"VLSI IMPLEMENTATION OF IMAGE ENCRYPTION AND DECRYPTION USING REVERSIBLE LOGIC GATES"

A PROJECT REPORT
Submitted by

A SIVA PRASAD	**21781A0404**
B POORNA CHANDU	**21781A0420**
B JASWANTH KUMAR	**21781A0421**
B VINAY KUMAR	**21781A0429**
B KARTHIK	**21781A0432**

in partial fulfilment of the requirements for the Award of Degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Under the guidance of

Mrs S.LAVANYA, M Tech
Assistant professor
at

SRI VENKATESWARA COLLEGE OF ENGINEERING& TECHNOLOGY
(AUTONOMOUS)
R.V.S NAGAR, CHITTOOR – 517 127. (A.P).
(Approved by AICTE, New Delhi, Affiliated to JNTUA, Anantapuramu)
(Accredited by NBA, New Delhi AND NAAC, Bengaluru)
(An ISO 9001:2000 Certified Institution)

APRIL 2023

SRI VENKATESWARA COLLEGE OF ENGINEERING& TECHNOLOGY
(AUTONOMOUS)
R.V.S NAGAR, CHITTOOR – 517 127. (A.P)
(Approved by AICTE, New Delhi, Affiliated to JNTUA, Anantapuramu)
Accredited by NBA, New Delhi & NAAC, Bengaluru)
(An ISO 9001:2000 Certified Institution)

CERTIFICATE

This is to certify that this project entitled **" VLSI IMPLEMENTATION OF IMAGE ENCRYPTION AND DECRYPTION USING REVERSIBLE LOGIC GATES "** is a bonafide work done by **A SIVA PRASAD (21781A0404) , B POORNA CHANDU (21781A0420),B JASWANTH KUMAR (21781A0421) ,B VINAY KUMAR (21781A0429) , B KARTHIK (21781A0432) ,** in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY in ELECTRONICS AND COMMUNICATION ENGINEERING**

SIGNATURE OF THE GUIDE **SIGNATURE OF THE HOD**

INTERNAL EXAMINER **EXTERNAL EXAMINER**

Viva-Voce Conducted on:

Acknowledgements

A Grateful thanks **to Dr R.Venkataswamy**,Chairman,Sri Venkateswara College of Engineering and Technology for providing education in their esteemed institution.

We, wish to record our deep sense of gratitude and profound thanks toour beloved Vice Chairman, **Sri R.V. Srinivas** for his valuable support throughout the course.

We, express our sincere thanks **to Dr M. Mohan Babu**, our beloved principal for his encouragement and suggestions during the course of study.

We, wish to convey our gratitude and express our sincere thanks to **Dr D. Sri Hari** , Head of the Department, Electronics and Communication Engineering, for giving us his inspiring guidance in undertaking our project report.

We, express our sincere thanks to the Project Guide **Mrs S. Lavanya** ,M Tech , Assistant Professor , Department of Electronics and Communication Engineering, for her keen interest , stimulating guidance , constant encouragement with our work during all stages, to bring this project into fruition.

We, wish to convey our gratitude and express our sincere thanks to all Project Review Committee members for their support and cooperation rendered for successful submission of our project work.

Finally, we would like to express our sincere thanks to all teaching,nonteaching faculty members, our parents, and friends and for all those who have supported us to complete the project work successfully.

A SIVA PRASAD	21781A0404
B POORNA CHANDU	21781A0420
B JASWANTH KUMAR	21781A0421
B VINAY KUMAR	21781A0429
B KARTHIK	21781A0432

ABSTRACT

Emerging research in reversible computation finds applications in diverse fields like optical computing, digital signal processing, nanotechnologies, bioinformation, and low-power computing. Ensuring data security is crucial in these applications due to power and area constraints. Cryptography protocols face challenges in this regard, as hackers can intercept data during transmission. Encryption is pivotal for safeguarding data, and the Reversible Logic Gate (RLG) enhances security.

Reversible logic synthesis and testing is a fascinating research area as it is an important approach for low power design and quantum computing. Reversible computations have different applications such as quantum computing, nanotechnology, digital signal processing, bio-information etc. All these applications require a cryptography system to restrict the unauthorized access and thus maintain the confidentiality of data. High area and power requirements are some of the major problems of well se cured cryptography algorithms. In this work, a Reversible Logic Gates Cryptography Design (RLGCD) is proposed to overcome these problems. RLGCD is used to design both encryption and decryption architectures. Linear Feedback Shift Register is used to generate the key for encryption and decryption processes. To further improve the security of data watermarking is done using Least Significant Bit (LSB) method. The FPGA performance of RLGCD architecture is evaluated. There is a great improvement in the performance of RLGCD architecture when compared to other conventional systems.

However, high power and area demands persist in secure cryptography. This work introduces Reversible Logic Gates Cryptography Design (RLGCD) for Encryption and Decryption. Key generation employs a Linear Feedback Shift Register, and data security is enhanced via watermarking using the Least Significant Bit (LSB) method. Reversible logic minimizes power dissipation, ensuring lossless information transfer.

Keywords: Reversible Logic Gate Cryptography Design (RLGCD), Linear Feedback Shift Register (LFSR), Cryptography, Encryption and Decryption, Watermarking.

TABLE OF CONTENTS

INDEX

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACRONYM	DESCRYPTION
VLSI	Very Large-Scale Integration
CMOS	Complementary Metal-Oxide Semiconductor
DES	Data Encryption Standard
VHSIC	Very High-Speed Integrated Circuits
VHDL	VHSIC Hardware Description Language
FPGA	Field-Programmable Gate Array
ASIC	Application Specific Integrated Circuit
RLG	Reversible Logic Gates
CLB	Configurable Logic Blocks
EDA	Electronic Design Automation
RTL	Register Transfer Level

LIST OF TABLES

CHAPTER-1

INTRODUCTION

1.1 OVERVIEW:

Reversible logic has received great attention in the recent years due totheir ability to reduce the power dissipation which is the main requirement inlow power VLSI design. It has wide applications in low power CMOS and Optical information processing, DNA computing, quantum computation and nanotechnology. Irreversible hardware computation results in energy dissipation due to information loss. According to Landauer's research, the amount of energy dissipated for every irreversible bit operation, the heat generated due to the loss of one bit of information is very small at roomtemperature but when the number of bits is more as in the case of high speed computational works the heat dissipated by them will be so large that it affects the performance and results in the reduction of lifetime of the components.

In 1973, Bennett showed that KTln2 energy would not dissipate from a system as long as the system allows the reproduction of the inputs from observed outputs. Reversible logic supports the process of running the system both forward and backward. This means that reversible computations can generate inputs from outputs and can stop and go back to any point in the computation history. A circuit is said to be reversible if the input vector can be uniquely recovered from the output vector and there is a one-to-one correspondence between its input and output assignments, i.e. not only theoutputs can be uniquely determined from the inputs, but also the inputs can be recovered from the outputs. Energy dissipation can be reduced or even eliminated if computation becomes Information lossless.

The adoption of reversible logic gates in VLSI implementation for image encryption and decryption holds great promise for achieving both security and power efficiency objectives. By leveraging the principles of reversibility, designers can develop robust and energy-efficient solutions for protecting sensitive visual data in various applications, ranging from digital communication to multimedia storage and transmission.

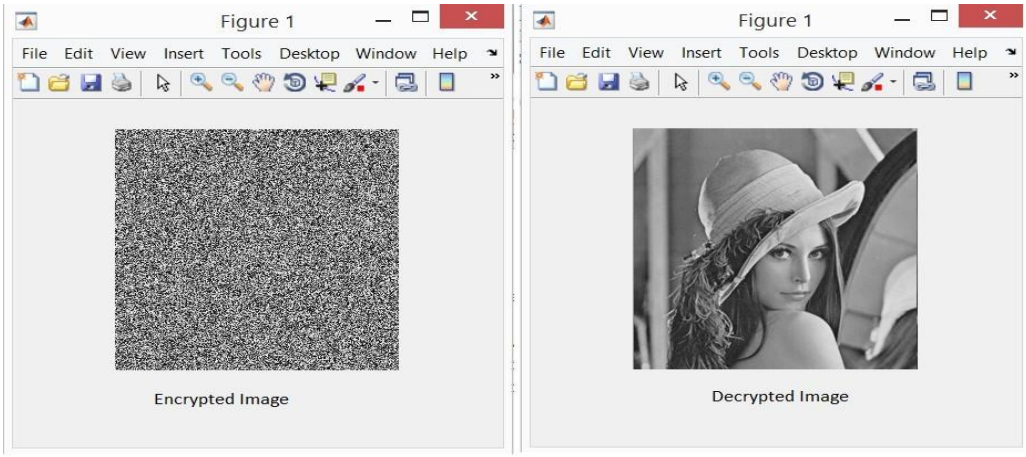


Fig.1.1: Image Encryption and Decryption

1.2 CONCEPT OF REVERSIBILITY:

Reversibility in computing implies that no information about the computational states can ever be lost, so we can recover any earlier stage by computing backwards or uncompacting the results. This is termed as logical reversibility. The benefits of logical reversibility can be gained only after employing physical reversibility. Physical reversibility is a process that dissipates no energy to heat. Absolutely perfect physical reversibility is practically unachievable. Computing systems give off heat when voltage levels change from positive to negative: bits from zero to one.

Most of the energy needed to make that change is given off in the form of heat. Rather than changing voltages to new levels, reversible circuit elementswill gradually move charge from one node to the next. This way, one can only expect to lose a minute Amount of energy on each transition. Reversible computing strongly affects digital logic designs. Reversible logic elements are needed to recover the state of inputs from the outputs. It will impact instruction sets and high-level programming languages as well. Eventually, these will also have to be reversible to provide optimal efficiency.

Cryptography is the process of protecting the information by convertingit in to unreadable format and thus maintains the confidentiality of the data. Thisprocess involves the conversion of plain text into cipher text by the process called encryption and the process by which the original data that is the plain text

is recovered back called decryption. One of the major challenges in VLSI designis the heat dissipation. Now reducing the size of ICs and increasing the number of transistors is happening day by day and up to now all these obeys Moore's law [1]. But with higher integration and scaling the amount of heat that is dissipated also increases.

Landauer's work showed that for each bit of data that is lost there will bea heat dissipation in the range of KTln2. Where, K is the Boltzmann constant and T is the temperature in Kelvin scale. The work done by Bennett presented that this heat dissipation can be eliminated if the traditional irreversible systems are converted in to reversible systems. Reversible computation is the operation in which there is no loss of information and thus scatters only a small amountof heat. That is, there is no decrease in the entropy of the system. In data and telecommunications, cryptography is one of the most necessary parts since the communication even take place over untrusted mediums where the data can be easily hacked out.

A cryptography system not only demands high security but also lowpower consumption. The cryptography system implementation using reversible logic gates offers the best solution for this. A Reversible Logic Gate Cryptography Design (RLGCD) is presented in this paper. The biggest motivation of including reversible technologies in to cryptography includes, it gives energy efficiency much better than other conventional systems and sucha cryptography system is useful for different applications such as medical field, banking, government organization etc. The key for cryptography is generatedby using LFSR [4]. The FPGA performance of the RLGCD architecture is better as compared to existing methods. Data security is importance in present time as lots of information is being communicated via network.

A suitable methodology for privacy transformation is best to make a data protected over network. Different methods are implemented in order to protect the sensitive data. Now a days most of the data is secured by the technique of encryption and certificates. Most of methods are based on cryptography technique. Multi-level encryption is a new concept that is used for making the system more secure than existing cryptosystems. Multi-level encryption is the process of encrypting the plain text with one or more time with same of different

no of keys. It makes the process more complex and powerful than existing.

1.3 CRYPTOGRAPHY:

Cryptography: It is a technique to which information is send in a secure manner so that only authorized user is able to receive this information. It refersto the scrambling of the data and make it meaningless for the third-party during transmission There are three basic components of cryptography system.

- **Plain text:** Source / information/data / original message
- **Key:** Necessary for encryption process.
- **Cypher text:** Unrecognized data /encrypted data / encrypted message

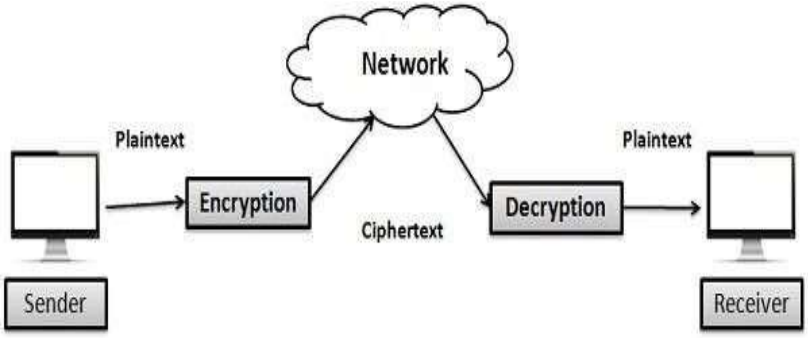


Fig 1.2: Encryption and Decryption Process

The original message is then encoded using encryption algorithm. This process is called encryption. The reverse process to get back the encrypted data into plain text by using decryption algorithm. This process is called decryption. The process of decryption is reverse that of encryption. Cryptography is used to achieve following objectives:

Confidentiality: Confidentiality means to the keep information secret / private. **Data integrity:** It refers to the accuracy and consistency (validity) of data over its lifetime.

Authentication: The property of being genuine and being able to be verified and trusted.

Of course, the algorithm requires the key to be kept secret or long enough so that it takes even longer to break. For example, a 40-bit key has about one trillion combinations whereas a 128-bit key has 3.4*1026 trillion combinations.

There are two general types of key-based algorithms: Symmetric and Asymmetric.

Symmetric Key Algorithms, also known as private-key, conventional, single-key or secret-key algorithms, require that sender and receiver agree on a key before they can communicate securely. In this type of algorithms, the encryption key and the decryption key are the same and security of information depends on the degree of key secrecy from the unauthorized user / intruders.

During the transmission key must remain secret. Encryption process can be done like ENCRYPTIONKEY(MESSAGE) = CIPHER TEXT and
Decryption processes as: DECRYPTIONKEY (CYPHER TEXT) = MESSAGE respectively. This method is extremely fast and efficient. It also provides integrity and confidentiality. But it fails to provide authentication.

Asymmetric key algorithm, also known as public-key algorithms which operate with different encryption and decryption key. Encryption key is made public and anyone can use to encrypt a message, but only an authorized user with the appropriate decryption key can decrypt the message. There are some examples which are based on this type of algorithms like RSA, Rabin and Elgamal.

Asymmetric algorithms are hard to implement and require significant processing power due to fundamental mathematical operations such as modulus. In this paper we will talk about the AES and RSA algorithm and their implementation in multilevel security layers which will be fast and as much more secure than existing AES and RSA. Proposed multi-level encryption can work better compared to single encryption. Multilevel encryption involved the encryption of a message one or more times by using same algorithm with same key or same algorithms with different keys or by using different algorithms. But the proposed algorithm works faster and provide extra security to data in an efficient manner. Not all algorithm with multiple computations is always better but an efficient algorithm can provide same layer of security in faster way.

1.4 INTRODUCTION TO VLSI DESIGN:

The electronics industry has achieved a phenomenal growth over the last two decades, mainly due to the rapid advances in integration technologies,large-scale systems design - in short, due to the advent of VLSI. The number of applications of integrated circuits in high-performance computing, telecommunications, and consumer electronics has been rising steadily, and ata very fast pace.

Typically, the required computational power (or, in other words, the intelligence) of these applications is the driving force for the fast developmentof this field. Gives an overview of the prominent trends in information technologies over the next few decades. The current leading-edge technologies (such as low bit-rate video and cellular communications) already provide the end-users a certain amount of processing power and portability.

