Designing Efficient Imprecise Adders using Multi-bit Approximate Building Blocks

Sarvenaz Tajasob, Morteza Rezaalipour, Masoud Dehyadegari
K.N. Toosi University of Technology
{s.tajasob,mrezaalipour}@email.kntu.ac.ir
dehyadegari@kntu.ac.ir

Mahdi Nazm Bojnordi
School of Computing
University of Utah
Salt Lake City
bojnordi@cs.utah.edu

ABSTRACT

Energy-efficiency has become a major concern in designing computer systems. One of the most promising solutions to enhance power and energy-efficiency in error tolerant applications is approximate computing that balances accuracy, area, delay, and power consumption based on the computational needs. By trading accuracy of computation, approximate computing may achieve significant improvements in speed, power, and area consumption.

Adders are important arithmetic units widely used in almost every digital processing system, which contribute to significant amounts of power dissipation. With the emergence of deep learning tasks and fault tolerant *big data* processing in every aspect of today's computing, the demand for low-power and energy-efficient approximate adders has increased significantly. Numerous designs have been proposed in the literature that build multi-bit adders using novel approximate full adder circuits. Regrettably, relying on single-bit building blocks only limits the design space of approximate adders and prevents the designers from achieving the most significant benefits of approximate circuits. This paper presents a novel approach to designing imprecise multi-bit adders, based on four novel approximate 2 and 3-bit adder building blocks. The proposed circuits are evaluated and compared with the existing low power adders in terms of various design characteristics, such as area, delay, power, and error tolerance. Our simulation results indicate that the proposed adders achieve more than 60% reduction in power and area consumption compared to the state-of-the-art approximate adders while introducing 12-17% less error in computation.

KEYWORDS

Approximate computing, imprecise adders, multi-bit approximate blocks.

ACM Reference Format:
Sarvenaz Tajasob, Morteza Rezaalipour, Masoud Dehyadegari and Mahdi Nazm Bojnordi. 2018. Designing Efficient Imprecise Adders using Multi-bit Approximate Building Blocks. In *ISLPED '18: ISLPED '18: International Symposium on Low Power Electronics and Design, July 23–25, 2018, Seattle, WA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3218603.3218638

1 INTRODUCTION

As the CMOS feature size keeps shrinking to less than ten nanometers and more transistors are packed on chip, reducing power dissipation and improving performance becomes increasingly difficult [1, 2]. Moreover, the increasing growth in the number of embedded computer systems used for mobile Internet of Things (IoT) devices creates a huge market for low-power and energy-efficient architectures and circuits. Lowering power dissipation in mobile devices while powering multiple billions of such systems are deployed around the world becomes crucial to address the energy-efficiency challenges in today's computer systems [3–5].

Most embedded systems and mobile IoT applications include multimedia, digital signal processing (DSP), pattern recognition, or data mining. A key feature of such applications is their significant tolerance against computational errors and design faults. Due to the human's perceptual limitation and fault-tolerant nature of these applications, their outputs may not need to be fully precise in most cases. This leads to a great opportunity for a form of computation, called *approximate computing*, that allows certain amounts of errors appear in the results without impacting the subjective quality of applications. The design principle of approximate computing relies on relaxing the computational precision to save power and to enhance performance of VLSI circuits [1, 4, 5]. This new design paradigm, aims at optimizing performance, power, and circuit complexity via degrading the computational precision without noticeable impacts on the quality of results [6].

Adder plays an important role in performance and system energy of DSP applications. The goal of approximate computing is to design an imprecise circuit that leverages its inexactness to reduce delay, die area, and power consumption [7]. Based on the input length, the design methodologies of approximate multi-bit adders may be classified into two categories: (1) single-bit full adders and (2) multi-bit approximate building blocks. Most existing proposals in the literature are based on the first approach to reducing the total number of transistors/gates in a multi-bit adder—e.g., ripple carry adder (RCA) and carry look-ahead adder (CLA). These proposals develop inexact versions of a conventional single-bit full adder to build an approximate multi-bit adder, while optimizing performance, circuit complexity, and power dissipation.

