01 Jan 1993

Faulty Behavior of Storage Elements and Its Effects on Sequential Circuits

Waleed K. Al-Assadi
Missouri University of Science and Technology, waleed@mst.edu

Y. K. Malaiya

A. P. Jayasumana

Recommended Citation

W. K. Al-Assadi et al., "Faulty Behavior of Storage Elements and Its Effects on Sequential Circuits," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Institute of Electrical and Electronics Engineers (IEEE), Jan 1993.
The definitive version is available at https://doi.org/10.1109/92.250192

Faulty Behavior of Storage Elements and Its Effects on Sequential Circuits

Waleed K. Al-Assadi, *Student Member, IEEE,* Yashwant K. Malaiya, *Senior Member, IEEE,*
and Anura P. Jayasumana, *Senior Member, IEEE*

***Abstract*—It is often assumed that the faults in storage elements (SE's) can be modeled as output/input stuck-at-faults of the element. They are implicitly considered equivalent to the stuck-at faults in the combinational logic surrounding the SE cells. Transistor-level faults in common SE's are examined here. A more accurate higher level fault model for elementary SE's is presented that better represents the physical failures. It is shown that a minimal (stuck-at) model may be adequate if only modest fault coverage is desired. The *enhanced* model includes some common fault behaviors of SE's that are not covered by the minimal fault model. These include *data-feedthrough* and *clock-feedthrough* behaviors, as well as problems with logic level retention. Fault models for complex SE cells can be obtained without a significant loss of information about the structure of the circuit. The detectability of *feedthrough* faults is considered.**

I. Introduction

FUNCTIONAL fault modeling is an effective approach to handling the complexities of large digital circuits. A functional fault model hides the complex fault behavior and presents a way of considerably simplifying test generation [1], [2]. Higher level fault models are easier to use because they represent the fault behavior independent of detailed lower level description. It has been shown in some situations, however, that a simple functional model may not adequately represent a significant fraction of failures. When this is the case, tests based on such a model may not be significantly better than random testing. If the fault model is *adequate*, a functional test set will test for most faults, while at the same time considerably reducing the test-generation effort. A fault model can be termed adequate if it explicitly covers (i.e., coverage is guaranteed for) a major fraction, say $x\%$, of all likely faults [3]. The number x cannot be obtained by using any fundamental considerations but would be based on a reasonable convention. The faults not explicitly covered may or may not be tested if the test vectors are obtained using a fault model. Thus, a fault model with low *explicit coverage* is likely to be inadequate.

A good strategy is to obtain a functional fault model for logic blocks derived from the physical structure of the circuit. This requires that accurate fault models for primitive blocks, such as elementary storage elements (SE's), be considered. Although test considerations at the low level can be computationally complex, an accurate fault model for complex logic blocks inferred from the physical structure of the circuit can reduce the test generation and fault simulation efforts significantly.

The elementary SE's are the basic primitives in complex logic blocks like registers, finite-state machines, and static memory blocks. This paper examines the major transistor-level faults for four elementary SE's. The behavior of each cell under the above faults is analyzed to evaluate possible functional fault models. Results for elementary SE cells are extended to characterize complex SE cells. In Section II, the minimal stuck-at model and the proposed enhanced model are described. SE cells are examined in Section III for all possible transistor-level faults to seek a fault model with high fault coverage. Section IV describes the fault modeling of complex SE cells based on the fault models of the constituent elementary SE cells. Section V considers the detectability of feedthrough faults.

II. Fault Modeling of Elementary SE Cells

The minimal (stuck-at) fault model assumes that internal faults in the SE's can be modeled as stuck-at-0/1 at the inputs or the outputs of the SE's. We examine below the effectiveness of the minimal fault model in representing physical failures. The results reveal the need for a more accurate fault model to better represent the physical failures at the transistor level of an elementary SE.

To examine an SE cell, in general, an input sequence is required rather than a single input vector. Let $T = \{t_1, \cdots, t_n\}$ be the set of all possible input combinations and $R(s, t_i)$ the response of the cell to the input vector t_i applied to the cell when the cell is at state s. The behavior of each cell under all possible transistor faults is examined for all input combinations and previous states. A multivalued logic representation is used to better represent voltage levels that are not exactly logic 1 (hard 1) or logic 0 (hard 0). Here, *high* level (H) corresponds to both "hard 1" and "soft 1," and *low* level (L) corresponds to both "hard 0" and "soft 0" [4]. A fault that causes the SE

Manuscript received March 25, 1993; revised July 23, 1993. This work was supported by an SDIO/IST-funded project monitored by ONR. A preliminary version of this paper was presented at the 6th International Conference on VLSI Design, India, January 1933.

W. K. Al-Assadi and A. P. Jayasumana are with the Department of Electrical Engineering, Colorado State University, Fort Collins, CO 80523.

