, one needs to combine the test sets derived above for each of its terms. Obviously,

all the second vectors of each term's test set can be combined into a single test $[P \cdot P \cdot P \cdot \overline{G}]$. Moreover, the last vector of each term's test set can be combined in a single vector, that is, $[O \cdot G \cdot G \cdot G]$. Therefore, a minimum test set for c_3 is:

$$\left|\begin{array}{c} [P \cdot P \cdot P \cdot G] \\ [P \cdot P \cdot G \cdot \bar{X}] \\ [P \cdot G \cdot \bar{X}^2] \\ [G \cdot \bar{X}^3] \\ \text{----------} \\ [P \cdot P \cdot O \cdot G] \\ [P \cdot O \cdot G \cdot G] \\ [O \cdot G \cdot G \cdot G] \\ \text{----------} \\ [P \cdot P \cdot P \cdot \overline{G}] \end{array}\right| \tag{4}$$

Following a similar procedure for c_2 and c_1, we can see that all the tests required for them are covered by the test set derived for c_3, which is therefore sufficient for all the carry functions of a modulo 32 adder. Moreover, since c_2 is the most significant carry of a modulo 16 adder, it becomes apparent that (4) is also a superset of the test set for the carry functions of a modulo 16 adder. In the general case of a modulo 2^n adder, all tests required for the carry functions are covered by the tests for c_{n-2}.

The pattern shown in (4) can be extended for deriving the test set required for testing the carry functions in a modulo 2^n adder which will have the form:

$$\left|\begin{array}{c} X \cdot P^{n-2} \cdot \overline{G} \\ \text{----------} \\ X \cdot P^{n-i} \cdot G \cdot \bar{X}^{i-1} \\ \text{----------} \\ X \cdot P^{n-2-i} \cdot O \cdot G^i \end{array}\right| \begin{array}{l} \\ \\ \rightarrow \text{for } \ 1 \le i \le (n-1) \\ \\ \rightarrow \text{for } \ 1 \le i \le (n-2) \end{array} \tag{5}$$

A complete test set for a modulo 2^n adder can be derived by merging the set (5) with (1), (2), and (3). One possible test set with $2 \times n$ vectors is:

$$\left|\begin{array}{c} P_a^n \\ P_b^n \\ \text{--------} \\ O \cdot P_a^{n-2} \cdot G \\ P_b^{n-1} \cdot G \\ G \cdot P_b^{n-3} \cdot G \cdot O \\ P_b^{n-1-i} \cdot G \cdot P_b^i \\ \text{--------} \\ P_b^{n-1-i} \cdot O \cdot G^i \end{array}\right| \begin{array}{l} \\ \\ \\ \\ \\ \rightarrow \text{for } \ 2 \le i \le (n\text{-}2) \\ \\ \rightarrow \text{for } \ 1 \le i \le (n\text{-}2) \end{array}$$

3.2 Modulo $2^n - 1$ Inclusive-OR Adder Test Set Derivation

As has been shown in [3] for CLA implementations and in [4] for parallel-prefix implementations, a modulo $2^n - 1$ CLA adder follows a structure similar to that of an integer adder, but with modified carry functions. More specifically, the carry function at bit position i, $-1 \le i \le (n-2)$, is given by [3]

$$c_i = g_{|n+i|_n} + \sum_{j=0}^{n-2} \left(\prod_{k=j+1}^{n-1} p_{|k+i+1|_n} \right) g_{|j+i+1|_n}.$$

In this paper, we consider inclusive-OR modulo $2^n - 1$ adders designed either with a single CLA level [3] or as parallel-prefix [4]. A modulo $2^n - 1$ adder has neither a carry input nor a carry output; however, the carry computed at the most significant bit position is XOR-ed with the half-sum at bit 0 for forming the least significant sum output.

The test sets given in (1), (2), and (3) are required at each pair of input bits both for sensitizing faults on, respectively, the half-sum, the generate, and the propagate functions, and for propagating possible fault effects through the carry computation circuitry. Consider now the case that $n = 4$. In this case, the carries in the modulo 15 adder are given by:

$$\begin{aligned} c_{-1} &= g_3 + p_3 \cdot g_2 + p_3 \cdot p_2 \cdot g_1 + p_3 \cdot p_2 \cdot p_1 \cdot g_0 \\ c_0 &= g_0 + p_0 \cdot g_3 + p_0 \cdot p_3 \cdot g_2 + p_0 \cdot p_3 \cdot p_2 \cdot g_1 \\ c_1 &= g_1 + p_1 \cdot g_0 + p_1 \cdot p_0 \cdot g_3 + p_1 \cdot p_0 \cdot p_3 \cdot g_2 \\ c_2 &= g_2 + p_2 \cdot g_1 + p_2 \cdot p_1 \cdot g_0 + p_2 \cdot p_1 \cdot p_0 \cdot g_3. \end{aligned}$$

We can observe that:

- The equations are derived in a cyclic manner, therefore, if one knows a test set for c_k, a test set for any of the rest of the carry functions can be devised by left/right rotations of every vector in the test set of c_k.
- In a modulo $2^n - 1$ adder, the carry function at each bit i has a form similar to that of the most significant carry function of a modulo 2^{n+1} adder. In the above example, this means that each carry of the modulo 15 adder has a similar form as c_3 of the modulo 32 adder.

