Dr. Siddeeq Y. Ameen
Computer Engineering and Information Technology Dept.,
University of Technology,
e-mail: siddeeq_ameen@coolgoose.com

Dr. Abbas A. Al-Shalchi
Dept. of Electronic and Communications Engineering,
Nahrain University

Muhanad D. Al-Bayati
Dept. of Computer Engineering,
Baghdad University

Design and Hardware Implementation of a Speech Cipher System

Siddeeq Y. Ameen; Prof., Abbas A. Al-Shalchi; Assisstant Prof.,
Muhanad D. Al-Bayati; M. Sc.

Abstract

Digital ciphering of speech signals based on one of modern cryptography algorithms, called the Rijndael algorithm, is studied and presented in this paper. The algorithm meets most of the requirements of security level in recent applications. A system to encrypt speech files recorded with Sound Blaster Card of a personal computer was proposed and simulated successfully using MATLAB® language.

Subjective measure and objective measure using segmental spectral signal-to-noise ratio, were used to test the proposed system performance. In these tests residual intelligibility of the encrypted speech and quality of the recovered speech were calculated and assessed.

Finally, a hardware implementation of the above cipher system has been proposed using the TMS320-C30. The real time requirements from the speech cipher system have been computed in terms of execution time together with factors affecting such implementation. The results show the capability of the cipher system to be implemented using the DSP device suggested. Furthermore, the results of hardware implementation also show the security of the system is very close to that of the simulated version.

Keywords: *Speech Cipher, AES, Speech Security, Cryptography*

1. Introduction

Speech is probably the most fundamental form of communication available to use. There are many possible channels for the transmission of speech signals. Obvious examples are the worldwide telephone network and the large number of private and public radio communication systems. However, there are some situations where the information being transmitted is confidential and the communicators detest any third party to understand their messages. The clearest examples are diplomatic communications and military communications during war and peace. The best way to solve such problems is to transform the message in some other ways prior to the transmission in order to conceal the content of the message, which is the object of a cipher system to create secure speech communication [1,2].

Speech has more redundancy as compared with written text or digital data. This makes encryption of a speech signal with low residual intelligibility and high cryptanalytic strength a very difficult task.There are two fundamentally distinct approaches to achieve speech security in speech communication systems; analog scrambling and digital ciphering.

In each type, there are many encryption techniques available to the designer of speech encryption equipment, but the choice of one of them depends on the following factors [3]:

i- The available communication channels and bandwidth requirements.
ii- Amount of security required.
iii- The synchronization requirement.
iv- The residual intelligibility resulting from the applied cipher system.

In the past, speech security systems depended on scrambling the analog speech by one of the scrambling techniques ; this is because of the limited capabilities at that time. Until now searchers have been intersted in speech scrambling because of its efficient facilities like small bandwidth of the scrambled speech, simple implementation and the capability of asynchronous transmission [2].

Most of the analog scrambler methods rely on digital signal processing. In these scrambling methods, the speech signal is first converted into discrete samples and the samples are scrambled in time, frequency, amplitude, or hybrid between two or more of the previous domains. The inverse operation at the receiver is performed. Such method is basically implemented using a transform domain to increase the level of security [3,4].

The main disadvantages of the analog scrambling are the low level of security reached and distorted quality of the recovered speech. Therefore, digital ciphering is used to avoid these problems. In digital ciphering the original analog signal is digitized at first to the digital form by any suitable coding method such as pulse code modulation, delta modulation, LPC, etc. This digital form is then enciphered into a different form by some ciphering algorithm. Digital ciphering can be classified into two types according to the method of processing of speech samples. These two types are stream ciphering and block ciphering. Generally, digital encryption can give low residual intelligibility and higher cryptanaltic strength but most coding techniques used in digital ciphering systems increase the signal bandwidth and synchronization is needed between transmitter and receiver. The problem of bandwidth expansion can be avoided by compressing the data before transmission [4].

2. RIJNDAEL: A Successor to Data Encryption Standard

The encryption algorithm used in this paper is the AES or Rijndael algorithm. Rijndael is a symmetric block encryption algorithm that encrypts blocks of 128, 192, or 256 bits and uses symmetric keys of 128, 192, or 256 bits, where all combinations of block and key lengths are possible. It has the following features [5]:

i- The implementation of Rijndael can at the least cost be protected against attacks that are based on measurements of the time behavior of the hardware (so-called timing attacks) or change in electrical current use (so-called power or differential power analysis attacks).

ii- Rijndael algorithm can most rapidly encrypt and decrypt data.

iii-Rijndael makes use of very limited resources of RAM and ROM memory.

iv- Rijndael has the best performance in hardware implementation.

