Architecture Considerations for Stochastic Computing Accelerators

Vincent T. Lee, Student Member, IEEE, Armin Alaghi, Member, IEEE, Rajesh Pamula, Member, IEEE, Visvesh S. Sathe, Member, IEEE, Luis Ceze, Senior Member, IEEE, Mark Oskin

Abstract—Stochastic computing (SC) is an alternative computing technique for embedded systems which offers lower area and power, and better error resilience compared to binary-encoded (BE) computation. However, the potential of and general design methodologies for SC in accelerator architectures are not well-understood. In this paper, we evaluate individual SC operations, and end-to-end accelerator architectures to understand when and why SC accelerators can achieve compelling energy efficiency gains. Based on these results, we present general design guidelines that should be considered when building energy-optimal SC accelerator architectures. We also evaluate a fully-fabricated ASIC prototype - the first of its kind - to empirically evaluate the error tolerance limits of voltage overscaling in SC. Our results show that energy efficiency gains from SC primarily stem from SC’s simpler datapaths which require fewer sequential elements compared to BE equivalents. This allows them to achieve energy efficiency gains as high as 2.4× and 30× at 8-bit and 4-bit fixed-point precision, respectively. We also find that voltage overscaling can improve the energy efficiency further by up to 1.9× by exploiting SC’s error tolerant encoding.

Index Terms—approximate computing, stochastic computing, accelerators

I. INTRODUCTION

The end of Dennard scaling has accelerated interest towards alternative computing substrates, technologies, and techniques in search of energy efficient, low power, and high density computing solutions. One computing technique that has enjoyed renewed interest in the circuits [16], VLSI [8], and CAD [37], communities is stochastic computing (SC). SC is a technique which promises high computational density, error tolerance, and low power, making it ideal for constrained environments like sensor nodes and mobile devices. Unlike many approximate computing techniques, the errors in SC can be made systematic and deterministic allowing SC to achieve hard error bounds. Note that the stochastic computing technique evaluated in this paper is different from “stochastic computing” proposed in [42], which is a collection of techniques that codesign error tolerant applications with potentially unreliable underlying devices and circuits.

SC uses unary bitstreams (time series of 1s and 0s) to encode values and perform arithmetic operations. A bitstream’s value is encoded by the number of constituent 1s and 0s, while the precision of the value is governed by the bitstream length. The value of a bitstream $X$ is defined as the probability $p_X$ of 1 occurring at a randomly chosen position. For example, the bitstream $X = 01100110$ has a value $p_X = 4/8$ since there are four 1s and the bitstream length is 8. One primary strength of SC’s encodings is that arithmetic units such as multiplication are small and low power. Given two independently generated bitstreams $X$ and $Y$ with respective values $p_X$ and $p_Y$, we can compute the product $p_X \times p_Y$ with an AND gate. Intuitively, this is because the probability of $X$ and $Y$ being 1 at the same position in the bitstreams is $p_X p_Y$. While the SC multiplier is small, it is relatively slow since run time is proportional to bitstream length; this highlights one of the key tradeoffs in SC.

While there have been many advances in circuit and implementation techniques for individual SC operations, there are no unifying guidelines for how accelerator architectures should be designed to leverage SC. Prior work has demonstrated the utility of SC in the context of neural networks [23], [25], [38], low-density parity checks [36], and image processing [7]. While these works are valuable, they do not provide more general architectural insights or guidelines as to when and why an application is compatible with SC. For instance, prior work do not characterize or even include the overheads associated with SC such as bitstream generation and conversion costs which can be fatal to the viability of SC. As a result, there is a timely need to evaluate SC accelerator architectures and identify design guidelines to facilitate effective integration.

To understand the costs and benefits of SC, we first quantify the individual energy, area, and power for SC operations by formulating a microbenchmark of commonly used SC operators. We find that most SC arithmetic operations are typically not more energy efficient than their BE counterparts. We then evaluate end-to-end accelerator architectures and tabulate architectural overheads such as pipelining, buffering, and control in BE, and conversion circuits and random number sources in SC. Our results show that despite being less energy efficient per operation, SC accelerators can still achieve up to 2.4× and 30× energy improvement at 8-bit and 4-bit precision respectively over BE ones. The key insight is that the architectural overheads associated with BE accelerators are fundamentally different and often more expensive than overheads associated with SC. We associate these gains with SC’s reduced need for sequential elements because of it’s simpler 1-bit datapaths.

We also evaluate the limits of trading error tolerance for energy efficiency by evaluating a fabricated ASIC prototype. Unlike BE encodings, SC encodings are error tolerant which enables designers to squeeze additional energy efficiency out of accelerators by trading this error tolerance. In this work, we quantify these limits by measuring the potential of voltage overscaling. Prior work [36] has postulated the compatibility of voltage overscaling for SC but only in simulation which does not fully capture the mixed-signal effects. Our results...
show that we can reduce SC accelerator energy by up to \(3 \times\) by exploiting SC’s error tolerant encoding. Based on these results, we then synthesize a set of general architectural design guidelines for designers when evaluating the SC design space.

