Can We Securely Use Approximate Computing?

Pruthvy Yellu and Qiaoyan Yu
University of New Hampshire
Durham, NH 03824, USA
Email: qiaoyan.yu@unh.edu

***Abstract*—[1] Approximate computing (AC) techniques bring in an alternative way to improve energy efficiency for computing systems, at the cost of acceptable reduction on accuracy. Unfortunately, recent literature indicates that approximate computing systems could be vulnerable to new security threats. Approximation mechanisms maybe leveraged to introduce more errors than the level that the AC systems can tolerate. This work analyzes the existing metrics for accuracy from attack detection point of view. A differential metric is proposed to enlarge the Trojan induced difference on accuracy, delay and power, thus easing Trojan detection. Furthermore, this work proposes a unique attack detection method for AC systems to reduce false negative rate on Trojan detection. Our case study shows that the proposed method can reduce the false negative rate by 93%.**

I. Introduction

Approximate Computing (AC) techniques are new methods that trade off accuracy for improved performance and energy efficiency. AC has great potential in the era of big data, especially in the applications of image processing, audio translation, and artificial neural network [1], [2], where the slightly degraded accuracy can be tolerated. For instance, the audio signal does not need exact translation since most humans are not sensitive to the sound beyond the range of 20 Hz to 20 kHz [3]. AC techniques are also widely utilized in image processing, due to limited perceptual capabilities of human beings for image or video.

Current research effort on AC techniques mainly focuses on the development of approximation algorithms at software level [4]–[6] or circuit design methods at hardware level [7]–[9]. Approximation on hardware is often achieved by utilizing imprecise arithmetic units [10], [11], approximate register files [12], or approximate accelerators [13]. Recently, there emerges strong attention to examine the security vulnerabilities of AC systems. The work [14] indicates that the utilization of AC techniques in a system will expose the system to new types of security attacks. For example, approximate storage elements are vulnerable to the memory refresh control attack, which lowers the DRAM data retention rate. Attacks on the threshold control in Phase Change Memory (PCM) will lead the memory to experience the increased number of quantization errors. Hardware tampering on the arithmetic approximate circuits could enlarge the inaccuracy on outputs, which may exceed the tolerable range as designed at the specification phase. As the attacks mentioned above are likely to be executed in a burst mode, the suddenly accumulated errors could be catastrophic from AC systems.

This work is dedicated to explore the feasible methods to determine whether the AC system of interest is compromised by a hardware Trojan or not. More specifically, the key contributions of this work are as below: we identify the best metric to measure accuracy for AC systems, and incorporate differential metrics into a new Trojan detection framework to reduce the false negative rate of Trojan detection. Our method will facilitate users to utilize AC techniques securely.

The rest of this paper is organized as follows. Section II introduces the approximation techniques applied in software and hardware. Section III presents the attack model interested in this work. Section IV proposes a metric suitable for attack detection in AC systems. Our novel attack detection method is described in Section V. Evaluation results are available in Section VI. This work is concluded in Section VII.

II. Related Work

Approximation techniques employed in computational applications can tolerate different amount of errors in outputs. The desired degree of approximation is either pre-determined in the system design phase or configured during the deployment stage. Functional approximation or over-scaling techniques are typically employed in arithmetic units to trade off accuracy for better power and speed [8], [15]–[17]. In work [18], an accuracy-controlled approximate adder is presented to reduce the critical path delay. To compensate the degraded accuracy, an error detection and correction mechanism is used after the adder module.

As the trade-off between accuracy and improved performance play a vital role in approximate computing, different error metrics are being used to assess the degraded accuracy. The most common metrics to measure accuracy are error distance (ED), relative error distance (RED), normalized error distance (NED), mean relative error distance (MRED) [18]–[21]. In work [18], MRED is adopted for accuracy tuning. In another recent work [19], an approximate quaternary adder is proposed to improve the performance of its deployment in FPGAs. The work [19] also uses MRED to evaluate the mean error distance introduced by employing their proposed approximate adder.

Although current approximate techniques have achieved remarkable progress in terms of balancing accuracy and power/delay performances, new security vulnerabilities induced by the utilization of AC algorithms and techniques

[1]This work is an invited paper for ISCAS 2020 Special Session: "Stochastic and Approximate Computing."

