Weight Update Generation Circuit Utilizing Phase Noise of Integrated Complementary Metal–Oxide–Semiconductor Ring Oscillator for Memristor Crossbar Array Neural Network-Based Stochastic Learning

Woorham Bae* and Kyung Jean Yoon*

Herein, a robust programmable stochastic weight generation method for a memristive neural network is proposed. There have been few prior algorithm suggestions for crossbar neural network-based stochastic learning; however, there has not been much attention focussed on robust physical implementations. As a result, coming up with a robust method to provide the probability generator is an essential knob for its physical implementation. Here, implanting such stochastic behavior into the weight update signal itself is proposed, by multiplying it with the randomized probability sequence. To generate such probability sequence, bang-bang dithering of a phase-locked loop (PLL) with a binary phase detector (PD) is used. The programmable probability is enabled by introducing an offset for the PD outputs. Yet the dithering sequence has deterministic nature, phase noise of complementary metal–oxide–semiconductor (CMOS) ring oscillator to randomize the deterministic dithering is exploited. As a result, this lower power oscillator offers a better probability sequence, which enables an ultralow power circuit implementation.

1. Introduction

Over the past 60 years, shrinking in the channel length of complementary metal–oxide–semiconductor (CMOS) transistors (Moore’s law) has driven major advances on computing power on the Earth. However, due to the power wall and rising leakage current, the improvement of microprocessor’s computing capability has been retarded. In addition, the scaling itself has almost reached the fundamental limit, so there have been huge demands on finding an alternate way of improving the computing capability. Neuromorphic network is one of the top prior candidates, based on the observation that the human brain outperforms computers in many computational tasks while consuming only a fraction of the energy that artificial computers require to accomplish such tasks. However, it turns out that traditional CMOS device and circuit is not an efficient vehicle for the brain-inspired computing, mainly because of their digital-oriented nature.

Memristors, in contrast, are very promising candidates because of their capability to store analog weights, in a very similar manner as how the biological synapses work. That is, the synaptic learning associated with its diffusive nature can be effectively emulated by memristors for their unique ionic migration-driven mechanisms, which otherwise would have required extra circuits to cope with the digital nature of the conventional charge-based devices. Its simple two-terminal structure is compatible to crossbar array (CBA) integration, which itself has the inherent vector-multiplication nature. In addition, the crossbar technology is very attractive for its great compatibility with the CMOS logic process. Integrating the memristive CBA (or cross-point array) with CMOS circuits would be a part of realizing non-Von Neumann architectures such as biological neuronal computation, while minimizing power consumption required for transferring weight values, as shown in Figure 1. Such architecture-based computing, often referred to as an integrated processing-in-memory (PIM), or computing-in-memory (CIM) have been demonstrated in the state-of-thearts works, with up to 1 MB memristor CBA, while showing

Dr. W. Bae
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, CA 94720, USA
E-mail: wrbae@eecs.berkeley.edu

Dr. W. Bae
Circuits Department
Ayar Labs
Santa Clara, CA 95054, USA

Dr. K. J. Yoon
Non-Volatile Memory Solutions Group (NSG)
Intel Corporation
Santa Clara, CA 95054, USA
E-mail: kyung.jean.yoon@intel.com

DOI: 10.1002/aisy.202000011
1000× improvement in energy consumption relative to the conventional microprocessors.\textsuperscript{[9–11]} Furthermore, much higher volume integration of CBA (32 GB) and CMOS circuitry has been demonstrated by Liu et al.\textsuperscript{[12]}

For a successful training, initial weight needs to be randomized in conventional neural networks.\textsuperscript{[11,14]} However, the weight initialization is not required for the stochastic learning, where the weight is updated in a stochastic manner based on the updating probability.\textsuperscript{[13–15]} Although previous studies\textsuperscript{[13,16]} proposed utilizing the inherent cycle-to-cycle variation of a memristor device to obtain such stochastic behavior, apparently its demonstration is yet to be presented, given the high cell-to-cell variation of memristors, and the nonuniform voltage drop across the CBA, which exacerbates with increasing array size and scaling due to the presence of array leakage currents.\textsuperscript{[17]} This poses a huge concern especially for an analog switching. In addition, there is little room for adjusting the probability as desired based on this method. Therefore, such method relying on the cell property faces challenges toward its real implementation. For a better controllability and the constant probability across the array, a robust and programmable probability generator in the macro, as shown in Figure 1, would be highly anticipated. By multiplying the probability sequence with the synaptic weight update, a stochastic update is enabled. Ideally, the power consumption of such probability generator should be negligible to sustain the energy efficiency of the neural network. In this article, we propose a robust stochastic weight generation technique exploiting phase noise of an electrical oscillator. As we are making use of the phase noise rather than trying to suppress it, the power consumption of the oscillator is dramatically reduced compared with the conventional methods.

