“Responsabilidad con pensamiento positivo”

UNIVERSIDAD TECNOLÓGICA ISRAEL

TRABAJO DE TITULACIÓN EN OPCIÓN AL GRADO DE:

INGENIERO EN ELECTRÓNICA DIGITAL Y TELECOMUNICACIONES

TEMA:

MÓDULO PARA PRÁCTICAS DE LABORATORIO DE COMUNICACIONES II UNIVERSIDAD ISRAEL PARA COMUNICACIÓN DE CÓDIGOS DE LÍNEA.

AUTOR:

WENDY MARISOL ALVARADO SUAREZ

TUTOR:

PHd. ALFONSO ZOZAYA

QUITO, ECUADOR

2018

UNIVERSIDAD TECNOLÓGICA ISRAEL

AUTORÍA DE TRABAJO DE TITULACIÓN

El abajo firmante, en calidad de estudiante de la carrera de Electrónica y Telecomunicaciones, declaro que los contenidos de este Trabajo de Titulación requisito previo a la obtención del Grado de Ingeniería en Electrónica Digital y Telecomunicaciones, son absolutamente originales, auténticos y de exclusiva responsabilidad legal y académica del autor.

Quito D.M., Julio del 2018

……..

Wendy Marisol Alvarado Suarez

C.I.: 1716195159

UNIVERSIDAD TECNOLÓGICA ISRAEL

CERTIFICACIÓN DEL TUTOR

En mi calidad de tutor del trabajo de titulación certifico:

Que el trabajo de titulación "**MÓDULO PARA PRÁCTICAS DE LABORATORIO DE COMUNICACIONES II UNIVERSIDAD ISRAEL PARA COMUNICACIÓN DE CÓDIGOS DE LÍNEA**."**,** presentado por la Sra. Wendy Marisol Alvarado Suárez, de la carrera de Electrónica Digital y Telecomunicaciones, reúne los requisitos y méritos suficientes para ser sometido a la evaluación del Tribunal de Grado, que se designe, para su correspondiente estudio y calificación.

Quito D.M. Febrero del 2019

TUTOR

……..
Ing. Alfonso Zozaya, PHd

AGRADECIMIENTO

Agradezco a Dios, a mis padres María y Darío que gracias a sus consejos y palabras de aliento me han ayudado a crecer como persona y a luchar por lo que quiero, gracias por enseñarme valores que me han llevado a alcanzar una gran meta. Los quiero mucho.

A mi hermana Johanna por su apoyo, cariño y por estar en los momentos más importantes de mi vida. Este logro también es de ella.

A mi revisor, Ing. David Cando, Ing. Luis Montoya por el tiempo, dedicación y paciencia en la elaboración de este documento.

Wendy Marisol Alvarado Suárez

DEDICATORIA

Dedico esta Tesis a mis padres María Domitila Suarez Procel y Darío Antonio Alvarado que siempre me apoyaron incondicionalmente en la parte moral y económica para poder llegar a ser un profesional de la Patria.

A mi hermana, cuñado y sobrino que por el apoyo y cariño que siempre me brindaron día a día en el transcurso de cada año de mi carrera Universitaria.

Dedico esta Tesis a Dios que siempre ha estado presente en mi camino.

Wendy Marisol Alvarado Suarez

TABLA DE CONTENIDO

AUTORÍA DE TRABAJO DE TITULACIÓN ii
CERTIFICACIÓN DEL TUTOR iii
AGRADECIMIENTO iv
DEDICATORIA v
TABLA DE CONTENIDO vi
LISTA DE FIGURAS ix
LISTA DE TABLAS x
RESUMEN xii
ABSTRACT xiii
INTRODUCCIÓN 1
Antecedentes de la situación objeto de estudio 1
Planteamiento del problema 1
Justificación del problema 2
Objetivos 2
General 2
Específicos 3
Descripción de capítulos 3
CAPÍTULO I 5
1. FUNDAMENTACIÓN TEÓRICA 5
1.1. Antecedentes de la enseñanza de las Telecomunicaciones. 6
1.2. MARCO TEORICO TECNICO 6
1.2.1. Digitalización de Datos 6
1.2.2. Conversor Analógico –Digital 6
1.2.3. Muestreo 7
1.2.4. Cuantificación 8
1.2.5. Codificación 8
1.3. CODIGOS DE LINEA 9
1.3.1. Código de línea tipo NRZ Polar 12
1.3.2. Código de línea Manchester y Manchester diferencial 12
1.3.3. Código de línea HDB3 14
1.3.4. Códigos de línea CMI 16
1.3.5. Codificación NRZI 17
1.4. ARDUINO 18
1.4.1. Arduino UNO R3 18

1.5. MATLAB **20**
1.6. MEDIOS DE TRANSMISION. **21**
1.6.1. Comunicación Inalámbrica **21**
1.7. ESPECTRO ELECTROMAGNETICO **22**
1.8. OSCILOSCOPIO **23**
1.9. USB - BUS SERIAL UNIVERSAL **24**
1.9.1. Características del USB **25**
1.9.2. Tipo de transmisión del cable USB **26**
1.9.3. Codificación USB **27**
1.10. Módulos Emisor y Receptor RF 433 MHz. **28**
1.11. Modulación ASK **29**
1.12. Ancho de Banda **30**
1.13. Adaptación de la señal a picos negativos Arduino: **30**
1.13. Proteus ISIS_ARES **32**
CAPÍTULO II **32**
2.1. MARCO METODOLOGICO **32**
2.2. Metodología **32**
2.3. TÉCNICAS DE RECOLECCIÓN DE INFORMACIÓN. **32**
2.3.1. Técnica inicial de fuentes virtuales **35**
2.4. VARIABLES **35**
2.4.1. Variables independientes y dependientes **35**
2.5. Metodología del proceso investigativo **36**
CAPÍTULO III **35**
3. PROPUESTA **35**
3.1. Descripción General **35**
3.2. Descripción de módulos **38**
3.3. Aspectos Técnicos del Trasmisor y Receptor **41**
3.4. Software **46**
3.5. Análisis de costos y tiempo **47**
3.6. Justificación de hardware y software **49**
3.7. Ventajas de los Módulos **50**
CAPÍTULO 4 ..... **51**
4. IMPLEMENTACIÒN **51**
4.1. Desarrollo **51**
4.2. Proceso de construcción del módulo Trasmisor **52**

4.3. Construcción placas de circuitos 53
4.4. Materiales utilizados en un circuito impreso 53
4.5. Proceso de construcción del módulo Receptor 56
4.6. Implementación 59
4.7. Pruebas de Funcionamiento 61
5. CONCLUSIONES 68
6. REFERENCIAS 70

LISTA DE FIGURAS

Figura 1.1 Digitalizacion de la informacion 6
Figura 1.2 Muestreo de una señal contínua 8
Figura 1.3 Codificación de una señal cuantificada 9
Figura 1.4 Códigos de línea más utilizados 11
Figura 1.5 Códigos de Línea Manchester 13
Figura 1.6 Codificación Manchester y Manchester Diferencial 14
Figura 1.7 Códigos de línea HDB3 y AMI 16
Figura 1.8 Codificación CMI 17
Figura 1.9 Codificación NRZI 18
Figura 1.10 Especificaciones Arduino UNO R3 19
Figura 1.11 Espacio de Trabajo Matlab 20
Figura 1.12 Rango de frecuencias del espectro electromagnético 22
Figura 1.13 Osciloscopio Ejm. 24
Figura 1.14 Cables y conectores USB 25
Figura 1.15 Esquema de transmisión asíncrona 27
Figura 1.16 Codificación USB 28
Figura 1.17 Módulos Emisor y Receptor RF 433mhz. 28
Figura 1.18 Modulación ASK 29
Figura 1.19 Amplificador operacional LM741 31
Figura 1.20 Ambiente Proteus 33
Figura 1.21 Ambiente ISIS 33
Figura 3.1 Elemento básicos del proyecto 35
Figura 3.2 Algoritmo Interfaz gráfica Tx 39
Figura 3.3 Algoritmo Interfáz gráfica Rx 40
Figura 3.4 Algoritmo Arduino Tx 40
Figura 3.5 Algoritmo Arduino Rx 41
Figura 3.6 Detalle de actividades a realizar en el presente proyecto 49
Figura 3.7 Detalle en porcentaje de actividades a realizar en el presente proyecto 49
Figura 4.1 Construcción Modulo Transmisor y Receptor 51
Figura 4.2 Módulo Transmisor 52
Figura 4.3 Placa diseñada para Tx 53
Figura 4.4 Elaboración de Placa Baquelita 54
Figura 4.5 Módulo receptor 56
Figura 4.6 Módulo receptor con osciloscopio 57
Figura 4.7 Circuito Receptor en programa Proteus Ares 57
Figura 4.8 Placa diseñada para Rx 59
Figura 4.9 Interfáz de Transmisión 60
Figura 4.10 Interfáz de Recepción 60
Figura 4.11 Código NRZ a enviar 62
Figura 4.12 Códigos NRZ que se debe recibir 62
Figura 4.13 código NRZ que se observa en el receptor 63
Figura 4.14 código NRZ visualización en Osciloscopio 64
Figura 4.15 código MANCHESTER DIFERENCIAL a enviar 65
Figura 4.16 Códigos MANCHESTER DIFERENCIAL que se debería recibir 65
Figura 4.17 Código MANCHESTER DIFERENCIAL que se recibe 66
Figura 4.18 Código MANCHESTER DIFERENCIAL visualización en osciloscopio 66

LISTA DE TABLAS

Tabla 1. 1 Reglas de sustitución HDB3 15
Tabla 2.1. Técnicas y métodos de investigación 36
Tabla 3. 1 Características técnicas del PC. 41
Tabla 3. 2. Características técnicas del Arduino UNO 42
Tabla 3. 3. Características técnicas del módulo transmisor RF 42
Tabla 3. 4 Características técnicas del módulo receptor RF 43
Tabla 3. 5 Costos de implementación. 47

LISTA DE ECUACIONES

Ecuación 1.1 Formula de Nyquist 7
Ecuación 1.2 Ancho de Banda 30
Ecuación 1.3 Voltaje de Salido LM741 32
Ecuación 3.1 ley de Ohm 46

RESUMEN

En la actualidad, prácticamente toda la información que se encuentra disponible está en forma digital “unos” y “ceros”, o se la puede digitalizar para su difusión con gran facilidad gracias al avance actual de la tecnología, para la transmisión de esta información digital de una manera eficiente es necesaria la aplicación de códigos de línea sobre la cadena de bits a enviar.

El buen manejo de estos conceptos es indispensable para el desarrollo de un profesional en el ámbito de las Telecomunicaciones, sin embargo, para un aprendizaje completo del tema, no basta con la teoría, es necesario ejemplificar los conceptos mediante un manejo práctico de los criterios desarrollados en el aula de clase, bajo esta premisa, el presente proyecto tiene como objetivo el desarrollo de un módulo de prácticas de laboratorio para el afianzar los conocimientos sobre comunicación de códigos de línea.

Dicho módulo fue desarrollado mediante la utilización de la plataforma de prototipos electrónicos de código abierto, Arduino, debido a la gran tendencia mundial de su uso y su basta comunidad de desarrolladores que permitirá que el módulo sea escalable a nuevas tecnologías, y de esta manera alargar así su vida útil

PALABRAS CLAVES: Telecomunicaciones, Códigos de línea, Arduino.

ABSTRACT

Actually all the information that we have available around us is in digital form "ones" and "zeros", or it can be digitized for its diffusion with great ease thanks to the current advance of the technology, for the transmission of this digital information in an efficient way is necessary the application of line codes on the string of bits to send.

The good management of these concepts is essential for the development of a professional in the field of Telecommunications, however, for a complete learning of the subject, theory is not enough, it is necessary to exemplify the concepts through a practical management of the developed on subjects, under this premise, the present project has the objective of developing a module of laboratory practices to strengthen the knowledge on communication of line codes.

This module was developed using the platform of open source electronic prototypes, Arduino, due to the great worldwide trend of its use and its vast community of developers that will allow the module to be scalable to new technologies, extending its useful life

KEY WORDS: Telecommunications, line codes, Arduino.

INTRODUCCIÓN

Antecedentes de la situación objeto de estudio

A partir de la creación de la carrera de Ingeniería en Electrónica Digital y Telecomunicaciones en la Universidad Tecnológica Israel el estudio de los códigos de línea ha sido de vital importancia para que los estudiantes comprendan la importancia de la codificación en un sistema de comunicaciones. Si bien en las últimas dos décadas estos estudios no han sido profundizados es importante conocer los avances que se tiene en la actualidad en base a códigos de línea, como ejemplo se conoce que en (Zapata y Otros, 2012) se realiza un módulo para la enseñanza de códigos de línea, mediante el uso de programación en labview, y equipos de la marca EDIBON.

Este ejemplo en particular ayuda a comprender que el desarrollo de una implementación similar a la empleada tendrá que utilizar recursos económicos elevados, es por ello que en la universidad se verifica la necesidad de implementar módulos de prácticas de laboratorio basados en prototipos electrónicos de código abierto.

Planteamiento del problema

Actualmente los estudiantes de la Carrera de Ingeniería en Electrónica Digital y Telecomunicaciones que cursan el séptimo semestre, toman la asignatura de Comunicaciones II, donde aprenden lo referente a códigos de línea, este tema es abordado exclusivamente de forma teórica, ya que genera dificultad en su entendimiento por parte de los alumnos.

La enseñanza de las telecomunicaciones se ha desarrollado tradicionalmente de manera teórico-práctica por su naturaleza experimental. En este sentido, el laboratorio de Comunicaciones II de la Universidad Israel en la carrera de Electrónica y Telecomunicaciones cumple con esta función esencial como ambiente de aprendizaje para la ejecución de trabajos

prácticos. Sin embargo, el aporte en la actualidad en cuanto a la enseñanza de comunicación de códigos de línea de una manera demostrativa es escasa.

Justificación del problema

La utilidad de los trabajos prácticos que se pueden realizar en el laboratorio con la utilización del módulo demostrativo de comunicaciones de datos de códigos de línea como parte de la enseñanza de este tema, se puede analizar en base a resultados que se obtengan, mediante el desarrollo de una forma particular de aprendizaje coherente con un potencial didáctico como aporte al laboratorio, debido a que en un extremo del módulo se ingresarán los datos con la ayuda de un software de 5 tipos de códigos de línea en un computador donde por medio de programación de la tecnología Arduino y con la ayuda de un circuito de radio frecuencia transmita los mismos a él módulos de recepción donde se podrá visualizar en un computador si los datos se trasmitieron en su totalidad por medio de una gráfica comparativa a la gráfica de los datos iniciales, también se visualizará los datos en un osciloscopio donde se podrá apreciar de igual manera características de datos ingresados inicialmente.

En dicho módulo para prácticas el estudiante podrá integrar el conocimiento conceptual con lo metodológico y de esta forma conseguir ampliar la visión en cuanto a los códigos de línea al realizar 3 prácticas en el laboratorio en los cuales se ejecutarán de la siguiente manera:

- En la práctica N°1 constará el código de línea del tipo NRZ Polar.
- Práctica N°2 conformarán los códigos de línea Manchester y Manchester Diferencial.
- Práctica N°3 se realizará de los códigos de línea HDB3 y CMI.

Objetivos

General

- Implementar un módulo para prácticas de Laboratorio de Comunicaciones de la Universidad Israel para comunicación de códigos de línea.

Específicos

- Establecer parámetros del funcionamiento del módulo para prácticas del Laboratorio de Comunicaciones de la Universidad Israel para comunicación de códigos de línea

- Definir los dispositivos según las necesidades del módulo de comunicación de datos.

- Diseñar los circuitos que faciliten el envío y recepción de datos de 5 tipos de códigos de línea exponiendo su funcionalidad para identificación gráfica.

- Desarrollar el algoritmo de programación para él envió de datos en forma de paquetes con tecnología Arduino.

- Implementar 3 guías de Laboratorio las cuales se centrarán en actividades verificativas, a prueba de errores y manipulación del módulo.

- Validar el módulo implementado mediante la ejecución de las 3 prácticas propuestas.

Descripción de capítulos

El presente documento contiene cuatro capítulos en los cuales se detalla todo el desarrollo realizado para cumplir con las metas y objetivos que fueron descritos en el apartado anterior. En términos generales en la documentación se encontrará aspectos como: fundamentación teórica, metodología experimental, propuesta, implementación, resultados, conclusiones y recomendaciones.

Específicamente, en el primer capítulo se presenta la fundamentación teórica para el desarrollo del proyecto, describiendo de forma general el proyecto y el estado del arte del mismo. En este capítulo se identifica los elementos que componen el proyecto, de tal manera que al compactar todos los elementos mencionados hacen que el módulo para prácticas de laboratorio tenga un óptimo funcionamiento que permita la correcta enseñanza de códigos de línea a los alumnos de Comunicaciones II.

Posteriormente en el capítulo dos, se describe de una manera detallada la metodología empleada para el desarrollo del presente proyecto.

A continuación, en el capítulo tres, se detalla la propuesta general realizada para cumplir con el desarrollo del proyecto y se realiza el estudio técnico profundo del producto en desarrollo es decir se hace un análisis técnico de todos los componentes que conforman el módulo de prácticas de laboratorio para con esta información tener un apropiado ensamblaje y funcionamiento del mismo.

Finalmente, en el capítulo cuatro se explica todo lo referente a la implementación y desarrollo práctico del proyecto, describiendo sobre todo el proceso de construcción, diseño, e implementación del mencionado proyecto en cuanto a Hardware y Software. También se realiza una descripción de todas las pruebas realizadas desde el momento de la construcción del módulo hasta la obtención del producto final.

Una vez culminada la implementación, se plantean las conclusiones y recomendaciones obtenidas durante la elaboración del proyecto tanto en el desarrollo práctico como en el desarrollo teórico debido a que los dos procedimientos complementan el proyecto final de titulación.

CAPÍTULO I

1. FUNDAMENTACIÓN TEÓRICA

1.1. Antecedentes de la enseñanza de las Telecomunicaciones.

Las nuevas tecnologías de la información y comunicación se dan bajo el desarrollo de tres medios que son la informática, la microelectrónica y las Telecomunicaciones, para que su desarrollo sea más significativo, debe realizarse de manera interactiva e interconectada **(Cabero, 2005)**.

Una enseñanza tradicional de las nuevas tecnologías, en donde el alumno recibe la información que el profesor transmite, se torna obsoleta para lograr el correcto aprendizaje de la nueva era, es por esto que se debe buscar un complemento adecuado entre la teoría impartida y la práctica ejemplificada de los conceptos, solo de esta forma se logrará un dominio de los conceptos recibidos. **(Belloch, 2012)**.

La Universidad Israel, con una visión futurista, apoya al desarrollo de las nuevas tecnologías, imparte a sus alumnos de la carrera de Electrónica Digital y Telecomunicaciones, la asignatura de Comunicaciones II, donde los alumnos adquieren conocimientos trascendentales para la formación de profesionales en el ámbito de las Telecomunicaciones. Este proyecto busca potenciar el correcto aprendizaje de estos temas, mediante la construcción de un módulo para prácticas de laboratorio.

1.2. MARCO TEORICO TECNICO

1.2.1. Digitalización de Datos

Como el mundo real es analógico, la forma de convertir lo real a lo digital, es mediante los llamados conversores Analógico – Digital (ADC- Analogue to Digital Converter). Estos circuitos tienen como finalidad conseguir una relación biunívoca entre una señal analógica y una digital o viceversa. **(DET, 2018)**

En la Figura 1.1 se observa el proceso general que se cumple para realizar una digitalización de la información.

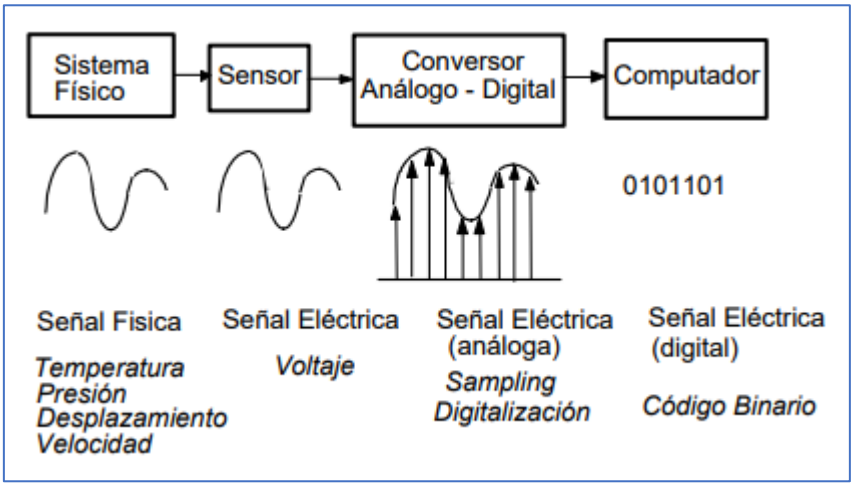


Figura 1.1 Digitalización de la información

Fuente: (Huircán)

1.2.2. Conversor Analógico –Digital

El desarrollo de las nuevas tecnologías como microprocesadores y procesadores digitales de señal (DSP – Digital Signal Processesor), ha permitido que actualmente se trabaje en su mayoría con sistemas electrónicos digitales.

