IMTC 2003 - Instrumentation and Measurement
Technology Conference
Vail, CO, USA, 20-22 May 2003

Evaluating the Repair of System-on-Chip (SoC) using Connectivity

M. Choi [1], N. Park [2], V. Piuri [3], Y.B. Kim [4] and F. Lombardi [4]

[1] Dept of ECE, University of Missouri-Rolla, Rolla, MO 65409-0040, USA, choim@umr.edu
[2] Dept of CS, Oklahoma State University, Stillwater, OK 74078-1053, USA, npark@a.cs.okstate.edu
[3] Dept of IT, University of Milan, Bramante 65, 26013 Crema, Italy, piuri@dti.unimi.it
[4] Dept of ECE, Northeastern University, Boston, MA 02115, USA, {ybk, lombardi}@ece.neu.edu

Abstract – This paper presents a new model for analyzing the repairability of RSoC (Reconfigurable System-on-Chip) Instrumentation with repair process. It exploits the connectivity of the interconnected cores in which unreliability factors due to both neighboring cores and interconnect structure are taken into account. Based on the connectivity, two RSoC repair scheduling strategies, **Minimum Number of Interconnections First (I-MIN)** *and* **Minimum Number of Neighboring Cores First (C-MIN)** *are proposed. Two other scheduling strategies,* **Maximum Number of Interconnections First (I-MAX)** *and* **Maximum Number of Neighboring Cores First (C-MAX)** *are also introduced and analyzed to further explore the impact of connectivity-based repair scheduling on the overall repairability of RSoCs. Extensive parametric simulations demonstrate the efficiency of the proposed RSoC repair scheduling strategies; thereby manufacturing ultimately reliabile RSoC instrumentation can be achieved.*

Keywords – Reconfigurable System-on-Chip (RSoC), Repair, Configurability, Repairability, Connectivity, Reliability.

I. INTRODUCTION

The increasing demand on operation speed, integration density, and customizability for tomorrow's high-performance instrumentation has motivated high performance system development. System-on-Chip (SoC) technology provides potential advantages of high integration density, small interconnection delay and high system performance [8], [12], [13], [14], [16], [17], [18], [21], [23]. Thus, SoC is one of the key technology choices for high-performance instrumentation development [9]. For the purpose of customizability and repairability, embedding reconfigurable components along with ordinary cores with fixed functionality are commonly practiced [1], [2], [3], [7], [10], [11], [15], [19], [20], [22]. The SoC with reconfigurable resources is commonly referred to as *Reconfigurable System-on-Chip (RSoC)*. In this paper, connectivity-driven repair algorithms for an RSoC which exploits the reconfigurable redundancy will be proposed. *Test* and *repair* are essential processes for achieving high yielding SoCs. After the fabrication phase, each SoC undergoes a test phase where defective cores are diagnosed and identified. Usually, defective cores on RSoCs are deemed to be reworked, which means that defective cores can be repaired by reconfigurable redundancy. The overall quality of the repair process significantly affects the final quality of the repaired RSoC. However, the repair process is not free from penalty, since faulty core isolation and reconfiguration processes may affect the overall system integrity, the reconfigured interconnect structure routability, and the neighboring cores' functionality due to the serious interconnection network reconfiguration and associated programmable logic gate programming. For the extra long interconnects, even signal boosters are required to guarantee the integrity of the routed signals [5]. For more structured and reliable operations, protocol-based interconnction networks are commonly implemented for SoCs as well [4].

How densely a *repair-candidate core* (i.e., the core to be repaired since it is diagnosed as defective) is connected to the neighboring parts of the RSoC is referred to as *connectivity* [6]. If a repair-candidate core's connectivity is high, the repair process applied to the core may impair the reliability of the RSoC while it repairs the core, because of the physically associated components with the core. Thus, selection of a repair-candidate core that results in the least negative effects on the overall reliability at each repair cycle seems to be crucial in the RSoC repair process.

The objective of this paper is to extensively investigate the effect of the connectivity of repair-candidate cores on the overall yield (i.e., ratio of the total number of functional RSoCs out of the total number of fabricated RSoCs) of the repaired RSoC and to propose various repair scheduling strategies. Also, how improper repair scheduling could degrade the overall quality of repaired RSoCs will be extensively studied. Thus, results and findings from this research will be beneficial to RSoC-based digital instrumentation developers.

