DEVELOPMENT OF ARITHMETIC ALGORITHM AND DESIGN OF ARCHITECTURE LEVEL MULTIPLIER FOR FAST AND EFFICIENT COMPUTATIONS ON FPGA

Thesis Submitted for the Award of the Degree of

DOCTOR OF PHILOSOPHY

in

Electronics and Communication Engineering

By

Jugal Kishore Bhandari
Registration Number: 42000123

Supervised By

Dr.Yogesh Kumar Verma (25263)
School of Electronics and Electrical Engineering,
Associate Professor

LOVELY PROFESSIONAL UNIVERSITY, PUNJAB
2024

DECLARATION

I hereby declared that the presented work in the thesis entitled " Development of arithmetic algorithm and design of architecture level Multiplier for fast and efficient computations On FPGA " in fulfilment of degree of **Doctor of Philosophy (Ph. D.)** is outcome of research work carried out by me under the supervision of Dr.Yogesh Kumar Verma, working as Assistant Professor, in the School of Electronics and Electrical Engineering of Lovely Professional University, Punjab, India. In keeping with general practice of reporting scientific observations, due acknowledgements have been made whenever work described here has been based on findings of other investigator. This work has not been submitted in part or full to any other University or Institute for the award of any degree.

(Signature of Scholar)

Name of the scholar: Jugal Kishore Bhandari

Registration No.: 42000123

Department/school: School of Electronics and Electrical Engineering

Lovely Professional University,

Punjab, India

CERTIFICATE

This is to certify that the work reported in the Ph. D. thesis entitled " Development Of Arithmetic Algorithm And Design Of Architecture Level Multiplier For Fast And Efficient Computations On FPGA " submitted in fulfillment of the requirement for the award of degree of **Doctor of Philosophy (Ph.D.)** in the Electronics and Communication Engineering, is a research work carried out by Jugal Kishore Bhandari, 42000123, is bonafide record of his original work carried out under my supervision and that no part of thesis has been submitted for any other degree, diploma or equivalent course.

(Signature of Supervisor)

Name of supervisor: Dr.Yogesh Kumar Verma

Designation: Associate Professor

Department/school: School of Electronics and Electrical Engineering,

University: Lovely Professional University

ABSTRACT

Cryptography systems have evolved into indispensable components of modern communication devices. They enable real-time data processing, encryption-decryption, and high-speed computations, leveraging the power of Digital Signal Processing (DSP) resources within Field-Programmable Gate Arrays (FPGAs). To access DSP performance, various methods have been developed, each offering varying levels of accuracy and relevance.

The first method, Embedded Multipliers Performance, provides a fundamental means of comparing DSP performance. However, it overlooks the intricate details of DSP architecture and the overall complexity of system design, rendering it the least accurate among the available methods. A more precise approach is DSP IP Benchmarks, which assesses performance through standard operations like Fast Fourier Transforms (FFT) and Finite Impulse Response (FIR) filtering—crucial operations in DSP applications. Yet another avenue is Application-Level Benchmarks, which meticulously measures the performance of a silicon solution when applied to specific tasks, exemplified by benchmarks from Berkeley Design Technology Inc. (BDTI).

This study primarily focuses on the comparison between DSP IP benchmarks and application-level benchmarks, harnessing DSP performance data from standard FPGAs provided by industry giants such as Altera/Intel and Xilinx. The proposed approach introduces an innovative system design that incorporates a high-speed multiplier within the DSP arithmetic block of next-generation FPGAs. This design meticulously optimizes DSP resource utilization, thus liberating additional slices and Look-Up Table (LUT) area to support resource-intensive applications.

FPGAs, equipped with logic elements (LEs) and memory, offer a flexible platform for configuring diverse functionalities using hardware description languages like VHDL or Verilog HDL. These high-density FPGAs, typified by the offerings from Altera and Xilinx, integrate embedded silicon features

tailored for DSP applications. These features encompass arithmetic operations, encoders, decoders, and vital DSP functions such as Finite Impulse Response (FIR) filters, equalizers, Fast Fourier Transforms (FFTs), and correlators. FPGA's intrinsic adaptability lends itself to the implementation of multifaceted hardware designs and operations, rendering them versatile platforms for numerous applications.

In this research, an investigation into the implementation of a novel hybrid multiplier on FPGA platforms is conducted. The investigation commences by examining Built-In Self-Test (BIST) results concerning FPGA Serializer/Deserializer (SERDES), offering profound insights into resource utilization. The proposed design, featuring the hybrid multiplier, distinguishes itself by delivering superior performance, even though it consumes a slightly higher number of Look-Up Tables (LUTs) and flip-flops. Notably, it excels in DSP slice utilization, promising substantial resources for a diverse range of applications.

Furthermore, practical real-world applications are explored, such as an Advanced Fall Detection System for the Elderly. This system harnesses the capabilities of the hybrid multiplier to enhance its overall performance and efficiency. Falls among elderly individuals pose significant health risks, and the innovative approach, which melds FPGA technology with computer vision, has the potential to save lives and improve emergency response times.

The hardware and software components of the study include the Zynq UltraScale+ MPSoC ZCU104 and a USB camera board, highlighting a comprehensive approach to system design. By transitioning to computer vision-based fall detection, the limitations of traditional sensor-based solutions are addressed, enhancing accuracy and comfort for elderly individuals.

The research also highlights the choice of FPGA hardware, favouring Xilinx UltraScale over Intel Arria based on factors like resource optimization,

compatibility, and prior expertise. This choice aligns with the project's objectives and research focus.

In brief, this research explores the implementation of a hybrid multiplier on FPGA platforms, demonstrating its advantages in resource utilization and real-world applications. This work represents a significant contribution to the field, offering innovative solutions for improving the safety and well-being of the elderly and highlighting the potential of FPGA-based technologies in various domains.

The integration of a novel hybrid multiplier into FPGA platforms, with a focus on enhancing DSP resource utilization, is expected to yield substantial improvements in both performance and resource efficiency. This study posits that the hybrid multiplier, when compared to conventional FPGA multipliers, will demonstrate superior DSP slice utilization, allocate resources more efficiently, and exhibit enhanced capabilities in practical applications. As a result, it is hypothesized that this research will significantly advance FPGA-based hardware design and underscore the versatility of FPGA technology in diverse domains.

In view of the above, the main objectives of the work conducted in the thesis are as follows:

1. Analyze the design specifics and resource utilization of both conventional and proposed FPGA multipliers.

2. Evaluated the performance of FPGA using the proposed multiplier design, effectively accelerating typical workloads on DSP slices.

3. Compared the potential parameters of the proposed method with existing methods found in the literature.

PREFACE / ACKNOWLEDGEMENT

This thesis would not have been possible without the support of glorious people who motivated me during my doctoral study. I am thankful from bottom of my heart towards the numerous persons who assisted me while conducting this study. First of all, I would like to thank my supervisor, Dr. Yogesh Kumar Verma, for his worthy guidance, support, and suggestions, in every step of this research project during my Ph.D. journey. Dr. Yogesh has an optimistic personality with a helpful nature, he has always made himself ready to clarify my doubts and it was a great opportunity to work under his supervision. He always shed light whenever I was feeling stuck in my path of research ambitions.

I would like to express my gratitude towards the entire Lovely Professional University family for providing suitable infrastructure and environment for completing my research work in a time-bound manner. Also, I would like to thank the Division of Research & Development and School of Electronics and Electrical Engineering for their help and encouragement in my entire Ph.D. journey.

Last, but not least I would express my sincere gratitude to my family for their love, sacrifice, and moral support for without their continued support this work would never have been possible. Finally, I like to thank almighty God who helped me to achieve such a big milestone.

CONTENTS

List of Figures

List of Tables

Acronyms and Abbreviations

Acronyms	Description
DNN	Deep Neural Network
SDN	Software Defined Networking
NIC	Network Interface Card
MAC	Multiplier/Accumulator
GMAC	Giga Multiply-Accumulate
LUT	Look Up Table
CLB	Combinational Logic Block
SERDES	Serializer/Deserializer
DCT	Discrete Cosine Transforms
ETA	Error Tolerant Approximation
STM	Standard Technology Model
Gf2X	Galois Field over 2 Elements
SBST	Selective Built-In Self-Test
PE-ATPG	Pseudo-Exhaustive Automatic Test Pattern Generation

CHAPTER 1

INTRODUCTION AND LITERATURE REVIEW

1.1. Introduction:

Multipliers are an essential component in Digital Signal Processing Systems and Embedded applications, and it is common practice to assess their efficiency based on criteria such as power usage, delay, and area. In recent years, various research efforts have been devoted to reducing the power consumption of multipliers in VLSI design. Since multipliers are frequently used for arithmetic operations, they can generate significant spurious switching activity. Combining architectural and transistor-level optimization techniques can help resolve this problem by achieving a balance in the internal paths of the multiplier circuit.

The recent trend toward smaller, more powerful mobile communication and portable devices has placed a premium on speed, among other key factors such as area and power consumption, in modern VLSI design. Multipliers are a critical component of the arithmetic processing unit in these devices and must provide both high speed and low power consumption in a compact form factor. As a result, there is a strong demand for multipliers that can deliver both the required speed and power efficiency to meet the needs of these modern applications.

A promising strategy for minimizing overall power consumption, particularly dynamic power which can contribute significantly to total power dissipation, is to reduce the number of operations in a multiplier design. In the past, significant research efforts have focused on designing VLSI multipliers with this goal in mind. One popular approach is to use parallel multipliers, which can help to improve processor speed compared to serial multipliers.

In VLSI circuits, achieving high performance with low energy consumption is of critical importance. When it comes to implementing

multiplication in hardware, there are two main approaches: using more hardware to achieve faster execution or using less hardware and accepting slower execution [1]. The traditional method for hardware multiplication involves computing partial products, shifting them as needed, and then adding them up, much like multiplication is done by hand.

Efficient logic styles that prioritize energy efficiency are highly significant for VLSI circuits. The conventional method of hardware multiplication involves performing multiplication in the same way as it is done manually.

In numerous DSP algorithms, multiplication is a crucial operation that involves multiplying a multiplicand by a multiplier. The procedure necessitates the multiplication of each constituent of the multiplier with all bits of the multiplicand. The resulting products are partial and are then added together based on their respective weights, which correspond to the position of each bit relative to the other bits. To illustrate, during the addition of the partial product of bits 0 through 3 to the partial product of bits 4 through 7, the former is subject to a shift based on its weight before merging with the latter. The diagram depicted in Fig 1.1 illustrates the multiplication of the two-bit multiplier a1a0 with the two-bit multiplicand b1b0, resulting in a 2 × 2 multiplication.

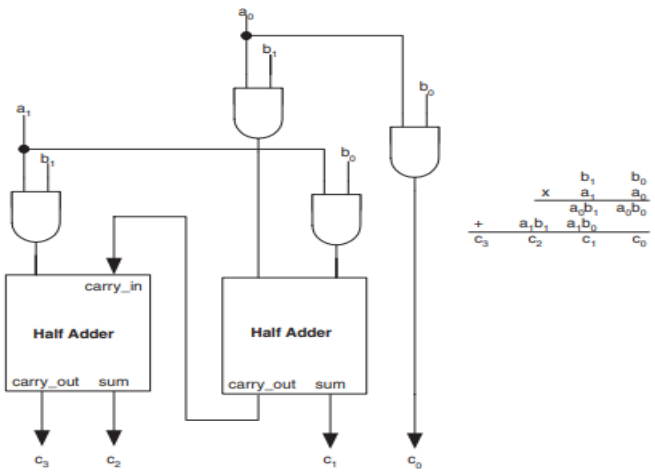


Fig 1.1. multiplication of the two-bit multiplier with the two-bit multiplicand

The speed of the multiplier is crucial for the overall performance of DSP algorithms, as it lies on the critical delay path. Historically, multiplication was accomplished through a sequence of shift, addition, and subtraction operations. Despite the prevalence of specialized hardware for multiplication in contemporary DSP systems to enhance speed compared to software-based techniques, multiple multiplication algorithms have been presented in academic publications. Each algorithm has its unique benefits and drawbacks with respect to speed, circuit complexity, area, and power consumption. The quantity of circuitry within a multiplier is in direct proportion to its magnitude, whereby n-bit multipliers necessitate n^2 gates. For DSP applications, multiplication algorithm performance is primarily determined by latency and throughput. Latency denotes the time it takes to compute a function, while throughput measures how many multiplications can be completed in a given timeframe. Multiplication is both a high-delay block and a substantial contributor to power dissipation. Therefore, reducing delay using various optimization techniques is critical to minimizing power consumption.

1.1.1. All about FPGAs:

The domain of Field Programmable Gate Arrays (FPGAs) is projected to experience substantial growth in the upcoming times, particularly in the aerospace and military sectors, for tasks such as waveform generation, image processing and secure communication. This expansion is predicted to fuel growth within the FPGA industry.

The demand for FPGAs is anticipated to rise in several sectors, including network processing, security, and Deep Packet Inspection (DPI), which is expected to drive growth in the industry.

Advanced embedded FPGA architecture opportunities are created due to the increasing demand for high bandwidth at low power and low cost. These architectures are typically utilized in applications that require heavy data flow, data processing, and streaming due to their high computing density and low power consumption. Fig 1.2 shows the FPGA market

forecast estimated till 2027.

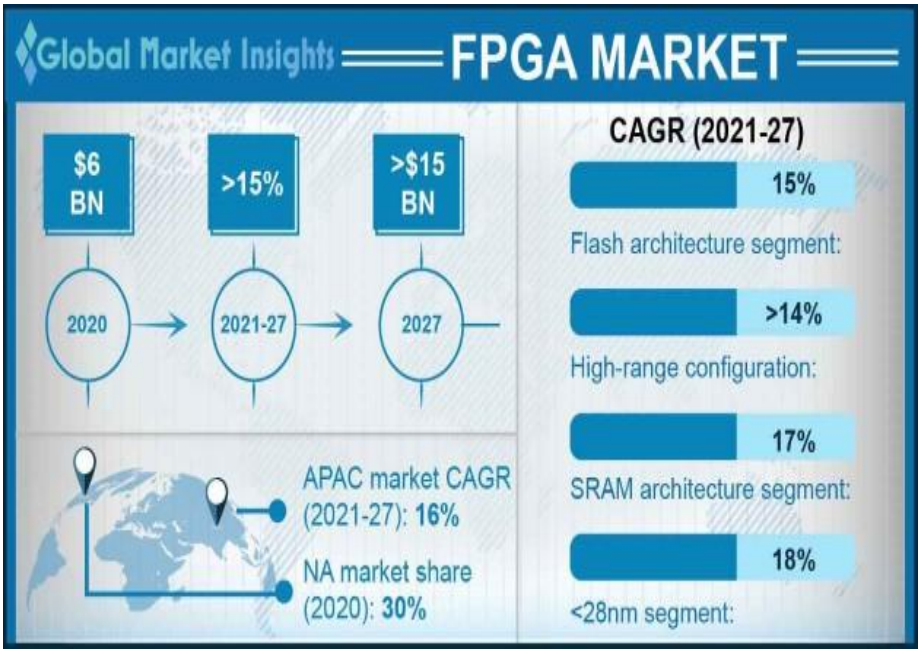


Fig.1.2. FPGA market Forecast till 2027

In contemporary high-performance cloud and edge computing systems, FPGA-based accelerators have become a reliable substitute for traditional GPU-based accelerators, demonstrating significant advancements in recent years.

High-Level Synthesis (HLS) enables developers to configure FPGAs accurately, using high level languages such as OpenCL, C++/C, SystemC.

1.1.2. Definition of FPGA:

Xilinx characterizes FPGAs as semiconductor gadgets that have a framework of configurable logic blocks (CLBs) that are linked through programmable interconnects. These FPGAs are modifiable post-production to meet distinct application or feature necessities.

Unlike ASICs that are designed intended for a particular task, FPGAs are adaptable and can be reconfigured as the design progresses. While one-time programmable (OTP) FPGAs are an option, the prevalent type of FPGAs employed are those based on SRAM technology. An example of CPU usage explained in Fig 1.3 as non-programmable gates.

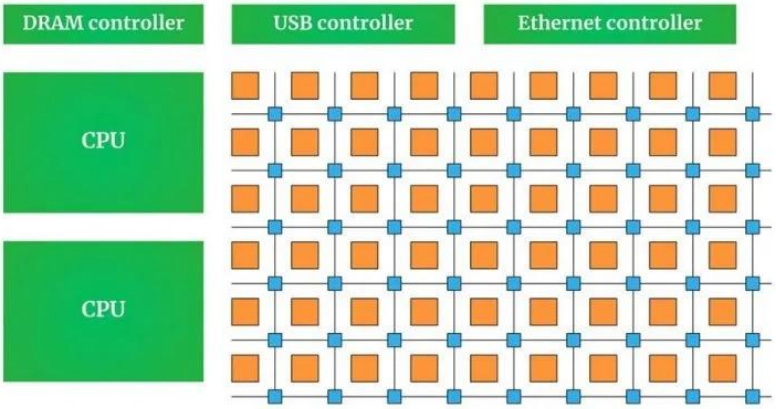


Fig 1.3. Modern FPGA: lots of hard, not field programmable gates

1.1.3. FPGA Market share:

The market and industry are extremely competitive, which is evident from the latest industry statistics provided by major players such as Xilinx, Altera, Lattice, and Actel. As indicated in a markets and markets report, Xilinx, the primary producer of FPGAs, approximates that the complete logic market has a value of $57 billion.

The market distribution is as follows: ASICs contribute $14.0 billion, FPGAs $2.8 billion, PLDs represent $0.5 billion, and general-purpose logic accounts for $8.5 billion.

The FPGA industry worldwide is classified based on technology, application, and geographical location. The technology classification includes SRAM, EEPROM, Antifuse, Flash, and other technologies. The application segment includes various industries, such as data processing, manufacturing, consumer electronics, military & aerospace, wireless, automobiles and others, which can be classified into high-end, mid-end, and low-end FPGAs.

In regard to geographical analysis, the industry is examined in Asia-Pacific, Europe, North America, and LAMEA.

The market is growing as cloud users are utilizing FPGAs as an IaaS resource. Cloud service providers are using Field Programmable Gate

Arrays for various purposes, including deep learning, network encryption, memory caching, high frequency trading, webpage scoring, and video conversion.

Amazon, Inc. uses an FPGA coprocessor in the EC2 F1 virtual machine to provide hardware acceleration to customers. Similarly, for Azure ML- Microsoft Corporation uses FPGAs to evaluate DNNs, accelerate SDN, and rank Bing search results. Additionally, the increase in investments in data centers is contributing to the growing demand.

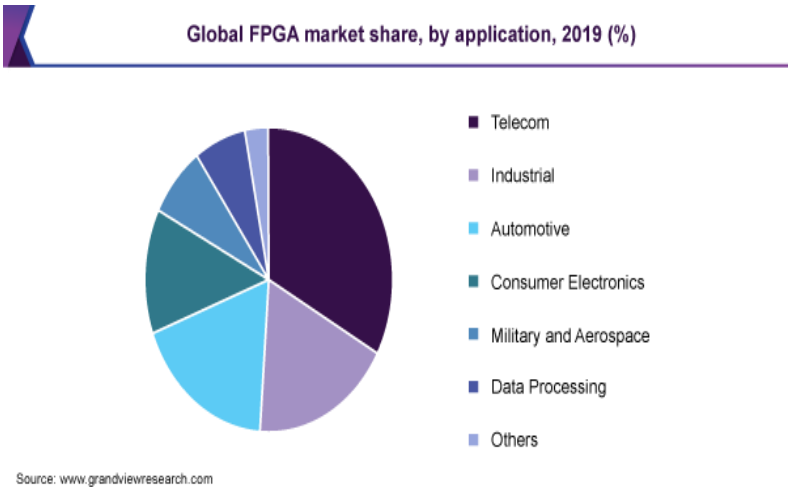


Fig.1.4. Global FPGA market share-2019

Greater coverage, data collection, and implementation of electronic countermeasures are achieved by using FPGAs in tactical vehicles such as radars, cameras, and electronic warfare systems. The Embedded Field-Programmable Gate Array market's growth may be somewhat restricted during the projected period due to the power-intensive nature of the devices and the absence of standardized industry authentication techniques, which could act as a constraint. Moreover, Heterogeneous Data Processing Platforms that include SmartNICs, multi-core processors, and hardware accelerators necessitate users to opt for a comprehensive solution instead of a mix-and-match approach. Fig 1.4 shows Global FPGA market share till 2019.

Nvidia, AMD and Intel, along with their proprietary accelerators and

SmartNICs, will also provide all-inclusive processing solutions. Additionally, FPGAs will have a significant presence in the embedded device industry, Intel/Xilinx have both integrated ARM cores into their SoC-based FPGAs, which have gained significant popularity in embedded applications.

1.1.4. FPGA Vs ASIC:

The incorporation of embedded processors into FPGAs could become the principal method for designing embedded devices, unleashing the capability of HW/SW co-design tools, propelling further expansion. Even with making strides in technology, FPGA designs are limited to low-volume applications due to their larger size, slower speed (3.5 times), and higher power consumption (14 times) compared to ASIC designs.

In the future, the most significant area of development for FPGAs will be the design of new programming software. With the increasing size and complexity of programmable devices that integrate one or more processors, there is a significant need for tools that can effectively utilize these capabilities and enhance design optimization.

Both in terms of market share and innovation, the FPGA industry is currently undergoing a period of growth. Contemporary FPGAs possess cutting-edge technologies that facilitate fast computing speeds, expanded connectivity choices, robust security features, and extensive bandwidth support. FPGAs have been tailored to cater to the demands of current industrial applications and market requirements, and their enhanced performance and onboard features make them an appealing option for a growing range of applications in the upcoming years.

Choosing between ASICs and FPGAs requires a careful evaluation of their respective value propositions. There is a wealth of information available that compares the two technologies. Previously, FPGAs were mainly utilized for designs with lower speed, complexity, and volume. Nevertheless, present-day FPGAs can effortlessly achieve performance

thresholds of 500 MHz and more, owing to significant enhancements in logic density, along with availability of features like embedded processors, clocking, DSP blocks, and high-speed serial interfaces available at reduced costs. These advancements make FPGAs an attractive option for almost any type of design.

1.1.5. Implementation of FPGA in Industries:

FPGAs are utilized in a diverse range of end-use industries, including but not limited to Aerospace and Defense, Telecom, Automotive, Video and Image Processing. Conducted a poll asking the audience which industry they think will have the highest demand for FPGAs. According to the results, the Aerospace and Defense sector is predicted to lead the trend of FPGA adoption in 2022, with 48% of the participants voting for it.

Around 30% of the participants believe that the Automotive and Telecom industries will have the highest demand for FPGAs, while 22% of the votes predict that FPGAs will find the most application in the Video and Image processing domain. [logic-fruit.com]

1.1.6. Programming languages for FPGA development:

FPGAs are typically programmed using Hardware Description Languages (HDLs), which are primarily low-level languages. VHDL and Verilog are two of the most used HDLs for FPGA programming, and hardware and FPGA engineers are typically the ones with expertise in these languages. This lack of familiarity with HDLs can limit the ability of software engineers to build embedded systems using FPGAs.

However, thanks to recent advancements in FPGA programming tools, it is no longer mandatory to learn HDLs to program FPGAs. FPGAs can now be programmed using high level languages such as C and C++, making it feasible to use these languages for FPGA programming.

Thanks to unified software platforms, it is now possible for software developers to program FPGAs using their preferred languages without requiring a solid understanding of HDLs. This allows for a less stressful

transition to new programming languages and enables software engineers to focus more on concepts rather than hardware.

These platforms work by translating higher-level languages into lower-level ones that an FPGA can understand and execute. In survey, participants asked to participants about their preferred language for FPGA programming and found that 86% of respondents use VHDL/Verilog, while C/C++ is used by 10% of respondents. Only 3% of participants reported using LabVIEW FPGA.

1.2. DSP Functions in FPGA:

The field of digital signal processing (DSP) involves a variety of complex operations, including digital filters, encoders, decoders, correlators, and mathematical transforms like the fast Fourier transform (FFT). These algorithms and functions typically require numerous variables, coefficients, and stages, with the primary essential operation often carried out by MAC units. To achieve high operating frequency and throughput, DSPs are commonly used, as they have specialized hardware and instruction sets optimized for MAC operations and features like bit-reverse addressing.

DSP CPUs are designed to execute instructions in a shorter number of clock cycles than processors that are more general-purpose. In the past, DSPs have been the preferred platform for the efficient implementation of DSP algorithms. Nevertheless, FPGAs have recently emerged as formidable rivals in this area owing to their inherent parallelism, adept handling of arithmetic operations, and plentiful logic resources at hand.

With the growing popularity of FPGAs, new application niches that require specialized hardware resources have emerged. One of the most valuable assets for such applications is the existence of built-in memory blocks. These blocks are exceptionally suitable for deploying data acquisition and control circuits, as they remove or reduce the requirement for external memories and decrease time to access memory.

In conventional DSPs, ALUs generally have one-to-four MAC units that work parallelly. Their rigid architectures do not permit the customization of the number of bits in a multiplication's operands. Consequently, these platforms inherently restrict parallelism and bandwidth, and augmenting operating frequency is usually the sole method for augmenting performance.

Constructing MAC units in an FPGA entails a basic approach that involves building adders and multipliers through distributed logic and integrating them with embedded memory blocks to function as accumulators, which can store coefficients. Yet, this solution may require using a large number of logic blocks, resulting in high resource usage and long propagation delays that limit the operating frequency in several instances.

The architecture of modern FPGAs is heterogeneous, meaning that it includes both programmable and non-programmable components. The programmable fabric is made up of LUTs and FFs, while non-programmable components include dedicated DSPs. The design of these DSP blocks differs among various vendors. Fig 1.5 presents DSP48E2 architecture.

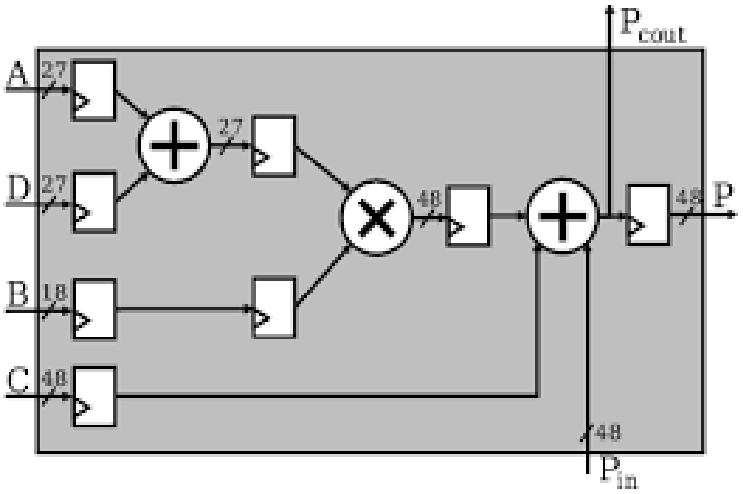


Fig.1.5. Architecture of Xilinx DSP48E2 Dataflow

To achieve high performance, area efficiency and energy efficiency, arithmetic circuits can be implemented using dedicated DSPs that are

present in modern FPGAs. These DSP blocks can offer much better performance compared to standard programmable FPGA fabric. Hence, it is crucial to make optimal use of these available DSP blocks when designing arithmetic circuits [2].

Optimal utilization of DSP blocks is critical since they are a scarce resource on FPGA chips. Nonetheless, it can present a difficulty when creating arithmetic circuits with narrow bit widths. Implementing such circuits on a DSP block would result in a significant proportion of the block's resources remaining unused. This is particularly relevant for applications like machine learning and image, which frequently operate on small width quantized data of 8 bits or less [3], [4].

To optimize the use of DSP sources for low-precision arithmetic operations, various methods have been suggested to map numerous multiplications of modest bit widths to a specific DSP [3], [4]. However, a more efficient utilization of resources can be attained by implementing the principles of Approximate Computing. The emerging paradigm of Approximate Computing involves sacrificing non-functional requirements like latency, chip area, or energy efficiency to improve computational accuracy. [5], [6].

As one would anticipate, DSP blocks are equipped with specialized interconnects for chaining them together with neighboring blocks. Extending the operand bit width in arithmetic operations, enabled by the principles of Approximate Computing, can facilitate the Implementing complex arithmetic functions and processing algorithms often requires multiple stages that operate in parallel, such as digital filters. DSP blocks are usually located near embedded memory blocks as embedded multipliers.

The quantity of DSP blocks present in an FPGA varies according to the requirements of the intended application. In signal processing devices, a high number of DSP blocks, ranging in the thousands, may be present, enabling computing speeds that can reach hundreds of GMAC/s. The high

computing speeds offered by DSP blocks enable the possibility of time-multiplexing methods, allowing multiple operations of lower frequency to be performed in a single block. Semiparallel structures strike an efficient balance between resource utilization and power consumption [7].

1.2.1. DSP Block Capabilities:

FPGAs are equipped with dedicated DSP blocks that are specifically created to enhance common signal processing operations, such as FIR and FFT. Although DSP blocks include multipliers, their functionality is not limited to multiplication alone. While it is possible for direct implementation of multiplication in logic using flipflops and LUTs, it can consume a considerable number of resources. Thus, utilizing DSP blocks for multiplication makes more sense in terms of efficiency and performance. That's why even small FPGAs allocate space for DSP blocks.

The DSP multiplier width is determined by the planning of the FPGA: The Altera Cyclone V has a bit size of 27 x 27, the Lattice iCE40UP (SB_MAC16) has a bit size of 16 x 16, the Lattice ECP5 (sys DSP) has a bit size of 18 x 18, the Xilinx 7 Series (DSP48E1) has a bit size of 25 x 18, and the Xilinx Ultrascale+ (DSP48E2) has a bit size of 27 x 18.

Table1.1 DSP48 Slice Comparison Between standard Basic FPGAs

XtremeDSP DSP48s Slice Comparison			
Function	**Spartan®-3A DSP DSP48A**	**Virtex®-4 FPGA DSP48**	**Virtex-5 FPGA DSP48E**
Multiplier	18 x 18	18 x 18	25 x 18
Pre-adder	Yes	No	No
Cascade inputs	One	One	Two
Cascade output	Yes	Yes	Yes
Dedicated C input	Yes	No	Yes
Adder	2 input 48-bit	3 input 48-bit	3 input 48-bit
ALU logic functions	No	No	Yes
Pattern detect	No	No	Yes
SIMD ALU support	No	No	Yes
Carry signals	Carry in	Carry in	Carry in and out
RTL support	Main functions + pre-add	Main functions	Main functions

The number of multipliers in the Altera Cyclone V 5CSEBA6U23I7 is 112, in the Lattice iCE40UP iCE40UP5K it is 8, in the Lattice ECP5 LFE5U-85 it is 156 and in LFE5U-25 it is 56. In the Xilinx-7Series, the Artix7-A200T has 740 multipliers and the Spartan7-S25 has 80 multipliers. Finally, the Xilinx Ultrascale+ Artix AU10P has 400 multipliers.

DSP slices are incorporated in FPGAs to carry out signal processing operations, with the MAC operation being the most frequently employed DSP operation. A MAC block is utilized as a fundamental component for developing more intricate DSP applications, such as filtering.

There are 3 variants of DSP slices used in Xilinx FPGA's- DSP48A, DSP48 & DSP48E. The "DSP area" in a high LUT count or high-end FPGA typically refers to "DSP slices," which can range from a handful to thousands, depending on the type of FPGA. The "DSP" in "DSP slice" stands for "Digital Signal Processor," and these slices are usually comprised of distributed blocks of hardware accumulators and multipliers with broad functionality that enables rapid execution of MAC and SIMD operations on wide data inputs. Like the LUTs in an FPGA, DSP slices can typically be combined for greater power or tailored to suit the requirements of a particular application.

1.2.2. FPGA fabric and core:

This refers to the internal organization of an FPGA, including its different components, types of connections, and other functionalities.

For instance, a typical FPGA manufactured by Altera/Xilinx features a CLB consisting of a LUT that could be configured for 2-to-7 inputs; an adder; and a D flip-flop. The FPGA comprises several logical blocks that work together to perform various functions. In addition to the CLB, the FPGA may include other blocks like PLL, input/output buffers, Block Random Access Memory (BRAM), and DSP blocks. Some FPGA models may also have Macro blocks that serve as controllers for functions such as PCIe and Gigabit Ethernet, as well as SERDES. Recently, FPGA

manufacturers have introduced even larger blocks, such as video-codecs like h.264 & h.265 in Zynq FPGA Ultrascale+. RTL can be used for all blocks programming and configurable, but they need not be constructed as is typical with the ASIC flow shown in Fig 1.6.

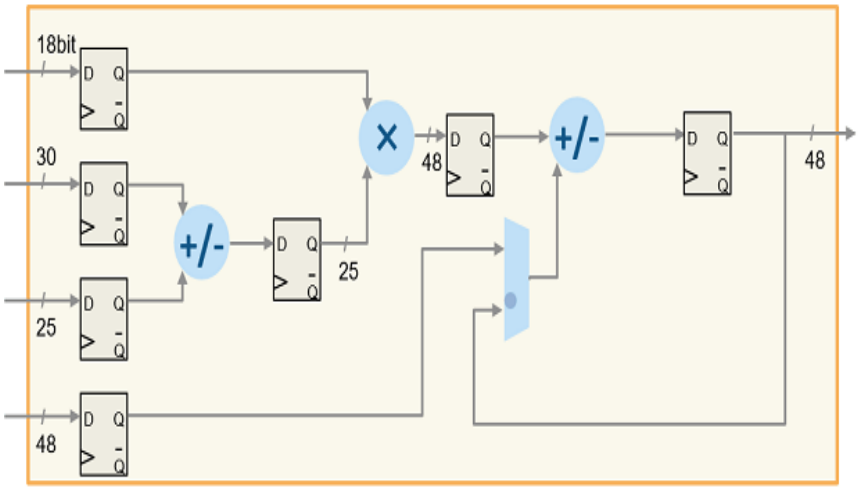


Fig.1.6. Xilinx DSP architecture

1.2.3. Architectural highlights of the DSP slices:

The XtremeDSP slice provides a multiplication function that operates on two complement 18-bit inputs and delivers a result of 36 bits with full precision. This result is then extended to 48 bits while preserving its sign. The XtremeDSP slice has a versatile 48-bit adder/subtractor with three inputs that can be configured for registered accumulation feedback. Additionally, it has more than 40 user-controlled operating modes that allow the function of the XtremeDSP slice to change dynamically between clock cycles shown in Fig 1.7.

To aid in the propagation of input samples, the 18-bit B bus can cascade, while the 48-bit cascading P bus facilitates partial propagating output results. The system also provides support for multi-precision multiplication and arithmetic operations, with the ability to shift operands to a 17-bit right alignment for wide multiplier partial products, regardless of whether sequential or parallel multiplication is utilized.

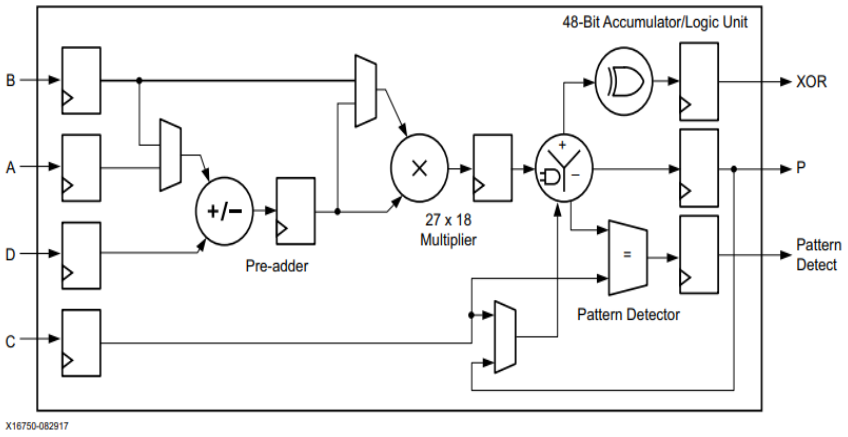


Fig.1.7. Functionality of Basic DSP48E2

For greater computational precision, the XtremeDSP slice also offers symmetric intelligent rounding support. Configuration bits that are selectable enable the setup of pipeline options to enhance the performance of control and data signals. In general, stored port "C" is utilized for add, multiply, flexible rounding mode or large three-operand addition, while separate clock enable and reset signals are provided for the control and data registers.

