# 확장된 DES 구현

한 승 조† · 김 판 구††


## 요 약

DES 암호 알고리즘을 대체 할 수 있는 112 비트의 키 길이를 갖는 EDES라는 새로운 알고리즘이 [1, 2]에서 제안 되었다. 평문은 96비트로 입력되며, 이것은 각각 32비트의 3개의 서브 블록으로 분리된다. EDES는 S-box를 8비트에서 16비트로 증가하였으며, 3개의 서브블럭에 있는 f 함수들이 비대칭적으로 수행되기 때문에 differential cryptanalysis 보다 암호 강도가 강하다. 본 논문은 EDES에 대한 하드웨어의 설계와 VLSI로 구현되는 것을 제안한다. 암호화와 복호화는 VLSI의 단일 칩내에서 구현되도록 하였으며, 만약 시스템의 클럭 주파수가 15Mhz이면 약 90Mbit/sec로 암호·복호화 할 수 있다. 그러므로 구현된 칩은 초고속망의 프로토콜에서 온라인으로 데이터를 암호화 하는데 적용될 수 있다.


# Implementation of the Extended Data Encryption Standard (EDES)

Seung-Jo Han† · Pan Koo Kim††


## ABSTRACT

A new encryption algorithm had been proposed as a replacement to the Data Encryption Standard (DES) in [1, 2]. It called the Extended DES (EDES) has a key length of 112 bits. The plaintext data consists of 96 bits divided into 3 sub-blocks of 32 bits each. The EDES has a potentially higher resistance to differential cryptanalysis than the DES due to the asymmetric number of f functions performed on each of the 3 sub-blocks and due to the increase of S-boxes from 8 to 16.

This paper propose a hardware design for the EDES and its implementation in VLSI. The VLSI chip implements data encrytion and decryption in a single hardware unit. With a system clock frequency of 15Mhz the device permits a data conversion rate of about 90 Mbit/sec. Therefore, the chip can be applied to on-line encryption in high-speed networking protocols.


## 1. Introduction

The Data Encryption Standard (DES)[3] has been adopted in 1977 by the National Bureau of Standards for ensuring privacy of information. Since them increasing concerns have been raised about its resistance against well directed cryptographic attacks in view of: 1) the continual advances in the processing power of hardware which will make the DES vulnerable to methods of exhaustive attacks in the near future[4] and 2) recently developed techniques such as differential cryptanalysis which have been shown to be effective in breaking the DES[5]. Those concerns have led

※This paper was supported by the research fund of Chosun University, 1996.

† 정 회 원:조선대학교 공과대학 전자·정보통신공학부
†† 정 회 원:조선대학교 정보과학대학 전자계산학과

논문접수:1997년 1월 6일, 심사완료:1997년 5월 28일

the National Institute of Standard and Technology (NIST) to search for a replacement algorithm. The various approaches that have been suggested to strengthen the cryptographic security of the DES have been shown to have drawbacks[6, 7, 8].

The importance of date decreases over time. As the speed of hardware has increased, encrypted data can be recovered faster. We are in need of algorithms that stretch the time of a cryptographic attack (recovering encrypted data), so as to decrease the value of recovered data to a minimum. An algorithm is unconditionally secure, if no matter how much ciphertext a cryptanalyst has, there is not enough information to recover the plaintext. the amount of computing time and power required to break it[9].

Until very recently, all encryption products were in the form of specilized hardware. These were encryption/decryption boxs that plugged into a communications line and encrypted all the data going across that line. Although software encryption is becoming more prevalent today, hardware is still the mode of choice for most commercial and military applications. There are several reasons why this is so[9].

The first is speed. Encryption algorithms contain many complicated operations on strings of bits. While some cryptographers have kept this in mind and have tried to make their algorithms more suitable for software implementation, specialized hardware will always win a speed race.

The second reason for the prevalence of hardware is security. There is no physical protection for an encryption algorithm written in software. Hardware encryption devices can be securely encapsulated to prevent this. Tamper-proof boxes prevent someone from modifying a hardware encryption device. Special-purpose VLSI chips can be coated with a chemical such that any attempt to access their interior will result in destruction of the chip's logic.

The final reason for the prevalence of hardware is the ease of installation. Most encryption applications do not invole general-purpose computers. People wish to encrypt their telephone conversations, facsimile transmissions,or data links. It is more efficient to put special-purpose encryption hardware in the telephones, facsimiles, and modems than it is to put in a microprocessor or software. The advantages of software implemantation is that any encryption algorithm can be implemented in it.

This paper implement in VLSI a DES-like cryptosystem which called the Extended-DES (EDES). The EDES was proposed as a replacement to the DES due to its resistance against cryptographic attacks.