2. Phase Noise of CMOS Ring Oscillator

CMOS ring oscillator is one of the most famous electrical oscillators for its simple structure and compact areal density. In contrast, phase noise refers to as frequency-domain representation of rapid, short-term (i.e., >10 Hz), random fluctuations in the phase of a periodic wave (clock). It is known that ring oscillators exhibit poor phase noise compared with other oscillators (i.e., LC oscillator). As the motivation of this work is to utilize the phase noise of a CMOS ring oscillator, phase noise of a CMOS ring oscillator is reviewed in this section, based on a linear time-invariant (LTI) analysis for simplicity. Figure 2 shows a circuit diagram of a ring oscillator and how the noise of CMOS transistors deviates the clock edge from an ideal position. When an additive noise current $i_n$ is injected as shown in Figure 2, the delay variation of each stage introduced by $i_n$ is given as

$$\Delta \tau = \Delta f = \frac{CV_{DD}}{2I} = \frac{CV_{DD}}{2I} - \frac{CV_{DD}}{2(I + i_n)} \approx \frac{CV_{DD}}{2I} \times \frac{i_n}{I^2} \quad (1)$$

where $C$, $V_{DD}$, and $I$ represent node capacitance, supply voltage, and current of the inverter, because the delay of a single-stage $\tau$ is $CV_{DD}$, assuming the magnitude of $i_n$ is much smaller than the DC current. Note that here we use the first-order waveform approximation of edge transition for simplicity. Then, because the frequency of a ring oscillator is $1/2\pi\tau$, the frequency variation is calculated to be

$$\Delta f = \frac{1}{2N\tau} - \frac{1}{2N(\tau + \Delta \tau)} \approx \frac{1}{2N} \times \frac{2}{CV_{DD}} \times i_n \quad (2)$$

where $N$ is the number of stages in the ring oscillator. Then, phase modulation ($\phi$) is calculated by integrating $\Delta f$

$$\phi(t) = 2\pi \int_0^t \Delta f \, dx = 2\pi \times \frac{1}{2N} \times \frac{2}{CV_{DD}} \times \int_0^t i_n \, dx \quad (3)$$

From Equation (3), we can obtain power spectral density (PSD) of ($\phi$) as

$$S_{\phi}(f) = \left(2\pi \times \frac{1}{2N} \times \frac{2}{CV_{DD}}\right)^2 \times S_{i_n}(f) \quad (4)$$
where

$$S_W(f) = S\left(\int_0^t i_n dx\right)(f)$$  \hspace{1cm} (5)$$

Assuming that $i_n$ is a white noise, $S_w(f)$ becomes as

$$S_w(f) = \left(\frac{1}{2\pi f}\right)^2 \times S_i(f)$$  \hspace{1cm} (6)$$
because the integral of white noise is a Wiener process. Note that $S_i(f)$ equals to the PSD of the white noise of the transistor, which is given to

$$\frac{i_n^2}{\Delta f} = 4kTgy_m = 4k\gamma \times \frac{2I}{V_{DD} - V_{th}}$$  \hspace{1cm} (7)$$

where $k$, $T$, $\gamma$, and $g_{m}$ denote the Boltzmann’s constant, the absolute temperature, the excess noise coefficient of transistor, and the total transconductance of N-type metal–oxide–semiconductor (NMOS) and P-type metal–oxide–semiconductor (PMOS) transistors, respectively. Substituting Equation (6) and (7), Equation (4) can be written as

$$S_{\phi}(f) = \frac{8kT\gamma}{I(V_{DD} - V_{th})} \times \left(\frac{f_0}{f}\right)^2$$  \hspace{1cm} (8)$$

where $f_0$ is the oscillation frequency ($=1/2\pi r$). Note that linear time-variant (LTV) analysis of ref. \cite{14} gives the same result. Equation (8) shows the primary design trade-off of a ring oscillator; with the fixed oscillation frequency, the phase noise is inversely proportional to the current consumption and the supply voltage of a ring oscillator. In other words, a higher current consumption or a higher supply voltage is required to improve the phase noise, which inevitably increases the power consumption. As the focus of this work is to utilize the phase noise than to suppress it, it enables the implementation of an extremely low-power ring oscillator.

