Received October 4, 2021, accepted October 31, 2021, date of publication November 8, 2021, date of current version November 15, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3125672

STAMP: A Real-Time and Low-Power Sampling Error Based Stochastic Number Generator

MAHMUT BURAK KARADENIZ, ISMAIL CEVIK, AND MUSTAFA ALTUN

Emerging Circuits and Computation Group, Department of Electronics and Communication Engineering, Istanbul Technical University, 34469 Maslak, Istanbul, Turkey

Corresponding author: Mahmut Burak Karadeniz (karadeniz17@itu.edu.tr)

This work was supported in part by the Scientific and Technological Research Council of Turkey (TUBITAK) through the Support Programme for Scientific and Technological Research Projects (1001) under Grant 116E250 and Grant 119E507, and in part by the Istanbul Technical University Scientific Research Project (BAP) under Grant 42435.

ABSTRACT In this paper, we introduce STAMP — a real-time and low-power sampling error based stochastic number generator — for stochastic computing circuits. STAMP exploits the stochastic nature of sampling error; its name is derived from 'stochastic' and 'sampling'. A unique feature of STAMP which distinguishes it from other random generators is the ability to control the output probability of the generated stochastic bit stream in real-time with no area overhead. STAMP is implemented in 180 nm CMOS. Measurements have shown that STAMP passes all tests in the suit of the National Institute of Standards and Technology (NIST) and outperforms the benchmark random number generators in terms of randomness quality. STAMP performs 140 Mb/s throughput with energy consumption of 77 pJ/bit.

INDEX TERMS CMOS, LFSR, random number generator, TRNG.

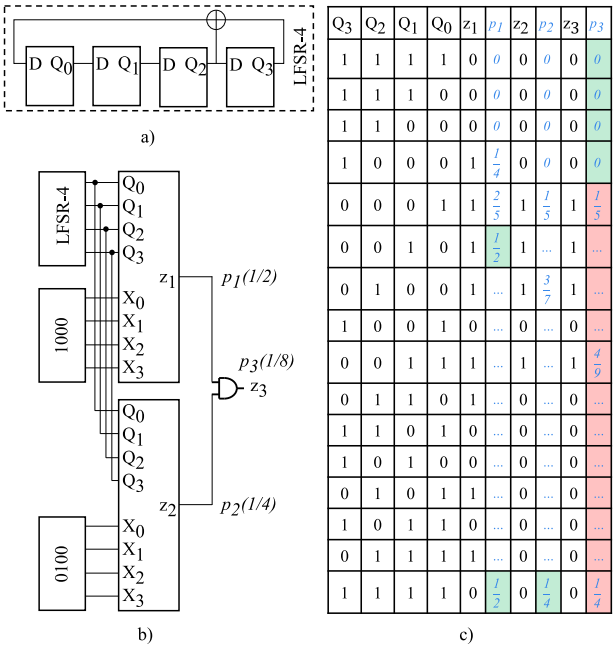
I. INTRODUCTION

Stochastic Computing (SC) has been emerging as a promising alternative for power-hungry applications in recent years. As CMOS technology approaches scaling limits, hardware usage can be minimized with SC to save area and power further for electronic devices. However, finding random sources and controlling them is a big challenge for utilizing SC. Uncorrelated stochastic numbers must be readily generated with little effort in order to maximize the advantages of SC.

There are two main classes of random number generators (RNGs) in the literature: 1) Linear Feedback Shift Register (LFSR) based RNGs, and 2) True Random Number Generators (TRNGs). LFSR-based RNGs are widely used in SC due to their area efficient simple circuitry. They can operate on high speeds, e.g., up to 15 Gb/s [1]. However, their output bits are heavily correlated; having poor randomness makes them vulnerable to failure in SC applications such as deep neural networks which have stringent accuracy specifications. Therefore, designers aim to decrease the correlation of outputs with uniquely configured LFSRs at the cost of hardware and design complexity [2]. On the other hand, the number of LFSRs needed in SC applications can go beyond practical limits which necessitates LFSR-sharing at the cost of randomness [3].

Consider a regular SC circuit implementing a multiplication with a single AND gate as shown in Fig. 1. 4-bit LFSR (LFSR-4) of Fig. 1(a) is used and shared between 2 RNGs. In RNGs, decimal 4 and 8 are used as references for the comparators to produce numbers having probabilities (p) of 1/2 and 1/4, respectively (Fig. 1(b)). The truth table of the circuit highlights that the circuit produces output with 100% relative error after full of 16 output cycles (Fig. 1(c)). There are 3 major drawbacks in this type of SC circuits. The first one is that the outputs of RNGs are not produced in real-time meaning that the each n-bit LFSR based RNG needs to wait 2^n cycles to produce the correct output with expected probability. Although the first RNG produces the correct output as early as in 6^{th} output cycle as highlighted in green in Fig. 1(c), the second RNG needs to wait 16 cycles in order to produce the output with expected probability. Difference waiting times between RNGs make the complexity of SCs worsen. The second flaw is that the output precision is $\frac{1}{2^n}$ and untouchable. In order to increase the precision, it is needed to increase the size of LFSRs and comparators. The third disadvantage of the LFSR-based SC is that the output can be very erroneous because of strong correlation between shared RNGs. For instance, the output can never be correct in this

The associate editor coordinating the review of this manuscript and approving it for publication was Chaitanya U. Kshirsagar.



Q_3	Q_2	Q_1	Q_0	z_1	p_1	z_2	p_2	z_3	p_3
1	1	1	1	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0
1	0	0	0	1	1/4	0	0	0	0
0	0	0	1	1	2/5	1	1/5	1	1/5
0	0	1	0	1	1/2	1	...	1	...
0	1	0	0	1	...	1	3/7	1	...
1	0	0	1	0	...	0	...	0	...
0	0	1	1	1	...	1	...	1	4/9
0	1	1	0	1	...	0	...	0	...
1	1	0	1	0	...	0	...	0	...
1	0	1	0	0	...	0	...	0	...
0	1	0	1	1	...	0	...	0	...
1	0	1	1	0	...	0	...	0	...
0	1	1	1	1	...	0	...	0	...
1	1	1	1	0	1/2	0	1/4	0	1/4

c)

FIGURE 1. LFSR based SC circuit: a) 4-bit LFSR schematic, b) SC multiplication, c) truth table with output probabilities (p) - correct results (green), wrong results (red).

specific example. Increasing the size of LFSRs can mitigate the randomness problem as well as the precision problem but the latency problem becomes even worse with increased delay. Also, the increased hardware cost can easily spoil the advantages of using SC.

TRNGs [4]–[6], on the other hand, are known as high quality RNGs which are preferred to be used in cryptography applications. However, harvesting true random entropy sources such as jitter in the ring oscillator (RO) and chaotic behavior of memory elements is a challenging task. The main reason is that jitter-based TRNGs suffer from weak noise which needs to be enhanced. Thus, the large number of ROs are needed [7], [8]. On the other hand, chaos-based TRNGs operate at low-speed, e.g., 10 Khz - 3 Mhz [9]–[11] because they rely on interchanging conditions on bulky memory elements which carry out complex math functions. Thus, they are not suitable to be used as random sources for SC applications.

