©2010 International Journal of Computer Applications (0975 - 8887)
Volume 1 – No. 21

144-Bit Encryption/Decryption Technique

Puvvula Bhargava
B.Tech
SASTRA University
Thanjavur,Tamilnadu,India

Kuppam Hari Kishan
B.Tech
SASTRA University
Thanjavur,Tamilnadu,India

ABSTRACT
In this paper, we introduce a new 144-bit cryptographic algorithm .This is done using a 16-segment electronic display and arranging the bits in a rectangular matrix and then converting it into nearest square matrix with the addition of dummy bits and rearranging the individual bits by a newly proposed mathematical approach.

General Terms
Cryptography, Encryption,Security,16-segment LED,Hexa Decimal Code

Keywords
Assignment mode, pattern mode, base code, element code.

1. INTRODUCTION
The **bit code** is generated for each corresponding alphabets/numerals/special symbols in the '**PLAIN TEXT'** depending on the LED's which glow, when they are represented using the 16- Segment displays.

The bit is set as '**1**' for the glowing LED and as '**0**' for the LED which is not glowing. The bit code thus generated will have 16 bits corresponding. The bit code generated is represented in the equivalent '**HEXA DECIMAL'** code.

The **HEX CODE** generated is dependent on the assignment of alphabets to the LED's (the alphabets **a-p** are assigned to the 16 LED's in the 16-segmented display). The assignment of alphabets to the 16 LED's can be done in sixteen factorial (16!) ways. Thus the hex code generated for the same alphabet/numeral/special symbol will change with the change in assignment of alphabets to the LED's.

The bit code is generated only for the upper case alphabets; the bit code for lower case alphabets is generated by just complementing the bit code generated for its corresponding upper case. So the actual bit code is generated only for 26 upper case alphabets(A-Z) ,10 numerals(0-9) and 5 special symbols(@,$,&,_,~).

For the generation of hex code for each element, any one possible way of assignment of alphabets to LED's is chosen from the available **16! Ways** and corresponding to the arrangement the hex code is generated for that particular element .Similarly the hex code is generated for all the elements by choosing different ways of assignment of alphabets to LED's for each and every element.

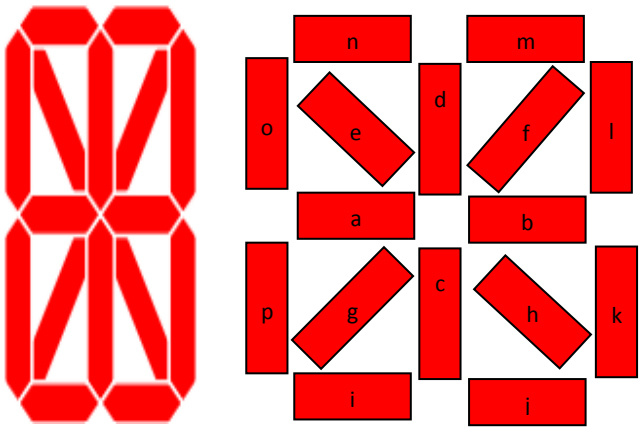
The 16 bits generated for 41 elements thus comes from the 41 different ways of assignment of alphabets to the LED's in the segmented displays. This 16 bit code is represented in the of **4X4 matrix**. The pattern in which the bits are arranged in the matrix is chosen at random.

The 16 bit code which is generated for each element corresponding to the assignment of alphabets to the LED's is called '**SOURCE CODE'**. Another 16 bits are prefixed to the 'SOURCE CODE' which is called the **'BASE CODE'**.

The upper 8 bits of the BASE CODE gives the information about the way in which the assignment of alphabets to the LED's in the segmented display is done and the lower 8 bits give the information about the pattern in which the bits are arranged inside the 4X4 matrix.

Hence there are 32 bits in total which represent each element. The 32 bits all together are called as **ELEMENT CODE**.

