A Majority-based Approximate Adder for FPGAs

Behnam Ghavami, Mahdi Sajedi, Mohsen Raji, Zhenman Fang, Lesley Shannon
Simon Fraser University, Burnaby, BC, Canada
Shahid Bahonar University of Kerman, Kerman, Iran
Shiraz University, Shiraz, Iran

***Abstract*—The most advanced ASIC-based approximate adders are focused on gate or transistor level approximating structures. However, due to architectural differences between ASIC and FPGA, comparable performance gains for FPGA-based approximate adders cannot be obtained using ASIC-based approximation ones. In this paper, we propose a method for designing a low-error approximate adder that effectively deploys the modern FPGA structure. We introduce an FPGA-based approximate adder, named as Majority Approximate Adder (MAA), with less error than the advanced approximate adders. MAA is constructed using an approximate part and an accurate one; i.e. the accurate part is based on a smaller carry-chain compared with the carry-chain of the corresponding accurate adder. In addition, approximate part is designed to use FPGA resources efficiently with a low mean error distance (MED). Experimental results based on Monte-Carlo simulation demonstrates that a 16-bit MAA has a 49.92% lower MED than the state of the art FPGA-based approximate adder. MAA also takes up less area and consumes less power than other FPGA-based approximate adders in the literature.**

Approximate computing, Approximate adder, FPGA, low error, low power, LUT, High speed

I. Introduction

Application-Specific Integrated Circuits (ASICs) are the highly performance-efficient platforms for the implementation of wide spectrum of real-time applications such as image processing, deep neural networks inference, and multimedia processing. However, the time-to-market, flexibility, and real-time reconfiguration of ASICs are their significant challenges. So, it is necessary to explore novel techniques for energy-efficient computing specialized for FPGA-based systems, in addition to traditional energy optimization techniques. In an emerging paradigm for error-tolerant applications such as deep neural networks and multimedia processing, approximate computing has gotten a lot of attention. With limited computational imperfections, approximate computing is viewed as a powerful way for reducing power consumption and enhancing the performance of digital systems. In this case, approximation computing reduces the accuracy of digital hardware in order to increase its area, power, and speed.

Due to the fact that approximate adders are an important functional component of most error-tolerant applications and the base of other arithmetic functions e.g, multiplier and division, approximate computing has attracted considerable research efforts. Therefore, approximation the adder may improves the performance and power of application. Approximate adders can generally be classified into two categories: Low-latency approximate adders and low power approximate adders. The n-bit adder is split into numerous r-bit sub-adders in low-latency approximate adders, which can be overlapped or not.

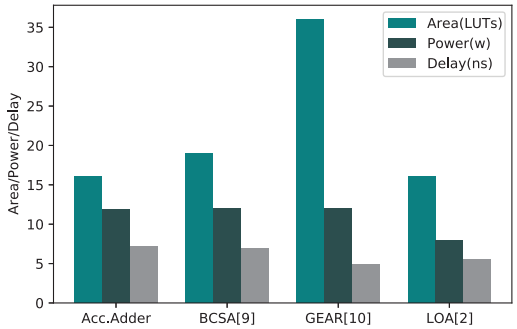

Fig. 1: FPGA Implementations of State-of-the-Art ASIC-based Approximate Adders

High speed is the prominent feature of these adders [1]. The full adders (FAs) are approximated and/or predicted carry utilizing only a few prior bits in low power approximate adders [2]. the approximation of the accurate FAs in this category can be at the transistor or gate level. Low energy and low area are the prominent features of these adders [3]. In a low-power approximate adder, the n-bit adder is constructed m-bit ($m < n$) least significant part (LSP) which is the approximate part, and most significant part (MSP) which is (n - m)-bit accurate adder.

It is noteworthy that advanced approximate adders are designed for ASIC implementation and cannot efficiently employ FPGA resources to increase output quality when implemented on FPGAs. This is primarily due to the differences in how ASIC and FPGA conduct logical processes. Lookup tables (LUTs) are used in FPGAs to implement logical functions, however logical gates are the basic building block of ASICs. As a result, ASIC-based designs cannot be mapped directly to FPGAs. We implemented some state-of-the-art ASIC-based approximate adder on Xilinx Virtex-7 FPGA using Vivado 15.4. tool-flow. Figure 1 depicts the area (LUTs), delay and power of these ASIC-based approximate adders when implemented on FPGA. [4].

