Globally Optimized Robust Systems to Overcome Scaled CMOS Reliability Challenges

Subhasish Mitra
Departments of Electrical Engineering and Computer Science
Stanford University, Stanford, CA

Abstract

Future system design methodologies must accept the fact that the underlying hardware will be imperfect, and enable design of robust systems that are resilient to hardware imperfections. Three techniques that can enable a sea change in robust system design are: 1. Built-In Soft Error Resilience (BISER), 2. Circuit Failure Prediction, and 3. Concurrent Autonomous self-test using Stored Patterns (CASP). Global optimization across multiple abstraction layers is essential for cost-effective robust system design using these techniques.

1. Introduction

Most digital systems, except high-end mainframes and safety-critical systems, assume error-free hardware, i.e., all logic transistors and interconnects must always operate correctly during useful lifetime of such systems. For future process technologies, such a design paradigm is either infeasible due to fundamental technology limitations, or is very expensive. One option is to accept that the underlying hardware will be imperfect, and design robust systems that are resilient to hardware imperfections. To make such systems practical, we must overcome the following challenges:

1. The most significant challenge is to achieve acceptable levels of robustness at minimum system-level costs (energy, power, performance, area, design and validation costs).

2. Classical redundancy assumptions such as the presence of a single faulty component or occurrences of independent failures may not be adequate.

3. Validation of robust systems can be very challenging.

Luckily, several new opportunities for cost-effective solutions are created by changing cost constraints in future technologies, and new killer applications. Examples include:

1. Dominance of long interconnects in VLSI designs, making it possible to add local transistors and local interconnects without significantly impacting chip area.

2. Availability of high-density non-volatile storage.

3. Wide proliferation of multi-core architectures.

4. Resilience of emerging workloads such as Recognition, Mining and Synthesis (RMS) [Dubey 05] to data errors.

This paper presents an overview of techniques that can serve as foundations for designing robust systems, and discusses global optimization opportunities across abstraction layers for cost-effective implementations of these techniques.

2. Message of this Paper

Major scaled CMOS reliability challenges are (Fig. 2.1):

- Mechanisms such as Negative Bias Temperature Instability (NBTI) aging [Agostinelli 05a, Reddy 02], that were largely benign in the past, are becoming important. The worry is that traditional speed margins to overcome such wearout problems may become too expensive.
- Burn-in for infant mortality is getting difficult [Borkar 05, Carulli 05, Kundu 04, Nigh 00, Van Horn 05]. Major burn-in challenges include power dissipation, cost, and serious concern about potentially reduced effectiveness of burn-in in future. Other infant mortality screens, e.g., I_{ddq} and VLV testing, are also running out of steam.
- Complete validation and testing are difficult and expensive due to design complexity and process variations.
- Designs are getting increasingly susceptible to intermittent and transient errors, e.g., radiation-induced soft errors, and erratic changes in V_{ccmin} (also referred to as erratic bit errors) [Agostinelli 05b, Baumann 02, Mitra 05].

An effective way of overcoming scaled CMOS reliability challenges is to design robust systems using a combination of the following techniques:

1. Built-In Soft Error Resilience (BISER) (Sec. 3) for correcting soft errors and erratic bit errors.

2. Circuit failure prediction (Sec. 4), combined with an on-line self-test technique called CASP (Sec. 5), to overcome infant mortality and wearout challenges.

The same underlying error protection circuitry can be efficiently time-multiplexed to support all these techniques cost-effectively. Such an approach creates unique opportunities for optimizing robust systems across abstraction layers – circuit, architecture, runtime and application.

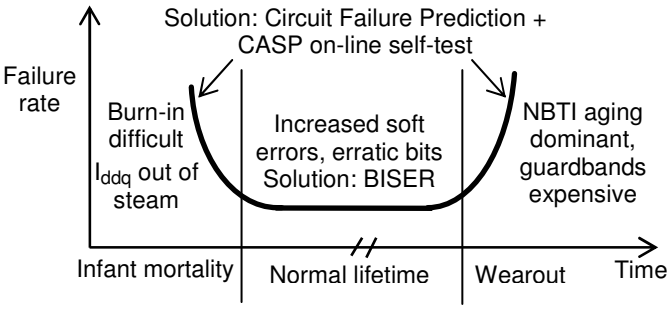

Figure 2.1. Scaled CMOS reliability challenges.

