An Accuracy-Controllable Approximate Adder for FPGAs

Masaki Sano [1], Hiroki Nishikawa [2], Xiangbo Kong [1], Hiroyuki Tomiyama [1], Tongxin Yang [3†], Tomoaki Ukezono [4], and Toshinori Sato [4]

[1] *Graduate School of Science and Engineering, Ritsumeikan University, Shiga, Japan*
[2] *Graduate School of Information Science and Technology, Osaka University, Osaka, Japan*
[3] *Sony Semiconductor Solutions Corporation, Kanagawa, Japan*
[4] *Department of Electronics Engineering and Computer Science, Fukuoka University, Fukuoka, Japan*

Abstract

In this paper, we propose an accuracy-controllable approximate adder for FPGAs. The proposed adder has a special input to dynamically change the accuracy in addition to two operands. When accurate computation is required, the adder computes accurately. On the other hand, when accurate computation is not required, the adder computes inaccurately but quickly at low power. The important feature of our adder is that it utilizes carry-chain modules which are built in FPGAs. By using the carry chains, our approximate adder computes much faster at lower power than an existing approximate adder.

Keywords

approximate computing, approximate adder, FPGA

1. Introduction

In recent years, approximate computing technology has attracted attention to achieve higher performance and lower power consumption by tolerating a certain degree of computational error. Approximate computing techniques are used especially in the fields of image processing and machine learning since they are computationally expensive and error-tolerant to some extent [1][2].

Research on approximate computing circuits has been conducted at various design levels from the transistor to the architecture levels [3][4]. This paper focuses on approximate adders since addition is one of the most fundamental arithmetic operations. There are many studies on approximate adders which improve power-performance efficiency by disconnecting carry propagation [5]-[9]. The work in [10] proposes an approach to accurately calculate errors of approximate adders, and the work in [11] provides a detailed analysis of the trade-off between accuracy and resource efficiency. The authors of [12][13] state that circuit design with variable computational accuracy is desirable for designing systems that meet diverse requirements. An approximate adder circuit, named carry-maskable adder (CMA), that can change the computational accuracy is proposed in [14]. Since CMA enables dynamic control of accuracy, it is possible to perform approximate operations within an error tolerable range.

Based on CMA in [14], in this paper, we propose an accuracy-controllable approximate adder for FPGAs. If the original CMA is implemented in FPGAs in a straightforward manner, lookup tables (LUTs) are connected in series, and hence, the delay and power increase significantly. Our proposed adder, named carry-chain based carry-maskable adder (CC-CMA), takes advantage of fast carry-chain modules which are built in FPGAs.

This paper is organized as follows. Section 2 presents CC-CMA, and Section 3 analyzes the hardware cost, delay, computational error and power consumption of CC-CMA. Finally, Section 4 summarizes this paper and discusses future work.

† This work was done while the author was with Fukuoka University, Japan.

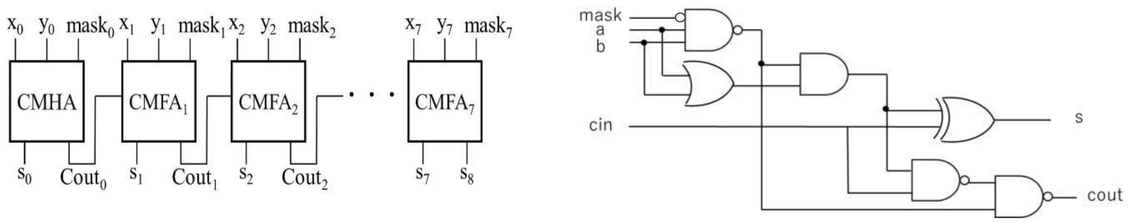
The 4th International Symposium on Advanced Technologies and Applications in the Internet of Things (ATAIT 2022), August 24-26, 2022, Ibaraki, Japan

2. Carry-Chain based Carry-Maskable Adder

In this section, we first explain the carry-maskable adder [14], and then, we propose a carry-chain based carry-maskable adder for FPGAs.

