INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact factor: 4.295
(Volume3, Issue3)
Available online at www.ijariit.com

Implementation of Secured Sensor Nodes Using Cryptographic Algorithms

M. Jothimeena
Dr. Mahalingam College of Engineering and Technology
jothimeena92@gmail.com

R. Sherin Jenny
Dr. Mahalingam College of Engineering and Technology
sjenny98@gmail.com

Abstract: Application of wireless sensor networks (WSN) to commercial, home environments and military purpose are increased in wide range. As the wireless sensor networks continue to grow, they are vulnerable to attacks and hence need for security mechanisms. Identifying suitable cryptography for WSN is important to challenge due to the limitation of energy, computation capacity and storage resources of sensor nodes. For this purpose, Cryptography plays a vital role in securing the data in W SN. In this paper we implemented two symmetric cryptographic algorithms: block cipher AES (Advanced Encryption Standards) and the Lightweight block cipher PRESENT. Our main aim of implementing the symmetric cryptographic algorithms in sensor nodes to provide secure communication which guarantees the confidentiality, integrity and avoids key exchange via an unsecured channel that imposes real threats. The experimental results are carried by implementing these algorithms in Atmega 32 bit microcontroller using simple C codes.

Keywords: Wireless Sensor Networks, Cryptographic Algorithms, AES Algorithm, Present Algorithm, Microcontroller, Simulation Results.

I. INTRODUCTION

Wireless Sensor Network (WSN) is a network of sensors in which Sensor nodes are capable of sensing the data and transmitting it to the base station for analysis [1] [3].The sensors may be installed in a busy traffic area, undersea deployment, in a war field, in a house or an apartment on movable vehicles, in a chemical or biological hazardous field, etc. [3][4]. There are numerous real-time applications of WSN such as Nature calamity forecasting, Greenhouse Monitoring, Traffic control and Vehicle Detection [1].Generally, WSNs are deployed in an inaccessible area; they are more susceptible to various attacks [2]. Attacks are two types: Active attack and Passive attack [3]. Inactive attacks, the attacker listens to and modifies the transmitted data. Denial of Service attack, node replication attack, routing attack etc., are few popular examples of active attacks [6]. In passive attacks, the attacker only eavesdrops the transmitted data [3].Therefore security of WSN becomes almost importance and cryptographic techniques become available solution [5].

The Cryptography is the technique which transforming the original text (plain text) into unreadable text (cipher text) and the cipher text is finally converted to get the original text [1]. It uses an encryption method for sending the original message over the insecure channel. For this, it requires encryption algorithms and a key which means the technique that has been used in encryption of original text. A reverse method of encryption is decryption to get the original text message from the cipher text, for this also a key is used along with decryption algorithms. A key is alphanumeric text or numeric or may be special symbols and the selection of a key for both encryption and decryption is more important [2] [3]. Various cryptography techniques are Symmetric key encryption; Asymmetric key encryption and lightweight encryption are used WSN networks

The asymmetric key encryption algorithm is widely using technology around the world. But it has hardware limitations as like memory and battery, so it is not applied to the sensor network [6] [2]. Therefore, Symmetric key encryption algorithm with low-Energy consumption is used in the sensor networks.

This paper deals with two different algorithms of symmetric key block cipher AES and lightweight block cipher PRESENT algorithms to provide secure communication.Section II deals with related works.Section III describes the brief explanation of AES and PRESENT algorithms.Simulation results are explained in section IV.Finally concluding remarks and future works are made in section V.

II. RELATED WORKS

The authors of [7] explained two most broadly used symmetric key encryption methods i.e. data encryption standard DES and advanced encryption standard AES and have been designed and simulated using MATLAB. The avalanche effect is explained the implementation of DES required 43.3 KB memory and it is very high.

The authors of [8] proposed an energy efficient security protocol first requires data related to the energy consumption of common encryption schemes. The results of an experiment with AES and RC4, two symmetric key algorithms that are commonly suggested or used in WLANs and show that RC4 is more suitable for large packets and AES for small packets.

