Stochastic Computation

Naresh R. Shanbhag, Rami A. Abdallah, Rakesh Kumar, and Douglas L. Jones
Coordinated Science Laboratory/ECE Department
University of Illinois at Urbana-Champaign, Urbana, IL-61801
[shanbhag,rabdall3,rakeshk,dl-jones]@illinois.edu

ABSTRACT

Stochastic computation, as presented in this paper, exploits the statistical nature of application-level performance metrics, and matches it to the statistical attributes of the underlying device and circuit fabrics. Nanoscale circuit fabrics are viewed as noisy communication channels/networks. Communications-inspired design techniques based on estimation and detection theory are proposed. Stochastic computation advocates an explicit characterization and exploitation of error statistics at the architectural and system levels. This paper traces the roots of stochastic computing from the Von Neumann era into its current form. Design and CAD challenges are described.

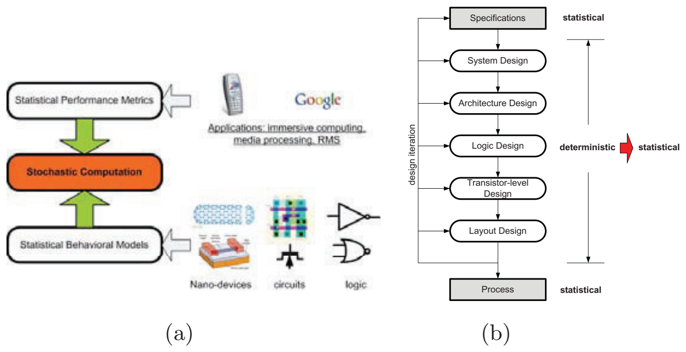

Figure 1: Stochastic computation: (a) framework, and (b) impact on SOC design methodology.

Categories and Subject Descriptors

B.8 [**Hardware**]: Performance and Reliability; F.1.2 [**Theory of Computation**]: Modes of Computation—*Probabilistic*

General Terms

Design, Reliability

Keywords

Error-Resiliency, Stochastic Computation, Algorithmic Noise-Tolerance, Soft Processing, Reliability, Energy-Efficiency

1. INTRODUCTION

[1]

Stochastic computation is presented as an elegant approach for the design of robust and energy-efficient systems-on-a-chip (SOC) in nanoscale process technologies. Moore's Law, the driving force behind the global semiconductor industry for the last 50 years, is under threat today from artifacts of nanoscale dimensions. Process, voltage and temperature (PVT) variations, leakage, soft errors, and noise in sub-45nm process technologies [1] are conspiring to make it difficult to reap the benefits of feature size scaling due to reliability concerns. A parallel trend is the growing functional complexity and power of next generation applications. The result is a power and reliability problem in nanoscale systems-on-a-chip (SOCs). Reliability and power are interlinked problems viewed by the semiconductor industry as the key inhibiters of Moore's Law. Not surprisingly, since 2001, the International Technology Roadmap for Semiconductors (ITRS) [2] has stated the achievement of reliability and energy-efficiency as two of the important challenges facing the semiconductor industry.

Stochastic computation, as presented in this paper (see Fig. 1(a)), exploits the statistical nature of application-level performance metrics of emerging applications, and matches it to the statistical attributes of the underlying device and circuit fabrics. It is fortuitous that a large class of the next generation of applications can be categorized into recognition, mining and synthesis (RMS), where massive amounts of potentially media-intensive data needs to be processed in order to extract, exploit and operate with models. Such model/knowledge-based applications are driven by modern day societal needs in security, health and energy [3]. RMS applications, especially those in media-rich immersive computing, exhibit statistical performance metrics. Examples include signal-to-noise ratio (SNR) in video compression, bit error-rate (BER) in data communications, probability of detection in face/target recognition, and many others.

[1]The authors acknowledge the support of the Gigascale Systems Research Center (GSRC), one of six research centers funded under the Focus Center Research Program (FCRP), a Semiconductor Research Corporation (SRC) entity.

