SYNTHESIS OF I_{DDQ}-TESTABLE CIRCUITS: INTEGRATING BUILT-IN CURRENT SENSORS

H.-J. Wunderlich[1], M. Herzog[1]
J. Figueras[2], J.A. Carrasco[2], and A. Calderón[2]

[(1)]Institute of Computer Structures, University of Siegen,
Hoelderlinstr. 3, 57068 Siegen, Germany

[(2)]Department of Electronics Engineering, Universitat Politecnica de Catalunya,
Diagonal 647, plta. 9, 08028 Barcelona, Spain

Abstract

"On-Chip" I_{DDQ} testing by the incorporation of Built-In Current (BIC) sensors has some advantages over "off-chip" techniques. However, the integration of sensors poses analog design problems which are hard to be solved by a digital designer. The automatic incorporation of the sensors using parameterized BIC cells could be a promising alternative. The work reported here identifies partitioning criteria to guide the synthesis of I_{DDQ}-testable circuits. The circuit must be partitioned, such that the defective I_{DDQ} is observable, and the power supply voltage perturbation is within specified limits. In addition to these constraints, also cost criteria are considered: circuit extra delay, area overhead of the BIC sensors, connectivity costs of the test circuitry, and the test application time. The parameters are estimated based on logical as well as electrical level information of the target cell library to be used in the technology mapping phase of the synthesis process. The resulting cost function is optimized by an evolution-based algorithm. When run over large benchmark circuits our method gives significantly superior results to those obtained using simpler and less comprehensive partitioning methods.

1 Introduction

The test methodology based on the observation of the quiescent current (I_{DDQ}) complements logic (voltage) testing in CMOS technologies. The quiescent current consumed by the IC is a good indicator of the presence of a large class of defects escaping logic test [1-6]. On-chip Built-In Current (BIC) sensors have been proposed to overcome some of the problems encountered in off-chip I_{DDQ} testing: long testing times and low discrimination of small defective currents. These problems can be solved by partitioning the Circuit Under Test (CUT) in subcircuits, each provided with a BIC sensor. In recent years, different BIC sensors have been proposed [7-11]. Some BIC sensors (i.e. pn junctions or bipolar devices) introduce a voltage drop during transient switching which can be unacceptable in some applications due to its effects in delay and noise margin reduction. For these applications the BIC sensors have to incorporate a bypass element so that the perturbation in the virtual ground is below a certain maximum.

The class of BIC sensors considered in this paper is illustrated in figure 1. The BIC sensor includes a sensing device, a bypass MOS switch and a detection circuitry. A control signal C is applied to the gate of the bypass MOS device. During normal operation, $C=1$, turns the MOS on. During testing, first C is set to 1 and a test pattern is applied to the CUT. When the transient i_{DD} current has decayed, C is set to 0, turning the MOS off, and the sensing device produces a voltage signal which is processed by the detection circuitry to produce a PASS/FAIL signal, depending on whether the sensed I_{DDQ} falls below/above a given threshold value $I_{DDQ,th}$. Several sensing devices can be used [7-12], each with its advantages and disadvantages. This kind of testability enhancement may cause extra delay and area overheads as well as a reduction of the noise immunity margins [8,9,12,13,15], and the objective of a sophisticated BIC sensor placement should be minimizing these drawbacks.

I_{DDQ}-test of large CUTs cannot be done effectively using a single BIC sensor. One obvious reason is the need for an appropriate discriminability. Effective test of defects in CMOS typically requires $I_{DDQ,th} = 1\mu A$ and non defective I_{DDQ} currents of large circuits can be larger than 1 μA. Also, the large parasitic capacitance introduced at the sensing node by the CUT and the sensing device adversely impacts test times. These problems can be alleviated by partitioning the CUT into

This work has been performed within the framework of the ESPRIT Project 7107 Archimedes

groups of gates and introducing a BIC sensor for each group [12,13].

