Emphasis on the existence of intermittent faults in embedded systems

Julien Guilhemsang, Olivier Heron, Nicolas Ventroux
CEA, LIST

91191 Gif-sur-Yvette CEDEX, France ;
Email: julien.guilhemsang@cea.fr

Alain Giulieri
LEAT
Université de Nice-Sophia Antipolis, CNRS
250, rue Albert Einstein, 06560, Valbonne, France
Email: giulieri@polytech.unice.fr

***Abstract*—Future embedded systems are going to be more sensitive to hardware faults. In particular, intermittent faults are going to appear earlier in future technologies. Understanding the occurrence of faults and their impact on systems and applications can help improve the fault-tolerance of systems. However, there is no study on their effects in more complex digital circuits.**

We propose an experimental platform for accelerating and catching the occurrence of intermittent errors in complex digital circuits. We experimentally show that intermittent errors can appear during the lifetime of the circuit, very early before the wear-out period.

I. Introduction

The reliability of embedded systems is an important issue, and continuous advances in integration technologies have a negative impact on it. Devices are going to be more sensitive to transient errors, likewise intermittent and permanent errors are going to appear earlier in future technologies. Understanding the occurrence of faults and their impact on systems and applications can help improve the fault-tolerance of systems.

Transient and permanent faults have been studied for long time, so many studies are available and many techniques exist to tolerate these faults. On the contrary, intermittent errors are going to appear earlier and might be predominant in future systems, but there is no published paper on present technologies and particularly in embedded systems. Intermittent faults arbitrarily occur over time and can appear in bursts from few nano-seconds to several seconds [1]. Intermittent faults are mainly due to process variations and aging [2], [3] combined with dynamic variations of supply voltage and junction temperature [4].

Process variations bring up both random and systematic errors during the fabrication process. These errors have an impact on the circuit behavior and can induce distortion of the output signal for analog devices and variable delays for digital circuits. At transistor level, process variations have an impact on the threshold voltage V_T , the body factor γ and the current factor β [5]. There are two main types of process variations: Die-To-Die variations induce parameters deviations between the dies in a same wafer and With-In-Die variations induce parameters deviations between the transistors of a same die. In particular, With-In-Die variations result in different working frequencies for different parts of a same chip [6]–[8].

Moreover, degradation mechanisms such as Electromigration (EM), Time-Dependent Dielectric Breakdown (TDDB), Negative Bias Temperature Instability (NBTI), Hot Carrier Injection (HCI) and Stress Migration (SM) will decrease timing margins over time and can cause timing problems [5], [9], [10].

Under nominal operating conditions, voltage drop, HCI and EM are activated by a dynamic stress of wires, vias and transistors (switching), while NBTI and TDDB are activated by a constant stress (no switching). All of these physical phenomena are closely related to junction temperature. A high temperature level accelerates the activation of NBTI, EM, TDDB, while HCI is accelerated by a low temperature level.

Increasing temperature will accelerate aging, decrease timing margins and then promote the occurrence of intermittent faults [4]. In normal operating conditions, the device temperature and the consumption must be minimized to avoid intermittent faults. However, in our case the temperature stress is used to accelerate the occurrence intermittent errors.

After analyzing the State of the Art, we can say that: Intermittent fault problem is mainly discussed based on the knowledge of physics and experimental analysis, such as accelerated stress tests, performed on very simple circuits (single transistor or inverter), but there is no study on their effects on more complex digital circuits (processor cores, memories, peripherals). Regarding the behavior of NBTI, HCI, EM, TDDB and voltage drop described above, it is not trivial to determine which ones are the major detractors for intermittent faults according to activity stress (dynamic vs. constant switching stress) and operating conditions.

This paper makes the following contributions: We propose an experimental platform for accelerating and catching the occurrence of intermittent errors in complex digital circuits: processors, memories and peripherals. Besides, for a chosen technology and circuit design (Xilinx Virtex5FX in 65nm with a PPC 440), we show experimentally that intermittent errors can appear during the lifetime period of the circuit, very early before the wear-out period of the circuit. Section 2 describes the major aging failures that affect the processor lifetime and their activation condition depending on transistor activity. This section allows understanding why it is not easy to compare the lifetime of two systems, even if their activities are very

different. Sections 3 and 4 present the experimental platform and the case study. Section 5 shows the results and section 6 concludes the paper.