Stochastic computing relies on exploiting the somewhat relaxed definition of "correctness" afforded by such applications. Naturally, there will always be a small class of critical applications such as those in finance/banking, flight control systems, and others where a precise definition of correctness is mandatory, and where stochastic computing may not be applicable in its current form. Stochastic computing views nanoscale circuit fabrics as noisy communication channels and networks (see Fig. 1(a)), and incorporates statistical behavioral models of the circuit fabric, to develop communications-inspired design techniques based on the well-established foundations of statistical estimation and detection [4]. Specifically, stochastic computation advocates an explicit characterization and exploitation of error statistics due to nanoscale artifacts, as seen at the architectural/algorithmic/system levels. The benefits of such a design philosophy are the tremendous gains in robustness and energy-efficiency in presence of an extremely high degree of unreliability (e.g., error-rates of 20%) at the circuit fabric. For example, orders-of-magnitude enhancement in system reliability has already been demonstrated for filtering [5], motion-estimation [6], Viterbi decoding [7], and CDMA pseudo-noise (PN) acquisition kernels [8], among others, along with 30%-to-60% energy-savings.

It is interesting to note that modern day SOC design methodology (see Fig. 1(b)) does indeed employ statistical analysis and design techniques at the highest (system level), and at the lowest (device modeling and characterization) levels. However, CAD algorithms and tools, with the exception of statistical static timing analysis (SSTA), are primarily based on deterministic foundations. The design of stochastic computing systems will require migration from a deterministic to a statistical basis for much of the design flow, and therefore represents a major shift in focus for the CAD community. This paper traces the roots of stochastic computing from the Von Neumann era to its current form, describing the potential gains in performance and power in the presence of device and circuit non-idealities. Design and CAD challenges resident in the design of stochastic computing systems are also described.

The rest of this paper is organized as follows. In Section 2, we describe relevant past work in the area of systems, architecture and logic design. Section 3 describes algorithmic noise-tolerance (ANT), which lays the foundations of stochastic computations as defined in this paper, followed by a networked/distributed version of ANT referred to as the stochastic sensor NOC (SSNOC). Finally, Section 5 presents stochastic computation in its current manifestation where error statistics are explicitly captured and processed. The paper ends with a discussion on some of the CAD challenges in the design of stochastic computing systems.

2. RELEVANT WORK

Von Neumann [9] was the first to address the problem of reliable computation in presence of unreliable components. Specifically, [9] showed that reliable automata/networks, i.e., networks with a probability of output error $P_{e,sys} < 0.5$ can be designed using a cascade of three-input majority gates, if the component probability of failure $p_e \leq 0.0073$, and that reliable computation is impossible if $p_e \geq 1/6$. Von Neumann also proposed a construction in which each Boolean variable x is represented by bundle of N lines carrying binary values such that the ratio of "1"-to-"0" wires represents $Pr(x = 1) = p_x$. Employing triplication of logic, it was shown that $P_{e,sys} < p_e$ is achievable. However, the overhead of this construction is enormous, e.g., to obtain $P_{e,sys} = 0.5p_e$ when $p_e = 0.005$ requires a replication factor of $N = 2000$. A number of later works followed up on [9] especially in relation to the bounds on p_e. Pippenger [10] and Feder [11] showed that if $p_e \geq 0.5 - 0.5k$, it is impossible to construct reliable networks using k-input gates. Evans [12] tightened the bound to $p_e \geq 0.5 - 0.5\sqrt{(k)}$ and Hajek [13] showed that it is possible to construct reliable networks using 3-input noisy gates if $p_e \leq \frac{1}{6}$. These results are pessimistic in many respects: 1) they show that intrinsic network reliability (without redundancy) is worse than component reliability, i.e., $P_{e,sys} > p_e$, 2) enormous complexity overhead via redundancy is required if $P_{e,sys} < p_e$, 3) the noisy gate model, i.e., individual gates independently making errors does not match reality, where errors occur when erroneous values are latched at a clock edge.

Other related work includes a statistical analysis of neural computation [14], where the classification property of a feedforward neural network is analyzed in an information-theoretic sense. The work in [14] does not embody any notion of computational errors. In [15], Markov random networks are employed to design robust logic. This implementation though quite robust to continuous-time voltage noise, has a large overhead in terms of transistor count. In [16], *stochastic logic* is proposed whereby Von Neumann's N-wire bundle representation of Boolean variables is employed, though it is assumed that that the logic is error-free, i.e., deterministic logic operating on stochastic signals.

