Volume: 12 Issue: VII Month of publication: July 2024

DOI: https://doi.org/10.22214/ijraset.2024.63723

A Unified Architecture For AES/Present Ciphers and Its Usage in a Soc Environment

Vatam Teja[1], G Naga Deepthi[2], R.L.B. Prasad Reddy[3]
[1]PG Scholar, Dept of ECE, Srinivasa Institute of Technology and Science, Kadapa.
[2]Assistant Professor, [3]Associate Professor, Dept of ECE, Srinivasa Institute of Technology and Science, Kadapa

Abstract: In order to monitor the operating status of vehicles, it is necessary to collect vehicle operating data in real time through IoT devices and analyze these data. However, the collected data has the characteristics of multi-source heterogeneous, network resources are limited and server performance is poor. It is difficult to truly realize data processing in real time. In addition, data needs to be transmitted over the network, it is particularly important to ensure the safety of data transmission. Considering the above problems, it is necessary to structure the data and define a unified data format to facilitate data transmission and analysis. At the same time, improve the server communication program and improve the server's concurrent processing capabilities. In addition, considering that data needs to be transmitted over the network, in order to ensure that the data is not stolen or tampered with, the PRESENT lightweight encryption algorithm is adopted to ensure the safety of data transmission. Compared with encryption algorithms such as AES, this algorithm has much lower hardware requirements. This article combines the characteristics of the project and uses the number of communications between the device and the server to achieve the dynamic key update which is approximately one-time pad, and greatly improves the security of the data.

Keywords: IoT, vehicle monitoring, present algorithm

I. INTRODUCTION

Information might be transmitted quickly over long distances using this way. The ability to connect various systems, devices, and eventually anything else is now available. The "Internet of Things" is now a reality. Smart houses, smart grids, smart keys, and smart public transportation are all part of the Internet of Things (IoT). There are many benefits to the Internet of Things (IoT), but there are also substantial negatives, such as privacy and security problems due to the ease with which an attacker can access them. At both the transmitter and receiver end, the data is encoded and decrypted using cyphers (cryptographers' algorithms for encoding and decoding). Because everyone is familiar with the cypher technique, an attacker can only steal data if he has the secret cypher key. IoT concepts such as IEEE 802.15.4, LoraWAN, SigFox, and ZWave all use AES [1] since it is the only block cypher that can meet the stringent requirements for compact size and cheap power, as well as high security. To begin, in AES designs, byte substitution was performed using lookup tables in S-boxes. Due to the enormous memory footprint and high hardware requirements, GF arithmetic was created as an alternative. Arithmetic circuits were built using finite field theory. Field operations could make good use of GF(28), as it often only deals with 8-bit data types. Many mathematical operations and circuits become extremely difficult to accomplish if GF is described over GF (28). To calculate the multiplicative inverse of a 7th-degree polynomial, for example, one must need an 8th-degree polynomial. Because of finite field decomposition, composite field arithmetic is possible (CFA). Isomorphic mapping is a finite field property that allows field elements to be transformed between fields. To move an element to GF(((22) 2) 2), use GF(28). The inverse of a first-degree polynomial (modulo a second-degree polynomial) can be calculated using this method, for example. In order to have a clear understanding of the final work, the AES architecture and S-box finite field arithmetic have been thoroughly examined.

II. LITERATURE SURVEY

1) *Adam J. Elbirt, W. Yip, B. Chetwynd, and C. Paar" An FPGA-Based Performance Evaluation of the AES Block Cipher Candidate Algorithm Finalists"*

It was decided which algorithm would be the Advanced Encryption Algorithm through efficiency testing of hardware and software implementations of candidate algorithms. Field-programmable gate arrays (FPGAs) can be used to create cryptographic algorithms because of their physical security and potential for higher performance than software alternatives (FPGAs). This article examines the significance of FPGA implementations of proposed algorithms for the Advanced Encryption Standard (AES). There are a number of architectural options for each method.

Current and future highbandwidth applications demand highthroughput solutions, which are being strongly emphasised in the design process. Lastly, the FPGA implementations of each approach will be compared to determine which one is the greatest candidate for commercial application.

2) Johannes Wolkerstorfer, Elisabeth Oswald, Mario Lamberger" An ASIC Implementation of the AES SBoxes"

The author of this piece has built SBoxes from the Advanced Encryption Standard (AES) in hardware (AES). Arithmetic operations in the finite field GF are used to produce an 8-bit output in the SBoxes (28). We show that this function and its inverse can be efficiently computed using combinational logic. It is less efficient to perform table lookups utilising read-only memory. There is a great deal of overlap between encryption and decryption functions. Having a tiny die size and fewer transistors, this design is well-suited to semi-custom design approaches like standard-cell design since it is simple to pipeline. A 0.6-micron standard cell with a delay of less than 15ns, which is equivalent to a clock frequency of 70 MHz, may be built in 0.108mm2 of area. These results were accomplished without the use of any speed-enhancing methods like pipelining.