II. Aging induced failure mechanisms

Failures in chips are mainly related to assembly, mounting, handling and wafer processes [10]. During assembly process, the die is mounted in the package and different reliability issues can be addressed such as wire bounding reliability, ion migration, etc. Failures related to mounting process are mainly cracks in surface-mounted packages. Failures related to handling are electrostatic discharge, latch-up, etc.

In this paper, we only focus on failure related to wafer process. After analyzing the State of the Art [5], [9]–[16], we can say that: The five following aging-induced failure mechanisms become a major issue for processor lifetime: *Time-Dependent Dielectric Breakdown* (TDDB), *Negative Bias Temperature Instability* (NBTI), *Hot Carrier Injection* (HCI), *Electromigration* (EM) and *Stress Migration* (SM).

The use of thin gate oxides in deep submicron technologies combined with the non-ideal scaling of voltage supply increases electrical field stress and thus, accelerates the activation of NBTI, HCI and TDDB failure mechanisms [9]. In contrast to previous failure mechanisms that occur in the transistor, EM and SM failure mechanisms occur in vias, contacts and along long metal wires. Advances in circuit speed, device miniaturization and density increase the current density and thus accelerate failure activation.

Mean Time-To-Failure (MTTF) is a common reliability metric for semiconductor devices. A lower MTTF means earlier activation of a specific failure mechanism and thus, a less reliable device. For any failure mechanism, the metric generally depends on process, design, operating (stress) and environmental conditions. MTTF models for NBTI, HCI, TDDB, EM and SM are given in [9], [10], [12].

Considering the behavior of the different failure mechanisms, it is not easy to determine if a complex system like a processor will age faster if the processor is in a running state or in an idle state. Our experimental platform will provide answers to this problem.

Indeed, transistor's switches activate EM, SM and HCI, while idle states of transistors activate TDDB and NBTI.

On one hand, EM depends on the current density, which is maximum when transistors are switching. Transistor's switches activate SM because the current pulses induce a temperature pulse, which leads to the dilatation of lines. Then it causes mechanical constraints that can activate SM. In addition, HCI is a phenomenon that occurs when there is a current in the canal.

On other hand, leakage current in the dielectric of the gate cause TDDB, and electric field intensity in the dielectric gate influences NBTI.

An important remark concerns the exponential dependence of these failure mechanisms to the temperature. Apart from HCI, the MTTF of all other mechanisms decreases when the temperature increases. An over temperature stress of the chip would accelerate the activation of most failure mechanisms. The platform will use this result to accelerate their occurrence in the systems under test.

Same techniques are used to analyze the reliability of semiconductor circuits along the manufacturing process, from wafer-level process to packaging-level process, before being shipped to the user. One objective is to eliminate weak circuits that do not verify the expected reliability requirements. In this way, accelerated life tests, also known as IC burn-in, are used [17]. These tests consist in stressing the IC over its limits in order to evaluate the Failure In Time (FIT) that represents the occurrence frequency of each kind of failure. FIT values are generally computed with the aid of complex statistical formulas and results from accelerated life tests [10]. In that way, the tests target a very large set of circuits.

III. Presentation of the experimental platform

The objective of our platform is to observe and analyze the occurrence of errors in processor cores and chip failures during the lifetime of a system. This is done by stressing processors with a high temperature and by applying a set of stimuli on the different entries of each processor under test.

Our platform is divided into three main elements: the Circuits Under Test (CUT), the External Stress Manager (ESM) and the Internal Stress Manager (ISM). The CUT contains at least a processor, a memory and communication peripherals. The ESM accelerates aging failures by increasing the circuit temperature and/or the power supply. Finally, the ISM loads application sets in the memories, synchronizes their executions and collects errors. Figure 1a illustrates a typical organization of the platform.

