SPINTASTIC: Spin-based Stochastic Logic for Energy-efficient Computing

Rangharajan Venkatesan, Swagath Venkataramani, Xuanyao Fong, Kaushik Roy, and Anand Raghunathan
School of Electrical and Computer Engineering, Purdue University
{rvenkate,venkata0,xfong,kaushik,raghunathan}@purdue.edu

***Abstract*—Spintronics is one of the leading technologies under consideration for the post-CMOS era. While spintronic memories have demonstrated great promise due to their density, non-volatility and low leakage, efforts to realize spintronic logic have been much less fruitful. Recent studies project the performance and energy efficiency of spintronic logic to be considerably inferior to CMOS. In this work, we explore Stochastic Computing (SC) as a new direction for the realization of energy-efficient logic using spintronic devices. We establish the synergy between stochastic computing and spintronics by demonstrating that (i) the peripheral circuits required for SC to convert to/from stochastic domains, which incur significant energy overheads in CMOS, can be efficiently realized by exploiting the characteristics of spintronic devices, and (ii) the low logic complexity and fine-grained parallelism in SC circuits can be leveraged to alleviate the shortcomings of spintronic logic. We propose SPINTASTIC, a new design approach in which all the components of stochastic circuits — stochastic number generators, stochastic arithmetic units, and stochastic-to-binary converters — are realized using spintronic devices. Our experiments on a range of benchmarks from different application domains demonstrate that SPINTASTIC achieves 2.8X improvement in energy over CMOS stochastic implementations and 1.9X over a CMOS binary baseline.**

I. INTRODUCTION

With CMOS technology approaching its fundamental scaling limits, several new technologies are being explored as potential replacements. A promising direction that has emerged from this search is *spintronics*, which uses electron "spin" (or equivalently, the magnetic orientation of nanomagnets) for storing and manipulating information. Spintronic devices have proven to be well-suited for realizing dense, non-volatile memories with low leakage power, leading to several prototypes and early commercial offerings [1], [2]. For realizing spintronic logic, various approaches have been proposed including Nano-Magnetic logic (NML) [3], Domain Wall Logic (DWL) [4], mLogic [5], and All Spin Logic (ASL) [6]. However, they are not considered to be competitive with CMOS from an energy-delay perspective [7]. For instance, ASL's speed is limited by the high current required to achieve fast non-local spin-torque switching, and its energy is limited by the short circuit power resulting from all-metallic devices and the buffers needed to overcome the limited spin-diffusion length in interconnects. On the other hand, logic styles based on magnetic switching, such as NML, incur energy for generating external magnetic fields. While materials and device structures are under active investigation to improve the competitiveness of spintronic logic, we take the complementary approach of investigating alternative computing models that better match the characteristics of spintronic devices.

This work was supported in part by C-SPIN, one of the six SRC STARnet Centers, sponsored by MARCO and DARPA, and by the NSSEFF Fellows program.

In this work, we explore *stochastic computing* (SC) [8] as a new direction to efficiently realize logic with spintronic devices. Stochastic computing is a model of computation in which pseudo-random bitstreams are used to represent numbers, and computations are cast in terms of operations on such bitstreams. While it is not a general-purpose replacement for Boolean logic, SC has been applied to a number of prevalent and emerging application domains such as digital signal processing, image processing, communications, recognition, mining and synthesis [9]–[11].

A key characteristic of SC is that it enables compact, low-complexity logic implementations of arithmetic operations. For example, a multiplier in the stochastic domain can be realized with only an AND gate, since the probability of '1' at the output of an AND gate is the product of the corresponding probabilities at its inputs [8]. However, this reduction in complexity comes at a cost. SC requires additional circuits to convert data between stochastic and binary number representations and to eliminate correlations across bitstreams. These peripheral circuits impose significant overheads — they consume 80-90% of the power in our CMOS implementations of stochastic circuits — and severely limit the overall energy efficiency of SC.

