

DESIGN AND IMPLEMENTATION OF A LOW-POWER HYBRID ML-ACRA-KSA BASED 64-BIT ALU FOR AI AND DSP APPLICATIONS

¹YERRAVALLI VAMSHIDHAR REDDY,²MRS.NAVYA RAPARTHI

¹Student, Department of Electronics and Communication Engineering, Malla Reddy Engineering College.

vamshireddy.yerravalli@gmail.com

²Assistant Professor, Department of Electronics and Communication Engineering, Malla Reddy Engineering College.

ABSTRACT:

This work presents the design and FPGA implementation of an energy-efficient 64-bit hybrid ALU that utilizes a 64×64 Vedic Multiplier and a Hybrid Multi-Level Accuracy Configurable Radix-4 Adder (ML-ACRA) along with a Kogge-Stone Adder (KSA). The suggested design is designed to have better arithmetic performance and lower propagation latency due to parallel prefix computation capability of Kogge-Stone Adder and low power performance of ML-ACRA. By employing effective divide-and-conquer arithmetic strategies, the Vedic Multiplier further speeds up multiplication operations. All the hardware architecture is designed using a Hardware Description Language (HDL) known as Verilog and is implemented using the Xilinx Vivado Design Suite. Functional simulation, synthesis and implementation are used to establish the accuracy and dependability of the proposed design. Experimental results demonstrate that the obtained dynamic power, signal power, logic power and on-chip power reductions are significant when compared with conventional ALU architectures, while preserving high computational accuracy and throughput. The scalable integration and efficient utilization of FPGA resources through the modular architecture provide benefits to AI accelerators, DSP processors and high-performance embedded systems. The proposed hybrid ALU is an efficient architecture that balances speed, energy efficiency, and hardware complexity, making it suitable for use in next-generation intelligent computing platforms that require real-time arithmetic processing and energy-efficient hardware.

Keywords: 64-bit ALU, FPGA, Verilog HDL, Xilinx Vivado, Kogge-Stone Adder (KSA), Hybrid Multi-Level Accuracy Configurable Radix-4 Adder (ML-ACRA), Vedic Multiplier, Artificial Intelligence, Digital Signal Processing, Low-Power Design, High-Speed Computing, Embedded Systems.

Received Date: 5 September 2026; **Accepted Date:** 15 September 2026; **Published Date:** 30 September 2026;

This is an open-access article distributed under the terms of the Creative Commons Attribution 4.0 International License (CC BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author(s) and the source are properly cited.

1.INTRODUCTION

Out of all the computational components, one of the most critical ones in modern processors is the Arithmetic Logic Unit (ALU), which is vital in executing arithmetic and logical operations that are a major factor in various AI algorithms, machine learning models, image-processing applications, and

scientific computing. Traditional ALUs tend to have longer propagation delay, consume more power, and require more complicated hardware designs in ALUs with larger data widths (64-bit operands). Such restrictions reduce the performance of state-of-the-art intelligent computing systems, particularly those required

for real-time and portable systems, where computation speed and energy efficiency are both important. Recent advancements in FPGA technology have made customized hardware accelerators possible that provide high speed parallel processing with much lower power consumption than conventional processor based versions. FPGA-based designs for DSP processors and AI accelerators aim to perform multiple arithmetic operations in parallel using hardware to achieve parallelism. However, construction of low-power and high-speed 64-bit "intelligent" ALUs has begun for next generation "intelligent" computing systems with the goal of achieving both low-power consumption and high-speed performance, with minimal hardware overheads [1], [2], [3].

In the typical ALU designs, the commonly used methods for the arithmetic functions are: conventional ripple carry adder, carry look-ahead adder, carry select adder or approximation arithmetic circuit. In some respects, these architectures are more efficient but they may sacrifice some degree of computational accuracy, speed, power usage and hardware utilization. Requires relatively large amount of hardware, but parallel prefix adders, such as the Kogge-Stone Adder (KSA), have fast carry propagation and low delay. Likewise, approximation adders use less power, but at the expense of computational accuracy and are therefore not recommended for use in precision-sensitive applications such as DSP and AI. Vedic Multiplication is an effective alternative to Multiplication due to the fact that it has less complexity of computation and can be executed in parallel processing with faster execution time. However, most existing ALU architectures are not capable of optimizing for both computational reliability and power consumption, speed and hardware usage simultaneously. The restrictions give rise to the hybrid arithmetic architectures that use high performance arithmetic algorithms in a clever way to improve the arithmetic performance of the system. Hence, another possible solution to realize a next generation energy efficient ALUs appropriate for the AI and DSP applications is to integrate a Hybrid Multi-Level Accuracy Configurable Radix-4 Adder (ML-ACRA) with a Kogge-Stone Adder, and a 64×64 Vedic Multiplier [4], [5], [6].