Y. K. Malaiya is with the Department of Computer Science, Colorado State University, Fort Collins, CO 80523.

IEEE Log Number 9213089.

output to be $L(H)$ for all $t_i \in T$, regardless of the state of the SE, can be modeled as stuck-at-0/(1). Under some faults, the output of the faulty cell cannot make a high to low (low to high) transition, and the corresponding behavior is represented by $H \not\to L(L \not\to H)$. Such faults generally appear as stuck-at-1 (stuck-at-0).

However, some faulty behaviors of the SE cell do not manifest as stuck-at-0/1. Such faults cause the SE cell to become *feedthrough*, either *data-feedthrough* or *clock-feedthrough*, as defined below.

Definition 1: A faulty SE cell is said to have a *feedthrough* fault if it becomes either *data-feedthrough* or *clock-feedthrough*.

1) A faulty SE cell is said to be *data-feedthrough* when its behavior becomes combinational such as $R(s, t_i) = f(y)$ for each $t_i \in T$, where y is the data part of t_i. (For example, for a D-latch, if D is the input data and Q is the output node, then y is a single element vector corresponding to D. The cell becomes transparent such that the output $Q = \Delta(D)$ or $Q = \Delta(\overline{D})$. $\Delta(D)$ corresponds to signal D delayed by Δ time units, where Δ is the propagation delay of the cell.)

2) A faulty synchronous SE cell is said to be *clock-feedthrough* if $R(s, t_i) = \text{CLK}$ or $R(s, t_i) = \overline{CLK}$ where CLK is the control signal.

The *feedthrough* faults can lead to timing problems or coupling between combinational blocks which are normally separated by the SE's.

Other faults may cause the cell to exhibit a behavior change in the latch phase, while still functioning properly in the transparent phase. *Nonretention* of logic 1 ($NR - 1$), *nonretention* of logic 0 ($NR - 0$), *conditional nonretention* of logic 1 ($CNR - 1$), and *conditional nonretention* of logic 0 ($CNR - 0$) behaviors which are defined below correspond to such faults.

Definition 2: An elementary SE cell exhibits a *nonretention* of logic 1 ($NR - 1$) behavior if the data input of the cell at the sampling edge of the clock is 1 but the state of the cell becomes 0 in the latch phase. ($NR - 0$) is defined in a similar manner.

Definition 3: Consider an SE cell in the state $Q = 1$, with the data input also at logic 1 at the sampling edge of the clock. The SE cell exhibits *conditional nonretention* ($CNR - 1$) behavior if a change in data input from 1 to 0 during the latch phase causes the output Q to change to 0. ($CNR - 0$) is defined similarly.

Some recent papers address the detection of several physical failures in CMOS latch cells. It has been found [5] that several stuck-opens in CMOS latches are not detectable, although they may degrade timing performance. A testable implementation for the transmission-gate latch cell in which any stuck-open fault can be detected is proposed in [6]. A symmetrical D-latch has been examined for stuck-open faults [7]. Bridging faults in CMOS scan registers implemented with a CMOS transmission-gate latch are investigated in [8]. Here a more complete set of possible faults that may alter the functional behavior of the cell is considered. An *enhanced fault model* is proposed, one which provides a higer explicit fault coverage for the SE's under consideration. The enhanced model includes faults that cause *feedthrough*, *nonretention*, and *conditional nonretention* behaviors as well as the stuck-at faults. Such faults can be detected by monitoring logical levels. Hence, they are termed logically testable.

III. Detailed Examinations of the Elementary SE Cells

In this section, a detailed examination of four different elementary SE's is presented. Each cell is examined for all possible transistor faults. Results obtained analytically based on a multivalued algebra have been verified by SPICE. A good functional fault model is sought such that the functional behavior of faulty SE cells can be adequately described. Both the *minimal* and the *enhanced* fault models are examined for effectiveness in representing functional faults. Because of the transistor sizing and technology used, "0" dominates if two nodes are bridged. All possible bridging faults between nodes in the same well are considered. We use $f(x, y)$ to indicate a bridging fault between nodes x and y. Bridging faults between internal nodes of different wells are not included because the probability of having such faults is very small [9]. Analysis assumes that a bridging fault corresponds to a *hard short*.

The clock is applied such that the cell is devoid of clock turn-on and turn-off hazards and glitches, i.e., the data are stable at the sampling edge of the clock [6]. The analysis shows that many stuck-on and bridging faults change the conductance path between V_{dd} and V_{ss} nodes. This suggests that monitoring the supply current (I_{DDQ}), which can be many orders of magnitude higher in the presence of such faults, can be used for testing such faults. In the presence of stuck-open faults, an SE cell could turn from static to dynamic under some input vectors. This means that the logic value of the output of the cell is maintained due to the charge stored in the capacitance associated with the output node. This state may last only for a short time due to the leakage of the charge. However, at normal clock rates such faults can be detected only if they manifest as delay faults. Faulty behavior of several SE cells are summarized next.

