UNIVERSIDAD AMERICANA DE ACAPULCO

EXCELENCIA PARA EL DESARROLLO

FACULTAD DE INGENIERÍA EN COMPUTACIÓN
INCORPORADA A LA UNIVERSIDAD NACIONAL
AUTONÓMA DE MÉXICO
CLAVE DE INCORPORACIÓN 8852-16

PROPUESTA DE DISEÑO DE ENCRIPTACIÓN EN UN FPGA

T E S I S

QUE PARA OBTENER EL TÍTULO DE:
INGENIERO EN COMPUTACIÓN

PRESENTA:
RAMÍREZ HERNÁNDEZ CHRISTIAN ALBERTO

DIRECTOR DE TESIS
ING. FRANCISCO NARCÉS DÁVILA ZURITA

ACAPULCO, GUERRERO, MAYO DEL 2013.

AGRADECIMIENTOS

A **Dios**.

Por haberme permitido nacer bajo el seno de una familia maravillosa.

A mis **Padres**.

Gracias por su apoyo incondicional a lo largo de toda mi vida, porque gracias a ustedes nunca me ha faltado nada.

A la **Universidad Americana de Acapulco**.

Por haberme permitido vivir momentos de crecimiento dentro de ella.

A la **Facultad de Ingeniería en Computación**.

Por el apoyo diario brindado de una manera incondicional.

A los **Maestros**

Porque sin ustedes nunca habría llegado a esta meta.

DEDICATORIA

A mis **Padres**:

Heriberto Ramírez Gómez^{+}, gracias por enseñarme a tomar la vida con alegría y sabiduría, siempre vivirás en mi corazón; e

Inés Hernández López, eres y serás una luz en mi camino, la fuerza que me impulsa a seguir con la frente en alto.

A mi **Esposa**:

Eloísa Mercedes Vivas Villasana, por ser novia, esposa pero sobre todo amiga, por brindarme tu apoyo en los momentos más difíciles de la carrera.

A mis **Hijos**:

Por ser el pretexto perfecto que me ha impulsado a terminar una faceta pendiente en mi vida; se han convertido en la razón de mi existencia.

A mi **Familia**:

A todos y cada uno de ustedes que la conforman, gracias por apoyarme en los momentos más significativos de mi vida.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

PREFACIO

La **encriptación** permite que dos partes, *emisor* y *receptor*, puedan intercambiar *información* sin que una tercera parte no autorizada, a pesar de que intercepte los datos, sea capaz de comprender el *mensaje* original.

El *algoritmo* propuesto para el **proceso criptográfico** (***encriptar***/***desencriptar***), permite realizar de manera confiable el procesamiento simultáneo de información. Tomando ventaja de esta característica se establece que el uso de técnicas de *procesamiento paralelo*, permite establecer mejoras en el desempeño del algoritmo.

Se ha desarrollado en este trabajo una *arquitectura*[1] para la encriptación de *voz*, empleando un **Lenguaje Descriptivo de Circuitos Integrados de Alta Velocidad** (**VHDL**, por sus siglas en inglés), utilizando la herramienta de edición, compilación y simulación *Active-VHDL™* de *Aldec*®.

[1] *Arquitectura*, en el sentido computacional.

ORGANIZACIÓN DE LA TESIS

La estructura de la tesis se encuentra dividida en los siguientes capítulos:

- *Capítulo 1.-* Se describe el planteamiento del problema, la justificación, el objetivo general y la hipótesis de este trabajo de investigación.

- *Capítulo 2.-* Ahí se expondrán los conceptos principales que serán utilizados en el desarrollo del sistema; ofreciendo un panorama general de las diferentes técnicas de encriptación.

- *Capítulo 3.-* En este capítulo se muestra el funcionamiento del algoritmo seleccionado, así como las pruebas y resultados.

- *Capítulo 4.-* En éste, se describe la arquitectura propuesta para la encriptación de la voz y se discuten los resultados de la simulación.

- *Capítulo 5.-* En esta parte final, se presentan las conclusiones y se proponen los trabajos futuros como alternativas de mejora.

CAPÍTULO 1. INTRODUCCIÓN

1.1. Planteamiento del Problema

El envío y recepción de la información en tiempo real, se ha vuelto una necesidad imperante en la vida cotidiana, tal es el caso de las transferencias bancarias, comercio electrónico, las telecomunicaciones e incluso las conversaciones telefónicas (por ejemplo, cada vez que alguien levanta el auricular para realizar una llamada telefónica, está en riesgo de revelar información sensible acerca de su personalidad, economía, gustos, hábitos sociales, residencia; los cuales pueden ser maliciosamente recolectados y utilizados por terceros, en perjuicio del usuario).Todo esto conlleva a la necesidad de crear sistemas de encriptación lo suficientemente rápidos y seguros para poder dar soporte a este tipo de transmisiones.

Existen diversos algoritmos capaces de cifrar información; el más utilizado durante poco más de tres décadas ha sido la Encriptación Estándar de Datos o DES (de las siglas de la frase en inglés: *Data EncryptionStandar*)[6], el cual presenta una clave de cifrado de 56 bits; este tamaño de clave en la actualidad no es suficiente para garantizar la seguridad de los sistemas de información (en Julio de 1998, el DES cracker de la EFF "*ElectronicFrontierFoundation*" conocido como *Deep Crack* rompe una clave DES en 56 horas)[24]; la necesidad de garantizar una mayor seguridad en los sistemas de información dan la pauta para que en el año 2000; surja un nuevo estándar de encriptación, el AES [2] el cual utiliza claves de cifrado que van desde 128 bits hasta 256 bits asiéndolo con esto prácticamente imposible de descifrar[25].

Dado que la transmisión de voz con calidad telefónica responden a frecuencias máximas de 4KHz, los sistemas de encriptación de voz necesitan muestrear a 8Khz como mínimo, lo que supone 64Kbps (kilo bits por segundo) tomando muestras de 8 bits. (Teorema de Nyquist)[26].

1.2. Justificación

La encriptación de voz no sólo requiere de algoritmos seguros, sino que también demanda velocidades altas de procesamiento. El uso de una computadora es poco fiable en la seguridad, y no resuelve el problema de la velocidad, lo que podría ocasionar pérdida en la información y tener una mala calidad del audio.

El uso del hardware especializado y reconfigurable para aplicaciones en tiempo real es una opción viable. La existencia de un sistema de procesamiento rápido y portable,acelera el trabajo en aquellas investigaciones que requieren seguridad y transmisión en tiempo real.

Actualmente existen diversos sistemas capaces de encriptar a velocidades altas; sin embargo, el costo de adquisición y el mantenimiento de la mayoría de estos sistemas es sumamente elevado, por lo que representa un problema bastante significativo para la mayoría de los usuarios.

