Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Chaotic digital cryptosystem using serial peripheral interface protocol and its dsPIC implementation*

Rodrigo MÉNDEZ-RAMÍREZ[1], Adrian ARELLANO-DELGADO[2], César CRUZ-HERNÁNDEZ[†‡1], Fausto ABUNDIZ-PÉREZ[3], Rigoberto MARTÍNEZ-CLARK[1]

[1]*Electronics and Telecommunications Department, Scientific Research and Advanced Studies Center of Ensenada, Ensenada 22860, Mexico*
[2]*CONACYT-UABC Engineering, Architecture, and Design Faculty, Autonomous University of Baja California, Ensenada 22860, Mexico*
[3]*Engineering, Architecture and Design Faculty, Autonomous University of Baja California, Ensenada 22860, Mexico*
[†]E-mail: ccruz@cicese.mx

Received June 17, 2016; Revision accepted Jan. 23, 2017; Crosschecked Feb. 15, 2018

Abstract: The current massive use of digital communications demands a secure link by using an embedded system (ES) with data encryption at the protocol level. The serial peripheral interface (SPI) protocol is commonly used by manufacturers of ESs and integrated circuits for applications in areas such as wired and wireless communications. We present the design and experimental implementation of a chaotic encryption and decryption algorithm applied to the SPI communication protocol. The design of the chaotic encryption algorithm along with its counterpart in the decryption is based on the chaotic Hénon map and two methods for blur and permute (in combination with DNA sequences). The SPI protocol is configured in 16 bits to synchronize a transmitter and a receiver considering a symmetric key. Results are experimentally proved using two low-cost dsPIC microcontrollers as ESs. The SPI digital-to-analog converter is used to process, acquire, and reconstruct confidential messages based on its properties for digital signal processing. Finally, security of the cryptogram is proved by a statistical test. The digital processing capacity of the algorithm is validated by dsPIC microcontrollers.

Key words: Chaotic systems; Statistical tests; Embedded systems; dsPIC microcontroller; Serial peripheral interface (SPI) protocol
https://doi.org/10.1631/FITEE.1601346 **CLC number:** TP309.7

1 Introduction

Embedded systems (ESs) are increasingly used to develop and integrate a large number of electronic devices for wired and wireless communication applications. Depending on the application, it is more convenient and economical to use a communication protocol that allows the use of fewer communication lines and in which the transmitted information is secure (Barr and Massa, 2006). Within the communication protocols there are the inter-integrated circuit (I²C) and the serial peripheral interface (SPI) protocols. Both are suitable for data transfer among peripherals with low or medium speed for communications of electronic integrated circuits (ICs). The I²C protocol was created by Philips Semiconductors (1995, 2003), and the SPI protocol was created by Motorola Inc. (2003). Both types of protocols coexist in modern digital electronic systems, and probably these protocols will continue complementing each other in the future because they are inexpensive for implementation (Oudjida et al., 2009).

The SPI protocol allows communication among peripherals and electronic devices. The mode of connection between these devices can be assigned in master and slave modes. The devices are controlled,

‡ Corresponding author

* Project supported by the CONACYT, México (No. 166654)

ORCID: Rodrigo MÉNDEZ-RAMÍREZ, http://orcid.org/0000-0002-2593-8658

for example, by using a microcontroller as the main unit in master-mode to control a digital-to-analog converter (DAC) as a peripheral in slave-mode (Leens, 2009). The SPI protocol has been adopted by many manufacturers of ICs in wired and wireless communications because SPI protocols require fewer communication lines and their implementations are low-cost. Examples are digital signal processing (DSP), digital image, digital audio, security, and some specific applications such as global positioning system (GPS), gyroscope, universal serial bus (USB) interface, controller area network (CAN) interface, Ethernet controller, radio transceiver module, Bluetooth communication, advanced touchscreen controller, DAC, analog-to-digital converter (ADC), audio codec chip, optical finger sensor module, and data storage with serial flash embedded memory. Some implementations on ESs using the SPI protocol are reported in the literature, for example, the control system design for agricultural vehicles together with CAN protocol (Tumenjargal et al., 2013), secure communications with data transmission based on encryption algorithms using SPI protocol (Jyothi et al., 2012), interfaces with SPI and I^{2}C protocols implemented with field programmable gate array (FPGA) (Oudjida et al., 2009), nano-power wireless wake-up receiver (Marinkovic and Popovici, 2011), and applications of SPI protocol for an integrated amplifier with passive neutralization of myoelectric interference from neural recording tripoles (Demosthenous et al., 2013).

On the other hand, modern cryptography includes different encryption techniques, for example, algorithms for image encryption like triple data encryption algorithm (3DES), advanced encryption standard (AES), and international data encryption algorithm (IDEA) (Muhaya et al., 2009). The literature reported the use of chaotic maps and hyperchaotic systems to encrypt information for different applications such as secure e-mail communication (Aguilar-Bustos et al., 2010), chaos-based cryptosystems on DSP (Guglielmi et al., 2009; Rhouma and Belghith, 2011), experimental network synchronization via plastic optical fiber (Arellano-Delgado et al., 2012), robust synchronization of chaotic systems using sliding mode and feedback control (Li et al., 2014), and recent RGB image encryption algorithms (Murillo-Escobar et al., 2015a; Liu et al., 2016).

To implement chaotic systems in an ES, we need robust processors based on DSPs or FPGAs because their architectures are robust, and these microprocessors have more processing capacity (Azzaz et al., 2013). In this sense, the microcontrollers in ESs represent an economical alternative for encryption applications using chaos to generate pseudo-random sequences (Murillo-Escobar et al., 2015a). The dsPIC microcontroller is an alternative for implementation at low cost, and the architecture and properties of DSP allow the performance of mathematical calculations with high precision. These microcontrollers gather sufficient conditions as a central part of an ES (Jasio, 2008). dsPIC microcontrollers have been used in some studies (Siddiqui et al., 2015; Uriz et al., 2016).

Microchip Technology Inc. is a manufacturer of dsPIC microcontrollers. The dsPICs were optimized by hardware and software for DSP applications and data encryption, for example, symmetric encryption keys, anti-symmetric encryption security testing for Web access, XML transactions security, virtual private networks, transfer and safety calibration data storage with cryptographic algorithms 3DES and AES (128 bits) (Microchip Technology Inc., 2006, 2018). However, studies on data encryption at the level of SPI protocol where the messages are hidden using algorithms based on chaos, and studies on the performance of these algorithms in ESs (in low cost implementation) for DSP applications have not yet been reported in the literature.

In this study, we present the design and experimental implementation of a chaotic encryption and decryption algorithm applied to the SPI communication protocol with DSP studies for the acquisition, processing, and reconstruction of messages. The implementation of the ES is developed using two dsPIC microcontrollers with low cost of implementation. The dsPIC transmitter contains the encryption algorithm and the dsPIC receiver contains the decryption algorithm in reverse order of the encryption algorithm. For the design of the encryption and decryption algorithms, the use of a chaotic discrete map in combination with DNA sequences, and two diffusion and confusion methods to hide the information are proposed. The properties of SPI protocol are used to establish the link and achieve synchronization of dsPIC microcontrollers. For DSP application, three experimental tests, sensitivity of secret keys, function generator, and voice recording, are performed. The experimental results are proved using an external DAC. In addition, statistical tests are carried out on a cryptogram to prove that the SPI protocol generates pseudo-random (PR) sequences.

2 Description of the embedded chaotic cryptosystem

2.1 SPI protocol

The SPI protocol is used for communication of the ESs because it has an easy configuration, fast serial data bus transmission, and low-quantity lines for connecting other peripherals (ICs or devices). Many manufacturers of microcontrollers and microprocessors have adopted the SPI protocol to be implemented directly by hardware from one or three dedicated ports. The data transmission communication protocol is full-duplex and the operation modes are master or slave. The SPI protocol specifies four wires connected using external pins. One of these pins is the master-output to slave-input (MOSI) used to connect from master to slave devices. This pin is also referred to as serial data output (SDO). The pin master-input to slave-output (MISO) is used to connect from master to slave devices. In addition, this pin is referred to as serial data input (SDI). The pin serial clock (SCK) is used to synchronize the data transfer from master to slave devices. The slave select (SS) is the pin that selects from master to slave devices (Motorola Inc., 2003).

The dsPIC devices feature SPI modules. This SPI module is configured using internal registers that consist of mainly 16-bit shift registers. This shift register (SPI*z*SR where *z* indicates the number of the SPI modules) is used to shift data in and out from the buffer register (SPI*z*BUF). The control register (SPI*z*CON) configures the SPI module. The statistical register (SPI*z*STAT) shows the status conditions of the operations of the SPI module, where the flag SPIRBF allows verification if the reception of a word with 16 bits from another external device is accomplished (Microchip Technology Inc., 2004).

In this study, we propose the SPI protocol configuration with 16 bits using two dsPICs and external DACs. To program the dsPICs we use the Mikroc Pro as the dsPIC compiler (Mikroelectronika, 2014). Fig. 1 shows the complete block diagram of the embedded chaotic cryptosystem transmitting and receiving encrypted messages using the SPI protocol. The dsPIC transmitter (configured in master-mode) first processes the initial message $m(t)$ with confidential information. This message is encrypted and transmitted using the SPI protocol. The receiver dsPIC is considered mainly as a transceiver where the SPI configuration is switched from slave-mode to master-mode. It first receives and decrypts $m(t)$, and after that it transmits the decrypted message $m'(t)$ to an external DAC (configured as slave-mode), and the message $m'(t)$ is rebuilt.