2. SOURCE CODE (16-BIT)
2.1 THE 16-SEGMENT DISPLAY



In the 16- segment display there are in all 16 LED's. Using a 16 – segment display we can show both **alphabets (A-Z)** and **umerals (0-9),** hence called **ALPHA – NUMERIC DISPLAY.** The **HEX CODE** generated is dependent on the assignment of alphabets to the LED's (the alphabets **a-p** are assigned to the 16 LED's in the 16-segmented display). The assignment of alphabets (**a-p**) to the 16 LED's can be done in sixteen factorial (16!) ways. Thus the hex code

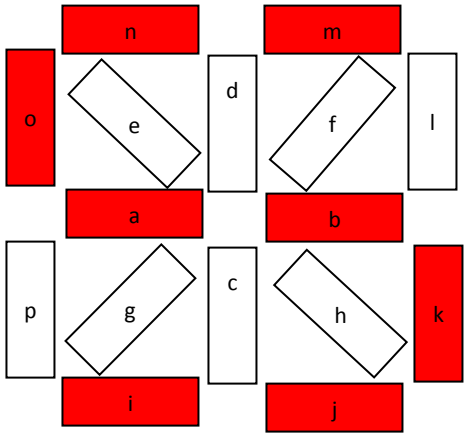
generated for the same alphabet/numeral/special symbol will change with the change in assignment of alphabets to the LED's.

2.2 THE SOURCE CODE GENERATION

Bit code is generated by setting '1' to the alphabets assigned to the glowing LED's and as '0' to the alphabets assigned to the LED's which are not glowing.

Thus for all the 16 alphabets (a-p),assigned to the 16 LED's there will be a value of either '0' or '1' which is set according to the state of the LED's(ON/OFF), when the element is displayed on the segmented display.

Assuming that we have chosen a letter 'S' and according to the assignment of alphabets to the LED's , the equivalent 16 bit binary code is generated . The 16 bit binary code for's' (lower case) is obtained by complementing the bit code generated for 'S' (upper case).



ele	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
S	1	1	0	0	0	0	0	0	1	1	1	0	1	1	1	0
s	0	0	1	1	1	1	1	1	0	0	0	1	0	0	0	1

The bit code for the element 'S' is **1100 0000 1110 1110**

The bit code for the element's' is **0011 1111 0001 0001**

The bit code generated for an element is known as **SOURCE CODE**

The equivalent hexadecimal value for 'S' will be **C0EE .**

3. THE BASE CODE GENERATION

The 16 bits that are prefixed to the 'SOURCE CODE' is called the **'BASE CODE'**.

BASE CODE = ASSIGNMENT MODE + PATTERN MODE

ASSIGNMENT MODE indicates the mode in which the assignment has been done i.e., the information about the way in which the assignment of alphabets to the LED's in the segmented display is done. For each way of assignment of alphabets to the LED's a MODE number is given to that assignment. The mode number is given randomly. But care is taken that no two assignments have the same mode number.

The lower 8 bits give the **PATTERN MODE** i.e., the information about the pattern in which the bits are arranged inside the 4X4 matrix.

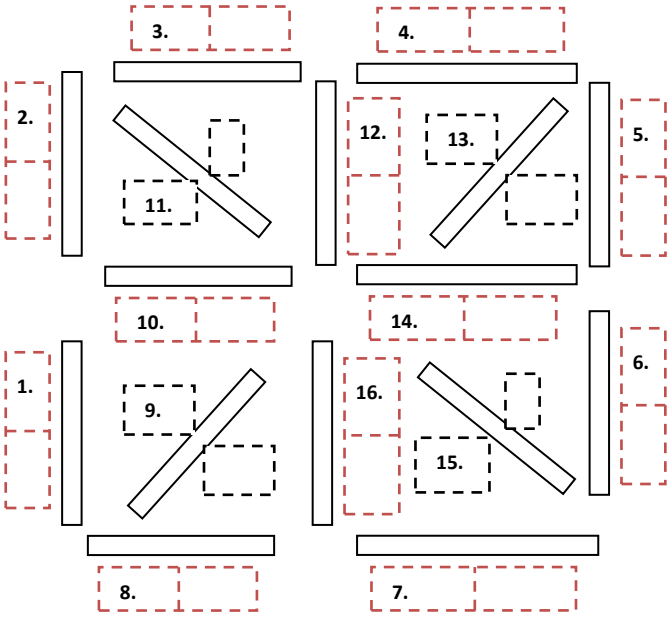
The **PATTERN MODE** indicates the way in which the 16 bits of the source code are arranged in a 4X4 matrix.