## 2. Design of the Extended DES

The EDES can best be appreciated by comparing it to the conventional DES. The algorithm of the DES use for both encryption and decryption. The DES is a block cipher which operates on 64-bit blocks of plaintext and yields a 64-bit cipher text block. It uses a 56 bit key for both encryption and decryption. The DES uses transformations that alternately apply substitutions and permutations to the data. This process is repeated for 16 rounds. A plaintext block (a block of data) is first transposed under an initial permutation IP. The effect of this initial permutation is cancelled by applying the inverse permutation $IP^{-1}$ at the end of the 16th round to obtain the final result. After the initial permutation the block of the 64 permuted data bits is divided into sub-block designated $R_i$ and $L_i$ respectively where the subscript i indicates the i-th round.

As mentioned above, the same algorithm serves both for encryption and for decryption except that the round keys are used in reverse order. The encryption consists of 16 rounds of identical operations called f function, in which the data is combined with the key.

### 2.1 Basic Architecture

It is clear from the description of the DES that the left and right sub-blocks have the same number of f function during the 16 rounds. Therefore when ex-

tending the DES to provide additional resistance to cryptographic attacks, there is a need to reorganize the structure of the DES such that the f functions are different for each sub-block during each of the 16 rounds. One way to achieve this is to create an asymmetry by operating on a block of 96 bits of data and dividing it into 3 sub-blocks (A, B and C) of 32 bits each. If we use this sub-block division and the f function and assuming without loss of generality the initial fixed relation $A_i = B_{i-1}$, then the only combination involving the three sub-blocks equally and being the same for encryption and decryption is given by the following pairs:

$$\begin{aligned} B_i &= C_{i-1} \oplus f(B_{i-1}, K_i) \\ C_i &= A_{i-1} \oplus f(B_{i-1}, K_i) \end{aligned} \tag{1}$$

We could similarly assume the initial condition $C_i = B_{i-1}$. We would then obtain a mirror position with the exact same characteristice. The combination given by (1) is symmetric and therefore this is the selected form for the EDES algorithm. The realization of these equations is shown in (Fig. 1). The equivalence of the encryption and decryption algorithms can be found from comparing the respective relations. From (Fig. 1) the encryption equations are:

$$\begin{aligned} A_i &= B_{i-1} \\ B_i &= C_{i-1} \oplus f(B_{i-1}, K_i) \\ C_i &= A_{i-1} \oplus f(B_{i-1}, K_i) \end{aligned} \tag{2}$$

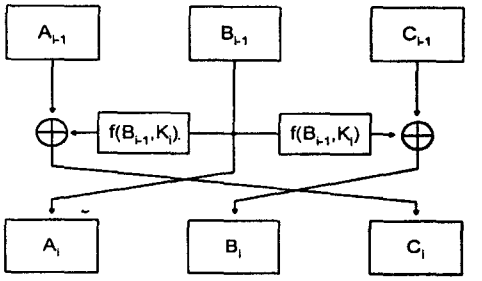
From equations (2), using the XOR operations on both sides of the last two equations of (2) and using $f(B_{i-1}, K_i) = f(A_i, K_i)$ the decryption equations are obtained as:

$$\begin{aligned} B_{i-1} &= A_i C_{i-1} = B_i \oplus f(A_i, K_i) \\ A_{i-1} &= C_i \oplus f(A_i, K_i) \end{aligned} \tag{3}$$

Interchanging A and B we obtain the following relations:

$$\begin{aligned} A_{i-1} &= B_i \\ B_{i-1} &= C_i \oplus f(B_i, K_i) \\ C_{i-1} &= A_i \oplus f(B_i, K_i) \end{aligned} \tag{4}$$

The decryption equations as shown in (4) have the same structure as the ones in (2). Therefore the same hardware can be used for both encryptiom and decryption. The encryption algorithm for the EDES is shown in (Fig. 2). As shown in the (Fig. 2), the data is divided into 3 sub-blocks of 32 bits each. The initial permutation block(IP) and its inverse block ($IP^{-1}$) are also extended to 96 bits.




(Fig. 1) Realization of equation. (1)

The f functions must operate on the right and left sides of the structure. These functions are labelled $f_1$ and $f_2$ for the left and right sides respectively. The number of S-boxes is increased to 16. They are labelled $S_1$ to $S_{16}$, with $S_1$ to $S_8$ on the left side and $S_9$ to $S_{16}$ on the right side. The $f_1$ function is associated with $S_1$ to $S_8$ and the $f_2$ function with $S_9$ to $S_{16}$. Finally according to the equations in the last round of encryption $A_{16}$ and $B_{16}$ should be interchanged.

