Fan-in- and fan-out-factor oriented BIST design for sequential machines

S. Roy, U. Maulik, S. Bandyopadhyay, S. Basu and B.K. Sikdar

Abstract: BIST design for sequential circuits is a difficult enterprise. The difficulty stems from the lack of uniformity in reachability and emitability of machine states. The paper introduces a BIST-quality metric termed the FiF-FoF (fan-in-factor and fan-out-factor) defined on FSM-states. Based on the FiF-FoF analysis, an efficient synthesis scheme is presented that ensures all state codes of FSM may appear with uniform likelihood at the present state lines during the test phase. The uniform mobility of states ensures higher fault efficiency in a BIST structure of the circuit. Extensive experimentation on MCNC benchmarks and randomly generated large FSMs shows that the proposed scheme improves the fault efficiency of sequential circuits significantly, with marginal area overhead.

1 Introduction

The field of testing has to adapt itself to the phenomenal changes at all levels of VLSI design during the submicron era. The test research community has put much higher emphasis on evolving innovative built-in self-test (BIST) structures [1, 2]. In general, pseudorandom pattern generators (PRPGs) report poor fault efficiencies in sequential circuits and in effect, efficient BIST design for sequential circuits has become difficult to accomplish [1, 2]. Several attempts were made to improve the fault efficiency in a BIST structure for sequential circuits: the modification of flip-flops to make the set of machine states reachable [3, 4], improving the quality of test pattern generators [5–7], transforming the patterns generated by a PRPG [8] are significant among them. In [9, 10] an FSM synthesis scheme based on degree-of-freedom (DOF) analysis in FSM-states targeting BIST quality and gate area has been presented. A detailed comprehensive study on BIST for sequential circuits is discussed in [11] that proposes BIST with minimum overhead.

In this paper we propose a new metric, termed the FiF-FoF (fan-in-factor and fan-out-factor), to characterise an FSM state with respect to its BIST quality. A state encoding scheme, based on the FiF-FoF analysis of an FSM, is then presented that makes all state codes equally likely to appear on present state (PS) lines during the test phase. The uniform mobility of states results in higher fault efficiency.

Paper first received 15th May 2002 and in revised form 5th February 2003

S. Roy and U. Maulik are with the Department of Computer Science & Technology, Kalyani Govt. Engineering College, Kalyani, India 741235

S. Bandyopadhyay is with Machine Intelligence Unit, Indian Statistical Institute, 203 B.T. Road, Calcutta 700035, India

S. Basu and B.K. Sikdar are with the Department of Computer Science & Technology, Bengal Engineering College (D U), Howrah, India 711103

2 Preliminaries

The general structure of a synchronous sequential machine (FSM) is shown in Fig. 1. It consists of a combinational circuit CC and the system register SR. The outputs y_1, $y_2, \ldots, y_k$ from the k memory elements of SR define the PS of the machine.

Built-in self test techniques for synchronous sequential circuits are proposed in [3–6]. The BIST structure developed for fully or partial scanned sequential circuits reconfigures the circuit flip-flops into scan registers and make them a part of the BIST test pattern generator (TPG) and signature analyser (SA), in test mode. If the combinational logic block (CC) of the sequential machine is not tested separately, then there is no need to configure the circuit flip-flops (SR), that is, the SR left unmodified. This structure requires two test registers: the PIs of the circuit are fed from the output of a pattern generator and the POs are fed into the signature analyser. Since the circuit flip-flops remain unmodified, this allows at-speed testing of the CUT under its normal functioning and saves the hardware overhead of reconfiguring the flip-flops. In this work the TPG is designed with a cellular automata (CA) [12] based pseudo random pattern generator (PRPG).

3 Causes of low BIST quality

The root causes of low BIST quality of a sequential machine can be ascribed to three kinds of FSM-states, *viz.*, the unreachable states, the hard-to-reach states, and the hard-to-exit states [9, 10] in a BIST structure. In the remaining parts of this Section the aforementioned factors are explained. Subsequent sections present the solutions to eliminate the hurdles due to these shortcomings.