Hoang Trang [9] proposed an efficient FPGA implementation of the Advanced Encryption Standard algorithm. Advanced Encryption Standard (AES) algorithm implementation is compared with other works to show the efficiency and the design uses an iterative looping approach with a block and a key size of 128 bits, lookup table implementation of S-box. This gives low complexity architecture and easily achieves low latency as well as high throughput. Simulation results, performance results are presented and compared with previously reported designs.

Hui Yue-chao [10] presented Secure RFID System Based on Lightweight Block Cipher Algorithm. He provided some design ideas of a block cipher, and then illustrated a new secure communication model based on the existing RFID security system, which meets the EPC global protocol. At last a lightweight block cipher algorithm is proposed based on the improved DES algorithm and features of the block cipher. For controlling complexity, the encrypted data and key bits are halved in the algorithm

In [11] authors presented work compares performance and resource requirements of symmetric block ciphers on three different embedded microcontroller platforms. The implementation of the algorithms is taken from freely available open source libraries of cryptographic primitives. The evaluated parameters are the generated code-size and the speed of encryption/decryption for the algorithms and microcontroller platforms for secure communication in WSN

Bogdanov. A [12] has described the new block cipher present and his goal has been an ultra-lightweight cipher that offers a level of security commensurate with a 64-bit block size and an 80-bit key. Intriguingly PRESENT has implementation requirements similar to many compact stream ciphers.

III.DESCRIPTION OF AES AND PRESENT ALGORITHM

Description of Advanced Encryption Standards

The AES algorithm is a symmetric block cipher that can encrypt and decrypt information. AES was developed by NIST (National Institute of Standards and Technology) in January 1997.

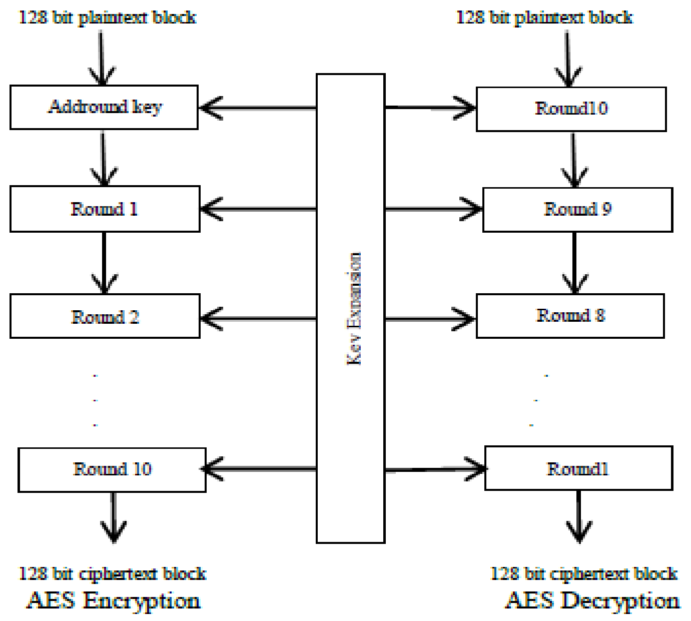


Fig 1: AES block diagram

A. AES encryption

The AES algorithm operates on a 128-bit block of data and executed N_r - 1 round time. A round is the number of iterations of N_r, can be 10**,** 12, or 14 depending on the key length. The key length is 128, 192 or 256 bits in length respectively. AES can be divided into four basic operation blocks of Substitute Bytes, Shiftrows, Mixcolumns, Add Roundy.

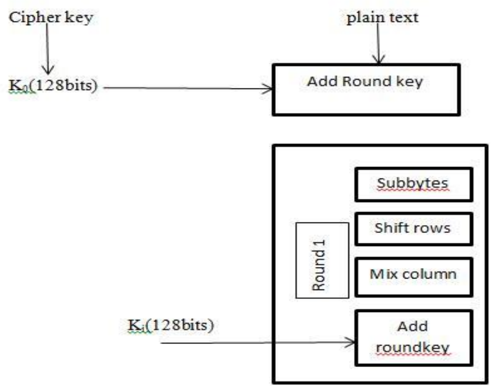


Fig 2: steps in AES rounds

The first and last rounds differ from other rounds in that there is an additional AddRoundKey transformation at the beginning of the first round and no MixCoulmns transformation is performed in the last round. In this paper, we use the key length of 128 bits [9].

