An LSI DIGITAL ENCRYPTION PROCESSOR (DEP)

R. C. Fairfield, A. Matusevich, and J. Plany

AT&T Bell Laboratories
25 Lindsley Drive
Morristown, N.J. 07960

ABSTRACT

This paper describes an LSI digital encryption processor (DEP) for data ciphering. The DEP combines a fast hardware implementation of the Data Encryption Standard (DES) published by the National Bureau of Standards (NBS) with a set of multiplexers and registers under the control of a user programmed sequencer. This architecture enables the user to program any of the DES modes of operation published by NBS. In addition, multiple ciphering operations and multiplexed ciphering operations using up to four different keys may be programmed and internally executed without any external hardware.

The DEP is designed as a standard microprocessor peripheral. This LSI device should reduce the current cost and simplify the process of encrypting digital data to a point where it is feasible to include a ciphering function in modems, terminals, and work stations. The ability to internally program cascaded ciphers should substantially increase the security of the DES algorithm and hence, the life of the encryption equipment.

INTRODUCTION

In January 1977 the National Bureau of Standards (NBS) adopted an IBM developed block cipher called the Data Encryption Standard (DES) [1]. Approximately four years later, in December 1980, the NBS published a follow-up document titled "DES Modes of Operation" [2] which describes four DES operating modes and some of their

G.R. Blakley and D. Chaum (Eds.): Advances in Cryptology - CRYPTO '84, LNCS 196, pp. 115-143, 1985.

characteristics. This paper describes a new LSI device called the Digital Encryption Processor (DEP) developed by AT&T Bell Laboratories and manufactured by AT&T Technologies. The DEP has been validated by the NBS as complying with the DES. Other devices have also been certified. This, however, is the first LSI device to incorporate all of the standard DES modes of operation into a single integrated circuit and provide the user with the flexibility to program unique or custom ciphering functions.

Unless provisions are made now to effectively lengthen the key space of current DES enciphering modes, the encryption equipment life may be shortened by integrated circuit technology advances. To date, the best known attack on the DES algorithm is a brute force search of the key space under a known plaintext attack. Today's fastest DES integrated circuits can perform a maximum of 250K ciphering operations/second. The operating speed may be increased by a factor of 12 by shrinking the design rules. Further performance improvements may be achieved by pipelining the DES algorithm (factor of 16). Pipelining would require sixteen sets of: (1) 64 bit L and R registers; (2) 56 bit C and D registers; (3) 32 "exclusive-or" gates ; (4) 48 "exclusive-or" gates; and (5) 8 s-boxes (2044 bits of ROM). This would increase the integrated circuit transistor count by approximately a factor of 16. Certainly, this would be a huge integrated circuit even by todays standards. Despite these speed improvements, an array of devices would still be required to search the key space in a reasonable amount of time. Therefore, each device should be capable of independent operation. This would require each integrated circuit to have an independent controller, a comparator function to flag matching ciphertext, and a read/write key counter register. Then, using an array of five-hundred devices each independently searching a different part of the key space, all possible keys could be checked in one month. This is a distressing result for a DES device or board manufacturer who would like to see a product in the field for a period of years. For this reason the DEP device may be programmed to internally perform cascaded ciphering operations using up to four different keys. A cascade of k ciphers may not be equivalent to increasing the key size (56 bits) by a factor of k since a time-memory tradeoff may be made; however, it is certainly far more work than searching the key space of a single cipher. See reference [3]. S. Even and O. Goldreich have shown that

a cascade of two DES's can be cracked in time 2**71 and space 2**41 [4]. Work increases on this order would extend the practical life of the DES for years. It may also be possible to rearrange the feedback in a cascaded cipher to prevent a meet in the middle known plaintext attack.

The DEP combines a fast hardware implementation of the DES with a sixty-four bit input shift register, a sixty-four bit output shift register, a set of multiplexers necessary to configure the operating modes, a data latch, and four sets of key and initial value registers. Control over this hardware is provided by a user programmed sequencer. This sequencer provides the flexibility necessary to program any of the four DES operating modes and to tailor the encryption function to the system requirements. Additionally, the four different key and initial value registers may be used to program multiplexed ciphering operations or to provide for enhanced security requirements by programming multiple ciphering operations using different keys.

The DEP is designed as a microprocessor peripheral and is packaged in a standard forty pin dual-in-line package. Figure 1 shows a block diagram of the device. There are two separate parallel bidirectional eight bit ports, two separate serial bidirectional data ports and a serial key port. All of these ports may be read or written asynchronously with respect to the clock input. The separate data ports are provided to increase data throughput and security by allowing separate plaintext and ciphertext buses. There are seven possible data port configurations. The serial key input port would typically be used to load a key from external circuitry, say a ROM, that the user keeps locked up when not in use. Microprocessor polled or interrupt systems may be configured, since output flags may be read from the data buses or on independent output pins. Maximum data rate for the device in any of the standard operating modes is as follows:

Input Clock (Tc)	2.5 MHz (worst case) -40 to 80 deg C	4.5 MHz (nominal)
Instruction Period	2*Tc	
Ciphering op/sec	73.5K (worst case)	132K (nominal)

All four NBS defined operating modes may be executed in a minimum of 17 instructions. If the entire DES output block (64 bits) is used for the ciphering operation, the worst case data throughput is 0.59 megabytes per second.

The internal user programmed sequencer enables the device to

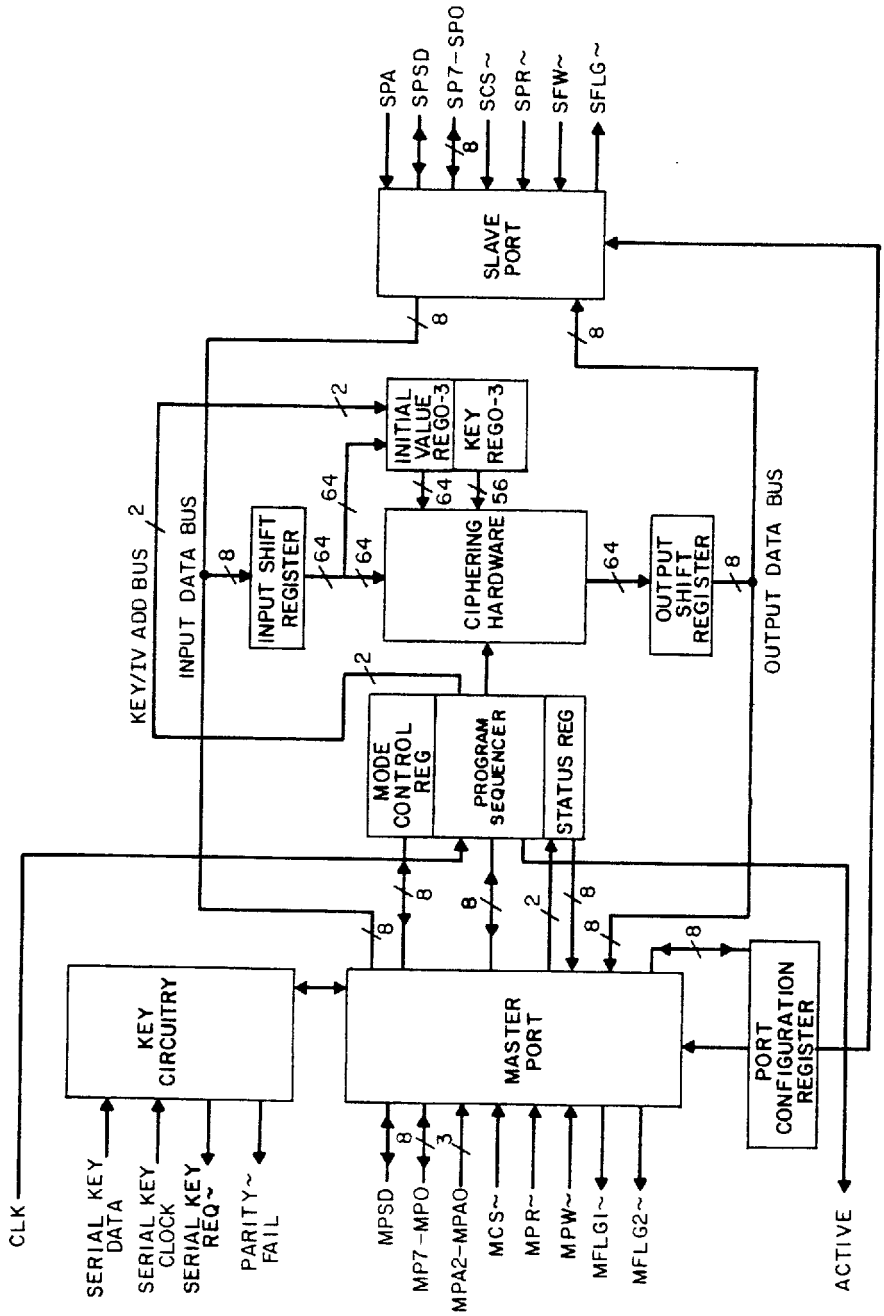


FIGURE 1
DEP BLOCK DIAGRAM

accommodate special system requirements and reduces host processor overhead. The DEP should reduce the cost and simplify the process of encrypting digital data to a point where it is possible to include a ciphering function in modems, terminals and work stations. In addition, the ability to internally program cascaded ciphers should substantially increase the security of the DES algorithm. This paper describes the DEP architecture, the micro-code instruction set, and then gives some unique applications.