N-modular redundancy (NMR) [17] is a commonly employed fault-tolerance technique where computation is replicated in N processing elements (PEs), and the outputs are majority voted upon. NMR's $N\times$ complexity and power overhead restricts its applicability to cost-insensitive critical applications, e.g., in the military, medical and high-end servers. Similarly, techniques such as checkpointing [18], and coding techniques [19] have been proposed, each of which though effective in enhancing robustness incur a significant energy-cost. Recently, RAZOR [20] focuses on error compensation at the logic and microarchitectural levels by employing shadow latches to detect timing errors.

Though each of the works described above is remarkable in its intent, they miss out on the opportunities available in tying application-level requirements to computation. Indeed, Von Neumann [9] was not satisfied with his results [21], and pointed out that errors in computation need to be addressed in a manner similar to how information transfer was addressed by Shannon by employing a statistical description of information. Stochastic computation, as presented in this paper, does exactly that.

3. STOCHASTIC COMPUTATION WITH ALGORITHMIC NOISE-TOLERANCE (ANT)

Algorithmic noise-tolerance [5] in Fig. 2(a) incorporates a *main* block and an *estimator*. The main block is permitted to make errors, but not the estimator. The estimator is a low-complexity (typically 5%-to-20% of the main block complexity) computational block generating a statistical estimate of the correct main PE output, i.e.,

$$y_a = y_o + \eta \quad (1)$$

$$y_e = y_o + e \quad (2)$$

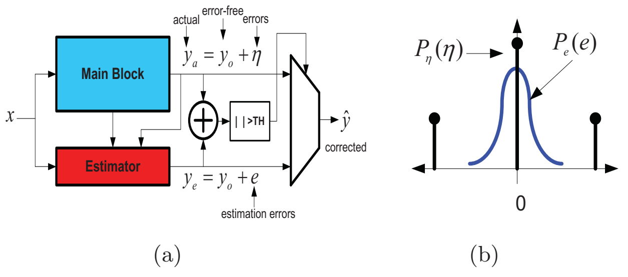

Figure 2: Algorithmic noise-tolerance (ANT): (a) framework, and (b) error distributions.

where y_a is the actual main block output, y_o is the error-free main block output, η is the hardware error, y_e is the estimator output, and e is the estimation error. Note: the estimator has estimation error e because it is simpler than the main block. ANT exploits the difference in the statistics of η and e as shown in Fig. 2(b). To enhance robustness, it is necessary that when $\eta \neq 0$, that η be large compared to e. In addition, the probability of the event $\eta \neq 0$, i.e., the component probability of error p_e of the main block, must be small. The final/corrected output of an ANT-based system $\hat{y}$ is obtained via the following decision rule:

$$\hat{y} = \begin{cases} y_a, & \text{if } |y_a - y_e| < \tau \\ y_e, & \text{otherwise} \end{cases} \qquad (3)$$

where τ is an application-dependent parameter chosen to maximize the performance of ANT. Under the conditions outlined above, it is possible to show that

$$SNR_{uc} \ll SNR_e \ll SNR_{ANT} \approx SNR_o \qquad (4)$$

where SNR_{uc}, SNR_e, SNR_{ANT} and SNR_o are the SNRs of the uncorrected main block (η dominates), the estimator (e dominates), the ANT system, and the error-free main block (ideal), respectively. Thus, ANT detects and corrects errors approximately, but does so in a manner that satisfies an application-level performance specification (SNR or BER). It employs estimation, by constructing an efficient estimator, and detection, by formulating the decision rule (3) derived from detection theory.

For ANT to also provide energy-efficiency, it is necessary that the errors in the main block be primarily due to enhancement of its energy-efficiency. In practice, these properties are easily satisfied when errors in the main block occur to voltage overscaling (VOS) [5], or a nominal case design being subjected to a worse case process corner (better than worst-case design (BTWC)). In VOS, the supply voltage is scaled below the critical voltage $V_{dd,crit}$ needed for error-free operation. As most computations are least-significant-bit (LSB) first, timing violations due to VOS or BTWC are generally large magnitude most-significant-bit (MSB) errors. Thus, timing violations satisfy the error distribution shown in Fig. 2(b).

A number of ANT techniques have been proposed in the past [5,22,23] for finite-impulse response (FIR) filters. ANT has been shown to achieve up to 3× energy savings in theory and in practice via prototype IC design [24] for finite impulse response (FIR) filters. ANT has also been employed in the design of error-resilient low-power motion estimators [6] and Viterbi decoders [7] (800× improvement in BER with 3× improvement in energy savings).