Each block of plaintext is encrypted several times with a repeating sequence of various functions, in so-called rounds [5]. The number of rounds Nr depends on the block length Nb and key length Nk. If at least the block or key length is 256 bits, there are 14 rounds; if both the block and key length are 128 bits, there are 10 rounds [5]. It consists of an initial round (AddRoundKey), and Nr standard rounds. The first Nr-1 rounds are similar and they consist of four transformations, called: ByteSub (Substitution Bytes), ShiftRow (Shift Rows), MixColumn (Multiply Columns), and AddRoundKey (XORed by key). The last round has only the transformations ByteSub, ShiftRow, and AddRoundKey. Further details about the Rijndael algorithm and its operation can be found elsewhere [5].

3. Software Simulation of the Proposed Speech Cipher System

Figure 1 shows the operation sequences that describe the proposed speech cipher system. This system has been simulated, using MATLAB® language, to encrypt speech files off-line then decrypt the ciphered speech files off-line too.

The operation of the proposed system is as follows:

i- Messages were recorded on the computer using one of the recording programs (under Windows) with specified sampling rate, number of bits per sample, and file extension. The selected parameters of the recorded files were sampled at frequency of 8 kHz and encoded by 8 bits per sample to produce a file with extension of .wav which is the Microsoft standard PCM recording. The recorded files already filtered by a low-pass filter with cut-off frequency of fs/2 (4 kHz in this case).

ii- The recorded file is opened in MATLAB® using the function wavread( ) which gives the sampling frequency, number of bits per sample, and vector of normalized speech samples with range of [-1 ~ +1].

iii- The Rijndael algorithm processes integer data with variable word length. Therefore, the floating data of the normalized speech vector are converted into an integer vector with range of [0 ~ 255] using the equation:

$$\text{Integer vector} = \text{fix}(\,255(\text{Float vector} + 1) / 2) \tag{1}$$

where 255(Float +1)/2 converts floating values of the vector from [-1 ~ +1] to [0 ~ 255] and fix( ) function truncates the fractions. The process of converting the speech vector from floating values [-1 ~ +1] to finite integer values [0 ~ 255], is similar to the analog-to-digital conversion process.

iv- The integer vector is encrypted, block by block, using Rijndael encryption algorithm with defined cipher-key. The encrypted block is saved in data file to keep a version of the encrypted speech for testing. This is the last stage of the transmitter side.

v- After transmission through the channel (lossless channel is assumed here), the receiver receives the encrypted block. The first stage of the receiver is the decryption of the received block using the Rijndael decryption algorithm with the same cipher-key of the transmitter side.

vi- The frame (block) of decrypted speech of range [0 ~ 255] is renormalized to the range [-1 ~ +1] using : Normalized vector = (2 Integer vector / 255) –1

vii- The normalized vector is passed through a Butterworth low-pass filter with cut-off frequency 4 kHz and 16th order to average this vector. This stage of the receiver is similar to digital-to-analog conversion. The output of this stage is the analog recovered speech frame.

viii- The recovered speech frame is stored in the wave file with sampling frequency of 8kHz and 8 bits per sample using the MATLAB® function wavwrite( ) and stored also as data files for tests.

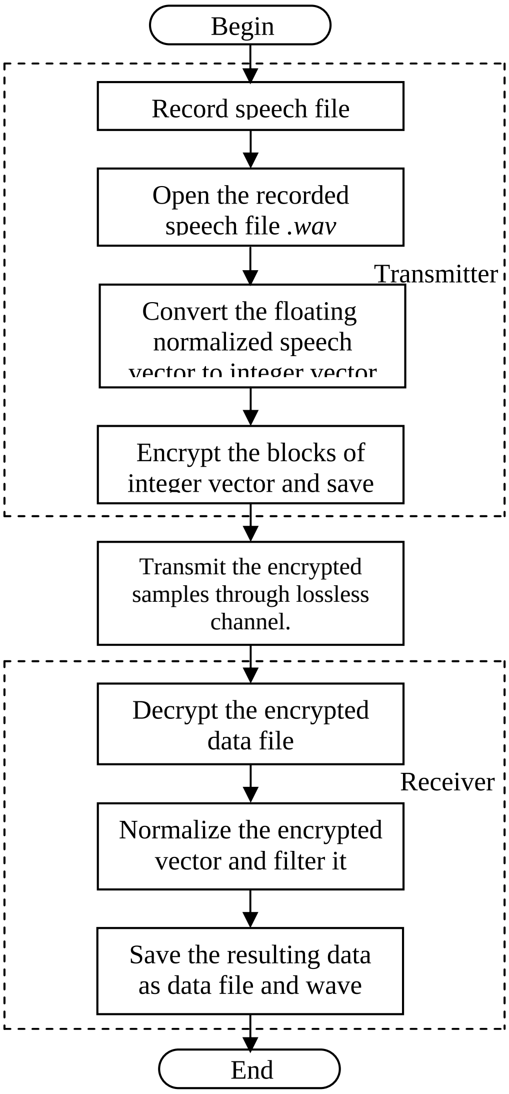
Begin
Record speech file
Open the recorded speech file .wav
Transmitter
Convert the floating normalized speech vector to integer vector
Encrypt the blocks of integer vector and save
Transmit the encrypted samples through lossless channel.
Decrypt the encrypted data file
Receiver
Normalize the encrypted vector and filter it
Save the resulting data as data file and wave
End

Figure (1) Operations sequence of the proposed speech cipher system.

