DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELÉCTRÓNICA Y TELECOMUNICACIONES

TRABAJO DE TITULACIÓN, PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

TEMA: ESTUDIO DE LAS TÉCNICAS DE CRIPTOGRAFÍA PARA LA IMPLEMENTACIÓN DE UN MÓDULO DE CIFRADO EN UNA RADIO DEFINIDA POR SOFTWARE MULTIBANDA DESARROLLADO EN UNA TARJETA USRP N200/210.

AUTORES: CHACHALO ARBOLEDA, EVELYN FABIOLA

PAREDES VALENCIA, PABLO ADRIANO

DIRECTOR: ING. BERNAL OÑATE, CARLOS PÁUL

SANGOLQUÍ

2018

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

CERTIFICACIÓN

Certifico que el trabajo de titulación, "**ESTUDIO DE LAS TÉCNICAS DE CRIPTOGRAFÍA PARA LA IMPLEMENTACIÓN DE UN MÓDULO DE CIFRADO EN UNA RADIO DEFINIDA POR SOFTWARE MULTIBANDA DESARROLLADO EN UNA TARJETA USRP N200/210**" realizada por la señorita **CHACHALO ARBOLEDA, EVELYN FABIOLA** y el señor **PAREDES VALENCIA, PABLO ADRIANO**, el mismo que ha sido revisado en su totalidad, analizado por la herramienta de verificación de similitud de contenido; por lo tanto cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

Sangolquí, 05 de diciembre de 2018

Ing. Carlos Paúl Bernal Oñate

CC. 1709775637

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORÍA DE RESPONSABILIDAD

Nosotros, **CHACHALO ARBOLEDA, EVELYN FABIOLA** y **PAREDES VALENCIA, PABLO ADRIANO** declaramos que el contenido, ideas y criterios del trabajo de titulación: **"ESTUDIO DE LAS TÉCNICAS DE CRIPTOGRAFÍA PARA LA IMPLEMENTACIÓN DE UN MÓDULO DE CIFRADO EN UNA RADIO DEFINIDA POR SOFTWARE MULTIBANDA DESARROLLADO EN UNA TARJETA USRP N200/210"** es de nuestra autoría y responsabilidad, cumpliendo con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Consecuentemente el contenido de la investigación mencionada es veraz.

Sangolquí, 05 de diciembre de 2018

EVELYN FABIOLA CHACHALO ARBOLEDA

C.C.: 172603686-4

PABLO ADRIANO PAREDES VALENCIA

C.C.: 1803272408-8

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORIZACIÓN

Nosotros, **CHACHALO ARBOLEDA, EVELYN FABIOLA** y **PAREDES VALENCIA, PABLO ADRIANO** autorizamos a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación "**ESTUDIO DE LAS TÉCNICAS DE CRIPTOGRAFÍA PARA LA IMPLEMENTACIÓN DE UN MÓDULO DE CIFRADO EN UNA RADIO DEFINIDA POR SOFTWARE MULTIBANDA DESARROLLADO EN UNA TARJETA USRP N200/210"** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de nuestra autoría y responsabilidad.

Sangolquí, 05 de diciembre de 2018

EVELYN FABIOLA CHACHALO ARBOLEDA

C.C.: 172603686-4

PABLO ADRIANO PAREDES VALENCIA

C.C.: 1803272408-8

DEDICATORIA

Este logro se lo dedico a Dios que me ha dado muchas bendiciones, fortaleza, sabiduría y salud para alcanzar este objetivo.

A mi hermano Marlon quien ha sido un soporte fundamental para mi vida y mis estudios, te debo mucho, eres el mayor ejemplo de superación.

A mi mami Elena porque aún en medio de todo me ha brindado su ayuda.

A Kathy y a mis sobrinas Valentina y Camila por su paciencia y apoyo aún en los días malos.

A mis amigos Mabe y Fer porque han estado ahí en los momentos donde quería votar la toalla, dándome ánimos, Mabe que ha sido esa amiga que me ha ayudado a tener un respiro en medio del camino.

A Chío y mis amigos de Espe al Máximo porque me enseñan cuan real es Dios.

Evelyn Fabiola Chachalo Arboleda

AGRADECIMIENTO

Agradezco a Dios por cada persona que ha puesto en mi vida, por cada detalle de su amor que me han permitido alcanzar este objetivo, el primero de muchos.

Al Ing. Paúl Bernal por el apoyo, el tiempo, la guía y la paciencia, en el desarrollo de este proyecto.

A Marlon un hermano como los que ya no se hacen, eres la persona más valiente, esforzada y trabajadora que conozco, tú nunca te das por vencido. Gracias por abrirme las puertas de tu casa, de tu familia, me has ayudado en mis estudios, me has tenido paciencia, has compartido los buenos y malos momentos. Gracias de todo corazón por cuidarme y creer en mí. Admiro tu tenacidad para afrontar cada situación. Te quiero mucho.

A mamá por darme la vida, por los valores que inculco en mí y porque de una forma u otra me ha apoyado.

A mi familia, Kathy, Valentina y Camila por su apoyo, por los momentos compartidos.

A mis amigos cada uno formo parte de un pedacito de mi vida, gracias por compartir esta etapa, esas amanecidas estudiando o haciendo proyectos.

A Mabe por su amistad, su tiempo al escucharme y ayudarme a no rendirme, tu apoyo ha sido importante, hemos compartido desde diversión hasta las amanecidas haciendo proyectos.

A Chío por su amistad, por ser una guía en mi camino espiritual, por su tiempo y sus consejos.

Evelyn Fabiola Chachalo Arboleda

DEDICATORIA

El presente trabajo lo quiero dedicar a mis abuelitos: Jaime, Eusebio, Teresa, Rebeca, pilares fundamentales de una gran familia y aunque ya no están con nosotros sus enseñanzas aún perduran en cada miembro de la familia. A ustedes abuelitos con mucho cariño.

Pablo Adriano Paredes Valencia

AGRADECIMIENTO

Quiero empezar agradeciendo a Dios por guiarme a lo largo de mi vida, ser el apoyo y fortaleza en aquellos momentos de dificultad.

Siempre me van a faltar palabras para agradecer a mis Padres: Jorge y Cecilia por eso solo puedo decirles gracias por ser los mejores Padres sin ustedes nada hubiera sido posible. Me gustaría extender mis agradecimientos a mi familia Hermanos, Tíos, Primos y Sobrinos por sus consejos, risas y cada momento ameno que compartimos juntos. Un agradecimiento especial es para Lizeth por su cariño y a toda su familia por su aprecio y siempre hacerme sentir como un miembro más de ellos.

De igual manera mis agradecimientos a mis profesores en especial al Ing. Paúl Bernal, Ing. Diego Villamarin, Ing. Darwin Aguilar e Ing. Jorge Alvares quienes con la enseñanza de sus valiosos conocimientos hicieron que pueda crecer día a día como profesional, gracias a cada una de ustedes por su paciencia, dedicación, apoyo incondicional y amistad.

Finalmente quiero expresar mi agradecimiento a todos los amigos y futuros colegas por los buenos y difíciles momento vividos dentro de la universidad, por su ayuda desinteresada y buena voluntad.

Pablo Adriano Paredes Valencia

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

Las radios definidas por *software* son una tecnología que actualmente se encuentra en desarrollo para el perfeccionamiento de las comunicaciones. Una de las partes más importantes de las comunicaciones es ocultar información para que no pueda ser intervenida e interpretada por personas ajenas a la misma, tomando en cuenta que se vive una era donde las comunicaciones digitales van reemplazando a grandes pasos los medios analógicos. La presente investigación pretende implementar algoritmos de cifrado en una radio definida por *software*, analizando diferentes sistemas criptográficos con el fin de proveer mayor seguridad a la radio. Utilizando los campos de Galois para varios medios, se desarrollaron códigos de cifrado que se adaptan a las características de seguridad deseadas para la implementación en las radios. Posteriormente se cuantificaron las pérdidas de datos en los diferentes modelos de radio simulados en función de la relación señal a ruido y la tasa de bit erróneos, además se usó como medida objetiva MBSD (del inglés *Modified Bark Spectral Distortion*). Finalmente se propone un modelo de radio que trabaja con las tarjetas USRP 2920 basado en el programa de LabVIEW®, con la aspiración de reemplazar los medios actuales de comunicación de las Fuerzas Armadas proveyéndolos de radios más seguras, más ligeras y con mayores prestaciones.

PALABRAS CLAVES:

- **CIFRADO**
- **ALGORITMOS**
- **CRIPTOGRAFÍA**

ABSTRACT

The Software Defined Radios are a technology that is currently in development for the improvement of communications. One of the most important parts of communications is to hide information so that it can not be intervened and interpreted by people outside of it, taking into account that there is an era where digital communications are replacing analogical media in a big way. The present research aims to implement encryption algorithms in a software-defined radio, analyzing different cryptographic systems in order to provide greater security to the radio. Using the Galois fields for various media, encryption codes were developed that adapt to the desired security features for the implementation in the radios. Subsequently, the data losses in the different simulated radio models were quantified as a function of the erroneous signal-to-noise ratio and bit rate, in addition it was used as objective measure MBSD (Modified Bark Spectral Distortion). Finally, a radio model that works with the USRP 2920 cards based on the LabVIEW® program is proposed, with the aspiration to replace the current means of communication of the Armed Forces by providing them with safer, lighter and higher performance radios.

KEYWORDS:

- **ENCRYPTION**
- **ALGORITHMS**
- **CRYPTOGRAPHY**

CAPÍTULO I

1 INTRODUCCIÓN

1.1 Formulación del problema y antecedentes

En la actualidad los medios digitales de comunicación dejaron de ser el privilegio de pocos para convertirse en la necesidad de grandes conglomerados, encontrando en el mercado cada vez mejores prestaciones de servicios a precios muy reducidos.

Sin embargo, para este gran desborde de información es fundamental contar con un sistema de seguridad efectivo que permita transmisiones sin vulnerabilidades, a fin de que personas ajenas a la comunicación no puedan interceptar y descifrar información que cause daño a la integridad personal y material de los usuarios.

Por décadas las fuerzas militares han buscado proteger sus mensajes en sus diferentes medios de comunicación, para mantener un comando y un control adecuado en todas sus operaciones, es ahí donde se manifiesta la importancia del cifrado de datos en las comunicaciones militares.

La recepción oportuna de la información, así como la seguridad de la misma, son fundamentales en la toma de decisiones para poder llevar a cabo coordinaciones verticales u horizontales entre los mandos, de esto dependerá el éxito o fracaso de una misión y es por ello que se realza la importancia de una comunicación segura, donde agentes externos al sistema de comunicación no puedan interpretar las intenciones del comandante.

Los ejércitos a lo largo de la historia han tenido importante trascendencia en el desarrollo tecnológico de un país, partiendo de una preponderante necesidad por comunicar las órdenes a las

tropas e impedir que los contrarios puedan hacer lo mismo, es así como formalmente nace el cifrado y con este el espionaje y contraespionaje de la información en medios electrónicos.

Los diferentes tipos de cifrados han venido evolucionando conforme se desarrollaba la invención de máquinas mecánicas y eléctricas complejas, desde la elaboración de la máquina de rotores enigma que reemplazó el cifrado a papel y lápiz, hasta la introducción de la computadora que permite elaborar sistemas de gran complejidad. Estos avances tecnológicos también han permitido diseñar nuevos modelos para descifrar información; esta es la razón para que en la actualidad se encuentren criptografías más robustas mediante el uso de cifrados analógicos y cifrados digitales.

A finales del siglo XIX el italiano G. Marconi fue uno de los más destacados impulsores de una forma de comunicarse prodigiosa: la radiotransmisión a larga distancia, así como la radiotelegrafía. En manos de los militares la radio fue un poderoso medio de transmisión, pero los mensajes podían caer también en manos enemigas, por lo que era necesario enviarlos cifrados (Meavilla, 2004).

La era de la criptografía moderna comienza con Claude Shannon, considerado como el padre de la criptografía matemática. En 1949 publicó el artículo *Communication Theory of Secrecy Systems* en *Bell System Technical Journal*, y poco después el libro *Mathematical Theory of Communication*, con Warren Weaver. Sin embargo, la NSA (del inglés *National Security Agency*) de Estados Unidos acaparó y bloqueó casi totalmente la publicación de cualquier avance en el campo de la criptografía desde principios de los 50 hasta mediados de los 70 (Tomé, 2015).

Sin embargo, luego de este periodo se publicó el primer cifrado aprobado por una agencia nacional estadounidense, conocido como *Data Encryption Standard* (DES) el cual fue reemplazado de forma oficial por AES (del inglés *Advanced Encryption Standard*) en el 2001, lo cual impulsó el estudio de la criptografía dentro de la Academia. *Whitfield Diffie* y *Martin Hellman* (1976) presentaron un avance muy importante denominado intercambio de claves *Diffie-Hellman,* mediante el cual se logró distribuir las claves criptográficas de forma segura y sentar las bases para el desarrollo de los algoritmos de cifrado asimétrico (Tomé, 2015).

En la actualidad, un procedimiento muy empleado a nivel internacional es la criptofonía (criptografía + telefonía) o sacrofonía, la cual consiste en proteger la información de las conversaciones telefónicas mediante el enmascaramiento de las voces de los interlocutores. Este mecanismo se basa en el uso de un modulador cuya función consiste en convertir señales en sonidos sin sentido y un demodulador que realiza la función inversa recuperando la señal original, de tal manera que únicamente el receptor autorizado es capaz de acceder a la información (Díaz, 2000).

1.2 Justificación e Importancia

El propósito de este proyecto es desarrollar códigos de cifrado que sirvan como sistema de seguridad durante la transmisión de información por radio minimizando al máximo costos de producción y adquisición, con la finalidad de establecer un medio de comunicación seguro en el Ejército Ecuatoriano. La presente investigación pretende llevar a cabo la implementación inicial de un código de cifrado en una USRP (del inglés *Universal Software Radio Peripheral*), para que aporte a una fabricación sólida y robusta de radios, con el objetivo de que únicamente los

miembros de dicha institución puedan acceder a información confidencial sobre la seguridad del Estado, evitando la manipulación de estos datos por personal externo a las Fuerzas Armadas.

1.3 Alcance del Proyecto

El presente proyecto pretende investigar e implementar códigos de cifrado de información en dos SDR (del inglés *Software Defined Radio*), en la que el cifrado y descifrado de voz, serán implementados en el *software* Matlab®, en el cual fueron configuradas las radios, y la parte física de la radio estará sustentada por una USRP, el cual estará hecho cargo de la transmisión y recepción de la señal por radio frecuencia.

Inicialmente, se implementará los códigos por segmentación de información, trabajando en el dominio del tiempo, y de la frecuencia. A continuación, se llevará a cabo la criptografía digital bit a bit por parámetros analíticos. Por último, se realizará el estudio de llaves para la criptografía, para lo cual se deben desarrollar pruebas anticipadas de las radios para evaluar los siguientes parámetros: (a) rendimiento con las diferentes codificaciones existentes, a fin de analizar tiempos de transmisión y recepción; (b) pérdidas en los paquetes de información; y (c) claridad del mensaje y efectividad en espacios abiertos. El enfoque de la presente investigación es brindar una solución óptima en la seguridad de la información sin afectar los demás aspectos de la transmisión.

1.4 Objetivos

1.4.1 General

Implementar algoritmos de cifrado en una Radio Definida por Software desarrollada en la tarjeta USRP N200/210.

1.4.2 Específicos

- Analizar los diferentes sistemas de criptografía seleccionados para la implementación en la SDR, incluyendo estándar militar existente.
- Adaptar el algoritmo de cifrado a diversos modelos de radio definida por software existente en el CICTE.
- Simular y modificar los parámetros de los sistemas de criptografía seleccionados para la implementación en la SDR.
- Implementar los algoritmos en la radio definida por software en la tarjeta USRP-2920.
- Evaluar el funcionamiento y seguridad obtenida de los algoritmos implementados.
- Comparar los parámetros de calidad de cada algoritmo implementado en la SDR mediante medidas objetivas de calidad.

CAPÍTULO II

2 MARCO TEÓRICO

2.1 Radio definida por software

Inicialmente, los procesos de modulación y codificación de los sistemas digitales se realizaban en las mismas radios sin embargo, en la actualidad los avances tecnológicos han permitido el desarrollo de procesadores de señales digitales y de las tarjetas FPGA (del inglés *Field-Programmable Gate Array*), que conjuntamente con la tendencia de minimizar el hardware para reemplazarlo con software, dieron origen a los sistemas de radio definido por software SDR a mediados de la década de los 90 (Selva Castañeda, 2014), los cuales han ido innovándose de forma permanente y han sido utilizados por todas las empresas que desarrollan industria militar (Nagurney, 2009).

SDR ha evolucionado, como la mayoría de las tecnologías, de entornos militares a civiles. El primer SDR operativo, conocido como *Speakeasy*, fue desarrollado por la Armada de los Estados Unidos entre 1991 y 1995. Desafortunadamente, este primer prototipo presentó varias desventajas. En primer lugar, la aplicación no se podía usar con otro que no fuera el *hardware* para el que se concibió. Además, el dispositivo tenía grandes dimensiones, llegando a ocupar por completo la parte trasera de un vehículo de transporte. Su hermano menor, *Speakeasy* II, logró un éxito mucho mayor debido principalmente a avances de principios básicos y aplicaciones en electrónica, circuitos de comunicaciones inalámbricas y técnicas de programación reutilizables y modulares (Fernández J. R., 2014).

SDR es un sistema de radiocomunicaciones que consiste en un radio reconfigurable, es decir, que permite trabajar en el mismo hardware con la capacidad de realizar diferentes funciones al reconfigurar el *software* (Meza, 2014), ya que posee prestaciones de radios multibanda. Una de sus principales aplicaciones es en la guerra electrónica, debido a que cualquier cambio tecnológico se realiza en *software*. La implementación de un sistema SDR se realiza en tarjetas FPGA a través de bloques que poseen algoritmos para su funcionamiento, gracias a que las FPGA permiten trabajar diferentes procesos de forma independiente o en paralelo.

SDR se destaca de las radios convencionales pues tiene varias capacidades, como: permitir el uso de diferentes tipos de modulaciones, codificaciones, bandas de frecuencias, potencia de transmisión, filtros, cifrado de información, entre otros módulos para el tratamiento de señales. Son estas opciones las que hacen posible trabajar en el mismo *hardware*, mediante cambios realizados en el *software* en tiempo real en cualquiera de las plataformas de implementación, y una vez finalizada, puede trabajarse sobre la SDR. Adicionalmente, posee la capacidad de realizar cambios en *software* de forma remota y de adaptarse para poder trabajar en varios tipos de estándar. (Rodríguez, 2016)

En la actualidad, tanto el software SDR, como el hardware, están disponibles a precios muy bajos (de hecho, la mayoría de las implementaciones de *software* son gratuitas), lo que invita a considerar la introducción del paradigma en soluciones de radio (Fernández J. R., 2014).

Son estas características las que hacen de la SDR una herramienta destacada en el área de telecomunicaciones pues cualquier cambio se puede realizar en la misma radio sin la necesidad

de adquirir un nuevo equipo para actualizaciones, lo que económicamente es de gran beneficio para una empresa o institución que haga uso de la SDR (Rodríguez, 2016).

En cuanto al cifrado de la información a transmitirse, se puede hacer uso de varios tipos de algoritmos, los cuales se pueden ir cambiando sin problemas debido a que son cambios en software.

2.2 USRP 2920

La tarjeta USRP 2920 es un dispositivo de Radio Definido por Software que trabaja como transceptor y cuenta con las siguientes características, un ancho de banda 20 MHz, opera en un rango de frecuencia de 50 MHz a 2.2 GHz. Al trabajar como un transceptor de RF, que se puede ajustar con un convertidor analógico-digital de alta velocidad y un convertidor digital-analógico usados para la transmisión de señales banda base I y Q hacia una PC en un puerto Gigabit Ethernet 1/10 (National Instruments Corporation, 2017).

En cuanto a prestaciones cuentan con un bajo precio global del sistema, capacidades expansivas y disponibilidad de *software*, los productos USRP son ampliamente utilizados por miles de ingenieros en todo el mundo y continúan siendo la mejor elección en *hardware* de SDR para el desarrollo de algoritmos, exploración y creación de prototipos (Ettus Research, 2011).

La USRP 2920, tiene una alta capacidad de procesamiento y trabaja en un amplio rango de frecuencias. A diferencia de las USRP de Ettus la USRP 2920 no permite el intercambio de *daughterboard*, lo que no sucede con la N210 la cual cuenta con diferentes *daughterboards* para distintos rangos de frecuencia y diferentes características en cuanto a la sincronización de fase. En cuento a la interfaz *GigaEthernet*, esta cuenta con una velocidad de transmisión de hasta 50

MS/s (del inglés *Mega Samples Per Second*), que permite trabajar con un computador para realizar el procesamiento de señales.

2.2.1 NI-USRP 290X

El NI-USRP es un controlador que incluye el *firmware* compatible con plataformas como Radio GNU e imágenes FPGA para dispositivos USRP. Además, permite el cambio de IP del dispositivo y la actualización del mismo, para ser visto por las plataformas de trabajo como Radio GNU y LabVIEW®.

2.2.2 Convertidores de la USRP 2920

La USRP 2920 cuenta con convertidores análogos/digitales ADC (del inglés *Analog to Digital Converter)* y convertidores digitales/análogos DAC (del inglés *Digital to Analog Converter*), que trabajan en la recepción realizando el proceso de decimación y en la transmisión realizando el proceso de interpolación respectivamente. La USRP 2920 tiene dos ADCs con un potencial para procesar 100MS/s y cada muestra con 14 bits de resolución, lo que le permite muestrear señales de máximo 50MHz de forma ideal, en cuanto al proceso inverso posee dos DACs con un potencial para procesar 400MS/s y cada muestra con 16 bits de resolución, permitiendo alta velocidad.

Otro de los componentes es el DDC (del inglés *Digital Down Converter*) el cual presenta dos funciones la primera es convertir una señal digital que trabaja en una banda IF (del inglés *Intermediate Frequency*) a una frecuencia BB banda base. Y la segunda función es adaptar la velocidad de los datos mediante la interpolación, para controlar la velocidad de los datos. Esto en cuanto a la transmisión de los datos (Alexander M. Wyglinski, 2013).

En la recepción trabaja el DUC (del inglés *Digital Up Converter*) al igual que DDC tiene 2 funciones la primera convierte de BB banda base a IF banda de frecuencia intermedia y la segunda es trabajar en la decimación de los datos para disminuir la tasa de muestreo. (Alexander M. Wyglinski, 2013)

El DDC permite transmitir los componentes de fase y cuadratura en un solo flujo antes de ser procesados por los filtros de interpolación (Ravi Kishore Kodali, 2013).

Para la sincronización de la transmisión y recepción de información, posee un reloj interno de 100MHz.

Todos los procesos que requieren un potencial de procesamiento alto se realizan en el FPGA, en cuanto al tratamiento de la señal de voz como es el proceso de modulación, cifrado, codificación y sus respectivos procesos inversos demodulación, descifrado, decodificación se realizan en un computador en diferentes plataformas como Simulink®, Matlab®, LabVIEW® y GNU Radio.

2.3 Simulación de una SDR

2.3.1 Códec G.711

Simulink® cuenta con un bloque de codificación basado en el estándar de la ITU G.711 que trabaja con señales de audio, basado en el cuantificador PCM (del inglés *Pulse Code Modulation*) utilizando la ley μ, para Estados Unidos y Japón y la ley A usada en Europa y el resto del mundo.

Al trabajar con audio usa el ancho de banda necesario para voz, que es de 4kHz, aplicando Nyquist se tiene una frecuencia de muestreo de 8kHz, además trabaja con 8 bits por muestra con lo que se tiene una velocidad de 64kHz (Voip-info, 2018).

2.3.2 Encoder y Decoder

2.3.2.1 Encoder Convolucional

Es un tipo de codificación que permite realizar corrección de errores mediante símbolos de paridad, realizando una aplicación deslizante de una función polinómica booleana.

Simulink® posee un bloque llamado *encoder convolutional* el cual trabaja con una estructura *Trellis*, la cual acepta dos parámetros el primero es la longitud de restricción y el segundo son los polinomios generadores. El bloque produce una salida digital (Miguel & Martinez, 2014).

2.3.2.2 Decodificador de Viterbi

El decodificador trabaja con el algoritmo de Viterbi. El algoritmo de decodificación usa dos métricas: la métrica de rama (BM) y la métrica de ruta (PM). La métrica de rama es una medida entre la distancia de lo que se transmitió y lo que se recibió, se define para cada flecha en la rejilla. En la decodificación de *hard desicion*, donde se da una secuencia de *bits* de paridad digitalizados. Se muestra un ejemplo en la Figura 1, donde los bits recibidos son 00. Para cada transición de estado, el número en las flechas muestra la métrica de rama para esa trasformación. Dos de las métricas de rama son 0, correspondientes a los únicos estados y transformaciones donde la distancia de Hamming correspondiente es 0. Las otras métricas de rama distintas de cero corresponden a los casos en que hay errores de *bit*.

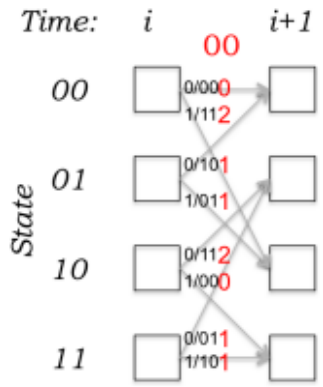


Figura 1. Diagrama trellis para la codificación
Fuente: (Massachusetts Institute of Technology, 2010)

La métrica de ruta es un valor asociado con un estado en la rejilla (es decir, un valor asociado con cada nodo). Para la decodificación de *hard desicion*, corresponde a la distancia de Hamming sobre la ruta más probable desde el estado inicial hasta el estado actual en el enramado. Por "más probable", nos referimos a la ruta con la menor distancia de Hamming entre el estado inicial y el estado actual, medido sobre todas las rutas posibles entre los dos estados. La ruta con la menor distancia de Hamming disminuye la cantidad total de errores de *bit*.

La información clave en el algoritmo de Viterbi es que el receptor puede calcular la métrica de ruta para un par (estado, tiempo) de forma incremental utilizando las métricas de ruta de los estados calculados anteriormente y las métricas de rama (Massachusetts Institute of Technology, 2010).

2.3.3 Modulación y Demodulación QPSK

Hasta hace poco en gran parte las comunicaciones seguían siendo analógicas, dependiendo de transmisores de alta potencia para superar otras fuentes que causen interferencia, condiciones climáticas y grandes edificios.

Actualmente con la aparición de las técnicas de modulación digital llegaron grandes ventajas con respecto a sus contrapartes, pueden satisfacer cualquier deficiencia en el enlace de comunicación mediante el uso de *software*. Se convirtieron en el esquema preferido para los servicios de telefonía en uso por ofrecer más inmunidad al ruido y mejores prestaciones de seguridad.

2.3.4 Modulación QPSK

Al hablar de la modulación digital de fase o modulación por desplazamiento de fase PSK (del inglés *Phase Shift Keying*) implica el cambio que se realiza en la fase, de la forma de onda transmitida. Se puede usar datos digitales para alternarlos entre dos señales de fase opuesta, pero de que tengan la misma frecuencia para generar una forma de onda modulada en fase, en su forma más simple. Si se multiplica una onda sinusoidal por la forma de onda resultante, las dos en la misma frecuencia, generaran dos componentes: un término independiente de la frecuencia que su amplitud es proporcional al coseno del cambio de fase y una forma de onda del coseno del doble de la frecuencia recibida. Se filtra el término de la frecuencia más alta produce los datos originales de la modulación antes de ser transmitida, es difícil imaginar, pero es como trabaja PSK (Herrera, Riofrio, & Vázquez, 2015).

El número de cambios de fase no se limita a dos estados, con QPSK (del inglés *Quadrature Phase-Shift Keying)* la portadora sufre cuatro cambios de fase, por lo que puede representar 2 bits de datos por símbolo, permitiéndole a la portadora transmitir 2 bits de información en lugar de 1. La señal "portadora" transmitida puede sufrir cualquier número de cambios de fase modificando así de manera efectiva su ancho de banda (Herrera, Riofrio, & Vázquez, 2015).

2.3.5 Demodulación QPSK

El demodulador para QPSK, está bajo la modulación de desplazamiento de fase PSK, modulación digital que transmite datos cambiando la fase de una señal portadora. Utiliza cuatro puntos en el diagrama de la constelación equidistantes alrededor de un círculo.

El demodulador QPSK descompone la señal de entrada a través de dos multiplicadores que los acciona un oscilador de frecuencia fija, con salidas desfasadas 90 grados.

El circuito utiliza compuertas lógicas, amplificadores y mezcladores para realizar la demodulación. La variedad de PSK que se utiliza depende de la velocidad de los datos a la que se esté transmitiendo (Herrera, Riofrio, & Vázquez, 2015).

2.3.6 Bloque de cifrado y descifrado

El bloque de cifrado es una herramienta muy útil cuando se desea tener seguridad informática; puede ser también entendida como un medio para garantizar las propiedades de confidencialidad, integridad y disponibilidad de los recursos de un sistema, pero hay que saber cómo utilizarla, para ello es importante tener claros los conceptos básicos que están detrás de los sistemas criptográficos modernos (SAISO, 2018).

