

LIFETIME VALIDATION VIA ON-LINE TESTING

Qingru Meng

*Department of Electrical and Computer Engineering
Box 90291, Duke University
Durham, North Carolina 27708-0291, U.S.A.
email: qmeng@ee.duke.edu*

Lifetime Validation via On-line Testing (EE269) Course Project, Spring(1999)

Abstract

1 Introduction

A billion-dollar satellite careens out of orbit, blacking out millions of frustrate wireless communications users. A late-model luxury car laden with advanced electronics suddenly swerves into incoming traffic. Such events could happen. As the electronic systems become more complex, some design defects and manufacturing faults could escape detection. When combining with environmental disturbances such as electromagnetic interference and extremes of vibration and temperature, those faults may cause problems in the field.

Such disasters could be avoided by the incorporation of built-in self-test(BIST) techniques. Test Solutions for electronic systems have been through four generations from completely external testing (generation 1 and 2) to automatic testing (generation 3 and 4).

Generation 1: Functional verification vectors used as manufacturing test vectors (fault simulate supplement with additional functional vectors to raise fault coverage)

Generation 2: Scan insertion and Automatic Test Pattern Generation

Generation 3: Automatic Built-In Self-Test (ABIST) using 1149.X

Generation 4: Hierarchical and integrated BIST (HIBIST) from chip to system

Built-in self-test(BIST) is a design technique in which parts of a circuit are used to test the circuit itself. Comparing to other testing techniques, BIST has the following advantages: ability to test in-system and at-speed; compatibility with HDL and synthesis-based design flow ; ability to spans the test hierarchy from IC to MCM/PCB to system; ability to tests embedded or hard-to-reach functionality; reduced test development time; less dependence on expensive test equipment; and ability to diagnose performance-related and other failures over the entire life cycle. A BIST scheme development may involve the work in different hierarchical levels. Incorporation on-line BIST constraints during high level synthesis phase is the most important feature of generation 4 test solutions. BIST is especially attractive as a solution for deep-submicron and system-on-chip test problem.

BIST techniques can be classified into two categories, i.e. on-line BIST and off-line BIST. A circuit with BIST capability has two operating modes, namely normal functional mode and test mode. In on-line BIST, testing occurs during normal functional operating conditions while off-line BIST deals with testing a system when it is not carrying out its normal functions. On-line BIST features with fault tolerance at source, low-latency fault detection and correction, and fault effect localization, which are crucial for systems that operate under harsh environment while have to respond continuously in real time. Most embedded systems are such kind of systems. Embedded systems are computers incorporated in consumer products or other devices to perform application-specific functions. Embedded system can contain a variety of computing devices, such as microcontrollers, application-specific integrated circuits, and digital signal processors. Testing strategy that covers all expected permanent, intermittent, and transient faults is required for embedded systems so that field failure can be prevented and fault tolerant and fail-safe systems can be realized.

By using the terminology BIST, different people may cover different techniques. Some refer only to standard (off-line) BIST featured with a test pattern generator(TPG), a test response evaluator (TRE) and a BIST control unit (BCU) or circuits which can be reconfigured as TPGs, TREs and BCUs while others refer it to all techniques by which a circuit can test itself, as we described above.

This project will focus on possibilities for on-line BIST application to testing of transient, intermittent and start-up faults in embedded systems. First, the on-line testing issues of embedded systems are explored. Then an overview of a comprehensive collection of on-line BIST test techniques for VLSI will be conducted. The issues for standard BIST to be used in on-line testing are discussed, which followed by various BIST schemes which try to combine on-line and off-line techniques. Finally a practical experience is presented to demonstrate how on-line BIST is implemented in a specific case.

For the sake of simplicity, in the following part of this report, I'll refer off-line BIST simply as BIST and on-line BIST as on-line testing.

2 On-Line Testing

2.1 on-line testing issues in embedded systems

Generally on-line testing is used to detect operational faults resulting from wear or environmental disturbances during normal system operation. The goal of on-line testing is to detect fault effects and take appropriate corrective action.

