An Effective Metric for System Survivability

Jin-Ho Ahn[1], Euntai Kim[1], Hyekyung Cho[2], Hongil Yoon[1], and Sungho Kang[1]
[1] Department of Electrical and Electronic Engineering, Yonsei University
134 Shinchon-dong, Sodaemun-gu, Seoul, 120-749, Korea
[2] Department of Information and Communication Engineering, Hansung University
389 Samseon-dong 3-ga, Seoungbuk-gu, Seoul, 136-792, Korea
E-mail: sominaby@soc.yonsei.ac.kr, hkcho@hansung.ac.kr, {etkim, hyoon, shkang}@yonsei.ac.kr

Abstract: A system survivability means the ability to achieve the mission although there may be some faults inside the system. In this paper, we propose a quantitative formalism of survival index (*SI*) which is extremely suitable for an SoC. The proposed *SI* is evaluated using a network-processor, and the results show *SI* can be a resonable metric for an SoC survivability.

1. Introduction

Some applications such as space shuttles and deep-sea submarines demand the most stringent reliability requirement due to the irrecoverable loss from operation failures. They must endure system malfunctions and carry out their missions in a timely manner without any external assistance. Survivability of a system is defined as the ability to achieve its missions though there are some faults inside the system. However, all systems do not need to have the similar survivability. Furthermore, it should be considered in the early stage of system development to determine the proper survivability of the system according to the system objective. Therefore, it is necessary to have a metric for measuring system survivability. While there have been some related works, most of them are just concerned about the modeling of system dependability excluding the performance loss such as hardware or timing overhead [1]-[2]. In this paper, we propose a new effective metric for an SoC survivability called as survival index (*SI*). *SI* includes dependability parameters and resource factors according to survival functions. Survival function means the possible operation to increase a system survivability. System-level tests or fault-tolerant functions are good examples of survival functions.

2. Definition

2.1 Survival Index

The probability of a failure for an SoC, P, is defined as

$$P = 1 - e^{-\mu T}$$

where μ is the failure rate of the SoC and T is the mission time [3]. The failure rate μ is the expected number of failures of the SoC per a given time period T. μ is given by

$$\mu = \sum \lambda \cdot p$$

where λ is the rate of occurrence of a fault, and p is the probability of the fault leading to a failure in the SoC. To reduce the failure rate, λ or p should be reduced. λ is related to the fault detection functions and p is related to the fault recovery. However, survival functions have both strong and weak points. For example, the supplement of hardware logic for fault detection can increase the probability of fault detection, but it can also increase the probability of fault occurrence. Therefore, proper survival functions should be selected according to the mission of an SoC. The proposed *SI* is represented as follows:

$$SI = e^{-\frac{\mu \cdot nT}{s}}, \quad P = 1 - SI$$

where μ is the failure rate, s is the effective survival rate, n is the number of operations of the SoC, and T is the time duration for the operation, respectively. In other words, nT indicates the total operation time for the effective survival rate s. Actually, *SI* and nT are determined by the SoC usage. Therefore, we can predetermine s satisfying *SI* and nT. If s is calculated, one can choose the proper survival functions on the basis of s.

The effective survival rate s is described as

$$s \cong \sum \frac{\alpha_f \cdot \Delta_f}{C_f} \qquad (if\ f = 0, s = 1)$$

where Δ_f is the reduction ratio of the failure rate due to the survival function f, C_f is the cost for Δ_f, respectively. α_f indicates the relative feasibility and significance of the function f according to the objective of the SoC. For example, the SoC for consumer electronics should be affordable. Thus, the survival functions with low cost will be preferable. One can include the preference per function using α_f considering the SoC application. s equals to the summation of the effective survival rate of survival functions used for the SoC.

2.2 Survival Function

There are many survival functions which are widely used in industry [4]-[5]. All of them are the similar objective, but their approaches are some different. We classify the survival functions roughly into two categories : Innate and Acquired. An innate survival function is executed at the initial stage of an SoC design, and cannot be re-executed after the SoC development is finished [6]. An acquired survival function can be executed whenever necessary for the whole lifetime of the SoC.

- Innate Survival Function

Innate survival functions can be divided as follows:

- Test Preparation Function
- Test Execution Function
- Test Quality Function

A test preparation function is related to the test pattern generation. A test execution function equals the function of a test equipment. A test quality function is the most serious factor in innate survival functions, and has a deep relation with a test reliability. A test quality index influencing the survivability is described as

$$F_{quality} = F_{reliable} + F_{ploss}$$

$$F_{reliable} = \left(Y^{(1-fc_{DFT})} - Y^{(1-fc_{noDFT})}\right) \cdot (1 + P_{reduction})$$

$$F_{ploss} = \frac{O_{noDFT} - O_{DFT}}{O_{noDFT}}$$

where $F_{quality}$ is the survivability due to a test quality and consists of a reliability and performance loss index. A reliability index indicates the ability to detect a fault and the reliability of results. In the formula of reliability index, Y is a yield, fc is a fault coverage, $P_{reduction}$ is a test power reduction ratio due to DFT (Design for Test). As DFT causes the performance loss of an SoC, the loss can be considered to build a test quality index. The timing overhead by the growth of critical path is a typical one. O is overhead factors that cause performance losses in F_{ploss} equation.