3) Tanzilur Rahman, Shengyi Pan, Qi Zhang" Design of a High Throughput 128- bit AES"

There is good throughput with Xilinx Spartan III XC3S1000 hardware-based FPGA implementation of the 128-bit Advanced Encryption Standard (AES). Bus width in the architecture is 32 bits. A more rapid design process was made possible by the application of the pipelining method. For the purpose of evaluating the effectiveness of the SubByte approach, it has been implemented both utilising the composite field technique and on a fixed Rom. To achieve rates of 1.11 Gbps to 3.22 Gbps, SBox and key Expansion methods were flawlessly coupled. The four different combinations of testing that the complete design underwent allowed for a more in-depth study (composite field and Rom for both sub bytes and key expansion). Each method has its own statistical analysis and performance graphs.

III. EXISITING SYSTEM

There was a pressing need for security because so many of them used a variety of approaches to safeguard desktop files. DES and IDEA are two of the most widely used algorithms nowadays. However, these algorithms are susceptible to being exploited at some point. The file cannot be decrypted if the secret key used to encrypt and decode the file is not the same. Right-clicking to delete files once data has been encrypted is no longer safe. In this circumstance, it is simple to delete encrypted files. Because of these issues, the proposed system adds additional measures to secure disc-based information. Advanced Encryption Standard (AES) is an encryption and decryption method employed by the United States government. The National Institute of Standards and Technology (NIST) issued a request for opinions on "Development of a Federal Information Processing Standard for Advanced Encryption Standard" on January 2, 1997. The National Institute of Standards and Technology (NIST) was looking for more secure alternatives to DES, IDEA, and RSA. The Data Encryption Standard (DES) was vulnerable to brute-force attacks because of its 56-bit effective key length. For AES candidates, symmetric-block cyphers with different key lengths were required. Having an algorithm that is easily implementable in hardware and software while also being explicitly stated was a requirement. The AES algorithm's design principles are symmetry and processing efficiency. AES was adopted by the National Institute of Standards and Technology (NIST) after a five-year process (AES). There are 128 bits in each of the blocks of AES. Key lengths of 128, 192, and 256 are all AES-256s, and each is called to as such.

IV. PROPOSED SYSTEM

The overall architecture of the vehicle management system is shown in Fig. 1. The architecture is composed of the perception layer, the network layer, the platform layer and the application layer. The perception layer is the core component of the Internet of Things and is also a key part of information collection and is the direct channel of data. This layer is composed of basic sensing components such as RFID tags and readers, various sensors, cameras, GPS and other sensor components, and a network composed of sensors such as RFID network and sensor network. The network layer is mainly composed of various private networks, local area networks, the Internet, mobile communication networks and wireless sensor networks, and is responsible for data transmission and reliable delivery. The platform layer is mainly for applications in the cloud environment, providing basic services required in the process of application development, testing, deployment and operation, including web and application servers, message servers, file storage servers, and management support services such as access control and application deployment, application performance management, usage metering and billing, etc. The application layer is located at the top of the structure of the Internet of Things. Its function is to "process data", that is, to process information through the cloud computing platform.

The application layer is mainly used to calculate, process and mine the data collected by the perception layer to achieve dynamic monitoring, real-time control, precise management and scientific decision-making of the physical world. The core functions of the application layer of the Internet of Things need to complete data management and data processing, and at the same time combine these data with various industry applications.

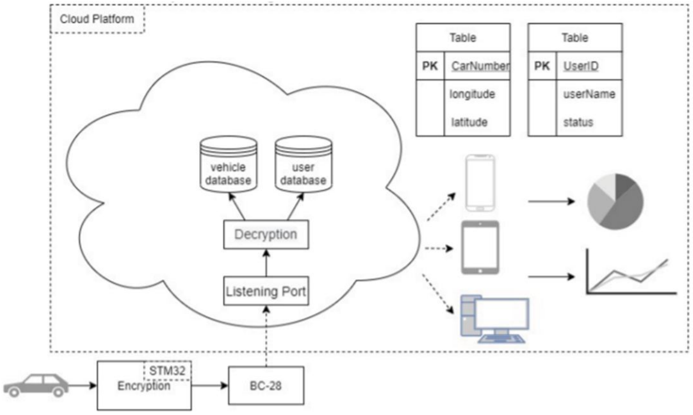


Figure: Architecture of vehicle management cloud platform

processing, which provides a foundation for dynamic monitoring, real-time control, and precise control as well as data mining, and can assist enterprises in precise management and scientific decision-making, In addition, equipment temperature monitoring, user management, equipment management, and location management are also implemented at the application layer. The platform is implemented based on the MVC design pattern, which can reduce the coupling between systems and facilitate later maintenance and function expansion.

