International Journal of Innovative Research in Computerand Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.194 |

||Volume 8, Issue 4, April 2020||

Implementation of AES – 128/192/256 Data Security Encryption Algorithm in STM32 Microcontroller

Inkee Chouhan[1], Dr. Monisha Sharma[2]

[1]M.Tech. Scholar, Dept. of Electronics Communication Engineering, ShriShankaracharya Technical Campus, Bhilai, C.G. India

[2]P.hD, Principal, ShriShankaracharya Technical Campus, Bhilai, C.G. India

ABSTRACT: Advanced Encryption Standard (AES) algorithm is one on the most common and widely symmetric block cipher algorithm used in worldwide. This algorithm has an own particular structure to encrypt and decrypt sensitive data and is applied in hardware and software all over the world. It is extremely difficult to hackers to get the real data when encrypting by AES algorithm. Till date is not any evidence to crake this algorithm. AES has the ability to deal with three different key sizes such as AES 128, 192 and 256 bit and each of this ciphers has 128 bit block size. This paper will provide an overview of AES algorithm and explain several crucial features of this algorithm in details and demonstration some previous researches that have done on it with comparing to other algorithms such as DES, 3DES, Blowfish etc.

The "blue pill" is a STM32F103 based development board. More than that, STM32F103 is a device with Cortex-M3 ARM CPU that runs at 72 MHz, 20 kB of RAM and 64 or 128 kB of flash memory. The microcontroller (MCU) has USB port, two serial ports, 16 bit PWM pins and 12 bit ADC pins. It runs at 3.3V, but some of its pins are 5V tolerant. Programming the board can be simplified using the popular Arduino IDE. But before this, an Arduino-like boot loader must be flashed to the board. This can be done via serial port or using the debug interface of the MCU with ST-Link tool.

In our project, we have saved our data in text file format into SD Card. We have used ATMEGA328 as interface between STM32 and SD Card for Encryption and Decryption time calculation. We are using variable massage and key length for testing of our embedded system project. Final result is also saving into the SD card with Cipher text. AES-128/192/256 are taking different time for encryption of same text message and producing different cipher text.

KEYWORDS: Data, STM32 microcontroller, Encryption, Decryption, AES-128/192/256.

I.INTRODUCTION

Cryptography, the use of codes and ciphers to protect secrets, began thousands of years ago. Until recent decades, it has been the story of what might be called classic cryptography - that is, of methods of encryption that use pen and paper, or perhaps simple mechanical aids. In the early 20th century, the invention of complex mechanical and electromechanical machines, such as the Enigma rotor machine, provided more sophisticated and efficient means of encryption; and the subsequent introduction of electronics and computing has allowed elaborate schemes of still greater complexity, most of which are entirely unsuited to pen and paper. The development of cryptography has been paralleled by the development of cryptanalysis - the "breaking" of codes and ciphers. The discovery and application, early on, of frequency analysis to the reading of encrypted communications has, on occasion, altered the course of history. Thus the Zimmermann Telegram triggered the United States' entry into World War I; and Allied reading of Nazi Germany's ciphers shortened World War II, in some evaluations by as much as two years. Before the modern era, cryptography was concerned solely with message confidentiality (i.e., encryption)—conversion of messages from a comprehensible form into an incomprehensible one and back again at the other end, rendering it unreadable by interceptors or eavesdroppers without secret knowledge (namely the key needed for decryption of that message). Encryption was used to (attempt to) ensure secrecy in communications, such as those of spies, military leaders, and diplomats.

In recent decades, the field has expanded beyond confidentiality concerns to include techniques for message integrity checking, sender/receiver identity authentication, digital signatures and interactive proofs and secure computation, among others. Towards the close of the 20th century, the National Institute for Standards and Testing (NIST) acted on the need for a new encryption algorithm capable of protecting top secret information. NIST is part of the Department of Commerce. It is a non-regulatory agency that, promotes "U.S. innovation and industrial competitiveness by advancing measurement science, standards, and technology in ways that enhance economic security and improve