SubBytes Transformation:

The SubBytes transformation is a non-linear byte substitution in which each of the state bytes operating independently. The SubBytes transformation is done using pre- calculated substitution table called S-box. The S-box table contains 256 numbers.

ShiftRows Transformation:

In ShiftRows transformation, the rows of the state are rotatory left shifted over different offsets. Row 0 is not shifted; row 1 is shifted one byte to the left; row 2 is shifted two bytes to the left and row 3 is shifted three bytes to the left.

MixColumns Transformation:

In MixColumns is a linear transformation, the columns of the state are considered as polynomials over GF (2^8) and multiplied by modulo $x^4 + 1$ with a fixed polynomial c(x), given by:

$c(x)=\{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$.

AddRoundKey Transformation:

In the AddRoundKey transformation, a Round Key is added to the State - resulted from the operation of the MixColumns transformation - by a simple bitwise XOR operation. The RoundKey of each round is derived from the main key using the KeyExpansion algorithm. The encryption/decryption algorithm needs eleven 128-bit RoundKey, which are denoted RoundKey[0] to RoundKey[10] (the first RoundKey [0] is the main key.

B. AES decryption

Decryption is a reverse operation of encryption which an encrypted cipher-text is converted into original plaintext in reverse order. The round transformation of decryption uses the functions AddRoundKey, InvMixColumns, InvShiftRows, and InvSubBytes successively. AddRoundKey: AddRoundKey is its own inverse function because the XOR function is its own inverse. The round keys have to be selected in reverse order.

AddRoundKey:

AddRoundKey is its own inverse function because the XOR function is its own inverse. The keys in each round have to be selected in reverse order.

InvMixColumns Transformation:

In the InvMixColumns transformation, the polynomials of degree less than 4 over GF(2^8), which coefficients are the elements in the columns of the state, are multiplied modulo ($x^4 + 1$) by a fixed polynomial d(x) = $\{0B\}x^3 + \{0D\}x^2 + \{09\}x + \{0E\}$, where {0B}, {0D}; {09}, {0E} denote hexadecimal values.

InvShiftRows Transformation:

In this transformation, exactly same as shift Shift Rows, only in the opposite direction. The first row is not shifted, while the second, third and fourth rows are shifted right by one, two and three bytes respectively.

InvSubBytes Transformation:

The InvSubBytes Transformation is done using a pre- calculated substitution table called InvS-box. That InvS-box table contains 256 numbers (from 0 to 255) and their corresponding values.

Description of block cipher PRESENT

PRESENT is a lightweight block cipher developed by the Orange Labs, Ruhr University Bochum and the Technical University of Denmark in 2007[12]. PRESENT is used in the area in which low power consumption and high chip efficiency are desired. The block size of PRESENT is 64 bits and supported by the key length of 80 bits and 128 bits consists of 31 rounds. PRESENT is an example of S/P networks [14]. It consists of single 4-bit S-box of not- linear layers mainly intended to design for hardware optimization.

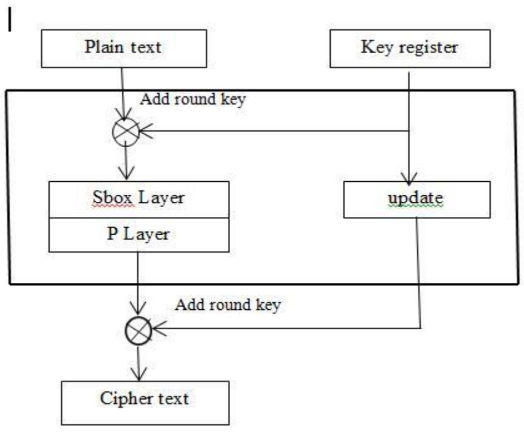


Fig 3: Block diagram of PRESENT

Each round consists of the following 3 steps: 1- AddRoundKey: Key XORed with a cipher. 2- Substitution: Used 4 bits S-box. 3- Permutation: Used P-layer.
AddRoundKey:

Given round key $K^i = \kappa^i_{63} \ldots \kappa^i_0$ for $1 \le i \le 32$ and current state $b_{63} \ldots b_0$, addRoundKey consists of the operation for $0 \le j \le 63$, $b_j \rightarrow b_j \oplus \kappa^i{}_j$.
SBoxlayer:
The S-box used in present is a 4-bit to 4-bit S-box. The action of this box in hexadecimal notation is given in the following table.