El desarrollo del hardware de encriptación se puede basar en dos tipos de arquitecturas: los Circuitos Integrados de Aplicación Especifica (ASIC, por sus siglas en inglés) y los Arreglos de Compuertas Programables (FPGA, por sus siglas en inglés) [1]. Tanto los ASIC como los FPGA pueden hacer uso completo del procesamiento paralelo y del pipeline. Sin embargo, los ASIC requieren de herramientas de diseño y simulación complejas, además de que el diseño debe ser preciso, ya que una vez fabricados no se pueden corregir los errores. En contraparte, la utilización de un FPGA, nos lleva a correr menos riesgos, gracias a que su diseño es modificable debido a que se configura a partir de un archivo y cuenta con sistemas de pruebas para su verificación.

1.3. Objetivo General

El objetivo de la tesis es el de proponer un diseño basado en un FPGA para la encriptación de datos en tiempo real, utilizando las ventajas que ofrece el manejar una arquitectura paralela reconfigurable (FPGA) y con arreglos pipeline[4] en los accesos a memoria, para así obtener un rendimiento eficaz en el proceso del algoritmo de encriptación AES.

El procesamiento a bajo nivel tiene como objetivo el analizar cada uno de los bloques de información con el menor número de operaciones posibles (aritméticas, direccionamiento a memoria, ciclos de transferencia).

1.4. Hipótesis

Actualmente existen diversos sistemas capaces de encriptar a velocidades altas, sin embargo, el costo de adquisición y el mantenimiento de la mayoría de estos sistemas es sumamente elevado; por lo que representa un problema para la mayoría de los usuarios.

La implementación hardware para la encriptación de datos permitirá la independencia de una computadora y un tamaño compacto, para ser empleado en aplicaciones gubernamentales, como llamadas telefónicas, minimizando el riesgo de comprensión de la información por alguna entidad no autorizada.

CAPÍTULO 2. CRIPTOGRAFÍA

2.1. Seguridad en los sistemas de comunicación

La intimidad es un derecho del individuo[27]; que con los medios de comunicación tradicionales, tales como el correo postal, correo certificado, los apartados de correos entre otros, están más que garantizados. Sin embargo, el uso generalizado de los sistemas electrónicos, pone en gran riesgo la intimidad y el anonimato de los usuarios.

Desde la antigüedad, la necesidad de seguridad ha traído consigo el desarrollo de infinidad de sistemas y métodos de encriptación; y con el tiempo, las técnicas se fueron perfeccionando. Con la llegada de las computadoras se aumenta la posibilidad de realizar sistemas de encriptación más complicados, pero también aumenta el riesgo de romperlos con mayor facilidad.

Hoy en día, la urgencia de transferirla información da la pauta para que los gobiernos, las empresas y la

ciudadanía en general busquen por cualquier medio mantener lejos de "*miradas indiscretas*", la información que día a día transmiten a través de canales inseguros.

La transmisión de la información en tiempo real se ha vuelto una necesidad imperante en la vida cotidiana, tal es el caso de las conversaciones telefónicas, transferencias bancarias, comercio electrónico, telecomunicaciones, autenticación, etc. Esto trae la necesidad de crear sistemas de encriptación lo suficientemente rápidos y seguros para poder soportar este tipo de comunicaciones.

2.2. Técnicas criptográficas

En la actualidad existen tres tipos de algoritmos de encriptación: los que están basados en llaves secretas –simétricos, los que trabajan bajo la influencia de las llaves públicas –asimétricos[16],[19] y los basados en los Algoritmos HASH. La principal diferencia entre estas técnicas de encriptación, es que la llave de encriptación utilizada por los algoritmos simétricos es exactamente igual a la que se utiliza para recuperar el texto original (desencriptación). Por otro lado, los algoritmos de encriptación asimétricos utilizan una llave distinta para cada proceso y la llave de desencriptado no puede ser por ningún motivo calculada de la llave de encriptado; mientras que el Algoritmo HASH efectúa un cálculo matemático sobre los datos que constituyen el documento y da como resultado un número único llamado MAC. Un mismo documento dará siempre un mismo MAC.

2.2.1 Criptografía Simétrica

La criptografía simétrica (Figura 2.1.) es el sistema de encriptación más antiguo en la historia de la humanidad[16], [21]. Este tipo de cifrado se ha utilizado desde los tiempos de Julio César hasta nuestros días.

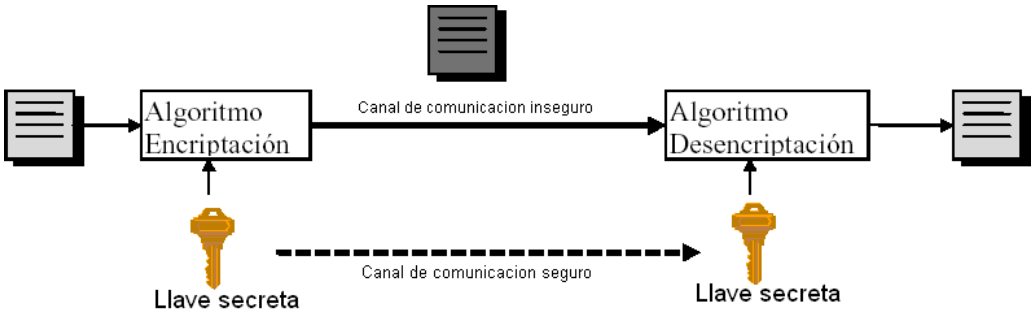

Figura 2.1. Criptografía Simétrica

La seguridad está basada en la privacidad de la llave secreta, llamada simétrica porque es la misma para el emisor y el receptor. El emisor del mensaje genera una llave y después la transmite mediante un canal seguro a todos los usuarios autorizados a recibir el mensaje.

La distribución de la llave es un problema para los sistemas simétricos. Este inconveniente se ha resuelto gracias a los sistemas asimétricos utilizados únicamente para la distribución de la llave simétrica.

2.2.2 Criptografía Asimétrica

Los algoritmos asimétricos (Figura 2.2.) permiten que la llave de encriptación sea del dominio público, permitiendo así a cualquier individuo (emisor) que conozca la llave poder encriptar la información[16], [21]. Sin embargo, solo el destinatario (receptor) puede desencriptar el mensaje ya que este es el único que conoce la llave de desencriptado. La llave de encriptado es conocida como llave pública, mientras que la llave utilizada para la desencriptación es conocida como llave privada o llave secreta.

