To Appear RAMS 2007

Modeling Failure Reduction for Combinational Logic using Gate Level NMR

Drew C. Ness, University of Minnesota
Christian J. Hescott, University of Minnesota
David J. Lilja, Ph.D., University of Minnesota

Key Words:

Fault-tolerance, N modular redundancy, soft errors

SUMMARY & CONCLUSIONS

We present a mathematical model for describing the relationship between overhead and error reduction under single-error conditions in combinational logic using N-modular redundancy (NMR) as an upper bound. We then provide an analysis for the model under more realistic circuit and overhead assumptions. We compare system, sub-circuit, and gate level NMR. Based on our results, implementing NMR at the gate level offers the benefits of NMR with customizable overheads but with reduced effectiveness when compared to system level implementations.

1 INTRODUCTION

Increasingly, research is focusing on the growing soft error rates (SERs) in CMOS and the related problems [1,2,3]. A number of investigations have stated that the SER in logic will become a significant factor for future devices [3,4]. To address this problem we will investigate the impact and trade-offs associated with NMR implemented at the gate level. As error rates scale, more options will be needed by logic designers to combat them.

Soft errors will arise due to a higher susceptibility to radiation (alpha and neutron), temperature (environment), power supply and ground noise, and electromagnetic interference among others [4,5,6]. The problem of dealing with these errors has traditionally been a problem for memory designers, however recent studies have indicated that within the next decade the problem of soft errors could be as great in logic as it is in memory now [3]. This will require designers to consider hardware solutions.

3MR is one of the oldest forms of fault-tolerance (FT) [7] . It is also one of the easiest to implement and to test. 3MR is achieved by using three redundant components to compute the same function as the original core component. The outputs of the redundant units are then combined and a consensus is arrived at by the use of a voting circuit or majority gate. Typically there is one voter per output. 3MR is fault-tolerant because it can generally tolerate one error in the inputs of the voter, per voter.

NMR is the generalization of 3MR, where N is usually odd. NMR has the capability to tolerate (N-1)/2 errors, with some exceptions[8]. We have chosen to examine NMR because it is one of the simplest forms of FT. It also serves a good basis for comparing other FT methods. Further it is relatively simple to adapt the implementation level from systems level to gate-level. This is an important consideration as increasing SERs may reduce or eliminate the effectiveness of system level solutions.

The rest of this paper is organized as follows: a brief review of relevant work is presented, followed by our methodology and simulation techniques, the paper is concluded with a presentation of our results, discussion of those results and concludes with a summary of current and future work.

2 BACKGROUND

2.1 Motivation and Related Works

A large body of work is emerging concerning the impact that soft errors will have on future computing devices [1-6],[9-13]. This research ranges from studying the causes such as cosmic rays [5] to the impact they will have on computer architectures [14]. The emerging view is that soft errors are going to be a serious problem for future computer designers and soft errors in logic will exceed those in memory, requiring fault tolerant techniques [3].

Many hardware FT techniques have been proposed to address these soft error increases [7,11,12], [15-23]. These range from hardware redundancy methods --such as NMR, NAND-multiplexing, and Duplication-- to informational redundancy –such as Error Detection Codes, and Error Correcting Codes—to temporal redundancy –such as recomputed and roll-back schemes. Studies such as [11,12] suggest that these system level methods may no longer be effective in the nanoscale due to increasing SER. We will

compare our gate-level implementations to system level NMR initially and we hope to be able to compare these methods across a broad range of these FT techniques.

Gate-level techniques have a long history in reliability studies. Non-voting methods such as NAND-multiplexing [7], quadded logic [24], interwoven logic [25], triply interwoven [23], fault-tolerant gates [19] have also been proposed. Bhaduri and Shukla [26] have investigated 3MR and Cascaded 3MR at the gate level. We begin by extending this work from 3MR to 9MR with voters, and model the error-reduction available from such methods, while Bhaduri and Shukla were more concerned with granularity trade-offs in a hierarchical system.