Software simulation has many parameters like the value of the block length, value of the key length, message to be encrypted and the speaker sex. All these parameters have been varied during the simulation tests. Following are the different parameters used in the simulation tests:

i- A message has been recorded with sampling frequency of 8 kHz and 8 bits per sample as speech files. The message in English states:

“The colors of the sun spectrum are: Violet, Dark Blue, Blue, Green, Yellow, Orange, and Red, respectively. The first one is near to the Ultra-Violet ray while the last one is near to the Infra-Red ray”.

ii- Speakers; the message may be spoken by a man or a woman.

iii- Values of the block and key lengths: Rijndael algorithm uses block and key of different lengths(16, 24, 32 bytes). This point has been considered also in the test, so that, both block and key taken with all possible lengths to examine the performance have been considered.

iv- Values of the user key elements: For each key length, two settings of the elements have been selected randomly to test the variation of the speech security level with respect to the user key variation. Table 1 illustrates the keys used and their lengths.

Table (1): The Cipher-Key samples used in the simulation tests

Key	Nk	Elements of the Key
1	**4**	***00010203 04050607 08090A0B 0C0D0E0F***
	6	***00010203 04050607 08090A0B 0C0D0E0F 00010203 04050607***
	8	***00010203 04050607 08090A0B 0C0D0E0F 00010203 04050607 08090A0B 0C0D0E0F***
2	**4**	***AA01095C 41893467 1101B306 FCFDF17B***
	6	***AA01095C 41893467 1101B306 FCFDF17B AA01095C 41893467***
	8	***AA01095C 41893467 1101B306 FCFDF17B AA01095C 41893467 1101B306 FCFDF17B***

Two types of tests have been used to examine the performance of the simulation, these are:

i- Subjective test in which the binary encrypted speech file is converted to normalized form, filtered and saved in wave file. These files were played back to a number of listeners to measure the residual intelligibility, subjectively. For all cases, The judge was that the files contain noise only, which means that the residual intelligibility is very low. The analog recovered speech files were tested in a similar way to measure the quality of the recovered speech files, the judge was that the files are exactly the same as the original copies.

To support the subjective tests a sample of duration 3 sec. had been taken for one case where English message spoken by a male voice with Nb=Nk=4 and the first key are used. The results from such test are shown in Fig.2 for the original speech signal, Fig.3 for the analog encrypted signal which appears as noise, and Fig.4 for the analog recovered speech signal.

