Svedberg Open
DISSEMINATION OF KNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: https://www.svedbergopen.com/



Research Paper Open Access

IDEA - Evaluation of Modern Cryptographic Techniques For Crypto-Processors

Abdulsamad Ibrahim

Department, College, University, City, Country Department of basic Science Dentistry, College of Dentistry, Hawler Medical University, Erbil 44001, Iraq. Email: abdulsamad.hussein@hmu.edu.krd

Abstract
In the digital age, data security is a major concern as there is a rapid increase in the Internet and data-driven systems. IDEA is a symmetric-key block cypher that uses 128 bits of key to encrypt 64-bit blocks of data using modular addition, modular multiplication, and exclusive OR (XOR). IDEA has good resistance to cryptanalytic attacks especially differential and linear cryptanalysis because of its mathematical structure.
Modern encryption algorithms like the Advanced Encryption Standard (AES) have superseded the use of IDEA in most applications, but it might still be relevant in embedded and resource constrained systems. But, with the new found need for low latency encryption, energy saving, and hardware compatibility, these classical cryptographic algorithms need to be reassessed with modern tools and benchmarks.
In this work, the functionality and performance of the IDEA algorithm on modern cryptographic processors is examined. It is analyzed, using simulation-based experiments, in terms of its memory footprint, computational efficiency, energy consumption and resistance to the cryptanalytic attacks selected. The aim of the study is to find out whether IDEA is still a viable encryption algorithm in today's secure-processing environment and to investigate security–performance trade-offs in today's data-processing applications. The results provide important clues to the ongoing validity of IDEA and to closing the design/implement gap.

Keywords: cryptographic processors, IDEA, International Data Encryption Algorithm, symmetric key encryption, block cipher, cryptographic analysis.

1. Introduction

The world of ubiquitous digital communications and linked computing systems places an ever increasing importance on sensitive data security. Cryptography plays a vital role in the protection of information (confidentiality, integrity, authentication, non-repudiation and access control) while it is being transmitted and stored. IDEA is a symmetric key block cipher which uses a 128-bit key to encrypt 64-bit blocks. Combination of modular addition, modular multiplication and exclusive OR (XOR).With the help of this mathematical design, IDEA has ensured a good resistance against various types of cryptanalyses, such as differential cryptanalysis, linear cryptanalysis etc. IDEA was employed in the past, but has been mostly replaced by more recent algorithms like Advanced Encryption Standard (AES). However, given the simplicity of its structure and its known security characteristics, it could be an interesting choice for particular applications, such as embedded systems and resource-limited cryptographic processors.
Although it is a historically significant tool, the performance and implementation of IDEA on modern hardware platforms have not been explored. Today, secure processing systems require low latency, high throughput, low memory, energy efficiency and hardware support like field-programmable gate arrays (FPGAs), application-specific integrated circuits (ASICs), and dedicated cryptographic processors. Hence, it is time to rethink classical encryption methods in the light of new simulation environment and performance measures.
This study will develop and evaluate an implementation of IDEA algorithm in Verilog and/or VHDL. The proposed implementation is evaluated on throughput, latency, memory utilization, computational efficiency and implementation complexity. It is compared to the performance of existing encryption algorithms, namely, AES and DES. The modular multiplication operation, one of the most time consuming operations in the IDEA algorithm, is presented in particular. The study investigates possible optimization techniques for this operation in order to improve the overall performance of the cipher.

The primary goal of this research was to see if IDEA is still usable as an encryption technique in today's secure-processing systems using hardware. It is evaluated through a simulation to assess its speed, memory usage, hardware implementation features, and ability to overcome some of the cryptanalytic attacks. The results can be useful for the ongoing validity of IDEA, and to understand the compromises between security, performance, and implementation cost in current cryptographic systems.

Cryptographic Security Requirements

Cryptography is applied to meet a number of basic security needs. Confidentiality guarantees that information that should be confidential will only be available to those authorized. Integrity ensures that data cannot be changed without permission. Authentication - establishes the identity of a party to a communication and the validity of the information source. Non-repudiation is the ability to ensure that neither the sender nor the receiver of a message denies the message or transaction. Access Control: This ensures that only authorized users obtain access to system resources. Lastly, availability means that users are able to access systems and information whenever they need it.

1-1 Types of Cryptography

There are two types of cryptographic algorithms, namely Symmetric-key and Asymmetric-key cryptography. In symmetric-key cryptography, both the key used to encrypt the message and the key used to decrypt it are the same. Examples are Data Encryption Standard (DES), Triple Data Encryption Standard (3DES or TDES), Advanced Encryption Standard (AES), Rivest Cipher 4 (RC4) and IDEA. Asymmetric-key cryptography, on the other hand, is based on two mathematically related keys, a public key and a private key. RSA is a widely-known asymmetric-key algorithm.

One of the bases of cryptography is that the security of an encryption system should not rely on the secret of the algorithm. Rather, security should be based on the secrecy of the cryptographic key. This principle is usually related to Kerckhoffs's principle which assumes that an attacker can know the encryption algorithm and can try to analyse or attack the encrypted data.

The original "readable" message is referred to as plaintext. After the encryption, the output is unintelligible and is called as Ciphertext. Encryption and decryption are the processes of converting plaintext to ciphertext and vice versa. In symmetric-key cryptography, the secret-key is the same for both operations. In general, the longer the key, the greater the key space, but also may make it more difficult to attack by brute force, depending on the design and implementation of the algorithm.

A block cipher encrypts plaintext in blocks of a fixed size and decrypts to blocks of fixed size or a closely related fixed size. The stream cipher, on the other hand, works on data continuously, typically bit by bit or even byte by byte. Encryption latency is the time it takes to convert plaintext to ciphertext and throughput is the amount of data that can be encrypted in a certain amount of time.

The rest of the paper is organized as follows: In Section 2 the IDEA algorithm and its mathematical structure are described in detail, in Section 3 the hardware implementation methodology is discussed, in Section 4 the experimental results and performances are presented, and, finally, in Section 6 the conclusions are presented and future work is outlined.