For this study we are specifically concerned with the work of Samudrala et al., Mohanram and Touba, and Bhaduri and Shukla[15-17,26]. The first two groups represent work towards partial NMR implemented at a level below the system level and the last using gate level 3MR. The first two use a system of determining gate error sensitivity to select candidates for NMR application. All groups limited their investigations to 3MR. We use a modified version of sensitivity to compare our results with the results of Mohanram and Touba. Our work is not directly compared to the method of Samudrala et al. because their work did not include error injection into voting circuits. We do plan on adapting their 3MR selection algorithm and voter reduction method for future comparisons.

2.2 Definitions

In this paper we use the terms errors, events, faults, and failures. Error is meant to denote an incorrect logical value, a result. A fault is the cause of that error, the event or transient. A failure then represents a specific type of error, namely an error occurring at one or more of the primary outputs for the circuit under test.

The core circuit refers to the circuit without any fault-tolerance applied.

3 METHODOLOGY

Our primary goal was to determine the effectiveness of NMR at the gate-level. In this investigation we examine NMR for N=3,5,7,9. The specific implementations are discussed in the next section, followed by the simulation methodology.

3.1 NMR Implementations

Traditionally NMR is implemented at the system level. Gate level implementation implies that the NMR is occurring for individual gates. For example, if 3MR were applied at the gate level, each individual gate (or circuit element) would be replicated 3 times with a voter to determine the gate's output.

We examined four implementations for NMR at the gate level for comparison. The first type is fully implemented. This type uses NMR on every gate in the original circuit. The second through fourth types are all partial implementations, meaning only a fraction of the total gates had NMR applied.

The second type of NMR implementation is based on error sensitivity. Sensitivity is measured by analyzing what fraction of gate-output errors translated into circuit output errors. A higher sensitivity means that the gate is more likely to cause a circuit error when an error is injected into the gate.

The third type is based on gate depth from a primary output. Our preliminary studies showed that depth from an output and sensitivity were usually inversely related; the greater the depth the less sensitive the gate. Depth was chosen as a selection criterion because it was often easier to measure than gate sensitivity. A gate with an output corresponding to a primary circuit output is considered to have a depth equal to 0. The shortest path to a primary output is considered to be the depth for a gate. Depth was measured in number of gates.

Finally, the fourth type is based on random selection. We wanted to understand if there was a relationship between error reduction and implementation or simply the amount of redundancy used.

3.2 Voting Logic

Voting circuits used with NMR are designed to obtain a consensus among inputs and produce an output(s). The most straight-forward way to derive such a circuit is to use a two level NAND circuit where the first level is comprised of all possible combinations of (N+1)/2 =m, inputs. The second level is a NAND-gate comprised of N!/[(N-m)!m!] inputs corresponding to the total number of possible combinations. This circuit is easily designed using 3-NAND2 gates and a NAND3 gate for N=3. When N=9, there are 126 different combinations do deal with and it would be unrealistic to use a NAND126 gate for the second level of our voter. For this reason our voter circuits were developed using the Synopsys Design Compiler and finding equivalent circuits to the NAND specified circuits for N=5,7, and 9.

4 SIMULATION

We simulate using a fault-injection environment using Verilog. There are 5 steps in our simulation. The first step is the circuit selection. Input sets are chosen second. The check-value set is then determined. The fourth step is to choose a FT implementation. Finally we simulate with error injection over a range of FT techniques and implementations. Each step is explored briefly in the following section.

4.1 Specific Methodology

We start with the circuit descriptions provided with the LGSynth93 benchmarks. We have selected a subset of the combinational logic circuits based on gate count and related relevant research by others [15-17]. We have translated those circuits into Verilog descriptions.

The input sets are comprised of a number of input vectors. These vectors represent a single set of inputs to a circuit. The input vectors are chosen from either a typical workload, if such information is available, or generalized data can be generated using distribution assumptions. For this investigation we assume uniform distribution of input values over a generalized input set of 1000 to 100000 vectors.

