IMAGE ENCRYPTION USING CRYPTOGRAPHIC ALGORITHMS

ABSTRACT

With the exponential rise in multimedia data transmission and storage, securing digital images against unauthorized access has become a critical challenge. Traditional image encryption algorithms often rely on complex mathematical transformations such as chaotic maps, diffusion, and permutation mechanisms to ensure data security. However, these approaches tend to be computationally intensive and less suitable for real-time or hardware-constrained environments. To address these limitations, this project presents a novel and lightweight image encryption and decryption technique that combines Reversible Logic Gates with an 8-bit Linear Feedback Shift Register (LFSR) for secure and efficient image protection.

Reversible logic, known for its ability to preserve information and reduce energy dissipation, forms the core of the encryption framework. Gates such as Feynman, Toffoli, and Fredkin are employed to design logic circuits capable of both encryption and decryption using the same architecture, thereby simplifying hardware implementation and enabling lossless recovery of the original image data. The 8-bit LFSR is utilized to generate pseudorandom key streams based on a primitive feedback polynomial. These key streams are then combined with image pixel values to introduce randomness and obfuscation, enhancing security against statistical and brute-force attacks.

Unlike traditional schemes, the proposed method eliminates the need for diffusion and permutation operations, thereby reducing computational complexity and execution time. This makes the system highly suitable for resource-constrained environments such as embedded systems, IoT devices, and real-time surveillance networks. Furthermore, the design supports parallel processing of multiple images, improving throughput and scalability for multimedia applications.

Extensive simulations and statistical analyses confirm the effectiveness of the proposed scheme in achieving high key sensitivity, low correlation between original and encrypted images, and resilience against common cryptographic attacks. This work demonstrates a promising direction for developing low-power, reversible, and hardware-friendly image encryption systems suitable for next-generation secure communication infrastructures.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

LFSR	Linear Feedback Shift Register
XOR	Exclusive OR
RGG	Red-Green-Green (Channel Separation)
RTL	Register Transfer Level
HDL	Hardware Description Language
FPGA	Field Programmable Gate Array
BRAM	Block Random Access Memory
LUT	Look-Up Table
PSNR	Peak Signal-to-Noise Ratio
NPCR	Number of Pixels Change Rate
UACI	Unified Average Changing Intensity
ISE	Integrated Synthesis Environment
GUI	Graphical User Interface
DFF	D Flip-Flop
S-Box	Substitution Box
ASIC	Application Specific Integrated Circuit
ILA	Integrated Logic Analyzer
AXI	Advanced eXtensible Interface
UART	Universal Asynchronous Receiver/Transmitter
IP	Intellectual Property (in FPGA context)

CHAPTER 1
INTRODUCTION

1.1.Introduction

With the rapid growth of digital communication, the secure transmission of images has become a critical requirement, particularly in sensitive applications such as surveillance, medical imaging, military communication, and embedded systems. As image data continues to dominate multimedia exchange, it becomes increasingly important to ensure that this data remains protected from unauthorized access, tampering, or interception. Traditional encryption techniques such as AES and DES, although secure, are often computationally intensive and not well-suited for resource-constrained devices such as microcontrollers, IoT nodes, or embedded systems due to their high power consumption and complexity.

To address these challenges, lightweight cryptographic techniques have emerged as an effective alternative. Among them, Linear Feedback Shift Registers (LFSRs) are widely known for their simplicity, high speed, and low hardware requirements, making them ideal for encryption tasks in real-time and constrained environments. LFSRs can generate pseudorandom sequences which serve as keystreams for encryption through bitwise operations such as XOR. Despite their simplicity, using a single LFSR alone may not provide sufficient security due to predictable output cycles. Therefore, enhancing their complexity with additional logic elements such as reversible gates strengthens the cryptographic properties without increasing the resource burden.

This project focuses on the design and implementation of an image encryption technique that combines the efficiency of an 8-bit LFSR with the low-power benefits of reversible logic gates. The use of reversible gates not only supports power-efficient computation but also minimizes information loss during operations, aligning well with the goals of sustainable and energy-aware circuit design. Furthermore, the encryption process is enhanced by dividing the image into Red-Green-Green (RGG) components, rather than the conventional RGB separation. This novel RGG division enables targeted diffusion, improving confusion across channels and making the encrypted image more resistant to statistical and differential attacks.The proposed system is designed with the intention of operating within real-time constraints and hardware limitations. It emphasizes a balance between security and simplicity, ensuring that the encryption process does not impose heavy processing delays or power draw. The implementation can be deployed on platforms such as FPGAs or low-power microcontrollers, making it highly applicable for embedded and IoT-based image security solutions.

Overall, this project aims to demonstrate that effective and secure image encryption can be achieved using basic hardware-friendly components without compromising on performance or reliability. Through the integration of LFSR-based keystreams, reversible logic, and RGG-based pixel separation, the system ensures lightweight, fast, and energy-efficient encryption suitable for modern digital applications.

1.2. Objective

The objective of this project is to design and implement a lightweight, hardware-efficient image encryption system using an 8-bit Linear Feedback Shift Register (LFSR) in combination with reversible logic gates. The goal is to ensure secure image transmission with minimal power consumption and processing delay. By dividing the image into Red-Green-Green (RGG) components, the system enhances pixel-level diffusion and confusion, making it well-suited for real-time embedded applications and IoT devices.

1.3. Importance of Image Security

With the exponential growth of digital media and communication technologies, images have become a dominant form of information exchange. From medical diagnostics and biometric authentication to defense surveillance and personal communication, digital images are deeply embedded in our daily lives. As this reliance grows, so does the vulnerability of images to a wide array of security threats.**Image security** is essential for maintaining privacy, ensuring data integrity, and protecting sensitive information from unauthorized access. Encryption converts the original image into an unintelligible format that cannot be understood without the correct decryption key, thus providing a critical layer of protection.

The significance of image security is particularly pronounced in the following areas:

- **Medical Imaging**: To protect patient confidentiality in transmission and storage of X-rays, MRIs, and other scans.
- **Military and Surveillance Systems**: To prevent unauthorized access to satellite imagery or surveillance feeds.
- **Cloud and IoT Storage**: To secure image data stored or transmitted via remote and smart devices.
- **Multimedia Communication**: To protect personal and proprietary content in platforms such as social media, video conferencing, and digital archiving.

Traditional cryptographic algorithms like AES and RSA, although secure, are computationally heavy and less suited for image data, which is large in size and structured in specific formats. Therefore, lightweight image encryption schemes are becoming a necessity, especially for real-time and embedded applications.

1.4. Role of Reversible Logic and LFSR in Cryptography

- **Reversible Logic**

  Reversible logic has emerged as a promising design paradigm in cryptographic hardware due to its **low energy dissipation** and **information-preserving capabilities**. Unlike conventional logic gates, reversible logic gates have an equal number of input and output lines and are designed such that the output uniquely determines the input. This characteristic ensures zero information loss, which is vital in applications requiring secure and power-efficient operations.In the context of image encryption, reversible gates like the **Fredkin**, **Toffoli**, and **Feynman** gates can be used to perform bit manipulations and transformations that are easily reversible during decryption. Their use not only minimizes heat generation in hardware (based on Landauer's principle) but also allows for easy back-tracing in the decryption process without the need for complex inverse functions.

