Fault Detection and Diagnosis Techniques for Molecular Computing

Mehdi B. Tahoori[1] and Subhasish Mitra[2]

[1]Northeastern University, Boston, MA USA, mtahoori@ece.neu.edu
[2]Intel Corp, Sacramento, CA USA, subhasish.mitra@intel.com

ABSTRACT

Fault detection and diagnosis techniques that are essential for defect tolerance, fault tolerance and self-repair of molecular computing systems are discussed. These techniques enable robust molecular computing system design protected from manufacturing defects and run-time errors in the underlying hardware.

Keywords: molecular electronics, test, diagnosis, BIST, defect-tolerance, fault-tolerance, self-repair.

1 INTRODUCTION

Nano-scale devices and molecular electronics promise to overcome the fundamental physical limitation of lithography-based silicon VLSI technology. It is projected that molecular electronics can achieve density of 10^{12} devices per cm^2 and operate at THz frequencies. Researchers have demonstrated several successful nano-scale electronic devices including carbon nano-tubes [Iijima 91] [Bachtold 01][Fuhrer 00][Rueckes 00], silicon nano-wires [Kamins 00] [Huang 01], single electron devices, and quantum dot cells [Tougaw 94]. Regular programmable architectures for nano-scale devices that are conceptually similar to *field programmable gate arrays* (FPGAs) are currently being investigated by several researchers [Dehon 03a][Dehon 03b][Beckett 03][Butts 02][Goldstein 01] [Rueckes 00][Ziegler 02]. Unlike FPGAs where logic functions are realized by *look-up tables* (LUTs), logic functions in these architectures are based on *programmable logic arrays* (PLAs).

Several earlier publications stressed the need for defect and fault tolerance for circuits realized by nano-scale devices [Butts 02] [Collier 99] [Dehon 03a][Goldstein 02]. *Defect-tolerance* techniques are generally used to tolerate manufacturing defects and are mainly useful for yield enhancement purposes [Siewiorek 92]. In the context of traditional systems, repair of memory defects using spare rows and columns is an example of defect tolerance. Some publications such as [Butts 02] observed that nano-scale devices are more vulnerable to temporary errors and permanent faults during system operation compared to conventional CMOS devices. Fault-tolerance techniques are used to tolerate errors occurring during normal operation. Fault-tolerance techniques include masking (e.g., Triple Module Redundancy), concurrent error detection, recovery (e.g., rollback and roll-forward recovery) and self-repair [Siewiorek 92, Pradhan 96].

Thorough testing and precise high-resolution location of the failing resource in a defective part are keys to successful implementations of defect and fault tolerance. Thorough manufacturing testing is required to identify a defective manufactured part. During system operation, periodic testing is required to identify a defective system component with a permanent fault. High-resolution diagnosis is essential to precisely locate a defective resource so that efficient repair can be performed. Since a molecular computing system is expected to have close similarities with reconfigurable systems (such as those implemented using FPGAs), testing and diagnosis techniques for such systems can be tailored to be efficient in the context molecular systems. In this paper, we discuss the applicability of test and diagnosis techniques originally developed for FPGAs to molecular computing systems.

The rest of this paper is organized as follows. Defect and fault tolerance techniques for reconfigurable systems are discussed in Sec. 2. Application-independent test and diagnosis techniques are presented in Sec. 3. These techniques are mainly useful for thorough manufacturing testing and defect-tolerance purposes. Application-dependent test and diagnosis techniques are discussed in Sec. 4. These techniques are used for fault-tolerance and self-repair, and to some extent for defect tolerance. Section 5 concludes this paper.

2 DEFECT AND FAULT TOLERANCE IN RECONFIGURABLE SYSTEMS

The Termac project at HP-labs is an example of the use of defect tolerance in a reconfigurable system [Culbertson 97]. In this project, defect tolerance is achieved by applying thorough testing and diagnosis to identify defective (unusable) resources from defect-free (usable) resources, and an efficient algorithm to map an entire design to the usable resources.

