Coherent Design of Hybrid Approximate Adders: Unified Design Framework and Metrics

Ardalan Najafi, *Student Member, IEEE,* Moritz Weißbrich,
Guillermo Payá Vayá and Alberto Garcia-Ortiz, *Member, IEEE*

***Abstract*—Numerous approximate adders have been proposed in the literature in response to the languishing benefits of technology scaling. However, they have been obtained with an ad-hoc and non-systematic methodology which does not fully exploit the design space possibilities. This work provides a conceptual framework for the systematic design of approximate adders including hybrid and non-equally segmented approaches as well as more robust error metrics. The framework discriminates the scenarios where approximate processing does not provide significant benefits from those where it does; in this later case, it aids to obtain optimal configurations for the adders. Experimental results with a commercial technology assesses the significant improvements of our systematic approach. Furthermore, a case study with a processor enhanced with an approximate accelerator highlights the usability of the methods.**

***Index Terms*—Approximate adders, error metrics, computer vision, generic template, automatic design framework.**

I. Introduction

INCREASING vulnerability of computing systems to errors in underlying circuits is a growing concern nowadays. As device sizes shrink, variability in physical circuit characteristics is increasing as a result of manufacturing challenges at the device level. Traditionally, performance and energy penalties are paid to ensure that all devices work correctly under all possible conditions. Accordingly, achieving deterministic behavior becomes increasingly expensive, as variability increases. In response to the languishing benefits of technology scaling, rather than hiding variations under expensive guard-bands, designers have begun to relax traditional correctness constraints and deliberately expose hardware variability to higher levels of the computing stack [1].

As one of the most promising energy-efficient paradigms, approximate computing increases energy efficiency that leverages application-level tolerance to few errors in many applications, including image processing, multimedia applications, and machine learning. Approximate computing, a promising technique to reduce power, area and delay in VLSI design, approximates a system by redesigning its logic circuit. It exploits the gap between the level of accuracy required by the applications and that provided by the computing system, for achieving diverse optimizations [2], [3].

The researchers in the field of approximate computing have paid special attention to adders, one of the key components of arithmetic circuits. In fact, a surprisingly large number of approximate adders [4]–[11] have been proposed in the literature: segmented adders where an n-bit adder is divided into k-bit sub-adders [4]–[6]; carry select adders in which multiple sub-modules are used [7], [8]; approximate full adders where the full adder is approximated [10], [11]; and speculative adders which are built upon the observation that the critical path is rarely activated in traditional adders [12]–[14]. The current situation is such that even a fair comparison of approximate adders is a challenging endeavor [15], [16].

A closer investigation into the variety of approximate adders (and in general, approximate units) shows that the current approaches use two philosophies for the error: 1) *small errors* or 2) *unlikely errors*. In the first philosophy, the errors of the approximate unit are engineered to be small in magnitude, even if they are frequent. The rationale is that those errors are masked by the intrinsic truncation and noise error of the system, and therefore they do not degrade considerably the quality of the application. Examples of this philosophy are the *Lower-part OR Adder* (LOA) [10] and the *Optimized Lower-part Constat-OR Adder* (OLOCA) [17]. In the second philosophy, the errors are engineered to appear infrequently, even if they are large when they appear. The rationale is that the application can overcome errors if they are sporadic. Examples of this philosophy are the *Almost Correct Adder* (ACA) [4], the *Generic Accuracy Configurable Adder* (GeAr) [9], the *Error Tolerant Adder* (ETAII) [6] and the Equal Segmentation Adder (ESA) [5]. Furthermore, although all the architectures are conceptually different, they share a common characteristic: they have been obtained with an ad-hoc and non-systematic methodology. A remarkable exception is, however, the GeAr Adder that uses the idea of a template [9], but it is not optimal. In real applications, hybrid solutions including both philosophies are desirable and even required; however, they cannot be obtained with current approaches. Therefore, an integrated CAD framework, able to model, analyze, and optimize all the previous architectures and error philosophies, is urgently required.

A key problem when considering simultaneously the two philosophies is the quantification of the errors. The authors working with the *small errors* philosophy tend to prefer error metrics as the standard deviation, or the mean average error that measure the average magnitude of the errors. This metric, however, strongly penalizes large infrequent errors. The

A. Najafi and A. Garcia-Ortiz are with the Institute of Electrodynamics and Microelectronics, University of Bremen, 28359 Bremen, Germany.
E-mail: {ardalan, agarcia}@item.uni-bremen.de
See http://www.ids.uni-bremen.de
M. Weißbrich and G. Payá Vayá are with the Institute of Microelectronic Systems, Leibniz Universität Hannover, 30167 Hannover, Germany.
E-mail: {weissbrich, guipava}@ims.uni-hannover.de
This work is funded by the German Research Foundation (DFG) project GA 763/4-1.

authors working with the *infrequent error* philosophy tend to favor metrics as the average number of errors which quantify the error probability; however, this metric strongly penalizes architectures with small errors. In a real scenario, the two effects have to be considered in a single metric, but it is not possible with the current approaches.

The selection of an approximate architecture is strongly influenced by the timing constraints of the hardware [16]. The constraints determine the internal structure of the unit (sequential structure with linear cost, parallel-prefix structure with a logarithmic cost, etc.), which determines the reduction in cost achieved by the approximate units. With relaxed timing constraints, an exact adder is implemented with a ripple-carry structure and an ESA almost decreases the delay by a factor of two. With more stringent timing constraints, an exact adder is implemented with a parallel-prefix architecture, where the delay increases as $\lceil \log_2(n) \rceil$ and the use of an ESA is just marginally reducing the delay. In this case, non-equally segmented sub-adders may be preferable. The effects of the internal adder architecture have been studied in [16]; it illustrates the potentials for non-equally segmented approximate adders which are currently disregarded. A unified description for approximate adders is still an open problem.

The goal of this work is to provide a conceptual framework for the systematic design of approximate adders, including hybrid and non-equally segmented approaches, as well as more robust error metrics. It is organized in five main sections. Firstly, in Section II, we extend the current error metrics to systematically analyze approximate units; secondly, in Section III, we develop a template to design and analyze approximate adders, including the *small-errors* and the *infrequent-errors* philosophies; it includes a detailed mathematical analysis of errors. Afterwards, Section IV describes how our design framework automatically chooses best configurations of the proposed template for given application constraints. For the validation, we provide two contributions: First, we present an experimental evaluation using a commercial CMOS technology in Section V; secondly, we present a case study with a MIPS processor ensued with an approximate accelerator in Section VI, to highlight the practical relevance of the proposed approaches. Finally, Section VII concludes the paper.

II. Metrics

The error is defined as the difference between approximate and accurate output results of the adder, i.e., $\varepsilon = \tilde{Y} - Y$, where $\tilde{Y}$ is the approximate (erroneous) output of the adder and Y is the accurate result. It is a random variable that can be characterized by its probability density function. However, from the perspective of an automated design framework, it is more convenient to use an error metric (a single number) to quantify the importance of the error. Several metrics have been proposed; among them, the most common ones are the *Average Error* (μ), the average number of errors (PE), the *Standard Deviation* (STD or σ), the *Mean Squared Error* (MSE) and the *Mean Absolute Error* (MAE). The authors in [18] defined the Error Distance and the Mean Error Distance (MED) to evaluate the arithmetic performance of approximate adders.

