RESEARCH ARTICLE OPEN ACCESS

Error Efficient Approximate Computing using Adder Optimization Technique for 32-bit Ripple Carry Adder

Mary getsy. D[1], Sharmila. B[2], Sivaranjini. S[3], Sudha kumari. R[4]

[1]Assistant Professor of Electronics and Communication Department, Manakula Vinayagar Institute of Technology, Pondicherry University.

[2]Student of Electronics and Communication Department, Manakula Vinayagar Institute of Technology, Pondicherry University.

[3]Student of Electronics and Communication Department, Manakula Vinayagar Institute of Technology, Pondicherry University.

[4]Student of Electronics and Communication Department, Manakula Vinayagar Institute of Technology, Pondicherry University.

[1]marygetsyece@mvit.edu.in, [2]sharmilababu2001@gmail.com, [3]ranjinisiva159@gmail.com, [4]sudharaghunath1526@gmail.com

ABSTRACT:Approximate Computing has a vital role in various fields such as microprocessors, processing in digital signal, data analytics, image processing, machine learning, and so on. When speed and accuracy are the main concerns in today's systems, an error-efficient approximate adder architecture is developed by using a technique called "Adder Optimization". Approximate Computing is a technique that allows digital circuits to trade off circuit accuracy with performance, which is particularly useful for applications that can tolerate some level of errors.

An approximate or subordinate solution is sufficient for the user and is not required to be perfect in every way.

Keywords: Accuracy, Approximate computing, Error-efficient, Ripple Carry Adder, Adder Optimization, Trade-off.

I.INTRODUCTION

Although they may not always be required in a practical application, accurate computation results are invaluable in areas like vision in computers, data mining, machine learning, cloud-based computing, microprocessors, processing of digital signals. Instead, it may be acceptable to accept the results of an approximately correct computation that are provided and fall within a certain error range. This is achieved through the use of human perception. A significant portion of the inspiration for approximation computing comes from the fact that many real-world applications, including those already stated, are naturally error-tolerant. Consequently, utilising an approximation could allow for the usage of fewer resources to provide an acceptable result instead of using all of the resources available to produce an accurate result, which will lead to lower power consumption and higher system performance.

Approximate computing aims to provide a way to optimize computing systems for performance, energy efficiency, and cost-effectiveness, while still providing results that are accurate enough for the intended application. Our scope of this paper is to develop an error-efficient approximate adder with a lesser number of gates and higher performance. In the context of ripple carry adders, we are using a technique called Adder Optimization which is used to improve their efficiency by reducing the gates in number and thereby decreasing delay and consumption of power.

The ripple carry adder circuit is made up of a number of full adders, each of that adds the relevant bits from the two input numbers as well as the carry produced by the earlier addition. Each full adders' output consists of carry bit and sum bit. While carry bit sis transmitted to the next step as the carry-in-input, while the sum bit is sent to the adders' following stage. "Full-adder optimization" is the process of lowering the number of gates in a 32-bit ripple carry adder to low its delay. Full adder optimization techniques aim to enhance the full adder's performance by reducing its delay, area, power consumption, or a combination of these metrics. A Full adder is modified by implementing a hybrid-CMOS full adder, these modified full-adder circuits have fewer transistors and gates, which reduces their delay and area.

B. Block Diagram:

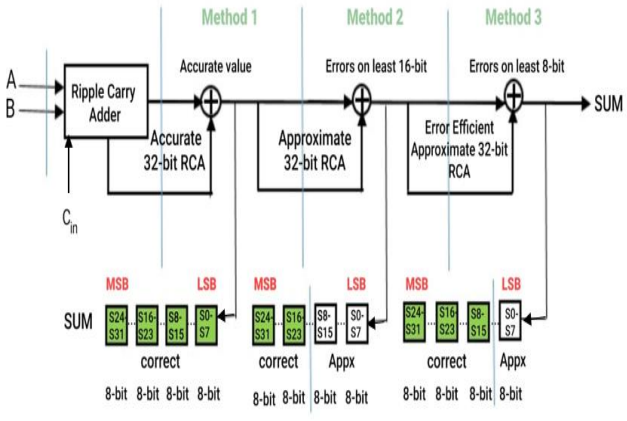

Fig.1 Block diagram

II.RATIONALITY OF OUR DESIGN AND PRIOR WORKS

We examine a few representatives' RCA designs projects and demonstrate how they relate to our approach. These designs can be classified broadly into three categories: Accurate method, Approximate method, and Error-efficient approximate method for designing full adder circuits using 32-bit RCA. These design methods are implemented using an adder optimization technique which results in the trade-off between the metrics such as delay and adding several transistors required for these designs.