2. Related Work

The implementation and performance of cryptographic algorithms such as IDEA over different application areas have been explored in several studies. This section summarizes the literature related to algorithm modifications, comparisons of performances, hardware implementation and security applications, and highlights the research gaps that are to be filled in this study.

2.1. Modifications to the Algorithm and Hybrid Solutions

For increased security of hash function in a single encryption operation, Almasri and Jani [1] suggested multiplicative additive block cipher (MABC). Their intent was to protect databases from hackers and network administrators who were not permitted to access the database. However, the study didn't take into account the hardware implementation related issues and didn't compare performance on real platforms.

Lubis et al. [2] used Elgamal and IDEA for digital signature applications. They employed Elgamal for encrypting/decrypting the signatures and IDEA for protecting the signature from any modification by unauthorized users. There were signature of public parameters (*p*, *q*, *g*, *y*) and the signature signed with the sender's private key. The results they presented were of the authenticity assurance of binary documents, without detailed analysis of the computational burden of the hybrid method.

Yastrebov et al. [8] used multi-objective optimization techniques and fuzzy cognitive map (FCM) learning using evolutionary algorithms for medical data analysis. They adopted a two-stage method, which includes

K-means clustering and automatic formulation of FCM model. A hybrid algorithm of the genetic algorithms (GA) and firefly algorithms (FA) for optimization problems was proposed by Zhang et al. [9]. Though these studies are not directly cryptographic in nature, they show the possibilities of evolutionary and hybrid approaches to optimization problems which can be extended to cryptographic key scheduling or hardware optimization.

2.2. Comparisons of performance and security evaluations.

Alemami et al. [3] made a comparison between AES, DES, Blowfish, RSA, and IDEA on Cloud platforms based on the encipherment capacity, security strength, memory requirement and encipherment time. They found that AES had the minimum encipherment time for large data sizes and Blowfish had the minimum memory size. Interestingly RSA and IDEA were identified as less secure than AES, Blowfish and DES. But it was limited to software implementations and hardware specific performance measures.

Thabit et al. [4] gave an extensive study of cryptographic algorithms in IoT environment, which includes lightweight block ciphers, stream ciphers, and hybrid ciphers. They evaluated the security algorithms based on the performance, robustness and computational complexity. In the survey, it was found that there is a high demand for lightweight cryptographic solutions in the resource-constrained IoT devices. In a related study, Thabit et al. [5] conducted an overview of cloud computing security solutions which included lightweight cryptography, genetics-based cryptography, and machine learning (ML) approaches (supervised, unsupervised, semi-supervised, and reinforcement learning).

Kumaran and Logu [6] made a comparison between HMAC encryption and IDEA in cloud database management system. Their evaluation focused on latency and storage efficiency. Results indicated HMAC was able to use 66.80% of its capacity while IDEA used 62.97% and HMAC outperformed IDEA in its performance ($p< 0.05$, two-tailed). The study concluded that HMAC encryption has a minimized security breach in the dataset management system. But the comparison was restricted to software types of cloud environments.

Ucal Sari and Ak [10] analyzed Industry 4.0 application with classical and fuzzy data envelopment analysis (DEA). The fuzzy DEA approach used in this paper, although not specifically related to cryptography, illustrated the importance of quantitative performance measurement for industrial systems and offered fuzzy algorithms for output-oriented DEA in both BCC and CCR models.

Upadhyay and Ghosh [11] expressed a concern about security issues associated with quantum computing and security of third-party hardware components. Their research pointed out the potential dangers for intellectual property (IP) theft in quantum systems, in which the Quantum Approximate Optimization Algorithm (QAOA) could expose delicate graph structures in cyber-physical systems, like power grids.

2.3. Implementation of IDEA in Hardware

There has been little research done on the hardware implementation of IDEA on modern cryptographic processors.

Syahroni et al. [7] implemented IDEA for data security in autonomous underwater vehicle (AUV) telemetry systems. Their application had the capability to access the data of the database server and permitted monitoring via telemetry using a smartphone, employing IDEA decryption. The testing results indicated that, on average the processing time of IDEA was 0.00065 s, which proves the feasibility of using IDEA for real-time embedded applications. However, the implementation was software based on mobile platforms and the hardware-specific metrics such as throughput, latency, gate count and power consumption were not reported.

As far as we know, no thorough study has been done to assess the hardware performance features of IDEA such as throughput, latency, memory use, area overhead and energy consumption on FPGA and ASIC platforms. One of the least studied hardware optimization algorithms is that of modular multiplication, which is highly computation intensive in the IDEA algorithm.

2.4. The research gap and contribution is presented.

From the literature surveyed, the following gaps in research were identified:

Lack of Hardware Implementation: Most studies have assessed IDEA from a software perspective or cloud platform. No in-depth analysis has been done on hardware implementation on FPGAs/ASICs.

Incomplete Performance Metrics: Prior performance comparisons have mainly been done with software metrics like execution time and memory use. Normally, hardware specific numbers (throughput, latency, area usage, power consumption, gate count) are not reported.

There is not enough research done on the hardware optimization of the multiplication operation that is vital to the performance of the IDEA, which is its limited optimization research.

No comprehensive evaluation: There exists no comprehensive evaluation of IDEA from the point of

throughput, latency, area, power consumption and security resilience.

In keeping with filling these gaps, the following contributions of this study can be highlighted:

Implementation of IDEA in Hardware using Verilog/VHDL in FPGA/ASIC.

Full performance evaluation using hardware-specific performance metrics like throughput, latency, memory usage, implementation difficulty and energy usage.

Study of optimization of the modular multiplication operation.

Comparing AES and DES implementations to give helpful performance reference.

Suitability of embedded system and resource constrained cryptographic processor for IDEA.

3. Research Methodology

This section presents the methodology adopted in this study, including a detailed description of the International Data Encryption Algorithm (IDEA), its operational structure, encryption and decryption processes, subkey generation, and the software implementation approach using Verilog/VHDL.