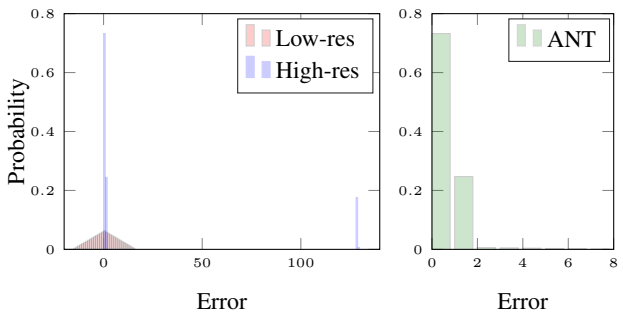

Fig. 1. Error histograms of a low-resolution input, a high-resolution input and an ANT output.

These metrics are actually the absolute error and the MAE, respectively. References [19], [20] use MED and error rate (similar to PE), while references [15], [21], [22] use relative error metrics. In summary, the most common metrics are:

$$PE = E\big[\delta(\varepsilon)\big] = \sum_j \delta(\varepsilon_j) Pr[\varepsilon_j] \ , \tag{1}$$

$$\sigma = \sqrt{E\big[(\varepsilon - \mu)^2\big]} = \sqrt{\sum_j (\varepsilon_j - \mu)^2 Pr[\varepsilon_j]} \ , \tag{2}$$

$$MSE = E\big[\varepsilon^2\big] = \mu^2 + \sigma^2 \ , \tag{3}$$

$$MAE = E\big[|\varepsilon|\big] = \sum_j |\varepsilon_j| Pr[\varepsilon_j] \ , \tag{4}$$

As mentioned in the introduction, PE favors the *infrequent error* philosophy, while STD and MSE favor the *small error* philosophy. MAE is more robust than MSE to the presence of large errors and it is commonly used as a compromise. However, it is insufficient. Let us consider a typical stochastic approach as the Algorithmic-Noise-Tolerant (ANT) approach [23]. In ANT, two versions of an algorithm are computed; one with a low-cost and low-resolution unit and another one with a high-resolution approximate-processing unit. Then, the two outputs are compared; if the difference is small, the approximate-processing output is chosen, otherwise the low-resolution one is chosen. After this selection process, the quality of the system is notably improved. An example is shown in Fig. 1, where the low-resolution signal uses 5 effective bits and the high-resolution signal is generated with a hybrid adder. The error histogram of the ANT output shows the quality of the resulting signal. Beside the simplicity of this example, none of the current metrics can be reliably used to quantify the error-cost of the high-resolution approximate-processing needed by ANT. To illustrate this fact, Table I reports the error metrics (PE and MSE) of the signal produced by three different adders as

Table I
Error metrics for approxiamte adders and resulting output quality after ANT processing

Adder	ANT input MSE	ANT input PE	ANT output MSE
ESA-4	10.9545	0.4688	7.4330
ETAII-3	14.9666	0.0547	0.8101
Hybrid	19.6405	0.2676	0.7539

well as the final signal quality measured as the MSE at the output of the ANT process. Analyzing only the MSE, ESA-4 should be the best adder, while analyzing the PE, ETAII-3 is preferable. In fact, a hybrid adder, whose MSE and PE are worse than the previous examples, provides the best solution. Accordingly, current metrics are misleading.

The key problem is that the current metrics do not capture the different behaviors of small and large errors. Low-magnitude errors get added to the final output and can be quantified with the MSE or STD; high-magnitude errors are detected by ANT and replaced by the low-resolution version of the algorithm. Thus, they contribute with a factor dependent on the error probability (PE) but relatively independent of the actual error magnitude. To palliate the lack of expressiveness of current metrics, we propose a new parameterizable metric that captures the requirements of stochastic applications, the *Saturated Mean Squared Error* (SMSE_τ). Mathematically, it is defined as:

$$SMSE_\tau = E\big[\min(\tau, |\varepsilon|)^2\big] \tag{5}$$

The parameter τ controls the behavior of the metric; large values of τ produce a cost similar to the MSE, while small values produce a cost similar to PE. As shown with more details in the experimental results, the new metric captures more precisely the error cost and can be used to explore different approximate units.

III. Hardware modeling

As previously mentioned, the existing approximate adders have been proposed based on two major philosophies: 1) *Infrequent errors*: Segmented adders where an n-bit adder is divided into k-bit sub-adders [4]–[6]. This class of adders mostly produce big errors with low probability of happening. 2) *Small errors*: Approximate full adders where the full adder is approximated [10], [11]. This group of adders approximate lower bits and hence, produce small errors with high probability of happening. Since the superiority of approximate adders depends on the application they are used in, a systematic and unified template for hybrid solutions, covering a relatively wide range of approximate adders, are desirable and even required.

At the same time, designers have so far considered segmented adders as *equally* segmented. The reason to consider only equal segmentation is to have the minimum delay and thereupon, an optimal approximate architecture. This consideration is appropriate when working with area-efficient, slow adders like Ripple-carry adder. However, working with fast and/or optimal exact adders replaced as sub-adders, asymmetric and non-equally segmented adders might outperform the equally-segmented ones. One example is the system reported in Table I.

In order to address the above-mentioned problems, in this paper, systematically, we propose a unified generic template for approximate adders, which combines the two philosophies.

A. Nomenclature

In this paper, each adder's name is followed by numbers. For ETAII and ESA, the number is the size of equal segmented sub-adders. For GeAr, the left number is the number of resultant bits contributing to the final sum, and the right one is the number of previous bits used for carry prediction. Regarding LOA, it is the number of OR gates. And finally, for LOCA [17], the numbers indicate the number of OR gates and constant-1s, from left to right, respectively.

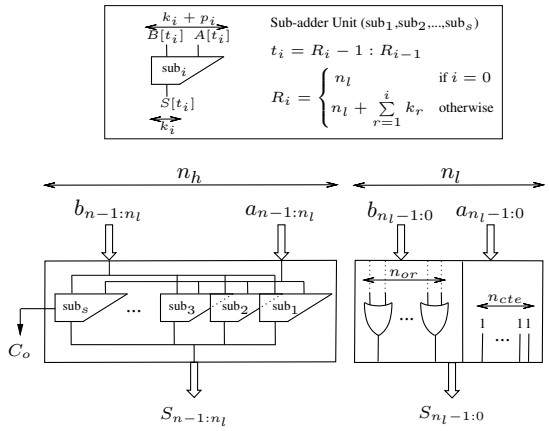

Fig. 2. The proposed generic template of approximate adders

In addition to that, we have a new naming system based on the proposed template architecture, which will be discussed later in this section.

B. The Generic Template of Approximate Adders

The generic template for approximate adders is shown in Fig. 2. As can be seen in the figure, the template is constructed using the two aforementioned approximation methodologies. An upper part which is constructed using a segmented adder (the infrequent-errors philosophy) and a lower part composed of OR gates and constant-1s (the small-errors philosophy). Variable n denotes the length of operands to be added, while n_l and n_h represent the bit-width of the lower and higher parts of the template, respectively.