There are four primary parameters to evaluate an on-line testing scheme:

◇ **Error coverage**—the ratio between the number of detected faults and the total number of modeled faults.

◇ **Error latency**—the difference between the first time an error becomes active and the first time it is detected, which depends on the test period. When error latency is difficult to determine, fault latency is considered. Fault latency is the difference between the onset of the fault, which may or may not lead to incorrect system states (errors), and its detection. Apparently, fault latency is greater than or equal to error latency.

◇ **Space redundancy**—extra hardware or firmware needed for on-line testing area overhead.

◇ **Time redundancy**—the extra time needed for on-line testing.

The ideal on-line testing scheme would have 100% error coverage, error latency of 1 clock cycle, no space redundancy and no time redundancy. [1] Actually it is impossible to achieve all these goals simultaneously in one testing scheme. High coverage usually requires high error latency, space redundancy and/or time redundancy. More hardware (high space redundancy) is needed to achieve immediate detection (short error latency) and low time redundancy. More hardware usually results in higher cost while long error latency usually means performance degradation.

Critical and highly available system requires very good error coverage and short error latency to minimize the probability of system failure so that performance is the overwhelming factor. However, embedded systems in consumer products are particularly cost sensitive. In many applications, low production and maintenance costs are as important as performance. Therefore, high hardware overhead is unacceptable. Tradeoffs have to be made according to the specific objectives of a testing plan.

2.2 on-line testing techniques

In on-line testing, the stimuli are provided by the patterns received during the normal mode of operation (functional data), which are unknown in advance, so that the response is unpredictable. Some invariant properties of response are needed in order to be able to check for errors during the system's operation. The general approach to on-line testing is based on reliable design techniques that create invariant properties easy to check. These techniques include hardware redundancy (duplication and spares), time redundancy (recomputing), information redundancy (error-detecting and error-correcting codes), and parametric monitoring (current, temperature and etc.)

Depending on the objectives of a testing, an on-line testing scheme can be implemented in two different modes, i.e., concurrent and nonconcurrent. A comprehensive online-testing to cover all fault types (permanent faults, intermittent faults and transient faults) usually require the combination of both test modes.

Permanent faults, remaining present after their occurrence, can be effectively detected by event-triggered and time triggered nonconcurrent testing. Intermittent faults exist only during some intervals, and thus periodic testing and/or concurrent checking techniques-if fault effects disappear quickly-can be used. Transient faults are one-time occurrence caused by some environmental disturbance and appear and disappear quickly so that concurrent testing is required to catch those faults.

Concurrent testing is performed simultaneously with normal functional operation. Common concurrent error detection techniques include the duplication-with-comparison (DWC), self-checking design, concurrent monitoring of reliability relevant parameters such as (current, temperature, clock waveforms, and etc.), using a watchdog timer.

The duplication-with-comparison (DWC) technique [4] detects any single error at the expense of 100% space redundancy. This technique requires two copies of the CUT, which operate in tandem with identical inputs. Any discrepancy in their outputs indicates an error. In space applications DWCs are very common but in many other applications DWC's high hardware overhead is unacceptable.

Based on information redundant techniques, Self-checking design implements functional blocks delivering an error detecting code and incorporates a checker to monitor this code (Fig. 1).

Figure 1: General Structure of Self-Checking Circuits.

Most useful of codes are: parity code, dual-rail code, unordered codes(m-out-of-n code, Berger code), and arithmetic codes (residue code and inverse residue code). To achieve Totally Self-Checking (TSC), some properties [2] must be verified by the functional block

and the checker. Self-check design has a much lower hardware cost than the duplication-with-comparison (DWC) technique. Moreover, the distribution of checkers through out a system provides a good measure of the location of a fault, which is indicated by the location of the checker where the error is detected. Self-checking designs assume that the fault is detected before the occurrence of another fault in the system. Therefore, multiple errors usually are not covered by self-checking techniques.