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.194 |

||Volume 8, Issue 4, April 2020||

quality of life. "Early in the development process, NIST decided to draw on the world's finest cryptographic minds and asked them to submit candidates for the new algorithm because the aging Data Encryption Standard (DES) has many weaknesses. DES has a relatively small 56-bit key which was becoming vulnerable to brute force attacks. In addition, the DES was designed primarily for hardware and is relatively slow when implemented in software. While Triple-DES avoids the problem of a small key size, it is very slow even in hardware; it is unsuitable for limited-resource platforms; and it may be affected by potential security issues connected with the (today comparatively small) block size of 64 bits. In 1997 NIST published a formal call which read in part: It is intended that the AES will specify an unclassified, publicly disclosed encryption algorithm available royalty free worldwide that is capable of protecting sensitive government information well into the next century. The purpose of this notice is to solicit candidate algorithms from the public, academic/research communities, manufacturers, voluntary standards organizations, and Federal, state, and local government organizations. Following the close of the submission period, NIST intends to make all submissions publicly available for review and comment. The entire process spanned five years. Fifteen competing algorithms with colorful names such as Rijndael (the eventual winner), Twofish and Serpent (the runners up) were scrutinized over a three year period. AES is now the industry standard for encryption. The NSA employs it for protecting secret information and industry uses the algorithm for creating commercially available encryption products. File encryption and email encryption are two common applications for AES. File encryption protects the information on your hard disk or thumb drive. With encryption, your data will be secure even if your computer is hacked or your USB drive stolen. Email encryption protects your messages as they journey through the cloud and keeps them from being read by unintended recipients.

II.SYSTEM DESIGN AND IMPLEMENTATION

The more popular and widely adopted symmetric encryption algorithm likely to be encountered nowadays is the Advanced Encryption Standard (AES). It is found at least six times faster than triple DES.

A replacement for DES was needed as its key size was too small. With increasing computing power, it was considered vulnerable against exhaustive key search attack. Triple DES was designed to overcome this drawback but it was found slow.

The features of AES are as follows −

- Symmetric key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- Stronger and faster than Triple-DES
- Provide full specification and design details
- Software implementable in C and Java

Operation of AES

AES is an iterative rather than Feistel cipher. It is based on ‘substitution–permutation network’. It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations).

Interestingly, AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix.

Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key.

The schematic of AES structure is given in the following illustration.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.194 |

||Volume 8, Issue 4, April 2020||

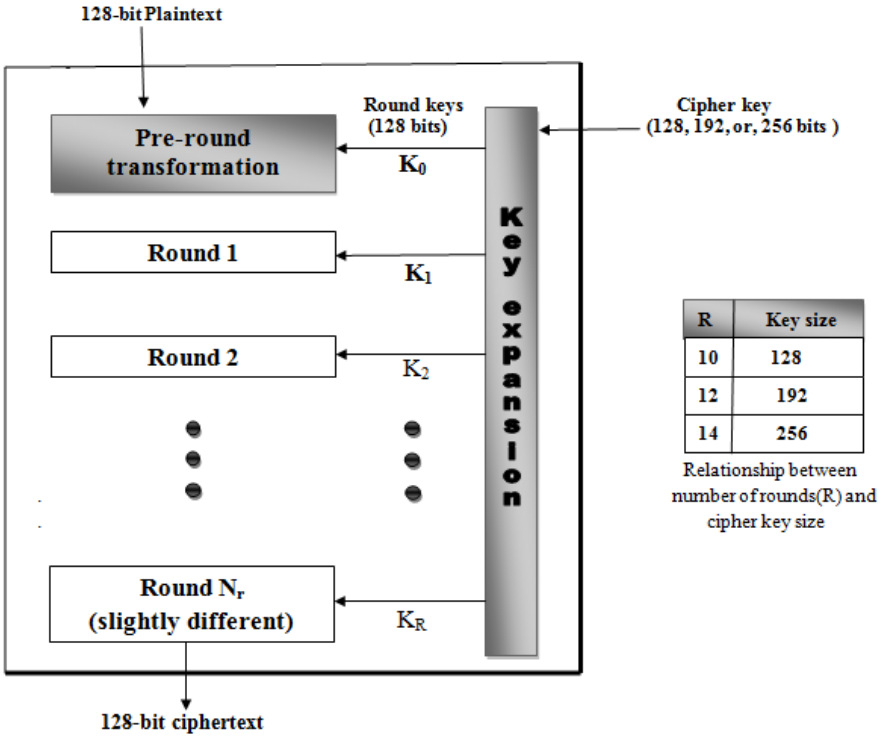

Fig1. Flowchart of AES-128 Encryption System

Encryption Process

Here, we restrict to description of a typical round of AES encryption. Each round comprise of four sub-processes. The first round process is depicted below

