IOSR Journal of Electrical and Electronics Engineering (IOSR-JEEE)
ISSN: 2278-1676 Volume 4, Issue 2 (Jan. - Feb. 2013), PP 20-27
www.iosrjournals.org

Microcontroller Based Cryptosystem with Key Generation Unit

CH.Gopi [1], M. Veda Chary[2]*, M.A. Khadar Baba [3]
[1] M. Tech (ES) Final Year, [2] Assoc. Professor, [3] Professor & HOD, Dept. of ECE, CMR College of Engineering & Technology, Hyderabad, Andhra Pradesh

Abstract: The embedded based applications need sensitive data transfer between different nodes. In order to increase the speed and to reduce the hardware complexity, this proposed system focuses on the light weight security algorithm Tiny Encryption. Algorithm TEA which can be implemented in microcontroller to adapt with many real time constraints such as memory, data loss and low cost. The additive feature of this proposed system is that it uses Key Generation Unit (KGU) to produce the random key to make it optimal for sensitive data transfer in many real-time applications. This above work uses microcontroller and the performances of this cryptosystem is analyzed by implementing the cryptographic algorithm TEA with key generation unit. The key generation unit uses the timers in microcontroller to generate the random bits. The work extends with implementing the two different modes of communication serial (UART) and wireless transmission (RF) to transfer the data from encryption unit to decryption unit.

I. Introduction:

` The communication system which requires sensitive data transfer uses secured cryptographic algorithms to convert the data into an unrecognizable format. These algorithms are classified into symmetric and asymmetric, which employs private and public keys respectively. The symmetric cipher is further classified into stream and block ciphers. This proposed paper focuses on the block cipher which allows feasibility for the key generation and these generated keys are used for cryptographic applications with reduced hardware complexity. In the existing system, the hardware implementation of block ciphers has limited feasibility in scheduling the key which is the primary resource for high secured data transfer. Since the existing system uses predefined key for the encrypting process, the system can offer narrowed security level though they use complex security algorithms. The major drawback in the existing system is that there is no key generation unit to increase the efficient change of key parameter for a secured data transfer. The proposed system implements the above statements using the light weighted, secure and efficient block cipher TEA with different modes of communication. The two modes of communication include serial (UART) and wireless communication (RF transmission) and the proposed system uses KGU to increase the key security further. The work also extends with the randomness test for the generated key using the principle component analysis method. This above research work uses 89c51 microcontroller and the performances are analyzed by implementing the KGU with TEA which offers moderate security and simplicity in implementation processes. In this paper, section 2 deals with the related works, section 3 describes the implementation of TEA with KGU. The section 4 explores the performance analysis and the results of proposed system. The conclusion is drawn in section 5.

II. Related Works:

The software analysis of different block ciphers using 8-bit AVR microcontroller were presented in. The performance of these block ciphers are compared with that of the Advanced Encryption Standard (AES) implementation and it has been proved that TEA consumes less memory than all the others. The implementation of TEA [2] is discussed using PIC18F4550 and shows that it can operate at 586 bytes per second. In [8], the unpredictable voltage oscillation signals generated by the oscillator are used to generate random bits and uses n number of oscillators to get the sample values. These values are fed to LFSR to produce robust random numbers. The proposed paper presents the implementation of TEA with KGU in microcontroller with reduced cost and increased speed which makes it suitable for many area and cost constraint applications such as RFID and wireless communications.

III. Implementation Of Tea With Kgu

A. Tiny Encryption Algorithm

TEA offers Shannon's twin properties of diffusion and confusion using the mixed (orthogonal) algebraic groups, to improve the security. Due to this feature, the TEA is a light weighted cryptographic algorithm which makes it suitable for many real time area constraint applications. The figure 1 shows the architecture for TEA encryption process.

