A CLASS OF EASILY PATH DELAY FAULT TESTABLE CIRCUITS

T. Haniotakis[1], *E. Kalligeros*[1], *D. Nikolos*[1,2], *G. Sidiropoulos*[1,2], *Y. Tsiatouhas*[3] & *H. T. Vergos*[1,2]

[1] Department of Computer Engineering and Informatics, University of Patras, 26 500, Greece
[2] Computer Technology Institute, 3 Kolokotroni Str, 262 21 Patras, Greece
[3] ISD S.A, 22, K. Varnali Str, 152 33 Maroussi, Athens, Greece

ABSTRACT

The number of physical paths in a carry save or modified Booth multiplier, as well as in a non restoring cellular array divider is prohibitively large for testing all paths for delay faults. Besides, neither all paths are robustly testable nor a basis consisting of SPP-HFRT paths exists.

In this paper we present sufficient modifications of the above mentioned circuits so that a basis consisting of SPP-HFRT paths to exist. The cardinality of the derived basis is very small. Also, hardware and delay overheads due to the modifications are respectively small and negligible.

I. INTRODUCTION

Imprecise delay modeling, the statistical variations of the parameters during the manufacturing process as well as the occurrence of physical defects in the integrated circuits result in chip malfunction at the desired speed. The path delay fault model [1] addresses distributed or accumulated delays due to the propagation through several lines, each affected by a delay defect. Two major problems are associated with path delay fault testing :

a) An excessively large number of physical paths needs to be tested. Usually it is not affordable to test all of them. For example the number of physical paths in a 32x32 Carry Save Array Multiplier is of the order of 10^{34} while in a non-restoring cellular array divider is of the order of 10^{324}.

b) Since the single fault assumption is not realistic for the path delay fault model (a single defect usually will affect a large number of paths), a robust test is usually required for detecting a path delay fault. However, for many circuits, a large number of path delay faults is not robustly testable.

A variety of path selection methods have been proposed [2-8] to alleviate problem a. above. The simplest delay-based approach is to select all paths whose calculated delay exceeds a specific threshold. However, the number of paths selected by this method is so large in general that all of the selected paths cannot be tested, especially in the case of optimized circuits [9]. The method given in [3] selects a set of paths such that for each interconnect *l* of a given circuit the set contains at least one path with the largest calculated delay among the paths through *l*. The number of the paths selected by this method is moderate. The method, recently proposed in [4], reduces the number of paths to be tested by judging which of two paths has the larger real delay. A common drawback of the above mentioned methods [3, 4] is that the selected paths may not be robustly testable (a large percentage of paths in a circuit is not robustly testable).

A number of functional approaches has also been proposed [5-8]. In those, path selection is achieved by excluding paths that do not have to be tested functionally, such as unsensitizable paths, robust-dependent paths etc. However, these approaches are not practical for large-sized circuits because computation time and the number of the selected paths are quite large.

It has been shown in [2] that by measuring the delays along a suitable very small set R of physical paths the propagation delays along any other path can be calculated (we will hereafter call such a set of paths a ***basis***). However, to be able to measure the propagation delay along the R paths they must be Single Path Propagating – Hazard Free Robustly Testable (SPP-HFRT) [2]. Unfortunately for most circuits, a basis consisting of SPP-HFRT paths does not exist.

Almost all contemporary general and special purpose processors include a high speed multiplier and often a divider circuit. Testing them for path delay faults is a very difficult task due to: a) their excessively large number of physical paths and b) all path delay faults are not robustly testable. In this paper, we focus on n x n carry-save (CSM) and modified Booth (MBM) multipliers as well as the Non – Restoring Cellular Array Dividers (NRCADs), originally introduced in [10]. The methods given in [3, 4] can not be applied in the case of a CSM, MBM or an NRCAD, because many of the longer paths are neither robustly testable nor non-robustly validatable. Also, a basis R, according to [2], consisting of SPP-HFRT paths does not exist for the above circuits.