3.1. ASSIGNMENT MODE

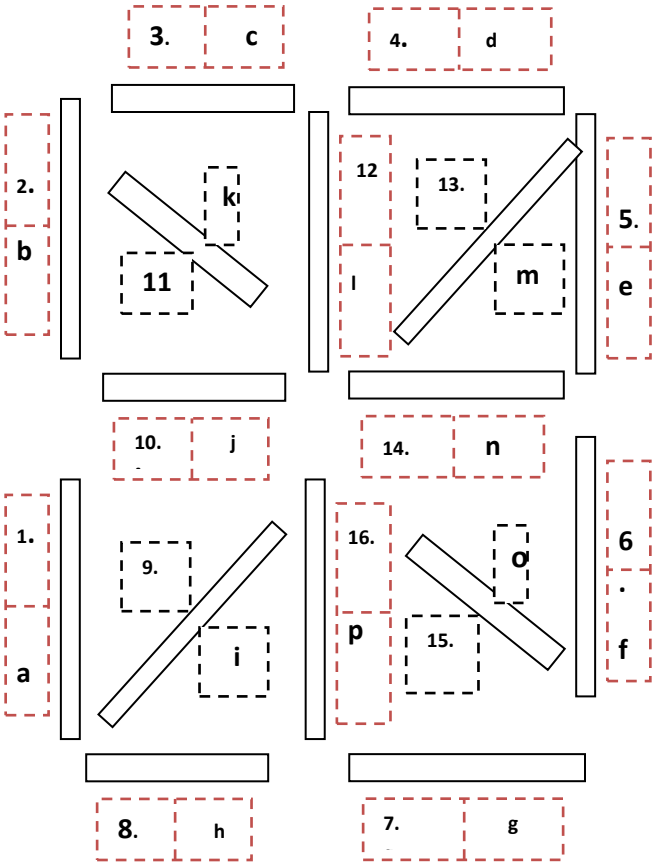
Let us consider a 16 segment display, as shown below. Let the 16 LED's be numbered from 1 to 16 as shown. The numbering to the LED's is permanent .The blank boxes provided are for the assignment alphabets (a-p). As discussed above, the assignment of alphabets (a-p) to the 16 LED's (1-16) changes for every element for which the code is being generated.

Each assignment is given a unique number called the **ASSIGNMENT MODE**. No two elements will have the same **ASSIGNMENT MODE**

Display with no alphabets Assigned



Now let us generate the code for the element 'A'. The assignment of alphabets to the 16 LED's (1-16), for the generation of code for the element 'A' is randomly done. Let the **assignment mode be '0A'** (in hex).



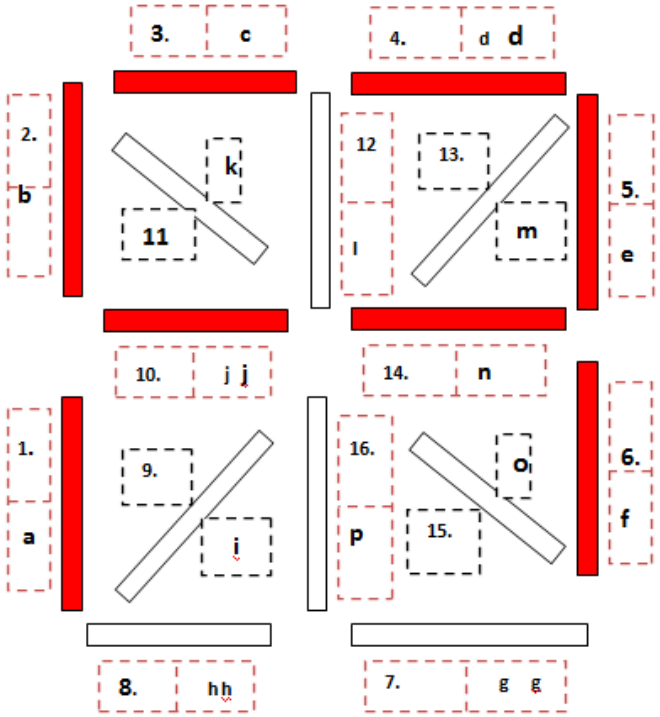
AM: Assignment mode; **LN**: Led number; **AA**: Assigned alphabet

AM	LN	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0A	AA	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p

3.2 SOURCE CODE WITH THE GIVEN ASSIGNMENT MODE

For the element ‘A’ to be displayed on a 16 segment display, the LED’s 1,2,3,4,5,6,10,14 should glow.

The LED’s 1,2,3,4,5,6,10,14 are assigned with alphabets a, b ,c ,d ,e ,f ,j , n respectively in assignment mode ‘OA’. Hence the bit code for element ‘A’ is generated by setting a value of ‘1’ to the alphabets assigned to the glowing LED’s and a value of ‘0’ to the others. The bit code for element ‘a’ is generated by complementing the bit code generated for ‘A’.