As depicted in Fig. 2, while the lower part is constructed using n_{or} OR gates as well as n_{cte} constants, the upper part is instantiated by an adder. The adder can be either exact or approximate. In order to have a range of adders using the template, we replace the higher-significant part of the template with a segmented adder. The segmented adder splits the entire carry propagation path into a number of short paths and completes the carry propagations in these short paths concurrently. Let $\mathbb{k} = \{k_s, k_{s-1}, \ldots, k_1\}$ denote a vector including the size of resultant bits of each sub-adder, where s is the number of its sub-adders, k_1 is size of the resultant bits of the first (the lowest significant) sub-adder, k_s is size of the resultant bits of the last (the most significant) sub-adder, and so on. In addition, $\mathbb{P} = \{p_s, p_{s-1}, \ldots, p_1\}$ represents a vector including number of previous bits used for carry prediction. The vectors $\mathbb{K}$ and $\mathbb{P}$ describe the upper part of the architecture; the i^{th} sub-adder of the segmented adder which has a size of k_i+p_i, uses p_i previous bits to construct the input carry for the sub-adder. In order to name each adder developed using the template, we use the following format:

$$\boxed{k_s(p_s)k_{s-1}(p_{s-1})\ldots k_1(p_1)|n_{or}, n_{cte}}$$

Table II
EXAMPLES TO ILLUSTRATE HOW TO GENERATE SOME OF THE EXISTING APPROXIMATE ADDERS USING THE TEMPLATE, 16-BIT ADDERS

Adder	$k_s(p_s)k_{s-1}(p_{s-1})\dots k_1(p_1)\|n_{or},n_{cte}$
Exact adder	$16(0)\|0,0$
ETAII-4	$4(4)4(4)8(0)\|0,0$
ETAII-6	$6(6)10(0)\|0,0$
ESA-4	$4(0)4(0)4(0)4(0)\|0,0$
GeAr-26	$2(6)2(6)2(6)2(6)8(0)\|0,0$
LOA-8	$8(1)\|8,0$
LOCA-26	$8(0)\|2,6$

This way, using the template, not only a wide range of the existing approximate adders can be developed, but also hybrid and non-equal segmented approximate adders can be generated.

Table II tabulates some examples to illustrate how some of the existing approximate adders are constructed using our template. As can be seen in the table, different classes of approximate adders can be implemented, changing variables of the template. Since the GeAr adder can be constructed using our proposed template, ACAI and ACAII are also in the list of the adders which can be developed by the template.

In order to obtain the error formulas for the template, we divide the architecture into lower and higher parts. Then, we formulate the error metrics for each part. Finally, the error metrics of the template can be obtained combining the error formulas for each part of the template. Let us first introduce a term in order to make the formulas, presented in the rest of this paper, more compact:

$$B_i = \begin{cases} 0 & \text{if } i = 0 \\ \sum_{r=1}^{i} k_r & \text{otherwise} \end{cases} \tag{6}$$

In order to have the histogram of the upper part of the template which is a segmented adder, first, we need to review the Generate and Propagate concepts. Generate G_j and propagate P_j signals are computed using bitwise operations:

$$P_j = a_j \oplus b_j \quad , \quad G_j = a_j b_j \ . \tag{7}$$

where a_j and b_j are the primary inputs of the adder corresponding to the bit position j.

From Fig. 2, when $p_i + k_i - p_{i+1}$ bits generate a carry signal and the p_{i+1} bits propagate the carry, the sub-adder sub_i produces error with the magnitude -2^{B_i}:

$$\varepsilon_{h_i} = -2^{B_i} \ . \tag{8}$$

Given the fact that for uniformly-distributed inputs $Pr[G_j] = \frac{1}{4}$ and $Pr[P_j] = \frac{1}{2}$; the probability of error ε_{h_i} is calculated as follows:

$$\begin{aligned} Pr_{h_i} = Pr\big[\varepsilon_{h_i}\big] =& \frac{1}{2}\big(1 - (\frac{1}{2})^{k_i+p_i-p_{i+1}}\big)(\frac{1}{2})^{p_{i+1}} \\ =& \frac{2^{k_i+p_i-p_{i+1}} - 1}{2^{k_i+p_i+1}}. \end{aligned} \tag{9}$$

where Pr_{h_i} denotes the probability of error ε_{h_i}, for the uniformly distributed input vectors, produced by the i^{th} sub-adder for $i \in \{1, 2, \dots, s-1\}$.

Replacing Eq. (8) and Eq. (9) in Eq. (3) as well as Eq. (4), MSE and MAE of the segmented adder of Fig. 2 can be expressed as:

$$MAE_h = \sum_{i=1}^{s-1} |\varepsilon_{h_i}| Pr_{h_i} = 2^{n_h - 1 - k_s - p_s} - \frac{1}{2^{p_1+1}}, \tag{10}$$

and

$$\begin{aligned} MSE_h =& \sum_{i=1}^{s-2} \varepsilon_{h_i}^2 Pr_{h_i} \\ =& 2^{2n_h - 2k_s - p_s - 1} + \sum_{i=1}^{s-1} 2^{2B_{i-1} - p_i - 1}\big(1 - 2^{k_i}\big) \ . \end{aligned} \tag{11}$$

Since all the errors have the same sign, the mean error value of the segmented adder has the same value as its MAE:

$$\mu_h = -2^{n_h - 1 - k_s - p_s} + \frac{1}{2^{p_1+1}}, \tag{12}$$

The metrics presented above, implicitly show the relation between the error metrics and the higher significant sub-adder variables, (i.e. k_s and p_s).

Regarding the lower part of the template, for the uniform distributed data, each bit is uncorrelated and the error metrics can be calculated as a function of the error characteristics of each block. The total error of the lower part, ε_l, is the summation of the errors of each block, ε_i, with the corresponding weight, i.e., $\varepsilon_l = \sum_{i=0}^{n_l-1} \varepsilon_i 2^i$ [17]. Given the fact for the blocks used in the lower part of the template, i.e., OR gate and Cte-1, the error values are $\mu_{or} = -\frac{1}{4}$, $\sigma_{or}^2 = \frac{3}{16}$ and $\mu_{cte1-1} = 0$, $\sigma_{cte-1}^2 = \frac{1}{2}$, the error metrics of the lower part of the template are as follows:

$$\mu_l = 2^{n_{cte}-2} - 2^{n_l-2} \ , \tag{13}$$

$$\sigma_l^2 = 2^{2n_l-4} + \frac{5}{3}2^{2n_{cte}-4} - \frac{1}{6} \ , \tag{14}$$

$$\begin{aligned} MAE_l =& 2^{n_l-2} - 2^{n_{cte}-2} + \\ &+ \frac{1}{3}\left(\frac{3}{4}\right)^{n_l - n_{cte}} \left(2^{n_{cte}} - \frac{1}{2^{n_{cte}}}\right). \end{aligned} \tag{15}$$

The more details of the equations for the lower part of the template can be found in [17].

