UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

Facultad de Ingeniería, Arquitectura y Diseño

Programa de Maestría y Doctorado en Ciencias e Ingeniería

Encriptado hipercaótico de imágenes digitales implementado en FPGA con bioclave de acceso

TESIS

que para cubrir parcialmente los requisitos necesarios para obtener el grado de

DOCTOR EN CIENCIAS

presenta:

Eduardo Rodríguez Orozco

Director de tesis

Dr. Enrique Efren García Guerrero

Ensenada, Baja California, México. Diciembre del 2017.

UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

Facultad de Ingeniería, Arquitectura y Diseño

Encriptado hipercaótico de imágenes digitales implementado en FPGA con bioclave de acceso

TESIS

Que para obtener el grado de Doctor en Ciencias presenta:

Eduardo Rodríguez Orozco

Y aprobada por el siguiente comité:

Dr. Enrique Efrén García Guerrero
Director de tesis

Dr. Everardo Inzunza González
Co-director de tesis

Dr. Oscar Roberto López Bonilla
Miembro del comité

Dr. Juan Iván Nieto Hipólito
Miembro del comité

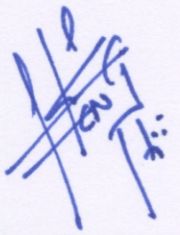
Dr. José Ricardo Cárdenas Valdez
Miembro del comité

Ensenada Baja California, México. Diciembre 2017

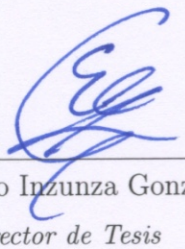
RESUMEN de la tesis de **Eduardo Rodríguez Orozco**, presentada como requisito parcial para obtener el grado de DOCTOR EN CIENCIAS, del programa de Maestría y Doctorado en Ciencias e Ingeniería de la UABC. Ensenada, B. C. México, Diciembre de 2017

Encriptado hipercaótico de imágenes digitales implementado en FPGA con bioclave de acceso

Resumen aprobado por:



Dr. Enrique Efren García Guerrero
Director de Tesis



Dr. Everardo Inzunza González
Co-director de Tesis

En este trabajo de tesis doctoral, se presenta el desarrollo de un sistema embebido para el encriptado hipercaótico de imágenes digitales, este sistema incluye un chip de reconocimiento de voz, la cual sirve como bioclave externa para accesar al sistema propuesto. El sistema encriptador se integra por tres tecnologías: i) una FPGA Spartan 3E-1600 de Xilinx, ii) una computadora de placa reducida (Single Board Computer) Raspberry Pi 2 de 32 bits y iii) un chip de reconocimiento de voz (CRV) fabricado por la compañia Sunplus. El sistema general funciona bajo algoritmos embebidos tales como: **1)** una **interfaz gráfica** desarrollada en lenguaje Python para el sistema embebido Raspberry Pi, que permite el manejo amigable del sistema, **2)** un **algoritmo de control interno** implementado en FPGA, que conlleva a) la puesta en operación del sistema embebido a partir del reconocimiento de la señal de voz como bioclave de acceso al sistema encriptador, b) el envío de los pixeles de la imagen a encriptar/desencriptar hacia la FPGA, y c) la ejecución propia del algoritmo de encriptado o desencriptado

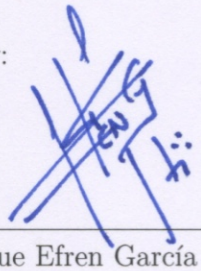
implementado en la FPGA, **3)** un **algoritmo generador binario pseudo-aleatorio caótico** (CPRBG) cuyos valores numéricos decimales se acondicionan a una escala binaria de 8 bits mediante la implementación de la operación *mod* 255 en la FPGA y **4) dos algoritmos de comunicación serial UART** (Universal Asynchronous Receiver Transmitter) desarrollados a partir del protocolo RS-232, ambos implementados en la FPGA en lenguaje VHDL. La operatividad, eficiencia, confiabilidad y robustez del sistema se evalúa mediante el análisis de seguridad y un análisis nivel de calidad del encriptado a los distintos criptogramas obtenidos empleando diversos mapeos caóticos. Adicionalmente, para evaluar y validar la aleatoriedad del CPRBG, se realizaron cinco pruebas empleando el estándar FIPS 140-2 de la NIST (National Institute of Standards and Technology).

Palabras clave: Caos, Encriptado, FPGA, CPRBG, Sistemas Embebidos, Raspberry Pi.

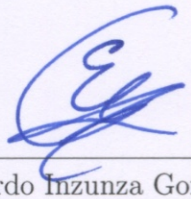
ABSTRACT of the thesis of **Eduardo Rodríguez Orozco**, presented as a partial requirement to obtain the degree of DOCTOR in SCIENCE, of the program of MSc and PhD in Sciences and Engineering of UABC. Ensenada, B. C., Mexico, December 2017

An FPGA implementation of hyperchaos based image encryption by using a biokey as access

Abstract approved by:



Dr. Enrique Efren García Guerrero
Thesis supervisor



Dr. Everardo Inzunza González
Thesis supervisor

In this PhD thesis, the development of an embedded system for hypercathetic encryption of digital images is presented, this system includes a speech recognition chip, which serves as an external biokey to access the proposed system. The encryption system is integrated by three technologies: i) A FPGA Spartan 3E-1600 Xilinx, ii) A Single Board Computer Raspberry Pi 2, and iii) A voice recognition chip (VRC), manufactured by the company Sunplus. The general system works under embedded algorithms such as: 1) a graphical user interface developed in Python language for the embedded system Raspberry Pi, that allows friendly system management, 2) an algorithm for internal control implemented in FPGA, what it entails, a) the start-up of the embedded system based on the recognition of the voice signal as a biokey to access the cryptosystem, b) the sending of the pixels of the image to be encrypted/decrypted towards the FPGA, and c) the execution of the encryption or decryption algorithm implemented in the FPGA, 3) a chaotic pseudo-random binary generator algorithm (CPRBG), whose decimal numerical values are conditioned on a binary scale of 8 bits through the implementation of the operation *mod* 255 in the FPGA, and 4) two serial communication algorithms UART (Universal Asynchronous Receiver Transmitter) developed from the RS-232 protocol, both implemented in VHDL language for FPGA. The

functionality, efficiency, reliability and robustness of the system is evaluated through the security analysis and an analysis of quality level of the encryption to the different cryptograms using various chaotic maps. Additionally, to evaluate and validate the randomness of the CPRBG, five tests were performed using the FIPS 140-2 standard of the NIST (National Institute of Standards and Technology).

Keywords: Chaos, Encryption, FPGA, CPRBG, Embedded Systems, Raspberry Pi.

A mi querida Madre Edelmira y a mi familia que siempre me ha apoyado en mis sueños, gracias..

Agradecimientos

Al ***Dr. Enrique Efrén García Guerrero****, por haberme dirigido en este camino tan complicado e interesante. Por toda su paciencia y por sus valiosos consejos que permitieron la culminación de este trabajo de investigación.*

Al ***Dr. Everardo Inzunza González****, por haberme co-dirigido en mi desarrollo profesional. Por toda su valioso tiempo dedicado en mi trabajo de tesis doctoral.*

Al ***Dr. Oscar Roberto López Bonilla****, por sus comentarios en las presentaciones y revisiones de la tesis, que complementaron la formación doctoral.*

Al ***Dr. Juan Iván Nieto Hipólito****, por su atención y amabilidad en las revisiones de la tesis durante mi estancia en el doctorado.*

Al ***Dr. José Ricardo Cárdenas Valdez****, por sus valiosas aportaciones en las presentaciones y revisiones de la tesis.*

Al ***coordinador de posgrado****, Dr. Miguel Enrique Martinez Rosas, por sus asesorías administrativas.*

A nuestra alma mater: ***Universidad Autónoma de Baja California****, que tanto me ha dado en lo profesional y en lo académico.*

A la ***Facultad de Ingeniería, Arquitectura y Diseño****, la cual considero como mi segundo hogar.*

Al ***CONACyT****, por la beca otorgada para mis estudios de doctorado.*

Ensenada, B. C. México. Eduardo Rodríguez Orozco

20 de Diciembre de 2017.

Tabla de Contenido

Página

Tabla de Contenido (Continuación)

Tabla de Contenido (Continuación)

Página

Lista de figuras

Lista de figuras (Continuación)

Lista de figuras (Continuación)

Lista de figuras (Continuación)

Lista de tablas

Lista de tablas (Continuación)

Capítulo I

Introducción

Los sistemas caóticos han tenido un auge significativo en la comunidad científica durante las últimas dos décadas, con un enfoque teórico o experimental. Una de sus principales aplicaciones se encuentra en el área de seguridad informática, específicamente en el diseño e implementación de algoritmos de encriptado con base en el caos ((Alvarez y Li, 2006), (Kocarev y Lian, 2011) y (Zhang *et al.*, 2010)).

La seguridad informática es el área de la informática que se enfoca en la protección de la infraestructura computacional, especialmente, la información contenida o circulante. Para ello existen una serie de estándares, protocolos, métodos, reglas, herramientas y leyes concebidas para minimizar los posibles riesgos a la infraestructura o a la información. La seguridad informática comprende software (bases de datos, metadatos, archivos), hardware y todo lo que la organización valore y signifique un riesgo si esta información confidencial llega a manos de otras personas, convirtiéndose, por ejemplo, en información privilegiada para terceros.

El presente trabajo de tesis doctoral, propone un sistema criptográfico caótico embebido. Está compuesto de: Una "Field Programmable Gate Array"(**FPGA**) como elemento encriptador. Una interfaz gra desarrollado en el sistema embebido **Raspberry Pi 2**, que tiene como objetivo capturar imágenes mediante una cámara digital y establecer una comunicación entre el FPGA y la Raspbery Pi 2 para obtener una imagen encriptada. Un Chip de **Reconocimiento de Voz** (sistema embebido) como clave de inicio del proceso de cifrado.

I.1 Motivación

Con el rápido desarrollo de la tecnología de Internet y el procesamiento digital de señales, un porcentaje de los recursos multimedia son imágenes, que son utilizados en muchas áreas como autenticación biométrica, ciencias médicas, militares e incluso fotografías personales, por consecuencía la protección y la transmisión segura de la información se está convirtiendo en un problema más importante. Por tal motivo, se propone la implementación digital de un generador caótico en un sistema emebebido, con aplicaciones potenciales en sistemas de telecomunicaciones y redes de comunicación.

I.2 Planteamiento del problema

En la actualidad se está ejerciendo un desarrollo tecnológico dirigido hacia la producción de materiales, dispositivos, sistemas o métodos incluyendo el diseño, desarrollo, mejora de prototipos, procesos, productos, servicios o modelos organizativos. La seguridad informática es el área de la informática que a tenido un auge en su desarrollo especificamente en la criptografía.

Las implementaciones analógicas de sistemas caóticos, por ejemplo [(Gao y Chen, 2008), (Orúe López *et al.*, 2012), (Wang *et al.*, 2004), (Li *et al.*, 2007), (Zhang *et al.*, 2006), (Qi *et al.*, 2009), (Pang y Liu, 2011)], exhiben varias dificultades en su delicada estructura y al recuperar la información tienen que usar observadores no lineales y sincronía, aunado a esto existe un problema con los componentes ya que varían con el tiempo, temperatura, etc.

La mayoria de los sistemas criptográficos actuales estan basados en computadoras personales y el encriptado de la información es mediante el uso de software, el cual es vulnerable a diferentes tipos de ataques. En este trabajo de tesis doctoral se propone implementar nuevas técnicas de encriptado caótico de la información en sistemas

embebidos, esto con la finalidad de optimizar los algoritmos criptográficos, que sean lo suficientemente eficientes para incrementar y garantizar los niveles de seguridad y privacidad que ya son necesarios en nuestros días y que seguramente serán más demandantes en el futuro.

I.2.1 Propuesta de solución

En la figura 1, se ilustra el esquema de la propuesta de solución, es un sistema embebido, utilizado tanto como cifrador o descifrador de imágenes y está constituido por un FPGA, un Raspbery Pi 2 y un Chip de Reconocimiento de Voz.

El sistema embebido Raspberry Pi 2 adquiere la imagen a cifrar y espera una señal de encendido del proceso, el cual proviene de la FPGA, el inicio del proceso de encriptado depende del Chip de Reconocimiento de Voz (CRV), está esperando al usuario con la clave de voz correcta, al iniciar el proceso, la Raspberry Pi 2 transmite la imagen al FPGA, esta recive pixel por pixel, lo encripta y lo regresa al Raspberry Pi 2. Posteriormente el usuario recive el criptograma y lo envia por la red pública a un receptor remoto donde se tiene otro sistema embebido idéntico al transmisor, dicho recepor remoto desencripta el criptograma utilizando el mismo proceso anteriormente comentado.

En este trabajo de tesis se propone emplear 5 generadores caóticos inmersos en tecnología FPGA, es decir los siguientes mapeos: Hénon (Hénon, 1976), Rössler (Rossler, 1979), Tinkerbell (Goldsztejn *et al.*, 2011), Karplan Yorke (Kaplan y Yorke, 1979) y Logistic 2D (Grassberger y Procaccia, 2004), en conjunto con la operación **mod 255** para generar una secuencia binaria psedoaleatoria caótica (**CPRBG**) y usarla para encriptar una imagen digital.

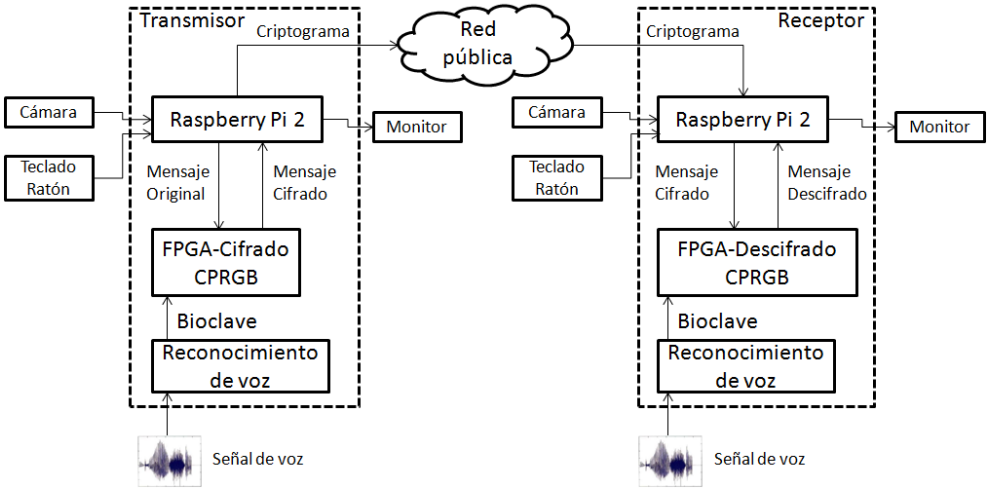


Figura 1: Esquema de la propuesta de solución.

I.3 Objetivos

Objetivo general:

Diseñar e implementar un sistema embebido de encriptado caótico basado en tecnología FPGA, empleando una bioclave como llave de acceso.

Objetivos específicos:

- *Implementar distintos mapeos caóticos en la FPGA.*
- *Proponer un nuevo algoritmo de encriptado caótico.*
- *Implementar el algoritmo propuesto de encriptado caótico en la FPGA.*
- *Evaluar el nivel de seguridad de la información encriptada en la FPGA.*
- *Seleccionar el circuitos para reconocimiento de voz.*
- *Integrar el chip de reconocimiento de voz con la FPGA.*

- *Diseñar e implementar un sistema embebido de cifrado caótico usando el circuito de reconocimiento de voz.*
- *Proponer y programar una interfaz gráfica en Raspberry Pi 2 para el sistema embebido de cifrado propuesto.*

I.4 Organización del manuscrito

El presente trabajo de tesis doctoral se forma de cinco capítulos y cuatro apéndices. En el **segundo capítulo**, se muestra una introducción al marco teórico que esta compuesto por ocho temas: la criptografía, análisis de seguridad, teoría de caos, los sistemas caóticos contínuos, los sistemas caóticos discretos, los sistemas embebidos. En el **tercer capítulo**, se presenta la metodología empleada en la elaboración del experimento: el sistema de encriptado propuesto, el algoritmo embebido de ebcriptado y el algoritmo generador binario pseudoaleatorio. En el **cuarto capítulo** se presentan los criptogramas obtenidos, los resultados del análisis estadísticos a los mismos y el análisis a la secuencia binaria FIPS 140-2. El **quinto capítulo** presenta las conclusiones y trabajos a futuro. El **apéndice A** presenta otros mapeos caóticos discretos, en el **apéndice B** se ilustran 4 mapeos implementados en FPGA, en el **apéndice C** se plasman los recursos necesarios para la implementación, los respectivos criptogramas y por último los análisis estadísticos de los 4 mapeos presentados en el apéndice B, el **apéndice D** muestra la configuración del Chip de reconocimiento de voz.

Capítulo II

Marco teórico

II.1 Criptografía

Las raíces etimológicas de la palabra Criptografía son criptos (oculto) y graphos (escritura). Según la *Real Academía Española* (RAE) se define como el arte de escribir mensajes con clave secreta o de un modo enigmático.

La Criptografía es la ciencia que se encarga del estudio de técnicas para transformar la información a una forma que no pueda entenderse a simple vista; sin embargo, el objetivo de la Criptografía no es sólo mantener los datos secretos, sino también protegerlos contra modificación y comprobar la fuente de los mismos (Menezes *et al.*, 1996).

El Criptoanálisis es la ciencia que se ocupa del análisis de un texto cifrado para obtener la información original sin conocimiento de la clave secreta, esto es, de forma ilícita rompiendo así los procedimientos de cifrado establecidos por la Criptografía, por lo que se dice que Criptoanálisis y Criptografía son ciencias complementarias pero contrarias (Menezes *et al.*, 1996).

II.2 Análisis de seguridad

La seguridad (del latín securitas) se puede referir a la ausencia de riesgo o a la confianza en algo o en alguien. Sin embargo, el término puede tomar diversos sentidos según el área o campo a la que haga referencia en la seguridad.

En (Behnia *et al.*, 2008) se define la seguridad como una medida crucial de la calidad de un sistema criptográfico, es la capacidad de resistir los ataques de intrusos o usuarios no autorizados para obtener conocimiento de la información sin encriptar (Inzunza, 2012).

II.2.1 Análisis de Espacio de Claves

El espacio de claves es una medida del número de claves posibles que pueden usarse en un cifrador. La longitud de la clave es crítica para determinar la susceptibilidad de un cifrador frente a un ataque de búsqueda exhaustivo, entonces el espacio de claves debe ser lo suficientemente grande para hacer inviable el ataque de fuerza bruta (Fu *et al.*, 2011).

Si el espacio de clave de un algoritmo de cifrado es lo suficientemente grande, típicamente mayor a 128 bits, ya se considera seguro para la mayoría de las aplicaciones criptográficas en términos de la velocidad de las computadoras actuales, por lo tanto, el ataque de fuerza bruta en tal algoritmo es inviable (Patidar *et al.*, 2011).

La seguridad de un sistema criptográfico debe depender sólo de la clave, esto fue formulado de manera explícita por Auguste Kerckhoffs (Kerckhoffs, 1978) y por Claude Shannon (Shannon, 2001). Estas declaraciones se conocen como principio de Kerckhoffs y Máxima seguridad de Shannon respectivamente.

II.2.2 Análisis de Sensibilidad de Claves

Una característica de un sistema criptográfico es la sensibilidad de la clave, esto es, debe ser sensible a pequeños cambios en la clave. El análisis a la sensibilidad de clave se realiza de dos maneras: (a) la imagen encriptada (criptograma) producida por un sistema criptográfico debe ser muy sensible a la clave secreta, es decir, si utilizamos claves ligeramente diferentes para cifrar la misma imagen original, entonces las imágenes

cifradas que se producen, deben ser completamente independientes (diferentes), en otras palabras, deben posser una correlación insignificante, (b) la imagen cifrada, no debe ser descifrada correctamente aunque exista una ligera diferencia entre las claves de cifrado y descifrado (Patidar *et al.*, 2011).

II.2.3 Histogramas

En estadística, un histograma es una representación gráfica de una variable en forma de barras, donde la superficie de ellas es proporcional a la frecuencia de los valores representados de la variable. El histograma de una imagen es una gráfica que muestra la frecuencia de cada valor de los pixeles. Para una imagen de 8 bits en escala de grises, existen 256 niveles de intensidades diferentes (Patidar *et al.*, 2011). La distribución de frecuencias de un histograma de un criptograma debe ser uniforme, de lo contrario el sistema de cifrado es suceptible a un ataque de histogramas.

Un criptograma debe ocultar la redundancia de la imagen original y no debería filtrarse alguna información de ella o que exista una relación entre el criptograma y la imagen original (Fu *et al.*, 2011).

II.2.4 Entropía de la información

La entropía, también llamada entropía de la información o entropía de (Shannon, 2001), es un criterio que muestra la aleatoriedad de los datos. También puede ser utilizada para evaluar la seguridad del cifrado de una imagen (Mao y Deng, 2011).

Para calcular la entropía $H(s)$ (Behnia *et al.*, 2007) de una fuente de pixeles (s), se utiliza la ecuación (1):

$$H(s) = \sum_{n=0}^{2^N-1} P(s_i) \times \log_2(\frac{1}{P(s_i)} bits) \tag{1}$$

donde $P(s_i)$ representa la probabiliadad del simbolo s_i, N es el número de bits que representan la unidad básica de la fuente s, 2^N son todas las combinaciones de la unidad básica, para una fuente puramente aleatoria se espera una entropía de $H(s) = N$, si tomamos imágenes con píxeles completamente aleatorios en la escala de grises de 8 bits, su entropía $H(s) = 8$.

Cuando $H(s)$ de un criptograma emite resultados menores de 8 bits, este cifrador tiene cierto grado de predicibilidad, por lo que la seguridad se pone en riesgo (Behnia *et al.*, 2008).

II.2.5 Ataques Diferenciales

Un ataque diferencial es básicamente un ataque sencible a cambios mínimos en la clave. Existen dos medidas diferenciales comúnes NPCR (Number Pixel Change Rate) y UACI (Unified Avarage Changing Intensity) (Chen *et al.*, 2004). Estas son utilizadas para probar la influencia del cambio de un pixel en toda la imagen encriptada.

El NPCR evalúa el porcentaje de la cantidad de píxeles diferentes entre dos imágenes y se puede calcular utilizando la ecuación (2):

$$NPCR = \frac{\sum_{i,j} D(i,j)}{W \times H} \times 100\% \tag{2}$$

donde $D(i,j)$ es un arreglo binario:
$D(i,j) = 0$, si $C_1(i,j) = C_2(i,j)$,
$D(i,j) = 1$, cuando $C_1(i,j) \neq C_2(i,j)$,
C_1 y C_2 son imágenes cifradas obtenidas con claves muy semejantes. W y H definen el tamaño de la imagen bajo análisis (Mao y Deng, 2011) y (Wang *et al.*, 2015).

El UACI evalua la intensidad promedio de las diferencias entre las dos imágenes cifradas C_1 y C_2, se calcula empleando la ecuación (3):

$$UACI = \frac{1}{W \times H} \sum_{i,j} \frac{|C_1(i,j) - C_2(i,j)|}{255} \times 100\% \tag{3}$$

donde C_1, C_2, W y H fueron definidas previamente por (Mao y Deng, 2011) y (Wang *et al.*, 2015).

II.2.6 Correlación de pixeles adyacentes

(Shannon, 2001) propune dos técnicas básicas para realizar el diseño de cifradores la difusión y confusión, estas pueden ser demostradas por una prueba en la correlación de píxeles adyacentes en el criptograma (Chen *et al.*, 2004).

Se sabe que los píxeles adyacentes de una imagen están altamente correlacionados, esto significa que el valor de un pixel y el valor de su pixel vecino (horizontal, vertical o diagonal) son muy parecidos. La correlación de una imagen se puede gráficar y también se puede medir entre -1 y 1, donde 0 significa correlación nula.

Se toman al menos 2000 pares de píxeles adyacentes (en dirección horizontal, vertical y diagonal) a partir de una imagen bajo análisis y se calcula el coeficiente de correlación respectivamente (Chen *et al.*, 2004), mediante la ecuación (4).

$$r_{xy} = \frac{cov(x,y)}{\sqrt{D(x)}\sqrt{D(y)}} \tag{4}$$

$$cov(x,y) = \frac{1}{N} \sum_{i=1}^{N} (x_i - E(x))(y_i - E(y)) \tag{5}$$

donde $cov(x,y)$ es la covarianza, $D(x)$ es la varianza, x e y denotan los valores en la

escala de grises de la imagen bajo análisis. Para este caso de computación numérica, se utiliza la ecuación (6) y (7).

$$E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{6}$$

$$D(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x)) \tag{7}$$

donde $E(x)$ es el valor promedio de los niveles de gris de los píxeles. La gráfica de la correlación (en dirección horizontal, vertical y diagonal) de píxeles adyacentes se genera a partir de las parejas de píxeles (x_i, y_i) de la imagen sujeta para análisis, es decir se gráfica el pixel x_i vs y_i.

II.2.7 Calidad del algoritmo de encriptado

La calidad del algoritmo de cifrado se evalua por medio de 3 pruebas (Elkamchouchi y Makar, 2005): el factor de desviación irregular, correlación y máximo factor de desviación.

El factor de desviación irregular es expresado como la desviación de la intensidad de los píxeles en la imagen cifrada (IE), con respecto a los de la imagen original(IO). La desviación se obtiene calculando la matriz X, la cual representa el valor absoluto de la desviación entre cada valor del pixel antes y después del encriptamiento. Se obtiene el histograma de las diferencias y se calcula el valor promedio D seguido de S (valor absoluto de la diferencia de los valores del histograma menos D). Finalmente se determina el parámetro de calidad de encriptamiento AS (suma de las diferencias S) (Mercado-Sánchez *et al.*, 2009).

Resumen de los pasos para obtener AS:

$$X = |IO - IE|$$

$$H = histograma(X)$$

$$D = \frac{1}{256}\sum_{i=0}^{255} h_i$$

$$S(i) = |H(i) - D|$$

$$AS = \sum_{i=0}^{255} D(i) \tag{8}$$

donde IO es la imagen original, IE es la imagen cifrada, h_i es la amplitud de las diferencias absolutas, AS es la desviación irregular. Para una imagen de NxM píxeles el valor esperado es superior a $(NxM)/2$ (Mercado-Sánchez *et al.*, 2009).

La correlación (CC) tiene el objetivo de medir el grado de semajanza entre imagen original e imagen encriptado y se evalúa por medio de la siguiente ecuación:

$$CC = \frac{cov(x,y)}{(\sigma_x, \sigma_y)} = \frac{\sum_{i=1}^{N}(x_i - E(x))(y_i - E(y))}{\sqrt{\sum_{i=1}^{N}(x_i - E(x))^2}\sqrt{\sum_{i=1}^{N}(y_i - E(y))^2}} \tag{9}$$

donde $E(x)$ es el promedio de los píxeles en la imagen x y $E(y)$ es el promedio de los píxeles en la imagen y. Se espera un valor cercano a cero si no existe correlación.

El máximo factor de desviación, mide la calidad del encriptado en términos de cómo maximiza la desviación entre las imágenes originales y encriptada. Se calcula a partir de la ecuación 10, la cual se ilustra a continuación:

$$D = \frac{h_0 + h_{255}}{2} + \sum_{i=1}^{254} h_i \tag{10}$$

donde h_i es amplitud de las diferencias absolutas y D es el máximo factor de desviación. Para una imagen de NxM píxeles el valor esperado es próximo a NxM.

II.2.8 Pruebas FIPS 140-2 a la Secuencia Binaria

El análisis secuencia binaria esta basado en las siguientes pruebas de acuerdo con el estándar FIPS 140-2, es el acrónimo de Federal Information Processing Standard (estándar federal de procesamiento de la información), es un estándar de seguridad de ordenadores del gobierno de los Estados Unidos para la acreditación de módulos criptográficos (FIPS, 2001), (Menezes *et al.*, 1996):

Monobit: El propósito de esta prueba es determinar si una secuencia binaria tiene aproximadamente la misma cantidad de $0's$ y $1's$.

$$X_1 = \frac{(n_0 - n_1)^2}{n} \tag{11}$$

donde n_0 es la cantidad de ceros en la secuencia y n_1 es la cantidad de unos en la secuencia, n es la cantidad total de bits en la secuencia.

Lo cual sigue aproximadamente una distribución cuadrada Chi con un grado de libertad, si $n \geq 10$.

Serial: Tiene el propósito de determinar que una secuencia binaria tiene la misma cantidad de bits de ocurrencias de subsecuencias 00 , 01, 10 y 11 de la secuencia binaria.

$$X_2 = \frac{4}{n-1}(n_{00}^2 + n_{01}^2 + n_{10}^2 + n_{11}^2) - \frac{2}{n}(n_0^2 - n_1^2) - 1 \tag{12}$$

donde n_{00} es la cantidad de doble cero que se repite, n_{01} es la cantidad un cero y un uno continuos que se repite, n_{10} es la cantidad un uno y un cero continuos que se

repite y n_{11} es la cantidad de dobles unos que se repite. n, n_0 y n_1 han sido declaradas previamente.

Lo cual sigue aproximadamente una distribución cuadrada Chi con dos grados de libertad, si $n \geq 21$.

Poker: Tiene el propósito de determinar que una subsecuencia binaria de longitud m, aparece la misma cantidad de veces en la secuencia binaria.

$$X_3 = \frac{2^m}{k}(\sum_{i=1}^{2^m} n_i^2) - k \tag{13}$$

donde $n/m \geq 5(2^m)$ y $k = n/m$.

La cual sigue aproximadamente una distribución cuadrada Chi con $2^m - 1$ grados de libertad, si ≥ 10. La prueba Poker es una generalización de la prueba monobit cuando $m = 1$.

Runs: El propósito de esta prueba es determinar que una subsecuencia binaria de bloques o "gaps"en una secuencia binaria tienen la misma ocurrencia.

$$X_4 = \sum_{i=1}^{k} \frac{(B_i - e_i)^2}{e_i} + \sum_{i=1}^{k} \frac{(G_i - e_i)^2}{e_i} \tag{14}$$

donde $e_i = (n - i + 3)/2^{i+2}$ para $e_i \geq 5$ y $(1 \leq i \leq k)$. Son bloques las subsecuencias de 010, 0110, 01110, 011110, 0111110, 01...10 y gaps las subsecuencias de 101, 1001, 10001, 100001, 1000001, 10...01. Sean B_i y G_i la cantidad de bloques y gaps de tamaño i respectivamente. La cantidad de bloques o gaps de tamaño i esperados en una secuencia pseudoaleatoria de tamaño n está determinada por e_i. Sea k igual al entero mayor de las iteraciones i, mientras $e_i \geq 5$.

La cual sigue aproximadamente una distribución cuadrada Chi con $2k - 2$ grados de libertad.

Autocorrelación: Tiene el propósito de probar la correlación de la secuencia binaria pseudoaleatoria s_i con la misma secuencia pero desplazada. Sea d un número entero

fijo,

$$X_5 = 2(A(d) - \frac{n-d}{2})/\sqrt{n-d} \tag{15}$$

donde $A(d) = \sum_{i=0}^{n-d-1} S_i \oplus S_{i+d}$ y $(1 \leq d \leq n/2)$.

II.3 Definición de caos

El concepto habitual de caos se refiere a lo impredecible. La palabra caos deriva de la raíz indoeuropea *ghn* o *ghen* del protoindoeuropeo. Debido a variaciones lingüísticas, el significado de la palabra se desplazó a desorden. Desde el punto de vista matemático y físico, más en particular en la teoría de sistemas dinámicos se habla de caos o comportamiento caótico para referirse a un comportamiento determinista aperiódico muy sensible a las condiciones iniciales (Kocarev y Lian, 2011).

II.3.1 Definición y propiedades del caos

Se le llama Teoría del Caos a la rama de las ciencias exactas, principalmente física y matemáticas, que estudia los comportamientos impredecibles de ciertos tipos de sistemas complejos y sistemas dinámicos muy sensibles a las variaciones en las condiciones iniciales. La Teoría del Caos plantea que el universo no sigue un patrón fijo y previsible, sino que se comporta de manera errática, que sus procesos y comportamiento dependen, en gran manera, de circunstancias inciertas. Esto plantea que una pequeña variación en el sistema o en un punto del mismo puede provocar que en un lapso futuro esté presente un comportamiento dinámico completamente diferente e impredecible.

Las dinámicas caóticas tienen una apariencia aleatoria aun cuando no constituyen hechos propiamente aleatorios, ya que son el resultado de dinámicas determinístas. La

teoría del Caos fue desarrollada muy especialmente a partir de los años 1960, si bien existen antecedentes como ciertos trabajos de Henri Poincaré (1854-1912).

Las caracteristicas principales de los sistemas caóticos son (Kocarev y Lian, 2011):

- **Dinámica no lineal.** El caos es un fenómeno exclusivo de los sistemas dinámicos no lineales. Un sistema lineal, sin importar el orden que tenga, no puede presentar este comportamiento.

- **Espectro amplio de frecuencia.** En el dominio de la frecuencia, una señal caótica presenta un espectro contínuo, muy parecido al ruido estocástico, pero con pico en las frecuencias dominantes.

- **Sensibilidad extrema a condiciones inicialiales.** A partir de condiciones iniciales diferentes, aunque muy cercanas unas de otras, las trayectorias correspondientes que se producen tienden a ser distintas o divergen exponencialmente conforme el tiempo transcurre, sin existir correlación alguna entre dichas trayectorias.