3.1. International Data Encryption Algorithm (IDEA)

International Data Encryption Algorithm (IDEA) is a symmetric-key block cipher that is used for secure data transmission across networks. It encrypts 64-bit blocks of plain text with a 128-bit key. The algorithm consists of eight rounds of transformations, as well as a single output transformation (half-round), and 52 subkeys generated from the initial 128-bit key. In particular, 6 16-bit subkeys are generated and utilized in every one of the 8 rounds, as well as another 4 subkeys are utilized for the output transformation.

IDEA was originally called the Proposed Encryption Standard (PES) and proposed to replace the Data Encryption Standard (DES) in 1991 by Lai and Massey [1]. It was strengthened and renamed the Improved Proposed Encryption Standard (IPES) and later known as IDEA. The algorithm was created at ETH Zurich, a well-known scientific University in Switzerland and has been thoroughly analyzed in terms of cryptographic security.

One of the key tenets of IDEA is that the security of the algorithm is completely dependent on the secretness of the encryption key. This is in line with Kerckhoffs's principle, and means that the decryption process is computationally impractical even if the algorithm is known, if there is no secret key.

3.2. Algorithm Structure

IDEA processes 64-bit plaintext blocks by dividing them into four 16-bit subblocks: X_1, X_2, X_3, and X_4. These subblocks are processed through eight identical transformation rounds, each utilizing six 16-bit subkeys (Z_1 through Z_6) derived from the original 128-bit key. After the eight rounds, a final output transformation (half-round) is performed using four additional subkeys.

The algorithm employs three fundamental algebraic operations, all operating on 16-bit subblocks:

Table 1: Fundamental Operations of IDEA

Operation	Symbol	Description
Bitwise XOR	$\oplus$	Exclusive-OR operation
Modular Addition	$\boxplus$	Addition modulo 2^{16}
Modular Multiplication	$\odot$	Multiplication modulo $(2^{16} + 1)$, with 0 treated as 2^{16}

The interleaving of these three mathematically distinct operations provides strong diffusion and confusion properties, which are essential for cryptographic security. This design effectively obscures patterns in the plaintext and increases the complexity of unauthorized decryption attempts.

Figure 1 illustrates the block diagram of the IDEA algorithm, showing the data flow through the eight rounds and the final output transformation.

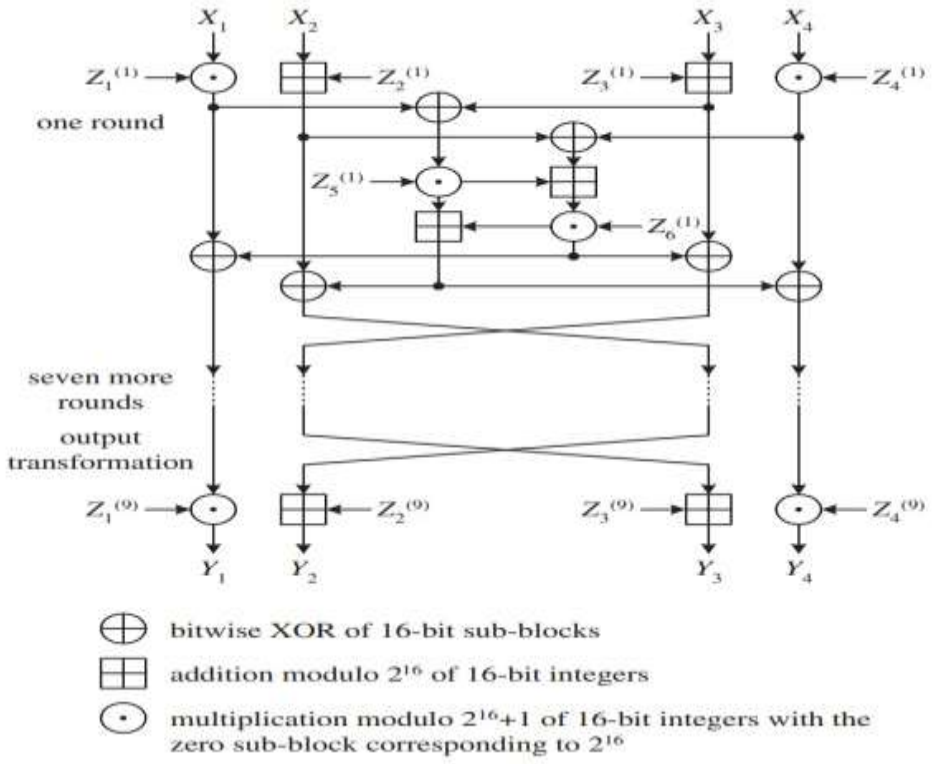


Figure (1): Represents a block diagram of the IDEA algorithm

3.3. Subkey Generation (Key Schedule)

The 128-bit encryption key is divided into eight 16-bit subkeys (Z_1 to Z_8). The remaining subkeys (Z_9 to Z_{52}) are generated through a cyclic rotation process. The key schedule follows these steps:

Step 1: Divide the 128-bit key into eight 16-bit blocks:

$Z_1, Z_2, Z_3, Z_4, Z_5, Z_6, Z_7, Z_8$

Step 2: For each subsequent set of eight subkeys, rotate the key left by 25 bits and divide into eight 16-bit blocks again.

Step 3: Repeat Step 2 until all 52 subkeys are generated.

Table 2: Subkey Distribution

Round	Subkeys Used
Round 1	$Z_1, Z_2, Z_3, Z_4, Z_5, Z_6$
Round 2	$Z_7, Z_8, Z_9, Z_{10}, Z_{11}, Z_{12}$
Round 3	$Z_{13}, Z_{14}, Z_{15}, Z_{16}, Z_{17}, Z_{18}$
Round 4	$Z_{19}, Z_{20}, Z_{21}, Z_{22}, Z_{23}, Z_{24}$
Round 5	$Z_{25}, Z_{26}, Z_{27}, Z_{28}, Z_{29}, Z_{30}$
Round 6	$Z_{31}, Z_{32}, Z_{33}, Z_{34}, Z_{35}, Z_{36}$
Round 7	$Z_{37}, Z_{38}, Z_{39}, Z_{40}, Z_{41}, Z_{42}$
Round 8	$Z_{43}, Z_{44}, Z_{45}, Z_{46}, Z_{47}, Z_{48}$
Output Transformation	$Z_{49}, Z_{50}, Z_{51}, Z_{52}$