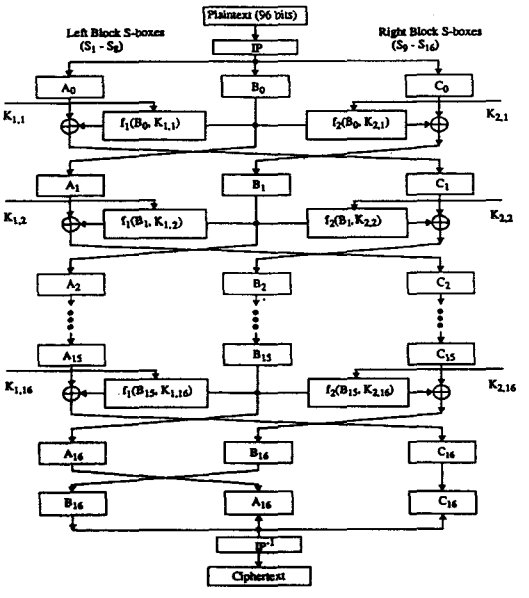
The EDES algorithm for decryption based on the equations stated above is shown in (Fig. 3). The decryption algorithm is the same as the encryption one, but $A_0$ and $B_0$ in the first round are interchanged. In order to have a different f function for each of the sub-blocks within each round the input key is extended from 64 bits to 128 bits. After removal of the 16 parity bits, the resulting input key is divided into

two keys of 56 bits each: $K_1$ on the left and $K_2$ on the right. After the first permuted choice box (labeled PC−1) the 56 bit keys are further divided into left and right keys of 28 bits each. During each round keys $K_1$ and $K_2$ are shifted left according to the key schedule. After passing through the permuted choice 2 box (labeled PC−2), they are reduced to 48 bits each labeled $K_{1,i}$ and $K_{2,i}$ on the left side and on the right side respectively for each round (i = 1 to 16).

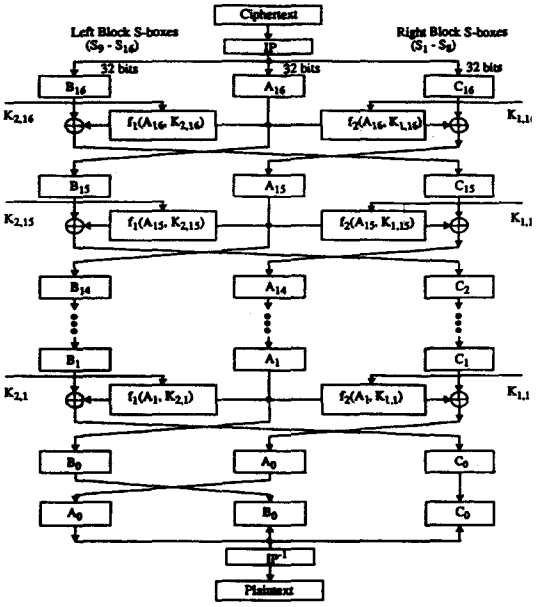
### 2.2 Key Schedule and f Function

In the decryption process the key is inserted in the reverse order to the one used in the encryption process. Also the S-boxes have to be interchanged. That is: $K_{1,16}$, $K_{1,15}$, ... $K_{1,1}$ and $K_{2,16}$, $K_{2,15}$, ... $K_{2,1}$ are interchanged: $K_{1,i}$ is now on the left and $K_{2,i}$ on the right. In addition the $S_1$ to $S_8$ boxes on the left should be interchanged with the $S_9$ to $S_{16}$ boxes on the right.

To make the EDES more resistant to differential cryptanalysis the number of f function operations has been increased. In addition this number is different




(Fig. 2) Encryption algorithm of the EDES




(Fig. 3) Decryption algorithm of the EDES

for the three sub-blocks. From (Fig. 4) it can be seen that in the EDES the f function is repeated 8 times in the process of converting $R_0$ to $L_{16}$ and $L_0$ to $R_{16}$, whereas in the EDES the f function is repeated 11 times when progressing from $A_0$ to $C_{16}$. Similarly for the two other sub-blocks, the f function operates 10 times between $B_0$ to $A_{16}$ and 11 times between $C_0$ to $B_{16}$. Thus for the same P, E and S-boxes, the EDES should provide better resistance to differential cryptanalysis attacks than the DES algorithm.

### 2.3 Design of the S-Boxes

The design of the S-boxes is crucial in determining the strength of the algorithm. In the DES all the security provided by the algorithm is based on the S-boxes as they are the only elements which are non-linear in module-2 arithmetic [10]. In order to improve the entries of the S-boxes used in the DES a careful analysis of the entries has to be made. This analysis based on the strict avalanche criterion (SAC) [11] as represented by the average bit probabilities ($P_{i,j}$) and their correlation coefficients [12] is presented in a more

comprehensive paper [1, 2] analyzing the EDES.

## 3. Implementation of the EDES

### 3.1 Blocks of the Extended DES

The main block of the Extended DES are the Data-In Register Set, the Data Register ("A", "B", and "C"), the Key Register, the Iteration Hardware, the Data-out Register set, and the Control/Status unit (partially complete). The main block of the EDES hardware implementation are shown in (Fig. 5).