The core concept of an accurate 32-bit RCA relies on accurate values, which refer to the exact values that would be obtained through traditional computing techniques. The approach begins with a full adder circuit, where each full adder's carry output connects withcarry input of the following full adder. This method starts with an approximate full adder design, formed by chaining two ex-or gates to give the sum, and the inverter is connected to the sum to get the carry-out in the 8-bit least significant positions and other bits are formed using full adder circuits. This method results in approximate values with some errors only in the 8-bit least significant bit positions and it is nearly equal to the accurate values.

The third method is for error efficient approximate adder which is developed for the impact of reducing the errors that result in the approximate method using 32-bit RCA. This method starts with an or gate that receives 2 inputs to give a sum in the 8-bit least significant bit positions and other bits are formed using full adder circuits in the most significant bit positions. This method results in fewer errors compared to the second method by using an adder optimization technique. In order to decrease the hardware complexity and delay of the circuit, these optimized adder techniques make some accuracy sacrifices. The output of the adder may have small errors due to the approximation, but these errors are generally within an acceptable range for many applications.

There is another method that is more useful for the user preference to select either accurate method sum as output or error efficient approximate method sum as output for 32-bit ripple carry adder using a 2*1 multiplexer.

TABLE I

COMPARISON OF ACCURATE AND ERROR-EFFICEINT APPROXIMATE SUMS

PRIMARY INPUTS			METHOD 1 (S=Ai+Bi+Ci)	METHOD 3 (S = Ai + Bi)
Ai	Bi	Ci		
0	0	0	0	0 (correct)
0	0	1	1	0 (incorrect)
0	1	0	1	1 (correct)
0	1	1	0	1 (incorrect)
1	0	0	1	1 (correct)
1	0	1	0	1 (incorrect)
1	1	0	0	1 (incorrect)
1	1	1	1	1 (correct)

The above table, explains the comparison of sums between the accurate 32-bit RCA method and error efficient approximate 32-bit RCA method. The sum of the first method is the combination of all three primary inputs Ai, Bi, and Ci and this is the same as the sum of a conventional full adder. The sum of the third method is the combination of two primary inputs only such as Ai and Bi which is the same as the sum of or gate.

S = Ai + Bi + Ci (first method)

S = Ai + Bi (third method)

From these above equations, the sum of the first method has no incorrect values, as it is compared with the sum of the conventional full adder values which results in accurate values. The sum of the third method has four incorrect values, as it is compared with the sum of the conventional full adder values which results in error-efficient approximate values.

III.DESIGN METRICS INVOLVED IN RCA

>>Delay: The circuit's delay is the amount of time it takes for the output to stabilize after a change in the input. A faster circuit has a lower delay, allowing for quicker processing of data.

>>Speed: The speed of the circuit is the rate at which it can perform additional operations. A faster circuit has a higher speed, enabling quicker processing of data.

>>Power consumption: The total amount of energy consumed for the operation of the circuit is indicated by its power consumption. Lower power consumption can be achieved by optimizing the design and reducing the reducing the transistors in number used.

>>Area: Area of the circuit is the physical size of the chip required to implement it. A smaller area typically means lower cost and higher integration density.

>>Number of transistors: Number of transistors which are used in the circuit is a measure of is complexity. Reducing the number of transistors can help reduce power consumption and area.