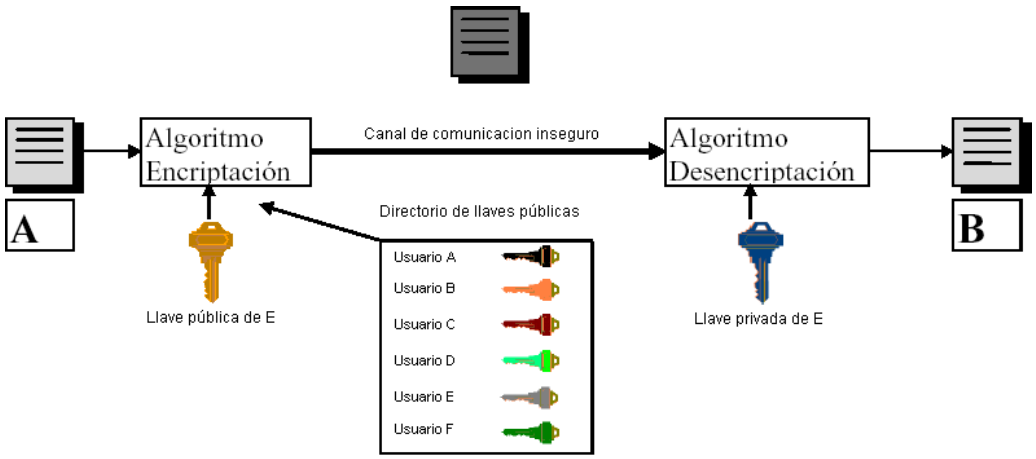

Figura 2.2. Criptografía Asimétrica

A la encriptación asimétrica también se le conoce como criptografía de llave pública, por el tipo de llaves que

maneja. Existen diferentes sistemas de llave pública, pero el más difundido y el que se considera un estándar es el RSA (Rivest, Shamir, Adleman)[16].

2.3. Algoritmos simétricos más utilizados

2.3.1 DES (Data Encryption Standard)

En 1971 IBM desarrolló un algoritmo de encriptación simétrico basado en la aplicación de todas las teorías existentes sobre criptografía. Se llamó LUCIFER y funcionaba con llaves simétricas de 128 bits. Fue vendido en exclusividad a la empresa de seguros Lloyd's.

En 1973 el Buró Nacional de Estándares (NBS, por sus siglas en ingles), hoy en día conocido como el Instituto Nacional de Estándares y Tecnología (NIST, por sus siglas en inglés) convocó a un concurso para elegir un estándar de encriptación para la seguridad de los documentos oficiales. El concurso fue ganado en 1977 por los inventores del LUCIFER con una versión mejorada, el DES [11].

El DES encripta bloques de 64 bits con una llave de encriptación de igual tamaño, de los cuales 8 son de paridad. El sistema de desencriptación es muy similar, con lo que facilita su implementación en hardware y software.

Según [6], [13] el DES tiene cuatro modos distintos de operación, para poder implementarse:

ECB (ElectronicCodebookMode) para mensajes cortos de menos de 64 bits.

CBC (Cipher Block ChainingMode) para mensajes largos.

CFB (Cipher Block Feedback) para cifrar bit por bit o byte por byte.

OFB (Output FeedbackMode) el mismo uso del CFB pero evitando propagación de error.

2.3.2 TDES (Triple DES)

El tamaño de llave del DES se ha considerado un problema ya que con máquinas potentes trabajando en paralelo a través de una red, se podría romper este algoritmo[21], [8]. Para evitar este problema y seguir utilizando el DES existe un sistema basado en tres iteraciones del algoritmo, llamado **triple DES** o **TDES**0, que utiliza una llave de 128 bits.

En la siguiente ecuación se describe la formula.

$$E2_K[D_K[E1_K[TEXTO\ ]]] = TE_K[TEXTO\ ]$$

Primero se usa una llave de Encriptación $E1_K$ junto con el $TEXTO$ original, el resultado de este proceso es Desencriptado con la llave D_K diferente de $E1_K$ y por último la información obtenida de la desencriptación es encriptada con $E2_K$ diferente de $E1_K$ y D_K

2.3.3 IDEA (International Data Encryption Algorithm)

En 1990*Lai*y *Massey*del Instituto Federal Suizo de Tecnologíadesarrollaron un nuevo sistema, el Algoritmo Internacional de Encriptación de Datos (IDEA, por sus siglas en inglés) [10].

Este algoritmo está libre de restricciones y permisos nacionales y es de libre distribución por Internet.

El IDEA maneja bloques de texto de 64 bits y una llave de 128 bits. Puede funcionar con los 4 modos: **ECB, CBC, CFB y OFB**.

2.3.4 RC5(Cifrado de Rivest)

El RC5 [15] fue inventado por Rivest (RSA), proviene del RC4, y es propiedad de *RSA Data Security Inc*. La empresa *Netscape* utiliza la versión RC5 para su sistema de seguridad SSL, por ese motivo se ha extendido mucho. Como la mayoría de nuevos algoritmos, permite diferentes longitudes de llave. Fuera de USA sólo se puede exportar la versión con llave de 56 bits.

Debido a su juventud, su seguridad no está muy probada frente acriptoanalistas. En 1996 una universidad francesa **rompió la versión del RC4 con llave de 40 bits,** utilizada en Europa, **en 8 día**s, esto ha hecho dudar de su seguridad.

El algoritmo de encriptación RC5, funciona como un generador de números aleatorios que se suman al texto mediante una OR-EXCLUSIVA.

Se pueden configurar muchos parámetros: número de iteraciones, longitud de llave y tamaño de bloque. Esto permite adaptarse a las necesidades de velocidad/seguridad de cada aplicación.

2.3.5 Rijndael (AES, Advanced Encryption Standard)

Este algoritmo fue el **ganador del concurso lanzado por el NIST**[16]para conseguir el nuevo sistema de encriptación simétrico. El rijndaeles ahora conocido como **AES**[16], [3]. Está diseñado por Joan Daemen y Vincent Rijmen.

El diseño se realizó buscando **3 objetivo**s:

i) Máxima **resistencia a ataque**s.

ii) **Velocidad y código compacto** para varias plataformas.

iii) **Simplicidad** de diseño.

El AES permite tres tamaños de bloques y tres tamaños de llaves, estos pueden ser: 128, 192 o 256 bits. El tamaño del bloque no condiciona el tamaño de la llave.

El algoritmo **utiliza 3 transformaciones llamadas capas**:

- **Capa de mezcla lineal**. Crea la difusión del algoritmo, como las permutaciones.
- **Capa no lineal**. Como las cajas *S* del DES. Realiza la confusión, o sea, las sustituciones.
- **Capa de suma de llave. ExOR.** Para mezclar la llave transformada con los resultados de las iteraciones.

En cada iteración se realiza una aplicación de cada capa. El número de iteraciones depende de la longitud de la llave o el bloque. Así siempre se toma la longitud mayor entre la llave o el bloque y se aplica la siguiente fórmula:

- 128 bits son 10 iteraciones.
- 192 bits son 12 iteraciones.
- 256 bits son 14 iteraciones.

El algoritmo resultante es **rápido y sencillo** de implementar en cualquier plataforma. Además, **su seguridad está probada** por los múltiples test realizados en el concurso del NIST[16].