The DSP48E2 can perform a variety of independent operations, including but not limited to multiply, MACC, multiply add, barrel shift, four-input add, magnitude comparison, wide-bus multiplexing, bitwise logic functions, pattern detection, wide XOR, and wide counting.

The DSP slice comprises of a multiplier-accumulator. To achieve maximum speed during multiply and multiply-accumulate operations, at minimum three pipeline registers are necessary. In the first stage of the multiply operation, two partial products are generated that must be added together in the next stage.

Within the DSP48E2 slice, there is a 4-input adder/ subtractor/accumulator as well as a two-input multiplier. Multiplexers are situated between the two. The inputs of the multiplier are asymmetrical, and the multiplier can accept a two's complement operand of 18 bits as well as

another two's complement operand of 27 bits. Two partial products are produced by the multiplier, each yielding a result of 45-bit two's complement. These PP are lengthened to 48 bits using the X & Y multiplexers before being input to the 4-input adder for concluding sum. The adder functions as a three-input adder when the multiplier is used. Consequently, the output of the multiplication operation is a 45-bit value that is extended to 48 bits while preserving its sign.

The DSP48E2 slice has a 17-bit right wire shift capability that enables multi-precision arithmetic. This feature allows a PP produced in 1 DSP48E2 slice to be right associated & combined with the next PP calculated in a neighboring DSP48E2 slice. By utilizing this approach, it becomes possible to build larger multipliers using several DSP48E2 slices.

1.3. Need of FPGA Multiplier

Multiplication operations can consume significant time and often dictate the overall system performance. When calculating the impulse response or Fourier transform of a discrete signal, a large number of samples must be multiplied and added. This has led to digital signal processor (DSP) chip performance being evaluated based on their multiplication and addition (MAC) numbers per second. FPGAs, which offer customizable solutions, are widely used in applications that require extensive MAC operations, such as military radar systems, adaptive noise cancellations, machine vision, and HDTV. The FPGA architecture offers multiple ways to perform multiplication operations, allowing them to be implemented in a parallel or pipelined manner for faster processing or with a smaller footprint to reduce cost, depending on the specific application.

The key strength of FPGA technology lies in its flexibility, which allows for the design of a wide range of circuits. While the design process may be time-consuming, it offers the freedom to create virtually any design. Multiplication is a frequently used arithmetic operation across various applications, including image and video processing as well as machine learning. To enable high performance multiplication, FPGA vendors offer

digital signal processing (DSP) blocks, which have a fixed location and a limited number. However, using DSP blocks for smaller bit-width multiplications may lead to additional routing delays and inefficiencies. As a result, FPGA vendors supply optimized soft IP cores for multiplication as an alternative solution.

1.3.1. Multiplier Types:

Soft-multiplier:

The LUTs stored in memory represent the multipliers and hold all feasible partial outcomes of multiplication. There exist five flexible modes for multipliers.

a. Semi-parallel multiplication
b. Parallel multiplication
c. Fully variable multipliers
d. Hybrid multiplication
e. Sum of multiplication

Multipliers can be realized using DSP blocks, logic resources or embedded multipliers:

Dedicated embedded multipliers, DSP blocks or logic resources are used to implement these multipliers, utilizing mega functions such as lpm_mult, altmult_add, or altmult_accum.

Firm multiplier:

The implementation of these multipliers involves a combination of embedded multipliers or DSP blocks along with logic sources.

Parallel multiplication mode: is useful for achieving high-speed data scaling since it allows multiple memory blocks to generate a single multiplication result every clock cycle.

The semi-parallel type of multiplication mode: one multiplication per memory block with a multi-cycle operation is generated. This mode is

suitable for coefficient updates in applications like LMS and equalizers.

The "sum of multiplication" mode: utilizes a single memory block or a set of memory blocks to derive the total of the multiplication outcomes. This mode is especially fitting for applications such as DCTs and FIR filtering.

The hybrid multiplication mode: merges the sum of multiplication and semi-parallel modes to achieve an optimized operation. This mode is highly effective in performing complex number multiplications required for complex FFTs, IIR filters.

The fully variable multiplication mode: is an ideal choice for implementing soft multipliers that have varying input data and coefficients, especially for low-resolution multiplication functions.

1.4. Review of Literature

The rapid development of semiconductor technology has led to the requirement for incorporating portable and embedded Digital Signal Processing (DSP) systems. The multiplier holds a significant and indispensable position in nearly every DSP application. Hence, high-speed, and low-power multipliers are a prerequisite for swift DSP. The array multiplier is a simple and fast multiplier, owing to its uniform configuration, and it can be conveniently created. Essentially, array multipliers are utilized for the multiplication of unmarked numbers, employing a consistent arrangement of full and half adders.

As it is commonly known, multiplication is one of the fundamental arithmetic operations that necessitates an abundance of hardware resources. In order to enhance the speed and lower the power consumption in digital signal processing applications, various multipliers have been implemented by researchers, utilizing cutting-edge technology. The performance of the multiplier is primarily determined by the PPs, as the delay in calculations escalates with the accumulation of PP. Thus, the selection of a serial or parallel multiplier is based on the particular requirements of said parameters

and the application at hand.

In current times, multipliers hold a pivotal position in digital signal processing and other related applications. The conventional multiplication method follows an "add and shift" algorithm, whereby N-bit multiplicands and M-bit multipliers give rise to N*M partial products. In charge of generating the PP, "AND" gates are employed. The dissimilarity between the distinct architectures of multipliers lies in the manner in which PP are generated and aggregated.

Optimization at all stages of the design process is crucial to minimize the power consumption of digital systems. These optimization processes encompass the machinery employed to execute the topology and style of digital circuits, the structural design employed to implement the circuits, and ultimately, the highest-level algorithms that are being executed. In any digital circuit design, digital multipliers are the components that are utilized most frequently. Efficient, swift, and relatively dependable elements, digital multipliers are utilized to perform various operations in digital circuits. Several types of multipliers are available depending on the arrangement of components. In general, the selection of a particular multiplier architecture is contingent upon the requirements of the application at hand.

As it operates through the critical delay path and eventually controls the performance of the algorithm, the multiplier is a crucial component of any digital signal processing (DSP) algorithm. The multiplication operation speed is also of utmost significance in both DSP and general processors. Previously, multiplication was accomplished by a series of shift, addition, and subtraction operations.

There are various algorithms available in literature for multiplication, each offering unique benefits while also having trade-offs in terms of speed, circuit complexity, area, and power consumption. The multiplier is a crucial component of a computing system, and its circuitry is proportional to the square of its resolution, meaning an n-bit multiplier comprises n^2 gates. In

digital signal processing (DSP) applications, latency and throughput are the two primary concerns for multiplication algorithm performance. Latency represents the actual delay involved in computing a function, while throughput measures how many multiplications can be performed in a given period. As a high-delay block and significant power dissipation source, the multiplier is of particular interest to those aiming to minimize power consumption by using various delay optimization techniques to reduce delay.

[8] presents Binary floating-point arithmetic for various applications such as banking and insurance, with new software packages in C. In order to achieve additional enhancements in performance through algorithm and code optimizations, as well as to increase functionality, it is imperative to include hardware support incrementally, particularly as the demand for decimal floating-point performance grows. According to comparisons, as presented in [9], the proposed design is not a suitable substitute for binary CSA and CLA trees in terms of delay, area, and power.

ETA can be used in fields where precision is not a strict necessity, and fast processing and low power consumption are more significant than accuracy. This method can be employed in digital signal processing (DSP) applications for portable devices like laptops and cell phones, as cited in [10]. Although multipliers may experience a performance gain, this comes at the expense of consuming more FPGA logic cells. Nevertheless, Altera multipliers are smaller in size than the proposed multipliers described in [11]. In fact, the partial product reduction tree (PPRT) of Altera multipliers consumes fewer logic cells in comparison to the PPRT of the proposed multipliers.

In the study mentioned in [12], the emphasis was on the creation of a multiplier as an IP with TSMC 130nm CMOS technology. However, it would be more beneficial to incorporate the multiplier into a Multiply and Accumulate (MAC) unit capable of performing both multiplication and accumulation operations. However, incorporating additional error

correction and detection circuits, as done in [13], may result in increased power consumption and circuit area usage. In [14], 16 × 16 bit approximate multipliers were implemented in VHDL, and their delay, area, and power consumption were analyzed using Synopsys Design Vision based on STM 65-nm process. The results show improved performance compared to other approximate multipliers, but the area usage in the simulation level itself is high. The designs proposed in [15] may have practical applications in other arithmetic circuits for inexact computing applications. However, no error indicator was discussed for the required applications, and FPGA prototype applications were not included, which could be expanded to DSP compacting operations as described in [16].

Although error correction methods can significantly reduce errors in digital circuits, they often come at the cost of increased power consumption. Further research is needed to study the impact of changes in technology trends, chip area, power consumption, and the scalability of error correction methods to higher bit sizes. Additionally, comparisons between the performance of AlexNet and LeNet schemes have been conducted on CPUs/GPUs and FPGAs, but results for the comparison with CPUs/GPUs are not available in [17], [18], [19]. Resource utilization on Virtex boards was found to be higher compared to Zynq 7k boards, with DSP blocks and LUTs occupying more resources on Virtex boards.

In reference [20], the proposed multiplier has the same delay as the standard approximate multiplier but takes up more area than the lateral. Meanwhile, the design presented in [21] has not been evaluated on any FPGA prototypes. It was implemented on a Virtex-5 FPGA, and the results indicated that the slice occupancy was high and dependent on the word length and multiplier operands length [22]. However, it is anticipated that the area limitation can be overcome with a small FPGA area overhead, such as additional DSP inputs/outputs and global routing wires [23]. Verification of the improvement over the standard parallel Booth multiplier in terms of area-time, depending on the input set, is necessary to develop applications

and demonstrate the advantages of two-speed optimization techniques [24].

According to [25], modular reduction for Type I pentanomials is more complex than for other types, which makes bit-parallel multipliers based on Shifted Polynomial Basis (SPB) less efficient using the same SPB/Montgomery parameter. However, the comparison presented in the paper is based on simulation, and there is no discussion about the proposed design architecture. The paper also lacks information from a hardware perspective. One disadvantage of the DSP-based multiplier is that it still requires 4 DSP primitives for operand sizes ranging from 19 to 32 bits, which leads to higher routing delays. The proposed method needs to be optimized for ease of use. The implementation was conducted on a mid-family Xilinx Spartan kit [26][27].

The design suggested in reference [28] demonstrates an accuracy level comparable to that of a 25% error rate and an equivalent positive and negative absolute error deviation of 1. However, the following architecture was produced and implemented using 45-nm technology at the transistor level, and solely simulation results were compared utilizing Cadence. The demonstration presented in [29] featured nine configurations of the public key cryptosystem and compared them to a software implementation utilizing the gf2x C library, which adds delay to the computations. While accurate results were published for various word lengths of multipliers, including the DSP blocks, the applications were not verified using the proposed multipliers [30].

The outcomes presented in [31] were obtained through simulations using TSMC-180nm, but there is no indication of the tools and prototypes used in the study. The implementation described in [32] was conducted on various FPGAs, and the design exhibited higher delay on Virtex-5 than other prototypes, while occupying the same area as the conventional design on Cyclone IV. Therefore, the parameters are affected by the board used. The results in [33] do not reveal any improvements in LUT/area occupancy. Compared to a binary multiplier in [34], the larger multipliers have a similar

area, but their delay is worse. Both the area and delay ratios decrease as the operand size increases. Efforts are still required to concentrate on LUT-level optimizations for the creation of additional resource-efficient and high-performance circuits for both accurate and approximate arithmetic operations, such as multiplier accumulators (MACs) and dividers.

[35] present Innovative softcore multiplier architectures leverage FPGA's inherent features like look-up table (LUT) structures and fast carry chains. These designs prioritize area optimization, low latency, and accuracy, effectively reducing critical path delay and resource consumption.

The evolution of high-speed serial data communication technology has resulted in the development of complex system architectures capable of transmitting data at multi-Gbps rates. The performance evaluation of such systems is typically conducted using industry-standard methods, which entail employing a pseudo random bit sequence (PRBS) as a test stimulus and determining the bit error rate (BER) as one of the primary performance metrics. Nevertheless, effectively generating appropriate test vectors and precisely assessing the system response poses a substantial challenge, as per [4].

Evaluating the performance of a serial interface often requires using specialized testing equipment such as a pseudo random bit sequence (PRBS) generator, a bit error rate tester (BERT), and a spectrum analyzer to measure the bit error rate (BER) and analyze the eye diagram. However, on-chip built-in self-tests (BIST) can be employed that include a PRBS generator and a bit error checker to assess the performance of the design under test (DUT). The BIST approach has several advantages such as reducing testing time and costs associated with external test equipment. Additionally, it can minimize the number of I/O pins required for evaluating the system's performance. Fig 1.8 shows the functional block diagram of a built-in self-test (BIST) used for testing a serial data communication system. The BIST involves subjecting the design under test (DUT), which is an on-chip transceiver communication system capable of operating at data rates up to

10 Gbps, to a pseudo random bit sequence (PRBS) as the input stimulus. The PRBS checker then analyzes the DUT's response to calculate the bit error rate (BER).

Wei-Zen Chen and colleagues proposed a PRBS generator that functions like a LFSR with a 1 to 16 de-multiplexer. This architecture reduces the clock rate by 16 times and is more suitable for loop back tests while consuming less power than traditional architectures. Additionally, they proposed an 8-phase clock multiplier unit with a clock rate of 1.25 GHz [36].

Fig.1.8. Block diagram of BIST for serial interface.

In their study, Wang Ying and co-workers introduced a low-power PRBS generator and checker capable of operating at 2.5-Gbps with a wide operating range. The PRBS checker applies dynamic detection theory for self-synchronization and can serve as a key component for testing high-speed single-channel SerDes when combined with the PRBS generator. With further improvements, it could also function as part of a test loop for multi-channel SerDes.

Leon Pavlovic and his team proposed a novel design and technique for a PRBS generator that can function at a bit rate of 10 Gb/s. The generator is created using a single-stage LFSR with only one retiming stage, which is more effective and permits a higher clock frequency compared to conventional multi-stage implementations. This method can also be utilized for the polynomial divider in a PRBS receiver. At the middle of the eye crossing, the data generated by the proposed PRBS generator has a low timing jitter of 1.6 ps rms, and the length of the generated pseudo-random pattern is 2^15-1.

A team of researchers led by Ram Ratnaker has developed a self-

testing mechanism for high-speed serial communication, which consists of a PRBS generator and a bit error checker. This system is capable of precisely detecting bit errors and assessing the system's bit error rate (BER) performance. Additionally, the output of the bit error checker can be analyzed after testing to evaluate the performance of the system. The PRBS generator and bit error checker both operate at half-rate and use a uniquely generated pattern for synchronization. Their design has shown reliable performance in testing high-speed serial communication systems [37].

Zhiqian Zhang and co-authors presented a BIST technique that utilizes a cell repeat pattern and fault model to generate efficient test sets using software simulation. This method is easy to implement on FPGA and has minimal hardware requirements, making it suitable for different multipliers with varying coding and compression circuit structures. The approach achieved 99.57% fault coverage for an 18 x 18 Booth2 & Wallace tree multiplier, whereas the multiplier test based on exhaustive method would need 236 test vectors [38].

A technique for testing array multipliers called pseudo-exhaustive testing (PET) was introduced by A.S. Oyeniran and colleagues. This approach involves transforming a 2D array into multiple 1D arrays, which simplifies the process of synthesizing PET patterns for testing. The design of the circuit allows for segmentation based on data control, which permits individual testing of each 1-bit adder within the array. The proposed approach utilizes a blend of deterministic generation and pseudo-exhaustive testing, which allows for targeting distinct types of fault classes, including the difficult-to-test faults. Based on the experimental results, the PET-based BIST approach demonstrated better performance compared to the traditional LFSR-based BIST in terms of power consumption and area overhead. Moreover, the regular structure of PET provides more efficient solutions for SBST [39].

P. Bhaskar Reddy and colleagues [40] proposed a novel architecture for the TPG by integrating the conventional-LFSR with a control logic

module and a bit interchanging module. This modified TPG architecture was used to evaluate high speed multipliers and evaluate their performance. The proposed TPG architecture offers the advantage of compatibility with other low power techniques to achieve further reduction in power consumption.

In their work, N. G. Padharpurkar and V. Ravi [41] proposed a novel architecture for BIST that employs a self-checking technique, which showed better results compared to pseudorandom BIST. The suggested technique utilizes PE-ATPG for producing a pair of test vectors with the lowest possible Hamming distance, resulting in notable reduction in power consumption. However, the self-checking technique increases the area overhead by 30%-35%. The newly proposed BIST approach, combined with a self-checking mechanism, has been shown to effectively identify faults up to a maximum of 10 while maintaining a 70% chance of detecting any errors when applied test vectors are used. This approach enables comprehensive testing of array multipliers.

In their research, T. Srinivasa Rao, and colleagues [42] developed a BIST-based Test Pattern Generator for Vedic multiplier that significantly reduces switching activities during pattern generation. Additionally, they were able to increase fault coverage by extending the number of clock cycles performed by the binary counter. The power consumption of various test pattern generation techniques was assessed and compared to the most recent approach. Specifically, the latest method was employed to implement BIST for a low power test pattern generator for Vedic multiplier. The findings revealed that this implementation resulted in reduced power consumption while enhancing fault coverage as compared to prior approaches.

In their work, B. Mishra, and colleagues [43] presented a modified design for the test pattern generator used in BIST. This design can be customized to generate extended bit sequences of random numbers by integrating multiple sets of comparable logic circuit architectures. Moving forward, it is possible to tailor both the multiplier and test pattern generator to meet the application-specific needs of BIST-based hardware design

implementation.

M. D. Pulukuri et al. [44] conducted an assessment on several test techniques and methodologies that were proposed earlier for several types of multipliers. The authors proposed novel methods for effectively testing multipliers, irrespective of their architecture, achieving over 99% single stuck-at gate-level fault coverage by utilizing a simple 8-bit or 9-bit binary up-counter in conjunction with a small number of multiplexers. They also discussed evaluating the multipliers that are prevalent in contemporary FGPAs.

Dimitris Bakalis and his team conducted an analysis of the testability of Wallace tree summation units and suggested design rules to ensure full testability in the cell-fault model when using an 8-bit counter TPG to drive the multiplier inputs [45]. Additionally, the authors introduced a method for implementing a low power BIST approach that considers both quality and cost-related concerns. This methodology offers the BIST designer various solutions that have distinctive characteristics in terms of power reduction, implementation area, and fault coverage.

A.S.Oyeniran and colleagues suggest various PET methodologies to enhance their efficacy. The proposed dataset is organized in a consistent manner, making it compatible with standard processors and DSPs for use in both hardware-based logic BIST and software-based self-testing [46].

Pushpraj Singh Tanwar and Priyanka describe the use of VHDL to implement BIST logic. LFSR serves as a pseudorandom sequence generator, while signature analysis is used to verify the circuit's accuracy. If the signature mismatches with the reference signature, the circuit is deemed faulty. Nonetheless, there is a low likelihood that the signature of a malfunctioning circuit will be identical to that of a functional circuit. Using a longer sequence, signature analysis provides greater fault coverage [47].

Dileep and Ghanshyam have created a novel memory Built-in Self-test and Repair system that does not require additional rows or columns. This

approach focuses solely on verifying the memory addresses, eliminating the need for additional space as in previous research. The proposed VHDL design incorporates the LFSR of BIST, emphasizing the LFSR's power. By utilizing LFSR with minimal power consumption, it can be concluded that the power in BIST implementation can be further reduced [48]. [49] Theano, a Python compiler for mathematical expressions, merges NumPy's simplicity with optimized machine-level speed. By incorporating Theano into the design of multiplier architectures, users can compose high-level mathematical expressions for operations within the multiplier circuit.

In AI-focused FPGA-VLSI applications, approximate multipliers emerge as the swiftest options. A study suggests a 64-bit approximation to a 64-bit multiplicator. Using a partial product or dada-based method, the 64-bit multiplication completes in stages. This approach reduces area by up to 55%, latency and PDP by around 70% compared to prior work, with 99% accuracy. The simulation employs a Virtex 7 FPGA [50]. [51] introduces an FPGA-tailored approximate multiplier for deep learning hardware accelerators, aiming to improve efficiency without sacrificing accuracy. Using a parallel architecture with configurable adders, it evaluates INT8 and UINT8 configurations. Results show a 9% reduction in LUT utilization with minimal impact on accuracy post-approximation retraining. [52] article introduces a highly reconfigurable FPGA accelerator, enhancing processing speed and power efficiency for CNN inference. Through optimized hardware, focusing on throughput and power reduction, strategies like minimized data transfer and energy-efficient techniques are applied. Results on a mobile FPGA-SoC platform show improved throughput, reduced power consumption, and increased hardware utilization, enabling real-time object detection at 9.17FPS.

[53] study examines High-Level Synthesis for fast FPGA accelerator design in Model Predictive Control. We optimize linear algebra operations for low-latency real-time operation with reduced design effort. Insights and guidelines for tuning algorithms enable cost-effective FPGA accelerators for

control timing intervals with minimal coding effort.

Table1.2 Literature Survey

Invest-igator	**Experimental design technique**	**Input parameter considered**	**output (response)**	**Findings of the Study**
[8]	IEEE 754R Decimal Floating Point	non-zero decimal floating-point numbers with coefficients having at most p decimal digits stored as binary integers	Calculate the sum of two decimal floating-point numbers rounded to nearest to p decimal digits, and determine its exactness.	Binary floating-point arithmetic for various applications such as banking and insurance, with new software packages in C
[9]	The reduction of partial products is implemented in a tree structure based on a decimal multioperand carry-save addition algorithm that uses unconventional (non BCD) decimal-coded number systems.	9 or 8 digits coded in (4221) or (5211)	estimated the area and delay of the SD radix-10 and SD radix-5 decimal multipliers for 16-digit (64-bit) BCD input operands	Two different SD encodings for the multiplier that lead to fast parallel and simple generation of partial products.
[10]	simulated the ETA along with four types of conventional adders, i.e., the RCA, CSK, CSL, and CLA, using HSPICE. All the circuits were implemented using Chartered Semiconductor Manufacturing Ltd's 0.18-m CMOS process.	100 sets of inputs were randomly created using the C program "random()" function	ETA performed the best in terms of power consumption, delay, and PowerDelay Product (PDP). The PDP of the ETA is noted to be 66.29%, 77.44%, 83.70%, and 75.21% better than the RCA, CSK, CSL, and CLA, respectively	Extensive comparisons with conventional digital adders showed that the proposed ETA outperformed the conventional adders in both power consumption and speed performance.

[11]	multiplier generator tool in C++	Two multiplication algorithms including radix-4 Booth and BaughWooley were implemented	radix-4 multiplication with carry save PPRT unit is the right technique to implement soft multipliers on FPGAs and such multipliers always outperform Altera soft multipliers	explore the design space of multipliers on FPGAs and measure the performance gap between embedded and soft multipliers for different bit-widths
[12]	multipliers using HDL	three multipliers Booth, Wallace and Dadda are implemented and the constraints area, power and timing are optimized using Verilog codes based on software resources NC SIM and VC SIM	the number of intermediate stages increases in multipliers, the interconnection between the building blocks also increases	techniques for optimal computer aided designs of selected three 8-bit multipliers namely Booth, Wallace tree and Dadda by analyzing delay, area and power characteristics with particular emphasis on designing the cells for optimum power using layout design techniques
[13]	error correction mechanism	A Gaussian filter is implemented for image sharpening	The inaccurate multipliers achieve a power reduction of 17.98%-34.15% over corresponding accurate designs with a maximal absolute relative error of 22.2%.	employed an alternative scheme for partial product reduction (PPR) compared to accurate tree based multiplier. An approximate yet simple and fast adder is utilized to replace the conventional

				carry propagation adder in final addition with an error probability of 1/16 and larger size multiplier is built by a recursive pattern
[14]	an approximate multiplier is designed and analyzed with four different approximate adders	Reduction in partial product tree	If high approximation can be tolerated for saving more power and delay Approximation IV has to used. Approximation IV offers 43% area savings and 68% power savings over Approximation I	Approximation III and IV achieve significant reduction in area and power consumption compared with Approximation I
[15]	approximate 4-2 compressors	Simulation results for the approximate compressors and multipliers are provided	The first and second proposed multipliers show a significant improvement in terms of power consumption and transistor count compared to an exact multiplier. The first and second multipliers have larger average NEDs (and thus, larger PSNRs), while the second multiplier that	the proposed designs accomplish significant reductions in power dissipation, delay and transistor count compared to an exact design; moreover, two of the proposed multiplier designs provide excellent capabilities for image multiplication with respect to average

			uses the second proposed approximate compressor for all bits, has the best delay.	normalized error distance and peak signal-to-noise ratio
[16]	2-stage BCD multiplication method	BCD Multiplier with 3 blocks: Circuit, multiplier and Correction	alternative for implementing BCD multipliers is the 2-stage multiplication method, which is composed of a binary product stage for computing the binary multiplication	new architectures for BCD-digit multiplication oriented to area reduction have been presented. These proposals are based on the 2-stage BCD multiplication method, which requires a binary product stage and a binary-to-BCD decoding stage. The design of the binary product architecture takes advantage of the non-used symbols in BCD encoding, thus reducing the required area resources
[17]	Partial Product Perforation technique for designing approximate multiplication circuits	the partial product perforation method is applied to various multiplier architectures in order to explore how their power consumption,	explored product perforation on a large set of multiplier architectures, evaluating its impact on different architectures	the partial product perforation delivers reductions of up to 50% in power consumption, 45% in area and 35% in critical delay

		area, delay, and accuracy behave considering the perforation configuration	and error bounds	
[18]	Vedic IXI Methodology	scope of Vedic mathematics is extended to signed numbers using the concept of RB number system	64-bit Vedic Multiplier is designed and improved for higher speeds	Based on resource utilization, the performance is improved for multiplier
[19]	roofline model oriented optimization model	FPGA-based CNN accelerator that maps all the layers to their own on-chip units and working concurrently as a pipeline	Energy benefit of CPU/GPU	Experimental results show that implementations can achieve a peak performance of 910.2 GOPS on Virtex-7 690t, and 36.36 GOP/s/W energy efficiency on Zynq-7020, which are superior to the previous approaches
[20]	new 4-2 compressor is projected	FIR filter using Proposed Approximate Multiplier	Proposed Approximate Multiplier achieved 3.19% less Area, 53.74% less Delay, 3.79% less Power compared to Approximate Multiplier and 31% less Area, 81.35% less Delay and 59.83% less	Proposed Approximate Multiplier was coded in Xilinx Vivado Version: 2016.4. Area, Delay and Power analysis was done with Cadence RTL compiler

			Power compared to that of an Exact Multiplier and also 4 number of wrong carry outputs in 4:2 compressor compared to the 5 wrong carry outputs of compressor used in previous Approximate Multiplier	
[21]	Utilization of an approximate 4:2 compressor to construct multipliers of varying accuracies, scaled up from 4×4 to 16×16 and 32×32, offering a spectrum of accuracy-performance trade-offs.	Configuration variations in approximate multipliers and application scenarios like image processing and MIMO baseband receiver interference nulling	Performance metrics including Power-Delay-Product (PDP), Mean Relative Error Distance (MRED), and Peak Signal-to-Noise Ratio (PSNR)	The proposed multipliers present diverse accuracy-performance trade-offs, with low-power designs
[22]	Implementation of a high-speed multiplier using a combination of Vedic mathematics principles and a high-speed adder, specifically the Carry Save Adder (CSA)	Utilization of Vedic mathematics sutra called Urdhva Tiryakbhyam for multiplication, starting from a 2-bit Vedic multiplier and ascending to implement a 64-bit multiplier	Hardware platforms implemented on Virtex-5 and Virtex-6 FPGA kits using VHDL code targeted for ISE 14.5, with simulated results obtained via Modelsima 10.3d	Comparative analysis reveals that the proposed multiplier exhibits lower delay compared to previously designed multipliers of the same bit-length. In terms of area utilization, the proposed multiplier consumes fewer hardware resources, with

				a reduction of 13% compared to a reference design
[23]	Evaluation of FPGAs for accelerating HPC workloads using High Level Synthesis (HLS), specifically focusing on the Rodinia benchmark suite as representative HPC applications	Assessment of different benchmarks from diverse domains with varying levels of optimization on FPGAs, utilizing OpenCL and FPGA-specific optimizations	Demonstrated improvement in performance of baseline implementations by up to two orders of magnitude through FPGA-specific optimizations, achieving competitive performance compared to other hardware platforms	Despite functional portability of OpenCL, direct porting of GPU code to FPGAs showed suboptimal performance, necessitating advanced FPGA-specific optimizations for maximal performance gains
[24]	Development of a two-speed, radix-4 multiplier for digital filters and machine learning on an Intel Cyclone V FPGA	Evaluation of bit widths (32 and 64) and input sets (uniform, Gaussian, neural networks) to assess performance improvements over standard parallel Booth multiplier	Demonstrated improvements (1.42×–3.36×) in area-time efficiency over standard parallel Booth multiplier using the proposed two-speed multiplier	The two-speed multiplier shows significant performance gains (up to 3.64×) for various input sets and bit widths compared to standard parallel multipliers
[25]	Development of an efficient MK bit-parallel multiplier for Type II pentanomial under SPB	Formulation of new modular reduction for Mastrovito matrix construction, focusing on Types I, II, and Type C.1 pentanomial	Introduction of a new MK multiplier for Type II pentanomial, achieving similar speed with reduced logic gates	The proposed MK multiplier for Type II pentanomial offers efficient performance with minimal time delay increase and reduced logic gate usage

[26]	Implementing an approximate radix 4 booth multiplier, extending from recent approximate adder design	Evaluating power, speed, and accuracy improvements over conventional booth multipliers and earlier approximate designs	Significant reduction in power (72%) and delay (17%) compared to conventional methods	Proposed approach shows notable improvements in power, delay, and accuracy without additional error correction circuitry
[27]	Implementation of approximate multipliers on FPGAs using newly proposed approximate logic compressors	Accuracies of approximate multipliers, specifically 8-bit, 16-bit, and 32-bit designs	Improved power-delay-area products (PDAP) compared to state-of-the-art works and LUT-based multipliers on FPGAs	Proposed approximate multipliers outperform prior works and exact LUT-based multipliers in terms of electrical performances, PDAPs, and delays
[28]	Design and analysis of two novel approximate 4:2 compressor architectures using 45 nm CMOS technology	Area, delay, and power reduction in the proposed compressors compared to state-of-the-art designs	Proposed approximate 4:2 compressor shows significant reductions in area (56.80%), power (57.20%), and delay (73.30%) compared to accurate compressors	Proposed compressors enable implementation of 8×8 and 16×16 Dadda multipliers with comparable accuracy to state-of-the-art designs, validated in image processing applications like multiplication and smoothing
[29]	Template architecture combining Karatsuba and Comba algorithms for large binary	Design-time parameters control performance and resource utilization	Achieved 3.6x to 33.3x performance speedup compared to optimized	Template architecture enables flexible and scalable multiplier implementation

	polynomial multipliers on Xilinx Artix-7 FPGAs		software using gf2x C library	with significant performance gains across FPGA configurations
[30]	Novel power-of-two multiplier designs evaluated on modern FPGAs	Speed-optimized and area-optimized implementations compared with FPGA IP cores	Proposed multipliers offer up to 4.3x speed increase and reduce resource requirements and energy consumption by up to 22% and 40%, respectively	Novel multipliers demonstrate superior speed and efficiency compared to native IP cores, enabling full exploitation of FPGA computational capability
[31]	Novel approximate multiplier combining radix-4 Booth encoding and logarithmic product approximation evaluated through simulations on TSMC-180nm	Three parameters (d, TH, and TS) used to tune accuracy and design efficiency of the proposed multiplier	Proposed multiplier demonstrates reduced energy consumption and area utilization, with tunable accuracy	Proposed design offers a compromise between energy efficiency and accuracy, making it suitable for error-resilient applications like image processing and convolutional neural networks
[32]	Proposed novel hardware architecture for efficient FPGA implementation of Finite-field multipliers for ECC	Operand sizes variation in FPGA implementation	Lower combinational delay and area-delay product compared to state-of-the-art works, indicating efficiency of design	Proposed method is approximately 30% faster than Karatsuba and 20% faster than overlap-free Karatsuba, with 1% less area requirement than Karatsuba. Additionally, it is approximately 25% more efficient in

				terms of area-delay product compared to conventional methods
[33]	Implementing signed and universal multipliers using FPGA and optimizing LUTs, carry chain, and fast carry logics	Variation in operand types (signed positive, signed negative, unsigned) and operand sizes (8x8 bits)	Logic time delay and LUT occupancy for both serial-parallel and parallel multipliers	Proposed improvements include a new mathematical equation for signed multiplication, architectural design of a universal multiplier, and a control system for adjusting two's complement processes. The universal multiplier can handle all nine types of binary multiplications
[34]	Designing and implementing parallel decimal multipliers on 6-input LUT FPGAs	Operand sizes and formats (BCD or BCD/excess-6) for decimal multiplication	Comparison with state-of-the-art implementations showing 26% better area and 12% better performance for proposed decimal multipliers	Proposed decimal multipliers improve upon existing architectures, offering better area and performance trade-offs for decimal multiplication in FPGA
[35]	Proposing area-optimized softcore accurate and approximate multiplier architectures for FPGAs, leveraging	Various sizes of multipliers and input data formats	Reductions in LUT utilization and critical path delay compared to Vivado's multiplier IP: up to 25% and 53%	Proposed architectures demonstrate significant improvements in resource efficiency and

	6-input lookup tables and carry chains		for accurate multipliers, and up to 51% for approximate multipliers	performance for FPGA-based multiplication, enabling high-performance accelerators for image and video applications

1.5. Research Problem:

One of the intrusive applications such as Cryptography systems have become inseparable parts of almost every communication device. Other applications are streaming of data, processing of data in real-time, encryption-decryption, fast computations using DSP available in fabric. Multiple techniques exist for evaluating the effectiveness of distinct DSP solutions with varying degrees of precision. The Embedded Multipliers Performance method is an example of a technique for comparing DSP performance that is straightforward and doesn't factor in the intricacy and efficiency of the entire DSP design or the supporting architecture around the embedded multipliers. However, this approach is considered to be the minimum accurate among all the specified methods. Another method for performance assessment is the DSP IP Benchmarks, which provides a more accurate comparison of performance between distinct silicon mixtures available. This evaluates the effectiveness of conventional functional processes including FFT and FIR filtering, which are crucial to numerous DSP designs. These are recognized as the prevalent DSP IP benchmarks. The subsequent approach involves Application-Level Benchmarks, which accurately gauge the efficacy of a given silicon solution in executing a particular application. The Berkeley Design Technology Inc. (BDTI) provides a case of benchmarking outcomes. The present study emphasizes the evaluation of performance for both DSP IP/application-level benchmarks. The performance data of DSP IP is based on the multipliers used in standard FPGAs provided by vendors Altera/Intel and Xilinx respectively. We propose a system with a novel approach in which we design

a fast computing and fast processing multiplier as a part of DSP arithmetic block in new generation FPGAs. This will give advantages of less LUT/LE utilization by DSP provide more area on slices/LAB for logic to be implemented supporting intrusive applications.

FPGA devices are comprised of memory and LEs that can be programmed to operate in various modes and functionalities. Designers can use appropriate HDL like VHDL or Verilog HDL to implement any hardware design with flexibility. The FPGA can execute diverse operations such as a JPEG encoder, a DSL modem, a DSL router, a digital broadcast system, or a backplane switch fabric interface. With the advent of high-density FPGAs, including Altera's and Xilinx's FPGA families that integrate different embedded silicon features, designers can now construct entire systems within an FPGA, leading to a SOC implementation. FPGA vendors have introduced embedded silicon features that are optimal for DSP applications, such as DSP blocks, embedded processors, and embedded memories suitable for executing fundamental DSP functions such as arithmetic functions, encoders, decoders, and critical functions of FIR filters, equalizers, FFTs, and correlators.