- Acquired Survival Function

Innate survival functions can nomally remove physical defects in an SoC. However, it cannot be adjustable to the SoC in the market. Thus, some surival functions that can inspect the operation of an SoC periodically are required. Most fault-tolerant functions are classifiied in this area.

We illustrate two acquired surivial functions. One is a concurrent error detector (CED). A CED analyzes the operation results of a hardware unit at real-time, and reports the matter to outside if there is any problem. For example, the survivability index of CED, F_{CED}, for a packet processor in a network processor (NP) can be descibed as

$$F_{CED} = \frac{1}{N_{NF}} \sum_{N_{CED}=0}^{N_{NF}} \left(N_{CED} \cdot e^{-\mu_p N_{CED}}\right)$$

where N_{NF} is the operation number of the packet processor in an unit time, N_{CED} is the operation number of the CED in the same time duration as N_{NF}. μ_p is a basic failure rate of the packet processor. As it can be seen from the equation, the survivability due to the CED varies through the operation number of the CED. The other is a test point monitor (TPM). A test point is the place that one can monitor efficiently whether a fault occurs, and is selected by analyzing the whole SoC. A cone point is a good example. The availability of selected test points and the monitoring number of the points dominate the SoC survivability. The moderate number of test points is also significant.

3. Evaluation

In order to evaluate the proposed metric, a NP based on an OSI Layer 3 forwarding IPv4 packets is used. The major blocks of the NP include a pre-processing unit, a packet processor, an embedded processor, several post-processing units, and memory components.

For testing at the core level, full-scan testing is deployed for the pre-processing and the post-processing units. Scan vectors generated by an ATPG tool are provided externally. The packet processor uses logic BIST. The logic BIST (Built-In Self Test) uses a 32bit linear-feedback shift register (LFSR) to generate pseudo-random test vectors and uses a signature analyzer to compact the test result. The memory modules use memory BIST that runs a Marching C algorithm for testing. All the blocks and the associated test structures are encapsulated with P1500-compliant wrappers. Besides DFT logics, a CED and a TPM are installed. The CED examines the packet processor, and the TPM monitors the pre-processing and the post-processing units.

Some statistical approaches are required to establish the accurate effective survival rate. However, Δ, C, and α per survival function used in this work are assumed on the basis of heuristic approximations. The estimated s in the NP is shown in Table 1.

While Δs and Cs of innate functions are predetermined, those of acquired functions can be varied. α is also able to be alterd by the architecture of NP. As a common NP has a large embedded RAM, memory BIST is more important than other functions. According to Table 1, the probability of a failure of the NP will be reduced about 65 ~ 80% in case that μnT is 1. One can conjecture the probability of a failure of an SoC in this way.

Table 1. Effective Survival Rate of NP

Function Name	Δ	C	α
Full Scan	1.1	1	1.2
Logic BIST	1.1	1	1
Memory BIST	1.1	1	1.5
CED	0 ~ 1.5	1	1.2
TPM	0 ~ 1.2	1	1
s	4.07 ~ 7.07		

4. Conclusion

The purpose of this paper is to design a metric for an SoC survivability that can be easily adapted in SoC development. Though some constraints remain, the proposed SI promises well for its feasibility.

References

[1] J. C. Knight, "Dependability of Embedded Systems," Proc. of 24th International Conference on Software Engineering, pp. 685-686, 2002

[2] N. Wattanapongsakorn and S. Levitan, "Integrating Dependability Analysis into the Real-time System Design Process," Proc. of IEEE Annual Reliability and Maintainability Symposium, pp. 327-334, 2000

[3] D. K. Pradhan, "Fault-Tolerant Computer System Design," Prentice Hall, 1996

[4] P. K. Lala, "Fault Tolerant and Fault Testable Hardware Design," Prentice Hall, 1985

[5] S. Mitra, N. Seifert, M. Zhang, Q. Shi and K. S. Kim, "Robust System Design with Built-In Soft-Error Resilience," IEEE Computer, Vol. 38, No. 2, pp. 43-52, February 2005

[6] M. L. Bushnell and V. D. Agrawal, "Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits," Kluwer Academic Publishers, 2000