A

PROJECT REPORT ON

CHIP IMPLEMENTATION OF LOW POWER ENCRYPTION USING RSA ALGORITHM

SUBMITTED TO SAVITRIBAI PHULE PUNE UNIVERSITY
FOR PARTIAL FULFILLMENT
OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE OF

BACHELOR OF ENGINEERING
In
Electronics and Telecommunication Engineering

By

APOORVA KUMAR	**B400050016**
MANSI DANGADE	**B400050055**
PRANAV INDURKAR	**B400050095**

GUIDE
MR. H. S. THAKAR

DEPARTMENT OF
ELECTRONICS AND TELECOMMUNICATION ENGINEERING
PUNE INSTITUTE OF COMPUTER TECHNOLOGY
PUNE – 43

Department of Electronics and Telecommunication Engineering
Pune Institute of Computer Technology, Pune – 43

CERTIFICATE

This is to certify that the Project Report entitled

CHIP IMPLEMENTATION OF LOW POWER ENCRYPTION USING RSA ALGORITHM

has been successfully completed by

APOORVA KUMAR B400050016
MANSI DANGADE B400050055
PRANAV INDURKAR B400050095

Is a bona fide work carried out by them under the guidance of Mr. H. S. Thakar and it is approved for the partial fulfillment of the requirement of the Savitribai Phule Pune University, Pune for the award of the degree of the Bachelor of Engineering (Electronics and Telecommunication Engineering). This project work has not been previously submitted to any other Institute or University for awarding any degree or diploma.

Mr. H. S. Thakar	Dr. M. V. Munot	Dr. S. T. Gandhe
Guide	HOD, E&TCE Dept.	Principal, PICT

Place: Pune
Date : 17-04-2025

ACKNOWLEDGEMENT

We would like to extend our heartfelt gratitude to Dr. M. V. Munot, Head of the Department of Electronics and Telecommunications, for her unwavering support throughout our project.

We are also deeply thankful to Mr. H. S. Thakar for his continuous encouragement and mentorship, which played a crucial role in our project's success. His expertise has been invaluable to our work.

Our appreciation goes to Mr. N. G. Nirmal, Mr. N. B. Patil, Dr. R. C. Jaiswal, and Mr. S. D. Hake for their assistance and valuable insights that significantly contributed to our work.

Furthermore, we acknowledge the contributions of various authors and sources whose research provided a strong foundation for our study.

Thank you all for your support and encouragement.

Thanking you,

Apoorva Kumar B400050016

Mansi Dangade B400050055

Pranav Indurkar B400050095

CONTENTS

ABSTRACT

As the demand for secure communication and reliable data transmission grows, encryption techniques like RSA are critical for ensuring the confidentiality and integrity of digital information. However, RSA's strong security comes with high computational and power demands, especially in resource-constrained environments. This project investigates the implementation of a low-power RSA encryption circuit, with a focus on optimizing both resource utilization and power consumption without compromising the algorithm's cryptographic strength.

The RSA algorithm was implemented using Verilog HDL and prototyped on the Spartan-7 FPGA (XC7S50-CSGA324-2) using the Xilinx Vivado platform. A comparative analysis was conducted for multiple configurations of prime bit sizes (4-bit to 8-bit for p, q, and message M). Two design strategies were explored, with the second approach demonstrating improved resource efficiency and reduced power consumption, making it a more suitable option for low-power embedded applications.

Additionally, ASIC-level simulations were performed using Cadence Genus and Innovus, where key physical design flows such as synthesis, floorplanning, placement, routing, and GDS-II generation were executed. This allowed for the evaluation of area, timing, and power characteristics of the design. The findings provide valuable insights into the development of low-power RSA-based encryption hardware, especially for use in IoT, mobile, and other resource-constrained digital systems where energy efficiency is crucial.

Abbreviations and Acronyms

RSA	Rivest Shamir Adleman
FPGA	Field programmable gate array
IoT	Internet of Things
VLSI	Very Large Scale Integration
IC	Integrated Circuit
EDA	Electronic Design Automation
RPT	Report

List of Symbols

p, q	Prime numbers
M	Original message
e	Encryption key/ public key
d	Decryption key/ private key

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Background

The RSA algorithm was introduced in 1977 and is one of the earliest and most secure public-key encryption systems. It works by generating a key pair: the **public key** (used to encrypt data) and the **private key** (used to decrypt it). RSA's security is based on the difficulty of factoring large numbers, a computationally hard problem [1].

Traditionally, RSA is implemented in software, which limits its use in devices that require:

- **Low power consumption**
- **Minimal hardware resources**
- **Fast real-time performance**

In modern electronics, especially in **smart embedded systems**, **wireless sensor networks**, and **portable devices**—there is a growing need for **hardware-based cryptographic engines**. These engines offer enhanced speed and energy efficiency compared to software solutions.

This project is developed keeping these goals in mind: to implement RSA as a hardware module that can perform encryption and decryption effectively with **optimized performance and reduced power**, suitable for integration into chips.

1.2 Relevance

This project has strong relevance to the field of **Electronics and Telecommunication Engineering** and directly connects to subjects such as:

- **Digital Logic and VLSI Design**
- **Hardware Security and Cryptography**
- **HDL (Verilog) Design and Simulation**
- **FPGA Development**
- **ASIC Design Methodology**

The process of modeling a cryptographic algorithm in Verilog, testing it in FPGA, and then analyzing it through an ASIC flow provides deep insight into the **real-world process of chip implementation**. It enhances technical skills in:

- Writing modular and testable Verilog code,
- Using simulation and synthesis tools like **Vivado** and **Cadence Genus**,
- Applying **power-aware RTL optimization techniques**, and
- Understanding the trade-offs in **chip design metrics** (power, area, timing).

Such experience is vital for careers in **VLSI**, **chip design**, **embedded systems**, and **hardware security domains** [32].

1.3 Literature Survey

The RSA algorithm has been extensively studied, with a focus on enhancing security while minimizing power consumption and optimizing resource usage. Several approaches to low-power encryption have been explored, particularly in FPGA-based implementations. Key research works include studies on RSA's performance on different hardware platforms, comparisons with other cryptographic algorithms, and efforts to reduce computational overhead while maintaining encryption strength. Below is a summary of key findings from the reviewed literature.

Table 1.3.1 : Literature Survey

Title	Authors	Findings	Advantages	Disadvantages
CMOS/ CNTF ET Circuits	Pendyala [14]-2023	This work addresses emerging security challenges in IoT, focusing on vulnerabilities like unauthorized access and data breaches. It proposes a three-module security system featuring User Authentication, a Lightweight RSA Key Exchange, and an additional security layer. The solution aims to enhance both performance and security, optimizing efficiency while fortifying defenses against evolving threats.	1.Reduced power consumption. 2.Improved performance. 3. Enhanced Scalability. 4.Longer battery life.	1.Increased Design Complexity 2. Higher Cost 3.Scalability Challenges 4.Leakage Power Issues
		The proposed audio cryptosystem demonstrated a high-quality performance with an SSNR of 0.3395 and favorable MSE values for		

Speech Process Using RSA.	Abouelkheir [15]-2022	decrypted audio signals. Comparative results revealed that it outperformed existing methods, achieving significantly higher SSNR values than those of Wahab and Mahdi (-14.54520) and Abd Elzaher et al. (-55.20000). This research, funded by Qassim University, was conducted with no competing interests declared by the authors.	1.Improved Processing Speed. 2.Enhanced Security. 3.Versatility.	1.Increased Complexity 2.Higher Computational Cost
RSA Encryption LSI	Satoh [17]-2021	A 1024-bit RSA encryption LSI was developed, integrating DES and MD5 functions, with the RSA accelerator core occupying an area of 4.9 mm^2. It achieves a maximum frequency of 45 MHz and operates on 1024 bits in 23 ms.	Used for higher security systems.	Encryption using 512 bit is not that safe.
RSA using VHDL and a Xilinx /FPGA	Saini [3]-2021	The AES encryption algorithm achieved a throughput of 1.2 Gbps on the SPARTAN-6 FPGA, utilizing about 25% of available LUTs and 30% of flip-flops. The design consumed approximately 150 mW during operation, reflecting efficient resource usage.	1.Higher performance. 2.Customizability 3. Reconfigurability	1.High Development time. 2. Less flexibility after deployment.
RSA Algorithm/Area	Thabah [18]-2019	The maximum frequency for RSA operations is 545 MHz for 8 bits & 298 MHz for 64 bits. Performance metrics for the RSA algorithm include propagation delay, power consumption, gates count & area measurement, with noted improvements over earlier implementations.	1.Speed Optimization 2. Efficient Resource utilization	1.Resource Constraints. 2.Less flexibility.
		RSA requires a 512-bit key, while ECC only needs		