In contrast to the previously discussed techniques that use hardware or information redundancy for online testing, Jien-chung Lo [5] makes a strong case for the use of built-in current sensors(BICSs) for both on-line and off-line quiescent(I_{DDQ}) and/or transient (I_{DD}) current in deep-submicron designs. BICSs provide good fault coverage for a wide spectrum of faults(failures caused by fabrication imperfections, transient faults caused by electromagnetic radiation and heavy ions, etc.) with much lower hardware redundancy than conventional self-checking circuits. However, it is not an easy task to make the BICS be as fast as the monitored circuit and still offer a good resolution. Also, inserting a BICS in the current path of a circuit may affect circuit performance adversely.

Error detection by means of a watchdog is a two phase process. In the first phase(setup phase) the watchdog is provided with some information about the systems to be checked. During the second(checking) phase, it monitors and collects the relevant information concurrently. Error detection is done by comparing the information collected concurrently with the information provided during the setup phase. The information provided to watchdog can be about the memory access behavior, the control flow, the control signals or the reasonableness of results. [6]

For different parts of a system, different concurrent checking techniques may be used. Single error correcting and double error detecting codes can be used for memories, parity bits for data buses, residue codes for ALU's, signature monitoring for FSMs. On-line monitoring of reliability relevant parameters(current, temperature, abnormal delay, clock waveforms, etc.), capability-based addressing, watchdog timers, and replication can deal with system-level errors, such as the memory access behavior, the control flow, and so on.[6]

In non-concurrent testing, system is in an idle state and testing is accomplished by executing diagnostic software routines (macrocode) or diagnostic firmware routines (microcode) or periodic monitoring of reliability indicators (current, temperature, and etc). Such testing can be performed sporadically (event-triggered) or periodically (time-triggered). Event-triggered testing is initiated by key events or state changes such as start-up or shut down, and its goal is to detect permanent faults. Time-triggered testing take place at predetermined times in the operation of the system. Usually time triggered tests are partitioned and interleaved so that only part of the test is applied during each test period. Even though nonconcurrent testing can be implemented in a system on software level. However, in complex systems composed of several complex functional blocks, this option may result in complex software, very long test lengths and low fault coverage. BIST is a solution for simplifying software, reducing test length and improve fault coverage.

3 Issues of BIST Used for Online Testing

Standard BIST in ICs is implemented for manufacturing testing purpose, namely for off-line testing. When BIST is employed, a VLSI system is partitioned into a number of CUTs, each

component CUT is logically configured as generic BIST scheme. A generic BIST architecture is shown in Figure 2, where a test pattern generator (TPG) applies a sequence of test patterns to the CUT, a test response evaluator (TRE) evaluates the test responses and indicates any detected faults, and a BIST control unit (BCU) controls the BIST circuitry and CUT during self-test.

Figure 2: BIST Scheme

Even though traditionally BIST is implemented for manufacturing testing purposes, it can be exploited for nonconcurrent online testing of a system's logic and memory. By connecting the BIST control to the system reset, the off-line BIST hardware can be configured for event-triggered testing during system start-up or shutdown. BIST detecting all target faults within a fixed time can also be exploited for periodic testing with low fault latency. Various BIST techniques adapted in a context of periodic testing have been proposed and will be introduced in the section num.

In developing a BIST methodology for embedded systems, one must consider four primary parameters related to those listed earlier for on-line testing:

- ◇ **fault coverage**-the ratio of the faults activated and detected to faults of interest.
- ◇ **test set size**-the number of test patterns produced by the test generator. Test set size is closely linked to fault coverage; generally large test sets imply high fault coverage. For on-line testing, test set size must be small to reduce fault and error latency.
- ◇ **hardware overhead**-the extra hardware needed for BIST. High hardware overhead is not acceptable.
- ◇ **performance penalty**- the impact of BIST hardware on normal circuit performance, such as worst-case (critical path) delays.

Designers usually implement online BIST with the goals of complete fault coverage, low fault latency as well as minimum hardware overhead and performance penalty.