3. Built-In Soft Error Resilience (BISER)

Efficient coding techniques exist for correcting soft errors in SRAMs. However, correction of soft errors in latches, flip-flops and combinational logic (often referred to as *logic soft errors*) is challenging.

What is BISER?

The *Built-In Soft Error Resilience (BISER)* technique can correct radiation-induced soft errors in latches, flip-flops and combinational logic [Mitra 05, Mitra 06, Zhang 06]. Figure 3.1 illustrates BISER for correcting soft errors in latches. During normal operation, when Clock = 1, the latch input is strongly driven by the combinational logic and the latch is not susceptible to soft errors. (This follows from the concept of Timing Vulnerability Factor or timing derating [Mitra 05]). When Clock = 0, C-OUT already has the correct value – any soft error in either latch results in a situation where the logic value on A does not agree with B. As a result, the error does not propagate to C-OUT, and the correct logic value is held at C-OUT by the keeper.

978-3-9810801-3-1/DATE08 © 2008 EDAA

Soft Error Rate Reduction using BISER

Extensive simulations using experimentally validated sub-90nm simulation tools [Nguyen 03] demonstrate that the BISER latch of Fig. 3.1 achieves more than 20-fold reduction in soft error rate compared to an unprotected latch. Note that, a soft error in the keeper does not have a major effect because C-OUT is strongly driven by the latch contents (assuming single error). Design of BISER flip-flops is discussed in [Mitra 05, Zhang 06]. The BISER idea can be extended to reduce combinational logic soft error rate by 12-64 times [Mitra 06]. As demonstrated in [Relangi 07], BISER latches can also correct erratic bit errors discussed in [Agostinelli 05b].

Recently, concerns have been raised about multiple bit upsets (MBUs) caused by single radiation events [Seifert 07]. For SRAMs, interleaving and low-cost coding techniques are sufficient for MBU protection. Cost-effective protection of logic from MBUs requires detailed characterization of MBUs and their effects on BISER structures. MBU analysis in [Seifert 07] indicates the superiority of BISER in correcting MBUs compared to circuit techniques such as [Calin 96].

BISER Cost Optimization

A BISER latch or flip-flop can be included as a standard cell in the technology library. The speed penalty introduced by the BISER cell is very small (~ 1%) [Zhang 06]. However, cell-level energy / power and area costs can be significant. Hence, global optimization is essential for understanding and optimizing chip-level costs of BISER.

Place-and-route results on several designs from industry and from www.opencores.org indicate that the chip-level area impact of BISER is very small – between 0 to 3%. This is because BISER adds local transistors and interconnects in latches and flip-flops, while chip sizes are dominated by global interconnects and memory blocks. The area impact of BISER can be further reduced by reusing on-chip scan resources for post-Silicon validation, production testing, and BISER soft error correction [Mitra 05].

Global optimization for minimizing the energy / power impact of BISER requires tight coupling between circuit design, architecture, and application. Fault injection simulations on an Alpha-like microprocessor show that BISER improves chip-level soft error rate by 10 times over an unprotected design with only 7-10% chip-level power penalty [Zhang 06]. This is because not all soft errors are equally important from system perspective. This creates BISER insertion opportunities for maximized protection at minimum cost. A formal verification technique for optimized BISER insertion results in 5-fold reduction in the chip-level power impact of BISER for a Spacewire communication protocol design (compared to a version of the same design where all flip-flops are protected using BISER) [Seshia 07].

Additional opportunities exist for application-aware optimization to reduce the energy / power impact of BISER. For example, BISER can be configured, during system operation, to operate in one of two modes – an *error resilient mode* in which BISER protection is turned on, and an *economy mode* in which BISER protection is turned off in order to reduce energy / power costs. As illustrated in [Zhang 06], such **configurability** is efficient and practical from system design perspective. Such configurability can help minimize system-level energy / power impact of BISER by turning on the error-resilient mode only for critical parts of applications. Furthermore, it may be possible to combine BISER with low-cost application-specific checks, e.g., assertions and Algorithm-Based Fault Tolerance (ABFT) checks [Huang 84], for globally-optimized robust systems.