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

For sBoxLayer the current state b63 ...b0 is considered as sixteen 4-bit words $w_{15} \ldots w_0$ where $w_i = b_{4*i+3} || b_{4*i+2} || b_{4*i+1} || b_{4*i}$ for $0 \le i \le 15$ and the output nibble S[wi] provides the updated state values in the obvious way.
pLayer :

The bit permutation used in present is given in the following table. Bit i of the state is moved to bit position P(i).

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$P(i)$	0	16	32	48	1	17	33	49	2	18	34	50	3	19	35	51
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$P(i)$	4	20	36	52	5	21	37	53	6	22	38	54	7	23	39	55
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$P(i)$	8	24	40	56	9	25	41	57	10	26	42	58	11	27	43	59
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$P(i)$	12	28	44	60	13	29	45	61	14	30	46	62	15	31	47	63

The key schedule:

The key schedule of PRESENT can take keys of either 80 or 128 bits. However, we focus on the version with 80-bit keys. The user-supplied key is stored in a key register K and represented as $k_{79}k_{78}$ . .k_0. At round i the 64-bit round keyK_i = $\kappa_{63}\kappa_{62}$ . . .κ_0 consists of the 64 leftmost bits of the current contents of register K. Thus at round i we have that: $Ki = \kappa_{63}\kappa_{62} \ldots \kappa 0 = k_{79}k_{78} \ldots k_{16}$.After extracting the round key Ki, the key register $K = k_{79}k_{78} \ldots k_0$ is updated as follows.

1. [k79k78 . . . k1k0] = [k18k17 . . . k20k19]

2. [k79k78k77k76] = *S*[k79k78k77k76]

3. [k19k18k17k16k15] = [k19k18k17k16k15] $\oplus$ round counter.

IV SIMULATION AND RESULT ANALYSIS Matlab platform for cryptography algorithms

In this section, we work with MATLAB and it is a high-level language and a comfortable environment for numerical calculations, visualization and programming. Using this platform we can create models, develop algorithms, and applications.

Simulation of AES algorithms

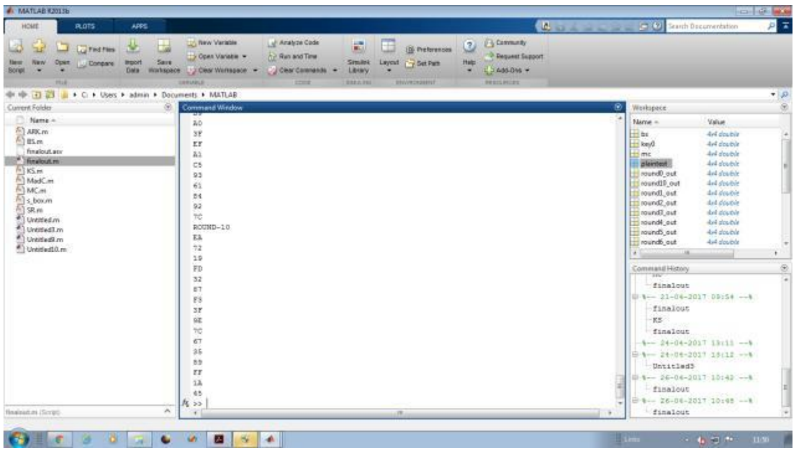
In the first step Initialisation of AES components are done like S-box creation and poly matrices are creation. In the second step round function command such as the shift rows, mix columns, cipher, encryption pseudo codes are written.

```
function [ ark ] = ARK( c,d )

function [ bs ] = BS( round_temp )
function [ out ] = s_box(i,j) function [ sr ] = SR( bs )

function [ roundout ] = KS( roundin )
```

These functions are performed in the MATLAB for AES encryption process.



```
A0
3F
EF
A1
C5
93
61
84
92
7C
ROUND-10
EA
72
19
FD
32
87
F3
3F
9E
7C
67
35
89
FF
1A
65
>>
```

Fig: simulation results of AES encryption

The plaintext of size 128bits are given as input using a symmetric key enable to four-byte oriented operations are performed to get the cipher text. The four-byte oriented operations involve substitution of bytes, shifting rows, mixing of columns, and add round key. This four-byte oriented operation undergoes 10 rounds for simulating 128bit input plaintext and gives the cipher text out so, encryption is done. The number of rounds depends on the size of the key used in this process. All this entire process is done in HP desktop 64-bit operating system with 4GB-RAM, i5 processor, and 2.20 GHz frequency.