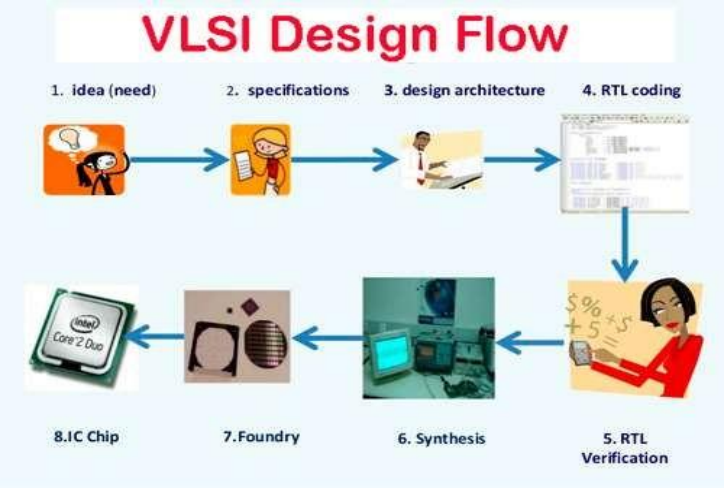


Fig 1.3: VLSI Design Flow

VLSI has been around for a long time, there is nothing new about it, but as a side effect of advances in the world of computers, there has been a dramatic proliferation of tools that can be used to design VLSI circuits. Alongside, obeying Moore's law, the capability of an IC has increased exponentially over the years, in terms of computation power, utilization of available area, yield. The combined effect of these two advances is that people can now put diverse functionality into the IC's, opening up new frontiers. Examples are embedded systems, where intelligent devices are put inside everyday objects, and

ubiquitous computing where small computing devices proliferate to such an extent that even the shoes you wear may actually do something useful like monitoring your heartbeats.

Verilog was developed at a time when designers were looking for tools to combine different levels of simulation. In the early 1980s, there were switch- level simulators, gate-level simulators, functional simulators (often written ad- hoc in software) and no simple means to combine them. Further, the more- widespread, traditional programming languages themselves were/are essentially sequential and thus "semantically challenged" when modeling the concurrencyof digital circuitry.

In 1989, Gateway Design Automation (and rights to Verilog) was purchased by Cadence who put Verilog in the public domain in the following year. This move did much to promote the use of Verilog since other companies were able to develop alternatives tools to those of Cadence which, in turn, allowed users to adopt Verilog without dependency on a single (primarily workstation-tool) supplier. In 1992, work began to create an IEEE standard (IEEE-1364) and in December 1995 the final draft was approved. Thus, Verilog has become an n international standard - which will further increase its commercial development and use. At present, there is standards activity toextend Verilog beyond purely digital circuits.

While Verilog emerged from developments within private companies, its main rival came from the American Department of Defense (DoD). In 1981, the DoD sponsored a workshop on hardware description languages as part of its Very High-Speed Integrated Circuits (VHSIC) program, and the outcome formed a specification for the VHSIC hardware description language (VHDL)in 1983. There is, of course, the question as to which language is better. And this, of course, is a hard question to answer without causing excitement and rebuttals from the marketing departments of the less preferred language. However, the following points featured in a recent debate in the VHDL and Verilog news groups.

The main factor is the language syntax - since Verilog is based on C and VHDL is based on ADA. Verilog is easier to learn since C is a far simpler language. It also produces more compact code: easier both to write and to read.

Furthermore, the large number of engineers who already know C (compared to those who know ADA) makes learning and training easier. VHDL is verystrongly typed, and allows programmer to define their own types although, in practice, the main types used are either the basic types of the language itself, or those defined by the IEEE. The benefit is that type checking is performed by the compiler which can reduce errors; the disadvantage is that changing types must be done explicitly.

Verilog has two clear advantages over VHDL It allows switch-level modeling which some designers find useful for exploring new circuits. It ensures that all signals are initialized to "unknown" which ensure s that all designers will produce the necessary logic to initialize their de sign - the base types in VHDL initialize to zero and the "hasty" designer may omit a global reset.

1.4.1 ASICs:

Progress in the fabrication of IC's has enabled us to create fast and powerful circuits in smaller and smaller devices. This also means that we can pack a lot more of functionality into the same area. The biggest application of this ability is found in the design of ASIC's. These are IC's that are created for specific purposes - each device is created to do a particular job, and do it well. The most common application area for this is DSP - signal filters, image compression, etc. To go to extremes, consider the fact that the digital wristwatchnormally consists of a single IC doing all the time-keeping jobs as well as extra features like games, calendar, etc.

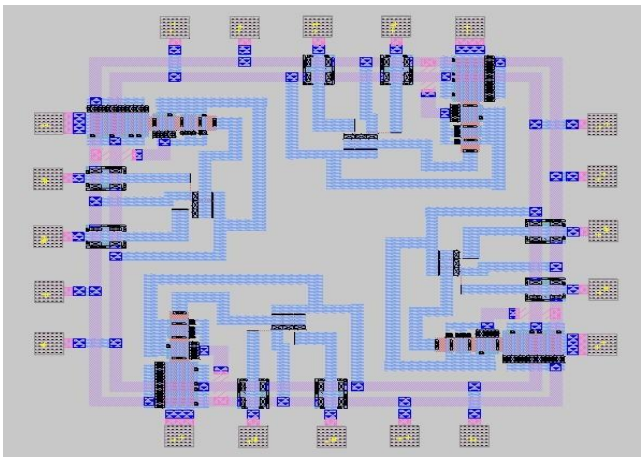


Fig 1.4 ASIC Design

1.4.2 FPGA:

Before the advent of programmable logic, custom logic circuits were built at the board level using standard components, or at the gate level inexpensive application-specific (custom) integrated circuits. The FPGA is an integrated circuit that contains many (64 to over 10,000) identical logic cells that can be viewed as standard components. Each logic cell can independently take on any one of a limited set of personalities. The individual cells are interconnected by a matrix of wires and programmable switches

Similar to microprocessors, FPGAs optionally load or boot themselves automatically from an external nonvolatile memory device. Alternatively, similar to microprocessor peripherals, Spartan-3 generation FPGAs can be downloaded or programmed by an external "smart agent", such as a microprocessor, DSP processor, microcontroller, PC, or board tester. In either case, the configuration data path is either serial to minimize pin requirements or byte-wide for maximum performance or for easier interfaces to processors or to byte-wide Flash memory.

1.5 FIELD PROGRAMMABLE GATE ARRAYS:

Fully fabricated FPGA chips containing thousands of logic gates or even more, with programmable interconnects, are available to users for their custom hardware programming to realize desired functionality. This design style provides a means for fast prototyping and also for cost-effective chip design, especially for low-volume applications. A typical field programmable gate array (FPGA) chip consists of I/O buffers, an array of configurable logic blocks (CLBs), and programmable interconnect structures.

The programming of the interconnects is implemented by programming of RAM cells whose output terminals are connected to the gates of MOS pass transistors. A general architecture of FPGA. A more detailed view showing the locations of switch matrices used for interconnect routing.

A simple CLB (model XC2000 from XILINX) It consists of four signal input terminals (A, B, C, D), a clock signal terminal, user-programmable multiplexers, an SR-latch, and a look-up table (LUT). The LUT is a digital memory that stores the truth table of the Boolean function. Thus, it can generate any function of up to four variables or any two functions of three variables.

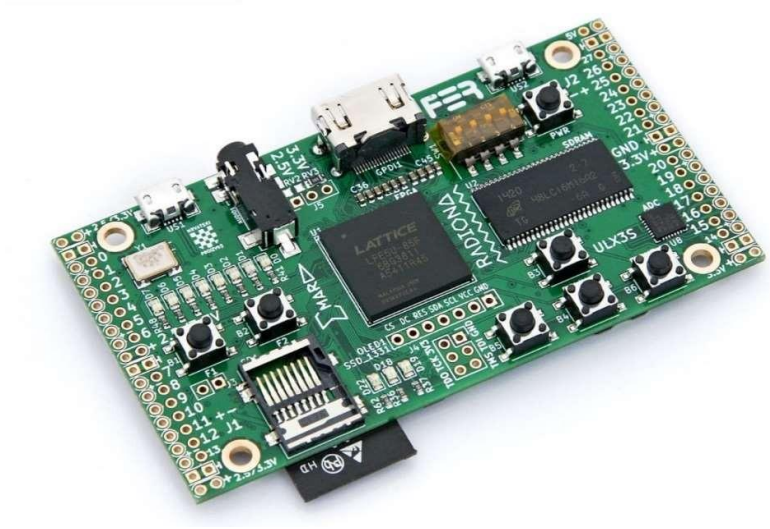


Fig 1.5: ECP5 FPGA board

CLB is configured such that many different logic functions can be realized by programming its array. More sophisticated CLBs have also been introduced to map complex functions. The typical design flow of an FPGA chip starts with the behavioral description of its functionality, using a hardwaredescription language such as VHDL. The synthesized architecture is then technology-mapped (or partitioned) into circuits or logic cells. At this stage, the chip design is completely described in terms of available logic cells. Next, the placement and routing step assigns individual logic cells to FPGA sites (CLBs) and determines the routing patterns among the cells in accordance with the net list. After routing is completed, the on-chip.

The largest advantage of FPGA-based design is the very short turn- around time, i.e., the time required from the start of the design process until a functional chip is available. Since no physical manufacturing step is necessary for customizing the FPGA chip, a functional sample can be obtained almost as soon as the design is mapped into a specific technology.

1.5.1 Gate Array Design:

In view of the fast-prototyping capability, the gate array (GA) comes after the FPGA. While the design implementation of the FPGA chip is done with userprogramming, that of the gate array is done with metal mask design and processing. Gate array implementation requires a two-step manufacturing process: The first phase, which is based on generic (standard) masks, results in

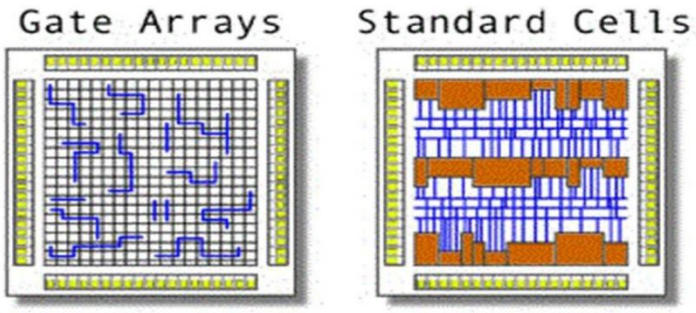
an array of uncommitted transistors on each GA chip.

1.5.2 Standard-Cells Based Design:

The standard-cells based design is one of the most prevalent full custom design styles which require development of a full custom mask set. The standardcell is also called the police. In this design style, all of the commonly used logic cells are developed, characterized, and stored in a standard cell library. A typicallibrary may contain a few hundred cells including inverters, NAND gates, NOR gates, complex AOI, OAI gates, D-latches, and flip-flops. Each gate type can have multiple implementations to provide adequate driving capability for different fan outs. For instance, the inverter gate can have standard size transistors, double size transistors, and quadruple size transistors so that the chip designer can choose the proper size to achieve high circuit speed and layout density. The characterization of each cell is done for several different categories.It consists of

- delay time vs. load capacitance
- circuit simulation model
- timing simulation model
- fault simulation model
- cell data for place-and-route
- mask data

To enable automated placement of the cells and routing of inter-cell connections, each cell layout is designed with a fixed height, so that a number of cells can be abutted side-by-side to form rows. The power and ground rails typically run parallel to the upper and lower boundaries of the cell; thus, neighboring cells share a common power and ground bus. The input and output pins are located on the upper and lower boundaries of the cell. The layout of a typical standard cell. Notice that the n-MOS transistors are located closer to the ground rail while the p-MOS transistors are placed closer to the power rail.



F Fig 1.6 Gate Arrays Vs Standard Cells

1.6 VERILOG

Verilog HDL is a Hardware Description Language (HDL). A Hardware Description Language is a language used to describe a digital system, for example, a computer or a component of a computer. One may describe a digital system at several levels. For example, an HDL might describe the layout of the wires, resistors and transistors on an Integrated Circuit (IC) chip, i.e., the switch level or, it might describe the logical gates and flip flops in a digital system, i.e., the gate level. An even higher level describes the registers and the transfers of vectors of information between registers. This is called the Register Transfer Level (RTL). Verilog supports all of these levels. However, this handout focuseson only the portions of Verilog which support the RTL level.

Verilog is one of the two major Hardware Description Languages (HDL)used by hardware designers in industry and academia. VHDL is the other one. The industry is currently split on which is better. Many feel that Verilog is easierto learn and use than VHDL. As one hardware designer puts it, "I hope the competition uses VHDL." VHDL was made an IEEE Standard in 1987, while Verilog is still in the IEEE standardization process.

1.6.1 History:

Verilog was introduced in 1985 by Gateway Design System Corporation, now a part of Cadence Design Systems, Inc.'s Systems Division. Until May, 1990, with the formation of Open Verilog International (OVI), Verilog HDL was a proprietary language of Cadence. Cadence was motivatedto open the language to the Public Domain with the expectation that the market for Verilog HDL-related software products would grow more rapidly with broader acceptance of the language. Cadence realized that Verilog HDL users

wanted other software and service companies to embrace the language anddevelop Verilog-supported design tools.

Verilog HDL allows a hardware designer to describe designs at a high level of abstraction such as at the architectural or behavioral level as well as the lower implementation levels (i. e., gate and switch levels) leading to VeryLarge-Scale Integration (VLSI) Integrated Circuits (IC) layouts and chip fabrication. A primary use of HDLs is the simulation of designs before the designer must commit to fabrication. This handout does not cover all of Verilog HDL but focuses on the use of Verilog HDL at the architectural or behavioral levels. The handout emphasizes design at the Register Transfer Level (RTL).

1.6.2 Verilog Design Issues:

There are now two industry standard hardware description languages, VHDL and Verilog. The complexity of ASIC and FPGA designs has meant an increase in the number of specialist design consultants with specific tools and with their own libraries of macro and mega cells written in either VHDL or Verilog. As a result, it is important that designers know both VHDL and Verilogand that EDA tools vendors provide tools that provide an environment allowing both languages to be used in unison. For example, a designer might have a model of a PCI bus interface written in VHDL, but wants to use it in a design with macros written in Verilog.

1.7 Importance of HDLs:

HDLs have many advantages compared to traditional schematic-based design.

- Designs can be described at a very abstract level by use of HDLs. Designers can write their RTL description without choosing a specific fabricationtechnology. Logic synthesis tools can automatically convert the design to any fabrication technology. If a new technology emerges, designers do not need to redesign their circuit. They simply input the RTL description to the logic synthesis tool and create a new gate-level net list, using the new fabrication technology. The logic synthesis tool will optimize the circuit in area and timing for the new technology.
- By describing designs in HDLs, functional verification of the design can be done early in the design cycle. Since designers work at the RTL level, they can optimize and modify the RTL description until it meets the desired

functionality. Most design bugs are eliminated at this point. This cuts down design cycle time significantly because the probability of hitting a functionalbug at a later time in the gate-level net list or physical layout is minimized.

- Designing with HDLs is analogous to computer programming. A textual description with comments is an easier way to develop and debug circuits. This also provides a concise representation of the design, compared to gate- level schematics. Gate-level schematics are almost incomprehensible for very complex designs.
- HDL-based design is here to stay. With rapidly increasing complexities of digital circuits and increasingly sophisticated EDA tools, HDLs are now the dominant method for large digital designs. No digital circuit designer can afford to ignore HDL-based design.