Cryptographic in Smart City	Bukhari [19]-2021	106 bits, making ECC more efficient. RSA is 5:1 more resource-intensive than ECC at 512 bits and 10:1 at 2048 bits. ECC's discrete logarithm problem also enhances security for low-power devices, making key breaks infeasible.	The Elliptic curve algorithm is best among RSA, AES, and DES algorithms.	In ECC, the attacker can secretly intercept and alter communication between two parties.
RSA chip/systolic array-based architecture	Sun [20]-2016	Power dissipation for various designs is 307.8 mW, 232.5 mW, 122.9 mW, and two unknown values. The gate counts are 37.5K, 98.5K, 175.8K, 61.0K, and 49.8K. The die area is 3.9 × 3.9 mm² without DFT and 4.58 × 4.58 mm² with DFT.	1.High throughput. 2.Efficient hardware utilization.	1.Complex Design. 2.Increased development time.
Algorithm/cryptanalysis	Kota [10]-2022	The public key is (n = 1013119) and (e = 237905). The private key (d) is (17), found using the Continued Fraction Algorithm, with primes (P = 1117) and (Q = 907). The condition for (d) is (d < 31.725). Increasing (e) (e.g., (e = 5.1pq)) can reduce(d) further, potentially leading to (d < 1).	1. Asymmetric Encryption. 2.Digital Signatures.	1.Slow performance. 2.Large Key size. 3.Vulnerable to Cryptanalysis.
RSA/double secure approach	Al-Barazanchi [22]-2019	RSA performance results reveal that for 10 bytes, RSA 2k has MSE = 0.0627 (0.0406 sec) and RSA 3k MSE = 0.0608 (0.0512 sec). For 20 bytes, RSA 2k MSE is 0.0571 (0.0265 sec) and RSA 3k is 0.0500 (0.0420 sec). With 30 bytes, RSA 2k shows MSE = 0.0518 (0.0306 sec) and RSA 3k is 0.0498 (0.0352 sec). For 40 bytes, RSA 2k has MSE = 0.0468 (0.0330 sec) and RSA 3k MSE = 0.0452	1.Enhanced security. 2.Protection against specific attacks.	1.Increased Computational Overhead. 2.Larger key sizes.

		(0.0367 sec). For 50 bytes, RSA 2k reports MSE = 0.0422. All configurations have BER = 0 and correlation = 1.		
RSA/Runge-Kutta	Raj [11]-2020	RK-RSA outperforms RSA in execution time and throughput, showing lower average execution times and higher encryption rates. It also has a greater avalanche effect (56 vs. 50) for improved security and lower power consumption for better efficiency.	1.High Precision. 2.Parallel Processing. 3.Adaptability.	1.Potential Overhead. 2.Compatibility issues.
Time Reduction in RSA Algorithm/Dynamic Approach	Krishnadoss [24]-2024	The dynamic RSA approach outperforms traditional RSA in encryption, decryption, and total execution time. Regression analysis shows traditional RSA has a slope of 1.75 and intercept of 1001.42, while dynamic RSA has a slope of 0.0113 and intercept of 28.58. This method significantly reduces computational overhead, especially for larger data volumes.	1. Reduced Processing time. 2. Adaptive resource utilization. 3. Scalability.	Dependence on accurate data analysis.
RSA Cryptosystem /Verilog	Hva [9]-2011	The random number generator produced 1091, 5455, and 20863. The primality tester confirmed 37 as prime, and the GCD for A=72 and B=5 gave public key e=5. RSA, developed by Rivest, Shamir, and Adleman in 1977, is widely used in public-key cryptography, implemented in Verilog and simulated in NC Launch.	1.Synthetic capabilities. 2.Low power consumption. 3.Parallel processing.	Limited Flexibility.
RSA		Most compilers' 64-bit integer operations are inadequate for RSA. Classic string storage is inefficient, requiring	1.High resistance to forgery. 2.Optimized performance.	Vulnerable to new attacks.

signature algorithm/co mplex numeric	Si, Hongwei [26]-2010	multiple nested loops for 1024- bit numbers. The n carry array beads scheme stores a 1024-bit number in a 32-element unsigned long array, enhancing speed. RSA key generation for a 1024-bit key takes under 2 minutes, with encryption/decryption under 1024 bits completed in under 2 seconds.		
RSA / 'n' Prime Numbers	Islam [27]-2018	The RSA algorithm is vulnerable due to the ease of computing keys from "N," the product of two primes. The proposed model improves security by using four distinct primes to create a more complex "N," making it harder to factor. Double encryption and decryption further strengthen the algorithm compared to traditional RSA.	1.Increased Key Size options. 2.Greater Complexity in attacks.	1.Longer key generation time. 2.Difficulty in key management.
New RSA_Improv ed Security	Sarjiyus [28]-2021	The RSA algorithm generates a public key for document encryption, relying on large primes to prevent key factorization. The proposed enhancement replaces the traditional modulus (n) with a new value (t) in the range ((n-q) < t < n), boosting security against brute force attacks. It also improves speed and reduces memory usage while maintaining strong cryptographic protection.	1.Enhanced security features. 2.Larger key sizes. 3.Increased performance.	1.Unproven Security. 2.Resource intensive.
RSA/Modified Keys	Nagar [29]-2024	The proposed RSA key generation method cuts generation time by up to 50%. A new decryption approach based on the Chinese remainder theorem (CRT) reduces	Flexibility in key management. Scalability.	Security risks in key exchange.

		computational costs by 66% and is 2.9 times faster than CRT-only methods.		
RSA/Text File Data Security	Sihotang [6]-2021	The RSA algorithm secures data through encryption and decryption, converting text to ASCII during encryption and restoring it during decryption. The software successfully encrypts and decrypts text files, ensuring the confidentiality of the data.	1.High security. 2.Asymmetric Encryption. 3.Widely accepted.	1.Slow performance. 2.Larger key sizes.
Area-efficient/RSA /RFID	Wang DM [4]-2022	The proposed design has a power dissipation of 15 mW at 13.56 MHz and occupies 0.27 mm^2. It requires 2 million clock cycles, yielding an APC value of 8.1, making it suitable for low-power systems.	1. Lower power consumption. 2. Enhanced security. Cost effective.	1.Limited Computational power. 2.Vulnerability to side-channel attacks.
RSA/LSI/Low Power	Satoh A [17]-2017	RSA operation for 1024 bits takes 23 ms at 45 MHz, with power dissipation of 330 mW (45 MHz) and 50 mW (5 MHz). Finding a 512-bit prime averages 27 Fermat tests and takes about 32 seconds for two primes. The chip's max frequency is 45 MHz, with peak currents of 100 mA and 15 mA, and a data transfer rate of 18.9 MB/sec.	1..High-speed operation. 2.Cost-effective. 3.Compact design	1.Potential vulnerabilities. 2.Quantum vulnerabilities.
RSA/ FPGA	Thobbi A [34]-2015	Devices: Spartan3 XC3s400fg320-4, Virtex6 XC6vlx75tff484-1. Max frequencies: 78.920 MHz (Spartan 3), 224.568 MHz (Virtex 6). Power: 420/3584 slices (Spartan 3), 187/93120 (Virtex 6). LUTs: 714/7168 (Spartan 3), 241/46560 (Virtex 6). Timing: 147.2311 µs (Spartan 3), 31 µs (Virtex 6).Encrypted output: 37.	1.High performance. 2.Scalability. 3.Power efficiency.	1. Complex design process. 2.Resource usage.

Fast Implementations of RSA Cryptography	Shand M [35]-2022	RSA multiplication cost grows quadratically with operand length, favouring smaller radices. Hensel's odd division and Karatsuba multiplication offer speedups of 1.05x and 1.22x. Optimizing squaring by rearranging operations before modular reduction improves efficiency.	1.Increased efficiency 2.Improved user experience. 3.Compatible with modern systems.	1.Hardware resource demands. 2.Increased Complexity.

1.4 Motivation

RSA encryption offers robust security, but its heavy computational requirements make it less practical for small, power-constrained systems when executed purely in software. There is a need for a **hardware-level solution** that retains RSA's security while reducing energy and area demands.

The motivation behind this project is to:

- **Bridge the gap** between theoretical RSA encryption and practical, efficient chip implementation,
- **Develop a power-efficient RSA engine** that is suitable for embedded use cases,
- And **demonstrate the complete design cycle**, from RTL development to FPGA prototyping and ASIC synthesis.

Through this project, the aim is to develop a working hardware model of RSA that not only performs the cryptographic functions but also meets **practical design constraints** like power, area, and performance.

1.5 Aim of the Project

To design and implement a low-power hardware model of the RSA algorithm using Verilog HDL, simulate and test the design on an FPGA board, and extend the same to ASIC-level analysis using Cadence tools.

1.6 Scope and Objectives

Scope of the Project:

- Implement RSA encryption, decryption, and key generation using Verilog HDL.
- Verify functionality through simulation in **Xilinx Vivado**.
- Prototype the design on an **FPGA Boolean Board** (XC7S550-1CSGA324).
- Optimize the design for **power and area**.
- Perform **ASIC semi-custom flow** using **Cadence Genus** for synthesis and analysis.

Objectives:

- Understand the RSA algorithm and convert it into functional hardware blocks.
- Develop clean, modular Verilog code for all RSA functions.
- Apply low-power design techniques (e.g., clock gating, pipelining).
- Use simulation tools for verifying functional correctness.
- Test the design on real hardware (FPGA) and evaluate its outputs.
- Synthesize the design for ASIC and analyze performance metrics such as area, power, delay.