In this work, we demonstrate the synergy between spintronic devices and stochastic logic. On the one hand, the physical characteristics of spin devices can be exploited to efficiently realize the peripheral circuits required for stochastic computing. On the other hand, the low complexity and bit-level parallelism of stochastic circuits can be leveraged to alleviate the shortcomings of spin-based logic. Based on the above insights, we explore SPINTASTIC (spin-based stochastic) logic and demonstrate its potential in improving the energy efficiency of spintronic logic. In contrast with conventional binary implementations where spintronic devices are not currently competitive with CMOS, our results suggest that spintronic logic can be superior to CMOS for stochastic computing.

The contributions of this work are as follows:

- We identify the synergy between stochastic computing and spintronic devices, and propose SPINTASTIC, an approach to the design of energy-efficient spintronic logic.
- We exploit the characteristics of spintronic devices to efficiently design the key building blocks required for stochastic processing *viz.* stochastic number generators, stochastic-to-binary converters, and stochastic arithmetic units.
- We evaluate the benefits of SPINTASTIC using a physics-based device modeling framework. Our experiments on 8 benchmark circuits from the recognition, mining, synthesis (RMS), signal processing, and image processing application domains demonstrate significant improvements in energy over well-optimized 16nm CMOS baselines.

978-3-9815370-4-8/DATE15/©2015 EDAA

II. Background: Stochastic Computing

Stochastic computing is a probabilistic model of computation [12] in which data is represented and processed in the form of pseudo-random bitstreams [8]. While early research in SC focused on the number representation and realization of basic arithmetic operations, recent research efforts have demonstrated SC implementations for a wide range of applications domains such as image processing [9], [10], neural networks [13], signal processing [14], and recognition, mining, and synthesis (RMS) [11].

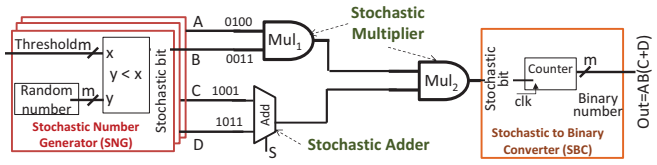


Fig. 1: Overview of a stochastic computing circuit

Figure 1 describes the structure of a typical stochastic computing circuit. First, stochastic number generators (SNGs) are used to generate stochastic bitstreams from binary inputs. Each SNG generates a series of pseudo-random numbers (*e.g.*, using a Linear Feedback Shift Register) and compares them with the given binary input, resulting in the stochastic bitstream. Next, the stochastic bitstreams are processed by stochastic arithmetic units (SAUs) that are connected together to compute the desired function. The example in Figure 1 utilizes two stochastic multipliers and a stochastic adder. A stochastic multiplier can be realized by just an AND gate, since the probability of '1' in the bitstream at the output of the AND gate is the product of the probabilities of the bitstreams at its inputs, assuming that the input bitstreams are uncorrelated. A scaled addition is realized in the stochastic domain using a MUX whose inputs are the bitstreams to be added and whose select signal is fed a random bitstream of probability 0.5; the output bitstream has a probability that is half the sum of the input probabilities. We note that stochastic circuit implementations of other complex blocks such as divider, exponent, log *etc.* have been proposed [8]. Stochastic computing has also been extended with the ability to represent and process negative numbers [8]. Finally, a stochastic-to-binary converter (SBC) circuit produces the output in binary format. This is achieved by using an up-counter that counts the '1's in the output bitstream.

From the above example, it is evident that although stochastic arithmetic units are compact, the peripheral circuits (SNGs and SBCs) dominate energy consumption, severely limiting the overall energy efficiency of SC. For example, the peripheral circuits accounted for over 89% of the total energy in our stochastic implementation of a 1D-DCT. In the following sections, we show how this drawback of stochastic computing can be addressed using spintronic devices.

III. Spintastic: Device Concepts

Spintastic uses two key concepts from spintronic devices — spin random number generator (Spin-RNG), and All Spin Logic (ASL). In this section, we present a brief description of these concepts and the context in which they are employed in Spintastic.

