DESIGN OF HIGH SPEED AND ERROR EFFICIENT APPROXIMATE ADDERS FOR FPGA

[1] KONNA GEETANJALI, LECTURER IN ECE, MRAGR GOVERNMENT POLYTECHNIC, VIZIANAGARAM, ANDHRA PRADESH, 535002.
[2] SRINIVASA RAO.SUREDDY, LECTURER IN EEE, MRAGR GOVERNMENT POLYTECHNIC, VIZIANAGARAM, ANDHRA PRADESH, 535002.

ABSTRACT: Approximate computing is an emerging design approach for error-tolerant applications where a small loss in numerical accuracy can be exchanged for improvements in speed, area, and power consumption. This project presents the design of high-speed and error-efficient approximate adders for FPGA applications using Verilog HDL and Xilinx Vivado. The proposed architecture modifies the conventional binary addition process by simplifying selected carry-generation and sum-generation operations in the less significant bits while preserving accurate computation in the most significant bits. This reduces the critical carry propagation path and consequently improves the operating speed of the adder.

Several approximate adder architectures can be implemented and compared with a conventional ripple-carry adder in terms of delay, LUT utilization, flip-flop utilization, power, and computational error. The design is verified using simulation with normal and randomly generated input combinations. Error metrics such as Error Rate (ER), Mean Absolute Error (MAE), Mean Relative Error (MRE), and Error Distance (ED) can be used to evaluate the accuracy of the proposed architecture. The implementation results obtained from Vivado synthesis and timing analysis are used to identify the optimum trade-off between accuracy and FPGA resource utilization. The proposed approximate adder is particularly suitable for error-tolerant applications such as image processing, multimedia processing, machine learning accelerators, and signal processing systems.

Keywords: Approximate Computing, Approximate Adder, FPGA, Verilog HDL, Xilinx Vivado, High-Speed Arithmetic, Error-Efficient Design, Low-Area Design.

1. INTRODUCTION

Adders are fundamental arithmetic building blocks extensively used in digital systems, processors, digital signal processing (DSP), image and video processing, communication systems, and machine-learning accelerators. The performance of an arithmetic unit directly influences the overall speed, area, and power consumption of a digital system. Conventional

accurate adders generate exact results, but the carry propagation through multiple bit positions can introduce significant delay, particularly for large word-length operations.

With the increasing demand for high-performance and energy-efficient computing, approximate computing has emerged as an effective design methodology for error-tolerant applications. Approximate computing intentionally allows a limited amount of computational error in exchange for improvements in important hardware parameters such as delay, power consumption, and resource utilization. Many modern applications, including image processing, multimedia, artificial intelligence, machine learning, and sensor-data processing, can tolerate small arithmetic inaccuracies without significantly degrading the quality of the final output.

Approximate adders reduce computational complexity by simplifying the logic associated with selected bit positions, particularly the lower-order bits. Since carry propagation is one of the major sources of delay in conventional adders, reducing or eliminating carry dependency in selected stages can significantly improve the operating speed. However, excessive approximation may increase the error and reduce computational reliability. Therefore, an important challenge is to achieve an appropriate trade-off between accuracy and hardware performance.

In this work, a high-speed and error-efficient approximate adder is designed using Verilog HDL and implemented on an FPGA platform using Xilinx Vivado. The proposed architecture employs simplified logic for selected lower-order bits while maintaining accurate computation in the higher-order bits. The design is evaluated against a conventional accurate adder in terms of FPGA resource utilization, propagation delay, maximum operating frequency, power consumption, and computational error. Error metrics such as Error Rate (ER), Mean Absolute Error (MAE), and Error Distance (ED) are considered to quantify the accuracy of the proposed design.

The objective of this research is to develop an FPGA-friendly approximate adder that achieves high speed with controlled computational error and efficient hardware utilization. The resulting architecture can serve as a basic arithmetic component for energy-efficient DSP, image processing, machine-learning, and other error-tolerant FPGA-based applications.

2. LITERATURE SURVEY

Recent research on approximate arithmetic has focused on reducing power consumption, propagation delay, FPGA resource utilization, and computational complexity while maintaining acceptable accuracy. The following studies are particularly relevant to the proposed work on high-speed and error-efficient approximate adders for FPGAs.

