Implementation of the Data Encryption Standard Using the TMS32010

APPLICATION REPORT: SPRA130

Author: Panos Papamichalis and Jay Reimer
Digital Signal Processing – Semiconductor Group

Digital Signal Processing Solutions
1989

TEXAS INSTRUMENTS

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain application using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

TRADEMARKS

TI is a trademark of Texas Instruments Incorporated.

Other brands and names are the property of their respective owners.

CONTACT INFORMATION

US TMS320 HOTLINE	(281) 274-2320
US TMS320 FAX	(281) 274-2324
US TMS320 BBS	(281) 274-2323
US TMS320 email	dsph@ti.com

Implementation of the Data Encryption Standard Using the TMS32010

Abstract

This report examines the implementation of a data encryption method using the TMS32010. The encryption algorithm chosen for the report is the Data Encryption Standard (DES). This report details the DES algorithm and its TMS32010 coding. Processor resource requirements are also provided for applying the DES to speech coding at different bit rates. For a 2.4-kbit/s LPC vocoder application, the DES only requires six percent of the TMS32010 CPU loading. The encryption scheme operates on a stream of bits that represent text, computer files, speech, or any other entity in binary form. The result is directly applicable to the design of any secure data/voice communications.

Product Support on the World Wide Web

Our World Wide Web site at www.ti.com contains the most up to date product information, revisions, and additions. Users registering with TI&ME can build custom information pages and receive new product updates automatically via email.

INTRODUCTION

The programmability of the TMS320 family of digital signal processors makes possible the implementation of different signal processing algorithms on the same device, rather than on several different custom chips. As an example, this application report describes the implementation of a data encryption method on the TMS32010. The encryption scheme operates on a stream of bits that represents text, computer files, or anything else presented in binary form. In particular, the encryption method is considered in conjunction with speech coding to achieve secure communication over voice channels. Another example consists of the encryption of data supplied to a transmitting modem and decryption of the corresponding data stream coming out of the receiving modem. The algorithm chosen for the encryption is the Data Encryption Standard (DES) as established by the National Bureau of Standards.[1]

The following paragraph has been taken from reference [1] and is brought to the attention of the readers of this report. For more information, please contact the national Bureau of Standards.

> "EXPORT CONTROL: Cryptographic devices and technical data regarding them are subject to Federal Government export controls as specified in Title 22, Code of Federal Regulations, Parts 121 through 128. Cryptographic devices implementing [the DES] and technical data regarding them must comply with these Federal regulations."

THE DATA ENCRYPTION STANDARD (DES)

Encryption/Decryption Algorithm

The Data Encryption Standard (DES), the algorithm of data encryption selected by the United States federal government, is described in detail in the NBS publications.[1] The description is summarized here in order to associate it with the implementation on the TMS32010.

Figure 1 shows a flowchart of the enciphering portion of the algorithm. The stream of bits to be encrypted (the plaintext) is segmented into blocks of 64 bits. Each 64-bit block is submitted to an initial permutation, and then is split into two 32-bit blocks (a left one L_0 and a right one R_0), which are input to the transformation section. The transformation section consists of 16 stages, where the data is scrambled by the use of the encryption key. At each stage i, the inputs are the left block L_{i-1} and right block R_{i-1} of the previous stage, and the outputs are the left block L_i and right block R_i of this stage. The outputs L_i and R_i of each stage are computed from L_{i-1}, R_{i-1}, and a subkey K_i that is generated from the encryption key. Note that the 16th stage is different from the other stages. The output of the last stage is subjected to a permutation, which is the inverse of the initial permutation. The resulting 64 bits are the ciphertext.