In this paper modifications of the above circuits are proposed leading to a basis Δ consisting of SPP-HFRT paths. The delay overhead due to the modifications is negligible while the hardware overhead for practical size circuits is small. Under these modifications a basis of SPP-HFRT paths with small cardinality can be derived. However due to the prohibitively large number of paths

0-7803-5975-5/00/$10.00 ©2000 IEEE

of these circuits (for example the number of paths of the 32 x 32 MBM is equal to 2.4×10^{15}) it is impossible to calculate the delay along all paths in order to derive the maximum path delay. This problem can be overcome using the method proposed in [4] to determine a relatively small number of paths Π the propagation delay along which must be calculated in order the maximum path delay of the circuit under test to be derived.

It has been shown in [11] that the fact that the circuit functions correctly at a speed does not imply that it will also function correctly at a lower speed. If we can test all primitive path delay faults of a circuit at a speed and during test application no delay fault is detected then the circuit functions correctly at any lower speed [11]. We will show in Section II that a circuit which does not have a strong delay – verification test set may have a basis consisting of SPP-HFRT paths. In this case measuring the delays along the paths of the basis and calculating the propagation delay along all paths included in primitive faults we derive the maximum delay of the circuit (as well as the maximum speed). Then the circuit functions correctly for all lower speeds.

We consider the inputs and outputs of the multiplier / divider as primary inputs (PIs) and primary outputs (POs) of the chip. In the case that the circuit is embedded in a larger module, its inputs and outputs can easily be made accessible by the PIs and the POs of the module using, for example, the method proposed in [12].

II. SPP-HFRT PATHS AND DELAY-VERIFIABLE CIRCUITS

A two pattern test $T = \langle V_1, V_2 \rangle$ is said to be a robust delay test for a path P, for a rising or falling transition at the input of the path, if and only if, when P is faulty and test T is applied, the circuit output is different from the expected state at sampling time, independent of the delays along gate inputs not on P [13]. A robust test that propagates the fault effect through only a single path to an output in the circuit will be called a Single-Path Propagating Robust Test (SPP-RT) for that output. A robust test is said to be a Hazard-Free Robust Test (HFRT) if no hazards can occur on the tested path during the application of the test, regardless of the gate delay values. Robust tests may not exist for all path delay faults in an arbitrary circuit.

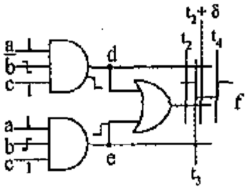


Figure 1.a

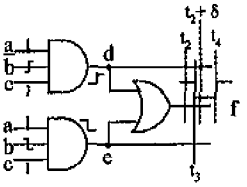


Figure 1.b

It has been shown in [11] that the fact that a circuit functions correctly at a speed does not imply that it will also function correctly at a lower speed. A set of path delay tests is called a strong delay–verification test set if the correct response of the CUT at a speed implies correct operation at any lower speed [11]. A circuit which has a strong delay-verification test set is called a delay-verifiable circuit [11]. Figure 1 [11] shows a circuit which does not have a strong delay-verification test set. All faults except $\uparrow\bar{b}df$, $\downarrow\bar{b}df$, $\uparrow\bar{b}ef$ and $\downarrow\bar{b}ef$ are testable by robust tests. In this case, even the exhaustive set consisting of all the vector pairs is not a strong delay verification test set. The signal values in the circuit for the two tests <101, 111> and <111, 101> are shown in Figures 1.a and 1.b, respectively. If there are no path delay faults, any output pulse that may occur will occur before the sampling time t_2. Faults on the paths from b and $\bar{b}$ may result in an output pulse occurring later. Such faults may or may not be detected at time t_2. Therefore, the correct response for these two tests only guarantees that the circuit will operate correctly if its period is set to the test period τ, but the delayed pulse due to the path delay fault may cause incorrect operation at a lower speed.