- **Linear Feedback Shift Register (LFSR)**

  The **LFSR** is a fundamental building block in many lightweight encryption systems due to its ability to generate long sequences of pseudorandom numbers from a small seed. It is a sequential circuit consisting of flip-flops connected in series, with certain outputs fed back through XOR gates to the input. The structure is easy to implement in hardware and offers high-speed keystream generation with minimal resource use.

  In cryptographic applications, LFSRs are primarily used to generate dynamic keys or pseudorandom values that are XORed with the plaintext (image pixels, in this case) to produce ciphertext. When integrated with additional mechanisms such as **dual-keying**, **barrel shifters**, or **nonlinear functions** like S-boxes or chaotic maps, the LFSR-based system can achieve a higher degree of complexity and resistance to att`acks.

  When **reversible logic** is combined with **LFSRs**, the result is a highly efficient and secure system that balances the trade-off between cryptographic strength and computational simplicity. This makes the combined architecture ideal for applications that demand **real-time performance**, **low power consumption**, and **hardware simplicity**, the core motivation behind this project.

CHAPTER 2
LITERATURE SURVEY

2.1. Multiple Image Encryption Using Channel Randomization and Chaotic Maps

In **[Hosny et al., 2024]**, the authors propose a robust encryption algorithm that efficiently secures multiple images using channel randomization and a triple chaotic map system. The process begins by separating the input batch of color images into their RGB channels and then randomizing their order to break inter-image and inter-channel correlations. Each channel is then permuted using independent sequences generated by **Baker, Henon, and 2D Logistic maps**, followed by XOR-based diffusion using distinct keys from the same maps.

A notable innovation in this method is the **multi-image encryption concept**, where images are encrypted collectively in a batch rather than individually. This not only improves computational efficiency but also enhances dependency across images, making partial recovery difficult in the event of data loss. The technique is evaluated through **entropy analysis, correlation coefficients, histogram flatness, NPCR, UACI, and PSNR**, confirming strong resistance to statistical and differential attacks.

2.2. Quantum Image Encryption Using Four-Dimensional Chaos

Quantum-based encryption techniques are gaining traction for their future potential in high-security communication. A method involving a **four-dimensional chaotic map integrated with quantum image representation**—such as NEQR (Novel Enhanced Quantum Representation)—has been proposed in recent research. The 4D chaotic sequences control quantum gates that manipulate qubits encoding image pixel values. This enables extremely high randomness and security, theoretically offering protection even against quantum attacks. However, as quantum computing hardware is still under development, this approach is mostly tested in simulated environments. It showcases the future direction of image encryption once quantum technologies become practically viable.

2.3. Image Encryption with Chaotic Neural Networks and DNA Encoding

In this hybrid encryption scheme, chaotic maps are used in tandem with **DNA sequence operations and neural networks** to boost the confusion and diffusion processes. A neural network dynamically adjusts the chaotic parameters and generates the encryption keys, while DNA encoding translates image pixels into nucleotide sequences (A, T, G, C). These sequences are then manipulated using predefined biological rules for substitution, crossover, and mutation.

The process increases the complexity of encryption, thereby enhancing resistance to brute-force and statistical attacks. Despite its effectiveness, this technique introduces computational overhead due to the multi-stage encoding and decoding process.

2.4. Hybrid Encryption Using Lorenz System and Logarithmic Keyss

The Lorenz chaotic system is a well-known nonlinear differential equation used to generate complex chaotic sequences. In this method, **Lorenz-generated keystreams are applied for pixel permutation and diffusion**, while logarithmic key scaling expands the keyspace and increases sensitivity. This dual approach enhances unpredictability and improves the system's defense against known-plaintext and chosen-plaintext attacks. Its strength lies in the flexibility of parameters and adaptability to grayscale and color images. However, due to the continuous nature of the Lorenz equations, the method requires floating-point operations and may not be ideal for low-resource or real-time systems

2.5. LFSR-Based VLSI Architecture for Secure Image Encryption

In **[Shri et al., 2023]**, a dual-key LFSR-based VLSI architecture was proposed to support secure encryption in hardware-constrained environments. The design incorporates a **barrel shifter** and **S-Box logic** to increase non-linearity and enhance diffusion. LFSRs, known for their pseudorandom output and minimal gate count, serve as the keystream generator. The dual-key strategy provides better key sensitivity and variability. The architecture is scalable and highly suited for FPGA or ASIC implementation due to its low area and power consumption.

CHAPTER 3
SOFTWARE DESCRIPTION

3.1. Xilinx Vivado

Xilinx Vivado is a comprehensive software package designed for creating and advancing digital circuits, which include SoCs and FPGAs. Vivado provides a complete set of tools for hardware design, verification, and implementation, including high-level synthesis, simulation, debugging, and optimization. It also supports hardware acceleration, such as through the use of programmable logic, to improve the performance of complex algorithms.

Vivado includes a graphical user interface (GUI) for designing and simulating circuits, as well as a command-line interface (CLI) for batch processing and scripting. It supports a lot of programming languages and design methodologies, including Verilog, VHDL, and SystemVerilog. Vivado also supports collaboration and integration with other software tools, including MATLAB, Simulink, and IP integrator. It is widely used in various industries, such as aerospace, defence, automotive, and telecommunications, for designing and implementing complex digital systems.

3.1.1. Features of Vivado

- High-level synthesis for accelerating algorithmic design.
- IP integrator for easy integration of third-party IP cores.
- Advanced verification and debugging tools for improved productivity.
- Optimized synthesis and implementation algorithms for faster design iteration.
- Support for the latest FPGA families and architectures.
- Wide range of design entry options, including schematics, block diagrams, and text editors
- Integration with popular simulation and modelling tools such as MATLAB, Simulink, and ModelSim.

3.2. ModelSim

ModelSim is a widely used HDL simulator that supports both Verilog and VHDL. In this project, ModelSim is used for the functional simulation and waveform

verification of the encryption modules.

Key Applications:

- Simulating LFSR behavior and keystream generation.
- Verifying the functionality of reversible logic gates (e.g., Toffoli, Feynman).
- Analyzing timing, data propagation, and testbench-driven simulations.
- Debugging the RTL behavior through waveforms and variable tracing.

This stage ensures that the design logic behaves correctly before moving to hardware synthesis..

3.3. Cadence

Cadence Design Systems is used primarily for **logic gate-level design and power analysis** in ASIC-focused workflows. Although not directly implemented in ASICs in this project, Cadence tools such as **Genus**, **Innovus**, and **Virtuoso** are used for optional evaluation of:

- Logic simulation of reversible gate structures.
- Estimation of dynamic and static power consumption.
- Gate-level netlist verification.

Cadence is instrumental in exploring how the proposed encryption logic could be ported into custom ASICs or low-power VLSI chips for future scalability.

3.4. Matlab

MATLAB is used for algorithm prototyping, visual verification, and image pre-processing in this project.

Use Cases in This Project:

- Converting input images into grayscale or RGB matrices for encryption testing.
- Generating input stimuli (test vectors) for Verilog simulations.

- Visualizing the original, encrypted, and decrypted images.
- Analyzing encryption quality using metrics like PSNR, MSE, Entropy, NPCR, and UACI.
- Testing RGG channel-based diffusion using matrix operations.