3.1 Basics of Standard BIST Methods

Test strategies according to the nature of test sequences can be classified into four classes:

- ◇ exhaustive or pseudo-exhaustive testing, applying all the logic combination at the inputs of CUT or segments of CUT, applicable only combinational circuits;
- ◇ pseudo-random or weighted pseudo-random testing, applying test patterns with characteristics of random patterns but generated deterministically and hence repeatable, applicable to both combinational and sequential circuits;
- ◇ deterministic testing, first generating pseudo-random test patterns and then tailoring the to the CUT's functional properties; and
- ◇ mixed testing or pseudo-deterministic testing, a tradeoff among test length, fault coverage and area overhead, which combines two different types of test sequences usually

a pseudo-random sequence followed by a deterministic sequence oriented to hard-to-detect faults. Hard-to-detect faults are usually called random pattern resistant (RPR) faults.

The most widespread **control scheme** are the test-per-scan scheme and the test-per-clock-scheme.

Test-per-scan scheme uses a complete or partial scan path which is serial filled by the TPG. At a capture clock, the content of the scan chain is applied to the circuit under test (CUT), and the CUT response is loaded into the scan chain in parallel. Then concurrently a new bit stream is shifted in, and the scan path output is compressed by the TRE. The test process can be accelerated if multiple scan chain are used. The test-per-scan scheme fits in any commercial flow which supports scan design.

Test-per-clock scheme uses special registers which work in four modes. In the system mode they operate just as a D-type flip-flops, in the pattern generation mode they perform autonomous state transitions, and the states are the test patterns, in the response evaluation mode the response of the CUT are compressed, and in the shift mode the registers work as a scan path.

In test-per-scan scheme, the BIST hardware is mainly kept apart from the mission logic, and the performance degradation is not higher than the impact of a scan design for external testing. The BIST control unit and the overall hardware overhead for the test-per-scan scheme are smaller than that for a test-per-clock scheme. However, the test-per-scan scheme has long test time for serial pattern generation, and the low detectability of transition faults which require a two-pattern test.

The advantages of test-per-clock scheme are its short testing time and high speed since a new pattern is generated in each clock cycle at least for a part of the circuit. On the other hand, the larger test registers may result in high area overhead and integrating test registers into the data path has a stronger impact on system performance than integrating a scan path.

Huge numbers of TPGs and TREs have been proposed for various BIST strategies and LFSRs are central elements for most of them.[7] An LFSR is formed from standard flip-flops with outputs of selected flip-flops being fed back (modulo 2) to its inputs. When used as a test generator, an LFSR is set to cycle rapidly through a large number of its states which serve as test patterns. These states are completely determined by the LFSR's design parameters which define a characteristic polynomial. When an LFSR serves as a TRE, it receives a sequence of test responses and then forms a fault signature, which is compared to a known or generated good signature to determine whether a fault is present. LFSRs are recognized for their efficiency, simplicity and randomness in BIST applications. But when circuits contain random-pattern-resistant faults, test length can be very long to insure a sufficiently high fault coverage. To improve fault coverage and cut test length, many approaches have been shown: insertion of additional logic to switch some random patterns into deterministic ones; changing the seeds or computing optimal seeds; using multiple polynomials; using bit-flipping function, and so on.

3.2 BIST Tailored for Online Testing

Various BIST techniques that achieve short test lengths and thus allow periodic testing with short interruption time of system operation have been proposed. Some pseudo-concurrent BIST techniques are also proposed.

Al-Assad et al. [18] explored the efficient test sets and test pattern generators targeting the on-line BIST of high-performance, scalable datapath circuits. High-level models are used to identify potential test sets for a small version of the circuit to be tested. Then a regular test set is extracted and a test generator TG is designed such that scalability, small test set size, full fault coverage, and very low hardware overhead are achieved. TG takes the form of a twisted ring counter with a small decoder array and is tailored to generate a regular, deterministic test set of near-minimum size.

Transparent BIST for memories [17] preserves the state of the memory so that the system is not losing its operation context after each test phase. This kind of BIST is very suitable for periodic testing of RAMs. Transparent BIST involves only slightly higher area overhead with respect to standard BIST while does not decrease the fault coverage for modeled faults and behaves better for unmodeled ones.