AM	ele	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
0A	A	1	1	1	1	1	1	0	0	0	1	0	0	0	1	0	0
	a	0	0	0	0	0	0	1	1	1	0	1	1	1	0	1	1

AM: Assignment mode; **Ele: Element**

The 16 bit code for element ‘A’ is 1111110001000100

The equivalent hex code is FC44.

3.3 PATTERN MODE

Now these 16 bits can be arranged in a 4X4 matrix in 16! Ways. The mode in which the 16 bits are arranged in the matrix is called PATTERN Mode. Each cell in the matrix is given a roman number from **i-xii.** The matrix is called **SOURCE MATRIX.** The matrix is shown below.

Corresponding to the bit which is filled in each cell of the matrix, the matrix is filled with those corresponding alphabets. Let the above bit code generated for element ‘A’ be filled in the given matrix.

I.	II.	III.	IV.
V.	VI.	VII.	VIII.
IX.	X.	XI.	XII.
XIII.	XIV.	XV.	XVI.

If the bit corresponding to alphabet 'a' is to be filled in the **iii** cell of the matrix, then alphabet 'a' is shown in the **iii** cell. The completely filled matrix is shown below. This matrix actually contains only the values 1 or 0, but for easy reference it is represented in terms of alphabets corresponding to the bits. Let **the pattern mode be '0C'.**

I. 1(e)	II. 0(i)	III. 1(a)	IV. 0(h)
V. 0(p)	VI. 1(c)	VII. 1(n)	VIII. 1(d)
IX. 0(0)	X. 1(f)	XI. 0(k)	XII. 0(m)
XIII. 0(g)	XIV. 0(l)	XV. 1(b)	XVI. 1(j)

PM- PATTERN MODE; CN-CELL NUMBER; ALP-ALPHABET

PM	CN	I	II	III	IV	V	VI	VII	VIII
OC	alp	e	i	a	h	p	c	n	d
	val	1	0	1	0	0	1	1	1
	CN	IX	X	X1	XII	XIII	XIV	XV	XVI
	alp	0	f	k	m	g	l	b	j
	val	0	1	0	0	0	0	1	1

So the element code is changed from FC44 to A743.

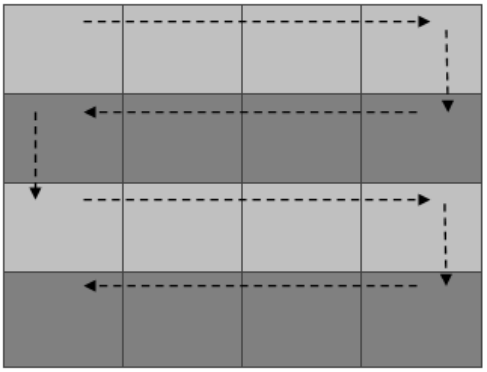
3.4. BASE CODE (32-bit)

BASE CODE = ASSIGNMENT MODE (8 bits) + PATTERN MODE(8 bits) = **0A + 0C**

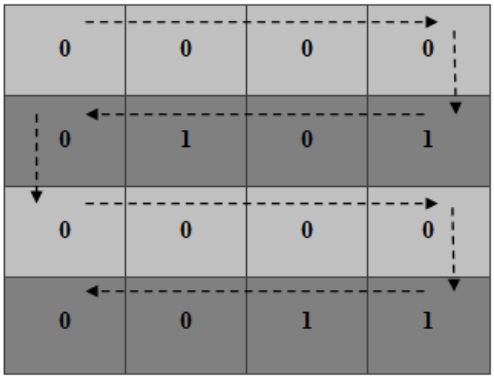
= **OAOC**

The BASE code is also represented in terms of a 4X4 matrix called BASE MATRIX. But the bit arrangement (PATTERN MODE) in this matrix is always fixed.