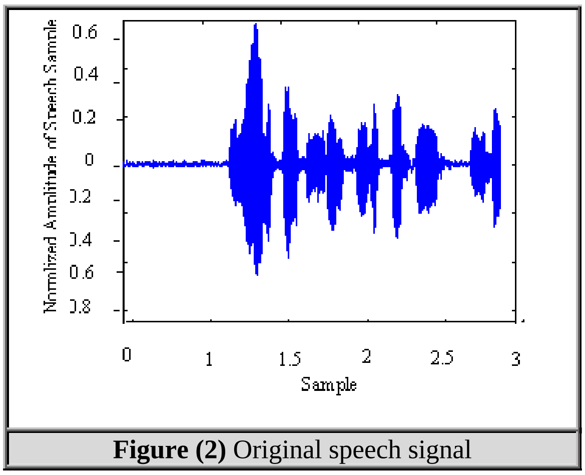

Figure (2) Original speech signal

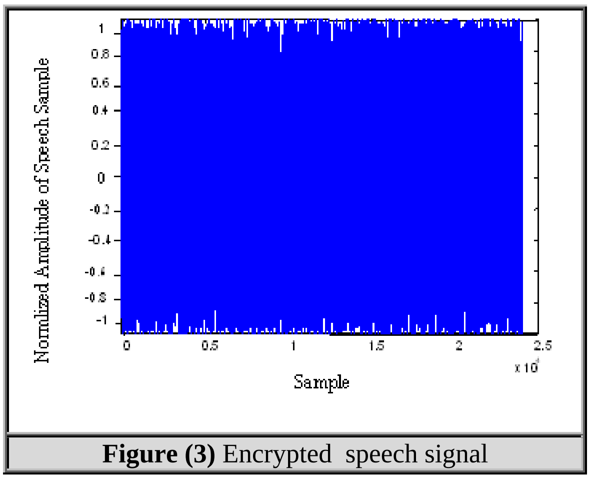

Figure (3) Encrypted speech signal

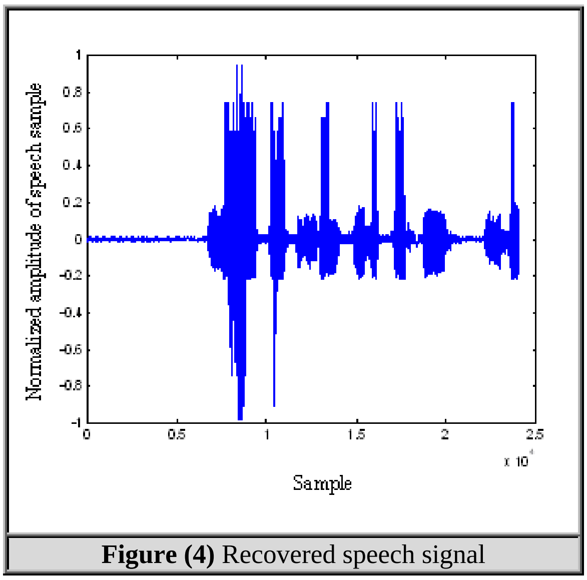

Figure (4) Recovered speech signal

Objective Test: As mentioned earlier, the objective test is a valuable measure of the residual intelligibility and the quality of the recovered speech. The segmental spectral signal-to-noise ratio measure was chosen to test the residual intelligibility and the quality of the recovered speech for all files of different Nb, which represents the frame length.

The segmental spectral signal to noise ratio in the frequency domain for the ith frame of speech can be defined as [3]:

$$SSNR_i = 10 \log \frac{\sum_{l=1}^{N} \left| X_i(l) \right|^2}{\sum_{l=1}^{N} \left[\left| X_i(l) \right| - \left| Y_i(l) \right| \right]^2} \qquad (2)$$

where Xi(l) is the DFT of the ith frame of the original speech samples and Yi(l) is the DFT of the corresponding frame of the encrypted or recovered speech samples. Since the human ear is generally insensitive to phase errors, comparison in eq.(2) is given in terms of magnitude of frequency domain samples. These SSNRs for different frames are averaged to give the SNR over the tested speech segments. The above segmental SNR, defined in the frequency domain, is a useful indicator of speech quality or intelligibility loss [3].

Table 2 shows the overall test results for all the selected settings of the system parameters. Generally, the SNR for all the encrypted speech files are very low (large negative values) which means that the residual intelligibility is very low, and the SNR for all the recovered speech files is very high (large positive values) which means that the quality of the recovered speech is very high. Tests with different block and key lengths, values of the user key elements and the message contents have no significant effects on the parameters of speech security measures. The only effective parameter was the speaker sex, but the difference of the SSNR between men and women is not large compared with the absolute value of the SNR so that, the SSNR is still large (negative or positive) for both.

Table (1): SNR for the encrypted and recovered English message

		Man		Woman	
Nk	Nb	SNRe	SNRd	SNRe	SNRd
4	4	-60.14	31.30	-54.04	32.61
	6	-60.01	31.30	-54.05	32.61
	8	-60.16	31.30	-54.04	32.61
6	4	-60.12	31.30	-54.09	32.61
	6	-60.00	31.30	-54.06	32.61
	8	-60.14	31.30	-54.07	32.61
8	4	-60.10	31.30	-54.03	32.61
	6	-60.04	31.30	-54.00	32.61
	8	-60.17	31.30	-54.05	32.61

4. Hardware Implementation of the Security System

One of the interesting aspects in the design of any microprocessor-based system is the interaction between hardware and software. The designer should therefore have the capability of designing efficient hardware with efficient software to obtain the optimum requirements. In this section all the software used to integrate the hardware operation will be illustrated. The software used can be divided into three types. The first type is the package programs (like Loder, Geo, etc.) which is used to drive the hardware and communicate the hardware with the PC. The second type is the program responsible for the encryption, decryption processes. Finally, the third type is the programs that manage system operation.

4.1 Supporting Package Programs

The following is a brief description of these programs[7]:

i- Loder: This program is stored in the EPROM of the DSP card. It is executed at power-on or when the hardware is reset. It initialises the DSP processor (registers, interrupts, stack, etc.) and by dialogue procedure communicates with the designer to load a program from the PC and locates the loaded programs in the RAM of the card. The program can be executed from any valid location, reads from/ writes into any memory location, etc.

ii-Geo: An interfacing communication program at the personal computer responsible for the interfacing job between the two sides, the card and the PC . It offers many options for general communication between PC and external devices through the RS232.

iii-Translator : All programs implemented on the DSP card are entered using any ASCII code editor with TMS320-C30 assembly language syntax. The edited program is the assembly language program, which is called the source file with an extension of .asm.

4.2 Hardware Implementation of the Rijndael Algorithm

Because of the efficient properties of the TMS320C30 processor (32 bit processing, high speed and large memory mapping), the Rijndael algorithm is to be rearranged to be more compatible with the hardware.