1.7 Technical Approach

The complete workflow of this project is as follows:

1. **Algorithm Understanding and Partitioning:**
    - RSA key generation, encryption, and decryption were studied and broken down into logical steps.
2. **Design Using Verilog HDL:**
    - Modules for modular multiplication, exponentiation, and key generation were developed using Verilog.
3. **Simulation and Verification:**
    - The modules were simulated and tested in **Xilinx Vivado** using various input test cases to validate correctness.
4. **FPGA Prototyping:**
    - The tested design is synthesized and implemented on the **Boolean Board (XC7S550-1CSGA324)** to verify real-time functionality.
5. **ASIC Flow Using Cadence Tools:**
    - The final RTL was imported into **Cadence Genus** for synthesis.
    - Power, area, and timing reports were generated.
    - Optimization techniques were applied for improving synthesis metrics, especially using a second implementation approach that reduced area usage by up to **94%**.
6. **Comparison and Final Validation:**
    - Results from FPGA and ASIC flows were compared.
    - Functionality and performance were validated for small message sizes (4 to 8 bits) to ensure design robustness.

CHAPTER 2

Theoretical Description of Project

2.1 Introduction

Building upon the motivation and objectives discussed in Chapter 1, this chapter presents the complete technical approach adopted for the hardware implementation of the RSA algorithm. The focus is on designing modular, low-power Verilog HDL blocks for encryption, decryption, and key generation, simulating the functionality in **Xilinx Vivado**, testing it on **FPGA**, and extending it to **ASIC flow using Cadence Genus, Innovus,Virtuoso [32]**.

The entire design is structured to achieve a balance between **security, power efficiency, and area optimization**, using VLSI best practices and modular design principles.

2.2 Overview of RSA Algorithm:

The RSA algorithm is a public-key cryptosystem that involves three major operations:

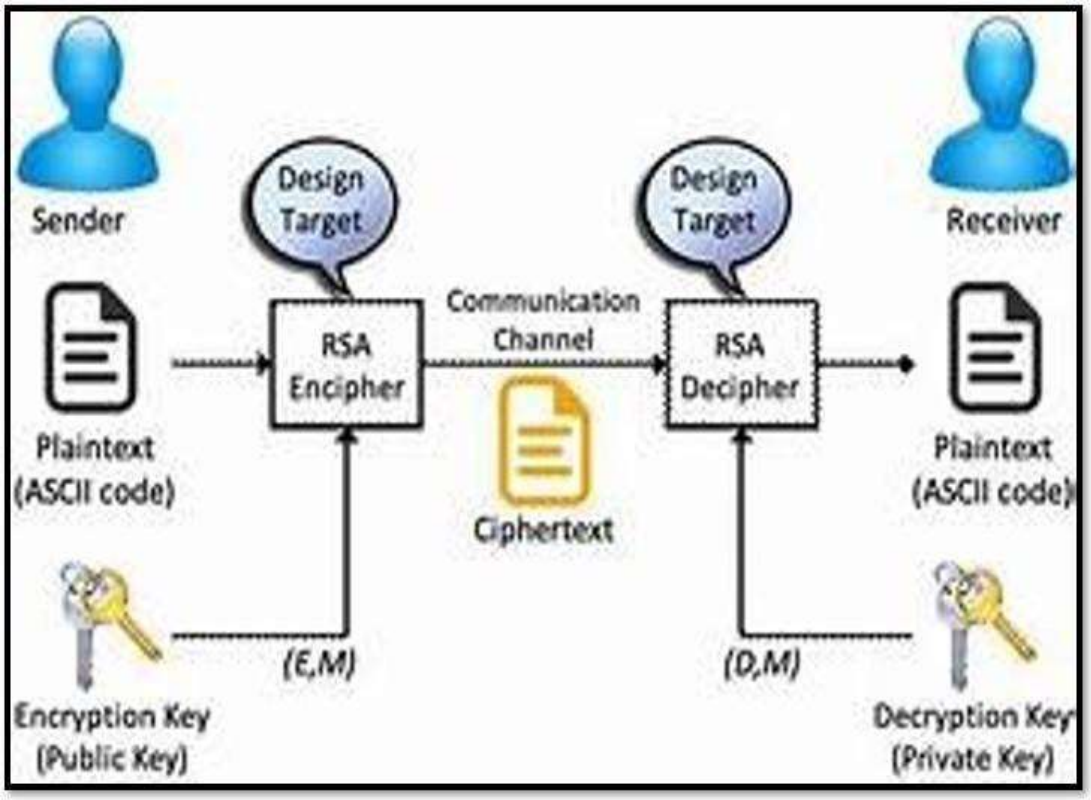

Fig 2.2.1: RSA Algorithm overview

1.**Key Generation**

- Select two distinct prime numbers p and q
- Compute : $n = p \times q$
- Compute : $\phi(n) = (p - 1) \times (q - 1)$
- Choose an integer e such that : $1 < e < \phi(n)$, $gcd(e, \phi(n)) = 1$
- Compute the private key d such that : $(d \times e).mod(\phi(n)) = 1$

2.**Encryption**

- Ciphertext : $C = M^e \, mod(n)$

3.**Decryption**

- Decrypted text : $M = C^d \, mod(n)$

Where:

- M: Message,
- C: Ciphertext,
- e: Public exponent,
- d: Private exponent,
- n: Modulus.

CHAPTER 3

System Design

3.1 System Architecture:

3.1.1. RSA Algorithm architecture:

The figure illustrates a four-stage modular hardware architecture for RSA encryption and decryption, designed to support both **key generation and secure message processing**.

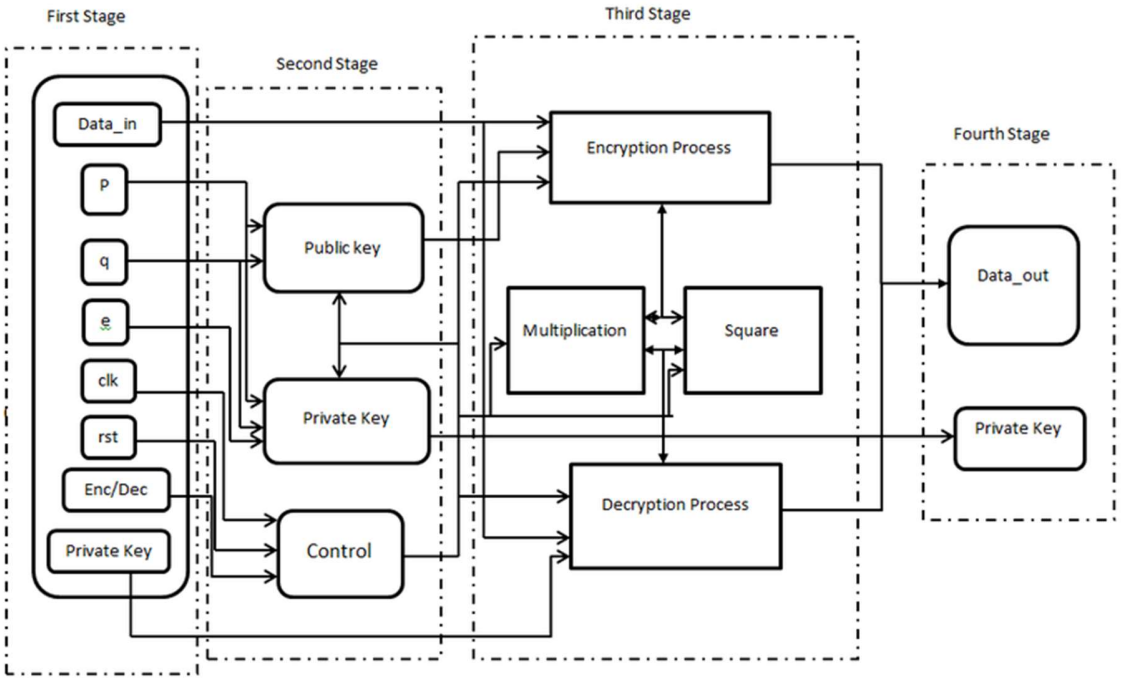

Fig 3.1.1: Block diagram of RSA algorithm flow

i. **First Stage**: This stage handles input parameters including message (Data_in), prime numbers (p and q), encryption key (e), clock (clk), reset (rst), and an encryption/decryption selection signal. These values are initialized and fed into subsequent stages.

ii. **Second Stage**: Key generation logic is implemented here. The system computes the **public key (e, n)** and **private key (d)** based on the mathematical foundations of RSA. A control block ensures proper synchronization and sequencing of operations.

iii. **Third Stage**: This is the core computational stage. Depending on the selected operation (encryption or decryption), the message is passed through the **Encryption Process** or **Decryption Process**, which involve **modular exponentiation** implemented using **multiplication and squaring modules**.

iv. **Fourth Stage**: The final output (Data_out) is generated after decryption or encryption, and the private key is optionally provided for validation or further cryptographic operations.

3.2 Implementation Flow:

The above figure represents the complete design and implementation flow for the **7-bit RSA encryption-decryption hardware architecture**. The flow encompasses both **FPGA prototyping** using Xilinx tools and **ASIC design** using Cadence tools.

