SN Computer Science (2022) 3:148
https://doi.org/10.1007/s42979-022-01039-7

ORIGINAL RESEARCH

Generalized Exponentiation Using STT Magnetic Tunnel Junctions: Circuit Design, Performance, and Application to Neural Network Gradient Decay

Adrian Tatulian[1] · **Ronald F. DeMara**[1]

Received: 7 October 2021 / Accepted: 17 January 2022 / Published online: 31 January 2022
© The Author(s), under exclusive licence to Springer Nature Singapore Pte Ltd 2022

Abstract
While nonlinear functions such as square and square root are critical for fields such as signal processing and machine learning, computation of these functions presents challenges in the digital domain, including power, area, and delay overheads. While selective computation in the analog domain is a viable alternative, tradeoffs of increased noise and reduced accuracy are prominent challenges. Herein, we propose a reconfigurable analog circuit which is capable of performing generalized exponentiation within a mixed-signal field programmable array. The resulting analog block of magnetic tunnel junctions along with FET-based sensing and amplification circuits are circuit-switched-configurable with terminal-level control. Herein the design is configured to rapidly evaluate various arithmetic operations within acceptable error tolerances for selected applications. When compared to a state-of-the-art approximate digital multiplier, our design yields an approximately 95% reduction in area and stable output within a period comparable to single-cycle execution. In addition, the analog circuit allows for efficient and versatile computation of activation functions in a neural network architecture; simulation results demonstrate the possibility of reducing network size while retaining accuracy through such an approach.

Keywords Analog computation · Magnetic tunnel junction · Artificial neural network · Activation function

Introduction

Multiplication and exponentiation operations are critical for a variety of applications, including computer vision [1], signal processing [2, 3], and machine learning [4, 5]. Square and square root, for example, are commonly used for normalizing vectors in signal processing applications, and square root may serve as an activation function for neural networks [4]. Despite their ubiquity, a traditional digital implementation of such functions can incur significant area and delay overheads in the digital domain, requiring 12 or more clock cycles to execute [6] and hundreds of logic gates [7]. As a result, there has recently been renewed interest in pursuing an analog approach to operations such as multiplication, square, and square root [8, 9].

Analog circuits trade off computational accuracy for reduction in overheads such as power and area; this is an attractive tradeoff for error-tolerant applications, where power and area are constrained, e.g., Internet of Things (IoT) devices. The benefits offered by analog computation are amplified when used with vector-valued data, since the output data can be transferred to a memristive crossbar array for further processing without the need for digital-to-analog conversion [3]. One example of an ideal use case is Compressive Sensing (CS). CS entails compression and transmission of a spectrally sparse signal, and then reconstruction of the signal at the receiving end. Machine learning via neural networks is another example.

In recent years, Field Programmable Analog Arrays (FPAAs) have been demonstrated as a viable approach to signal processing applications [10]. FPAAs serve as analog equivalents of Field Programmable Gate Arrays (FPGAs),

This article is part of the topical collection "Hardware for AI, Machine Learning and Emerging Electronic Systems" guest edited by Himanshu Thapliyal, Saraju Mohanty and VS Kanchana Bhaaskaran.

✉ Adrian Tatulian
adrian.tatulian@ucf.edu

Ronald F. DeMara
ronald.demara@ucf.edu

[1] Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL, USA

with analog circuit components being used to replace lookup tables found in FPGAs. Analog devices, such as field-effect transistors (FETs), capacitors, resistors, and diodes, are integrated into a reconfigurable fabric architecture (Fig. 1). Recent innovations including the Reconfigurable Analog Signal Processor (RASP) and associated high-level tools have provided a pathway for system-level analog design [11].

Vast improvements in energy efficiency have been demonstrated by taking an analog computation approach [12], and FPAA technology has been demonstrated for power-constrained IoT sensing applications, such as temperature sensors and heart rate alarms [13].

Herein, we extend our previous work [14] on an analog fabric for execution of generalized exponentiation. Area overheads are minimized by leveraging intrinsic computation and reconfigurability, as well as 3D integration capabilities attained by the embedded Magnetic Tunnel Junction (MTJ) devices.

The remainder of the manuscript is organized as follows: the following section gives background and related works relating to analog computation circuits and spin-based devices. Next, we introduce the proposed design, and proceed to provide simulation results on circuit performance, followed by application-level results relating to Compressive Sensing and Machine Learning. Conclusions are given in the final section.

Background and Related Works

Analog Computational Circuits

Analog computation leverages intrinsic properties of electronic devices for efficient execution of mathematical functions. Thus, analog circuits may be superior to digital equivalents in latency, power consumption, and area, at the cost of computational error.

A wide variety of approaches have been proposed for analog computation within AI applications (Table 1). Abuelma'Atti and Abuelmaatti [15] achieves generalized computation through Taylor series approximation. By successively applying a squaring unit based on a class AB current mirror architecture, this circuit yields a maximum error of 10% for a 5th order polynomial. In [16], a time-mode circuit is proposed for achieving root and power computations

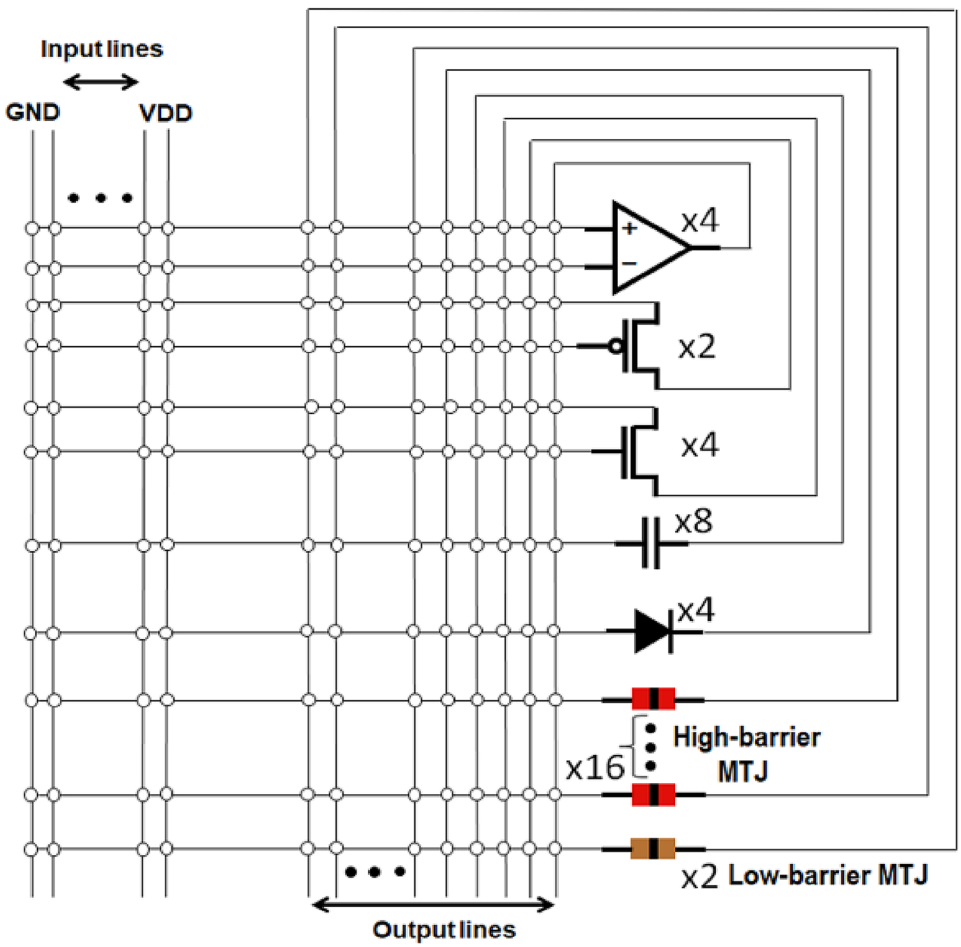

Fig. 1 FPAA fabric comprised of active and passive analog devices such as NMOS/PMOS transistors, capacitors and diodes, along with spin-based magnetic tunnel junction (MTJ) devices

Table 1 Comparison of analog computation architectures

Work	Functionality	Mode of operation	No. of components	Highlighted contributions
[15]	*n*th power via Squaring Unit	Class AB current mirror	22	Arbitrary nonlinear functions in terms of Taylor series expansion
[16]	Square, cube, 4th power	Translinear time-to-voltage and voltage-to-time convertors	~100	Nonlinear operations through the time-mode translinear principle
[17]	Cube root	Evolved computational circuit	48	Pioneer in evolutionary circuit design
[18]	Square, square root, cube, cube root	Evolved computational circuit	≤44	Genetic algorithms for optimizing analog circuits for non-conventional applications
Herein	Generalized *n*th power and root; inverse functions	Op-amps in reconfigurable fabric	43	Reconfigurable design, and intrinsic stochasticity

based on the translinear principle, i.e., cascading hardware with exponential and logarithmic outputs to yield the desired result.