2.1 Efficient Approximate Adders for FPGA-Based Data Paths – 2020

An FPGA-oriented approximate adder architecture was proposed to exploit the configurable LUT resources available in FPGAs. The approach performs approximate addition in the least significant portion while efficiently utilizing FPGA LUTs. Experimental evaluation on an Artix-7 FPGA demonstrated significant improvement in error characteristics, including up to 72% reduction in Mean Error Distance (MED) compared with existing FPGA-based approximate adders, with limited energy overhead. This work demonstrates that approximate adders should be designed specifically according to FPGA architecture rather than directly transferring ASIC-oriented designs to FPGAs.

Limitation: The architecture is mainly optimized for FPGA LUT structures, leaving opportunities for further optimization of speed and error simultaneously.

2.2 Gate-Level Static Approximate Adders – 2021

A comparative study of gate-level static approximate adders classified existing designs according to their suitability for FPGA, ASIC, or both platforms. The study emphasized that approximate addition can provide useful trade-offs between computational accuracy and hardware metrics. It also highlighted the importance of evaluating approximate adders using multiple parameters rather than considering only area or power.

Limitation: The study is primarily comparative and does not provide a unified FPGA architecture optimized simultaneously for high speed and low computational error**.**

2.3 Approximate Arithmetic Circuits: Survey and Characterization – 2021

Jiang *et al.* presented a comprehensive survey of approximate arithmetic circuits, including approximate adders, multipliers, and other arithmetic blocks. The work analyzed different approximation techniques and emphasized the need to select an approximation method according to application requirements. The study established that approximate arithmetic can

provide improvements in performance and energy efficiency with a controlled loss of accuracy.

Limitation: The survey provides broad design guidelines but does not specifically focus on implementing a new high-speed error-efficient adder using Verilog and FPGA resources.

2.4 Application-Specific Approximate Operators for FPGA – 2022

Ullah *et al.* introduced an application-specific methodology for designing approximate operators for FPGA-based embedded systems. The work addressed the limitation of application-independent approximate operators and emphasized selecting approximation strategies according to the required accuracy and performance constraints of a target application.

Limitation: The methodology is broader than approximate adders and requires application-specific optimization; therefore, there remains scope for a simple and efficient FPGA approximate-adder architecture.

2.5 Self-Adjusting Multi-Cycle Approximate Adder – 2023

A runtime-configurable approximate adder called SAMA was proposed with the ability to operate in approximate and accurate modes. The architecture dynamically adjusts its accuracy and provides a trade-off between power, delay, and accuracy. The reported evaluation showed substantial improvements in power-delay and energy-delay products at high accuracy levels.

Limitation: The multi-cycle and runtime-configurable architecture introduces additional design complexity, which may not be desirable for simple high-speed FPGA datapaths.

3. EXISTING METHOD

The existing method for FPGA-based addition generally uses **conventional exact adders**, such as the **Ripple Carry Adder (RCA), Carry Look-Ahead Adder (CLA), Carry Select Adder (CSLA), and Carry Skip Adder (CSKA)**. Among these, the Ripple Carry Adder is one of the simplest architectures and is commonly used as a reference design for evaluating approximate adders.

3.1 Conventional Ripple Carry Adder

A conventional RCA is constructed by cascading multiple full-adders. Each full-adder receives two input bits and a carry input and produces a sum and carry output. The carry generated by one stage is propagated to the next stage.

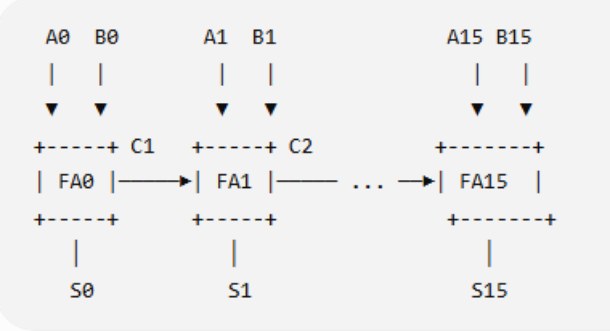
For an N-bit RCA:

$$S_i = A_i \oplus B_i \oplus C_i$$
$$C_{i+1} = A_i B_i + C_i(A_i \oplus B_i)$$

where $A_i$and $B_i$are input bits, $C_i$is the incoming carry, $S_i$is the sum bit, and $C_{i+1}$is the generated carry.