In order to address the above-mentioned issues, we propose STAMP as a real-time stochastic signal generator which combines features such as low hardware cost, high quality of randomness and operating at high speeds. We utilize the stochastic behavior of sampling error which is generated by subtraction of the sampled signal from the analog signal. A similar approach uses the noise of analog-to-digital converters (ADCs) [12], [13]. These RNGs use ADCs and digital-to-analog converters (DACs) and outputs are subtracted from each other to generate the random source (entropy source). Then, the difference is converted into digital stochastic bits. However, unlike our approach these approaches limit the throughput since the delay increases proportional to the hardware complexity. Therefore, we come up with the idea of harnessing sampling error solely as entropy source. By doing that, we aim to minimize the hardware complexity until we derive the stochastic signal and we achieve higher throughput while keeping the quality of randomness.

ASIC and FPGAs are two of the most common platforms for deploying RNGs. FPGA-based RNGs are preferred in the literature because FPGA platforms provide rapid prototyping and user-friendly hardware language interface to be able to implement numerical methods [14]–[16]. Rapid prototyping of RNGs with FPGAs comes with a price at the cost of flexibility since the RNGs are in this way restricted to operate in the limited clock frequency. FPGA-based LFSRs are the ones that suffer most since the high throughput advantage of LFSR-based RNGs is trimmed as seen in [17], [18]. Apart from this, lengthy LFSRs are needed to be used for RNGs to be in safe place of unpredictability as discussed above, but then, the synthesis tool can easily be choked up, ending up with simulation problem by excessive memory usage as seen in [19]. Considering these issues, we prefer to built STAMP in 180 nm ASIC so that we can utilize the hardware efficiently and we can better analyze the practical limits of our proposed method.

This paper is organized as follows. Section II details the motivation behind STAMP. Section III explains the system architecture of STAMP. Section IV presents performance and evaluation of STAMP in comparison with the state-of-the-art RNGs and Section V concludes the paper.

II. MOTIVATION

STAMP has two main functions: 1) it generates the entropy source, and 2) it controls to yield a stochastic bit stream with desired probability.

A. ENTROPY SOURCE GENERATION

STAMP derives sampling error from sampling of analog signal as entropy source. Such entropy source is fed from 3 main factors, i.e., the nonideality of sampling circuit, the jitter of sampling clock and the thermal noise on sampling capacitor. The last one can be neglected in a singe-pole system where a switch with lower on-resistance and larger sampling capacitor are used. The first factor is relatively user-dependent and can be minimized with using larger design ratio (W/L) for the sampling switches. Thus, STAMP will focus on frequency of sampling clock and inescapable jitter on this clock for generating random signal. In order to elucidate this, we present 3 cases for the sampling clock and analog signal in Fig. 2. In the first case, Fig. 2(a) depicts that the input signal, *V(in)*, is sampled by a sampling clock, $\phi(clk)$, whose frequency is multiple of frequency of the input signal. The error voltage, *V(sample) - V(in)*, generated by subtracting sampled signal, *V(sample)*, from *V(in)* is compared to a reference voltage and digitized. The generated pseudo-random cycle continues on a cycle of the *V(in)*. On the other hand, Fig. 2(b) depicts a case where frequencies of *V(in)* and $\phi(clk)$ are co-prime. The pseudo-random cycle is

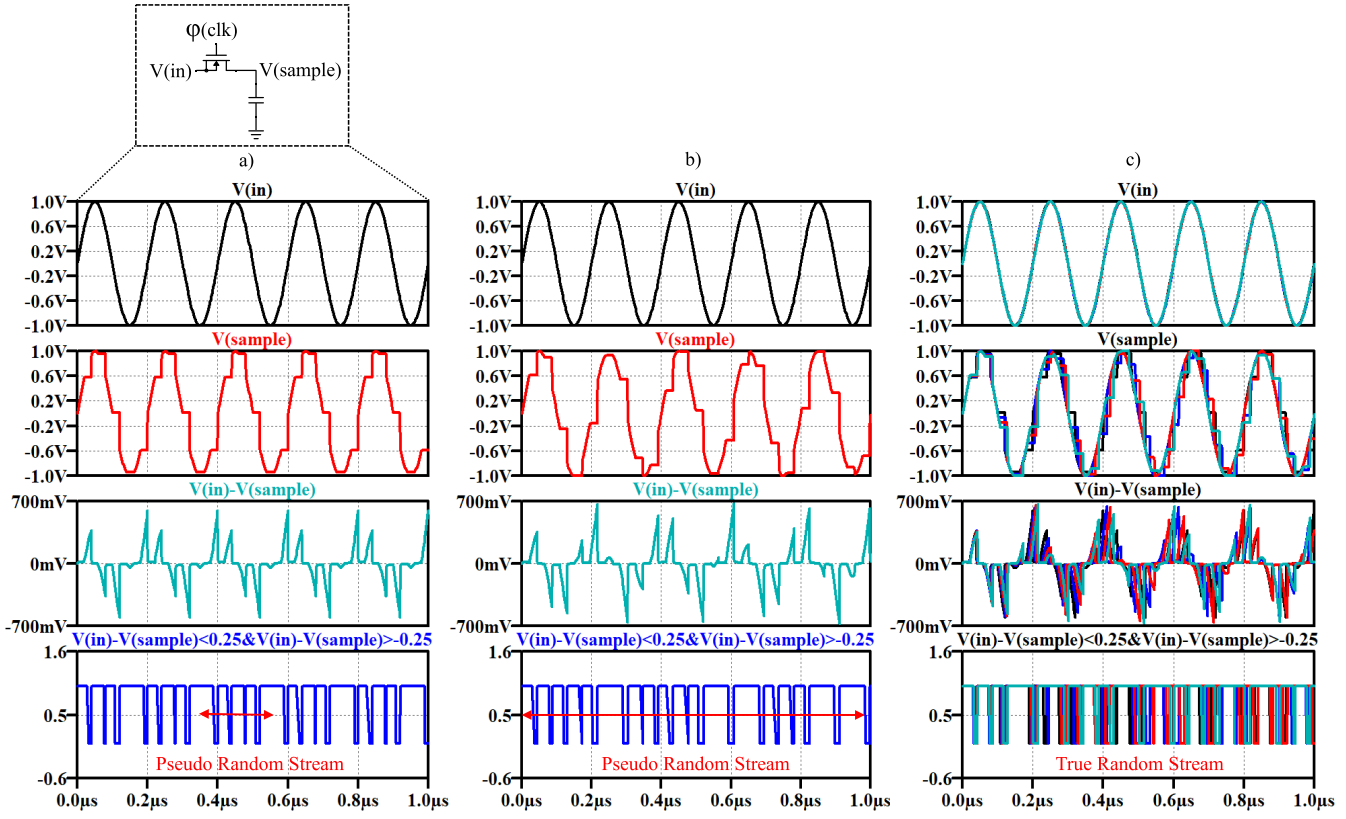


FIGURE 2. STAMP's entropy source generation: a) case (a), frequency of *V(in)* is multiple of $\phi(clk)$, b) case (b), frequency of *V(in)* is co-prime with $\phi(clk)$, c) case (c), frequency of *V(in)* is co-prime with $\phi(clk)$ where a low-jitter occurs.

extended throughout the cycles of *V(in)* until the original of these two signals intercept each other again. This is because in each cycle of $\phi(clk)$ different portion of *V(in)* is sampled and copied to error signal. This suggests that a jitter in $\phi(clk)$ provides non-overlapping random bit streams which constructs the entropy source of STAMP. This is illustrated in Fig. 2(c) where a low-jitter on $\phi(clk)$ is present over case (b).