However, although a strong-delay verification test set does not exist for the circuit of Figure 1 under the definition given in [11], we will show below that we can calculate the maximum speed of the circuit and that for any lower speed the circuit will function correctly. The propagation delay along the paths ad, adf, ae, aef, $\bar{b}d$ and be which are SPP-HFRT can be measured applying the test vector pairs <001, 101>, <001, 101>, <011, 111>, <011, 111>, <101,111> and <101,111> respectively. Then the propagation delay, pd, along the paths $\bar{b}df$ and bef can be calculated by :

$pd(\uparrow\bar{b}df) = pd(\uparrow\bar{b}d) + pd(\uparrow adf) - pd(\uparrow ad)$,

$pd(\downarrow\bar{b}df) = pd(\downarrow\bar{b}d) + pd(\downarrow adf) - pd(\downarrow ad)$,

$pd(\uparrow bef) = pd(\uparrow be) + pd(\uparrow aef) - pd(\uparrow ae)$ and

$pd(\downarrow bef) = pd(\downarrow be) + pd(\downarrow aef) - pd(\downarrow ae)$.

For the output waveforms of figures 1.a and 1.b we get $pd(\uparrow\bar{b}df) = t_4$, $pd(\downarrow\bar{b}df) = t_3$, $pd(\uparrow bef) = t_4$, $pd(\downarrow bef) = t_3$, therefore the maximum delay of the circuit is equal to t_4 and for lower speeds the circuit will function correctly. From the above discussion it becomes evident that for a circuit having a basis consisting only of SPP-HFRT paths we can calculate the delay along all paths or along the paths included in primitive faults [11] and the calculated maximum propagation delay implies that the circuit will function correctly for any lower speed.

III. CARRY - SAVE MULTIPLIERS

An n x m CSM is a circuit with inputs (A_1, A_2, ... A_n) and (B_1, B_2, ... B_m) and outputs (O_1, O_2, ... O_{n+m}). Figure 2 presents the 4 x 4 carry save multiplier. We consider that the multiplier consists of two blocks. The first block D_0 consists of the network of carry save adders and the associated logic. The second block D_1 is an (n-1)-bit adder, which can be implemented as a ripple carry or group carry look ahead adder.

In [12] we have shown that using multiplexers for making the inputs and outputs of the embedded blocks accessible by the primary ports of the circuit, the path delay fault testing of the circuit is reduced to the path delay fault testing of the blocks that constitute it. By adding multiplexers in the original CSM design (Figure 3), we can manipulate the two blocks, D_0 and D_1, individually. We will hereafter deal only with the path delay fault testing of D_0, since efficient path delay fault testing techniques of both ripple-carry and carry look ahead implementations of D_1 have been presented in [14]. A basis consisting only of SPP-HFRT paths of D_0 does not exist. Hence some additional modifications of D_0 are required. Specifically, the half adders of D_0, that is, the adders of the first row, are substituted by full adders. The extra input of the leftmost adder, of the first row, is driven by a test input t_1. The extra inputs of the rest adders of the first row are driven alternately by the t_2 and t_3 test inputs. During normal circuit operation all three test inputs t_1, t_2 and t_3 are driven to 0.

The selection of the SPP-HFRT paths of a basis for the CSM after the above modifications was presented in [15].

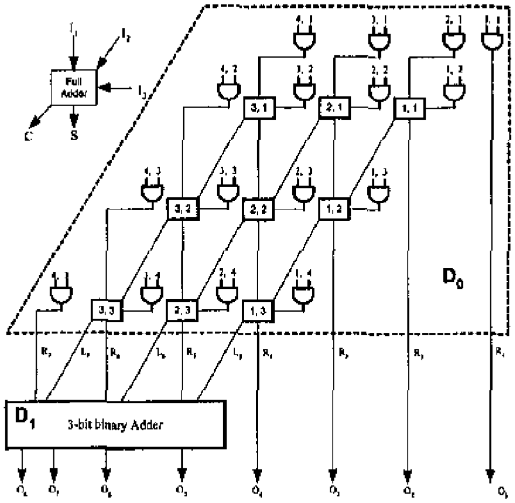


Figure 2.