2.2 Hardware description of implemented ES

The benefits of the dsPIC microcontrollers are

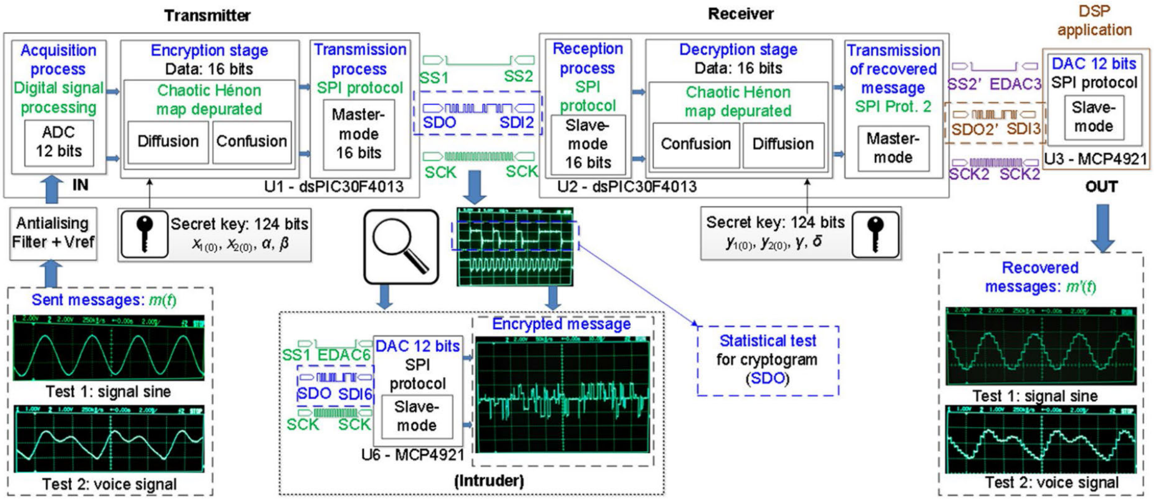


Fig. 1 Block diagram of the embedded cryptosystem (references to color refer to the online version of this figure)

used to obtain and prove the results using the SPI communication protocol with DSP applications.

Table 1 shows the description of hardware assigned to an ES to connect ICs through the SPI protocol.

2.3 Generating PR sequences

To demonstrate the experimental results, Fig. 2 shows the design of an alternative ES for a discrete chaotic system with two dimensions (Méndez-Ramírez et al., 2015). The SPI configuration of the dsPIC U1 is in master-mode, and the DACs U3 and U4 are in slave-mode. The algorithm of the chaotic map is processed by U1. The states $x_{1(n)}$ and $x_{2(n)}$ show the chaotic behaviors reproduced by U3 and U4, respectively. The results of processing algorithms, to encrypt and decrypt, are implemented in transmitter U1 and receiver U2.

To generate PR sequences we consider the chaotic Hénon map (Hénon, 1976):

$$\begin{cases} x_{1(n+1)} = 1 - \alpha x_{1(n)}^2 + x_{2(n)}, \\ x_{2(n+1)} = \beta x_{1(n)}, \end{cases} \tag{1}$$

Table 1 Hardware description of embedded cryptosystem

Unit	IC description	Function
U1	Microcontroller dsPIC30F4013	Transmitter
U2	Microcontroller dsPIC30F4013	Receiver (transceiver)
U3	DAC MCP4921	State representation $x_{1(n)}$
U4	DAC MCP4921	State representation $x_{2(n)}$
U5	DAC MCP4921	Recovered message $m'(t)$
U6	DAC MCP4921	Cryptogram representation (intruder)

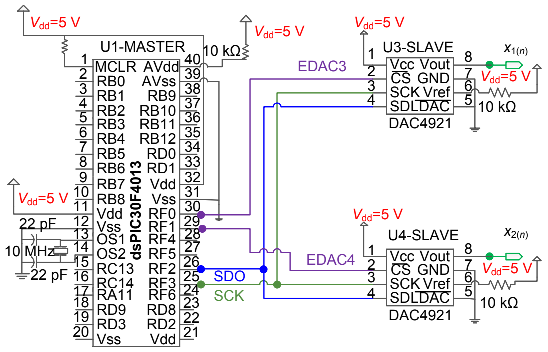


Fig. 2 An ES to represent the states ($x_{1(n)}$ and $x_{2(n)}$) of the discrete chaotic map

where α and β are bifurcation parameters.

We use parameters α=0.5, β=0.3, and initial conditions $x_{1(0)}$=0.5 and $x_{2(0)}$=0.1.

The Hénon map is implemented on the ES (Fig. 2). The phase plane of states $x_{2(n)}$ vs. $x_{1(n)}$ is shown in Fig. 3, where unfortunately the data concentration and dispersion of the strange attractor is homogeneous.

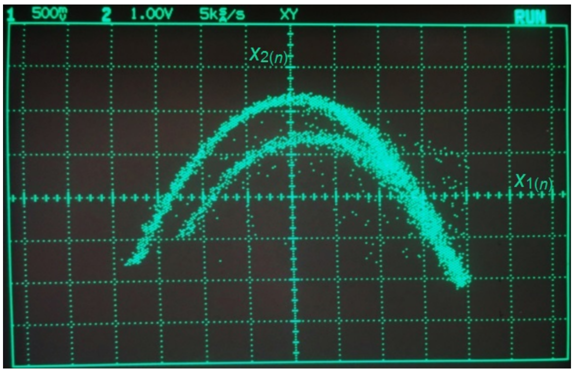


Fig. 3 Phase plane representation $x_{2(n)}$ vs. $x_{1(n)}$ of the chaotic Hénon map (Eq. (1))

We propose a depurated Hénon map to improve the heterogeneity in the concentration and dispersion of data, where the values of states $x_{1(n)}$ and $x_{2(n)}$ are converted to values between 0 and 1. The first three and four integer numbers in the states $x_{1(n)}$ and $x_{2(n)}$ are removed respectively to obtain only the decimal numbers on Hénon map (Eq. (1)), and the depurated Hénon map is obtained,

$$\begin{cases} x_{1\mathrm{m}(n)} = 1000(1.3 + x_{1(n)}) \\ \qquad - \operatorname{floor}(1000(1.3 + x_{1(n)})), \\ x_{2\mathrm{m}(n)} = 10\,000(0.5 + x_{2(n)}) \\ \qquad - \operatorname{floor}(10\,000(0.5 + x_{2(n)})), \end{cases} \tag{2}$$

where $x_{1\mathrm{m}(n)}$, $x_{2\mathrm{m}(n)} \in [0, 1]$ are the states of the depurated chaotic map, and the subscript 'm' denotes that the chaotic discrete standard Hénon map (SHM) (Eq. (1)) is modified. Two depurated chaotic maps with similar features were reported by Murillo-Escobar et al., (2015b) and Liu et al. (2016). Fig. 4 shows the phase plane $x_{2\mathrm{m}(n)}$ vs. $x_{1\mathrm{m}(n)}$ with the depurated attractor on U1, where there is a change in the data dispersion of the chaotic depurated Hénon map (DHM) (Eq. (2)).

To show the improved data distribution of chaotic DHM with respect to SHM, the histogram comparison is shown in Fig. 5. Fig. 5a shows the state $x_{1(n)}$ of SHM (Eq. (1)), and Fig. 5b shows the state $x_{1\mathrm{m}(n)}$ of DHM (Eq. (2)) with n=20 000, where the data distribution is clearly improved.

Finally, the states $x_{1\mathrm{m}(n)}$ and $x_{2\mathrm{m}(n)}$ of DHM (Eq. (2)) are considered to generate two PR sequences in the encryption and decryption stages, and each number of the PR sequences is generated in one iteration n in the U1 and U2 algorithms.

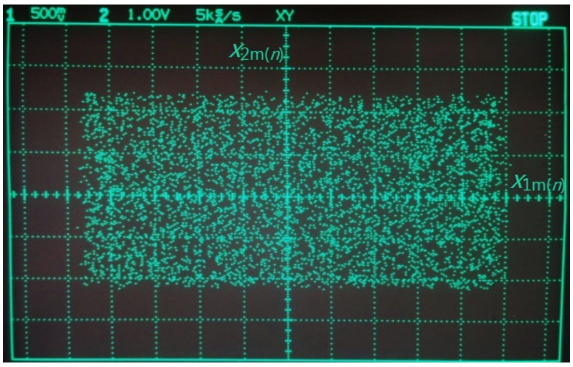


Fig. 4 Phase plane representation $x_{2\mathrm{m}(n)}$ vs. $x_{1\mathrm{m}(n)}$ of the chaotic depurated Hénon map (Eq. (2)) (n=20 000)

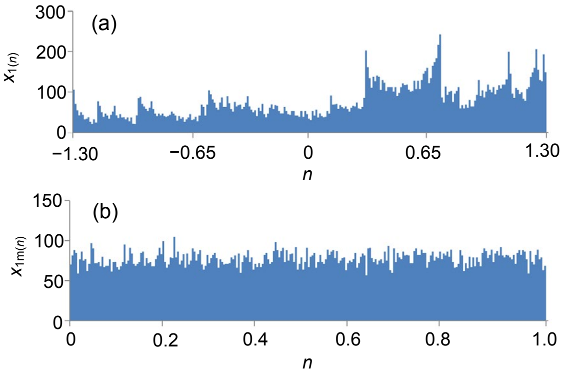


Fig. 5 Histograms: (a) state $x_{1(n)}$ of SHM; (b) state $x_{1\mathrm{m}(n)}$ of DHM

2.4 Secret key definition

The secret key is symmetric, and is defined by initial conditions and parameters of SHM and DHM. The key must be secure, and it should be subject to a value greater than 2^{100} (Alvarez and Li, 2006).

The standard IEEE-754 with extension AN575 to 32 bits for the numerical representation of dsPICs is used, as reported by Microchip Technology Inc. (1997). Table 2 shows the numerical representation of floating point formats with operating intervals that dsPICs microcontrollers use as standard.