- **Presencia de atractores extraños.** Teniendo dos trayectorias que parten muy próximas en el espacio de fases, divergen rápidamente alejándose cada vez más. Con esto, se forman atractores los cuales tienen una estructura complicada, que reflejan dos tendencias opuestas: al tratarse de una atractor, las trayectorias deben converger hacia él, pero por tratarse de un caso de sensibilidad a las condiciones iniciales, las trayectorias deben diverger, al mismo tiempo, distanciendose cada vez mas. Para concebir estas figuras es necesario salir de la geometría Euclides, donde las dimensiones son números enteros, para dar cabida a formas irregulares y fragmentadas. A estas dimensiones el matemático Benoit Mandelbrot las llamó fractales.

- **Exponentes de Lyapunov positivos.** Los exponentes de Lyapunov, dan una medida de la expansión exponencial, en la cual, las órbitas cercanas se van apartando o acercando. Determinan la complejidad de un sistema no lineal. Se dice que el sistema es caótico, si el sistema tiene al menos un exponente de Lyapunnov positivo.

II.4 Sistemas caóticos en tiempo continuo

El atractor de (Lorenz, 1963), concepto introducido por Edward Lorenz en 1963, es un sistema dinámico determinista tridimensional no lineal, se observa que la ecuación 16 es una ecuación diferencial en el dominio del tiempo, obtenidas de las ecuaciones simplificadas de rollos de convección que se producen en las ecuaciones dinámicas de la atmósfera terrestre.

$$\begin{aligned} \frac{dx}{dt} &= a(y - x) \\ \frac{dy}{dt} &= x(b - z) - y \\ \frac{dz}{dt} &= xy - cz \end{aligned} \tag{16}$$

Para los parámetros $a = 10$, $b = 28$ y $c = 8/3$, el sistema exhibe un comportamiento caótico y muestra lo que actualmente se llama un atractor extraño, ver Figura 2.

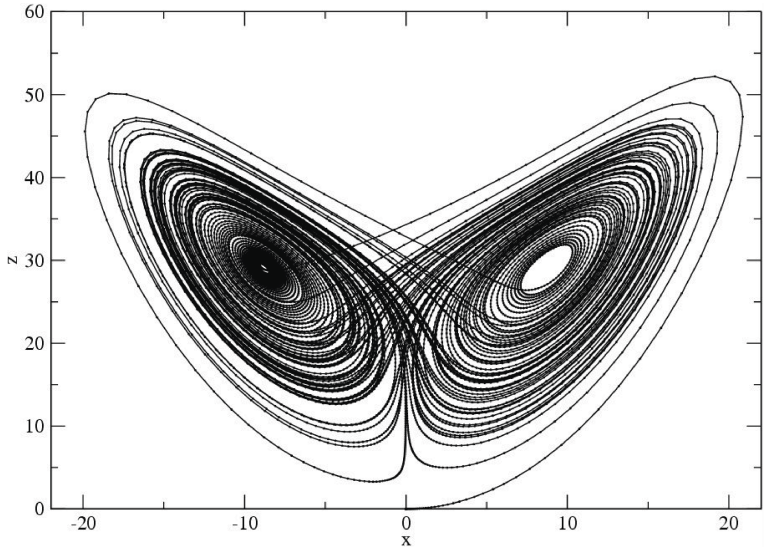


Figura 2: Atractor caótico del sistema Lorenz.

II.5 Sistemas caóticos en tiempo discretos

Una sistema discreto, es un sistema matemático cuyo dominio de definición es un conjunto numerable (o discreto). Es decir, es una definición:

$$f(s) = S \subseteq \mathbb{N} \rightarrow S' \tag{17}$$

Este tipo de sistemas son muy comunes en las ramas de matemática discreta y teoría de computación, dado el manejo finito de datos que en ellos se utiliza. Por ejemplo, las funciones de distribución de variable discreta en estadística son un conocido ejemplo de funciones discretas.

Los sistemas de ecuaciones Rössler (Rossler, 1979), Hénon (Hénon, 1976), Tinkerbell (Goldsztejn *et al.*, 2011), Karplan-Yorke (Kaplan y Yorke, 1979) y Logístic 2D (Grassberger y Procaccia, 2004) caen en la dinámica caótica discreta, los últimos 4 se ilustran en el apéndice A.

II.5.1 Mapeo de Rössler

El mapeo de Rössler (Rossler, 1979) es un sistema dinámico discreto en el tiempo. El mapeo tiene 3 estados (x_n, y_n, z_n) en el plano y mapea el siguiente punto.

$$\begin{aligned} x_{n+1} &= \alpha x_n(1 - x_n) - \beta(z_n + \gamma)(1 - 2y_n) \\ y_{n+1} &= \delta y_n(1 - y_n) + \zeta z_n \\ z_{n+1} &= \eta((z_n + \gamma)(1 - 2y_n) - 1)(1 - \theta x_n) \end{aligned} \tag{18}$$

El mapeo depende de 7 parámetros, α, β, γ, δ, ζ, η y θ, para mantener en régimen caótico el mapeo de Rössler, los valores son $\alpha = 3.8$, $\beta = 0.05$, $\gamma = 0.35$, $\delta = 3.78$, $\zeta = 0.2$, $\eta = 0.1$ y $\theta = 1.9$. Las condiciones iniciales son $x_0 = 0.1$, $y_0 = 0.3$ y $z_0 = 0.01$ (Rossler, 1979). La Figura 3 muestra su mapeo, el conjunto de ecuaciones de Rössler se programaron en Matlab.

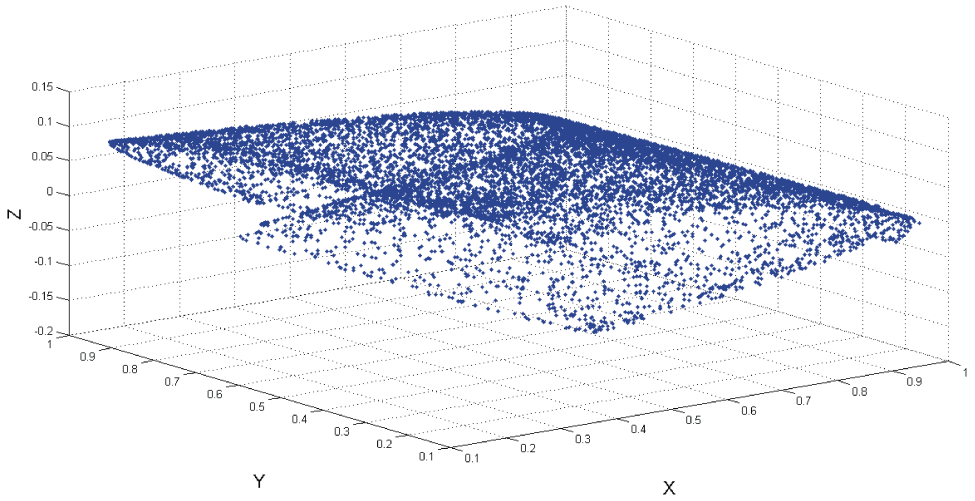


Figura 3: Atractor extraño del sistema caótico Rössler.

II.6 Sistemas caóticos y su implementación en sistemas de encriptado

La teoría del caos resultó ser una herramienta con aplicaciones a muchos campos de la ciencia y la tecnología. Gracias a estas aplicaciones el nombre se torna paradójico, dado que muchas de las prácticas que se realizan, con la matemática caótica tienen resultados concretos porque los sistemas que se estudian están basados estrictamente en leyes deterministas aplicadas a sistemas dinámicos. Los sistemas caóticos se caracterizan por sensibilidad extrema a condiciones inicialiales, comportamiento similar a la aleatoriedad, y un espectro en frecuencias contínuo. De esta forma, el caos tiene un potencial crítico en bloques funcionales de sistemas de comunicación digital: compreción, encriptado y modulación.

A inicios de los años $90's$ las investigaciones estaban enfocadas en la utilización de un sistema caótico para modular y encriptar al mismo tiempo. Eventualmente las investigaciones evolucionaron en dos áreas de investrigación distintas: modulación basada en caos y encriptado basado en caos.

Existe una relación estrecha entre el caos y la criptográfia, la Tabla I ilustra dicha relación, la ergocidad, la confusión, la sensibilidad en las condiciones iniciales y la difisión son las principales caracteristicas relacionadas con el espacio de clave.

Propiedades caóticas	**Propiedades Criptográficas**	**Descripción**
Ergocidad	Confusión	La salida tiene la misma distribución para cada entrada
Sensibilidad a condiciones iniciales	Difusión con un pequeño cambio en la clave	Un pequeño cambio en la entrada tiene un gran cambio en la salida
Propiedad de Mixeo	Difusión con pequeño cambio en un bloque de texto plano de todo el espacio de texto plano	Un pequeño cambio en el área local puede causar un cambio considerable en el espacio completo
Dinámica determinista	Pseudoaleatoriedad determinista	Un proceso determinístico puede causar cierto comportamiento pseudoaleatorio
Estructura compleja	Complejidad del algoritmo (fuerte contra ataques)	Un proceso simple tiene una muy alta complejidad

Tabla I: Comparación entre las propiedades del caos y la criptográfia (Alvarez y Li, 2006).

La Tabla II muestra las similitudes y diferencias entre los sistemas caóticos y los algoritmos de criptográficos. Las rondas de los algoritmos criptográficos conducen a las propiedades deseadas de difusión y confusión del algoritmo. Una importante diferencia entre los algoritmos de encriptado y los sistemas caóticos está en el encriptado de la información, el primero utiliza conjunto de números enteros y el segundo utiliza conjunto de números reales, además el segundo es extremadamente sencible a condiciones iniciales y variaciones paramétricas.

Algoritmos Criptográficos	**Sistemas Caóticos**
Espacio de fase: conjunto finito de enteros	Espacio de fase: conjunto de números reales
Métodos algebraicos	Métodos algebraicos
Rondas	Iteraciones
Difusión	Sensibilidad a condiciones iniciales
Realizaciones digitales por aritmética entera	Realizaciones digitales por aritmética no entera que aproxima sistemas continuos de valores reales
Seguridad y rendimiento	Seguridad y rendimiento

Tabla II: Similitudes y diferencias entre los sistemas caóticos y los algoritmos de cifrado (Mishkovski y Kocarev, 2011).

II.7 Sistemas embebidos

Un sistema embebido se define como una máquina computacional que emplea componentes de hardware y de software para la realización de una función específica en tiempo real. Se han convertido en parte fundamental de nuestra vida cotidiana automatizando tareas específicas y optimizando los recursos, por lo que su desarrollo es de suma importancia para que se brinden soluciones eficaces a algunos de los problemas que atañen a la sociedad actual.

Los dispositivos FPGA (del inglés Field Programmable Gate Array), Raspberry Pi 2 modelo B y el chip SPCE061A son sistemas embebidos, los cuales son usados en la realización de la presente tesis.

II.7.1 Arreglo de compuertas progrables en campo (FPGA)

Un FPGA es un dispositivo semiconductor que contiene componentes lógicos programables e interconexiones programables entre ellos. Los componentes lógicos programables pueden ser programados para duplicar la funcionalidad de compuertas lógicas básicas,

tales como AND, OR, XOR, NOT o funciones combinacionales más complejas tales como decodificadores o funciones matemáticas simples. En muchos FPGA, estos componentes lógicos programables (o bloques lógicos, según el lenguaje comúnmente usado) también incluyen elementos de memoria, los cuales pueden ser flip-flops simples o bloques de memoria más complejos (Woods *et al.*, 2008).

Una jerarquía de interconexiones programables permite a los bloques lógicos de un FPGA, ser interconectados según la necesidad del diseñador del sistema, algo parecido a un breadboard programable. Estos bloques lógicos e interconexiones pueden ser programados después del proceso de manufactura por el usuario/diseñador, así que el FPGA puede desempeñar cualquier función lógica necesaria.

Las FPGA son generalmente más lentas que sus contrapartes, los circuitos integrados de aplicaciones específicas (ASIC por sus siglas en inglés), no pueden soportar diseños muy complejos, y consumen más energía. Sin embargo, tienen muchas ventajas tales como la reducción del tiempo para la salida al mercado de productos, la habilidad para ser reprogramadas después de haber salido al mercado a fin de corregir posibles errores, y reduce los costos de ingeniería tales como investigación, diseño y prueba de un producto nuevo.

II.7.2 Aplicaciones de las FPGA

Las FPGA encuentran aplicaciones en muchas áreas donde se requiera del paralelismo ofrecido por su arquitectura. Las aplicaciones de las FPGA incluyen a los DSP (Digital Signal Processor), radio definido por software, sistemas aeroespaciales y de defensa, prototipos de los ASIC, sistemas de imágenes para medicina, sistemas de visión para computadoras, reconocimiento de voz, bioinformática, emulación de hardware de computadora, y tienen un crecimiento de aplicaciones en otras áreas.

II.7.3 Arquitectura de FPGA

Históricamente, los arreglos de lógica programable (PLA) fueron los primeros dispositivos lógicos programables. Los PLA contenían compuertas AND y OR con una estructura de dos niveles (ver Figura 4), con conexiones programables por el usuario. La estructura de los PLA sufrió mejoras y su costo disminuyó con la introducción de dispositivos lógicos de arreglo programable (PAL), en los que el "array"de puertas OR es fijo (no programable) y los dispositivos de arreglos genéricos (GAL). Actualmente, tales dispositivos se denominan de manera genérica como dispositivos de lógica programable y se puede decir que son los MSI("Medium Scale Integration") de la industria de la lógica programable.

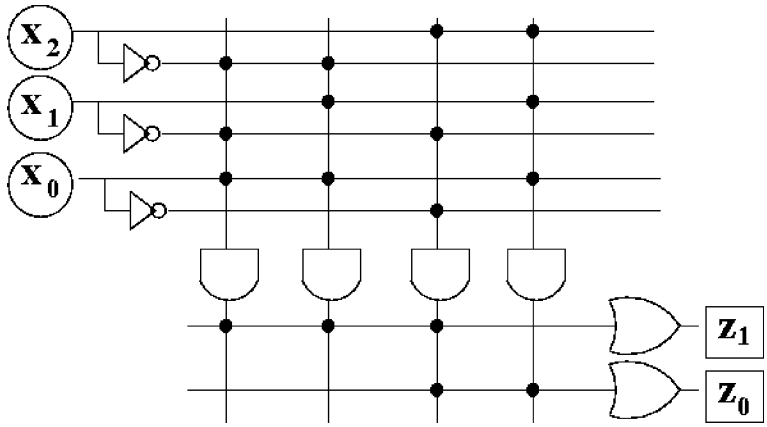


Figura 4: PLA de 3x4x2, arreglo de puertas AND y OR programables.

Estos dispositivos, antecesores de las FPGAs, ya se configuraban haciendo uso de un lenguaje de descripción de hardware. Sin embargo, la configuración de los primeros circuitos se realizaba, inicialmente, mediante la destrucción física de unos fusibles y sólo se podían programar una vez, con todas las limitaciones que esto suponía. Con el paso del tiempo, la tecnología permitió el borrado o desconfiguración del dispositivo mediante exposición a rayos UV o bien eléctricamente y, por tanto, su reconfiguración.

La siempre creciente capacidad de los circuitos integrados creó una oportunidad para los fabricantes de diseñar PLDs más grandes para aplicaciones mayores áreas de diseño digital. Sin embargo, un dispositivo demasiado grande usando una estructura alambrada del estilo sería demasiado lento y, desde el punto de vista del fabricante, no se haría un uso efectivo en cuanto al costo del área del chip.

Los fabricantes idearon un CPLD que consistía, básicamente, en una colección de PLDs individuales en un simple chip, acompañado de una estructura de interconexión programable que permitía que los PLD fueran conectados entre sí en el chip de la misma manera que un diseñador conectaría externamente PLDs individuales, ver Figura 5 (Parnell y Mehta, 2002).

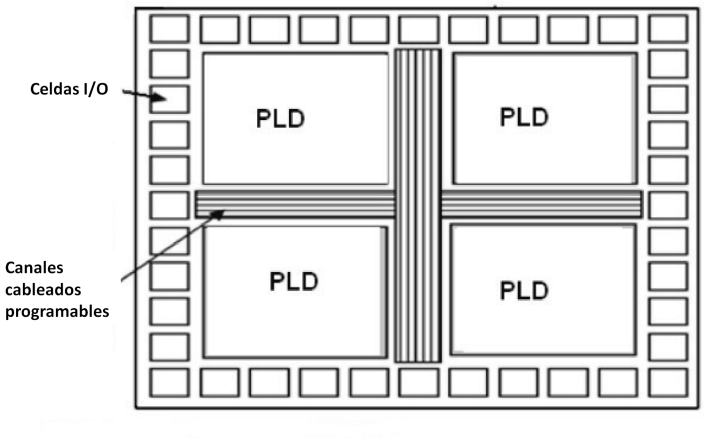


Figura 5: Arquitectura de un CPLD.

Xilinx (Xilinx, 2016) sacó al mercado un "Gate Array"programable en campo, es decir, la sustitución de la interconexión fija de los "Gate Arrays"por una serie de pistas metálicas conectables por transistores de paso controlados por un conjunto de bits de control almacenados en una memoria interna. Así, estos dispositivos surgen en 1985

con el nombre de LCA (Logic Cell Array), aunque posteriormente se renombraron como FPGA.

En una FPGA la lógica se divide en un gran número de bloques lógicos programables que son individualmente más pequeños que un PLD. Se encuentran distribuidos a través de todo el chip en un mar de interconexiones programables y todo el arreglo se encuentra rodeado de bloques de E/S programables (IOBs). Un bloque lógico programable (CLB o slice) de FPGA es menos eficiente que un PLD típico, pero un chip FPGA contiene muchos más bloques lógicos que los PLD que contiene un CPLD del mismo tamaño (Parnell y Mehta, 2002).

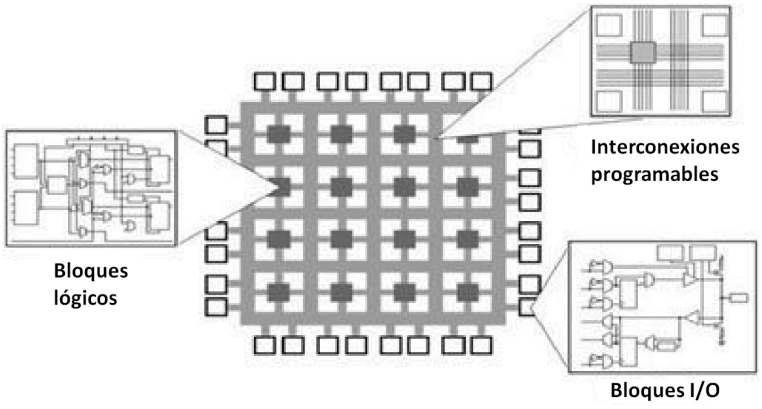


Figura 6: Arquitectura de un FPGA.

- **Slide:** En los "slices"se realiza la mayor parte de la funcionalidad de la FPGA y suelen estar agrupados de 2 en 2 o de 4 en 4 formando bloques lógicos configurables (CLBs). Dentro de este componente encontramos los módulos LUT, registros y multiplexores programables en un número que depende de familia de FPGA, pero la arquitectura básica común es la que se muestra en la Figura 7.

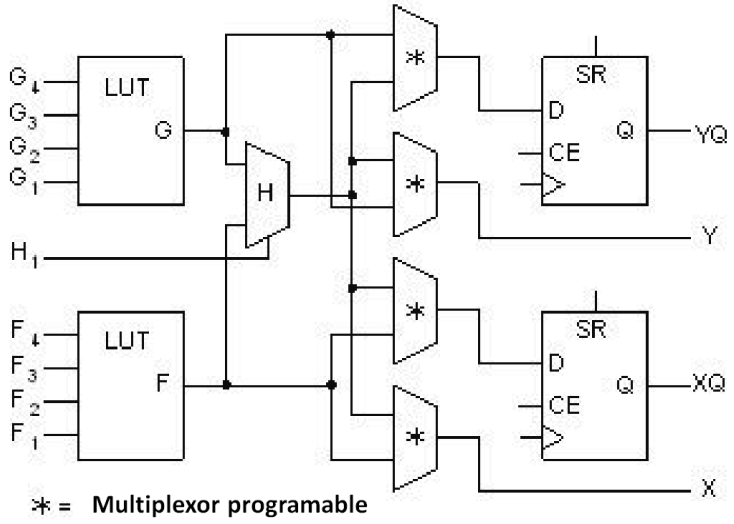


Figura 7: Slide de una FPGA.

Los elementos programables más importantes son los generadores reprogramables de función lógica, realizadas por las denominadas LUT (Look-up Table) o tablas de búsqueda, que son celdas de memoria SRAM y multiplexores para seleccionar la salida, ver Figura 8.

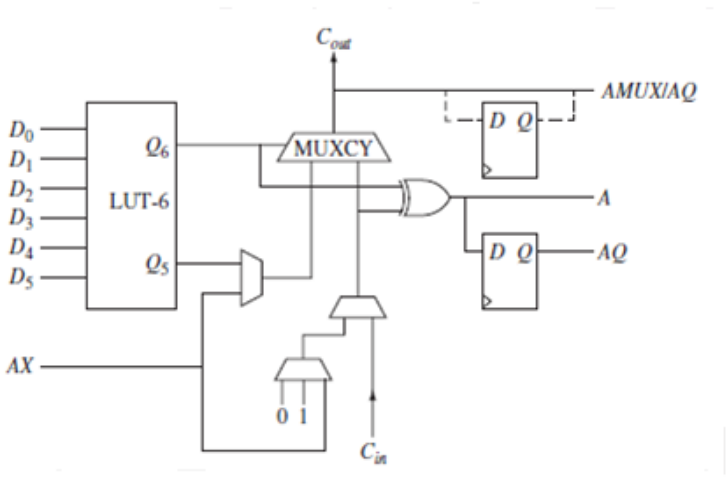


Figura 8: LUT de 6 entradas de proposito general.

- **IOBs:** Los bloques de Entrada/Salida de las FPGAs cumplen la misma función que las macroceldas de salida en otros dispositivos lógicos programables, pero con más controles lógicos, entre los que se incluyen, configuraciones de entrada y salida combinacionales o registradas, alta impedancia, elementos de retardo, controles analógicos y otros.

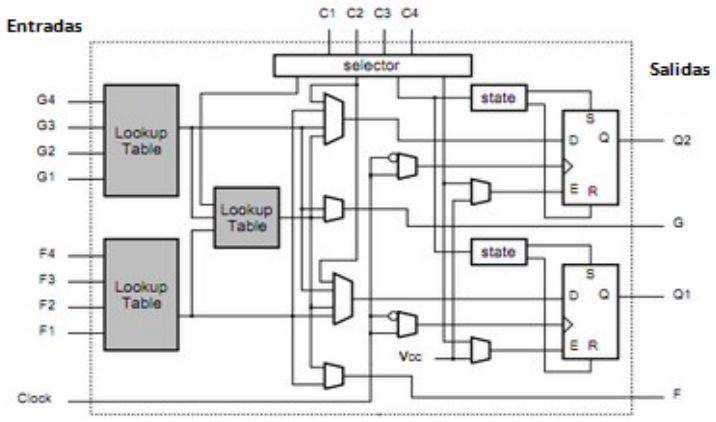


Figura 9: Bloque de salida/entrada de la arquitectura FPGA.

- **Interconexión programable:** Según la Figura 6, cada CLB en la FPGA se encuentra incrustado en la estructura de interconexión, que se componen en realidad de cables con conexiones programables para ellos.

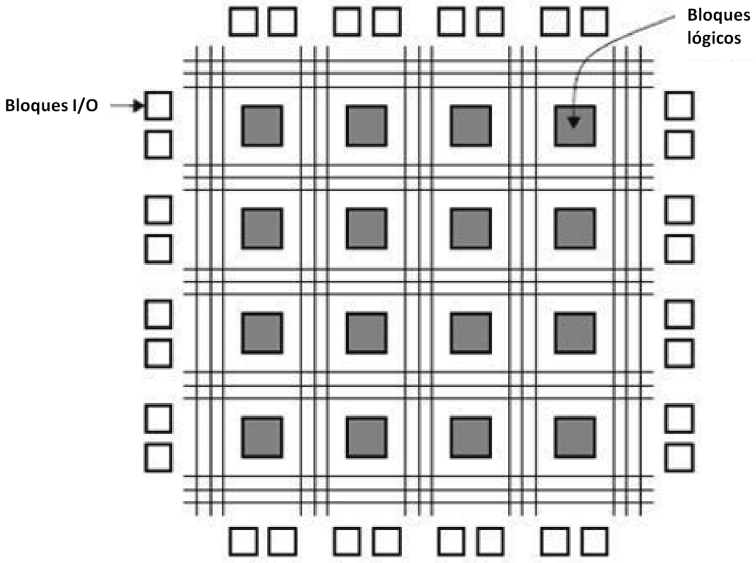


Figura 10: Estructura de interconexión de una FPGA.

- **Otros componentes dentro de una FPGA:** Las FPGAs muchas veces se evalúan en función de la flexibilidad de sus arquitecturas y la consistencia de los resultados obtenidos de un ajuste después de que se han efectuado pequeños cambios de diseño. De esta manera, los fabricantes proporcionan recursos adicionales en sus arquitecturas para ayudar a asegurar resultados consistentes e implementar algunos sistemas de manera muy eficiente.

Actualmente, las compañías Xilinx, Altera (Altera, 2016) y Lattice (lattice, 2016) fabrican FPGAs con diversos dispositivos embebidos dentro del dispositivo, como memorias, multiplicadores, DCMs (administradores de reloj), e incluso microprocesadores.

II.7.4 FPGA Spartan-3E XC3S1600E

La tarjeta Spartan 3E-1600 Development suministra un poderoso y altisimo avance contenido dentro del desarrollo de la plataforma para diseños enfocados a la Spartan 3E FPGA de Xilinx. En la Tabla III se muestran las características principales del chip XC3S1600E tomadas de las especificaciones del fabricante (Xilinx, 2009).

Dispositivo	**Atributo**	**Unidades**	**Atributo**	**Unidades**
XC3S1600E	Compuertas del Sistema	$1600K$	RAM distribuida (bits)	$231K$
	Celdas lógicas equivalentes	$33,192$	Bloques de RAM (bits)	$648K$
	Filas	76	Multiplicadores dedicados	36
	Columnas	58	DCMs	8
	Total CBLs	$3,688$	I/O Máxima	376
	Total Slice	$14,752$	Pares de I/O Máxima	156

Tabla III: Atributos de la FPGA Spartan-3E XC3S1600E

Las características de la tarjeta Spartan 3E-1600 Development (ver Figura 11), se ilustran en la Tabla IV (Xilinx, 2011).

II.7.5 Lenguaje de descripción de hardware de los FPGAs

Para describir cualquier circuito digital y principalmente para programar un PLD, FPGA, ASIC y similares, puede utilizarse VHDL o Verilog, lenguajes descriptivos que a continuación se abordarán.

Verilog es un lenguaje de descripción de hardware (HDL, del Inglés Hardware Description Language) usado para modelar sistemas electrónicos. El lenguaje, algunas veces llamado Verilog HDL, soporta el diseño, prueba e implementación de circuitos analógicos, digitales y de señal mixta a diferentes niveles de abstracción.

Los diseñadores de Verilog querían un lenguaje con una sintaxis similar a la del lenguaje de programación C, de tal manera que le resultara familiar a los ingenieros y

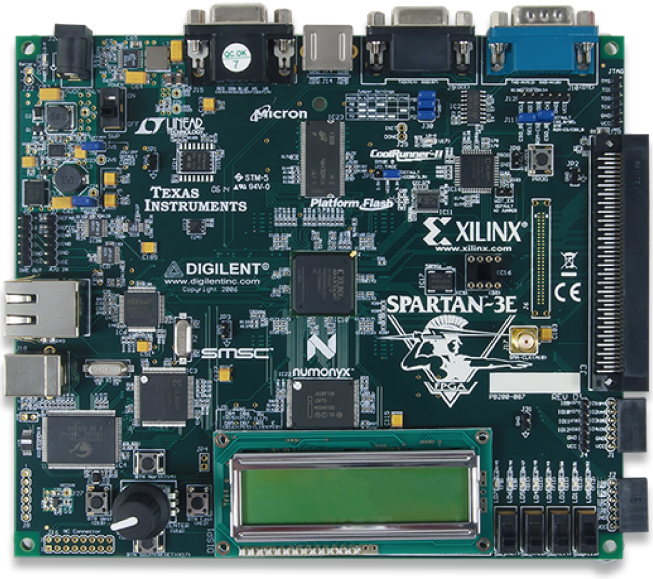


Figura 11: Tarjeta Spartan 3E-1600 Development.

así fuera rápidamente aceptada, aun así no fue aceptada al momento. El lenguaje tiene un preprocesador como C, y la mayoría de palabras reservadas de control como "if", "while", etc, son similares. El mecanismo de formateo en las rutinas de impresión y en los operadores del lenguaje (y su precedencia) son también similares.

Es un lenguaje obsoleto, actulmente está siendo actualizado con la intención de que sea el nuevo estándar de IEEE (Institute of Electrical and Electronics Engineers).

En cambio el VHDL, es un lenguaje definido por el IEEE (ANSI/IEEE 1076-1993) usado por ingenieros y científicos para describir circuitos digitales o modelar fenómenos científicos respectivamente. VHDL es el acrónimo que representa la combinación de VHSIC y HDL, donde VHSIC es el acrónimo de "Very High Speed Integrated Circuit"y HDL es a su vez el acrónimo de "Hardware Description Language". Originalmente, el

Tarjeta	Característica
Spartan 3E-1600 Development Board	Switches, Botones, Boton de perilla
	LEDs
	Fuente de reloj
	Configuración del FPGA
	Pantalla LCD
	Puerto VGA
	Puerto serial RS-232
	PS/2 puerto para ratón/teclado
	Convertidor Digital á Análogo
	Circuito convertidor Análogico a Digital
	Memoria PROM
	Memoria SPI
	Memoria DDR SDRAM
	10/100 puerto Ethernet
	Puertos de expanción
	Memoria 1-wire EEPROM
	CPLD

Tabla IV: Características Spartan 3E-1600 Development.

lenguaje VHDL fue desarrollado por el departamento de defensa de los Estados Unidos a inicios de los años 80's basado en ADA, con el fin de realizar simulación de circuitos eléctricos digitales; sin embargo, posteriormente se desarrollaron las herramientas de síntesis e implementación en hardware a partir de los archivos ".VHD", como lo es el "ISE Design Suite"de Xilinx el cual es un software de diseño de alto desempeño que aporta un rápido desarrollo en la producción de tecnologías.

El ISE trabaja con los dos lenguajes mencionados anteriormente más las Propiedades Intelectuales (IP), son circuitos digitales creados con diferentes propósitos listos para ser implementados en el desarrollo tecnolgíco en cualquiera de los dos lenguajes.

Actualmente no solo Xilinx proporciona software, también Altera que es su más cercano competidor junto con Lattice. Y brindan un ToolBox para Matlab que genera el código ".VHD", sintetiza y puede programar los FPGA. De acuerdo a la arquitectura

que se tiene en las instalaciones de la FIADUABC, se ha seleccionado el software de Xilinx que genera la ToolBox System Gnerator ("SysGen") en Matlab. La siguiente sección se brinda una introducción al ToolBox.

II.7.6 Lenguaje descriptivo de alto nivel

"SysGen"es una herramienta de modelado a nivel de sistema que facilita el diseño de hardware del FPGA. Extiende el desarrollo de modelado de Simulink a un nivel de diseño de Hardware. La herramienta provee un nivel alto de programación que automáticamente compilada el código con sólo hacer click un botón. La herramienta también ofrece el acceso a los recursos del FPGA por medio de una programación de bajo nivel, permitiendo la construcción de eficientes diseños (Xilinx, 2012).

II.7.7 Flujo de diseño usando "SysGen"

"SysGen"puede ser útil en muchas situaciones. Es posible que desee explorar una algoritmo sin implementar el diseño en el hardware. Otras veces se puede planear el uso de un diseño del "SysGen"como parte de algo más grande. Una tercera posibilidad es que un sistema de diseño del "SysGen"es completa en su propio derecho, y se va a utilizar en hardware FPGA. A continuación describen las tres posibilidades (Xilinx, 2012):

- **Exploración de Algoritmos:** es especialmente útil para la exploración de algoritmos, la creación de prototipos de diseño, y análisis de modelos. Cuando estos son los objetivos, se puede utilizar la herramienta para dar cuerpo a un algoritmo con el fin de tener una idea de los problemas de diseño que probablemente tendrán que ajustarse, y tal vez para estimar el costo y el rendimiento de una implementación en hardware.

- **Implementación de una parte de un Diseño:** A menudo se utiliza para implementar una parte de un diseño más grande. Por ejemplo, es un buen escenario para poner en práctica los caminos y el control de datos, pero es menos adecuado para las interfaces externas sofisticados que tienen estrictos requisitos de tiempo. En este caso, puede ser útil para poner en práctica las partes del diseño, usando el "SysGen", implementar otras partes fuera, y luego combinar las partes en un todo de trabajo.

- **Implementación de un Diseño:** Muchas veces, todo lo necesario para un diseño está disponible en el interior del generador del sistema. Para un diseño de este tipo, al pulsar el botón Generar instruye Sistema Generador de traducir el diseño en HDL, y para escribir los archivos necesarios para procesar el HDL utilizando herramientas.

II.7.8 Sistema de modelado en "SysGen"

Permite diseños de hardware específicas del dispositivo que se construirá directamente en un entorno flexible de modelado de sistemas de alto nivel (Xilinx, 2012).