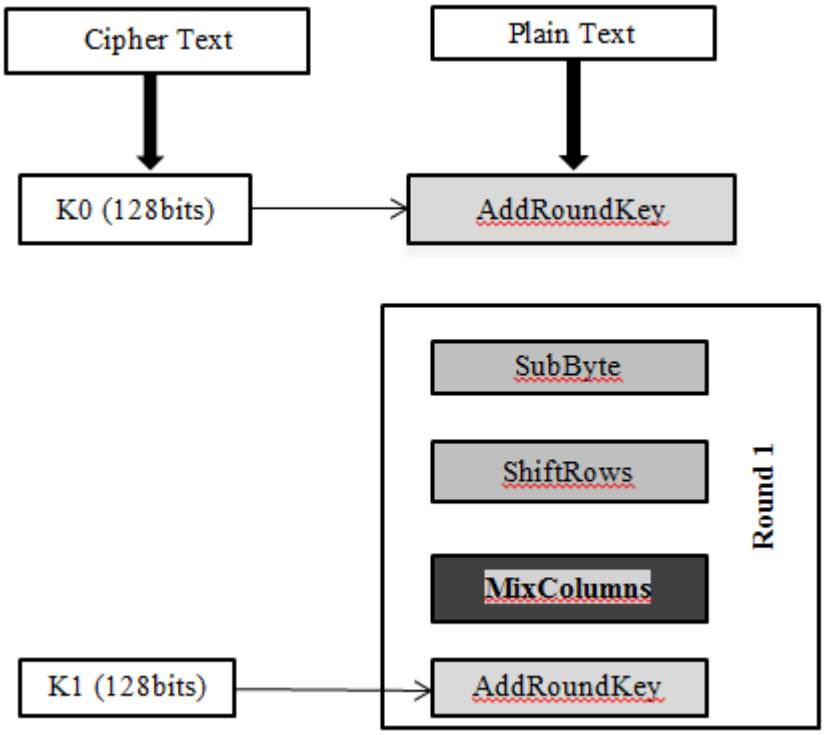

Fig2. Flowchart of Round1 in AES-128

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.194 |

||Volume 8, Issue 4, April 2020||

SubBytes

The 16 input bytes are substituted by looking up a fixed table (S-box) given in design. The result is in a matrix of four rows and four columns.

Shiftrows

Each of the four rows of the matrix is shifted to the left. Any entries that 'fall off' are re-inserted on the right side of row. Shift is carried out as follows −

- First row is not shifted.
- Second row is shifted one (byte) position to the left.
- Third row is shifted two positions to the left.
- Fourth row is shifted three positions to the left.
- The result is a new matrix consisting of the same 16 bytes but shifted with respect to each other.

MixColumns

Each column of four bytes is now transformed using a special mathematical function. This function takes as input the four bytes of one column and outputs four completely new bytes, which replace the original column. The result is another new matrix consisting of 16 new bytes. It should be noted that this step is not performed in the last round.

Addroundkey

The 16 bytes of the matrix are now considered as 128 bits and are XORed to the 128 bits of the round key. If this is the last round then the output is the ciphertext. Otherwise, the resulting 128 bits are interpreted as 16 bytes and we begin another similar round.

Decryption Process

The process of decryption of an AES ciphertext is similar to the encryption process in the reverse order. Each round consists of the four processes conducted in the reverse order −

- Add round key
- Mix columns
- Shift rows
- Byte substitution

Since sub-processes in each round are in reverse manner, unlike for a Feistel Cipher, the encryption and decryption algorithm needs to be separately implemented, although they are very closely related.

