Digital Security System Design

Vũ Minh Đăng
Toán 2 22-25, VNU-HCM High School for the Gifted

Đặng Bảo Ngọc
Toán 2 22-25, VNU-HCM High School for the Gifted

Trần Thiên Phú
Tự Nhiên 2 22-25, VNU-HCM High School for the Gifted

Instructors: **Dr. Lê Đức Hùng** (University of Science VNUHCM)

Abstract

In the contemporary landscape, characterized by escalating security challenges, the domain of cryptography has undergone a profound and expansive evolution. Concurrently, microcontrollers have found diverse applications in electronic devices, thereby engendering a demand for lightweight encryption solutions. The principal objective of this research is the design and implementation of electronic solar charger circuits utilizing Altium Designer software, with a specific focus on the operational intricacies of the LM317 integrated circuit (IC). Moreover, this study encompasses the evaluation of the performance of AES, ChaCha20 encryption algorithms, and PRINCE lightweight encryption, tailored for preconfigured microcontrollers. The applied methodology in this research centers on the meticulous design of printed circuit boards and the programming and encryption of on-chip systems within the Quartus Prime environment. Notably, cryptographic acceleration cores, specifically the Nios II, play a pivotal role in this process. The envisaged outcomes of this study are geared towards the development and implementation of data encryption within constrained hardware environments, such as the preconfigured microcontroller FPGA DE10. These programmable on-chip systems are intended to serve as foundational software for the execution of rudimentary data encryption algorithms. The anticipated contributions of this research extend to the advancement of secure data encryption practices. The focal point lies in the adept application of lightweight encryption algorithms on hardware platforms characterized by limited computational capabilities. Ultimately, this endeavor seeks to fortify security within the field of microcontroller technology, addressing the contemporary challenges posed by the ever-evolving landscape of electronic devices.

Introduction

Contemporary society witnesses an extensive proliferation of electronic devices, underscoring the imperative need for robust security and reliability measures in their operational frameworks. As articulated by Sciencedirect Topics, lightweight encryption emerges as a pivotal solution capable of operating on hardware with diminished computational capacities while still ensuring a commendable degree of security. A myriad of lightweight encryption algorithms has been devised, applied, and tailored for specific applications, each showcasing diverse performance metrics in both hardware and software domains across various contexts.

This report meticulously delves into the foundational knowledge essential for the application of lightweight encryption on Field-Programmable Gate Array (FPGA) hardware, capitalizing on the inherent advantages that FPGA presents in customizing hardware configurations, as elucidated in Electronic Design Help. The meticulous design of the printed circuit board assumes a paramount role in this nuanced research area. Consequently, the adept utilization of Altium Designer becomes imperative, addressing the intricacies of Schematic Layout and Component Footprint assembly integral to this research. Moreover, the configuration of the System on Chip (SoC) within the FPGA board, a critical component of this study, facilitates the analysis of signal wave characteristics pertaining to the Advanced Encryption Standard (AES) and PRINCE encryption algorithms. The successful execution of this task is attributed to a nuanced understanding of boolean algebra in the context of electronics boards, coupled with the utilization of the Quartus Prime platform. This amalgamation of sophisticated methodologies and platforms underscores the rigor and precision employed in addressing the multifaceted challenges inherent in the examination of lightweight encryption on FPGA hardware.

Methodology

Part 1: Printed circuit board (PCB)

1.1 Components

In order to facilitate the fabrication of a Printed Circuit Board (PCB) conjoined with an integrated chip, our initiative commenced with the meticulous design of a low-dropout voltage regulator circuit tailored for the charging of 18650 batteries. In the preliminary phase of this endeavor, a judicious selection of the requisite components was conducted, adhering to a rigorous criterion for appropriateness and compatibility.

Designator	Footprint	LibRef	Quantity
C1	CN1	CN1	1
C2, C3	CP1/2.5	CP2	2
C4	CN1	CN1	1
D1	DO-201	1N5819	1
D2	DO-201	1N4007	1
D3	DZ0.5A	DZ	1
IC1	SOP8	SO8	1

IC2	QTO-220	LM317	1
JP1, JP2	HED2	Header 2_1	2
L1	L01	L1	1
LED1, LED2	LED-3	LED	2
Q1	QTO-220	BD139	1
R1	R2W	R2W	1
R2	R1/4W	R1/4W	1
R3, R5, R7	0805_Res	R	3
R4, R6	R1/4W	R1/4W	2
R8	R5W	R5W	1
VR1	VR	VR1	1

1.2 Design

Subsequent to the procurement of the essential components, the progression advanced to the creation of a schematic library meticulously constructed for these components. This task was executed with precision using Altium Designer software, a sophisticated tool that facilitates intricate design processes. The library containing the pinout diagrams can be found in this repository.