- **"Blocksets SysGen":** Describe cómo se organizan los bloques del "SysGen"en las bibliotecas de Matlab, y cómo los bloques se pueden parametrizar y utilizados.
- **Tipo de Señales:** Describe los tipos de datos utilizados por el "SysGen"y formas en que los tipos de datos pueden ser asignados automáticamente por la herramienta o por el usuario.
- **Modelado de Bit Verdadero y Ciclo Verdadero:** Especifica la relación entre la simulación basada en Simulink de un modelo de generador del sistema y el comportamiento del hardware que puede ser generada de ella.
- **Relojes y Temporizadores:** Describe cómo los relojes se implementan en hardware, y cómo su aplicación está controlada en el interior del "SysGen". Explica cómo Sistema Generador traduce un modelo Simulink multivelocidad trabajar en el hardware del reloj síncrono.
- **Mecanismos de Sincronización:** Describe los mecanismos que se pueden utilizar para sincronizar el flujo de datos a través de los elementos de la ruta de datos en un diseño de alto nivel del "SysGen", y describe cómo se pueden implementar funciones de ruta de control.
- **Máscara de Bloques y Parámetros:** Explica cómo se crean los sistemas y subsistemas parametrizados en Simulink.
- **Estimación de Recursos:** Describe cómo generar estimaciones del hardware necesario para implementar un diseño de sistema de generador.

II.7.9 Raspberry Pi 2 modelo B

El Raspberry Pi 2 B es un pequeño dispositivo pensado para la enseñanza de programación y computación de los jovenes, ver Figura 12, desarrollado en Reino Unido por la fundación Raspberry Pi (Pi, 2016).

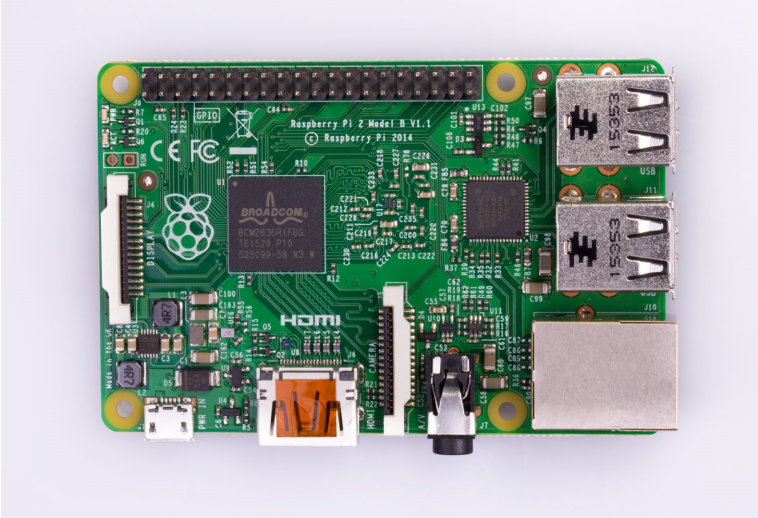


Figura 12: Tarjeta Raspberry Pi 2 modelo B (Pi, 2016).

La fundación Raspberry Pi surge con un objetivo de desarrollar el uso y entendimiento de las computadoras en los niños y jovenes. La idea es conseguir computadoras portables y muy economicas que permitan a los niños usarlos sin miedo, abriendo su mentalidad y educándolos en la ética del ábrelo y mira cómo funciona. El ideólogo del proyecto, David Braven, un antiguo desarrollador de videojuegos, afirma que su objetivo es que los niños puedan llegar a entender el funcionamiento básico de las computadoras de forma divertida, y sean ellos mismos los que desarrollen y amplíen sus dispositivos. El co-fundador de la fundación es Eben Upton, un antiguo trabajador de la empresa Broadcom, el cual es el responsable de la arquitectura de software y hardware de la Raspberry Pi (Pi, 2016).

La fundación da soporte para las descargas de las distribuciones para arquitectura ARM, Raspbian (derivada de Debian), RISC OS y Arch Linux; y promueve principalmente el aprendizaje del lenguaje de programación Python, y otros lenguajes como Tiny BASIC, C y Perl (Halfacree y Upton, 2012).

II.7.10 Características del Raspberry Pi 2 modelo B

En la Tabla V se muestran las características en el modelo 2 B. La placa cuenta con dimensiones de 8.5 por 5.3 cm, un chip integrado Broadcom BCM2836, que contiene un procesador ARM11, un procesador gráfico VideoCore IV, y 1GB de memoria RAM. Tiene una salida de vídeo HDMI, una salida de audio a través de un minijack 3.5 mm, conexión Ethernet 10/100, 4 puertos USB incluidos, por último cuenta con el puerto GPIO (General Purpose Input/Output) permite a la Raspberry Pi comunicarse con el exterior tanto para activar elementos como para leer el estado de los mismos, UART, I2C, SPI y PWM (Pi, 2016).

Recurso	**Característica**
SoC	Broadcom BCM2836
CPU	ARM11 ARMv7 ARM Cortex A7, 4 núcleos @ 900 MHz.
GPU	Broadcom VideoCore IV 250 MHz. OpenGL 2.0
RAM	1 GB L PDDR2 SDRAM 450 MHz.
USB 2.0	4
Salidas de vídeo	HDMI 1.4 @ 1920x1200 píxeles
Almacenamiento	microSD
bus de expansión GPIO	40 pines

Tabla V: Características de RPI 2 Modelo B (Pi, 2016)

La Figura 13 muestra la descripción de pines de la Raspberry Pi 2 B. Tiene 3 diferentes columnas, la columna del centro identifica la numeración de la cabecera, la contigua la función del pin y la externa el nombre asignado en la programación GPIO.

GPIO.BCM	Function	GPIO.BOARD		Function	GPIO.BCM
<50mA	3V3	1	2	5V	
BCM GPIO02	SDA1 ARM	3	4	5V	
BCM GPIO03	SCL1 ARM	5	6	GND	
BCM GPIO04		7	8	TX	BCM GPIO14
	GND	9	10	RX	BCM GPIO15
BCM GPIO17	SPI1 CE1	11	12	PWM0/SPI1 CE0	BCM GPIO18
BCM GPIO27		13	14	GND	
BCM GPIO22		15	16		BCM GPIO23
<50mA	3v3	17	18		BCM GPIO24
BCM GPIO10	SPI0 MOSI	19	20	GND	
BCM GPIO9	SPI0 MISO	21	22		BCM GPIO25
BCM GPIO11	SPI0 SCLK	23	24	SPI0 CE0	BCM GPIO08
	GND	25	26	SPI0 CE1	BCM GPIO07
BCM GPIO00	SDA0 VC	27	28	SCL0 VC	BCM GPIO01
BCM GPIO05		29	30	GND	
BCM GPIO06		31	32	PWM0	BCM GPIO 12
BCM GPIO13	PWM1	33	34	GND	
BCM GPIO19	SPI1 MISO/PWM1	35	36	SPI1 CE2	BCM GPIO16
BCM GPIO26		37	38	SPI1 MOSI	BCM GPIO20
	GND	39	40	SPI1 SCLK	BCM GPIO21

Figura 13: Descripción de pines de la Raspberry Pi 2 B.

II.7.11 Chip de reconocimiento de voz

Un chip de reconocimiento de voz, es una herramienta computacional capaz de procesar la señal de voz emitida por el ser humano y reconocer la información contenida en ésta. Los sistemas de reconocimiento de voz pueden ser primeramente de dos tipos dependiente e independiente del locutor. Dependiente del locutor significa que si dos personas con la misma lengua mandan comandos al chip, el chip sólo reconoce al que haya programado sus comandos con su propia voz, e ignorará los comandos del segundo, ya que es sensible a, por ejemplo la diferencia de timbre, énfasis o dialecto lingüístico de una misma lengua. Independiente del locutor implica que, sean dos personas mandando lenguajes con la misma lengua, el chip reconocerá el comando sin hacer distinción de la fuente. El chip de reconocimiento de voz SPCE061A, ver Figura 14, implementado en el sistema cifrado embebido como llave de acceso es fabricado por la compañia Sunplus (Sunplus Technology Co., 2016), es un microcontrolador de uso dedicado en aplicaciones de procesamiento digital de sonido, reconocimiento de voz e independiente del hablante (Sunplus Technology Co., 2002).

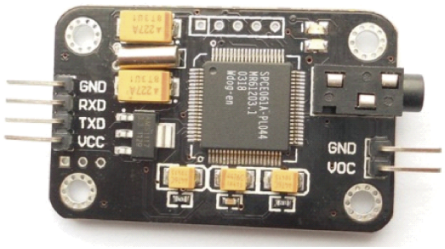


Figura 14: Chip de reconocimiento de voz SPCE061A.

En el apendice D se presenta el método de la configuracón del chip CRV.

II.8 Revisión bibliográfica

En un lapso de 20 años muchas técnicas de cifrado basados en caos han sido investigadas y desarrolladas como son la comunicación secreta basada en caos, encriptado "block/stream"basado en caos, generación numérica aleatoria basada en caos, etc.

Un Generador Binario Pseudoaleatorio Caótico (CPRBG) es un algoritmo deterministico caótico el cual parece producir secuencias numéricas binarias al azar, pero no lo hace realmente. Las secuencias binarias pseudo-aleatorias no muestran ningún patrón o regularidad aparente desde un punto de vista estadístico, a pesar de haber sido generadas por un algoritmo completamente determinista, en el que las mismas condiciones iniciales producen siempre el mismo resultado. Por ejemplo, en la serie de documentos (Kocarev y Jakimoski, 2003), (Cristina *et al.*, 2012), (Addabbo *et al.*, 2011), (Wang *et al.*, 2015) y (Liu *et al.*, 2015), los autores proponen diferentes métodos para generar la secuencia pseudoaleatoria, los resultados son sometidos a la prueba estadística FIPS 140-2, sin embargo no son utilizadas en aplicaciones de cifrado caótico.

Algunas aplicaciones de las técnicas de cifrado caótico son la autentificación o cifrado de imágenes basado en caos, codificación video/audio, protección de derechos de autor multimedia, etc. Por ejemplo en (Min *et al.*, 2014), diseñan en Matlab un CPRGB con una metodología de sincronización para cifrar imágenes, a partir de los análisis al CPRGB y criptograma obtuvieron resultados competitivos, pero no declaran una implementación analógica o digital.

Una implementación analógica de un algoritmo de cifrado caótico enfrenta el cambio de valores en los componentes, ya que varían con el tiempo, temperatura, humedad, etc., por lo tanto es difícil utilizar las técnicas de recuperación de información de forma precisa, en cambio la implementación digital presenta algunas ventajas, precisión y la posibilidad de integración en aplicaciones actuales especialmente para cifrar datos y

comunicaciones seguras entre sistemas embebidos de última generación.

Se ha demostrado la implementación de CPRBG en Arduino (Volos, 2013), utilizando un par de mapeos Logístic y una técnica de "skewed"con la operación binaria XOR.

Por otro lado, se han desarrollado implementaciones por medio de hardware programable como los FPGA, los cuales tienen un excelente equilibrio entre el poder de cómputo y flexibilidad de procesamiento que provee (Ref. (Tanougast, 2011), (Qi *et al.*, 2011), (Dabal y Pelka, 2014) ,(Azzaz *et al.*, 2011), (Mansingka *et al.*, 2013) y (Fang *et al.*, 2014)). Las empresas desarrolladoras del hardware proveen las licencias del software necesario, incluso han desarrollado herramientas de programación de alto nivel para Matlab. En (Tlelo-Cuautle *et al.*, 2015) implementan en FPGA , de la familia de Altera, un sistema de cifrado de imágenes utilizando un sistema caótico de 6 estados empleando una metodología de sincronización Hamiltoniana, el dasarrollo del sistema está basado en Matlab Simulink con una co-simulación por medio de ACTIVE-HDL, una herramienta de Altera. En (Dabal y Pelka, 2015) implementan un CPRNG (Chaotic Pseudo-random Number Generator) en FPGA utilizando la herramienta "SysGen".

II.9 Conclusión

En este capítulo se presentaron las bases teóricas de la criptografía, marco principal de la presente tesis. Se presentaron las características de los sistemas caóticos como una propuesta alternativa para generar sistemas criptográficos. Se presentó el mapeo caótico Rössler como ejemplo. Se presentó la tecnología utilizada en la realización de la presente tesis, el chip FPGA, el Raspberry Pi 2 y el chip de reconocimiento de voz. Por último se presentó una revisión bibliográfica sobre trabajos relacionados a este tema de tesis.

Capítulo III

Desarrollo del sistema de cifrado propuesto

El hardware FPGA, que se ha usado en la implementación de los sistemas caóticos discretos en este actual trabajo de tesis, forma parte de la familia Xilinx , Spartan 3E-1600 Development Board es una tarjeta de desarrollo fabricada por la empresa Digilent.

Xilinx ofrece un software gratis "ISE Design Suite: WebPACK Edition"para el desarrollo de los programas y el programador de sus productos, es la empresa pionera en la tecnología FPGA. Una de las ventajas de trabajar con la familia Xilinx es la importación de un ToolBox a Matlab, esto es permitido si y solo si se utiliza la versión "ISE Design Suite: System Edition", da a lugar el modelado de sistemas y la generación automática del código a partir de diagramas de bloques en Simulink.

III.1 Sistema embebido propuesto para el encriptado

El sistema embebido propuesto es utilizado tanto como encriptador o desencriptador de imágenes y está constituido por un FPGA, un Raspbery Pi 2 y un Chip de Reconocimiento de Voz, tal como se ilustra en la Figura (15).

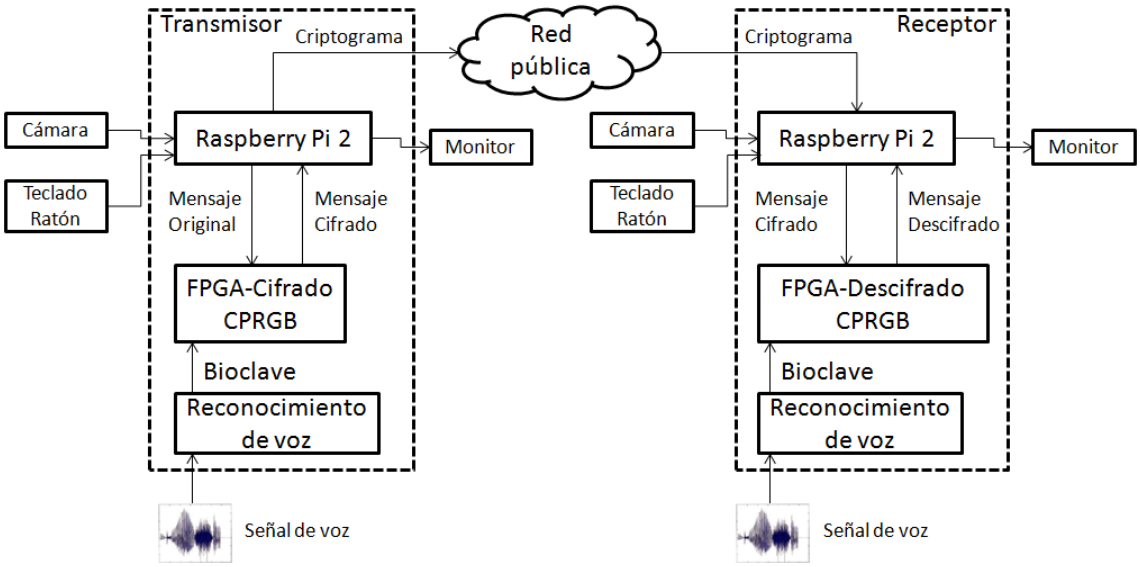


Figura 15: Sistema embebido de encriptado propuesto propuesta para el cifrado.

El Chip de Reconocimiento de Voz SPCE061A, fabricado por la compañia Sunplus, es un microcontrolador de uso dedicado en aplicaciones de procesamiento digital de sonido y reconocimiento de voz (Liu y Chen, 2014), el cual es usado para el inicio del proceso de encriptado, reconociendo la voz o la clave de inicio del proceso. El SPCE061A tiene un puerto serial RS-232 el cual se comunica con el FPGA.

El proceso de encriptado se logra por medio del FPGA, el cual configura al SPCE061A por el puerto serial RS-232 y está esperando la clave de inicio. Cuando un usuario utiliza la clave de voz correcta, el SPCE061A genera una clave de inicio y la manda por el puerto RS-232 al FPGA, una vez obtenida la clave de acceso el FPGA se

comunica por otro puerto serial con la Raspberry Pi 2 modelo B, el usuario selecciona una imagen de NxN píxeles y envia por el puerto RS-232 cada uno de los píxeles al FPGA. El FPGA calcula una secuencia binaria pseudoaleatoria por cada pixel que adquiere y cifra pixel por pixel utilizando la operación binaria XOR, cada pixel encriptado es regresado al Raspberry Pi 2 y se forma el criptograma de la imagen. El FPGA que se ha utilizado para hacer el experimento fue una tarjeta de desarrollo Spartan 3E-1600 de Xilinx.

La interfaz gráfica que utiliza la Raspberry Pi 2 permite de una interacción sencilla con el usuario, selecciona ya sea una imagen para encriptar o un criptograma para desencriptar. También puede capturar una nueva imagen desde una cámara digital o web cam. El programa manda un pixel de la imagen seleccionada y lo recibe encriptado o desencriptado, lo guarda en la memoria y al finalizar forma una imagen MxN a partir de los píxeles alojados en la memoria.

En la Figura 16 se muestra el arreglo experimental de la propuesta.

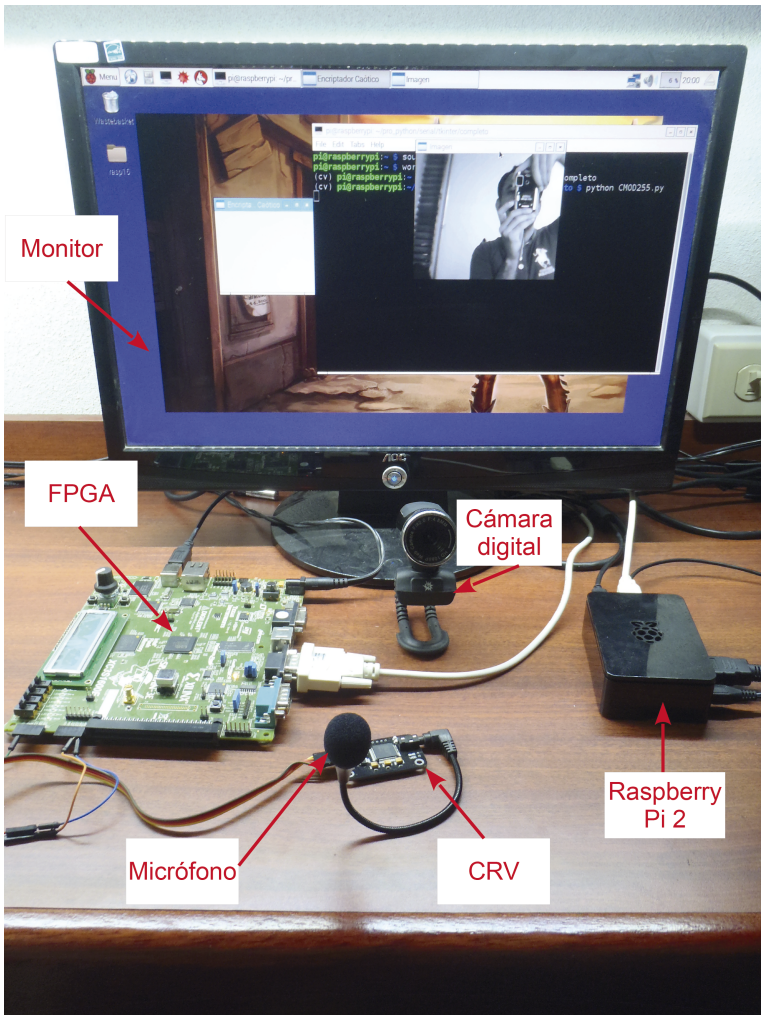


Figura 16: Arreglo experimental del sistema embebido de encriptado propuesto.

III.2 Algoritmo embebido en FPGA propuesto para el encriptado

En la Figura 17 se muestra en un diagrama a bloques el algoritmo embebido para llevar acabo el proceso de encriptado caótico de imágenes digitales. El sistema se dividió en 5 algoritmos, el primero fue una interfaz gráfica (IG) programada en lenguaje Python para la plataforma Raspberry Pi 2 modelo B y los 4 siguientes fueron diseñados en VHDL e implementados en entidades de una FPGA: UART1, UART2, Entidad de

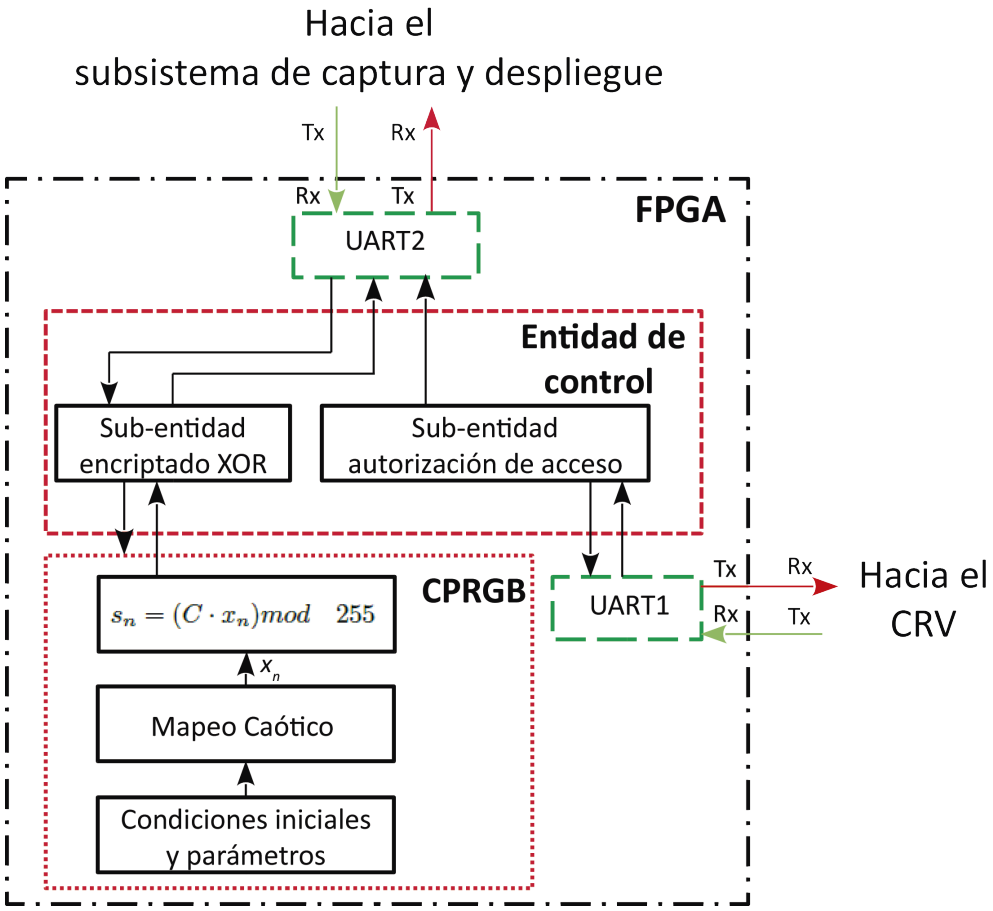


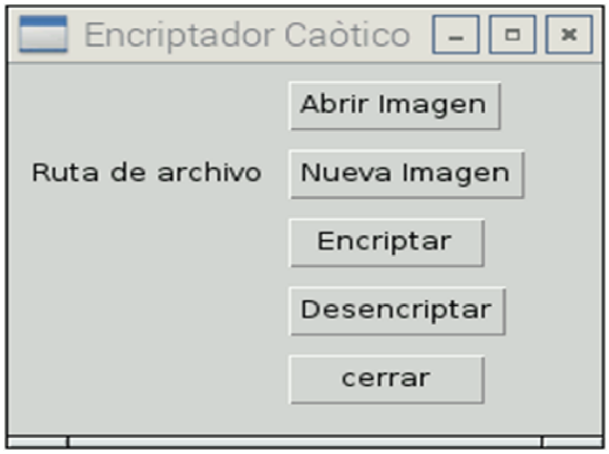
Figura 17: Diagrama a bloques del algoritmo embebido en FPGA de encriptado-desencriptado de imágenes digitales.

Control y CPRGB.

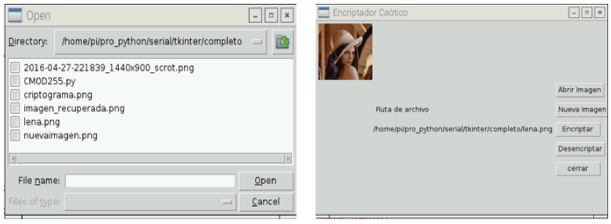
Las siguientes subsecciones describen cada uno de los algoritmos implementados en FPGA excepto UART1 y UART2 fueron diseñados a partir del protocolo de comunicación RS-232 integradás por la librería GNU Cosmiac (2014).

III.2.1 Subsistema de captura y despliegue

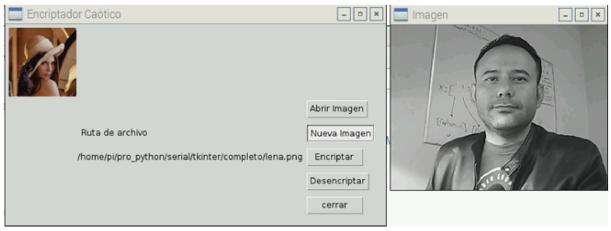
La Figura 18 ilustra la interfaz gráfica, fue programada en lenguaje Python 3.4 en la Raspberry Pi 2, sobre el sistema operativo Raspbian, utilizando las siguientes librerías Tkinter, filedialog, cv2, time, numpy, PIL, subproces, sys y pyserial.



(a) Menu de la interfaz.



(b) Menu para seleccionar imagen a cifrar o descifrar.



(c) Captura de imagen por la cámara.

Figura 18: Interfaz gráfica en Raspberry Pi 2 B.

La interfaz implementada en Raspberry Pi 2 modelo B consta de 4 sub-rutinas, ver Figura 18a, “Abrir Imagen”, “Nueva Imagen”, “Encriptar”y “Desencriptar”.

La Figura 18b muestra el resultado de la rutina Abrir Imagen, abre una ventana de dialogo por la cual se selecciona la imagen a encriptar o desencriptar. La Figura 19 muestra el diagrama de flujo de la sub-rutina, inicia abriendo la imagen, en la seleccción de la imagen, ya sea una plantilla o un criptograma, se abre un cuadro de díalogo para explorar la memoria micro del Raspberry Pi 2 modelo B y después carga la imagen en la interfaz gráfica y en la memoria del programa para un futuro proceso.

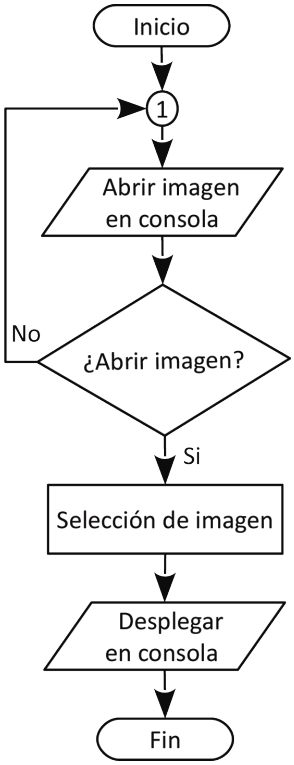


Figura 19: Diagrama de flujo de la sub-rutina “Abrir Imagen”.

La Figura 18b ilustra la operatividad de la cámara por la cual se obtiene una nueva imagen, para finalizar la rutina y guardar la imagen se espera una intervención por el usuario por medio del teclado. El diagrama de flujo de la rutina "Nueva Imagen"se muestra en la Figura 20, como primer paso comprueba si se creará una nueva imagen, si es así se activará una cámara digital y el algoritmo estará en un ciclo esperando una entrada del teclado, la letra 'q', al guardar la imagen se crea una archivo en la memoria micro del Raspberry Pi 2 modelo B.

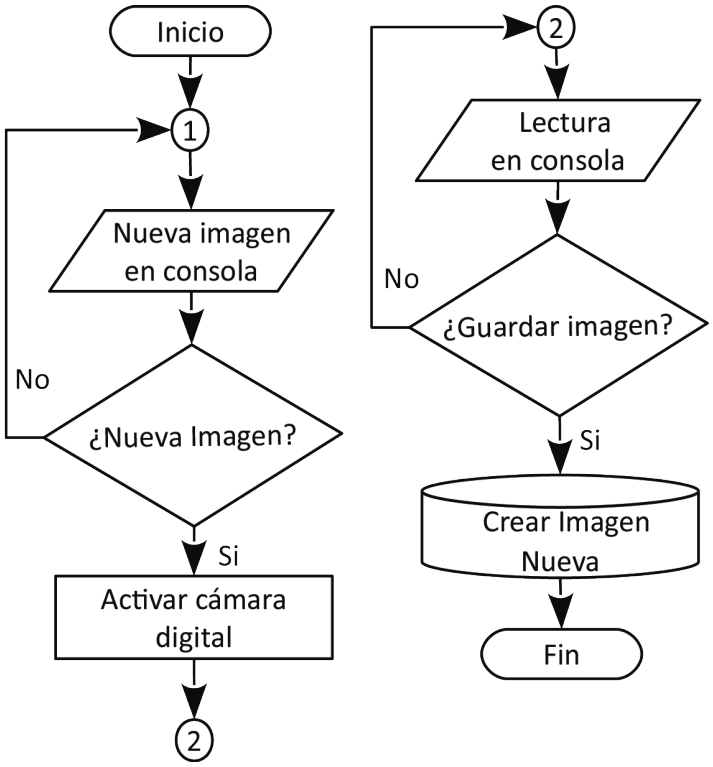


Figura 20: Diagrama de flujo de la sub-rutina "Nueva Imagen".

La sub-rutina de “Encriptar”se muestra en la Figura 21, su actividad inicia si y solo sí el usuario es validado por la rutina de Lectura de Chip de Reconocimiento de Voz, ver Figura 22, el proceso inicia configurando la UART2, pasando después a un ciclo esperando la respuesta del FPGA, es un número hexadecimal de 8 bits el hx66, al tener el número correcto se inicia la carga de la imagen ya sea una plantilla o un criptograma e inicia variables i y j, en este instante se prepara la imagen a enviar, se recibe pixel por pixel y se envia un pixel por el puerto UART2, el FPGA recibe el dato y lo procesa, ver Figura 24, regresa el dato encriptado o desencriptado segun sea el caso, lee el dato de regreso y lo almacena en la memoria, una vez que la cantidad total de pixeles es procesada, se integran en una imagen encriptada o desencriptada. La sub-rutina “Desencriptar”se describe en la secció III.5.

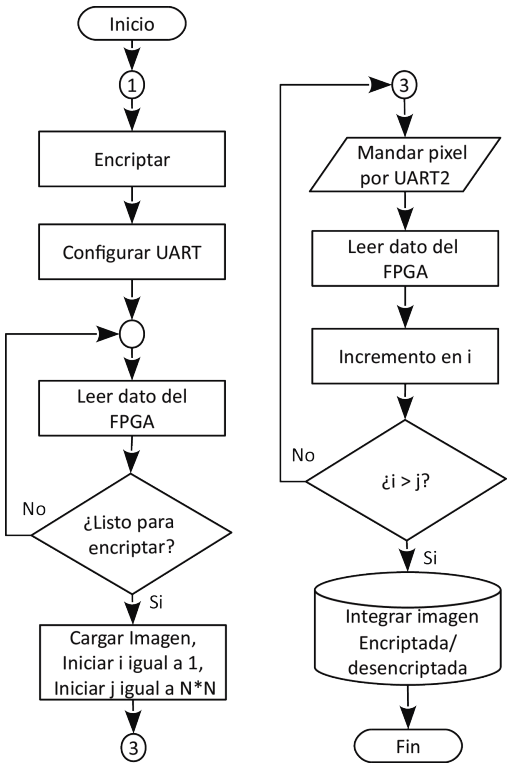


Figura 21: Diagrama de flujo de la sub-rutina “Encriptar”.

III.3 Entidad de control

La entidad de control es el algoritmo principal del sistema embebido, está constituido por 3 sub-entidades programadas en lenguaje VHDL: configuración del Chip de Reconocimiento de Voz, Lectura del Chip de Reconocimiento de Voz y Cifrado XOR.

III.3.1 Sub-entidad de autorización de acceso

El primer paso en el algoritmo es configurar el chip SPCE061A. Se logra por medio del FPGA utilizando el puerto RS-232 y el protocolo UART.

El chip SPCE061A tiene un algoritmo previamente cargado que consta de dos modos de comunicación, modo corto y modo extendido, también tiene 3 bloques de 5 campos de voz (elechouse, 2016).

la Figura 22 ilustra el diagrama de flujo de la sub-entidad que configuró al CRV, asegura la habilitación del chip SPCE061A en modo texto corto y carga el banco de voz 1. El proceso inicia al configurar y habilitar la comunicación por el puerto serial UART1, comprueba la conexión física entre el FPGA y el CRV, inicia la configuración modo corto y comprueba si ha sido correcto, con el objetivo de utilizar comandos cortos como respuesta del CRV, inicia la carga del banco de voz y por último comprueba si la carga ha sido la correcta. La sub-entidad monitorea la respuesta del chip SPCE061A el cual valida la voz del usuario por medio de la entidad UART1 sí y solo sí el algoritmo configuración del CRV concluye de forma positiva. El reconocimiento de voz se logra utilizando distintos algoritmos los cuales son propiedad de la firmware pero también permite implementar externos (Liu y Chen, 2014). La descripción del procedimiento empieza al terminar de comprobar si el banco de voz es correcto, el primer paso es recibir datos por el puerto UART1 donde está conectado el CRV y el resultado espera

un código con formato hexadecimal de dos dígitos (hx13), si es verdadera el paso a continuar es la configuración del UART2, el cual está conectado físicamente al Raspberry Pi 2 modelo B por medio de otro puerto serial, para analizar se envia el código listo a cifrar el cual es (hx66).