II. REVIEW AND PRELIMINARIES

In this work, an RSoC is modeled as a set of cores, their interconnect structure, reconfigurable interconnects, and reconfigurable logic redundancy as shown in Figure (1), in which the RSoC has six cores and corresponding interconnect structure. The repair process induces faulty core isolation, programmable logic reconfiguration, and interconnection rerouting. Although the process repairs the RSoC, the unreliability

0-7803-7705-2/03/$17.00 ©2003 IEEE

induced by the reconfiguration process (i.e., imperfect faulty core isolation, programmable logic reconfiguration and interconnection rerouting) may have negative effects on the overall quality of the repaired RSoC.

Once fabricated, embedded cores cannot be physically replaced. Thus, embedded redundancy must be practiced for better yielding SoCs. Since a number of embedded hybrid cores are usually involved to design a SoC, legacy modular redundancy scheme (i.e., embedding of extra cores to repair faulty cores) may require significant die area investment and its redundancy utilization also may be very low (i.e., unused spare cores are likely). The proposed reconfigurable redundancy architecture for SoC repair consists of two key components: *reconfigurable logic redundancy* and *reconfigurable interconnect redundancy*. The embedded cores are tested in order to identify faulty cores, if any. Then, the faulty cores and their interconnects are emulated by the reconfigurable logic and interconnect redundancy to restore the original functionality of the RSoC. The following case study clarifies the proposed RSoC core repair scheme based on the reconfigurable redundancy; suppose that an RSoC shown in Figure (1) is tested and diagnosed, and its core 4 is identified as faulty. Then, an emulated core 4 is implemented by using the reconfigurable logic redundancy and core 4 's interconnects are rerouted to the core 4 via the reconfigurable interconnect redundancy. As a result, the repaired RSoC is shown in Figure (1).

Upon proper fault simulation and analysis, the optimized amount of the reconfigurable redundancy can be determined prior to the fabrication of the RSoC. Thereby, both minimization of the die area overhead due to the redundancy and maximization of the RSoC yield can be achieved. Customized circuits can be also implemented by the reconfigurable redundancy, of course.

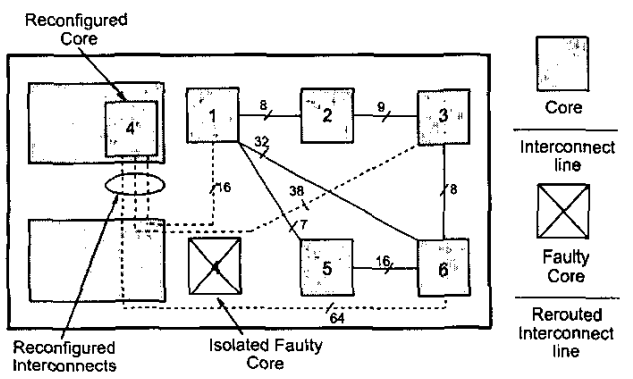

Fig. 1. Example of repaired RSoC

The following assumptions are made in this work: (1) RSoC is fabricated with embedded cores and each core can be tested and diagnosed as faulty or not. (2) No escaped cores are considered (i.e., 100% test coverage is assumed). (3) Repair process, including defective core isolation, redundancy reconfiguration and interconnect reconfiguration, can be applied to the RSoC. (4) Each core may have an uneven number of ports which connects to other core(s) via intra-chip interconnects. (5) Reconfigured and rerouted interconnects are considered as less dependable than the original interconnects due to the complexity of the resulting interconnect configuration.