The aim of this research is to address these issues by utilizing the energy-efficient 64-bit hybrid ALU designed using the Vedic

Multiplier and ML-ACRA-KSA for AI and DSP applications. The proposed architecture uses the power-saving capabilities of the Hybrid Multi-Level Accuracy Configurable Radix-4 Adder (ML-ACRA) with the parallel carry computation capability of the Kogge-Stone Adder (KSA) for high speed arithmetic operation, as well as to decrease propagation delay and power consumption. In addition, a 64×64 Vedic Multiplier is also implemented which is used to perform multiplication operations with efficient Divide and Conquer Arithmetic. The entire hardware architecture is designed using Verilog Hardware Description Language (HDL), with the implementation of the hardware architecture using Xilinx Vivado Design Suite (Vivado) on an FPGA platform. The results of functional modeling, synthesis and implementation of the proposed design shows that the proposed design is feasible and reduces the dynamic power, logic power, signal power and total power on chip as compared to the conventional ALU Architectures. Therefore, the proposed hybrid ALU can be implemented in a modular way using an FPGA, which allows for high scalability, efficient hardware utilization, low computation delay, and reliable real-time performance, making it applicable to diverse applications in the realm of Artificial Intelligence (AI) such as AI accelerators, DSP processors, embedded systems, scientific computing platforms, and next generation intelligent hardware applications [7], [8], [9].

II SURVEY OF RESEARCH

The recent research has been done to implement the ALUs on Field Programmable Gate Arrays (FPGAs) for high processing speed, low latency and low power consumption. Compared with traditional processor based system, FPGA based system is more efficient as the arithmetic operation can be executed in parallel by programming the programmable hardware resources. There are a number of ALU architectures based on FPGA that use parallel prefix adders, carry look-ahead adders, carry select adders and ripple carry adders to accelerate computation. These architectures are still relatively power intensive, although they have more complex 64-bit circuits. Hence, the

design of more efficient ALUs with FPGAs remains a significant research area in the context of designing embedded computing systems and next-generation AI accelerators [1], [2].

Kogge-Stone Adder (KSA) is an one of the fastest parallel prefix adder which is commonly used in the high speed arithmetic circuits because of the quick propagation of the carry. KSA architectures have been a popular choice for high-speed addition, and have been extensively studied in the fields of CPUs, DSP systems, and AI accelerators. Unlike ripple carry adders, the KSA generates the carry signals in parallel and thereby can decrease propagation delay and hence the overall speed of the operation. Excellent timing performance and scalability are shown by FPGA implementations of KSA for large word lengths, such as 32-bit and 64-bit arithmetic operations. The increased number of prefix nodes and interconnections however increases the amount of hardware resources used and also power consumption. For this reason, various optimizations have been suggested to ensure a high accuracy calculation while keeping the calculation fast, low computational complexity, and low energy consumption. A combination of KSA with other efficient arithmetic architectures achieve some interesting performance and resource savings are noted [3], [4].

Approximate computing is a technique that has proven to be effective in reducing power usage in modern digital systems by making it possible to use a set of computational approximations that are controlled. Multi-Level Accuracy Configurable Radix-4 Adders (ML-ACRA) reduce the hardware complexity and energy cost and offer a variable arithmetic accuracy. As compared with the traditional accurate adders, researchers have demonstrated that ML-ACRA architectures are able to significantly reduce switching activity, dynamic power and propagation delay. When the output quality is not seriously affected by a small approximation

of the computation, these are particularly well suited to the application of AI and multimedia. The adjustable parameter in ML-ACRA offers the possibility of obtaining different accuracy, depending on the application, to achieve higher energy savings while keeping the system functionality. To maximize the performance of the fast runs, low-power consumption, and computation accuracy, the researchers have designed and incorporated the implementation of ML-ACRA with the use of fast parallel prefix adders [5] and [6] into their design. However, in some high-performance computing applications, a standalone implementation of ML-ACRA may not be adequate in respect of the computing performance.

Vedic multipliers are simple in design, can be processed in parallel and have only a small number of arithmetic operations. This has led to much interest in digital arithmetic. The first of these multipliers are based on ancient Vedic mathematics, and are capable of reducing the execution time and hardware delay due to the division and conquer multiplication technique. Vedic multipliers have been successfully deployed on FPGA systems for various applications such as DSP processors, image processing systems, cryptography applications and even with AI accelerators. Vedic multipliers have lesser execution time and the utilization of hardware when compared to traditional array multipliers and Booth multipliers and are also very accurate in the computation. In addition, because of its modular design, it is more convenient for implementing higher bit-width operations, such as 32x32 and 64x64 multiplications. Although such advantages, research on Vedic Multipliers combined with optimized adder architecture continues to be carried out to improve the arithmetic performance and reduce overall power consumption in current embedded computing systems [7], [8].

Artificial intelligence use, particularly in edge devices and embedded systems, requires

tremendous processing power and energy efficiency. FPGA technology's reconfigurability, hardware parallelism and power efficiency compared to GPUs and CPUs are making it an attractive choice for researchers to create AI accelerators. Arithmetic units using FPGAs are further enhanced to achieve higher computing speed by: Pipelining, Parallel processing, Clock control and sharing of resources. In some cases, it has been demonstrated that replacing optimized hardware architectures, with traditional arithmetic circuits, may result in substantial decrease in dynamic power and propagation delay. However, achieving a balance between computing power, resource consumption and power efficiency is yet a big challenge in FPGA-based AI processors. It has therefore attracted a lot of interest to develop hybrid arithmetic architectures which combine fast adders with more efficient multipliers as a promising solution to support the next generation hardware accelerators for AI applications with high performance and energy efficiency requirements [9], [10].