3.1 Unreachable states

In the state transition graph (STG) of an FSM if there are n states, then at least k flip-flops are required to encode the n states, where $2^{k-1} < n \leq 2^k$. Out of 2^k state codes, only n of them will be assigned to the states of the FSM. The remaining $(2^k - n)$ codes are kept unused. These unused

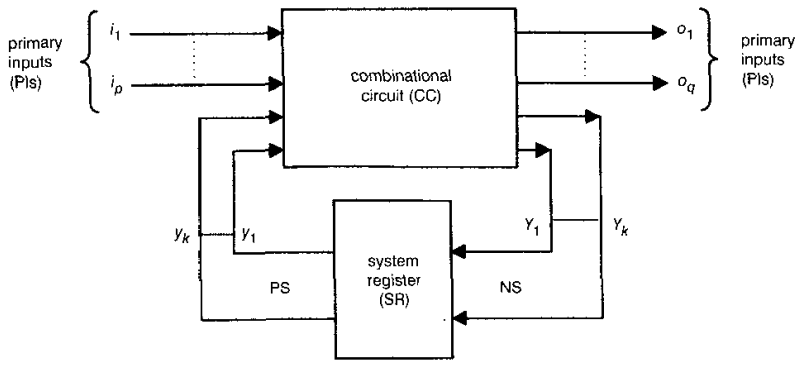


Fig. 1 *Structure of sequential machine*

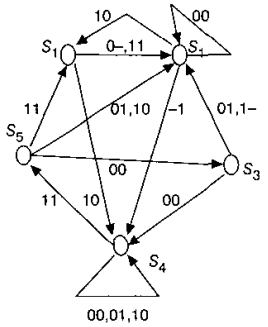
state codes are referred to as the unreachable states of FSM. Since the FSM never attains an unreachable state for any input sequence, the corresponding state code will never appear on the PS lines. This fact restricts the testability of the sequential machine. It might be the case that such a code is required to test some faults of the combinational logic block CC (Fig. 1).

Example 1: Fig. 2*b* shows a random state assignment for the sequential machine shown in Fig. 2*a*. The codes 000, 011 and 110 are not assigned to any states. Therefore these are the unreachable states of the given FSM, for the said state code assignment.

3.2 Hard-to-reach states

Quite often an FSM contains large number of states having very few transitions falling on them. These are called the hard-to-reach states. During circuit functioning these hard-to-reach states will be scarcely arrived at and consequently, the bit patterns for those hard-to-reach state code will seldom appear on the PS lines. This adversely affects the randomness of the test patterns at the input lines of the combinational (CC) part of the sequential machine. The result, again, is low fault coverage with the PRPG-based TPG.

Example 2: In Fig. 2*a* there is only one incoming transition on the state S_3, whereas there are four outgoing transitions from this state. Therefore S_3 is recognised to be a hard-to-reach state.



a

State	Code
S_1	100
S_2	010
S_3	001
S_4	111
S_5	101

Unused codes
000
011
110

b

Fig. 2 *Sequential machine M_1 and arbitrary state encoding for it*
a State transition diagram
b Random state encoding

3.3 Hard-to-exit states

There are states of an FSM with a large number of self loops. These states act like a sink in the sense that once the FSM reaches such a state, it tends to remain there for an indefinite period of time. Such states with a large number of self loops are called hard-to-exit states. Getting stuck at a specific state implies appearance of the same test patterns repeatedly on the PS lines over consecutive clock cycles. The ultimate effect, again, on testability is poor fault coverage.

Example 3: In Fig. 2*a* the state S_4 has only one outgoing transition from it whereas, there are four incoming transitions on this state. Hence, S_4 can be recognised as a hard-to-exit state.

4 FiF-FoF analysis

The basic criterion of obtaining a sequential machine with improved BIST quality is to make all state codes appear at the PS lines during circuit testing and make them appear with equal probability. To achieve this one has to ensure that the following conditions are satisfied during circuit testing:

a The unreachable state codes should appear at the PS lines.
b The hard-to-reach states should be made easy to reach.
c The hard-to-exit states should become easy to exit.

Before going into the details of the scheme to achieve these criteria, we define a few terms used throughout the paper.