As clearly addressed in the assumptions given above, the repair procedure of an RSoC has not only an advantage but also a disadvantage. Proper testing and diagnosis of the embedded cores and reconfigurable redundancy utilization may enhance the overall yield of the RSoC, since faulty cores can be replaced by reconfigured cores and rerouted reconfigurable interconnects. However, the reconfigured and rerouted interconnects may be less reliable due to the complexity of the resulting interconnect configuration. The unreliability associated with the reconfigured redundancy is modeled as the *unreliability impact factor (uif)*. For example, in Figure (1), repairing core 4 is assumed to affect neighboring (i.e., interconnected) cores 1, 3, 6 and associated interconnect structure. To accurately and effectively model the effect of both advantageous repair process and disadvantageous reconfigured and rerouted interconnects' unreliability at the same time, the following parameters are used to model the overall reliability of RSoC under repair:

N : Number of cores in the RSoC.
(i) : Probability that the i individual core is functioning, which is called *reliability*.
: Maximum number of Repair cycles.
: Overall reliability of cores in the RSoC.
i : Overall reliability of interconnect structure.
: Overall reliability of the RSoC which takes into account both the cores and the interconnect structure.
ui : Base unreliability impact factor due to the repair process penalty.
ui $_{in}$: Incremental rate of ui per each repair cycle.
ui (i,j) : Unreliability impact factor of the neighboring j core due to repair of the i core.
α : Core unreliability impact factor coefficient. $0 \leq \alpha \leq 1$. It is fully dependent on the repair technology used. As α 1, more reliability degradation due to the repair process is assumed to be applied to neighboring cores of the repair-candidate core.
$_{i,j}$: Expected number of interconnect lines between the i core and the j core.
iui (i) : Interconnect structure unreliability impact factor due to repair of the i core.
: Interconnect structure unreliability impact factor coefficient. $0 \leq$ ≤ 1. It is fully dependent on the repair technology used. As 1, more reliability degradation due to the repair process is assumed to be applied to the interconnect structure.
$_i$: Expected number of interconnect lines of the i core.

$_{in}$: Reliability increase rate of a core due to repair.
(i) : Number of interconnect lines from the i core.
(i,j) : Number of interconnect lines between the i and the j cores.
$F(\ (i,j)\ \ _{i,j})$: cumulative Poisson probability function of (i,j) (i.e., $\sum\ _{=0}^{i,j}\ \frac{e^{-}}{\quad}$).
R : Success rate of overall repair process, which is called *repairability* (e.g., if 8 out of 10 defective RSoCs are repaired during the repair process, $R = 8\ \ 10 = 80\ \ )$.
: Overall yield of RSoCs in which repair process is taken into account.

III. CONNECTIVITY-BASED RSOC REPAIR PROCESS

The repair process of a core enhances the overall reliability of the RSoC, but the process is also likely to introduce reliability degradation due to the complication of the reconfiguration process and is also prone to impair its neighboring (i.e., interconnected) cores' reliability since serious rerouting of connectivity would be experienced afterwards. Thus, the unreliability impact factor is modeled to be mainly determined by the number of interconnect lines between the repair-candidate core and its neighboring cores.

The characteristics of the repair process, so called *connectivity-based repair*, analyzed in this paper are given as follows :

1. The i core is assumed to have initial reliability of (i). Then, (i.e., overall reliability of cores) of the given RSoC is initially determined by $= \prod_{i=1}^{N}\ (i)$. Then, the overall initial reliability of RSoC (denoted by) is $=\quad i$, where i is the reliability of the interconnect structure of RSoC.
2. The test and repair processes are performed after the fabrication phase.
3. Repair of a core degrades the reliability of the neighboring cores and the interconnect structure of the RSoC under repair. It is assumed that the unreliability impact factor (denoted by ui) due to the repetition of repair cycles increases as the RSoC undergoes a number of repair cycles. $ui\ _{n}$ at the n repair cycle is given as $ui\ _{n} = ui\ _{n-1} + (1 - ui\ _{n-1})ui\ _{in}$, where $ui\ _{in}$ is the incremental rate of ui at each repair cycle due to the increasing complexity of the repair process as the number of repair cycles increases.
4. The repair of the i core is assumed to affect the neighboring (i.e., interconnected) core j, if it exists. The unreliability impact factor of the j core due to the repair of the i core is denoted by $ui\ \ (i,j)$. $ui\ \ (i,j)$ is modeled as a function of (i,j). The probability that the interconnect lines between the i and the j cores consist of exactly (i,j) lines is