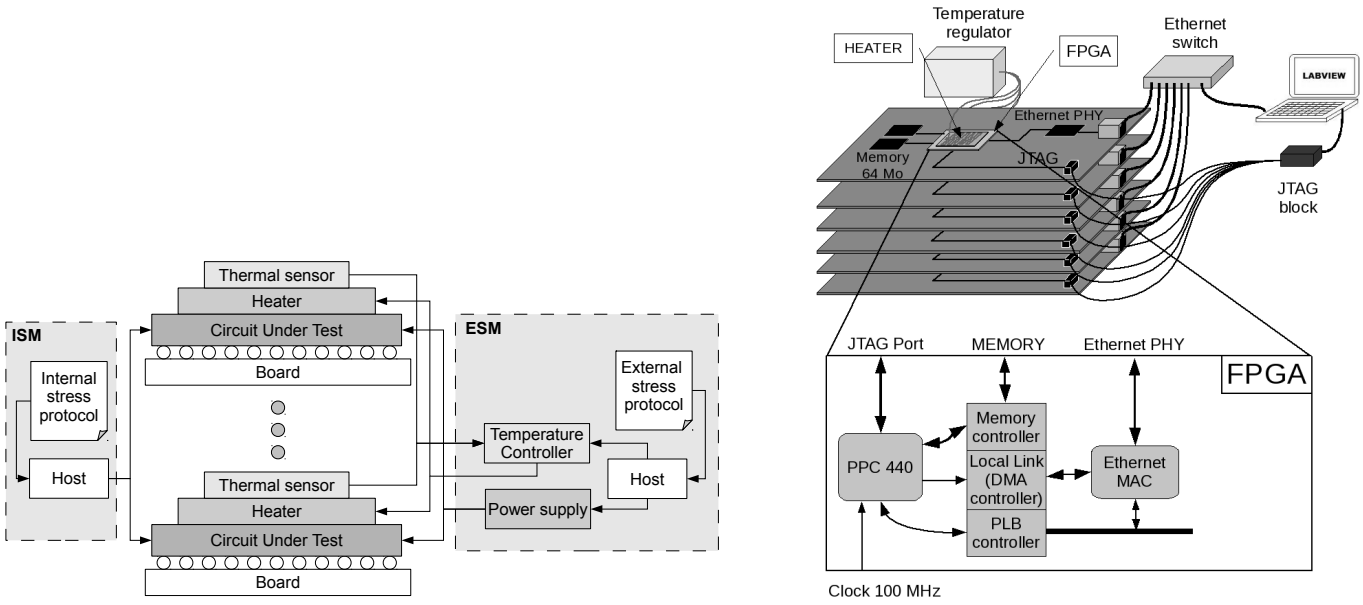
A. Circuit Under Test (CUT)

Each CUT is implemented into a board, in which I/O pads are connected to the following elements: clock generator, power supply, JTAG[1] port, peripheral transceiver and connectors. The JTAG port allows the connection between the host computer and the processor core, and will be used to download programs into the CUT. The CUT contains a R/W memory, a communication bus and a peripheral. The memory stores both program and data. The peripheral enables communications of data from the CUT to the external host computer.

In our case, the experiment is composed of six Avnet AESV5FXT boards [18]. The figure 1b shows a schematic of our platform.

Each board contains a Virtex5-FX FPGA and each FPGA contains a PowerPC 440 processor core implemented in hardware.

[1]IEEE 1149.1 standard entitled Standard Test Access Port and Boundary-Scan Architecture



(a) (b)

Fig. 1. (a) Platform functional diagram (b) Platform schematic

B. External stress management (ESM)

Aging analysis is done through temperature and power supply stresses. Both temperature and power management parts are controlled by the host computer through a specific bus (GPIB[2]). ESM controls the start of the experiment, sets the temperature and voltage values from a stress protocol file, records the output values and monitors the safety values. The recorded values are the instantaneous circuit voltage, current and temperature. In our case, the junction temperature cannot be monitored directly, so a temperature sensor is connected to the bottom side of the board under the CUT. For safety rules, if the circuit temperature or current grows above a predefined maximum threshold, the experiment is stopped and requests maintenance. The program used to control the ESM is implemented with the commercial tool LabView [19].

A stress protocol file describes the temperature and the voltage values over time. For stress temperature, the protocol lists the temperature value and the temperature pitch for each time. For voltage stress, the file lists the voltage values for each time.

In order to increase the junction temperature of the CUT over its limit, we use flexible heaters [20]. In our approach, the flexible heater is mounted to the top side of the circuit package and an acrylic pressure sensitive adhesive, that supports high temperature, links the heater to the package. To reach very hight temperature and to prevent variations of the temperature, the boards are placed in an oven at the temperature of 50°C.