A. The Clocked D-Latch

The clocked D-latch cell is shown in Fig. 1. This cell is commonly used in the implementation of scan-path registers such as LSSD latches and scan-path flip-flops in the TITUS system [10]. The naming convention used for bridging faults is shown in Fig. 2. The behavior of the cell under all possible faults is summarized in Table I. While a large fraction of faults is covered by the enhanced fault model described above, certain faults manifest as delay faults or parametric faults. Bridging faults between internal nodes in the same well are also included. They are $f_1 = (a, b), f_2 = (b, c), f_3 = (c, d), f_4 = (a, c), f_5 = (a, d)$, and $f_6 = (b, d)$, where the nodes involved are as

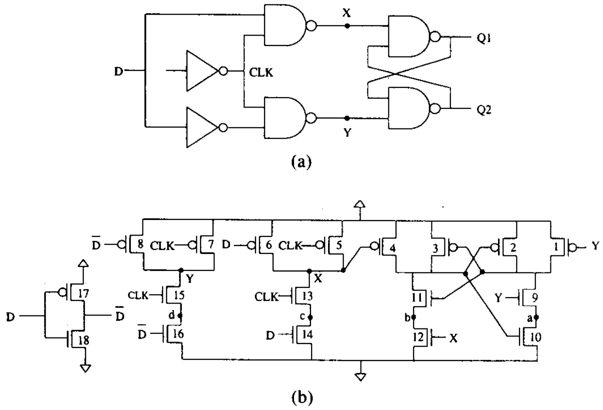


Fig. 1. The clocked D-latch. (a) The gate level and (b) the transistor level.

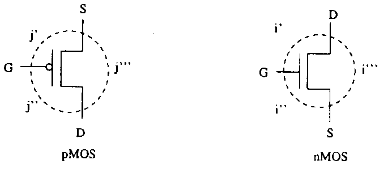


Fig. 2. Naming convention for bridging faults.

TABLE I
BEHAVIOR OF THE CLOCKED D-LATCH UNDER ALL POSSIBLE FAULTS

Output (Q1)	Faults	Model	% of testable faults
H	S-on: 3, 4 Bridging: $2'$, $3'''$, $4'''$, $12''$	minimal stuck-at	56%
H$\not\to$L	S-open: 1, 11, 12, 15, 16, 17 S-on: 7, 8, 18 Bridging: $1'$, $4''$, $6'$, $6''$, $7'''$, $8'''$, $9'''$, $11''$, $12'$, $15''$, $16'$, $16''$, $17'$, $18'''$, f_7, f_{10}		
L	Bridging: $10''$		
L$\not\to$H	S-open: 4, 13, 14 S-on: 5, 6 Bridging: $4'$, $5'''$, $6'''$, $12'''$, $13''$, $14''$, $18''$, f_8, f_9		
D	Bridging: $5'$, $5''$, $7'$, $7''$, $13'$, $15'$	feed-through	7%
NR-1	S-open: 9, 10, 18 S-on: 1, 2 Bridging: $1''$, $1'''$, $2'''$, $3'$, $8'$, $8''$, $9'$, $9''$, $10'$, $11'''$, $17''$, $17'''$, $18'$	NR/CNR	28%
NR-0	Bridging: $10'''$, f_1		
CNR-1	S-open: 8 Bridging: $15'''$		
CNR-0	Bridging: $13'''$		
Complex	Bridging: $14'$, $14'''$	complex	2%
Fault-free with delay	S-open: 6 S-on: 9, 12 Bridging: f_2, f_3	delay	7%
Indeterminate	S-open: 2, 3, 5, 7 Bridging: $2''$, $3''$, $11'$	parametric	
Fault-free	S-open: 2, 3, 5, 7 S-on: 10, 11, 13, 14, 15, 16, 17 Bridging: $16'''$, f_4, f_5, f_6	----	

shown in Fig. 1. In the case of delay faults, the cell shows fault-free logic behavior, with increased delay for low to high or high to low logic transitions. Many faults cause the cell to exhibit a *complex behavior*; in most such cases, the output is a function of *CLK* and *D* input signals. All faults included in the enchanced fault model, as well as complex and delay faults, are considered to be logically testable for the purpose of evaluating test generation. For the complex faults and those covered by the enhanced model, a test sequence (t_1, t_2) can be found that can drive the faulty cell to a value different from that for the fault-free cell. The delay faults simply cause the output to be delayed. The indeterminate behavior is considered to be *parametric* because it requires current (I_{DDQ}) monitoring. Some stuck-on and bridging faults that cause fault-free behavior are also I_{DDQ} testable. Table I shows that the minimal stuck-at model covers 56% of the logically testable faults, while the enhanced model covers 91%.

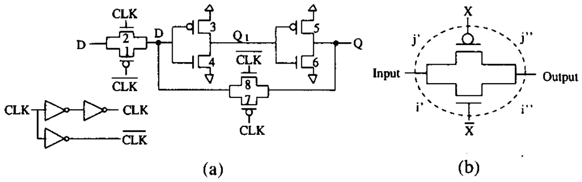


Fig. 3. The transmission gate latch.

B. The Transmission Gate Latch

The D-latch implementation termed *transmission gate latch* is shown in Fig. 3(a). This structure is used in some scan-path register designs [4], [7]. When two nodes are separated by a transmission gate, the node with stronger logic value dominates over the weaker one while the transmission gate is enabled. The two clock phases are assumed to be individually buffered, i.e., bridging of one clock signal does not affect the complementary clock signal.