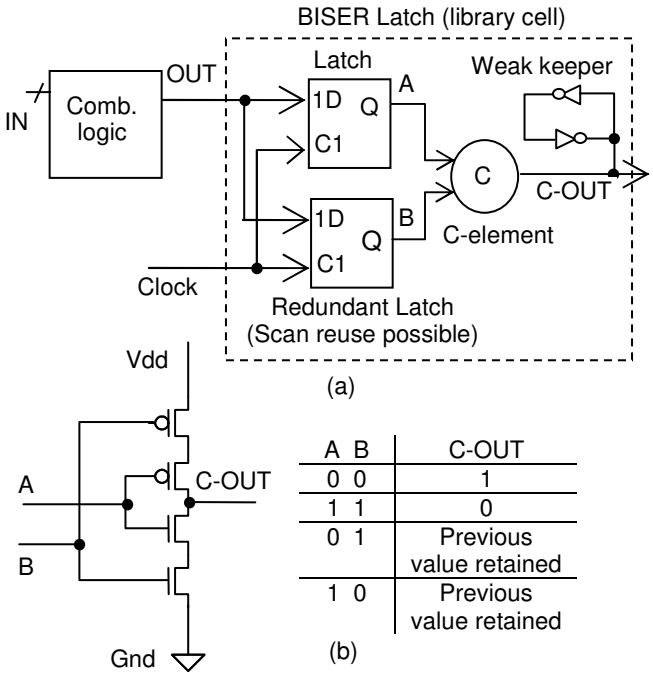

Figure 3.1. Built-in Soft Error Resilience (BISER) latch error correction. (a) Overall BISER latch. (b) C-element.

BISER vs. Traditional Techniques

Table 3.1 presents a comparative analysis of BISER vs. several soft error protection techniques available in the literature, and demonstrates the cost-effectiveness of BISER. Metrics used in Table 3.1 are explained below.

1. Silent Data Corruption (SDC) reduction refers to the reduction of undetected errors.

2. Detected and Uncorrected Error (DUE) reduction: SDC reduction techniques based on error detection can significantly increase DUEs. Error recovery triggered by such DUEs can result in expensive system downtime.

3. Chip-level energy penalty, speed penalty and die size increase are self-explanatory. Chip-level energy penalties for configurable error protection techniques are reported separately for the error resilient mode (when soft error protection is turned on) and the economy mode (when soft error protection is turned off).

4. Recovery mechanism design and validation effort: Designing proper error recovery mechanisms and validating them are non-trivial tasks and can incur high costs.

5. As discussed earlier, configurability of error protection techniques enables significant benefits such as global optimization over a wide range of applications.

6. Applicability: Since a wide range of future designs must be protected from soft errors, it is desirable to have protection techniques with general applicability.

Table 3.1. BISER vs. existing soft error protection techniques.

Metrics	BISER [Mitra 05, 06, Zhang 06]	Hardened latches and flip-flops [Calin 96]	Hardware duplication [Bartlett 04]	Time redundancy [Mukherjee 02, Oh 02, Saxena 00]
SDC reduction	Latch: 20X Comb. Logic: 12-64X	Latch: 20X Comb. Logic: None	Almost all	Almost all
DUE reduction	Latch: 20X Comb. Logic: 12-64X	Latch: 20X Comb. Logic: None	Increased DUEs	Increased DUEs
Chip-level energy penalty (error resilient mode)	6-10%	12 - 18%	40 – 100%	> 40% (published penalties unavailable)
Chip-level energy penalty (economy mode)	1.5%	12 – 18%	Small	Unclear
Speed penalty	Latch correction: ~ 1% Comb. logic correction: ~ 5%	~ 1%	Small	~40% or more
Die size increase	Not expected	Not expected	> 40%	Unclear
Recovery design & validation efforts	None	None	Significant	Significant
Configurability	Yes	No	Yes	Yes
Applicability	General	General	General	Processors
Latch, flip-flop and comb. logic protection	Both	Latches & flip-flops only	Both	Both

7. Latch, flip-flop and combinational logic protection: It is desirable for protection techniques to address soft errors in latches, flip-flops and combinational logic. Separate techniques for sequential elements and combinational logic introduce additional penalties.

4. Circuit Failure Prediction

Circuit failure prediction predicts the occurrence of a circuit failure **before** errors actually appear in system data and states [Agarwal 07]. This is in contrast to traditional error detection where a failure is detected **after** errors appear in system data and states. The basic principle is to collect information about the evolution of various system parameters over time, and to analyze the collected data to predict failures. The information is collected concurrently during normal system operation using special circuits called *failure prediction sensors*, and/or during periodic on-line self-test. System parameters that we focus on in this section involve timing characteristics of logic signals because they are effective in predicting scaled-CMOS reliability failures. Failure prediction sensors, used for data collection during normal system operation, are different from traditional process monitors such as ring oscillators (details later in this section).