Several authors [17–19] have sought automated hardware synthesis and optimization through the use of genetic algorithms. Specifically, Thangavel et al. [19] explores synthesis of arbitrary functions through Puiseux series, with genetic algorithms used to minimize error. A generalized computation approach as described in [15] and [19] enables efficient computation of complex mathematical functions, thus bringing these functions into the realm of feasibility for IoT applications. This can result in significant improvements in machine learning performance by reducing constraints on the choice of activation function [4].

Spin-Based Devices

Magnetic tunnel junctions (MTJs) are a class of spin-based emerging logic device which have been recently researched for applications such as non-volatile memory due to their numerous advantages, including near-zero standby power dissipation [3], high endurance [20] and vertical integration capabilities resulting in high density [21]. MTJs are composed of two ferromagnetic layers: a fixed layer and free layer, separated by a thin oxide barrier. The two stable states of the MTJ are determined by the relative orientation of the free-layer magnetization with respect to the fixed layer: the Parallel (P) state and Anti-parallel (AP) state, with the latter state corresponding to a significantly higher device resistance. A bi-directional spin-polarized current passing through the device may flip the state of the device.

While various materials may be chosen for MTJ fabrication, one common choice is the use of Fe for ferromagnetic layers, and MgO for the oxide barrier. This structure may be achieved using existing fabrication methods, e.g., the use of molecular beam epitaxy for preparation of the Fe layer, followed by growth of the MgO layer using e-beam evaporation, and patterning using photolithography [22]. The fabricated device is then placed on chip at the fourth metal line, in a CMOS backend process [23]. MTJs are commonly vertically integrated with CMOS technology using through-silicon vias in a 3D architecture, thus maximizing area efficiency and simultaneously minimizing data transfer overheads [24, 25]. As the building block of Magnetoresistive Random Access Memory (MRAM) technology, MTJs have been proposed as a nonvolatile alternative to SRAM in cache memory [26]. Further applications benefiting from a hybrid CMOS/MRAM approach include full adders [23] and analog-to-digital converters [27].

The device resistance is given by $R_P = R_{MTJ}$ and $R_{AP} = R_{MTJ}\,(1 + \mathrm{TMR})$, whereby [28]

$$R_{MTJ} = \frac{t_{ox}}{\mathrm{Factor} \times \mathrm{Area}\sqrt{\varphi}} \exp\left(1.025 t_{ox}\sqrt{\varphi}\right), \tag{1}$$

$$\mathrm{TMR} = \frac{\mathrm{TMR}_0}{1 + \left(\frac{V_b}{V_h}\right)^2}, \tag{2}$$

in which TMR is tunneling magnetoresistance, t_{ox} the oxide layer thickness, Factor a material-dependent parameter which depends on the resistance-area product of the device, Area the surface area of the device, φ the oxide layer energy barrier height, V_b bias voltage, and V_h the bias voltage at which TMR drops to half of its initial value. MTJs have been fabricated at varying resistance levels ranging from the kilo-Ohm [27] to mega-Ohm [29] range.

In addition to device resistance, the energy barrier, E_B, between the P and AP states of an MTJ device can be tuned based on fabrication dimensions. The device is considered to be low-barrier under the condition $E_B \ll 40$ kT, in which case thermal fluctuations at room temperature are sufficient to change the state of the device. This observation has led to construction of the probabilistic bit (p-bit) device, as shown in Fig. 2. A p-bit [30, 31] takes analog input and yields a digital output whose probability of being logic 1 depends on

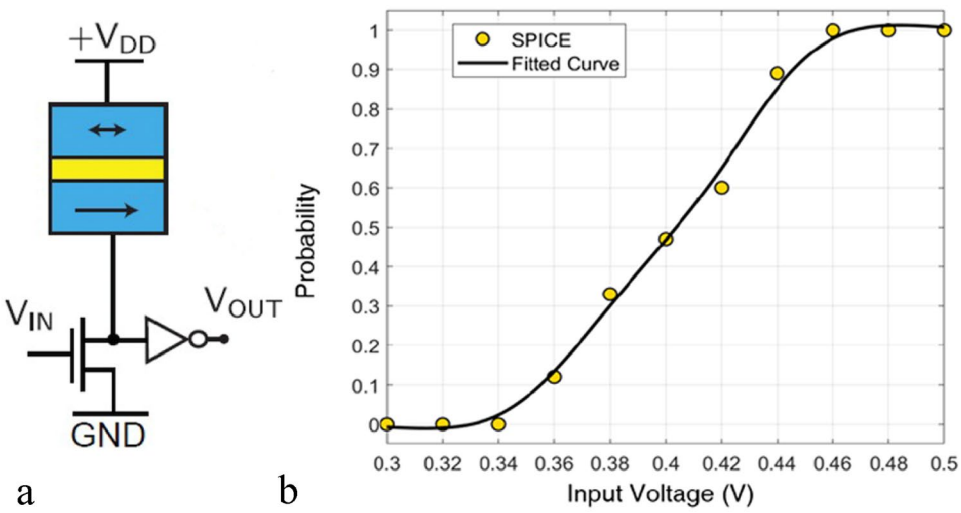

Fig. 2 Structure of a p-bit device consisting of a voltage divider between a low-barrier MTJ device and NMOS transistor (**a**); probability of a logic-1 output value (**b**)

the supplied input voltage. This functionality is due to the p-bit's structure as a voltage divider between a low-barrier MTJ and NMOS transistor. A higher voltage applied to the gate of the transistor results in reduced drain–source voltage, r_{ds}, which increases the probability of delivering sufficient voltage to the input of the inverter to yield a logic 1 output.

The p-bit output is described by the equation [30]:

$$V_{out} = V_{DD}\text{sgn}\{\tanh\left(V_b/V_0\right) + \text{rand}(-1,\ 1)\}, \tag{3}$$

where sgn represents the sign function, rand$(-1,\ 1)$ represents a random number in $[-1,\ 1]$, V_b is bias voltage and V_0 is a model parameter. Thus, the probability of obtaining a logic-1 output from the p-bit device is given by Eq. (4):

$$P(1) = \frac{1}{2}\left(1 + \tanh\left(\frac{V_b}{V_0}\right)\right). \tag{4}$$

It is common to average the p-bit output to realize the hyperbolic tangent function through Eq. (4).

Field Programmable Mixed-Signal Arrays

Due to the significant benefits offered by analog computation to certain use cases, there has been a renewed interest in extending the scope of reconfigurable computing to the analog domain. The Reconfigurable Analog Signal Processor (RASP) [11] introduced in 2012 was an attempt to overcome two of the main challenges preventing widespread adaptation of analog processing: lack of a programmable interface and lack of robust design tools. The RASP provided a set of high-level design tools for system-level analog design.

Concurrently with the introduction of the RASP, researchers [32] developed a Field Programmable Mixed-Signal Array (FPMA) consisting of both analog and digital elements arranged in a Manhattan-style fabric. Their design consisted of Computational Logic Blocks (CLBs) comprised of LUTs and D Flip-flops, and Computational Analog Blocks (CABs) consisting of analog elements such as op-amps, capacitors, and transistors. Their design used a global interconnect to route signals between computational blocks; in addition, each block contained a local interconnect which operated via a set of reconfigurable switches.

Subsequent innovations included the addition of a 16-bit microprocessor to the mixed-signal array for added computational capability [33] and the use of Time-domain Configurable Analog Blocks [34] for dynamic reconfigurability of the analog function being implemented.