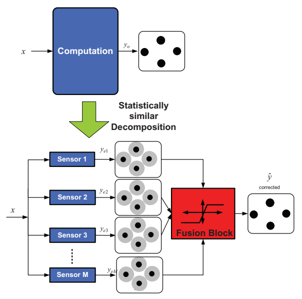

Figure 3: The stochastic sensor network-on-a-chip (SSNOC).

4. STOCHASTIC SENSOR NETWORK-ON-A-CHIP (SSNOC)

SSNOC [8] relies only on multiple estimators or *sensors* to compute, permitting hardware errors to occur (see Fig. 3), and then fusing their outputs to generate the final corrected output $\hat{y}$. Thus, the output of the i^{th} sensor is given as

$$y_{ei} = y_o + e_i + \eta_i \qquad (5)$$

where η_i and e_i are the hardware and estimation errors in the i^{th} estimator, respectively.

If hardware errors are due to timing violations, one can approximate the error term in (5) as $(1-p_e)e_i + p_e\eta_i$, where p_e is the probability of $\eta_i \neq 0$, i.e., the component probability of error. Such an ϵ-contaminated model lends itself readily to the application of robust statistics [25] for error compensation. SSNOC has been applied to a CDMA PN-code acquisition system [8], where the sensors were obtained through polyphase decomposition of the matched filter. Simulations indicate an 800× improvement in detection probability while achieving up to 40% power savings. A key drawback of SSNOC is the feasibility of decomposing computation into several sensors whose outputs are *statistically similar*, i.e., its generality. SSNOC has been applied to a CDMA PN-code acquisition system, where the sensors were obtained through polyphase decomposition.

5. STOCHASTIC COMPUTATION WITH ERROR STATISTICS

ANT and SSNOC rely on certain properties of the distribution of hardware errors η and the estimation error e. For ANT, the distributions of η and e should be sufficiently distinct, and for SSNOC, the composite error distribution should be ϵ-contaminated. ANT and SSNOC both have been shown to be powerful in enhancing robustness while providing significant energy-savings. In this section, we show that even more powerful versions of stochastic computation can be developed if error statistics are explicitly employed in computation.

We first provide an example of an error distribution. The timing error distribution at the output of a 8×8, 8-bit input, 14-bit output, 2-D DCT block using Chen's algorithm [26], with mirror adders and array multipliers [27] as fundamental

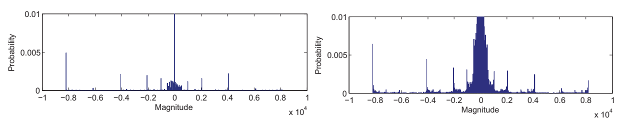

Figure 4: **Error statistics of a voltage overscaled DCT block in a 45 nm, 1.2 V CMOS process with $V_{dd,crit} = 1.2$ V: (a) $V_{dd} = 1$ V (probability of error is 0.0374), and (b) $V_{dd} = 0.8$ V (probability of error is 0.7142).**

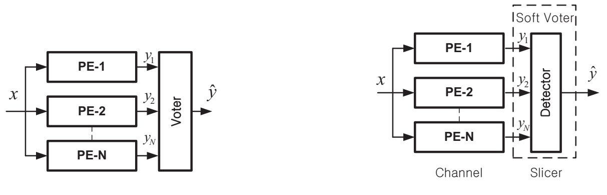

Figure 5: Block diagram of: (a) NMR, and (b) soft NMR.

building blocks, implemented in a commercial 45 nm, 1.2 V CMOS process, is shown in Fig. 4 for two different voltages. In Fig. 4, one observes that the error PMFs become more spiky as the supply voltage decreases, and that a few large amplitude errors have a high probability of occurrence. This is to be expected as the DCT architecture is LSB-first and hence timing errors will appear in the MSBs, i.e., large amplitude error will occur. The latest generation of stochastic computation employ the error PMFs such as those in Fig. 4 to enhance robustness. Two approaches are described: 1) *soft NMR*, and 2) *soft-input soft-output (SISO)* computation.