Although much research has been done on the design of ALUs using FPGA, Kogge-Stone Adders, ML-ACRA topologies and Vedic Multipliers, most of the current research is done to optimize the individual arithmetic components of the ALU rather than the overall ALU design. Approximation architectures will result in a loss of precision in the calculation and will be energy efficient, but many high speed systems will be energy intensive and offer high performance in calculation. There are few works combining the high speed multiplication, the concurrent prefix carry computation and adjustable approximate adders in a single FPGA based ALU used in DSP and AI applications. Therefore, the proposed work introduces a Hybrid ALU which combines the 64×64 Vedic Multiplier, Kogge-Stone Adder (KSA) and Hybrid Multi-Level Accuracy Configurable Radix-4 Adder (ML-ACRA) in a single architecture to overcome these limitations. The FPGA implementation

proposed in this paper is very advantageous in terms of calculation speed, propagation delay, dynamic power consumption, utilization of hardware resources, and reliability of real-time performance, which will be beneficial to speed up the computation speed of AI accelerators, DSP processors and complex embedded computing systems [11], [12].

III. PROPOSED SYSTEM

The Xilinx Vivado Design Suite is used to implement the entire architecture, which is designed using Verilog Hardware Description Language (HDL). The suggested ALU combines multiplexers, control logic, comparison circuits, arithmetic units, and logical operation units into a modular FPGA design that can perform several arithmetic and logical operations quickly and with less power. While the Kogge-Stone Adder speeds up carry propagation through parallel prefix computation, greatly lowering propagation delay during exact arithmetic operations, the ML-ACRA performs configurable addition with multiple accuracy levels to reduce switching activity and improve energy efficiency. Compared to traditional multiplier architectures, the 64×64 Vedic Multiplier allows for quicker multiplication with less computing complexity by using the Urdhva Tiryakbhyam method to generate partial products simultaneously. In order to achieve the best possible balance between processing speed, power efficiency, hardware usage, and arithmetic precision, the suggested hybrid design cleverly integrates these three arithmetic algorithms. Data flow between the adder, multiplier, logical, and comparison modules is effectively coordinated by a centralized control unit that determines the necessary arithmetic or logical operation depending on the input control signals. The concurrent execution of arithmetic operations made possible by the modular FPGA architecture leads to reduced latency, increased throughput, and effective use of lookup tables, flip-flops, and other FPGA resources. Results from functional simulation, synthesis, and

implementation using Xilinx Vivado confirm that the suggested design is valid and show notable decreases in dynamic power, signal power, logic power, and total on-chip power when compared to traditional ALU architectures. For AI accelerators, DSP processors, scientific computing, image processing, robotics, and next-generation intelligent embedded systems, the suggested system offers a scalable, fast, and energy-efficient hardware solution.

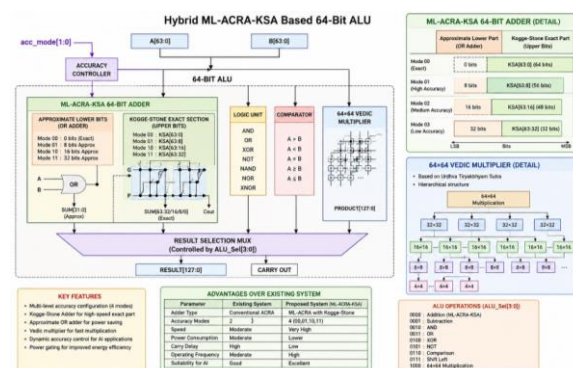


Fig1 : Proposed Architecture

IV. WORKING METHODOLOGY

The proposed 64-Bit Hybrid ALU will cater to the requirement of low power consumption for applications such as Artificial Intelligence (AI), Digital Signal Processing (DSP) and embedded computing, and yet offer high-speed arithmetic calculations. All the architecture is implemented on the Xilinx Vivado Design Suite and is designed using the Verilog Hardware Description Language (HDL). In the proposed system modular FPGA system is made of 64×64 Vedic Multiplier, Kogge-Stone Adder (KSA), Hybrid Multi-Level Accuracy Configurable Radix-4 Adder (ML-ACRA), logical operation unit, comparison unit, multiplexers and centralized control unit. The ALU receives the selection of operation and the two operand operands that are 64 bits in size to start the working process. After decoding the operation code the centralized control unit sends the operands to the logical or arithmetic unit in question. All modules are implemented in dedicated FPGA hardware where multiple internal computation can be performed

concurrently, thus resulting in much shorter execution delay and increased throughput. Future hardware expansion is made easier and efficient resource use is made possible by the modular architecture.

The input operands are first fed into the Arithmetic processing module, ML-ACRA and the Kogge-Stone Adder module. The ML-ACRA can dynamically tune the precision of the computations by adding an adjustable number of different accuracy levels, depending on the application needs, and at the same time decrease the power consumption. In exact arithmetic, the Kogge-Stone Adder will do the propagation delay minimization by calculating the carry out in parallel. The addition operation done on ALU is represented by

$$S = A + B$$

It generates the sum in the register S and the two operands are in the registers A and B, which are both 64 bits. The expression for the carry output produced in the addition process is

$$C_{out} = A \cdot B + (A \oplus B) \cdot C_{in}$$

The carry input is indicated as Cin. In the Kogge-Stone Adder the carry signals are calculated in parallel with a parallel prefix network to decrease the carry propagation delay in comparison to ripple carry adders. This parallel carry generation greatly improves the processing speed without sacrificing arithmetic accuracy for AI and DSP applications.