Our contributions are as follows: (1) We evaluate the potential of SC accelerators for several applications shared across image processing, machine learning, and linear algebra. (2) We demonstrate that individual SC operations are not necessarily more energy efficient than BE ones, but as a whole can produce more energy efficient SC accelerators. (3) We present architectural design guidelines to guide the design and implementation of SC accelerator architectures. (4) We present the first fabricated ASIC prototype that empirically evaluates the limits of SC’s error tolerance and energy efficiency gains under voltage overscaling conditions.

The paper is organized as follows. Section II provides a background on SC and defines terminology. Section III presents our characterization of individual SC and BE elements. Section IV evaluates SC and BE accelerators for several well-known applications. Section V presents our ASIC prototype and evaluates SC’s error resilience, and Section VI discusses design methodology principles and guidelines for SC accelerator design. Finally, we present related work in Section VII.

II. BACKGROUND AND TERMINOLOGY

This section provides basic definitions and background for stochastic computing.

A. Basic Definitions

Stochastic computing (SC) was first introduced in the 1960s [17] and is a computation technique that uses finite-length, unary bitstreams to encode numbers. In SC, a bitstream is a serialized stream of 1s and 0s where the number of 1s and 0s in the bitstream and the bitstream length determine the encoded value. Each position in the bitstream has the same weight. This is in contrast to binary-encoded (BE) numbers in which the spatial position of the bit determines its weight (i.e., most significant bits have more weight than least significant bits.)

SC bitstreams can be either unipolar or bipolar. In unipolar bitstreams, the 1s and 0s in the bitstream are weighted at face value: 1s are +1 and 0s are 0. For instance, the unipolar interpretation of bitstream \(X = 01100000\) yields 0.375 or \(3/8\) since there are three 1s and the bitstream length is 8. Since weights are strictly non-negative, unipolar bitstreams are restricted to values in the positive range \([0, +1]\). Bipolar bitstreams, on the other hand, can encode negative numbers by weighting 1s as +1 and 0s as \(-1\); this allows them to operate on numbers in the range \([-1, +1]\). For example, the same bitstream \(X = 01100000\) would encode \(-0.25\) or \(-1/4\) since there are three 1s, five 0s, and the bitstream length is 8. The bitstream representation \(X\) is commonly referred to as a stochastic number (SN) and the variable \(p_x\) refers to the encoded unipolar value of the SN \(X\).

The precision of a SN is governed by its length \(N\). Each bit in the SN has equal weight so a SN of length \(N\) can represent a total of \(N + 1\) distinct values: \(0/N\), \(1/N\), \(2/N\), \(

\ldots\), \((N-1)/N\), \(N/N\). In other words, the effective precision of a SN is \(\log_2(N)\). As a result, SNs require exponentially longer lengths to encode higher precision values, often making them slower than equivalent BE circuits. For the remainder of the paper we will refer to SC designs by their effective BE precision (i.e., SC computation with SN length \(N\) has \(\log_2(N)\)-bit precision).

Despite this tradeoff, SC encodings provide two principle benefits: (1) the unary encoding allows for compact 1-bit datapaths, and (2) the sparse encoding provides better error tolerance. Unlike BE operations which require complex parallel datapaths, SC’s unary encoding makes the underlying arithmetic circuits smaller and lower power compared to BE circuits. SC encodings are also redundant in that multiple bitstreams encode the same value; for instance, the SNs \(X_1 = 01100110\) and \(X_2 = 10011001\) both encode the same value 0.5. While this may appear inefficient, the redundancy makes SC encodings more resilient to bit errors. For instance, a random bit flip in an \(N\) length SN will at worst increase or decrease its value by \(1/N\) whereas in BE a bit flip could change an \(m\)-bit value by up to \(2^{(m-1)}\). Furthermore, bit errors may cancel in SC since a 1-to-0 combined with a 0-to-1 error effectively result in a net change of zero.

An end-to-end example of an SC circuit for fused-multiply add is shown in Fig. 1. To operate in the SC domain, BE input values are first converted to SNs (Fig. 1a). The conversion from
a BE value to a SN is done using a digital-to-stochastic (D/S) converter. D/S converters are composed of a random number generator (RNG) and a comparator; the BE value is compared against the RNG value to generate the SN. The choice of RNG is vitally important to reduce correlation between SNs (discussed later). In this example, we assume the RNGs are uncorrelated. Once SNs are generated, they are fed to SC arithmetic units which execute the computation ([Fig. 1](#fig1)). The final phase of SC computation is converting SNs back to the BE domain for later stages of computation or storage ([Fig. 1](#fig1)). To convert SNs back to BE values, we use stochastic-to-digital (S/D) converters implemented as a counter or accumulator which sum up each position in the SN.

### B. Correlation Sensitivity and Determinism

The behavior of many SC circuits is correlation sensitive in that their accuracy depends on how correlated their input SNs are. Some SC circuits require their input SNs to be uncorrelated while others such as the SC subtractor circuit only operate properly when inputs are positively correlated. As a result, a key challenge in SC is managing and manipulating correlation between SNs. There does not exist a general set of design guidelines for managing or modeling the correlation among SNs. However, there are several ways to mitigate the effects of correlation and avoid correlation design bugs: (1) judiciously select RNGs for D/S conversions, (2) use correlation agnostic (CA) circuits, or (3) use correlation manipulation circuits.

