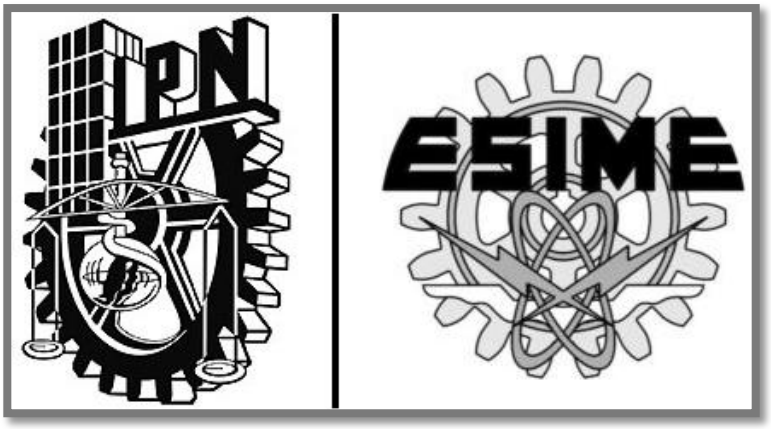


“IMPLEMENTACIÓN DE ESQUEMAS DE CODIFICACIÓN DE CANAL MEDIANTE UN GENERADOR DE SEÑALES ARBITRARIAS”

INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE INGENIERÍA MECÁNICA Y ELECTRICA

UNIDAD PROFESIONAL "ADOLFO LÓPEZ MATEOS"

T E M A D E T E S I S

QUE PARA OBTENER EL TITULO DE: **INGENIERO EN COMUNICACIONES Y ELECTRÓNICA**

POR LA OPCIÓN DE TITULACIÓN: **TESIS COLECTIVA Y EXAMEN ORAL INDIVIDUAL**

DEBERA(N) DESARROLLAR: **C. JUAN MARTIN NAPOLES MUNGUIA**
C. OMAR JAIR VELAZQUEZ GUZMAN

"IMPLEMENTACIÓN DE ESQUEMAS DE CODIFICACIÓN DE CANAL MEDIANTE UN GENERADOR DE SEÑALES ARBITRARIAS"

INTEGRAR EXPERIMENTALMENTE UNA APLICACIÓN DE ESQUEMAS DE CODIFICACIÓN DE CANAL EN EL LABORATORIO DE COMUNICACIONES DE ESIME-ZACATENCO PARA EL ESTUDIO Y ANÁLISIS DE LAS UNIDADES DE APRENDIZAJES QUE LO REQUIERAN.

- INTRODUCCIÓN
- JUSTIFICACIÓN
- ELEMENTOS DE COMUNICACIONES DIGITALES
- CANAL DE COMUNICACIÓN
- CODIFICACIÓN DE CANAL
- DESARROLLO EXPERIMENTAL
- CONCLUSIONES
- BIBLIOGRAFÍA/REFERENCIAS

MÉXICO D.F. A 15 DE OCTUBRE DE 2013

A S E S O R E S

ING. JOSE ERNESTO ROJAS LIMA

ING. ERIC GOMEZ GOMEZ

ING. PEDRO GUSTAVO MAGAÑA DEL RIO

ING. PATRICIA LORENA RAMIREZ RANGEL
JEFE DEL DEPARTAMENTO DE
INGENIERÍA EN COMUNICACIONES Y ELECTRÓNICA

Dedicatorias

A **Julieta**, mi **madre** por tu apoyo incondicional

a lo largo de mi carrera, por tus consejos que

lograron mantenerme en equilibrio y

me permitieron lograr este éxito,

por las amonestaciones

merecidas

cuando eran necesarias recibirlas, por el abrazo amoroso

que me otorgaste cuando más lo necesitaba.

Gracias Mamá.

Juan

Dedicatorias

Antes de todo, quiero agradecer a **Rosa María** mi **madre** y a **mi padre Donato**, quienes me inculcaron un deseo maravilloso de triunfo y que quizá inconscientemente, me inspiraron un deseo inigualable de crecer y servir. Agradezco su dedicación e incondicional amor que preservaré hasta el fin de mi existencia. **Gracias por todo.**

A mi **abuela Rosaura**, quien improvisada y maravillosamente me enseñó el significado de “vivir”. Gracias por las memorias.

Omar Jair

Agradecimientos

Agradecemos al **Instituto Politécnico Nacional** por la oportunidad que se brindó a los autores de esta tesis, ya que sin la confianza y la oportunidad a nosotros otorgadas por la Institución, hubiera sido imposible la realización profesional.

A ESIME,

Por ser la escuela que nos brindó un sinfín de conocimientos y experiencias que ayudaron a formar los profesionales que hoy en día somos.

A los **profesores**, por su dedicación y su cátedra inigualable, quienes definitivamente forjaron nuestro carácter y perspectiva; tanto a los poseedores de un título, como a los que no.

Del mismo modo, se agradece el apoyo incondicional y profesional de los académicos que nos impulsaron en el desarrollo de este trabajo, así como su humildad y la inigualable cantidad de enseñanzas propiciadas a lo largo de esta travesía: **José Ernesto Rojas Frías, Eric Gómez Gómez y Pedro Gustavo Magaña del Río.**

Ultimando y no con menos importancia, se hace amplio el sincero agradecimiento **a nuestras familias**, quienes durante nuestra carrera tanto profesional como de vida nos proveyeron de cariño y estabilidad, y a las personas que participaron en el óptimo progreso de este trabajo, subjetiva y objetivamente.

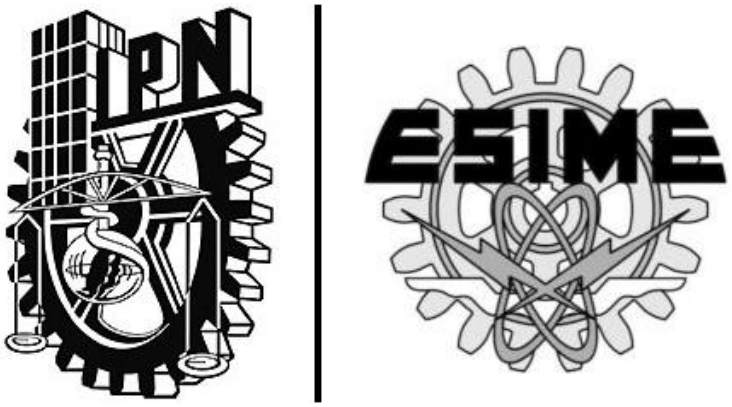
.

Juan y Omar Jair

OBJETIVO

Integrar experimentalmente una aplicación de esquemas de codificación de canal en el Laboratorio de Comunicaciones de ESIME-Zacatenco para el estudio y análisis de las unidades de aprendizajes que lo requieran.

ÍNDICE



INTRODUCCIÓN

INTRODUCCIÓN

El Instituto Politécnico Nacional, a lo largo de su historia, desarrollo y participación social, se ha dado a la tarea de formar ingenieros capaces de satisfacer las necesidades tecnológicas que van surgiendo en el país. Ha sido la institución más representativa que a lo largo de la historia nacional ha destacado en el campo tecnológico-evolutivo mundial.

Podría argumentarse que este hecho es debido a la calidad de los académicos que conforman la plantilla educacional, específicamente refiriéndose a la Escuela Superior de Ingeniería Mecánica y Eléctrica, dónde se mezcla la calidad humana, la labor profesional y una experiencia inimaginable.

Desafortunadamente, por cuestiones generalmente ajenas a las educacionales, la materia educativa se ha visto limitada, ya que para estar a la vanguardia de las nuevas tecnologías del día a día, se debe precisar de una amplia gama de elementos fundamentales para competir en la industria, de los cuales se puede citar: actualización metodológica, supervisión académica frecuente, desarrollo de trabajo conjunto de las academias, y quizá la más importante, la adquisición de nuevas tecnologías para el estudio de temas específicos.

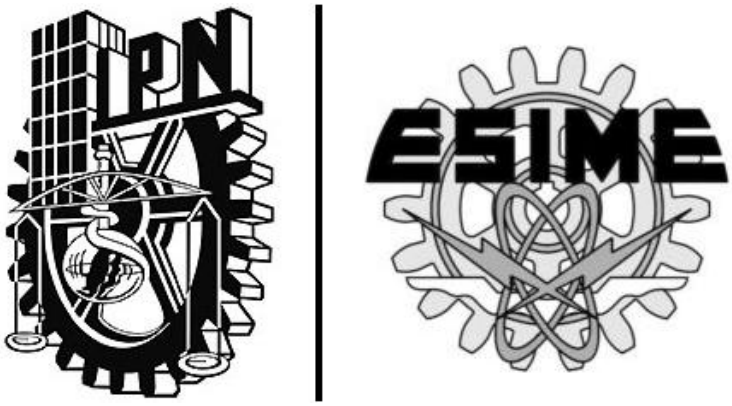
Centrándose en el último tema de crucial importancia, cabe destacarse que dichas herramientas tecnológicas pueden considerarse de mayor importancia ya que existe una gran variedad de fenómenos que son perceptibles solamente a través de determinado equipo, cuyo precio es inimaginable, la mayoría de las veces.

Afortunadamente, la intuición ingenieril que se ha desbordado y preservado en nuestra prestigiada institución, hace posible la generación de “tangentes” u “oportunidades alternativas” por medio del desarrollo de prototipos de índole virtual o de dispositivos diseñados con elementos de menor precio, lo que a su vez,

evoca un reconocimiento aun más importante de la calidad educativa del Instituto Politécnico Nacional.

El presente trabajo, es un ejemplo de desarrollo y subsistencia académica específicamente evocado para el estudio de la *codificación de canal mediante algoritmos en conjunto con un generador de señales arbitrarias*, un tema de suma importancia hoy en día en los sistemas de comunicación digital, con lo que se ha pretendido satisfacer necesidades y postergar una enriquecedora aportación con la única y vivaz intención de secundar a las generaciones emergentes, pero más que nada, para retribuir la enorme riqueza intelectual que la Escuela Superior de Ingeniería Mecánica y eléctrica nos ha facilitado.

Un desarrollo, una necesidad, una expectativa. Tres elementos hilados por una noble entereza que pretende generar frutos y ampliar horizontes para el intelecto mexicano venidero. Un esfuerzo inigualable del que se espera una reacción favorable de todas las partes, tanto subjetivas como objetivas, que se han visto y se verán involucradas en este progreso.



ANTECEDENTES

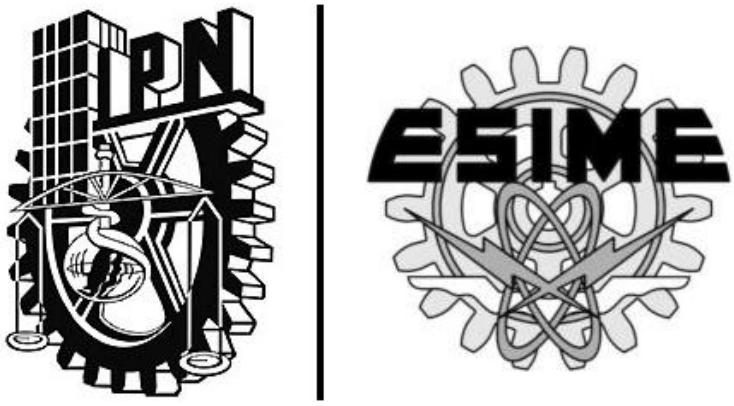
JUSTIFICACIÓN

Con las nuevas tecnologías y los niveles intelectuales de los estudiantes contemporáneos que frecuentan las aulas de la Escuela Superior de Ingeniería Mecánica y Eléctrica, se ha generado la necesidad de campos de estudio un tanto más complejos por lo que hoy, con el nuevo contexto tecnológico, se ha presentado la oportunidad de generar aplicaciones virtuales para el estudio de temas tan complejos, como lo es la codificación de canal, en el campo de las comunicaciones.

Siendo un tema profundo y sofisticado, la codificación de canal como tal solo ha sido penetrada de manera abstracta debido a su naturaleza, y a la ausencia de los elementos tecnológicos necesarios para su observación y comprensión.

Ya que es más difícil satisfacer las carencias económicas, se debe buscar la alternativa “virtual”, para que así como este tema, otros tópicos afines a los sistemas de comunicación puedan ser atacados y digeridos con el único fin de incrementar el valor del raciocinio y la materia académica del intelecto politécnico.

Abordando los conceptos elementales de la teoría de las comunicaciones, se empieza una justificación sobre el por qué de la aplicación que se ha generado en este proyecto, su relación con el canal de transmisión, las afectaciones que pudiese tener el mismo, y la teoría necesaria para comprender la importancia que tienen los sistemas de comunicación en todas y cada una de sus variantes.



CAPÍTULO I | "ELEMENTOS DE COMUNICACIONES DIGITALES"

CAPÍTULO 1. ELEMENTOS DE COMUNICACIONES DIGITALES

La necesidad primordial que el humano tiene de comunicarse con sus relativos ha ido generando, conjunto al paso del tiempo, una evolución inminente en los sistemas que sirven como medio para llevar a cabo las comunicaciones a un nivel más personal.

Desde que se desarrolló el primer aparato telefónico, su demanda ha ido creciendo exponencialmente refiriéndonos solamente a un enfoque comercial, pero en su totalidad, todos los eslabones del sistema de comunicaciones han sido víctima de cambios irrevocables por su naturaleza evolutiva.

Esencialmente, las *comunicaciones digitales* se definen como la transmisión física de información, generalmente un *flujo de bits*, desde un transmisor hasta un receptor a través de un *canal de comunicaciones*.

Cabe entonces, partir de los fundamentos teóricos que se han ido rescatando a lo largo de los años para emprender con mesura y total comprensión un progreso del estudio del sistema de comunicación fundamental, haciendo énfasis en las etapas de emisión de flujo de información y codificación del mismo.

1.1 Conceptos básicos de Comunicaciones Digitales

Para tener una concepción eficaz y un punto de partida para el estudio de las comunicaciones digitales, hay que tener en cuenta el *diagrama básico de un sistema de comunicaciones*, el cual se presenta a continuación en la figura 1.1:

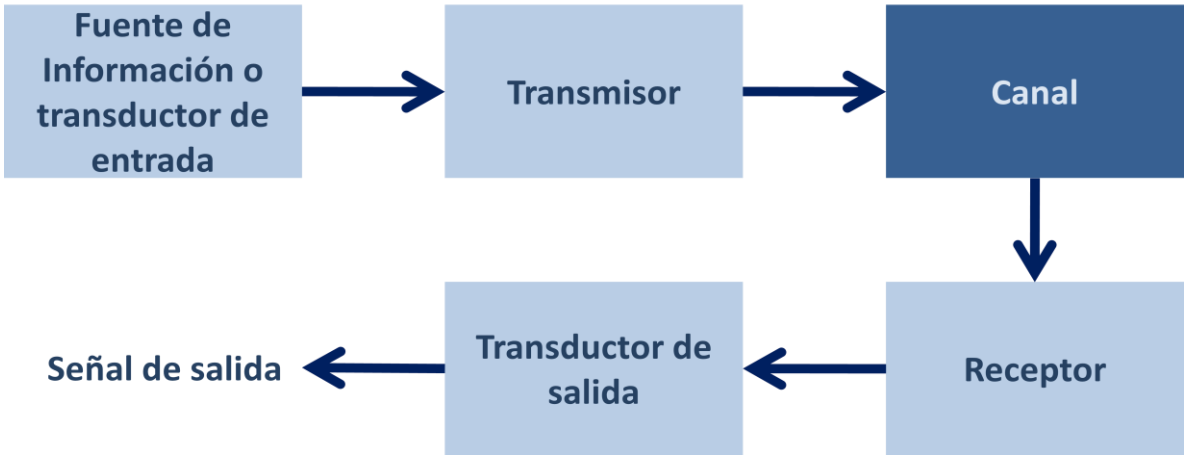


Figura 1.1 Diagrama básico de un sistema de comunicaciones

Se puede decir que un sistema *de comunicaciones digitales*, parte del diagrama básico de un sistema de comunicaciones, por lo que es conveniente hacer un preámbulo del sistema elemental.

El principal motivo por el que se construye un sistema de comunicaciones es para el *envío de información* desde una fuente de información que genera un *mensaje* que se hará llegar a uno o más *destinatarios.* La información que se genera puede ser voz, o texto simple en algún lenguaje en particular, por citar algunos ejemplos. Una característica esencial de cualquier *fuente de información* es que su salida es descrita en *términos probabilísticos*. Esto quiere decir que la salida del sistema de comunicaciones *no es determinística,* porque de otro modo, la transmisión del mensaje no sería necesaria. ¿Qué caso tendría enviar un mensaje si ya se conoce o puede ser predicho o calculado antes de ser transmitido?

Un *transductor* será requerido ya que se necesita convertir la salida de la fuente de información en una señal eléctrica (o luminosa en los sistema de Fibra Óptica) para que sea adecuada para la transmisión. En la etapa final, donde se está por llegar al destino, se requiere un *transductor* similar al de la primera etapa con el propósito de convertir la señal eléctrica en información interpretable.

El corazón de un sistema de comunicaciones consiste básicamente en: *el transmisor, el canal* y *el receptor.* Ahora bien, cabe presentar en la figura 1.2 el *diagrama a bloques de un sistema de comunicaciones digitales* para después hacer un análisis del mismo:

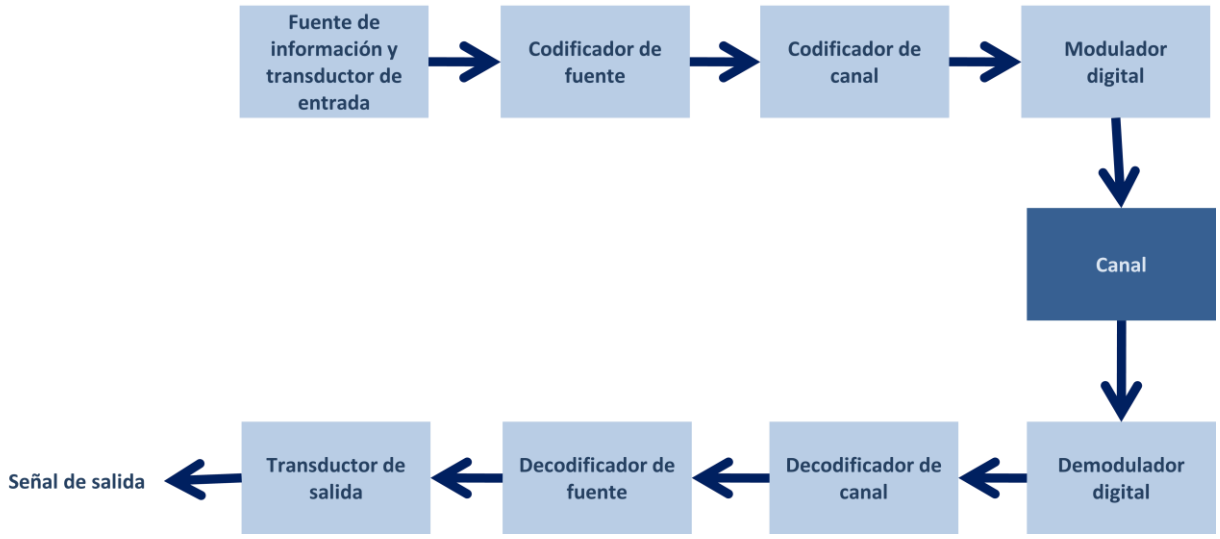


Figura 1.2 Diagrama a bloques simplificado de un sistema de comunicaciones digitales

En un sistema de comunicaciones digitales, las operaciones funcionales hechas por el transmisor y el receptor deben expandirse con el fin de incluir una discretización de la señal del mensaje para el *transmisor* y una síntesis o interpolación del mensaje en la etapa del *receptor*. Adjunto a estas operaciones funcionales se incluye supresión de redundancia y la codificación y decodificación de canal.

La salida de la fuente podría ser analógica, pero requeriría de una conversión analógica-digital adicional, por lo que generalmente, dicha salida es digital. En la mayoría de los casos, el mensaje de la fuente se convierte en una secuencia de dígitos binarios. El proceso de convertir la salida, ya sea analógica o digital, de la fuente en una secuencia de dígitos binarios se llama *codificación de la fuente* o

compresión de datos que tendrá como fin poca o nula *redundancia*. Esta información compresa (*secuencia de información*) se lleva al *codificador de fuente* el cual tiene como propósito introducir controladamente, cierta *redundancia* en la información binaria que será usada para combatir los efectos del ruido e interferencia que se encontrará en la transmisión de la información a través del *canal*.

La secuencia binaria a la salida del codificador de canal es llevada a la parte del *modulador digital* que sirve como interfaz para el canal de comunicación. Uno de los motivos primordiales de la modulación digital es mapear la secuencia binaria de información.

Una vez atravesado el canal, la información llega al *demodulador digital* que hará una representación de la secuencia y hará tomas de decisiones sobre los símbolos detectados de la secuencia binaria que se está recibiendo, con el fin de optimizar la señal a través de la detección.

En la etapa final cuando se desea una salida analógica, el *decodificador de fuente* acepta la secuencia de salida del *decodificador de canal* y se lleva a cabo una reconstrucción original de la fuente de información. Debido a los errores que se adhieren a la señal en la decodificación de canal y la posible distorsión que se produce el decodificador de la fuente se hace una aproximación de la señal original de salida de la fuente de información. La diferencia entre la señal original y la señal reconstruida es una ponderación de la distorsión introducida por el sistema de comunicaciones digitales.

Cabe reconocer que las comunicaciones digitales se han vuelto notoriamente usadas ya que dichos sistemas cuentan con facilidades y características que los vuelven más amigables, ventajosos y certeros comparados con sistemas analógicos. Dichas características de los sistemas de comunicación digital, serán citadas a continuación.

1.1.1 Fuente de Información y elementos conceptuales vinculados

La *fuente de información* es el elemento del cuál parte el sistema entero. Se trata de un recurso que genera la información que será transmitida a través del sistema. Puede tratarse de una fuente analógica, que posteriormente será sometida a una conversión digital, de modo que sea compatible con el medio. Dicha transformación se lleva a cabo a través de un muestreo (Una discretización temporal de la señal analógica) y una cuantización (Una jerarquización de los niveles energéticos de la señal en conversión).

Un ejemplo de fuente de información podría ser un *dispositivo de almacenamiento* de cuyo contenido se extrae la información requerida a través de un computador y es procesada a través del mismo.

La señal emitida por la fuente de información en un sistema digital suele llamarse *mensaje textual* que es una sucesión de símbolos. En el entorno digital, el mensaje es una *secuencia de dígitos*, también llamados *símbolos*, que se toman de un conjunto finito de elementos o de un alfabeto.

El *carácter*, es miembro de un alfabeto o conjunto finito de símbolos. Dicho elemento puede ser representado por una secuencia de dígitos binarios.

Se tiene conocimiento de códigos estandarizados, de los cuales se pueden citar el ASCII (*Código Estándar Estadounidense para el Intercambio de Información)* o el Morse (representación de letras o números mediante tonos o señales intermitentes).

Un *dígito binario (bit)*, es la unidad fundamental de información que todos los sistemas digitales manejan. Suele tener dos estados representados con un 1 (encendido) o un 0 (apagado).

Al secuenciar cierto número de bits, se logrará generar un *flujo de bits* (unos y ceros). Generalmente, los bits secuenciados pueden representarse gráficamente como “pulsos” teniendo la mayoría de las veces solo dos niveles (alto/bajo, encendido/apagado). El flujo de bits es considerado banda base, lo que quiere decir que su contenido espectral se extiende desde (o muy cerca, en el caso real y no ideal) un nivel de dc hasta un valor finito que generalmente oscila en la escala de los MHz. El término *banda base* sirve para designar la banda de frecuencias que representan una señal original entregada por determinada fuente de información. El uso adecuado del canal de comunicación requiere un cambio del rango de frecuencias de la banda base en otro rango de frecuencias apropiado para la transmisión y un cambio correspondiente de vuelta al rango original de frecuencias después de la recepción.

El *símbolo* es un número de bits determinado, que se considera como la unidad o carácter primario de un grupo finito de símbolos, con un número de elementos específico, cuya representación de cada uno de ellos se llevará a cabo a través de determinadas formas de onda correspondientes.

Existe una diferencia perceptible entre un *carácter* y un *símbolo.* Se argumenta que el *carácter* es una unidad de información que no *implica una exhibición visual de sí mismo*, ya que suele usarse para un control o una representación específica de datos. En cambio, el *símbolo* es la cantidad más pequeña que puede ser transmitida a través de un sistema. Esta cantidad puede ser hasta un pulso que ha de tener alguna *representación específica*.