Each input set is also looped over a set number of times. A loop represents simulation over the entire input set. Experiments consist of a minimum of 200000 individual trials. (i.e. at 1k input set, 200 loops would be used.)

The values for input set size and loop size were chosen by setting values to ensure a variance of less than 2\% at 99\% confidence. These calculations are based on maximum total variance values (i.e. at 50\%).

After setting input set size and loop size, each input set must also have a corresponding check-value set. A check-value set represents the values of the outputs for the circuit corresponding to the inputs. These check-values are determined and checked using the circuits themselves, run in a simulation without any faults injected.

To implement FT, we create a special library for each instance of FT to be tested (in this case 3, 5, 7, and 9 MR). We make FT libraries for both full and partial NMR. These libraries are used to replace all gates in the FNMR case and selected gates in the PNMR case. The implementations are described above.

After the implementation has been selected, the experiments can be simulated. Each experiment consists of multiple trials. Each trial is considered to be a single cycle of the circuit. A cycle consists of applying inputs and observing the outputs. During each cycle, a single SEU is injected. The overall order is: Inputs are applied, a SEU is injected, and then outputs are monitored. Any difference from the check-values and the monitored output is recorded as an error.

A multiple-bit error in an output is counted as a single error.

The number of errors is recorded and averaged for each circuit. This represents the base number of errors that each FT method and implementation will be compared against.

After a base error number is recorded, each FT method and implementation is simulated individually on each circuit. An example would be to implement TMR fully over each of the circuits chosen. We simulate the fully implemented TMR on each circuit in the same way the base case was done. At the end we have a number representing the total number of faults injected, the total number of errors, and the total number of inputs tested. Based on these numbers we then calculate the percentage difference between the FT case and the base case. Each plotted data point in figures 1-4 represents a single experiment for a single circuit, over the entire input set.

5 ANALYSIS

5.1 Bounding Model

We began by constructing a bounding model using some fairly optimistic assumptions. This is useful for comparing similar methods. These assumptions are:

- Our error model assumes high or low pulses injected into the outputs of gates.
- Circuit inputs cannot be faulty.
- We inject a single pulse every trial. Experiment is considered to be 200k independent single error trials.
- We assume inputs are uniformly distributed.
- All gates are equally likely to have a pulse injected
- Majority gates are treated as a single gate. This includes overhead and injection.
- Overhead is measured in gate counts.

It became quickly apparent that these assumptions are too optimistic. An improved model was made relaxing these assumptions and is described in the next section. This optimistic model is intended to act as an upper bound on what is capable using NMR implemented at the gate-level.

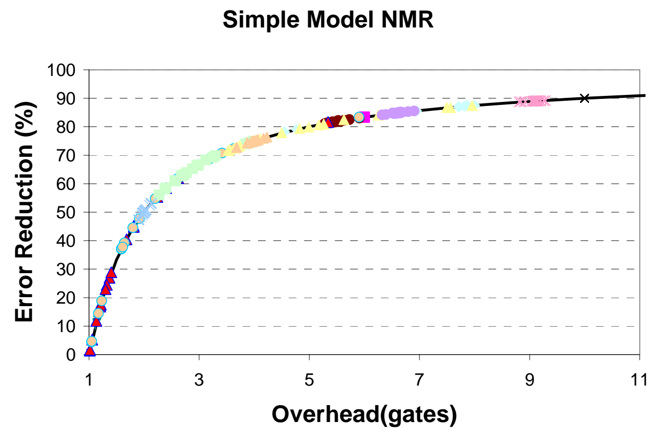

Figure 1- All four gate-level NMR implementations plotted along with model, using the assumptions of the simple model