Spin random number generator: A spin random number generator (Spin-RNG) [15], [16] is designed by exploiting the naturally occurring random switching of a nanomagnet. A nanomagnet, due to its shape anisotropy, exhibits two stable states that are separated by an energy barrier (E_a). If the energy barrier is sufficiently lowered, it leads to thermal instability within the nanomagnet and it periodically flips its magnetization orientation due to thermal noise. The period between successive flips, referred to as the characteristic switching time (τ), is given in Equation 1.

$$\tau = t_0 e^{(E_a/k_B T)} \quad (1)$$

In Equation 1, t_0 is the attempt period (typically 1ns), k_B is the Boltzmann constant and T is the absolute temperature. Recent efforts have demonstrated switching times as low as 10s of nanoseconds and developed mechanisms to read the random state of the nanomagnets without disturbing the same [15]. In the next section, we show how to use Spin-RNGs to design stochastic number generators that can generate bitstreams of any desired probability. We note that, unlike cryptographic applications, high-quality random numbers are not required for SC — this is leveraged even in CMOS implementations, where LFSRs are typically used [9]–[11], [13], [14].

All Spin Logic: All Spin Logic (ASL) [6] is a spin-based logic style that utilizes the principle of *non-local spin torque* to achieve logic switching. The operation of ASL is illustrated

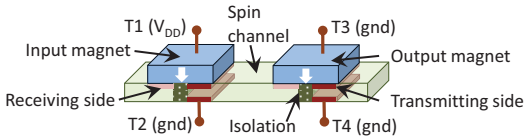
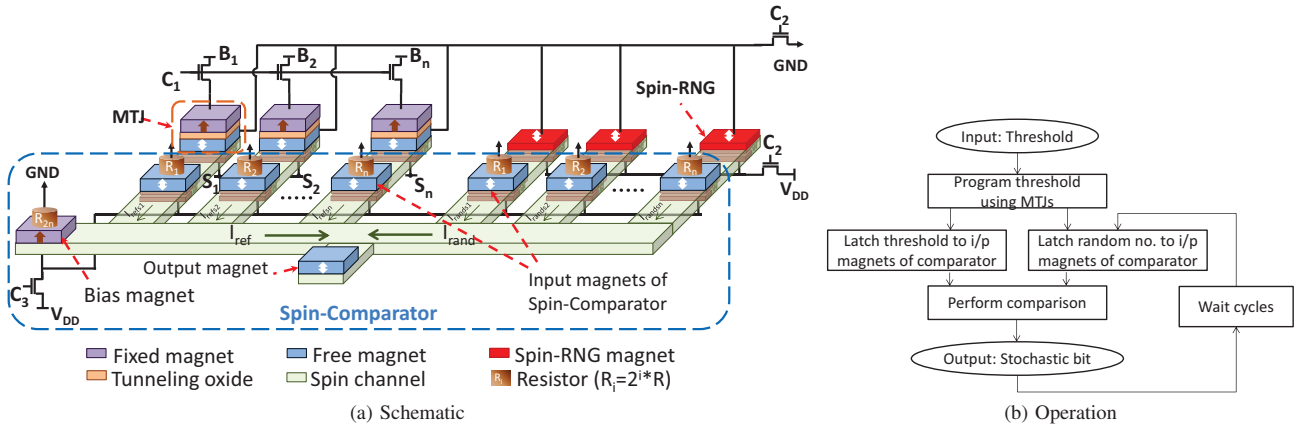


Fig. 2: ASL inverter

in Figure 2 using an ASL inverter that consists of 2 magnets – an input magnet and an output magnet – connected through a spin channel. When we apply a small voltage (V_{DD}) to Terminal-1 (T1), while the other terminals (T2,T3 and T4) are connected to GND, a charge current flows from T1 to T2 through the input magnet, which results in accumulation of electrons with spin orientation anti-parallel to that of the input magnet in the channel. These electrons flow through the channel and exert a spin torque on the output magnet. When sufficient spin torque is generated, the output magnet switches its orientation opposite to that of the input, thus realizing the functionality of an inverter. By employing the above principle and suitably connecting multiple nanomagnets, logic gates of different functionality can be realized using ASL.

A key benefit of ASL is that magnets are non-volatile and have the ability to retain their state without power supply. A magnet can therefore act as a latch, enabling fine-grained pipelined implementations with minimal overheads [17]. However, most logic circuits typically have feedback paths and input-output dependencies, which limit the scope for pipelining. Fortunately, stochastic circuits offer parallelism across the processing of bits in a bitstream, and are therefore amenable to fine-grained pipelining.