El objetivo de un Conversor Analógico- Digital (ADC) es transformar una señal eléctrica análoga en un número digital equivalente. De la misma manera, un conversor Digital –

Analógico (DAC Digital to Analogue Converter) transforma un número digital en una señal eléctrica analógica. **(Huircán)**

Todo ADC tiene las tres etapas que son muestreo, cuantificación y codificación

1.2.3. Muestreo

El muestreo es la cantidad de veces que se mide el valor de la señal en un periodo de tiempo como se puede observar en la figura 1.2.

Según el teorema de Nyquist-Shannon Ecuación 1.1 la cantidad de veces que se debe medir una señal para no perder información debe de ser al menos el doble de la frecuencia máxima que alcanza dicha señal, a continuación, se detalla la frecuencia de Nyquist:

$$f_s \geq 2 f_{\max}$$

Ecuación 1.1 Fórmula de Nyquist

Fuente: (Musiki, 2019)

Toda la tecnología digital (ejm, audio, video) está basado en la técnica de muestreo (sampling en inglés). En música, cuando una grabadora digital toma una muestra, básicamente toma una fotografía fija de la forma de onda y la convierte en bits, los cuales pueden ser almacenados y procesados. Comparado con la grabación analógica, la cual está basada en registros de voltaje como patrones de magnetización en las partículas de óxido de la cinta magnética. El muestreo digital convierte el voltaje en números (0s y 1s) los cuales pueden ser fácilmente representados y vueltos nuevamente a su forma original.

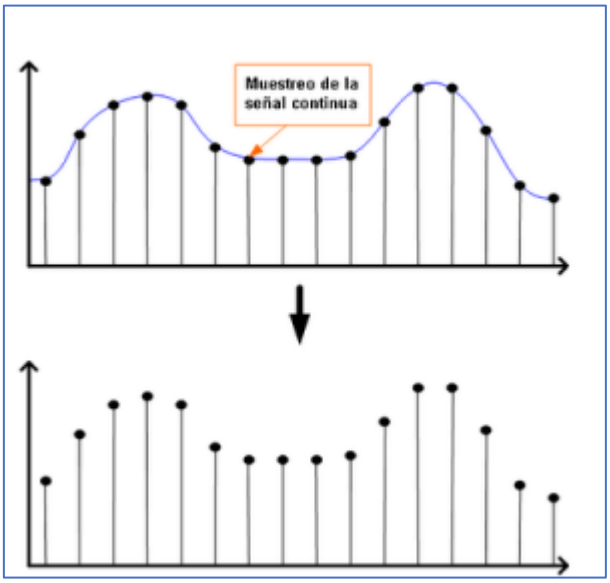


Figura 1.2 Muestreo de una señal contínua

Fuente: (Eveliux, s.f.)

1.2.4. Cuantificación

La cuantificación es el proceso de convertir valores continuos en series de valores discretos. Es decir que la cuantificación es el número de símbolos que se utiliza para guardar una medida de una señal. Mientras que el muestreo representa el tiempo de captura de una señal, la cuantificación es el componente amplitud del muestreo. En otras palabras, mientras que el muestreo mide el periodo de tiempo, la cuantificación transforma este dato a un valor numérico.

Para hacer esto, la amplitud de la señal es representada en una serie de pasos discretos. Cada paso está dado entonces por un número en código binario que digitalmente codifica el nivel de la señal. La longitud de la palabra determina la calidad de la representación.

1.2.5. Codificación

La codificación es la representación numérica de la cuantificación con la utilización de códigos ya establecidos y estándares. El código que frecuentemente se utiliza es el código

binario, pero también existen otros tipos de códigos que son empleados. Como se puede observar en la Figura 1.3, la codificación en términos simples es asignar un código establecido a cada nivel de la señal ya cuantificada.

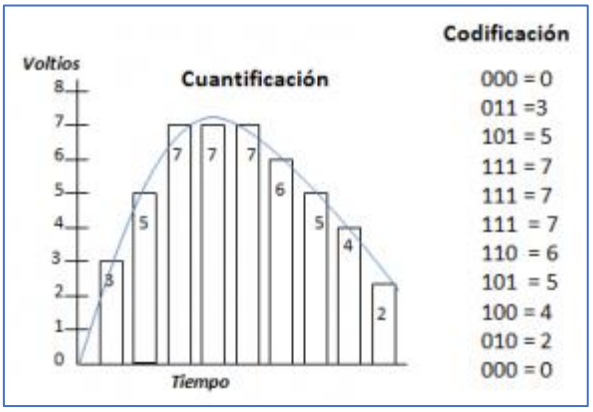


Figura 2.3 Codificación de una señal cuantificada

Fuente: (Oir, ver y contar, s.f.)

1.3. CODIGOS DE LINEA

Para trasmitir señales digitales a través de medios de transmisión es necesario convertir la secuencia de códigos o símbolos, a una forma de onda de banda base denominada códigos de línea, donde cada pulso es un elemento de la señal. La información digitalizada en forma binaria se transmite al codificar cada bit de datos en cada elemento de la señal.

Aunque existen numerosos códigos de línea, los más conocidos son los llamados Retornos a Cero (RZ) y no retorno a cero (NRZ). El código NRZ mantiene constante el nivel de uno y cero durante todo el intervalo de bit. Si es polar, el uno y el cero tienen representaciones opuestas. En la codificación RZ, a la mitad del intervalo de bit el nivel de uno o del cero va a cero. También es conocido el código Manchester donde el símbolo uno se representa por medio de un pulso positivo seguido de uno negativo, ambos de igual amplitud y de medio símbolo de anchura; para el símbolo cero las polaridades de estos pulsos se invierten.

En la figura 1.4 se puede observar un ejemplo de lo anteriormente mencionado

Una línea de formato de codificación consiste en una línea de código específica de una cadena de dígitos binarios que se transforma en una forma de onda de código de línea. Hay dos clases principales de códigos de línea: códigos binarios y códigos de nivel de transición.

Los códigos de nivel transportan la información en su nivel de tensión o voltaje, que puede ser alta o baja (positivo o negativo) para un período completo de bits o para parte del período de bit.

Los códigos de nivel generalmente son instantáneos, ya que codifican un dígito binario en una forma de onda distinta, sin importar el dato binario previo. En su mayoría los códigos de nivel no presentan memoria. Mientras que los códigos de transición llevan la información en el nivel que representa la forma de onda del código de línea.

Los códigos de transición también pueden ser instantáneos, pero generalmente tienen memoria, es decir, su forma de onda depende del dato binario previo. Hay dos formas comunes de códigos de línea de nivel: uno se llama retorno a cero (RZ) y el otro se lo conoce como retención a cero o de no retorno a cero (NRZ). En la codificación RZ, el nivel del pulso vuelve a cero para una porción del periodo de bit. En la codificación NRZ, el nivel del pulso se mantiene durante todo el periodo de bit.

Los códigos de línea se clasifican de acuerdo a la polaridad de los niveles de tensión o voltaje utilizados para representar los datos. Si sólo se utiliza una polaridad de nivel de tensión, es decir, positivo o negativo (además del nivel cero), entonces se llama codificación unipolar. Si se utilizan ambos niveles de tensión positivos y negativos, con o sin un nivel de voltaje cero, entonces se llama señalización polar. **(Brito, 2014)**. En la Figura 1.4 se muestran los códigos de línea que se utilizan continuamente.

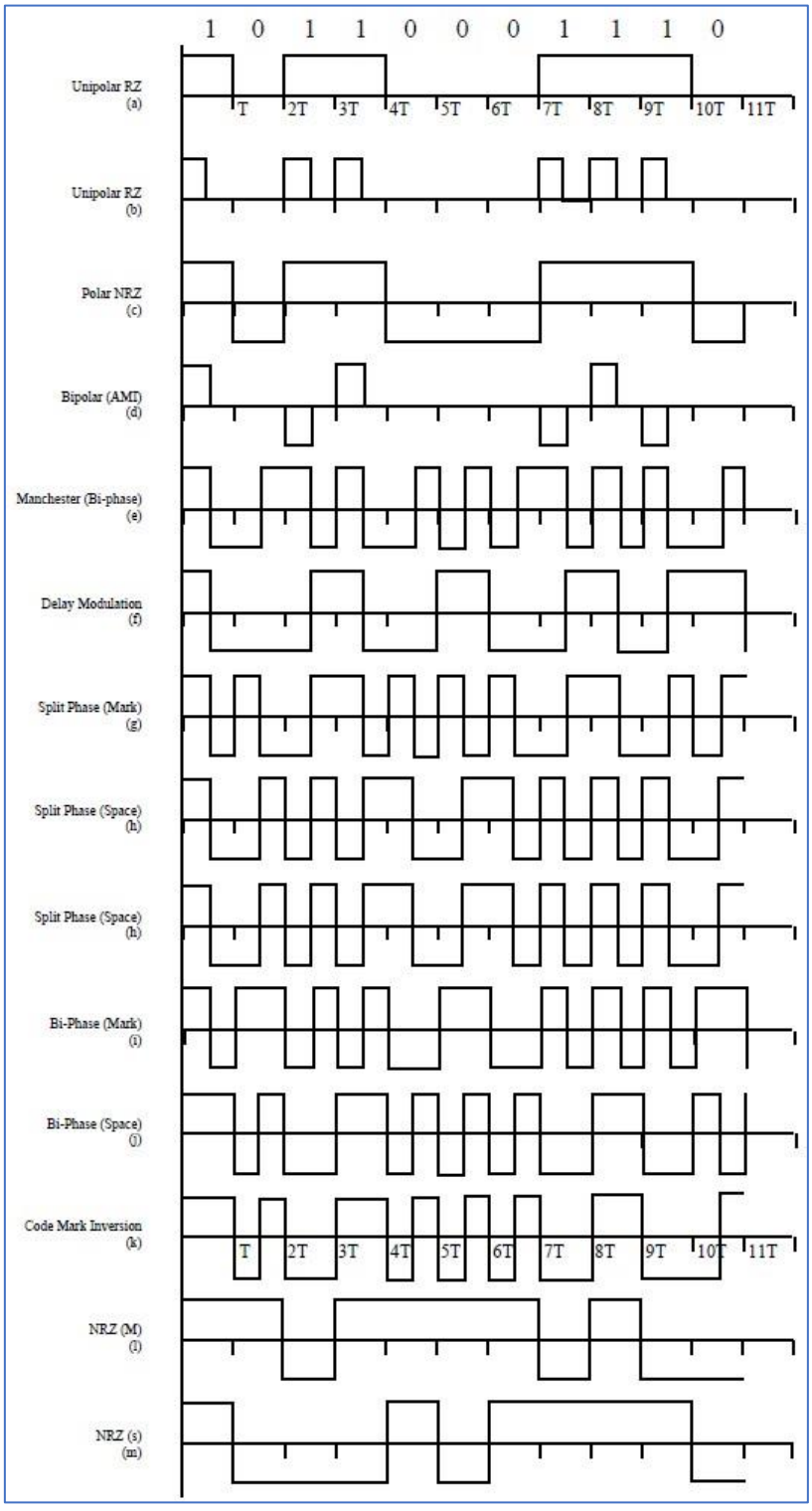


Figura 1.4 Códigos de línea más utilizados

Fuente: (Brito, 2014)

1.3.1. Código de línea tipo NRZ Polar

El tipo de código de línea tipo NRZ Polar, se caracteriza al representar el "1" binario por un voltaje positivo $+Vy$, mientras que el "0" binario se representa por un voltaje negativo $-Vy$ durante el período de bit completo. Este tipo de código se lo conoce también como NRZ (L), ya que un bit se ve representado mediante el mantenimiento de un nivel (L) durante su periodo completo. Un ejemplo de una secuencia binaria mediante el uso del código Polar NRZ se puede observar en la figura 1.4.

Este tipo de código es de nivel instantáneo, donde alternativamente, un "1" binario se puede representar por un nivel de voltaje $-Vy$, mientras que un "0" binario se puede representar por un nivel de voltaje $+Vy$, sin que cambien las características espectrales y de rendimiento del código de línea **(Brito, 2014)**.

1.3.2. Código de línea Manchester y Manchester diferencial

El código de línea Manchester, es un método de codificación de una señal binaria en el que en cada periodo de bit hay una transición entre dos niveles de señal. Es una codificación auto sincronizada, ya que en cada bit se puede obtener la señal de sincronización de reloj, lo que hace posible una sincronización exacta de la cadena binaria a trasmitir.

La codificación Manchester o hackeo provee una forma simple de codificar secuencias de bits, incluso cuando hay largas secuencias de periodos sin transiciones de nivel que puedan significar la pérdida de sincronización, o incluso errores en las secuencias de bits. Esta codificación también asegura que la componente continua de las señales sea cero si se emplean valores positivos y negativos para representar los niveles de la señal, este proceso hace más fácil la regeneración de la señal, y evita las pérdidas de energía de las señales. **(Atom, 2018)**

En la figura 1.5 se observa una clara representación de una cadena binaria transmitida mediante códigos de línea Manchester.

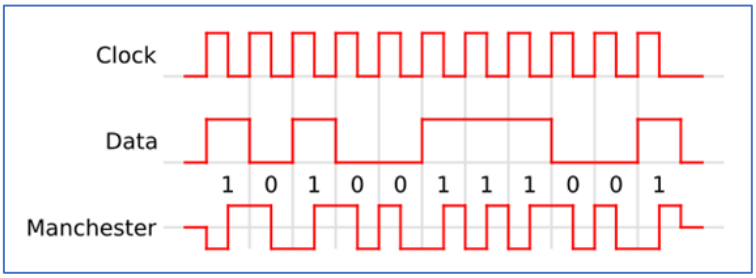


Figura 1. 5 Códigos de Línea Manchester

Fuente: (Atom, 2018)

Al utilizar la codificación de línea Manchester Diferencia, la transmisión a mitad del intervalo se utiliza tan únicamente para proporcionar sincronización:

0: Transición al principio del intervalo del bit

1: Ausencia de transición al principio del intervalo del bit

Al hablar de Manchester Diferencial se cita los siguientes aspectos importantes:

- Utiliza un esquema de codificación diferencial.
- La componente continua siempre es nula, independientemente de la proporción de unos y ceros que contenga la secuencia original.
- La ausencia de componente continua es la eliminación de fenómenos de corrosión electrolítica en los conectores y de fallos en los mismos.
- Es utilizado para protocolos IEEE 802.5.

En la figura 1.6, se visualiza de una manera más gráfica la diferencia existente entre la codificación Manchester, y el código de línea Manchester Diferecial.

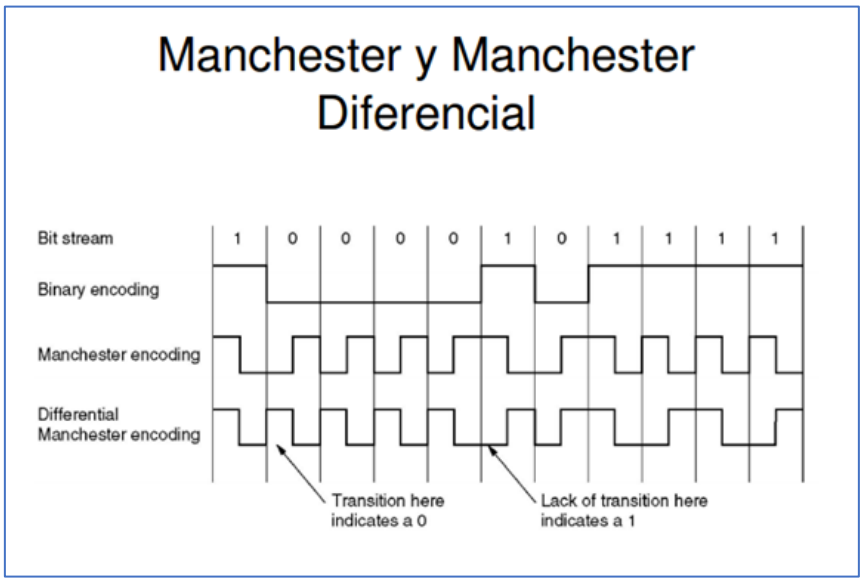


Figura 1.6 Codificación Manchester y Manchester Diferencial

Fuente: (Atom, 2018)

1.3.3. Código de línea HDB3

Este tipo de códigos HDBn, se utilizan en transmisión, dado su mejor rendimiento respecto a los anteriores. Son códigos Tipo AMI (Alternate Mark Inversion – Inversión de Marcas Alternadas) que modifican las secuencias de múltiples "0" continuos en una cadena y permiten introducir al n+1 "0" (en este caso, al cuarto “0”) un bit de la misma polaridad del último bit generado.

Debido a que una cadena demasiado larga de bits iguales “0” o “1” puede causar un error de conexión entre el receptor y el emisor, mantener cadenas de bits con la misma polaridad va contra la ley de formación del código, este bit es denominado "bit de violación". En recepción, una vez obtenido el sincronismo del código, estos bits añadidos se eliminan para recuperar la señal en su totalidad.

El código de línea más utilizado de este tipo es el HDB3, es decir, cuando se tiene una secuencia de más de tres "0" seguidos; como la secuencia "0000", la codifica mediante el uso de los siguientes parámetros:

- El último bit "0" se convierte en un “bit de violación”, V.

- Se añade a la secuencia un "bit de compensación" B, para eliminar la componente continúa generada por el bit de violación V.

Para ejemplificar lo descrito se puede decir que:

- Una secuencia "0000" se codifica como "000V" cuando el número de "1s" desde el último cambio es impar y cuando su última pulsación haya sido negativa
- La secuencia "0000" se codifica, en cambio, "B00V" cuando el número "1s" desde el último cambio es par y cuando su última pulsación haya sido negativa

La utilización de la codificación HDB3 tiene las siguientes ventajas:

- Carecen de componente continua
- Permite sincronización muy buena.
- Prácticamente toda la potencia de transmisión se acumula en el lóbulo principal.
- Ayuda en detección de errores.

Mientras que como inconvenientes o desventajas de su uso se puede nombrar lo siguiente:

- Para su utilización de requiere una instrumentación compleja
- Tasa de error más alta que códigos AMI simples

(Stiven, 2009)

En la Figura 1.7 se observa ejemplos con codificación mediante HDB3, basada en la Tabla 1.1

Tabla 1. 1 Reglas de sustitución HDB3

	Número de pulsos bipolares (unos) desde la última sustitución.	
Polaridad del pulso anterior.	**Impar**	**Par**
-	000-	+00+
+	000+	-00-

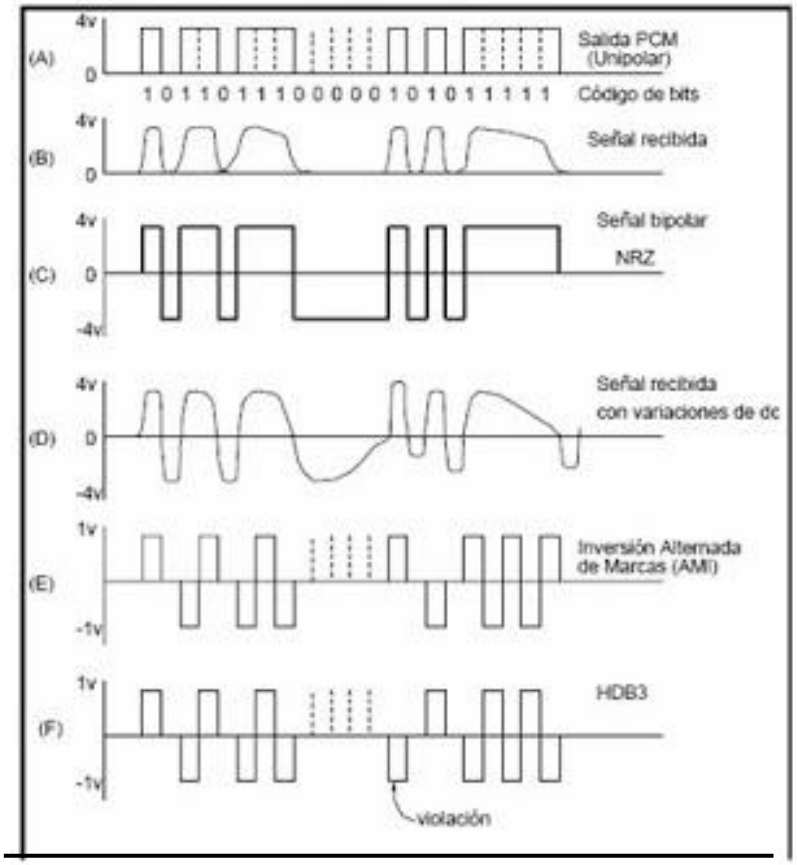


Figura 1.7 Códigos de línea HDB3 y AMI

Fuente: (Stiven, 2009)

1.3.4. Códigos de línea CMI

Para utilizar codificación CMI se aplica la señal de datos NRZ y el bit del reloj de transmisión. A partir de esto último se obtiene también una señal de sincronización de frecuencia doble, precisado para el funcionamiento interno del codificador CMI.