Unlike the existing techniques, this paper centers on the second approach to designing efficient imprecise multi-bit adders using three novel 2 and 3-bit approximate adder building blocks that provide better accuracy, area, delay, and power dissipation. To the best of author's knowledge, this paper is the first to provide such design methodology for building efficient approximate adders for digital signal processing systems. Our simulations indicate that the proposed 2 and 3-bit building blocks reduce the dynamic and static power respectively by 60% and 65% as compared to the best existing proposal for the lower-part-OR adder (LOA) architecture. Moreover, the mean error distances (MED) of the proposed adders are improved by 17% for 2-bit and 12% for 3-bit blocks as compared to the LOA adders and the area consumption is reduced by 60%.

2 BACKGROUND

Ever since approximate computing was seriously considered by many researchers as an effective technique to overcome the technology limitations, system designers have made numerous proposals to make the key functional units of processors faster and more energy-efficient at the cost of introducing acceptable amounts of errors in computation. In particular, a great number of proposals on approximate adders can be found in the literature. This section reviews several classes of the state-of-the-art approximate adders and briefly explains the commonly used metrics for error evaluation.

2.1 Approximate Adders

Gupta *et al.* propose five different approximate full adders with reduced logic complexity through a set of optimizations at the transistor level [8, 9]. The conventional mirror adder (MA) [10] is employed as the design framework for building the approximate mirror adders. A few serially connected transistors are removed from the MA structure to reduce the switching capacitance of gates and the critical path of the circuit, which leads to decreasing power dissipation and area consumption.

Almurib *et al.* propose a set of three inexact full adders designed for low-power approximate computing [11]. The key idea is to replace accurate adder designs with inexact adder (InXA) cells to decrease the critical path and switching capacitance of the gate outputs. A low-power and high-speed ripple carry adder (RCA) is designed such that the InXA cells are used to compute the least significant bits of a multi-bit adder. The InXA cells consists of fewer transistors than an exact full adder and perform better than the AMA designs with respect to power consumption, speed, mean relative error distance (MRED), and normalized mean error distance (NMED) in image processing applications.

Yang *et al.* propose two low-power designs for approximate adders based on multiplexer [12]. As the gate-level structures for the conventional CMOS multiplexers are rather complex, transmission gates are proposed to implement approximate adders with low power dissipation. The transmission gate-based adder (TGA) circuits are used for sharpening edges in image processing algorithms. In a different study, three low-power and area efficient approximate full adders are proposed that employ multiplexers and XOR/XNOR gates implemented with pass transistors [13]. The approximate XOR/XNOR-based Adder (AXA) designs reduce the transistor count in two previously proposed circuits on low-power and high-speed 10-transistor full adders [14, 15]. AXA achieves a lower power consumption, faster speed, and less area occupation as compared with the 10-transistor fully precise adders.

Mahdiani *et al.* propose an efficient multi-bit adder for soft computing applications [16]. A p-bit lower-part-OR adder (LOA) comprises two m-bit and n-bit sub-adders; where, $p = m + n$. The most significant part of the adder that comprises m bits, is implemented with a conventional precise full adder such as a ripple-carry adder (RCA) or a carry look-ahead adder (CLA). The least significant part comprises n bits and is computed using two-input OR gates that generate the sum values and an AND gate that generates a carry input for the most significant part. LOA provides the most efficient designs among all existing adders based on accuracy, power dissipation, delay, and area [7].

Another class of approximate adders focus on circuits with the capability of configuring the accuracy of multi-bit addition at runtime. GEAR is a generic reconfigurable approximate adder proposed by Shafique *et al.* [17]. Akbari *et al.* propose RAP-CLA, which is a reconfigurable approximate carry look-ahead adder [18]. Accurate adders may be over-provisioned for most application that need low precision computation. Therefore, several speculative approximate adders are proposed that predict the carry bits to reduce the power, area, and delay overheads. Hu *et al.* propose an approximate adder circuit that generates signals for predicting carry bits [19]. The main idea is to break the carry chain into multiple segments to reduce the delay of circuit. The design employs additional hardware for correcting possible errors that may occur at the sign bit position, thereby lowering the error rate.

