On Testing Wave Pipelined Circuits

Jui-Ching Shyur, Hung-Pin Chen, Tai-Ming Parng
Department of Electrical Engineering, National Taiwan University
Taipei, Taiwan, 10764 R. O. C.

Abstract--We present currently the first approach to test wave pipelined circuits. Wave faults are identified and well modeled. The fault coverage is obtained by statistics and fault probabilities. We also present the robust wave test, and give a TG algorithm, by testing wave pipelined and ISCAS combinational circuits, capable of reaching on average 85.8% fault coverage.

1. INTRODUCTION

Wave pipelining is a technique to boost the operating speed of combination circuits. By using internal capacitance as data storage and balancing delays on physical paths, three to four times of speed-up is observed in [1], and an order of speed-up is even reported. Recent researches related to this technique are generally classified in two categories: the synthesis ones [1,2,3], and the analysis ones [4,5,6,7,8,9,10,11]. The synthesis of wave pipelined circuits aims at achieving *maximal pipelining*. There are methods proposed to help automate the insertion of delay elements [1,3], and exhibition of the design processes. Among these works, [7] also discusses the area-time trade-offs that can be made for the design of wave pipelined circuits, which provides an economical point of view. The analysis of wave pipelined circuits concerns with the *optimum clocking schemes* for circuits with level-sensitive latches. In [11], the analysis further extends to circuits with multiple stages of pipelining and feedback loops.

While these papers have provided ways toward the design and analysis of wave pipelined circuits, they rarely pay attentions to the testing issues of wave pipelined circuits, or only provide simulations results. The difficulties lie on the operating mode and the complexity of testing all possible faults in the circuits. For the first difficulty, one *sees* several *waves* traversing the combinational circuit simultaneously during its normal operation. Without proper modeling, traditional delay-fault testing is not able to detect faults of successive transitions in the circuits. In addition, the high speed of wave pipelining may require the MCM technique [12] to facilitate at-speed testing. As for the second one, the major type of faults in the wave pipelined circuits, in addition to the stuck-at and delay faults, is that two successive waves come too close to stabilize the signal on a node. Since waves may come through all paths, such a fault (hereafter as the *wave faults*) involves more than one path for an activation. It is thus important to determine which paths are relevant to the fault and should be elected for testing. Such difficulties make testing wave pipelined circuits a complex problem, and require the following issues be properly handled: 1) The identification and well modeling of faults in wave pipelined circuits. 2) Provision of reliable fault coverage calculations that reflect real situations in live circuits. 3) Robust test whose correctness is not invalidated by hazards, and 4) TG algorithms electing paths for efficient test.

Traditional testing for the timing of combinational circuits uses the delay fault models [13,14,15,16,17,18]. Two fault models have been proposed: The gate delay fault model [19,20] assumes lumped delay at a faulty gate, while the path delay fault model [21,22] aims at paths which have larger propagation delays than the clock periods. However, to effectively test wave pipelined circuits, a hybrid model is used. In this paper, we concentrate on the faults caused by successive waves along the path, named as the *wave faults*. In brief, a wave fault is caused on a node if of which a former signal transition (the first wave) is invalidated by an immediate following signal transition (the second wave) in the opposite direction. This fault results in a spike, or even no transition at all, instead of a stable signal level between transitions. Since there are two transitions under consideration for this fault, at least two paths from primary input lines (PI's) to the node are involved, with each corresponding to a transition. The activated fault is then propagated toward the primary output lines (PO's) through a propagation path. From the primary output lines we can observe the waveform and determine whether the fault exists. Note that even both the transitions on a node are valid and there is no wave fault, the path delay on the propagation path may still be too long or too short and fails to satisfy the timing requirements on the PO's. The resulting errors may be set-up or hold-time errors, which we classify as the *wave delay faults*. From these observations, we propose a wave pipelining test with the following features:

1. *Statistical modeling for wave faults.* The probability that faults will occur differs from gates to gates in a circuit. It is thus desired that faults more likely to happen be tested first.
2. *Reliable fault coverage.* Instead of using the ratio of testable gates over total gates as the fault coverage, the fault probabilities take parts as weights during calculation, resulting in a more reliable fault coverage.
3. *Robust test.* Under the assumption of single wave fault, we are able to maintain the correctness of tests without the affectation of hazards. The robust test criterion also provides the `minimal' conditions for propagation of off-path sensitizing input transitions.
4. *Heuristic TG algorithm.* Proposed algorithm is a direct search process that uses an objective to guide forward and backward propagation to go through the paths most likely to activate the wave faults and wave delay faults.