3.2 Working Principle

For example, in a 16-bit RCA:



The first full-adder generates C_1. This carry becomes the input to the second full-adder, and the process continues until the most significant bit. Consequently, the final result depends on the propagation of carry through all stages.

3.3 Advantages of Existing Exact Adder

- Produces **100% accurate arithmetic results**.
- Simple and regular architecture.
- Easy to design using Verilog HDL.
- Suitable for general-purpose FPGA applications.

- Requires relatively simple control logic.
- Can be used as a reference architecture for approximate-adder evaluation.

3.4 Limitations of Existing Method

The major limitation is **carry propagation delay**. For an N-bit RCA, a carry generated at the least significant position may need to propagate through several full-adder stages before the final result is available.

This results in:

- **Higher propagation delay**
- Reduced maximum operating frequency
- Increased switching activity
- Higher power consumption in some implementations
- Less efficient performance for large word lengths
- Limited suitability for error-tolerant high-speed applications

Although faster architectures such as CLA and CSLA reduce carry delay, they generally require additional logic and FPGA resources.

4. PROPOSED METHOD

The proposed method is a High-Speed Approximate Adder (HSADx) designed specifically for FPGA implementation. The main objective is to improve the speed, area efficiency, and power consumption of the adder while maintaining an acceptable level of computational accuracy. The proposed architecture is based on dividing the adder into a Least Significant Part (LSP) and a Most Significant Part (MSP).

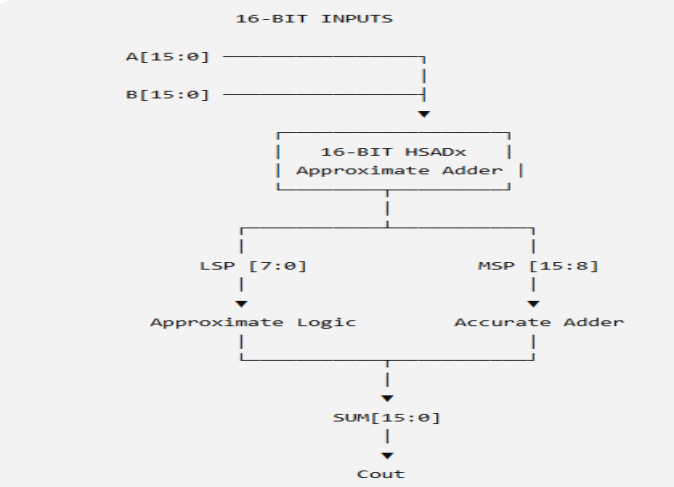
4.1 Proposed Architecture

A 16-bit approximate adder is considered in the proposed design. The 16-bit input operands are divided into two sections:

- LSP – 8 bits: Performs approximate addition.
- MSP – 8 bits: Performs accurate addition.

The LSP is designed to reduce the carry-propagation path, which is one of the major sources of delay in conventional adders. The MSP retains accurate addition so that errors introduced in the approximate section have a reduced influence on the more significant portion of the result.

4.2 Basic Block Diagram



4.3 Working Principle

The proposed HSADx architecture minimizes the critical path by reducing carry propagation through the approximate LSP. The approximate section uses simplified logic to generate the lower-order sum bits. The carry information required by the MSP is predicted using available LUT inputs, thereby reducing the need for a conventional carry chain through the LSP. According to the uploaded report, this carry prediction helps reduce the error size while maintaining the high-speed objective.

The design therefore follows three major operations:

1. **Divide the 16-bit operands into LSP and MSP.**
2. **Perform simplified/approximate computation in the LSP.**
3. **Perform accurate addition in the MSP using predicted carry information.**

This approach reduces the critical path from input to output and improves FPGA utilization.