**Figure 3** shows the simulated phase noise of four ring oscillators used in this work. The ring oscillators are designed in commercial 45 nm CMOS technology where the nominal supply voltage is 1.0 V. However, 0.5 V supply voltage is used for the ring oscillators to minimize the power consumption at the cost of poor-phase noise. A two-stage topology presented by Bae et al.\cite{15} is adopted for maximizing the frequency at such low supply voltage. The $1 \times$ ring oscillator is designed with the minimum size transistors allowed in the given technology and with a variable load capacitance per stage (8fF–13fF), which is used to tune the frequency. The transistor and the total capacitance are linearly scaled to build $3 \times$, $9 \times$, and $27 \times$ designs while keeping the same oscillation frequency. As expected from Equation (8), therefore, the simulated phase noise is inversely proportional to the power consumption of the ring oscillator.

### 3. Bang-Bang Phase Detector and Phase-Locked Loop

As observed in Equation (8) and Figure 3, the frequency instability of an oscillator, which diverges at a low frequency, limits the use of a standalone oscillator in practical application. Instead, a phase-locked loop (PLL) forms a negative feedback loop where the oscillator clock is frequency- and phase-locked to a reference signal. Just like any other feedback loop, a PLL is composed of producer, sensor, and loop filter. As shown in the simplified block diagram of PLL in **Figure 4**, the voltage-controlled oscillator (VCO), which generates a clock whose frequency that is controlled by a voltage input, serves as the producer. Also, the phase detector (PD), which measures the phase difference between the VCO clock and the reference clock, is used for the sensor of the loop. The loop filter determines how to control the producer based on the measured value from the sensor. Because a PLL controls the frequency of the VCO to match the phase and frequency of the reference clock and the output clock, the loop filter needs to adjust both the phase and frequency so that the system should be second order or higher. Here, there are two possible implementations of the PD; linear PD and binary PD. The output of linear PD is proportional to the input phase difference; however, the binary PD only detects the polarity of the phase difference. This kind of binary quantized PD is referred to as a bang-bang PD (BBPD). Despite the shortcomings from the nonlinear nature, the BBPDs are widely used because of its high-speed capability, digital-friendly structure, and simple/low-power operation. An example of BBPD is shown in **Figure 4**. A simple D-flipflop (D-FF), where one clock is
sampled by another clock, can serve as a BBPD. When the sampled clock (CK A) leads the sampling clock (CK B), the output becomes logical high, and vice versa, so the D-FF tells which clock leads the other.

Because of the quantization nature, a BBPD never produces zero output so a bang-bang phase-locked loop (BBPLL) cannot lock to an exact phase even though we neglect any nonideal conditions, unlike a linear PLL. Instead, a BBPLL achieves lock by dithering its phase around the intended phase. Figure 5 shows the waveform of the BBPD output and the phase error of the BBPLL. Because the output (Q) of the D-FF goes to logical high when the input clock leads the VCO clock and then the loop filter makes the VCO clock faster, the Q and Q-bar are referred to as UP and DN, respectively. Neglecting nonidealities such as the VCO phase noise and the D-FF metastability, the UP and DN pulses are triggered alternately to locate the zero-phase error in between the dithered phase error.

4. Stochastic Weight Generation Based on Phase Noise

4.1. Pattern Scrambling Using Phase Noise on BBPD Output

From the observation in Section 3, one may consider taking the BBPD output from a BBPLL, as a probability signal \( p = 0.5 \). However, even though the BBPLL example in Figure 5 produces the expected value of 0.5 at the output of the BBPD, it is hard to say that it produces a probability of 0.5 because it is a deterministic “0101” pattern instead of a randomized sequence. The phase noise of the VCO can be utilized to scramble the pattern. That is, the phase noise introduces uncertainty on the VCO output clock, and then the uncertainty propagates to the BBPD output. To verify the scrambling effect of the oscillator phase noise on the BBPD, a BBPLL is simulated using the ring oscillators described in Section 2. The ring oscillator circuits are generated from an analog-circuit generator based on Berkeley Analog Generator (BAG) framework, and Verilog-A models are used for the rest of BBPLL components for the simulation program with integrated circuit emphasis simulation (SPICE).