Following the establishment of the schematic library, the subsequent phase entailed the generation of a comprehensive schematic diagram delineating the intricate interconnections within the circuit. This critical phase was meticulously executed to encapsulate the functional relationships among the various components. Subsequently, the endeavor transitioned to the physical manifestation as we meticulously arranged and interconnected the components on the Printed Circuit Board (PCB), ensuring alignment with the established schematic. This iterative process culminated in a harmonious integration of elements, poised for subsequent phases of implementation and testing. The design file for the charging circuit can be found in this repository.

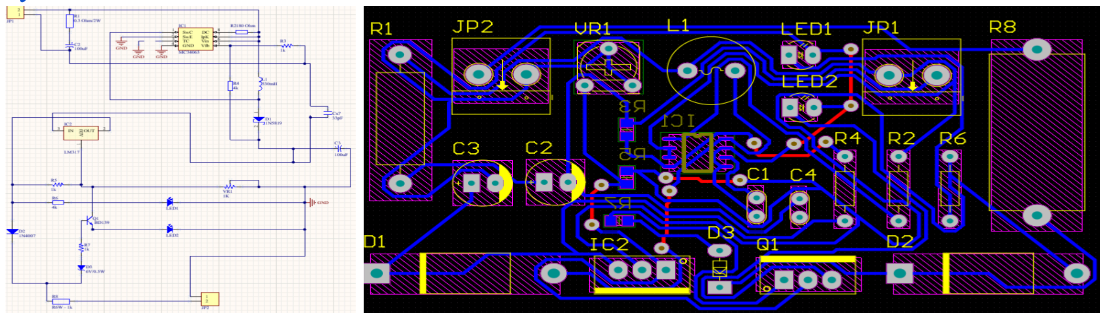


Figure 1.2a. Solar charger schematic and layout

Part 2: Digital design

2.1 Overview

Within this section, a comprehensive exploration was undertaken to scrutinize the software frameworks, algorithms, and algebraic operations integral to the reconfigurable System-on-Chip (SoC). A particular focus was directed towards the hardware infrastructure underpinning this system, notably leveraging the DE10-Standard board. This hardware platform boasts a versatile array of features, accommodating an extensive spectrum of circuit designs ranging from elementary constructs to multifaceted multimedia multi-project applications.
In elucidating the intricacies of the system, meticulous attention was devoted to furnishing detailed data, which can be found in this repository, while specifications of the FPGA used can be found in this repository.

2.2 Boolean Algebra Simulation

Boolean algebra, a foundational component in digital circuit design, assumes a pivotal role in the construction and optimization of logic gates within these circuits. Logic gates, encompassing AND, OR, NOT, XOR, NAND, NOR, and XNOR, are systematically fashioned by mapping Boolean algebra operations. Consequently, Boolean algebra emerges as an indispensable tool within the ambit of this research.
In elucidating the operational nuances of semiconductor devices within the circuit, a judicious reliance on the Digital Logic Simulator (DLS) software is observed. Through the utilization of this software, simulations are conducted for both combinational and sequential circuits. This systematic approach facilitates a nuanced comprehension of binary arithmetic, enabling the adept design of electronic and computer circuits.

2.3 1 bit Half Adder

2.3.1 Design

Employing logic gates and adhering to the principles of Boolean algebra, we conducted simulations to instantiate a 1-bit half-adder circuit within the Digital Logic Simulator (DLS) software. The resulting schematic diagram is presented herewith:

