IEEE Open Journal of Nanotechnology

Received 27 October 2021; revised 20 December 2021; accepted 11 February 2022. Date of publication 23 February 2022; date of current version 15 March 2022. The review of this article was arranged by Associate Editor Yunlong Zi.

Digital Object Identifier 10.1109/OJNANO.2022.3153329

Low-Power Approximate RPR Scheme for Unsigned Integer Arithmetic Computation

KE CHEN [1] (Member, IEEE), WEIQIANG LIU [1] (Senior Member, IEEE), AHMED LOURI [2] (Fellow, IEEE), AND FABRIZIO LOMBARDI [3] (Fellow, IEEE)

[1]College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China
[2]Department of Electric and Computer Engineering, George Washington University, Washington, DC 20052 USA
[3]Department of Electric and Computer Engineering, Northeastern University, Boston, MA 02115-500502115 USA

CORRESPONDING AUTHORS: KE CHEN; WEIQIANG LIU (e-mail: chen.ke@nuaa.edu.cn; liuweiqiang@nuaa.edu.cn)

This research was supported in part by the NSFC under Grants 62101252 and 62022041, and in part by the NSF under Grants 1812495, 1953980, 1901165, CCF-1953961, and 1812467.

ABSTRACT A scheme often used for error tolerance of arithmetic circuits is the so-called Reduced Precision Redundancy (RPR). Rather than replicating multiple times the entire module, RPR uses reduced precision (inexact) copies to significantly reduce the redundancy overhead, while still being able to correct the largest errors. This paper focuses on the low-power operation for RPR; a new scheme is proposed. At circuit level, power gating is initially utilized in the arithmetic modules to power off one of the modules (i.e., the exact module) when the inexact modules' error is smaller than the threshold. The proposed design is applicable to (unsigned integer) addition, multiplication, and MAC (multiply and add) by proposing RPR implementations that reduce the power consumption with a limited impact on its error correction capability. The proposed schemes have been implemented and tested for various applications (image and DCT processing). The results show that they can significantly reduce power consumption; moreover, the simulation results show that the Mean Square Error (MSE) at the proposed schemes' output is low.

INDEX TERMS Digital arithmetic, approximation methods, redundant systems, integrated circuit design.

I. INTRODUCTION

The development of ever smaller devices brings promise for further improvement in the performance of future integrated circuits, yet it also leads to several new technical challenges, including the need for nanoarchitectures that reduce the uncertainty inherent to computation and communication at such small scales [1]. For example, the space radiation environment has a serious impact on the reliability of integrated circuits in the avionics industry. The large number of particles in space, such as α-particles and high-energy neutrons, etc., generates radiation effects on integrated circuits which cause Single Event Effects (SEEs). Single Event Transients (SET) can affect logic circuits by introducing transient spikes in voltage values; when sampled by registers or memory elements, this event may result in data corruption. Single Event Upsets (SEU) can changes the logic state of a discrete sequential element, such as a latch of flip-flop or a the value of a memory cell. The continued miniaturization of electronic devices has raised reliability concerns for mission-critical applications [2]. Besides, Nanoelectronic systems where the device architecture is not a crossbar array generally utilize hardware redundancy for fault tolerance [3]. There are many techniques to protect a digital circuit against so-called soft errors; one of the most common schemes is the use of replication, so either by Dual Modular Redundancy (DMR) to detect errors or Triple Modular Redundancy (TMR) to correct them [4]. A significant issue of modular redundancy is its implementation overhead; for example, TMR requires additional circuit area and power dissipation of more than twice as much as an unprotected design.

Recently, approximate computing has been proposed as a new approach for efficient low-power design; approximate computing generates good enough results rather than always accurate. It is primarily driven by applications related to human perception and inherent error resilience. Approximate computing can be applied to these applications due to the redundant nature of data sets with significant noise to relax numerical exactness. Approximate computing reduces power

consumption and increases performance, for example, by reducing the critical path delay. Approximate techniques can be applied at several levels, including circuits, architectures, and software [5]. The application of approximate computing to deep learning has also been studied [6]. Approximate arithmetic units' design has received significant interest at the circuit level due to its importance in many computing applications. Typical applications, such as DSP and machine learning, require arithmetic computing for addition (or accumulation) and multiplication. Addition has been extensively studied for approximate circuit implementations; various approximate adders have been proposed to reduce power consumption and delay [7].

An alternative to full redundancy (such as triplication or duplication) is to use Reduced Precision Redundancy (RPR) [8], [9]. RPR uses an exact (full precision) version of the module and two reduced precision (inexact) copies to implement a decision/voting logic to correct errors. It significantly lowers the redundancy overhead because the reduced precision copies are significantly smaller than the full precision implementation. The disadvantage of RPR is that minor errors cannot be corrected, because the reduced precision may induce minor differences in output values that are difficult to discriminate. However, this is not a problem for many applications in which error-tolerance is possible in the absence of large errors. It has been recently shown that this can also be applied to emerging circuits, such for example, Deep Learning Neural Networks (DNNs) [5] during training.

A further issue is encountered when RPR is used for simple modules and copies; namely, the voting logic is significantly more complex than TMR [8]. This occurs because the difference between the full precision and reduced precision copies needs to be computed and compared to a threshold, so both subtraction and comparison are needed. This logic may be as complex as some modules, such as an adder. The design of RPR for addition has been investigated in [10], in which it has been shown that the voting logic can be simplified. Addition, multiplication, and Multiply-And-Accumulate (MAC, also known as multiply and add) are some of the most commonly used operations in computing, such as for matrix multiplication [11], [12] and in Deep Learning Neural Networks [13]. In [14], stochastic computation has been proposed for implementing artificial neural networks with reduced hardware and power consumption, while retaining in most cases the required accuracy and still improving processing speed.