As we can see in figure 1, using NMR at the gate level gives an upper bound of circuit error reduction equal to

$$\text{Error Reduction} = 1 - 1/\text{overhead}. \quad (1a)$$

$$y = M*(A-B/x) \quad (1b)$$

Equation (1b) is the general for equation (1a), the model for error reduction. This shows us that Error Reduction and overhead are inversely proportional. The graphs in figure 1 are generated using values of M = 100%, A = 1, and B = 1. B represents the x intercept, and M*A represents the upper bound, in this case 100% error reduction, A is the fractional limit and M is the scaling factor. When B = 1 this tells us that we expect 0% error reduction at an overhead = 1, for B = 2, this would tell us that we would need an overhead of 2 before

we saw any benefit from this FT implementation. This is expected behavior. An overhead of 1 would be the core circuit without redundancy. An overhead greater than 1 implies some level of redundancy, such as an overhead of 2 equals a circuit with twice as many gates as the core circuit. With B = 1 we see an immediate reduction of errors. If B > 1, then we would see some increase in errors before the overhead exceeded the B value. If M*A < 100%, this would indicate an upper bound on the maximum reduction of errors possible using these FT methods.

For the simplified models, we see that there is no difference in implementation in error reduction, figure 1. The amount of reduction is inversely proportional to the overhead.

When we relax the assumptions listed above, specifically the last three assumptions, we get a different picture, and begin to see some differentiation between the various implementations.

5.2 Revised Model

After re-conducting the experiment with more realistic assumptions we arrive at a new model. Specifically these changes included:

- Errors are distributed by area, not uniformly
- Majority gates were constructed and each gate within the majority circuit was able to have injections
- Overhead was measured in area. Area based on static CMOS standard layout.

Modifying these assumptions gives us a more realistic picture of how overhead and reliability are related at the gate-level implementation for NMR. Overhead calculations are now based on area. Because of the dependency on specific technology libraries we have opted to use generic static CMOS layout information.

Using equation (1b) as our model we find that we can no longer set B = 1 as a best fit. This implies that there is some region of adding overhead that decreases our circuits reliability rather than increasing it.

In figure 2 we see the data for sensitivity based NMR plotted. B = 1.84, A = 1. This implies that for any overhead < 1.84 we will see a net decrease in reliability.

For the depth based NMR implementation we see in figure 3 that B = 1.125, A=1.

Fully implemented NMR is fitted with B = 2.55, A=1 . This is by far the worst performing implementation.

Finally our NMR implemented randomly presented the need for two models. The first model represents the case of N=3 and the second for N=5, 7, and 9. For N=3, A = .75, B =1.35. For N=5, 7, and 9, A=.97 and B=1.35. We see that for N=3, A has a much smaller value, implying a much less efficient implementation.

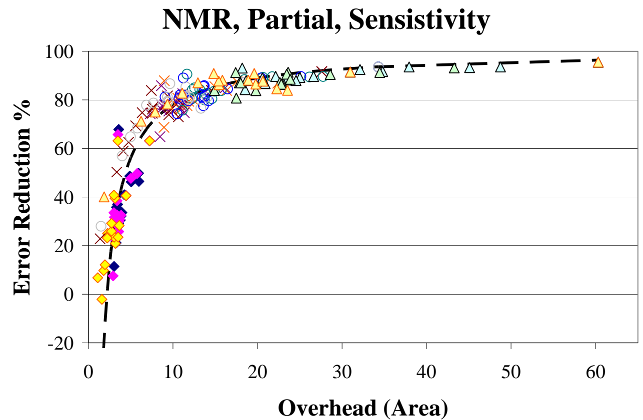

Figure 2 – The second model, showing NMR implementation. A=1, B=1.84.

