iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 6 **Issue:** IV **Month of publication:** April 2018

DOI: http://doi.org/10.22214/ijraset.2018.4596

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 6 Issue IV, April 2018- Available at www.ijraset.com

Design of Reusable Context Pipelining for Coarse Grained Reconfigurable Architecture

P. Murali[1] (M. Tech), Dr. S. Tamilselvan[2], S. Yazhinian (Research Scholar) [3]
[1, 2, 3] *Dept of Electronics and Communication Engineering Pondicherry Engineering College, Pondicherry*

Abstract: ***A Coarse-Grained Reconfigurable Architecture (CGRA) is a processing platform which constitutes an interconnection of coarse-grained computation units. CGRAs are a well-researched topic and the design space of a CGRA is quite large. A typical CGRA requires many processing elements and a configuration cache for reconfiguration of its processing element array. However, such a structure consumes significant area and power. Therefore, designing cost-effective CGRA has been a serious concern for reliability of CGRA-based embedded systems. Reusable Context Pipelining is a universal approach in reducing power and enhancing performance for CGRA because it can be achieved by closing the power performance gap between the low powers oriented spatial mapping and high performance oriented temporal mapping. By focusing on the processor elements the power and area will be reduced. The processing element consists of arithmetic and logic unit, array multiplier, saturation arithmetic logic and multiplexer. The above components are designed for processing element which are used in CGRA. Each components in processing element are simulated using Xilinx ISE. Finally, simulation results and final transient response of the schematic design of processing element are visualized.***

Keywords: Coarse-Grained Reconfigurable Architecture (CGRA), Reusable Context Pipelining, processing element.

I. INTRODUCTION

Nowadays, stream-based applications, such as multimedia, telecommunications, signal processing, and data encryptions, are the dominant workloads in many electronic systems. The real-time constraints of these applications, especially over portable devices, often have stringent energy and performance requirements. Many other military applications, including real-time synthetic aperture radar imaging, automatic target recognition, surveillance video processing, optical inspection, and cognitive radio systems, have similar needs. General purpose processors (GPPs), are widely used in conventional data-path oriented applications due to their flexibility and ease of use. However, they cannot meet the increasing requirements on performance, cost, and energy in the data streaming application domain due to their sequential software executions.

The application-specific integrated circuits (ASICs) become inevitably a customized solution to meet these ever-increasing demands for highly repetitive parallel computations. It is reported that they are potentially two to three orders of magnitude more efficient than the processors in terms of combined performances of computational power, energy consumption, and cost. The long design cycle and high Non-Recursive Engineering (NRE) cost also become an obstacle to meet the stringent cost and time-to-market requirements.

Reconfigurable architectures (RAs) have long been proposed as a way to achieve a balance between flexibility as of GPP and performance as of ASICs [1]. The hardware-based RA implementation is able to explore the spatial parallelism of the computing tasks in targeted applications, meanwhile avoiding the instruction fetching, decoding, and executing overhead of the software implementations, which results in an energy and performance gain over general purpose processors. On the other hand, RAs maintain the post fabric flexibility to be configured, either offline or on the fly, to accommodate to new system requirements or protocol updates that is not feasible in ASIC implementations. Also, the flexibility provided by RAs can improve fault tolerance and system reliability.

Field-programmable gate arrays (FPGAs) are still the dominating semiconductor technology in the reconfigurable computing area. The most common SRAM-based FPGAs decompose complex logic functions into smaller ones and map them onto the Lookup Tables (LUTs) or other on-chip embedded resources. However, this flexibility comes at a significant cost in terms of area, power consumption, and speed, due to its huge routing area overhead and timing penalty. Furthermore, due to the fine-grained nature, the compilation and configuration of FPGAs take much longer than those in general purpose processors.

This paper is organized as follows: Section I gives the detail introduction .Section II explains the various methods which have been used in the literature. Section III describes about the Coarse Grained Reconfigurable Architecture. In Section Iv give the detail about the Reusable Context Pipelining. Section V describes the simulation results.