En la figura 1.3 se presenta una descripción gráfica de los conceptos que se abordaron en esta primera parte, con la finalidad de proporcionar un mejor entendimiento de los mismos.

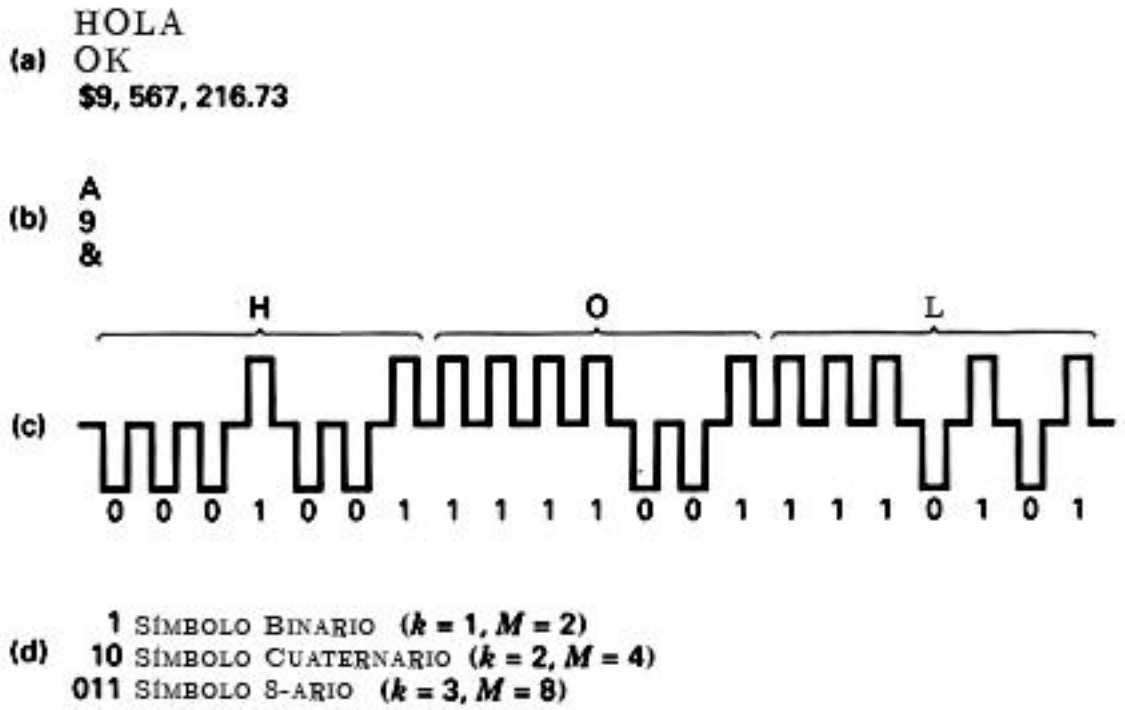


Figura 1.3 a) Mensaje Textual b) Carácter c) Flujo de bits d) Símbolos

1.1.2 Velocidad de símbolo

La velocidad de símbolo (Symbol Rate) es la cantidad de símbolos emitidos por segundo. Ya que el bit es la unidad básica en los sistemas de comunicaciones digitales tradicionales, esta velocidad suele denominarse bits/s.

En un sistema M-ario, la fuente de información emite una secuencia de símbolos de un alfabeto que consiste en M símbolos. Cada nivel de amplitud a la salida del modulador de amplitud de pulsos del sistema corresponde a un símbolo distinto, por lo que habrá M niveles de amplitud para ser transmitidos.

Se considerará un sistema PAM (Pulse-amplitude modulation ó *Modulación por amplitud de pulso*) M-ario con un alfabeto que contiene M símbolos

estadísticamente equiprobables e independientes, donde se denota la duración de símbolo como T segundos. Entonces, se referirá a $1/T$ como la *velocidad de señalización* o *velocidad de símbolo* del sistema, que se expresa en *símbolos/segundo* o *Bauds*.

Cabe relacionar la *velocidad de símbolo* de éste sistema PAM M-ario con un sistema PAM binario donde el valor de M es 2 y los símbolos binarios pueden ser 0 o 1 estadísticamente equiprobables e independientes, con una duración de símbolo denotada por T_b segundos. Bajo estas condiciones, el sistema PAM binario produce información a una velocidad de $1/T_b \ \ bits/segundo$.

Respecto al *Baud* se puede declarar que tomando en cuenta un sistema PAM M-ario, 1 *Baud* es igual a $\log_2 M \ \ bits/segundo$, y la duración de símbolo T de un sistema PAM M-ario está relacionada a la duración de bit T_b del sistema equivalente PAM binario como:

$$T = T_b \log_2 M \tag{1.1}$$

Por lo tanto, en un ancho de banda de canal dado, se encuentra que usando un sistema PAM M-ario se puede transmitir información a una velocidad que es $\log_2 M$ más rápida que la correspondiente a un sistema PAM M-ario. Sin embargo, para que se tenga la misma *probabilidad media de error de símbolo*, un sistema PAM M-ario requiere más potencia de transmisión.

La *velocidad de símbolo* está muy relacionada con el ancho de banda. El medio de transmisión de las señales limita mucho las componentes de frecuencia a las que puede ir la señal, por lo que el medio sólo permite la transmisión en cierto ancho de banda. Se puede demostrar que al duplicar el ancho de banda, se duplica la velocidad de transmisión a la que puede ir la señal ya que al considerar que el ancho de banda de una señal está concentrado sobre una frecuencia central, al

aumentar ésta, aumenta la velocidad potencial de transmitir la señal; pero al aumentar el ancho de banda, aumenta el coste de transmisión de la señal aunque disminuye la distorsión y la posibilidad de ocurrencia de errores.

1.2 Clasificación de Señales

Una señal es un conjunto de información o datos. Algunos ejemplos de señales pueden ser la señal de televisión o de teléfono, o la gráfica de las ventas mensuales en una corporación. En estos ejemplos, las señales están en función del *tiempo*, que es una variable independiente. En el caso de los sistemas de comunicación se manejan exclusivamente señales en función del tiempo.

Las señales se procesan a través de sistemas, los cuales pueden modificar las señales o extraer información de ellas. Así, un *sistema* es una entidad que procesa un conjunto de señales (entradas) para producir otro conjunto de señales (salidas).

Un sistema puede estar compuesto por componentes físicos, o también puede estar compuesto por un algoritmo que calcule una salida a partir de una señal de entrada.

Existe una gran variedad de señales, por lo que se presenta una clasificación de las mismas, adecuada y relacionada con los sistemas de comunicaciones.

1.2.1 Analógicas y Digitales

Una señal que está definida para cada valor del tiempo t es una *señal continua en el tiempo*. Una señal que está definida solo para puntos discretos de $t = nT$ es una *señal discreta en el tiempo*. En la figura 1.4 se muestran gráficamente las propiedades de las señales *continuas en tiempo* y *discretas en tiempo.* Las señales de audio y de video son señales continuas en tiempo, mientras que la representación gráfica de las ventas mensuales en una corporación son señales discretas en tiempo.

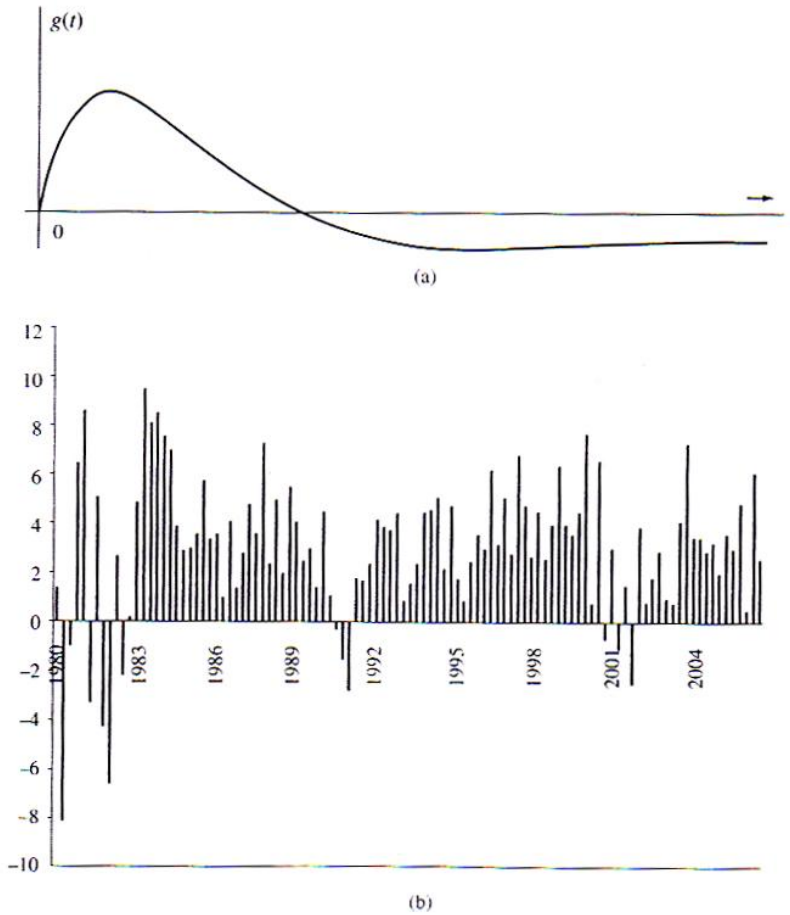


Figura 1.4 a) Señal continua en tiempo b) Señal discreta en tiempo

No se debe confundir una *señal análoga* con una *señal continua en el tiempo* ya que los conceptos son diferentes. El mismo criterio de diferenciación debe utilizarse para las *señales discretas en el tiempo* y las *señales digitales*.

Una señal cuya amplitud puede tomarse en cualquier valor de un rango continuo es una *señal analógica*. Esto significa que la amplitud de una señal analógica puede ser tomada de un número infinito de valores.

Por el otro lado, una *señal digital* es aquella cuya amplitud solo puede ser tomada solo de un número finito de valores. Las señales asociadas con las computadoras digitales se consideran señales digitales ya que su amplitud solo se toma de dos valores (señales binarias). Para que una señal sea digital, no es necesario que cuente solo con dos valores, pero tiene que ser un valor finito. Una señal digital cuyos valores en amplitud son tomados de M valores es una señal M-aria donde la señal binaria ($M = 2$) es un caso especial.

Los términos *continuo en tiempo* y *discreto en tiempo* definen la naturaleza de las señales a través del tiempo. Por el otro lado, los términos *analógico* y *digital* definen la naturaleza de las señales en su amplitud. En la figura 1.5 se muestran ejemplos de los diferentes tipos de señales en cuanto a éstos términos refiere:

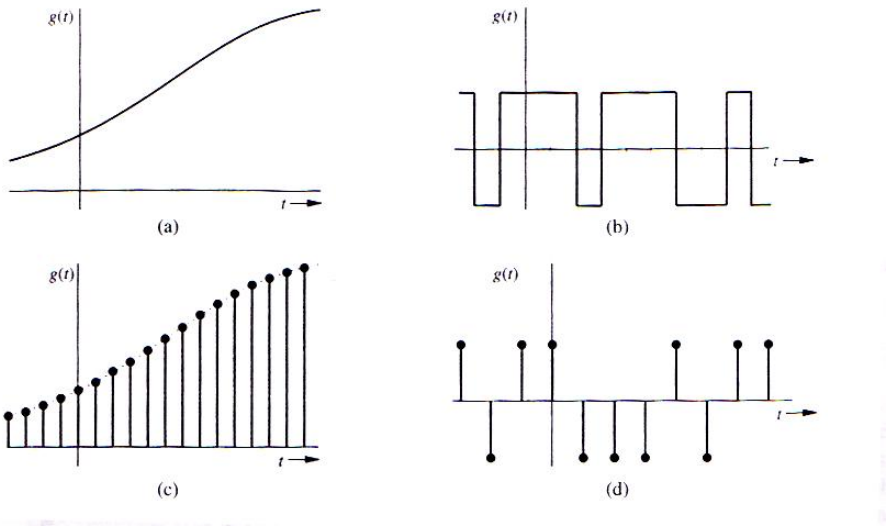


Figura 1.5 Ejemplos de señales: a) Analógica y continua en tiempo b) Digital y continua en tiempo c) Analógica y discreta en tiempo d) Digital y discreta en tiempo

1.2.2 Periódicas y No Periódicas

Se dice que una señal es periódica en el tiempo si existe una constante $T_0 > 0$ tal que:

$$x\ t\ = x(t + T_0) \qquad Para - \infty < t < \infty \tag{1.2}$$

Donde t denota el tiempo. El valor mínimo de T_0 que satisface ésta condición es llamada el *periodo* de $x\ t$. El periodo T_0 define la duración de un ciclo completo de $x\ t$. En la figura 1.6 se hace una representación gráfica de tres ejemplos de señales periódicas (senoidal, cuadrada, triangular y diente de sierra) en donde se resalta el periodo T_0 en cada una de ellas:

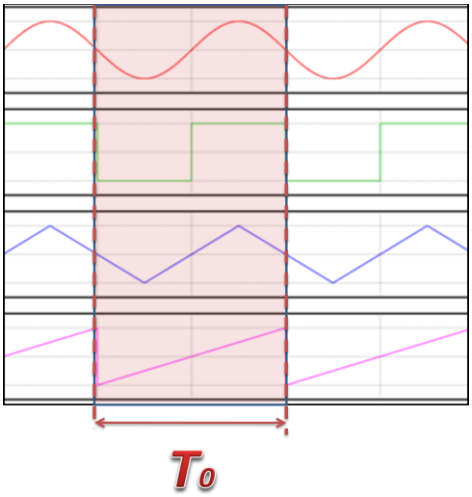


Figura 1.6 Ejemplos de señales periódicas, donde T_0representa un periodo.

Si una señal no tiene un valor T_0 que satisfaga a $x\ t$, entonces se tiene el caso de una función *no periódica,* como lo es el ruido, que se presenta gráficamente en la figura 1.7, ya que en su comportamiento no es posible la acotación de un ciclo:

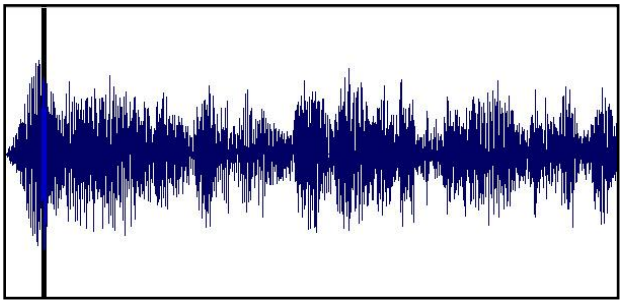


Imagen 1.7 Ejemplo de señal no periódica

1.2.3 Determinísticas y Aleatorias

Una señal cuya descripción física es conocida en su totalidad, ya sea matemática o gráficamente, es considerada una *señal determinística.* Su comportamiento carecerá de incertidumbre respecto a cualquier valor del tiempo. Se sabe que una vez que la señal se ha producido, es entonces cuando se adhiere cierto grado de azar a la misma.

Una señal que es solo conocida por una descripción probabilística, tal como valor promedio, valor eficaz y distribuciones en lugar de su descripción matemática o gráfica es una *señal aleatoria.* La mayor parte de las señales de ruido que se encuentran en la práctica son señales aleatorias. Todas las señales de mensajes son señales aleatorias porque para que esta transmita información debe llevar consigo cierta incertidumbre (aleatoriedad).

Un ejemplo de señal determinística podría ser la función matemática que describe cómo una manzana cae de un árbol ya que, como se conoce su comportamiento de caída, la manzana se regirá por dichas leyes de gravedad aplicables en este planeta. Por el contrario, el ejemplo más obvio de un evento aleatorio es la moneda que es arrojada al aire, cuyo resultado es incierto hasta que la moneda quede inmóvil en el suelo.

Ya que se refiere a modelos en la forma de descripción probabilística de los procesos aleatorios, esto será particularmente útil para ciertas señales involucradas en los sistemas de comunicación, por ejemplo el ruido.

1.2.4 Energía y Potencia

Para un análisis concreto de este tipo de señales, se hablará de una señal $x(t)$ que representará el voltaje a través de una resistencia R . No se sabrá si $x(t)$ será una señal de corriente o de voltaje, y con el propósito de normalizar la potencia, se asume que el voltaje a través de R será de 1Ω con lo que se tendrá una potencia asociada con la señal, partiendo de la Ley de Ohm. De acuerdo a esto, se define:

La energía de la señal sobre un intervalo de tiempo de longitud 2L:

$$E_{2L} = \int_{-L}^{L} \left|x(t)\right|^2 dt \tag{1.3}$$

La energía total de la señal en el intervalo t desde -infinito hasta infinito:

$$E = \lim_{L\to\infty} \int_{-L}^{L} \left|x(t)\right|^2 dt \tag{1.4}$$

La potencia promedio:

$$P = \lim_{L\to\infty} \left[\frac{1}{2L} \int_{-L}^{L} \left|x(t)\right|^2 dt \right] \tag{1.5}$$

Si una señal $x(t)$ tiene *energía total* (E) finita y mayor que cero, se clasifica como una *señal de energía*. Estas señales tienen, además, una *potencia promedio* igual a cero.

Si la señal $x(t)$ tiene *potencia promedio* (P) finita y mayor que cero, se clasifica como una *señal de potencia*. Las *señales periódicas* que existen para todos los valores de t, tienen energía infinita, pero en muchos casos tienen una *potencia promedio finita*, lo que las convierte en *señales de potencia*. Las señales limitadas en tiempo, es decir de duración finita, son *señales de energía*.

Cada señal observada en la vida real es una señal de energía. Una señal de potencia, por otro lado, debe tener una duración infinita. Obviamente es imposible generar una verdadera señal de potencia ya que la señal tendría una duración infinita y energía infinita.

1.3 Conceptos de Procesos Aleatorios

Es pertinente tener en cuenta que los sistemas de comunicación en todas sus variantes, tiene un comportamiento no ideal. La transferencia de datos se ve afectada por las imperfecciones de las cuales el canal siempre será víctima. Ya que se ha tenido una concepción concreta de lo que significa una *señal aleatoria,* es oportuno ahondar en la teoría de los procesos aleatorios y lo que ellos conllevan en el análisis de los sistemas de comunicación.

1.3.1 Definición y propiedades

La importancia de hablar de procesos aleatorios, reside en el hecho de que la mayoría de los modelos matemáticos que se sustentan y se representan en la teoría de la información y los sistemas de comunicación son de naturaleza aleatoria, debido a que su comportamiento no cuenta con un patrón estable y se rige por procesos probabilísticos.

Un *proceso aleatorio*, $X\ \lambda, t$, puede ser visto como una función de dos variables, las cuales son *un evento* λ y el *tiempo* t. Para un análisis concreto de un proceso aleatorio, conviene generar N funciones muestra en el dominio del tiempo, $\{X_j(t)\}$. Cada una de las funciones tomadas se puede considerar como la salida de diferentes generadores de ruido; esto para una manipulación más adecuada. Para el evento específico que se ha considerado desde el principio, se tendrá una sola función en el tiempo. Esto se llevará a cabo para cada una de las muestras. Al final, el total de funciones muestra, es denotado como un *conjunto*.

En la imagen 1.8 se muestra gráficamente el concepto de *proceso aleatorio* donde de un espacio muestral se selecciona de manera aleatoria x muestras o procesos λ_x :

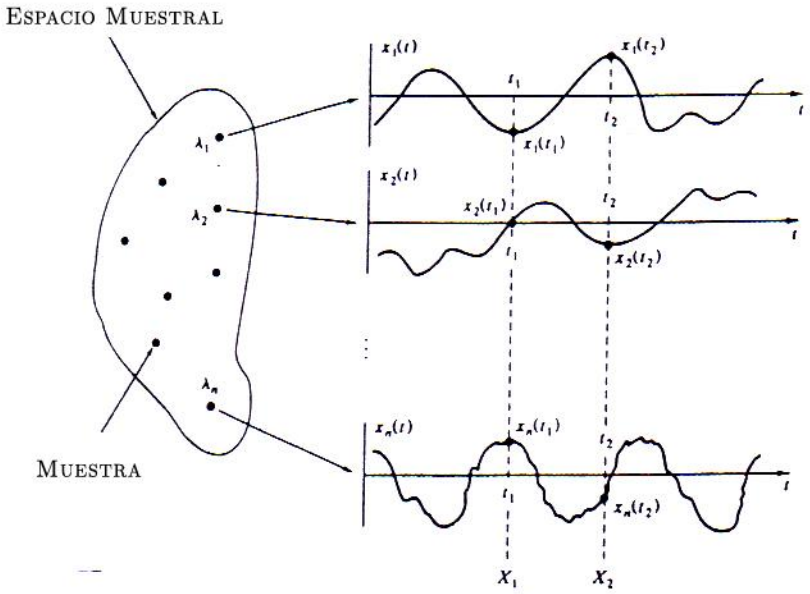


Imagen 1.8 Descripción gráfica de un *proceso aleatorio*

De dichos procesos se generan x funciones en las que se tomarán muestras en el domino del tiempo en y intervalos $x_x(t_y)$ simultáneamente en las funciones tomadas, con el fin de generar un comportamiento probabilístico del *proceso aleatorio*.

1.3.2 Promedios estadísticos

Los *promedios estadísticos* sirven para llevar a cabo una descripción de los procesos aleatorios, con respecto a las variables aleatorias.

Ya que el valor de un proceso aleatorio es indefinido e impredecible, se necesita de promedios estadísticos para generar un pronóstico del comportamiento del

proceso aleatorio. El comportamiento se puede describir con una *función de densidad de probabilidad* (PDF).

Los procesos aleatorios son descritos a menudo usando *promedios estadísticos* como en el caso de la siguiente expresión:

$$E\ X\ t_k\ = \int_{-\infty}^{\infty} x p_{X_k}(x)\, dx = m_X(t_k) \quad \textbf{(1.6)}$$

Donde $X\ t_k$ es la variable aleatoria obtenida de la observación del proceso aleatorio a un determinado tiempo t_k . La función de densidad de probabilidad de $X\ t_k$ (la densidad del conjunto de eventos al tiempo t_k) se designa como $p_{X_k}(x)$.

Dos de los promedios estadísticos que son los de mayor uso en la descripción de de un proceso aleatorio se tiene la *media* $\mu_X\ t$ y la función de autocorrelación $R_{XX}(t_1, t_2)$ definidos como:

$$\mu_X\ t\ = E\{X\ t\ \} \quad \textbf{(1.7)}$$

$$R_{XX}\ t_1, t_2\ = E\{X\ t_1\ X(t_2)\} \quad \textbf{(1.8)}$$

Los valores esperados son tomados con respecto a una función de densidad de probabilidad apropiada.

1.3.3 Estacionariedad y Ergodicidad

La *estacionariedad* de un proceso aleatorio se presenta cuando ninguna de las estadísticas de dicho proceso es afectada por un cambio en el origen de su tiempo. Se dice que un proceso aleatorio es *estacionario en el sentido estricto*, si sus estadísticas no son afectadas por un cambio en el origen temporal, lo cual significa que $X\ t$ y $X(t+\in)$, donde $\in$ es un cambio arbitrario del tiempo, tienen las mismas propiedades estadísticas. Se dice que un proceso aleatorio es *estacionario en un sentido amplio* si dos de sus estadísticas (a través de la autocorrelación) no varían por un cambio en el origen de su tiempo. Entonces, definiendo esta propiedad se dice que:

$$E\ X\ t\ \ = m_x = c \tag{1.9}$$

La mayoría de los resultados usados en la teoría de la comunicación están sostenidos firmemente en la conciencia del manejo de señales de información desconocida y ruido. Y partiendo desde un perfil práctico, un proceso aleatorio no tiene que ser estacionario para todo el tiempo; solo para el intervalo de observación.

Un proceso aleatorio $X(t)$ se considera *ergódico* si el tiempo promedio es el mismo para todas las funciones muestra e igual en la correspondencia con el conjunto promedio. Esto quiere decir que todas las estadísticas de un *proceso ergódico* pueden obtenerse observando solo una función muestra del proceso.

Usualmente no se está interesado en los promedios de un conjunto de un proceso aleatorio; solo en ciertos promedios, como en la media o la autocorrelación.