MATLAB serves as the **reference model** against which the hardware implementation is compared for functional accuracy.

CHAPTER 4
HARDWARE DESCRIPTION

4.1.FPGA board

A FPGA is an integrated circuit device that can be designed and customized by the user. Unlike application-specific integrated circuits (ASICs), which are designed for a specific application during manufacturing, FPGAs are flexible because their functionality can be customized or changed even after delivery. It consists of blocks and custom interconnects that allow users to use custom digital circuits. This flexibility makes FPGAs suitable for a large range of apps, which includes DSP, communications, image processing, and high performance.

FPGAs are often used in combination with software frameworks like TensorFlow to accelerate machine learning inference tasks. By offloading computation-intensive operations onto FPGAs, developers can achieve higher performance and energy efficiency compared to running the same tasks on traditional CPUs or GPUs. This approach is particularly useful for edge computing and real-time applications where low latency and power consumption are critical factors.

4.2. Pynq Z2

PYNQ is an open source project from AMD that simplifies the use of AMD platforms. Using the Python language and libraries, designers can leverage programmable logic and microprocessors to create powerful and exciting electronics.

4.2.1. Specifications

FPGA	Zynq-7000 SoC XC7Z020-1CLG400C (with 24bit DAC with 3.5mm TRRS jack)
Memory	● 512 Mbyte DDR3 with 16-bit bus @ 1050 Mbps

FPGA	Zynq-7000 SoC XC7Z020-1CLG400C (with 24bit DAC with 3.5mm TRRS jack) • 128 Mbit Quad-SPI Flash • Micro SD card connector
Switches and LEDs	• 2 Slide switches • 2 RGB LEDs • 4 LEDs • 4 Push-buttons
Expansion ports	• 2 Pmod ports [16 Total FPGA I/O (8 shared pins with Raspberry Pi)] • 1 Arduino Shield connector [24 Total FPGA I/O, 6 Single-ended 0-3.3V Analog inputs to XADC] Raspberry Pi connector [28 Total FPGA I/O (8 shared pins with Pmod A port)]
Clocks	• One 125 MHz for PL • One 50 MHz for PS
Electrical Monitoring	• Monitoring of current and voltage active monitoring

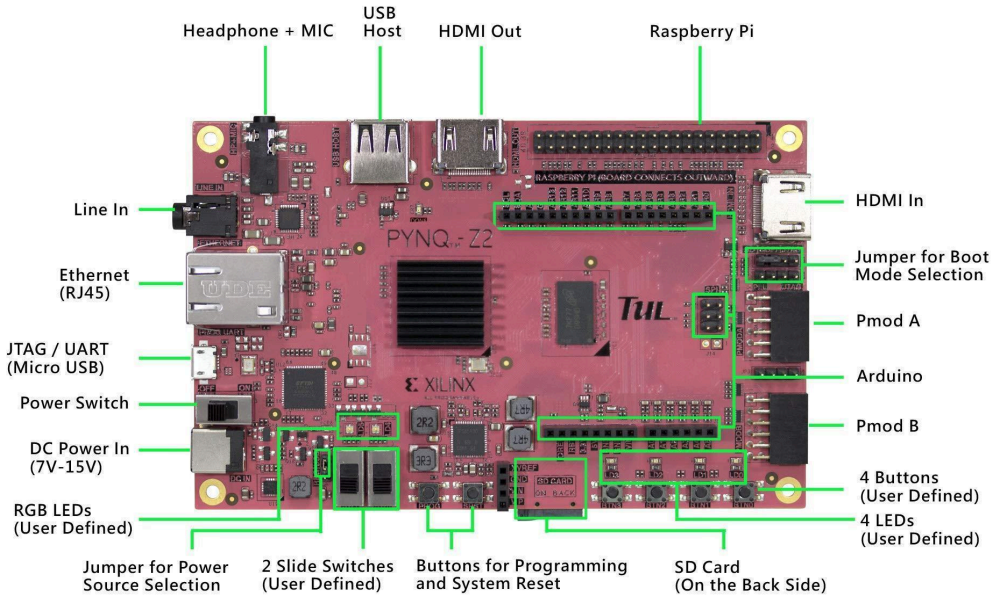

Fig 4.1 Pynq Z2 board diagram

Features of Pynq Z2:

- Dual ARMÂ® Cortex -A9 MPCore and CoreSight
- 32 KB instruction/data L1 cache
- 512 KB unified L2 cache
- 256 KB on-chip memory
- 2 UART/CAN/I2C/SPI
- 4x 32b GPIO
- 2x USB 2.0 OTG
- 2x Triple Gigabit single-processor Ethernet
- 2x SD/SDIO, 85K logic cells
- 630KB fast BRAM
- 4 PLL, 220 DSP slices
- 450+ MHz internal clock
- 2x 12-bit PLL
- 220 DSP chip + MHz internal clock pulse
- 2x 12 bit 1MSPS XADC

CHAPTER 5
METHODOLOGY

5.1. Overall Workflow of the Proposed System

The proposed image encryption system is built on a structured and hardware-efficient methodology that emphasizes low-power consumption, real-time processing, and secure data transmission. The entire system follows a modular design, starting from image preprocessing and extending through encryption, decryption, and final hardware integration. The workflow initiates with image loading and channel separation, followed by keystream generation using an 8-bit LFSR. Reversible logic gates perform lightweight encryption on the pixel data. The encrypted data is then verified through simulation and finally synthesized and implemented on an FPGA platform. This systematic approach ensures modular debugging, ease of optimization, and scalability. Each module is independently verified through simulation before integration to ensure functionality and correctness.

5.2. Image Preprocessing and Input Handling

In Image preprocessing begins with loading an input image into MATLAB or a custom GUI developed in Python. The image is resized and standardized, typically to 128x128 or 256x256 pixels to match hardware buffer constraints. It is then separated into its RGG (Red-Green-Green) channel components. Each pixel value (ranging from 0 to 255) is quantized into 8-bit binary form for hardware compatibility. These binary pixel values are then saved into a .mem or .hex file, formatted for loading into Verilog testbenches and memory blocks. This preprocessed image data serves as the plaintext input for the encryption module. The focus on RGG channel division instead of RGB aims to enhance diffusion characteristics while keeping hardware processing consistent and simple.