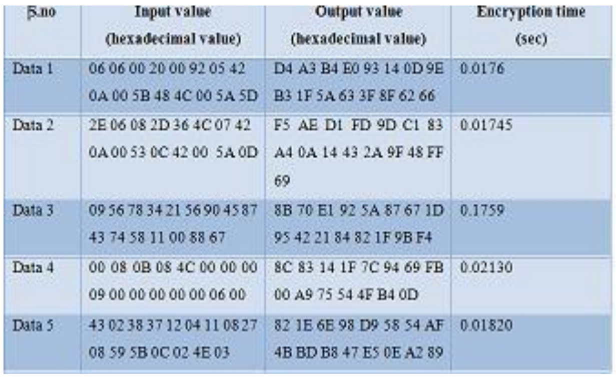
Experimental Results and Analysis for AES

A block of pain text data of 128 bits is given as input to the AES encryption algorithm. A master key of size 128 bit is used by encryption algorithms. Experiments are conducted to find out, one round encryption time and full encryption.

A.One round encryption time:

This time is measured by using *tic toc* command available in the Matlab. This command used at the beginning and end of the one round of encryption code. The measured key set up time is ranging from 0.017s to 0.021s corresponding to same master key values.

Table 1.Time for Encryption for different inputs for one round in AES

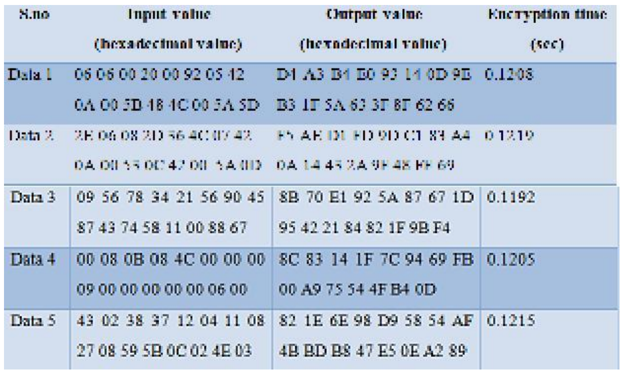


S.no	Input value (hexadecimal value)	Output value (hexadecimal value)	Encryption time (sec)
Data 1	06 06 00 20 00 92 05 42 0A 00 5B 48 4C 00 5A 5D	D4 A3 B4 E0 93 14 0D 9E B3 1F 5A 63 3F 8F 62 66	0.0176
Data 2	2E 06 08 2D 36 4C 07 42 0A 00 53 0C 42 00 5A 0D	F5 AE D1 FD 9D C1 83 A4 0A 14 43 2A 9F 48 FF 69	0.01745
Data 3	09 56 78 34 21 56 90 45 87 43 74 58 11 00 88 67	8B 70 E1 92 5A 87 67 1D 95 42 21 84 82 1F 9B F4	0.1759
Data 4	00 08 0B 08 4C 00 00 00 09 00 00 00 00 00 06 00	8C 83 14 1F 7C 94 69 FB 00 A9 75 54 4F B4 0D	0.02130
Data 5	43 02 38 37 12 04 11 08 27 08 59 5B 0C 02 4E 03	82 1E 6E 98 D9 58 54 AF 4B BD B8 47 E5 0E A2 89	0.01820

B*. *Full encryption time:

This is measured using the method described in the one round encryption time measurement. The tic toc command used in the beginning and the end of the main program.

