MRF Reinforcer: A Probabilistic Element for Space Redundancy in Nanoscale Circuits

SHRINKING DEVICES TO NANOSCALE, INCREASING INTEGRATION DENSITIES, AND REDUCING VOLTAGE LEVELS TO THE THERMAL LIMIT—ALL CONSPIRE TO PRODUCE FAULTY SYSTEMS. A POSSIBLE SOLUTION IS A FAULT-TOLERANT PROBABILISTIC FRAMEWORK BASED ON MARKOV RANDOM FIELDS. THIS ARTICLE INTRODUCES A NEW REDUNDANCY ELEMENT, THE MRF REINFORCER, WHICH ACHIEVES SIGNIFICANT IMMUNITY TO SINGLE-EVENT UPSETS AND NOISE.

Kundan Nepal
R. Iris Bahar
Joseph Mundy
William R. Patterson
Alexander Zaslavsky
Brown University

As silicon CMOS devices scale down into the nanoscale domain, current microarchitecture approaches are reaching their practical limits. So far, the semiconductor industry has successfully overcome many hurdles by miniaturizing transistors with advanced lithography, introducing new materials (silicon-on-insulator currently, high-κ and low-κ dielectrics in the near future), reducing supply voltages, and increasing wiring levels. Technological advances keep pushing the day of reckoning into the future—the latest *International Technology Roadmap for Semiconductors* extends technology scaling to 2018 (http://www.public.itrs.net). Nevertheless, physical constraints, as well as power dissipation limits and commodity manufacturing economics, will eventually curtail the CMOS scaling paradigm.

Alongside the massive research effort supporting silicon CMOS downscaling, many researchers are pursuing nonsilicon devices. Researchers hope to integrate such alternative devices with CMOS, resulting in hybrid systems combining CMOS-based digital logic with analog, optical, or quantum devices.

Although no clear consensus exists on how far and how fast CMOS will downscale and which emerging hybrid technology will eventually enter production, it is certain that future nanodevices will have high manufacturing defect rates. Furthermore, it is clear that manufacturers will aggressively scale down supply voltage to reduce dynamic power dissipation. A V_{DD} of 0.5 V is the current *ITRS* prediction for low-power CMOS in 2018, but extrapolations to an even lower 0.3 V have appeared in the literature. The resulting reduction in noise margins will expose computation to higher soft-error rates.

Obviously, future circuit designers won't be able to assume error-free operation. Researchers have proposed numerous methods for improving circuit reliability in the presence of increased soft-error rates. Some recent microarchitectural approaches exploit thread-level redundancy.[1,2] Although these approaches have promise, we've chosen to

0272-1732/06/$20.00 © 2006 IEEE Published by the IEEE Computer Society

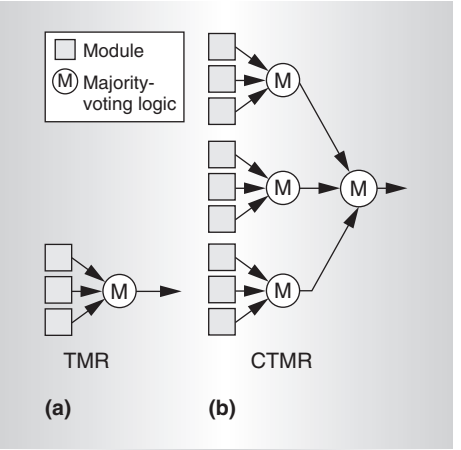


Figure 1. Triple-modular redundancy (a) and cascaded triple-modular redundancy (b) elements.

focus on finer-grained redundancy. Traditional approaches that attempt to increase reliability against soft errors by employing space redundancy include NAND multiplexing, triple-modular redundancy,[3] N-modular redundancy, and cascaded triple-modular redundancy.

Triple-modular redundancy (TMR) uses three copies of the same module working in parallel. The modules send their outputs to majority-voting logic. The modules can be single gates, logical blocks, or functional units, depending on the amount of error tolerance the system requires. TMR provides relief from an error caused in a single unit, but errors in two or more of the three modules will cause the logic to fail. Another method, cascaded TMR (CTMR), repeats the TMR process by taking each TMR module as a single module, making three copies, and then combining the outputs of the individual TMR units with another majority gate. A TMR module can be considered a 0th-order CTMR module. Both methods greatly increase system reliability in the presence of transient faults. However, the main assumption underlying these redundancy processes is that the final majority-voting gate is perfect and free from the failures faced by the rest of the circuitry. Figure 1 shows the basic TMR and CTMR structures.[4]

Several studies have evaluated a TMR system's loss of reliability under different assumptions about error sources. Lyons and Vanderkulk, using Monte Carlo simulations, find that imperfect voting circuitry significantly affects a TMR computer's reliability.[3] Pippenger shows that with noise in a system, reliable computation requires higher levels of redundancy.[5] Favalli and Metra show that the effect of crosstalk faults at voter inputs impairs both the voting and the system's reliability.[6]