2.2 Error Evaluation Metrics

Researchers have employed several metrics for evaluating the accuracy of approximation circuits, such as error distance (ED), mean error distance (MED), and error rate (ER). Error distance (ED) is defined as the absolute difference between an approximate value a and its expected exact value b. Mean error distance (MED) is the average of error distances computed over all output values of an approximate circuit. Notice that the error rate (ER) is the percentage of errors computed by $\frac{\varepsilon}{2^n}$ for an n-bit adder with k approximate bits; where, ε represents the number of erroneous outputs. MED has been considered as a representative metric to optimize n-bit adders with k least significant approximate bits [20].

In particular, MED is a suitable metric for comparing approximate adders of the same bit widths. We present Equation 1 to compute MED for approximate adders; where, i represents the approximation length and C is a constant less than 1 specifically computed for every approximate adder. MED is a convenient metric for representing accuracy of large approximate adders with more than 5 approximated bit positions [20].

$$MED_i = 2 \times MED_{i-1} + C \tag{1}$$

3 MULTI-BIT APPROXIMATE BUILDING BLOCKS

This section presents a new family of low-power disjoint adders (i.e., without carry chains) that rely on novel k-bit approximate building blocks (k-AxB). We explain how an n-bit approximate adder using a k-AxB blocks (with $k < n$) exhibits more accuracy, better area consumption, and more energy-efficiency in comparison with the existing designs. Four design cases based on 2-AxB and

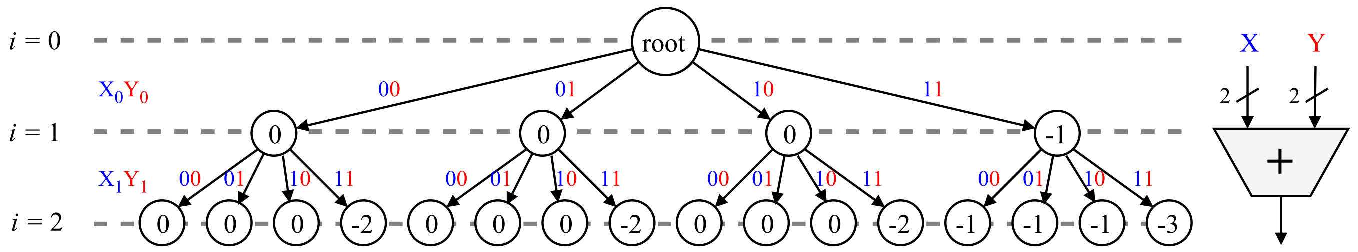

Figure 1: Illustrative example of error distribution for a 2-bit disjoint LOA architecture.

3-AxB blocks are explained and used for building efficient adder architectures ranging from 12 to 60 bits. We first provide an analysis of the design space before explaining the proposed architectures.

3.1 Exploring the Design Space

Figure 1 shows the error distribution of an example 2-bit LOA architecture using a complete quad-tree, where the adder width (denoted by i) is equal to the tree depth. Starting from the root, every node corresponds to a bit position of the input operands and is connected to the lower level nodes through four possible branches, each of which represents a 2-bit value selected from the same bit positions of the input operands. Therefore, every path from the root to a leaf represents the possible bit patterns for the two input operands in a particular n-bit addition. This design space grows exponentially with respect to the number of computed bits (2^{2n}).

3.1.1 *Error Representation.*

The proposed quad-tree representation is used for error evaluation. Every node at depth i contains the error distance of the bit positions 0 to $i-1$ in the output. Leaves are then used to represent the error distances encountered for a given pair of n-bit input operands, which makes the tree suitable for analyzing the precision of approximate adders. For example, error probability and the mean error distance metrics are proportional to the total number of non-zero leaves and sum of their absolute value, respectively. Therefore, adders exhibiting identical behaviors (with the exact same values and order) at the leaves provide equal levels of computational accuracy. We employ the proposed quad-tree representation to explore the design space of multi-bit adders using the proposed and existing baseline architectures. To the best of our knowledge, this paper is the first to examine this proposed tree-based methodology for designing efficient approximate adders.

