Data-feed-through Faults in Asynchronous Circuits

Waleed K. Al-Assadi, Ding Lu, Carol Q. Tong,
Anura P. Jayasumana, Yashwant K. Malaiya

Technical Report CS-93-120

September 8, 1993

Data-feed-through Faults in Asynchronous Circuits[*]

Waleed K. Al-Assadi, Ding Lu
Carol Q. Tong, Anura P. Jayasumana, Yashwant K Malaiya [†]
Department of Electrical Engineering
[†] Computer Science Department
Colorado State University
Fort Collins, CO 80523

Abstract - It is often assumed that the faults in storage elements (SEs) 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. A more accurate higher level fault model for elementary SEs used in asynchronous circuits is presented. This model offers better representation of the physical failures. It is shown that the stuck-at model may be adequate if only modest fault coverage is desired. The *enhanced* model includes some common fault behaviors of SEs that are not covered by the the stuck-at model. These include *data-feed-through* behaviors that cause the SE to be combinational. Fault models for complex SE cells can be obtained without a significant loss of information about the structure of the circuit. The implications of the *data-feed-through* faults on the behavior of asynchronous circuits are considered.

1 Introduction

Functional fault modeling is an effective approach to handle the complexities of large digital circuits. A functional fault model hides the complex fault behavior and prese
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 le
description. However it has been shown in some situations that a simple functional model may not adequately represent a significant fraction of failures. When this is the case, t

*This work was supported partly by a SDIO/IST funded project monitored by ONR.

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 t considerably reducing the test generation effort. A fault model can be termed adequate i it explicitly covers (i.e. coverage is guaranteed for) a major fraction, say x%, of all faults [2]. 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 wit low *explicit coverage* is likely to be inadequate.

A good strategy is to obtain a functional fault model for logic blocks derived from the t physical structure of the circuit. This requires that accurate fault models for primitive b such as elementary storage elements (SEs) be considered. Although test considerations a 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 fault simulation efforts significantly.

The elementary SEs are the basic primitives in complex logic blocks like registers, fini state machines, and static memory blocks. This paper examines the major transistor-leve faults for two elementary SEs used basically in asynchronous circuits. The behavior o each cell under the above faults is analyzed to evaluate possible functional fault mode Results for elementary SE cells are extended to characterize complex SE cells. In section the minimal stuck-at model and the proposed enhanced model are described. SE cells are examined in section 3 for all possible transistor-level faults to seek a fault model with fault coverage.

2 Fault Modeling of Elementary SE Cells

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

To examine a SE cell, in general, an input sequence is required rather than a single inpu vector. Let T=$\{t_1 \ldots t_n\}$ be the set of all possible input combinations and $R(t_i, s)$ be 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 combina

and previous states. A multivalued logic representation is used to better represent volt
levels that are not exactly logic 1 (hard 1) or logic 0 (hard 0) [3]. Here *high* level (H)
corresponds to both 'hard 1' and 'soft 1', and *low* level (L) corresponds to both 'hard 0' an
'soft 0' [4]. A fault that causes the SE output to be L (H) for all $t_i \in T$, regardless 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 i
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 *data-feed-through* as defined in [2].

Definition 1: A faulty SE cell is said to be *data-feed-through* when its behavior becomes
combinational such that $R(s_i) = f(y)$ for each $t_i \in T$, where y is the data part of t_i.
For example, for a NAND pair latch, y is a double element vector corresponding to A and
B inputs.

Some recent papers address the detection of several physical failures in CMOS syn
chronous latch cells [1], [2], [4], [7]. A comprehensive faults model for such latches is presented in
[2]. The proposed *enhanced* fault model is presented. It was observed that *data-feed-throu*
faults cause a *race-ahead* condition in sequential circuits, i.e. the circuit reaches a st
clock period too early [8]. In this paper, we investigate the *enhanced fault model* for SEs
that are used in building asynchronous circuits. The enhanced model includes faults tha
cause *data-feed-through* and problems of *non-retention* of logic level behaviors as well as
stuck-at faults. Such faults can be detected by monitoring logical levels. Hence they
termed logically testable.