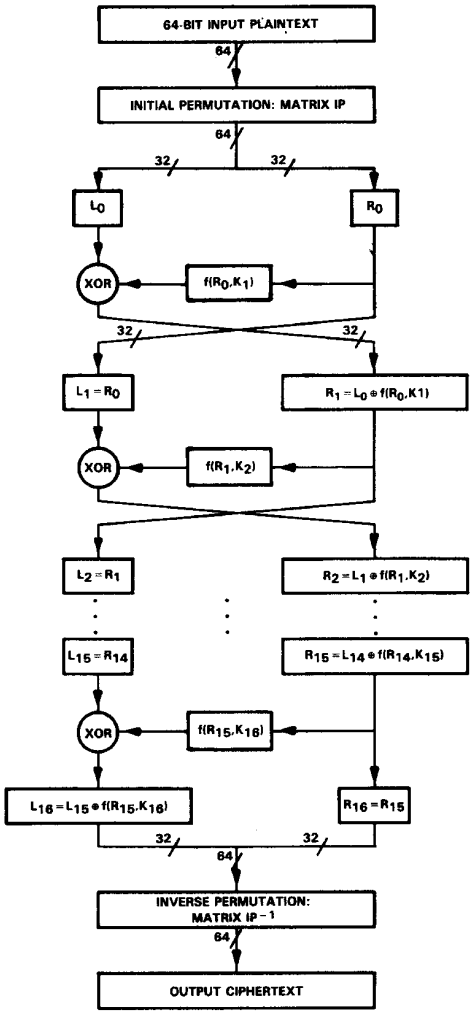

Figure 1. Flowchart of the DES Encryption

The initial permutation and inverse initial permutation matrices IP and IP^{-1} in Figure 1 are shown in Figures 2 and 3. Where there is bit manipulation according to a matrix, the matrix should be read from left to right and row by row. Each entry of the matrix corresponds to a bit in the output block. In other words, the information presented in the matrix is sequential and not two-dimensional. The matrix arrangement is used here only for convenience of presentation.

Each entry of the matrix indicates the order of the bit in the input bit stream, starting from the leftmost bit. For example, during the initial permutation, the leftmost bit after the permutation is the 58th bit from the left in the input stream; the second leftmost bit is the 50th bit, etc. If the matrix has as many entries as the number of bits of the input block, the input and output blocks from the bit manipulation have the same number of bits. If, however, the matrix has more entries (by repeating some of the bits of the input block) or fewer entries (by skipping some input-block bits), the output of the bit manipulation is longer or shorter, respectively, than the input. Such examples, shown later in this report, are matrix E that transforms 32 bits into 48 bits, and matrix PC-2 that transforms 56 bits to 48 bits.

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Figure 2. Matrix IP

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Figure 3. Matrix IP^{-1}

The function $f(R_{i-1},K_i)$ of Figure 1, which combines the right block R_{i-1} of the previous stage and the subkey K_i of the present stage i, is shown in detail in Figure 4. The input to the function is a block of 32 bits, and the output is also a block of 32 bits. Figures 5, 6, and 7 give the matrices E, P, and S1 through S8 of Figure 4. K_i's are subkeys generated from the main key, as described in the next section. The S-boxes, which convert a 6-bit sequence to a 4-bit sequence, are used as follows: the first and last bits of the sequence, taken together, represent a number I between 0 and 3, while the middle 4 bits represent a number J between 0 and 15. The S-boxes are 4 x 16 matrices whose entries take values between 0 and 15. Each of these entries can be represented by a 4-bit number. For each S-box, the I and

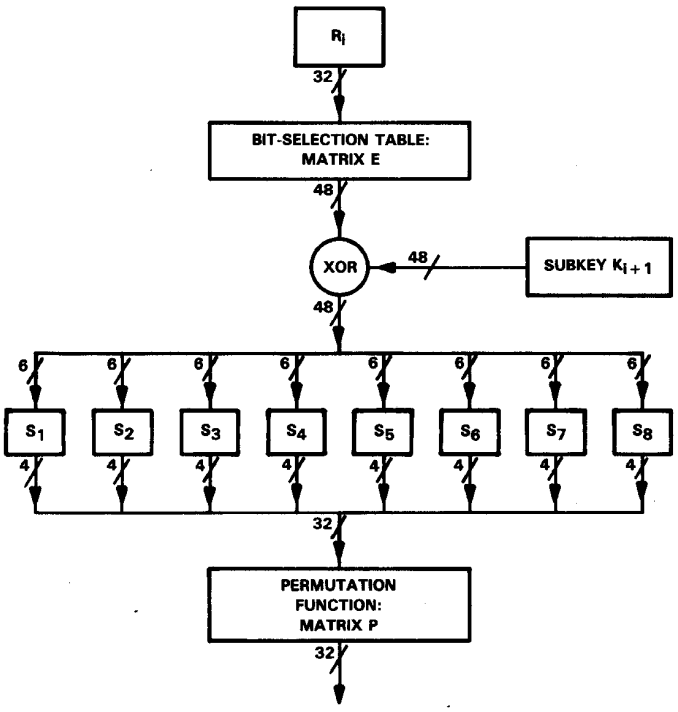

Figure 4. Computation of the Function $f(R_i,K_{i+1})$ of Figure 1

J are computed, and the corresponding 4-bit number in the matrix is the output. For example, if the input to S1 box is 001010, then I = 0, J = 5, and the output is the decimal 15, which corresponds to the 4-bit binary number 1111.