AES Analysis

In present day cryptography, AES is widely adopted and supported in both hardware and software. Till date, no practical cryptanalytic attacks against AES have been discovered. Additionally, AES has built-in flexibility of key length, which allows a degree of 'future-proofing' against progress in the ability to perform exhaustive key searches. However, just as for DES, the AES security is assured only if it is correctly implemented and good key management is employed.

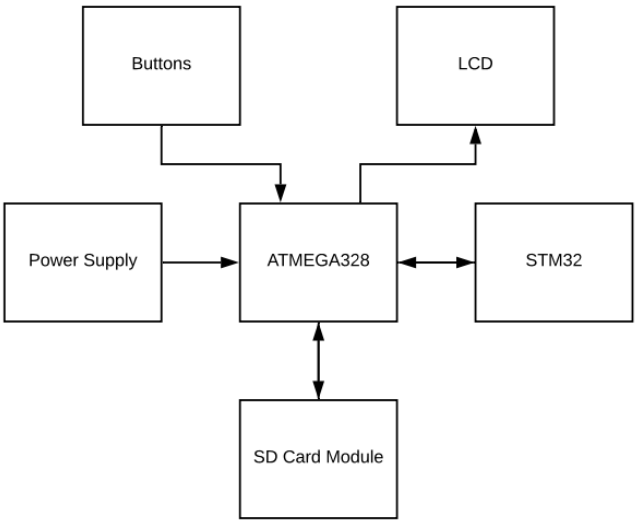

Fig3. Block Diagram of AES Encryption System

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.194 |

||Volume 8, Issue 4, April 2020||

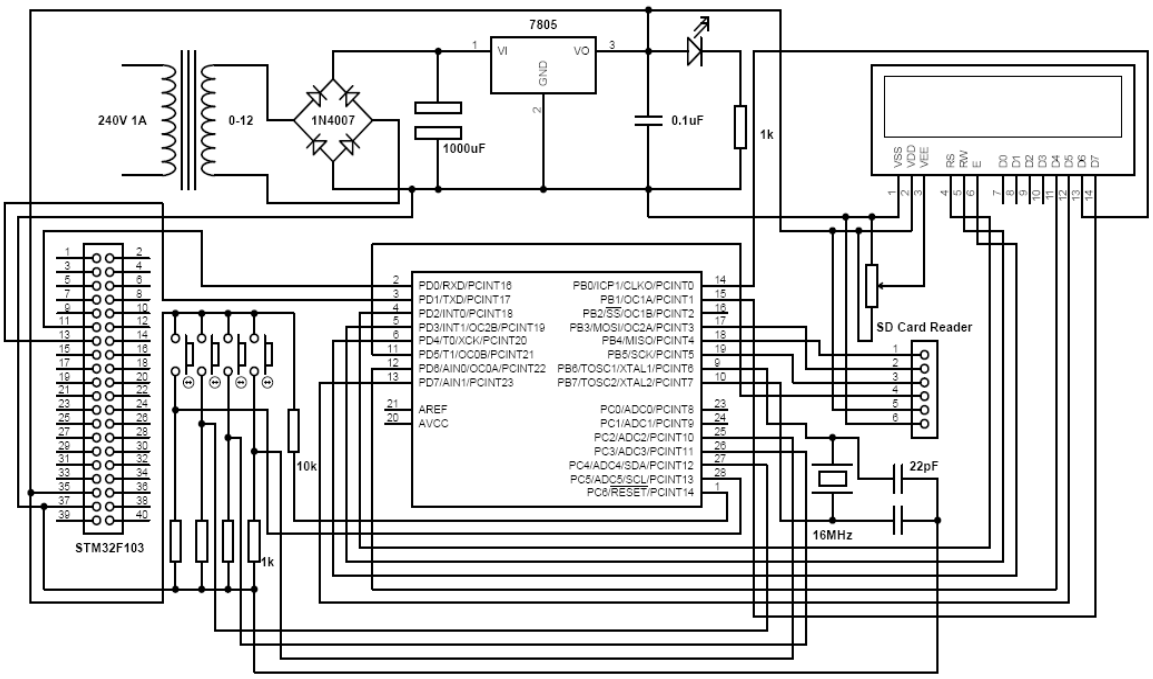