A. Encryption Algorithm Design

The PRSENT algorithm uses the SP network structure. The number of iterations is 31 rounds. Each round goes through the SP structure. Each round is composed of three operations: addRoundKey, sBoxLayer, and pLayer.

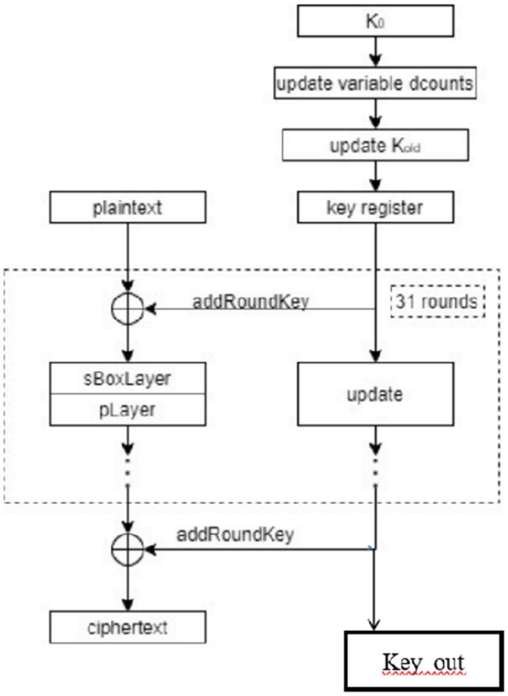


Figure: Description of present algorithm

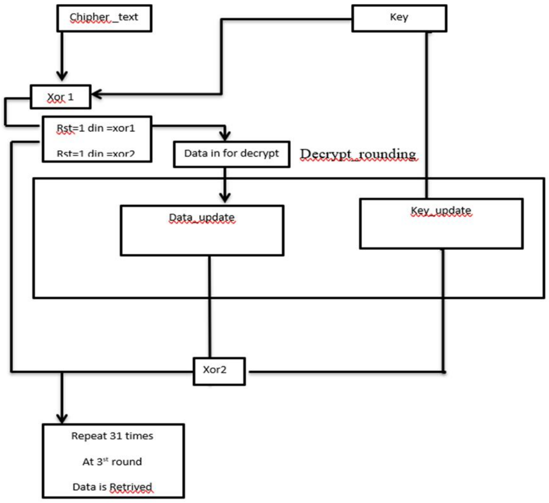


Figure: Decryption Block Diagram

Based on the PRESENT-80 algorithm and combined with actual engineering scenarios, this article realizes dynamic key update, which achieves an effect of approximately one-time pad. Compared with the use of static key for encryption, it increases the difficulty of being deciphered and effectively prevents data from being encrypted, stolen and tampered. The encryption process is implemented as follows: an initial key K is stored on the IoT device and the server, and a variable dCounts is stored to save the number of normal communication between the device and the server. The process is shown in Fig.8. 3. We use this variable dCounts to dynamically update the initial key, the key update rule is as follows:

K = (80dCounts%80)
(dCounts%80)(1)

When encrypting for the first time, dCounts is initialized to 0, and the key K is the initial key. The device uses the initial key for encryption, and the server uses the initial key for decryption. When the device establishes a TCP connection with the server, the ciphertext is transmitted to the server through the Internet. After the server is decrypted normally, the dCounts variable on the server side is incremented. When the TCP connection is disconnected, the dCounts variable on the device side is incremented, and ready to transmit data next time. The dynamic key is realized by using the dCounts variable, and the key update rule can be expressed as formula, which cleverly realizes the synchronous key update.

V. BLOCK CIPHER PRESENT

The deployment of small computing devices, such as RFID tags, conventional algorithms like AES has become insufficient. Because while they are secure against known cryptanalyst is attacks, they also require a large area to implement, an area which cannot be provided by small devices. Lightweight cryptographic algorithms have potential to solve this problem. PRESENT is a very good example of light weight algorithms. It has been designed with the goal of hardware optimization. Therefore, the power consumption and area have been given the utmost consideration. In order to still have a secure algorithm, the designers have found the most suitable trade-off of security, area, and power at the area of 1570GE and simulated power consumption of 5μw for PRESENT-80 and 1886 GE and 3.3μw for PRESENT-128 [6]. Its design was modeled after AES finalist Serpent and DES.