The VCO functionality is enabled by controlling the variable load capacitance. The BBPLL runs at 2 GHz frequency while powered from 0.5 V supply voltage. The loop filter is designed to have phase margin of \( \approx 60^\circ \). Figure 6a shows a fraction of the transient waveform of the BBPD output with respect to the existence of the phase noise. As described in Section 3, it always shows a fixed 0101 pattern if there is no noise. However, it is observed that the pattern is scrambled when the transistor noise of the ring oscillator is included. Figure 6b shows the histogram of the average of the BBPD output (UP) over the 200 clock cycles, where it follows a Gaussian-like distribution. Anderson–Darling test over Gaussian distribution for the bare data of Figure 6b is shown in Figure 6c. The Anderson–Darling test gives \( p \) value of 0.795, which is enough to justify the observation that this process follows Gaussian distribution.

To fully verify the scrambling effect of the phase noise, the PSD of the BBPD output has been evaluated with respect...
to the phase noise of the oscillator (Figure 7). Without the noise, the PSD exhibits almost pure single tone at 1 GHz, which is half the clock frequency. However, as the power consumption of the ring oscillator decreases (in other words, as the phase noise of the ring oscillator is degraded), the power of the main spectral tone decreases, whereas that of the other frequency component increases. Note that the total amount of the signal power keeps the same regardless of the ring oscillator because the BBPD output has the same signal amplitude of VDD as shown in Figure 6a. In the case of the 1× oscillator, the PSD is quite flat; therefore, it becomes like a white noise (the main tone over the noise floor is less than 20 dB), which proves that it is sufficiently randomized. It is important to note that such randomization is obtained by reducing the power consumption of the oscillator, so it enables an extremely low-power implementation. For instance, the BBPLL with the 1× oscillator dissipates only 118 μW (from 0.5 V supply, 38 μW from the 1× oscillator) for producing a 1 GB s⁻¹ random probability stream (0.118 pJ bit⁻¹) and gives

Figure 6. a) Simulated BBPD output with respect to the ring oscillator (1×) phase noise; b) histogram of expected value of UP over 200 cycles with respect to the phase noise; c) Anderson–Darling test for normality check with the bare data of part (b).

Figure 7. PSD of BBPD output with respect to the ring oscillator power consumption and phase noise.
the best PSD profile than the other oscillators. Compared with the conventional random number generators whose energy efficiency is several hundreds of picojoule per bit (i.e., 275 \textsuperscript{[21]} and 400 pJ bit \textsuperscript{[22]}), the proposed scheme achieves more than 1000\times improvement on energy efficiency due to its inherent low-power nature. In fact, the traditional random number generators mainly focus on achieving pure randomness, so they pursue 1/0 probability of 0.5 while nullifying the autocorrelation. In contrast, the proposed scheme focuses more on generating a controllable 1/0 probability, which is purely deterministic without any noise. We utilize the phase noise of ring oscillator to scramble the deterministic sequence, hence enabling substantial degree of randomness that is not as perfect as the random number generators.

4.2. Programmable Probability Based on UP/DN Weight Control

In Section 4.1, it is verified that an extremely low-power BBPLL can be used to scramble the weight function while keeping the fixed probability. However, in the conventional BBPLL, only the probability of 0.5 is available. In this subsection, a technique for enabling programmable probability is presented. Figure 8a shows a block diagram of BBPLL where different weight constants (\(\alpha\), \(\beta\)) are introduced for the BBPD output, UP and DN. For example, when \(\alpha = 1/3\) and \(\beta = 2/3\), two UP pulses are equivalent to a single DN pulse, in terms of the amount of the updated phase at the VCO clock. In other words, at the steady state of BBPLL, it should meet the following criteria

\[ \beta \times P_{\text{UP}} = \alpha \times P_{\text{DN}} \]

(9)

\[ P_{\text{UP}} + P_{\text{DN}} = 1 \]

(10)

where \(P_{\text{UP}}\) and \(P_{\text{DN}}\) represent the probability of appearance of UP and DN at each bang-bang decision, respectively. Therefore, adjusting the weight of UP/DN enables the programmable probabilities \(P_{\text{UP}}\) and \(P_{\text{DN}}\). Combined with the scrambling of UP/DN sequence by the phase noise, the BBPLL produces a randomized pattern with a programmable probability (i.e., \(P_{\text{UP}} = \alpha/\alpha + \beta\)). In a digital PLL, it is very easy to include such UP/DN mismatch in a digital loop filter. In an analog PLL, it is also easy to implement by introducing the current mismatch to the charge pump \textsuperscript{[23]}.