2.1. Carry-Maskable Adder

This work is based on an approximate adder, named carry-maskable adder (CMA), which was originally proposed in [14]. CMA can dynamically change the accuracy level according to the special input signal. Figure 1 (a) shows the diagram of an 8-bit CMA. The 8-bit CMA consists of a carry-maskable half adder (CMHA) and seven carry-maskable full adders (CMFAs) connected in series, as shown in Figure 1 (b). In addition to three inputs *x*, *y* and carry-in denoted as *cin*, CMFA has a special input named a *mask* to dynamically control the accuracy. If the *mask* is 0, CMFA performs exact addition. If the *mask* is 1, the carry-out signal *Cout* is 0, and the sum *s* is the logical sum of *a* and *b* assuming that the carry-in signal from the lower bit is 0. This computation is not accurate but approximate. However, since carry signals are not propagated from lower bits to upper bits, the delay and power consumption are reduced. By setting masks of lower bits to 1 and those of upper bits to 0, the upper (more significant) bits are computed accurately, and the lower (less significant) bits are computed approximately. Thus, by controlling *mask*, we can explore the trade-off between accuracy, delay, and power consumption, depending on the requirement of the applications.


x_0 y_0 $mask_0$ x_1 y_1 $mask_1$ x_2 y_2 $mask_2$ x_7 y_7 $mask_7$
CMHA
$CMFA_1$
$CMFA_2$
$CMFA_7$
s_0 $Cout_0$ s_1 $Cout_1$ s_2 $Cout_2$ s_7 s_8
mask
a
b
cin
s
cout

(a) 8-bit CMA (b) Carry-maskable full adder
Figure 1. Carry-maskable adder [14]

2.2. Carry-Chain Based CMA

Originally, CMA was designed for ASICs, and how to implement CMA on FPGAs is not presented or discussed in [14] despite the widespread use of FPGAs. If the Boolean expressions of CMA are given to FPGA synthesis tools, each CMFA is mapped to one or two lookup tables (LUTs), and the LUTs are connected in series, as shown in Figure 1 (a). This implementation is not efficient in terms of delay and power consumption since LUTs are slow and power-consuming.

Recent FPGAs are equipped with carry-chain modules for fast addition. Figure 2 shows a schematic diagram of an accurate 4-bit adder using a built-in carry chain module. The carry-chain module consists of four multiplexers and four EXOR gates, and is connected from four LUTs. For accurate addition, each LUT is configured to compute EXOR of *x* and *y* as follows.

$$P = x \oplus y \tag{1}$$

As seen in Figure 2, carry signals go through the carry-chain module. In other words, the carry signals do not go through LUTs which are slow and power-consuming. Thanks to the built-in carry-chain module, addition is computed fast at low power.

In this work, we take advantage of the carry-chain modules in the design of approximate adders. Figure 3 shows our proposed approximate adder, named carry-chain based carry-maskable adder (CC-CMA. LUTs labeled P0-P3 and G0-G3 compute Equations (2) and (3), respectively.