The choice of RNG is important to managing correlation and the fidelity of SC operations. For instance, to generate uncorrelated SNs for multiplication, one should use two uncorrelated RNGs. A common misconception about SC is that numbers are always drawn from purely random RNG sequences and, as a result, achieving comparable accuracy to BE circuits requires long bitstream lengths. This is not always the case in recent work which uses deterministic number sequences. Deterministic sequences provide strong systematic error bounds on the computation, and (2) significantly better accuracies. Some SC circuits even produce exact results with no accuracy loss at all.

Prior work has also shown low-discrepancy sequences (e.g., Van der Corput, Halton, and Sobol sequences) can achieve more accurate results than LFSRs because they are minimally correlated. In this paper, we will use deterministic low-discrepancy sequences such as Van der Corput (VDC) or Halton (HLT) sequences to generate SNs. We will also use ramp sequences which have been shown to produce highly accurate multiplication results.

### C. Precision, Accuracy, and Error

Recall that SNs can only represent discrete values in the unipolar or bipolar range. As a result, values must be quantized to representable values. The granularity to which a number can be quantized is referred to as the quantization level. In SC, we define the precision of a number as its quantization level, measured by the equivalent number of bits in the encoded BE number. Typically, a SN of length \(N\) has a BE precision of \(\log_2(N)\).

On the other hand, the accuracy of a computation is governed by the output error: \(\epsilon = ||Z - Z'||\), where \(|| - ||\) denotes a distance function applied to the golden value (ground truth) \(Z\) and the computation output \(Z'\). Ideally, the golden value is obtained from the real valued or high precision floating-point computation. When such golden values are unavailable, we use the highest available fixed-point computation result as the golden value. Based on the application, we will report the accuracy of computations using mean squared error (MSE), peak signal-to-noise ratio (PSNR), and classification accuracy. For images, PSNR is defined as:

\[
 PSNR = 20\log(\text{MAX}_I) - 10\log(\text{MSE})
\]

where \(\text{MAX}_I\) is defined as the max intensity value.

In SC, there are several sources of computation error: (1) forced quantization errors due to precision reduction, (2) correlation errors introduced through the computation, and (3) errors introduced by circuit operating conditions such as voltage overscaling. The forced quantization errors occur because SC operations have the same input and output precision. For SC operations such as addition, the output does not have sufficient precision to carry the golden value causing the computation to truncate the least significant bits. On the other hand, correlation errors (Section II-B) degrade the accuracy of SC circuits but can be resolved by judiciously engineering RNGs and correlation manipulating circuits. Finally, circuit operating conditions can also lead to computation errors by introducing timing violations and erroneous results. An example of this is voltage overscaling which lowers the operating supply voltage to reduce energy and power usage (evaluated in Section V).

### III. CHARACTERIZATION

This section compares individual SC operations against BE equivalents and quantifies the costs of individual SC operations without S/D and D/S conversion overheads.

#### A. Power, Area, and Energy per Operation

We formulate a set of hardware microbenchmarks to measure energy efficiency, power, accuracy, and area at different input precisions (Table I). In this analysis, we compare results using BE and SC units operating at the same precision (iso-precision). For instance, an \(m\)-bit BE unit is compared against an SC unit using SN length \(N = 2^m\); we refer to both designs as \(m\)-bit designs in the rest of the paper. For each benchmark, we measure post-placement and route area and power via Synopsys Design Compiler, IC Compiler, and PrimeTime using a TSMC 65nm library. We use a 400 MHz frequency target and random input values to generate activity traces for power measurement. To measure PSNR, we calculated MSE over every set of inputs using the most accurate known RNG configuration. We use deterministic RNGs such as Van der Corput (VDC), Halton (HLT), linear feedback shift registers (LFSR), and ramp. Using deterministic RNGs means the output result will not fluctuate across executions.
TABLE I: Iso-precision power, area, energy efficiency, and accuracy comparison of common BE and SC arithmetic units operating at 0.8V, 25C. Not all SC units are smaller and lower power than BE equivalents. Most SC units are not more energy efficient than BE equivalents.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RNG f(px, py)</td>
<td>VDC/VDC  pX-pY</td>
<td>VDC/Ramp pXpY</td>
<td>VDC/VDC  pXpY</td>
<td>VDC/VDC max(pX, py)</td>
<td>VDC max(pX, 0)</td>
</tr>
<tr>
<td>Area (μm²)</td>
<td>Energy (J/op)</td>
<td>Power (μW)</td>
<td>PSNR (dB)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

We first compare the impact of numerical precision on energy efficiency for commonly used arithmetic circuits: addition and multiplication. For BE arithmetic units, we use the architectures synthesized by the Synopsys DC Compiler. For SC addition, we evaluate the MUX-based SC adder in Fig. 1(b) with and without a T-flip-flop (TFF) select signal, and the more recent CA adder [25]. We find that the individual energy cost per operation for SC addition is actually worse than equivalent BE circuits; at best, the margin between SC addition and BE addition is only 1.44× at 3-bit precision but quickly grows to 10× at 8 bits of precision. We also find that the more accurate CA adder is up to 10.6× less energy efficient than the standard SC adder because it contains a TFF. SC multiplication on the other hand exhibits better energy per operation gains over BE multiplication between 2 and 10 bits of precision which is roughly in line with the findings of [15].