VI. STREAM CIPHER TRIVIUM

TRIVIUM could be considered another good example of a lightweight algorithm. Like PRESENT, TRIVIUM is also a hardware-oriented algorithm. However, while PRESENT is a block cipher, TRIVIUM is a synchronous stream cipher. It was de- signed to see how much can a stream cipher be simplified without compromising its security and is a part of the eSTREAM project [8].

The designers state in [7] that as their main design consideration, they wanted the algorithm to generate key stream bits without linear correlations so that the keystream bits can't be exploited that way. In order to achieve that, they analyze some block ciphers and their linear characteristics, and apply them to stream ciphers by modifying them.

VII. RESULTS

By Using ISE DESIGN SUITE Project Navigator, Xilinx 14.7 version in the simulation procedure the better ooutputs.

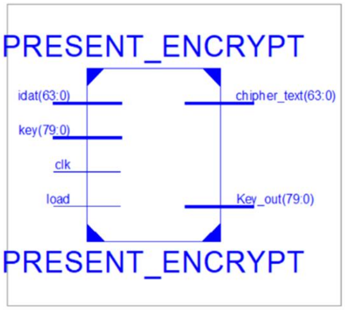


Figure : Encryption

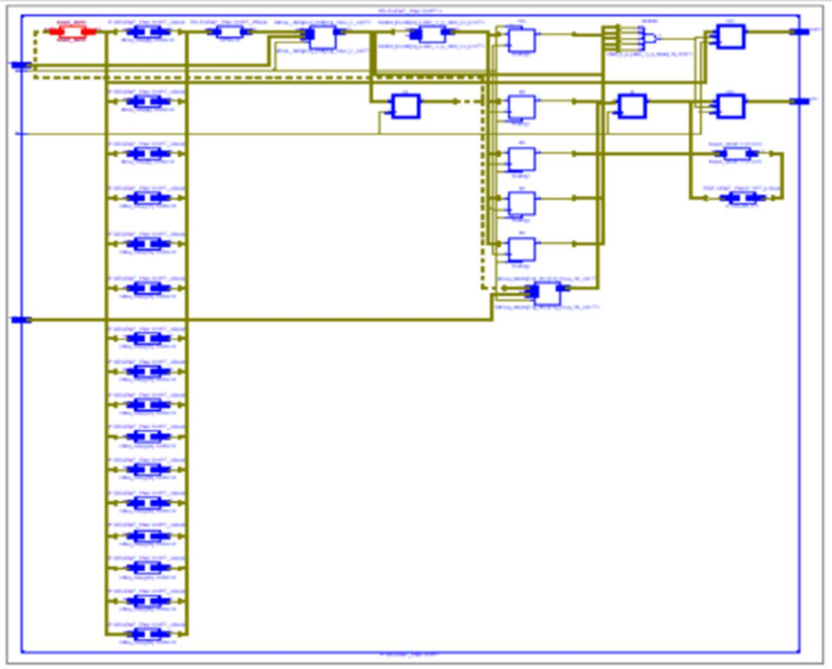


Figure: Encryption Internal Circuit

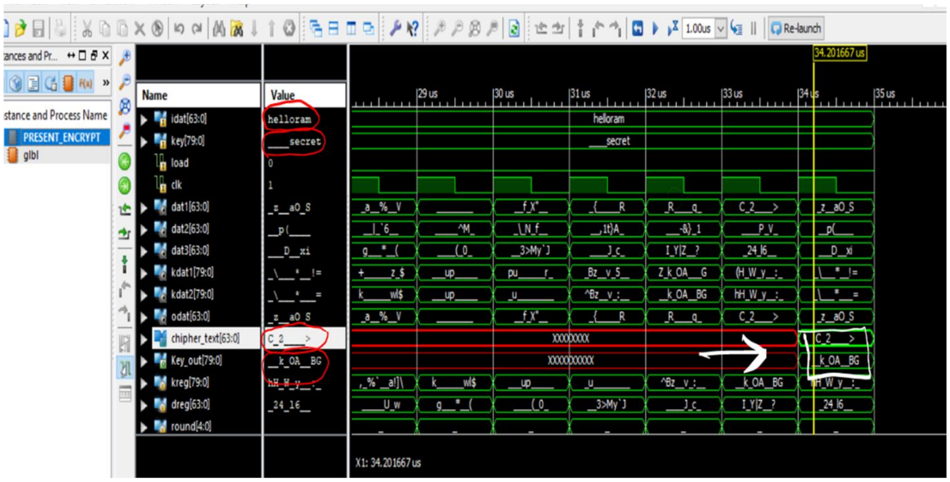
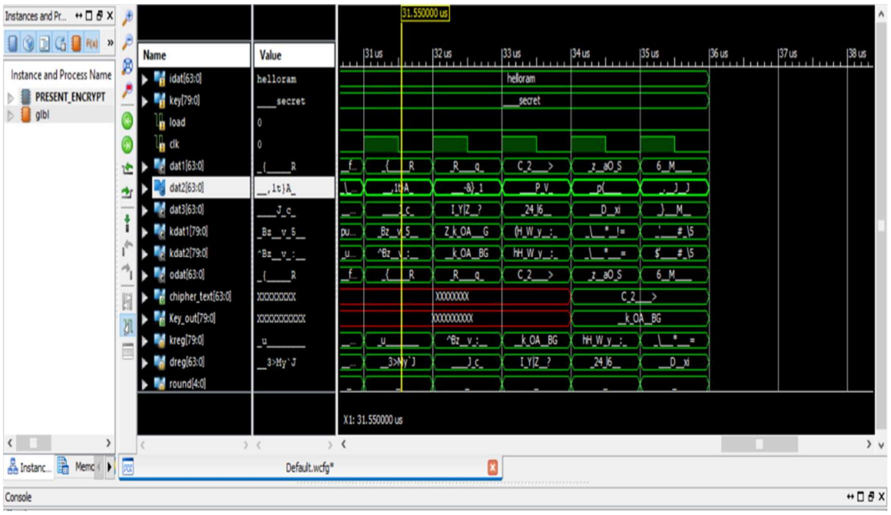
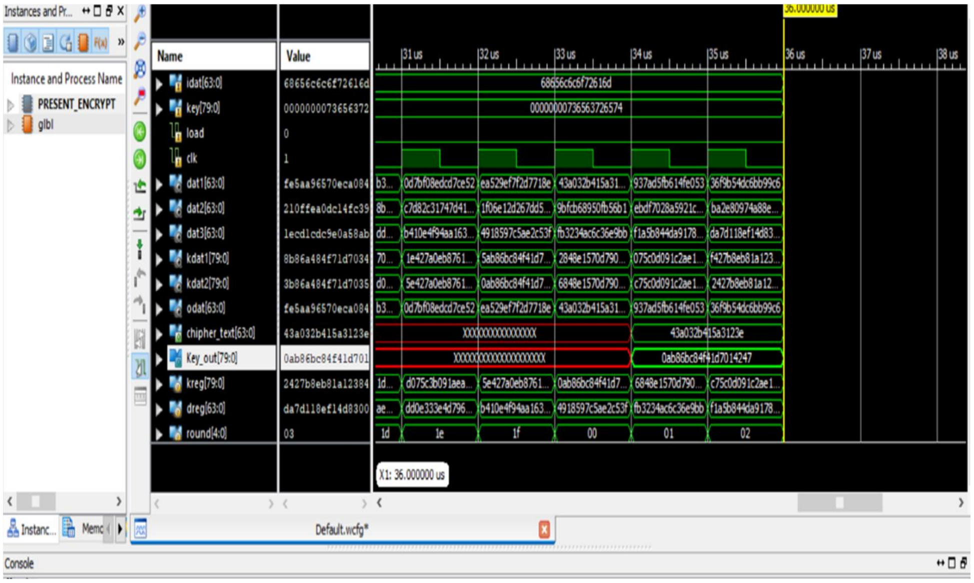


Figure: Encryption Simulation Results



```
Name               Value
idat[63:0]         helloram
key[79:0]          ____secret
load               0
clk                0
dat1[63:0]         _{_____R
dat2[63:0]         __,1t}A_
dat3[63:0]         _____J_c_
kdat1[79:0]        _Bz__v_5__
kdat2[79:0]        ^Bz__v_:__
odat[63:0]         _{_____R
chipher_text[63:0] XXXXXXXX
Key_out[79:0]      XXXXXXXXXX
kreg[79:0]         _u_________
dreg[63:0]         __3>My`J
round[4:0]
```