Which Failures can be Predicted?

Not all circuit failures can be predicted, e.g., radiation-induced soft errors or erratic bit errors. Such errors can be corrected using the BISER technique in Sec. 3. Circuit failure prediction is ideal for infant mortality (e.g., weak gate-oxide due to random defects) and wearout (e.g., NBTI-induced aging). This is because of the gradual nature of degradation associated with these mechanisms, as demonstrated in [Agostinelli 05a] for NBTI-induced aging, and in [Chen 08] for gate-oxide infant mortality.

Circuit Failure Prediction vs. Error Detection

Table 4.1 compares and contrasts various aspects of circuit failure prediction vs. traditional error detection. Major benefits of circuit failure prediction arise from the fact that it enables a system to initiate corrective measures before system data and states actually get corrupted by errors – this prevents silent errors, error propagation and error detection latency problems associated with error detection.

Table 4.1. Circuit failure prediction vs. error detection.

Circuit failure prediction	Error detection
Before errors appear	After errors appear
No corrupt data or states	Corrupt data & states
No error	High error rates difficult
Self-diagnostics possible	Limited diagnostics
Incorrect prediction can be problematic	Insufficient coverage can be problematic
Not all failures predictable	General applicability
Both can be efficiently combined	

Circuit Failure Prediction for NBTI-Induced Transistor Aging: Why and How?

NBTI-induced aging slows down PMOS transistors over time. As a result, the speed of a chip can degrade over time which can result in delay faults. This problem became significant at 90nm technology node, and is expected to get worse in future generations. Chip-level effects of NBTI can be complex. For example, delays of different paths on the same chip or same paths on different chips can degrade very differently depending on the workload. This can lead to changes in timing-critical paths over time. A circuit path which is not timing-critical at manufacture time may become timing-critical later.

The current industrial practice to cope with NBTI aging is to slow down the clock frequency, i.e., add timing margin during design / test based on the worst degradation the transistors might suffer during lifetime due to worst-case temperature, voltage and workload. This is called *guardbanding*. However, the actual aging of a chip depends on its field usage – temperature, voltage, and workload. With worst-case guardbanding, all chips may suffer significant reduction in speed although most of them may not be

maximally stressed in the field. The statistical component of NBTI-induced aging makes worst-case guardbanding more difficult, i.e., similar nominal devices under similar environmental and workload conditions may age at different rates. Hence, worst-case guardbanding must also account for standard deviations in aging distributions for high confidence. This further limits the use of worst-case guardbands.

Figure 4.1, taken from [Agarwal 07], illustrates one application of circuit failure prediction to overcome NBTI aging challenges. This approach enables designs with close to best-case performance. Instead of introducing a worst-case guardband over the entire lifetime of a design (e.g., 7 to 10 years for enterprise systems), the system starts off with a small worst-case timing guardband that guarantees correct circuit operation under worst aging over a short period of time, e.g., 15 days. Trade-offs associated with the choice of this initial period ranging from 1 day to 1 month are discussed in [Agarwal 07]. We refer to this timing guardband as the *guardband interval* T_g (Fig. 4.1). During these 15 days, the system collects lots of data about relative aging of various circuit paths. This data is collected in two ways:

1. Concurrently during system operation using special failure prediction sensors (referred to as *aging sensors*);
2. Special on-line self tests that are scheduled periodically.

At the end of 15 days, the collected data is analyzed to check whether there has been enough aging that requires adjustment of any system parameter. If so, the system adapts itself according to its operation history by using one or more of several self-healing options (details later), and continues its operation. This circuit failure prediction approach enables up to 4-fold reduction in PMOS aging guardband [Agarwal 07].

The biggest challenge in implementing such a circuit failure prediction technique is the collection of data necessary for high-confidence aging estimation and prediction.