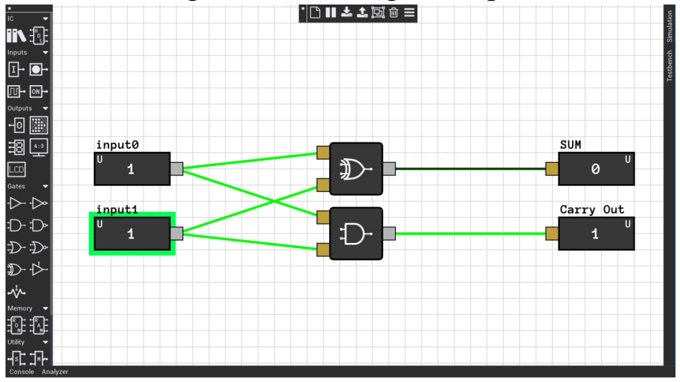


Figure 2.3.1a. Schematic diagram of the 1-bit half-adder circuit

2.3.2 Theory

Inputs A and B are two 1-bit binary numbers that we will add together. The half-adder circuit produces two results: the output Sum is the result of adding A and B, and the output Carry is the carry bit, if any.
The operation of the half-adder circuit is as follows:

- The output Sum (addition result) is computed using the XOR (exclusive or) operation between A and B. In other words, the Sum result will be 1 if either input A or B is 1, but not both. If both inputs are 1, Sum will be 0.

$$\text{Sum} = A \wedge B$$

- The output Carry (carry bit) is computed using the AND operation between A and B. This means that Carry will be 1 only when both inputs (A and B) are 1. If at least one of them is 0, Carry will be 0.

Carry = A & B

2.4 1 bit Full Adder

2.4.1 Design

By harnessing the principles of logic gates and applying Boolean algebra, a meticulous simulation of a 1-bit full adder circuit was executed within the Digital Logic Simulator (DLS) software. The resultant schematic diagram, encapsulating the intricacies of the simulated 1-bit full adder circuit, is presented herewith:

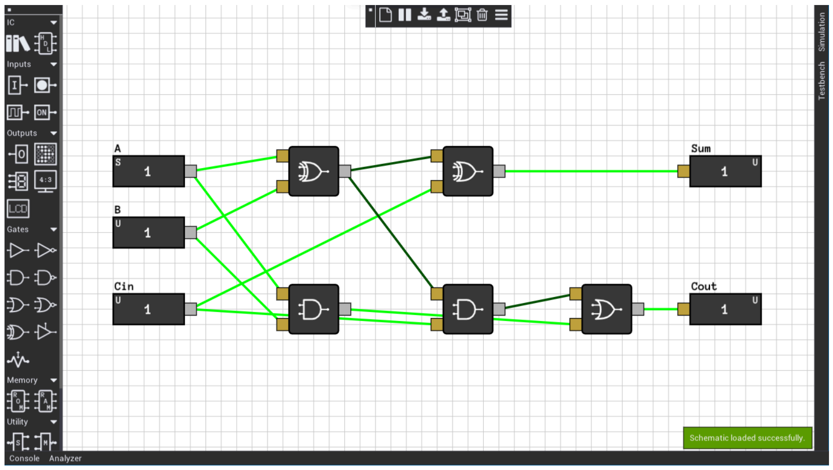


Figure 2.4.1a. Schematic diagram of the 1-bit full adder circuit

2.4.2 Theory

The full adder circuit has three inputs and two outputs.

Input A holds the value of the first number, input B holds the value of the second number, and input Cin takes the value from the previous carry bit if any.

The two outputs of the full adder circuit are:

The output S is the algebraic sum, the addition result of the two numbers A, B, and Cin. The output Cout carries the value to the next input bit if available.

The operation of the full adder circuit is as follows:

- The Sum (S) is the result of adding the A bit, the B bit, and the previous carry bit (Cin). S is calculated using the XOR gate of the three inputs:

S = A ^ B ^ Cin

- The next carry bit (Cout) is determined using AND and OR gates:

Cout = (A & B) || (Cin & (A ^ B))

2.5 Hardware Description Language

In the programming endeavor targeted at configuring the hardware on the DE10 Standard board, a strategic decision was made to adopt the Verilog programming language. This choice was underpinned by several discerning reasons, including:

- The DE10 Standard board uses Intel's FPGA and supports Verilog in Intel's development tools like Quartus Prime and ModelSim.
- Verilog is one of the two most popular Hardware Description Languages (HDL) in the electronics industry, alongside VHDL, as noted in What is the future of Verilog in VLSI industry.

2.6 Hardware Programming

Within this section, the Verilog language was employed as the medium for programming, with the primary objective of loading the source code into the FPGA system through the Quartus software. The source code that has been loaded into the system can be found on the research log. This table describes the corresponding locations of each component on the DE10 Standard board.