ARCHITECTURE

The DEP architecture may be divided into two sections: the ciphering hardware and the user programmed sequencer.

THE CIPHERING HARDWARE

The DES specifies a cryptographic algorithm which is a nonlinear sixty-four bit block cipher using a fifty-six bit key. The components of the algorithm are simple and individually weak. They consist of permutations, combinations ("exclusive-or" sums) of the data and internal key bits, and nonlinear substitutions. These weak elements are combined and the data are encrypted in sixteen iterations through them. Sixteen 48 bit internal keys are generated by rotating and permuting the 56 bit DES input key. Although NBS has published the DES [1], no cryptographic analysis or justification for the specific elements in the algorithm has been published. The published literature does, however, provide some insight into the inner workings of the algorithm [5] and [6]. The key input to the DEP device is a 64 bit number with the least significant bit in each byte, or every eighth bit in a serial key load, a parity bit. Odd parity is checked and a flag is set if parity fails. Device operation is not inhibited by a parity failure.

Figure 2 shows a block diagram of the DEP ciphering hardware, with the DES key schedule and enciphering circuitry enclosed in dotted lines. The algorithm specified in the DES was designed to be implemented in hardware (not software). There are several permutation matrices specified in the standard and the penalty for a software implementation is an inordinate amount of time spent shuffling or permuting bits. This operation has no overhead in hardware, since the

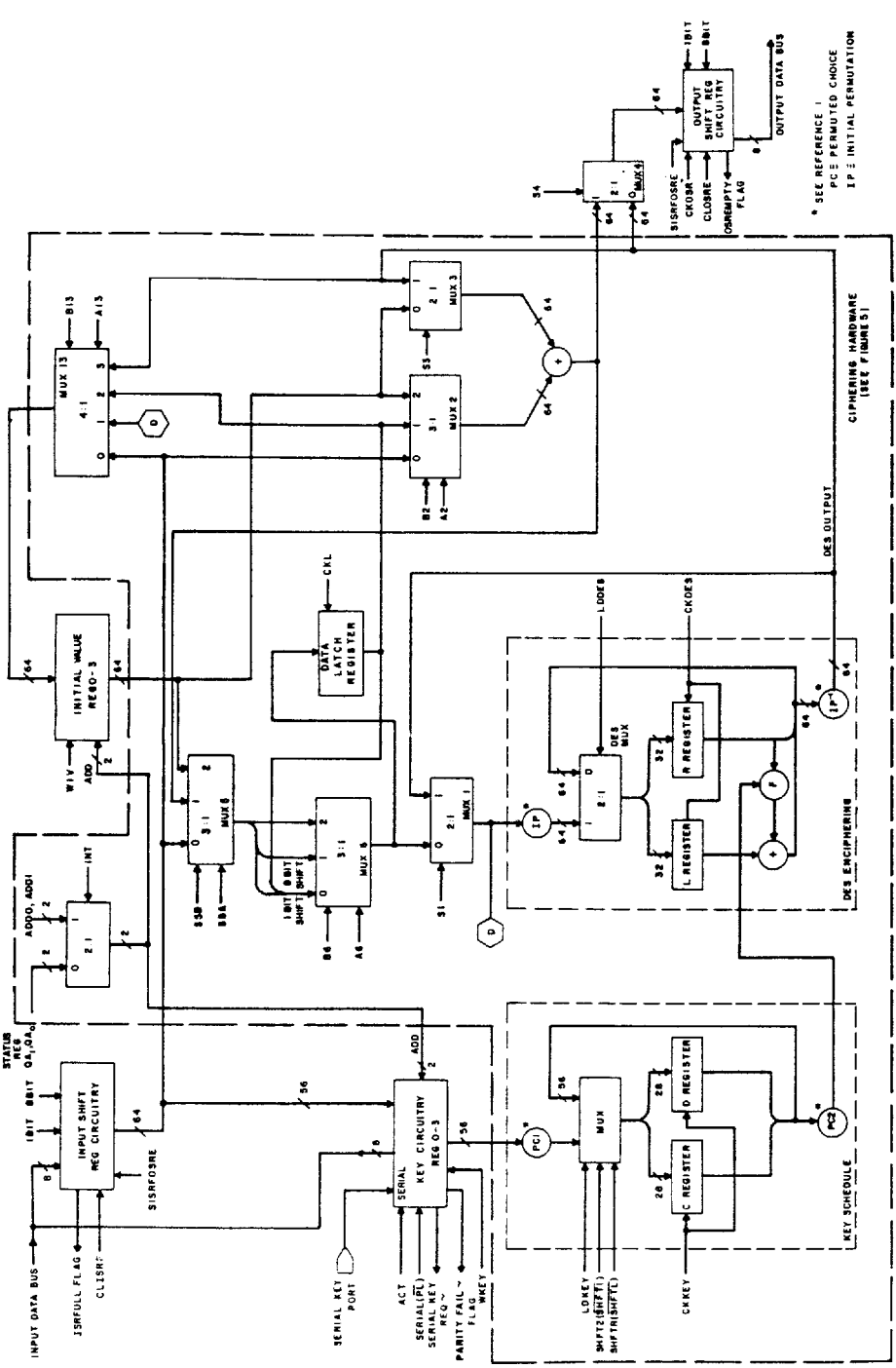


FIGURE 2 DEP CIPHERING HARDWARE AND PERIPHERAL CIRCUITRY

permutation matrix is simply a crisscross of wires. The DES section of the enciphering circuitry consists of: a 2:1 multiplexer with 64 sections; two 32 bit L and R registers; 32 "exclusive-or" gates; and a cipher function, F, of the internal key and R register. Figure 3 illustrates the F function. The eight s-boxes shown are the nonlinear algorithm elements and are implemented as eight ROMs, each consisting of 64, four bit words (six address lines and four outputs).

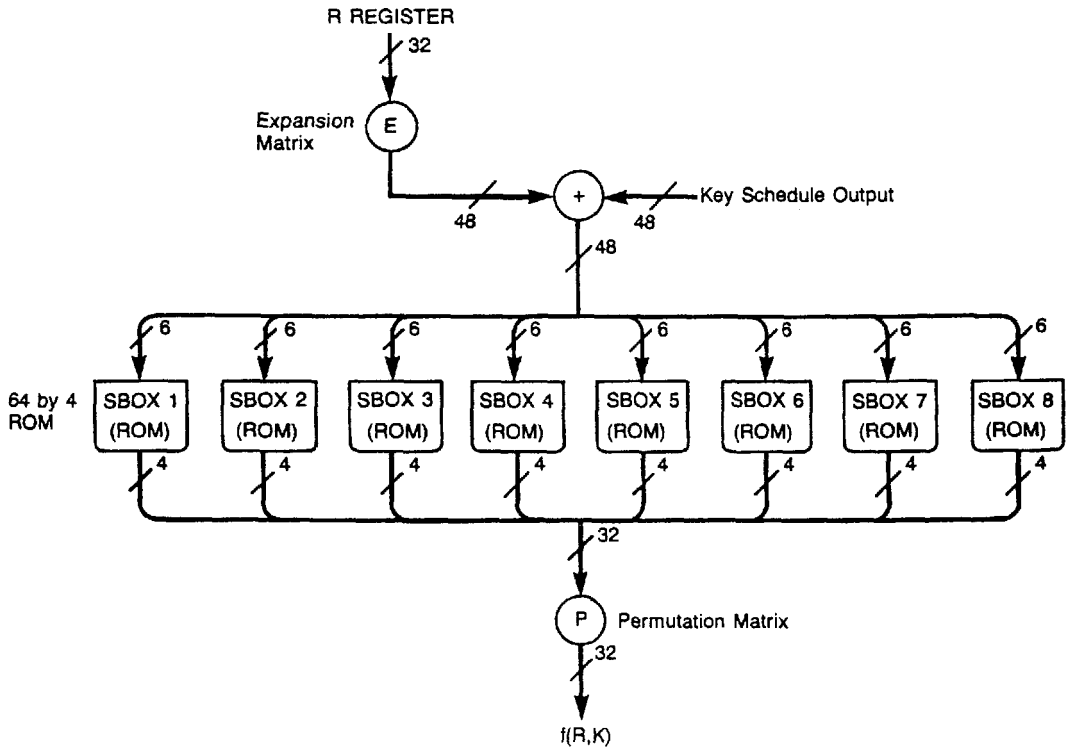


FIGURE 3. F FUNCTION

Since the DES algorithm is a block cipher, there is a one to one mapping of the input block to the output block. To a cryptographer, it is disconcerting to know that a recurring plaintext block will duplicate the earlier ciphertext block. This leaves the crypto system