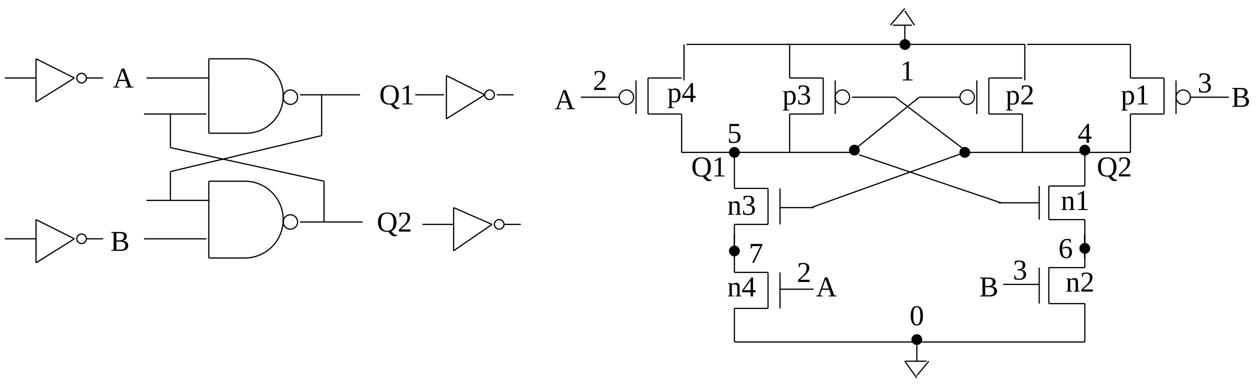
3 Detailed Examinations of the Elementary SE Cells

In this section, a detailed examination of two different elementary asynchronous SEs i
presented. Each cell is examined for all possible transistor faults. Results obtained
lytically 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 adequate
described. Both the *minimal* and the *enhanced* fault models are examined for the effective-
ness in representing the functional faults. Because of the transistor sizing and techn
used, '0' dominates if two nodes are bridged. All possible bridging faults between nodes
the same well are considered. We use (x, y) to indicate a bridging fault between nodes x
and y. Bridging faults between internal nodes of different wells are not included becaus

the probability of having such faults is very small. Analysis assumes that a bridging fa
corresponds to a *hard short*. 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, a SE
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 associa
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 a
delay faults. Faulty behavior of two SE cells are summarized next.

3.1 The NAND pair latch

The cell is shown in Figure 1. A single-rail output is considered here and the output i
observed at $Q1$. Similar but somewhat more complicated results for double-rail case can be
obtained. In fault free cell, vector $AB = 11$ causes both $Q1$ and $Q2$ nodes to retain thei
previous logical values and the cell remains static, while vector $AB = 00$ is to be avoid
because it causes race problem in the cell. However, only vectors $AB = \{01, 10\}$ make the
cell in the transparent phase, i.e. can derive the cell to a known logic values.



(a) The gate level (b) The transistor level

Figure 1: The NAND-pair cell

Table 1 shows the faulty behavior of the cell for transistor-level faults. The results
that some of these faults cause enhancement in the supply current (I_{DDQ}) because a direct
path between V_{dd} and V_{ss} under some test vectors is established. Consider stuck-open fault
in transistors n1 or n2 or both. When vector $AB = 01$ is applied, a direct path between V_{dd}

and V_{ss} is formed. The same observation is applied for stuck-open fault in transistor n3 o
n4 or in both when vector $AB = 10$ is applied. The results show that only two stuck-open
faults show fault free behavior. Consider stuck-open fault in p3. This fault causes the
to exhibit dynamic behavior when vector $AB = 11$ is applied and the cell is initialized t
logic 1. Due to this fault, node $Q1$ cannot keep its logical value of 1 indefinitely since
only way to refresh node $Q1$ if this vector applied is through transistor p3. Stuck-open fa
in transistor p2 causes similar change when the same vector applied and the cell initiali
to logic 0. These two fault are considered undetectable if tests are applied at normal ra
However, if tests are applied much slower than normal rate, which is unusual in testing
then when vector $AB = 11$ is applied, then the node $Q1$ capacitance can discharge to bring
node $Q1$ to logic 0. This method of detection may also be unreliable because the voltage
on floating nodes may settle down at an inderminate value rather than at logic 0. Hence
these two faults are regarded undetectable. All stuck-on faults cause enhancement in I_{DDQ}
compared with the fault free case.