978-1-7281-3320-1/20/$31.00 ©2020 IEEE

have not been widely investigated yet. The metrics adopted for accuracy assessment do not take the intentional errors into consideration. An improper metric could overlook the tailored errors originated from hardware Trojans. Moreover, the specially designed hardware tampering may not yield noticeable changes on delay and power. Thus, it will be more challenging to detect a hardware Trojan in AC systems than in precise computing systems.

III. Attack Model

We assume that malicious modification is performed after the approximation computing system is completed by the designer. The attacker has full knowledge of the approximation algorithm and has access to the netlist of the approximation unit. In contrast, users will see the approximate computing unit as a black box, but they can obtain the outputs of the approximate unit via simulation. The behavioral model of the approximate unit is available as a golden reference. The goal of hardware tempering on approximate computing systems is to suddenly accelerate system errors and lead to catastrophic effects.

IV. Proposed Metrics for Attack Detection

Before we deploy an approximate computing module in applications, it is critical to examine if any malicious modification has been made on the approximate module. As AC systems do not achieve 100% accuracy in nature, the verification metrics for AC functions will be different than what we typically use for precise functions. In this section, we compare the existing metrics and select the proper metric for the purpose of hardware Trojan detection.

A. Metrics to Measure Accuracy

We first compare the metrics used to measure accuracy in existing literature. Equation (1) [18], [22]–[24] defines the mean value of accuracy, which measures to what extent the imprecise output R_e deviates from the precise output R_c.

$$ACC_{amp} = 1 - \frac{|R_c - R_e|}{R_c} \tag{1}$$

In which, R_e and R_c are the results of the imprecise and precise outputs being translated to a decimal number, respectively. A higher ACC_{amp} means that the imprecise output is closer to the precise result.

Accuracy can also be measured by Hamming distance as expressed in Eq. (2) [18], [22]. Different than ACC_{amp}, ACC_{hm} does not consider the weight that the incorrect bit carries in the output. Instead, we pay attention to the ratio of the number of mismatched bits B_e between the precise and imprecise outputs and the bit width of output B_w.

$$ACC_{hm} = 1 - \frac{B_e}{B_w} \tag{2}$$

The third metric for accuracy ACC_{apx} only considers the bits from precise computation submodules (e.g. for most significant bits) and ignores those bits produced by the approximate submodules (e.g. for least significant bits). Assume T_{apx} is the total number of correct cases after we neglect the bits computed by the approximate submodules. Equation (3) introduces the ratio of T_{apx} over the total number of test cases T_{total}. This metric assesses the accuracy of the critical portion of interest.

$$ACC_{apx} = \frac{T_{apx}}{T_{total}} \tag{3}$$

Equation (4) defines the accuracy ACC_{gen} as the ratio of the total number of completely correct cases T_c over the total number of test cases T_{total}. This metric equally considers all the mismatch bits between the imprecise and precise outputs. ACC_{gen} is generally referred as pass rate [22].

$$ACC_{gen} = \frac{T_c}{T_{total}} \tag{4}$$

To quantitatively compare the accuracy metrics defined by Eqs. (1)-(4), we report the different accuracies for a 8-bit adder, compromised of varied number of approximate 1-bit full adders in Table I. As can be seen, all accuracy metrics decrease with the increasing number of precise modules. Among all metrics, the speed of accuracy reduction of ACC_{amp} is the slowest one. ACC_{gen} is more strict than ACC_{amp}, ACC_{hm} and ACC_{apx} since ACC_{gen} computation leads to quicker accuracy drop with increase in number of approximate modules in a circuit. For instance, when five of the precise 1-bit full adders in the 8-bit adder are replaced with the approximate ones, the accuracy ACC_{gen} is reduced to 22.66%.

TABLE I: Comparison of accuracy measured with different metrics applied to 8-bit hybrid (precise and approximate) adders.