4.1 FiF-FoF: A new BIST metric

The concept of FiF-FoF is presented in this subsection along with an example. The idea behind the FiF-FoF analysis of an FSM is to quantify the merit of its states from the point of view of BIST quality. Particularly, the FiF value of an FSM state denotes the relative ease with which it can be entered into, while the FoF value expresses the ease of leaving that state.

Definition 1: For each state S of an FSM, *reachability*(S) is the number of edges incident on S from other states. The self loops are not counted in *reachability*(S) analysis.

Definition 2: For each state S of an FSM, *emitability*(S) is the number of edges exit from S. The self loops are not counted as they fall on themselves.

Definition 3: For each state S, fan-in-factor of S, denoted as $FiF(S)$, is the ratio of $reachability(S)$ and $emitability(S)$. Therefore $FiF(S) = reachability(S)/emitability(S)$.

Definition 4: For each state S, fan-out-factor of S, denoted as $FoF(S)$, is the ratio of $emitability(S)$ and $reachability(S)$. Therefore $FoF(S) = emitability(S)/reachability(S)$.

It may be noted that for each state S of an FSM $FiF(S) \times FoF(S) = 1$.

Example 4: Fig. 3 illustrates the computation of the parameters. The number of input edges falling on the states S_1 (Fig. 3*a*) and S_2 (Fig. 3*b*) from outside are five and one respectively; $reachability(S_1) = 5$, whereas $reachability(S_2)$ is 1. Out of $2^2 = 4$ input combinations the number of times the FSM exits from state S_1 is equal to 1, which defines the $emitability(S_1) = 1$. The fan-in-factor of S_1 is computed as $FiF(S_1) = reachability(S_1)/emitability(S_1) = 5/1 = 5$. Similarly, $FoF(S_1) = 1/5 = 0.20$. For S_2 the FiF and FoF values are 0.33 and 3, respectively (Fig. 3*b*).

The FiF-FoF analysis of an FSM requires computation of these values for each state of the FSM. In the following subsection we present the algorithms for this purpose.

4.2 Computation of FiF-FoF values

The objective of computing the FiF-FoF values is to identify the states having high accessibility and those having high exitability and then to incorporate certain efficient mechanism to establish a high mobility between them. In the state transition table (STT) of an FSM, the entries of the form (⟨*Input*⟩ ⟨*PresentState*⟩ ⟨*NextState*⟩ ⟨*Output*⟩) represent a state transition of the FSM. To compute the FiF-FoF values, the STT is to be scanned and as each transition is being read, the reachability and the emitability values of the states involved in that transition are updated. The algorithm to compute the exact values FiF-FoF of FSM states is given in Algorithm 1: Exact_FiF-FoF.

Algorithm 1: Exact_FiF-FoF
Input: STT of the FSM with n states $S_1, S_2 \ldots S_n$
Output: $FiF(S_i)$ and $FoF(S_i)$, $\forall i = 1, \ldots, n$
Step 1: Initialise
$reachability(S_i) \leftarrow 0$, $emitability(S_i) \leftarrow 0$
$\forall i = 1, \ldots, n$
Step 2: Do Step 3 *for* successive entries of the *STT*
Step 3: Let the current entry is
⟨*InputPattern*⟩ S_i S_j ⟨*OutputPattern*⟩
If $(S_i \neq S_j)$ *Then Do*
$reachability(S_j) \leftarrow reachability(S_j) + 1$
$emitability(S_i) \leftarrow emitability(S_i) + 1$
Else the transition is a self loop and ignore it
Step 4: Compute
$FiF(S_i) = reachability(S_i)/emitability(S_i)$ and
$FoF(S_i) = emitability(S_i)/reachability(S_i)$
$\forall i = 1, \ldots, n$

Example 5: Table 1 shows the FiF-FoF values of all the states of the FSM shown in Fig. 2*a*.

4.2.1 Analysis of algorithm Exact_FiF-FoF: In an FSM with N primary inputs (PIs) there are 2^N transitions for each state, that is, the STT may consists of $n \times 2^N$ entries, where n is the number of FSM states. Since, to compute reachability and emitability algorithm 1 has to scan each entry of the STT (*step 3*), the worst case complexity of the algorithm Exact_FiF-FoF is $O(n \times 2^N)$. In the STT, there exists a large number of don't cares in ⟨*InputPattern*⟩, and thus the average complexity reduces to $O(n \times 2^{N-d})$, where on average d number of don't cares exist in an entry of the STT. However, for a large value of N, the complexity is still prohibitively high and therefore we propose a heuristic to compute the FiF-FoF values.