We also evaluate several other SC circuits and compare their power, area, and energy savings relative to BE circuits in Table I. We find that arithmetic operations like multiplication and division are able to achieve better power and area overheads compared to their BE counterparts. However, we generally find individual SC circuits are not more energy efficient per operation than their BE counterparts (except for multiplication). We also show that certain SC circuits such as the counter-based SC maximum [38], and SC rectified linear unit (ReLU) [27] are neither more energy efficient, lower power, nor denser than their equivalent BE circuits. Finally, the accuracy results show that different SC operators realize significantly different accuracies. In some cases, the BE and SC PSNR is almost identical (e.g., CA adder) while in other cases SC can yield exact results with no accuracy loss (e.g., subtract).

In addition to compute units, SC circuits have additional overheads in D/S and S/D conversion circuits (Fig. 1(b) and [Fig. 1(c)]). Some prior work do not report the magnitude of these overheads or ignore the costs of these circuits all together. On average, we find S/D converters require 8.14× and 4.41× more power and area respectively than D/S converters. Relative to a SC multiplication, S/D converters are 13.8× larger and require 5× more power at 8-bit precision. While SC accelerators can amortize these overheads over many arithmetic operations by maximizing data reuse and locality, the power and area costs of these conversions cannot be completely ignored as we will show later.
We assume \( V_{th} = 0.4 \) (threshold voltage) and \( \alpha = 2 \), and do not consider subthreshold operating points. Fig. 2a shows the results of the iso-throughput and iso-precision comparison for SC and BE multipliers.

To impose the iso-area constraint, we duplicate the smaller circuit (always SC) so that the overall area matches that of the bigger circuit. This increases the throughput for SC because multiple instances of the circuit operate in parallel. The iso-area, iso-throughput, and iso-precision comparison of SC and BE multipliers is illustrated in Fig. 2a. Overall, we find that the energy efficiency of most configurations do not deviate significantly from the simple iso-precision analysis.

An iso-accuracy comparison of SC and BE multipliers is difficult and not informative. The quantization effects of the two encodings are different, leading to different accuracy levels. Our experiments show that the accuracy of a SC multiplier with SN length \( N \leq 256 \) lies between the accuracy of BE multipliers with \( m = \log_2 N \) and \( m - 1 \) bits of precision. For instance, at \( N = 256 \), SC multipliers are less accurate than a 8-bit BE multiplier, but it is more accurate than a 7-bit multiplier. As we will show later, this accuracy loss is tolerable in the context of end-to-end applications since precision tends to be more important than accuracy in many applications.

Finally, we compare “exact” SC and BE multipliers. For BE, we implemented fixed-point multipliers and assumed that their error is zero. For SC, we implemented the exact multiplier in [19] which yields the same value as the BE multiplier given sufficiently long SNs and deterministic number sources. The results in Fig. 2a show that SC multipliers are not efficient for exact multiplication and confirms prior observations by Manohar [22]. However, Manohar assumes purely random number sources for SC so this discrepancy is smaller in practice since CMOS implementations of SC can use deterministic RNGs to improve accuracy. In addition, exact computation is often not necessary for many emerging applications which are error tolerant (shown later). As a result, it is often of limited utility to provide an empirical evaluation that normalizes all parameters to achieve exact results. We therefore limit our evaluations to iso-precision or iso-area going forward.

IV. APPLICATION EVALUATION

This section evaluates SC accelerator architectures under normal operating conditions and quantifies the gains of SC accelerators.

A. Methodology

We evaluate SC accelerators for six application workloads commonly used in image processing, machine learning, and computer vision in prior works [24], [13], [45]: a 3 × 3 Gaussian blur (GB), a 2 × 2 geometric interpolation (INT), a modified Roberts cross edge detector (ED), a 5 × 5 general convolution (CONV), a modified Sobel kernel (SBL), and a matrix-vector multiplication (MV). These benchmarks contain abundant
parallelism which provides a good opportunity to evaluate the architectural design space and tradeoffs for SC accelerators.

In terms of evaluation metrics, we measure power, area, and run time. We also further breakdown power and area measurements into different accelerator components like computation and conversion overhead. In addition, we compare normalized metrics such as energy efficiency and area-normalized throughput (throughput/area) which are agnostic to design parameters. This is because iso-precision SC and BE accelerators can have different operating power, area utilization, and number of functional units. Area-normalized throughput (throughput/unit area) is a normalized performance metric that is agnostic to variations in total accelerator design area. Energy efficiency (J/op) is agnostic to run times and operating power, and is calculated as the run time multiplied by total power. Each of these metrics also takes into account SC and BE accelerator overheads such as conversion circuits, RNGs, pipelining registers, buffers, and control circuitry.

For each BE and SC accelerator design, we evaluate design points with 1 to 16 bits of input precision. For RNGs, we use deterministic sources consisting of Van der Corput (VDC), Halton (HLT), and ramp sequences. For kernels with multipliers, we use Van der Corput and ramp sequences as proposed in [23]. For kernels which only require positive or negative correlation, we use VDC for both SNs. To evaluate accelerator area, we synthesize, place, and route each SC and BE accelerator design using a TSMC 65nm technology library, Synopsys Design Compiler, and Synopsys IC Compiler. To obtain data agnostic power results, we use random data traces using the Synopsys PrimeTime power estimator. The accuracy results of each workload are calculated using a custom built cycle-level model for SC circuits; the models were verified against cycle-level RTL hardware simulations.