Testing the cell for bridging faults reveal the importance of the multivalued algebra. T
results are given in Table 2. The indeterminate value is observed when both nodes $Q1$ and
$Q2$ are shorted. This fault can cause charge sharing if vector $AB = 11$ is applied. Such
behavior can only be detected by observing the supply current I_{DDQ}. Bridging fault between
nodes 7 and 6 shows stuck-at behavior depending on the initial conditions. Therefore thi
fault is modeled as stuck-at regardless of the logical values at nodes $Q1$ and $Q2$. Simila
bridging fault between input nodes A and B has consequences assuming 0 dominance. If
input vector $AB = 01$ is the initialization vector, then $Q1$ is always at logic high (H) a
$Q2 = A \oplus B$, however the observation is reverse-versa if vector $AB = 10$ is applied first
Therefore for the sake of fault coverage, we model this fault as stuck-at.

Table 1 shows that only 18 faults out of 38 faults (i.e. 47%) are modeled as stuck-at an
17 faults (i.e. 45%) turn the cell combinational. Therefore the *enhanced* fault model woul
cover 92% of the logically testable faults. Testing this latch using robust tests is not t
because test patterns depend on both combination of primary inputs (A and B) and also on
the state variables, which are not directly controllable and dependent on change of prima
inputs. Consider stuck-open fault in transistor p4. In general to test for a stuck-open f
a two-pattern test, the initialization pattern and the test pattern are required. To tes
this fault, node Q1 has to be initialized to logic 0 by applying $AQ2 = 11$. $Q2$ however is
function of A and $Q1$ and can be driven to 1 by making either $B = 0$ or $Q1 = 0$. For this fault,
only possible way is to make $B = 0$ always. Therefore to ensure the robustly of the test, a
3-pattern test; $AB = \{10, 11, 01\}$ instead of a two-pattern test is required. This shows

the test is not trivial robust because we have to take into consideration the state varia
or outputs of the latch into consideration.

Table 1: Faulty behavior of the NAND pair latch under possible faults

Bridging fault	**Logical behavior**		**Model**
	$Q1$	$Q2$	
Stuck-open: p1 Stuck-on: n2 Bridging: (1,3), (6,0), (4,7)	H$\not\rightarrow$L	L$\not\rightarrow$H	stuck-at
Stuck-open: p4 Stuck-on: n4 Bridging: (7,0), (1,2), (5,6)	L$\not\rightarrow$H	H$\not\rightarrow$L	
Stuck-open: n3, n4 Stuck-on: p3, p4 Bridging: (2,7)	H$\not\rightarrow$L	$\overline{B}$	
Bridging: (1,5) Bridging: (6,7), (2,3)	H $----$	$\overline{B}$ $----$	
Stuck-open: n1, n2 Stuck-on: p1, p2 Bridging: (3,0), (3,6)	$\overline{A}$	H$\not\rightarrow$L	data-feed-through
Bridging: (2,5)	A	$\overline{A}+\overline{B}$	
Stuck-on: n3 Bridging: (2,4), (3,7)	$\overline{A}$	A	
Bridging: (5,7)	$\overline{A}$	A+$\overline{B}$	
Stuck-on: n1	$\overline{B}$	B	
Bridging: (1,4)	$\overline{A}$	H	
Bridging: (4,6), (3,5), (2,6)	B	$\overline{B}$	
Bridging: (3,4)	$\overline{A}+\overline{B}$	B	
Bridging: (4,5)	Indeterminate	Indeterminate	parametric
Stuck-open: p2, p3	fault-free†	fault-free†	----

†: The cell turns into dynamic under vector $AB=11$

3.2 The C-element

The C-element shown in Figure 2 is the storage element used in self-timed asynchronous
circuits. Such circuits like a pipeline interconnection circuit that controls data tr

between computation blocks, which is basically a half or full handshake [1], [10]. This circuit [9
is a dynamic cell because there is no feedback in the cell and the output is observed at noc
C. Its logic function can be described by the Boolean equation $C = AB + AC' + BC'$, where
C is the present state and C' the previous state. Hence only two vectors $AB = \{11, 00\}$
make the cell in the transparent phase, while vectors $AB = \{01, 10\}$ make the cell latch i
previous value.