1.3.4 Autocorrelación y Densidad Espectral de Potencia

Ya que la variación de las características del proceso aleatorio nos brinda una medida la aleatoriedad, la *función de autocorrelación* nos facilita una medida similar para un proceso aleatorio. Para un proceso estacionario, la autocorrelación solo es una función de una diferencia en el tiempo, o dicho en otras palabras un *desplazamiento* del proceso en el tiempo $\tau = t_1 - t_2$. Dicho esto, la función de autocorrelación es definida del siguiente modo:

$$R_{XX}\ \tau\ = E\{X\ t\ X\ t+\tau\ \} \qquad para - \infty < \tau < \infty \tag{1.10}$$

$R_{XX}\ \tau$ Nos da una idea de la respuesta en frecuencia del proceso. Si $R_{XX}\ \tau$ cambia lentamente mientras τ incremente desde cero hasta algún valor, nos indica que en promedio, los valores de muestra de $X(t)$ tomados en el intervalo de tiempo de observación, son muy cercanos. Es aquí donde se espera una representación en el dominio de la frecuencia para $X(t)$ que contenga un dominio sobre bajas frecuencias.

Algunas propiedades de la función de autocorrelación de un proceso estacionario con valores reales son:

$R_X\ \tau\ = R_X\ -\tau$	Simetría en τ con respecto a cero
$R_X\ \tau\ \leq R_X\ 0\ \ para\ todo\ \tau$	El valor máximo se presenta en el origen
$R_X\ \tau\ \leftrightarrow G_X\ f$	La autocorrelación y la densidad espectral de potencia forman un Transformada de Fourier par
$R_X\ 0\ = E\{X^2(t)\}$	El valor en el origen es igual al promedio de potencia de la señal

Tabla 1.1 Propiedades de la función de autocorrelación de un proceso estacionario

La transformada de Fourier de la función de autocorrelación de una señal determinística es la función de *densidad espectral de potencia.* La densidad espectral de potencia de un proceso aleatorio-estacionario se define de manera similar. Si se dice que la función de autocorrelación $R_{XX}\ \tau$ de un proceso aleatorio-estacionario es:

$$\int_{-\infty}^{\infty} R_{XX}\ \tau\ \ dt < \infty \qquad \textbf{(1.11)}$$

Entonces su transformada de Fourier $G_X(f)$ esta dada por:

$$G_X\ f\ = \int_{-\infty}^{\infty} R_{XX}\ \tau\ \exp(-2\pi j f \tau) dt \qquad \textbf{(1.12)}$$

Que es llamada *densidad espectral de potencia* de $X\ t$.

Existe un teorema muy importante llamado *teorema de Wiener-Khintchine* que nos presenta la densidad espectral de un proceso aleatorio en términos de la función de autocorrelación. Dicho teorema estipula que si para todo τ en cualquier intervalo $\mathcal{A}$ de $|\tau|$, la función de autocorrelación de un proceso aleatorio $X(t)$ satisface:

$$\int_{\mathcal{A}} R_X\ t+\tau, t\ dt\ < \infty \tag{1.13}$$

Entonces la densidad espectral de potencia de $X(t)$ es la transformada de Fourier de $R_X\ t+\tau, t$, donde:

$$R_X\ t+\tau, t \quad \stackrel{\text{def}}{=} \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} R_X\ t+\tau, t\ dt \tag{1.14}$$

El concepto de densidad espectral de potencia tiene el mismo significado para señales determinísticas de potencia y para procesos aleatorio-ergódicos. Algunas funciones muestra de procesos ergódicos se usan para determinar parámetros como la localización de determinada señal en el dominio de la frecuencia y anchos de banda, por citar algunas.

Cabe considerar el ejemplo de una señal binaria-aleatoria que consiste en una secuencia de pulsos, ya que es un ejemplo clave para el entendimiento de la función primordial de la interfaz gráfica de este proyecto de titulación, que básicamente se trata de una fuente binaria-aleatoria, cuyos símbolos serán codificados; todo con el fin de concebir gráficamente el proceso de codificación de canal.

Dicha señal binaria-aleatoria con las siguientes propiedades:

1. Cada pulso tiene duración T_b.
2. Los pulsos tienen la misma probabilidad de ser ± 1.
3. Todos los pulsos (amplitud) son estáticamente independientes.
4. Los pulsos no están sincronizados, lo que quiere decir que el tiempo de inicio T del primer pulso tiene la misma probabilidad de estar en cualquier punto entre 0 y T_b.

La figura 1.9 muestra una función miembro de este proceso aleatorio.

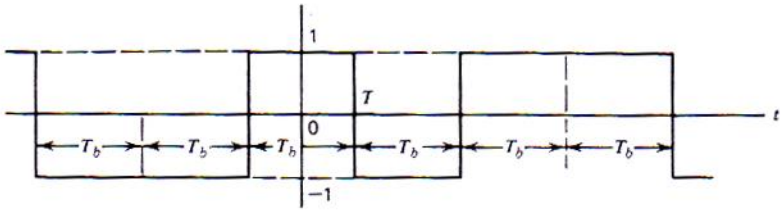


Imagen 1.9 Función miembro de la señal binaria aleatoria $\boldsymbol{X(t)}$

Se puede verificar que $E\ X\ t\ \ = 0$.

Para calcular la autocorrelación para esta función miembro, se tomarán dos muestras arbitrarias del tiempo t, t_1 y t_2 asumiendo que $0 < t_1 < t_2 < T_b$ y $t_1 - t_2\ < T_b$. Después, dependiendo del valor de T, $X(t_1)$ y $X(t_2)$ pueden o no pueden estar en el mismo intervalo de un pulso, como se muestra en la figura 1.10:

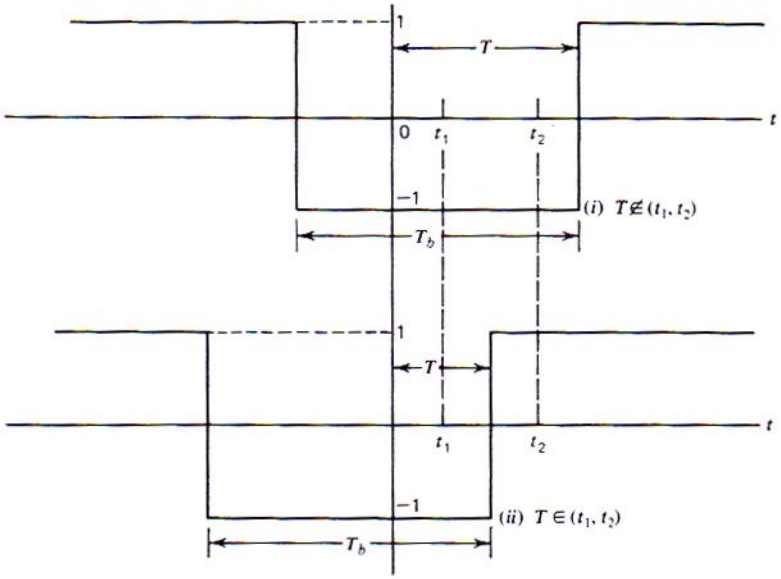


Imagen 1.10 Diagrama en el que se asume que $t_1 < t_2 < T_b$

$A\ t_1, t_2$ será el evento aleatorio del valor de T en el que t_1 y t_2 ocurren en el mismo pulso. Para esto, se dice que:

$$P\ A\ t_1,\ t_2\ = P\ [\ T < t_1\ or\ T > t_2]$$

$$= \frac{t_1}{T_b} + \frac{T_b - t_2}{T_b}$$

$$= 1 - \frac{(t_2 - t_1)}{T_b}$$

$$P\ A\ t_1,\ t_2\ = P\ [\ t_1 <\ T < t_2]$$

$$= \frac{t_2 - t_1}{T_b}$$

Asumiendo que $t_2 < t_1$, entonces:

$$P\ A\ t_1,\ t_2\ \ = 1 - \frac{(t_1 - t_2)}{T_b}$$

En general se tiene:

$$P\ A\ t_1,\ t_2\ \ = 1 - \frac{|t_1 - t_2|}{T_b}$$

Ahora:

$$R_{XX}\ t_1, t_2\ \ = E\{X\ t_1\ X(t_2)\}$$

$$= E\ X\ t_1\ X\ t_2\ \ A\ P\ A\ + E\ X\ t_1\ X\ t_2\ \ A\ P\ A\ * E\ X\ t_1\ X\ t_2\ \ A$$

$$= \frac{1}{2}\ 1\ ^2 + \frac{1}{2}\ -1\ ^2 = 1$$

$$E\ X\ t_1\ X\ t_2\ \ A\ \ = \frac{1}{4}\ 1\ \ 1\ \ + \frac{1}{4}\ -1\ \ -1\ \ + \frac{1}{4}\ 1\ \ -1\ \ + \frac{1}{4}\ -1\ \ 1\ \ = 0$$

Por lo tanto:

$$R_{XX}\ t_1, t_2\ \ = 1 - \frac{|t_1 - t_2|}{T_b}$$

Haciendo que $\tau = \ t_1 - t_2$, se tendrá:

$$R_{XX}\ \tau\ \ = 1 - \frac{|\tau|}{T_b},\ \ \tau\ < T_b$$

Cuando $t_1 - t_2\ > T_b,\ P\ A\ t_1,\ t_2\ \ = 0$ y $R_{XX}\ \tau\ \ = 0$. Así que:

$$R_{XX}\ \tau\ \ = \begin{matrix} 1 - \frac{|\tau|}{T_b}, & para\ \ \tau\ < T_b \\ 0, & para\ \ \tau\ > T_b \end{matrix}$$

Se puede notar que $R_{XX}\ \tau\ $ tendrá el mismo valor para cualquier elección arbitraria de $t_1 = t$ y $t_2 = t + \tau$. Tomando la transformada de Fourier de $R_{XX}\ \tau\ $, se obtendrá la densidad espectral de potencia de la señal binaria-aleatoria $X(t)$:

$$G_X\ f\ = T_b\ \frac{sen(\pi f T_b)}{\pi f T_b}^{\ 2} = \frac{|P(f)|^2}{T_b}$$

Donde $P\ f$ es la transformada de Fourier para un pulso de amplitud unitaria con una duración (ancho) de T_b.

La señal aleatoria-binaria es bastante útil para el modelado de señales usadas en comunicaciones digitales.

Graficando los resultados arrojados en el cálculo de la autocorrelación y densidad espectral de potencia, se tiene como resultado la figura 1.11:

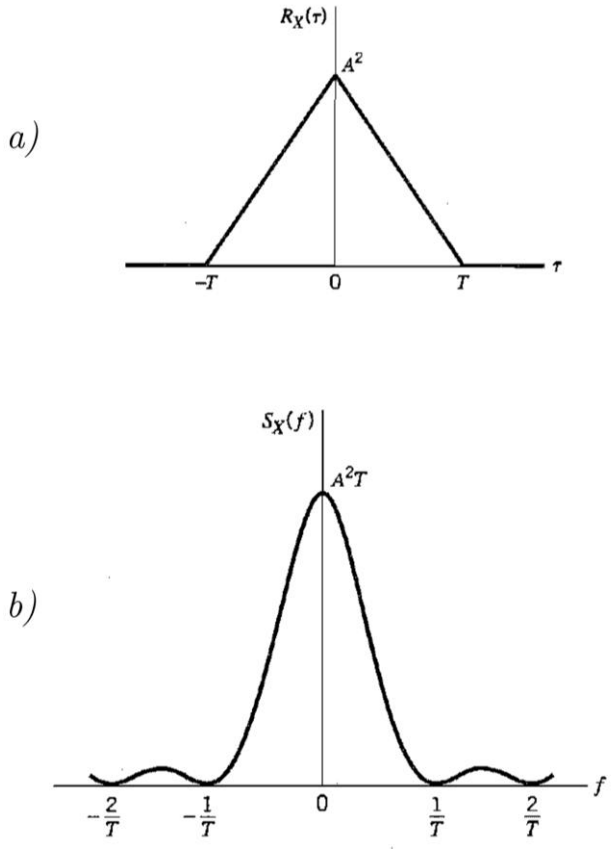


Imagen 1.11 a) Función de autocorrelación de una señal binaria-aleatoria b) Densidad espectral de potencia de una señal binaria-aleatoria.

1.4 Transmisión de señales a través de sistemas lineales invariantes en el tiempo

Ya que a lo largo del capítulo se ha hablado de características de los sistemas de comunicaciones, así como de sus efectos en las señales conjuntas al ruido, es necesario un análisis la respuesta de un sistema lineal al presentársele una señal arbitraria a la entrada. Este análisis se puede llevar a cabo bajo el dominio del tiempo, o bajo el dominio de la frecuencia; esto para lograr un entendimiento pleno del comportamiento del sistema ante determinada señal.

1.4.1 Respuesta Impulso

Se sabe que la respuesta impulso en el dominio del tiempo es la respuesta a un impulso $h(t)$, que a su vez es la respuesta al generar a la entrada un impulso unitario tal como $\delta(t)$:

$$h\ t\ = y(t) \qquad cuando\ x\ t\ = \delta(t) \tag{1.15}$$

La respuesta del sistema ante una entrada arbitraria, es ubicada a través de la convolución entre la entrada y la salida donde * representa la operación de convolución:

$$y\ t\ = x\ t\ * h\ t\ = \int_{-\infty}^{\infty} x\ \tau\ h(t-\tau)\, d\tau \tag{1.16}$$

Hecho esto, entonces se asume que el sistema es causal, lo cual significa que no puede tenerse salida alguna antes de que se aplique una entrada al sistema a determinado origen de tiempo.

Teniendo esto en cuenta, se puede generar una representación alternativa de la salida, simplemente cambiando el límite inferior de la integral en la función de convolución:

$$y\ t\ = x\ t\ * h\ t\ = \int_0^{\infty} x\ \tau\ h(t-\tau)\, d\tau \tag{1.17}$$

Conocida también como integral de superposición o integral de convolución.

1.4.2 Función de Transferencia

Se puede obtener una señal de salida en el dominio de la frecuencia $Y(f)$, aplicando una transformada de Fourier de los dos lados de la ecuación de convolución entre la entrada y la salida.

$$Y\ f\ = X\ f\ H\ f \tag{1.18}$$

$$H(f) = \frac{Y(f)}{X(f)} \tag{1.19}$$

Tendiendo bien en cuenta que $X(f) \neq 0$ para todo f. Entonces se sabe que $H(f) = \Im\{h(t)\}$, que denota la transformada de Fourier de la respuesta impulso de

la función en cuestión, es llamada *Función de Transferencia en Frecuencia* o *Respuesta en Frecuencia*.

Generalmente $H(f)$ se trata de una función compleja lo que nos permite expresarle como:

$$H(f) = |H(f)|e^{j\theta(f)} \quad \textbf{(1.20)}$$

Donde $|H(f)|$ es la magnitud, y $\theta(f)$ es la fase que se define del siguiente modo:

$$\theta\ f\ = \tan^{-1}\frac{Im\{H(f)\}}{Re\{H(f)\}} \quad \textbf{(1.21)}$$

Sabiendo que Im y Re representan las partes real e imaginaria de la función, respectivamente.

La función de transferencia en el dominio de la frecuencia de una función sistema Lineal e Invariante en el tiempo puede ser calculada fácilmente con un generador sinusoidal a la entrada del sistema, y un osciloscopio a la salida. Suponiendo que la entrada $x(t)$ es:

$$x\ t\ = A\cos 2\pi f_0 t \quad \textbf{(1.22)}$$

Entonces, a la salida se tendrá:

$$y\ t\ = A|H(f_0)|\cos[2\pi f_0 t + \theta\ f_0\ ] \quad \textbf{(1.23)}$$

Donde la frecuencia a la entrada, f_0, es medida en el nivel de interés en el sistema. A cada medición hasta llegar a la salida, la amplitud y la fase se puede medir para tener un control de la evolución de la señal de entrada del sistema hasta su etapa final, que es la señal de salida.

1.5 Analógico VS Digital

Los sistemas digitales en los últimos años han sido de los principales temas de discusión técnica y comunicativa como resultado de las nuevas posibilidades de servicios y contenidos que surgen con el paso definitivo del tiempo y de nuevos modelos para modelar sistemas digitales con fines de explotación comercial.

A medida que los sistemas digitales avanzan, los interesados, principalmente económicos, del sector audiovisual tratan de tomar una posición que contribuya a situarse en una estatus privilegiado en cuanto a rendimientos económicos refiere, más allá de las técnicas que el campo digital pueda brindar en el futuro.

Los sistemas digitales comienzan su andadura con la emisión vía satélite, trascendiendo a través del empleo de ondas terrestres y la transmisión vía internet.

Estableciendo las principales diferencias entre los sistemas analógicos y digitales se puede llegar a una concepción de las ventajas y desventajas de cada uno de ellos:

Sistemas Analógicos	Sistemas Digitales
Usan señales de naturaleza continua.	Manejan señales discretas cuyos elementos oscilan en la codificación entre 0 y 1.
Guardan analogía con los fenómenos físicos que producen las señales transmitidas. Se derrocha espectro electromagnético al variar por muy poco que sea la señal transmitida.	No guarda la analogía de los sistemas digitales, pero gracias a técnicas de compresión de información, tales como video y audio, tienen la capacidad de abordar gran número de variables que dependerán de la velocidad a la que se vaya a transmitir.
La interferencia es un problema muy frecuente debido a la presencia de muchas difusoras de información, tales como televisoras y emisoras de radio.	La interferencia es mucho menor que en el caso de sistemas analógicos y dependerá del soporte tomado para su transmisión y más de factores relacionados a la recepción que de factores técnicos propios.

Tabla 1.2 Ventajas y desventajas de los sistemas analógicos y digitales

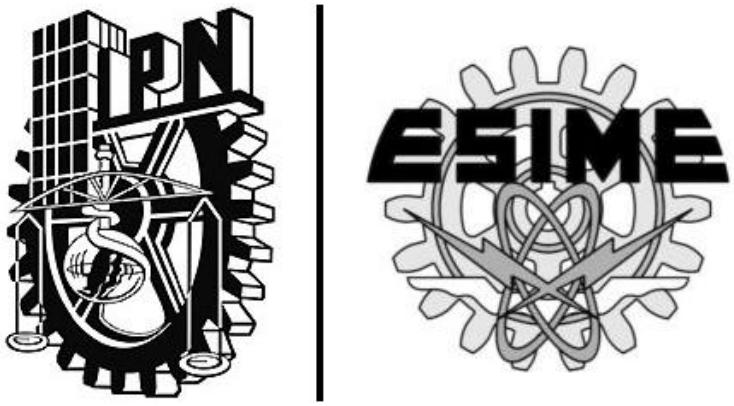
Es necesario tomar muy en cuenta que para la digitalización de una señal se exige la previa existencia analógica de la misma a pesar de tratarse de dos puntos extremos.

El punto de partida de este contraste entre sistemas de tipo analógico y sistemas de tipo digital es el desempeño que cada uno tiene en diferentes contextos. La figura de mérito para el rendimiento de los sistemas de comunicación analógicos es un criterio de fidelidad, tales como la relación señal-ruido, porcentaje de distorsión, o error cuadrático esperado entre la transmisión de una forma de onda recibida.

Del otro lado, se tienen a los sistemas digitales de comunicación, cuyas señales son representadas meramente por dígitos. Dichos dígitos conforman un conjunto finito, que siempre será conocido por el receptor. El valor remarcable de los sistemas digitales de comunicación reside en la probabilidad de detección de error de los dígitos. (Probabilidad de error P_E).

Tomando en cuenta las ventajas y desventajas más relevantes tanto de los sistemas analógicos y digitales cabe destacar que los sistemas analógicos son menos tolerantes al ruido, hacen un buen uso de ancho de banda, y son fácilmente manipulables por su simpleza. Sin embargo, las señales analógicas requieren receptores y transmisores muy particulares, tomando en cuenta el tipo de señal y bandas de trabajo de las mismas.

Concluyendo, se necesita recordar que nos encontramos en un momento en el que las personas cuentan con un mayor abanico de oportunidades de encuentros sociales en espacios y comunidades virtuales que se rigen tanto por sistemas analógicos y digitales, pero sí, es verdad que la transición de los sistemas analógicos al campo digital se ha vuelto un hecho inminente.



CAPÍTULO II | “CANAL DE COMUNICACIÓN”

CAPÍTULO 2 : CANAL DE COMUNICACIÓN

El *canal,* se define como el medio de propagación o trayectoria electromagnética que conecta a un transmisor con un receptor. Físicamente, este canal consiste en cables, fibras ópticas, enlaces de RF (en el caso de sistemas vía radio), guías de onda, atmósfera o el espacio. Hablando de comunicaciones terrestres, el espacio ocupado por el canal es la atmósfera, y limitado por la superficie de la tierra. En el caso de enlaces satelitales, el espacio usado por el canal es el vacío.

En el campo de la Teoría de la Información, el canal se refiere a un *modelo teórico* que contiene determinadas características de error.

Un canal puede ser modelado tratando de calcular el proceso físico que modifica la transmisión de determinada señal. Estadísticamente, un canal de comunicación puede ser modelado postulando que se tiene un alfabeto de entrada, uno de salida, y para cada respectivo par entrada-salida una probabilidad de transición.

El término *canal de comunicación* puede tener diferentes significados y caracterizaciones dependiendo de sus puntos terminales y su funcionalidad. En la figura 2.1 se muestra la caracterización de un canal de comunicación binario, con fines de ejemplificar gráficamente un canal de comunicación:

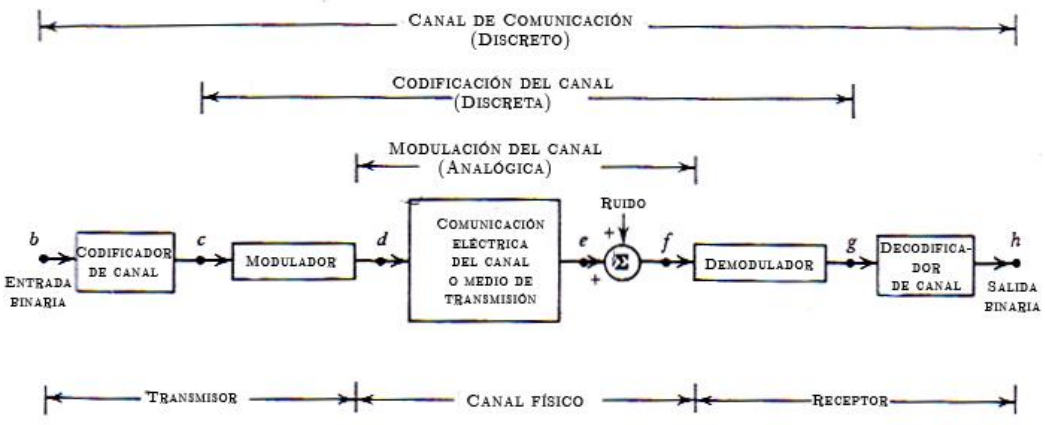


Figura 2.1 Caracterización de un canal de comunicación binario

En la figura 2.1, entre los puntos c y g en el sistema, se tiene una fracción *discreta* del canal, llamado comúnmente *codificación del canal,* que acepta una secuencia de símbolos a su entrada y produce una secuencia de símbolos de salida. Esta parte del canal de comunicación está caracterizada por un conjunto de probabilidades que dependerán de los parámetros de modulación, medios de transmisión, ruido y demodulación. Sin embargo, esta dependencia es transparente para el diseñador del sistema quien debe prestar mayor atención al proceso digital codificador/decodificador.

Entre los puntos d y f de la figura 2.1 se tiene la conexión eléctrica entre el transmisor y el receptor. La entrada y la salida en esta parte del sistema son analógicas. Esta fracción del canal de comunicación se denomina *modulación del canal.* Esta parte del canal está sujeta a diversas contrariedades debidas a variaciones en amplitud y frecuencia dentro de su banda de paso, por ejemplo. Estas contrariedades resultan en una modificación de la señal que se está transmitiendo, lo que a su vez introduce márgenes de error considerables.

Hablar de la *capacidad* de un canal es importante ya que representa la velocidad máxima a la que la información puede ser transferida entre dos puntos de un sistema, con una pequeña probabilidad de error arbitrario; este parámetro denominado *capacidad de canal* se define por el teorema de Shannon-Hartley para determinados tipos de canales continuos, cuyo estudio se abordará posteriormente.