FPGAs, on the other hand, are frequently utilized to construct error-tolerant applications in which approximate computing through basic operations, such as addition and multiplication, could be used for performance enhancement. Based on our knowledge, only a few FPGA-based approximate adders have been presented in the literature in which either has a high error rate or is hardware inefficient (section II-A2) [4]–[8]. As a result, developing an efficient approximation adder for FPGA-based designs is a critical research topic.

2771-2508/22/$31.00 ©2022 IEEE
DOI 10.1109/DSD57027.2022.00017

To address the aforementioned issues, we present the following contributions:

- *Design a Majority FPGA-Based Approximate Adder:* We propose a unique approximate adder architecture using all possible inputs of a LUT6 that approximates three bits of input with just one bit, which we call the Majority Approximate Adder (MAA3).
- *FPGA-Based Error Reduction (FER) Method:* suggest a method for reducing approximate adder error by making efficient use of FPGA resources, such as unused LUT inputs.

The remainder of this paper is structured as follows. Section II gives an overview of prior approximate adders as well as the background. The introduced approximate adder is presented in section III. The implementation details and experimental results are shown in Section IV. Finally, Section V concludes the paper.

II. BACKGROUND

A. Previous and Related works

The design of approximate adders encompasses a wide spectrum of approximate computing research activities. We reviewed the advanced literature in terms of ASIC and FPGA-based design in this section.

1) ASIC-based design: The majority of approximation adder design is centered on ASIC-based approximate adders. Low-latency approximation adders, such as [1], [9], and [10], are ASIC-based approximate adders that are built by truncating the carry chain. Despite the fact that these techniques improve speed, area, and power consumption, they might result in high error values and MED [11].

Some other ASIC-based approximate adders are focused on designing low-power approximate adder [2]. Lower-part-OR adder (LOA) [2] is a low-consumption approximate adder composed of two parts, the LSP and the MSP. Although the MSP is precise, the LSP is made up of two-input OR gates. By logical AND of the LSP's most significant bit location, MSP carry-in can be obtained. Through enhancing the LOA architecture, [12] proposes the OLOCA approximation adder. To calculating the two most significant sum bits, OLOCA only requires at least two OR gates in the LSP. The remainder of the LSP is determined by calculating as an estimate that used a fixed value.

2) FPGA-based design: Recently, a few FPGA-based approximation adders have been introduced. Becher et. al. [5] suggested a LUT-based approximation adder (LBA). Both the LSP and the MSP are capable of performing accurate addition. Only the most significant bit (MSB) of the LSP is transferred to MSP when a carry is formed. In case another carry has to be transmitted to the MSP, all bits of the LSP are set to 1. Prabakaran et. al. [4] introduced an approach for designing approximate adders based on an approximate full-adder that has been optimized. Using the optimized truth table, eight distinct multi-bit approximation adder variations were demonstrated.

A quaternary addition-based approximation adder that leverages FPGA fast carry chains is described in [6]. [6] introduced that in the quaternary addition, only one carry could be used, resulting in an approximation result.

Jha et. al. [7] presented a single exact dual approximate adder (SEDAAF). The adder may add two n-bit inputs either accurately or approximatively. A 2-bit addition's carry is calculated precisely, and the inverse of the carry out is equivalent to the sum bits.

Two FPGA approximate adders with low error are presented in [8]. Although the first approximate adder is hardware efficient, it has a low accuracy. The second approximate adder has better accuracy but is not completely optimal.

All previous work either has a high error rate or is hardware inefficient, meaning that it does not have a good trade-off between error, area, power, and delay. In this work, we resolve these two problems simultaneously using the error reduction method and the MAA structure.