2.7 Signal Waveform Simulation

Within this dedicated section, algorithm simulations were executed utilizing the Verilog programming language in conjunction with the ModelSim software. The overarching objective of these simulations was to design, validate, and assess the functionality of both circuits and algorithms deployed on the DE10 Standard FPGA. The source code for these circuits can be found in the research log.

- Multiplexer (MUX) circuit is designed to assign the output based on the control signal, following the principles of a multiplexing circuit. It has a simple structure and operates as a channel selector.
- Sequential LED shifter circuit is designed to shift LEDs from left to right on the DE10 board when a designated button is pressed. The LED shifting speed aligns with the DE10 Standard board's clock signal. The circuit employs a counter to count pulses and a register to store the LED states.
- Greatest Common Divisor (GCD) circuit is designed to calculate the greatest common divisor of two 16-bit integers. It uses clock signals, reset signals, and control signals to perform the computation.

Part 3: Cryptography

3.1 Algorithms

In the course of this research, a focused exploration is undertaken into various encryption methods, constituting a comprehensive examination of the Caesar cipher, ChaCha, Advanced Encryption Standard (AES), and PRINCE. This strategic emphasis on a diverse set of encryption techniques aims to garner insights into their respective strengths, vulnerabilities, and applicability within contemporary cryptographic frameworks.

3.1.1 Symmetric-key Algorithms

Symmetric-key encryption uses the same key for both encrypting and decrypting data. This key is known as a symmetric secret key.

Applications	Examples
- Encrypting data and messages to maintain confidentiality. - Providing security for online connections, such as through HTTPS.	- Serpent, Data Encryption Standard (DES), Advanced Encryption Standard (AES), Salsa, ChaCha20

3.1.2 Asymmetric-key Algorithms

Asymmetric-key encryption uses a pair of public and private keys for access authorization from different parties.

Applications	Examples

- Digital signature for verifying the data source. - Secure database storage. - Secure communication and protection of personal information.	- RSA, Elliptic Curve Digital Signature Algorithm (ECDSA), Ed25519

3.1.3 Hash Functions

Hash functions perform one-way encryption by converting data into a fixed character string.

Applications	Examples
- Checking and verifying the integrity of files and data. - Secure password storage. - Generating digital signatures and verifying data integrity.	- MD5, SHA-256, SHA3-256, SHA3-512

3.2 Configuring the System on a Chip (SoC)

Within this pivotal section, the integration of hardware and software on a System-on-Chip (SoC) platform is executed with the aim of consolidating features and functions onto a singular board. The meticulous process of designing and implementing this SoC system transpires through the utilization of Quartus Prime and Platform Designer. The ensemble of components within this integrated system encompasses a Nios II CPU, RAM, a JTAG/UART interface, output LEDs, and a purposeful encryption accelerator employing the ChaCha encryption algorithm. This amalgamation of hardware and software on an SoC platform represents a holistic approach to system design, leveraging cutting-edge tools and technologies for the creation of a cohesive and multifunctional computing environment.

3.2.1 SoC System Structure

The SoC system was built with the following key components:

- Nios II CPU capable of system control and management.
- RAM for data and program storage for the CPU.
- JTAG/UART interface for programming, debugging, and interacting with the system.
- Output LEDs for displaying system information and results.
- ChaCha encryption accelerator for data encryption.

3.2.2 Verilog Code for the SoC System

The Verilog source code that describes the SoC system is a module used to establish connections and utilize the components within the SoC system.

Following the configuration of the FPGA, the subsequent step involved the utilization of encryption algorithm source code tailored for the Nios II CPU. This source code encompasses the implementation of diverse encryption algorithms, namely ChaCha, Advanced Encryption Standard (AES), and PRINCE, all tailored to operate on the DE10 Standard board. The integration of these encryption algorithms into the Nios II CPU's source code underscores a comprehensive approach to cryptographic functionality within the FPGA environment. This amalgamation not only broadens the range of cryptographic capabilities but also exemplifies the adaptability of the Nios II CPU in handling diverse encryption methodologies on the DE10 Standard board, which can be found in this research log.