3.1.2 *Approximate Adder Design.*

The proposed quad-tree is used to represents the design of approximate adders with a variety of precisions. For example, Figure 2 shows the error distribution of a single-bit adder using a sequence of four binary values (E_0-E_3). Therefore, a total of 2^4 error sequences are possible, each of which corresponds to an approximate single-bit adder. For an n-bit adder, $(2^{n+1})^{2n}$ approximate designs with different levels of computational accuracy are possible, which makes any exhaustive search for logic optimization within the design space infeasible.[1]

[1]Notice that applying know logic design algorithms to the search becomes even more complex. For example, the computational complexity of the Quine McClusky algorithm [21] within this search space is $3 \times 2^{n2^n}$ for an n-bit approximate adder.

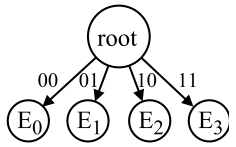

Figure 2: Quad-tree representation of a 1-bit adder.

3.2 Designing Adders with k-AxB

We propose to design every n-bit adder using k-bit approximate building blocks (k-AxB). Every n-bit adder comprises $\frac{n-k}{k}$ disjoint k-AxBs to perform the n-bit addition by (1) calculating $n-k$ least significant bits of the output using $\frac{n-k}{k}$ disjoint k-AxBs and (2) computing the remaining $k+1$ bits of the output using another k-bit approximate adder capable of producing the carry out signal. Figure 3 illustrates the necessary building blocks for computing an n-bit addition using the proposed k-AxBs. This modular organization is general enough to represent a variety of existing approximate adders. For example, an n-bit LOA calculates the $n-1$ least significant bits of the output using $n-1$ OR gates, while the n^{th} and $(n+1)^{th}$ bits of the output are computed using OR and AND gates.

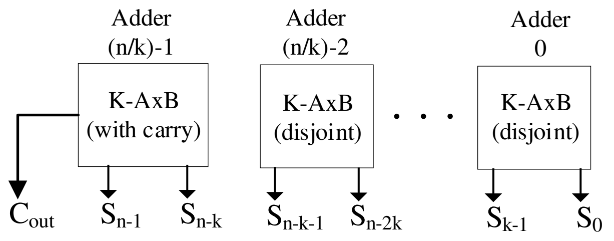

Figure 3: The design of an n-bit approximate adder using *k*-AxB.

As mentioned in Section 3.1, the design space of multi-bit approximate adders is intractable, thereby making a search for the optimum design point computationally hard and impractical. Therefore, heuristic techniques are necessary to examine a fraction of the design space towards finding an optimized approximate adder. Notice that a design with improved power efficiencies and reduced accuracy losses is almost always preferred by the system designers [20]. LOA is one of the most power efficient approximate adders in the literature due to effectively trading the computational accuracy for power reduction [7]. This paper consider the LOA architecture as a baseline for comparisons and limits the search to only those design points that provide better energy, delay, and accuracy than the LOA baseline. Therefore, we define the following conditions for every search step: (1) every k-AxB must achieve less power dissipation and area consumption compared to a k-bit LOA circuit; and (2) the resulting n-bit adders (for $n > 5$) must produce an MED value less than that of an n-bit LOA circuit. Note that holding these conditions are necessary to design a more efficient adder than any given baseline architecture.[2]

Next, we explain a two-step design approach for building an n-bit approximate adder comprising disjoint 2-AxBs. To satisfy the first constraint, the simplest 2-bit adder with the lowest possible power dissipation, area, and delay is selected as the baseline that consists of no logical gates. Figure 4(a) shows the design space of this baseline approximate adder using a two level quad-tree. The adder always outputs 00 for any combination of the input operands; therefore, its error distribution tree comprises 16 leaves. The amount of error per each leaf can be 0, 1, 2, or 3, which results in 4^{16} different 2-bit approximate adder.

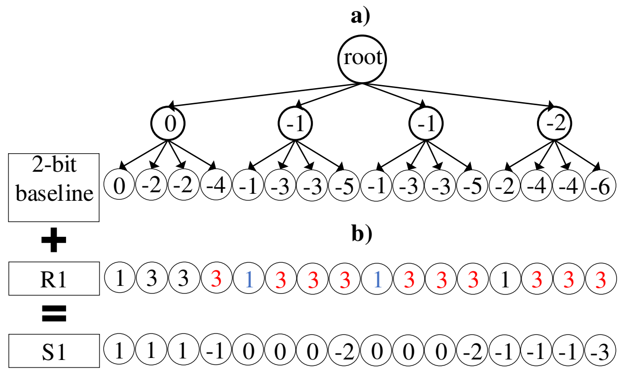

Figure 4: The design space and improved error distribution of the proposed 2-AxB.