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Figure 5. Matrix E

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Figure 6. Matrix P

S1

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S2

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S3

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S4

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S5

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S6

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

Figure 7. S Matrices

S7

4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S8

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Figure 7. S Matrices (continued)

For decrypting the ciphertext, the same procedure is used as for encrypting, except that the transformation block in Figure 1 is modified as shown in Figure 8. The left and right 32-bit word locations are exchanged, and the algorithm starts with subkey 16 and ends with subkey 1. The implementation for decryption is similar to encryption. The TMS32010 code (see the appendix) includes a flag to signal if encryption or decryption is performed. In decryption, the subkeys are read backwards, but everything else remains the same.

Generation of the Subkeys

The encryption key is 64 bits long, out of which only 56 bits are used for the encryption. In every group of eight bits, the eighth bit is such that the byte has odd parity. This provides some guarantee of key integrity during transmission. From the 64-bit key, 16 subkeys, each 48-bits long, are generated as shown in Figure 9. The matrices for permuted choices 1 and 2, PC-1 and PC-2, are given in Figures 10 and 11, respectively. Table 1 shows the number of left shifts at each stage. The indicated left shifts are circular left shifts for each half of keys C_i and D_i.

Table 1. Left Shifts for Subkey Generation

Iteration Number	Number of Left Shifts
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