Analog Circuit Design

Op-amp Design

The proposed reconfigurable analog multiplier is based on the op-amp design presented in Fig. 3. The op-amp

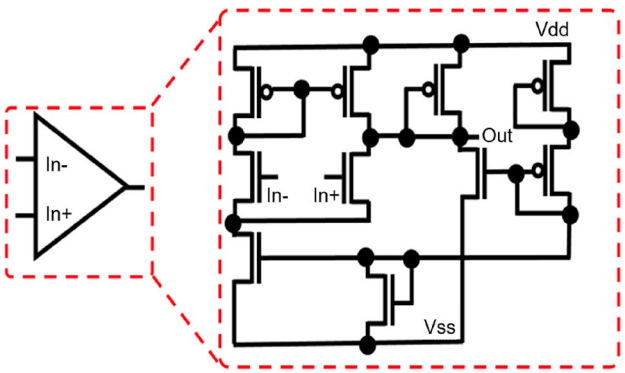

Fig. 3 Op-amp comprised of 10 MOSFETs offering high speed and compact area

consists of two cascaded stages; an input stage consisting of a differential amplifier, followed by a gain stage. A simple design consisting of only 10 CMOS transistors is chosen to optimize for power consumption as well as area. The op-amp is simulated using models from the PTM 14 nm LSTP library, at $V_{DD} = 0.8$ V.

Figure 4a then presents a layout of the proposed op-amp design. The layout indicates dimensions of $43F \times 23F$, for a total area of $989F^2$. This layout is contrasted with a CMOS NAND gate in Fig. 4b, which has dimensions of $18F \times 14.5F$, for a total area of $261F^2$. The op-amp and NAND gate form an interesting comparison as common building blocks of analog and digital multipliers, respectively.

Three-Stage Analog Multiplier

Similarly to [16], the translinear principle is applied to attain exponentiation of the input signal. As shown in Fig. 5, we introduce a three-stage design whose output is a power function of the input. The design as shown in Fig. 5 accepts a single input for performing exponentiation operations; the design can also be reconfigured to accept two inputs for performing analog multiplication.

The first stage, outlined in red in Fig. 5, is a logarithmic amplifier with output given by

$$V_1 = -A_{OL} V_0, \tag{5}$$

$$-\frac{V_0 - V_{in}}{R_1} = I_{S1}\left[\exp\left(\frac{V_0 - V_1}{V_T}\right) - 1\right], \tag{6}$$

where A_{OL} represents open-loop gain and I_{S1} represents the saturation current of diode D_1. Equation (5) is from general op-amp theory and Eq. (6) follows from KCL. Thus, solving Eqs. (5) and (6) simultaneously yields

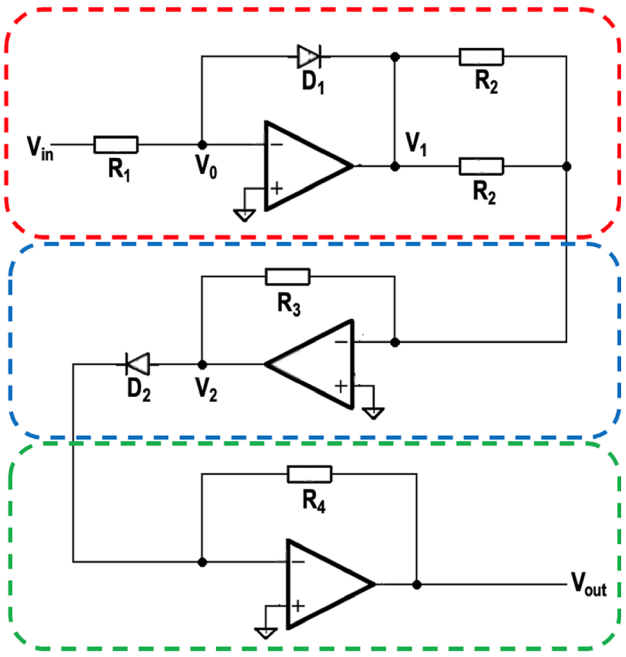

Fig. 5 Analog circuit for generalized exponentiation. The first, second, and third stage are outlined in red, blue, and green, respectively

$$V_1\left(1 + \frac{1}{A_{OL}}\right) = -V_T \ln\left(\frac{V_{in} + \frac{V_1}{A_{OL}}}{R_1 I_{S1}} + 1\right). \tag{7}$$

In the limit of infinite open-loop gain and sufficiently high input voltage, Eq. (7) is approximated as

$$V_1 = -V_T \ln\left(\frac{V_{in}}{R_1 I_{S1}}\right). \tag{8}$$

The second stage is an analog adder, whereby a similar analysis yields $V_2 = -\frac{2V_1 R_3}{R_2}$. Finally, the third stage is an anti-log amplifier with output approximately given by

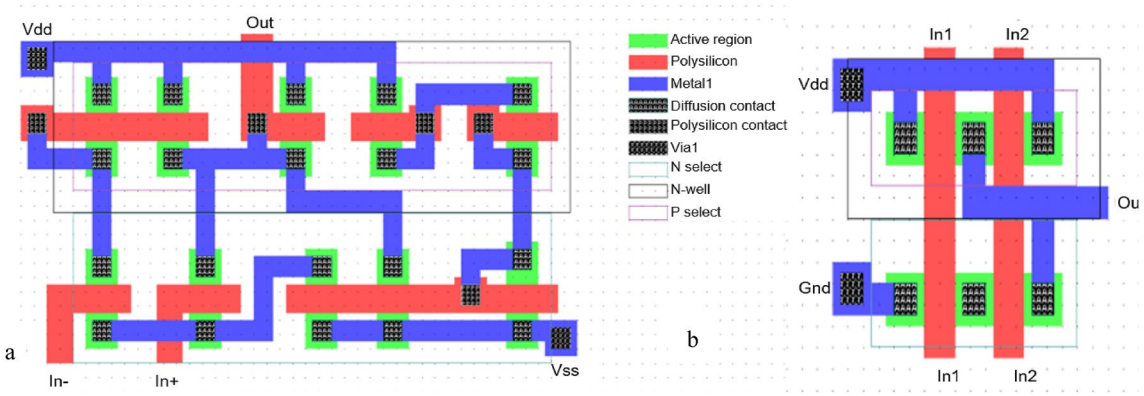

Fig. 4 **a** Layout of op-amp used in this paper vs. **b** a CMOS NAND gate

$$V_{\text{out}} = -R_4 I_{S2} e^{\frac{V_2}{V_T}}, \tag{9}$$

where I_{S1} represents the saturation current of diode D_2. Overall, it is simple to see that the output of this circuit is given by

$$V_{\text{out}} = -\frac{R_4 I_{S2}}{(R_1 I_{S1})^a} (V_{\text{in}})^a, \tag{10}$$

where $a = 2R_3/R_2$.

The above theory demonstrates the ability to implement any positive power function of the input via the design shown in Fig. 5. In addition, a dual-input stage consisting of two logarithmic amplifiers can be inserted to attain an analog multiplier. Finally, an inverting amplifier can be inserted between the second and third stages to realize inverse power functions as well. Each mode can be implemented using the elements included in the fabric presented in Fig. 1. To minimize area, MTJs in the P state are used to implement the resistors shown in Fig. 5; MTJs in the P state have roughly linear I–V characteristics as implied by Eq. (1) and validated through experimental data [35].

Equations (8–10) hold only for infinite open-loop gain which is not attained in practice. Thus, the equations provide a starting point for the design, after which parameters must be adjusted to minimize output errors. Final parameters are: $R_1 = 3500$ kΩ, $R_2 = 50$ kΩ, $R_3 = 150$ kΩ, $R_4 = 75$ kΩ, $I_{S1} = 50$ nA, and $I_{S2} = 5.4$ nA. In addition, a load capacitance of 100 fF and load resistance of 1000 kΩ is included at the output stage of each op-amp.

Analog Circuit Performance

Analog Multiplier

First, the performance of the analog circuit is evaluated as a multiplier. In this mode, two separate logarithmic amplifiers serve as the input stage, receiving inputs V_{in1} and V_{in2}. The circuit is evaluated in terms of DC transfer characteristics, frequency response, and Total Harmonic Distortion (THD), for various DC amplitudes of V_{in2} within the operational range between 0.3 and 0.7 V.

DC transfer characteristics are presented in Fig. 6. In each trial, V_{in1} is swept across the operational range and the average non-linearity error is determined based on the percentage deviation from a linear regression line. As listed in Table 2, the maximum non-linearity error of 0.55% occurs at $V_{\text{in2}} = 0.7$ V.