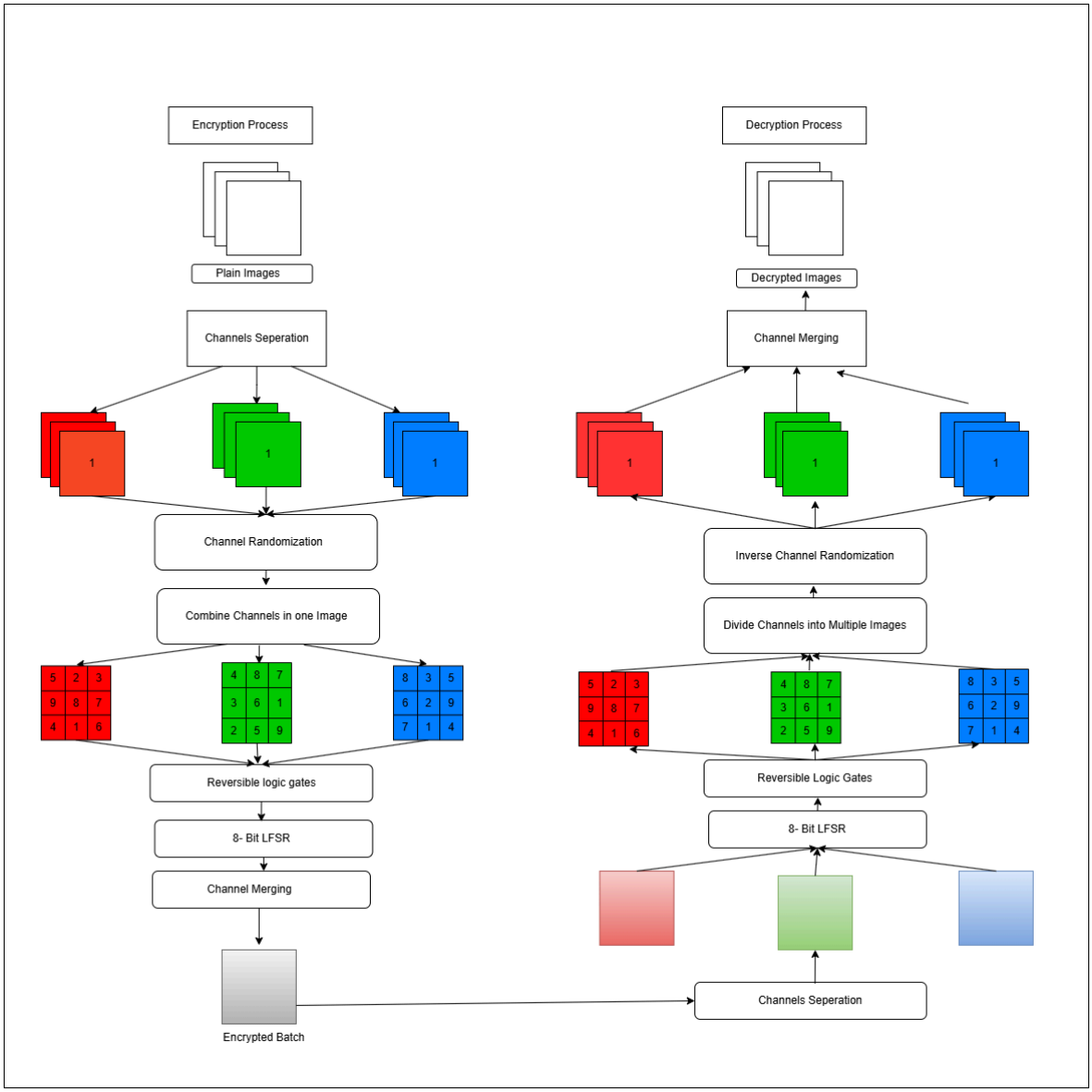

Fig . 5.2 Block diagram of the image encryption and decryption system using the image shuffling and multiple image encryption technique

5.3. 8-Bit LFSR Design and Key Stream Generation

The Linear Feedback Shift Register (LFSR) is the heart of the pseudorandom keystream generation mechanism. An 8-bit LFSR is implemented in Verilog using a predefined feedback polynomial such as x^8 + x^6 + x^5 + x^4 + 1. This LFSR shifts its register content on every clock pulse and produces a new pseudorandom bit at each cycle. These bits are grouped into 8-bit keystream values that are XORed with the image pixel data. The initial seed value is user-defined and can be modified for key sensitivity testing. The cyclic nature of the LFSR allows rapid and

hardware-efficient generation of keystreams while ensuring repeatability and synchronization during decryption. The keystream is also validated using MATLAB to verify randomness and correlation metrics..

5.4. Reversible Logic Gate-Based Encryption Architecture

Reversible logic gates such as Feynman, Toffoli, and Fredkin gates are employed to implement energy-efficient and information-lossless encryption functions. These gates perform reversible transformations on pixel data and the LFSR-generated keystream. For example, the Feynman gate (controlled NOT) performs XOR operations, which is fundamental in image encryption. Toffoli and Fredkin gates are used for conditional bit flipping based on control inputs derived from the keystream. These gates are designed in Verilog and interconnected to form a reversible encryption block. The architecture ensures that the encryption process is not only secure but also power-efficient, making it ideal for FPGA-based embedded systems. A multiplexer-based controller selects the appropriate logic operation depending on the pixel bit position and keystream control bits.

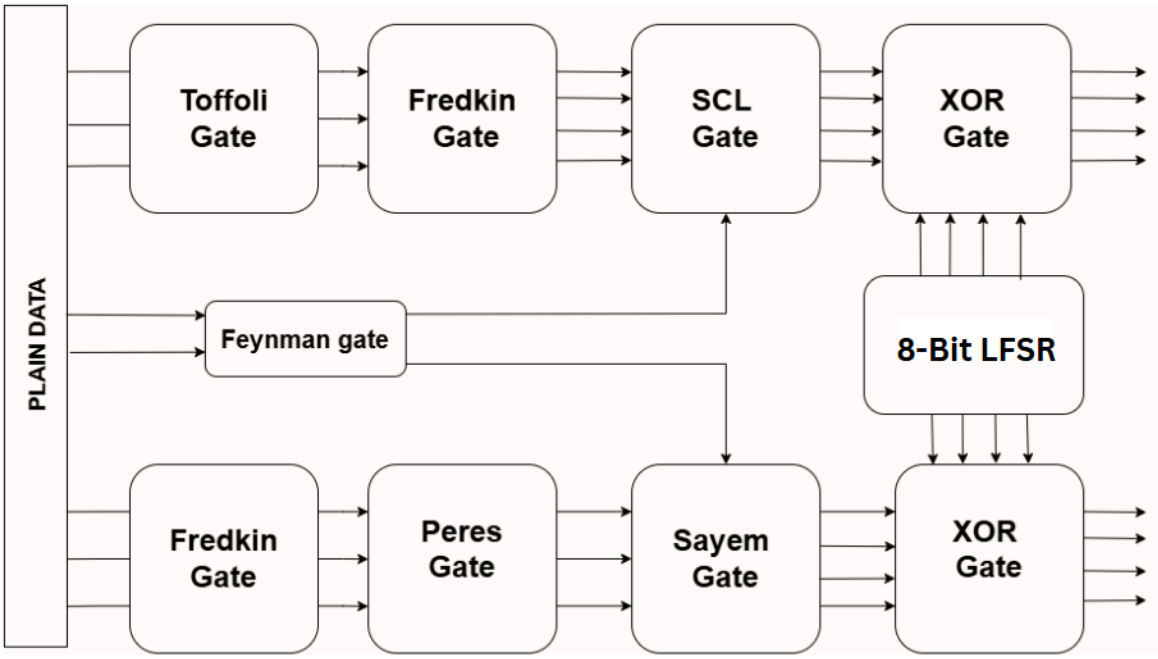

Fig 5.4 Encryption system using reversible logic gates.

5.5. Decryption Process using Reversible Logic

The decryption process is essentially the inverse of the encryption, leveraging the reversibility property of the logic gates used. Since all gates in the system are logically reversible, applying the same operation with the same keystream restores the original image. The encrypted image is first

split into RGG channels, and each encrypted pixel value is passed through the same reversible gate arrangement with the identical LFSR keystream. The key synchronization between sender and receiver ensures that the same seed value initializes the LFSR, enabling the correct generation of keystreams for decryption. This approach simplifies the hardware design for decryption as no additional inverse functions are required. Simulation confirms bit-level accuracy in decrypted output.