By the above, utilizing test set (4), we can derive the test sets required for each carry function:

$$\left|\begin{array}{l}[P\cdot P\cdot P\cdot \overline{G}]\\ \hline [P\cdot P\cdot P\cdot G]\\ [P\cdot P\cdot G\cdot X]\\ [P\cdot G\cdot X\cdot X]\\ [G\cdot X\cdot X\cdot X]\\ \hline [P\cdot P\cdot O\cdot G]\\ [P\cdot O\cdot G\cdot G]\\ \hline [O\cdot G\cdot G\cdot G]\end{array}\right| \text{for } c_{-1}, \quad \left|\begin{array}{l}[P\cdot P\cdot \overline{G}\cdot P]\\ \hline [P\cdot P\cdot G\cdot P]\\ [P\cdot G\cdot X\cdot P]\\ [G\cdot X\cdot X\cdot P]\\ [X\cdot X\cdot X\cdot G]\\ \hline [P\cdot O\cdot G\cdot P]\\ [O\cdot G\cdot G\cdot P]\\ \hline [G\cdot G\cdot G\cdot O]\end{array}\right| \text{for } c_0,$$

$$\left|\begin{array}{l}[P\cdot \overline{G}\cdot P\cdot P]\\ \hline [P\cdot G\cdot P\cdot P]\\ [G\cdot X\cdot P\cdot P]\\ [X\cdot X\cdot P\cdot G]\\ [X\cdot X\cdot G\cdot X]\\ \hline [O\cdot G\cdot P\cdot P]\\ [G\cdot G\cdot P\cdot O]\\ \hline [G\cdot G\cdot O\cdot G]\end{array}\right| \text{for } c_1 \text{ and } \left|\begin{array}{l}[\overline{G}\cdot P\cdot P\cdot P]\\ \hline [G\cdot P\cdot P\cdot P]\\ [X\cdot P\cdot P\cdot G]\\ [X\cdot P\cdot G\cdot X]\\ [X\cdot G\cdot X\cdot X]\\ \hline [G\cdot P\cdot P\cdot O]\\ [G\cdot P\cdot O\cdot G]\\ \hline [G\cdot O\cdot G\cdot G]\end{array}\right| \text{for } c_2.$$

The above four test sets can be merged into a single test set for all the carry functions as follows:

- The vector $[P\cdot P\cdot P\cdot P]$ is used to replace all first vectors.
- In the second group of vectors, all "don't cares" are assumed as P and therefore only the vectors $[P\cdot P\cdot P\cdot G]$, $[P\cdot P\cdot G\cdot P]$, $[P\cdot G\cdot P\cdot P]$, and $[G\cdot P\cdot P\cdot P]$ are required.

According to the above, a test set for all carry functions of a modulo 15 adder is:

$$\left|\begin{array}{l}[P\cdot P\cdot P\cdot P]\\ \hline [P\cdot P\cdot P\cdot G]\\ [P\cdot P\cdot G\cdot P]\\ [P\cdot G\cdot P\cdot P]\\ [G\cdot P\cdot P\cdot P]\\ \hline [O\cdot G\cdot P\cdot P]\\ [P\cdot O\cdot G\cdot P]\\ [P\cdot P\cdot O\cdot G]\\ [G\cdot P\cdot P\cdot O]\\ \hline [O\cdot G\cdot G\cdot P]\\ [P\cdot O\cdot G\cdot G]\\ [G\cdot P\cdot O\cdot G]\\ [G\cdot G\cdot P\cdot O]\\ \hline [O\cdot G\cdot G\cdot G]\\ [G\cdot O\cdot G\cdot G]\\ [G\cdot G\cdot O\cdot G]\\ [G\cdot G\cdot G\cdot O]\end{array}\right| \tag{6}$$

By extrapolating the above pattern, it is possible to derive the test set required for testing the carry functions of any modulo 2^n-1 adder, which will have the form:

$$\left|\begin{array}{c}P^n\\ \hline P^{n-1}\cdot G\\ \hline O\cdot G^i\cdot P^{n-1-i}\end{array}\right| \begin{array}{l}\\ \rightarrow \text{and all possible rotations of it}\\ \\ \rightarrow \text{for } 1\le i\le n-1 \text{ and all possible rotations of it.}\end{array} \tag{7}$$

The latter shows that n^2+1 vectors are required for testing the carry computation unit of a modulo 2^n-1 adder. The cardinality of the derived test set shows that a modulo 2^n-1 adder compared to a modulo 2^n adder or an integer adder is a far harder to test circuit. By the way that the above vectors were derived, it is also clear that (7) is a superset of the test set required for testing the carry function at the most significant bit of a modulo 32 adder. In general, a test set for the carry functions of a modulo 2^n-1 adder is also a test set of the carry functions of a modulo 2^{n+1} adder which, according to the discussion of the previous subsection, is a superset of the test set for the carry functions of a modulo 2^n adder. Moreover, by observing the form of the vectors of (7), we can see that extracting any subset of k bit positions in order will lead to a reduced width test for the carry functions of a modulo 2^k-1 adder, with $k<n$.