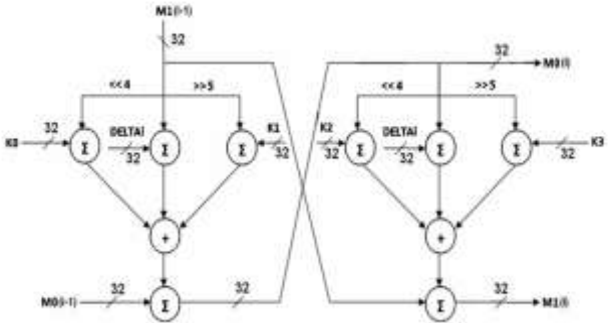


Figure 1 TEA Encryption Process

TEA operates on 64 (block size) data bits at a time using a 128-bit key with 32 rounds. TEA is an iteration cipher, where each round *i* has inputs M0[*i*-1] and M1[*i*-1] as in (2) and (3), which is derived from the previous round. The subkey K[*i*] is derived from the 128 bit overall K. The constant delta (_) in (1), is the derivative of golden number ratio to ensure that the subkeys are distinct.

$$\partial = (\sqrt{5} - 1) * 2\hat{\ }31 = 9E3779B9_{h} \qquad (1)$$

$$M0[i] = M0[i\text{-}1] \boxplus F(M1[i\text{-}1], K[0,1], DELTA[i]) \qquad (2)$$

$$M1[i] = M1[i\text{-}1] \boxplus F(M0[i\text{-}1], K[2,3], DELTA[i]) \qquad (3)$$

The round function F (4), is defined by

$$F(M,K[j,k],DELTA[i]) = ((M<<4) \boxplus K[j]) \oplus (M \boxplus DELTA[i]) \oplus ((M>>5) \boxplus K[k]) \qquad (4)$$

The single TEA round function performs the simple mixed orthogonal algebraic functions such as Right/Left shifts, Integer addition and exclusive – or operations. The steps carried out in round function:

1. The one half M1[i-1] of the block cipher is Left shifted by 4 times and Right shifted 5 times.
2. The left shifted block is added with the subkey K0 and right shifted block is added with the subkey K1.
3. It is also added with the constant delta value DELTA[i] which is the multiples of delta, where I represents the number of iterations.
4. The results are then Ex–ORed and added with the other half of the block cipher M0[i-1] which produces one half of the block cipher M0 for next iteration.
5. Similar operations are performed for the next half round function with the above result.

Finally, the Ex–ORed result is added with the first half of the block cipher M1[i-1] to produce the next half block M1 for the further rounds. Similar operations are performed for decryption process which is described in figure 2. In this case, the constant delta value DELTA(i-1) is “C6EF3720”, where ‘i’ represents the number of iterations. In each iteration, the delta value “9E3779B9” is subtracted with the constant delta value.

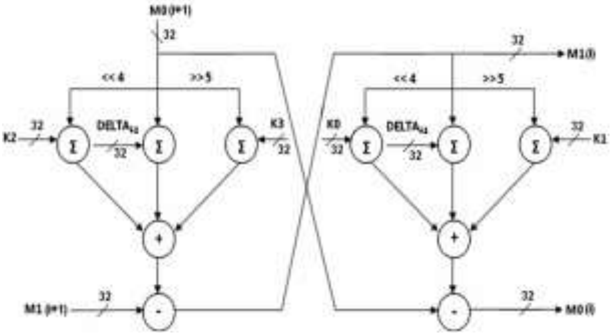


Figure 2 TEA Decryption Process

Since, the TEA has a Feistel structure the reverse operation of encryption process is performed to obtain the plain text in the decryption process.

B. Block Description for TEA with KGU

The main block diagram of "MICROCONTROLLER BASED CRYTOSYSTEM WITH KEY GENERATION UNIT" is depicted in the figure 3, which shows the complete operation of the system. The principle feature of this system is the usage of KGU to generate the random bits which is used as a key for security algorithm. The security algorithm is chosen in such way that it occupies reduced area and time constraints, with high performance. TEA is such block cipher which is known for its high security with reduced area.