II. RELATED WORK

Recently, the methods have been explored by using Coarse-Grained Reconfigurable Architecture Shouyi Yin, Xianqing Yao, Dajiang Liu, Leibo Liu, and Shaojun Wei [1], discussed about the coarse-grained reconfigurable architectures and extra overhead of memory access and the limited communication bandwidth between the processing element array and local memory. Shouyi Yin, Dajiang Liu, Yu Peng, Leibo Liu, and Shaojun Wei [2], provides the CGRA with high performance and the loop pipelining techniques are usually used to exploit the parallelism of loops.The paper [3] discussed about the CGRA based multi core architecture achieving high performance by using kernel level parallelism. it as the high energy consumption and performance bottleneck. Bing xu, Shouyi Yin, and Shaojun Wei [4], contributed to the development of efficient compilers for mapping compute-intensive applications on CGRA using modulo scheduling. Ganghee Lee, Kiyoung Choi, Senior Member, IEEE, and Nikil D. Dutt, Fellow [5], discussed about the CGRAs supporting both integer and floating point arithmetic. After presenting an optimal formulation using integer linear programming and floating-point applications such as 3-D graphics. The Paper [6], "Implementing Flexible Reliability in a Coarse Grained Reconfigurable Architecture" discussed about the flexible reliability to deal with soft errors and aging. N. Ravindran, R. Mary Lourde [7], analysed the VLSI design of a 16 Bit ALU and design is optimized in terms of Speed, Power Consumption and Chip Area. pseudo-NMOS for AND logic and Pass transistor logic for multiplexers, in order to optimize the overall performance of the design. Jonghee W. Yoon and A viral Shrivastava [8], Utilizing fewer resources is directly translated into increased opportunities for novel power and performance optimization techniques. The Paper [9], discussed about the spurious power suppression technique (SPST) for a high speed low-power multiplier which can dramatically reduce the power dissipation of combinational VLSI designs for multimedia. Abhijit Asati and Chandrashekhar [10], presented a multipliers have a time complexity better than array multipliers, and and multiplier implementation shows large reduction in propagation delay and the average power consumption.

III.COARSE GRAINED RECONFIGURABLE ARCHITECTURE

A recent trend in the architectural platforms for embedded systems is the adoption of Reconfigurable computing elements for cost, performance, and flexibility issues. Coarse-Grained Reconfigurable Architectures (CGRAs) exploit both the flexibility and efficiency, and are shown to be a generally better solution for compute-intensive applications than fine-grained reconfigurable architectures.

The RAA has identical processing elements containing functional units and a few storage units such as ALU, multiplier, shifter and register file. The data buffer provides operand data to PE array through a high bandwidth data bus. The configuration cache (or context memory) stores the context words used for configuring the PE array elements. The context register between a PE and a cache element (CE) in configuration cache is used to keep the cache access path from being the critical path of the CGRA [6]. These CGRAs can be classified into two cases: mesh-based reconfigurable array and linear reconfigurable array. Mesh-based reconfigurable arrays arrange their processing elements (PEs) mainly as a rectangular 2-D array with horizontal and vertical connections, which support rich communication resources for efficient parallelism. In the case of linear reconfigurable arrays, they support pipelined execution for stream-based applications with static or dynamic reconfiguration.