$$-\ ^{i,j}\ _{i,j}^{\ i,j}\quad (i,j) \qquad (1)$$

The increase in the possibility of having more degradation due to an increment of one interconnect line from $(i,j)-1$ to (i,j) is also modeled to be determined by Equation (1), since the occurrence of degrading repair is directly influenced by the number of interconnect lines attached to the repair candidate core. Thus, without loss of generality, the cumulative Poisson probability function of (i,j) (i.e., $\sum\ _{=0}^{i,j}\ \frac{e^{-}}{\quad}$) is the reasonable one to simulate an incremental rate of ui imposed by the number of interconnect lines between the i and j cores. Thus, $ui\ \ (i,j)$ is $ui\ _{n} + (1 - ui\ _{n})\ \alpha\ F(\ (i,j)\ \ _{i,j}$, where α is a technology-dependent core unreliability impact factor coefficient and $F(\ (i,j)\ \ _{i,j})$ is a cumulative Poisson probability function of (i,j). The parameter α simulates the efficiency of the repair process technology. As α approaches to 1 and (i,j) increases, more reliability degradation is assumed to take place on the j core since the $\alpha\ \ F(\ (i,j)\ \ _{i,j})$ part approaches to 1. Thus, the reliability of the j core after the repair of the i core can be formulated as $(j)_{n} = (j)_{n-1}(1 - ui\ \ (i,j)_{n})$.
5. Repair of the i core is assumed to impair the interconnect structure as well. The reliability impact factor of the interconnect structure due to the repair of the i core is denoted by $iui\ \ (i)$, $iui\ \ (i)_{n} = ui\ _{n} + (1 - ui\ _{n})$ $F(\ (i)\ \ _{i})$, where is a technology-dependent interconnect structure damage coefficient and $F(\ (i)\ \ _{i})$ is a cumulative poisson probability function $(= \sum\ _{=0}^{i}\ \frac{e^{-}}{\quad})$ which simulates the incremental rate of the reliability degradation due to the number of interconnect lines of the i core. As approaches to 1 and (i) increases, more reliability degradation is assumed to be applied to the j core since the $F(\ (i,j)\ \ _{i,j})$ part approaches to 1.
6. The reliability of the i core after the repair process is given by $(i)_{n} = (i)_{n-1} + (1 - \ (i)_{n-1})\ _{in}$.
7. The overall reliability of the cores on RSoC after the n repair cycle becomes $\ _{n} = \prod_{i=1}^{N}\ (i)_{n}$.
8. The overall reliability of the interconnect structure on RSoC after the n repair cycle can be formulated as $i_{n} = i_{n-1}(1 - iui\ \ (i)_{n})$.
9. The overall reliability of the RSoC after the n repair cycle then becomes $\ _{n} =\ \ _{n}\ \ i_{n}$.

IV. CONNECTIVITY-BASED RSOC REPAIR SCHEDULING

Every core on an RSoC is tested after the fabrication phase. The i core can be tested and diagnosed as non-faulty with the probability of (i) and as faulty with the probability of (i). If there is only one faulty core detected during the test phase, the core will be isolated and repaired. If more than one faulty core is detected during the test phase, the order of repair (referred to as the *repair schedule*) must be properly arranged. In each repair cycle, in other words, selecting an appropriate repair-candidate core which has the least impact on the overall RSoC

reliability is a natural choice for optimal scheduling.

The RSoC structure shown in Figure (1) can be viewed as a weighted graph with six vertices and nine weighted edges. A simple way to represent the graph is to use a two-dimensional array so called an *adjacency matrix* representation. The space requirement of the representation is (N^2) where N is the number of cores on the RSoC.

If the RSoC is sparsely interconnected, a better solution is *adjacency list* representation. The space requirement for this representation is $(N+$ $)$ where N is the number of cores and is the number of edges between cores on the RSoC. For RSoCs with a greater number of cores which are sparsely interconnected, the adjacency list representation can save the space requirement. For RSoCs with a lesser number of cores which are densely interconnected, the adjacency matrix is the choice. One of the two representations can be chosen accordingly, in practice.

For the proposed RSoC model, the number of interconnect lines and the number of neighboring cores attached to a repair-candidate core determines the resulting yield of the RSoC after each repair cycle. Four possible repair scheduling strategies are proposed as follows:

Minimum Number of Interconnects First (I-MIN) - Among those diagnosed as faulty cores, the one which has the smallest number of interconnect lines is to be repaired first.