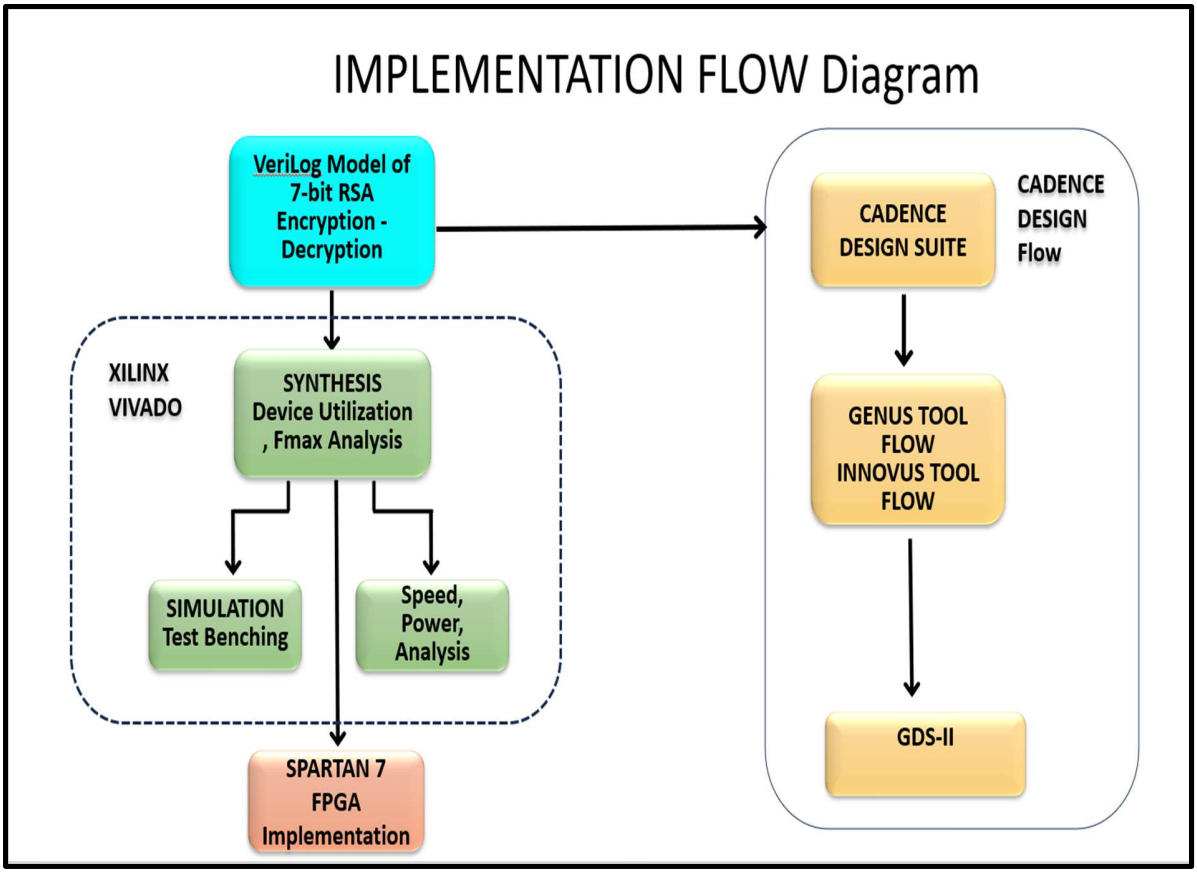

Fig 3.2.1: Implementation flow diagram

- The process begins with the **Verilog-based RTL design** of the RSA algorithm. This model captures the encryption, decryption, and key generation logic for 7-bit data.
- In the **Xilinx Vivado flow**, the RTL design undergoes:
    - **Synthesis**, where metrics such as **device utilization** and **maximum operating frequency (Fmax)** are evaluated.
    - **Simulation and testbenching**, ensuring functional correctness of the encryption and decryption modules.
    - **Performance analysis**, including **speed, area, and power** evaluation.
    - The design is then deployed on the **Spartan-7 FPGA Boolean Board**, where real-time functional verification is performed.
- Parallelly, the design is also prepared for ASIC implementation using the **Cadence Design Suite**:
    - The Verilog model is passed through the **Genus synthesis tool** followed by the **Innovus physical design flow**.
    - This results in the final **GDS-II layout**, which is suitable for tape-out and fabrication.

This dual-flow approach—**FPGA prototyping** and **ASIC semi-custom flow**—provides comprehensive validation and optimization, supporting both low-power performance and physical layout accuracy. The methodology ensures the RSA architecture is **design-rule compliant**, functionally accurate, and optimized for silicon implementation.

Semi-Custom IC Design Flow:

The Semi-Custom design flow is a structured approach used in the VLSI industry to develop integrated circuits (ICs) using a combination of pre-designed standard cells and custom logic. This method balances customization and efficiency, enabling faster design cycles while meeting performance requirements.

The design flow can be divided into three main stages:

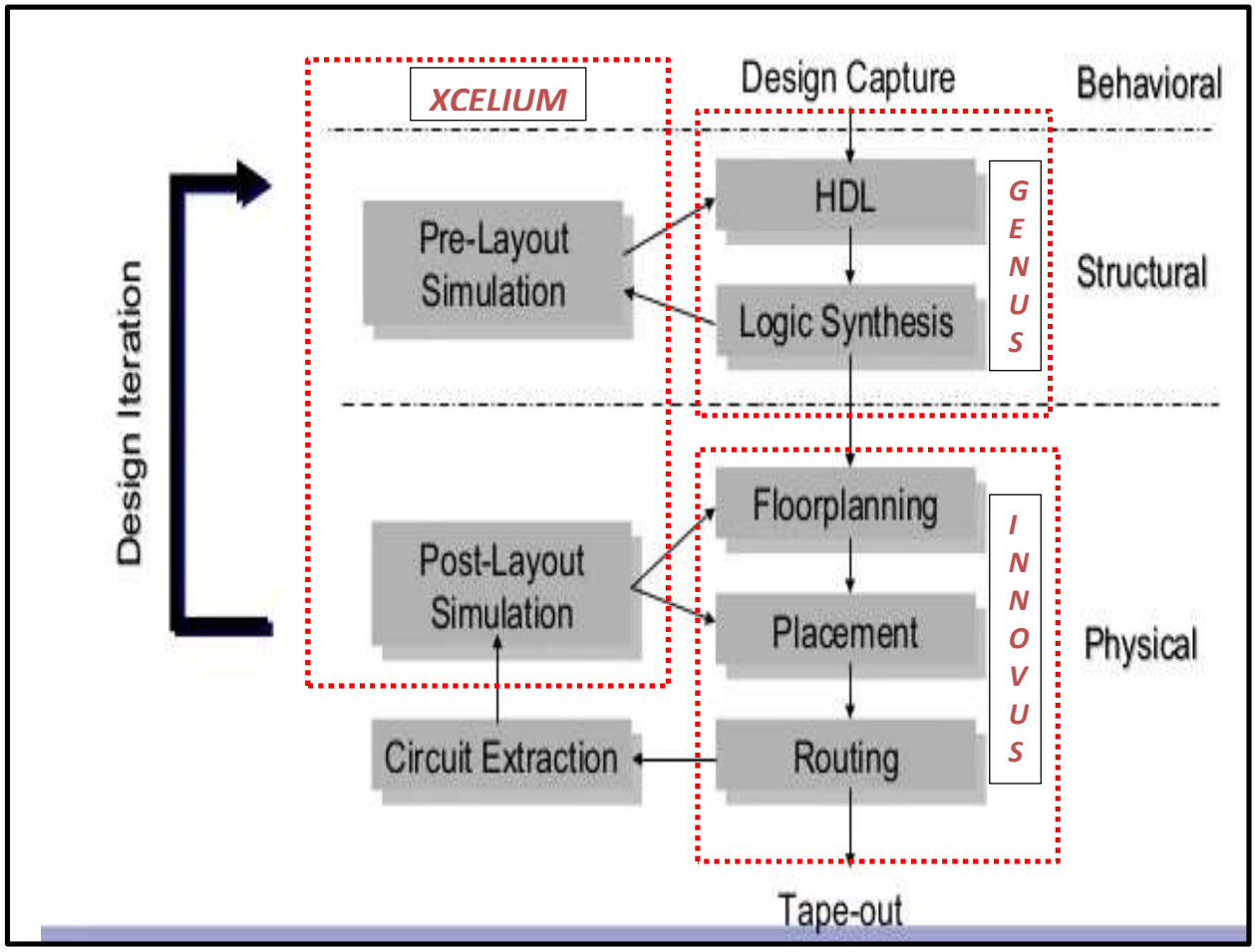

Fig 3.2.2 : Semi Custom IC Design flow

1. Design Capture (Behavioral Level)

- **HDL (Hardware Description Language):** The design process begins by describing the desired functionality using a high-level hardware language like Verilog or VHDL.
- **Pre-Layout Simulation:** Before any physical design, simulations are conducted to verify the functional correctness of the HDL description.

2. Structural Design

- **Logic Synthesis:** The verified HDL code is translated into a gate-level netlist using synthesis tools. This step converts behavioral logic into actual circuit elements.