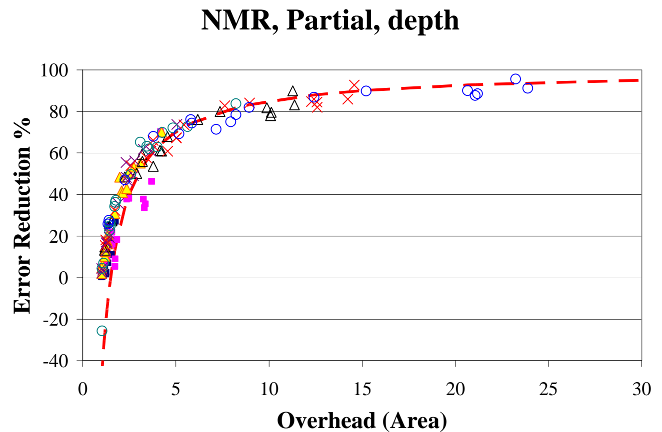

Figure 3 –NMR implemented by Depth.A=1, B=1.125

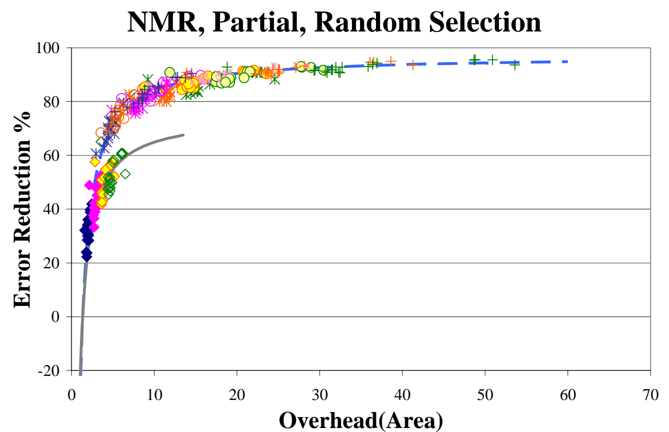

Figure 4- NMR implemented randomly. For the N=5,7,9(top) A= .97, B=1.35. For N=3(Bottom), A=.75, B=1.35

5.3 Comparison

When we compare these models with system level and

sub-circuit level, we see that gate level NMR performs the worst of the three in terms of error reduction. Specifically randomly implemented 3MR and fully implemented NMR perform much worse than the other implementations. System level NMR performs the best in terms of error reduction. Mohanram and Touba's method performs the best for efficiency and overhead.

In figure 4 we see the X's representing the error reduction by using system level NMR are all clustered around 99.9% error reduction. We see the results of Mohanram and Touba for sub-circuit partial 3MR shown as 0's. The remaining points on the graph represent the various gate-level implementations from Figures 2-4.

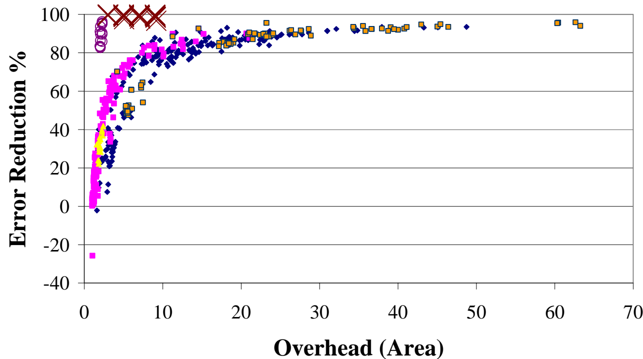

Figure 5- A comparison of system, sub-system, and gate level implementations of NMR. X's are System level across the top in the upper left. 0's in the vertical line in the upper left represent the results of Mohanram and Touba (DFT2003) using sub-circuit NMR.

5.4 Discussion

We have identified nearly 20 unique FT methods that are applicable at the sub-circuit, gate, or device level. For this investigation we were primarily concerned with analyzing the effectiveness of NMR at the gate-level. For a more complete picture of where these implementations should be placed, it will be necessary to implement and compare these additional methods. These include methods with improved voter circuits, quadded and interwoven logic, cascaded logic, selective duplication, using less sensitive gates, using alternative technologies, multiplexing, comparison, informational redundancy (e.g. error correcting codes), temporal redundancy (e.g. recomputed and compare), and duplicating these methods with cones or chains of gates.

As these implementations are developed further, as a basis of comparison with other methods or as a FT technique, it will become important to consider validation and testing requirements as well.

Power consumption and performance metrics will also be needed to give a more complete and reasonable comparison between the various techniques.