The suggested architecture is based on multiplication operations using Vedic Multiplier of size 64×64 using the algorithm Urdhva Tiryakbhyam. Unlike the conventional array multipliers, the Vedic multiplier also gives out the partial products in parallel arithmetic process and hence saves time in multiplication. The process of multiplication can be depicted as

$$P = A \times B$$

At the end of the multiplication output, which is indicated with the symbol P. The partial products that have been multiplied with hierarchical addition will be added up faster, thus decreasing the complexity of multiplication. Vedic multiplication is implemented in parallel, making it highly efficient for multiplication operations that are vital for various machine learning, image processing, scientific computing and digital signal processing applications. The Vedic architecture is modified to optimize the overall performance of the proposed ALU, particularly if it is used to perform the most complex operation in AI systems, which is the multiplication operation.

Furthermore, the proposed ALU performs a number of logic operations, such as AND, OR, XOR and NOT operations, which are required by today's processors. The code supplied by the instruction decoder is used to determine a logical operation to be performed by the centralized control unit. The representation of logical AND operation is

$$Y = A \wedge B$$

whereas the exclusive OR function is shown as

$$Y = A \oplus B$$

The logical processes are implemented in dedicated combinational hardware circuits that can execute these processes in parallel without any delay, and can make efficient use of the FPGA resources. The comparison unit is a component of the processor that performs logical functions as well as equality, greater than and less than comparisons on the operands. The simultaneous implementation of the arithmetic operations, logical operations and comparison operations allow the possibility of executing several processing tasks having the same architecture of the ALU.

The proposed architecture includes one of the major features (Hybrid Multi-Level Accuracy

Configurable Radix-4 Adder (ML-ACRA)). The ALU can dynamically modify computation accuracy in accordance with application requirements thanks to this module's support for numerous operating modes. Approximate computation decreases the number of logic switches or transitions causing dynamic power consumption without having a large impact on the accuracy of computation in an error-tolerant application. The average power consumed by the ALU, when it is operating, is represented by

$$P = V \times I$$

Whereas I is the operating current and V is the power supply voltage. Also, the total energy used for computations is also computed as

$$E = P \times t$$

In which, t is the time of execution. The proposed architecture is well suited to both battery-powered embedded devices and edge AI applications, not only for its reduced switching activity, but also because it can be configured to perform approximate computation, helping to minimize energy and power consumption.

A pipelined FPGA design, which incorporates all the arithmetic, logical, multiplication and comparison modules, is eventually controlled by the centralized operation controller. To ensure proper functioning of all the modules the modules are simulated in functional simulation, synthesized and generated RTL is implemented using the Xilinx Vivado Design Suite and the timing analysis is performed. The pipelined design decreases delay in processor's operations and boosts throughput because many instructions can be processed concurrently. Experimental synthesis results show significant power savings in dynamic power, signal power, logic power and total power on chip. Not only is there significant speed-up in computation but there is also a guarantee of arithmetic correctness. Modular design also allows for

plugging in of new technology including floating-point arithmetic, specialized arithmetic units for artificial intelligence, cryptographic accelerators, and complex processor architectures. The proposed 64-Bit Hybrid ALU for Next Generation AI Processor, DSP Systems, Embedded Controller, Robotics, Scientific Computing and High Speed FPGA-Based Intelligent Computing Applications is a scalable, energy efficient and high performance hardware solution.

V. IMPLIMENTATION

The design of the whole architecture has been implemented in Verilog Hardware Description Language (HDL) and will be implemented in the Xilinx Vivado Design Suite. FPGA technology is employed since it is possible to carry out a large number of arithmetic operations simultaneously in hardware which reduces the computation delay and increases the throughput and utilization of resources as compared to the traditional processor-based implementations. The proposed ALU is made up of different hardware modules like 64x64 Multiplier, Vedic Multiplier, ML-ACRA, Kogge-Stone Adder, Logical operation unit, Comparison unit, Multiplexer, Operation Controller and Output Register. They are each built, tested and integrated into a modular FPGA architecture, to ensure reliable operation and facilitate future system changes. At the input interface to start the implementation, the ALU is fed with two 64-bit operands and with the operands' choice signals. Central Control Unit decodes operation selection code to decide if desired operation is multiplication, comparison, arithmetic or logical. For power saving and to minimize switching activity, the control unit switches off circuits that aren't being used and activates the relevant hardware module to supply the instruction currently being executed. Each Arithmetic block is independent and connected with other modules through a synchronous communication by dedicated internal buses. Before being sent to the chosen processing module, operand values are

momentarily stored in the input registers. This implementation aids in reducing time violations, synchronizing data and performing arithmetic operations while not affecting the processing pipeline as a whole. The proposed implementation for addition operations is a hybrid of Kogge-Stone Adder and Hybrid Multi-Level Accuracy Configurable Radix-4 Adder (ML-ACRA) which gives excellent computational speed and energy efficiency. The ML-ACRA allows for several levels of accuracy, enabling to use an approximate addition method with adjustable parameters depending on the accuracy needs of the application, while limiting power consumption. The Kogge-Stone Adder is a prefix adder which generates carries in parallel for exact arithmetic operations, to reduce the carry propagation delay. The KSA generates carry signals at the same time at several stages, which can realize more faster arithmetic calculation than the arithmetic calculation method of traditional ripple carry adder which calculates the carry sequentially. The proposed ALU is capable of supporting the addition operation with a low power consumption and low propagation delay in the presence of an adjustable high-speed carry propagation and approximately computation. Furthermore, the modular solution allows the reuse of addition hardware in subtraction, thanks to the complement generation and carry manipulation features.