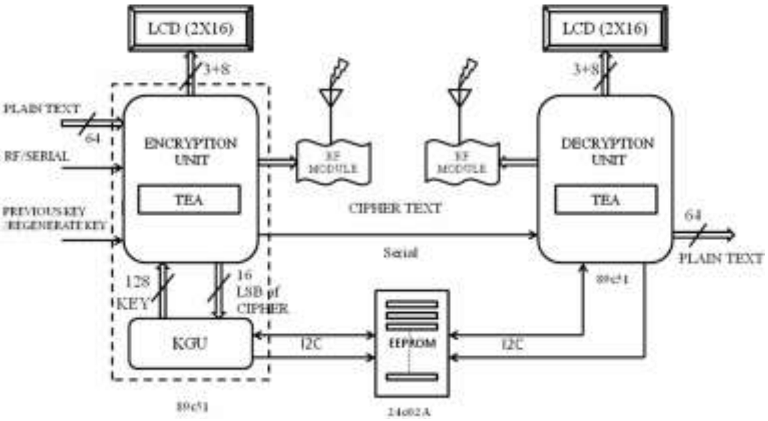


Figure 3 Block Diagram of TEA with KGU

The encryption unit receives the key from KGU to encrypt the data. It has the feasibility to choose either of the modes of transmission, in additive with the option of choosing the predefined/currently generated key. When the system is working with serial mode of transmission, the key and cipher text along with mode are transmitted through the serial port. In case of RF transmission mode, the decryption unit receives the above data through wireless communication.

The algorithm is implemented using Atmel 89c51 with reduced code size and increased speed. The performance of the system is discussed in the following chapters, based on the number of cycles it takes to execute the encryption and decryption process, throughput and code size. The speed of the controller varies depend upon crystal frequencies (say 24MHz,12MHz).The block comprises of the following units:

1. KGU
2. Encryption unit
3. Decryption unit

1) KGU: The KGU is implemented using timers in the microcontroller to generate the random bits. The generated random numbers acts as a key for block cipher and is stored in EEPROM, so that the key is secured and it is then transferred to the decryption unit. The figure 4 depicts the block diagram of the KGU.

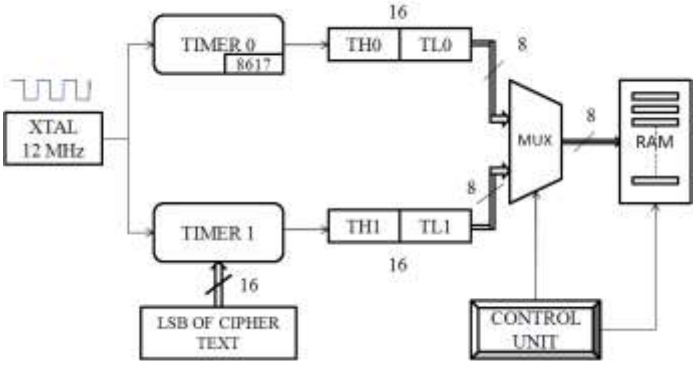
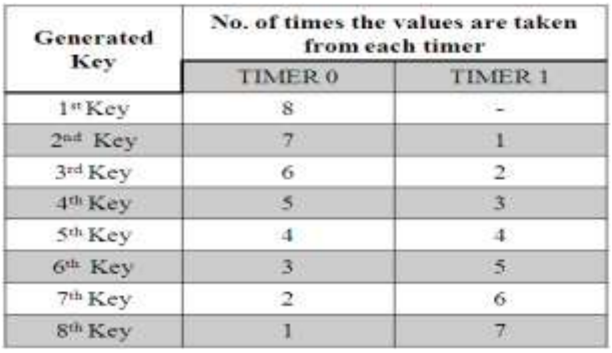


Figure 4 KGU

The KGU uses the in-built timers (T0 & T1)which accept the input from the crystal oscillator by dividing its value by 12. In order to generate 128 bit key from 16 bit timers, it has to take 8 samples from the timer. These sample values are taken from the timers, as per the table I.

TABLE I. SAMPLE VALUES



Generated Key	No. of times the values are taken from each timer	
	TIMER 0	TIMER 1
1st Key	8	-
2nd Key	7	1
3rd Key	6	2
4th Key	5	3
5th Key	4	4
6th Key	3	5
7th Key	2	6
8th Key	1	7

The table I indicates the number of samples to be taken from each timer. This repeats for every 8 set of keys. The initial value for the timer0, to generate the first key is predefined and its starts running with the normal timer operation, then the sample values are taken from the timers whenever it is required, as per the table I whereas, the timer1 is not used for the generation of first key.For the initial value of second key, the timer1 uses the LSB (16-bit) value of most recently updated cipher txt. This value is then replaced by the LSB of updated cipher text, for the next key generation, and this repeats for every key generation process. The control unit is used to select either of the timer values to take the samples. These generated keys are then stored in the EEPROM for the use of decryption process. The advantage of using such method will increase the strength of the key generation unit and makes it more complex for related key attacks.