- The ChaCha encryption accelerator is used to perform data encryption, including configuring the number of algorithm rounds. Then, data is input and encrypted. Finally, the actual encryption result is compared to the theoretical value to verify the accuracy of the encryption.
- PRINCE lightweight encryption is a type of symmetric encryption designed for improved performance by reducing encryption complexity.

Results

1. Efficiency

To simulate encryption algorithms, especially to evaluate the efficiency of lightweight encryption on constrained hardware, configuring and setting up the SoC system appropriately will provide results regarding speed and accuracy. This allows for optimization in terms of energy consumption, computer resources, and security.

In general, each encryption algorithm used and simulated in this research has its own speed, security, and efficiency characteristics. The Caesar encryption algorithm has fast processing speed but does not guarantee security and is relatively easy to break due to the algorithm's characteristics. The AES encryption algorithm provides high security but is computationally intensive and challenging to break due to its block-based encryption method. Meanwhile, the lightweight encryption algorithm PRINCE is a resource-efficient approach for weaker hardware in smart electronic devices while still ensuring reasonable reliability.

2. Cost-effectiveness

Throughout the project, a self-designed PCB board for a voltage-reducing and self-cut-off charger for 18650 batteries was successfully conducted, with a cost of approximately 50,000 VND per board. Detailed component costs can be found in this research log.

Additionally, this project was made possible with the Terasic DE10 Standard board to build the SoC system for simulating encryption algorithms. The price for this product is approximately $790 (refer to DE10-STANDARD DEV KIT P0493 Terasic).

3. Performance comparison:

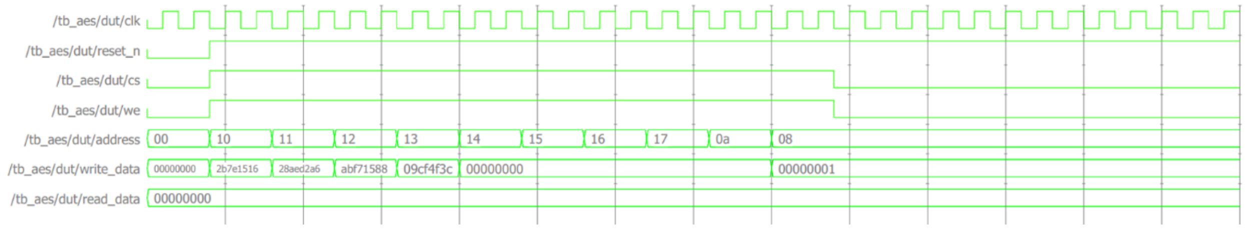
The comparative analysis between the AES algorithm and the PRINCE lightweight encryption is presented in the following table, outlining key parameters such as block size, clock cycle, maximum frequency, throughput, total slices, efficiency per slice, power consumption, and delay:

Algorithm	Block size	Clock cycle	Max freq. (MHz)	Throughput (Mbps)	Total slices	Efficiency (Mbps/slice)	Power (watt)	Delay (ns)
PRINCE-128	64	1	31	2032	956	2.126	0.165	31.48
AES (Rouvroy et al., 2004)	128	32	123	358	1214	0.29	-	-
AES (Gielata et al., 2008)	128	11	165	-	77	-	-	-

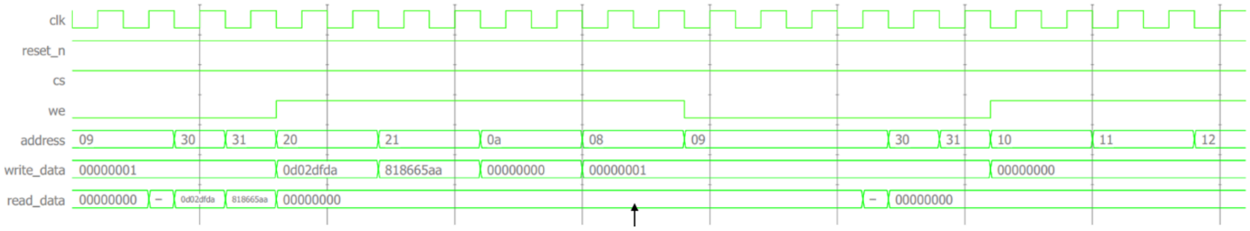
Algorithm	Block size	Clock cycle	Max freq. (MHz)	Throughput (Mbps)	Total slices	Efficiency (Mbps/slice)	Power (watt)	Delay (ns)
AES (Chodowiec and Gaj, 2003)	128	32	60	166	522	0.32	-	-

Additionally, the signal waveforms illustrate the computational processes and data flow for both AES and PRINCE encryption methods during the encryption of a quoted text. These visual representations provide a deeper insight into the intricacies of the encryption tasks performed by each algorithm.