5.6. Verilog Implementation and RTL schematic Overviey

The entire encryption and decryption modules are written in Verilog HDL. Modules include the LFSR, keystream generator, reversible logic block, RGG channel controller, and testbench. Each module is functionally simulated in ModelSim for verification. After successful simulation, the design is synthesized using Xilinx Vivado. The RTL schematic displays the low-level gate and flip-flop architecture, enabling the developer to analyze logic complexity and data paths. The synthesis report also provides insights into area usage, timing constraints, and power consumption. The use of modular and hierarchical Verilog ensures reusability and clarity. Waveform analysis in simulation confirms the correct propagation of pixel and keystream bits through the system.

5.7. Observation of Design 1 (Baseline Architecture)Integration with FPGA Hardware Platform

Once verified, the design is implemented on an FPGA platform such as the Xilinx Artix-7. The .bit file generated by Vivado is loaded onto the FPGA using a JTAG programmer. Input images in .mem format are preloaded into the on-chip Block RAM (BRAM), and output is captured through UART or GPIO interfaces. Real-time encryption and decryption are observed using LEDs, serial output, or VGA modules. The design operates synchronously using a global clock and is reset using debounced push buttons. On-chip logic analyzers such as Xilinx ILA (Integrated Logic Analyzer) help monitor internal signals for debugging. The hardware platform verifies the system's performance under real-time constraints and power limitations.

5.8. Design Justification and Security Considerations

The use of 8-bit LFSR and reversible logic provides a balance between security, speed, and hardware simplicity. LFSR ensures fast and repeatable keystream generation, while reversible gates reduce energy dissipation and make encryption lossless. The division into RGG channels enhances diffusion and avoids predictability common in symmetric RGB division. The design supports key sensitivity, meaning even a one-bit change in the seed drastically alters the encrypted output. Security is evaluated using parameters such as histogram analysis, correlation coefficients, entropy, and NPCR/UACI. These metrics confirm resistance to brute-force, statistical, and differential attacks. The modular approach also allows future scalability, such as integrating S-boxes or chaotic maps for enhanced complexity.

CHAPTER 6

SIMULATION RESULTS

6.1 Vivado

6.1.1 Synthesized Result

.