**B. Accelerator Architectures**

The architectures for both SC and BE accelerators are shown in [Fig. 3]. The BE image processing accelerators use a line-buffered, sliding window architecture similar to the architectures in [20]. In this architecture, image values are fed into the accelerator serially in row major order and line-buffered to maximize data reuse; this results in a compact accelerator architecture. For BE matrix-vector multiplication, we use a simple SIMD-like architecture with several multiply-accumulate units processing matrix rows in parallel. The input matrix is batched by column and fed into the accelerator one column at a time along with the kernel.

In contrast, our SC accelerator architectures have more parallel datapaths since SC’s inherent density enables more arithmetic units to be instantiated per unit area than BE circuits. To exploit this parallelism, our SC accelerator architecture tiles the input image, and computes each output within that window in parallel. For instance, for the 3 × 3 Gaussian blur, the SC accelerator would process a 10 × 10 input tile of the image and compute all 8 × 8 output pixels in parallel. This allows SC to also exploit maximal data reuse which is critical to amortize D/S data conversion costs since each pixel in the input tile is used by multiple convolution windows.

We determine the energy optimal tile width for SC across benchmarks by sweeping tile sizes 2, 4, 6, 8, 12, and 16 for SN lengths 16, 256, and 65536. The results in [Fig. 4] show that most energy optimal points across benchmarks occur at an 8 × 8 tile size followed by a 4 × 4 tile size. Thus for our evaluation, we compare SC accelerator architectures with an output window size of 8 × 8. Finally, for GB, CONV, and MV, we evaluate two accelerator design points: (1) with the smaller conventional SC adder, and (2) with the larger CA adder.
C. Evaluation Results

We now evaluate power, area, throughput, energy efficiency, and quality for SC accelerators, and quantify SC overheads. **Power:** [Fig. 5] shows the breakdown of power usage decomposed into their individual components at 4-bit (N=16) and 8-bit (N=256) precision: S/D converters, D/S converters, RNGs, kernel computation, and buffers. We generally find that the power breakdown is similar across different precisions within the same design. For BE accelerators, we find that the buffer overheads used to manage data makes up a hefty fraction of the power usage relative to kernel computation. For SC accelerators, we typically find S/D and D/S conversion overheads are non-trivial. For example, the interpolation kernel which has 4× as many S/D converters as other kernels devotes a painfully high fraction of its power budget to conversion overheads; this ultimately results in poor energy efficiency gains (discussed later). On the other hand, these conversion overheads are relatively small for compute-heavy kernels like convolution; finally, we find RNG overheads are small compared to cross domain conversion overheads and compute kernels; this is because their costs can be amortized over many D/S conversion circuits.

**Area:** In terms of density, we find SC accelerators have an effective arithmetic operator density (arithmetic units / unit area) up to 212× higher than BE accelerators (not shown). In terms of total accelerator area (Fig. 6), SC accelerators are roughly the same size as BE accelerators despite having one to two orders of magnitude more arithmetic units. This operator density allows SC accelerator architectures to instantiate many more datapaths in parallel. The additional parallel datapaths also allows SC accelerators to improve throughput to make up for longer execution times per operation due to SN length. We therefore conclude, it is not only feasible but necessary for SC accelerators to capitalize on multiple parallel datapaths to efficiently support computation.

**Throughput and Run Time:** Unlike BE accelerators where run times are agnostic to precision, SC bitstream lengths are exponentially shorter at each bit of lower precision. [Fig. 8a] shows the run time of the tiled architectures for different SC accelerator tile widths. For both SC and BE architectures, the run time is mostly agnostic of the kernel so curves are combined for simplicity. For an 8×8 tile size, we find that SC image processing accelerators are 4× slower at 8-bit precision, break even at 6-bit precision, and are faster at 5-bit precision or lower. However, a simple run time comparison does not properly account for differences in accelerator area so we also analytically calculate area-normalized throughput by scaling the SC run time by BE/SC accelerator area [Fig. 8b]. Under area-
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2018.2858338, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.

TABLE II: Image processing results and PSNR at 8-bit precision.

<table>
<thead>
<tr>
<th>Kernel</th>
<th>Fixed Point</th>
<th>Stochastic</th>
<th>Normal Operating Conditions</th>
<th>Voltage Overscaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian Blur</td>
<td>46.05 dB</td>
<td>24.04 dB</td>
<td>4.6x</td>
<td>1.2x</td>
</tr>
<tr>
<td>Edge Detection</td>
<td>55.41 dB</td>
<td>56.26 dB</td>
<td>1.0x</td>
<td>1.0x</td>
</tr>
<tr>
<td>Interpolate</td>
<td>55.42 dB</td>
<td>34.89 dB</td>
<td>1.0x</td>
<td>1.0x</td>
</tr>
<tr>
<td>Sobel SBL</td>
<td>50.92 dB</td>
<td>12.45 dB</td>
<td>1.6x</td>
<td>19x</td>
</tr>
</tbody>
</table>