3. Physical Design

- **Floorplanning:** Determines the general placement of different logic blocks within the chip area to optimize performance and minimize interconnect delays.
- **Placement:** Actual placement of standard cells within the defined floorplan.
- **Routing:** Establishes the metal connections between placed cells.
- **Circuit Extraction:** Extracts parasitic components like resistance and capacitance after routing.
- **Post-Layout Simulation:** Simulates the extracted circuit to ensure timing and functionality are not degraded due to layout effects.
- **Tape-out:** Once the design is verified and finalized, it is sent for fabrication. This stage is referred to as tape-out.

Design Iteration

Throughout the flow, feedback loops allow designers to make iterative improvements based on simulation results, synthesis reports, or layout performance.

3.3 Resources Required:

3.3.1 Hardware required:

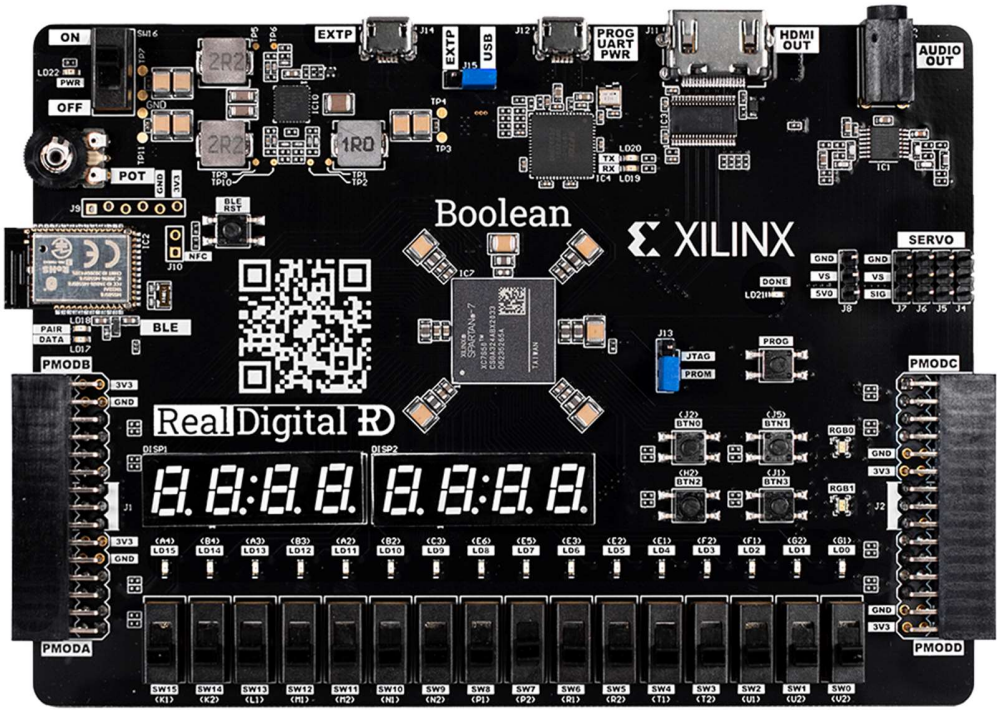

Fig 3.3.1 : Spartan 7 FPGA Board

Table 3.3.1 : Specifications of SPARTAN 7 FPGA

FPGA	XC7S50-1CSGA324
I/O Interfaces	USB-UART for programming and serial communication HDMI output On-board Bluetooth Low Energy radio (option)
Memory	128 Mbit Serial Flash
Display	Two 4-digit 7-Segment displays HDMI source (up to 1080p)
Audio	Two identical channels connected to 1/8" stereo audio jack
Switches and LED'S	16 Slide switches 16 LEDs Four Push-buttons
Clocks	One 100 MHz crystal oscillator
Expansion Ports	10K Potentiometer connected to XADC Four Pmod ports

EDA Tools Required:

Xilinx Vivado:

Fig.3.3.2 : Xilinx Vivado Software

CADENCE Design Suite:

ENTUPLE TECHNOLOGIES

Standard Bundle Specification

Conformal® Low Power GXL
Genus™ Low Power Option
Genus™ Physical Option
Genus™ Synthesis Solution
Cadence® SKILL Development Environment
Virtuoso® Schematic Editor Verilog Interface
Virtuoso® AMS Designer Environment
Virtuoso® Schematic Editor XL
Virtuoso® ADE Assembler
Virtuoso® Layout Suite GXL
Voltus™-Fi Custom Power Integrity Solution XL
Innovus™ Mixed Signal Option
Innovus™ Hierarchical Design Option
Innovus™ Implementation System
JasperGold® Formal Verification Platform
Modus DFT Option
Modus ATPG
Innovus™ DFM Option
Generator to generate Assura® compatible verification decks
Pcell Generator
Cadence® QuickView Layout and Mask Data Viewer
Cadence® Physical Verification System Design Rule Checker XL
Cadence® PVS Layout vs Schematic Checker XL
Cadence® Physical Verification System Results Manager
Cadence® Physical Verification System Design Analysis Option
Cadence® Physical Verification System Design Review
Cadence® Physical Verification System Advanced Analysis Option
Cadence® Physical Verification System Advanced Device Option
Virtuoso® Integrated PVS Option for Layout Suite
Cadence® Quantus™ Extraction XL
Cadence® Quantus™ Advanced Analysis GXL Option
Cadence® Quantus™ Advanced Modeling GXL Option
Advanced SI
Advanced PI
Allegro® Venture PCB Designer
Spectre® RelXpert Reliability Simulator
Spectre® AMS Designer
Spectre® MMSIM with Spectre X Simulator
Tempus™ Timing Signoff Solution XL
Voltus™ IC Power Integrity Solution XL (VTS-XL)
Tempus™ Timing Signoff Solution ECO
vManager™ Project Server
vManager™ Linux Client (Quantity 5)
Xcelium™ Single Core
Xcelium™ Digital Mixed Signal Option

Fig 3.3.3 : Cadence Design Suite

3.4 Design Implementation

The RSA algorithm was implemented using **Verilog HDL**, including modules for:

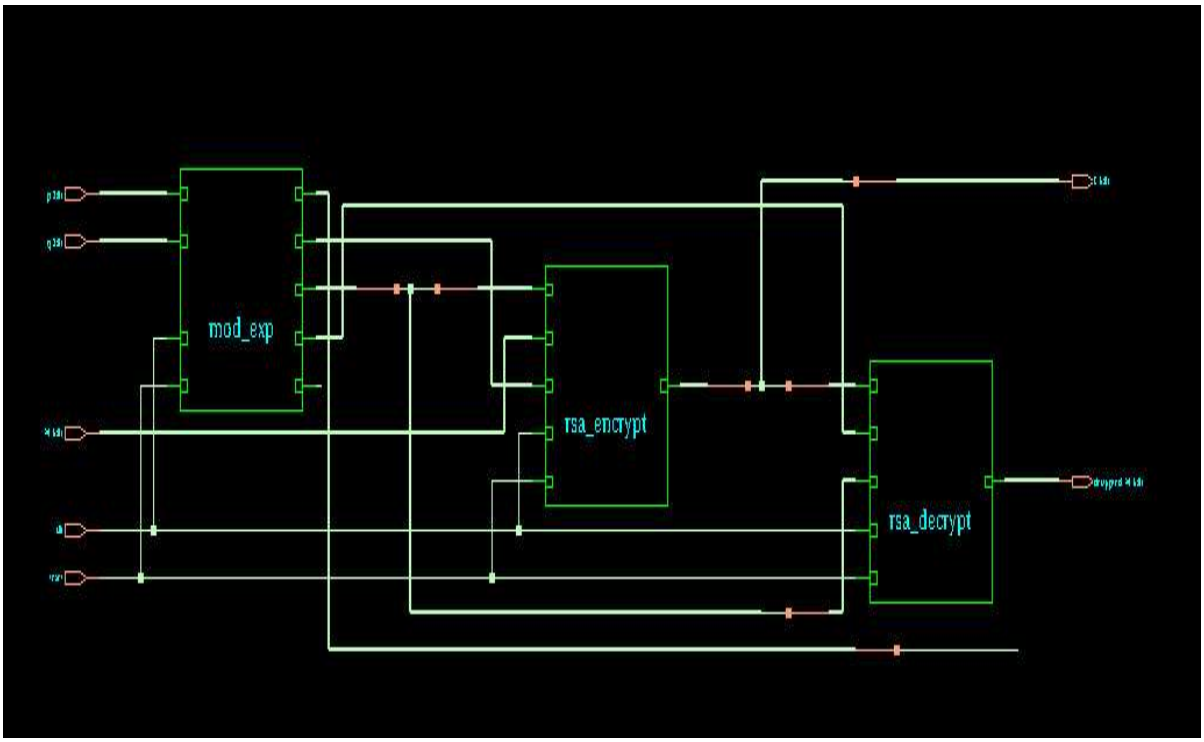

Fig 3.4.1: Synthesized Block diagram of RSA Algorithm

- Key generation (public and private keys)
- Encryption logic
- Decryption logic
- Control logic for mode selection and synchronization

The Verilog model was structured to allow **modular design**, enabling separate simulation and synthesis of encryption and decryption pipelines.