Table 2 Microchip standard IEEE-754 for 32 bits

Value	*eb*	*e*	$\|A^*\|f^*$	Decimal
MAX	0xFF	128	7FFFFF	$6.805\,646\,930\times10^{38}$
MIN	0x01	−128	000000	$1.175\,494\,35\times10^{-38}$

eb is a biased exponent; *e* is the exponent or characteristic; f^* is the fraction or mantissa; $|A^*|$ is the decimal number

According to the intervals in Table 2, there are a maximum 0xFF7FFFFF and a minimum 0x01000000 in the hexadecimal base. The maximum interval represented in the decimal base is 4 286 578 687 which is equivalent to subtracting 2^{32} less 2^{23}, the result being 2^{31}. By performing the calculations, representation of four numbers is obtained as

$$2^{31}\cdot2^{31}\cdot2^{31}\cdot2^{31}=2^{124}>2^{100}. \qquad (3)$$

A space of keys can be represented in numerical base. We propose the following hexadecimal notation as the 124-bit secret key: FF7F FFFF FF7F FFFF FF7F FFFF FF7F FFFF.

The notation of Microchip standard IEEE-754 for 32 bits is used to build the key space with the parameters and initial conditions of SHM. Table 3 shows the build of the secret key for 124 bits where float decimal and hexadecimal notations are used.

The secret key is built with one slight variation of the parameters and initial conditions of SHM. Table 4 shows the set of three keys using parameters and different initial conditions of SHM.

3 Transmission process

The transmission process of confidential message $m(t)$ is divided into three stages: acquisition

Table 3 Notation of secret key

Secret key notation	Parameters and initial conditions of SHM			
	α	β	$x_{1(0)}$	$x_{2(0)}$
Float decimal	1.4	0.3	1.112 121 2	0.465 465 5
Float 32 bits (IEEE)	3F B3 33 33	3E 99 99 9A	3F 8E 59 FD	3E EE 51 7E

Table 4 Set of secret keys

Secret key	Notation	Parameters and initial conditions of SHM			
		α	β	$x_{1(0)}$	$x_{2(0)}$
K_1	Float decimal	1.4	0.3	1.1121213	0.4654655
	Float 32 bits (IEEE)	3F B3 33 33	3E 99 99 9A	3F 8E 59 FE	3E EE 51 7E
K_2	Float decimal	1.4	0.3	1.1121212	0.4654659
	Float 32 bits (IEEE)	3F B3 33 33	3E 99 99 9A	3F 8E 59 FD	3E EE 51 8E
K_3	Float decimal	1.4	0.298046886920929	1.1121212	0.4654655
	Float 32 bits (IEEE)	3F B3 33 33	3E 98 99 9A	3F 8E 59 FD	3E EE 51 7E

process, encryption stage, and transmission of encrypted message. Fig. 6 illustrates the description of the transmission process.

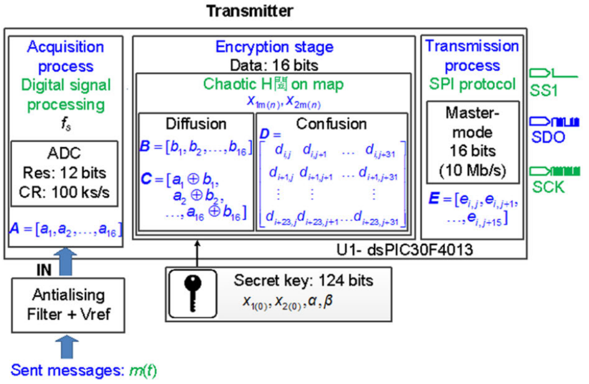

Transmitter
Acquisition process
Digital signal processing
f_s
ADC
Res: 12 bits
CR: 100 ks/s
$A=[a_1, a_2, \ldots, a_{16}]$
Encryption stage
Data: 16 bits
Chaotic H闘 on map
$x_{1m(n)}, x_{2m(n)}$
Diffusion
$B=[b_1, b_2, \ldots, b_{16}]$
$C=[a_1 \oplus b_1, a_2 \oplus b_2, \ldots, a_{16} \oplus b_{16}]$
Confusion
$D=\begin{bmatrix} d_{i,j} & d_{i,j+1} & \cdots & d_{i,j+31} \\ d_{i+1,j} & d_{i+1,j+1} & \cdots & d_{i+1,j+31} \\ \vdots & \vdots & & \vdots \\ d_{i+23,j} & d_{i+23,j+1} & \cdots & d_{i+23,j+31} \end{bmatrix}$
Transmission process
SPI protocol
Master-mode
16 bits
(10 Mb/s)
$E=[e_{i,j}, e_{i,j+1}, \ldots, e_{i,j+15}]$
SS1
SDO
SCK
U1- dsPIC30F4013
IN
Antialising Filter + Vref
Secret key: 124 bits
$x_{1(0)}, x_{2(0)}, \alpha, \beta$
Sent messages: $m(t)$

Fig. 6 Block diagram of the transmitter (references to color refer to the online version of this figure)

Table 5 Bit description of register ADCBUF0

Bits 15–12 (RGP)	Bits 11–0
Elements a_{13} to a_{16}	The elements a_1 to a_{12} contain the message $m(t)$

3.1 Acquisition process

The feature of the ADC within U1 allows the acquisition of digital or continuous signals (Microchip Technology Inc., 2004). This dsPIC U1 has 13 ADC channels with a conversion rate (CR) of 100 ks/s and a 12-bit resolution for each channel. The conversion result of message $m(t)$ is contained in the ADCBUF0 register with 16 bits, where 12 bits corresponding to ADC results and the remaining 4 bits are used as registers of general purpose (RGP). We define vector $\boldsymbol{A}$ as the equivalent of ADCBUF0 where each element is expressed as a_k (k=1, 2, …, 16),

$$\boldsymbol{A}=[a_1, a_2, \ldots, a_{16}], \tag{4}$$

where $a_k \in [0, 1]$. The elements a_1 to a_{12} represent the analogic conversion result using the ADC of U1 and the elements a_{13} to a_{16} are regarded as RGP that can be used for channel communication, control instructions, etc. Table 5 shows the bit conversion of register ADCBUF0 and the representation of vector $\boldsymbol{A}$ where $m(t)$ is contained.

In traditional signal processing, to not lose information, the Nyquist-Shannon theorem shows that in uniformly sampling a signal we must sample at least twice (Nyquist, 1928; Shannon, 1949). The message size directly depends on the maximum frequency $f_{\max}$ of message $m(t)$, and sampling frequency f_s that the algorithm of U1 can process. These frequencies are related by

$$f_s \geq 2 f_{\max}. \tag{5}$$

An antialiasing filter is required at the inputs of the analog channels of U1, where the cut-off frequency of this filter f_c has to be fixed such that

$$f_{\max} < f_c < f_s. \tag{6}$$

3.2 Encryption stage

The message encryption considers two subprocesses: the so-called diffusion and confusion methods. We describe these methods using the states of the depurated chaotic Hénon map (Eq. (2)).

3.2.1 Diffusion method

The blur method consists in hiding the information of message $m(t)$ by performing a logic operation on the elements of vector $\boldsymbol{A}$. We use the state $x_{1m(n)}$ of the depurated chaotic map with a numerical arrangement to generate values from 0 to 65 535. The result is described as

$$O_{\mathrm{ex(U1)}} = 2^k x_{1\mathrm{m}(n)} = 2^k[1000(1.3 + x_{1(n)}) - \mathrm{floor}(1000(1.3 + x_{1(n)}))]. \tag{7}$$

Eq. (7) can also be defined as vector $\boldsymbol{B}$ with 16 equivalent elements:

$$\boldsymbol{B} = [b_1, b_2, ..., b_{16}]. \tag{8}$$

The OR-EX logic operation is calculated with each element of vectors $\boldsymbol{A}$ and $\boldsymbol{B}$. The information of vector $\boldsymbol{A}$ is hidden, and the results are defined by vector $\boldsymbol{C}$:

$$\boldsymbol{C} = [a_1 \oplus b_1, a_2 \oplus b_2, ..., a_{16} \oplus b_{16}] = [c_1, c_2, ..., c_{16}]. \tag{9}$$

3.2.2 Confusion method

This method consists in building a matrix with PR numbers extracted from a DNA database. The construction of the DNA matrix is described in Appendix. The matrix (Eq. (A1)) shows the DNA sequences with the details of positions. This matrix has a dimension of r×s, where r represents the number of rows and s represents the number of columns. The elements show disordered positions from 1 to higher values of k. These disordered elements are considered as one sequence. For each row r of the matrix (Eq. (A1)), the sequence of k elements is repeated twice.