31^{ST} ACM/IEEE Design Automation Conference ®

For the rest of this paper, the basic wave pipelining concepts are presented in Section 2. Test for wave pipelined circuits is proposed in Section 3. In this section, we will present the wave fault model, statistical fault coverage, the robust test for wave faults, and the proposed TG algorithm. In Section 4, some preliminary experimenting results are presented. Finally, we make concluding remarks in Section 5.

2. WAVEPIPELINING MODEL

Information about the timing models for wave pipelining can be found in [1,3,5,7]. In this section, we review some important properties in the fault aspect.

A typical wave pipelined circuit, when operating at a certain instant has several concurrent waves propagating along the combinational circuit, as shown in Fig.1. Each wave is formed by the transitions caused from the simultaneous stimulus on the output leads of the input latches. It propagates toward the POs along the gates in it, and may cause some of them to change their output states. Before the wave can reach the output latches, another stimulus may be activated by the control signals of the input latches. The second wave is then generated and immediately follows the former wave. After a wave has completely reached the POs, the control signals of the output latches will help store the output states of the PO's in the latches. In general, a single phase system capable of holding *n* waves will require *n* clock periods for a wave to propagate from the input latches to the output latches. For a multiphase system, the propagation time required will be added with the phase difference between the control signals of the input and output latches. To make sure that the wave pipelined circuit will function correctly, any wave shall not surpass its former wave; otherwise, a fault will occur.

Since in this paper, we aim at the testing issues instead of design or synthesis issues, a brief summary of the timing models is given as follows.

1. The gate delay model describes a gate *g* by four delays to quantify the bounds on its rise and fall delays:

$$Tg_{\min}^{fall},\ Tg_{\max}^{fall},\ Tg_{\min}^{rise},\ Tg_{\max}^{rise} \quad (1)$$

2. On each input lead (*i*) of a gate, the data arrival time, or input assertion time (*IA*), describes the timing on the input part of the gate:

$$IAgi_{\min}^{fall},\ IAgi_{\max}^{fall},\ IAgi_{\min}^{rise},\ IAgi_{\max}^{rise} \quad (2)$$

3. While on the output lead of a gate, we have the output assertion times (*OA*) to describe the timing on the output part of the gate:

$$OAg_{\min}^{fall},\ OAg_{\max}^{fall},\ OAg_{\min}^{rise},\ OAg_{\max}^{rise} \quad (3)$$

4. The path delay model is calculated based on the relation of input/output assertion times of gates. Taking the NAND gate for example, the input/output timing relation can be described as follows:

$$OAg_{\min}^{fall}=\min(IAgi_{\min}^{rise})+Tg_{\min}^{fall} \quad (4)$$
$$OAg_{\max}^{fall}=\max(IAgi_{\max}^{rise})+Tg_{\max}^{fall} \quad (5)$$
$$OAg_{\min}^{rise}=\min(IAgi_{\min}^{fall})+Tg_{\min}^{rise} \quad (6)$$
$$OAg_{\max}^{rise}=\max(IAgi_{\max}^{fall})+Tg_{\max}^{rise} \quad (7)$$

From the relations, the short/long path delays can be obtained by using the ASAP/ALAP algorithms.

5. From the input/output assertion time relationship, the *minimum wave length* $WL_{\min}$ is also defined on each node including PIs, POs, and output leads of gates. The minimum wave length defines the separation required to keep successive waves on a node from potential overlapping. The minimum wave length on the output lead of a gate, for example, is

$$WLg_{\min}=\max(OAg_{\max}^{rise}-OAg_{\min}^{rise},\ OAg_{\max}^{fall}-OAg_{\min}^{fall}) \quad (8)$$

To ensure all nodes from wave overlapping, the clock period *cycle_time* is bounded by all minimum wave lengths of the primary output terminals:

$$cycle\_time \geq \max(WL_{\min}) \quad (9)$$

The timing errors in a wave pipelined circuit are in fact the delay faults. The hold time errors are caused by short path delay faults, while the set-up time errors are by long path delay faults. Refer to Fig.2, we formulate the set-up time (*U*) and hold time requirements on the input leads (*I*) of the output latches respectively as follows:

$$U \leq fall-\Delta f-IAI_{\max} \quad (10)$$
$$fall+\Delta f \leq IAI_{\min}+cycle\_time \quad (11)$$

where *fall* and Δf are the fall time and its deviation of the control signal (phase) of the output latches.