TABLE II

COMPARISON OF DELAYS AND NUMBER OF TRANSISTORS USED IN ALL METHODS

METHODS	NUMBER OF TRANSISTORS USED	TOTAL DELAY
Method 1	1984	84.6 ns
Method 2	1856	78.7 ns
Method 3	1536	61.4 ns

The above table, explains the comparison of delays and several transistors that are used in all three methods. The total propagation delay(tpd) is the sum of the interconnection delay and cell delay. The number of transistors used in method 1 is comparatively higher than in method 3 and similarly, the delay of method 3 is relatively less than method 1 which is more error efficient than method 1 which is more error efficient than method 2. This result increases in speed, with lesser number of transistors used in method which covers less area compared to other methods using the adder optimization technique.

IV.METHODOLOGY

A. *ACCURATE 32-bit RCA*

>>In this first method, we have implemented 32-bit RCA using 32 full adder circuits, where each full adder requires a minimum of 62 transistors, and that gives us accurate values without any errors in it.

>>To construct a 32-bit RCA, carry output of each full adder is coupled with carry input of full adder beyond it in chain manner. The two 32-bit input values A and B as well as 0 as the initial carry-in value A and B as well as 0 as the initial carry-in value are fed into the first complete adder circuit. The carry-out, which indicates any overflow that occurred during the addition, and the 32-bit total produced by the last full adder are the final outputs of the RCA.

>>In approximate computing, term “error-efficiency” refers to the ratio between the accuracy of the approximations and the exact results obtained using traditional computing methods. Since the results produced by the full adders are exact or accurate results in the case of an accurate 32-bit RCA, the error efficiency would be 1.

>>It’s important to note that error efficiency is not the only metric for evaluating the effectiveness of approximate computing techniques. Other metrics such as power consumption, speed, number of transistors used, and resource usage should also be considered.

>>The circuit’s use of full adders has a propagation delay that, in relation to the context of an accurate 32-bit RCA full adder, determines the delay. The delays of every full adder in the chain are added to determine the circuit’s overall delay. By using this method, the longest propagation delay is 84.6 ns.

>>In the context of an accurate 32-bit RCA using full adders, the number of transistors used in the circuit, each full adder typically requires 62 transistors to implement. Nearly 1984 transistors are required to implement this method.

>>Overall, the delay and number of transistors are important metrics to consider when designing an accurate 32-bit RCA using full adders, as they can affect the performance, cost, and complexity of the circuit.

B. *APPROXIMATE 32-bit RCA:*

>>In the second way, we designed 32-bit RCA using an approximate full adder circuit in the 8-bit least significant bit positions and 24 full adder circuits in the most significant bit positions.

>>The approximate full adder circuits are formed by using two ex-or gates to give the sum and the inverter is connected to the sum to give the carry and this carry is given as input to the next block.

>>This method produces an approximation with a few errors only in the 8-bit least significant positions and with no further changes in the positions of the most significant positions bits. The results have some tolerable errors that are nearer to the actual value.

>>An approximate 32-bit RCA’s delay is determined precisely by the propagation delays of full adders used in most significant positions and the approximation of full adders used in least significant positions.

>>The approximate full adders’ and full adder circuits’ delays are added to determine the overall delay. In this method, the longest propagation delay is 78.7 ns.

>>In the context of an approximate 32-bit RCA,number of transistors used in each full adder circuit is 62 transistors and each approximate full adder circuit is 46 transistors. Nearly 1856 transistors are required to implement this method.

C. *ERROR EFFICIENT APPROXIMATE 32-bit RCA:*

>> In this third way, we have developed an error-efficient 32-bit RCA to reduce the error that occurs in the approximate technique by reducing number of gates, which in turn reducesdelay of the circuit.

>>To build an error-efficient approximate 32-bit RCA, we need to form a chain that connects the 24 full adder circuits in the most significant bit positions and or gate in the 8-bit least significant positions.

>>This method results in fewer errors with a lesser number of transistors and delay of the circuits when compared with the approximate 32-bit RCA method. This method is more efficient for the user in many applications.

>>For example, in image processing applications, it may be acceptable to use approximate computing techniques such as adder optimization techniques that sacrifice some level of image quality in exchange for faster processing times or lower power consumption. In this case, the accurate values may be defined as the ideal pixel values that would be obtained with exact precision, but the approximate values that are used may differ slightly from these accurate values.