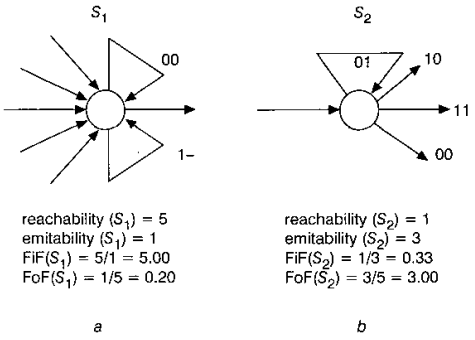


Fig. 3 *Calculation of FiF-FoF values*
a High FiF, low FoF state
b Low Fif, high FoF state

4.2.2 Heuristic for FiF-FoF computation: Since our aim is to identify states with suitable BIST qualities, there is no need to be too precise about the FiF-FoF values. The proposed computation tries to find the approximate FiF-FoF values of an FSM state. Here, instead of scanning the entire *STT*, we apply a sequence of test patterns on the FSM and trace the behaviour as it jumps from one state to another. At each step, the reachability and emitability values of the relevant states are updated. A number of such test pattern sequences are to be applied in succession and the average reachability and emitability values may be computed from them. Finally, the FiF-FoF values are computed from the average reachability and emitability. The algorithmic steps of the heuristic are described in Algorithm 2.

Algorithm 2: Approximate_FiF-FoF
Input: The FSM and the test pattern sequences
Output: Estimated values of $FiF(S_i)$ and $FoF(S_i)$,
$\forall i = 1, \ldots, n$
Step 1: Repeat step 2 *To* step 3 *For* each test pattern sequence
Step 2: Initialise
$reachability(S_i) \leftarrow 0$, $emitability(S_i) \leftarrow 0$,
$\forall i = 1, \ldots, n$
Step 3: Apply the successive test patterns on the FSM. Let the test pattern takes the FSM from state S_i to state S_j
If $(S_i \neq S_j)$ *Then Do*
$reachability(S_j) \leftarrow reachability(S_j) + 1$
$emitability(S_i) \leftarrow emitability(S_i) + 1$
Else the transition is a self loop and ignore it
Step 4: Compute mean values of reachability and emitability for each FSM state
Step 5: Multiply them with a factor r, where $r = (n \times 2^l)/l$, l = no. of PI lines, and l = average length of pattern sequences.
Step 6: $\forall i = 1, \ldots, n$ compute
$FiF(S_i) = mean\_reachability(S_i)/mean\_emitability(S_i)$, and
$FoF(S_i) = mean\_emitability(S_i)/mean\_reachability(S_i)$

Table 1: FiF-FoF values in states of M_1 shown in Fig. 2

State	Reachability	Emitability	FiF	FoF
S_1	2	4	0.500	2.000
S_2	8	3	2.670	0.375
S_3	1	4	0.250	4.000
S_4	4	1	4.000	0.250
S_5	1	4	0.250	4.000

4.2.3 Analysis of algorithm approximate_FiF-FoF: If t_s be the number of test sequences and l is the average length of the pattern sequences, then Algorithm 2 has a complexity of $O(t_s l)$.

Having computed the FiF-FoF values for each FSM state, we get an idea about the worth of the individual states with respect to their BIST quality. The following subsection describes how, on the basis of these values, an efficient state assignment can be performed that enhances the BIST quality of the resultant sequential machine.

5 FiF-FoF-based BIST scheme

We first describe how the unreachable states can be made reachable. The method of making hard-to-reach states easy-to-reach and hard-to-exit states easy-to-exit will be discussed subsequently.

5.1 Handling unreachable states

To establish mobility between unreachable and reachable states the space of reachable state codes is confined within the interval $[0, n-1]$. As a result, the unreachable state codes are shifted to the region $[n, 2^k]$, where k is the number of flip-flops used. The technique is based on the fact that, in this way, for every unreachable state code there exists a reachable state with MSB as 0 [9, 10].