Inasmuch as nanoscale technology can no longer guarantee error-free logic states, a possible solution is a paradigm shift to a fault-tolerant probabilistic framework based on Markov random fields.[7] The MRF framework represents each input or output as a random variable whose value can vary over the range of the logic signal level between 0 V and V_{DD}. In this representation, unlike the classical case, there is no notion of a correct logic signal at a given node at all times. Instead, the framework determines logic states through observation of the probability distribution of the signal values, and the correct states are those that maximize the joint probability distribution of all the logic variables, given observed logic signals. Ultimately, redundancy in time and space reduces to an acceptable level the error of choosing the most probable answer.

We presented the mapping of the MRF-based probabilistic framework to modified CMOS-based circuitry in an earlier work.[8] We also showed that operation at subthreshold operation (V_{DD} = 0.15 V) is viable for reliable computation at reduced dynamic power levels. Here, we extend that approach by creating a new MRF element for handling soft errors and single-event upsets in sequential logic. The idea is to reinforce the most probabilistically correct state and prevent the logic from taking on erroneous values. Our proposed mechanism is a feedback-based reinforcer that can operate without requiring that any part of the circuit be free from soft errors or upsets.

MRF reinforcer: Theory

The MRF defines a set of random variables called *sites*, $X = \{x_1, x_2, \ldots, x_k\}$, where each variable x_i can take on various values called *labels*. As Figure 2 shows, the sites in X are related to one another through a neighborhood system defined by a set of variables from $X - \{x_i\}$. The probability of a particular site in the neighborhood depends only on its immediate neighbors, to which it is connected by an edge. The edge of the neighborhood represents the

conditional dependence between connected variables in the neighborhood. In terms of logic circuitry, these nodes represent the inputs and outputs of the circuits all treated on an equal footing, and the cliques represent the dependencies between the nodes. The conditional probability of a given site in terms of its neighborhood can be formulated in terms of the graph structure's associated clique. Figure 2 shows one such neighborhood with one first-order clique and one second-order clique.

Using the Hammersley-Clifford theorem,[9] we write the conditional probability distribution as

$$P\left(x_i \left| \left\{ X - x_i \right\} \right.\right) = \frac{1}{Z} \sum_{c \in C} e^{\frac{-U(x_c)}{kT}}$$

Where X is the set of all nodes in the neighborhood, C is the set of cliques, x_c is the set of nodes in clique c, and $U(x_c)$ is the clique energy function, also called the logic compatibility function. Z is the partition function, a constant required to normalize the probability function to [0, 1]. The term kT can be interpreted as thermal energy from the physical point of view, but here we treat it merely as a constant in proportion to the clique energy that controls the probability distribution's sharpness. This form of the distribution is called the Gibbs distribution. The Gibbs representation provides an attractive platform for applying the MRF concept to computation.

Figure 3 shows a simple multilevel circuit and its corresponding dependence graph. In this case, the graph is equivalent to an MRF in which nodes are random logic variables that can hold values ranging from 0 V to V_{DD}, and edges are conditional dependencies between variables. All the logic variables in the example (s_0, s_1, s_2, s_3, s_4, s_5) vary in a random manner over the range of logic signal levels. The correct logic states are those that maximize their joint probability; that is, the correct logic operation for the example corresponds to the variables that maximize $p(s_0, s_1, s_2, s_3, s_4, s_5)$. The graph in Figure 3 contains three distinct sets of cliques (sets of fully connected node subsets): $\{s_0, s_1, s_3\}$, $\{s_2, s_3, s_4\}$, and $\{s_4, s_5\}$. These cliques represent the local statistical dependencies of the logic states.

We based the reinforcer on the Gibbs MRF formulation. Consider a simple three-node network s_0, s_1 and s_2, in which we want each node to maintain a given logic value. As Figure 4 shows, in the ideal case, in which all three nodes store either a logic one or a logic zero state, the gate's successful operation is designated by compatibility function $f(s_0, s_1, s_2)$.

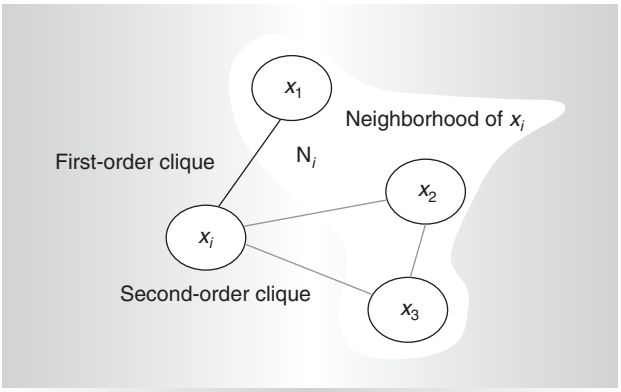


Figure 2. The MRF neighborhood system.