1.7.1 Popularity of Verilog HDL:

Verilog HDL has evolved as a standard hardware description language. Verilog HDL offers many useful features

- Verilog HDL is a general-purpose hardware description language that is easy to learn and easy to use. It is similar in syntax to the C programming language. Designers with C programming experience will find it easy to learn Verilog HDL.
- Verilog HDL allows different levels of abstraction to be mixed in the same model. Thus, a designer can define a hardware model in terms of switches, language for stimulus and hierarchical design.
- Most popular logic synthesis tools support Verilog HDL. This makes it the language of choice for designers.
- All fabrication vendors provide Verilog HDL libraries for post logic synthesis simulation. Thus, designing a chip in Verilog HDL allows the widest choice of vendors.
- The Programming Language Interface (PLI) is a powerful feature that allows the user to write custom C code to interact with the internal data structures of Verilog. Designers can customize a Verilog HDL simulator to their needs with the PLI.

1.7.2 Trends in HDLs:

- The speed and complexity of digital circuits have increased rapidly. Designers have responded by designing at higher levels of abstraction. Designers have to think only in terms of functionality. EDA tools take care of the implementation details. With designer assistance, EDA tools have become sophisticated enough to achieve a close-to-optimum implementation.
- The most popular trend currently is to design in HDL at an RTL level, because logic synthesis tools can create gate-level netlists from RTL level design. Behavioral synthesis allowed engineers to design directly in termsof algorithms and the behavior of the circuit, and then use EDA tools to do the translation and optimization in each phase of the design. However,behavioral synthesis did not gain widespread acceptance. Today, RTL design continues to be very popular. Verilog HDL is also being constantly enhanced to meet the needs of new verification methodologies.
- Formal verification and assertion checking techniques have emerged. Formal verification applies formal mathematical techniques to verify the correctness of Verilog HDL descriptions and to establish equivalency between RTL and gate-level netlists.
- New verification languages have also gained rapid acceptance. These languages combine the parallelism and hardware constructs from HDLs with the object-oriented nature of C++. These languages also provide support for automatic stimulus creation, checking, and coverage. However, these languages do not replace Verilog HDL. They simply boost the productivity of the verification process. Verilog HDL is still needed todescribe the design.

CHAPTER-2

LITERATURE SURVEY

TOPIC (1): Architecture Design and VLSI Hardware Implementation of Image Encryption/Decryption System Using Re-configurable 2-D Von Neumann Cellular Automata

AUTHORS: Rong-Jian Chen, Yi-TE Lai, Jui-Lin Lai

The first architecture design and VLSI hardware implementation of imageencryption/decryption system using re-configurable two-dimensional (2-D) von Neumann cellular automata (CA) is presented in this paper. Its encryption scheme is based on replacement of the pixel values using a progressive cellular automata (CA) substitution. In our scheme, we used the re-configurable 2-D von Neumann CA to generate high quality random sequence as key stream. To enhance the flexibility of our system, we used 16 x 16 re-configurable 2-D von Neumann CA which produces 1 set (256) CA, or concurrently produces 4 sets (64/set) 8x8 CA and 16 sets (16/set) 4x4 CA, respectively.

They have accomplished simulations of our image encryption/decryption system by using CADENCE tools. We also have completed the circuit synthesis using the SYNOPSYS tools with the T SMC 0.18um cell-library. The area size was 15.6816 MM, and the maximum operation frequency was 100 MHz with

27.74 MW total dynamic power. It shows that the architecture of the proposedimage encryption/decryption system is suitable for VLSI realization. **Advantages:** High performance, Flexibility, Security and Hardware efficiency **Disadvantages:** Complexity, Power consumption And Area overhead

TOPIC (2): A Nonlinear equation-based cryptosystem for image encryptionand decryption

AUTHORS: Rithmi Mitter, M. Sridevi Sathya Priya

In this paper a new approach for image encryption and decryption using chaotic map and a Non-Linear equation known as BB equation is described. Chaotic maps have been widely used in data encryption. Various chaos Map based encryption and decryption algorithms are used but are found to be insecure. Hence a new method is implemented based on BB (Brahmagupta-

Bhaskara) equation which is combined with chaos to give a no linear dependency and thus improved security. VLSI architecture for the proposed algorithm is designed and realized using Xilinx ISE VLSI software for hardwareimplementation

Advantages: High security and Suitability for real-time applications.

Disadvantages: Complexity, Computational overhead

TOPIC (3): VLSI realization of a secure cryptosystem for image encryption and decryption

AUTHORS:K. Durgha Rao, Ch. Gangadhar

Chaotic maps have been widely used in data encryption. However, a number of chaos-based algorithms have been shown to be insecure. The application of BB equation for encryption is reported in a recent article. In this paper, new algorithms based on chaos and BB equation are reported for image encryption and decryption. The algorithms are illustrated through an example. For practical use, VLSI architectures of the proposed algorithms are designed and realized using Xilinx ISE VLSI software for hardware implementation.

Advantages: High performance, Low power consumption, High reliability and smallsize and weight.

Disadvantages: Complexity and Vulnerability to side-channel attacks.

TOPIC (4): Implementation of encryption and decryption Algorithms forSecurity of Mobile Devices

AUTHOR: B.V.Varun , Abhishek M.V., Akshay Chanabasappa Gangadhar Progress of mobile communication and VLSI technology has aided in development of smart devices. These devices process the information of various formats and sizes in a limited amount of time. This information will be either stored in the devices or in cloud, hence there is a need for some kind ofmethodology to process and secure the data.

Implementation of new algorithms to secure the information is always of immense interest. These algorithms will improve the performance of smart devices and helps for better human-machine interaction. Generally, symmetric and asymmetric approaches are used to secure the data from unauthorized users or attacks. Considering the amount of delay and complexity involved in

processing the data, various forms of algorithms are used. In this paper, we propose a novel algorithm to secure the data from vulnerable attacks. These algorithms can be implemented on various platforms. The experimental results demonstrate an improvement of 10% for contacts and 15% for the encryptionof images as compared to other conventional approaches.

Advantages: Protection of sensitive data, Compliance with regulations,Reduced risk of data breaches, Enhanced security for mobile payments and Improved privacy.

Disadvantages: Reduced performance, Complexity, Key management and Potentialfor data loss.

CHAPTER-3

DES ALGORITHM

3.1 DES Algorithm:

DES is a secret-key archetypal block cipher with block size of 64 bits. DES encrypts a block of 64-bit plaintext into 64-bit cipher text using 64-bit secret key (Left most bit of a block is bit one). Block diagram of the DES algorithm is shown in the Figure 3.DES adopted in 1977 by the National Bureau of Standards now the National Institute of Standards and Technology (NIST), as Federal Information Processing Standard 46 (FIPS PUB 46).

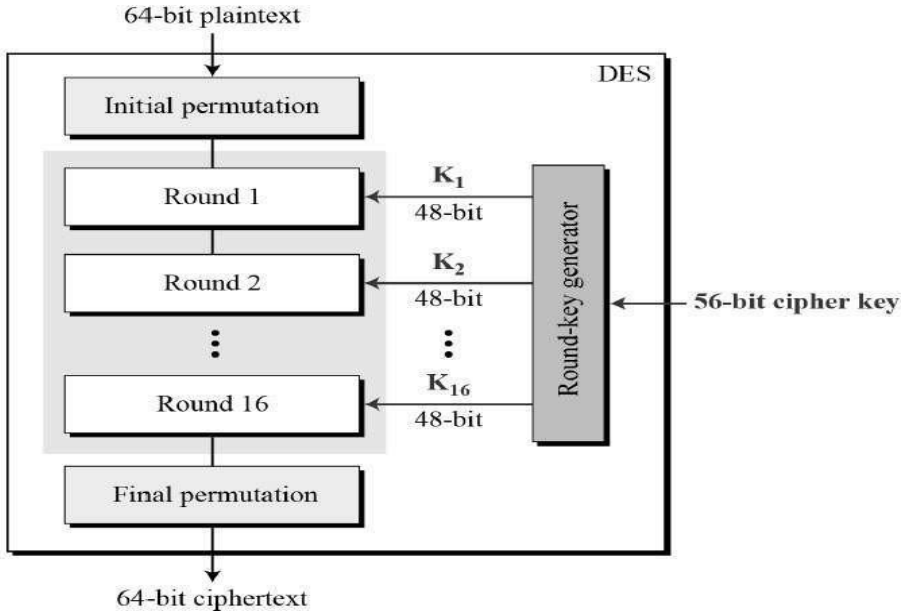


Fig 3.1: General block diagram of DES algorithm

A DES key consists of 64 binary digits ("0"s or "1"s) of which 56 bitsare randomly generated and used directly by the algorithm. The other 8 bits, which are not used by the algorithm, may be used for error detection. The 8 errordetecting bits are set to make the parity of each 8-bit byte of the key odd, i.e., there is an odd number of "1"s in each 8-bit byte. A TDEA key consists of three DES keys, which is also referred to as a key bundle. Authorized users ofencrypted computer data must have the key that was used to encipher the datain order to decrypt it. The encryption algorithms specified in this standard are

commonly known among those using the standard. The cryptographic securityof the data depends on the security provided for the key used to encipher and decipher the data.

Data can be recovered from cipher only by using exactly the same key used to encipher it. Unauthorized recipients of the cipher who know the algorithm but do not have the correct key cannot derive the original data algorithmically. However, it may be feasible to determine the key by a brute force "exhaustion attack." Also, anyone who does have the key and the algorithm can easily decipher the cipher and obtain the original data. A standard algorithm based on a secure key thus provides a basis for exchanging encrypted computer data by issuing the key used to encipher it to those authorized to have the data.

DES Encryption process has two functions

A. Processing the plaintext

B. Round-Key generation

A. Processing the plaintext:

The processing of plaintext proceeds in three phases.

1. Conversion of Plain text into permuted input
2. Production of pre-output using Feistel cipher structure
3. Conversion of pre-output to cipher text

1. Conversion of Plain text into permuted input:

The 64-bit plaintext passes through an initial permutation (IP) that rearranges the bits to produce the permuted input, which is split into two 32-bit halves L0 and R0 where first 32 bit is L0 and next 32-bit is R0.

Permutation is keyless and can be predetermined. This has no cryptographic significance but included to facilitate loading blocks in and out of hardware and to make DES run slower in software.

2. Production of pre-output using Feistel cipher structure:

Most symmetric block encryption algorithms are based on Feistel structure. Feistel proposed the use of a cipher that alternates substitutions and permutations which is a practical application of a product cipher that alternates confusion and diffusion functions producing. Substitution-Permutation

Network (SP Network).

3. ***Conversion of pre-output to cipher text:*** The pre-output is passed through a permutation (IP-1) that is the inverse of the initial permutation function, to produce the 64-bit cipher text. This stage has no cryptography significance in DES. The initial and final permutations are straight P-boxes that are inverses of each other.

B. Function 2- Round-Key generation:

DES takes 64-bit key as input. Among 64-bit key only 56 bits areeffective and used directly by the algorithm. The other 8 bits, which are not usedby the algorithm, may be used for error detection or set arbitrarily or can be ignored. The 8 error detecting bits are set to make the parity of each 8-bit byteof the key odd, i.e., there is an odd number of "1"s in each byte. The round-key generator creates sixteen 48-bit round/sub keys out of a 56-bit cipher key. The round key generation block is shown in Figure 3.2.

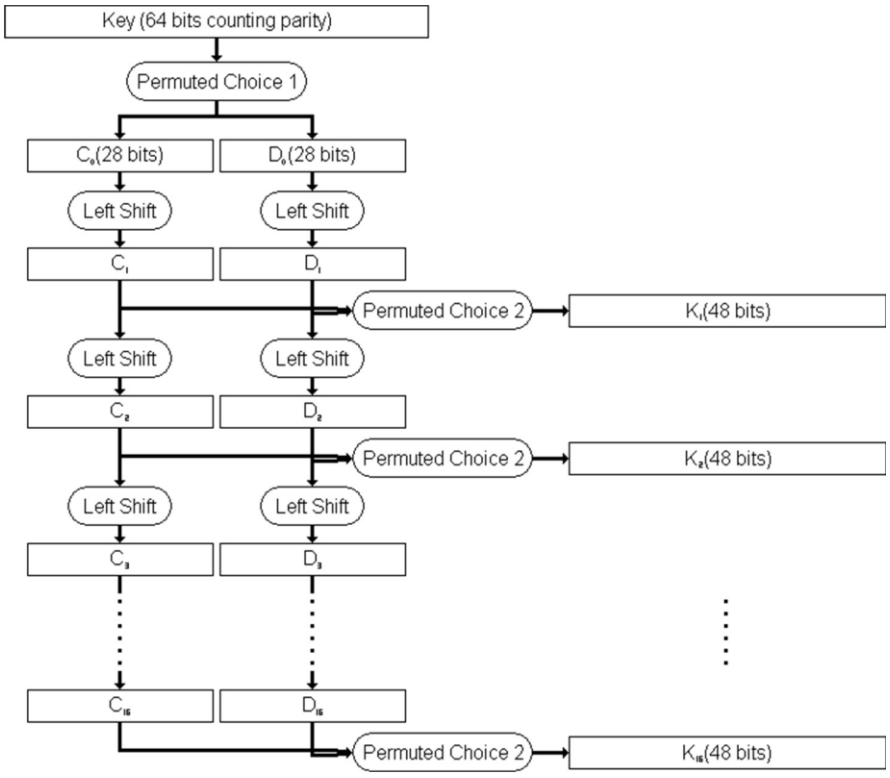


Fig 3.2 Round key generation block diagram

3.2 DRAWBACKS OF EXISTING SYSTEM:

The Data Encryption Standard (DES) algorithm has several drawbacks:

Key Length: DES uses a relatively short key length of only 56 bits, which makes it vulnerable to brute force attacks using modern computing power.

Security Concerns: DES is considered insecure by modern standards due to its vulnerability to brute force attacks.

Limited Block Size: DES has a fixed block size of 64 bits, which can be a limitation in certain applications where larger block sizes are needed for secure encryption.

Weaknesses in the Algorithm: DES has known weaknesses, such as its vulnerability to differential cryptanalysis, which can be exploited by attackersto recover plaintext from ciphertext without knowledge of the key.

Single Algorithm: Unlike modern encryption standards, DES relies on a single algorithm without support for different modes of operation, which limits its flexibility and versatility in different cryptographic scenarios.

Overall, while DES was once widely used and considered secure, itsvulnerabilities and limitations have rendered it obsolete in modern cryptographic applications.

CHAPTER-4

REVERSIBLE LOGIC GATES

4.1 Reversible Logic Gates (RLGs):

RLGs are the circuits that having equal number of inputs and outputs with a unique one to one mapping relationship. Thus, it is possible to recover the input pattern from the output pattern, so that there is no information loss during computation. For example, let 110 is the pattern which is given as input to RLG. Then after completing the logic operation, it produces 001 as output. If we applythis 001 as input and obtained 110 as output then it depicts the occurrence of a reversible operation. while using traditional combinational logic circuits, for every bit of data that is lost during operation there will be an equivalent heat energy dissipation.

The reason behind this is according to the second law of thermodynamics there is no way to reproduce the information once lost. So, when the computation is performed in a reversible manner then it is possible to achieve a logical zero power dissipation. i.e., there is no decrease in the entropy of the system. Constraints for designing RLGs include [9]

- RLGs do not allow fanout.
- Quantum cost should be minimum as possible.
- Optimize the design to make garbage outputs minimum.
- A reversible logic circuits should have least gate level.

The original motivation was that reversible gates dissipate less heat (or, in principle, no heat). In a normal gate, input states are lost, since less information is present in the output than was present at the input. This loss of information loses energy to the surrounding area as heat, because of thermodynamic entropy. Another way to understand this is that charges on a circuit are grounded and thus flow away, taking a small quantity of energy with them when they change state. A reversible gate only moves the states around, and since no information is lost, energy is conserved.