The main part of the Rijndael encryption/decryption program is the round transformation. Therefore, the round transformation can be simplified to save execution time for real time implementation. The different steps of the round transformation can be combined in a single set of look-up tables.

One column of the output round s can be expressed in terms of the input round t where tij denotes the byte of t in the row i and column j. The symbol t j denotes the column j of state t. For the Round-Key addition and the MixColumn transformation:

$$\begin{bmatrix} s_{0,j} \\ s_{1,j} \\ s_{2,j} \\ s_{3,j} \end{bmatrix} = \begin{bmatrix} d_{0,j} \\ d_{1,j} \\ d_{2,j} \\ d_{3,j} \end{bmatrix} \oplus \begin{bmatrix} k_{0,j} \\ k_{1,j} \\ k_{2,j} \\ k_{3,j} \end{bmatrix}$$

and

$$\begin{bmatrix} d_{0,j} \\ d_{1,j} \\ d_{2,j} \\ d_{3,j} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} c_{0,j} \\ c_{1,j} \\ c_{2,j} \\ c_{3,j} \end{bmatrix} \tag{3}$$

For the ShiftRow and ByteSub transformations

$$\begin{bmatrix} c_{0,j} \\ c_{1,j} \\ c_{2,j} \\ c_{3,j} \end{bmatrix} = \begin{bmatrix} b_{0,j} \\ b_{1,j-C1} \\ b_{2,j-C2} \\ b_{3,j-C3} \end{bmatrix} \quad \text{and} \quad b_{i,j} = S\lfloor t_{i,j} \rfloor \tag{4}$$

In this expression the column indices must be taken modulo Nb. By substitution, the above expression can be combined into

$$\begin{bmatrix} s_{0,j} \\ s_{1,j} \\ s_{2,j} \\ s_{3,j} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \bullet \begin{bmatrix} S[t_{0,j}] \\ S[t_{1,j-C1}] \\ S[t_{2,j-C2}] \\ S[t_{3,j-C3}] \end{bmatrix} \oplus \begin{bmatrix} k_{0,j} \\ k_{1,j} \\ k_{2,j} \\ k_{3,j} \end{bmatrix} \tag{5}$$

The matrix combination can be expressed as linear combination of vectors

$$\begin{bmatrix} s_{0,j} \\ s_{1,j} \\ s_{2,j} \\ s_{3,j} \end{bmatrix} = S\left[d_{0,j}\right] \begin{bmatrix} 02 \\ 01 \\ 01 \\ 03 \end{bmatrix} \oplus S\left[d_{1,j-C1}\right] \begin{bmatrix} 03 \\ 02 \\ 01 \\ 01 \end{bmatrix} \oplus$$

$$S\left[d_{2,j-C2}\right] \begin{bmatrix} 01 \\ 03 \\ 02 \\ 01 \end{bmatrix} \oplus S\left[d_{3,j-C3}\right] \begin{bmatrix} 01 \\ 01 \\ 03 \\ 02 \end{bmatrix} \oplus \begin{bmatrix} 01 \\ 03 \\ 02 \\ 01 \end{bmatrix} \oplus \begin{bmatrix} k_{0,j} \\ k_{1,j} \\ k_{2,j} \\ k_{3,j} \end{bmatrix} \tag{6}$$

The multiplication factors S[ti j] of the four vectors are obtained by performing look-up table on input bytes tij in the S-box table. Tables T0 to T3 can be defined as

$$T_o[t] = \begin{bmatrix} S[t]\bullet 02 \\ S[t] \\ S[t] \\ S[t]\bullet 03 \end{bmatrix} \quad T_1[t] = \begin{bmatrix} S[t]\bullet 03 \\ S[t]\bullet 02 \\ S[t] \\ S[t] \end{bmatrix}$$

$$T_2[t] = \begin{bmatrix} S[t] \\ S[t]\bullet 03 \\ S[t]\bullet 02 \\ S[t] \end{bmatrix} \quad T_3[t] = \begin{bmatrix} S[t] \\ S[t] \\ S[t]\bullet 03 \\ S[t]\bullet 02 \end{bmatrix} \tag{7}$$

These four tables with 256 4-byte entries make up for 4 kbyte of total space. Using these tables, the round transformation can be expressed as

$$s_j = T_0\left\lfloor t_{0,j} \right\rfloor \oplus T_1\left\lfloor t_{1,j-C1} \right\rfloor \oplus T_2\left\lfloor t_{2,j-C2} \right\rfloor \oplus T_3\left\lfloor t_{3,j-C3} \right\rfloor \oplus k_j \tag{8}$$