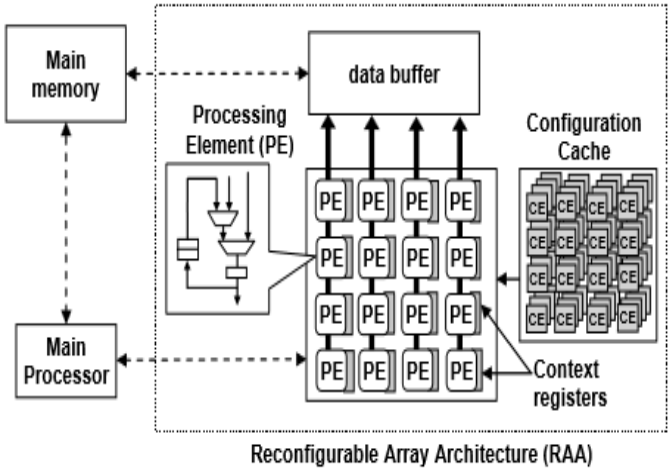


Fig. 1.1 Block Diagram of General CGRA

The implementation of reconfigurable architecture connecting the RAA as an Attached IP. In this case, the speed improvement using the RAA may have to compensate for significant communication overhead. It consists of a RISC processor, a main memory block, a DMA controller, and an RAA. The RISC processor is a 32-bit processor which is small and simple with three pipeline stages and the communication bus is AMBA AHB, which couples the RISC processor and the DMA controller as master devices and the RAA as a slave device. The RISC processor executes control intensive, irregular code segments and the RAA executes data-intensive kernel code segments.

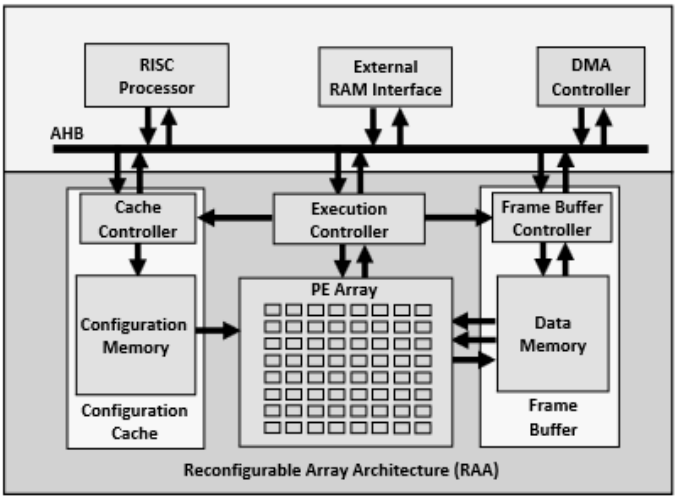


Fig. 1.2 Block Diagram of Base CGRA

IV. REUSABLE CONTEXT PIPELINING

Reusable Context Pipelining (RCP) is a universal approach in reducing power and enhancing performance for CGRA because it can be achieved by closing the power-performance gap between low powers oriented spatial mapping and high performance-oriented temporal mapping [2]. The proposed new configuration cache structure (called hybrid configuration cache) to support reusable context pipelining with reduced memory size. RCP is the technique which involves the combination of spatial mapping and temporal mapping.

A. Arithmetic and Logic Unit

An arithmetic logic unit (ALU) is a combinational digital electronic circuit that performs arithmetic and bitwise operations on integer binary numbers. The inputs to an ALU are the data to be operated on, called operands, and a code indicating the operation to be performed; the ALU's output is the result of the performed operation. In many designs, the ALU also has status inputs or outputs, or both, which convey information about a previous operation or the current operation, respectively, between the ALU and external status registers. The ALU's input signals, which are held stable until the next clock, are allowed to propagate through the ALU and to the destination register while the CPU waits for the next clock. When the next clock arrives, the destination register stores the ALU result and, since the ALU operation has completed, the ALU inputs may be set up for the next ALU operation.

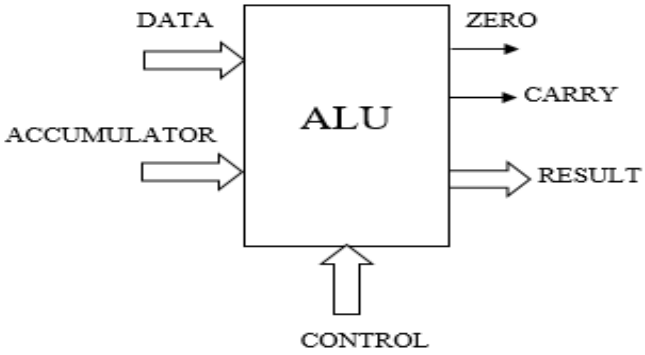


Fig. 1.3 Block Diagram of ALU

Multiplier is mainly used to implement algorithms like frequency domain filtering (FIR and IIR), frequency-time transformations (FFT), Correlation etc. Most DSP tasks require real-time processing; it must perform these tasks speedily while minimizing Cost and Power [7]. In this architecture the 2×2 unsigned multiplier is used as a basic building block in a hierarchical design of a larger bit size multiplier. The truth table for a 2×2 combinational multiplier is shown in fig 1.4.