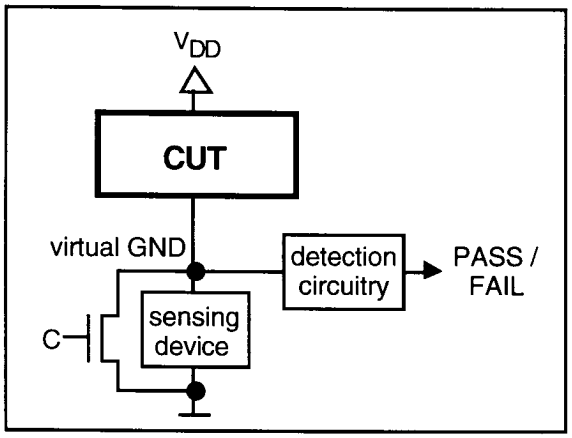

Figure 1: Architecture of a BIC sensor with a bypass device.

Fine-grain partitions yield high discriminability and low test times but incur in high area overhead due to the (replicated) detection circuitry, extra routing caused by the introduction of the virtual rail and routing among BIC sensors. Coarse-grain partitions have smaller area overheads but give smaller discriminabilities and longer test times. In addition, the „shape" of the groups can have a great influence on the required BIC sensor area. This is illustrated in figure 2, which shows two different partitions for a CUT with a two-dimensional array structure involving three cell types. Partition 1 has an average maximum i_{DD} in each group smaller than partition 2 as the three cells C1, C2, C3 will not switch in parallel; thus, using partition 2, the switching devices have to be greater to guarantee the same limits of the virtual rail perturbation, and partition 1 should be preferred.

Finding good partitions for I_{DDQ} testability is a complex problem where discriminability between faulty and fault free current, area overhead, delay degradation and test application time have to be considered. In this paper we propose an evolutive optimization method for minimizing a cost function subject to restrictions. The constraints are discriminability and virtual rail perturbation. The cost function is obtained by weighting estimators of the different costs involved in the trade-off. These estimators make an appropriate trade-off between accuracy and computation complexity and are evaluated using parameterized electrical level information of the target technology. The rest of the paper is organized as follows. Section 2 states formally the partitioning problem and its constraints. Section 3 describes the estimators used in the cost function. Section 4 surveys the evolution optimization algorithm. The experimental results discussed in section 5 show the superiority of this approach over a straightforward manual BIC sensor placement.

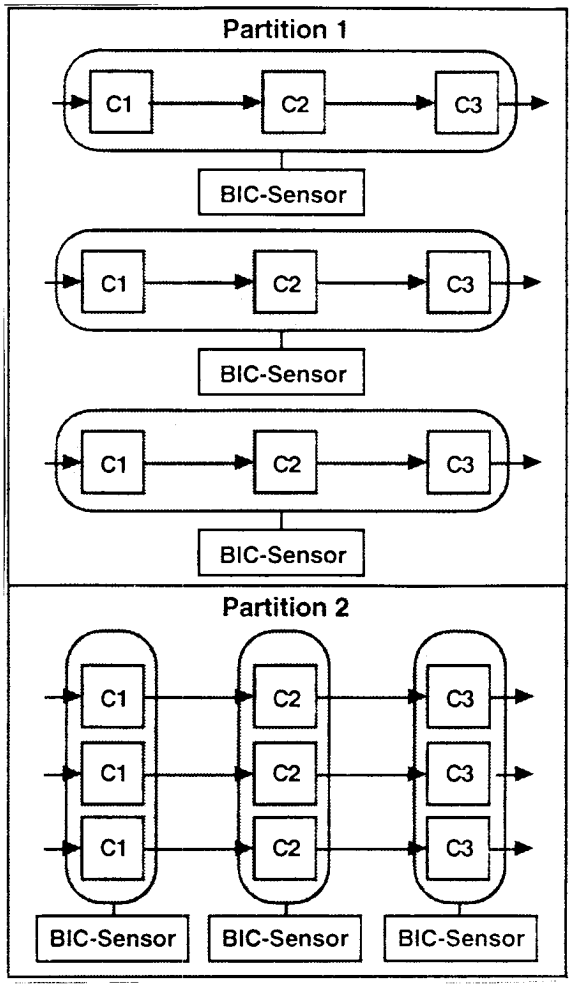

Figure 2: Two partitions illustrating the impact of the group shape in BIC sensor area.