3.3 Formal Test Set for an RNS Adder

A formal test set for testing an RNS adder for the moduli set $2^n, 2^n-1, 2^k-1, 2^l-1, \ldots$, with $l<k<n$, according to the discussion in the previous subsection, can be obtained by merging the test sets indicated in (1), (2), (3), and (7). One possibility that leads to n^2+2 vectors is:

$$\left|\begin{array}{l}P_a^n\\ \hline P_a^{n-1}\cdot G\\ P_b^{n-1}\cdot G\\ P_b^{n-2}\cdot G\cdot P_a\\ P_b^{n-1-i}\cdot G\cdot P_b^i\\ \hline O\cdot G^i\cdot P_b^{n-1-i}\end{array}\right| \begin{array}{l}\\ \\ \\ \text{for } 2\le i\le (n\text{-}1)\\ \\ \text{for } 1\le i\le (n\text{-}1) \text{ and all possible rotations of it}\end{array} \tag{8}$$

Example 1. Consider an RNS adder based on the moduli $\langle 2^4, 2^4-1, 2^3-1\rangle$. Suppose that $x_3x_2x_1x_0$ and $y_3y_2y_1y_0$, $w_3w_2w_1w_0$, and $z_3z_2z_1z_0$ and $g_2g_1g_0$ and $f_2f_1f_0$ are the operands of the modulo 2^4, the modulo 2^4-1, and the modulo 2^3-1 addition channels, respectively. Then, a common test for parallel testing of the above addition channels is given in Table 1. The gray shaded vectors of Table 1 are not required for testing the modulo 2^3-1 adder.

4 Proposed BIST Scheme

In this section, we introduce a test-per-clock BIST scheme that applies the test set derived earlier for testing all the channels of the RNS adder. Scalable test generators for the case of inclusive-OR CLA integer adders have been presented in [15]. However, the generators of [15] cannot be used in our case since they are capable of producing

TABLE 1
Test Set for the RNS Adder Defined by the Moduli $\langle 2^4, 2^4-1, 2^3-1\rangle$

x_3, w_3	y_3, z_3	x_2, w_2, g_2	y_2, z_2, f_2	x_1, w_1, g_1	y_1, z_1, f_1	x_0, w_0, g_0	y_0, z_0, f_0
1	0	1	0	1	0	1	0
1	0	1	0	1	0	1	1
0	1	0	1	0	1	1	1
0	1	0	1	1	1	1	0
0	1	1	1	0	1	0	1
1	1	0	1	0	1	0	1
0	0	1	1	0	1	0	1
0	1	0	0	1	1	0	1
0	1	0	1	0	0	1	1
1	1	0	1	0	1	0	0
0	0	1	1	1	1	0	1
0	1	0	0	1	1	1	1
1	1	0	1	0	0	1	1
1	1	1	1	0	1	0	0
0	0	1	1	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	0	0	1	1
1	1	1	1	1	1	0	0

Shaded Vectors are not required for testing the 2^3-1 *adder*

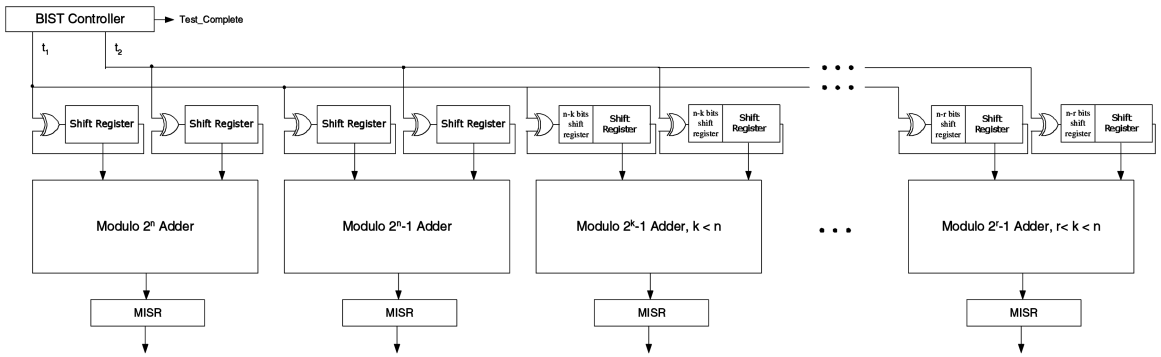

Fig. 1. RNS adder with the proposed BIST scheme.