Inputs				outputs			
A_1	A_0	B_1	B_0	P_3	P_2	P_1	P_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	0
0	1	0	1	0	0	0	1
0	1	1	0	0	0	1	0
0	1	1	1	0	0	1	1
1	0	0	0	0	0	0	0
1	0	0	1	0	0	1	0
1	0	1	0	0	1	0	0
1	0	1	1	0	1	1	0
1	1	0	0	0	0	0	0
1	1	0	1	0	0	1	1
1	1	1	0	0	1	1	0
1	1	1	1	1	0	0	1

Fig .1.4 Truth Table for 2×2 Multiplier

The step in the design of 4 bit multiplier will be finding the different combinations of input bit pairs that are derived in terms of 2×2 multiplier. Each input bit-pair is handled by a separate 2×2 combinational multiplier to produce 4 partial product rows [9]. These partial products rows are then added optimally to generate final product bits. The design procedure for 4×4 combinational multiplier is the schematic of a 4×4 combinational multiplier designed using 2×2 combinational multiplier. These partial products rows are then added optimally using 5-bit full adder cells.

V. SIMULATION RESULTS

The following results are the graphs, RTL schematic, and design summary of the components which are processing element and buffer. These results are help for designing the CGRA.

A. Processing Element

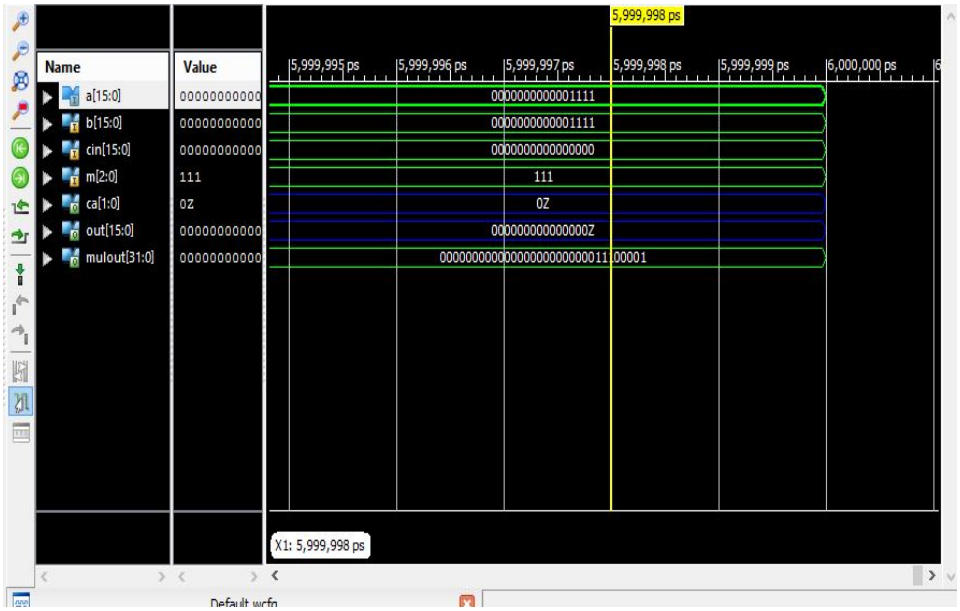


Fig.1.5 Transient Response of PE

The PE graph is shown in fig. 1.5 the inputs a, b, Cin which are 16 bits and output is 16 bits which are based on the selected line input. The array multiplier's output is 32 bits because the inputs are two 16 bits. The PE performs 8 different operations based on the multiplexer selected lines. The PE which gives response behavioral modelling. Register Transfer Level (RTL) is consist ALU, array multiplier and multiplexer. RTL is a design abstraction which models a synchronous digital circuit in terms of the flow of digital signals between hardware registers, and the logical operations performed on those signals. The schematic diagram of PE is shown in fig. 1.6.