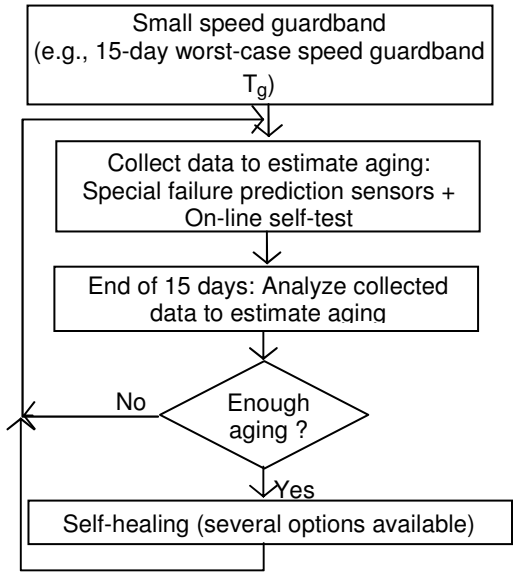

Figure 4.1. Circuit Failure prediction for NBTI-induced aging.

Circuit Failure Prediction Implementation for Aging

As discussed before, data collection for high-confidence aging estimation and prediction can be implemented in two ways:

1. Concurrently during system operation using special aging sensors;
2. Thorough on-line self-tests assisted by aging sensors and / or clock control.

For NBTI-induced PMOS aging prediction, on-chip ring oscillators and temperature sensors are inadequate [Agarwal 07]. This is because the amount of aging strongly depends on the workload of a chip. The workload of an on-chip ring oscillator can be very different from the actual workload experienced by various circuit blocks in a chip.

Figure 4.2a shows the working principle of a new aging-resistant failure prediction sensor design for a system with rising edge-triggered flip-flops [Agarwal 07]. This technique modifies a standard flip-flop by inserting a "monitoring" circuit block which detects 'significant' shifts in the delay of the combinational logic whose output is connected to the data input of that flip-flop (Fig. 4.2b). The monitoring circuit block is based on the concept of *stability checking* through detection of signal transitions at the combinational logic output during the guardband interval T_g in Fig. 4.1. Signal transitions during T_g imply that one or more paths in the combinational logic have aged enough to creep into the guardband interval. Note that, the flip-flop still continues to capture correct values unlike traditional error detection. The delay element in Fig. 4.2b creates the interval T_g for stability checking, and the output latch stores stability checking results.

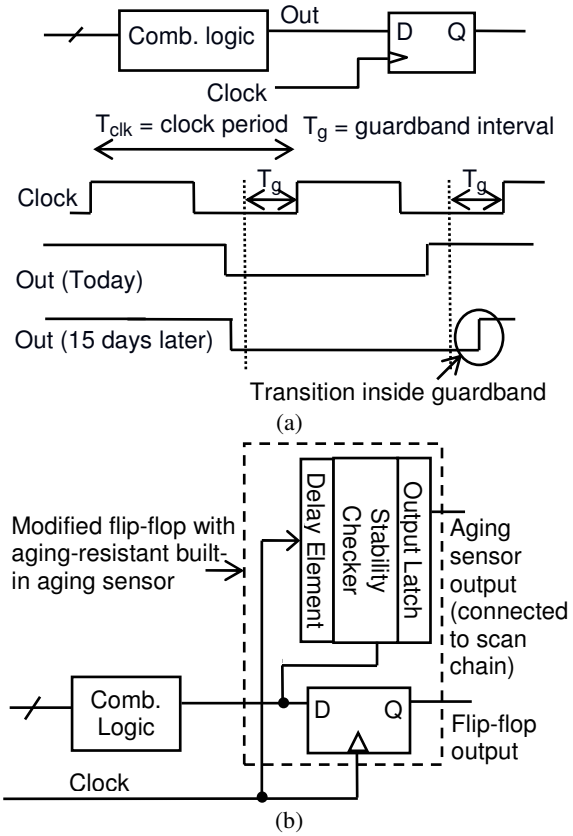

Figure 4.2. Flip-flop with built-in aging sensor. (a) Working principle. (b) Block diagram.

Special design considerations to ensure the aging-resistant property of the aging sensor in Fig. 4.2b are described in [Agarwal 07]. Aging resistance is achieved by infrequently

turning on the aging sensor, e.g., 1-5% of the time, by blocking signal transitions in the delay element most of the time. This also helps in minimizing the chip-level energy / power impact of the aging sensor, but requires careful design of the delay element (details in [Agarwal 07]). As shown in Fig. 4.2b, the information collected by aging sensors can be scanned out using on-chip scan chains eliminating the need for additional global interconnects. Such scan chain reuse is possible because of the failure prediction (rather than error detection) nature of our approach. The operation of the flip-flop with built-in aging sensor (Fig. 4.2b) has been validated using a 90nm test chip prototype.

