International Journal of advancement in electronics and computer engineering (IJAECE)
Volume 1, Issue 2, May 2012, pp.62-68, ***ISSN 2278 - 1412***

Arm Microcontroller Implementation of Des Using Concept with Time-Variable Key

Amol D. Tupkar[1], Prof. U.A. Rane[2]
[1]*P.G. student (M.E. Digital Electronics), SSGMCE. Shegaon, SGB Amravati University, Maharashtra, amoltupkar3@gmail.com, India;*
[2]*Assistant Professor, Dept. of ECE, SSGMCE. Shegaon, SGB Amravati University, Maharashtra, raneuan@yahoo.com, India;*

Abstract - *In this paper we describe ARM Microcontroller implementations of the Data Encryption Standard (DES) algorithm, with time-variable key technique to increase its immunity to Cryptanalysis attack. This proposed scheme changes the key with time. Using time-variable key technique enhances the security of DES because the same plain text is ciphered to different cipher texts by time. Hackers will face difficulty to hack into the proposed scheme because of time-variant behavior. This method provides more security against cryptanalysis attack than that provided by conventional old DES algorithm. LPC2148 ARM microcontroller was selected as the target for implementation with the embedded C as the software programming language.*

Keywords*: Cryptographic algorithm, DES, ARM Processor, Encryption algorithm, Embedded Systems and applications.*

I. Introduction

The Data Encryption Standard (DES) [1],[2] was published by the United States national bureau standards (NBS) in January 1977. It has been used by United States federal agencies since 1977. It is used in IPSec protocols, ATM cell encryption, and the Secure Socket Layer. The plaintext is encrypted to cipher text by the key with a length of 64 bits, in which 56 bits are used for encryption, and others are employed for parity test. Encryption and decryption use the same algorithm as well as the key. It is no longer a question to attack the 56-bit key with the development of computer technology. The attackers can hack into DES within 20 hours through exhaustive key search. There are different algorithms to enhance DES. T-DES [3] was proposed to increase the length of the key which uses 3 keys. It takes more times thrice than that for DES execution. Also, G-DES [6] was proposed to obtain faster algorithm but it is less safe than DES. This paper introduces a strategy to obtain time-variable key. The first strategy depends on a counter. For every counter value, the main key is rotated by this value. The second strategy changes the main key value according to the output of a pseudorandom number generator that enables us to increase the randomness of key variation. Every time, the used key is changed, so hackers will face difficulty to attack the proposed schemes. This paper also describes the ARM Microcontroller implementation of the two strategies. The two strategies are implemented with embedded C as a programming language.

ARM LPC2148 was selected because of it ISP feature, along with embedded C because it offers high flexibility for up gradation of algorithm, with low cost design which will suit the need of innovative embedded application. [10][11].

II. Overview of DES Algorithm

DES is a block cipher, as shown in Fig. 1. It takes 64-bit input and 64-bit key. A 64-bit output is produced. The effective key length of key is 56 bits because every 8th bit is used as parity checking bit. The DES algorithm consists of 16 rounds as shown in Fig.1 Data is 64-bit firstly permutated and then divided to 32 bits blocks Right Plain Text (RPT) and Left Plain Text (LPT)[12]. It is processed through DES function as shown in Fig.2 in which 32 bit RPT is expanded to 48 bits to be processed through XOR function with the round key. The XOR output is converted from 48 bits to 32 bits through Substitution boxes (S-boxes). The S boxes output is XORed with the 32 bit LPT and the output is the RPT to the next round. The RPT bits of the previous round are the LPT of the next round as illustrated by the following formula:

$$L_i=R_{i-1}, \quad R_I=L_{i-1} \oplus F(R_{i-1}, K) \qquad (1)$$

There are 16 different round sub keys that are generated from the main key. Each round has its own sub key generated from the main key. Fig. 3 shows the sequence of generating the 16 sub keys. The main key is firstly permutated and then will be shifted according to the round number. The round number indicates the shift bits. After that, the output is permutated by the second permutation.

The same algorithm can be used for encryption or decryption. The method described above will encrypt a 64 bit block of plaintext and returns a 64 bit block of cipher text. In order to decrypt the cipher text and get the original plaintext again, the procedure is simply repeated but the sub keys are applied in reverse order from K(16) to K(1). Fig. 2 Show the function of a DES round. In each round the RPT and LPT of plain text are processed. The 32-bit RPT is expansion permutated to 48-bits by repeating specific bits. This permuted text is then XORed with the 48 bit sub key K of specific round. The 48 bit XOR output is then applied to S-Box for substitution using a lookup table mechanism. 6 bit XOR data is applied to eight S-boxes. Each S-box outputs 4 bit data. 32 bit output from 8 S-boxes is XORed with unused LPT which in turns becomes the RPT for next round of DES. While the RPT becomes LPT for next round of DES. The same thing is repeated by selecting appropriate sub key.