The cell is examined to verify the effectiveness of the stuck-at model. It was observe
that this model can not cover all the possible physical defects within the element. Actual
most of the defects within the cell like transistor stuck-on, transistor stuck-open and bri
between internal nodes have some other faulty behavior which can not be interpreted by the
stuck-at fault model.

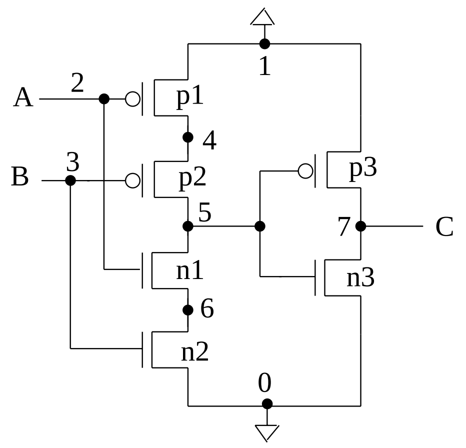


Figure 2: The C-element

The results given in Table 3 show that some faults cause the cell to exhibit a behavio
change in the latch phase, while still functioning properly in the transparent phase. *Co
tional no-retention* of logic 1/0 ($CNR - 1/0$) behaviors are observed. The definition is gi
below [2]:

Definition 2: Consider a SE cell in the state $Q = 1$ in the transparent phase, the cell
exhibits *conditional non-retention* ($CNR - 1$) behavior if the cell fails the latch logic
instead Q turns to logic 0, i.e., $R(t_i, 1) = 0$, where t_i is an input vector such that $t_i \in$ latch
phase vectors. ($CNR - 0$) is defined similarly.

As an example, consider stuck-on fault in transistor p1. If the cell is at logic 1 and ve
$AB = 10$ is applied, then the cell is unable to latch logic 1 as in the fault free case, but
vector will turn the cell to logic 0. therefore the behavior is modeled as $CNR - 1$. The tab

shows also that the cell become combinational under several faults, and therefore they a
modeled as *data-feed-through*. Some faults cause the cell to be *parametric*, where the logi
value of the cell is indeterminate and logical testing cannot be used. Such faults can o
be detected by monitoring the supply current (I_{DDQ}), which in the presence of the fault is
many orders higher than the fault free current. This enhancement in I_{DDQ} is due to the
forming of conducting paths between V_{dd} and V_{ss} under some faults.

Table 3 shows that among 33 possible defects within the C-element, only 18 (i.e. 53%)
can be modeled by the stuck-at fault model, while the enhanced model covers 13 faults more
(i.e. 91%). This means that 100% of the logically testable faults are covered by the enhanc
fault model, while only 3 faults can be tested by monitoring the supply current I_{DDQ}

Table 2: Faulty behavior of the C-element

Fault	**logical Behavior**	**Model**
Bridging: (3,4), (1,7), (5,0)	stuck-at-1	
Bridging: (2,6), (7,0), (1,5)	stuck-at-0	
Stuck-open: p1, p2, n3 Bridging: (1,2), (1,3), (2,4)	$H \not\to L$	stuck-at
Stuck-open: p3, n1, n2 Bridging: (2,0), (3,0), (3,6)	$L \not\to H$	
Stuck-on: p1, p2 Bridging: (1,4), (4,5)	CNR-1	CNR
Stuck-on: n1, n2 Bridging: (5,6), (6,0)	CNR-0	
Bridging: (2,3)	$C = AB$	
Bridging: (2,5)	$C = \overline{A}$	
Bridging: (3,5)	$C = \overline{B}$	data-feed-through
Bridging: (2,7)	$C = A$	
Bridging: (3,7)	$C = B$	
Stuck-on: p3, n3 Bridging: (5,7)	indeterminate	parametric