Fig4. Circuit Diagram of AES Encryption System

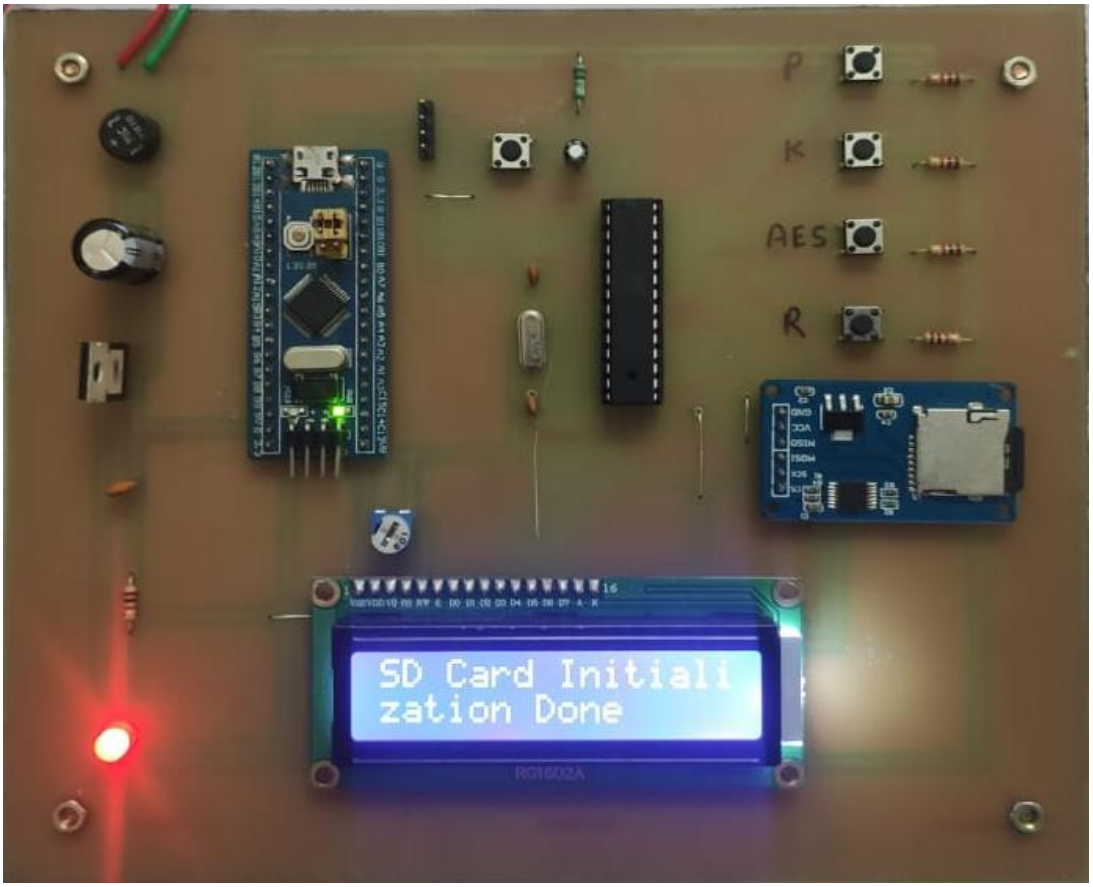

Fig5. Hardware Implementation of AES Encryption System

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.194 |

||Volume 8, Issue 4, April 2020||

III.RESULT AND DISCUSSION

We got below result in SD Card after encryption process:

AES-128bit
Encryption Time:1111 µs
Decryption Time:1798 µs

Plain Text:Once opened, use myFile.prlready on the card, that file would be opened. Name the instance of the opened file "myFile". Once opened, use myFile.prlready on car

Cipher Text:™tw3oóx4ù+^[µQ—¶ì<Âjä‰Œ_¹íÐÎOî l€u³I0WV7¡…'<áà?=ÏÛ
*¯Ÿö**ÙÙ**WîqDó±Ø^¥«º;|+qz„ý*j,'d$žÃÓÐ@gr÷ØÓìa˜ùÕu”KºÂ<¨xåÜŠ§Öª**A9ù¢Û8‰Ð**
ÒdŠ™=¥q•‹ããC.·Ëi