For this process the same elements of the matrix (Eq. (A1)) are considered for the DNA sequence matrix referred to as $\boldsymbol{D}$. This matrix consists of r=24 sequences of s=32 elements, i.e.,

$$\boldsymbol{D} = \begin{bmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,s} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,s} \\ \vdots & \vdots & & \vdots \\ d_{r,1} & d_{r,2} & \cdots & d_{r,s} \end{bmatrix}. \tag{10}$$

The confusion algorithm consists in permuting the position of k=16 elements of vector $\boldsymbol{C}$. We calculate a numerical arrangement from states of DHM (Eq. (2)), and these states define the initial coordinates to obtain a new vector with 16 new positions permuted from Eq. (10). The initial coordinates are described as

$$\begin{cases} i = x_{1\mathrm{m}(n)} r = [1000(1.3 + x_{1(n)}) - \mathrm{floor}(1000(1.3 + x_{1(n)}))]r, \\ j = x_{2\mathrm{m}(n)} \dfrac{s}{2} = [10\ 000(0.5 + x_{2(n)}) - \mathrm{floor}(10\ 000(0.5 + x_{2(n)}))]\dfrac{s}{2}, \end{cases} \tag{11}$$

where i, j are integers, and i={0≤$x_{1\mathrm{m}(n)}$≤1}∈[1, r] and j={0≤$x_{2\mathrm{m}(n)}$≤1}∈[1, s] define the rows and columns of the matrix $\boldsymbol{D}$, respectively. The initial coordinates of $d_{i,j}$ allow the finding of the initial position of a sequence up to the complete 16 positions with the last referred to as $d_{i,j+15}$,

$$\boldsymbol{D} = \begin{bmatrix} d_{i,j} & d_{i,j+1} & \cdots & d_{i,j+31} \\ d_{i+1,j} & d_{i+1,j+1} & \cdots & d_{i+1,j+31} \\ \vdots & \vdots & & \vdots \\ d_{i+23,j} & d_{i+23,j+1} & \cdots & d_{i+23,j+31} \end{bmatrix}. \tag{12}$$

Finally, the confusion process is calculated by Eq. (12), and the result is contained in the new row vector $\boldsymbol{E}$. The first element e_j of vector $\boldsymbol{E}$ contains the first element c_1 of vector $\boldsymbol{C}$ that is referred to as the first position of element $d_{i,j}$ of vector $\boldsymbol{D}$ until 16 elements are completed, and the last element c_{16} corresponds to the element e_{j+15} that is referred to as the position $d_{i,j+15}$ of vector $\boldsymbol{D}$,

$$\boldsymbol{E} = [c_1, c_2, ..., c_{16}] = [d_{i,j}, d_{i,j+1}, ..., d_{i,j+15}] = [e_j, e_{j+1}, ..., e_{j+15}]. \tag{13}$$

3.3 Transmission of encrypted messages

The encryption algorithm is processed on U1. The encrypted information of $m(t)$ is contained in row vector $\boldsymbol{E}$, where the 16 elements are equivalent to one word of 16 bits, and when this word is transmitted it corresponds to one iteration n.

Now, we configure the SPI protocol in master-mode. U1 is connected with U2 using the pins. SS1 is connected with SS2, SDO is connected with SDI2, and SCK is the common clock to synchronize the dsPICs. Finally, the word contained in vector $\boldsymbol{E}$ is transmitted from U1 to U2.

4 Cryptogram

In this section, we describe the statistical tests to prove the hypothesis that $m(t)$ is secure from the observer standpoint. To show whether the proposed algorithm reproduces PR sequences, we perform the statistical tests to the cryptogram extracted from U1.

Fig. 7 shows the block diagram of an ES where an intruder intervenes in the SPI protocol using an external DAC U6. U1 is connected with U6 using the following pins: SS1 connected with EDAC6, SDO connected with SDI6, and SCK is the common clock to synchronize the dsPIC with the external DAC. The rebuilt encrypted message $m(t)$ is obtained, but the intruder cannot read the data.

We present test results of the behavioral analysis of the sequence generator obtained from the proposed encryption algorithm. Here, we consider encrypted binary sequences with length n=20 000 bits generated from U1. The binary sequence is read and stored by a computer after we calculate the statistical tests by MATLAB. We perform statistical tests to check whether the encrypted algorithm generates sufficiently secure PR sequences (Menezes et al., 1996; Yalcin et al., 2004; Fúster et al., 2012). Table 6 shows the results of five tests. These results demonstrate that the proposed encryption algorithm successfully passes all the statistical tests and the algorithm can generate PR sequences.

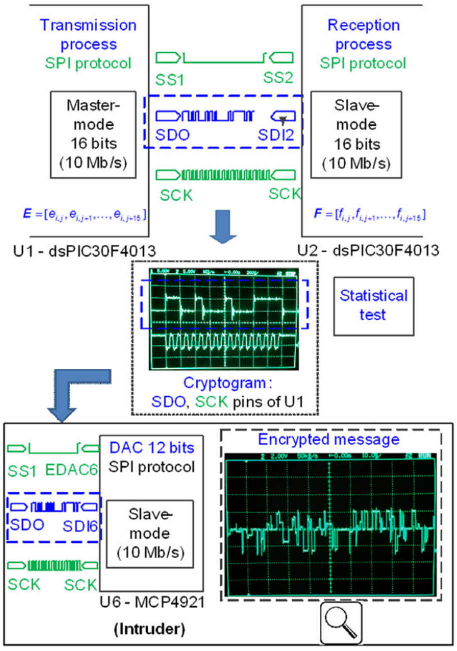


Fig. 7 Block diagram of an ES, where an intruder intervenes the cryptogram generated from U1 to U2 using U6 (references to color refer to the online version of this figure)

5 Reception process

The reception process is the same as that for the transmission process (Section 3), but the diffusion and confusion processes are performed in inverted stages. The processes are divided into three stages: reception of encrypted message, decryption process, and transmission of recovered message (Fig. 8).

Table 6 Statistical test results

Test	Proposed system	Required values	Parameter
Frequency test X_1	1.6562	<3.8415	–
Serial test X_2	3.0873	<5.9915	–
Poker test X_3	10.870 08	<14.0671	m=3
	17.2160	<24.9958	m=4
	40.5280	<44.9853	m=5
	71.750	<82.5287	m=6
Runs test X_4	6.1505	<9.4877	k=3
	6.9181	<12.5916	k=4
	8.7414	<15.5073	k=5
	11.9391	<18.3070	k=6
Autocorrelation test X_5	−0.3960	$-1.96<X_5<1.96$	–

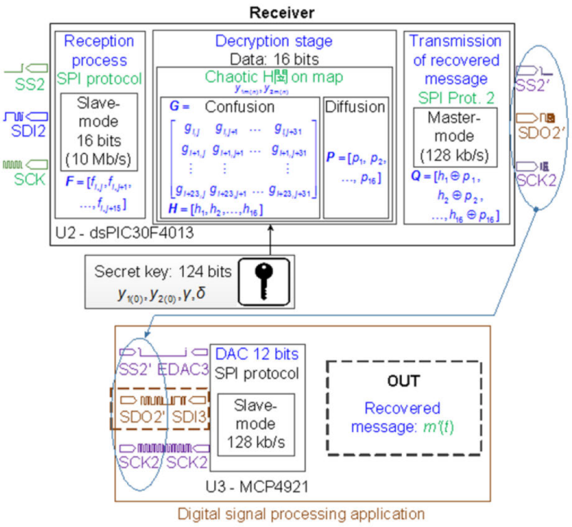

Receiver
SS2
SDI2
SCK
Reception process
SPI protocol
Slave-mode
16 bits
(10 Mb/s)
$F=[f_{i,j}, f_{i,j+1}, \dots, f_{i,j+15}]$
Decryption stage
Data: 16 bits
Chaotic H閱 on map
$y_{1m(n)}, y_{2m(n)}$
$G=$ Confusion
$\begin{bmatrix} g_{i,j} & g_{i,j+1} & \cdots & g_{i,j+31} \\ g_{i+1,j} & g_{i+1,j+1} & \cdots & g_{i+1,j+31} \\ \vdots & \vdots & & \vdots \\ g_{i+23,j} & g_{i+23,j+1} & \cdots & g_{i+23,j+31} \end{bmatrix}$
$H=[h_1, h_2, \dots, h_{16}]$
Diffusion
$P=[p_1, p_2, \dots, p_{16}]$
Transmission of recovered message
SPI Prot. 2
Master-mode
(128 kb/s)
$Q=[h_1 \oplus p_1, h_2 \oplus p_2, \dots, h_{16} \oplus p_{16}]$
SS2′
SDO2′
SCK2
U2 - dsPIC30F4013
Secret key: 124 bits
$y_{1(0)}, y_{2(0)}, \gamma, \delta$
SS2' EDAC3
SDO2' SDI3
SCK2 SCK2
DAC 12 bits
SPI protocol
Slave-mode
128 kb/s
U3 - MCP4921
OUT
Recovered message: $m'(t)$
Digital signal processing application

Fig. 8 Block diagram of receiver (references to color refer to the online version of this figure)

5.1 Reception of the encrypted message

We use the properties of the SPI protocol to synchronize U1 with U2. The dsPIC U2 is always waiting for the first word with 16 bits that contains the 16 elements of vector $\boldsymbol{E}$ from U1, which is the first iteration n. At first, the SPI configuration of U2 is in slave-mode, because it is waiting for the encrypted message from U1. Then the SPI configuration changes to master-mode because the decrypted message $m'(t)$ is transmitted to U5.

The decryption algorithm of U2 is configured to constantly check if the word of 16 bits from U1 is successfully received. We consider z=1 in the configuration of the SPI module on U2. The flag SPIRBF of the internal register SPI1STAT allows verification on whether reception of a word with 16 bits from U1 is accomplished. Finally, vector $\boldsymbol{F}$ contains the reception of the encrypted message:

$$\boldsymbol{F}=[f_j, f_{j+1}, \ldots, f_{j+15}]. \tag{14}$$

5.2 Decryption stage

We use the same proposed methods with the same features as in the encryption stage (Section 3.2), but in the reverse order. The secret key is symmetric, and therefore it is mandatory that the U2 algorithm contains the same initial condition and parameters of the chaotic map (Eq. (1)).

