International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 2 Volume No.9 March - April – 2025
DOI:10.46647/ijetms.2025.v09i02.005 ISSN: 2581-4621

LOW-LATENCY APPROXIMATE ADDER IN FPGA

Sowmiyaa P[1], Saranya P[2], Sabena M[3], Saranya R[4], Subhisha K[5]
[1]*Assistant Professor, Electronics and Communication Engineering, Muthayammal Engineering College, Namakkal, Tamilnadu.*
[2]*Assistant Professor, Electronics and Communication Engineering, Muthayammal Engineering College, Namakkal, Tamilnadu.*
[3]*UG - Electronics and Communication Engineering, Muthayammal Engineering College, Namakkal, Tamilnadu.*
[4]*UG - Electronics and Communication Engineering, Muthayammal Engineering College, Namakkal, Tamilnadu.*
[5]*UG - Electronics and Communication Engineering, Muthayammal Engineering College, Namakkal, Tamilnadu.*

Abstract
Approximate computing has gained significant attention for applications where absolute precision is not critical, such as image processing, machine learning, and signal processing. The proposed design divides the addition process into two stages: a main sub-adder for high-speed approximate computation and an error sub-adder for refining accuracy. By limiting carry propagation in the main sub-adder, the critical path delay is significantly reduced, achieving low latency. Simultaneously, the error sub-adder operates in parallel to correct errors, ensuring a balance between performance and precision. Experimental results demonstrate the design's superior trade-off between speed, accuracy, and energy efficiency compared to conventional adders. The proposed architecture is highly scalable and suitable for resource-constrained and performance-critical applications, such as real-time image processing and low-power machine learning accelerators.
Keywords: Low-latency, approximate adder, dual sub-adder architecture, error recovery mechanism, carry propagation, critical path delay, high-speed computation, energy efficiency.

1. INTRODUCTION
Gone are the days when huge computers made of vacuum tubes sat humming in entire dedicated rooms and could do about 360 multiplications of 10 digit numbers in a second. Though they were heralded as the fastest computing machines of that time, they surely don't stand a chance when compared to the modern day machines. Modern day computers are getting smaller, faster, and cheaper and more power efficient every progressing second. But what drove this change? The whole domain of computing ushered into a new dawn of electronic miniaturization with the advent of semiconductor transistor by Bardeen (1947-48) and then the Bipolar Transistor by Shockley (1949) in the Bell Laboratory. Since the invention of the first IC (Integrated Circuit) in the form of a Flip Flop by Jack Kilby in 1958, our ability to pack more and more transistors onto a single chip has doubled roughly every 18 months, in accordance with the Moore's Law. Such exponential development had never been seen in any other field and it still continues to be a major area of research work. The development of microelectronics spans a time which is even lesser than the average life expectancy of a human, and yet it has seen as many as four generations. Early 60's saw the low density fabrication processes classified under Small Scale Integration (SSI) in which transistor count was limited to about 10.

1.1. VLSI Design
VLSI chiefly comprises of Front End Design and Back End design these days. While front end design includes digital design using HDL, design verification through simulation and other

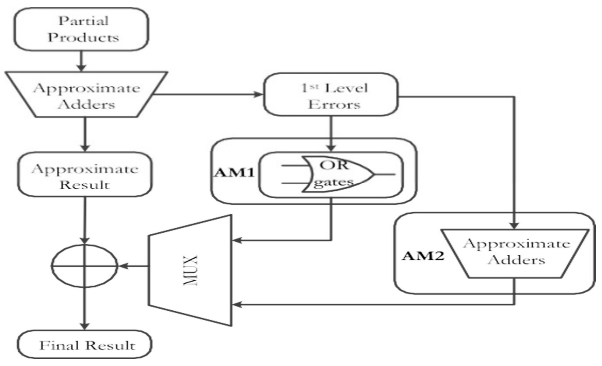
verification techniques, the design from gates and design for testability, backend design comprises of CMOS library design and its characterization. It also covers the physical design and fault simulation.

1.2. Architecture Definition

Basic specifications like Floating point units, which system to use, like RISC (Reduced Instruction Set Computer) or CISC (Complex Instruction Set Computer), number of ALU's cache size etc.

2. METHOD

The standard approach is to use carry chains only when the arithmetic operations are defined by high-level primitives (e.g., '+' operator) in a Register-Transfer Level (RTL) language. For × example, Verilog-to-Routing (VTR), the de facto academic tool flow for FPGA architectural and CAD research
uses such an approach. However, this is not always optimal nor possible. In some cases, the RTL code is not available because the design comes as a gate-level netlist. Additionally, suboptimal mapping decision can be made by assigning adders to carry chains before technology mapping (i.e., design step that transforms the gate-level description of the input into a network of LUTs). Luu et al. have recently shown that using carry chains provides an average speed up of approximately 15% for an area penalty of approximately 5%.



Existing architectures for speculative addition are all based on the assumption that operands have uniformly distributed bits, which is rarely verified in real applications. As a consequence, they may be disadvantageous for real-world workloads, although in principle faster than standard adders. To address this limitation, we introduce a new architecture based on an innovative technique for speculative global carry evaluation.