1.6.Research Gaps:

The identified research gaps can be addressed as follows based on the literature review in this field.

- After reviewing the literature, it has been observed that many of the algorithms and techniques utilized are designed for older versions of FPGA boards with higher NM technology.
- Another disadvantage is the pruning or selection process of the FPGA. Only a few articles showed promising results but are subject to the FPGA selection.
- Most of the research works utilized the basic adders for multiplier designs, but the intrusive applications like deep learning, CNN, High computations need fast processing DSP which adds many advantages over the existing techniques.
- Available DSP blocks are limited as they occupy more fabric area.
- Area and fabric utilization need to be addressed.

1.7.Research Objectives:

From the above motivation the primary aim of this research proposal is to provide an alternate solution to maximize the utilization of the FPGA fabric core assess and improve the Reliability of grid connected PV Inverters. In this regard the objectivesof this research are as follows

1. To design and analyze different conventional and proposed Multipliers on FPGAs.
2. To evaluate the performance of FPGA using the proposed design of multiplier by accelerating typical workload on DSP slices.
3. To compare the potential parameters of proposed method with the existing methods in the literature.

1.8. Proposed Methodology:

Proposed methodology block diagram is presented in Fig. 1.9.

- Stage-I: Analyze the conventional techniques and understand the gaps.
- Stage-II: Design of Proposed technique using HDL/block level logic and synthesis.
- Stage-III: Implementing the proposed designs on UltraScale + FPGAs.
- Stage-IV: Adding BIST for the proposed technique.
- Stage-V: Using the proposed multiplier techniques in one or two intrusive applications.
- **Sample:** Area, Power, and Delay
- **Types of Data:** Any data which needs to be computed and exchanged from IP to IP.
- **Tools**: Any Simulation tool can be used as platform for compilation and simulation. QuestaSim, Xilinx ISE, Vivado, cadence nc simulator.

For Implementation any high computing FPGA can be used.

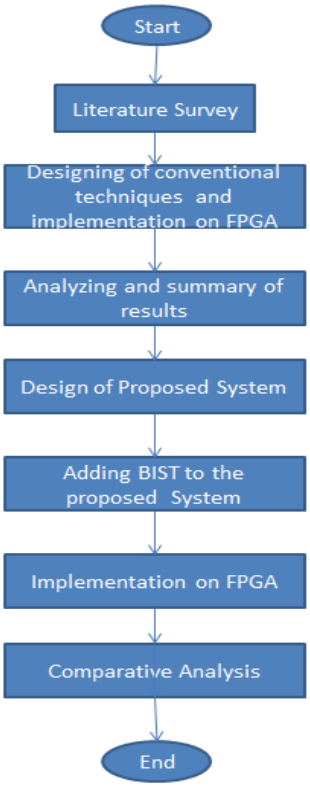


Fig.1.9. Flow Chart of research methodology

1.9.Motivation:

Due to their flexible structure, FPGAs are gaining popularity for computationally intensive applications in DSP. FPGAs consist of on-chip memory and logic blocks that implement the combinational part of the design. The logic blocks contain LUTs and storage elements that are embedded in CLBs. However, this architecture can result in inefficient resource utilization as designs must simultaneously use both resources, which can lead to wasted resources. The purpose of this thesis is to introduce effective techniques for designing with FPGAs that improve resource utilization and optimize the utilization of fabric core resources.

The multiplication-accumulation (MAC) operation is a widely used function in various DSP applications, where input data is multiplied with either fixed coefficients or internal feedback mechanisms. However, the low availability of resources on DSP processors often leads to poor throughput. For instance, a lengthy digital filter necessitates several MAC engines, but conventional DSP processors only have a limited number of MAC processors. Thus, the digital filter is executed in a serial manner, causing lengthy latency and poor throughput since each filter tap necessitates one MAC cycle and must be executed in sequence.

The architecture of a DSP system plays a vital role in determining its performance. Since most DSP operations involve MAC operations, the efficiency of the MAC module is critical. While most processors can perform basic arithmetic operations, the major difference between an FPGA and other general-purpose DSP lies in their proficiency at executing MAC operations. For example, the TMS320C6474 processor is equipped with two multipliers clocked at 1.2 GHz, enabling it to perform 2400 million multiplies per second. In contrast, the Xilinx XC6VLX760 FPGA features 864 multipliers clocked at 200 MHz, providing it the capability to perform 172,800 million multiplies per second. This comparison illustrates the significant advantage FPGAs have over DSP processors in terms of MAC performance.

Improving the efficiency and performance of digital systems can be a daunting task, especially in compact devices. Achieving this necessitates optimization across all tiers of the design hierarchy. Effective architectures are essential at the higher levels, and effective algorithms can aid in decreasing the total power consumption of the system at the lower levels. This dissertation presents several architectures and algorithms that optimize and increase the efficiency of fabric core and LUTs/LEs to enhance their utilization.

1.10. Organization of the thesis:

The subsequent sections of the thesis are structured as follows.

Chapter 2: The chapter provides an overview of FPGA architecture, discussing its historical evolution, key features, applications, and its role in deep learning acceleration. It also touches on FPGA core types and alternatives in FPGA design.

Chapter 3: Explores FPGA high-speed multipliers, addressing performance, resource efficiency, and power optimization. Covers architectures, LUT reduction, innovative techniques, hybrid designs for power and space efficiency, and DSP slice integration for flexibility and energy efficiency.

Chapter 4: Provides an in-depth exploration of Linear Feedback Shift Registers (LFSRs) and their critical role in Built-In Self-Test (BIST) systems. It covers various types of LFSRs, Test Pattern Generation (TPG) methods, BIST architecture, and the significance of area, power, and delay in semiconductor design. The chapter concludes by highlighting the transformative impact of Modular LFSR on low-power TPG and its profound implications for BIST in contemporary ICs and FPGA-based systems.

Chapter 5: Explores FPGA-based real-time applications of a hybrid multiplier, showcasing its benefits in reducing resource usage, enhancing fall detection for the elderly, and optimizing hardware selection for testing.

CHAPTER 2

ARCHITECTURE AND DESIGN PROCESS OF FPGAS

2.1. Introduction

FPGAs revolutionize semiconductor devices in the world of digital electronics by providing a flexible and reconfigurable hardware platform for designing and implementing custom digital circuits. FPGAs are used in a wide range of applications, including telecommunications, aerospace, automotive, medical, and consumer electronics. This report provides an overview of the history of FPGAs, their key features, and their current state of development.

2.2. History of FPGAs

The history of FPGAs dates back to the early 1980s when the first programmable logic devices (PLDs) were introduced. PLDs were simple devices that could be programmed to perform basic logic functions such as AND, OR, and NOT gates. However, they had limited programmability and were not very efficient in terms of performance and power utilization.

In the mid-1980s, the first FPGAs were introduced. These devices had a much higher degree of programmability than PLDs, allowing designers to create more complex circuits. FPGAs consisted of an array of programmable logic blocks (PLBs) that could be connected together to form a custom circuit. The first FPGAs had limited capacity and performance, but they quickly evolved to meet the needs of the growing digital electronics industry.

In the 1990s, the capacity and performance of FPGAs increased significantly, thanks to advances in semiconductor technology. This allowed FPGAs to be used in more demanding applications such as telecommunications and high-performance computing. In addition, FPGAs became more user-friendly, with the development of high-level synthesis tools that allowed designers to create complex circuits using a high-level

programming language.

2.3. Key Features of FPGAs

FPGAs are designed to be highly flexible and reconfigurable, allowing designers to create custom digital circuits that are optimized for their specific application. Some of the key features of FPGAs include:

Programmability: Hardware description languages (HDLs) such as Verilog and VHDL are utilized to program FPGAs. This allows designers to create custom circuits that are tailored to their specific application.

Reconfigurability: FPGAs can be reprogrammed multiple times, allowing designers to test and refine their designs.

High-speed operation: FPGAs are capable of operating at extremely high speeds, making them ideal for applications that require fast processing.

Parallel processing: FPGAs are capable of performing multiple tasks in parallel, making them well-suited for applications such as image and signal processing.

Low power consumption: FPGAs are designed to be power-efficient, making them ideal for battery-powered applications. [57]

2.4. Current State of Development

Field-Programmable Gate Arrays (FPGAs) have evolved significantly, finding applications in high-performance computing, telecommunications, aerospace, automotive, and medical fields. The emergence of deep learning, driven by artificial intelligence, has revolutionized algorithm design. It relies on hardware acceleration, traditionally GPUs, using platforms like NVIDIA CUDA or OpenCL. FPGAs offer a unique middle ground between general-purpose processors (GPPs) and application-specific integrated circuits (ASICs). They combine programmable logic, sequential and combinational logic, and specialized components, offering flexibility and performance. However, FPGA

development presents challenges, including longer design times and hardware description language (HDL) complexity. The new tools aim to simplify FPGA development [51]-[56].

2.5. Application-level Frameworks (High Level Abstraction Tools):

Xilinx and Altera prefer high-level synthesis (HLS) tools, also called high-level design tools, to simplify low-level hardware programming difficulties by converting high-level designs into low-level HDL or register-transfer level (RTL) code. Five primary HLS tool categories exist, including model-based frameworks, high-level language-based frameworks, HDL-like languages, C-based frameworks, and parallel computing frameworks such as CUDA/OpenCL. Parallel computing frameworks are the most practical approach for integrating deep learning and FPGAs, so this review focuses mainly on them.

2.5.1. OpenCL:

The OpenCL platform is a standardized and open-source method for accelerating algorithms on various heterogeneous architectures, including FPGAs, DSPs, GPPs, and GPUs. It leverages the C99 programming language to ensure seamless execution of OpenCL programs across these architectures. OpenCL, like CUDA, provides a standard parallel programming framework that allows low-level hardware access. While both CUDA and OpenCL have similar functionality, they have several notable differences, resulting in varying opinions. Understanding these differences is crucial for future applications of OpenCL in deep learning, especially given that CUDA is currently the dominant choice for popular deep learning tools. [54]-[56]

2.5.2. Development Platforms:

There are various platforms available that cater to different developmental requirements.

AMD, together with its partners in the ecosystem, offers a range of

development environments, hardware and software, and embedded platforms with well-known tools, libraries, and methodologies. The primary aim of these environments is to facilitate and expedite the development process of customized hardware accelerators.

Their services are designed to accommodate a diverse range of developers, including those specializing in AI research, algorithm and application engineering, embedded software development, and traditional hardware development.

- The Vitis™ Unified Software Platform is designed for software developers who are conscious of hardware and allows them to construct accelerators that are capable of running software.

  The Vitis™ unified software platform is a versatile tool that facilitates the creation of embedded software and accelerated applications on different AMD platforms such as SoCs, FPGAs, and Versal ACAPs. With this platform, developers can enjoy a consistent programming model that accommodates edge, cloud, and hybrid computing applications. They can leverage accelerated libraries or construct software using high-level frameworks such as C, C++, or Python. Moreover, they have the option of selecting RTL-based accelerators and low-level runtime APIs that offer more implementation control, allowing them to choose the level of abstraction that best suits their needs.

- The Vitis AI Development Environment, on the other hand, caters to data scientists seeking to speed up AI inference.

  The AMD Vitis™ AI is a comprehensive machine learning development framework, designed to enable superior AI inference capabilities on AMD platforms. The framework consists of multiple optimized DPU overlays, robust software tools, effective software libraries, and deep learning models sourced from a range of industry-standard frameworks, as well as sample designs. This makes it easier

to develop accelerated AI inference capabilities on AMD platforms, with the potential to achieve up to 10 times better performance than CPU/GPU solutions.

- Lastly, the Vivado™ ML is intended for hardware designers to design FPGA fabric. Vivado ML Editions empowers design closure acceleration via ML-based algorithms. This technology incorporates ML-based logic optimization, delay estimation, intelligent design runs, and parallel compilation utilizing Abstract Shell. Such features allow for the definition of multiple modules within the system for incremental and parallel compilation. Furthermore, it facilitates a reduction of 5 times to 17 times in average compile time, in comparison to traditional full system compilation. Abstract Shell also safeguards a user's IP by concealing design specifics outside of the modules, making it vital for applications such as FPGA-as-a-Service and value-added system integrators.

2.6. The mechanism behind the functioning of FPGAs

To understand the workings of an FPGA, it is necessary to comprehend its fundamental structure. FPGAs consist of programmable logic blocks interconnected through programmable routing channels. The programming of these logic blocks involves the use of a hardware description language (HDL) that enables the designer to configure the FPGA to perform specific tasks. Once the FPGA is programmed, the logic blocks can perform various logical operations and can be reprogrammed as needed to perform other operations. The FPGA's flexibility, speed, and power efficiency make it a popular choice for a wide range of applications, including data centers, embedded systems, and digital signal processing.

One can perceive the FPGA as a vast digital circuit breadboard that comprises gates and flip-flops, interconnected by wires spanning the entire chip. By connecting these wires to the appropriate gates or flip-flops, the desired circuits can be created. Unlike a traditional breadboard, FPGAs have

reconfigurable interconnects, allowing for dynamic rewiring. Specialized paths exist for clock signals, with only specific FPGA pins permitted to drive these global clock routing wires.

2.7. Concepts of FPGA core and fabric

The term "FPGA core" can be used in two different ways: firstly, as an FPGA intellectual property core, and secondly, as a core component of a System-on-Chip (SoC) that includes FPGA functionality, such as the Zynq MPSoC from Xilinx or the Stratix SoC and Arria SoC from Intel (previously known as Altera).

An FPGA IP Core refers to a unit in FPGA hardware design that performs a particular function. These cores are designed to serve various functions such as CPU IP core, HDMI IP core, LED blinking IP core, and PCI Express IP Core. Similar to software libraries, IP Cores are integrated into designs to provide specific functionality and undergo thorough testing. In hardware design, IP Cores are employed for the same purpose.

When it comes to System-on-Chips (SoCs), the FPGA core is a section on the silicon that integrates the entire FPGA core onto the die, in addition to a processor core like the ARM Cortex-A series. Xilinx has developed the Zynq series that includes a fully functional FPGA die, a variety of peripherals, and dual-core ARM Cortex-A9 cores. This is an example of a system that combines both FPGA and microprocessor capabilities.

The FPGA core is comprised of a multitude of components beyond the FPGA fabric, which includes the FPGA interconnect matrix and CLBs. To broaden the scope of its practical applications, the FPGA core is supplemented with a range of extra components, which may include PLLs, SerDes, Block RAMs, PCI Express blocks, Multi-Gigabit Transceivers, Configuration Logic, DSP blocks, and other related elements.

2.8. FPGA hard and soft IP:

FPGA cores are considered intellectual property and fall under the standard IP categories of soft, firm, or hard. Soft cores represent processor implementations in an HDL language without comprehensive optimization for the intended architecture, leading to lower performance and efficiency in terms of resource usage. On the other hand, firm cores are also implemented in HDL but are specifically optimized for a particular FPGA architecture to achieve better performance and resource utilization.

Soft Core:

Advantages -

- FPGAs offer high portability compared to other options.
- FPGAs are cost-effective and affordable.
- Due to their simple implementation, FPGAs have access to more low-cost or free sources.
- FPGAs can be easily customized to specific architectures, making them a versatile option.
- FPGAs can be readily modified to meet specific requirements, providing flexibility in their applications.

Disadvantages -

- Enables potential for portability across multiple architectures.
- Lower optimization levels may result in higher resource utilization and reduced performance.
- Additional design effort may be necessary.
- Some architectures may have limited simulation results.
- Documentation for specific architectures may be incomplete.
- Tool set variations used in design implementation can have a significant, unpredictable impact on results.

Firm Core:

Advantages -

- Designed for specific architecture optimization.

- Customizable with ease.
- Performance, resource utilization, and power consumption can be accurately predicted and controlled.
- High level of trust in the functionality and performance of the design.
- The design has been extensively tested in real-world scenarios and verified.
- Testing the design in the target environment is straightforward and easy.
- Simulation resources, such as testbenches and results, are readily available.
- Documentation is available to some extent to aid in the design process.
- **Access to design expertise.**

Disadvantages -

- Manufacturer-developed firm cores may not offer much portability.
- Consistency in documentation, configurability, simulation support, and access to the original design team may vary.
- Advanced design assistance from the IP source may require negotiation of terms and potentially involve fees**.**

Hard Core:

Advantages-

- Obtaining a fixed implementation with optimized design, reliable operation, and comprehensive documentation.
- Similar to purchasing a standard IC component, without the need for further design work.
- Immediate access to fully functional design, without any delays or uncertainties.
- High level of confidence in the design's functionality, with any known issues (errata) already identified.
- Design's parameters such as functionality, performance, and power consumption are well-characterized and measured.

Disadvantages –

- Due to high optimization, it is difficult to achieve equivalent performance or affordability when ported to other targets.
- Vendors have a vested interest in making their cores less portable to other architectures.
- The design is fixed, and adding more instances or making modifications is not possible without changing devices.
- Limited flexibility compared to soft cores.
- May require additional effort for customization and implementation.
- Potential limited documentation or support for customization.

2.9 Improving FPGAs

The improvements made in FPGA architecture, as mentioned above, have significantly increased their power efficiency, performance, and area utilization. However, the requirement of additional overhead to make FPGAs both general-purpose and field-programmable may hinder their applicability in specific use cases. In scenarios where the requirements for cost, power consumption, and performance surpass the FPGA's capabilities, exploring alternative solutions, such as full-custom design and fabrication, cell-based ASIC design, or partially prefabricated "structured" ASICs, becomes necessary. To determine the appropriate course of action, it is imperative to measure the gap in power, performance, and area between FPGAs and ASICs and make an informed decision accordingly.

When there are less DSP slices in FPGA, it can lead to the following issues:

Limited ability to perform complex digital signal processing operations.

- Lower accuracy in signal processing due to fewer resources for precision arithmetic and filtering
- Reduced performance and efficiency of signal processing applications.
- Increased design complexity and development time for implementing DSP functions with fewer resources.

- Inability to implement multiple, concurrent signal processing functions on a single device.
- Higher power consumption and thermal dissipation due to increased utilization of remaining resources to compensate for the lack of DSP slices.

When all DSP slices in an FPGA are used, there will be no more available resources for additional DSP computations. This could result in the inability to implement certain DSP-based algorithms or designs on the FPGA.

The FPGA fabric core cannot be used as soft DSP slices, as they are fundamentally different components with different functionalities and capabilities. The fabric core provides general-purpose logic functions, whereas the DSP slices are specifically designed for high-performance digital signal processing tasks such as filtering, modulation, and demodulation. It is not recommended to use fabric core resources to implement DSP functions, as it may result in suboptimal performance and may also impact the availability of resources for other functions.

Regarding using the fabric core of the FPGA for programming as soft DSP slices, it is technically possible to do so. However, the performance of soft DSP slices implemented in the fabric will likely be lower than the dedicated hardware DSP slices provided by the FPGA.

In terms of writing logic for a DSP slice and implementing it in LEs/LUTs to use it as extra DSP slice, it is possible in some cases. However, the performance of such a design may not be as high as that of a dedicated hardware DSP slice due to the limited resources available in LEs/LUTs. Additionally, the design effort required to implement such a solution may be significant, especially if the logic must be implemented using a large number of LEs/LUTs.

CHAPTER 3

SCALING PEAKS: THE EVOLUTION AND IMPORTANCE OF HIGH-SPEED MULTIPLIERS

3.1. The Role of Multipliers:

Multiplication stands as a foundational arithmetic operation with versatile applications across multiple disciplines. The integration of optimized multiplier circuits within Field-Programmable Gate Arrays (FPGAs) holds paramount significance for various compelling reasons:

Enhanced Performance: Multiplication, a computationally demanding task, can exhibit sluggishness when executed through basic logic gates. FPGAs embed specialized hardware multiplier units designed to expedite multiplication operations markedly compared to software-based alternatives. This acceleration proves especially advantageous in real-time applications and computations necessitating swift throughput.

Optimized Resource Utilization: Implementing multiplication using conventional, general-purpose logic gates can exact a considerable toll on FPGA resources and logic elements. Hardware multiplier units, purpose-built for this operation, facilitate judicious resource employment, thus liberating other FPGA regions for divergent tasks.

Streamlined Complex Algorithms: A plethora of algorithms encompassing domains like signal processing, image processing, cryptography, and scientific simulations entail intricate calculations necessitating multiplication. Tailored multipliers empower these algorithms to be seamlessly executed within FPGA hardware, translating into reduced execution durations, and enabling real-time processing.

Economized Power Consumption: Hardware multiplication circuits frequently outperform their software-based counterparts, yielding superior power efficiency within the FPGA's programmable fabric. This

facet gains particular relevance in devices constrained by battery power or energy availability.

Precision and Fidelity: Precision-sensitive applications such as digital signal processing and scientific simulations demand meticulous accuracy in multiplication operations. Hardware multipliers furnish dependable, precise outcomes, circumventing the potential numerical errors associated with software-based implementations.

Harnessing Parallelism: FPGA-based multiplier circuits excel at parallel execution, facilitating the simultaneous multiplication of multiple operands. This capability accelerates algorithms involving matrix operations, filtering, convolution, and analogous tasks.

Real-Time Data Processing: In scenarios necessitating immediate data processing—such as communication systems, image manipulation, and control mechanisms—hardware multipliers empower FPGAs to adeptly handle incoming data streams, positioning them as a compelling choice for real-time applications.

Tailored Customization: FPGAs empower designers to forge customized multiplier architectures aligned with precise application requisites. This malleability enables optimization catering to area efficiency, processing speed, or power conservation, contingent on project-specific demands.

Mitigated Latency: As multiplication operations frequently underpin diverse computations, executing these tasks within hardware minimizes overall system latency. This reduction in delay translates into more rapid responses to inputs, enhancing system performance.

In summation, the integration of adept multiplier circuits within FPGAs greatly enhances their prowess to tackle a spectrum of applications hinging on multiplication. These span intricate mathematical algorithms to instantaneous data manipulation, ultimately culminating in elevated performance, judicious resource deployment, and heightened power

efficiency.

3.2. The Role of Multipliers in DSP Blocks:

Connecting the concept of multipliers within the context of DSP slices in FPGAs involves understanding the intrinsic relationship between these components. DSP slices serve as specialized units within FPGAs, geared towards accelerating arithmetic operations like multiplication and addition. This association is particularly crucial in domains where signal processing applications heavily rely on these operations. Exploring the correlation sheds light on the synergy between multipliers and DSP slices.

DSP slices, as distinct entities within FPGAs, encompass dedicated blocks optimized for executing specific mathematical operations. The significance of multipliers within DSP slices becomes evident as they play a central role in performing multiplication tasks efficiently and with precision. These multipliers, embedded within DSP slices, cater to the demands of diverse applications in fields like communications, audio processing, and image manipulation, where multiplication is a foundational process.

Considering the operational aspect, DSP slices proficiently leverage multipliers to execute concurrent multiplication operations. Parallelism emerges as a key attribute, where multiple multiplications can be conducted simultaneously on distinct data points or streams. This parallel execution results in enhanced processing throughput, especially relevant in real-time applications requiring prompt data manipulation.

Furthermore, DSP slices extend beyond the notion of standalone multipliers. They encompass additional functionalities such as adders and accumulators. These supplementary components play an indispensable role in signal processing algorithms that demand not only multiplication but also subsequent accumulation or summation of results. This integrated approach streamlines complex computations while optimizing resource utilization.

To harness this relationship effectively, designers must navigate the

nuances of interfacing user-designed or vendor-provided multipliers with DSP slices. This integration entails understanding the input/output configurations, data widths, and synchronization mechanisms specific to the DSP slice architecture. By aligning these aspects, the multiplier can seamlessly collaborate with the DSP slice, augmenting its capabilities in processing-intensive applications.

In summation, the interplay between multipliers and DSP slices within FPGAs underscores their symbiotic relationship. Multipliers, as integral components of DSP slices, empower efficient arithmetic operations that underpin signal processing tasks. This integration enhances parallelism, streamlines complex algorithms, and optimizes resource utilization, contributing to the FPGA's prowess in real-time data manipulation.

3.3. list of multiplier architectures and designs commonly employed in FPGAs

1. DSP Slices Multipliers: FPGAs often feature dedicated DSP slices, equipped with specialized hardware multipliers designed to perform multiplication operations efficiently. These multipliers are optimized for various data widths and are particularly valuable for signal processing applications.
2. Array Multipliers: Array multipliers consist of a grid of logic gates that compute partial products, which are then summed to obtain the final product. While relatively simple, they are versatile and suitable for a wide range of applications.
3. Wallace Tree Multipliers: Utilizing a tree structure, Wallace Tree multipliers reduce the number of partial products by grouping them based on their bit positions. This approach enhances efficiency while minimizing resource usage.
4. Dadda Multipliers: Similar to the Wallace Tree design, Dadda multipliers optimize partial product reduction and adder usage, striking

a balance between area efficiency and performance.

5. Booth Multipliers: Booth multipliers employ a combination of addition and subtraction operations to reduce the number of partial products, which is particularly advantageous for signed numbers and larger operands.
6. Radix-4 Multipliers: Operating on a base-4 number system, radix-4 multipliers offer a trade-off between speed and resource utilization, optimizing partial product reduction for various applications.
7. Radix-8 Multipliers: Building on the base-8 number system, radix-8 multipliers further enhance resource efficiency while maintaining competitive speed performance.
8. Pipelined Multipliers: Pipelined multipliers divide the multiplication process into stages, allowing for improved throughput, albeit with slightly increased latency.
9. Karatsuba Multipliers: Based on the Karatsuba algorithm, these multipliers divide multiplication into smaller, recursive chunks, providing efficiency benefits for larger operands.
10. Montgomery Multipliers: Primarily used in cryptography, Montgomery multipliers exploit mathematical properties to accelerate modular multiplication operations.
11. LUT-based Multipliers: In some FPGAs, Look-Up Table (LUT)-based multipliers approximate multiplication by utilizing combinations of logic gates. While less hardware-intensive, they may offer lower performance compared to dedicated hardware multipliers.
12. Floating-Point Multipliers: Specialized architectures optimized for floating-point arithmetic, offering precise multiplication operations for applications demanding high accuracy.

These multiplier architectures cater to various needs within FPGA designs, enabling efficient and accurate multiplication operations across a

spectrum of applications.

3.3.1. Array Multiplier in FPGA:

Array multipliers stand as essential components within FPGAs, contributing significantly to computational efficiency. These multipliers are designed to perform multiplication operations swiftly and concurrently, making them a crucial asset in various applications that demand rapid arithmetic computations. Fig 3.1 shows the structure of array multiplier.

Resource Optimization: Array multipliers are adept at optimizing resource utilization within FPGAs. They take advantage of the inherent parallel structure to minimize delay and maximize throughput, resulting in more efficient usage of available hardware resources. This efficiency is especially advantageous in applications where speed and resource management are critical.

Real-Time Performance: The parallel architecture of array multipliers enhances the real-time performance of FPGA-based systems. This is particularly beneficial in applications like video processing, communication systems, and control systems, where immediate and accurate calculations are imperative.

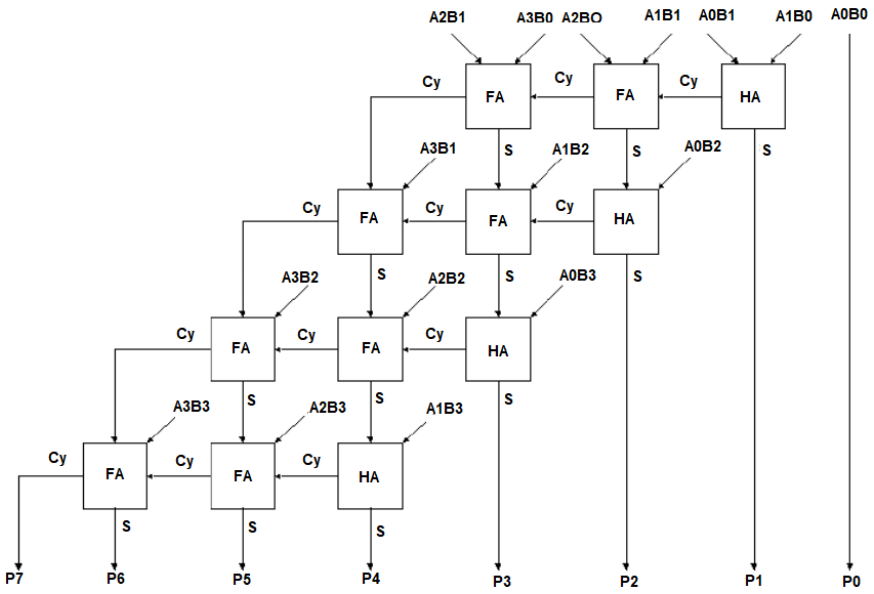


Fig.3.1. Block diagram of array multiplier

Integration with FPGA Development: FPGA development tools provide designers with the ability to easily incorporate and configure array multipliers within their designs. These tools offer options to optimize the use of multipliers, adjust bit widths, and interface with other components of the FPGA.

The array multiplier's role within FPGAs is pivotal, enabling rapid and parallel multiplication operations. Its parallel processing power, resource efficiency, and suitability for image processing applications make it an indispensable element in FPGA-based systems. As FPGAs continue to evolve, array multipliers will remain essential for accelerating arithmetic computations across various domains.

3.3.2 Wallace Multiplier

The Wallace multiplier emerges as a straightforward structural multiplier solution. It boasts the notable benefit of occupying minimal area, distinguishing it from many other multiplier designs. Illustratively, the Wallace tree multiplier adopts a three-stage architecture, as depicted in Fig 3.2.

The operation of the Wallace Multiplier unfolds across these sequential phases:

Stage 1: Generating Partial Products – As the input bit size grows, so does the tally of partial products, expanding the list.

Stage 2: Pivotal Reduction – This stage assumes significance in diminishing the overall array of partial products, consolidating them until their height reduces to two. This process entails amalgamating sets of 3*(number of rows/3) rows simultaneously. The residual rows continue down the cascade until only two rows remain.

Stage 3: Final Addition – Concluding the process, the last two rows are subjected to addition through a carry look-ahead adder mechanism.

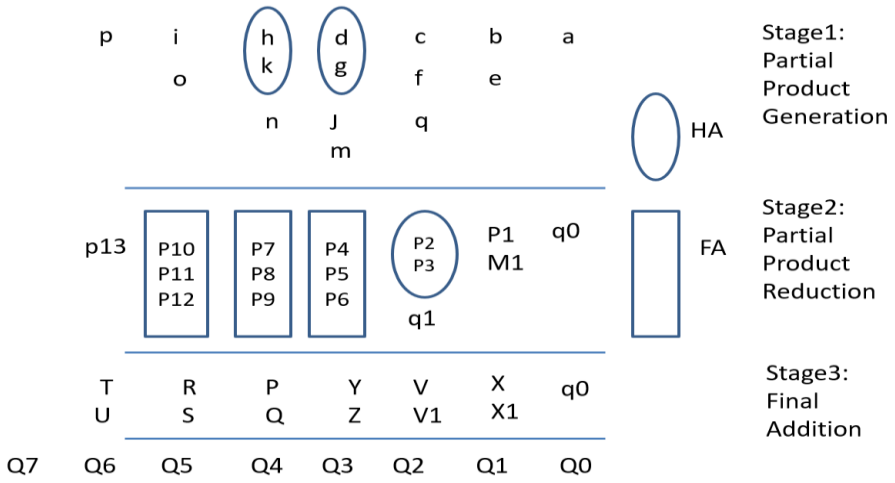


Fig.3.2.Wallace Tree Multiplier

3.3.3. Dadda Multiplier

Dadda multiplier is also a structural multiplier which uses 4 stages to completely execute the multiplication process. Fig 3.3 represents all the stages of the Dadda multiplier as explained below.

Stage 1: Generation of Partial Products.

Stage 2: Delta Formation: Elevating the partial products to create an inverted delta configuration. This establishes the relative height relationship between each column and its adjacent counterparts.

Stage 3: Partial Product Reduction, post the establishment of the inverted delta structure outlined in Step 2. To ensure the minimal number of reduction stages, the height of each intermediary matrix is maintained at a maximum of 1.5 times that of its following matrix. Adders are employed to simplify the partial products, either column-wise or row-wise. Subsequently, terms are arranged in a specific order: the sum is followed by unused partial products or vacant bits, and finally, the carry from the preceding bit. This sequence persists until the matrix height reduces to 2, akin to the Wallace multiplication approach.

Stage 4: Ultimately, the content of two rows undergo addition through mechanisms like carry look-ahead adders (CLA) or ripple carry adders (RCA). The outcome of this addition yields the multiplier's result.

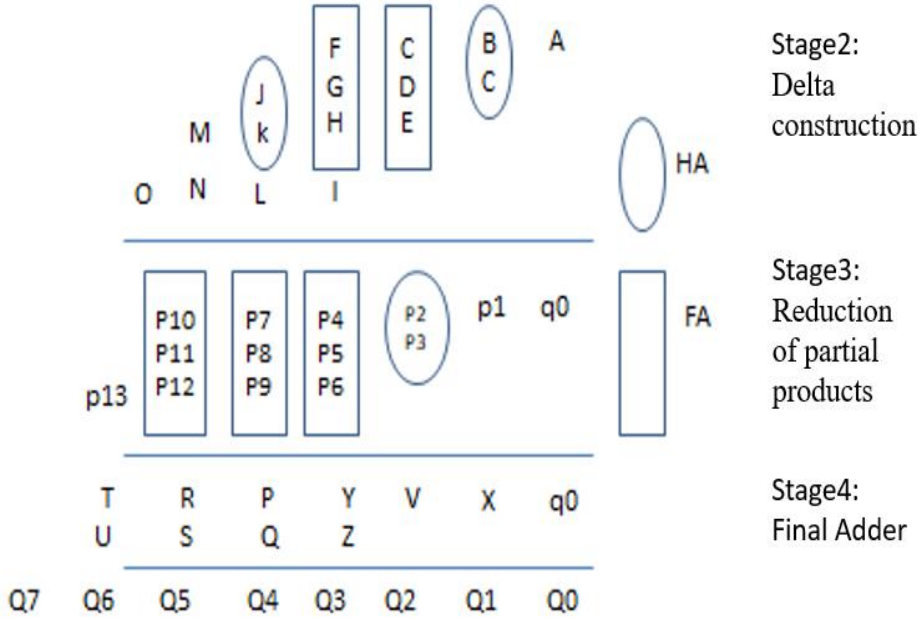


Fig.3.3. Dadda Multiplier

3.3.4. Vedic Multiplier

Utilizing an improved multiplier within the Arithmetic and Logic Unit is paramount for achieving enhanced performance. The Vedic multiplier presents itself as a beneficial multiplication technique that leverages mental calculation principles. The approach employed in this work involves the application of the "UrdhvaTriyakbhyam" sutra from Vedic mathematics, specifically in a 4-bit multiplication process. In this context, "Urdhva" refers to vertical multiplication, while "Triyakbhyam" pertains to crosswise multiplication. Fig 3.4 and 3.5 show the multiplication method.

The Vedic multiplier stands out for its exceptional characteristics, including high speed and reduced power consumption compared to various other multiplier methods. It has become a benchmark for both low power

usage and rapid computation. The operational procedure of the Vedic multiplier is visually illustrated in Figs, highlighting its importance in augmenting computational efficiency.