El bloque de descifrado invierte los pasos del algoritmo de cifrado que se encuentra en el receptor, mediante el uso de llaves públicas o privadas, con lo que se garantiza la confidencialidad e integridad de la información transmitida.

2.4 Sincronización

La sincronización entre la señal enviada y la señal recibida es de vital importancia en los sistemas de telecomunicaciones, ya que pueden existir desfases, retardos, entre otros efectos

causados por el canal o por los equipos en los que se trabaje. Es así como se presentan los diferentes tipos de desincronización cuyo efecto se percibe en el receptor ya que este no puede decodificar una señal que no llegue sincronizada.

Los tipos de sincronización que se requieren son los siguientes:

- Sincronización de tiempo de símbolo: es la determinación de la muestra correcta del inicio de símbolo que se realiza previo a la demodulación.
- Sincronización de frecuencia portadora: el objetivo de este tipo de sincronización es el suprimir la frecuencia *offset* que se genera debido a los osciladores locales del transmisor y el receptor cuando presentan un desajuste, además del desplazamiento que se da en la portadora provocado por el medio de propagación.
- Sincronizar el reloj de muestreo: esta herramienta es usada para disminuir los errores del reloj de muestreo que se produce por el desajuste de los osciladores de cristal (Fernández M. A., 2008).
- Sincronización de paquete: se busca sincronizar los paquetes validos informando al receptor cuando se inicia la transmisión mediante el uso de un preámbulo al inicio. Pero este producirá un bloqueo cuando se presentan varios errores en el preámbulo evitando que se propague la señal al siguiente bloque en el receptor (Hidalgo, 2014).

Para lograr la sincronización de las tarjetas USRP N210 existen dispositivos externos como:

- Cable MIMO (del inglés *Multiple-input Multiple-output*): permite la sincronización de frecuencia y tiempo de dos USRP de la serie N200/210. Trabajando en modo maestro esclavo, el reloj se encuentra en el USRP maestro.

- Sincronización con oscilador GPS (del inglés *Global Positioning System*) restrictivo: permite la sincronización de tiempo en un área geográfica usando un oscilador restrictivo por GPS (GPSDO), deriva señales de 10 MHz/PPS (del inglés *Pulse per Second*) del sistema GPS. Su precisión es de ± 50 nano segundo.
- Sincronización con señales de 10 MHz y 1 PPS: permite sincronizar mediante el uso de una referencia externa de alta precisión denominada OctoClock que puede ser una fuente de Rubidio. Se recomienda usar para un sistema MIMO con más de dos canales.
- API UHD: permite configurar parámetros de sincronización para la USRP N200/N210, los cuales también se encuentran a través de GNU radio (Ettus Research, A National Instruments Company, 2018).
- La plataforma Simulink® cuenta en su librería con bloques que realizan la sincronización mediante software entre los cuales se presenta:
- *Symbol Synchronizer*: el bloque se encarga de corregir el sesgo del reloj de sincronización de símbolos entre el transmisor y el receptor con modulación PAM (del inglés *Pulse Amplitude Modulation*), PSK o QAM. Acepta una entrada de tamaño fijo o variable pero su salida es una señal de tamaño variable. Presenta varios métodos para la detección del error de temporización como son *Zero-Crossing*, *Gardner*, *Early-Late*, o Mueller-Muller, los cuales se pueden seleccionar en el bloque. Permite configurar las muestras por símbolo, el factor de amortiguamiento del filtro, ancho de banda normalizado del filtro de bucle y un detector de ganancia (MathWorks, 2018).

- *Carrier Synchronizer*: realiza la compensación de la frecuencia de portadora y los desplazamientos de fase de señales con modulación PAM, PSK o QAM. Se configura el tipo de modulación, desplazamiento de fase, la muestras por símbolo, el factor de amortiguación del lazo y el ancho de banda normalizado del ciclo (MathWorks, 2018).
- *Coarse Compensator Frequency*: realiza la compensación del desplazamiento en frecuencia de las señales recibidas con modulación PAM. PSK o QAM. Utiliza un algoritmo basado en la correlación (MathWorks, 2018).
- *Compensator Imbalance* I/Q: el bloque se encarga de compensar el desequilibrio entre las componentes en fase y en cuadratura de una señal modulada (MathWorks, 2018).
- *Frame Synchronizer*: realiza la detección de datos válidos mediante la sincronización de cuadros, para lo que requiere de un preámbulo o datos fijos para distinguir el inicio de cada cuadro el cual se añade en el transmisor, en el receptor se realiza la detección del preámbulo para sincronizar los cuadros. Trabaja con datos complejos y su salida es compleja de tamaño fijo, por lo que se usa a la salida del bloque *Symbol Synchronizer* para recuperar una señal de tamaño fijo (MathWorks, 2018).

2.5 *Modifiel Bark Spectral Distorsion* MBSD

MBSD es una medida objetiva de calidad de voz, que estima la distorsión de la voz en el dominio de la sonoridad teniendo en cuenta el umbral de enmascaramiento del ruido con el objetivo de incluir las distorsiones en el rango audible. La medida MBSD se correlaciona con la medida MOS (del inglés *Mean Opinion Sccore)* lo cual es una ventaja, ya que realizar el proceso para la medida MOS representa un mayor consumo de recursos económicos.

La medida MBSD se basa en la medida BSD (del inglés *Bark Spectral Distorsion*), la cual se basa en el cálculo promedio de la distancia euclidiana al cuadrado entre la señal original y la señal recuperada del proceso de codificación para poder calcular la sonoridad de la señal recuperada (Yang & Yantorno, 1999).

2.6 Cifrado de Información

El cifrado de la información es de vital importancia para la comunicación entre un emisor y un receptor desde tiempos remotos, en la actualidad los sistemas tecnológicos actuales son más vulnerables. Con el fin de generar seguridad en la transmisión de información que se generaron diversas formas para brindar seguridad, un método para ello es el uso de algoritmos de cifrado cuya función es cambiar la información, dándole un mensaje errado para un tercero ajeno a la comunicación.

Las telecomunicaciones se dan por medios físicos e inalámbricos en donde pueden ser vulneradas por un atacante con el fin de obtener acceso a la información que se transmite por estos medios, con los algoritmos de cifrado el objetivo no es ocultar que existe información sino alterar la información evitando que sea comprendida por el intruso.

Partiendo del concepto de algoritmo que no es más que un conjunto de pasos que permiten realizar una tarea, por lo que deben ser claros y concisos. Los algoritmos de cifrado son los pasos que permiten realizar la transformación de información mediante métodos o técnicas matemáticas con el fin de generar nueva información cifrada mediante lo que se denomina clave de cifrado, que debe ser lo más segura posible dependiendo del nivel de privacidad que requiera la

información, además de que existen dos tipos de claves de cifrado que son simétricas y asimétricas (Luz, 2010).

Para el caso de la información de voz existen algoritmos de cifrado que trabajan con segmentos de la señal de voz ya sea en tiempo, en frecuencia, o en tiempo-frecuencia, también existen algoritmos de cifrado digitales que trabajan bit a bit, cifrados por parámetros analíticos. Los algoritmos de cifrado por segmentos de información básicos van desde el intercambio de posiciones de los segmentos de información, hasta algoritmos más complejos los cuales generan mayores niveles de seguridad y un mayor costo computacional.

Los algoritmos de cifrado básicos se basan en el cifrado de César que lo que hacía era desplazar una letra un número de espacios en el alfabeto.

Los algoritmos de cifrado buscan alcanzar aspectos de seguridad como son la confidencialidad, la integridad y la disponibilidad.

2.6.1 Historia del cifrado (Importancia)

La historia de la criptografía es considerada tan antigua como la misma escritura, ya que fue parte de todas las civilizaciones, lo cual se respalda mediante documentos, pero no se usaba de forma frecuente, su uso empieza a regularizar en la Edad Media.

(Díaz, 1995) dice. “El vocablo criptografía proviene de dos voces griegas: *κρύπτος* (criptos), que significa *escondido,* y *γραφη* (grafé), *grafo* o escritura” (p.15). Desde el punto de vista etimológico, la criptografía es la ciencia que se dedica al estudio de la escritura oculta. Para ello, esta disciplina emplea un lenguaje basado en claves o cifras, conocido también como cifrario (Díaz, 1995).

Existe documentación que respalda que la criptografía pudo haber iniciado en las civilizaciones antiguas, sin embargo, se empezó a desarrollar más habitualmente durante la Edad Media y Renacimiento (Jesus J. Ortega Triguero, 2006). En datos históricos se encuentra que en el siglo V A.C., los griegos conocían varios métodos para enviar mensajes ocultos. Entre ellos, destaca el uso de una tabla recubierta de cera, e incluso mensajes secretos escritos en el cuero cabelludo (Fernández S. , 2004).

En 1411, aparecen las primeras cifras homofónicas. En esta época, con el objetivo de combatir el análisis de frecuencias, se introducen los homófonos y los nulos. Estos últimos, no estaban vinculados a ninguna letra, eran colocados aleatoriamente a lo largo del texto cifrado para confundir cualquier criptoanalista que hiciera un análisis de frecuencias del texto cifrado (Fiarresga, 2010).

La criptografía empezó a ser considerada como una ciencia aplicada luego de que Claude Shannon publicó la "*Teoría de las comunicaciones secretas"* en 1949, la cual fue la base para el desarrollo del sistema de criptografía DES realizado por la NBS de Estados Unidos. Además, se reconoció que la criptografía es una ciencia interdisciplinaria, que requiere aplicaciones de aritmética, estadística, teoría de números, teoría de la información y teoría de la complejidad computacional (Martín, 2016).

2.7 Algoritmos o técnicas de cifrado

2.7.1 Criptografía por Segmentación de la Información (CSI)

La Criptografía por segmentación de la información es una técnica que combina disciplinas como las Matemáticas, la Informática y la Telemática, con el objetivo de proteger un mensaje

dividiendo el bloque de la señal en secciones de información para realizar permutaciones y hacer ilegible la señal de salida. Se busca con esto dejar al espectro de la señal cifrada lo más plano posible alterando los parámetros de frecuencia, amplitud y posicionamiento temporal para ofrecer resistencia al criptoanálisis.

2.7.2 Criptografía por Segmentación de la Información en el dominio del Tiempo (CSI-T)

Esta técnica realiza la criptografía por medio de las permutaciones de N elementos temporales que por lo general son segmentos que se toman dentro del bloque de la señal, para realizar este cambio se debe almacenar los bloques en una memoria esto quiere decir que en la transmisión existe un retraso de dos veces el tamaño del bloque, limitando en el proceso de comunicación el número de segmentos de los bloques.

El limitante del número de bloques resulta un problema al momento de la seguridad ya que al aumentar el valor de esta variable aumentaría la resistencia al criptoanálisis, pero aumentaría el tamaño de la banda necesitando un mejor sincronismo para la comunicación.

Para implementar CSI-T la forma más común consiste en dividir la señal de voz, digitalizada $x(n)$ en bloques con una duración típica igual a $N \times 20$ ms, donde N es el número de segmentos de permutación utilizados. Los bloques divididos en N segmentos, se intercambian para formar los bloques cifrados como se muestra en la Figura 2. Antes de realizar el proceso de transmisión, se debe convertir la señal de vuelta al formato analógico (Junior J. F., 2008).

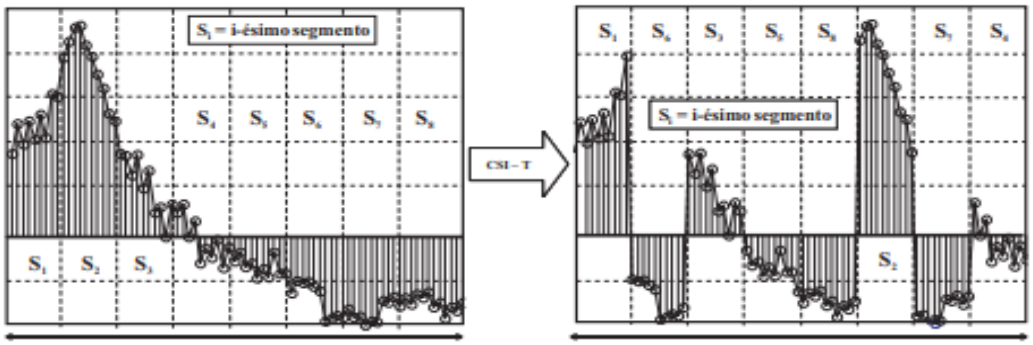


Figura 2. Permutación de los datos
Fuente: (Junior J. F., 2008)

Para una señal de voz con M bloques, $\vec{\boldsymbol{x}}_m$, $m = 1,\ldots,$ M, en que cada bloque tiene N segmentos, cada segmento que contiene R muestras de la señal. El i-esimo bloque puede ser representado por el vector $\vec{\boldsymbol{x}}_i = [\overrightarrow{\boldsymbol{S}_1^T}\ \ \overrightarrow{\boldsymbol{S}_2^T} \ldots \overrightarrow{\boldsymbol{S}_N^T}]^T$ donde el i-esimo segmento está definido como $\vec{\boldsymbol{x}}_j = [\frac{x^i_{(j-1)R}}{N} + 1\ \frac{x^i_{(j-1)R}}{N} + 2 \ldots . \frac{x^i_{(j-1)R}}{N} + R\ ]^T$ Los elementos componentes del vector $\vec{x}_i$ pueden ser reordenados en la forma matricial:

$$\boldsymbol{X}_i = [\vec{\boldsymbol{S}}_1\ \vec{\boldsymbol{S}}_2 \ldots \vec{\boldsymbol{S}}_N]^T \qquad (1)$$

$$= \begin{bmatrix} x_1^i & x_{R+1}^i & \cdots & x_N^i \\ x_2^i & x_{R+2}^i & \cdots & \vdots \\ \vdots & \vdots & \ddots & x_{N*R-1}^i \\ x_R^i & x_{2R}^i & \cdots & x_{R*1}^i \end{bmatrix}_{R*N} \qquad (2)$$

Entonces se puede definir una matriz de permutación **P** de dimensión $N \times N$, cuya composición admite sólo un elemento no nulo en cada fila y en cada columna. Para garantizar la preservación de la energía de la señal, la norma de la matriz de permutación P_i debe ser unitaria, para ello, el elemento no nulo debe tener valor igual a la unidad.

Realizando el producto de las matrices **P** y $\boldsymbol{X}_i$, concatenando las filas de la matriz resultante, se llega al bloque de la señal de voz cifrada $\vec{y}_i$:

$$\boldsymbol{Y}_i = \boldsymbol{P}\boldsymbol{X}_i = [\vec{\boldsymbol{y}}_1\, \vec{\boldsymbol{y}}_2 \ldots \vec{\boldsymbol{y}}_N]^T \quad (3)$$

$$\boldsymbol{Y}_i = \begin{bmatrix} y_1^i & y_{R+1}^i & \cdots & y_N^i \\ y_2^i & y_{R+2}^i & \cdots & \vdots \\ \vdots & \vdots & \ddots & y_{N*R-1}^i \\ y_R^i & y_{2R}^i & \cdots & y_{N*1R}^i \end{bmatrix}_{R*N} \quad (4)$$

$$\vec{\boldsymbol{y}}_i = \left[y_1^i\, y_2^i \ldots y_{N*R}^i\right]^T \quad (5)$$

El proceso para descifrar la señal es similar al proceso de cifrado, donde la matriz **P** es sustituida por su inversa, como se detalla a continuación:

$$\boldsymbol{X}_i = \boldsymbol{P}^{-1}\boldsymbol{Y}_i = \boldsymbol{P}^{-1}\boldsymbol{P}\boldsymbol{X}_i \quad (6)$$

En el receptor, $\boldsymbol{Y}_i$ puede ser obtenida cambiando el vector $\vec{\boldsymbol{y}}_i$ en la forma de una matriz de dimensiones $R \times N$. Entonces $\vec{x}_i$ obtenido por la cocatenación de las líneas de $\boldsymbol{X}_i$ es calculada por la ecuación (6).

La estimación precisa del retraso provocado por los esquemas de CSI-T depende del nivel de seguridad exigido, lo que demuestra que el problema del retraso excesivo no puede ser tratado de manera aislada. Un sistema típico, con N segmentos y un tiempo de duración de segmento (Ts), presenta un retraso total de 2NTs, que, para $N = 8$ y $Ts = 20ms$, es de 320 ms. Por otro lado, si segmentos menores que 20 ms se utilizan no hay la preservación de la banda de la señal de voz original. Los siguientes factores limitan la aplicación de la CSI-T:

- La introducción de retrasos demasiado grandes y que aumentan con la longitud de la clave de cifrado (número de permutaciones);
- Proceso de sincronismo critico; y
- Llaves capaces de producir una inteligibilidad residual baja (Junior J. F., 2008).

2.7.3 Criptografía por Segmentación de Información en el dominio de la Frecuencia (CSI-F)

Este sistema realiza una criptografía similar a la CSI-T por medio de permutaciones de los N elementos divididos en segmentos, pero esta vez en elementos de frecuencia, en esta técnica encontramos desde simples convertidores hasta mezcladores muy elaborados, estos sistemas ofrecen niveles de seguridad muy bajos por lo que se los utiliza para protección contra oyentes que no disponen de buenos equipos de criptoanálisis.

Los primeros codificadores de CSI-F emplearon técnicas de inversión de frecuencia, que consiste en la inversión del espectro de la señal o de parte de éste con el fin de hacer la señal ininteligible a los oyentes que no que tengan receptores capaces de deshacer la inversión espectral de la señal. Estos inversores, debido a la simplicidad de deshacer el proceso de criptofonía, no son más empleados, excepto en radios domésticos del tipo FRS, conocidos comercialmente como *Talk-About* (Junior J. F., 2008).

Con el surgimiento de nuevos circuitos DSP, capaces de realizar tareas complejas con alto nivel de miniaturización, fue posible proyectar sistemas de CSI-F implementados con bancos de filtros y transformaciones ortogonales.

Si el número de sub-bandas (o subfases) es suficientemente pequeño, la señal muestra la inteligibilidad residual. Para superar este problema, se debe elegir un número de sub-bandas y una clave entre aquellas que generan baja inteligibilidad residual. Otra forma de mejorar el desempeño de los sistemas CFI-F es realizar cambios en las claves de manera periódica y aleatoria, de acuerdo con un polinomio generador de secuencias pseudo-aleatorias (Junior J. F., 2008).

2.8 Criptografía simétrica

Los cifrados simétricos pueden ser divididos en dos tipos:

- Cifrado en flujo: En este tipo de cifrado cada carácter del mensaje se cifra del mismo modo independientemente de los caracteres que le rodean. Los más destacados son:
- Cifrado de César.
- Cifrado con máquina enigma.
- Cifrado de Vernam.
- Cifrado en bloque: El resultado de cifrar no depende sólo de un carácter, sino de un grupo de caracteres que le rodean. Los algoritmos más destacados son:
- Cifrado de Hill.
- DES: Primer estándar de cifrado en bloque.
- AES: Actual estándar de cifrado en bloque (Santos, 2010).

2.8.1 Algoritmo de Cifrado DES

La técnica de cifrado DES es un método de cifrado de datos de clave simétrica y consta de dos procesos, el proceso de cifrado y el proceso de descifrado. Cada uno de estos procesos tiene un proceso de ocho pasos. El proceso de descifrado se lo realiza invirtiendo los pasos del proceso del cifrado. La clave utilizada en el método de cifrado DES tiene 56 bits de longitud efectiva (Ratnadewi, 2009).

El proceso de cifrado utilizando el método criptográfico DES consiste en un proceso de ocho pasos, que son:

1. Convertir el mensaje y la clave que se procesará en *bits* binarios. El texto plano y la clave que se ha convertido se divide en bloques de datos, donde cada bloque tiene una longitud de 64 *bits* (ocho *bytes*). Si el mensaje está en forma de alfabeto o símbolos**,** se debe convertir primero a formato decimal y hexadecimal siguiendo la tabla de caracteres ASCII, y luego se convierten en *bits* binarios.
2. Se vuelve aleatorios los *bits* en el bloque de datos sin formato basado en la tabla de permutación inicial (IP), de modo que la secuencia de *bits* se vuelva aleatoria en comparación con la secuencia de *bits* del bloque de texto sin formato inicial. La secuencia de *bits* después del segundo paso sigue los resultados de la tabla IP presentada en la Tabla 1, con el primer *bit* derivado del *bit* de secuencia 58th de bloques de texto original, y luego el segundo bit derivado de los bits de secuencia 50th hasta el bit de secuencia 64th derivado del séptimo.

Tabla 1

Tabla de Permutación Inicial IP

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

3. La codificación de la clave se basa en la tabla permutada del paso 1 (PC - 1). Los resultados de PC - 1 tienen una longitud de 56 *bits* porque los últimos *bits* de cada *byte* de la clave (8, 16, 24, 32, 40, 48, 56 y 64 *bits*) que actúa como el *bit* de paridad no se utilizan nuevamente en el siguiente paso. Una vez completados, los resultados de PC - 1 se dividen en 2 partes que se denominan C_0 y D_0, en C_0 hay 28 *bits* a la izquierda y D_0 hay 28 *bits* a la derecha de PC - 1.
4. Se cambia los *bits* a la izquierda (desplazamiento a la izquierda) en C_i y D_i, con el valor de i basado en una ronda de proceso de cifrado que consta de 16 rondas. El resultado de los *bits* de cambio de cada ronda de C_i y D_i se combinan luego en C_iD_i con una longitud de 56 bits. Después de eso, los *bits* clave C_iD_i se vuelven aleatorios en base a la tabla PC - 2 (elaborada en el paso 2 del proceso) hasta que se produzca la variable Ki.
5. La ejecución del proceso de expansión de datos de Ri-1 con una longitud de 32 *bits* (comenzando desde el R_0 de los resultados del segundo paso del proceso) se convierte en Ri con una longitud de 48 *bits*, donde i es la ronda durante el proceso. Este proceso se llevará a cabo hasta 16 veces. Los resultados del proceso anterior se conocen como E (Ri-1),

comenzando desde E (R_0) hasta E (R_{15}). Después, E (Ri-1) será procesado XOR con Ki que se ha obtenido en el proceso del cuarto paso para cada *bit* corresponde a la ronda de proceso en ejecución para producir una variable A_i con una longitud de 48 bits y en forma de vector.

6. Una vez obtenido A_i se descompone en ocho bloques, cada bloque consta de seis *bits*. Cada bloque se distribuye luego en ocho casillas de *S-Box* (Caja de sustitución), con el primer bloque distribuido al *S-Box* 1, el segundo bloque distribuido al *S-Box* 2 y así sucesivamente. El resultado del proceso de sustitución que usa *S - Box* será recolectado y producirá la variable B_i.
7. Una vez que se obtiene la variable B_i, el siguiente paso es realizar el proceso de permutación en cada *bit* de la variable B_i usando la tabla P-Box. Los resultados obtenidos de la permutación usando *P - Box* se denominan P (B_i), con i adaptado a la ronda durante el proceso, comenzando desde P (B_1) hasta P (B_{16}). A partir de entonces, P (Bi) será procesado XOR con el L_i-1 obtenido a partir del segundo paso del proceso de acuerdo con los procesos que se ejecutan para producir una variable R_i con una longitud de 32 bits y en forma de vector. Los resultados de R_i luego se fusionan con L_i, que resulto del R_i -1, hacia L_i R_i, que es el resultado del proceso de cifrado del texto sin formato para cada proceso de ronda con una longitud de 64 bits.
8. El octavo paso del proceso llevado a cabo cuando el séptimo paso del proceso ha obtenido el L_{16} y el R_{16} desde la 16va ronda de proceso. El siguiente paso es el proceso de invertir posiciones en L_{16} y R_{16}, y luego se combinan para obtener la forma $R_{16}L_{16}$. Estos resultados luego se permutan usando una tabla IP-1 (Permutación Inversa Inicial). Los resultados

obtenidos a partir de los ocho pasos del proceso es un bloque cifrado, una combinación de varios cifrados se llama texto cifrado.

El primer y el segundo paso del proceso se realizan solo una vez al comienzo del proceso de cifrado DES, mientras que los ocho pasos del proceso se realizan solo una vez al final del proceso.

El tercer paso hasta el séptimo paso del proceso se lleva a cabo 16 veces de acuerdo con el número de rondas de proceso Feistel (método de cifrado en bloque creado por Horst Feistel que cuenta con una estructura particular) utilizado por el método criptográfico DES. El diagrama de bloques del proceso de cifrado DES se muestra en la (Ratnadewi, 2009).

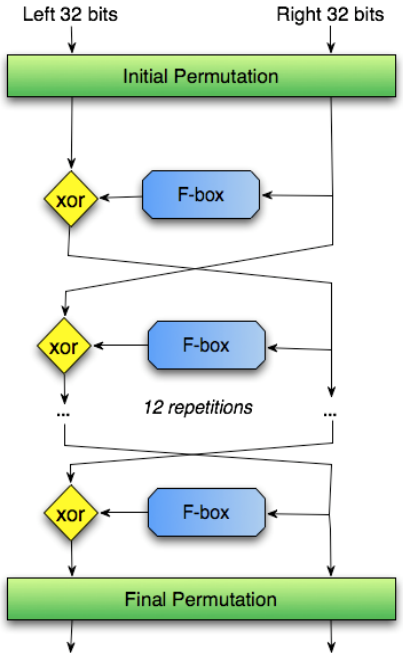


Figura 3. Diagrama de Bloques del cifrado DES
Fuente: (Ratnadewi, 2009)

2.8.2 Algoritmo de Cifrado AES

El AES es un estándar de cifrado de bloques simétrico basado en iteraciones de cuatro transformaciones (*AddRoundKey*, *SubBytes*, *ShiftRows* y *MixColumns*). El número de iteraciones también llamadas rondas depende de la variante seleccionada y de la longitud de la clave. Cada transformación se realiza una vez por ronda, excepto en la última ronda donde no se incluye la transformación *MixColumns*. Comúnmente, el AES-128 falla en las últimas rondas para recuperar la clave de la última ronda completa o un subconjunto de la misma. Con este conocimiento, la clave inicial de 128 bits se calcula gracias al cálculo del algoritmo inverso de expansión de clave (Floissac & L'Hyver, 2011)**.**

El *Federal Information Processing Standard* 197 usó una versión estandarizada del algoritmo llamado Rijndael para *Advanced Encryption Standard.*

2.8.2.1 Bloques y Clave Rijndael

Antes de aplicar el algoritmo a los datos, se deben determinar los tamaños de bloque y clave. AES permite tamaños de clave de 128, 192 y 256 bits. La encriptación estándar usa AES-128 donde el bloque y el tamaño de la clave son 128 bits. El tamaño del bloque se denomina comúnmente como N_b y el tamaño de la clave como N_k. N_b se refiere al número de columnas en el bloque donde cada fila de la columna consta de cuatro celdas de 8 bits cada una para AES-128. Se puede observar el diagrama de bloques del sistema en la Figura 4 (Selent, 2010).

2.8.2.2 Rondas de Rijndael

El algoritmo de Rijndael utiliza varias rondas para transformar los datos de cada bloque. El número de rondas utilizadas es 6 + el máximo de N_b y N_k. Como N_b y N_k son ambos 4, el

número de rondas es 6 + 4 = 10. El bloque inicial (también conocido como estado) se agrega a una clave expandida derivada de la clave de cifrado inicial. Luego, se produce el procesamiento de ronda que consiste en turnos de operaciones. El estado resultante se agrega a la siguiente clave expandida. Esto se hace para las diez rondas, con la excepción de la operación MixColumn de la ronda final. El resultado final es el bloque de cifrado encriptado (Selent, 2010).

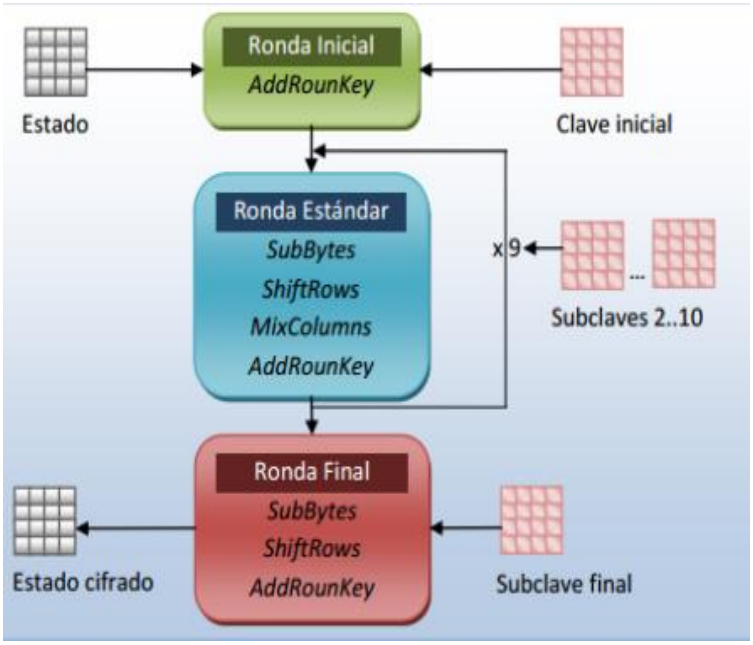


Figura 4. Diagrama de Rondas, cifrado AES
Fuente: (Pousa, 2018)

2.8.2.3 Rijndael Key Expansion

La clave de cifrado original debe ampliarse con en un valor de (r + 1). Se necesita una llave redonda después de cada ronda y antes de la primera ronda. Cada clave redonda debe ser de 16 bytes porque el tamaño del bloque es de 16 bytes. Por lo tanto, la clave de cifrado debe expandirse de 16 bytes a 16*(r + 1) bytes o 176 bytes. La clave expandida se divide en llaves redondas como se muestra en la Figura 5 (Selent, 2010).