The Iteration hardware consists of the following units:

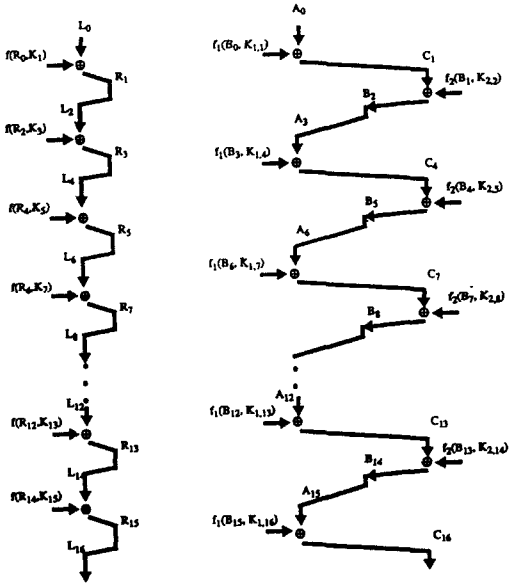
1) The key generation circuit.

2) The S-boxes.

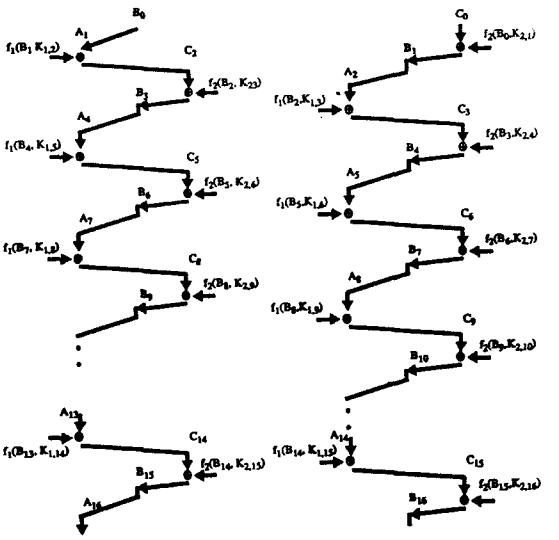
3) The XOR arrays.

4) The expansion and permutation Boxes.

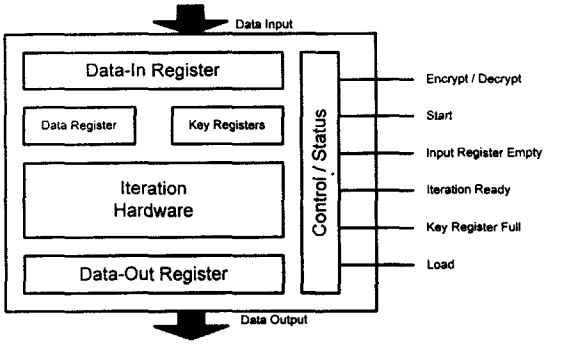
The Iteration Hardware and the registers shown above in (Fig. 5) from the datapath block. The data



(Fig. 4) Comparison of the number of operations of the f function in the DES and the EDES : (a) The DES : 8 operations, (b) $A_0$ sub-block of the EDES : 11 operations



(Fig. 4) Comparison of the number of operations of the f function in the DES and the EDES : (c) $B_0$ sub-block of the EDES : 10 operations, (d) $C_0$ sub-block of the EDES : 11 operations




(Fig. 5) Block of the EDES hardware implementation

is encrypted and decrypted in the datapath block. This is achieved by processing the incoming data out of the data registers with two sets of 16 sub-keys generated by the Key generation circuit. The sub-keys are derived from the 112-bit "master" key (the "master" key is the original encryption or decryption key loaded at the start of an encryption or decryption process) from loaded into the Key Register after parity removal. The Control/Status (CS) unit consists of two

controllers, each dedicated to controlling the datapath during Encryption and Decryption respectively. The Encryption and Decryption controllers govern the sub-key generation and the data multiplexing in the Datapath block[13]. Apart from the crypting controllers the CS unit has a system-level controller which controls the transfer of data to and from the chip. This control unit makes the transformation process independent of the communication with the rest of the process. The Status signals generated by the CS unit govern the timing of the next set of data input to the chip after the present data has been operated on (encrypted or decrypted).

### 3.2 Control and Status Signals

The interface to the outside world is based on a number of status signals. The START signal starts the encryption or decryption process (determined by the ENCRYPT/DECRYPT signal). After the first iteration the INPUT REGISTER EMPTY will indicate that new data can be read into the DATA IN register. At the end of a crypting cycle the result of the transformation is written into the DATA OUT register and the ITERATION READY signal will be active. If another 64-bit data word is already read the crypting can continue without any delay.