As a practical industrial effort for defect tolerance, Xilinx has announced Easy-Path solution [Xilinx Easy Path]. Due to manufacturing defects, some FPGA parts cannot be used for all possible applications. For example, if a programmable switch is permanently off, any design which uses that particular switch will become faulty. In the regular manufacturing flow, these parts result in yield loss. However, these chips may still be usable for some designs that do not use the defective resources. By testing the resources of an FPGA with respect to a specific design to be implemented on it, these previously marked "faulty" chips can be used and sold to customers. These FPGAs, which are good only for a few particular design and do not have general programmability of typical FPGAs are called

application-specific FPGAs (ASFPGAs). ASFPGAs are profitable for relatively large volume designs which have been completely finalized, i.e. the final placed and routed version is fixed. The main challenge in this flow from defect tolerance perspective is to thoroughly test the FPGA chip with respect to a particular application and to identify the defective resources.

The ROAR (Reliability Obtained by Adaptive Reconfiguration) project at the Stanford Center for Reliable Computing is an example of the design of a self-repairing reconfigurable architecture based on dual FPGAs with embedded "soft" micro-controllers [Mitra 04]. The difference between the ROAR project and the Teramac project is in the "on-line" aspect of error detection, recovery and self-repair addressed by the ROAR project.

The ROAR project achieves fault-tolerance in reconfigurable systems using on-line error detection during system operation also called *concurrent error detection* (CED), very fast fault location and quick recovery from temporary failures and fast repair of the system from permanent faults. Unlike conventional fault-tolerant systems where the *Field Replaceable Unit* (FRU) is a chip or a board, the FRU for an *adaptive computing system* (ACS) is a logic or a routing resource; thus, there the system can be repaired taking advantage of the fine granularity of the FRU without any standby spare hardware.

Concurrent error detection (CED) is a major component of any dependable system. CED techniques detect errors during system operation. The major objective of any CED technique is to guarantee system data integrity. By data integrity we mean that the system either produces correct outputs or produces an error signal when incorrect outputs are produced. Various CED techniques using in the ROAR project are discussed in [Saxena 03]. For the system to recover from errors after error detection, any system with CED must be equipped with sufficient capabilities to locate the faulty part and perform recovery.

There are two types of failures that can affect system operation, temporary and permanent failures. When an error is detected, the system first assumes that the error is temporary and performs temporary error recovery. This is because, the general belief is that temporary failures are more common compared to permanent faults. However, if the error still persists after a pre-determined number of recoveries, the system assumes that the error is due to a permanent failure. After a permanent fault is detected and the faulty part is located, permanent fault recovery schemes that reconfigure the system to replace the faulty part with originally unused resources are used to repair the system. The reader is referred to [Mitra 04] for a comprehensive description of temporary error recovery and permanent fault repair techniques used in the ROAR project.

Abramovici et al. also presented a roving STAR approach for fault detection and location during system operation [Abramovici 99]. One concern with the roving STAR approach is the performance and availability degradation of the system because of the moving of the STAR, even when failures are not present.

The purpose of Fig. 1 is to show where the various test and diagnosis techniques discussed in this paper fit in. Application-independent test and diagnosis are used after manufacturing mainly for identifying defective parts and also for defect tolerance techniques such as those used in the Teramac project. Application-dependent test and diagnosis is used for fault tolerance during system operation and also for defect-tolerance purposes such as the Xilinx EasyPath flow.