Built-in Concurrent Self-test (BICST) [16] proposes a solution that exploited BIST techniques for performing pseudo-concurrent checking. The extra hardware for BICST consists of a concurrent test circuit and a comparator which is shared between identical circuits. This technique provides the circuits with enhanced detection capability for transient and intermittent faults in addition to permanent faults.

A novel test architecture called Mutual Checking BIST(MC-BIST)[22] is proposed for highly concurrent systems having a large number of functional modules of a similar nature. MC-BIST uses a judicious combination of mutual testing and signature testing to achieve low test area overhead, low aliasing probability and low test-application time.

Voyiatzis et al propose a Windowed comparative concurrent BIST (w-CBIST) to reduce test latency.

High-level synthesis (such as register and interconnect assignment techniques address the BIST issue [23]) help with minimizing extra test logic, minimizing test sessions, adding test behavior, using arithmetic units as test generators and compactors. [20]

Orailoglu [19]presented a microarchitectural synthesis for rapid BIST testing, which targets test concurrency during high-level and structural synthesis. High-level synthesis generates RTL circuits which guarantee concurrent controllability and observability of all hardware components form test registers. Structural synthesis for testability completes the microarchitectural definition by specifying the test registers in the circuit and define a BIST test plan. Remaining test conflicts are avoided without reducing test throughput by using partial-intrusion BIST to enable test data to be pipeline through non-test registers. The use of pipelined BIST testing in conjunction with high-level synthesis for test conflict minimization enables reduced test time through high test concurrency.

3.3 Unified BIST

As we have noticed, traditionally, BIST structures used for on-line testing have been different from those for off-line production testing. Replicated hardware, comparators, and checkers

are typical on-line BIST structures. On the other hand, typical off-line BIST includes test patterns generators and test response evaluators which are used only during the test mode. In a system with both structures, hardware overhead can be very high. Therefore there is a need for unified BIST structures which support both on-line and off-line with shared hardware resources. Moreover, BIST can be used for checking the elements which may not be exercised by normal function patterns in a given period of time and self-checking circuits may introduce additional testability to off-line testing.

Nicolaidis [8] and Gupta *et al.* [9] employ conventional self-checking and off-line BIST techniques to provide dual testability. Modified checkers, called self-exercising checkers are used to introduce additional testability during off-line testing. However, since the testing schemes employ separate testing techniques in on-line and off-line tests, the hardware overhead remains high. Another approach for combining BIST and self-checking is reported by Lala *et al.*, [10]

Saluja *et al.*[11] propose to use signature analysis, a commonly used off-line test techniques for both on-line and off-line tests. The off-line resources are modified such that during system operation, they can also be used to observe the normal inputs and outputs of a circuit, which means the two test modes share the hardware resources and leads to a small hardware overhead. However, since a pattern matching between an input vector and a test pattern generated by the test generator is required during on-line testing, the concurrent error latency for exhaustive testing can be long.

Sun *et al.* utilize the existing hardware resources for off-line BIST to provide a degree of on-line monitoring with a lower hardware overhead in comparison to self-checking systems. The proposed design method employs cyclic code testing to combine on-line testing and signature analysis in a built-in fashion implemented by bit-sliced LFSR's/LCAR's. The drawback of the scheme is its low error coverage and long error latency for on-line testing.

Mukherjee *et al.*[12] report a Versatile BIST approach (VBIST) that targets both off-line and on-line self-test of data-path architectures in digital signal processing. VBIST uses off-line BIST circuitry for on-line testing as well. Unlike traditional on-line self-test approaches, VBIST does not use functional data as test inputs. Rather, VBIST generates test patterns and compacts test response during the normal mode of operation by exploiting the spare computation cycles of functional units in the design. VBIST entails little additional impact on performance and area of the design. Furthermore, due to regular iterative structures for most functional blocks in data-path architectures, the proposed method yield very short test latencies.