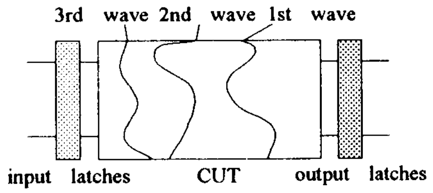

Fig.1 Concurrent waves propagating on a wavepipelining circuit.

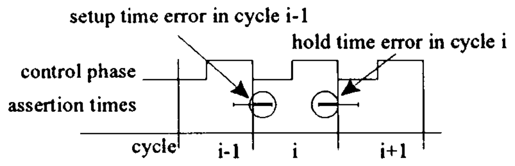

Fig.2 Set-up and hold time errors in a wavepipelining circuit.

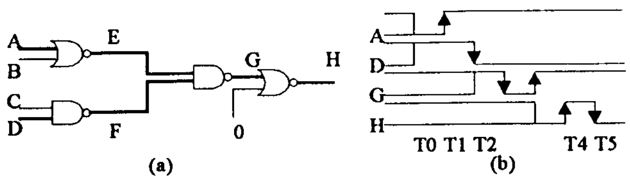

Fig.3 Wave faults (a)example circuit and emphasized paths, (b)timing diagrams on focused nodes.

In this formulation, it assumes but is not limited to level-sensitive latches, which are on when the control signals go high, and closes at the falling edges of the control signals.

3 WAVE FAULT TESTING

3.1 Wave Fault Models

Focusing on an internal node of a circuit, a wave fault can be viewed as the result of *internal* delay faults. Consider the circuit in Fig.3(a), where we have two path A-E-G-H (as L_{x1}) and D-F-G-H (as L_{x2}). Let (A,B,C,D) at T0, T1, T2 be (0,0,1,1), (1,0,1,1), (1,0,1,0). The internal node G is expected to have a rise transition after a fall one, and accordingly 0-1-0 transitions at H, as shown in Fig.3(b). To correctly latch the signal of H at T4 and T5, both transitions at H must satisfy the setup and hold time requirements.

To classify the wave faults, we define the 0-1-0 and 1-0-1 wave faults as follows.

Definition 1: A gate has a *0-1-0 fault* if the 0-1 (rise) transition and the following 1-0 (fall) transition fail to happen. Likewise, A gate has a *1-0-1 fault* if the 1-0 (fall) transition and the following 0-1 (rise) transition fail to happen.

Since wave pipelining suggests the use of simple gates, we may enumerate the fault activation for two-input NAND and NOR gates. However, we note that the activation can also be applied to complex gates.

3.2 *Statistical Model and Fault Coverage*

Since device parameter variations are often caused by random fluctuations during fabrication, the gate delays can be described as distributions instead of exact values. By the Central Limit Theorem, each min/max rise/fall delay of gates of the same type can be viewed as a normal distribution with a mean and a standard deviation. Using the path delay model of the wave pipelined circuits (described in Section 2), the assertion times at a node are also normal distributions. For example,

$$M = MEAN(OAg_{\min}^{fall}) = m_1 + \cdots + m_2 \tag{12}$$

$$Z^2 = z_1^2 + \cdots + z_n^2 \tag{13}$$

where Z is the standard deviation, $m_1, \ldots, m_n$ the minimum rise or fall delays of gates on the path, and $z_1, \ldots, z_n$ their corresponding standard deviations.

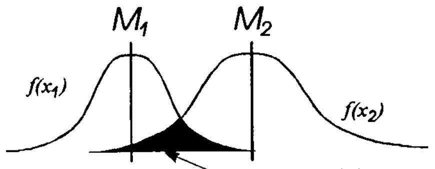

Fig. 4 Wave fault probability.

With these assertion time distributions, we can further define the *wave fault probability* for a node, and based on the fault probability, the *wave fault coverage* of a test.