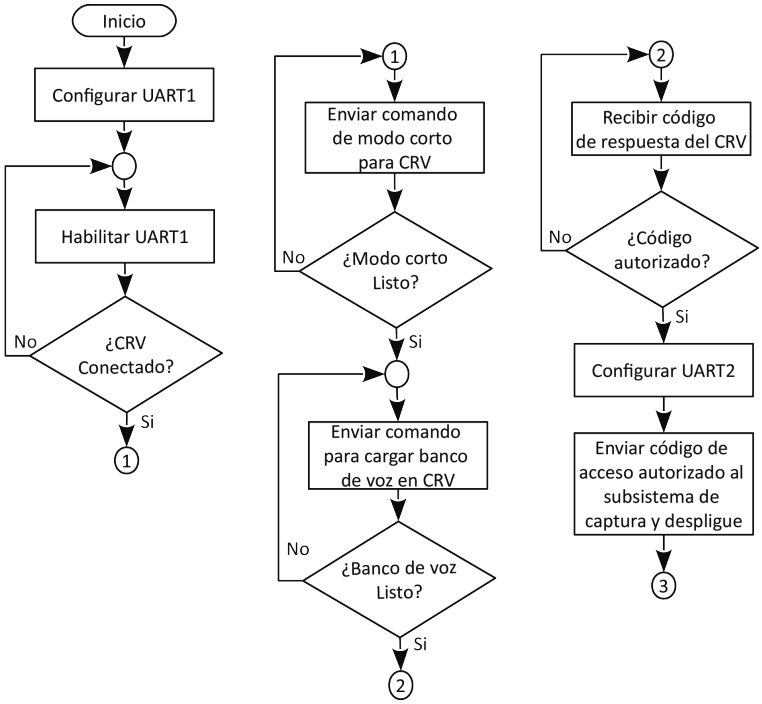


Figura 22: Diagrama de flujo de la configuración de la sub-entidad autorización de acceso.

En la parte superior de la figura 23 se ilustra la gráfica de la señal de audio utilizada en los experimentos finales de la propuesta y en la parte inferior se visualiza el espectro en frecuencias de la señal de audio. La clave de voz utilizada para el experimento es "arroyo". El algoritmo de reconocimiento de voz en el dispositivo es independiente del usuario y tiene falso positivo por lo tanto se debe selecccionar un código de acceso particular, no se realizó un estudio del falso positivo por que el enfoque del trabajo de tesis tiene otros objetivos. En la sección D se presentan otros posibles codigos de acceso.

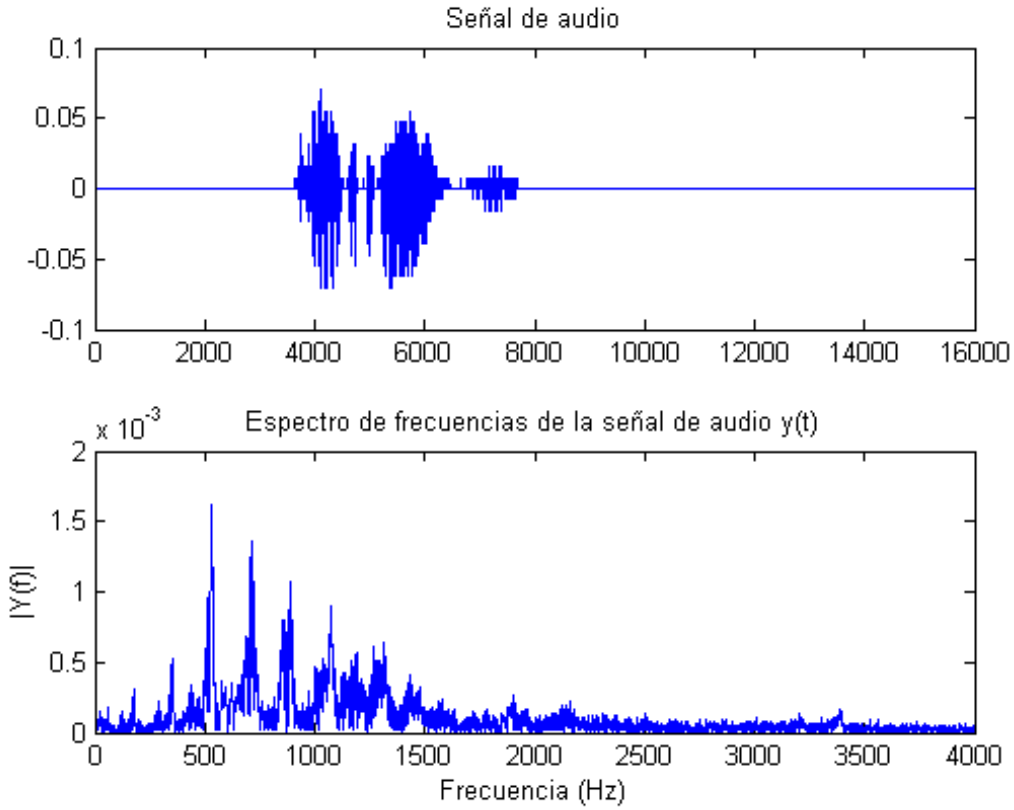


Figura 23: Espectro en frecuencia de la señal de voz de audio del código de acceso autorizado.

III.3.2 Sub-entidad de encriptado XOR

La sub-entidad encriptado XOR es de importancia significativa, se ilustra en la Figura 24. Tiene diferentes objetivos como entablar la comunicación con el subsistema de captura y despliegue por medio de la entidad UART2, adquirir los píxeles de 8 bits de la imagen a encriptar o desencriptar, generar la señal de habilitación de la entidad CPRGB, adquirir la secuencia binaria pseudoaleatoria caótica de 8 bits, encriptar o desencriptar los píxeles por medio de la operación XOR, y por último mandar el resultado al subsistema de captura y despliegue. La rutina exhibe su dinámica en siguientes pasos: La sub-entidad exhibe su dinámica comprobando una bit de inicio, sí y solo sí el código autorizado ha sido el correcto, indicando que existe un dato en el puerto UART2, a continuación se recibe el pixel de la imagen, sea un pixel de una imagen o criptograma, inmediatamente se habilita el CPRGB generando un sólo dato pseudoaleatorio caótico de 8 bits, recibe el dato generado y posteriormente deshabilitando el CPRGB, teniendo los 2 datos se utiliza la operación binaria XOR entre los 2 datos y se obtiene uno cifrado o descifrado, después el resultado es enviado por el puerto UART2 y se continua el proceso dependiendo de la cantidad de datos de la imagen a cifrar o descifrar.

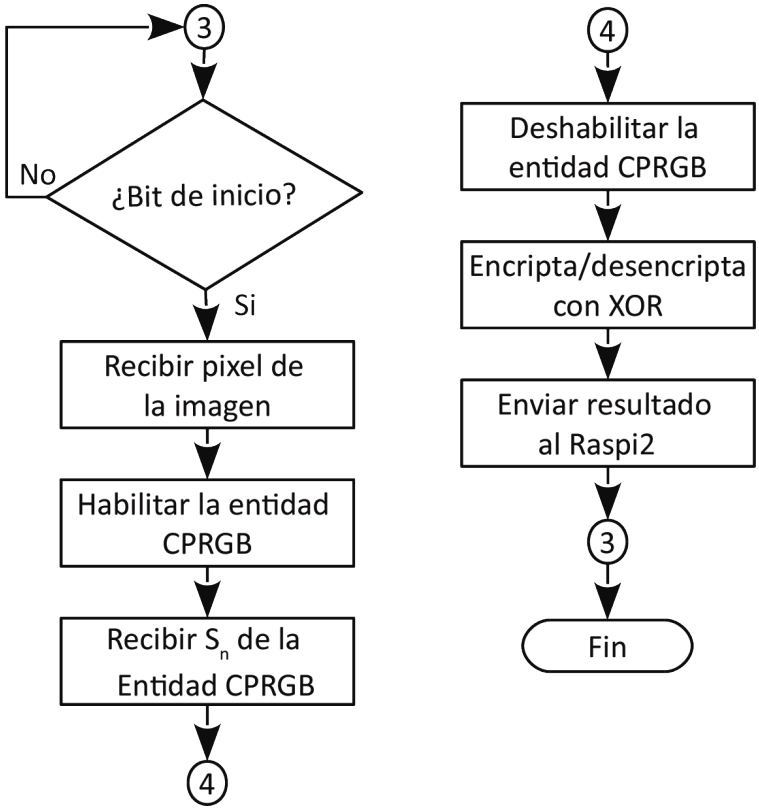


Figura 24: Diagrama de flujo de la sub-entidad de encriptado XOR.

III.4 Algoritmo generador binario pseudoaleatorio caótico implementado en FPGA

Se propone un CPRGB implementado en FPGA utilizando un sistema caótico más una metodología de operación módulo M, específicamente la operación *mod* 255. En las dos sub-secciones siguientes se describe su metodología.

Los sistemas caóticos discretos que se han implementado en "SysGen"para el trabajo de tesis son Rössler, Hénon, Tinkerbell, Karplan-Yorke y por último el Logístic 2D, los 4 últimos se presentan en el apéndice B.

III.4.1 Mapeo hipercaótico de Rössler implementado en FPGA

Tomando la ecuación 18 del sistema caótico Rössler se diseña e implementa en Simulink utilizando "SysGen" y el resultado se visualiza en la Figura 25, se utilizan los parámetros de (Rossler, 1979).

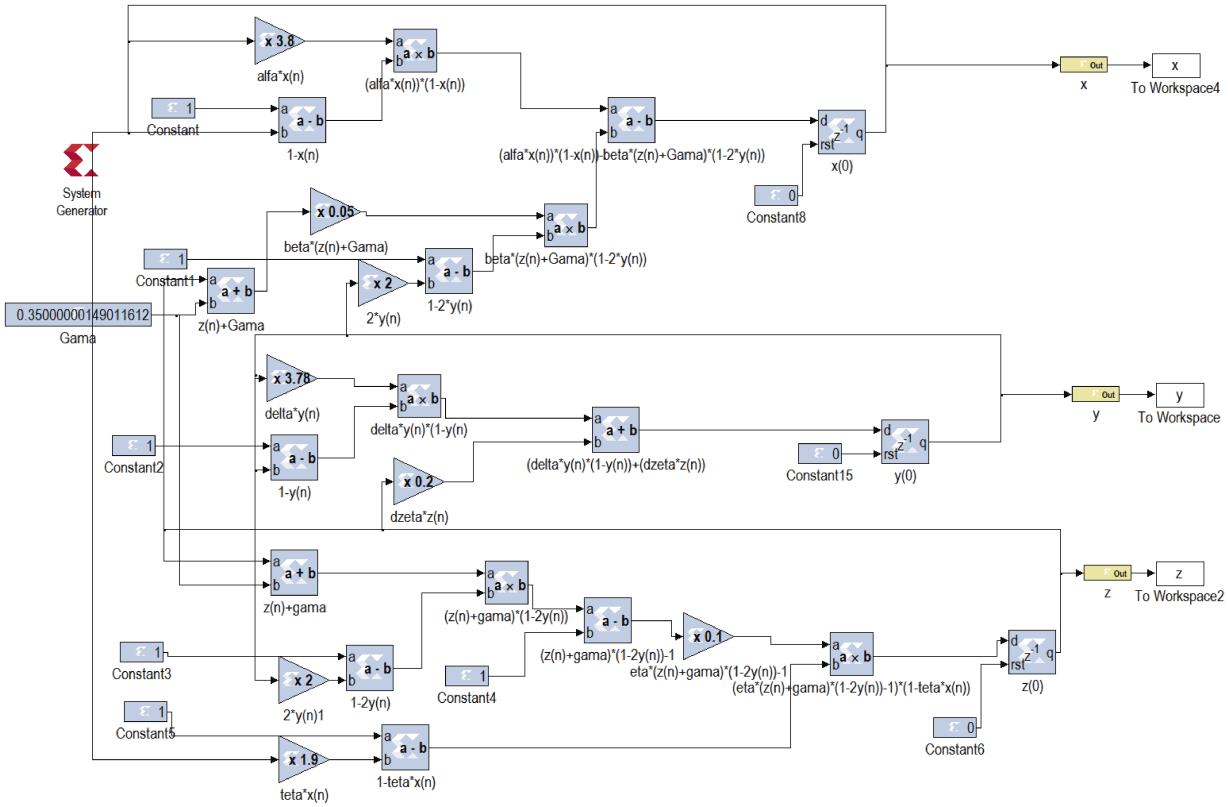
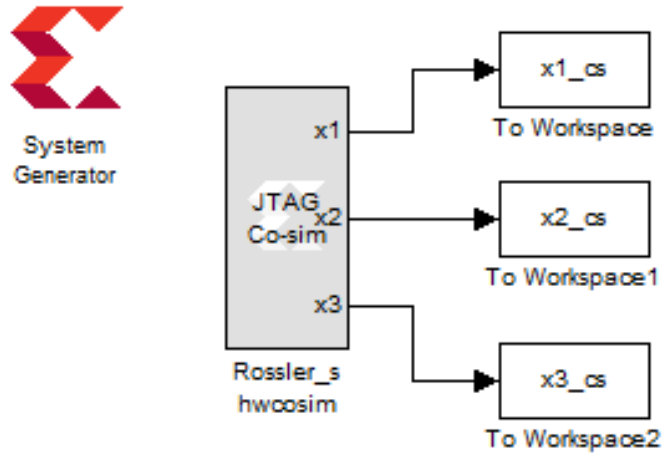


Figura 25: Esquema del modelo basado en el sistema hipercaótico Rössler.

Se implementó una co-simulación, ver Figura 26b, por medio del puerto Joint Test Action Group (JTAG). Una co-simulación es una programación del FPGA por medio de Matlab usando Simulink y la SysGen, el modelo se ilustra en la Figura 26a, se configuró el sistema Rössler ilustrado en la Figura 25, la cual tiene la capacidad de adquirir los datos del FPGA por medio del JTAG. El mismo proceso se realizó para los sistemas caóticos Hénon, Tinkerbell, Karplan-Yorke y Logístic 2D ilustrados en la apéndice B.

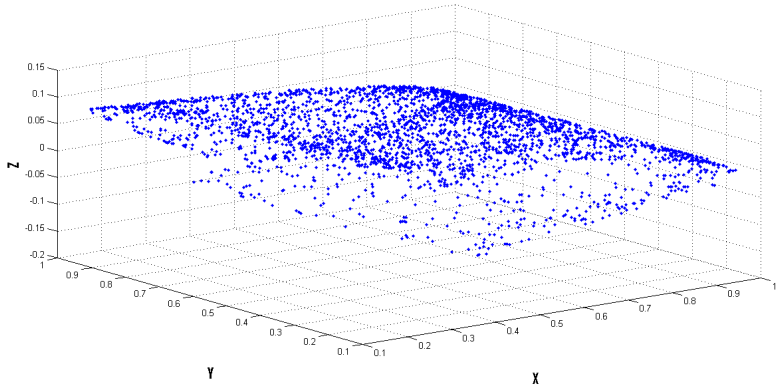


(a) Modelo de la co-simulación en Simulink.

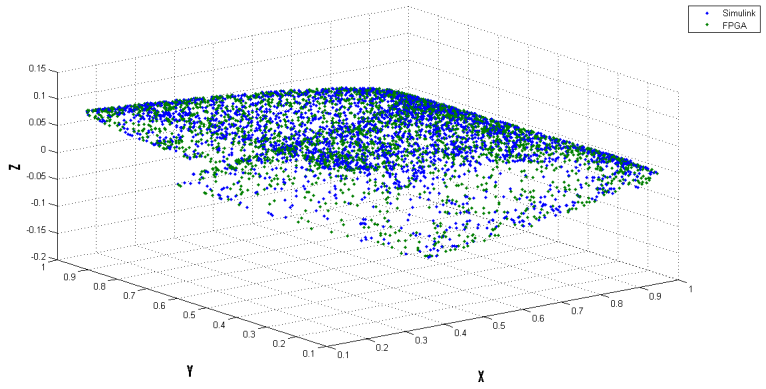
(b) Conexión entre la PC y la FPGA por medio del puerto USB, usando el protocolo JTAG.

Figura 26: Implementación de la co-simulación del sistema caótico Rössler.

El mapeo del sistema Rössler implementado en FPGA se ilustra en la Figura 27a el cual mantiene su forma característica, comprobando que la implementación ha sido satisfactoria usando una mantissa de $32Q30$, 1 bit de signo, 1 bit de tipo entero y 30 bits de la parte fracionaria. La Figura 27b muestra una comparación entre dos mapeos del sistema Rössler, en azul es el programado en Matlab y en verde el implementado en FPGA, se observa una diferencia entre los puntos que conforman el mapeo y se debe a la diferencia de la mantisa, pero los dos mantienen el mapeo.



(a) Mapeo del sistema implementado en FPGA.



(b) Comparación entre los dos mapeos.

Figura 27: Mapeo del sistema hipercaótico Rössler obtenido por medio de la co-simulación.

III.4.2 Generador Binario Pseudoaleatorio Implementado en FPGA

La secuencia pseudoaleatoria es generada por medio de la operación *mod* 255, se obtiene a partir de modelar la operación con la plataforma "SysGen"de la familia Xilinx en Simulink, la Figura 28 ilustra el modelo. La ecuación (19) es la representación matemática del CPRGB.

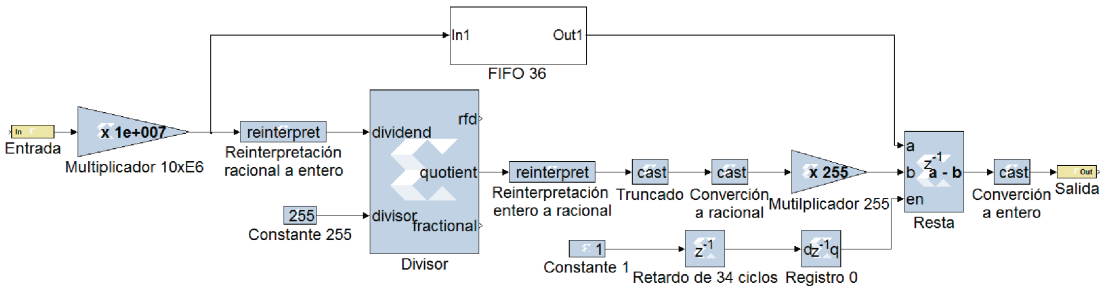


Figura 28: Diagrama de bloques de la operación *mod* 255 implementada en Simulink con bloques de "SysGen".

La descripción de los bloques, ver Figura 28, inicia con la multiplicación entre la entrada que es un dato caótico de tipo punto fijo de mantisa $32Q23$ con un bit de signo, 8 bits de entero y 23 fraccionarios, al realizar la multiplicación desplazamos el punto decimal 6 espacios manteniendo el atractor caótico pero cambiando la mantisa del resultado a $32Q8$. La ecuación (20) representa matemáticamente la función Mod y se observa la importancia de la división, el bloque Divisor de xilinx, el cual se ilustra en la Figura 28, solo acepta números enteros con signo, por lo tanto la reinterpretación es necesaria, pasando el resultado de la multiplicación a un número de $32Q0$, el resultado del divisor es la salida "quotient", es de tipo entero con signo de $32Q0$ y tiene un retraso de 34 ciclos máquina, se necesita una reinterpretación a $32Q8$, la ecuación (20) ilustra

una operación "floor"y es proporcional a un truncado pasando de un número con signo de $32Q8$ a uno de $24Q0$ eliminando los 8 bits menos significativos, los cuales representan la parte fraccional del número, para mantener un número de $32Q8$ se deben agregar los 8 bits restantes y convertir de un número entero con signo de $24Q0$ a uno racional tipo entero con signo de $32Q8$, la ecuación (20) muestra una multiplicación despues del "floor ", multiplicamos el resultado de la conversión a racional por la constante 255 y el resultado está listo en 35 ciclos máquina, para finalizar el proceso de la función Mod 255 se debe restar la salida del multiplicador 255 a la salida del multiplicador $10xE6$, la resta debe tener retraso de 36 ciclos, el bloque de resta de Xilinx tiene un puerto habilitador "en"el cual se activa después de 35, el bloque llamado "FIFO 36"tiene 35 bloques "Registros 0"en serie, al pasar 36 ciclos máquina la salida del bloque es el primer dato ingresado al divisor con $32Q8$ de mantisa. El bloque "Resta"tiene una salida de $32Q23$, el resultado de la función Mod es resto de una división, realmente es un número decimal alojado en los 8 bits menos significativos y para finalizar se debe cambiar de mantisa pasando de una $32Q23$ a uno tipo unsigned de 8 bits $8Q0$, el cual representa el resultado de un cálculo del CPRGB.

$$s_n = (C \cdot x_n) mod \quad 255 \quad \in \quad \mathbb{Z}\{0 - 255\} \tag{19}$$

donde $C = 10x10^6$ y x_n es el estado caótico seleccionado para generar la secuencia pseudoaleatoria. La ecuación siguiente define matemáticamente la finción $mod(x, y)$ si $y \ = 0$.

$$\begin{aligned} &mod(x, y) = x - ny \\ &n = floor(x/y) \end{aligned} \tag{20}$$

III.5 Algoritmo embebido propuesto para el descifrado

El algoritmo embebido de descifrado es similar que el algoritmo de cifrado. La diferecia radica en la imagen a desencriptar, ya que es un criptograma, en el subsistema de captura y despliegue se debe seleccionar el boton Desencriptar, ilustrado en la Figura **??**, el cual tiene el algoritmo representado en la Figura 21. Al final del proceso del algoritmo se forma en la memoria micro SD del Raspberry Pi 2 una imagen descifrada con el nombre de "imagenrecuperada".

III.6 Conclusión

En este capítulo se propuso un sistema de encriptado embebido utilizando 3 diferentes tecnologías un CRV, un Raspberry Pi 2 y un FPGA. También se propuso un algoritmo CPRGB a partir de la función Mod 255 con un mapeo caótico. Se propuso una interfaz gráfica implementada en una Raspberry Pi 2. Se propuso el CRV como llave de inicio del proceso de encriptado. Se presentó el desarrollo de los algoritmos embebidos en la FPGA y en Raspberry Pi 2 modelo B para el manejo de la información a encriptar.

Capítulo IV

Resultados

IV.1 Introducción

En este capítulo se presentan los resultados de los recursos de hardware necesarios para la implementación en FPGA de los diferentes mapeos caóticos, asi como el análisis de seguridad realizado al algoritmo de encriptado propuesto. Se presentan resultados de análisis estadísticos, análisis de sensibilidad de clave, entropía de la información, ataques diferenciales, correlación de pixeles adyacentes y pruebas FIPS 140-2 a las secuencoas binarias genereradas con el CPRGB implementadas en FPGA.

IV.2 Recursos de la Implementación

La viabilidad de la implementación del sistema caótico en FPGA depende exclusivamente de la cantidad de multiplicadores que tiene. Las dos características fundamentales de todos los sistemas caóticos es su dependencia a las condiciones iniciales y parámetros, al utilizar hardware para su desarrollo e implementación de sistemas caóticos de sistemas caťicos en hardware depende de la cantidad de mantisa utilizada en su diseño, un error en el cálculo de mantisa probocaría que el mapeo salga de su regimen caótico.

Las Tablas VI y VII muestran la cantidad de recursos necesarios para implementar el mapeo caótico en FPGA Spartan 3E-1600.

Recursos	**Hénon**	**Rössler**	**Disponible**
Slide Flip Flops	48	72	29,504
4 inputs LUTs	430	1447	29,504
Slices	241	795	14,752
IOBs	66	98	250
BUFGMUXs	1	1	24
MULT18X18SIOs	4	20	36

Tabla VI: Recursos necesarios para implementar los mapeos caótico de Hénon y Rössler en FPGA Spartan 3E-1600.

La Tabla VI ilustra los recursos utilizados por el sistema Hénon y Rössler para generar caos, en la columna de la derecha se muestran los recursos totales de la tarjeta los cuales cubren con las necesidades del experimento.

Recursos	**Tinkerbell**	**Logístic 2D**	**Karplan-Yorke**	**Disponible**
Slide Flip Flops	48	48	64	29,504
4 inputs LUTs	698	948	427	29,504
Slices	365	513	233	14,752
IOBs	66	66	82	250
BUFGMUXs	1	1	1	24
MULT18X18SIOs	12	20	4	36

Tabla VII: Recursos necesarios para implementar los mapeo caótico Tinkerbell, Logític 2D y Karplan-Yorke en FPGA Spartan 3E-1600.

La Tabla VII muestra el total de recursos necesarios para implementar los mapeos caóticos Tinkerbell, Logístic 2D y Karplan-Yorke en la FPGA Spartan 3E-1600.

En la Tabla VIII se muestran los recursos necesarios para la operación *mod* 255, que en conjunto con el sistema caótico se implementan en el FPGA para integrar el CPRGB.

Recursos	*mod* 255
Slide Flip Flops	1,762
4 inputs LUTs	657
Slices	670
IOBs	41
BUFGMUXs	1
MULT18X18SIOs	0

Tabla VIII: Recursos necesarios para la implementación operación *mod* 255 en FPGA Spartan 3E-1600, ver Figura 28.

Por lo tanto, se puede observar que la targeta Spartan 3E-1600 contiene los recursos necearios para implementar el algoritmo de encriptado caótico propuesto.

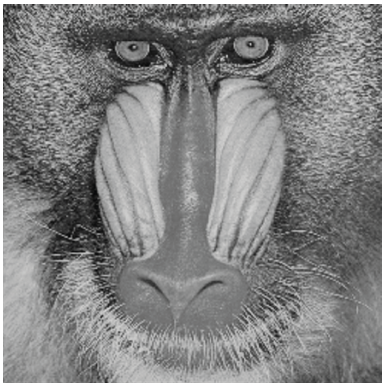
IV.3 Criptogramas

En esta sección se presentan los criptogramas de las imágenes Lena, Cameraman, Mandril y Lena RGB, obtenidos mediante los distintos mapeos caótico mencionados previamente y utilizando el algoritmo de encriptado propuesto (ver Figura 17). Adicionalmente se muestran los resultados del análisis de espacio de claves, histogramas, entropía de la información, ataques diferenciales, correlación de pixeles adyacentes y calidad del encriptado.

(a) Imagen Lena $255x255$.

(b) Imagen Cameraman $512x512$.

(c) Imagen Lena RGB $512x512$.



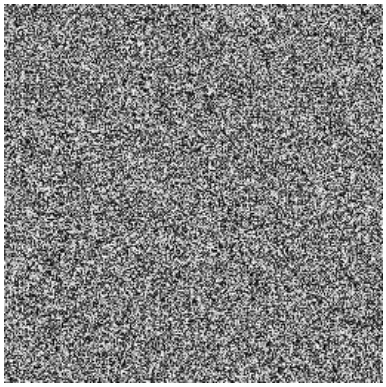
(d) Imagen Mandril $512x512$.

Figura 29: Imagen original: Fig. 29a Lena $255x255$, Fig. 29b Cameraman $512x512$, Fig. 29c Lena RGB $512x512$ y Fig. 29d Mandril $512x512$.

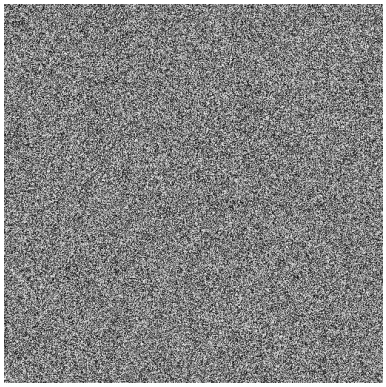
Para este análisis de seguridad se propuso utilizar estas imágenes (Lena, Cameraman y Mandril), debido a que son las más referenciadas en la literatura relacionada.

IV.3.1 Criptograma usando el mapeo de Rössler

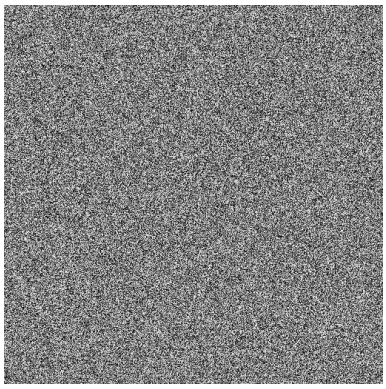
La Figura 30 presenta 3 criptogramas a partir del CPRBG con el mapeo Rössler. Las 3 Figuras ilustran el encriptado caótico de las imágenes digitales originales. Se puede observas en las Figuras 30a, 30b y 30c que son completamente inteligibles por cualquier persona o intruso.



(a) Criptograma de Lena $255x255$.



(b) Criptograma de Cameraman $512x512$.

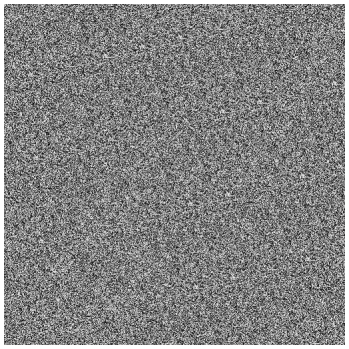


(c) Criptograma de Mandril $512x512$.

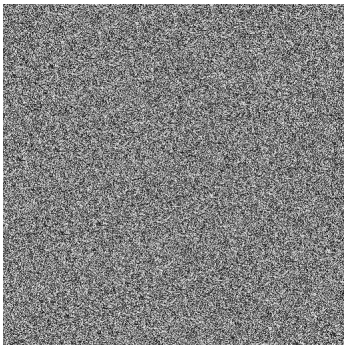
Figura 30: Criptogramas a partir del CPRBG con el mapeo de Rössler: Fig. 29a Lena $255x255$, Fig. 29b Cameraman $512x512$ y Fig. 29d Mandril $512x512$.

IV.3.2 Criptograma de la Imagen Lena RGB $512x512$ usando el mapeo de Rössler

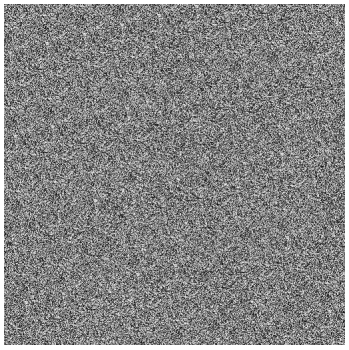
La Figura 31 presenta 4 criptogramas a partir del CPRBG con el mapeo de Rössler encriptando la imagen de Lena RGB $512x512$ (ver Figura 29c). Es bien conocido que una imagen RGB está conformada por 3 matrices MxN píxeles, una matriz representa el color rojo, una para el verde y otra para el azul.



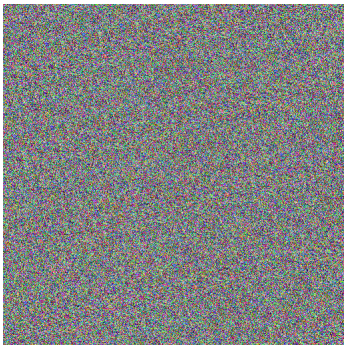
(a) Criptograma de la componente roja de Lena RGB $512x512$.



(b) Criptograma de la componente verde de Lena RGB $512x512$.



(c) Criptograma de la componente azul de Lena RGB $512x512$.



(d) Criptograma de Lena RGB $512x512$.

Figura 31: Criptogramas obtenidos a partir del CPRBG con mapeo de Rössler usando la imagen de Lena RGB $512x512$.

Las 4 Figuras ilustran el resultado del encriptado caótico. Las Figuras 31a, 31b y 31c muestran, respectivamente, el criptograma de la matriz para el color rojo, verde y azul. Cada una está constituida por $512x512$ píxeles. La Figura 31d es un criptograma RGB de $512x512$ píxeles, está constituido por los criptogramas 31a, 31b y 31c.

IV.4 Análisis Estadísticos

A partir de todos los criptogramas obtenidos en la sección IV.3 se realizan los análisis estadísticos descritos en la sección II.2.

IV.4.1 Análisis de Espacio de Claves

La Tabla IX ilustra la mantisa, el total de parámetros y condiciones iniciales, el espacio de claves de cada mapeo caótico.

La mantisa utilizada por cada mapeo es la necesaria para generar los estados caóticos emplenado el hardware de la FPGA.

Sistema/Mantisa	**Parámetros y condiciones iniciales**	**Espacio de claves**
Hénon/$24Q20$	4	2^{159}
Rössler/$32Q30$	10	2^{497}
Tinkerbell/$24Q20$	6	2^{238}
Logistic/$24Q20$	6	2^{238}
Karplan-Yorke/$32Q30$	3	2^{149}

Tabla IX: Análisis de espacio de claves.

El espacio de clave depende de las condiciones iniciales y parámetros de los sistemas caóticos, que a su vez dependen de la mantisa utilizada (ver sección II.2.1).

Los 5 sistemas implementados en el algoritmo de encriptado propuesto cumplen con los criterios de Shannon y Kerckhoffs.

IV.4.2 Análisis de Sensibilidad de claves

Tomando la imagen encriptada de Lena (ver 31a), y utilizando el algoritmo con la clave de encriptado correspondiente, se obtiene a la salida la imagen recuperada igual a la original (ver Figura 29).

Sin embargo, al efectuar un mínimo cambio en el valor de alguna de las condiciones iniciales, manteniendo sin variación el resto de las condiciones iniciales y parámetros de acceso, se obtiene la imagen mostrada en la Figura 53a del Apéndice C 5.1, se puedo observar que la imagen original no fue recuperada, por lo que el sistema es muy sensible a pequeñas variaciones en alguna de sus condiciones iniciales o parámetros.

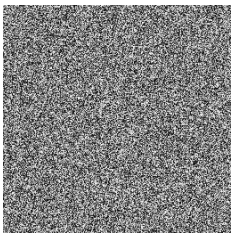
En la Tabla X se ilustra el cambio mínimo por mapeo, en este caso, el cambio mínimo por mapeo es en la condición inicial x_0 y utilizando aritmética con presición de punto fijo.