2.4. Algoritmos asimétricos más utilizados

2.4.1 RSA (Rivest, Shamir and Adleman)

El algoritmo RSA[18] utiliza dos llaves públicas (dos números grandes elegidos por un programa), *e* y *n*, y una llave privada (un número grande consecuencia de los dos anteriores) *d*. Este proceso se realiza de la siguiente forma:

- Se buscan dos números primos grandes (entre 100 y 300 dígitos): *p* y *q*.
- Se calcula $\phi = (p-1)*(q-1)$ y $n = p*q$.
- Se busca *e* como un número sin múltiplos comunes a ϕ.
- Se calcula $d = e^{-1} \bmod \phi$.
- Se hacen públicas las llaves *n* y *e*, se guarda *d* como llave privada y se destruye *p*, *q* y ϕ.

Para encriptar el RSA utiliza $C = M^{e} \bmod n$ donde *M* es el mensaje y el proceso de desencriptación es llevado procesado por $M := C^{d} \bmod n$.

2.4.2 RABIN (Rivest, Shamir and Adleman)

Por ser de tipo asimétrico, es necesario poseer dos claves, una pública y una privada, que se utilizarán de la misma forma que las del RSA; su cálculo se realiza de la siguiente forma:

- Se eligen dos números primos "*p*" y "*q*", ambos congruentes con $3(mod4)$ cuyos dos últimos bits sean 1 y se calcula el producto de estos.

$$n = pq$$

- De estos productos realizados los números "*p*" y "*q*" serán tomados como la clave privada, mientras que su producto "*n*" será la pública.

- Con las claves previamente elegidas es posible iniciar la codificación utilizando la ecuación:

$$C = m^2 mod n$$

- Mientras que para decodificar el mensaje, debe resolverse el conjunto siguiente de ecuaciones:

$$m_1 = c^{\frac{p+1}{4}} \, mod \, p$$

$$m_2 = \left(p - c^{\frac{p+1}{4}}\right) \, mod \, p$$

$$m_3 = c^{\frac{q+1}{4}} \, mod \, q$$

$$m_4 = \left(q - c^{\frac{q+1}{4}}\right) \, mod \, q$$

- Los valores de m_1 a m_4 no son el mensaje. Debido a la naturaleza matemática del algoritmo de RABIN surgen 4 posibles mensajes, los cuales pueden estar constituidos así:

$$m_a = (am_1 + bm_3)\ mod\ n$$

$$m_b = (am_1 + bm_4)\ mod\ n$$

$$m_c = (am_3 + bm_3)\ mod\ n$$

$$m_d = (am_2 + bm_4)\ mod\ n$$

- Donde *a* y *b* deben calcularse de tal manera que cumplan la siguiente ecuación:

$$a = q(q^{-1}\ mod\ p)$$

$$b = p(p^{-1}\ mod\ q)$$

- Al descifrar los cuatro mensajes, no existe ninguna forma de saber cuál es el original, puesto que los cuatro mensajes m_a, m_b, m_cy m_d cumplen con la solución del problema por tanto el emisor del mensaje debe colocar alguna señal que le indique al receptor cual es el mensaje verdadero.

CAPÍTULO 3. ADVANCEDENCRYPTION STANDARD

3.1. Funcionamiento

El Estándar de Encriptación Avanzado (AES) es un algoritmo criptográfico que puede ser utilizado para proteger información electrónica. El algoritmo AES es un bloque de código simétrico que puede cifrar y descifrar información. La encriptación convierte los datos en una forma inteligible llamada *ciphertext* (*texto cifrado*); descifrando el *ciphertext* convierte los datos de regreso a su forma original, llamada *plaintext* (*texto plano*).

La longitud de los bloques de datos y la llave de encriptación pueden ser de 128, 192 o 256 bits. El bloque de datos y la llave de encriptación son considerados como un arreglo de bytes de 4 renglones por *n* columnas (4=128 bits, 6=192 bits, 8 = 256 bits).

3.2. El Encriptador

El algoritmo de encriptación se divide en cuatro partes como se ilustra en la(Figura 3.1):

i) Transformación de bytes (S-box),

ii) Corrimiento de renglones (ShiftRow),

iii) Manipulación de columnas (MixColumn),

iv) Mezcla de la llave de encriptación (Key XOR).

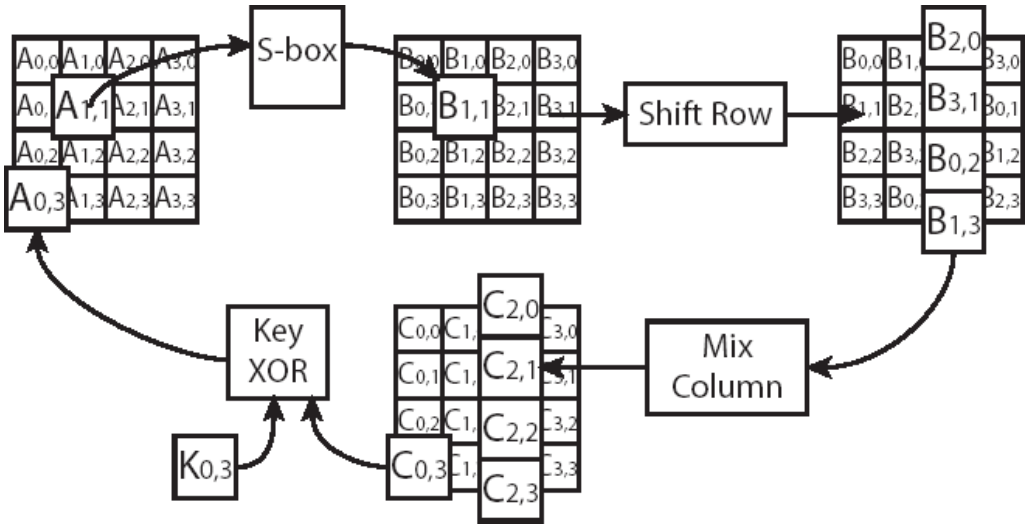

Figura 3.1. Algoritmo de Encriptación

3.2.1 Transformación de Bytes

La primera fase de la encriptación (*i*) es una sustitución de bytes no lineal(Figura 3.2) que actúa sobre cada byte del estado individualmente para producir un nuevo valor del byte utilizando la tabla de sustitución Sbox (Tabla 3-1).

hex	y															
	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

x

Tabla 3-1 Tabla de Sustitución (Sbox)

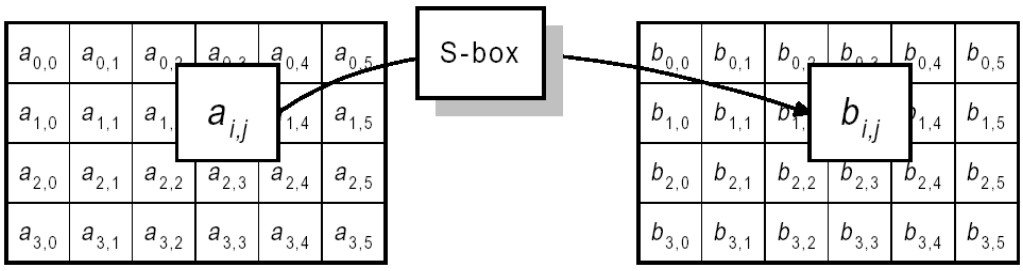

Figura 3.2. Sustitución de Bytes

Aplicación de S-box a cada uno de los bytes del estado

3.2.2 Corrimiento de Renglones

En el segundo paso (*ii*) se realiza un corrimiento de renglones (Figura 3.3) los bytes en los últimos tres renglones del estado son cíclicamente desplazados de acuerdo a la *Ecuación 1*.