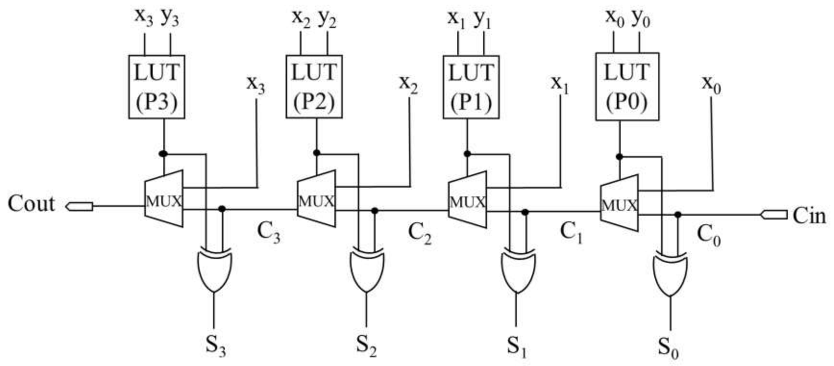


Figure 2. 4-bit accurate adder using carry chain

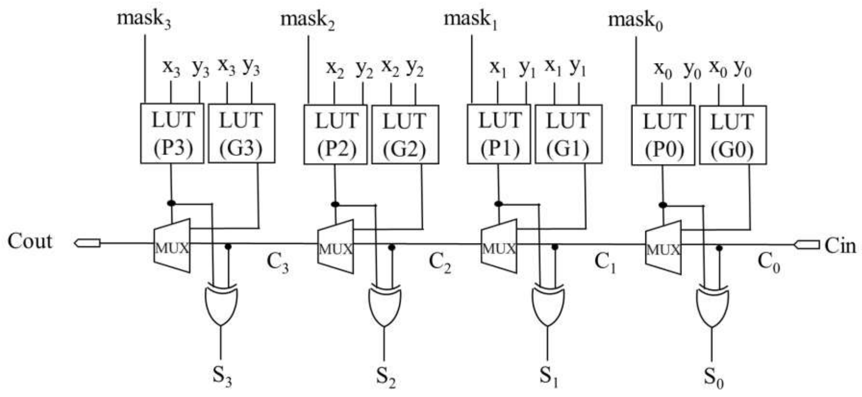


Figure 3. 4-bit CC-CMA

$$P = (x \cdot y \cdot mask) + (x \oplus y) \tag{2}$$
$$G = x \cdot y \tag{3}$$

Similar to the accurate adder shown in Figure 2, carry signals in CC-CMA do not go through LUTs but go through fast carry-chain modules when *mask* is 0. When *mask* is 1, carry signals do not propagate to upper bits, which achieves faster and lower-power computation than the accurate adder at the expense of computational inaccuracy.

3. Evaluation

We have designed 32-bit and 64-bit CC-CMAs in Verilog-HDL, and synthesized them for Xilinx Artix-7 device with Xilinx Vivado 2019.2. For comparison, we have also designed accurate adders and original CMAs [14] in Verilog-HDL. The synthesized accurate adders utilize built-in carry-chain modules, as shown in Figure 2, while the synthesized CMAs do not. The three adders are compared in terms of hardware resources, delay, power consumption, maximum error and average error. Delay, average error and power consumption are obtained by post-synthesis simulation using the Vivado toolkit. For 32-bit and 64-bit adders, 100,000 and 1,000,000 random simulations are performed, respectively. Delay, error and power consumption of CMA and CC-CMA depend on the values of masks. Recall that CMA and CC-CMA compute accurately when masks of all bits are 0. When the masks of the least significant *n*-bits are set to 1, the lower *n*-bits are added approximately, and the upper bits are added accurately. In our experiments, we vary the number of lower bits whose masks are set to 1.

3.1. Hardware Resources

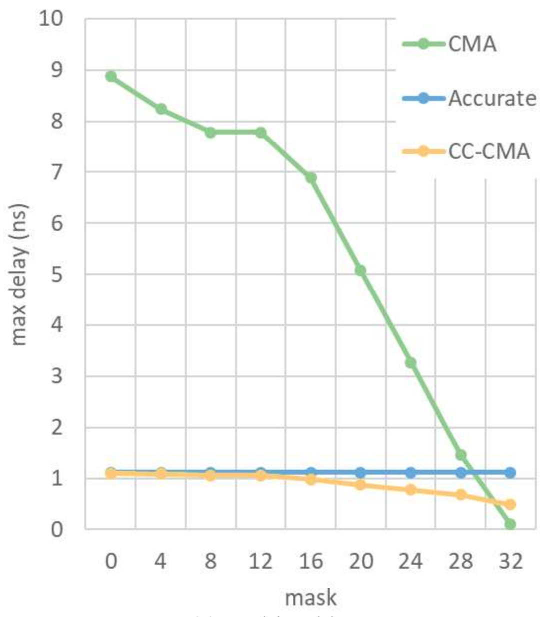
Table 1 compares hardware resources in terms of the number of 6-input LUTs. The original CMAs use twice as many LUTs as the accurate adders and CC-CMAs. CC-CMAs are as small as the accurate adders, although the functionality of CC-CMAs is more complex.

Table 1. Number of LUTs

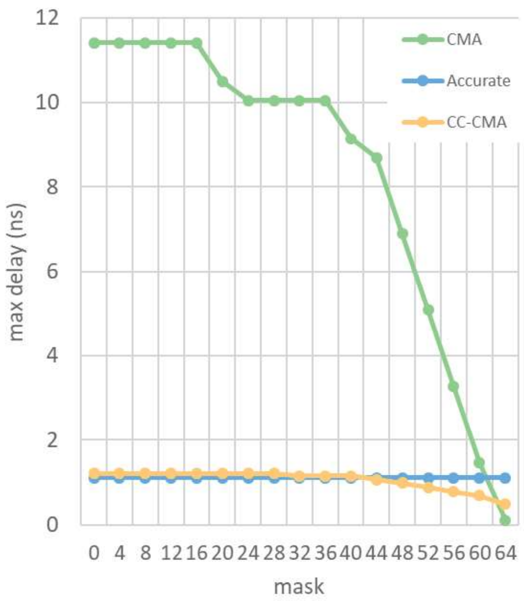
	Accurate adder	CMA	CC-CMA
32-bit adders	32	63	32
64-bit adders	64	127	64

3.2. Delay

Figure 4 shows the delay of the adders. The longest delays among 100,000 and 1,000,000 random simulations are shown for 32-bit and 64-bit adders, respectively. For CMA and CC-CMA, the value of the mask is varied. The X-axis in the figure shows the number of lower-bits whose mask is set to 1. For example, in Figure 4 (a), when the mask is 0, CMA and CC-CMA compute accurately. When the mask is 4, the lower 4 bits are computed approximately, and the upper 28 bits are computed accurately.