Si el dato en entrada es 0, en la salida CMI se obtiene una forma de onda correspondiente a la señal de clock. Cuando el dato en entrada es 1 a la salida CMI se obtiene una forma de onda con niveles altos y bajos alternados a frecuencia de cifra. Cuando los datos son casualmente 0 ó 1, en la salida CMI se tendrá (fig. 1.8):

· Un periodo de clock en correspondencia del "0"

· Alternativamente un nivel alto o bajo. Por una duración equivalente al intervalo de bit, si el dato es "1".

Obsérvese en la figura 1.8 que el codificador suministra también la salida CMI negada. Las salidas CMI y CMI negada serán conectadas a las entradas correspondientes del transmisor.

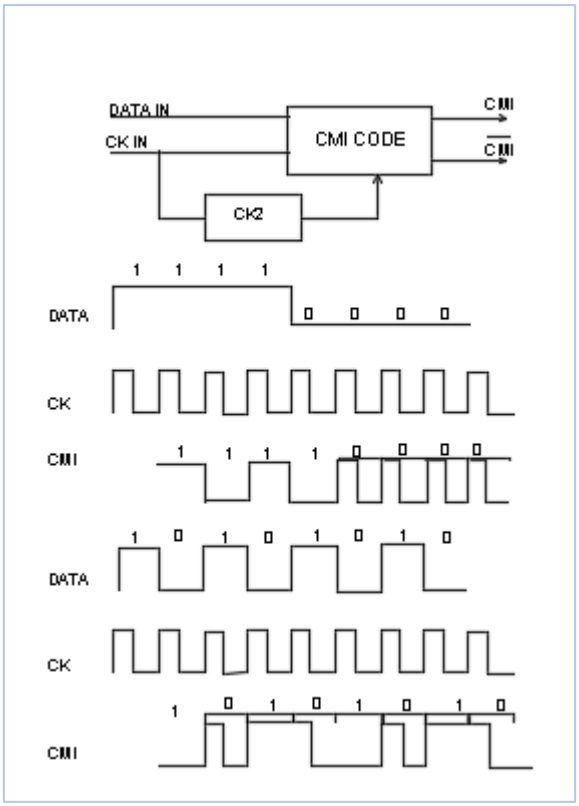


Figura 1. 8 Codificación CMI

Fuente: (Universidad Don Bosco, 2015)

1.3.5. Codificación NRZI

Una variante de la codificación NRZ Polar, es la codificación NRZI (No retorno a cero, invertido). En esta codificación al igual que en la NRZ, se mantiene constante el voltaje durante el periodo de un bit, sin embargo, la codificación de los datos está sujeta a la condición de si existe o no un cambio de la señal al inicio de cada bit. Un uno lógico 1, se codificará mediante un cambio de nivel de la señal al inicio del bit, mientras que la ausencia del cambio significará un cero 0, lo mencionado se puede observar en la figura 1.9.

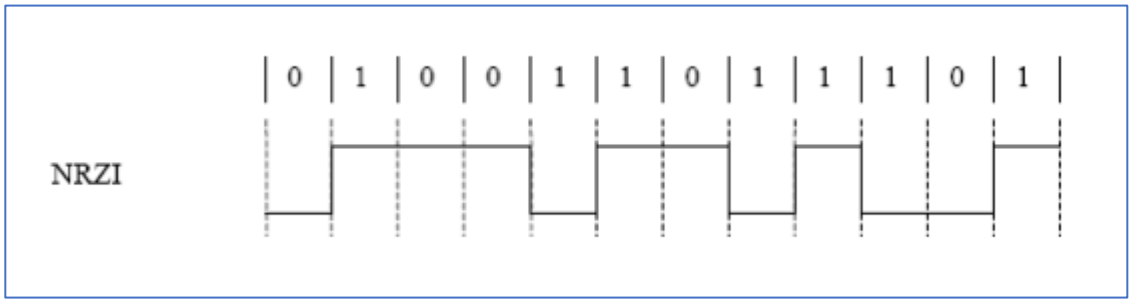


Figura 1.9 Codificación NRZI

Fuente: (Aldaz Corrales & Corrales, 2009)

1.4. ARDUINO

"Arduino es una plataforma de prototipos electrónica de código abierto (open-source) basada en hardware y software flexibles y fáciles de usar." **(Arduino, 2018)**

Actualmente es la principal tendencia de prototipos electrónicos a nivel mundial, tiene una de las más grandes comunidades de desarrolladores que afianzan su continuidad a largo plazo en el desarrollo de prototipos.

Entre las ventajas de utilizar la plataforma de Arduino se encuentran como principales las mencionadas a continuación:

- Bajo costo.
- Multiplataforma (Windows, Macintosh, Linux, etc).
- IDE de programación simple y claro, basado en C.
- Código abierto y software extensible.
- Código abierto y hardware extensible.

(Arduino, 2018)

1.4.1. Arduino UNO R3

El Arduino UNO R3, es la versión mejorada del Arduino UNO convencional, trabaja con el microcontrolador ATmega328. Como diferencia de los modelos anteriores, el Arduino Uno utiliza el ATmega16U2 para el manejo de USB en lugar del 8U2 (o del FTDI encontrado en

generaciones previas). Esto permite ratios de transferencia más rápidos y más memoria. No se necesitan drivers para Linux o Mac (el archivo inf para Windows es necesario y está incluido en el IDE de Arduino).

Las principales características de esta placa son las siguientes:

- Microcontrolador ATmega328.
- Voltaje de entrada 7-12V.
- 14 pines digitales de I/O (6 salidas PWM).
- 6 entradas análogas.
- 32k de memoria Flash.
- Reloj de 16MHz de velocidad.

(Arduino, 2018)

En la Figura 1.10, se puede observar las especificaciones detalladas de la placa Arduino UNO, utilizada para el desarrollo del presente proyecto.

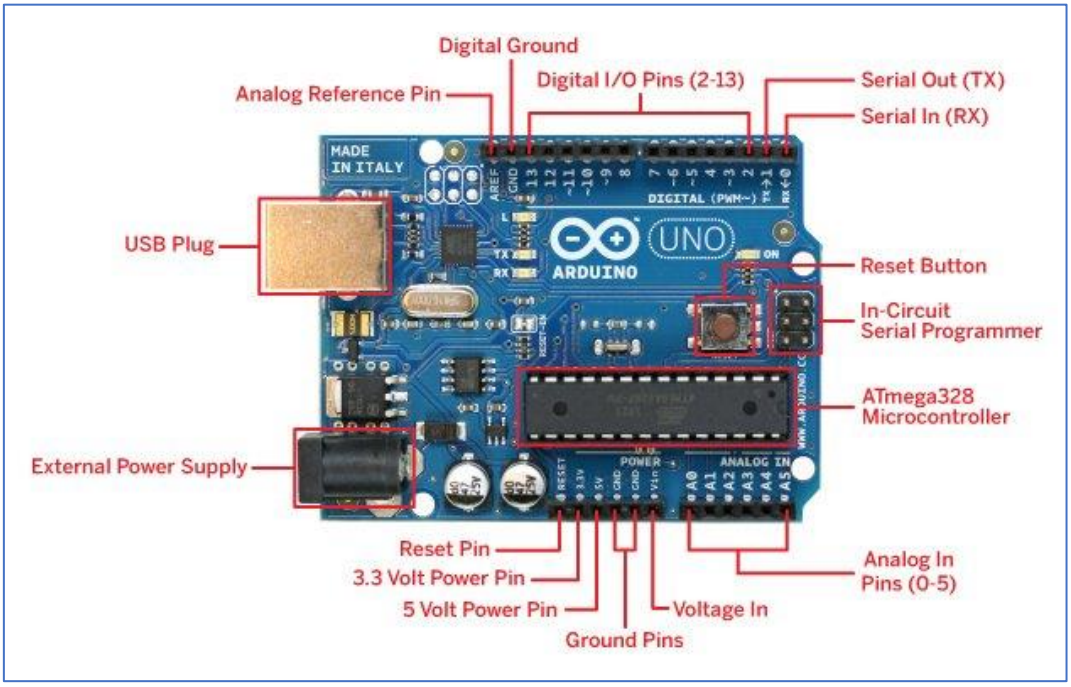


Figura 1.10 Especificaciones Arduino UNO R3

Fuente: (Crespo, 2018)

1.5. MATLAB

MATLAB es el nombre abreviado de “MATriz LABoratory”. Es un programa para realizar cálculos numéricos con vectores y matrices, y por tanto se puede trabajar también con números escalares (tanto reales como complejos), con cadenas de caracteres y con otras estructuras de información más complejas, en la figura 1.11 se puede apreciar el espacio de Trabajo.

MATLAB es un lenguaje de alto rendimiento para cálculos técnicos, es al mismo tiempo un entorno y un lenguaje de programación. Uno de sus puntos fuertes es que permite construir propias herramientas reutilizables. Se puede crear fácilmente funciones y programas especiales (conocidos como M-archivos). Se puede agrupar en un Toolbox (también llamadas librerías): colección especializada de M-archivos para trabajar en clases particulares de problemas. MATLAB, a parte del cálculo matricial y álgebra lineal, también puede manejar polinomios, funciones, ecuaciones diferenciales ordinarias, gráficos.

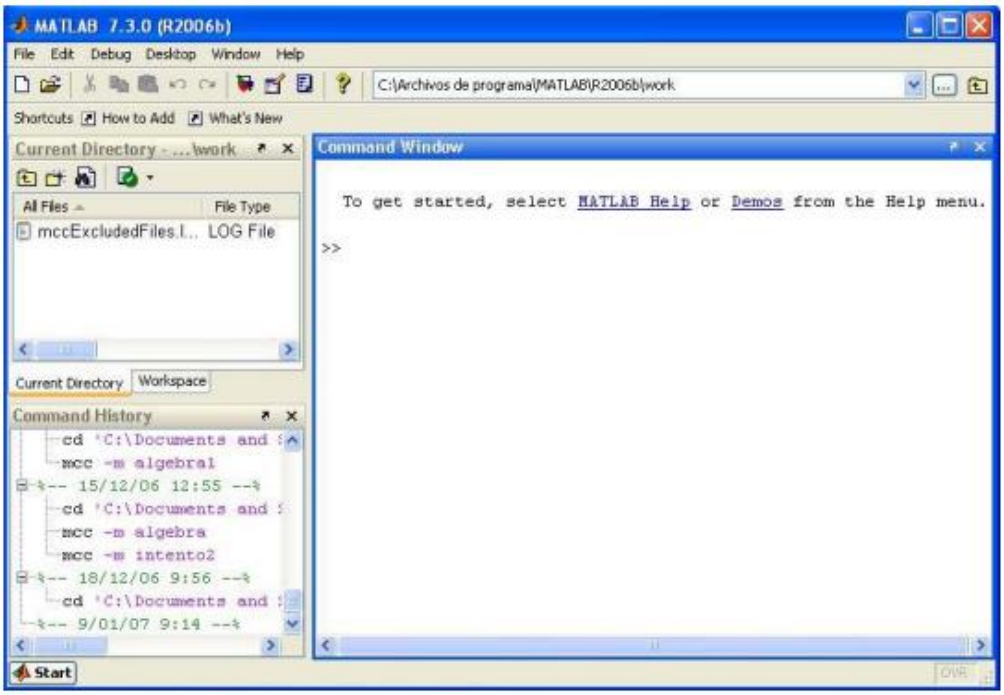


Figura 1.11 Espacio de Trabajo Matlab

Fuente: (Microsoft, 2018)

1.6. MEDIOS DE TRANSMISION.

Al medio de transmisión se lo puede definir como el camino o ruta física que existe entre el transmisor y el receptor. Bajo esta premisa todo medio físico que pueda transportar información mediante señales electromagnéticas puede ser utilizado en las redes de comunicación como un medio de transmisión.

El medio de transmisión depende de sus particularidades intrínsecas y puede ser objeto condicionante para las características de la conexión a realizar, puede influir en parámetros como distancia, velocidad de transporte, topología de la red, y en el seleccionar el tipo de acceso. **(Manuel, 2010)**

Entre los principales medios de transmisión están:

- **Guiados**, cuando las ondas se transmiten confinándolas a lo largo de un camino (medio) físico como por ejemplo un cable.
  Los principales medios guiados emplean cobre y fibra óptica, algunos ejemplos son: el par trenzado, el cable coaxial y el cable de fibra óptica.

- **No guiados** (inalámbricos), la propagación de la señal se hace a través del aire, el mar o el espacio.

1.6.1. Comunicación Inalámbrica

Las comunicaciones inalámbricas han logrado un gran desarrollo que actualmente se habla de una revolución tecnológica de las comunicaciones inalámbricas, un salto tan importante para el desarrollo como en su momento lo fueron la electricidad, la televisión, o el computador. Una de las ventajas de mayor transcendencia de esta tecnología es la movilidad que ofrece a sus usuarios ya que no dependen de un cable o medio tangible.

Para lograr la comunicación inalámbrica se utiliza como medio de propagación el espectro electromagnético, que coloquialmente se denomina aire. **(Prieto)**

1.7. ESPECTRO ELECTROMAGNETICO

"Se denomina espectro electromagnético a la distribución energética del conjunto de ondas electromagnéticas." **(Prieto)**

Como se aprecia en la definición, las ondas electromagnéticas tienen un componente eléctrico y otro magnético, para estudiantes y profesionales de telecomunicaciones la forma de onda que se puede asemejar a esta definición es la luz.

Las frecuencias más utilizadas en las comunicaciones inalámbricas se pueden clasificar en los tipos que se mencionarán a continuación y se las puede observar en la Figura 1.12.

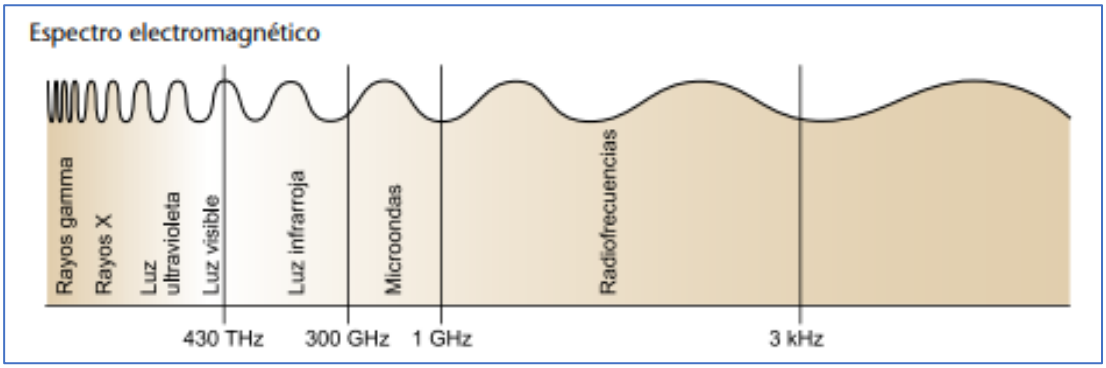


Figura 1.12 Rango de frecuencias del espectro electromagnético

Fuente: (Prieto)

- **Infrarrojos (IR).** Utilizadas en comunicaciones punto a punto de corto alcance, son muy direccionables y no pueden atravesar obstáculos. Este medio se utiliza habitualmente en el mando a distancia de la televisión y también su uso permite tener un sistema de comunicación que se utiliza a menudo para conectar dispositivos situados uno al lado del otro. Es el rango de frecuencia más alto para comunicaciones inalámbricas.

- **Microondas (MW).** Este rango de frecuencias es adecuado para transmisiones de largo recorrido (comunicaciones por satélite, comunicaciones terrestres punto a punto como alternativa al cable coaxial o la fibra óptica, y también la mayoría de las tecnologías

inalámbricas más habituales que existen actualmente como UMTS, Bluetooth o WLAN). Las microondas suelen ser direccionales y utilizan una parte del espectro con frecuencias más pequeñas que los infrarrojos.

- **Radiofrecuencias (RF).** Es el rango que utilizan las transmisiones de radio (FM, AM) y televisión digital terrestre (TDT). Las radiofrecuencias son omnidireccionales y pueden atravesar obstáculos sin ningún problema.

- **Otras frecuencias**. Existen otros rangos de frecuencias del espectro electromagnético, como la luz ultravioleta, los rayos X o los rayos gamma, que tienen mejores prestaciones que los infrarrojos, los microondas y las radiofrecuencias, dada su frecuencia tan alta, pero no se utilizan porque pueden llegar a ser peligrosos para los seres vivos y, además, son difíciles de producir y modular

1.8. OSCILOSCOPIO

El osciloscopio es un instrumento electrónico (Figura 1.13) que permite visualizar de forma gráfica señales eléctricas variables en el tiempo. Los osciloscopios son de los equipos instrumentales más utilizados debido a su versatilidad, pueden utilizarlos desde técnicos para la reparación de televisores, como doctores, para medir fenómenos eléctricos propios del cuerpo humano.

Un osciloscopio puede ser utilizado básicamente para lo siguiente:

- Determinar directamente el periodo y el voltaje de una señal.
- Determinar indirectamente la frecuencia de una señal.
- Determinar que parte de la señal es DC y cual AC.
- Localizar averías en un circuito.
- Medir la fase entre dos señales.
- Determinar que parte de la señal es ruido y como varia este en el tiempo

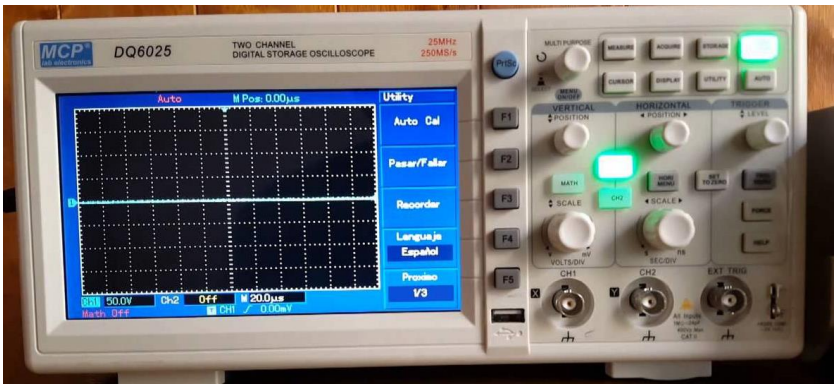


Figura 1.13 Osciloscopio Ejm.

Fuente: (Final Test, 2018)

1.9. USB - BUS SERIAL UNIVERSAL

El USB es una tecnología que permite la conexión de comunicaciones, Soporta velocidades de transmisión bajas en periféricos de transmisión lenta como palancas de juego y mouses, pero también velocidades de transmisión alta (más de 12Mbps) en periféricos especializados que requieren tasas de transmisión mayores como cámaras de video, y dispositivos externos de almacenamiento. Esta especificación fue consensuada por un conjunto de industrias y marcas reconocidas del mercado tecnológico que incluye a empresas como IBM, Compaq, NEC, entre otras. Su principal enfoque fue la incorporación de dispositivos periféricos a ordenadores personales.

En la actualidad el uso de USB ha sido acogido de una manera global debido a sus ventajas de utilización, entre las cuales se nombra las siguientes:

- Fácil expansión de periféricos en la PC: no debe hacer falta más que conectar el periférico y emplearlo. Ni pensar en abrir la computadora.
- Bajo costo para aplicaciones que demandan por encima de los 12Mbps, particularmente aplicaciones y hardware multimediales: micrófonos, parlantes, teléfonos, etc.
- Soporte completo para transmisión en tiempo real de voz, audio y video.