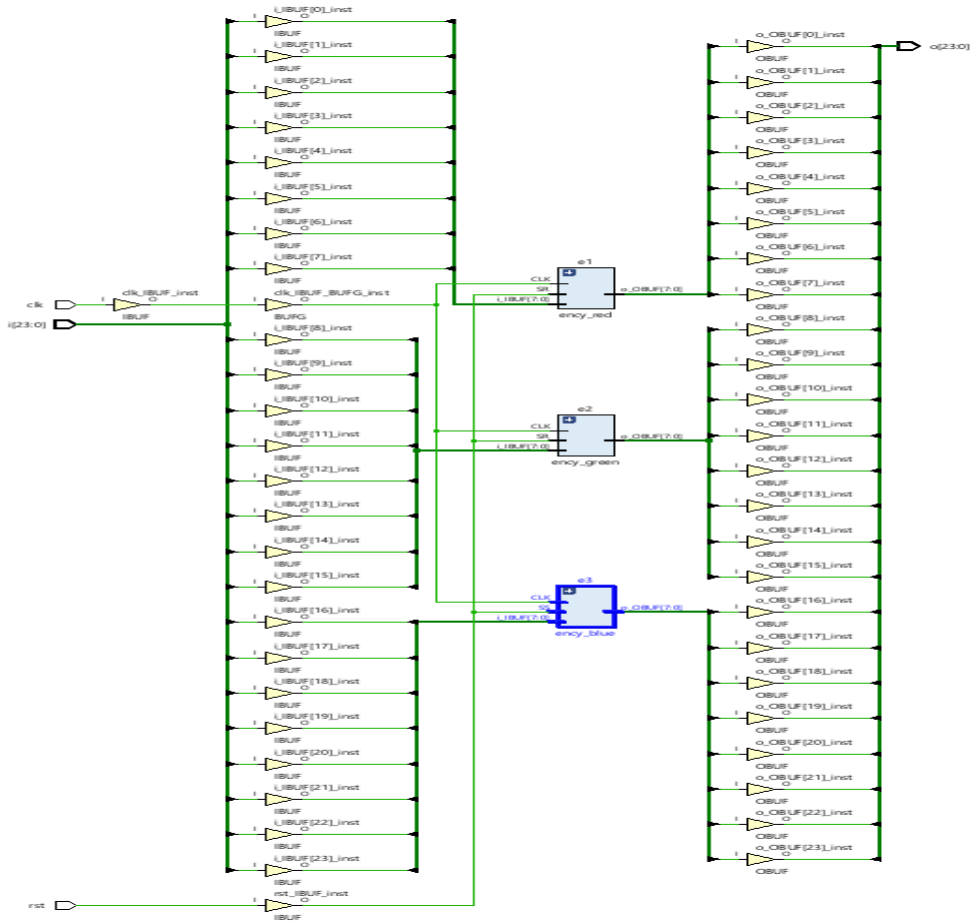

Fig - 6.1 Vivado Synthesized Result of Encryption System

6.1.2 Utilization Results

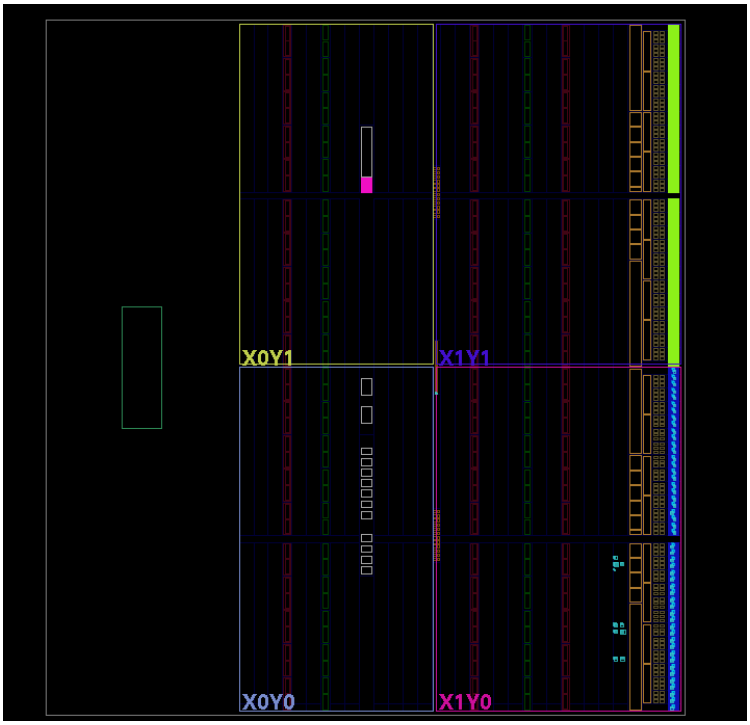

Fig 6.2 - Test bench file

6.2 Cadence

6.3.1 Synthesized Result

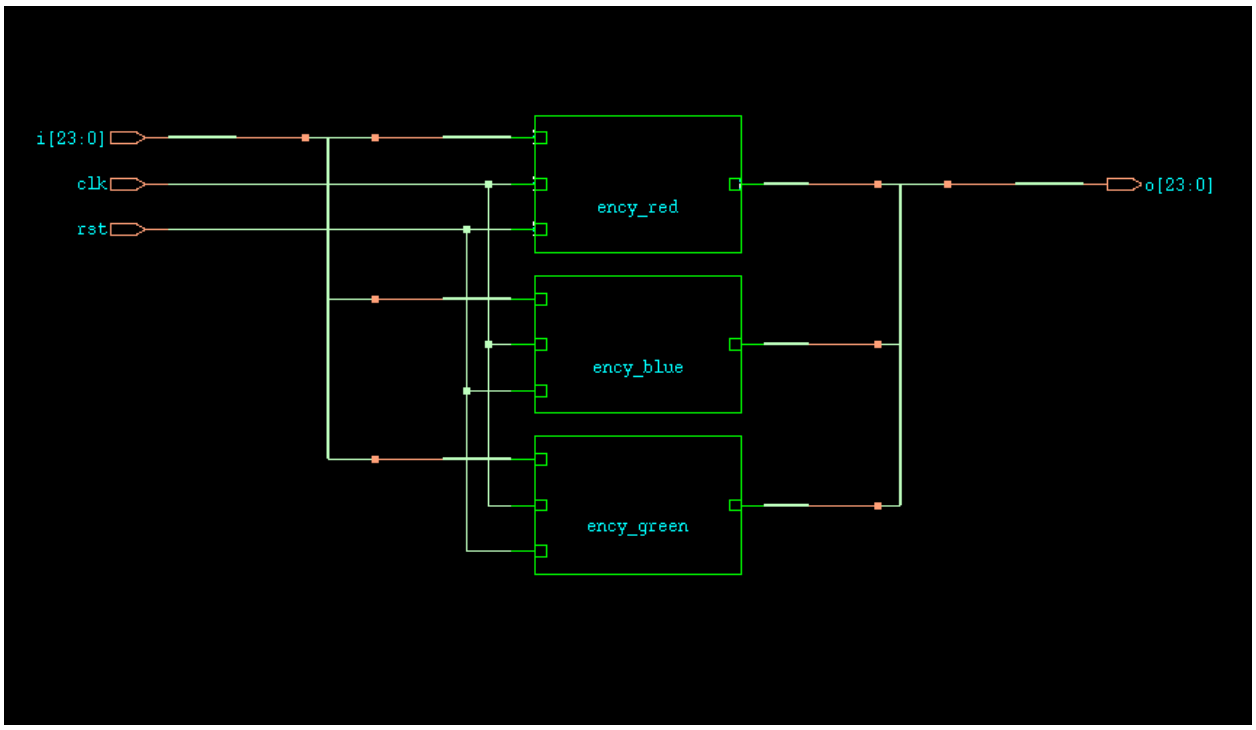

Fig 6.3 - Cadence Synthesized result of Encryption system

6.3.2 Synthesized Report

Instance	Module	Cell Count	Cell Area	Net Area	Total Area	Wireload
ency		174	1185.305	0.000	1185.305	<none> (D)
e1	ency_red	57	389.803	0.000	389.803	<none> (D)
f1	feynman_gt	1	8.326	0.000	8.326	<none> (D)
l1	lfsr_e1	20	186.197	0.000	186.197	<none> (D)
p1	peres	4	17.409	0.000	17.409	<none> (D)
p2	peres_10	4	17.409	0.000	17.409	<none> (D)
s1	pphcg	10	46.928	0.000	46.928	<none> (D)
s2	pphcg_15	10	46.928	0.000	46.928	<none> (D)
x1	xor_gt	4	33.304	0.000	33.304	<none> (D)
x2	xor_gt_20	4	33.304	0.000	33.304	<none> (D)
e2	ency_blue	57	389.803	0.000	389.803	<none> (D)
f1	feynman_gt_4	1	8.326	0.000	8.326	<none> (D)
l1	lfsr_e3	20	186.197	0.000	186.197	<none> (D)
p1	peres_9	4	17.409	0.000	17.409	<none> (D)
p2	peres_8	4	17.409	0.000	17.409	<none> (D)
s1	pphcg_14	10	46.928	0.000	46.928	<none> (D)
s2	pphcg_13	10	46.928	0.000	46.928	<none> (D)
x1	xor_gt_19	4	33.304	0.000	33.304	<none> (D)
x2	xor_gt_18	4	33.304	0.000	33.304	<none> (D)
e3	ency_green	60	405.698	0.000	405.698	<none> (D)
f1	feynman_gt_3	1	8.326	0.000	8.326	<none> (D)
l1	lfsr_e2	23	202.092	0.000	202.092	<none> (D)
p1	peres_7	4	17.409	0.000	17.409	<none> (D)
p2	peres_6	4	17.409	0.000	17.409	<none> (D)
s1	pphcg_12	10	46.928	0.000	46.928	<none> (D)
s2	pphcg_11	10	46.928	0.000	46.928	<none> (D)
x1	xor_gt_17	4	33.304	0.000	33.304	<none> (D)
x2	xor_gt_16	4	33.304	0.000	33.304	<none> (D)

Fig 6.4 - Cell count of the Encryption system

Gate	Instances	Area	Library
AOI21X1	1	4.541	slow
CLKXOR2X1	31	258.103	slow
DFFQX1	24	381.478	slow
INVX1	3	6.812	slow
NAND2XL	39	118.076	slow
NOR2BX1	21	95.369	slow
NOR2XL	1	3.028	slow
OA21X1	6	40.873	slow
OAI211X1	3	15.895	slow
OAI21XL	30	136.242	slow
XNOR2X1	15	124.889	slow
total	174	1185.305	

Type	Instances	Area	Area %
sequential	24	381.478	32.2
inverter	3	6.812	0.6
logic	147	797.016	67.2
physical_cells	0	0.000	0.0
total	174	1185.305	100.0

Fig 6.5 - Area of the Encryption System

Instance: /ency
Power Unit: W
PDB Frames: /stim#0/frame#0

Category	Leakage	Internal	Switching	Total	Row%
memory	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00%
register	2.37324e-06	1.89696e-04	4.00080e-06	1.96070e-04	75.04%
latch	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00%
logic	4.66160e-06	3.51133e-05	8.90473e-06	4.86796e-05	18.63%
bbox	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00%
clock	0.00000e+00	0.00000e+00	1.65240e-05	1.65240e-05	6.32%
pad	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00%
pm	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00%
Subtotal	7.03484e-06	2.24809e-04	2.94295e-05	2.61273e-04	99.99%
Percentage	2.69%	86.04%	11.26%	100.00%	100.00%