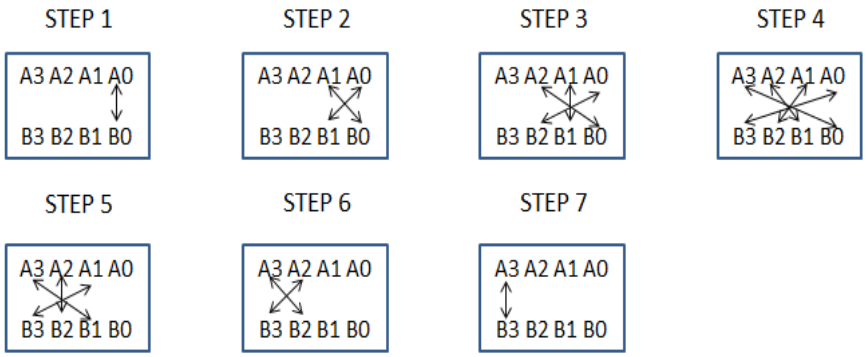


Fig.3.4.Vertical and Crosswise Technique

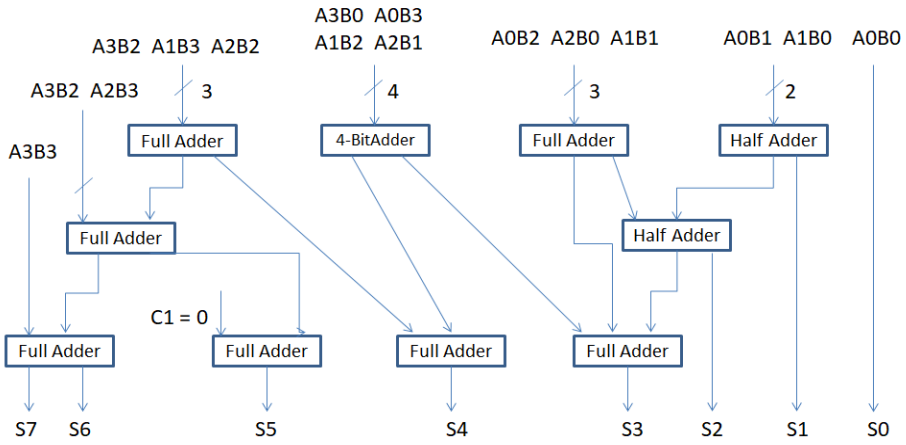


Fig.3.5. 4X4 Vedic Multiplier

3.3.5. Karatsuba multipliers

These are a class of efficient multiplication algorithms that aim to reduce the number of elementary multiplications required for large number multiplication. They are named after Anatolii Alexeevitch Karatsuba, who introduced the technique in the 1960s. Karatsuba's approach utilizes a divide-and-conquer strategy to perform multiplication with fewer multiplication operations compared to traditional methods.

Here's how Karatsuba multipliers work, along with an example:

Working Principle:

Karatsuba multiplication operates on the principle of breaking down large numbers into smaller parts, recursively computing partial products, and then combining these products to obtain the final result. The key idea is to divide the numbers into equal-sized halves and exploit algebraic properties to perform fewer multiplications.

Example:

Let's consider two 2-digit numbers: A = 23 and B = 56.

Step 1: Splitting

Split A into two halves: A1 = 2 and A0 = 3.

Split B into two halves: B1 = 5 and B0 = 6.

Step 2: Recursive Multiplication

Compute P0 = A0 * B0 = 3 * 6 = 18 (elementary multiplication).

Compute P2 = A1 * B1 = 2 * 5 = 10 (elementary multiplication).

Step 3: Recursive Karatsuba Multiplication

Compute P1 = (A0 + A1) * (B0 + B1) = (3 + 2) * (6 + 5) = 5 * 11 = 55 (elementary multiplication).

Step 4: Combining Partial Results

Combine the results: P = (P2 * 100) + P1 * 10 + P0 = 1000 + 550 + 18 = 1568.

In this example, the traditional multiplication would have required three multiplications (A0 * B0, A1 * B1, and (A0 + A1) * (B0 + B1)). However, Karatsuba's approach reduced it to just two multiplications (P0 and P2), resulting in a more efficient computation.

Karatsuba multipliers exhibit improved efficiency for larger

numbers, especially when implemented recursively in hardware or software. By optimizing the number of elementary multiplications, Karatsuba multiplication contributes to faster arithmetic operations in various applications like cryptography, signal processing, and scientific computing.

3.4. DSP Slice Multiplier

The DSP Slice Multiplier constitutes a highly specialized hardware entity meticulously embedded within the realm of FPGAs. Its intrinsic purpose revolves around the rapid acceleration of arithmetic computations, particularly within the intricate domain of DSP applications. This class of multipliers assumes a distinct role, distinctively primed for the execution of multiplication operations with exceptional efficiency, rendering it indispensable for a plethora of tasks ranging from filtering and convolution to modulation and more.

Remarkably distinctive in its attributes, the DSP Slice Multiplier boasts a dedicated hardware presence within the FPGA infrastructure. This dedicated stature translates into a prowess for handling multiplication and accumulation tasks with a degree of finesse that is unparalleled. The multiplier's standout quality lies in its aptitude for parallel processing, resourcefully composing the simultaneous execution of multiple multiplication and accumulation operations. This orchestrated parallelism translates into a remarkable enhancement in processing throughput, effectively catapulting the overall performance of DSP applications.

The versatility of the DSP Slice Multiplier is worth noting, as it gracefully accommodates different operand sizes and operational modes, including both fixed-point and floating-point arithmetic. This intrinsic adaptability equips it to seamlessly integrate into an array of DSP algorithms, each with its unique set of precision prerequisites. Moreover, many DSP slice multipliers come fortified with pipeline capabilities, fragmenting the multiplication process into distinct stages. This strategic

division significantly diminishes latency, ushering in a continuous flow of data through the multiplier.

Intricately entwined with the concept of the Multiplier-Accumulator (MAC) unit, the DSP Slice Multiplier readily undertakes the dual task of multiplication and accumulation within a single operational unit. This duality is of paramount significance for real-time DSP applications where streamlined efficiency is of the essence. Furthermore, it capably addresses the intricacies of bit growth, a hallmark challenge in multiplication and accumulation processes, by ensuring results remain harmoniously within the confines of desired precision boundaries.

This specialized facet of FPGAs forms a cohesive tapestry with the broader FPGA fabric, seamlessly integrating with diverse components such as logic cells, memory blocks, and intricate interconnections. Through this seamless integration, DSP slice multipliers harmoniously collaborate with these components, crafting a unified processing symphony.

3.5. Disadvantages of Existing FPGA Multipliers:

The limitations associated with existing multipliers within FPGAs prompt a thorough exploration of potential improvements. These drawbacks encompass resource consumption, fixed precision constraints, limited adaptability, power consumption concerns, routing intricacies, and potential cost implications. These shortcomings underscore the need for innovative approaches to elevate the efficiency and effectiveness of FPGA multipliers.

1. *Resource Consumption*: FPGA multipliers utilize significant resources such as LUTs and flip-flops, constraining design complexity and flexibility.
2. *Fixed Precision*: Some multipliers offer fixed precision, leading to inefficiencies when handling varying precision requirements.
3. *Limited Adaptability*: Lack of adaptability to specific algorithms can

result in suboptimal performance for certain tasks.

4. *Power Consumption*: Multipliers contribute significantly to power consumption, affecting energy efficiency and battery life.
5. *Routing Challenges*: Complex routing due to high fan-out signals can lead to longer interconnect delays.
6. *Cost Implications*: Dedicated multipliers can contribute to FPGA device costs, especially in resource-constrained designs.

3.6. Improvements and Changes:

- *Resource Optimization:* Developing multiplier architectures that make more efficient use of FPGA resources, such as minimizing LUT usage and flip-flops, can address resource constraints.
- *Configurable Precision:* Multipliers with adjustable precision options allow designers to match precision requirements to specific application needs.
- *Algorithm-Specific Designs:* Tailoring multiplier architectures for specific algorithms, such as DSP algorithms, can improve efficiency and performance.
- *Customizable Architectures:* Multiplier architectures that allow designers to configure operand sizes, bit widths, and functional units can enhance flexibility.
- *Low-Power Techniques:* Incorporating low-power design techniques, like clock gating and power gating, can mitigate power consumption.
- *Parallel Processing:* Exploring parallel processing methodologies within multipliers can improve throughput without significantly increasing resource usage.

3.7. Reducing LUT Usage in FPGA Fabric for Multipliers:

When the available DSP slices prove inadequate and the imperative to minimize LUT (Look-Up Table) usage intensifies, a spectrum of strategies emerges to effectively address this challenge:

Custom Bitwidths: Opting for bespoke bitwidths that precisely align with the algorithm's precision requisites stands as a potent approach. This deliberate selection prevents unwarranted bit proliferation, concurrently mitigating LUT consumption.

Optimized Resource Sharing: Enlisting resource sharing techniques emerges as a strategic maneuver to judiciously utilize LUTs across diverse segments of the design. This tactic optimizes the allocation of resources, engendering a harmonized distribution of LUT consumption.

Partial Reconfiguration: The dynamic spectrum of partial reconfiguration unfurls as a formidable recourse. By selectively loading distinct multiplier configurations in response to varying processing demands, FPGA fabric can be maximally leveraged, safeguarding efficient resource utilization.

Algorithm Approximation: The realm of algorithmic approximation surfaces as an avenue to navigate LUT utilization complexities. By introducing algorithmic approximations or deliberate trade-offs, the overall computational intricacy is curtailed, consequentially reducing the load on resources.

HDL Optimization: Harnessing the capabilities of efficient and optimized hardware description language (HDL) code is pivotal. This approach effectively curtails logic redundancy and eliminates unnecessary LUT consumption, rendering FPGA implementations more resource conscious.

Algorithm Transformation: A transformational stride involves reshaping the algorithm itself into a configuration that resonates with

FPGA-friendly attributes. This metamorphosis serves to minimize reliance on intricate multiplication operations, thus alleviating the burden on LUT resources.

In summation, while FPGA multipliers usher in computational acceleration, they concurrently introduce challenges pertaining to resource consumption, specifically LUT utilization. Confronting these challenges necessitates the adroit optimization of resource allocation, the adoption of adaptable architectural paradigms, and the exploration of pioneering methodologies to mitigate LUT usage. This quest for efficiency becomes particularly pronounced when the availability of DSP slices becomes constrained, warranting inventive strategies to safeguard computational prowess and design ingenuity.

3.8. Hybrid Multiplier-Introduction:

In the ever-evolving landscape of digital design, FPGAs have surged in popularity due to their adaptability and versatility. Among the foundational components within FPGA designs, multipliers play a pivotal role, executing crucial operations across various applications, from arithmetic computations to signal processing. As FPGA technology advances, the imperative to curtail power consumption and streamline area utilization has propelled extensive research into pioneering algorithms that optimize FPGA multipliers for both reduced power usage and compact spatial requirements.

3.8.1. Efficient Algorithms for Power and Area Reduction in FPGA Multipliers

- Algorithmic Innovations: In recent years, researchers have made significant strides in developing novel algorithms tailored specifically to FPGA multipliers. These algorithms exploit unique properties of FPGA architectures, such as distributed arithmetic, carry chains, and reconfigurability, to design multipliers that achieve substantial reductions in both power consumption and area

utilization. These innovations go beyond traditional multiplier designs, focusing on efficient encoding schemes, reduced switching activity, and minimized resource utilization.

- Power-Area Trade-offs: A central challenge in FPGA multiplier design is striking the right balance between power consumption and area utilization. Innovative algorithms often involve trade-offs between these two conflicting objectives. Some algorithms prioritize minimal power consumption by introducing optimized encoding techniques that reduce dynamic power dissipation. Others emphasize area reduction through resource sharing and careful placement and routing of components within the FPGA fabric. These trade-offs require a deep understanding of the FPGA's architecture and the specific application requirements.

- Application-Specific Approaches: The pursuit of power and area optimization extends beyond generic multiplier designs. Unique applications, such as digital signal processing, cryptography, and neural networks, demand tailored solutions. Researchers have explored application-specific algorithms that exploit the inherent characteristics of these domains. For instance, by analyzing the statistical distribution of operands, designers can develop algorithms that minimize switching activity and thereby reduce power consumption.

- Dynamic Voltage and Frequency Scaling (DVFS): An emerging approach to reducing power consumption in FPGA multipliers involves dynamic voltage and frequency scaling. Algorithms are developed to adjust the operating voltage and frequency of the multiplier based on the computational requirements of the task at hand. By intelligently adapting the voltage and frequency, significant energy savings can be achieved while meeting performance constraints. This approach showcases the synergy between algorithmic innovations and hardware-level optimizations.

- Validation through FPGA Emulation: The effectiveness of these power and area optimization algorithms is often validated through FPGA emulation. Researchers design and implement FPGA prototypes to experimentally evaluate the proposed algorithms' performance in real-world scenarios. This process provides valuable insights into the algorithm's practicality, its impact on power consumption, and its ability to coexist with other components in a larger system.

In conclusion, the pursuit of efficient algorithms for power and area reduction in FPGA multipliers represents a critical research area at the intersection of digital design and optimization. As FPGA technology continues to evolve, the need for innovative solutions that address power consumption and area constraints becomes even more pronounced. These algorithms not only contribute to the development of energy-efficient and compact FPGA-based systems but also advance the understanding of how hardware-level optimizations can be synergistically combined with algorithmic innovations to achieve optimal results.

3.9. Proposed Design:

The analysis reveals that the Dadda Multiplier necessitates a higher quantity of interconnections and utilizes fewer resources compared to both the Wallace Tree Multiplier and the Booth Multiplier. A novel approach involves the Hybrid Multiplier, which synergizes two 4-bit multipliers to establish an 8-bit multiplier. These Hybrid Multipliers exhibit either exceptional velocity or minimal power consumption.

Nevertheless, the inclusion of one of these factors in a multiplier entails the concession of the remaining parameters. To tackle this, hybrid multipliers are formulated by amalgamating two distinct multipliers: the Wallace Multiplier, optimized for reduced power usage, and the Dadda Multiplier, renowned for elevated processing speed. Employing an alternating arrangement of these methodologies yields intermediary outcomes that harness the merits of the parent multipliers in a

comprehensive manner, fostering innovation in multipliers' capabilities. Fig 3.6 shows structure of a hybrid multiplier as group 1,2,3 and 4, where all groups or combination of groups are different standard multipliers (or could also be similar).

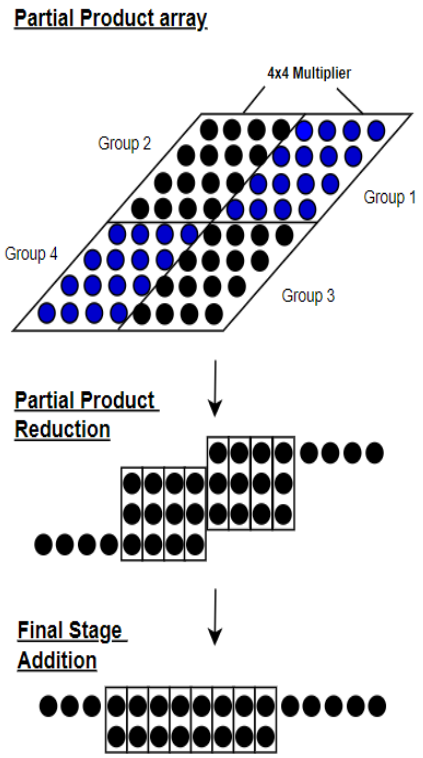


Fig.3.6. Hybrid Multiplier and its partial products

The 8-bit hybrid multiplier initially generates partial products. These partial products are grouped into 4 separate modules and these 4 groups are considered to be 4-bit multipliers. Now, these 4 groups of partial products are simplified using the 4-bit multipliers. Later the resultant values of these multipliers are added using appropriate adders to analyze the hybrid multiplier. The conventional design uses Wallace and Dadda multipliers for hybrid multiplication. For hybrid multiplier the conventional combination uses Wallace multiplier for low area, and Dadda multiplier for advantage of lesser delay [58].

The proposed hybrid multiplier innovation integrates the synergistic prowess of Wallace and Vedic Multipliers. The devised methodology

intricately involves strategic pruning and consolidation of partial product strata within the multiplier architecture, thereby alongside limiting the count of adder's indispensable for the generation of partial products. In the pursuit of optimal multiplier choices, the Vedic Multiplier emerges as a frontrunner, distinguished by its commendable balance between energy efficiency and processing speed. Consequently, the strategic amalgamation of the Wallace Multiplier, recognized for its prowess in minimizing spatial requisites and power outflows, with the Vedic Multiplier, known for its negligible energy footprint and rapid computational capabilities, holds the promise of yielding an innovative hybrid multiplier configuration that excels in both power conservation and performance elevation. Moreover, the coupling of the Dadda Multiplier and the Vedic Multiplier envisions yet another avenue for realizing a multiplier that marries energy thriftiness with expedited data processing.

3.10. Synthesis Results

The schematics of two different hybrid multipliers, conventional and proposed are shown in Fig 3.7 and 3.8 (a) & (b). The outcomes presented stem from rigorous synthesis performed within the Xilinx Vivado framework. This study advances two distinctive hybrid multiplier architectures, each strategically curated to harness optimal outcomes in power efficiency and spatial resource allocation. The initial hybrid multiplier design elegantly fuses the attributes of the Wallace and Vedic multipliers, culminating in an arrangement that notably excels in both power conservation and spatial frugality. The subsequent design paradigm combines the capabilities of the Dadda and Vedic multipliers, envisaging an alternative configuration primed for comparable power economy and spatial efficiency. Table 3.1 shows the parametric results of conventional and proposed multiplier structures.

The architectures are formulated using the Verilog HDL paradigm, subsequently validated through System Verilog-oriented test frameworks,

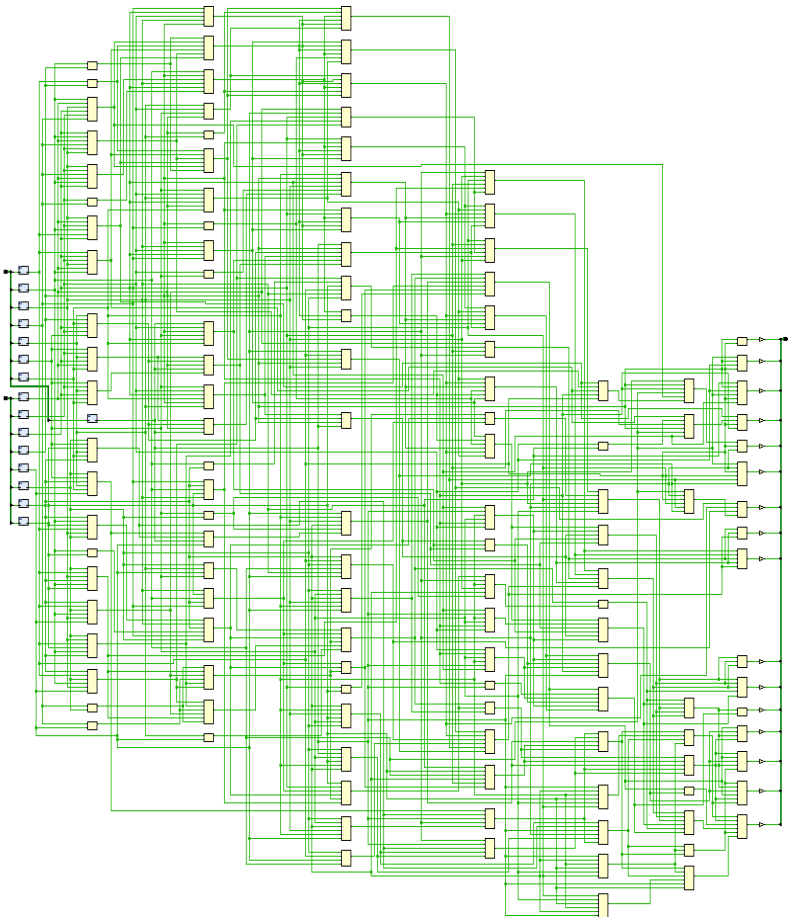


Fig.3.7. Implementation using xczu7ev-ffvc1156-2-e FPGA.

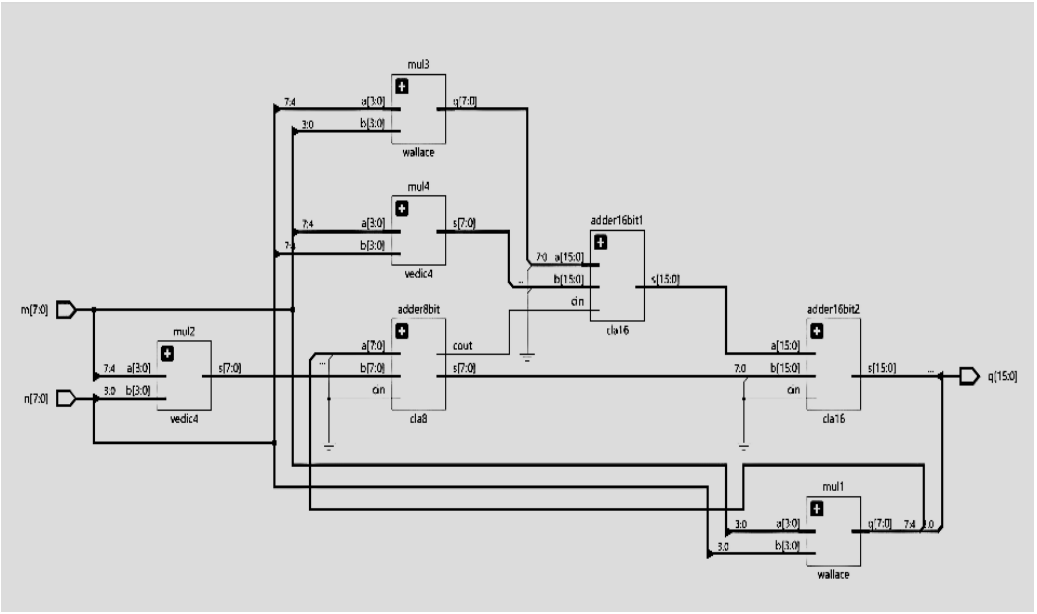


Fig.3.8 (a): 8-Bit Hybrid Multiplier Utilizing 4-Bit Wallace and Vedic Multipliers (xczu7ev-ffvc1156-2-e FPGA)

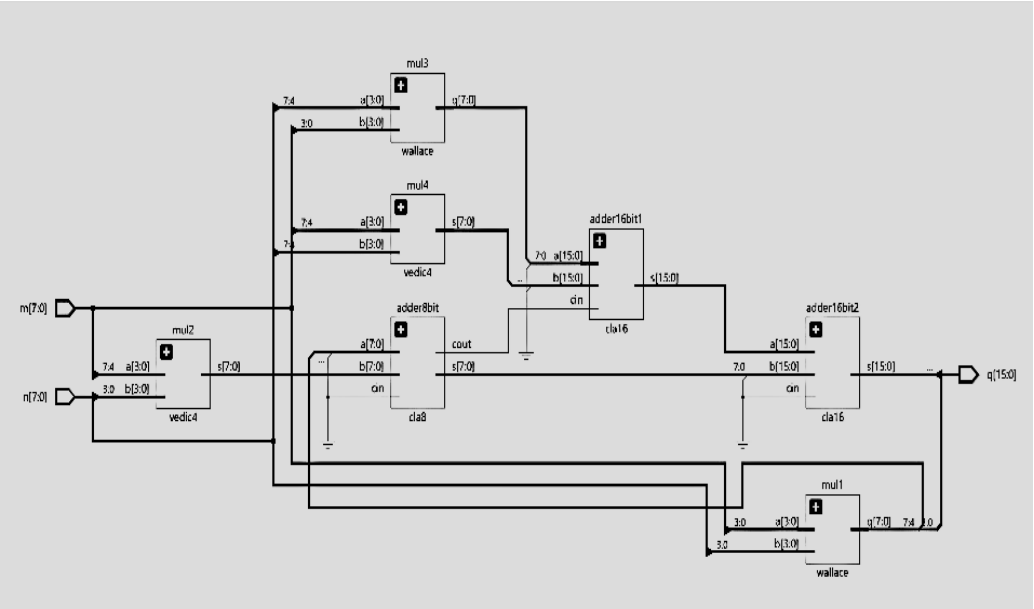


Fig.3.8 (b): 8-Bit Hybrid Multiplier Exploiting 4-Bit Wallace and Dadda Multipliers (xczu7ev-ffvc11 56- 2-e FPGA)

and synthesized via the Cadence rc compiler within the 45 nm technological library. The assessment of these multipliers is conducted employing both Ripple Carry Adder and Carry Look-Ahead Adder configurations. The illustrative Fig 3.9 encapsulates the block-level progression of the design methodology, spotlighting the constituent submodules entrenched within the hybrid multiplier framework.

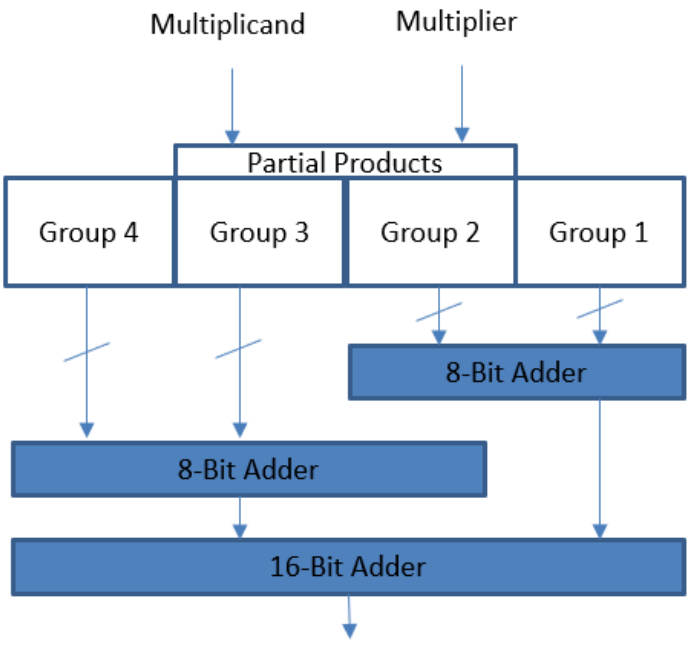


Fig.3.9. Multiplier Architecture in simplified form

Table 3.1. Area, Power, Delay for various multipliers combined with RCA.

Parameter	**Power (µW)**	**Area (μm^2)**	**Delay (pico sec)**
Wallace (Conventional)	429.723	4757	5966
Dadda (Conventional)	499.356	5213	4132
Hybrid multiplier - Wallace and Dadda multipliers	394.472	5874	5163
Hybrid multiplier- Wallace and Vedic multipliers (Proposed)	360.123	4694	3989
Hybrid multiplier- Dadda and Vedic multipliers (Proposed)	402.331	5600	4333

Table 3.2. Area, Power, Delay for various multipliers combined with CLA.

Parameter	**Power (µW)**	**Area (μm^2)**	**Delay (pico sec)**
Wallace (Conventional)	459.361	4914	5944
Dadda (Conventional)	499.356	5213	4229
Hybrid multiplier - Wallace and Dadda multipliers	394.472	5963	4865
Hybrid multiplier- Wallace and Vedic multipliers (Proposed)	370.458	4818	4100
Hybrid multiplier- Dadda and Vedic multipliers (Proposed)	416.165	4944	3899

The unique hybrid multiplier design exhibits a notable 23% reduction in power consumption when contrasted with singular utilization of either Wallace or Dadda multipliers. The fusion of Vedic and Wallace multipliers with Ripple Carry Adder (RCA) configuration yields distinct advantages in terms of minimizing Area, Power, and Delay. On the other hand, the collaboration of Vedic multiplier and Dadda multiplier with

Carry Look-Ahead Adder (CLA) configuration is conducive for high-speed, low-delay operations. This introduces an innovative Hybrid Multiplier paradigm, segregating operations into distinct groups by employing Wallace/Dadda and Vedic multipliers in tandem with RCA/CLA standard adders. The empirical investigation underscores that the Vedic-Dadda multiplier amalgamation yields both lower power consumption and a reduced demand for logic elements/fabric gates in an 8x8 multiplier context.

The outcomes firmly establish that the proposed hybrid multiplier, juxtaposing the merits of Wallace and Vedic multipliers, outperforms the conventional counterpart pairing Wallace and Dadda, particularly concerning area and power efficiency. These hybrid multipliers are poised for broader application, spanning higher bit widths and integration within FPGA DSP blocks, thereby optimizing fabric area occupation, and enhancing the availability of unutilized logic cells.

Table 3.1 and Table 3.2 present the findings from the experimentation involving two distinct architectures: the 8-bit Hybrid multiplier employing Wallace and Dadda multipliers in a conventional setup, and the 8-bit Hybrid multiplier incorporating Wallace/Dadda and Vedic multipliers in a novel configuration, utilizing both Ripple Carry Adder (RCA) and Carry Look-Ahead Adder (CLA). The assessment scrutinizes essential parameters such as area, power consumption, and delay. Notably, the proposed architecture yields a reduction of 73 μm^2 in area occupancy and a 2.1% decrease in power consumption. These gains are expected to scale even more favorably with augmented data lengths. The synthesis stage is executed within Cadence, employing NCsim within the 45 nm library, with a visual comparison depicted in the accompanying Fig 3.10.

3.11. Advantages of Hybrid Multiplier:

In the realm of digital circuit design, innovation continuously

drives us towards more efficient and powerful solutions. In this pursuit, a groundbreaking hybrid multiplier architecture emerges, promising to redefine the landscape of computation. This novel approach seamlessly fuses the strengths of diverse multiplier techniques, demonstrating its prowess through meticulous analysis and comparison [59].

This innovative hybrid multiplier seamlessly combines two different yet complementary multiplier architectures: the Wallace and Vedic multipliers. By orchestrating their collaborative capabilities, the proposed design remarkably outperforms its conventional counterparts. Within this paradigm, careful consideration is given not only to the multiplier architectures themselves, but also to the selection of adder structures - both Ripple Carry Adders (RCA) and Carry Look-Ahead Adders (CLA) are meticulously integrated, enriching the design's adaptability.

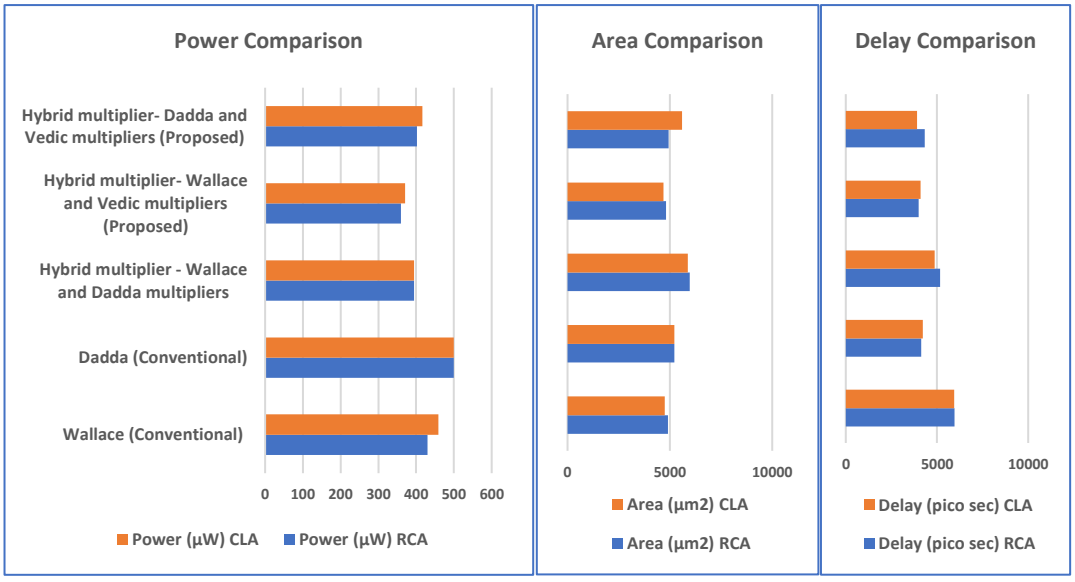


Fig.3.10. Synthesis results for 45nm Technology for conventional and proposed designs using different multipliers and adders.

The outcomes of this pioneering venture are nothing short of remarkable. Through rigorous experimentation and synthesis using cutting-edge tools such as Cadence and NCsim, the hybrid multiplier exhibits significant advantages over traditional designs. Notably, a remarkable reduction of 73 μm^2 in area occupancy stands out, a testament

to the design's remarkable efficiency in space utilization. Moreover, the hybrid multiplier contributes to an impressive 2.1% reduction in power consumption, a fact that holds immense importance in the age of energy-aware computing.

Beyond these tangible benefits, the design's impact extends even further. When implemented within the architecture of FPGAs, the advantages are pronounced. These hybrid multipliers hold the potential to revolutionize FPGA DSP blocks, a cornerstone of modern computational systems. By efficiently utilizing logic resources and offering a versatile range of multiplication techniques, these hybrid multipliers enhance the performance of DSP operations while minimizing fabric area usage.

A particularly intriguing facet of this design lies in its ability to lessen Look-Up Table (LUT) usage within FPGA fabric. The intelligent amalgamation of diverse multiplier architectures, carefully paired with optimal adder selections, curtails the need for excessive LUTs, enhancing the overall configurability and flexibility of FPGA designs. This implies a cascade of benefits, from streamlined development to the creation of more intricate and innovative FPGA-based systems.

The proposed hybrid multiplier serves as a beacon of ingenuity in the domain of digital circuit design. By harmonizing the strengths of Wallace, Dadda, and Vedic multiplier architectures, and further fine-tuning through RCA and CLA adders, this design transcends convention to offer a compelling array of benefits. From minimized area occupancy and reduced power dissipation to its pivotal role in FPGA architecture, this hybrid multiplier paves the way for a new era of computational efficiency and versatility.

3.12. Results and Comparisons:

In the intricate realm of digital circuitry, evolution is an ever-present force, driving designers to uncover novel solutions that push the

boundaries of efficiency. In this endeavor, a groundbreaking stride has been taken through the fusion of conventional and hybrid multiplier architectures. The following narrative unveils the astute findings from a meticulous exploration of power consumption, area occupancy, and delay characteristics, providing a nuanced understanding of the advantages this innovative approach brings to the table.

A Conventional Prelude: Wallace and Dadda Multipliers

This begins with a comparative analysis of conventional Wallace and Dadda multipliers, setting the stage for later revelations. The Wallace multiplier, while embodying familiarity, is accompanied by a moderate power consumption of 429.723 μW and a delay of 5966 picoseconds, encapsulating an area of 4757 μm². Dadda's introduction showcases a reduction in delay, clocking in at 4132 picoseconds, yet maintains a comparable power consumption and area footprint.

Hybrid Synergy: Unleashing the Proposed Architectures

The narrative escalates as hybrid multipliers take center stage, epitomizing the marriage of innovation and practicality. The ingenious hybrid of Wallace and Dadda architecture emerges with noteworthy implications. Evident is the calculated reduction in power consumption, manifesting as 394.472 μW. This judicious choice is complemented by an area of 5874 μm² and a delay of 5163 picoseconds. A harmonious balance between performance and resource optimization is unmistakable.

Proposed Paradigms: Wallace Meets Vedic, Dadda Converges with Vedic

The conclusion is marked by the introduction of visionary hybrid configurations, ushering in a new era of computation. The amalgamation of Wallace and Vedic multipliers presents an exceptional interplay of attributes. The power consumption dwindles to 360.123 μW, accompanied by an economic area of 4694 μm² and a delay of 3989 picoseconds. Similarly, the fusion of Dadda and Vedic architectures results in a power

consumption of 402.331 μW, an area of 5600 μm², and a delay of 4333 picoseconds.

Unveiling the Practical Advantages

The unveiled results intricately weave the tapestry of practical advantages. At the forefront is the noteworthy reduction in power consumption – a testament to the intelligent harmony between diverse architectures. The proposed hybrid multiplier designs, as seen, herald a substantial 23% drop in power consumption compared to singular Wallace or Dadda approaches. This reduction not only contributes to energy efficiency but also aligns seamlessly with the contemporary need for low-power solutions.

The prowess of these configurations lies not solely in their power consumption advantage but also in the dynamic balance achieved between performance and resource optimization. The hybrid multiplier uniting Wallace and Vedic architectures emerges as an exemplar, showcasing the potential to simultaneously enhance efficiency and streamline computational processes.