Decrypted Text:Once opened, use myFile.prlready on the card, that file would be opened. Name the instance of the opened file "myFile". Once opened, use myFile.prlready on car

AES-192bit
Encryption Time:1314 µs
Decryption Time:2148 µs
Plain Text:Once opened, use myFile.prlready on the card, that file would be opened. Name the instance of the opened file "myFile". Once opened, use myFile.prlready on car

Cipher
Text:rlîµr¬´(Ü+_Ç□W?ï[”£Ž!U°`ÔQft2hÓÚséBˆº**Ö**ªÊj-‡‡-P³s»Š#™Õ¡6†m)ÍË£÷*ƒ*)ÇìÊÄcŠ-½□=Ùsí>ãèÿ#t¿ñÉË´
ÑhŽÓ&õ‘™ªJåbZó*ƒ*úB…>5‰!8‘ÖUð<!»{¤F=P**U**e-¯þßÍÜ¥S€˜

Decrypted Text:Once opened, use myFile.prlready on the card, that file would be opened. Name the instance of the opened file "myFile". Once opened, use myFile.prlready on car

AES-256bit
Encryption Time:1529µs
Decryption Time:2511µs

Plain Text:Once opened, use myFile.prlready on the card, that file would be opened. Name the instance of the opened file "myFile". Once opened, use myFile.prlready on car

Cipher Text: qf$È¦Èow9éu**ñü**‰Üc›Ã]Ù™Ëg/ç<rï†%¥mUÛêý5` X³=NŽÑ¶Æ**Ï**1îfg-
dÏ&**Œ**qPälWg»óÏ**n**Å*ƒ*&žbÇÐ-8å<Ò.uï5ä…d.è‰üð9öi÷ÎÀÅ**)**$-"v¶Xu7,qëÞHá`ó]ÎÙŒU# ~£

Decrypted Text:Once opened, use myFile.prlready on the card, that file would be opened. Name the instance of the opened file "myFile". Once opened, use myFile.prlready on car

########

IV.CONCLUSION

In our project, we implement AES algorithm using STM32F103. We take plain text into SD card and we have used ATMEGA328 as interface between STM32 and SD Card for Encryption and Decryption time calculation. The project based on to calculating encryption and decryption time with cipher text of AES-128/192/256bit. The final result also saved into SD card. In this project we can see our overall result into a single board with the help of LCD. It had been detected that if we take different types of plain text then the encryption and decryption time for each AES-128/192/256 bit is different and it provide high security. In future work we can use this project where security is main purpose like ATM, Mobile applications, Bank transfer application etc.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.194 |

||Volume 8, Issue 4, April 2020||

V.FUTURE WORK

We have implemented AES – 128/192/256 on STM32 and got excellent result, which is near equal to 1ms conversion rate.We can implement this system in Artix-7 FPGA evaluation board for optimizing the design with very fast conversion rate and power consuming calculation. After synthesization, we can develop this design into a chip for development of many security related product such as ATM, Card machine etc.