Approximate computing has emerged as a promising paradigm for applications where absolute accuracy is not critical, such as image processing, machine learning, and signal processing. low latency approximate adders aim to achieve high performance and energy efficiency by trading off a small amount of computational accuracy. This work presents a novel dual sub-adder-based approximate adder with an integrated error recovery mechanism to balance speed, accuracy, and energy efficiency.

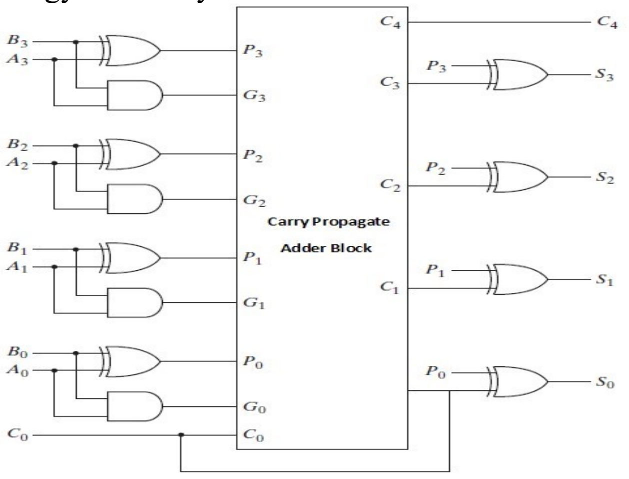


Fig 3.3 carry propagation adder

The addition and subtraction operations can be done using an Adder Subtractor circuit. The figure shows the logic diagram of a 4-bit Adder Subtractor circuit Binary Parallel Adder The circuit has a mode control signal M which determines if the circuit is to operate as an adder or a subtractor. So when M = 0, the output of XOR gate will be Bi ⊕ 0 = Bi. If the full adders receive the value of B, and the input carry C0 is 0, the circuit performs A plus B. When M = 1, the output of XOR gate will be Bi ⊕ 1 = Bi ’. If the full adders receive the value of B’, and the input carry C0 is 1, the circuit performs A plus 1’s complement of B plus 1, which is equal to A minus B

3. RESULTS AND DISCUSSION

3.1. Results

This work presents a low-latency approximate adder design based on a dual sub-adder architecture with an integrated error recovery mechanism, offering an effective balance between speed, accuracy, and energy efficiency

3.2. Discussion

The proposed adder demonstrates significant improvements in performance, making it ideal for applications like image processing, signal processing, and machine learning, where exact accuracy is not essential but high speed and energy efficiency are crucial. Its scalability and adaptability to various bit widths further extend its applicability to a wide range of resource constrained systems.

4. CONCLUSION

This work presents a low-latency approximate adder design based on a dual sub-adder architecture with an integrated error recovery mechanism, offering an effective balance between speed, accuracy, and energy efficiency. By dividing the addition process into a high-speed main sub-adder and an error sub-adder, the design achieves reduced latency by limiting carry propagation in the critical path. The error recovery mechanism enhances computational precision, mitigating the impact of approximation errors. The proposed adder demonstrates significant improvements in performance, making it ideal for applications like image processing, signal processing, and machine learning, where exact accuracy is not essential but high speed and energy efficiency are crucial. Its scalability and adaptability to various bit widths further extend its applicability to a wide range of resource constrained systems. Future enhancements can focus on optimizing the error correction mechanism and exploring adaptive approximation techniques to further improve accuracy and energy savings. This innovative approach to approximate addition provides a robust foundation for high-performance, low-power designs

5. ACKNOWLEDGEMENTS

Place Acknowledgments, including information on the source of any financial support received for the work being published. Place Acknowledgments, including information on the source of any financial support received for the work being published.

Journal reference style:

1. J. Han and M. Orshansky, “Approximate computing: An emerging paradigm for energy-efficient design,” in Proc. 18th IEEE Eur. Test Symp., May 2013, pp.1–6.
2. S.-L. Lu, “Speeding up processing with approximation circuits,” Com- puter,vol.37,no.3,pp.67–73,Mar.2004.
3. A. K. Verma, P. Brisk, and P. Ienne, “Variable latency speculative addition: A new paradigm for arithmetic circuit design,” in Proc. Design, Automat. Test Eur., Mar. 2008, pp.1250–1255.
4. N. Zhu, W. L. Goh, and K. S. Yeo, “An enhanced low-power high speed adder for error-tolerant application,” in Proc. 12th Int. Symp. Integer Circuits, Dec. 2009, pp.69–72.
5.H.R.Mahdiani,A.Ahmadi,S.M.Fakhraie,andC.Lucas,“Bio-inspired imprecise computational blocks for efficient VLSI implementation of soft-computing applications,” IEEE Trans. Circuits Syst. I, Reg. Systems, vol. 57, no. 4, pp.850–862, Apr. 2010. 6. V. Gupta, D. Mohapatra, S. P. Park, A. Raghunathan, and K. Roy, “IMPACT: IM Precise adders for low-power approximate computing,” in Proc. IEEE/ACM Int. Symp. Low Power Electron. Design, Aug. 2011, pp.409–414.