5.2.1 Reverse process of confusion method

To generate PR sequences we use the SHM and DHM, but the notations of state variables are changed to differentiate the transmission and reception stages. Define the state variables $y_{1(n)}$, $y_{2(n)}$, and parameters γ, δ of the new Hénon map used in this stage as

$$\begin{cases} y_{1(n+1)}=1-\gamma y_{1(n)}^2+y_{2(n)}, \\ y_{2(n+1)}=\delta y_{1(n)}. \end{cases} \tag{15}$$

The depurated states are referred to as $y_{1\mathrm{m}(n)}$ and $y_{2\mathrm{m}(n)}$. The initial coordinates are described as

$$\begin{cases} l=y_{1\mathrm{m}(n)}v=[1000(1.3+y_{1(n)}) \\ \qquad -\mathrm{floor}(1000(1.3+y_{1(n)}))]v, \\ u=y_{2\mathrm{m}(n)}\dfrac{w}{2}=[10\ 000(0.5+y_{2(n)}) \\ \qquad -\mathrm{floor}(10\ 000(0.5+y_{2(n)}))]\dfrac{w}{2}, \end{cases} \tag{16}$$

where l and u are integers, and l={0≤$y_{1\mathrm{m}(n)}$≤1}∈[1, v] and u={0≤$y_{2\mathrm{m}(n)}$≤1} with l∈[1, w] define the rows and columns respectively to determine the positions of the elements for the new DNA matrix $\boldsymbol{G}$. This matrix contains the same elements as matrix $\boldsymbol{D}$ (Eq. (10)). The initial coordinates $g_{l,u}$ allow the finding of the initial position of a sequence until 16 positions are completed and the last position is referred to as $g_{l,u+15}$,

$$\boldsymbol{G}=\begin{bmatrix} g_{l,u} & g_{l,u+1} & \cdots & g_{l,u+31} \\ g_{l+1,u} & g_{l+1,u+1} & \cdots & g_{l+1,u+31} \\ \vdots & \vdots & & \vdots \\ g_{l+23,u} & g_{l+23,j+1} & \cdots & g_{l+23,u+31} \end{bmatrix}. \tag{17}$$

Vector $\boldsymbol{F}$ contains $m(t)$, and the confusion process is performed using Eqs. (14)–(17). The new row vector $\boldsymbol{H}$ is obtained. The first element h_u of vector $\boldsymbol{H}$ contains the first element f_j of vector $\boldsymbol{F}$, and the last element f_{j+15} is referred to as h_{u+15},

$$\begin{aligned} \boldsymbol{H}&=[f_j, f_{j+2}, \ldots, f_{j+15}]=[g_{l,u}, g_{l,u+1}, \ldots, g_{l,u+15}] \\ &=[h_u, h_{u+1}, \ldots, h_{u+15}]. \end{aligned} \tag{18}$$

5.2.2 Reverse process of diffusion method

This method uses the SHM (Eq. (15)) and one state $y_{1\mathrm{m}(n)}$ of DHM (Eq. (16)). A numerical arrangement is calculated to generate values from 0 to 65 535 and the result is defined as

$$\begin{aligned} O_{\mathrm{ex(U1)}}=2^k y_{1\mathrm{m}(n)}=2^k[&1000(1.3+y_{1(n)}) \\ &-\mathrm{floor}(1000(1.3+y_{1(n)}))], \end{aligned} \tag{19}$$

where vector $\boldsymbol{P}$ defines the 16 equivalent elements,

$$\boldsymbol{P}=[p_1, p_2, \ldots, p_{16}]. \tag{20}$$

The OR-EX logic operation is calculated on each element of vector $\boldsymbol{H}$ with respect to vector $\boldsymbol{P}$. Finally, the decrypted message $m'(t)$ is found and the data are contained in vector $\boldsymbol{Q}$:

$$\begin{aligned} \boldsymbol{Q}&=[h_1\oplus p_1, h_2\oplus p_2, \ldots, h_{16}\oplus p_{16}] \\ &=[q_1, q_2, \ldots, q_{16}]. \end{aligned} \tag{21}$$

5.3 Transmission of the recovered message

Once the decrypted message is recovered, we configure the second SPI protocol using U2 to transmit $m'(t)$ for an external DAC U6. The hardware of dsPIC U2 has only one SPI port that enables the linking of U1 and U2 in slave-mode. It is necessary to enable another SPI port in master-mode to link U2 with U5. The Soft SPI library of Mikroc Pro for dsPIC compiler allows enabling another SPI port using a software and hardware configuration. Define the pins connection using the hardware from U2 to U5, where SS2′ is connected with EDAC3, SDO2 is connected with SDI3, and SCK2 is the common clock to synchronize the dsPICs with the DAC. Now, the decrypted message $m'(t)$ can be transmitted from U2 to U5 using the new SPI port. Fig. 8 shows the second SPI protocol of U2 connected with the external DAC U5.

6 Experimental results

Three tests are developed to determine the performance of the proposed ES: sensitivity of secret keys, analog signal reproduced by one function generator, and voice recording. Fig. 9 shows the schematic diagram of the proposed ES.

6.1 Time complexity of the algorithms

The time complexity (TC) is estimated by counting the number of elementary operations performed for the algorithms of U1, where an elementary operation takes a fixed amount of time to perform the running time of one iteration n (Sipser, 2006). To determine the performance of the proposed ES using TC for DSP applications, the number of iterations n that U1 generates in 1 s is calculated. The TC of U1 is calculated with the time period referred to as $T_{Q(\mathrm{U1})}$, and it is the reciprocal of the frequency referred to as $f_{Q(\mathrm{U1})}$:

$$f_{Q(\mathrm{U1})} = \frac{1}{T_{Q(\mathrm{U1})}}. \tag{22}$$

The encryption algorithm is implemented on U1. The TC of U1 is calculated using Eq. (22). Table 7 shows the experimental results of the algorithm of U1. The algorithm of U1 presents TC=350 μs and $f_{Q(\mathrm{U1})}$=2857 Hz. In our study, we consider $f_{Q(\mathrm{U1})}$ as n=2857 iterations per second. The processing capacity of U1 is calculated by inequality (5), where f_s=2857 Hz and $f_{\max}$=1429 Hz. This means that U1 can sample the message structure $m(t)$ until $f_{\max}$=1429 Hz.

In the case of U2, the minor TC is needed in all processes; the configuration of U2 is accelerated with the modification of the internal phase-locked loop (PLL). This allows processing to be faster in the decryption algorithm. Therefore, the TC of U2 must be less than that of U1,

$$T_{Q(\mathrm{U1})} > T_{Q(\mathrm{U2})}. \tag{23}$$

Table 8 shows the experimental results of the decryption algorithm.

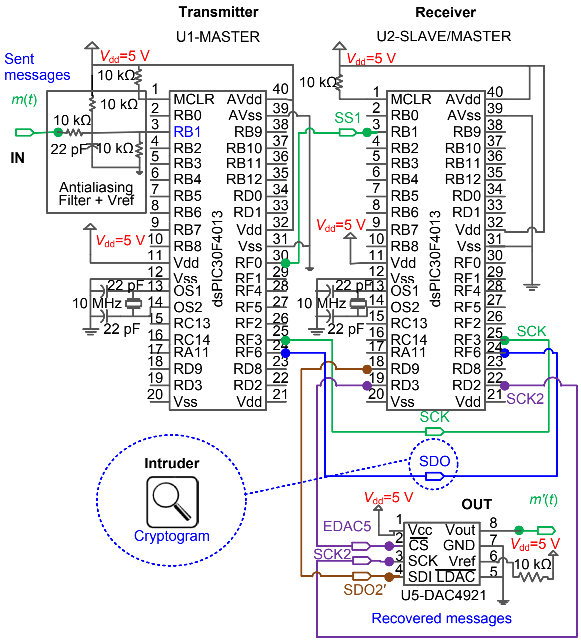


Fig. 9 Schematic diagram of the proposed ES (references to color refer to the online version of this figure)

Table 7 Time complexity of the algorithms of U1

Encryption algorithm	$T_{Q(\mathrm{U1})}$ (μs)	$f_{Q(\mathrm{U1})}$ (Hz)	Mathematical denotation
Acquisition process	10	100 000	***A***
Encryption stage	335	3205	***B***→***C***→***D***
Transmission of encrypted message	5	200 000	***E***
Total	350	2857	–

Table 8 Time complexity of the algorithms of U2

Decryption algorithm	$T_{Q(\text{U2})}$ (μs)	Mathematical denotation
Reception of encrypted message	5	$\boldsymbol{F}$
Decryption stage	160	$\boldsymbol{G}\rightarrow\boldsymbol{H}\rightarrow\boldsymbol{P}$
Transmission of recovered message	125	$\boldsymbol{Q}$
Total	290	–

According to the results shown in Tables 7 and 8, $T_{Q(\text{U1})}$ is larger than $T_{Q(\text{U2})}$. The condition (inequality (23)) to synchronized U1 and U2 is fulfilled. Therefore, according the processing capacity of U1, the proposed ES supports messages for maximum frequency until 1429 Hz.

6.2 Experimental results of the ES

The performance of the encryption and decryption algorithms is calculated using the TC on the ES. TC is obtained to characterize the message $m(t)$ according to $f_{Q(\text{U1})}$, f_s, and f_max. The hardware for the acquisition of one message $m(t)$ is shown in Fig. 9. Define pin RB1 as the input of the ADC channel. The reference of the ADC channel is fixed between voltages AV_ss=0 V and AV_dd=5 V, where AV_ss and AV_dd are the interval-voltages of the internal ADC of the dsPIC. The external voltage divisor is fixed with $V_\text{ref}=V_\text{dd}/2=$ 2.5 V, and an antialiasing filter with f_c=2 kHz is fixed according to inequality (6). Two kinds of messages are considered to carry out three experimental tests on the ES. The first message is referred to as $m_1(t)$. One function generator to reproduce one sine signal is considered for the experimental tests, and it is defined as

$$m_1(t) = 2\sin(2\pi ft + \theta) + V_\text{ref}, \tag{24}$$

where f is the frequency delimited by $f \leq f_\text{s}$ and θ is the phase angle.