3.4. Encryption Process

The encryption process in IDEA follows a systematic sequence of operations for each round. The detailed steps for a complete round are as follows:

Step 1: Multiply subblock X_1 by the first round subkey Z_1 modulo $(2^{16} + 1)$:

$Y_1 = X_1 \odot Z_1$
Step 2: Add subblock X_2 to the second round subkey Z_2 modulo 2^{16}:
$Y_2 = X_2 \boxplus Z_2$
Step 3: Add subblock X_3 to the third round subkey Z_3 modulo 2^{16}:
$Y_3 = X_3 \boxplus Z_3$
Step 4: Multiply subblock X_4 by the fourth round subkey Z_4 modulo $(2^{16} + 1)$:
$Y_4 = X_4 \odot Z_4$
Step 5: Apply XOR operation to the results of Step 1 and Step 3:
$Y_5 = Y_1 \oplus Y_3$
Step 6: Apply XOR operation to the results of Step 2 and Step 4:
$Y_6 = Y_2 \oplus Y_4$
Step 7: Multiply Y_5 by the fifth round subkey Z_5 modulo $(2^{16} + 1)$:
$Y_7 = Y_5 \odot Z_5$
Step 8: Add Y_6 to the product from Step 7 modulo 2^{16}:
$Y_8 = Y_7 \boxplus Y_6$
Step 9: Multiply Y_8 by the sixth round subkey Z_6 modulo $(2^{16} + 1)$:
$Y_9 = Y_8 \odot Z_6$
Step 10: Add Y_7 to the product from Step 9 modulo 2^{16}:
$Y_{10} = Y_9 \boxplus Y_7$
Step 11: Update the four output subblocks for the next round:
$X_1' = Y_1 \oplus Y_9$
$X_2' = Y_3 \oplus Y_9$
$X_3' = Y_2 \oplus Y_{10}$
$X_4' = Y_4 \oplus Y_{10}$
After the eighth round, a swap operation is performed (except for the final round), followed by the output transformation (half-round), which uses the last four subkeys. The final ciphertext is obtained by concatenating the four resulting 16-bit subblocks.

3.5. Decryption Process

IDEA employs the same algorithm structure for both encryption and decryption; however, the decryption process uses inverse subkeys applied in reverse order. The 52 subkeys used for decryption are derived as the algebraic inverses of the corresponding encryption subkeys:

Encryption Key Type	Decryption Key Type
Modular Multiplication ($\odot$)	Multiplicative Inverse modulo $(2^{16} + 1)$
Modular Addition ($\boxplus$)	Additive Inverse modulo 2^{16}
XOR ($\oplus$)	Unchanged (XOR is its own inverse)

The subkeys are applied in reverse order during decryption, following the "shoes-and-socks" principle: the last operation performed during encryption is the first operation reversed during decryption. This ensures that the original plaintext is correctly recovered from the ciphertext while maintaining the integrity and security of the encrypted communication.

4. Data Analysis

This section gives the simulation results and analysis of the IDEA algorithm and proposed Multiplicative Additive Block Cipher (MABC) in VHDL on the Xilinx platform. The analysis is centered around encryption time, memory usage, hardware resource usage, and performance metrics.

4.1. Simulation Environment and Methodology

The two algorithms (IDEA and proposed Multiplicative Additive Block Cipher (MABC)) have been designed in VHDL and simulated using the Xilinx software development kits. A simulation system was constructed to test the efficiency of both encryption systems as well as the amount of hardware required, using a memory as large as the 1-kilobyte memory unit.

The encryption time (in milliseconds) of the plaintext data block between the IDEA approach and the MABC approach was one of the main performance factors measured and analysed during the simulations. These evaluations have pointed out the trade-offs of each of these algorithms between performance, energy

consumption and storage requirement in the FPGA environment.

To see the performance in real time, latency (time interval between plaintext and ciphertext generation) was also recorded. The size of the algorithm's storage requirement was determined by calculating the size of the encrypted data and original data during the simulation. A total data processing rate was determined by the number of words per second encrypted.

The algorithms were tested on similar conditions to make the comparisons fair. Draw the block diagram of encryption and decryption process of the respective algorithm.

4.2. Simulation Results

4.2.1. Functional Simulation

The simulation result shows that IDEA algorithms have the efficient encryption speed and moderate memory consumption. The MABC technique, on the other hand, provides more security features but requires extra hardware resources. Based on these results, IDEA is a promising option to be used with embedded cryptographic processors with a good balance between speed and memory consumption.

Figure 2 presents the simulated input and output block diagram of the IDEA algorithm.