2. Problem statement and constraints.

To set-up formally the partitioning problem, the CUT is modelled by a directed graph $C=(G,T)$, where G is the set of gates and T includes the connections among the gates. A partition Π of G is a collection $\{M_1,\ldots,M_k\}$ of disjoint groups of gates (modules) covering G ($\cup_{i=1}^{K} M_i = G$). Each gate is completely included in one group, hence no transistor group is split among groups, avoiding potential latchup problems [11,14]. Let $I_{DDQ,th}$ be the minimum defective I_{DDQ} current which has to be detected and let $I_{DDQ,nd,i}$ be the maximum non-defective current of module M_i The discriminability for M_i is defined as

$$d_i(M_i) = \frac{I_{DDQ,th}}{I_{DDQ,nd,i}} \geq d.$$

For the feasibility of an I_{DDQ} test, $d>1$ is required,

and a typical value is 10. This restriction can be expressed by a *constraint evaluation function* defined on the set of all possible partitions P as $r: P \rightarrow (0,1)$ with

$$r(\Pi) = 1 \Leftrightarrow \bigwedge_{i=1}^{K} d(M_i) \geq d$$

The other targets concerning speed and area can be described by a global cost function C on each partition. It is defined as $C: P \rightarrow R$ in terms of the relative weights α_i of the metrics c_i:

$$C(\Pi) = \sum_{i=1}^{n} \alpha_i * c_i(\Pi)$$

The parameters defined above allow establishing the global cost function for optimization in the design space Speed-Area-Testability according to different priorities reflected on the values of the weight factors α_i. Constraints and costs lead to a partitioning problem as follows:

Partitioning Problem PART-I_{DDQ}: Find a partition Π^* satisfying the constraints $r(\Pi^*)$ and at the same time minimizing the global cost function $C(\Pi^*)$.

In general, this problem is NP-hard, and heuristics to find acceptable solutions are proposed. In addition, the precise evaluation of the constraints and the cost functions are electrical level problems which are not solvable in acceptable time except for very small circuits. In the next section, approximate estimating procedures are presented using the gate level description of the CUT and the target cell library.

3 Maximum current and cost estimators

In this section a set of estimators for the variables necessary to evaluate the constraints and costs at logic level are proposed. A target cell library fully characterized at electrical level is assumed available.

3.1 Area overhead due to BIC sensing.

A perturbation of the Virtual Ground (Virtual V_{DD}) is caused by the incorporation of the I_{SSQ} (I_{DDQ}) BIC sensors. This variation of the power supply voltage causes a reduction of the noise immunity margins. In addition, circuits with memory elements may loose the memorized information. The worst perturbation occurs during switching when the current is maximal. A constraint put on this voltage dropping is usual for choosing the size of the BIC sensors. The voltage dropping depends on the maximum transient current $I_{DD,\max,j}$ through module M_j and on the sensor area.

For each module M_j the maximum transient current $I_{DD,\max,j}$ is estimated by the maximum number of gates or modules switching simultaneously. In order to estimate a value at logic level we assume that all gates located at the same depth on different paths, switch in a way that their maximum currents add. This is a pessimistic assumption as we do not consider paths possibly blocked. This simplifies the problem, and only the possible paths of all transitions and the maximum simultaneity of transitions are identified. For each gate g_i all possible L_i transition paths and the times of transition arrival are determined. In this way a set of integers $t_1^i, t_2^i, \ldots, t_{P_K}^i, \ldots, t_{P_{L_i}}^i$ is computed indicating the possible times of transitions at the gate g_i for each path P_k, where $k := 1, \ldots, L_i$. An upper limit of $\max\{i_{DD}\}$ of a group M_j of gates is computed by

$$I_{DD,\max,j} = \max_{n} \left\{ \sum_{\substack{g_i \in M_j \\ \exists k, t_{P_k}^i = t_n}} i_{DD}^*(g_i) \right\}$$

The estimate for $I_{DD,\max,j}$ is approximate and pessimistic, but is computationally efficient enough to allow exploration of a large number of partitions in reasonable amounts of CPU time.