These blocks are responsible for resolution of the probability control, so there is a design trade-off between the resolution and the hardware overhead. For example, if a digital loop filter is adopted, the resolution is limited by the number of bits of the filter. For the charge-pump, unintended current offset due to finite output impedance or random mismatch may limit the resolution so additional power or area (i.e., feedback amplifier) is consumed to suppress such offset. Figure 8b shows the simulated histogram of the averaged out BBPD UP over the 200 clock cycles with \(P_{\text{UP}}\) of 0.8, where the scrambling by the phase noise is

---

Figure 8. a) Probability adjustment using different UP/DN weights; b) histogram of expected value of UP over 200 cycles with respect to the phase noise at 80% probability setup (\(\alpha = 0.2, \beta = 0.8\)); c) Anderson–Darling test for normality check with the bare data of part (b).
Figure 9. PSD of BBPD output with respect to the ring oscillator phase noise and the probability setup.

(a)

(b)

Figure 10. Jitter histogram with respect to the phase noise of ring oscillator. a) $p=0.5$, b) $p=0.8$. 
observed. Anderson–Darling test over Gaussian distribution is shown in Figure 8c. Figure 9 shows the simulated PSD for different probability setups. Ideal BBPD sequences are also shown in Figure 9, which are set to have its average being equal to the given probability. In all examples, nevertheless, we find that the power of the main tones decreases, whereas the noise floor increases as the power consumption of the oscillator decreases, which evidences that the learning sequence is being randomized, similar to the conventional BBPLL example ($\alpha = 0.5$, $\beta = 0.5$).

The jitter histograms for $p = 0.5$ and 0.8 cases shown in Figure 10 demonstrates that the uncertainty introduced from the ring oscillator propagates to the BBPD. As shown in Figure 10a, if the phase noise is small enough (27× ring), there are two hills in the histogram due to the 0101 bang-bang dithering (Figure 5), which means that the jitter is quite deterministic. However, as the phase noise level increases, the jitter histogram spreads, which implies that the jitter becomes randomized rather than deterministic. Because this randomized jitter is fed to the BBPD, such uncertainty propagates to the BBPD output which we want to exploit as a scrambled probability sequence. This observation gives a good explanation why the PSD of BBPD output becomes noisy in Figure 10 as the phase noise increases. Another example of $p = 0.8$ is shown in Figure 10. Because this case produces 01111 sequence in the noiseless condition, there are five quantized jitter locations. Same as the $p = 0.5$ case, the jitter profile is randomized as the phase noise increases, in accordance with the PSD of Figure 9. Meanwhile, as the bang-bang dithering jitter dominates the total jitter in the given design parameter, no significant jitter increase is observed when the phase noise increases.

5. Conclusion

This article describes a programmable probability generator for CBA neural network-based stochastic learning, exploiting the phase noise of CMOS ring oscillator. The proposed technique originates from a bang-bang dithering of BBPLL. The phase noise is utilized to scramble the deterministic dithering pattern for a true-random probability sequence. Because the proposed technique utilizes the phase noise, there is no need to suppress the phase noise, which facilitates an extremely low-power implementation. The programmable probability is enabled by introducing a programmable mismatch at the BBPD output. The randomizing effect of the phase noise is verified through various ring oscillator designs from distribution, spectral, and jitter standpoints. The current work provides a key knob toward the physical implementation of the stochastic learning in CBA, of which only the algorithms have been suggested most likely due to the lack of a robust probability generation method.

Conflict of Interest

The authors declare no conflict of interest.

Keywords

crossbar neural networks, memristors, oscillator phase noise, stochastic learning, weight generation circuits