$O(n)$ vectors, whereas, in our case, we need $O(n^2)$ vectors, for testing the modulo 2^n-1 adder. Fig. 1 presents a block diagram of the proposed test-per-clock BIST scheme.

The proposed BIST circuitry is composed of the following:

1. Modifications of the input buffers of the adders in order to transform them, in test mode, into shift registers. During test mode, these registers only perform a one bit right rotation. $n-k$ flip-flops are added at each input buffer of each modulo 2^k-1 adder, with $k<n$, in order to form an n-bit shift register. In the most common case, an RNS adder consists of only three channels, a modulo 2^n adder, a modulo 2^n-1 adder, and a $2^{n-1}-1$ adder, implying that only two flip-flops need to be inserted in the most common case. When test mode is entered, the left register of each channel is initialized to 111...111 and the right register to 000...001.
2. A control module, that generates three signals, t_1, t_2, and Test_Complete. The first two signals are used to conditionally complement the value of the bit that is shifted back into each register. t_1 is used to occasionally toggle the bit that is shifted into the left register of each channel, whereas t_2 is used for the same reason and the right registers. The control module is composed of a test vector counter and combinational logic that decodes the states of the test vector counter and produces t_1, t_2, and Test_Complete. Signal t_1 should be 1 at the fourth, fifth, ..., $(n+2)\mathrm{th}$ cycles, at the $(2n+2)\mathrm{th}$ and the $(2n+3)\mathrm{th}$ cycles, and at the $k(n+1)\mathrm{th}$ cycle with $3\le k\le n$. t_2 should be 1 at the second, third, ..., $(n+2)\mathrm{th}$ cycles, at the $(2n+3)\mathrm{th}$ cycle, and at the $k(n+1)$ and $k(n+1)+1$ cycles with $3\le k\le n$. We observe that t_1 is at 1 in the vast majority of test cycles that t_2 is also at 1, so we conclude that some of the logic that decodes the test vector counter states is shared among the implementations of t_1 and t_2. Test_Complete is asserted at the end of the $(n^2+2n)\mathrm{th}$ cycle. The control module can be easily described in HDL and synthesized for different values of n.
3. Modifications of the adders' output buffers so as, in test mode, to behave as Multiple Input Signature Analyzers (MISRs).

TABLE 2
Test Sequence Produced by the Proposed BIST

Cycle Number	t_1	t_2	Left Register Value	Right Register Value	Vector Applied
1	0	0	1111…1111	0000…0001	$P_a^{n-1} \cdot G$
2	0	1	1111…1111	0000…0000	P_a^n
3	0	1	1111…1111	1000…0000	$G \cdot P_a^{n-1}$
4	1	1	0111…1111	1100…0000	$P_b \cdot G \cdot P_a^{n-2}$
…	1	1	…	…	$P_b^2 \cdot G \cdot P_a^{n-3} ... P_b^{n-3} \cdot G \cdot P_a^2$
n+1	1	1	0000 …0011	1111…1110	$P_b^{n-2} \cdot G \cdot P_a$
n+2	1	1	0000…0001	1111…1111	$P_b^{n-1} \cdot G$
n+3	0	0	1000…0000	1111…1111	$G \cdot P_b^{n-1}$
n+4	0	0	0100…0000	1111…1111	$P_b \cdot G \cdot P_b^{n-2}$
…	0	0	…	…	$P_b^2 \cdot G \cdot P_b^{n-3} ... P_b^{n-3} \cdot G \cdot P_b^2$
2n+1	0	0	0000…0010	1111…1111	$P_b^{n-2} \cdot G \cdot P_b$
2n+2	1	0	1000…0001	1111…1111	$G \cdot P_b^{n-2} \cdot G$
2n+3	1	1	0100…0000	0111…1111	$O \cdot G \cdot P_b^{n-2}$
2n+4	0	0	0010…0000	1011…1111	$P_b \cdot O \cdot G \cdot P_b^{n-3}$
…	0	0	…	…	$P_b^2 \cdot O \cdot G \cdot P_b^{n-4} ... P_b^{n-2} \cdot O \cdot G$
3n+2	0	0	1000…0000	1111…1110	$G \cdot P_b^{n-2} \cdot O$
3n+3	1	1	1100…0000	1111…1111	$G^2 \cdot P_b^{n-2}$
3n+4	0	1	0110…0000	0111…1111	$O \cdot G^2 \cdot P_b^{n-3}$
3n+5	0	0	0011…0000	1011…1111	$P_b \cdot O \cdot G^2 \cdot P_b^{n-4}$
…	0	0	…	…	$P_b^2 \cdot O \cdot G^2 \cdot P_b^{n-5} ... P_b^{n-3} \cdot O \cdot G^2$
4n+3	0	0	1100…0000	1111…1110	$G^2 \cdot P_b^{n-3} \cdot O$
4n+4	1	1	1110…0000	1111…1111	$G^3 \cdot P_b^{n-3}$
4n+5	0	1	0111…0000	0111…1111	$O \cdot G^3 \cdot P_b^{n-4}$
4n+6	0	0	0011…0000	1011…1111	$P_b \cdot O \cdot G^3 \cdot P_b^{n-5}$
…	0	0	…	…	$P_b^2 \cdot O \cdot G^3 \cdot P_b^{n-6} ... P_b^{n-4} \cdot O \cdot G^3$ & $O \cdot G^i \cdot P_b^{n-1-i}$ and all possible rotations of it for $4 \le i \le n-2$ ending at $G^{n-3} \cdot P_b \cdot O \cdot G$
n^2+n-1	0	0	1111…1100	1111…1110	$G^{n-2} \cdot P_b \cdot O$
n^2+n	1	1	1111…1110	1111…1111	$G^{n-1} \cdot P_b$
n^2+n+1	0	1	0111…1111	0111…1111	$O \cdot G^{n-1}$
n^2+n+2	0	0	1011…1111	1011…1111	$G \cdot O \cdot G^{n-2}$
…	0	0	…	…	$G^2 \cdot O \cdot G^{n-3} ... G^{n-2} \cdot O \cdot G$
n^2+2n	0	0	1111…1110	1111…1110	$G^{n-1} \cdot O$