The energy consumption of ASL stems primarily from the short circuit current that flows through the metallic device structures. Since current must supplied until the circuit has completely evaluated, the energy consumption can be quite substantial when multiple ASL gates are cascaded together. Even after various optimizations such as clocking and stacking magnets, *etc.* [17], ASL incurs significantly higher energy than CMOS for implementing complex Boolean logic circuits. Fortunately, stochastic logic circuits typically have lower logic depth and hence shorter delay. Therefore, the overhead arising from short circuit current is greatly reduced. Furthermore, as



(a) Schematic (b) Operation

Fig. 3: Spintronic Stochastic Number Generator (Spin-SNG)

mentioned above, the parallelism across bits within a bitstream enables fine-grained pipelining, which has been shown to greatly improve the efficiency of ASL [17]. In SPINTASTIC, we utilize ASL to design stochastic arithmetic units and stochastic-to-binary converters.

IV. SPINTASTIC: LOGIC DESIGN

In this section, we present the design of three key components, *viz.* spintronic stochastic number generators, spintronic stochastic-to-binary converters, and spintronic stochastic arithmetic units, which form the building blocks of SPINTASTIC.

A. Spintronic Stochastic Number Generator

In SPINTASTIC, we utilize Spin-RNGs, described in Section III, to design spintronic stochastic number generators (Spin-SNGs). Figure 3 shows the structure and operation of a Spin-SNG. A set of Spin-RNG magnets are used to independently generate the bits of a binary random number. An array of magnetic tunneling junctions (MTJs) are used to store the binary input. Any binary number can be programmed into the MTJs by driving nodes B_i and S_i in Figure 3 to appropriate voltages. Next, the random number and input are compared using a *spin comparator*. The spin comparator operates by injecting a current through each input nanomagnet that is proportional to the place value of the bit that it stores. As illustrated in Figure 3, this is achieved by placing resistors of exponentially varying magnitudes (R_i=$2^i * R$) in series with the spin comparator's input nanomagnets. As a result, spin-polarized currents are injected into the channel, and the net spin polarization of the channel is determined by the relative magnitudes of the random number and input. The net spin polarization in the channel switches an output magnet using the phenomenon of non-local spin torque, to produce the stochastic bit. Note that, a bias magnet with fixed magnetization is also connected to the spin comparator's channel to ensure that the output magnet flips to a desired value when the comparator's inputs are of equal magnitude.

A key consideration in the design of the Spin-SNG is the wait time ($\sim 20ns$) between two consecutive stochastic bit outputs. Producing stochastic bitstreams at slow speeds can degrade both the performance and energy of stochastic circuits. In order to address this issue, we utilize multiple Spin-SNG instances to generate different stochastic bitstreams of the same probability, which are then time-multiplexed to generate a single stochastic bitstream at a higher throughput. Fortunately, the intrinsic compactness of Spin-SNGs implies that the overheads due to the use of multiple instances are small.

B. Spintronic Stochastic-to-Binary Converter

Spintronic Stochastic-to-Binary converters (Spin-SBCs) are responsible for translating stochastic bitstreams into their binary equivalents. This involves counting the number of 1s in the stochastic bitstream, thereby inferring its magnitude. In SPINTASTIC, we design the Spin-SBC using the optimized ASL-based full adder cells proposed in [18]. Figure 4

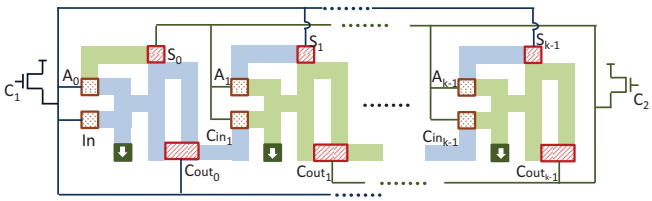


Fig. 4: Spintronic Stochastic-to-Binary Converter (Spin-SBC)

shows the design of the Spin-SBC, which consists of multiple spintronic full adder cells chained together in a ripple-carry fashion. To improve the efficiency of the Spin-SBC, we exploit the parallelism between adjacent bits in the stochastic bitstream and the non-volatility of nanomagnets to operate the different stages (spin-full adder cells) of the Spin-SBC in a two-phase pipelined fashion. We employ two non-overlapping control signals $C1$ and $C2$ to enable the odd and even stages as shown in Figure 4. Thus, a K-Stage Spin-SBC can process K/2 stochastic bits concurrently in a pipelined manner.