4.1.1 Universality and Toffoli gate:

Any reversible gate must have the same number of input and output bits, by the pigeonhole principle. For one input bit, there are two possible reversible

gates. One of them is NOT. The other is the identity gate which maps its input to the output unchanged. For two input bits, the only non-trivial gate is the controlled NOT gate which XORs the first bit to the second bit and leaves the first bit unchanged.

Truth table

INPUT		OUTPUT	
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Permutation matrix form

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Unfortunately, there are reversible functions that cannot be computed using just those gates. In other words, the set consisting of NOT and XOR gates is not universal. If we want to compute an arbitrary function using reversible gates, we need another gate. One possibility is the Toffoli gate, proposed in 1980by Toffoli.

This gate has 3-bit inputs and outputs. If the first two bits are set, it flips the third bit. The following is a table of the input and output bits

Truth table

INPUT			OUTPUT		
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

Permutation matrix form

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

It can be also described as mapping bits a, b and c to a, b and c XOR (a ANDb).

The Toffoli gate is universal; this means that for any Boolean function $f(x_1, x_2, ..., x_m)$, there is a circuit consisting of Toffoli gates which takes x_1, x_2, ..., x_m and some extra bits set to 0 or 1 and outputs x_1, x_2, ..., x_m, $f(x_1, x_2, ..., x_m)$, and some extra bits (called garbage). Essentially, this means that one can use Toffoli gates to build systems that will perform any desired Boolean function computation in a reversible manner.

4.1.2 Fredkin gate:

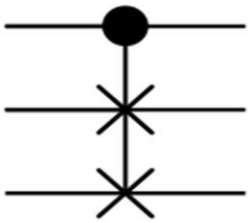


Fig 4.1.3 Circuit representation of Fredkin gate

The Fredkin gate (also CSWAP gate) is a computational circuit suitable for reversible computing, invented by Ed Fredkin. It is universal, which means that any logical or arithmetic operation can be constructed entirely of Fredkin gates. The Fredkin gate is the three-bit gate that swaps the last two bits if thefirst bit is 1.

Definition:

The basic Fredkin gate is a controlled swap gate that maps three inputs (C, I_1, I_2) onto three outputs (C, O_1, O_2). The C input is mapped directly to theC output. If C = 0, no swap is performed; I_1 maps to O_1, and I_2 maps to O_2. Otherwise, the two outputs are swapped so that I_1 maps to O_2, and I_2 maps to O_1. It is easy to see that this circuit is reversible, i.e., "undoes itself" when run backwards. A generalized n×n Fredkin gate passes its first n-2 inputs unchanged to the corresponding outputs, and swaps its last two outputs if and only if the first n-2 inputs are all 1. The Fredkin gate is the reversible three-bit gate that swaps the last two bits if the first bit is 1.

Truth table **Matrix form**

INPUT OUTPUT

C	I_1	I_2	C	O_1	O_2
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	0	1
1	1	1	1	1	1

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

It has the useful property that the numbers of 0s and 1s are conserved throughout, which in the billiard ball model means the same number of balls are output as input. This corresponds nicely to the conservation of mass in physics, and helps to show that the model is not wasteful.

Logic function with XOR and AND gates:

$O_1 = I_1$ XOR $SO_2 = I_2$ XOR S

with S = (I_1 XOR I_2) AND C

It can also be implemented by the following logic:

O_1 = (NOT C AND I_1) OR (C AND I_2) = $CI_1+CI_2O_2$ = (C AND I_1) OR (NOT C AND I_2) = CI_1+CI_2

$C_{out}= C_{in}$

4.1.3. Feynman gate:

Feynman gate is a 2*2 one through reversible gate as shown in figure 1. The input vector is I (A, B) and the output vector is O (P, Q). The outputs are defined by P=A, Q=A⊕B. Quantum cost of a Feynman gate is 1. Feynman Gate(FG) can be used as a copying gate. Since a fan out is not allowed in reversible logic, this gate is useful for duplication of the required outputs.

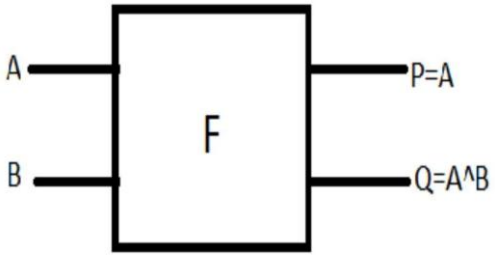


Fig 4.1.4.1 Feynman Gate

Table 4.1.4.1 Truth table of Feynman gates

A	B	P	Q
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

4.1.4 Double Feynman Gate (F2G):

Fig 4.4 shows a 3*3 Double Feynman gate. The input vector is I (A, B, C) and the output vector is O (P, Q, R). The outputs are defined by P = A, Q=A⊕B, R=A⊕C. Quantum cost of double Feynman gate is 2.

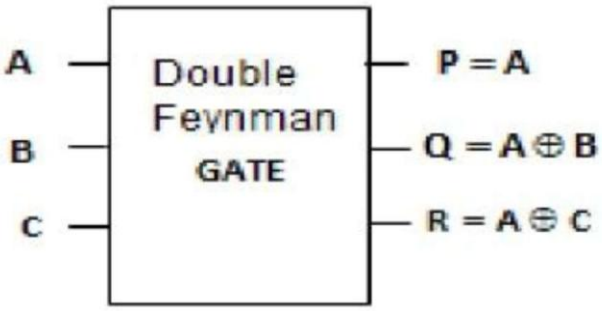


Fig 4.1.4.2 Double Feynman Gate

Table 4.1.4.2 Truth table of double Feynman gate

A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	1	1
1	0	1	1	1	0
1	1	0	1	0	1
1	1	1	1	0	0

4.1.5 Peres Gate:

Fig shows a 3*3 Peres gate. The input vector is I (A, B, C) and the outputvector is O (P, Q, R). The output is defined by P = A, Q = A⊕B and R=AB⊕C.Quantum cost of a Peres gate is 4. In the proposed design Peres gate is used because of its lowest quantum cost.

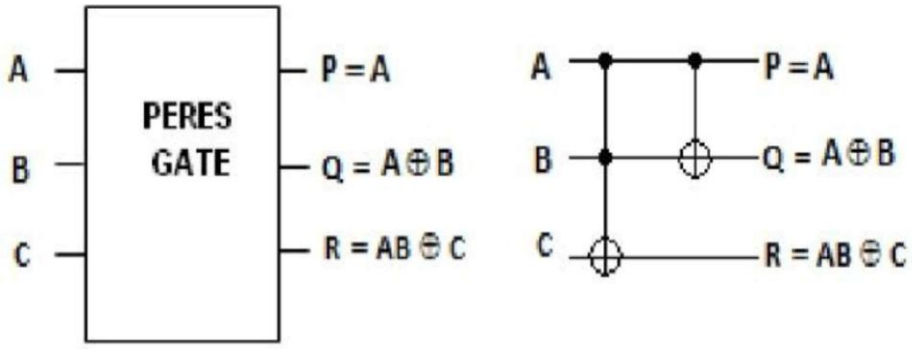


Fig 4.1.5.1 Peres gate

Table 4.1.5.1 Truth table of Peres gate

A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

A full- adder using two Peres gates is as shown in fig. The quantum realizationof this shows that its quantum cost is 8 two Peres gates are used.

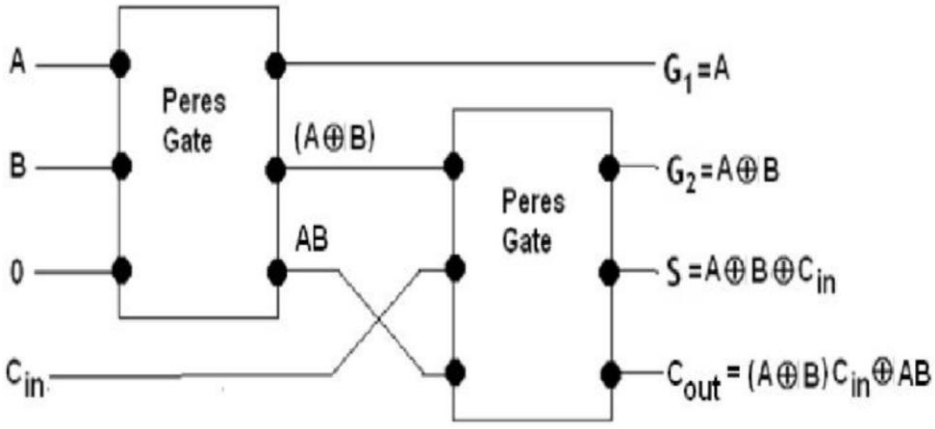


Fig 4.1.5.2 Full adder using two Peres gates

A single 4*4 reversible gate called PFAG gate with quantum cost of 8 is usedto realize the multiplier.

4.1.6 TSG Gate:

Fig shows a 4*4 TSG gate. The input vector is I (A, B, C, D) and the output vector is O (P, Q, R, S). The output is defined by P = A, Q = A'C'⊕B',R = (A'C'⊕B') ⊕ D and S = (A'C'⊕B'). D⊕ (AB⊕C) Quantum cost of a Peres gate is 4. In the proposed design Peres gate is used because of its lowest quantum cost. It can be verified that the input pattern corresponding to a

particular output pattern can be uniquely determined. The proposed TSG gate is capable of implementing all Boolean functions and can also work singly as a reversible Full adder

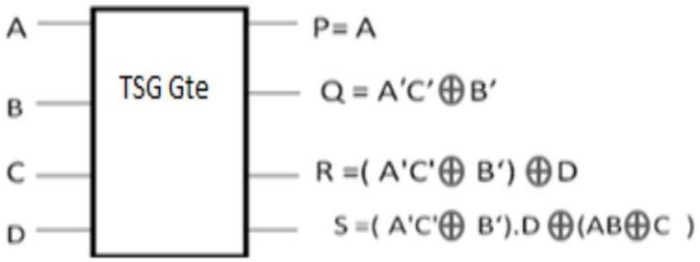


Fig 4.1.6 TSG gate

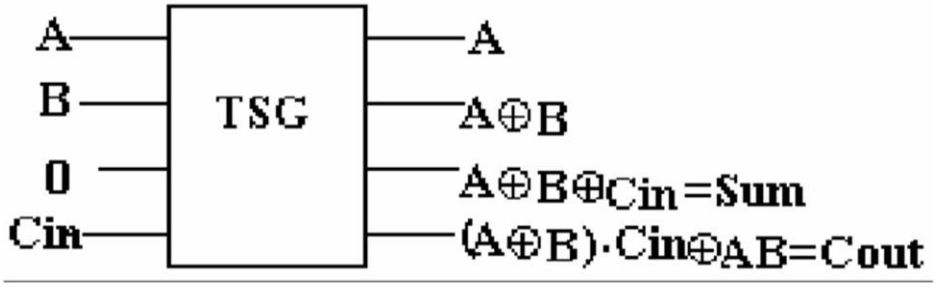


Fig 4.1.6 TSG Gate Working as Reversible Full Adder

The RLG that are used to design this new cryptography system includes Feynman gate, Fredkin gate, Toffoli's gate and SCL gates and are shown in Fig.4.8.

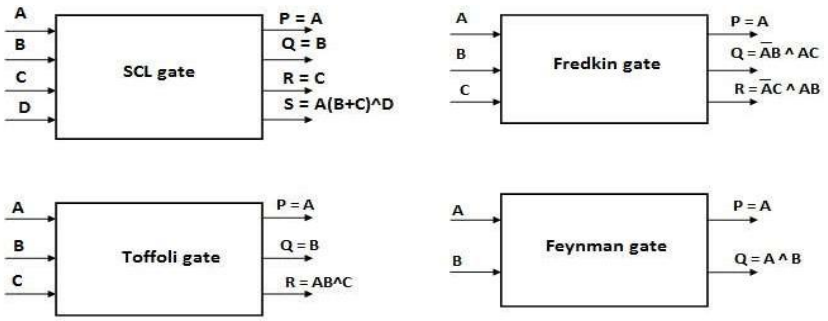


Fig.4.1.7 Block diagram of RLGs

4.2 Proposed Methodology:

4.2.1 Working Principle:

The working principle of the proposed RLGCD is described below.

Step 1: MATLAB is used to read the input image and, on this image,watermarking is performed.

Step 2: The LSB watermarking is used and after watermarking process thewatermarked input image is converted into binary image.

Step 3: The binary image pixel values will be written into a text file with in theMATLAB.

Step 4: In order to perform the cryptography processes a key is required. This key is created using the LFSR.

Step 5: The input to Verilog code is the text file output from MATLAB and in Verilog the cryptography processes such as encryption and decryption are performed.

Step 6: Then, both the encryption output and decryption output are copied in to text files in Verilog for output verification.

Step 7: In MATLAB the pixels are reconstructed from the encrypted binary pixel values and decrypted binary pixel values in the text file. The encrypted and decrypted images are then generated from these pixel values.

Step 8: Then, input image and decrypted image will be the same.

Step 9: The watermark is extracted back from the decrypted image.

4.2.2 LSB watermarking:

The least significant bit (LSB) is one of the simplest watermarks embedding technique. In LSB watermarking the LSB of the original image pixelbits are changed by the bits of watermarking data. The changes thus created cannot be find out by human visibility system. Since an LSB can hold 1 bit of data, an image of size 256x256 can thus able to store a total of 65536 bits of secret data. Simple LSB watermarking can survive transformations like lossy compression, cropping or other additions of noise but a more sophisticatedattacker can easily extract the changed bits. So, in this work watermark embedding is performed into the third and fourth LSB of the original image.

There is less probability that anyone expects insertion of secret data in

these LSB positions. This will help to enhance the security of the system. First the original input image of size 256x256 is read in MATLAB and transfer the watermark into binary value after typing it. The data is then embedded into the third and then fourth LSBs of the image starting from the first.

First the length of the binary watermark data is embedded in the third and fourth LSBs of the first eight pixels with a gap of 5 pixels. So, the maximumlength of binary watermark is 817 and thus it will ask us to rewrite the data if the length is more than 817. After the length of the data, the watermark data is written in to the third and fourth LSB with a gap of five pixels.

Thus, the watermarked input image is obtained. In watermark extraction process after decryption, the reverse operation of embedding is performed. First of all, the length of secret data is extracted from the third and fourth LSBs starting from the first pixel and jump by five until it gets it from the eight pixels. Then in the same way we get the embedded data from the third and fourth LSBs. The obtained binary data is then converted back to the character which will give the watermark that we applied. The input image can be gray scale image or a color image. For color image watermarking is performed on the blue component of the image. This is because it is less sensitive to human visual system.