Maximum Number of Interconnects First (I-MAX) - Among those diagnosed as faulty cores, the one which has the largest number of interconnect lines is to be repaired first.

Minimum Number of neighboring Cores First (C-MIN) - Among those diagnosed as faulty cores, the one which has the smallest number of neighboring cores is to be repaired first.

Maximum Number of neighboring Cores First (C-MAX) - Among those diagnosed as faulty cores, the one which has the largest number of neighboring cores is to be repaired first.

Since I-MAX and C-MAX repair scheduling strategies are supposed to repair the most reliability degrading core first, they do not have advantages in practice. However, they are also analyzed to be compared with the I-MIN and C-MIN repair scheduling strategies.

V. PARAMETRIC ANALYSIS

In this section, the effects of the connectivity-based RSoC repair scheduling are investigated through numerical experiments. An RSoC system with $N = 15$, $(i) = 0.99$ for all i, and $i = 0.9999$ is considered. The yield of the RSoC before an application of the repair process can be calculated as a series product of the (i) of all the cores (i.e., $\prod_{i=1}^{N}$ (i)) and the yield of the interconnect structure (i.e., i). In Table (I), the overall RSoC yield and $^-$ are given where $^-$ is subdivided into six categories according to the number of defective cores on the RSoC (denoted by). The following can be observed from Table (I):

Among those 14.0028% RSoCs with defects, 13.0298% of them have one defective core identified, 0.9213% of them have two defective cores identified, 0.000403% of them have three defective cores identified, 0.000012% of them have four defective cores identified, and 0.000016% of them have more than five defective cores identified.

Since RSoCs with 5 are very few and then almost ignorable, the maximum allowed number of repair cycles $= 5$ is applied.

0.00086% RSoCs in the category i (i.e., defective interconnect structure) does not have defective cores, but they have a defective interconnect structure. Since RSoCs in the i category are very few, no repair process is applied in this example.

To compare the proposed repair scheduling strategies, the values of $_{in}$, ui , and ui $_{in}$ are set to 0.1 and the value of $_i$ is set to 9, arbitrarily. In Tables (II) and (III), the repair performances of those proposed strategies, measured in the percentage of repaired RSoCs at each repair cycle, are shown. For example, 13.0298% of RSoCs contain one defective core and 62.4108% of them are repaired in the first repair cycle of I-MIN, and 0.9213% of RSoCs contain two defective cores and 27.4829% of them are repaired in the second repair cycle, and so on.

TABLE II
PERFORMANCE COMPARISON OF THE PROPOSED REPAIR STRATEGIES AT EACH REPAIR CYCLE WHERE 0.05

	1	2	3	4	5
I-MIN	62.4108%	27.4829%	0.0661%	0%	0%
I-MAX	33.7273%	0.09249%	0.5%	0%	0%
C-MIN	67.2667%	33.2356%	8.1886%	0%	0%
C-MAX	36.3137%	9.2478%	1.4888%	0%	0%

TABLE III
PERFORMANCE COMPARISON OF THE PROPOSED REPAIR STRATEGIES AT EACH REPAIR CYCLE WHERE 0.5

	1	2	3	4	5
I-MIN	59.3532%	14.6424%	0.2481%	0%	0%
I-MAX	0.7552%	0.0109%	0%	0%	0%
C-MIN	45.1135%	9.8665%	0.4963%	0%	0%
C-MAX	2.5941%	0.0868%	0%	0%	0%

By comparing the results shown in Tables (II) and (III), the following can be observed:

1. Even with relatively small values of α and $= 0.05$ (i.e., less core and interconnect degradation due to the repair

TABLE I

AND ¯ OF THE GIVEN RSOC WITHOUT REPAIR

85.9972%	14.0028%					
	$=1$	$=2$	$=3$	$=4$	$=5$	i
	13.0298%	0.9213%	0.000403%	0.000012%	0.000016%	0.000086%

process), the repair scheduling plays an important role in the RSoC repair process. Thus, it is shown that I-MIN and C-MIN outperform I-MAX and C-MAX at every repair cycle.