A lookup table is an array that replaces runtime computation with a simpler array indexing operation. The savings in terms of processing time can be significant, since retrieving a value from memory is often faster than undergoing an "expensive" computation or input/output operation. The tables may be recalculated and stored in static program storage, calculated (or "pre-fetched") as part of a program's initialization phase (memorization), or even stored in hardware in application-specific platforms. The Performance Metrics of processing element has Power (13.65mw), Delay (4.4ns) and Number of Gates used is 301.3k

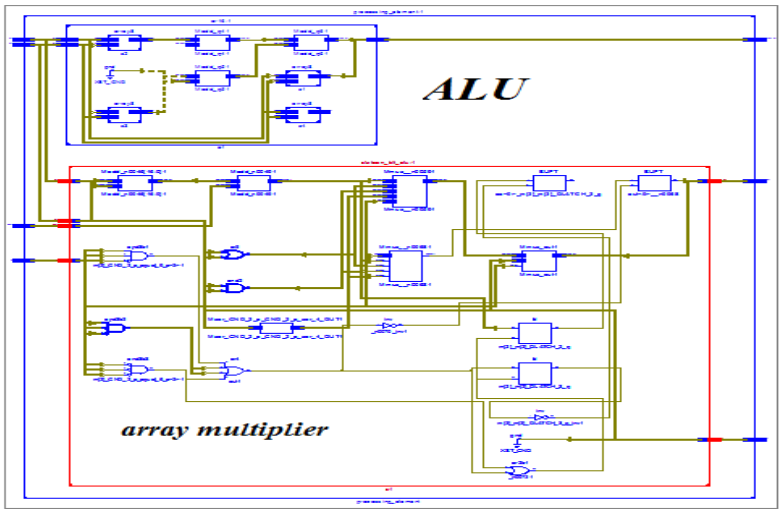


Fig. 1.6 Schematic Diagram of PE

B. Buffer

Buffer is a region of a physical memory storage used to temporarily store data while it is being moved from one place to another. Typically, the data is stored in a buffer as it is retrieved from an input device . In that CGRA, buffer is used in the cache element is shown in fig. 1.7. The Cache element is used for recovering the data from the previous PEs. The cache element used by the central processing unit (CPU) of a computer to reduce the average cost (time or energy) to access data from the main memory. A cache is a smaller, faster memory, closer to a processor core, which stores copies of the data from frequently used main memory locations. Most CPUs have different independent caches, including instruction and data caches, where the data cache is usually organized as a hierarchy of more cache levels.