B. Preliminaries

An N-bit adder takes two N-bit inputs $A = (a_{N-1}, a_{N-1}, ..., a0)$ and $B = (b_{N-1}, b_{N-1}, ..., b0)$. In the conventional Carry Look Ahead (CLA), the carry-in for each FA is computed as:

$$C_i = G_i + P_i G_{i-1} + ... + G_1 \prod_{j=2}^{i} P_k + C_0 \prod_{j=1}^{i} P_k \quad (1)$$

where C_i is the carry of i^{th} bit and P_i and G_i are the propagation bit and generation bit for i^{th} bit, respectively. The propagation and generation bit are obtained by $G_i = a_i.b_i$ and $P_i = a_i \oplus b_i$, respectively.

The number of bits released for destruction or reconstruction by an n-bit binary addition defines the size of a carry chain. The duration of a carry chain is described as $j - i$ bits when a carry is formed just at ith bit location and lost or rebuilt just at jth bit position ($j > i$). The current and preceding generation and propagation bits, based on Eq. 1, are used to derive the outgoing carry at bit location i in n-bit binary addition. Thus, the length of carry chain is considered as key principle for designing a approximate adders. To accomplish this, the investigators attempt to truncate the carry chain place at a single or even more points in order to minimize the critical path of adders.. This method increases the speed of an approximate adder while sacrificing accuracy.

Under the worst situation, a carry is formed inside the least significant bit (LSB) as well as communicated to the most significant bit (MSB) . Inside this situation, the carry chain would be half the size of the adder bit. A worst scenario, on the other hand, is uncommon, as well as the size of a carry signal is typically much smaller than the adder bit width. As a result, by capitalizing on this opportunity, the truncating carry error can be greatly reduced by simply inspecting a few earlier input bits..

III. PROPOSED DESIGN METHODOLOGY

Fig 2 presents a high-level summary of our work. It is divided into three sections:

1. Use FPGAs that support LUT6 and carry chain.

TABLE I: The impacts of raising the bit count (k) in carry predictor (C_{MSB}).

K	# LUTs	ER(%)
0	0	50
1	1	25
2	1	12.5
3	1	6.25
4	2	3.125
5	2	1.56

2. Design approximate adder with proposed methodology, which consists of 5 subsections.
3. Implementation on FPGA

The LSP's M-bit approximate sub-adder and the MSP's $(N - M)$-bit accurate sub-adder are combined to form an n-bit suggested approximation adder. The proposed design methodology is divided into several sections. Section III-A presents the carry prediction design. Section III-B presents the error reduction design. Section III-C presents the 3 bit Majority approximate adder. Section III-D Explains the adder MAA and Section III-E presents the calculation of the error metric by analytical method. Although the suggested design was developed using Xilinx FPGAs, the technology given here can be used with FPGAs from other vendors that have fractural 6-input LUTs and carry chains.

A. FPGA-Based Carry Prediction

The final summation has an erroneous value of 2^M due to truncating the carry chain at bit-position M. The error rate and error value of approximate adder can be mitigated by estimating the carry-in of the MSP (C_{MSP}). So we compensated the error of truncating carry by altering the logic function of the LSP. Each and every k-bit $(k \leq m)$ input pairs of the LSP can be used to forecast the MSP's carry-in. There is a lower probability of error rate with more bits of LSP to predict the carry-in of MSP C_{MSP}. Using additional bits of LSP to estimate MSP carry-in will, however, increase the approximation adder's area and delay. The LUT numbers and error rate for various values of k are shown in table I. The truth table and analytical method are used to calculate the error rate.

As can be seen from the use of k = 3, approximation adders for FPGAs provide a fair mix of accuracy and performance. Equations (2), (3), (4) and (5), respectively, derive the carry of the three most significant bits of the M-bit LSP (C_{M-3},C_{M-2},C_{M-1}) and C_{MSP}:

$$\mathrm{C_{M-3}} = \mathrm{G_{M-3}} \tag{2}$$

$$\mathrm{C_{M-2}} = \mathrm{G_{M-2}} + \mathrm{A_{M-2}C_{M-3}} + \mathrm{B_{M-2}C_{M-3}} \tag{3}$$

$$\mathrm{C_{M-1}} = \mathrm{G_{M-1}} + \mathrm{A_{M-1}C_{M-2}} + \mathrm{B_{M-1}C_{M-2}} \tag{4}$$

$$\mathrm{C_{MSP}} = \mathrm{C_{M-1}} \tag{5}$$

In which G_i and P_i denote the generation and propagation signals for the ith bit, respectively. A_i and B_i are i^{th} of the approximate adders' inputs.