C. Spintronic Stochastic Arithmetic Units

Spintronic Stochastic Arithmetic Units (Spin-SAUs) perform computation on the stochastic bitstreams generated by Spin-SNGs. In SPINTASTIC, we design different Spin-SAUs using ASL. ASL logic implementations for various Boolean logic gates (*e.g.,* AND, MUX, *etc.*) are available in the literature [18], and can be directly employed in SPINTASTIC to perform computations on bitstreams. As with Spin-SBCs, we exploit the parallelism across bits in a stochastic bitstream and the non-volatility of nanomagnets to design fine-grained pipelined implementations of stochastic arithmetic units, resulting in improved throughput.

In summary, SPINTASTIC utilizes the characteristics of spintronic devices to efficiently design the different components of stochastic logic circuits.

V. Experimental Methodology and Results

In this section, we present the evaluation methodology and results comparing the energy consumption of SPINTASTIC with CMOS binary and stochastic implementations.

A. Experimental Methodology

We employed physics-based modeling frameworks proposed for ASL gates [19] and Spin-RNGs [15] to characterize the components of SPINTASTIC, *viz.* Spin-SNGs, Spin-SAUs and Spin-SBCs. We designed SPINTASTIC implementations of the following 8 benchmark circuits from the signal processing, image processing and Recognition, Mining and Synthesis (RMS) application domains: 32-tap FIR filter, 8-input 1D DCT, 16-point FFT, Euclidean distance (EUD), sum of absolute differences (SAD), perceptron classifier (PC), and artificial neural network (ANN). For all the benchmarks, we assume a bit-width of 8 bits for the CMOS binary implementation and an equivalent bitstream of length 256 bits for the stochastic designs.

We compare SPINTASTIC with two different CMOS baselines—Binary CMOS (B_CMOS) and Stochastic CMOS (St_CMOS). The benchmark circuits were implemented at the Register-Transfer Level (RTL) and synthesized to the 16nm technology library using Synopsys Design Compiler. The technology-mapped netlists were used to estimate the power, delay, and energy consumption of the CMOS baselines.

B. Energy Comparison

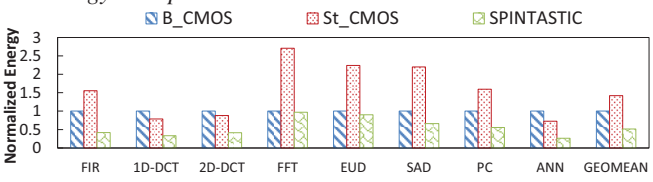


Fig. 5: Energy of SPINTASTIC *vs.* CMOS baselines

Figure 5 presents the normalized energy consumption of SPINTASTIC and the two CMOS baselines. Compared to binary CMOS, SPINTASTIC achieves between 1.02X – 3.79X (1.9X average) reduction in energy. The benefits stem from three key factors: (i) The intrinsic advantages of spintronic devices such as their low operating voltage (25 mV), (ii) The ability of spintronic devices to efficiently implement the peripheral circuits associated with SC, and (iii) The low logic complexity and logic depth of stochastic circuits and the ability to pipeline them in a fine-grained manner, which mitigate the disadvantages of spintronic logic. The energy benefits of SPINTASTIC are between 2.12X – 3.73X (2.8X on average), when compared to the stochastic CMOS baseline. Note that, the stochastic CMOS implementations are energy inefficient compared to the binary CMOS implementations due to the large energy overheads associated with the peripheral circuits (SNGs, SBCs) as well as the exponential dependency between the bitstream length and bit-width. While spintronic devices cannot directly address the latter disadvantage, they greatly reduce the energy overhead of peripheral circuits, resulting in a net lower energy consumption compared to both the CMOS baselines.