The round transformation of the decryption algorithm can be implemented with look-up tables in exactly the same manner as the round of the encryption algorithm and there is no performance degradation with respect to the encryption. The look-up tables for the decryption are of course different; these tables can be computed as

$$T_0^{-1}[s] = \begin{bmatrix} S^{-1}[s]\bullet 0E \\ S^{-1}[s]\bullet 09 \\ S^{-1}[s]\bullet 0D \\ S^{-1}[s]\bullet 0B \end{bmatrix} \quad T_1^{-1}[s] = \begin{bmatrix} S^{-1}[s]\bullet 0B \\ S^{-1}[s]\bullet 0E \\ S^{-1}[s]\bullet 09 \\ S^{-1}[s]\bullet 0D \end{bmatrix} \tag{9}$$

$$T_2^{-1}[s]=\begin{bmatrix}S^{-1}[s]\bullet 0D\\ S^{-1}[s]\bullet 0B\\ S^{-1}[s]\bullet 0E\\ S^{-1}[s]\bullet 09\end{bmatrix}\qquad T_3^{-1}[s]=\begin{bmatrix}S^{-1}[s]\bullet 09\\ S^{-1}[s]\bullet 0D\\ S^{-1}[s]\bullet 0B\\ S^{-1}[s]\bullet 0E\end{bmatrix} \tag{10}$$

where s = 0,1,…,255 and S-1[s] denotes the inverse of the S-box replacement. The result of an inverse operation can be determined by

$$\begin{aligned} t_j &= T_0^{-1}\left[b_{0,j}\right]\oplus T_1^{-1}\left[b_{1,j+Nb-C1}\right]\oplus \\ &\quad T_2^{-1}\left[b_{2,j+Nb-C2}\right]\oplus T_3^{-1}\left[b3_{1,j+Nb-C3}\right]\oplus k_j^{-1} \end{aligned} \tag{11}$$

In the final round of the encryption and decryption algorithm, there is no MixColumn operation. This boils down to the fact that the S table must be used instead of the T tables. The need for additional tables can be suppressed by extracting the S tables from the T tables by masking while executing the final round.

Most operations in the key expansion can be implemented by 32-bit word EXORs. The additional transformations are the application of the S-box and a cyclic shift over 8 bits. This can be implemented very efficiently.

4.3 Off-Line Speech Files Ciphering Software Implementation

This program encrypts speech files stored in the PC using the modified Rijndael algorithm which is implemented by the DSP card. Mainly, the program consists of two parts, one written in high level language (C language) and run on the PC Oflin.c, and the other part written in the low level language (Assembly) and run on the DSP-card Oflin.asm. The operation of this program can be described as follows:

i- At the beginning, the Oflin.c program runs. This program asks for the name of the speech file to be encrypted. The name of the encrypted speech file, the numbers of key length and block length, cipher key and other information. It instructs the user to press the hardware reset push-button.

ii- When the hardware reset is pressed , the loader program runs on the card. The loder sends a message containing a group of function choices and asks the PC for the number of the choice to be executed. On the other side the PC program receives the message and at the end of the message the PC sends the number of choices that execute the loading function. The loader now requests the name of the file to be loaded. Therefore the PC program sends the name of the compiled assembly Oflin.lod.

iii- Having finished loading Oflin.lod the choice menu is transmitted again to the PC. The PC receives the number of the choice that executes the run from RAM function and the address of that location in RAM.

iv- After the Oflin.lod is executed, the program requests information from the PC like the cipher-key , the values of the key length and block length, and other data specifying how encryption can be carried out. The PC program sends all these data to the card. At the end of the data exchange, the card sends a message to the PC that the card will expand the cipher-key to expand-key. The PC displays this message after a small delay.

v- When the card finishes key expansion, it sends a message to the PC that the card begins the encryption of the speech file. The PC displays the received message and begins to send frames of speech file to the card. The received encrypted speech frame is stored in the encrypted file and the same for the other frames. In the other side, after the card sends the message, it receives the frame, identifies whether this frame is a speech frame or a key-changing flag. If it is a speech frame the card encrypts and sends it to the PC and receives another frame and so on.

vi- If the frame is a key-changing flag, the card again begins to receive the new cipher-key and repeats steps 5-6.

4.4 Real-Time Speech Ciphering System Implementation

Stand-alone operation of the proposed cipher system needs special software. This software also consists of two types of programs, high-level language language program and low-level language program. The high-level language program is used to initialise the system and input the user data. This program was written using C language and runs on the PC. The low-level language program initializes the hardware, expands the user cipher-key and encrypts or decrypts speech samples taken from ADC as blocks buffered in specific memory location. This program which is written using TMS320C30 DSP-processor assembly language, loaded in the EPROM of the DSP-card and run from that location (no need for loader program). The operation of these two programs is as follows:

i- At the beginning, the PC asks for the values of the key length, block length, cipher-key. It instructs the user to press the hardware reset, and waits for acknowledgment from the card.

ii-When the hardware reset is pressed, the assembly program runs. This program initializes the system (interrupt, registers, stack, etc.) and sends acknowledgment to the PC.

iii-The PC sends the allocated information from the user with more additional data to the card. At the end of data exchange, the card sends amessage to the PC that the external processor will expand the cipher-key to produce the expanded key. The PC displays this message with proper delay.

iv-When the card finishes the key expansion, it sends a message to the PC that the system is ready to exchange encrypted speech between the transmitter and receiver sides. By displaying this message, the PC finishes its job and the card will operate in stand-alone.

v-The card continues in stand-alone operation and according to the mode of operation it transmits or receives encrypted speech. Changing the cipher-key needs to run PC program again, which will reset the hardware and repeat the above procedure.