Step 1. First we generate a new sequence of errors (i.e., S1 in Figure 4) through defining and adding an intermediate sequence R1 to improve the accuracy of the results. Applying this change in the error values corresponds to a gate-level modification in the baseline design. We set the corresponding R1 values of those leaves with $E \leqslant -3$ to 3 (shown in red in figure 4). This is equivalent to changing the baseline's output from 00 to 11. Similarly, for the -1 leaves in the baseline, their corresponding leaves is set to 1 (shown in blue in figure 4). This narrows down the search space to 4^4 different sequences; therefore, the values for the remaining four leaves are then found through exhaustive search. Sequence S1 represents the error distribution of a novel and improved 2-AxB called 2-AxB1. Figure 5 shows two gate level implementations for 2-AxB1 and 2-AxB2. Similarly, enhanced versions of a 3-AxB are produced and shown as 3-AxB1 and 3-AxB2 in Figure 6.

Step 2. The top sequence in Figure 7 represents the error distribution of the 2-bit baseline adder.[3] Here we find two new sequences and add them to the baseline leaves. The first sequence is R0 that presents the changes resulted by carry out generation; therefore, its

[2]Here, we explain the proposed methodology; while, designing algorithms and heuristics towards finding the most optimum design point is left for future work.
[3]To save space, the figure does not show the whole design space of the baseline.

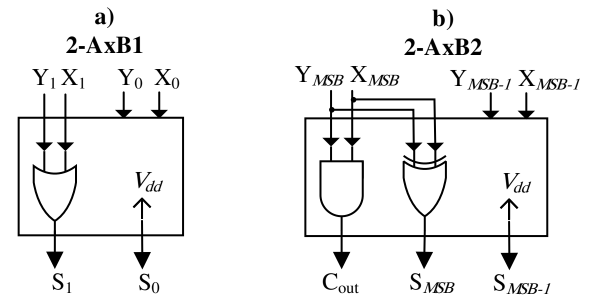

Figure 5: Gate level implementation of 2-AxB1 and 2-AxB2.

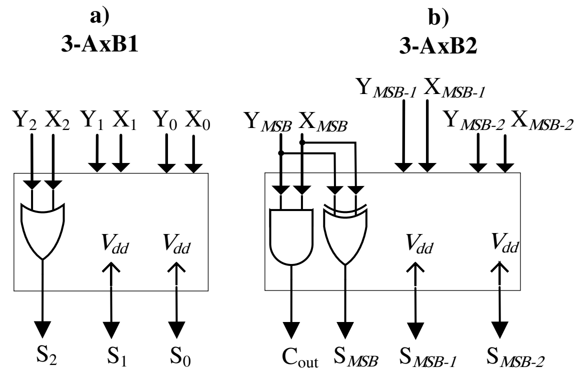

Figure 6: Gate level implementation of 3-AxB1 and 3-AxB2.

leaves can be either 0 or 4. The second sequence is R2 that contains the changes required for the least significant bits of the output. To reduce the large errors of the baseline, As shown in figure 7 in green, we set four leaves of R0 to 4 (which means that the baseline should generate a carry out signal for these cases). To further reduce the errors, for three of these four leaves, their corresponding values in R2 are set to 1 (shown in green in figure 7). Among the remaining leaves of R2, those correspond to -3 and -4 errors are set to 3 (shown in red in figure 7). Finally, the leaves with -1 are set to 1 (shown in blue in figure 7). The values for the remaining five leaves are then found through exhaustive search. Sequence S2 represents the error distribution of another novel and improved 2-AxB called 2-AxB2.

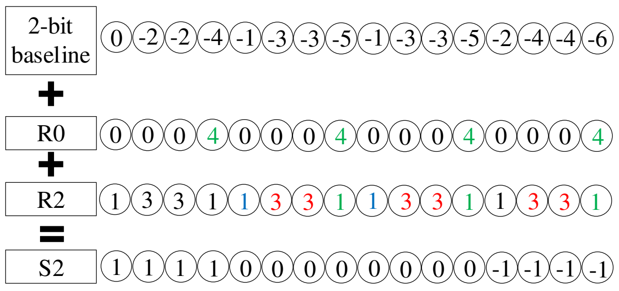

Figure 7: Error distribution of the proposed 2-AxB2.