As described in [Zhang 07], it is possible to efficiently design a special flip-flop standard cell which can be configured, during system operation, into one of the following modes: 1. BISER flip-flop for soft error correction; 2. flip-flop with built-in aging sensor; and, 3. scan flip-flop for test and debug. Such a flip-flop provides the flexibility to efficiently time-multiplex the same underlying error protection circuitry for both soft error correction and circuit failure prediction.

Aging data collection concurrently during system operation may not be adequate for the following reasons:

1. As discussed before, aging sensors must be turned on infrequently to minimize their aging. Hence, thorough coverage may not be guaranteed from the applications alone.

2. Path delay coverage of system applications may not be known *a priori*.

Hence, accurate aging estimation requires on-line self-test to thoroughly exercise a circuit using test patterns targeting circuit paths with slacks less than or equal to the maximum delay degradation a circuit may suffer. The CASP technique, described in Sec. 5, provides a cost-effective way of applying extremely thorough on-line self tests. Aging sensors are useful but not necessary during on-line self-test. Special test clock control techniques, e.g., those described in [Iyengar 06] for at-speed structural test, can be employed to perform aging estimation during on-line self-test without requiring special aging sensors.

Circuit Failure Prediction Costs and Optimization Opportunities

Detailed simulations using 90nm and 65nm technologies demonstrate that chip-level performance and power costs of the aging sensor of Fig. 4.2b are very small, e.g., < 1% performance cost, and < 0.4% chip-level power cost for an OpenRISC processor core. The chip-level power cost is very small because aging sensors are turned on infrequently to minimize their aging. Similar to BISER, a flip-flop standard cell with built-in aging sensor adds local transistors and interconnects, and does not require any global interconnect. Hence, the chip-level area impact of such a flip-flop cell is very small. NBTI modeling tools can be used to further minimize the chip-level area cost by instantiating such a cell only at selected flip-flops. For cost-sensitive designs, aging sensors can be eliminated by performing aging estimation only during on-line self-test by reusing test clock control techniques for at-speed testing, e.g., [Iyengard 06].

The design flow impact of using flip-flops with built-in aging sensors is minimal because no additional hold time constraints are imposed. However, it is necessary to verify the closed-loop behavior of the overall circuit failure prediction and self-healing approach.

Self-healing Options for NBTI-induced Aging

Self-healing approaches for NBTI-induced aging include:

1. Variable clock frequency: The clock frequency may be reduced over time depending on the actual aging of a chip.

2. Adaptive body bias: Since the primary effect of NBTI is to increase the magnitude of threshold voltage of a PMOS transistor, forward biasing the source-body junction can compensate for the change [Tschanz 07].

3. Adaptive power supply: Power supply voltage may be increased to compensate for aging. Care must be taken to ensure that the system does not enter a runaway situation since increased supply voltage accelerates aging.

4. Spare cores: For designs with several cores, it may be possible to use spare cores for self-adaptation purposes.

5. Healing: It may be possible to utilize the recovery effects of NBTI [Agostinelli 05a] for self-healing purposes.

5. CASP Technique for On-line Self-Test

On-line self-test is necessary for a wide variety of reasons – circuit failure prediction (explained in Sec. 4), error detection based on periodic, time-triggered or event-triggered self-testing, and diagnostics for self-repair. Effective on-line testing techniques must satisfy the following requirements:

1. High test coverage.
2. Minimal system-level performance impact.
3. No system level downtime visible to the end-user.
4. Minimal hardware cost.
5. No major changes in design and validation flows.

The CASP technique for on-line self-test, introduced in [Li 08], satisfies these requirements.

What is CASP?

CASP is an acronym for **C**oncurrent **A**utonomous chip self-test using **S**tored test **P**atterns. It is a special kind of self-test where a system tests itself concurrently during normal operation without any downtime visible to the **end-user.** Two fundamental ideas that enable CASP are:

1. Storage of very thorough test patterns with quantified test coverage in non-volatile memory such as flash memory or hard disks. These test patterns include the entire suite of high-quality scan and functional tests applied during production. Extremely thorough tests, some of which may not be applied during production for cost reasons, can also be included. Moreover, CASP test patterns can be changed (e.g., through patches) according to application requirements and failure characteristics after a system is deployed in the field. Availability of high-density and low-cost non-volatile memory, and extensive use of Design-for-Testability (DFT), test compression, at-speed testing and Boundary Scan are major technology drivers that favor this idea.