4 Data-feed-through Faults in Asynchronous Circuits

In this section the detection of *feed-through* faults in sequential circuits is consider
asynchronous sequential circuit is composed of combinational primitives and SE primitive
Functional fault models for such circuits can be obtained using the proposed fault mode
for the elementary SE cell. Such models retain the accuracy of the low level fault mode

of the primitives, by allowing physical failures that cannot be modeled as stuck-at faul
be functionally characterized at a higher level. This can reduce the test generation ef
because a low level failure can be tested by considering the corresponding higher level fa
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 implications of *d*
feed-through faults in asynchronous sequential circuits. Here a *substate* contains a subs
the state variables, thus it could describe some or all of the storage elements of the cir
A race-ahead involves change in the state of the machine as defined below [1, 12].

Definition 3: A *race-ahead* occurs when a faulty sequential circuit starts from substate s_i
and back to the same substate in y transitions such that $y < x$, where x is the number of
transitions in the fault-free circuit.

Example 1: Consider the Gray code up-down counter shown in Figure 3 using two nand-pair
latches as an example of an asynchronous sequential circuit.

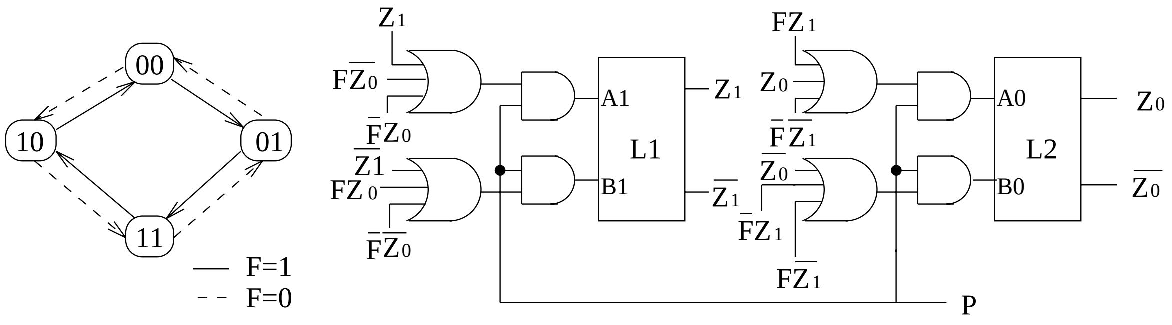


Figure 3: The up-down Gray code counter

When $F = 1$, the circuit displays on Z_1Z_0 the modulo 4 Gray code representation of
the number of positive control pulses received (P in Figure 3). When $F = 0$, the circuit
counts backward (modulo 4). The effects of two different faults are illustrated in Figur
4, which gives the state diagrams in the presence of each of these faults. States A, B,
and D represent the states corresponding to $Z_1Z_0 = 00, 01, 11, 10$ respectively. Consider $L1$
becomes data-feed-through such that $Z_1 = A1$. This means that latch $L1$ becomes completely
combinational, i.e. the data inputs $A1$ and $B1$ are not processed as individual events and t
latch is no longer a storage element. Therefore the events where the cell latches its prev
value are not available, instead at that events, the latch behaves combinational. Figure 4
shows the state diagram corresponding to this fault. Figure 4(b) corresponds to the sam
fault in $L2$. Consider $L1$ stuck-at-1, then the circuit will stuck to state 10 when $F = 1$, an

to state 11 when $F = 0$ as shown in Figures 4(c) and 4(d) respectively.