No. precise (approx.) units	ACC_{amp}	ACC_{hm}	ACC_{apx}	ACC_{gen}
1 (7)	90.28%	65.15%	82.95%	16.46%
2 (6)	95.04%	68.39%	82.57%	19.34%
3 (5)	97.45%	71.87%	81.84%	22.66%
4 (4)	98.64%	75.48%	80.47%	26.56%
5 (3)	99.22%	79.24%	78.13%	31.25%
6 (2)	99.52%	83.38%	75%	37.50%
7 (1)	99.72%	88.91%	75%	50.00%
8 (0)	100%	100%	100%	100%

B. Differential Metric to Measure Delay and Power

In Section IV-A, we observe that different metrics for accuracy make large difference on the measured values for the same configuration of adder. In this subsection, we propose a differential metric to indicate if the approximation unit is compromised or not. Assume the measurement from the precise functional unit is P, the measurement from the original approximate functional unit is A, and the test result from the tampered approximate functional unit is A'. In a typical verification, one will compare the delay or power of the untampered and tampered approximate unit in a way expressed in Eq. (5).

$$DIFF_{scs} = \frac{A' - A}{A} \tag{5}$$

Where $DIFF_{scs}$ stands for the differential side-channel signal (e.g. delay and power). Since the hardware Trojan logics

are designed to be stealthy, the Trojan-induced increase on delay and power is typically insubstantial. Thus, the metric $DIFF_{scs}$ is not suitable for attack detection.

We propose to bring the side channel signals obtained from the precise implementation to the differential metric expressed in Eq. (6).

$$\Delta_{DIFF_{scs}} = \frac{P(A'-A)}{A*A'} = \frac{P}{A} * \frac{A'-A}{A'} \quad (6)$$

The ratio $\frac{P}{A}$ in Eq. (6) is a constant enlargement factor. By multiplying the $\frac{P}{A}$ factor, the tampering induced difference on the side-channel signals of approximate functional units will be increased. Thus, the proposed metric will improve the sensitivity of Trojan detection and thus reduce the false negative detection rate.

V. Proposed Attack Detection Method for Approximate Computing Systems

We incorporate the proposed metrics mentioned in Section IV in a unique Trojan detection method designed for approximate computing systems. Figure 1 depicts the key idea of our proposed Trojan detection method, which exploits difference among Precise-Approximate-Unit under test for Security Examination (PAUSE). Our method does not only rely on the difference between the output/delay/power of the tamper-free approximate system and the unit under test, but also leverages the difference that are intentionally introduced in the phase of designing the approximation algorithm. The approximate system designer typically provides performance improvement and trades-off accuracy. The ratio $\frac{P}{A}$ mentioned in Eq.(6) is available at design time since $\frac{P}{A}$ is usually reported by the designer. A small number of input patterns will be used to measure the output accuracy, delay and power of the behavioral approximate computation unit and its hard/firm macro. Next, the metric $\Delta_{DIFF_{scs}}$ will be computed accordingly. We compare that metric with the threshold ξ suggested by the approximate system designer. If $\Delta_{DIFF_{scs}}$ exceeds the threshold, something abnormal occurs (likely induced by a Trojan attack). Then more evaluation is required to increase the confidence level of attack detection. The computation of $\Delta_{DIFF_{scs}}$ and threshold comparison are performed by the unit under test (UUT) user.

VI. Experimental Results

A. Experimental Setup

In this section, we evaluated the effectiveness of proposed metrics and the PAUSE method for hardware Trojan detection. We implemented two approximate ripple-carry adders with 8-bit and 16-bit inputs. Three 1-bit full adder (FA) modules in the precise adder were selected to apply the approximation algorithm proposed in [15]. In our simulation, the inserted hardware Trojans can swap the inputs for the approximate and precise full adders. We also implemented the corresponding 8-bit and 16-bit precise adders as comparison references. All the adders were synthesized with a TSMC 65nm technology. The delay and power consumption were reported by the Synopsys Design Compiler. Logic-gate level simulations for output verification were conducted in the Cadence NCVerilog tool.

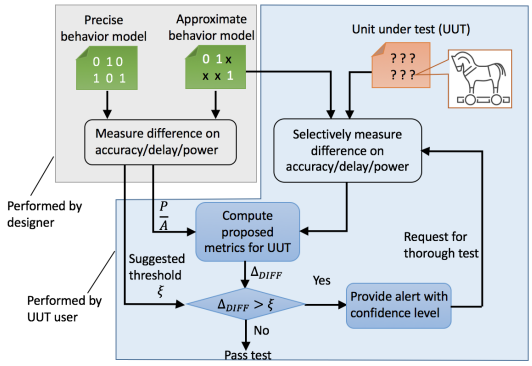


Fig. 1: Proposed attack detection flow for approximate computing systems.