- Flexibilidad de protocolos para transmisiones mixtas isocóricas y asincrónicas (se analiza el tipo de transmisión isocronía más adelante, ya que es el eje de transmisión del Bus Serial Universal y tiene un nivel de conocimiento y difusión relativamente bajo dada su novedad).
- Cómoda integración de dispositivos de tecnologías y fabricantes diferentes.
- Soporte para plataformas diversas de la línea de las PC compatibles
- Posibilitar la producción de nuevos dispositivos capaces de aprovechar sus ventajas

1.9.1. Características del USB

Como se trató en el apartado anterior, la finalidad de esta estandarización fue la de permitir múltiples dispositivos periféricos al computador, es por esto que una de sus principales características y normas que rigen al USB, es que todos los dispositivos sin importar la función que cumplan, o el tipo de periférico que sea, deben tener el mismo tipo de conectores y cables, los mismos que se pueden observar en la Figura 1.14.

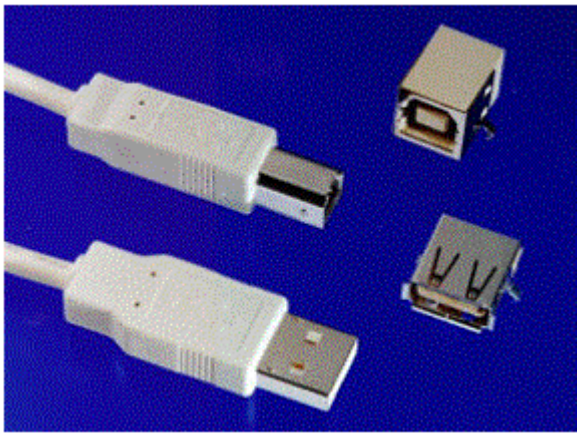


Figura 1.14 Cables y conectores USB

Fuente: (RODRIGUEZ, 2016)

Como un compendio de las características de USB, se puede nombrar las mencionadas a continuación:

- Los detalles de consumo y administración eléctrica del dispositivo deben ser completamente transparentes para el usuario.
- El ordenador debe identificar automáticamente un dispositivo agregado mientras trabaja y configurarlo.
- Los dispositivos pueden ser también desconectados mientras el ordenador está en uso.
- Deben compartir un mismo bus tanto dispositivos que requieren de unos pocos Kbps como los que requieren varios Mbps.
- El bus debe permitir periféricos multifunción, es decir aquellos que pueden realizar varias tareas a la vez, como lo son algunas impresoras que adicionalmente son fotocopiadoras y máquinas de fax.
- Capacidad para manejo y recuperación de errores producidos por un dispositivo cualquiera.
- Bajo costo.

1.9.2. Tipo de transmisión del cable USB

Pese a ser una transmisión serial, el cable USB pose el siguiente tipo de transmisión:

- **ASINCRÓNICA**

La denominación de asíncrona no implica que no exista una señal de sincronismo, sino porque el sincronismo se halla en una señal distinta a la cadena a transmitir, esta señal de reloj o sincronía se encuentra en los equipos mismos. Un ejemplo de este tipo de transmisión se la puede apreciar en la Figura 1.15.

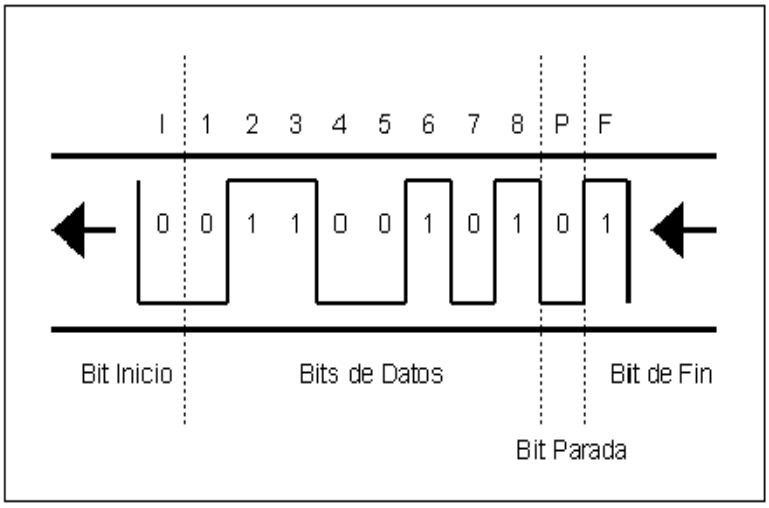


Figura 1.15 Esquema de transmisión asíncrona

Fuente: (RODRIGUEZ, 2016)

1.9.3. Codificación USB

- Mantiene constante el nivel de tensión mientras dura un bit.
- Los datos se codifican mediante la presencia o ausencia de una transición de la señal al principio del intervalo de duración del bit.
- Un 1 se codifica mediante la transición bajo a alto o alto a bajo al principio del intervalo del bit.
- Un cero se representa por la ausencia de transición, es decir, cando el valor del bit es 1, la señal cambia de estado luego de que el reloj lo indica y cuando el valor del bit es 0, la señal no cambia de estado, en esta codificación, la señal cambia de tensión solo si se cambia el valor lógico, en caso contrario la señal se mantiene.
- Se puede observar en la figura 1.17 que al transmitir un 0 no se produce transición y en cambio al enviar un 1 se produce una transición a nivel positivo o negativo.

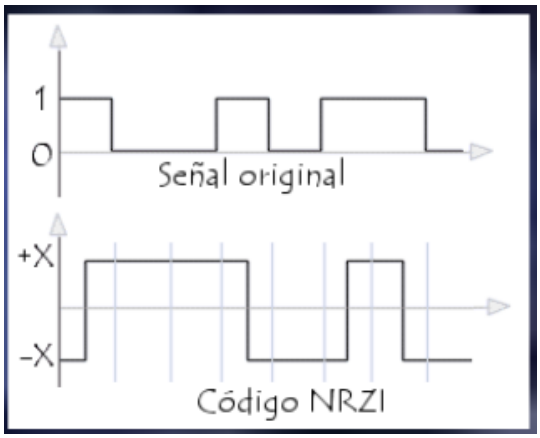


Figura 1.16 Codificación USB

Fuente: (Gomez, 2013)

1.10. Módulos Emisor y Receptor RF 433 MHz.

Estos módulos (figuran 1.18), son utilizados para lograr una comunicación inalámbrica mediante el uso de frecuencias de radio, en este caso particular, la frecuencia de los módulos es de 433MHz debido a ser una banda de libre uso. Para utilizar estos elementos basta con energizarlos y conectar el pin de Datos al microcontrolador a utilizar tanto en el transmisor como en el receptor. Para la programación no se requiere añadir ningún tipo de librerías adicionales, ya que el dispositivo funciona con una comunicación serial "transparente". La antena a pesar que no está incluida de fábrica en estos módulos, tiene una importancia significativa para el alcance, por tanto, se recomienda la utilización de un cable de un mínimo de 2 cm hasta un máximo de 17 cm para poder garantizar la transmisión.

Figura 1.17 Módulos Emisor y Receptor RF 433mhz.

Fuente: (NAYLAMP Mechatronics, 2013)

Las características de fabricación de los módulos RF están desarrollados en base a la frecuencia a la que se desea trabajar, por lo tanto, la distancia que pueden alcanzar los módulos está estrictamente determinados mediante las especificaciones técnicas del fabricante. Sin embargo, al tener en cuenta que los módulos operan con un canal inalámbrico como medio de transmisión, la señal que viaja a través del espectro electromagnético está expuesta a perturbaciones que provocan que mientras más distante se encuentran el transmisor del receptor la señal pierde fuerza y por ende la transmisión pierda información y no sea 100% confiable.

1.11. Modulación ASK

Amplitude Shift Keying (ASK) es un tipo de modulación de amplitud que representa los datos binarios en forma de variaciones en la amplitud de una señal. Cualquier señal modulada tiene una portadora de alta frecuencia. La señal binaria cuando se modula ASK, da un valor cero para la entrada Baja mientras que da salida a la portadora para la entrada Alta. La figura 1.19 representa la forma de onda modulada ASK junto con su entrada.

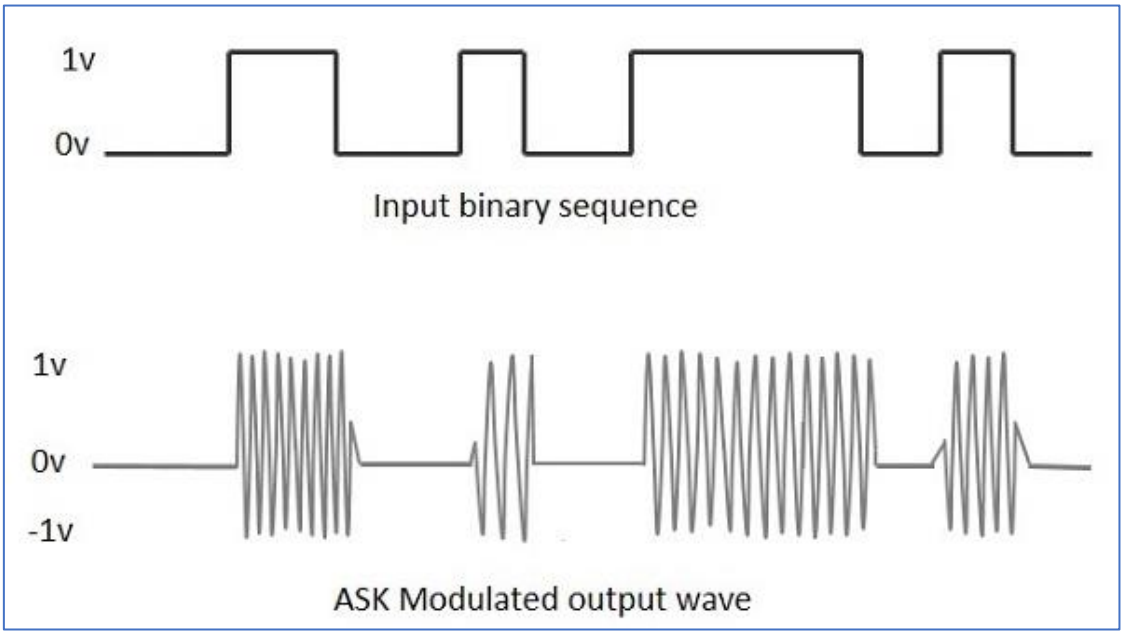


Figura 1.18 Modulación ASK

Fuente: (Tutoriales Point, 2018)

1.12. Ancho de Banda

Para calcular el ancho de banda que ocupan estos módulos, es necesario considerar el tipo de modulación que utilizan los mismos, para efectos de una modulación ASK, el ancho de banda vendrá dado por:

$$\boldsymbol{BW = \left(1 + \ v_{m1}(t)\right) * f_b}$$

Ecuación 1.2 Ancho de Banda

Fuente: Universidad Don Bosco. (2015)

Donde

$\boldsymbol{BW}$: ancho de banda

$\boldsymbol{v_{m1}(t)}$ Señal digital a modular. Puede tomar dos valores: +1 cuando el bit enviado es "1", y -1 cuando el bit enviado es 0.

$\boldsymbol{f_c}$: Frecuencia de la señal portadora.

$\boldsymbol{f_b}$: es la velocidad de transmisión

Por tanto, en base a las características de los módulos de radiofrecuencia descritos en el apartado anterior, se puede calcular el BW de la siguiente manera:

$$\boldsymbol{BW = (1 + \ 1) * 4KB/s}$$

$$\boldsymbol{BW = (1 + \ 1) * 32Kbps}$$

$$\boldsymbol{BW = 64\ Kbps}$$

1.13. Adaptación de la señal a picos negativos Arduino:

Para la adaptación de la señal para picos negativos se utiliza un amplificador operacional LM741 utilizado como comparador.

➢ Amplificador Operacional LM741

Es un dispositivo electrónico que tiene dos entradas y una salida. En esta configuración, la salida del dispositivo es, generalmente, mayor que la diferencia de potencial entre sus entradas.

Características:

- Voltaje de alimentación max: ±22 V
- Ancho de Banda típico: 1 MHz
- Slew rate típico: 0.5 V/μs
- Voltaje offset de entrada típico: 1 mV
- Entradas de ajuste de offset
- Compensado en frecuencia internamente
- Alta ganancia
- Salida protegida contra corto circuito continuo
- Encapsulado: DIP 8 pines

El dispositivo posee dos entradas: una entrada no inversora (+) como se puede observar en la imagen 1.19, en la cual hay una tensión indicada como V+ y otra inversora (–) sometida a una tensión V-

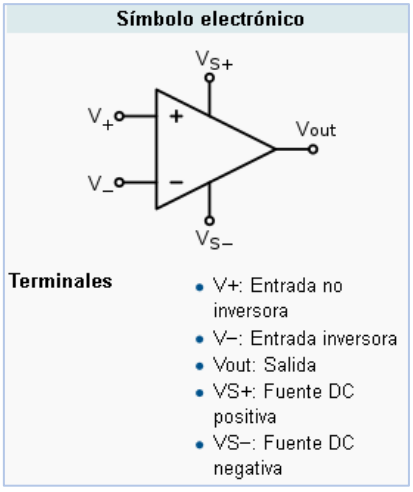


Figura 1.19 Amplificador operacional LM741

Fuente: **(Gomez, 2013)**

Aplicación sin retroalimentación que compara señales entre las dos entradas y presenta una salida en función de qué entrada sea mayor. Se puede usar para adaptar niveles lógicos.

$$\bullet\ V_{\text{out}} = \begin{cases} V_{S+} & V_1 > V_2 \\ V_{S-} & V_1 < V_2 \end{cases}$$

Ecuación 1.3 Voltaje de Salido LM741

Fuente: **(Gomez, 2013)**

Arduino maneja 2 estados 0 (V) y 5 (V) que representan 0 y 1 lógicos , se necesita que mediante un proceso el 0 (V) se convierta en menos voltaje para eso se necesita fuentes negativas para esto se utiliza el comparador LM741 de la siguiente manera.

- Si el voltaje entrada 1 es mayor que el voltaje de entrada 2 la salida va a ser +Vcc
- Si el voltaje de entrada 2 es mayor al voltaje de entra 1 la salida va a ser -Vcc

Para alimentar el circuito operacional se necesita 2 fuentes que en este caso se utilizó 2 pilas que unidas dan + 3Voltios y -3 Voltios.

La segunda entrada del operacional estará conectado a un voltaje de referencia de 3 (V) el cual provee la unión de 4 Pilas AAA, esta entrada también estará conectada a un potenciómetro de 10 $\boldsymbol{K}\Omega$, ese voltaje permitirá comparar con los voltajes que saldrán del Arduino.

1.13. Proteus ISIS_ARES

ISIS es la herramienta principal de Proteus. Combina un entorno de diseño de una potencia excepcional con una enorme capacidad de controlar la apariencia final de los dibujos. Es ideal para una rápida realización de complejos diseño de esquemas electrónicos destinados tanto para tareas de simulación y pruebas como para la construcción de equipos electrónicos. Permitirá realizar el esquema electrónico del circuito que se desee diseñar posteriormente a través del entorno ARES. Posee una amplia colección de librerías de componentes y aparte permite crear nuevos componentes y su modelización para la simulación.

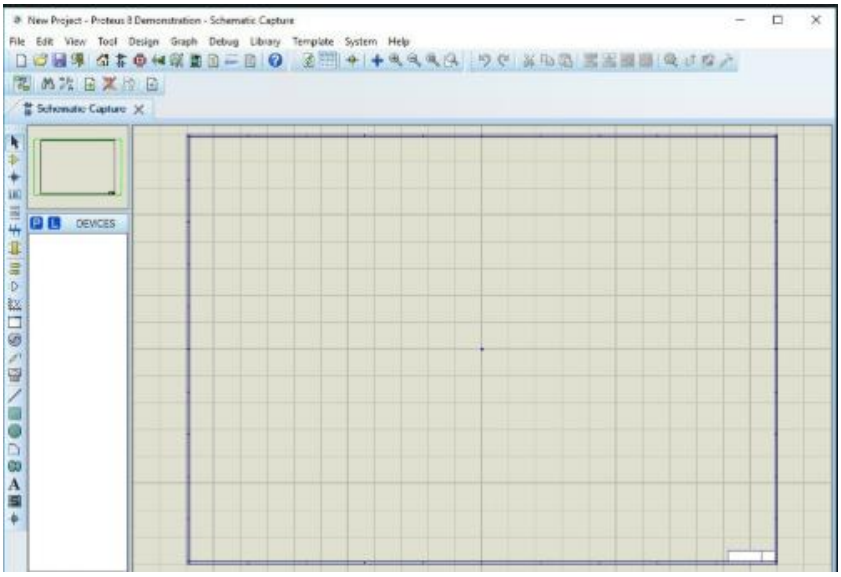


Figura 1.20 Ambiente Proteus

Fuente:(**Elaborado por el Autor, 2018**)

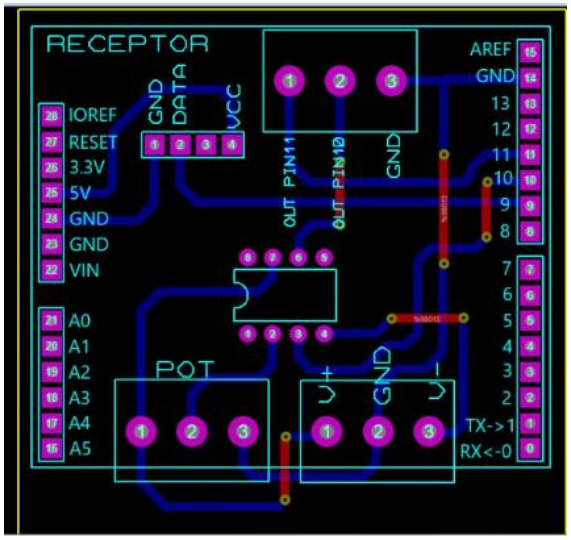


Figura 1.21 Ambiente ISIS

Fuente:(**Elaborado por el Autor, 2018**

CAPÍTULO II

2.1. MARCO METODOLOGICO

2.2. Metodología

El presente trabajo de titulación fue realizado con un enfoque experimental, debido a que el proyecto está orientado al fortalecimiento del aprendizaje de códigos de línea impartidos dentro de la asignatura de Comunicaciones II, los recursos ocupados en la realización de este proyecto son un computador para la interfaz gráfica realizada en MATLAB, Arduino UNO R3 programados bajo el IDE de Arduino, transmisores de radiofrecuencia de 477 MHz para una intercomunicación entre los equipos mencionados anteriormente y un osciloscopio propio del laboratorio utilizado para visualizar los resultados del módulo. La investigación se dividirá en tres temáticas trascendentales para culminar el proyecto en mención, dichas etapas son: técnicas de recolección de información, variables dependientes e independientes, y resultados obtenidos.

2.3. TÉCNICAS DE RECOLECCIÓN DE INFORMACIÓN.

Las técnicas de recolección de información planteadas son las que se enumeran a continuación:

2.3.1. Técnica inicial de fuentes virtuales

Esta técnica será empleada en las etapas iniciales del proyecto, tanto en la planificación y en el diseño del sistema, estas fuentes virtuales serán provenientes de la web en su mayoría. Una vez recopilada exactamente 80 fuentes de información, se aplicará un filtro de selección para obtener los documentos de mayor relevancia para el proyecto, mencionado filtro se realizará mediante la aplicación de tres parámetros que son descritos a continuación:

- **Validez científica del documento**. - Es necesario que toda información o documento recopilado cuente con un sustento científico comprobado de forma previa, para lo cual, toda información o documento receptado en la presente investigación deberá contar con el sustento de una entidad reconocida a nivel científico, académico o en el ámbito privado a nivel industrial comercial, tales como publicaciones de universidades a nivel mundial, entidades dedicadas a la investigación, o empresas privadas que se dediquen al diseño y ensamblaje de equipos experimentales de laboratorio.
- **Pertinencia del documento.** - Toda la información recolectada, deberá tener una relación directa con los temas vinculados al proyecto de investigación desarrollado tales como, códigos de línea, programación en Arduino y, módulos de radiofrecuencia, etc.
- **Soporte Bibliográfico**. - Es necesario que todo documento consultado cuente con un respaldo bibliográfico sin importar si es un documento físico o virtual conseguido en la red.
- Aplicando los diferentes filtros se utilizó un total de 28 documentos en los cuales se respalda la investigación del presente proyecto.

2.4. VARIABLES

2.4.1. Variables independientes y dependientes

- **Variables Independientes**: es una variable que representa una cantidad que se modifica en una investigación.
- **Variable Dependiente:** es la variable que se investiga y se mide.

Las variables dependientes se ven regidas a las características funcionales de cada uno de los elementos que componen el módulo de laboratorio, y las particularidades de los mismos que pueden llegar a afectar en el desempeño del módulo, como recursos de memoria, procesamiento que requieren el tratamiento o manipulación de dichas especificaciones técnicas para un correcto funcionamiento

Por otro lado, las variables independientes estar directamente relacionadas con las condiciones de los materiales eléctricos, y las condiciones ambientales para la propagación de las ondas que transmitirán los códigos de línea planteados como objetivo del presente proyecto.