Throughput

<table>
<thead>
<tr>
<th>Kernel</th>
<th>Fixed Point</th>
<th>Stochastic</th>
<th>Normal Operating Conditions</th>
<th>Voltage Overscaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian Blur</td>
<td>46.05 dB</td>
<td>24.04 dB</td>
<td>4.6x</td>
<td>1.2x</td>
</tr>
<tr>
<td>Edge Detection</td>
<td>55.41 dB</td>
<td>56.26 dB</td>
<td>1.0x</td>
<td>1.0x</td>
</tr>
<tr>
<td>Interpolate</td>
<td>55.42 dB</td>
<td>34.89 dB</td>
<td>1.0x</td>
<td>1.0x</td>
</tr>
<tr>
<td>Sobel SBL</td>
<td>50.92 dB</td>
<td>12.45 dB</td>
<td>1.6x</td>
<td>19x</td>
</tr>
</tbody>
</table>

Energy Efficiency: [Fig. 7] shows the relative energy efficiency improvement of SC accelerators over their equivalent BE accelerators. In terms of energy efficiency, we find SC accelerators under normal operating conditions are able to achieve better energy efficiency at 8 or fewer bits of precision, and are less energy efficient at higher precision. Note that SC accelerator energy efficiency decreases exponentially for each bit of increased precision since SNs become exponentially longer. We generally find kernels such as the Gaussian blur, and convolution with many arithmetic operations tend to achieve better energy efficiency gains. In these particular kernels, the power and area expenditures of each distinct S/D and D/S conversion is amortized by 6.6 to 18.2 arithmetic operations because of data locality and reuse. In contrast, kernels like interpolation only have 1.11 arithmetic operations per conversion, and achieve poor energy savings because of high conversion overheads. As a result, we conclude that workloads which have a significantly higher proportion of arithmetic operations to conversion circuits are more likely to have better energy savings in SC over BE.

The energy results also illustrate several other trends. First, SC accelerators using MUX-based adders achieve energy efficiency parity with BE accelerators at 8 bits of precision while accelerator designs using the CA adder break even at 7 bits. Second, we observe SC matrix-vector multiplication performs poorly relative to the BE SIMD accelerator; this is because the BE SIMD accelerator is compute-dominated and requires minimal state elements. Since individual BE operators are more energy efficient (as shown in Section III), the BE matrix-vector multiplication accelerator fares better against the SC one.

Quality: To evaluate quality, we use cycle-level simulation models to evaluate the image processing workloads using single precision floating point, 8-bit fixed point, and N = 256 bitstream length. We generally find PSNR is within 20 dB of fixed point baselines between the three scenarios except for the modified Sobel kernel (Table II). For applications like ED, SC can achieve identical PSNR to fixed-point baselines since the underlying SC circuits are exact. For matrix multiplication, we evaluate linear support vector machine (SVM) classifiers trained on several datasets from the UCI Machine Learning Repository [30] shown in Table III. We perform cross-validation over each dataset and report the average accuracy. Compared with fixed-point implementations, SC classifiers are within 1.1% of the BE classifier implementation. In some cases, classification accuracy for SC is still within 1% of the fixed-point implementation at 4 bits of precision.

Fig. 7: Energy efficiency improvement of SC over BE accelerators (higher is better). SC designs operating at normal voltage break even with BE designs at 8-bits. Designs with * use the CA adder. Voltage overscaled (VOS) designs are discussed in Section V.
This section evaluates the error tolerance limits of SC by evaluating the impact of voltage overscaling on a fabricated ASIC prototype. Voltage overscaling (VOS) is a technique which scales the circuit supply voltage beyond its critical point, introducing timing violations and potential bit flips in the computation but improves power and energy consumption. Intuitively, because SC uses a sparse encoding, it should be more error tolerant to these bit errors compared to BE. Recall bit errors in SC only change the encoded value by one and may cancel out (e.g., a 0-to-1 and 1-to-0 bit error occurs). On the other hand, bit errors in BE values may prove to be catastrophic as not all bits are weighted equally. Prior work has shown how this error tolerance can be exploited in the context of SC circuit VOS [36], [1] but only uses error models and simulations to measure the effects of VOS. Such models and simulations do not completely capture the complexities and actual behavior of fabricated silicon since voltage scaling is a hybrid analog-digital technique.

We evaluate VOS on a fabricated ASIC prototype for a handful of SC functional units and their BE counterparts. Our chip is fabricated using a 65nm TSMC technology node and includes SC and BE implementations of the Roberts cross edge detector kernel (ED) and a dot product unit (DP) using the CA adder. Fig. 9 shows the chip micrograph and layout of the SC and BE portions. The chip uses deterministic RNGs: for ED we use positively correlated ramp RNGs while for DP we use a VDC and a ramp RNG.

Our experiments are run using a 500 MHz operating frequency. We sweep operating voltages from 1V (normal operation), where both SC and BE work correctly, down to 0.5V at which both SC and BE implementations fail. For each operating voltage, we measure the average absolute error for SC and BE functional units using randomly generated inputs. We also test the edge detector using real image data. Fig. 11 shows the error versus voltage results for all tested functional units. Notice that the critical operating voltage at which timing violations appear is different for each functional block. For BE circuits, timing violations immediately prove fatal to the computation leaving no margin for energy reduction. In contrast, SC circuits tolerate timing violations well beyond the critical operating voltage, and can yield large energy savings.