Let $R_{S,i}$ be the ON resistance of the BIC sensor of M_i, the maximum virtual rail perturbation can be approximated by $R_{S,i} * I_{DD,\max,i}$. The maximum virtual rail perturbation of each module is limited to a given predefined value r^*, yielding

$$R_{S,i} * I_{DD,\max,i} \leq r^* \quad for\ 1 \leq i \leq K.$$

Since the requirements for r^* are typically very stringent (between $100mV$ and $300mV$) the impact of the feasible $R_{S,i}$ on the delay of the CUT tends to be small. Then, to simplify the optimization problem we take:

$$R_{S,i} = \frac{r^*}{I_{DD,\max,i}}, \quad 1 \leq i \leq K.$$

The sensor area cost estimator is computed using an area model of the form $A_0 + A_1 / R_S$ for each BIC sensor, where the term A_0 accounts for the detection circuitry and the term A_1 / R_S accounts for the sensing element and the bypass device. This gives

$$A = KA_0 + \sum_{i=1}^{K} \frac{A_1}{R_{S,i}}.$$

As all components of the objective function should have similar range and variation for optimization reasons we actually compute

$$c_1(\Pi) := \log(A)$$

3.2 Delay overhead due to BIC sensors.

The delay overhead cost is computed as:

$$c_2(\Pi) = \frac{D_{BIC} - D}{D},$$

where D is the delay of the circuit without BIC sensors and D_{BIC} is the delay of the circuit with BIC sensors. Both delays are computed using a longest path algorithm. The gate delays $D_{BIC}(g,t)$ (these delays are time grid functions) used for D_{BIC} are obtained from the nominal gate delays (without BIC sensor) $D(g)$ and degradation factors $\delta(g,t)$ as

$$D_{BIC}(g,t) = \delta(g,t)D(g).$$

The gate delay degradation factor $\delta(g,t)$ is obtained using a second order electrical network model having as parameters R_S (the BIC sensor ON resistance), C_S (the parasitic capacitance at the virtual rail node), C_g (the equivalent capacitance at the output of g), R_g (an average equivalent ON resistance for the discharging network of a gate of the CUT), and $n(t)$ (the activity-number of simultaneously switching gates at time t). With these assumptions the expression for the gate delay degradation becomes

$$\delta(g,t) = 2\left(\gamma(t) - \sqrt{\gamma(t)^2 - 4\gamma_S}\right)^{-1},$$

with $\gamma(t) = \gamma_S + \gamma_{gS}(t) + 1$, $\gamma_S = (R_g C_g)/(R_S C_S)$ and $\gamma_{gS}(t) = n(t)C_g / C_S$.

3.3 Interconnection costs.

The interconnection costs are divided into costs for linking the BIC sensors to the gates within each module, and the costs for connecting the BIC sensors of all modules by the test clock and test output. The module interconnection costs take into account the difficulty in sensing gates placed in remote locations. The separation parameter $S(g_i,g_j)$ of two gates g_i and g_j is the minimum number of nodes traversed when going from g_i to g_j in the *undirected graph* of the logic circuit. If $S(g_i,g_j)$ exceeds a certain parameter say p, or if no path exists between g_i and g_j, we set $S(g_i,g_j) := p$.

The separation parameter $S(M)$ of a module M is defined as the sum of the separation parameters of all gate pairs:

$$S(M) := \sum_{g_i,g_j \in M} S(g_i,g_j).$$

Intuitively, the parameter decreases if many nodes of $S(M)$ are connected, and it is minimum if M is a clique of the undirected circuit graph.

The overall interconnection cost of a partition Π of K groups of gates M_k where $k = 1,\ldots,K$ can be estimated by:

$$S(\Pi) = \sum_{k=1}^{K} S(M_k).$$

As a cost function we use

$$c_3(\Pi) := \log(S(\Pi)).$$

The calculation time for this metric grows quadratically with the number of gates of each partition group M_k. Since, in practice, the number of gates of a partition group is relatively small the problem is not too severe.

As also the sensors must be interconnected by the test clock line and the test signal line we use

$$c_4(\Pi) := K.$$

3.4 Test application costs.

The test application time of a precomputed test vector set of the global CUT is estimated. Since the proposed partitioning approach does not modify the logic structure, the test vector set needed to achieve a certain quality goal, does not change. However, the time required for each vector may be significantly different due to different I_{DDQ} *settling times* for different partitions.