The second message is referred to as $m_2(t)$, and external audio devices on the ES are considered to carry out the sound test to obtain the recorded voice message. The recovered massages of $m_1(t)$ and $m_2(t)$ are referred to as $m_1'(t)$ and $m_2'(t)$, respectively.

6.2.1 First test: sensitivity of secret keys

The sensitivity test is carried out using the secret keys K_1, K_2, and K_3 in Table 4. The secret keys are entered by programming directly the algorithms of U1 and U2, but the secret keys can be registered using an external keyboard connected to dsPICs U1 and U2. Table 9 shows the combination of secret keys K_1, K_2, and K_3 to carry out the sensitivity test. The message $m_1(t)$ is reproduced (Eq. (24)) for f=20 Hz on the ES.

Table 9 Sensitivity test of secret keys on ES

Case	Secret key on U1	Secret key on U2
1	K_1	K_1
2	K_1	K_2
3	K_1	K_3

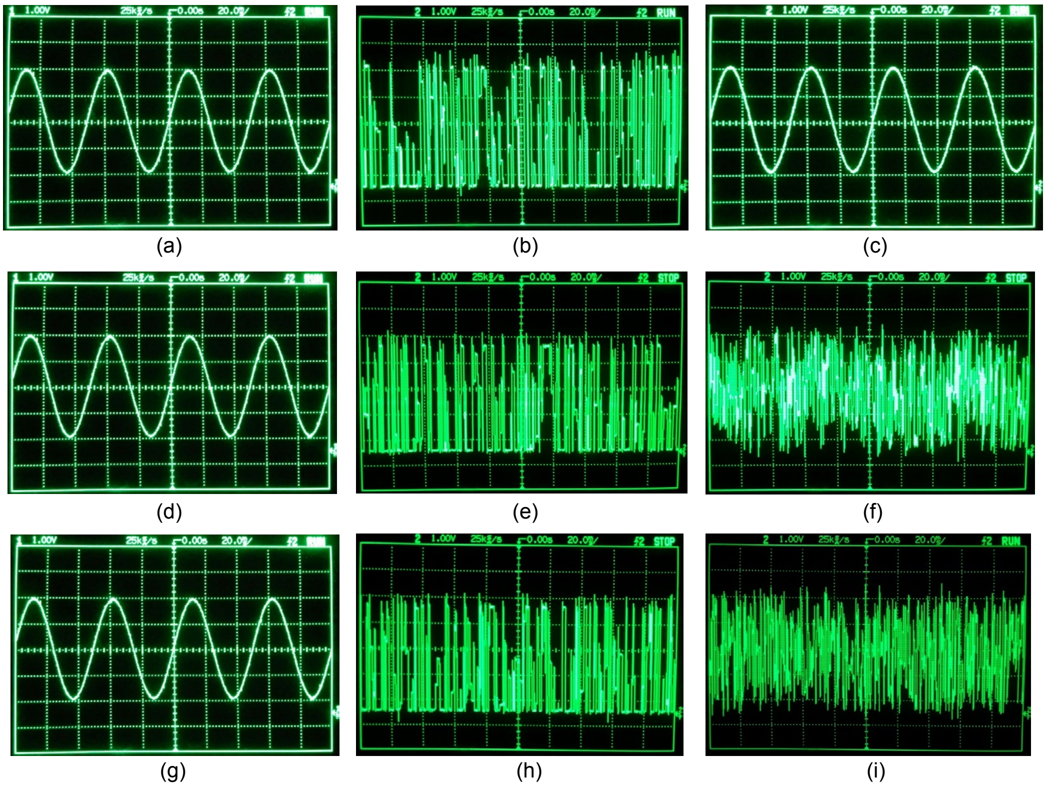
Fig. 10 shows the result of the sensitivity test among the set of three secret keys. We conclude that the definition of secret key supports the hypothesis that its construction of 124 bits is secure because of slight variations of the initial conditions and parameters of secret keys, and the result of the recovered message $m_1'(t)$ is different from the original message $m_1(t)$ for different secret keys. Similar results were reported by Liu et al. (2016).

6.2.2 First test: sine signal as message $m_1(t)$

We use the same sine signal described by Eq. (24) to reproduce $m_1(t)$ for different parameters of f. In this test, the same secret key K_1 is considered for U1 and U2. Fig. 11 shows the results and description of the message sent $m_1(t)$ and message recovered $m_1'(t)$ on the ES. Fig. 11a shows three signals: messages $m_1(t)$, $m_1'(t)$, and the error signal $e_1(t)$ between them that is equivalent to $m_1(t)–m_1'(t)$, with f=10 Hz. Fig. 11b shows the phase plane of $m_1(t)$ vs. $m_1'(t)$ with f=10 Hz. Fig. 11c shows the phase difference between $m_1(t)$ and $m_1'(t)$, where the segmented line represents a zoom of 10 times of the sine signal using f=210 Hz. Fig. 11d shows the phase plane of $m_1(t)$ vs. $m_1'(t)$ where θ is increased to π/2 with each f=210 Hz and θ=π/2 as the phase difference.

6.2.3 Second test: voice message $m_2(t)$

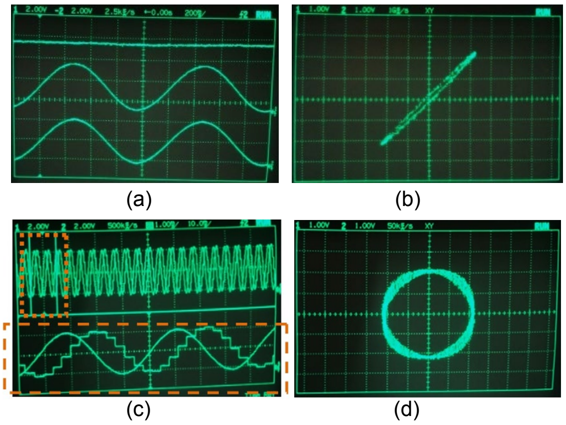
Fig. 12 shows the ES with external devices used to record voice. In this test, the same key K_1 is considered for U1 and U2. The hardware used includes a microphone Shure SM58, an external soundcard Focusrite 8i6, and a Notebook i5 processor with Cubase audio professional software (Fig. 12a). The


(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Fig. 10 Results of sensitivity tests of secret keys on an ES with case 1: original $m_1(t)$ (a), cryptogram (b), and reception of $m_1'(t)$ (c); case 2: original $m_1(t)$ (d), cryptogram (e), and reception of $m_1'(t)$ (f); and case 3: original $m_1(t)$ (g), cryptogram (h), and reception of $m_1'(t)$ (i)


(a) (b)

(c) (d)

Fig. 11 Test results using the sine signal message (Eq. (24)): (a) $m_1(t)$, $m_1'(t)$, and error message $e_1(t)$; (b) phase plane $m_1(t)$ vs. $m_1'(t)$ with f=10 Hz; (c) zoom of messages $m_1(t)$ and $m_1'(t)$; (d) phase plane $m_1(t)$ vs. $m_1'(t)$ with f=210 Hz and θ=π/2

transmitted voice message is 'hello world'. Fig. 12b shows three signals (recorded by the Cubase software) that are represented as audio tracks: $m_2(t)$ voice message sent (track 1), $m_2'(t)$ voice message recovered (track 2), and the cryptogram acquired by U6 (track 3). Finally, the $m_2'(t)$ voice message is recovered. This message has less audio fidelity because the capacity of audio voice processing is delimited by bandwidth and sampling frequency f_s=1429 Hz.

7 Conclusions

We have presented the design and implementation of a chaotic encryption algorithm using the communication protocol SPI 16-bit microcontrollers on dsPICs to acquire, process, encrypt, transmit, synchronize, receive, decrypt, and re-transmit messages using a DAC together with the DSP theory. Three kinds of tests have been carried out to prove the message performance in the ES: sensitivity secret key, analog signal generated from a function generator, and voice recording. The cryptogram is subjected to a statistical test from the proposed algorithm. The cryptogram has good performance for generating PR sequences. This algorithm can be used in an IC with the standard SPI protocol.

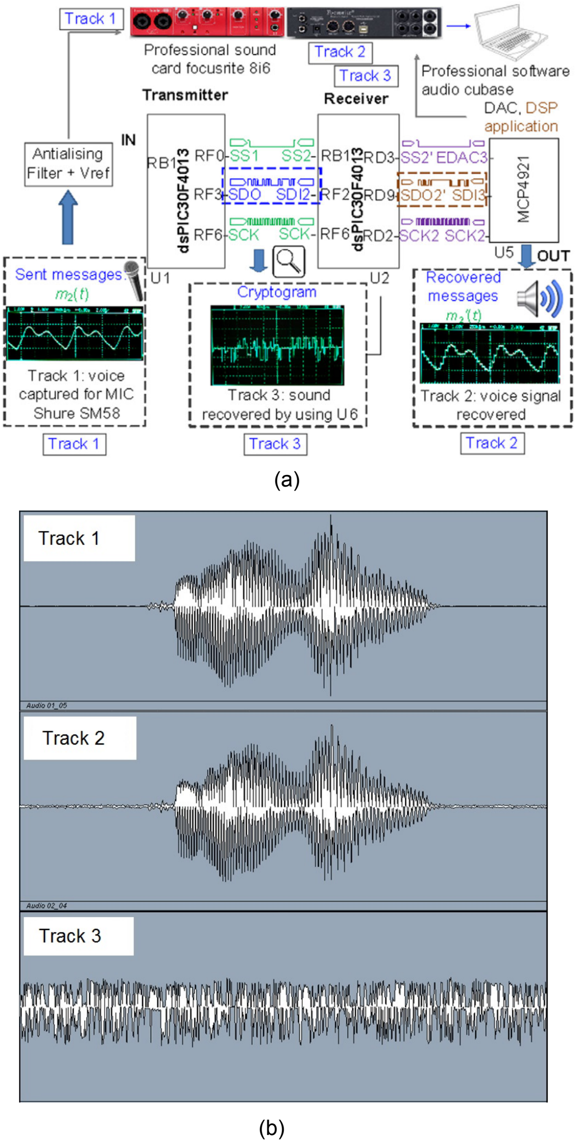


Fig. 12 Test results on the ES using voice message: (a) ES connected with external audio devices; (b) the audio tracks with the messages recorded using software Cubase (references to color refer to the online version of this figure)

The numerical results have been verified using Mikroc Pro for the dsPIC compiler that includes the properties of standard IEEE-754, AN575 of 32 bits into the dsPICs. The quality of message depends on the processing capacity of the ICs. The ES is easy to be installed on a protoboard because it uses the dual inline-package (DIP) and it has a low-cost implementation because the system costs less than 30 USD.