Fig. 10 compares the edge detector results for the BE and SC kernel at different operating voltages. Unlike the BE edge detector, the SC edge detector suffers minimal accuracy losses despite the introduction of timing violations below 1V down to 0.55V. This means SC can yield an estimated 3.3× energy improvement bringing the total energy savings of SC at 8-bit and 4-bit precision to 1.38× and 17.8× respectively. This is equivalent to pushing the energy efficiency break-even precision of SC compared to BE up by one to two bits. We also observe that the SC DP can reduce voltage down to 0.6V and still exhibit reasonable error rates. Combined with the results in Section IV these results can yield up to 4.45× better energy efficiency at 8-bit precision, and up to 56× better energy efficiency at 4-bit precision for kernels like convolution.

There is a vast body of existing work demonstrating the merits of voltage scaling to save energy [22]. However, our work is the first to empirically explore the energy savings potential of trading error resilience for energy in the context of stochastic computing kernels on an actual fabricated chip. While our voltage scaling evaluation is simplistic, it provides a good empirical estimate of the energy efficiency yields available to SC accelerators. We leave a more thorough exploration against more advanced voltage scaling techniques such as those proposed by [22] to future work.

VI. DESIGN GUIDELINES

This section discusses design considerations for SC accelerators.

Higher computation-to-conversion ratio better amortizes SC overheads: Unlike BE accelerators, SC accelerators must pay for overheads such as S/D and D/S conversions, and RNGs. As a result, it is important for SC accelerator designers to also quantify SC overheads when measuring SC accelerator gains which is not always the case in previous work. While RNG costs can be amortized and shared across SC operators, our results have shown that the S/D and D/S conversions can
be painfully high relative to the computation; this is because every unique value encoded and decoded from the SC domain incurs a conversion cost. From an architectural perspective, we have found that a useful quantity to tabulate is the ratio of arithmetic operations over distinct conversions which can help better gauge whether an application is a good candidate for accelerating with SC. Actual energy efficiency gains will be governed by the exact arithmetic operation mix, but a higher compute over conversion ratio makes it more likely that SC will provide better energy efficiency gains since the conversion overheads will be better amortized. As a result, computations with a higher ratio of compute operations per conversion will achieve better power, area, and energy efficiency improvements by better amortizing SC overheads.

**Limited viable operating precision demands judicious application codesign:** A fundamental challenge with SC is that the performance and energy costs scale exponentially with precision. As a result, the viable range in which SC is more energy efficient than BE is limited to between 2 and 8 bits of precision; voltage scaling can bring the viable precision up to 9-bits. This limited viable operating precision demands judicious application codesign to minimize application accuracy loss. Such precision restriction is clearly fatal to many classes of computation which require exact results. However, there are many applications which are resilient enough to precision reduction errors such as classification, neural network and image processing tasks. In these cases, SC can achieve gains with reasonable quality degradation.

Identifying and gauging when to exploit SC therefore requires judicious selection of applications or partitioning of error tolerant parts of an application. A common characteristic among applications that perform well in SC in prior work and this work is that they are *threshold-based computations* where the overall application accuracy is not affected by small errors. Instead the accuracy depends on whether the result value surpasses a threshold. For instance, the precise output value of an SVM inference is not important as long as it surpasses the threshold that yields the correct classification label. Thus, we conclude that *codesigning SC accelerators with threshold-based applications is more likely to yield better energy/accuracy tradeoffs than applications which require exact numerical accuracy.*

**Exploiting data reuse and parallelism reduces power and area overheads:** A key strength of SC circuits are their exceptional density and simpler 1-bit datapaths with fewer levels of logic; this lends them to more parallel architectures with less pipelining. For instance, the SC accelerators evaluated in this work have one to two orders of magnitude more SC arithmetic units than the BE circuit; but, the final SC accelerators were roughly the same size as the BE accelerator. The operator density allows our SC accelerator architectures to eliminate the need to buffer data by instantiating many parallel datapaths to process each window of data. In contrast, the BE accelerators require pipelining elements and buffering to maximize data reuse and achieve a compact accelerator size. Notice that a parallel architecture is only possible because the application itself contains abundant parallelism.

Another important consideration when designing SC accelerators is exploiting application data reuse. Recall that S/D converters and D/S converters are many times larger and consume more power than individual arithmetic units. However, designs only require one S/D converter per unique value generated in the computation. Thus a single S/D converter can be shared to a value which can be reused multiple times which amortizes its power and area cost. For instance, applications such as convolution or matrix multiplication can reuse values generated for the weights across multiple dot products. As a result, we conclude that applications with significant data reuse and abundant parallelism are better suited to reap power, density, and ultimately energy efficiency gains in SC.

**VII. RELATED WORK**

**Design techniques:** While there are no established SC architecture design methodologies, there are techniques for synthesizing individual SC functional units and managing their accuracy. Prior works [6], [37] show how to use spectral transformations to synthesize SC circuits for polynomial evaluation. Both [29] and [59] also show how to synthesize sequential element SC operations. Chen et al. [13] introduce the concept of stochastic equivalence classes to reason about equivalent SC circuits. Finally, Neugebauer et al. [34] propose a general framework for managing the accuracy degradation for a SC design.