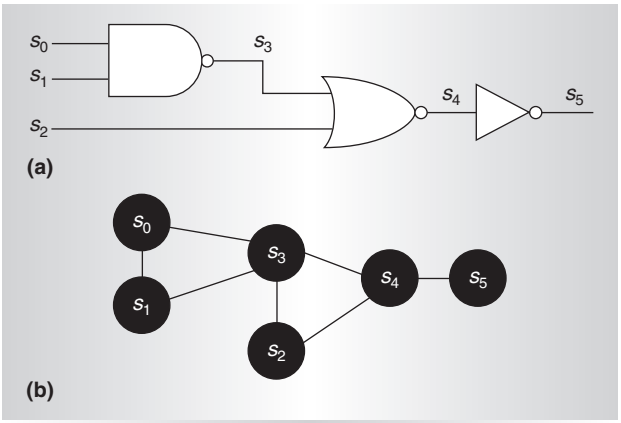
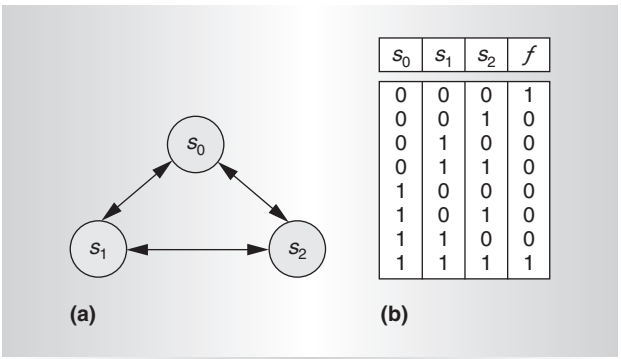


Figure 3. A logic circuit (a) and its dependence graph (b) for a simple Markov random field.



s_0	s_1	s_2	f
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Figure 4. Three-state reinforcer (a); logic compatibility function with all possible states (b).

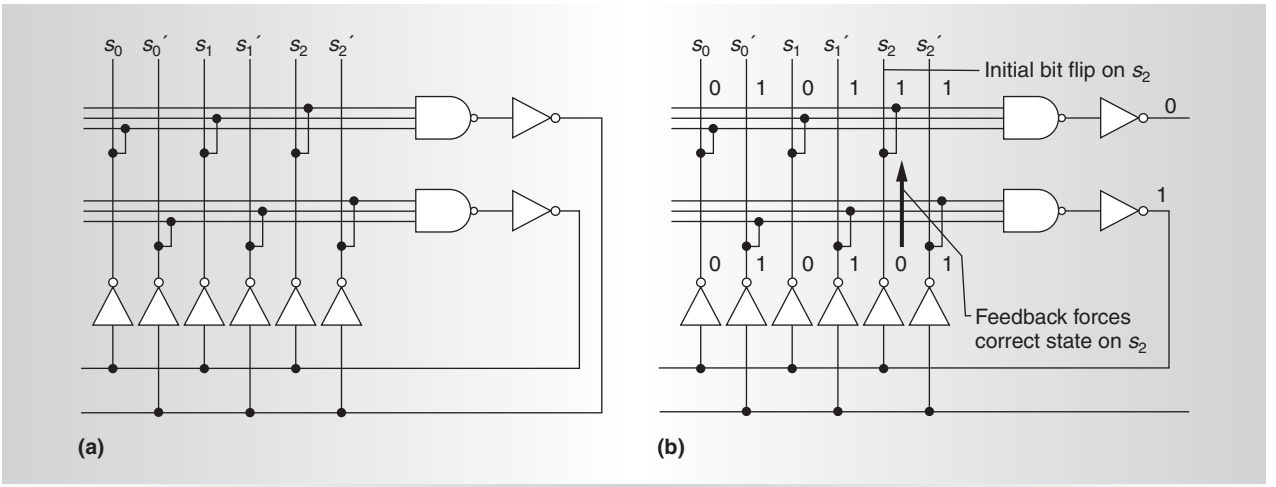


Figure 5. A circuit encoding the MRF reinforcer's clique energy function (a). The MRF reinforcer corrects the value (indicated by an arrow) at node s_2, which initially deviated from its correct state (b).

Figure 4b lists all possible states: valid states with $f = 1$ and invalid states with $f = 0$. We obtain the logic compatibility, or the clique energy,[7] of the function by summing over all valid states: $U_c = (s_0's_1's_2' + s_0s_1s_2)$. Using this logic compatibility function, our goal is to create a CMOS mapping that will maximize the joint probability of the three states corresponding to the valid logic operation.