Figure: ASCII Encryption Output



```
Name               Value
idat[63:0]         68656c6c6f72616d
key[79:0]          0000000073656372
load               0
clk                1
dat1[63:0]         fe5aa96570eca084
dat2[63:0]         210ffea0dc14fc39
dat3[63:0]         1ecd1cdc9e0a58ab
kdat1[79:0]        8b86a484f71d7034
kdat2[79:0]        3b86a484f71d7035
odat[63:0]         fe5aa96570eca084
chipher_text[63:0] 43a032b415a3123e
Key_out[79:0]      0ab86bc84f41d701
kreg[79:0]         2427b8eb81a12384
dreg[63:0]         da7d118ef14d8300
round[4:0]         03
```

Figure: HEXA Encryption Output

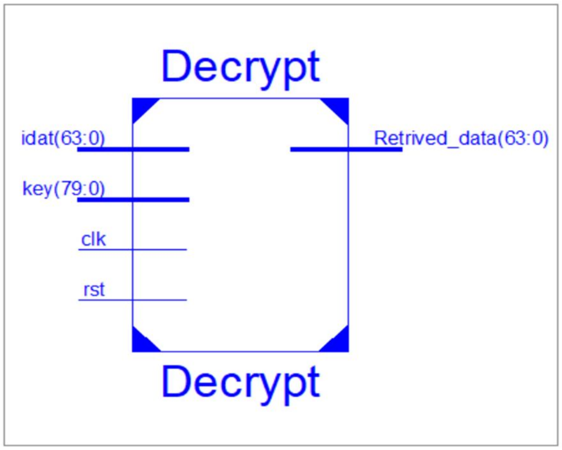


Figure: Decryption

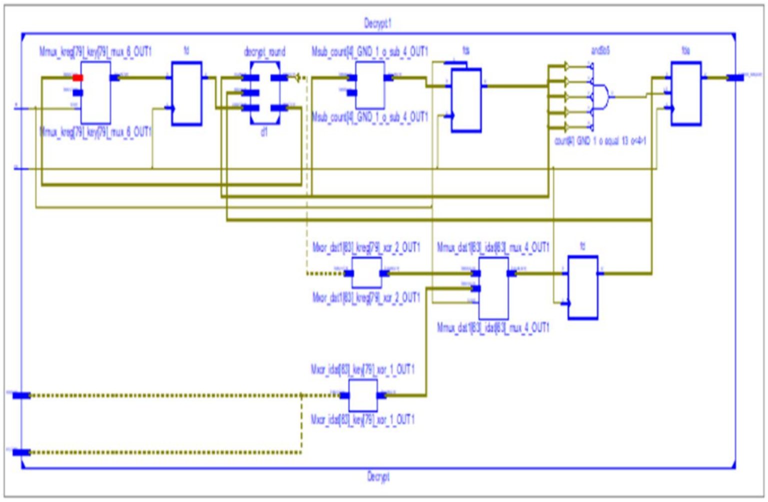
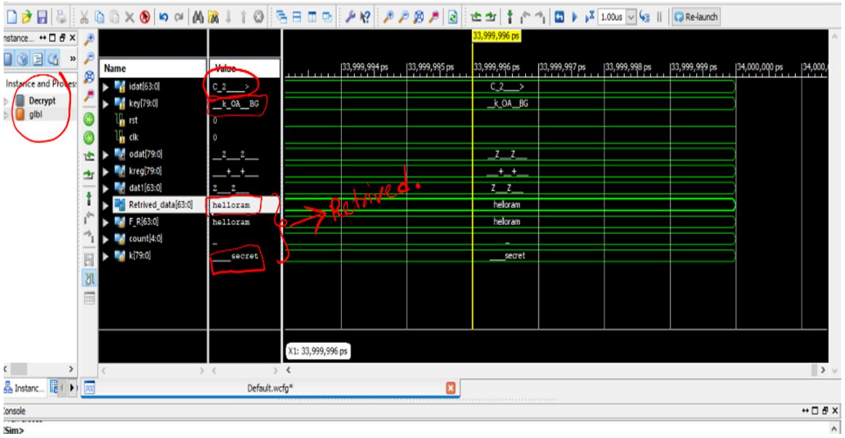

Decrypt1
Mmux_kreg[79]_key[79]_mux_6_OUT1
decrypt_round
Msub_count[4]_GND_1_o_sub_4_OUT1
and6
count[4]_GND_1_o_equal_13_o<4>1
Mxor_dat1[63]_kreg[79]_xor_2_OUT1
Mmux_dat1[63]_idat[63]_mux_4_OUT1
Mxor_idat[63]_key[79]_xor_1_OUT1
Decrypt