REFERENCES

1. 2015, Kanchan, PriyankaAgarwal, Mahesh Vibhute , **"Home Automation Using Android and Bluetooth"**, International Journal of Science and Research (IJSR), 4 (10): 85-89
2. 2014, A. S. Gundale, P. A. Kamble , **"Wireless Data logger Using ZigBee"**, International Journal of Science and Research (IJSR), 3(8):1799-1802.
3. 2012, Ahmed ElShafee, KarimAlaaHamed, **"Design and Implementation of a WiFi Based Home Automation System"**, International Journal of Computer, Electrical, Automation, Control and Information Engineering, 6(8):1074-1080.
4. 2008, Min-kyu Choi, Rosslin John Robles, Chang-hwa Hong, Tai-hoon Kim, **"Wireless Network Security: Vulnerabilities, Threats and Countermeasures"**, International Journal of Multimedia and Ubiquitous Engineering, 3(3):77-86.
5. 2012, N.Sklavos, **"Cryptographic Hardware & Embedded System for Communications"**, Satellite Telecommunications (ESTEL), 2012 IEEE First European Conference.
6. 2016, Afritha. A, Julham, Bakti. V. S., **"PengamananAplikasi Login Berbasis Web Dari Packet Sniffer SSLSTRIP denganJavascript"**, JurnalIlmiahTeknikElektro&Komputer, 2(1): 1-9
7. 2016, FadlurRohman, Mohammad Iqbal, **"ImplementasiIotDalamRancangBangunSistem Monitoring Panel Surya BerbasisArduino"**, Prosiding SNATIF Ke -3 Tahun 2016, 189-196.
8. 2000, Gaj.k and Chodowiec.P, "**Comparison of the hardware performance of the AES candidates using reconfigurable hardware",** AES candidate conference, 40-54
9. 2004, Zhang. X et al, **"HIGH-speed VLSI Architectures for the AES algorithm"**, IEEE transaction on very large scale integration system, 12(9): 957-967
10. 2005, Mangard.S et al, "**Successfully attacking masked AES hardware implementation",** International workshop on cryptographic hardware and embedded system, 157-171
11. 2006, Alho. T et al, "**Desing and implementation of low area and low power AES encryption hardware core"**, 9th EUROMICRO conference on digital system design , 577-583
12. 2009, Rais MH. Et al," **Efficient hardware realization of AES using VIRTEX-5 FPGA"** , International journal of computer science and network security , 9(9):59-63
13. 2007,Huang .c et al**, "Compact FPGA implementation of 32- bit AES algorithm using Block RAM",** TENCON IEEE region 10 conference, 1-4
14. 2010, Ghewari PB. Et al, **"Efficient hardware design and implementation of AES cryptosystem"**, International journal of engg. Science and technology,2(3):213-219
15. 2011,Guo J. et al, **"The LED block cipher"**, International workshop on cryptographic hardware and embedded system, 326-341
16. 2012, Kumar A. et al, **"Effective implementation and avalanche effect of AES",** International journal of security, privacy and trust management, 1(3/4):31-35
17. 2016, Zhang Y. et al, **"A compact 446Gbps/w AES accelerator for mobile SoC and IoT in 40 nm",** IEEE symposium on VLSI circuit, 1-2
18. 2018, S. Neelima et al, "**FPGA based Implementation of AES algorithm using MIX-Column",** Microelectronics, Electromagnetics and Telecommunications, 233-245
19. 2019, Banik.S et al, "**Compact circuits for combined AES Encryption / Decryption" ,** Journal of Cryptographic Engineering , 9(1): 69-83
20. 2007, Poschmann.A et al, "**New Light-Weight Crypto Algorithms for RFID**", IEEE International symposium on circuit and systems, 1843-1846
21. 2012, Bhaskar.R et al, **"An efficient hardware model for RSA Encryption system using Vedic mathematics"**, International Conference on Communication Technology and System Design, 30: 124-128
22. 2013, Iyer NC. et al**, "Implementation of Secure Hash Algorithm-1 using FPGA"**, International Journal of Information and Computation Technology, 3(8):757-764
23. 2014, Al-Haija.QA et al, **"A Tiny RSA Cryptosystem based on Arduino microcontroller useful for small scale network"**, Procedia computer science, 34:639-646
24. 2015 , Acho.L et al, **"An experimental realization of a Chaos-Based Secure communication using Arduino Microcontroller"**, The scientific world journal, 9:1-11
25. 2016, Kanchi S. et al, **"Smart as a Cryptographic processor"**, Proceeding of the SIXTH Internationalconference on computer science, Engg. And Information technology, 1-11
26. 2017, Chiranjeevi C. et al, **"Design end to end Encryption based Biometric system for Security"**, International journal of research, 4(2):838-843
27. 2018, Sangeetha G. et al, **"Prediction of cardiovascular disease using sensor and technique of data mining"**, IRJET, 5(9):1539-1542
28. 2019, Dinu D. et al, **"Triathlon of lightweight Block Ciphers for the Internet of Things"** , Journal of Cryptographic engineering , 9(3):238-302
29. 2012, Sharma M. et al, **"Analysis and comparison between AES and DES Cryptographic Algorithm"**, International Journal of Engg. And Innovative Technology, 2(6):362-365