5. Hardware System Test and Results

The implemented hardware system that satisfies the real-time operation had been tested in different stages separately before system operation. Some stages like LPFs and AGC were tested by simulation package (Electronic Workbench) while the other stages were tested by practical implementation directly. These tests for checking the performance of that stage with the specifications of the design requirements.

To build the overall system, the tested stages were connected together . The microphone was connected to the input of the AGC which is connected to the input of the LPF. The output of the LPF is connected to the ADC which is connected to the data bus of the DSP-card. The data bus was connected to the digital input of the DAC whose analog output connected to a LPF whose output is connected to an audio amplifier and a speaker. Decoding circuit is used to select the ADC/DAC to read from/write to data by the DSP-processor. To examine the operation of the overall system, a simple program was written on the DSP-card which takes samples of speech from the microphone and outputs them on the speaker with 8 kHz sampling frequency. The system operated succesfully.

5.1 Encryption of Speech Files

The encryption process of the Rijndael has been implemented and executed on the DSP-card. The encrypted speech file is stored in the PC. Subjective and objective measures used to examine the output files (encrypted and recovered speech files) with the same variation of the system parameters (key length, block length, cipher-key, etc.). The test results show that the output is exactly the same as that obtained from the software simulation.

5.2 Evaluation of The Real-Time Operation

To ensure the real-time operation of the proposed program, the execution time of the program has been analyzed. The program consists of two main processes. The first process is the PC and external system initialization with cipher-key expansion and the the second process is managing the encrypted conversation between the two authorized parties. Real-time operation does not depend on the initialization process because there is no conversation but the self setting for each side. Therefore, initialization is not included in execution time calculation, i.e., only the second process will be considered. The main task of the second process occurs when the PC displays the message which acknowledges the user that the external card is ready to handle conversation encryption process (with this message the job of the PC is finished). Following are the steps of the conversation managing process with execution time calculation details:

i- Frame of speech with length (4*Nb) allocated from the ADC and stored in the buffering memory. This needs a minimum of 0.125 msec of time between successive samples (1/8 kHz, where 8 kHz is the sampling frequency). This time is composed of the handshake signalling, read sample from ADC, write sample into the buffer and complimentary delay.

ii- Encryption of the bufferd frame and storing the encrypted frame in another buffering memory specifid for transmitting/receiving unit. The encrypted frame must be transferred from the transmitter to the receiver (between cards) with time equal to (4*Nb/64000) msec. to satisfy transmission rate of 64 kbit/sec.

iii- In the receiver, the encrypted frame is reallocated, decrypted and stored in the buffering memory specified for the DAC.

iv- Byte by byte the decrypted frame is transferred to the DAC at a rate of 8 kHz.

Tables 3-5 show the execution time for each step with Nk= 4, 6, and 8, respectively. These tables illustrate the variation of the execution time with Nb, and Nk.

Table (3): Execution time of the conversation encryption process with Nk = 4

	Execution Time (msec.)		
Step	Nb = 4	Nb = 6	Nb = 8
1	2.0000	3.0000	4.0000
2	0.0725	0.1123	0.2673
3	0.2500	0.3750	0.5000
4	0.0725	0.1123	0.2673
5	2.0000	3.0000	4.0000
Total	4.4350	6.7196	9.0346

Table (4): Execution time of the conversation encryption process with Nk = 6

	Execution Time (msec.)		
Step	Nb = 4	Nb = 6	Nb = 8
1	2.0000	3.0000	4.0000
2	0.0953	0.1123	0.2673
3	0.2500	0.3750	0.5000
4	0.0953	0.1123	0.2673
5	2.0000	3.0000	4.0000
Total	4.4806	6.7196	9.0346

Table (5): Execution time of the conversation encryption process with Nk = 8

	Execution Time (msec.)		
Step	Nb = 4	Nb = 6	Nb = 8
1	2.0000	3.0000	4.0000
2	0.1241	0.2008	0.2673
3	0.2500	0.3750	0.5000
4	0.1241	0.2008	0.2673
5	2.0000	3.0000	4.0000
Total	4.4982	6.7766	9.0346

From Tables 3-5, it is clear that the time required for frame encryption of speech (from reading the first sample to writing the first sample) does not exceed the polling time of two successive speech samples (0.125 msec.) for the cases of Nb = 4 and 6. While in the case of Nb = 8, this time limit has been exceeded which means that the values of Nb = 4 and 6 are suitable for real-time operation. In Table 5, only the values of Nb = 4 and 6 satisfy the above condition, while the other values failed to do that.