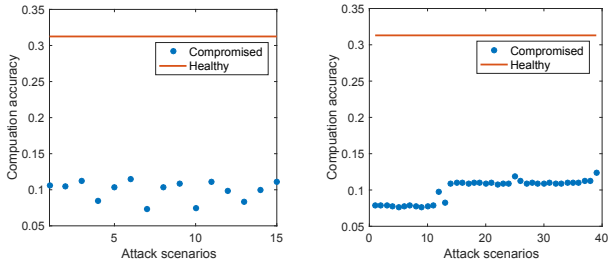
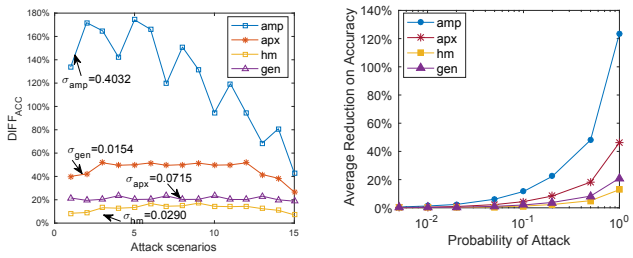


Fig. 2: Computation accuracy ACC_{gen} of approximate (a) 8-bit and (b) 16-bit adders.

B. False Negative Rate of Proposed Trojan Detection

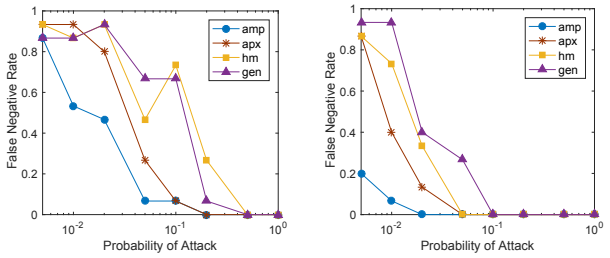
An approximate adder tampered by a hardware Trojan will yield a different accuracy on its primary output than what is designed. For instance, as reported in Table I, the accuracy ACC_{gen} for a 8-bit adder with 5 precise FAs and 3 approximate FAs is 0.3175 if no Trojans are inserted. In total, there will be 15 different attack cases of precise↔approximate input swapping for an 8-bit adder. As shown in Fig. 2(a), the computation accuracy drops to about 0.1. We repeated the same experiment for a 16-bit approximate adder (composed of 13 precise FAs and 3 approximate FAs) and observed the similar accuracy reduction. The comparison of accuracy shown in Fig. 2 indicates that the Trojan attacks on the approximate arithmetic unit will lead to significant decrease on accuracy and different attack scenarios results in a variation on accuracy.

We continue to use the 8-bit adder to examine the difference on accuracy $DIFF_{ACC}$ induced by the Trojan attack. Figure 3(a) shows that the reduction on accuracy varies with different accuracy metrics. The use of ACC_{amp} leads to the most significant variation on $DIFF_{ACC}$ if the attack is injected in different locations. In contrast, ACC_{hm} is the least sensitive to the attack location. The standard deviation of the $DIFF_{ACC}$ for each accuracy metric is labelled in Fig. 3(a). Because the Trojan attack is stealthy and is not always-



(a) (b)

Fig. 3: Impact of diverse attack scenarios on accuracy. (a) Differential accuracy and (b) accuracy reduction due to Trojan.



(a) (b)

Fig. 4: Impact of number of test cases and accuracy metrics on false negative attack detection rate of baseline. (a) 2^{10} and (b) 2^{14} test cases.