>>Full adders are employed in most significant positions, andor gate is employed in 8-bit least significant positions define the delay in the context of an error-efficient approximate 32-bit RCA.

>>The or gate and the full adder circuits’ combined delays give rise to the overall delay. In this method, the longest propagation delay is 61.4 ns.

>>In the context of an error-efficient approximate 32-bit RCA, the number of transistors used in each full adder is 62

transistors and each or gate us 6 transistors. Nearly 1536 transistors are required to implement this method.

*D. NEED OF 2*1 MULTIPLEXER FOR USER:*

>> In order to choose between an accurate and an approximation circuit depending on the required trade-off between accuracy, speed, and power consumption, a 2*1multiplexer is implemented in approximate computation for 32-bit RCA.

>>In approximate computing, the aim is to reduce the computational complexity and power consumption of a circuit while maintaining an acceptable level of accuracy. One way to achieve this is by using an approximate 32-bit RCA circuit instead of an accurate one.

>>The approximate 32-bit RCA circuit can be designed using simplified logic gates or by using a reduced number of logic gates and power consumption. However, using such an approximate circuit may introduce some errors in the output.

>>To address this issue, a 2:1 multiplexer can be used to select between the accurate and error-efficient approximate 32-bit RCA circuits. The selection can be made based on the application requirements, where high accuracy is required, the accurate 32-bit RCA circuit can be selected, and where low power consumption and fast operation ae important, the error-efficient approximate 32-bit RCA circuit can be selected.

>>Based on the selected input signal, the 2:1 multiplexer decides whether to output the accurate 32-bit RCA circuit or the approximate 32-bit RCA circuit.

>>If the user gives input as ‘zero’ to the selection line, then the 2:1 multiplexer selects an accurate 32-bit RCA sum as the output.

>>If the user gives input as ‘one’ to the selection line, then the 2:1 multiplexe3r selects error efficient approximate sum as the output for this method.

>>Overall, the use of a 2:1f multiplexer to select between an accurate 32-bit RCA and an approximate 32-bit RCA provides a flexible and efficient approach to error-efficient computing. This allows the circuit to achieve a balance between accuracy, speed, and power consumption, making it suitable for a wide range of applications.