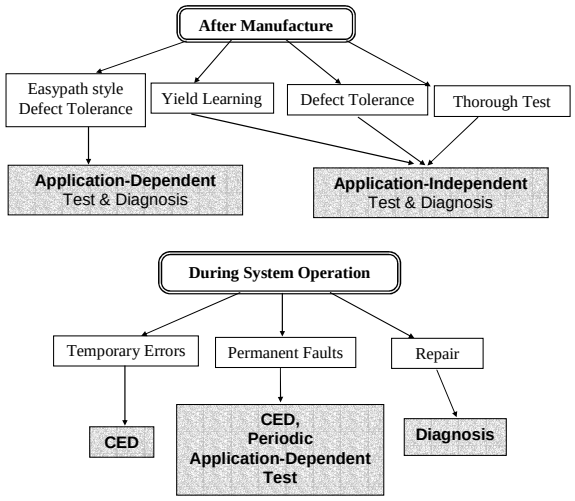

Figure 1 Test and diagnosis for defect and fault tolerance

3 APPLICATION-INDEPENDENT TEST & DIAGNOSIS

New manufacturing process techniques and steps for nano-scale devices that are conceptually different from conventional lithography-based process techniques may result in new failure mechanisms not completely understood today. Various fault models such as stuck-at, bridging, stuck-open, transition, delay and bridging faults are used for conventional CMOS logic testing [Abramovici 90]. Among them, single stuck-at fault model seems to be the most effective, in terms of detection of defective parts although it is not very accurate [McCluskey 00]. The industry uses transition fault model for detecting delay defects. It's not known whether more sophisticated fault models will be required for molecular technologies. Therefore, test techniques for such systems shouldn't be restricted to a particular fault model, but very comprehensive to cover all logical fault models. Some examples of such tests are exhaustive or pseudo-exhaustive testing in which detection of all faults in a given class (for example, all combinational faults inclusive of all stuck-at faults) is guaranteed [McCluskey 84]. The major drawback of pseudo-exhaustive testing is the test length and test time.

Due to reprogrammability, a reconfigurable chip can be configured in an incredibly large number of ways by the

user. From manufacturing testing point of view, it must be ensured that the part is functional under all these configurations, as the actual user configuration is not known during production test. Application-independent testing not only consists of developing and applying test vectors, as in conventional application-specific integrated circuits (ASICs), but also of generating and configuring the chip with a set of test configurations, i.e. similar to FPGA testing. The chip must be configured into a test configuration prior to test vector application, in which some resources are programmed (activated) as the resource under test (RUT) and possibly appropriate test circuitry is implemented on a different portion of the chip. The main challenge in application-independent testing is to achieve a high fault coverage with respect to an appropriate fault list, while the number of test configurations is minimized. Note that the total test time is dominated by loading test configurations rather than application of test vectors.

In order to manage the complexity of testing millions of reconfigurable devices on chip, different type of resources (i.e. programmable logic, programmable memory, and programmable interconnect) are tested in different set of test configurations.

Most of the area of a programmable chip is dedicated to interconnects. Interconnect faults consist of permanent on and off faults for programmable switches, and open and bridging faults for wires. Testing interconnect resources is performed by configuring the device into a number of *wires under test* (WUTs). The logic resources are configured as transparent logic, identity function, followed by flip-flops. This improves controllability and observability, as well as the diagnostic resolution. Delay testing for interconnects ensures the performance of the mapped designs. Automatic interconnect test configuration generation techniques are presented in [Tahoori 03a].

Since the logic resources in reconfigurable molecular computing is based on PLA (regular AND and OR planes), easily testable PLA structures and corresponding test sets can be exploited to fully test the logic blocks. Special fault models for PLAs, PLA test and design for testability techniques are discussed in [Khakbaz 82, McCluskey 86].

High resolution diagnosis is required to identify defective resources such as programmable switch, wire, or logic cell. In the Teramac project, this was done by considering only those resources that are used in the passing test configurations to be fault-free. No conclusion is made for resources in the failing test configurations [Culbertson 97]. Such a simple approach can be effective for manufacturing processes with low defect densities. For high defect densities, more sophisticated diagnosis techniques such as those discussed in [Abramovici 00, Tahoori 02] may be required.

4 APPLICATION-DEPENDENT TEST & DIAGNOSIS

In application-dependent testing, the resources are tested only with respect to a particular mapped design. In other words, unused resources are not tested. This test typically consists of fewer test configurations and hence shorter test time compared to application-independent test.

Similar to an application-independent test approach, logic and interconnect resources can be tested separately. In this specific strategy, the interconnect is first tested by modifying the logic block configuration and preserving the interconnect configuration. Then, logic blocks are tested by modifying the interconnect configuration and preserving the logic block configuration. . In the first phase, only the configuration of logic blocks are changed, and the interconnect configuration is *not* modified. Hence, no extra placement and routing is required for test configuration generation. In the second phase, the configuration of original used logic blocks is preserved and configuration of interconnects and unused logic blocks are changed to exhaustively test all used logic blocks. A *Built-in Self Test* (BIST) architecture can be embedded in this test configuration in order to generate the test vectors and observe the outputs in order to generate Go/NoGo signal [Tahoori 03b][Tahoori 04a].