5.1 Soft NMR

Soft NMR incorporates communication-inspired techniques into NMR in order to improve its effectiveness, while preserving its generality. Structurally, soft NMR differs from NMR in that it incorporates a *soft voter*, which is composed of a *detector*. Soft NMR makes explicit use of two types of statistical information: (1) *data statistics*, and (2) *error statistics*. Data statistics are the distribution of the error-free PE output. This is referred to as the *prior* distribution, or prior. Error statistics are the distribution of the errors at the PE output. Note: the prior depends only upon input data statistics and the input-output mapping of the computation. The error distribution depends upon input data statistics, the functionality, the PE architecture, circuit style, and other implementation parameters. The role of the soft voter in Fig. 5(b) is to determine the output $\hat{y}$ that would, on average, optimize a pre-specified performance metric. The well-established detection theory [4] can be employed in order to systematically derive the soft voting algorithm. The detector maps the PE observations $y_1, y_2, \ldots, y_N$ to the "closest" *hypothesis*. Thus, the detection problem requires the definition of a *hypothesis set* $\mathcal{H}$, from which the corrected output $\hat{y}$ is selected. This is done

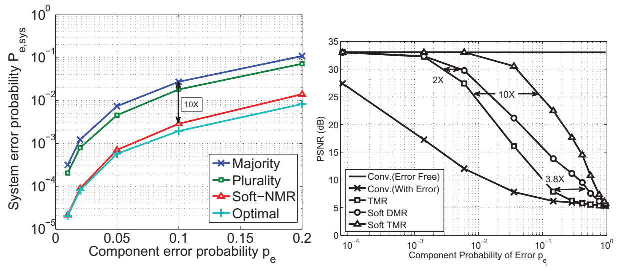

Figure 6: Simulation results: (a) $P_{e,sys}$ for an 8-bit multiplier, and (b) PSNR vs. p_e for DCT.

by solving the following:

$$\hat{y} = \arg\max_{\mathcal{H}_i \in \mathcal{H}} P(y_1, y_2, ..., y_N | \mathcal{H}_i), \tag{6}$$

where $\mathcal{H} = \{\mathcal{H}_i\}_{i=1}^{m}$ the set of all hypotheses.

As the arg max operation needs to perform a search over all possible hypotheses, for practical implementations, the hypothesis space $\mathcal{H}$ needs to be limited. There are several ways to limit $\mathcal{H}$, the simplest being to choose $\mathcal{H} = y_1, y_2, \ldots, y_N$.

The soft NMR error model is given by

$$y_i = y_o + \eta_i \tag{7}$$

where we see that there is no estimation error e in either NMR or soft NMR.

Figure 6(a) shows the results for an 8-bit multiplier where the error statistics are those of a 16-bit RCA with timing errors. Overall, reduction in $P_{e,sys}$ of 10× over NMR can be achieved by soft NMR. Figure 6(b) shows the results for an 8×8 DCT whose error statistics are shown in Fig. 4. Soft NMR provides between 2×-to-10× improvement in robustness along with 13%-to-35% savings in power (not shown) over NMR [28].

5.2 Soft-input Soft-output (SISO) Computation

One can employ error statistics to generate *soft information* λ_j for an output bit b_j,

$$\lambda_j = \ln \frac{p(b_j = 1)}{p(b_j = 0)} \tag{8}$$

where $-\infty \leq \lambda_j \leq \infty$, also called the *log-likelihood ratio* (LLR), represents the reliability (or confidence) one has in the value of whether $b_j = 1$ or $b_j = 0$. The use of soft information in the form of LLR is well-established in the design of modern day communication systems. Such systems incorporate forward error-control (FEC) decoders such as the soft Viterbi decoder, the turbo decoder and the low-density parity check (LDPC) decoder, which compute soft information in order to correct for channel errors. Thus, soft information can be exploited to enhance the robustness of computational kernels in nanoscale process technologies.