Combining the error metrics for the higher and lower part of the template, the error metrics for the template adder are calculated as follows:

$$\mu_{temp} = 2^{n_l}.\ \mu_h + \mu_l \ , \tag{16}$$

$$\sigma_{temp}^2 = 2^{2n_l}.\ \sigma_h^2 + \sigma_l^2 \ , \tag{17}$$

$$MAE_{temp} = 2^{n_l}.\ MAE_h + (1 - PE_h).MAE_l \ . \tag{18}$$

In order to have the formulas of the saturated metrics, the understanding of the histogram of the template is required. The histogram of the lower part of the template (LOCA) is a bimodal distribution with mean value of μ_l and standard deviation of σ_l presented in Eq. (13) and Eq. (14), respectively. The number of bins (intervals) in the histogram of the upper part of the template (Segmented adder) is equal to the number of segments. These intervals are -2^{B_i}, as mentioned earlier in

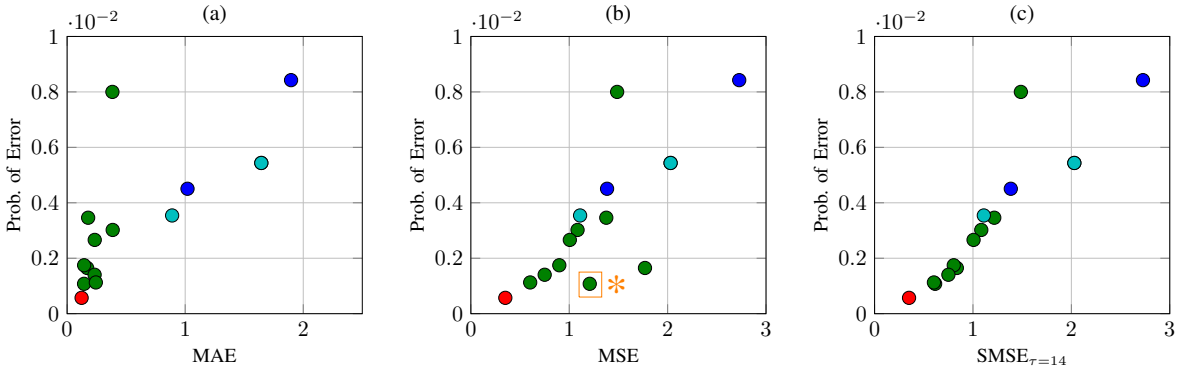

Fig. 3. The error prediction of different metrics in an image processing algorithm. Different colors correspond to different classes of approximate adders. The star * is used to mark an adder as an example to show a case where conventional metrics are misleading.

this section, in addition to the one at zero with $Pr_0 = 1 - PE_h$. The histogram of the template, as a result, is the repetition of the lower part histogram at the intervals of the higher part histogram. Let us consider that there is no overlap in the histogram of the template (which is true in the practical cases). Hence, the histogram consists of multiple bimodal distribution. Applying a threshold between these distributions, the following formula calculates the precise saturated MSE:

$$\begin{aligned} SMSE_\tau &\approx \sum_{i=1}^{s-1} Pr_{h_i} \min(\tau^2 \ , \ (\varepsilon_{h_i} + \mu_l)^2 + \sigma_l^2) \\ &= \sum_{i=1}^{s-1} Pr_{h_i} \min(\tau^2 \ , \ \varepsilon_{h_i}^2 + 2\varepsilon_{h_i}.\mu_l + MSE_l) \ . \end{aligned} \tag{19}$$

where s denotes the number of segments in the upper part segmented adder.

Note that this formula is a piecewise linear approximation of the exact error. It provides a remarkable accuracy for any τ that does not collide with one of the bimodal distributions that compose the PDF of the error. In the unlikely situation that τ hits one of those bimodal distributions, the actual error decreases slightly and Eq. (19) provides an upper bound of the error.

The other metrics can be calculated in a similar way. We have precisely modeled all the metrics, from conventional to saturated ones (SMSE, SMAE, SSTD), using the histogram of the template. It is part of the developed framework which will be discussed in the next section with more details.

IV. Design Methodology

In this section, the methodology of our framework is discussed. The proposed framework discriminates between the scenarios where approximate processing does not provide significant benefits from those where it does. In the later case, it aids in obtaining optimal configurations of the template.

In the first step, using precise error models, the framework finds all the possible configurations of the template for a given error constraint. As mentioned in the previous section, the possible configurations are conventional approximate adders, new hybrid and/or non-equal segmented adders, truncated exact adders, etc. The designer can choose between a traditional metric (MSE, MAE, STD, etc) or the saturated counterpart. In fact, in this step, a given threshold, based on the target application, needs to be specified. Based on the definition presented earlier in this paper, large enough thresholds result in conventional metrics.

In the next step, for a target frequency, the framework sorts the adders based on their expected silicon area. The sorting process is done based on the bit-width of the upper part adder, i.e. $\sum_{i=1}^{s}(k_i + p_i)$, as well as the length of the sub-adders, i.e. $\max(k_i + p_i) for \ i \in \{1, 2, \ldots, s\}$.

Finally, out of several possible architectures which have been sorted, the top ten architectures are synthesized to determine precisely the figures of merit. The complete procedure is automatic.

Observe that the current framework does require a precise characterization of the error (see the equations presented in the previous section), but just a relative rank for the area. However, it is also possible to model the area of the architectures. For example, we have modeled the area and delay of ETAII and ESA adders. Nevertheless, this precise characterization is out of scope of the current paper.

V. Experimental results

We have structured this section in four steps: Firstly, we evaluate the fidelity of our metric in a real algorithm. Secondly, we evaluate the exactness of the presented error formulas. Thirdly, we assess the quality of the adders designed with our framework. And finally, we show the usability of our framework.

In order to evaluate our new metric, we have compared the adders in a simple image processing algorithm which first calculates the average of the pixels using approximate adders and then the error probability is calculated after binarization. In Fig. 3, the probability of errors for 14 different 8-bit adders are shown versus the calculated error metrics. The colors correspond to different families of adders. The red color is the precise adder, the green ones are different configurations of ETAII (including the non-equal segmented ones), the light blue ones are OLOCA architectures and the dark blue ones are ESA adders. As can be seen in the figure, our saturated metric predicts the errors in this algorithm more precisely. There is

Table III
SIMULATION AND FORMULAS RESULTS FOR 16-BIT ADDERS

Adder	μ		MAE		σ		$SMSE_{\tau=16}$		$SMAE_{\tau=64}$		$SSTD_{\tau=256}$	
	sim	*math*	*sim*	*math*	*sim*	*math*	*sim*	*model*	*sim*	*model*	*sim*	*model*
4(8)12(0)|0,0	-7.54	-7.50	7.54	7.50	175.57	175.11	0.47	0.47	0.12	0.12	10.97	10.94
4(4)4(4)8(0)|0,0	-127.56	-127.50	127.56	127.50	690.95	690.78	15.01	15.00	3.75	3.75	60.14	60.12
7(1)1(1)8(0)|0,0	-127.51	-127.50	127.51	127.50	181.03	181.02	95.50	95.50	23.88	23.88	123.80	123.81
6(6)10(0)|0,0	-7.54	-7.50	7.54	7.50	87.53	87.32	1.88	1.87	0.47	0.47	21.88	21.83
8(4)4(1)4(0)|0,0	-7.50	-7.50	7.50	7.50	32.00	32.00	60.07	60.00	4.51	4.50	32.00	32.00
8(0)8(0)|0,0	-127.54	-127.50	127.54	127.50	127.99	128.00	127.50	127.50	31.89	31.88	128.00	128.00
12(0)|2,2	-3.00	-3.00	3.70	3.70	4.18	4.18	26.50	26.50	3.70	3.70	4.18	4.18
7(1)5(0)|2,2	-123.02	-123.00	123.55	122.83	216.93	217.22	80.31	80.30	17.84	17.84	107.20	107.24
8(4)5(0)|2,1	-5.50	-5.50	5.78	5.75	31.82	32.31	10.40	10.40	2.75	2.75	31.62	31.61