The third row of the tables shows the time required for transferring the encrypted frame between the two parties. This time, in all tables, is smaller than the encryption time that allows the transmitting/receiving unit to load a new encrypted frame without overlapping.

Generally, speech frames of length 16 bytes (4*Nb and Nb = 4) can be used efficiently for real-time operation with different values of key length (4*Nk). The frames of length 24 bytes can be used for real-time operation with only 16 byte key length. Finally the frames of length 32 bytes can never be used for real-time operation.

6. Conclusions

The most important conclusions derived from such work are:

i- The speech security system is of high level of security, since it uses the well-recommended Rijndael algorithm which has been adopted as the advanced encryption standard because of its good features.

ii- The software implementation of the proposed system shows the ability of using this software as a package for PC applications. Instead of encryption frames of speech stored in a file, one can encrypt frames of speech taken from the microphone of the computer and the encrypted speech is transmitted serially through the media of the network. In the other side the received serial data can be decrypted, converted to analog form, filtered and sent to the loudspeakers of the receiving computer.

iii-From the previous tests on the SSNR, it can be concluded that, Rijndael algorithm can be implemented to encrypt speech with high efficiency.

iv- The variable user key and plaintext lengths give the algorithm strength against attack but has no effect on the objective measures of speech signal.

v- All the security measures and tests prove that the speech security measures vary with the variation of the speaker (man, woman, child) and that women have quite smaller security than men.

vi- From the execution time calculations of the assembly language programs of the integrated system using the DSP-card, it can be concluded that the proposed system can be used as real-time cipher system.

vii- The used DSP processor (TMS320C30) has properties that ensure the real-time operation like speed and portability. Recent hardware can be used to give a portable cipher system in secure speech communication through special channels (telephone, secure military, or other channels).

7. References

1. Gold, B. and Morgan, N. “Speech and Audio Signal Processing, “John Wiley and Sons, New York, 2000.
2. Beker, H. J. and Piper, F. C. “Secure Speech Communications,” Academic Press, New York, London, 1985.
3. Natem, S. A. " Narrow Band Asynchronous Scrambling of Speech Signals ", M.Sc. Thesis, Engineering College, Al-Mustansiriyah University, Baghdad, 1999.
4. Enrico, D. R., Fantacci, R. and Maffucci, D. “ A New Speech Signal Scrambling Method for Secure Communications: Theory, Implementation, and Security Evaluation,” IEEE Journal on Selected Areas in Communications, Vol. 7, No. 4, May 1989, pp. 474-480.
5. Stallings, W. " Cryptography and Network Security: Principles and Practice," 3rd edition, Prentice-Hall, 2003.
6. Welschenbad, M. "Cryptography in C and C++, " A press, New York, USA, 2001.
7. Sorensen and Chen " A Digital Signal Processing Laboratory Using The TMS320C30 ", Prentice Hall, 1997.

تصميم و تنفيذ الكيان المادي لمنظومة تشفير الكلام

د. صديق يوسف أمين
أستاذ
قسم هندسة الحاسبات وتكنولوجيا المعلومات
الجامعة اتكنولوجية

د. عباس أحمد الشالجي
أستاذ مساعد
قسم الهندسة الالكترونية والاتصالات
كلية الهندسة – جامعة النهرين

مهند ضياء البياتي
مدرس مساعد
قسم هندسة الحاسبات – كلية الهندسة
جامعة بغداد

الخلاصة

في هذا البحث تم عرض و دراسة تشفير اشارة الكلام اعتمادا على خوارزمية رايندل لأن الخوارزمية تحقق معظم المتطلبات الأمنية في التطبيقات الحديثة. تم اقتراح ومحاكاة برامجيات بواسطة لغة ال MATLAB لتشفير ملفات الكلام من خلال تسجيل الإشارات في الحاسبة.

تم اجراء قياسات من نوع الموضوعية والشيئية بواسطة قياس نسبة الإشارة للضوضاء الجزئية للطيف لفحص اداء المنظومة. في هذه القياسات تم قياس قيمة المفهومية المتبقية والجودة في الإشارة المشفرة والمسترجعة كما تم تقييم النتائج.

أخيرا تم تنفيذ الكيان المادي لمنظومة التشفير المقترحة باستخدام المعالج TMS320-C30 كما تم حساب متطلبات التنفيذ في الزمن الحقيقي لمنظومة تشفير الكلام وقد تم التوصل الى تقارب الأداء بين المنظومة المحاكاة والمنظومة المنفذة عمليا.