In this paper, the RPR scheme is further investigated by considering a low-power operation. Power gating is employed for the RPR scheme to power off one of the modules (i.e., the exact module) when the error introduced by the inexact RPR modules is smaller than the threshold. The proposed scheme is utilized for implementing addition, multiplication, and MAC operations. A detailed assessment is also pursued; the results show that the proposed scheme improves the average power consumption of RPR at a given truncation level, while incurring in a lower overhead in terms of error correction capabilities. The rest of the paper is organized as follows: Section II provides a brief overview of RPR and power gating. The proposed scheme is presented in Section III. The proposed RPR scheme by employing multiplier, adder, and MAC is evaluated in Section IV. The proposed scheme is then utilized in various applications (such as image and DCT processing) in Section V. The paper ends with the conclusion in Section VI.

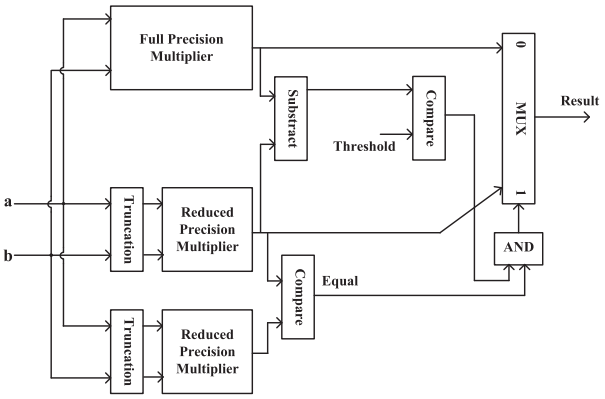

FIGURE 1. Traditional Reduced Precision Redundancy (RPR) for a Multiplier.

II. REVIEW

A. REDUCED PRECISION REDUNDANCY (RPR)

This section gives a brief overview of reduced precision redundancy (RPR) and an example using multipliers as modules. The implementation of a Reduced Precision Redundant (RPR) scheme for multipliers is shown in Fig. 1; in addition to the original full precision multiplier, two redundant copies with reduced precision are also used. The simplest method to reduce precision is to truncate the inputs by removing the K least significant bits. The error detection and correction logic compute the difference between the full precision output and the reduced precision copies. If such a difference is more significant than a specified threshold, it is assumed that an error has occurred.

To locate the error, the outputs of the two reduced precision copies are compared; if they are different, an error has occurred in one of the redundant copies. However, if they are equal, an error may have affected the full precision multiplier.

Consider an (exact) multiplier with N-bit inputs and reduced precision copies with $N - K$ bit inputs (so with a K-bit truncation); then, the maximum error on the inputs due to truncation is 2^{K-1}, and thus, the maximum error at the output is approximately $2 \times 2^K \times 2^N = 2^{N+K+1}$. However, such an error only occurs when both inputs are close to the largest value, and truncation on both input operands also introduces the worst-case error, so an occurrence with a low probability.

[15] has proposed a scheme combining IDMR (Improved Duplex Modular Redundancy) with TMR (Triple Modular Redundancy); this scheme uses novel conditions to compare the three module outputs. Unlike a DMR voter, a TMR voter considers the three inputs to establish the majority as the correct output. Within an inexact framework, the majority

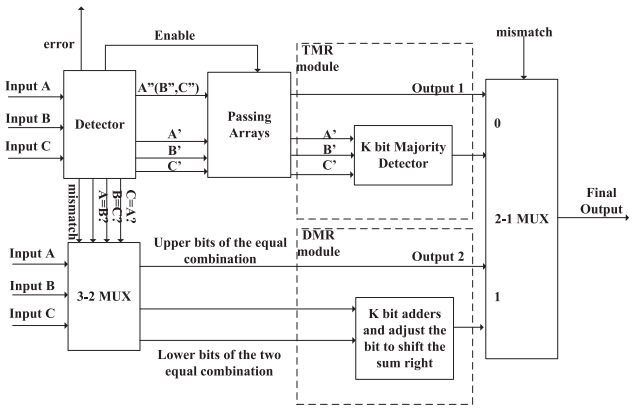

FIGURE 2. Schematic of Inexact TMR-DMR (ITDMR) voting scheme.

is established using various criteria. In this scheme, if the pair-wise differences between all three pairs of inputs are not larger than the threshold, then the three inputs are considered valid; however, in some cases, only two of the three inputs satisfy the threshold condition, so a different scheme must be used. Hence, ITDMR operates adaptively using TMR with approximate voting on a pair-wise fashion followed by IDMR (if required as a further voting configuration). Fig. 2 shows ITDMR in a block diagram form.

B. POWER GATING

In approximate systems, an inexact computational module must be designed in hardware to meet specific application requirements; so, at least some parts of a computational module could be turned on/off according to the desired configuration for an arithmetic operation. In this paper, power gating is used (so at circuit-level) to accomplish a bit-width reduction and modules shutting off. In this scheme, two transistors are added to a generalized CMOS gate: a PMOS in series with a pull-up network (PUN) and an NMOS in parallel with a pull-down network (PDN). There is also a slight increase in operational delay due to the additional PMOS transistor. The targeted error determines the value of the control signal CON; if the control signal CON is low, then the circuit operates as a normal gate. When the CON signal is high, the gate's output is forced to zero, regardless of the input signal. If only the input gates of a functional block (or module) are modified, a high value on CON turns off the functional block, because there is no switching activity in the connected circuits. Power gating incurs in hardware and delay overheads (albeit they are both very small, nearly negligible in most cases). This power-gating scheme is demonstrated in Fig. 3.

