20th IMEKO TC4 International Symposium and
18th International Workshop on ADC Modelling and Testing
Research on Electric and Electronic Measurement for the Economic Upturn
Benevento, Italy, September 15-17, 2014

Failure Analysis Based on Emulation Systems

L. Angrisani[1], F. Cennamo[1], G. Ianniello[1], A. Stellato[2]

[1] *Dip. di Ingegneria Elettrica e Tecnologie dell'Informazione, Università degli Studi di Napoli Federico II, Via Claudio 21, Napoli – 80125 – Italy, +390817683866, +390817683816*
{angrisan, giacomo.ianniello, felice.cennamo}@unina.it

[2] *TME Srl, via C. Alberto Dalla Chiesa, Portico di Caserta-81050-Italy, +390823794628*
aniello.stellato@tmesrl.net

***Abstract* – To increase the throughput of electronic manufacturing companies, design, prototyping, production, installation and maintenance processes of electronic devices are generally complemented by a number of performance and parametric tests, known as *Failure Analysis* (FA). In this paper, major FA proposals are considered. In particular, two non-invasive solutions are presented in detail: an advanced boundary scan, using an FPGA to speed up the tests, and a logic combinational test procedure, performed by means of a fault injectable emulation system. Moreover, an innovative CPU emulation approach is proposed, particularly suited to devices that include CPU chips. In order to detect hardware faulty, the approach provides for the emulation of the same boot code designed for DUT normal operation.**

I. INTRODUCTION

In recent decades, the company has experienced rapid changes due to the wide diffusion of electronic applications in social life, in work, in the industry. Actually, the production of electronic devices (systems, devices, boards, components) is growing at an unprecedented rate.

The main life cycle stages of an electronic device, such as the design, prototyping, production, installation and maintenance, must be accompanied by the frequent and thorough measurement and test processes, including performance, parametric and functional testing, Failure analysis (FA). In particular, the *FA* is essential when some quality indicators, such as the yield, exceed critical thresholds.

Application of *FA* ensures improved designs, results in higher yields and fewer field returns, and shortens the development cycle since the sources of failure that cause redesign and manufacturing changes are identified early. Considering the bottom line, investment in *FA* personnel and equipment results in reduced cost and improved quality. In [1], the general approaches of *FA* has investigated. First, the authors have divided the *FA* process in six main steps:

1. *Verify that there is a failure;*
2. *Characterize the symptoms of the failure;*
3. *Verify that these symptoms caused the observed failure;*
4. *Determine the root cause of the failure;*
5. *Suggest (and preferably participate in) corrective action;*
6. *Document the results of the failure analysis;*

In addition, several techniques and tools available for integrated circuits (IC) *FA* are described, such as:

(i) *Electrical Measurement*: the current-voltage (*I-V*) characteristic of the device under test is measured and compared with the ideal one, so, making decision for failure;
(ii) *Optical, Infrared, and Scanning Optical Microscopy*: Method for direct and nondestructive examination of topographical features on IC's.
(iii) *Physical Techniques*: invasive method that consist of probing and cutting the circuit under test, allowing the analyst to trace electrical signals on IC's to determine the functionality of the circuit;
(iv) *Thermal Imaging Analysis*: These techniques may be necessary for defect localization when there are no readily detectable symptoms such as light emission. These techniques can be nondestructive and relatively convenient to perform;

However, in order to perform a complete analysis of the circuit under diagnosis, no-one technique, taken individually, is completely exhaustive. Nevertheless, more techniques need to be performed together.

Based on the analysis performed, another classification of diagnosis techniques can be accomplished. In particular, there are two main categories: (1) cause-effect analysis [2], [3], and (2) effect-cause analysis [4], [5].

The first category, cause-effect analysis, attempts to build a fault dictionary that stores the faulty response of every modeled defect in the device under test (DUT). During the diagnosis process, the output responses of the DUT are compared with the syndromes stored in the fault dictionary. Faults are detected when both responses match well. However, these approaches require a specific fault model. The correctness will reduce when the fault model does not characterize the actual defects very well. Furthermore, fault dictionaries require large storages, and the storage requirements grow rapidly as the design size

ISBN-14: 978-92-990073-2-7

or the defect number increases.

The second category, effect-cause analysis, deduces the locations of faults by observing the faulty behaviors of the DUT and the fault-free behaviors. This process does not have to build a fault dictionary. It is more memory-efficient and scalable with the design size. However, this category of diagnosis techniques is still very time consuming and could lengthen the time to market.