STAMP uses sampling signal and analog signal whose frequencies are co-prime in order to enrich the randomness. This preference is to prevent pattern loss in some cases where a low-magnitude jitter shows up which is also a severe problem for some true random number generators harnessing jitter in oscillator.

STAMP uses sine-waves as analog signals. Sampling a nonlinear signal introduces irregular sampling errors which contributes the randomness. In addition to that, any signal model can be formed by using sine-waves. This allows us to control probability density function (PDF) of the sampling errors for probability conversion which is detailed in the following section.

B. PROBABILITY CONVERSION

If an analog signal is sampled by a periodic signal, the maximum and the minimum sampling error, denoted by

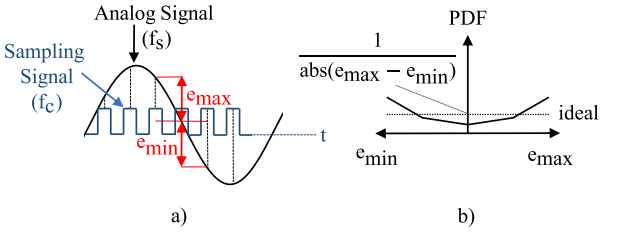


FIGURE 3. Sampling error: a) e_{max}-e_{min}, b) PDF of sampling error.

e_{max} and e_{min} in Fig. 3(a), respectively, can be approximated as

$$\frac{dv}{dt} = 2\pi f_s cos(2\pi f_s t) \tag{1}$$

where v and f_s are the magnitude and the frequency of the analog signal, respectively. Considering the maximum term, the absolute magnitude of the maximum or the minimum sampling error is defined as

$$|e_{max}, e_{min}| = \frac{\pi f_s}{f_c}, \tag{2}$$

where f_c is the sampling frequency. The PDF of the sampling errors inherits the nonuniform structure of the analog signal as seen in Fig. 3(b). This suggests that if a uniform signal is used as signal source, the sampling error distribution would

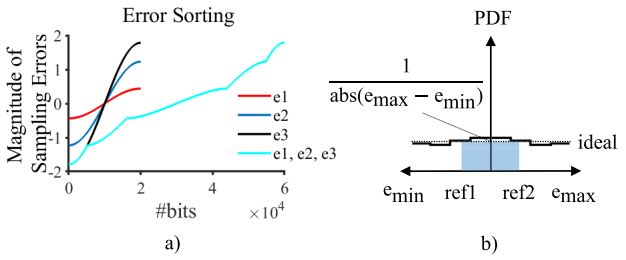


FIGURE 4. Sorting and PDF of sampling errors: a) sorting of sampling errors from each harmonic and sorting of mixed sampling errors (e1,e2,e3), b) PDF of mixed sampling errors.

be uniform. Referring to Fourier series for square-wave as

$$v_{square} = sin2\pi f_s t + \frac{1}{3} sin2\pi 3 f_s t + \frac{1}{5} sin2\pi 5 f_s t \ \ldots, \quad (3)$$

it is implied that if odd harmonics of the sine-wave are sampled, PDF of the mixed sampling errors can approximate to uniform.

Therefore, STAMP generates the first three odd harmonics of sine wave as $v_{1,2,3}(t) \rightarrow sin(f_s)t, sin(3f_s)t, sin(5f_s)t$ and samples each harmonic with a frequency, f_c, with at most Nyquist rate for the third harmonic and over-sampling rate for the first harmonic ($f_c \leq 10 f_s$). After subtracting each sampled wave from the analog signal and combining them, STAMP gets mix of three sampling errors each of which is distributed over its max and min values as

$$PDF(e_{max}, e_{min}) = \pm\frac{1}{10} e_1 \vee \pm\frac{3}{10} e_2 \vee \pm\frac{5}{10} e_3 \quad (4)$$

where $e1$, $e2$ and $e3$ are error signals acquired from subtracting each sampled harmonics from the input signal. Fig. 4(a) displays sorting of these signals and also sorting if they are mixed together, i.e., $e1$, $e2$ and $e3$. Each sampling error has PDF of Fig. 3(b), but the PDF of error mix is rectified as seen in Fig. 4(b). In other words, the sparse distribution of mean-magnitude sampling errors from a odd harmonic is strengthened by the dense distribution of max or min-magnitude sampling errors from the next odd harmonic. After that, probability conversion is done by adjusting the reference points ($ref\,1$, $ref\,2$) as

$$P_{out} = |(ref\,2 - ref\,1)/2e_{max}|. \quad (5)$$

Based on the above-mentioned considerations, STAMP utilizes sampling error as entropy source. In order to enrich the randomness, it uses input and clock signals whose frequencies are co-prime. In order to scale the input reference voltage to the probability of the random bit stream in real-time, it needs to obtain sampling errors from 3 odd harmonics of input sine-wave and mix them together.

III. SYSTEM ARCHITECTURE

STAMP has the hierarchy depicted in Fig. 5. It consists of 6 main blocks. These are system clock generator, signal generator, sampler, subtractor, comparator and counter

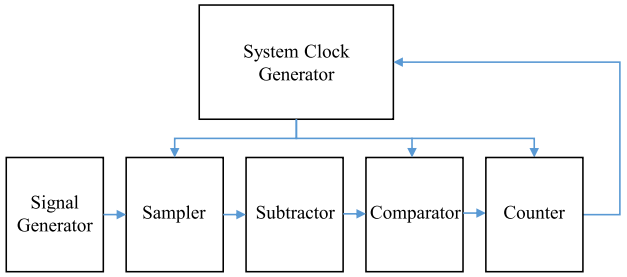


FIGURE 5. Hierarchy diagram of STAMP.

blocks. The arrows between blocks represent the relationship between blocks in the direction of arrow. For instance, sampler block gets inputs from system clock generator and signal generator blocks and generates output for the subtractor block. These inputs and outputs signals as well as the internal signals are detailed below block by block in the sub-sections. The deterministic signal is generated by signal generator block and then, processed by the sampler, the subtractor and the comparator blocks. The output stochastic data is collected by the counter block. The counter, the comparator and the sampler blocks get their reference signals from system clock generator block which generates a general system clock and divides it to be distributed to these sub-blocks. System clock generator block has a feedback from the counter block to generate a variable system clock as an alternative to the constant system clock. The details are as follows:

A. SIGNAL GENERATOR BLOCK

STAMP's architecture is given in Fig. 6. The highlighted areas refer to blocks depicted in hierarchy diagram of Fig. 5. STAMP can either use off-chip signals (Sin1, Sin2 and Sin3) as inputs or it can generate signals on-chip (Sin1′, Sin2′, Sin3′). On-chip signals are generated by a combination of ring oscillators (ROs) and RC low pass filters (RC-LPFs). Frequency of a RO is $f_{RO} = 1/(2 \times N \times t_d)$ where N is the number of inverters and t_d is the average delay of the inverters. One of the inverters in the RO is voltage-controlled for compensating the effects of the parasitics in the feed-back loop and for controlling the total delay in the loop. Cut-off frequency of each RC-LPF (fc_{LP}) is set equal to f_{RO} to obtain the first 3 odd harmonics of the sine-wave, and $fc_{LP} = 1/(2\pi \times R \times C)$. Their frequencies are neatly coupled through RC product to acquire fc_{LP}, $3fc_{LP}$, $5fc_{LP}$.