2.1 Imperfecciones en el canal y tipos de errores

Posibles imperfecciones en un canal de comunicación son el *ruido impulsivo, ruido térmico, tiempo de propagación, función de transferencia de canal no lineal, caídas súbitas de la señal, limitaciones en el ancho de banda y reflexiones de señal.*

Muchos de los sistemas modernos de telecomunicación obtienen ventaja de algunas de estas imperfecciones para, finalmente, mejorar la calidad de transmisión del canal. A continuación se presentan los modelos más elementales en el estudio de los sistemas de comunicación.

2.1.1 Canal AWGN y errores aislados

El termino *ruido* se refiere casi siempre a señales eléctricas no deseadas que siempre estarán presentes en los sistemas eléctricos. La presencia de este fenómeno en señales, tiende a "enmascarar" la señal. Del mismo modo, el *ruido* limita al receptor de tal forma que es incapaz de realizar decisiones oportunas en la recepción de símbolos. El *ruido* se genera de diferentes maneras, tanto artificiales (a causa de los procesos del hombre), como naturales, y se define como AWGN ya que se dice que el ruido es aditivo, con compones espectrales de la luz blanca y con comportamientos Gaussianos.

El canal AWGN es un modelo de canal en el que el impedimento de la comunicación es una adición lineal en el ancho de banda con una densidad espectral constante y una distribución gaussiana en amplitud. El modelo no es tomado en cuenta para cuestiones de desvanecimiento, interferencia o dispersión.

Las características principales de un canal AWGN son:

- Su espectro comprende todas las frecuencias.
- Su amplitud sigue una distribución Gaussiana-Aleatoria.
- El ruido AWGN se adhiere a la señal de información.

En los canales AWGN se cuenta con factores potenciales de ruido los cuales pueden ser propios o ajenos al sistema. Al hablar de los factores externos se

puede referir al ruido térmico que se genera por la agitación térmica de los electrones que se encuentran presentes en los conductores o líneas de transmisión. Al referirnos a los factores internos, cabe mencionar el ruido que capta la antena receptora por el ruido atmosférico o ruido galáctico. El ruido atmosférico se considera como la energía eléctrica que ocurre por naturaleza y que se origina dentro de la atmósfera de la tierra; además es comúnmente llamado electricidad estática. La magnitud de los impulsos generados por la estática se ha dicho que es inversamente proporcional a la frecuencia de trabajo, y por consecuencia, se dice que a frecuencias superiores de 30 MHz, el ruido atmosférico es insignificante. A demás, a estas frecuencias, la propagación está limitada al manejo de líneas de vista. A pesar de todo esto, tanto en los factores internos como externos del sistema, el modelado se considera como ruido térmico.

Existe una fuente natural muy conocida de ruido llamada *ruido térmico* o *ruido de Johnson*, que es imposible de eliminar. El *ruido térmico* se genera por el movimiento térmico de los electrones en todas sus componentes de disipación; dicho ruido generado por el movimiento es imposible de corregir, ya que es un proceso que se presenta en cada transmisión eléctrica a través de cualquier cable, resistencia, o medio de transmisión, por lo que se dice que los mismos electrones responsables de la conducción eléctrica, son también responsables por el ruido térmico.

Se puede describir el ruido térmico como un proceso aleatorio Gaussiano con media cero. Un proceso Gaussiano $n(t)$ es una función aleatoria cuyo valor n en cualquier tiempo arbitrario t está estadísticamente caracterizado por la función de densidad de probabilidad Gaussiana:

$$p\ n\ = \frac{1}{\sigma\ \sqrt{2\pi}} exp\ -\frac{1}{2}\ \frac{n}{\sigma}\ ^2 \qquad \textbf{(2.1)}$$

Donde σ^2 es la varianza de n. La *función de densidad de probabilidad Gaussiana estandarizada* o *normalizada* de un proceso con media cero se obtiene asumiendo que $\sigma = 1$. Dicha función se muestra en la figura 2.2:

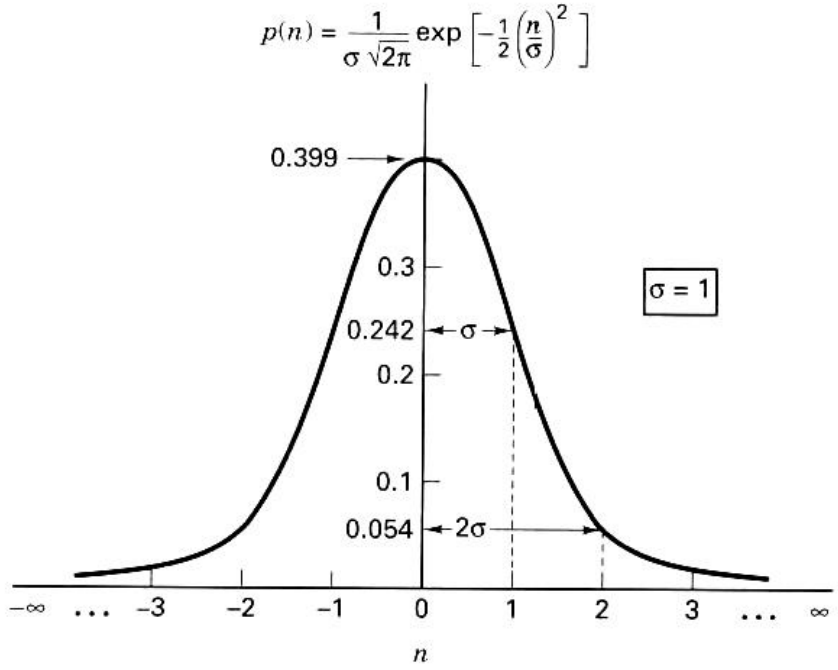


Figura 2.2 Función de densidad de probabilidad Gaussiana estandarizada ($\boldsymbol{\sigma = 1}$)

A menudo también se representará una señal aleatoria como la suma de una variable aleatoria de ruido Gaussiano con una señal de dc, lo que es:

$$z = a + n \tag{2.2}$$

Donde z es la señal aleatoria, a es la componente de dc y n es la variable aleatoria de ruido Gaussiano. Entonces, la función de densidad de probabilidad $p(z)$ se expresa como:

$$p\ z\ = \frac{1}{\sigma\ \overline{2\pi}} exp\ \ -\frac{1}{2}\ \ \frac{z-a}{\sigma}\ \ ^2 \tag{2.3}$$

Donde, como anteriormente se mencionó, σ^2 es la varianza de n. La distribución Gaussiana es a menudo usada como el modelo de ruido del sistema debido al *teorema de límite central*, que establece que bajo condiciones muy generales la distribución de probabilidad de la suma de j variables aleatorias estadísticamente independientes se aproximan a la distribución Gaussiana como $j \to \infty$, sin importar cuales podrían ser las funciones de distribución individuales. Por lo tanto, aunque mecanismos individuales de ruido puedan tener distintas distribuciones Gaussianas, el conjunto de varios de estos mecanismos tenderán hacia una sola distribución Gaussiana.

La primera característica espectral del ruido térmico es que su densidad espectral de potencia *es la misma* para todas las frecuencias de interés de la mayoría de los sistemas de comunicaciones; en otras palabras, una fuente de ruido térmico emana una cantidad equitativa de ruido por unidad de ancho de banda a todas las frecuencias, desde niveles de dc hasta aproximadamente $10^{12} Hz$. Por lo tanto, un modelo simple de ruido térmico asume que su densidad espectral de potencia $G_n(f)$ es plana para todas las frecuencias, como se muestra en la figura 1.3, cuya denotación es:

$$G_n\ f\ = \frac{N_0}{2}\ watts/Hz \qquad \textbf{(2.4)}$$

Donde 2 se agrega para indicar que $G_n(f)$ es una densidad espectral de potencia bilateral. Cuando el ruido tiene una densidad espectral uniforme, se referirá a él como *ruido blanco*. El adjetivo *blanco* se usa en el mismo sentido que se usa para referirnos a la luz blanca, que contiene cantidades equitativas de todas las frecuencias dentro de la banda visible de radiación electromagnética.

La función de autocorrelación del ruido blanco está dada por la transformada inversa de Fourier de la densidad espectral de potencia del ruido, denotada como:

$$R_n\ \tau\ = \Im^{-1}\ G_n\ f\ = \frac{N_0}{2}\delta(\tau) \quad \textbf{(2.5)}$$

Así, la autocorrelación del ruido blanco es una función delta ponderada por el factor $N_0/2$ ocurriendo en $\tau = 0$, como se observa en la figura 1.3. Cabe destacar que $R_n\ \tau$ vale cero para $\tau \neq 0$ lo que significa que cualquier par de muestras diferentes de ruido blanco, sin importar qué tan cerca en tiempo sean tomadas, carecen de correlación.

La potencia promedio P_n del ruido blanco es infinita ya que su ancho de banda es infinito.

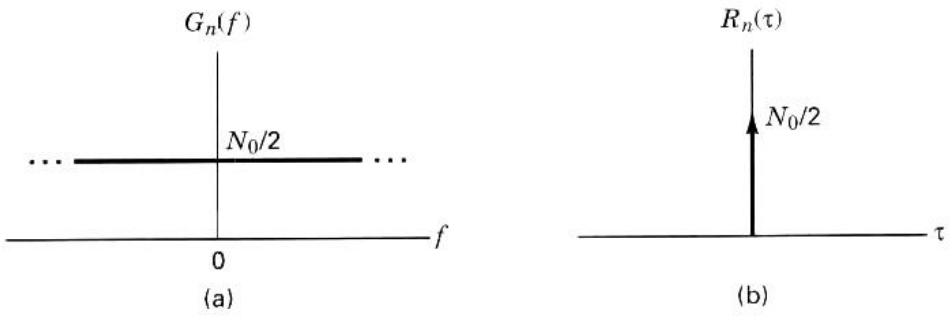


Figura 2.3 a) Densidad espectral de potencia del ruido blanco b) Función de autocorrelación del ruido blanco

A pesar de que el ruido blanco se trata de una abstracción útil, ningún proceso del ruido puede ser verdaderamente blanco; sin embargo, el ruido encontrado en muchos sistemas reales puede ser considerado *aproximadamente blanco*. Solo se puede observar el ruido una vez que ha pasado a través de un sistema real, cuyo ancho de banda será finito. Así, siempre que el ancho de banda del ruido se

apreciado mayor que el del sistema, el ruido puede ser considerado con un ancho de banda infinito.

Ya que el ruido térmico es un proceso Gaussiano y sus muestras carecen de correlación, las muestras de ruido son consideradas *independientes.* Por lo tanto, el efecto en el proceso de detección de un canal AWGN (additive White Gaussian noise) es que el ruido afecta cada símbolo transmitido *independientemente*. Éste canal es considerado un *canal sin memoria*. El término *aditivo* infiere que el ruido simplemente es superpuesto o sumado a la señal (no hay mecanismos multiplicativos en este proceso).

Ya que el ruido térmico está presente en todos los sistemas de comunicaciones y es la fuente de ruido prominente para todos los sistemas, las características del ruido térmico (aditivo, blanco y Gaussiano) es utilizado para el modelado de ruido en los sistemas de comunicaciones. Ya que el ruido Gaussiano con media cero es caracterizado completamente por su varianza, éste modelo es particularmente simple para el uso en la detección de señales y en el diseño de receptores óptimos.

Debido a que las muestras de ruido en un canal AWGN no tienen correlación por muy cercanas que estén dichas muestras en el tiempo, los errores en un canal AWGN son conocidos como *errores aislados* o de un solo bit. En la figura 2.4 se representa gráficamente un error aislado.

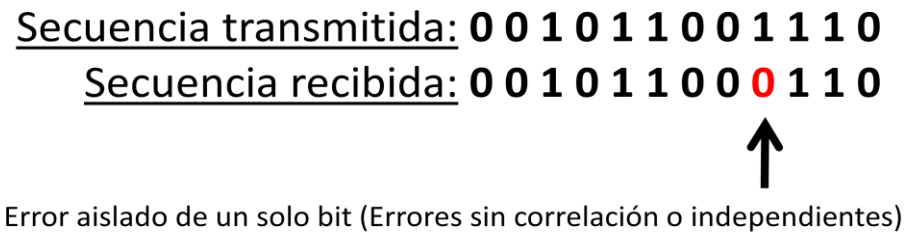


Figura 2.4 Ejemplo de error aislado

Por ejemplo, cuando el emisor envía una trama, y un bit de la trama que correspondía a un 1 binario es recibido como un 0 binario, se presenta el error aislado; estos bits erróneos son independientes de los demás.

Los errores aislados son más fáciles que detectar y corregir que los errores por ráfagas; éstos últimos son más frecuentes en la transmisión de datos.

2.1.2 Canal con desvanecimiento y errores por ráfagas

Los canales analógicos están sujetos a diversas variaciones e impedimentos sujetos a su naturaleza. Algunos son debido a la amplitud o la respuesta en frecuencia del canal en determinada banda de paso. Por si mismo, el canal tiene la capacidad de modificar la señal de entrada en el sistema de transmisión afectándola estadísticamente debido a varios tipos de ruido aditivo y multiplicativo, y a *desvanecimientos* que es básicamente una atenuación aleatoria que cambia a lo largo del medio de transmisión. En otras palabras, se puede decir que el *desvanecimiento* es la desviación de la atenuación que afecta a una señal a través de determinados medios de propagación. El *desvanecimiento* se presenta generalmente en canales inalámbricos y son las variaciones que experimenta la señal emitida conforma ésta se propaga desde el transmisor al receptor. Es un fenómeno que puede variar tiempo, la ubicación geográfica y la frecuencia de trabajo, por lo que generalmente se modela como un *proceso aleatorio* (Rayleigh o Rice).

El *desvanecimiento* tiene como principales categorías las siguientes:

1. Obstáculos en la trayectoria de propagación (*propagación sin línea de vista*, NLOS: *Non-Line of Sigth).*
2. Propagación por trayectorias múltiples (*Multipath*).

3. Efectos atmosféricos (*f>10GHz. Entornos con desvanecimiento por nubes, lluvia, nieve, vapor de agua, etc.*)

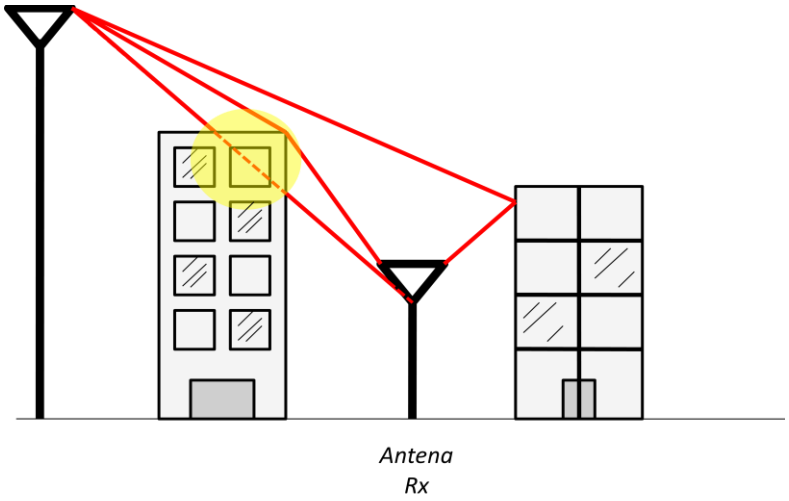


Figura 2.5 Ejemplo de desvanecimiento producido por trayectorias múltiples y obstáculos en la trayectoria de propagación.

En la figura 2.5, se representa gráficamente la ocurrencia de las trayectorias múltiples y los obstáculos presentes en todos los sistemas de propagación. Los ejemplos más conocidos y estudiados de canales con desvanecimiento son el *canal con desvanecimiento Rayleigh y canal con desvanecimiento Rice.*

El *canal con desvanecimiento Rayleigh* es un modelo estático para efecto de un entorno de propagación de una radio señal, como la usada en dispositivos inalámbricos.

El modelo de desvanecimiento de Rayleigh es aplicable a entornos con diversos objetos que generan dispersión en la señal antes de que esa llegue al receptor.

El *canal con desvanecimiento Rice* es un modelo estocástico para las anomalías de propagación radial causadas por la misma señal, es decir, que la señal llega al receptor por diferentes trayectorias lo cual causa interferencia por trayectorias

múltiples; al menos una de estas trayectorias ha sufrido una modificación (alargamiento o acortamiento). Se produce cuando alguna de las líneas de vista tiene mayor presencia que las demás.

Se sabe que las características de transmisión de los canales con desvanecimiento varían con el tiempo. Ejemplos de estos canales son los canales con dispersión topográfica y los canales que usan la ionosfera para reflexiones de radio para lograr largas distancias de comunicación. Las variaciones en tiempo de las propiedades del canal surgen debido a cambios semi-periódicos y aleatorios en la propagación del medio. Las propiedades de reflexión de la ionosfera, por ejemplo, están relacionadas con las condiciones meteorológicas que cambian según la temporada, el día, e incluso con el paso de las horas, tanto como el clima. Periodos de tormenta repentina también suelen ocurrir. Por lo tanto, la función de transferencia de canal efectivo también varía semi-periódica y aleatoriamente, causando atenuación aleatoria a la señal. Una manera de reducir los efectos del desvanecimiento es usar controles automáticos de ganancia, aunque también suprimirá variaciones bajas de la señal original.

El desvanecimiento puede depender fuertemente en frecuencia en aquellos intervalos en los que las componentes de frecuencia son afectadas desigualmente. También, la propagación por trayectorias múltiples puede causar desvanecimientos selectivos en frecuencia.

Una *ráfaga de errores* comienza y termina con un bit erróneo, pudiendo estar o no alterados los bits intermedios. Haciendo una definición, una *ráfaga de errores* es el número de bits entre dos bits erróneos previamente identificados.

Al determinar la longitud de una ráfaga de errores, el último bit erróneo de una ráfaga y el primer bit erróneo de la siguiente ráfaga deben estar separados por x o más bits erróneos, donde x es la longitud de ráfaga de errores, como se muestra en la figura 2.6:

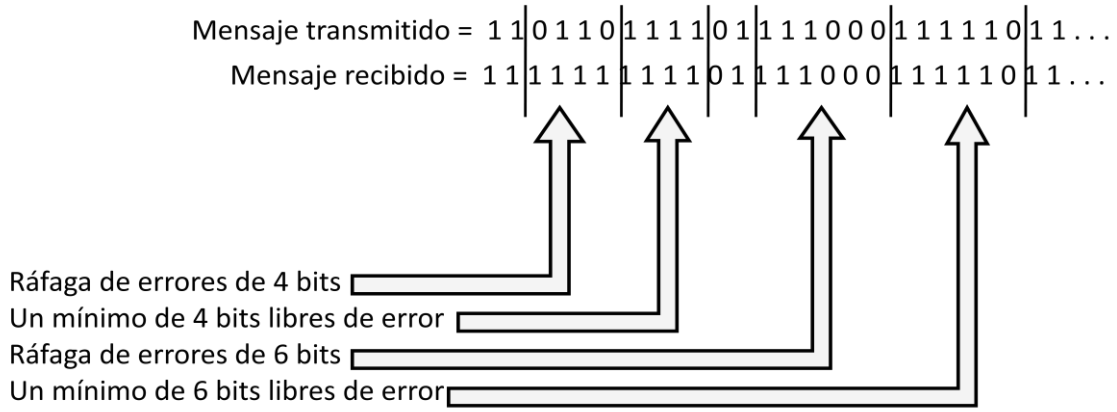


Figura 2.6 Ejemplo de ráfaga de errores

Al presentarse errores por ráfagas se necesitan métodos más rigurosos que los ocupados en la detección de errores aislados. Un modelo que ha sido moderadamente exitoso en la caracterización de los errores por ráfagas ha sido el modelo de Gilbert. En dicho modelo, el canal se supone discreto y sin memoria, donde la probabilidad de error es un parámetro variante en el tiempo. Los cambios en la probabilidad de error se modelan por medio de un proceso de Markoff mostrado en la figura 2.7:

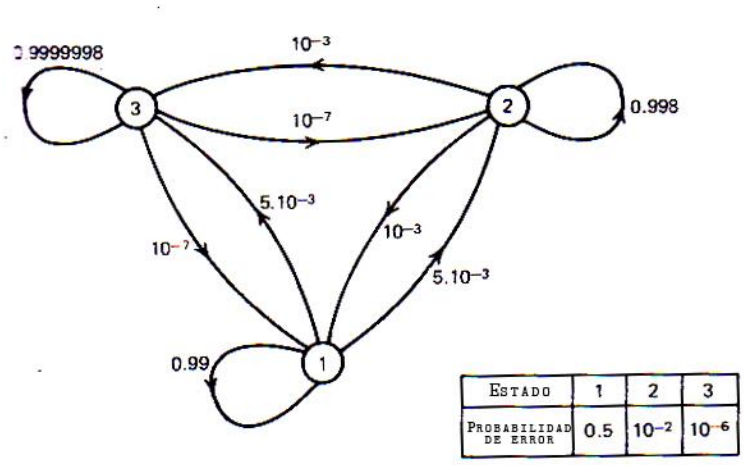


Figura 2.7 Modelo de Gilbert de tres estados para un canal de comunicación

El mecanismo generador de error en el canal ocupa uno de tres estados, y la transición de un estado a otro es modelada como un proceso discreto y estacionario de Markoff. Cuando el canal está en estado 2, por ejemplo, la probabilidad de bit erróneo durante un intervalo de bit es 10^{-2} y el canal queda en ese estado durante el intervalo de bit siguiente con una probabilidad de 0.998. Sin embargo, el canal quizá se traslade al estado 1 donde la probabilidad de bit erróneo es de 0.5. Ya que el sistema se queda en el estado con probabilidad 0.99, los errores tienden a ocurrir en ráfagas (grupos). El estado 3 representa una tasa de bit erróneo baja, y los errores en este estado son causados por ruido Gaussiano. Los errores en ráfagas ocurren rara vez en este estado. Algunas otras características del canal como lo son el tiempo medio entre los errores por ráfagas, y duración promedio de los errores por ráfagas, pueden ser calculados a través de este modelo.

2.2 Clasificación de canales de comunicación

Como se sabe, el canal es definido como el único camino por el cual se va a transmitir una señal específica, ya sea en una sola dirección o en ambas direcciones.

Con la gran variedad de posibilidades en el modelado y estudio del canal, es difícil definir un parámetro de carácter general aplicable a cualquiera de los modos de transmisión conocidos.

En efecto, el modo de transmisión utilizado en el sistema determina, en gran medida, las perturbaciones que experimentará la señal transmitida a través del canal.

2.2.1 Canales discretos sin memoria

En el modelo del *canal discreto*, la atención se concentra en los símbolos de entrada, los símbolos de salida y el conjunto de probabilidades condicionales relativas al mismo modelo de canal.

Un factor importante, es la transición de probabilidades que contiene el canal y puede ser conocida sabiendo las características del canal y la estructura del receptor.

Se sabe que un *canal discreto sin memoria* es un modelo estadístico que tiene como entrada un *alfabeto finito de símbolos* (*E.g.* $x_1, x_2, x_3, \dots, x_m$) y genera como respuesta *otro alfabeto finito de símbolos* (*E.g.* $y_1, y_2, y_3, \dots, y_n$).

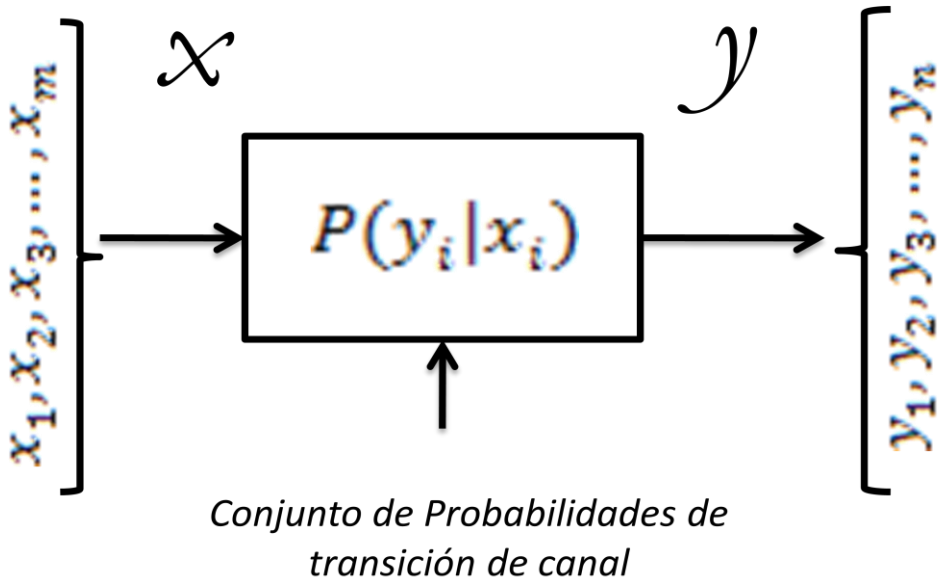


Figura 2.8 **Canal discreto sin memoria**

En un *canal discreto sin memoria*, un símbolo de salida actual dependerá solo de un símbolo de entrada actual y no de entradas previas.