Definition 2: Let $f(x_1)$ be the latest assertion time distribution (p.d.f.) of a certain transition direction at some node, and $f(x_2)$ be the earliest assertion time distribution (p.d.f.) of the opposite direction in the next cycle, the fault probability is defined as the probability that the transition in the former cycle is invalidated by that of the later cycle. As suggested in Fig.4, the probability is an integration of these p.d.f.'s:

$$p = \int_{-\infty}^{\infty} f(x_1) \cdot f(x_2) dt \tag{14}$$

To derive the fault probability, let M_1, M_2 and Z_1, Z_2 be the means and standard deviations of p.d.f. $f(x_1)$ and $f(x_2)$, we have:

$$\int f(x_1) \cdot f(x_2) dt = \int \frac{1}{2\pi Z_1 Z_2} e^{-(t-M_1)/2Z_1^2-(t-M_2)/2Z_2^2}$$
$$= \frac{1}{\sqrt{2\pi(Z_1^2+Z_2^2)}} e^{-\frac{(M_1-M_2)^2}{2(Z_1^2+Z_2^2)}} \tag{15}$$

Definition 3: We define the wave fault coverage as the ratio of the sum of wave fault probabilities of the nodes that can be tested, over the sum of wave fault probabilities of all nodes in the circuit under test. That is,

$$\textit{fault coverage} = \frac{\sum p_j | j: \textit{testable}}{\sum p_i | \forall i} \tag{16}$$

As the formula shows, the wave fault coverage behaves like *weighted* SSF coverage, where the fault probabilities are the weights. Note that the faults encompass the 0-1-0 and 1-0-1 wave faults, and the faults must be propagated to the PO's for observation, the delay faults on the propagation paths may mask the wave faults. However, the test is still valid since a fault has been observed.

3.3 *Robust Test and Test Generation*

As explained in Section 3.1, we apply three vectors V_0, V_1, V_2 at time T_0, T_1, T_2 to the input of CUT to activate the focused wave fault, and sample the output at T_4, T_5 for signal transitions. It is assumed that at time T_1 and T_2, there are no transients or hazards on primary input lines. Furthermore, it is noted that in wave pipelining, we use only one or two-input gates. For each wave fault, there are two disjoint physical paths, denoted by L_1 and L_2, separately connects a primary input to the faulty site, and another physical path, denoted by L_3, that connects the site to a primary output line. For simplicity, we denote these paths as a whole by the *wave-path L*. These paths are said to have faults if on the primary output line of wave-path L, the corresponding transitions are not observed at time T_4 and T_5. By this, we mean both the wave faults and wave delay faults.

To help afford hazard free test and to provide the minimal signal conditions to propagate required transitions, we define the *robust wave fault test* as follows:

Definition 4: A three-pattern $<V_0, V_1, V_2>$ is said to be a *robust wave fault test* for a wave-path L if and only, when L is faulty and test $<V_0, V_1, V_2>$ is applied, the circuit output is different from the expected state at sampling time T_4, T_5, independent of the input assertion times of the off-path sensitizing signals.

Without detailed proof here, we can determine the signal values of the off-path gate input leads to achieve robust wave fault test by the following theorem.

Theorem 1: A three-pattern $<V_0, V_1, V_2>$ is a robust wave fault test if and only if 1) $<V_0, V_1, V_2>$ will launch the *desired transitions* at the input leads of the wave-path L at time T_1, T_2, and 2) at each gate on path L, the off-path sensitizing inputs have logic values that are *covered* by the values indicated in Tab.1.

In this table, we are free to assume that A and C have the control of the first transitions at T_1, and that B and D have the control at T_2. The symbol $\uparrow$ and $\downarrow$ stand for the rise and fall transition, while S_0 and S_1 for stable at 0 and 1 logic level. In some papers, '$\downarrow$ or S_0' is refer to as U_0, and '$\uparrow$ or S_1' as U_1. The symbol 'x' means 'don't care' condition for a signal subject to its previous condition. For example, a signal of logic 1 at T_1 may not rise at T_2 assuming there is no transients or hazards on primary input lines. Also note that we do not consider the cases where only one gate input signal takes both transitions to activate the wave fault. Since in such undesirable cases, detected fault is in fact a stuck-at fault and not a wave fault.

3.4 *Wave Fault TG Algorithm*

To facilitate the TG process, a preparation task is necessary to calculate the assertion times and the fault probability according to Equ.4-6, 11-14. All faults, together with their transitions to justify the activation are then lined up in a

Table 1: Sensitizing input transitions.