AES:



PRINCE:



Discussion

1. Printed circuit board (PCB)

The voltage-reducing and load-cutting charging circuit for 18650 rechargeable batteries has helped charge the battery and protect it from overheating or overloading. In the circuit design, we used the 1N5819 diode and LM317 voltage regulator, where the diode controls the current and prevents reverse flow, and the LM317 circuit allows for adjustable output voltage based on a custom signal. Therefore, the 18650 rechargeable battery can adapt to various input voltage levels, preventing damage or hazards.

2. Digital design

2.1 1 bit Half Adder

A 1-bit half adder has a simple structure with basic logic gates like AND and XOR. Therefore, it is easy to implement and resource-efficient. Additionally, 1-bit half adders play a crucial role in building processors, memory units, and other digital logic circuits. Furthermore, 1-bit half adders serve as building blocks for full adders, enabling the addition of multi-bit binary numbers.

2.2 1 bit Full Adder

A 1-bit full adder can be used to alter values in encryption through bitwise addition and in substitution-permutation networks (S-boxes). This adder can also be employed to rearrange bits in input data, creating asymmetry in the P-box substitution network. In one-way encryption, a 1-bit full adder can be used to create irreversible transformations, making it harder to break the encryption. In symmetric encryption, a 1-bit full adder can be used in data transformation operations, such as performing XOR operations between character strings (keys) and original data to generate encrypted data.

3. Cryptography

While the PRINCE encryption algorithm has made improvements in terms of speed, performance, and security, we still observe some potential issues and risks.

- Although PRINCE encryption has improved in terms of asymmetry, the algorithm can still be broken or subjected to brute-force attacks.
- PRINCE encryption uses relatively small bit and byte sizes, so this type of encryption may not be suitable for applications that require encryption with a larger number of bits or bytes.

The lightweight PRINCE encryption has applications in various fields:

- PRINCE encryption can be used in IoT devices with limited resources to ensure security and performance.
- IoT Devices: PRINCE encryption can be used in IoT devices with limited resources to ensure security and performance.

4. Experimental results

- Clock cycles: PRINCE-128 requires 1 clock cycle, while AES requires 32 clock cycles (according to Rouvroy et al., 2004). This means that PRINCE-128 is much faster than AES in terms of clock cycles.
- Throughput: PRINCE-128 has a throughput of 2032 Mbps, while AES has a throughput of 358 Mbps (according to Rouvroy et al., 2004). This means that PRINCE-128 is also much faster than AES in terms of throughput.
- Area: PRINCE-128 requires 956 slices, while AES requires 1214 slices (according to Rouvroy et al., 2004). This means that AES is slightly more area-efficient than PRINCE-128.
- Security: Both AES and PRINCE are considered secure algorithms. AES is the current Advanced Encryption Standard, while PRINCE is a newer algorithm that is being considered for standardization.
- Speed: PRINCE-128 is much faster, requires 1 clock cycle and has 2032 Mbps throughput while AES is lower, requires 32 clock cycles and has 358 Mbps throughput.

Overall, based on the data provided, PRINCE-128 appears to be significantly faster than AES for encrypting 64-bit blocks. However, AES is slightly more area-efficient.

Conclusion

The application of encryption in electronic devices has become widespread, leading to increased requirements for lightweight encryption security. Therefore, the development of electronic circuits and the application of encryption, especially lightweight encryption, on pre-configured microcontrollers are essential.

We have successfully applied chip-based systems with programmable capabilities and integrated encryption accelerators. The results of this research will support the development of hardware for data encryption in microcontrollers and programmable chip-based systems. This will lay the foundation for enhancing the security of microcontrollers and advancing the development of lightweight encryption algorithms on resource-constrained hardware.

In the field of smart electronic devices, security and reliability are becoming crucial concerns. Lightweight encryption is a solution capable of operating on resource-constrained hardware while ensuring security. Many lightweight encryption algorithms, such as PRINCE, have been applied and optimized for applications using less powerful hardware, providing promising results with adequate security.