(a) 32-bit adders



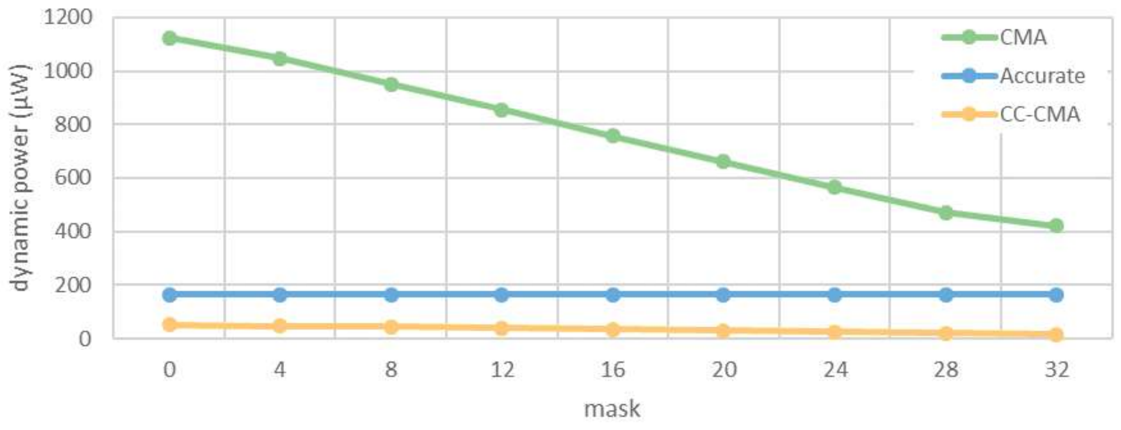
(b) 64-bit adders

Figure 4. Delay (ns)

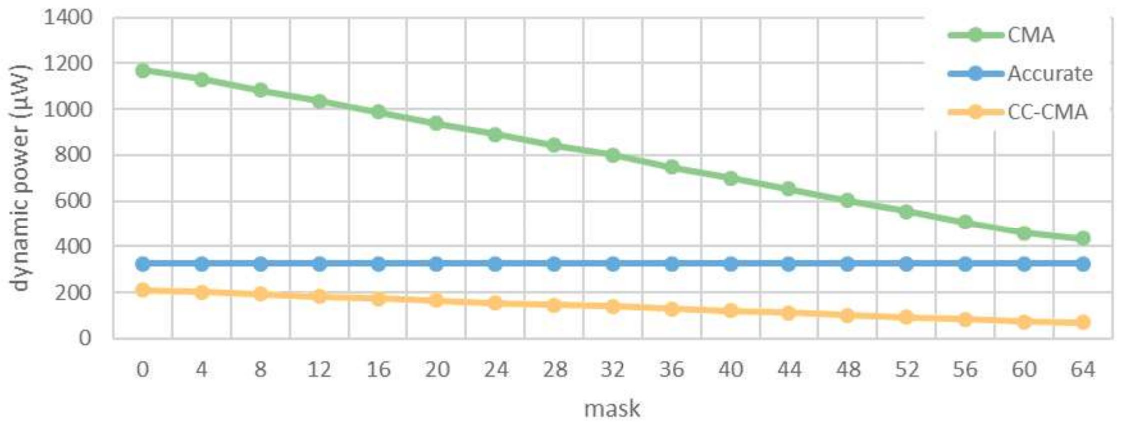
Figure 4 clearly shows that addition is computed faster when more bits are approximated. The figure also demonstrates that CC-CMAs are not efficient because they do not take advantage of built-in carry-chains.

3.3. Power Consumption

Figure 5 shows the power consumption of the adders. The original CMAs consume much more power than the accurate adders and CC-CMAs. An interesting observation in the figure is that CC-CMAs consume less power than the accurate adders even in case mask is 0 and CC-CMAs compute accurately. Although the Boolean function of CC-CMA is more complex than that of the original CMA, the internal signals (e.g., the input signals to the multiplexers) of CC-CMA switch less frequently,



(a) 32-bit adders



(b) 64-bit adders

Figure 5. Power consumption (μW)

leading to lower power consumption. It is also observed in Figure 5 that the power consumption of CMA and CC-CMA decreases as more bits are masked and approximated.