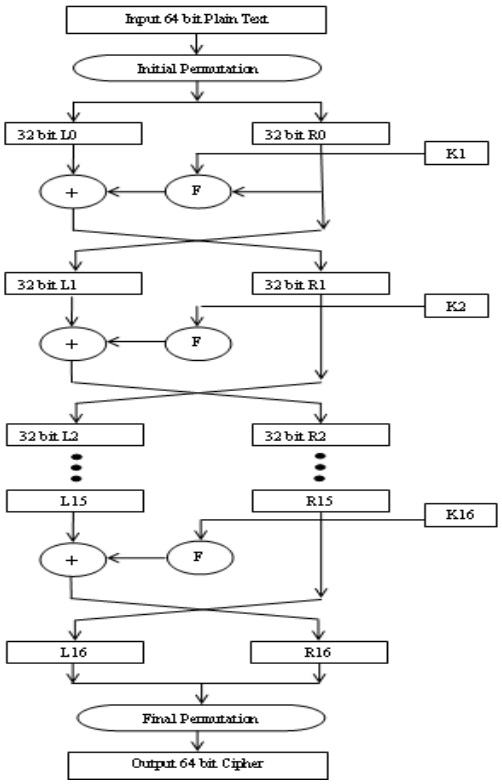

Fig. 1. DES Algorithm

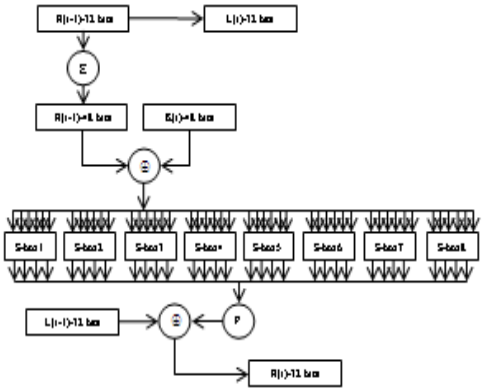

Fig. 2. Function of DES

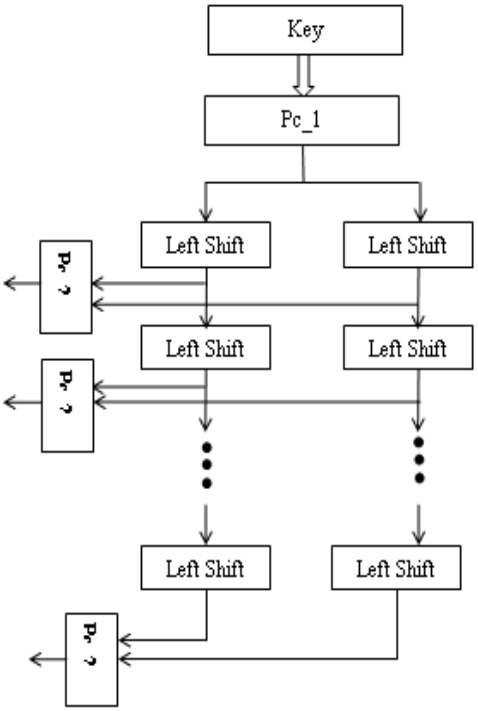

Fig. 3. DES Sub-key Generation

Fig. 3. Shows the method of generation of sub keys required for each DES round. Initial 56 bit key is formed by dropping the parity bit from original 64 bit key. This 56 bit key is split into two different 28 bit stream each stream is left shifted by one bit. This two left shifted bits are then concatenated by Pc_2 function to generate 48 bit permuted sub-key K1 which is used in first round of encryption [12]. Similarly K1, K2….K16 are generated by shifting number of bit location as prescribed in DES standards.

III. DES using Time Variable key

In order to obtain time-variant behavior that increases the security of DES,[2] the new scheme for key generation has been used as shown in Fig. 5. For any counter value, the main key is rotated using this value. For example, when the counter value is 2 (N=2), the main key is rotated two times to achieve a new key that will be used in encryption process [3]. As a result of this, for the same plain text, there are different cipher texts because of the time variant behavior. The counter value is encrypted and then transmitted with the ciphered data to guide the receiver to use the intended key[1].

The disadvantage of this scheme is that the counter sequence is repeated after time, so we have to develop a randomness property in key variation.