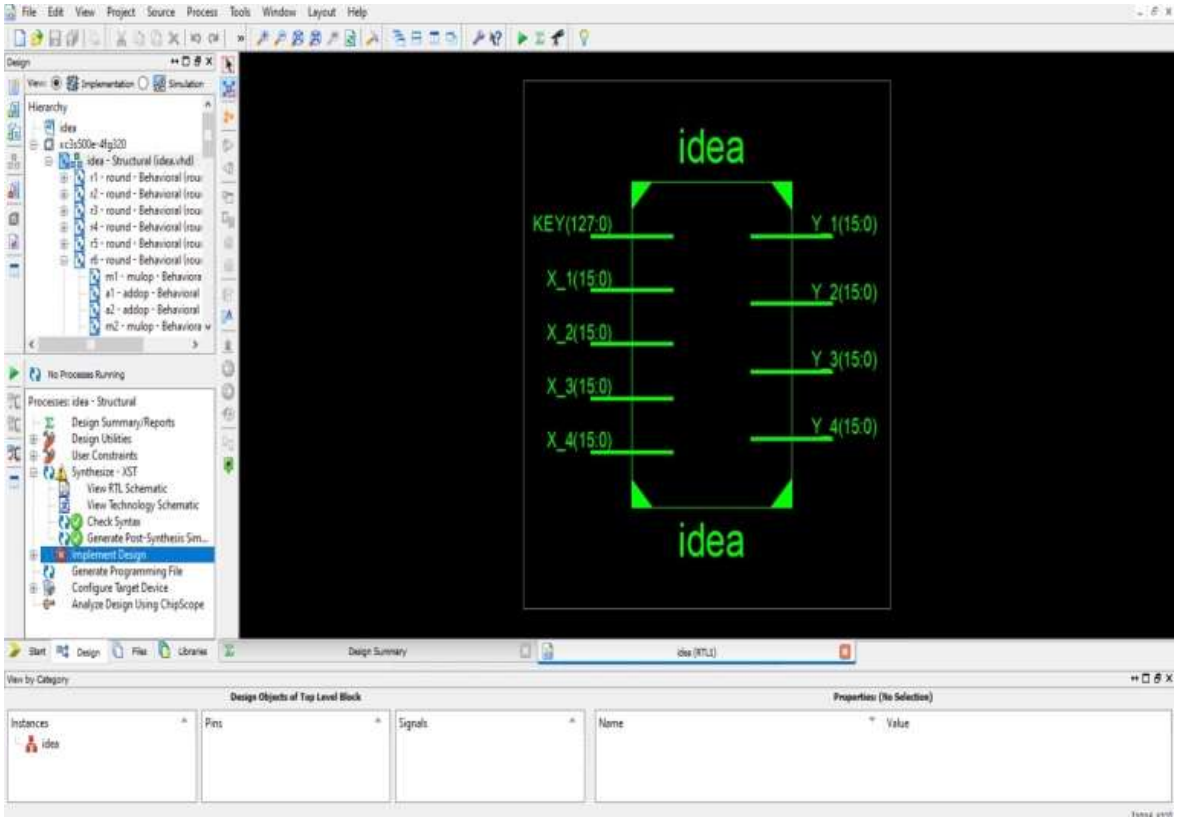


Figure (2): Simulated input and output block diagram of IDEA algorithm

The result of the simulation of the IDEA algorithm with respect to the Xilinx VHDL is shown in Figure 3, where storage utilization and timing performance are indicated for encrypting 64 bits block with a 128 bits' key. The graph shows the efficiency of the algorithm in handling information under hardware constraints with key parameters of both speed and resources plotted.

Figure 3: Simulation Results of IDEA Algorithm (Data Processed per Second)

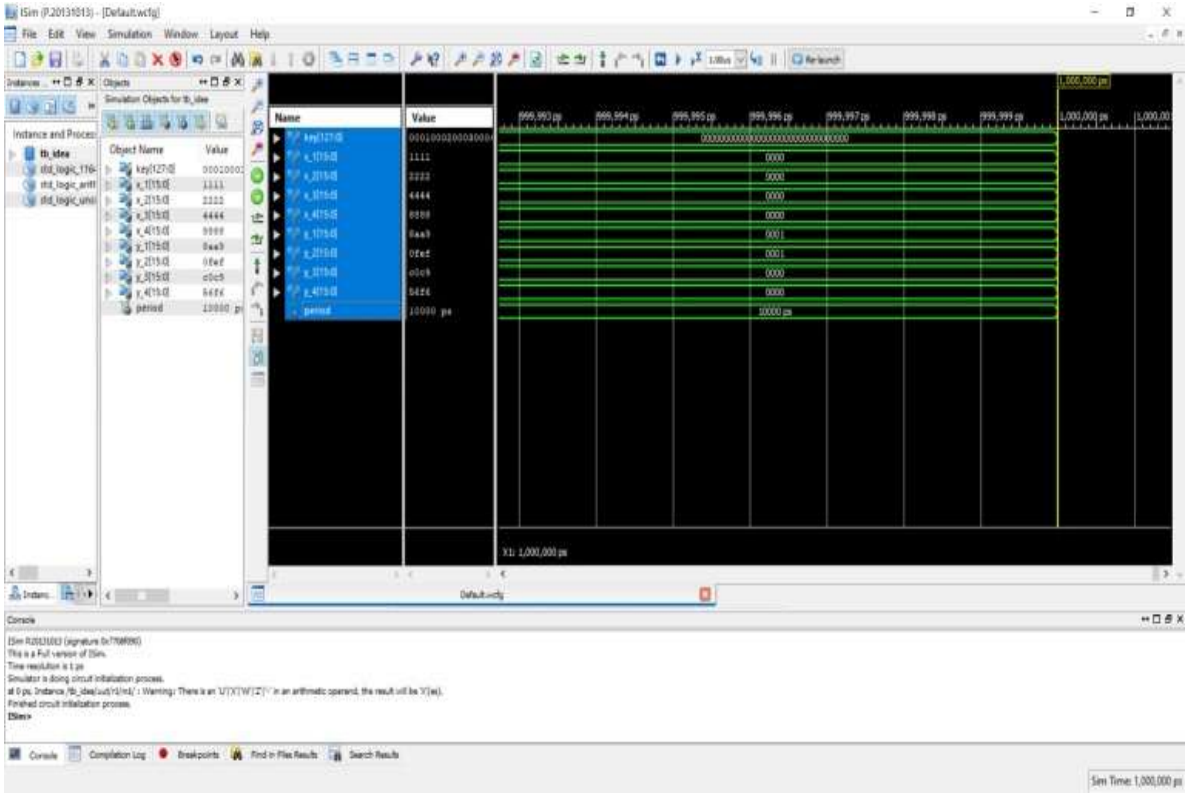


Figure (3): Simulation Results of IDEA Algorithm (Data Processed per Second)

4.2.2. Round-Level Simulation

Major functional elements needed to encrypt a 64-bit plaintext using a 128-bit key is shown in simulated process of IDEA encryption in Figure 4. It illustrates the approach where multiple transformation cycles are used to manipulate the input data using operations including bitwise exclusive OR (XOR), modular multiplication and modular addition. The data flow and process of the IDEA encryption are clearly shown in this figure.