Following, two noninvasive methods for *FA* has been detailed. In particular, an advanced boundary scan *FA* technique, proposed in [6] is described. The method consist of introduce a programmable FPGA module in boundary scan chain. The module is programmed to providing the needed test functionalities in order to test the several devices in the DUT. In the section II more details are presented. Then, *FA* techniques based on logic emulation are analyzed. Logic emulation systems are very fast schemes, so they are widely used for real-time operations. They are generally based on logic fault injection, and logic verification. More details about these techniques are presented in the section III.

Finally an *FA* scheme based on CPU-Emulation, is presented in the section IV. The scheme can be considered as a combination of features and advantages of boundary scan and logic emulation techniques.

II. BOUNDARY SCAN TECHNIQUES

Boundary scan based testing has been used successfully for many years. However, the growing complexity of integrated circuits and board structures make this kind of testing needs more and more time, increasing the gap between the testable and available features. Additionally new high speed circuits, operating at high frequencies (e.g. DRAM), cannot be tested within their real-time specification because of the slow frequencies boundary scan is working with. Therefore many components and systems cannot be tested to a degree that satisfies the standard that many companies set for their products.

In [6], the authors have proposed a new approach that makes use of a programmable logic (FPGA) inside the boundary scan chain to test the components connected to this FPGA. The approach is based on the possibility to create almost any test logic inside the FPGA´s structure to speed up the testing significantly and to provide higher test coverage compared to standard boundary scan testing. The logic is used to implement higher-level functions up to the whole functionality of a simple processor inside the FPGA. These functions are made available for testing purposes via JTAG (joint test action group). This eliminates the need to emulate the system´s behavior via the relatively slow JTAG interface and implements part of the test functions directly in FPGA-hardware.

The approach in [6] involves the implementation of logic functions, inside the FPGA, suitable to test the connected components. However, for every component that should be tested such a device model is needed. The model includes a description of the device interface and functionality as well as parameters. Also, In [6], the authors have included in the models the related logic test functions. Finally, with these models, the already existing test software and new extension software, the test engineer can be create the bit-file to program the FPGA, without writing any other additional software. The potentiality of using the FPGA is in the parallelism of the custom logic test functions. The test procedures of the connected component can contemporary run, reducing the time spent for testing. Also in the FPGA can be implemented a custom processor, resulting in testing of communication busses.

III. *FA* TECHNIQUES BASED ON LOGIC EMULATION

Logic emulation systems are widely used for fast prototyping, real-time operation, and logic verification. It consists of both hardware and software. Gate-level designs can be implemented with logic emulation systems, which consist of complex programmable logic device (CPLD) chips. The software part of the emulation system consists of circuit compilation, CPLD configuration, and fault discriminator (analyze the responses of the emulator to locate the faults). Since, the difficulty in fault emulation is mainly in fault injection, it will be the bottleneck if reprogramming is required to inject a fault. Therefore, a combinational circuit fault diagnosis using logic emulation is proposed in [10]. In the paper, the authors use a logic emulator in order to reproduce the combinational circuits in the DUT. So, a controlled pattern generator is used to inject erroneous stimulus in the DUT and in the logic emulator. Finally their responses of the two devices are compared and a fault discriminator locates the most likely faults. In the approach proposed in [10] the hardware emulator allows the insertion of fault circuits. Fault injection is done by using the fault injection elements and a fault injection scan chain.

Finally, in order to reduce the fault emulation time the authors make use of circuit partitioning techniques. These techniques consist of logically partitioning the circuit into subcircuits. Candidate subcircuits that potentially contain the defect sites are identified and further partitioned until the defect site is located [11].

IV. CPU-EMULATION BASED *FA* TECHNIQUES

Nowadays overall systems consist of smart units, in which the CPU is the core of them, since it controls the most of DUT functionalities. For diagnosis purposes, considering DUTs containing microcontroller systems, some limitations can be identify in the methods detailed

above, In such systems, the internal CPU manages the various control and communication peripherals to interfacing with the microcontroller connected chips inside the DUT. Because the difficulty to interfacing to the communication chains between CPU and the peripheral chips, it is difficult to use fault diagnosis techniques based on advanced Boundary scan diagnosis. Also, combinational logic emulation based techniques are hopeless. However, in this scenario, it can be used the CPU itself for the diagnosis and testing purposes. Methods, based on CPU-EMULATION , provide advantages in terms of invasively, speed and versatility.

In normal operations CPU executes algorithms to control and communicate with peripheral chips. Whereas, CPU can run diagnosis algorithms in order to test the same peripheral chips inside the DUT. The execution speed is the max allowed by the CPU and/or the peripheral chip itself.

In addition to its normal functionalities, the CPU can be controlled by an external application, such as CPU-Emulation system (CES), that is able to connect and fully control the CPU.