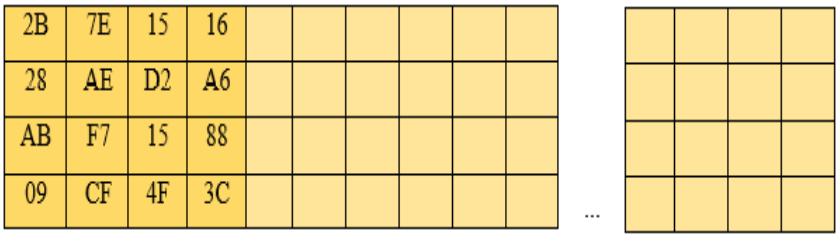


Figura 5. Cálculo de las Sub Claves

2.8.2.4 Rijndael *S-Box*

El primer paso de una ronda es hacer la sustitución de cada byte del texto original por bytes de una tabla de búsqueda llamada S-box. Un S-box es un mapeo uno a uno para todos los valores hexadecimales de 0 a F. El S-box se usa para cambiar el texto plano original en bytes al texto de cifrado. El S-box se muestra a continuación en la Tabla 2. Todos los valores están representados en notación hexadecimal. Así es como comúnmente se ve el S-box (Selent, 2010)..

Tabla 2

Tabla S-Box

	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	EC	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

Por ejemplo, si se tiene el texto simple de un carácter "D", se traduciría al valor hexadecimal de 44 utilizando una tabla de búsqueda ASCII. Cuando se utiliza la *S-box*, el primer dígito en hexadecimal representa las filas de la tabla o los valores en el lado izquierdo disminuyen. El segundo dígito representa la columna de la *S-box*, que, en este ejemplo, también es 4. Usando la S-box encontramos el número de fila 4 y el número de columna 4 y encontramos que el valor hexadecimal en esa celda es "1b". Así es como funciona la *S-box*. El uso de este método con todo el texto simple generará los nuevos valores hexadecimales que se utilizarán más adelante en el algoritmo.

La S-box proviene de la aritmética modular y a un campo de Galois. Un campo de Galois es un campo con un número finito de elementos. El campo Galois es siempre un campo que tiene el poder de un número primo. Para cada número primo existe exactamente un campo de Galois. La notación para representar un campo de Galois es GF (p), donde p es el número primo.

Generar la S-box del campo elegido requiere mucho trabajo. Se pensó cuidadosamente en la transformación por motivos de seguridad. Utilizar texto plano para los siguientes pasos del algoritmo lo haría más vulnerable, por lo que se utilizó una sustitución de bytes en forma de *S-box*. Los bytes originales se transforman utilizando el inverso multiplicativo y una matriz de afinidad para hacer que el texto de cifrado sea resistente a los ataques algebraicos (Selent, 2010).

2.8.2.5 Rijndael *Shifts*

El siguiente paso en la ronda es cambiar las filas del estado. Las filas se desplazan x número de casillas a la izquierda donde x es el número de fila. Esto significa que la fila 0 no se desplazará, la fila 1 se desplazará 1 casilla hacia la izquierda, la fila 2 se desplazará 2 casillas

hacia la izquierda y la fila 3 se desplazará 3 casillas hacia la izquierda. El estado resultante se muestra en la Figura 6, aplicando los cambios al estado anterior. Es necesario considerar que los cambios no afectan a la sustitución de bytes, por lo que las operaciones de desplazamiento y de *S-box* podrían realizarse en un orden diferente (Selent, 2010).

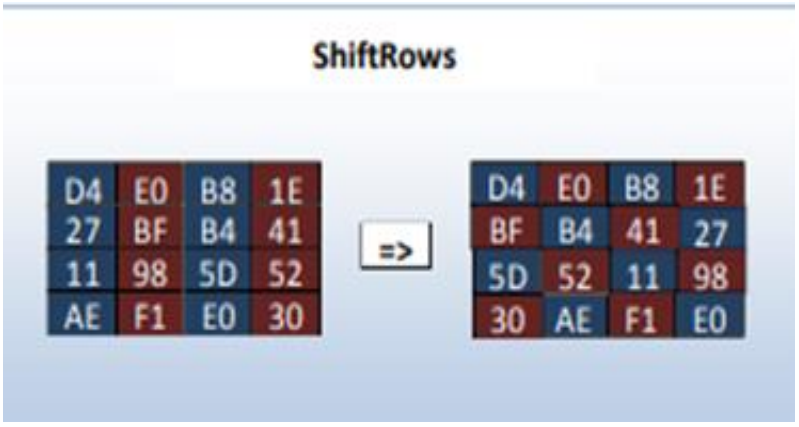


Figura 6. Ejemplo de la función ShiftRows
Fuente: (Pousa, 2018)

2.8.2.6 Rijndael MixColumns

Después de aplicar la S-box y cambiar al estado, se usa la operación llamada *MixColumn*. La función toma un byte y lo transforma en cuatro bytes. Cada elemento de la tabla consta de cuatro bytes generalmente representados como valores hexadecimales. El segundo y tercer bytes son siempre los mismos que el byte de entrada. Para el valor de 54, se obtendría un segundo y tercer valor de 54 también. El primer byte es multiplicado por 2. Si el resultado es mayor que 0xFF, al resultado se le aplica XOR con el valor 0x1B y mod 100. Por lo tanto, el byte 0x54 es (54 XOR 1b) mod 100 = A8. El último byte es agregado al primer byte. Entonces, el último byte es A8 XOR 54 = FC. Esta es una forma simple de generar la tabla *MixColumn* (Selent, 2010).

2.8.2.7 Descifrado Rijndael

El descifrado es simple después de comprender el proceso de encriptación. Básicamente solo es lo inverso. El algoritmo fue diseñado para que todos los pasos sean reversibles, por lo que el descifrado es básicamente como hacer todo al revés. Por lo que para descifrar se comienza en la última ronda. Al procesar cada ronda, se realiza el proceso al revés. El paso de *MixColumn* se aplica a todas las rondas excepto a la última. También se usa la tabla inversa *MixColumn*. Esta tabla se genera con otra matriz similar a la forma en que se generó la tabla *MixColumn*. La diferencia es que no hay atajos para generar la tabla. Por lo tanto, la multiplicación de la matriz debe realizarse en el campo GF (2^8).

Todos los *shifts* se hacen al revés también. Entonces, en lugar de cambiar la tabla a la izquierda, cambiamos a la derecha. Por último, el S-box se aplica utilizando la tabla S-box inversa. La tabla S-box inversa se puede generar fácilmente tomando el valor S-box en algún índice de fila y columna y asignando la fila y columna al valor S-box inverso, en el índice S-box inverso definido por el valor S-box. La tabla de S-box inversa se muestra en la Tabla 3.

Después de que todas las rondas hayan sido completadas en orden opuesto, el estado final contendrá el texto plano original (Selent, 2010).

Tabla 3

Tabla S-Box inversa

		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

CONTINÚA

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

CAPÍTULO III

3 ALGORITMOS Y SIMULACIONES

En el presente capitulo se presentan todas las simulaciones y pruebas realizadas en diferentes plataformas de simulación como son: Simulink®, Matlab® y LabVIEW®, esto debido a que el trabajo inicial estaba planeado en Simulink® ya que el objetivo fue dar seguridad a la SDR desarrollada por (Capt. Paredes & Angulo, 2011).

Para poder trabajar con las tarjetas propuestas que son USRP N210, se procedió a probar el correcto funcionamiento de las mismas mediante transmisión y recepción de los modelos ya implementados en la tesis (Capt. Paredes & Angulo, 2011) , para lo cual se contó con las tarjetas Ettus N210.

Las pruebas iniciales de funcionamiento de la SDR no tuvieron éxito por lo que se procedió a realizar varias pruebas trabajando con la capacidad de los DAC y ADC, mediante el diezmado e interpolación de la señal, con el fin de verificar el correcto funcionamiento de las tarjetas USRP N210 y de las *daugtherboard* SBX y WBX, se realizaron pruebas modificando el valor de la interpolación en el transmisor y del diezmado en el receptor esto se realizó de dos formas:

Primero de forma teórica basadas en otras tesis donde se encontró que el valor del diezmado es la mitad del valor de interpolación.

Segunda realizando un cálculo en base al ancho de banda de recepción y transmisión, tomando como base la velocidad de muestreo de las tarjetas encargadas de la conversión análogo/digital ADC y digital/análogo DAC.

$$BW_{TX} = \frac{400\ MS/s}{Interpolación} \quad (7)$$

$$BW_{RX} = \frac{100\ MS/s}{Decimación} \quad (8)$$

Dando como resultado:

$$BW_{TX} = BW_{RX} \quad (9)$$

Con lo que se obtiene una relación de $\frac{1}{4}$ entre el valor de decimación e interpolación.

$$Decimación = \frac{1}{4} Intepolación \quad (10)$$

En las primeras pruebas se utilizó un valor de Interpolación I=512 en el transmisor y Diezmado D=256 en el receptor aplicando la primera forma teórica.

Con lo que no se obtuvieron grandes cambios ya que se comprobó que existía recepción mediante un analizador espectral propio de Simulink® Figura 7, donde se observó que si recibe la señal transmitida ya que el pico de potencia sube a -55dBm cuando el transmisor entra en funcionamiento, pero los datos recibidos se escucharon con ruido.

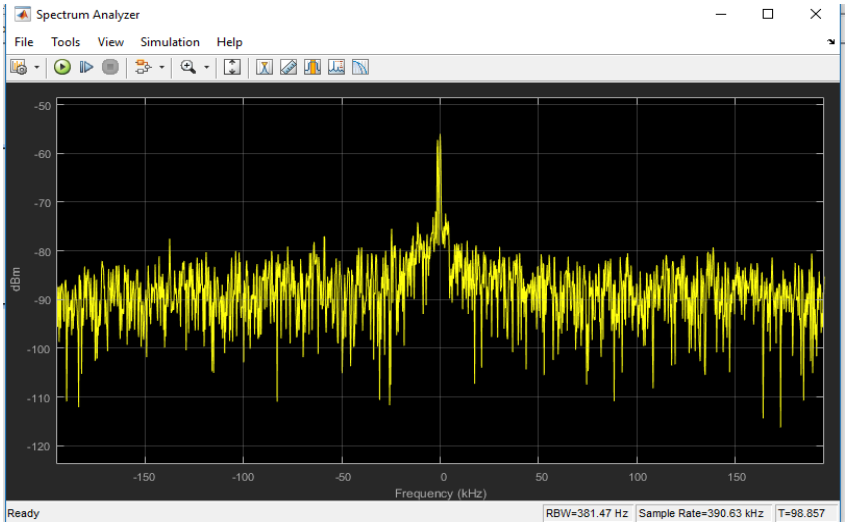


Figura 7. Analizador Espectral en la Recepción

Usando la misma relación se probaron varios valores de interpolación y decimación, pero no mejoro la recepción.

Al aplicar la segunda forma para colocar los valores de interpolación y decimación, empezando con una interpolación I=512 y un diezmado D=128, de igual forma los datos receptados se seguía escuchando ruido. Se probaron varios valores con la misma relación, pero no se obtuvieron buenos resultados.

Para comprobar que si se producía la modulación digital se realizaron varios ejemplos con modulación QPSK, BPSK y OQPSK, para comprobar si una modulación más robusta produciría mejoras, lo cual no se obtuvo, lo siguiente fue comprobar los diagramas de constelación tanto en la transmisión como en la recepción y obtuvieron los diagramas de la Figura 8, tanto para QPSK y BPSK respectivamente.

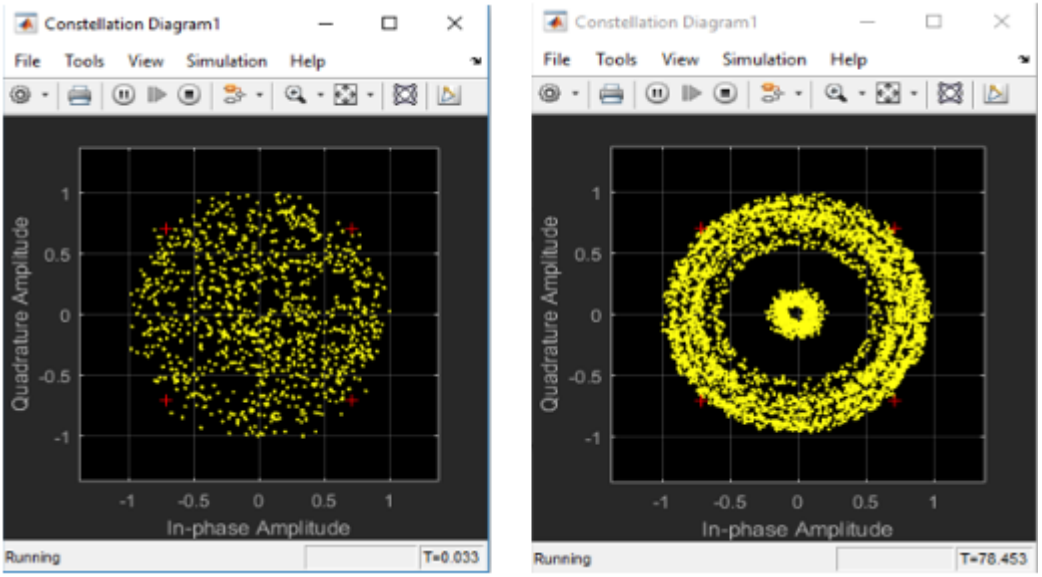


Figura 8. Constelación Recibida QPSK y BPSK

De acuerdo a la Figura 8 se observa desincronización tanto en fase como en cuadratura, por lo que se utiliza el bloque de modulación OQPSK para compensar el desfase de $\pi/4$ de la modulación QPSK, sin embargo los resultados no mejoraron.

Por lo que no se logró implementar la SDR en las tarjetas USRP N210 debido a que las tarjetas hijas no se sincronizaron ya que son distintas en el transmisor se encontraba la *daughterboard* SBX la cual cuenta con sincronización automática y en el receptor la *daughterboard* WBX la cual no cuenta con sincronización, además no se contaba con otras tarjetas hijas. Por lo que se presentaron dos formas de sincronización la primera era por *hardware* mediante el uso de dispositivos externos como: el reloj externo, cable MIMO, GPSDO, entre otros, de los cuales no se cuenta con ninguno, por lo que se optó por compensar el sincronismo mediante *software* utilizando todas las herramientas que presenta Simulink® para la compensación de frecuencia, compensador de portadora, sincronización de símbolos, los

resultados obtenidos en las simulaciones recuperaban fase y cuadratura, pero no de forma óptima ya que aún se presentaban desfases en la constelación, por lo que no se recuperaba toda la cantidad de información enviada, lo que impide que la señal procesa en el receptor sea entendida.

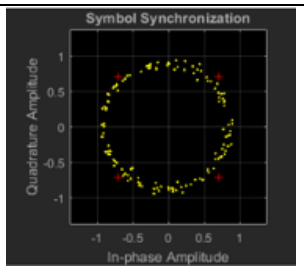
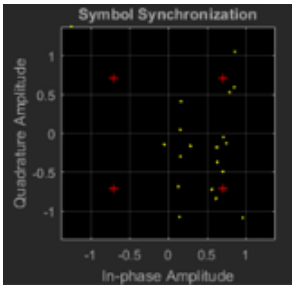
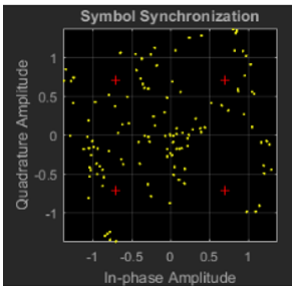
Por lo que se toma como solución el trabajar con las tarjetas USRP-2920 de *National Instruments* debido a que estás tarjetas ya tienen sincronización propia además de que cuentan con cable MIMO para sincronización. Se confirmó la sincronización mediante pruebas de transmisión y recepción utilizando modulación QPSK enviando secuencias PN. Además de los ejemplos propios de LabVIEW® con la USRP-2920.

Para poder trabajar con el modelo SDR de Simulink® se realizó el paso de todos los bloques a archivos de Matlab® .m para poder trabajar con los bloques MathScript de LabVIEW®, en lo que se halló un problema debido a que LabVIEW® no reconoce todas las funciones de Matlab®, entonces se optó por utilizar parte del código de Matlab® en los MathScript hasta donde se reconozcan los comandos de Matlab® y continuar con bloques propios de LabVIEW® para el desarrollo de la SDR desde la codificación de fuente, la codificación de canal, la modulación y el proceso contrario.

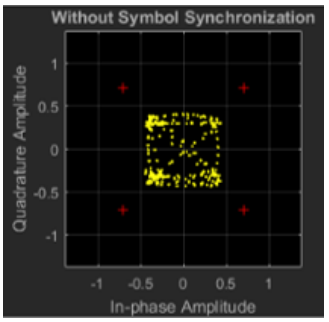
A continuación, se presentan todas las pruebas realizadas en las tarjetas para recuperar el sincronismo en la Tabla 4.

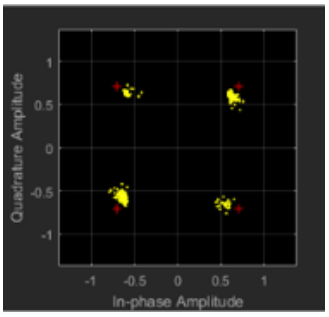
Tabla 4

Pruebas realizadas para sincronización en la USRP N210

Número de Prueba	Bloques usados	Resultado
RX_USRP1	Coseno levantado *Symbol Synchronization* Demodulador QPSK	 No se recuperó la sincronización, porque existe un desfase en la frecuencia.
TX_USRP1	Modulador QPSK Coseno Levantado	
RX_USRP2	Coseno levantado *Symbol Synchronization* Demodulador QPSK	 Al agregar el codificador de fuente en el transmisor, los datos recibidos están totalmente dispersos y no se identifica que tipo de sincronización es necesaria.
TX_USRP2	Códec G711 Modulador QPSK	
RX_USRP3	*Carrier Synchronizer* *Symbol Synchronization* *Coarse Frequency Compensator* Demodulador QPSK	 No se recuperó la sincronización debido a que en el coseno levantado se trabajó con 4 bits por símbolo, y al trabajar con una modulación QPSK se debe trabajar con 2 bits por símbolo.
TX_USRP3	Coseno levantado Modulador Qpsk	

CONTINÚA

<table>
<tr><td>RX_USRP4</td><td>Coseno levantado
Coarse Frequency
Compensator
Carrier Synchronizer
Demodulador QPSK</td><td rowspan="2"> 

Se logró compensar la frecuencia, pero los símbolos no se sincronizan por lo que es necesario el uso del bloque Symbol Synchronizer.</td></tr>
<tr><td>TX_USRP4</td><td>Codec G711
Coseno levantado
Modulador QPSK</td></tr>
<tr><td>RX_USRP6</td><td>Coseno levantado
Coarse Frequency compensator
Carrier Synchronizer
Symbol Synchronizer
I/Q Imbalance compensator
Demodulador QPSK, Decoder
Decodec G.711</td><td rowspan="2"> 

En cada bloque para sincronizar se observa como la señal recibida es compensada en frecuencia, cuadratura en símbolos, con lo que se recupera la sincronización y se visualiza en el diagrama de constelación.</td></tr>
<tr><td>TX_USRP6</td><td>Concatena al mensaje de voz con un preámbulo.
Codec G711
Encoder Convoluacional
QPSK
Coseno levantado</td></tr>
</table>

En las tres plataformas nombradas se realizó las respectivas simulaciones del cifrado en tiempo y frecuencia, para el cifrado DES y AES se lo realizó en Matlab® y LabVIEW®.

3.1 Algoritmos de cifrado en Matlab®

Se realizaron 4 tipos de algoritmos simétricos como son: el cifrado en tiempo, en frecuencia, cifrado DES de 64 bits y el cifrado AES de 128 *bits* para la SDR que fueron realizados en el *Software* Matlab®, dichos algoritmos trabajan con llave simétrica k, la cual es la misma tanto para el transmisor como para el receptor.

3.1.1 Algoritmo de Cifrado en Tiempo

El presente algoritmo se basa en ingresar todo el mensaje a transmitirse, para ser divido en N bloques con lo que se pueden obtener $N!$ combinaciones posibles para cifrar el mensaje, pero no todas las combinaciones son consideradas seguras debido a que son fáciles de descifrar, por lo que se usan dos números que serán parte de la llave de cifrado, estos son k_1 para el cifrado y k_2 para el descifrado, las cuales deben cumplir con los siguientes parámetros:

- Los valores de $K \ll N!$
- Los valores de K deben ser números primos.

Llave de cifrado se representa de forma matemática en la ecuación (11) :

$$s = k_1 \ r \ (mod \ N) \qquad (11)$$

Donde:

- k_1: *Valor de la llave de cifrado.*
- r: Posición inicial del bloque.

- N: Número de bloques en los que se divide al mensaje.
- s: Posición final del bloque.

Para el descifrado se utiliza la llave k_2, para la elección de la llave está debe cumplir con la ecuación (12):

$$k_1\, k_2 (mod\ N) = 1 \qquad (12)$$

Una vez que se verifica el cumplimiento de la ecuación (12) se ejecuta la ecuación (13) para recuperar el mensaje recibido.

$$r = k_2\, s (mod\ N) \qquad (13)$$

A este algoritmo por ser el más básico se le pueden agregar otras operaciones matemáticas a la llave de cifrado para elevar la complejidad del mismo, como se presenta en la ecuación (14).

$$s = k_1\ \ r^3\ (mod\ N) \qquad (14)$$

Donde:

- k_1: Valor de la llave cifrado.
- r: Posición inicial del bloque
- N: Número de bloques en los que se divide el mensaje.
- s: Posición final del bloque

El elevar la complejidad de operaciones matemáticas no implica que el cifrado sea más resistente al criptoanálisis, pero lo que si representa es mayor costo computacional.

3.1.2 Simulación de cifrado en tiempo Matlab®

Una vez desarrollado el algoritmo en Matlab®, se realizaron pruebas directamente con el mensaje de audio para observar el resultado del intercambio de bloques como se presenta en la Figura 9, donde la señal en naranja representa el mensaje original y la señal azul representa el mensaje cifrado, el cual al ser reproducido no se entiende, pero con un valor de N menor a 16, los bloques serán de mayor tamaño por lo que se van a entender ciertas palabras del mensaje, pero si el valor de N es mucho mayor representa un mayor costo computacional.

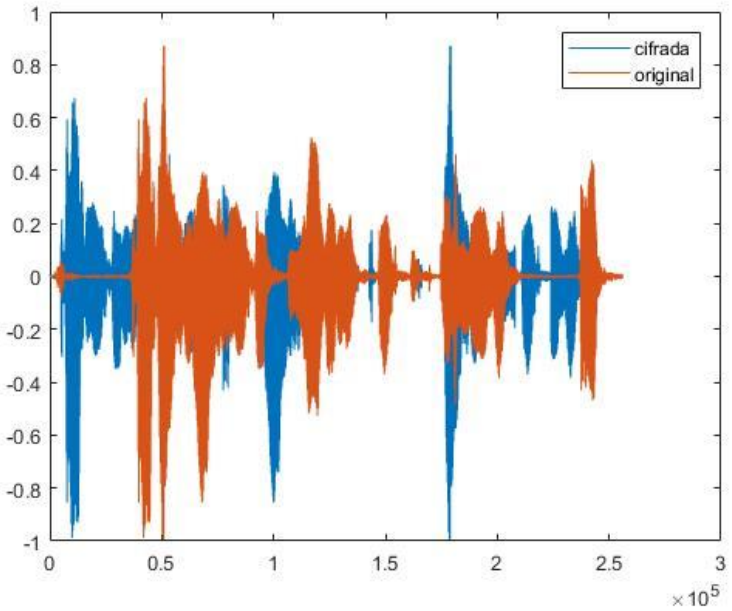


Figura 9. Señal Cifrada en el Tiempo vs señal original

Para elaborar el código de la radio en Matlab® similar al modelo de Simulink® que se presenta en la Figura 10 se lo realizó con el diagrama que se observa en la Figura 11 ya que Matlab® no cuenta con una función para simular el códec G711, es por ello que se utilizó funciones y operaciones en Matlab®.

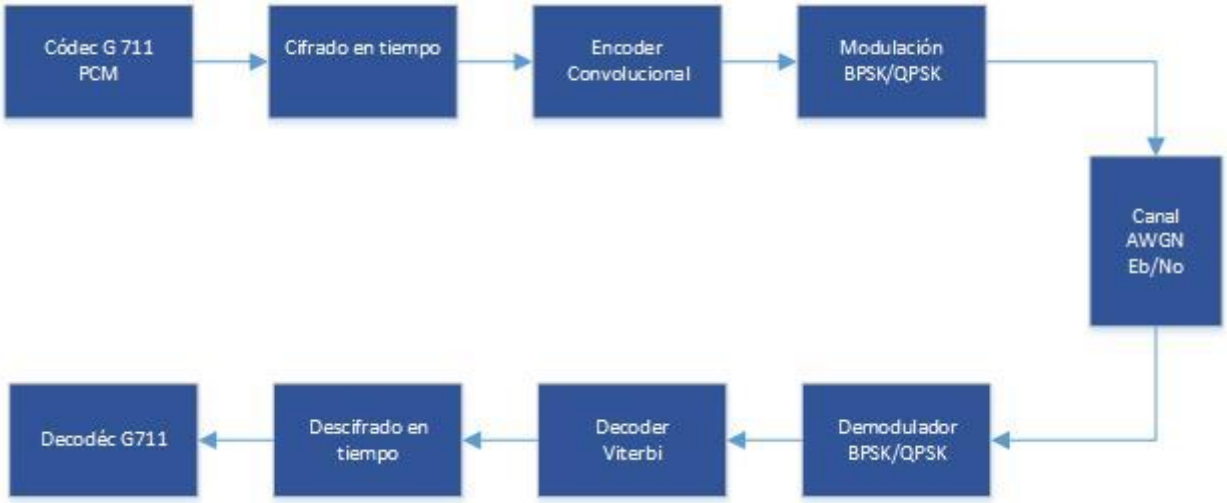


Figura 10. SDR simulada en Simulink® con Cifrado en Tiempo

Figura 11. PCM y codificación

Una vez realizada la codificación de fuente, se llama a la función que contiene el algoritmo de cifrado en tiempo una vez cifrado el mensaje se pasa a la codificación de canal la que trabaja con la función convolucional (que se realiza mediante el uso de un registro de desplazamiento y una lógica combinacional encargada de la suma en módulo 2) la cual necesita del polinomio trellis, para transmitir la señal, se modula con BPSK y QPSK de ahí se transmite por una canal AWGN y

para el receptor se realiza el proceso inverso, pero para la decodificación de canal se utiliza el decoder *viterbi* propio de Matlab®. La señal transmitida y la señal recibida se observa en la Figura 12.

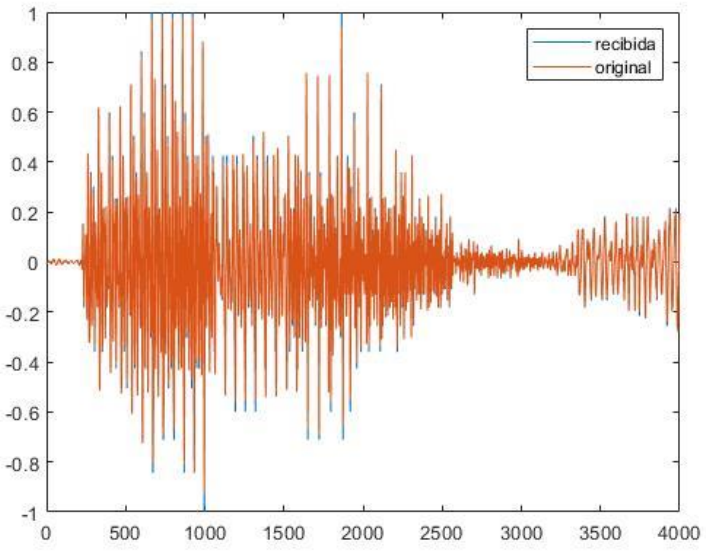


Figura 12. Señal original vs recuperada Cifrado en Tiempo

3.1.2.1 Medidas BER para QPSK y BPSK con cifrado en Tiempo

La tasa de error de bit BER (del inglés *Bit Error Rate*) es un parámetro fundamental ya que permite la determinación de la calidad la señal en el receptor comparándola con la señal original, de forma numérica mediante el cálculo del número de errores.

3.1.2.2 Modulación QPSK cifrado en tiempo

Como se observa en la Figura 13 se tienen las medidas del BER simulado en color azul comparado con el BER teórico en rojo, ambas curvas son decrecientes ya que mientras aumenta la relación Eb/No el número de bits errados disminuye. Y la curva del BER simulado tiene el mismo comportamiento que la curva del BER teórico. Pero la curva del BER simulado es visible

hasta 10dB ya que la simulación es ideal y para una relación Eb/No superior a 10dB no existen errores es decir que el BER es 0, mientras que en el BER teórico se observa que el número de errores continúa disminuyendo.