Un canal discreto sin memoria (DMC) es un modelo eficiente de las comunicaciones entre el modulador de forma de onda y la salida del demodulador. Un canal DMC incluye los efectos de la distorsión de la señal. Usando la salida de un canal DMC los decodificadores de canal intentan la reconstrucción de la secuencia origina de salida de la fuente.

El buen diseño de un sistema codificador/decodificador será capaz de corregir algunos de los errores de transmisión.

El ejemplo más común de modelado de canal es el canal binario simétrico sin memoria. En este canal se asume que tanto como la entrada y la salida del canal son binarios (0 y 1) y subsecuentemente los símbolos de salida dependen solamente de los símbolos de entrada.

En la figura 2.9 una descripción gráfica del proceso en un canal binario simétrico sin memoria:

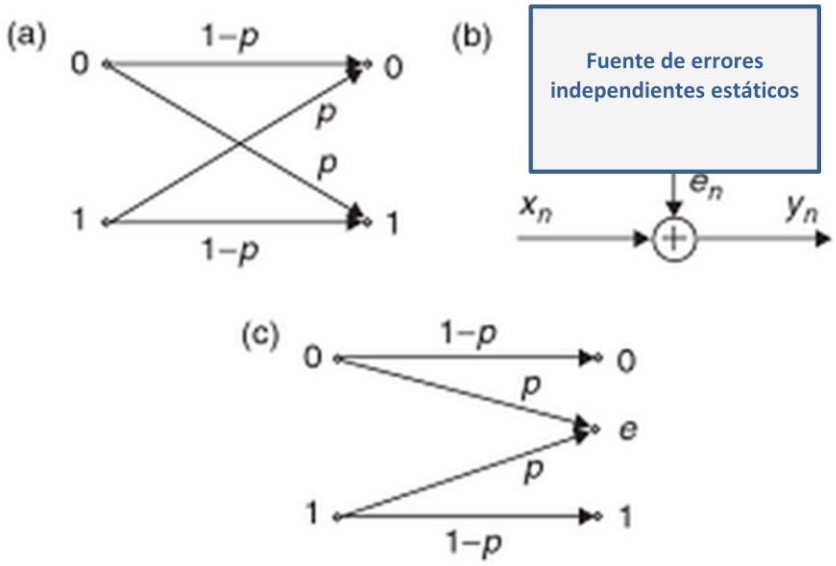


Figura 2.9 **Modelos de canales binarios sin memoria: a) Canal simétrico binario, b) Variante de canal simétrico binario, c) Canal BEC (Binary Erasure Channel)**

En los modelos presentados en la imagen 2.9, la ocurrencia de error es considerada como la adición del símbolo de entrada con un error $e_n = 1$ o $e_n = 0$.

2.2.2 Canales discretos con memoria

Ya que se habla de los canales con memoria donde la ocurrencia de error en un intervalo de símbolo particular no afecta la ocurrencia de error en símbolos posteriores, cabe destacar que en muchos canales, los errores no ocurren como eventos independientes. Estos canales se conocen como *canales discretos con memoria*. En ellos, el ruido impulsivo predomina sobre el ruido Gaussiano y los errores ocurren en largas ráfagas infrecuentes. Debido al fenómeno físico tan complejo presentado en este proceso, la caracterización detallada de los canales con memoria es muy difícil. Canales telefónicos que son afectados por conmutaciones transitorias y enlaces de radio de microondas que son sometidos a desvanecimiento son ejemplos de canales con memoria.

2.2.3 Canales continuos

La fracción comprendida entre los puntos *d* y *f* del canal de comunicación presentado en la figura 2.1, llamado *proceso de modulación de canal*, es analógica o de naturaleza continua. En esa porción de canal, las señales de salidas son funciones continuas en el tiempo, y la función del canal es producir a su salida la forma de onda eléctrica presentada a su entrada. Un canal real logra esto solo de manera aproximada. Primero, el canal modifica la forma de onda de una manera determinística, y su efecto puede ser modelado adecuadamente tratando el canal como un sistema lineal. El canal también modifica la en señal a la entrada de una

manera aleatoria, debido a ruido aditivo y multiplicativo. Generalmente los modelos del ruido se consideran aditivos, ya que los modelos multiplicativos de ruido tienen menor ocurrencia en la realidad. El ruido aditivo puede ser Gaussiano o impulsivo en su naturaleza. El ruido Gaussiano incluye ruido térmico y de disparo generado por el equipo y la radiación captada por la antena receptora. De acuerdo con el teorema de límite central, el ruido que resulta de los efectos sumados de varias fuentes suele tener una distribución Gaussiana. Debido a esta omnipresencia, el ruido Gaussiano es el más utilizado para la caracterización de la fracción análoga del canal de comunicación. Las técnicas de modulación y demodulación se diseñaron con el objetivo primario reducir los efectos del ruido Gaussiano.

El ruido impulsivo también es encontrado en el canal. El ruido impulsivo es caracterizado por largos intervalos seguidos de ráfagas de pulsos de ruido de gran amplitud. Este tipo de ruido es debido a conmutaciones transitorias, descargas, y golpes accidentales durante los trabajos de mantenimiento y así sucesivamente. La caracterización del ruido impulsivo es mucho más difícil que la caracterización del ruido Gaussiano. A demás, las técnicas analógicas de modulación no son tan adecuadas como los métodos digitales de codificación al momento de enfrentarse al fenómeno del ruido impulsivo. Por esas razones, los efectos del ruido impulsivo son a menudo incluidos la fracción discreta del canal, y solo el ruido Gaussiano es incluido en la fracción analógica del canal.

La porción analógica del canal puede ser modelada como se describe en la figura 2.10:

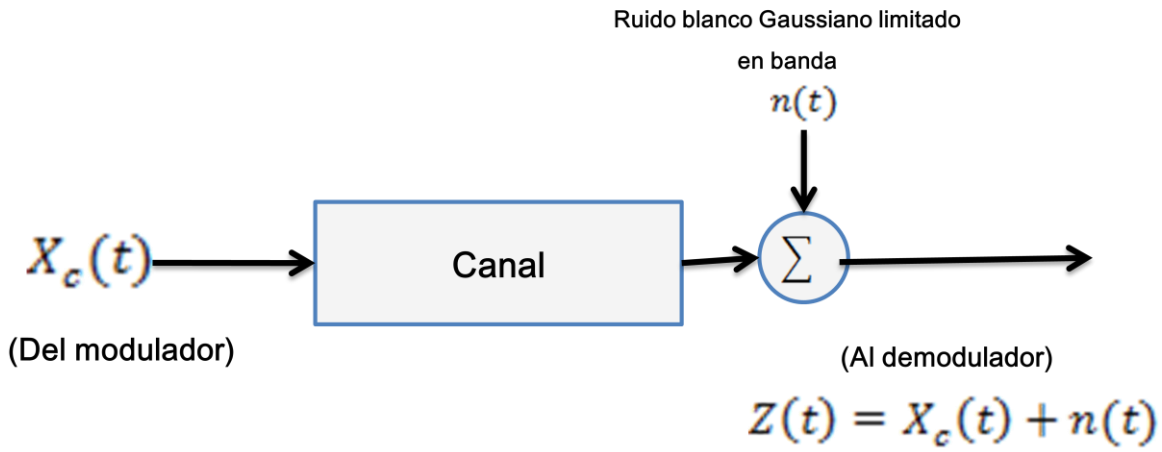


Imagen 2.10 Fracción analógica del canal de comunicaciones.

La entrada analógica al canal es un proceso aleatorio $X_c(t)$ que consiste en una colección de todas las señales producidas por el modulador. El ruido aditivo a la salida del canal se considera ruido blanco Gaussiano limitado en banda con media cero $n\ t$. La capacidad de esta porción del canal se encontrará maximizando la tasa de transmisión de información con respecto a la distribución de $X_c(t)$.

El concepto de canal continuo es de gran utilidad para el entendimiento del uso de canales reales.

2.3 Capacidad de canal

La capacidad de un canal con ruido (discreto y sin memoria) es definida como la tasa máxima de transmisión de información posible sobre un canal. La tasa máxima de transmisión ocurre cuando la fuente se “armoniza” con el canal. Se definirá a capacidad de canal C como:

$$C = \triangleq max_{P(X)}\{D_t\}$$

$$= max_{P(X)}\ H\ X\ - H\ X\ Y\ \ r_s$$

Donde r_s es la tasa de símbolo (símbolos/seg) y D_t es la tasa promedio de transmisión,

La capacidad de un canal C es el número de información de bits por segundo que puede transmitir teóricamente hablando, con una tasa de error arbitraria baja sobre el mismo canal.

La capacidad es la razón de la señal entre el ruido, potencia y ancho de banda. Por ejemplo, si a la salida del codificador de la fuente se tiene una secuencia binaria cuya tasa de bits es R_m por segundo, que a su vez es menos que la capacidad de canal C, la tasa de error en la transmisión puede reducirse a cualquier nivel deseado usando técnicas de corrección de error progresiva (FEC por las siglas en inglés de *forward error correction)* sin necesidad de incrementar la potencia de transmisión. El precio que se paga por reducir la probabilidad de error es el complejo incremento del ancho de banda.

Otra estrategia para reducir la probabilidad de error son los *códigos de bloques* en donde la salida del codificador de la fuente se recauda por T segundos y se procesa como un bloque.

A continuación, se estipulan los fundamentos de la capacidad de canal y los teoremas que rigen la capacidad de un canal de comunicaciones.

2.3.1 Teorema de Shannon-Hartley

Es importante hablar de este teorema, debido a que su postura representa algunos comportamientos para los canales de ruido.

Shannon postuló que la capacidad de un sistema C , de un canal perturbado por ruido AWGN, es una función del promedio de potencia de la señal recibida, S, el

promedio de potencia de ruido, N, y el ancho de banda, W. La relación de capacidad, que en sí es el teorema, puede ser representado como:

$$C = W \log_2(1 + \frac{S}{N}) \quad \textbf{(2.6)}$$

Donde el ancho de banda W está dado en hertz, y el logaritmo es implementado en base 2, ya que la capacidad es dada en bits/s . Teóricamente es posible transmitir información a través de un canal a cualquier velocidad R, donde $R \leq C$, con una probabilidad de error arbitrario relativamente pequeña, usando un esquema complejo de codificación. En el caso de $R > C$, es imposible encontrar un código que pueda satisfacer la necesidad de tener una probabilidad de error arbitrario relativamente pequeña. En esta expresión matemática se demuestra que los valores de S, W y N establecen un límite en la tasa de transición, no en la probabilidad de error.

Partiendo de la ecuación 2.6, Shannon generó una gráfica en la que exhibe un límite del rendimiento alcanzable en un sistema práctico. En la figura 2.11, se muestra de canal normalizada, C/W en bits/s/Hz en función con la relación señal-ruido del canal (SNR).

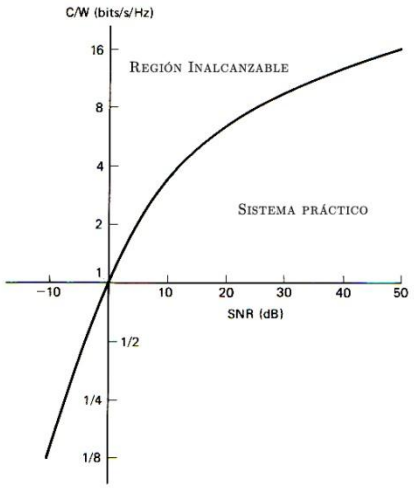


Figura 2.11 Capacidad de canal normalizada VS relación señal-ruido del canal

En la figura 2.12 se muestra el ancho de banda normalizado del canal, W/C en Hz/bits/s en función con la SNR del canal.

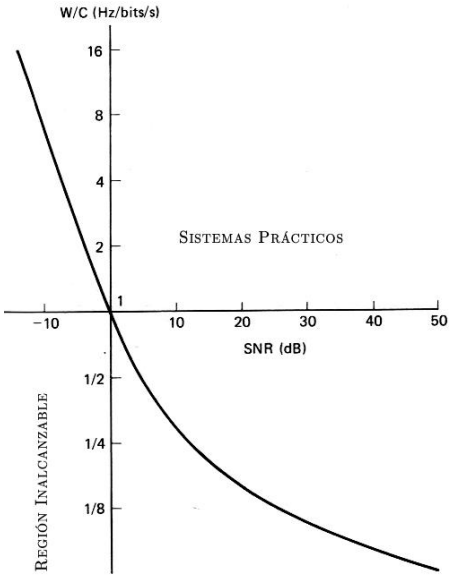


Figura 2.12 Ancho de banda normalizado del canal VS relación señal-ruido del canal

La figura 2.12 se usa algunas veces para ilustrar la compensación de potencia de ancho de banda adjunta en un canal ideal. Sin embargo, no es una compensación pura porque la potencia de ruido detectada es proporcional al ancho de banda:

$$N = N_0 W \tag{2.7}$$

Sustituyendo la ecuación 2.7 en la ecuación 2.6, reacomodando términos se tiene que:

$$C = W \log_2(1 + \frac{S}{N_0 W}) \tag{2.8}$$

2.3.2 Límite de Shannon

Existe un valor límite que puede alcanzar E_b/N_0 por debajo del cual no puede existir comunicación libre de errores a cualquier velocidad de transmisión de información. Usando la identidad:

$$\lim_{x \to 0}(1 + x)^{1/x} = e \tag{2.9}$$

Se puede calcular el valor límite de E_0/N_0 que se ha hablado:

$$x = \frac{E_b}{N_0} \quad \frac{C}{W} \tag{2.8}$$

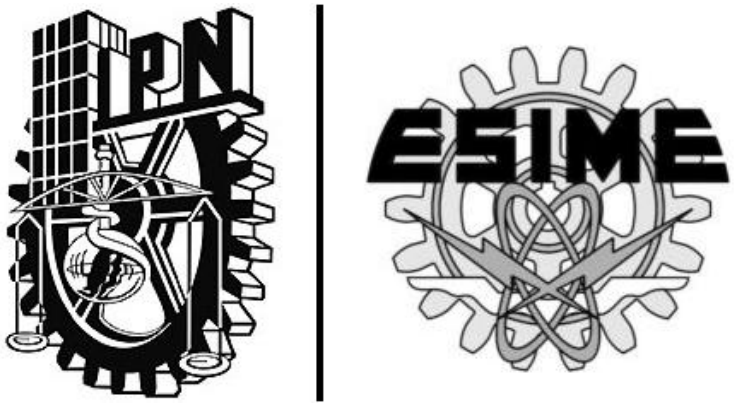
En el límite donde $\frac{C}{W} \to 0$, se tiene que:

$$\frac{E_b}{N_0} = \frac{1}{\log_2 e} = 0.693 \quad \textbf{(2.9)}$$

O en decibeles:

$$\frac{E_b}{N_0} = -1.6\ dB \quad \textbf{(2.10)}$$

Este valor de E_0/N_0 es conocido como *límite de Shannon*. En la práctica no es posible encontrar el límite de Shannon ya que como los valores que componen la expresión matemática incrementan sin límite, el requerimiento de ancho de banda y la complejidad de implementación incrementan sin límite. El trabajo de Shannon otorga una prueba teórica de la necesidad de la reducción de E_b/N_0 desde niveles de esquemas de modulación no codificada binaria hasta niveles cercanos al límite de la curva (figura 2.12). El diseño de sistemas óptimos puede ser mejor descritos como la búsqueda de compensaciones racionales entre las diversas limitaciones. La modulación y la compensación de codificación, lo que significa la selección técnicas de modulación y codificación que hagan mejor uso de la potencia de transmisión y ancho de banda del canal, son importantes desde la existencia iniciativas fuertes para la reducción de costo de la generación de potencia y la conservación del espectro radioeléctrico.



CAPÍTULO III |

“Codificación de Canal”

CAPÍTULO 3: CODIFICACIÓN DE CANAL

Básicamente, al llevar a cabo la codificación de canal, lo que se busca es la detección y corrección de errores que se presentan por la propia naturaleza del canal de transmisión o medios de propagación como una inminente consecuencia del ruido y la distorsión introducidos en el mismo.

3.1. Definición y objetivos

En los sistemas de comunicación habituales existen dos factores principales en el deterioro de una señal recibida, los cuales son el ruido del canal, y el ruido de cuantificación, siendo este último una consecuencia del proceso de codificación introducido en el transmisor, que a su vez es transportado por todo el sistema hasta el receptor. El ruido de canal genera errores de transmisión. Se habla de una fidelidad de transmisión de información que es medida en los términos de una *tasa de errores* o *probabilidad de error*.

Una gran pero limitada ventaja de los sistemas digitales sobre los sistemas analógicos es la capacidad para reconocer y corregir errores causados por factores propios del sistema.

En los sistemas analógicos se presenta una degradación de señal suave, lo que quiere decir que la relación señal a ruido disminuye lentamente con la distancia y la señal, y puede ser recibida en condiciones aceptables de su rango de distancia aun con dicho ruido. Por el contrario, en sistemas digitales la señal deja de ser recibida cuando la tasa de error aumenta por encima de determinado valor que el receptor es capaz de manejar. En este caso, la degradación se considera brusca, lo que quiere decir que de un estado de recepción óptimo se pasa a una recepción

nula, abruptamente. Debido a esos inesperados cambios de estado en los sistemas digitales, debe protegerse aunque parte de ella se pierda en la transmisión, ya que esta parte destruida puede ser recuperada o reconstruida en el receptor.

Hay que destacar que la codificación del canal no tiene que ver con la codificación de la fuente, ya que el codificador de canal tiene como entrada una señal digital que proviene de determinada fuente.

El principal objetivo de la *codificación de canal* es la protección de información digital a transmitir o almacenar, asegurándole inmunidad frente al ruido, de modo que esta información no se vea alterada a su llegada al receptor muy independiente de la calidad que pudiese tener el canal de transmisión.

Se sabe que la información transmitida parte de una fuente binaria que por medio de determinado tratamiento reduce el costo de la transmisión.

Lo que en realidad se pretende, a pesar de todos los procesos en el sistema que pretenden mejoras en el ancho de banda y en la transmisión, el punto de partida es proteger la información de la menor modificación posible, y que los bits entregados sean lo más semejantes posibles al de la fuente de transmisión. La manera en la que se cuantifica el parecido entre emisión y recepción se toma en base de la fiabilidad del canal. Esto se lleva a cabo a través de una tasa de error (BER por sus siglas en ingles Bit Error Rate). Una vez teniendo un valor de BER, se puede preguntar si la tasa de error del sistema de transmisión es menor o no a la aceptable por el receptor. Cuando se sobrepasa la tasa de error máxima, la información se hace intangible. Algunas veces se cree que el aumento de la potencia de transmisión podría solucionar el problema, pero hay que tener en cuenta la relación señal-ruido del sistema ya que no siempre es una propuesta factible.

A continuación se analizará la segunda alternativa, que se trata de la *codificación de canal*, teniendo en cuenta que una codificación apropiada de la información

puede reducir los errores introducidos del canal arbitrariamente, sin sacrificar la tasa de transmisión o almacenamiento.

3.2 Clasificación de la codificación de canal

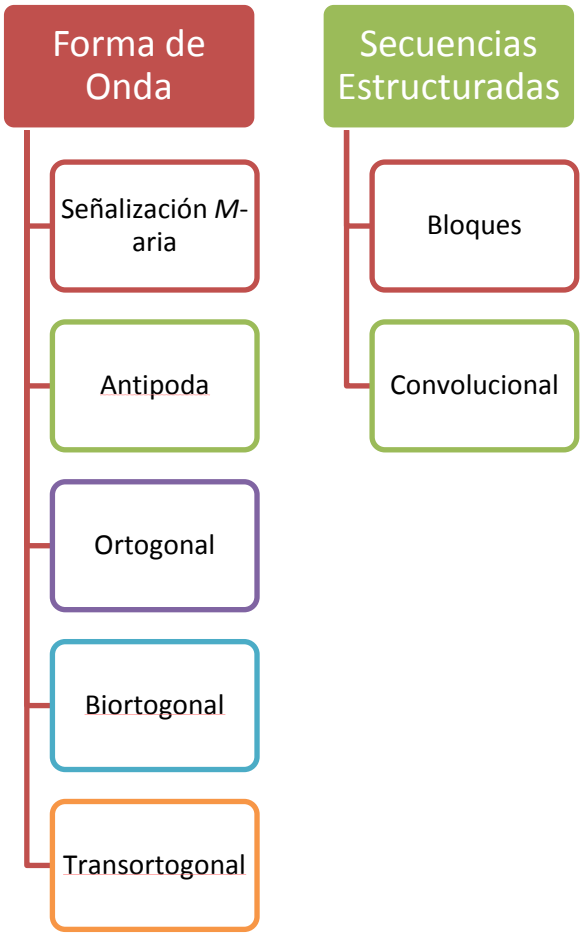


Figura 3.1 Áreas de estudio de la codificación de canal

La codificación de canal, como se muestra en la figura 3.1, puede abordarse de dos modos: *codificación mediante forma de onda* o bien la *codificación mediante secuencias estructuradas*. En el primer caso, se tiene como propósito realizar una

modificación de la forma de onda de la señal con el fin de que el proceso de detección sea menos vulnerable a los errores en la transmisión. En el segundo caso sucede una transformación a las secuencias de datos en las que se agrega cierta *redundancia* (más bits a la información de entrada del canal para que el receptor detecte y corrija los errores en la información transmitida).

Si se tiene pensado simplemente aplicar la detección de errores, no será suficiente para proteger el flujo de datos, sino que será necesario implementar un medio que al detectar determinado error, estime y reconstruya la información que se ha perdido.

3.3 Codificación mediante forma de onda

La codificación mediante formas de onda tiene que ver con la transformación de la forma de onda en una “mejor forma de onda”, como se ha comentado anteriormente, para que la detección de errores se lleve a cabo más objetivamente.

La *codificación de forma de onda* tiene como propósito la transformación de un conjunto de pulsos en un conjunto más elaborado y mejorado, de tal manera que cada una de las formas de onda transmitida sea codificada y sea lo menos parecida posible a cualquier otra forma de cualquier otro conjunto; es decir, se busca que los coeficientes de correlación cruzada entre todos los pares de señales de determinado conjunto sea el menor posible.

Se conocen principalmente dos modos de *codificación mediante forma de onda* los cuales son mediante *señalización antípoda* y mediante *señalización ortogonal*, que se presentan a continuación.