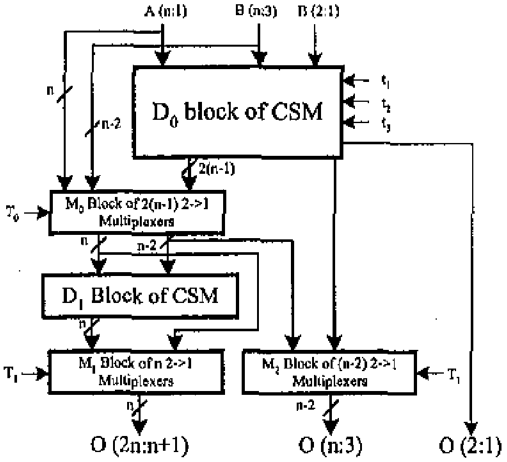


Figure 3.

IV. MODIFIED BOOTH MULTIPLIERS

We consider n x n modified Booth multipliers where n = 2^k, with sign generate. An n x n MBM is a combinational circuit with inputs (A_1, A_2, ..., A_n) and (B_1, B_2, ..., B_n) and outputs (P_1, P_2, ..., P_{2n}). Figure 4 presents the 8 x 8 MBM.

We also consider that the multiplier consists of two blocks. The first block D_0 contains the logic that forms and adds the partial products and consists of two parts :

a) The first part is responsible for the 2-bit recoding function, and is implemented by the n/2 cells at the left end, named r-cells. The implementation of an r-cell is presented in Figure 5.

b) The second part is responsible for the generation and addition of the partial sums and is implemented by:

b1) (n-1)*(n/2) cells, named ps-cells. The implementation of a ps-cell is presented in Figure 6.

b2) n/2 cells, named l_ps-cells. An l_ps-cell is the leftmost cell in a ps-cell row. It can be a normal ps-cell with an inverter at the output and its I_4 and I_5 inputs driven from the same signal (A_n). Since such modifications result in an non robustly testable l_ps-cell, in Figure 7 we present the implementation of a robustly testable l_ps-cell.

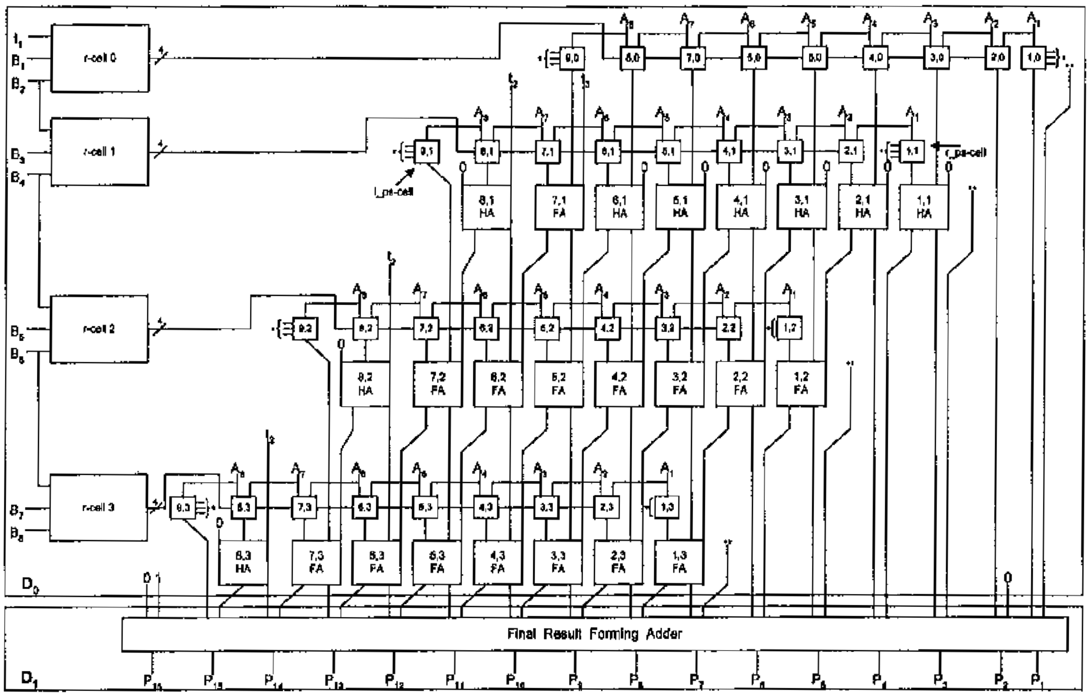
b3) n/2 cells, named r_ps-cells. An r_ps-cell is the rightmost cell in a ps-cell row. It can be designed as a normal ps-cell with its I_5 input connected to ground. This results in a non-robustly testable cell. A robustly testable implementation for an r_ps-cell is shown in Figure 8.