Acknowledgment

We would like to express our sincere gratitude to all those who have contributed to the completion of this project.

Firstly, we truly value the guidance and aid given to us during this project by our instructors, Mr. Lê Đức Hùng. His insightful feedback and suggestions helped to shape the project's scope. Secondly, we would also like to express gratitude to Mr. Thái Hồng Hải and Mr. Mã Khải Minh, our mentors, for their unwavering assistance whenever our project faced technical issues. Additionally, we would like to thank Deslab - University of Science for giving us the supplies, environment, and resources that were crucial to the completion of our project. Lastly, we wish to acknowledge the PIISE coordination team for their continuous supervision and assistance during our participation in the program, as well as the PIISE organizing team for developing such a valuable learning and experience program.

Without the contributions and assistance of all those mentioned above, this project would not have been possible. Thank you all for your valuable contributions and support.

Appendix

This research project is carried out within the PTNK Initiative in Interdisciplinary Science and Engineering (PIISE) Summer Research Internship Program under the topic Information Technology at Deslab - University of Science from 27/07/2023 - 29/10/2023.

This research project has achieved significant milestones, including:

Time period	Milestone
24/07/2023 - 20/08/2023	Implementing PCB Design
21/08/2023 - 01/10/2023	FPGA and Encryption Approach
02/10/2023 - 29/10/2023	Building a SoC System

With the aim of sharing knowledge, aligned with the spirit of continuous learning and accumulation in science, we would like to share the tools, knowledge, and useful keywords that have been beneficial to us during the course of this project, including:

- Altium
- Quartus Prime

At the end of the program, we have gained valuable experiences for ourselves, and we are ready to continue our path of continuous learning and research. We hope that the results we share will make a meaningful contribution to the community and will be continuously built upon.

References

[1] Terasic Inc (2018). DE10-Standard Development Kit Specifications. Retrieved from Terasic - DE Boards - Cyclone - DE10-Standard

[2] Joachim Strömbergson (2019). The Prince lightweight block cipher in Verilog. Retrieved from secworks/prince: The Prince lightweight block cipher in Verilog. (github.com)

[3] Joachim Strömbergson (2019). Verilog 2001 implementation of the ChaCha stream cipher. Retrieved from secworks/chacha: Verilog 2001 implementation of the ChaCha stream cipher. (github.com)

[4] Joachim Strömbergson (2019). Verilog implementation of the symmetric block cipher AES (NIST FIPS 197). Retrieved from secworks/aes: Verilog implementation of the symmetric block cipher AES (Advanced Encryption Standard) as specified in NIST FIPS 197. This implementation supports 128 and 256 bit keys. (github.com)

[5] Norman, T. L. (2014). Integrated security systems design: A complete reference for building enterprise-wide digital security systems. Butterworth-Heinemann.

[6] Verma, G. K., & Tripathi, P. (2010). A digital security system with door lock system using RFID technology. International Journal of Computer Applications, 5(11), 6-8.

[7] Gasser, M., Goldstein, A., Kaufman, C., & Lampson, B. (1989, October). The Digital distributed system security architecture. In 12th National Computer Security Conference (pp. 305-319).

[8] Gehrmann, C., & Gunnarsson, M. (2019). A digital twin based industrial automation and control system security architecture. IEEE Transactions on Industrial Informatics, 16(1), 669-680.

[9] Riel, A., Kreiner, C., Macher, G., & Messnarz, R. (2017). Integrated design for tackling safety and security challenges of smart products and digital manufacturing. CIRP annals, 66(1), 177-180.

[10] Chen, H. C., Guo, J. I., Huang, L. C., & Yen, J. C. (2003). Design and realization of a new signal security system for multimedia data transmission. EURASIP Journal on Advances in Signal Processing, 2003, 1-15.

[11] Rivest, R. L. (1990). Cryptography. In Algorithms and complexity (pp. 717-755). Elsevier.

[12] Salomaa, A. (2013). Public-key cryptography.

[13] Hellman, M. E. (2002). An overview of public key cryptography. IEEE Communications Magazine, 40(5), 42-49.

[14] Van Tilborg, H. C., & Jajodia, S. (Eds.). (2014). Encyclopedia of cryptography and security. Springer Science & Business Media.

[15] Koblitz, N. (1994). A course in number theory and cryptography (Vol. 114). Springer Science & Business Media.