2.5. Metodología del proceso investigativo

Para definir el método investigativo a utilizar, se utiliza las técnicas y métodos descritos en la Tabla 2.1, donde se especifica la metodología utilizada para cada etapa del proyecto de titulación, y lograr el objetivo de la implementación del módulo de prácticas de laboratorio para el aprendizaje de códigos de línea.

Tabla 2.1. Técnicas y métodos de investigación

Etapa de investigación	Métodos			Técnicas
	Empíricos	Teóricos	Matemáticos	
Fundamentación Teórica		• Inductivo • Deductivo • Sistemático		• Revisión bibliográfica
Marco Metodológico	• Revisión documental • Recolección de información		• Pruebas de hipótesis	• Revisión de Criterios de expertos
Propuesta		• Inductivo • Deductivo • Sistemático		
Implementación	• Experimentos • Otros métodos empíricos		• Pruebas de hipótesis	

Fuente: **(Elaborado por el Autor, 2018)**

CAPÍTULO III

3. PROPUESTA

3.1. Descripción General

Como se considera que la propuesta es el sustento para empezar con la implementación del proyecto, se crea la necesidad de contar con un diagrama general que permite identificar cada uno de los componentes del módulo a desarrollar. Como se puede observar en la Figura 3.1, el módulo a desarrollar contará esencialmente con un transmisor y un receptor que se comunicarán de manera inalámbrica a través de los módulos de RF a 477 MHz.

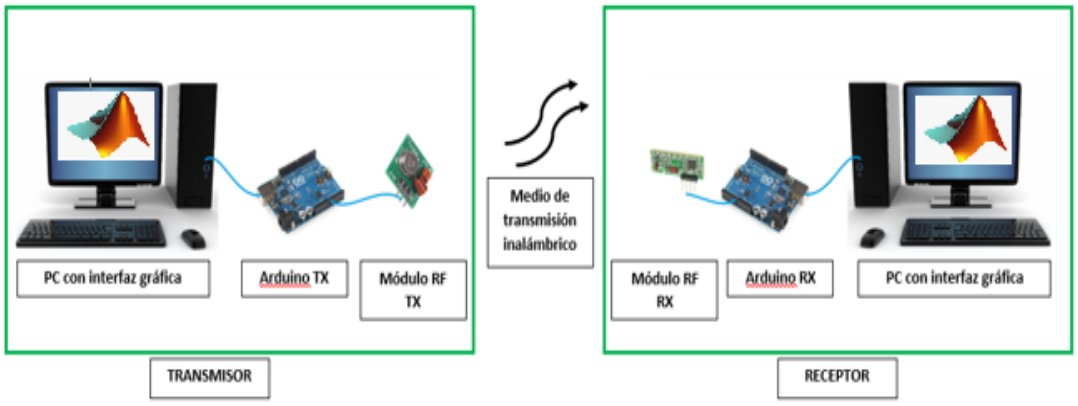


Figura 3.1 Elemento básicos del proyecto

Fuente: (Elaborado por el Autor, 2018)

El funcionamiento en general del presente proyecto de titulación, tiene dos instancias, el transmisor y el receptor, en el transmisor mediante un programa desarrollado en Matlab y el

cual se detalla en el siguiente Capítulo, se envía una cadena serial de bits al Arduino UNO mediante comunicación serial.

El Arduino UNO, lee la cadena recibida y a su vez envía al módulo transmisor de RF.

En el receptor los datos enviados por este son recibidos mediante un módulo de recepción de las mismas características del transmisor y leídos por el módulo Arduino UNO esta lectura es idéntica a la descrita en el proceso de transmisión una vez que los datos han sido leídos por el Arduino, son enviados al computador mediante trasmisión serial. En el PC los datos recibidos son identificados y mostrados en pantalla mediante una aplicación desarrollada en Matlab, la cual se detalla en el CAPÍTULO 4.

3.2. Descripción de módulos

En el computador que funciona en el transmisor, se desarrolló una aplicación en MATLAB, cuya función principal es generar la cadena de bits para este propósito, que será enviada a la placa de desarrollo Arduino UNO. En la Figura 3.2, se observa el diagrama de flujo que cumplirá el computador en el transmisor en este proyecto.

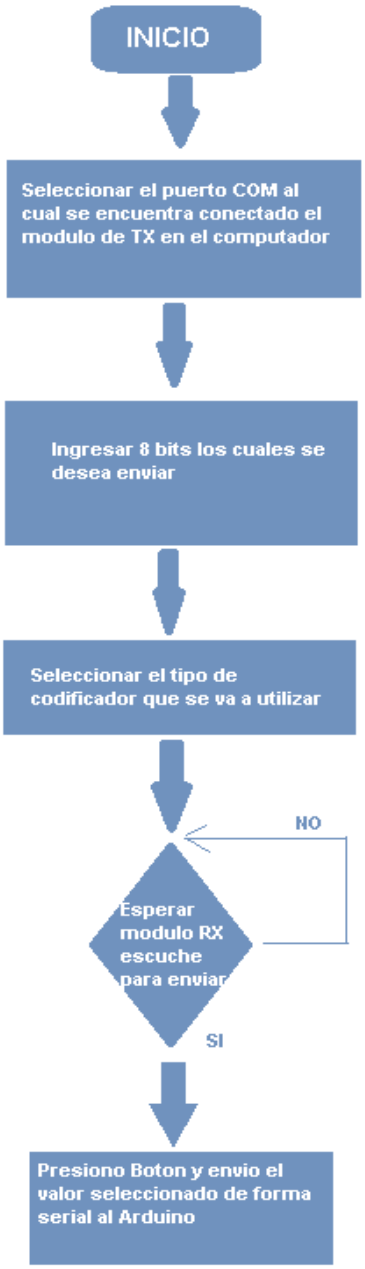


Figura 3.2 Algoritmo Interfaz gráfica Tx

Fuente: **(Elaborado por el Autor, 2018)**

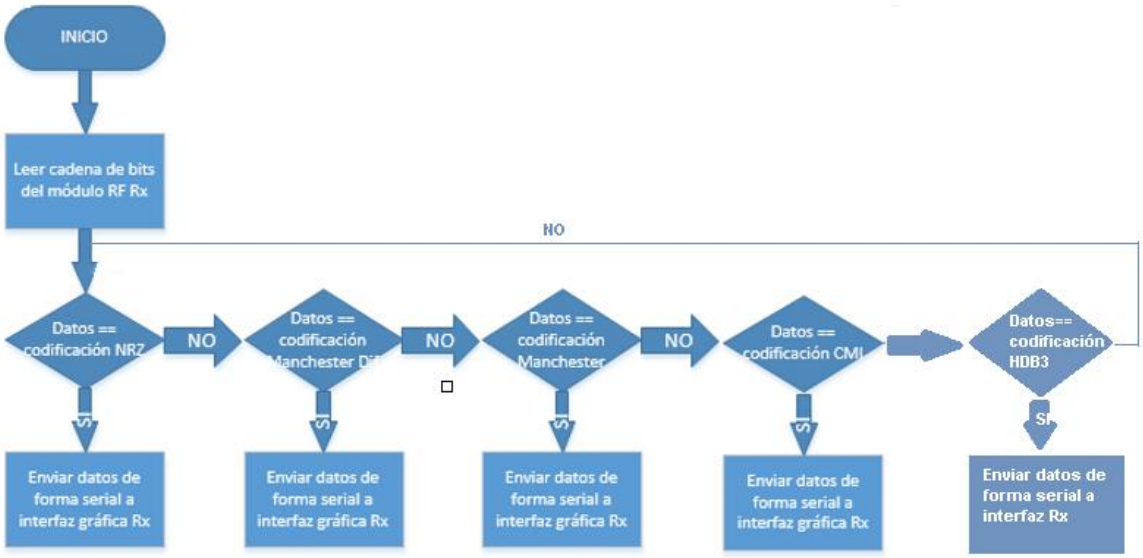


Figura 3.3 Algoritmo Interfáz gráfica Rx

Fuente: (Elaborado por el Autor, 2018)

Una vez que la cadena de bits se encuentra en el Arduino UNO transmisor o receptor, este cumplirá la función especificada en las Figura 3.2 y 3.3, donde se visualiza que la tarjeta Arduino es la encargada de realizar la codificación de línea requerida, ya que el computador transmisor será encargado de enviar las cadenas de bits ingresados.

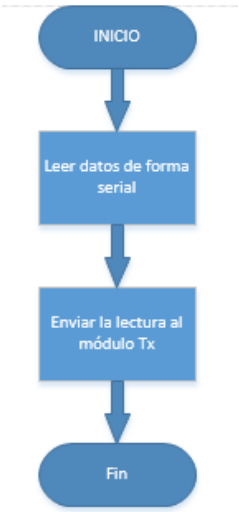


Figura 3.4 Algoritmo Arduino Tx

Fuente: (Elaborado por el Autor, 2018)

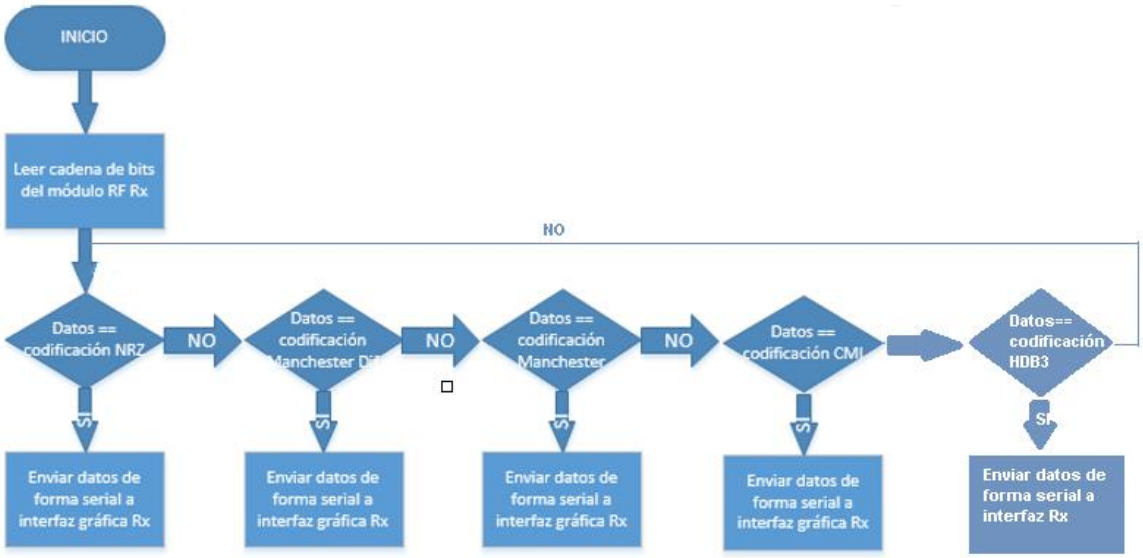


Figura 3.5 Algoritmo Arduino Rx

Fuente: (Elaborado por el Autor, 2018)

3.3. Aspectos Técnicos del Trasmisor y Receptor

Para la implementación del presente proyecto de titulación es indispensable conocer las características técnicas de cada uno de los equipos y componentes que son necesarios para cumplir con los objetivos planteados, es por ello, que lo descrito en el apartado anterior permite identificar al trasmisor y receptor con los siguientes componentes: computador, Arduino UNO y módulo transmisor de RF, mismos que técnicamente se especifican a continuación:

Tabla 3. 1 Características técnicas del PC.

Característica	Detalle
Procesador	Intel Core i5-2350M
Frecuencia del procesador	2,3 GHz
Modo operativo procesador	64 bits
Pantalla	14 pulgd.
Memoria Interna	4 GB
Tipo de memoria interna	DDR3-SDRAM

Capacidad almacenamiento	500 GB
Sistema Operativo	Windows 7 Home Basic

Fuente**: (Elaborado por el Autor,2019)**

Tabla 4. 2. Características técnicas del Arduino UNO

Característica	Detalle
Microcontrolador	ATmega328P
Voltaje de operación	5v
Voltaje de entrada	7-12v
Entradas digitales	14
Entradas PWM	6
Entradas análogas	6
Memoria flash	32 KB
SRAM	2 KB
EEPROM	1 KB

Fuente: **(Arduino, 2018)**

Tabla 5. 3. Características técnicas del módulo transmisor RF

Características	Detalle
Modelo	MX-FS-03V
Alcance	20-200 metros
Alimentación	3.5 – 12v
Dimensiones	19 * 19 mm
Tasa de transmisión	4 KB/S
Potencia de transmisión	10 mW
Frecuencia de transmisión	433 Mhz
Antena recomendad	25 cm

Fuente: **(patagoniatec, 2015)**

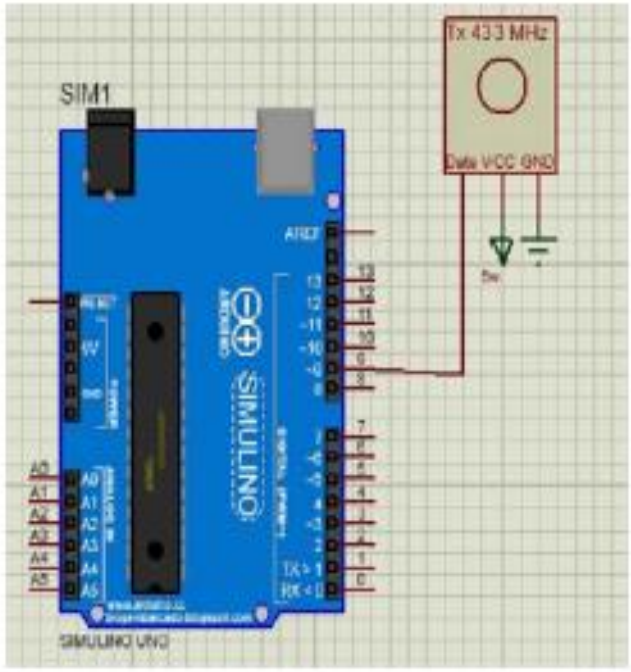
Tabla 6. 4 Características técnicas del módulo receptor RF

Características	Detalle
Modelo	MX-05V
Voltaje de operación	DC 5v
Consumo	4 mA
Frecuencia de recepción	433 Mhz
Tamaño	30 * 14 * 7
Modulación	ASK
Antena	No incluida cable Cobre

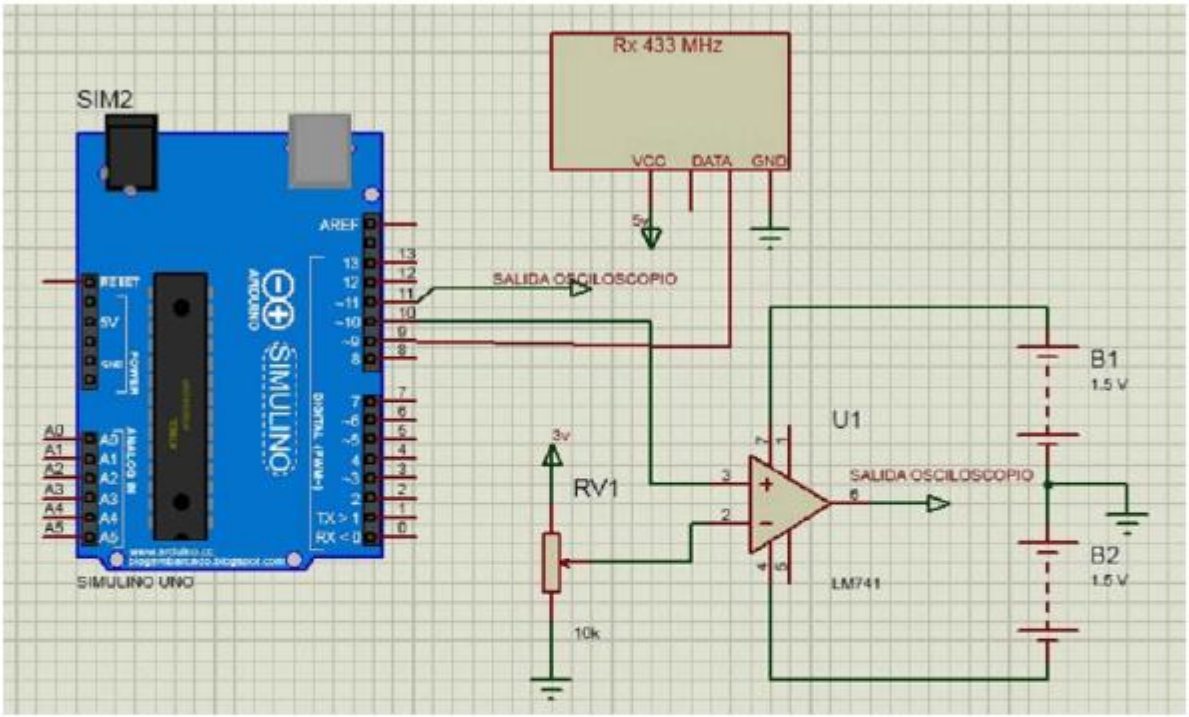
Fuente: **(patagoniatec, 2015)**

Cada uno de los componentes descritos en las Tablas 3 – 6 permiten que la implementación estipule características que ayudan al rendimiento del presente proyecto y de igual manera permiten que el proyecto sea eficiente y eficaz al momento de realizar una transmisión y recepción de datos, cabe recalcar que las características especificadas entregan datos que ayudan a conocer valores nominales de voltajes y corrientes, tales que permiten de una u otra manera conocer valores eléctricos para proteger los componentes electrónicos.

A continuación, se detalla los circuitos que permitieron el funcionamiento de los módulos planteados, tanto de transmisión como de recepción, en los mismos se podrá observar cada uno de los componentes que lo conforman y formaron parte de la placa llamada baquelita



UNIVERSIDAD TECNOLOGICA ISRAEL		
TEMA: Circuito Modulo Transmision		
NOMBRE: Wendy Alvarado	FECHA:18 FEBRERO 2019	
TUTOR: Alfonso Zozaya PHd	LAMINA N°:	1



UNIVERSIDAD ISRAEL	UNIVERSIDAD TECNOLOGICA ISRAEL	
	TEMA: Circuito Modulo Recepcion	
	NOMBRE: Wendy Alvarado	FECHA:18 FEBRERO 2019
	TUTOR: Alfonso Zozaya PHd	LAMINA N°: 2

En la Lamina N°1 se puede observar los elementos necesarios para cumplir los requisitos para una correcta transmisión de señales, los cálculos de elementos pasivos como resistencias se determinan mediante la siguiente expresión:

$$\boldsymbol{I = \frac{V}{R}}$$

Ecuación 3.1 ley de Ohm

Fuente:(Manuel, Ángel, 2010)

Donde:

$\boldsymbol{V}$ = Voltaje máximo que soporta una entrada arduino.

$\boldsymbol{I}$ = Corriente máxima para arduino.

Para seleccionar la resistencia adecuada se utiliza y se reemplaza el valor máximo de voltajes de operación que el datasheet de arduino indica, para ello se tiene en cuenta el valor máximo de una salida que entrega la tarjeta, el resultado que se obtiene es el siguiente:

$$\boldsymbol{I = \frac{5v}{10\ k\Omega} = \ 0.5\ mA}$$

Las resistencias que se deben utilizar en el diseño de la Lamina 2 (Circuito Modulo de Recepción) son de 10 $\boldsymbol{k\Omega}$, ya que con esto se garantiza que los pines de salida alimenten el circuito.

3.4. Software

Los programas con los que se realizó la implementación del presente proyecto están ligados directamente con la utilización del hardware, es decir, para la comunicación con la PC se utiliza el software Matlab ya que este permite tener una interfaz gráfica amigable con el usuario y transmitir datos de forma serial, los recursos necesarios que necesita este programa para hacer uso del PC se detalla en la Tabla 3.1 En cuanto al software que permite interpretar el proceso de transmisión y codificación de los códigos de línea que serán

embebidos en la tarjeta arduino, mediante el programa propio del fabricador que se encuentra disponible en la web para ser descargado, además este no necesita de ninguna versión pro, simplemente con una de las versiones que permita controlar la tarjeta Arduino uno es suficiente.

3.5. Análisis de costos y tiempo

El costo del presente proyecto está basado directamente con la adquisición de materiales y componentes electrónicos, mismos que se detallan en la Tabla 3.5.

Tabla 7. 5 Costos de implementación.