2. Architectural and system-level support for autonomous testing of one or more cores in a multi-core system, concurrently during normal system operation, while the rest of the system continues to operate normally. As a result, it is not necessary to bring down the entire system. This is possible because of the wide proliferation of multi-core architectures.

CASP is applicable for a wide range of applications including microprocessors, graphics and networking systems.

As detailed in [Li 08], CASP overcomes limitations of existing self-test techniques such as Logic Built-In-Self-Test (BIST), periodic functional testing, and roving emulation [Breuer 86], and enables flexible and high-quality on-line self-test at low cost without significant impact on design flow.

CASP Benefits, Costs and Optimization Opportunities

Implementation of CASP in the open-source OpenSPARC T1 processor, with 8 processor cores supporting 32 threads, demonstrates its effectiveness and practicality [Li 08]. The CASP test controller incurs less than 0.01% chip-level area overhead, and hardware support for on-line architectural isolation incurs less than 4% area overhead. Extremely thorough (automatically generated) test patterns, with 99.5% stuck-at coverage, 96% transition coverage and 93.5% True Time path delay coverage, require 60 MBytes of non-volatile storage. CASP test time is 1.2 seconds for a processor core (using off-chip flash memory), and is dominated by test data transfer time between off-chip storage and on-chip scan chains. Wide availability of low-cost and high-density non-volatile storage (e.g., flash memory) in most systems (not necessarily only processors) makes CASP effective and practical.

Unique opportunities exist for global optimization of CASP for large-scale systems. Examples include optimization of CASP isolation support across virtualization and hardware abstraction layers to minimize area impact, and virtualization support to enable application-aware CASP test scheduling to minimize system-level performance, power and test time impact. CASP also creates several new opportunities beyond on-line self-test, e.g., effective post-Silicon system validation and characterization.

6. Conclusions

BISER-based soft error correction, circuit failure prediction, and CASP-based on-line self-test enable a sea change in the design of robust systems that can effectively overcome scaled-CMOS reliability challenges. These techniques create several unique opportunities for designing cost-effective robust systems that are globally optimized across multiple abstraction layers, i.e., circuit, architecture, runtime and application.

Acknowledgment

This research was supported in part by the FCRP Gigascale Systems Research Center (GSRC), FCRP Center for Circuit and System Solutions (C2S2), Semiconductor Research Corporation, and the National Science Foundation CAREER Award. Contributions of Stanford Robust Systems Group students and collaborators are gratefully acknowledged.

References

[Agarwal 07] Agarwal, M., *et al.*, "Circuit Failure Prediction and Its Application to Transistor Aging," *Proc. VLSI Test Symp.*, pp. 277-286, 2007.

[Agostinelli 05a] Agostinelli, M., *et al.*, "Random charge effects for PMOS NBTI in ultra-small gate area devices," *Proc. Intl. Reliability Physics Symp.*, pp. 529–532, 2005.

[Agostinelli 05b] Agostinelli, M., *et al*., "Erratic Fluctuations of SRAM Cache Vmin at the 90nm Process Technology Node," *Proc. Intl. Electron Devices Meeting*, pp. 655-658, 2005.

[Bartlett 04] Bartlett, W., and L. Spainhower, "Commercial Fault Tolerance: A Tale of Two Systems," *IEEE Trans. Dependable and Secure Computing*, Vol. 1, No. 1, pp. 87-96, 2004.

[Baumann 02] Baumann, R., "The Impact of Technology Scaling on Soft Error Rate Performance and Limits to the Efficacy of Error Correction," *Intl. Electron Devices Meeting*, pp. 329 – 332, 2002.

[Borkar 05] Borkar, S.Y., "Designing Reliable Systems from Unreliable Components: The Challenges of Transistor Variability and Degradation," *IEEE Micro*, pp. 10-16, Nov.-Dec. 2005.

[Breuer 86] Breuer, M., and A. Ismaeel, "Roving Emulation as a Fault Detection Mechanism," *IEEE Trans. Comput.*, 1986.