Figure 7 shows the frequency response of the multiplier, evaluated in the range from 100 MHz to 1 GHz. In this case, V_{in1} is a sinusoidal signal with offset of 0.45 V and amplitude

Table 2 Error, bandwidth, and delay data

V_{in2} (V)	Error (%)	− 3-dB bandwidth (MHz)	Delay (ns)
0.3	0.48	195	3.8
0.4	0.11	191	3.9
0.5	0.25	186	4.1
0.6	0.43	178	4.4
0.7	0.55	174	5.0

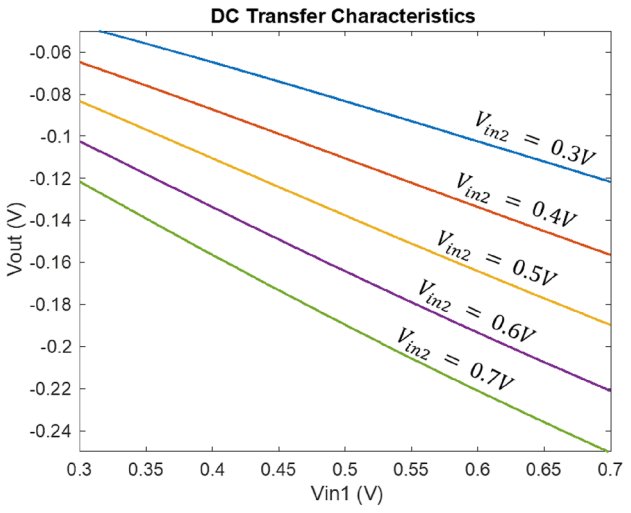

Fig. 6 DC transfer characteristics for the proposed multiplier, with one input fixed and the second input varying across the operational range

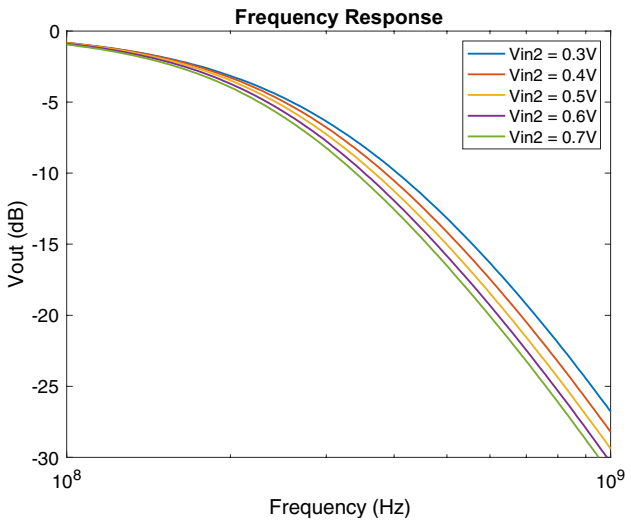

Fig. 7 Frequency response, with one input fixed and the second input sinusoidal with offset of 0.45 V and amplitude of 0.25 V

of 0.25 V. The − 3-dB bandwidth, listed in Table 2, is in the 100 MHz range in each case. While this bandwidth may be high for applications with limited signal-to-noise ratio, the circuit can be reconfigured to attain various bandwidths depending on the RC time constant at the op-amp output.

Table 2 also lists the delay in reaching 90% of the target voltage in the case when $V_{in1} = V_{in2}$; the delays are in the nanosecond range, consistent with the circuit bandwidth.

Next, Table 3 provides THD in the case, where one input is 0.45 V DC, and the second input is sinusoidal with amplitudes of 0.05 V and 0.25 V. THD is within 1% up to a frequency of approximately 1 MHz, indicating practical functionality of the system.

*n*th Power and Root

The circuit is next evaluated in its ability to compute square and square root functions. Simulation results demonstrate high-accuracy implementation of *n*th-root functions; power functions beyond squaring introduce challenges related to voltage saturation. It is, however, possible to obtain these functions via a squaring unit by iteratively applying the mathematical identity:$(A + B)^2 - (A - B)^2 = 4AB$. For example, in the case of the cubing function, x^2 is substituted for A and x for B; any *n*th power function, $n \geq 2$, can thus be computed [15]. The authors of [15] were able to compute a 5th order polynomial function within 10% error through this approach.

Table 3 THD with one DC and one sinusoidal input

Frequency	Amplitude = 0.25 V (%)	Amplitude = 0.05 V (%)
10 kHz	0.80	0.76
100 kHz	0.81	0.77
1 MHz	0.81	0.75
2 MHz	1.08	1.10
3 MHz	1.82	1.61

Data on cube root, square root, and squaring circuits implemented using the proposed design are given in Table 4, including technology node, supply voltage, total number of elementary components, power dissipation, and mean error over an input range of 0.2–0.6 V.

Comparing to the approximate digital multiplier described in [7], at the design point giving nearly identical power consumption, the analog circuit described herein yields slightly improved mean error across the operational range. Furthermore, the approximate digital design requires an area equivalent to 245 CMOS NAND gates, i.e., 980 CMOS transistors. Thus, our design achieves a 97% reduction in transistor count. In addition, the layout presented in Fig. 4 indicates that the layout of an op-amp is approximately 3.79 × the area of a NAND gate; this indicates an approximately 95% reduction in area if three op-amps are used for squaring.

The design presented in [9] demonstrates reduced power consumption but significantly higher error, and a relatively limited bandwidth of 51.2 kHz. [16] introduces a similar design to the one described herein, relying on the translinear principle to implement *n*th power functions by combining hardware with logarithmic and exponential output characteristics; a limitation of this design is that its reliance on time-mode circuitry intrinsically leads to significant time delays, on the order of microseconds.

In addition, a Monte Carlo simulation is performed to determine the effects of process variation in MTJ devices. For this simulation, 100 trials are conducted considering a 1.5% standard deviation in the resistance of each MTJ device; this value is consistent with the variation seen in a 4-Mb MRAM array [36]. The resulting standard deviations in the circuit outputs are listed in Table 5. While the maximum standard deviation due to PV is 6.36%, the presence of only 16 high-barrier MTJ devices in the reconfigurable fabric may allow for improved device tolerances and thus improved computational accuracy in the fabricated design.

Given the temperature dependence of diodes as well as MTJ devices, a brief temperature analysis of the circuit performance is conducted to complement the previous

Table 4 Comparison of results

	Herein	Herein	Herein	[7]	[9]	[15]	[16]
Mode	Analog	Analog	Analog	Digital	Analog	Analog	Analog
Operation	Cube root	Square root	Square	Multiplier	Multiplier	Square	Square
Tech node (nm)	14	14	14	28	130	500	180
V_{DD} (V)	0.8	0.8	0.8	1	0.6	1.5	1.3
No. of components	43	43	43	~ 1000	35	12	~ 100
Power (μW)	123	122	126	126	23	600	149
Mean error (%)	0.50	0.66	1.30	1.87	9.1[a]	N/A	0.24[b]

[a]RMS noise vs. max. output

[b]At $V_{in} = 0.4$ V

Table 5 Error due to process variation of MTJ devices

V_{in} (V)	Square (%)	Square root (%)
0.3	5.96	3.81
0.4	6.36	3.88
0.5	6.13	3.92
0.6	5.72	3.95
0.7	5.30	3.96

data attained at a temperature of 25 °C. In this analysis, the temperature dependency of the mean error of Fig. 5 is determined when configured to function as a squaring unit. While HSPICE includes temperature dependent models for diodes, a custom model is used for MTJs. Specifically, the MTJ conductance may be modeled as [37, 38]

$$G(T,\ \theta) = G_{\mathrm{T}}(T)\left\{1 + P(T)^2 \cos\theta\right\} + G_{\mathrm{SI}}(T) \tag{11}$$

$$G_{\mathrm{T}}(T) = \frac{G_0 CT}{\sin(CT)} \tag{12}$$

$$P(T) = P_0\left(1 - \alpha T^{\frac{3}{2}}\right) \tag{13}$$

$$P_0 = \sqrt{\frac{\mathrm{TMR}_0}{2 - \mathrm{TMR}_0}} \tag{14}$$

$$G_{\mathrm{SI}}(T) = ST^{4/3}. \tag{15}$$

In these equations, G represents MTJ device conductance, T represents absolute temperature, θ represents relative magnetization orientation, P represents polarization, and TMR represents tunneling magnetoresistance. G_0, TMR_0, α, C and S are model parameters. The proposed design uses P-state MTJs, so $\theta = 0$ is chosen for this analysis. Furthermore, the model parameters $\mathrm{TMR}_0 = 1$, $\alpha = 2 \times 10^{-5}\ \mathrm{K}^{-3/2}$, $C = 0.015\ \mathrm{K}^{-1}$ and $S = 10^{-12}\ \Omega^{-1}\ \mathrm{K}^{-4/3}$ are used; G_0 is chosen based on the target MTJ conductance value. TMR_0 is consistent with [27]; moreover, α and S are consistent in order of magnitude with [37]. C is chosen based on the equation [37]:

$$C = 1.387 \times 10^{-4} t/\sqrt{\varphi}, \tag{16}$$

where t represents oxide barrier height in Angstroms, and φ. is the oxide barrier potential in electron-volts. The figure $C = 0.0015$ is derived assuming an oxide barrier thickness of 10 Å [27], and a barrier potential on the order of 1 eV [37].