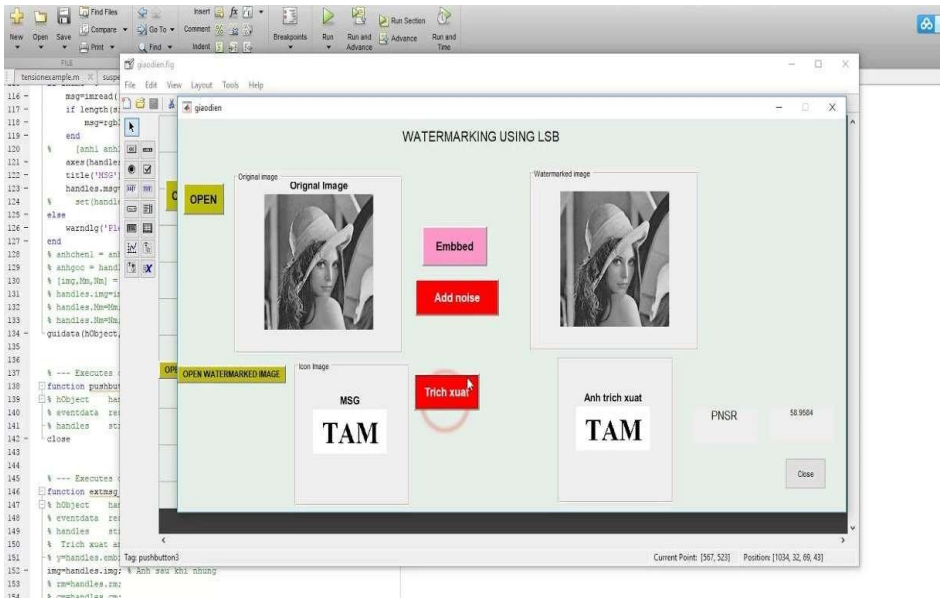


Fig 4.2.2.1 Watermarking using LSB

4.2.3. Block Diagram:

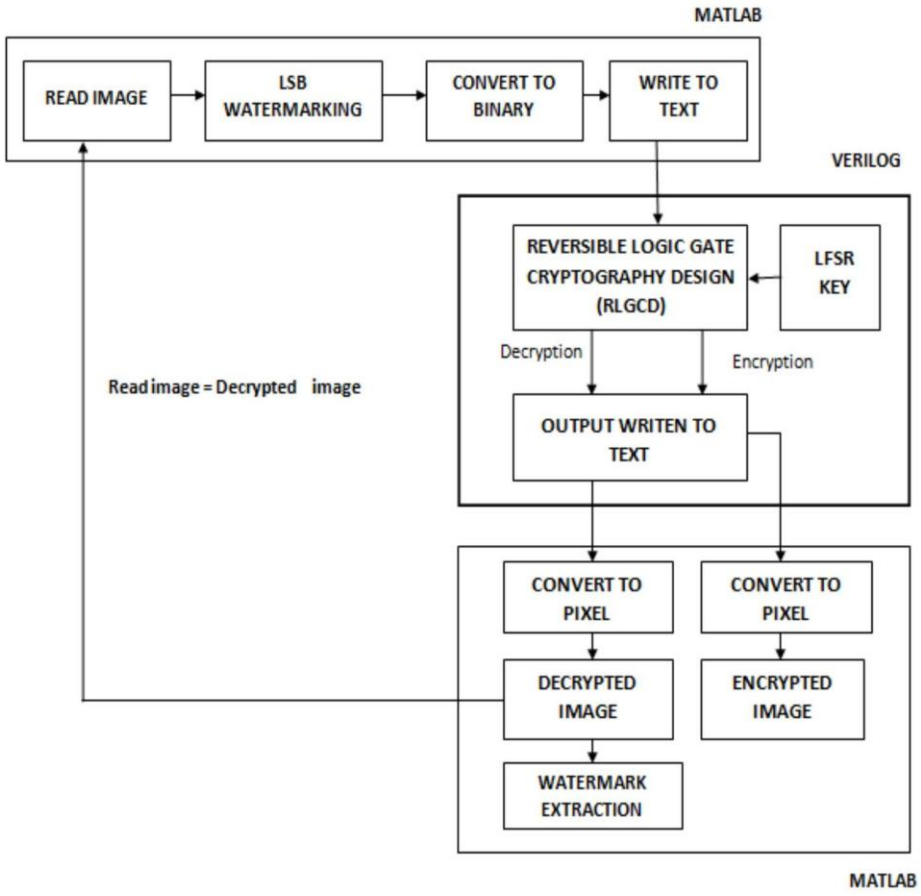


Fig 4.2.3 Block diagram

4.3 Encryption process:

The encryption process is shown in Fig below. The pixel values are thus 8 bit binary word: i[0], i[1], i[2], i[3], i[4], i[5], i[6], i[7]. The first four LSB input bits is applied to the below SCL gate and the above SCL gate is fed by the first four MSB.

LFSR:

LFSR is central to generating pseudo-random sequences for encryption.It operates by shifting and XORing bits to create sequences used as encryption keys. In decryption, the same LFSR with the same initial state regenerates the key sequence, allowing for reversal of encryption. LFSR's design in VLSI considers efficiency and reversibility for successful implementation.

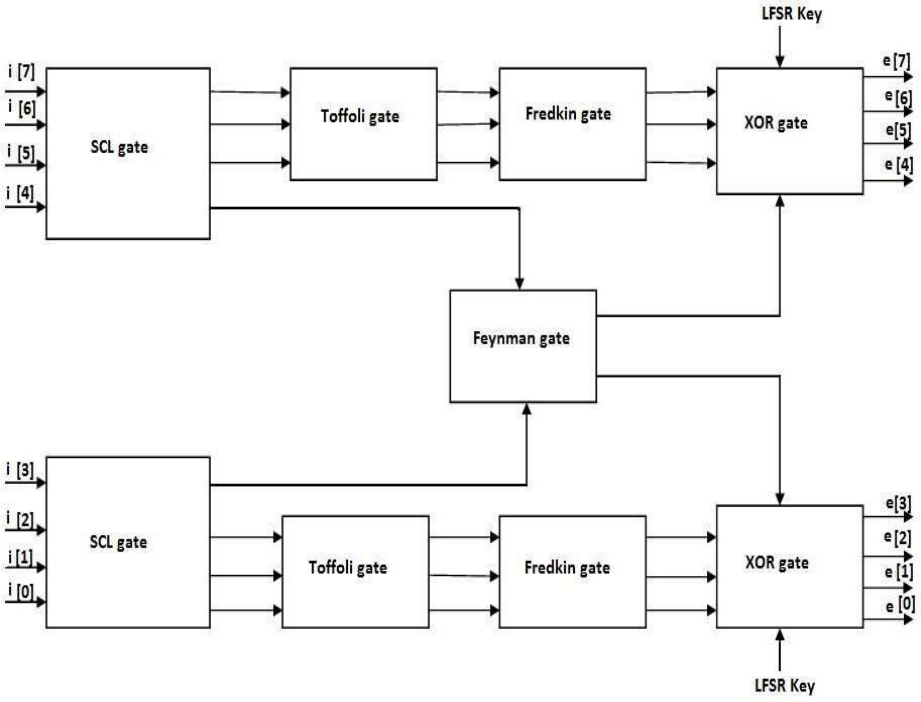


Fig. 4.3.1 Encryption block

Four of these inputs complete the SCL gate operation and thus produce four result bits. The first three LSB outputs from the below SCL gate perform Toffoli gate operation and provides three different output bits. Similarly, thefirst three MSB value outputs of SCL gate feed Toffoli gate and provides three output bits. One of the output bits from the above and below SCL gates perform Feynman gate operation. Both Toffoli gates are followed by Fredkin gate and thus its outputs perform Fredkin gate. The Fredkin gate outputs and the Feynman gate outputs are connected to the XOR gates and thus perform XOR operation with LFSR key. Then, the XOR gate output provides the encrypted binary image pixel value e[0], e[1], e[2], e[3], e[4], e[5], e[6], e[7].

4.4 Decryption process:

The process of decryption is shown in Fig.3. The decryption process is just the reverse operation of the encryption. Thus, encryption process output is fed as input to decryption process block. First, the encrypted pixel bits perform XOR operation with the key generated by the LFSR. After performing the four reversible gate operation one followed by the next the decrypted outputs are obtained at the SCL gate output. The decrypted output eight bit pixel values are d[0], d[1], d[2], d[3], d[4], d[5], d[6], d[7]. The encrypted as well as the

decrypted binary output values are written into a text file. In MATLAB encrypted image and decrypted image are generated from the output text file.

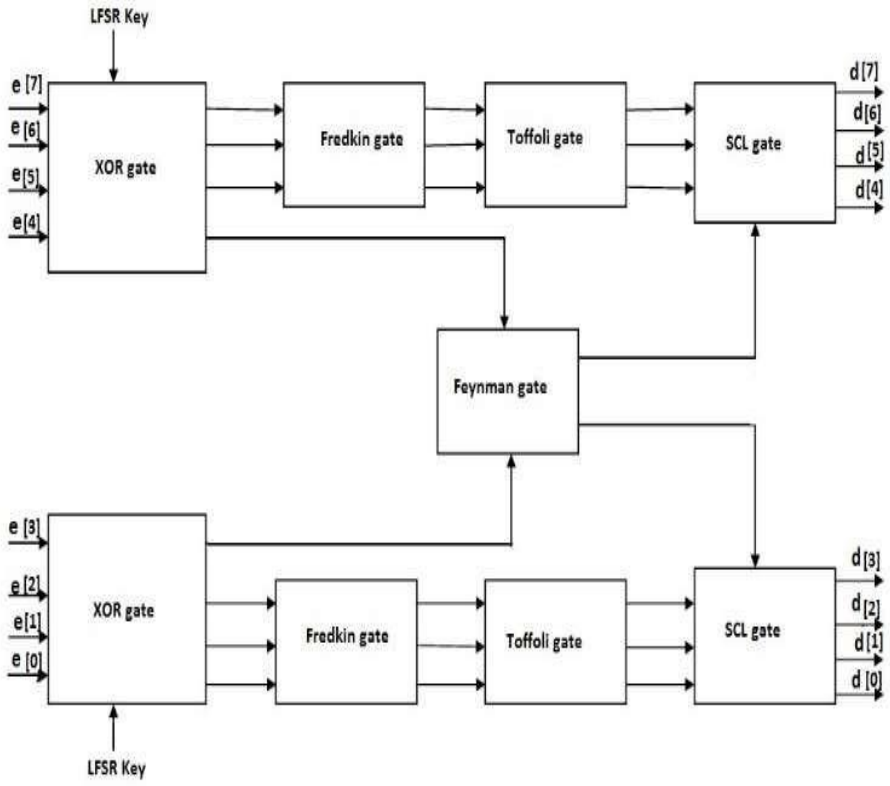


Fig 4.4 Decryption block

4.5 ADVANTAGES OF PROPOSED SYSTEM:

Low power consumption: Reversible logic gates consume less power than conventional logic gates because they do not lose information during computation. This is because reversible logic gates use feedback to generate their outputs, which allowsthem to reuse the same inputs multiple times.

Reduced heat dissipation: Reversible logic gates dissipate less heat than conventional logic gates because they do not generate garbage outputs. Garbage outputs are outputs that are not needed for the computation, but are generated anyway. Conventional logic gates can generate garbage outputs, which can lead to increased heat dissipation.

Improved security: Reversible logic gates can be used to implement more secure encryption and decryption algorithms than conventional logic gates. This is because reversible logic gates are more resistant to power analysis attacks. Power analysis attacks are a type of attack that can be used to extract secret

information from a system by analyzing its power consumption. Reversiblelogic gates are more resistant to power analysis attacks because they do not generate garbage outputs, which can be used by attackers to extract secretinformation.

Suitability for quantum computing: Reversible logic gates are well-suited for implementation in quantum computers. This is because reversible logic gates are unitary operations, which are the building blocks of quantum computation. Unitary operations are operations that preserve the information content of a quantum state. Reversible logic gates are unitary operations, which means that they can be used to implement quantum encryption and decryption algorithms.

4.6 COMPARISON BETWEEN REVERSIBLE AND IRREVERSIBLEGATES:

Table 4.6 Comparison between Reversible and Non-Reversible Gates

PARAMETERS	**VERSIBLE**	**REVERSIBLE**
Reversibility	n be reversed to ain the original ut from the put	nnot be fectly reversed obtain original ut from the put
Information loss	information s	volves ormation loss
Energy Consumption	nsumes low ergy	nsumes more ergy
Application Domain	antum mputing, nano hnology, ellite mmunication .	gical and thmetic mputations.
Examples	ffoli gate, dkin gate, ynman gate etc.	ND gate, OR e, NOT gate, OR gate etc.

4.7 Software Requirements:

4.7.1 XILINX

It requires Xilinx ISE 14.7 version of software where Verilog source code canbe used for design implementation.

Introduction to XILINX ISE:

This instrument can be utilized to make, execute, reenact, and integrateVerilog outlines for usage on FPGA chips.

ISE: Instigated Software Environment

Environment for the improvement and trial of computerized systemsconfiguration focused to FPGA or CPLD

Integrated gathering of apparatuses available through a GUI

Based on an intelligent combination motor (XST: Xilinx Synthesis Technology)XST underpins diverse dialects:

VerilogVHDL

XST create a net rundown incorporated with requirements Supports every one of the means required to finish the plan:Translate, guide, place and course

Bit stream era

For this situation, it is conceivable to utilize Verilog to compose a test seat to confirm the usefulness of the outline utilizing documents on the host PC to characterize jolts, to interface with the client, and to contrast comes about and those normal.

A Verilog show is converted into the "doors and wires" that are mapped onto a programmable rationale gadget, for example, a CPLD or FPGA, and after that it is the real equipment being designed, instead of the Verilog code being "executed" as though on some type of a processor chip.

4.7.2 Implementation:

Synthesis (XST)

-Produce a netlist file starting from an HDL descriptionTranslate (NGD Build)

Converts all input design netlists and then writes the results into a single merged

file, that describes logic and constraints.Mapping (MAP)

Maps the logic on device components.

Takes a netlist and groups the logical elements into CLBs and IOBs(components of FPGA).

Place And Route (PAR)

Place FPGA cells and connects cells.Bit stream generation

4.7.3 XILINX Design Process:

Step 1: Design entry

HDL (Verilog or VHDL, ABEL x CPLD), Schematic Drawings, BubbleDiagram

Step 2: Synthesis

Translates. v, .vhd, .sch files into a netlist file (.ngc)

Step 3: Implementation

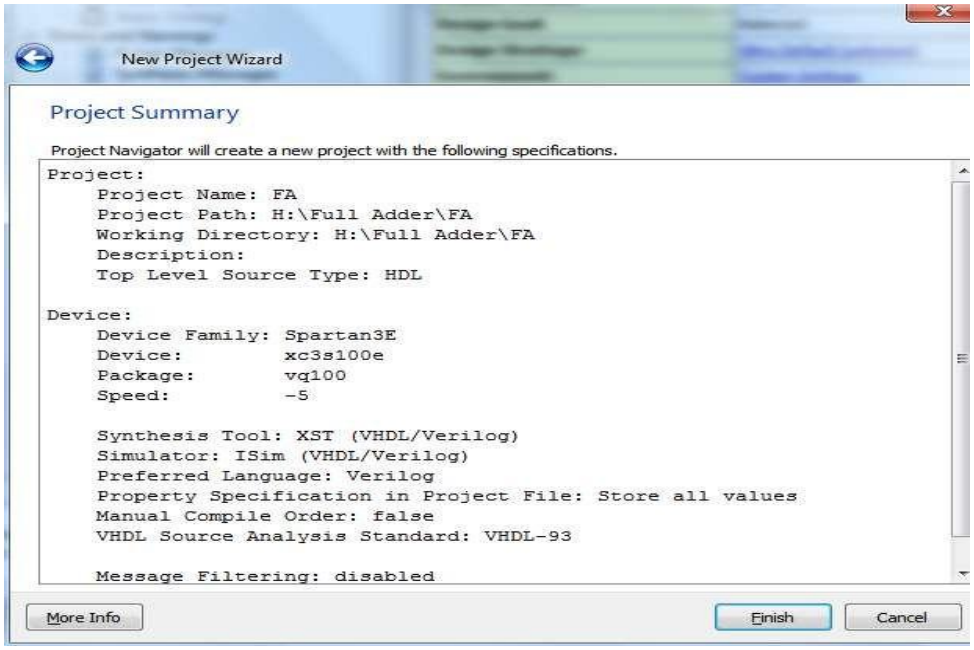
FPGA: Translate/Map/Place & Route, CPLD: Fitter

Step 4: Configuration/Programming Download a BIT file into the FPGA Program JEDEC file into CPLD Program MCS file into Flash PROM Simulation can occur after steps 1, 2, 3

The tools used in this thesis are **XILINX ISE 14.7** for simulation and Synthesis.The programs are written in Verilog language.