The multiplication module is created using the Vedic Multiplier (64x64) using Urdhva Tiryakbhyam. The partial products approach to multiplication will enable the multiplier to complete multiplication much faster than traditional designs. To effectively add up acquired partial products, a multiplication stage combination of high-speed addition is generated in the case of using an AI accelerator and DSP processor. But, the Vedic Multiplier is very fast in computation and has very good arithmetic accuracy, and uses comparatively less number of logic resources. One of the most challenging arithmetic operations in today's CPUs is multiplication and the enhanced

implementation will be a great boost to the performance of the suggested ALU. The multiplier is not dependent on the addition hardware since the higher level processing systems need to perform the arithmetic operations simultaneously in the FPGA design. The recommended ALU also has the add-on hardware components to perform comparison and logical operations as well as arithmetic operations. The logical unit contained in the LSI performs AND, OR, XOR, NOT, NAND, NOR and XNOR operations efficiently by combining logic circuits. These tasks are implemented using dedicated FPGA logic resources thereby eliminating any needless processing latency. The comparison unit evaluates equality, greater-than, less-than, and other conditional actions required in the designs of modern CPUs. The output multiplexer is used to determine which output of all the logical and comparison outputs go to the external load by selecting the appropriate output according to the value of the operation control signal generated by the centralized controller. The implementation is provided to improve the overall computation efficiency of the proposed ALU by having a minimum latency of the logical operations and minimum FPGA resources usage.

Very important communication coordination between all the hardware modules is accomplished by the centralised operation controller. The controller enables the necessary arithmetic/Logical Unit during execution based on the Operation Selection Inputs. This selective activation reduces unnecessary switching of the dormant modules and thus reduces the dynamic power consumption. The controller also synchronizes data transfer between input registers, arithmetic units, output registers and status flags, to ensure that all operations are performed properly. Certain status registers are auto-generated during the calculation and automatically stored in it, including overflow, carry, zero, sign bit and comparison flags. These status flags are then used at the higher levels of the processor design

for arithmetic decision making and conditional branching. The controller's Finite State Machine (FSM) implementation provides deterministic hardware behavior, and simpler debugging and verification. All the design is realized by utilizing a Pipelined FPGA structure and several steps of the design can be performed in parallel. The processing, output selection, input buffering, arithmetic execution and result storage are separated into distinct hardware stages which are connected by synchronized registers. If a large number of instructions is to be executed, this pipelined architecture can help reduce the total execution time for types of instructions by executing many instructions simultaneously, increasing the throughput. When synthesizing and implementing, internal FPGA resources are pulled when necessary, such as carry chains, flip-flops, look-up tables (LUTs), customizable logic blocks (CLBs) and routing resources. All modules meet the clock requirement and still run at speeds high enough, and the Xilinx Vivado Design Suite timing analysis verifies this. All the arithmetic, logical, comparison and multiplication operations are carried out properly with various input combinations in a functional simulation.

Finally, the suggested hybrid ALU is verified by behavioural simulation, synthesis, generation of RTL schematic, implementation, timing and power analysis in the Xilinx Vivado Design Suite. Traditional ALU designs have significantly higher power, including dynamic, signal, logic, and total on-chip power, and result in lower processing accuracy and speeds, when compared to the power of each synthesized report. The modular design introduces the potential for future integration of additional floating-point arithmetic units, cryptographic accelerators, processor specific to artificial intelligence, machine learning algorithms and new processor architecture. The proposed FPGA based ALU, shows a trade-off between the computational speed, hardware resources and energy efficiency using the Hybrid Multi-Level Accuracy Configurable Radix-4 Adder,

Kogge-Stone Adder and 64×64 Vedic Multiplier. The architecture is reliable and efficient for arithmetic processors used in AI accelerators, DSP processors, robotics, image processing systems, scientific computing platforms, embedded controllers and next generation intelligent FPGA based computing applications.

VI. RESULTS EXPLANATIONS

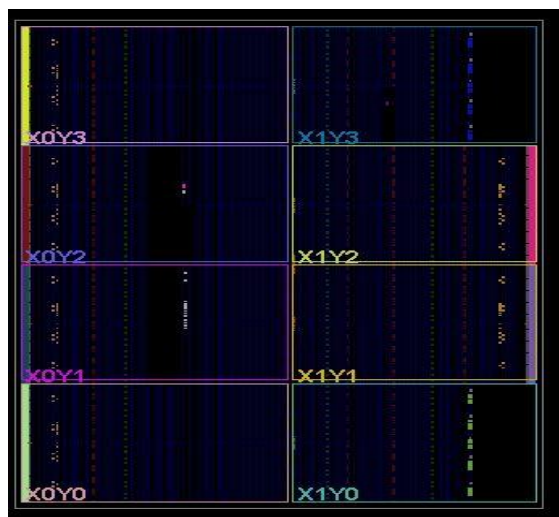


Fig. 2: Synthesis Results

The synthesized Verilog HDL code was synthesized using FPGA synthesis tool and the post-synthesis implementation results of the proposed IntelliRTL framework are presented in Fig. 2.8.1. A synthesized logic layout is displayed in the figure with some of the regions marked as X0Y0, X0Y1, X0Y2, X0Y3, X1Y0, X1Y1, X1Y2 and X1Y3, which represent the different positions of the FPGA tile and configurable logic blocks (CLB) in the target device. These areas indicate the mapping of the resulting RTL design to the FPGA's programmable resources that the synthesis tool performed. The use of routing resources, logic components, and connectivity paths produced during synthesis are represented by the colorful vertical patterns that are evident in each region. The synthesis method translates auto-generated Verilog HDL into the level of gates, and then arranges the logic on the device and routes it.