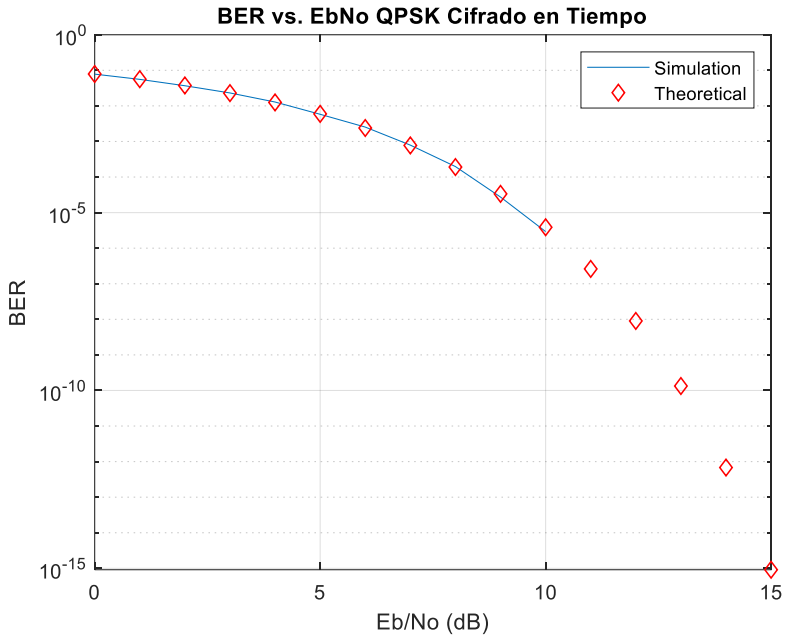


Figura 13. BER señal de voz modulada con QPSK y con cifrado en tiempo

En la Figura 14 se puede observar los valores del BER simulado vs teórico donde se observa que la curva del BER simulado sigue el comportamiento de la curva del BER teórico, pero el BER simulado en Matlab® presenta un menor número de bits errados debido a que el canal es ideal y la modulación BPSK es más robusta. El BER simulado en 11dB ya no presenta bits errados debido a que el canal AWGN es ideal.

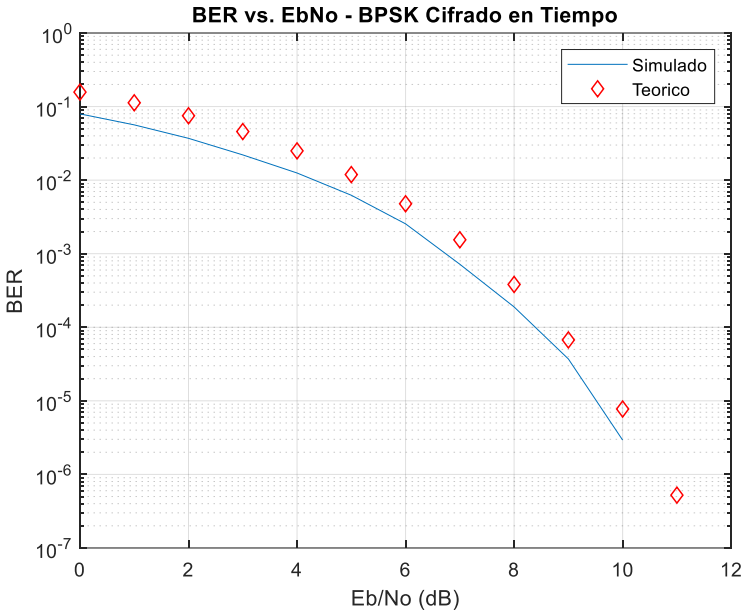


Figura 14. BER señal de voz modulada con BPSK y con cifrado en tiempo.

En la Figura 15 se puede observar los valores del BER simulado versus el teórico sin el algoritmo de cifrado y con modulación BPSK, donde se observa que la curva del BER simulado acompaña a la curva del BER teórico, se observa que el resultado es similar al de la Figura 14 es decir que el cifrado no afecta al número de bits errados.

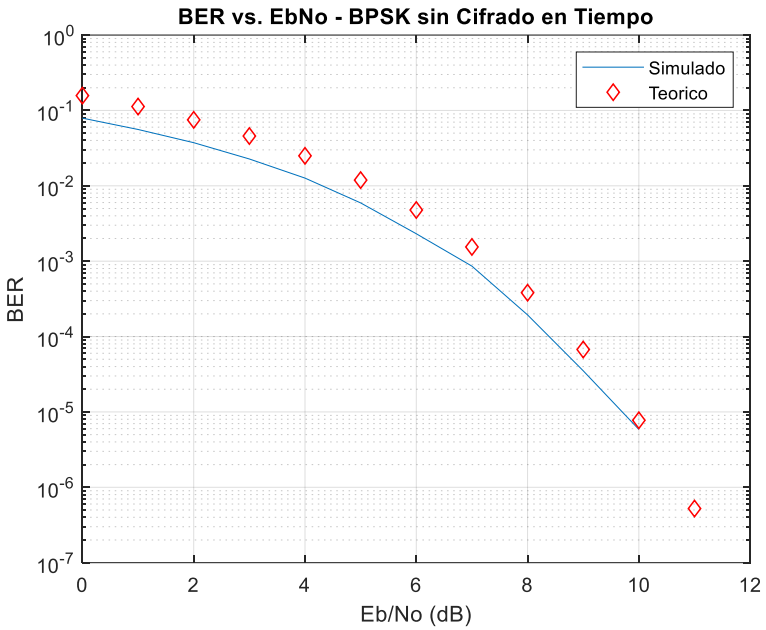


Figura 15. BER modulado con BPSK sin cifrado en tiempo

3.1.2.3 Medidas de Distorsión MBSD, QPSK y BPSK Cifrado en Tiempo

Para medir el nivel de distorsión de la señal recibida, se utiliza el método MBSD desarrollado por (Yang & Yantorno, 1999), los valores obtenidos se presentan en la Tabla 5 donde se realizó la prueba para una señal de voz de 1 segundo muestreada a 8kHz, con 160 muestras por ventana y con valores de relación Eb/No de 0, 3, 6, 9, 12, 15 y 20 además de la modulación QPSK y BPSK y el cifrado en tiempo, en donde se puede verificar que los valores MBSD se hacen más pequeños mientras mayor es la relación Eb/No, la cual se estabiliza en el valor de 0.0652 con una relación Eb/No de 9dB para modualción QPSK y 6dB para modulación BPSK, en general los valores MBSD con BPSK son menores a los valores MBSD con modulación QPSK ya que para una relación Eb/No de 0 dB se obtuvo un valor MBSD de 0.477, en cambio para la modulación BPSK se obtuvo un valor de 0.2416, el nivel de distorsión es menor para la modulación BPSK, lo que representa un nivel de distorsión bajo concluyendo una buena recuperación de la señal.

Tabla 5

Valores MBSD con modulación QPSK y BPSK en Matlab

Eb/No	MBSD QPSK	MBSD BPSK
0	0.477	0.2416
3	0.3059	0.0749
6	0.0823	0.0652
9	0.0652	0.0652
12	0.0652	0.0652
15	0.0652	0.0652
18	0.0652	0.0652
20	0.0652	0.0652

En la Figura 16 se comparan los valores MBSD con modulación QPSK y BPSK donde se puede visualizar que la modulación BPSK se estabiliza más rápido que la la modulación QPSK.

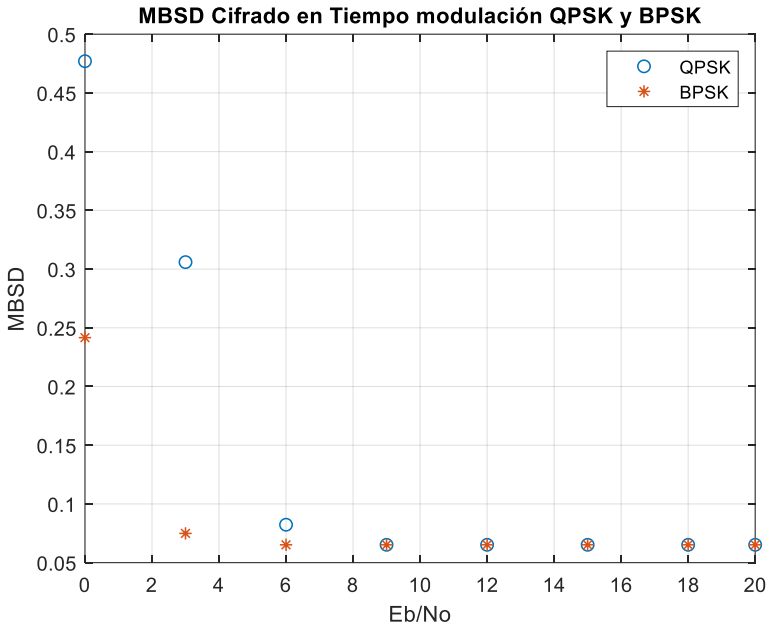


Figura 16. MBSD Cifrado en Tiempo con modulación QPSK y BPSK en Matlab®

3.1.3 Algoritmo de cifrado en tiempo en el *Software* Simulink®

Este algoritmo se trabajó como una función de Matlab® la cuál fue probada en el *Software* Simulink® mediante el uso del bloque *S-Function*, se agregó el bloque de cifrado en tiempo a la SDR desarrollada en el mismo *Software* como se observa en la Figura 17, se usó el codificador G711, para el bloque de cifrado se debe tener cuidado con el tiempo de simulación ya que este procesa la señal de entrada acorde al tiempo que se coloque en el bloque *S-Function*, su resultado fue verificable mediante audición ya que el mensaje se recupera, además se hizo uso del BER, comparando los datos cifrados en la transmisión y los datos receptados.

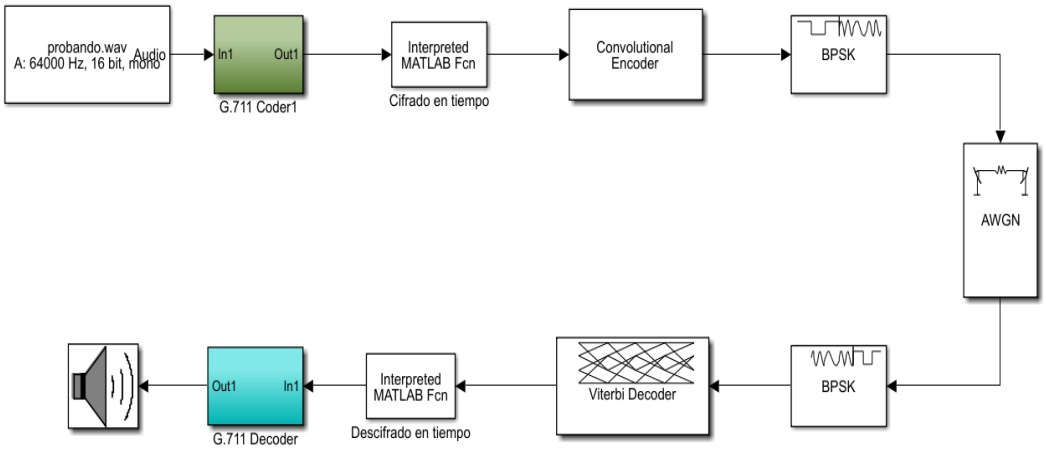


Figura 17. SDR en Simulink® Cifrado en Tiempo

La señal de entrada comparada con la señal de salida en la Figura 18, en las primeras simulaciones existe una saturación del sistema por lo que al comparar la señal de entrada versus la señal de salida se observa que la señal de salida está recortada, por lo que se agrega un atenuador en la transmisión para evitar que se sature el sistema.

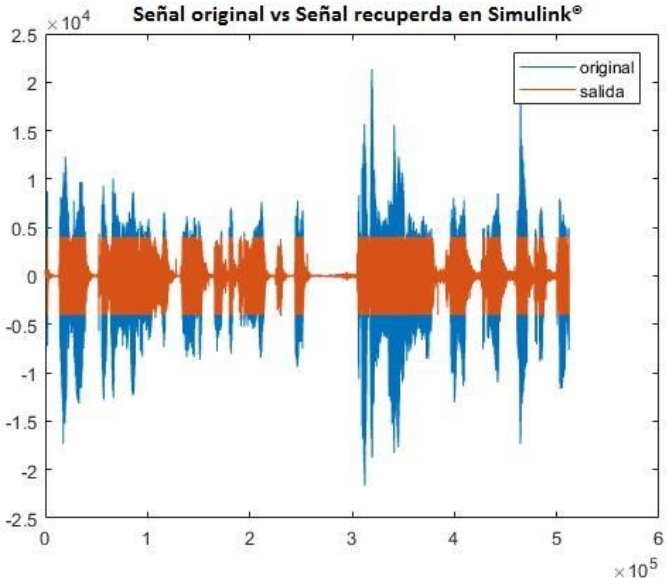


Figura 18. Señal transmitida vs Señal recuperada

3.1.3.1 Medidas BER con modulación QPSK y BPSK cifrado en Tiempo en Simulink®

El BER para una señal de voz muestreada a 64kHz con el bloque de cifrado en tiempo se varia la relación Eb/No desde 3 a 27dB, en la Figura 19 se observa el resultado de la simulación en color naranja y la señal original en azul, en donde el número de bits errados es menor mientras mayor es la relación Eb/No.

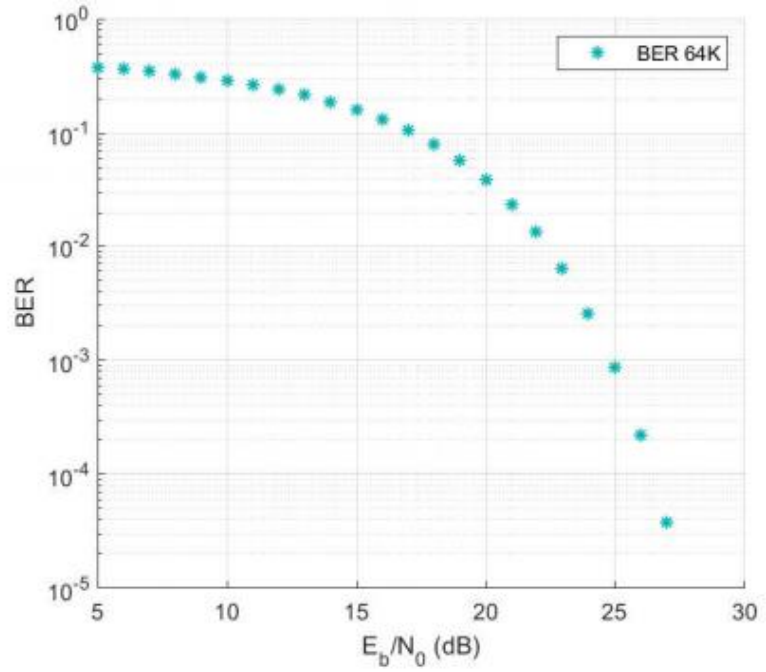


Figura 19. BER con una relación Eb/No de 3 a 27dB

En la Figura 20 se realiza el análisis de 1 a 18dB, en donde se observa que la simulación presenta retrasos por lo que los 2 primeros valores de BER son cero y en 3dB aparece el primer valor BER, pero hasta los 18dB existe una gran pérdida de bits debido a la gran cantidad de datos que se están transmitiendo ya que es una señal muestreada a 64kHz. El valor BER es 0.158 y el número de bits es de 640000. El BER teórico cae más rápido es decir presenta menos bits errados, el BER teórico presenta un mayor número de bits errados debido al procesamiento de cada bloque.

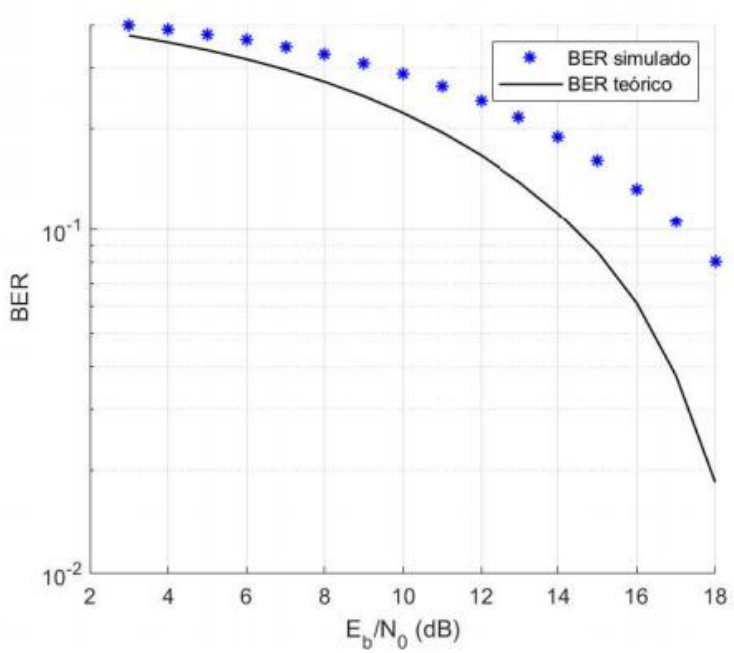


Figura 20. BER simulado vs BER teórico con QPSK y Eb/No de 1 a 18dB en Simulink®

La Figura 21 muestra el BER versus la relación Eb/No teórico y simulado para una señal muestreada a 8kHz por lo que es claro que mientras mayor es la relación Eb/No los bits errados van disminuyendo, en 20dB el BER se estabiliza en 10^{-2}, es decir que si se continúa aumentan la relación Eb/No los bits errados seguirán siendo la misma cantidad. La curva del BER simulado sigue el comportamiento del BER teórico hasta los 16dB ya que ahí el BER disminuye rápido mientras que el BER simulado tiene un mayor número de bits errados.

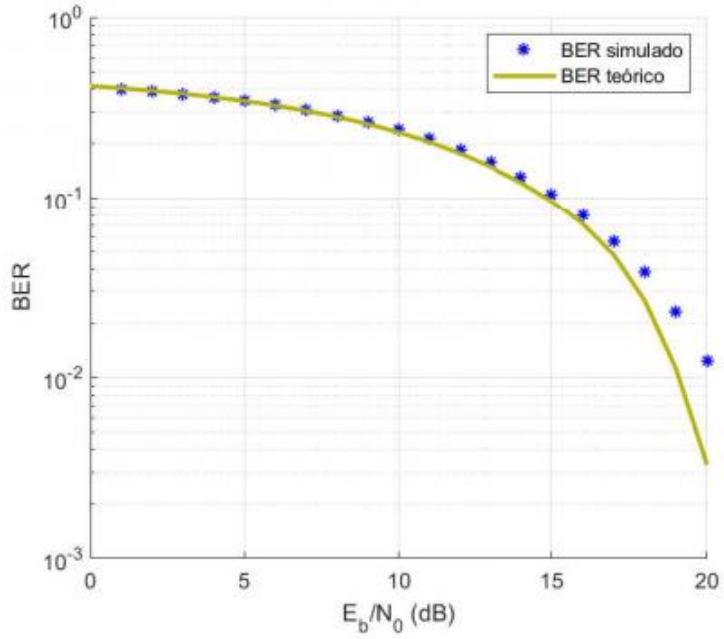


Figura 21. BER simulado vs BER teórico con BPSK y Eb/No de 0 a 20dB en Simulink®

3.1.3.2 Medidas de Distorsión MBSD, QPSK y BPSK Cifrado en Tiempo Simulink®

En la Tabla 6 se observan las medidas de distorsión para el cifrado en tiempo con modulación QPSK una relación Eb/No que va de 0dB a 20dB, donde los valores MBSD va disminuyendo mientras la relación Eb/No aumenta y en 15dB la medida de distorsión se estabiliza en 0.5994 ya que si la relación Eb/No aumenta la medida de distorsión no disminuye.

En cambio, para la modulación BPSK por ser más robusta se estabiliza en 6dB con un valor MBSD de 0.5994 que es el mismo valor de estabilización para QPSK. Pero los valores de 0dB a 6dB no son considerados como válidos ya que no hay una buena percepción audible es decir que no se entiende el mensaje.

Tabla 6

Valores MBSD para modulación QPSK con cifrado en Tiempo en Simulink®

Eb/No	MBSD QPSK	MBSD BPSK
0	1.2489	0.8242
3	1.1988	0.7492
6	1.1988	0.5994
9	0.8991	0.5994
12	0.8242	0.5994
15	0.5994	0.5994
18	0.5994	0.5994
20	0.5994	0.5994

En la Figura 22 se observan los resultados MSBD para la modulación QPSK y BPSK. Se vizualiza que la modulación BPSK se estabiliza primero.

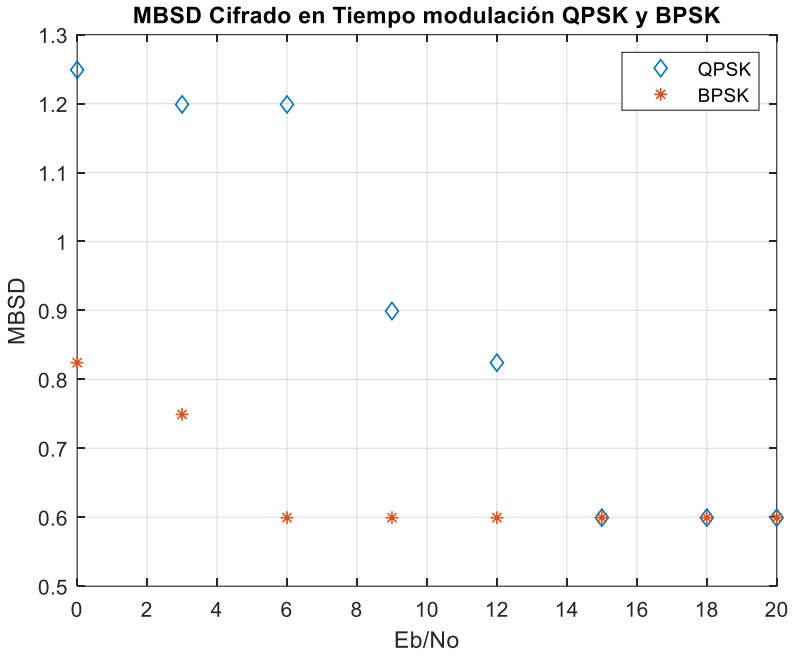


Figura 22. MBSD Cifrado en Tiempo con modulación QPSK y BPSK en Simulink®

3.1.4 Algoritmo de cifrado en tiempo en el *Software* LabVIEW®

Las primeras pruebas se basaron en el funcionamiento de la SDR, para lo cual se realizó el desarrollo de la SDR con las mismas características de la SDR de Simulink® de la tesis (Capt. Paredes & Angulo, 2011).

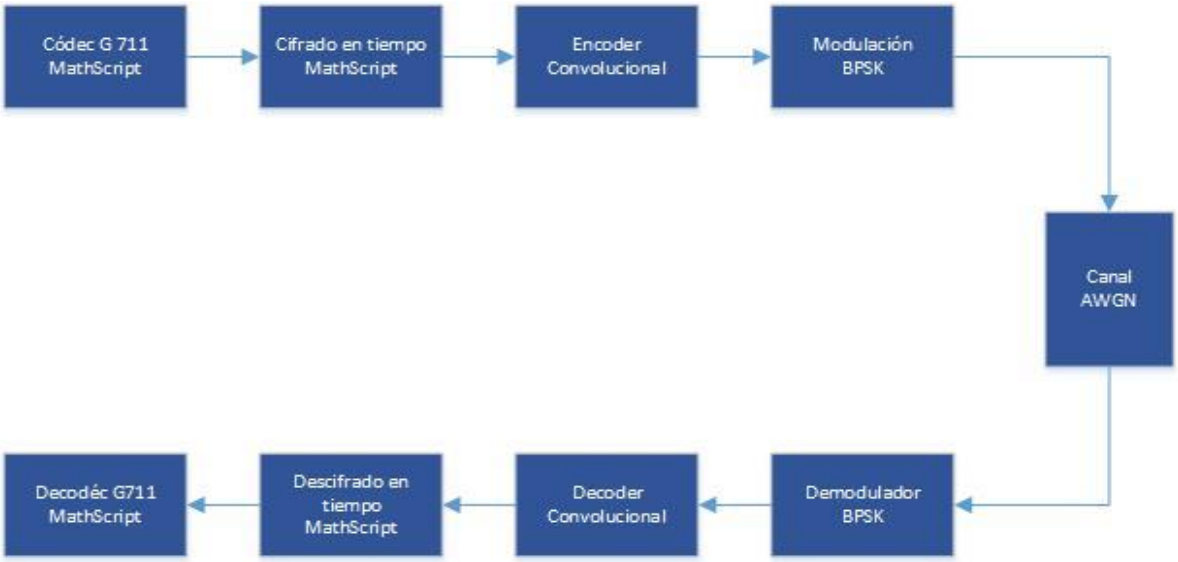


Figura 23. Diagrama de bloques de la SDR en LabVIEW® Cifrado en Tiempo

Para la realización del códec G711 en un MathScript se implementó mediante las fórmulas para la compresión y expansión con la Ley A, ecuación (15) y ecuación (16) respectivamente.

$$F(x) = sgn(x) * \begin{cases} \dfrac{A|x|}{1+Ln(A)} & si\ |x| < \dfrac{1}{A} \\ \dfrac{1+Ln(A*|x|)}{1+Ln(A)} & si\dfrac{1}{A} \leq |x| \leq 1 \end{cases} \qquad (15)$$

$$F(x)^{-1} = sgn(y) * \begin{cases} \dfrac{|y|\,(1+\ln(A))}{A} & si\ |y| < \dfrac{1}{1+Ln(A)} \\ \dfrac{\exp(|y|(1+Ln(A))-1)}{A} & si\ \dfrac{1}{1+Ln(A)} \leq |y| < 1 \end{cases} \quad (16)$$

Una vez realizada la compresión de la señal de voz se realiza la cuantificación y codificación con 8 bits, con que se obtiene la codificación de fuente a continuación, se agrega el algoritmo de cifrado en tiempo en el MathScript de LabVIEW®, como las funciones del encoder convolucional y la función trellis no son reconocidos en el MathScript, se utilizó un bloque propio de LabVIEW® para la codificación de canal, una vez realizada la codificación de canal se realiza la modulación QPSK de igual forma utilizando el bloque de PSK de LabVIEW®.

Para la transmisión se simula un canal AWGN (del inglés *Additive White Gaussian Noise*) controlando el valor de la relación Eb/No. En el receptor se realiza el proceso contrario partiendo de la demodulación PSK, luego se realiza la decodificación usando el decoder convolucional con una tasa de 1/2, con los datos decodificados se realiza el proceso de descifrado, se decodifica y re cuantiza la señal para finalmente ser expandida utilizando la fórmula de la ley A.

3.1.4.1 Medidas de Distorsión MBSD, QPSK y BPSK Cifrado en Tiempo en LabVIEW®

Como se observa en la Tabla 7 los valores de distorsión de la señal transmitida comparada con la señal recibida en el *software* LabVIEW® donde los valores de distorsión son altos para una relación Eb/No de 0dB alcanzando un valor de 3.5744 pero mientras esta relación va aumentando los valores MBSD van disminuyendo, hasta el valor de 0.0462 que es el valor donde se estabiliza, en 6dB, aunque se siga aumentando la relación Eb/No el valor de distorsión de la señal recibida sigue siendo 0.0462, y este valor nos indica que el sistema de la SDR permite recuperar la señal

enviada con un bajo nivel de distorsión. Algo similar sucede para el mismo sistema, pero con una modulación BPSK, donde se observa que el valor de distorsión MBSD se estabiliza en 3dB, debido a que la modulación BPSK es más robusta frente al ruido, sin embargo, para 0dB presenta mayor distorsión que para la modulación QPSK, al ser una señal totalmente distorsionada por el ruido se toman los dos valores como válidos, sin importar la modulación.

Tabla 7

Valores MBSD con modulación QPSK y BPSK, cifrado en Tiempo en LabVIEW®

Eb/No	**MBSD QPSK**	**MBSD BPSK**
0	3.5744	4.476
3	3.3801	0.0462
6	0.0462	0.0462
9	0.0462	0.0462
12	0.0462	0.0462
15	0.0462	0.0462
18	0.0462	0.0462
20	0.0462	0.0462

En la Figura 24 se observan los resultados MBSD para la modulación QPSK y BPSK con cifrado en frecuencia en LabVIEW®, la modulación BPSK se estabiliza primero, presentando una distorsión mínima.