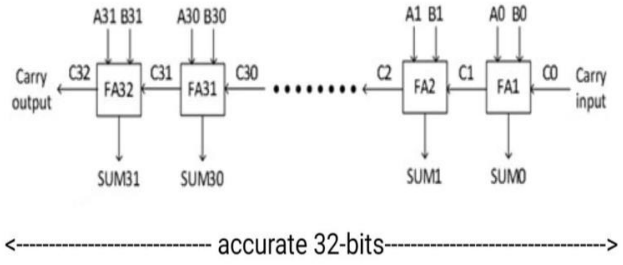

Fig.3.Accurate 32-bit RCA

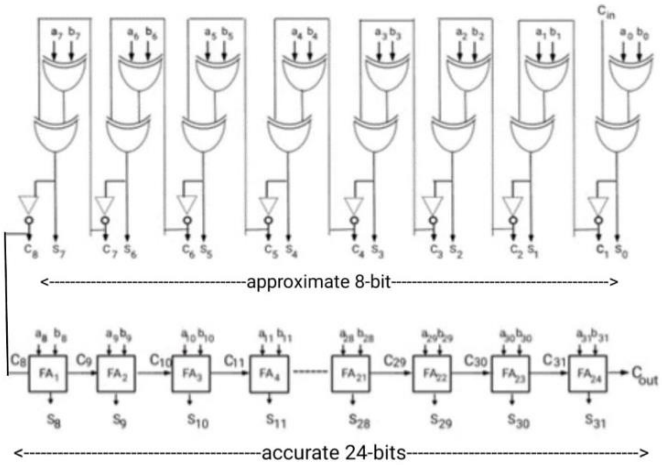

Fig.4. Approximate 32-bit RCA

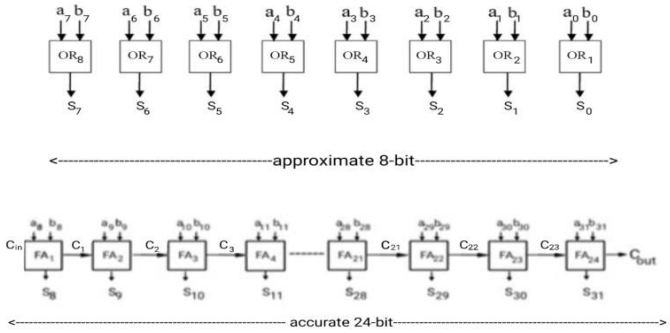

Fig.5. Error efficient Approximate 32-bit RCA

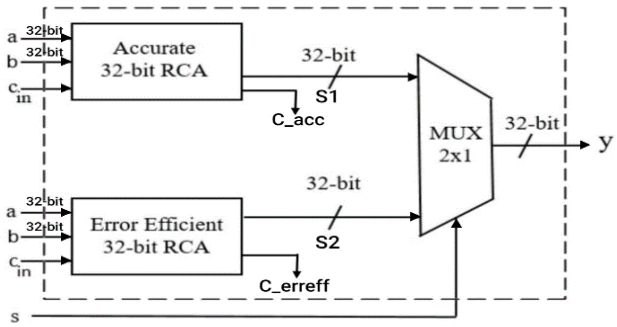

Fig.6.Need of 2x1 Multiplexer for User

V.HARDWARE IMPLEMENTATION

The project’s backend design is a different term for the hardware implementation. The Synthesis, Fitter, Assembler, and Programming procedures are just a few of the processes used in the backend design. The software phase concludes with the simulation procedure, which enables us to validate the outputs using waveform simulation in the QUARTUS II (version 8.1) tool. Digital logic circuits are designed implemented, and tested using the Quartus II software tool. It

includes a number of design flow stages, such as synthesis, fitter, assembler, and programming, each of which are explained in more depth below:

>>Synthesis: The synthesis stage is where the user's high-level RTL (Register Transfer Level) design is converted into a low-level netlist of logic gates, flip-flops, and other components. During synthesis, the Quartus II software analyses the RTL design and creates an optimized netlist based on the specified target device (FPGA). It is a process of converting HDL coding into an equivalent circuit diagram. The output of the synthesis process is seen through the software tool itself. To see the output of synthesis, go to the tools option in the menu bar and click on the Netlist viewers and to the RTL viewers. By enabling the full-mode screen option, we can view the overall circuit diagram of the project.

>>Fitter: The fitter stage is where the netlist created during synthesis is mapped to the FPGA device's specific architecture. The fitter analyses the netlist and places the logic elements onto the FPGA device's physical resources,such as logic cells, multiplexers, and routing resources. The fitter also determines the timing constraints, such as setup and hold times, to ensure that the design meets the required timing specifications. It is a process of a combination of Floorplan, Placement, and Routing.

>>Assembler: The assembler stage is where the fitter-generated output is converted into a format that can be programmed into the FPGA device. The assembler stage in Quartus II is responsible for generating. sof (SRAM Object File) format file used for programming the FPGA device. The .sof file is a configuration file that contains the bitstream data required to program the FPGA device. The .sof file includes information such as the device architecture, the location of logic elements on the device, the routing information, and the timing constraints.

>>Generation of .sof file: To generate the .sof file, the Assembler stage in Quartus II takes the netlist created by Fitter stage and converts it into a format that can be programmed onto the FPGA device. The Assembler stage performs checks on the netlist and ensures that it meets the timing and resource requirements of the FPGA device.

>>Programming: The programming stage is where the compiled design is downloaded onto the FPGA device. The programming file generated in the assembler stage is used to program the FPGA device using a JTAG programmer, such as the Byteblaster or USB-Blaster. The programming process involves loading the .sof file into the programmer and then connecting the programmer to the FPGA device using a cable. Once the FPGA device is programmed, it is ready to be tested and validated.

In summary, the Quartus II software tool provides a complete design flow for digital logic circuits, including synthesis, fitter, assembler, and programming stages. These stages help to ensure that the design is optimized for the target FPGA device, meets timing specifications, and can be programmed onto the device successfully.