b4) (n-1)*[(n/2)-2)]+1 full adders. We consider every full adder implemented as in [14].

b5) n + (n / 2) – 3 half adders.

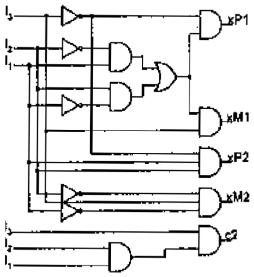
The second block D_1 is an (2*n)-bit adder which forms the final result. D_1 can be implemented as a ripple carry or group carry look ahead adder.

Using multiplexers for making the inputs and outputs of the embedded blocks controllable and observable



* : Receive the same inputs as the r-cell of the corresponding row,
** : Connected to the c2 output of the r-cell of the corresponding row

Figure 4. An 8 x 8 MBM



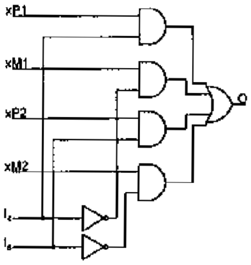


Figure 5.

Figure 6.

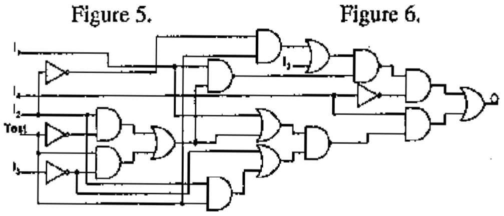


Figure 7. l_ps-cell implementation

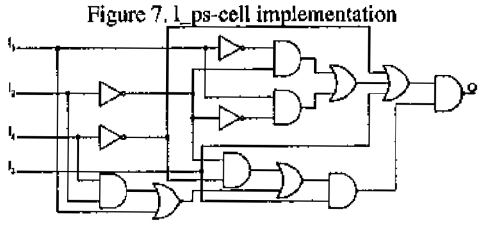


Figure 8. r_ps-cell implementation

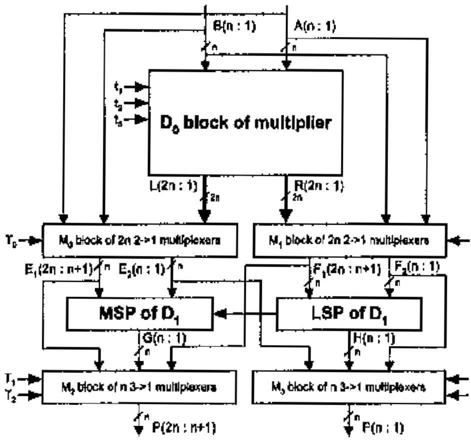


Figure 9. Modified n x n MBM

respectively by the primary ports of the circuit, the path delay fault testing of the circuit is reduced to the path delay fault testing of the blocks that constitute it [12]. In Figure 9, we show how this technique can be applied to n x n MBMs. By adding multiplexers in the original MBM design, we can manipulate the blocks, D_0 and D_1, individually [12].