Based on the above model, the mean computational error of the squaring unit across an input voltage range between 0.2 and 0.6 V is determined, for temperatures ranging from −20 to 70 °C, with results summarized in Table 6. An error below 10% is observed between temperatures of −10 °C and 40 °C. This operational interval includes common environmental temperatures as well as physiological temperature but does not cover the commercial temperature range. Thus, further work is necessary to improve the temperature resilience of the design for use in more extreme environments.

Table 6 Mean error of analog squaring circuit as a function of temperature, T

T (°C)	Mean error (%)
−20	14.1
−10	3.17
0	0.748
10	0.352
20	0.551
30	1.70
40	5.71
50	14.2
60	22.9
70	27.3

Generalized Functions

The proposed hardware may also be used to implement generalized functions. For one, the analog circuit can function in a third mode, where inverse power and root functions are computed by adding an inverting amplifier before the final stage; a $1/\sqrt{x}$ function yields an average error of 0.4%. Furthermore, exponential and logarithmic functions can be computed using only one op-amp stage. Other generalized functions can be implemented using a Taylor series approximation.

Algorithm 1 Approximate Message Passing

Inputs: The measurement matrix, $\mathbf{\Phi}$
The measurement vector, $\mathbf{y}$
The number of measurements, m
Output: The approximate signal vector, $\hat{\mathbf{x}}$

Procedure:
1) Initialize the residual $\mathbf{r_0} = \mathbf{y}$, the signal approximation $\hat{\mathbf{x}}_0 = \mathbf{0}$, and the counter $i = 1$.

while $i < k$ **do**
2) $\theta = \|\mathbf{r}_{i-1}\|/\sqrt{m}$
3) $\mathbf{a} = \hat{\mathbf{x}}_{\mathbf{i-1}} + \mathbf{\Phi}^{\mathbf{T}}\mathrm{r}_{i-1}$
4) $\hat{\mathbf{x}}_{\mathbf{i}} = \mathrm{sign(a)max(|\ a\ | - \theta, 0)}$
5) $b_i = \|\mathbf{x}_i\|_0/m$
6) $\mathbf{r}_i = \mathbf{y}$ - $\mathbf{\Phi}\mathrm{x}_i + b_i\mathbf{r}_{i-1}$
end while

Figure 8 shows an approximation of the function $f(x) = x - x^2 - x^3 - x^4 - x^5$ based on the proposed analog squaring unit. This simulation includes squaring errors, but does not include errors in addition, subtraction, and voltage rescaling; the resulting output is a fair approximation of the target function, with an average error of 4.83% over the

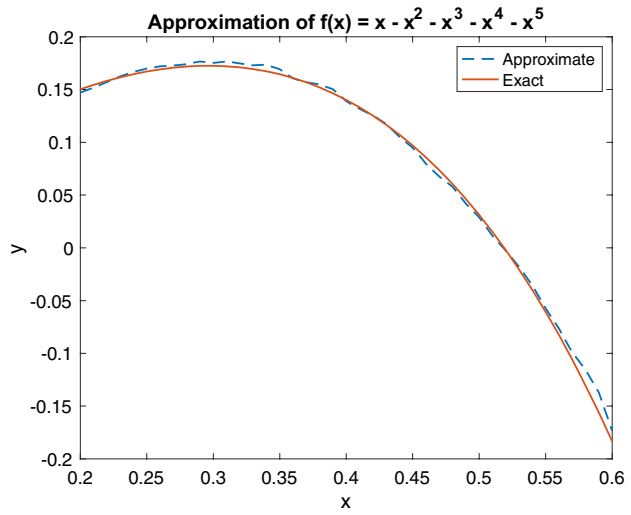

Fig. 8 Approximation of a 5th order polynomial function using the proposed hardware, showing agreement with an error-free implementation

tested range. This demonstrates the feasibility of generating generalized functions through Taylor series using our analog approach.

Application to Compressive Sensing

Compressive Sensing (CS) is an emerging signal processing technique which allows for sub-Nyquist sampling of spectrally sparse and wideband data. Applications of CS include reduction of power consumption and complexity in 5G communication networks [39], and reduction of sampling duration in time-critical applications such as MRI [40]. CS applies a linear transformation to signal $\boldsymbol{x} \in \mathbb{R}^n$ via the measurement matrix, $\boldsymbol{\Phi} \in \mathbb{R}^{m \times m}$, to obtain a compressed measurement vector, $\boldsymbol{y} \in \mathbb{R}^m$, with $m < n$. The receiver must then solve an undetermined system of linear equations to reconstruct the original signal.

One possible approach to the CS reconstruction problem is to choose the solution with the lowest sparsity, such that the sparsity, k, is defined as the number of nonzero elements in the signal. This translates directly to the minimization problem: $\hat{\mathbf{x}} = \text{argmin}\|\boldsymbol{x}\|_0$ s.t. $\boldsymbol{y} = \boldsymbol{\Phi x}$. Due to this problem being NP-hard [41], an alternative approach is to solve the *basis pursuit* problem:

$$\hat{\mathbf{x}} = \text{argmin}\|x\|_1 \text{ s.t. } \boldsymbol{y} = \boldsymbol{\Phi x}. \tag{17}$$

The condition for $\hat{\mathbf{x}}$ being an accurate reconstruction of the original signal vector is the *Restricted Isometry Property* (*RIP*) [42], i.e., that for any k-sparse vector $\boldsymbol{x}$:

$$\|\boldsymbol{x}\|_2^2(1-\delta) \leq \|\boldsymbol{\Phi x}\|_2^2 \leq \|\boldsymbol{x}\|_2^2(1+\delta). \tag{18}$$

Besides basis pursuit, a variety of algorithms with different tradeoffs allow for CS reconstruction [43]. One example, Approximate Message Passing (AMP), is a soft thresholding algorithm designed for fast convergence [44]. The design is presented as Algorithm 1. In this notation, sign($\mathbf{a}$)max($|\mathbf{a}| - \theta$, 0) refers to elementwise vector operations, where the constant θ is applied to each element. The function sign(x) is defined to be $+1$ for $x > 0$ and -1 for $x < 0$.

The AMP algorithm begins by initializing the residual vector, $\mathbf{r_0}$, to the measurement vector $\mathbf{y}$, as well as initializing the estimate of the signal vector $\mathbf{x}$ to zero (Line 1). Next, the threshold θ is computed as the root mean square error of the residual (Line 2). Lines 3–4 provide an estimation of the reconstructed signal vector as a function of the thresholding parameter, in accordance with the Iterative Soft Thresholding technique. Finally, Lines 5–6 demonstrate the key difference between AMP and the Iterative Soft Thresholding approach [45], i.e., the residual is updated based on not only the current signal estimate $\hat{\boldsymbol{x}}_i$ but also based on the residual of the previous iteration, $\boldsymbol{r}_{i-1}$.

AMP reconstruction, as an error tolerant use case requiring square and square root computations in each iteration, serves as a viable application for the hardware presented herein. The performance of AMP is evaluated in MATLAB based on signals of length $n = 1000$, with sparsity rate $k/n = 0.1$. The number of measurements, m, is varied from 200 to 500 to determine the magnitude of the reconstruction error in decibels, defined as

$$\text{Error (dB)} = 20 \log \left(\frac{\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|}{\|\boldsymbol{x}\|} \right). \tag{19}$$

Figure 9 shows AMP performance considering an exact implementation (blue dots), approximation errors intrinsic to the analog hardware as detailed in Table 4 (red dots), and finally approximation errors considering process variation errors detailed in Table 5 (yellow dots).

The results demonstrate a negligible impact of the intrinsic circuit error on AMP performance; certain data points such as $m = 500$ demonstrate a lower error with the approximate approach, indicating statistical insignificance of the error. Even the increased computational error due to process variation amounts to only a slight degradation in performance and consistently requires less than 50 additional measurements to regain the reconstruction accuracy of the AMP algorithm.