4 EVALUATION

To assess the benefits of enlarging the design space for disjoint approximate adders, this section evaluates the proposed designs regarding both error and circuit characteristics. For the error evaluation of adders with approximation widths equal or less than 16, we exhaustively perform simulations for the entire set of inputs through accurately circuit modeling and software programming. As the search in larger domains may become impractical, we limit the number of simulations for adders with more than 16 approximate bits to a subset of inputs with 1 million randomly selected samples. Circuit characteristics are evaluated using VHDL description and synthesis in Synopsys Design Compiler with a TSMC180 CMOS library [22]. We use the LOA architectures as baselines for all comparisons.

4.1 Error Characteristics

Figure 8 shows the structure of *n*-bit adders using the proposed building blocks. Figures 9 and 10 compare the MED values computed for the 16- and 24-bit adders comprising 2 and 3-AxBs with those obtained for LOA. (Notice that the 16-bit adders are used for evaluating the approximation widths equal or less than 16 and the 24-bit adders are used for more than 16-bit approximation widths.) As can be seen in Figure 9, MED of 6 to 24-bit adders comprising 2-AxBs, is 17% less than LOA with an equivalent approximation width. Figure 10 shows that MED of 6 to 24-bit adders constructed with 3-AxBs, is 12% less than 6 to 24-bit LOA adders. MED increases exponentially regarding the number of approximate bits [20]. Thus the percentage of MED improvements is almost independent of the adder size. The MED improvements shown in Figures 9 and 10 are likely to occur for larger adders with 12 or more bits. Therefore, the proposed approximate adders exhibit better performance than the LOA baselines with respect to the MED metric.

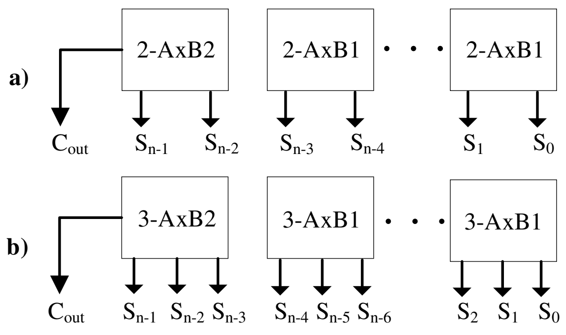

Figure 8: Constructing two n-bit approximate adders comprising proposed 2 and 3-AxBs.

4.2 Circuit Characteristics

Figures 11 and 12 compare dynamic and static power values for several multi-bit adders comprising 2 and 3-AxBs with LOA. It should be noted that the size of adders are chosen in a way that are divisible by both 2 and 3. According to Figure 11, dynamic power consumption in multi-bit adders comprising 2-AxBs is averagely 38% and up to 44% less than the power consumption of LOA

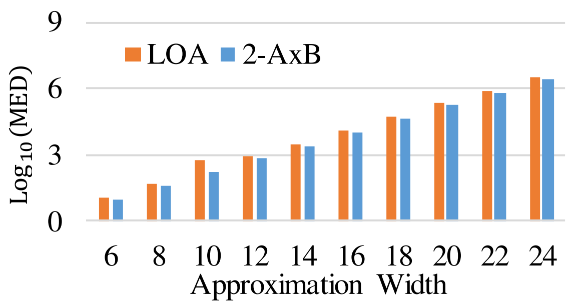

Figure 9: The MED values computed for the adders designed with the proposed 2-AxBs and LOA.