2. As relatively larger values of α and = 0.5 are applied (i.e., more core and interconnect degradation due to the repair process), the difference in the repairability between I-MIN (C-MIN) and I-MAX (C-MAX) becomes even more clear.
3. An appropriate selection of the repair schedule definitely affects repairability regardless of the values of α and .

In Figures (2)-(4), (i.e., overall yield of the RSoC) of I-MIN at different values of N (i.e., 5, 10, and 15), α and (i.e., 0.05 and 0.25), and (i) for all i (i.e., 0.8 - 1.0) are shown versus I-MAX. In Figure (3)-(5), of C-MIN at different values of N (i.e., 5, 10, and 15), α and (i.e., 0.05 and 0.25), and (i) for all i (i.e., 0.8 - 1.0) are shown versus C-MAX. By comparing the results of Figures (4)-(5), the following observations can be drawn:

1. Using proper connectivity-based repair scheduling strategies (i.e., I-MIN and C-MIN), a higher of RSoC can be achieved.
2. I-MIN and C-MIN always outperform I-MAX and C-MAX.
3. As α and increase, the difference between of I-MIN and of I-MAX increases. It is the same for C-MIN and C-MAX.
4. With relatively smaller α and values, of both I-MIN and C-MIN perform similarly. However, I-MIN performs better than C-MIN as α and increase.
5. In practice, C-MIN is likely to be the choice when smaller α and values are applied, since it counts only the number of neighboring cores which is simpler than counting the number of interconnect lines.
6. In practice, I-MIN is likely to be the choice when larger α and values are applied since it has less impact on of RSoC than C-MIN.

VI. DISCUSSION

This paper has presented a new model for analyzing the repairability of RSoC instrumentation with repair processes based on the effect of the connectivity of the repair-candidate core where reliability degradation of both neighboring cores and interconnect structure due to the complexity of the reconfigured logic and interconnect redundancy is taken into account. Two approaches, *Minimum Number of Interconnections First (I-MIN)* and *Maximum Number of Neighboring cores First (C-MIN)* have been proposed. Two other scheduling policies, *Maximum Number of Interconnections First (I-MAX)* and *Maximum Number of Neighboring cores First (C-MAX)* also have been introduced and analyzed, and it has been shown how improperly scheduled repair processes could impair the overall repairability of RSoCs under repair. Extensive parametric analysis and comparison of the proposed approaches have demonstrated the efficiency of the proposed RSoC repair scheduling strategies (i.e., I-MIN and C-MIN). From the results, it is obvious that a higher repairability of RSoCs can be expected when proper RSoC repair scheduling strategies such as I-MIN and C-MIN are applied. Also, it has been shown that I-MIN tolerates a higher core and interconnect reliability degradation due to the repair process (i.e., higher α and) than C-MIN does, while C-MIN results in a higher repairability when less core and interconnect reliability degradation due to the repair process (i.e., lower α and) is assumed.

References

[1] J. Greenbaum, "Reconfigurable logic in SoC systems," *Proceedings of the IEEE 2002 Custom Integrated Circuits Conference*, pp. 5-8, May. 2002.

[2] S. Lee, S. Yoo and K. Choi, "Reconfigurable SoC design with hierarchical FSM and synchronous dataflow model," *Proceedings of the Tenth International Symposium on Hardware/Software Codesign, 2002.*, pp. 199-204, May. 2002.

[3] B. Lewis, I. Bolsens, R. Lauwereins, C. Wheddon, B. Gupta and Y. Tanurhan, "Reconfigurable SoC–what will it look like," *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition, 2002.*, pp. 660-662, Mar. 2002.

[4] L. Benini and G. De Micheli, "Networks on chips: a new SoC paradigm," *IEEE Computer*, Vol. 35 Issue 1, pp. 70-78, Jan. 2002.

[5] A. Nalamalpu, S. Srinivasan and W.P. Burleson, "Boosters for driving long onchip interconnects - design issues, interconnect synthesis, and comparison with repeaters," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, Issue 1, pp. 50-62, Jan. 2002.

[6] M. Choi,N. Park, F. Meyer and F. Lombardi, "Connectivity-based multichip module repair", *Proceedings of 2001 IEEE Pacific Rim International Symposium onDependable Computing* , pp. 19-26, Dec. 2001.