The key is read by a separate datapath. When the LOAD KEY signal is active the key and its parity bits are loaded into the key register. The circuit to check the parity of the key and generate the Key Error status signal has not been built yet and is a proposal for future work.

### 3.3 One Round of the EDES

A block diagram of the EDES datapath is shown in (Fig. 6). There are two multiplexers in the circuit. The multiplexer at the output of the S-BOX multiplexes between the function generated by S-BOX 1 to 8 and by S-BOX 9 to 16. These are two-to-one multiplexers. The output of these multiplexers is connected to one input of the XOR-gate array. The other input is the "A" or "C" register bits. One input of the multiplexer is the output of S-BOX 1 to 8 and the other input is the output of S-BOX 9 to 16. The controller selects which one to use depending on whether we are doing encryption or decryption. The output of the "B" register is connected to the intermediate (from position 33 to 64) bits of the DATA-OUT REGISTER. The output of the Left side XOR array is connected to the LSB's of the DATA-OUT REGISTER. (as shown in (Fig. 5))

The output of the "B" register is 32 bits. These 32 bits are expanded to 48 bits by passing them through an EXPANSION PERMUTAION BOX. The expansion permutation box is built by routing 32 input pins 48 output pins using metal 1 and metal 2 layers.

These 48 bits are XOR-ed in parallel with the 48 bits SUB-KEY from the KEY GENERATION CIRCUIT. One for the Left side and the other for the Right side. The 48 bits output of the EXPANSION BOX is routed in parallel to two paths. One set of the sub-key gets XOR-ed with one path and the other set gets XOR-ed with the other path. The 48 bits output of the Left XOR array goes to the 48 input bits of S-BOX 1 to 8. The 6 least significant bits go the S-BOX 1, the next 6 go to S-BOX 2 and so on. The 48 bits go to S-BOX 9.

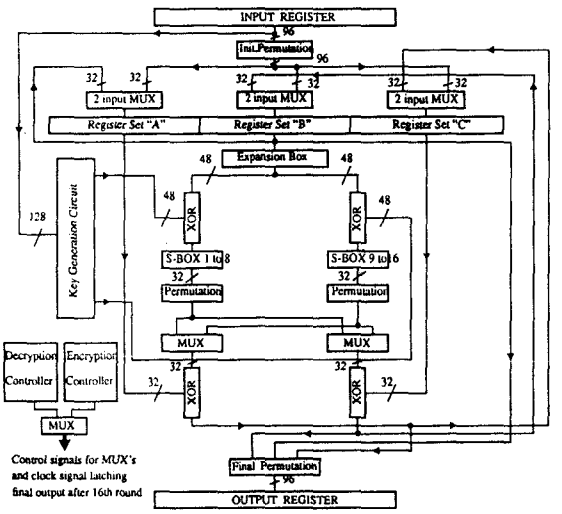
The 32 bits output of S-BOX 1 to 8 is XOR-ed with the 32 bits data stored in the "A" register. The 32 bits output of S-BOX 9 to 16 is XOR-ed with the 32 bits data stored in the "C" register. The output of the above mentioned XOR arrays goes back as feedback to the "A", "B" and "C" register, but basically this is what constitutes one round of the EDES algorithm. The output of the XOR-array which has as input the "A" register output and the output of S-BOX1 to 8 is fed back to the "C" register. The output of the other XOR-array is fed back to the "B" register. The output of the "B" register is fed back to the "A" register. The bits stored in the "A", "B", and "C" registers will be used in the next round as they go though the datapath.

### 3.4 Multiplexing between Data Inputs

A two-input multiplexer placed at the input of each of the "A", "B", and "C" registers selects between the loading of these registers at the start of the encryption or decryption process and the feedback at the end of a round. The "select" signal comes from the controller The first time the "A", "B", and "C" register are clocked, the multiplexer selects the data from the DATA-IN register. Now the "A", "B", and "C" registers contain the plaintext or the ciphertext.

### 3.5 16 Round EDES

Using time multiplexing the hardware built to accommodata one round can be used to carry out all 16 rounds of the EDES algorithm. The output of the datapath block capable of carrying out one round is connected to the DATA-OUT REGISTER. After 16 rounds are over, (calculated by the Encryption or Decryption controller) the DATA-OUT REGISTER is clocked. The DATA-OUT REGISTER now holds the ciphertext or the plaintext.