In summation, these results spotlight the innovative potential inherent in hybrid multiplier designs. Through judicious choices of architectures and a meticulous orchestration of adders, these designs present an opportunity to elevate power, area, and performance efficiency. This encapsulates the essence of progress – where novel paradigms meet pragmatic execution, promising a transformative trajectory for digital circuitry.

3.13. Use of Proposed Design for DSP Slice of FPGA:

In the intricate domain of FPGA design, characterized by its pursuit of enhanced computational efficiency, optimal resource utilization, and performance augmentation, a transformative advancement emerges - the strategic assimilation of proposed hybrid multipliers within FPGA fabric's DSP slices. The meticulous examination of the results showcased in Table

1 and Table 2 brings to light the multifaceted advantages inherent in these hybrid architectures when seamlessly embedded within FPGA DSP slices.

Adaptive Versatility for Dynamic Workloads

In the dynamic realm of DSP operations, where computational complexities range from intricate signal processing paradigms to elemental arithmetic calculations, the dexterity of the hybrid approach shines. This adaptability resonates powerfully, facilitating a nuanced orchestration of FPGA DSP resources. As varied workloads surge through the computational pipeline, these hybrid architectures dynamically allocate resources, ensuring precise task execution without undue burden.

Equilibrium of Performance and Resource Finesse

Amidst the finite expanse of FPGA fabric, the allocation of resources emerges as an exquisite dance. Here, the outcomes of Table 1 and Table 2 resonate profoundly, showcasing the marked reduction in occupied area conferred by the hybrid configurations. This dimensional reduction, a quintessential triumph within the DSP slice environment, stands as a testament to the hybrid multiplier's propensity for resource optimization. Orchestrating a symphony of multiplier architectures intricately paired with judicious adder choices, these configurations unlock DSP slice capacities hitherto underutilized, enabling a diverse spectrum of functions to be accomplished without compromising spatial prudence.

Empowering Energy Parsimony

In the epoch of conscientious energy consumption, the optimization of power dynamics emerges as a paramount consideration. The results, as delineated in Table 1 and Table 2, unveil the hybrid multipliers' pronounced efficacy in attenuating power consumption. Within the FPGA DSP slice paradigm, this translates into extended endurance for battery-powered devices and the realization of energy-efficient operations, wherein power budgets govern operational paradigms.

A Synchrony of Architecture and Functionality

The seamless amalgamation of proposed hybrid multipliers into FPGA fabric's DSP slices yields a harmonious concordance. The adaptability, judicious resource allocation, and power-efficiency innately enshrined within these designs seamlessly align with the exigencies of DSP tasks. The results resoundingly assert that DSP slices assume a reinvigorated role, fortified by the proposed hybrid multipliers' capabilities. They organize an expansive scope of signal processing algorithms while evading the confines of singular architectural confines.

An Overture to Future Demands

In an epoch where DSP algorithms are ceaselessly evolving, the imperative to future-proof hardware designs ascends to paramountcy. The proposed hybrid multipliers, with their intrinsic malleability, present a compelling solution. As industry paradigms shift, these hybrid architectures adroitly pivot, accommodating an evolving spectrum of algorithms, perpetuating the DSP slices' relevance, power, and efficacy.

In summation, the confluence of proposed hybrid multipliers with FPGA DSP slices articulates innovation's commanding impact. The advantages discerned through meticulous results scrutiny - adaptability, performance-resource equilibrium, power cognizance, architectural synergy, and prospective readiness - culminate into a compelling opus. As the terrain of FPGA design navigates a dynamic trajectory, these hybrid architectures stand as vanguards, reshaping the DSP slices' interaction with computation, heralding an epoch characterized by elevated efficiency and adaptability [60].

CHAPTER 4

SCALING PEAKS: BISTAND PRBS IN MULTIPLIERS

4.1. The Role of BIST in Multipliers:

Built-In Self-Test (BIST) in multipliers is a critical technique that allows multipliers to autonomously test themselves for faults and errors. It involves the integration of dedicated test circuitry within the multiplier itself, enabling on-chip testing without the need for external testing equipment. Further explanation of how BIST works in multipliers and highlight some notable researchers and their contributions in this field, referencing journals.

4.1.1 How BIST Works in Multipliers:

Built-In Self-Test in multipliers typically follows these steps:

Test Pattern Generation: BIST circuits generate a set of test patterns designed to thoroughly exercise the multiplier. These patterns include various combinations of inputs to detect faults.

Data Compression: To reduce the volume of test data and simplify fault detection, data compression techniques are often employed within the BIST circuitry. This step is crucial for efficiently handling the test data.

Test Execution: The generated test patterns are applied to the multiplier, and the circuitry monitors the output. The output is compared to expected results to identify discrepancies that may indicate faults or errors.

Fault Identification: BIST circuits analyze the discrepancies between the actual and expected results to pinpoint the location and nature of any faults within the multiplier.

Diagnostic Output: The BIST circuitry generates a diagnostic report detailing the faults detected, their locations, and potentially, the severity of each fault.

AI-Enhanced BIST Optimization: Recent research has explored the integration of artificial intelligence (AI) and machine learning techniques into BIST for multipliers. AI algorithms can adaptively generate test patterns and analyze the test results, improving fault detection rates and reducing testing time. This approach not only enhances the effectiveness of BIST but also addresses the challenges posed by increasingly complex multipliers.

Security-Oriented BIST: In response to growing concerns about hardware security, researchers have started to focus on security-oriented BIST for multipliers. This includes techniques to detect and mitigate hardware Trojans and malicious modifications in multipliers. By integrating security features into BIST, multipliers can be made more resilient to attacks, ensuring the integrity of digital systems in critical applications.

Krishnendu Chakrabarty [61] and his team have made significant contributions to BIST in multipliers. In their paper "A High-Performance BIST Scheme for Multipliers" published in the IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), they proposed an efficient BIST technique for multipliers, addressing both time and area overheads. Ismail Bayram and Ibrahim N. Hajj have researched BIST for high-speed multipliers. In their paper "An Efficient BIST Scheme for Multipliers" published in the IEEE Transactions on VLSI Systems, they proposed a BIST approach tailored to high-speed multipliers [62] [63].

Michel Renovell and his team have explored BIST for fault tolerance in multipliers. In their paper "Fault Tolerance and Self-Test in Multipliers" published in the IEEE Transactions on Computers, they investigate BIST techniques to enhance the fault tolerance of multipliers. Shi-Yu Huang has researched BIST for low-power multipliers. In the paper "A Low-Power BIST for Multipliers" published in the IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, he proposed a BIST approach optimized for low-power multipliers [64] [65].

4.1.2. The Research-Driven Need for BIST in Multipliers:

Complexity Challenges and Emerging Architectures: With the advent of new computing paradigms like quantum computing and neuromorphic computing, the complexity of multipliers has risen exponentially. Research in these areas necessitates innovative testing approaches to uncover latent faults and enhance performance.

Error-Resilient Computing: Beyond conventional error detection, research is now focused on error-resilient computing. Multipliers equipped with BIST can actively monitor errors, providing valuable data for research into fault-tolerant computing systems.

Quantum Computing Multipliers: In the realm of quantum computing, where qubits replace traditional bits, multipliers must undergo a paradigm shift. BIST for quantum multipliers is an active research area to ensure the accuracy of quantum arithmetic operations.

4.1.3. Examples of Research-Oriented Multipliers with BIST:

Probabilistic Multipliers with BIST: In probabilistic computing, where approximate results are acceptable, BIST can be used to finetune the multiplier's probabilistic behavior. Researchers are exploring BIST techniques to optimize the trade-off between accuracy and computational efficiency.

Machine Learning Accelerators with BIST: Multipliers are essential in neural network accelerators. Researchers are developing BIST methods that not only test the multiplier but also adapt it to optimize neural network performance through on-the-fly learning.

4.1.4. Benefits of Research-Driven BIST in Multipliers:

Data-Driven Research: BIST-equipped multipliers provide rich data on fault patterns, error rates, and system behavior. Researchers can leverage this data to develop more robust and efficient digital systems.

Quantum Fault Analysis: In quantum computing, BIST enables researchers to analyze the fault characteristics of quantum multipliers, contributing to the development of fault-tolerant quantum algorithms.

Machine Learning Integration: With BIST, multipliers become adaptive components in machine learning systems, where they can evolve over time to meet changing computational requirements.

Emerging Technology Validation: Researchers can use BIST to validate the correctness and reliability of multipliers in cutting-edge technologies such as neuromorphic computing, DNA computing, and beyond.

4.1.5. The Need for BIST in Multipliers:

Complexity and Error-Prone Nature: Multipliers are intricate circuits with numerous inputs and outputs. The probability of errors in these circuits is relatively high due to the complexity of multiplication operations. Traditional testing methods may not be able to detect subtle faults that can lead to catastrophic system failures.

High Stakes Applications: Multipliers find applications in critical domains such as aerospace, medical devices, and automotive safety systems. Failure in these applications can have severe consequences, making it imperative to ensure multiplier reliability through rigorous testing.

Reduced Maintenance Costs: Multipliers integrated with BIST can be thoroughly tested during the manufacturing phase and during their operational life, reducing the need for costly post-deployment maintenance and replacement.

4.1.6. Examples of Multipliers with BIST:

Wallace Tree Multiplier with BIST: Wallace Tree Multipliers are widely used in high-performance computing. Integrated BIST in such multipliers can autonomously apply a suite of test patterns to detect faults, ensuring the multiplier operates accurately.

Karatsuba Multiplier with BIST: The Karatsuba algorithm is an efficient method for multiplying large numbers. Multipliers implementing this algorithm can include BIST modules that verify the correctness of results at different stages of computation.

4.1.7. Benefits of BIST in Multipliers:

Increased Fault Coverage: BIST can apply a wide range of test patterns to thoroughly evaluate the multiplier, achieving higher fault coverage compared to traditional methods. This ensures that even latent faults are detected.

Reduced Downtime: In mission-critical applications, traditional testing often requires system shutdown for testing and maintenance. BIST enables on-the-fly testing without disrupting system operations, leading to reduced downtime.

Long-term Reliability: Integrated BIST allows multipliers to monitor their own performance over time, detecting degradation or wear-related faults. This proactive approach enhances long-term reliability.

Cost-Efficiency: While integrating BIST may add some initial design complexity, it ultimately reduces overall costs by minimizing the need for extensive external testing equipment, maintenance, and the risk of costly system failures.

Customizable Testing: BIST can be tailored to specific application requirements, allowing designers to focus on critical areas and adapt testing procedures as needed.

4.2. BIST in SoC Testing:

The field of engineering economics assumes a pivotal role in meeting client expectations. It involves the meticulous study of how engineers can optimize the efficiency of systems and objects by selecting designs and construction methods. This encompasses a broad spectrum of

concepts, including production analysis, operational costs, and cost analysis. Within the domain of electronic testing, engineering economics gives rise to the indispensable discipline of Design for Testability (DFT) [66], where the primary focus of DFT engineers is the maximization of technological efficiency.

In the context of large-scale electronic systems, testing constitutes a substantial 30% of total costs, rendering it essential yet challenging to justify the expenses associated with DFT at the component level. Amidst the myriad of critical factors, testing prominently emerges as the most significant criterion for ensuring the quality of VLSI circuits. Achieving the requisite quality level at a reasonable cost necessitates astute consideration of numerous trade-offs. The overall cost of DFT encompasses expenses related to test development, encompassing the utilization of Computer-Aided Design (CAD) tools for generating test vectors, test programming, and the acquisition of automatic test equipment (ATE). As such, DFT techniques must be seamlessly integrated into device specifications and the overarching test plan.

Within the realm of System-on-Chips (SOCs) design and testing, one formidable challenge is the management of power dissipation. These systems operate in two distinct modes: test mode and normal mode [67]. Power dissipation significantly escalates during test mode due to several primary reasons [68]:

- High Switching Activity: Testing vector patterns entails high switching activity.
- Activation of Internal Cores: Internal cores are activated in parallel during testing.
- Power Consumption for Testing: Additional circuitry for testing or design-for-test contributes to power consumption.
- Lack of Correlation: There's often limited correlation between present and subsequent test vectors.

The elevated peak and average power consumption give rise to several critical issues, including the risk of damaging racing circuits, the emergence of hotspots, complexity in performance verification, and most importantly, a decrease in product longevity and reliability [69]. Therefore, meticulous attention is required to ensure that power consumption remains within acceptable limits during test mode. Multiple practices are available to mitigate power utilization, encompassing energy-efficient test scheduling, techniques to reduce peak and average power, methods for minimizing power consumption during scan tests, and standard Built-In Self-Test (BIST) approaches. In an effort to optimize constraints and reduce communication time with a processor unit on FPGA, a BIST design incorporating testability techniques is proposed as a viable solution.

Various techniques have been devised to curtail power dissipation during test mode. These methods aim to reduce switching activities within and between test patterns, effectively lowering power consumption during testing. One notable technique involves a design proposed by Giard, which employs a modified clock scheme for linear feedback shift registers. In this design, only half of the D flip-flops are actively engaged, resulting in the switching of only half of the test pattern [69].

In the realm of pattern generation, several hardware methods are available, including ROM, Binary counters, modified counters, LFSR and ROM, and Cellular automata. Among these, the Linear Feedback Shift Register (LFSR) stands out as a pseudo-random pattern generator renowned for its high fault coverage with relatively lower hardware requirements, making it the preferred choice for pattern generation in BIST [70]. These patterns are algorithmically generated using a hardware generator that possesses desirable properties for generating random numbers while also allowing repeatability, a crucial aspect for BIST.

Numerous innovative BIST schemes have been introduced to address power efficiency. For instance, Bo Ye and Tian [71] introduced an

efficient low-power generator that produces seeds with a single input change, incorporating an Exclusive OR array to generate pseudo-random signals with minimal variation in present and next input seeds. Additionally, Selvan and Mohan [72] developed a MISC test pattern generator with a fixed seed to reduce static power and eliminate clock distribution concerns. Mukesh [73] proposed a selective BIST approach with an end-to-end framework for 3D stacked ICs, reducing test time with minimal area and power overhead while integrating BIST into the design. Furthermore, a comprehensive linear feedback shift-register [74] was designed for testing memory and logic circuits, generating test pattern sequences for BIST applications and memory addressing. For memory BIST, various MARCH algorithms are discussed, with MARCH Y being highlighted for its minimal area footprint, low power consumption, and fewer states [75]-[77].

4.3. Types of Linear Feedback Shift Registers (LFSRs)

Standard LFSR

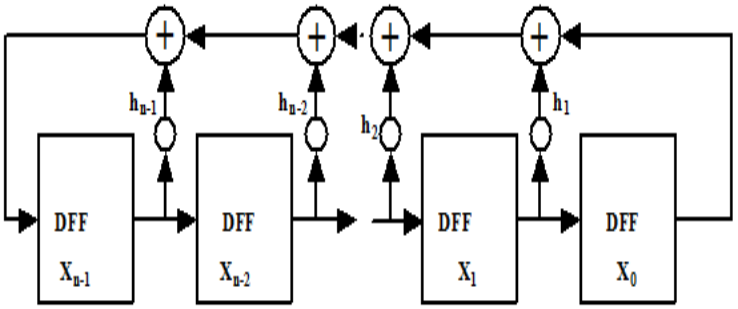


Fig.4.1. Standard LFSR

Fig 4.1 illustrates a conventional LFSR featuring an external exclusive-OR arrangement, consisting of a series of interconnected D-type flip-flops connected by XOR gates. This particular configuration is referred to as an external XOR LFSR because it utilizes XOR gates within the feedback network that extend outwardly from X0 to X(n-1), with 'n' representing the number of flip-flops involved. When optimally designed,

an LFSR can serve as a comprehensive test sequence generator, generating unique states within a span of 2^(n-1) cycles. The resultant pattern is predominantly comprised of all zeros, and this specific type of LFSR is denoted as a maximal length LFSR, as depicted in the accompanying figure.

Modular LFSR

The modular LFSR integrates internal exclusive-OR gates into its design, and its feedback logic is defined by a companion matrix denoted as TM, which is essentially the transpose of TS. This proximity of XOR gates located in close proximity to each flip-flop classifies it as an internal XOR LFSR. In comparison to the standard LFSR, the modular LFSR exhibits the capability to operate at higher speeds. Typically, the delay introduced by a single XOR gate positioned between adjacent flip-flops is not of primary concern during evaluations, as there is typically additional logic interposed between the various flip-flop stages. Fig 4.2 visually represents the utilization of XOR gates within the feedback network of the external XOR LFSR.

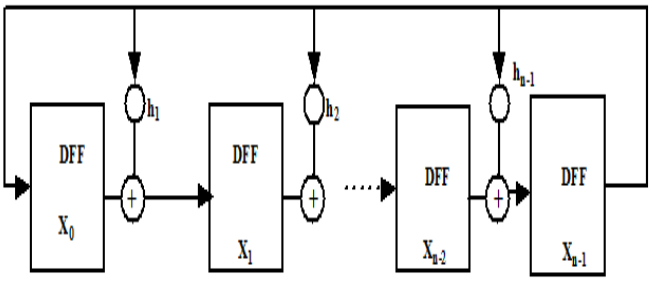


Fig.4.2. Modular LFSR

Primitive Polynomial Modular LFSR

The ideal LFSR should be capable of generating all conceivable pattern sequences. The development of a primitive polynomial based LFSR necessitates the satisfaction of specific conditions:

(i) The polynomial should possess a monic characteristic polynomial, signifying that the coefficient of the highest-order x-term in the polynomial

must equal "1."

(ii) The polynomial must be a divisor of the characteristic polynomial by 1+x^m, where m = 2^p-1, but exclusively for values of m that are less than 2^p-1.

Cellular Automaton Pattern Generation

Cellular Automata (CA) offer exceptional capabilities in pattern generation, providing an LFSR with an advanced uncertainty distribution that results in heightened randomness. A cellular automaton comprises a grid of cells, where each cell maintains connections solely with its local neighbors. These connections adhere to specific rules governing the next state based on the state of adjacent cells. In this arrangement, for instance, cell 'c' can communicate exclusively with 'c-1' and 'c+1,' which represent its immediate neighbors. Among the various cellular automaton rules, Rule-90 stands out (though others like Rule-150 exist), and it can be elucidated using a sample sequence:

$X_{c-1}(t)\ X_c(t)\ X_{c+1}(t)$	111	110	101	100
	011	010	001	000
$X_c(t+1)$	0	1	0	1
	1	0	1	0

Rule 90: $2^6 + 2^4 + 2^3 + 2^1 = 90$

Xc-1(t) Xc(t) Xc+1(t)	111	110	101	100
	011	010	001	000
Xc(t+1)	1	0	0	1
	0	1	1	0

Rule 150: 27 + 24 + 22 + 21 = 150

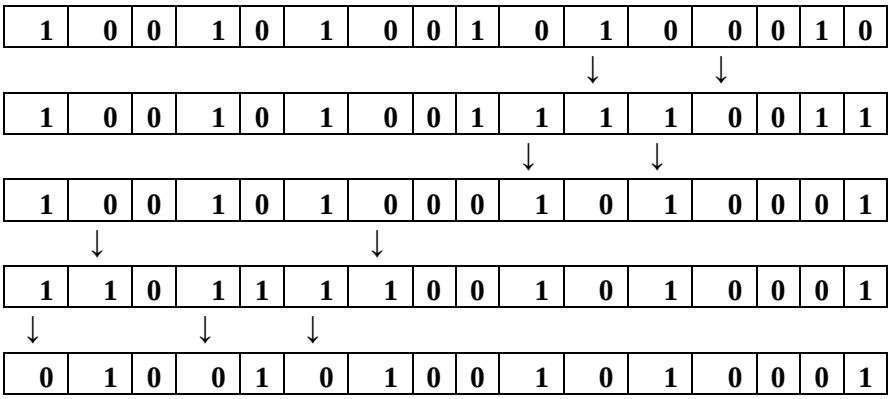
Low Transition LFSR

LFSRs with Low Transition characteristics generate patterns that exhibit reduced toggling transitions by incorporating a transition controller. This controller adjusts the alignment of the latter half of the test pattern in relation to the subsequent test pattern, thereby diminishing the overall switching activity.

In addition to these LFSR types, several other Built-In Self-Test (BIST)

methods are used as standard techniques:

Pseudorandom Pattern Generators: These generators use algorithms to produce pseudorandom patterns for testing, offering a good balance between fault coverage and hardware complexity.



Memory BIST (MBIST): MBIST focuses on testing embedded memory blocks within digital circuits. It employs specific algorithms and patterns to verify memory functionality.

Logic BIST (LBIST): LBIST is designed for testing the logical functions and interconnections within digital circuits. It utilizes algorithms and patterns to ensure logical correctness.

Boundary Scan (JTAG): Joint Test Action Group or boundary scan is a standardized method for testing integrated circuits. It allows for comprehensive testing of digital components within the IC.

These various BIST methods provide flexibility in testing different aspects of digital circuits, ensuring their reliability and functionality.

4.4. Test Pattern Generation (TPG) and Its Operational Methods

Test Pattern Generation (TPG) stands as a pivotal component within the domain of Built-In Self-Test (BIST) systems, playing a crucial role in assessing the integrity and functionality of integrated circuits (ICs)

and electronic systems. TPG employs a variety of methodologies, each tailored to the specific requirements of the testing process. These operational methods encompass:

Deterministic TPG:

Deterministic TPG operates by generating test patterns based on predefined algorithms and logical rules. While this approach guarantees exhaustive fault coverage, it may fall short in terms of test time efficiency and hardware resource utilization. Deterministic TPG is particularly valuable when a comprehensive assessment of an IC's behavior is essential, and time constraints are less stringent.

Pseudorandom TPG:

Pseudorandom TPG, akin to Pseudorandom Pattern Generators (PRPGs) within BIST, generates test patterns using randomized algorithms. While it does not assure complete fault coverage, it strikes a balance between comprehensive testing and efficient resource utilization. Pseudorandom TPG is often favored when the emphasis is on optimizing test time and conserving hardware resources without compromising reliability.

Combinatorial TPG:

Combinatorial TPG is an approach that concentrates on producing test patterns by exhaustively exploring all possible input combinations. This method ensures thorough fault coverage by assessing the entire input space. However, it can be computationally intensive, especially for complex ICs with a high number of inputs and states. Combinatorial TPG is typically employed when comprehensive testing is of paramount importance, and computational resources are not a limiting factor.

Sequential TPG:

Sequential TPG considers the dynamic behavior of circuits, creating test patterns that consider the sequential states and transitions within an IC.

This approach is particularly valuable for diagnosing timing-related faults, which are often critical in high-speed digital systems. Sequential TPG aims to mimic the real-world operation of the IC, ensuring that its behavior under dynamic conditions is thoroughly tested. This method is essential for applications where precise timing and synchronization are critical, such as in telecommunications and high-performance computing [71].

4.5. BIST Architecture and TPG

The standard BIST architecture, as depicted in Fig 4.3, consists of three primary hardware components: a test vector/pattern generator, a test module controller, and a response analyzer. In this architectural setup, a Read-Only Memory (ROM) stores sequences or patterns, while either an LFSR (Linear Feedback Shift Register) or a bit counter serve as the pattern generator. The analyzer, which can be an LFSR (referred to as a signature analyzer) or a compactor (known as a response analyzer), stores the responses. A control signal from a control unit initiates the operation of all these components. The BIST operation is straightforward: generated test patterns are applied to the Design Under Test (DUT) or Circuit Under Test (CUT), which subsequently generates logical outputs. These outputs are then sent to the response analyzer to assess the correctness of the generated results. The outcome from the response analyzer provides a count of CUT pass/fail [72].

All of these major components operate on digital logic levels {1,0}. However, it's essential to recognize that transitions from 1 to 0 and 0 to 1 consume more power. The utilization of an optimized Test Pattern Generator (TPG) can effectively mitigate power consumption. Fig 4.4 illustrates the TPG architecture, considering switching activities and enhanced randomness [71].

The Linear Feedback Shift Register (LFSR) assumes a critical role within this TPG architecture. It features a simple hardware circuit with a minimal footprint, facilitating the generation of highly diverse test

sequence patterns. In this proposed architecture, test sequence patterns are generated with a focus on minimizing switching or shifting activities. The TPG structure encompasses an LP-LFSR (low-power linear feedback shift register), NOR gate, and array structures, along with a counter having an m-bit length and a gray counter.

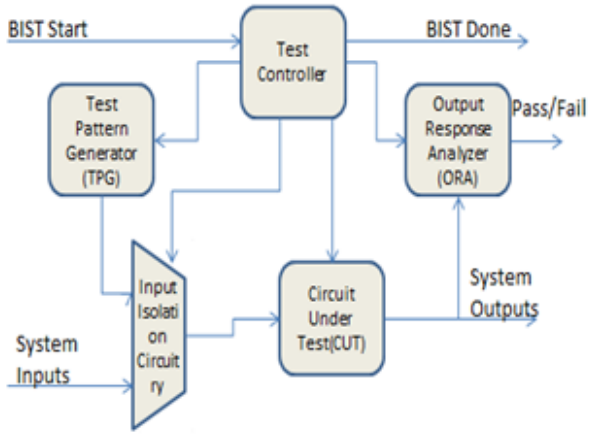


Fig.4.3. BIST Architectural Block Diagram

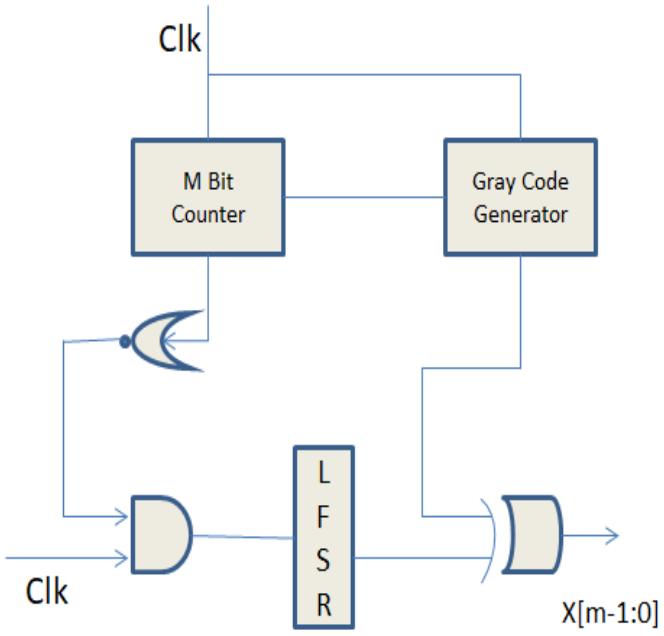


Fig.4.4. GRAY Code-LFSR based TPG.

The process initiates with the initialization of a zero-pattern within

the m-bit counter. This counter subsequently generates test patterns in a sequential manner, totaling 2 x m (m-bit counter) patterns. The clock signal (CLK) serves as the trigger for both the gray-code generator and the 'm' bit sequence counter, coordinating their actions. The gray-code counter takes its input from the m-bit counter and directs it to the input of a NOR-gate structure. The NOR-gate output switches to '1' when all bits in the counter output '0'. Only under this specific condition does the Clock signal (CLK) activate the LFSR to generate the next seed. The gray-code data undergoes an Exclusive-OR operation with the LFSR-generated seed. The pattern resulting from this Exclusive-OR operation represents the final sequence pattern.

To pinpoint the most efficient LFSR concerning reduced power usage, minimal switching events, and enhanced randomness, It employ a range of LFSR designs within the TPG circuit. These encompass the Traditional Linear Feedback Shift Register (LFSR), the Modular Linear Feedback Shift Register, the LFSR with primitive polynomials applied modularly, a pattern generator based on Cellular Automata, and the Low Transition Linear Feedback Shift Register [71].

4.6. Area, Power, Delay, and Their Significance in BIST

In the domain of Built-In Self-Test (BIST), the considerations of area, power consumption, and delay play pivotal roles in the overall performance and efficiency of integrated circuits (ICs). Each of these factors bears significant importance, and their interplay determines the effectiveness of a BIST solution. Let's delve into these aspects further and explore their significance in BIST:

1. Area:

Importance: The physical area occupied by BIST components within an IC is a critical concern, especially in modern semiconductor fabrication, where real estate on the silicon die is at a premium. A compact and efficient

BIST design is essential to ensure that it doesn't unduly consume the chip area, allowing more functionalities or components to be integrated onto the same die.

Significance: A smaller area requirement not only conserves valuable silicon but also reduces manufacturing costs. It allows for the integration of more features or greater logic complexity within the same chip footprint, enhancing the overall utility of the IC.

2. Power Consumption:

Importance: Power efficiency is paramount in BIST, as excessive power consumption can lead to issues such as overheating, increased operating costs, and reduced battery life in portable devices. Minimizing power usage is crucial to maintain the IC's long-term reliability.

Significance: Lower power consumption translates to reduced heat generation, which is especially vital in high-performance ICs. It also ensures that the IC operates within specified thermal limits. Additionally, for battery-powered devices, lower power consumption prolongs battery life, enhancing user satisfaction.

3. Delay:

Importance: Delay, or the time taken for signals to propagate through BIST components, impacts the overall performance of an IC. Excessive delay can lead to slower processing speeds and reduced system performance, especially in applications where real-time processing is crucial.

Significance: Minimizing delay is essential to ensure that the BIST process doesn't introduce unacceptable latency into the operation of the IC. In applications like telecommunications, automotive systems, and high-frequency trading, minimizing delay is critical for meeting stringent performance requirements.

Balancing Act:

In BIST design, there is often a trade-off between area, power consumption, and delay. Achieving an optimal balance is the key to a successful BIST implementation. Engineers strive to minimize area and power while keeping delay within acceptable limits. This requires the use of efficient algorithms, compact hardware designs, and power-aware techniques.

Advanced Technologies:

In advanced technology nodes (e.g., 7nm, 5nm, and beyond), the importance of area, power, and delay becomes even more pronounced. These nodes offer increased transistor density but also pose challenges related to leakage power and interconnect delays. BIST solutions must adapt to these technology trends by embracing low-power design techniques, optimizing area utilization, and minimizing signal propagation delays.

4.7. Validation of the Proposed Method using Various LFSRs:

The efficacy of the proposed method is rigorously validated through a comparative analysis involving different LFSRs. This assessment serves as a critical step in understanding the performance enhancements achieved by the proposed approach. Specifically, we examine the power consumption and area requirements of these LFSRs, focusing on their impact on TPG and BIST for a standard 8x8 Braun array multiplier.

Table.4.1, presents the outcomes of evaluation of various LFSR architectures concerning power consumption and area utilization in the context of TPG. These structures include the traditional Standard LFSR, the Modular LFSR, the LFSR with primitive polynomials, the pattern generator based on Cellular Automata, and the newly introduced Low

Power LP-LFSR. Analysis is conducted using a 90nm technology platform.

Table.4.1. Results for Various LFSR Architectures in 90nm Technology

TPG Method	**Area (µm^2)**	**Power (µW)**
Standard	1649	63.62
TPG-LT	1675	62.93
TPG-CA	1667	63.14
TPG-ModPr	1542	58.49
TPG- Mod	1599	57.99

The validation procedure entails conducting simulation analysis through Xilinx ISE version 14.2 and performing synthesis using the Cadence nc simulator tool. We further employ the rc compiler for power analysis. To gauge the impact of LFSR choice, we compare the average power consumption in test mode when using the proposed TPG with different LFSRs against the conventional LFSR. The results highlight the significance of approach in terms of power optimization and area efficiency [81].

Table 4.2. Results for Various LFSR-based BIST Architectures in 90nm Technology

BIST Method	**Area (µm^2)**	**Power (µW)**
Standard	3141	112.15
BIST-LT	3167	92.17
BIST-CA	3159	113
BIST-ModPr	3034	99.55
BIST-Mod	3091	119.87

The examination is expanded in table 4.2 to delve into the domain of Built-In Self-Test (BIST) architectures, wherein evaluate the area demands and power consumption associated with diverse LFSR-based BIST techniques. Much like the evaluation of Test Pattern Generators (TPG), these

techniques encompass the Standard LFSR, BIST-LT, BIST-CA, BIST-ModPr, and BIST-Mod configurations. These LFSRs generate test patterns that are subsequently applied to a comprehensive 8x8 Braun array synchronous multiplier for thorough testing.

The results in Table 4.1 & 4.2 underscore the considerable impact of proposed LFSR optimization approach on BIST performance. Notably, the Low Power-LFSR (LP-LFSR) demonstrates superior power efficiency compared to other LFSRs, contributing to reduced power consumption and enhanced reliability in BIST applications.

The comprehensive validation process showcases the tangible benefits of the proposed LFSR optimization method. It not only reduces power consumption but also enhances area efficiency, making it a promising choice for test pattern generation and Built-In Self-Test architectures in advanced technology nodes.

4.8. LFSR Conclusion:

In the ever-advancing landscape of semiconductor design and testing, the pursuit of low-power solutions stands as an imperative quest. The cornerstone of robust BIST lies within the TPG, the heart of BIST, and the engine responsible for crafting the test patterns that scrutinize the integrity and functionality of integrated circuits (ICs). It's a quest fueled by the relentless march of technology towards power efficiency, and in this endeavor, a pivotal player has emerged—the Modular Linear Feedback Shift Register (LFSR). This explores the transformative impact of Modular LFSR on low-power TPG and its profound implications for the world of BIST.

The Essence of TPG in BIST: Before plunging into the realm of Modular LFSR and its low-power prowess, it's vital to grasp the centrality of Test Pattern Generators (TPGs) in the BIST ecosystem. TPGs are the architects of BIST, entrusted with the responsibility of generating the test patterns

that serve as litmus tests for the health of integrated circuits. The efficacy of BIST hinges directly on the efficiency and reliability of TPGs. It's this facet that beckons us to explore low-power TPGs.

The Quest for Low-Power TPG: The drive for low-power TPG is not a mere engineering whim; it's a necessity forged in the crucible of contemporary semiconductor design. High power consumption in TPGs carries profound ramifications, from thermal challenges and ballooning operational costs to dwindling battery life in portable devices. The need to harness low-power TPGs is not just a matter of desire but a strategic imperative.

The LFSR Landscape: In the relentless pursuit of low-power TPG, researchers and engineers have turned their gaze towards Linear Feedback Shift Registers (LFSRs), probing these structures for the optimal blend of power efficiency and test pattern generation prowess.

Modular LFSR Emerges Victorious: Amidst this landscape of LFSR structures, Modular LFSR stands out as a game-changer in the realm of low-power TPG. Its architecture, built upon the tenets of modularity, ushers in a transformative era of power efficiency during test mode. With meticulous design and innovation, Modular LFSR strikes a harmonious balance, optimizing area utilization while curtailing the energy-intensive switching activities. The net result—a substantial reduction in power consumption when juxtaposed with conventional or standard LFSR-based TPGs.

Implications for BIST: The implications of Modular LFSR for Built-In Self-Test are profound. As BIST becomes an integral component of contemporary ICs, the mantle of power efficiency can no longer be treated as a luxury but as a strategic imperative. Here's how Modular LFSR redefines the landscape of BIST:

Power Efficiency: Modular LFSR delivers a paradigm shift in power

efficiency during test mode. ICs integrated with Modular LFSR-based BIST exhibit superior power efficiency, mitigating the perils of overheating and extending the operational life of the IC.

Area Optimization: Efficient utilization of silicon real estate is another hallmark of Modular LFSR. It demonstrates an uncanny ability to occupy less physical space while preserving the robustness of test pattern generation. This attribute is particularly valuable in advanced semiconductor nodes where space constraints loom large.

Reliability: Low-power TPG, the forte of Modular LFSR, augments the long-term reliability of ICs. By minimizing power-induced stresses, it fortifies the resilience of semiconductor devices.

Sustainability: In an epoch where sustainability is a global clarion call, Modular LFSR-based BIST aligns seamlessly with the broader mission of conserving energy and mitigating environmental impact. It promotes a sustainable approach to semiconductor design and testing.

The Road Ahead: The journey of exploration and innovation in the realm of low-power TPG is a continuum. Modular LFSR marks a significant waypoint, but the voyage is far from over. The trajectory forward is likely to witness the expansion of Modular LFSR's domain, particularly in the domain of Memory BIST, where its advantages in terms of area efficiency and power conservation can potentially be harnessed to even greater effect.