B. SAMPLER BLOCK

In the next step, each signal is directly stored on a capacitor and sampled on a second capacitor. Resistance of the transmission gate in the Sample and Hold (S/H) block is determined as $R_{on} = 1/(2\pi \times f_{3dB} \times C_{total})$.

Sampling capacitor is set to relatively large value of 1 pF to cancel out the parasitic capacitance of the adjacent blocks thus stabilizing the sample and hold time. Since STAMP samples under Nyquist rate for the third harmonic, nominal

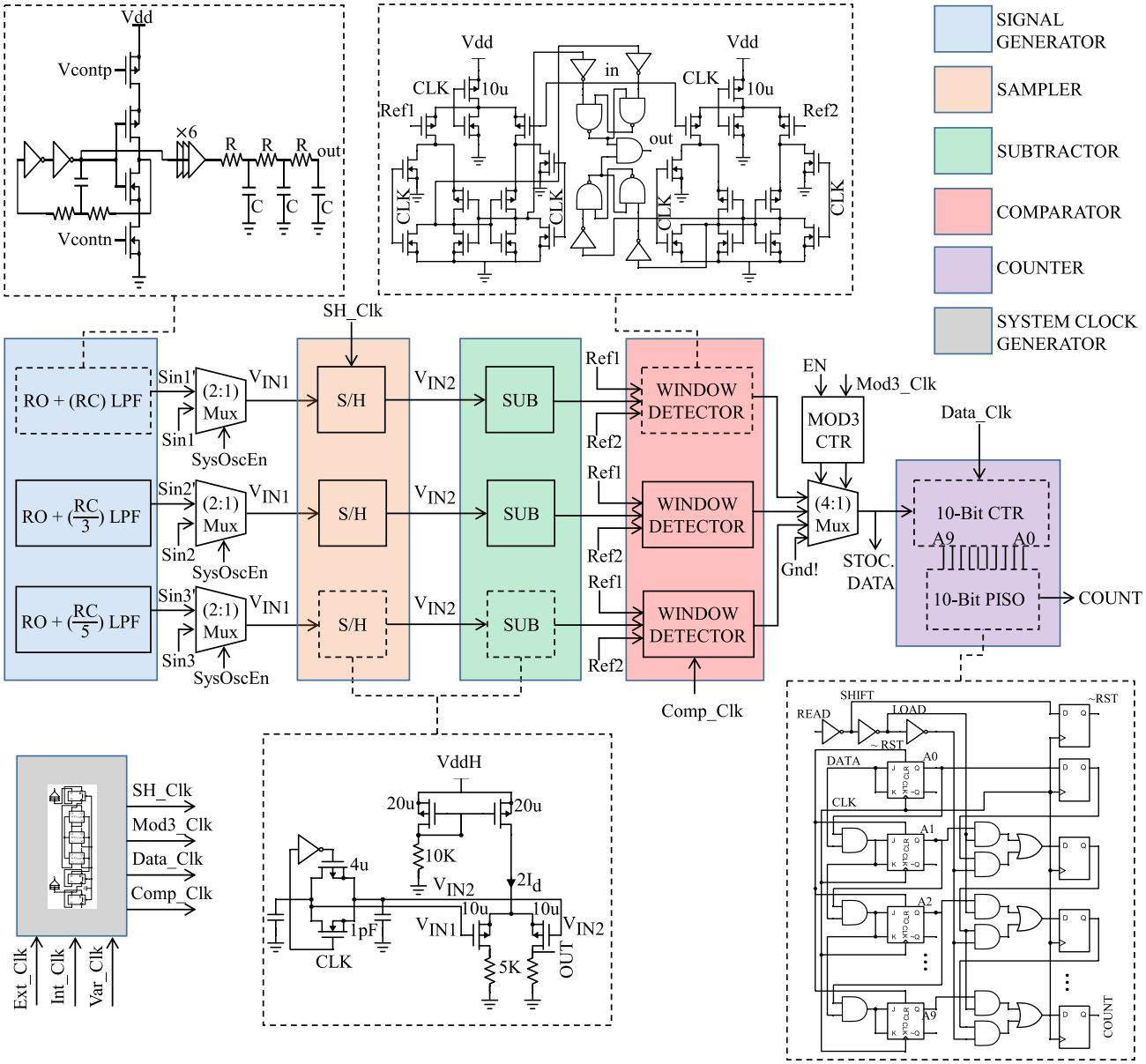


FIGURE 6. Architecture of STAMP.

resistance of sampling circuit should be below 16 kΩ within the required minimum bandwidth of 10 Mhz. Considering 10× design margin, W/L of 20 is used for both PMOS and NMOS transistors in the switch to cancel out clock feedthrough.

C. SUBTRACTOR BLOCK

The sampling circuit is followed by the voltage subtractor (SUB block) which is a single stage differential amplifier. Amplifier subtracts the analog and sampled signals to obtain sampling error. The gain (A_V) is simplified as $\triangle V_{OUT}/\triangle(V_{IN1} - V_{IN2})$ where V_{IN1} and V_{IN2} are the analog signal and sampled signal, respectively. A_V is set to 1. Transconductance is defined as $g_M = I_d/V_{GS} - V_T$. Therefore, RI_d should be equal to $V_{GS} - V_T$. Sampling error swing of 0.5 V around 0.5 V is targeted. Considering the threshold voltage of 0.7 V, subtract block uses a higher V_{DD} given by $V_{DDH} = V_{IN2} + V_{GS} + V_{DS}$. V_{DDH} should be $\geq$ 3.2 V if saturation drain-source voltage of the current mirror transistor is set as 1 V. Further analysis is done for calculating the aspect ratio (W/L) of each transistor such that drain current I_d is 500 $mV/5\ k\Omega = 100\ \mu A$. It is defined in the saturation region as $I_{dsat} = k'_p \frac{W}{2L}(V_{GS} - V_T)^2$ where the process transconductance k'_p is 85 uA/V^2. The aspect ratios of the amplifier and the current mirror transistors are set to 20 and 40, respectively. Subtraction block has 31 mV

TABLE 1. STAMP system clock partition.

	Frequency Division Factor	Signal
System Clock	1:1	Data_Clk, Mod3_Clk
	1:2	Comp_Clk
	1:40	SH_Clk
	1:2000	READ

subtraction offset and 150 Mhz f_{3dB} for 1 V reference voltage in post-layout simulations. Thus, the reference voltage should be offset by 31 mV to get desired probability.

D. COMPARATOR BLOCK

Probability conversion is done by window detectors each of which consists of double dynamic latch comparators followed by SR Latches. The minimum bandwidth of the detectors is set to 200 Mhz in order to generate up to 20 bits in each cycle of sampling clock. Thus, the switch-on resistor of the inverter where the clock is fed should be less than 795 Ω. The aspect ratio of p-transistor is determined as $W/L = 1/k'_p R_{on}(V_{GS} - V_T)$. Considering the design margin of 2×, (W/L) of 60 is used for all transistors which yields 1 mV selectivity for 0.5 V inputs (Ref1 and Ref2) at operating frequency of 200 Mhz.

Digital stochastic signals obtained by detectors are then combined in MUX, controlled by Mod3 counter which counts from (00) to (10) on binary form to mix the signals. In order to mix them properly, Mod3 counter clock (Mod3_Clk) needs to be at least 2× faster than the comparator clock (Comp_Clk).