3.4. Computational Errors

So far, we have seen that CC-CMA computes faster at lower power than the accurate adders. These advantages come at the cost of computational error. Table 2 shows the degree of computational errors of CMA and CC-CMA. Recall that the functions of CMA and CC-CMA are exactly the same, and therefore, the amounts of errors of the two adders are the same. In the table, “max” denotes the maximum error derived theoretically, and “average” denotes the average error obtained from the random simulation. As more bits are masked, the computational error increases.

Table 2. Computational errors of CMA and CC-CMA

(a) 32-bit adders

mask	4	8	12	16	20	24	28	32
max	2^4-1	2^8-1	$2^{12}-1$	$2^{16}-1$	$2^{20}-1$	$2^{24}-1$	$2^{28}-1$	$2^{32}-1$
average	4.139	64.45	1.025×10^3	1.638×10^4	2.634×10^5	4.185×10^6	6.701×10^7	1.071×10^9

(a) 64-bit adders

mask	4	8	12	16	20	24	28	32
max	2^4-1	2^8-1	$2^{12}-1$	$2^{16}-1$	$2^{20}-1$	$2^{24}-1$	$2^{28}-1$	$2^{32}-1$
average	4.047	64.85	1.028×10^3	1.642×10^4	2.618×10^5	4.195×10^6	6.688×10^7	6.667×10^8
mask	36	40	44	48	52	56	60	64
max	$2^{36}-1$	$2^{40}-1$	$2^{44}-1$	$2^{48}-1$	$2^{52}-1$	$2^{56}-1$	$2^{60}-1$	$2^{64}-1$
average	1.616×10^{10}	2.735×10^{11}	4.381×10^{12}	7.065×10^{13}	1.122×10^{15}	1.804×10^{16}	2.886×10^{17}	2.875×10^{18}

3.5. Trade-off between Error, Delay and Power

From Figure 4 and Table 2, the trade-off between delay and computational error for 32-bit CC-CMA can be derived as shown in Figure 6. The figure shows that the delay can be shortened at the cost of computational error, but the cost is not low. Also, it is not beneficial to dynamically change the delay of adders unless the adders exist on the critical path of the entire circuits.

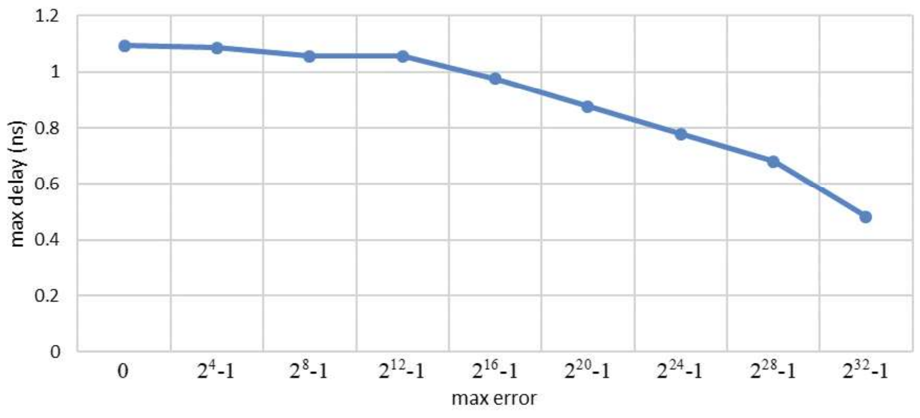


Figure 6. Trade-off between delay and error for 32-bit CC-CMA

From Figure 5 and Table 2, the trade-off between power consumption and computational error for 32-bit CC-CMA can be derived as shown in Figure 7. A significant amount of power can be saved at the expense of computational error.