The BASE CODE is OAOC .Equivalent binary is **0000 1010 0000 1100**



The pattern in which alphabets are Arranged is fixed is shown by arrows. This is always fixed



4. ELEMENT CODE

ELEMENT CODE (32 bits) = BASE CODE (16 bits) + SOURCE CODE (16 bits)

BASE CODE = ASSIGNMENT MODE (8 bits) + PATTERN MODE(8 bits)

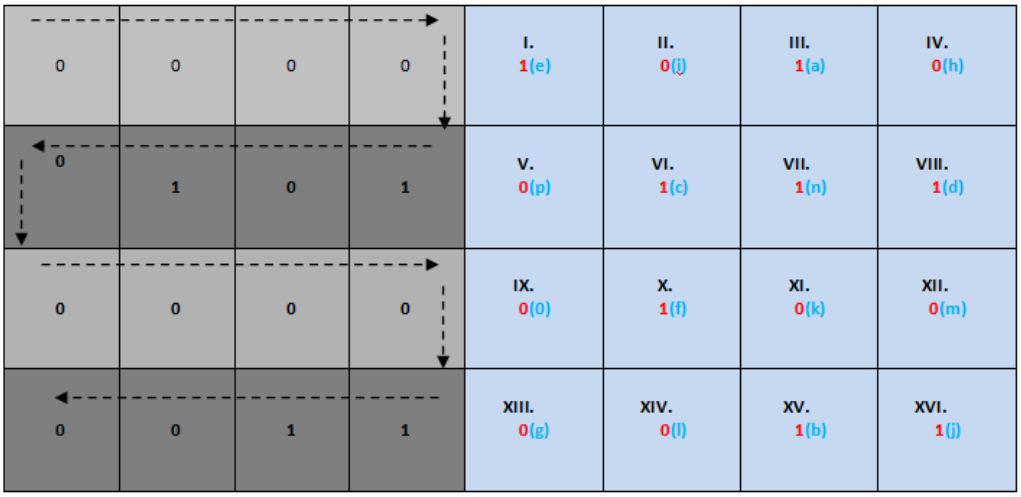
ELEMENT CODE (32 bits) = ASSIGNMENT MODE + PATTERN MODE + SOURCE CODE

= **0A + 0C + A743**

= **OAOCA743**

Thus the 32 bit code is generated for element.

The 32 bit code is represented in terms of a **4X8** matrix, which is the **ELEMENT CODE MATRIX**. It is shown below



5. RANDOM ARRANGEMENT OF BITS IN THE MATRIX

The arrangement of the bits in the above matrix (say 'A') is done by a newly proposed mathematical approach in the following way. Let the rows in the matrix be 'M' and columns be 'N'. The bit position in the matrix is given by variables i , j where i < M and j < N. The bits are represented as $A_{i,j}$.

Now choose any random variables say x,y . For the bit $A_{i,j}$ to find its new random bit position ,these x,y are added with i.j which results in new variables p,q. These variables p,q should be subtracted by the least number of multiples of M,N such that x<M and y<N. Now interchange row number with column number and vice versa.

The mathematical representation is as follows

Let the matrix be of the order M, N.

Let the bit be represented as $A_{i,j}$.

Let the randomly chosen variables be x,y.

let p ,q be two instance variables

$p = i + x$ and $q = j + y$

If p > M and q > N , subtract k_1 times M from p and K_2 times N from q where k_1 and K_2 are smallest possible whole numbers such that

$$p - K_1 M = u \leq M \quad \text{and} \quad q - K_2 N = v \leq N$$

Now the bit $A_{i,j}$ is changed to $B_{v,u}$ where B is the new matrix with re-arranged bits.

$$A_{i,j} \rightarrow B_{v,u}$$

NOTE: The above procedure is only valid for a square matrix

So we convert our rectangular matrix (4 X 8) into nearest possible square matrix (6 X 6) with the 4 dummy bits (36-32 = 4).

5.1 CONVERSION TO SQUARE MATRIX

We convert our rectangular matrix (4 X 8) into nearest possible square matrix (6 X 6) with the 4 dummy bits (36-32 = 4). This is done in the following way

Let M1 X N1 be the order of the rectangular matrix A and let the bits in the matrix be expressed in terms of $A_{p,q}$ where p is the row number and q is column number.

Let M2 X N2 be the order of the nearest possible square matrix and let the bits in the matrix be expressed in terms of $B_{r,s}$ where r is the row number and s is column number.

For the square matrix let M2 = N2 = K.

If N1 > M1, then find n ,the scaling factor

$$n = N1 - N2 .$$

Then the bits in the square matrix B are given as

for all $q < K$, $B_{r,s} = B_{p,q} = A_{p,q}$.

and for $q \geq k$, $B_{r,s} = B_{q-n,p} = A_{p,q}$.

If M1 > N1, then find n ,the scaling factor

n = M1 – M2 .

Then the bits in the square matrix B are given as

for all $p < K$, $B_{r,s} = B_{p,q} = A_{p,q}$.

and for $p \geq k$, $B_{r,s} = B_{q,p-n} = A_{p,q}$.