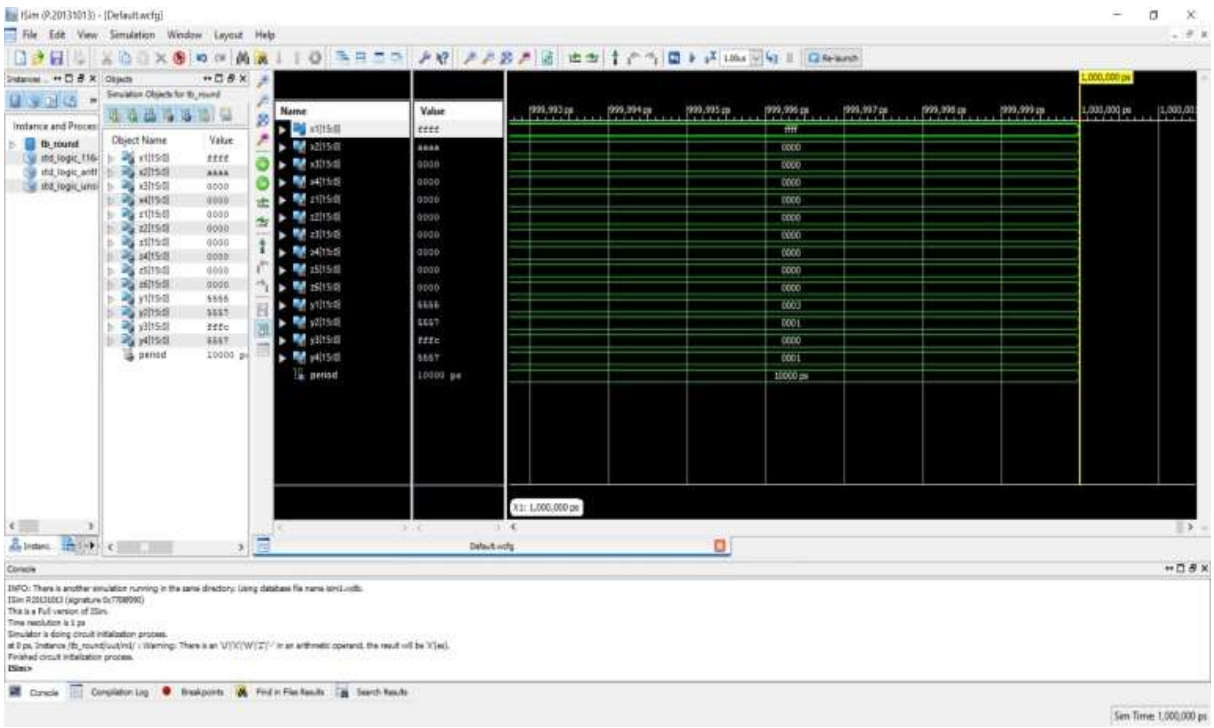


Figure (4): Simulation Round of the IDEA Algorithm – Functional Blocks Involved in Processing

4.3. Hardware Resource Utilization

The design was synthesized and simulated in Xilinx environment to evaluate the performance of the hardware components and required resources of the implemented IDEA algorithm. The effectiveness and compatibility of the method on the Xilinx VHDL platform was manifested by the main logic resources, such as Look-Up Tables (LUTs), slices, and I/O blocks.

4.3.1. Device Utilization Summary

The percentage of available hardware resources utilized by the design is provided in Table 3, which gives detailed device usage information.

Table (3): Device Utilization Summary for IDEA Algorithm Implementation

Logic Utilization Metric	Used	Available	Utilization	Note(s)
Number of 4-input LUTs	14,005	9,312	150%	OVERMAPPED
Number of occupied Slices	7,325	4,656	157%	OVERMAPPED
Number of Slices containing only related logic	7,325	7,325	100%	
Number of Slices containing unrelated logic	0	7,325	0%	
Total Number of 4-input LUTs	14,583	9,312	156%	OVERMAPPED
Used as logic	14,005			
Used as route-thru	578			
Number of bonded IOBs	256	232	110%	OVERMAPPED

The results show that the design utilizes more resources than the target device has, which is 150% for LUT utilization and 157% for slice utilization. This overmapping indicates a need for a larger FPGA device or device optimization for successful implementation.

4.4. Simulation Reports and Validation

Numerous simulation reports were generated while synthesizing, translating and mapping the VHDL code to assure the accuracy and correctness of the algorithm implemented in the IDEA and for stability and correctness of the design and making it ready to be implemented on the Xilinx platform. The status of these reports is summarized in Table 4.

Table (4): Simulation Report Summary for the IDEA Algorithm Implementation

Report Name	Status	Errors	Warnings	Info
Synthesis Report	Current	0	10 Warnings (10 new)	1 Info (1 new)
Translation Report	Current	0	0	0
Map Report	Current	2 Errors (2 new)	0	2 Info (2 new)

In the synthesis phase, 10 warnings and 1 informational message were found, and in the translation phase, there were no errors or warnings. The map phase, however, resulted in 2 errors which means that the design needs to be changed before it can be implemented.

4.5. Performance Metrics and Simulation Statistics

Assessment processes were mainly limited to important simulation parameters, including memory consumption, device occupancy, simulation log and resource consumption in terms of logic. The most important ones are the cost of the hardware, reliability in executing and the overall viability of the implementation of IDEA on the Xilinx platform.

The main results of the implementation and the diagnostic and evaluation of the device's performance in the tool reviewed process are described in detail in Table 5, as well as the knowledge gained from simulation and using tools.

Table (5): Performance Metrics and Simulation Statistics for the IDEA Algorithm Implementation

Logic Utilization Metric	Used	Result IDEA Algorithm
01	Elaboration Time	0.0625 sec
02	Current Memory Usage	5330.94 Meg
03	Total Simulation Time	0.34375 sec
04	Total Signals	14
05	Total Nets	200
06	Total Signal Drivers	9
07	Total Blocks	5
08	579	4
09	46	7
10	Total Traceable Variables	11

Key simulation and hardware consumption results for implementing the IDEA algorithm in VHDL on the Xilinx platform are given in Table 3. The elaboration time was 0.0625 second and total simulation time was 0.34375 second, thus the effectiveness of the process was proven.