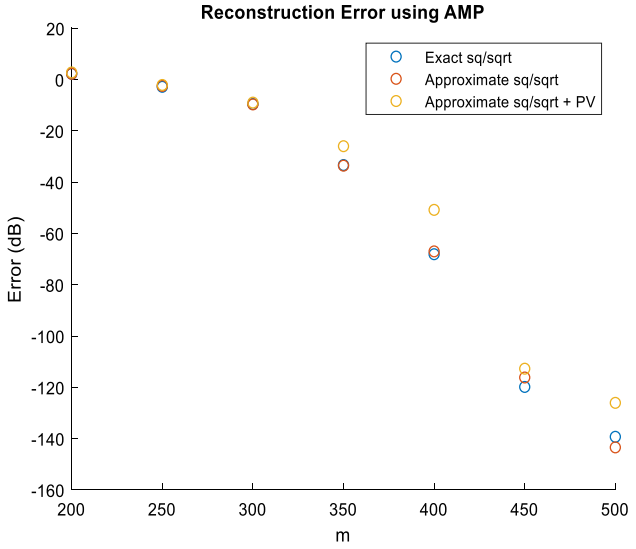

Fig. 9 Signal reconstruction error of the AMP algorithm as a function of number of measurements, where square and square root operations are performed exactly (blue circles), with approximation error of the presented hardware (red circles), and with approximation error due to process variation (yellow circles)

Application to Machine Learning

Gradient Decay Problem

Deep neural networks (DNNs) have been gaining popularity in the context of diverse applications including computer vision [46] and speech recognition [47]. At each layer, the DNN takes a vector input, **x**, and outputs a linear transformation of the input, **z,** according to the equation $\mathbf{z} = \mathbf{W}\mathbf{x}$, where **W** is the weight matrix. To facilitate learning non-linear relationships, the output **z** is transformed by an activation function to yield a final layer output, $\mathbf{h} = f(\mathbf{z})$. The choice of activation function has recently been a subject of research interest due to its significant impact on the success of a neural network [5]. While hyperbolic tangent has been commonly used, this function suffers drawbacks including the gradient decay problem, i.e., the gradient is diminished for multi-layer networks due to repeated multiplication of values having absolute value less than 1 [48].

The vanishing gradient problem has been addressed by choice of alternative activation functions, e.g., the Rectified Linear Unit (ReLU) which is defined as $f_{\mathrm{ReLU}}(x) = \max(0, x)$ and has a gradient of 1 for all $x > 0$. Another alternative is the square root function, which experiences significantly slower gradient decay compared with hyperbolic tangent. It has been observed that the derivative of the hyperbolic tangent function at $x = 10$ is less than the derivative of the square root function at $x = 10^{16}$. Previous research has demonstrated that replacing hyperbolic tangent with a square root activation function can allow for a 5% improvement in classification accuracy on the CIFAR-10 data set [4].

Given the robust capabilities of the analog circuit presented herein, we next evaluate its ability to generate improved activation functions for DNN performance. The evaluation is performed in the context of a Deep Belief Network (DBN) used to classify samples from the MNIST data set.

Deep Belief Networks (DBNs)

Restricted Boltzmann Machines (RBMs) are a class of recurrent stochastic neural network [49] in which the energy of the network in state k is determined by Eq. (20):

$$E(k) = -\sum_i s_i^k b_i - \sum_{i<j} s_i^k s_j^k w_{ij}, \tag{20}$$

where s_i^k refers to the state of node i, while the network is in state k, and w_{ij} represents the weight between nodes i and j. Each node in an RBM has a probability of being in state 1 given by Eq. (21):

$$P(s_i = 1) = \sigma\left(b_i + \sum_j w_{ij} s_j\right), \tag{21}$$

where σ represents the sigmoid function. Over time, a Boltzmann distribution is reached, where the probability of finding the system in state k is defined as

$$P(k) = \frac{e^{-E(k)}}{\sum_u e^{-E(u)}}, \tag{22}$$

where the summation in the denominator is taken over all possible states of the system. An RBM is a two-layer neural network consisting of a visible layer and hidden layer; by stacking RBMs, it is possible to realize a DBN of arbitrary length [49].

Probabilistic Inference Network Simulator (PIN-Sim)

DBN simulations on the MNIST data set can be readily performed at both the software and hardware level, using the Probabilistic Inference Network Simulator (PIN-Sim) [49]. PIN-Sim consists of five modules: first, *trainDBN* reads the training images in MATLAB and outputs the weight and bias matrices representing the DBN; a second MATLAB module, *mapWeight,* converts the weight and bias data into device conductance values. Next, the Python module, *mapRBM,* generates SPICE representations of multiple crossbar weighted arrays based on the outputs of *mapWeight* and the given network topology. A final Python module, *testDBN*, executes a SPICE circuit simulation of the DBN to determine classification error rate as well as power consumption.

The inputs to the *testDBN* module consist of the outputs of *mapWeight* and *mapRBM* as well as the module, *neuron*, which is a SPICE representation of the circuit used for computing the activation function. A visual description of PIN-Sim is shown in Fig. 10.

Impact of Activation Function

Given the robustness of the analog circuit presented herein, we investigate the impact of three separate activation functions on DBN performance: $f_1(x) = \frac{1}{2}(1 + \tanh(x))$, $f_2(x) = \sqrt{f_1(x)}$, and $f_3(x) = (1 + e^{-x})^{-1}$. Since $f_2'(x) > f_1'((x)$ for $x < -0.55$ and $f_3'(x) > f_1'(x)$ for $|x| > 1.06$, substitution of these functions may potentially alleviate the rate of gradient decay for certain inputs. Moreover, each function may be implemented using the FPAA fabric shown in Fig. 1; the presence of low-barrier MTJ devices allows for construction of p-bit devices, at which point f_1 is computed via an op-amp integrator at the output. Computation of f_2 requires an additional 3 op-amps to execute the square root function in analog; finally, f_3 requires a total of 6 op-amps to execute based on the computational units described in Fig. 5.

A DBN software simulation is performed in MATLAB for each activation function to evaluate the classification accuracy for the MNIST data set, based on 3000 training samples and 1000 test samples. Figure 11 shows the results based on various network topologies. Over the network topologies tested, both f_2 and f_3 demonstrate a consistent improvement in error rate over f_1; the average improvement is 6.4% for f_2 and 8.7% for f_3. Moreover, in certain cases, selection of f_3 vs. f_1 as an activation function allows

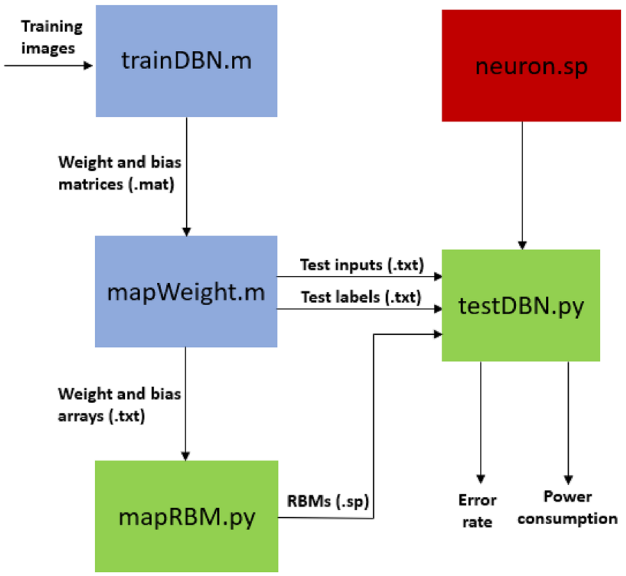

Fig. 10 Logical flow of PIN-Sim, including the five main modules involved in DBN simulation

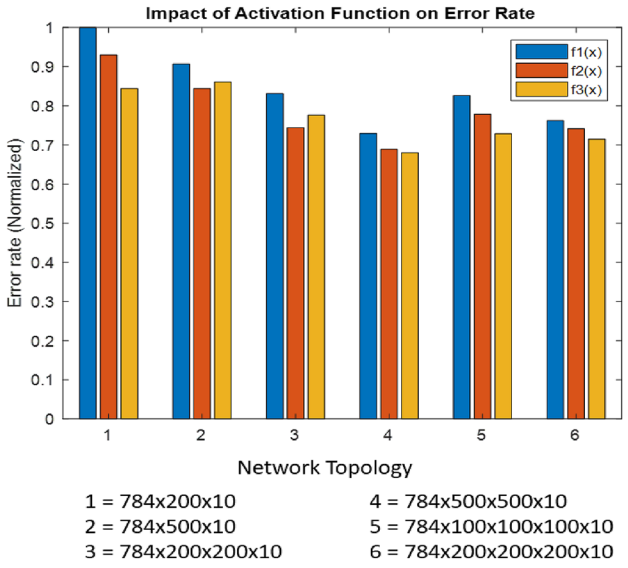

Fig. 11 Normalized error rate for image classification, based on various DBN topologies and the activation functions, $f_1(x) = \frac{1}{2}(1 + \tanh(x))$, $f_2(x) = \sqrt{f_1(x)}$, and $f_3(x) = (1 + e^{-x})^{-1}$ represented by blue, red, and yellow bars, respectively

for reduction in error rate while decreasing the size of the array, e.g., from $784 \times 500 \times 10$ to $784 \times 200 \times 10$, and from $784 \times 200 \times 200 \times 200 \times 10$ to $784 \times 100 \times 100 \times 100 \times 10$.