The following diagrammatically shows the above

Original rectangular matrix

0,0	0,1	0,2	0,3	0,4	0,5	0,6	0,7
1,0	1,1	1,2	1,3	1,4	1,5	1,6	1,7
2,0	2,1	2,2	2,3	2,4	2,5	2,6	2,7
3,0	3,1	3,2	3,3	3,4	3,5	3,6	3,7

Here the order of rectangular matrix is 4 X 8.(M1,N1)

And the square matrix is of the order 6 X 6 (M2,N2)

K = M2 = N2 = 6.

Scaling factor n =N2 - N1 = 8 – 6 = 2.

The bits in the cells represented in red can be placed directly in the square matrix.

The bits in the cells represented in black needs further processing to be placed in the square matrix.

For element in 0,6 of the rectangular matrix.

$6 \geq k$, so $A_{0,6} = B_{6-2,0} = B_{4,0}$.

Similarly $A_{0,7} = B_{5,0}$; $A_{1,6} = B_{4,1}$; $A_{1,7} = B_{5,1}$;

$A_{2,6} = B_{4,2}$; $A_{2,7} = B_{5,2}$; $A_{3,6} = B_{4,3}$; $A_{3,7} = B_{5,3}$;

New square matrix

0,0	0,1	0,2	0,3	0,4	0,5
1,0	1,1	1,2	1,3	1,4	1,5
2,0	2,1	2,2	2,3	2,4	2,5
3,0	3,1	3,2	3,3	3,4	3,5

4,0	4,1	4,2	4,3	4,4	5,4
5,0	5,1	5,2	5,3	5,4	5,5

The bits represented in green are the dummy bits.

5.2 CONVERSION OF ELEMENT MATRIX TO SQUARE MATRIX

The modified element matrix is shown below i.e.., after converting the 4X8 matrix into 6X6 matrix by the above mentioned procedure.

Transformed element matrix

0	0	0	0	1	0
0	1	0	1	0	1
0	0	0	0	0	1
0	0	1	1	0	0
1	1	0	1	0	1
0	1	0	1	0	1

The dummy bits (in green) are randomly chosen.

5.3 RANDOM ARRANGEMENT OF BITS

The bits in the above transformed element matrix are randomly arranged by the procedure mentioned above after taking the random variables x,y as 3,4.

Ex: The first bit in the transformed element matrix be $B_{0,0}$.

From the above mentioned procedure

$B_{x+0,y+0} = B_{3,4} = B_{p,q}$

These variables p,q should be subtracted by the least number of multiples of M,N such that x<M and y<N i.e.., $K_1 = 0$ and $K_2 = 0$ for this case.

p- $K_1 M = u \leq M$ and q- $K_2 N = v \leq N$

p- $0 * M = u \leq M$ and q- $0 * N = v \leq N$

$p = u \leq M$ and $q = v \leq N$

Now the bit $B_{p,q}$ is changed to $B_{v,u}$ where B is the new matrix with re-arranged bits.

$$B_{p,q} \rightarrow B_{v,u}$$

$$B_{0,0} \rightarrow B_{4,3}$$

Similarly all other bits are reordered. the table below shows the reordered locations of the bits for the random variables x,y=3,4.

Rearrangement table-1

original location	reordered location	original location	reordered location	original location	reordered location
0,0	4,3	2,0	4,5	4,0	4,1
0,1	5,3	2,1	5,5	4,1	5,1
0,2	0,3	2,2	0,5	4,2	0,1
0,3	1,3	2,3	1,5	4,3	1,1
0,4	2,3	2,4	2,5	4,4	2,1
0,5	3,3	2,5	3,5	4,5	3,1
1,0	4,4	3,0	4,0	5,0	4,2
1,1	5,4	3,1	5,0	5,1	5,2
1,2	0,4	3,2	0,0	5,2	0,2
1,3	1,4	3,3	1,0	5,3	1,2
1,4	2,4	3,4	2,0	5,4	2,2
1,5	3,4	3,5	3,0	5,5	3,2

The matrix with randomly arranged bits is shown below.

Matrix E

1	0	0	0	0	0
1	1	1	0	1	0
0	0	0	1	0	0
0	1	1	0	1	1
0	1	0	0	0	0
0	1	1	0	1	0

5.4 144-BIT ENCRYPTION

Each of the bits in the above matrix are expressed in terms of another 4 bits which are placed in a 2 X 2 matrix .The 4 bits are chosen such that the output of a fixed logical operation (randomly chosen by server per request from client) yields the corresponding bit.
For demonstration let the randomly chosen logical expression be AB + CD.