Mapeo caótico	**Condición inicial teórica X_0**	**Experimental**	**Cambio mínimo**
Hénon	$x_0 = 0.1$	0.100000381469727	0.099999427795410
Rössler	$x_0 = 0.1$	0.100000000000364	0.099999046326047
Tinkerbell	$x_0 = 0.1$	0.100000381469727	0.099999427795410
Logístic 2D	$x_0 = 0.1$	0.100000381469727	0.099999427795410
Karplan-Yorke	$x_0 = 0.025$	0.025000000372529	0.024999999441206

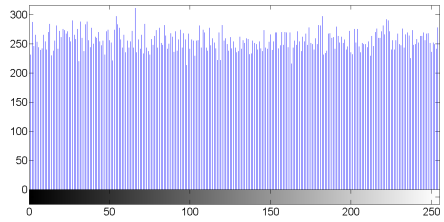
Tabla X: Sensibilidad de los mapeo: Hénon, Rössler, Tinkerbell, Logístic 2D y Karplan-Yorke.

Existe una diferencia mínima entre la condición inicial y la experimental, está diferencia disminuye si aumenta la mantisa utilizada del mapeo caótico, pero el costo recae en la cantidad de LUTs y multiplicadores que requiere la arquitectura de la FPGA utilizada. El cambio mínimo mostrado en la Tabla X se debe a las mantisas configuradas para cada mapeo caótico que previamente fueron mostradas en la Tabla IX.

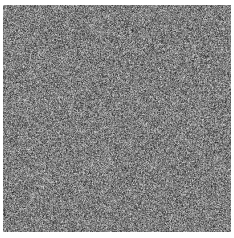
La Figura 32 muestra el resultado gráfico del análisis a la sensibilidad al mínimo cambio en la clave, las Figuras adquiridas 32a, 32c y 32e son ilegibles, tanto así que las imágenes 32b, 32d y 32f muestran, respectivamente, el histograma de la imagen recuperada, se observa que tiene una distribución uniforme, por lo tanto los criptogramas obtenidos no contienen información o vestigios de la imagen original, lo cual significa que el algoritmo propuesto es robusto contra ataques estadísticos.



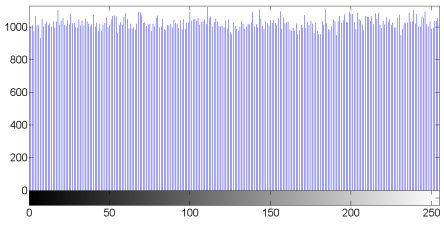
(a) Imagen no recuperada de Lena 29a.



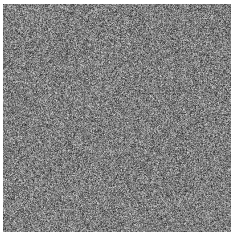
(b) Histograma del criptograma de Lena no recuperada 32a.



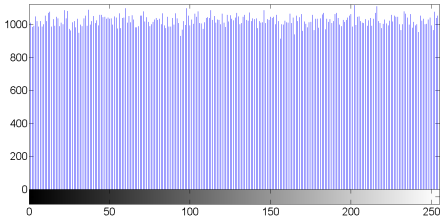
(c) Imagen no recuperada de Cameraman 29b.



(d) Histograma del criptograma de Cameraman no recuperada 32c.



(e) Imagen no recuperada de Mandril 29c.

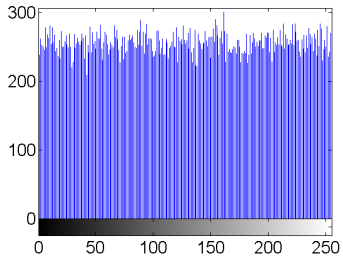


(f) Histograma del criptograma de Mandril no recuperada 32e.

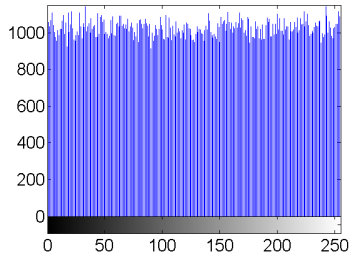
Figura 32: Análisis de sensibilidad de clave utilizando el mapeo Rössler.

IV.4.3 Histogramas de los criptogramas obtenidos con el mapeo Rössler

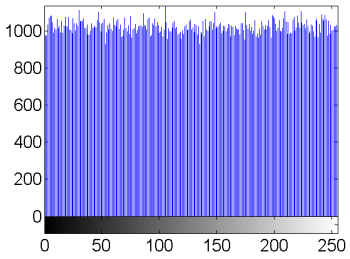
La Figura 33 ilustra los histogramas de los criptogramas obtenidos a partir del CPRGB con el mapeo de Rössler, utilizando las Figuras 30a, 30b y 30c.



(a) Histograma del criptograma de Lena $255x255$ la de Figura 30a.



(b) Histograma del criptograma Cameraman $512x512$ de la Figura 30b.



(c) Histograma del criptograma Mandril $512x512$ de la Figura 30c.

Figura 33: Histogramas de los criptogramas obtenidos a partir del CPRBG con mapeo Rössler de las Figuras 30a, 30b y 30c.

Se puede observar que los histogramas de los criptogramas mostrados en la Figura 33 muestran una distribución uniforme, por lo tanto no contienen información de las imágenes originales, lo que comprueba que el algoritmo propuesto e implementado en FPGA es robusto contra ataques estadísticos.

IV.4.4 Resultado de la entropía de la información

La Tabla XI muestra los resultados del análisis de entropía de la infromación obtenidos del criptograma de Lena con CPRBG empleando los distintos mapeos. Se puede observar que en todos los casos, el resultado de la entropía es cercano al valor ideal, es decir, a 8 Bits, tal como se menciona en la sección II.2.4.

Mapeo Caótico	**Entropía**	**Criptograma**
Hénon	7.99728224	Figura 49a
Rössler	7.99756955	Figura 30a
Tinkerbell	7.99709578	Figura 50a
Karplan Yorke	7.99696075	Figura 51a
Logístic 2D	7.99739303	Figura 52a

Tabla XI: Entropía de la información de la imagen de Lena encriptada en la FPGA con los distintos mapeos.

La Tabla XII muestra los resultados del análisis de entropía de la infromación obtenidos del criptograma de Cameraman con CPRBG empleando los distintos mapeos. Se puede observar que en todos los casos, el resultado de la entropía es cercano al valor ideal, es decir, a 8 Bits.

Mapeo Caótico	**Entropía**	**Criptograma**
Hénon	7.99927416	Figura 49b
Rössler	7.99870848	Figura 30b
Tinkerbell	7.99929761	Figura 50b
Karplan Yorke	7.9989076	Figura 51b
Logístic 2D	7.99928301	Figura 52b

Tabla XII: Entropía de la información de la imagen de Cameraman encriptada en la FPGA con los distintos mapeos.

La Tabla XIII muestra los resultados del análisis de entropía de la infromación obtenidos del criptograma del Mandril con CPRBG empleando los distintos mapeos. Se puede observar que en todos los casos, el resultado de la entropía es cercano al valor ideal, es decir, a 8 Bits.

Mapeo Caótico	**Entropía**	**Criptograma**
Hénon	7.999166835	Figura 49c
Rössler	7.999185122	Figura 30c
Tinkerbell	7.999297506	Figura 50c
Karplan Yorke	7.999239746	Figura 51c
Logístic 2D	7.999396502	Figura 52c

Tabla XIII: Entropía de la información de la imagen de Mandril encriptada en la FPGA con los distintos mapeos.

Tal como se observó en las Tablas XI, XII y XIII, se comprueba experimentalmente la confiabilidad y robustez del algoritmo propuesto e implementado en FPGA.

La Tabla XIV muestra el análisis de la Entropía de información aplicado a los criptogramas de las Figuras 31a, 31b y 31c que conforman las componenetes RGB de la imagen Lena RGB $512x512$ mostrada en la Figura 31d.

Lena RGB $512x512$	
Color	**Entropía**
Rojo	7.99915746
Verde	7.99920835
Azul	7.999218050

Tabla XIV: Entropía del criptograma Lena RGB $5212x512$ de la Figura 31d

De acuerdo a la Tabla XIV, se observa que el resultado de la entropía de la información es sercano a 8 bits, lo cual comprueba nuevamente la confiabilidad y robustes del algoritmo de encriptado implementado en FPGA.

IV.4.5 Resultados de ataques diferenciales

En la Tabla XV se muestran los porcentajes de los ataques diferenciales $NPCR$ y el $UACI$ obtenidos con CPRGB usando los distintos mapeos caóticos y el criptograma de la imagen de Lena mostrada en la Figura 29a. Se puede observar que los resultrados mostrados son cercanos a su ideal, $NPCR$ es 100% y el $UACI$ es 34%.

Mapeo caótico	$NPCR$ (%)	$UACI$ (%)	**Criptograma**
Hénon	99.6307373	33.5661825	Figura 49a
Rössler	99.6200562	33.4577912	Figura 30a
Tinkerbell	99.5758057	33.3647365	Figura 50a
Karplan-Yorke	99.6002197	33.5527128	Figura 51a
Logístic 2D	99.5773315	33.5271439	Figura 52a

Tabla XV: Ataques diferenciales $NPCR$ y $UACI$ de la imagen del criptograma de Lena $256x256$ obtenida con los distintos mapeos caóticos implementados en FPGA.

En la Tabla XVI se muestran los porcentajes $NPCR$ y el $UACI$ obtenidos con el CPRGB emplenado los distintos mapeos y usando el criptograma de la imagen de Cameraman $512x512$. Nuevamente se observa que los resultrados mostrados son próximos al ideal, $NPCR$ sercanos 100% y el sercano $UACI$ 34%, nuevamente se comprueba la robustez del algoritmo contra ataques diferenciales.

Mapeo caótico	$NPCR$ (%)	$UACI$ (%)	**Criptograma**
Hénon	99.60212708	33.52865032	49b
Rössler	99.61204529	33.56798209	30b
Tinkerbell	99.61242676	33.37476543	50b
Karplan-Yorke	99.61128235	33.38566649	51b
Logístic 2D	99.59602356	33.44978482	52b

Tabla XVI: Resultados de ataques diferenciales $NPCR$ y $UACI$ de la imagen del criptograma de Cameraman $512x512$ obtenida con los distintos mapeos caóticos implementados en FPGA.

En la Tabla XVII muestra los porcentajes $NPCR$ y el $UACI$ obtenidos con el CPRGB emplenado los distintos mapeos y usando el criptograma de la imagen de Mandril $512x512$. Nuevamente se observa que los resultrados mostrados son próximos al ideal, $NPCR$ sercanos 100% y el sercano $UACI$ 34%, nuevamente se comprueba la robustez del algoritmo contra ataques diferenciales.

Mapeo caótico	$NPCR$ (%)	$UACI$ (%)	**Criptograma**
Hénon	99.60212708	33.52865032	Figura 49c
Rössler	99.61204529	33.56798209	Figura 30c
Tinkerbell	99.61242676	33.37476543	Figura 50c
Karplan-Yorke	99.61128235	33.38566649	Figura 51c
Logístic 2D	99.59602356	33.44978482	Figura 52c

Tabla XVII: Ataques diferenciales $NPCR$ y $UACI$ del criptograma de Mandril.

En la Tabla XVIII se plasman los porcentajes $NPCR$ y el $UACI$ obtenidos con el CPRGB empleando el mapeo Rössler y usando el criptograma de la imagen de Lena RGB $512x512$. Nuevamente se observa que los resultrados mostrados son próximos al ideal, $NPCR$ sercanos 100% y el sercano $UACI$ 34%, nuevamente se comprueba la robustez del algoritmo contra ataques diferenciales.

Componente RGB	$NPCR$ (%)	$UACI$ (%)	**Criptograma**
Rojo	99.61738586	33.58308979	31a
Verde	99.61967468	33.57611413	31b
Azul	99.62005615	33.54349622	31c

Tabla XVIII: Ataques diferenciales $NPCR$ y $UACI$ del criptograma de Lena RGB $512x512$.

De acuerdo a las Tablas XV, XVII, XVI y XVIII, se observa que los resultado del ataque diferencial es sercano a ideal, lo cual comprueba nuevamente la confiabilidad y robustes del algoritmo de encriptado implementado en FPGA.

IV.4.6 Resultados de la correlación de pixeles adyacentes

Numéricamente evaluamos los coeficientes de correlación (r_{xy}) de los criptogramas, siguiendo el procedimiento del capitulo IV.3, según la teoría referenciada en la sección II.2.6, para píxeles horizontales, verticales y diagonales, se espera un número próximo a cero por los negativos o postivos.

La Tabla XIX muestra la correlación de píxeles adyacentes por CPRBG utilizando cada uno de los mapeos caóticos aplicado al criptograma de Lena $256x256$. Se concluye que no existe correlación de píxeles en los criptogramas.

Mapeo	**Horizontal**	**Vertical**	**Diagonal**	**Criptograma**
Hénon	$-1.1210E^{-03}$	$-5.8573E^{-02}$	$-3.2015E^{-02}$	Figura 49a
Rössler	$-9.8539E^{-02}$	$-7.4748E^{-02}$	$5.9399E^{-02}$	Figura 30a
Tinkerbell	$-1.0641E^{-01}$	$1.7812E^{-02}$	$-9.4017E^{-02}$	Figura 50a
Karplan Yorke	$2.2356E^{-02}$	$-1.1600E^{-02}$	$3.3231E^{-02}$	Figura 51a
Logístic 2D	$-5.4137E^{-02}$	$-4.3078E^{-02}$	$2.3518E^{-02}$	Figura 52a

Tabla XIX: Coeficientes de correlación del criptograma de Lena $256x256$.

La Tabla XX muestra el análisis de la correlación de píxeles adyacentes al criptograma de Camaraman $512x512$ por CPRBG utilizando cada uno de los mapeos caóticos. nuevamente se observa que no existe correlación de píxeles en los criptogramas, por lo tanto se comprueba la fortaleza del algoritmo de encriptado implementado en FPGA.

Mapeo	**Horizontal**	**Vertical**	**Diagonal**	**Criptograma**
Hénon	$-3.3808E^{-02}$	$3.5779E^{-02}$	$3.5536E^{-02}$	Figura 49b
Rössler	$-4.9263E^{-03}$	$-9.2114E^{-02}$	$-3.7723E^{-02}$	Figura 30b
Tinkerbell	$1.2448E^{-01}$	$3.2571E^{-02}$	$-3.1768E^{-02}$	Figura 50b
Karplan Yorke	$8.0906E^{-02}$	$5.2241E^{-02}$	$1.5412E^{-03}$	Figura 51b
Logístic 2D	$-9.9549E^{-02}$	$8.7098E^{-02}$	$-1.1875E^{-02}$	Figura 52b

Tabla XX: Coeficientes de correlación del criptograma de Cameraman $512x512$.

La Tabla XXI muestra el análisis de la correlación de píxeles adyacentes al criptograma del Mandril $512x512$ por CPRBG utilizando cada uno de los mapeos caóticos. por tal motivo nuevamente se observa que no existe correlación de píxeles en los criptogramas, por lo tanto se comprueba la fortaleza del algoritmo de encriptado implementado en FPGA.

Mapeo	**Horizontal**	**Vertical**	**Diagonal**	**Criptograma**
Hénon	$3.5782E^{-02}$	$-8.7234E^{-03}$	$-2.6292E^{-02}$	Figura 49c
Rössler	$-4.0111E^{-02}$	$8.275E^{-03}$	$-3.591E^{-02}$	Figura 30c
Tinkerbell	$1.1935E^{-01}$	$-6.5018E^{-02}$	$-1.5191E^{-02}$	Figura 50c
Karplan Yorke	$-5.016074E^{-02}$	$-1.1862E^{-03}$	$1.0427E^{-01}$	Figura 51c
Logístic 2D	$-7.3348E^{-02}$	$8.9931E^{-02}$	$1.3103E^{-02}$	Figura 52c

Tabla XXI: Coeficientes de correlación del criptograma de Mandril.

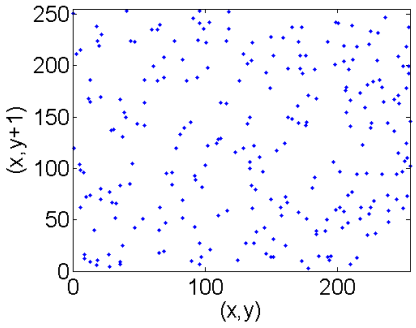
La Tabla XXII muestra la correlación de píxeles adyacentes de los criptogramas de los componentes de la imagen de lena RGB $512x512$ (ver Figuras 31a, 31b y 31c), usando CPRBG con el mapeo Rössler. Se concluye que no existe correlación de píxeles en los criptogramas.

Criptogramas de los componentes Lena RGB $512x512$			
Correlación	**Rojo**	**Verde**	**Azul**
Horizontal	$18.6062E^{-03}$	$-12.2641E^{-03}$	$22.7746E^{-03}$
Vertical	$-43.6481E^{-03}$	$-17.8192E^{-03}$	$94.725E^{-03}$
Diagonal	$44.3792E^{-03}$	$18.6089E^{-03}$	$68.8857E^{-03}$

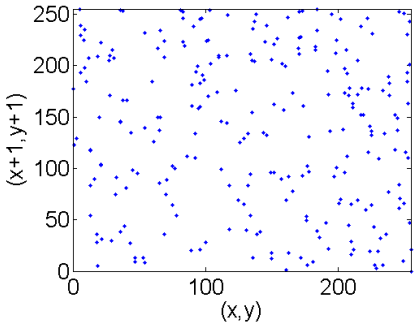
Tabla XXII: Coeficientes de correlación de los criptogramas de los componenetes de la imaggen de Lena RGB $512x512$.

IV.4.7 Gráficas de la correlación de píxeles adyacentes usando el mapeo Rössler

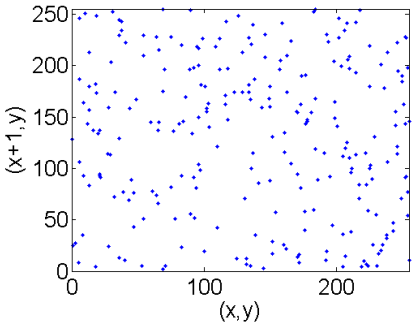
La Figura 34 ilustra gráficamente la correlación de píxeles adyacentes del criptograma 30a utilizando el CPRGB con el sistema Rössler. La Figura 34a ilustra la correlación de píxeles adyacentes de forma vertical, donde la distribución de los puntos en la gráfica son de manera erratica por todo plano, por lo tanto no existe una correlación de pẋeles adyacentes. El mismo comportamiento aparece en las Figuras 34b y 34c que son las correlaciones de píxeles adyacentes en forma diagonal y horizontal resptectivamente, como la distrución pseudo-aleatoria en la gráfica no existe correlación de pixeles adyacentes, por lo tanto muestra la robustes del algoritmo de encriptado en FPGA.



(a) Correlación vertical.



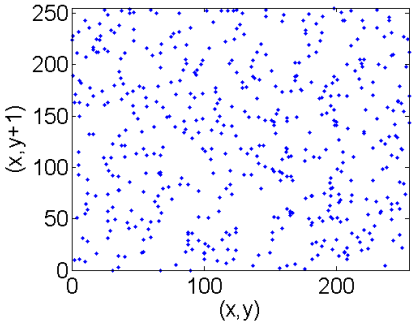
(b) Correlación diagonal.



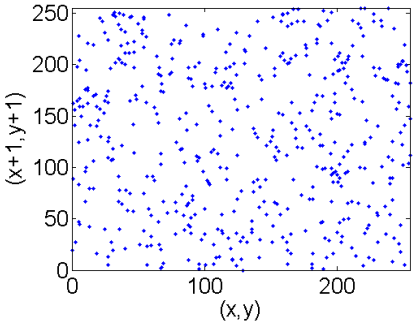
(c) Correlación horizontal.

Figura 34: Correlación de píxeles adyacentes del criptograma Lena usando el mapeo Rössler.

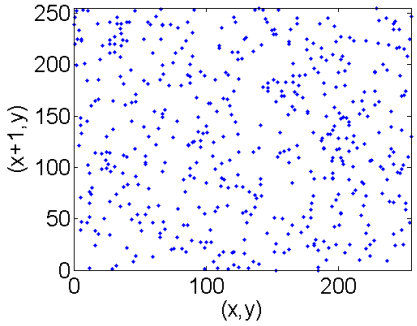
La Figura 35 muestra gráficamente la correlación de pẋeles adyacentes del criptograma 30b utilizando el CPRGB con el mapeo Rössler. La Figura 35a ilustra la correlación de píxeles adyacentes de forma vertical, donde la distribución de los puntos en la gráfica es disperza por lo tanto no existe una correlación de pẋeles adyacentes. El mismo comportamiento aparece en las Figuras 35b y 35c que son las correlaciones de pixeles adyacentes en forma diagonal y horizontal resptectivamente, como la distrución es una disperciońn pseudo-aleatoria en la gráfica no existe correlación de pixeles adyacentes, por lo tanto se comprueba que el algoritmo es robusto.



(a) Correlación vertical.



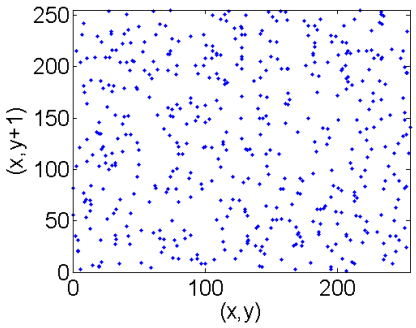
(b) Correlación diagonal.



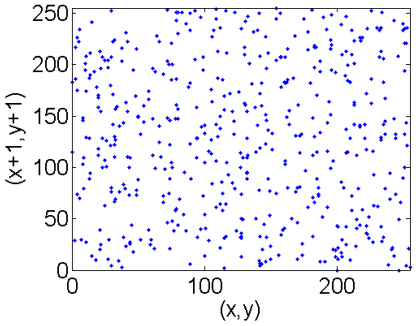
(c) Correlación horizontal.

Figura 35: Grafíca de correlación de píxeles adyacentes del criptograma Cameraman usando el mapeo Rössler.

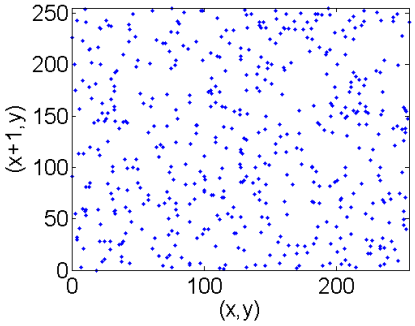
La Figura 36 ilustra gráficamente la correlación de pẋeles adyacentes del criptograma 30c utilizando el CPRGB con el mapeo Rössler. La Figura 36a ilustra la correlación de píxeles adyacentes de forma vertica, donde la distribución de los puntos en la gráfica es de forma aleatoria, por lo tanto no existe una correlación de pẋeles adyacentes. El mismo comportamiento aparece en las Figuras 36b y 36c que son las correlaciones de pixeles adyacentes en forma diagonal y horizontal resptectivamente, como la distrución es aleatoria en la gráfica no existe correlación de pixeles adyacentes, por lo tanto nuevamente se comprueba la robustez del algoritmo de encriptado implementado en FPGA.



(a) Correlación vertical.



(b) Correlación diagonal.



(c) Correlación horizonta.

Figura 36: Grafíca correlación de píxeles adyacentes del criptograma Mandril usando el mapeo Rössler.

IV.4.8 Calidad del encriptado

En la primer columna de las tablas XXIII y XXIV se muestran los resultados de la máxima D por cada sistema caótico implementado en el algoritmo y los resultados son cercanos al ideal.

Observando la correlacion CC en la segunda columna de las tablas XXIII y XXIV por cada mapeo caótico implementado en la FPGA y los resultados son próximos al ideal.

Se ilustra en la tercer columna de las tablas XXIII y XXIV los resultados de la AS por cada mapeo caótico implementado en el algoritmo y los resultados son competitivos.

Mapeo caótico	**Máxima** D	**Correlación** CC	**Irregular** AS
Hénon	65399	0.00908234	44804
Tinkerbell	65390.5	0.004472406	44826
KarplanYorke	65414.5	−0.000286007	45094
Logistic 2D	65394.5	0.002105741	45078
Rössler	65408.5	0.001003561	45000

Tabla XXIII: Calidad del encriptado obtenido a partir del criptograma de Lena $256x256$.

Mapeo caótico	**Máxima** D	**Correlación** CC	**Irregular** AS
Hénon	261606	−0.000674241	157706
Tinkerbell	261607	0.001918371	158116
KarplanYorke	261634.5	0.005342149	159376
Logistic 2D	261610	0.000764522	158266
Rössler	261591	0.001736202	157352

Tabla XXIV: Calidad del encriptado obtenido a partir del criptograma de Camaraman $512x512$.

Con los resultados mostrados en las tablas XXIII y XXIV se demuestra que el algoritmo implementado en la FPGA es de excelente calidad.

IV.4.9 Pruebas FIPS 140-2 a la secuencia binaria

Las pruebas se aplican al CPRGB con el mapeo caótico Rössler usando la función *mod* 255.

Segun la teoría ilustrada en la sección II.2.8, se seleccionaron 100 muestras con 100000 bits para aplicar las pruebas, cada muestra con diferente clave, siguiendo el protocolo se realizaron 100 veces cada una de las pruebas, los resultados se muestran en la Tabla XXV, ilustran que el CPRGB propuesto es rechazado como generador binario pseudoaleatorio ideal, es decir los resultados son competitivos.

Prueba	**Porcentaje aprobado**	**Valores requeridos**
Frequency test X_1	93%	< 3.8415
Two bits test X_2	99%	< 5.9915
Poker test X_3	95%	< 14.067
Runs test X_4	96%	< 33.9244
Autocorrelation test X_5	97%	$-1.96 < X_5 < 1.96$

Tabla XXV: Pruebas estadísticas FIPS 140-2.

IV.4.10 Conclusión

En este capítulo se presentaron los resultados del sistema embebido de encriptado caótico usando el CPRGB con los mapeos Rössler, Hénon, Tinkerbell, Logístic y Karplan-Yorke. Con respecto al análisis de seguridad de la propuesta usando diferentes sistemas caóticos, se puede observar que son competitivos y no existe diferencia entre ellos, por lo tanto la fortaleza del CPRGB radica en al función *mod* 255. Se presentó un análisis FIPS 140-2 a la secuencia de datos, los resultados son aceptables y competitivos. La diferencia significativa resultante se encuentra el espacio de clave el cual depende de los parámetros y condiciones iniciales del mapeo caótico.

Capítulo V

Conclusiones

Se propuso un sistema embebido de cifrado caótico de imágenes implementado en FPGA con una llave de acceso biométrica y se desarrollo una interfaz gráfica en Raspberry Pi 2 modelo B.

La potencialidad del sistema se integra a partir de las bondades de los elementos que lo componen:

i. El CPRGB embedido en hardware

ii. La posibilidad de integrar diversos sistemas caóticos

iii. El chip de reconocimiento de voz como llave de acceso

iv. Una interfaz gráfica implementada en Raspberry Pi 2 B

El CPRGB no es aceptado en las pruebas FIPS 149-2 como un generador ideal, es decir es competitivo ya que potencializa cualquier sistema caótico empleado en el algoritmo, esto se observa en los resultados de los diferentes ataques a la seguridad del algoritmo cambiando de mapeo caótico obteniéndose estándares de seguridad muy competitivos a nivel internacional.

Se aumenta la seguridad al tener un usuario que use su voz como clave de acceso, por lo tanto el acceso al sistema por un intruso se complica al integrar el chip de reconocimiento de voz.

La integración del Raspberry Pi 2 B y la interfaz gráfica provee una interación dinámica con otros periféricos para proporcionar privacidad y seguridad en el encriptado de información confidencial.

Como trabajo futuro se propone mejorar la interfaz gráfica para manipular las condiciones iniciales y parámetros del mapeo caótico implementado en el FPGA, esto es modificar la secuencia binaria pseudo-aleatoria generada por el CPRGB, teniendo diferentes llaves del sistema de encriptado embebido propuesto, también se propone utilizar otros datos para cifrar como son video, audio, texto, etc.

Se propone sustituir el chip de reconocimiento de voz por un algoritmo semejante e implementarlo en la FPGA.

Como los algoritmos criptográficos tienen un tiempo de vida finito, siempre será necesario proponer nuevas metodologías cada vez más complejas y seguras para proteger la información confidencial, por lo tanto, se propone implementar otras metodologías en FPGA para generar CPRGB diferentes, manteniendo la seguridad de la información.

Bibliografía

Addabbo, T., Fort, A., Rocchi, S., y Vignoli, V. (2011). Digitized chaos for pseudo-random number generation in cryptography. En *Chaos-Based Cryptography*, páginas 67–97. Springer.

Altera (2016). www.altera.com.

Alvarez, G. y Li, S. (2006). Some basic cryptographic requirements for chaos-based cryptosystems. *International Journal of Bifurcation and Chaos*, **16**(08): 2129–2151.

Azzaz, M. S., Tanougast, C., Sadoudi, S., y Dandache, A. (2011). New hardware cryptosystem based chaos for the secure real-time of embedded applications. En *Signal Processing Systems (SiPS), 2011 IEEE Workshop on*, páginas 251–254. IEEE.

Behnia, S., Akhshani, A., Ahadpour, S., Mahmodi, H., y Akhavan, A. (2007). A fast chaotic encryption scheme based on piecewise nonlinear chaotic maps. *Physics Letters A*, **366**(4): 391–396.

Behnia, S., Akhshani, A., Mahmodi, H., y Akhavan, A. (2008). A novel algorithm for image encryption based on mixture of chaotic maps. *Chaos, Solitons & Fractals*, **35**(2): 408–419.

Chen, G., Mao, Y., y Chui, C. K. (2004). A symmetric image encryption scheme based on 3D chaotic cat maps. *Chaos, Solitons & Fractals*, **21**(3): 749–761.

Cosmiac (2014). http://www.cosmiac.org.

Cristina, D. A., Radu, B., y Ciprian, R. (2012). A new pseudorandom bit generator using compounded chaotic tent maps. En *Communications (COMM), 2012 9th International Conference on*, páginas 339–342. IEEE.

Dabal, P. y Pelka, R. (2014). Fast pipelined pseudo-random number generator in programmable soC device. En *Signals and Electronic Systems (ICSES), 2014 International Conference on*, páginas 1–4. IEEE.

Dabal, P. y Pelka, R. (2015). An efficient post-processing method for pipelined pseudo-random number generator in soC-FPGA. En *Mixed Design of Integrated Circuits & Systems (MIXDES), 2015 22nd International Conference*, páginas 607–611. IEEE.

elechouse (2016). http://www.elechouse.com/elechouse/.

Elkamchouchi, H. y Makar, M. (2005). Measuring encryption quality for bitmap images encrypted with rijndael and kamkar block ciphers. En *Proceedings of the Twenty-Second National Radio Science Conference, 2005. NRSC 2005.*, páginas 277–284. IEEE.

Fang, X., Wang, Q., Guyeux, C., y Bahi, J. M. (2014). FPGA acceleration of a pseudorandom number generator based on chaotic iterations. *Journal of Information Security and Applications*, **19**(1): 78–87.

FIPS, P. (2001). 140-2. *Security Requirements for Cryptographic Modules*, **25**.

Fu, C., Lin, B.-b., Miao, Y.-s., Liu, X., y Chen, J.-j. (2011). A novel chaos-based bit-level permutation scheme for digital image encryption. *Optics communications*, **284**(23): 5415–5423.

Gao, T. y Chen, Z. (2008). A new image encryption algorithm based on hyper-chaos. *Physics Letters A*, **372**(4): 394–400.

Goldsztejn, A., Hayes, W., y Collins, P. (2011). Tinkerbell is chaotic. *SIAM Journal on Applied Dynamical Systems*, **10**(4): 1480–1501.

Grassberger, P. y Procaccia, I. (2004). Measuring the strangeness of strange attractors. En *The Theory of Chaotic Attractors*, páginas 170–189. Springer.

Halfacree, G. y Upton, E. (2012). *Raspberry Pi user guide*. John Wiley & Sons.

Hénon, M. (1976). A two-dimensional mapping with a strange attractor. *Communications in Mathematical Physics*, **50**(1): 69–77.

Inzunza, E. (2012). *Encriptado caótico en sistemas biométricos*. Tesis de doctorado, Universidad Autónoma de Baja California.

Kaplan, J. y Yorke, J. (1979). Functional differential equations and approximation of fixed points. *Lecture notes in mathematics*, **730**: 228.

Kerckhoffs, A. (1978). *La cryptographie militaire*. University Microfilms.

Kocarev, L. y Jakimoski, G. (2003). Pseudorandom bits generated by chaotic maps. *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on*, **50**(1): 123–126.

Kocarev, L. y Lian, S. (2011). *Chaos-based cryptography: theory, algorithms and applications*, Vol. 354. Springer.

lattice (2016). www.latticesemi.com.

Li, S., Alvarez, G., Li, Z., y Halang, W. A. (2007). Analog chaos-based secure communications and cryptanalysis: A brief survey. *arXiv preprint arXiv:0710.5455*.

Liu, J. y Chen, J. (2014). The application of speech synthesis in car warning system. En *The Proceedings of the Second International Conference on Communications, Signal Processing, and Systems*, páginas 657–662. Springer.

Liu, L., Miao, S., Hu, H., y Deng, Y. (2015). Pseudorandom bit generator based on non-stationary logistic maps. *IET Information Security*.

Lorenz, E. N. (1963). Deterministic nonperiodic flow. *Journal of the atmospheric sciences*, **20**(2): 130–141.

Mansingka, A. S., Zidan, M. A., Barakat, M. L., Radwan, A. G., y Salama, K. N. (2013). Fully digital jerk-based chaotic oscillators for high throughput pseudo-random number generators up to 8.77 gbits/s. *Microelectronics Journal*, **44**(9): 744–752.