$$S'_{r,c} = S_{r,(c+h(r,Nb))\bmod Nb}$$
$$0 < r < 4 \quad \text{y} \quad 0 \le c < Nb$$

Ecuación 1

r Renglón
c Columna
$S_{r,c}$ Estado a transformar
$S'_{r,c}$ Estado resultante
Nb Numero de Palabras
$h(r, Nb)$ Desplazamiento

donde el valor de $h(r, Nb)$ depende del número del renglón, *r*.

h[r, Nb]		row (r)		
		1	2	3
Nb	4	1	2	3
	6	1	2	3
	8	1	3	4

Tabla 3-2 Corrimiento para diferentes longitudes de bloque

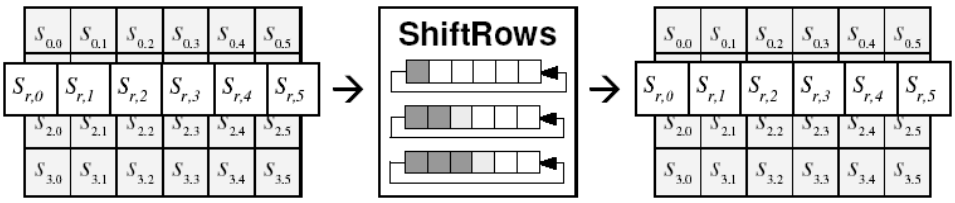

Figura 3.3. Corrimiento de Renglones

3.2.3 Manipulación de Columnas

Después de realizado el corrimiento de renglones se procede a transformar independientemente cada una de las columnas del estado (iii) y tratarlas como un polinomio de cuatro términos (Figura 3.4). Esta transformación está dada por la Ecuación 2:

$$\begin{bmatrix} \dot{S}_{0,c} \\ \dot{S}_{1,c} \\ \dot{S}_{2,c} \\ \dot{S}_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix}$$

$$0 \leq c < Nb$$

Ecuación 2

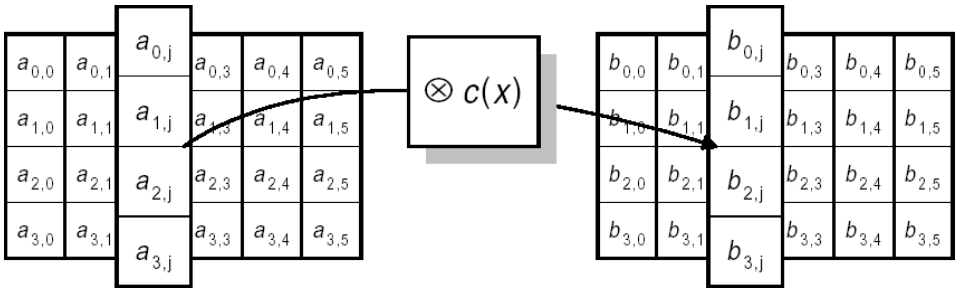

Figura 3.4. Manipulación de Columnas

3.2.4 Mezcla de la Llave

En la última fase (iv) la llave de encriptación es mezclada con el estado resultante de la transformación anterior (iii) por medio de una operación XOR. Cada bloque de llaves consiste de Nb palabras del vector de llaves. Cada una de estas Nb palabras son mezcladas con las columnas del estado, tal que:

$$\left[S'_{0,c}, S'_{1,c}, S'_{2,c}, S'_{3,c}\right] = \left[S_{0,c}, S_{1,c}, S_{2,c}, S_{3,c}\right] \oplus \left[w_{round*Nb+c}\right]$$
$$0 \le c < Nb$$

Ecuación 3

Donde $[w_i]$ es el vector de llaves en el rango $0 \le round \le Nr$ y $round$ es el número de ciclos requeridos para la encriptación.

El vector de llaves es obtenido al expandir la llave original. La expansión de la llave genera un total de $Nb(Nr+1)$ palabras. El algoritmo requiere inicialmente de un conjunto de Nb palabras, y cada uno de los Nr ciclos requiere Nb palabras de la llave principal. El vector resultante consiste de un arreglo lineal de 4 bytes, denotado por $[w_i]$, en el rango $0 \le i < Nb(Nr+1)$.

3.3. El Desencriptador

El algoritmo de desencriptación es implementado en forma inversa para producir una desencriptación directa.

La transformación de bytes es similar a la utilizada en la encriptación, con la única diferencia de la tabla de transformación que en este caso será InvSbox (Tabla 3-3).

hex		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Tabla 3-3Tabla de Sustitución (InvSbox)

En el corrimiento de renglones se procede de acuerdo a la Ecuación 4

$$S'_{r,(c+shift\,(r,Nb))\bmod Nb} = S_{r,c}$$
$$0 < r < 4 \quad \text{y} \quad 0 \le c < Nb$$

Ecuación 4

La transformación de columnas está dada por

$$\begin{bmatrix} \dot{S}_{0,c} \\ \dot{S}_{1,c} \\ \dot{S}_{2,c} \\ \dot{S}_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix}$$
$$0 \le c < Nb$$

Ecuación 5

Y por último la mezcla de la llave es exactamente la misma utilizada en la encriptación, utilizando el vector de llaves en forma inversa.

3.4. Pruebas y resultados

Se presenta un ejemplo del vector de pruebas (Figura 3.5) utilizado para la fase de encriptación del AES, mostrando la transformación del estado a través de cada uno de los módulos del algoritmo.