The heater can reach the temperature of 200°C. The thermal resistance within recent devices becomes very low due to the use of efficient thermal sink and spreader. Thus, the temperature junction of the device is almost equal to the heater temperature. A PID-based controller (Proportional Integral Derivative) controls the heater current, as shown in figure 1a.

A programmable power generator controls the power supply. For allowing voltage stress, the power supply generator must be connected directly to the device pad and so, must bypass any voltage regulator on the board. A current monitor is serially inserted between the power supply and the board so that the instantaneous current variations can be monitored and recorded during the experiment.

In our case, we do not apply voltage stress.

C. Internal stress management (ISM)

The ISM controls the stimuli applied to the CUT, and especially in the processor core. We use a set of applications to create activity on the different parts of the processor, and to detect errors in these parts. Each application is executed several times, as shown in figure 2.

The ISM runs on the host computer with the LabView tool. It reads an internal stress protocol file that lists an application set to be run at different dates (see § IV-A). At the end of the execution of each application, it checks the recorded data received from the CUT peripheral to detect errors.

The following steps show how the applications are loaded in the CUT and how results are checked:

1) The first step configures the FPGA by downloading the bitstream. This step is done only once at the beginning of the experiment.
2) Then the application binary is downloaded in the external memory and the processor is started with JTAG

[2]IEEE-488 standard

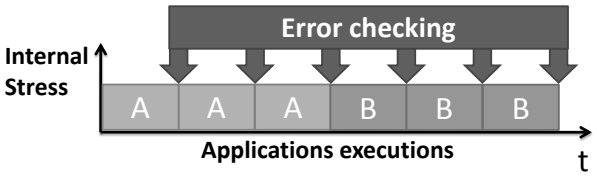


Fig. 2. Internal stress examples. A and B blocks represent one execution of two different applications. Errors are checked at the end of each execution of each application.

commands. This step is done when the ISM changes the application running on the processors (in figure 2 when changing from application *(A)* to application *(B)*).

3) The Ethernet communication between each CUT and the host computer is checked. If there is no connection, the processor is reset and step 2 repeated. If there is no connection the second time, the processor is considered as failed, experiment stops and maintenance is required.
4) Once the processor received the *start* command sent by the ISM, it executes one occurrence of the application (one occurrence of *(A)* in figure 2). In that way, the ISM can synchronize the different CUT.
5) The processor stores the results in the external memory and computes a signature (one word).
6) The host computer requests the results and the processor sends them three times to prevent transient communication errors.
7) The processor returns to the step 4 and waits for the *start* command before executing the application again.
8) Then the ISM compares the results with golden values. If a difference appears, the ISM logs it for post-analysis. The comparison verifies the data size and content. If no result is received by the ISM, the experiment is stopped and maintenance is requested by the ISM.
9) According to the internal stress protocol, if the application *(A)* do not ends, then the ISM sends the *start* command to each processor. Processors execute the same application *(A)* and go to step 5. Else, next application *(B)* is download in memory (step 2).

IV. Description of the experiment

The previous section described the different components of the platform. This part describes the different type of errors observed in the platform and the different stress protocols used to provide the results of section V.

A. Stress protocols

Two different stresses are considered in the platform, the internal stress and the external stress.

The internal stress protocol is composed by seven applications: *QuickSort* is a part of MiBench [21]. *DCT*, *Quant* and *Fir* are classical embedded applications used in signal processing, and *TestFunct*, *TestFunct2*, *TestFunct3* are Software-Based Self-tests (SBST) [22] programs based on ATPG techniques. Table I shows the characteristics of the applications. In particular, the table shows the differences in the number

TABLE I
Applications characteristics

N°	Applications	# load/store per s	# Instr. per s	Duration (s)
A	Qsort	8,567,619.96	25,157,627.08	9.28
B	FIR	5,113,756.02	12,600,864.70	7.60
C	Quant	3,982,256.49	10,202,307.42	11.48
D	Test_funct2	1,323,048.81	6,084,324.06	1.52
E	Test_funct	925,207.17	3,544,534.22	2.18
F	DCT	27,544.99	180,340.58	1.98
G	Test_funct3	110.17	9,531,901.79	67.84

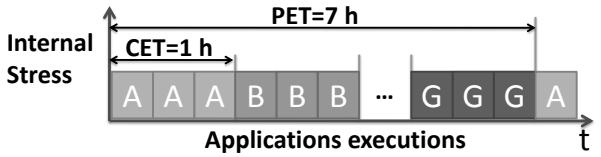


Fig. 3. Internal stress protocol. Each of the seven applications runs continuously during one hour (CET) each seven hours (PET). CET=Continuous Execution Time and PET=Periodic Execution Time.

of load/store per second and in the number of instructions per second for each application. These applications have very different behaviors. They are used to produce a representative activity of an embedded systems.