[Calin 96] Calin, T., M. Nicolaidis, and R. Velaco, "Upset Hardened Memory Design for Submicron CMOS Technology," *IEEE Trans. Nucl. Sci.,* Vol. 43, pp. 2874-2878, Dec. 1996.

[Carulli 05] Carulli, J.M., and T.J. Anderson, "Test Connections – Tying Application to Process," *Proc. Intl. Test Conf.*, 2005.

[Chen 08] Chen, T.W., *et al.*, "Gate-Oxide Early Life Failure Prediction," *Proc. VLSI Test Symp.*, 2008.

[Dubey 05] Dubey, P., "Recognition, Mining and Synthesis Moves Computers to the Era of Tera", *Technology at Intel*, Feb. 2005.

[Huang 84] Huang, K.H., and J.A. Abraham, "Algorithm Based Fault Tolerance for Matrix Operations," *IEEE Trans. Computers*, Vol. C-33, No. 6, pp. 518-528, June 1984.

[Iyengar 06] Iyengar, V., *et al.*, "At-Speed Structural Test for High-Performance ASICs," *Proc. Intl. Test Conf.,* pp. 1-10, 2006.

[Kundu 04] Kundu, S., T.M. Mak and R. Galivanche, "Trends in Manufacturing Test Methods and their Implications," *Proc. Intl. Test Conf.*, pp. 679-687, 2004.

[Li 08] Li, Y., S. Makar and S. Mitra, "CASP: Concurrent Autonomous Chip Self-Test using Stored Test Patterns," *Proc. Design Automation and Test in Europe (DATE)*, 2008.

[Mitra 05] Mitra, S., *et al.*, "Robust System Design with Built-In Soft Error Resilience," *IEEE Computer*, pp. 43-52, Feb. 2005.

[Mitra 06] Mitra, S., *et al.*, "Combinational Logic Soft Error Correction," *Proc. Intl. Test Conf.*, 2006.

[Mukherjee 02] Mukherjee, S., M. Kontz, S. Reinhardt, "Detailed Design and Evaluation of Redundant Multithreading Alternatives," *Proc. Intl. Symp. Computer Architecture*, 2002.

[Nigh 00] Nigh, P., and A. Gattiker, "Test Method Evaluation Experiments and Data," *Proc. Intl. Test Conf.*, pp. 454-463, 2000.

[Oh 02] Oh, N., P.P. Shirvani and E.J. McCluskey, "Error Detection by Duplicated Instructions in Super-Scalar Processors," *IEEE Trans. Reliability*, Vol. 51, Issue 1, pp. 63-75, March 2002.

[Reddy 02] Reddy, V., *et al.*, "Impact of Negative Bias Temperature Instability on Digital Circuit Reliability," *Proc. Intl. Reliability Physics Symp.*, 2002.

[Relangi 07] Relangi, P., and S. Mitra, "Erratic Bit Errors in Latches," *Proc. Intl. Reliability Physics Symp.*, 2007.

[Saxena 00] Saxena, N.R., *et al.*, "Dependable Computing and On-line Testing in Adaptive and Reconfigurable Systems," *IEEE Design and Test of Computers*, pp. 29-41, Jan-Mar 2000.

[Seifert 07] Seifert, N., *et al.*, "On the Scalability of Redundancy based SER Mitigation Schemes," *Proc. ICICDT*, 2007.

[Seshia 07] S. Seshia, W. Li and S. Mitra, "Verification Guided Soft Error Resilience," *Design Automation and Test in Europe*, 2007.

[Tschanz 07] Tschanz, J.W., *et al.*, "Adaptive Frequency and Biasing Techniques for Tolerance to Dynamic Temperature-Voltage Variations and Aging," *Proc. Intl. Solid-State Circuits Conf.*, 2007.

[Van Horn 05] Van Horn, J., "Towards Achieving Relentless Reliability Gains in a Server Marketplace of Teraflops, Laptops, Kilowatts, & "Cost, Cost, Cost," *Proc. Intl. Test Conf.*, 2005.

[Zhang 06] Zhang, M., *et al.*, "Sequential Element Design with Built-In Soft Error Resilience," *IEEE Trans. VLSI*, 2006.

[Zhang 07] Zhang, M., *et al.*, "Design for Resilience to Soft Errors and Variations," *Proc. Intl. On-line Test Symp.*, 2007.