Cantidad	Descripción	V. Unitario	V. Total
2	Arduino UNO R3	15.00	30.00
1	Módulo RF	7.50	7.50
1	Cables para conexiones	2.50	2.50
300	Horas de trabajo	6.25	1875
		Sub total	2615
		Total	2615

Fuente: **(Elaborado por el Autor, 2018)**

Como se observa en la Tabla 3.5, la optimización de costes está basado en la utilización de la tarjeta arduino, misma que es capaz de interpretar mediante su código fuente la codificación de códigos de línea establecida mediante procesos incluidos en diagramas de flujo, mismos que son transmitidos y recibidos por los módulos de radio frecuencia desde y hacia las PC. El tiempo que tardará en ser implementado el proyecto se describe en la figura 3.6.

Es importante mencionar que los diagramas de flujo detallados en el Capítulo 3 y que son interpretados por el dispositivo arduino mediante su programación basada en lenguaje C han sido embebidos en la tarjeta misma y en la cual mediante librerías se puede realizar una transmisión serial para que los módulos de radio frecuencia, mediante su medio de transmisión (inalámbrico), puedan transmitir y recibir, esto permite obtener un sistema con una comunicación unidireccional.

El detalle de actividades se puede observar en la siguiente Figura 3.6 :

DIAGRAMA DE GANTT

	Modo de	Nombre de tarea	Duración	Comienzo	Fin	Prede
1		Aprobación del plan de tesis	1 día	jue 03/05/18	jue 03/05/18	
2		Planteo de fundamentación teórica	5 días	vie 04/05/18	jue 10/05/18	1
3		Elaboracion de la metodología	2 días	vie 11/05/18	lun 14/05/18	2
4		**Desarrollo de la propuesta**	**9 días**	**lun 14/05/18**	**jue 24/05/18**	**3**
5		Desarrollo del esquema general	1 día	mar 15/05/18	mar 15/05/18	3
6		Identificación de módulos de TX y RX	3 días	mié 16/05/18	vie 18/05/18	5
7		Identificación de módulos de hardware y software	3 días	lun 21/05/18	mié 23/05/18	6
8		Análisis de costos	1 día	jue 24/05/18	jue 24/05/18	7
9		**Implementación del proyecto**	**29 días**	**jue 24/05/18**	**mar 03/07/18**	
10		Construcción de hardware	10 días	vie 25/05/18	jue 07/06/18	8
11		Programación de software	10 días	vie 08/06/18	jue 21/06/18	10
12		Pruebas de funcionamiento	4 días	vie 22/06/18	mié 27/06/18	11
13		Análisis de resultados	4 días	jue 28/06/18	mar 03/07/18	12
14		Documentación del proyecto	26 días	mié 04/07/18	mié 08/08/18	13

Figura 3.6 Detalle de actividades a realizar en el presente proyecto

Fuente: (Elaborado por el Autor, 2018)

A continuación, se detalla como mediante el diagrama de Gant un análisis del tiempo aproximado que se podría llevar en el desarrollo de la presente tesis se contempla tanto la parte de hardware como la de software.

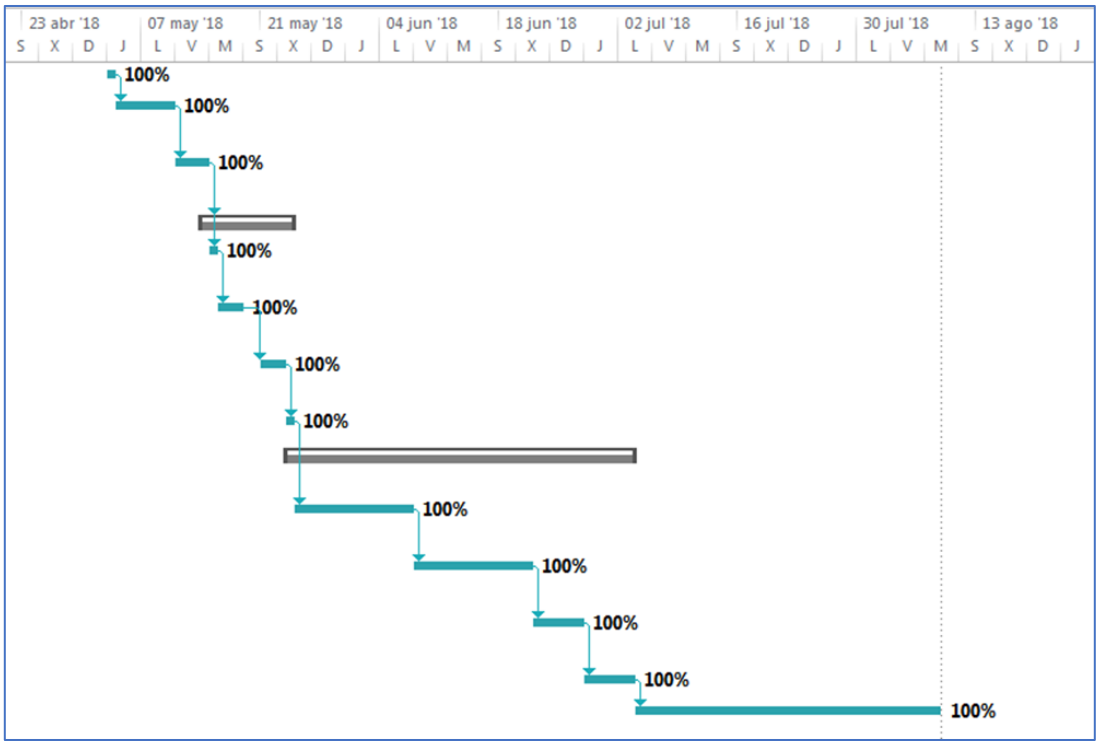


Figura 3.7 Detalle en porcentaje de actividades a realizar en el presente proyecto

3.6. Justificación de hardware y software

Los componentes electrónicos y dispositivos que se utilizan en el presente proyecto han sido seleccionados con base en la aplicación que se desea conseguir, por lo tanto, si se tiene en cuenta que el objetivo principal está basado en realizar transmisión y recepción de datos a través de un canal inalámbrico, las necesidades se especifican con conocimientos de principios básicos de un sistema de comunicación.

Entonces, se parte de la necesidad de tener un origen de datos para ello el uso de un PC con características técnicas como las mencionadas en la Tabla 3.1 en la cual se desarrolla una aplicación en el software Matlab que como se observó en el Capítulo I es un software

que se adapta a las necesidades de los procesos de desarrollo que se van a necesitar en el presente proyecto debido a que es un lenguaje de alto rendimiento por para proceder a la transmisión de datos; una vez que se conoce el origen de los datos para transmitir es necesario seguir el proceso de transmisión para ello es indispensable codificar los datos (con códigos de línea) que envía el computador y ahí es necesario la utilización de la tarjeta Arduino ya que al tener internamente un microcontrolador es posible realizar los procesos de codificación de datos mediante programación en lenguaje C.

Al terminar el proceso mencionado en el anterior párrafo queda solo enviar los datos a través de un medio inalámbrico (de corta distancia), para ello el uso de los módulos RF ya que a través de ondas electromagnéticas envía los datos provenientes del arduino hacia el receptor que posee los mismos componentes electrónicos y dispositivos que el transmisor, cabe recalcar que el receptor posee las mismas herramientas ya que en este se realiza el proceso inverso al de la transmisión para evaluar y analizar la transmisión que se realiza.

3.7. Ventajas de los Módulos

- Sirven como implementos del laboratorio de Comunicaciones II de la Universidad Israel donde se puede observar de mejor manera el tema de codificación de líneas.
- Se pueden realizar varias practicas donde el estudiante puede seguir los pasos que especifican en las guías de laboratorio, de esta manera se puede dar cuenta el proceso de transmisión de datos.
- El software Matlab correspondiente a la interfaz gráfica de los módulos mencionados es amigable permitiendo al usuario poder desarrollar las practicas sin dificultad.
- Los elementos que conforman el módulo son estudiados y analizados a lo largo de la carrera de Electrónica Digital y Telecomunicaciones, siendo así fácil de identificar y analizar de parte del usuario.
- Los módulos tanto de transmisión y recepción son ligeros y fáciles de transportar de un lugar a otro.
- Las cajas o parte externa que recubre a los módulos son transparentes que permiten al mismo tiempo ofrecer fácil visibilidad y protección a los mismos.

CAPÍTULO 4

4. IMPLEMENTACIÒN

En el presente capitulo se detallará todo lo referente a la implementación en cuanto a Hardware y Software y desarrollo practico del proyecto, proceso de construcción, también se realiza una descripción de pruebas realizadas del producto final.

4.1. Desarrollo

A continuación, se detalla los pasos a seguir para la creación de los módulos en cuanto a hardware y software.

Figura 4.1 Construcción Modulo Transmisor y Receptor

Fuente: **(Elaborado por el Autor, 2018)**

4.2. Proceso de construcción del módulo Trasmisor

Para lograr el correcto funcionamiento del módulo de prácticas de laboratorio se observa en la Figura 4.1 los pasos a seguir en la construcción del mismo de igual manera en un transmisor y un receptor.

En la Figura 4.2, se puede observar el diagrama general de conexión del trasmisor, que esencialmente inicia con un computador, donde a través del software desarrollado en Matlab, se enviará la cadena de bits al Arduino UNO mediante una comunicación serial. El Arduino UNO leerá la cadena de bits enviada desde el computador, y realizará una codificación de línea (depende del código de línea seleccionado).

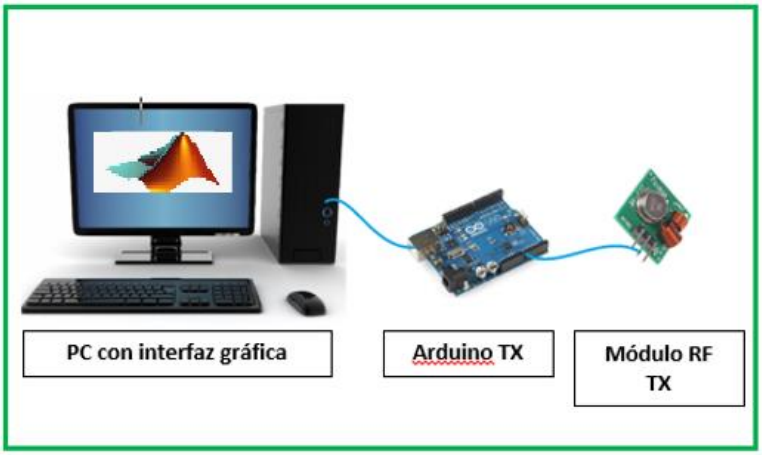


Figura 4.2 Módulo Transmisor

Fuente: **(Elaborado por el Autor, 2018)**

Una vez que la cadena de bits se encuentre en la tarjeta de desarrollo Arduino UNO, esta cadena será enviada de al módulo transmisor de radiofrecuencia FS1000A, que realizará una comunicación con su par receptor, mediante un solo canal de comunicación unidireccional. El diseño de la placa que permite realizar la transmisión de datos que se encuentra basada en un diagrama lógico que posteriormente será implementada en una placa conocida como baquelita (ver Figura 4.3).

Figura 4.3 Placa diseñada para Tx

Fuente: **(Elaborado por el Autor, 2018)**

4.3. Construcción placas de circuitos

Para la construcción de placas de los circuitos se utilizó el software ISIS conjuntamente simulado en el software ARES como se explicó en el CAPITULO I ISIS es una herramienta principal de PROTEUS debido a que combina un entorno de diseño con una capacidad de controlar la apariencia final de los dibujos siendo ideal para la simulación y verificación del funcionamiento del circuito

Una vez obtenido el diseño realizado en Proteus tanto en la parte de Isis y Ares se procede a imprimir el circuito para poder impregnar en la baquelita (Ver Figura 4.4) realizando la técnica del planchado.

4.4. Materiales utilizados en un circuito impreso

- Impresora Laser
- Material conductor Pistas
- Placa Fibra de Vidrio
- Lija de Agua
- Plancha
- Ácido Férrico
- Tiñer

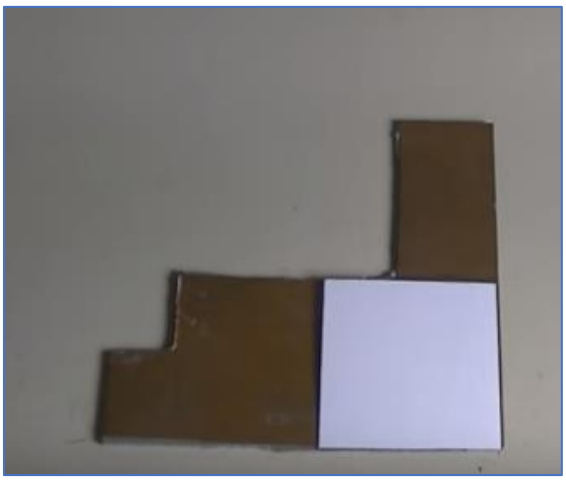
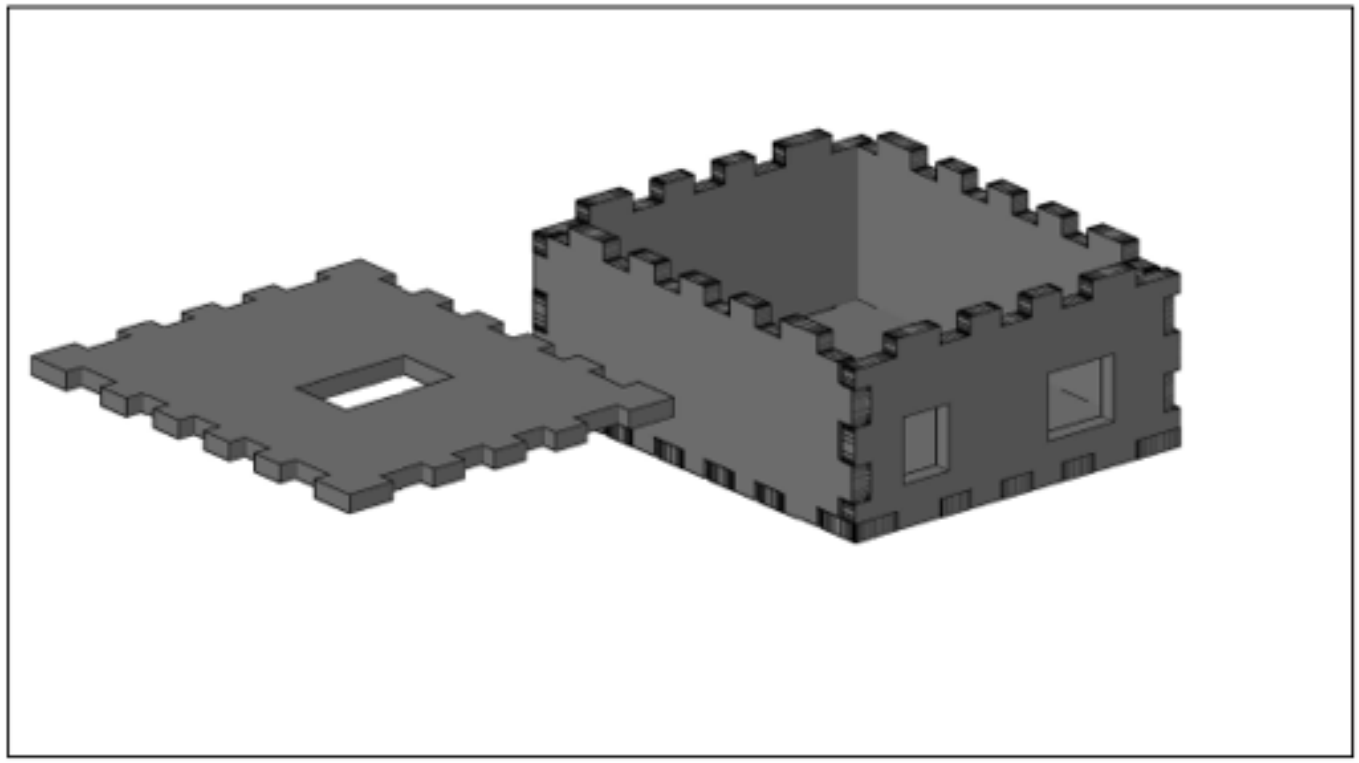


Figura 4.4 Elaboración de Placa Baquelita

Fuente: **(Elaborado por el Autor, 2018)**

Una vez obtenido el circuito impreso en la baquelita se procede a sumergir el mismo con ácido férrico para que queden impregnado el diseño del circuito y poder realizar las perforaciones necesarias y poder soldar sus componentes

Finalmente, a continuación, la lámina N°3 se puede observar la caja realizada para protección de los componentes del módulo Transmisor.



CAJA EMISOR

UNIVERSIDAD ISRAEL	**UNIVERSIDAD TECNOLÓGICA ISRAEL**		
	TEMA: Módulo de comunicación de códigos de línea para el laboratorio de comunicaciones II de la Universidad Israel.		
	NOMBRE: WENDY ALVARADO	**FECHA:** 16 FEBRERO DEL 2019	
	TUTOR: Alfonso Zozaya PHd	**LAMINA Nº:**	3

4.5. Proceso de construcción del módulo Receptor

Como se puede observar en la Figura 4.5, el proceso de recepción lo iniciará el módulo de radiofrecuencia XY-MK-5V que recibe la cadena de bits de su par que se encuentra en el transmisor, este elemento enviará la cadena recibida al Arduino UNO del receptor, quien será el encargado de interpretar la cadena recibida y la enviará mediante comunicación serial al computador, donde mediante el software desarrollado en Matlab, realizará la graficación de la cadena recibida, para que el estudiante logre un mejor aprendizaje de los códigos de línea.

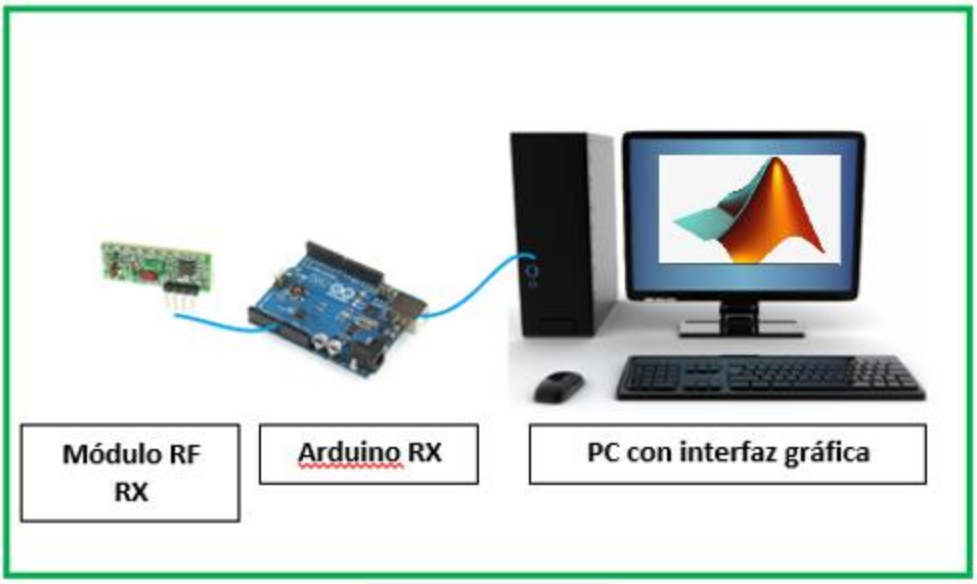


Figura 4.5 Módulo receptor

Fuente: **(Elaborado por el Autor, 2018)**

De igual forma, el módulo fue diseñado para en su componente de recepción, la computadora fuese reemplazada por un osciloscopio como se muestra en la Figura 4.6, esto se lo realiza con fines de comprobación, para verificar que las cadenas de bits que se recibe sean las esperadas. El diseño de la placa que permite realizar la recepción de datos se encuentra basada en un diagrama lógico (Lámina 2) que posterior será implementada en una placa conocida baquelita (ver Figura 4.4).