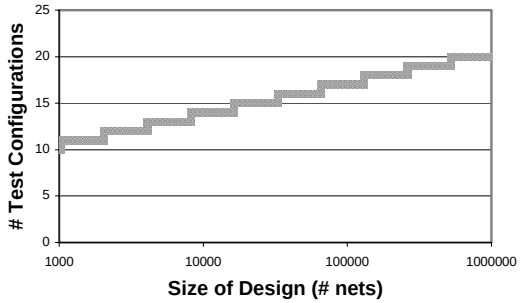

Figure 2 Test configurations for 100% coverage

By implementing special logic functions in the used logic resources, all possible interconnect faults can be detected in very few test configurations. For example, if all used logic resources are configured as AND function, by applying all-1 pattern as the test vector, any stuck-at-0 fault in interconnects will be detected. As presented in [Tahoori 03b], the required number of test configurations is logarithmic to the number of nets in the design. This is shown in Figure 2. For example, for 10^{12} programmable resources, at most 40 test configurations are required to detect all possible stuck-at, open, and bridging faults. This approach can also be extended to generate very few test configurations for thorough testing of defects affecting the interconnects. The technique in [Tahoori 04b] uses only two to four configurations to verify the timing of all paths in the design in order to guarantee the required system performance.

An extended version of the test configurations used for application-dependent test can be applied for diagnosis. The idea is to activate different sets of faults in each test configuration so that the failing pattern for the total set of test configurations uniquely identifies faulty resources. The number of required test configurations is still logarithmic to the size of the design. For example, for 10^{12} devices, at

most 120 configurations are required to uniquely identify any stuck-at, open, or bridging fault.

5 CONCLUSION

Thorough testing and precise diagnosis techniques discussed in this paper can enable design of molecular systems with defect and fault tolerance. The discussed techniques are effective for reconfigurable hardware such as FPGAs. Future work includes a thorough (probably simulation-based) study of the implementation and improvement of these techniques in a molecular computing system.

REFERENCES

[Abramovici 00] M. Abramovici, C. Stroud, "BIST-Based Detection and Diagnosis of Multiple Faults in FPGAs," Proc. of Int'l Test Conf., 2000.

[Abramovici 99] M. Abramovici, C. Stroud, C. Hamilton, C. Wijesuriya and V. Verma, "Using Roving Stars for On-line Testing and Diagnosis of FPGAs," Proc. Intl. Test Conf., pp. 973-982, 1999.

[Abramovici 90] M. Abramovici, M. Breuer and A. Friedman, "Digital Systems Testing and Testable Design," IEEE Press, 1990.

[Bachtold 01] A.Bachtold, P.Harley, T.Nakanishi, C.Dekker, "Logic Circuits with Carbon Nanotube Transistors", Science vol 294, pp. 1317-1320, 2001.

[Beckett 03] P. Beckett, "Exploiting multiple functionality for nano-scale reconfigurable systems", Proc. ACM Great Lakes Symposium on VLSI, pp. 50-55, 2003.

[Butts 02] M.Butts, A. DeHon,S.C.Goldstein, "Molecular Electronics: Devices, Systems and Tools for Gigagate, Gigabit Chips", Proc. Int'l Conf. on Computer-Aided Design, pp. 443--440,2002.

[Collier 99] C.P.Collier, E.W.Wong, M.Belohradsky, F.M.Raymo, J.F.Stoddart, P.J.Kuekes, R.S.Williams, J.R.Heath, "Electronically Configurable Molecular-Based Logic Gates", Science, vol 285, pp. 391-394, 1999.

[Culbertson 97] Culbertson, W., R. Amerson, R. Carter, P. Keukes and G. Snider, "Defect-Tolerance in the Teramac Custom Computer," Proc. Intl. Symp. Field Programmable Custom Computing Machines (FCCM), pp. 116-124, 1997.

[Dehon 03a] A. DeHon, "Array-Based Architecture for FET-Based, Nanoscale Electronics", IEEE Trans. on Nanotechnology, Volume 2, Number 1, Pages 23--32, Mar 2003.

[Dehon 03b] A. DeHon, "Stochastic Assembly of Sublithographic Nanoscale Interfaces", IEEE Trans. on Nanotechnology, Volume 2, Number 3, Pages 165--174, September 2003.

[Fuhrer 00] M.S.Fuhrer, J.Nygard, L.Shih, M.Forero, Y.Yoon, M.S.C.Mazzoni, H.J.Choi,J.Ihm, S.G.Louie, A.Zettl, P.L.McEuen, "Crossed Nanotube Junctions", Science vol 288, pp. 494-497, 2000.

[Goldstein 02] S.C. Goldstein,D.Rosewater, "Digital Logic Using Molecular Electronics", Proc. IEEE Int'l Solid-State Circuits Conf., 2002.