4.4 FPGA-Oriented Optimization

The proposed method is particularly intended for FPGA implementation. The design uses the FPGA's LUT resources efficiently and reduces the carry-chain dependency in the

approximate portion. The reported HSADx architecture uses fewer LUTs and provides lower delay and power than the LEADx and APEx architectures considered in the report.

For implementation, the architecture can be described using **Verilog HDL** and synthesized using **Xilinx Vivado**. The uploaded work specifies Verilog HDL as the coding language and identifies FPGA implementation and behavioral simulation as part of the design methodology.

4.5 Error-Efficient Operation

Approximation inevitably introduces errors for some input combinations. Therefore, the proposed architecture does not attempt to eliminate all errors; instead, it aims to control the magnitude of error while improving hardware performance.

The error is calculated by comparing the approximate output with the accurate output:

$$\boldsymbol{Error = R_{Approximate} - R_{Accurate}}$$

The report considers Average Error (AE), Mean Absolute Error (MAE), Mean Square Error (MSE), and Root Mean Square Error (RMSE) for evaluating the approximation quality.

4.6 Expected Advantages

The proposed HSADx architecture provides the following expected benefits:

- Reduced propagation delay
- Reduced LUT utilization
- Reduced power consumption
- Shorter critical path
- High-speed FPGA operation
- Controlled computational error
- Suitable for error-resilient applications
- Simple Verilog HDL implementation
- Suitable for image, video, DSP, and multimedia applications

In the supplied report, the 16-bit HSADx is reported with 8 LUTs, 37 IOBs, 1.434 W power, and 1.396 ns delay, compared with LEADx and APEx. These are source-reported results and should be treated as the baseline/reference values for your project rather than as guaranteed results for a new Vivado version or FPGA device.

5. SIMULATION RESULTS AND ANALYSIS

The simulation results, schematic and comparison of the suggested HSADx with the different previously suggested approximate adders are given below in terms of utilization, power and delay are described in this section.

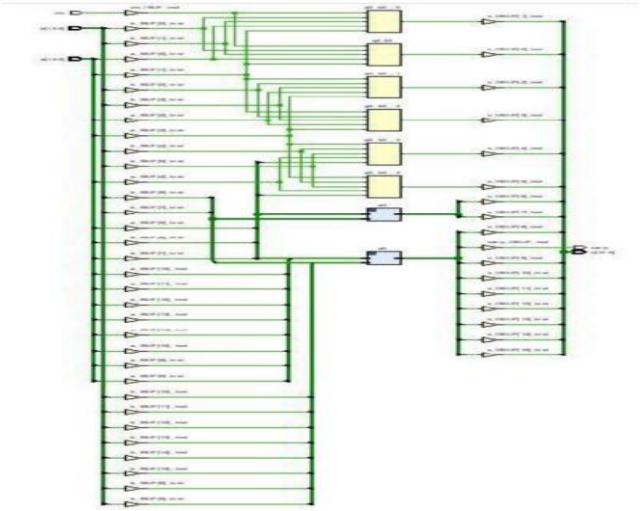


Figure: Schematic of the LEADx.