Example 6: Consider the FSM shown in Fig. 2*a*. There are five states in it. If we confine the reachable state codes within the range 000 to 100, the unreachable state codes are {101, 110, 111}. Then the reachable state 001 of the FSM can be mapped to the unreachable state code 101 by inverting the MSB. All the unreachable states have their reachable counterpart that differs only at the MSB. Therefore to input the codes corresponding to the unreachable states to the CC part of the sequential circuits in test mode we propose to add a control point at the MSB of PS lines. The control point is implemented with an XOR gate as shown in Fig. 5. One input to XOR is the MSB of PS lines and the other one is the control primary input (CPI). Since the reachable states are likely to appear on the PS lines, unreachable state codes also can be fed to CC by keeping CPI as 1. In test mode, the CPI is fed from the pattern generator, whereas for normal functioning of the FSM the CPI is fixed to logic 0.

5.2 FiF-FoF-based state encoding

In this section we present a state encoding scheme that takes FiF-FoF values of FSM states into consideration and ensures easy mobility between the set H_{FiF} of states with high FiF (and low FoF) values, and the set H_{FoF} of states with high FoF (and low FiF) values.

The essence of the proposed scheme lies in establishing a pair of artificial transitions between two states one of which has a high *FiF* but low *FoF* values and the other has a low *FiF* but high *FoF* values. The technique is illustrated in Fig. 4. Here S_1 is a high-*FiF* and low-*FoF* state and the state S_2 is the reverse.

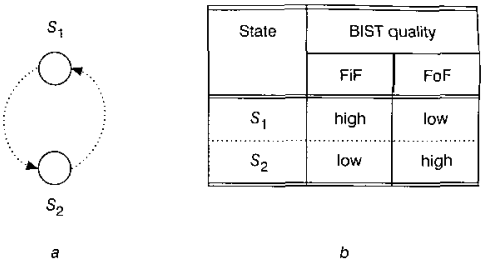


Fig. 4 *Ensuring mobility between high FiF, low FoF and low FiF, high FoF states*

a Artificial transition active only during circuit testing
b BIST qualities of S_1 and S_2

There is no direct transition between S_1 and S_2 in the original FSM. To establish an easy mobility between these states a pairs of transitions (shown in dotted lines in the Figure) are artificially inserted between them. These artificial transitions are active only during testing and not in the normal mode of operation. There is no primary input pattern associated with them because these transitions are solely achieved by directly manipulating the outputs of the flip-flops. The details of the scheme follow.

Let us take a pair of states (S_{hi}, S_{ho}), where $S_{hi} \in H_{FiF}$ and $S_{ho} \in H_{FoF}$. To ensure an easy mobility between S_{hi} and S_{ho} pair, the codes C_{hi} and C_{ho} are assigned to S_{hi} and S_{ho}, respectively, such that C_{hi} and C_{ho} differ only in a single bit (say, the LSB). If an extra control point with XOR is added at the least significant PS line, then the FSM can easily switch from the state S_{hi} to S_{ho} and *vice versa*, which makes the said pair, as a whole, quite easy to reach as well as exit. The resulting architecture is shown in Fig. 5.

If k bits are required to encode the state, then by adding two control points, at MSB and the LSB, we can ensure mobility between 2^{k-2} pairs of states. The complete scheme in algorithmic form is given in Algorithm 3.

Algorithm 3: Encode_with_FiF-FoF
Input: Description of sequential machine as a STT or STG
Output: A state encoding with enhanced BIST quality
Step 1: Find the number of states n and k, $2^{k-1} < n \leq 2^k$
Step 2: Compute FiF-FoF values for all the states $S_1, \ldots, S_n$ with *algorithm 1* or *2*
Step 3: Create two sorted lists of the states $S_1, \ldots, S_n$, one in ascending order of FiF and the other in ascending order of FoF values. Let the sorted lists be $S_1^{hi}, S_2^{hi}, \ldots, S_n^{hi}$ and $S_1^{ho}, S_2^{ho}, \ldots, S_n^{ho}$
Step 4: Make pairs of states (S_i^{hi}, S_i^{ho}) until all states are exhausted. The resulting pairs (S_i^{hi}, S_i^{ho}), are referred to as the BIST-pairs
Step 5: Let CS is the code set containing codes $\{0, 1, \ldots, n-1\}$. Decide the bit position p (say, LSB) where second control point is to be inserted.
Step 6: Take a pair of code C_1 and C_2 from $\{CS\}$, where C_1 and C_2 differ only in the pth bit position. Assign codes C_1 and C_2 to a BIST-pair of states. Repeat the process for all BIST-pairs.