E. COUNTER BLOCK

The output stochastic data is then counted by 10-bit synchronous counter and serialized by 10-bit parallel input - serial output (PISO) block to observe the probability of the stochastic bit stream which is controlled by the reference voltages of the detectors. The stochastic value of the bit stream is read by active high 'READ' signal. When READ is '1', 10 bits (A9…A0) from counters are loaded to PISO and after 1 clock cycle, the count is set to reset. Until the next high, the parallel bits are shifted to the output of PISO. To calculate the probability (P_{out}), frequency of READ signal can be set to thousandth of f_{Comp_Clk} and $P_{out} = dec(STOC)/1000$ where $dec(STOC)$ is the decimal value of 10 bit serial data read after the READ pulse. Further description of clock partition is given in Table 1.

F. SYSTEM CLOCK GENERATOR BLOCK

System clock generator block produces the needed clocks for sub-blocks of STAMP, such as sampling and hold clock (SH_Clk), comparator clock (Comp_Clk), Mod3 counter clock (Mod3_Clk) and stochastic data counter clock (Data_Clk) as shown in Fig. 6 (below-left). Block details are displayed in Fig. 7. 3 options are available for STAMP to get system clock: 1) it can get it from external source (Ext_Clk), 2) it can generate it internally (Int_Clk) by means of buffered ring oscillator, and 3) it can generate variable system clock

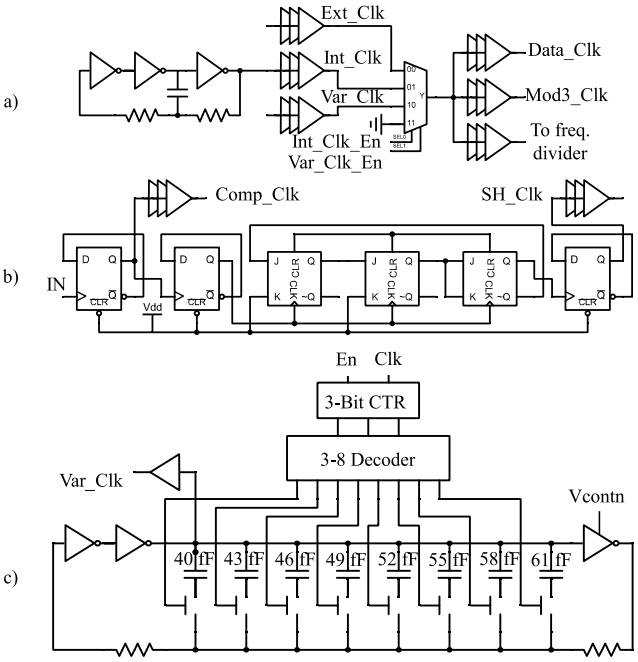


FIGURE 7. System clock generator: a) clock configuration, b) frequency division for comparator and sampling clocks, c) variable system clock generation.

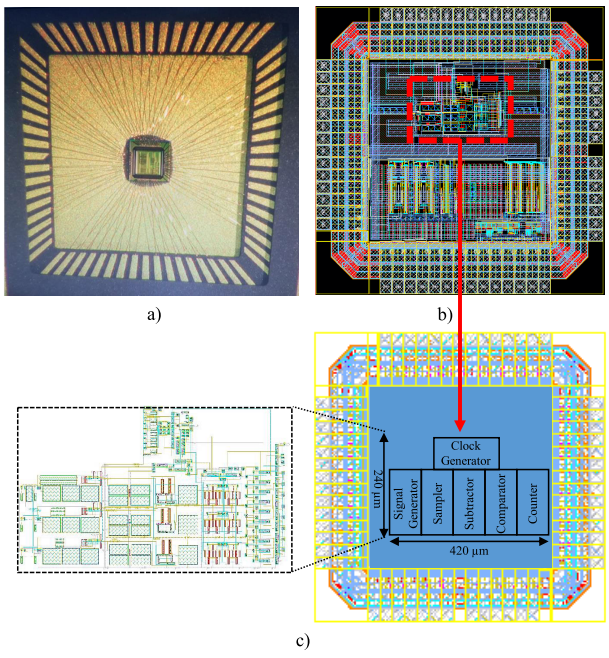


FIGURE 8. STAMP chip: a) STAMP wire bonding in CLCC 68-pin package, b) STAMP's routing, c) STAMP's block diagram and layout.

(Var_Clk). Fig. 7(a) shows the system clock configuration. For example, if Int_Clk_En is high and Var_Clk_En is low, internal constant clock is fed to network. The frequency divider is depicted in Fig. 7(b). Simply half and fortieth of the system clock are reserved for comparator clock (Comp_Clk) and sampling clock (SH_Clk), respectively. Fig. 7(c) depicts the variable system clock generation. We set