S.NO	A	B	C	D	AB	CD	AB+CD
1	0	0	0	0	0	0	0
2	0	0	0	1	0	0	0
3	0	0	1	0	0	0	0
4	0	0	1	1	0	1	1
5	0	1	0	0	0	0	0
6	0	1	0	1	0	0	0
7	0	1	1	0	0	0	0
8	0	1	1	1	0	1	1
9	1	0	0	0	0	0	0
10	1	0	0	1	0	0	0
11	1	0	1	0	0	0	0
12	1	0	1	1	0	1	1
13	1	1	0	0	1	0	1
14	1	1	0	1	1	0	1
15	1	1	1	0	1	0	1
16	1	1	1	1	1	1	1

The serial numbers for the output ‘1’ are 4,8,12,13,14,15 and 16 and for the output ‘0’ are 1, 2, 3,5,6,7,9,10 and 11.

Each bit is formed with the logical operations of four grouped bits in the below matrix.

Let us consider first group of four bits of a=0, b=1,c=1 and d=1 from the above matrix. We have ab=0, cd=1 and ab+cd=1 which is the first bit present in the matrix E.

Similarly all the above bits are arranged in the below table.

Original 144 bits

0	1	0	0	0	0	0	0	0	1	0	1
1	1	0	0	0	1	1	0	0	0	0	1
1	0	1	1	1	1	0	1	1	1	0	1
1	1	0	0	0	1	1	0	1	0	0	1
1	0	1	0	1	0	1	1	0	1	0	0
0	0	0	1	1	0	1	1	0	1	0	0
0	1	0	1	1	0	0	1	1	1	1	1
0	0	1	1	1	1	1	0	0	1	1	0
1	0	1	1	0	0	0	1	0	0	0	0
0	0	1	1	0	1	0	1	1	0	0	0
1	0	1	1	0	0	1	0	1	1	1	0
0	0	0	0	1	1	0	1	1	0	1	0

Again the bits in the 2 X 2 grouped matrixes are rearranged randomly as per the procedure discussed earlier with random variables x,y as 3,4.

Rearrangement table-2

original location	reordered location	original location	reordered location
0,0	0,1	1,0	1,1
0,1	0,0	1,1	1,0

The final 144 bit matrix is shown below

Reordered 144 bits ready for transmission,

1	1	0	0	0	1	0	0	1	0	1	1
0	1	0	0	0	0	0	1	0	0	0	0
0	1	1	0	1	1	1	0	1	0	1	0
1	1	1	0	1	0	0	1	1	1	0	0
0	0	0	1	0	0	1	1	1	1	0	0
1	0	1	0	1	1	1	1	0	0	0	0
1	0	1	1	0	1	1	0	1	1	1	0
0	0	0	1	1	1	0	1	1	0	1	1
0	0	1	1	0	1	1	0	0	0	0	0
1	0	1	1	0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1	1	0	0	0
1	0	1	0	0	1	1	0	1	1	1	1

Thus the plaintext ‘A’ is encrypted into 144 bits.

6. THE DECRYPTION

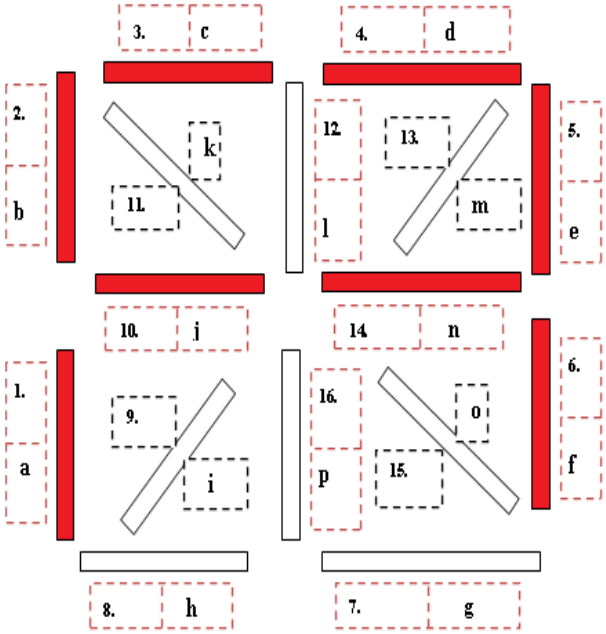
The 144 bits received are divided by 4 , we get 36 sets of 4 bit 2 X 2 matrices. The bits in each set are arranged back to their original positions using the **Rearrangement table-2** which results in the **original 144 bits** . Then the given logical operation (AB + CD in this case) is performed with the 4 bits of the each set. The output of each set will be 1-bit . The entire set (4 bits) is replaced with its corresponding output. This result in a 6X6 **matrix E**. the bits in the **matrix E** are arranged back to original location with the help of **rearrangement table-1.** This gives the **Transformed element matrix.** The **Transformed element matrix** which is a square matrix of the order 6 X 6 is converted back to **original rectangular matrix i.e.., the ELEMNT CODE MATRIX** of the order 4 X 8.