CMOS mapping of MRF reinforcer

MRF logic elements are probabilistic: Inputs and outputs are on an equal footing, and the logic element is designed to maximize the joint probability of the correct I/O logic values. Mapping the MRF model to CMOS circuitry requires the following:

- Each logic state x_i should be represented as a bistable storage element, taking on logic values of 0 and 1. The probability of any other signal value should be low.
- The constraints of each logic graph clique should be enforced by feedback to the appropriate storage elements, implementing the Gibbs distribution functions to factorize and maximize the joint probability of the correct logic values.

The necessary feedback from output to input inevitably complicates the logic gate and increases the device count. On the other hand, recent simulations on combinational circuit elements and benchmark circuits demonstrate that such elements can operate correctly at supply voltages so low and with input signal noise so high as to make standard CMOS gates completely inoperable.[8]

For combinational circuits, we can enforce this feedback notion by realizing the relationship between inputs and outputs of each gate or function. For example, consider a NAND gate with input variables x_0 and x_1 and output variable x_2. A legal variable assignment of $x_0 = 1$, $x_1 = 1$, and $x_2 = 0$ will reinforce the complemented variable $x_2' = 1$. An illegal assignment of input and output variables will fail to reinforce any variables.

A static combinational gate always has available input and output variables, so we can easily construct a compatibility function or logic graph clique based on the constraints between input and output, thus exploiting causality when generating feedback. This is the basic idea behind our earlier approach.[8] However, in sequential logic, where signals latch into memory elements, this notion of causality is lost at the clock edge boundary, so we need a somewhat different approach. To reinforce correct state behavior for sequential logic, we resort to state duplication to create constraints for a logic graph clique. Now what we reinforce through feedback is the dominant state assignment's value.

Following the recipe for mapping MRF networks into CMOS structures,[8] we can now create a programmable logic array (PLA) structure for the reinforcer, using CMOS logic

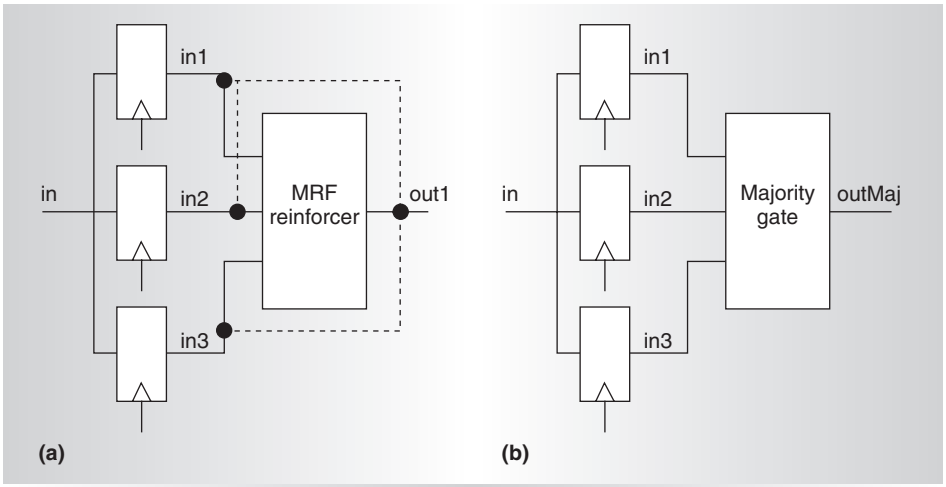


Figure 6. Simulation setup for comparison between MRF reinforcer (a) and CMOS majority gate (b). The reinforcer is allowed to act on the latched data when the clock signal is low.

gates. Figure 5 shows the MRF implementation of the reinforcer.

The circuit consists of three storage nodes, one each for s_0, s_1, and s_2. The nodes' stable states correspond to the variables' maximum probability configurations (when all nodes are storing the same value). For example, suppose that $s_0 = s_1 = s_2 = 0$ and their complements $s_0' = s_1' = s_2' = 1$. The top NAND-inverter gate outputs a 0 and feeds the logic state 1 back to the complemented inputs through the feedback inverters, thereby reinforcing the expected output value. The other NAND-inverter gate feeds back a logic state 0, also through the feedback inverters. These feedback values are consistent with input values $\{s_0, s_1, s_2\}$, and the overall circuit latches into this state. The other configuration, $s_0 = s_1 = s_2 = 1$, corresponding to the other valid logic state, is also stable.

On the other hand, in the same nodes, let's assume that all nodes stay at the same value as the previous example, but by virtue of a single-event upset node, s_2 rises to an intermediate value or even all the way to logic 1. The state of the nodes at that moment would be s_0, $s_1 = 0$, $s_2 = 1$, and $s_0' = s_1' = s_2' = 1$. On the basis of the compatibility function and the node interaction shown in Figure 4, this is an invalid state. The invalid node combination causes the top NAND-inverter pair to output a 0, while the bottom NAND-inverter pair outputs a 1. The feedback inverters then complement these values and feed a value of 0 to nodes s_0, s_1, and s_2, and a 1 to the other complement nodes. This forces the entire circuit to latch a valid combination of the storage nodes, as Figure 5b shows.