[Goldstein 01] S.C.Goldstein, M.Budiu, "NanoFabrics: Spatial Computing using Molecular Electronics" Proc. Int'l Symp. on Computer Architecture, 2001.

[Iijima 91] S. Iijima, "Helical Microtubules of Graphitic Carbon," Nature, vol. 354, pp. 56, 1991.

[Kamins 00] T.I. Kamins, R.S.Williams, Y. Chen, Y.-L. Chang, and Y.A. Chang, "Chemical vapor deposition of Si nanowires nucleated by TiSi2 islands on Si," Applied Physics Letters, vol. 76, no. 562, 2000.

[Khakbaz 82] J. Khakbaz, and E.J. McCluskey, "Concurrent Error Detection and Testing for Large PLA's," Joint Special Issue on VLSI, IEEE Trans. on Electron Devices, pp. 756-764 and IEEE J. of Solid-State Circuits, pp. 386-394, Apr. 1982.

[McCluskey 00] E. J. McCluskey, and C. W. Tseng, "Stuck-Fault Tests vs. Actual Defects," Proc. Intl. Test Conf., 2000, pp. 336-343.

[McCluskey 86] E. J. McCluskey, Logic Design Principles, Prentice-Hall Inc., Englewood Cliffs, N.J., 1986.

[McCluskey 84] E. J. McCluskey, "Verification Testing - A Pseudoexhaustive Test Technique," IEEE Trans Comp., pp.541-546, 1984.

[Mitra 04] Mitra, S., W.J. Huang, N. Saxena, S.Y. Yu and E.J. McCluskey, "Dependable Reconfigurable Computing: Reliability Obtained by Adaptive Reconfiguration," ACM Trans. Embedded Computing Systems, To appear.

[Rueckes 00] T.Rueckes, K.Kim, E.Joselevich, G.Y.Tseng, C.Cheung, C.M.Lieber, "Carbon Nanotube-Based Nonvolatile Random Access Memory for Molecular Computing", Science, vol 289, pp. 94-97, 2000.

[Huang 01] Y.Huang, X.Duan, Y.Cui, L.J.Lauhon, K.Kim, C.M.Lieber, "Logic Gates and Computation from Assembled Nanowire Building Blocks", Science vol 294, pp. 1313-1317,2001.

[Pradhan 96] D. K. Pradhan, "Fault-Tolerant Computer System Design", Prentice Hall, 1996.

[Saxena 03] N. Saxena, S. Mitra, C. Zeng and E. J. McCluskey, "Concurrent Error Detection and Design Diversity In Reconfigurable Computing - New Opportunities," IEEE Design and Test of Computers, 2003.

[Siewiorek 92] Siewiorek, D. P., and R. S. Swarz, "Reliable Computer Systems: Design and Evaluation," 2nd Edition, Digital Press, 1992.

[Tahoori 02] M.B.Tahoori, "Diagnosis of Open Defects in FPGA Interconnects," Proc. IEEE Int'l Conf. on Field-Programmable Technology, pp. 328-331, 2002.

[Tahoori 03a] M. B. Tahoori, S. Mitra, "Automatic Test Configuration Generation for FPGA Interconnect Testing," Proc. IEEE VLSI Test Symp., 2003.

[Tahoori 03b] M. B. Tahoori, "Application-Dependent Interconnect Testing of FPGAs", Proc. IEEE Symp. On Defect and Fault Tolerance in VLSI, pp. 409-416, 2003.

[Tahoori 04a] M. B. Tahoori, E. J. McCluskey, M. Renovell, P. Faure, "A Multi-Configuration Strategy for an Application Dependent Testing of FPGAs", to appear in Proc. VLSI Test Symp., 2004.

[Tahoori 04b] M. B. Tahoori, S. Mitra, "Thorough Delay Testing of Designs on Programmable Logic Devices," in preparation.

[Tougaw 94] P.D. Tougaw and C.S. Lent, "Logical Devices Implemented Using Quantum Cellular Automata," Applied Physics, Vol 75(3), pp. 1818-1825, 1994.

[Ziegler 02] M.M.Ziegler, M.R.Stan, "Design and analysis of crossbar circuits for molecular nanoelectronics", Proc. IEEE Int'l Conf. on Nanotechnology, 2002.

[Xilinx EasyPath] Xilinx Easy Path Solution, http://www.xilinx.com, 2002.