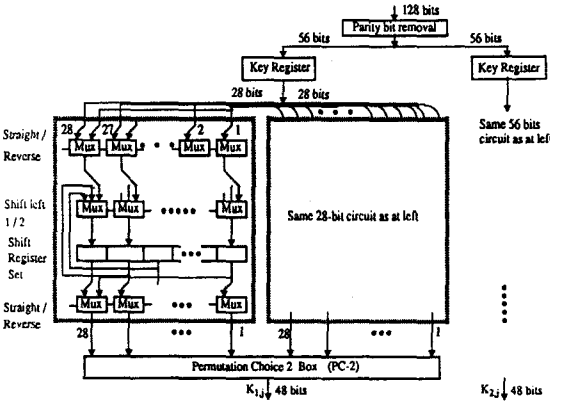



(Fig. 6) Implementation of one round of the EDES algorithm and ancillary circuits

### 3.6 Key Generation

The top-level architecture of the Key Generation Circuit is shown in (Fig. 7). We observe that the key-shift register has only three operations(loading, left shift by one, left shift by two) carrided out upon key. The architecture of one half of the key generation circuit which generates a 48 bit sub-key for each iteration is shown in (Fig. 7). A similar block generates the other 48 bit sub-key. A 3-to-1 multiplexer needs to be placed at the front end of each register of the Key-Shift Register set to carry out one of three functions as discussed above. One input of the multiplexer is the Key stored in the Key Register. The second input comes from the output of the shift register to the right of the register under consideration. The third input comes from the output of the shirt register two-away to the right of the register under consideration. This architecture enables us to either load the register, left shift by one or left shift by two the key in the shift register set.

The 56 bits sub-key generated by each of the K1 and K2 halves goes through a key permutation PC-2 box before it is XOR-ed with the data bits from the "B" register. The key-permutation box PC-2 generates a 48 bits output, compatible with the 48 bits data from the "B" register which is expanded to 48 bits after it passes though the Expansion box.




(Fig. 7) Key generation circuit

### 3.7 Controller

The controller is built as a ROM. The ROM has sixteen words for decryption and seventeen words for encryption. The output of the two controllers is multiplexed. The select signal of the multiplexer is connected to the pin encrypt/decrypt, which depends on the USER command of encryption or decryption.

The ROM word advances depending on the count of a 5 bit counter connected to the front-end of the decoders of both the controllers. Since all 16 rounds are carried out on the hardware which can accommodate only one round, it is the controller which keeps track of the sub-key generation for each round. The controller clocks the Final Register after 16 round are over. Each word of the controller is 7 bits wide. The 7 bits pertain to providing control signals for the following units.

1) 2-to-1 multiplexer to select between loading the "A", "B" and "C" registers and taking the feedback from the previous rounds.
2) 3-to-1 multiplexer to select between loading the key-shift register, shifting the data in the key-shift register set by 1 or by 2.
3) Providing Clock signals to the Final Output Register set, which would then contain either the plaintext or the ciphertext.

The counter which advances each word of the ROM works on a frequency of 15MHz.

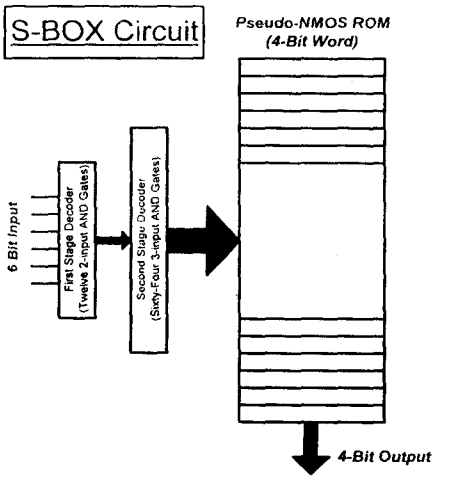
### 3.8 S-BOXES

The contents of the S-BOX is crucial to the proper operation of the EDES algorithm in context to security and correctness. In design the S-BOX was implemented as ROM's. At the front end of the memory element lies a two-stage decoder. The decoder has 6 inverters, 12 2-input and gates, and 64 3-input and gates.

If the six bit input to a particular S-box is x1, x2, x3, x4, x5, x6, and if the input pins of the S-BOX are named as a, b, c, d, e, f, then they are connected to each other in the following manner.

x1 ⇒ a, x2 ⇒ f, x3 ⇒ b, x4 ⇒ c, x5 ⇒ d, x6 ⇒ e

The first and the last bit select the row and the middle four bits select the column of the S-BOX. The selected element is the four-bit output. The decoder generates the 64 possible functions of six variables. Each word is 4 bits wide representing a number form 0 to 15. The block diagram of one S-BOX is shown in (Fig. 8).




(Fig. 8) Architecture of the S-box.

## 4. Implementation of EDES Chip and Simulation

The design process of the EDES chip can be classified into the following phases:

1) building schematic for the different blocks.
2) testing the blocks at the gate level for correctness in functionality.
3) generating the layout of the blocks from the schematic using an automatic router and compactor tool.
4) testing the blocks at the transistor level, for verifying the layout vs. schematic, and carry out timing analysis.
5) creating the top level design using the basic

blocks which are already built and tested.

6) testing the top-level circuit for functionality.

7) generating the layout of the top-level using a floorplanning tool.