In future work we will test the ES and apply the encrypted and decrypted algorithms with other communication protocols such as I²C or USB-CAN, and Ethernet with larger quantity of bits using other dsPIC or PIC32 microcontrollers with more processing capacity.

References

Aguilar-Bustos AY, Cruz-Hernández C, López-Gutiérrez RM, et al., 2010. Hyperchaotic encryption for secure e-mail communication. In: Chbeir R, Badr Y, Abraham A, et al. (Eds.), Emergent Web Intelligence: Advanced Information Retrieval. Springer, London, p.471-486. https://doi.org/10.1007/978-1-84996-074-8_18

Alvarez G, Li SJ, 2006. Some basic cryptographic requirements for chaos-based cryptosystems. *Int J Bifurc Chaos*, 16(8):2129-2151. https://doi.org/10.1142/S0218127406015970

Arellano-Delgado A, López-Gutiérrez RM, Cruz-Hernández C, et al., 2012. Experimental network synchronization via plastic optical fiber. *Opt Fiber Technol*, 19(12):93-108. https://doi.org/10.1016/j.yofte.2012.11.007

Azzaz MS, Tanougast C, Sadoudi S, et al., 2013. A new auto-switched chaotic system and its FPGA implementation. *Commun Nonl Sci Numer Simul*, 18(7):1792-1804. https://doi.org/10.1016/j.cnsns.2012.11.025

Barr M, Massa A, 2006. Programming Embedded Systems: with C and GNU Development Tools (2nd Ed.). O'Reilly Media, Cambridge, USA, p.18-20, 240-242.

Benson D, Cavanaugh M, Clark K, et al., 2013. GenBank. *Nucl Acids Res*, 41:D36-D42. https://doi.org/10.1093/nar/gks1195

Cornish-Bowden A, 1985. Nomenclature for incompletely specified bases in nucleic acid sequences: recommendations 1984. *Nucl Acids Res*, 13(9):3021-3030. https://doi.org/10.1093/nar/13.9.3021

Demosthenous A, Pachnis I, Jiang D, et al., 2013. An integrated amplifier with passive neutralization of myoelectric interference from neural recording tripoles. *IEEE Sens J*, 13(9):3236-3248. https://doi.org/10.1109/JSEN.2013.2271477

Fúster Sabater A, Hernández Encinas L, Martín Muñoz A, et al., 2012. Criptografía, Protección de Datos Y Aplicaciones: Guía Para Estudiantes Y Profesionales. RA-MA, Madrid, Spain, p.133-157 (in Spanish).

Guglielmi V, Pinel P, Fournier-Prunaret D, et al., 2009. Chaos-based cryptosystem on DSP. *Chaos Sol Fract*, 42(4):2135-2144. https://doi.org/10.1016/j.chaos.2009.03.160

Hénon M, 1976. A two-dimensional mapping with a strange attractor. *Commun Math Phys*, 50(1):69-77. https://doi.org/10.1007/BF01608556

Jasio LD, 2008. Programming 32-Bit Microcontrollers in C: Exploring the PIC32. Newnes, Burlington, USA.

Jyothi M, Chandra LR, Sahithi M, et al., 2012. Implementation of low complex and high secured SPI communication system for multipurpose applications. *Int J Comput Sci*

Inform Technol, 3(1):3214-3219.

Leens F, 2009. An introduction to I^{2}C and SPI protocols. *IEEE Instrum Meas Mag*, 12(1):8-13. https://doi.org/10.1109/MIM.2009.4762946

Li LL, Liu Y, Yao QG, 2014. Robust synchronization of chaotic systems using slidingmode and feedback control. *J Zhejiang Univ-Sci C (Comput & Electron)*, 15(3):211-222. https://doi.org/10.1631/jzus.C1300266

Liu WH, Sun KH, Zhu CX, 2016. A fast image encryption algorithm based on chaotic map. *Opt Lasers Eng*, 84:26-36. https://doi.org/10.1016/j.optlaseng.2016.03.019

Marinkovic SJ, Popovici EM, 2011. Nano-power wireless wake-up receiver with serial peripheral interface. *IEEE J Sel Areas Commun*, 29(8):1641-1647. https://doi.org/10.1109/JSAC.2011.110913

Méndez-Ramírez R, Cruz-Hernández C, Arellano-Delgado A, et al., 2015. Implementación del Circuito Hipercaótico de Chua en un Sistema Embebido de Bajo Costo. http://amca.mx/memorias/amca2015/articulos/0032_MiBT3-04.pdf [Accessed on Nov. 10, 2016] (in Spanish).

Menezes AJ, van Oorschot PC, Vanstone SA, 1996. Handbook of Applied Cryptography. CRC Press, Boca Raton, USA.

Microchip Technology Inc., 1997. IEEE 754 Compliant Floating Point Routines. http://www.microchip.com/stellent/groups/techpub_sg/documents/appnotes/cn010961.pdf [Accessed on Mar. 21, 2018].

Microchip Technology Inc., 2004. dsPIC30F3014, dsPIC30F4013 Data Sheet. http://ww1.microchip.com/downloads/en/devicedoc/70138c.pdf [Accessed on Nov. 10, 2016].

Microchip Technology Inc., 2006. AN1044 Data Encryption Routines for PIC24 and dsPIC® Devices. http://ww1.microchip.com/downloads/en/AppNotes/AN1044a.pdf [Accessed on Nov. 10, 2016].

Microchip Technology Inc., 2018. Encryption Routines for PIC24, dsPIC, and PIC32. http://www.microchip.com/SWLibraryWeb/product.aspx?product=Encryption%20Routines [Accessed on Mar. 21, 2018].

Mikroelectronika, 2014. MikroC Pro for dsPIC Manual. http://download.mikroe.com/documents/compilers/mikroc/dspic/mikroc-dspic-manual-v100.pdf [Accessed on Mar. 21, 2018].

Motorola Inc., 2003. SPI Block Guide V03.06. https://opencores.org/usercontent,doc,1499360489 [Accessed on Mar. 21, 2018].

Muhaya FB, Usama M, Khan MK, 2009. Modified AES using chaotic key generator for satellite imagery encryption. In: Huang DS, Jo KH, Lee HH, et al. (Eds.), Emerging Intelligent Computing Technology and Applications. Springer, Berlin Heidelberg, p.1014-1024. https://doi.org/10.1007/978-3-642-04070-2_107

Murillo-Escobar MÁ, Cruz-Hernández C, Abúndiz-Pérez F, et al., 2015a. A RGB image encryption algorithm based on total plain image characteristics and chaos. *Signal Process*, 109:119-131. https://doi.org/10.1016/j.sigpro.2014.10.033

Murillo-Escobar MÁ, Cruz-Hernández C, Abúndiz-Pérez F, et al., 2015b. A robust embedded biometric authentication system based on fingerprint and chaotic encryption. *Exp Syst Appl*, 42(21):8198-8211. https://doi.org/10.1016/j.eswa.2015.06.035

Nyquist H, 1928. Certain topics in telegraph transmission theory. *Trans Am Inst Electr Eng*, 47(2):617-644. https://doi.org/10.1109/T-AIEE.1928.5055024

Oudjida AK, Berrandjia ML, Tiar R, et al., 2009. FPGA implementation of I^{2}C & SPI protocols: a comparative study. 16th IEEE Int Conf on Electronics, Circuits, and Systems, p.507-510. https://doi.org/10.1109/ICECS.2009.5410881

Philips Semiconductors, 1995. The I²C-bus and how to use it. http://www.i2cbus.org/fileadmin/ftp/i2c_bus_specification_1995.pdf [Accessed on Nov. 10, 2016].

Philips Semiconductors, 2003. AN10216-01 I²C Manual. http://www.nxp.com/documents/application_note/AN10216.pdf [Accessed on Nov. 10, 2016].

Rhouma R, Belghith S, 2011. Cryptanalysis of a chaos-based cryptosystem on DSP. *Commun Nonl Sci Numer Simul*, 16(2):876-884. https://doi.org/10.1016/j.cnsns.2010.05.017

Shannon CE, 1949. Communication in the presence of noise. *Proc IRE*, 37(1):10-21. https://doi.org/10.1109/JRPROC.1949.232969

Siddiqui RA, Grosvenor RI, Prickett PW, 2015. dsPIC-based advanced data acquisition system for monitoring, control and security applications. 12th Int Bhurban Conf on Applied Sciences and Technology, p.293-298. https://doi.org/10.1109/IBCAST.2015.7058519

Sipser M, 2006. Introduction to the Theory of Computation (2nd Ed.). Thomson Course Technology, Boston, USA.