III. PROPOSED ARPR SCHEME

This section presents the proposed approximate RPR scheme (ARPR); this scheme is based on approximate computing and power gating. Consider as an example a multiplier as a basic module. The proposed RPR (multiplier) scheme is shown in Fig. 4. In Fig. 4, a'_{MSB} and b'_{MSB} are the MSBs of the truncated digits of the two operands. The signal LTE (Large Truncation

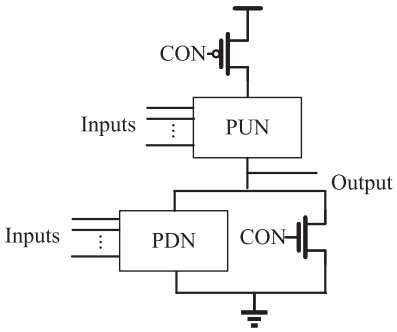

FIGURE 3. Schematic of Power gating scheme.

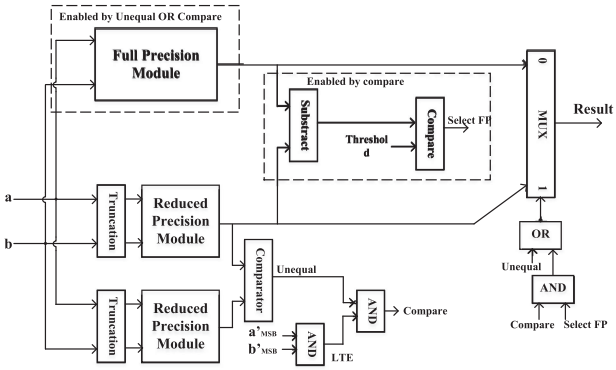

FIGURE 4. Proposed Approximate Reduced Precision Redundancy (ARPR) Scheme.

Error) is given by:

$$LTE = a_{MSB}{}' \ \& \ b_{MSB}{}' \tag{1}$$

ARPR consists of three groups of components depending on their operating state. The two RP multipliers, the comparator, and the control signals logic make up the first group and are always enabled.

- The RP (Reduced Precision) group generates the Inexact Result and two control signals: Unequal and compare.
- The FP (Full Precision) group is the FP multiplier which is enabled by $Unequal|compare$.
- The Compare group consists of a subtractor and a comparator. They are enabled by the signal *compare*.

The following cases can be distinguished:

- **CASE 1**: The two RP calculations are conducted, and the two RP results are compared. If the two results are different, the *Unequal* signal is set to '1,' and the FP calculation is enabled to generate the FP result as result. Otherwise, the system must decide whether it needs to compare the FP and the RP results.
- **CASE 2**: If LTE is '1,' so the MSB of the truncated parts of both inputs are '1's; then, in this case, a large truncation error is generated, and the comparison is needed. If LTE is '0,' the truncation error is small enough to be neglected, and the RP result is provided as the result. If the input operands follow a uniform distribution, the MSB of the truncated part has the same probability of

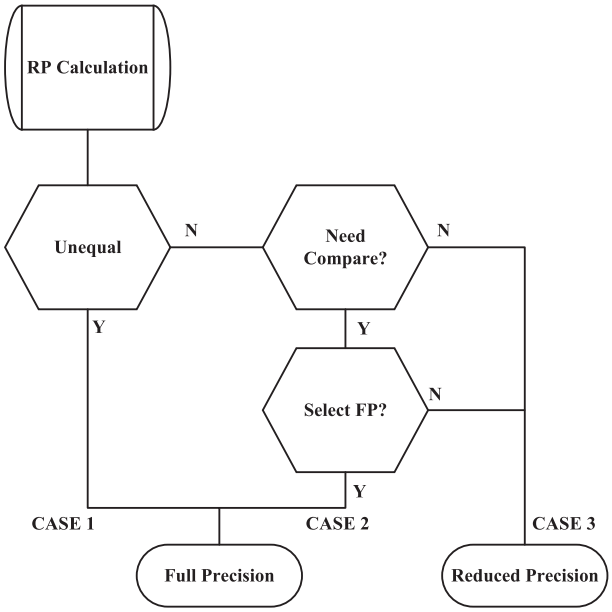

FIGURE 5. Flowchart of ARPR scheme.

being '0' or '1'. Thus, the probability of the LTE signal is '1' is $0.5^2 = 25\%$.

- **CASE 3**: If the truncation error is within the range $[TL, TU]$, the FP result is selected as output; otherwise, the RP result is selected. When the truncation error is smaller than TL (lower bound of the threshold), the error is considered small enough to be acceptable. However, if the truncation error is larger than TU (upper bound of the threshold), the error occurs at the upper bits of the full precision module. In this case, to prevent a significant error, the approximate result is select.

The flowchart of the proposed ARPR is shown in Fig. 5. CASE 1, CASE 2, and CASE 3 are the above three cases for generating the result. Power gating is employed to save power; in the error-free case, 75% of the decisions (voting) are undertaken only by the two RP copies, reducing power consumption and circuit delay.

Next, the two widely used operations of multiplication and addition are analyzed for error analysis. So, consider first an unsigned (exact) integer multiplier with N-bit inputs and K-bit truncation. Then, the output of the multiplier can take values in a range between 0 and $2^{2N} - 1$. If the operands are a and b, the output of an RPR multiplier can be expressed as:

$$\begin{aligned} a_r \times b_r &= (a + a_t) \times (b + b_t) \\ &= a \times b + a_t \times b + a \times b_t + a_t \times b_t \end{aligned} \quad (2)$$

where a_t and b_t are the errors introduced when K bits are truncated (in which case they can take values in the range of $1 - 2^K$ to 0). In an unsigned format, truncation can only produce negative errors; therefore, the difference between the outputs of the exact and RPR multipliers is given by:

$$a \times b - a_r \times b_r = -(a_t \times b + a \times b_t + a_t \times b_t) \quad (3)$$

Since both a_t and b_t are negative, the most significant error (in magnitude) occurs when a and b are large (the most significant values in magnitude are given by $2^N - 1$), i.e., when $a_t = b_t = 1 - 2^K$, and its magnitude is given by $2^{N+K+1} - 2^{N+1} - 2^{2K} + 1$,which can be approximated to 2^{N+K+1}, and this is also an upper bound.