Table II shows the behavior of the transmission gate in the presence of the bridging faults shown in Fig. 3(b). It can be used to evaluate the effect of faults in the transmission gates on the behavior of the latch. For example, the short between nodes Q and $\overline{CLK}$ (bridging fault $8'$) is modeled as a complex behavior. Note that the corresponding fault in a transmission gate is modeled as fault-free in the transmission gate model shown in Table II. This fault makes $\overline{CLK}$ to be low and therefore the forward transmission gate is not fully off. When CLK becomes 0 ($\overline{CLK} = 1$) and node Q is 0, the resulting level can drive the pMOS transistor of the forward transmission gate partly on. Thus the logic level at the node $D1$ is not predictable. Possible bridging faults such as $f_1 = (D, Q1)$ and $f_2 = (D, Q)$ are also considered. Table III shows that the minimal stuck-at fault model covers 53% of the logically testable faults, while the enhanced model covers 84%.

C. The Symmetric D-Latch

The third cell to be considered, the *symmetric D-latch*, is shown in Fig. 4. The clock is applied the same way as with the previous cell. Table IV shows the results under all possible transistor faults. Stuck-open faults in the forward branch (transistors 1, 2, 5, and 6) are detectable. The stuck-open faults in the feedback branch (transistors 3, 4, 7, and 8) are not detectable although the cell becomes dynamic under some input vectors. In such a case, the state of the latch is maintained by a charge stored at a node. Under normal clock frequency, however, a faulty

TABLE II
BEHAVIOR OF A TRANSMISSION GATE WITH BRIDGING FAULTS
(z-High Impedance State)

Inputs		Output behavior				
A	X	normal	i'	j'	i''	j''
0	0	0	0	0	0	1
0	1	z	z	z	1	0
1	0	1	0	1	0	1
1	1	z	z	z	1	0
Model		FF	s-a-0	FF	X	$\overline{X}$

TABLE III
BEHAVIOR OF THE TRANSMISSION GATE LATCH UNDER ALL POSSIBLE FAULTS

Output (Q)	Faults	Model	% of testable faults
H	S-on: 4 Bridging: $3', 4''', 5''', 6''$	minimal stuck-at	53%
$H \not\to L$	S-open: 3 Bridging: $2'$		
L	S-on: 6 Bridging: $3''', 4'', 5', 6'''$		
$L \not\to H$	S-open: 4 Bridging: $1', 7''', 8''', f_2$		
D	Bridging: $1''', 2'''$	feed-through	22%
$\overline{D}$	Bridging: f_1		
CLK	Bridging: $1'', 8''$		
$\overline{CLK}$	Bridging: $2'', 7''$		
NR-1	S-on: 3 Bridging: $7'$	NR/CNR	9%
CNR-1	S-on: 2		
Complex	Bridging: $8'$	complex	3%
Fault free with delay	S-open: 1, 2 S-on: 5, 8	delay	13%
Indeterminate	S-open: 5, 6 Bridging: $3'', 4', 5'', 6'$	parametric	
Fault free	S-open: 7, 8 S-on: 1, 7	– – – –	

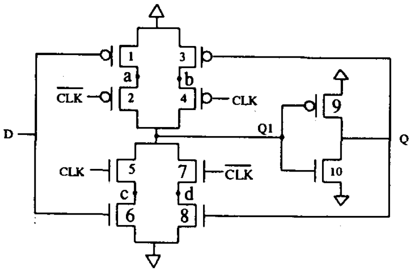


Fig. 4. The symmetric D-latch.

TABLE IV
BEHAVIOR OF THE SYMMETRIC D-LATCH UNDER ALL POSSIBLE FAULTS

Output (Q)	Faults	Model	% of testable faults
H	Bridging: $3', 9''', 10''$	minimal stuck-at	62%
$H \not\to L$	S-open: 1, 2, 10 S-on: 6, 7, 9 Bridging: $1', 1'', 2', 5''', 6''', 7''', f_2$		
L	S-on: 10 Bridging: $8'', 9', 10'''$		
$L \not\to H$	S-open: 5, 6, 9 Bridging: $1''', 4''', 5'', 6''$		
$\overline{D}$	Bridging: f_3	feed-through	11%
CLK	Bridging: $2'', 7'$		
$\overline{CLK}$	Bridging: $5', 4''$		
NR-1	Bridging: $3''', 8'$	NR/CNR	16%
NR-0	S-on: 8 Bridging: $3'', 4', 8'''$		
CNR-1	Bridging: $2''', 7''$		
CNR-0	S-on: 5		
Complex	Bridging: $6'$	complex	4%
Fault free with delay	S-on: 1, 4 Bridging: f_1	delay	7%
Indeterminate	Bridging: $9'', 10'$	parametric	
Fault free	S-open: 3, 4, 7, 8 S-on: 2, 3	– – – –	

behavior may not be observable. Bridging faults $f_1 = (a, b)$, $f_2 = (c, d)$, and $f_3 = (D, Q1)$ are included in Table IV. The table shows that the minimal stuck-at fault model covers 62% of the logically testable faults, while the enhanced model covers 89%.