The input carry to MSP would be anticipated to use the three most significant bits (MSBs) of LSP, as explained in (5).. When a carry (C_{M-3}) is produced at bit location $i < (m - 3)$ and propagated to MSP in this case, an error occurs.

B. FPGA-Based Error Reduction

To lower the amount of the error, we suggest a 2-bit FPGA-based error reduction scheme (FER). FER function is described by (5) and (6) to calculate S_{M-1} and S_{M-2} which is the sum of $(M-1)^{th}$ and $(M-2)^{th}$ bit. The FER design is implemented using a LUT6_2 as shown in Figure 2.B.

$$\mathrm{S_{M-2}} = \mathrm{P_{M-2}} \tag{6}$$

$$\mathrm{S_{M-1}} = \mathrm{P_{M-1}} \oplus \mathrm{G_{M-1}} \tag{7}$$

The proposed approximate adder's general architecture for FPGAs is shown in Figure 3. In this architecture, 3 MSBs of LSP is used to predict the C_{MSP}, whereas the sum of its two MSBs are computed using FER module.

C. FPGA-Based Majority Approximate Adder

Six-input LUTs are being used in modern FPGAs., and they can be used to create either one six-input function or two five-input functions. The performance of LUT-based implementation is unaffected by the complexity of the implemented logic function. There are 6 inputs and 3 outputs in a 3-bit adder. As a result, regardless of the carriers, a LUT6 can be utilized to create a 3-bit approximation adder.

To make FPGAs more efficient, the first $M-2$ bit of LSP is divided into three groups, each of which is mapped to a LUT. Furthermore, independent of the input carry, adding two 3-bit inputs from each group utilizes a 3-bit Majority Approximate Adder (MAA3). As a result, the input carry is eliminated in this scenario, and the critical path delay is minimized.

The MAA3 structure and a tiny truth table are depicted in Figure 2.C. Regardless of the input carry, the MAA3 calculates the sum of all three bits separately and accurately, and the MSB of MAA3 is given priority to reduce the magnitude of the error (when one of the two bits A_2 or B_2 is one), and then MAA3 votes from the total of three bits and takes the majority to the output.

The sum of three bits and MAA3 can be described by (8), (9), (10) and (11), respectively.

$$\mathrm{S_0} = \mathrm{P_0} \tag{8}$$

$$\mathrm{S_1} = \mathrm{P_1} \oplus \mathrm{G_0} \tag{9}$$

$$\mathrm{S_2} = \mathrm{P_2} \oplus \mathrm{G_1} \tag{10}$$

$$\mathrm{MAA3} = \mathrm{A_2} + \mathrm{B_2} + \mathrm{Majority(S_2, S_1, S_0)} \tag{11}$$