Shaded vectors do not belong to the formal test set given by (8).

The vectors produced by the proposed BIST are summarized in Table 2. Comparing Table 2 and (8), it is obvious that the test vectors of (8) are included in the vectors produced by the proposed test pattern generator. The shaded vectors of Table 2 are vectors produced by the proposed BIST scheme that do not belong to the set defined by (8). The required n^2+2 vectors are applied by the proposed BIST scheme in n^2+2n cycles. A counter of $\log_2(n^2+2n)$ bits is therefore required in the BIST control module of Fig. 1.

Example 2. Consider the proposed BIST for a three channel RNS adder based on the moduli 2^4, 2^4-1, and 2^3-1. Table 3 presents the input buffers' contents along with the t_1 and t_2 signals' values. Comparing Table 3 and Table 1, one can easily verify that the vectors required for testing the modulo 16, 15, and 7 adders are well within the test vectors provided by the BIST scheme given in

TABLE 3
Test Patterns Generated by the Proposed BIST for Example 2

Cycle Number	t_1	t_2	Modulo 16 Adder		Modulo 15 Adder		Modulo 7 Adder	
			Operand 1	Operand 2	Operand 1	Operand 2	Operand 1	Operand 2
1	0	0	1111	0001	1111	0001	111	001
2	0	1	1111	0000	1111	0000	111	000
3	0	1	1111	1000	1111	1000	111	000
4	1	1	0111	1100	0111	1100	111	100
5	1	1	0011	1110	0011	1110	011	110
6	1	1	0001	1111	0001	1111	001	111
7	0	0	1000	1111	1000	1111	000	111
8	0	0	0100	1111	0100	1111	100	111
9	0	0	0010	1111	0010	1111	010	111
10	1	0	1001	1111	1001	1111	001	111
11	1	1	0100	0111	0100	0111	100	111
12	0	0	0010	1011	0010	1011	010	011
13	0	0	0001	1101	0001	1101	001	101
14	0	0	1000	1110	1000	1110	000	110
15	1	1	1100	1111	1100	1111	100	111
16	0	1	0110	0111	0110	0111	110	111
17	0	0	0011	1011	0011	1011	011	011
18	0	0	1001	1101	1001	1101	001	101
19	0	0	1100	1110	1100	1110	100	110
20	1	1	1110	1111	1110	1111	110	111
21	0	1	0111	0111	0111	0111	111	111
22	0	0	1011	1011	1011	1011	011	011
23	0	0	1101	1101	1101	1101	101	101
24	0	0	1110	1110	1110	1110	110	110

Shaded vectors do not belong to the formal test set given by (8)

Fig. 1. The gray shaded vectors of Table 3 are those vectors produced by the proposed BIST that do not belong to the formal test set indicated by (8).

5 BIST Evaluation and Comparisons.

In this section, we compare the proposed BIST against ROM and FSM-based BIST schemes as well as against pseudorandom LFSR-based BIST schemes.

For embedding the test set provided by an ATPG tool, a designer may either design a TPG as a finite state machine or store the test patterns in an embedded ROM and successively retrieve them using a ROM address counter. We will denote these alternatives as FSM_BIST and ROM_BIST, respectively.

In the pseudorandom LFSR-based BIST schemes, we consider that the input buffers of each addition channel are modified to function, in test mode, as a single distinct LFSR. Two distinct cases are investigated regarding the test completion signal:

1. The test is complete when the channel that requires the largest number of LFSR states for achieving 100 percent precompaction fault coverage has received all required test vectors (we will hereafter refer to this scenario as *Single_Check*) and
2. Each channel's LFSR and MISR freezes when all states required for 100 percent fault coverage before compaction have appeared. We will hereafter refer to this scenario as *Distinct_Checks*. This scheme has less energy consumption than the first one.