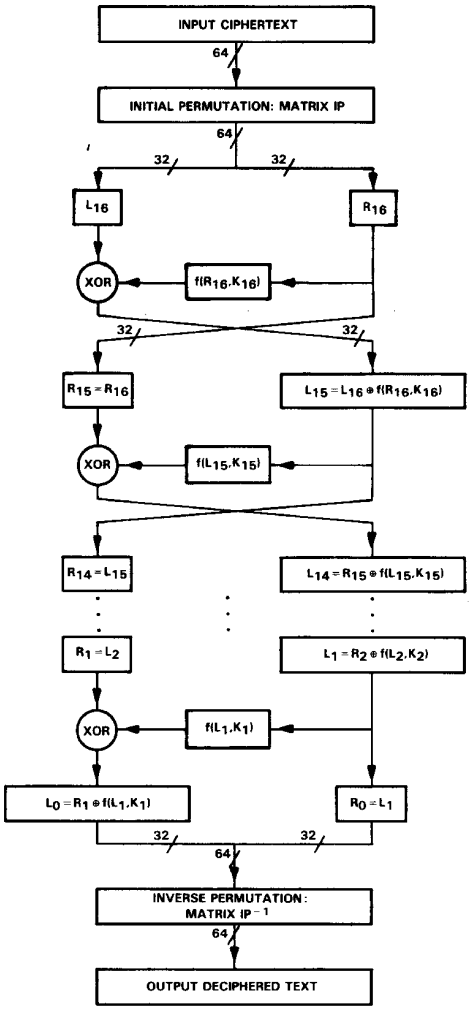

Figure 8. Flowchart of the DES Decryption

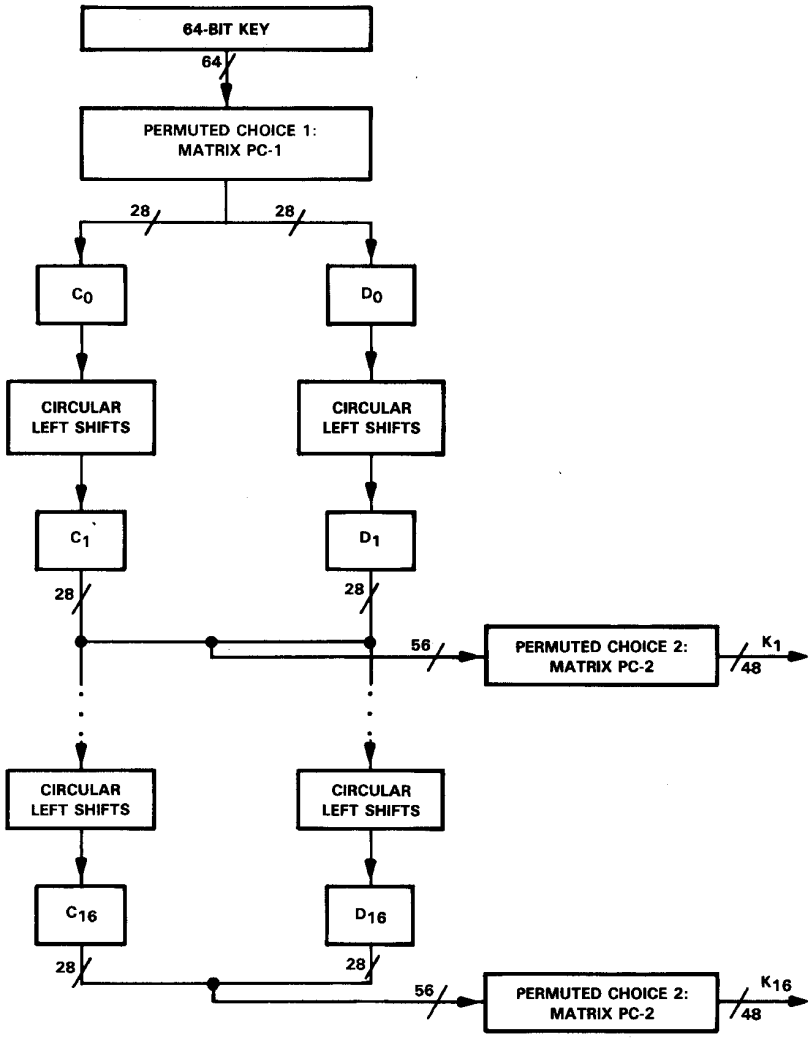

Figure 9. Generation of 16 48-Bit Subkeys from the Encryption Key

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Figure 10. Matrix of PC-1

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Figure 11. Matrix PC-2

MODES OF OPERATION

The DES algorithm can be implemented in different modes of operation[2], two of which are the Electronic CodeBook (ECB) mode and the Output FeedBack (OFB) mode. Selection of a mode determines the kind of plaintext to serve as input to the algorithm and how the ciphertext is used.

Figure 12 is another representation of Figure 1 and corresponds to the most straightforward application of the algorithm, the Electronic CodeBook (ECB) mode. The plaintext is the actual bit stream to be encrypted, and the ciphertext is what is actually transmitted over the communications medium.

Figure 13 depicts the 64-bit Output FeedBack (OFB) mode. In this case, the 64 bits of the output of the DES algorithm are exclusive-ORed with 64 bits of the actual bit stream to generate the encrypted stream for transmission. The same 64 bits of the DES output are then fed back to the input of the DES as the new input to the algorithm. The whole process is initialized by inputting a predetermined block of 64 bits into the DES algorithm. The receiver has a similar operation. As long as it has the same initial 64-bit block as input, the receiver operates synchronously with the transmitter. Note that in this mode there is no need to implement the decrypting portion of the algorithm, because both the transmitter and the receiver are using only the encrypting portion of the DES. The OFB mode has the advantage over the ECB that it confines any transmission errors to single bits, rather than a block of 64 bits.

TMS32010 IMPLEMENTATION OF THE DES

In addition to implementing the main encryption/decryption algorithm, it is necessary to process the encryption key in order to generate the subkeys used in the DES. The key consists of 64 bits, of which only 56 are active. In every byte of the key, the eighth bit is an odd-parity bit. To avoid any mistakes, the user is asked to supply the 56 bits. The device then generates the correct 64-bit version that can be used for storage and transmission. This function is realized by the routine DESKEY. The whole implementation of the Data Encryption Standard in TMS32010 assembly language can be found in the appendix. Figure 14 is a flowchart illustrating the arrangement of the TMS32010 programs that implement the DES.

The largest part of the code consists of a repetitive construction, which implements the permutation of a block of bits according to a matrix. This construction is discussed here to provide a better understanding when implementing the code.