CHAPTER 4

Results and Discussion

4.1 Synthesis, Simulation and FPGA Prototyping using Xilinx Vivado.

Using **Xilinx Vivado**, the following design steps were executed:

- RTL Coding in Verilog
- Functional Simulation using **XSim**
- Synthesis (device utilization, timing, area reports)
- Test bench development to validate encryption and decryption
- Analysis of **Fmax**, **LUTs**, **flip-flops**, and power estimates
- Deployment on **Spartan-7 FPGA (Boolean Board)** for prototyping and real-time validation

Refer to Fig 3.2.1 : Implementation Flow Diagram.

4.1.1 Vivado Result Summary

Table 4.1.1 : Utilization Summary for 7-bit RSA

Resource	Utilization	Available	Utilization %
LUT	22932	32600	70.34
FF	303	65200	0.46
IO	31	210	14.76

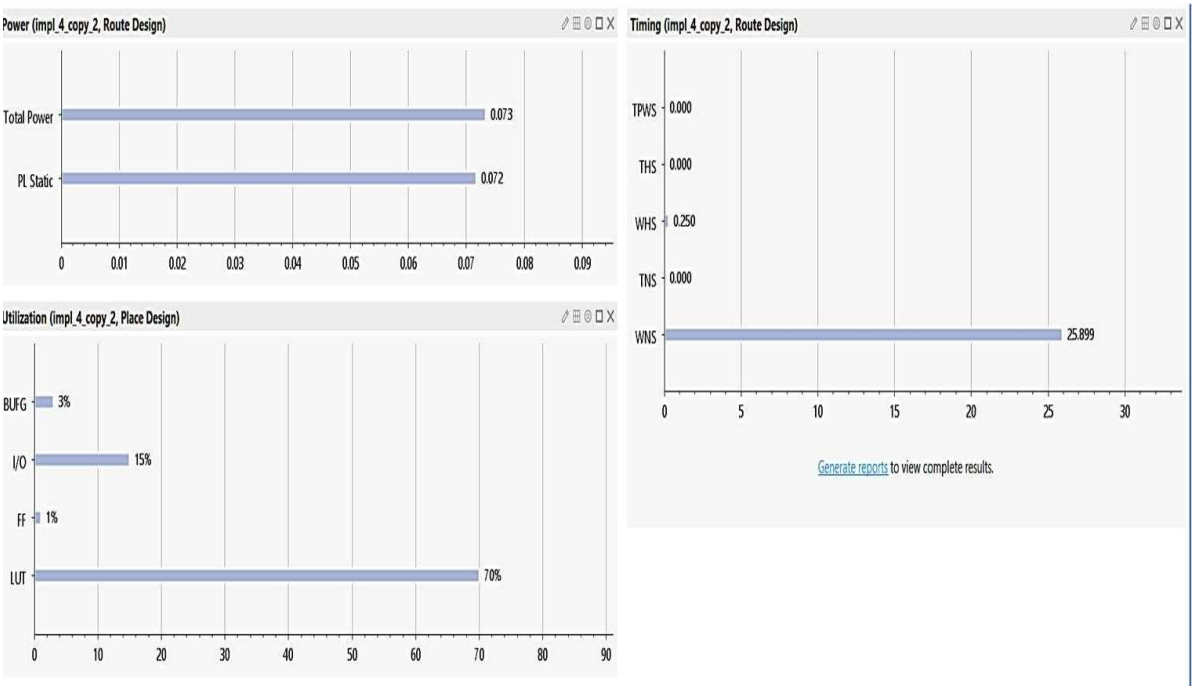

Fig 4.1.1 : Power Summary for 7-bit RSA

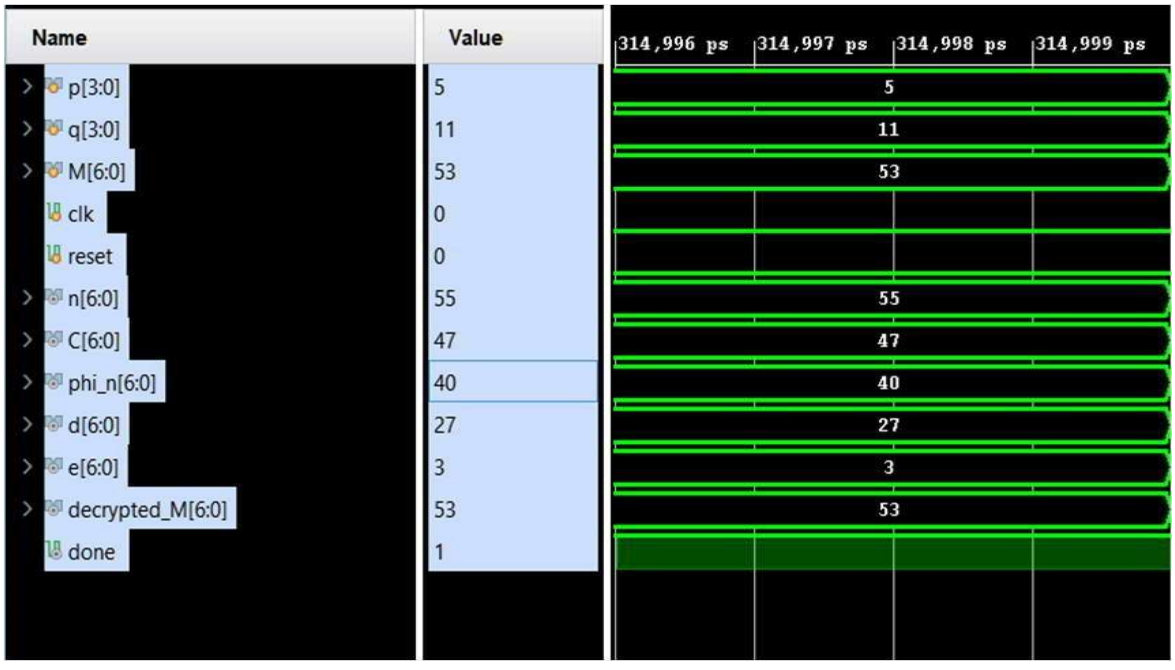

Fig 4.1.2 : Simulation result for 7-bit RSA

Fig 4.1.3 : FPGA Prototyping Result

Table 4.2.1 : Comparison of Different bit data ranging from 4 to 8

Size	LUT Utilization (%)	D-FFs Utilization (%)	IO Utilization (%)	Power Consumption (mW)	Remarks
4-bit	-	-	-	-	RSA- an asymmetric algorithm not valid.
5-bit	-	-	-	-	Algorithm not valid.
6-bit	26.09	0.14	13.33	72	Suitable for moderate security, low resource utilization and power consumption.
7-bit	70.34	0.46	14.76	73	Increased Security, Most-efficient resource usage with minimum power consumption.
8-bit	Over-utilization (>100)	-	-	-	Due to over-utilization and high fanout, it is non – Synthesizable.

4.2 ASIC Implementation using Cadence Tools

The same Verilog model was synthesized for ASIC using **Cadence Design Suite**, where:

- **Cadence Genus** performed RTL synthesis and generates technology dependent gate-level netlist.
- **Innovus** carried out FloorPlanning, placement, clock tree synthesis (CTS) and routing.
- Design is optimized for **area**, **power**, and **timing** in the ASIC flow.
- Final routed design is exported in **GDS-II format**, suitable for fabrication.

This flow demonstrates a **Semi-custom ASIC methodology** for deploying cryptographic IP blocks.

- **CADENCE Results :**

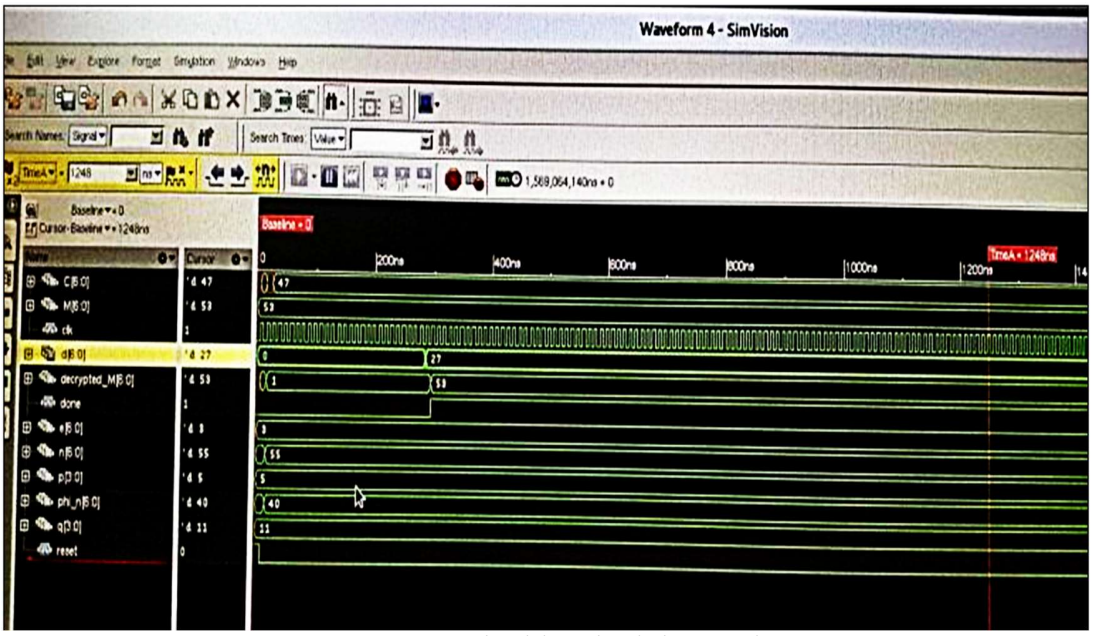

Fig 4.2.1 : Simvision Simulation Result

Simulation of **Encryption & Decryption** Process by **RSA Algorithm** for :