While implementing Algorithm 3 we occasionally found states with almost the same values of reachability and emitability so that the FiF-FoF values do not faithfully

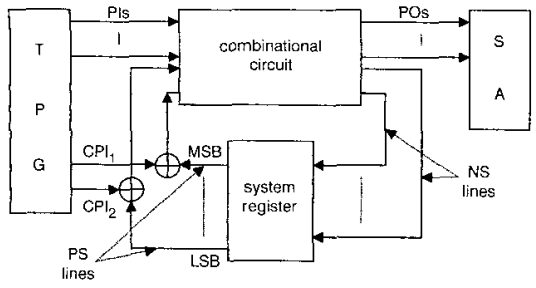


Fig. 5 *BISTed FSM with two CPIs*
CPI_1 control primary input 1
CPI_2 control primary input 2

represent the BIST qualities for them. For example, a state with 100/100 reachability/emitability yields the same FiF-FoF value as a state with 1/1 reachability/emitability, though it is obvious that the former state has higher BIST quality than the later. For such small number of states the value of reachability plus emitability is used instead of FiF-FoF, and state assignment is done on the basis of this quantity.

Example 7: Consider the FSM with five states S_1, S_2, S_3, S_4 and S_5 as shown in Fig. 2*a*. Table 1 shows the computed values of the reachability, emitability, FiF values and FoF values of each individual states. The sorted list of states with descending order of FiF values is $\{S_4, S_2, S_1, \{S_3, S_5\}\}$. The same with descending order of FoF values is $\{\{S_3, S_5\}, S_1, S_2, S_4\}$. States with the same FiF or FoF values are enclosed in braces. Therefore the set P_{BIST} of BIST-pairs is given by $P_{BIST} = \{(S_4, S_3), (S_2, S_5)\}$. The pool of codes $C = \{000, \ldots, 111\}$. The set $\{101, 110, 111\}$ represents the unreachable state codes. If the second control point is inserted at $p =$ LSB of the PS lines, then a state code assignment according to the present scheme is

$$(000) \rightarrow S_4, (001) \rightarrow S_3, (010) \rightarrow S_2, (011) \rightarrow S_5, (100) \rightarrow S_1$$

5.3 Analysis of algorithm encode_with_FiF-FoF

Step 2 of *Algorithm Encode_with_FiF-FoF* has a complexity $O(t_s l)$ where t_s is the number of test sequences and l is the average length of the pattern sequences, when the FiF-FoF values are computed with Algorithm 2. The complexity of steps 3, 4 and 6 are $O(n\log_2 n)$, $O(n)$, and $O(n)$, respectively. Therefore the overall complexity of Algorithm Encode_with_FiF-FoF is $O(t_s l + n\log_2 n)$.

6 Experimental results

The proposed scheme has been carried out in the framework of SIS [13]. Extensive experimentation was done on a large number of MCNC benchmark circuits and some randomly generated large FSMs. Table 2 represents the parameters of the MCNC benchmarks and the randomly generated circuits as shown in Column 1. The number of primary inputs, primary outputs, states in the FSM, flip-flops required and the estimated gate area using the output dominance algorithm in JEDI [13] with MCNC GENLIB are noted in columns 2, 3, 4, 5, and 6, respectively.

The results are shown in Table 3. Column 2 presents the number of test vectors applied to the corresponding circuit.