Assume that a certain step of the algorithm requires the scrambling of 64 bits according to a matrix. The 64 input bits are stored in four words, WI1 to WI4. The scrambled-bit output is also stored in four words, WO1 to WO4. The scrambling is accomplished by constructing the

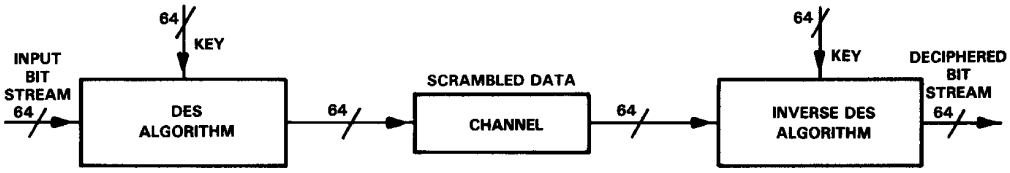

Figure 12. Electronic CodeBook (ECB) Encryption Mode

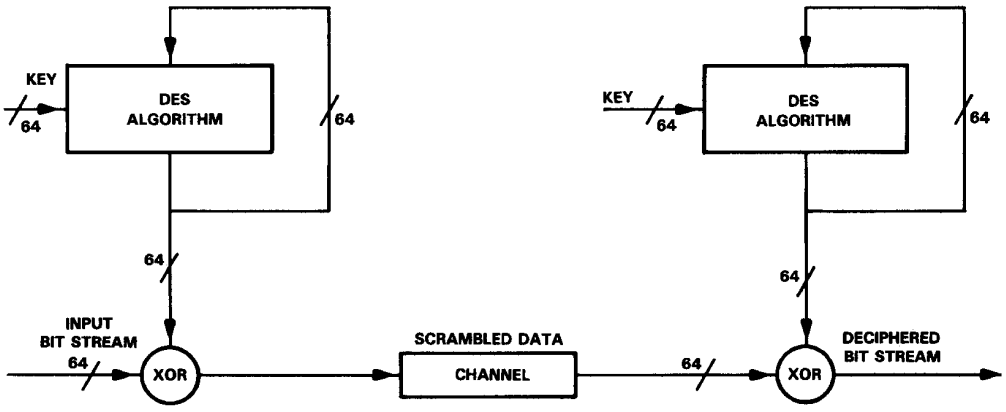

Figure 13. Output FeedBack (OFB) Encryption Mode

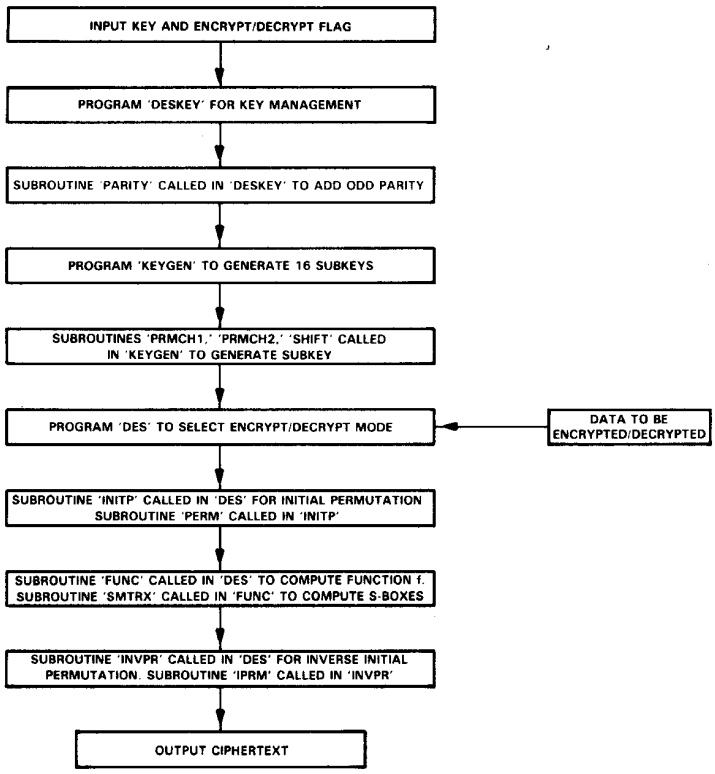

Figure 14. Flowchart of TMS32010 Programs Implementing the DES

WO1 to WO4, one bit at a time from left to right. For example, let the current entry of the matrix be the 35th entry and have the value 53. This means that WO1 and WO2 are already filled (containing 2 × 16 = 32 bits), and now the third bit of WO3 is being determined. Since that bit is the 53rd bit of the input stream, it occupies the fifth location in WI4 (53 = 3 × 16 + 5). If M1 contains >8000 (hexadecimal value), the transfer of the bit from the input stream to the scrambled output stream is performed by the following code:

```
LAC     W14,4
AND     M1
ADDH    WO3
SACH    WO3,1
```