Note that the MRF circuit feedback is in contention with the initial circuit inputs. Proper sizing of the feedback inverters allows the feedback from the MRF circuit to contend with and overpower the node's incorrect value. Also, if one of the nodes deviates from its stable ideal 0 and 1 states and goes to an intermediate value, the reinforcer drives it back to the correct logic state.

Simulation results

To measure the MRF reinforcer's effectiveness, we compared its fault tolerance and noise immunity with those of a traditional majority gate. For our simulations, we first triplicated a single signal source and stored it in three different flip-flops. We then passed the flip-flop outputs through a traditional majority gate. We later repeated the same simulation, this time sending the flip-flop outputs to the MRF reinforcer. All devices were subject to faulty conditions. If a dynamic fault occurred in any device or a single-event upset occurred in the storage elements, the reinforcer corrected the errors when the clock was low (when new values cannot propagate from the input to the output of the flip-flop). Figure 6 shows the simulation setup.

We simulated the circuits using Spice and the 70-nm Berkeley predictive technology model (http://www-device.eecs.berkeley.edu/~ptm/), with V_{DD} at 0.15 V and temperature

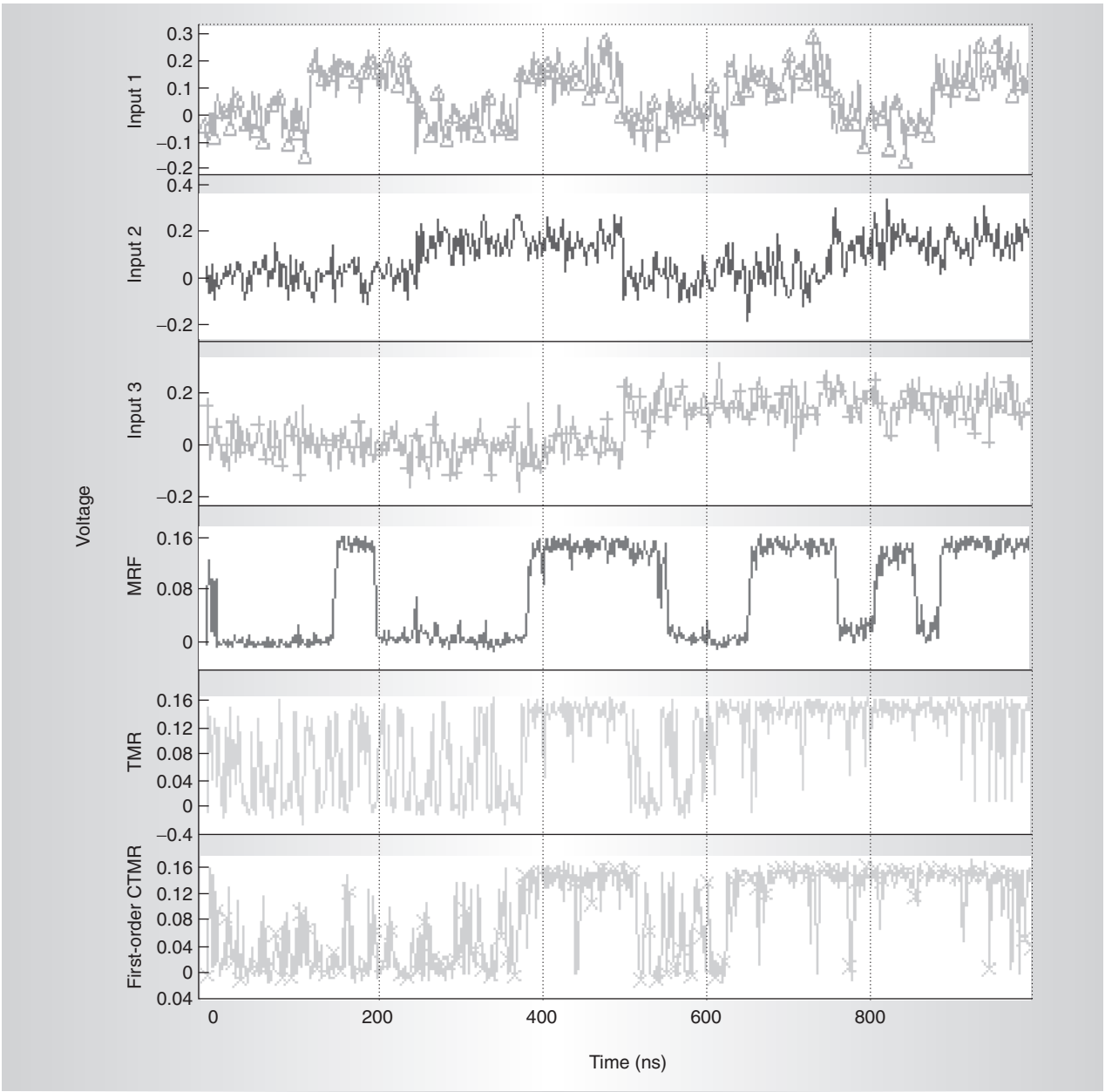


Figure 7. Standard CMOS majority gate and MRF reinforcer operation at subthreshold supply voltage. The MRF reinforcer output is stable, whereas the TMR and first-order CTMR switch between correct and incorrect output values.