Mao, Y. Y. y Deng, Z. C. (2011). A new image encryption algorithm of input-output feedback based on multi-chaotic system. En *Applied Mechanics and Materials*, Vol. 40, páginas 924–929. Trans Tech Publ.

Menezes, A. J., Van Oorschot, P. C., y Vanstone, S. A. (1996). *Handbook of applied cryptography*. CRC press.

Mercado-Sánchez, J., Rodríguez-Sahagún, M., y López-Mancilla, D. (2009). Encriptamiento de imágenes basado en mapeo caótico trigonométrico para comunicaciones seguras. En *Congreso Anual*, página 5.

Min, L., Lan, X., Hao, L., y Yang, X. (2014). A 6 dimensional chaotic generalized synchronization system and design of pseudorandom number generator with application in image encryption. En *Computational Intelligence and Security (CIS), 2014 Tenth International Conference on*, páginas 356–362. IEEE.

Mishkovski, I. y Kocarev, L. (2011). Chaos-based public-key cryptography. En *Chaos-Based Cryptography*, páginas 27–65. Springer.

Orúe López, A., Martínez García, M. J., Pastor Dégano, G., Montoya Vitini, F., y Sánchez Ávila, C. (2012). Criptoanálisis de un criptosistema de dos canales basado en una función no lineal caótica.

Pang, S. y Liu, Y. (2011). A new hyperchaotic system from the Lü system and its control. *Journal of Computational and Applied Mathematics*, **235**(8): 2775–2789.

Parnell, K. y Mehta, N. (2002). Programmable logic design quick start hand book. *Xilinx inc*.

Patidar, V., Pareek, N., Purohit, G., y Sud, K. (2011). A robust and secure chaotic standard map based pseudorandom permutation-substitution scheme for image encryption. *Optics Communications*, **284**(19): 4331–4339.

Pi, R. (2016). model b. *Available onlin e: https://www. raspberrypi. org/products/raspberry-pi-2-model-b/(accessed on 24 October 2016)*.

Qi, A., Zhang, C., y Wang, H. (2011). A switched hyperchaotic system and its FPGA circuitry implementation. *Journal of Electronics (China)*, **28**(3): 383–388.

Qi, G., van Wyk, M. A., van Wyk, B. J., y Chen, G. (2009). A new hyperchaotic system and its circuit implementation. *Chaos, Solitons & Fractals*, **40**(5): 2544–2549.

Rossler, O. (1979). An equation for hyperchaos. *Physics Letters A*, **71**(2): 155–157.

Shannon, C. E. (2001). A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, **5**(1): 3–55.

Sunplus Technology Co., L. (2002). SPCE061A, 16-bit sound controller with 32K x 16 flash memory.

Sunplus Technology Co., L. (2016). Sunplus technology.

Tanougast, C. (2011). Hardware implementation of chaos based cipher: Design of embedded systems for security applications. En *Chaos-Based Cryptography*, páginas 297–330. Springer.

Tlelo-Cuautle, E., Carbajal-Gomez, V., Obeso-Rodelo, P., Rangel-Magdaleno, J., y Nuñez-Perez, J. C. (2015). FPGA realization of a chaotic communication system applied to image processing. *Nonlinear Dynamics*, **82**(4): 1879–1892.

Volos, C. K. (2013). Chaotic random bit generator realized with a microcontroller. *J. Comput. Model*, **3**(4): 115–136.

Wang, X., Zhan, M., Lai, C.-H., y Gang, H. (2004). Error function attack of chaos synchronization based encryption schemes. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, **14**(1): 128–137.

Wang, X., Min, L., y Zhang, M. (2015). A generalized stability theorem for continuous chaos systems and design of pseudorandom number generator. En *Proc. 11th Int. Conf. Computational Intelligence and Security (CIS)*, páginas 375–380.

Woods, R., McAllister, J., Yi, Y., y Lightbody, G. (2008). *FPGA-based implementation of signal processing systems*. John Wiley & Sons.

Xilinx (2009). Spartan-3e FPGA family: Data sheet.

Xilinx (2011). Spartan-3E FPGA Starter Kit Board User Guide.

Xilinx (2012). System Generator for DSP-User Guide.

Xilinx (2016). www.xilinx.com.

Zhang, Q., Guo, L., y Wei, X. (2010). Image encryption using DNA addition combining with chaotic maps. *Mathematical and Computer Modelling*, **52**(11): 2028–2035.

Zhang, Y., Tao, C., y Jiang, J. J. (2006). Parameter estimation of an asymmetric vocal-fold system from glottal area time series using chaos synchronization. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, **16**(2): 023118.

Apéndice A

Otros Sistemas Caóticos Discretos

A.1 Mapeo de Hénon

El mapeo de Hénon es un sistema dinámico de tiempo discreto (Hénon, 1976). Es uno de los sistemas dinámicos, que muestran comportamiento caótico, más estudiados. El mapeo toma dos estados (x_n, y_n) en el plano y atractor el siguiente punto.

$$\begin{aligned} x_{n+1} &= 1 - \alpha x_n^2 + y_n \\ y_{n+1} &= \beta x_n \end{aligned} \tag{21}$$

El mapeo depende de dos parámetros, α y β, para mantener el mapeo clásico caótico de Hénon los valores fueron $\alpha = 1.4$ y $\beta = 0.3$. Las condiciones iniciales fueron $x_0 = 0.1$ y $y_0 = 0.15$ (Hénon, 1976). La Figura 37 muestra su atractor, el conjunto de ecuaciones de Hénon se programaron en Matlab.

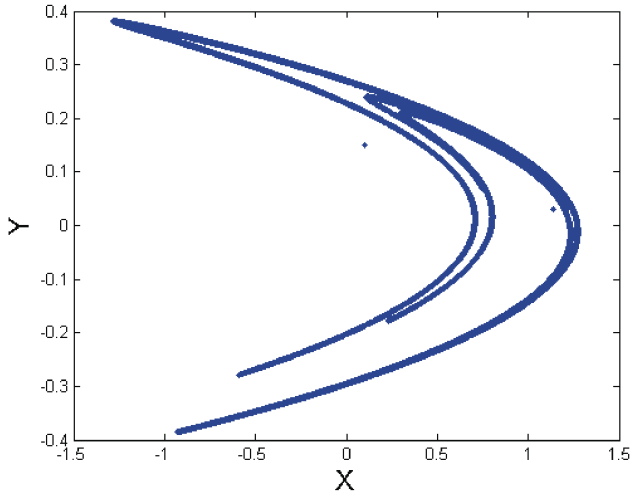


Figura 37: Atractor del mapeo caótico Hénon.

A.2 Mapeo de Tinkerbell

El mapeo de Tinkerbell (Goldsztejn *et al.*, 2011) es un sistema dinámico discreto en el tiempo. El mapeo toma dos estados (x_n, y_n) en el plano y mapea el siguiente punto.

$$\begin{aligned} x_{n+1} &= x_n^2 - y_n^2 + ax_n + by_n \\ y_{n+1} &= 2x_n y_n + cx_n + dy_n \end{aligned} \tag{22}$$

El mapeo depende de 4 parámetros, a, b, c, y d, para mantener el mapeo caótico de Tinkerbell los valores fueron $a = 0.9$, $b = -0.6013$, $c = 2$ y $d = 0.5$. Las condiciones iniciales fueron $x_0 = 0.1$ y $y_0 = 0.15$ (Goldsztejn *et al.*, 2011). La Figura 38 muestra su atractor, el conjunto de ecuaciones de Tinkerbell se programaron en Matlab.

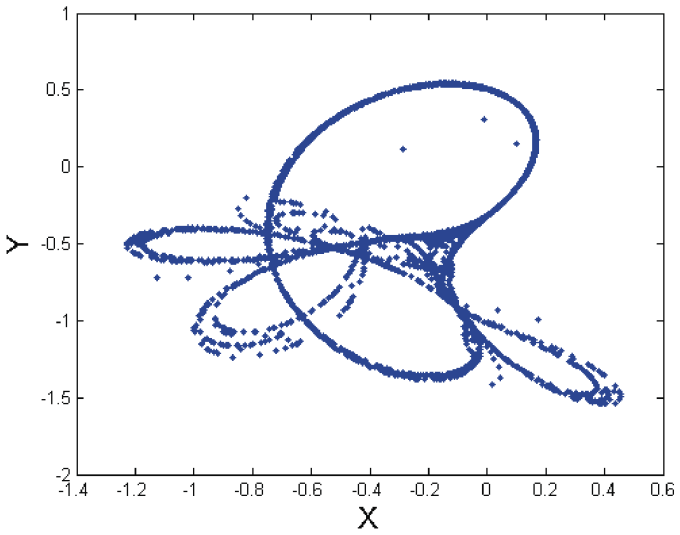


Figura 38: Atractor del mapeo caótico Tinkerbell.

A.3 Mapeo de Karplan-Yorke

El mapeo de Karplan-Yorke (Kaplan y Yorke, 1979) es un sistema dinámico discreto en el tiempo. El mapeo toma dos estados (x_n, y_n) en el plano y mapea el siguiente punto.

$$\begin{aligned} x_{n+1} &= 1 - 2(x_n^2) \\ y_{n+1} &= a y_n + x_n \end{aligned} \tag{23}$$

El mapeo depende de un parámetro, a, para mantener el mapeo caótico de Karplan-Yorke el valor es $a = 0.4$. Las condiciones iniciales son $x_0 = 0.025$ y $y_0 = 0.025$ (Kaplan y Yorke, 1979). La Figura 39 muestra su mapeo, el conjunto de ecuaciones de Karplan-Yorke se programaron en Matlab.

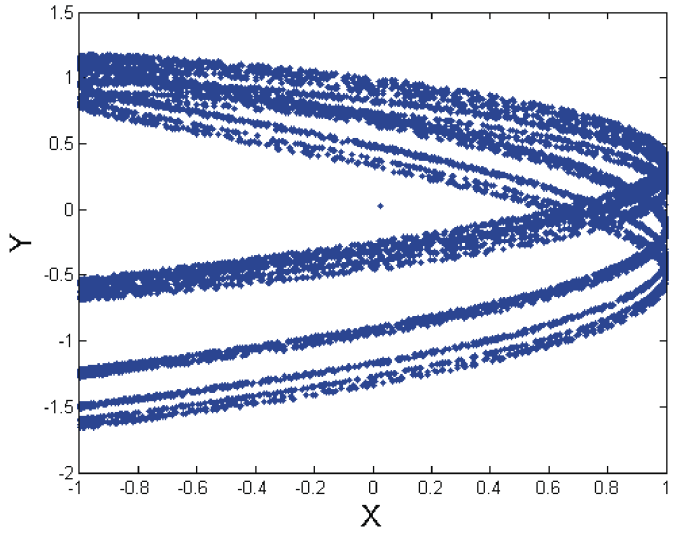


Figura 39: Atractor del mapeo caótico Karplan-Yorke.

A.4 Mapeo Logístic 2D

El mapeo de Logistic 2D (Grassberger y Procaccia, 2004) es un sistema dinámico discreto en el tiempo. El mapeo toma dos estados (x_n, y_n) en el plano y mapea el siguiente punto.

$$\begin{aligned} x_{n+1} &= \mu_1 x_n(1 - x_n) + \gamma_1 y_n^2 \\ y_{n+1} &= \mu_2 y_n(1 - y_n) + \gamma_2(x_n^2 + x_n y_n) \end{aligned} \tag{24}$$

El mapeo depende de 4 parámetros, μ_1, γ_1, μ_2, y $/gamma_2$, para mantener el mapeo caótico de Logistic 2D los valores son $\mu_1 = 3$, $\gamma_1 = 0.2$, $\mu_2 = 3.4$ y $\gamma_2 = 0.14$ (Grassberger y Procaccia, 2004). Las condiciones iniciales son $x_0 = 0.1$ y $y_0 = 0.3$. La Figura 40 muestra su mapeo, el conjunto de ecuaciones de Logistic 2D se programaron en Matlab.

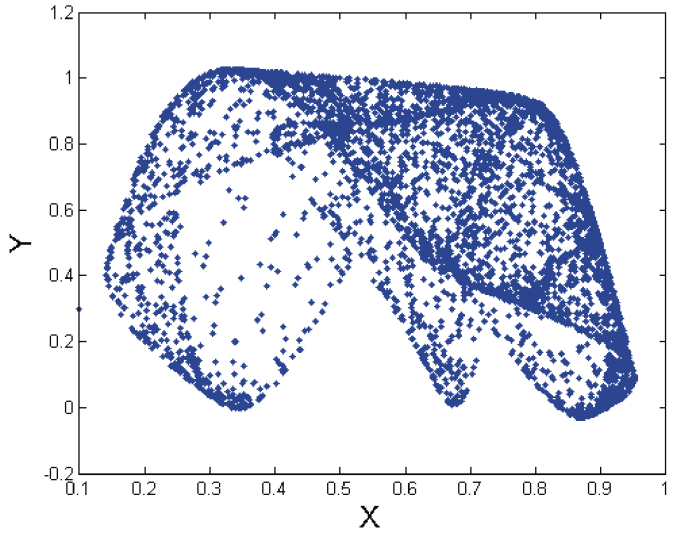


Figura 40: Atractor del mapeo caótico Logístic 2D.

Apéndice B

Otros Mapeos Implementados en el CPRGB

El presente apéndice muestra 4 mapeos implementados en FPGA utilizados en la propuesta de tesi, Hénon, Tinkerbell, Karplan-Yorke y Logśtic 2D.

B.1 Mapeo de Hénon Implementado en FPGA

A partir de la ecuación 21 del sistema caótico Hénon se diseña e implementa en Simulink utilizando "SysGen"y el resultado se visualiza en la Figura 41, se utilizan los parámetros de (Hénon, 1976).

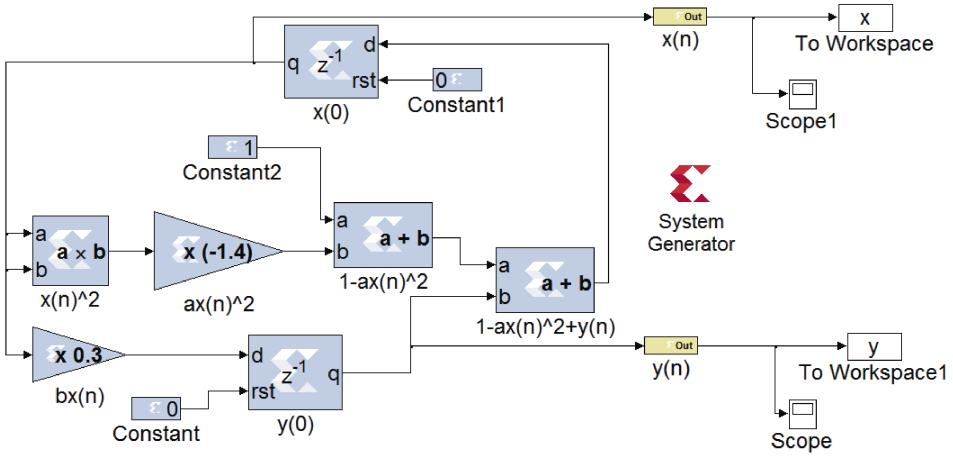
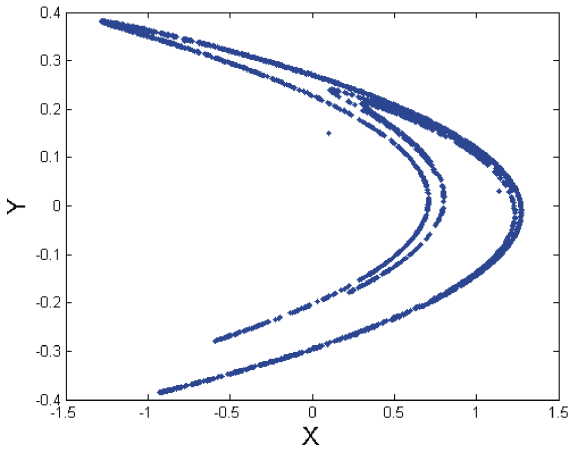
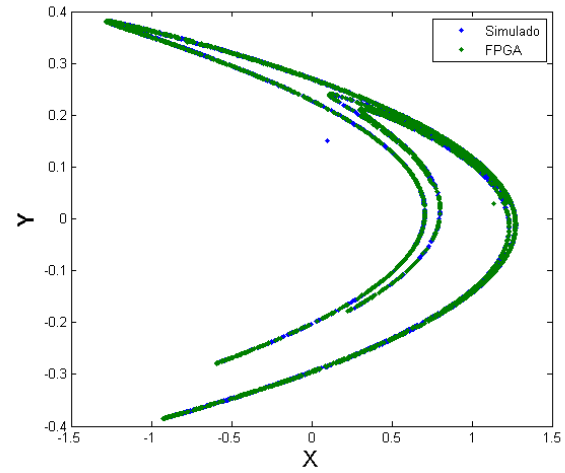


Figura 41: Implementación del mapeo de Hénon en "SysGen"para la FPGA.

El resultado de la co-simulación se ilustra en la Figura 42 y muestra el mapeo característico de Hénon.



(a) Atractor de Hénon implementado en FPGA.



(b) Comparación entre los dos mapeos (FPGA vs PC).

Figura 42: Atractor del mapeo caótico Hénon obtenido por medio de la co-simulación.

La Figura 42a ilustra el mapeo Hénon el cual mantiene su forma característica, comprobando que la implementación ha sido correcta usando una mantisa de 24Q20, 1 bit de signo, 3 bits de tipo entero y 20 bits de la parte fracionaria. La Figura 42b muestra una comparación entre dos mapeos del sistema Hénon, en azul es el programado en Matlab y en verde el implementado en FPGA se observa una diferencia entre los puntos que conforman el mapeo y se debe a la diferencia de la mantisa, pero los dos mantienen el mapeo.

B.2 Mapeo de Tinkerbell Implementado en FPGA

A partir de la ecuación 22 del sistema caótico Tinkerbell se diseña e implementa en Simulink utilizando SysGen y el resultado se visualiza en la Figura 43, se utilizan los parámetros de (Goldsztejn *et al.*, 2011).

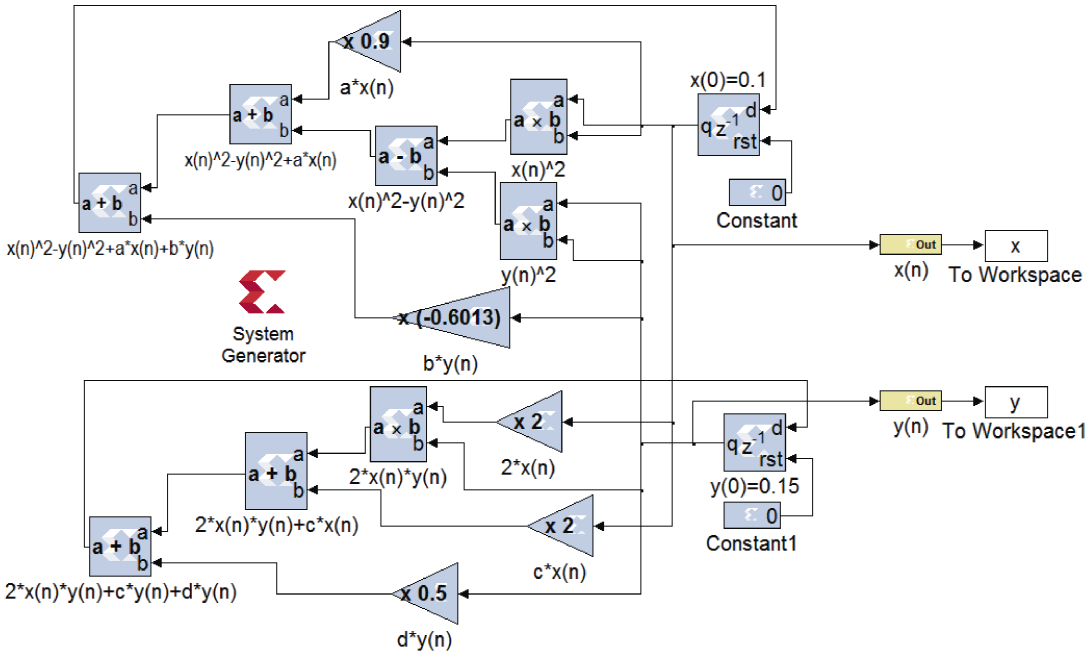
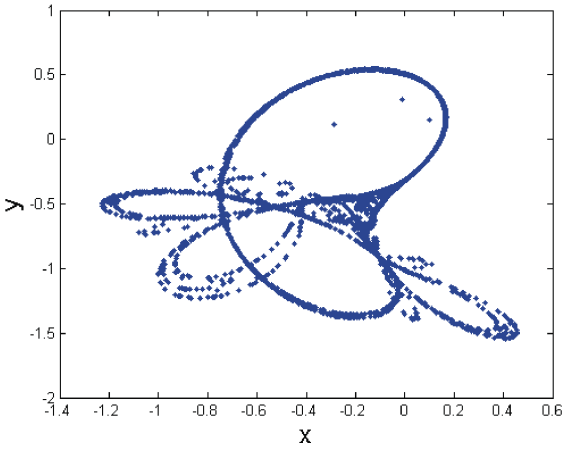
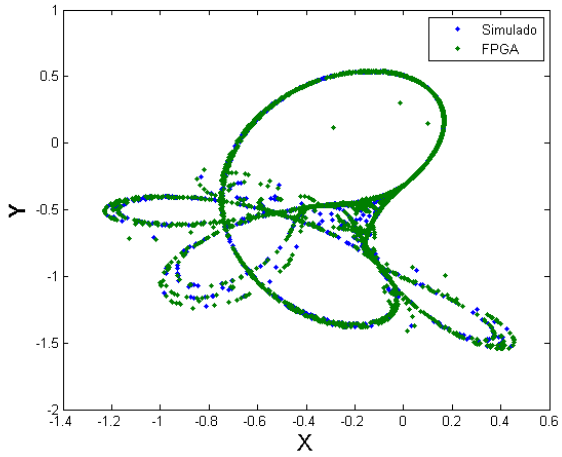


Figura 43: Implementación del mapeo de Tinkerbell en "SysGen"para la FPGA.

Se implementó una co-simulación. En la Figura 44 se ilustra el mapeo correspondiente al sistema caótico Tinkerbell.



(a) Atractor del mapeo implementado en FPGA.



(b) Comparación entre los dos mapeos obtenidos en FPGA vs PC.

Figura 44: Atractor del sistema caótico Tinkerbell obtenido por medio de la co-simulación.

La Figura 44a ilustra el mapeo Tinkerbell el cual mantiene su forma característica, comprobando que la implementación ha sido correcta usando una mantisa de 24Q20, 1 bit de signo, 3 bits de tipo entero y 20 bits de la parte fracionaria. La Figura 44b muestra una comparación entre dos mapeos del sistema Tinkerbell, en azul es el programado en Matlab y en verde el implementado en FPGA se observa una diferencia entre los puntos que conforman el mapeo y se debe a la diferencia de la mantisa.

B.3 Mapeo de Karplan-Yorke Implementado en FPGA

Tomando la ecuación 23 del sistema caótico Karplan-Yorke se diseña e implementa en Simulink utilizando SysGen y el resultado se visualiza en la Figura 45, se utilizan los parámetros de (Kaplan y Yorke, 1979).

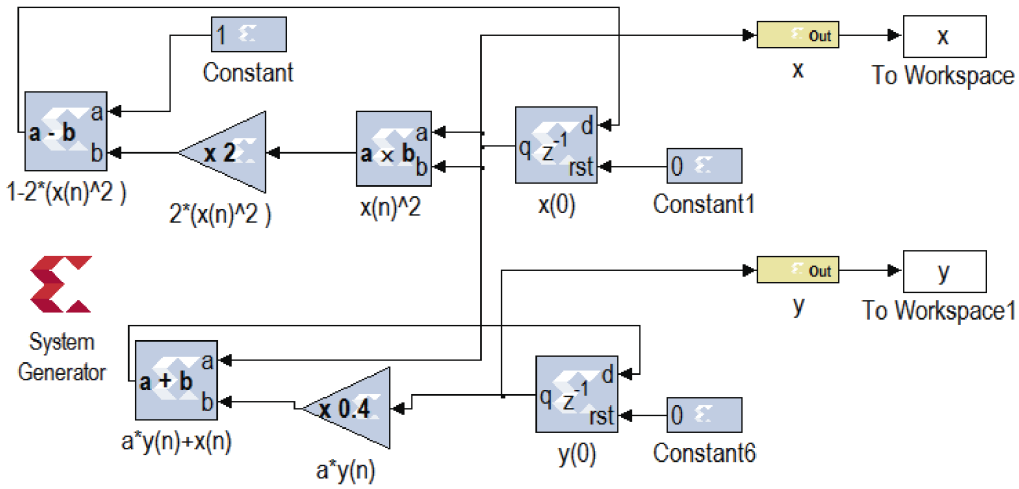
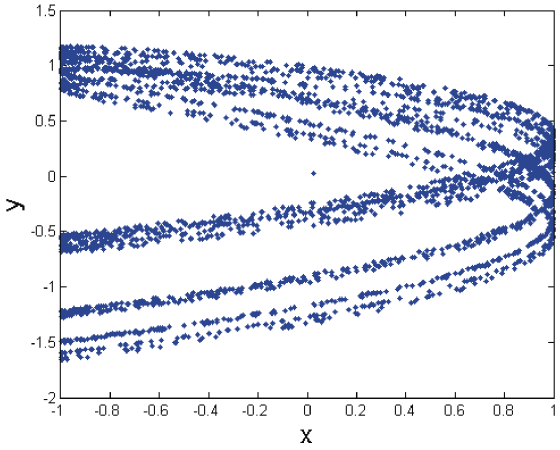
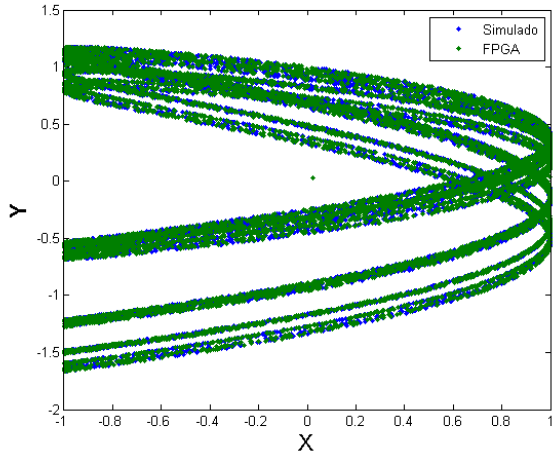


Figura 45: Implementación del mapeo de Karplan-Yorke en "SysGen"para la FPGA.

Se implementó una co-simulación. En la Figura 46 se muestra el mapeo correspondiente al sistema caótico Karplan-Yorke.



(a) Atractor del mapeo implementado en FPGA.



(b) Comparación de los atractores en FPGA vs PC.

Figura 46: Mapeo del sistema caótico Karplan-Yorke obtenido por medio de la co-simulación.

La Figura 46a muestra el mapeo Karplan-Yorke el cual mantiene su forma característica, comprobando que la implementación ha sido correcta usando una mantisa de $32Q30$, 1 bit de signo, 1 bits de tipo entero y 30 bits de la parte fracionaria. La Figura 46b muestra una comparación entre dos mapeos del sistema Karplan-Yorke, en azul es el programado en Matlab y en verde el implementado en FPGA se observa una diferencia entre los puntos que conforman el mapeo y se debe a la diferencia de la mantisa.

B.4 Mapeo Logístic 2D Implementado en FPGA

A partir de la ecuación 24 del sistema caótico Logístic 2D se diseña e implementa en Simulink utilizando SysGen y el resultado se visualiza en la Figura 47, se utilizan los parámetros de (Grassberger y Procaccia, 2004).

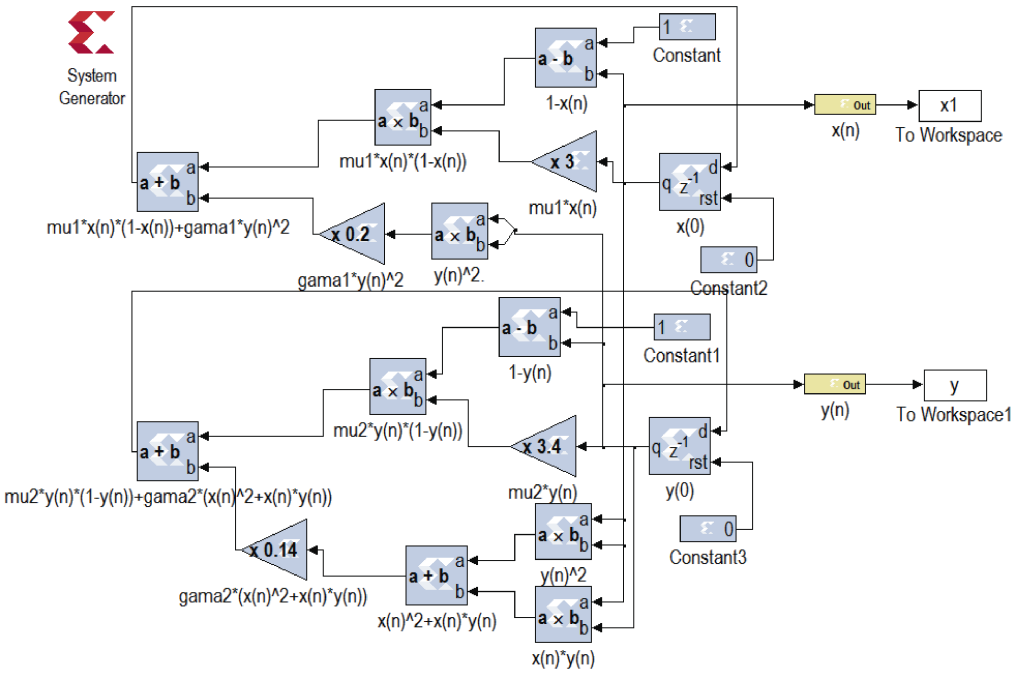
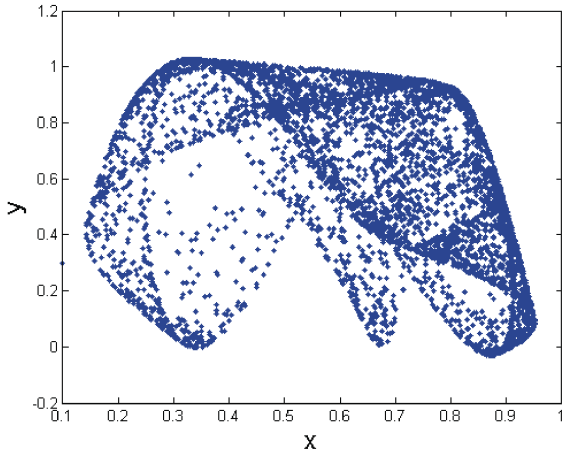
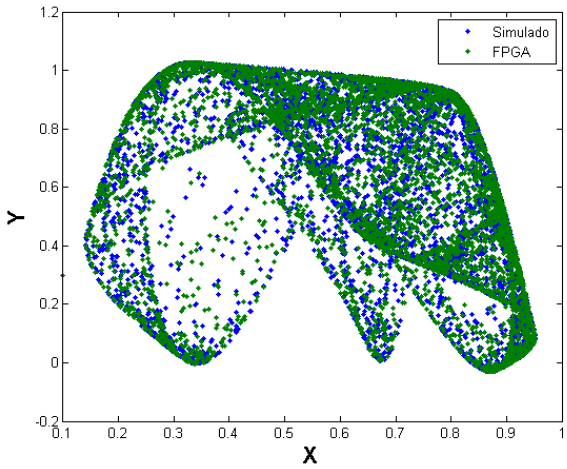


Figura 47: Implementación del mapeo de Logístic 2D en "SysGen"para la FPGA.

Se implementó una co-simulación. En la Figura 48 se ilustra el mapeo correspondiente al sistema caótico Logístic 2D.



(a) Atractor del mapeo implementado en FPGA.



(b) Comparación entre los dos mapeos FPGA vs PC.

Figura 48: Mapeo del sistema caótico Logístic 2D obtenido por medio de la co-simulación.

La Figura 48a muestra el mapeo Logístic 2D el cual mantiene su forma característica, comprobando que la implementación ha sido correcta usando una mantisa de 24Q20, 1 bit de signo, 3 bits de tipo entero y 20 bits de la parte fracionaria. La Figura 48b muestra una comparación entre dos mapeos del sistema Logístic 2D, en azul es el programado en Matlab y en verde el implementado en FPGA se observa una diferencia entre los puntos que conforman el mapeo y se debe a la diferencia de la mantisa.

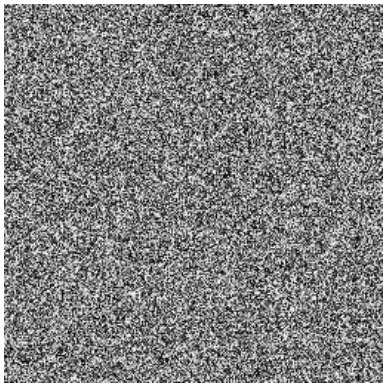
Apéndice C

Otros Resultados

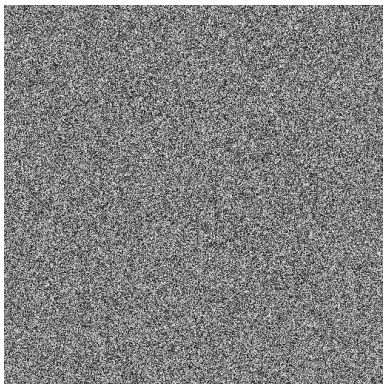
El apendice ilustra los recursos necesarios para la implementación, los respectivos criptogramas y por último los análisis estadísticos.