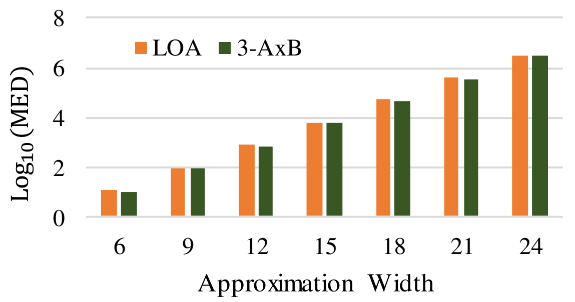

Figure 10: The MED values computed for the adders designed with the proposed 3-AxBs and LOA.

considering those bit widths. Adders comprising 3-AxBs consume averagely 54% and up to 60% less power than LOA considering the bit widths determined in the figures. As shown in Figure 12, static power is averagely 46% and up to 48% decreased in adders comprised of 2-AxBs. Also, the amount of static power reduction in adders comprised of 3-AxBs is averagely 62% and up to 65%.

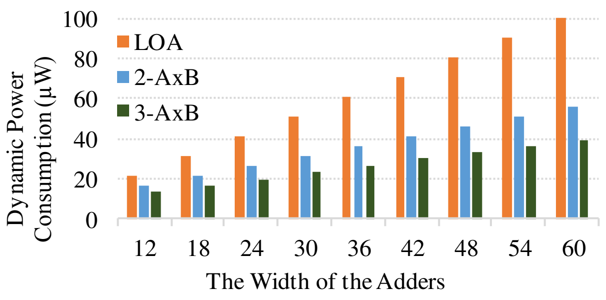

Figure 11: Comparing dynamic power consumption of several multi-bit adders comprising 2 and 3-AxBs with LOA.

Figure 13 shows the amount of area occupied by the proposed adders and LOA. Area of largest adder comprised 2-AxBs, is 45% less than LOA with its equivalent length. Area of largest adder constructed using 3-AxBs, is 60% less than LOA. Delay for all the examined adders in every bit width remains constant due to breaking the carry propagation chain and is equal to 0.13*ns*. Figure 14 is a radar chart comparing an optimized 12-bit LOA architecture

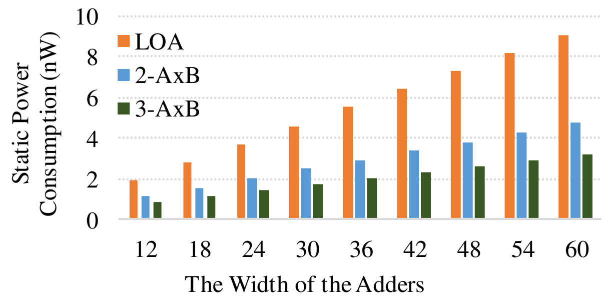

Figure 12: Comparing static power consumption of several multi-bit adders comprising 2 and 3-AxBs with LOA.

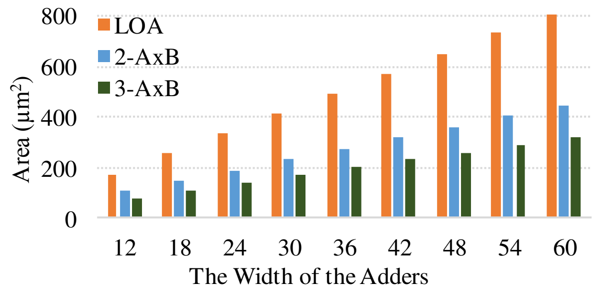

Figure 13: Comparing the area of several multi-bit adders comprising 2 and 3-AxBs with LOA.

with two 12-bit approximate adders using the proposed 2- and 3-AxB units. The proposed adders achieve better accuracy in terms of MED values and consume lower area and power consumption. We also observe that the 3-AxB design provides better area and power consumption compared to the 2-AxB adder, while it provides a slightly lower computational accuracy. All of the examined 12-bit adders in this chart are optimized to achieve a fixed computational delay.

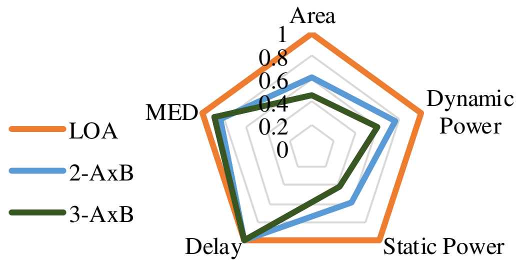

Figure 14: Comparing error and circuit characteristics of two 12-bit adders comprising 2 and 3-AxBs with LOA.