[7] R. Hartenstein, "Reconfigurable computing: a new business model-and its impact on SoC design," *Proceedings. Euromicro Symposium on Digital Systems Design, 2001.*, pp. 103-110, Sep. 2001.

[8] R.A. Bergamaschi, S. Bhattacharya, R. Wagner, C. Fellenz, M. Muhlada, F. White, J.-M. Daveau and W.R. Lee, "Automating the design of SoCs using cores," *IEEE Design & Test of Computers*, Vol. 18, Issue 5, pp. 32-45, Sep.-Oct. 2001.

[9] M. Oberle, R. Reutemann, R. Hertle, Qiuting Huang, "A 10-mW two-channel fully integrated system-on-chip for eddy-current position sens-

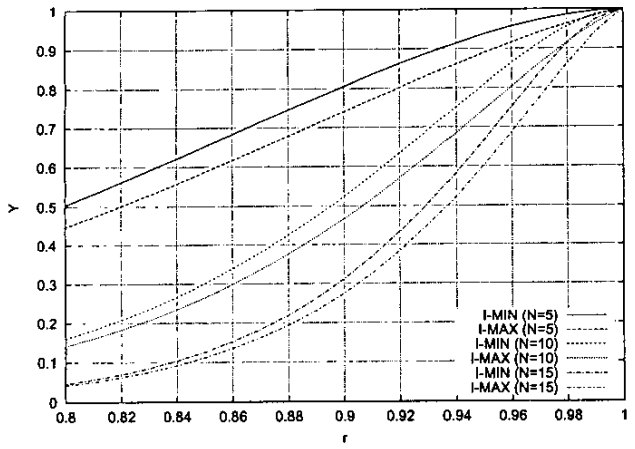

Fig. 2. Repairability of I-MIN and I-MAX at 0.05

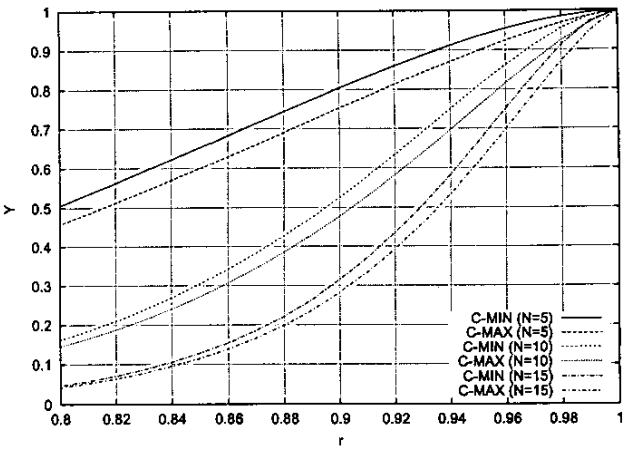

Fig. 3. Repairability of C-MIN and C-MAX at 0.05

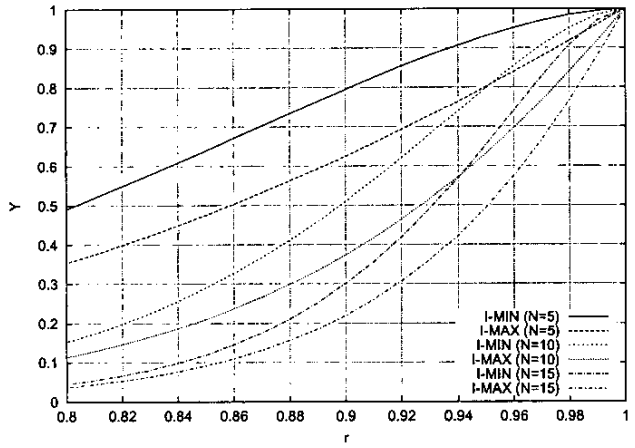

Fig. 4. Repairability of I-MIN and I-MAX at 0.25

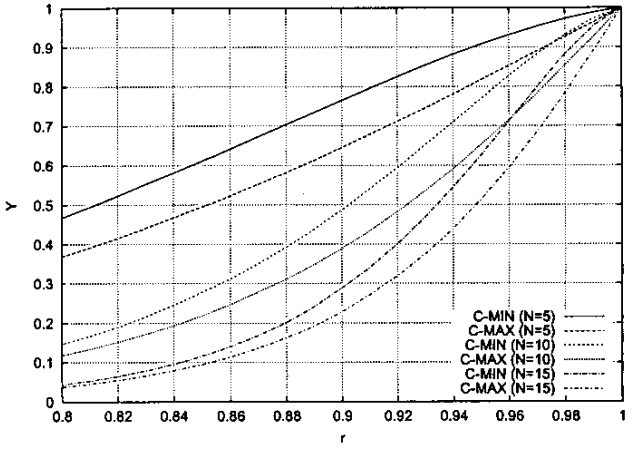

Fig. 5. Repairability of C-MIN and C-MAX at 0.25

ing [in biomedical devices]," *IEEE Journal of Solid-State Circuits*, Vol. 37, Issue 7, pp. 916-925, Sep. 2001.

[10] B.I. Hounsell and T. Arslan, "Programmable multiplierless digital filter array for embedded SoC applications," *Electronics Letters*, pp. 735-737, Jun. 2001.

[11] S.J.E. Wilton and R. Saleh, "Programmable logic IP cores in SoC design: opportunities and challenges," *IEEE Conference on Custom Integrated Circuits, 2001.*, pp. 63-66, May. 2001.

[12] T. Bautista and A. Nunez, "Quantitative study of the impact of design and synthesis options on processor core performance," *19th IEEE Proceedings on VLSI Test Symposium*, pp. 169-175, Apr.-May 2001.

[13] Shang-yi Chiang, "Foundries and the dawn of an open IP era," *IEEE Computer*, Vol. 34, Issue 4, pp. 43-46, Apr. 2001.

[14] S. Ravi, G. Lakshminarayana, N.K. Jha, "Testing of core-based systems-on-a-chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 20, Issue 3, pp. 426-439, Mar. 2001.

