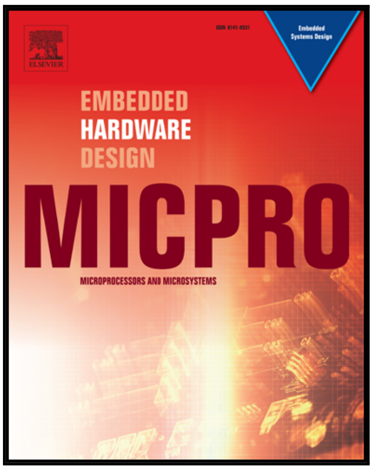

S-Box-Based Random Number Generation for Stochastic Computing

Florian Neugebauer , Ilia Polian , John P. Hayes

PII: S0141-9331(18)30059-0
DOI: 10.1016/j.micpro.2018.06.009
Reference: MICPRO 2709

To appear in: *Microprocessors and Microsystems*

Received date: 5 February 2018
Revised date: 12 May 2018
Accepted date: 14 June 2018

Please cite this article as: Florian Neugebauer , Ilia Polian , John P. Hayes , S-Box-Based Random Number Generation for Stochastic Computing, *Microprocessors and Microsystems* (2018), doi: 10.1016/j.micpro.2018.06.009

S-Box-Based Random Number Generation for Stochastic Computing

Florian Neugebauer[€], Ilia Polian[§] and John P. Hayes[$]

[€]Faculty of Computer Science and Mathematics
University of Passau,
Innstr. 43, D-94032 Passau, GER
florian.neugebauer@ uni-passau.de

[§]Institute of Computer Architecture and Computer Engineering
University of Stuttgart,
Pfaffenwaldring. 47, D-50679 Stuttgart, GER
ilia.polian@informatik.uni-stuttgart.de

[$]Computer Engineering Laboratory
University of Michigan,
Ann Arbor, MI 48109, USA
jhayes@umich.edu

Abstract—**Stochastic circuits (SCs) offer tremendous area and power-consumption benefits at the expense of computational inaccuracies. They require random number sources (RNSs) to implement stochastic number generators (SNGs) for all of their inputs. It is common for an SC to have a large number of primary and auxiliary inputs. Often the associated SNGs take up as much as 80% of the entire circuit area, so sharing RNSs is a very important goal in stochastic computing. Such sharing often leads to large correlation errors that have to be resolved via costly decorrelation methods. Linear feedback shift registers (LFSRs) are typically used as RNSs. However, we show that their deterministic and linear behavior can interfere with commonly used decorrelation methods, causing systematic computation errors, and limiting the possibilities of sharing LFSRs between SNGs. We therefore propose a novel pseudo-random number generator SBoNG for stochastic circuits that combines an LFSR with a non-linear S-box function. An SBoNG does not interfere with decorrelation and can be shared efficiently by multiple SNGs. Consequently, SBoNGs scale very well in SCs with large numbers of inputs.**

Keywords: Emerging technologies, pseudo-random number generator, simulation, stochastic computing.

I. INTRODUCTION

Stochastic computing [3][10] can provide compact, error-tolerant and low-power implementations of complex functions. It has been proposed for numerous applications such as low-density parity codes [19], image [2][17] and signal processing [7][14][30], as well as neural networks [6], and vector quantization [31]. Stochastic computing however has the downside of long computation times, calculation inaccuracies, and the need for many stochastic number generators (SNGs) to produce stochastic numbers (SNs). These take the form of *n*-bit sequences in which the SN's value is the probability of a randomly chosen bit being 1. Typically, each copy of a variable or constant input SN applied to a stochastic circuit requires its own independent SNG. Furthermore, as SNs interact during computation they become correlated, which reduces accuracy and may require additional SNGs for decorrelation purposes. As a result, the hardware needed for the SNGs can be huge, taking up more than 80% of a stochastic circuit's area [24].

An SNG consists of a random number source (RNS) driving a combinational circuit that converts a binary number *B* to an equivalent SN form *X*; see Fig. 1. In practice, a deterministic "pseudo-random" circuit that produces random-like number sequences is used as the RNS. In most prior SC work, linear feedback shift registers (LFSRs) of maximum sequence length are used as RNSs, we will refer to this type of SNG as an LFSR-SNG throughout this work. LFSRs are low-cost circuits whose behavior is relatively well understood. It is often assumed that consecutive values generated by an LFSR of suitable size are sufficiently independent of each other to allow adequate precision. This however may not be the case for LFSR-SNGs, which can lead to significant, systematic errors in stochastic circuits, as we show in this work. For example, the computation of $0.5/0.75$ by the CORDIV divider circuit using a conventional SNG consistently produces a result of 0.5 instead of the correct value 0.67, as we will discuss in detail in section II.B.

An important method in the reduction of SNG area cost in a stochastic circuit is the sharing of RNSs [15][4]. The method in [4] is however not directly applicable to stochastic computing, as it only produces one random bit per clock cycle instead of the random binary number that is needed in an SNG. Sharing can cause cross-correlation of SNs, which can be removed by either regeneration or isolation. *Regeneration* involves the conversion of the SNs to binary and back to stochastic form, thus introducing a significant hardware overhead. This method is therefore not considered useful in the literature. *Isolation* is performed by the insertion of flip-flops, so-called *isolators*, in the circuit [28]. This method is much less costly; however, it requires consecutive bits of a stochastic number to be independent of each other in order to work reliably, an assumption that does not hold for LFSRs. In most cases, errors induced due to the dependency of bits are negligible, but in some specific cases, they can cause a large, systematic error in the calculation.

In this work, we first examine the problems caused by the use of LFSRs as random number sources in SNGs by applying various correlation metrics. *Autocorrelation* is a measure of independence of bits in a single SN. *Cross-correlation* between SNs is caused by sharing SNGs. We also show that the accuracy of LFSR-based stochastic circuits can be significantly worse than what is expected for circuits with good pseudo-random number generators. For this purpose, we will