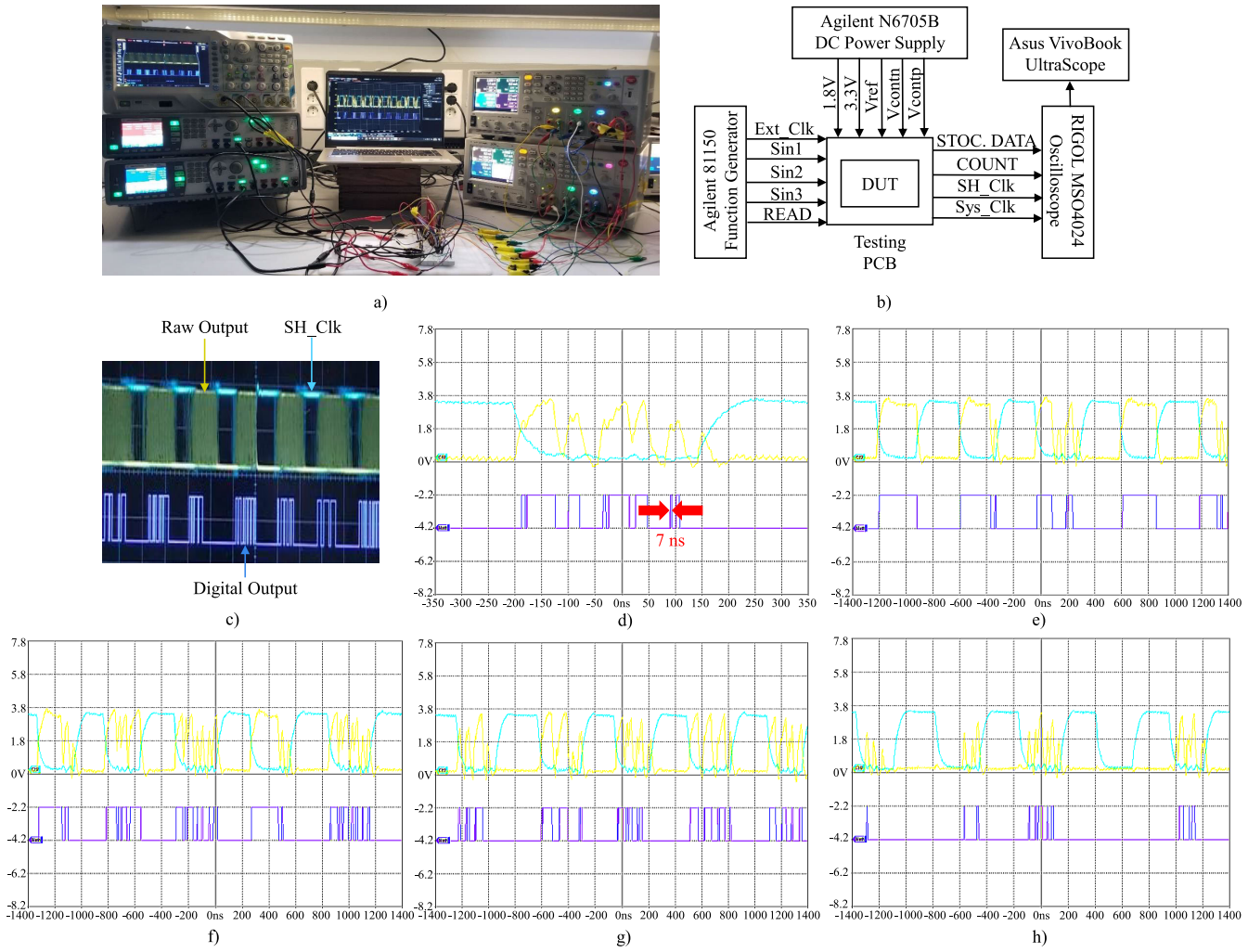


FIGURE 9. Test setup of STAMP: a) test setup view in VLSI Lab, b) test setup block diagram, c) oscillator output: raw output(yellow), sample and hold clock (cyan), digital output (purple), d) UltraScope views: minimum measured stochastic bit with, e-h) output waveform when Vref is set to 0.9 V, 0.75 V, 0.55 V, 0.3 V, respectively.

off from the idea of using stochastic sampling clock to enrich the randomness. To do so, we use capacitor bank which is scattered around 20% of the mean capacitor. In order to swing the frequency of system clock, one of 8 capacitors is selected by 3-8 decoder which decodes 3-bit counter. The counter is clocked by the third LSB (A2) of 10-bit synchronous counter of the stochastic data. In order to adjust the mean sampling frequency, delay is controlled by Vcontn signal of the third inverter. Once the Vcontn is fixed, sampling signal is swinging randomly around 20% of the mean sampling frequency.

IV. CHIP MEASUREMENTS AND RESULTS

STAMP is displayed in Fig. 8. It is fabricated in die area of 1.5 mm^2 with 180 nm CMOS process. Wire bonding of STAMP in 68-pin ceramic leaded chip carrier (68-CLCC) is shown in Fig. 8(a). Chip placement and layout are shown in Fig. 8(b) and Fig. 8(c), respectively. STAMP consumes core area of 0.11 mm^2. VLSI Lab test setup and its block diagram are shown in Fig. 9(a) and Fig. 9(b), respectively. Two function generators supply 3 sine-waves (Sin1, Sin2 and Sin3) and system clock (Ext_Clk) if STAMP uses sources externally. Two power supplies (5 terminals) feed STAMP chip as Vdd (1.8 V) for core, VddH-VddR-VddO (3.3 V) for subtractor, chip input-output pads, respectively, and Vcontp-Vcontn (Vdd-0) for controlling built-in oscillators. The raw output stochastic data (STOC. DATA) and sample and hold clock are observed by the oscilloscope which is connected to the local computer for capturing figures and registering the output waveforms in volts. System clock (Sys_Clk) and counted output (COUNT) can also be observed to ensure that STAMP is using internal or external sources and to be able to calculate output probability stand-alone. A capture of oscilloscope is displayed in Fig. 9(c). Since the chip is a prototype, many parameters are configurable. The built-in network is dedicated to generate sine-waves with frequencies of 1 Mhz, 3 Mhz and 5 Mhz. Internal system clock is measured as 281.7 Mhz which produces sampling clock of 7.042 Mhz. Given that the minimum pulse width of the stochastic bit stream is determined by comparator clock

(Comp_Clk), the output data is analyzed on UltraScope considering the stochastic bit rate to be about 140 Mhz (Fig. 9(d)). We register the signals of sampling clock and stochastic data and we capture 7M bits per frame. We control the probability (number of binary 1's in bit stream over length of the bit stream) with one reference voltage (the other reference is set to ground for simplicity). We scale 'Vref' from 0 to 1V to adjust the probability from 0 to 1 (100%) as shown in Fig. 9(e - h). Since we use sample and hold circuitry, we eliminate half of the bit stream keeping in mind that there is no data when sampling clock is set to high (sampling error is zero). In order to activate the other half, double sampling can be done by another sampler with an inversely connected clock. STAMP controls the probability of stochastic output in real-time with no area overhead. For example, when Vref is set to 0.9 V as shown in Fig. 9(e), the measured probability of the output stochastic signal is approximated to 90% or when it is set to 0.55 V, the measured probability is 50% (Fig. 9(g)). The output signal takes the form of sampling clock when Vref is set to 1 V. So, the stochastic bit stream has probability of 100% in this case.

Configurations used for testing include internal signal sources sampled with internal constant/variable system clock, external signal sources (Sin1, Sin2 and Sin3) in different frequencies sampled with internal/external constant/variable system clocks, etc. In every configuration, 3.5M bits having probability of 50% are analyzed and tested with NIST SP 800-22. We experience that STAMP passes most of NIST tests but can fail in some tests such as Runs and Longest Run. We figured out that the system clock partition is the reason. Recall from Table 1 that we record 20 bits in each cycle of sampling clock. This implies that STAMP may generate successive bits of binary 0 in some cases such as Vref being lower than the sampling error of the 3rd harmonic. To overcome this issue, we need to de-correlate bits from one sample with bits from another sample. For example, if we capture 10 groups of 1M output bits and do bit-wise XOR operation group by group, STAMP passes all NIST tests. From this we deduce that we need to de-correlate at least 10 samples each of which has 20 bits which makes total of 200 bits. In order to de-correlate 200 bits, we do bit-wise XOR operation for the output of STAMP with regular 8-bit LFSR which is dedicated to yield throughput of 256 pseudo-random bits as depicted in Fig. 10.

We compare STAMP with benchmark RNGs (Blum Blum Shub (BBS), Micali Schnorr (MS), etc.) and random data (*e*, *pi*, etc.) which are available in test suit. We also compare the output of MATLAB ('randi' function) and conventional 20-bit LFSR with ours. As proposed in the user manual, we generate 1M bits, 10 groups of 100K. Table 2 shows that STAMP passes all NIST tests (15/15) after post-processing. We follow a simple method for comparison. Since none of the benchmarks generate P-value for 10th test because of insufficient data, we average the P-value of 12 tests (10,12,13th tests are excluded) and we sum all of the tests which are passed out of the total tests (total proportion

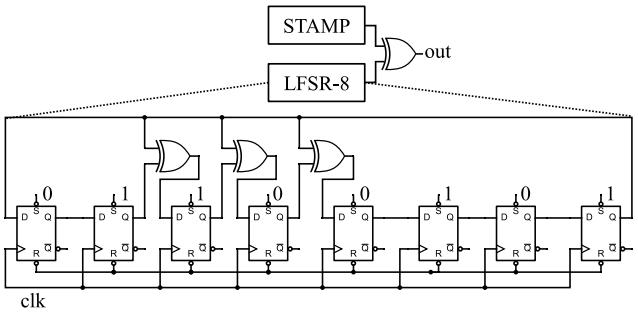


FIGURE 10. STAMP's post-processing.

TABLE 2. STAMP NIST SP 800-22 results in 1M bits (10 of 100K) and comparison w/ benchmark RNGs/Datasets.