Due to the fact that both the least (LSP) and the most (MSP) parts of D_1 under the above modifications receive the same test inputs, some of the paths that in [14] are tested in parallel, in this case must be tested explicitly. In the case that each part of D_1 is an ILA with four or more

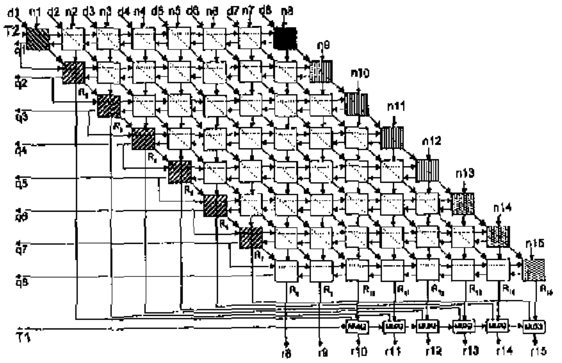


Figure 10. 8 x 8 NRCAD.

cells, the paths that should be tested explicitly are only those including carry propagation from LSP to MSP, while for all the rest parallel robust path delay fault testing can be carried as in [14].

The selection of the SPP-HFRT paths of a basis for the CSM after the above modifications was given in [16].

V. NON-RESTORING CELLULAR ARRAY DIVIDERS

An N x N NRCAD is a combinational circuit with inputs the nominator (n_1, n_2, ... n_{2N-1}) and the denominator (d_1, d_2, ... d_N) and outputs the quotient (q_1, q_2, ... q_N) and the remainder (r_N, r_{N+2}, ... r_{2N-1}). Figure 10 presents the 8x8 NRCAD (ignore the multiplexers at the bottom of Figure10). The dashed lines indicate propagation of the signals to the next cell in either horizontal or diagonal direction. The NRCAD is formed as a two dimensional matrix of identical logic cells. The implementation of each cell of the divider considered in this paper is presented in Figure 11.a and requires 19 gates, without considering the gate with the bold outline. Since the first row cells have their P input driven by logical 1, they can be implemented by 15 gates. Since the leftmost cell of each row, except the last, does not produce an S output it can be implemented by 9 gates as shown in figure 11.b. The upper and leftmost cell can be implemented by 5 gates. Since the rightmost cell of each row, except the first, has connected P and C_i inputs, it can be implemented by 9 gates as shown in Figure 11.c, without considering the gate with the bold outline. The upper and rightmost cell requires only 8 gates. Summarizing the above, we can express the total area of the divider in gates as A_{total} = 19*[(N-1)*(N-2)+1]+15*(N-2)+9*(N-2)+5+9*(N-1)+8.

In the sequel we propose several design modifications for making the NRCAD design easily testable. Excluding the leftmost cells of all rows and the cells of last row, we augment every other cell with an extra AND gate (the AND gate with the bold outline of figures 11.a, 11.c). An extra test input T_0 is used to drive the second input of the added AND gate for all cells. T_0 is only used during testing. During normal circuit operation T_0 is driven to 1. The hardware overhead of the addition of the AND gate in terms of gate equivalents is $(N-1)^2/A_{total}$. The above relation, for N = 8, 16 and 32 leads to a hardware overhead of 4.7, 5.0 and 5.14% respectively. The critical path of the design is from a primary input of the upper and rightmost CAS cell, through the carry chains of each of the N levels of the NRCAD. The chains of two adjacent levels are connected together through the C_{i+1} output of the leftmost cell which is connected to P input of the rightmost cell of next row. The AND gates that we have added do not add any delay on this critical path and all the rest paths that include any sub-path along an added AND gate have smaller propagation delay times than this critical path.For providing observability of the S output of the next to the leftmost cell of each row, excluding the first and last rows, we include N-2 2->1 multiplexers to the NRCAD design. All these multiplexers are controlled by the same test input T_1 and connected as shown in Figure 10. T_1 during normal operation is driven to 1 and the remainder bits are observable at the primary outputs, whereas during testing of specific paths is driven to 0 and the S outputs of the next to the leftmost cells become observable at the remainder outputs of the divider. The output of the multiplexer which drives the r_{i+N} primary output is denoted as O_{i+N}. Since the hardware implementation of the multiplexer requires 4 gate equivalents, the hardware overhead due to multiplexer insertion is: $4*(N-2)/A_{total}$. This relation, for N = 8, 16 and 32 leads to a hardware overhead of 2.3, 1.4 and 0,64 % respectively. Our simulations showed that the insertion of the multiplexers does not cause any delay overhead.