Figure: Decrypt Internal Circuit

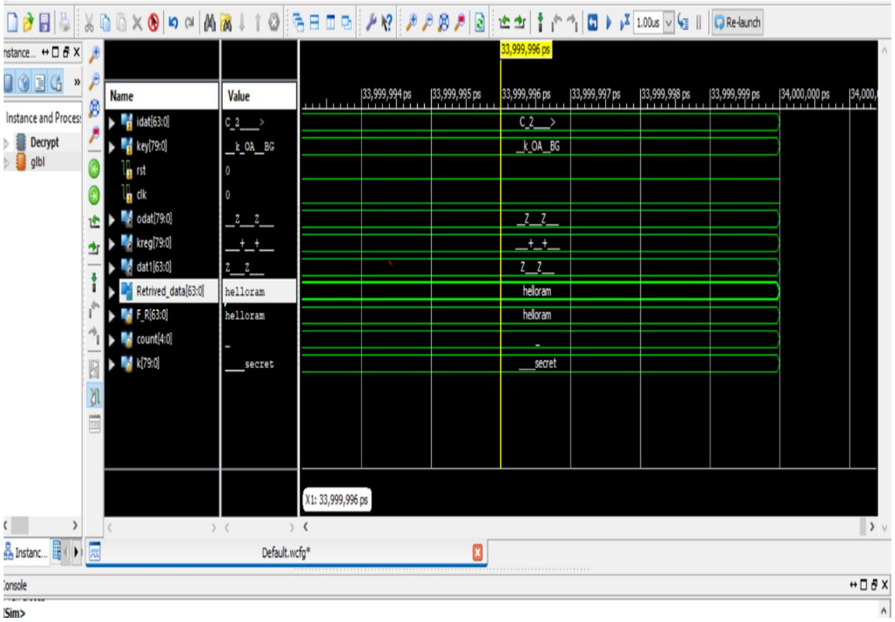


Name	Value
idat[63:0]	C_2____>
key[79:0]	__k_OA__BG
rst	0
clk	0
odat[79:0]	__Z___Z___
kreg[79:0]	___+__+___
dat1[63:0]	Z___Z___
Retrived_data[63:0]	helloram
F_R[63:0]	helloram
count[4:0]	_
k[79:0]	____secret

→ Retrived.

X1: 33,999,996 ps

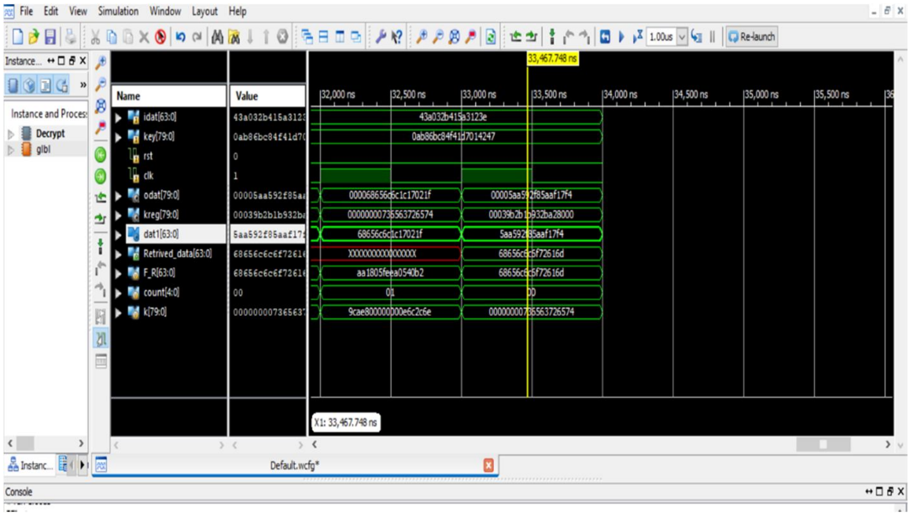
Figure: Decrypt Simulation Results



Name	Value
idat[63:0]	C_2____>
key[79:0]	__k_OA__BG
rst	0
clk	0
odat[79:0]	__Z___Z___
kreg[79:0]	___+__+___
dat1[63:0]	Z___Z___
Retrived_data[63:0]	helloram
F_R[63:0]	helloram
count[4:0]	_
k[79:0]	____secret

X1: 33,999,996 ps

Figure: ASCII Decryption Output



Name	Value
idat[63:0]	43a032b415a3123
key[79:0]	0ab86bc84f41d70
rst	0
clk	1
odat[79:0]	00005aa592f85aa
kreg[79:0]	00039b2b1b932ba
dat1[63:0]	5aa592f85aaf17f
Retrived_data[63:0]	68656c6c6f72616
F_R[63:0]	68656c6c6f72616
count[4:0]	00
k[79:0]	00000000736563