The CAD tools by Mentor Graphics(M. G) were used in the design of the EDES circuit. The tools we used were Generated Development Tools(GDT) which supports the schematic and layout generation. In the GDT environment there is a grapics editor called Led. The Led grapics editor can be used for drawing schematics and layouts.

### 4.1 S-BOX

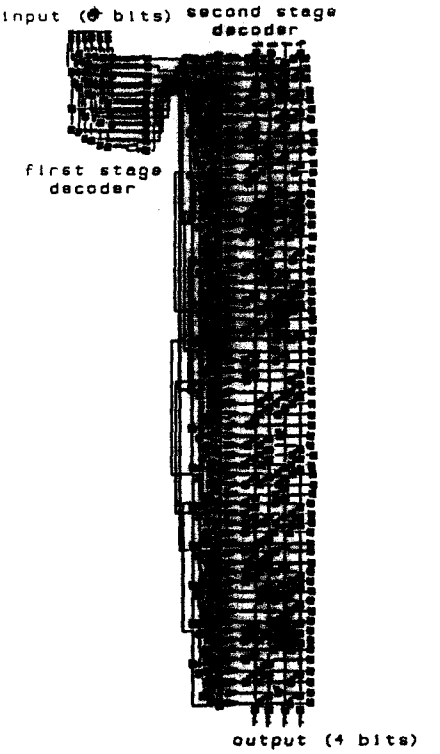
The S-BOX have been implemented as ROM's. The schematic for each S-Box was built using the standard cells from the Lx standard cells library. The list of standard cells used for the S-BOX is given in 〈Table 1〉. The ROM has a decoder at the front end which selects the word. This decoder was built as a two-stage decoder. The decoder was built seperately and stored as an individual cell. For each of the S-BOX this cell containing the decoder was added and then the memory element for the corresponding S-Box was built. The GDT schematic of the S-Box1 is in (Fig. 9). The GDT environmemt that supports the schematic and layout generation. When the design is a schematic form, it can be simulated at the gate-level for functionality.

〈Table 1〉 List of standard cells in each S-Box1

| Standard Cell | Count |
|---|---|
| inverter | 6 |
| 2-input AND gate | 12 |
| 3-input AND gate | 64 |
| p-type transistor | 4 |
| n-type transistor | 120(app) |

### 4.2 Key Generation and Encryption/Decryption Controller Circuit

The two 56-bit key generation circuits were then



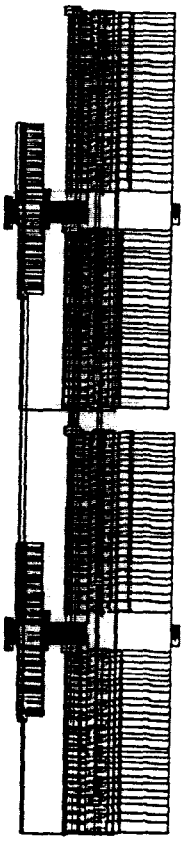

(Fig. 9) Schematic of S-Box1

instantiated in the top-level circuit of the EDES to form the complete key generation circuit. The standard cell used for the key generation circuit are listed in 〈Table 2〉. The schematic of the entire Key Generation circuit is shown in (Fig. 10).

The encryption/decryption controllers have been built as ROM's. The decoder of the encryption/decryption controllers have been built in the same way as those for the S-boxes. The standard cells used for the controllers are listed in 〈Table 3〉. The GDT schemetic of the EDES circuit is shown in (Fig. 11).

〈Table 2〉 List of standard cells in key generation circuit

| Standard cell | count |
|---|---|
| 3-to-1 multiplexer | 112 |
| 2-to-1 multiplexer | 224 |
| static d latch | 224 |



(Fig 10) Schematic of key generation circuit.

〈Table 3〉 List of standard cell in encryption and decryption controller

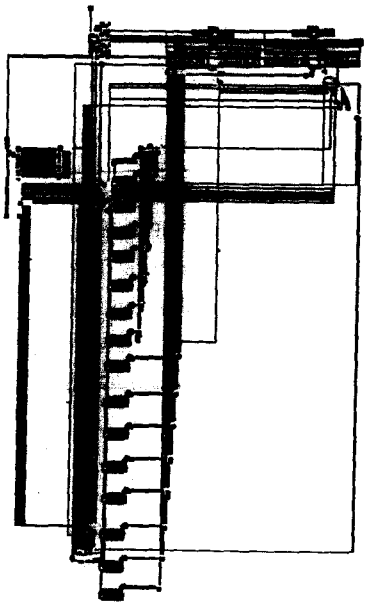
| Standard cell | count |
|---|---|
| inverter | 5 |
| 2-input AND gate | 8 |
| 3-input AND gate | 17 |
| p-type transistor | 4 |
| n-type transistor | 30 |

4.3 Implementation Summary :

The EDES algorithm has been realized in VLSI using a 1.2 micron CMOS technology with three levels of metals.