Xilinx Tools is a suite of software tools used for the design of digital circuits implemented using Xilinx Field Programmable Gate Array (FPGA) or Complex Programmable Logic Device (CPLD). The design procedure consists of (a) design entry, (b) synthesis and implementation of the design, (c) functional simulation and (d) testing and verification. Digital designs can be entered in various ways using the above CAD tools: using a schematic entry tool, using a hardware description language (HDL) – Verilog or VHDL or a combination of both. In this thesis we will only use the design flow that involvesthe use of Verilog HDL.

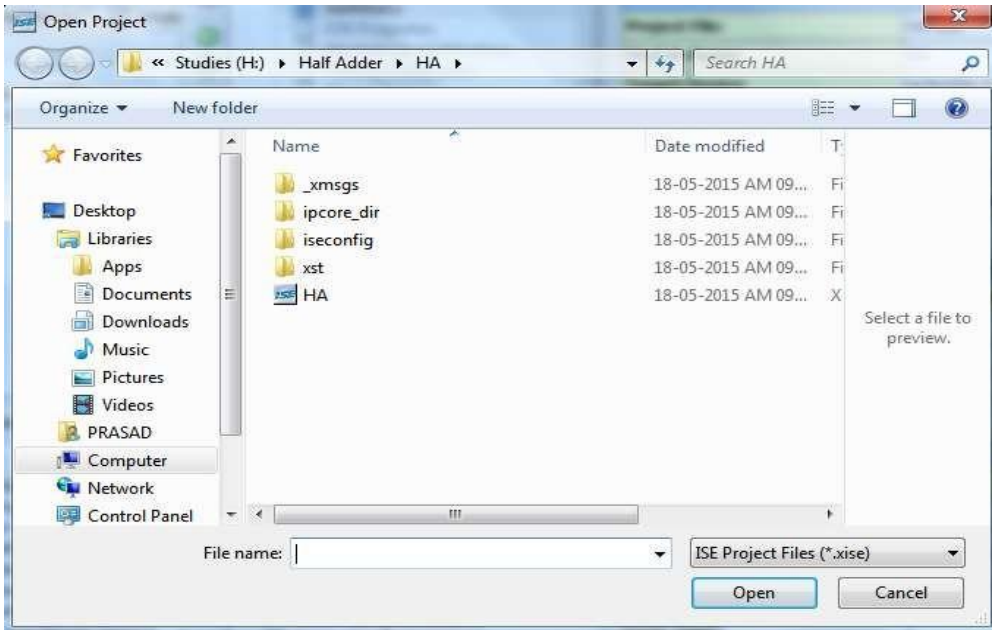
Creating a New Project



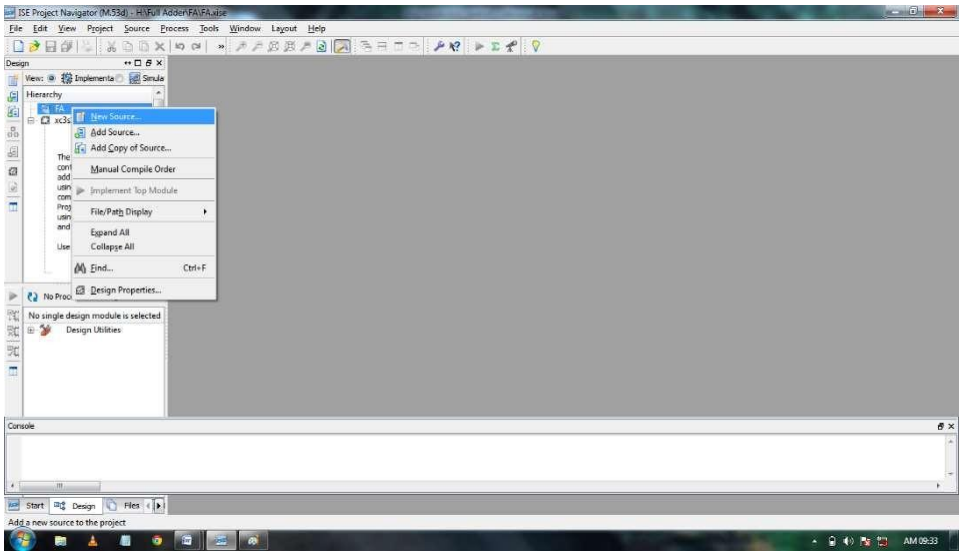
A project summary window is opened click on finish.

In order to open an existing project in Xilinx Tools, select **File->Open Project** to show the list of projects on the machine. Choose the project you want and click **OK.**

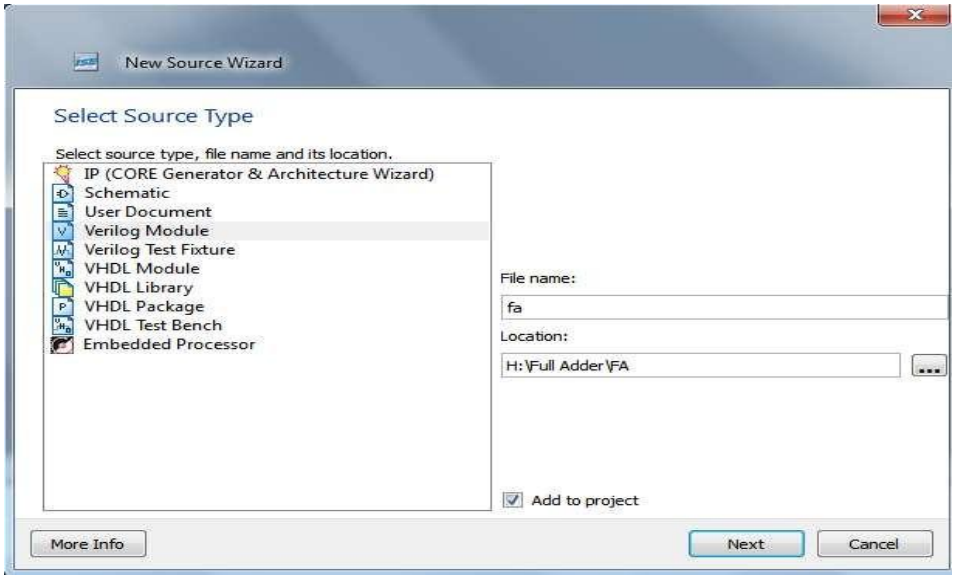
Clicking on NEXT on the above window brings up the following window:



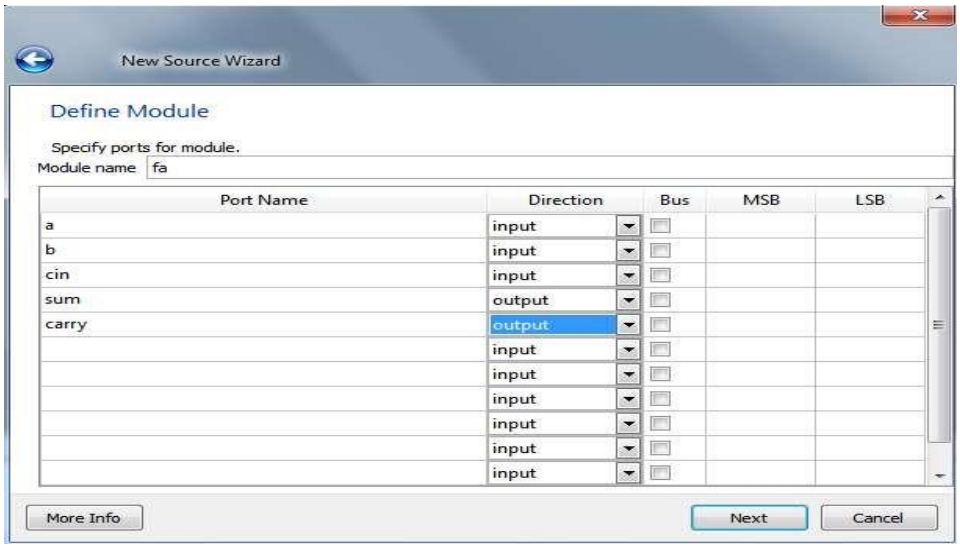
If creating a new source file, Click on the NEW SOURCE.



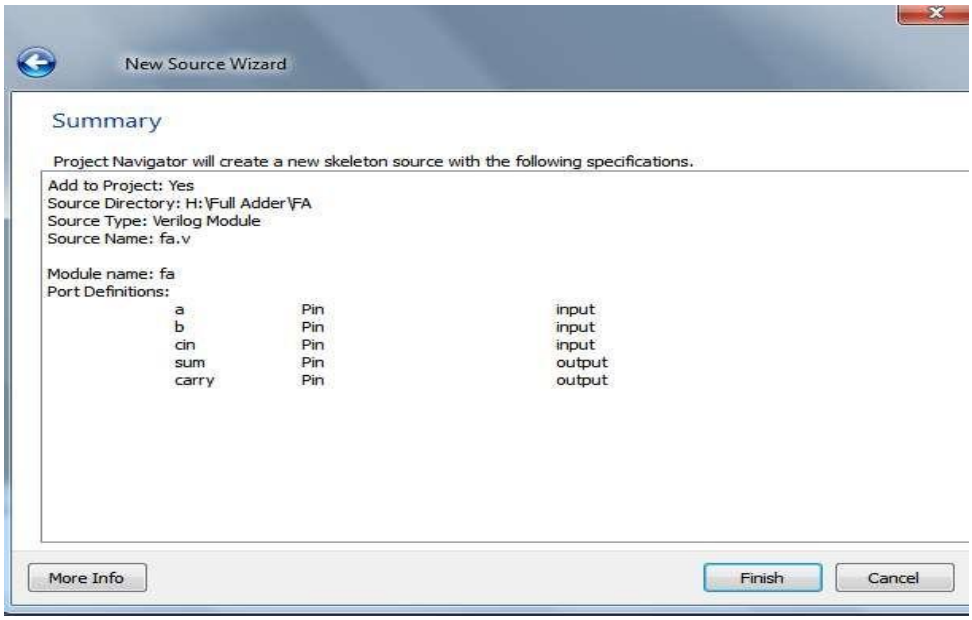
A window pop up is opened.



Select **Verilog Module** and in the "File Name:" Enter the name of the Project.Then click on **Next** to accept the entries. This pops up the following window.

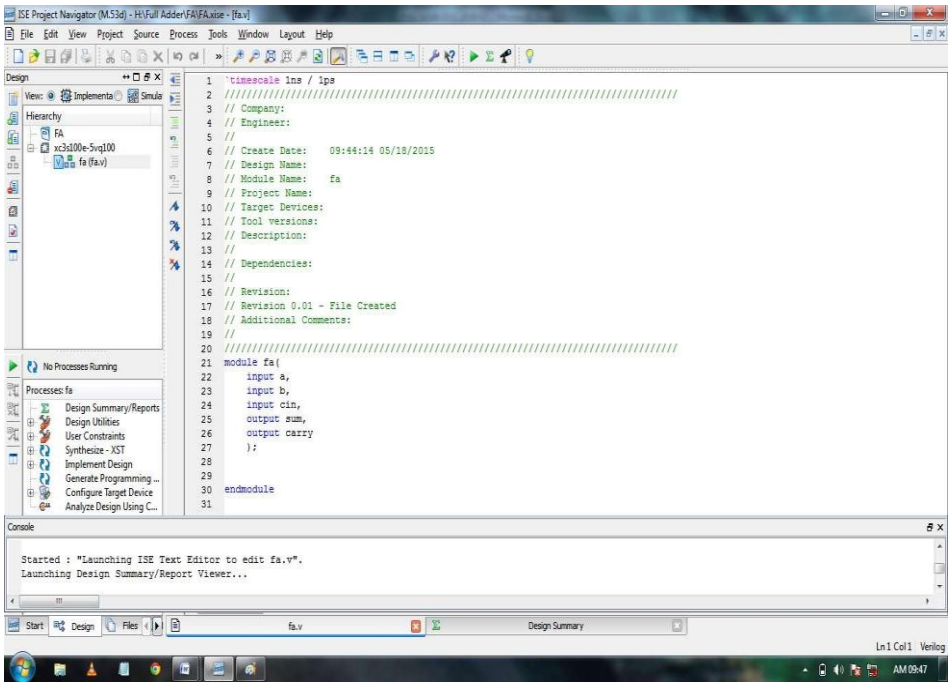


In the **Port Name** column, enter the names of all input and output pins and specify the **Direction** accordingly. A Vector/Bus can be defined by entering appropriate bit numbers in the **MSB/LSB** columns. Then click on **Next>**to geta window showing all the new source information.

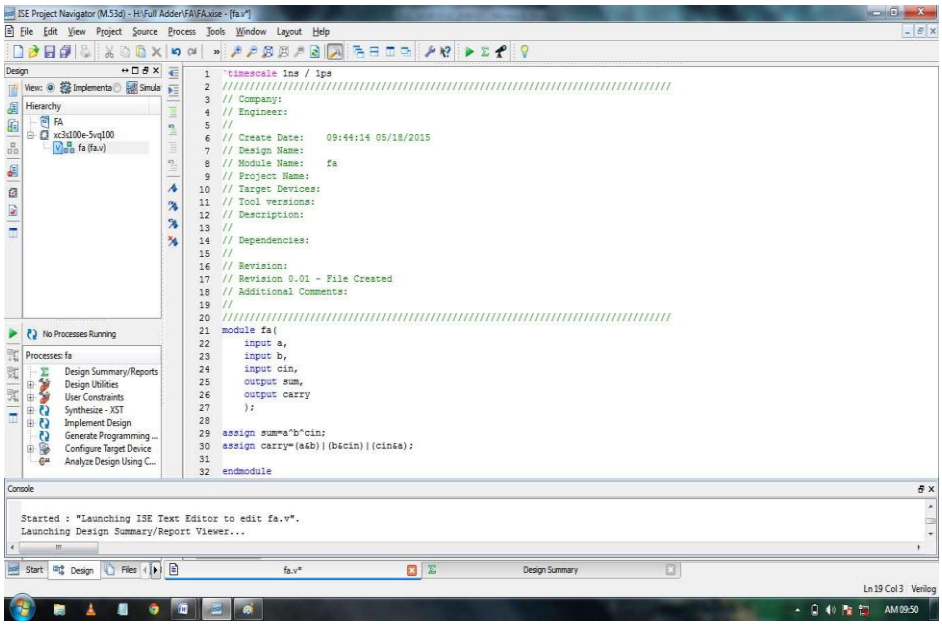


Click on **Finish** to continue.

The source file will now be displayed in the **Project Navigator** window.



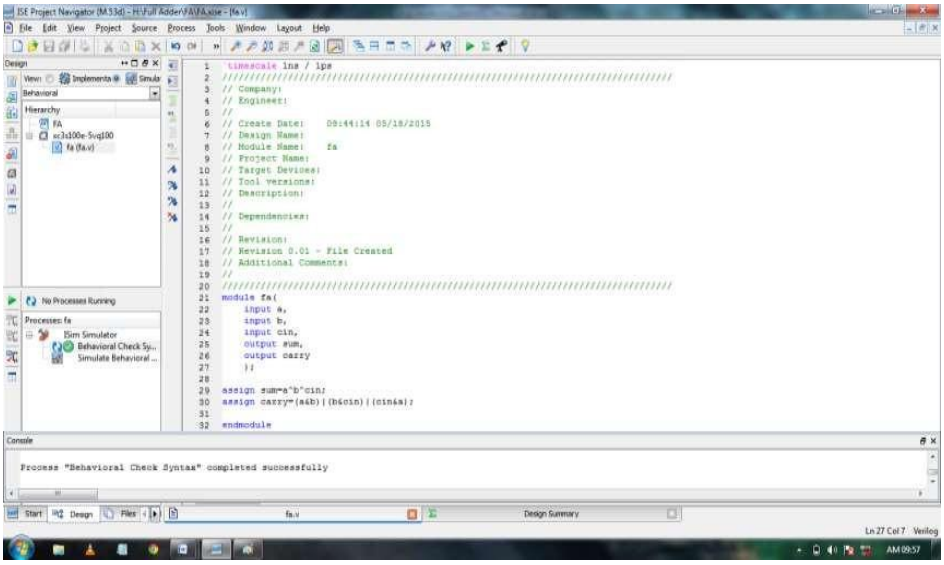
The source file window can be used as a text editor to make any necessary changes to the source file. All the input/output pins will be displayed. Save your Verilog program periodically by selecting the **File->Save** from the menu. You can also edit Verilog programs in any text editor and add them to the project directory using "Add Copy Source".