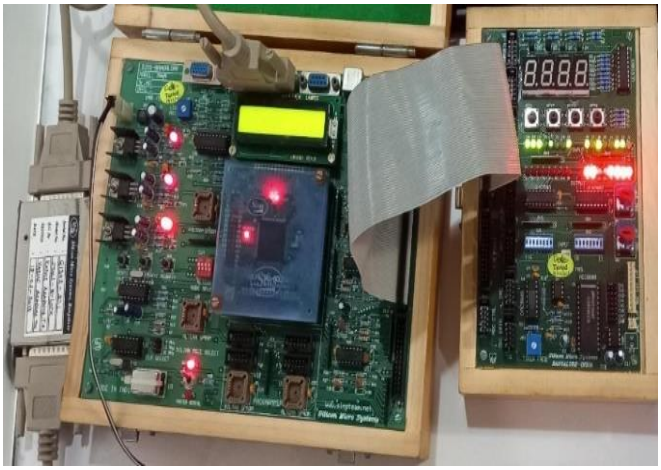

Fig.7. Hardware output using Acex 1K FPGA kit.

Hardware components: The list of hardware components used in this project are listed below:

1. Acex 1K FPGA kit: This is the main component that you will be using for your VLSI project. The kit consists of a development board that houses the FPGA chip, connectors for input/output, and various other components required for interfacing with the FPGA.

2. Computer/Laptop: You will require a computer/laptop with Quartus II (version 8.1) installed on it. Quartus II is a software tool that you will use to design and implement your VLSI project.

3. USB cable: You will need a USB cable to connect the FPGA development board to your computer/laptop.

4. Power supply: The FPGA development board requires a power supply to function power supply can be either a battery or a DC power adapter.

5. JTAG programmer: You will need a JTAG programmer to program the FPGA chip. The JTAG programmer allows you to download the bitstream generated by Quartus II onto the FPGA.

6. Oscilloscope/Logic analyser: Depending on the nature of your VLSI project, you may require an oscilloscope or logic analyser to measure and analyse the signals generated by the FPGA.

Once you have all the necessary hardware components, you can proceed with designing and implementing your VLSI project using Quartus II and Acex 1K FPGA kit.

VII.CONCLUSION

Error-efficiency in approximate computing refers to the ratio of the accuracy of the approximate results to the accuracy of the exact results obtained through traditional computing methods. It's important to note that error efficiency is not the only metric for evaluating the effectiveness of approximate computing techniques. Other metrics such as power consumption, speed, number of transistors used, and resource

usage should also be considered. In this paper, we have reduced the delay and complexity of the circuit by decreasing the number of transistors which are used in the above three methods using a 32-bit ripple carry adder. We compared the delay and number of transistors relative to the three method is for user preference of selecting either accurate 32-bit RCAs sum output or error efficient approximate 32-bit RCAs sum output.

REFERENCES

[1]. Shalini Singh, Pavan Kumar Pothula, Madhav Rao. "Design and Evaluation of On-chip DCT accelerators based on Novel Approximate Reverse Carry Propagate Adders",2022 IEEE

[2]. Neha Varshney, Greeshma Arya. "Design and Execution pf Enhanced Carry Increment Adder using Han-Carlson and Kogge-Stone adder Technique: Han-Carlson and Kogge-Stone adder is used to increase speed of adder circuitry",2019 3rdInternational conference on Electronics, Communication and Aerospace Technology(ICECA), 2019.

[3]. Weiqiang Liu,Jiahua Xu, Danye Wang, Chenghua Wang, Paolo Montuschi, Fabrizio Lombardi. "Design and Evaluation of Approximate Logarithmic Multipliers for Low Power Error-Tolerant Application", IEEE Transaction on Circuits and Systems I: Regular Papers, 2018.

[4]. Wenbin Xu, Sachin S. Saptnekar, Jiang Hu. "A Simple Yet Efficient Accuracy-Configurable Adder Design". IEEE Transaction on Very Large-Scale Integration (VLSI) Systems, 2018.