The CES is an embedded smart system including a logic CPU. This system is able to fully control the target to which it is connected. In particular the CES functions as interface between the test engineer program and the target device.

A general scheme of *CES* is shown in Fig. 1. In the figure, the RAM memory is divided in several sub blocks relatively to their specific functions. In particular the Emulation RAM is a backup. Since any CPU allows to set one or more breakpoints, a RAM sub block of the CES is purposely reserved to

The main functions allowed by means of a CES are listed in the following:

Display: different user program languages (ASM, SCRIPT mode); different data formats and structures; CPU and peripheral registers (bits and sense).

Program control: Step and Go; program (synchronous) and data (asynchronous) breakpoints.

Analyzer: Program flow and selective trace

Performance Analyzer: Performance analysis and statistic information using analyzer memory and trigger unit.

Code Coverage using flag ram Controlling of read and write access: variables and data areas

A. Test Algorithms

Test algorithms strictly depend to the CPU and connected peripheral chips, so to the application to be implemented. There are two test methods to be implemented.

The first method uses test algorithms, specially designed to inspect the full functionalities of the chips. Meanwhile the second one consists of step by step debugging the boot code designed for normal operations of the DUT. Generally debugging is used for removing bugs from software. Also, debugging of codes without bugs can be used to find hardware problems. For example, debugging communication algorithms between CPU peripherals and external chips can discover hardware bugs such as signal integrity problems, or external chips malfunctioning. The procedure consists of four main steps, as listed below:

(i) Identify the key points in the boot code; (ii) insert the key breakpoints to stop the code execution for each key point; (iii) configure CPU target with the key breakpoints; (iv) start the debug.

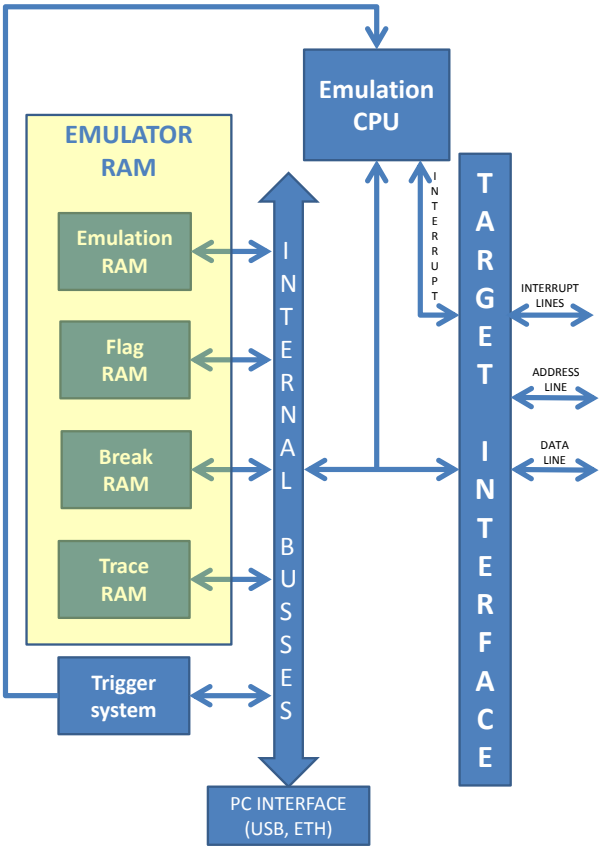

Fig. 1: General scheme of a CPU-EMULATION system,

Fig. 2 shows the flowchart of the boot code emulation-based method.

The key idea beyond this method is that any control and communication function in the boot code algorithm uses the related hardware connected to the CPU such as peripheral chips and bus. So if code execution crashes somewhere, the involved hardware, like chips, drivers, and/or PCB tracks, has some problem.

The proposed approach does not test all CPU peripherals but only the ones used by the nominal boot code at full speed, taking advantages in terms of time reduction. Then test engineers don't have to design no test algorithms but they only have to divide the existing

nominal boot code identifying the control and communication functions. This takes advantages in terms cost reduction for the tests design even so preserving full covering at full speed of the DUT nominal functionality.