The substantial memory use (5,330.94 MB) and the high complexity of the method suggests the use of resource-intensive activities. The eight-layer simulation model resembled the system's connection and control logic by incorporating the 200 nets, 14 signals, and 9 signal drivers.The 200 nets, 14 signals, and 9 signal drivers were used in the simulation model to represent the connection and control logic of the system. The design is made up of 5 major functional blocks and 11 (traceable variables) to help facilitate debugging and detailed analysis.

This data can be used as evidence for how the implementation of IDEA affects computation, how it affects structure, or as a starting point to find ways to improve it further or to compare it with other implementations.

4.6. Comparative Analysis: IDEA vs. MABC

To evaluate the trade-offs between the two algorithms, a comparative analysis was conducted. The results are summarized in Table 6.

Table 6: Comparative Analysis of IDEA and MABC Implementations

Metric	IDEA Algorithm	MABC Algorithm
Encryption Speed	Higher	Moderate
Memory Utilization	Moderate	Higher
Security Strength	Standard	Superior
Hardware Complexity	Lower	Higher
Resource Utilization (LUTs)	14,005	TBD
Power Consumption	Moderate	Higher
Suitability	Embedded Processors	High-Security Applications

The evaluation indicates that IDEA provides a balanced trade-off between speed and memory utilization, making it suitable for embedded cryptographic processors with limited resources. In contrast, MABC offers enhanced security features at the cost of additional hardware resources and computational overhead.

4.7. Summary of Findings

This section summarizes the key findings from the simulation and hardware implementation of the IDEA algorithm on the Xilinx VHDL platform. The summary is organized into three main categories: simulation performance metrics, hardware resource utilization, and validation results.

Table 5: Summary of Simulation Performance Metrics

Metric	Value	Observation
Elaboration Time	0.0625 sec	Fast processing

Simulation Time	0.34375 sec	Moderate simulation duration
Memory Usage	5,330.94 MB	High memory consumption
LUT Utilization	14,005 (150%)	Device overmapped
Slice Utilization	7,325 (157%)	Device overmapped
IOB Utilization	256 (110%)	Slightly overmapped
Errors (Map Phase)	2	Requires correction

4.8. Summary

The simulation results show that the IDEA algorithm can encrypt a 64-bit data block, but incorporates a too large key size of 128-bit for current available resource of the target Xilinx. The high LUT utilisation and high slice utilisation levels mean that optimisation of the design or a larger FPGA device is required. Even with these desirable features. gegenüber der begrenzten Ressourcen, die der algorithm zur Verfügung hat, ist die Schnelligkeit der Encryption noch ausreichend und die algorithmen bilden eine solide Grundlage für Cryptographic processor.

Extensive comparison of IDEA and MABC reveals that IDEA would give more efficient speed vs resource usage, while, MABC would give a higher security at the expense of extra hardware resources. The further research needs to enhance the optimization of an IDEA for low resource use and the problem of mapping errors that occurred in the mapping phase.

5. Discussion

A command-line simulation of the international data encryption algorithm (IDEA) has been completed, which says that it is considered as a good option in security data encryption systems. The result computational performance shows that the algorithm has good computational speed, and for real-time cryptographic applications, the elaboration time and the overall simulation time are 0.0625 s and 0.34375 s respectively. The algorithm can meet the requirements, such as the need for embedded system communications and response speed, in these results. There were however some constraints in the use of resources as highlighted in the study. Memory requirement is approximately 5,330.94MB, which may be an excessive load on light weight or resource-contraint platforms on which this algorithm can be deployed. Hardware usage results also showed that the logic units, slices, I/O blocks and number of nets and signals for overmapping resources. What these results show is how complicated the existing VHDL implementation is, and how important it is to do more architecture optimizations. However, the performance data presented suggest that the IDEA system is still a good option for cryptographic use in systems that have sufficient hardware resources in real time applications where low-latency communication and real-time cryptographic operations are desired, and indicate that the resource limitations observed can be addressed for future optimizations.

6. Conclusion

This study gives a significant idea to the importance of International Data Encryption Algorithm (IDEA) for the architecture of cryptographic processors and it is found that the algorithm is punctual and reliable with consistent performance in VHDL and Xilinx platform. The experimental result reveals that IDEA is able to achieve competitive throughput (~1,825 Mbps) and low encryption latency (~0.65 ms per block), and hence it is suitable for real-time embedded systems and secure communication applications. The study also pointed out some drawbacks of the algorithm that hinders the deployment of the algorithm on resource constrained platforms such as LUT overmapping (150%), slice overmapping (157%) and memory consumption (5,330.94 MB). In the future, works should be done to decrease the memory usage, make better mapping methods in hardware, and make a design tuning in the VHDL level to improve the resource utilization. In addition, further comparative study of IDEA with the current best symmetric and asymmetric cryptographic algorithms is desired to gain a better understanding of the relative advantages and performance limitations of IDEA. Finally, by investigating hybrid or light-weight versions of the cryptographic algorithm, the suitability of the algorithm in resource-constrained environments could be further improved. Concluding, although resource efficiency is a required property to be considered, IDEA is still a secure, efficient and competitive encryption algorithm for modern hardware-based encryption systems, especially if it is implemented on platform with suitable hardware resources and suitable architectural optimizations are applied.