The **ELEMNT CODE** has 32 bits. The higher 16 bits are separated to find the **BASE CODE**. The **BASE CODE** is further divided into two sets of 8-bits. The upper 8 bits gives the **ASSIGNMENT MODE** and the lower 8-bits give the **PATTERN MODE.**

Depending on the **ASSIGNMENT MODE** the alphabets (a-p) are assigned to the 16 LED’s of the segmented display (display with alphabets assigned).

The lower 16 bits of the **ELEMENT CODE** is the **SOURCE CODE** .The element code obtained will be **A743** (in hex) which is not the actual **SOURCE CODE**. The original **SOURCE CODE** is obtained by the use of **PATTERN MODE.** The original **SOURCE CODE** will be equal to **FC44** (in hex). From this Source code the LED’s for which the assigned alphabet has bit ‘1’ are switched on and the

remaining off. The element 'A' will be displayed on the segmented display (as shown below) which is the original plain text character.



7. COMPARISION AND FUTURE WORK

The proposed algorithm is compared with the existing algorithms like the Message –Digest (MD), secure Hash (SHA)and the Whirlpool. The output size of the proposed encryption algorithm is 144 bits which is a bit more than the 128 bits in MD2,MD4 [1] ,MD5[2] , but less than the 160 bits in SHA [3]and 512 in whirlpool.

The internal state size is 36 which is very small compared to other existing algorithms like the 128 bits for MD5, 160 bits for SHA, and 512 bits for Whirlpool[4] . The word size is 16 bits which is again smaller than all the MD algorithms and the SHA, except for the whirlpool which has 8 bits size . The future work includes evaluating the complexity of the proposed algorithm and the complexity of the collision attacks on it.

8. CONCLUSION

A new 144 bit encryption algorithm is presented in this paper which can offer enhanced privacy and security. The encryption is done using segmented displays. The paper also introduces concept of PATTERN MODE, ASSIGNMENT MODE and also a new mathematical approach of randomly arranging the bits in a matrix with the use of random variables x,y.

A new 144 bit encryption algorithm is presented in this paper which can offer enhanced privacy and security. The encryption is done using segmented displays. The paper also introduces concept of PATTERN MODE, ASSIGNMENT MODE and also a new mathematical approach of randomly arranging the bits in a matrix with the use of random variables x,y.

9. ACKNOWLEDGMENTS

Our thanks to ACM for giving us this excellent opportunity. we also thank SASTRA university for encouraging us to participate in this event.

10. REFERENCES

[1] Ronald L. Rivest: The MD4 Message Digest Algorithm. Internet : RFC 1320 April 1992.

[2] Ronald L. Rivest: The MD5 Message Digest Algorithm. Internet : RFC 1321 April 1992.

[3] Xiaoyun Wang, Yiqun Lisa Yin and Hongbo Yu, Finding Collisions in the Full SHA-1, Crypto 2005

[4] S. L. M. Barreto, V. Rijmen, "The WHIRLPOOL Hashing Function," *First open NESSIE Workshop*, Leuven, 13-14 November 2000.

[5] Florent Chabaud, Antoine Joux: Differential Collisions in SHA-0. CRYPTO 1998. pp56–71

[6] Petitcolas, F. A. P., Anderson, R. J., Kuhn, M. G.: Information Hiding: A Survey. Proc. of the IEEE, Vol. 87, No.7, pp.1062-1078 (1999).

[7] Bruce Schneier. *Applied Cryptography*. John Wiley & Sons, 1996.

[8] Data Security -By Johnson and Peter (Tata Mc -Hill)

[9] Secure communication -By Marquis (Tata Mc -Hill)