A PIN-Sum simulation is conducted, based on the MTJ parameters taken from [50] and listed in Table 7, for average RBM power consumption in select network topologies using the f_1 and f_2 activation functions. For simulations implementing f_2, the neuron.sp file in the PIN-Sim framework is modified by adding an analog square root unit to the output, based on the design shown in Fig. 5.

Simulation results are listed in Table 8, including average power consumption and corresponding software error rates; the power-error-product (PEP) is computed as a product of these data points and listed in the table as well. Similar to the previously used energy-error-product [50], PEP is a useful metric for attaining an overall evaluation of each design. Based on the results, the f_2 activation function yields an

Table 7 MTJ parameters used for simulating p-bit devices [50]

Parameter	Value
Saturation magnetization	1100 emu/cm^3
Free layer diameter, thickness	22 nm, 2 nm
Polarization	0.59
TMR	110%
MTJ RA-product	9 μΩ-cm^2
Damping coefficient	0.01
Temperature	300 K

Table 8 Error rate, average DBN power consumption, and power-error-product for various network topologies and activation functions

Network topology	Activation function	Error rate	Power (mW)	Power-error-product
784×200×10	f_1	0.1239	72.4	8.97
784×200×10	f_2	0.1152	76.1	8.77
784×200×200×10	f_1	0.1030	106.3	10.95
784×200×200×10	f_2	0.0922	88.5	8.16
784×200×200×200×10	f_1	0.0945	153.7	14.52
784×200×200×200×10	f_2	0.0919	119.2	10.95

improvement in PEP for each of the tested topologies; the average improvement is 17.4%.

Conclusions

Herein, we have presented an analog circuit capable of multiplication and general exponentiation operations. The circuit is based on a reconfigurable fabric which allows for versatility in the mode of operation as well as tunability in bandwidth, allowing for adaptation to diverse signal processing and machine learning applications.

Simulation results on circuit performance indicate reduction in error and 95% reduction in area when compared to a state-of-the-art approximate digital multiplier; a significant reduction in execution time in addition to reduction in complexity is attained in comparison to a time-mode analog exponentiation circuit operating on similar principles.

A key use case of the proposed design is computation of activation functions in DNN applications. Simulation results demonstrate that varying the activation function of a neural network can allow for similar improvements in error rate as increasing the network size. Thus, the intrinsic computation capabilities of our analog design can allow for synergistic area benefits, not only in the design itself, but also in a DNN crossbar architecture making use of the design.

Neural network activation functions constitute a field of active research, and many potential alternatives have been proposed. Future work may include a more complete analysis on the applicability of an analog computation approach to machine learning scenarios.

Acknowledgements This work was supported in part by the Center for Probabilistic Spin Logic for Low-Energy Boolean and Non-Boolean Computing (CAPSL), one of the Nanoelectronic Computing Research (nCORE) Centers as task 2759.006, a Semiconductor Research Corporation (SRC) program sponsored by the NSF through CCF-1739635, and by NSF through ECCS-1810256.

Declarations

Conflict of Interest The authors declare no conflict of interests.

References

1. Strickland RN, Draelos T, Mao Z. Edge detection in machine vision using a simple L1 norm template matching algorithm. Pattern Recognit. 1990;23(5):411–21. https://doi.org/10.1016/0031-3203(90)90064-R.
2. Shi Y, Xia S, Zhou Y, Shi Y. Sparse signal processing for massive device connectivity via deep learning. In: 2020 IEEE international conference on communications workshops (ICC Workshops); 2020. p. 1–6. https://doi.org/10.1109/ICCWorkshops49005.2020.9145284.
3. Tatulian A, Salehi S, DeMara RF. Mixed-signal spin/charge reconfigurable array for energy-aware compressive signal processing. In: 2019 International conference on ReConFigurable computing and FPGAs (ReConFig); 2019. p. 1–8. https://doi.org/10.1109/ReConFig48160.2019.8994799.
4. Yang X, Chen Y, Liang H. Square root based activation function in neural networks. In: 2018 International conference on audio, language and image processing (ICALIP); 2018. p. 84–9. https://doi.org/10.1109/ICALIP.2018.8455590.
5. Sipper M. Neural networks with À La Carte selection of activation functions. SN Comput Sci. 2021;2(6):1–9. https://doi.org/10.1007/s42979-021-00885-1.
6. Hasnat A, Bhattacharyya T, Dey A, Halder S, Bhattacharjee D. A fast FPGA based architecture for computation of square root and inverse square root. In: 2017 Devices for integrated circuit (DevIC); 2017. p. 383–7. https://doi.org/10.1109/DEVIC.2017.8073975.
7. Jiang H, Liu C, Lombardi F, Han J. Low-power approximate unsigned multipliers with configurable error recovery. IEEE Trans Circuits Syst I Regul Pap. 2018;66(1):189–202. https://doi.org/10.1109/TCSI.2018.2856245.
8. Arya N, Soni T, Pattanaik M, Sharma G. Area and energy efficient approximate square rooters for error resilient applications. In: 2020 33rd international conference on VLSI design and 2020 19th international conference on embedded systems (VLSID); 2020. p. 90–5. https://doi.org/10.1109/VLSID49098.2020.00033.
9. de Sousa AJS, et al. A very compact CMOS analog multiplier for application in CNN synapses. In: 2019 IEEE 10th Latin American symposium on circuits and systems (LASCAS); 2019. p. 241–4. https://doi.org/10.1109/LASCAS.2019.8667594.
10. Wunderlich RB, Adil F, Hasler P. Floating gate-based field programmable mixed-signal array. IEEE Trans Very Large Integr (VLSI) Syst. 2012;21(8):1496–505. https://doi.org/10.1109/TVLSI.2012.2211049.
11. Schlottmann C, Hasler P. FPAA empowering cooperative analog-digital signal processing. In: 2012 IEEE international conference on acoustics, speech and signal processing (ICASSP); 2012. p. 5301–4. https://doi.org/10.1109/ICASSP.2012.6289117.