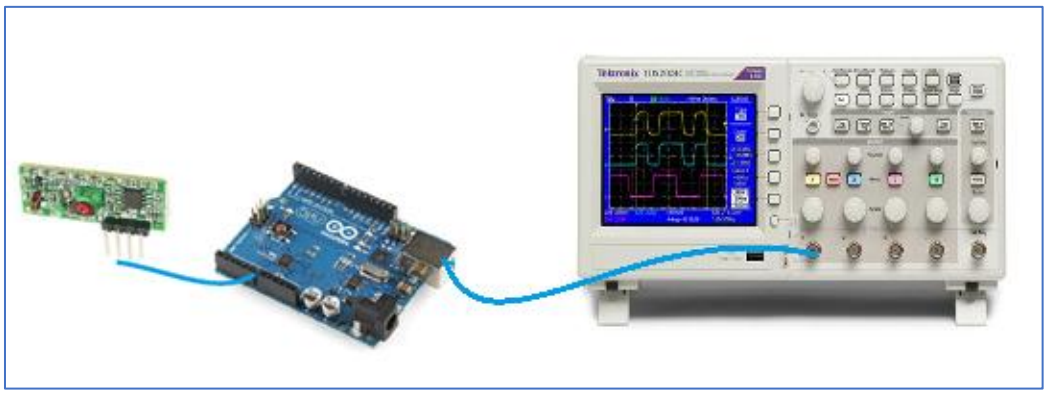


Figura 4.6 Módulo receptor con osciloscopio

Fuente: **(Elaborado por el Autor, 2018)**

La realización de la baquelita es el mismo proceso explicado en el emisor se detalla brevemente a continuación;

- Diseño de circuito en programa Proteus (ver figura 4.7)
- Impresión del circuito con impresora Laser
- Impregnación de circuito en placa baquelita realizando técnica de planchado
- Quemar la placa con ácido férrico
- Perforación y soldado de piezas.

A continuación, la lámina N°4n se puede observar la caja realizada para protección de los componentes del módulo Receptor

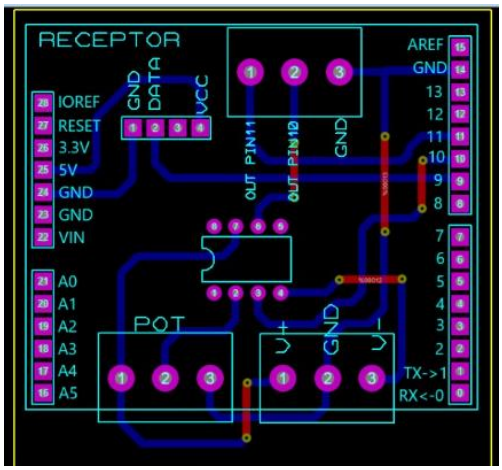
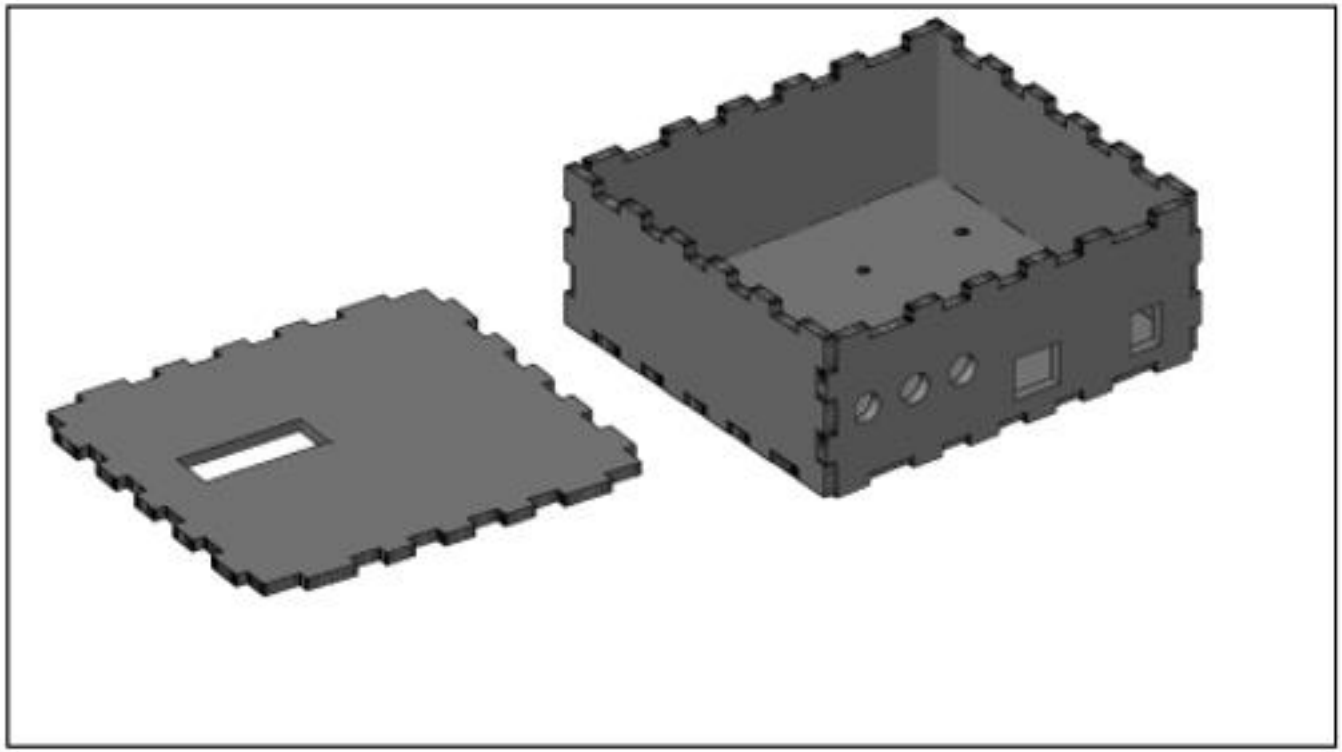


Figura 4.7 Circuito Receptor en programa Proteus Ares

Fuente:(Elaborado por el Autor, 2018)



CAJA TRANSMISOR

UNIVERSIDAD ISRAEL	**UNIVERSIDAD TECNOLÓGICA ISRAEL**		
	TEMA: Módulo de comunicación de códigos de línea para el laboratorio de comunicaciones II de la Universidad Israel.		
	NOMBRE: WENDY ALVARADO	**FECHA:** 16 FEBRERO DEL 2019	
	TUTOR: Alfonso Zozaya PHd	**LAMINA N°:**	4

Finalmente se puede observar en la siguiente imagen el resultado final del módulo de recepción Figura 4.8

Figura 4.8 Placa diseñada para Rx

Fuente: (Elaborado por el Autor, 2018)

4.6. Implementación

Una vez definido los parámetros indicados en los apartados anteriores, se procede a revisar la conectividad que se realizó en la parte del transmisor y se verifica el comportamiento del mismo. En el computador que se encuentra en la parte del transmisor, se encuentra instalado el software realizado en Matlab, una vez ejecutado el programa para el envío de las cadenas de bits, se abrirá la siguiente interfaz (Figura 4.9)(Figura 4.10)

Figura 4.9 Interfáz de Transmisión

Fuente: **(Elaborado por el Autor, 2018)**

Figura 4.10 Interfáz de Recepción

Fuente: **(Elaborado por el Autor, 2018)**

Una vez enviado las cadenas de bits, esta cadena pasará al Arduino, que tiene la conectividad con el módulo RF, El módulo RF de transmisión será en encargado de enviar inalámbricamente la cadena codificada previamente por la tarjeta Arduino, la transmisión se la realiza a una frecuencia determinada de 477MHz, con el uso de un canal unidireccional, hacia su par complemento de recepción colocado en el módulo receptor del sistema.

Cuando la cadena de bits es recibida por el módulo de RF receptor, la envía a la tarjeta Arduino receptor que será enviada al computador con el software de recepción instalado en el computador receptor, los mismos componentes se encuentran conectados bajo el diagrama mostrado en la Figura 4.1.

La cadena de bits recibida, será mostrada al usuario en el interfaz del computador receptor, donde el estudiante podrá visualizar la cadena recibida con la codificación correspondiente.

4.7. Pruebas de Funcionamiento

Una vez culminado la implementación, se realizaron las pruebas de funcionamiento, donde con ayuda de un osciloscopio se pudo corroborar que las cadenas de bits transmitidas, son equivalentes a las receptadas, estos procesos de pruebas realizadas permiten validar el correcto funcionamiento del producto final entregado en el presente proyecto

A continuación, se detalla ejemplos de los diferentes códigos de línea demostrando el funcionamiento de los módulos.

- **NRZ POLAR**

Para demostrar el correcto funcionamiento del código NRZ se detalla a continuación un ejemplo el cual como se explica en el Capítulo I IMAGEN 1.4 dicho código se caracteriza por representa el “1” binario por un voltaje +V, mientras que el “0” binario se representa por un voltaje negativo-V durante un Bit Completo.

- ✓ Ejemplo de Código a enviar ingresado en Interfaz gráfica Tx:

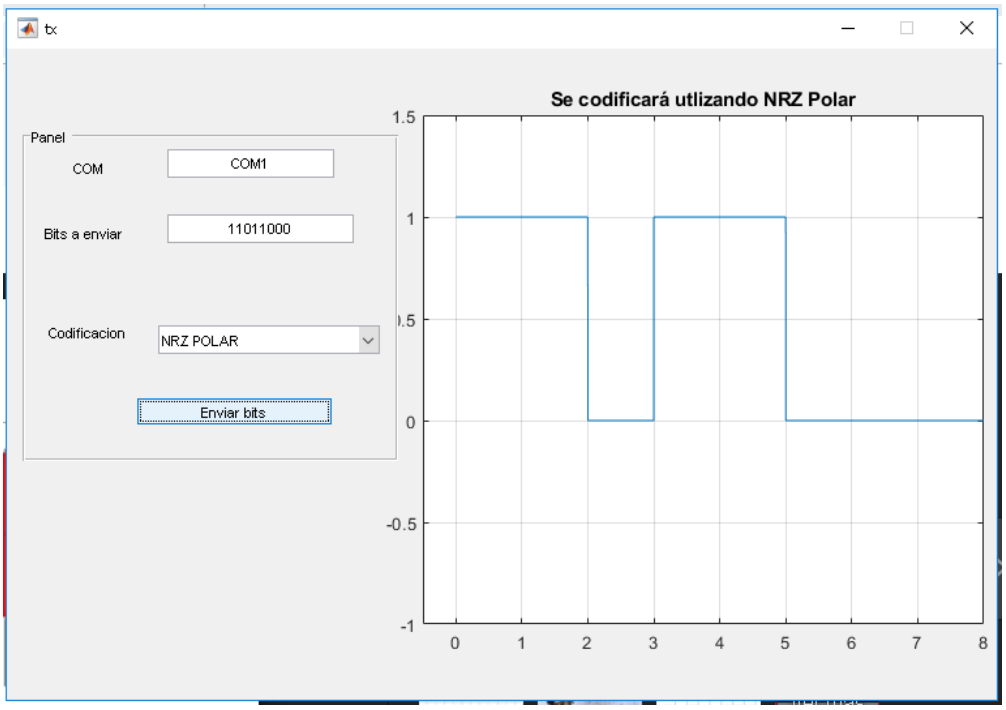


Figura 4.11 Código NRZ a enviar

Fuente: **(Elaborado por el Autor, 2018)**

Como se puede observar en la imagen 4.11 el código NRZ funciona alternando voltaje tanto positivos como negativos dependiendo del bit sea “0” o “1” la imagen 4.10 detalla el resultado esperado como fuente de información (Brito,2014)

- ✓ Código que se debería recibir según fuente bibliográfica investigada.

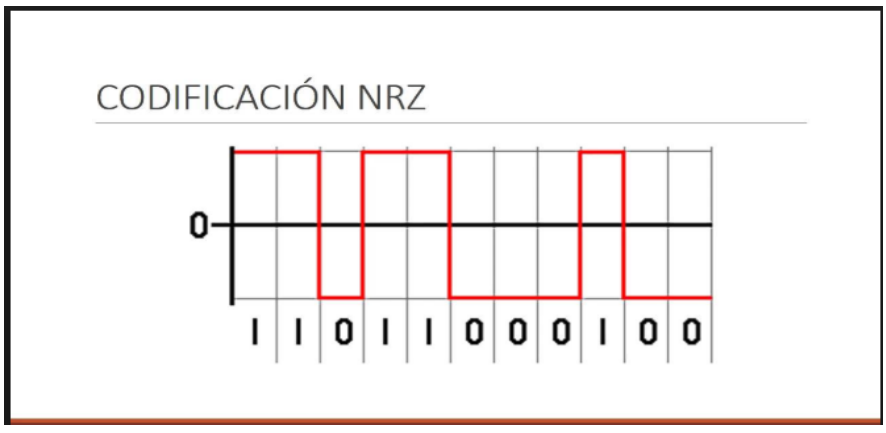


Figura 4.12 Códigos NRZ que se debe recibir

Fuente: **(Brito, 2014)**

Finalmente, para fortalecer los resultados esperados (ver Figura 4.12) se puede corroborar el correcto funcionamiento de los módulos tanto de transmisión como el receptor , debido que la imagen se reflejó en el receptor en un lapso de 5 segundos y con los resultados comparados con la Figura 4.11

- ✓ **Código que se recibe:**

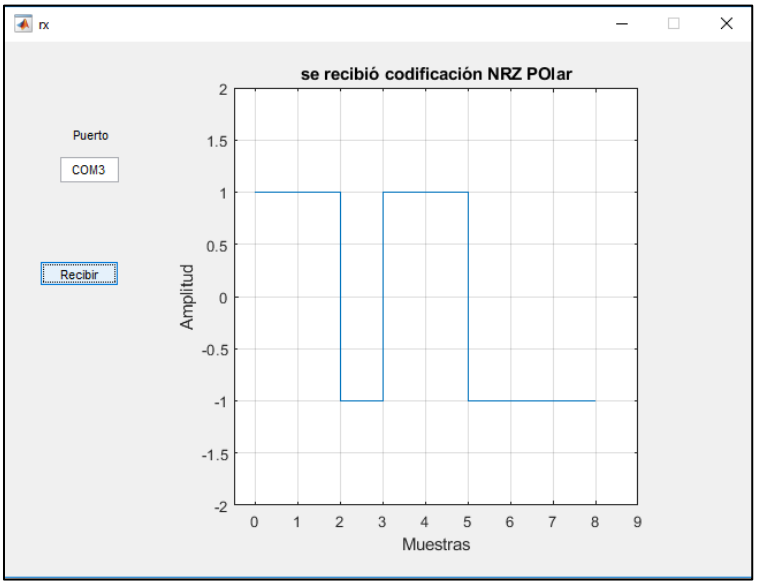


Figura 4.13 código NRZ que se observa en el receptor

Fuente: **(Elaborado por el Autor, 2018)**

Como parte de los alcances del presente proyecto esta verificar la gráfica tanto en el receptor como en el osciloscopio para poder comparar el buen funcionamiento de los módulos (ver Figura 4.13)

✓ **Código recibe en osciloscopio**

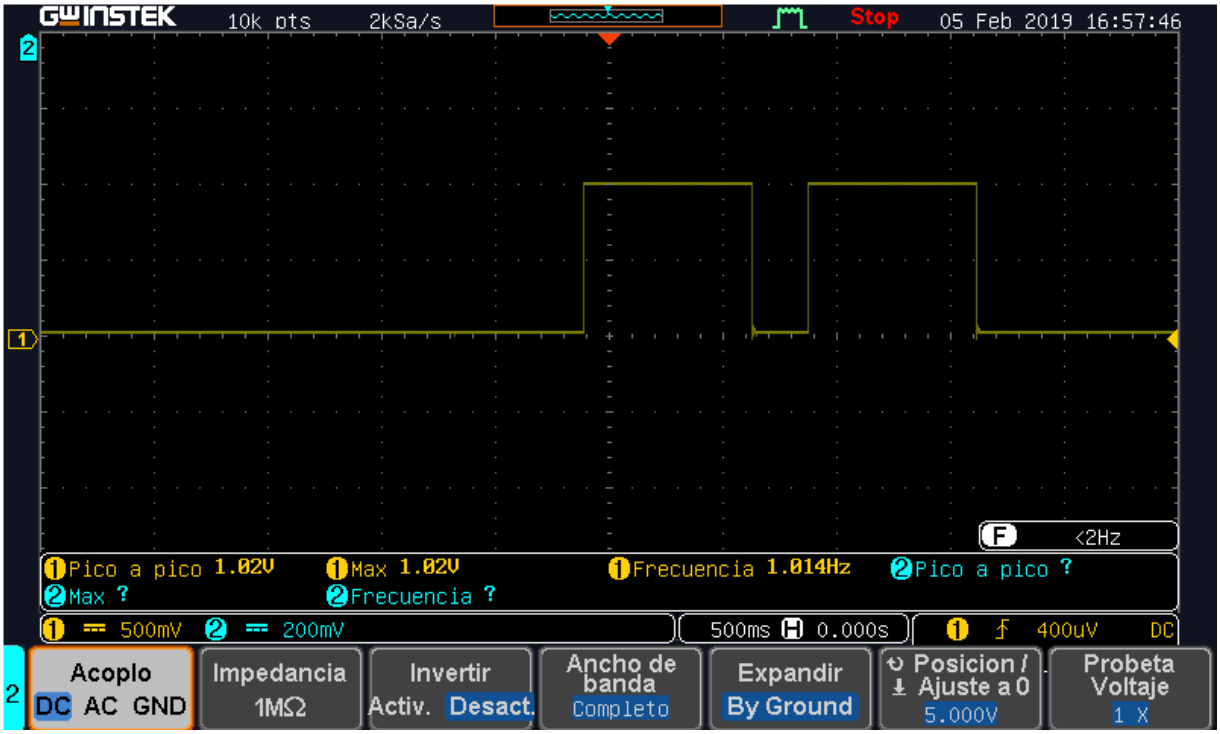


Figura 4.14 código NRZ visualización en Osciloscopio

Fuente: **(Elaborado por el Autor, 2018)**

MANCHESTER DIFERENCIAL:

Como parte de la verificación del correcto funcionamiento de los módulos predestinados al uso en el laboratorio de Comunicaciones II se realiza pruebas dentro de las cuales se encuentra el código MANCHESTER DIFERENCIAL el mismo que a continuación Figura 4.14 se visualiza los bits que se desea enviar.

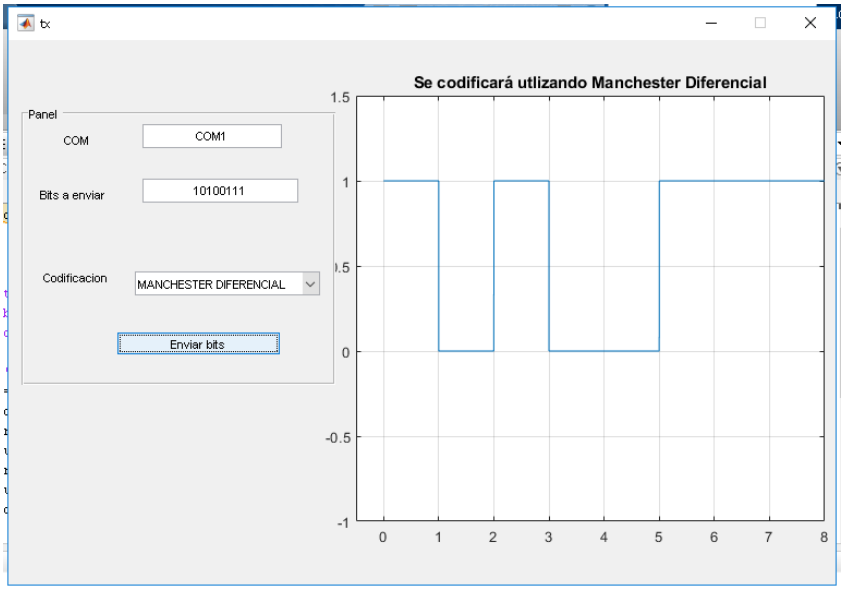


Figura 4.15 Código MANCHESTER DIFERENCIAL a enviar

Fuente: **(Elaborado por el Autor, 2018)**

Cabe recalcar que la interfaz gráfica es bastante amigable y fácil de utilizar en la Figura 4.16 se puede verificar las características de la codificación en MANCHESTER DIFERENCIAL con fuente de investigación (Brito,2014)

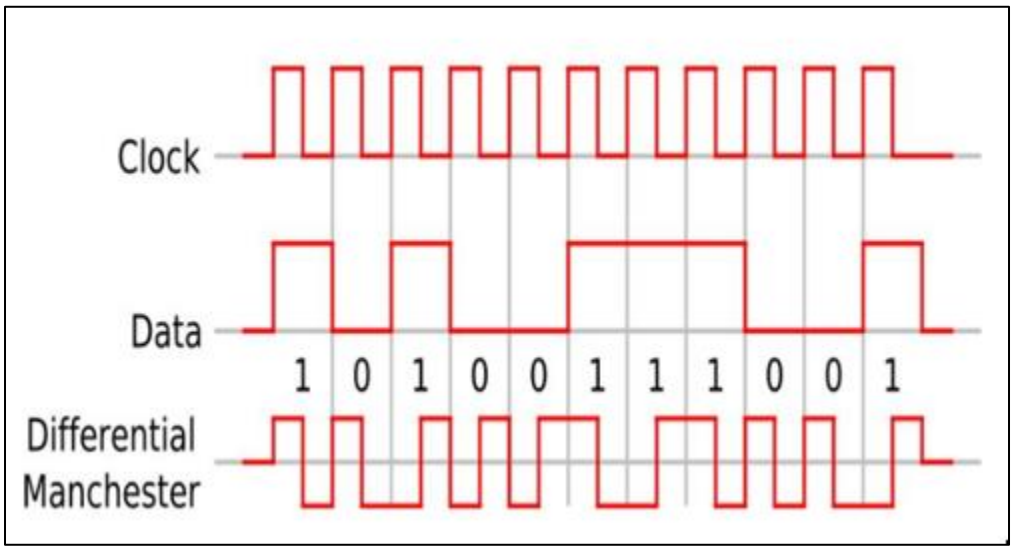


Figura 4.16 Códigos MANCHESTER DIFERENCIAL que se debería recibir

Fuente: **(Brito, 2014)**

Al presionar recibir ver figura 4.17 se puede visualizar la imagen que se recibe en el modulo Receptor.