In order to find a seed for each LFSR capable of ensuring short pseudorandom sequence for achieving 100 percent fault coverage we used the optimization procedure described in [16]. One hundred percent precompaction fault coverage was targeted.

For our comparisons, we use as metrics:

1. the area overhead imposed by each BIST scheme,
2. the postcompaction fault coverage (PCFC) attained, and
3. the test application time in number of test vectors.

Since an RNS adder is usually embedded in a larger circuit, its inputs and outputs are not accessible by the primary inputs and outputs of the chip. For applying a test set in such embedded circuits, one usually relies on scan paths. Therefore, for evaluating the hardware overheads imposed by the different BIST schemes, we use as a basis a scheme in which the flip-flops of the input and output registers of the RNS adder are chained together in a single scan path.

We examined three different RNS adders. These are defined, respectively, by the moduli $\langle 2^8, 2^8-1, 2^7-1\rangle$, $\langle 2^{16}, 2^{16}-1, 2^{15}-1\rangle$, and $\langle 2^{32}, 2^{32}-1, 2^{31}-1\rangle$. We will hereafter refer to them as the $\langle 2^8, 2^8-1, 2^7-1\rangle$, the $\langle 2^{16}, 2^{16}-1, 2^{15}-1\rangle$, and the $\langle 2^{32}, 2^{32}-1, 2^{31}-1\rangle$ RNS adder, respectively.

For getting realistic measures of the area overhead, we described the examined RNS adders in HDL and used the Synopsys® tools driven by the UMC-VST 25 implementation technology (0.25 μm, up to 5-metal layers, 1.8/3.3 V) for our implementations. Our targeted operating frequency was set to 200 MHz, for typical process parameters and

TABLE 4
Area Comparisons

RNS Adder	*FSM_BIST*	*ROM_BIST*	*Single_Check LFSR based*	*Distinct_Checks LFSR based*	*Proposed BIST*
$\langle 2^8,\ 2^8-1,\ 2^7-1\rangle$	112.2 %	54.2 %	25.6 %	33.2 %	30.3 %
$\langle 2^{16},\ 2^{16}-1,\ 2^{15}-1\rangle$	148.5 %	59.4 %	13.9 %	24.8 %	17.4 %
$\langle 2^{32},\ 2^{32}-1,\ 2^{31}-1\rangle$	228.3 %	78.0 %	12.7 %	22.7 %	15.4 %

TABLE 5
Fault Coverage and Test Times

	FSM_BIST / ROM_BIST	*Single_Check LFSR based*	*Distinct_Checks LFSR based*	*Proposed BIST*
$\langle 2^8,\ 2^8-1,\ 2^7-1\rangle$ *Adder*				
PCFC (%)	99.92	100	100	100
Test Cycles	(31, 31, 31)	(67, 67, 67)	(26, 63, 67)	(80, 80, 80)
$\langle 2^{16},\ 2^{16}-1,\ 2^{15}-1\rangle$ *Adder*				
PCFC (%)	(100, 100, 100)	(100, 100, 100)	(100, 100, 100)	(100, 100, 100)
Test Cycles	(55, 55, 55)	(564, 564, 564)	(98, 564, 168)	(288, 288, 288)
$\langle 2^{32},\ 2^{32}-1,\ 2^{31}-1\rangle$ *Adder*				
PCFC (%)	(100, 100, 99.97)	(100, 100, 100)	(100, 100, 100)	(100, 100, 100)
Test Cycles	(98, 98, 98)	(293404, 293404, 293404)	(2835, 229641, 293404)	(1088, 1088, 1088)

irrespective of the insertion or not of any Design-For-Testability (DFT) hardware.

Table 4 presents the area overhead of an RNS adder supported by the examined BIST schemes as a percentage of the implementation area of an RNS adder with a single scan path. In the case of ROM_BIST, we can implement the ROM either as a memory array (with one transistor per bit) or as a combinational circuit. Because of the small ROM sizes required in the examined adders, the latter approach gives the best area results and is indicated in Table 4. As we can see from Table 4, both FSM_BIST and ROM_BIST require excessive implementation areas. In all examined cases, the hardware required for implementing FSM_BIST is larger than the RNS adder itself. Although smaller, the implementation area required by ROM_BIST is also very large. The required area for implementing ROM_BIST in the $\langle 2^{32}, 2^{32}-1, 2^{31}-1\rangle$ adder case raises to 78 percent of an RNS adder with a single scan path, compared to 15.4 percent of the proposed BIST. The values in Table 4 reveal that the proposed BIST scheme requires an implementation area similar to that of the LFSR-based approaches for all examined RNS adders.