C. Energy Breakdown

To better understand the sources of energy benefits in SPINTASTIC, Figure 6 breaks down the energy consumed by the different components in stochastic CMOS and SPINTASTIC designs for the 8-input 1D-DCT benchmark. The 1D-DCT benchmark uses 72 SNGs, 8 SBCs, 64 stochastic multipliers and 8 stochastic adders. In the stochastic CMOS case, the SNGs dominate the energy consumption, contributing ∼86% of the overall energy, while the stochastic multipliers and adders only consume ∼11% of the energy. SPINTASTIC addresses the dominant component of the energy consumption, as the proposed Spin-SNGs are ∼7X more energy efficient compared to CMOS SNGs. This translates to ∼2.3X improvement in the overall energy.

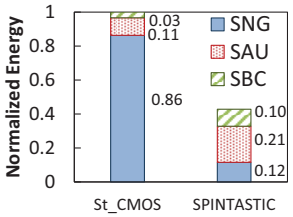


Fig. 6: Energy breakdown for 1D-DCT

VI. Conclusion

We presented SPINTASTIC a new approach to spintronic logic based on stochastic computing. We demonstrated the synergy between stochastic computing and spintronics and showed how the building blocks of stochastic logic can be realized using spintronic devices. Our experiments demonstrate that, in contrast to the energy inefficiency of spintronics compared to CMOS for general Boolean logic, SPINTASTIC circuits demonstrate improved energy consumption over both binary and stochastic CMOS baselines.

References

[1] T. Kawahara et al. 2 Mb SPRAM (SPin-Transfer Torque RAM) With Bit-by-Bit Bi-Directional Current Write and Parallelizing-Direction Current Read. *IEEE JSSC*, 43(1):109 –120, 2008.

[2] S. Fukami et al. Low-Current Perpendicular Domain Wall Motion Cell for Scalable High-Speed MRAM. In *Proc. VLSI Tech. Symp.*, pages 230 –231, 2009.

[3] G. Csaba et al. Nanocomputing by field-coupled nanomagnets. *IEEE TNANO*, 1(4):209–213, 2002.

[4] D. A. Allwood et al. Magnetic Domain-Wall Logic. *Science*, 309(5741):1688–1692, 2005.

[5] D. Morris et al. mLogic: Ultra-low voltage non-volatile logic circuits using STT-MTJ devices. In *Proc. DAC*, pages 486–491, 2012.

[6] B. Behin-Aein et al. Proposal for an all-spin logic device with built-in memory. *Nature Nanotechnology*, 5(4):266–270, 2010.

[7] D. E. Nikonov and I. A. Young. Overview of beyond-CMOS devices and a uniform methodology for their benchmarking. *Proceedings of the IEEE*, 101(12):2498–2533, 2013.

[8] B.R. Gaines. Stochastic computing systems. In *Advances in Information Systems Science*, pages 37–172. 1969.

[9] W. Qian et al. An architecture for fault-tolerant computation with stochastic logic. *IEEE Trans. Computers*, 60(1):93–105, 2011.

[10] A. Alaghi and J. Hayes. Survey of stochastic computing. *ACM TECS*, 12(2s):92:1–92:19, 2013.

[11] V. Chippa et al. StoRM: A Stochastic Recognition and Mining Processor. In *Proc. ISLPED*, pages 39–44, 2014.

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

[13] B.D. Brown and H.C. Card. Stochastic neural computation I: Computational elements. *IEEE Trans. Computers*, 50(9):891–905, 2001.

[14] S. Tehrani et al. Fully parallel stochastic LDPC decoders. *IEEE Trans. Signal Processing*, 56(11):5692–5703, 2008.

[15] X. Fong, M. C. Chen, and K. Roy. Generating true random numbers using on-chip complementary polarizer spin-transfer torque magnetic tunnel junctions. In *Proc. DRC*, pages 103–104, 2014.

[16] A Magnetic Tunnel Junction Based True Random Number Generator with Conditional Perturb and Real-Time Output Probability Tracking. In *Proc. IEDM*, 2014.

[17] M. Sharad et al. Design of ultra high density and low power computational blocks using nano-magnets. In *Proc. ISQED*, 2013.

[18] C. Augustine et al. Low-power functionality enhanced computation architecture using spin-based devices. In *Proc. NANOARCH*, pages 129–136, 2011.

[19] B. Behin-Aein et al. Switching energy-delay of all spin logic devices. *APL*, 98(12):123510–123510–3, 2011.