- **Input** parameters : p = 5, q =11, e =3 and M = 53
- Intermediate Parameters : n = 55, phi_n = 40, d = 27
- **Output** Results : C = 47 ; decrypted_M = 53

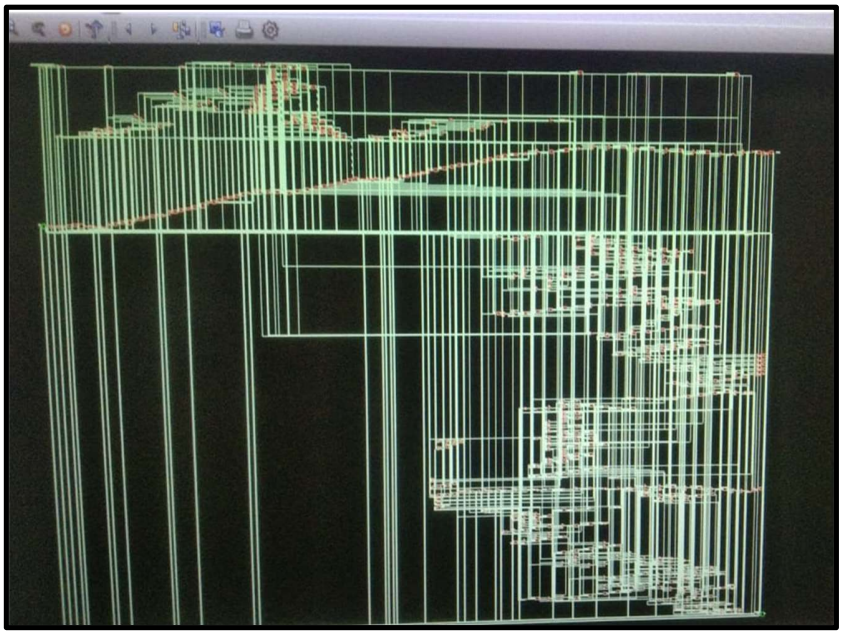

Fig 4.2.2 : Synthesized netlist view in Genus

- View shows **gate-level netlist** mapped to 45nm technology standard cells.
- Reported total area: **0.4 mm²**, optimized for performance and area.

Fig 4.2.3 : Floorplan

- Shows the initial layout of the core, standard cell rows, and I/O pins.
- Allocates space for placement, routing, and power distribution.
- Forms the base for all downstream physical design stages

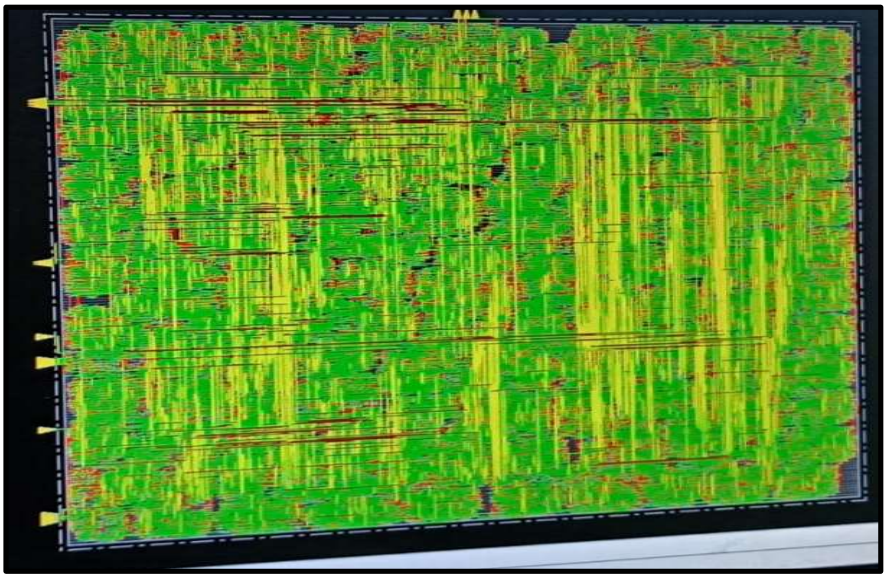

Fig 4.2.4 : Placement

- Standard cells are placed within the defined core area.
- Aims to minimize wirelength and optimize timing.
- No routing is done yet—only cell locations are fixed.

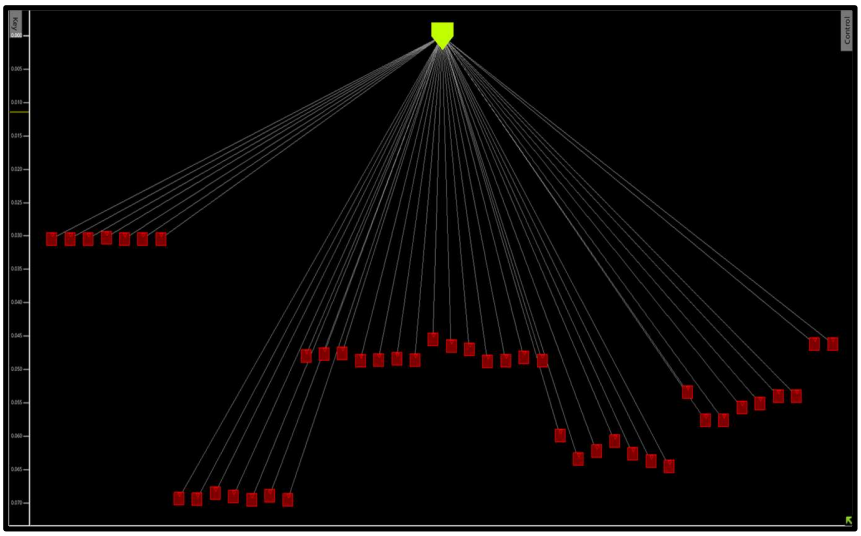

Fig 4.2.5 : Clock Tree Synthesis

- Distributes the clock signal uniformly to all sequential elements.
- Inserts buffers/inverters to balance clock skew and latency.
- Ensures synchronized timing across the entire design.

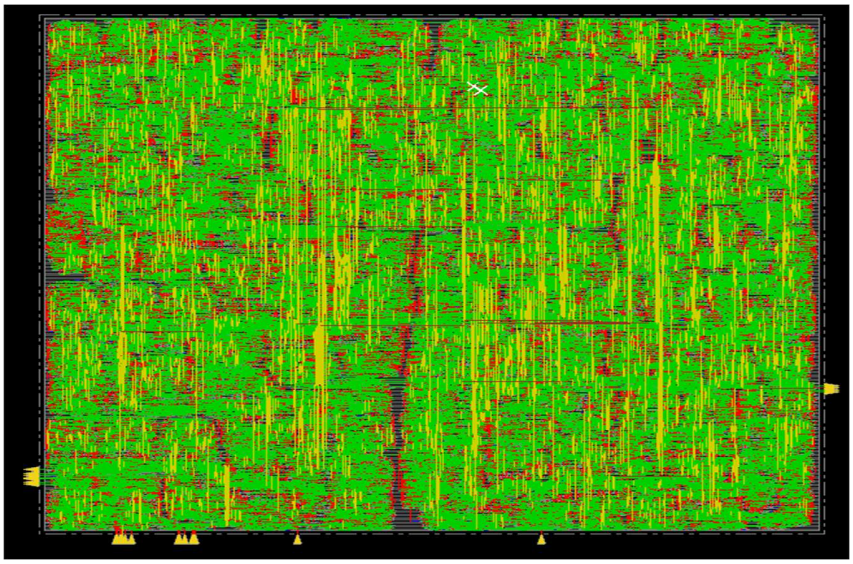

Fig 4.2.6 : Routing

- Connects all placed standard cells using metal layers.
- Completes signal paths while meeting timing and design rules.
- Final step before layout verification and GDS-II generation.