Consider next addition; for unsigned addition with N-bit inputs and K bit truncation, the adder's output can take values in a range between 0 and $2^{N+1} - 2$.

If the operands are a and b, the output of the RPR adder copies can be expressed as:

$$a_r + b_r = (a + a_t) + (b + b_t) = a + b + a_t + b_t \quad (4)$$

The difference between the outputs of the full and reduced precision adders is given by:

$$a + b - (a_r + b_r) = -(a_t + b_t) \quad (5)$$

Similarly, the most significant error occurs when $a_t = b_t = 2^K - 1$. Its magnitude is given by $2^{(K+1)} - 2$, which can be approximated as $2^K + 1$. This is also an upper bound.

IV. EVALUATION AND COMPARISON OF APPROXIMATE RPR SCHEME

This evaluation considers circuit and error measures as well as a combined metric. For circuit performance, figures of merit such as delay and power are measured. As an error metric and consistent with other papers dealing with RPR, the MSE is measured.

Circuit Evaluation: To assess performance, the computing and voting modules are implemented in Verilog HDL and Synthesized by Synopsys Design Compiler with 28 nm technology. For the proposed 2 multiplier ARPR, the worst case and average scenarios are measured. For the worst-case (CASE 2), the RP, FP, and compare groups are all enabled. For the average case, the percentages of CASE 1, CASE 2, and CASE 3 are measured, and then the weighted average delay and power are calculated. For a traditional RPR, all computation modules are utilized and powered on during the entire computation process, so there is only one case. The weighted average delay and weighted average power are given in (6) and (7), where $Delay_{Ci}$ and $Power_{Ci}$ denote the latency and power consumption of CASE i, and P_{Ci} denotes the percentage of CASE i in the simulation sequence.The weighted average delay and weighted average power are given in (6) and (7), where $Delay_{Ci}$ and $Power_{Ci}$ denote the latency and power consumption of CASE i, and P_{Ci} denotes the percentage of CASE i in the simulation sequence.

$$\begin{aligned} Delay_{we} &= Delay_{C1} \times P_{C1} + Delay_{C2} \\ &\quad \times P_{C2} + Delay_{C3} \times P_{C3} \end{aligned} \quad (6)$$

$$\begin{aligned} Power_{we} &= Power_{C1} \times P_{C1} + Power_{C2} \\ &\quad \times P_{C2} + Power_{C3} \times P_{C3} \end{aligned} \quad (7)$$

Error Evaluation: To evaluate the impact of errors on ARPR/RPR schemes, random (uniformly distributed) errors

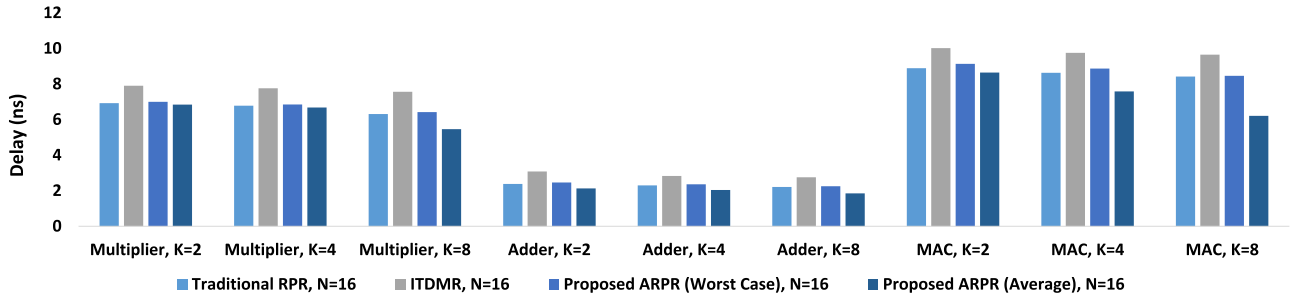

FIGURE 6. Delay for Different Approximate Redundant Scheme with N=16.

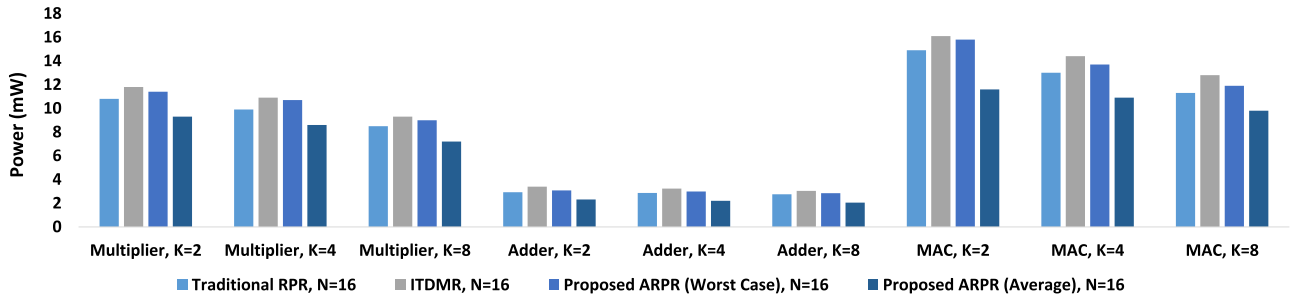

FIGURE 7. Power Consumption for Different Approximate Redundant Scheme with N=16.