Table 2: Time of Encryption for different inputs for all round in AES



S.no	Input value (hexadecimal value)	Output value (hexadecimal value)	Encryption time (sec)
Data 1	06 06 00 20 00 92 05 42 0A 00 5B 48 4C 00 5A 5D	D4 A3 B4 E0 93 14 0D 9E B3 1F 5A 63 3F 8F 62 66	0.1208
Data 2	2E 06 08 2D 36 4C 07 42 0A 00 53 0C 42 00 5A 0D	F5 AE D1 FD 9D C1 83 A4 0A 14 43 2A 9F 48 FF 69	0.1219
Data 3	09 56 78 34 21 56 90 45 87 43 74 58 11 00 88 67	8B 70 E1 92 5A 87 67 1D 95 42 21 84 82 1F 9B F4	0.1192
Data 4	00 08 0B 08 4C 00 00 00 09 00 00 00 00 00 06 00	8C 83 14 1F 7C 94 69 FB 00 A9 75 54 4F B4 0D	0.1205
Data 5	43 02 38 37 12 04 11 08 27 08 59 5B 0C 02 4E 03	82 1E 6E 98 D9 58 54 AF 4B BD B8 47 E5 0E A2 89	0.1215

Simulation of PRESENT algorithm

In the first step Initialisation of PRESENT components are done like S-box creation and p-layer box creation.In the second step round function updates, pseudo codes and key schedule are written.

function [ark] = ARK(a,b) function [out] = s_box(e,d)

function [round _n]= BS(round_temp) function [round_ out]=update(BS)

These functions are performed in the MATLAB for PRESENT encryption process as similar to AES

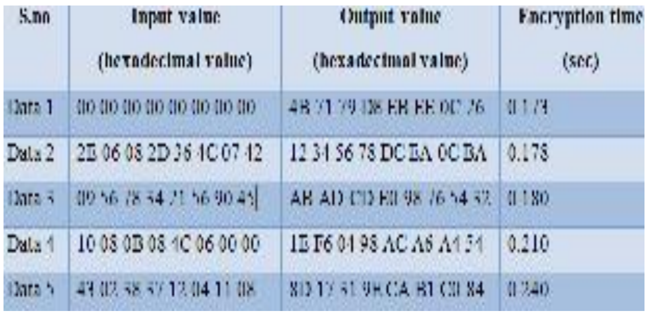
Experimental Results and Analysis for PRESENT

A block of pain text data of 64 bits is given as input to the PRESENT encryption algorithm. A master key of size 80 bit is used by encryption algorithms. Experiments are conducted to find time encryption.

Full encryption time:

This is measured using the method described in the AES algorithm encryption time measurement. The tic toc command used in the beginning and the end of the main program.

Table 2: Time for Encryption for different inputs for all round in PRESENT



S.no	Input value (hexadecimal value)	Output value (hexadecimal value)	Encryption time (sec)
Data 1	00 00 00 00 00 00 00 00	4B 71 79 D8 EB EE 0C 76	0.173
Data 2	2B 06 08 2D 36 4C 07 42	12 34 56 78 DC BA 0C BA	0.178
Data 3	09 56 78 34 21 56 90 45	AB AD CD EF 98 76 54 32	0.180
Data 4	10 08 0B 08 4C 06 00 00	1B F6 04 98 AC A6 A4 51	0.210
Data 5	48 02 58 57 12 04 11 08	8D 12 51 9B CA B1 C0 84	0.240

Implementation of algorithms on microcontroller

This section provides the results on implementation of both algorithms in AVR atmega 32 microcontrollers by using Arduino IDE followed by pre-implementation done by Proteus tool. Due to its easy usage, its low paper consumption and the freely available developments and the freely available development tool, the AVR microcontroller family is widely used in many areas of embedded systems.Typical application areas involving security for WSN and smart cards**.**

Implementation of AES algorithms

In our work first we intend to implement AES-128 algorithms on the ATmega32 8 bit microcontroller integrated chip using simple C codes in Arduino IDE and our simulation is carried out by using Proteus tool to test and verify the encryption. The results are monitored using a virtual terminal.

A block of pain text data of 128 bits is given as input to the AES encryption algorithm and a master key of size 128 bit is used by encryption algorithms. Coding is written using embedded c language in Arduino ide. The hex file function is called for Proteus implementation of AES algorithms.