3.3.1 Señalización Antípoda

Se sabe que una señal es antípoda si cuenta con la siguiente característica:

$$s_1\ t\ = -s_2\ t\ = \sin \omega_0 t\ 0 \leq t \leq T \tag{3.1}$$

Se puede afirmar que un conjunto de señales antípodas son *señales espejo*, es decir, una señal es la negativa de la otra, o las señales tiene un desfasamiento de 180° como se muestra a continuación gráficamente:

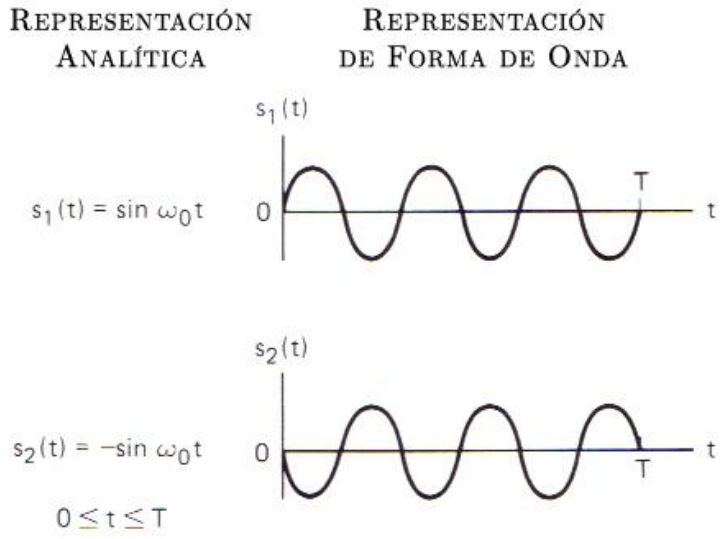


Figura 3.2 Conjunto de señales antípodas

3.3.2 Señalización Ortogonal

Se puede hacer una demostración del concepto de señales ortogonales partiendo de un pulso; un pulso que por sus características nos permitirá hacer una demostración más limpia. Entonces se describe un conjunto de pulsos como:

$$s_1 = p\ \ t \tag{3.2}$$

$$s_2 = p(t - \tfrac{T}{2}) \tag{3.3}$$

$$0 \leq t \leq T\ para\ s_1\ y\ s_2$$

En donde $p\ \ t$ es un pulso con duración $\tau = T/2$ y T es un símbolo de duración. Para una demostración más concisa, a continuación se presenta una gráfica donde se aclara el concepto de ortogonalidad:

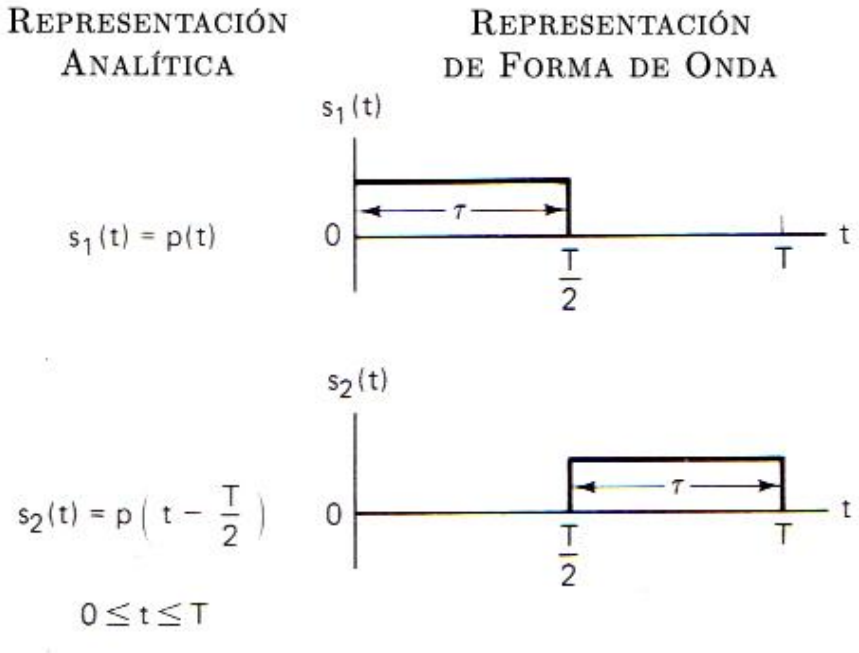


Figura 3.3 Conjunto de señales ortogonales

En esta imagen, se ilustra que $s_1(t)$ y $s_2\ t$ nunca interfieren entre sí porque están desarticuladas en tiempo.

Los procedimientos para llevar a cabo la codificación por forma de onda producen transformaciones en un conjunto de ondas para así generar otro conjunto de formas de onda "perfeccionado". Este conjunto de formas de onda "perfeccionado" puede generar una mejor comparación de probabilidad de bit erróneo con la señal original. Los códigos por forma de onda más conocidos son los códigos ortogonales y los códigos *bi-ortogonales*.

La codificación mediante forma de onda antípoda tiene menor probabilidad de bit erróneo para una misma relación entre Energía de bit y densidad espectral de potencia de ruido (E_b/N_0) respecto a una codificación mediante forma de onda ortogonal. En cambio, la codificación mediante forma de onda ortogonal puede extenderse a más de dos señales, es decir *señalización ortogonal M-aria*.

Al aumentar *M*, se reducirá la probabilidad de bit erróneo, pero aumentará el ancho de banda requerido.

La codificación mediante forma de onda está ampliamente ligada con el proceso de *modulación digital*.

3.3.3 Códigos de línea

Los 0 y 1 binarios pueden representarse en varios formatos de señalización serial de bit llamados *códigos de línea.* Algunos de éstos se presentan en la figura 3.4. Existen dos principales categorías: *con retorno a cero (RZ)* y *sin retorno a cero (NRZ)*. En la codificación RZ la forma de onda regresa a un nivel de cero volts para una porción, generalmente una mitad, del intervalo de bit. Algunos casos de los esquemas de codificación de línea, son parte de la señalización antípoda y

ortogonal. Las formas de onda para los códigos de línea pueden clasificarse aún más de acuerdo a la regla empleada para asignar niveles de voltaje para presentar los datos binarios; a demás, este trabajo se centrará en los códigos de línea presentados en este proyecto de interfaz, los cuales son *señalización unipolar, señalización bipolar, señalización Manchester* y *señalización AMI.*

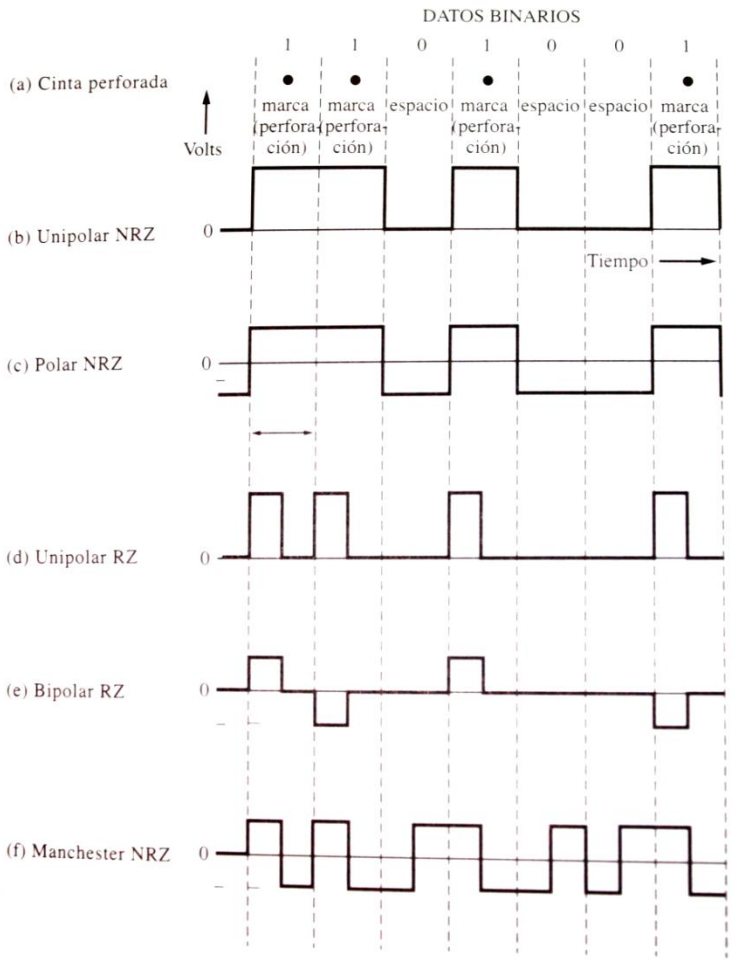


Figura 3.4 Formatos de señalización binaria

En la *señalización unipolar* de lógica positiva, el 1 binario está representado por un nivel alto (+*A* Volts) y el 0 binario por un nivel de cero. Esta señalización también

se conoce como modulación de *encendido-apagado*. Éste tipo de señalización es un ejemplo de señalización ortogonal, la cual se describió en el tema anterior. La energía promedio para éste tipo de señalización se define como:

$$E_b = \frac{1}{2}A^2T \quad \textbf{(3.4)}$$

Donde A es la amplitud de los símbolos, y T es la duración de los mismos.

En la *señalización bipolar NRZ* los 1 y 0 binarios están representados por niveles positivos y negativos iguales. Éste tipo de señalización es un ejemplo de señalización antípoda, la cual se describió en el tema anterior. La energía promedio por dígito binario para la señalización polar se define como:

$$E_b = A^2T \quad \textbf{(3.5)}$$

Donde A es la amplitud de los símbolos, y T es la duración de los mismos.

En la *señalización bipolar* (seudoternaria) los 1 binarios están representados por valores alternativamente positivos y negativos. El 0 binario está representado por un nivel de cero. El término *seudoternario* se refiere al uso de tres niveles de señal codificados para representar datos de dos niveles (binarios). Esto se conoce como señalización de *inversión alterna de marca* (AMI, por sus siglas en inglés).

En la *señalización de Manchester* cada 1 binario está representado por un medio periodo de pulso de bit positivo, seguido de uno negativo. De la misma manera, un 0 binario está representado por un medio periodo de pulso de bit negativo seguido de uno positivo. A este tipo de señalización también se le conoce como *codificación por fase dividida.*

Cada uno de los códigos de línea mostrados en la figura 3.4 tiene sus ventajas y sus desventajas. Por ejemplo, el código de línea unipolar NRZ tiene la ventaja de utilizar circuitos que requieren sólo una fuente de alimentación (por ejemplo de +5 Volts), pero tiene el inconveniente de necesitar canales que están acoplados a DC (es decir, con respuesta de frecuencia hasta $f = 0$) debido a que la forma de onda tiene un valor de DC diferente de cero. El código de línea polar NRZ no requiere un canal acoplado a DC, siempre y cuando los datos alternen entre 1 y 0 binarios frecuentemente, y que se envíe un número igual de 1 y 0 binarios. Sin embargo, los circuitos que pueden producir la señal polar NRZ necesitan de una fuente de alimentación de voltaje negativo, y otra de voltaje positivo. El código de línea de Manchester NRZ tiene la ventaja de que siempre tiene un valor para DC de 0, independientemente de la secuencia de datos, pero tiene el doble de ancho de banda de los códigos unipolar NRZ o polar NRZ, ya que los pulsos son de la mitad del ancho.

Las siguientes son algunas propiedades deseables de un código de línea:

- *Autosincronización*. Existe suficiente información de sincronización incorporada en el código, de manera que los sincronizadores de bit pueden diseñarse para extraer la señal de reloj o sincronización. Una serie larga de 1 y 0 binarios no deberá causar problemas en recuperación de tiempo.
- *Baja probabilidad de error de bit.* Los receptores pueden diseñarse de tal manera que recuperarán los datos binarios con una baja probabilidad de bit erróneo cuando la señal de datos de entrada está afectada debido al ruido.
- *Espectro adecuado para la señal.* Por ejemplo, si el canal está acoplado a AC, la PSD de la señal del código de línea será despreciable en frecuencias cercanas a cero. A demás el ancho de banda de la señal requiere ser lo suficientemente pequeña en comparación al del canal.
- *Ancho de banda de transmisión.* Tiene que ser lo más pequeño posible.

- *Capacidad de detección de errores.* Debe ser posible implementar fácilmente esta característica mediante la adición de codificadores y decodificadores de canal, o la característica debe incorporarse dentro del código de línea.
- *Transparencia.* El protocolo de datos y el código de línea deben estar diseñados de tal forma que *toda posible secuencia de datos* se reciba fiel y *transparente.*

La *densidad espectral de potencia* (PSD) puede evaluarse con una técnica determinista o estocástica. Para determinar la PSD con una técnica determinista, se emplea la forma de onda para un código de línea que resulta de una secuencia en particular de datos. La PSD puede evaluarse con un enfoque estocástico que se usa para obtener la PSD de los códigos de línea mostrados en la figura 3.4, ya que da el resultado de la PSD para el código de línea con una secuencia de datos aleatorios, en lugar de aquella para una secuencia de datos en particular.

Una señal digital (o código de línea) puede representarse de la siguiente manera:

$$s\ t\ =\ \ \sum_{n=-\infty}^{\infty} a_n f(t - nT_s) \qquad \textbf{(3.6)}$$

Donde $f(t)$ es la forma de pulso de símbolo y T_s es la duración del símbolo. Para una señalización binaria $T_s = T_b$, donde T_b es el tiempo que se toma para enviar 1 bit.

La expresión general para la PSD de una señal digital es:

$$\mathfrak{p}_s\ f\ \ = \frac{F(f)^{\ 2}}{T_s}\ \ \sum_{K=-\infty}^{\infty} R(k) e^{j2\pi k f T_s} \qquad \textbf{(3.7)}$$

Donde $F(f)$ es la transformada de Fourier de la forma de pulso $f(t)$ y $R(k)$ es la autocorrelación. Ésta autocorrelación está dada por:

$$R\ k\ =\ \ {}^{I}_{i=1}(a_n a_{n+k})_i P_i \qquad \textbf{(3.8)}$$

Donde a_n y a_{n+k} son los niveles (de voltaje) de los pulsos de datos de la n-ésima y $(n+k)$-ésima posición del símbolo, respectivamente, y P_i es la probabilidad de tener el i-ésimo producto de $a_n a_{n+k}$. La ecuación 3.7 muestra que el espectro de la señal digital depende de dos cosas: 1) La forma de pulso utilizada y 2) las propiedades estadísticas de los datos.

Utilizando las ecuaciones 3.7 y 3.8, que representan un enfoque estocástico, se puede evaluar la PSD para los distintos códigos de línea que se están abordando.

En la tabla 3.1 se muestran algunas propiedades importantes de los códigos de línea abordados en el tema, los cuales se implementaron en la interfaz. Dichas propiedades son los niveles, autocorrelación, PDS y probabilidad de error. A demás en la figura 3.5 se muestra gráficamente la PDS de dichos códigos de línea:

Señalización	Posibles niveles de a (Volts)	Autocorrelación $R(k)$	Densidad Espectral de Potencia (PDS)	Probabilidad de error (P_E)
Unipolar NRZ	$0, +A$	$R_{unipolar}(k) = \begin{cases} \frac{1}{2}A^2 & k = 0 \\ \frac{1}{4}A^2 & k \neq 0 \end{cases}$	$\mathfrak{p}_{unipolarNRZ}(f) = \frac{A^2T_b}{4}\left(\frac{sen\pi fT_b}{\pi fT_b}\right)^2\left[1 + \frac{1}{T_b}\delta(f)\right]$	$P_E = \frac{1}{2}erfc\left(\sqrt{\frac{E_b}{2N_0}}\right) = Q\left(\sqrt{\frac{E_b}{N_0}}\right)$
Bipolar NRZ (Polar NRZ)	$+A, -A$	$R_{polar}(k) = \begin{cases} A^2, & k = 0 \\ 0 & k \neq 0 \end{cases}$	$\mathfrak{p}_{polarNRZ}(f) = A^2T_b\left(\frac{sen\pi fT_b}{\pi fT_b}\right)^2$	$P_E = \frac{1}{2}erfc\left(\sqrt{\frac{E_b}{N_0}}\right) = Q\left(\sqrt{\frac{2E_b}{N_0}}\right)$
AMI (Bipolar RZ)	$+A, -A, 0$	$R_{bipolar}(k) = \begin{cases} \frac{A^2}{2}, & k = 0 \\ -\frac{A^2}{4}, & \lvert k\rvert = 1 \\ 0, & \lvert k\rvert > 1 \end{cases}$	$\mathfrak{p}_{bipolarRZ}(f) = \frac{A^2T_b}{4}\left(\frac{sen(\frac{\pi fT_b}{2})}{\pi fT_b/2}\right)^2 sen^2(\pi fT_b)$	$P_E \approx \frac{3}{4}erfc\left(\sqrt{\frac{E_b}{2N_0}}\right) = \frac{3}{2}Q\left(\sqrt{\frac{E_b}{N_0}}\right), \frac{E_b}{N_0} > 2$
Manchester NRZ			$\mathfrak{p}_{manchesterNRZ}(f) = A^2T_b\left(\frac{sen(\frac{\pi fT_b}{2})}{\pi fT_b/2}\right)^2 sen^2(\frac{\pi fT_b}{2})$	$P_E \approx Q\left(\sqrt{\frac{2E_b}{N_0}}\right)$

Tabla 3.1 Códigos de línea

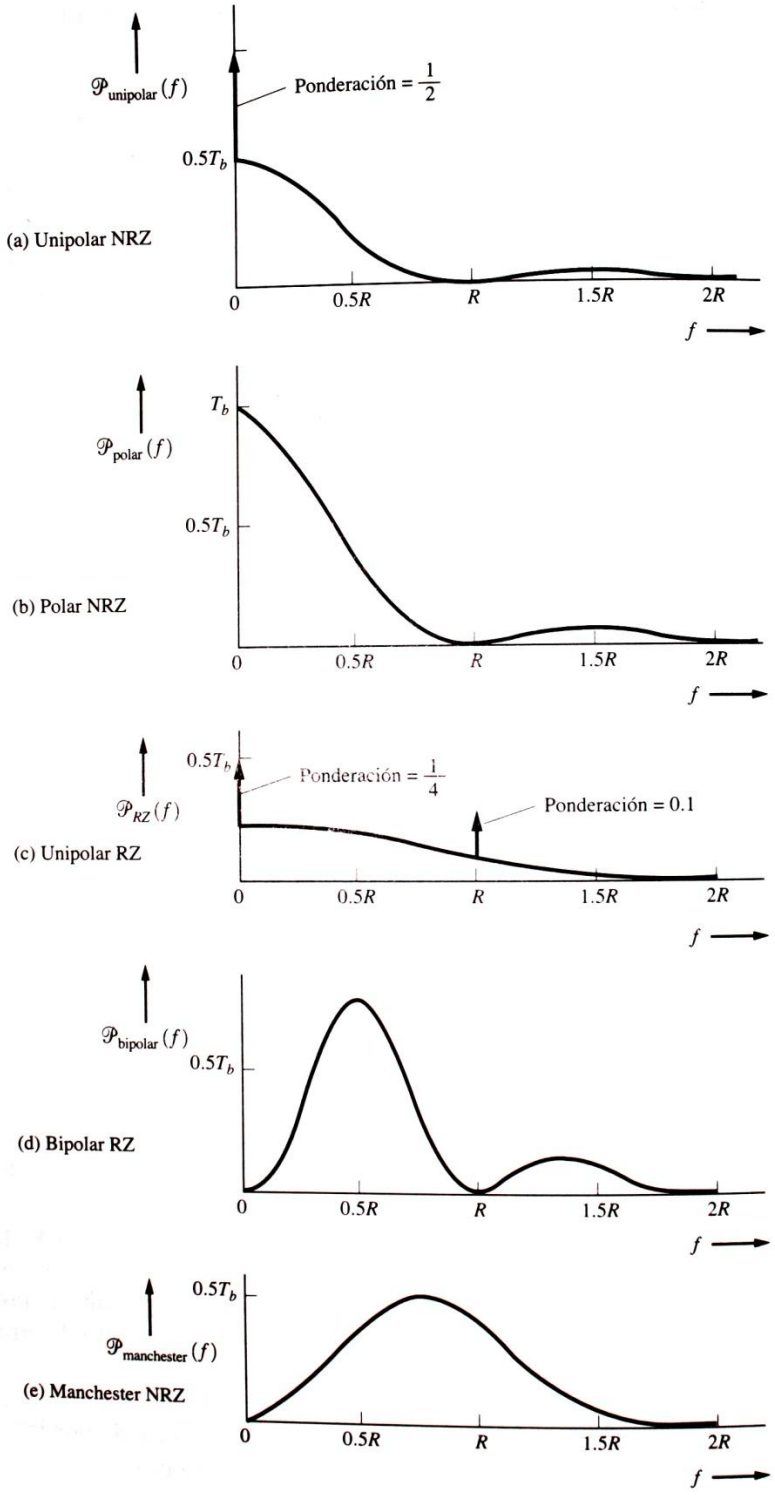


Figura 3.4 PDS de códigos de línea

3.4 Codificación mediante secuencias estructuradas

Algunos procedimientos de codificación de canal son clasificados como *secuencias estructuradas* porque representan métodos de inserción de redundancia estructurada en la información de la fuente para que la presencia de errores pueda ser detectada o para que los errores puedan ser corregidos.

Los *códigos mediante secuencias estructuradas* también son conocidos como *códigos de verificación de paridad*. En estas técnicas de codificación, se agrega *redundancia* a determinada secuencia de datos, de modo que la presencia de errores sea detectable y corregible. A continuación se presenta gráficamente cómo se lleva a cabo la adición de bits redundantes, básicamente:

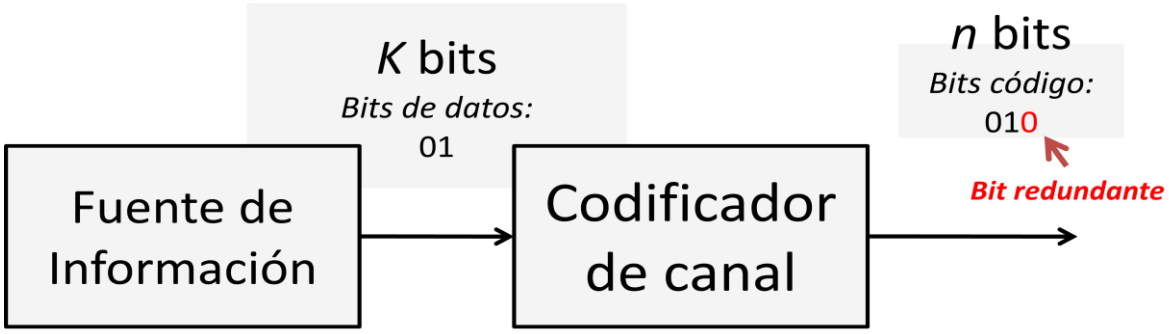


Figura 3.4 Proceso de adición de redundancia a una secuencia de datos

Los *bits redundantes* también se conocen como *bits de paridad* o *bits de verificación*.

Se puede hacer una definición de *redundancia* tal como:

$$\gamma_c = \frac{n-k}{K} = \frac{bits\ redundantes}{bits\ de\ datos} \quad (3.4)$$

Donde k es el número de bits que se están enviando, y n es el número de bits que se están enviando más los bits redundantes.

Del mismo modo, se puede definir la *razón de codificación* como:

$$R = \frac{k}{n} = \frac{bits\ de\ datos}{bits\ código} \quad \textbf{(3.5)}$$

Cuando se agrega redundancia a una secuencia de datos, se incrementa *el ancho de banda* requerido para que se lleve a cabo la transmisión de información.

Para eso existe el *factor de incremento de ancho de banda* que es la razón inversa de la *razón de codificación* y se define como:

$$\Delta BW = \frac{1}{R} \quad \textbf{(3.6)}$$

En canales con ancho de banda fraccionados, se realiza una combinación de modulación M-aria y codificación de canal, con fines de preservar dicha distribución.

La mayor desventaja de con respecto a las técnicas de codificación ortogonal es la ineficiencia asociada con el ancho de banda. Para un conjunto codificado ortogonalmente con formas de onda $M = 2^k$, el ancho de banda requerido para la transmisión es de M/k veces que el requerido para el caso no codificado.

3.4.1 Códigos de verificación de paridad simples

Los códigos de verificación de paridad usan sumas lineales de los bits de información, llamados *símbolos de paridad* o *bits de paridad*, para la detección de errores o corrección de los mismos. Un *código de verificación de paridad simple* se construye añadiendo un bit de paridad simple a un bloque de bits de información. El bit de paridad toma el valor de uno o cero según sea necesario para asegurar

que la suma de todos los bits en la palabra código arroja un valor par (o impar). La suma se lleva a cabo utilizando aritmética de módulo-2 (exclusiva - o lógica). Si la paridad agregada arroja un valor par, el método se designa *paridad par*; si arroja un valor impar, el método se designa *paridad impar*. En la figura 3.5 se ilustra una transmisión de datos en serie (el bit más hacia la derecha es el primero). Un bit de paridad simple es agregado (el bit más a la izquierda en cada bloque) para producir paridad par.