have been injected in the implementations; then, the Mean Square Error (MSE) across all simulations is used to compare them, the equation of the MSE is listed in (8). As discussed previously, a SET is considered for error insertion. The error is only inserted to one of the three computational modules (the three modules have the same probability for SET injection). For each module, one of the bits is flipped in the partial product element; for example, in an 8-bit multiplier and truncation level $K = 2$, 8^2 partial product elements are present in a full precision multiplier. If this module is affected by a SET, then the value of one of the 64 partial product elements is opposite of the correct value (for the RPR multiplier, 6^2 partial product elements are utilized).

$$MSE = \frac{1}{N}\sum_{i=0}^{N-1}[APR(i) - EXA(i)]^2 \tag{8}$$

Power-Area-MSE (PAM) Product: The proposed ARPR is a low-power scheme but with mid-level performance in area and error. Thus, a combined figure of merit referred to as the Power-Area-MSE Product (PAM Product) is calculated to evaluate the three schemes (i.e., proposed ARPR, traditional RPR, and ITDMR); the scheme with the smaller PAM product indicates that it has the overall best-combined performance for these three crucial figures of merit. In this the paper, the PAM Product ratio is employed to denote the combined performance of the different schemes. The PAM product for a traditional RPR scheme are defined as normalized to 1; the PAM product ratio is given in (9).

In this the paper, the PAM Product ratio is employed to denote the combined performance of the different schemes. The PAM product for a traditional RPR scheme are defined as normalized to 1; the PAM product ratio is given in (9).

$$\begin{aligned} PAM\ Product &= Power \times Area \times MSE) \\ PAM\ Product\ Ratio &= \frac{PAM\ Product_{target\ scheme}}{PAM\ Product_{traditional\ scheme}} \end{aligned} \tag{9}$$

The Delay and the power consumption evaluation for the approximate redundant scheme with 16-bit operands are shown in Figs. 6 and 7. Table 1 to Table 3 demonstrate the area, MSE, and the PAM Ratio. The following conclusion can be drawn from the results.

1) The delay and power consumption are reduced with an increase of the truncation level (while introducing a larger error at the output).
2) The average power consumption and delay are significantly reduced using power gating as only the two RPR modules are initially active for computation.
3) For comparison purposes, ITDMR has the worst performance among the three schemes when considering power, delay, and area.
4) As for the error, the traditional RPR scheme has the least MSE, while the proposed scheme is the second best.
5) The PAM product ratio shows that the proposed scheme has the smallest value, that indicates the best performance when considering this combined figure of merit.

V. APPLICATIONS

A. IMAGE PROCESSING: PIXEL BRIGHTEN WITH ADDTION ARPR

This application requires the input of two identically sized images and produces a third image of the same size; each pixel value of the output image is the sum of the corresponding

TABLE 1 Error and Hardware Evaluation of Multiplier ARPR Scheme

		N=8		N=16		
		K=2	K=4	K=2	K=4	K=8
Area (μm^2)	Convention RPR	3400.54	2430.97	14167.11	12663.63	9064.02
	ITDMR	4028.74	2665.68	15103.33	13498.41	9704.85
	Proposed ARPR	3647.39	2567.54	14521.54	13064.84	9239.57
MSE ($\times 10^3$)	Convention RPR	9.73	13.62	30.86	35.74	47.97
	ITDMR	10.54	15.37	31.57	36.01	50.17
	Proposed ARPR	10.43	14.98	31.05	35.88	49.04
PAM Product Ratio	Convention RPR	1.00	1.00	1.00	1.00	1.00
	ITDMR	1.68	1.78	1.71	1.49	1.59
	Proposed ARPR	0.89	0.9	0.96	0.85	0.82

TABLE 2 Error and Hardware Evaluation of Adder ARPR Scheme

		N=8		N=16		
		K=2	K=4	K=2	K=4	K=8
Area (μm^2)	Convention RPR	1108.67	978.28	2203.75	1910.44	1596.38
	ITDMR	1407.84	1284.71	2558.16	2194.85	2014.43
	Proposed ARPR	1254.89	1087.65	2385.49	2062.31	1704.12
MSE ($\times 10^3$)	Convention RPR	3.43	15.98	3.05	15.88	226.04
	ITDMR	3.92	18.48	4.06	19.07	263.61
	Proposed ARPR	3.73	17.62	3.86	17.74	247.97
PAM Product Ratio	Convention RPR	1	1	1	1	1
	ITDMR	1.39	0.99	1.12	1.1	1.19
	Proposed ARPR	0.78	0.91	0.84	0.92	0.93

TABLE 3 Error and Hardware Evaluation of MAC ARPR Scheme

		N=8		N=16		
		K=2	K=4	K=2	K=4	K=8
Area (μm^2)	Convention RPR	4473.28	3258.34	17131.67	15032.06	10431.6
	ITDMR	5027.87	3878.18	17864.25	15786.33	11205.74
	Proposed ARPR	4632.92	3420.25	17384.74	15260.22	10703.79
MSE ($\times 10^3$)	Convention RPR	10.62	14.15	31.53	37.25	50.01
	ITDMR	11.94	16.47	32.52	38.94	52.33
	Proposed ARPR	11.87	15.61	32.47	38.78	52.14
PAM Product Ratio	Convention RPR	1	1	1	1	1
	ITDMR	1.5	1.68	1.13	1.19	1.21
	Proposed ARPR	0.98	0.97	0.85	0.95	0.96

pixel values from the two input images. A common variant simply allows a constant to be added to every pixel to enhance brightness.