2) Encryption Unit: The figure 5 shows flow chart for the encryption unit which illustrates the overview of the encryption block. As it is described in the flow diagram, the encryption process starts with the operation of key generation followed by receiving the plain text, performs TEA encryption process and selects the mode of communication (RF & Serial) to transmit the data to decryption unit. Finally, display the encrypted result in LCD and repeats the above process. Figure 5

The function of the key generation unit is shown in figure 6. The function starts with initializing the timer0 and number of samples for both the timers (Timer0 & Timer1). The timers generate the random bits of size 16-bit. The random bits from the timers are stored in the RAM until the sample count reaches zero. The pattern for these sample values are shown in the table I

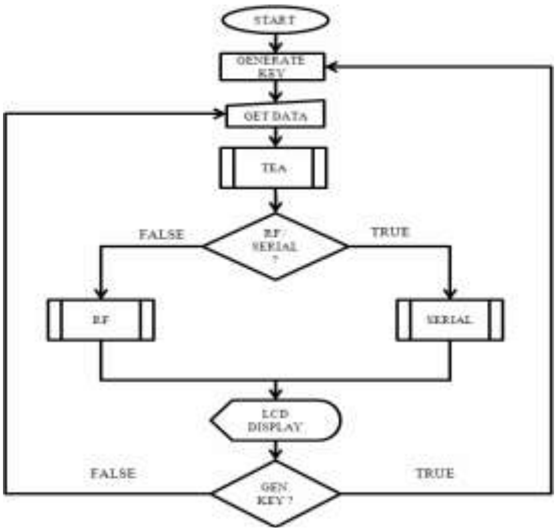


Figure 5 Flow chart for Encryption Unit

. The sample count is initialized as 8 to get 128-bit key. The counter is decremented and once the sample count reaches zero, the sample values of the timers are reinitialized, where the timer1 is initialized with the LSB of updated cipher text for the generation of next set of random bits as it is described in the KGU.

This below process is done only when the user prefers to generate a new set of random bits otherwise it uses the predefined random bits.

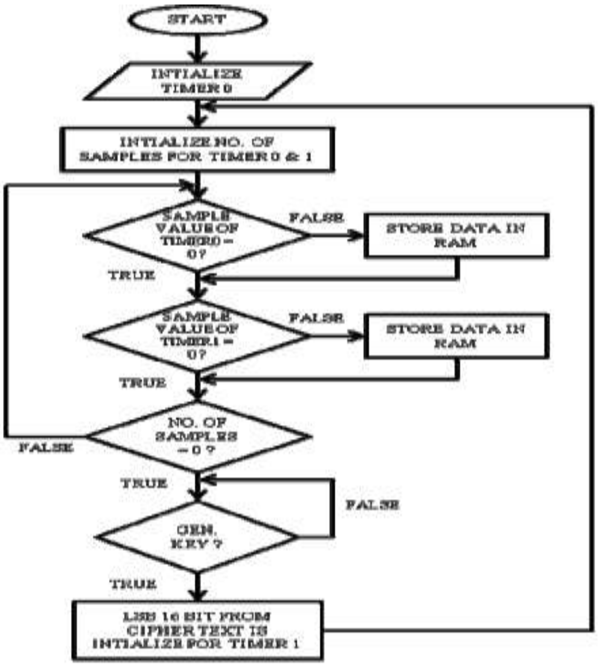

START
INTIALIZE TIMER 0
INTIALIZE NO. OF SAMPLES FOR TIMER 0 & 1
SAMPLE VALUE OF TIMER0 = 0?
FALSE
STORE DATA IN RAM
TRUE
SAMPLE VALUE OF TIMER1 = 0?
FALSE
STORE DATA IN RAM
TRUE
NO. OF SAMPLES = 0 ?
FALSE
TRUE
GEN. KEY ?
FALSE
TRUE
LSB 16 BIT FROM CIPHER TEXT IS INTIALIZE FOR TIMER 1