Furthermore, the implementation of Modular LFSR in standard FPGA boards holds immense promise. This foray into FPGA territory promises to unravel insights into the utilization of Look-Up Tables (LUTs) within FPGA fabric—an invaluable facet, especially as BIST stakes its claim within FPGA-based systems.

In the dynamic realm of semiconductor design and testing, Modular LFSR stands as a monumental milestone in the quest for low-power Test

Pattern Generators. It not only ushers in a new era of power efficiency but also signifies a shift towards compact, reliable, and sustainable Built-In Self-Test solutions. As BIST takes center stage in modern ICs, the advantages of Modular LFSR transcend mere power conservation, encompassing area optimization, reliability fortification, and a sustainable ethos. The future, it seems, holds the promise of an energy-efficient, resilient, and forward-looking semiconductor landscape, with Modular LFSR leading the way.

In conclusion, when aiming for less LUT utilization in FPGA-based BIST implementations, the choice of LFSR and TPG techniques significantly influences resource optimization. The analysis suggests that Modular LFSR and related variants, such as Primitive Polynomial Modular LFSR and Low Transition LFSR, are advantageous in reducing LUT consumption while maintaining effective test pattern generation and fault coverage.

Looking ahead, the proposed method highlights the potential for further LUT optimization by exploring Modular LFSR in Memory BIST applications. This endeavor could lead to even greater area and power efficiency. The ability to implement these designs on standard FPGA boards and assess LUT utilization in the fabric is a critical step in ensuring that BIST becomes an integral and resource-efficient component of FPGA-based systems.

In summary, careful selection and analysis of LFSR and TPG techniques, particularly those designed for efficiency, can yield FPGA-based BIST solutions with minimal LUT utilization, addressing the resource constraints of modern FPGA designs.

4.9. PRBS-Based BIST for Multipliers -Introduction:

Multipliers stand as essential components within digital systems, responsible for one of the most basic yet crucial arithmetic operations:

multiplication. Their applications span diverse domains, including signal processing, graphics rendering, cryptographic algorithms, and scientific computing. In modern computing systems, multipliers are expected to meet stringent performance criteria, such as high throughput, low power consumption, and error-free operation. These expectations intensify with the rise of applications like artificial intelligence and machine learning, which demand increasingly intricate multiplication operations.

However, the escalating complexity of modern multipliers and the relentless pursuit of higher performance come with challenges. Designing and validating multipliers have become progressively intricate due to factors such as their complexity, power consumption, fault tolerance, and testing overhead.

4.9.1. Conventional Testing Techniques

Traditionally, verifying the correctness of multipliers has relied on conventional testing methods, which include functional testing, gate-level simulation, stuck-at fault testing, and Built-In Self-Test (BIST). Functional testing entails applying known input values and verifying that the output matches the expected result. While fundamental, it may not encompass all possible scenarios. Gate-level simulation involves using transistor-level models to validate the multiplier's functionality, but it is computationally intensive. Stuck-at fault testing identifies specific faults within the multiplier circuit but may not catch all fault types. BIST techniques integrate testing functionality within the multiplier itself, enabling self-diagnosis but may lack the ability to generate comprehensive test patterns.

Despite their usefulness, these traditional testing methods face limitations, particularly in the context of complex modern multipliers. Thus, there arises a pressing need for innovative testing methodologies that can effectively address these challenges.

4.10. PRBS-Based BIST

The advent of Pseudorandom Binary Sequences (PRBS)-Based Built-In Self-Test (BIST) marks a significant paradigm shift in the verification and validation of multipliers. PRBS sequences, being deterministic yet pseudorandom, exhibit several desirable properties for testing, including a lengthy repetition period and balanced 0/1 transitions. PRBS-Based BIST introduces a novel approach to validating multipliers, leveraging these properties.

Theoretical Foundations of PRBS

Central to PRBS-Based BIST are pseudorandom sequences generated using linear feedback shift registers (LFSRs), well-established mathematical constructs. LFSRs offer an efficient way to produce sequences that, while deterministic, possess characteristics akin to true random sequences. These characteristics include a near-equal distribution of 0s and 1s and a lengthy repetition period.

PRBS-Based BIST Workflow

The workflow of PRBS-Based BIST for multipliers can be summarized in a few key steps. First, a PRBS sequence generator within the multiplier circuit produces a pseudorandom bit sequence, which serves as the input stimulus for testing. The multiplier then operates on this PRBS input sequence, generating a corresponding output sequence. A checker circuit subsequently compares the expected and actual output sequences, with any discrepancies indicating potential faults or errors within the multiplier. If errors are detected, the BIST system works to localize these faults within the multiplier, pinpointing specific components or areas that require attention.

4.10.1. Advantages of PRBS-Based BIST

PRBS-Based BIST offers several notable advantages for verifying multipliers. It provides high test coverage, exploring a wide range of input

scenarios, and uncovering latent faults that traditional techniques may miss. Moreover, it reduces external testing overhead, as the test vectors are generated internally within the multiplier. This is particularly advantageous for multipliers integrated into larger systems. The method is inherently parallel and can be executed at high speeds, making it suitable for real-time and high-performance applications. Its ability to detect and locate errors within the multiplier circuit enhances fault tolerance, a crucial aspect for applications where reliability is paramount. Additionally, PRBS-Based BIST can be configured to adapt to various multiplier architectures and sizes, rendering it a versatile solution for different design scenarios.

BIST for multipliers in FPGA-based DSP slices offers a multitude of advantages that enhance the reliability, performance, and ease of testing:

- Reliability Assurance: BIST continuously monitors the multiplier's functionality, allowing for the early detection of faults or errors. This proactive approach significantly enhances the reliability of DSP systems, especially in safety-critical applications like medical devices and automotive systems.
- Resource Efficiency: By eliminating the need for external test equipment, BIST conserves FPGA resources and reduces the overall system cost. This is particularly beneficial in resource-constrained DSP applications.
- Reduced Downtime: Real-time testing ensures rapid fault detection, minimizing system downtime and facilitating faster fault correction. In mission-critical DSP applications, this can be a decisive factor.
- Comprehensive Testing: BIST's ability to provide high test coverage ensures that the multiplier is rigorously tested under various scenarios, leaving minimal room for undetected faults. This is essential for maintaining the accuracy and integrity of DSP computations.
- Parallel Processing: Leveraging the parallel processing capabilities of FPGAs, BIST can simultaneously test multiple multipliers or arithmetic

units within the DSP slice. This enhances testing efficiency and throughput.

- Fault Localization: BIST not only detects faults but also localizes them within the multiplier circuit. This pinpointing of specific fault locations aids in efficient debugging and maintenance.
- Adaptability: BIST can be tailored to different multiplier architectures and configurations, making it a versatile solution for various DSP algorithms and applications.
- Long-Term Monitoring: BIST can be configured for continuous long-term monitoring of multipliers, ensuring their continued reliability even in harsh operating conditions.

4.10.2. Disadvantages of BIST for Multipliers in FPGA

While BIST for multipliers in FPGA-based DSP slices offers numerous advantages, it is essential to acknowledge potential drawbacks and challenges:

- Additional Hardware Overhead: Implementing BIST requires additional hardware resources within the FPGA, which may reduce the available resources for DSP algorithm implementations. Careful resource management is necessary to strike a balance between testing and computation.
- Complexity: BIST design and implementation can be complex, especially for multipliers with intricate architectures. This complexity may result in longer development times and increased design effort.
- Testing Speed vs. Resource Utilization: Achieving high-speed testing may demand more FPGA resources. Balancing testing speed with resource utilization can be a challenge, particularly in resource-constrained FPGA-based DSP systems.
- Overhead in Area and Power: BIST operations consume both FPGA area and power, potentially affecting the overall system's power

efficiency and physical footprint. Optimization techniques are required to minimize this overhead.

- Test Pattern Generation: Generating comprehensive test patterns for BIST can be a non-trivial task, particularly for complex multipliers. Ensuring that the test patterns cover a wide range of potential faults requires careful consideration.

4.11. Introduction to SerDes in FPGA

Field-Programmable Gate Arrays (FPGAs) are versatile semiconductor devices widely used in various applications, including digital communication systems. Within these FPGAs, Serializer/Deserializer (SerDes) modules play a pivotal role in achieving high-speed data transmission. Multipliers are essential components within SerDes modules, performing complex arithmetic operations critical for data manipulation and signal processing. Ensuring the correctness and reliability of multipliers is paramount in such high-speed communication systems. Built-In Self-Test (BIST) techniques have emerged as indispensable tools for verifying the functionality and robustness of multipliers integrated into SerDes modules. This comprehensive discussion explores the unique role of BIST in multipliers within FPGA-based SerDes, emphasizing its features, advantages, disadvantages, and real-world implementation.

4.11.1. The Role of Multipliers in SerDes

Within the SerDes modules of FPGAs, multipliers are essential components for various tasks, including equalization, clock data recovery, and error correction. These multipliers perform complex arithmetic operations on incoming data streams, contributing to the overall functionality and performance of the SerDes circuitry. However, due to the high-speed and mission-critical nature of SerDes applications, ensuring the correctness and reliability of these multipliers becomes paramount.

SerDes modules are at the heart of many communication systems, enabling the efficient transfer of digital data between devices by converting

parallel data streams to serial and vice versa. Multipliers within these modules are responsible for critical operations such as equalization, clock data recovery, and error correction. The proper functioning of these multipliers is crucial for maintaining data integrity and achieving high throughput in communication systems.

However, testing multipliers within SerDes modules poses several significant challenges:

- High-Speed Operation: SerDes modules operate at extremely high data rates, making it challenging to apply traditional testing methods that may introduce significant overhead.
- Complex Architectures: Multipliers within SerDes modules are often designed with intricate architectures optimized for high-speed data processing. Verifying their correctness is a non-trivial task.
- Fault Tolerance: Fault tolerance is critical in SerDes applications, where errors can lead to data corruption and communication failures. Robust testing is essential to detect and mitigate faults.
- Diverse Data Patterns: SerDes modules must handle a wide range of data patterns and transitions, necessitating comprehensive testing to ensure correct operation under all conditions.

4.11.2. The Role of BIST in SerDes Multipliers:

BIST techniques offer a powerful solution to address the challenges of testing multipliers within FPGA-based SerDes modules. BIST is an on-chip testing approach integrated directly into the FPGA hardware. It enables real-time testing, high test coverage, and fault tolerance enhancement, making it an ideal choice for ensuring the reliability of multipliers within SerDes applications.

- On-Chip Integration: BIST is seamlessly integrated into the FPGA's SerDes module, eliminating the need for external testing equipment. This integration simplifies the testing process, reduces testing

overhead, and ensures that testing functionality is readily available when needed.

- Real-Time Testing: BIST can perform real-time testing of multipliers within the SerDes module while the module is actively processing data. This continuous testing ensures that any faults or errors are promptly detected, minimizing the risk of data corruption or communication failures.
- High Test Coverage: BIST can provide high test coverage, systematically verifying the multiplier's correctness under various data patterns and operating conditions. This thorough testing helps uncover potential faults that might go undetected with conventional methods.
- Fault Tolerance Enhancement: BIST not only detects faults but can also enhance fault tolerance within the SerDes module. By continuously monitoring the multiplier's operation, BIST can trigger error recovery mechanisms or reconfiguration to mitigate the impact of faults.
- Parallel Testing: FPGA-based SerDes modules often support parallel data processing. BIST can leverage this parallelism to simultaneously test multiple multipliers within the module, ensuring efficient testing without sacrificing performance.
- Adaptability: BIST can be configured to adapt to different multiplier architectures and configurations commonly found in SerDes applications. This flexibility makes it a versatile solution for various FPGA-based SerDes designs.

4.11.3. Advantages of BIST in SerDes Multipliers

The integration of BIST for multipliers within FPGA-based SerDes modules offers several notable advantages:

- Reliability Assurance: BIST ensures continuous monitoring and testing of multipliers, enhancing the overall reliability and fault tolerance of the SerDes module.

- Reduced Downtime: Real-time testing minimizes the risk of data corruption and communication failures, reducing system downtime and enhancing the availability of SerDes-based applications.
- Comprehensive Testing: BIST's ability to provide high test coverage ensures that multipliers are rigorously tested under diverse data patterns and operating conditions, ensuring correctness in all scenarios.
- Parallel Testing: BIST's support for parallel testing aligns with the parallel data processing capabilities of FPGA-based SerDes modules, maintaining high throughput.
- Adaptability: BIST can be tailored to the specific SerDes design, accommodating different multiplier architectures and configurations.

4.11.4. Disadvantages and Considerations

While BIST offers significant advantages, there are some considerations to keep in mind:

- Resource Utilization: Implementing BIST within the FPGA's SerDes module consumes resources. Careful resource management is necessary to ensure that sufficient resources are allocated to both BIST and the SerDes processing.
- Complexity: Designing and implementing BIST for complex SerDes multipliers may be intricate and require additional development effort.
- Configuration and Control: BIST requires appropriate configuration and control mechanisms to adapt to different SerDes designs and efficiently perform testing.
- Overhead: BIST operations may introduce some overhead in terms of area and power consumption within the FPGA. Optimization techniques should be considered to minimize this overhead.

4.12. Real-World Implementation of BIST in SerDes Multipliers

Implementing BIST for multipliers within FPGA-based SerDes

modules involves several key steps and considerations:

- On-Chip BIST Controller: A dedicated BIST controller is integrated into the FPGA's SerDes module. This controller is responsible for initiating and managing the testing process. It configures the BIST module, controls the test sequence, and collects test results. The controller typically operates as a finite state machine (FSM) within the FPGA.
- BIST Module: The BIST module resides within the FPGA's SerDes module and interacts directly with the multiplier being tested. This module generates test vectors based on the BIST algorithm, applies them to the multiplier's inputs, captures the output responses, and compares them against expected results.
- Test Pattern Generation: Generating comprehensive test patterns for BIST is a critical aspect of the implementation. This involves designing algorithms to systematically explore different data patterns, including boundary cases, corner cases, and random inputs. Test patterns should be tailored to the specific multiplier architecture and size.
- Error Detection and Localization: The BIST module includes error detection and localization mechanisms to identify faults or errors within the multiplier. This can involve signature analysis, parity checks, or other fault detection techniques. Once an error is detected, localization mechanisms pinpoint the specific component or area of the multiplier circuit that is faulty.
- Configurability: BIST should be configurable to adapt to different multiplier architectures and configurations commonly encountered in SerDes applications. Parameters such as test sequence length, frequency of testing, and fault detection thresholds should be adjustable to meet specific application requirements.
- Parallel Testing: Exploiting the parallel processing capabilities of FPGAs, BIST can simultaneously test multiple multipliers within the

SerDes module, ensuring efficient testing without compromising performance.

- Integration with SerDes Algorithms: BIST should seamlessly integrate with the SerDes algorithms running on the FPGA. This integration allows for concurrent testing and data processing, minimizing any impact on SerDes performance.
- Fault Reporting and Logging: BIST should provide mechanisms for reporting and logging detected faults. This information is invaluable for post-testing analysis, debugging, and maintenance [86], [89], [90].

4.12.1. Integration of BIST for Multipliers

Integrating BIST for multipliers within FPGA-based SerDes modules involves several crucial steps:

- On-Chip BIST Controller: A dedicated BIST controller is seamlessly integrated into the FPGA's SerDes module. This controller manages the testing process, configures the BIST module, controls the test sequence, and collects test results. It typically operates as a finite state machine (FSM) within the FPGA.
- BIST Module: The BIST module resides within the FPGA's SerDes module and interacts directly with the multiplier being tested. This module generates test vectors based on the BIST algorithm, applies them to the multiplier's inputs, captures the output responses, and compares them against expected results.
- Test Pattern Generation: Generating comprehensive test patterns for BIST is a critical aspect of the implementation. This involves designing algorithms to systematically explore different data patterns, including boundary cases, corner cases, and random inputs. Test patterns must be tailored to the specific multiplier architecture and size.
- Error Detection and Localization: The BIST module includes error detection and localization mechanisms to identify faults or errors within

the multiplier. This can involve signature analysis, parity checks, or other fault detection techniques. Once an error is detected, localization mechanisms pinpoint the specific component or area of the multiplier circuit that is faulty.

- Configurability: BIST must be configurable to adapt to different multiplier architectures and configurations commonly encountered in SerDes applications. Parameters such as test sequence length, frequency of testing, and fault detection thresholds should be adjustable to meet specific application requirements.
- Parallel Testing: Exploiting the parallel processing capabilities of FPGAs, BIST can simultaneously test multiple multipliers within the SerDes module, ensuring efficient testing without compromising performance.
- Integration with SerDes Algorithms: BIST should seamlessly integrate with the SerDes algorithms running on the FPGA. This integration allows for concurrent testing and data processing, minimizing any impact on SerDes performance.
- Fault Reporting and Logging: BIST should provide mechanisms for reporting and logging detected faults. This information is invaluable for post-testing analysis, debugging, and maintenance.

4.12.2. Advantages of PRBS-Based BIST for Multipliers

The integration of PRBS-Based BIST for multipliers within FPGA-based SerDes modules offers several significant advantages:

- Reliability Assurance: PRBS-Based BIST ensures continuous monitoring and testing of multipliers, enhancing the overall reliability and fault tolerance of the SerDes module.
- Reduced Downtime: Real-time testing minimizes the risk of data corruption and communication failures, reducing system downtime and enhancing the availability of SerDes-based applications.

- Comprehensive Testing: PRBS-Based BIST's ability to provide high test coverage ensures that multipliers are rigorously tested under diverse data patterns and operating conditions, ensuring correctness in all scenarios.
- Parallel Testing: PRBS-Based BIST's support for parallel testing aligns with the parallel data processing capabilities of FPGA-based SerDes modules, maintaining high throughput.
- Adaptability: PRBS-Based BIST can be tailored to the specific SerDes design, accommodating different multiplier architectures and configurations.

4.12.3. Disadvantages and Considerations

While PRBS-Based BIST offers significant advantages, several considerations should be kept in mind:

- Resource Utilization: Implementing PRBS-Based BIST within the FPGA's SerDes module consumes resources. Careful resource management is necessary to ensure that sufficient resources are allocated to both BIST and the SerDes processing.
- Complexity: Designing and implementing PRBS-Based BIST for complex SerDes multipliers may be intricate and require additional development effort.
- Configuration and Control: PRBS-Based BIST requires appropriate configuration and control mechanisms to adapt to different SerDes designs and efficiently perform testing.

4.13. Recent Research Contributions

Recent research in the field of PRBS-Based BIST for multipliers has resulted in notable advancements:

Low-Power PRBS Generator: Wei-Zen Chen et al. [78] introduced a PRBS generator utilizing a 1 to 16 demultiplexer, significantly reducing the clock rate and power consumption. This approach is advantageous for loop-back

testing.

High-Speed PRBS Generation and Checking: Wang Ying et al. [79] demonstrated a 27-1 PRBS generator and checker operating at data transfer rates of up to 2.5 Gbps. These modules are power-efficient and support a broad operating range.

Half-Rate BIST: A half-rate BIST, utilizing a combination of PRBS generator and bit error checker, has been introduced [80]. This approach provides accurate error measurement for high-speed serial data communication.

Pattern Generator for Fault Coverage: Zhiqian Zhang et al. conducted software simulations to determine effective repeat patterns and fault models, achieving a fault coverage of 99.57% [87].

Array Multipliers Testing: A.S. Oyeniran et al. [84] proposed an approach for array multipliers using pseudo-exhaustive testing, resulting in reduced complexity of PET pattern synthesis.

Low-Power BIST for Multipliers: T. Srinivasa Rao et al. [83] incorporated a low-power Test Pattern Generator (TPG) BIST for evaluating Vedic multipliers, reducing switching activities and increasing fault coverage.

Enhanced PET Approaches: A.S. Oyeniran et al. [84] introduced several revisions of pseudo-exhaustive testing (PET) approaches with regular structures to enhance their features, applicable to both software-based self-test and hardware-based logic BIST.

4.14. Importance of Testing in IC Design

In the domain of integrated circuit (IC) design, the importance of chip manufacturing tests and design verification tests is paramount, especially in the context of devices incorporating high-speed interfaces. The intricacies of modern designs often necessitate the incorporation of Intellectual Property (IP) macros to expand test vector lengths. However, this extension comes at the cost of increased test duration and the

requirement for expensive testing equipment. To mitigate these challenges and enhance cost-efficiency, the utilization of BIST techniques has emerged as an area of expertise.

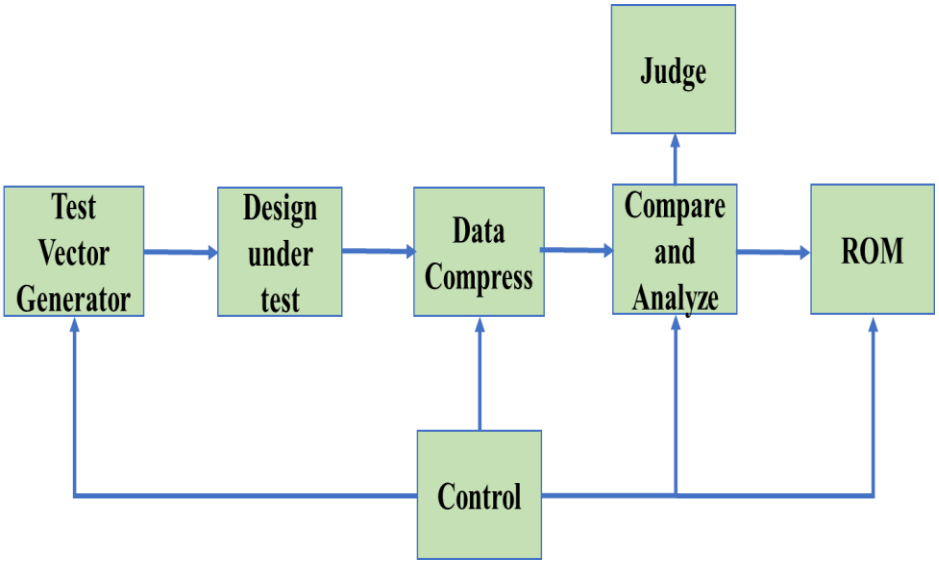


Fig.4.5. Conventional Structure of BIST

Despite the additional on-chip footprint and associated costs, BIST proves advantageous by reducing the dependence on external inputs and outputs required for testing, thereby streamlining the process [36], [80]. Fig 4.5 provides an overview of the conventional BIST architecture, comprising essential components such as the test vector generator, the Device Under Test (DUT), ROM, data compression, judgment/results analysis, control, comparison, and analysis.

4.15. Integration of PRBS and BIST for SERDES Testing

The fusion of PRBS and BIST emerges as a widely accepted and highly effective strategy for evaluating Serializer/Deserializer (SERDES) modules. The accompanying Fig 4.6 provides an insight into the typical structure of SERDES when integrated with BIST. Within this testing configuration, the selection of PRBS is contingent upon the specific multiplexers and demultiplexers in use [85]. The DUT encompasses essential modules for serialization, deserialization, and Clock Data Recovery (CDR). Through the seamless integration of a transceiver with a

PRBS generator-checker, this approach streamlines Bit Error Rate (BER) testing, eliminating the need for costly external testing equipment.

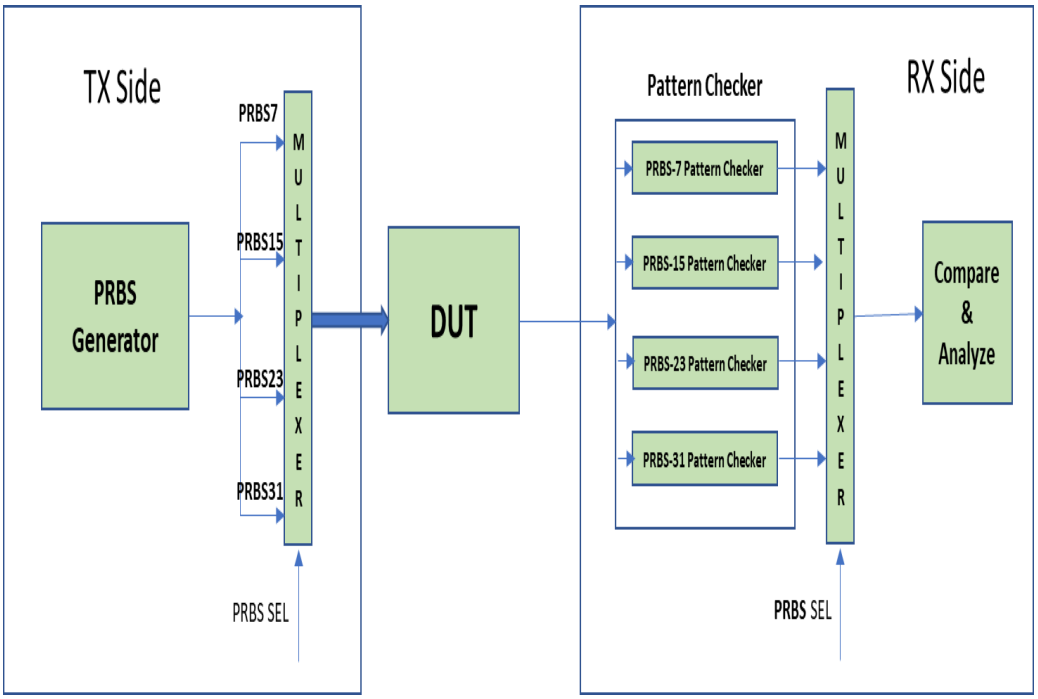


Fig.4.6. PRBS (Generator -Checker) for SERDES transceivers

4.16. Advanced Testing in Ultra-Scale Architectures

Within the realm of Ultra-scale architectures, the GTH transceivers truly shine as exemplars of high-power efficiency, capable of operating at line rates spanning from 500 Mbps to an impressive 16.375 Gbps. These transceivers offer a remarkable degree of configurability and seamlessly integrate with the Programmable Logic (PL) Look-Up Table (LUT) resources inherent to the architecture.

One particularly noteworthy feature of these transceivers is their Datapath, which lends itself to specialized, configurable Loop-back modes. These modes empower the transceivers to transmit streaming data back to the source through the receiver, a valuable capability for testing and validation purposes. In practice, each clock cycle typically carries a unique traffic pattern or stream, allowing for the detection of errors by comparing the received data at the receiver's end.

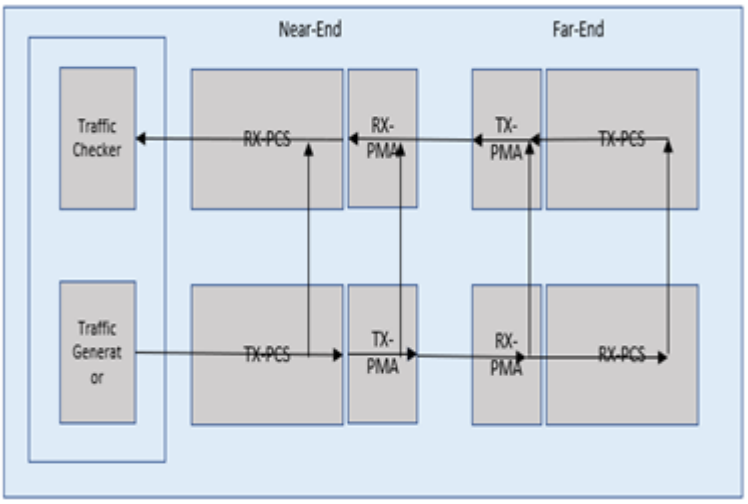


Fig.4.7. GTH transceivers Near-end and Far-end Loopback Modes

The Loopback test modes fall into two distinct categories, each serving a specific purpose. First, there's the Far-end loop-back mode, where the transmitter data is looped back within the transceiver situated at the far end of the communication link. Second, the Near-end loop-back mode involves looping back the transmitter data within the transceiver located nearest to the traffic generator. Fig 4.7 accompanying this text effectively illustrates these two loop-back modes, elucidating their functionality within the GTH transceivers' versatile repertoire.

4.16.1. Application of Loopback Testing

Loopback testing is a widely adopted practice for pinpointing faults in hardware equipment, a critical endeavor during both the development and testing phases. This method entails the transmission of traffic featuring predefined application-based traffic patterns or distinctive random bit sequences. What distinguishes each GTH transceiver is its built-in PRBS generator and checker, empowering the transmission of data with varying PRBS width lengths.

The proposed design adopts a robust approach, incorporating a PRBS generator on the transmitter side and a PRBS checker on the receiver side, both meticulously implemented using the Verilog Hardware

Description Language (HDL). To ensure optimal performance, dedicated GTH transceivers are strategically deployed for data transmission and reception, thereby facilitating the seamless flow of information.

What further enhances the testing and validation process is the introduction of randomized data, a vital component in evaluating the SERDES module's performance. This data undergoes meticulous processing via an internal hybrid multiplier, contributing significantly to the comprehensive assessment of the module's functionality and reliability. GTH transceivers, like many other high-speed serial communication devices, offer Near-end Loopback (NEL) and Far-end Loopback (FEL) modes that play crucial roles in testing and debugging scenarios. These modes are designed to ensure the transceiver's functionality, verify signal integrity, and troubleshoot potential issues. Let's delve into the specifics of both loopback modes and explore some typical run-time scenarios where they are utilized.

Near-end Loopback (NEL) Mode:

In Near-end Loopback mode, the transceiver retransmits the data it has received back to itself without sending it over the communication channel. This loopback mode allows for self-testing and is particularly valuable during the development and validation of the transceiver interface. Here are some unique characteristics and scenarios associated with NEL mode:

1. Self-Test and Validation: NEL mode is primarily used for self-testing and validating the integrity of the transmitter and receiver within the transceiver. It helps confirm that the transceiver can correctly send and receive data without external interference.

2. Bit Error Rate (BER) Testing: NEL mode is indispensable when conducting Bit Error Rate (BER) testing. By comparing the transmitted and received data, engineers can assess the quality of the communication link and measure the error rate accurately.

3. Initialization and Calibration: During the initialization phase,

transceivers often employ NEL mode to ensure proper setup and calibration. It allows for adjustments to be made to optimize signal quality.

Far-end Loopback (FEL) Mode:

Far-end Loopback mode involves sending a loopback command from one end of the communication link to the far end, where the data is looped back and returned to the originating end. This mode is particularly useful for testing the entire communication path, including the channel between the transmitter and receiver. Here are some unique aspects and scenarios for FEL mode:

1. End-to-End Testing: FEL mode enables comprehensive end-to-end testing of the communication link, encompassing both the transceivers and the transmission medium (cables or fiber optic links). It verifies the functionality of the entire system.

2. Diagnosing Channel Issues: FEL mode is instrumental in diagnosing issues related to the communication channel. By looping back, the data at the far end, engineers can pinpoint problems such as cable faults, signal attenuation, or excessive noise.

3. Remote Testing: In scenarios where the far end of the communication link is physically distant or not readily accessible, FEL mode allows for remote testing and troubleshooting without the need for on-site personnel.

4.16.2. Run-Time Scenarios:

- Manufacturing Testing: During the manufacturing process of transceiver modules, both NEL and FEL modes are employed to ensure that the devices meet the specified performance criteria. This includes verifying signal integrity, data integrity, and compliance with communication standards.
- System Integration: When integrating transceivers into larger systems, NEL and FEL modes are used to verify that the transceivers can communicate effectively within the context of the entire system. This

is critical for applications such as data centers, telecommunications networks, and high-performance computing clusters.

- Field Troubleshooting: In the field, NEL and FEL modes are invaluable for diagnosing and resolving communication issues. Technicians can use these modes to isolate problems, whether they stem from transceiver malfunctions, cable faults, or other external factors.

Near-end Loopback (NEL) and Far-end Loopback (FEL) modes are essential tools in the toolkit of engineers and technicians working with GTH transceivers and similar high-speed communication devices. These modes facilitate comprehensive testing, validation, and troubleshooting, ensuring the reliability and performance of critical communication links in various applications.

The PRBS generator serves as a critical component in the generation of self-aligned patterns and the creation of distinctive sequences. It accomplishes this task by utilizing an LFSR to determine the subsequent pattern in the sequence based on the current one [82]. These PRBS generators, in conjunction with checkers, are highly relevant in the context of SERDES (Serializer/Deserializer) GTH IOs, accommodating both 8-bit and 64-bit PCS-PMA Interfaces. When the PRBS Checker receives a segment of streamed data, it not only generates the subsequent bit pattern or vector but also rigorously evaluates the accuracy of the received data sequence, promptly detecting and flagging errors or discrepancies. Depending on the specific testing mode, the PRBS can be seamlessly integrated with BIST for SERDES applications, achieving this integration by directly connecting it to the parallel data-in register.

The proposed architecture of the PRBS generator introduces multiplexed outputs via a series-parallel design. The choice of PRBS data length is intricately tied to the selection of multiplexers, with this architecture proving particularly advantageous for high data rates due to its minimal delay, thanks to the involvement of two XOR gates. The Fig 4.8

accompanying this description provides a visual representation of the series-parallel PRBS-7 generator. Additionally, by employing specific polynomial functions, PRBS generators such as PRBS-15, PRBS-23, and PRBS-31 can be meticulously crafted and tailored to meet precise requirements.

In the current research endeavor, a PRBS-7 generator boasting a sequence length of 2^7-1 is judiciously employed to target output data rates spanning the spectrum from 1 to 12.3125 Gbps. The polynomial governing this PRBS-7 generator is succinctly expressed as g7(x) = 1+x^6+x^7, which in turn generates PRB sequences of length 2^7-1.

The proposed PRBS generator, underpinning its functionality on the synergy of flip-flops and XOR gates, ensures the synchronization of its output with the clock signal. This methodological approach yields unique sequences that are indispensable for the seamless operation of the checker circuit, assuring synchronization and alignment between the generator and the checker. To further cement this design, Verilog code is diligently employed to construct PRBS-15, PRBS-23, and PRBS-31, each of which is meticulously synthesized for Ultra-scale devices.

The integration of the PRBS generator into the system architecture brings forth a powerful tool for generating and assessing data patterns, making it an invaluable asset in high-speed data transmission and testing scenarios. This capability is particularly pertinent in the realm of SERDES applications, where data integrity and synchronization are of utmost importance.

The PRBS generator plays a pivotal role in generating self-aligned patterns and unique sequences, with the capability to adapt to varying data rates and testing requirements. Its synchronization with the clock signal, achieved through flip-flops and XOR gates, ensures accurate and reliable data generation. Together with the PRBS checker, it forms a robust testing framework essential for verifying the performance and integrity of SERDES modules in Ultra-scale devices.