Transition	0-1-0 wave fault				1-0-1 wave fault			
	NAND		NOR		NAND		NOR	
time step	A	B	C	D	A	B	C	D
T_1	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$ or $S0$	$\uparrow$	$\uparrow$ or $S1$	$\uparrow$	$\uparrow$
T_2	$\uparrow$	$\uparrow$	x	$\uparrow$	x	$\downarrow$	$\downarrow$	$\downarrow$

priority queue Q with the fault most likely to happen in the front. The TG algorithm then each time gets an item from the fault priority queue for test, and accumulates the fault coverage according to Equ.16. Proposed TG algorithm is as follows; note that *Imply* will perform the maximum implication and maintain a *W-frontier*. Similar to the *D-frontier*, the *W-frontier* contains gates ready for forward propagation.

```
TG()
begin
if ( error at PO ) then return SUCCESS
if ( test not possible )
   then return FAILURE
( $k$, $t_{k1}$, $t_{k2}$) = Objective()
Backtrace($k$, $t_{k1}$, $t_{k2}$)
Imply()
returnTG()
end
```

The procedure *Backtrace* is responsible for propagating sensitizing transitions toward PI according to Tab.1. Note that *Backtrace* will assign sensitizing transitions for gate output leads, if there is a conflict of transition assignments (e.g., both rise and fall on the same line at the same time,) the Backtrace fails.

```
Backtrace($k$, $t_{k1}$, $t_{k2}$)
/* map objective into PI transition assignments */
begin
    if ( ConflictCheck(k) ) = FAILURE then
        return FAILURE
    if $k$ is a PI then
        perform PI transition assignment
        return SUCCESS
    for every input ($j$) of $k$
        obtain sensitizing transitions $t_{j1}$, $t_{j2}$ by
Tab.1
        if Backtrace($j$, $t_{j1}$, $t_{j2}$) = FAILURE then
            return FAILURE
        return SUCCESS
    end
```

The *Objective* procedure in *TG* algorithm handles the forward propagation tasks. It chooses the shortest paths to PO's for short path delay faults (hold time errors,) and the longest paths to PO's for long path delay faults (set-up time errors.)

```
Objective()
begin
    if test for short path delay faults then
        select from W-frontier a gate $g$ with smallest
        $OAg_{min}$.
    if test for long path delay faults then
        select from W-frontier a gate $g$ with largest
        $OAg_{max}$
    obtain sensitizing transitions $t_{g1}$, $t_{g2}$ for $g$
    return ($g$, $t_{g1}$, $t_{g2}$)
end
```

4. RESULTS

We tested three wave pipelined adders and three wave pipelined Max-Min selectors, together with ISCAS combinational benchmarks c432, c499, and c1355, as listed in Tab.2. In this table, we note that the fault coverage is obtained by Equ.16, and not the ratio between testable and total gates. The tested faults include both 0-1-0 and 1-0-1 wave faults, and each fault includes the short and long path delay faults. Circuit A162 is a 16 bit binary lookahead adder, A323 is a 32 bit ternary lookahead adder, and A324 is a 32 bit quadruple lookahead adder. Circuit M162, M323, and M324 are the Max-Min selectors of 16 bit binary, 32 bit ternary, and 32 bit quadruple lookahead, respectively. From the results, we have an average of 85.8\% fault coverage for circuits having less than 1000 gates. We note that since the adder and Max-Min selectors are designed for wave pipelining, there are less reconvergent fanouts than that of the ISCAS benchmarks, resulting in higher fault coverage. Compared with the adders, the Max-Min selectors have more chances that reconvergent fanouts cause transition assignment conflicts, their fault coverage is in general less than that of the adders.

5. CONCLUSION

Wave pipelined circuits has been difficult to test since there are not yet a fault model, neither a reliable fault coverage calculation for their tests. In this paper, we have proposed a solution toward testing wave pipelined circuits. First we give a formal definition, namely the wave fault, to model potential faults in wave pipelined circuits. Second, we define the probability of fault to happen using statistical models; and based on the probability, the fault coverage. We also present the robust wave test, and our test generation algorithm. From inspecting the preliminary results of testing wave pipelined circuits, an average of 89.3% fault coverage has been achieved. Since testing wave pipelined circuits is a very complex problem, such coverage is believed to be acceptable.

Table 2: Wave fault testing results

circuit name	number of gates	number of testable faults	fault coverage
A162	192	351	92.4%
A323	538	903	93.3%
A324	680	1072	90.1%
M162	320	577	88.7%
M323	949	1547	82.1%
M324	1176	-	-
c432	160	311	74.7%
c499	202	395	81.2%
c1355	546	844	83.9%

6. REFERENCE

[1] T. Gray, T. Hughes, S. Arora, W. Liu, R. Cavin, "Theoretical and Practical Issues in CMOS Wave Pipelining,"VLSI Design '91, pp. 9.2.1-9.2.5.