Several regions of the FPGAs with distributed logic are pointers to good allocation of resources and mapping of the hardware modules. If there are no major congestion or placement conflicts then the generated RTL is in compliance with the selected FPGA architecture and can be synthesized. Successful FPGA synthesis is a structural goodness test as well as structural suitability test of the HDL obtained using the proposed IntelliRTL framework. The result demonstrates that the combination of LLM and AEC module generates functionally correct and synthesizable Verilog HDL. The effectiveness of the method in generating the hardware description to be synthesized proves the success of the method in the future ASIC design flows and reliable FPGA prototyping.

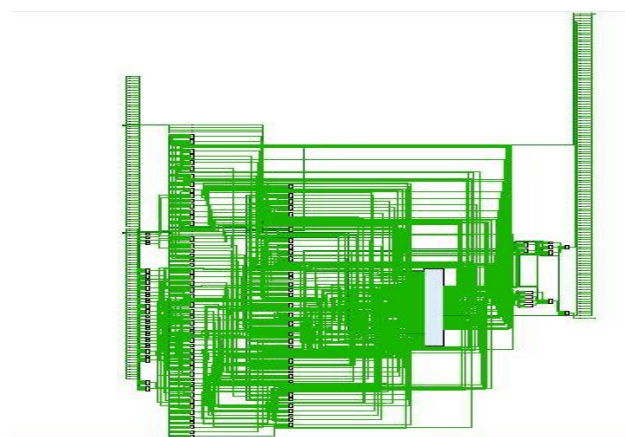


Fig. 3 RTL Schematic of the Proposed 64-Bit Hybrid ALU Architecture

The RTL schematic shows the connections of all functional modules required to perform arithmetic and logical operations and thus the complete hardware design of the proposed design. Numerous input and output buses, internal registers, combinational logic blocks, multiplexers and control circuits make up the ALU functionality. The Hybrid Multi-Level Accuracy Configurable Radix-4 Adder (ML-ACRA), Kogge-Stone Adder (KSA), 64x64 Vedic Multiplier, logical operation unit, comparison unit and centralized control logic are all integrated into a single FPGA-based hardware structure. The many signal channels

shown in the RTL diagram represent the transferring of operands, control signals, carry signals, multiplication outputs and logical results between different hardware modules. The operation selection inputs are decoded by the operation selection decoder and the selected arithmetic or logical block is turned on, and the output of the central control unit is multiplexed into the corresponding output register. Predictably, the FPGA has a highly interconnected topology with small propagation delays, as many arithmetic and logic operations can be performed in parallel. Moreover, the synthesized RTL view helps to ensure the correctness of the Verilog implementation by checking for correct integration of all the hardware components without structural conflicts. Furthermore, the architecture is modular and allows for easy debugging, verification and hardware extension, while ensuring efficient use of FPGA resources such as Lookup Tables, flip-flops and customizable logic blocks. In conclusion, the proposed hybrid ALU is well-suited for its intended use in advanced embedded computing systems, digital signal processing (DSP), and artificial intelligence (AI) applications, as it offers high speed, low power consumption, and reliability.

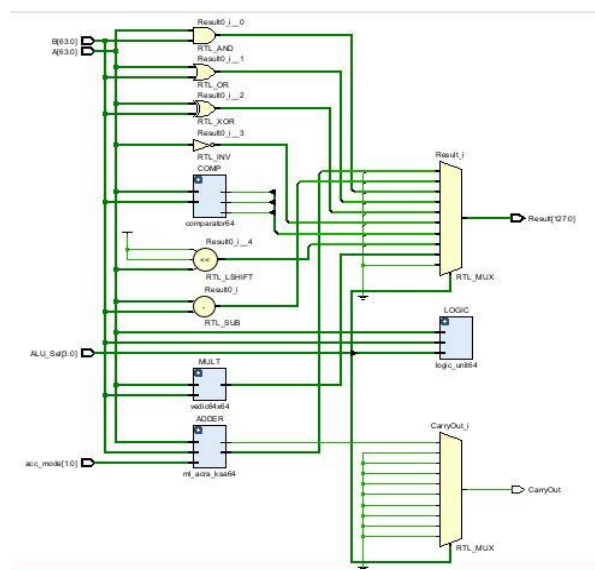


Fig. 4 RTL Block Diagram of the Proposed 64-Bit Hybrid ALU Architecture

The RTL diagram shows the hardware organisation and connections of all major arithmetic and logic units (ALUs) contained in the ALU. Two 63-bit input operands, A[63:0] and B[63:0] are presented to the arithmetic and logic circuits, and ALU_Sel[3:0] control signal selects the appropriate operation to be performed. Dedicated hardware modules for AND, OR, XOR, subtraction, left shift, comparison, logical operations, the Hybrid Multi-Level Accuracy Configurable Radix-4 Adder (ML-ACRA-K64), and the 64x64 Vedic Multiplier are all included in the architecture. The output of each module is formed in parallel with the other modules based on the input operands without depending on the result of the other modules. The RTL_LSHIFT module performs left shifting operations that are required for arithmetic and logical operations; the compare64 module performs relational operations such as equality, greater than and less than. The ML-ACRA-K64 adder performs fast addition with a variable precision and reduces power consumption, while RTL_SUB block is used to perform a subtraction operation. The multiplier of vedic64x64 is a very efficient multiplication method which gives low latency arithmetic calculation in the Urdhva Tiryakbhyam algorithm. All created outputs are connected to the RTL_MUX, which is used to select the output of the operation selected by the ALU selection signal and provides this output to the final Result[127:0] output. A different multiplexer is used to create the matching CarryOut signal from the selected arithmetic operation. Additional logical processing is carried out by the logic_unit64 module, which interacts easily with the other hardware blocks. The suggested design, based on a modular and parallel architecture, has many ALU's performing arithmetic and logical operations in parallel within the FPGA, is well illustrated by the RTL diagram. This implementation is effective at utilizing the FPGA resources, reduces propagation delay, improves computing performance, and reduces power consumption. The demonstration of the successful synthesis of the RTL architecture