Table 2: Circuit descriptions

Circuit	PI	PO	States	FFS	Area
ex1	09	19	20	5	345
s27	04	01	06	3	071
s208	11	02	18	5	184
s386	07	07	13	4	199
s420	19	02	18	5	157
s820	18	19	25	5	437
s832	18	19	25	5	450
s1488	08	19	48	6	901
s1494	08	19	48	6	892
bbsse	07	07	16	4	199
styr	09	10	30	5	740
keyb	07	02	19	5	319
opus	05	16	10	4	153
sse	07	07	16	4	197
kirkman	12	06	16	4	262
ex6	05	08	08	03	130
scf	27	56	121	07	1072
tbk	06	03	32	05	299
$RAND_1$	56	58	624	10	3182
$RAND_2$	64	77	819	10	9333
$RAND_3$	82	73	1016	10	12771

An FSM, synthesised from the proposed scheme, has been tested with a fixed number of test vectors generated by the cellular automata (CA) [12] based PRPG that used to test the original circuit. The input to n primary inputs were taken from a $(n+2)$ cell $GF(2)$ CA. The two CPIs have been fed from two separate cells of such a CA. The single stuck-at fault coverage of the original circuit (synthesised from SIS [13]) has been shown in columns 3 and 4. Column 5 presents the fault coverage of the circuits synthesised through the proposed scheme with two/one CPI(s). Different seeds have been applied for different schemes though the number of test vectors remained the same for a given circuit. While simulating the fault coverage, the flip-flops were kept uninitialised. The best result obtained out of three seeds has been considered. To encode FSM states the FiF-FoF values has been computed using the heuristic proposed in Section 4.2.2. Column 6 and 7 of Table 3 depict the area of the circuit synthesised through JEDI and from proposed state encoding scheme with two/one control points (XORs).

The results shown in Tables 2 and 3 are obtained through a set of similar logic optimisation steps (available in SIS) for all the test cases. For fault simulation the synthesised FSM is first converted to *blif* format and then to bench using public domain tool *blif2bench*. All circuits were subjected to test after prior state minimisation with stamina of SIS.

The test results, shown in Table 3, express the performance of the proposed scheme. It is observed that the proposed scheme enhances the detectability of faults by a significant amount for all the circuits. The design targets only fault efficiency and not the gate area. The results establish that the proposed method can enhance the BIST quality of the synthesised circuits with little area overhead.

For all the cases the simulation is done in the platform of fault simulator *hope* [14], and the fault coverage figures are expressed in terms of

$$fault\ coverage = \frac{total\_number\ of\ detected\ faults}{total\ number\ of\ faults\ in\ CUT}$$

Table 3: Experimental results

Circuit name	Test vectors	Fault cov (%) with PRPG JEDI *n* cell TPG	Fault cov (%) with PRPG JEDI $(n+2/1)$ cell TPG	Proposed scheme	Area JEDI (FSM + 2/1 XOR)	Area Proposed Scheme
ex1(†)	1000	68.22	76.59	91.62	345	354.0
s27	200	51.49	66.34	91.74	71	72.0
s208	1200	40.53	56.83	82.98	184	129.0
s386 (†)	500	47.15	78.46	87.88	199	213.0
s420	1000	50.51	61.62	76.83	157	147.0
s820	8000	37.88	42.79	88.34	437	503.0
s832	8000	41.88	46.05	90.28	450	442.0
s1488	1200	41.27	50.74	96.31	901	915.0
s1494	1200	42.89	51.54	95.09	892	901.0
bbsse	500	66.15	73.54	91.74	199	218.0
styr(†)	2000	42.78	64.51	85.43	740	836.0
keyb(†)	1000	20.44	22.85	70.29	319	324.0
opus	700	57.78	63.89	85.86	153	156.0
sse (*)	1500	65.88	73.72	93.86	197	211.0
kirkman (*)	7000	78.75	82.32	88.83	262	270.5
ex6 (*)	400	64.82	83.92	93.59	130	148.5
scf	10000	38.84	39.00	93.74	1072	1113.0
tbk (*)	1300	08.49	17.88	20.38	299	316.5
$RAND_1$	20000	82.36	87.45	89.91	3182	3674.0
$RAND_2$	25000	54.49	56.56	63.39	9333	11280.0
$RAND_3$	35000	48.81	53.77	60.78	12771	14556.0

* One XOR used since no unreachable state exists for these circuits
† Flip-flops are initialised to all 0s

7 Conclusions and scope of further research

An efficient synthesis scheme for sequential machines targeting enhanced BIST quality of the resultant circuits has been presented. We introduced a novel BIST-characterising metric termed *fan-in-factor* and *fan-out-factor* in FSM-states. The scheme is based on the FiF-FoF analysis of a given FSM. The design permits the improvement of parameter values for a set of selected state pairs and thus enhances the detectability of the circuit faults in a BIST environment. Experimental results, performed on MCNC benchmark circuits, establish the claim of enhanced BIST quality of the proposed design. Though the scheme targets testable design of sequential circuits, the resulting area penalties are insignificant. Further work can be done by incorporating area overhead and delay of the resulting circuit during the design phase.