[1] M. Horowitz, in IEEE Int. Solid-State Circuits Conf. Digest of Technical Papers, IEEE, San Francisco, CA 2014, p. 10.
[2] K. Kim, in IEEE Int. Solid-State Circuits Conf. Digest of Technical Papers, IEEE, San Francisco, CA 2015, p. 8.
[3] F. Alibart, E. Zamanidoost, D. B. Strukov, Nat. Commun. 2013, 4, 2072.
[4] Z. Wang, S. Joshi, S. Saveliev, W. Song, R. Midy, Y. Li, M. Rao, P. Yan, S. Asapu, Y. Zhuo, H. Jiang, P. Lin, C. Li, J. H. Yoon, N. K. Upadhyay, J. Zhang, M. Hu, J. P. Strachan, M. Barnell, Q. Wu, H. Wu, R. S. Williams, Q. Xia, J. J. Yang, Nat. Electron. 2018, 1, 137.
[5] D. S. Jeong, K. M. Kim, S. Kim, B. J. Choi, C. S. Hwang, Adv. Electron. Mater. 2016, 2, 1600090.
[6] H.-J. Yoo, in IEEE Int. Solid-State Circuits Conf. Digest of Technical Papers, IEEE, San Francisco, CA 2019, p. 20.
[7] S. Yu, P. Y. Chen, IEEE Solid State Circuits Mag. 2016, 8, 43.
[8] M. Prezioso, F. Merrikh-Bayat, B. D. Hoskins, C. G. Adam, K. K. Likharev, D. B. Strukov, Nature 2015, 521, 61.
[9] P. Yao, H. Wu, B. Gao, S. B. Eryilmaz, X. Huang, W. Zhang, Q. Zhang, N. Deng, L. Shi, H.-S. P. Wong, H. Qian, Nat. Commun. 2017, 8, 15199.
[10] F. Su, W. H. Chen, L. Xia, C. P. Lo, T. Tang, Z. Wang, K. H. Hsu, M. Cheng, J. Y. Li, Y. Xie, Y. Wang, M. F. Chang, H. Yang, Y. Liu, in Proc. Symp. on VLSI Circuits, IEEE, Kyoto, Japan 2017 p. 260.
[11] C.-X. Xue, W.-H. Chen, J.-S. Liu, J.-F. Li, W.-Y. Lin, W.-E. Lin, J.-H. Wang, W.-C. Wei, T.-W. Chang, T.-C. Chang, T.-Y. Huang, H.-Y. Kao, S.-Y. Wei, Y.-C. Chi, C.-Y. Lee, C.-C. Lo, Y.-C. King, C.-J. Lin, R.-S. Liu, C.-C. Hsieh, K.-T. Tang, M.-F. Chang, in IEEE Int. Solid-State Circuits Conf. Digest of Technical Papers, IEEE, San Francisco, CA 2019, p. 388.
[12] T. Y. Liu, T. H. Yan, R. Scheuerlein, Y. Chen, J. K. Lee, G. Balakrishnan, G. Yee, H. Zhang, A. Yap, J. Ouyang, T. Sasaki, A. Al-Shamma, C. Chen, M. Gupta, G. Hilton, A. Kathuria, V. Lai, M. Matsumoto, A. Nigam, A. Pai, J. Pakhale, C. H. Siau, X. Wu, Y. Yin, N. Nagel, Y. Tanaka, M. Higashitani, T. Minvielle, C. Gorla, T. Tsukamoto, et al., IEEE J. Solid State Circuits 2014, 49, 140.
[13] G. Kim, C. S. Hwang, D. S. Jeong, in Proc. IEEE Int. Symp. Circuits and Systems., IEEE, Sapporo, Japan 2019, p. 1.
[14] M. A. Nielsen, Neural Networks and Deep Learning, Determination Press, San Francisco, CA 2015.
[15] O. Krestinskaya, A. P. James, L. O. Chua, IEEE Trans. Neural Netw. Learn. Syst. 2020, 31, 4.
[16] R. Dauxou, M. Alscheidat, E. Neffci, G. Cauwenberghs, K. N. Salama, AIP Adv. 2016, 6, 111304.
[17] K. J. Yoon, W. Bae, D.-K. Jeong, C. S. Hwang, Adv. Electron. Mater. 2016, 2, 1600326.
[18] A. Hajimiri, T. H. Lee, IEEE J. Solid State Circuits 1999, 33, 179.
[19] W. Bae, H. Ju, K. Park, S.-Y. Cho, D.-K. Jeong, IEEE J. Solid State Circuits 2016, 51, 2357.
[20] E. Chang, J. Han, W. Bae, Z. Wang, N. Narevsky, B. Nikolic, E. Alon, in Proc. of the IEEE Custom Integrated Circuits Conf., IEEE, San Diego, CA 2018, p. 1.
[21] F. Pareschi, G. Setti, R. Rovatti, IEEE Trans. Circuits Syst. I Reg. Papers 2010, 57, 3124.
[22] D. Liu, Z. Liu, X. Zou, IEEE Trans. Circuits Syst. II Express Briefs 2016, 63, 608.
[23] W. Bae, H. Ju, K. Park, J. Han, D.-K. Jeong, IEEE Trans. Ind. Electron. 2018, 65, 5979.