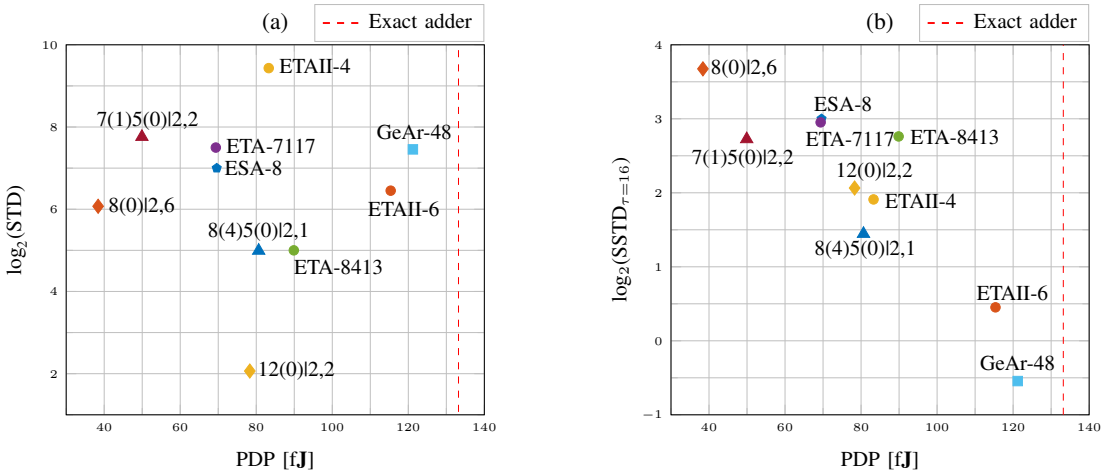

Fig. 4. Simulation results Comparison of 16-bit approximate adders: (a)STD vs. PDP, (b)SSTD applying threshold=2^4 vs. PDP. In order to keep the graphs readable, ETA-8413 denotes the configuration: 8(4)4(1)4(0)|0,0 ; and ETA-7117 is: 7(1)1(1)8(0)|0,0 .

an almost linear relationship between quality (probability of errors) and our saturated metric. The conventional metrics are misleading in some cases. For instance, in Fig. 3(b), the adder marked with $*$ is among the best configurations with the lowest probability of errors; but, conventional MSE cannot predict this and misleads to ignore the adder.

In order to evaluate the accuracy of the presented error formulas, the simulation and formulas results of 16-bit approximate adders, developed using our proposed template, are tabulated in Table III. The numbers presented in the table verify the perfect accuracy of the presented formulas for the generic template of approximate adders. As a result, using one compact accurate formula, the error behavior of a wide range of approximate adders can be modeled.

To assess the circuit characteristics and evaluate the approximate architectures, we have generated VHDL description of the proposed template. Different configurations of approximate adders ranging from existing to new hybrid structures are synthesized in a commercial low-power 65 nm library, for 16-bit operands. Using back-annotated simulations, dynamic power dissipation of the adders are evaluated after synthesis. All the adders have been simulated for 10^7 uniformly distributed random input patterns. In addition, we have developed a framework to distinguish the Pareto optimal architectures for a given accuracy and hardware quality, based on the presented formulas.

Fig. 4 illustrates a comparison of approximate adders generated by the proposed template. The compared adders are all optimal (area-delay efficient) architectures. In Fig. 4(a), STD of approximate adders versus their Power-Delay Product (PDP) are depicted. As can be seen in this figure, considering the trade-off between error and PDP of the adders, OLOCA architectures outperform all the other adders. Furthermore, hybrid architectures, developed by the proposed template, perform better than other existing approximate adders including ETAII, ESA and GeAr. Another notable fact in this figure is the superiority of non-equal segmented ETAIIs over the equal segmented ones. As discussed in the previous section, since the optimal adders are instantiated as sub-adders, non-equal segmentation boosts the efficiency of segmented adders.

On the other hand, Fig. 4(b) depicts the Saturated STD (SSTD) of each adder versus its PDP, applying a relatively small threshold. As can be seen in the figure, considering the trade-off between error and energy consumption of the adder architectures, the hybrid architectures developed using the proposed template outperform the rest of the adders. For the applications whose error can be modeled using SSTD with a small threshold, OLOCAs are not superior adders anymore, due to their high probability of errors. The superiority of adders depends on how stringent the threshold of the SSTD metric is. Our new error metric helps to understand qualitatively and to quantify numerically the intrinsic characteristics of the adders. Furthermore, Fig. 4 shows that the optimal solutions cannot be obtained using the standard and conventional methodologies