In order to measure the PCFC, we used a custom developed fault simulator. In the $\langle 2^{16}, 2^{16}-1, 2^{15}-1\rangle$ and $\langle 2^{32}, 2^{32}-1, 2^{31}-1\rangle$ adder cases, we assume that each channel's output register is transformed during test into a distinct MISR. Since, under this assumption, in the $\langle 2^8, 2^8-1, 2^7-1\rangle$ adder case, none of the considered schemes leads to complete 100 percent PCFC, we assume that, in this case, the three output registers of the RNS adder are converted into a single MISR, increasing in this way the degree of the primitive polynomial of the MISR. The results obtained for the PCFC and test length are presented in Table 5. Each entry in Table 5, excluding the PCFC percentage of the $\langle 2^8, 2^8-1, 2^7-1\rangle$ adder, is an ordered triplet, whose elements refer to the modulo 2^n, the modulo 2^n-1 and the modulo $2^{n-1}-1$ channel, respectively. The results of Table 5 show that all schemes, except the FSM_BIST/ROM_BIST schemes in the case of the $\langle 2^8, 2^8-1, 2^7-1\rangle$ adder, achieve 100 percent PCFC in all examined cases.

Table 5 also reveals that the test time required by the proposed BIST is significantly less than that required by pseudorandom BIST schemes. LFSR-based schemes require 95.83 percent more time than the proposed BIST to test the $\langle 2^{16}, 2^{16}-1, 2^{15}-1\rangle$ adder. This percentage grows to 26,967 percent when the $\langle 2^{32}, 2^{32}-1, 2^{31}-1\rangle$ adder is considered. In the case of LFSR-based schemes, a significant amount of time is also required for finding an initial seed for the LFSRs so as to minimize the number of states required for achieving 100 percent fault coverage. In the

case of the $\langle 2^{16}, 2^{16}-1, 2^{15}-1\rangle$ adder, for finding the three initial LFSR seeds, 31 minutes of CPU time (on a Pentium III, 500 MHz machine) were required. The CPU time required raised to more than eight hours in the case of the $\langle 2^{32}, 2^{32}-1, 2^{31}-1\rangle$ adder.

6 Conclusions

Rapid test-pattern generation contributes to meet the ever-shrinking time-to-market. To this end, in this paper, we derive a formal test set for RNS adders consisting of n^2+2 vectors. This set can be used for parallel testing of addition channels that use the moduli $2^n, 2^n-1, 2^k-1, 2^l-1, \ldots$, with $l<k<n$ as their base. The test set was derived based on the adder's equations and can therefore be equally well applied to full CLA and parallel-prefix implementations. Moreover, it is parameterized and independent of a specific implementation library; therefore, it is totally reusable.

A test-per-clock BIST scheme has also been proposed, that applies the derived test set in n^2+2n cycles. Experimental evidence on three benchmark RNS adders shows that the proposed BIST scheme requires an implementation area close to that of LFSR-based schemes, whereas it is far more efficient than the latter in terms of test application time. BIST solutions based on embedding the test set provided by Automatic Test Pattern Generation (ATPG) tools by using a ROM or by designing a TPG as an FSM require too much area for their implementation.

Acknowledgments

This research was supported by the Research Committee of Patras University within the framework of the K. Karatheodoris scholarships program and by the Public Benefit Foundation "Alexander S. Onassis" via its scholarships program. The fault simulator, along with several utilities on response compaction and on the LFSR seed optimization used in this paper, have been developed by D. Bakalis as a part of his MSc thesis. The authors wish to acknowledge his valuable contribution.

References

[1] V. Paliouras and T. Stouraitis, "Multifunction Architectures for RNS Processors," *IEEE Trans. Circuits and Systems-II: Analog and Digital Signal Processing,* vol. 46, no. 8, pp. 1041-1054, Aug. 1999.

[2] E.D. Di Claudio, F. Piazza, and G. Orlandi, "Fast Combinatorial RNS Processors for DSP Applications," *IEEE Trans. Computers,* vol. 44, no. 5, pp. 624-633, May 1995.

[3] C. Efstathiou, D. Nikolos, and J. Kalamatianos, "Area-Time Efficient Modulo 2^n-1 Adder Design," *IEEE Trans. Circuits and Systems-II,* vol. 41, no. 7, pp. 463-467, July 1994.

[4] L. Kalamboukas, D. Nikolos, C. Efstathiou, H.T. Vergos, and J. Kalamatianos, "High-Speed Parallel-Prefix Modulo 2^n-1 Adders," *IEEE Trans. Computers,* Special Issue on Computer Arithmetic, vol. 49, no. 7, pp. 673-680, July 2000.

[5] C. Efstathiou and H.T. Vergos, "Modified Booth 1's Complement and Modulo 2^n-1 Multipliers," *Proc. Seventh IEEE Int'l Conf. Electronics, Circuits & Systems (ICECS '2K),* vol. II, pp. 637-640, Dec. 2000.