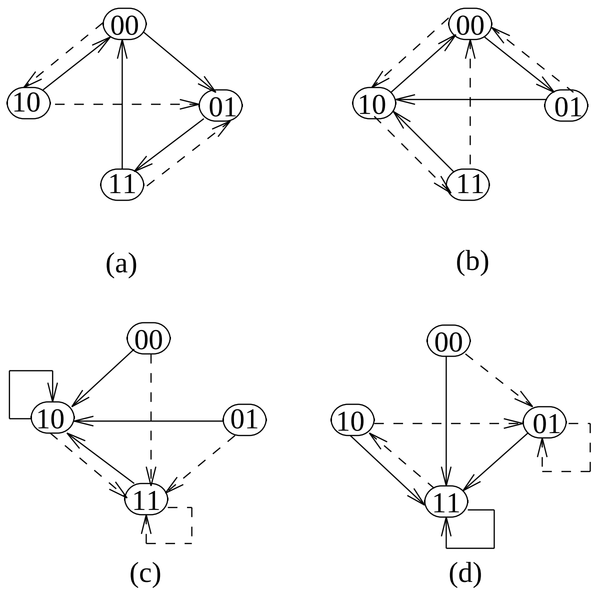


Figure 4: Faulty behavior of the up-down Gray code counter (a) $I1$ data-feed-through ($Z_1 = \overline{A1}$), (b) $I2$ data-feed-through ($Z_0 = \overline{A0}$) (c) $I1$ stuck-at-1, (d) $I2$ stuck-at-1

Example 2: Another example is the handshaking circuit implemented by the C-elements shown in Figure 5. It was claimed that the circuit is self-diagnostic for the stuck-at f model [9]. Since the circuit is semi-modular, or every operating cycle in its state gra including all possible transitions (see Figure 6), a stuck-at fault will halt the oper Therefore, the detection of the stuck-at fault can be guaranteed. This may not be true unfortunately, for the other faults we modeled in the last section.

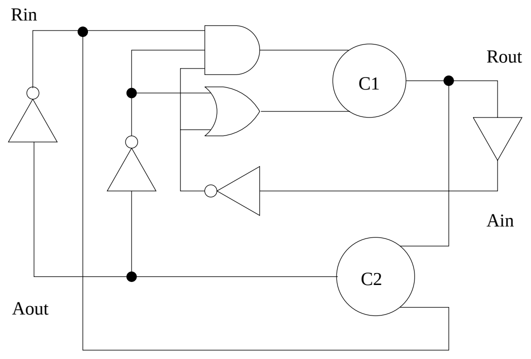


Figure 5: A handshaking circuit

As an example, consider the date-feed-through fault in which the two inputs of C2 in Figure 5 are shorted. Because of this fault, C-element C2 is reduced to an AND gate as we discussed in the last section. The effect of this fault can be seen from the faulty state gr in Figure 7. Two new states are possible and several new transitions are created. Unlike t stuck-at fault, this fault may not be self-diagnostic because the new state graph is no lo semi-modular and circuit can have such operating cycle as [$Ain^+, Rin^-, Aout^-, Rin^+$]. Moreover, this date-feed-through fault may mask the detection of some other stuck-at faul

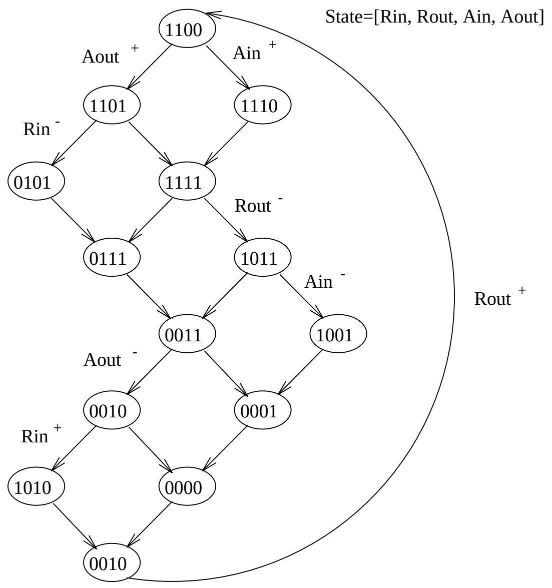


Figure 6: State graph of the handshaking circuit

5 Conclusion

The effectiveness of the minimal fault model for two basic SEs used in asynchronous circuits is evaluated. The enhanced fault model for the two cells is proposed, which provi higher explicit fault coverage compared to the minimal fault model. Higher level functio fault models for complex circuits using the two cells considered in this paper as primitives be inferred from the proposed model, with higher fault coverage. This allows the testing low level failures that cannot be characterized as stuck-at-0/1 at the functional level wi the need to consider the physical implementation of the circuit. Thus the advantages o functional testing is retained with a higher coverage of low level failures. Test gener could be based on the change in the state-transition graph of the complex circuit due t such faults. This can be used to enhance the existing testing techniques for self-timed asynchronous sequential circuits based on the changes in state transition graph which present only consider stuck-at faults only.