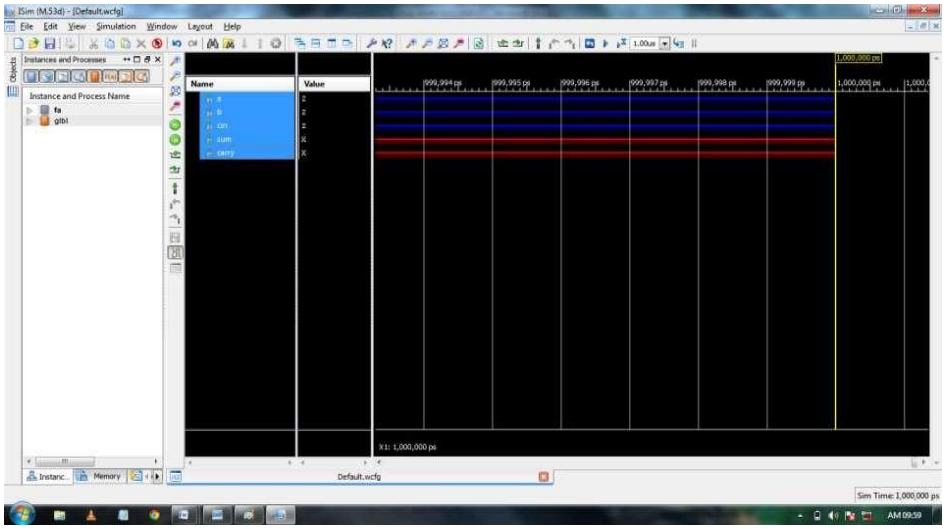
4.7.4 Simulating and Viewing the Output Waveforms:

Click on simulation select the existing file and expand ISim Simulator and

click on Behavioral check syntax to check the Errors.

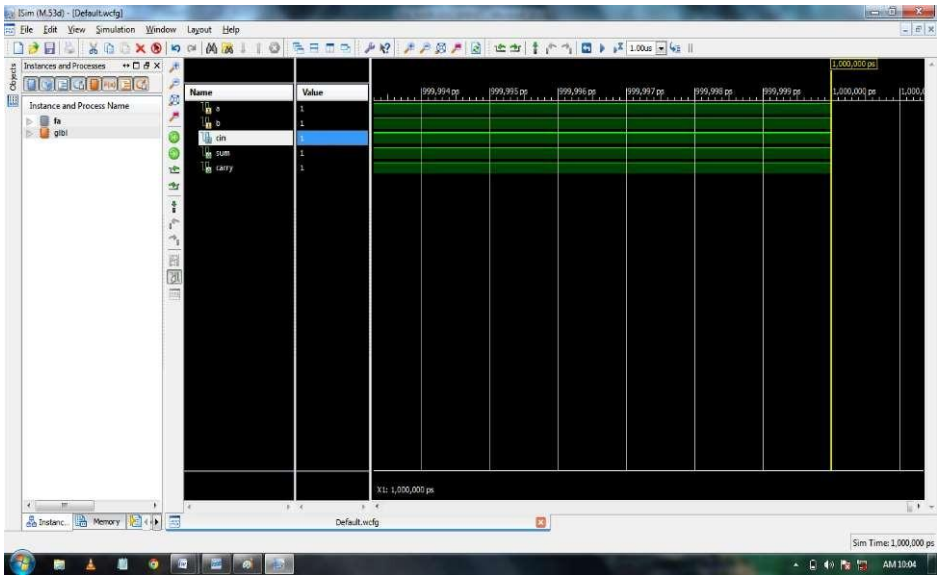


If there are no errors click on simulate behavioral model. A window pop up isopened.



Here we can give the inputs. Right click on the selected input click on forceconstant and enter the input value click on Ok.

Click on Run option in the tool bar to check input and output waveforms.



Synthesis and Implementation of the Design:

Click on Implementation select the existing file and double click on Synthesize- XST. If there are errors correct it. If there are no errors click on design summary and reports.

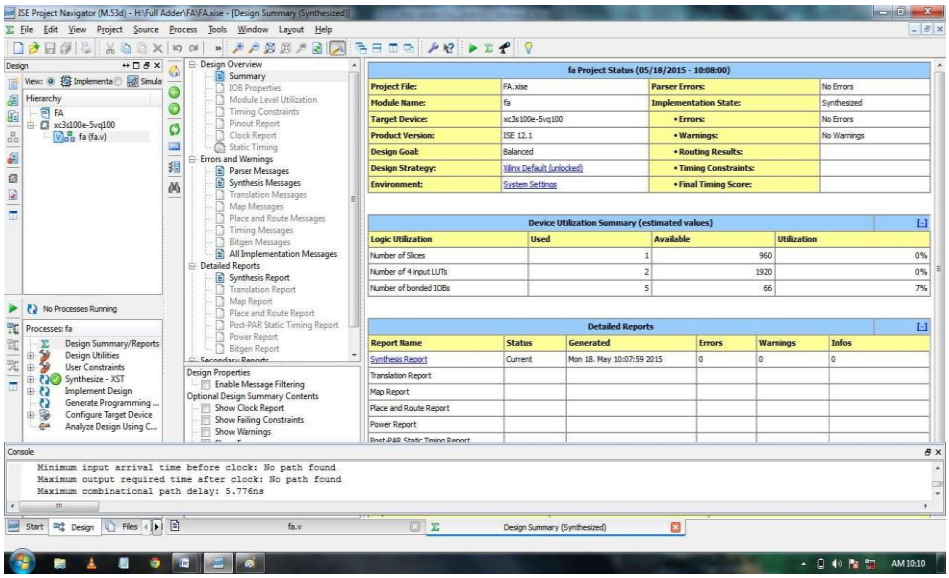


Fig 4.6.3.1 Device Utilization Summary

Open the Synthesis Report in the Detailed Reports to see the Device Utilization Summary and Timing Report of the current project.

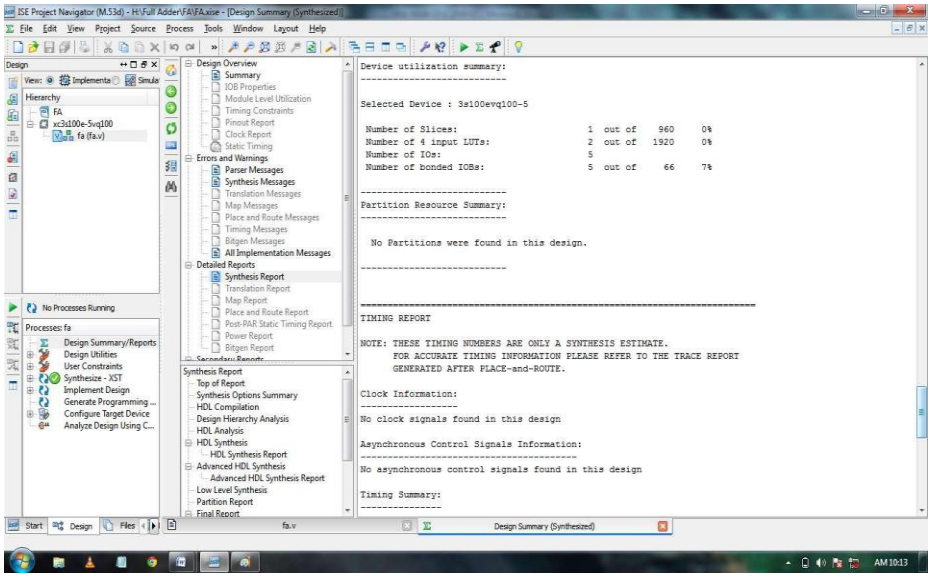


Fig 4.6.3.2 Synthesis Report

View RTL Schematic:

Expand Synthesize-XST and click on view RTL Schematic and click ok.

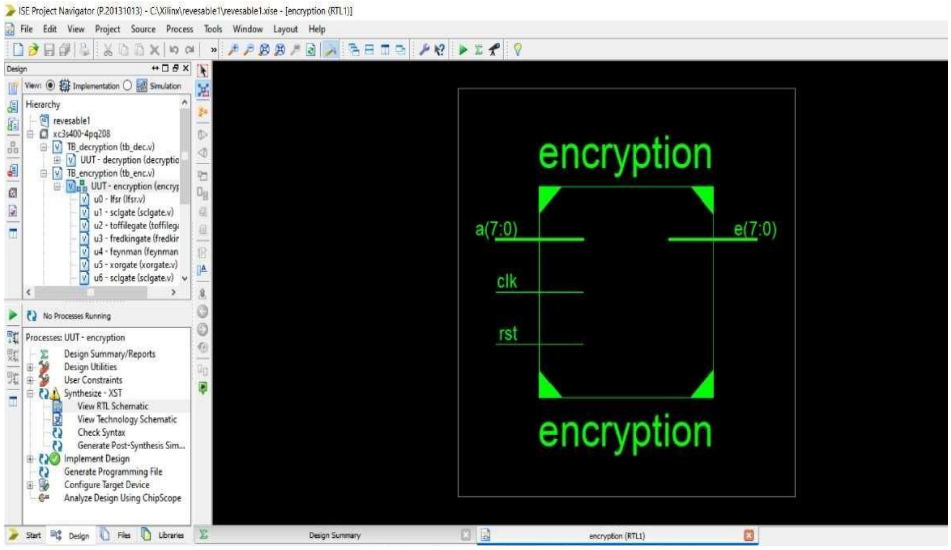


Fig 4.6.3.3 RTL Schematic

The window with Top module is opened to view the internal modules click onthe top module.

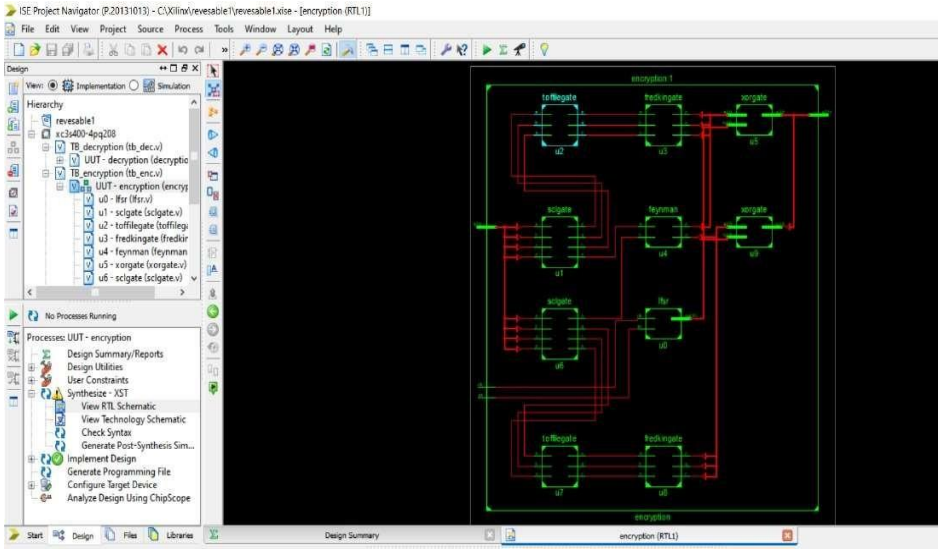


Fig 4.6.3.4 Internal modules

CHAPTER-5

RESULTS AND CONCLUSION

5.1 Software Results:

For image encryption and decryption using reversible logic gates, thefollowing results were obtained.

1.Reading an input image:

The input image is read by the MATLAB.

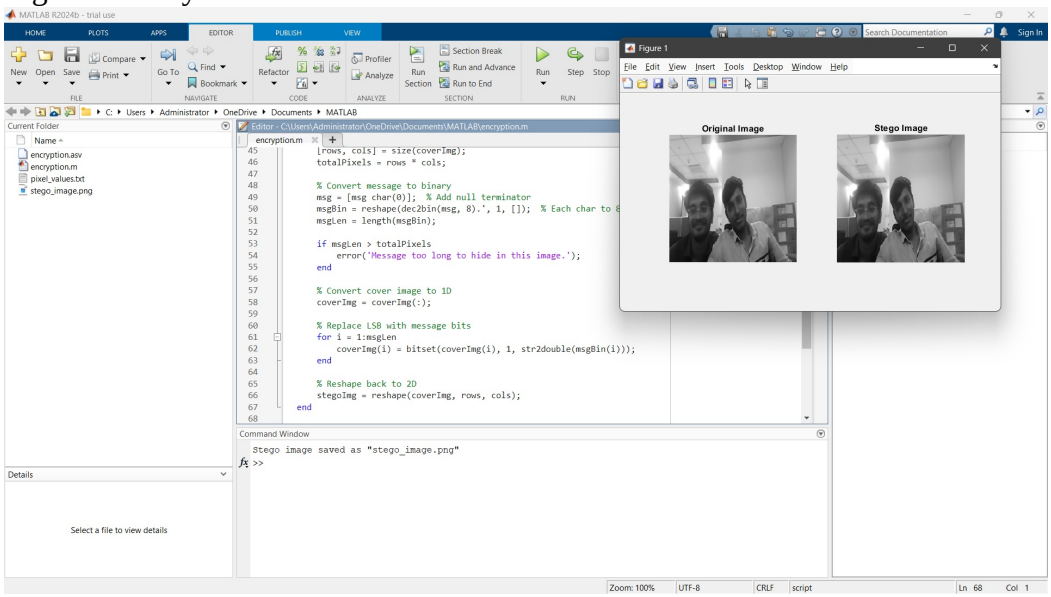


Fig 5.1: Reading Input Image

2.LSB Watermarking:

In LSB watermarking, the LSB of the original image pixel bits are changed by the bits of watermarking data. The changes thus created cannot be find out by human visibility system. First the length of the binary watermark data is embedded in the third and fourth LSBs of the first eight pixels with a gap of 5 pixels. So, the maximum length of binary watermark is 817 and thus it will askus to rewrite the data if the length is more than 817.