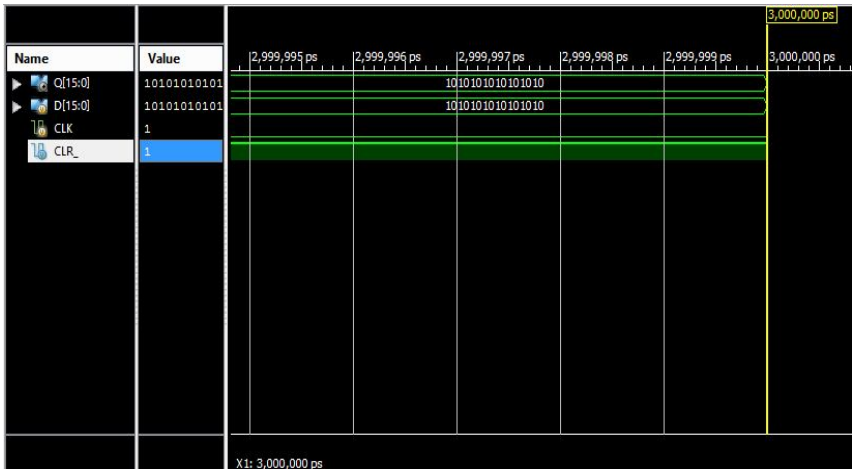


Fig. 1.7 Transient Response of Buffer

VI. CONCLUSION

Coarse Grained Reconfigurable Architecture (CGRA) is thoroughly detailed by each elements present in this architecture. The CGRA design is described beginning with the block diagram down to the components in the design. CGRA brings many opportunities as well as challenges to the VLSI. Earlier a brief introduction and basic concepts of CGRAs are reviewed. A detailed description about its architecture, reconfigurable array architecture coupling, RAA and its elements like processor element, frame buffer, execution controller, configuration cache are discussed. Also the breakdown of area, delay and power cost CGRA are presented. Also RCP technique which is new configuration cache structure (called hybrid configuration cache) to support reusable context pipelining with reduced memory size. RCP is the technique which involves the combination of spatial mapping and temporal mapping.

The components of PE are designed using Verilog library Xilinx ISE 14.2. Construction of the element consisted of producing a detailed schematic of each modules constituted in the design. Finally, simulation results of the schematic design of processing element are visualized with necessary terminologies. The final response of the PE produced a clean signal for different inputs and produces different operations.

VII. FUTURE SCOPE

The CGRA will be designed in Xilinx ISE 14.2 by using connecting PEs. The operations can be extended to 32 bit, 64 bit and 128 bit also it consumes the more power. Based on the applications CGRA will be designed.

REFERENCES

[1] Shouyi Yin, Xianqing Yao, Dajiang Liu, Leibo Liu, and Shaojun Wei, "Memory-Aware Loop Mapping on Coarse-Grained Reconfigurable Architectures" IEEE transactions on very large scalintegration (VLSI) systems, vol. 24, no. 5, pp.1895 - 1908 May 2016.

[2] Shouyi Yin, Dajiang Liu, Yu Peng, Leibo Liu, and Shaojun Wei "Improving Nested Loop Pipelining on Coarse-Grained Reconfigurable Architecture "IEEE transactions on very large scale integration (VLSI) systems, vol. 24, no. 2, pp. 507-520 February 2016.

[3] Yoonjin Kim, Hyejin Joo, Sohyun Yoon "Inter-coarse- grained reconfigurable architecture reconfiguration technique for efficient pipelining of kernel-streamon coarse grained reconfigurable architecture-based multi-core architecture"IET circuits, devices & System, vol. 10, issue: 4,pp. 251 - 265, 2016.

[4] Bing xu, Shouyi Yin, Leibo Liu, Shaojun Wei "Low-Power Loop Pipelining Mapping onto CGRA Utilizing Variable Dual-V_{dd}"IEEE 57th International Midwest Symposium on Circuits and Systems (MWSCAS),pp. 242-245, 2014.

[5] Ganghee Lee, Kiyoung Choi, Senior Member, IEEE, and Nikil D. Dutt, Fellow "Mapping Multi-Domain Applications onto Coarse-Grained Reconfigurable Architectures"IEEE transactions on computer-aided design of integrated circuits and systems, vol. 30, no. 5,pp. 637-650, May 2011.