Each of the seven applications runs continuously during one hour, that is the Continuous Execution Time (CET). For example, the application *FunctionnalTest3* runs 53 times during one hour. The execution time of all applications is equal to seven hours. The applications are executed again until all boards fail. Thus, each application is executed periodically during one hour (CET) each seven hour, that is Periodic Execution Time (PET). The figure 3 shows this internal stress protocol.

Only temperature stress is used in our experiment. Four processors are heated at 170°C. Two processors remain at ambient temperature. No error should be observed on these two witness processors. They will produce golden values.

B. Observation mechanism

In the platform, errors can occur at any stage of the ISM protocol. We enumerate four types of errors:

- Bitstream loading error: this error can occur during the first part of the test, when the configuration file (bitstream) is loaded into the FPGA. In our experiments, this error happens only when a CUT failed.
- Program loading error: this error can occur during the second part of the test, when the program is loaded into the main memory of the board. As the previous type of error, we show this error only after the failure of a system.
- Communication error: this type of error occurs when the Labview program tries to communicate with the boards to get the computation results. The communication is done through an Ethernet port with the TCP/IP protocol. This error can be caused by an error in the processor, in the DMA controller, in the Ethernet controller or in the memory controller. Therefore, we cannot determine where exactly the error occurs. However, this error can be caused by an intermittent fault in the peripherals.

- Computation error: this error occurs if a fault appears when the processor executes different computations. In this case, produced signatures will differ from expected ones.

Computation errors are the most common errors observed in the experiments and are most representative of intermittent errors. They refer to errors in the processor or in the peripheral bus. In this paper we will focus only on this type of error.

Next section will present the behavior of intermittent computation errors observed during the experiments.

V. Results

The first experiment was done at 145°C during six months and did not display any computation errors on the different boards. Therefore, we can consider that errors shown after are not due to the hardware or software design. However, during this period the different systems were heated and then subject to accelerated aging.

The following results show errors on boards under an external stress about 170°C. Witness boards did not show any errors during the experiments.

A. Emphasis on intermittent errors

After the first experiment at 145°C, we heat the four boards at the temperature of 170°C. The first computing error was detected after 3.12 hours and all of the boards fail after 241.04 hours.

The figure 4 shows the total number of intermittent computing errors observed on all the boards all along the experiment. Each point represents the number of errors observed during a period of seven hours (one PET).

Firstly, these results show that it is possible to observe intermittent errors at system level and very early before the system fails. Thus, we can use methods to detect if a system is about to be faulty, and develop methods for the recovery. We present in [23] on-line techniques able to detect intermittent errors in multiprocessor embedded systems.

Secondly, the dotted line in the figure 4 shows the average of the number of errors detected during every period PET. We can see that the the number of intermittent errors increases before the failure. So, the monitoring of the number of intermittent errors can be used to predict the failure of the system.

B. Sensibility of the applications

The table II shows the total number of computation errors detected for each application for all the boards at the end of the experiment. We can see that the impact of intermittent errors depends on the applications. Besides, we note that applications with the greatest number of errors are also those that perform the greatest number of load/store per second (see table I). The number of load/store per second denotes the traffic with the external memory. So, applications with the greatest memory traffic seem to be more sensitive to intermittent errors. We can observe that errors appear most likely in long paths between processor and external memory.

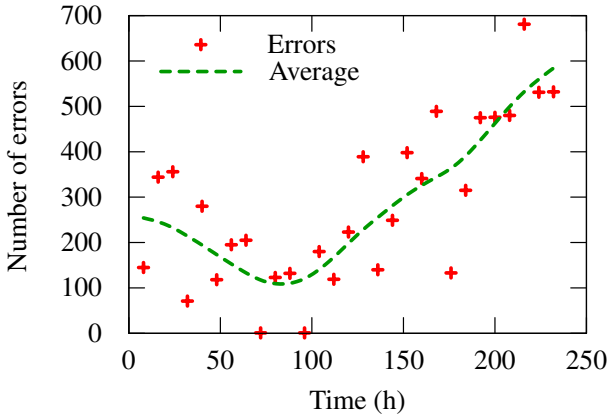


Fig. 4. Number of intermittent errors observed on all the devices at 170°C all along the experiment. Each point represents the total number of errors during one period PET about seven hours.