For the test application time we use

$$c_5(\Pi) := \frac{D^*_{BIC}}{D},$$

were D^*_{BIC} is the sum of D_{BIC} and a term $\Delta(\tau_i)$ accounting for the i_{DD} decay time and the sensing time, estimated from SPICE level simulations as a function of the BIC sensor time constant $\tau_{S,i} = R_{S,i}C_{S,i}$ ($C_{S,i}$ is the parasitic capacitance at the virtual rail of group M_i).

4 Partitioning Algorithm

Looking for a partition that satisfies condition $r(\Pi)$ and minimizes the cost function is a very complex, NP-hard optimizing problem, and it is impossible to evaluate the cost function $C(\Pi)$ for all the partitions. Also evaluating $C(\Pi)$ for a single partition is rather complex as it varies between $O(n)$ and $O(m_i^2)$ where m_i is the size of a module.

Hence a heuristic algorithm is needed which evaluates just a moderate number of partitions but is not caught in a local minimum. A variety of algorithms has been proposed for such kind of problems (force-driven, simulated annealing, Monte Carlo, genetic, e. g.), in this paper an evolution-based algorithm is applied.

4.1 Solving combinatorial problems by evolution based algorithms

For controlling the optimization steps an evolution-based algorithm is adapted [17,18,19]. A single cycle of such an algorithm consists of three steps repeated sufficiently often: recombination, mutation, and selection.

Recombination: A population consists of a number of individuals called parents which correspond to partitions in our case. These parents produce descendants by recombining their attributes. In our case it turned out that this step should be simplified such that just one parent is sufficient for a child, and recombination is just duplication.

Mutation: The child partitions are modified randomly corresponding to certain rules. These rules ensure that the modified partition is still in the neighbourhood of the original one.

Selection: The individuals produced so far are evaluated. The best of them are the parents in the next cycle.

The convergence of this procedure depends on the start population, and on the set of control parameters used.

4.2 Adaptation to PART-I_{DDQ}

The start partitions are determined by simplifying the cost function such that just $c_1(\Pi)$ (area overhead) and $c_2(\Pi)$ (delay overhead) are considered. First the appropriate module size is estimated. This can be done by evaluating $c_1(\Pi)$ and $c_2(\Pi)$ by average numbers for the required parameters and by abstraction from structural information. Then gates are clustered to modules as follows: starting from a gate close to a primary input gate, chains are formed towards a primary output. The process stops if this path reaches a primary output, or if there is no free gate anymore, or if the maximum module size is reached. Modules are formed as long as there are free gates. Using different chains the required number of start partitions is constructed.

The evolution cycle is controlled by a set of parameters as proposed in [18,19]:

μ:	Number of parents
λ:	Number of children per parent
χ:	Number of Monte Carlo descendants
ω:	Maximum lifetime
m:	Maximum number of gates to be moved
ε:	Variation of m

The evolution cycle starts with μ different start partitions $\Pi_1,\ldots,\Pi_\mu$. Children are generated by copying each of the parent individuals λ times, and by mutation afterwards.

The mutation scheme for each of the $(\mu * \lambda)$ descendants is as follows: A module M^i_{start} from the partition $\Pi^i := \{M^i_1,\ldots,M^i_{k_i}\}$ is selected, and the number $m^i_{boundary}$ of its boundary gates is determined. A gate of M^i_{start} is a boundary gate if it is directly connected to a gate outside M^i_{start}. As a uniformly distributed random variable we select the actual number $m_{move} \in \{1,\ldots,\min\{m, m^i_{boundary}\}\}$ of gates to be moved. The m_{move} gates are chosen randomly, and put into the target module they are connected with. If they are connected with several target modules a random one is chosen.

In addition to these $(\mu * \lambda)$ mutated children, $(\mu * \chi)$ *Monte-Carlo* children are generated. Again, each parent individual is copied χ times. A random number of gates of a random module M_{start} is moved into a random module M_{target}. The random variation of these descendants is higher compared with mutations, and they reduce the probability of being caught in a local minimum. If all gates of M_{start} are moved, this module is deleted. After gate moving, costs are recomputed just for the modified modules, and the global costs of the partition are updated. As not the entire cost function has to be recomputed, the partitions generated this way can be evaluated very efficiently.