D. The Dynamic Latch

The last SE cell considered is the simple dynamic SE cell, consisting of a pass transistor and an inverter as shown in Fig. 5. This cell will preserve data only as long as charge can be retained at node $Q1$. Typically, dynamic latches are used in designs that operate at a clock frequency fast enough to maintain the charge for the required period. This latch must be reloaded every clock period. Table V shows the behavior of this latch under transistor faults when the output is observed at node Q where two bridging faults, $f_1 = (D, Q)$ and $f_2 = (CLK, Q)$, are also included. Table V shows that *feedthrough* faults are more likely to occur in this cell than in the static SE's. This can be explained by the fact that *NR/CNR* cannot occur in this structure as there is no feedback. The minimal fault model covers 67% of the logically testable faults, while the enhanced fault model covers all 100%.

IV. Fault Modeling for Complex SE Cells

In this section, we examine the fault model for complex SE's, composed of more than one elementary SE cell. We analyze the master–slave cell, but the results can be applied to other complex SE cells as well. An accurate fault model for such circuits with explicit fault coverage can be obtained by mapping the proposed fault model for an elementary SE cell to a higher level of abstraction. Hence, a fault model that keeps the accuracy of the low level model with the primitives at the functional level can be achieved. Consider the master-slave cell shown in Fig. 6. The following discussion assumes Fig. 6(a) but is also applicable to the cell in Fig. 6(b). If the slave latch is faulty, the cell will show the same faulty behavior as that of the corresponding faulty elementary SE. However, mapping faults in the master latch to the output of the cell depends on the timing parameters of the control signals used. Consider the two clock phases shown in Fig. 7. The requirement for correct operation for this cell is

$$t_{\phi 1 \phi 2} \geq \max\ (d_{CQ}, g_{\phi 1 \phi 2}) + t_s \tag{1}$$

where $g_{\phi 1 \phi 2}$ is the gap between the falling edge and the rising edge as shown in Fig. 7, t_s and t_h are the setup and hold times of a single latch, and d_{CQ} is the clock-to-output propagation delay of the cell.

Consider the *data-feedthrough* fault in the $L1$ latch such that $Q1 = \Delta(D)$. $\Delta(\Delta \leq d_{DQ})$ is the transparent propagation delay during the transparent phase. Consider $t_{D_{ab}}$ to be the instant the input signal changes from a to b, where a, b $\in \{0, 1\}$ and $a \neq b$. If $t_{D_{ab}}$ occurs between the two successive rising edges of $\phi 1$ such that

$$t_{D_{ab}} \leq T_1 - t_s \tag{2}$$

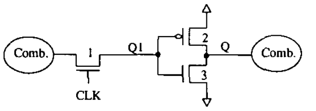
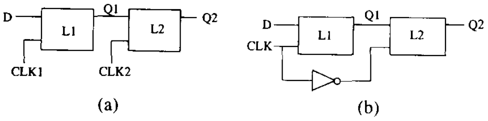


Fig. 5. The dynamic latch.

TABLE V
BEHAVIOR OF THE DYNAMIC LATCH UNDER ALL POSSIBLE FAULTS

Output (Q)	Faults	Model	% of testable faults
H	Bridging: $2''', 3'$	minimal	67%
H$\not\to$L	S-open: 1, 3 S-on: 2		
L	Bridging: $2', 3'''$	stuck-at	
L$\not\to$H	S-open: 2 S-on: 3 Bridging: $1'$		
D	Bridging: f_1	feed-through	33%
$\overline{D}$	S-on: 1		
CLK	Bridging: $1'''$ Bridging: f_2		
$\overline{CLK}$	Bridging: $1''$		
Indeterminate	Bridging: $2'', 3''$	parametric	



(a) (b)

Fig. 6. The master–slave cell.

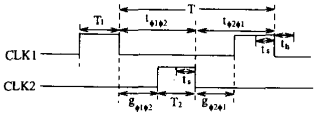


Fig. 7. The two clock phases.

and the data stay stable till the next clock period, then the behavior of the master–slave cell under the family master latch has the same behavior as that of the fault-free cell. However, if $t_{D_{ab}}$ occurs such that

$$T_1 + t_h \leq t_{D_{ab}} + \Delta \leq T_1 + t_{\phi1\phi2} - t_s \tag{3}$$

then logic b is latched at the falling edge of $\phi 2$ instead of logic a. Therefore, we have a *race-ahead* condition. This means that the change in input data may be observed at the output of the cell within the same clock period while in the fault-free cell, this change is observed during the following clock period. However, if $Q1 = \Delta(\overline{D})$, then the above observations are reversed, i.e., a *race-ahead* can be observed if inequality (2) above is satisfied, and the behavior is fault free if inequality (3) is satisfied.