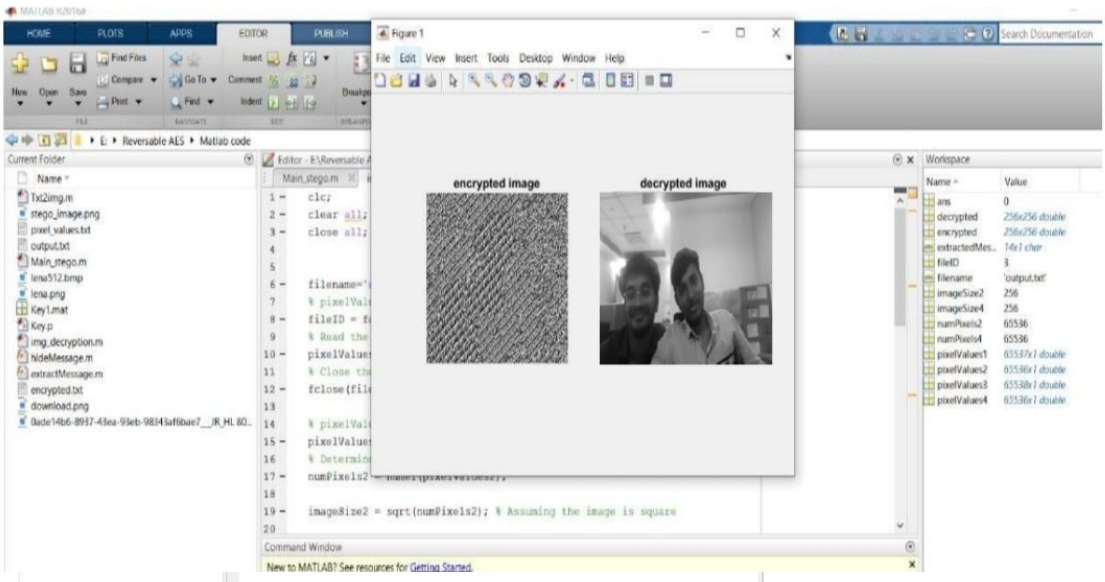


Fig 5.2 LSB Watermarking

3.RTL Schematic:

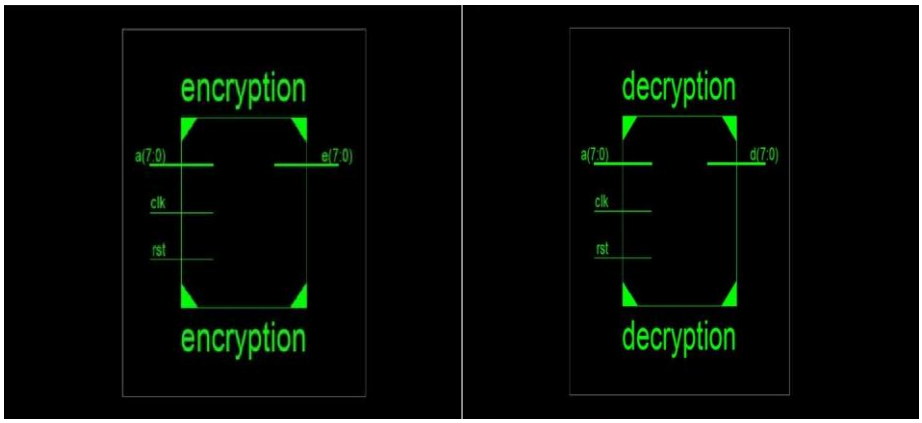


Fig 5.3 Encryption and Decryption blocks

4.Block Diagram:

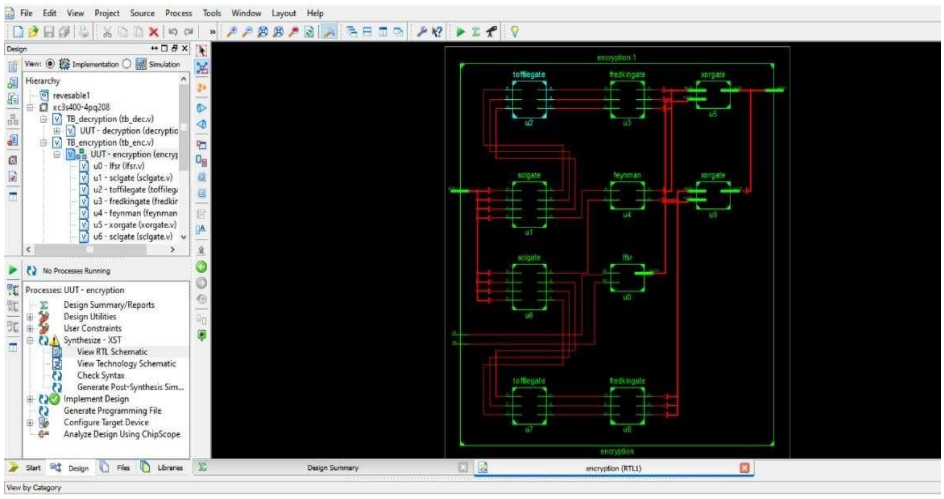


Fig 5.4 Block Diagram of Encryption

5.Encrypted and Decrypted Image Output:

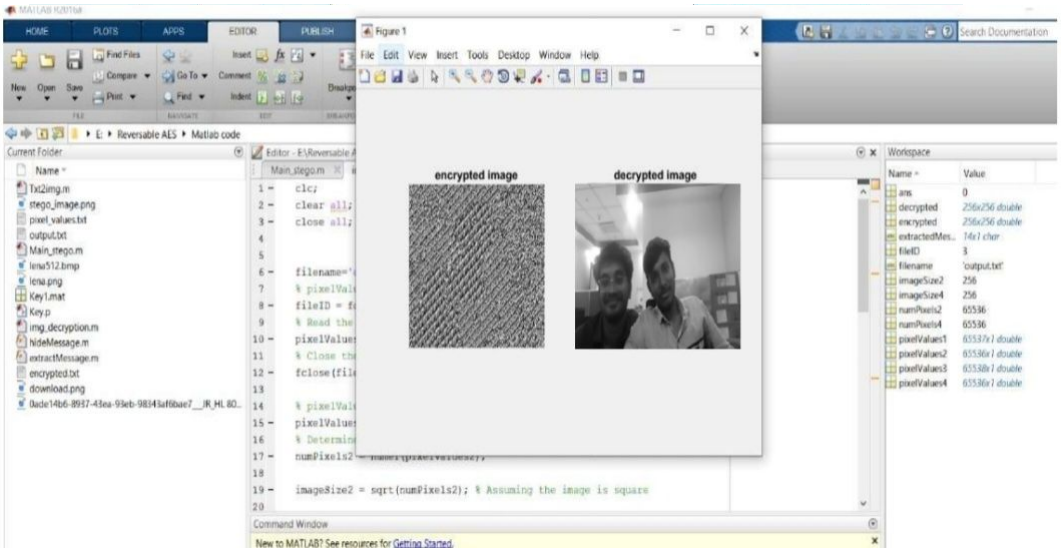


Fig 5.5 Encrypted and Decrypted Image Output

6.Synthesis Report

6.1 Delay Report:

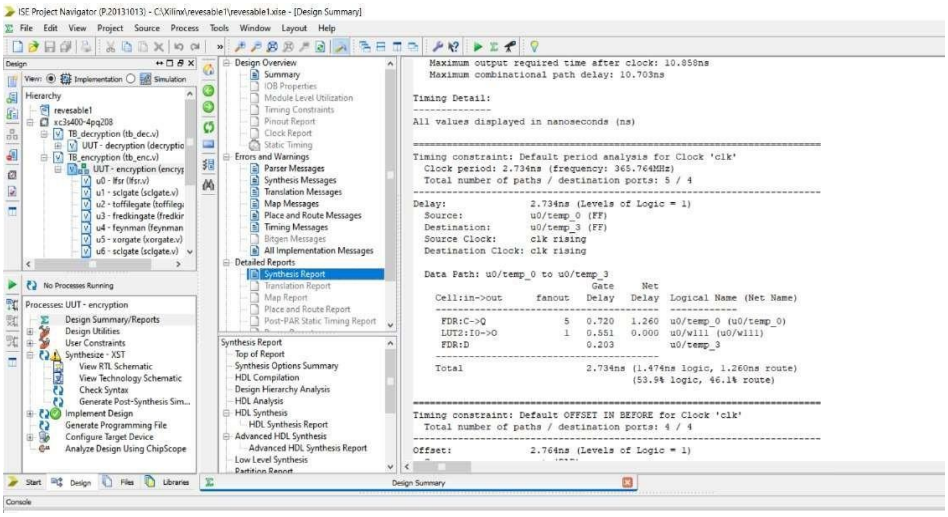


Fig 5.6: Delay Report

6.2 Device Utilization Summary:

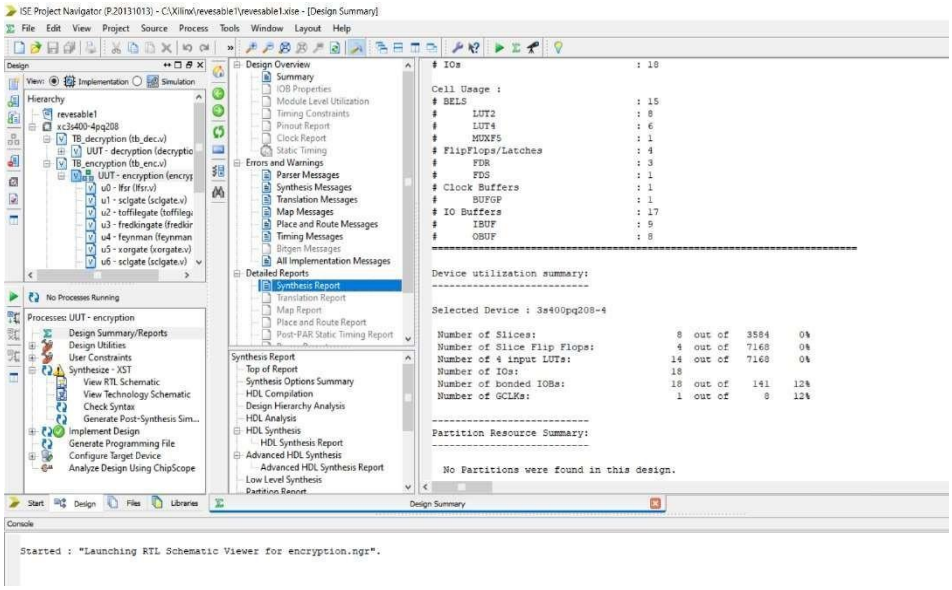


Fig 5.7: Device Utilization Summary

7. Comparison between Existing System and Proposed System:

parameters	Existing System	Proposed System
Delay Report	3.08ns	2.734ns
Area Report	69 LUTS	12 LUTS
Power	25mw	17.50mw

Table 5.1.7.1 Comparison between Existing System and Proposed System

5.2 APPLICATIONS:

Reversible logic gates have several applications, particularly in emerging fields such as quantum computing and low-power computing. Here are some notable applications:

Quantum Computing: Reversible logic gates are fundamental in quantum computing circuits because they preserve quantum information. Quantum algorithms often rely on reversible operations to ensure that computations canbe run both forwards and backwards, which is crucial for error correction and optimization.

Low-Power Computing: In conventional computing, irreversible gates dissipate energy by generating heat when information is lost. Reversible gates, however, can potentially reduce energy consumption by minimizing information loss, making them suitable for low-power computing applications, such as in portable devices and IoT devices.

Cryptography: Reversible logic gates play a role in cryptographic algorithms, particularly in designing secure hash functions and encryption schemes. The reversibility property ensures that information can be securely encoded and decoded without loss.

Data Compression: Reversible logic gates are used in data compression algorithms to ensure lossless compression, where the original data can be perfectly reconstructed from the compressed form. This is important in applications where preserving data integrity is critical, such as medical imaging and scientific data analysis.

Signal Processing: Reversible logic gates can be utilized in signal processing

applications, such as in digital filters and waveform analysis, where the reversibility property helps maintain signal integrity and minimize distortion.

Nanotechnology: In nanoscale computing, where power consumption and heat dissipation are significant concerns, reversible logic gates offer a promising solution due to their potential for low-power operation and minimal information loss.

Fault-Tolerant Computing: Reversible logic gates are also used in fault- tolerant computing systems, where the ability to reverse operations allows for error correction and recovery without data loss.

Overall, reversible logic gates have diverse applications across various fields, driven by their ability to efficiently manipulate information while preserving its integrity.

CHAPTER-6

6.1 CONCLUSION AND FUTURE SCOPE

CONCLUSION:

This work presents a Reversible Logic Gate Cryptography Design using LFSR key with watermarking. The reversible gates like Feynman, Fredkin,Toffoli and SCL gates are used in this new cryptography system design. Sincea cryptography system demands not only high security but low power consumption this work is one of the best among existing systems. those binary values are written into a text file. This input pixel values are read using Xilinx ISE. The RLGCD architecture consisting of LFSR, encryption block and decryption block is implemented in Xilinx software.

This architecture is suitable for both gray scale images and color images. The watermarking using LSB technique is performed to improve the security of the data. The Xilinx performance result for Spartan3E XC3S500E device gives a far better performance as compared to other existing systems. The reversible logic gates are the fundamental requirement in the emerging field of quantum computation. Thus, each work using the reversible logic gates will help to move forward in the field of quantum logic.

FUTURE SCOPE:

Since the successful implementation of Reversible Logic Gates-based Image Encryption and Decryption (RLGCD) using Verilog code, there is a strong potential for its future deployment on Application-Specific Integrated Circuits (ASICs). This suggests that the encryption and decryption process,already proven effective in a digital simulation environment, can be translated into dedicated hardware, potentially offering enhanced performance andefficiency.

BIBLIOGRAPHY

1) Gordon E. Moore, "Cramming more components onto integrated circuits," Electronics, pp14-117, April 1965.

2) Rolf Landauer, Irreversible and heat generation in the computing process, IBM Research and Development, vol.5, pp.183–191, July 1961.

3) C.H. Bennett, "Logical reversibility of computation" IBM Research and Development, vol.17, pp.525–532, 1973.

4) Saranya Karuna Murthi, Vineya Kumar Krishnasamy Natarajan, "VLSI implementation of reversible logic gates cryptography with LFSR key," Microprocessors and Microsystems, Elsevier, vol. 69, pp.68–78, September 2019.

5) Mehran Mozaffari Kermani, Kaj Reza Azarderakhsh, Siavash Bavat Sarmadi, "Fault resilient lightweight cryptography block cipher for secure embedded systems," in IEEE Embedded System Letters, vol. 6, no. 4, pp.89–92, Dec.2014.

6) Shikha Kuchhal, Rakesh Verma, "Security design of DES using reversible logic," Int. J. Comput. Sci. Netw. Security, vol. 15, no. 9, pp. 81–84, September 2015.

7) Z. H. A. O. Guo sheng, W. A. N. G. Jain, "Security analysis and enhanced design of a dynamic block cipher," China Commun., vol. 13, pp. 15– 160, January 2016.

8) Srivatsam Subramanian, Mehran Mozaffari Kermani, Reza Azarderakhsh, Mehrdad Nojoumaian, "Reliable hardware architectures for cryptographic block ciphers LED and HIGHT," in IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., vol. 36, no.10, pp. 1750-1758, Oct. 2017.

9) Raghava Garipelly, P. Madhu Kiran, A. Santhosh Kumar, "A review on reversible logic gates and their implementation," in International Journal of Emerging Technology and Advanced Engineering, vol. 3, no. 3, March 2013.

10) Abdullah Bamatraf, Rosziati Ibrahim, Mohd. Najib. B, Mohd. Salleh, "Digital watermarking algorithm using LSB," in 2010 International Conference on Computer Applications and Industrial Electronics, Kuala Lumpur, pp. 155-159, 2010.

11) Meenal Dadhe, Prof. Anup. R. Nage, "Design of high-speed VLSI architecture for LFSR with maximum length feedback polynomial," in International Journal for Scientific Research & Development, vol .3, no. 5, 2015.

12) Y. G. Praveen Kumar, B. S. Kriyappa, M. Z. Kurian, "Implementation of power efficient 8-bit reversible linear feedback shift register for BIST," in 2017 International Conference on Inventive Systems and Control, Coimbatore, 2017.

13) B. Koziel, R. Azarderakhsh, M. Mozaffari Kermani, D. Jao, "Post- SEMANTIC SEGMENTATION AND CLASSIFICATION OF LIVER CANCER FROMMRI IMAGES Dept of E.C. E, SRIT Page 54 quantum cryptography on FPGA based on isogenies on elliptical curve," in IEEE Trans.Circuits Syst.I, vol. 64, no. 1, pp. 86–99, Jan. 2017.

14) B. Koziel, R. Azarderakhsh, M. Mozaffari Kermani, "A high performance and scalable hardware architecture for isogeny-based cryptography," in IEEE Trans.Comput., vol. 67, no. 11, pp. 1594–1609, Nov,2021.

15) H. Zodpe, A. Sapkal, "An efficient AES implementation using FPGA withenhanced security features," in J. King Saud Univ.Eng.Sci., 2022, in press.