vulnerable to traffic analysis and the possibility of insertion or deletion of messages by an active intruder. Hence, the NBS published the "DES modes of operation" [2]. Four modes are defined:

1. Electronic Codebook (ECB) is a straightforward implementation of the DES algorithm.

2. Cipher Block Chaining (CBC). To begin the operation, an initial value is added modulo 2 to the first plaintext block to form the DES input block. The DES output is the ciphertext. This output is fed back and added modulo 2 to the next plaintext block forming the new DES input block. CBC produces a ciphertext dependent on previous plaintext blocks.

3. K-bit Cipher Feedback (CFB). Starting with an initial value as the DES input block, K plaintext input bits are added modulo 2 to the K most significant bits in the DES output block. The result is the K-bit ciphertext which is fed back and shifted into the K least significant bits of the DES input block to form the next DES input block.

4. Output Feedback (OFB). Starting with an initial value, the DES is operated as a pseudo random bit stream generator (the DES output is fed back as the input). Ciphertext is produced by adding the plaintext to the random bit stream modulo 2.

Figure 2 shows the sets of multiplexers, the "exclusive-or" gates, and the data latch necessary to configure the DES operating modes. MUX 6 and the latch register are used to shift the input data block for the CFB mode. MUX 13 is used to select the input to initial value registers 0 through 3. The initial value registers may be used to hold temporary products in a multiple encryption operation, or to store the next DES input block for the current ciphering operation, before jumping to a different ciphering operation. The input and output shift register circuitry is clocked by the rising edge of the decoded data write and read strobes applied to the chip. When the input shift register is filled, an ISRFULL flag is set and the DEP can cipher the new data and clear the flag. When the output shift register is empty, an OSREMPTY flag is set and the DEP can reload this register and clear the flag. These two flags may be read by the user on either of the two eight bit data ports or on separate output pins. This structure allows the external read and write strobes to be independent of the DEP clock. To achieve maximum data throughput a

user would have to complete the reading and writing of data during the DEP ciphering operation.

THE USER PROGRAMMED SEQUENCER

The two eight bit bidirectional data ports, master and slave, may be thought of as plaintext and ciphertext ports, respectively. All control registers must be written through the master port. Other than data, which may be read or written for ciphering, only three flag bits of the status register may be read from the slave port. See Tables 1 and 2. Control over the ciphering hardware is provided by the user programmed sequencer. A block diagram is shown in Figure 4. The sequencer executes a 22 bit instruction every two clock cycles. Depending on the address in the program counter, these instructions may come from either a RAM or ROM program memory.

The ROM contains three programs and one subroutine. The subroutine executes the DES algorithm using whatever key is currently in the C and D registers (Key Schedule, Figure 2) to encipher whatever data is sitting at the input to the initial permutation matrix (labeled IP, DES Enciphering circuitry). There are four pairs of key and initial value registers that may be externally loaded. These registers are loaded by writing the address (0 through 3) of the key/initial value pair to an internal status register. Then the appropriate ROM program is executed. The three programs are described (ROM Code, Table 3):

1. A load initial value program waits for an eight byte number to be written to the master port. When the ISRFULL flag is set, this number is clocked into the initial value register addressed by the status register.

2. A load key program waits for an eight byte number to be written to the master port. When the ISRFULL flag is set, this number is clocked into the key register addressed by the status register. Odd parity of each byte is checked. The least significant bit in each byte is the parity bit.

3. A serial load key program waits for a sixty-four bit number to be clocked into the serial key data port using the serial key clock. When this program is executed, a hardware key request pin goes active. When the key is loaded into the input shift register, the sequencer clocks the number into the key register

MASTER PORT ADDRESS	REGISTER	READ OR WRITE	MSB CONTENT LSB							
			B7	B6	B5	B4	B3	B2	B1	B0
0	INPUT SHIFT REGISTER (ISR)	W	DI1	DI2	DI3	DI4	DI5	DI6	DI7	DI8
	OUTPUT SHIFT REGISTER (OSR)	R	DO1	DO2	DO3	DO4	DO5	DO6	DO7	DO8
1	STATUS	W	X	X	X	X	X	X	QA1	QA0
		R	PARITY FAIL	ACTIVE	OSR-EMPTY	ISR-FULL	SERIAL KEY REQ	0	QA1	QA0
2	PORT CONFIGURATION	R/W	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
3	MODE CONTROL	R/W	0	0	MC5	MC4	MC3	MC2	MC1	MC0
4	M1 (PROGRAM MEMORY)	R/W	M17	M16	M15	M14	M13	M12	M11	M10
5	M2	R/W	M27	M26	M25	M24	M23	M22	M21	M20
6	M3	R/W	0	0	M35	M34	M33	M32	M31	M30

TABLE 1. MASTER PORT REGISTERS (READ/WRITE)

SLAVE PORT ADDRESS	REGISTER	READ OR WRITE	MSB CONTENT LSB							
			B7	B6	B5	B4	B3	B2	B1	B0
0	INPUT SHIFT REGISTER (ISR)	W	DI1	DI2	DI3	DI4	DI5	DI6	DI7	DI8
	OUTPUT SHIFT REGISTER (OSR)	R	DO1	DO2	DO3	DO4	DO5	DO5	DO7	DO8
1	STATUS	R	0	ACTIVE	OSR-EMPTY	ISR-FULL	0	0	0	0

TABLE 2. SLAVE PORT REGISTERS (READ/WRITE)