In other words, the bit of interest is positioned at the leftmost location of the accumulator's low word, and then masked out by M1. The current contents of WO3 are added to the accumulator's high word, which is then stored back to WO3 shifting by one. In this way, the bit of interest is also picked up.

In order to run the main algorithm, 16 subkeys, each 48-bits long, must be generated, to be used at the 16 stages of the algorithm. The algorithm for the generation of the subkeys involves permutations and shifts of the original key. The method of generating the 16 subkeys is implemented in the routine KEYGEN. For convenience, each subkey is stored in four 16-bit words so that only the top 12 bits are active in each word.

Having generated the subkeys, an incoming stream of bits can be encrypted or decrypted. The body of the DES algorithm is implemented in the routine DES. This routine implements both encryption and decryption, and the appropriate choice is flagged by a bit. In the present implementation, this bit is supplied together with the key.

Table 2 shows the program storage in words and the execution time in instruction cycles and milliseconds, as required by the DES algorithm. The information given for the routine DES is the time required per 64 input data bits. The execution time of the DES algorithm corresponds to a loading of 100 percent of the device for a bit rate of about 42.5 kbit/s. This makes the TMS32010 a faster encryption device than some of the currently available dedicated devices.[9]

Table 2. TMS32010 Memory and Speed for the DES

Function	Program Memory (words)	Execution Time (instruction cycles)	(ms)
Key Management and Subkey Generation	600	4527	0.905
Encryption	1102	7464	1.493
Decryption	(both)	7542	1.508

NOTE: The encryption/decryption execution time is an average for the half-duplex operation and for processing 64 bits, based on a 200-ns instruction cycle. The corresponding actual bit rates are:

Encryption: 42.86 kbit/s
Decryption: 42.42 kbit/s

The data memory required for the implementation of the DES includes 64 locations for the subkeys and another 51 locations as scratch registers. If an application running concurrently requires more data memory, the minimum memory requirement is four words, which contain the DES key. The subkeys can be regenerated at the beginning of every encrypting (or decrypting) session (but with a corresponding penalty in execution time).

Note that no attempt of optimization is made in this implementation. Because of tradeoffs between data memory, program memory, and execution speed, the designer must optimize the implementation of the algorithm to fit the needs of the application.

ENCRYPTION OF CODED SPEECH

The TMS32010 Digital Signal Processor has been applied to encoding of speech signals in a variety of approaches. Its architecture and instruction set are particularly convenient for easy implementation and fast execution of speech algorithms, such as those used in vocoders. Several algorithms have been implemented on the TMS32010 to encode speech data at 2.4, 9.6, 16, and 32 kbit/s.[3-8] These speech-encoding algorithms can be readily combined with a data encryption method to provide security both in store-and-forward applications and in telecommunications applications. In this section, a review of the requirements for the speech coding algorithms is presented to show how the DES can be incorporated in a speech coding application. Figure 15 illustrates how the DES may be implemented on the TMS32010 processor for either data or speech encryption.

Table 3 presents the different speech-coding algorithms that have been implemented on the TMS32010. Table 4 summarizes the loading of the TMS32010, when running the DES, for several bit rates corresponding to the bit rates of Table 3. Note that these loadings are for half-duplex operation, i.e., for either encryption or decryption only.