If $L1$ latch becomes $NR - 1$ ($NR - 0$), it is obvious that the output of the cell is stuck-at-0(1). However, if $L1$ becomes $CNR - 1$ ($CNR - 0$), then the output will appear as stuck-at-0 (stuck-at-1) only under some conditions. In this case, the behavior is termed complex. Otherwise the faulty cell exhibits fault-free behavior.

A summary of the fault behavior of the master–slave cell for faults in the master latch is given in Table VI. The faults in the slave latch are not included because the faulty behavior of the master–slave cell for this case is identical to that of the slave latch. Timing dependent implies that the behavior depends on the time of input change relative to clock signal.

TABLE VI
MAPPING OF AN ELEMENTARY SE FAULT MODEL TO THE MASTER–SLAVE CELL FAULT MODEL

Behavior of faulty master latch	Behavior of master-slave latch	Model
Data-feed-through	timing dependent*	timing dependent
Clock-feed-through	stuck-at-0	stuck-at-0/1
Stuck-at-0/1	stuck-at-0/1	stuck-at-0/1
$NR - 1/0$	stuck-at-0/1	stuck-at-0/1
$CNR - 0/1$	complex	complex
Fault free	Fault free	– – – –

*Data-feedthrough in a single phase clock.

The observations here model the complex cell as a single primitive. When it is part of a large circuit, the slave latch will derive some combinational logic which in turn derives other complex cells. The clock-feedthrough and $NR - 0/1$ faults may appear as stuck-at-0/1. However, the data-feedthrough faults would generally require separate consideration.

V. FEEDTHROUGH IN SEQUENTIAL CIRCUITS

In this section, the detection of *feedthrough* faults in sequential circuits is considered. A sequential circuit is composed of combinational primitives and SE primitives. Functional fault models for such circuits can be obtained using the proposed fault model for the elementary SE cell. Such models retain the accuracy of the low level fault model of the primitives by allowing physical failures that cannot be modeled as stuck-at faults to be functionally characterized at a higher level. This can reduce the test generation effort because a low level failure can be tested by considering the corresponding higher level fault. Therefore, the number of elements to be considered is reduced since there is no need to consider test generation at the transistor level. Here, we examine the detectability of *feedthrough* faults in sequential circuits. A pipeline implementation of a sequential circuit is used in this section. The observations are also applicable when feedback is present. Here a *substate* contains a subset of the state variables and thus it can describe some or all of the flip-flops. A race-ahead involves change in the state of the machine as defined below [11].

Definition 4: A *race-ahead* occurs when a sequential circuit goes from substrate s_i to s_{i+2} in one clock period, whereas normally a transition from s_i to s_{i+1} occurs, followed by an s_{i+1} to s_{i+2} transition in the next clock period.

Consider the sequential circuit shown Fig. 8. C1 and C2 are combinational blocks, separated by pairs of latches triggered by signals $\phi 1$ and $\phi 2$. For the normal circuit, the propagation of a transition (new logical values) can be described by the following sequence.

1) $\phi 1 \downarrow$: a transition latched in $L11$.
2) $\phi 2 \downarrow$: corresponding transition (CT) latched in $L12$.
3) $\phi 1 \downarrow$: CT latched in $L21$ (after passing through $C1$).
4) $\phi 2 \downarrow$: CT latched in $L22$.

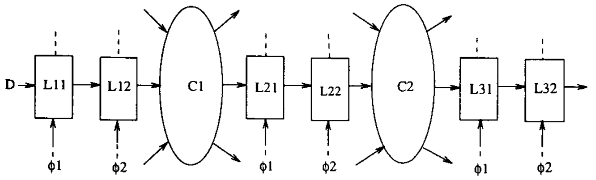


Fig. 8. Two-pipeline stage sequential circuit.

5) $\phi1 \downarrow$: CT latched in $L31$ (after passing through C_2).
6) $\phi2 \downarrow$: CT latched in $L\,32$.

Here, $\downarrow$ indicates the falling edge.

The major requirements for correct operation are

$$t_{\phi1\phi2} \geq \max\,(d_{CQ}, g_{\phi1\phi2}) + t_s \tag{4}$$

$$t_{\phi2\phi1} \geq \max\,(d_{CQ}, g_{\phi2\phi1}) + d_{c1} + t_s \tag{5}$$

$$T \geq d_{CQ_{\max}} + d_{DQ_{\max}} + \max\,(d_{c1}, d_{c2}) + t_s \tag{6}$$

where d_{c1} and d_{c2} are propagation delays through $C1$ and $C2$, respectively.

Now let us consider the case when the latch $L21$ has a data-feedthrough fault. When the $\phi2$ signal is active, there exists a combinational path $L12$-$C1$-$L21$-$L22$ from the output of $L11$ to the input of $L31$. The following sequence is possible.

1) $\phi1 \downarrow$: a transition latched in $L11$.
2) $\phi2 \downarrow$: CT latched in $L22$, provided the inequality (7) below is satisfied.
3) $\phi1 \downarrow$: CT latched in $L31$.
4) $\phi2 \downarrow$: CT latched in $L32$.