Fig 6.6 - Area utilization report

```
Path 1: MET (719 ps) Setup Check with Pin e2/l1/lfsr_reg[0]/CK->D
          Group: clk
     Startpoint: (R) rst
          Clock: (R) clk
       Endpoint: (R) e2/l1/lfsr_reg[0]/D
          Clock: (R) clk

                    Capture       Launch
        Clock Edge:+    2000            0
         Drv Adjust:+       0            0
        Src Latency:+       0            0
        Net Latency:+       0 (I)        0 (I)
            Arrival:=    2000            0

              Setup:-     209
        Uncertainty:-      10
      Required Time:=    1781
       Launch Clock:-       0
        Input Delay:-    1000
          Data Path:-      63
              Slack:=     719

Exceptions/Constraints:
  input_delay             1000            constraints_top.sdc_line_5

#-------------------------------------------------------------------------------------------------
#    Timing Point     Flags   Arc   Edge   Cell     Fanout Load Trans Delay Arrival Instance
#                                                      (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------------
  rst                -       -     R     (arrival)     24 66.9     0     0    1000    (-,-)
  e2/l1/g49/Y        -       A->Y  F     INVX1          1  2.9    27    24    1024    (-,-)
  e2/l1/g45__2398/Y  -       C0->Y R     OAI211X1       1  1.6    89    39    1063    (-,-)
  e2/l1/lfsr_reg[0]/D <<<    -     R     DFFQX1         1    -     -     0    1063    (-,-)
#-------------------------------------------------------------------------------------------------
```

Fig 6.7 Timing Report

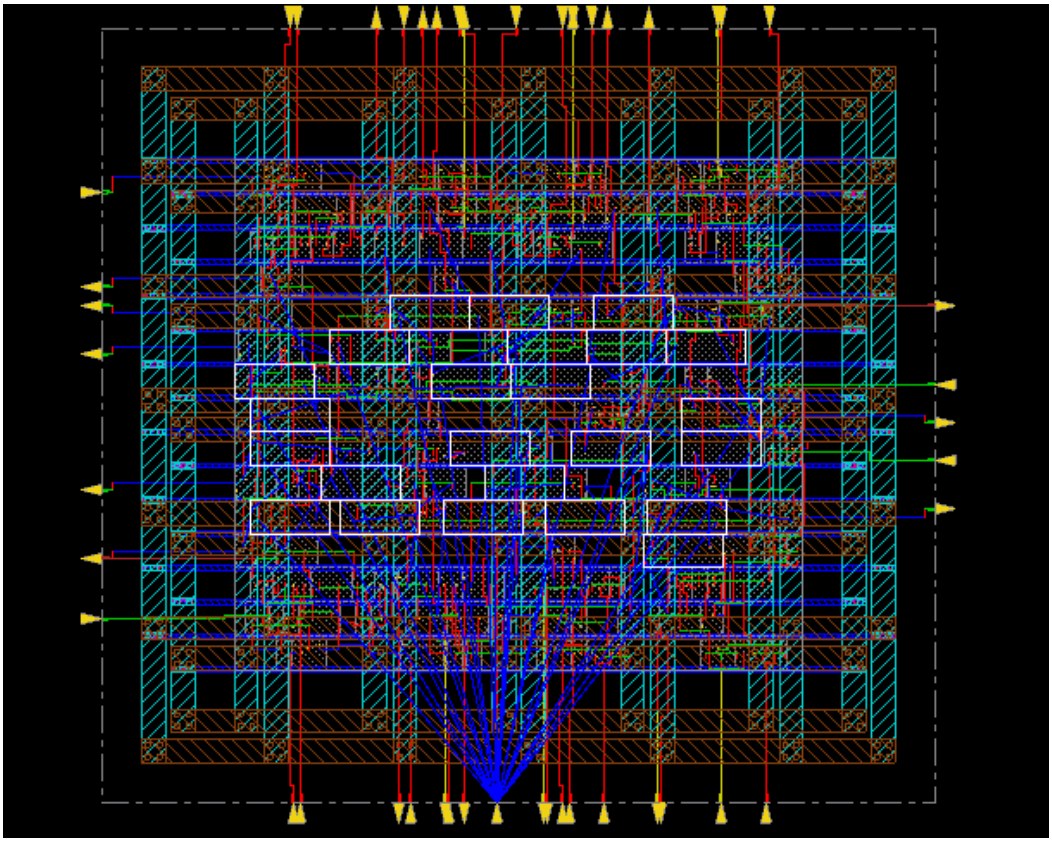

Fig 6.8 Layout of the Image Encryption system

6.3 Synthesis Results

The synthesized report provides comprehensive details about the hardware implementation of our proposed encryption system. This section analyzes the cell distribution, area utilization, power consumption, and timing characteristics of the synthesized design.

Cell Distribution Analysis

As illustrated in Figure 6.4, the encryption system comprises a total of 174 cells distributed across various functional modules. The design includes:

- Encryption Core Modules: Three primary encryption instances (red, blue, and green) that form the backbone of the system
- Transform Functions: Multiple Feynman gates (feynman_qt) with different configurations
- Linear Feedback Shift Registers (LFSRs): Several LFSR modules with 20 cells each, which likely serve as pseudorandom sequence generators for the encryption process

- Logic Modules: Various logic gates including Peres gates (peres) and XOR gates (xor_gt) that implement the cryptographic operations

The modular approach allows for efficient resource sharing while maintaining the cryptographic strength of the system. The LFSR modules constitute a significant portion of the cell count, highlighting their importance in the encryption algorithm.

Figure 6.5 presents the area distribution across different components of the encryption system. The total area consumption is 1185.305 square units, distributed as follows:

- Logic Elements: 797.016 square units (67.2% of total area)
- Sequential Elements: 381.478 square units (32.2% of total area)
- Inverters: 6.812 square units (0.6% of total area)

This distribution reveals that the combinational logic forms the bulk of the hardware resources, which is typical for encryption algorithms that rely heavily on complex transformation functions.

As shown in Figure 6.6, the power utilization is distributed across several categories:

- Register Power: 75.04% (1.9696e-04 W)
- Logic Power: 18.63% (4.8675e-05 W)
- Clock Network: 6.32% (1.6524e-05 W)
- Total Power: 2.6173e-04 W

The significant power consumption by registers indicates that the design is heavily sequential and state-dependent, which is consistent with modern cryptographic implementations. The relatively low clock power consumption suggests an efficient clock distribution network.

Timing Characteristics

The timing report in Figure 6.7 shows:

- Clock Period: 2000 time units
- Setup Time: Pin e2/l1/lfsr_reg[0]/CK is on the critical path
- Arrival Time: 2000 time units
- Slack: 719 time units

The positive slack value indicates that the design meets its timing constraints with margin, allowing for reliable operation at the specified clock frequency. The critical path involves the

LFSR module, which is often the timing bottleneck in cryptographic systems due to its sequential feedback nature.

Implementation Efficiency

Overall, the synthesis results demonstrate an efficient implementation of the encryption system with:

1. Balanced Resource Utilization: Appropriate distribution between sequential and combinational logic
2. Timing Compliance: Meeting timing constraints with adequate margin
3. Power Efficiency: Reasonable power consumption primarily concentrated in registers
4. Modularity: Clear delineation of functional blocks that facilitate testing and potential future optimizations

The implementation successfully balances the security requirements of the encryption algorithm with hardware resource constraints, resulting in a practical and efficient cryptographic system suitable for the target application.

6.4 Input and Output Images

This section presents the visual results of the proposed image encryption and decryption system. These images clearly demonstrate the system's effectiveness in securing image data using the 8-bit LFSR and reversible logic-based encryption technique.