Table 3. Speech-Coding Algorithms Implemented on the TMS32010

Coding Method	Bit Rate (kbit/s)	TMS32010 Loading (percent realtime)	
		Analysis	Synthesis
LPC	2.4	43	44
RELP	9.6	91	79
APC	16	87	35
SBC	16	38	38
ADPCM	32	47	45
ADPCM-CCITT	32	91	95

NOTE: LPC = Linear Predicitive Coding, 10th-order model
RELP = Residual Excited Linear Predictive vocoder
APC = Adaptive Predictive Coding
SBC = Sub-Band Coding
ADPCM = Adaptive Differential PCM (CCITT algorithm, not bit-by-bit compatible)
ADPCM-CCITT = CCITT ADPCM (bit-by-bit compatible)

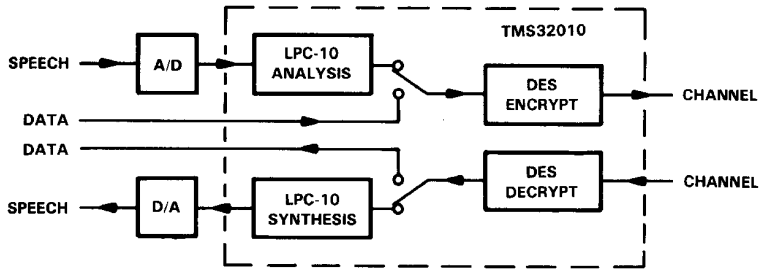

Figure 15. Speech or Data Encryption/Decryption with the TMS32010

Table 4. TMS32010 Loading Time for the DES (Half-Duplex)

Bit Rate (kbit/s)	TMS32010 Loading (percent realtime)
42.8	100
32	75
16	37
9.6	22
2.4	6

NOTE: The loading (for encryption) is given as a percentage of 42.8 kbit/s. The other bit rates were selected to match the bit rates in Table 3.

A comparison of Tables 3 and 4 shows that the 2.4-kbit/s full-duplex system can be implemented in realtime, including the encryption, in only one TMS32010. However, for the other coding schemes, it is necessary to use a second chip. The speech algorithms used have been selected without consideration for any other processing of the speech signal. It may be possible to restructure the marginal cases in order to implement the combined speech and encryption algorithms with fewer chips. Other speech coding algorithms with lighter computational load may also be considered.

CONCLUSION

The TMS32010 Digital Signal Processor can be used to implement algorithms, such as the DES, which in the past have been implemented on dedicated devices. The 200-ns cycle time of the TMS32010 makes possible a half-duplex rate of 42.5 kbit/s for the DES. This rate is superior to that achieved by some of the available devices. The single-cycle multiplication is used in the computation of the S-boxes and helps increase encryption speed. Applications of the encryption include not only speech, as discussed in this report, but any kind of digital data. For example, implementation of a V.22 bis modem can be combined with the DES to achieve security of the transmitted information. An important point to be considered is that, having designed the TMS32010 in a system, the data encryption function can be added with minimal system cost because of the programmability of the device. This feature is a natural extension in the functionality of a digital signal processor.

REFERENCES

1. *Data Encryption Standard,* FIPS Publication 46, National Bureau of Standards (January 1977).
2. *DES Modes of Operation,* FIPS Publication 81, National Bureau of Standards (November 1981).
3. A.W. Holk and W.W. Anderson, "A Single-Processor LPC Vocoder,"*Proceedings 1984 IEEE International Conference on Acoustics, Speech and Signal Processing,* 44.13.1-4 (March 1984).
4. W.K. Gass and M.M. Arjmand, "Real-Time 9600 Bits/Sec Speech Coding on the TI Professional Computer," *Proceedings 1984 IEEE International Conference on Acoustics, Speech and Signal Processing,* 27.9.1-4 (March 1984).
5. B.S. Atal and M.R. Schroeder, "Adaptive Predictive Coding of Speech Signals," *Bell System Technology Journal,* **Vol 49**, 1973-1986 (October 1970).
6. T.P. Barnwell, III, R.W. Schafer, R.M. Mercereau, and D.L. Smith, "A Real Time Speech Subband Coder Using the TMS32010," *Proceedings IEEE Southcon 1984* (1984).
7. "Recommendation G.721: 32 Kbit/s Adaptive Differential Pulse Code Modulation (ADPCM)," *CCITT* (October 1984).
8. J.B. Reimer, M. McMahan, M.M. Arjmand, *32-kbit/s ADPCM with the TMS32010,* Texas Instruments (1985).
9. C.R. Abbruscato, "Data Encryption Equipment," *IEEE Communications Magazine,* **Vol 22**, 15-21 (September 1984).

Appendix

TMS32010 Assembly Language Programs

The TMS32010 code implementing the DES is not published in this appendix because it falls within the U.S. Department of State Export Control Regulations. If a copy of the code is desired, please contact your local TI representative.