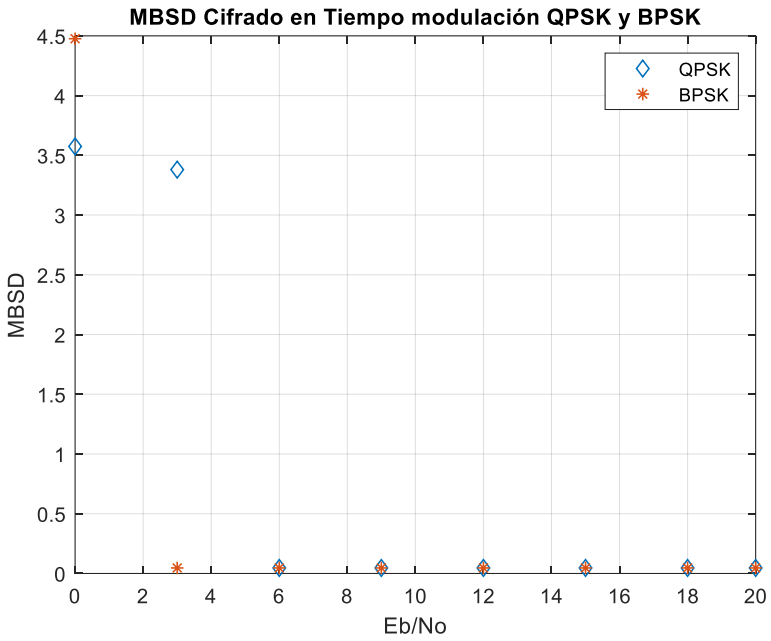


Figura 24. MBSD Cifrado en Tiempo con modulación QPSK y BPSK en LabVIEW®

3.1.4.2 Medidas BER vs Eb/No con QPSK y BPSK Cifrado Tiempo en LabVIEW®

Los resultados observados en la Figura 25 son tomados directamente de los 128000 bits transmitidos comparados con los bits errados mediante el uso de la ecuación (17).

Los resultados obtenidos muestran un comportamiento similar a los valores MBSD ya que estos se estabilizan en aproximadamente 6dB.

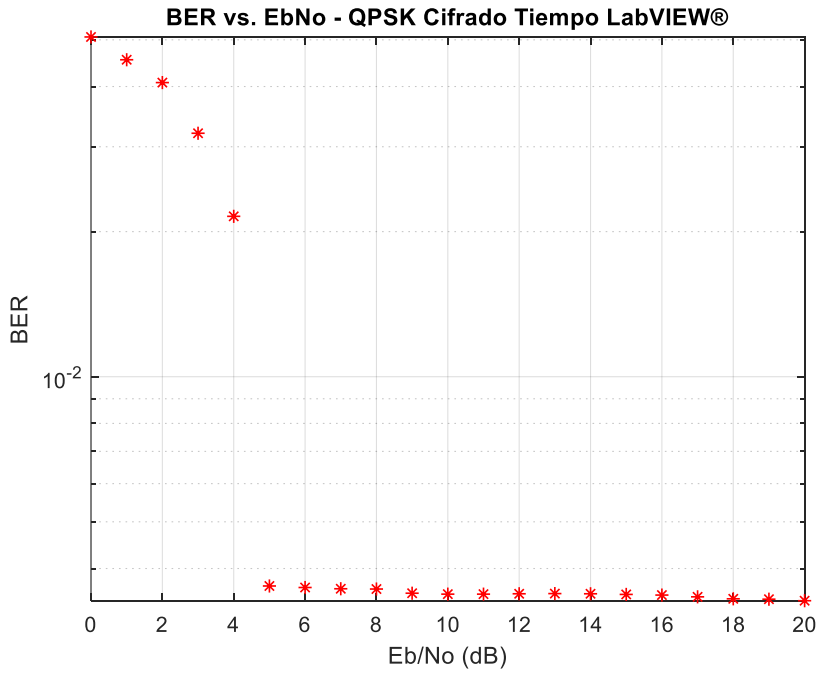


Figura 25. BER vs Eb/No modulación QPSK Cifrado en Tiempo LabVIEW®

$$BER = \frac{Número\ de\ errores}{Número\ de\ bits\ \ transmitidos} \qquad (17)$$

En la Figura 26 se observan los valores BER que se obtuvieron usando la ecuación (17). En la que se observa que los bits errados van disminuyendo cuando se aumenta la relación Eb/No, pero se estabilizan aproximadamente en 3dB, como sucede con los valores MBSD, pero los bits errados si existen pequeños cambios que por la escala de la gráfica no son visibles, en cambio los valores MBSD si se estabilizan en un valor fijo.

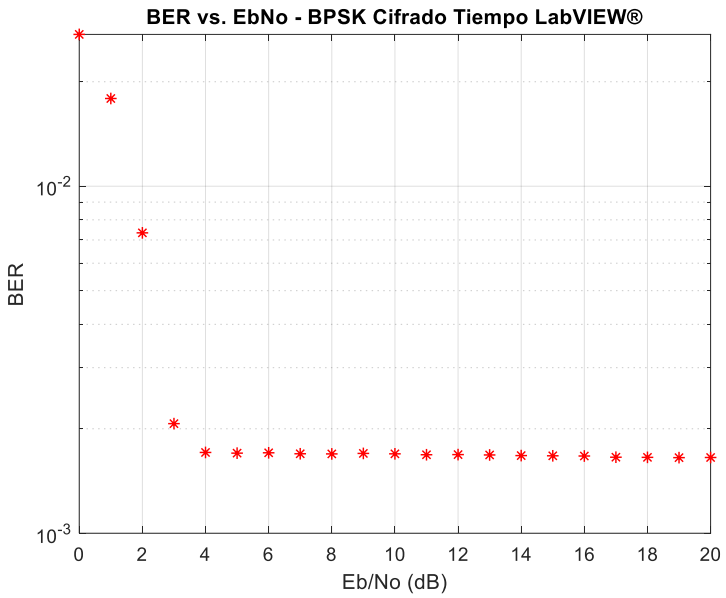


Figura 26. BER vs Eb/No BPSK Cifrado en Tiempo LabVIEW®

3.1.5 Algoritmo de cifrado en frecuencia

El algoritmo de CSI en el dominio de la frecuencia fue elaborado de manera similar al CSI en el dominio del tiempo, trabaja con la misma división de datos en N bloques y el intercambio de los mismos mediante las llaves k1 y k2, con la diferencia que al momento de ingresar los datos a los bloques de cifrado y descifrado fueron transformados al dominio de la frecuencia mediante la transformada de Fourier.

3.1.5.1 Simulación del CSI-F Matlab®

Elaborado el algoritmo en Matlab® se implemento el codigo en la SDR ilustrada en la Figura 10 para realizar las medidas de calidad de audio, obteniendo como resultado en la Figura 27 la señal transmitida y señal recuperada por la SDR utilizando el CSI-F.

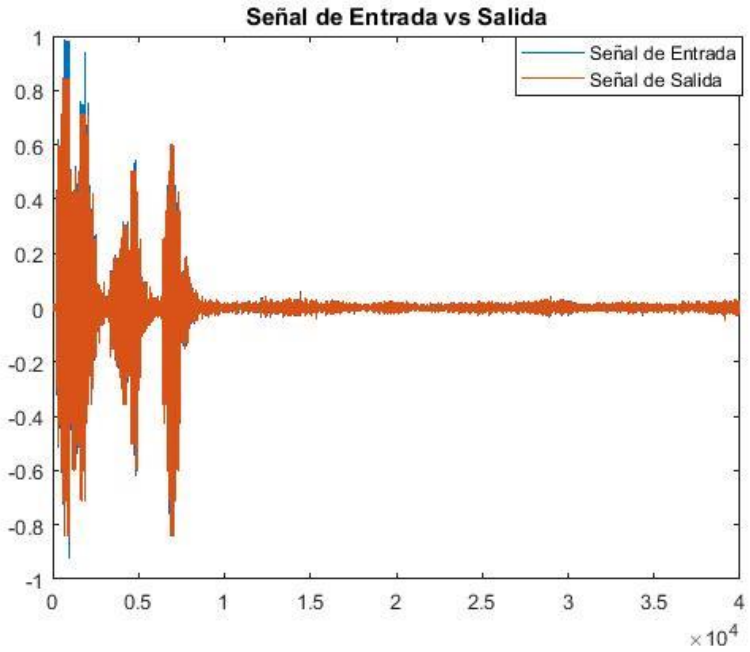


Figura 27. Señal original vs recuperada Cifrado en Frecuencia

3.1.5.2 Medidas BER para modulación QPSK y BPSK con cifrado en Frecuencia en Matlab®

El BER es escencial para analizar la calidad de mensaje que se esta receptando, ya que realiza pruebas en funcion de la relacion Eb/No, midiendo el rendimiento de los sistemas entre la señal transmitida y la señal recuperada.

Como se puede observar en la Figura 28 en relación al BER teórico el sistema a los 12 dB presenta una estabilización en sus datos, dejando de disminuir el error conforme aumenta la relación señal a ruido del canal.

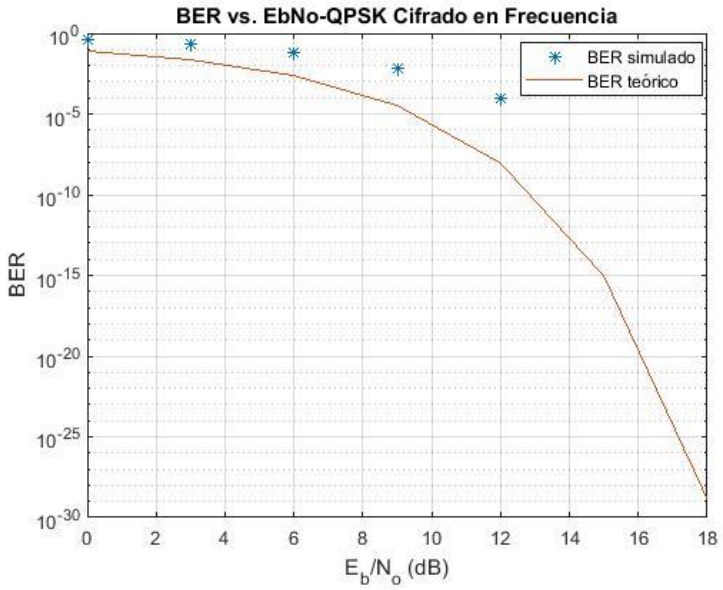


Figura 28. BER con modulación QPSK y con cifrado en frecuencia en Matlab®

En la Figura 29 se observa el BER simulado con modulación BPSK en relación al cálculo teórico del BER para BPSK, obteniendo el comportamiento esperado para esta modulación ya que es más robusta que QPSK por ende tendrá menos perdida de datos en la transmisión.

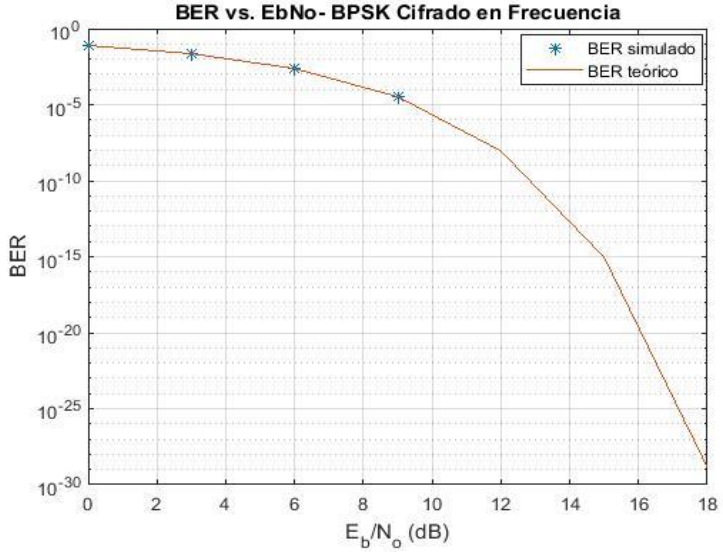


Figura 29. BER con modulación BPSK y con cifrado de frecuencia en Matlab®

Al igual que en el cifrado del tiempo se realizó las medidas de distorsión de la señal recibida, con el método MBSD, los datos obtenidos en la Tabla 8 se realizaron a 20 mili segundos muestreada a 8kHz, y con relación Eb/No de 0, 3, 6, 9, 15, 18 y 20dB con modulación QPSK y BPSK, con el cifrado de la frecuencia.

Tabla 8

Valores MBSD con modulación QPSK y BPSK en Matlab®

Eb/No	MBSD QPSK	MBSD BPSK
0	0,6333	0,6527
3	0,6384	0,6442
6	0,5528	0,5974
9	0,6335	0,6219
12	0,6131	0,6131
15	0,6131	0,6131
18	0,6131	0,6131
20	0,6333	0,6527

En la Figura 30 se puenden observar los valores de la Tabla 8 que reflejan las medidas de distorcion en las modulaciones QPSK y BPSK con el cifrado en frecuencia simulado en Matlab®, como se esperaba los valores del BPSK son menores a los de QPSK debido a su robustez.

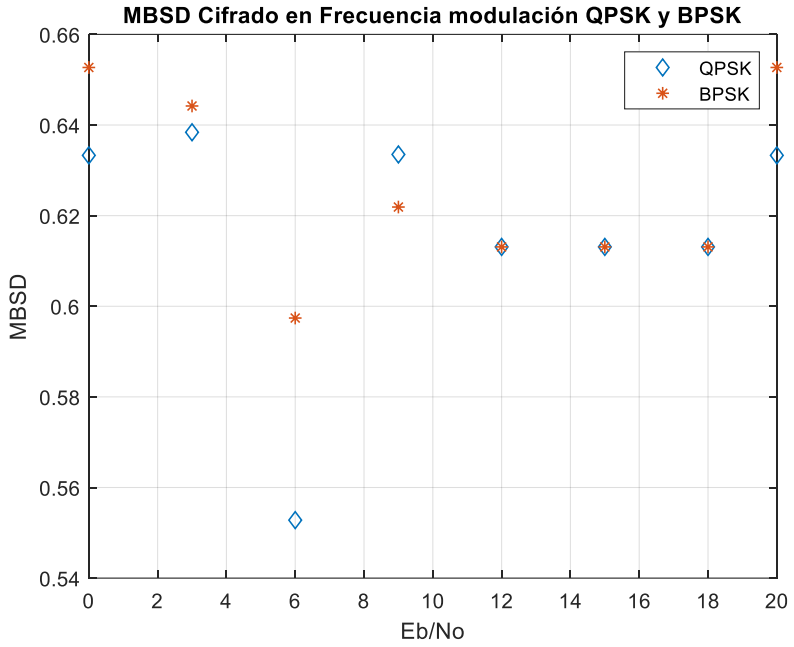


Figura 30. MBSD cifrado en Tiempo con modulación QPSK y BPSK en Matlab®

3.1.5.3 Simulación del CSI-F en el *software* Simulink®

En el modelo de Simulink® se trabajó con los bloques detallados en la Figura 16, agregando el código del cifrado en el bloque de *S-Function* el cual permite a una función de Matlab® trabajar directamente en Simulink®.

En la Figura 31 se puede observar la medida del BER con una modulación QPSK en relación a la medida teórica.

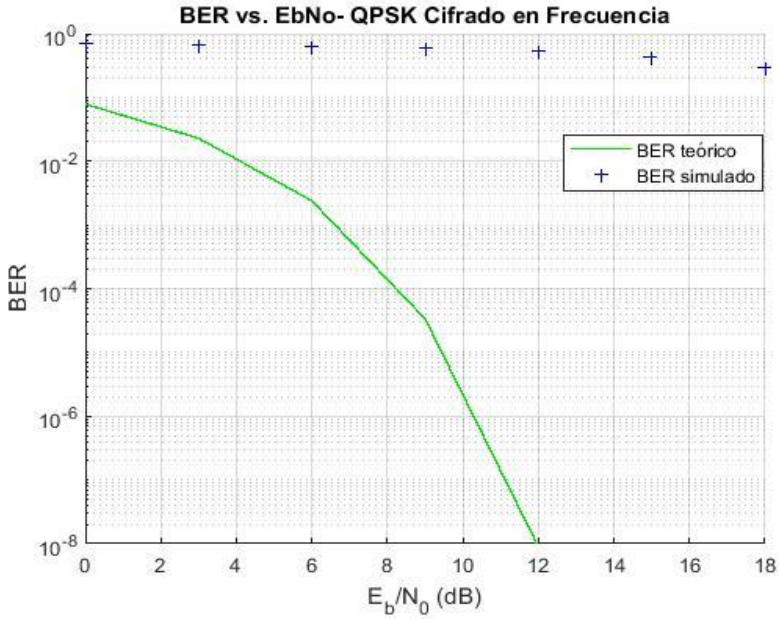


Figura 31. BER señal de voz modulada con QPSK, con cifrado de frecuencia en Simulink®

Como se puede observar en la Figura 32 se calculó el BER de la transmisión con el CSI-F utilizando una modulación BPSK la misma que se realizó de 0 a 20 dB teniendo una gran pérdida de bits en Simulink®.

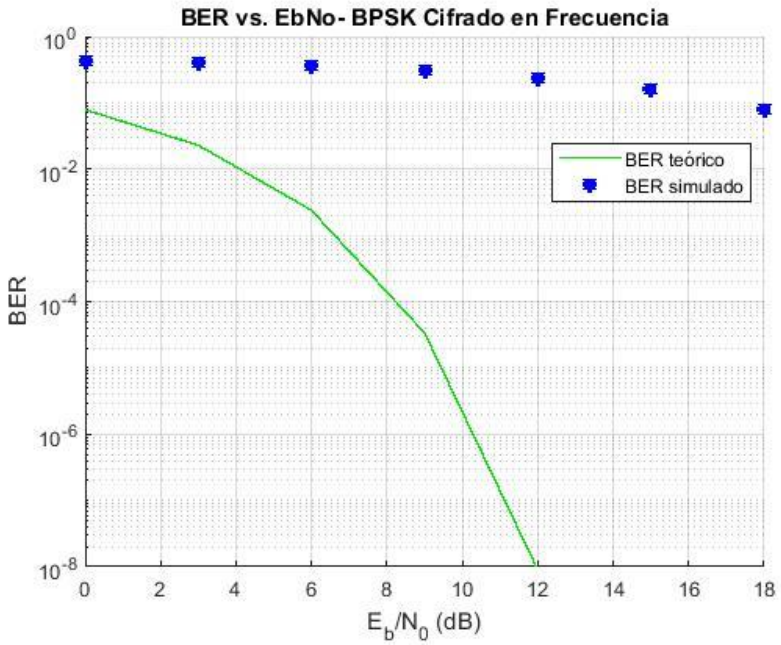


Figura 32. BER con modulación BPSK y cifrado de frecuencia en Simulink®

3.1.5.4 Valores MBSD con modulación QPSK CSI-F Simulink® vs LabVIEW®

En la Tabla 9 se muestran los datos del cálculo MBSD resultantes de la simulación del cifrado en frecuencia utilizando Simulink® y LabVIEW® con modulación QPSK.

Tabla 9

Valores MBSD con modulación QPSK en Simulink® y LabVIEW®

Eb/No	MBSD SIMULINK	MBSD LABVIEW
0	1.5930	1.2414
3	1.4178	1.2539
6	1.5192	1.2725
9	1.5485	1.3046
12	1.9683	1.4916
15	1.7316	1.7305
18	1.7545	1.7998
20	1.5930	1.2414

La Figura 33 muestra el comportamiento de los valores del MBSD de la Tabla 13 en las simulaciones de Simulink® y LabVIEW®, teniendo mayor perdida de datos en Simulink® como se observa en la grafica.

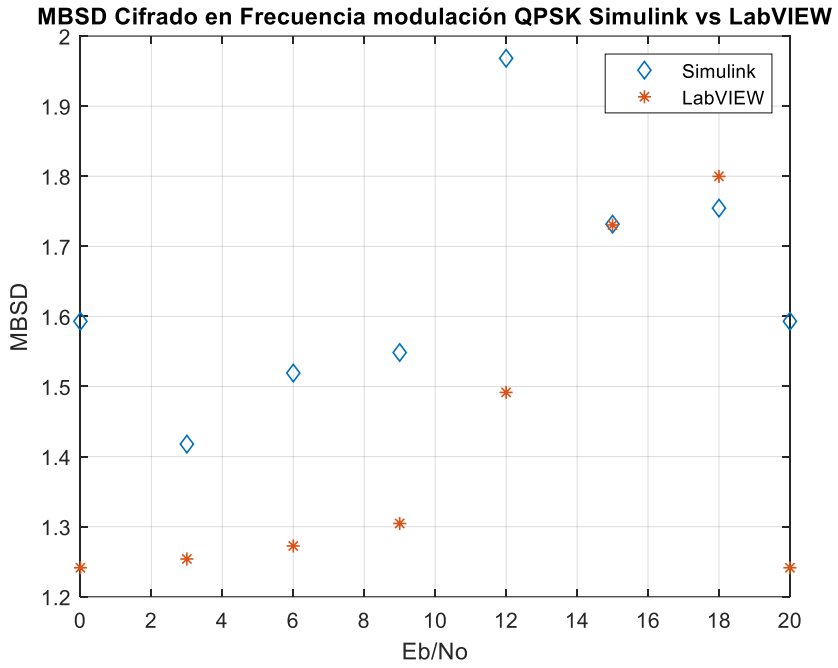


Figura 33. MBSD QPSK con cifrado en frecuencia en Simulink® y LabVIEW®

3.1.5.5 Valores MBSD con modulación BPSK CSI-F Simulink® vs LabVIEW®

Utilizando el código de CSI-F se calculó el MBSD en el programa de Simulink® y LabVIEW® con modulación BPSK para 20ms como se muestra en la Tabla 10.

Tabla 10

Valores MBSD con modulación BPSK en Simulink® y LabVIEW®

Eb/No	MBSD LABVIEW	MBSD SIMULINK
0	1,2243	1,4600
3	1,213	1,3444
6	1,2622	1,4266
9	1,2383	1,5727

CONTINÚA

12	1,5053	1,6754
15	1,7193	1,7298
18	1,794	1,7540
20	1,2243	1,4600

En la Figura 34 se muestra los valores ilustrados de la Tabla 10 los mismos que representan los datos tomados de las simulaciones en Simulink® y LabVIEW® con modulación BPSK.

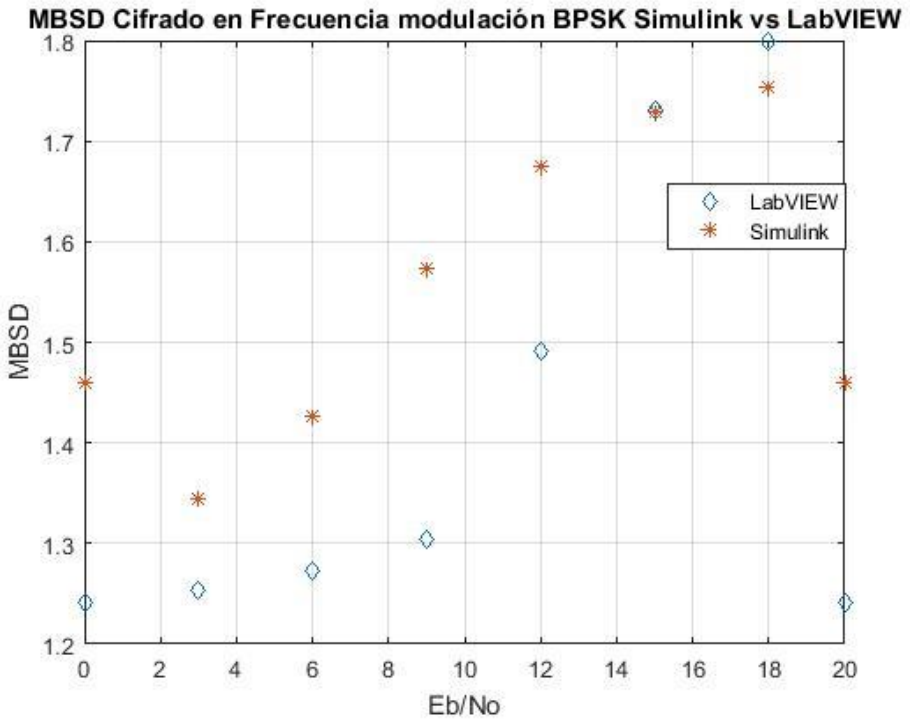


Figura 34. MBSD BPSK con cifrado en frecuencia en Simulink® y LabVIEW®

3.1.6 Algoritmo de Cifrado DES

Para la aplicación del algoritmo DES lo que se requiere es tener el mensaje en bloques de 64 bits y trabajar con una clave de 64bits, el primer paso es trabajar con la clave en grupos de 8 bits, en cada grupo se comprueba la paridad mediante sumas, si la suma es impar en cada grupo la

clave es válida con paridad impar, ahora se tienen 56 bits sin paridad a los que se les aplicará los siguientes pasos:

Primero se le aplicó una permutación inicial.

Segundo, con los datos ya permutados se los divide en dos partes iguales, teniendo una parte a la derecha y otra a la izquierda, con estos datos se procede a realizar 16 cambios cíclicos aplicando la ecuación (18) como se observa en la Figura 35.

$$\ell_i = \begin{cases} 1, & si\ i = 1,2,9,16 \\ 2, & otros \end{cases} \qquad (18)$$

Tercero una vez aplicadas las 16 rondas cíclicas para la generación de llaves, se procede a unir los datos resultantes tanto de la parte izquierda como de la derecha se realiza una premutación final y de los 56 bits resultantes se toman 48 bits que serán las llaves para cada una de las rondas de Feistel aplicadas a los datos de información como se observa en la Figura 36.

Para trabajar con el mensaje se lo dividió en grupos de 64 bits a los que se les aplicó la Tabla 1 conocida como PI (permutación inicial) y a continuación se lo divide en dos partes de 32 bits cada una y para poder trabajar con la llave generada de forma paralela, se realiza una expansión de la cadena de 32 bits del mensaje en 48 bits que es el tamaño de la llave ki, lo siguiente es aplicar el Cifrado de Feistel de 16 rondas, y la respectiva llave k^i de 48 bits, la cual proviene de la clave K.