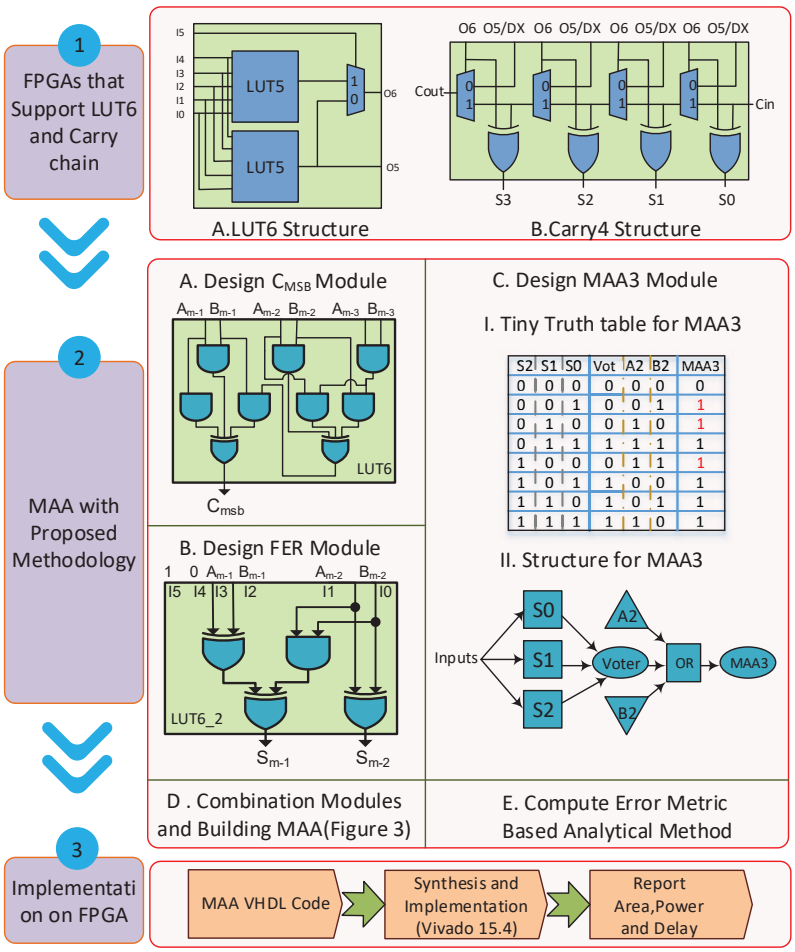

Fig. 2: Overview of our work flow.

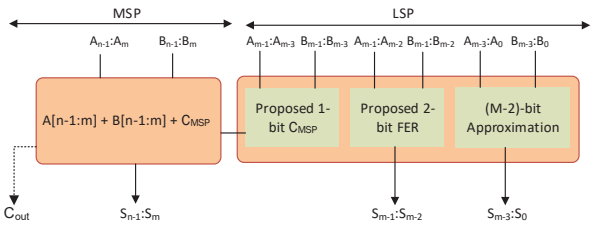

Fig. 3: The proposed approximate adder's general architecture for FPGAs.

A n-bit approximation adder architecture named MAA is suggested based on carry prediction, FER, and majority approximate adder. The $m-2$ LSBs of the LSP are approximated using several approximate functions in this proposed n-bit approximate adder.

D. Proposed Approximate Adder

In this section, we introduce the MAA approximation adder, which is low-error and hardware-efficient. Figure 4 depicts the proposed approximate adder (MAA). MAA3 provides a 6-to-1 logic function, which in MAA is mapped to a LUT. As the same way, FER and C_{MSP} are mapped to a LUT. As a result, for LSP, $[m/2]$ LUT is used. These LUTs operate in parallel. As a result, the LSP delay is the same as the LUT delay (TLUT). From the $Am-3$ input to the $Sn-1$ output, the MAA critical path exists.

An n-bit MAA's error probability is determined by the number of MAA3s used in the approximate section.The output of each of these three-bit adders (MAA3) can contain an error.

Table II shows the inputs with the INIT quantities used per LUT, as well as the configuration of the input/output pins for MAA.

E. Compute Error Metric Based Analytical Method

We calculate the MED of the two different proposed modules MAA3, proposed FER, and approximate part(LSP) by analytical method and based on truth table. The Mean Error Distance (MED) is the mean about an Error Distance (ED), being the absolute difference among approximation and exact outputs for all possible inputs [11]. The MED of MAA3 are given by:

$$\begin{aligned}\mathrm{MED}_{\mathrm{MAA3}}(\mathrm{C_{in}}=0) = 10*(0/3)+30*(1/3)+\\ 18*(2/3)+6*(3/3)=0.4375\end{aligned} \quad (12)$$