[6] M.A. Soderstrand, "A High-Speed Low-Cost Recursive Digital Filter Using Residue Number Arithmetic," *Proc. IEEE,* vol. 65, pp. 1065-1067, July 1977.

[7] W.K. Jenkins, "A Highly Efficient Residue-Combinatorial Architecture for Digital Filters," *Proc. IEEE,* vol. 66, pp. 700-702, June 1978.

[8] W.K. Jenkins, "Recent Advances in Residue Number Techniques for Recursive Digital Filtering," *IEEE Trans. Acoustics, Speech, and Signal Processing,* vol. 27, pp. 19-30, Feb. 1979.

[9] A.A. Hiasat and H.S. Abdel-Aty-Zohdy, "Residue-to-Binary Arithmetic Converter for the Moduli Set $(2^k, 2^k-1, 2^{k-1}-1)$," *IEEE Trans. Circuits and Systems-II,* vol. 45, no. 2, pp. 204-209, Feb. 1998.

[10] W. Wang, M.N.S. Swamy, M.O. Ahmad, and Y. Wang, "A High-Speed Residue-to-Binary Converter for Three Moduli $(2^k, 2^k-1, 2^{k-1}-1)$ RNS and a Scheme for Its VLSI Implementation," *IEEE Trans. Circuits and Systems-II,* vol. 47, no. 12, pp. 1576-1581, Dec. 2000.

[11] M.J. Batek and J.P. Hayes, "Optimal Testing and Design of Adders," *VLSI Design,* vol. 1, no. 4, pp. 285-298, 1994.

[12] W.R. Moore, "Minimal C-Testable Tests for Block-CLA Adders," *Int'l J. Electronics,* vol. 85, no. 5, pp. 611-628, Nov. 1998.

[13] I. Koren, *Computer Arithmetic Algorithms.* Prentice Hall, 1993.

[14] K. Hwang, *Computer Arithmetic: Principles, Architecture and Design.* John Wiley & Sons, 1979.

[15] H. Al-Asaad, J.P. Hayes, and B.T. Murray, "Scalable Test Generators for High-Speed Datapath Circuits," *J. Electronic Testing: Theory and Applications,* vol. 12, nos. 1-2, pp. 111-125, Feb.-Apr. 1998.

[16] P. Girard et al., "Low Power Pseudo-Random BIST: On Selecting the LFSR Seed," *Proc. XIII Conf. Design of Circuits and Integrated Systems (DCIS),* 1998.

Haridimos T. Vergos received the Diploma and PhD degrees in computer engineering from the University of Patras in 1991 and 1996, respectively. During 1998, he worked on the development of the first IEEE 802.11 MAC processor for the Multimedia and Networking Group of Atmel Inc. He currently holds a lecturer position in the Department of Computer Engineering and Informatics at the University of Patras, Greece. Dr. Vergos has authored more than 30 scientific papers on computer arithmetic and architecture and on design for testability. He also holds one world-wide patent. He is a member of the IEEE, the Greek Computer Engineering Society, and the Technical Chamber of Greece.

Dimitris Nikolos received the BSc degree in physics, the MSc degree in electronics, and the PhD degree in computer science, all from the University of Athens. He is currently a professor in the Computer Engineering and Informatics Department of the University of Patras and head of the Technology and Computer Architecture Laboratory. He has served as program cochairman of five (1997-2001) IEEE International On-Line Testing Workshops. He also served on the program committees for the IEEE International Symposia on Defect and Fault Tolerance in VLSI Systems (1997-1999), for the Third and Fourth European Dependable Computing Conference, and for the DATE (2000-2003) Conferences. He was guest coeditor of the June 2002 special issue of the *Journal of Electronic Testing, Theory, and Applications* (*JETTA*) devoted to the 2001 IEEE International On-Line Testing Workshop. His main research interests are fault-tolerant computing, computer architecture, VLSI design, test, and design for testability. He has authored or coauthored more than 110 scientific papers and holds one world-wide patent. He also was co-recipient of the Best Paper Award for his work "Extending the Viability of I_{DDQ} Testing in the Deep Submicron Era" presented at the Third IEEE International Symposium on Quality Electronic Design (ISQED 2002). He is a member of the IEEE and the IEEE Computer Society.

Maciej Bellos received the Diploma in computer engineering in 1999 and the MSc degree in computer science and technology in 2001 from the Department of Computer Engineering and Informatics, University of Patras, Greece. He is currently pursuing the PhD degree. His research is focused on efficient IC testing and design for testability. He is a member of the Technical Chamber of Greece.

Costas Efstathiou received the BS degree in physics and the MS degree in electronics from the University of Athens and the PhD degree in computer science from the University of Thessaloniki. He is currently a professor in the Department of Informatics of the TEI of Athens, Greece. Previously, he worked at the Research Institute of the Hellenic Air Force and the Hellenic Telecommunications Organization. His research and teaching interests are in digital design and computer architecture with emphasis in computer arithmetic and fault-tolerant systems and in computer networks and telecommunications.