Consider a processing element (PE) B whose 2-bit output $y = (b_1, b_2)$ is

$$y = y_o + \eta \tag{9}$$

where y_o is the correct output and η is the error. For example, if $y = (1, 1)$ and $y_o = (0, 1)$ then $\eta = (1, 0)$, i.e., 2. In

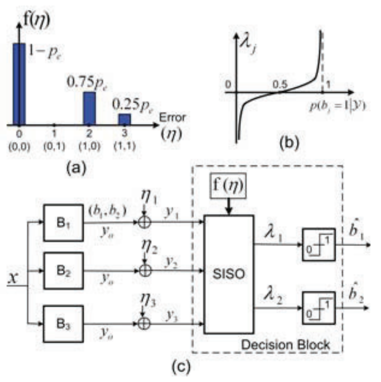

Figure 7: A motivational example: (a) a 2-bit sample error probability mass function $f(\eta)$, (b) mapping from probability to reliability measure, and (c) the $N = 3$ SISO processor.

general, y, y_o, and $\eta \in \mathcal{V}$, where $\mathcal{V} = \{(0,0),(0,1),(1,0),(1,1)\}$ is the output space.

Assume that block B's error statistics are available in the form of a probability mass function (PMF) $f(\eta)$, e.g., as shown in Fig. 7(a). Next, consider an $N = 3$ system employing three identical PEs, B_i $(i = 1, 2, 3)$, as shown in Fig. 7(c). Without any loss of generality, we assume that the PE errors e_i are indeed spatially independent, and have the PMF $f(\eta)$ in Fig. 7(a), i.e., $f_1(\eta_1) = f_2(\eta_2) = f_3(\eta_3) = f(\eta)$. Now, if $y_1 = (1,0)$, $y_2 = (1,0)$, and $y_3 = (1,1)$, then TMR selects $\hat{y} = (1,0)$ irrespective of the error statistics. On the other hand, a smart voter with the knowledge of error statistics would realize that the correct output $y_o \neq (1,0)$ since in that case $\eta_3 = y_3 - y_o = (0,1)$ but $f_3(\eta_3 = (1,0)) = 0$ (see Fig. 7(a) at $e = 1$).

Suppose we wish to exploit error statistics by computing the bit-level LLR as defined in (8). The resulting processor, in the context of NMR, is referred to as SISO-NMR (see Fig. 7(c)). In SISO-NMR, the LLR of bit b_j is given by the a-posteriori probability ratio (APR):

$$\lambda_j = \ln \frac{p(b_j = 1|\mathcal{Y})}{p(b_j = 0|\mathcal{Y})} = \ln \frac{p(b_j = 1|\mathcal{Y})}{1 - p(b_j = 1|\mathcal{Y})} \qquad (10)$$

where $\mathcal{Y}$ is the observation space and is equal to $\{y_1, y_2, y_3\}$ in Figure 7(c). Figure 7(b) plots λ_j as a function of $p(b_j = 1|\mathcal{Y})$. Note: large positive (negative) values of λ_j implies that bit b_j is highly likely to be a 1 (0), i.e., large magnitude λ_j implies greater confidence in the value of b_j. In addition, a *hard decision* on b_j is made by assigning $\hat{b}_j = sgn(\lambda_j)$, i.e., $\hat{b}_j = 1$ if $\lambda_j \geq 0$ and $\hat{b}_j = 0$ if $\lambda_j < 0$ (see Fig. 7(c)).

Given $\mathcal{Y} = \{y_1 = (1,0), y_2 = (1,0), y_3 = (1,1)\}$, the error PMF $f(\eta)$ in Fig. 7(a), and assuming $p_e = 0.2$, one can show from (10) that $\lambda_1 = 3.79$, which is greater than zero, and thus $\hat{b}_1 = 1$. Similarly, one can show that $\lambda_2 = 3.79$ and $\hat{b}_2 = 1$, and thus SISO-NMR generates the output $\hat{y} = (1,1)$. It is interesting to note that SISO-NMR can generate a correct output even when all observations y_i are incorrect. This is something NMR cannot achieve. The use of soft information for robust computation opens up the possibility of devising turbo-architectures where LLRs are exchanged between two statistically similar blocks much like a turbo decoder.

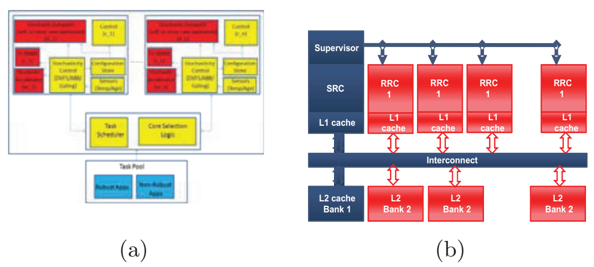

Figure 8: Stochastic processors: (a) stochastic SOC/soft processor [29], and (b) error-resilient system architecture (ERSA) [30].