Una vez realizada la expansión del mensaje tanto para la parte izquierda I y la derecha D, a los 48 bits de cada lado se los segmenta en 8 bloques de 6 bits, a los que se les aplicará el juego de

rondas del algoritmo Feistel usando las ecuaciones (19) y (20) como se observa en la Figura 37.

$$I_i = D_{i-1} \quad (19)$$

$$D_i = I_{i-1} \oplus f_{k^i}(D_{i-1}) \quad (20)$$

Una vez realizada las 16 rondas de Feistel como se observa en la Figura 36 se obtendrá el mensaje de 32 bits cifrado tanto al lado izquierdo como al lado derecho los cuales serán unidos para formar un bloque de 64 bits.

$$Rondas\ Feistel = (D_{NR}, I_{NR}) \quad (21)$$

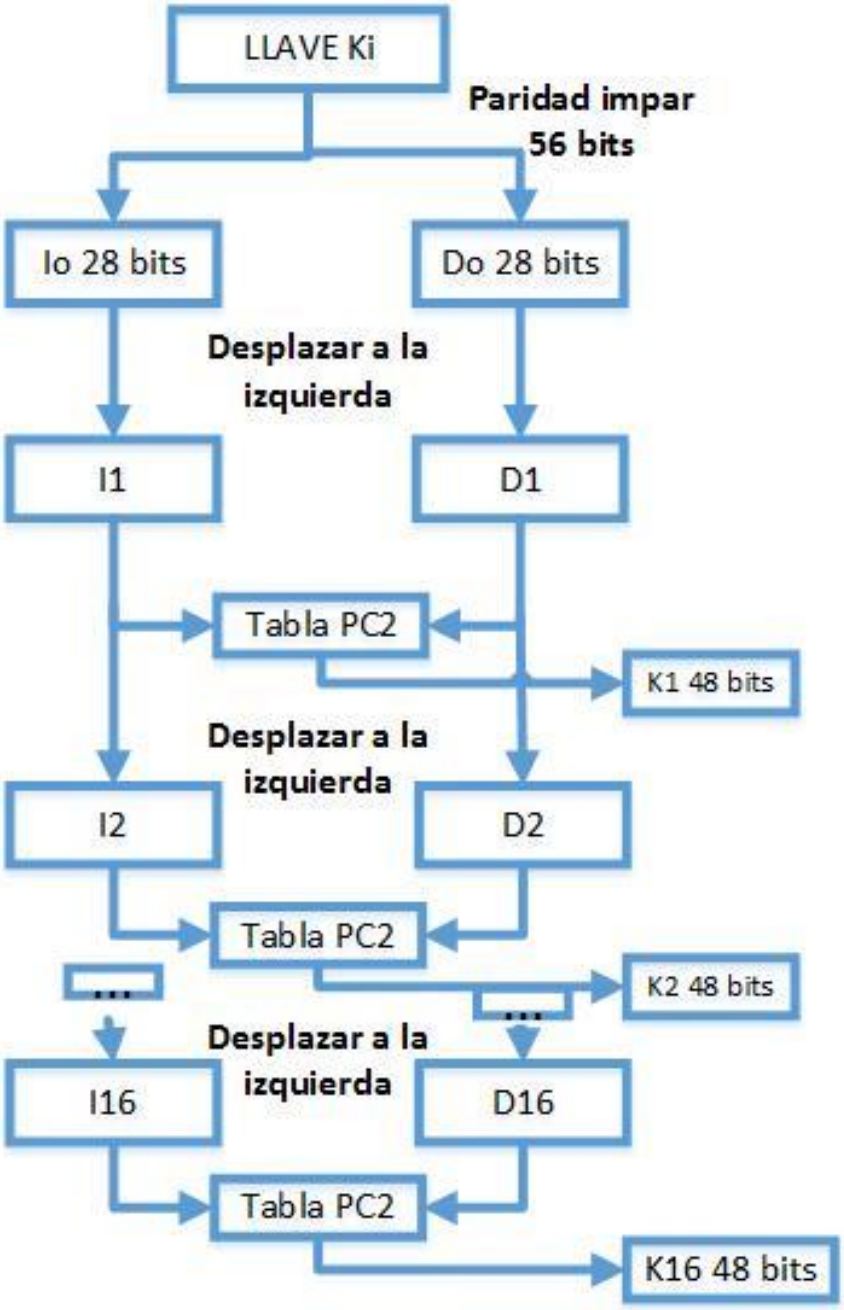


Figura 35. Algoritmo para generar la llave K

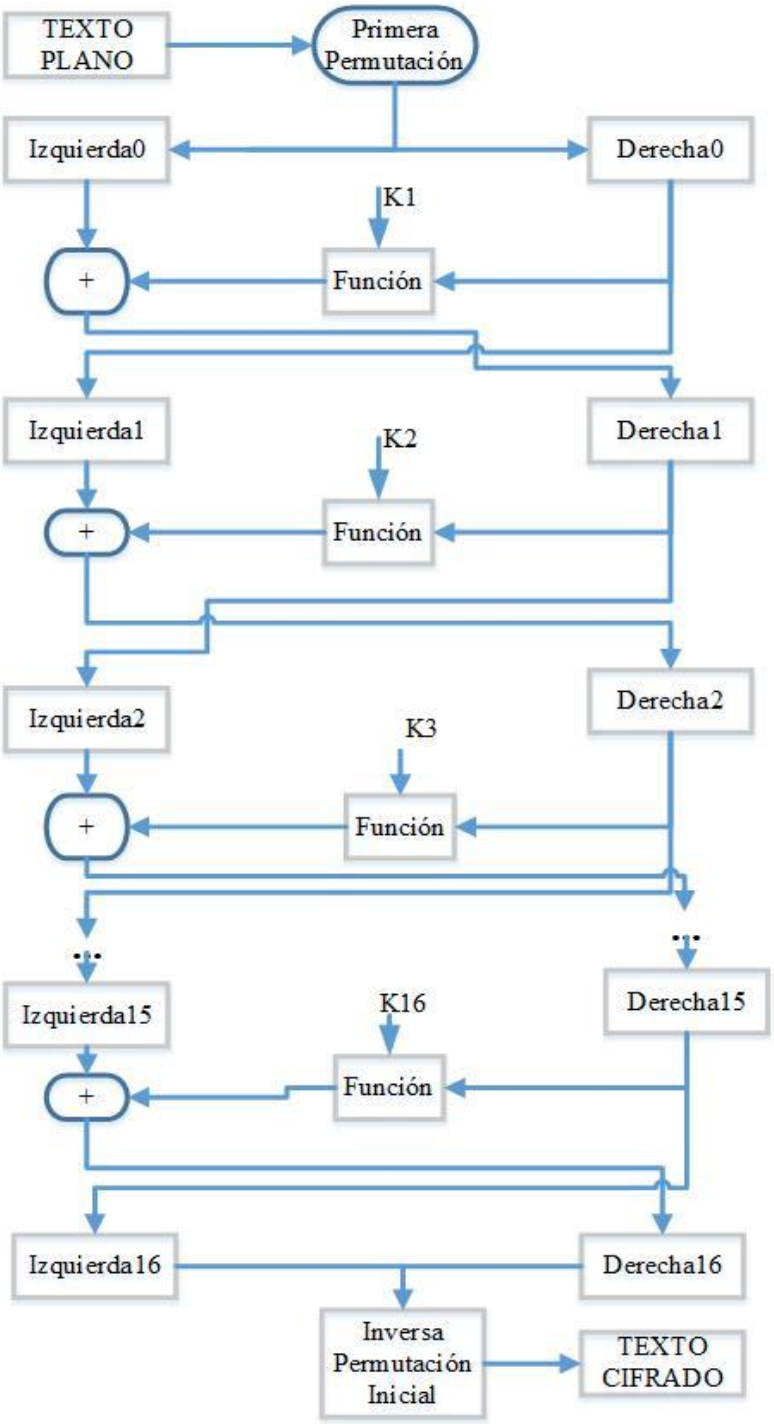


Figura 36. Algoritmo de Feistel

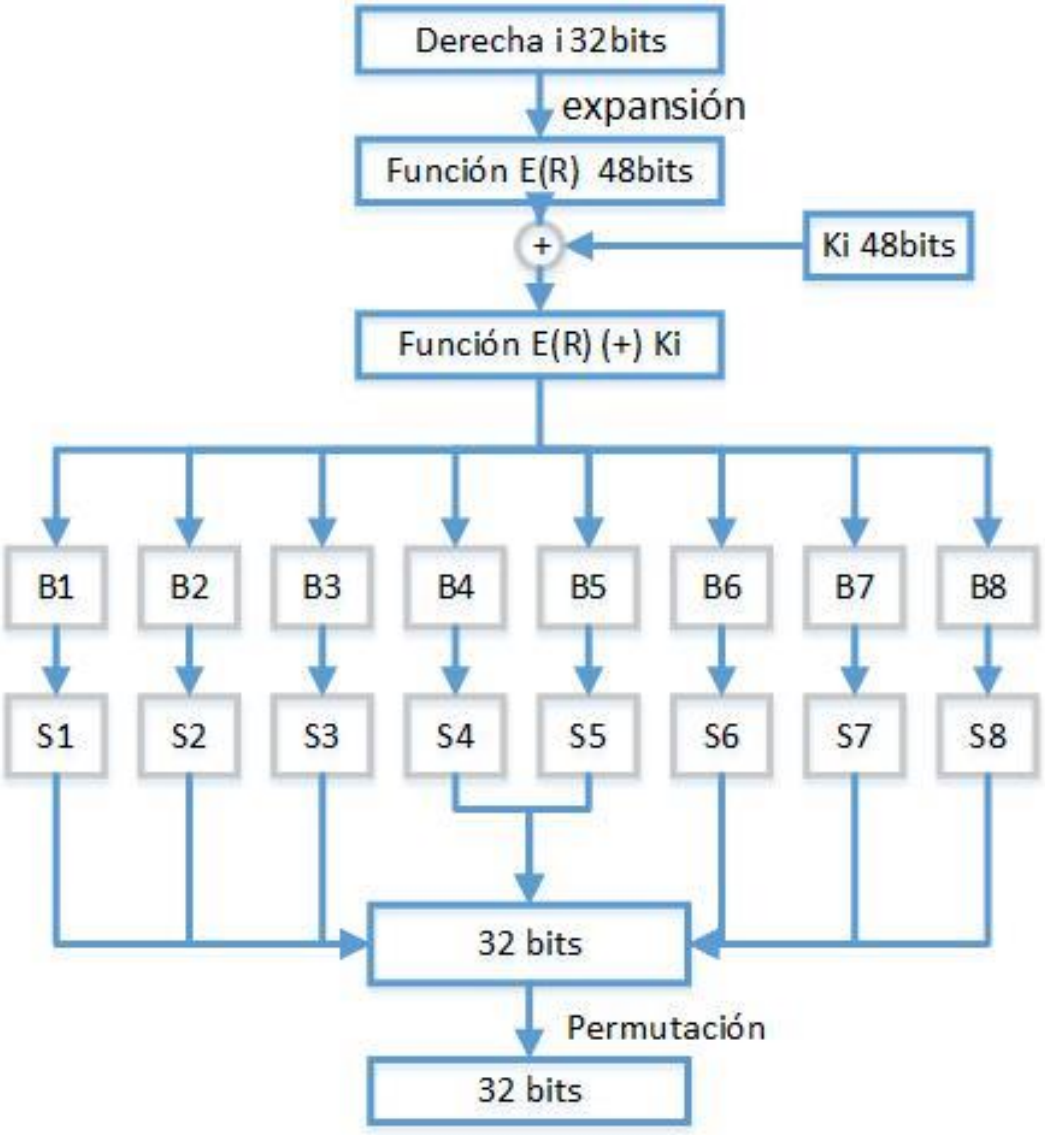


Figura 37. Algoritmo de la Función

A continuación, se realizó una última permutación que es la inversa de la permutación inicial como se presenta en la siguiente ecuación (22):

$$Cifrado = IP^{-1}(D_{16}, I_{16}) \quad (22)$$

La ecuación es prácticamente el mensaje de 64 bits cifrado, este proceso se realiza hasta que se haya cifrado todo el mensaje a transmitirse.

El algoritmo de cifrado DES utiliza las denominadas *S-Boxes* (cajas de Substitución), las cuales son componentes fundamentales para los algoritmos simétricos. Su objetivo es evitar que el fácil hallazgo de una relación entre la información original y la información cifrada. Las cajas-S se seleccionan cuidadosamente buscando que sean más resistentes al criptoanálisis. De forma general una caja-S no es más que una tabla de sustitución de $m * n$ bits, donde m bits son las cadenas de entrada y n bits son las cadenas de salida. Su uso es simple ya que se trabaja con un mensaje completo que será el bloque original el cual será dividido en varios segmentos de m bits los cuales son sustituidos por otros n bits. Un algoritmo es más complejo de descifrar mientras más grandes sean las cajas-S.

3.1.6.1 Rotación de bits

Se trata en que el bit que sale entra en el otro extremo, aquí no se pierden bits. Su funcionamiento se da por rotación a la izquierda, en donde el bit final ingresa en la posición del primer bit, en caso de ser la rotación a la derecha lo que sucede lo contrario ya que el primer bit sale e ingresa en la última posición del vector.

3.1.6.2 Descifrado DES

Para el descifrado del mensaje nuevamente se los divide en bloques de 64 bits y se realiza el proceso inverso de la se secuencia de cifrado. Como primer paso se divide en dos partes izquierda I y derecha D que en este caso serán los segmentos finales del proceso de cifrado siento I16 y

D16, a continuación, se aplica la secuencia inversa en las rondas de Feistel. Finalmente se aplica la Tabla 1 denominada permutación inicial con lo que se obtiene el mensaje original.

3.1.7 Algoritmo de cifrado DES en Matlab®

Para la simulación del cifrado DES se basa en el diagrama de la Figura 10, pero para el bloque de cifrado se utiliza el algoritmo DES, el cual está compuesto por varias funciones para la permutación de bits y las rondas de Feistel, trabajando en grupos de 64 bits.

3.1.7.1 Medidas de Distorsión MBSD, QPSK y BPSK Cifrado DES en Matlab®

En la Tabla 11 se observa los valores MBSD, donde el nivel de distorsión va disminuyendo mientras mayor sea la relación Eb/No, por lo que se confirma que el bloque de cifrado DES no genera problemas de distorsión de la señal, además se puede observar que desde 9dB el valor MBSD se estabiliza en 0.0652 para la modulación QPSK, pero para la modulación BPSK se puede observar que desde 6dB el valor MBSD se estabiliza en 0.0652, con lo que se considera que la señal recuperada es válida.

Tabla 11
Valores MBSD con modulación QPSK y BPSK cifrado DES

Eb/No	MBSD QPSK	MBSD BPSK
0	0.8073	0.6764
3	0.7685	0.2156
6	0.266	0.0652
9	0.0652	0.0652
12	0.0652	0.0652
15	0.0652	0.0652
18	0.0652	0.0652
20	0.0652	0.0652

En la Figura 38 se observa el comportamiento de los valores MBSD con la variación de la relación Eb/No para la modulación QPSK y BPSK.

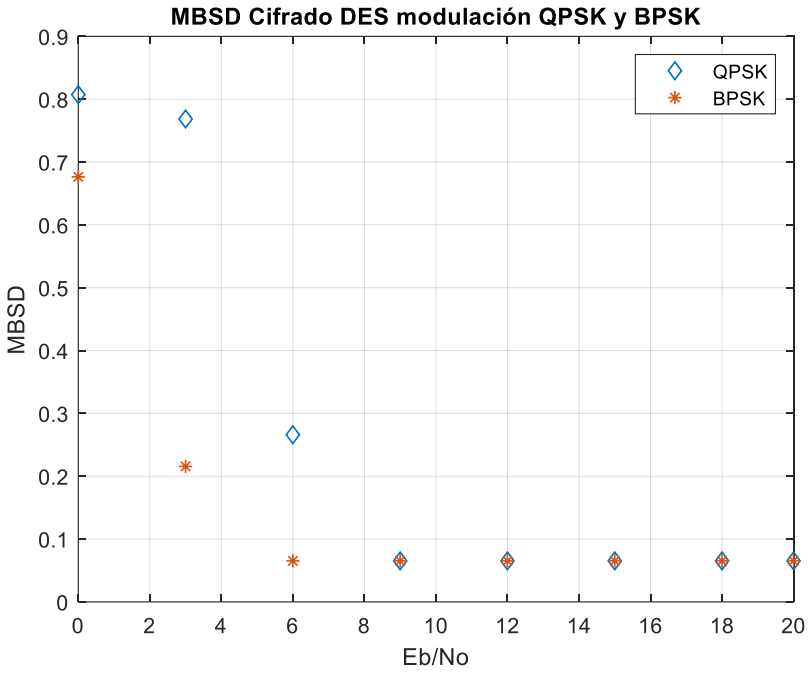


Figura 38. MBSD DES con modulación QPSK y BPSK en Matlab®

3.1.7.2 Medidas BER Cifrado DES en Matlab®

En la Figura 39 se observa que la gráfica del BER simulado acompaña la gráfica del teórico por lo que la SDR simulada en Matlab® funciona correctamente y los bits errados van disminuyendo conforme la relación Eb/No se va incrementando. Pero no se completa la gráfica ya que el sistema SDR llega a un punto donde se estabiliza y se considera un caso ideal ya que después de 10dB, los bits errados se calcularon como 0.

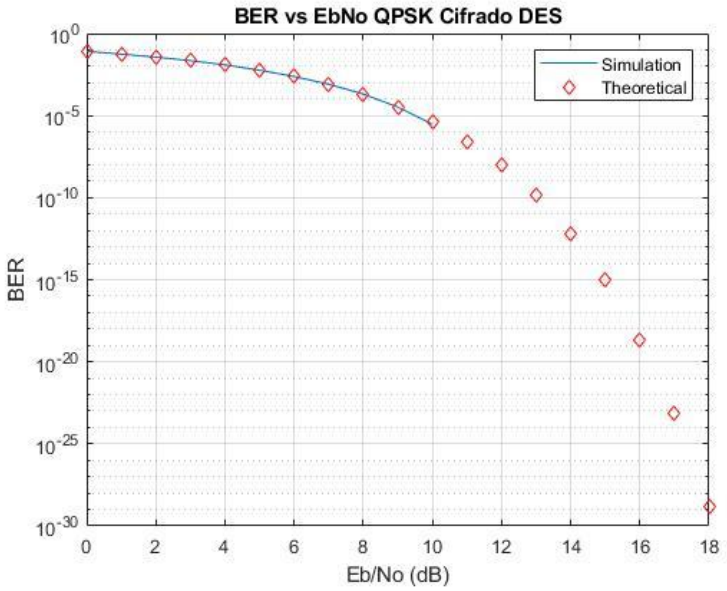


Figura 39. BER vs Eb/No QPSK Cifrado DES

En la Figura 40 se observa que el comportamiento del BER simulado sigue la trayectoria del BER teórico por lo que la SDR simulada en Matlab® funciona correctamente y los bits errados van disminuyendo conforme la relación Eb/No va aumentando y en 10dB el BER es cero.

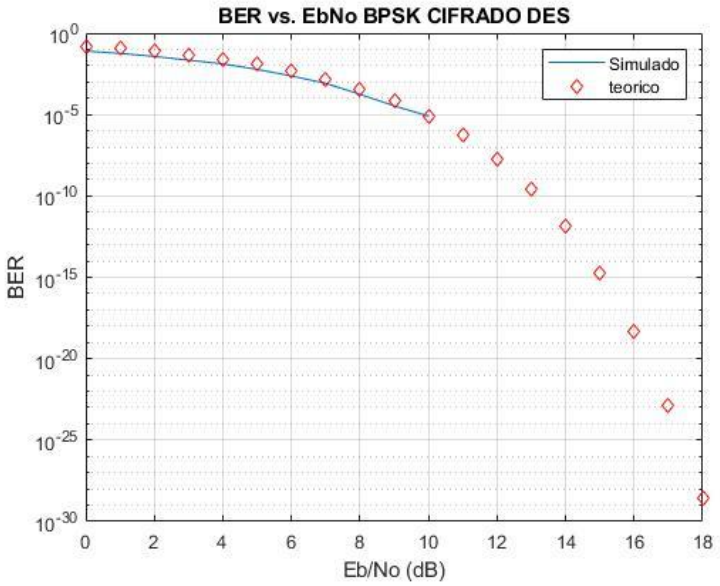


Figura 40. BER vs Eb/No con modulación BPSK Cifrado DES

3.1.8 Algoritmo de cifrado DES en LabVIEW®

3.1.8.1 Medidas de Distorsión MBSD, QPSK y BPSK Cifrado DES en LabVIEW®

En la Tabla 12 se observa los valores MBSD, donde el nivel de distorsión va disminuyendo mientras mayor sea la relación Eb/No, pero para la modulación QPSK con 6dB la distorsión alcanza su mínimo valor ya que con 20dB se tiene la misma distorsión que con 6dB. Además, presenta menor distorsión para una baja relación Eb/No ya que como se observa en la Tabla 11 este se estabiliza en 9dB con modulación QPSK.

Para la modulación BPSK el nivel de distorsión va disminuyendo mientras mayor sea la relación Eb/No, pero con 3dB la distorsión alcanza su mínimo valor ya que con 20dB se tiene la misma distorsión que con 3dB, por lo que se confirma que el bloque de cifrado DES no genera problemas de distorsión de la señal, con lo que se considera que la señal recuperada en LabVIEW® es válida. Además, presenta menor distorsión para una baja relación Eb/No y modulación BPSK ya que como se observa en la Tabla 11 este se estabiliza desde 6dB.

Tabla 12
Valores MBSD con modulación QPSK y BPSK en LabVIEW®

Eb/No	MBSD QPSK	MBSD BPSK
0	3.5744	4.476
3	3.3801	0.0462
6	0.0462	0.0462
9	0.0462	0.0462
12	0.0462	0.0462
15	0.0462	0.0462
18	0.0462	0.0462
20	0.0462	0.0462

En la Figura 41 se observa el comportamiento de los valores MBSD con la variación de la relación Eb/No para la modulación QPSK y BPSK.

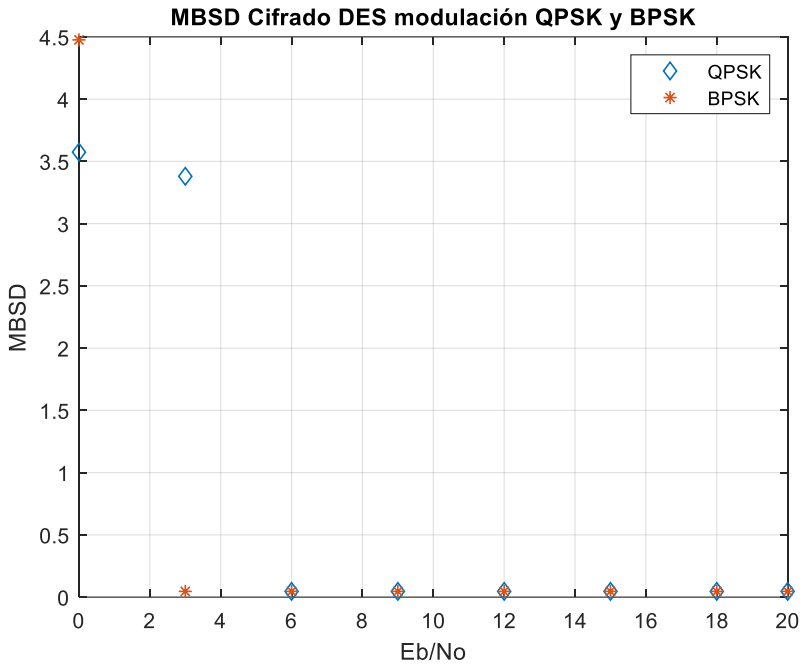


Figura 41. MBSD Cifrado DES con modulación QPSK y BPSK en LabVIEW®

3.1.8.2 Medidas BER Cifrado DES en LabVIEW®

La Figura 42 presenta la cantidad de bits errados en una transmisión de 64kbits donde la cantidad de bits errados máxima es 6531bits cuando la relación Eb/No es 0dB, pero una relación de 7dB la cantidad de bits errados es 2824 por lo que se observa que la señal se recibe y mejora el número de bits errados mejorando las características del canal, debido a que es una simulación.

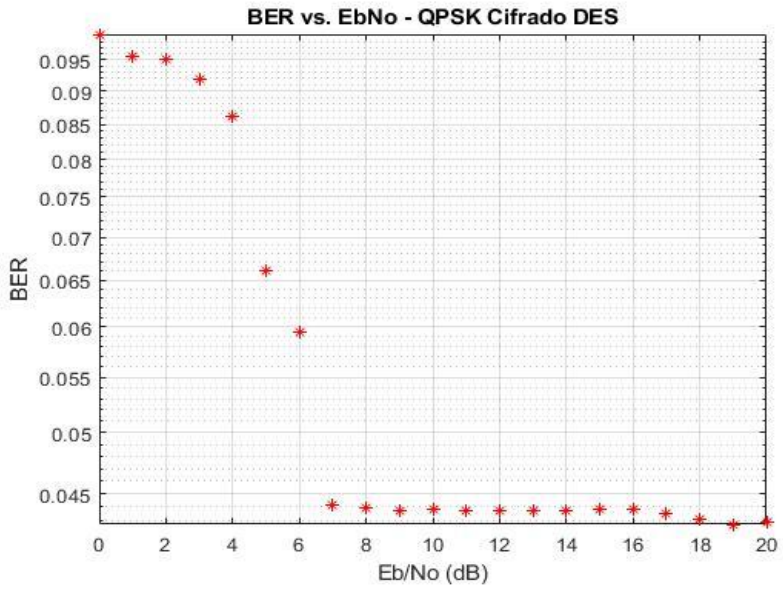


Figura 42. BER vs Eb/No QPSK Cifrado DES en LabVIEW®

La Figura 43 presenta la cantidad de bits errados en una transmisión de 64kbits con modulación QPSK, se observa que la máxima cantidad de errores es con una relación Eb/No es 0dB, pero una relación de 6dB la cantidad de bits errados disminuye significativamente.

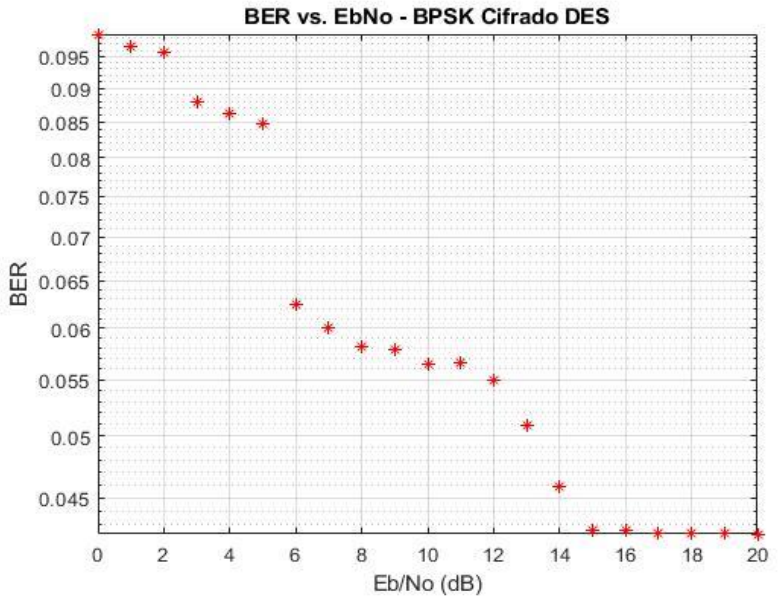


Figura 43. BER vs Eb/No BPSK Cifrado DES LabVIEW®

En la Figura 44 se observa en la parte izquierda la señal transmitida y en la derecha la señal recuperada al utilizar el cifrado DES, con lo que se concluye que se recupera la señal.

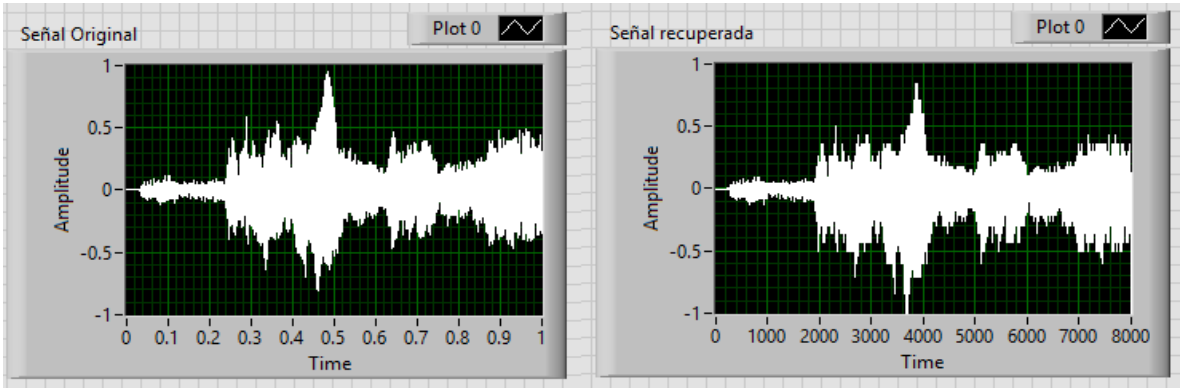


Figura 44. Señal transmitida y señal recibida en la SDR Cifrado DES

3.1.9 Algoritmo de cifrado AES

3.1.9.1 Cifrado

- **Cálculo de las Sub Claves**

El algoritmo de Cifrado AES es considerado una de las mejores técnicas de cifrado por bloques, este cifrado puede trabajar en bloques de 128, 192 y 254 bits. Para el desarrollo del código se utilizó el algoritmo que divide los datos en segmentos de 16 bytes (128 bits).

Para empezar con la elaboración del algoritmo, se inicia con la confección de diez sub claves que sirven para remplazar en los datos en las rondas de operaciones. La primera columna de la primera sub clave se calculó tomando la última columna de la clave original y se aplicó la operación llamada *Rotword* la cual rota el primer byte de la columna hacia el último lugar.

A la columna que resulto del paso anterior se le aplicó la operación llamada *SubByte* la que reemplaza cada byte por otro almacenado en la Tabla 2, los primeros 4 bits (primer número

hexadecimal) indican la posición de la fila y los siguientes 4 bits denotan la posición de la columna.

Cuando ya se reemplazaron todos valores de la columna con los bytes de la Tabla 2, a la columna resultante se le aplica la función XOR con los datos de la primera columna de la matriz de estado y con los bytes de la columna que se encontraba 4 posiciones atrás de la nueva columna. De esta manera obtenemos la primera columna de la primera sub clave.

Para calcular las tres columnas restantes de la primera sub clave se aplicó un XOR entre una columna anterior y la columna que se encuentra 4 posiciones atrás.

De esta manera se calculó la primera sub clave y para las nueve restantes aplicamos el mismo método utilizando la sub clave posterior a la que se está encontrando. Al finalizar obtuvimos las 11 claves que sirvieron para las rondas de operaciones.

- **Etapas de Operación**

El algoritmo para 128 bits realiza 11 rondas de cifrado, cada ronda utiliza 1 sub clave y varias operaciones para ocultar el texto plano. Las rondas se dividen en las siguientes etapas.

- Ronda Inicial
- Ronda Estándar
- Ronda Final

En cada etapa las operaciones que básicamente se utilizaron fueron las siguientes:

- **Ronda Inicial**

En la ronda inicial se aplicó la siguiente operación:

- ***AddRoundKey:*** Esta operación realiza un XOR byte a byte entre la clave original y el primer estado (Texto Plano).

- **Rondas Estándar**

Para las nueve rondas estándar se aplicaron las siguientes operaciones:

- ***SubBytes:*** En esta operación se realizó una sustitución byte a byte tomando los datos de la Tabla 2 para el cifrado, la misma que se utilizó para el cálculo de las sub claves.
- ***ShiftRows:*** Tomando el estado de la operación anterior se aplicó *ShiftRow* la cual rota bytes hacia la izquierda exceptuando la primera fila, la segunda fila rota un byte, la tercera rota dos bytes y la cuarta fila rota tres bytes.
- ***MixColumns:*** En esta operación se tomó cada columna del estado anterior y se la multiplicó por los valores de la Tabla 13.

Tabla 13
Matriz GF preestablecida

02	03	01	01
01	02	03	01
01	01	02	03
03	01	01	02

$$c1 = (A * 02)XOR(B * 03)XOR(C * 01)XOR(D * 01) \quad (23)$$

Antes de aplicar la ecuación (23) para cada casillero del nuevo estado se cambió los valores del estado anterior y de la Tabla 13.

- ***AddRoundKey:*** Al igual que en la ronda inicial se aplica un XOR entre la sub clave correspondiente y el estado correspondiente.

- **Ronda Final**

En la ronda final al igual que en las rondas estándar se aplicó las operaciones mencionadas anteriormente con la excepción de *MixColumn*, obteniendo como resultado el texto cifrado.

3.1.9.2 Descifrado

Para realizar el proceso de descifrado se aplicó los mismos pasos del cifrado en forma inversa utilizando las mismas sub claves.