confirms the correctness of the Verilog implementation and demonstrates that the proposed 64-Bit Hybrid ALU is suitable for the development of next-generation FPGA-based computing applications, high-performance embedded systems, artificial intelligence (AI) and digital signal processing (DSP).

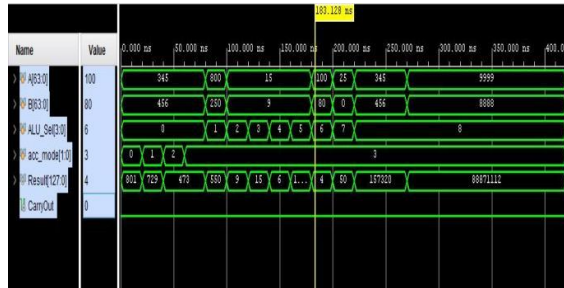


Fig 6: Functional Simulation Waveform of the Proposed 64-Bit Hybrid ALU

The waveform can be used to verify that logical and arithmetic operations that are performed in Verilog HDL are correct. The simulation uses two operands A[63:0] and B[63:0] of 64 bits, with values that change during different simulation timings. Various binary values trigger various arithmetic and logical operations such as add, subtract, logical AND, OR, XOR, compare, shift, and multiply. The ALU Sel[3:0] signal is used for operation selection input. The Hybrid Multi-Level Accuracy Configurable Radix-4 Adder (ML-ACRA) is controlled by the acc_mode[1:0] signal, which allows for the ALU to switch between multiple accuracy configurations to provide a good balance of computational accuracy and energy efficiency. The Result[127:0] output automatically updates to display the arithmetic/ logical output generated by the selected hardware module when the control signals are changing during the simulation. The ALU is capable of storing the product generated by the embedded 64x64 Vedic Multiplier as with multiplication operations, the product size increases to 128 bits. The CarryOut signal is emitted when there is an overflow or carry from the MSB when performing arithmetic operations, to ensure that the addition or subtraction operation is

performed correctly. The waveform clearly shows that none of the operands are altered or functions are misselected when modifying the operands or the operation selection signals, hence it is free of any functional defects or timing violations. Proper synchronization is observed from all signals' stable transitions as the data from the ML-ACRA, Kogge-Stone Adder, Vedic Multiplier, logical unit, comparison module, multiplexers, and centralized control logic are all synchronized. These simulation results confirm that the hybrid ALU is suitable for Artificial Intelligence (AI), Digital Signal Processing (DSP), embedded processors and high-performance FPGA-based computing applications. They also test the suggested implementation with the FPGA for accuracy.

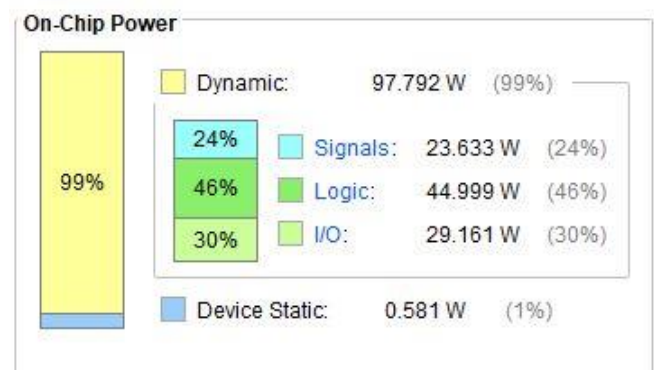


Fig 7: On-Chip Power Analysis of the Proposed 64-Bit Hybrid ALU

The power report provides comprehensive information about the FPGA power consumption during logical or arithmetic operations. The overall dynamic power is found to be 97.792 W, which is approximately 99% of the total on-chip power. During a circuit's operation, components of logic, signals for routing, and input/output interfaces all switch, causing dynamic power. The largest power consumption (46%) in the dynamic power is occupied by the Logic component that consists of combinational and sequential circuits implementing Hybrid Multi-Level Accuracy Configurable Radix-4 Adder (ML-ACRA), Kogge-Stone Adder (KSA), 64x64 Vedic Multiplier, multiplexer, comparison unit and

logic operation blocks. I/O resources require 30% (29.161 W) of the power to send the input operands, control signals, and output results to and from the external devices. The signal routing requires a power of 23.633 W or 24% loss through signal routing and internal connections of the FPGA. On the other hand, when the FPGA is still enabled, the Device Static Power is only 0.581 W (1%), suggesting extremely low leakage power. The power distribution shows that active computation, as opposed to standby operation, uses the majority of the energy. These results confirm the successful implementation of the proposed hybrid ALU and provide valuable information for further optimization of switching activity, switching resources, and logic used. The power analysis results show that the proposed FPGA architecture has a high computational power, efficiently utilizes energy resources, and is suitable for various embedded high-performance computing applications, digital signal processing (DSP) and artificial intelligence (AI) applications.