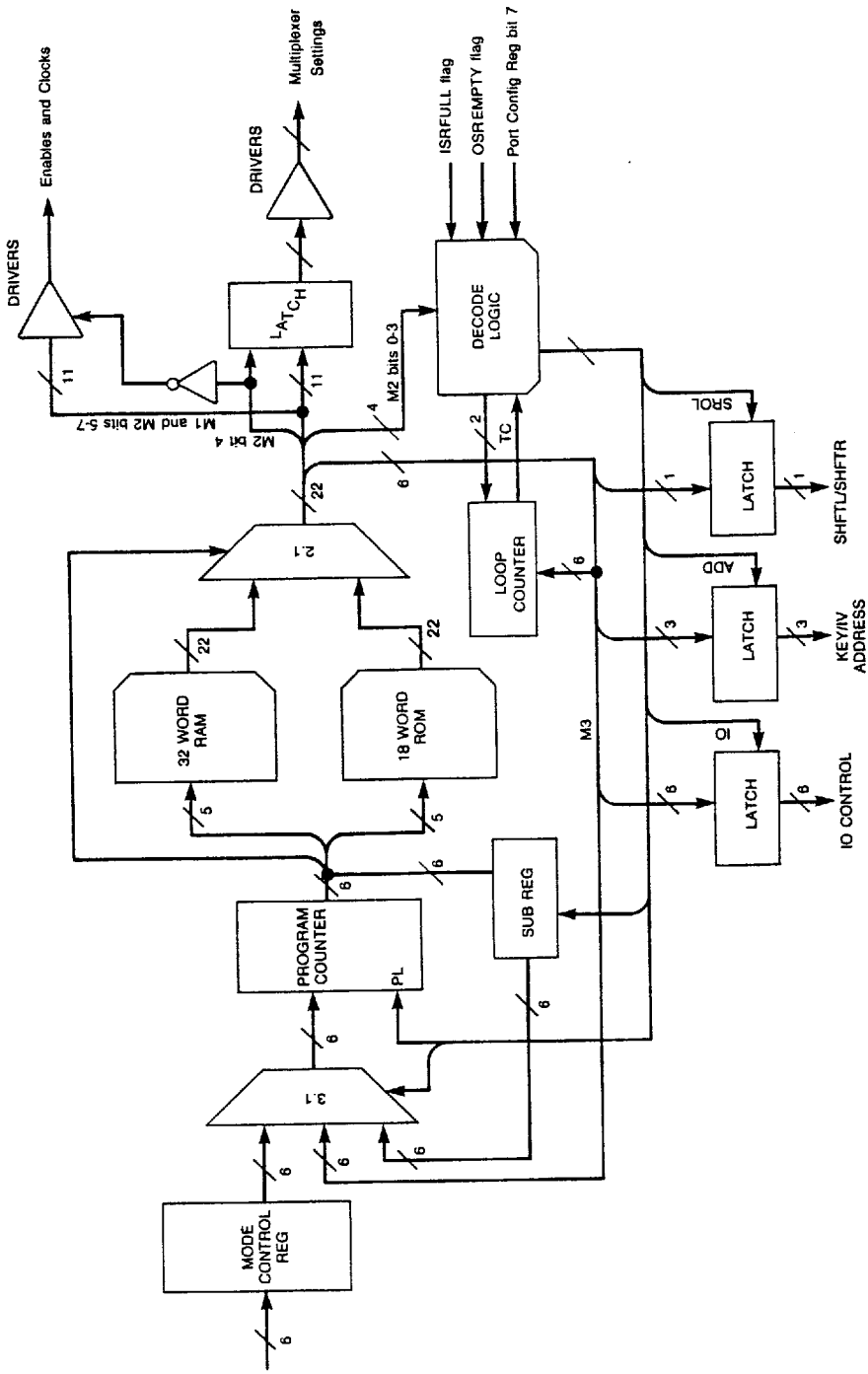


FIGURE 4. PROGRAM SEQUENCER

```
      CODE
ADDR  M1 M2 M3          ASSEMBLER MNEMONICS

  /* DES SUBROUTINE
  /*
   0 c2 1f  0       :00 LDDES CKDES CKKEY
   1 42 10  5       :01 CKDES CKKEY LLC 5
   2 52 11  2       :02 CKDES SHFT2 CKKEY ILC 02
   3 42 10  5       CKDES CKKEY LLC 5
   4 52 11  4       :03 CKDES SHFT2 CKKEY ILC 03
   5 42 13  0       CKDES CKKEY RET 0
  /*
  /* LOAD INITIAL VALUE*
  /*
   6  1  b  3       B6 IO LDMP ACT**
                     DES INPUT = ISR   OSR INPUT = DESOUT
                     IV INPUT = ISR   LATCH INPUT = ISR
   7  1 1a  0       CLISRF ADD
   8  0 15  8       :10 ISRFT? 10
   9  0 3c  0       WIV CLEAR
   a  0 14  a       :20 GTO 20
  /*
  /* PARALLEL LOAD KEY*
  /*
   b  1  b  3       B6 IO LDMP ACT**
                     DES INPUT = ISR   OSR INPUT = DESOUT
                     IV INPUT = ISR   LATCH INPUT = ISR
   c  1 1a  0       :25 CLISRF ADD
   d  0 15  d       :30 ISRFT? 30
   e  8 1c  0       WKEY CLEAR
   f  0 14  f       :40 GTO 40
  /*
  /* SERIAL LOAD KEY*
  /*
  10  1  b  7       B6 IO LDMP SERIAL ACT**
                     DES INPUT = ISR   OSR INPUT = DESOUT
                     IV INPUT = ISR   LATCH INPUT = ISR
  11  0 14  c       GTO 25
```

* These are self contained programs. They may not be called as subroutines from another program.

** B6 is an unnecessary mnemonic in this code.

PROGRAM ROM CODE

TABLE 3

addressed by the status register. The key request flag is then cleared. Odd parity of every eight bits is checked. Each eighth bit input is the parity bit.

At the end of all three of these programs the sequencer goes into an endless loop (wait state) until a new program is executed.

The RAM contains the ciphering program and must be written by the user prior to any ciphering operation. The RAM may hold up to thirty-two instructions, more than enough to program both encrypt and decrypt of any standard DES mode. The user loads the RAM through the eight bit master port. After first writing the RAM address (20H to 3fH) to the mode control register, the user writes three bytes for each 22 bit program instruction. The two most significant bits, in one of the bytes, are not used. The RAM, or the ROM (address 00H to 11H), may be read in a similar manner.

To begin ciphering or the execution of a program located in either RAM or ROM, the user writes the program memory starting address to the mode control register. Two clock cycles later this address is loaded into the program counter (Figure 4) and execution begins. Data flow through the ports and the associated assignments of the master and slave flags are controlled by the port configuration register; see Table 4. Normally this register would be written before executing a ciphering program.

PORT CONFIG	HEX CODE	OUTPUT PIN/FLAG ASSOCIATIONS
MP - - ► SP	04 OR 84	MFLG1 ~ - ISRFULL SFLG ~ - OSREMPTY
MP ◄ - - SP	11 OR 91	MFLG1 ~ - OSREMPTY SFLG ~ - ISRFULL
- - ► MP ◄ - -	01 OR 81	MFLG1 ~ - OSREMPTY MFLG2 ~ - ISRFULL
MPSD - - ► SPSD	28 OR A8	MFLG1 ~ - ISRFULL SFLG ~ - OSREMPTY
MPSP ◄ - - SPSD	62 OR E2	MFLG1 ~ - OSREMPTY SFLG ~ - ISRFULL
MP - - ► SPSD	08 OR 88	MFLG1 ~ - ISRFULL SFLG ~ - ISRFULL
MP ◄ - - SPSD	61 OR E1	MFLG ~ - OSREMPTY SFLG ~ - ISRFULL

NOTE: THE MOST SIGNIFICANT BIT IN THE HEX CODE FOR THE PORT CONFIGURATION IS AN INPUT FLAG. IT IS TESTED BY THE MICROCODE MNEMONIC LT?. THIS BIT MAY BE USED AS A GENERAL PURPOSE CONDITIONAL JUMP.

TABLE 4. PORT CONFIGURATION (MASTER PORT ADDRESS = 2)

MICRO-CODE INSTRUCTION SET

Mnemonics, corresponding to actual signal names, were defined for the program instruction set. Table 5 defines a 22 bit instruction composed of three bytes, M1, M2, and M3.

Bit 4 of M2 controls the interpretation of M1 and the three most significant bits in M2. If bit 4 of M2 is low, the multiplexer select lines are latched. In the program convention used, the presence of a mnemonic, S1 for example, indicates the control line is latched high. Conversely, the absence of a multiplexer mnemonic indicates the control line is latched low. If bit 4 of M2 is high, the specified signal is enabled only for the duration of the instruction period, two clock cycles. An enable and the associated clock signal, e.g., LDDES and CKDES, must be programmed in the same instruction since none of these signals are latched.

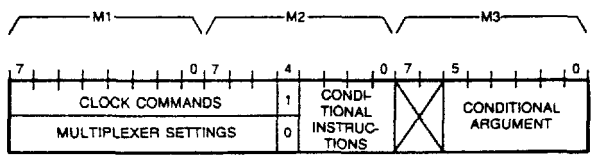
Bits 0 through 3 of M2 are decoded and select one of twelve commands. With the exception of RET and CLEAR, all of these commands use all or some of the bits in M3 as an argument. The three commands SROL, ADD, and IO latch bits of M3 until overwritten, or a subsequent CLEAR command is issued.

A C language* assembler was written to facilitate the development of ciphering programs. The output of that assembler is shown in Table 3 for the ROM code. Whenever bit 4 of M2 is set low, the ciphering multiplexers are set-up and the assembler program prints the inputs to the DES (DES INPUT), output shift register (OSR INPUT), initial value register (IV INPUT), and data latch (LATCH INPUT). This is useful in checking that the multiplexer configuration latched is correct. The six instruction DES subroutine may then be explained as follows:

1. The input to the DES initial permutation matrix is clocked into the L and R registers (Figure 2). Simultaneously, the key schedule C and D registers are clocked or shifted. The direction of the shift is dependent on the state of the SHFTR signal. SHFTR is set low to encrypt (left shift) and high to decrypt.