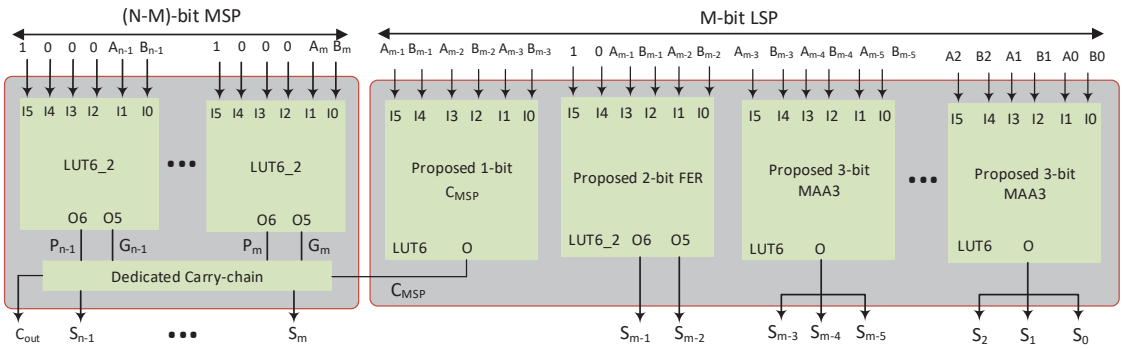

Fig. 4: The proposed approximate adder structure of MAA

TABLE II: Setup of LUT input and output pins for 16bit MAA with an 8bit approximation(LUT1 to LUT4 for approximate part and LUT5 to LUT12 for accurate part with Carry4)

Module	LUT Input Pins Configuration						INIT value (Hex)	LUT Output Pins Configuration	
	I5	I4	I3	I2	I1	I0		O5	O6
MAA3	A2	B2	A1	B1	A0	B0	FFFFFFFFFFFFE660	S[2:0]	-
MAA3	A5	B5	A4	B4	A3	B3	FFFFFFFFFFFFE660	S[5:3]	-
ER	1	0	A7	B7	A6	B6	8778877866666666	S6	S7
C_{MSP}	A7	B7	A6	B6	A5	B5	FFFFF880F8800000	C_{MSP}	-
PG0	1	0	0	0	A0	B0	0000000600000008	G0	P0
PG1	1	0	0	0	A1	B1	0000000600000008	G1	P1
PG2	1	0	0	0	A2	B2	0000000600000008	G2	P2
PG3	1	0	0	0	A3	B3	0000000600000008	G3	P3
PG4	1	0	0	0	A4	B4	0000000600000008	G4	P4
PG5	1	0	0	0	A5	B5	0000000600000008	G5	P5
PG6	1	0	0	0	A6	B6	0000000600000008	G6	P6
PG7	1	0	0	0	A7	B7	0000000600000008	G7	P7

$$\mathrm{MED_{MAA3}(C_{in}=1)} = 6*(0/3)+30*(1/3)+ 17*(2/3)+11*(3/3)=0.5 \quad (13)$$

$$\mathrm{MED_{MAA3}} = \frac{\mathrm{MED_{MAA3}(C_{in}=0)+MED_{MAA3}(C_{in}=1)}}{2} = 0.4687 \quad (14)$$

where 0/3 means three sums are correct and the rest of the cases are the same.

The MED of FER are given by:

$$\mathrm{MED_{FER}(C_{in}=0)} = 0 \quad (15)$$

$$\mathrm{MED_{FER}(C_{in}=1)} = 10*(1/2)+6*(2/2)=0.6875 \quad (16)$$

$$\mathrm{MED_{FER}} = \frac{\mathrm{MED_{FER}(C_{in}=0)+MED_{FER}(C_{in}=1)}}{2} = 0.3437 \quad (17)$$

where 1/2 means one of sums are incorrect and the rest of the cases are the same.

The MED of 8-bit LSP are given by:

$$\mathrm{MED_{LSP}} = \frac{\mathrm{MED_{MAA3}+MED_{MAA3}+MED_{FER}}}{3} = 0.42 \quad (18)$$

IV. Experimental Results and Analysis

We compare its experimental results of the proposed FPGA-based approximate adder to those of advanced FPGA-based approximate adders such as DeMAS [4], SEDAAF [7], LEADx [8] and APEx [8].

DeMAS can be created in a variety of ways. Each of these configurations has the same area for a given number of estimated bits. As a result, we compared the setup with the lowest mean error.

A. Error Metrics

Functional descriptions of the approximate adders are implemented via Python. We used the Monte Carlo simulation method to verify the accuracy of adders. The approximate metrics of errors are calculated by using 10^4 unified randomly generated input. Besides reducing the exact result as from the approximate result, the error value for each input is estimated as well.

Analytical and simulation-based approaches have been presented to ensure performance of approximate adders.