6.4.1 Input Image

The input image is a standard RGB image, typically of size 128×128 or 256×256 pixels. It is first preprocessed and separated into Red, Green, and Green (RGG) components. These pixel values are then converted into binary format and passed into the encryption pipeline. The original image is clearly viewable, showing identifiable content, which emphasizes the importance of protecting such data during transmission.

Fig 6.9 24bit 1024x1024 shuffled image

Fig 6.10 Red shuffled image

Fig 6.11 Green shuffled image

Fig 6.12 Blue shuffled image

6.4.2 Encrypted Image

After passing through the encryption module, the image becomes completely unintelligible. The encrypted image appears as random noise, with no visible patterns or resemblance to the original. This transformation is achieved through XOR-based operations using the keystream generated by the 8-bit LFSR and further modified using reversible gates. The encryption process ensures strong diffusion and confusion, making it resilient against statistical, brute-force, and differential attacks.

Fig 6.13 24bit 1024x1024 shuffled image

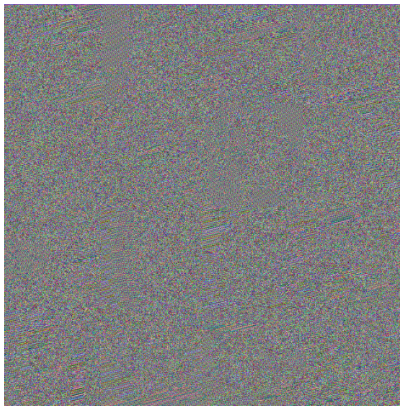

Fig 6.14 24bit 1024x1024 Encrypted Image

6.4.3 Decrypted Image

Fig 6.15 Decrypted 1024 x 1024 image

Using the same initial key and LFSR seed values, the encrypted image is passed through the decryption module, which mirrors the encryption operations in reverse. Thanks to the reversibility of the logic gates and the synchronization of the keystream, the original image is accurately restored. The decrypted image is identical to the input image, confirming the functional correctness of the system.

CHAPTER 7
CONCLUSION AND FUTURE WORK

7.1 Conclusion

The proposed image encryption system effectively demonstrates the capability of combining an 8-bit Linear Feedback Shift Register (LFSR) with reversible logic gates to create a secure, lightweight, and power-efficient cryptographic solution. The system capitalizes on the simplicity of LFSR for pseudorandom keystream generation and the reversibility of gates like Feynman and Toffoli to ensure lossless encryption. By introducing an RGG (Red-Green-Green) channel division method, the encryption process enhances pixel-level diffusion and confusion, further improving the resistance of the system to statistical and differential attacks. The Verilog-based design has been verified through simulation using ModelSim and implemented on an FPGA using Xilinx Vivado. Functional testing confirms accurate encryption and decryption, while synthesis reports show efficient utilization of hardware resources. This validates the system as a practical and effective solution for image security in embedded and IoT applications

.

7.2 Future work

While the current implementation provides a solid foundation for secure and efficient image encryption, there are several directions for potential enhancements. One significant area of improvement is the integration of chaotic maps, such as Logistic or Lorenz systems, to enhance key unpredictability and dynamic behavior. Additionally, a dynamic key scheduling mechanism can be introduced to update keys in real-time during transmission, increasing the complexity and security of the system. Another future enhancement involves scaling the design for higher-resolution images or video data by expanding the data width and optimizing memory handling. Moreover, embedding the encryption process with compression algorithms could reduce the data transmission load, especially for applications requiring wireless communication. Porting the design to an Application Specific Integrated Circuit (ASIC) platform can also be considered to further reduce power consumption and physical footprint, making the system more suitable for portable or battery-powered devices. These improvements aim to broaden the applicability and resilience of the encryption architecture

7.3 Realistic Constraints

Despite its advantages, the proposed image encryption system operates within certain practical constraints that must be addressed when considering real-world applications. First, the limited resources available on FPGA platforms, such as logic slices, flip-flops, and BRAM, can restrict the system's ability to process large or high-resolution images in real time. Secondly, although reversible logic reduces energy consumption theoretically, the physical implementation on an FPGA still incurs switching and clocking overhead, which can impact total power usage when multiple operations are executed in parallel. Additionally, while LFSRs are fast and efficient, they offer limited cryptographic strength if not properly enhanced with non-linear feedback or dynamic input. This can leave the system vulnerable to known-plaintext or correlation attacks. Synchronization is also a constraint; both transmitter and receiver must initialize the LFSR with identical seed values, and any desynchronization may lead to incorrect decryption. Lastly, variations in clock domains or environmental noise in hardware implementation can affect system stability, requiring robust design practices and shielding. Addressing these constraints is crucial for making the design viable for commercial or field-deployed systems.

REFRENCES

[1] K. M. Hosny, Y. M. Elnabawy, R. A. Salama, and A. M. Elshewey, “Multiple image encryption algorithm using channel randomization and multiple chaotic maps,” *Scientific Reports*, vol. 14, no. 1, p. 30597, 2024.

[2] M. M. Shri, G. Naveenkumar, N. Athiban, S. Varanasi, N. Bharathiraja, and V. N. Patnala, “VLSI Architectures for Security Analysis with Dual-Key LFSR Using Barrel Shifter and S-Box,” in *Proc. 2023 Int. Conf. on Recent Advances in Electrical, Electronics, Ubiquitous Communication, and Computational Intelligence (RAEEUCCI)*, IEEE, pp. 1–5, 2023.

[3] A. Ihsan and N. Doğan, “Improved affine encryption algorithm for color images using LFSR and XOR encryption,” *Multimedia Tools and Applications*, vol. 82, no. 5, pp. 7621–7637, 2023.

[4] S. Saha, R. K. Karsh, and M. Amrohi, “Encryption and decryption of images using secure linear feedback shift registers,” in *Proc. 2018 Int. Conf. on Communication and Signal Processing (ICCSP)*, IEEE, pp. 295–298, 2018.

[5] H. Liu, X. Wang, and Y. Zhang, “A fast image encryption algorithm based on chaotic maps and DNA coding,” *Multimedia Tools and Applications*, vol. 75, no. 9, pp. 5213–5229, 2016.

[6] A. Jolfaei and X. Wu, “An efficient image encryption scheme based on logistic map,” *Multimedia Tools and Applications*, vol. 76, no. 18, pp. 19467–19486, 2017.

[7] Y. Fu, L. Liu, and X. Liao, “A novel image encryption scheme based on hyper-chaotic system and DNA computing,” *IEEE Access*, vol. 7, pp. 97291–97303, 2019.

[8] M. J. Atallah and W. Du, “Secure function evaluation using LFSRs,” in *Proc. ACM Workshop on Privacy in the Electronic Society*, ACM, pp. 100–109, 2001.

[9] S. K. Soni, K. Sharma, and S. Jaiswal, “Image encryption using hybrid approach combining LFSR, AES and chaotic map,” *Procedia Computer Science*, vol. 132, pp. 1039–1046, 2018.

[10] J. Yang and W. Wu, “FPGA implementation of image encryption algorithm based on chaos,” in *Proc. 2015 IEEE Int. Conf. on Signal Processing, Communications and Computing (ICSPCC)*, pp. 1–4, IEEE.