Complexity : about 40,000 Transistors.

Chip Area : 7.5 × 7.0 mm 2.



(Fig. 11) The GDT schemetic of the EDES circuit

Clock frequency : 15 Mhz.

# of I/O Pins : 32.

## 5. Conclusion

The increase in the amount of information by electronic means such as internet, wireless phones, FAX, ect. has also increased its vulnerability to unauthorizd accesses and invasion of privacy. This clearly demonstrate the need for the development of a cryptosystem to solve these problems.

The objective of the paper was to propose a hardware to implement the Extended Data Encryption Standard Algorithm in VLSI. The hardware design had a few drawbacks with respect to the layout generated by the automatic router and layout generator used in the Mentor Graphics. The circuit was built without any power considerations hence it is likely to constume a lot of power. In case this chip is to be used in cellular phones it would be worthwhile to design a low power chip for the EDES algorithm.

The functionality tests carried out on the EDES

hardware yielded correct results for 10 different sets of plaintext messages and generated ciphertext message for encryption and decryption respectively. The input data block to the encryption or decryption algorithm is 96 bits long. To exhaustively test the circuit we would need to generate $2^{96}$ test vectors, which is very time consuming. It is proposed at this stage that further exhaustive testing of the designed circuit be carried out before the layout mask is sent for fabrication.

## References


[1] S. J. Han, "The Improvement Data Encryption Standard(DES) Algorithm," Proceedings of ISSSTA'96, IEEE, pp.1167-1170, Sept. 1996.

[2] S. J. Han, "Improved-DES Crytosystem Design," Journal of Kiss, vol.24, no.1, pp.57-67, Jan. 1997.

[3] NBS, "Data Encryption Standard," FIPS pub. 46, U.S. National Bureau of Standards, Washington DC, 1997.

[4] M. E. Hellman, "DES will be totally Insecure within ten Years," IEEE Spectrum, vol.16, no.7, pp.32-39, 1979.

[5] E. Biham & A. Shamir, "Differential Cruptanalysis of the Full 16-Round DES," Advances in Cryptology-CRYPTO '92 Proceedings. Berlin: Springer-Verlag, 1993.

[6] H. H. Evertse, "Linear Structures in Block Ciphers," Advabces in Cryptology-EUROCRYPT '87, Proceedings. Berlin: Springer-Verlag, pp.249-266, 1987.

[7] G. I. Davida, D. J. Linton, C. P. Szelag & D. L. Well, "Data base security," IEEE Transactions on Software Engineering, vol.SE-4, no.6, pp.531-533, 1978.

[8] J. H. Moore, & G. J. Simmons, "Cycle Structure of the DES for Keys Having Palindromic (or Antipal-indromic) Sequences of Round Keys," IEEE Transaction on Software Engineering, vol. 13, no.6, pp.858-864, 1982.

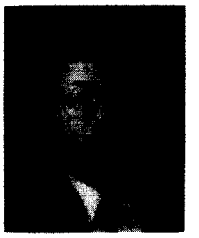
[9] Schneirer Bruce, *Applied Cryptography*, John Wiley & Sons, Inc., pp.219-296, 1988.

[10] J. B. Kam & G. I. Davida, "Structured Design of Substitution Permution Encrytion Networks," IEEE Transactions on Computer, vol.28, no.10, pp.747-753, 1989.

[11] A. F. Webster, & S. E. Tavares, "On the design of S-boxes," Proceedings of Crypto '85, 1985.

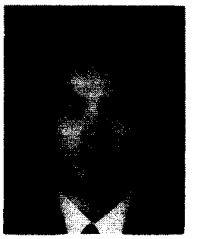
[12] J. B. Kam, & G. I. Davida, "Structured Design of Substitution Permutation Encryption Networks," IEEE Transactions on Compiters, vol.28, no.10, pp.747-753, 1989.

[13] Frank Hoornart, Jo Goubert & Yvo Desmedt, "Efficient hardware implementation of the DES," Journal of Cryptology, vol.4, no.1, pp.148-151, 1987.


**한 승 조**

1980년 조선대학교 전자공학과 학사
1982년 조선대학교 대학원 전자공학과 석사
1994년 충북대학교 대학원 전자계산학과 박사
1997년 3월~현재 조선대학교 전자·정보통신공학부 교수
1986년 6월~1987년 3월 Univ. of New Oleans 객원교수
1995년 2월~1996년 1월 Univ. of Texas 객원교수
관심분야: 통신보안, ASIC, 음성합성


**김 판 구**

1988년 조선대학교 컴퓨터공학과(공학사)
1990년 서울대학교 컴퓨터공학과(공학석사)
1994년 서울대학교 컴퓨터공학과(공학박사)
1995년~현재 조선대학교 전자계산학과 조교수 재직
관심분야: 시스템 보안, 운영체제, 정보검색, 영상처리