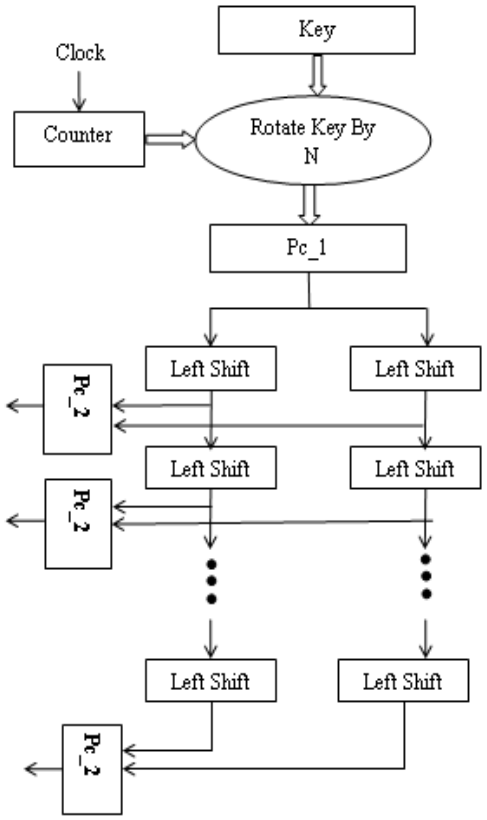

Fig. 4. Key Variation based on counter

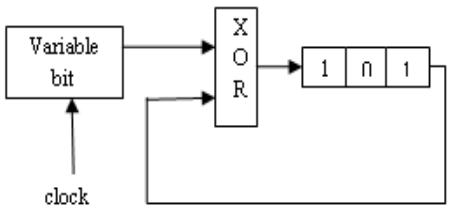

Fig. 5. Pseudorandom number generator

Fig.5 Describes a pseudorandom number generator which is based on Linear Feedback Shift Register (LFSR). This type is also known as

a hybrid type [5] because it depends on feedback property and variable input that is varied with the clock. The start value that will be in the register is called a seed. Every clock cycle the rotation process is achieved through the XOR function. Using a pseudorandom number generator to control key variation as shown in Fig. 7 increases the randomness of key variation. So, it enhances the security of the algorithm.

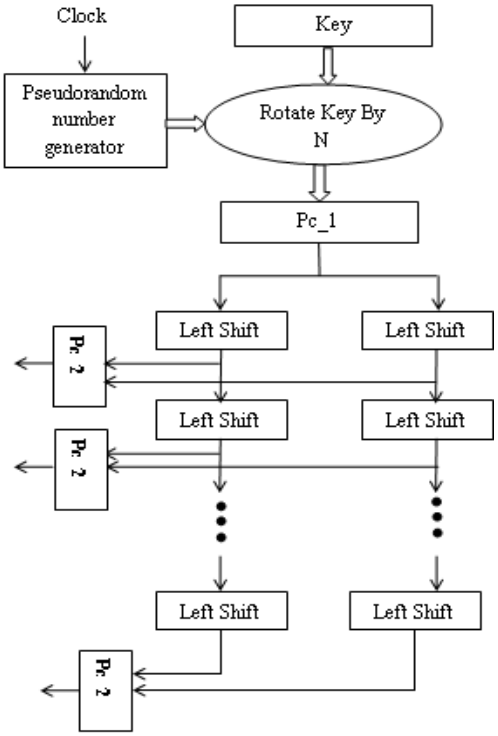

Fig. 6. Key Variation based on pseudorandom number generators

Fig..6. Describes the key variation based on pseudorandom number generation using Linear Feedback Shift Register (LFSR). Focus of this paper is to study implementation of DES using these two schemes.

IV. Implementation of DES using Embedded C

These two schemes were implemented using Philips LPC2148 microcontrollers are based on a 32/16 bit ARM7TDMI-S CPU with real-time emulation and embedded trace support, that combines the microcontroller with embedded high speed flash memory of 512 kB. Using Triton IDE tool for programming with embedded C language.

The DES algorithm was implemented by defining functions for various steps in DES [6][7]; also look up table approach was preferred to have a table for the expansion and for each permutation. The selection functions **("S**boxes"S1... S8**)** were implemented as eight two-dimensional lookup tables.