C.1 Criptograma usando el Sistema Hénon

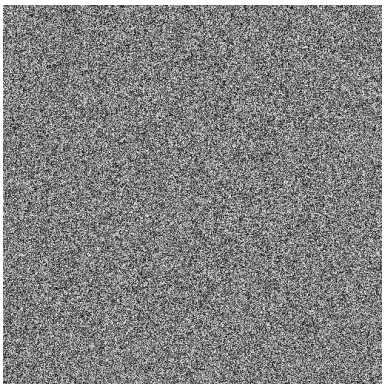
La Figura 49 presenta 3 criptogramas a partir del CPRBG con Hénon, las 3 imágenes representan el cifrado caótico. La Figura 49b y 49c estan constituidas por $512x512$ píxeles, con una simple observación se identifica la diferencia entre las mismas.



(a) Criptograma de Lena $255x255$.



(b) Criptograma de Cameraman $512x512$.

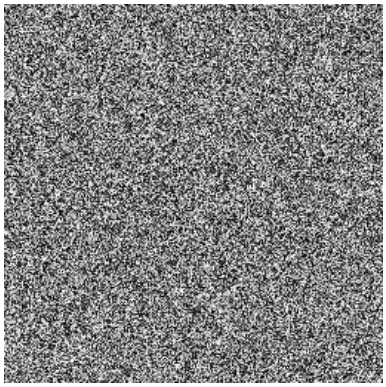


(c) Criptograma de Mandril $512x512$.

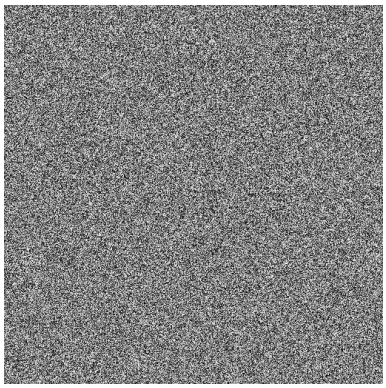
Figura 49: Criptogramas a partir del CPRBG con Hénon: Lena Figura 29a, Cameraman Figura 29b y Mandril Figura 29d.

C.2 Criptograma usando el Sistema Tinkerbell

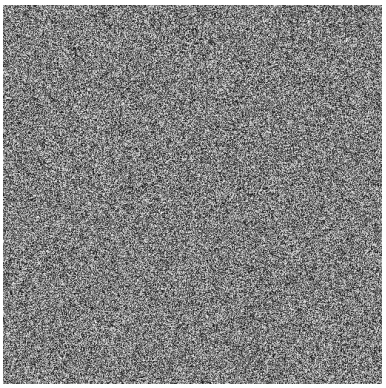
La Figura 50 presenta 3 criptogramas a partir del CPRBG con Tinkerbell, las 3 imágenes representan el cifrado caótico. Las Figuras 50b y 50c estan constituidas por $512x512$ píxeles, con una simple observación se identifica la diferencia entre las mismas.



(a) Criptograma de Lena $255x255$.



(b) Criptograma de Cameraman $512x512$.

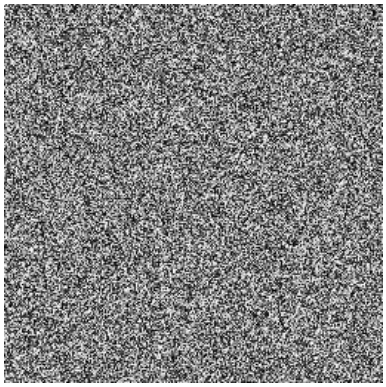


(c) Criptograma de Mandril $512x512$.

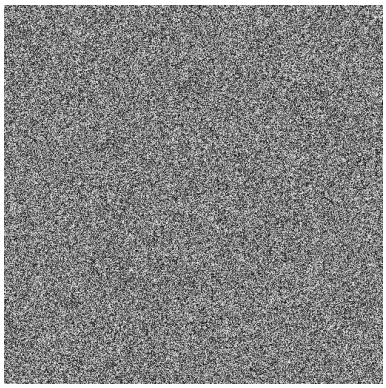
Figura 50: Criptogramas a partir del CPRBG con Tinkerbell: Lena 29a, Cameraman 29b y Mandril 29d.

C.3 Criptograma usando el Sistema Karplan-Yorke

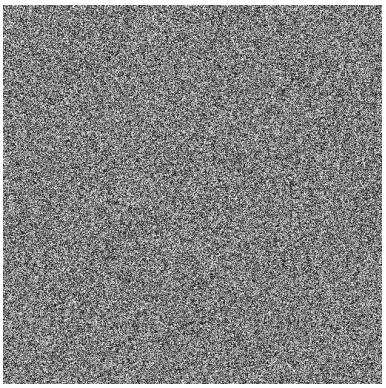
La Figura 51 presenta 3 criptogramas a partir del CPRBG con Karplan-Yorke, las 3 figuras ilustran el enmascaramiento caótico. Las Figuras 51b y 51c tienen $512x512$ píxeles, analizando las dos imágenes se encuentran diferentes entre si.



(a) Criptograma de Lena $255x255$.



(b) Criptograma de Cameraman $512x512$.

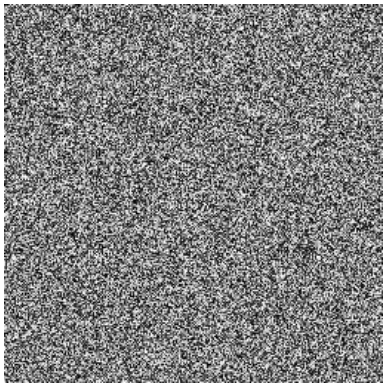


(c) Criptograma de Mandril $512x512$.

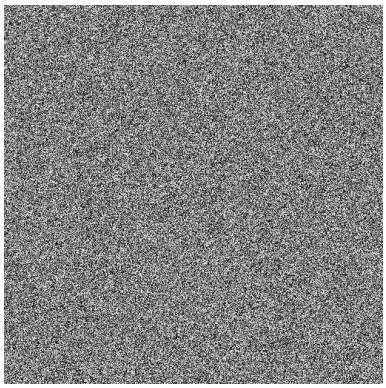
Figura 51: Criptogramas a partir del CPRBG con Karplan-Yorke: Lena 29a, Cameraman 29b y Mandril 29d.

C.4 Criptograma usando el Sistema Logístic 2D

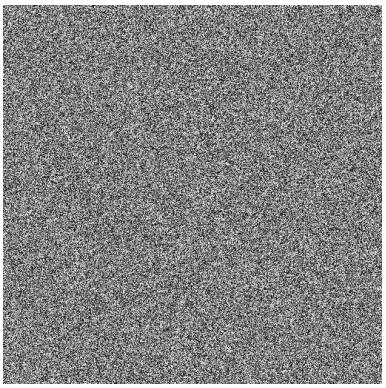
La figura 52 presenta 3 criptogramas a partir del CPRBG con Logístic 2D, las 3 imágenes representan el cifrado caótico. Las Figuras 52b y 52c estan constituidas por $512x512$ píxeles, con una simple observación se identifica la diferencia entre las mismas.



(a) Criptograma de Lena $255x255$.



(b) Criptograma de Cameraman $512x512$.



(c) Criptograma de Mandril $512x512$.

Figura 52: Criptogramas a partir del CPRBG con Logístic 2D: Lena 29a, Cameraman 29b y Mandril 29d.

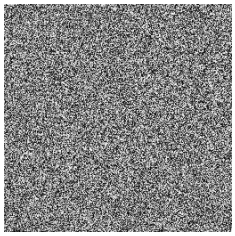
C.5 Análisis estadísticos de seguridad a otros criptogramas

En está sección se presentan los anánalisis estadísticos gráficos a los criptogramas obtenidos a partir de usar los sistemas caóticos Hénon, Tinkerbell, Karplan-Yorke y Logístic 2D. En la sección II.2 se prersenta la teoría del análisis por histogramnas y gráficas de correlación de pixeles adyacentes.

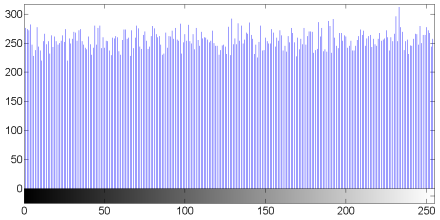
Cabe señalar que los resultados son simnilares a simple vista, se recomienda observar detenidamente los rersultados y se comprobará visualmente la existencia de la miníma diferencia que existe al usar diferentes mapeos caóticos.

C.5.1 Análisis de Sensibilidad de claves

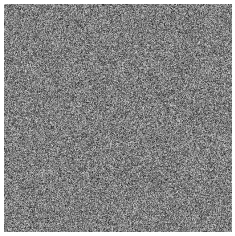
La Figura 53 ilustra el análisis gráfico de la sensibilidad al mínimo cambio en la clave, las imágenes recuperadas 53a, 53c y 53e son irreconocibles, tanto así que las Figuras 53b, 53d y 53f muestran, respectivamente, el histograma de la imagen recuperada el cual es uniforme, por lo tanto las imágenes recuperadas no contienen información de la imagen original.



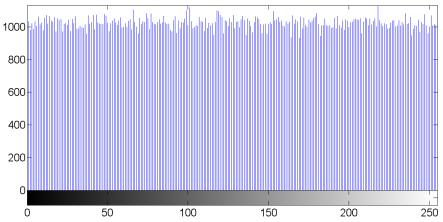
(a) Imagen no recuperada de Lena 29a.



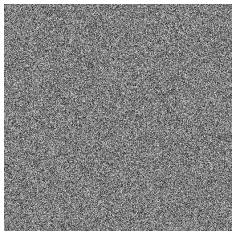
(b) Histograma del criptograma de Lena no recuperada 53a.



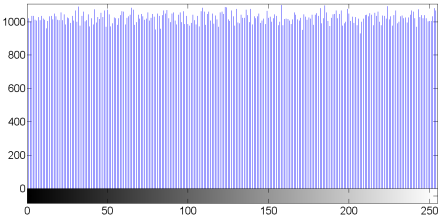
(c) Imagen no recuperada de Cameraman 29b.



(d) Histograma del criptograma de Cameraman no recuperada 53c.



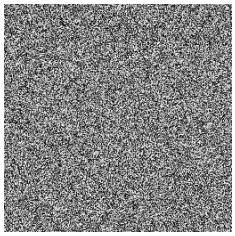
(e) Imagen no recuperada de Mandril 29c.



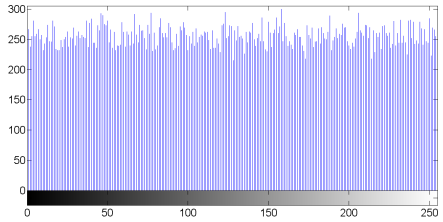
(f) Histograma del criptograma de Mandril no recuperada 53e.

Figura 53: Análisis de sensibilidad de clave utilizando el sistema Hénon.

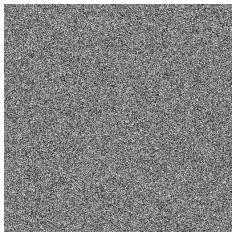
La Figura 54 ilustra el análisis gráfico de la sensibilidad al mínimo cambio en la clave, las imágenes recuperadas 54a, 54c y 54e son irreconocibles, tanto así que las Figuras 54b, 54d y 54f muestran, respectivamente, el histograma de la imagen recuperada el cual es uniforme, por lo tanto las imágenes recuperadas no contienen información de la imagen original.



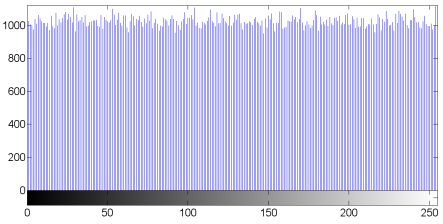
(a) Imagen no recuperada de Lena 29a.



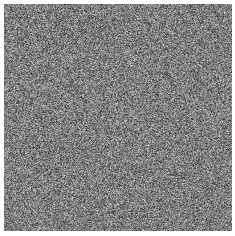
(b) Histograma del criptograma de Lena no recuperada 54a.



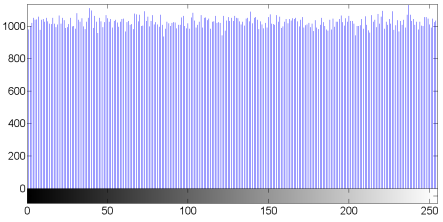
(c) Imagen no recuperada de Cameraman 29b.



(d) Histograma del criptograma de Cameraman no recuperada 54c.



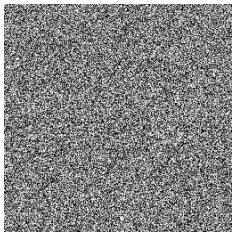
(e) Imagen no recuperada de Mandril 29c.



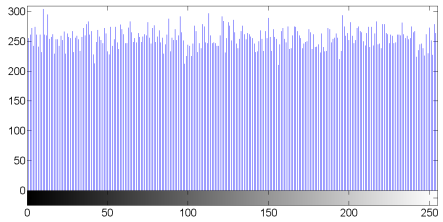
(f) Histograma del criptograma de Mandril no recuperada 54e.

Figura 54: Análisis de sensibilidad de clave utilizando el sistema Tinkerbell.

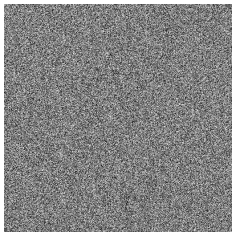
La Figura 55 muestra el resultado gráfico del análisis a la sensibilidad al mínimo cambio en la clave, las Figuras adquiridas 55a, 55c y 55e son ilegibles, tanto así que las imágenes 55b, 55d y 55f muestran, respectivamente, el histograma de la figura recuperada el cual es uniforme, por lo tanto las figuras adquiridas no contienen información de la imagen original.



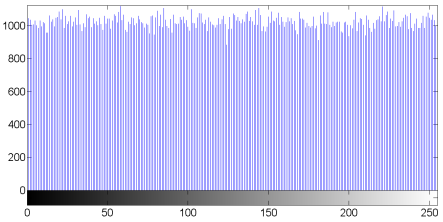
(a) Imagen no recuperada de Lena 29a.



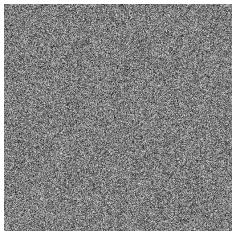
(b) Histograma del criptograma de Lena no recuperada 55a.



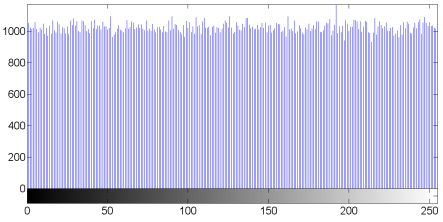
(c) Imagen no recuperada de Cameraman 29b.



(d) Histograma del criptograma de Cameraman no recuperada 55c.



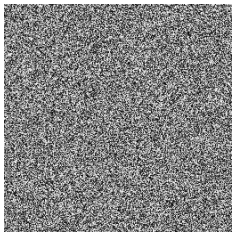
(e) Imagen no recuperada de Mandril 29c.



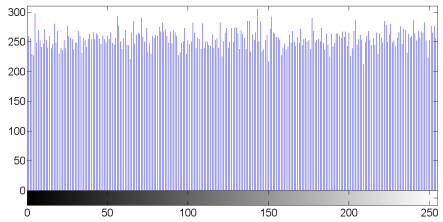
(f) Histograma del criptograma de Mandril no recuperada 55e.

Figura 55: Análisis de sensibilidad de clave utilizando el sistema Karplan-Yorke.

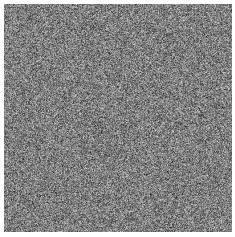
La Figura 56 ilustra el análisis gráfico de la sensibilidad al mínimo cambio en la clave, las Figuras recuperadas 56a, 56c y 56e son irreconocibles, tanto así que las figuras 56b, 56d y 56f muestran, respectivamente, el histograma de la imagen recuperada el cual es uniforme, por lo tanto las imágenes recuperadas no contienen información de la imagen original.



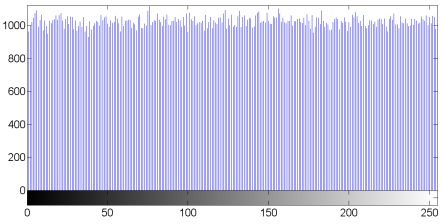
(a) Imagen no recuperada de Lena 29a.



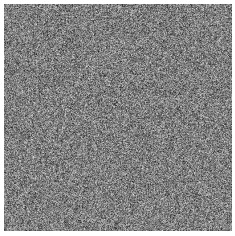
(b) Histograma del criptograma de Lena no recuperada 56a.



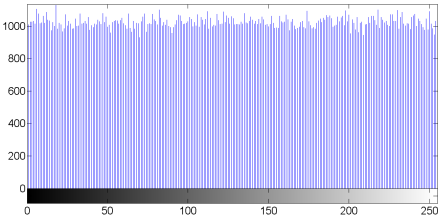
(c) Imagen no recuperada de Cameraman 29b.



(d) Histograma del criptograma de Cameraman no recuperada 56c.



(e) Imagen no recuperada de Mandril 29c.

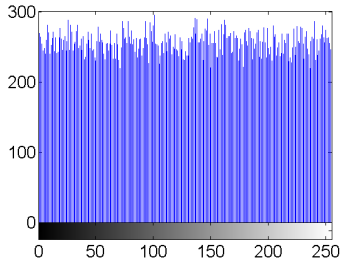


(f) Histograma del criptograma de Mandril no recuperada 56e.

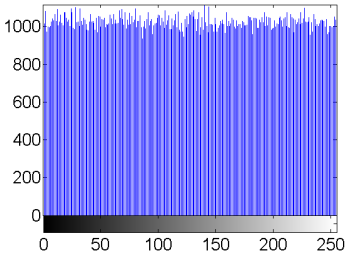
Figura 56: Análisis de sensibilidad de clave utilizando el sistema Logístic 2D.

C.5.2 Histogramas de los criptogramas del sistema Hénon

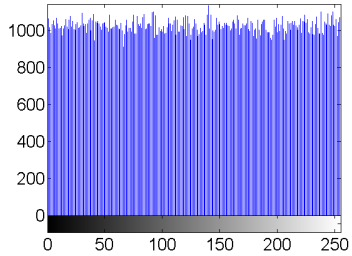
El histograma de una imagen regresenta de forma gráfica la información por la cual está constituida la imagen. La Figura 57 representa los histogramas de los criptogramas, ver Figuras 49a, 49b y 49c, a partir del CPRBG con Hénon.



(a) Histograma del criptograma de Lena $255x255$ figura 49a.



(b) Histograma del criptograma Cameraman $512x512$ figura 49b.



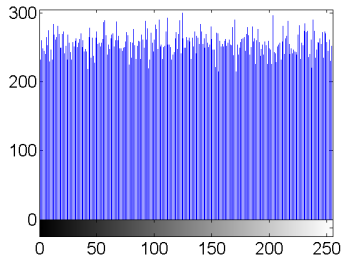
(c) Histograma del criptograma Mandril $512x512$ figura 49c.

Figura 57: Histogramas de los criptogramas a partir del CPRBG con Hénon: 49a, 49b y 49c.

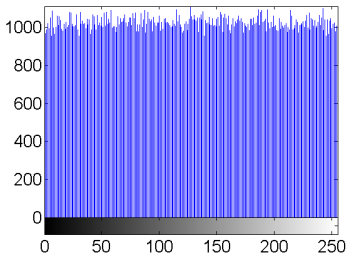
Los histogramas de los criptogramas 49a, 49b y 49c tienen una distribución uniforme, por lo tanto no contienen información de las imágenes originales.

C.5.3 Histogramas de los criptogramas del sistema Tinkerbell

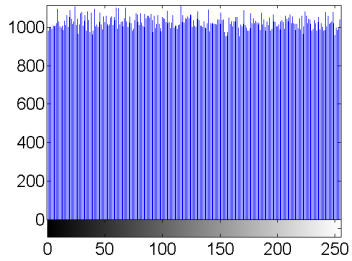
La Figura 58 representa los histogramas de los criptogramas, ver Figuras 50a, 50b y 50c, a partir del CPRBG con Tinkerbell.



(a) Histograma del criptograma de Lena $255x255$ figura 50a.



(b) Histograma del criptograma Cameraman $512x512$ figura 50b.



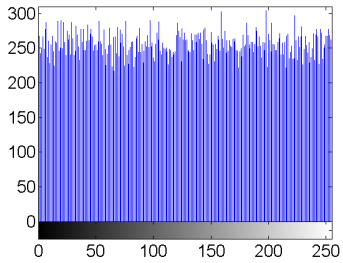
(c) Histograma del criptograma Mandril $512x512$ figura 50c.

Figura 58: Histogramas de los criptogramas a partir del CPRBG con Tinkerbell: 50a, 50b y 50c.

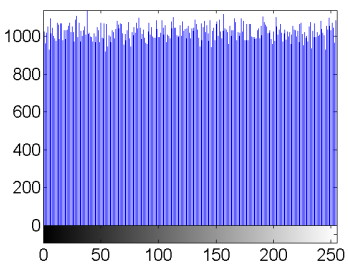
Los histogramas de los criptogramas 50a, 50b y 50c tienen una distribución uniforme, por lo tanto no contienen información de las imágenes originales.

C.5.4 Histogramas de los criptogramas del sistema Karplan Yorke

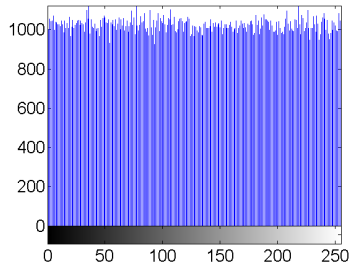
La Figura 59 ilustra los histogramas de los criptogramas, ver Figuras 51a, 51b y 51c, a partir del CPRBG con Karplan Yorke.



(a) Histograma del criptograma de Lena $255x255$ figura 51a.



(b) Histograma del criptograma Cameraman $512x512$ figura 51b.



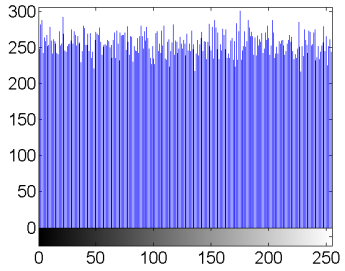
(c) Histograma del criptograma Mandril $512x512$ figura 51c.

Figura 59: Histogramas de los criptogramas a partir del CPRBG con Karplan Yorke: 51a, 51b y 51c.

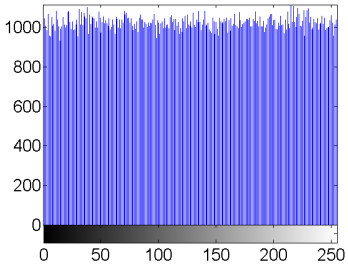
Los histogramas de los criptogramas 51a, 51b y 51c ilustran una distribución uniforme, por lo tanto no contienen información de las imágenes originales.

C.5.5 Histogramas de los criptogramas del sistema Logístic 2D

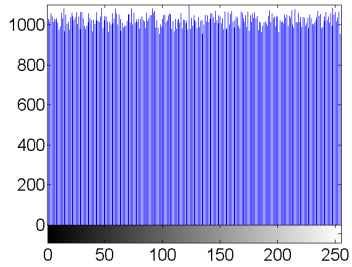
La Figura 60 representa los histogramas de los criptogramas, ver Figuras 52a, 52b y 52c, a partir del CPRBG con Logístic 2D.



(a) Histograma del criptograma de Lena $255x255$ figura 52a.



(b) Histograma del criptograma Cameraman $512x512$ figura 52b.



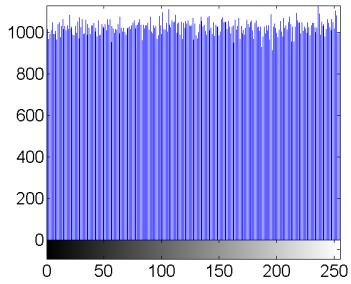
(c) Histograma del criptograma Mandril $512x512$ figura 52c.

Figura 60: Histogramas de los criptogramas a partir del CPRBG con Logístic 2D: 52a, 52b y 52c.

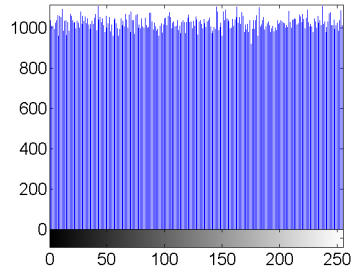
Los histogramas de los criptogramas 52a, 52b y 52c tienen una distribución uniforme, por lo tanto no contienen información de las imágenes originales.

C.5.6 Histogramas de los criptogramas de la imagen RGB usando el sistema Rössler

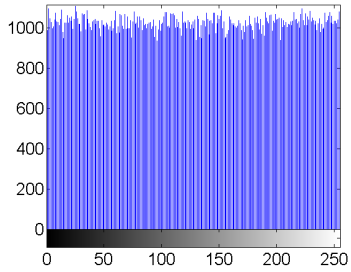
La Figura 61 ilustra los histogramas de los 3 criptogramas, ver Figuras 61a, 61b y 61c, de la imagen Lena RGB, ver Figura 29c, a partir del CPRBG con Rössler. La Figura 61d muestra el histograma del criptograma 31d.



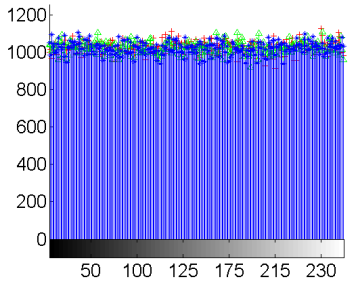
(a) Histograma criptograma de la parte roja de Lena RGB $512x512$.



(b) Histograma del criptograma de la parte verde de Lena RGB $512x512$.



(c) Histograma del criptograma de la parte azul de Lena RGB $512x512$.



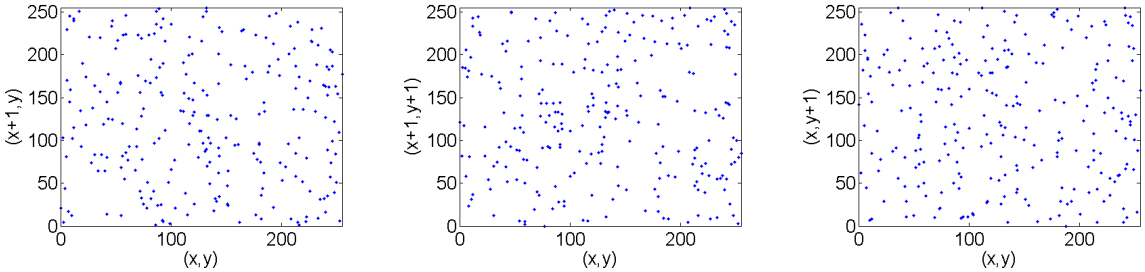
(d) Histograma del criptograma de Lena RGB $512x512$.

Figura 61: Histograma de los criptogramas a partir del CPRBG con Rössler usando a la Imagen Lena RGB $512x512$ 29c.

Los histogramas de la igura 61 tienen una distribución uniforme, por lo tanto no contienen información de las imágenes originales.

C.5.7 Gráfica de la correlación de píxeles adyacentes usando Hénon

La Figura 62 ilustra gráficamente la correlación de pẋeles adyacentes del criptograma 49a utilizando el CPRGB con el sistema Hénon. La Figura 62a ilustra la correlación de píxeles adyacentes de forma vertica, donde la distribución de los puntos en la gráfica es disperza por lo tanto no existe una correlación de pẋeles adyacentes. El mismo comportamiento aparece en las Figuras 62b y 62c que son las correlaciones de pixeles adyacentes en diagonal y horizontal resptectivamente, como la distrución es una disperciońen la gráfica no existe correlación de pixeles adyacentes.

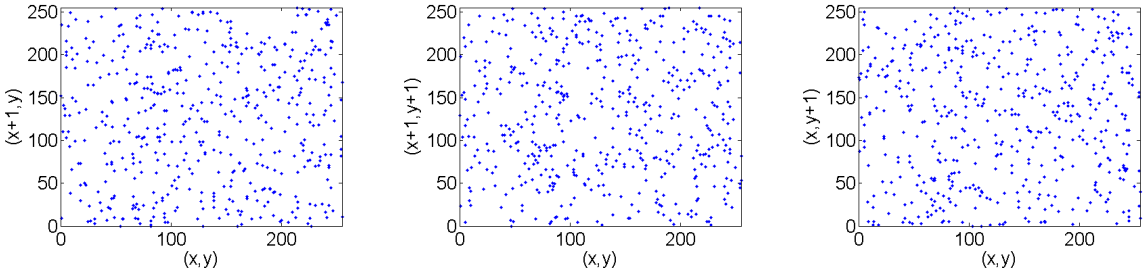


(a) Correlación vertical. (b) Correlación diagonal. (c) Correlación horizontal.

Figura 62: Correlación de píxeles adyacentes del criptograma Lena usando el sistema Hénon.

La Figura 63 muestra gráficamente la correlación de pẋeles adyacentes del criptograma 49b utilizando el CPRGB con el sistema Hénon. La Figura 63a ilustra la correlación de píxeles adyacentes de forma vertica, donde la distribución de los puntos en la gráfica es disperza por lo tanto no existe una correlación de pẋeles adyacentes. El mismo comportamiento aparece en las Figuras 63b y 63c que son las correlaciones de pixeles adyacentes en diagonal y horizontal resptectivamente, como la distrución es una disperciońen la gráfica no existe correlación de pixeles adyacentes.

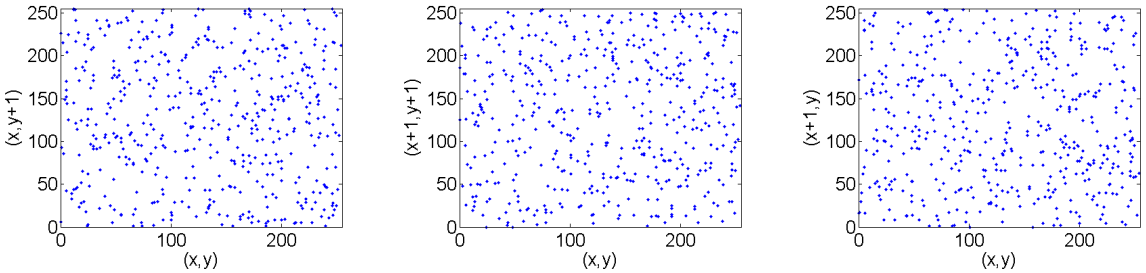
La Figura 64 ilustra gráficamente la correlación de pẋeles adyacentes del criptograma



(a) Correlación vertical. (b) Correlación diagonal. (c) Correlación horizontal.

Figura 63: Correlación de píxeles adyacentes del criptograma Cameraman usando el sistema Hénon.

49c utilizando el CPRGB con el sistema Hénon. La Figura 64a ilustra la correlación de píxeles adyacentes de forma vertica, donde la distribución de los puntos en la gráfica es disperza por lo tanto no existe una correlación de pẋeles adyacentes. El mismo comportamiento aparece en las Figuras 64b y 64c que son las correlaciones de pixeles adyacentes en diagonal y horizontal resptectivamente, como la distrución es una disperciońen la gráfica no existe correlación de pixeles adyacentes.

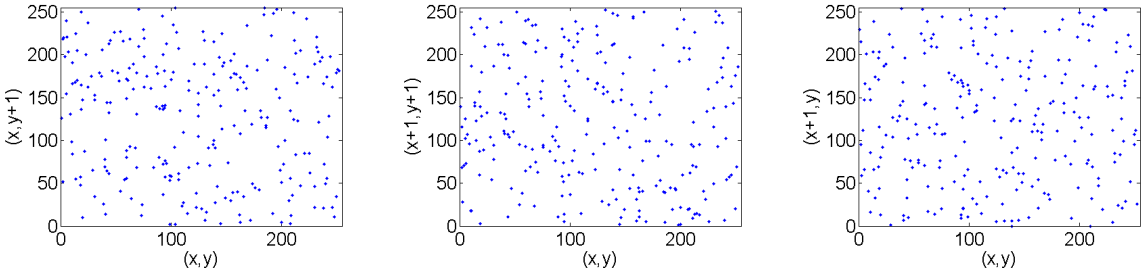


(a) Correlación vertical. (b) Correlación diagonal. (c) Correlación horizontal.

Figura 64: Correlación de píxeles adyacentes del criptograma Mandril usando el sistema Hénon.

C.5.8 Gráfica de la correlación de píxeles adyacentes usando tinkerbell

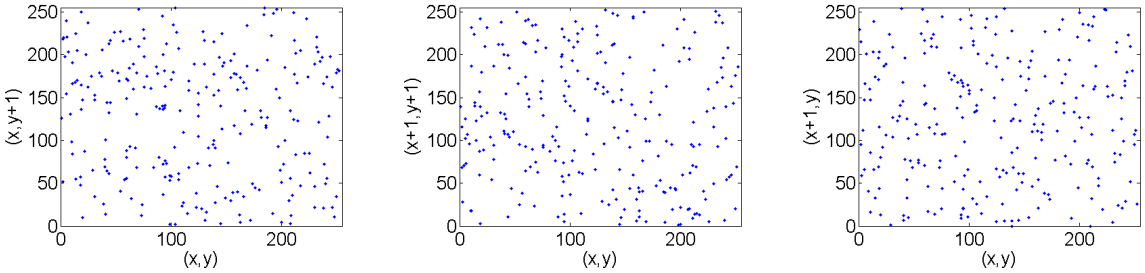
La Figura 65 presenta gráficamente la correlación de pẋeles adyacentes del criptograma 49a utilizando el CPRGB con el sistema Tinkerbell. La Figura 65a ilustra la correlación de píxeles adyacentes de forma vertica, donde la distribución de los puntos en la gráfica es disperza por lo tanto no existe una correlación de pẋeles adyacentes. El mismo comportamiento aparece en las Figuras 65b y 65c que son las correlaciones de pixeles adyacentes en diagonal y horizontal resptectivamente, como la distrución es una disperciońen la gráfica no existe correlación de pixeles adyacentes.