2. The first iteration of the DES is clocked into the L and R registers and the key schedule C and D registers are again

* C is a general purpose programming language designed for and implemented on the UNIX (registered trademark of AT&T Bell Labs) operating system.



INSTRUCTION FORMAT

CLOCK COMMANDS (M2-bit 4 = 1)

BIT	MNEMONIC	DEFINITION
M1-7	LDDES	Enables the DES enciphering multiplexer to pass the output from MUX1 when high and pass the DES output when low.
-6	CKDES	Clocks the DES L and R registers.
-5	CKL	Clocks the latch register.
-4	SHFT2	Enables the key schedule circuitry to rotate 2 positions when high, and 1 position when low.
-3	WKEY	Write the output from the ISR into the key register currently addressed.
-2	LDKEY	Enables the key schedule circuitry multiplexer to pass the key register output when high and pass the key schedule output when low.
-1	CKKEY	Clocks the key schedule C and D registers.
-0	CLISRF	Clears the ISRFULL flag and allows data to be written into ISR.
M2-7	CLOSRE	Clears the OSREMPTY flag and allows data to be read from the OSR.
-6	CKOSR	Clocks the output from MUX4 into the OSR.
-5	WIV	Writes the output from MUX13 into the initial value register currently addressed.

MULTIPLEXER SETTINGS (M2-bit 4 = 0)

BIT	MNEMONIC	DEFINITION		
M1-7	S1	Selects input line for MUX1.	S1	INPUT
			0	0
			1	1
-6	B2	Select input lines for MUX2.	B2 A2	INPUT
-5	A2		0 0	0
			0 1	1
			1 0	2
			1 1	UNKNOWN
-4	S3	Selects input line for MUX3.	S3	INPUT
			0	0
			1	1
-3	S4	Selects input line for MUX4.	S4	INPUT
			0	0
			1	1
-2	S5B	Select input lines for MUX5.	S5B S5A	INPUT
-1	S5A		0 0	0
			0 1	1
			1 0	2
			1 1	UNKNOWN
-0	B6	Select input lines for MUX6.	B6 A6	INPUT
M2-7	A6		0 0	0
			0 1	1
			1 0	2
			1 1	UNKNOWN
-6	B13	Select input lines for MUX13.	B13 A13	INPUT
-5	A13		0 0	0
			0 1	1
			1 0	2
			1 1	3

NOTE: The multiplexer settings are all latched.

TABLE 5. MICRO-CODE INSTRUCTION FORMAT (part 1 of 2)

CONDITIONAL INSTRUCTIONS

M2-BIT 3	2	1	0	MNEMONIC	DEFINITION
0	0	0	0	LLC	Loads the loop counter with the least significant nibble in M3. There is only one loop counter.
0	0	0	1	ILC	Decrements the loop counter and jumps to the address in M3 if the loop counter is zero.
0	0	1	0	SUB	The current program instruction address is incremented and stored before the program jumps to the address specified in M3. Only one level of subroutine call is allowed.
0	0	1	1	RET	The program jumps to the address stored when the preceding SUB command was executed.
0	1	0	0	GTO	The program jumps to the address in M3.
0	1	0	1	ISRFT?	If the ISR is not full, the program jumps to the address specified in M3.
0	1	1	0	OSRET?	If the OSR is not empty, the program jumps to the address specified in M3.
0	1	1	1	ISRFOSRET?	If the ISR is full and the OSR is empty, then the program jumps to the address specified in M3.
1	0	0	0	LT?	If bit 7 of the port configuration register is set low, the program jumps to the address specified in M3. This bit may be used to control the order in which the key schedule is invoked.
1	0	0	1	SROL	BIT: M3-0 = 1 — MNEMONIC: SHFTR — DEFINITION: Latches a right key schedule rotation. BIT: M3-0 = 0 — MNEMONIC: SHFTL — DEFINITION: Latches a left key schedule rotation.
1	0	1	0	ADD	BIT: M3-0 — MNEMONIC: INT — DEFINITION: A high latches the internal key/IV address bus. A low latches the external bus (Status Register QA1, QA0) BIT: M3-1, M3-2 — MNEMONIC: ADD0, ADD1 — DEFINITION: Internal Key/IV address bus.
1	0	1	1	IO	BIT: M3-0 — MNEMONIC: ACT — DEFINITION: A high latches the ACTIVE flag. BIT: M3-1 — MNEMONIC: LDMP — DEFINITION: A high latches the input circuitry to receive data from the master port. Overrides the port configuration setting. BIT: M3-2 — MNEMONIC: SERIAL — DEFINITION: A high latches the key circuitry for a serial key input. BIT: M3-3 — MNEMONIC: 1BIT* — DEFINITION: A high latches the I/O circuitry to write/read a single bit. If the parallel ports are programmed only the most significant bit in the byte is used. BIT: M3-4 — MNEMONIC: 8BIT* — DEFINITION: A high latches the I/O circuitry to write/read 8 serial bits or 1 parallel byte. BIT: M3-5 — MNEMONIC: SISRFOSRE — DEFINITION: A high sets both ISRFULL and OSREMPTY flags active.
1	1	0	0	CLEAR	A high sets all bits in the latches controlled by SROL, ADD, and IO low.

ADD1	ADD0	KEY/IV ADDRESS
0	0	0
0	1	1
1	0	2
1	1	3

*NOTE: If both of these bits are low, the I/O circuitry is set to write/read 64 serial bits or 8 parallel bytes. The condition with both bits being high is undefined.

TABLE 5. MICRO-CODE INSTRUCTION FORMAT (part 2 of 2)

shifted one position. A five is loaded into the loop counter.

3. This statement is executed six times as the next six DES iterations are clocked into the L and R registers. Simultaneously, the key is shifted two positions six times.

4. The eighth iteration of the DES is clocked into the L and R registers and the key schedule C and D registers are again shifted one position. A five is loaded into the loop counter.

5. This statement is executed six times as the next six DES iterations are clocked into the L and R registers. Simultaneously, the key is shifted two positions six times.

6. The fifteenth iteration of the DES is clocked into the L and R registers and the key schedule C and D registers are again shifted one position.

At this point the output of the DES enciphering circuitry, the inverse initial permutation matrix (IP-1), will have the sixteenth DES iteration or the output block.

A sample of the RAM micro-code for the standard ECB and CBC operating modes is given in Table 6. The code for the remaining standard operating modes is available and documented.

APPLICATIONS

The following applications illustrate the unique capabilities of the DEP. In order to perform similar operations with available integrated DES devices, considerable processor overhead or multiple devices might be required.

TWO WAY ENCRYPTION APPLICATION

The first application describes a two way encryption system using separate receive and transmit keys. A drop-in box between a terminal (or computer) and a modem was built. Clearly, this system requires a character oriented protocol. The eight bit cipher feedback mode was used. In a typical terminal to computer connection, the number of characters transmitted and received are unequal. The ciphering operation desired is shown in Figure 5.