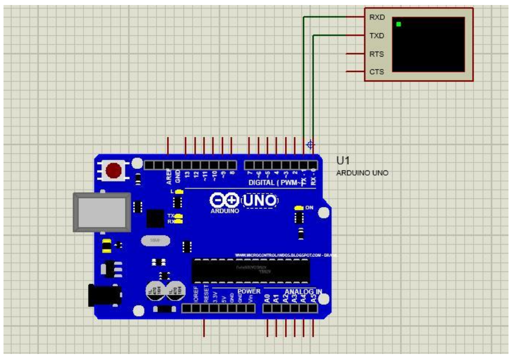
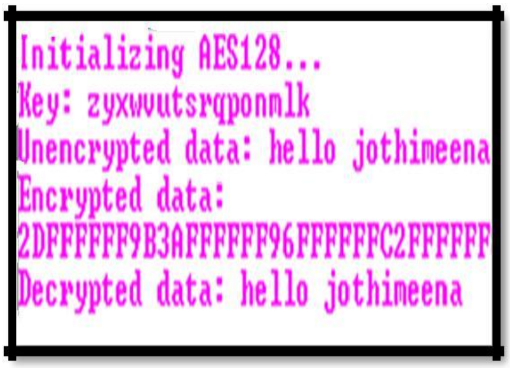


Fig5: Proteus layout of atmega 32



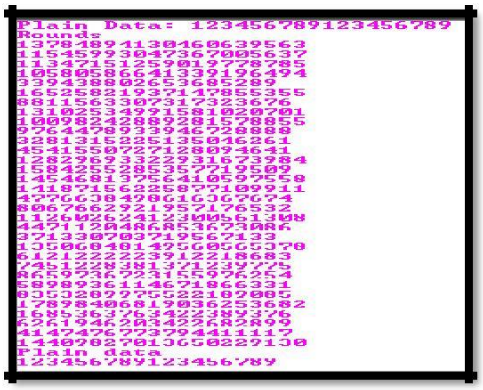
```
Initializing AES128...
Key: zyxwvutsrqponmlk
Unencrypted data: hello jothimeena
Encrypted data:
2DFFFFFF9B3AFFFFFF96FFFFFFC2FFFFFF
Decrypted data: hello jothimeena
```

Fig6: Encryption Result of AES algorithm

The input is given as string value and corresponded encrypted values are obtained in hexadecimal values.

Implementation of PRESENT algorithms

As done in AES algorithm implementation the same procedure is repeated for PRESENT algorithms. A block of pain text data of 64 bits is given as input to the PRESENT encryption algorithm and a master key of size 80 bit is used by encryption algorithms coding are written using embedded c language in Arduino ide.The input is given as long integer value and corresponded encrypted values are obtained in long integer value.

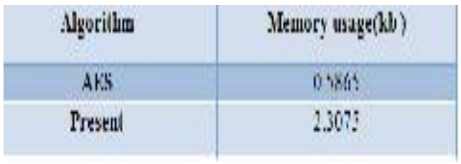


```
Plain Data: 123456789123456789
Rounds
13784891130160639563
11545993047367005637
11347151259019778785
10580586641339196494
3394388026536852089
16525821937147855355
8811563307317323676
13102534997158102070 1
10098242889281578855
9764478933946728888
3281315225135016261
4541550727128094641
12829693322931673984
1584255285357719509
14546813756410597558
14118715622587710991 1
4776630490616367674
8067662921957176532
11260262412300561308
4471120486853673086
3713307037195671 33
13506040149560565370
6121222222391221868 3
7451228381371239775
8659736723155976254
5898936114671866331
03532097975522107005
17898406819036253682
1685363763422389376
6261946203422682899
4147476773794411117
14407027013650229130
Plain data
123456789123456789
```

Fig 7: Encryption Result of PRESENT algorithm

And the memory required for both algorithms are monitored and tabulated.

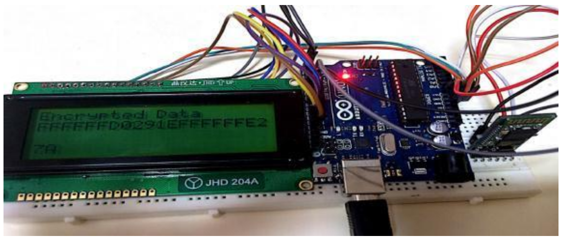
Table 4: memory required for both algorithms



Algorithm	Memory usage(kb)
AES	0.5865
Present	2.3073

Theses Encryption technique can be implemented on microcontrollers in Sensor nodes to guard the information and prevents communication from unauthorized access of information thereby achieving confidentiality.