[15] Z. Huang and S. Malik, "Managing dynamic reconfiguration overhead in systems-on-a-chip design using reconfigurable datapaths and optimized interconnection networks," *Proceedings of Design, Automation and Test in Europe Conference and Exhibition 2001.*, pp. 735-740, Mar. 2001.

[16] F.J. Meyer, N. Park, "Predicting the yield efficacy of a defect-tolerant embedded core," *Proceedings on 2000 IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, pp. 30-38, Oct. 2000.

[17] S. Dey, D. Panigrahi, Li Chen, C.N. Taylor, K. Sekar and P. Sanchez, "Using a soft core in a SoC design: experiences with picoJava," *IEEE Design & Test of Computers*, Vol. 17, Issue 3, pp. 60-71, Jul.-Sep. 2000.

[18] I. Ghosh, S. Dey and N.K. Jha, "A fast and low-cost testing technique for core-based system-chips," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 19, Issue 8, pp. 863-877, Aug. 2000.

[19] H. Singh, M.-H. Lee, G. Lu, F.J. Kurdahi, N. Bagherzadeh and E.M. Chaves Filho, "MorphoSys: an integrated reconfigurable system for data-parallel and computation-intensive applications," *IEEE Transactions on Computers*, Vol. 49, Issue 5, pp. 465-481, May 2000.

[20] S. Knapp and D. Tavana, "Field configurable system-on-chip device architecture," *Proceedings of the 2000 IEEE Custom Integrated Circuits Conference*, pp. 155-158, May 2000.

[21] R.K. Gupta and Y. Zorian, "Introducing core-based system design," *IEEE Design & Test of Computers*, Vol. 14, Issue 4, pp. 15-25, Oct.-Dec. 1999.

[22] S. Winegarden, "Bus architecture of a system on a chip with user-configurable system logic," *IEEE Journal of Solid-State Circuits*, vol. 35, Issue 3, pp. 425-433, May 1999.

[23] A.M. Rincon, C. Cherochetti, J.A. Monzel, D.R. Stauffer and M.T. Trick, "Core Design and System-on-a-Chip Integration", *IEEE Design & Test of Computers*, Vol. 14, Issue 4, Oct.-Dec. 1997