In this section, image brightness is calculated by employing the proposed adder RPR. A constant value of 50 is added to the original image. The random error is inserted as discussed in Section IV; simulation runs for 1 million times to obtain the average. The results are shown in Fig. 8.

B. IMAGE PROCESSING: GAUSSIAN SMOOTHING USING MAC ARPR

For most image processing applications, noise can be present in an image during the steps of acquisition, compression, and even transmission. An additional step of noise reduction is often required for eliminating such noise and ensuring an overall acceptable performance. A Gaussian filter has excellent capabilities for blurring and noise reduction; moreover, it can be combined with an edge detector for digital image preprocessing with no excessive distortion of the original images

The Gaussian blur is a type of image-blurring filter that uses a Gaussian function (that is given by a normal distribution in statistics) when calculating the transformation of each pixel in an image. The Gaussian function in two dimensions is given by:

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2+y^2}{2\pi\sigma^2}} \tag{10}$$

In theory, the Gaussian distribution is always non-zero, because it would require an infinitely large convolution kernel; however, in practice, it can be effectively considered to be zero

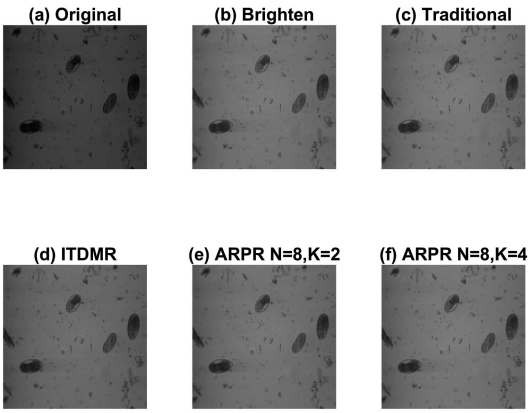

FIGURE 8. Image brighten using adder ARPR and Traditional RPR: (a) Original image; (b) Brighten image; (c) Traditional RPR (PSNR=47.54 dB); (d)ITDMR N=8, k=2 (PSNR=40.79) (e) ARPR N=8 k=2 (PSNR=44.79 dB); (f) ARPR N=8 k=4 (PSNR=41.86 dB).

TABLE 4 Gaussion Smoothing Simulation Results for N=8, K=2

Image	Traditional RPR	ITDMR	Proposed ARPR
Lena	38.87dB	34.87dB	35.53dB
Clock	37.84dB	34.56dB	34.99dB
Bridge	38.12dB	34.97dB	35.20dB
Tank	38.54dB	34.08dB	34.43dB

at more than three standard deviations from the mean. Hence, in this case, the kernel is truncated. The Gaussian kernel value is first converted from a floating point to an integer point format, and then the convolution is computed. To convert the kernel from a floating point to an integer value, the least (non zero) value of the kernel is found, and all elements of the kernel are then divided by this value, i.e., an integer kernel is used. For the correctness of a blurred image, the sum of the integer kernel is calculated, and after convolution of the input image with the integer kernel, the output is divided by the sum to obtain the corrected image.

In this section, Gaussian smoothing is implemented with the proposed MAC ARPR. Convolution with Gaussian filters is executed for four commonly used images in Table 4. The random error is inserted as discussed previously. One million simulations have been executed to find the average.

C. INTEGER DISCRETE COSINE TRANSFORM WITH MULTIPLICATION ARPR

The Discrete Cosine Transform (DCT) removes the cor- relation of image elements in the transform domain [19]; it is considered a quasi-optimal transform and has been widely applied in image and video coding/compression. In [20], rather than floating-point numbers, the DCT uses integers in the transform matrix; the core transformation is a signed integer transform involving no floating-point calculation and achieving high accuracy. The core transformation can be completed only with simple signed integer matrix operations, so multiply and addition operations; therefore, its computational complexity is significantly reduced. In matrix notation, the discrete two-

FIGURE 9. Gaussian Smoothing with MAC RPR Scheme (N=8, K=2): (a) Exact Gaussian Smoothing; (b) With Traditional RPR; (c) With Proposed ARPR; (d)ITDMR.

TABLE 5 Average PSNR for DCT for N=16

Truncation Level (K)	Traditional RPR	ITDMR	Propose ARPR
2	39.85dB	34.72dB	36.63dB
4	37.96dB	33.54dB	35.50dB
8	33.48dB	32.15dB	33.34dB

dimensional Radix-8 DCT is given by $Y = P \cdot X \cdot P^T$, where X is the 8×8 input image frame, P is the transform matrix, and Y is the transformed (output) matrix. As described in [21], one of the most common transform matrix (as also utilized in this paper) is:

$$p = \begin{bmatrix} 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 \\ 89 & 75 & 50 & 18 & -18 & -50 & -75 & -89 \\ 83 & 36 & -36 & -83 & -83 & -36 & 36 & 83 \\ 75 & -18 & -89 & -50 & 50 & 89 & 18 & -75 \\ 64 & -64 & -64 & 64 & 64 & -64 & -64 & 64 \\ 50 & -89 & 18 & 75 & -75 & -18 & 89 & -50 \\ 36 & -83 & 83 & -36 & -36 & 83 & -83 & 36 \\ 18 & -50 & 75 & -89 & 89 & -75 & 50 & -18 \end{bmatrix}$$

ARPR is used as the essential processing step for matrix multiplication. For evaluating the unsigned integer DCT, 16-bit images are used, and these 20 images are from the TESTIMAGE database. For the matrix element with a negative number, subtraction is used for the accumulated result. Random (uniformly distributed) errors are injected in the ITDMR, traditional RPR, and the ARPR schemes; then, the average Peak Signal Noise Ratio (PSNR) is measured for all 20 images at different values of K. The IDCT is implemented using the proposed MAC ARPR. The original 20 images undertake the convolution operation with the transform matrix; the random error is inserted as discussed in Section IV. Simulation is run one million times to calculate the average result.