at 100°C. Using such a low V_{DD} has two benefits: It allows us to show the advantage of probabilistic computation in ultimate CMOS devices and to simulate future noise problems. With a supply voltage lower than the transistor's threshold voltage, we can capture the noise margin reduction caused by thermal noise, electromagnetic coupling, and hot-electron effects, as well as threshold variations.[10] We also used a Gaussian noise model for the simulations.[8] We sized all circuits with a PMOS to NMOS width ratio of two to achieve equal rise and fall times.

Figure 7 shows simulations of the MRF reinforcer, a standard majority-gate-based TMR, and a first-order CTMR. The simulation results are for all possible combinations of the three triplicated inputs. Under perfect conditions, all three inputs would be the same; however, in the presence of extreme noise and single-bit errors, any of the eight input cases

are likely. The figure shows responses for all the possible cases.

All three designs are equally good at handling single-event upsets in a noise-free environment, but only the MRF reinforcer performs well under noisy conditions. The noisy (uncorrelated) inputs cause the standard majority gate operated at subthreshold V_{DD} of 0.15 V to switch between correct and incorrect output values. The first-order CTMR shows an improvement over the simple TMR, but it doesn't come close to matching the MRF reinforcer's noise immunity and tolerance. A higher-order CTMR might provide even better tolerance to noise and soft errors. However, studies have shown that there is no advantage in using a higher-order CTMR when the majority gates consist of the same faulty devices as the units being monitored for errors.[4] Also, increasing the CTMR's order causes an explosion in overall area with limited additional benefit in noise immunity over the MRF reinforcer.

Our implementation of the MRF reinforcer gate provides stable and correct voltage operation at the cost of eight extra transistors compared with the standard majority gate. However, the CMOS mapping of the MRF reinforcer shown in Figure 5 is a PLA-type mapping and not necessarily the optimal implementation in terms of area.

To further compare the different approaches, we calculated the probability distributions of the ideal signal and of the TMR, first-order CTMR, and MRF reinforcer outputs. Figure 8 shows the distributions, calculated by sampling the output voltage at discrete points (every 0.1 ns). Analysis shows that the highest probability clusters occur around the stable output states, but the probability of being in the intermediate voltage ranges is nonzero for all three implementations. However, the probability of being in an incorrect state is much smaller for the MRF reinforcer than the TMR and first-order CTMR implementations.

To quantitatively compare logic elements' noise immunity, we used the Kullback-Leibler distance.[11] The KLD is a popular measure of discrepancy between the probability distribution of ideal output P_{ideal} and the probability distribution of the real noisy output P_{real}. For a digital system with two levels (0 and 1), the KLD is the measure of distance between P_{ideal} and P_{real} (where output is sampled and noise leads to some probability of finding an incorrect output value), calculated as follows:

$$KLD\left(P_{ideal}, P_{real}\right) = \sum_{states} P_{ideal} \log_2 \left(\frac{P_{ideal}}{P_{real}} \right)$$

Here, the smaller the KLD, the better the circuit's noise immunity. Table 1 compares KLDs of the standard CMOS TMR, the first-order CTMR, and the MRF reinforcer. The table also shows the average error rate of the three designs. We calculated the error rates by