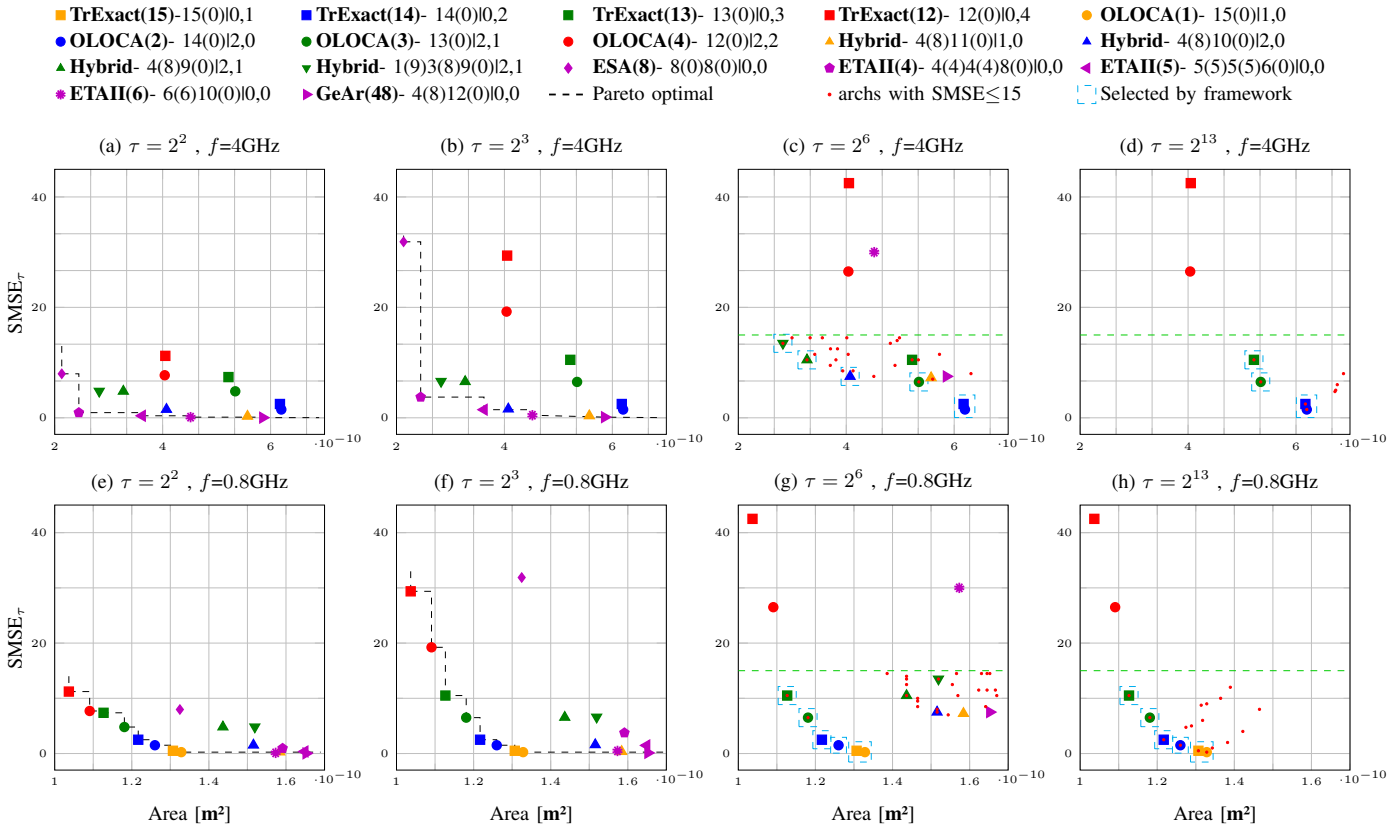

Fig. 5. Comparison of 16-bit adders using the proposed framework, applying two timing constraints and different thresholds. Saturated Mean Squared Error (SMSE) versus silicon area of adders: (a)-(d) Stringent timing constraint: frequency=4GHz ; (e)-(h) Relaxed timing constraint: frequency=800MHz. The final silicon area has been collected from synthesis. TrExact is the short form of truncated exact adder. The green dashed line is used to show the error limit applied, as an example, to our framework.

in most of the cases.

Finally, at the end of this section, we evaluate our framework. In an attempt to assess our framework, we progress in two steps: First, using our framework, we analyze various 16-bit approximate adders ranging from conventional approximate adders to new architectures developed from our template. In this step, we show the functionality of our framework, while comparing the adder architectures. Secondly, in order to show the exactness of our framework, we synthesize more architectures for a given error constraint. Here, the goal is to show that the selected architectures by our framework are actually the optimal architectures for a defined error limit.

Fig. 5 depicts a comparison of 16-bit adders developed using our template. The SMSE of the architectures for four different thresholds have been calculated using the error models. The applied thresholds range from small, stringent thresholds to the one which is large enough to represent conventional MSE. The silicon area of the architectures have been obtained synthesizing the selected architectures. Different colors in the figure correspond to different bit-width of the upper part adder. For example, the TrExact(13) is a truncated exact adder, in which the upper part adder is a 13-bit exact adder, and the lower part bits are constant-1s. As can be seen in Fig. 5(a)-(d), for relatively small timing constraints, when a stringent threshold applied, ETAII architectures are superior adders due to their low probability of errors. As the threshold increases, SMSE of ETAII architectures increase and hybrid architectures show superiority, as illustrated in Fig. 5(c). Continuing the threshold increment, the OLOCA adders tend to be superior architectures, as depicted in Fig. 5(d). As a result, qualitatively, the range of possible thresholds can be divided into three regions: small, medium and large thresholds. For small thresholds, conventional ETAIIs; for medium thresholds, hybrid architectures; and for large thresholds (conventional MSE), OLOCA adders are superior architectures.

The same comparison is shown in Fig. 5(e)-(h), carried out with relaxed timing constraint corresponding to area-efficient implementation of the architectures. As can be seen in the graphs, regardless of the threshold, there is no reason to use conventional approximate adders in this scenario. With relaxed timing constraints, for all the thresholds, truncated exact adders or OLOCA adders are superior architectures. Accordingly, the framework discriminates this scenario, where approximate adders do not provide considerable benefits from the scenarios which they do.

To assess the exactness of our framework, in this step of the experimental validation, we consider an illustrative error limit and analyze that case in detail. Then, we generate all the possible architectures which meet the defined constraint. As an example, we consider a practical value for the error constraint (SMSE$_\tau \leq 15$). The architectures which meet the above-mentioned constraint are shown with small red dots

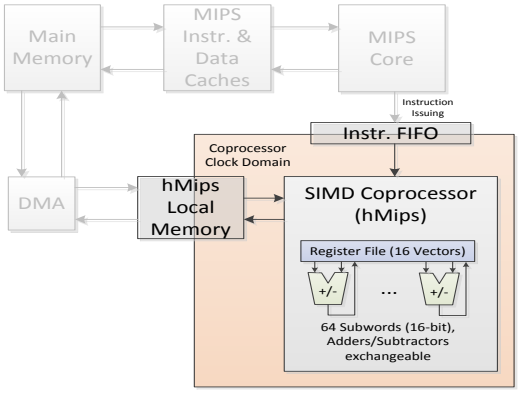

Fig. 6. SIMD coprocessor framework. Components that are not considered in the case study are grayed out.

in Fig. 5. Additionally, the error limit is marked with green dashed lines in the figure. The number of architectures that meet the constraint increases, as the threshold is decreased. Because of the fact that there are many architectures that meet the constraint for small thresholds, in order to keep the graphs readable, we show the example for larger thresholds: τ=64 and τ=8192. In addition, we limit the architectures for the case of τ=64, and we consider only the architectures where the bit-width of their upper-part adders are 13-bit, as an illustration.

As can be seen in the figure, the architectures which have been chosen by our framework (marks with light blue border), among all the possible architectures (red dots), are in fact the optimal ones. In short, Fig. 5 not only shows the functionality of our framework, but also depicts how, in different scenarios, different approximate adders are superior. A notably important message of Fig. 5 is that not always using approximate adders results in a better design. It should be discriminated where approximate adders provide significant benefits from where they do not. As mentioned, our framework is able to distinguish these scenarios.

VI. Case Study

In the following section, a case study using different approximate adders in the ALU of an SIMD coprocessor for a MIPS CPU is presented. The goal of the case study is to assess the applicability of the proposed design framework in more complex scenarios and not to evaluate the gains achieved using the approximate adders. Clearly, the gains are not considerable here since we are changing only one adder in the ALU of a SIMD processor. The performance enhancement as well as circuit area and power consumption reduction are evaluated, while executing a Sobel image filtering application. The Sobel filtering has been chosen as a relatively simple application which uses addition in its algorithm. In addition, Sobel algorithm is one of the typical applications used for approximate computing bechmarking listed in [24].

A. Processor: Approximate SIMD Coprocessor

The underlying processor substructure for this case study is depicted in Fig. 6. To decouple the performance of the SIMD coprocessor from the main MIPS CPU, the coprocessor resides in an own clock domain. The main MIPS CPU is used for handling the application control flow, issuing vector instructions over a clock-domain-crossing FIFO and initiating DMA transfers between the system main memory and the coprocessor's local memory, which also crosses the clock domain boundary.

For a meaningful evaluation, the coprocessor contains only the necessary hardware resources required by the executed Sobel filtering application, i.e., a register file containing 16 registers with 64 16-bit element subwords each. Furthermore, each subword ALU is reduced to an exchangeable and configurable approximate adder/subtractor and fixed element subword permutation multiplexers required to calculate the Sobel matrix convolution. The critical path and thus the performance limitation is located in the approximate adder/subtractor unit.