12. Huang Y. Hybrid analog-digital co-processing for scientific computation. New York: Columbia University; 2018.
13. Rumberg B, Graham DW. A low-power field-programmable analog array for wireless sensing. In: Sixteenth international symposium on quality electronic design; 2015. p. 542–546. https://doi.org/10.1109/ISQED.2015.7085484.
14. Tatulian A, DeMara RF. A reconfigurable and compact spin-based analog block for generalizable nth power and root computation. In: 2021 IEEE computer society annual symposium on VLSI (ISVLSI); 2021. p. 302–7. https://doi.org/10.1109/ISVLSI51109.2021.00062.
15. Abuelma'Atti MT, Abuelmaatti AM. A new current-mode CMOS analog programmable arbitrary nonlinear function synthesizer. Microelectron J. 2012;43(11):802–8. https://doi.org/10.1016/j.mejo.2012.07.003.
16. D'Angelo RJ, Sonkusale SR. A time-mode translinear principle for nonlinear analog computation. IEEE Trans Circuits Syst I Regul Pap. 2015;62(9):2187–95. https://doi.org/10.1109/TCSI.2015.2451912.
17. Koza JR, Bennett FH, Andre D, Keane MA, Dunlap F. Automated synthesis of analog electrical circuits by means of genetic programming. IEEE Trans Evol Comput. 1997;1(2):109–28. https://doi.org/10.1109/4235.687879.
18. Sapargaliyev YA, Kalganova TG. Open-ended evolution to discover analogue circuits for beyond conventional applications. Genet Program Evolvable Mach. 2012;13(4):411–43. https://doi.org/10.1007/s10710-012-9163-8.
19. Thangavel V, Song ZX, DeMara RF. Intrinsic evolution of truncated Puiseux series on a mixed-signal field-programmable soc. IEEE Access. 2016;4:2863–72. https://doi.org/10.1109/ACCESS.2016.2537983.
20. Miura S, et al. Scalability of quad interface p-MTJ for 1× nm STT-MRAM with 10 ns low power write operation, 10 years retention and endurance > 1011. 2020 IEEE symposium on VLSI technology; 2020. p. 1–2. https://doi.org/10.1109/TED.2020.3025749.
21. Verma S, Kaushik BK. Low-power high-density STT MRAMs on a 3-D vertical silicon nanowire platform. IEEE Trans Very Large Scale Integr (VLSI) Syst. 2016;24(4):1371–6. https://doi.org/10.1109/TVLSI.2015.2454859.
22. Shinji Y, Fukushima A, Nagahama T, Ando K, Suzuki Y. High tunnel magnetoresistance at room temperature in fully epitaxial Fe/MgO/Fe tunnel junctions due to coherent spin-polarized tunneling. Jpn J Appl Phys. 2004;43(4B):L588–90. https://doi.org/10.1143/JJAP.43.L588.
23. Shoun M, Hayakawa J, Ikeda S, Miura K, Hasegawa H, Endoh T, Ohno H, Hanyu T. Fabrication of a nonvolatile full adder based on logic-in-memory architecture using magnetic tunnel junctions. Appl Phys Express. 2008;1(9): 091301. https://doi.org/10.1143/APEX.1.091301.
24. Joshi VK, Barla P, Bhat S, Kaushik BK. From MTJ device to hybrid CMOS/MTJ circuits: a review. IEEE Access. 2020;8:194105–46. https://doi.org/10.1109/ACCESS.2020.3033023.
25. Zhu L, et al. Heterogeneous 3D integration for a RISC-V system with STT-MRAM. IEEE Comput Archit Lett. 2020;19(1):51–4. https://doi.org/10.1109/LCA.2020.2992644.
26. Chun KC, Zhao H, Harms JD, Kim T, Wang J, Kim CH. A Scaling roadmap and performance evaluation of in-plane and perpendicular MTJ based STT-MRAMs for high-density cache memory. IEEE J Solid-State Circuit. 2013;48(2):598–610. https://doi.org/10.1109/JSSC.2012.2224256.
27. Salehi S, DeMara RF. SLIM-ADC: spin-based logic-in-memory analog to digital converter leveraging she-enabled domain wall motion devices. Microelectron J. 2018;81:137–43. https://doi.org/10.1016/j.mejo.2018.09.012.
28. Zhang Y, et al. Compact modeling of perpendicular-anisotropy CoFeB/MgO magnetic tunnel junctions. IEEE Trans Electron Devices. 2012;59(3):819–26. https://doi.org/10.1109/TED.2011.2178416.
29. Parkin SSP, Fontana RE, Marley AC. Low-field magnetoresistance in magnetic tunnel junctions prepared by contact masks and lithography: 25% magnetoresistance at 295 K in mega-ohm micron-sized junctions. J Appl Phys. 1997;81(8):5521. https://doi.org/10.1063/1.364588.
30. Camsari KY, Salahuddin S, Datta S. Implementing p-bits with embedded MTJ. IEEE Electron Device Lett. 2017;38(12):1767–70. https://doi.org/10.1109/LED.2017.2768321.
31. Datta S. p-Bits for probabilistic computing. In: 2019 Device Research Conference (DRC); 2019. p. 35–6. https://doi.org/10.1109/DRC46940.2019.9046390.
32. Wunderlich RB, Adil F, Hasler P. Floating gate-based field programmable mixed-signal array. IEEE Trans Very Large Scale Integr Syst. 2012;21(8):1496–505. https://doi.org/10.1109/TVLSI.2012.2211049.
33. George S, et al. A programmable and configurable mixed-mode FPAA SoC. IEEE Trans Very Large Scale Integr Syst. 2016;24(6):2253–61. https://doi.org/10.1109/TVLSI.2015.2504119.
34. Choi Y, Lee Y, Baek SH, Lee SJ, Kim J. CHIMERA: a field-programmable mixed-signal IC with time-domain configurable analog blocks. IEEE J Solid-State Circuits. 2017;53(2):431–44. https://doi.org/10.1109/JSSC.2017.2757005.
35. Kubota H, et al. Quantitative measurement of voltage dependence of spin-transfer torque in MgO-based magnetic tunnel junctions. Nat Phys. 2008;4(1):37–41. https://doi.org/10.1038/nphys784.
36. Wang S, Lee H, Grezes C, Khalili P, Wang KL, Gupta P. MTJ variation monitor-assisted adaptive MRAM write. In: 2016 53rd ACM/EDAC/IEEE design automation conference (DAC); 2016. p. 1–6. https://doi.org/10.1145/2897937.2897979.
37. Yuan L, Liou SH, Wang D. Temperature dependence of magnetoresistance in magnetic tunnel junctions with different free layer structures. Phys Rev B. 2006;73(13): 134403. https://doi.org/10.1103/PhysRevB.73.134403.
38. Madec M, Kammerer JB, Hébrard L. Compact modeling of a magnetic tunnel junction—part II: tunneling current model. IEEE Trans Electron Devices. 2010;57(6):1416–24. https://doi.org/10.1109/TED.2010.2047071.
39. Gao Z, Dai L, Han S, Chih-Lin I, Wang Z, Hanzo L. Compressive sensing techniques for next-generation wireless communications. IEEE Wirel Commun. 2018;25(3):144–53. https://doi.org/10.1109/MWC.2017.1700147.
40. Chartrand R. Fast algorithms for nonconvex compressive sensing: MRI reconstruction from very few data. In: 2009 IEEE international symposium on biomedical imaging: from nano to macro; 2009. p. 262–5. https://doi.org/10.1109/ISBI.2009.5193034.
41. Septimus A, Steinberg R. Compressive sampling hardware reconstruction. In: Proceedings of 2010 IEEE international symposium on circuits and systems; 2010. p. 3316–9. https://doi.org/10.1109/ISCAS.2010.5537976.
42. Candès EJ. The restricted isometry property and its implications for compressed sensing. CR Math. 2008;346(9–10):589–92. https://doi.org/10.1016/j.crma.2008.03.014.
43. Marques EC, Maciel N, Naviner L, Cai H, Yang J. A review of sparse recovery algorithms. IEEE Access. 2018;7:1300–22. https://doi.org/10.1109/ACCESS.2018.2886471.
44. Bai L, Maechler P, Muehlberghuber M, Kaeslin H. High-speed compressed sensing reconstruction on FPGA using OMP and AMP. In: 2012 19th IEEE international conference on electronics, circuits, and systems (ICECS 2012); 2012. p. 53–6. https://doi.org/10.1109/ICECS.2012.6463559.

45. Maechler P, Studer C, Bellasi D, Maleki A, Burg A, Felber N, Kaeslin H, Baraniuk RG. VLSI design of approximate message passing for signal restoration and compressive sensing. IEEE J Emerg Select Top Circuits Syst. 2012;2(3):579–90. https://doi.org/10.1109/JETCAS.2012.2214636.
46. Protas E, Bratti JD, Gaya JFO, Drews P, Botelho SSC. Visualization methods for image transformation convolutional neural networks. IEEE Trans Neural Netw Learn Syst. 2018;30(7):2231–43. https://doi.org/10.1109/TNNLS.2018.2881194.
47. Juang C, Chiou C, Lai C. Hierarchical singleton-type recurrent neural fuzzy networks for noisy speech recognition. IEEE Trans Neural Netw. 2007;18(3):833–43. https://doi.org/10.1109/TNN.2007.891194.
48. Basodi S, Ji C, Zhang H, Pan Y. Gradient amplification: an efficient way to train deep neural networks. Big Data Min Anal. 2020;3(3):196–207. https://doi.org/10.26599/BDMA.2020.9020004.
49. Zand R, Camsari KY, Datta S, DeMara RF. Composable probabilistic inference networks using MRAM-based stochastic neurons. ACM J Emerg Technol Comput Syst (JETC). 2019;15(2):1–22. https://doi.org/10.1145/3304105.
50. Pourmeidani H, Sheikhfaal S, Zand R, DeMara RF. Probabilistic interpolation recoder for energy-error-product efficient DBNs with p-bit devices. IEEE Trans Emerg Top Comput. 2020. https://doi.org/10.1109/TETC.2020.2965079.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.