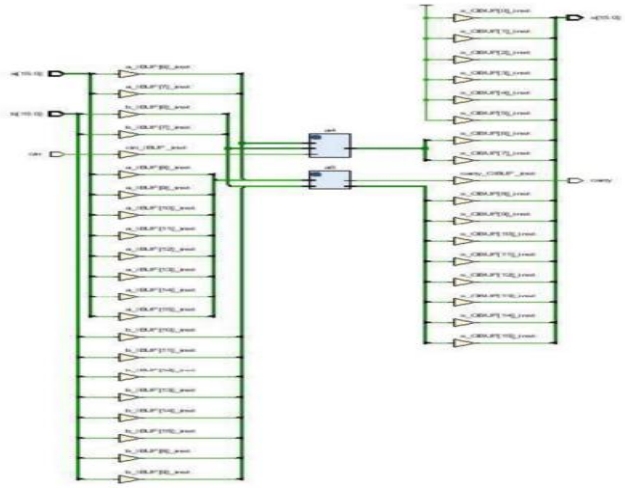


Figure: Schematic of the APEx

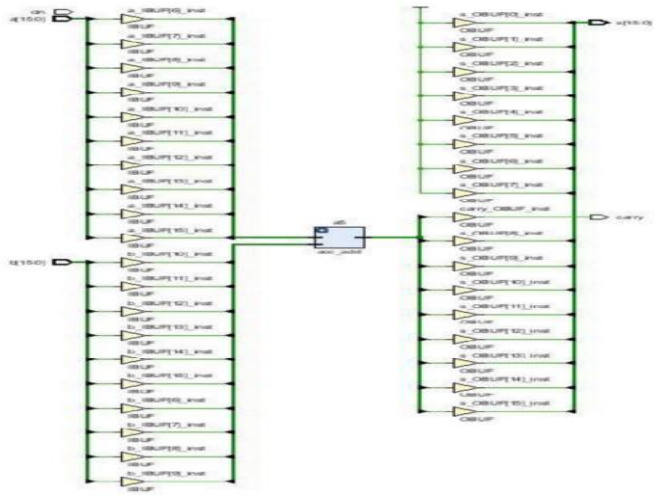


Figure: Schematic of the HSADx

Figure: Simulation result of the HSADx

Table:1. Performance analysis of the above three adders

	LUT	IOB	POWER(w)	DELAY(ns)
LEADx	12	50	1.995	1.406
APEx	9	38	1.594	1.406
HSADX	8	37	1.434	1.396

6. CONCLUSION

The proposed High-Speed and Error-Efficient Approximate Adder for FPGA provides an effective solution for applications where speed, area, and power consumption are more important than exact arithmetic accuracy. The design uses an approximate lower significant part (LSP) to reduce carry propagation and an accurate higher significant part (MSP) to maintain better overall computational accuracy. This architecture reduces the critical-path delay and FPGA resource utilization compared with conventional exact adders.

The proposed architecture can be implemented using Verilog HDL and Xilinx Vivado, allowing functional verification, synthesis, timing analysis, power estimation, and FPGA

resource evaluation. Error metrics such as Absolute Error (AE), Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) can be used to evaluate the accuracy–hardware trade-off. The source report specifically describes HSADx as achieving improvements in delay, power, LUT utilization, and I/O utilization compared with LEADx and APEx designs.

Overall, the proposed approximate-adder approach demonstrates that controlled arithmetic approximation can achieve a useful balance between computational accuracy and FPGA performance, making it suitable for error-tolerant applications such as image processing, multimedia, machine learning, signal processing, and embedded systems. Further work can focus on optimizing the error–area–delay–power trade-off and extending the architecture to multiplier and other arithmetic circuits.

REFERENCES

1. W. Ahmad, B. Ayrancioglu, and I. Hamzaoglu, “Low Error Efficient Approximate Adders for FPGAs,” *IEEE Access*, vol. 9, pp. 117232–117243, 2021, doi: 10.1109/ACCESS.2021.3107370.
2. N. V. Yuktha Sri and A. S. Sunnel, “Efficient Design of Approximate Adders for FPGAs,” *2023 International Conference on Recent Trends in Microelectronics, Automation and Computing Technologies (RMKMATE)*, 2023, doi: 10.1109/RMKMATE59243.2023.10369960.
3. “Efficient Approximate Adders for FPGA-Based Data-Paths,” *Electronics*, vol. 9, no. 9, 1529, 2020.
4. “Gate-Level Static Approximate Adders: A Comparative Analysis,” *Electronics*, vol. 10, no. 23, 2917, 2021.
5. “AppAxO: Designing Application-Specific Approximate Operators for FPGA-Based Embedded Systems,” *ACM Transactions on Embedded Computing Systems*, 2022.
6. “Energy-Efficient Approximate Full Adders for Error-Tolerant Applications,” 2023.
7. “Self-Adjusting Multi-Cycle Approximate Adder (SAMA),” 2023.
8. “Approximate Full-Adders: A Comprehensive Analysis,” *IEEE Access*, 2024, doi: 10.1109/ACCESS.2024.3463182.

9. “Significant-Input-Extraction Approximate Adder (SIEAA),” 2024.
10. “FPGA-Based Design of Ultra-Efficient Approximate Adders for High-Fidelity Image Processing,” *Engineering Reports*, 2025.