To transmit an encrypted character the number in initial value

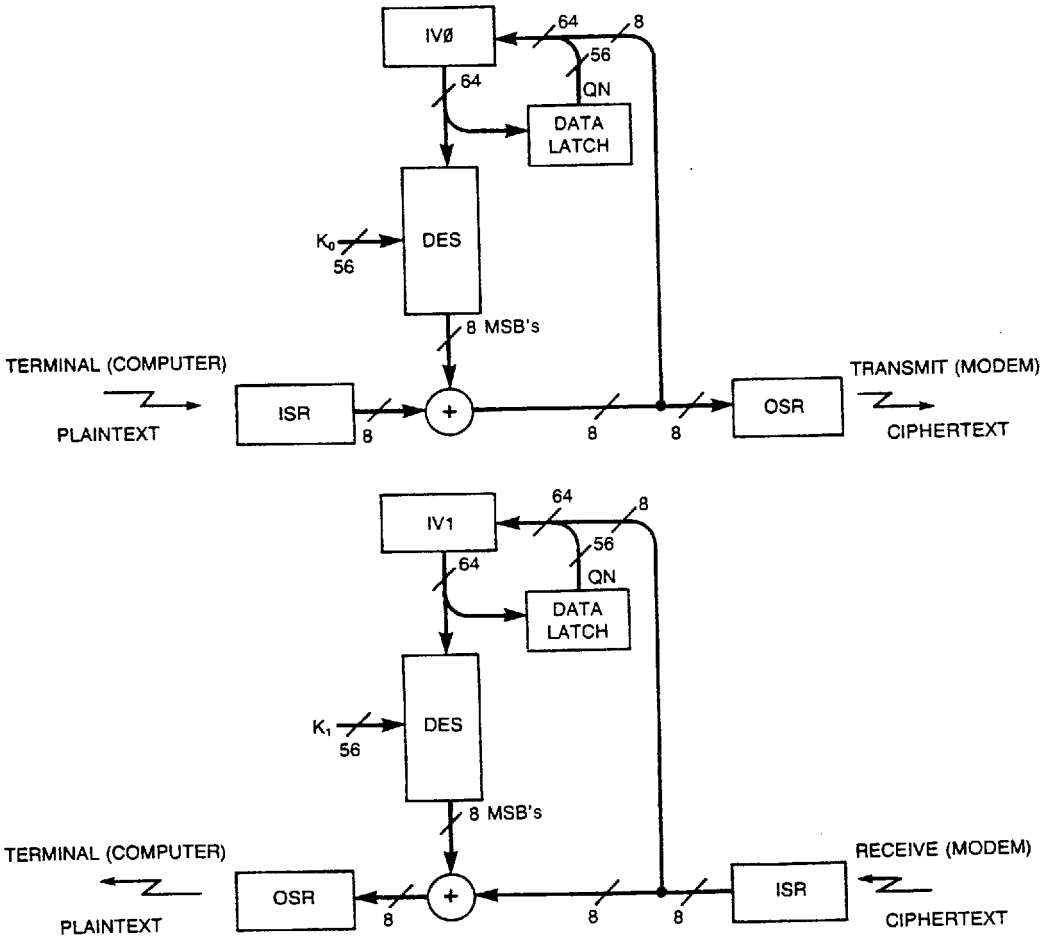
```
      CODE
ADDR  M1 M2 M3       ASSEMBLER MNEMONICS

  /* ECB ENCRYPT OR DECRYPT
  /*
  20  1  c  0      B6 CLEAR
                     DES INPUT = ISR    OSR INPUT = DESOUT
                     IV INPUT = ISR    LATCH INPUT = ISR
  21  7 18 23      LDKEY CKKEY CLISRF LT? 100
  22  2 19  1      CKKEY SROL SHFTR
  23  0 15 23      :100 ISRFT? 100
  24 c3 12  1      CLISRF LDDES CKDES CKKEY SUB 01
  25  0 17 28      ISRFOSRET? 120
  26  0 16 26      :110 OSRET? 110
  27  0 d4 23      CLOSRE CKOSR GTO 100
  28 c3 d2  1      :120 CLISRF CLOSRE CKOSR LDDES CKDES CKKEY SUB 01
  29  0 17 28      :130 ISRFOSRET? 120
  2a  0 14 26      GTO 110
  /*
  /* CBC ENCRYPT
  /*
  2b  3  c  0      S5A B6 CLEAR
                     DES INPUT = ISR^IV    OSR INPUT = DESOUT
                     IV INPUT = ISR    LATCH INPUT = ISR^IV
  2c  7 18 2e      LDKEY CKKEY CLISRF LT? 200
  2d  2 19  1      CKKEY SROL SHFTR
  2e  0 15 2e      :200 ISRFT? 200
  2f c3 12  1      CLISRF LDDES CKDES CKKEY SUB 01
  30 13  4 29      :210 S3 S5A B6 GTO 130
                     DES INPUT = ISR^DESOUT    OSR INPUT = DESOUT
                     IV INPUT = ISR    LATCH INPUT = ISR^DESOUT
  /*
  /* CBC DECRYPT
  /*
  31  7 1c  0      LDKEY CKKEY CLISRF CLEAR
  32 59 48 34      B2 S3 S4 B6 B13 LT? 250
                     DES INPUT = ISR    OSR INPUT = IV^DESOUT
                     IV INPUT = Qn    LATCH INPUT = ISR
  33  2 19  1      CKKEY SROL SHFTR
  34  0 15 34      :250 ISRFT? 250
  35 e3 12  1      CLISRF CKL LDDES CKDES CKKEY SUB 01
  36  0 17 39      ISRFOSRET? 230
  37  0 16 37      :220 OSRET? 220
  38  0 f4 34      CLOSRE CKOSR WIV GTO 250
  39 e3 f2  1      :230 CLISRF CKL WIV CLOSRE CKOSR LDDES CKDES CKKEY SUB 01
  3a  0 17 39      ISRFOSRET? 230
  3b  0 14 37      GTO 220

  ASSEMBLER OUTPUT CODE FORMAT
  Example:     31  7 1c  0
               four hex bytes
               31  Memory address to be loaded into the MODE CONTROL REGISTE
               07  M1 byte
               1c  M2 byte
               00  M3 byte
  NOTE:  User programs must be written between hex addresses 20 and
         3f hex, inclusive.
```

RAM CODE FOR ECB AND CBC OPERATING MODES

TABLE 6



TWO WAY ENCRYPTION

FIGURE 5.

register 0 is encrypted using key register 0. As this number is being clocked into the DES enciphering hardware, it is also clocked into the data latch. When a plaintext character is input, it is added modulo 2 to the eight most significant bits in the DES output block, DESOUT ^ ISR. (The symbol ^ is used to define the "exclusive-or" operator.) This byte of ciphertext output is clocked into the output shift register for transmission by the modem. It is also clocked into initial value register 0 as the least significant byte. The seven other bytes are simply the previous initial value shifted one byte to the left. The most significant byte of the previous initial value is discarded.

The receive operation is nearly identical. The number in initial value register 1 is encrypted using key register 1. As this number is being clocked into the DES enciphering hardware it is also clocked into the data latch. When a ciphertext character is input, it is added modulo 2 to the eight most significant bits in the DES output block. This byte of plaintext output is clocked into the output shift register for reception by the local terminal (computer). The ciphertext in the input shift register is clocked into initial value register 1 as the least significant byte. The seven other bytes are simply the previous initial value shifted one byte to the left. The most significant byte of the previous initial value is discarded.

There are two differences, then, between transmit and receive. One difference is the feedback to the initial value register. During transmit, ISR ^ DESOUT is fed back. During receive, only ISR is fed back. The second difference is that key/initial value register pair 0 is used in transmit and pair 1 is used in receive.

Code for this ciphering mode is shown in Table 7. The statement "DES INPUT = Qn<<8 || ISR^DESOUT", taken from Table 7, should be read as follows: the input to the DES enciphering block equals the data latch output (Qn) shifted eight bits to the left and concatenated with the eight most significant bits in the "exclusive-or" sum of the input shift register and the DES enciphering block output. After loading the RAM program memory with the hex data in Table 7, the program start address (2bH) is written to the mode control register and execution begins. The program will remain in a loop (2cH to 2fH) until the input shift register is filled. Depending on the most significant bit in the port configuration register, the DEP will either encrypt (transmit) using key/initial value register pair 0 or decrypt

```
     CODE
ADDR  M1 M2 M3          ASSEMBLER MNEMONICS

  /* ECB ENCRYPT OR DECRYPT
  /*
  20   1   c   0      B6 CLEAR
                       DES INPUT = ISR    OSR INPUT = DESOUT
                       IV INPUT = ISR    LATCH INPUT = ISR
  21   7 18 23       LDKEY CKKEY CLISRF LT? 100
  22   2 19  1       CKKEY SROL SHFTR
  23   0 15 23       :100 ISRFT? 100
  24  c3 12  1       CLISRF LDDES CKDES CKKEY SUB 01
  25   0 17 28       ISRFOSRET? 120
  26   0 16 26       :110 OSRET? 110
  27   0 d4 23       CLOSRE CKOSR GTO 100
  28  c3 d2  1       :120 CLISRF CLOSRE CKOSR LDDES CKDES CKKEY SUB 01
  29   0 17 28       :130 ISRFOSRET? 120
  2a   0 14 26       GTO 110
  /* 8 BIT CFB ENCRYPT INTERNAL KEY-IV 0
  /* OR DECRYPT INTERNAL KEY-IV 1
  /* MIN OF 26 INST IN DECRYPT
  /*
  2b   1 1b 10       CLISRF IO 8BIT
  2c  1d 2a  1       :101 S3 S4 S5B B6 A13 ADD INT
                       DES INPUT = IV    OSR INPUT = ISR^DESOUT
                       IV INPUT = IV    LATCH INPUT = IV
  2d   0 18 2f       LT? 102
  2e   0 1a  3       ADD INT ADD0
  2f   0 15 2c       :102 ISRFT? 101
  30  26 12  0       CKL LDKEY CKKEY SUB 00
  31   0 16 31       :103 OSRET? 103
  32   0 df  0       CKOSR CLOSRE
  33  1a a8 35       S3 S4 S5A A6 A13 LT? 104
                       DES INPUT = Qn<<8 || ISR^DESOUT    OSR INPUT = ISR^DESOUT
                       IV INPUT = Qn<<8 || ISR^DESOUT    LATCH INPUT = Qn<<8 || ISR^DESOUT
  34  18 af  0       S3 S4 A6 A13
                       DES INPUT = Qn<<8 || ISR    OSR INPUT = ISR^DESOUT
                       IV INPUT = Qn<<8 || ISR    LATCH INPUT = Qn<<8 || ISR
  35   1 34 2c       :104 CLISRF WIV GTO 101
```