[5]. P. Balasubramanian, C.Dang,D.L. Maskell, K. Prasad. "Approximate ripple carry and carry lookahead adders - A comparative analysis", 2017 IEEE 30th International Conference on Microelectronics(MIEL), 2017.

[6]. I. Alam, K.T. Lau, "Approximate adder for low-power computations," Intl. Jour. Of Electronics Letters, 2017, vol.5, no.2, pp.158-165.

[7]. Bhavani Koyada, N. Meghana, Md. Omari Jaleel, Praneet Raj Jeripotula. "A comparative study on adders". 2017 International Conference on Wireless communications, Signal Processing and Networking(WiSPNET),2017.

[8]. P. Balasubramanian, "Asynchronous carry select adders", Engineering Science and Technology, an International Journal,2017.

[9]. Advances in Intelligent Systems and Computing, 2016

[10]. K. Roy, A. Raghunathan, "Approximate computing: an energy-efficient computing technique for error resilient applications," Proc. IEEE Computer Society AnnualSymp.on VLSI, 2015, pp.9-13.

[11]. T. Moreau, A. Sampson, L. Ceze, "Approximate computing:making mobile systems more efficient," IEEE Pervasive Computing, 2015, vol.14, no.2, pp.9-13.

[12]. M. Shafique, W. Ahmad, R. Hafiz, J. Henkel, "A low latency generic accuracy configurable adder," Proc. Design Automation Conf., 2015, pp.1-6.

[13]. Saranya, S., M. Sarumathi, B. Swathi, P. Victer Paul, S. Sampath Kumar, and T. Vengattaraman. "Dynamic Preclusion of Encroachment in Hadoop Distributed File System", Procedia Computer Science, 2015.

[14]. ZdenelVasicek, Lukas Sekannia. "Evolutionary Approach to Approximate Digital Circuits Design", IEEE Transcations on Evolutionary computation, 2015.

[15]. V. Gupta, D. Mohapatra, A. Raghunathan, K. Roy, "Low-power digital signal processing using approximate adders," IEEE Trans. on CAD of Integrated Circuits and Systems, 2013, vol.32, no.1, pp.124-137.

[16]. A.B. Kahng, S. Kang. "Accuracy-configurable adder for approximate arithmetic designs," Proc. Design Automation Conf., 2012, pp.820-825.

[17]. Synopsys SAED_EDK32/28_CORE Databook, 2012.

[18]. K. Gupta, N. Pandey, M. Gupta. "A novel active shunt-peaked MCML-based high speed four-bit Ripple-Carry adder",2010 International Conference on Computer and Communication Technology(ICCCT), 2010.

[19]. N. Shu, W.L. Goh, W. Shang, K.S. Yeo, S.H Kong, "Design of low-power high-speed truncation-error-tolerant adder and its application in digital signal processing." IEEE Trans. On VLSI Systems,2010,vol. 18, no. 8, pp. 1225-1229, Aug.2010.

[20]. P. Balasubramanian, N.E. Mastorakis, "A low power gate level full adder design," Proc. 3rd Intl. Conf. on Circuits, Systems, and Signals, Invited Paper, 2009, pp.246-484.

[21]LakshmiN.B.KirthiKrishnaAvinashJasonKrishnaV.ChakrapaniMuntimadu guLingamneniGeorgePalem. "Highly energy and performance efficient embedded computing through approximately correct arithmetic", Proceedings of the 2008 international conference on Compliers architecture and synthesis for embedded systems-CASES 08 CASES 08,2008.

[22]. A.P. Chandrakasan, S. Sheng, R.W. Brodersen, "Low power CMOS digital design." IEEE Jour, of Solid-State Circuits, 1992, vol.27, no.4, pp.473-484.

[23]. Merin Loukrakpam, MadhuchhandaChondhury. "Implementation of energy-efficient approximate multiplier with guaranteed worst case relative error", Microelectronics Journal, 2019.

[24]. U. Garlando, Q. Wang, O.V. Dobrovolskiy, A.V. Chumak, F. Riente. "Numerical Model for 32-bit Magnonic Ripple Carry Adder", IEEE Transactions on Emerging Topics in Computing, 2023.