Given the accurate and approximation results of R and R', the error distance (ED) with $ED = |R - R'|$ will be counted for the total of n bits [11].Relative error distance (RED) with $RED = |ED/M|$ it will be counted which means a relative difference according to the accurate result [11].Mean error distance (MED), normalized mean error distance (NMED) and mean relative error distance (MRED) have been used to determine its error metrics of different design. Equations (19), (20) and (21) are commonly used to determine the accuracy of an approximation design, in where A represents the total number of input samples for a circuit and B represents the number of bits

$$\mathrm{MED} = \frac{1}{\mathrm{A}}\sum_{\mathrm{A}=1}^{\mathrm{A}} |\mathrm{ED(A)}| \quad (19)$$

$$\mathrm{NMED} = \frac{\mathrm{MED}}{2^{\mathrm{B}}} \quad (20)$$

$$\mathrm{MRED} = \frac{1}{\mathrm{A}}\sum_{\mathrm{A}=1}^{\mathrm{A}} |\mathrm{RED(A)}| \quad (21)$$

TABLE III: Implementation results and error metrics of FPGA-based approximate adders with a 16-bit length and an 8-bit approximation.

Design	#LUTs	Power(mw)	Latency(ns)	MED	NMED	MRED
Accurate_Vivado	16	11.887	7.119	0.0	0.0	0.0
DeMas_6[4]	16	8.947	5.159	47.412	0.00072	0.0010
SEDAAF[7]	12	9.234	6.394	66.04	0.00100	0.0013
APEx [8]	**9**	**6.851**	5.159	53.23	0.00081	0.0011
LEADx [8]	12	9.654	5.159	45.9815	0.00070	0.0009
Proposed MAA	12	8.448	**5.159**	**30.67**	**0.00046**	**0.0006**

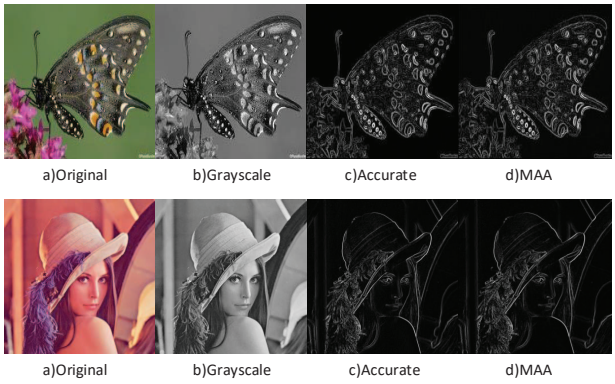

Fig. 5: Sobel edge detection output for accurate adder and 8-bit MAA

Table III shows metrics of errors of adders with a 16-bit length and an 8-bit approximation. The MED of our proposed approximate adder is the lowest. MED of approximate adder MAA are at least 49.92% lower than approximate adders in the research literature. By precisely estimating the carry to the MSP using C_{MSP}, the inaccuracy magnitude of our proposed approximate adder has been greatly reduced. MAA3 and C_{MSP} both make extensive use of LUT inputs to obtain low error.

MAA is intended to reduce not only the count of error cases, but also the amount of errors. According to experimental results, MAA has a higher accuracy and a smaller MED than other similar approximate adders.

B. Implementation Results

All approximate adders were implemented via VHDL. All adders have the same identical portion, which is implemented using one LUT6 2 each bit and one Carry4 per four bits. Vivado 15.4 is used to implement VHDL RTL code on the Xilinx Virtex 7 family device xc7vx485tffq1157-1. the default strategy is used for synthesis.

Table III shows the results of implementing adders with a 16-bit length and an 8-bit approximation. Input and output registers are used to implement all adders. The release of carriers in SEDAAF's LSP causes it to be slower than other adders. The critical path delay is the same for all other approximate 16-bit adders.

In comparison to accurate adders, all 16-bit approximate adders use less LUTs. Because all of these adders use an accurate adder in their MSP, the LUTs are only reduced in the LSP. MAA3 uses 200% less LUTs than a 3-bit accurate adder because it performs 3-bit summation in one LUT.

The least amount of LUTs are used by APEx. The inclusion of fixed functions in their LSPs results in a considerable decrease in the number of LUTs for this approximate adder. The approximate approaches used by the synthesis tool permit the synthesis tool to combine two or three output sums into a single LUT, resulting in a decrease in the number of LUTs for those other approximate adders.