RAM PROGRAM CODE FOR THE TWO WAY ENCRYPTION SYSTEM

TABLE 7

(receive) using key/initial value register pair 1. The mnemonic LT? is used to test the most significant port bit. A low is used for transmit (jump condition) and a high for receive (next instruction). The only timing requirement on the input to the DEP, when changing from transmit to receive, is that the data byte written be delayed from the port register write by three DEP program instructions. This guarantees the LT? instruction (2dH) will be executed after the port register write and before data ciphering. With a 4 Mhz DEP clock, this is 1.5 microseconds. After the data byte is written, the DEP program sequencer will detect an input shift register full condition and cipher the data. If the previous output data has been read, the new cipher byte will be written to the output shift register; the next initial value will be stored; and the sequencer will again cycle waiting for the input shift register to be filled. If the previous output data has not been read, the sequencer will wait (31H) until the output shift register is emptied. It will take at most twenty-four instructions from the time an input byte is written until the cipher text is available to be read. For a 4 Mhz clock, this is twelve microseconds.

In order for two stations to communicate properly, if k0 and k1 are input to key registers 0 and 1 (respectively) of a DEP device at station one, then k0 and k1 must be input to key registers 1 and 0 (respectively) of the DEP device at station two. The two stations need not have the same initial value, since a station will synchronize after eight characters have been received. This is a property of the eight bit CFB mode. Therefore, to begin a session the two stations only have to establish session keys. The protocol shown in Figure 6 was used to exchange session keys. This protocol does not require either station to be a master or slave; both stations perform exactly the same operations. A master key is input to key register 2. A random number loaded into key register 0 is encrypted in the ECB mode under the master key. This ciphertext is then transmitted. The received ciphertext is decrypted and loaded into key register 1. After these three operations, the session key exchange is complete and two way communications may begin.

Before this system could become a viable encryption product some additional work should be done. The error rates over the public telephone network combined with the eight byte error extension property of the CFB mode results in an unacceptable error rate in the

decoded plaintext. The OFB mode does not have the error extension property associated with CFB. A single transmission bit error results in a single plaintext bit error, however, telephone line noise frequently generates characters never legitimately transmitted. In the OFB mode or in any other key stream mode this would result in a loss of synchronization. This condition would have to be detected and initial values re-established, very messy. Since ASCII is a seven bit code, programming the DEP for 7 bit CFB, appending parity, buffering several bytes, and retransmitting bytes in the event of a parity failure would substantially reduce the error rate. In the current system a single ECB encryption of the session key is performed. It is suggested that a double or even a triple encryption of the session key be done since master keys would probably be changed infrequently. Some check should be made for transmission errors in exchanging session keys. If the wrong session key is used, nothing will be decoded in one direction.

TRIPLE ENCRYPTION APPLICATION

In the second application, the DEP is programmed for a triple encryption. Such a program might be used to increase security in applications involving very sensitive or valuable data. The use of multiple keys to encrypt the data effectively increases the key space an intruder must search to decode the ciphertext. Three separate cipher block chaining (CBC) operations are performed on a single DES input block. Three different keys and initial values, register pairs 0, 1 and 2, are used for the ciphering. The data latch is needed in the decrypt operation to hold intermediate products. Figure 7 shows the ciphering mode and Table 8 lists the code.

The first instruction in the CBC encrypt code is to clear the input shift register so the user may begin loading data. The multiplexers are then setup with the input to the DES enciphering circuitry equal to the "exclusive-or" sum of the input shift register and initial value register. In this same instruction, the address of key/initial value register pair 0 is latched. Nothing further happens until the input shift register is filled. When that occurs: the first ciphering operation is performed; the input shift register is cleared so the second block of data may be entered; the DES output is written to initial value register 0; and register pair 1 is addressed. Next,

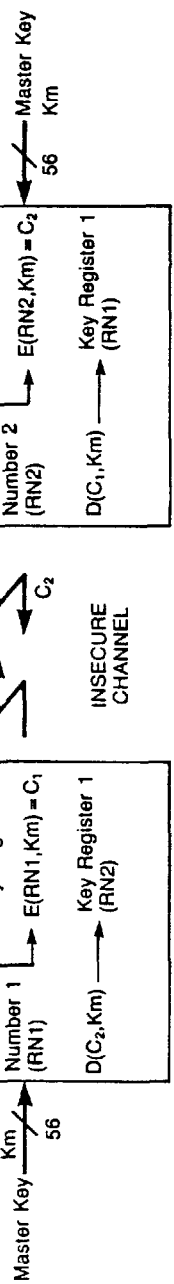
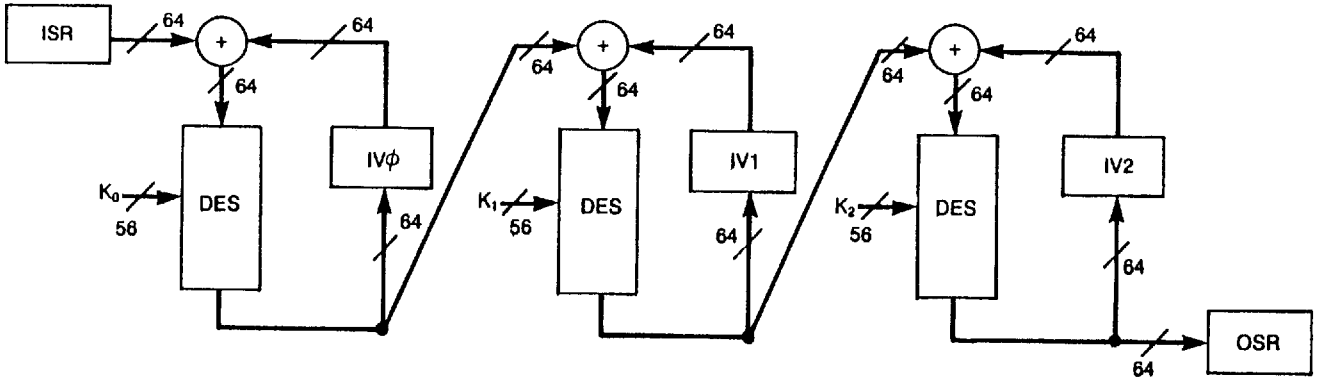


FIGURE 6. SYMMETRICAL PROTOCOL FOR SESSION KEY EXCHANGE

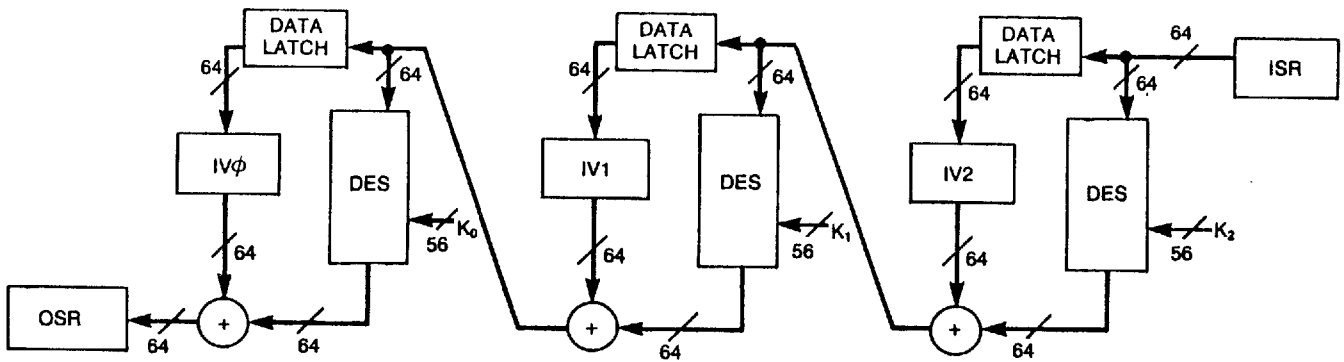
the multiplexers are reset so the input to the DES comes from the "exclusive-or" sum of the current DES output and initial value register 1. The second key (key register 1) and the new DES block are clocked into the DES circuitry. The key is shifted one position to the left and the DES subroutine is called. This completes the second ciphering iteration. The DES output is written to initial value register 1, and register pair 2 is addressed. The last ciphering operation is performed. The sequencer waits until the output shift register flag is set before clocking that register, clearing the output shift register empty flag, and jumping back to the second program instruction (21H).