8 Acknowledgment

The authors wish to thank the anonymous referee for the constructive comments that helped to improve the paper. This research work is supported by the AICTE sponsored project 8020/RID/R&D-142/01-02.

9 References

1 AGRAWAL, V.D., KIME, C.R., and SALUJA, K.K.: 'A tutorial on built-in-self-test part 1: Principles', *IEEE Des. Test Comput.*, 1993, pp. 73–82
2 AGRAWAL, V.D., KIME, C.R., and SALUJA, K.K.: 'A tutorial on built-in-self-test part 2: Applications', *IEEE Des. Test Comput.*, 1993, pp. 69–77
3 WUNDERLICH, H.: 'The design of random testable sequential circuits', Proc. 19th Symp. on Fault-tolerant computing, Chicago, USA, 1989, pp. 110–117
4 MURADALI, F., NISHIDA, T., and SHIMIZU, T.: 'A structure and technique for pseudorandom-based testing of sequential circuits', *J. Electron. Test. Theory Appl.*, 1995, pp. 107–115
5 NACHMAN, L., SALUJA, K.K., UPADHAYA, S., and REUSE, R.: 'Random pattern testing for sequential circuits revisited', Proc. 26th Symp. Fault-tolerant computing, Sandai, Japan, June 1996, pp. 44–52
6 POMERANZ, I., and REDDY, S.M.: 'Built-in-test generation for synchronous sequential circuits', Proc. Int. Conf. on Computer-aided design, San Jose, USA, November 1997, pp. 421–426
7 POMERANZ, I., and REDDY, S.M.: 'Improved built-in test pattern generators based on comparison units for synchronous sequential circuits', Proc. Int. Conf. on Computer-aided design, San Jose, USA, November 1998, pp. 26–31
8 WUNDERLICH, H., and KIEFER, G.: 'BIT-flipping BIST', Proc. Int. Conf. on Computer-aided design, San Jose, USA, November 1996, pp. 337–343
9 SIKDAR, B.K., ROY, S., and DAS, D.K.: 'Enhancing BIST quality of sequential machines through degree-of-freedom analysis', Proc. of 10th Asian Test Symposium, Kyoto, Japan, November 2001, pp. 285–290
10 ROY, S., SIKDAR, B.K., MUKHERJEE, M., and DAS, D.K.: 'Degree-of-freedom analysis for sequential machines targeting BIST quality and gate area', Proc. of the 7th Asia and South Pacific conference on Design automation and 15th Int. Conf. on VLSI Design (ASP-DAC/VLSID-2002), Bangalore, India, January 2002, pp. 671–676
11 STORELE, A.P., and WUNDERLICH, H.: 'Hardware-optimal test register insertion', *IEEE Trans. Comput.-Aided Des.*, 1998, **17**, (6), pp. 531–539
12 CHAUDHURI, P.P., CHOWDHURY, D.R., NANDI, S., and CHATTERJEE, S.: 'Additive cellular automata – Theory and application vol. 1' (IEEE Computer Society Press, CA, USA, 1997)
13 SANTOVICH, E.M., SINGH, K.J., LAVAGNO, L., MOON, C., SALADANHA, A., SAVOJ, H., STEPHEN, P.R., MURGAI, R., BRAYTON, R., and SANGIOVANNI-VINCENTELLI, A.L.: 'SIS: A system for sequential circuit synthesis', Tech. Rep. UCB/ERL M92/41, Electronic Research Laboratory, 1992
14 LEE, H.K., and HA, D.S.: 'HOPE: An efficient parallel fault simulator for synchronous sequential circuits', *IEEE Trans. Comput. Aided Des. Integr. Circuits and Syst.*, 1996, **15**, (9), pp. 1048–1058