Round Number	Start of Round	After SubBytes	After ShiftRows	After MixColumns		Round Key Value	
input	32 88 31 e0 43 5a 31 37 f6 30 98 07 a8 8d a2 34				⊕	2b 28 ab 09 7e ae f7 cf 15 d2 15 4f 16 a6 88 3c	=
1	19 a0 9a e9 3d f4 c6 f8 e3 e2 8d 48 be 2b 2a 08	d4 e0 b8 1e 27 bf b4 41 11 98 5d 52 Ae f1 e5 30	d4 e0 b8 1e bf b4 41 27 5d 52 11 98 30 ae f1 e5	04 e0 48 28 66 cb f8 06 81 19 d3 26 e5 9a 7a 4c	⊕	a0 88 23 2a fa 54 a3 6c fe 2c 39 76 17 b1 39 05	=
2	a4 68 6b 02 9c 9f 5b 6a 7f 35 ea 50 f2 2b 43 49	49 45 7f 77 de db 39 02 d2 96 87 53 89 f1 1a 3b	49 45 7f 77 db 39 02 de 87 53 d2 96 3b 89 f1 1a	58 1b db 1b 4d 4b e7 6b ca 5a ca b0 f1 ac a8 e5	⊕	f2 7a 59 73 c2 96 35 59 95 b9 80 f6 f2 43 7a 7f	=
3	aa 61 82 68 8f dd d2 32 5f e3 4a 46 03 ef d2 9a	ac ef 13 45 73 c1 b5 23 cf 11 d6 5a 7b df b5 b8	ac ef 13 45 c1 b5 23 73 d6 5a cf 11 b8 7b df b5	75 20 53 bb ec 0b c0 25 09 63 cf d0 93 33 7c dc	⊕	3d 47 1e 6d 80 16 23 7a 47 fe 7e 88 7d 3e 44 3b	=
4	48 67 4d d6 6c 1d e3 5f 4e 9d b1 58 ee 0d 38 e7	52 85 e3 f6 50 a4 11 cf 2f 5e c8 6a 28 d7 07 94	52 85 e3 f6 a4 11 cf 50 c8 6a 2f 5e 94 28 d7 07	0f 60 6f 5e d6 31 c0 b3 da 38 10 13 a9 bf 6b 01	⊕	ef a8 b6 db 44 52 71 0b a5 5b 25 ad 41 7f 3b 00	=
5	e0 c8 d9 85 92 63 b1 b8 7f 63 35 be e8 c0 50 01	e1 e8 35 97 4f fb c8 6c d2 fb 96 ae 9b ba 53 7c	e1 e8 35 97 fb c8 6c 4f 96 ae d2 fb 7c 9b ba 53	25 bd b6 4c d1 11 3a 4c a9 d1 33 c0 ad 68 8e b0	⊕	d4 7c ca 11 d1 83 f2 f9 c6 9d b8 15 f8 87 bc bc	=

Figura 3.5. Modelo de Prueba (1)

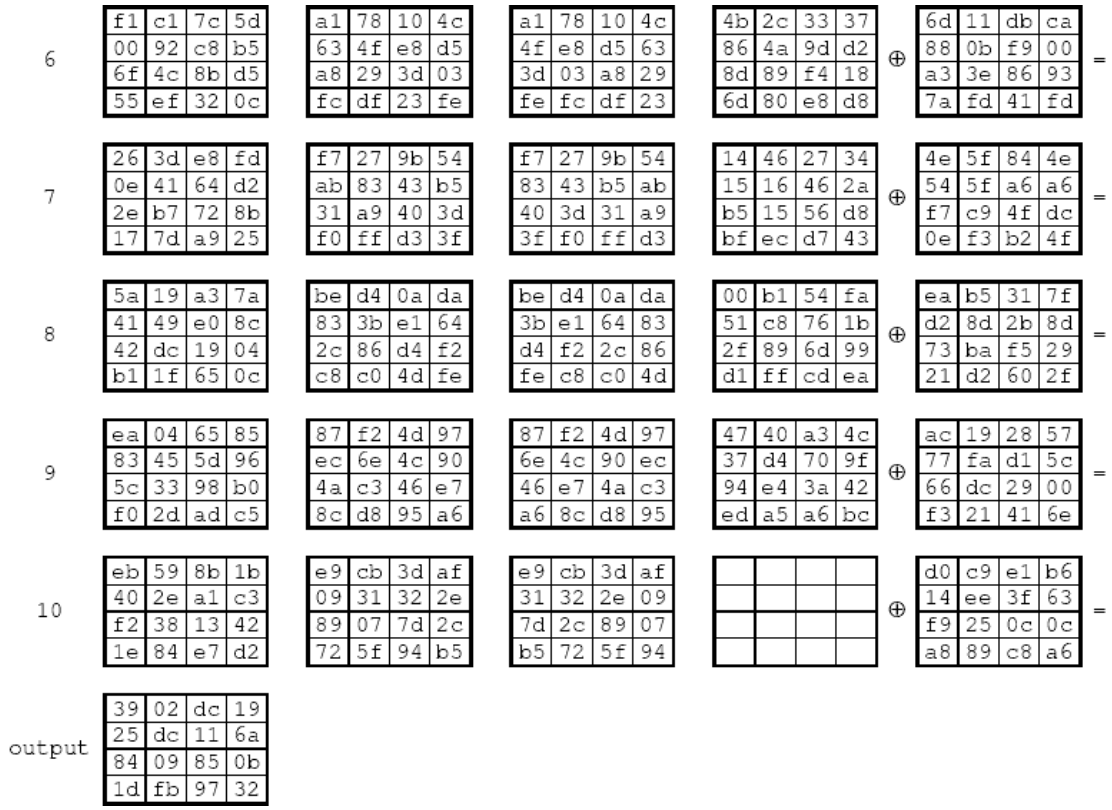

Figura 3.6. Modelo de Prueba (2)

CAPÍTULO 4. ARQUITECTURA PROPUESTA

4.1. Introducción

En los capítulos 2 y 3, se mencionó la importancia de mantener la información lejos de personas ajenas al emisor y receptor, para aligerar el costo computacional del proceso de búsqueda y sobre todo para garantizar el control de acceso, se planteó que el algoritmo a utilizar fuera el AES en una arquitectura hardware dentro de un FPGA con las ventajas que este conlleva. A continuación se presenta la propuesta de la arquitectura, así como también la adaptación del algoritmo a la arquitectura.

4.2. Especificaciones de Diseño.

La arquitectura está basada en FPGAs, tomando ventajas de la disponibilidad de registros para implementar módulos compactos y eficientes.

La arquitectura está diseñada de tal forma que minimiza el número de ciclos requeridos en la encriptación de datos y está basada en módulos pipeline, trabajando con vectores de 32 bits, con operaciones paralelas internas y vectores de 8 bits, con el fin de mejorar el rendimiento.

El diseño se realizó utilizando el lenguaje de descripción hardware VHDL [23] y se sintetizó para un FPGA Spartan II [20], utilizando como base la tarjeta de desarrollo Strathnuey de Nallatech[14].

4.3. Bloques Funcionales.

La arquitectura propuesta, que está esquematizada en la Figura 4.1.) consta principalmente de un multiplexor (Mux),una ROM de transformación de datos (SubWord), unaRAM (Key Memory), el generador del vector de llaves (Key Schedule), un módulo de encriptación (Cipher).