The decrypt is similar to the encrypt operation with certain important exceptions. The keys and initial value register pairs must be invoked in reverse order. Similarly, the DES key schedule must be reversed. Hence, the instruction CKKEY SROL SHFTR (30H, 36H and 3aH) is required. This sets the key schedule circuitry for a right instead of a left shift. A third difference is that the data latch holds the new initial value while the current one is being used. Consequently, the decryption code requires three more statements than the encryption code. In decryption, sixty-one program instructions are executed, provided there is no waiting for the input shift register to be loaded or the output shift register to be emptied. With a 4 Mhz clock, 32.8K ciphering operations per second could be performed. This triple encryption takes 3.6 times as long as a single encryption (seventeen program instructions), so the additional overhead is only 20%.

There are many ways to implement cascaded ciphers and to feed back data blocks. The one just described suffers from error propagation. A single error in the input block to the decryption chain results in a 50% error rate in the current and the next two output blocks as well as a single bit error in the fourth block. At CRYPTO 84 Adi Shamir suggested two possible configurations for cascaded ciphers with no additional error propagation and no "meet in the middle" known plaintext attacks. These are shown in Figure 8. Under a known plaintext attack, if the initial value is kept secret, the DES input remains unknown for these configurations. The DEP may be easily programmed for either of these modes.



CBC ENCRYPT — 3 KEYS AND INITIAL VALUES



CBC DECRYPT — 3 KEYS AND INITIAL VALUES

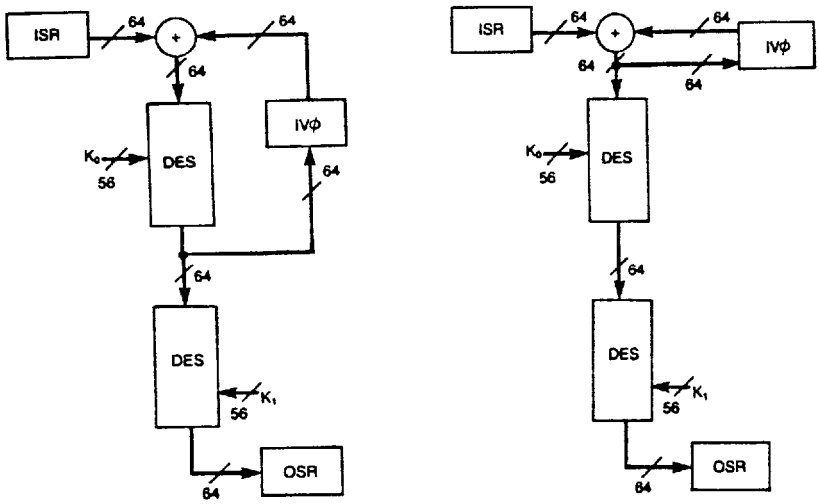
FIGURE 7.

```
      CODE
ADDR  M1 M2 M3          ASSEMBLER MNEMONICS

  /* CBC ENCRYPT 3 KEYS
  /*
  20  1 1c  0      CLISRF CLEAR
  21  3 6a  1      :50 S5A B6 B13 A13 ADD INT
                     DES INPUT = ISR^IV   OSR INPUT = DESOUT
                     IV INPUT = DESOUT   LATCH INPUT = ISR^IV
  22  0 15 22      :60 ISRFT? 60
  23  7 12  0      CLISRF LDKEY CKKEY SUB 00
  24  0 3a  3      WIV ADD INT ADD0
  25 53 6f  0      S3 B2 S5A B6 B13 A13
                     DES INPUT = IV^DESOUT   OSR INPUT = DESOUT
                     IV INPUT = DESOUT   LATCH INPUT = IV^DESOUT
  26 c6 1f  0      LDKEY CKKEY LDDES CKDES
  27  2 12  1      CKKEY SUB 01
  28  0 3a  5      WIV ADD INT ADD1
  29 c6 1f  0      LDKEY CKKEY LDDES CKDES
  2a  2 12  1      CKKEY SUB 01
  2b  0 16 2b      :70 OSRET? 70
  2c  0 f4 21      WIV CLOSRE CKOSR GTO 50
  /*
  /* CBC DECRYPT 3 KEYS
  /*
  2d  1 1c  0      CLISRF CLEAR
  2e 59 4a  5      :160 B2 S3 S4 B6 B13 ADD INT ADD1
                     DES INPUT = ISR   OSR INPUT = IV^DESOUT
                     IV INPUT = Qn   LATCH INPUT = ISR
  2f  6 19  0      LDKEY CKKEY SROL SHFTL
  30  2 19  1      CKKEY SROL SHFTR
  31  0 15 31      :170 ISRFT? 170
  32 e3 12  1      CLISRF CKL LDDES CKDES CKKEY SUB 01
  33 5b 4f  0      B2 S3 S4 S5A B6 B13
                     DES INPUT = IV^DESOUT   OSR INPUT = IV^DESOUT
                     IV INPUT = Qn   LATCH INPUT = IV^DESOUT
  34 e0 3a  3      CKL LDDES CKDES WIV ADD INT ADD0
  35  6 19  0      LDKEY CKKEY SROL SHFTL
  36  2 19  1      CKKEY SROL SHFTR
  37  2 12  1      CKKEY SUB 01
  38 e0 3a  1      CKL LDDES CKDES WIV ADD INT
  39  6 19  0      LDKEY CKKEY SROL SHFTL
  3a  2 19  1      CKKEY SROL SHFTR
  3b  2 12  1      CKKEY SUB 01
  3c  0 16 3c      :180 OSRET? 180
  3d  0 f4 2e      WIV CLOSRE CKOSR GTO 160
```

RAM PROGRAM CODE FOR TRIPLE ENCRYPTION

TABLE 8



TWO CASCADE CIPHERS (SUGGESTED BY ADI SHAMIR)

FIGURE 8

CONCLUSIONS

A user programmed Digital Encryption Processor based on the National Bureau of Standards DES algorithm has been described. The DEP has been certified by the NBS as complying with the DES. All four of the NBS defined operating modes may be programmed. Multiple (cascaded) or multiplexed ciphering operations may be programmed, eliminating the need for more than one encryption device in some

applications. The internal program sequencer allows the user to tailor the ciphering function for the specific system application. These features place the DEP beyond existing commercial devices. In order to extend the life of the DES, we would like to see more secure modes developed and analyzed. The DEP may be programmed to perform cascaded ciphering using all four key registers. The data throughput rate of 0.59 megabytes per second, for the standard modes under worst case conditions, is comparable with the fastest commercial part now available. For some of the unique modes, the data rate will be much faster since there is no host processor overhead.

The proliferation of smart terminals and computers is leading to distributed networks with access to large data bases. These networks, along with the booming cable television market and satellite communications networks, are prime candidates for low cost secure encryption.

REFERENCES

[1] Federal Information Processing Standards Publication 46, "Data Encryption Standard," January 15,1977, published by the National Bureau of Standards.

[2] Federal Information Processing Standards Publication 81, "DES Modes of Operation," December 2, 1980, published by the National Bureau of Standards.

[3] W. Diffie and M. E. Hellman, "Exhaustive Cryptanalysis of the NBS Data Encryption Standard," Computer, June 1977.

[4] S. Even and O. Goldreich, "On the Power of Cascade Ciphers," Advances in Cryptology, Proceedings of Crypto 83.

[5] Whitfield Diffie and Martin E. Hellman, "Privacy and Authentication to Cryptography", Proceeding of the IEEE, Vol. 67, No. 3, March 1979.

[6] Alan G. Konheim, "Cryptography: A Primer", John Wiley and Sons, INC., 1981, chapter 6.