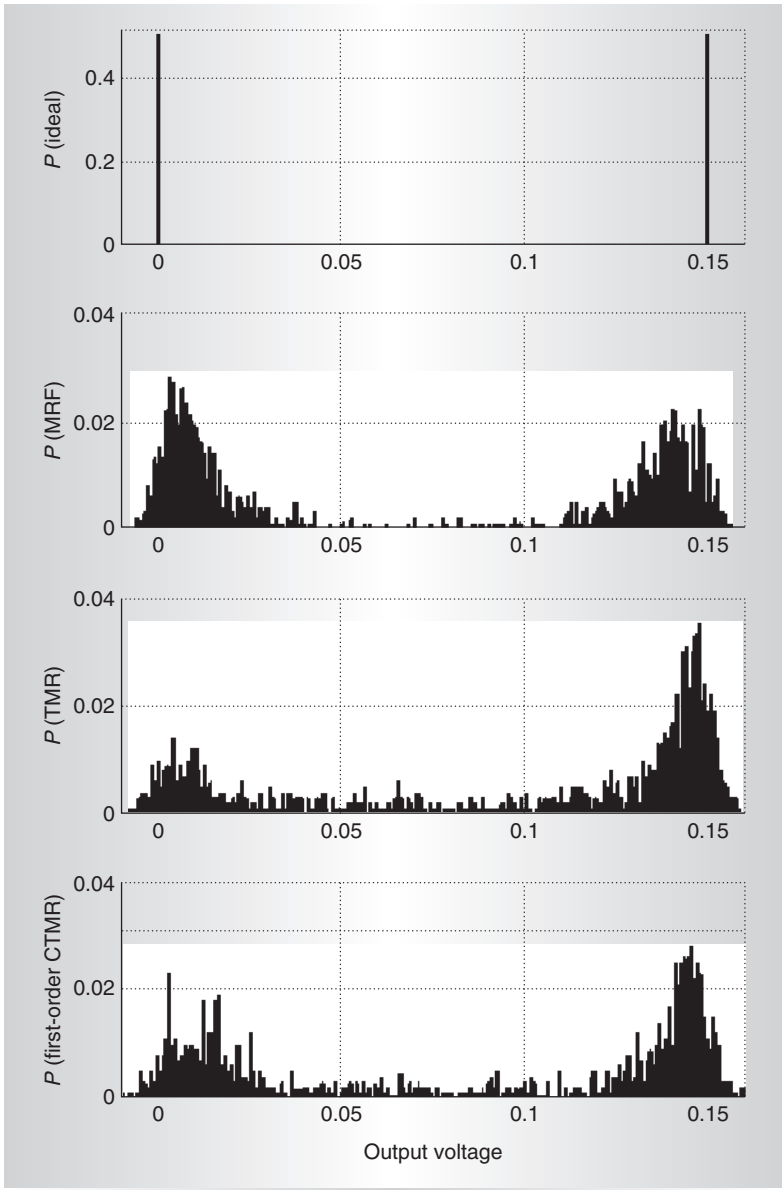


Figure 8. Probability (P) distributions of ideal output signal, TMR output, first-order CTMR output, and MRF reinforcer output.

Table 1. Comparison of error tolerance.

Approach	KLD	Error rate (%)
TMR	0.5699	8.125
First-order CTMR	0.4309	4.995
MRF reinforcer	0.2190	0.625

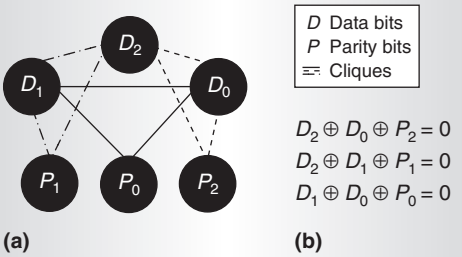


Figure 9. Dependence graph (a) and constraint equations (b) for a 3-bit data protection codeword.

running multiple simulations in the presence of uncorrelated noise for a total period of 1,000 μs. We define error as the circuit's output signal being in the middle 50 mV of the voltage range (50 mV to 100 mV). Clearly, the MRF reinforcer has far better noise immunity as measured by the KLD (for perfectly correct operation, the KLD is zero).

MRF approach for multibit protection

So far, we have focused on redundancy for single-bit states. We have also considered using redundancy based on MRF principles for multibit data fields. Traditionally, designers use error-correcting code to protect multibit data. A system using ECC encodes a number of redundant bits into the data bits, forming a code word. The code word is then transmitted over a noisy channel, and a decoder at the other end checks the parity bits to detect and correct any errors added by noise in the channel. Geman and Kochanek propose an approach applying MRF theory to ECC.[12] In their approach, the parity constraints define MRF neighborhoods in the site dependency graph, and the algorithm retrieves the correct code word by finding the maximum joint probability of data assignments using dynamic programming.

Here, we present a preliminary investigation that extends the MRF reinforcer to provide a hardware implementation of ECC redundancy. Our MRF ECC differs from traditional ECC as follows:

- Once the code word enters a noisy system, we no longer treat the data and the parity bits as two separate entities. Rather, both groups of bits become equally treated nodes of the MRF graph structure.
- Although encoding occurs in a traditional manner, we don't require explicit decoding. The system performs error detection and correction naturally using a dependence graph. Because no explicit decoding is required, we can do away with the extra overhead imposed by traditional XOR tree decoders.

Consider a 3-bit data protection scheme using Hamming encoding. Protection of an N-bit data field against single-bit errors using Hamming encoding requires a total of $\log_2 N + 1$ redundant bits. Hence, for our 3-bit data, we need 3 redundant bits, making the code word's total length 6 bits. Figure 9 shows a dependence graph and constraint equations representing the interaction between the data and redundant bits. We are currently working on a CMOS implementation of an ECC circuit based on the MRF reinforcer approach outlined in this article. Initial simulations show that the circuit correctly detects 1-bit errors and settles to the appropriate code state.