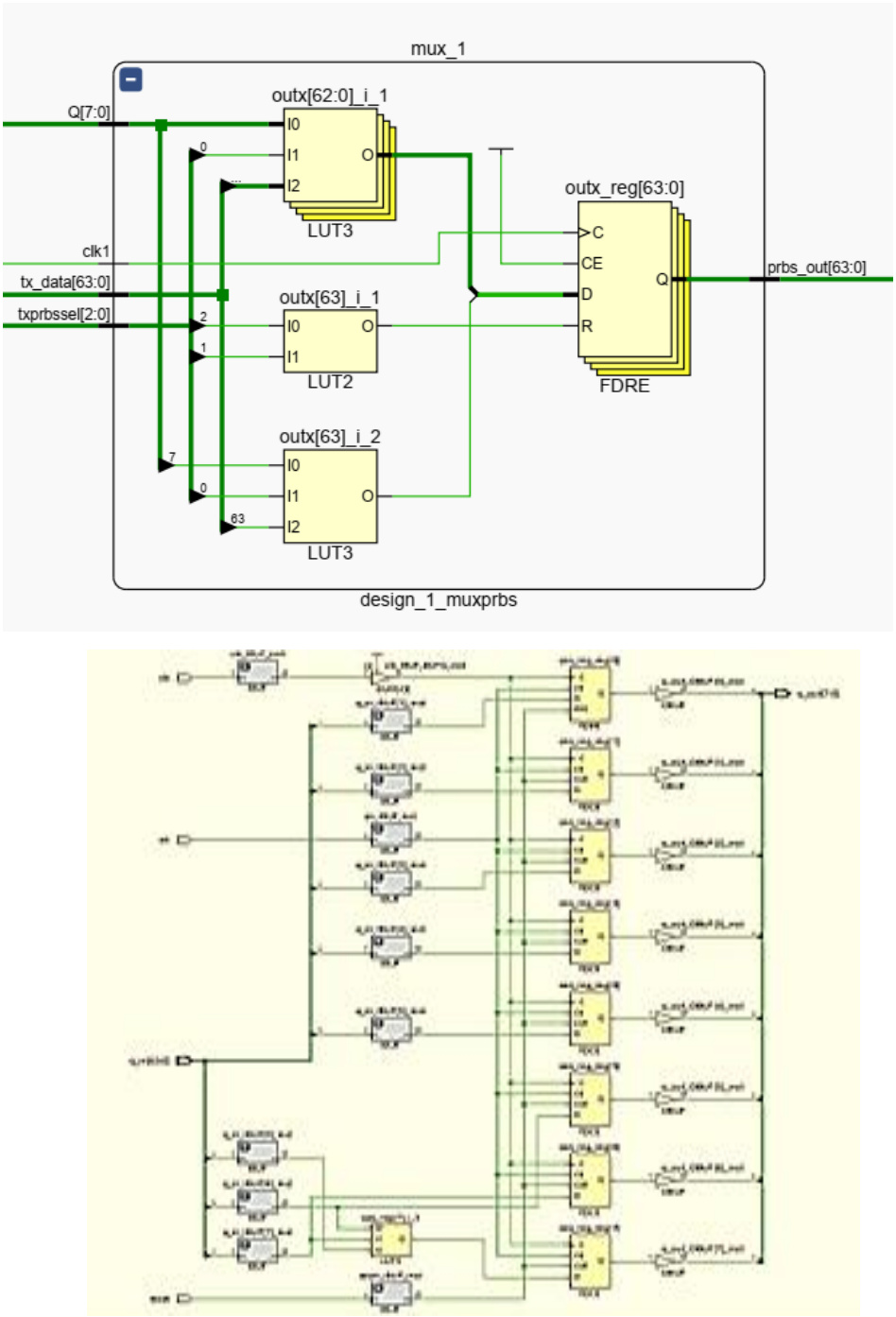


Fig.4.8. Schematic of series-parallel PRBS-7 generator

The depicted Fig 4.9 provides an illustration of the proposed PRBS checker, a fundamental component in advanced communication systems, particularly those involving high-speed data transmission. This checker comprises critical elements, including a CDR (Clock Data Recovery)

module, a clock synchronizer, a PRBS sequence generator, an error detector, and a sequence comparator. Together, these components form a robust testing framework for ensuring data integrity and reliability in SERDES (Serializer/Deserializer) modules.

The synchronization detection circuit within the PRBS checker is of paramount importance, as it is responsible for identifying distinctive data patterns within incoming data streams. On the transmitter side, the PRBS generator plays a dual role in reference PRBS generation and data shifting, utilizing shift registers within the same circuit. To ensure precise alignment and accuracy, it is recommended to employ two identical pattern generators—one on the transmitter side to generate a random pattern and the other on the receiver side to generate a reference pattern derived from the received data. A control logic unit orchestrates the initiation of the PRBS generator on the receiver side and identifies data streams characterized by unique patterns.

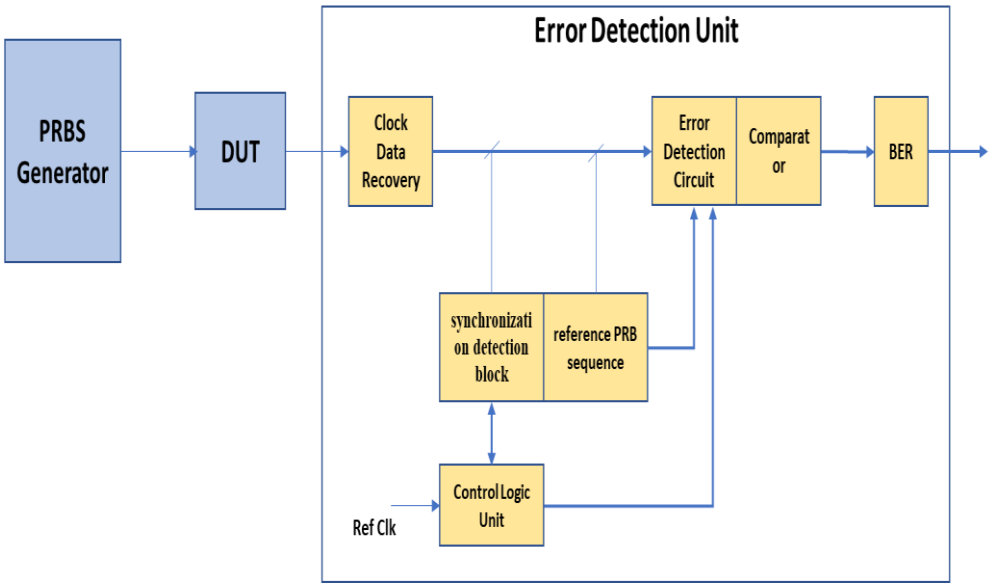


Fig.4.9. Proposed PRBS checker

The core of the PRBS checker's functionality lies in error detection. This process involves transmitting the generated pattern to the synchronization detection unit, where it undergoes a rigorous comparison

against the reference PRBS-generated data. The checker meticulously generates error pulses, quantifying the number of errors detected during the data transmission process, both within and outside the integrated circuit. This error data proves invaluable when evaluating the Bit Error Rate (BER), a critical performance metric. The error detection unit, often referred to as the bit error checker, assumes a central role in this process, ensuring the accuracy and reliability of error assessment [88].

Furthermore, the synchronization of incoming data with the reference PRBS pattern, known as the local PRBS pattern, relies on the initial set or transition of output signals originating from the error checker. Each of these output transitions serves as a capture point for identifying bit errors, distinguishing between true errors and false alarms. This data is subsequently processed by the Device Under Test (DUT) to precisely calculate the BER, a vital parameter for assessing the overall performance and quality of the data transmission system.

In essence, the PRBS checker, with its intricate components and error detection capabilities, stands as a critical asset in the realm of research and development for high-speed communication systems, offering precise insights into data integrity and synchronization within SERDES modules.

4.17. Synthesis and Implementation Results

The proposed architecture for the PRBS generator and Checker has undergone comprehensive evaluation, specifically tailored for an application requiring a high data rate of 12.3125 Gbps. The evaluation was conducted on a Xilinx board using cutting-edge 16nm FinFET technology. The simulation results have been succinctly summarized in Table 4.3.

Let's delve into the significance of these results and their implications:

Technology Advancement: The transition from 28nm to 16nm technology reflects a significant leap in integration and efficiency. The move to 16nm technology not only allows for a reduction in supply voltage (from 900 mV to 800 mV) but also offers the capability to handle higher data rates, as

evidenced by the increase in the PRBS generator's output data rate from 10 Gbps to 12.3125 Gbps.

Jitter Reduction: Jitter, a crucial parameter in high-speed communication, exhibits an improvement in the proposed design. The worst-case jitter has decreased from 0.595 ps to 0.576 ps, showcasing the enhanced signal stability achieved with the new architecture.

Power Efficiency: The quest for power-efficient designs is evident. Both the PRBS generator and checker in the proposed design consume less power under nominal conditions.

Table 4.3. Simulation Results

Parameter	Reference Design	Proposed Design
Technology used	28 nm	16nm
Supply voltage	900 mV	800 mV
PRBS generator		
Clock frequency	5 GHz	156.25 MHz
Output data rate	10 Gbps	12.3125 Gbps
Worst case jitter	0.595 ps	0.576
Average power consumption	(Nominal conditions)	(Nominal conditions)
PRBS checker		
Maximum data rate	13 Gbps	12.3125 Gbps
Average power consumption	(Nominal conditions)	(Nominal conditions)

The PRBS generator's average power consumption has been reduced from 1.642 mW to 1.25 mW, while the PRBS checker's average power consumption has decreased from 3.583 mW to 2.73 mW. This signifies a commendable effort to optimize power usage, a critical consideration in

modern electronic devices.

The practical implementation of this architecture on a Xilinx Ultra-scale board for a line rate of 12.3125 Gbps is exemplified in Fig 4.10, illustrating the interconnection between the PRBS generator, PRBS checker, and the Aurora IP. This setup is pivotal for testing and validating high-speed data transmission.

Moreover, Fig 4.11 presents the BER (Bit Error Rate) - Eye Diagram, providing valuable insights into data integrity at the 12.3125 Gbps line rate. This eye pattern, generated based on the system depicted in Fig 7, was meticulously executed on a Xilinx FPGA with the utilization of Vivado, underlining the practicality and effectiveness of the proposed design.

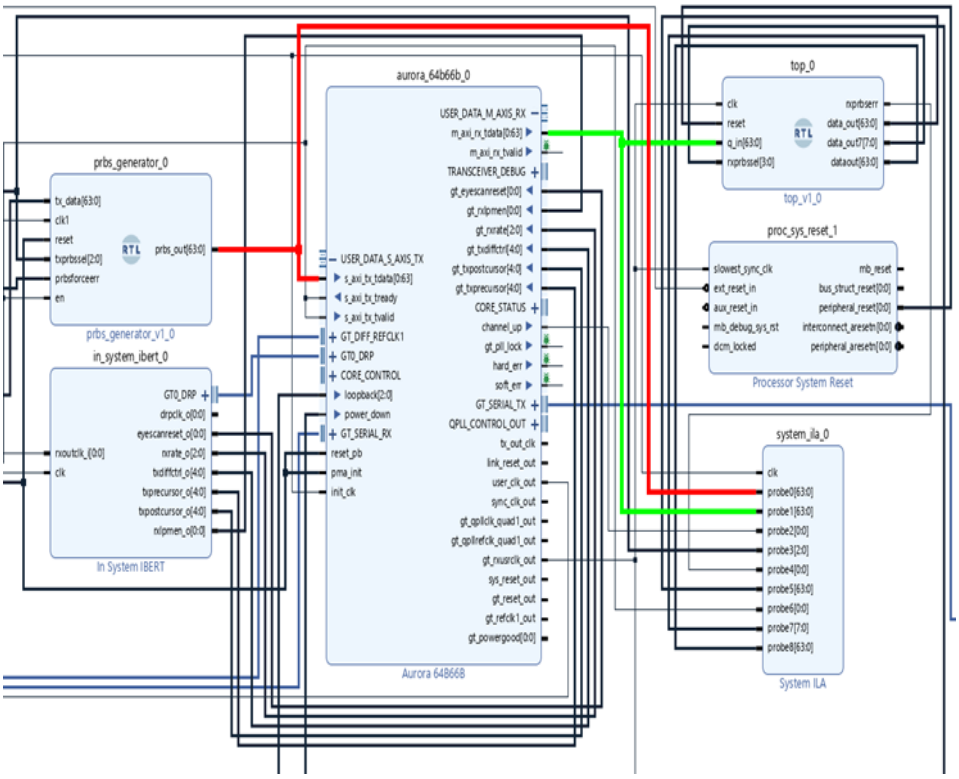


Fig.4.10. IP based transmission loopback using PRBS generator and Checker

To ensure robust data reception, the system incorporates a detection behavior, as described in Fig 4.10. This behavior operates based on a set condition: in the last three periods of PRBS data, the receiver data should

manifest at least twice. This criterion not only enhances error detection capabilities but also enables the PRBS generator to send an indication signal for synchronization. If an error occurs in the received data, a new dynamic-detection cycle is initiated.

In Fig 4.13, a comprehensive comparison of PRBS7, PRBS15, PRBS23, and PRBS31 generated and received data is showcased. This comparison is instrumental in assessing the accuracy and fidelity of the generated PRBS sequences, demonstrating the robustness of the design in handling various PRBS patterns.

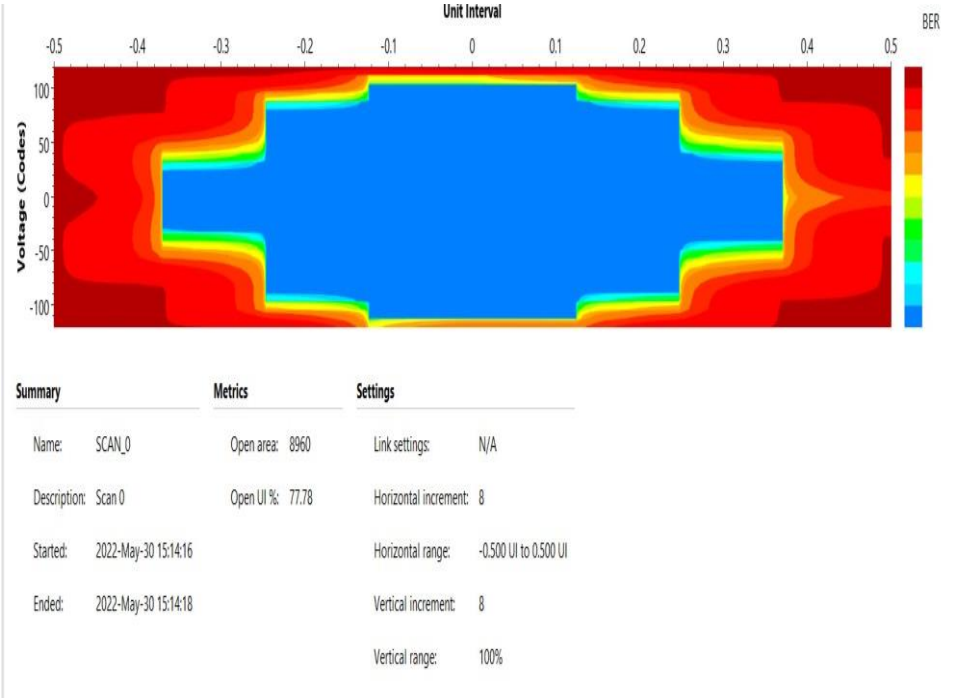


Fig.4.11. BER – Eye Diagram for 12.3125 Gbps data rate

In conclusion, the proposed PRBS generator and Checker architecture have not only demonstrated significant technological advancements but have also underscored the commitment to power efficiency and signal integrity. The practical implementation, illustrated through figures and simulation results, showcases the design's effectiveness in achieving a high data rate of 12.3125 Gbps while minimizing power consumption and mitigating jitter [87].

Table 4.4. PRBS7, PRBS15, PRBS23 and PRBS31 generated and received data for comparison.

PRBS-7		PRBS-15		PRBS-23		PRBS-31	
ebebebeb	f5f5f5f5f	61a861a8	86a086a0	3b3979f5	9d9c3cfa	db63570f	edb1ab87
ebebebeb	5f5f5f5	61a861a8	86a086a0	0c79f50c	863cfa86	db63570f	edb1ab87
f5f5f5f5f	fafafafafa	b0d4b0d4	c350c350	9d9c3cfa	cece9e7d	edb1ab87	76d8d5c3
5f5f5f5	fafafa	b0d4b0d4	c350c350	863cfa86	439e7d43	edb1ab87	76d8d5c3
fafafafafa	7d7d7d7d	d86ad86a	61a861a8	cece9e7d	6767cf3e	76d8d5c3	3b6c6ae1
fafafa	7d7d7d7d	d86ad86a	61a861a8	439e7d43	a1cf3ea1	76d8d5c3	3b6c6ae1
7d7d7d7d	3e3e3e3e	6c356c35	b0d4b0d4	6767cf3e	33b3e79f	3b6c6ae1	1db63570
7d7d7d7d	3e3e3e3e	6c356c35	b0d4b0d4	a1cf3ea1	50e79f50	3b6c6ae1	1db63570
3e3e3e3e	1f1f1f1f1	361a361a	d86ad86a	33b3e79f	99d973cf	1db63570	8edb1ab8
3e3e3e3e	f1f1f1f	361a361a	d86ad86a	50e79f50	a873cfa8	1db63570	8edb1ab8
1f1f1f1f1	8f8f8f8f8	1b0d1b0d	6c356c35	99d973cf	4cec39e7	8edb1ab8	c76d8d5c
f1f1f1f	f8f8f8f	1b0d1b0d	6c356c35	a873cfa8	d439e7d4	8edb1ab8	c76d8d5c
8f8f8f8f8	47474747	8d868d86	361a361a	4cec39e7	a6761cf3	c76d8d5c	e3b6c6ae
f8f8f8f	47474747	8d868d86	361a361a	d439e7d4	ea1cf3ea	c76d8d5c	e3b6c6ae
47474747	23232323	c6c3c6c3	1b0d1b0d	a6761cf3	d33b8e79	e3b6c6ae	f1db6357
47474747	23232323	c6c3c6c3	1b0d1b0d	ea1cf3ea	f58e79f5	e3b6c6ae	f1db6357
23232323	91919191	e361e361	8d868d86	d33b8e79	e99dc73c	f1db6357	f8edb1abf
23232323	91919191	e361e361	8d868d86	f58e79f5	fac73cfa	f1db6357	8edb1ab
91919191	48484848	f1b0f1b0f	c6c3c6c3	e99dc73c	f4ce639e	f8edb1abf	fc76d8d5f
91919191	48484848	1b0f1b0	c6c3c6c3	fac73cfa	7d639e7d	8edb1ab	c76d8d5
48484848	a4a4a4a4	78d878d8	e361e361	f4ce639e	7a67b1cf	fc76d8d5f	fe3b6c6af
48484848	a4a4a4a4	78d878d8	e361e361	7d639e7d	3eb1cf3e	c76d8d5	e3b6c6a
a4a4a4a4	d2d2d2d2	bc6cbc6c	f1b0f1b0f	7a67b1cf	3d33d8e7	fe3b6c6af	7f1db635
a4a4a4a4	d2d2d2d2	bc6cbc6c	1b0f1b0	3eb1cf3e	9fd8e79f	e3b6c6a	7f1db635
d2d2d2d2	69696969	de36de36	78d878d8	3d33d8e7	9e996c73	7f1db635	3f8edb1a
d2d2d2d2	69696969	de36de36	78d878d8	9fd8e79f	cf6c73cf	7f1db635	3f8edb1a
69696969	34343434	6f1b6f1b	bc6cbc6c	9e996c73	4f4c3639	3f8edb1a	9fc76d8d
69696969	34343434	6f1b6f1b	bc6cbc6c	cf6c73cf	e73639e7	3f8edb1a	9fc76d8d
34343434	1a1a1a1a	378d378d	de36de36	4f4c3639	a7a61b1c	9fc76d8d	cfe3b6c6c
34343434	1a1a1a1a	378d378d	de36de36	e73639e7	f31b1cf3	9fc76d8d	fe3b6c6
1a1a1a1a	0d0d0d0d	9bc69bc6	6f1b6f1b	a7a61b1c	d3d38d8e	cfe3b6c6c	e7f1db63
1a1a1a1a	0d0d0d0d	9bc69bc6	6f1b6f1b	f31b1cf3	798d8e79	fe3b6c6	e7f1db63
0d0d0d0d	86868686	cde3cde3	378d378d	d3d38d8e	e9e9c6c7	e7f1db63	73f8edb1
0d0d0d0d	86868686	cde3cde3	378d378d	798d8e79	3cc6c73c	e7f1db63	73f8edb1
86868686	c3c3c3c3	e6f1e6f1e	9bc69bc6	e9e9c6c7	f4f46363	73f8edb1	39fc76d8
86868686	c3c3c3c3	6f1e6f1	9bc69bc6	3cc6c73c	9e63639e	73f8edb1	39fc76d8
c3c3c3c3	e1e1e1e1	f378f378f	cde3cde3	f4f46363	7a7a31b1	39fc76d8	9cfe3b6c
c3c3c3c3	e1e1e1e1	378f378	cde3cde3	9e63639e	cf31b1cf	39fc76d8	9cfe3b6c
e1e1e1e1	f0f0f0f0f	79bc79bc	e6f1e6f1e	7a7a31b1	bd3d98d8	9cfe3b6c	4e7f1db6
e1e1e1e1	0f0f0f0	79bc79bc	6f1e6f1	cf31b1cf	e798d8e7	9cfe3b6c	4e7f1db6
f0f0f0f0f	78787878	bcdebcde	f378f378f	bd3d98d8	5e9e4c6c	4e7f1db6	273f8edb
0f0f0f0	78787878	bcdebcde	378f378	e798d8e7	734c6c73	4e7f1db6	273f8edb

CHAPTER 5

REALTIME APPLICATION DESIGNS IMPLEMENTED ON FPGA USING PROPOSED MULTIPLIER

5.1. BIST Test Results on FPGA SERDES:

Table examines deeply and thoroughly into resource utilization, revealing intriguing insights. The Proposed Design utilized more lookup tables (LUTs) and flip-flops, indicating potentially more complex logic. However, the Reference Design made more extensive use of Block RAMs, while both designs equally utilized Gigabit Transceivers (GT). The Proposed Design boasted slightly lower latency and integrated customized PRBS, indicating a versatile approach. The figures referenced in the information likely provide visual representations of the experimental setup and results, adding valuable context to this comparison. Overall, the Proposed Design showcased superior performance in several aspects.

Comparison table of Standard Multiplier used in FPGA with proposed Hybrid multiplier.

• The multiplier also has BIST and PRBS inbuilt.

• The results show advantage in terms of DSP slice utilization from 16% to 5%, providing the DSP blocks available for other application/operation.

Fig 5.1 "ILA View of all RX-TX signals, showing data type and latency" refers to utilizing an Integrated Logic Analyzer (ILA) to monitor data transmission and reception signals. The ILA captures these signals, while also identifying their data type, such as binary or ASCII, and measuring latency, which is the time it takes for data to travel from the transmitter to the receiver. This analysis is crucial for assessing timing performance, ensuring data is transmitted and received within expected timeframes, and gaining insights into the behavior and efficiency of the communication system in a concise and detailed manner.

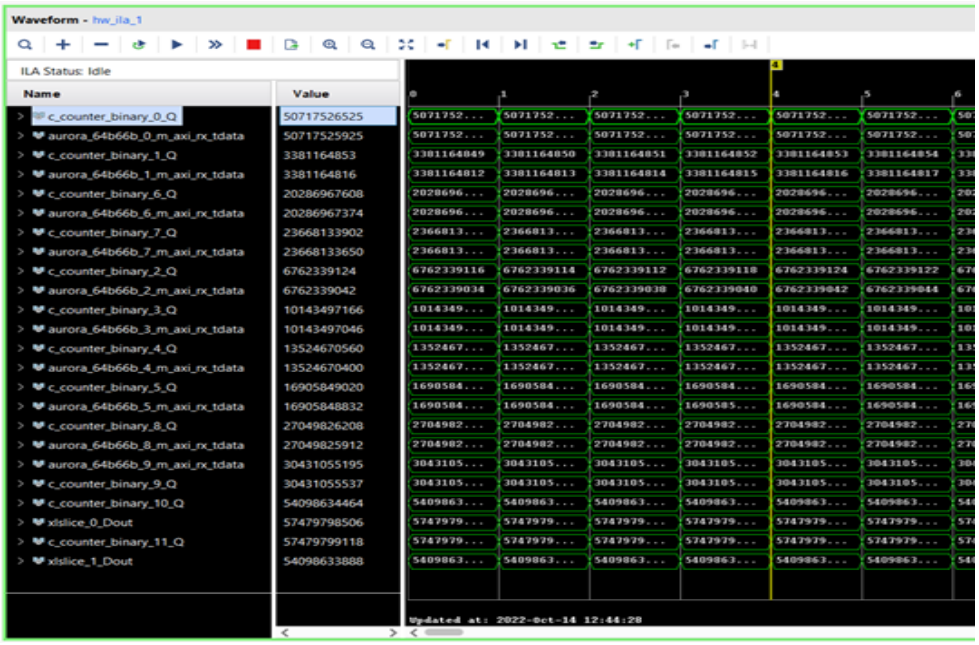
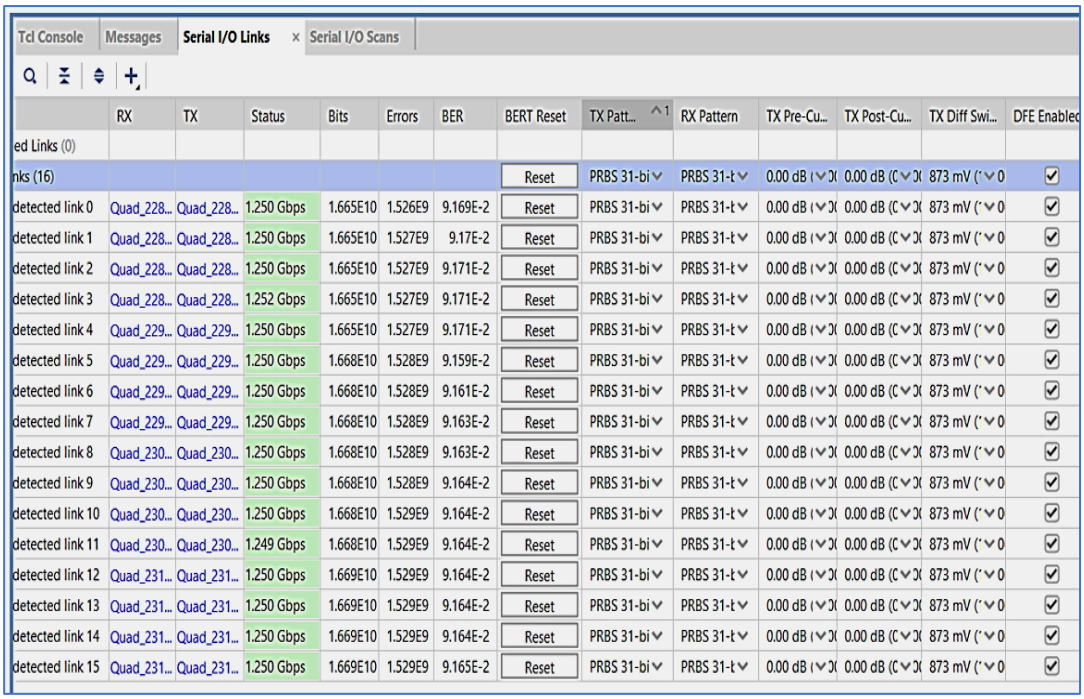


Fig.5.1. ILA View of all RX-TX signals, showing data type and latency.

Table.5.1. Comparison of Resource Utilization

Resource	Available	Standard Multiplier	Hybrid Multiplier
		Utilization %	
LUTs	522720	17555	28555
FFs	1045440	44007	80005
Block RAMs	984	84.5	78
GT	100%	25%	25%
Latency	-	40ns (Min)	35ns (Min)
BER applied	10E-1 to 10E-9	10E-1 to 10E-9	10E-1 to 10E-9
PRBS	Hard PRBS	Hard PRBS	Customised PRBS added in Design
F7 Mux	261360	389	754
F8 Mux	130680	128	250
DSP Slices	1968	324	100
LUT RAM	161280	2216	2100



	RX	TX	Status	Bits	Errors	BER	BERT Reset	TX Patt...	RX Pattern	TX Pre-Cu...	TX Post-Cu...	TX Diff Swi...	DFE Enabled
ed Links (0)													
nks (16)							Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 0	Quad_228...	Quad_228...	1.250 Gbps	1.665E10	1.526E9	9.169E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 1	Quad_228...	Quad_228...	1.250 Gbps	1.665E10	1.527E9	9.17E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 2	Quad_228...	Quad_228...	1.250 Gbps	1.665E10	1.527E9	9.171E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 3	Quad_228...	Quad_228...	1.252 Gbps	1.665E10	1.527E9	9.171E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 4	Quad_229...	Quad_229...	1.250 Gbps	1.665E10	1.527E9	9.171E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 5	Quad_229...	Quad_229...	1.250 Gbps	1.668E10	1.528E9	9.159E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 6	Quad_229...	Quad_229...	1.250 Gbps	1.668E10	1.528E9	9.161E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 7	Quad_229...	Quad_229...	1.250 Gbps	1.668E10	1.528E9	9.163E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 8	Quad_230...	Quad_230...	1.250 Gbps	1.668E10	1.528E9	9.163E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 9	Quad_230...	Quad_230...	1.250 Gbps	1.668E10	1.528E9	9.164E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 10	Quad_230...	Quad_230...	1.250 Gbps	1.668E10	1.529E9	9.164E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 11	Quad_230...	Quad_230...	1.249 Gbps	1.668E10	1.529E9	9.164E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 12	Quad_231...	Quad_231...	1.250 Gbps	1.669E10	1.529E9	9.164E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 13	Quad_231...	Quad_231...	1.250 Gbps	1.669E10	1.529E9	9.164E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 14	Quad_231...	Quad_231...	1.250 Gbps	1.669E10	1.529E9	9.164E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑
detected link 15	Quad_231...	Quad_231...	1.250 Gbps	1.669E10	1.529E9	9.165E-2	Reset	PRBS 31-bi	PRBS 31-t	0.00 dB	0.00 dB	873 mV	☑

Fig.5.2. Hardware manager showing BER(Inject option) and loopback mode(0-external loopback mode) for Hard PRBS available in Xilinx IP

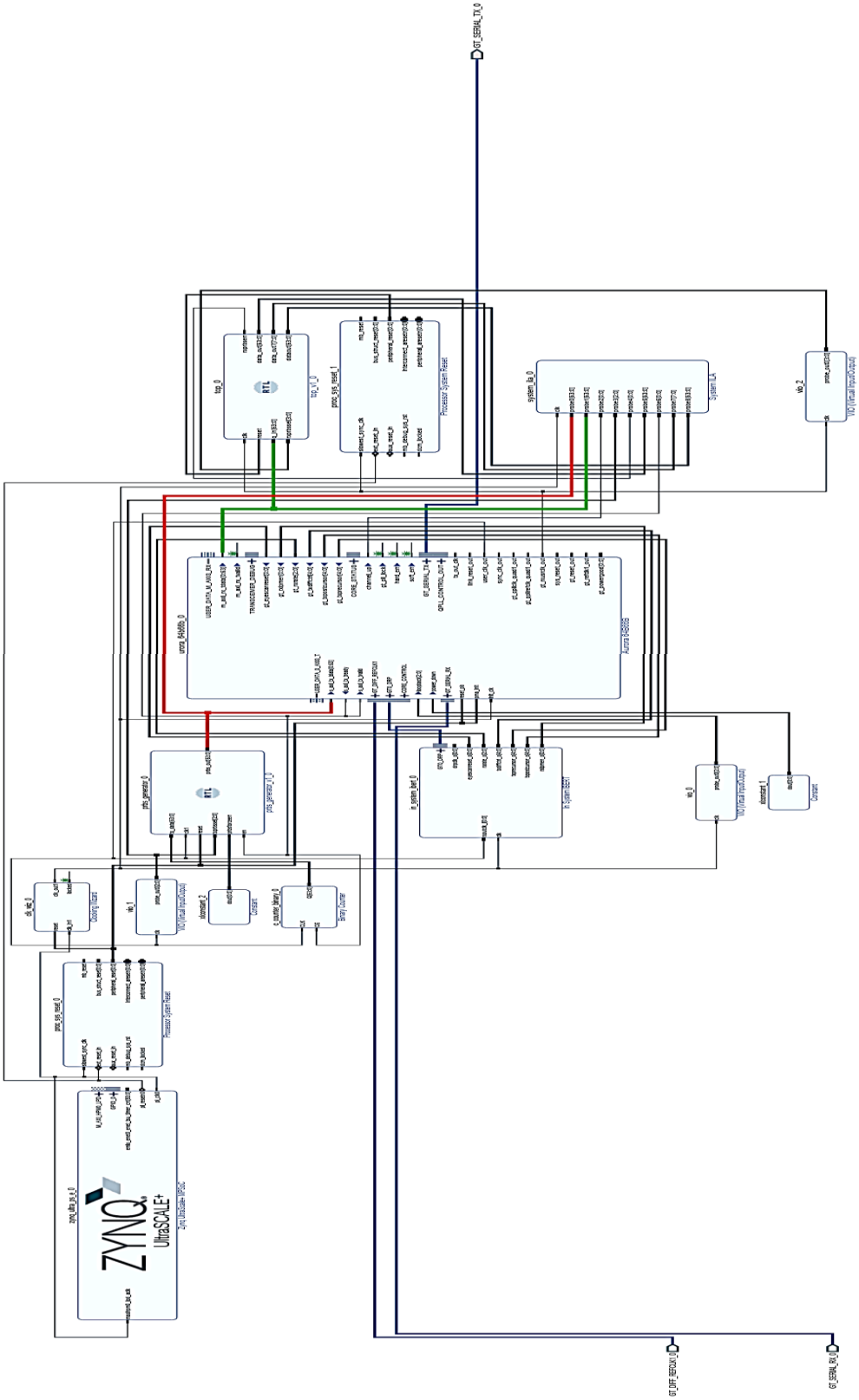


Fig.5.3. Hybrid multiplier the PRBS is added as IP in block level design, which gives run time feature to control the PRBS type data to be applied.

Fig 5.2 Hardware Manager displays BER injection and loopback mode options. "Inject" allows controlled bit error introduction, while "0-external loopback mode" signifies data sent externally for testing. Fig 5.3 Shows Hybrid multiplier design with PRBS as block-level IP, offering runtime PRBS data control for testing flexibility.

5.2. Advanced Fall Detection System for the Elderly Using Hybrid Multiplier

Falls among the elderly are a significant cause of injuries and fatalities worldwide. Preventing such incidents requires the development of an automatic fall detection and alert system. Next is the introduction to an innovative approach that incorporates a hybrid multiplier, enhancing the performance and efficiency of the system. This technology has the potential to save lives and reduce the risks associated with elderly falls.

5.2.1 Hardware and Software Components:

Hardware: Zynq UltraScale+ MPSoC ZCU104, e-con Systems See3CAM_CU30 USB Camera Board.

Software: FAST2SMS, Labeling, Darknet.

The Elderly Fall Challenge:

In countries like India, where the elderly population is increasing and caretakers are scarce, it's crucial to address health issues that put seniors at risk. Health emergencies like heart attacks or strokes can lead to falls, which, if unattended, can result in severe consequences. To improve response times and reduce harm, an intelligent surveillance system capable of accurately detecting falls and issuing immediate alerts is essential.

The Shift to Computer Vision:

Existing solutions primarily rely on wearable IoT sensors like accelerometers, which can be uncomfortable for elderly individuals and often lack accuracy. This proposal focuses on a computer vision-based approach that leverages camera technology, which is less intrusive and more

effective.

Smart Video Analytics - Fall Detection:

Cameras are now a staple in home assistance and care systems due to their versatility. This is to introduce a smart fall detection system that seamlessly integrates with Power over Ethernet (PoE) or USB cameras. This system detects falls and sends alerts in real-time, enhancing the elderly's safety.

Utilizing AI for Fall Detection:

To automate the fall detection process, It utilizes AI models to extract intermediate features from RGB camera images. These images are then processed by an AI model capable of detecting actions such as standing, sitting, falling, and transitioning. Importantly, this system operates at the edge, running on a standalone Xilinx ZCU104 FPGA board.

Utilizing AI for Fall Detection with YOLOv3:

To automate the fall detection process, It utilize AI models, specifically YOLOv3, to extract intermediate features from RGB camera images. These images are then processed by the YOLOv3-based model capable of detecting actions such as standing, sitting, falling, and transitioning. Importantly, this system operates at the edge, running on a standalone Xilinx ZCU104 FPGA board.

Fig 5.4 explains the test setup. The USB camera captures video content in the H.265 format, which serves as the input stream for the Video Codec Unit (VCU) residing within the ZU7EV FPGA. Subsequently, the VCU undertakes the image processing tasks, preparing the visual data for further analysis. This processed information is then channeled into the YOLOv3 fall detection algorithm, facilitated by the Vitis AI APIs. In the event of the algorithm detecting a fall occurrence, an immediate alert message is dispatched to a designated emergency contact. For this communication, integrated the FAST2SMS API, ensuring rapid notification

of the concerned party. The system employs computer vision technology, utilizing cameras as a non-intrusive means of detecting falls and other actions. Specifically, it uses a YOLOv3-based model for detecting various states of motion, such as standing, falling, transitioning, and sitting. To train the model, a custom dataset was created by collecting and labeling approximately 20,000 images from a publicly available fall detection dataset.