5.3 Stochastic Processors

Stochastic computing ideas described in the previous sections have a clear path from system design to a dedicated ASIC implementation. One can implement stochastic accelerator cores in present day technologies. One can also conceive of a programmable platform, i.e., stochastic processors, where the principles of stochastic computing are employed. Figure 8(a) depicts Variation-Adaptive Stochastic Computer Organization (VASCO) [29] that can dynamically adapt to workload characteristics and environmental conditions to maximize energy/performance benefits from hardware stochasticity. VASCO is based on a tiled, adaptable, multi-core architecture consisting of an adaptable programmable core with reliable control and a stochastic datapath, timing speculative memory, a stochastic accelerator, and sensors to monitor environmental conditions.

Another recent approach is the Error Resilient System Architecture (ERSA) [30], a low-cost robust system architecture RMS applications. ERSA achieves high error resilience to high-order bit errors and control errors (in addition to low-order bit errors) using a judicious combination of 3 key ideas: (1) asymmetric reliability in many-core architectures, (2) error-resilient algorithms at the core of probabilistic applications, and (3) intelligent software optimizations. Soft processors and ERSA both embody the notion that reliable systems can be designed with unreliable components, but that application requirements need to be brought into the design process.

6. CAD CHALLENGES

Design of stochastic computing systems presents a number of CAD challenges in all aspects of the design methodology. Some of these are:

1. **Engineering Error-Statistics**: Develop macro-level design and synthesis techniques that result in: a) a graceful increase in error-rates, b) a desired error PMF or error rate [31], c) and uncorrelated and/or independent errors.

2. **System/Architectural Level Error Modeling**: Develop efficient CAD techniques for obtaining error PMFs from transistor-level models, and being able to compose the error PMFs of larger blocks until architectural-level models are available.

3. **Verification and Test of Stochastic Computing Systems**: Define appropriate verification and test met-

rics for reliable systems designed using unreliable components.

4. **Compilation/mapping onto Stochastic Processors**: Develop techniques to efficiently map applications on to programmable stochastic processors.

As semiconductor technology marches into the deep nanoscale regime, it is inevitable that stochasticity of the device and circuit fabric will need to be grappled with by both the design and CAD communities. This paper has described stochastic computation as an elegant SOC design philosophy for the design of reliable and energy-efficient SOCs. Hampering the design and deployment of such systems is a lack of appropriate CAD support. We hope that the CAD community will take up this challenge so that the semiconductor industry can continue to reap the benefits of Moore's law for many years to come.

7. REFERENCES

[1] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarazi, and V. De, "Parameter variations and impact on circuits and microarchitecture," in *Proc. of DAC.*, June 2003, pp. 338–342.

[2] Intl. technology roadmap for semiconductors 2008 update. ITRS. [Online]. Available: http://www.itrs.net/Links/2008ITRS/Home2008.htm

[3] J. Rabaey, D. Burke, K. Lutz, and J. Wawrzynek, "Workloads of the future," *IEEE Design and Test of Computers*, vol. 25, no. 4, pp. 358–365, July/August 2008.

[4] H. Poor, *An Introduction to Signal Detection and Estimation.* New York, NY: Springer-Verlag, 1994.

[5] R. Hegde and N. R. Shanbhag, "Soft digital signal processing," *IEEE Trans. VLSI Syst.*, vol. 9, no. 6, pp. 813–823, Dec. 2001.

[6] G. V. Varatkar and N. R. Shanbhag, "Error-resilient motion estimation architecture," *IEEE Trans. VLSI Syst.*, vol. 16, no. 10, pp. 1399–1412, Oct. 2008.

[7] R. Abdallah and N. Shanbhag, "Error-resilient low-power viterbi decoder architectures," *IEEE Trans. Signal Process.*, vol. 57, no. 12, pp. 4906–4917, Dec. 2009.

[8] G. V. Varatkar, S. Narayanan, N. R. Shanbhag, and D. Jones, "Sensor network-on-chip," in *2007 Intl. Symp. on System-on-Chip*, Nov. 2007, pp. 1–4.

[9] J. Von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," *Automata Studies*, pp. 43–98, 1956.