Thus, between two successive falling edges of $\phi1$ (i.e., within a single clock period), both $C1$ and $C2$ are traversed. This race-ahead can occur only if

$$t_{\phi1\phi2} \geq \max\,(d_{CQ}, g_{\phi1\phi2}) + d_{DQ} + d_{c1} + d_{L21} + t_s \tag{7}$$

where d_{L21} is the propagation delay of faulty latch $L21$, and d_{DQ} is the propagation delay through the transparent latch $L12$.

In some situations, the condition in inequality (7) may not be satisfied and the following sequence can occur.

1) $\phi1 \downarrow$: a transition latched in $L11$.
2) $\phi2 \downarrow$: CT does not arrive at $L22$ in time but is latched in $L12$.
3) $\phi1 \downarrow$: has no effect on $L21$, which is faulty.

The rest of the sequence is the same as steps 4–6 of the fault-free sequence. This suggests that in some cases higher propagation delays can mask the *data-feedthrough* faults.

Example: Consider the Gray code up–down counter shown in Fig. 9 as an example of a finite-state machine [11]. This is a double latch design using two non-overlapping clock signals.

When $F = 1$, the circuit displays on Z_1, Z_2 the modulo 4 Gray code representation of the number of positive clocks received. When $F = 0$, the circuit counts backward (modulo 4). The effects of four different faults are illustrated in Fig. 10, which gives the state diagrams in the presence of each of these faults. States A, B, C, and D represent the states corresponding to $Z_1Z_2 = 00$, 01, 11, 10, respectively. In Fig. 10(a), corresponding to $NR - 0$ of $L\,1$, transitions into states A and B are not possible. The behavior shown in Fig. 10(b) is similar except that the latch ($L1$) is $CNR - 0$. Fig. 10(c) illustrates the behavior when $L3$ is data-feedthrough. The state diagram shows that the fault results in *race-ahead*, i.e., the state in which the circuits go from state B to state D in one clock period instead of two. Fig. 10(d) shows the effect when $L3$ is clock-feedthrough, which appears as stuck-at-0. The observation here is that *clock-feedthrough* faults in the master latch appear as stuck-at faults because the slave will always latch a steady signal. Therefore, tests for stuck-at faults may detect such faults. However, a *clock-feedthrough* in the slave latch causes the output to follow the clock signal.

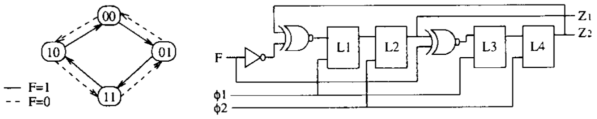
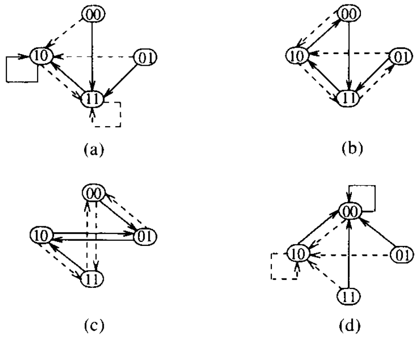


Fig. 9. The Gray code up–down counter.



(a) (b)

(c) (d)

Fig. 10. Behavior of the Gray code up–down counter under different faults. (a) L1 NR-0, (b) L1 CNR-0, (c) L3 data-feedthrough, (d) L3 clock-feedthrough.

VI. Conclusion

The effectiveness of the minimal fault model for elementary SE's is evaluated. An enhanced fault model is proposed which provides higher explicit fault coverage compared with that of the minimal fault model, as shown in Table VII. Higher level functional fault models for complex circuits using elementary SE's as primitives can be inferred from the proposed model, with higher fault coverage. This allows testing for low level failures that cannot be characterized as stuck-at-0/1 at the functional level without the need to consider the physical implementation of the circuit. Thus, the advantages of functional testing are retained with a higher coverage of low level

TABLE VII
SUMMARY OF THE RESULTS

Model	% of testable faults
Minimal stuck-at	(53-62)%
Feed-through	(7-33)%
NR/CNR	(9-28)%
Complex behavior	(2-4)%
Fault-free with delay	(7-13)%

failures. Test generation could be based on the change in the state-transmission graph (STG) of the complex circuit due to such faults. This can be used to enhance the existing testing techniques based on the changes in STG which consider stuck-at faults only.

REFERENCES

[1] M. Abramovici, M. A. Breuer, and A. D. Friedman, *Digital Testing and Testable Design*. Computer Society Press, 1990.

[2] J. A. Abraham and K. Fuchs, "Fault and error models for VLSI," *Proc. IEEE*, vol. 74, pp. 639-653, May 1986.

[3] W. K. Al-Assadi, Y. K. Malaiya, and A. P. Jayasumana, "Use of storage elements as primitives for modeling faults in sequential circuits," in *Proc. 6th Int. Conf. VLSI Design*, Jan. 1993, pp. 118-123.

[4] P. Banerjee and J. A. Abraham, "A multivalued algebra for modeling physical failures in MOS VLSI circuits," *IEEE Trans. Comput.-Aided Design*, vol. CAD-4, pp. 312-321, July 1985.