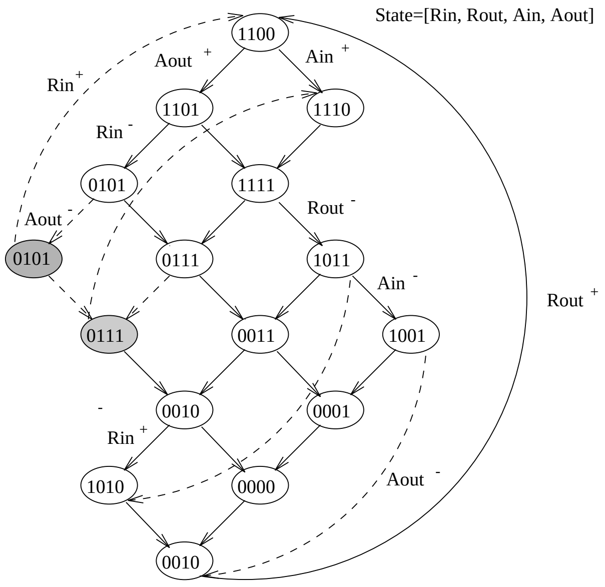


Figure 7: Effect of a date-feed-through fault on The C-element

References

[1] W. K. Al-Assadi, Y. K. Malaiya and A. P. Jayasumana, " Use of Storage Elements as Primitives for Modeling Faults in Sequential Circuits," *Proc. 6th. Int. Conf. on VLS Design*, pp. 118-123, January 1993.

[2] W. K. Al-Assadi, Y. K. Malaiya and A. P. Jayasumana, "Faulty Behavior of Storage Elements and its Effects on Sequential Circuits," To Appear in *IEEE Transaction on VLSI Systems*.

[3] P. Banerjee and J. A. Abraham, "A Multivalued Algebra for Modeling Physical Failures in MOS VLSI Circuits," *IEEE Transaction on Computer-Aided Design,* vol. CAD-4, no. 3, pp. 312-321, July 1985.

[4] M. K. Reddy and S. M. Reddy, "Detecting FET Stuck-Open Faults in CMOS Latches and Flipflops," *IEEE Design and Test,* pp. 17-26, October 1986.

[5] D. L. Liu and E. J. McCluskey, "A CMOS Cell Library Design for Testability," *VLSI Systems Design,* pp. 58-65, May 4, 1987.

[6] R. Anglada and R. Rubio, "Functional Fault Models for Sequential Circuits," *Research Report DEE-3*, Electronic Engineering Department, Polytechnical University Catalunya, Barcelona 1987.

[7] K. J. Lee and M. A. Breuer, "A Universal Test Sequence or CMOS Scan Registers," *Proc. IEEE Custom Integrated Circuits Conf.* pp. 28.5.1-28.5.4., 1990.

[8] W. K. Al-Assadi, Y. K. Malaiya, and A. P. Jayasumana, "Detection of Feed-Through Faults in CMOS Storage Elements," *Proc. NASA Symposium on VLSI Design*, pp. 7.2.1-7.2.5, October 1992.

[9] T. H.-Y. Meng, R. W. Brodersen and D. G. Messerschmitt, "Automatic synthesis of Asynchronous Circuits from High-Level Specifications", *IEEE Trans. Computer-Aided Design*, vol. 8, pp. 1185-1205, 1989.

[10] P. A. Beerel and T. H.-Y. Meng, "Semi-Modularity and Self-Diagnostic Asynchronous Control Circuits", *Advanced Research in VLSI* (MIT press 1991), pp. 118-132.