TABLE II
Number of error in function of applications.

N°	Applications	# Errors
A	Qsort	1936
B	FIR	1274
C	Quant	913
D	Test_funct2	1244
E	Test_funct	923
F	DCT	446
G	Test_funct3	189

C. Bursts of intermittent errors

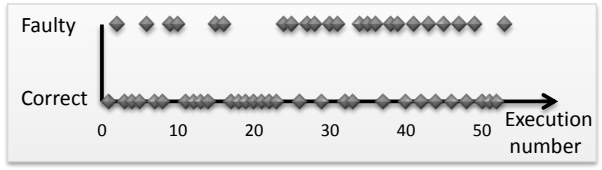
The figure 5a shows an execution trace for the program *FunctionnalTest3* in presence of intermittent errors during one CET. The fact that several errors occur during the execution of a same application, during a short time is called *burst*.

Each point corresponds to the result of each execution (as explained in figure 2). The result is either *Faulty* or *Correct*. We can define an intermittent error with two parameters, the frequency of appearance and the occurrence duration. In this case, the mean number of execution between two intermittent errors is about 2.1 executions and the mean duration of one intermittent errors is about 1.5 executions. These values show the intermittent behavior of computation errors.

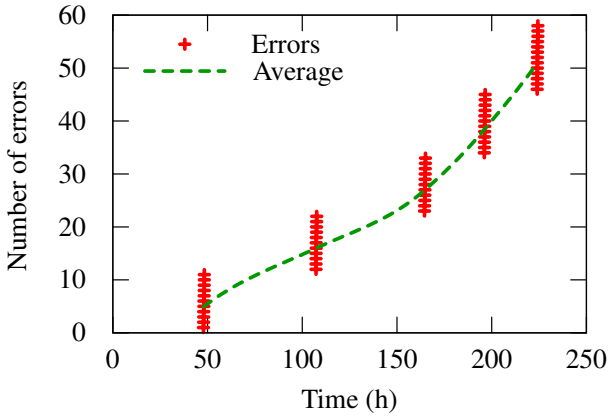
The figure 5b shows the accumulation of errors observed on one system for the program *FunctionnalTest3* all along the experiment.

Each program runs periodically one hour (CET) every seven hours (PET), thus each program runs 34 periods PET. The figure 5b shows only 5 periods with errors. This program is not erroneous for each of its executions period. Moreover, once the error is triggered, if no action is taken to stop it, the program continues to produce errors intermittently. According to our observations, all computation errors come in *burst*.

Here, the error is corrected by stopping the processor for loading the next program (from application *(X)* to *(Y)*). Therefore, this result confirms that suspension techniques [1] used to recover from intermittent errors are sufficient.



(a)



(b)

Fig. 5. Bursts of intermittent computation errors. (a) Execution trace for the program *FunctionnalTest3* in presence of a *burst* of intermittent errors, during a continuous execution of the application (CET) about one hour. Each point correspond to the result of the execution: either *Faulty* or *Correct* (b) Number of errors for one board for the program *FunctionnalTest3* all along the experiment. Each set of error corresponds to a *burst* of intermittent errors..

VI. Conclusion

For now, no study has observed intermittent errors on an embedded system in actual technology. We present an experimental platform able to stress an embedded system (Xilinx Virtex5FX in 65nm with a PPC 440) and generate intermittent faults. Our platform can be adapted to any digital test vehicles design, technology and chip assembly. For the same external stress, several experiments can be done in different conditions, like different internal activities.

Before the failure of a system appears, intermittent errors can be observed. Our results confirm the occurrence of intermittent faults and show that intermittent errors can be observed at system level. According to our observations, all intermittent computation errors appear in *burst*. Detection techniques must be adapted to this type of error [23]. Indeed, we show that if no action is taken, the error still occurs intermittently, but a simple shutdown/reboot of the processor seems to be sufficient for recovery.

References

[1] P. M. Wells, K. Chakraborty, and G. S. Sohi, "Adapting to intermittent faults in multicore systems," in *International conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*. New York, NY, USA: ACM, 2008, pp. 255–264.