The same trend is found for the three applications as shown in Table 5, namely, the traditional RPR scheme has the highest PSNR while the image quality from ITDMR is the worst.

The proposed ARPR scheme has a middle-level error performance, and the PSNR is reduced by increasing the truncation level. Fig. 9 provides an example of the DCT application with different RPR scheme using the Lena as test image. The simulation results also confirm the conclusion drawn in Section IV.

VI. CONCLUSION

This paper has presented novel schemes for Approximate Reduced Precision Redundancy (ARPR) as applicable to commonly used operations, such as multiplication, addition, and MAC in an unsigned integer format. The proposed schemes utilize power gating in the multipliers and adders when processing is executed to target errors. Novel designs are proposed; initially, the thresholds used for detecting errors within the RPR schemes have been determined analytically and checked by simulation. Then, both error tolerance and related implementation figures of merit have been analyzed. The results have shown that a reduction of 60 percent on the power can be achieved with only a marginal impact on computation quality, as established using the MSE. As further applications, computation of brightness and Gaussian smooth in image processing and integer DCT have been presented; the results have confirmed a reduction in power of at least 50% on average using the proposed ARPR scheme. Using the combined metric of the PAM (power area MSE) product for performance and error tolerance, it has been shown that the proposed ARPR scheme improves over ITDMR and a traditional RPR.

A discussion on the technical contents of this paper follows to address the relevance of this work to nanotechnology systems. Computing in the nano ranges necessitates novel approaches [22], [23] to address those scenarios that affect the reliable operation of computing systems at such low feature sizes [3]. At nanoscales, soft errors are more likely to occur; computational errors/faults in computational modules/units for many applications need stringent requirements for reliable data storage and processing [24]. It has been widely reported that due to the increased density and the complexity of the physical layout [25], nanotechnology devices show considerable variations to process, voltage, and temperature (PVT) at manufacturing [26]; at operation time, these variations can cause at least an erroneous outcome to occur at the output of a nanoscale circuit [27], in few cases leading to a catastrophic failure.

Remedial schemes such as those involving redundancy are therefore required, especially for mission-critical and high-integrity systems. Current schemes employ a large redundancy (such as TMR) resulting also in high power dissipation; while utilizing still replication, the proposed RPR scheme achieves a reduction in hardware (at the expense of introducing a reduced accuracy) that makes such arrangement very suitable for nanotechnology. Hence, the proposed RPR scheme can be applied to nanotechnology systems as in the following cases:

- Mitigation of data corruption in emerging memories due to hard and soft errors [28].
- Tolerance of processing (mostly of an arithmetic nature) using redundancy-based schemes [18], [29].

In both these two cases, the arrangement proposed in this manuscript is generic and therefore, it can be directly employed with little or no change in decision/selection hardware and only the input module designs are changed to accommodate the reduced redundancy configuration. Therefore, the proposed RPR scheme can be used in many other specific applications to enhance resilience (as demonstrated in previous sections); this paper has focused on its fundamental principles for implementation and analysis to enable a designer to properly use it. Future research will be directed to identify more nanotechnology-based applications (beside the two identified above) to which the proposed RPR scheme can be utilized.

REFERENCES

[1] S. Roy and V. Beiu, "Majority multiplexing-economical redundant fault-tolerant designs for nanoarchitectures," *IEEE Trans. Nanotechnol.*, vol. 4, no. 4, pp. 441–451, Jul. 2005, doi: 10.1109/TNANO.2005.851251.

[2] Z. Luo and M. Martonosi, "Accelerating pipelined integer and floating-point accumulations in configurable hardware with delayed addition techniques," *IEEE Trans. Comput.*, vol. 49, no. 3, pp. 208–218, Mar. 2000, doi: 10.1109/12.841125.

[3] T. J. Dysart and P. M. Kogge, "Reliability impact of N-modular redundancy in QCA," *IEEE Trans. Nanotechnol.*, vol. 10, no. 5, pp. 1015–1022, Sep. 2011, doi: 10.1109/TNANO.2010.2099131.

[4] J. T. Kao, M. Miyazaki, and A. R. Chandrakasan, "A 175-MV multiply-accumulate unit using an adaptive supply voltage and body bias architecture," *IEEE J. Solid-State Circuits*, vol. 37, no. 11, pp. 1545–1554, Nov. 2002, doi: 10.1109/JSSC.2002.803957.

[5] G. Li *et al.*, "Understanding error propagation in deep learning neural network (DNN) accelerators and applications," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal.*, 2017, pp. 1–12.

[6] M. de la Guia Solaz and R. Conway, "Razor based programmable truncated multiply and accumulate, energy-reduction for efficient digital signal processing," *IEEE Trans. Very Large Scale Integration Syst.*, vol. 23, no. 1, pp. 189–193, Jan. 2015, doi: 10.1109/TVLSI.2014.2300173.

[7] R. H. Brackert, M. D. Ercegovac, and A. N. Willson, "Design of an on-line multiply-add module for recursive digital filters," in *Proc. 9th Symp. Comput. Arithmetic*, 1989, pp. 34–41, doi: 10.1109/ARITH.1989.72807.

[8] C. Jeangoudoux and C. Lauter, "A correctly rounded mixed-radix fused-multiply-add," in *Proc. IEEE 25th Symp. Comput. Arithmetic*, 2018, pp. 21–28, doi: 10.1109/ARITH.2018.8464818.