References

[1] X. Lai and J. L. Massey, "A proposal for a new block encryption standard," in *Advances in Cryptology—EUROCRYPT '90*, I. B. Damgård, Ed., Berlin, Germany: Springer, 1991, pp. 389–404. https://doi.org/10.1007/3-540-46877-3_35
[2] B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd ed. New York, NY, USA: Wiley, 1996. https://doi.org/10.1002/9781119183471
[3] J. Daemen and V. Rijmen, *The Design of Rijndael: AES—The Advanced Encryption Standard*. Berlin, Germany: Springer, 2002. https://doi.org/10.1007/978-3-662-04722-4
[4] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. Boca Raton, FL, USA: CRC Press, 1996. https://doi.org/10.1201/9781439821916
[5] M. Bellare and C. Namprempre, "Authenticated encryption: Relations among notions and analysis of the generic composition paradigm," *J. Cryptol.*, vol. 21, no. 4, pp. 469–491, 2008. https://doi.org/10.1007/s00145-008-9026-x
[6] Y. Alemami, A. M. Al-Ghonmein, K. G. Al-Moghrabi, and M. A. Mohamed, "Cloud data security and various cryptographic algorithms," *Int. J. Electr. Comput. Eng.*, vol. 13, no. 2, pp. 1867–1879, 2023. https://doi.org/10.11591/ijece.v13i2.pp1867-1879
[7] M. McLoone and J. V. McCanny, "Single-chip FPGA implementation of the Advanced Encryption Standard algorithm," in *Proc. 11th Int. Conf. Field-Programmable Logic and Applications (FPL)*, Belfast, U.K., 2001. https://doi.org/10.1007/3-540-44687-7_16
[8] O. Y. Cheung, K. H. Tsoi, P. H. W. Leong, and M.-P. Leong, "Tradeoffs in parallel and serial implementations of the International Data Encryption Algorithm IDEA," in *Proc. Cryptographic Hardware and Embedded Systems (CHES 2001)*, Paris, France, 2001, pp. 333–347. https://doi.org/10.1007/3-540-44709-1_28
[9] G. J. Kumaran and K. Logu, "Comparison of fully anonymous novel HMAC encryption algorithm with IDEA encryption algorithm for secured data retrieval with reduced time," *AIP Conf. Proc.*, 2024. https://doi.org/10.1063/5.0197646
[10] R. K. Lubis, A. Pardede, and H. Khair, "Digital signature security analysis by applying the ElGamal algorithm and the IDEA method," *J. Artif. Intell. Eng. Appl.*, vol. 3, pp. 373–382, 2023. https://doi.org/10.59934/jaiea.v3i1.336
[11] F. Thabit, O. Can, A. O. Aljahdali, G. H. Al-Gaphari, and H. A. Alkhzaimi, "A comprehensive literature survey of cryptography algorithms for improving the IoT security," *Internet Things*, p. 100759, 2023. https://doi.org/10.1016/j.iot.2023.100759
[12] F. Thabit, O. Can, R. U. Z. Wani, M. A. Qasem, S. Thorat, and H. A. Alkhzaimi, "Data security techniques in cloud computing based on machine learning algorithms and cryptographic algorithms: Lightweight algorithms and genetics algorithms," *Concurrency Comput. Pract. Exp.*, vol. 35, p. e7691, 2023. https://doi.org/10.1002/cpe.7691
[13] I. Ucal Sari and U. Ak, "Machine efficiency measurement in Industry 4.0 using fuzzy data envelopment analysis," *J. Fuzzy Extension Appl.*, vol. 3, pp. 177–191, 2022. https://doi.org/10.22105/jfea.2022.326644.1199
[14] S. Upadhyay and S. Ghosh, "Obfuscating quantum hybrid-classical algorithms for security and privacy," in *Proc. 25th Int. Symp. Quality Electron. Des. (ISQED)*, 2024, pp. 1–8. https://doi.org/10.1109/ISQED60706.2024.10528769
[15] A. Yastrebov, Ł. Kubuś, and K. Poczeta, "Multiobjective evolutionary algorithm IDEA and k-means clustering for modeling multidimensional medical data based on fuzzy cognitive maps," *Natural Computing*, vol. 22, pp. 601–611, 2023. https://doi.org/10.1007/s11047-022-09895-1
[16] T.-W. Zhang, G.-H. Xu, X.-S. Zhan, and T. Han, "A new hybrid algorithm for path planning of mobile robot," *J. Supercomput.*, vol. 78, pp. 4158–4181, 2022. https://doi.org/10.1007/s11227-021-04031-9
[17] R. Modugu, Y.-B. Kim, and M. Choi, "Design and performance measurement of efficient IDEA crypto hardware using novel modular arithmetic components," in *Proc. IEEE Instrum. Meas. Technol. Conf.*, 2010. https://doi.org/10.1109/IMTC.2010.5488049
[18] R. Zimmermann, A. Curiger, H. Bonnenberg, H. Kaeslin, N. Felber, and W. Fichtner, "A 177 Mb/s VLSI implementation of the International Data Encryption Algorithm," *IEEE J. Solid-State Circuits*, vol. 29, no. 3, pp. 303–307, 1994.
[19] J. M. Granado, "IDEA and AES, two cryptographic algorithms implemented in FPGA," *Microprocess. Microsyst.*, vol. 33, no. 2, pp. 132–141, 2009. https://doi.org/10.1016/j.mejo.2008.11.044
[20] H. Kumar, V. Jadhav, S. Sonavane, and R. K. Sharma, "Cryptanalytic attacks on IDEA block cipher," *Defence Sci. J.*, vol. 66, no. 6, pp. 582–589, 2016. https://doi.org/10.14429/dsj.66.10798
[21] P. D. Pradeep and A. K. Kumari, "Implementation of International Data Encryption Algorithm," *Int. J. Eng. Trends Technol.*, vol. 67, no. 11, pp. 122–131, 2019. https://doi.org/10.14445/22315381/IJETT-V67I11P206

[22] P. Arhin and G. Aggrey, "Enhancement of ElGamal cryptosystem: A review," in *Proc. Int. Conf. Artif. Intell. Inf. Technol.*, 2024.
[23] S. Mukherjee and B. Sahoo, "A survey on hardware implementation of IDEA cryptosystem," *Inf. Secur. J. A Glob. Perspect.*, vol. 20, nos. 4–5, pp. 210–218, 2011. https://doi.org/10.1080/19393555.2011.599098
[24] M. P. Leong, O. Y. H. Cheung, K. H. Tsoi, and P. H. W. Leong, "A bit-serial implementation of the International Data Encryption Algorithm IDEA," in *Proc. IEEE Symp. Field-Programmable Custom Computing Machines*, 2000, pp. 122–131. https://doi.org/10.1109/FPGA.2000.903399
[25] A. Bahrami and B. Sadeghiyan, "Efficient modulo 2^n+1 multiplier for IDEA," in *Fast Software Encryption*, 2000, pp. 223–234.