B. Evaluation Flow

For evaluation, the coprocessor ALUs are implemented with different approximate adder entities, including those identified by our framework. The SIMD coprocessor is synthesized to ASIC gate-level netlists for coprocessor clock periods between 3.4 and 4.9 ns, using a TSMC 40 nm low-power standard cell library. After the circuit area is obtained from synthesis, gate-level netlist-based timing simulations using annotated parasitics are performed to obtain the switching activity in the coprocessor for power analysis.

The remaining main MIPS CPU system is simulated functionally and clocked fast enough to ensure that the coprocessor's performance is not limited by the instruction issuing. Moreover, all necessary DMA image transfers are performed prior to all switching activity and performance measurements. Therefore, the pure computational performance of the coprocessor is considered, being only a function of the clock period. For the Sobel filtering application, the performance ranges between 275 µs per 512×512 pixels grayscale image frame for a clock

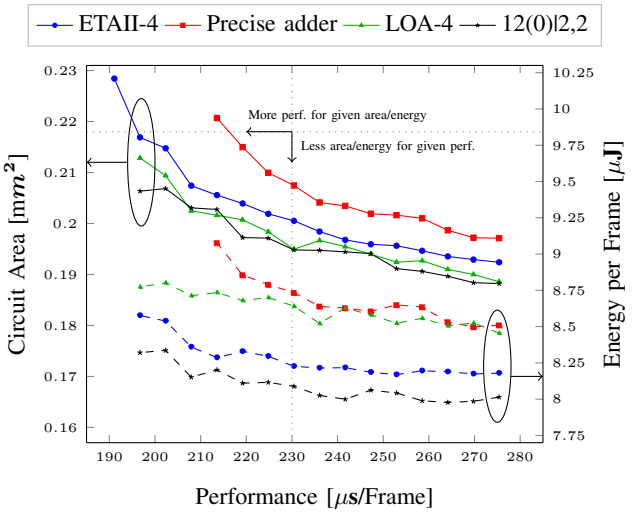

Fig. 7. Comparison of performance, circuit area and energy per frame of the SIMD coprocessor equipped with different approximate adder entities and parameterizations. The solid lines denote the circuit area, the dashed lines denote the energy per frame.

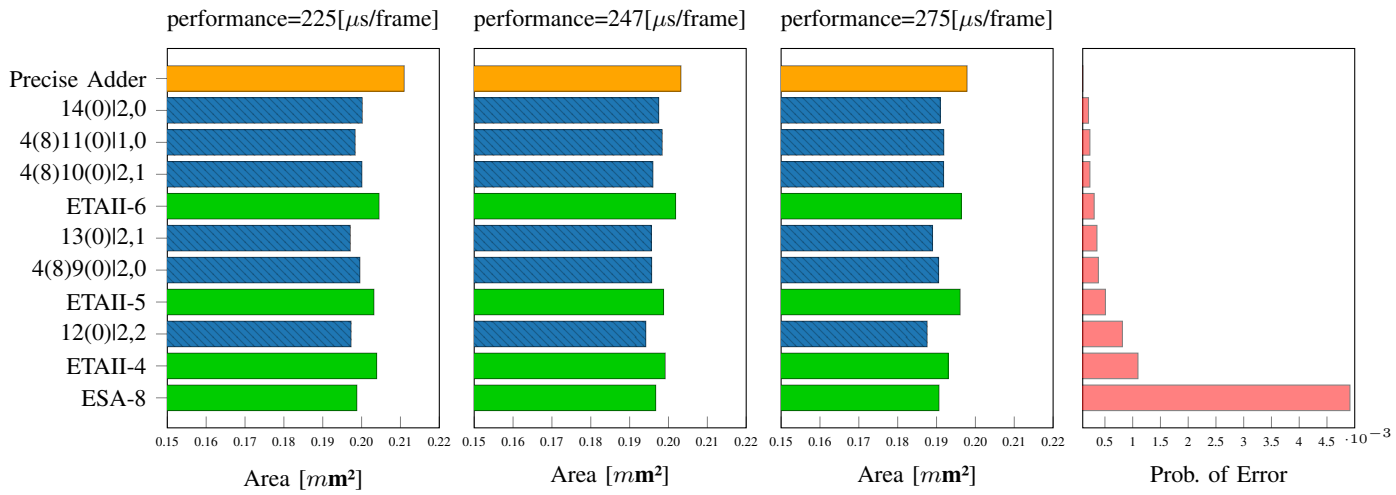

Fig. 8. Comparison of accuracy, and circuit area of the SIMD coprocessor implemented with different approximate adder entities and parameterizations. The circuit area of the coprocessor are compared for different adder architecture for three different frequencies: the yellow bar is the precise adder, the green bars are conventional approximate adders, and the hatched blue bars are the hybrid architectures selected by the framework.

period of 4.9 ns and 190 µs per frame for 3.4 ns. The power consumption of the coprocessor is converted to an energy budget per image frame using these performance values.

C. Evaluation Results

In order to evaluate the adder architectures inside the coprocessor, first we compare a precise adder, an OLOCA architecture [17], and two conventional approximate adders for all the frequencies. This way, the figure is readable because of choosing limited number of adder architectures. In a separate figure, using bar graphs, we compare more adder architectures for their error values as well as their silicon area for three different frequencies.

Fig. 7 compares the total coprocessor circuit area and total energy per frame as a function of the performance for selected adder configurations. This figure compares a precise adder, a selected adder from our framework, and two classical approximate adders for the performance ranges between 275 µs per frame and 190 µs per frame. The arrows indicate how to find area- and energy-efficient configurations for a desired performance value or more performant implementations for given area and energy budget constraints. It can be seen in the figure that the OLOCA architecture developed using our template outperforms the existing approximate architectures, from both circuit area and energy consumption points of view.

The reference precise adder configuration reaches a maximum performance of 215 µs per frame. By inserting approximate adders, the performance can be boosted by up to 12%. At a fixed performance, the evaluated approximate adders can reduce the coprocessor circuit area by up to 10% and the energy dissipation by up to 15%. Given the fact that the precise ALU occupies 12% of the total coprocessor area and consumes 7% of the total power, the significant reduction in hardware cost becomes clear. Especially, the energy dissipation in non-ALU parts of the coprocessor is reduced as well.

Fig. 8 depicts the comparison of the total coprocessor circuit area for three different frequencies as well as the probability of the error of adder architectures. More adder architectures are compared in this figure: the precise adder, conventional approximate adders and hybrid adders selected by the framework. The probability of the error of the adders are shown for a chosen binarization which corresponds to the threshold ($\tau = 16$) of our new metric. As can be seen in the figure, considering the error-area trade-off for all the three frequencies, the selected architectures by the framework outperform the conventional approximate architectures for the selected application.

VII. Conclusion

In this paper, a generic template for approximate adders combining the *small-errors* and the *infrequent-errors* philosophies has been proposed. A wide range of approximate adders, along with new hybrid and non-equal segmented adders, can be developed using the proposed template. The accurate mathematical error formulas of the template has been conjointly presented. Consequently, using one compact formula, the error metrics of a wide range of approximate adders can be calculated. In addition, a new error metric has been introduced in this paper to address the insufficiency of the existing metrics. The new parameterizable metric captures the requirements of the stochastic applications.