Following are some of the important functions used in the program each function is commented with its operation.

```
#include <LPC21xx.h>
#include<board.h>
/*Header files supporting ARM Microcontroller*/
void des_parity_key_permutation(int *, int *);
/*for removal of parity bit from 64 bit key to form 56 bit key*/
void des_make_half(int *, int *, int *);
/* splits 56 bit key into two 28 bit parts*/
void des_single_shift(int *, int *);
/* shifts 28 bit key part by single bit position*/
void des_double_shift(int *, int *);
/* shifts 28 bit key part by 2 bit position*/
void Pc_2(int *,int *,int *);
/* concatenates two 28 bit key parts to single 56 bit key for subkey generation*/
void des_permutation_48(int *,int *);
/* key compression permutation from 56 bit to 48 bit*/
void des_permutation_64(int *,int *,int *);
/*Initial permutation of 64 bit plaintext and splitting it in LPT & RPT*/
void des_round(int *,int *,int *,int *,int *);
/*DES function on LPT, RPT plaintext and subkey to give LPT & RPT for next DES round*/
Void des_ permut_48(int *,int *);
/*Expansion of 32 bit RPT to 48 bits*/
void common_permutation(int *,int *);
/*Final/Inverse permutation*/
void des_counter_initialization(int *,int *)
/*initialize counter*/;
void des_counter_shiftingnumber(int *,int *);
/*reads number form counter to decide shift in encryption key*/
void des_randomnumber_generation(int *,int *);
/*reads random number form linear shift register to decide shift in encryption key*/
```

Additional functions were also used for plaintext to hex, hex to binary, binary to hex, and hex to plain text conversion.[13].

Peripheral function for interfacing Hex keypad and 16x2 LCD display with ARM were also used along with UART function for interfacing HyperTerminal of Desktop PC with ARM LPC2148 with a Baud rate of 38400. Mode of encryption and decryption was selected as Electronic Code Book (ECB).

V. Results

In our Implementation plain text can be taken from HyperTerminal as well as hex keypad interfaced to the ARM development board[10][11]. In this paper result from HyperTerminal input are shown.

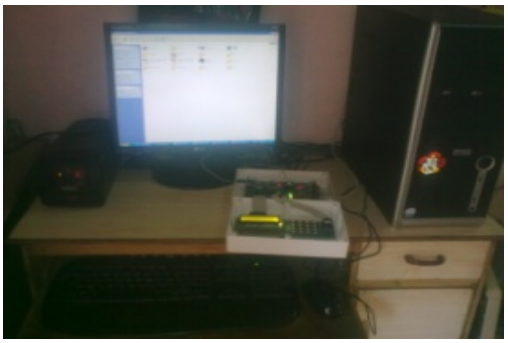

Fig. 7(a). Experimental setup for programming LPC 2148 board

Fig. 7(b). ARM LPC2148 board

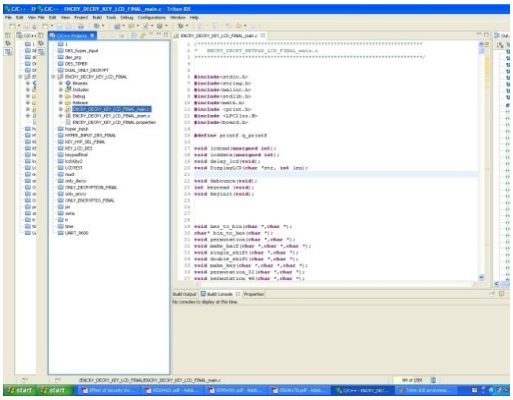

Fig. 8. Triton IDE software

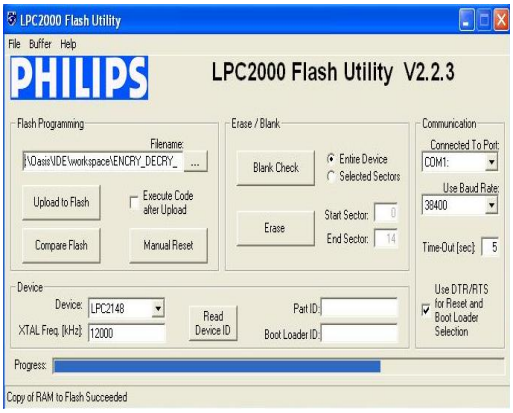

Fig. 9. Burning time varying key DES HEX file to ARM microcontroller

Fig. 7(a), 7(b) & Fig.8 display the images of the setup and IDE used for the implementation. Fig. 9. displays the HEX file burning procedure to the ARM in ISP mode using Philips flash Utility tool. Following figures shows the result of the implementation. HyperTerminal was used to transmit the plaintext to the ARM development board using UART at 38400 baud rate. Fig. 10 displays the plaintext transmitted through the HyperTerminal and the encrypted text received from the ARM board by using DES using pseudorandom number generator approach. The plaintext is 128 bit wide so DES [4][5]required two ECB execution, though the plaintext was repeated twice ABCDEFGH & ABCDEFGH but their equivalent encrypted cipher was not repeated which confirms the variable key approach of our implementation.