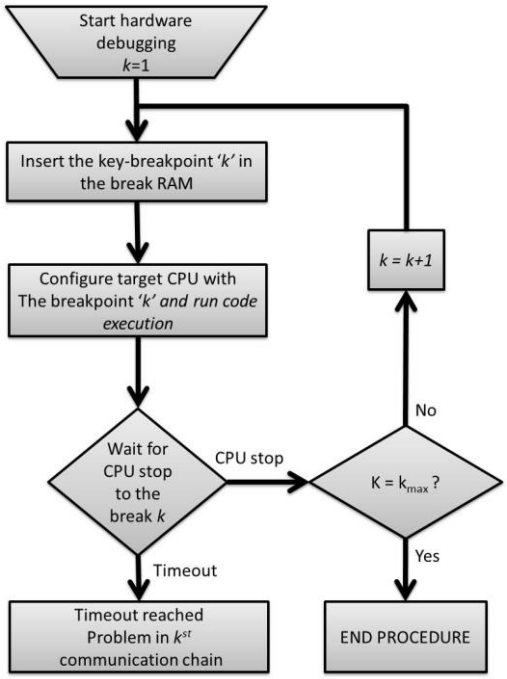

Fig. 2: flowchart boot code emulation based method

V. CONCLUSION

In this paper, the role of the *Failure Analysis* in the electronic production has been investigated, and the main *FA* techniques has been studied. Then two noninvasive methods for *FA* have been detailed. At first, an advanced boundary scan approach including FPGA in boundary chain in order to speed up the test of fast chips like memories, and fast communication protocols. The second method involves combinational logic emulation performed by means of fault injection. Finally, a new CPU-emulation approach has been proposed. The approach to perform tests using the boot code algorithms, designed for normal operation of the DUT. Since the proposed approach does not test all CPU peripherals but only the ones used by the nominal boot code, it takes advantages in terms of time and cost reduction for the tests and fully covering at full-speed of the DUT nominal functionality.

VI. REFERENCES

[1] Soden, J.M.; Anderson, Richard E., "*IC failure analysis: techniques and tools for quality reliability improvement*," Proceedings of the IEEE , vol.81, no.5, pp.703,715, May 1993 doi: 10.1109/5.220902

[2] Ryan, P.G.; Rawat, S.; Fuchs, W.K., "*TWO-STAGE FAULT LOCATION*," Test Conference, 1991, Proceedings., International , vol., no., pp.963,, 26-30 Oct 1991 doi: 10.1109/TEST.1991.519762

[3] J. Richman and K. R. Bowden, "*The Modem Fault Dictionary,*" Proc. Znt'l Test Conference, pp. 696-702,

[4] Cox, H.; Rajski, J., "*A method of fault analysis for test generation and fault diagnosis*," Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on , vol.7, no.7, pp.813,833, Jul 1988 doi: 10.1109/43.3952

[5] Shi-Yu Huang; Kwang-Ting Cheng; Kuang-Chien Chen; David Ihsin Cheng, "*Error Tracer: a fault simulation-based approach to design error diagnosis*," Test Conference, 1997. Proceedings., International , vol., no., pp.974,981, 1-6 Nov 1997 doi: 10.1109/TEST.1997.639713

[6] Ostendorff, S.; Wuttke, H. -D; Sachsse, J.; Kohler, S., "*A new approach for adaptive failure diagnostics based on emulation test*," Design, Automation & Test in Europe Conference & Exhibition (DATE), 2010 , vol., no., pp.327,330, 8-12 March 2010 doi: 10.1109/DATE.2010.5457183

[7] IEEE, "*IEEE Standard Test Access Port and Boundary-Scan Architecture*", United States of America, 2001

[8] S. Devadze, A. Jutman, I. Aleksejev, R. Ubar, "*Fast Extended Test Access via JTAG and FPGA's*", International Test Conference, Austin, Texas, November 1-6, 2009

[9] "*International Technology Roadmap for Semiconductors: Test and Test Equipment,"* Edition 2001.

[10] Shyue-Kung Lu; Jian-Long Chen; Cheng-Wen Wu; Ken-Feng Chang; Shi-Yu Huang, "*Combinational circuit fault diagnosis using logic emulation*," Circuits and Systems, 2003. ISCAS '03. Proceedings of the 2003 International Symposium on , vol.5, no., pp.V-549,V-552 vol.5, 25-28 May 2003 doi: 10.1109/ISCAS.2003.1206347

[11] Pomeranz, I.; Reddy, S.M., "*Location of stuck-at faults and bridging faults based on circuit partitioning*," Computers, IEEE Transactions on , vol.47, no.10, pp.1124,1135, Oct 1998 doi: 10.1109/12.729795

[12] A. Vahdat et al., "*Scalability and accuracy in a large-scale network emulator*," SIGOPS Oper. Syst. Rev., vol. 36, no. SI, pp. 271–284, 2002.

[13] S. Perarnau and G. Huard, "*krash: reproducible CPU load generation on many cores machines*," in IPDPS '10, 2010.

[14] T. Buchert, L. Nussbaum, and J. Gustedt, "*Accurate emulation of cpu performance*," in HeteroPar, 2010