STAMP		
Test	**P-Value**	**Result**
01-Frequency	0.7399	PASS
02-Block Frequency	0.3504	PASS
03-Cumulative Sums	0.8257	PASS
04-Runs	0.9114	PASS
05-Longest Run	0.9114	PASS
06-Rank	0.7399	PASS
07-FFT	0.2133	PASS
08-Non-Overlapping Template	0.4972	PASS
09-Overlapping Template	0.5341	PASS
10-Universal	PASS	PASS
11-Approximate Entropy	0.0668	PASS
12-Random Excursions	0.6313	PASS
13-Random Excursions Variant	0.4809	PASS
14-Serial	PASS	PASS
15-Linear Complexity	0.9114	PASS

BENCHMARK RNGs	**P - Value**	**TEST SCORE / OUT OF**
BBS	0.479	1595/1620
CCG	0.427	1679/1698
G-SHA1	0.08	373/1620
LCG	0.464	1626/1646
MODEXP	0.438	1657/1672
MS	0.434	1597/1620
QCG1	0.462	1651/1672
QCG2	0.476	1599/1620
XOR	0	215/1620
e	0.457	1602/1620
pi	0.539	1656/1672
sqrt2	0.523	1621/1646
sqrt3	0.486	1620/1646
LFSR-20	0.155	1629/1672
MATLAB	0.5	1732/1750
STAMP	**0.558**	**1623/1646**

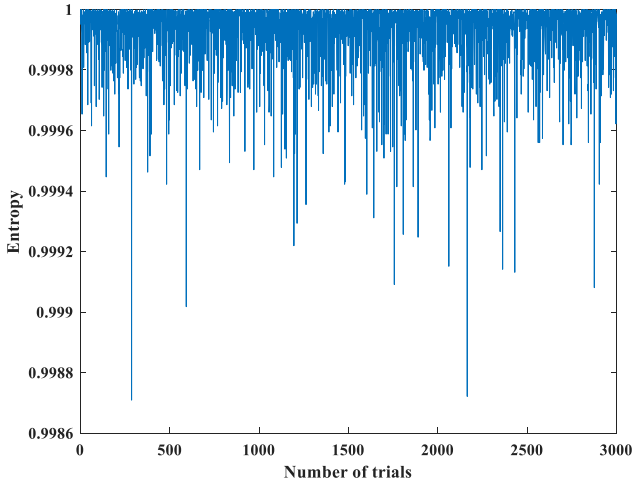


FIGURE 11. Entropy of output across trials.

of 15 tests). Table 2 shows that STAMP generates the highest quality of stochastic bit sequence. We also recorded 30M bits and run 1000 sequence of 30K bits. STAMP passes all NIST tests. Fig. 11 shows entropy of the output. It corresponds Shannon entropy at nominal conditions. Mean entropy is measured > 0.999924 in 3000 trials. The minimum entropy is measured as > 0.9987. This shows that the randomness of the output is in acceptable level ($\simeq 0.999$) even if in some cases where a low jitter in the sampling clock occurs.

Table 3 compares randomness quality of STAMP with the-state-of-the-art FPGA-based TRNGs. FPGA-based LFSRs aren't considered for comparison because of their low NIST results as seen in [17]. This conforms with the 20-bit LFSR

TABLE 3. NIST comparison of STAMP with the state-of-the-art FPGA-based TRNGs.

RNGs	PRNG1 (Chua	PRNG2 (Rössler)	PRNG3 (Lorenz)	PRNG4 (FDRN)	PRNG5	PRNG6	STAMP (this work)
NIST Pass	14/15	5/15	15/15	10/15	14/15	14/15	15/15
NIST Test Score	0.9849	0.5126	0.9836	0.9467	0.9848	0.9746	0.9860 (1623/1646)

PRNG1-PRNG6 [15]

TABLE 4. STAMP performance comparison w/ state-of-the-arts.

Design	STAMP	REF [20]	REF [4]	REF [5]	REF [12]	REF [6]
Technology (nm)	180	180	180	130	65	40
Supply Voltage(V)	1.8	1.8	1.8	1.2	0.9	VDD***
Entropy Source	Digital Jitter and Noise	Digital Jitter	ReRAM Jitter	ReRAM Metastability	Digital Jitter and Noise	ReRAM Noise
Bit Rate (Mb/s)	140	1.08	8	N.A.	52	32
NIST Pass	15	15	15	12	15	15
Area (mm^2)	0.11	0.007	0.0045	0.25	0.06	N.A.
Normalized Area*	24	1.5	1	105	100	N.A.
Power Efficiency (pJ/bit)	77	101	120	138	6.9	40
Normalized Power Efficiency **	1.4	1.8	2.2	8.5	1	4.2
Probability Control	YES	NO	NO	NO	NO	NO
Post Process	YES	NO	NO	YES	NO	YES

*Technology Scaling[2] is considered
**Voltage Scaling[3] is considered
***Tech. process voltage of 1 V is referred

(LFSR-20) randomness test result shown in Table 2. The reason is that the random bits are strongly correlated in the shifting process of LFSR. On the other hand, STAMP outdoes the FPGA-based chaotic TRNGs (PRNG1-PRNG6) [15] in terms of NIST test pass and score in which the STAMP's rates are imported from Table 2.

Table 4 compares STAMP's overall performance with the-state-of-the-arts. For comparison, we consider CMOS-compatible ReRAM and jitter-based TRNGs. We have compared RNGs in terms of area, speed and power consumption. Normalized area and power are added into the figure of merits for offsetting the technology used in the design. Furthermore, we have analyzed if referenced RNGs need post-processing and if output probability is controlled.

For fair comparison we have considered only ASIC-based RNGs in Table 4 since FPGA-based RNGs cannot readily embedded into stand-alone system, thus the power or the area consumption comparison is not applicable between FPGA and ASIC-based RNGs.

STAMP produces stochastic number 130× faster and 1.3× more efficient than [20], 18× faster and 1.6× more efficient than [4]. STAMP is 6× and 3× more efficient than [5] and [6], respectively. In comparison with [12], STAMP is 2.7× faster and consumes 4× less area with energy overhead of 1.4× in the worst case scenario.

For cryptography applications, STAMP is superior to referenced TRNGs since it is more efficient than the most of the RNGs and the fastest (∼3× the closest [12]) while bearing the same test result of NIST. It means that it is harder for the attackers to sync in with the code that STAMP produces. Furthermore, STAMP has unique feature of ability to control output probability in real-time. To keep up with this STAMP feature, the referenced TRNGs need to add additional hardware which may even decrease the efficiency. Also, controlling true random source is a challenging task because of unbounded nature of noise source. So, apart from the referenced RNGs, STAMP is a compact RNG which can be used for both the stochastic circuits and the cryptography applications.

V. CONCLUSION

Random number generators are crucial part of the stochastic computing systems and cryptography applications. RNG should efficiently produce rich-in random numbers to not to reverse the early benefits such as low-power and simplicity of stochastic circuits. In this paper we proposed STAMP as a low-power and high quality RNG for stochastic circuits.

STAMP is a compact RNG, having ability to control output probability in real-time, which can replace the LFSRs that have common problem of poor randomness and latency in stochastic circuits. STAMP generates random numbers at the rate of multiple times of sampling frequency which unleashes the throughput problem of TRNGs such as ADC-DAC based TRNG.