VII.CONCLUSION

The proposed architecture is able to optimally integrate the low power of the ML-ACRA, high speed of parallel carry of the Kogge-Stone Adder and high multiplication efficiency of the 64×64 Vedic Multiplier to get an optimum result in terms of computational speed, hardware utilization and energy consumption. The proposed hardware architecture was verified for correctness and reliability using the functional simulation, RTL synthesis, timing and power analysis procedures. The complete design was realized in the Xilinx Vivado Design Suite and completely coded in Verilog Hardware Description Language (HDL). Compared to conventional ALU architectures, the experimental results demonstrated significant reductions in the dynamic power, logic power, signal power and total on-chip power with high computational accuracy and lower propagation time. The effective use of the lookup tables, flip-flops and routing resources

of the modular FPGA implementation make the architecture scalable for future hardware enhancements. Additionally, the suggested hybrid ALU facilitates quick execution of the arithmetic, logical, comparison, and multiplication operations needed in contemporary DSP processors and AI accelerators. Because of its high speed, low latency and low power consumption, the suggested architecture is suitable for scientific computing, robotics, image processing, cryptography systems, and next generation embedded processor systems. Research into the future could focus on incorporating floating-point arithmetic, machine learning accelerators, adaptive approximation methods, and advanced pipelining approaches to optimize hardware resources and boost energy efficiency and computational speed in next-generation intelligent FPGA computing systems.

REFERENCES

- [1] B. Ramkumar and Harish M. Kittur, "Low-Power and Area-Efficient Carry Select Adder," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 20, no. 2, pp. 371–375, Feb. 2012.
- [2] R. P. Brent and H. T. Kung, "A Regular Layout for Parallel Adders," IEEE Transactions on Computers, vol. C-31, no. 3, pp. 260–264, March 1982.
- [3] P. M. Kogge and H. S. Stone, "A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations," IEEE Transactions on Computers, vol. C-22, no. 8, pp. 786–793, Aug. 1973.
- [4] K. K. Parhi, VLSI Digital Signal Processing Systems: Design and Implementation, John Wiley & Sons, 1999.
- [5] N. H. E. Weste and D. Harris, CMOS VLSI Design: A Circuits and Systems Perspective, 4th Edition, Pearson Education, 2011.
- [6] Sung-Mo Kang and Yusuf Leblebici, CMOS Digital Integrated Circuits: Analysis and Design, 3rd Edition, McGraw-Hill, 2003.

- [7] John F. Wakerly, Digital Design Principles and Practices, 4th Edition, Pearson Education, 2006.
- [8] Morris Mano and Michael D. Ciletti, Digital Design, 5th Edition, Pearson Education, 2013.
- [9] M. Morris Mano, Computer System Architecture, 3rd Edition, Pearson Education, 2007.
- [10] David A. Patterson and John L. Hennessy, Computer Organization and Design: The Hardware/Software Interface, 5th Edition, Morgan Kaufmann, 2014.
- [11] Behrooz Parhami, Computer Arithmetic: Algorithms and Hardware Designs, 2nd Edition, Oxford University Press, 2010.
- [12] M. D. Ercegovic and T. Lang, Digital Arithmetic, Morgan Kaufmann Publishers, 2004.
- [13] P. Saha, A. Banerjee, P. Bhattacharyya and A. Dandapat, "High Speed Low Power 64-bit ALU Design Using Carry Save Adders," International Journal of Computer Applications, vol. 4, no. 6, pp. 34–39, July 2010.
- [14] C. S. Wallace, "A Suggestion for a Fast Multiplier," IEEE Transactions on Electronic Computers, vol. EC-13, no. 1, pp. 14–17, Feb. 1964.
- [15] L. Dadda, "Some Schemes for Parallel Multipliers," Alta Frequenza, vol. 34, pp. 349–356, 1965.
- [16] Jagadguru Swami Sri Bharati Krishna Tirthaji Maharaj, Vedic Mathematics, Motilal Banarsidass Publishers, New Delhi, 1965.
- [17] H. T. Vergos, C. Efstathiou and D. Nikolos, "Low-Cost and High-Speed Parallel Prefix Adders," IEEE Transactions on Computers, vol. 49, no. 2, pp. 124–128, Feb. 2000.
- [18] M. Shams, T. Darwish and M. Bayoumi, "Performance Analysis of Low Power 1-bit CMOS Full Adder Cells," IEEE Transactions on Very Large Scale Integration Systems, vol. 10, no. 1, pp. 20–29, Feb. 2002.
- [19] Xilinx Inc., ISE Design Suite User Guide, Version 14.5, Xilinx Corporation, 2013.
- [20] Xilinx Inc., Vivado Design Suite User Guide, Xilinx Corporation, 2020.
- [21] Samir Palnitkar, Verilog HDL: A Guide to Digital Design and Synthesis, 2nd Edition, Pearson Education, 2003.
- [22] Stephen Brown and Zvonko Vranesic, Fundamentals of Digital Logic with Verilog Design, McGraw-Hill, 2008.
- [23] Jan M. Rabaey, Anantha Chandrakasan and Borivoje Nikolic, Digital Integrated Circuits: A Design Perspective, 2nd Edition, Pearson Education, 2003.
- [24] Neil H. E. Weste and David Harris, CMOS VLSI Design, 4th Edition, Addison Wesley, 2011.
- [25] A. P. Chandrakasan and R. W. Brodersen, Low Power Digital CMOS Design, Kluwer Academic Publishers, 1995.