(a) Correlación vertical. (b) Correlación diagonal. (c) Correlación horizontal.

Figura 65: Correlación de píxeles adyacentes del criptograma Lena usando el sistema Tinkerbell.

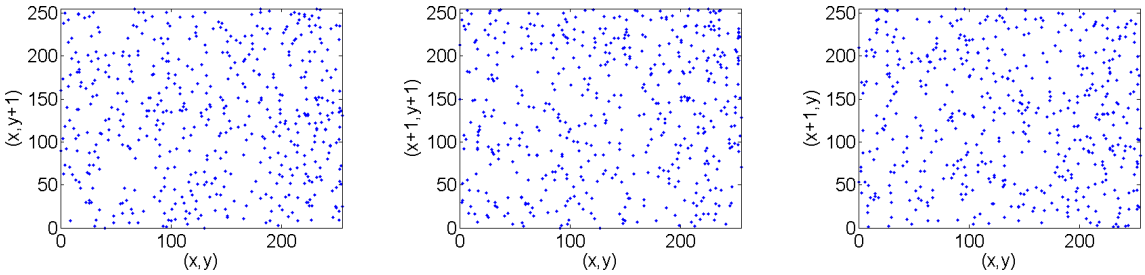
La Figura 66 ilustra gráficamente la correlación de pẋeles adyacentes del criptograma 49b utilizando el CPRGB con el sistema Tinkerbell. La Figura 66a ilustra la correlación de píxeles adyacentes de forma vertica, donde la distribución de los puntos en la gráfica es disperza por lo tanto no existe una correlación de pẋeles adyacentes. El mismo comportamiento aparece en las Figuras 66b y 66c que son las correlaciones de pixeles adyacentes en diagonal y horizontal resptectivamente, como la distrución es una disperciońen la gráfica no existe correlación de pixeles adyacentes.



(a) Correlación vertical. (b) Correlación diagonal. (c) Correlación horizontal.

Figura 66: Correlación de píxeles adyacentes del criptograma Cameraman usando el sistema Tinkerbell.

La Figura 67 muestra gráficamente la correlación de pẋeles adyacentes del criptograma 49c utilizando el CPRGB con el sistema Tinkerbell. La Figura 67a ilustra la correlación de píxeles adyacentes de forma vertica, donde la distribución de los puntos en la gráfica es disperza por lo tanto no existe una correlación de pẋeles adyacentes. El mismo comportamiento aparece en las Figuras 67b y 67c que son las correlaciones de pixeles adyacentes en diagonal y horizontal resptectivamente, como la distrución es una disperciońen la gráfica no existe correlación de pixeles adyacentes.

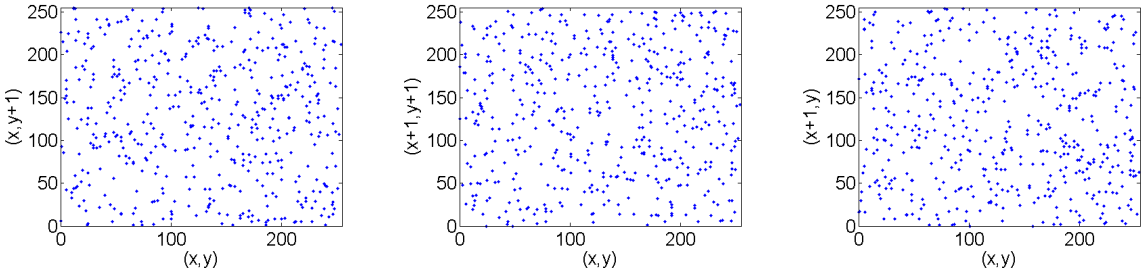


(a) Correlación vertical. (b) Correlación diagonal. (c) Correlación horizontal.

Figura 67: Correlación de píxeles adyacentes del criptograma Mandril usando el sistema Tinkerbell.

C.5.9 Gráfica de la correlación de píxeles adyacentes usando Karplan-Yorke

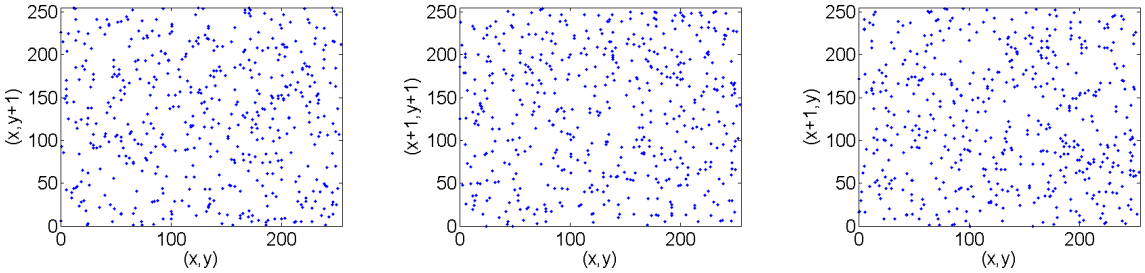
La Figura 68 muestra gráficamente la correlación de pẋeles adyacentes del criptograma 49a utilizando el CPRGB con el sistema Karplan Yorke. La Figura 68a ilustra la correlación de píxeles adyacentes de forma vertica, donde la distribución de los puntos en la gráfica es disperza por lo tanto no existe una correlación de pẋeles adyacentes. El mismo comportamiento aparece en las Figuras 68b y 68c que son las correlaciones de pixeles adyacentes en diagonal y horizontal resptectivamente, como la distrución es una disperciońen la gráfica no existe correlación de pixeles adyacentes.



(a) Correlación vertical. (b) Correlación diagonal. (c) Correlación horizontal.

Figura 68: Correlación de píxeles adyacentes del criptograma Lena usando el sistema Karplan Yorke.

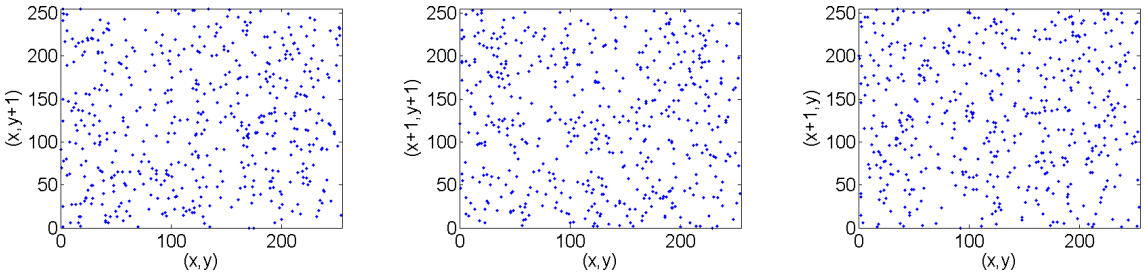
La Figura 69 ilustra gráficamente la correlación de pẋeles adyacentes del criptograma 49b utilizando el CPRGB con el sistema Karplan Yorke. La Figura 69a ilustra la correlación de píxeles adyacentes de forma vertica, donde la distribución de los puntos en la gráfica es disperza por lo tanto no existe una correlación de pẋeles adyacentes. El mismo comportamiento aparece en las Figuras 69b y 69c que son las correlaciones de pixeles adyacentes en diagonal y horizontal resptectivamente, como la distrución es una disperciońen la gráfica no existe correlación de pixeles adyacentes.



(a) Correlación vertical. (b) Correlación diagonal. (c) Correlación horizontal.

Figura 69: Correlación de píxeles adyacentes del criptograma Cameraman usando el sistema Karplan Yorke.

La Figura 68 muestra gráficamente la correlación de pẋeles adyacentes del criptograma 49a utilizando el CPRGB con el sistema Karplan Yorke. La Figura 68a ilustra la correlación de píxeles adyacentes de forma vertica, donde la distribución de los puntos en la gráfica es disperza por lo tanto no existe una correlación de pẋeles adyacentes. El mismo comportamiento aparece en las Figuras 68b y 68c que son las correlaciones de pixeles adyacentes en diagonal y horizontal resptectivamente, como la distrución es una disperciońen la gráfica no existe correlación de pixeles adyacentes.

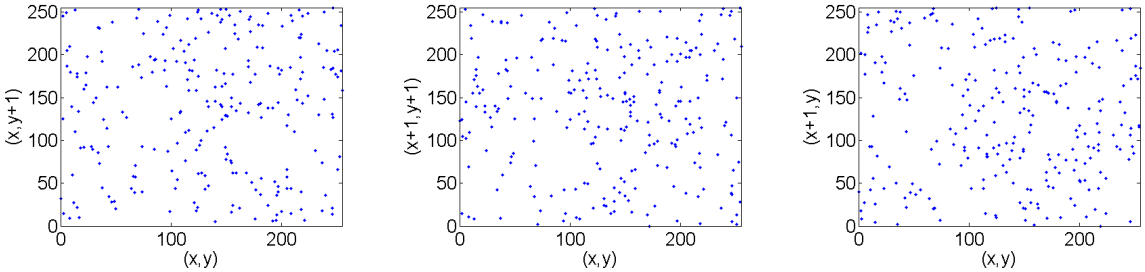


(a) Correlación vertical. (b) Correlación diagonal. (c) Correlación horizontal.

Figura 70: Correlación de píxeles adyacentes del criptograma Mandril usando el sistema Karplan Yorke.

C.5.10 Gráfica de la correlación de píxeles adyacentes usando Logístic 2D

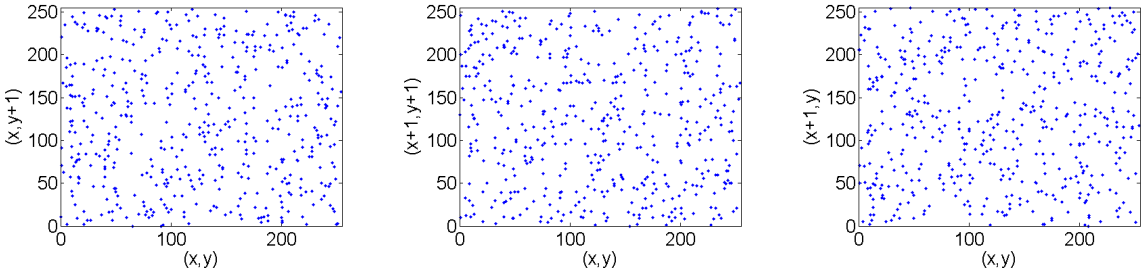
La Figura 71 ilustra gráficamente la correlación de pẋeles adyacentes del criptograma 49a utilizando el CPRGB con el sistema Logístic 2D. La Figura 71a ilustra la correlación de píxeles adyacentes de forma vertica, donde la distribución de los puntos en la gráfica es disperza por lo tanto no existe una correlación de pẋeles adyacentes. El mismo comportamiento aparece en las Figuras 71b y 71c que son las correlaciones de pixeles adyacentes en diagonal y horizontal resptectivamente, como la distrución es una disperciońen la gráfica no existe correlación de pixeles adyacentes.



(a) Correlación vertical. (b) Correlación diagonal. (c) Correlación horizontal.

Figura 71: Correlación de píxeles adyacentes del criptograma Lena usando el sistema Logístic 2D.

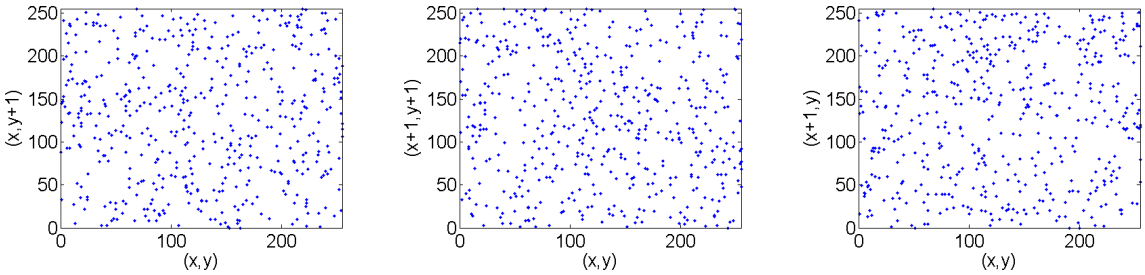
La Figura 72 muestra gráficamente la correlación de pẋeles adyacentes del criptograma 49b utilizando el CPRGB con el sistema Logístic 2D. La Figura 72a ilustra la correlación de píxeles adyacentes de forma vertica, donde la distribución de los puntos en la gráfica es disperza por lo tanto no existe una correlación de pẋeles adyacentes. El mismo comportamiento aparece en las Figuras 72b y 72c que son las correlaciones de pixeles adyacentes en diagonal y horizontal resptectivamente, como la distrución es una disperciońen la gráfica no existe correlación de pixeles adyacentes.



(a) Correlación vertical. (b) Correlación diagonal. (c) Correlación horizontal.

Figura 72: Correlación de píxeles adyacentes del criptograma Cameraman usando el sistema Logístic 2D.

La Figura 73 muestra gráficamente la correlación de pẋeles adyacentes del criptograma 49c utilizando el CPRGB con el sistema Logístic 2D. La Figura 73a ilustra la correlación de píxeles adyacentes de forma vertica, donde la distribución de los puntos en la gráfica es disperza por lo tanto no existe una correlación de pẋeles adyacentes. El mismo comportamiento aparece en las Figuras 73b y 73c que son las correlaciones de pixeles adyacentes en diagonal y horizontal resptectivamente, como la distrución es una disperciońen la gráfica no existe correlación de pixeles adyacentes.



(a) Correlación vertical. (b) Correlación diagonal. (c) Correlación horizontal.

Figura 73: Correlación de píxeles adyacentes del criptograma Mandril usando el sistema Logístic 2D.

Apéndice D

Configuración del chip de reconocimiento de voz

En la D.1 se muestra la configuración del Chip de reconocimiento de voz y en la D.2 se ilustran otras vocez grabadas en el chip.

D.1 Proceso de grabación de la voz en el microcontrolador SPCE061A

El microcontrolador SPCE061A puede reconocer la voz. Recibe su configura
ción a través del puerto serial. El modulo almacena hasta 15 instrucciones de voz, divididos en 3 grupos, cada voz está vinculada a una salida en texto o hexadecimal dependiendo del modo de trabajo, modo corto o modo extendido. Bajo un ambiente controlado tiene un 99% de reconocimiento. La tabla siguiente muestra el resultado de la grabación de voz en modo corto.

Grupo 1		**Grupo 2**	
Voz (Señal analógica)	Salida (Señal binaria)	Voz (Señal analógica)	Salida (Señal binaria)
Audio	$0x11$	Audio	$0x21$
Audio	$0x12$	Audio	$0x22$
Audio	$0x13$	Audio	$0x23$
Audio	$0x14$	Audio	$0x24$
Audio	$0x15$	Audio	$0x25$
Grupo 3			
Voz (Señal analógica)	Salida (Señal binaria)		
Audio	$0x11$		
Audio	$0x32$		
Audio	$0x33$		
Audio	$0x34$		
Audio	$0x35$		

Tabla XXVI: Banco de voz del chip SPCE061A.

La interface del chip es de 5V TTL. El formato de los datos es 8 bits de datos, no paridad, 1 bit de paro. El baudio es de 9600 por default pero puede ser cambiado.

El fabricante del chip recomienda el software AccessPort para la configuración y proceso de grabación de las diferentes voces.

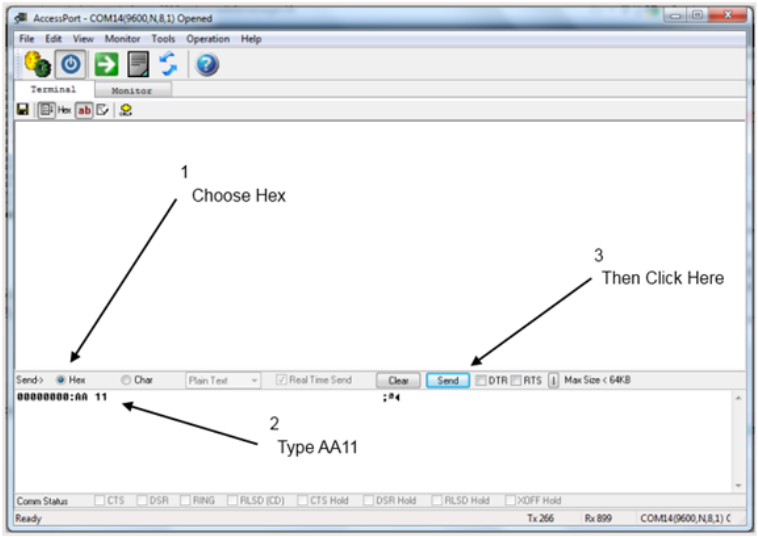


Figura 74: Interfaz gráfica del software AccessPort.

El chip tiene 38 instrucciones para el proceso de grabado de la voz. Si se utiliza $0xAA$ después $0x11$, se empieza a grabar las 5 palabras del grupo 1, de forma similar al usar $0x12$ o $0x13$ despues de $0xAA$ se inicia a grabar los grupos 2 o 3 respectivamente.

D.2 Otras voces grabadas en el chip

En la parte superior de la figura 75 se ilustra la grafíca de la señal de audio y la parte inferior muestra el espectro en frecuencias. “Asteroide”es el audio que se ha grabado en el banco 1 del CRV.

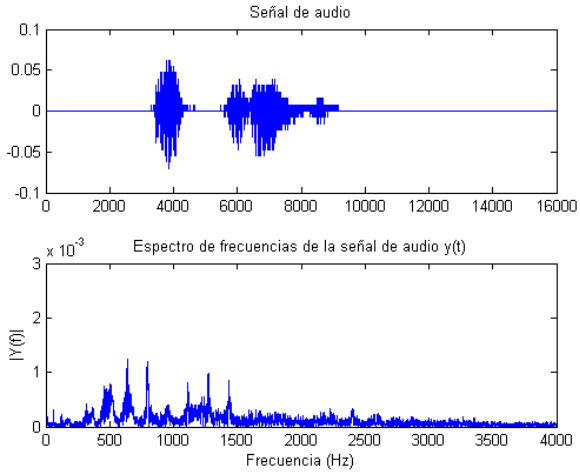


Figura 75: Espectro en frecuencia de la señal de voz del audio “Asteroide”.

La figura 76 ilustra grafíca de la señal de audio y en la parte inferior se observa el espectro en frecuencias. “Cueva”es el audio que se ha grabado en el banco 1 del CRV.

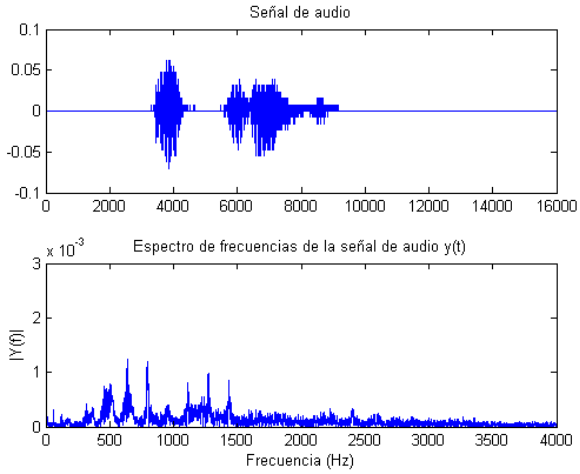


Figura 76: Espectro en frecuencia de la señal de voz del audio “Cueva”.

En la parte superior de la figura 77 se ilustra la grafíca de la señal de audio y la

parte inferior muestra el espectro en frecuencias. “Femur”es el audio que se ha grabado en el banco 1 del CRV.

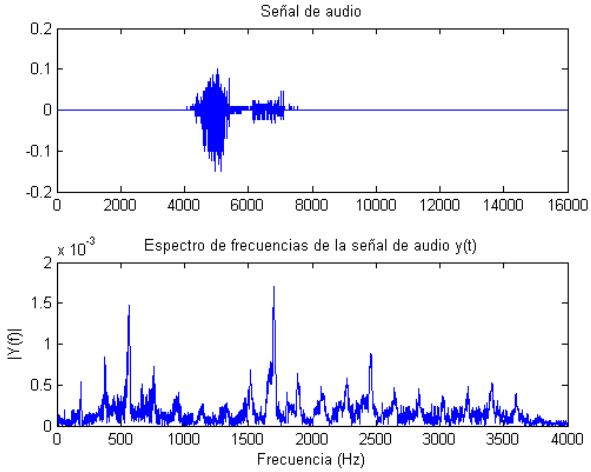


Figura 77: Espectro en frecuencia de la señal de voz del audio “Femur”.

En la parte superior de la figura 78 se ilustra la grafíca de la señal de audio y la parte inferior muestra el espectro en frecuencia. “Galaxia”es el audio que se ha grabado en el banco 1 del CRV.

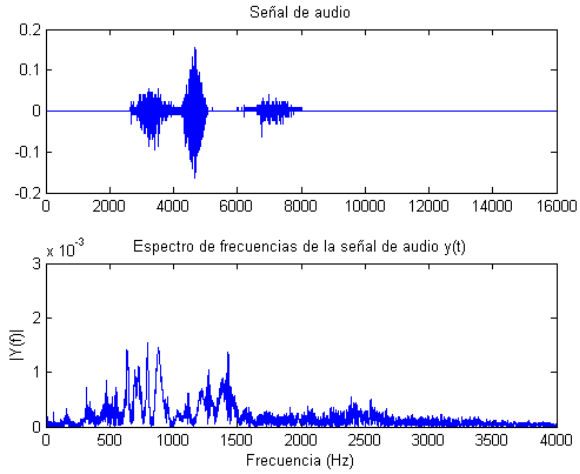


Figura 78: Espectro en frecuencia de la señal de voz del audio “Galaxia”.

En la parte superior de la figura 79 se ilustra la grafíca de la señal de audio y la parte inferior muestra el espectro en frecuencia. “Higado”es el audio que se ha grabado en el banco 2 del CRV.

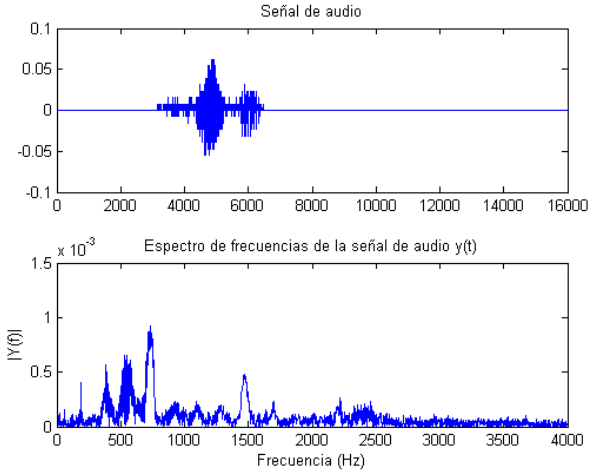


Figura 79: Espectro en frecuencia de la señal de voz del audio “Hígado”.

En la parte superior de la figura 80 se ilustra la grafíca de la señal de audio y la parte inferior muestra el espectro en frecuencia. “Hongo”es el audio que se ha grabado en el banco 2 del CRV.

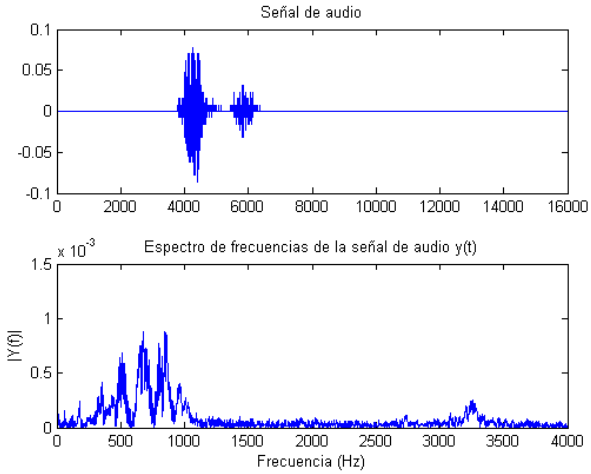


Figura 80: Espectro en frecuencia de la señal de voz del audio “Hongo”.

En la parte superior de la figura 81 se ilustra la grafíca de la señal de audio y la parte inferior muestra el espectro en frecuencia. “Luna”es el audio que se ha grabado en el banco 2 del CRV.

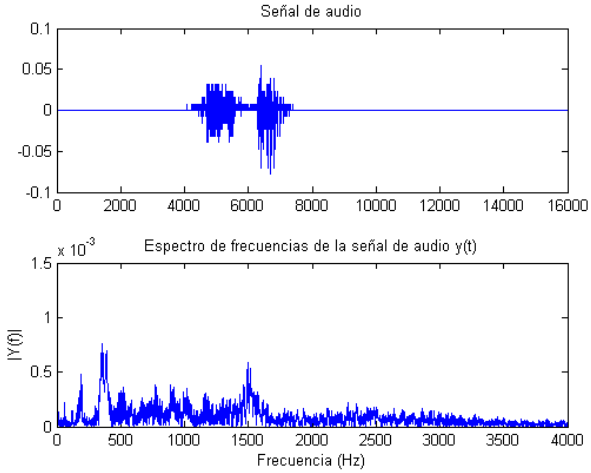


Figura 81: Espectro en frecuencia de la señal de voz del audio “Luna”.

En la parte superior de la figura 82 se ilustra la grafíca de la señal de audio y la parte inferior muestra el espectro en frecuencia. “Montaña”es el audio que se ha grabado en el banco 2 del CRV.

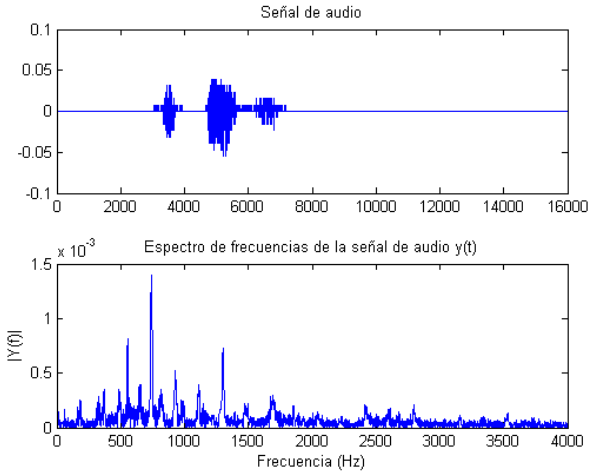


Figura 82: Espectro en frecuencia de la señal de voz del audio “Montaña”.

En la parte superior de la figura 83 se ilustra la grafíca de la señal de audio y la parte inferior muestra el espectro en frecuencia. “Piel”es el audio que se ha grabado en el banco 2 del CRV.

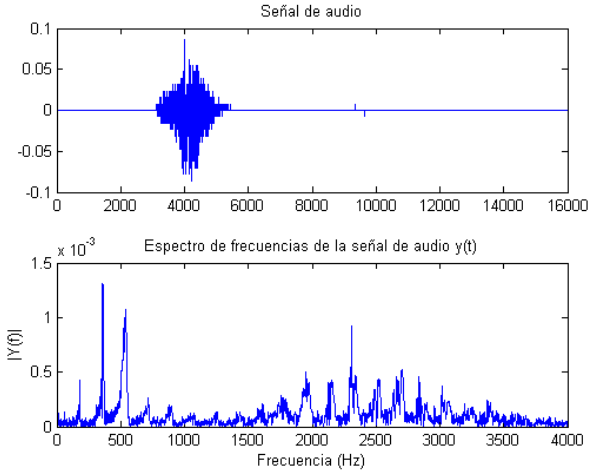


Figura 83: Espectro en frecuencia de la señal de voz del audio “Piel”.

En la parte superior de la figura 84 se ilustra la grafíca de la señal de audio y la parte inferior muestra el espectro en frecuencia. “Pulmon”es el audio que se ha grabado en el banco 3 del CRV.

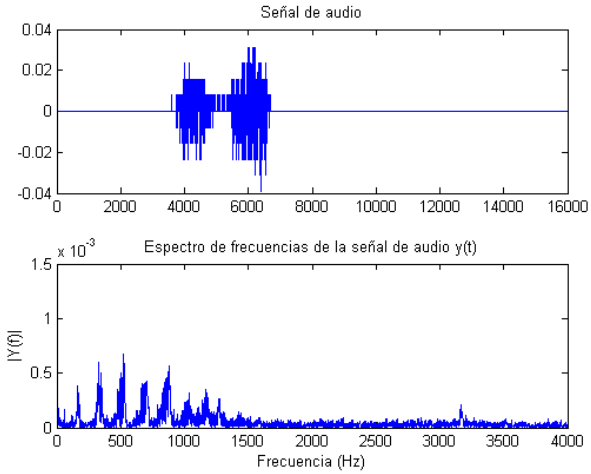


Figura 84: Espectro en frecuencia de la señal de voz del audio “Pulmón”.

En la parte superior de la figura 85 se ilustra la grafíca de la señal de audio y la parte inferior muestra el espectro en frecuencia. "Saturno"es el audio que se ha grabado en el banco 3 del CRV.

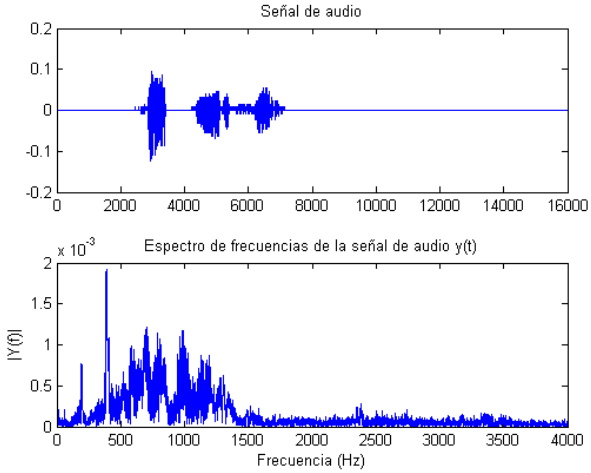


Figura 85: Espectro en frecuencia de la señal de voz del audio "Saturno".

En la parte superior de la figura 86 se ilustra la grafíca de la señal de audio y la parte inferior muestra el espectro en frecuencia. "Telescopio"es el audio que se ha grabado en el banco 3 del CRV.

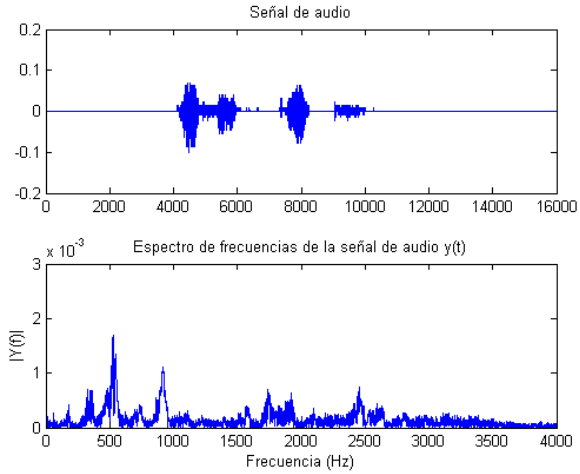


Figura 86: Espectro en frecuencia de la señal de voz del audio "Telescopio".

En la parte superior de la figura 87 se ilustra la grafíca de la señal de audio y la parte inferior muestra el espectro en frecuencia. “Vaso”es el audio que se ha grabado en el banco 3 del CRV.

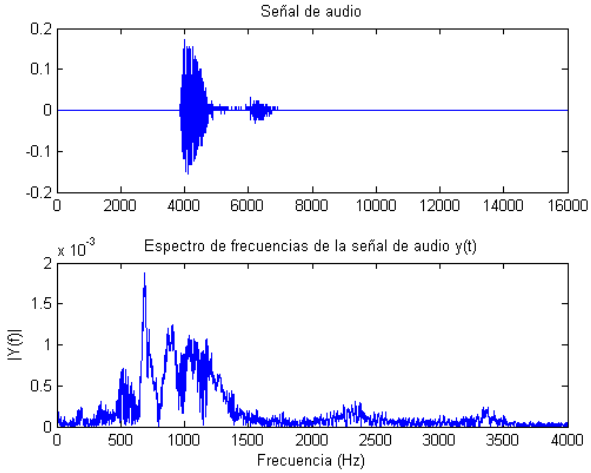


Figura 87: Espectro en frecuencia de la señal de voz del audio “Vaso”.

En la parte superior de la Figura 88 se ilustra la grafíca de la señal de audio y la parte inferior muestra el espectro en frecuencia. “Venado”es el audio que se ha grabado en el banco 3 del CRV.

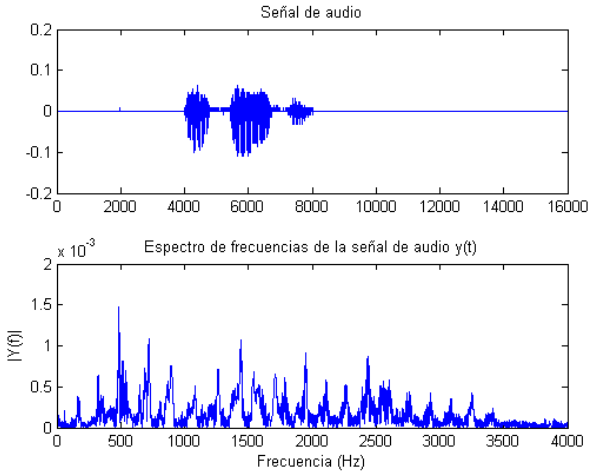


Figura 88: Espectro en frecuencia de la señal de voz del audio “Venado”.