- **Ronda Inicial**

Como primer paso se realizó la ronda final del cifrado empezando con la operación *AddRoundKey* con el texto cifrado y la última sub clave calculada.

En segundo lugar, se aplicó una operación inversa de *ShiftRows* rotando esta vez los bytes hacia la derecha exceptuando la primera fila. Para terminar con la ronda final se realizó la operación inversa de *SubBytes* utilizando la Tabla 3.

- **Rondas Estándar**

Al igual que en el cifrado se realizaron nueve rondas aplicando las cuatro operaciones inversas, se inició como en el paso anterior con un *AddRoundKey* realizando un XOR entre el estado y la sub clave correspondiente.

En la siguiente operación se realizó el paso inverso de la función *MixColumns*, el cual es similar al del cifrado con la diferencia que se utiliza la Tabla 14.

Tabla 14

Tabla GF para la función MixColumns inversa

0E	0B	0D	09
09	0E	0B	0D
0D	09	0E	0B
0B	0D	09	0E

Como tercer punto se aplicó la operación inversa de ShiftRows como ya fue explicado en el primer paso.

La operación final de las rondas estándar es la inversa de *SubBytes*, obteniendo la matriz final para el descifrado.

- **Ronda Final**

En la última ronda del descifrado se aplicó un AddRoundKey como en el primer paso del cifrado, de esta manera se obtuvo el texto plano original.

3.1.10 Algoritmo de cifrado AES en Matlab®

3.1.10.1 Medidas BER Cifrado AES en Matlab®

Se procedio a calcular el BER, en codigo de Matlab® para el cifrado AES, como se puede observar en la Figura 45 los valores del BER descienden mientras aumenta la relacion señal a ruido hasta nueve dB despues los valores toman el valor de cero.

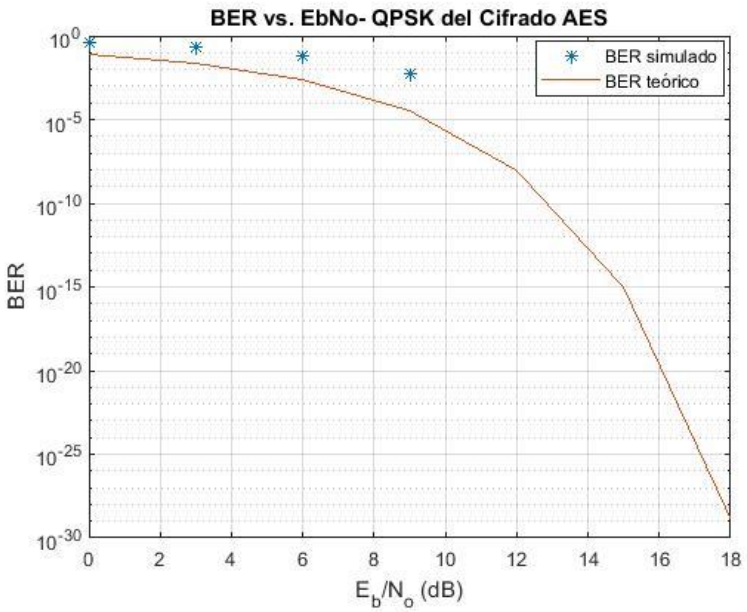


Figura 45. BER con el cifrado AES modulación QPSK en Matlab®

Como se observa en la Figura 46 se los valores del BER descendieron hasta nueve dB después tomaron el valor de cero, por lo que la simulación en Matlab® funciona correctamente.

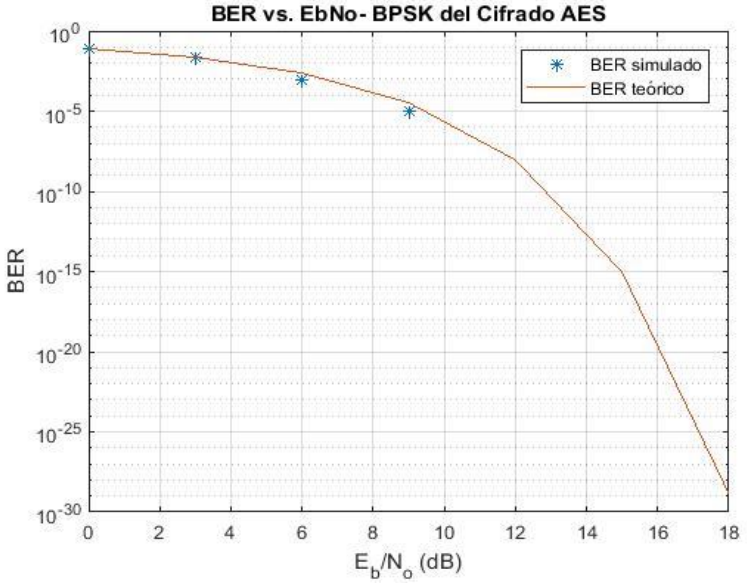


Figura 46. BER con el cifrado AES modulación QPSK en Matlab®

3.1.10.2 Medidas de Distorsión MBSD, QPSK y BPSK con Cifrado AES en Matlab®

Utilizando el algoritmo de cifrado AES se calculó MBSD a 20 ms y 8kHz como con una modulación QPSK como se observa en la Tabla 15 teniendo un decremento en los valores del MBSD conforme aumenta la relación señal a ruido.

Tabla 15

Valores MBSD con modulación QPSK y BPSK en Matlab®

Eb/No	MBSD QPSK	MBSD BPSK
0	1,5425	1,5238
3	1,5239	1,3495
6	1,3449	1,3340
9	1,6202	1,6193
12	1,3828	1,3785
15	1,2457	1,2285
18	1,3190	1,3039
20	1,2350	1,2425

La Figura 47 ilustra los valores de la Tabla 15 haciendo una comparativa de los valores de distorsión MBSD con modulación QPSK y BPSK en Matlab®.

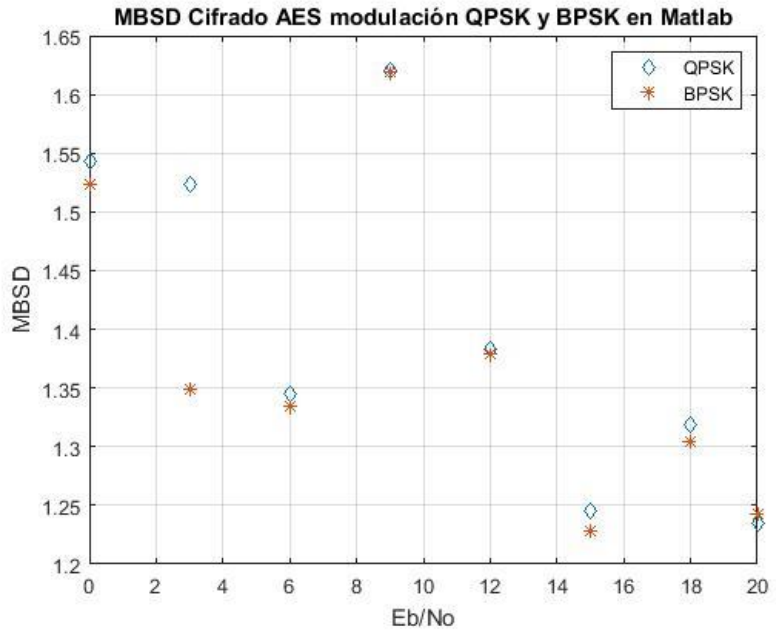


Figura 47. *MBSD Cifrado AES con modulación QPSK y BPSK en* Matlab®

3.1.11 Algoritmo de cifrado AES en LabVIEW®

3.1.11.1 Medidas de Distorsión MBSD, QPSK y BPSK con Cifrado DES en LabVIEW®

En la Tabla 16 se observa que los niveles de distorsión MBSD de la señal recibida son altos para una relación Eb/No de 0dB pero la distorsión disminuye rápidamente con una relación Eb/No de 3dB, llegando a no presentar distorsión en 6dB para QPSK, para BPSK tiene el mismo comportamiento que la modulación QPSK con la diferencia que se estabiliza antes, esto debido a que es un caso más cercano a lo ideal.

Tabla 16

Valores MBSD con modulación QPSK y BPSK en LabVIEW®

Eb/No	MBSD QPSK	MBSD BPSK
0	1,7510	1.8237
3	0.8977	0
6	0	0

CONTINÚA

9	0	0
12	0	0
15	0	0
18	0	0
20	0	0

La Figura 48 se muestra los valores de las medidas de distorcion de la Tabla 25 los mismos que muestran una gran estabilidad respecto a los demás cifrados.

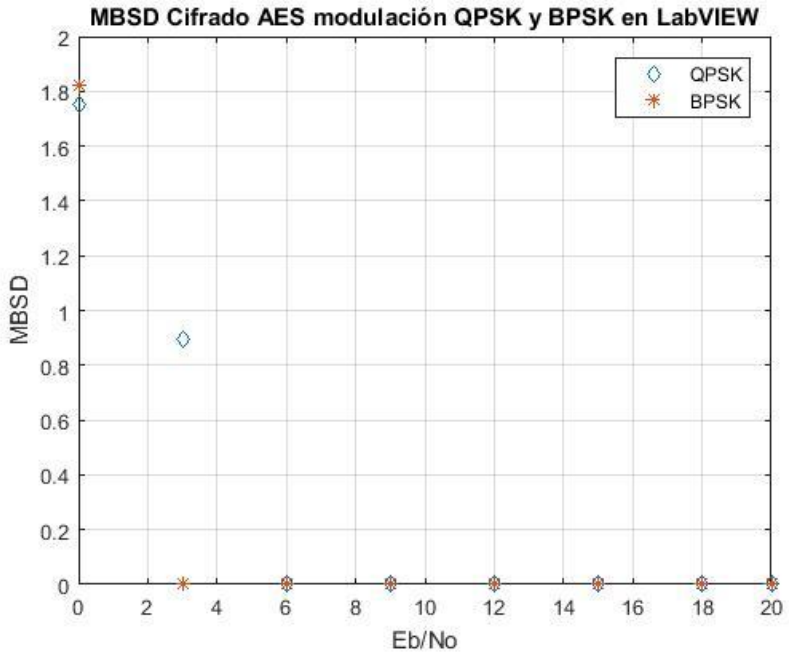


Figura 48. MBSD Cifrado AES con modulación QPSK y BPSK en LabVIEW®

3.1.11.2 Medidas BER Cifrado AES en LabVIEW®

En las gráficas de la Figura 49 y Figura 50 se presentan la cantidad de bits errados en una transmisión de 64kbits con una relación Eb/No que va de 0dB a 20dB, se observa que tienen el mismo comportamiento, presentándose menos bits errados con modulación BPSK. Y se observa que los bloques de cifrado y descifrado trabajan correctamente en la recuperación del mensaje.

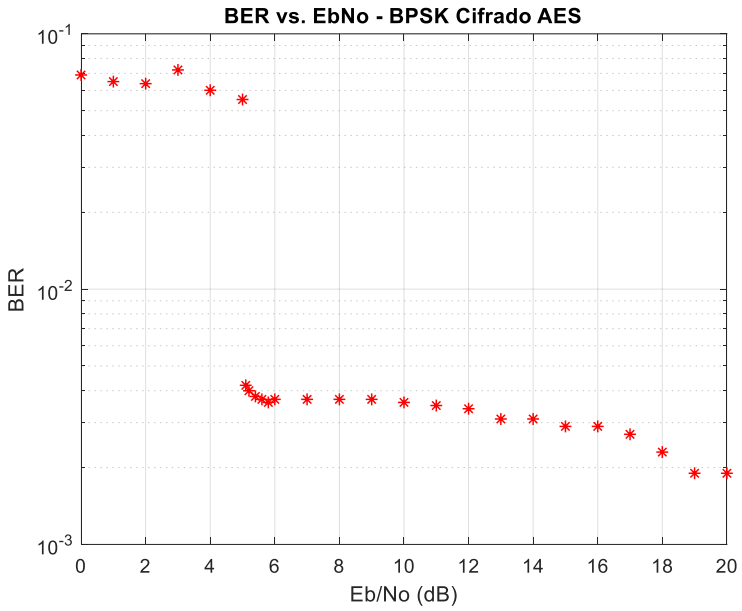


Figura 49. BER con el cifrado AES modulación BPSK en LabVIEW®

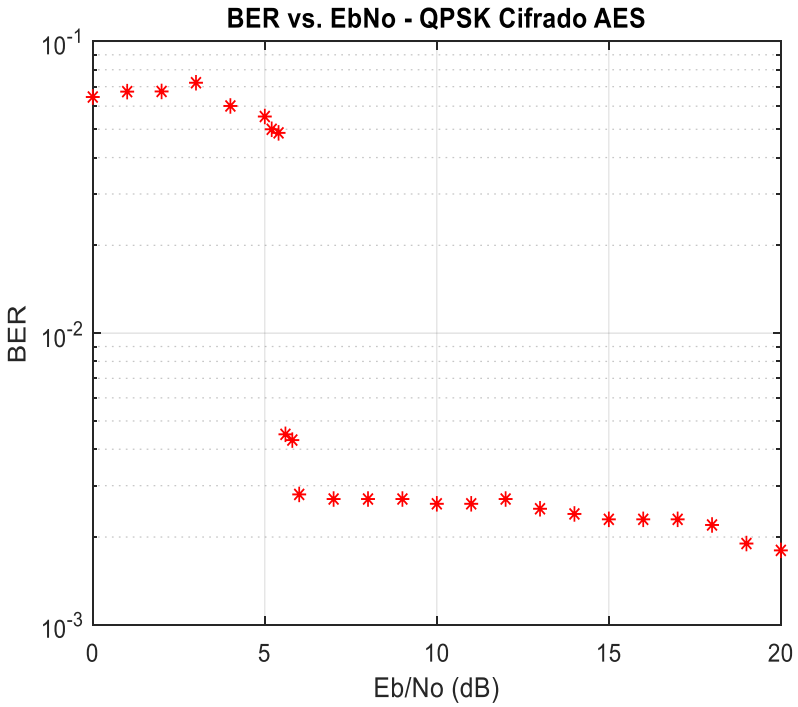


Figura 50. BER con el cifrado AES modulación QPSK en Matlab®

3.2 Señal original vs señal recuperada con una relación Eb/No de 40dB

Cifrado en Tiempo en Matlab® para una relación Eb/No mayor a 6dB se estabiliza y los bits errados van disminuyendo en 1 bit, permitiendo una visualización que al parecer se recuperó la señal sin errores como se observa en la Figura 51. Para esta prueba se usó una relación Eb/No de 40dB, al realizar una ampliación de la señal original comparada con la señal recuperada se puede ver que aun con una alta relación Eb/No, no se logra recuperar la señal completamente como se observa en la Figura 52.

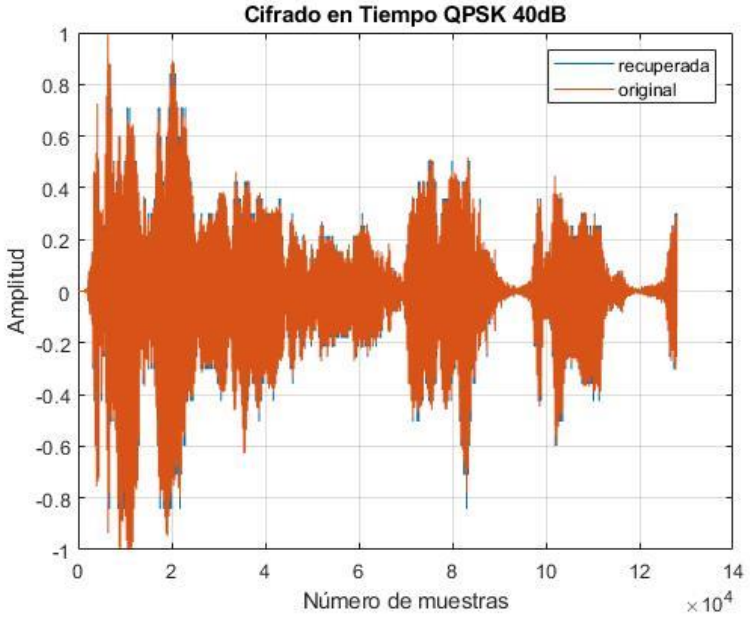


Figura 51. Cifrado en Tiempo con Eb/No de 40dB en Matlab®

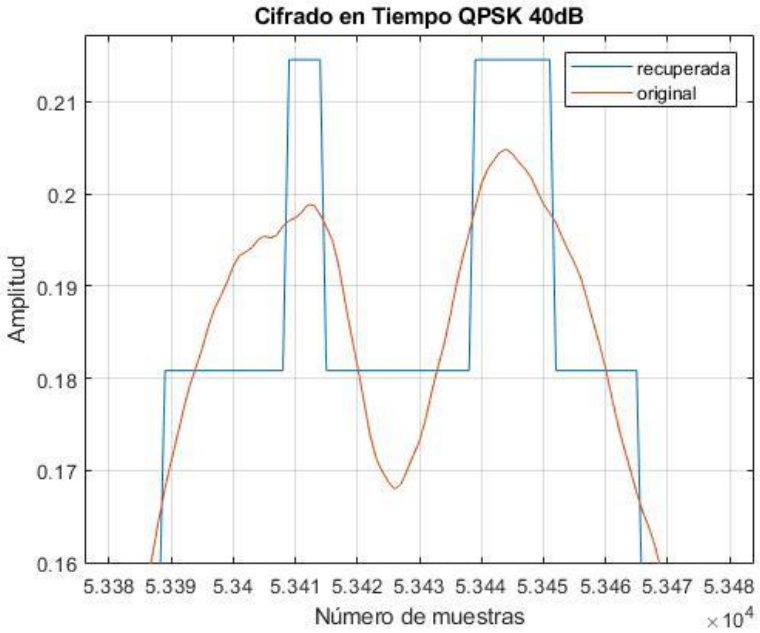


Figura 52. Ampliación de la señal recuperada con Cifrado en tiempo a 40dB

En la Figura 53 se observa la señal original versus la señal recuperada con Cifrado en Tiempo en LabVIEW® y una relación Eb/No de 40dB, de forma general parece que se recuperó sin errores, pero al realizar una ampliación de la misma se observa que aun con una alta relación Eb/No aún existen pequeñas variaciones entre la señal recuperada y la original como se observa en la Figura 49 esto debido a todos los bloques de procesamiento como son: la codificación de fuente, el cifrado, codificación de canal y el canal, además de su proceso inverso en el receptor.

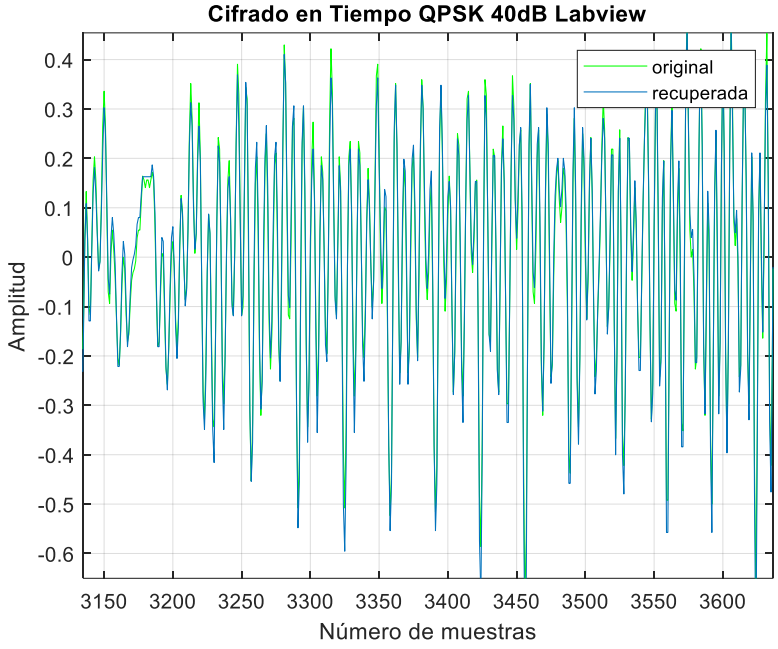


Figura 53. Cifrado en Tiempo con Eb/No de 40dB en LabVIEW®

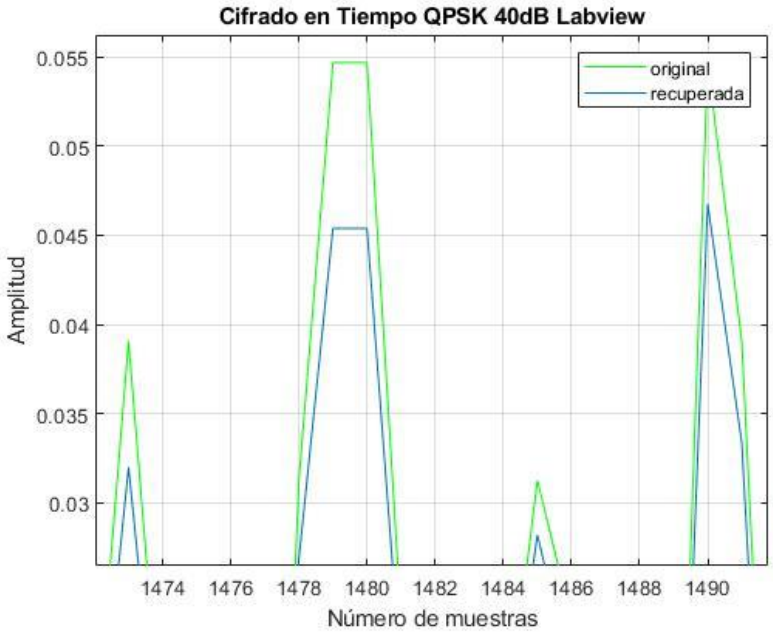


Figura 54. Ampliación señal recuperada Cifrado en Tiempo en LabVIEW®

CAPÍTULO IV

4 ANÁLISIS DE RESULTADOS IMPLEMENTADOS

La implementación de los algoritmos de cifrado en la SDR se realizó en dos tarjetas USRP 2920, usando como medio de transmisión el aire como se observa en la Figura 55, la configuración de las tarjetas se presenta en la Tabla 17**.**

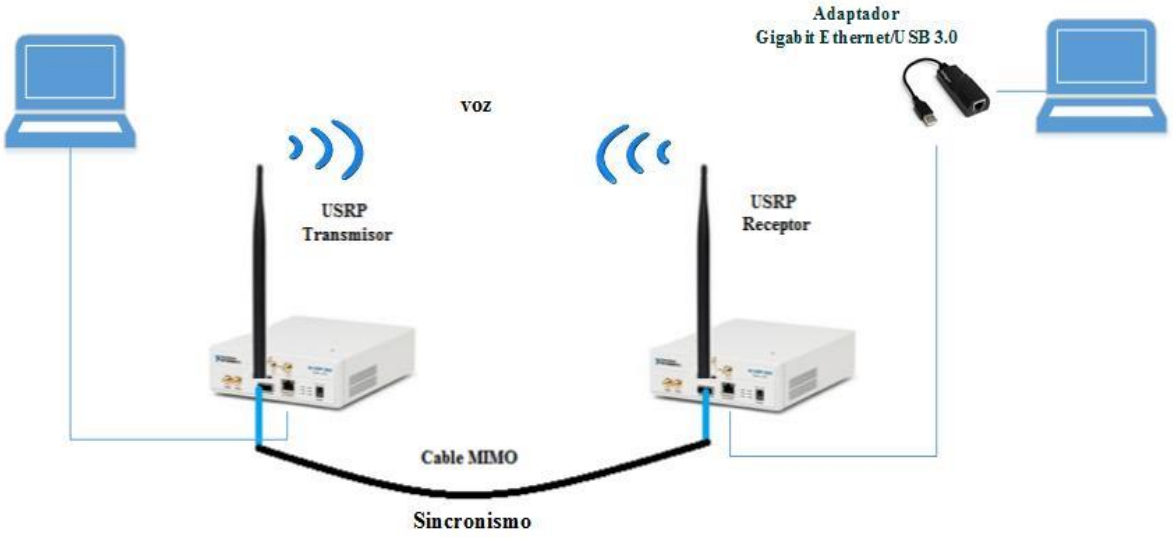


Figura 55. Implementación de la SDR con los algoritmos de cifrado

Tabla 17

Configuración de las tarjetas USRP 2920

Dirección IP Transmisor	192.168.10.4
Dirección IP Receptor	192.168.10.2
IQ Sampling rate [S/seg]	200k
Frecuencia Portadora	1.4GHz
Potencia de salida [dBm]	15-18
Antena Tx, Rx	TX1, RX2

En LabVIEW® se trabajó usando los bloques MathScript para los algoritmos de cifrado, lo que conlleva un alto costo en el tiempo de procesamiento por lo que en la implementación se trabajó con un bajo números de muestras.

Debido a que no se logró sincronizar completamente la transmisión y la recepción se tomó muestras de la señal recibida, con lo que se obtuvieron los siguientes resultados.

4.1 Cifrado en Tiempo

4.1.1 Modulación BPSK

En la Figura 56 se presentan las señales transmitidas versus las señales recibidas con cifrado en tiempo implementada en la tarjeta USRP 2920, la señal de color azul es la señal transmitida también denominada señal original y la señal de color naranja es la señal recibida también denominada señal recuperada. Estos son los resultados de la implementación del cifrado en tiempo con modulación BPSK. Las medidas objetivas de distorsión MBSD y las medidas BER se promediaron de 6 muestras y se presentan en la Tabla 18**.**

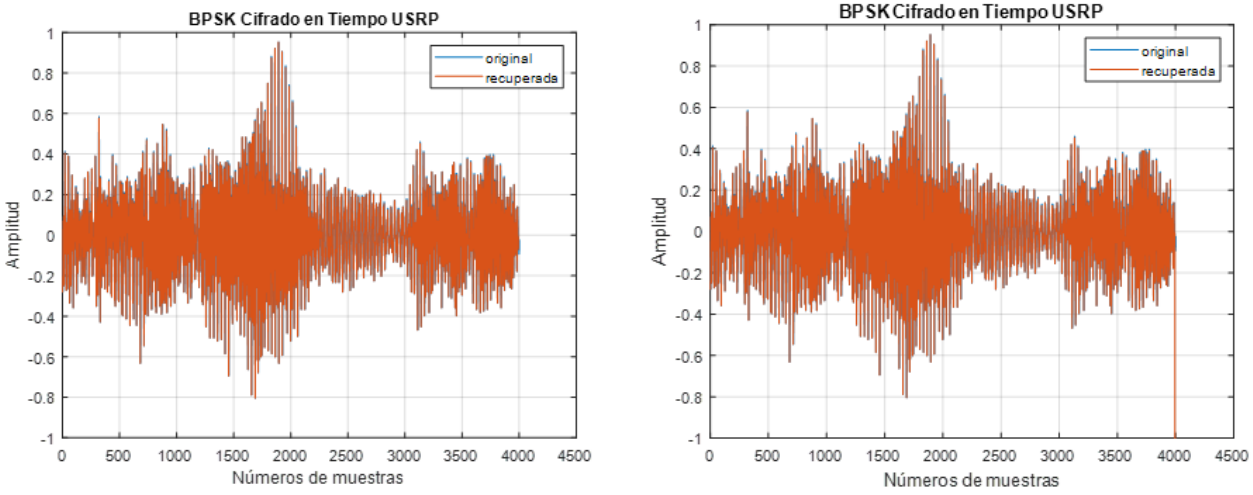


Figura 56. Señales recibidas en la USRP con cifrado en tiempo BPSK

Tabla 18

Valores MBSD y BER cifrado en tiempo con modulación BPSK

MBSD	**BER**
0.3436	0.05421

Los resultados obtenidos indican que la señal recibida no presenta un nivel de distorsión significativo, por lo que el mensaje se cifra y descifra sin problemas de distorsión.

4.1.2 Modulación QPSK

En la Figura 57 se presentan las señales originales en color azul y las señales recuperadas en color naranja, con cifrado en tiempo y modulación QPSK, se puede observar que las señales recuperadas siguen a la señal original, por lo que el sistema implementado con cifrado en tiempo si está trabajando en cifrar y descifrar el mensaje.

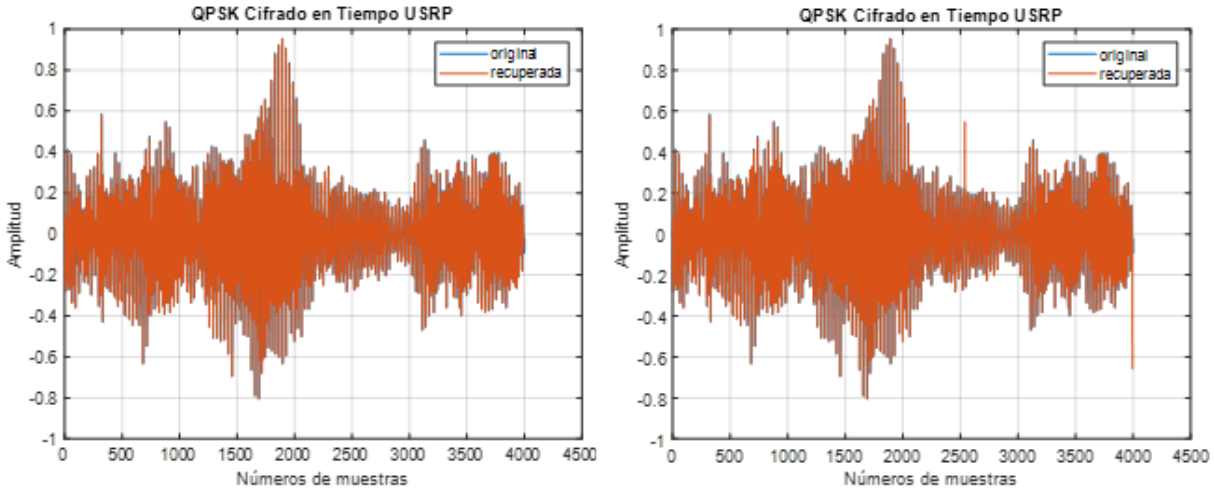


Figura 57. Señales recibidas en la USRP con cifrado en tiempo QPSK

En la Tabla 19 se presenta el valor MBSD y el BER del cifrado en tiempo con modulación QPSK, se observa que hay mayor distorsión.