**Application-driven SC accelerators:** There is also a substantial body of work that evaluates the application-specific merits of SC. Recent work has proposed SC for neural network computation [25], [23], [9], [43], [38] and has shown SC to be inherently robust to low precision and systematic computation errors. Canals et al. [10] even proposed a new encoding - extended stochastic logic - to better support neural network computation. SC can has also been shown to achieve compelling results for image processing [7], [28], error-correcting codes [35], [18], [44], signal processing [7], [11], [40], matrix operators [45], data mining [14], machine learning [26], and discrete cosine transforms [33]. However, not all prior work considers the cost of SC overheads like D/S and S/D conversion nor do they always compare against BE ASIC implementations; both comparison are crucial when trying to evaluate the viability of SC. Our work not only quantifies these SC overheads, but also evaluates the power, area, and energy efficiency breakdown per operation, and presents guidelines to better identify promising SC accelerators applications.

**Limit studies:** To our knowledge, only a handful of studies quantify the limitations of SC circuits exist. Manohar [32] uses an analytical model to estimate that, under iso-accuracy conditions, SC will never be more energy efficient than BE. However this work assumes purely random RNGs which is not the case in recent SC work and does not holistically consider end-to-end architectural trade offs. Aguiar et al. [15] quantify the energy efficiency gains of SC multipliers over BE ones under various iso-metric conditions. These prior work do not consider architectural SC and BE design overheads like state elements, control circuitry, pipelining, and RNGs. We find that these overheads are non-trivial; but despite that, SC accelerators can still provide energy efficiency gains once both BE and SC overheads are all fully considered.
VIII. CONCLUSIONS

This paper provides a detailed architectural evaluation and comparison of SC accelerator architectures against equivalent BE accelerator architectures. We find that individually SC primitives are not decisively more energy efficient per operation than BE equivalents. However, after considering architectural overheads such as conversion circuits, buffering, and control signals, we find end-to-end SC accelerators can achieve compelling energy savings. This paper also evaluates the first fabricated ASIC prototype that empirically measures the effects of voltage overscaling on SC energy efficiency and error resilience. Based on these findings, we provide architectural design guidelines that should be carefully considered when integrating SC into accelerator architectures.

IX. ACKNOWLEDGMENTS

This work was supported in part by a Qualcomm Innovation Fellowship, gifts from Oracle, and NSF CCF grant #1518703.

REFERENCES


0278-0070 (c) 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Vincent T. Lee received his B.S. degree in Electrical Engineering and Computer Science from the University of California, Berkeley (2013). He is currently pursuing a Ph.D. degree at the University of Washington in Computer Science and Engineering and is a recipient of two Qualcomm Innovation Fellowships. His current research interests include stochastic computing, computer architecture, hardware/software codesign, and accelerator design for emerging applications.

Armin Alaghi (S’06-M’15) received the B.Sc. degree in electrical engineering (2006) and M.Sc. degree in computer architecture (2009) from the University of Tehran, Iran. He received his Ph.D. from the Electrical Engineering and Computer Science Department at the University of Michigan (2015). From 2005-2009, he was a research assistant in the Field-Programmable Gate-Array Lab and the Computer-Aided Design Lab at the University of Tehran. From 2009-2015, he was with the Computer Engineering Lab at the University of Michigan. He is currently an Affiliate Assistant Professor at the University of Washington. His research interests include digital system design, VLSI circuits, computer architecture, and electronic design automation.

Rajesh Pamula received his BTech in electrical engineering (2007) from the Indian Institute of Technology and M.Sc. from Imperial College London (2010). He worked in various capacities with the Samsung Semiconductors before starting to work towards his PhD at KU Leuven in association with imec Belgium. He is currently a visiting scientist with the Department of Electrical Engineering, University of Washington, Seattle. His major areas of research are ultra low power sensor readouts, and energy-efficient mixed-signal circuits and systems. Mr. Pamula is a recipient of the Government of Andhra Pradesh Gold Medal in 2003, ISSCC analog devices outstanding student designer award in 2016, and four gold medals from IIT in 2007 for outstanding academic performance.

Visvesh S. Sathe is an Assistant Professor at the University of Washington Department of Electrical Engineering. Till 2013, he served as a Member of Technical Staff in the Low-Power Advanced Development Group at AMD, where his research focused on digital, mixed-signal and power technologies for next-generation microprocessors. Dr. Sathe led the research and development effort at AMD that resulted in the first-ever resonant clocked commercial microprocessor. His research interests lie at the intersection of circuits and architectures for energy-efficient computing and bio-medical electronics.

Luis Ceze is a Professor of Computer Science and Engineering at the University of Washington. He is a recipient of an NSF CAREER Award, a Sloan Research Fellowship, a Microsoft Research Faculty Fellowship, and the 2013 IEEE TCCA Young Computer Architect Award. His current research focuses on the intersection of computer architecture, programming languages, and molecular biology.

Mark Oskin is an Associate Professor of Computer Science and Engineering at the University of Washington. His research interests are in computer architecture.