As a future work, we will continue to adapt the proposed work into recent technologies and applications in searching for utmost limits of stochastic circuits.

REFERENCES

[1] J. Hu, Z. Zhang, and Q. Pan, "A 15-Gb/s 0.0037-mm^2 0.019-pJ/bit full-rate programmable multi-pattern pseudo-random binary sequence generator," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 67, no. 9, pp. 1499–1503, Sep. 2020.

[2] S. A. Salehi, "Low-cost stochastic number generators for stochastic computing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 4, pp. 992–1001, Apr. 2020.

[3] V. Sehwag, N. Prasad, and I. Chakrabarti, "A parallel stochastic number generator with bit permutation networks," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 65, no. 2, pp. 231–235, Feb. 2018.

[4] J. Yang, Y. Lin, Y. Fu, X. Xue, and B. A. Chen, "A small area and low power true random number generator using write speed variation of oxidebased RRAM for IoT security application," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2017, pp. 1–4.

[5] H. Aziza, J. Postel-Pellerin, H. Bazzi, P. Canet, M. Moreau, V. D. Marca, and A. Harb, "True random number generator integration in a resistive RAM memory array using input current limitation," *IEEE Trans. Nanotechnol.*, vol. 19, pp. 214–222, Mar. 2020.

[6] Z. Wei, Y. Katoh, S. Ogasahara, Y. Yoshimoto, K. Kawai, Y. Ikeda, K. Eriguchi, K. Ohmori, and S. Yoneda, "True random number generator using current difference based on a fractional stochastic model in 40-nm embedded ReRAM," in *IEDM Tech. Dig.*, Dec. 2016, pp. 4.8.1–4.8.4.

[7] G. D. P. Stanchieri, A. De Marcellis, E. Palange, and M. Faccio, "A true random number generator architecture based on a reduced number of FPGA primitives," *AEU Int. J. Electron. Commun.*, vol. 105, pp. 15–23, Jun. 2019.

[8] Ü. Güler and S. Ergün, "A high speed, fully digital IC random number generator," *AEU Int. J. Electron. Commun.*, vol. 66, no. 2, pp. 143–149, Feb. 2012.

[9] J.-C. Hsueh and V. H.-C. Chen, "An ultra-low voltage chaos-based true random number generator for IoT applications," *Microelectron. J.*, vol. 87, pp. 55–64, May 2019.

[10] N. Miura, M. Takahashi, K. Nagatomo, and M. Nagata, "Chip-package-board interactive PUF utilizing coupled chaos oscillators with inductor," *IEEE J. Solid-State Circuits*, vol. 53, no. 10, pp. 2889–2897, Oct. 2018.

[11] M. Kim, U. Ha, K. J. Lee, Y. Lee, and H.-J. Yoo, "A 82-nW chaotic map true random number generator based on a sub-ranging SAR ADC," *IEEE J. Solid-State Circuits*, vol. 52, no. 7, pp. 1953–1965, Jul. 2017.

[12] S. T. Chandrasekaran, V. E. G. Karnam, and A. Sanyal, "0.36-mW, 52-mbps true random number generator based on a stochastic Delta–Sigma modulator," *IEEE Solid-State Circuits Lett.*, vol. 3, pp. 190–193, 2020.
[13] E. Fatemi-Behbahani, E. Farshidi, and K. Ansari-Asl, "Analysis of chaotic behavior in pipelined analog to digital converters," *AEU Int. J. Electron. Commun.*, vol. 70, no. 3, pp. 301–310, 2016.
[14] A. Senouci, H. Bouhedjeur, K. Tourche, and A. Boukabou, "FPGA based hardware and device-independent implementation of chaotic generators," *AEU Int. J. Electron. Commun.*, vol. 82, pp. 211–220, Dec. 2017.
[15] A. A. Rezk, A. H. Madian, A. G. Radwan, and A. M. Soliman, "Multiplierless chaotic pseudo random number generators," *AEU Int. J. Electron. Commun.*, vol. 113, Jan. 2020, Art. no. 152947.
[16] M. O. Meranza-Castillón, M. A. Murillo-Escobar, R. M. López-Gutiérrez, and C. Cruz-Hernández, "Pseudorandom number generator based on enhanced Hénon map and its implementation," *AEU Int. J. Electron. Commun.*, vol. 107, pp. 239–251, Jul. 2019.
[17] P. Zode, P. Zode, and R. Deshmukh, "FPGA based novel true random number generator using LFSR with dynamic seed," in *Proc. IEEE 16th India Council Int. Conf. (INDICON)*, Dec. 2019, pp. 1–3.
[18] T. Zhou, Y. Ji, M. Chen, and Y. Li, "PL-MRO PUF: High speed pseudo-LFSR PUF based on multiple ring oscillators," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Oct. 2020, pp. 1–5.
[19] A. K. Panda, P. Rajput, and B. Shukla, "FPGA implementation of 8, 16 and 32 bit LFSR with maximum length feedback polynomial using VHDL," in *Proc. Int. Conf. Commun. Syst. Netw. Technol.*, May 2012, pp. 769–773.
[20] K. Yang, D. Blaauw, and D. Sylvester, "An all-digital edge racing true random number generator robust against PVT variations," *IEEE J. Solid-State Circuits*, vol. 51, no. 4, pp. 1022–1031, Apr. 2016.

MAHMUT BURAK KARADENIZ received the B.S. degree in electronics engineering from Turkish Naval Academy, Istanbul, Turkey, in 2011, and the M.S. degree in electrical engineering from the University of Washington, Seattle, USA, in 2015. He is currently pursuing the Ph.D. degree in electronics and communication engineering with Istanbul Technical University (ITU), Istanbul.

Since 2016, he has been the Director of electronic warfare systems at Istanbul Naval Shipyard, Istanbul. He is a member of the Emerging Circuits and Computation (ECC) Group, ITU. His research interests include very large-scale integration circuits, stochastic computing, and energy efficient artificial intelligence hardware designs.

ISMAIL CEVIK received the B.Sc. degree in electrical engineering from Bogaziçi University, Istanbul, Turkey, in 2003, the M.Sc. degree in electrical engineering from the University of Southern California, Los Angeles, CA, USA, in 2004, and the Ph.D. degree in electrical engineering from the University of Idaho, Moscow, ID, USA, in 2014.

He is currently working as an Assistant Professor with the Electronics and Communication Engineering Department, Istanbul Technical University, Istanbul. His research interests include high-speed analog and mixed-signal integrated circuit (IC) design, CMOS image sensors, ultra-low power IC design, micro solar cell structures, energy efficient DC-DC converters, and high-speed data converters.

MUSTAFA ALTUN received the B.Sc. and M.Sc. degrees in electronics engineering from Istanbul Technical University, in 2004 and 2007, respectively, and the Ph.D. degree in electrical engineering from the University of Minnesota, in 2012, with a minor in mathematics.

Since 2012, he has been working as a Faculty Member of electrical engineering at Istanbul Technical University. He currently runs the Emerging Circuits and Computation (ECC) Group, Istanbul Technical University. He has been serving as a Principal Investigator/Researcher for various projects, including EU H2020 RISE Project, the National Science Foundation of USA (NSF) Project, and TUBITAK Project. He is the author of more than 60 peer-reviewed papers and a book chapter. He was a recipient of the Science Academy Young Scientist Award, the TUBITAK Success Award, the TUBITAK Career Award, and the Werner von Siemens Excellence Award.