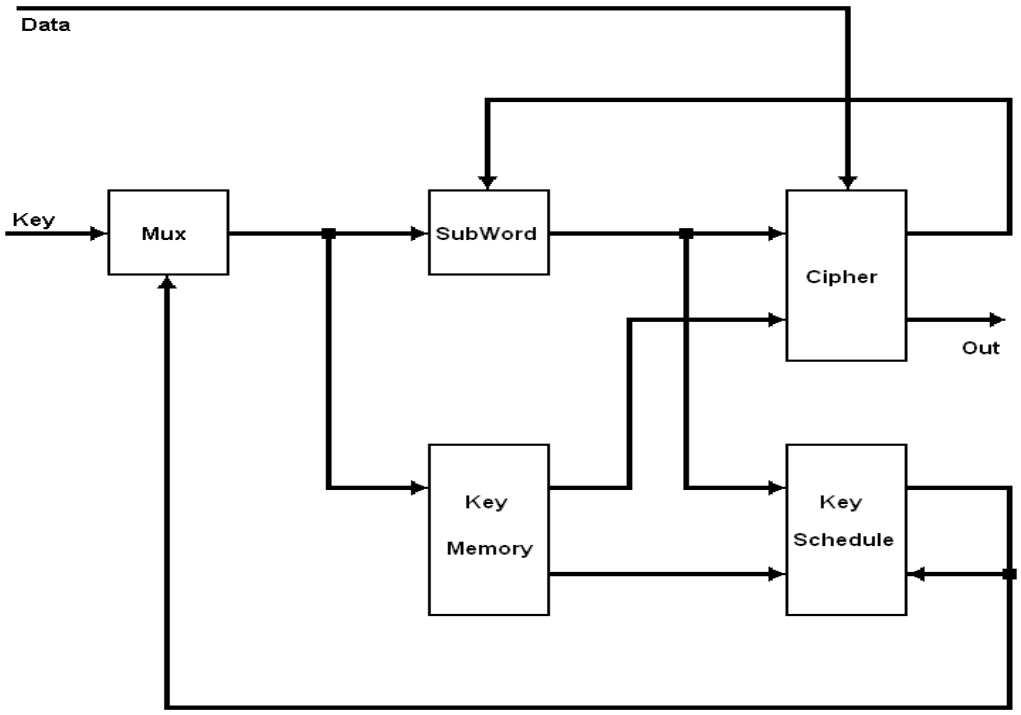

Figura 4.1. Esquema General de Encriptación

La arquitectura toma ventaja de la disponibilidad de registros en el FPGA, para implementar módulos eficientes y compactos para procesar la encriptación y la generación del vector de llaves.

4.3.1 Generador de Llaves (Key Shedule)

El módulo generador de llaves (Figura 4.2.) está compuesto por dos multiplexores, un módulo XOR, una constante de ceros y un arreglo de constantes (Rcon).

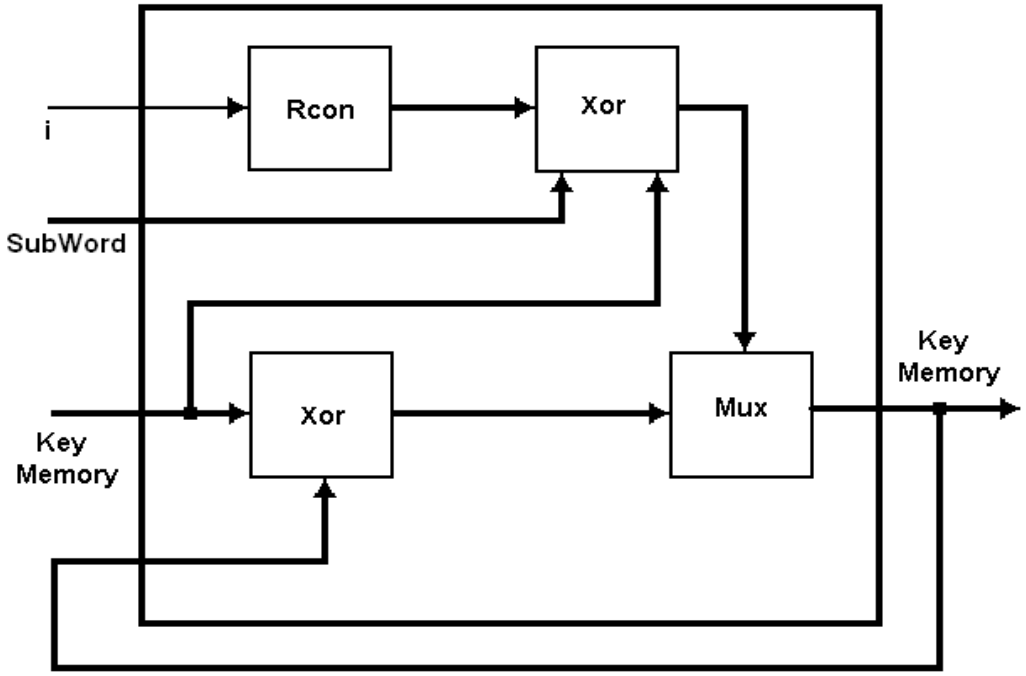

Figura 4.2. Generador del Vector de Llaves

El módulo es alimentado mezclándose cada 4 ciclos con Rcon y SubBytes (sustitución de bytes).

Una vez que este proceso ha registrado sus resultados en el banco de memoria, se procede a la fase de encriptación y/o desencriptación

El módulo de encriptación (Figura 4.3.) está compuesto por un banco de memoria, un multiplexor, un módulo XOR, el módulo SubBytes (sustitución de bytes) y el módulo de transformación de columnas (MixColumns).

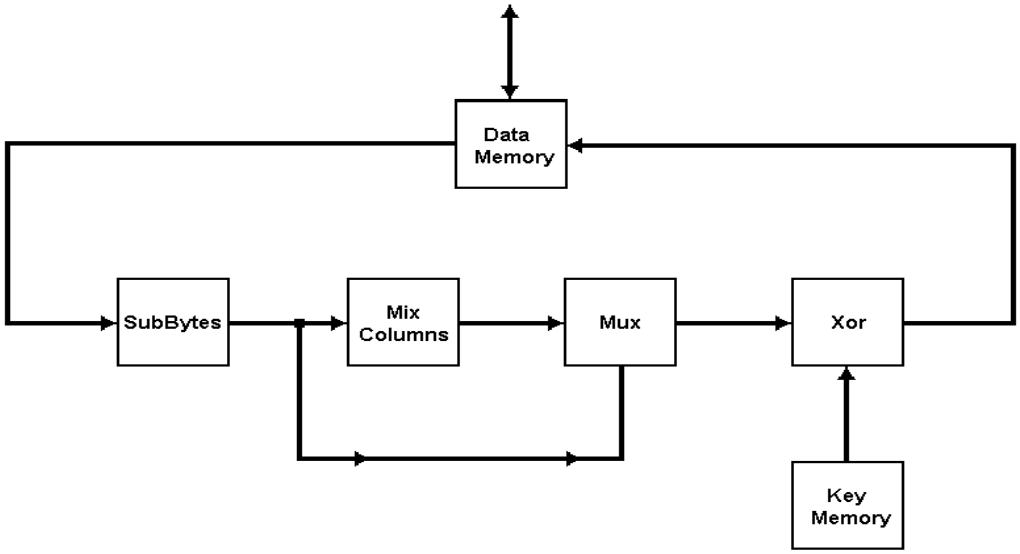

Figura 4.3. Módulo de Encriptación.