6 CONCLUSION

6.1 Future Work

It is important that we explore the effectiveness of these methods under error scaling, (i.e. scaling error rates according to predicted technology rates and proportional to overhead). It has been suggested that this scaling will reveal the inability of system level NMR to cope with the error rates, where gate-level may succeed.

We are also investigating a number of other FT implementations, such as NAND multiplexing, and modifications to NMR, such as voter reliability improvement.

6.2 Conclusion

We have shown an upper bound to error reduction using gate level NMR is:

$$\text{Error Reduction} = 100\% * (1 - 1/\text{Overhead}). \quad (2a)$$

And generally:

$$\text{Error Reduction} = 100\% * (A - B/\text{Overhead}). \quad (2b)$$

These are restatements of equation 1b. We have also shown the overheads associated with using these implementations to be greatly in excess of system level NMR.

We have shown a mathematical model for describing the relationship between overhead and error reduction under single-error injection conditions in combinational logic using NMR based fault-tolerance methods at the gate level. Additionally, we have provided an analysis for the model under more realistic circuit and overhead assumptions and how the simple model can be adjusted.

When compared to system level or sub-circuit level NMR, gate-level offers greater freedom of overhead with reduced and bounded error reduction effectiveness.

ACKNOWLEDGEMENTS

This work was supported in part by Semiconductor Research Corporation contract no. 2004-HJ-1190, National Science Foundation grant CCR-0210197, and additional support from IBM, Intel, the University of Minnesota Digital Technology Center, and the Minnesota Supercomputing Institute.

REFERENCES

1. C. Constantinescu, "Trends and Challenges in VLSI circuit reliability", IEEE MICRO, 23(4), 2003, 14-19.
2. C. Constantinescu, "Impact of Deep Submicron Technology on Dependability of VLSI circuits", IEEE International Conference on Dependable Systems and Networks, 2002.
3. P. Shivakumar, M. Kistler, S.W. Keckler, D. Burger, and