[2] C. Constantinescu, "Impact of deep submicron technology on dependability of vlsi circuits," in *International Conference on Dependable Systems and Networks (DSN)*, Bethesda, MD, USA, 2002, pp. 205–209.

[3] S. Borkar *et al.*, "Parameter variations and impact on circuits and microarchitecture," in *Design Automation Conference (DAC)*. New York, NY, USA: ACM, 2003, pp. 338–342.

[4] C. Constantinescu, "Impact of intermittent faults on nanocomputing devices," *Proc. IEEE/IFIP DSN (Supplemental Volume), Edinburgh, UK*, pp. 238–241, 2007.

[5] G. Gielen *et al.*, "Emerging yield and reliability challenges in nanometer cmos technologies," *Design, Automation and Test in Europe (DATE)*, pp. 1322–1327, 2008.

[6] S. Duvall, "Statistical circuit modeling and optimization," in *IEEE International Workshop on Statistical Metrology (IWSM)*, 2000, pp. 56–63.

[7] K. Bowman, S. Duvall, and J. Meindl, "Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration," *IEEE Journal of Solid-State Circuits*, vol. 37, no. 2, pp. 183 –190, feb 2002.

[8] K. Bowman, A. Alameldeen, S. Srinivasan, and C. Wilkerson, "Impact of die-to-die and within-die parameter variations on the clock frequency and throughput of multi-core processors," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 17, no. 12, pp. 1679 –1690, dec. 2009.

[9] J. Srinivasan, S. Adve, P. Bose, and J. Rivers, "The impact of technology scaling on lifetime reliability," in *DSN*, Florence, Italy, 2004, pp. 177–186.

[10] Renesas Technology, "Semiconductor reliability handbook," Renesas Technology, Tech. Rep. Rev 1.01, Nov. 2008.

[11] V. Reddy *et al.*, "Impact of negative bias temperature instability on digital circuit reliability," in *Reliability Physics Symposium Proceedings, 2002. 40th Annual*, 2002, pp. 248–254.

[12] Jedec Publication, "Failure mechanisms and models for semiconductor devices," Jedec, Tech. Rep. JEP122C, Mar. 2003.

[13] S. Lin *et al.*, "Impact of off-state leakage current on electromigration design rules for nanometer scale cmos technologies," in *2004 IEEE International Reliability Physics Symposium Proceedings, 2004. 42nd Annual*, 2004, pp. 74–78.

[14] M. White *et al.*, "Product Reliability Trends, Derating Considerations and Failure Mechanisms with Scaled CMOS," in *2006 IEEE International Integrated Reliability Workshop Final Report*, 2006, pp. 156–159.

[15] B. Greskamp, S. Sarangi, and J. Torrellas, "Threshold voltage variation effects on aging-related hard failure rates," in *IEEE International Symposium on Circuits and Systems, 2007. ISCAS 2007*, 2007, pp. 1261–1264.

[16] W. Goes and T. Grasser, "Charging and discharging of oxide defects in reliability issues," in *IEEE IIRW*, 2007, pp. pp. 27–32.

[17] Jedec Publication, "Foundry process qualification guidelines," Jedec, Tech. Rep. JP001.01, May 2004.

[18] AVNET, "AES-V5FXT-EVL30-G Evaluation Kit." [Online]. Available: http://www.em.avnet.com

[19] National Instruments, "Labview 7.1." [Online]. Available: http://ni.com/labview

[20] Minco, "Flexible heaters design guide." [Online]. Available: http://www.minco.com/

[21] M. Guthaus *et al.*, "MiBench: A free, commercially representative embedded benchmark suite," in *IEEE International Workshop on Workload Characterization*, 2001, pp. 3–14.

[22] N. Kranitis, A. Paschalis, D. Gizopoulos, and G. Xenoulis, "Software-based self-testing of embedded processors," *Computers, IEEE Transactions on*, vol. 54, no. 4, pp. 461–475, April 2005.

[23] O. Heron, J. Guilhemsang, N. Ventroux, and A. Giulieri, "Analysis of On-Line Self-Testing Policies for Real-Time Embedded Multiprocessors in DSM Technologies," in *IEEE International On-Line Testing Symposium (IOLTS)*, Corfu Island, Greece, 5-7 July 2010.