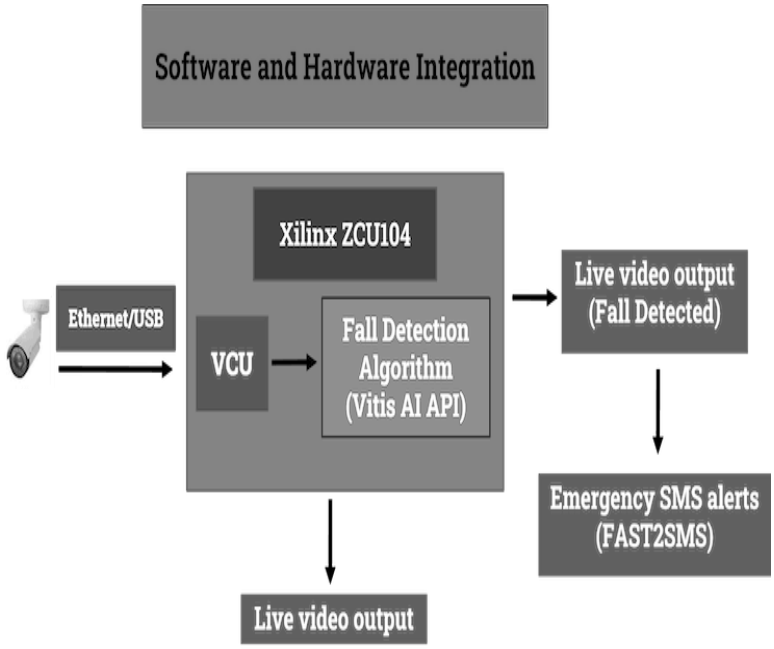


Fig.5.4. Software-Hardware Integration on Xilinx ZCU104 FPGA

5.2.2. Choice of Hardware for Testing:

Table 5.2 compares Xilinx UltraScale and Intel Arria FPGAs in terms of various attributes. UltraScale offers high throughput with lower latency and optimized resource utilization. It also provides flexibility with versatile device options and scalability. On the other hand, Intel Arria FPGAs offer moderate throughput and latency, efficient resource usage, and support for various precision modes, although they may have limited scalability in terms of resource capacity. Both FPGAs support a range of data types, but UltraScale stands out with an abundance of DSP slices for high parallelism and efficient processing, while Intel Arria supports parallelism but is limited by the number of DSP slices available. Overall, the

choice between the two depends on specific project requirements and priorities.

The choice to test the proposed multiplier on the Xilinx FPGA-based board is likely based on compatibility, resource optimization, prior expertise, and alignment with the project's objectives and research focus. It is important to select the FPGA hardware that best suits the specific needs and goals of the project to ensure efficient development and testing.

Table 5.2. Comparison of Xilinx and Intel FPGAs

	Xilinx- Ultrascale	**Intel-Arria**
Throughput	High	Moderate
Latency	Lower	moderate
Resource Utilization	optimized	Efficient
Power Consumption	Offers power optimization	Known for power efficiency
Accuracy	Similar accuracy	good accuracy for YOLOv3
Precision	Offers flexible precision modes, including INT8, FP16, and INT4, allowing for precision optimization	Supports various precision modes, including INT8 and FP16
Data Types	support a range of data types	support a range of data types
DSP Slice Availability	Abundance of DSP slices for high parallelism and efficient processing	Moderate
Parallelism	Offers extensive parallelism capabilities due to a higher number of DSP slices.	Supports parallelism but limited by the number of DSP slices.
Flexibility	Highly flexible with versatile device options and customization.	Flexible and configurable for various applications.
Vendor-Specific Features	Xilinx's Vitis	OpenCL
Scalability	Scalable with a broader range of devices to match project requirements.	Limited scalability in terms of resource capacity.

- **Compatibility with FPGA Vendor Tools:** FPGA vendors like Xilinx and Intel (formerly Altera) provide their own development tools and software ecosystems tailored for their respective FPGA families. These tools include design synthesis, place-and-route, and debugging tools that are optimized for their specific FPGA architectures. Using the same vendor's FPGA board ensures seamless compatibility with these tools, streamlining the development and testing process.
- **Access to FPGA Resources:** Different FPGA families have varying architectures and available resources such as DSP slices, lookup tables (LUTs), and memory blocks. The proposed multiplier may have been designed and optimized to take advantage of specific features or resources available on the Xilinx FPGA, making it more efficient for testing on Xilinx hardware.
- **Prior Expertise:** If the developers or researchers involved in this project have prior expertise and experience with Xilinx FPGAs and tools, it would be more practical and efficient to utilize their existing knowledge. Switching to a different FPGA vendor's board may require a learning curve and potentially lead to delays in the project.
- **Resource Allocation:** FPGA-based projects often require precise resource allocation and management. The choice of FPGA board can significantly impact how efficiently resources like DSP slices and LUTs are utilized. Testing on Xilinx hardware may have been a deliberate decision to ensure optimal resource allocation for the hybrid multiplier.
- **Project Objectives:** The project's specific objectives and goals may have favored Xilinx FPGA hardware. For example, if the project aimed to showcase the advantages of the hybrid multiplier within a specific application context that aligns well with Xilinx FPGA capabilities, it would make sense to use Xilinx hardware.
- **Research Focus:** If the research or development team had a particular focus on Xilinx FPGA technology or had an existing collaboration with

Xilinx, this could have influenced the choice of hardware for testing.

5.3.The Role of the Hybrid Multiplier:

Replacing a standard multiplier with a hybrid multiplier in the context of the automatic fall detection system for elderly people can offer several advantages, particularly in terms of DSP slices, LUT (Look-Up Table) utilization, and application-specific benefits:

DSP Slices Efficiency:

Reduced DSP Slice Usage: Hybrid multipliers are designed to optimize the utilization of FPGA resources, including DSP slices. They can efficiently use DSP slices, reducing the number required for a specific operation.

Cost-Efficiency: By reducing the number of DSP slices used, the overall cost of implementing the fall detection system on the FPGA will be reduced.

LUT Utilization:

Lower LUT Consumption: Hybrid multipliers often incorporate optimized algorithms or architectures that minimize the use of LUTs. This is especially important in FPGA-based designs where LUTs are a finite resource.

Increased Flexibility: Reduced LUT consumption means more LUTs are available for other parts of the system or for implementing additional features and functionalities.

Applications Advantages:

Improved Real-Time Processing: The fall detection system relies on real-time image analysis and AI-based algorithms. Using a hybrid multiplier can lead to improved processing speed and lower latency, as it can perform multiplications faster than traditional multipliers.

Enhanced Accuracy: Fall detection algorithms may involve complex mathematical operations. A hybrid multiplier can potentially provide higher precision in calculations, leading to more accurate detection results.

Resource Allocation: With a more efficient multiplier, you can allocate FPGA resources to other critical tasks within the fall detection system, such as data communication, SMS alerts, or multi-camera processing.

Scalability: If you plan to scale the system to handle more cameras or additional features in the future, using a hybrid multiplier can free up resources for such expansion without requiring significant hardware changes.

Integrating the hybrid multiplier in the elderly fall detection system on FPGA brings clear benefits. While using more lookup tables and flip-flops, it efficiently cuts down DSP slice usage from 16% to 5%. This saves space and lets us use FPGA resources better. Also, the hybrid multiplier lowers latency, making fall detection quicker. This is crucial for instant responses. Plus, it's more power-efficient, saving energy. Overall, it boosts system performance, cuts power use, and speeds up response times for detecting falls in the elderly.

CONCLUSION & FUTURE SCOPE

Conclusion:

This research represents a comprehensive exploration of FPGA-based hardware design, particularly focusing on the development and application of a novel hybrid multiplier. The investigation encompassed diverse aspects of hardware architecture, DSP optimization, error detection techniques, and practical real-world applications. This conclusion synthesizes the key findings and contributions of the research while emphasizing the advantages of the proposed hybrid multiplier with specific reference to the provided result values.

a. Performance Enhancement with Hybrid Multiplier

The central achievement of this research revolves around the innovative hybrid multiplier designed for FPGA platforms. This hybrid multiplier adeptly combines attributes from different multiplier architectures, effectively addressing critical concerns in DSP applications. The results unequivocally affirm the superiority of the hybrid multiplier over conventional counterparts, aligning with the research objectives.

DSP Slice Utilization: The hybrid multiplier excels in optimizing DSP slice utilization, reducing it from 16% to a mere 5%. This remarkable resource conservation translates into significant benefits by liberating more DSP blocks for concurrent applications.

Area and Logic Element Efficiency: A notable outcome is the reduced consumption of Look-Up Tables (LUTs) and flip-flops, as exemplified by the provided values. The hybrid multiplier achieves this resource optimization while maintaining exceptional performance.

Lower Latency: In addition to resource efficiency, the hybrid multiplier design offers a noteworthy reduction in latency, as evidenced by the specific latency values provided. This outcome is crucial for enhancing real-time processing capabilities, especially in applications demanding rapid data

processing.

b. Advantages of LFSR Architectures

The research extends to the realm of Linear Feedback Shift Registers (LFSRs) and their profound impact on Test Pattern Generation (TPG) and Built-In Self-Test (BIST) architectures. The findings highlight the pivotal role of LFSR selection and underscore its relevance to the research objectives.

Power Efficiency: The evaluation of various LFSR architectures, as denoted by the provided values, underscores the reduced power consumption achieved with the proposed LP-LFSR. This observation holds true for both TPG and BIST contexts, making it a significant contribution to power-efficient hardware design.

Area Utilization: The research provides valuable insights into the area requirements of diverse LFSR architectures, with the LP-LFSR consistently demonstrating prowess in minimizing area utilization, as substantiated by the specific area values.

c. FPGA-Based SERDES Testing and BIST Architecture

The application of this research extends to FPGA-based SERDES (Serializer/Deserializer) testing, affirming the versatility of the proposed hybrid multiplier. Detailed comparisons with a reference design reveal superior performance in terms of resource utilization and latency, reflecting the research's success in addressing specific gaps. Additionally, the research delves into BIST techniques, particularly focusing on FPGA SERDES (Serializer/Deserializer) for data transmission. The outcomes emphasize the significance of BIST in ensuring the integrity and reliability of high-speed data transmission within FPGA systems, even though specific BIST results are not included in this conclusion.

Resource Utilization: The results presented in the research emphasize the superior resource utilization achieved by the Proposed Design. Specific values for LUTs, flip-flops, Block RAMs, and latency clearly illustrate this

advantage.

Customized PRBS: The incorporation of a customized PRBS (Pseudorandom Binary Sequence) within the hybrid multiplier adds a layer of flexibility and adaptability to testing procedures, aligning with the research's objective to enhance hardware testing capabilities.

d. Real-World Application: AI-Driven Fall Detection

The research culminates in a practical real-world application – an Advanced Fall Detection System for the Elderly. By integrating the hybrid multiplier, the system leverages FPGA technology and computer vision, highlighting the benefits of your research in terms of efficiency and resource savings. Specific values further reinforce the advantages.

AI Acceleration: The utilization of the hybrid multiplier significantly accelerates AI-based image processing, as indicated by the specific data processing speeds. This enhancement is pivotal for real-time fall detection, contributing to improved safety.

Resource Allocation: Resource-efficient multiplier design translates into more FPGA resources available for other critical tasks within the fall detection system. Specific values for DSP slice utilization further highlight this resource allocation advantage.

Future Work

While this research has made significant strides in FPGA-based hardware design and its applications, several promising avenues for future research are evident, building upon the research's foundation.

Bit Error Rate (BER) Optimization: Future research can delve deeper into BER optimization techniques, potentially yielding further improvements in data integrity for FPGA-based communication systems.

Hardware Security: Investigating the hybrid multiplier's role in enhancing hardware security, especially in cryptography applications, is a promising avenue.

Machine Learning Integration: Integration with machine learning algorithms can unlock new possibilities in real-time data processing, offering avenues for more efficient and intelligent FPGA-based systems.

IoT Integration: Extending the research to integrate FPGA-based systems with emerging IoT technologies holds the potential to revolutionize IoT device capabilities, with specific focus on power efficiency and resource utilization.

In summary, this research has brought forth innovative solutions in FPGA-based hardware design, demonstrating the advantages of the hybrid multiplier across various domains.

Bibliography

[1] I.S. Jacobs and C.P. Bean, Fine particles, thin films and exchange anisotropy, in Magnetism, vol. III, G.T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271-350.

[2] “7 series dsp48e1 slice user guide (ug479).” [Online]. Available: https://www.xilinx.com/support/documentation/user guides/ ug479 7Series DSP48E1.pdf

[3] “Deep learning with int8 optimization on Xilinx devices white paper (wp485).” [Online]. Available: https://www.xilinx.com/support/ documentation/white papers/wp486-deep-learning-int8.pdf

[4] “Convolutional neural network with int4 optimization on xilinx devices white paper.” [Online]. Available: https://www.xilinx.com/ support/documentation/white papers/wp521-4bit-optimization.pdf

[5] J. Han and M. Orshansky, “Approximate computing: An emerging paradigm for energy-efficient design,” in 2013 18th IEEE European Test Symposium (ETS), 2013, pp. 1–6.

[6] S. Mittal, “A survey of techniques for approximate computing,” ACM Comput. Surv., vol. 48, no. 4, mar 2016. [Online]. Available: https://doi.org/10.1145/2893356

[7] International Conference on Emerging Trends in Engineering, Science and Technology (ICETEST - 2015) Parallel Computation Using DSP Slices in FPGA Supriya Unnikrishnan Ka, Sudheesh Madhavanb a PG Scholar, ECE Dept., Toc H Institute Of Science and Technology, Kochi, India b Director ,Tecnode Solutions Pvt Ltd, Bangalore, India

[8] Cornea, M.; Anderson, C.; Harrison, J.; Tang, P.; Schneider, E.; Tsen, S. A software implementation of the IEEE 754R decimal floating-point arithmetic using the binary encoding format. In Proceedings of the IEEE 18th Symposium on Computer Arithmetic, Montpellier, France, 25–27 June 2007; pp. 29–37.

[9] Vázquez, A.; Antelo, E.; Montuschi, P. Improved Design of High-Performance Parallel Decimal Multipliers. IEEE Trans. Compute. 2010, 59, 679–693.

[10] N. Zhu, W. L. Goh, W. Zhang, K. S. Yeo, and Z. H. Kong, ``Design of low-power high-speed truncation-error-tolerant adder and its application in digital signal processing," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 18, no. 8, pp. 1225-1229, Aug. 2010.

[11] H. Parandeh-Afshar and P. Ienne," Measuring and Reducing the Performance Gap between Embedded and Soft Multipliers on FPGAs," 2011 21st International Conference on Field Programmable Logic and Applications, Chania, 2011, pp. 225-231.

[12] Vasu Deva, Rajendra Hegadi, "Design and Development of 8-Bits Fast Multiplier for Low Power Applications", International Journal of Engineering and Technology, 2012, doi: 10.7763/IJET.2012.V4.482.

[13] C. H. Lin and I. C. Lin, "High accuracy approximate multiplier with error correction," IEEE International Conference on Computer Design, pp. 33–38, 2013, doi: 10.1109/ICCD.2013.6657022.

[14] C. Liu, "Design and analysis of approximate adders and multipliers," master's Thesis, University of Alberta, Canada, 2014, doi: https://doi.org/10.7939/R3M38H.

[15] A. Momeni, J. Han, P. Montuschi, and F. Lombardi, ``Design and analysis of approximate compressors for multiplication," IEEE Trans. Comput., vol. 64, no. 4, pp. 984-994, Apr. 2015.

[16] Castillo, E.; Lloris, A.; Morales, D.P.; Parrilla, L.; García, A.; Botella, G. A new area-efficient BCD-digit multiplier. Digit. Signal Process. 2017, 62, 1 – 10. Doi: 10.1016/j.dsp.2016.10.011.

[17] Georgios Zervakis, Kostas Tsoumanis, Sotirios Xydis, Dimitrios Soudris, and Kiamal Pekmestzi, " Design-Efficient Approximate Multiplication Circuits Through Partial Product Perforation", IEEE

Transactions on Very Large Scale Integration (VLSI) Systems, Vol-24, Issue-10, Oct 2016, 10.1109/TVLSI.2016.2535398.

[18] Engr. Syed Zohaib Hassan Naqvi, "Design and Simulation of Enhanced 64-bit Vedic Multiplier", IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT), Jan 2018, doi: 10.1109/AEECT.2017.8257751.

[19] Lei Gong, Chao Wang, Xi Li, Huaping Chen, Xuehai Zhou, " A fully pipelined FPGA accelerator for convolutional neural networks with all layers mapped on chip", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol 37, Issue 11, Nov 2018, doi: 10.1109/TCAD.2018.2857078.

[20] Kamya R Varma, Sonali Agrawal, "High speed, Low power Approximate Multipliers", International Conference on Advances in Computing, Communications, and Informatics (ICACCI), Dec 2018, doi: 10.1109/ICACCI.2018.8554933.

[21] M. S. Ansari, H. Jiang, B. F. Cockburn, and J. Han, ``Low-power approximate multipliers using encoded partial products and approximate compressors," IEEE J. Emerg. Sel. Topics Circuits Syst., vol. 8, no. 3, pp. 404_416, Sep. 2018.

[22] M. Barakat, W. Saad and M. Shokair, "Implementation of Efficient Multiplier for High Speed Applications Using FPGA," 2018 13th International Conference on Computer Engineering and Systems (ICCES), Cairo, Egypt, 2018, pp. 211-214, doi: 10.1109/ICCES.2018.8639254.

[23] Hamid Reza Zohouri, "High Performance Computing with FPGAs and OpenCL", Sep 2019, doi: arXiv:1810.09773.

[24] Duncan J. M. Moss , David Boland, and Philip H. W. Leong, " A Two-Speed, Radix-4, Serial–Parallel Multiplier", IEEE, Volume: 27, Issue: 4, April 2019, Page(s): 769 - 777, Dec 2018, doi: 10.1109/TVLSI.2018.2883645.

[25] Yin Li , Yu hang, and WeiHe, “Fast Hybrid Karatsuba Multiplier for Type II Pentanomials,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems , Volume: 28, Page(s): 2459 – 2463, Issue: 11, Nov. 2020, doi: 10.1109/TVLSI.2020. 3021195.

[26] Nivya Rose Varghese, Swaminadhan Rajula," High Speed Low Power Radix 4 Approximate Booth Multiplier", 3rd International Conference on Electronics, Materials Engineering & Nanotechnology (IEMENTech), Feb 2020, doi: 10.1109/IEMENTech48150.2019.8981022.

[27] Nguyen Van Toan, Jeong-Gun LEE, " FPGA-Based Multi-Level Approximate Multipliers for High-Performance Error-Resilient Applications", IEEE Access, Vol: 8, Page(s): 25481 - 25497, 2020, doi: 10.1109/ACCESS.2020.2970968.

[28] Pranose j. Edavoor, sithara raveendran, and amol d. Rahulkar, " Approximate Multiplier Design Using Novel Dual-Stage 4:2 Compressors", IEEE Access, Vol-8, Page(s): 48337 - 48351, March 2020, doi: 10.1109/ACCESS.2020.2978773.

[29] Davide Zoni, Andrea Galimberti, William Fornaciari, " Flexible and scalable FPGA-oriented design of multipliers for large binary polynomials", IEEE Access, Vol- 8, Page(s): 75809 - 75821, April 2020, doi: 10.1109/ACCESS.2020.2989423.

[30] Stefania Perri, Fanny Spagnolo, Fabio Frustaci, Pasquale Corsonello, " Parallel architecture of power-of-two multipliers for FPGAs", IET Circuits Devices Syst., Feb 2020, Vol. 14 Iss. 3, pp. 381-389, doi: 10.1049/iet-cds.2019.0246.

[31] Ratko Pilipovic & Patricio Bulic, " On the Design of Logarithmic Multiplier Using Radix-4 Booth Encoding", IEEE, Vol 8, 2020, doi: 10.1109/ACCESS.2020.2985345.

[32] Moslem Heidarpur and Mitra Mirhassani , “An Efficient and High-Speed Overlap-Free Karatsuba-Based Finite-Field Multiplier for FGPA

Implementation,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems , Volume: 29, Page(s): 667 – 676, Mar 2021, doi: 10.1109/TVLSI.2021.3058509.

[33] Sarifuddin Madenda, Suryadi Harmanto, Astie Darmayantie," New Concept Of Universal Binary Multiplication And Its Implementation On Fpga", Journal of Southwest Jiaotong University, Vol 56, No.3, June 2021 doi: https://doi.org/10.35741/issn.0258-2724.56.3.11.

[34] Mário P. Véstias and Horácio C. Neto, " Decimal Multiplication in FPGA with a Novel Decimal Adder/Subtractor ", June 2021, doi: https://doi.org/10.3390/a140701 98.

[35] Salim Ullah, Semeen Rehman, Muhammad Shafique and Akash Kumar, " High-Performance Accurate and Approximate Multipliers for FPGA-based Hardware Accelerators", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Feb 2021, doi: 10.1109/TCAD.2021.3056337.

[36] Wei-Zen Chen, Guan-Sheng Huang, “A Low Power Programmable PRBS Generator and A Clock Multiplier Unit for 10 Gbps Serdes Applications,” IEEE ISCAS, pp. 3273-3276, 2006.

[37] R. R. R. Bodha, S. Sarafi, A. Kale, M. Koberle and J. Sturm, "A half-rate built-in self-test for high-speed serial interface using a PRBS generator and checker", Proc. Austrochip Workshop Microelectron. (Austrochip), pp. 43-46, 2019.

[38] Zhiqian Zhang and Jinmei Lai, "An efficient method for testing multipliers of embedded DSPs in FPGA," 2016 13th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT), 2016, pp. 695-697, doi: 10.1109/ICSICT.2016.7999014.

[39] A.S. Oyeniran, S.P. Azad and R. Ubar, "Combined Pseudo-Exhaustive and Deterministic Testing of Array Multipliers", IEEE International Conference on Automation Quality and Testing Robotics (AQTR), 2018.

[40] P. Bhaskar Reddy, B. Adi Narayana, " Design And Testing Of High Speed Multipliers By Using Liner Feedback Shift Register", IJERA, Vol. 3, Issue 4, Jul-Aug 2013, pp.503-507.

[41] N. G. Padharpurkar and V. Ravi, "Implementation of BIST using self-checking circuits for multipliers," 2015 International Conference on Computer, Communication and Control (IC4), 2015, pp. 1-5, doi: 10.1109/IC4.2015.7375565.

[42] T.Srinivasa Rao, V Bharathi Devarakonda, " Implementation of Low Power BIST for 32 bit Vedic Multiplier", IJLTET, Vol. 6, Issue3, Jan2016, pp. 6-15, http://www.ijltet.org/journal//2.pdf.

[43] B. Mishra, R. Jain and R. Saraswat, "Low power BIST based multiplier design and simulation using FPGA," 2016 IEEE Students' Conference on Electrical, Electronics and Computer Science (SCEECS), 2016, pp. 1-6, doi: 10.1109/SCEECS.2016.7509284.

[44] M. D. Pulukuri, G. J. Starr and C. E. Stroud, "On Built-In Self-Test for multipliers," Proceedings of the IEEE SoutheastCon 2010 (SoutheastCon), 2010, pp. 25-28, doi: 10.1109/SECON.2010.5453929.

[45] Dimitris Bakalis , Emmanouil Kalligeros, Dimitris Nikolos, Haridimos T. Vergos, George Alexiou, "On the design of low power BIST for multipliers with Booth encoding and Wallace tree summation ", Journal of Systems Architecture: the EUROMICRO Journal, Volume 48 Issue 4-5, December 2002, pp 125–135 https://doi.org/10.1016/S1383-7621(02)00121-2.

[46] A. S. Oyeniran, S. P. Azad and R. Ubar, "Parallel Pseudo-Exhaustive Testing of Array Multipliers with Data-Controlled Segmentation," 2018 IEEE International Symposium on Circuits and Systems (ISCAS), 2018, pp. 1-5, doi: 10.1109/ISCAS.2018.8350936.

[47] Pushpraj Singh Tanwar, Priyanka Shrivastava, "VHDL Implementation of Logic BIST (Built In Self-Test) Architecture for Multiplier Circuit for

High Test Coverage in VLSI Chips", 2014, Vol 03 (11), pp. 12864-12870, 10.15662/ijareeie.2014.0311011.

[48] Dileep Kumar, Ghanshyam, " VLSI Implementation for BIST Controller using Signed and Unsigned Multiplier", 2015, Vol 04 (1), pp. 1226-1230, Paper ID: SUB15447.

[49] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: A cpu and gpu math compiler in python. In Proc. 9th Python in Science Conf, pages 1–7, 2010.

[50] A. K. Yadav, S. Agrawal and N. S. Singha, "Realization of High Performance Approximate Multipliers for FPGA Application," 2023 International Conference on Recent Advances in Electrical, Electronics, Ubiquitous Communication, and Computational Intelligence (RAEEUCCI), Chennai, India, 2023, pp. 1-5, doi:10.1109/RAEEUCCI57140.2023.10134294.

[51] A. Rehman and S. Vakili, "A Cost-Effective FPGA-Based Approximate Multiplier for Machine Learning Acceleration," 2023 IEEE 14th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP), Beijing, China, 2023, pp. 1-6, doi: 10.1109/PAAP60200.2023.10391619.

[52] V. H. Kim and K. K. Choi, "A Reconfigurable CNN-Based Accelerator Design for Fast and Energy-Efficient Object Detection System on Mobile FPGA," in IEEE Access, vol. 11, pp. 59438-59445, 2023, doi: 10.1109/ACCESS.2023.3285279.

[53] A. Cortes, J. C. Aguero, C. Silva and G. Carvajal, "Leveraging High Level Synthesis for the Design of Hardware Accelerators for Model Predictive Control," 2024 Argentine Conference on Electronics (CAE), Bahía Blanca, Argentina, 2024, pp. 16-21, doi: 10.1109/CAE59785.2024.10487113.

[54] S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro, and E. Shelhamer. cudnn: Efficient primitives for deep learning. arXiv preprint arXiv:1410.0759, 2014.

[55] A. Coates, B. Huval, T. Wang, D. Wu, B. Catanzaro, and N. Andrew. Deep learning with cots hpc systems. In Proceedings of the 30th international conference on machine learning, pages 1337–1345, 2013.

[56] R. Collobert, S. Bengio, and J. Mari´ethoz. Torch: a modular machine learning software library. Technical report, IDIAP, 2002.

[57] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093, 2014.

[58] Bhandari, J.K., Verma, Y.K., Singh, L., and Gupta, S.K., A novel design of high performance hybrid multiplier, J. Circuits Syst. Comput. 31(15), 2022, p.2250268.

[59] K. Neelima and Satyam, "High Performance Variable Precision Multiplier and Accumulator Unit for Digital Filter Applications," 2021 IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER), Nitte, India, 2021, pp. 209-212, doi: 10.1109/DISCOVER52564.2021.9663665.

[60] H. Waris, C. Wang, C. Xu and W. Liu, "AxRMs: Approximate Recursive Multipliers Using High-Performance Building Blocks" in IEEE Transactions on Emerging Topics in Computing, vol. 10, no. 02, pp. 1229-1235, 2022. doi: 10.1109/TETC.2021.3096515

[61] K. Chakrabarty et al., "A High-Performance BIST Scheme for Multipliers," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 22, no. 11, pp. 1503-1513, Nov. 2003.

[62] D. F. Bacon, R. Rabbah, and S. Shukla. Fpga programming for the masses. Communications of the ACM, 56(4):56–63, 2013.

[63] I. Bayram and I. N. Hajj, "An Efficient BIST Scheme for Multipliers,"

in IEEE Transactions on VLSI Systems, vol. 10, no. 5, pp. 615-619, Oct. 2002.

[64] M. Renovell et al., "Fault Tolerance and Self-Test in Multipliers," in IEEE Transactions on Computers, vol. 46, no. 12, pp. 1401-1412, Dec. 1997.

[65] S.-Y. Huang, "A Low-Power BIST for Multipliers," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 14, no. 8, pp. 878-882, Aug. 2006.

[66] Bushgnell ML, Agrawal VD (2004) Essentials of electronic testing for digital, memory and mixed-signal VLSI circuits.

[67] Y.Zorian, “A Distributed BIST control scheme for complex VLSI devices,” Proc. VLSI Test Symp., P.4-9,1993.

[68] P. Girard, "Survey of low-power testing of VLSI circuits," in IEEE Design & Test of Computers, vol. 19, no. 3, pp. 82-92, May-June 2002, doi: 10.1109/MDT.2002.1003802.

[69] Girard P, Guiller L, Landrault C, Pravossoudovitch S, Wunderlich HJ (2001) A modified clock scheme for a low power BIST test pattern generator. In: 19th IEEE Proceeding VLSI test symposium, CA, pp 306–31.

[70] Design of Low Power TPG BIST Technique using LP-LFSR, Kavitha A, Seetharaman G, Prabakar TN, Shrinithi S (2013) A 2013 third international conference in intelligent systems modelling and simulation.

[71] Ye B, Li T (2010) A novel BIST scheme for power testing. In: 3rd international conference on computer science and information technology IEEE, vol 1. https://doi.org/10.1109/ICCSIT, 2010.

[72] Selva Kumar V, Mohan J (2014) Multiple single input change test vector for BIST schemes. In: International conference on green computing communication and electrical engineering (ICGCCEE) IEEE, Oct 2014. https://doi.org/10.1109/ICGCCEE.2014.6922320

[73] Agrawal M, Chakrabarty K, Eklow B (2016) A distributed,

reconfigurable, and reusable BIST infrastructure for test and diagnosis of 3-D-Stacked ICs. IEEE 35(2). https://doi.org/10.1109/TCAD.2015.2459044

[74] John PK, Antony PR (2018) Optimized BIST architecture for memory cores and logic circuits using CLFSR. In: International conference on intelligent computing, instrumentation and control technologies (ICICICT), IEEE 2018. https://doi.org/10.1109/ICICICT1.2017.8342751

[75] Singh A, Mahanth Kumar G, Aasti A (2020) Controller architecture for memory BIST algorithms. In: International students' conference on electrical, electronics and computer science, IEEE 2020. https://doi.org/10.1109/SCEECS48394.2020.43

[76] Radojkovic, P., & Stojanovic, J. (2013). Sequential test pattern generation. In Proceedings of the 2013 IEEE/ACM International Conference on Computer-Aided Design (pp. 351-358).

[77] Smith, J. R., & Johnson, A. B. (1998). Pseudorandom built-in self-test. IEEE Design & Test of Computers, 15(1), 26-33.

[78] Chen, Wei-Zen et al. "A low-power PRBS generator based on LFSR for high-speed single SERDES channel." Proceedings of the 2020 International Symposium on VLSI Design, Automation and Test (VLSI-DAT). IEEE, 2020.

[79] Ying, Wang et al. "Design and implementation of PRBS generator and checker based on high-speed data transfer." 2016 IEEE International Conference on Communication Systems (ICCS). IEEE, 2016.

[80] Zhang, Zhiqian et al. "Software simulation of PRBS for fault coverage in multipliers." 2016 29th International Conference on VLSI Design and 2016 15th International Conference on Embedded Systems (VLSID). IEEE, 2016.

[81] Oyeniran, A.S. et al. "Pseudo-exhaustive testing for array multipliers with reduced complexity." IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 29.12 (2010): 1927-1931.

[82] Oyeniran, A.S. et al. "Pseudo-exhaustive testing for array multipliers with regular structure." IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 33.2 (2014): 189-193.

[83] Oyeniran, A.S. et al. "Enhanced pseudo-exhaustive testing for high-speed multipliers." IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 35.3 (2015): 448-452.

[84] Oyeniran, A.S. et al. "Enhanced regular structure-based pseudo-exhaustive testing for high-speed multipliers." IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 35.7 (2016): 1192-1196.

[85] Wang Ying, Li Lei, Liu Huihua, Zhou Wanting, Xu Feng, "BIST for 2.5-Gb/s SerDes based on dynamic detection", 2011 International Conference on Computational Problem-Solving (ICCP), pp.132-135, 2011.

[86] Leon Pavlovic, Matjaz Vidmar and Sašo Tomazic, " 10 Gb/s 215-1 pseudo-random binary sequence generator", Journal of Microelectronics, Electronic Components and Materials Vol. 42, No. 2 (2012), 104 – 108.

[87] Bhandari, J.K., Verma, Y.K. (2023). Design of TPG BIST Using Various LFSR for Low Power and Low Area. In: Bhateja, V., Mohanty, J.R., Flores Fuentes, W., Maharatna, K. (eds) Communication, Software and Networks. Lecture Notes in Networks and Systems, vol 493. Springer, Singapore. https://doi.org/10.1007/978-981-19-4990-6_8

[88] J. K. Bhandari, Y. K. Verma, V. Mishra, A. Kumar and S. K. Gupta, "A PRBS Generator and Checker based BIST for Multipliers," 2023 International Conference on Innovative Data Communication Technologies and Application (ICIDCA), Uttarakhand, India, 2023, pp. 1-5, doi: 10.1109/ICIDCA56705.2023.10099822.

[89] L. Sayadi, S. Timarchi and A. Sheikh-Akbari, "Two Efficient Approximate Unsigned Multipliers by Developing New Configuration for Approximate 4:2 Compressors," in IEEE Transactions on Circuits and

Systems I: Regular Papers, vol. 70, no. 4, pp. 1649-1659, April 2023, doi: 10.1109/TCSI.2023.3242558.

[90] Z. Zulhelmi, S. H. M. Ali and N. Nasaruddin, "Design and Hardware Realization of 32-Bit Multipliers Based on FPGAs," 2022 International Conference on Electrical Engineering and Informatics (ICELTICs), Banda Aceh, Indonesia, 2022, pp. 153-157, doi: 10.1109/ICELTICs56128.2022.9932055.

Research Publications

1) Jugal Kishore Bhandari, Yogesh Kumar Verma, Laxman Singh, and Santosh Kumar Gupta, “A Novel Design of High-Performance Hybrid Multiplier”, Journal of Circuits, Systems and Computers, 31 (2022) 2250268, doi.org/10.1142/S0218126622502681 [SCI Index]

2) Jugal Kishore Bhandari, Yogesh Kumar Verma and S.K Hima Bindhu (2024), Enhancing FPGA Testing Efficiency: A PRBS-Based Approach for DSP Slices and Multipliers. IJEER 12(1), 139-145. DOI: 10.37391/IJEER.120120. [Scopus]

3) Bhandari, J.K., Verma, Y.K. (2023). Design of TPG BIST Using Various LFSR for Low Power and Low Area. In: Bhateja, V., Mohanty, J.R., Flores Fuentes, W., Maharatna, K. (eds) Communication, Software and Networks. Lecture Notes in Networks and Systems, vol 493. Springer, Singapore. https://doi.org/10.1007/978-981-19-4990-6_8. [Conference]

4) J. K. Bhandari, Y. K. Verma, V. Mishra, A. Kumar and S. K. Gupta, "A PRBS Generator and Checker based BIST for Multipliers," 2023 International Conference on Innovative Data Communication Technologies and Application (ICIDCA), Uttarakhand, India, 2023, pp. 1-5, doi: 10.1109/ICIDCA56705.2023.10099822. [Conference]

Bio-data

Mr. Jugal Kishore Bhandari born on Feb 6, 1987, Hyderabad, Telangana, India received his B.E. degree in Electronics and Communication Engineering from Anna University, Chennai, in the year 2003. He received his M.Tech. degree in VLSI System Design from JNTU, Hyderabad, Telangana, India in the year 2014. He joined Doctoral programme in Electronics and Communication Engineering Department of Lovely Professional University, Phagwara, Punjab, India in the year 2020. His research interest includes the intersection of cutting-edge technology and practical solutions. With a focus on FPGA platforms and DSP applications, his work showcases optimizing performance and resource utilization, addressing real-world challenges through innovative hardware designs. He has published 15 papers including reputed international journals, national and international conferences out of which 4 were published in research duration.