[5] M. K. Reddy and S. M. Reddy, "Detecting FET stuck-open faults in CMOS latches and flipflops," *IEEE Design and Test*, pp. 17-26, Oct. 1986.

[6] D. L. Liu and E. J. McCluskey, "A CMOS cell library design for testability," *VLSI Syst. Design*, pp. 58-65, May 4, 1987.

[7] R. Anglada and R. Rubio, "Functional fault models for sequential circuits," Research Rep. DEE-3, Elect. Eng. Dep., Polytechnical Univ. Catalunya, Barcelona, Spain, 1987.

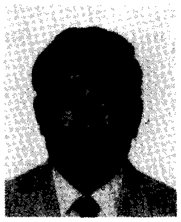
[8] K. J. Lee and M. A. Breuer, "A universal test sequence or CMOS scan registers," in *Proc. IEEE Custom Integrated Circuits Conf.*, 1990, pp. 28.5.1-28.5.4.

[9] R. Rajsuman, "Analysis, modeling and testing of major failure modes in MOS VLSI," Ph.D. dissertation, Dep. Elec. Eng., Colorado State Univ., 1988.

[10] V. D. Agrawal, S. K. Jain, and D. M. Singer, "Automation in design for testability," in *Proc. IEEE Custom Integrated Circuits Conf.*, 1984, pp. 159-163.

[11] W. K. Al-Assadi, Y. K. Malaiya, and A. P. Jayasumana, "Detection of feed-through faults in CMOS storage elements," in *Proc. NASA Symp. VLSI Design*, Oct. 1992, pp. 7.2.1-7.2.5.

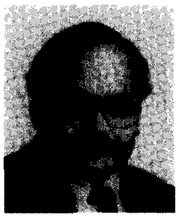
[12] E. J. McCluskey, *Logic Design Principles*. Englewood Cliffs, NJ: Prentice-Hall, 1986, ch. 8



Waleed K. Al-Assadi received the B.Sc. and M.Sc. degrees in electrical engineering from the University of Baghdad, Iraq, in 1978 and 1980, respectively. He is currently working towards the Ph.D. degree in the Electrical Engineering Department at Colorado State University.

His research interests include design and testing of VLSI systems, design automation, and fault-tolerant computing. He served as a technical program committee member for the 5th NASA Symposium on VLSI Design, November 1993.

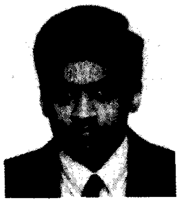
Mr. Al-Assadi is a member of Eta Kappa Nu.



Yashwant K. Malaiya (SM'89) received the B.Sc. degree from Government Degree College, Damoh, the M.Sc. degree from the University of Saugor, and the M.Sc. Tech. degree from BITS, Pilani in India. In 1978, he received the Ph.D. degree in electrical engineering from Utah State University.

He was with the State University of New York at Binghamton during 1978-1982. He is now a Professor in the Computer Science Department and the Electrical Engineering Department at Colorado State University. He has published widely in the areas of fault modeling, software and hardware reliability, testing, and testable design. He has also been a consultant to industry.

Dr. Malaiya was the general chair of the 24th International Symposium on Microarchitecture and the 6th International Conference on VLSI Design. He is the general chair of the 4th International Symposium on Software Reliability Engineering. He has co-edited the IEEECS Technical Series books *Software Reliability Models, Theoretical Developments, Evaluation and Applications* and *Bridging Faults and IDDQ Testing*. He was a guest editor of special issue of the *IEEE Software* and *IEEE Design and Test of Computers* magazines. He has been the chair of the TC on Microprogramming and Microarchitecture. He is the chair of the software test subcommittee of TTTC and a vice-chair of the TCSE subcommittee on software reliability engineering. He is a member of the IEEECS TAB Executive Committee and the IEEECS Awards Committee.



Anura P. Jayasumana (SM'90) received the M.S. and Ph.D. degrees in electrical engineering from Michigan State University in 1982 and 1984, respectively, and the B.Sc. degree in electronics and telecommunication engineering, with first class honors, from the University of Sri Lanka, Moratuwa, in 1978.

He is an Associate Professor in the Electrical Engineering Department and the Computer Science Department at Colorado State University. He worked as an electrical engineer at the National Engineering Research and Development Center of Sri Lanka and as an assistant lecturer in the Electronics and Telecommunication Engineering Department at the University of Sri Lanka from 1979 to 1980. His research interests include VLSI design and testing, design automation, and data communication networks. He has served as a consultant to NCR Microelectronic Division, Fort Collins, CO, since 1988.

Dr. Jayasumana is a member of Phi Kappa Phi, the IEEE Computer Society, and the IEEE Communications Society. He is an associate editor of *IEEE Network*. He was the winner of the award for the best student in electrical engineering at the University of Sri Lanka, Moratuwa, in 1978, and the College of Engineering Outstanding Academic Achievement Award, Michigan State University, in 1982 and 1983. In January 1990, he received the Outstanding Faculty of the Year Award from the Mountain States Council of the American Electronics Association.