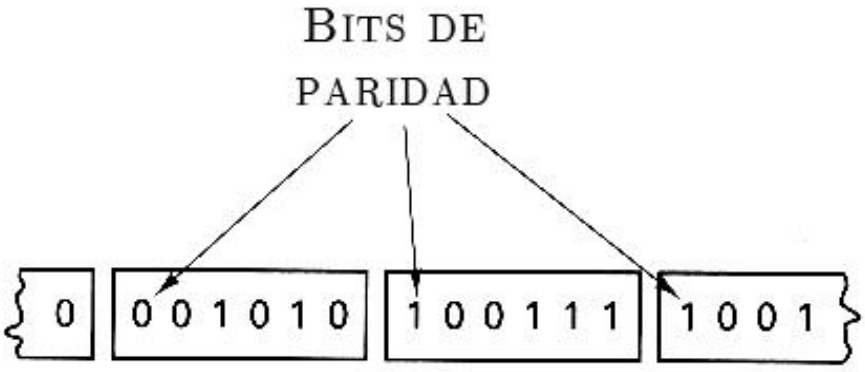


Figura 3.5 Verificación de paridad para una estructura de codificación en serie

En el receptor, el proceso de decodificación consiste en que la suma módulo-2 de los bits de la palabra código produce como resultado cero (paridad par). Si el resultado es uno, se dice que la palabra código contiene errores. La tasa del código se puede expresar como $k/(k+1)$. El decodificador es incapaz de corregir automáticamente los errores; solo puede detectar la presencia de un número impar de bits erróneos. Si hay un número par de errores, la presencia de errores es invertida, la verificación de paridad parecerá correcta, lo cual representa el caso de un *error no detectado*. Asumiendo que la ocurrencia de bits erróneos es equiprobable e independiente, se puede definir la probabilidad de j errores ocurriendo en un bloque de n símbolos como:

$$P\ j,n\ = \ \binom{n}{j}\ p^j(1-p)^{n-j} \tag{3.7}$$

Donde p es la probabilidad de que un símbolo del canal sea recibido con error, y donde:

$$\binom{n}{j} = \frac{n!}{j!\ n-j\ !} \tag{3.8}$$

Es el número de varias formas en las que j bits procedentes de n puedan ser erróneos. Así, para un código detector de errores de verificación de paridad simple, la probabilidad de un error no detectado P_{nd} con un bloque de n bits se representa como:

$$P_{nd} = \begin{matrix} \frac{n}{2}\ para\ paridad\ n \\ \sum_{j=1}^{\frac{n-1}{2}(para\ imparidad\ n)} \end{matrix} \binom{n}{2j}\ p^{2j}(1-p)^{n-2j} \tag{3.9}$$

3.4.2 Códigos rectangulares

Un *código rectangular* puede ser considerado para una estructura de codificación paralela, como se ilustra en la figura 3.6:

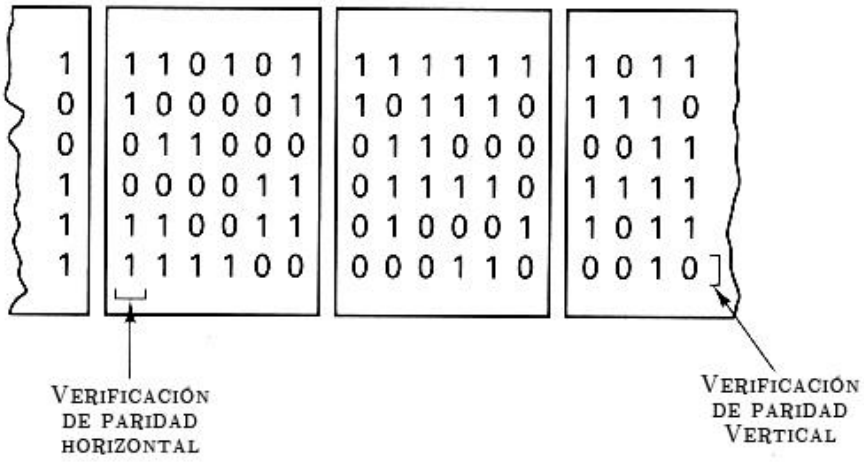


Figura 3.6 **Verificación de paridad para una estructura de codificación paralela**

Primero se forma un rectángulo compuesto por bits de mensaje comprendido por M filas y N columnas; después, se adjunta una verificación de paridad horizontal para cada fila, y una verificación de paridad vertical para cada columna, resultando en una matriz aumentada de dimensiones $M+1 \times N+1$. La tasa del código rectangular k/n puede ser descrita como:

$$\frac{k}{n} = \frac{MN}{M+1\ (N+1)} \tag{3.10}$$

Cualquier bit erróneo causará una falla en la verificación de paridad en una de las columnas de la matriz y en una de sus filas. Por lo tanto, el código rectangular puede corregir cualquier patrón único de error puesto que únicamente un error es localizado en la intersección entre la fila detectora de errores y la columna detectora de errores.

Para el ejemplo mostrado en la figura 3.6, las dimensiones de la matriz son $M = N = 5$; por lo tanto la figura muestra un código que puede corregir un solo error localizado en cualquier posición.

Empezando con la probabilidad de j errores en un bloque de n símbolos, como se expresa en la ecuación 3.7, se puede describir la probabilidad de *error de bloque* para un código capaz de corregir t y menores patrones de error, como:

$$P_M = \sum_{j-t+1}^{n} \binom{n}{j} p^j (1-p)^{n-j} \tag{3.11}$$

Donde p es la probabilidad de que un símbolo del canal sea recibido con error. Para el ejemplo de la figura 3.6, el código puede corregir todos los patrones únicos de error ($t = 1$) dentro del bloque rectangular de $n = 36$ bits. Por lo tanto, la sumatoria en la ecuación 3.11 empieza con $j = 2$:

$$P_M = \sum_{j=2}^{36} \binom{36}{j} p^j (1-p)^{36-j}$$

Dónde p es razonablemente pequeña y el primer término en la sumatoria es el dominante; por lo tanto, para el código rectangular del ejemplo de la figura 3.6, se puede decir que:

$$P_M \approx \binom{36}{2} p^2 (1-p)^{34}$$

La *probabilidad de bit erróneo* P_B depende particularmente del codificador y del decodificador.

3.4.3 Códigos de bloques lineales

Los códigos de bloques lineales son una clase de los códigos de verificación de paridad que pueden ser caracterizados por la notación (n, k). Cada bloque de k bits de mensaje es codificado en un bloque de $n > k$ bits, como se ilustra en la figura 3.7, añadiendo $n - k$ bits de verificación derivados de los k bits de mensaje.

El bloque de n-bits a la salida del codificador del canal se conoce como *palabra código*, y los códigos (o esquemas de codificación) en donde los bits de mensaje aparecen al principio de la palabra código son conocidos como *códigos sistemáticos*. A demás, si cada una de las 2^k palabras códigos puede ser expresadas como combinaciones *lineales* de k vectores código lineales e independientes, entonces el código es llamado *código de bloque lineal sistemático.*

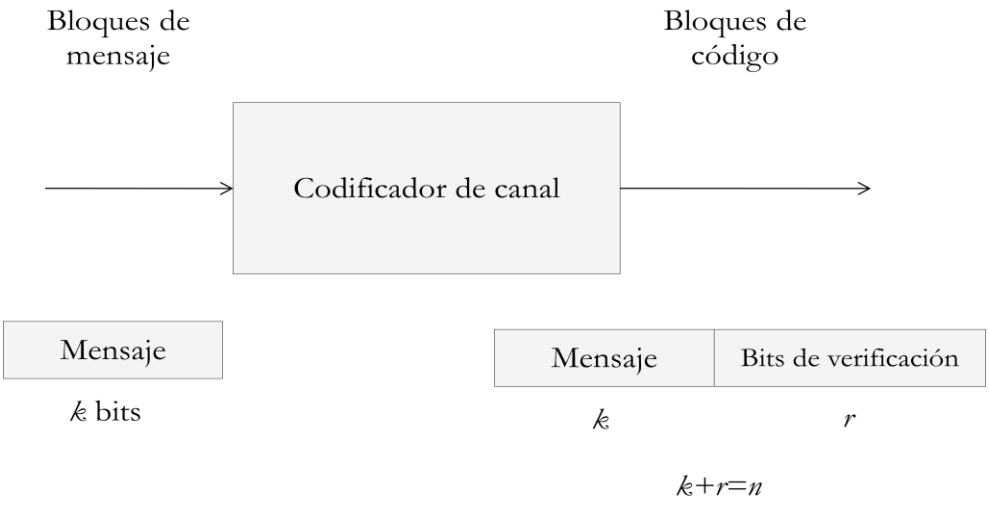


Figura 3.7 Ejemplo de codificador de bloques

La operación de codificación en un esquema de codificación de bloques lineal consiste en dos pasos básicos: 1) La secuencia de información es segmentado en bloques de mensaje; cada bloque consiste en k bits de información sucesivos. 2) El codificador transforma cada bloque de mensaje en un bloque más grande de n bits de acuerdo con un algún conjunto predeterminado de reglas. Estos $n-k$ bits adicionales son generados por combinaciones lineales de bits de mensaje, y se puede describir las operaciones de codificación usando matrices.

Se denotará el bloque de mensaje como un vector fila o secuencia de k-bits $D=(d_1, d_2, d_3, \dots, d_k)$ donde cada bit de mensaje puede ser 0 o 1. Así, tenemos 2^k bloques de mensaje distintos. Cada bloque de mensaje se transforma en una palabra código C con una longitud de n bits ($C=(c_1, c_2, c_3, \dots, c_n)$) por el codificador y hay 2^k palabras código; una palabra código única para cada bloque de mensaje distinto. Éste conjunto de 2^k palabras código distintas, también llamadas vectores código, es denominado código de bloque (n, k). La razón de codificación de este código es k/n.

En un código de bloque lineal sistemático, los primeros k bits de la palabra código son los bits de mensaje, lo cual es:

$$c_i = d_i \qquad i = 1,2,3, \dots, k \tag{3.12}$$

Los últimos $n-k$ bits en la palabra código son bits de verificación generados a partir de los k bits de mensaje, de acuerdo a alguna regla predeterminada:

$$\begin{aligned} c_{k+1} &= p_{11}d_1 + p_{21}d_2 + \cdots + p_{k1}d_k \\ c_{k+2} &= p_{12}d_1 + p_{22}d_2 + \cdots + p_{k2}d_k \\ &\vdots \\ c_n &= p_{1,n-k}d_1 + p_{2,n-k}d_2 + \cdots + p_{k,n-k}d_k \end{aligned} \tag{3.13}$$

Los coeficientes p_{ij} en la ecuación 3.13 son 0's y 1's y las sumas son realizadas en aritmética de módulo-2 ($1+1=0, 0+1=1, 1+0=1, 0+0=0$) de modo que las c_k's son 0's y 1's. Se puede combinar las ecuaciones 3.12 y 3.13 y escribirlas matricialmente:

$$c_1, c_2, \dots, c_{n,} = d_1, d_2, \dots, d_{k,} \begin{matrix} 1000 \dots 0 \\ 0100 \dots 0 \\ 0010 \dots 0 \\ \vdots \quad \vdots \\ 0000 \dots 1 \end{matrix} \begin{matrix} p_{11}p_{12} \dots p_{1,n-k} \\ p_{21}p_{22} \dots p_{2,n-k} \\ p_{31}p_{32} \dots p_{3,n-k} \\ \vdots \quad \vdots \\ p_{k1}p_{k2} \dots p_{k,n-k} \end{matrix}_{k\times n} \tag{3.14}$$

O

$$C = DG \tag{3.15}$$

Donde G es la matriz del lado derecho de la ecuación 3.14. La matriz G $k \times n$ es llamada *matriz generadora* del código y tiene la forma:

$$G = I_k \vdots P_{k\times n} \tag{3.16}$$

La matriz I_k es la matriz identidad de orden k y P es una matriz arbitraria k $(n-k)$. Cuando P se especifica, define el código de bloques (n, k) completamente.

Un paso importante en el diseño de un código de bloques $(n.k)$ es la selección de una matriz P de modo que el código generado por G tenga ciertas propiedades deseables como facilidad de implementación, habilidad de corregir errores

aleatorios y de ráfagas, alta tasa de eficiencia, y así sucesivamente. Mientras que no exista un procedimiento para seleccionar una matriz P que satisfaga todas las limitaciones mencionadas, investigaciones pasadas en la teoría de la información han dado clases especiales de códigos de bloques lineales que poseen muchas propiedades deseables y una cantidad justa de estructura matemática.

El codificador esencialmente tiene que almacenar la matriz G (o por lo menos la submatriz P de G) y realizar operaciones aritméticas binarias para generar los bits de paridad. La complejidad del codificador incrementa a medida del tamaño del bloque n y el número de bits de paridad $n-k$ aumenta.

Una *matriz de verificación de paridad* H es asociada con cada bloque de código (n, k), la cual se define como:

$$H = \left[\begin{array}{cccc|cccccc} p_{11} & p_{21} & \cdots & p_{k1} & 1 & 0 & 0 & 0 & \cdots & 0 \\ p_{12} & p_{22} & \cdots & p_{k2} & 0 & 1 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & & \vdots & & & & & \vdots \\ p_{1,n-k} & p_{2,n-k} & \cdots & p_{k,n-k} & 0 & 0 & 0 & 0 & \cdots & 1 \end{array}\right] = [P^T \vdots I_{n-k}]_{(n-k)\times n} \quad \textbf{(3.17)}$$

La matriz de verificación de paridad puede ser usada para verificar si una palabra código C es generada por la matriz $G = \ I_k \vdots P\ _{k\times n}$. Esta verificación puede ser hecha como a continuación. C es una palabra código en el bloque de código (n, k) generado por $G = \ I_k \vdots P\ _{k\times n}$ sí y solo sí:

$$CH^T = 0 \quad \textbf{(3.18)}$$

Donde H^T es la transpuesta de la matriz H. La prueba de la ecuación 3.18 es simple, al realizar las operaciones en aritmética de módulo-2.

Mientras la matriz generadora es usada en la operación de codificación, la matriz de verificación de paridad es usada en la operación de decodificación, como a continuación. Se considerará un código de bloques lineal (n, k) con una matriz generadora $G = I_k \vdots P_{k\times n}$ y una matriz de verificación de paridad $H = [P^T \vdots I_{n-k}]_{(n-k)\times n}$. C será considerada un vector código que fue transmitido a través de un canal ruidoso y se considerará R como el vector dañado por el ruido que fue recibido. El vector R es la suma del vector código original C y el vector erróneo E, lo que es:

$$R = C + E \tag{3.19}$$

El receptor no conoce C ni E; su función es decodificar C a partir de R, y el bloque mensaje D a partir de C. El receptor realiza la operación de decodificación determinando un vector $S\ (n-k)$ definido como:

$$S = RH^T \tag{3.20}$$

El vector S es denominado *síndrome de error* de R. Se puede reescribir como:

$$S = \ C + E\ H^T = CH^T + EH^T$$

Obteniendo:

$$S = EH^T \tag{3.21}$$

Ya que $CH^T = 0$. Así, el síndrome de un vector recibido será cero si R es un vector código válido. Si ocurren errores en la transmisión, entonces el síndrome S del vector recibido será distinto a cero. A demás, S está relacionado al vector erróneo E y el decodificador usa S para detectar y corregir errores.

El peso Hamming de un vector código C es definido como el número distinto a cero de componentes de C. La distancia Hamming entre dos vectores C_1 y C_2 es definida como el número de componentes en los que ambos vectores difieren. Finalmente, la distancia mínima de un bloque de código es la distancia más pequeña entre un par de palabras código en el código.

3.5 Códigos convolucionales

Un *código convolucional* está descrito por tres elementos n, k y K, donde la proporción k/n tiene el mismo significado de velocidad de código (información por bit codificado) que el de códigos de bloques. Sin embargo, n no define un bloque o longitud de palabra código como en los códigos de bloques. El elemento K es un parámetro conocido como *tamaño de la memoria.* Representa el número de etapas de k-bits en la variación del registro de codificación. Una característica importante de de los códigos convolucionales, a diferencia de los códigos de bloques, es que el codificador tiene memoria (la secuencia de n-bits emitida por un proceso de codificación convolucional no es solo función de una secuencia de k-bits a la entrada, sino que también es una función de $K-1$ secuencias de k-bits a la entrada). En la práctica, n y k son elementos pequeños y K es variado para controlar la capacidad y complejidad del código.

En la figura 3.8 se muestra una versión del diagrama a bloques típico de un sistema de comunicaciones digitales adaptado en función de los procesos de codificación/decodificación convolucional:

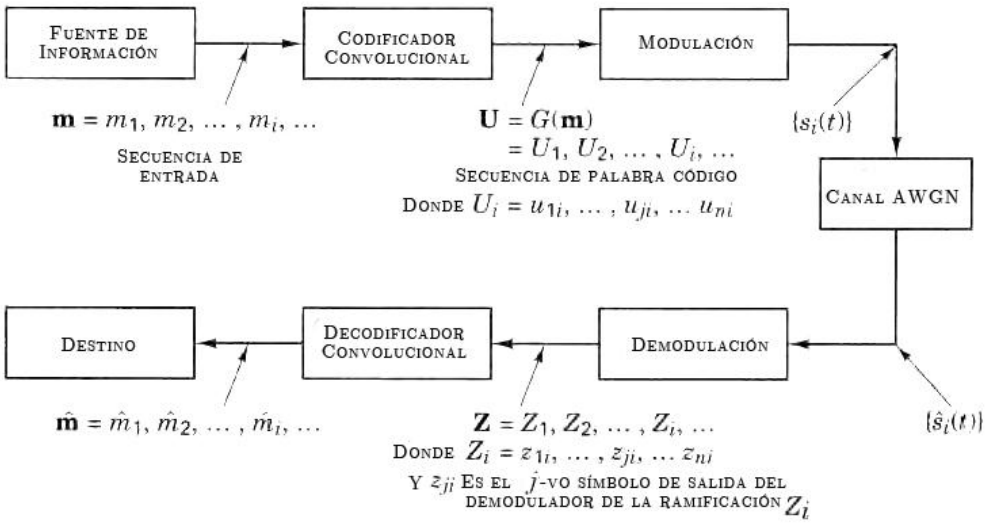


Figura 3.8 Diagrama a bloques típico de un sistema de comunicaciones digitales adaptado a procesos de codificación/decodificación convolucional

La fuente de mensaje a la entrada es denotada por la secuencia $m = m_1, m_2, \ldots, m_i, \ldots$ donde cada m_i representa un dígito binario (bit), e i es un índice de tiempo. Para ser precisos, se deben denotar los elementos de m con un índice de pertenencia (1 o 0 para códigos binarios) o un índice de tiempo. Se asumirá que cada m_i igualmente probable de ser uno o cero, e independientes entre dígitos. Siendo independiente, la secuencia de bits carece de redundancia; esto significa que al conocer un bit m_i no tendremos información a cerca de m_j ($i \neq j$). El codificador transforma cada secuencia m en una secuencia de palabra código única $U = G(m)$. Aunque la secuencia m defina únicamente a una secuencia U una característica clave de los códigos convolucionales es que dado una secuencia de k-bits en m no definirá únicamente su secuencia de n-bits asociada en U pues la codificación de cada secuencia de k-bits no es solamente función de esa secuencia de k-bits, pero también está en función de $K-1$

secuencias de k-bits de entrada que lo preceden. La secuencia U puede ser particionada en una secuencia de ramificaciones: $U = U_1, U_2, \dots, U_i, \dots$ Cada ramificación U_i se compone *símbolos de codificación binarios,* a menudo llamados *símbolos de canal, bits de canal* o *bits de código*. A diferencia de los bits de mensaje a la entrada, los símbolos de código no son independientes.

En una aplicación típica de comunicaciones la secuencia de palabras código U modula una forma de onda $s(t)$. Durante la transmisión, la forma de onda $s(t)$ es afectada por el ruido, resultando en una forma de onda recibida $s(t)$ y en una secuencia demodulada $Z = Z_1, Z_2, \dots, Z_i, \dots$ (Figura 3.8). La tarea del decodificador es producir una estimación $m = m_1, m_2, \dots, m_i, \dots$ de la secuencia original de mensaje, usando la secuencia recibida Z conjunto a un conocimiento a priori en el proceso de decodificación.

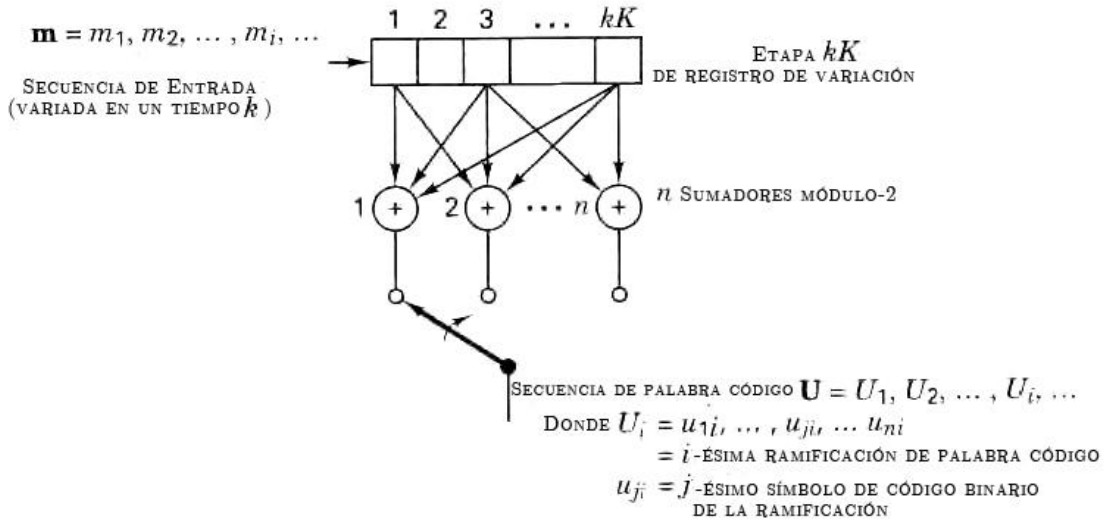


Figura 3.9 Codificador convolucional con tamaño de memoria K y tasa k/n

Un codificador convolucional general, como se muestra en la figura 3.9, está mecanizado con una etapa-kK de registro de variación y n sumadores módulo-2, donde K es la constante de restricción.

El tamaño de la memoria representa el número de variaciones de k-bits sobre el cual un solo bit de información puede influenciar la salida del codificador. A cada unidad de tiempo, k bits son variados en las primeras k etapas del registro; todos los bits en el registro son variados k etapas a la derecha, y la salida de los n sumadores son secuencialmente muestreados para producir los símbolos de código binarios o bits de código. Estos símbolos de código son usados por el modulador para especificar la forma de onda que será transmitida a través del canal. Puesto que hay n bits de código para cada grupo de salida de k bits de mensaje, la tasa de codificación es k/n bits de mensaje por bit codificado, donde $k < n$.

3.5.1 Diagrama de conexión

Para describir un código convolucional, se necesita caracterizar la función de codificación $G(m)$, de modo que dada una secuencia de entrada m, uno pueda fácilmente calcular la secuencia de salida U. Existen varios métodos para representar un codificador convolucional, entre los cuales se encontrará el *diagrama de conexión.*

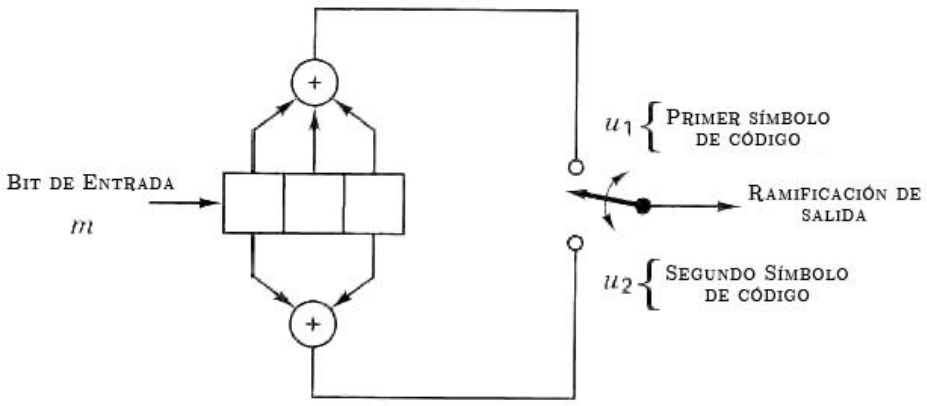


Figura 3.10 Codificador convolucional (tasa $1/2$, $K = 3$)

En la figura 3.10 se ilustra un modelo para discutir acerca de los codificadores convolucionales. La figura ilustra un codificador convolucional $(2,1)$ con tamaño de memoria $K = 3$. Tiene $n = 2$ sumadores módulo-2 y su tasa de codificación es $k/n = 1/2$. En cada intervalo de entrada de bit, un bit es variado hacia la etapa más a la izquierda, y los bits en el registro son variados una posición hacia la derecha. Después, el conmutador de salida muestrea la salida de cada sumador módulo-2 (primero la salida del sumador superior) formando así el par de símbolos de código que constituyen la ramificación asociada con el bit recién introducido. El muestreo es repetido para cada bit introducido. La selección de las conexiones entre los sumadores y las etapas (stages) de registro da lugar a las características del código. Cualquier cambio en la selección de las conexiones resulta en un código diferente. Las conexiones no son escogidas o cambiadas arbitrariamente. El problema al escoger las conexiones para producir buenas propiedades de distancia es complicado y no ha sido resuelto en general; sin embargo, códigos útiles han sido encontrados a través de búsqueda por computadora para todos los tamaños de memoria menores a 20, aproximadamente.