After exchanging gates, the step width of mutation m is recomputed for each descendant. The new m is subject to normal distribution with variance ε around the m of step before. This scheme results in $(1+\lambda+\chi)*\mu$ partitions. During selection, all parent individuals older than ω generations are deleted. Out of the remaining ones, the μ partitions with best cost functions are selected as parents for the next cycle.

4.3 Example

The partitioning steps are illustrated using the ISCAS85 benchmark circuit C17 as an example [16]. First start partitions $\Pi_1,\ldots,\Pi_\mu$ have to be constructed as described above, figure 4 shows partition Π_1. The recombination step creates partition Π^2_1 by duplicating Π_1. During mutation $M_{start} := (4,6)$ is selected randomly, and the boundary gates $S_{bound} := \{g_4, g_6\}$ are determined. Randomly $m_{move} = 1$ and g_4 as gates to be moved are chosen. With $M_{target} = (2,3)$ the descendant is shown in figure 4. During the next generation the algorithm creates partition Π^3_1, $M_{start} := (2,3,4)$ is selected and the boundary set becomes $S_{bound} = \{g_2, g_3, g_4\}$. Then gate g_3 moves to module (6) (figure 5). For the last step the algorithm selects module (3,6) of Π^3_1. Now both gates g_3, g_4 move to their specific destination-modules and module (3,6) becomes empty. After these 3 generations the partition Π^4_1 consists of two modules (1,3,5), (2,4,6) and is the optimum partition for C17.

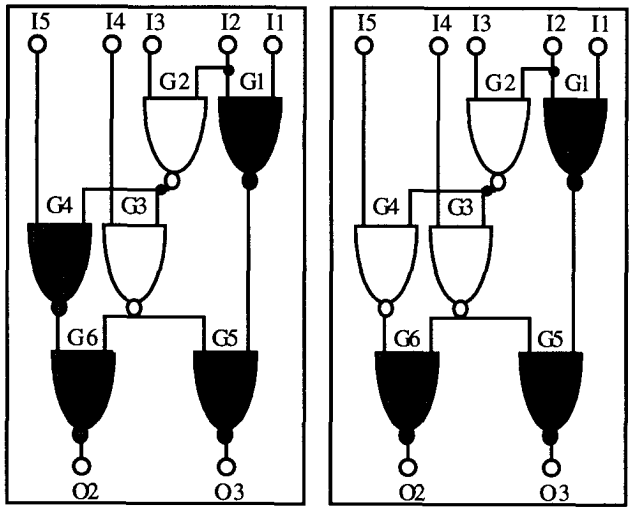

Figure 4: $\Pi_1 = \{(1,5)(2,3)(4,6)\}$, $\Pi_1^2 = \{(1,5)(2,3,4)(6)\}$

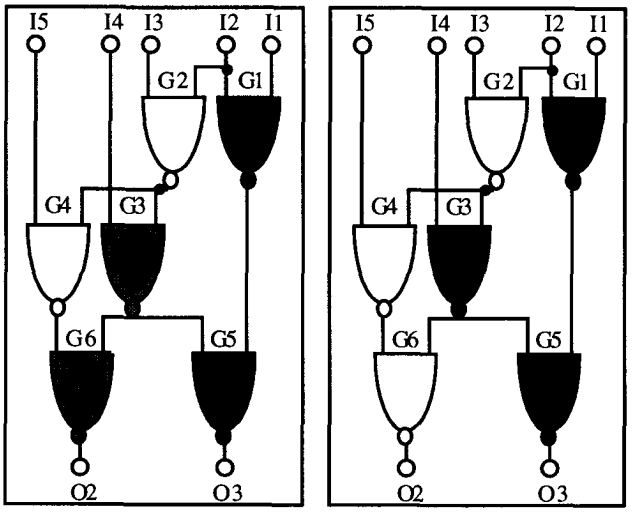

Figure 5: $\Pi_1^3 = \{(1,5)(2,4)(3,6)\}$, $\Pi_1^4 = \{(1,3,5)(2,4,6)\}$

5 Experimental Results

In this section we present a straightforward method for standard partitioning, and we compare the results of the methods discussed so far.