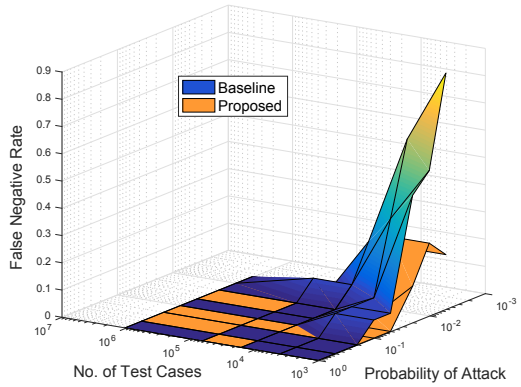


Fig. 5: Impact of number of test cases and Trojan triggering probability on false negative rate of our method and baseline.

on, we emulate the Trojan attack with different triggering probabilities. When the probability of attack decreases to 10^{-2}, the average reduction on accuracy is almost close to zero (shown in Fig. 3(b)). This means, it is challenging to detect the presence of Trojans based on the variation on accuracy.

We further examined the false negative rate (FNR) of Trojan detection due to the use of different accuracy metrics. As shown in Figs. 4(a) and (b), ACC_{amp} allows us to obtain the least FNR compared with other metrics. This is because that metric is more sensitive to the attack location and the probability of Trojan triggering. This experiment also indicates that a smaller probability of attack will lead to a higher

TABLE II: FNR reduction achieved by proposed Trojan detection method.

Metric	$\Delta_{DIFF_{amp}}$	$\Delta_{DIFF_{hm}}$	$\Delta_{DIFF_{apx}}$	$\Delta_{DIFF_{gen}}$
FNR reduction	67%	80%	67%	93%

TABLE III: Differential metrics of two approximate adders

Metrics	8-bit approximate adder		16-bit approximate adder	
	Average	Std	Average	Std
$DIFF_{Delay}$	4.80%	0.0539	3.22%	0.0253
$DIFF_{Power}$	18.75%	0.0131	8.08%	0.0082
$DIFF_{PDP}$	17.72%	0.0435	3.18%	0.0231
Proposed	36.98%	0.0994	13.30%	0.0321

FNR. To overcome the high FNR, more test cases will be required. The overall impact of the number of test cases and the probability of attack on FNR is provided in Fig. 5. The metric adopted here was $\Delta_{DIFF_{amp}}$. The maximum reduction on FNR achieved by other metrics is listed in Table II. Our case study shows that our method can reduce the FNR by up to 93% over the baseline, which compares the accuracy ACC_{amp} with a pre-determined accuracy range.

C. Average and Deviation of Proposed Metrics

In addition to accuracy, traditional Trojan detection methods for precise functional designs also use delay and power as side-channel signals to examine if a unit under test is sabotaged by Trojans or not. A Trojan which yields a smaller change on delay or power is more stealthy for detection. The goal of our new metric Δ_{DIFF} in Section IV-B is to enlarge the impact of a Trojan on the side-channel signals. The average differential delay $DIFF_{Delay}$ (differential power $DIFF_{Power}$) for the adder with/without hardware Trojans are compared in Table III. As can be seen, $DIFF_{Delay}$ is smaller than $DIFF_{Power}$. Following the definition provided Eq. (6), we calculated $DIFF_{PDP}$ for the power-delay-product (PDP). As shown in Table III, the proposed metric $DIFF_{PDP}$ improves the difference between the healthy design and the Trojan tampered copy. Comparing with the metric $DIFF_{Delay}$, our method increases the difference by over 32% for the 8-bit adder. For the 16-bit adder, the proposed method obtains more than 10% difference than using delay. The standard deviation (std) in our Trojan detection experiments is less than 0.1.

VII. Conclusion

The security vulnerabilities of approximate computing systems are highlighted in recent literature. As there are a lack of effective methods to detect hardware Trojans inserted in AC systems, this work fulfills the pressing need. We identify suitable metrics for accuracy assessment and propose a differential metric for Trojan attacks in AC systems. We further incorporate those metrics in a new attack detection method *PAUSE*. Our case studies confirm that the proposed metrics and Trojan detection method can significantly reduce the false negative rate of Trojan detection for approximate adders.

REFERENCES

[1] F. Qiao, N. Zhou, Y. Chen, and H. Yang, "Approximate computing in chrominance cache for image/video processing," in *Proc. 2015 IEEE Intl. Conf. on Multimedia Big Data*, pp. 180–183, April 2015.

[2] S. Venkataramani, A. Ranjan, K. Roy, and A. Raghunathan, "Axnn: Energy-efficient neuromorphic systems using approximate computing," in *Proc. ISLPED'14*, pp. 27–32, Aug 2014.