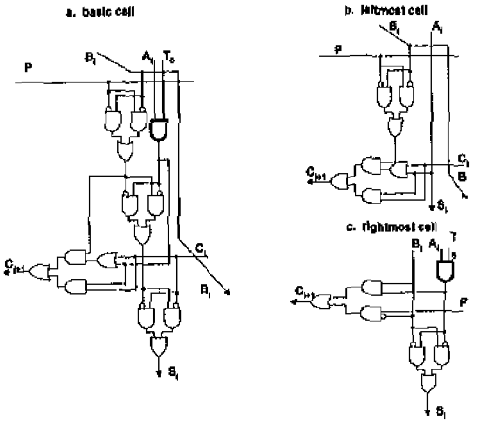


Figure 11. Building blocks of the NRCAD

For providing controllability of the P input for the cells of the first row, we drive all these inputs by a third test input T_2. T_2 is set to 1 for normal circuit operation and is occasionally driven to 0 during testing. The hardware overhead of this change is equal to 4 gates per cell of the first row. Thus the hardware overhead is $(4*N)/A_{total}$ or equivalently 3.0, 1.4 and 0.68 % respectively for N= 8, 16 and 32. This change may increase the critical path of the design by a time equal to the difference of the worst propagation delays between an XOR gate and an inverter, which is overall a negligible delay.

After the above modififctaions the selection of an SPP-HFRT basis can be done as presented in [17].

CONCLUSIONS

The hardware overhead for CSM, MBM and NRCAD of various sizes are presented in Tables I, II and III respectively. We can see that for multipliers and dividers of practical sizes the hardware overhead is very small. The delay overhead of the easily path delay fault testable multipliers is equal to two gate levels, while the delay overhead for the dividers is negligible. From Tables I, II and III we can easily see that the number of paths along which we have to measure the propagation delay is small. However, the number of paths along which the propagation delay must be calculated is prohibitively large. This problem can be overcome by calculating only the delays along the paths included in primitive faults. In this way we derive the maximum speed of the circuit and as we have shown we also ensure that the circuit will function correctly at all lower speeds.

To the best of our knowledge robustly path – delay fault testable or delay verifiable CSM, MBM as well as NRCAD designs have not been reported in the open literature. Therefore the proposed circuits are the only known easily path delay fault testable multiplier and divider circuits.

References

[1] G. L. Smith, "Model for Delay Faults Based upon Paths", Proc. ITC - 85, pp. 342 - 349.

[2] J. D. Lesser and J. J. Shedletsky, "An Experimental Delay Test Generator for LSI Logic", IEEE Trans. on Computers, C-29 (3), March 1980, pp. 235 – 248.

[3] W.-N. Li, S. M. Reddy, and S. K. Sahni. "On path selection on combinational logic circuits", IEEE Trans. on CAD, pp. 56-63, Jan. 1989.

[4] S. Tani, et.al, "Efficient Path Selection for Delay Testing Based on Partial Path Evaluation", Proc. of 16th IEEE VLSI Test Symp., pp. 188 - 193, 1998.