Tabla 19

Valores MBSD y BER cifrado en tiempo con modulación QPSK

MBSD	BER
0.3464	0.05417

4.2 Cifrado en Frecuencia

4.2.1 Modulación BPSK

En la Figura 58 se presentan los resultados de la implementación del cifrado en frecuencia con modulación BPSK donde se tiene las señales originales en azul y las señales recuperadas o recibidas en naranja, se observa que las señales recuperadas siguen a las señales originales.

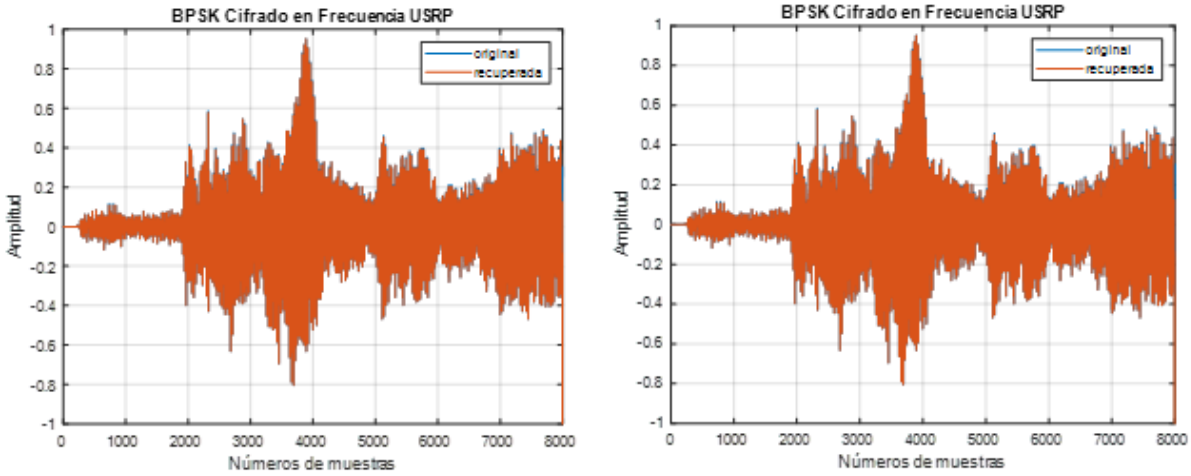


Figura 58. Señales recibidas en la USRP con cifrado en frecuencia BPSK

En la Tabla 20 se observa un valor mínimo de distorsión, por lo que el cifrado en frecuencia presenta menor distorsión que el cifrado en tiempo.

Tabla 20

Valores MBSD y BER cifrado en frecuencia con modulación BPSK

MBSD	BER
0.0021	0.06180

4.2.2 Modulación QPSK

En la Figura 59 se presentan los resultados de la implementación del cifrado en frecuencia con modulación QPSK, se puede observar las señales originales en azul y las señales recuperadas o recibidas en naranja, se observa que las señales recuperadas siguen a las señales originales.

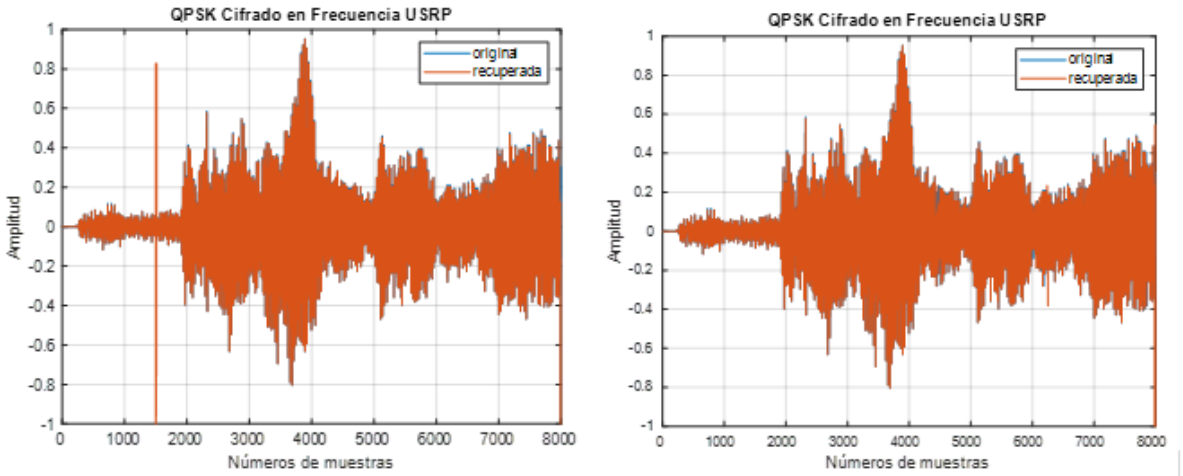


Figura 59. Señales recibidas en la USRP con cifrado en frecuencia QPSK

En la Tabla 21 se observa que el cifrado en frecuencia presenta menor distorsión que el cifrado en tiempo, pero con la modulación QPSK se presenta una mayor distorsión que con la modulación BPSK de la Tabla 20.

Tabla 21

Valores MBSD y BER cifrado en frecuencia con modulación QPSK

MBSD	**BER**
0.0065	0.06392

4.3 Cifrado DES

4.3.1 Modulación BPSK

En la Figura 60 se presentan los resultados de la implementación del cifrado DES en las tarjetas USRP 2920 con modulación BPSK, se puede observar las señales originales en azul y las señales recuperadas o recibidas en naranja, además se visualiza que las señales recuperadas siguen a las señales originales.

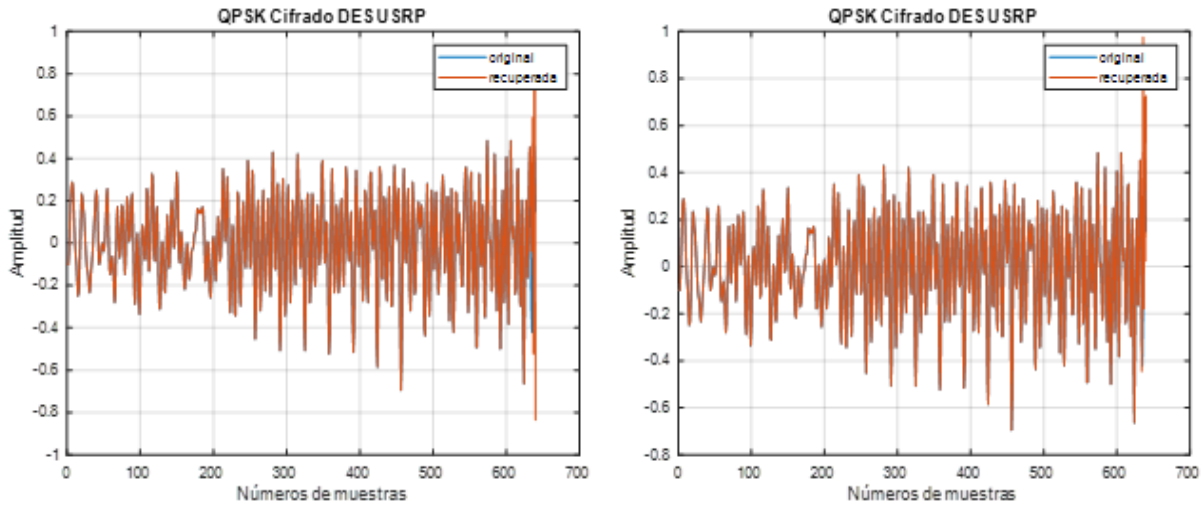


Figura 60. Señales recibidas en la USRP con cifrado DES, BPSK

En la Tabla 22 se observa el nivel de distorsión MBSD promedio de las señales analizadas, el cual es mínimo es decir que existe un nivel bajo de distorsión de la señal recibida respecto a la señal transmitida.

Tabla 22

Valores MBSD y BER cifrado DES con modulación BPSK

MBSD	BER
0.01372	0.01606

4.3.1 Modulación QPSK

En la Figura 61 se presentan los resultados de la implementación del cifrado DES con modulación QPSK en las tarjetas USRP 2920.

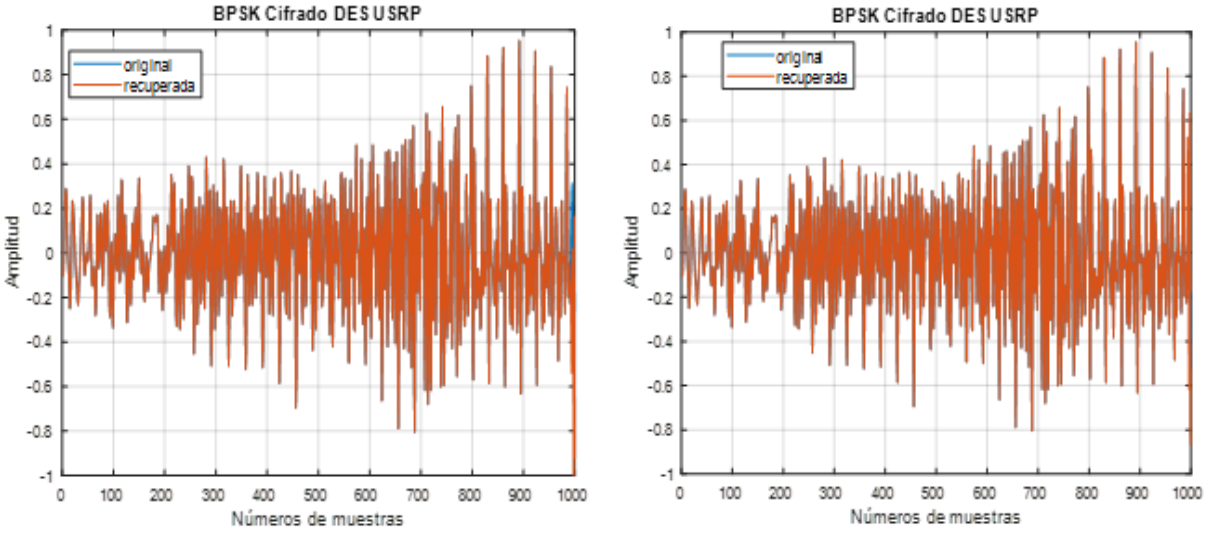


Figura 61. Señales recibidas en la USRP con cifrado DES, QPSK

En la Tabla 23 se observa que el nivel de distorsión de la señal recibida respecto a la señal transmitida es mínimo. Y se comprueba que existe mayor distorsión con modulación QPSK.

El cifrado que presenta mayor distorsión es el cifrado en tiempo debido a que trabaja directamente con los datos sin realizar ningún cambio.

Tabla 23
Valores MBSD y BER cifrado DES con modulación QPSK

MBSD	BER
0.09485	0.01597

4.4 Cifrado AES

4.4.1 Modulación QPSK

En la Figura 62 se presentan los resultados de la implementación del cifrado AES con modulación QPSK, donde se puede observar las señales originales en azul y las señales

recuperadas o recibidas en color naranja, además se visualiza que las señales recuperadas acompañan a las señales originales.

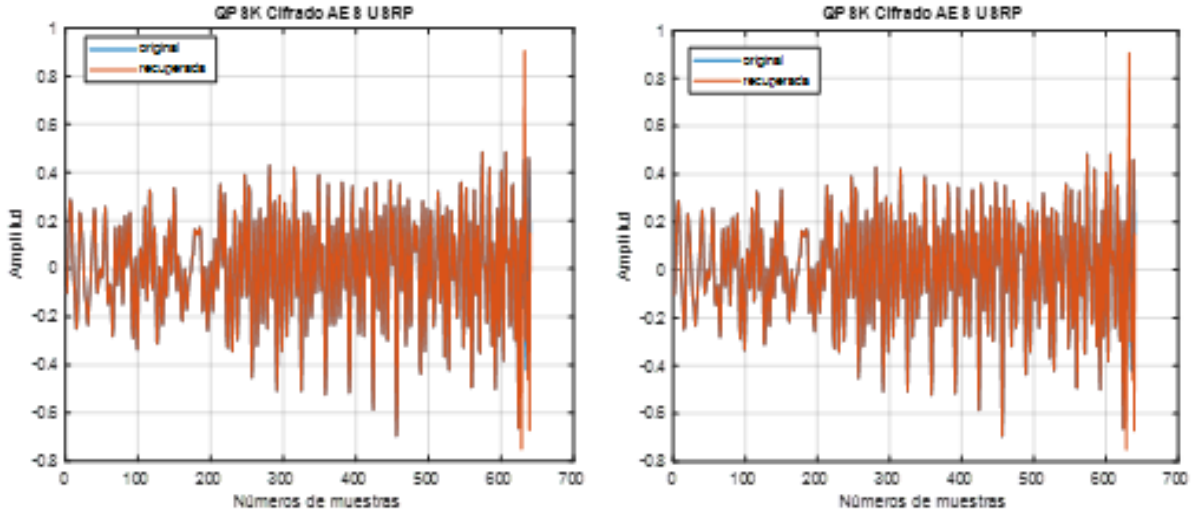


Figura 62. Señales recibidas en la USRP con cifrado AES, QPSK

En la Tabla 24 se observa que el nivel de distorsión de la señal recibida respecto a la señal transmitida es mínimo y menor que la distorsión que presenta la señal recuperada con el cifrado DES. Y se comprueba que existe mayor distorsión con modulación QPSK.

Tabla 24

Valores MBSD y BER cifrado AES con modulación QPSK

MBSD	BER
0.0227	0.005691

4.4.2 Modulación BPSK

En la Figura 63 se presentan los resultados de la implementación del cifrado AES con modulación BPSK, donde se puede observar las señales originales en azul y las señales

recuperadas o recibidas en color naranja, además se visualiza que las señales recuperadas acompañan a las señales originales prácticamente como si no hubiera pérdidas.

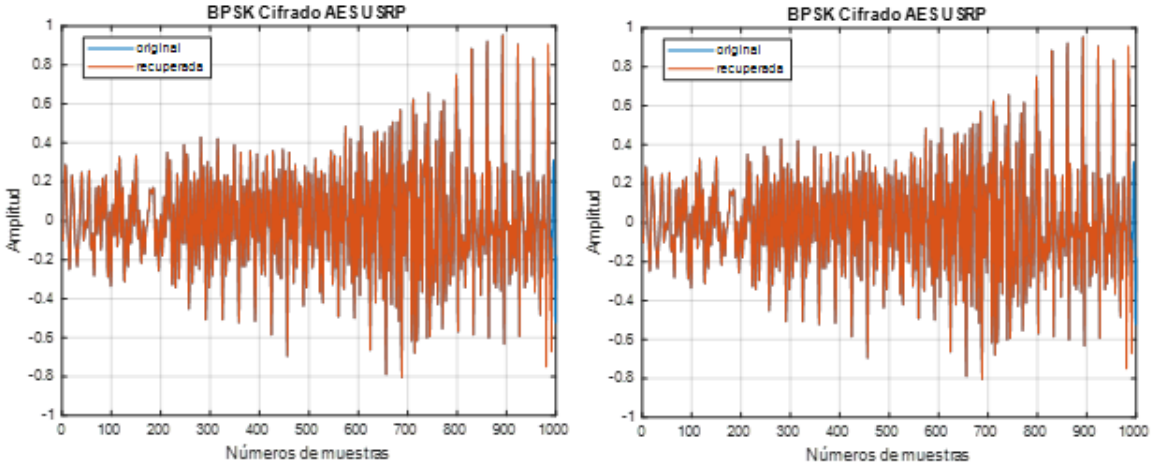


Figura 63. Señales recibidas en la USRP con cifrado AES, BPSK

En la Tabla 25 se observa que el nivel de distorsión de la señal recibida respecto a la señal transmitida la distorsión se presenta como cero lo cual se analizará en las conclusiones. Pero si existe un porcentaje de bits errados.

Tabla 25

Valores MBSD y BER cifrado AES con modulación BPSK

MBSD	**BER**
0	0.004563

4.5 Resultados Generales de la implementación de los cuatro algoritmos de cifrado

En la Tabla 26 se observan los resultados de las medidas de distorsión MBSD y la medida BER para los cuatro algoritmos en la SDR implementados en las tarjetas USRP-2920 donde se puede observar que el menor valor de distorisón MBSD se da con el algoritmo AES debido a que

es un algoritmo más robusto y conjuntamente con la una tasa de codificación de ½, no se produce distorsión, además que debido al uso del cable MIMO se tiene una distancia máxima de 40cm entre el transmisor y el receptor. El algoritmo en frecuencia es el que le sigue en cuanto a la medida de distorsión MBSD, y el que mayor distorsión presentó fue el cifrado en tiempo con modulación QPSK. En cuanto a la medida BER es AES el que presenta en menor número de bits errados y que el máximo número de bits errados se presenta con el algoritmo de cifrado en frecuencia.

Tabla 26

Valores MBSD y BER de los algoritmos implementados

Cifrado	Modulación	MBSD	BER
Tiempo	QPSK	0.3464	0.05417
	BPSK	0.3436	0.05421
Frecuencia	QPSK	0.0065	0.06392
	BPSK	0.0021	0.0618
DES	QPSK	0.09485	0.01597
	BPSK	0.03172	0.01606
AES	QPSK	0.0227	0.005691
	BPSK	0	0.004563

CAPÍTULO V

5 CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

- Se implemento los algoritmos de cifrado en la tarjeta USRP 2920 que es compatible con el *software* de LabVIEW® o GNU radio, se trabajó en este *hardware* debido a que permitían sincronizar las radios por medio de un cable MIMO, recomendado por la empresa Ettus para aplicaciones de USRP.
- Se analizaron varios sistemas criptográficos empezando por los básicos que tienen una buena respuesta en tiempo de procesamiento, pero tienen un alto nivel de vulnerabilidad, hasta llegar a los sistemas de cifrados actuales que mejoran la seguridad del cifrado, pero a su ves aumenta el tiempo de procesamiento, necesitando equipos de mayor capacidad para su desempeño.
- Durante las primeras pruebas de la radio N210, se requería de dispositivos externos para lograr la sincronización entre transmisor y receptor, ya que no se lograba la decodificación de ningún mensaje al usar modulación digital. De esta manera, se comprobó que la sincronización es uno de los factores de mayor importancia en la SDR.
- En la simulación del cifrado en tiempo, debido a que las tarjetas poseen precisión finita, los valores del BER simulado llegan a cero. Por lo tanto, no hay bits errados, ya que con la relación Eb/No de 10dB el canal simulado se torna ideal.

- Los cifrados basados en combinaciones matemáticas son más vulnerables, ya que las computadoras actuales poseen mayor capacidad de procesamiento. Consecuentemente, este tipo de cifrado se puede decodificar sin la necesidad de equipos más potentes.
- Las llaves de cifrado deben ser lo más complejas posibles para evitar que puedan ser interpretadas fácilmente, ya que todos los sistemas son en algún modo vulnerables.
- Los cifrados de tiempo y frecuencia trabajan con segmentos de voz permutados, por lo que son más vulnerables. Por otro lado, los cifrados digitales como DES y AES son más seguros, debido a que su cifrado es bit a bit, y al momento de cambiar los bits la señal no puede ser recuperada fácilmente.
- En cuanto a los valores del BER en Matlab®, se concluye que con BPSK existen menos bits errados que con QPSK.
- El procesamiento de cada bloque y el tipo de modulación afectan de forma significativa la plataforma Simulink®. Por ejemplo, con BPSK hay una mejor respuesta en el BER y en la medida de distorsión MBSD.
- Se realizaron varias simulaciones de los algoritmos de cifrado en Simulink®, Matlab® y LabVIEW®, obteniendo buenos resultados en cuanto a la calidad de la voz, lo que se pueden observar en los resultados de las medidas MBSD y BER, para implementar los cifrados en la SDR se enviaron señales de 1 segundo para no afectar el tiempo de procesamiento.
- La simulación del cifrado DES en LabVIEW® es más resistente al ruido ya que se estabiliza con una relación Eb/No menor que la simulación en Matlab®.

- La medida MBSD con la implementación del cifrado AES dio como resultado cero distorsiones, por lo tanto, el cifrado AES es más robusto en precisión finita con el codificador convolucional.
- Considerando un canal ideal, donde la relación señal a ruido es de 40dB, se observó que la señal recibida nunca será igual a la señal transmitida debido a todos los bloques de tratamiento de la señal.
- El bloque MathScript de LabVIEW® genera un alto costo en el tiempo de procesamiento.
- Los valores de distorsión obtenidos en la radio presentan un valor mayor que el obtenido con la simulación, para un canal con una relación Eb/No de 3dB, debido a que el medio de transmisión es el aire y no se conoce que está sucediendo en el canal y que otras señales se encuentran en el mismo ambiente.
- Se adaptaron los códigos de cifrado para los diferentes modelos de radios manejados por el CICTE, que a su vez sirvieron como base para la creación de nuevos modelos elaborados en Matlab® y LabVIEW®. De esta manera, se modernizó las SDR, ya que *Ettus Research*, actualmente es parte de la empresa *National Instruments*, pretende utilizar como software principal a LabVIEW® para las radios.
- Al evaluar los sistemas de cifrado implementados se demostró que, a mayor seguridad de los algoritmos, mayor era el tiempo de procesamiento en la SDR, limitando la implementación de cifrados más elaborados. Por lo tanto, para la implementación de los cifrados digitales, los algoritmos más extensos en código del presente estudio, se trabajó de forma paralela con la finalidad de reducir el tiempo de procesamiento.

5.2 Recomendaciones

- Se recomienda adquirir un kit de desarrollo completo para evitar problemas de sincronismo y de soporte.
- Es necesario considerar que las técnicas de criptografía avanzan a pasos agigantados, por lo tanto, se debe actualizar por lo menos cada año los modelos criptográficos para evitar vulneraciones en la seguridad de las transmisiones.
- Con la finalidad de reducir el tiempo de procesamiento y mejorar la calidad de la transmisión, se recomienda trabajar los cifrados simétricos de forma paralela.

5.3 Trabajos Futuros

- En pruebas futuras se recomendaría enviar otros tipos de audio para probar la capacidad de las radios.
- En el futuro se debería probar la seguridad de los cifrados en otras radios y en otras tarjetas USRP.
- En el futuro se pueden utilizar otras formas de sincronismo para la implementación de la SDR que permitan la transmisión un mayor número de bits, y una mayor distancia entre el transmisor y el receptor.
- A furuto se puede optimizar el código mediante la implemtación de los algoritmos de cifrado utilizando bloques propios de LabVIEW® en lugar de los *Mathscrips*.

REFERENCIAS

Alexander M. Wyglinski. (2013). *Digital communication systems engineering with software-defined radio.* Boston: ARTECH HOUSE.

Capt. Paredes, C. D., & Angulo, O. H. (2011). *Diseño y desarrollo de un radio definido por doftware, para el ejército ecuatoriano, mediante la utilización de una tarjeta USRP y la herramienta simulinlk de matlab. Sangolquí, Ecuador.*

Díaz, J. C. (1995). *Criptografía: historia de la escritura cifrada.* Madrid: COMPLUTENSE, S.A.

Díaz, J. C. (2000). Sistemas criptográficos empleados en hispanoamérica. *Complutense de Historia de América*, 57-71.

Ettus Research. (Diciembre de 2011). *Ettus research*. Obtenido de Ettus Research: https://www.ettus.com/sdr-software/detail/usrp-hardware-driver

Ettus Research, A National Instruments Company. (20 de Junio de 2018). *Ettus research, National instruments brand*. Obtenido de Synchronization and MIMO Capability with USRP Devices: https://kb.ettus.com/Synchronization_and_MIMO_Capability_with_USRP_Devices

Fernández, J. R. (2014). *Software defined radio: basic principles and applications*. Facultad de Ingenieria, 3.

Fernández, M. A. (18 de septiembre de 2008). *Implementación de técnicas de estimación y sincronización para sistemas OFDM. Bellaterra*.

Fernández, S. (2004). *La criptografía clásica.* Apirila: SIGMA.

Fiarresga, V. M. (2010). *Criptografia e matemática.* Lisboa: Universidade de Lisboa, Faculdade de Ciências.

Floissac, N., & L'Hyver, Y. (2011). From AES-128 to AES-192 and AES-256, How to adapt differential fault analysis attacks on keyExpansion. *2011 Workshop on Fault Diagnosis and Tolerance in Cryptography* (págs. 43-53). Pessac: SERMA TECHNOLOGIES ITSEF.

Herrera, J., Riofrio, J., & Vázquez, A. (2015). *Implementación de técnicas de modulación digital mediante un generador de señales arbitrarias. México*.

Jesus J. Ortega Triguero, M. A. (2006). *Introducción a la criptografía: historia y actualidad.* La Mancha: Servicios de Publicaciones de la Universidad de Castilla- La Mancha.

Junior, J. F. (2008). *Criptofonia aplicada a sistemas modernos de comunicações moveis.* Rio de Janeiro: Universidade Federal do Rio de Janeiro.

Junior, J. F. (2008). *Criptofonia aplicada a sistemas modernos de comunicações moveis.* Rio de Janeiro: Universidade Federal do Rio de Janeiro.

Luz, S. D. (4 de enero de 2010). *Criptografía : algoritmos de cifrado de clave simétrica*. Obtenido de https://www.redeszone.net/2010/11/04/criptografia-algoritmos-de-cifrado-de-clave-simetrica/

Martín, C. A. (2016). *Introducción a la criptografía.* Chillán: Universidad de Bío-Bío.

Massachusetts Institute of Technology. (6 de October de 2010). *MIT 6.02 draft lecture notes.* Obtenido de Viterbi Decoding of Convolutional Codes: http://web.mit.edu/6.02/www/f2010/handouts/lectures/L9.pdf

MathWorks. (12 de 03 de 2017). *MathWorks.* Obtenido de GUI de MATLAB: https://es.mathworks.com/discovery/matlab-gui.html

MathWorks. (2018). *MathWorks*. Obtenido de G711 Codec: https://la.mathworks.com/help/dsp/ref/g711codec.html

Meavilla, V. (2004). *La criptografia clasica.* Sigma, 136.

Miguel, J., & Martinez, C. (Junio de 2014). *e-REdING trabajos y proyectos fin de estudios de la E.T.S.I.* Obtenido de MODELADO DE LA CAPA FÍSICA PARA LA ESPECIFICACIÓN PRIME Y ESTUDIO DEL COMPORTAMIENTO FRENTE AL RUIDO: http://bibing.us.es/proyectos/abreproy/12199/fichero/Modelado+de+la+capa+fisica+para+la+especificacion+PRIME+y+estudi%252F5.CAPITULO+3.pdf

Nagurney, L. S. (2009). *Software defined radio in the electrical and computer engineering curriculum. 39th IEEE Frontiers in Education Conference*, (págs. 1-6).

National Instruments Corporation. (13 de Julio de 2017). Software defined radio device. U.S.

Pousa, A. (24 de 09 de 2018). *Algoritmo de cifrado simétrico AES.* Obtenido de http://sedici.unlp.edu.ar/bitstream/handle/10915/4210/Documento_completo.pdf?sequence=1&isAllowed=y

Ratnadewi, R. P. (2009). *Implementation cryptography data encryption standard (DES) and triple data encryption standard (3DES) method in communication system based near field communication (NFC). Journal of Physics: Conference Series, 2-3*.

Ravi Kishore Kodali, D. L. (2013). *DDC and DUC filters in SDR platforms. Conference on Advances in Communication and Control Systems.*

Rodríguez, V. M. (2016). *Diseño e implementación de un transmisor AM, FM y ATSC en radio definida por software para docencia. México.*

SAISO, E. P. (2018). Introducción a la criptografía. *Revista Digital Universitaria (1607 - 6079). Vol. 7, No.7 (2006)*, 1607-6079.

Santos, E. (2010). *Criptografía simétrica*. Obtenido de CRIPTOGRAFÍA SIMÉTRICA: https://www.ugr.es/~esantos/simetrica.html

Selent, D. (2010). *Advanced encrryption standard*. RIVIER ACADEMIC JOURNAL, 1-11.

Tomé, C. (30 de Agosto de 2015). Protocolo criptográfico basado en atributos para almacenamiento en cloud público. Barcelona, Cataluña, España.

Voip-info. (2018). *Voip-info.org*. Obtenido de ITU G.711: https://www.voip-info.org/itu-g711/

Yang, W., & Yantorno, R. (1999). Improvent of MBSD by scaling noise masking threshold and correlation analysis with MOS difference instead of MOS. Philadelphia.