Figure 6 Flow chart for Key Generation Unit

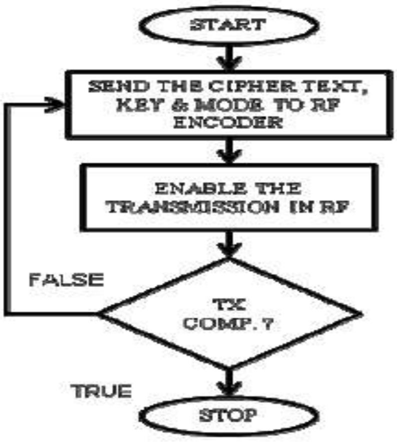

START
SEND THE CIPHER TEXT, KEY & MODE TO RF ENCODER
ENABLE THE TRANSMISSION IN RF
FALSE
TX COMP. ?
TRUE
STOP

Figure 7 : Flow chart for RF Transmitter

The figure 7 represents the flow of RF transmitter. It requires a RF encoder to make compatible with the RF transmitter. The steps involved with the serial communication are shown in figure 8, which includes the initialization of timers. This process involves with the mode and cipher text whereas, the key is stored in EEPROM which is used later for the decryption process.

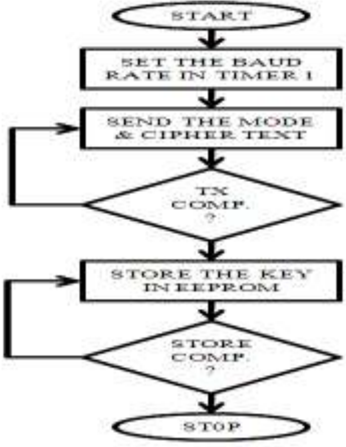


Figure 8 Flow chart for Serial Communication

3) Decryption Unit: **The processes performed by the**

Decryption unit are shown in the figure 9. The two different modes of operations that are performed in this unit are normal and interrupt modes. The normal mode deals with the RF transmission and interrupt mode performs the serial data transfer which receives the data through UART. In the case of serial communication it requires to receive the key from EEPROM to perform TEA decryption process. The LCD is used to display the received cipher text, key and the decrypted text (plain text).

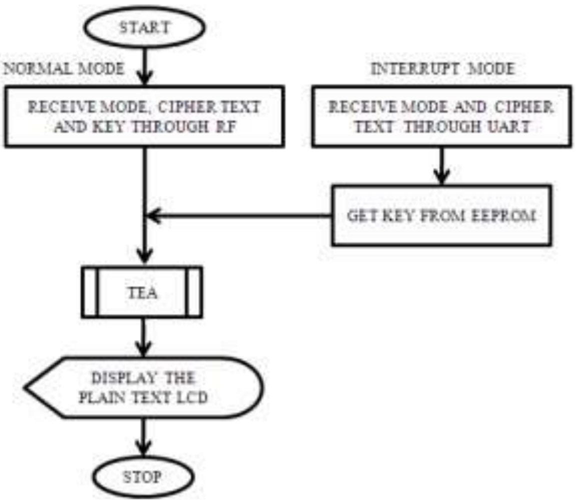
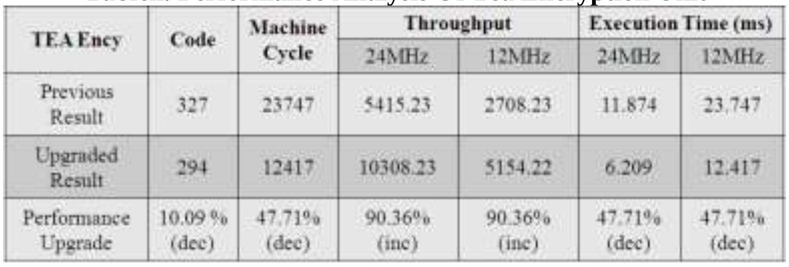

START
NORMAL MODE
INTERRUPT MODE
RECEIVE MODE, CIPHER TEXT AND KEY THROUGH RF
RECEIVE MODE AND CIPHER TEXT THROUGH UART
GET KEY FROM EEPROM
TEA
DISPLAY THE PLAIN TEXT LCD
STOP

Figure 9 Flowchart for Decryption Unit

IV. Performance Analysis Of Tea Implementation :

The performance analysis of TEA encryption is shown in the table II in terms of code size, machine cycles, throughput and execution time with the upgraded performance. From the table it is inferred that the throughput is improved to 10308.23bps in 24MHz and 5154.22bps in 12MHz. The additive upgraded result with the execution time is that the time is decreased down to 47.71%.