[5] K.-T. Cheng and H.-C. Chen. "Delays testing for non robust untestable circuits", Proc. of Int. Test Conf., pp. 954-961. IEEE, 1993.

[6] W. K. Lam, et. al., "Delay fault coverage and performance tradeoffs", Proc. of the 30th Design Automation Conf. , pp. 446-452. ACM/IEEE, 1993.

[7] M. A. Gharaybeh, M. L. Bushnell, and V. D. Agrawal. "Classification and test generation for path-delay faults using single stuck-fault tests", Proc. of Int. Test Conf., pp. 139-148. IEEE, 1995.

[8] U. Sparmann, et. al.,"Fast identification of robust dependent path delay faults', Proc. of the 32th Design Automation Conf. , pp. 119-125. ACM-IEEE, 1995.

[9] T. W. Williams, et. al., "The interdependence between delay optimization of synthesized networks and testing", Proc. of the 28th DAC, pp. 87-92. ACM/IEEE, 1991.

[10] H. H. Guild, "Some Cellular Logic Arrays for Nonrestoring Binary Division", The Radio and Elec. Engr., vol. 39, June 1970, pp. 345 – 348.

[11] W. Ke and P. R. Menon, "Synthesis of Delay – Verifiable Combinational Circuits", IEEE Trans. on Computers, Feb. 1995, pp. 213-222.

[12] D. Nikolos et al., "Path Delay Fault Testing of ICs with Embedded Intellectual Property Blocks", Proc. of DATE '99, March 1999, pp. 112 – 116.

[13] C. J. Lin and M. Reddy, "On Delay Fault Testing in Logic Circuits", IEEE Trans. on CAD, Sep. 1987, pp. 694-703.

[14] T. Haniotakis, Y. Tsiatouhas and D. Nikolos, "C-Testable One-Dimensional ILAs with Respect to Path Delay Faults : Theory and Applications", Proc. of IEEE Int. Symp. on Defect and Fault Tolerance in VLSI Systems, November 1998, pp. 155 – 163.

[15] Th. Haniotakis, et al., "Easily Testable Carry Save Multipliers with respect to Path Delay Faults", 2nd Electronic Circuits and Systems Conference, September 6-8 1999, Bratislava, Slovakia, pp. 13-16.

[16] E. Kalligeros, et al., "Path Delay Fault Testable Modified Booth Multipliers", Proc. of the XIV Design of Circuits and Integrated Systems Conference (DCIS '99), Nov. 16-19 1999, Palma de Mallorca, Spain, pp. 301 – 306.

[17] G. Sidiropoulos, et al., "Easily Path Delay Fault Testable Non-Restoring Cellular Array Dividers", Proc. of ATS '99, Nov. 16-18 1999, Shanghai, China, pp. 47 – 52.

Table I. CSM Comparisons

CSM size	Hardware Overhead (%)	Number of physical paths	Number of paths to be tested	Reduction %
8×8	17.93	5.825×10^{8}	1032	99.9998
16×16	8.64	3.189×10^{17}	4392	≈100
32×32	4.24	6.245×10^{34}	18408	≈100
64×64	2.10	2.142×10^{69}	75624	≈100

Table II. MBM Comparisons

CSM size	Hardware Overhead (%)	Number of physical paths	Number of paths to be tested	Reduction %
16×16	12.4	$1.5*10^{9}$	7222	≈100
32×32	6.5	$2.4*10^{15}$	18654	≈100
64×64	3.3	$2.3*10^{27}$	65326	≈100

Table III. NRCAD Comparisons

CSM size	Hardware Overhead (%)	Number of physical paths	Number of paths to be tested	Reduction %
8 x 8	10	$3,7 \text{ x } 10^{21}$	735	≈100
16×16	7.8	$5,3 \text{ x } 10^{80}$	3231	≈100
32×32	6.5	$6,4 \text{ x } 10^{324}$	13599	≈100