[9] K. Chen, F. Lombardi, and J. Han, "Design and analysis of an approximate 2D convolver," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Nanotechnol. Syst.*, 2016, pp. 31–34, doi: 10.1109/DFT.2016.7684065.

[10] N. Kanekawa, E. H. Ibe, T. Suga, and Y. Uematsu, *Dependability in Electronic Systems: Mitigation of Hardware Failures, Soft Errors, and Electro-Magnetic Disturbances*. Berlin, Germany: Springer, 2010.

[11] M. Nicolaidis, "Design for soft error mitigation," *IEEE Trans. Device Mater. Rel.*, vol. 5, no. 3, pp. 405–418, Sep. 2005, doi: 10.1109/TDMR.2005.855790.

[12] I. A. Noufal and M. Nicolaidis, "A CAD framework for generating self-checking multipliers based on residue codes," in *Proc. Des., Automat. Test Eur. Conf. Exhib. (Cat. No PR00078)*, 1999, pp. 122–129, doi: 10.1109/DATE.1999.761107.

[13] M. Nicolaidis and R. O. Duarte, "Fault-secure parity prediction booth multipliers," *IEEE Des. Test Comput.*, vol. 16, no. 3, pp. 90–101, Jul.–Sep. 1999, doi: 10.1109/54.785842.

[14] B. Shim, S. R. Sridhara, and N. R. Shanbhag, "Reliable low-power digital signal processing via reduced precision redundancy," *IEEE Trans. Very Large Scale Integration Syst.*, vol. 12, no. 5, pp. 497–510, May 2004, doi: 10.1109/TVLSI.2004.826201.

[15] K. Chen, J. Han, and F. Lombardi, "Two approximate voting schemes for reliable computing," *IEEE Trans. Comput.*, vol. 66, no. 7, pp. 1227–1239, Jul. 2017, doi: 10.1109/TC.2017.2653780.
[16] B. Shim and N. R. Shanbhag, "Energy-efficient soft error-tolerant digital signal processing," *IEEE Trans. Very Large Scale Integration Syst.*, vol. 14, no. 4, pp. 336–348, Apr. 2006, doi: 10.1109/TVLSI.2006.874359.
[17] M. A. Sullivan, "Reduced precision redundancy applied to arithmetic operations in field programmable gate arrays for satellite control and sensor systems," M.S. thesis, Dept. Elect. Comput. Eng., Naval Postgraduate School, Monterey, CA, USA, 2008.
[18] A. Ullah, P. Reviriego, S. Pontarelli, and J. A. Maestro, "Majority voting-based reduced precision redundancy adders," *IEEE Trans. Device Mater. Rel.*, vol. 18, no. 1, pp. 122–124, Mar. 2018, doi: 10.1109/TDMR.2017.2781186.
[19] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. C-23, no. 1, pp. 90–93, Jan. 1974, doi: 10.1109/T-C.1974.223784.
[20] J. Wu and Y. Li, "A new type of integer DCT transform radix and its rapid algorithm," in *Proc. Int. Conf. Electric Inf. Control Eng.*, 2011, pp. 1063–1066, doi: 10.1109/ICEICE.2011.5778056.
[21] P. K. Meher, S. Y. Park, B. K. Mohanty, K. S. Lim, and C. Yeo, "Efficient integer DCT architectures for HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 1, pp. 168–178, Jan. 2014, doi: 10.1109/TCSVT.2013.2276862.
[22] A. Amirany, K. Jafari, and M. H. Moaiyeri, "High-performance radiation-hardened spintronic retention latch and flip-flop for highly reliable processors," *IEEE Trans. Device Mater. Rel.*, vol. 21, no. 2, pp. 215–223, Jun. 2021, doi: 10.1109/TDMR.2021.3060055.
[23] S. Liu, P. Reviriego, and F. Lombardi, "Protection of associative memories using combined tag and data parity (CTDP)," *IEEE Trans. Nanotechnol.*, vol. 20, pp. 1–9, 2021, doi: 10.1109/TNANO.2020.3042114.
[24] F. Sun and T. Zhang, "Defect and transient fault-tolerant system design for hybrid CMOS/nanodevice digital memories," *IEEE Trans. Nanotechnol.*, vol. 6, no. 3, pp. 341–351, May 2007, doi: 10.1109/TNANO.2007.893572.
[25] F. Martorell, S. D. Cotofana, and A. Rubio, "Fault tolerant structures for nanoscale gates," in *Proc. 7th IEEE Conf. Nanotechnol.*, 2007, pp. 605–610, doi: 10.1109/NANO.2007.4601264.
[26] S. Tabrizchi, F. Sharifi, A. Badawy, and Z. M. Saifullah, "Enabling energy-efficient ternary logic gates using CNFETs," in *Proc. IEEE 17th Int. Conf. Nanotechnol.*, 2017, pp. 542–547, doi: 10.1109/NANO.2017.8117467.
[27] S. Jeng, J. Lu, and K. Wang, "A review of reliability research on nanotechnology," *IEEE Trans. Rel.*, vol. 56, no. 3, pp. 401–410, Sep. 2007, doi: 10.1109/TR.2007.903188.
[28] S. Liu, P. Reviriego, P. Junsangsri, and F. Lombardi, "Probabilistic approximate computing at nanoscales: From data structures to memories," *IEEE Nanotechnol. Mag.*, vol. 16, no. 1, pp. 16–24, Feb. 2022, doi: 10.1109/MNANO.2021.3126092.
[29] K. Chen, L. Chen, P. Reviriego, and F. Lombardi, "Efficient implementations of reduced precision redundancy (RPR) multiply and accumulate (MAC)," *IEEE Trans. Comput.*, vol. 68, no. 5, pp. 784–790, May 2019, doi: 10.1109/TC.2018.2885044.