Como se observa en la Figura 4.3, el módulo de corrimiento de renglones es omitido, ya que un módulo de control es el encargado de direccionar al banco de memoria de tal manera que los datos son obtenidos como si se hubiera realizado un corrimiento con anterioridad.

CIFRADOR

CONCEPTO	CANTIDAD	PORCENTAJE
External GCLKIOBs	01 de 4	25%
External IOBs	97 de 176	55%
LOCed External IOBs	0 de 97	0%
BLOCKRAMs	04 de 12	33%
SLICEs	438 de 1728	25%
GCLKs	1 de 4	25%

Tabla 4-1 Resultados del Cifrador

Generador del Vector de Llaves.

CONCEPTO	CANTIDAD	PORCENTAJE
ExternalGCLKIOBs	1 de 4	25%
ExternalIOBs	33 de 176	18%
LOCedExternal IOBs	0 de 33	0%
BLOCKRAMs	2 de 12	16%
SLICEs	364 de 1728	21%
GCLKs	1 de 4	25%

Tabla 4-2 Resultados del Generador de Llaves

CAPÍTULO 5. CONCLUSIONES

5.1. Conclusiones

Sobre los resultados presentados de la arquitectura implementada, se observa que supera el requisito de procesar 64Kbps y permite la transmisión de datos en tiempo real (voz). El utilizar diseños basados en FPGAs para aplicaciones de procesamiento de datos, resulta en un diseño fácil de integrar, además de reducir tiempos y costos.

El diseño de la arquitectura es compacto y no requiere de otros dispositivos (como memoria externa), solamente necesita un FUGA Spartan II u otro FUGA que contenga bloques de memoria interna con la cantidad de espacio suficiente para almacenar las tablas de conversión (s-boxes). Por lo tanto es adecuado para sistemas móviles y portátiles, donde se requiere un bajo consumo de potencia y poco espacio físico.

5.2. Trabajos Futuros

- En un futuro se planea implementar un sistema de digitalización de voz. Se tomará algún sistema existente en el mercado.

- Otra propuesta a futuro es la de incorporar un algoritmo asimétrico, para la distribución de la clave del AES, puede ser el RSA, o alguno basado en curvas elípticas.

REFERENCIAS

[1] Brown, S. AND ROSE, Jonathan. “Architecture of FPGAs and CPLDs: A Tutorial”, IEEE Design and Test of Computer, Vol. 13, No. 2, pp. 42-57, 1996.

[2] Federal Information Processing Standards Publication, FIPS PUB 197, “Advanced Encryption Standard (AES)”, 26-Noviembre-2001. http://csrc.nist.gov/publications/fips/

[3] J. Daemen and V. Rijmen, Answer to "new observations on Rijndael’, AES Forum comment, August 11, 2000. http://www.esat.kuleuven.ac.be/~rijmen/rijndael/.

[4] J. Hennessy y D. Patterson; "Computer Architecture The Hardware-Software Interface"; McGraw Hill, 1996

[5] M. Pons; “Cristología”; Escuela Universitaria Politécnica de Matarao, Departamento de Telecomunicaciones.

[6] Federal Information Processing Standards Publication, FIPS PUB 46-3, “Data Encryption Standard (DES)”, 25-Octubre-1999. http://csrc.nist.gov/publications/fips/

[7] Federal Information Processing Standards Publication; “DES modes of operation” December 2, 1980

[8] P. van Oorschot, M. Wiener; “A Known-plaintext attack on two-kwy triple encryption”; Advances in Cryptology EUROCRYPT ’90, LNCS 473, pp 318-325, 1991

[9] Federal Information Processing Standards Publication; “TDES”; 1999.

[10] X. Lai, J.R. Massey, “A proposal for a new block encryption standard”, Advances in Cryptology EUROCRYPT’90, LNCS 473, pp 389-404, 1991

[11] J. Sánchez Arriazu; “Descripción del Algoritmo DES (Data Encryption Standard)”; 1999.

[12] DIMETM Motherboard with Xilinx VirtexTM II or SpartanTM II FPGA.

[13] “Information processing – Modes of operation for a 64-bit block cipher algorithm”, 1997.

[14] Nallatech, Desarrollador de sistemas de computo reconfigurable basados en FPGA´s. www.nallatech.com/solutions/applications/app_areas.asp

[15] R.L. Rivest; “The RC5 Encryption Algorithm”; MIT Laboratory for Computer Science. rivest@theory.lcd.mit.edu http://citeseer.nj.nec.com/63393.html

[16] J. Nechvatal, E. Barker, L. Bassham, W. Burr, M. Dworking, J. Foti, E. Roback “Report on the Development of the Advanced Encryption Standard (AES)”; Computer Security Division, National Institute of Standards and Technology; October 2, 2000.

[17] J. Daemen (Proton World Int.l), V. Rijmen (KatholiekeUniversiteit Leuven, ESAT-COSIC); “The Rijndael Block Cipher, AES Proposal”

[18] S. Burnett, S. Paine; “RSA Security´s Oficial Guide to CRYPTOGRAPHY”; McGraw Hill, 2001.

[19] P. Caballero; “SEGURIDAD INFORMATICA, Técnicas criptográficas”; ALFAOMEGA, 1997.

[20] Xilinx; “Spartan-II 2.5V FPGA Family, Functional Description”, Preliminary Product Specification; DS001-2 (v2.1) March 5, 2001.www.xilinx.com

[21] A. Fuster Sabater, D. de la Guisa Martines, L. Hernández Encinas; “Técnicas criptográficas de protección de datos”; ALFAOMEGA, 2ª edición, 2001.

[22] R. Ibarra, M. Serrano, C. Calixto; “Teoría de la Información y Encriptamiento de Datos”; IPN, 1ª edición, 2001.

[23] Peter J. Ashenden; “The VHDL Cookbook, First Edition”; Dept. Computer Science, University of AdelaideSouth Australia; July, 1990.

[24] EFF DES cracker; From Wikipedia, the free encyclopedia http://en.wikipedia.org/wiki/EFF_DES_cracker

[25] Vincent Rijmen (2010). “Practical-Titled Attack on AES-128 Using Chosen-Text Relations” (http:// http://eprint.iacr.org/2010/337.pdf)

[26] Curso de Ingeniería y Telemática http://www.securisite.org/biblioteca/seguridad/Ingenieria%20Telematica-curso/TELE_15-Redes%20Multimedia/ampli_2.pdf

[27] CONSTITUCION POLITICA DE LOS ESTADOS UNIDOS MEXICANOS; TITULO PRIMERO, CAPITULO I DE LOS DERECHOS HUMANOS Y SUS GARANTÍAS (Reformada la denominación por decreto publicado en el Diario Oficial de la Federación el 10 de Junio de 2011) Artículo 16