A unified on-line and off-line BIST scheme for ROMs is presented by Yeung *et al.*[15] This scheme is based on memory partitioning and signature analysis and supports both on-line and off-line testability with a hardware cost comparable to parity checking. The on-line testing scheme provides much better error detecting capability than conventional strategies. The off-line test retains a high error coverage of at least 99.9985

Karpovsky suggests to integrate on-line and off-line error detection mechanisms in the coding theory framework. An optimal selection of error detecting codes for combining online checking with offline BIST can reduce the aliasing probability of off-line BIST by monitoring the output of a concurrent checker and the probability of not detecting an error in the computing mode by a short periodic BIST. [21]

4 Case Study—On-line BIST in Embedded-systems

In this section, a practical design is present to show how BIST combining with other on-line checking features to employ a comprehensive on-line testing to a embedded system.

AE11 microcontroller is a single-chip design using SGS-Thomson's CMOSM5 technology with a 0.7-micron feature size and two metal layers. Compatible with Intel's 8051 eight bit instruction-set architecture, the AE11 has a 4-Kbyte main memory(RAM) and a fairly conventional set of peripheral I/O modules. It is designed to detect and respond to hardware faults within milliseconds under all operating conditions. A supervisory system can respond either with a fail-safe shutdown or by gracefully degrading the system's performance.

The main design objectives for the AE11 were very high fault coverage and very short fault detection latency. To meet these stringent goals, the developers included essentially all the online-testing features discussed earlier-particularly concurrent checking based primarily on parity coding, and time-triggered checking based on special BIST circuitry. Because the fault detection latency is very short, the BIST functions must be invoked periodically at least once every fault latency period. Thus, these functions belong in the on-line category.

The main test features of AE11 include the following:

- ◇ parity code checking throughout the system
- ◇ parity checking and ALU parity prediction in the CPU data path
- ◇ program control flow checking via signature monitoring
- ◇ self-checking address-decoding logic in the RAM
- ◇ programmable watchdog timer
- ◇ pseudo-random and I_{DDQ} testing of peripheral modules
- ◇ power supply and temperature monitoring
- ◇ test control logic using boundary scan and a test access port controller

The overall test process:

At system start-up, the test controller executes a set of tests to check that the major subsystems – CPU, RAM, and peripheral modules – are fault-free. During normal operation, the concurrent-checking circuits flag errors as soon as they occur. In addition, the microcontroller stops periodically to execute I_{DDQ} and various functional tests. Functional tests are needed to test circuits that are not self-testing, such as the interrupt controller and the ADC's input channels.

The concurrently self-checking of RAM is achieved by adding a parity-check bit to both the address and data buses and memory parity checkers. Additional self-testing logic detects both word line and address decoder faults. Special circuits are used to detect bridging faults.

For the CPU, designers use an approach called parity prediction to check the CPU's data path, which has the advantages of lower cost and compatibility with the AE11's overall use of parity codes. The CPU's control unit also applies parity checking to control words. In addition, software signatures are inserted into the application program flow during compilation. The AE11 executes special instructions that monitor these signatures to detect many types of hardware and software errors.

The AE11 uses on-line BIST and I_{DDQ} testing on the peripheral modules. A test controller conforming to the IEEE 1149.1 boundary-scan standard handles test functions. The BIST logic employs pseudo-random test and signature generation implemented by LFSRs in the CPU and multiple-input signature registers (MISR) in the peripheral modules. The use of many pseudo-random test patterns provides a measure of protection against unknown

or nontargeted hardware faults, as well as standard stuck-at faults. The system data bus transmits the pseudo-random test vectors from the LFSRs to the circuits being tested. The MISRs compress the resulting responses to obtain signatures that control the bist_ok signals.

Some of the test vectors also setup I_{DDQ} tests that use on-chip current monitors. The AE11 also includes special fault detection and error-monitoring procedures for components such as the analog-to-digital converter (ADC). Other specialized test circuits monitor the power supply voltage and the chip temperature.