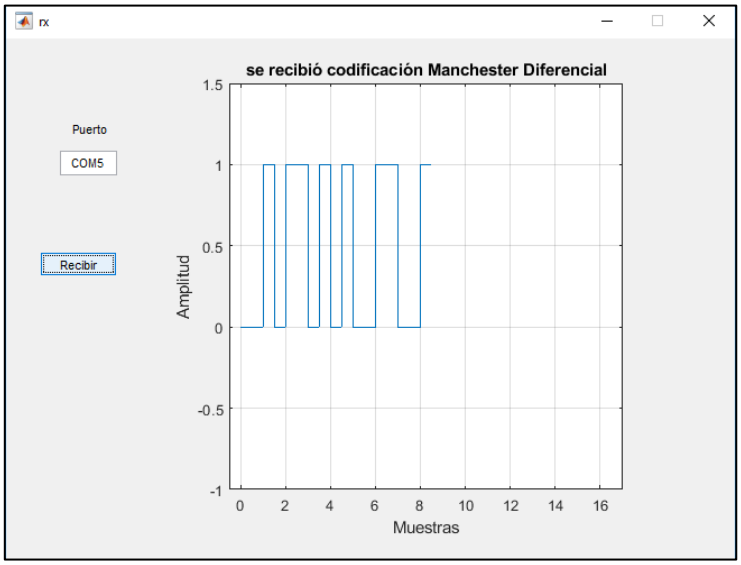


Figura 4.17 Código MANCHESTER DIFERENCIAL que se recibe

Fuente: **(Elaborado por el Autor, 2018)**

De igual manera como ejercicio de verificacion del correcto funcionamiento de los codigos lineales en este caso MANCHESTER DIFERENCIAL se puede observar en la imagen 4.18 la imagen en el osciloscopio el cual es el mismo que en la figura esperada 4.17.

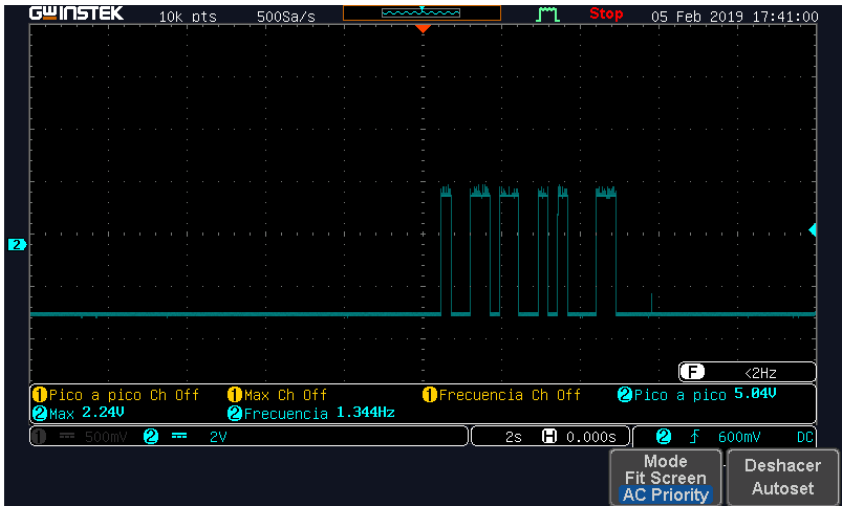


Figura 4.18 Código MANCHESTER DIFERENCIAL visualización en osciloscopio

Fuente: **(Elaborado por el Autor, 2018)**

En las gráficas se observa los resultados que se obtiene al culminar las etapas de Tx y Rx de datos, en estos se destaca el correcto funcionamiento del hardware y software, adicional se verifica la importancia de la codificación de datos para conseguir de esta manera la representación de unos y ceros que componen una señal digital y adaptarla eficientemente al medio de transmisión que en este caso es un medio inalámbrico.

De igual manera, se enfatiza que los parámetros y características que se han mencionado e implementado en el presente proyecto de titulación, han sido de vital importancia para evitar perturbaciones en nuestra señal que viaja a través del espectro radioeléctrico. Finalmente, los resultados principales que se consiguió con el proceso implementado permiten obtener una transmisión eficiente y eficaz.

5. CONCLUSIONES

- Se establecieron los parámetros del funcionamiento del módulo para prácticas del Laboratorio de Comunicaciones de la Universidad Israel, en comunicación de códigos de línea a nivel de banda base, con lo cual se definen los dispositivos de procesamiento de señal; así como, de transmisión mediante modulación ASK.

- Se diseñaron los circuitos electrónicos, que incluyen procesamiento de señal en función de Arduino, así como, un transmisor y receptor en la banda de 433 MHz (UHF); ambos interactúan con aplicaciones programadas en Matlab.

- Par la interpretación de los resultados de las simulaciones códigos de línea, fue necesario el uso de librerías, las misma que permiten la extracción de los datos y su codificación en función de los parámetros de los códigos de línea que son objeto de estudio.

- En las pruebas realizadas se pudo verificar la transmisión de datos de manera serial con comunicación bidireccional, este tipo de comunicación permite al receptor enviar una señal para abrir la comunicación y con ello recibir los datos que envía el módulo transmisor, esta manera de comunicarse no será posible si no se tiene el driver del cable serial denominado CH341SER.

RECOMENDACIONES

- Crear una nueva interfaz gráfica en otra aplicación diferente a Matlab para enviar no solo un carácter sino una cadena de caracteres y verificar el funcionamiento del proceso de codificación y transmisión de datos.

- El cambio de módulos de transmisión inalámbrica para alcanzar distancias mucho más largas a las detalladas en el presente proyecto son una recomendación importante, ya que si se logra una mayor distancia que se lo puede conseguir con módulos Xbee se puede realizar un análisis mucho más específico de señales no deseadas que puedan alterar a la transmisión de datos.

- Una recomendación muy útil desarrollar el proyecto con un diferente medio de transmisión y realizar una comparación con los datos obtenidos en el presente proyecto de titulación.

- Es necesario un adecuado mantenimiento de los componentes electrónicos del módulo entregado en el presente proyecto de titulación para mantener una larga vida útil de los elementos y garantizar su correcto funcionamiento

6. REFERENCIAS

Aldaz Corrales, C. F., & Corrales, L. (2009). *Estudio de la tecnología SDR (Software Defined Radio) y posibles aplicaciones en comunicaciones inalámbricas.* Obtenido de Escuela Politécnica Nacional: http://bibdigital.epn.edu.ec/bitstream/15000/1429/1/CD-2092.pdf

Arduino. (2018). *ARDUINO UNO REV3*. Obtenido de https://store.arduino.cc/usa/arduino-uno-rev3

Arduino. (2018). *http://arduino.cl*. Obtenido de Arduino: http://arduino.cl/que-es-arduino/

Atom. (2018). *INGENIERIA DE REDES*. Obtenido de http://ingk3lmyrd.blogspot.com/p/codigo-amnchester.html

Belloch, C. (2012). Las Tecnologías de la Información y Comunicación en. Valencia, España. Recuperado el 2018, de https://www.uv.es/bellochc/pedagogia/EVA1.pdf

Brito, M. S. (2014). *SIMULACIÓN DE CÓDIGOS DE LÍNEA DESTINADA A TRANSMISIONES.* Universidad Católica de Guayaquil, Guayaquil. Obtenido de http://repositorio.ucsg.edu.ec/bitstream/3317/2876/1/T-UCSG-PRE-TEC-ITEL-73.pdf

Cabero, J. (2005). Cibersociedad y juventud: la cara oculta (buena) de la Luna. Coruña, España. Obtenido de http://tecnologiaedu.us.es/bibliovir/pdf/ciberjuve.pdf

Crespo, J. E. (2018). *Aprendiendo Arduino*. Obtenido de Aprendiendo a manejar Arduino en profundidad: https://aprendiendoarduino.wordpress.com/2016/06/27/arduino-uno-a-fondo-mapa-de-pines-2/

CyberPuerta. (2018). *Laptop Dell Inspiron 14*. Obtenido de https://www.cyberpuerta.mx/Computadoras/Laptops/Laptop-Dell-Inspiron-14-14-Intel-Core-i3-2350M-2-30GHz-4GB-500GB-Windows-7-Home-Basic.html

DET. (2018). *Convertidores Digital- Analógico y Analógico-Digital* . Obtenido de www.acomee.com.mx: https://www.acomee.com.mx/CONVERTIDORES.pdf

Elaborado por el Autor. (2018).

Eveliux. (s.f.). *eveliux.com*. Obtenido de Conversión Analógico-Digital (ADC): http://www.eveliux.com/mx/Conversion-Analogico-Digital-ADC.html

Final Test. (2018). *Tektronix TDS2022C Osciloscopio.*

Gomez, P. M. (2013). *Simposio Argentino de Sistemas Embebidos.* Obtenido de SASE: http://www.sase.com.ar/2013/files/2013/09/SASE2013-USB-P-Gomez.pdf

Huircán, J. I. (s.f.). Conversores Análogo-Digital y Digital-Análogo. Obtenido de http://quidel.inele.ufro.cl/~jhuircan/PDF_CTOSII/ad03.pdf

Manuel, F. (2010). *Redes de datos (Medios de transmisión).* Universidad de Cadiz, Facultad de Ciencias Sociales y de la Comunicación. Obtenido de http://rodin.uca.es/xmlui/bitstream/handle/10498/16867/tema05_medios.pdf

Microsoft. (2018). *Microsoft.com*. Obtenido de Visual Basic Guide: https://docs.microsoft.com/en-us/dotnet/visual-basic/

Microsoft. (s.f.). *How to send USB bulk transfer requests*. Obtenido de Hardware Dev Center: https://docs.microsoft.com/en-us/windows-hardware/drivers/usbcon/usb-bulk-and-interrupt-transfer

NAYLAMP Mechatronics. (2013). *Módulo RF 433MHz TX y RX*. Obtenido de NAYLAMP: https://naylampmechatronics.com/inalambrico/13-modulo-rf-433mhz.html

Oir, ver y contar. (s.f.). *El Conflicto del Audio*. Obtenido de https://oirverycontar.wordpress.com/2011/09/30/el-conflicto-del-audio/

patagoniatec. (2015). *Modulos Emisor Y Receptor RF 433mhz.* Obtenido de https://saber.patagoniatec.com/2015/04/modulos-emisor-y-receptor-rf-433mhz/

Prieto, J. (s.f.). *Introducción a los sistemas de comunicación inalámbricos.* Universidad Oberta de Cataluña.

RODRIGUEZ, S. I. (2016). *UNIVERSAL SERIAL BUS.* Obtenido de ULPGC: http://www.iuma.ulpgc.es/~avega/int_equipos/trab9899/usb_1/index.html

Stefanelli, A. P. (2013). Códigos de Linea. *This work is produced by OpenStax-CNX and licensed under the*. (OpenStax-CNX, Ed.) m46736. Obtenido de https://cnx.org/exports/1231eada-bbdf-4e38-bbb3...pdf/4códigos-de-linea-1.pdf

Stiven, M. (2009). *Electronics*. Obtenido de http://marwinteck.blogspot.com/2009/01/codigos-de-linea.html

Telecomundo. (2016). *Modulaciones digitales ASK, FSK y PSK.* Obtenido de telecomundo.

Tutoriales Point. (2018). *Simple & Easy Learning.* Obtenido de https://www.tutorialspoint.com/digital_communication/digital_communication_amplitude_shift_keying.htm

Universidad Don Bosco. (2015). Guia sistemas de comunicación II. El Salvador. Obtenido de http://www.udb.edu.sv/udb/archivo/guia/electronica-ingenieria/sistemas-de-comunicacion-ii/2012/ii/guia-4.pdf

ANEXO 1

ALGORITMO ARDUINO TRANSMISOR

```
#include <VirtualWire.h>
 //llibrería duplicar puerto serial
const int dataPin = 9;
const int ledPin = 13;
//configuración de pines para rf

void setup()
{
   vw_setup(2000); // velocidad de transmisión para modulos RF
   vw_set_tx_pin(dataPin); // iniciar transmisión de modulos rf en pin 9
   Serial.begin(9600); //velocidad de transmisión para arduino y Matlab
}

void loop()
{
  char *msg = "";
       // para enviar la cadena de bits, empiezo a almacenar los datos en un puntero
if(Serial.available() > 0){
   char c = Serial.read();
   //los datos que recibo en el puerto serial (matlab) los envio por los módulos RF

    //los if colocados son para realizar la conversión de comillas simples, a comillas dobles (incompatibilidad de formato) debido a las librerias utilizadas
    if(c=='1'){
```

```
 msg="1";

}
if(c=='2'){
 msg="2";
}
    if(c=='3'){
 msg="3";
}
    if(c=='4'){
 msg="4";
}
    if(c=='5'){
 msg="5";
}
       if(c=='6'){
 msg="6";
}
if(c=='7'){
 msg="7";
}
 if(c=='0'){
 msg="0";
}
if(c=='a'){
 msg="a";
}
if(c=='b'){
```

```
     msg="b";
    }
   if(c=='c'){
     msg="c";
    }
        if(c=='d'){
     msg="d";
    }

 }
  //char msg=Serial.read();

    digitalWrite(ledPin, true); // cuando empiezo la transmisión enciendo un pin
    vw_send((uint8_t *)msg, strlen(msg)); //envio de la cadena mediante el puerto virtual que ocupa los módlos rf
    vw_wait_tx(); //pausa necesaria para realizar el envio sin sobreposición de datos
    digitalWrite(ledPin, false); // apago el led que indica que la transmisicón culmino
    delay(200); // colocamos un retardo para evitar que inlfuya la velocidad de transmisión sobre el procesamiento de los datos
}
```

ANEXO 2

PROGRAMACIÓN ARDUINO RECEPTOR

```
#include <VirtualWire.h>
//llibrería duplicar puerto serial

const int dataPin = 9;
const int ledPin = 13;
//configuración de pines para rf

void setup()
{
   Serial.begin(9600); //velocidad de transmisión para arduino y Matlab
   vw_setup(2000);   // velocidad de transmisión para modulos RF
    vw_set_rx_pin(dataPin); // iniciar transmisión de modulos rf en pin 9
   vw_rx_start();
   pinMode(10,OUTPUT); //configurar pines de salida para ver en osciloscopio
   pinMode(11,OUTPUT);//configurar pines de salida para ver en osciloscopio
}

void loop()
{
   uint8_t buf[VW_MAX_MESSAGE_LEN];
   uint8_t buflen = VW_MAX_MESSAGE_LEN;
   //limpiar buffer de puerto com, necesario para comunicación serial
```

```
  if (vw_get_message(buf, &buflen))
  //Obtiene la candena recibida mediante los modulos rf
  {
    digitalWrite(ledPin, true);
 //cuando inicio una transmisión enciendo un led
    for (int i = 0; i < buflen; i++)
    {
//       Serial.print((char)buf[i]);
 char aux=(char)buf[i];
 // la cadena se almacena un vector que debo ir leyendo mediante un for
       Serial.print(aux);
     for (int i = 0; i < buflen; i++)
    {
      Serial.print((char)buf[i]);
      //envio mediante comunicación serial la cadena a Matlab
 char aux=(char)buf[i];



     if(aux=='1'){
         digitalWrite(10, HIGH);
        digitalWrite(11, HIGH);
        //cada que reciba un 1 lógico enciendo dos pines del arduino
         }
       if(aux=='0'){
        digitalWrite(10, LOW);
        digitalWrite(11, LOW);
//cada que reciba un 0 lógico apago dos pines del arduino
```

```
        }
              if(aux=='2'||aux=='3'||aux=='4'||aux=='5'||aux=='6'||aux=='7'){
         digitalWrite(10, LOW);
        //para detener la transmisión recibo un 2
        //para determinar el tipo de codificación que envíe, recibo un numero entre 3 y
7
        }
           }
       }

    Serial.println("");
     digitalWrite(ledPin, false);
  }
```

ANEXO 3

PROGRAMACIÓN TRANSMISOR MATLAB

```
function varargout = tx(varargin)
% TX MATLAB code for tx.fig
%      TX, by itself, creates a new TX or raises the existing
%      singleton*.
%
%      H = TX returns the handle to a new TX or the handle to
%      the existing singleton*.
%
%      TX('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in TX.M with the given input arguments.
%
%      TX('Property','Value',...) creates a new TX or raises the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before tx_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to tx_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help tx

% Last Modified by GUIDE v2.5 24-Jan-2019 13:34:22

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @tx_OpeningFcn, ...
                   'gui_OutputFcn',  @tx_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
```

```
% End initialization code - DO NOT EDIT


% --- Executes just before tx is made visible.
function tx_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to tx (see VARARGIN)

% Choose default command line output for tx
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes tx wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = tx_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
v=get()
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a
double


% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
popupmenu1


% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

ing=get(handles.edit2,'string');
b=[];
for i=1:length(ing)
    b(i)=str2num(ing(i));
end

bits=b;
% bits=[1 0 1 1 1 0 1]
```

```
f=[];
for j=1:length(bits)
anc=0:0.001:1;
b=square(anc);
f=[f b*bits(j)];

end
t=0:length(bits)/length(f):length(bits)-(length(bits)/length(f));
plot(t,f)
v=get(handles.popupmenu1,'Value');
switch v
    case 1
        title('Se codificará utlizando NRZ Polar');
    case 2
        title('Se codificará utlizando Manchester Diferencial');
    case 3
         title('Se codificará utlizando Manchester');
    case 4
         title('Se codificará utlizando CMI');


end
axis([-0.5 length(bits) -1 1.5])
grid on;
puerto=get(handles.edit3,'string');
enviarDatosArduino(bits,v,puerto)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3 as a
double


% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');end
```

ANEXO 4

PROGRAMACIÓN RECEPTOR MATLAB

```
function varargout = rx(varargin)
% RX MATLAB code for rx.fig
%      RX, by itself, creates a new RX or raises the existing
%      singleton*.
%
%      H = RX returns the handle to a new RX or the handle to
%      the existing singleton*.
%
%      RX('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in RX.M with the given input arguments.
%
%      RX('Property','Value',...) creates a new RX or raises the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before rx_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to rx_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help rx

% Last Modified by GUIDE v2.5 28-Jan-2019 15:45:30

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @rx_OpeningFcn, ...
                   'gui_OutputFcn',  @rx_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
```

```
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before rx is made visible.
function rx_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to rx (see VARARGIN)

% Choose default command line output for rx
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes rx wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = rx_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
puerto=get(handles.edit2,'string');
y1=[];

delete(instrfind({'Port'},{puerto}));
puerto_serial=serial(puerto);
puerto_serial.BaudRate=9600;
fopen(puerto_serial)
pause(1);
while (true)
 a=fscanf(puerto_serial,'%d');
if(a==2)
    break;
end
 y1=[y1 a];
end
```

```
fclose(puerto_serial);
 y2=y1(1:length(y1)-1);

y2
tipo=y1(length(y1))

switch tipo
    case 3
        %bc=codNrzPolar(y2);
%         disp('Codificación NRZ');
        bc=rceroneg(y2);
        grafCodNrzPolar(bc);
    case 4
        %bc=codManchester(y2);
%         disp('Codificación Manchester Diferencial');
        grafCodManchester(y2);
    case 5
        %bc=codManchesterDif(y2);
%         disp('Codificación Manchester ');
        grafCodManchesterDif(y2);
    case 6
%         disp('Codificación Cdmi');
        grafCodCmi(y2);
     case 7
%          disp('Codificación Hdb3');
        grafcodHdb3(y2);

end
%osci('COM3', bc);



function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a
double


% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end



function edit2_Callback(~, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a
double


% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

ANEXO 5

PRÁCTICAS DE LABORATORIO