Figure: HEXA Decryption Output

Table - 1: Comparison of Existing method and Proposed method of different parameters

Parameters	Existing Method	Proposed Method
Clk Frequency	**200 Mhz**	**514.64 M hz**
Look Up Table (LUT)	**800**	**526**
Slice Registers	**726**	**105**
Delay (ns)	**5.0**	**1.943**

VIII. CONCLUSION

The recent technological changes, all kinds of devices, from powerful computing devices such as desktop computers, to small computing devices, such as RFID tags are being connected to each other via internet. With these changes, conventional cryptographic algorithms are slowly failing to satisfy the security and performance requirements especially on resource constrained devices. Therefore, the cryptographic community has been working to design efficient algorithms that can be implemented on resource constrained devices without compromising security or performance. PRESENT is one of these algorithms. It is a lightweight block cipher designed in 2007 [6], and since then, it has been analyzed by the cryptographers. While there have been attacks on PRESENT, none of those attacks have been able to break the full 31-rounds of PRESENT. Thus, PRESENT remains as a good example of a light weight algorithm. TRIVIUM can also be considered a good example in this context. It was designed in 2005 as part of the eSTREAM stream cipher project. Like PRESENT, TRIVIUM has also been analyzed since its submission. Because of its block cipher-like design, and the nonlinear update of its state, it is secure against effective attacks on stream ciphers. Although both TRIVIUM and PRESENT seem secure algorithms, their designers strongly encourage further analysis.

REFERENCES

[1] Cryptographic competitions CAESAR submissions, 2019 (last accessed July 2020), Available at https://competitions.cr.yp.to/caesar-submissions.html.

[2] 83 Federal Register 22251 (May14,2018), pp 22251-22252, Announcing Request for Comments on Light weight Cryptography Requirements and Evaluation Criteria; Notice, (last accessed July 2020), Available at https://www.federalregister.gov/documents/2018/05/14/2018-10127/announcing-request-for-comments-on-lightweight-cryptography- requirements-and-evaluation-criteria.

[3] 83 Federal Register 43656 (August 27, 2019), pp 43656-43657, Announcing Request for Nominations for Lightweight Cryptographic Algorithms; Notice, (lastaccessed July 2020), Available at https://www.federalregister.gov/documents/2018/08/27/2018-18433/announcing-request-for-nominations-for-lightweight-cryptographic- algorithms.

[4] S. Bedi and N. R. Pillai, Cube attacks on trivium., IACR Cryptol. ePrint Arch., 2009, p. 15, 2009.

[5] A. Biryukov and L. Perrin, State of the art in lightweight symmetric cryptogra- phy, 2017, Available at https://eprint.iacr.org/2017/511.pdf.

[6] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. Rob- shaw, Y. Seurin, and C. Vikkelsoe, Present: An ultra-light weight block cipher, in International workshop on cryptographic hardware and embedded systems, pp. 450–466, Springer, 2007.

[7] C. De Cannière, Trivium: A stream cipher construction inspired by block cipher design principles, in International Conference on Information Security, pp. 171– 186, Springer, 2006.

[8] C. De Cannière and B. Preneel, Trivium specifications, in eSTREAM, ECRYPT Stream Cipher Project, Citeseer, 2005.

[9] I. Dinur and A. Shamir, Cube attacks on tweakable black box polynomials, in Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 278–299, Springer, 2009.

[10] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer, Ascon v1.2, Sub- mission to NIST, 2019.

[11] M. Dworkin, Recommendation for block ciphers modes of operation methods and techniques, 2001, NIST Special Publication 800-38A.

[12] S. Islam and I. U. Haq, Cube attack on trivium and a5/1 stream ciphers, in 2016 13th International Bhurban Conference on Applied Sciences and Technology (IBCAST), pp. 409–415, IEEE, 2016.

[13] R. Kurzweil, The law of accelerating returns, in Alan Turing: Life and Legacy of a Great Thinker, pp. 381–416, Springer, Berlin, Heidelberg, 2004.

[14] K. McKay, NIST Lightweight Cryptography Standardization: Next Steps, National Institute of Standards and Technology, 2019, NIST Lightweight Cryptography Workshop 2019 Selected Presentations. Available at https://csrc.nist.gov/CSRC/media/Presentations/nist-lightweight-cryptography-standardization-next/images-media/session12-mckay-next-steps.pdf.

[15] K. A. McKay, L. Bassham, M. Sönmez Turan, and N. Mouha, Report on lightweight cryptography, 2017, NIST Internal Report (IR) 8114.