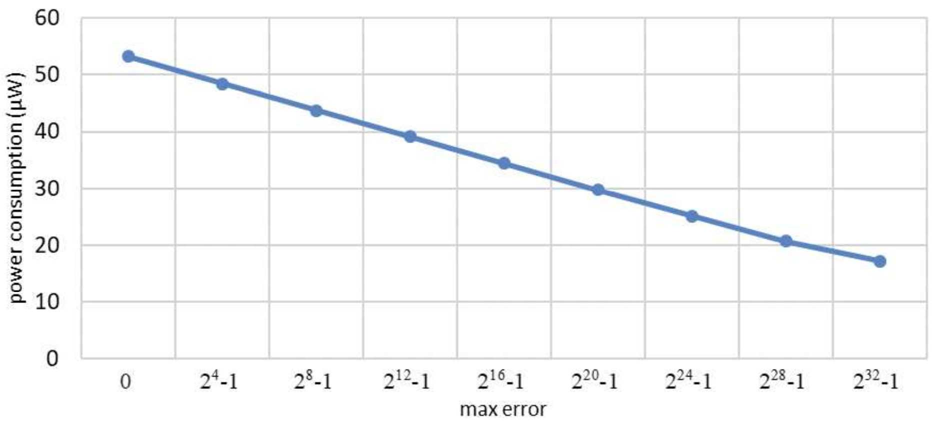


Figure 7. Trade-off between power consumption and error for 32-bit CC-CMA

4. Conclusions

In this paper, we have proposed an approximate adder named carry-chain based carry maskable adder (CC-CMA) for FPGAs. CC-CMA has a special input signal named a mask to dynamically control the computational accuracy. CC-CMA takes advantage of fast carry-chain modules which are equipped in modern FPGAs. By using the built-in carry chains, CC-CMA computes fast at low power. The experimental results demonstrate the efficiency of CC-CMA compared with an accurate adder and an existing carry-maskable adder. In future, we plan to evaluate CC-CMA using real-world applications. Also, we plan to develop accuracy-controllable multipliers for FPGAs based on CC-CMA.

Acknowledgments

This work is supported partly by KAKENHI 20H00590, 19H04081 and 21K19776.

References

[1] H. Esmaeilzadeh, A. Sampson, L. Ceze and D. Burger, "Neural acceleration for general-purpose approximate programs," *IEEE/ACM International Symposium on Microarchitecture*, 2012.

[2] V. Gupta, D. Mohapatra, S. P. Park, A. Raghunathan and K. Roy "IMPACT: IMPrecise adders for low-power approximate computing," *IEEE/ACM International Symposium on Low Power Electronics and Design*, 2011.

[3] S. Mittal, "A survey of techniques for approximate computing," *ACM Computing Surveys*, 2016.

[4] Q. Xu, T. Mytkowicz, and N. S Kim, "Approximate computing: A survey," *IEEE Design & Test*, 2016.

[5] R. Gollu and P. Neeta, "New carry maskable adder using modified full swing GDI," *Journal of Physics Conference Series*, vol.1921, no.1, article. 012049, 2021.

[6] P. K. Sujit, G. Bharat, and R. K. Shireesh, "A power and area efficient approximate carry skip adder for error-resilient applications," *Turkish Journal of Electrical Engineering and Computer Sciences*, vol.28, no.1, pp.443-457, 2019.

[7] J. Babita, M. Vishesh, and S. Saurabh, "EFCSA: An efficient carry speculative approximate adder with rectification," *IEEE 23rd International Symposium on Quality Electronic Design*, 2022.

[8] L. Jungwon, S. Hyoju, K. Yerin, and K. Yongtae, "Approximate adder design with simplified lower-part approximation," *IEICE Electronics Express*, vol. 17, no. 15, pp. 1-3, 2020.

[9] A. Kanani, J. Mehta and N. Goel, "ACA-CSU: A carry selection based accuracy configurable approximate adder design," *IEEE Computer Society Annual Symposium on VLSI*, 2020.

[10] M. Rezaalipour, M. Rezaalipour, M. Dehyadegari and M. N. Bojnordi, "AxMAP: Making approximate adders aware of input patterns," *IEEE Transactions on Computers*, vol. 69, no. 6, pp. 868-882, 2020.

[11] D. Catelan, R. Santos and L. Duenha, "Accuracy and physical characterization of approximate arithmetic circuits," *XXI Simpósio em Sistemas Computacionais de Alto Desempenho*, 2020.

[12] S. Venkataramani, V. K. Chippa, S. T. Chakradhar, K. Roy, and A. Raghunathan, "Quality programmable vector processors for approximate computing," *International Symposium on Microarchitecture*, 2013.

[13] A. B. Kahng and S. Kang, "Accuracy-configurable adder for approximate arithmetic designs," *Design Automation Conference*, 2012.

[14] T. Yang, T. Ukezono and T. Sato, "An accuracy-configurable adder for low-power applications," *IEICE Trans. on Electronics*, vol. E103-C, no. 3, pp. 68-76, 2020.