HARDWARE IMPLEMENTATION



Hardware implementation of AES

The above figure shows the hardware implementation of AES algorithm on the 8bit microcontroller. The results of encryption data are displayed using 16×4 LCD display

CONCLUSION

The above work provides a secure communication in wireless sensor nodes using cryptographic algorithms. It describes different types of symmetric cryptographic algorithms and also explains the working steps of AES and PRESENT algorithms. Furthermore, the encryption and decryption for various data using AES and PRESENT algorithms are simulated through MATLAB. The time required for AES and PRESENT algorithms for encryption and codes are measured. Also, this work implements the AES and PRESENT algorithms on an 8-bit microcontrollers to provide fast and secure communication in sensor nodes. AES algorithm is compared with a PRESENT algorithm based on the requirements of memory size to encrypt a given data using Arudino IDE tool.Finally, hardware implementation of AES algorithm is done.

REFERENCES

[1] Shivangi Goyal, “A Survey on the Application of Cryptography”, International Journal of Science and Technology. Volume 1 no.3March, 2012

[2] M. Guru Vimal Kumar and U.S. Ragupathy, “A Survey on Current Key Issues and Status in Cryptography”, IEEE WiSPNET conference 2016

[3] Haythem Hayouni1, Mohamed Hamdi1, Tai-Hoon Kim2, “A Survey on Encryption Schemes I Wireless Sensor Networks”, 7th International Conference on Advanced Software Engineering & It Applications on IEEE,2014

[4] Jamal N. Al-Karaki, “Routing Techniques in Wireless Sensor Networks: A Survey”, information Security and Assurance, 2008. ISA 2008. International Conference on IEEE Wireless Communications, December 2004.

[5] G.R. Sakthidharan & S. Chitra, “Survey on Wireless Sensor Network: An Application Perspective” International Conference on Computer Communication and Informatics on IEEE, Jan, 2012

[6] Zhang, Xeuying, Howard M.Heys, Cheng Li, “Energy efficenciency of symmetric key cryptographic algorithms in wireless sensor networks,”25th Biennial Symposiumon, pp.168-172, IEEE,2010

[7] Akash Kumar Mandal, Mrs. Archana Tiwari, (2012) “Performance Evaluation of Cryptographic Algorithms: DES and AES”, Students Conference on Electrical, Electronics and Computer Science 978-1- 4673-1515-9/12/$31.00 ©2012 IEEE

[8] P. Prasithsangaree and P. Krishnamurthy, “Analysis of Energy Consumption of RC4 and AES Algorithms in Wireless LANs,” The IEEE International Conference on Information and Automation on IEEE GLOBECOM, 2003

[9] Hoang, Trang. "An efficient FPGAimplementation ion of the Advanced Encryption Standard algorithm. “Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF) RIVF International Conference on. IEEE, 2012

[10] Hui Yue-Chao, Wang Yi-ming,“ Secure RFID system based on Lightweight Block cipher algorithms of optimised S-box,”proceeding of International Conference on RFID technology and application ,Guangz, IEEE, june,2010.

[11] Bogdanov1 A L.R. Knudsen2, G. Leander1, C. Paar1, A. Poschmann1, M.J.B. Robshaw3, Y. Seurin3, and C. Vikkelsoe2, “PRESENT: An Ultra-Lightweight Block Cipher”, LNCS @ SPRINGER-VERLAG BERLIN HEIDELBERG 2007

[12] William Stallings, “Cryptography and Network Security principles and practices “, fifth Edition, Pearson Education, 2003.

[13] Woo Kwon Koo, Hwaseong Lee, Yong Ho Kim, Dong Hoon Lee, Implementation and Analysis of New Lightweight Cryptographic Algorithm Suitable for Wireless Sensor Networks,” International Conference on Information Security and Assurance, IEEE, 2008.

[14] Haohao Liao and Howard M. Heys, “An Integrated Hardware Platform for Four Different Lightweight Block ciphers”, Proceeding of the IEEE 28TH Canadian Conference on Electrical and Computer Engineering Halifax, Canada, IEEE 2015.