It's estimated that the AE11's self-testing hardware features achieve over 99.7% of modeled faults and errors, with less than 3515

References

- [1] H. AL-Asaad, et al., "Online BIST for Embedded Systems," *IEEE Design & Test of Computers*, vol. 15, No. 4, pp. 17-24, 1998.
- [2] W. C. Carter and P. R. Schneider, "Design of Dynamically Checked Computers," *Proc. 4th Congress IFIP*, Edinburgh, vol. 2, pp878-883, 1968.
- [3] M. Nicolaidis and Y. Zorian, "On-Line Testing for VLSI-A Compendium of Approaches," *J. Electron. Testing*, Vol. 12, pp.7-20(1998).
- [4] B.W.Johnson, *Design and Analysis of Fault Tolerant Digital Systems*, Addison-Wesley, 1989.
- [5] J. Lo, "Online Current Testing," *IEEE Design and Test of Computers*, vol.15, No.4, pp 49-56, 1998.
- [6] A. Mahmood and E. McCluskey, "Concurrent Error Detection Using Watchdog Processors -A Survey," *IEEE Trans. Computers*, Vol. 37, No. 2, pp.160 174(1988).
- [7] C. Dufaza, "Theoretical properties of LFSRs for built-in self test," *Integration, the VLSI Journal*, pp17-35, Vol. 25, No. 1, 1998.
- [8] M. Nicolaidis, "Self-exercising Checker for Unified built-in self-test (UBIST)," *IEEE Trans. Computer-Aided Design*, Vol.8, pp. 203-214, 1989.
- [9] S. K. Gupta and D. K. Pradhan, "Can Concurrent Checkers help BIST," in *Proc. Int. Test Conf.*, pp140-150, 1992.
- [10] P. Lala, et al., "A Unified Approach for Off-line and On-line Testing for VLSI systems," *Proc. IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems*, pp195-203, 1996.
- [11] K. K. Saluja, et al., "Concurrent comparison Testing Using BIST Resources," in *Proc. Int. Conf. Computer-Aided Design*, pp336-339 (1987).
- [12] N. Mukherjee and R. Karri, "Versatile BIST: An Integrated Approach to On-line/Off-line BIST for Data-Dominated Architectures," *J. Electronic Testing: Theory and Applications*, vol. 13, pp189-200, 1998.

- [13] R. Singh and J. Knight, "Concurrent Testing in High-Level Synthesis," *Proc. Int. Symp. High-Level Synthesis*, pp96-103, 1994.
- [14] S. Hellebarand, et al., "Mixed-mode BIST Using Embedded Processor," *J. Electronic Testing: Theory and Applications*, vol. 12, pp127-138, 1998.
- [15] D. Yeung, et al., "A unified on-line and off-line BIST scheme for ROMs," *Proc. IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing*, pp469-472, 1995.
- [16] R. Sharma and K. Salyja, "An Implementation and Analysis of a Concurrent Built-in Self-Test Technique," *Proc. 18th Int. Symp. Fault Tolerant Computing*, 1988.
- [17] M. Nicolaidis, "Theory of Transparent BIST for RAMs," *IEEE Trans. Computers*, Vol.45, No.10, 1996.
- [18] Hussain Al-asaad and John P. Hayes, "Scalable Test generators for high speed datapath circuits," *J.electronic testing: theory and applications*, Vol.12, pp111-125, 1998.
- [19] A. Orailoglu, "Microarchitectural synthesis for rapid BIST testing," *IEEE Trans. Computer-Aided Design*, vol. 16, no. 6, pp573-586, 1997.
- [20] I. Ghosh and N. K. Jha, "High level Synthesis: a Survey," *Integration-the VLSI Journal*, Vol.26, pp79-100, 1998.
- [21] M. Karpovsky, "Integrated on-line and off-line error detection mechanisms in the coding theory framework," *VLSI Design*, vol.5, no.4, pp313-331, 1998.
- [22] M. Adulla, et al., "Optimization of Mutual and Signature testing Schemes for Highly Concurrent Systems," *J. Electron. testing: Theory and Applications*, Vol.12, No.3, pp199-216, 1998.
- [23] I. Parulkar, et al., "Allocation Techniques for Reducing BIST area Overhead of Data paths," *J. Electron. testing: Theory and Applications*, Vol.14, pp149-166, 1998.
- [24] F. Pichon, "Testability Features for a Submicron Voice-coder ASIC," *1996 IEEE Intl. Test Conf.*, pp377-385, 1996.