L. Alvisi, "Modeling the effects of technology trends on th Soft Error Rate of Combinational Logic", IEEE International Conference on Dependable Systems and Networks, 2002.
4. P. Dodd, "Basic Mechanisms and Modeling of Single-Event Upset in Digital Microelectronics", IEEE Transactions on Nuclear Science, 50(3), 2003, 583-602.
5. J. F. Ziegler, "Terrestrial Cosmic Rays", IBM Journal of Research and Development, 40(1), 1996.
6. E. Normand, "Single-Event Effects in Avionics", IEEE Transactions on Nuclear Science, 43(2), 1996, 461-474.
7. J. von Neumann, "Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components", Automata Studies, 1953, Princeton University Press.
8. J. Biernat, "The Effect of Compensating Faults Models on NMR System Reliability", IEEE Transactions on Reliability, 43(2), 1994, 294-300.
9. P. Liden, P. Dahlgren, R. Johansson, J. Karlsson, "On Latching Probability of Particle Induced Transients in Combinational Networks", International Symposium on Fault-Tolerant Computing, 1994, 340-349.
10. D. Alexandrescu, L. Anghel, M. Nicolaidis, "New Methods for Evaluating the Impact of Single-Event Transients in VDSM ICs", International Symposium on Defect and Fault Tolerance in VLSI systems, 2002.
11. N. Rollins, M. Wirthlin, "Evaluating TMR Techniques in the Presence of Single Event Upsets", International Conference on Military and Aerospace Programmable Logic Devices, 2003.
12. S. Krishnamohan, N. R. Mahapatra, "Combining Error Masking and Error Detection Plus Recovery to Combat Soft Errors in Static CMOS Circuits", International Conference on Dependable Systems and Networks, 2005, 40-49.
13. A. Maheshwari, I. Koren, W. Burleson, "Accurate Estimation of Soft Error Rate (SER) in VLSI Circuits", International Symposium on Defect and Fault Tolerance in VLSI Systems, 2004.
14. S. Mukherjee, J. Emer, S. Reinhardt, "The Soft Error Problem: An Architectural Perspective", International Symposium on High-Performance Computer Architecture, 2005, 243-247.
15. K. Mohanram, N. A. Touba, "Cost-Effective Approach for Reducing Soft Error Failure Rate in Logic Circuits", International Test Conference, 2003, 893-901.
16. K. Mohanram, N. A. Touba, "Partial Error Masking to Reduce Soft Error Failure Rate", International Symposium on Defect and Fault-Tolerance in VLSI Systems, 2003, 433-440.
17. P.K. Samudrala, J. Ramos, S. Katkoori, "Selective Triple Mondular Redundancy (STMR) Based Single-Event Upset (SEU) Tolerant Synthesis for FPGAs", IEEE transactions on Nuclear Science, 51(5), 2004, 2957-2969.
18. D. D. Thaker, R. Amirtharajah, F. Impens, I. L. Chuang, F.T. Chong, "Recursive TMR: Scaling Fault Tolerance in the Nanoscale Era", IEEE Design and Test of Computers, 22(4), 2005, 298-305.
19. Y. Takefuji, M. Ikeda, "A Novel Approach to Fault-Tolerant Logic", Journal of Information Processing, 3(3), 1980.
20. K. Nikolic, A. Sadek, M. Forshaw, "Architectures for Reliable Computing with Unreliable Nanodevices", IEEE Conference on Nanotechnology, 2001, 254-259.
21. A. E. Barbour, A. S. Wojcik, "A General, Constructive Approach to Fault-Tolerant Design Using Redundancy", IEEE Transactions on Computers, 38(1), 1989, 15-29.
22. I.L. Yen, I. Ahmed, R. Jagannath, S. Kundu, "Implementation of a Customizable Fault Tolerance Framework", International Symposium on Object-Oriented Real-Time Distributed Computing, 1998.
23. J. Han, J. Gao, P. Jonker, Y. Qi, J. A.B. Fortes, "Toward Hardware-Redundant, Fault-Tolerant Logic for Nanoelectronics", IEEE Design and Test of Computers, 22(4), 2005, 328-339.
24. J. Tryon, "Redundancy Techniques for Computing Systems", 1962, Spartan Books.
25. W. Pierce, "Interconnection Structure for Redundant Logic", in "Failure Tolerant Computer Design" 1965, Academic Press.
26. D. Bhaduri, S. Shukla, "NANOPRISM: A Tool for Evaluating Granularity vs. Reliability Trade-offs in Nano Architectures", Great Lakes Symposium on VLSI, 2004, 109-112.

BIOGRAPHIES

Drew C. Ness
Department of Electrical and Computer Engineering
University of Minnesota
EE/CSci 4-178
200 Union Street SE
Minneapolis, MN 55455-0167

e-mail: dness@ece.umn.edu

Drew C. Ness is a graduate student at the University of Minnesota, Twin Cities. He earned his bachelor degree in Physics at Hamline University. His research interests are fault-tolerance, quantum computing and novel computer architectures.

Christian J. Hescott
Department of Electrical and Computer Engineering

e-mail: research@hescott.com

Chris Hescott is pursuing his PhD in Electrical Engineering at the University of Minnesota. He received a MS and a BS degree in Computer Engineering. His interests include architectures and reliability techniques for current and future technologies.

David J. Lilja

To Appear RAMS 2007

Department of Electrical and Computer Engineering
EE/CSci 4-178
200 Union Street SE
Minneapolis, MN 55455-0167

e-mail: lilja@ece.umn.edu

David J. Lilja received a Ph.D. and an M.S., both in Electrical Engineering, from the University of Illinois at Urbana-Champaign, and a B.S. in Computer Engineering from Iowa State University in Ames. He is currently a Professor and Head of Electrical and Computer Engineering at the University of Minnesota in Minneapolis, where he also serves as a member of the graduate faculty in Computer Science and Scientific Computation, and as a Fellow of the Minnesota Supercomputer Institute. Previously, he worked as a research assistant at the Center for Supercomputing Research and Development at the University of Illinois, and as a development engineer at Tandem Computers Incorporated in Cupertino, California.