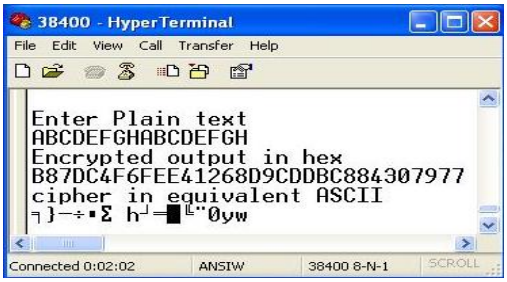

Fig. 10. Pseudorandom variable key approach encrypted output

Fig. 11 show the decryption of same ciphertext using synchronous pseudorandom number generation approach which is output to LCD displayed. The decrypted text was same as input plaintext, which showed successful implementation of variable key technique.

Fig. 11. Decrypted data output on 16x2 LCD

The counter approach of variable key was also implemented on same board successfully.

When counting the total clock cycles required for encryption of 64 bit data it was observed that time required for counter approach was more as compared to the random number generator approach.

Also the randomness in counter approach was less as compared to LSFR linear shift register random number generator.

VI. Conclusion

In this paper, two designs for DES encryption/decryption algorithm are implemented on ARM LPC2148 hardware in combination with part of software using embedded C. Our two designs depend on time-variable key concept. We used two methods to obtain time-variable key. The first one uses the counter to change the key and the second uses the hybrid pseudorandom number generator to increase the randomness of key variation. For the same plaintext and key, the ciphered text is varied with time. As a result of this, the security of the algorithm has been increased. The security provided second approach was more because of more randomness in its key generation as compared to first method. But the speed of encryption was more for first method was more.

References

[1] FIPS PUB 46-3, *Data Encryption Standard (DES),* January 15, 1977. Reaffirmed 1999 October 25.

[2] National Bureau of Standard (U.S.), "DES modes of operation," *Federal Information Processing Standard Publication 81, National Technical Information Service*, December 1980.

[3] Triple Data Encryption Algorithm Modes Of Operation, X9.52 – 1998,Accredited Standards Committee X9, *American National Standards Institute*, July 27, 1998.

[4] McLoone, McCanny," High-performance FPGA implementation of DES using a novel method for implementing the key schedule," *Circuits, Devices and Systems*, vol. 150, pp. 373-378, 2003.

[5] Matthew Kwan. *Bitslice DES.* May 1998,*http://www.darkside.com.au/bitslice/*

[6] M. Matsui, "linear cryptanalysis method for DES cipher," *Advances in Cryptology - EUROCRYPT '93,* vol. 765, pp. 387-397, May 1993.

[7] E.Biham, "A fast new DES implementation in software," *4th International Workshop on Fast Software Encryption, FSE '97*, pp.260-271, Israel, 1997.

[8] V. Patel, R. C. Joshi and A. K. Saxena, "FPGA implementation of DES using pipelining concept with skew core key-scheduling," *Journal of Theoretical and Applied Information Technology*, vol. 5, no. 3, pp. 295- 300, March 2009.

[9] K. Wong, M.Wark, E. Dawson, "A single-chip FPGA implementation of the data encryption standard (DES) algorithm,"*IEEE*

Globecom Communication, vol.2, pp. 827-832, Sydney, Australia, 1998.

[10] ARM Architecture. Reference *Manual ARM DDI 0100D*, ARM Limited, Feb 2000..

[11] ARM7. Data Sheet ARM DDI 0020C, ARM Limited, Dec 1994.

[12] Stallings, William, *Cryptography and Network Security*, Prentice Hall, 1998.

[13] Sklyarov, Valery, FPGA-based implementation of recursive algorithms *Microprocessors and Microsystems*, 28, March 2004.

Author's Profile

Mr. Amol D. Tupkar*, has done his B.Tech in E&TC SGGSIE&T, from Nanded in 2008. He is ME scholar in Digital Electronics from SSGMCE, Shegaon from Sant Gadge Baba University, Amravati, Maharashtra, India. His working areas are Microcontroller, Microprocessor, VLSI Design.*

Prof. Umesh Arun Rane*, is graduate in BE Electronics and Power Engineering in 1989 from Nagpur University. He has completed his M.Tech in Electronics from Nagpur University in 1992. He is having Teaching experience of 18 years.*
Presently he is working as Assistant Professor at Shri Sant Gajanan Maharaj College of Engg., SHEGAON, Maharashtra, India. His working areas are Microcontroller, Microprocessor Power Electronics, VLSI Design..