Tumenjargal E, Badarch L, Kwon H, et al., 2013. Embedded software and hardware implementation system for a human machine interface based on ISOAgLib. *J Zhejiang Univ-Sci C (Comput & Electron)*, 14(3):155-166. https://doi.org/10.1631/jzus.C1200270

Uriz AJ, Agüero PD, Moreira JC, et al., 2016. Flexible pseudorandom number generator for tinnitus treatment implemented on a dsPIC. *IEEE Latin Am Trans*, 14(1):72-77. https://doi.org/10.1109/TLA.2016.7430063

Yalcin ME, Suykens JAK, Vandewalle J, 2004. True random bit generation from a double-scroll attractor. *IEEE Trans Circ Syst I*, 51(7):1395-1404. https://doi.org/10.1109/TCSI.2004.830683

Appendix: DNA sequence matrix

The DNA sequence matrix is used in the confusion process (Section 3.2.2) and inverse confusion process (Section 5.2.1). The DNA sequences can be obtained from MATLAB functions ‘getgenbank’ and ‘randseq(*N*)’ or the information can be directly requested by accessing ‘genbank’ (Benson et al., 2013). In this study, the DNA matrix is performed using the MATLAB functions. Table A1 shows one sequence of four nucleotides, referred to as Ad, Cy, Gu, and Th to represent the difference with the vectors and variables used in previous sections, that represents the base of a DNA strand (Cornish-Bowden, 1985).

For the design of the elements of the DNA matrix, we consider pairs of nucleotides to represent the *k* positions; e.g., the pair of nucleotides Ad (adenine) and Th (thymine) represents binary position $(0011)_2$. This means that the pair AdTh represents position 3. The elements determine positions from 1 to the higher value of k=16, because the SPI protocol is configured in 16 bits. These elements are disordered, and cannot be repeated.

Table A1 Binary representation of nucleotides

Nucleotide	Description	Binary number
Ad	Adenine	$(00)_2$
Cy	Cytosine	$(01)_2$
Gu	Guanine	$(10)_2$
Th	Thymine	$(11)_2$

Numerical arrangements of 16 DNA pairs represent one sequence for the DNA matrix. We performed 24 sequences of DNA referred to as r=24 rows, and s=32 columns. The sequences are twice repeated according to the design of algorithms for encryption and decryption. The results of the DNA sequences are contained on matrices $\boldsymbol{D}$ and $\boldsymbol{G}$.

Initially, the data include 16 elements contained in vectors $\boldsymbol{C}$ and $\boldsymbol{F}$. The initial coordinates i, j, l, and u are defined by the chaotic DHMs and Eq. (16). We evaluate these coordinates on matrices $\boldsymbol{D}$ and $\boldsymbol{G}$ to determine a DNA sequence with 16 new positions. The new positions of these elements are contained in vectors $\boldsymbol{E}$ and $\boldsymbol{H}$. Finally, the complete information of these DNA sequences are defined in Eq. (A1).

$$\boldsymbol{D}=\boldsymbol{G}=$$

$$\begin{bmatrix}
5 & 10 & 0 & 13 & 2 & 7 & 4 & 11 & 6 & 1 & 8 & 9 & 15 & 14 & 3 & 12 & 5 & 10 & 0 & 13 & 2 & 7 & 4 & 11 & 6 & 1 & 8 & 9 & 15 & 14 & 3 & 12 \\
14 & 5 & 12 & 9 & 4 & 7 & 8 & 10 & 11 & 3 & 1 & 15 & 13 & 2 & 0 & 6 & 14 & 5 & 12 & 9 & 4 & 7 & 8 & 10 & 11 & 3 & 1 & 15 & 13 & 2 & 0 & 6 \\
10 & 5 & 9 & 2 & 7 & 3 & 13 & 4 & 14 & 6 & 8 & 1 & 11 & 15 & 12 & 0 & 10 & 5 & 9 & 2 & 7 & 3 & 13 & 4 & 14 & 6 & 8 & 1 & 11 & 15 & 12 & 0 \\
13 & 7 & 5 & 10 & 0 & 9 & 8 & 11 & 2 & 4 & 1 & 15 & 14 & 6 & 3 & 12 & 13 & 7 & 5 & 10 & 0 & 9 & 8 & 11 & 2 & 4 & 1 & 15 & 14 & 6 & 3 & 12 \\
11 & 13 & 12 & 8 & 14 & 9 & 4 & 7 & 5 & 2 & 6 & 0 & 3 & 1 & 15 & 10 & 11 & 13 & 12 & 8 & 14 & 9 & 4 & 7 & 5 & 2 & 6 & 0 & 3 & 1 & 15 & 10 \\
5 & 4 & 2 & 15 & 11 & 12 & 7 & 8 & 1 & 0 & 3 & 13 & 9 & 10 & 14 & 6 & 5 & 4 & 2 & 15 & 11 & 12 & 7 & 8 & 1 & 0 & 3 & 13 & 9 & 10 & 14 & 6 \\
14 & 13 & 15 & 5 & 10 & 3 & 11 & 12 & 7 & 9 & 1 & 2 & 0 & 6 & 4 & 8 & 14 & 13 & 15 & 5 & 10 & 3 & 11 & 12 & 7 & 9 & 1 & 2 & 0 & 6 & 4 & 8 \\
7 & 9 & 11 & 14 & 15 & 6 & 5 & 0 & 10 & 4 & 2 & 8 & 13 & 1 & 12 & 3 & 7 & 9 & 11 & 14 & 15 & 6 & 5 & 0 & 10 & 4 & 2 & 8 & 13 & 1 & 12 & 3 \\
3 & 4 & 5 & 6 & 1 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 0 & 7 & 2 & 3 & 4 & 5 & 6 & 1 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 0 & 7 & 2 \\
8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 & 15 & 14 & 13 & 12 & 11 & 10 & 9 & 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 & 15 & 14 & 13 & 12 & 11 & 10 & 9 \\
2 & 4 & 8 & 6 & 10 & 12 & 14 & 0 & 1 & 3 & 5 & 7 & 9 & 11 & 13 & 15 & 2 & 4 & 8 & 6 & 10 & 12 & 14 & 0 & 1 & 3 & 5 & 7 & 9 & 11 & 13 & 15 \\
3 & 15 & 1 & 13 & 11 & 9 & 7 & 5 & 2 & 14 & 0 & 12 & 10 & 8 & 6 & 4 & 3 & 15 & 1 & 13 & 11 & 9 & 7 & 5 & 2 & 14 & 0 & 12 & 10 & 8 & 6 & 4 \\
10 & 12 & 11 & 7 & 13 & 8 & 3 & 6 & 15 & 1 & 5 & 9 & 2 & 0 & 4 & 14 & 10 & 12 & 11 & 7 & 13 & 8 & 3 & 6 & 15 & 1 & 5 & 9 & 2 & 0 & 4 & 14 \\
6 & 5 & 3 & 0 & 12 & 13 & 8 & 9 & 2 & 11 & 4 & 10 & 14 & 1 & 15 & 7 & 6 & 5 & 3 & 0 & 12 & 13 & 8 & 9 & 2 & 11 & 4 & 10 & 14 & 1 & 15 & 7 \\
4 & 12 & 9 & 5 & 0 & 13 & 15 & 1 & 6 & 7 & 11 & 8 & 10 & 2 & 14 & 3 & 4 & 12 & 9 & 5 & 0 & 13 & 15 & 1 & 6 & 7 & 11 & 8 & 10 & 2 & 14 & 3 \\
1 & 0 & 6 & 15 & 10 & 5 & 4 & 2 & 8 & 3 & 14 & 13 & 7 & 12 & 9 & 11 & 1 & 0 & 6 & 15 & 10 & 5 & 4 & 2 & 8 & 3 & 14 & 13 & 7 & 12 & 9 & 11 \\
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\
15 & 14 & 13 & 12 & 11 & 10 & 9 & 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 & 15 & 14 & 13 & 12 & 11 & 10 & 9 & 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\
12 & 6 & 14 & 8 & 0 & 2 & 4 & 10 & 11 & 13 & 15 & 9 & 1 & 7 & 3 & 5 & 12 & 6 & 14 & 8 & 0 & 2 & 4 & 10 & 11 & 13 & 15 & 9 & 1 & 7 & 3 & 5 \\
1 & 3 & 5 & 7 & 9 & 11 & 13 & 15 & 0 & 2 & 4 & 6 & 8 & 10 & 12 & 14 & 1 & 3 & 5 & 7 & 9 & 11 & 13 & 15 & 0 & 2 & 4 & 6 & 8 & 10 & 12 & 14 \\
2 & 0 & 6 & 4 & 8 & 14 & 12 & 3 & 10 & 1 & 7 & 5 & 9 & 11 & 15 & 13 & 2 & 0 & 6 & 4 & 8 & 14 & 12 & 3 & 10 & 1 & 7 & 5 & 9 & 11 & 15 & 13 \\
7 & 15 & 11 & 9 & 7 & 5 & 3 & 14 & 12 & 4 & 6 & 8 & 10 & 2 & 0 & 1 & 7 & 15 & 11 & 9 & 7 & 5 & 3 & 14 & 12 & 4 & 6 & 8 & 10 & 2 & 0 & 1 \\
13 & 5 & 10 & 3 & 8 & 9 & 6 & 2 & 0 & 15 & 14 & 13 & 1 & 11 & 12 & 4 & 13 & 5 & 10 & 3 & 8 & 9 & 6 & 2 & 0 & 15 & 14 & 13 & 1 & 11 & 12 & 4 \\
9 & 6 & 15 & 11 & 0 & 10 & 4 & 13 & 2 & 1 & 8 & 14 & 12 & 3 & 7 & 5 & 9 & 6 & 15 & 11 & 0 & 10 & 4 & 13 & 2 & 1 & 8 & 14 & 12 & 3 & 7 & 5
\end{bmatrix}. \tag{A1}$$