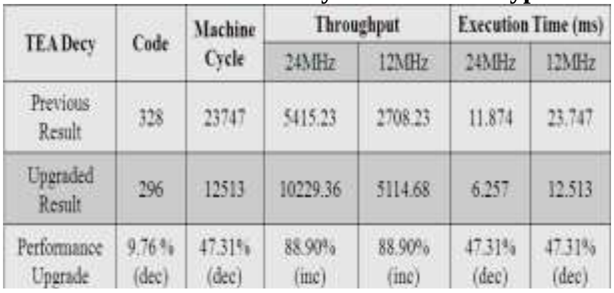
Tableii. Performance Analysis Of Tea Encryption Unit



TEA Ency	Code	Machine Cycle	Throughput		Execution Time (ms)	
			24MHz	12MHz	24MHz	12MHz
Previous Result	327	23747	5415.23	2708.23	11.874	23.747
Upgraded Result	294	12417	10308.23	5154.22	6.209	12.417
Performance Upgrade	10.09 % (dec)	47.71% (dec)	90.36% (inc)	90.36% (inc)	47.71% (dec)	47.71% (dec)

Similar comparison is made with the TEA decryption
which is illustrated in the table III whose upgraded performance of throughput is increased to 88.90% and the execution time is decreased down to 47.31%.

TABLE III. **Performance Analysis Of Tea Decryption Unit**



TEA Decy	Code	Machine Cycle	Throughput		Execution Time (ms)	
			24MHz	12MHz	24MHz	12MHz
Previous Result	328	23747	5415.23	2708.23	11.874	23.747
Upgraded Result	296	12513	10229.36	5114.68	6.257	12.513
Performance Upgrade	9.76 % (dec)	47.31% (dec)	88.90% (inc)	88.90% (inc)	47.31% (dec)	47.31% (dec)

The performance of KGU is depicted in the table IV whose throughput is 1641025.64bps in 24MHz and 820512.82bps in 12MHz. In the case of execution time, it is very high when compared with TEA processes.

TABLEIV.Performance Analysis Of Kgu Unit

KGU	Code	Machine Cycle	Throughput		Execution Time (µs)	
			24MHz	12MHz	24MHz	12MHz
Previous Result	105	183	1391304.35	699453.55	91.5	183
Upgraded Result	93	156	1641025.64	820512.82	78	156
Performance Upgrade	11.43% (dec)	14.75% (dec)	56.92% (inc)	56.92% (inc)	36.27% (dec)	36.27% (dec)

The table V shows the encryption module along with the KGU, TEA, I2C and Serial units, whose throughput is 17516.65bps in 24MHz and 8758.38bps
in 12MHz.

TABLEV. Performance Analysis Of Encryption Module :

Modules	Code Size	Machine Cycles	Through Put	
			24MHz	12MHz
KGU, TEA, I2C & SERIAL	582	21922	17516.65	8758.38
TEA, I2C & SERIAL	489	21766	17642.19	8821.10
SERIAL	35	7784	16444	8273

A. Randomness Test Results for Generated Random Bits : The figure 10 shows the randomness test result. The result is obtained from the Unscrambler 9.8 software using the Principle Component Analysis method. It is also proved that the proposed method (Cipher mix random bits) provides better result with distinct random bits.

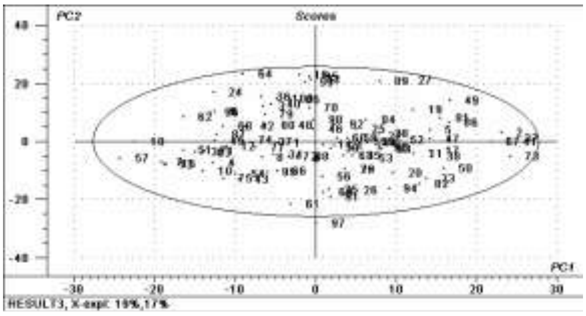
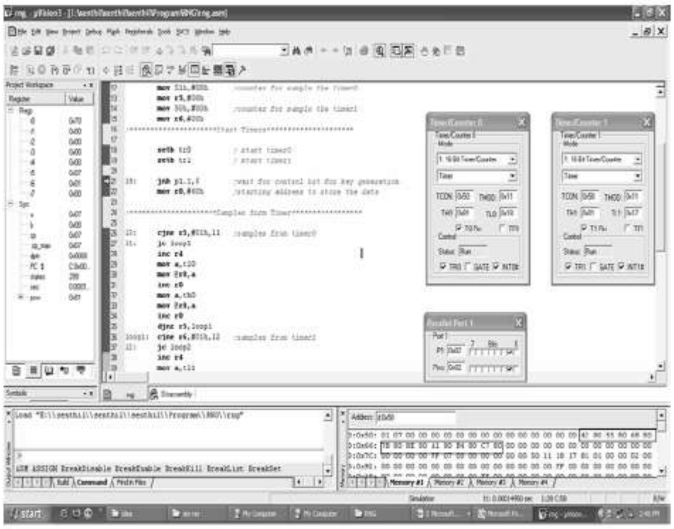