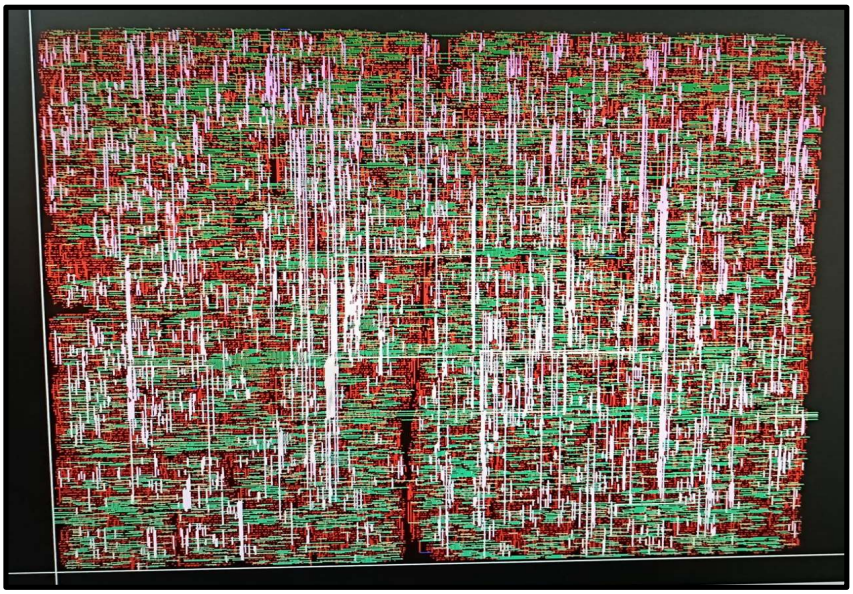

Fig 4.2.7 : GDS II Layout

- Final layout file ready for fabrication.
- Includes all layers: diffusion, poly, metal, vias, and text.
- Represents the complete physical design of the chip.

Comparison between Vivado & Cadence Results:

Table 4.2.2 : Comparison of Results

Feature	**Vivado (FPGA) (28nm)**	**Cadence (ASIC) (45nm)**
Technology	Spartan-7 FPGA	GSC 45nm CMOS (Standard cell)
Synthesis tool	Xilinx Vivado	CADENCE Genus.
Implementation Tool	Vivado Implementation	Innovus (RTL to GDS-II)
Max Clock Frequency(Post-STA)	0.2 MHz	2.5 MHz
Utilization /Area	70.34 % LUTs, 14.76 % of IOs 0.46% of D FFs utilisation in FPGA	Optimized on-chip area is 0.4 mm^2
Power	73 mW	1.3 mW
Verification	Simulation (Vivado xsim)	SimVision Simulation (RTL, Gate-level)
Final Output	Bitstream & Hardware test	GDS-II file

Key Observations :

- **Vivado Pros**: Fast prototyping, easy to test on hardware, ideal for quick iterations.
- **Vivado Cons**: Resource constraints, limited speed compared to ASIC.
- **Cadence Pros**: High performance, better optimization, close to real IC fabrication.
- **Cadence Cons**: Longer flow, requires more precise timing and physical constraint handling.

CHAPTER 5

Conclusion

.

The RSA cryptographic design was successfully implemented using both **FPGA** and **ASIC** design methodologies.

In the **FPGA flow**, Xilinx Vivado and Spartan-7 FPGA were used for functional validation, achieving:

- **LUT utilization:** 70.34%
- **Maximum Frequency:** 0.2 MHz (post-STA)
- **Total Power Consumption :** 73 mW

In the **ASIC flow**, the design was implemented using the Cadence toolchain:

- **Simulation:** SimVision
- **Synthesis (Genus):** 0.426 mm² area, 105567 cells
- **Timing:** 2.5 MHz maximum frequency
- **Power:** 1.348 mW total power consumption
- **Physical Design:** Floorplanning, placement, CTS, routing, and **GDS-II** generation completed in Innovus

This project demonstrated the full **RTL-to-GDSII flow**, highlighting trade-offs:

- **FPGA**: Faster prototyping and testing
- **ASIC**: Optimized for power, area, and speed

Overall, the work provided hands-on experience in both reconfigurable and semi-custom IC design, offering valuable insights into the digital VLSI design process.

CHAPTER 6

Future Scope

The design can be further optimized by implementing it using **FinFET**-based standard cell libraries. This approach is expected to provide improved performance, reduced leakage power, and enhanced area efficiency compared to the current planar CMOS implementation. Exploring FinFET technology will enable better scalability for advanced nodes and enhance the design's suitability for low-power VLSI applications.

References

[1] Ajay C Shantilal, “A Faster Hardware Implementation of RSA Algorithm” (2022)
[2] Keshav Kumar,K.R. Ramkumar, Amanpreet Kaur, Somanshu Choudhary,“A Survey on Hardware Implementation of Cryptographic Algorithms Using Field Programmable Gate Array”
[3] Sandeep Saini, Kusum Lata, Abhishek Sharma and G R Sinha, “An FPGA implementation of the RSA algorithm using VHDL and a Xilinx system generator for image applications” (2021)
[4] D.M. Wang, Y.Y. Ding, J. Zhang, J.G. Hu and H.Z. Tan, “Area-efficient and ultra-low-power architecture of RSA processor for RFID”(2022)
[5] Xinjian Zheng, Zexiang Liu, Bo Peng, “Design and Implementation of an Ultra Low Power RSA Coprocessor”
[6] Hengki Tamando Sihotang, Syahril Efendi, Elvyawati M Zamzami, Herman Mawengkang,“Design and Implementation of Rivest Shamir Adleman’s (RSA)Cryptography Algorithm in Text File Data Security”(2021)
[7] Pradeep Krishnadoss, Palani Thanaraj Krishnan, Nishanth Paramasivam,Deepesh Sai Kesavan, Anish Thishyaa Raagav, “Dynamic Approach for Time Reduction in RSA Algorithm through Adaptive Data Encryption and Decryption”
[8] Sheba Diamond Thabah, Mridupawan Sonowal, Rekib Uddin Ahmed, Prabir Saha,“Fast and Area Efficient Implementation of RSA Algorithm ”(2019)
[9] Chiranth E, Chakravarthy H.V.A, Nagamohanareddy P, Umesh T.H, Chethan Kumar M.,“Implementation of RSA Cryptosystem Using Verilog”
[10] Chandra M. Kota and Cherif Aissi, “Implementation of RSA Algorithm and cryptanalysis”
[11] V. Joseph Raj, R. Felista Sugirtha Lizy,“Performance Enhancement of RSA Using Runge-Kutta Technique”
[12] Xin Zhou, Xiaofei Tang,“Research and Implementation of RSA Algorithm for Encryption and Decryption”
[13] Kritsanapong Somsuk, “The Improving Decryption Process of RSA by Choosing New Private Key”
[14] Gopathoti KK, Pendyala SS, “A Review on low-Power VLSI CMOS and CNTFET Circuits” (2023)
[15] Abouelkheir E, El-Sherbiny S, “Enhancement of Speech Encryption/Decryption Process Using RSA Algorithm Variants”(2022)
[16] Kumar K, Ramkumar KR, Kaur A, Choudhary S, “A Survey on Hardware Implementation of Cryptographic Algorithms Using Field Programmable Gate Array”(2020)
[17] Thabah SD, Sonowal M, Ahmed RU, Saha P,“Fast and Area Efficient Implementation of RSA Algorithm”

[18] Dar MA, Bukhari SN, Shafi M, “Cryptographic Algorithms on Low power devices used in Smart City: Issues And Enhancements” (2021)
[19] Chi-Chia Sun, Bor-Shing Lin, Gene Eu Jan & Jheng-Yi Lin,“VLSI Design of a RSA Encryption/Decryption Chip using Systolic Array-based Architecture”(2016)
[20] Chandra M. Kota and Cherif Aissi1, “Implementation of the RSA algorithm and its cryptanalysis” (2022)
[21] Israa Al_Barazanchi* 1, Shihab A. Shawkat2 , Moayed H. Hameed3, Khalid Saeed Lateef Al-badri4 “, Modified RSA-based algorithm: a double secure approach” (2019)
[22] Pradeep Krishnadoss1* Palani Thanaraj Krishnan1 Nishanth Paramasivam1 Deepesh Sai Kesavan1 Anish Thishyaa Raagav1 ,“Dynamic Approach for Time Reduction in RSA Algorithm through Adaptive Data Encryption and Decryption” (2024)
[23] Hongwei Si, Youlin Cai, Zhimei Cheng,“An Improved RSA Signature Algorithm based on Complex Numeric Operation Function” (2010)
[24] Muhammad Ariful Islam1, Md. Ashraful Islam1, Nazrul Islam1*, Boishakhi Shabnam2 ,“A Modified and Secured RSA Public Key Cryptosystem Based on “n” Prime Numbers” (2018)
[25] O. Sarjiyus, B.Y Baha and E.J Garba “New RSA Scheme For Improved Security”(2021)
[26] Sami A. Nagar and Saad Alshamma “High Speed Implementation of RSA Algorithm with Modified Keys Exchange”(2024)
[27] V. N. Hemanth Kolliparal, Sai Koushik Kalakota1, Sujith Chamarthi, S. Ramani1, Preeti Malik and Marimuthu Karuppiah,“Timestamp Based OTP and Enhanced RSA Key Exchange Scheme with SIT Encryption to Secure IoT Devices”(2023)
[28] A. Satoh ~, Y. Kobayashi], H. Niijima ~, N. Ooba ~, S. Munetoh 1, and S. Sone “A High-Speed Small RSA Encryption LSI with Low Power Dissipation” (2021)
[29] Amit Thobbi1 ShriniwasDhage2 Pritesh Jadhav3 Akshay Chandrachood4 ,“Implementation of RSA Encryption Algorithm on FPGA” (2015)
[30] M. Shand J. Vuillemin Digital Equipment Corp., Paris “Fast Implementations of RSA Cryptography” (2022)
[31] AMD Vivado, Cadence Genus, Innovus Documentation
[32] Youtube Tutorials