The MRF probabilistic model provides a framework for designing CMOS circuits that can operate effectively under conditions of ultra-low supply voltage and extreme noise conditions. We plan further investigation into better benchmarking of the MRF circuits in terms of area and speed, as well as more extensive comparison of our MRF reinforcer element with other higher-order CTMR techniques and other forms of space redundancy. We are also developing an additional mechanism to support logic state consensus based on Hamming distance. MICRO

References

1. T.M. Austin, "Diva: A Reliable Substrate for Deep-Submicron Microarchitecture Design," *Proc. 32nd Ann. Int'l Symp. Microarchitecture* (Micro 32), IEEE Press, 1999, pp. 196-207.
2. S.K. Reinhardt and S.S. Mukherjee, "Tran-

sient-Fault Detection via Simultaneous Multithreading," *Proc. 27th Ann. Int'l Symp. Computer Architecture* (ISCA 05), IEEE Press, 2000, pp. 25-36.
3. R.E. Lyons and W. Vanderkulk, "The Use of Triple-Modular Redundancy to Improve Computer Reliability," *IBM J. Research and Development*, vol. 6, no. 2, Apr. 1962, pp. 200-209.
4. K. Nikoliç, A. Sadek, and M. Forshaw, "Fault-Tolerant Techniques for Nanocomputers," *Nanotechnology*, vol. 13, no. 3, 2002, pp. 357-362.
5. N. Pippenger, "Reliable Computation by Formulas in the Presence of Noise," *IEEE Trans. Information Theory*, vol. 34, no. 2, Mar. 1988, pp. 194-197.
6. M. Favalli and C. Metra, "TMR Voting in the Presence of Crosstalk Faults at the Voter Inputs," *IEEE Trans. Reliability*, vol. 53, no. 3, Sept. 2004, pp. 342-348.
7. R.I. Bahar, J. Mundy, and J. Chen, "A Probabilistic-Based Design Methodology for Nanoscale Computation," *Proc. Int'l Conf. Computer-Aided Design* (ICCAD 03), IEEE Press, 2003, pp. 480-486.
8. K. Nepal et al., "Designing Logic Circuits for Probabilistic Computation in the Presence of Noise," *Proc. 42nd Design Automation Conf.* (DAC 05), ACM Press, 2005, pp. 485-490.
9. J. Besag, "Spatial Interaction and the Statistical Analysis of Lattice Systems," *J. Royal Statistical Soc.*, Series B, vol. 36, no. 3, 1994, pp. 192-236.
10. S. Narendra et al., "Full-Chip Subthreshold Leakage Power Prediction and Reduction Techniques for Sub-0.18 µm CMOS," *IEEE J. Solid-State Circuits*, vol. 39, no. 3, Mar. 2004, pp. 501-510.
11. S. Kullback, *Information Theory and Statistics*, Dover, 1969.
12. S. Geman and K. Kochanek, "Dynamic Programming and the Graphical Representation of Error-Correcting Codes," *IEEE Trans. Information Theory*, vol. 47, no. 2, Feb. 2001, pp. 549-568.

Kundan Nepal is pursuing a PhD in electrical and computer engineering in the Division of Engineering of Brown University. His research interests include fault-tolerant computing for nanoarchitectures, VLSI system design, and computer-aided design of digital integrated circuits. Nepal has a BS in electrical engineering from Trinity College, Hartford, and an MSEE from the University of Southern California. He is a student member of the IEEE.

R. Iris Bahar is an associate professor in the Division of Engineering of Brown University. Her research interests include computer architecture; computer-aided design for synthesis, verification, and low-power applications; and design, test, and reliability for nanoscale systems. Bahar has BS and MS degrees in computer engineering from the University of Illinois, Urbana-Champaign, and a PhD in electrical and computer engineering from the University of Colorado, Boulder. She is a member of the IEEE.

Joseph Mundy is a professor of engineering in the Engineering Division of Brown University. His research interests include video-based object recognition and nanoarchitecture. Mundy has a BS and a PhD in electrical engineering from Rensselaer Polytechnic Institute.

William R. Patterson is a senior lecturer and senior research engineer in the Electrical Sciences Group of the Division of Engineering of Brown University. His research interests include low-power analog circuit design for biomedical applications, probabilistic-logic circuits and architectures, and geological-spectroscopy instrumentation. Patterson has an ScB in physics and an ScM in electrical engineering, both from Brown University. He is a member of the IEEE.

Alexander Zaslavsky is an associate professor of engineering and physics at Brown University. His research interests include tunneling semiconductor physics and devices, alternative device materials such as carbon nanotubes and germanium-on-insulator, flexible electronics, and CMOS-compatible probabilistic computing. Zaslavsky has an AB in physics from Harvard University and a PhD in electrical engineering from Princeton University. He is a member of the American Physical Society.

Direct questions and comments about this article to Kundan Nepal, Brown University, Division of Engineering, Providence, RI 02912; knepal@lems.brown.edu.