[3] S. W. Smith, "The scientist and engineer's guide to digital signal processing." https://www.dspguide.com/ch22/1.htm. [Online; accessed 15-Dec-2019].

[4] L. C. H. Esmaeilzadeh, A. Sampson and D. Burger, "Architecture support for disciplined approximate programming," in *Proc. 7th Intl. Conf. on Architectural Support for Programming Languages and Operating Syst.*, pp. 301–312, March 2012.

[5] A. Raha, S. Venkataramani, V. Raghunathan, and A. Raghunathan, "Quality configurable reduce-and-rank for energy efficient approximate computing," in *Proc. 2015 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2015.

[6] W. Baek and T. Chilimbi, "Green: A framework for supporting energy-conscious programming using controlled approximation," in *Proc. ACM SIGPLAN Conference on Programming Language Design and Implementation '10 (PLDI)*, June 2010.

[7] A. Sampson, J. Nelson, K. Strauss, and L. Ceze, "Approximate storage in solid-state memories," in *Proc. 2013 MICRO*, pp. 25–36, Dec 2013.

[8] N.Zhu, W. L. Goh, W. Zhang, K. S. Yeo, and Z. H. Kong, "Design of low-power high-speed truncation-error-tolerant adder and its application in digital signal processing," *IEEE Trans. on Very Large Scale Integ. (VLSI) Syst.*, vol. 18, pp. 1225–1229, Aug 2010.

[9] L. Yang and B. Murmann, "Sram voltage scaling for energy-efficient convolutional neural networks," in *Proc. 2017 ISQED*, pp. 7–12, March 2017.

[10] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, pp. 124–137, Jan 2013.

[11] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," *IEEE Trans. on Computer-Aided Design of Integr. Circuits and Syst.*, vol. 32, pp. 124–137, Jan 2013.

[12] D. Jeong, Y. H. Oh, J. W. Lee, and Y. Park, "An edram-based approximate register file for gpus," *IEEE Design Test*, vol. 33, pp. 23–31, Feb 2016.

[13] H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger, "Neural acceleration for general-purpose approximate programs," in *Proc. 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, Dec 2012.

[14] P. Yellu, N. Boskov, M. Kinsy, and Q. Yu, "Security threats on approximate computing systems," in *Proc. GLSVLSI'19*, pp. 387–392, May 2019.

[15] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, pp. 124–137, Jan 2013.

[16] M. Shafique, W. Ahmad, R. Hafiz, and J. Henkel, "A low latency generic accuracy configurable adder," in *Proc. 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–6, June 2015.

[17] N.Zhu, W. L. Goh, and K. S. Yeo, "An enhanced low-power high-speed adder for error-tolerant application," in *Proc. 2009 Intl. Symp. on Integr. Circuits*, pp. 69–72, Dec 2009.

[18] A. B. Kahng and S. Kang, "Accuracy-configurable adder for approximate arithmetic designs," in *Proc. DAC Design Automation Conference 2012*, pp. 820–825, June 2012.

[19] S. Boroumand, H. P. Afshar, and P. Brisk, "Approximate quaternary addition with the fast carry chains of fpgas," in *Proc. 2018 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 577–580, March 2018.

[20] J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders," *IEEE Transactions on Computers*, vol. 62, pp. 1760–1771, Sep. 2013.

[21] C. Liu, J. Han, and F. Lombardi, "A low-power, high-performance approximate multiplier with configurable partial error recovery," in *Proc. 2014 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 1–4, March 2014.

[22] V. Benara and S. Purini, "Accurus: A fast convergence technique for accuracy configurable approximate adder circuits," in *Proc. 2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 577–582, July 2016.

[23] Ning Zhu, W. L. Goh, and K. S. Yeo, "An enhanced low-power high-speed adder for error-tolerant application," in *Proc. of the 2009 12th International Symposium on Integrated Circuits*, pp. 69–72, Dec 2009.

[24] M. Osta, A. Ibrahim, H. Chible, and M. Valle, "Approximate multipliers based on inexact adders for energy efficient data processing," in *Proc. 2017 New Generation of CAS (NGCAS)*, pp. 125–128, Sep. 2017.