The process of standard partitioning starts with a gate as near to a primary input as possible. New gates are added until a specified size of the module is generated, the module size can be determined electrically as described in section 3, in our case we take the numbers obtained by the evolution based algorithm. The new gate added is that gate whose path length to all the gates already clustered gives a minimum sum. If there are multiple choices, a gate of this set is selected such that the path lengths to all the gates not yet clustered give a maximum sum. A partition generated this way contains modules such that their gates are connected most closely.

5.1 Partitioning the ISCAS85 Circuits

As all components of the cost function should have similar range and variation for optimization reasons and in order to obtain I_{DDQ}-testable circuits with minimal area-overhead which still satisfy performance requirements, the following weight factors were chosen:

$$C(\Pi) = 9 * c_1(\Pi) + 10^5 * c_2(\Pi) + c_3(\Pi) + c_4(\Pi) + 10 * c_5(\Pi)$$

The evolution-based algorithm was applied to the ISCAS85 circuits using this cost function until the results converged to a stable value. Computing time depends on the start population, and is not deterministic. But even for the largest circuit convergence was obtained within a few hours on a Sun Sparc workstation. The results are listed in table 1.

A standard partitioning needs from 14.5% to 30.6% more hardware for BIC sensors than the optimal partitioning of the evolution based algorithm, but does not show any improvement in system performance and test

circuit		C1908	C2670	C3540	C5315	C6288	C7522
#modules		2	3	4	6	5	6
area of BIC sensors	standard	1.08E+6	5.67E+5	2.79E+6	2.87E+6	9.19E+5	5.65E+6
	evolution	8.27E+5	4.95E+5	2.27E+6	2.29E+6	7.30E+5	4.72E+6
sensor area overhead for standard partitioning		30.6%	14.5%	22.9%	25.3%	25.9%	19.7%
delay	standard	1.89E-5	6.97E-5	1.79E-5	2.37E-5	1.79E-4	1.11E-5
	evolution	1.86E-5	6.91E-5	1.80E-5	1.92E-5	8.84E-5	1.08E-5
test application time	standard	0.140	5.95E-2	0.102	5.50E-2	1.96E-2	6.44E-2
	evolution	0.141	5.94E-2	0.100	5.46E-2	1.94E-2	6.43E-2

Table 1: Results of standard partitioning and evolution-based partitioning.

performance. Delays and application times are given as percentage how the incorporation of BIC sensors slows down performance. The area of the BIC sensors is given in units whose actual size depends on technology. As technology mapping is not carried out so far wiring is not considered. Since both in case of the standard and the evolution-based approach the number of modules is the same, the actual routing costs are not expected to differ significantly.

6. Conclusions

Partitioning criteria are established for guiding the automated design of I_{DDQ}-testable circuits. They are expressed by a multi-target objective function to be optimized by an evolution-based algorithm. Circuit extra delay, area overhead of the BIC sensors, connectivity costs of the test circuitry, and the test application time are considered as cost criteria. The parameters are estimated based on logical as well as electrical level information of the target cell library to be used in the technology mapping phase of the synthesis process. For the benchmark circuits investigated, the final design is superior to results of a standard partitioning. So far only resynthesis for including BIC sensors has been considered. Next step is controlling the logic synthesis procedure such that the presented cost function is considered at the early beginning.

7. Glossary

BIC:	built-in current sensor
CUT, *C*:	circuit under test
n:	number of gates of CUT
gi:	gate number i
M:	group of gates (module)
$\Pi := \{G_1, \ldots, G_K\}$:	partition, disjoint group of gates
d:	discriminability of faulty and fault-free case
$r: P \rightarrow (0,1)$:	constraints
α_i:	weight factor
c_i:	cost function
C:	global cost function
r^*:	maximum virtual ground perturbation
L_i:	transition path
t_j^L:	transition time for path j
A_{S_j}	area for BIC sensor i
δ_{BIC}:	delay with connected BIC sensor
δ:	delay without BIC sensor
S_{lm}:	forced separation parameter
μ:	number of parents
λ:	number of children per parent
χ:	number of Monte Carlo descendants
ω:	maximum lifetime
m:	maximum number of gates to be moved
ε:	variation of m