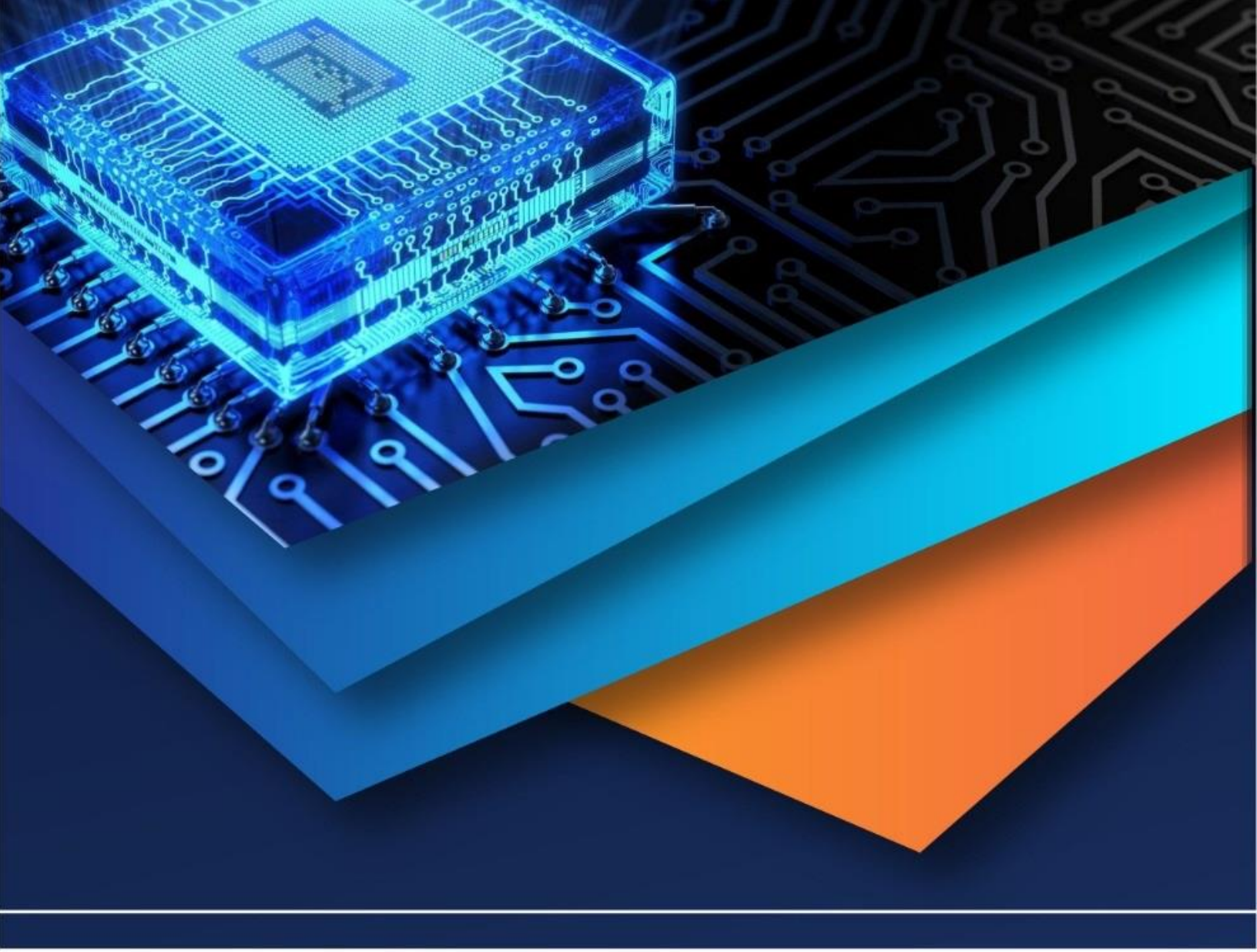
[6] Dawood Alnajjar, Hiroaki Konoura, Younghun Ko, Yukio Mitsuyama, Masanori Hashimoto, Takao Onoye "Implementing Flexible Reliability in a CGRA "IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 21, No. 12, pp. 2165 – 2178 December 2013.

[7] N. Ravindran, R. Mary Lourde, "An optimum VLSI design of a 16-BIT ALU" International Conference on Information and Communication Technology Research (ICTRC), PP. 52-55, 2015.

[8] Jonghee W. Yoon, Aviral Shrivastava, Sanghyun Park, MinwookAhn, YunheungPaek "A Graph Drawing Based Spatial Mapping Algorithm for Coarse-Grained Reconfigurable Architectures " IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 17, Issue: 11, pp. 1565-1578,2009.

[9] ChadalavadaDivya, Sai KokaVinila, "A Spst Based 16x16 Multiplier For High Speed Low Power Applications Using RadiX-4 Modified Booth Encoder" International Journal of Advancements in Research & Technology, vol. 4, Issue 6,pp.31-35 June -2015.

[10] Abhijit Asati, Chanderashekar, "A high speed hierarchical array of array multiplier design" International multimedia signal processing and communication technologies,pp. 161-164,2009.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY