Structural Fault Modelling in Nano Devices

Manoj S. Gaur, Raghavendra Narasimhan, Vijay Laxmi, and Ujjwal Kumar

Malaviya National Institute of Technology, Jaipur-302017, India

Abstract. In this paper we present a model for structural failures in nano-devices. Fault being considered include stuck-at and bridge faults only. This model is an extension of probabilistic model based on Gibbs energy distribution and belief propagation as presented in NANOLAB [1]. Results have been carried out on a 8-bit full adder circuit. Simulation results indicate that probabilistic TMR model represents bridge and stuck-at-1 faults better while deterministic model is more suited for stuck-at-0 faults.

Keywords: Structural fault, stuck-at-0, stuck-at-1, bridge, MRF, TMR.

1 Introduction

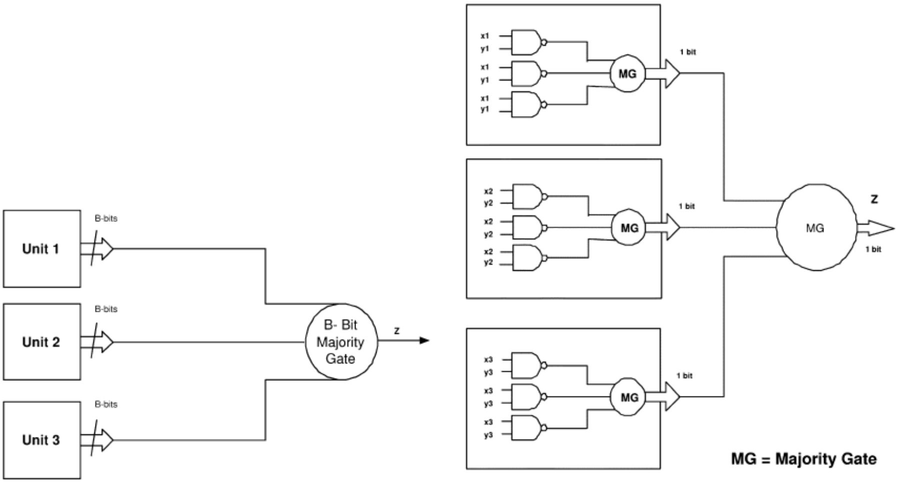
Nano-structures are inherently unreliable and this uncertainty stems from low operating energy levels, thermal perturbations/noise, significant quantum effects at nanoscale and high probability of manufacturing defects. Reliable computation requires fault-tolerant architecture and alternate computational model. Triple Modular Redundancy (TMR) and its deviants have been proposed for reliable computation and fault-tolerance [2]. Recently Chen et. al [3] has proposed Markov Random Field and Gibbs' Energy distribution based probabilistic computational model. This probabilistic approach is more reliable computational mode.

Manufacturing defects at nanoscale can lead to permanent structural failures. In addition, thermal perturbations can lead to transient failures. A single NAND gate is not expected to compute reliably at all times. Many fault-tolerant architectures [4] have been reported to deal with this problem. TMR has three similar gates working in parallel and a majority gate to compute output to improve reliability and fault tolerance. Figure 1 illustrates the circuit. Cascade Triple Modular Redundancy (CTMR) has three TMR units of the same type combined with another majority gate to form a 'second-order' TMR unit with a higher reliability. RMR is a generalization of TMR where instead of 3 we have $R = 3, 5, 7, \cdots$ units working in parallel.

2 Probabilistic Model for Nanoscale Computation

Chen [3] proposed a Markov Random Field (MRF) based probabilistic approach for nano-scale computing. This approach adapts to errors as a natural consequence

M. Cheng (Ed.): NanoNet 2008, LNICST 3, pp. 6–10, 2009.



(a) TMR

(b) CTMR

Fig. 1. Fault tolerance through TMR and CTMR

of probability maximization, thereby removing the need to actually detect faults. In MRF model, state of a node depends upon those of its neighbors or clique.

MRF model defines a finite set of random variables, $\Lambda = \{\lambda_1, \lambda_2, ..., \lambda_k\}$. Each variable λ_i has a neighbourhood, $N_i = \{\Lambda - \lambda_i\}$. Energy distribution of any variable depends only on its neighbourhood in the form of a clique. States of nodes in a Boolean network represent variables. In this model, uncertainty and noise handling is through conditional probabilities of energy values with respect to its clique. Gibbs energy distribution is used to model conditional probabilities.

$$P(\lambda_i \mid \{\Lambda - \lambda_i\}) = \frac{1}{Z} e^{\frac{1}{KT} \Sigma_{c \epsilon C} U_c(\lambda)} \qquad (1)$$

Here Z is the normalizing constant and, C is the set of cliques. U_c is the clique energy function and depends only on the neighborhood of the nodes. In a logic circuit this clique energy (logic energy) is computed as the summation over the minterms of the valid states ($F_i = 1$). This clique energy definition reinforces that the energy of the invalid logic state is greater than valid state. For a two-input x_0, x_1 NAND gate with output x_2, function $F(x_0, x_1, x_2) = 1$ when $x_2 = (x_0 \wedge x_1)\prime$. The clique energy U (-1 for valid and 0 for invalid state) for NAND gate is

$$U(x_0, x_1, x_2) = -x_2 + 2x_0x_1x_2 - x_0x_1 \qquad (2)$$

The probability of the different energy configurations of x_2 is

$$p(x_2) = \frac{1}{Z} \sum_{x_0 \epsilon \{0,1\}} p(x_0) \sum_{x_1 \epsilon \{0,1\}} p(x_1) e^{\frac{-U(x_0, x_1, x_2)}{KT}} \qquad (3)$$

2.1 Modeling Noise at Inputs and Interconnects

The probabilistic nondiscrete computing scheme described above can be extended to analyze the impact caused by signal noise at the inputs and interconnects of combinational circuits. For a two input NAND gate, there are three nodes, the inputs x_0 and x_1 and the output x_2. Noise is introduced at the input x_0 as a Gaussian process with mean μ and variance σ^2. The probability distribution of x_2 being in different energy configurations ϵ to {0.0, 0.1, 0.2, ..., 0.8, 0.9, 1.0} is

$$p(x_2) = \frac{1}{Z}\int_0^1 \sum_{x_1} e^{\frac{-U}{KT}} (\frac{e^{-(x_0-\mu)^2/2\sigma^2}}{\sqrt{2\pi\sigma}}) dx_0 \cdot p(x_1) \tag{4}$$

The energy distribution at x_2 if uniform distribution is used to model signal noise is given by the

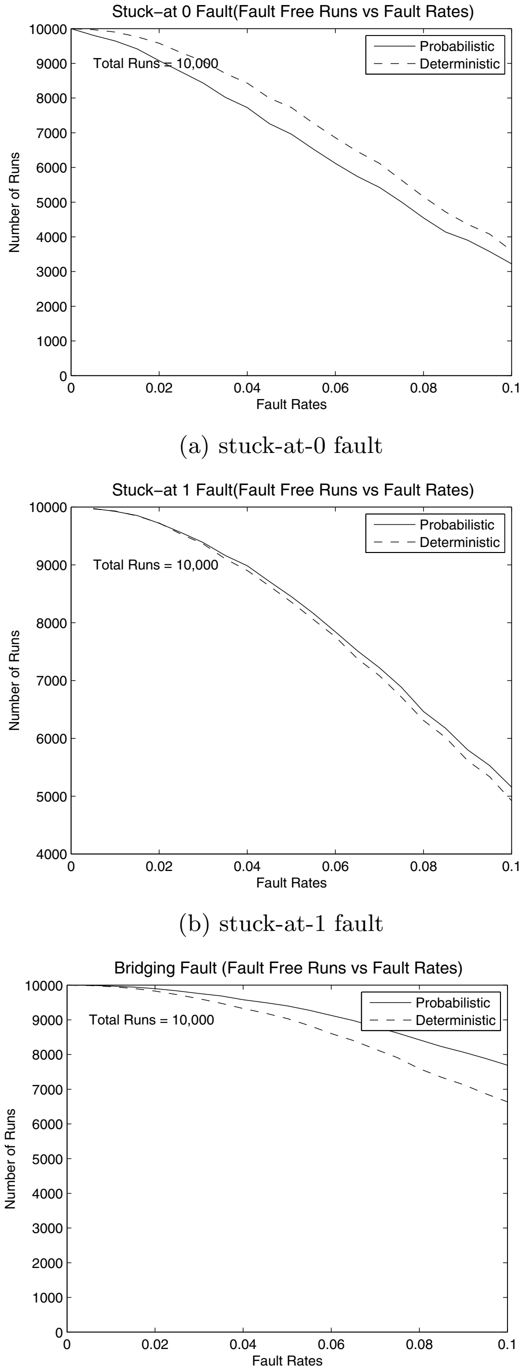
$$p(x_2) = \frac{1}{Z}\int_0^1 \sum_{x_1 \epsilon \{0,1\}} e^{\frac{-U}{KT}} dx_0 \cdot p(x_1) \tag{5}$$

2.2 NANOLAB

NANOLAB [1] is a tool developed in MATLAB for modelling of logic gates at nanolevel. This model can handle discrete energy distributions at the inputs and interconnects of any specified architectural configuration. These functions work for any generic one, two, or three-input logic gates and can be extended to handle multi-input logic gates and circuits. NANOLAB functions are parameterized and take in as inputs the logic compatibility function and the initial energy distribution for the inputs of a gate. Outcome of these functions is a probability vector indicating the probability of the output node being at different energy levels between [0..1]. These probabilities are calculated over different values of KT so as to analyze thermal effects on the node. The belief propagation algorithm is used to propagate these probability values to the next node of the network. The tool can also calculate entropy values at different nodes of the logic network. It also verifies that, for each logical component of a Boolean network, the valid states have an energy level less than the invalid states. NANOLAB functions can model noise either as uniform or Gaussian distributions or combinations of these, depending on the user specifications. Arbitrary Boolean networks in any redundancy-based defect-tolerant architectural configuration can be analyzed by writing simple MATLAB scripts that use the NANOLAB library functions.

2.3 Fault Models

Fault models are simplifications of phenomena caused by defects on the circuit. The oldest and most common model is the stuck-at fault model. Defects are modelled as a node shorted to either power supply (stuck-at-1) or to ground (stuck-at-0). Bridge fault is an extension of stuck-at fault model wherein two or more links get connected.



(a) stuck-at-0 fault

(b) stuck-at-1 fault

(c) Bridge fault

Fig. 2. Simulation results for structural faults

3 Results

The main circuit that we implemented was a 8-bit full adder circuit. Fault occurrence was assumed uniform. No gate (link) has more chance of having a fault than other gates (links). At a given location only two links get bridged. Three sets of simulations were run for stuck-at-0, stuck-at-1 and bridge faults. Both traditional deterministic and probabilistic gates were considered in all. Each simulation was repeated $n = 10000$ times with different error rates. For stuck-at fault, error rates ranged from 0.005 to 0.1, with an interval of 0.005. For bridge fault, error rates were varied from 0.5% to 10% with a difference of 0.5% at each step. Output of each simulation was considered 1 (same as expected) or 0 (otherwise). In probabilistic case, output equalled logic level with higher probability. Output was labeled invalid, if probability of any logic level was between 0.4 and 0.6. After n simulations number of correct outputs were noted for each case. Figures 2(a), (b) and (c) present results for stuck-at-0, stuck-at-1 and bridge faults respectively.

4 Conclusions

From these graphs, we see that probabilistic gate model gives better result than traditional deterministic one for bridging as well as stuck-at-1 faults. In deterministic approach the majority gate gives as output which comes maximum number of times. But in probabilistic approach the gate tries to maximize the probability of the output. For example, if the input to the majority gate is [0.8, 0.2],[0.8, 0.2],[0.8, 0.2], output probabilities should be [0.8,0.2]. But in case of probabilistic approach the output comes out to be [0.9 0.1]. However, for stuck-at-0 faults the deterministic approach yields marginally better results than the probabilistic one. In case of NAND gate, the output probability is biased towards 1. This is to be expected as its truth table has three ones and only one zero in output. For stuck-at-0 fault, one of the inputs is guaranteed to be 0 implying a guaranteed output of 1. In probabilistic approach, some of the states may become invalid if output of two gates in TMR block fall below 0.6.

References

1. Bhaduri, D., Shukla, S.: Nanolab a tool for evaluating reliability of defect-tolerant nanoarchitectures. IEEE Transactions on Nanotechnology 4(4) (July 2005)
2. von Neumann, J. (ed.): Probabilistic Logic and the Synthesis of Reliable Organisms from Unreliable Components. Princeton University Press, Princeton (1956)
3. Chen, J., Mundy, J., Bai, Y., Chan, S.M.C., Petrica, P., Bahar, R.I.: A probabilistic approach to nano-computing. Technical report, Division of Engineering, Brown University, RI 02912, USA
4. Nikolic, K., Sadek, A., Forshaw, M.: Architecture for reliable computing with unreliable nanodevices. IEEE-NANO M2.1 Nano-Devices (II), 254–259 (2001)