Figure 10 Randomness test results for KGU – Cipher mix random bits

B. Simulation Results:
The figure 11, 12 and 13 shows the simulation results for KGU, TEA encryption and decryption units respectively.


```
        mov 31h,#00h        ;counter for sample the timer0
        mov r5,#00h
        mov 30h,#00h        ;counter for sample the timer1
        mov r6,#00h
;**********************Start Timers**********************

        setb tr0            ; start timer0
        setb tr1            ; start timer1

l0:     jnb p1.1,$          ;wait for control bit for key generation
        mov r0,#60h         ;starting address to store the data

;*******************Samples from Timer*****************

l2:     cjne r5,#11h,l1     ;samples from timer0
l1:     jc loop1
        inc r4
        mov a,tl0
        mov @r0,a
        inc r0
        mov a,th0
        mov @r0,a
        inc r0
        djnz r5,loop1
loop1:  cjne r6,#11h,l2     ;samples from timer1
l2:     jc loop2
        inc r4
        mov a,tl1
```

```
Load "E:\\senthil\\senthil\\senthil\\Program\\RNG\\rng"
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

Figure 11 Simulation Result for KGU

Figure 12 Simulation Result for TEA Encryption Unit

Figure 13 Simulation Result for TEA Decryption Unit

V. Conclusion

The system employs the utilization of TEA algorithm, which is a light-weight block cipher and offers low power and area consumption with moderate security. The additional feature of this system is the usage of KGU to generate the random key which still improves the security of data transfer. Therefore, the system is suitable to implement in the high secured low cost applications. The implementation of TEA using AT89C51 is to reduce the cost of the system. The application includes various wired and wireless communications which requires secured data transfer

References:

[1] Soren Rinne, Thomas Eisenbarth, and Christof Paar, "Performance Analysis of Contemporary Light-Weight Block Ciphers on 8-bit Microcontrollers", Horst Gortz Institute for IT Security Ruhr University Bochum 44780 Bochum, Germany.

[2] Edi Permadi, "The Implementation of Tiny Encryption Algorithm (TEA) on PIC18F4550 microcontroller", Electrical Engineering 2005, President University.

[3] Devesh C. Jinwala, Dhiren R. Patel, Department of Computer Engineering S. V. National Institute of Technology, INDIA; Kankar S. Dasgupta Space Applications Centre, Indian Space Research Organization, INDIA, "Investigating and Analyzing the Light-weight ciphers for Wireless Sensor Networks", March 10, 2009.

[4] Thomas Eisenbarth, Ruhr University Bochum; Sandeep Kumar, Philips Research Europe; Christof Paar and Axel Poschmann, Ruhr University; Bochum Leif Uhsadel, Catholic University of Leuven, "A Survey of Lightweight-Cryptography Implementations", IEEE Design & Test of Computers dtco-24-06-pos.3d 4/10/07.

[5] P.Israsena, Thailand IC Design Incubator (TIDI) National Electronics & Computer Technology Center (NECTEC), "Design and implementation of low power hardware encryption for low cost secure RFID using TEA", 2005 IEEE ICICS.

[6] Issam Damaj, Samer Hamade and Hassan Diab "Efficient tiny hardware cipher under verilog", in High Performance Computing & Simulation Conference, 2008.

[7] Laszlo Hars, Cortlandt Manor, NY (US), "Switching electronic circuit for Random Number Generation", US Patent August 3, 2004.

[8] Laszlo Hars, Cortlandt Manor, NY (US),"Latching electronic circuit for random number generation", US Patent October 17, 2006.

[9] David A.Carlson, Haslet, TX (US); Gregg A.Bouchard. Round Rock,TX (US); Anand Varadharajan Framingham, MA (US); Derek S. Brasili. Westminster,MA (US), "Random Number Generator", US Patent October11,2005..