[10] N. Pippenger, "Reliable computation by formulas in the presence of noise," *IEEE Trans. Info. Th.*, vol. 34, no. 2, pp. 194–197, March 1988.

[11] T. Feder, "Reliable computation by networks in the presence of noise," *IEEE Transaction Information Theory*, vol. 35, no. 3, pp. 569–572, May 1989.

[12] W. Evans and L. Schulman, "Signal propagation, with application to a lower bound on the depth of noisy formulas," in *Proc. of Annual Symp. on Foundations of Computer Science*, 1993, pp. 594–603.

[13] B. Hajek and T. Weller, "On the maximum tolerable noise for reliable computation by formulas," *IEEE Transaction Information Theory*, vol. 37, no. 2, pp. 388–391, March 1991.

[14] J. Cortese and R. Goodman, "A statistical analysis of neural computation," in *IEEE Int. Symp on Information Theory*, July 1994, p. 215.

[15] K. Nepal, R. Bahar, J. Mundy, W. Patterson, and A. Zaslavsky, "Designing logic circuits for probabilistic computation in the presence of noise," in *Design Automation Conf.*, June 2005, pp. 486–490.

[16] W. Qian and M. Riedel, "The synthesis of robust polynomial arithmetic with stochastic logic," in *Proc. of Design Automation Conf.*, Jun 2008, pp. 648–653.

[17] N. Vaidya and D. Pradhan, "Fault-tolerant design strategies for high reliability and safety," *IEEE Trans. Comput.*, vol. 42, no. 10, pp. 1195–1206, Oct. 1993.

[18] Y. Tamir, M. Tremblay, and D. Rennels, "The implementation and application of micro rollback in fault-tolerant VLSI systems," in *Proc. of IEEE FTC*, 1988, pp. 234–239.

[19] S. J. Piestrak, "Design of fast self-testing checkers for a class of berger codes," *IEEE Trans. Comput.*, vol. 36, no. 5, pp. 629–634, 1987.

[20] D. Ernst et al., "Razor: A low-power pipeline based on circuit-level timing speculation," in *Proc. 36th IEEE/ACM Intl. Symp. on Microarchitecture*, Dec. 2003, pp. 7–18.

[21] S. Winograd and J. D. Cowan, *Reliable Computation in the Presence of Noise.* Cambridge, MA: MIT Press, 1963.

[22] B. Shim, S. Sridhara, and N. R. Shanbhag, "Reliable low-power digital signal processing via reduced precision redundancy," *IEEE Trans. VLSI Syst.*, vol. 12, no. 5, pp. 497–510, May 2004.

[23] L. Wang and N. R. Shanbhag, "Low-power filtering via adaptive error-cancellation," *IEEE Trans. VLSI Syst.*, vol. 51, no. 2, pp. 575–583, Feb. 2003.

[24] R. Hegde and N. R. Shanbhag, "A voltage overscaled low-power digital filter IC," *IEEE J. Solid-State Circuits*, vol. 39, no. 2, pp. 388–391, Feb. 2004.

[25] P. Huber, *Robust Statistics.* New York, NY: Wiley, 1981.

[26] W.-H. Chen, C. Smith, and S. Fralick, "A fast computational algorithm for the discrete cosine transform," *IEEE Trans. Commun.*, vol. 25, no. 9, pp. 1004–1009, Sep. 1977.

[27] J. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits*, 2nd ed. Upper Saddle River, NJ: Prentice Hall, 2002.

[28] E. P. Kim, R. A. Abdallah, and N. R. Shanbhag, "Soft NMR: Exploiting statistics for energy-efficiency," in *2009 Intl. Symp. on System-on-Chip (SOC)*, Oct. 2009, pp. 52–55.

[29] A. Kahng, S. Kang, R. Kumar, and J. Sartori, "Designing processors from the ground up to allow voltage/reliability tradeoffs," in *Proc. 16th HPCA*, Jan. 2010.

[30] L. Leem, H. Cho, J. Bau, Q. Jacobson, and S. Mitra, "Error-resilient system architecture for probabilistic applications," in *IEEE/ACM DATE*, Mar. 2010.

[31] A. Kahng, S. Kang, R. Kumar, and J. Sartori, "Recovery-driven design: A methodology for power minimization for error tolerant processor modules," in *47th DAC, Anaheim, June 2010*, Jun. 2010.