5 CONCLUSION

In this paper examined a new approach for designing multi-bit approximate low power adders. Two *k*-bit approximate building blocks were designed and evaluated to demonstrate the benefit of the proposed approach. Simulation results indicated that significant energy, delay, and area benefits as well as better computational precision are attainable using the proposed design approach. We believe that designing multi-bit approximate adder using *k*-AxBs provides more opportunities for optimization than only relying on single-bit full adders; This paper explored a fraction of this new design space and opens up opportunities for further research in this area.

6 ACKNOWLEDGMENT

The authors would like to thank anonymous reviewers for useful feedback.

REFERENCES

[1] D. Esposito, D. De Caro, E. Napoli, N. Petra, and A. G. Strollo, "On the use of approximate adders in carry-save multiplier-accumulators," in *Circuits and Systems (ISCAS), 2017 IEEE International Symposium on.* IEEE, 2017, pp. 1–4.
[2] 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, 2015.
[3] M. Gao, Q. Wang, M. T. Arafin, Y. Lyu, and G. Qu, "Approximate computing for low power and security in the internet of things," *Computer*, vol. 50, no. 6, pp. 27–34, 2017.
[4] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," in *Test Symposium (ETS), 2013 18th IEEE European.* IEEE, 2013, pp. 1–6.
[5] 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 Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, no. 8, pp. 1225–1229, 2010.
[6] Q. Xu, T. Mytkowicz, and N. S. Kim, "Approximate computing: A survey," *IEEE Design & Test*, vol. 33, no. 1, pp. 8–22, 2016.
[7] 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.* ACM, 2015, pp. 343–348.
[8] V. Gupta, D. Mohapatra, S. P. Park, A. Raghunathan, and K. Roy, "Impact: imprecise adders for low-power approximate computing," in *Proceedings of the 17th IEEE/ACM international symposium on Low-power electronics and design.* IEEE Press, 2011, pp. 409–414.
[9] 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, 2013.
[10] J. M. Rabaey, A. P. Chandrakasan, and B. Nikolic, *Digital integrated circuits.* Prentice hall Englewood Cliffs, 2002, vol. 2.
[11] H. A. Almurib, T. N. Kumar, and F. Lombardi, "Inexact designs for approximate low power addition by cell replacement," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2016.* IEEE, 2016, pp. 660–665.
[12] Z. Yang, J. Han, and F. Lombardi, "Transmission gate-based approximate adders for inexact computing," in *Nanoscale Architectures (NANOARCH), 2015 IEEE/ACM International Symposium on.* IEEE, 2015, pp. 145–150.
[13] Z. Yang, A. Jain, J. Liang, J. Han, and F. Lombardi, "Approximate xor/xnor-based adders for inexact computing," in *Nanotechnology (IEEE-NANO), 2013 13th IEEE Conference on.* IEEE, 2013, pp. 690–693.
[14] H. A. Mahmoud and M. A. Bayoumi, "A 10-transistor low-power high-speed full adder cell," in *Circuits and Systems, 1999. ISCAS'99. Proceedings of the 1999 IEEE International Symposium on*, vol. 1. IEEE, 1999, pp. 43–46.
[15] J.-F. Lin, Y.-T. Hwang, M.-H. Sheu, and C.-C. Ho, "A novel high-speed and energy efficient 10-transistor full adder design," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 54, no. 5, pp. 1050–1059, 2007.
[16] 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, 2010.
[17] M. Shafique, W. Ahmad, R. Hafiz, and J. Henkel, "A low latency generic accuracy configurable adder," in *Design Automation Conference (DAC), 2015 52nd ACM/EDAC/IEEE.* IEEE, 2015, pp. 1–6.
[18] O. Akbari, M. Kamal, A. Afzali-Kusha, and M. Pedram, "Rap-cla: A reconfigurable approximate carry look-ahead adder," *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2017.
[19] J. Hu and W. Qian, "A new approximate adder with low relative error and correct sign calculation," in *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition.* EDA Consortium, 2015, pp. 1449–1454.
[20] 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, 2013.
[21] E. J. McCluskey, "Minimization of boolean functions," *Bell Labs Technical Journal*, vol. 35, no. 6, pp. 1417–1444, 1956.
[22] I. Artisan Components, "Tsmc0.18μm standard cell library."