Using experimental results, it has been shown that for the scenarios that approximate adders provide considerable benefits, our framework finds the optimal architectures. It has been shown that applying different thresholds (τ) change the superiority of the adders. As a result, the parameter τ can be considered as a knob in our new metric to model more accurately errors in real applications. In conclusion, for relaxed timing constraints, there is no need to use conventional approximate adders, because they do not provide considerable benefits. For stringent timing constraints, OLOCA, hybrid, and ETAII architectures are superior classes of adder architectures for large, medium and small thresholds, respectively.

REFERENCES

[1] J. Sartori and R. Kumar, "Stochastic computing," *Found. Trends Electron. Des. Autom.*, vol. 5, no. 3, pp. 153–210, Mar. 2011.

[2] A. Ghofrani, A. Rahimi, M. A. Lastras-Montaño, L. Benini, R. K. Gupta, and K. T. Cheng, "Associative memristive memory for approximate computing in gpus," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 6, no. 2, pp. 222–234, June 2016.

[3] S. Mittal, "A survey of techniques for approximate computing," *ACM Comput. Surv.*, vol. 48, no. 4, pp. 62:1–62:33, Mar. 2016.

[4] A. B. Kahng and S. Kang, "Accuracy-configurable adder for approximate arithmetic designs," in *Proceedings of the 49th Annual Design Automation Conference*, ser. DAC '12. New York, NY, USA: ACM, 2012, pp. 820–825.

[5] D. Mohapatra, V. K. Chippa, A. Raghunathan, and K. Roy, "Design of voltage-scalable meta-functions for approximate computing," in *2011 Design, Automation Test in Europe*, March 2011, pp. 1–6.

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

[7] K. Du, P. Varman, and K. Mohanram, "High performance reliable variable latency carry select addition," in *2012 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2012, pp. 1257–1262.

[8] I. C. Lin, Y. M. Yang, and C. C. Lin, "High-performance low-power carry speculative addition with variable latency," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 9, pp. 1591–1603, Sept 2015.

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

[10] H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie, and C. Lucas, "Bio-inspired imprecise computational blocks for efficient VLSI implementation of soft-computing applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 4, pp. 850–862, April 2010.

[11] 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, no. 1, pp. 124–137, Jan 2013.

[12] A. K. Verma, P. Brisk, and P. Ienne, "Variable latency speculative addition: A new paradigm for arithmetic circuit design," in *Proceedings of the Conference on Design, Automation and Test in Europe*, ser. DATE '08, 2008, pp. 1250–1255.

[13] S.-L. Lu, "Speeding up processing with approximation circuits," *Computer*, vol. 37, no. 3, pp. 67–73, Mar 2004.

[14] D. Esposito, D. D. Caro, and A. G. M. Strollo, "Variable latency speculative parallel prefix adders for unsigned and signed operands," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 8, pp. 1200–1209, Aug 2016.

[15] H. Jiang, J. Han, and F. Lombardi, "A comparative review and evaluation of approximate adders," in *Proceedings of the 25th Edition on Great Lakes Symposium on VLSI*, ser. GLSVLSI '15. New York, NY, USA: ACM, 2015, pp. 343–348.

[16] A. Najafi, M. Weißbrich, G. P. Vayá, and A. Garcia-Ortiz, "A fair comparison of adders in stochastic regime," in *2017 27th International Symposium on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, Sept 2017, pp. 1–6.

[17] A. Dalloo, A. Najafi, and A. Garcia-Ortiz, "Systematic design of an approximate adder: The optimized lower part constant-or adder," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pp. 1–5, 2018.

[18] J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders," *IEEE Transactions on Computers*, vol. 62, no. 9, pp. 1760–1771, Sept 2013.

[19] C. Liu, J. Han, and F. Lombardi, "An analytical framework for evaluating the error characteristics of approximate adders," *IEEE Transactions on Computers*, vol. 64, no. 5, pp. 1268–1281, May 2015.

[20] H. Jiang, J. Han, F. Qiao, and F. Lombardi, "Approximate radix-8 booth multipliers for low-power and high-performance operation," *IEEE Transactions on Computers*, vol. 65, no. 8, pp. 2638–2644, Aug 2016.

[21] J. Schlachter, V. Camus, and C. Enz, "Near/sub-threshold circuits and approximate computing: The perfect combination for ultra-low-power systems," in *2015 IEEE Computer Society Annual Symposium on VLSI*, July 2015, pp. 476–480.

[22] J. Schlachter, V. Camus, K. V. Palem, and C. Enz, "Design and applications of approximate circuits by gate-level pruning," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 5, pp. 1694–1702, May 2017.

[23] R. Hegde and N. R. Shanbhag, "Energy-efficient signal processing via algorithmic noise-tolerance," in *Proceedings. 1999 International Symposium on Low Power Electronics and Design (Cat. No.99TH8477)*, Aug 1999, pp. 30–35.

[24] A. Yazdanbakhsh, D. Mahajan, H. Esmaeilzadeh, and P. Lotfi-Kamran, "Axbench: A multiplatform benchmark suite for approximate computing," *IEEE Design Test*, vol. 34, no. 2, pp. 60–68, April 2017.

Ardalan Najafi received his M.Sc. degree in Electronics Engineering from Shahid Beheshti University of Tehran, Iran, in 2013. He is currently working towards the Ph.D. degree in the Institute of Electrodynamics and Microelectronics at the University of Bremen, Germany. His research interests focus mainly on low-power computing units using stochastic and sub-threshold approaches.

Moritz Weißbrich received his M.Sc. degree in electrical engineering from Leibniz Universität Hannover, Germany, in 2016. Mr. Weißbrich is currently working as a research engineer towards the Ph.D. degree at the Institute of Microelectronic Systems, Leibniz Universität Hannover. His research interests include high-performance and low-energy processor architectures using approximate arithmetic and stochastic computing approaches.

Guillermo Payá-Vayá obtained his Ing. degree from the School of Telecommunications Engineering, Universidad Politécnica de Valencia, Spain, in 2001. During 2001-2004, he was a member of the research group of Digital System Design, Universidad Politécnica de Valencia, where he worked on VLSI dedicated architecture design of signal and image processing algorithms using pipelining, retiming, and parallel processing techniques. In 2004, he joined the Department of Architecture and Systems at the Institute of Microelectronic Systems, Leibniz Universität Hannover, Germany, and received a Ph.D. degree in 2011. He is currently Junior Professor with the Institute of Microelectronic Systems, Leibniz Universität Hannover, Germany.

Alberto Garcia-Ortiz Alberto Garcia-Oritz obtained the diploma degree in Telecommunication Systems from the Polytechnic University of Valencia (Spain) in 1998. After working for two years at Newlogic in Austria, he started the Ph.D. at the Institute of Microelectronic Systems, Darmstadt University of Technology, Germany. In 2003, he received his Ph.D. degree with "summa cum laude". From 2003 to 2005, he worked as a Senior Hardware Design Engineer at IBM Deutschland Development and Research in Böblingen. After that he joint the start-up AnaFocus (Spain), where he was responsible for the design and integration of AnaFocus' next generation Vision Systems-on-Chip. He is currently full professor for the chair of integrated digital systems at the university of Bremen.

Dr. Garcia-Ortiz received the "Outstanding dissertation award" in 2004 from the European Design and Automation Association. In 2005, he received from IBM an innovation award for contributions to leakage estimation. Two patents are issued with that work. He serves as editor of JOLPE and is reviewer of several conferences, journals, and European projects.