The proposed approximate adder here consumes less power than the precise adder. Among all approximate adders, MAA after APEx uses the least amount of power. The power consumption of MAA is 28.93% lower than accurate adder and 23.31% higher than the best low-consumption adder, APEx, for 16-bit adders with 8-bit approximation. However, as previously stated, APEx is of lesser quality than MAA. When compared to an accurate adder, MAA significantly reduces power well at cost of a tiny reduction in accuracy.The SEDAAF has low accuracy compared to other FPGA-specific adders, but it reduces LUTs by 25% especially in comparison to the accurate adder. In comparison to various FPGA adders in the research literature, except APEx, These findings indicate that our proposed adder is lesser in size, has less power, and, of course, is of higher quality. While using an 8-bit approximation, MAA has a 33.33% smaller area and a 59.54% lower MED than DeMAS. According to [8], the LEADx approximation adder is amongst the most effectual FPGA-based approximation adders in the research literature. When implemented on FPGAs, MAA has greater quality than LEADx at the same cost; with 8-bit approximation, MAA has 49.92% less MED than LEADx at the same cost.

We also evaluate the proposed approximate adder's performance in image processing applications. MAA was used in the sobel edge detection applications, and the results are shown in Figure 5. As it can be seen, edge detection outcomes of approximate one still provides high accuracy compared to exact one.

V. Conclusion

In this paper, we introduced a novel method for building approximate adders that deploy the target FPGA architecture resources. Using this method, we created a low-error efficient approximate adder for Xilinx FPGAs with LUT6. The area, power and delay of the proposed adder are all successful in achieving reductions of 34%, 41%, and 38%, respectively. The MED of MAA, the approximation adder, is lower than that of the other approximate adders in the literature. As a result, the proposed approximate adder can be used to implement error-tolerant applications on an FPGA such as image processing.

ACKNOWLEDGEMENTS

We acknowledge the support from Government of Canada Technology Demonstration Program and MDA Systems Ltd; NSERC Discovery Grant RGPIN341516, RGPIN-2019-04613, DGECR-2019-00120, Alliance Grant ALLRP-552042-2020, COHESA (NETGP485577-15), CWSE PDF (470957); CFI John R. Evans Leaders Fund; Simon Fraser University New Faculty Start-up Grant.

REFERENCES

[1] N. Zhu, W. L. Goh, G. Wang, and K. S. Yeo, "Enhanced low-power high-speed adder for error-tolerant application," in *2010 International SoC Design Conference*. IEEE, 2010, pp. 323–327.

[2] 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, 2009.

[3] H. Jiang, C. Liu, L. Liu, F. Lombardi, and J. Han, "A review, classification, and comparative evaluation of approximate arithmetic circuits," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 13, no. 4, pp. 1–34, 2017.

[4] B. S. Prabakaran, S. Rehman, M. A. Hanif, S. Ullah, G. Mazaheri, A. Kumar, and M. Shafique, "Demas: An efficient design methodology for building approximate adders for fpga-based systems," in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2018, pp. 917–920.

[5] A. Becher, J. Echavarria, D. Ziener, S. Wildermann, and J. Teich, "A lut-based approximate adder," in *2016 IEEE 24th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*. IEEE, 2016, pp. 27–27.

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

[7] C. K. Jha, K. Prasad, A. S. Tomar, and J. Mekie, "Sedaaf: Fpga based single exact dual approximate adders for approximate processors," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2020, pp. 1–5.

[8] W. Ahmad, B. Ayrancioglu, and I. Hamzaoglu, "Low error efficient approximate adders for fpgas," *IEEE Access*, vol. 9, pp. 117 232–117 243, 2021.

[9] F. Ebrahimi-Azandaryani, O. Akbari, M. Kamal, A. Afzali-Kusha, and M. Pedram, "Block-based carry speculative approximate adder for energy-efficient applications," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, no. 1, pp. 137–141, 2019.

[10] 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)*. IEEE, 2015, pp. 1–6.

[11] M. Masadeh, O. Hasan, and S. Tahar, "Error analysis of approximate array multipliers," *arXiv preprint arXiv:1908.01343*, 2019.

[12] 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*, vol. 26, no. 8, pp. 1595–1599, 2018.