[2] S. Anderson, J. R. Goldschmidt, D. Powers, "The IBM System/360 Model 91 Floating Point Execution Unit," Jan. 1967,IBM Journal of Research and Development, pp. 34-53.

[3] D. Wong, G. De Micheli, M. Flynn, "Inserting Active Delay Elements to Achieve Wave Pipelining," *ICCAD-89*, ACM/IEEE, pp. 270-273, 1989.

[4] K. A. Sakallah, T. N. Mudge, O. A. Olukotum, "Analysis and design of latch-controlled synchronous digital circuits," *Proc. of 27th DAC*, ACM/IEEE, pp. 111-117, 1990.

[5] K. A. Sakallah, T. N. Mudge, O. A. Olukotum, "Analysis and Design of Latch-Controlled Synchronous Digital Circuits," *IEEE tran. on CAD*, VOL. 11, NO. 3, pp.322-333, 1992.

[6] K. A. Sakallah, T. N. Mudge, O. A. Olukotum, "*check* T_c and *min* T_c: Timing Verification and Optimal Clocking of Synchronous Digital Circuits," *ICCAD-90*, ACM/IEEE, pp. 552-555, 1990.

[7] J.C. Shyur, H.P. Chen, T.M. Parng,, "Achieve Timing Error-Free Wave Pipelining for Latch Based CMOS Synchronous Systems," *Tau'93*, ACM, Sep.14-16, 1993, Germany.

[8] S. H. Unger, C. J. Tan, "Clocking schemes for high-speed digital systems," *IEEE tran. on computer*, VOL. C-35, NO. 10, pp.880-895, 1986.

[9] N. Weiner, A. Sangiovanni-Vincentelli, "Timing analysis in a logic synthesis environment," *Proc. of 26th DAC*, ACM/IEEE, pp.655-661, 1989.

[10] M. R. Dagenais, N. C. Rumin, "On the calculation of optimal clocking parameters in synchronous circuits with level-sensitive latches," *IEEE tran. on CAD*, VOL. 8, NO. 3, pp.268-278, 1989.

[11] C. Th. Gray, W. Liu, R. K. Cavin III, "Optimum Clocking for Wave Pipelined Operation in Multiple Stage Systems with Feedback," *Tau'93*, ACM, Sep.14-16, 1993, Germany.

[12] T. Hughes, T. Gray, W. Farlow, W. Liu, R. Calvin, "Advantages of MCM Technology for CMOS Wavepipelined Systems," VLSI Design '92, pp. 33-39.

[13] C. J. Lin, S. M. Reddy, "On Delay Fauly Testing in Logic Circuits," *IEEE tran. on CAD*, VOL. 6, NO. 5, pp.694-703, 1987.

[14] P. Franco, E. J. McCluskey, "Delay Testing of Digital Circuits by Output Waveform Analysis," *Proc. 1991 Int. Test Conf.*, pp.798-807, 1991.

[15] E. S. Park, M. R. Mercer, T. W. Williams, "A Statical Model for Delay-Fault Testing," *Design and Test*, IEEE, pp.45-55, 1989.

[16] E. S. Park, T. W. Williams, M. R. Mercer, "Delay Testing Quality in Timing-Optimized Designs," Proc. 1991 Int. Test Conf.,} pp.897-905, 1991.

[17] J. Savir, W. H. McAnney, "Random Pattern Testibility of Delay Test," *IEEE tran. on computer*, VOL. 37, NO. 3, pp.291-298, 1988.

[18] A. K. Pramanick, S. M. Reddy, "On Multiple Path Propagating Tests for Path Delay Faults," *Proc. 1991 Int. Test Conf.*, pp.393-402, 1991.

[19] A. K. Pramanick, S. M. Reddy, "On the Fault Coverage of Delay Fault Detecting Tests," *Proc. 1990 EDAC*, pp.334-338, 1990.

[20] V. S. Iyengar et al., "Delay Test Generation 1 -- Concepts and Coverage Metrics," *Proc. 1991 Int. Test Conf.*, pp.857-866, 1988.

[21] G. L. Smith, "Model for Delay Faults Based Upon Paths," *Proc. 1985 Int. Test Conf.*, pp.342-349, 1985.

[22] M. H. Schulz et al., "Advanced Automatic Test Pattern Generation Techniques for Path Delay Faults," *Proc. 19th. IEEEE Int'L Fault Tolerant Computing Symp.*, pp.44-51, 1989.