References

[1] C. F. Hawkings and J. M. Soden,. „Electrical characteristics and testing considerations of Gate Oxide Short in CMOS ICs," in *Proc. IEEE International Test Conference,* pp. 544-555, Washington, November 1985

[2] W. Maly, P. K. Nag and P. Nigh, „Testing oriented analysis of CMOS ICs with opens," in *Proc. ACM/IEEE International Conference on Computer Aided Design,* pp. 344-347, Santa Clara, November 1988

[3] R. Rodríguez-Montañes, J.A. Segura, V.H. Champac, J. Figueras and J.A. Rubio, „Current vs. logic testing of gate oxid short, floating gate and bridging failures in CMOS," in *Proc. International Test Conference,* pp. 510-519, Nashville, October 1991

[4] R.C. Aitken, „A Comparison of Defect Models for Fault Location with I_{DDQ} Measurements," in *Proc. IEEE International Test Conference,* pp. 778-787, Baltimore, September 1992

[5] R.C. Aitken, I Chiang, V. Johansen, and P.C. Maxwell „The Effectiveness of I_{DDQ}, Functional and Scan Test: How Many Fault Coverages Do We Need?," in *Proc. IEEE International Test Conference,* pp. 168-177, Baltimore, September 1992

[6] O. Stern and H.-J. Wunderlich, „Simulation Results of an Efficient Defect Analysis Procedure," in *Proc. IEEE International Test Conference,* Washington 1994

[7] W. Maly and P. Nigh, „Build-In Current Testing - A Feasibility Study," in *Proc. IEEE International Conference on Computer Aided Design,* pp. 340-343, Santa Clara, 1988

[8] A Rubio, J. Figueras and J. Segura, „Quiescent current sensor circuits in digital VLSI CMOS testing," *Electronics Letters,* vol. 26, no. 15, pp. 1204-1206, 1990

[9] R. Rius and J. Figueras, „Proportional BIC Sensor for Current Testing," *Journal of Electronic Testing, Theory and Applications (JETTA)*, vol. 3, no. 4, pp. 387-396, December 1992

[10] W. Maly and P. Nigh, „Build-In Current Testing - A Feasibility Study," in *Proc. IEEE International Conference on Computer Aided Design,* pp. 340-343, Santa Clara, 1988

[11] L.R. Carley, D.B.I. Feltham, W. Maly and P.J. Nigh, „Current Sensing for Built-In Testing of CMOS Circuits," in *Proc. IEEE International Conference on Computer Design,* pp. 454-457, New York, October 1988

[12] W. Maly and M. Patyra, „Design of ICs Applying Built-In Current Testing," *Journal of Electronic Testing, Theory and Applications (JETTA),* vol. 3, no. 4, pp. 397-406, December 1992

[13] Y.K. Malaiya, A.P. Jayasumana, C.Q. Tong and S.M. Menon, „Resolution enhancement in I_{DDQ} testing for

large ICs," Technical report CS-92-143, Dept. CS and EE, Colorado State University, Fort Collins, CO 80523, 1988

[14] K-J. Lee and A. Breuer, „Design and Test Rules for CMOS Circuits to facilitate I_{DDQ} Testing of Bridging Faults," *IEEE Transaction on Computer-Aided Design,* vol. 11, no. 5, pp. 659 - 670, May 1992

[15] Y.K. Malaiya, A.P. Jayasumana and R. Rajsuman: A detailed examination of bridging faults. *Proceedings IEEE International Conference on Computer-Aided Design ICCAD*, pages 78-81, Santa Clara, November 1986.

[16] F. Brglez et al: Accelerated ATPG and fault grading via testability analysis. *Proceedings IEEE International Symposium on Circuits and Systems*, Kyoto 1985

[17] Y. Saab and V. Rao: An Evolution-Based Approach to Partitioning ASIC Systems. 26th ACM/IEEE *Design Automation Conference,* pages 767-770, Las Vegas, June 1989.

[18] I. Rechenberg: Evolutionsstrategie - Optimierung technischer Systeme nach dem Prinzip der biologischen Evolution. Frommann-Holzboog Verlag, Stuttgart 1972.

[19] H.P. Schwefel: Numerische Optimierung von Computermodellen mittels der Evolutionsstrategie. Birkhäuser Verlag, Basel und Stuttgart, 1977.