A diferencia de un código de bloques con una longitud de palabra fija n, un código convolucional no tiene un tamaño de bloque particular. Sin embargo, los códigos convolucionales a menudo se ven forzados a componer una estructura de bloques por medio de un *truncamiento periódico*. Esto requiere un número de bits cero adjuntos al final de la secuencia de información de entrada con el propósito de limpiar el registro de variación de codificación de los bits de información. Ya que los ceros agregados no cuentan con información, la tasa de efectividad del código cae por debajo de k/n. Para mantener la tasa de código cerca a k/n, el periodo de truncamiento se implementa en la práctica.

Una manera de representar al codificador es especificando un conjunto de n vectores de conexión, uno por cada uno de los n sumadores módulo-2. Cada vector tiene una duración K y describe la conexión del registro de variación de codificación al sumador. Un uno en la i-ésima posición del vector indica que la

etapa correspondiente en el registro de variación es conectada al sumador, y un cero en la posición dada indica que no existe conexión entre la etapa y el sumador. Para el ejemplo de codificador en la figura 3.10, se puede asignar un vector de conexión g_1 para las conexiones superiores y g_2 para las conexiones inferiores como a continuación:

$$g_1 = 111$$

$$g_2 = 101$$

Ahora, se considerará que un vector mensaje $\mathrm{m} = 101$ es codificado convolucionalmente con el codificador de la figura 3.10. Los tres bits de mensaje son introducidos, uno por uno, a tiempos $\mathrm{t}_1, \mathrm{t}_2$ y t_3, como se muestra en la figura 3.11. Subsecuentemente, $\mathrm{K} - 1 = 2$ ceros son introducidos a tiempos t_4 y t_5 para limpiar el registro y así asegurar que el final del mensaje es desplazado a la longitud total del registro. La secuencia de salida será 1110001011, donde el símbolo más a la izquierda representa la primera transmisión. La secuencia de salida total, incluyendo los símbolos de código como resultado del "lavado" son necesarios para decodificar el mensaje. Para limpiar el mensaje desde el codificador se requiere un cero menos que el número de etapas en el registro, o $K - 1$ bits de limpieza. Otro cero a la entrada será mostrado al tiempo t_6, para verificar que la limpieza ha sido completada en t_5. Así, un nuevo mensaje podrá ser introducido en el tiempo t_6.

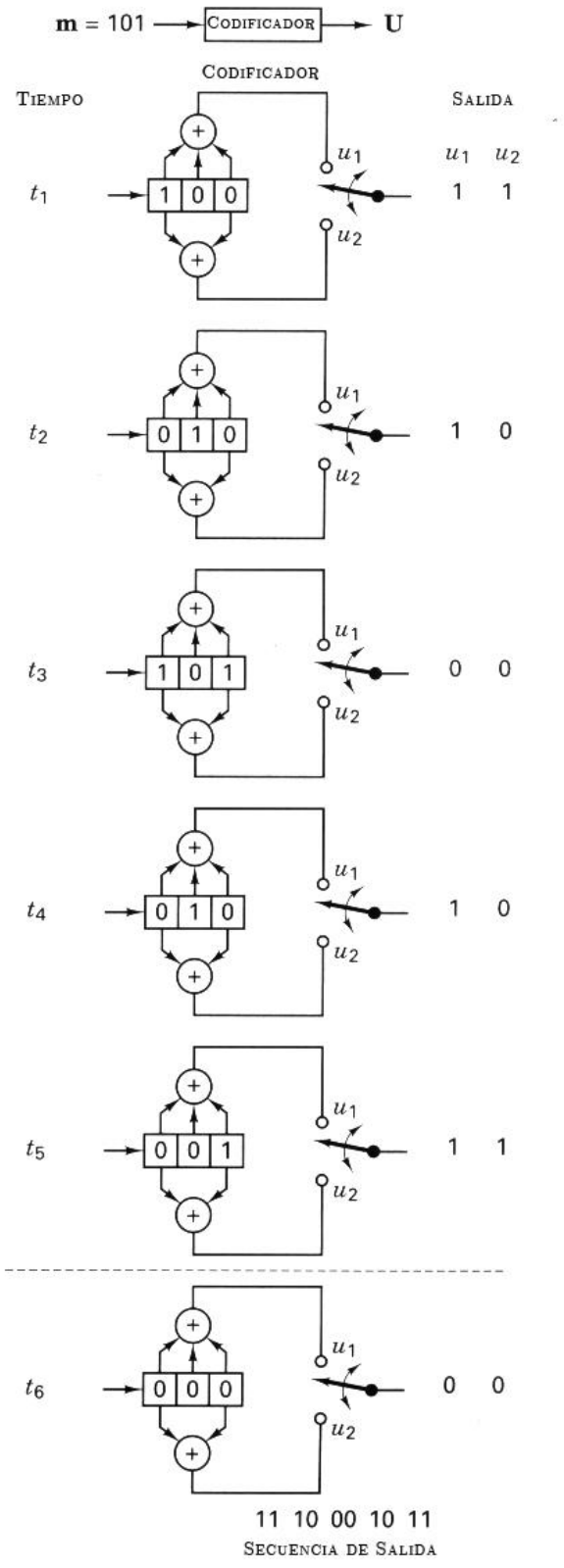


Figura 3.11 Codificación convolucional de una secuencia (tasa $1/2$, $K = 3$)

3.5.2 Respuesta al impulso

Se puede hacer un enfoque del codificador en términos de la respuesta al impulso (la respuesta del codificador ante un solo bit de valor uno que se desplaza a través de él). Considerando que el contenido de los registros del codificador en la figura 3.10 es un uno que lo atraviesa, se puede decir que:

Contenido de registros	Ramificación	
	u_1	u_2
100	1	1
010	1	0
001	1	1

Secuencia de entrada: 1 0 1

Secuencia de salida: 11 10 11

La secuencia de salida para el “uno” de entrada es denominada la respuesta impulso del codificador. A demás, para la secuencia de entrada $m = 101$, la salida puede ser encontrada por la *superposición* o la *suma lineal* de los impulsos de entrada desplazados en el tiempo como a continuación:

Entrada m	Salida				
1	11	10	11		
0		00	00	00	
1			11	10	11
Suma módulo-2	11	10	00	10	11

Obsérvese que ésta es la misma salida obtenida que en la figura 3.11, demostrándose así que los códigos convolucionales *son lineales*, así como los códigos de bloques. El nombre de *codificador convolucional* se deriva a partir de esta propiedad de generar una salida por medio de la suma lineal de impulsos desplazados en el tiempo, o la convolución de la secuencia de entrada con la respuesta al impulso del codificador.

Nótese que la tasa de efectividad de código del ejemplo con secuencia de entrada de 3 bits y secuencia de salida de 10 bits es $k/n = 3/10$, un poco menos que la tasa $1/2$ que puedo haber sido esperada a partir del hecho de que cada bit de entrada produce un par de bits de canal a la salida. La razón de esta disparidad es que el último bit de información en el codificador necesita ser desplazado a lo largo del codificador. Todos los bits de canal a la salida se necesitan en el proceso de decodificación. Si el mensaje hubiera sido más largo, por ejemplo de 300 bits, la palabra código a la salida contendría 604 bits, resultando en una tasa de codificación de $300/604$, mucho más cercana a $1/2$.

3.5.3 Representación polinomial

Algunas veces, las conexiones del codificador son caracterizadas por medio de *polinomios generadores*. Se puede representar un codificador convolucional con un conjunto de n *polinomios generadores*, uno para cada uno de los n sumadores módulo-2. Cada polinomio es de grado $K-1$ o menor y describe la conexión de los registros de variación de codificación con los sumadores, de la misma manera que los vectores de conexión lo hacen. El coeficiente de cada término de grado $K-1$ es, ya sea 1 o 0, dependiendo de si la conexión existe o no entre los registros de variación y los sumadores en cuestión. Para el ejemplo que se ha estado utilizando, se puede escribir el polinomio generador $g_1(X)$ para las conexiones superiores y $g_2(X)$ para las conexiones inferiores como:

$$g_1\ X\ = 1 + X + X^2$$

$$g_2\ X\ = 1 + X^2$$

Donde el término de menor orden en el polinomio corresponde a la etapa de entrada del registro. La salida de secuencia se encuentra de la siguiente manera:

$$U\ X\ = m\ X\ g_1\ X\ entrelazado\ con\ m\ X\ g_2\ X$$

Primero, se expresa el vector mensaje $m = 101$ como un polinomio. Esto es $m\ X\ = 1 + X^2$. Se volverá a hacer uso de los “ceros” después de los bits de mensaje para limpiar el registro. Por lo tanto, el polinomio de salida $U(X)$ o la

secuencia de salida U puede ser encontrada a partir del mensaje de entrada m como:

$$\mathbf{m}(X)\mathbf{g}_1(X) = (1 + X^2)(1 + X + X^2) = 1 + X + X^3 + X^4$$
$$\mathbf{m}(X)\mathbf{g}_2(X) = (1 + X^2)(1 + X^2) = 1 + X^4$$

$$\mathbf{m}(X)\mathbf{g}_1(X) = 1 + X + 0X^2 + X^3 + X^4$$
$$\mathbf{m}(X)\mathbf{g}_2(X) = 1 + 0X + 0X^2 + 0X^3 + X^4$$

$$\mathbf{U}(X) = (1,1) + (1,0)X + (0,0)X^2 + (1,0)X^3 + (1,1)X^4$$
$$\mathbf{U} = 1\ 1 \quad 1\ 0 \quad 0\ 0 \quad 1\ 0 \quad 1\ 1$$

Se representará el codificador con *polinomios generadores* como los que se usan para describir códigos cíclicos. Sin embargo, llegamos a la misma secuencia de salida que en la figura 3.11, que se trata de la misma que obtuvimos con el enfoque de respuesta al impulso.

3.5.4 Diagrama de estados

Un codificador convolucional pertenece a una clase de dispositivos conocidos como *máquinas de estado finito*, que es el nombre general dado a máquinas que tienen memoria de señales anteriores. El adjetivo *finito* se refiere al hecho de que solo hay un número finito de estados únicos que la máquina puede encontrar. Un *estado* consiste en la cantidad más pequeña de información que, junto a una entrada en la máquina, puede predecir la salida de la misma. El estado provee algún conocimiento de las señales anteriores a través de un estado anterior. Para un codificador convolucional con tasa $1/n$, un estado es representado a través del contenido de $K - 1$ etapas más hacia la derecha. El conocimiento del estado, junto

con el conocimiento de la siguiente entrada es necesario y suficiente para determinar la siguiente salida. Se asumirá que el estado del codificador a un tiempo t_1 está definido como $X_i = m_{i-1}, m_{i-2}, \dots, m_{i-K+1}$. La i-ésima ramificación U_i es determinada completamente por el estado X_i y el presente bit de entrada m_i. Así, el estado X_i representa el historial pasado del codificador en determinado entrada del mismo. El estado del codificador se dice que es *Markov,* en el sentido de que la probabilidad $P(X_{i+1}|X_i, X_{i-1}, \dots, X_0)$ de estar en el estado X_{i+1} dados todos los estados previos, depende solamente del estado más reciente X_i; esto significa que la probabilidad es igual a $P(X_{i+1}|X_i)$.

Una manera de representar codificadores simples es con un *diagrama de estados*. Una representación del ejemplo que se ha manejado se presenta en la figura 3.12. Los estados, mostrados en las casillas del diagrama, representan los posibles contenidos de la etapa $K-1$ más a la derecha del registro, y las trayectorias entre los estados representan las ramificaciones a la salida resultantes de las transiciones de estados. Los estados de los registros se designan como $a = 00, b = 10, c = 01,$ y $d = 11$; el diagrama mostrado en la figura 3.12 ilustra todas las transiciones de estados que son posibles para el codificador que se ha estado usando como ejemplo. Solo hay dos transiciones emanando de cada estado, que corresponden a los dos posibles bits de entrada. Junto a cada trayectoria entre estados, está escrita la ramificación de salida asociada con la transición de estado. En el diagrama, se usa la asociación de que una línea sólida denota una trayectoria asociada con un bit de entrada, cero, y una línea discontinua denota una trayectoria asociada con un bit de entrada, uno. Nótese que no es posible en ninguna sola transición moverse de un estado dado a un estado arbitrario. Como consecuencia de un cambio a la vez entre bits, solo hay dos posibles transiciones de estado que el registro puede hacer en cada intervalo de bit. Por ejemplo, si el estado presente del codificador es 00, las únicas posibilidades de estado para la siguiente variación son 00 o 10.

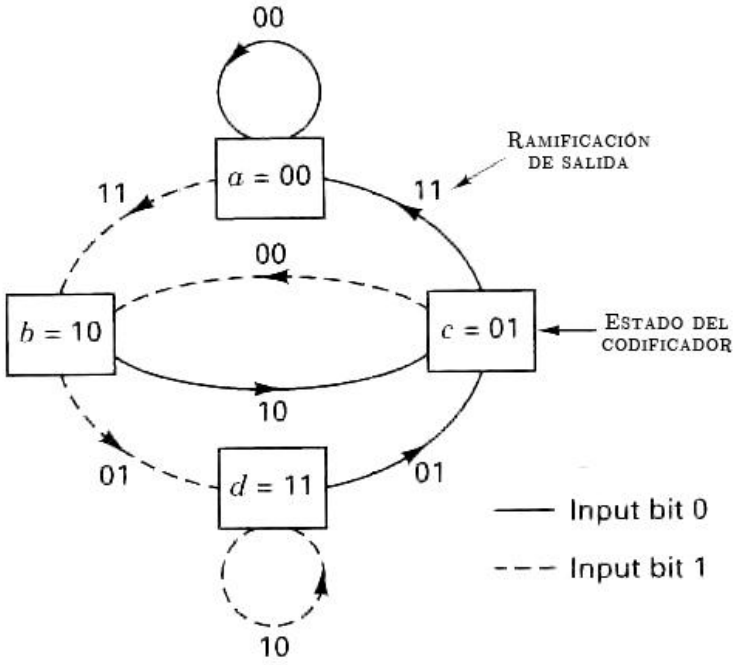


Figura 3.12 Diagrama de estados (tasa $1/2$, $K = 3$)

3.5.5 Diagrama de árbol

Aunque el diagrama de estados caracterice completamente al codificador, uno no puede usarlo fácilmente para rastrear las transiciones del codificador en función del tiempo ya que el diagrama no representa el historial del codificador en función del tiempo. El *diagrama de árbol* agrega una dimensión temporal al diagrama de estados. El diagrama de árbol para el ejemplo que se ha estado utilizando (Figura 3.10) se ilustra en la figura 3.13. El proceso de codificación puede ser descrito atravesando el diagrama de izquierda a derecha en cada bit a la entrada (sucesivo y en función del tiempo). Cada rama del árbol describe una ramificación del proceso de codificación convolucional. La regla de ramificación del diagrama de árbol para encontrar una palabra código es la siguiente: Si el bit de entrada es un cero, su ramificación asociada será encontrada moviéndose a la siguiente rama

ubicada más hacia la derecha en dirección superior. Si el bit de entrada es un uno, su ramificación asociada será encontrada moviéndose a la siguiente rama ubicada más hacia la derecha en dirección inferior. Asumiendo que el contenido inicial del codificador es cero, el diagrama mostrará que si el primer bit de entrada es un cero, la ramificación de salida es 00 y, si el primer bit de entrada es un uno, la ramificación de salida es 11. Semejantemente, si el primer bit de entrada es un uno, y el segundo bit de entrada es un cero, la segunda ramificación de salida es 10. O, si el primer bit de entrada es un uno, y el segundo bit de entrada es un uno, la segunda ramificación de salida es 01. Siguiendo este procedimiento, se verá que la trayectoria de la secuencia de entrada 11011traza la línea gruesa en el diagrama de árbol de la figura 3.13. Ésta trayectoria corresponde a la palabra código de salida 1101010001.

La dimensión temporal agregada en el diagrama de árbol (comparado con el diagrama de estados) permite dinámicamente describir el codificador en función de una secuencia de entrada en particular. Sin embargo, el número de ramas del diagrama se incrementa en función de 2^L, donde L es el número de ramificaciones en la secuencia.

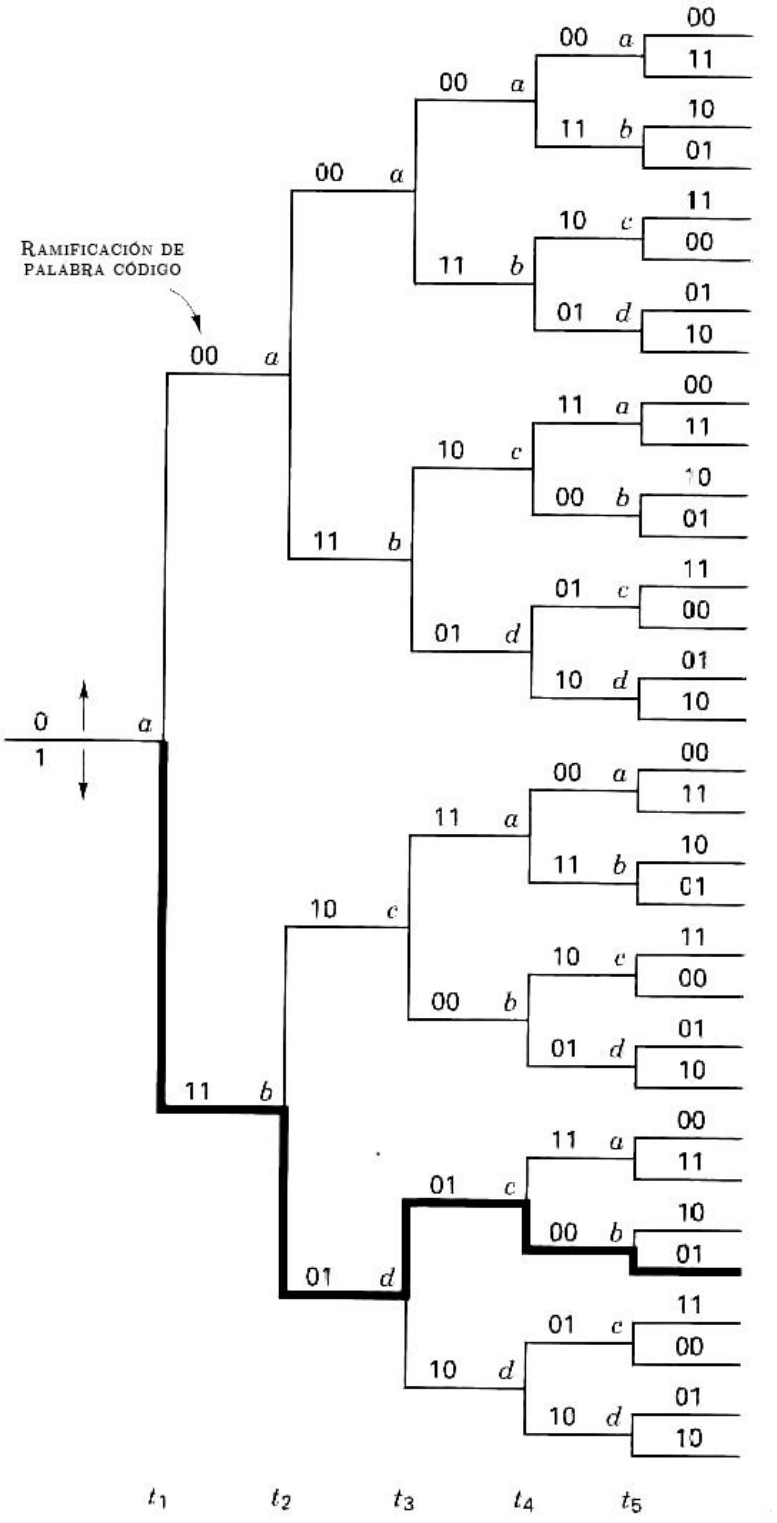
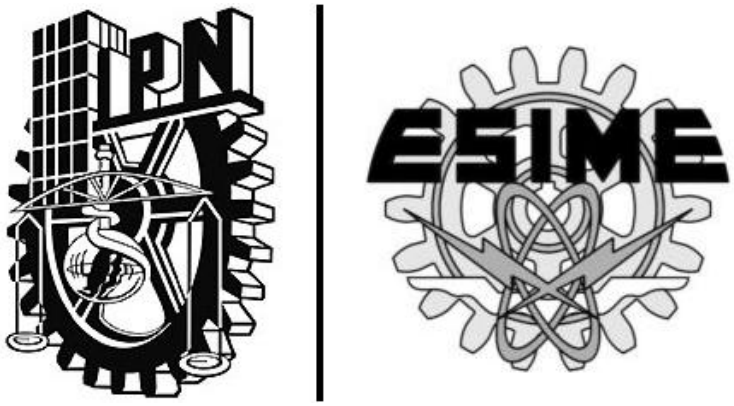


Figura 3.13 Diagrama de árbol (tasa $1/2$, $K = 3$)



CAPÍTULO IV | “Desarrollo experimental”

CAPÍTULO 4: DESARROLLO EXPERIMENTAL

4.1. Generador de señales arbitrarias

Para el propósito de este proyecto se utiliza el generador RIGOL DG2000 el cuál es un equipo del laboratorio que provee gran estabilidad, precisión y fácil familiarización con el usuario, lo que lo hace un dispositivo versátil para diferentes aplicaciones. Tiene la capacidad generar formas de onda ya establecidas, así como también funciones arbitrarias creadas por el usuario por medio de un software. A continuación se mencionan algunas características y especificaciones importantes:

- 10 formas de onda definidas:
  SENO, CUADRADA, RAMPA, PULSO, RUIDO, EXPONENCIAL ASCENDENTE, EXPONENCIAL DESCENDENTE, CARDIACA Y C.D.
- Características en Frecuencia:
  Seno/ Cuadrada: 1μHz a 40 MHz
  Rampa: 1μHz a 400 kHz
  Pulso: 500μHz a 16MHz
  Ruido Blanco: Ancho de banda de 20MHz (-3dB)
  Señales Arbitrarias: 1μHz a 12MHz
- Rango de Amplitud: 2mVpp a 10Vpp (50 ohms) y 40mVpp a 20Vpp (High Z).

Frecuencia de muestreo de 100 Millones de muestras/segundo que permite editar señales arbitrarias con 14-bit de resolución, 512K puntos.

Soporta dispositivos de almacenamiento USB. Puede guardar y leer formas de onda arbitrarias previamente editas en un dispositivo USB.

El panel frontal (figura 4.1) presenta un diseño amigable para el usuario, proporcionando el fácil control y ubicación de las funciones, botones, conectores y pantalla para una completa interacción.

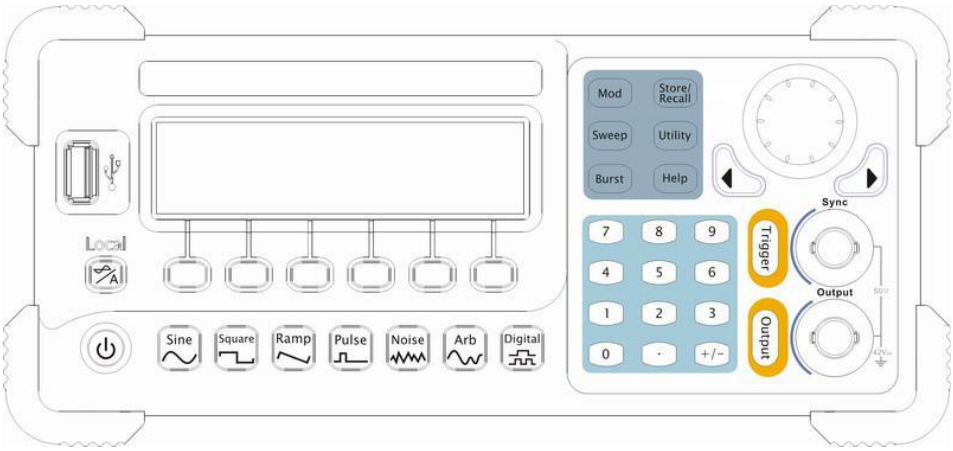


Figura 4.1 Panel frontal RIGOL DG2000 Series

4.1.1. Ubicación del panel de trabajo y grupos de botones/función

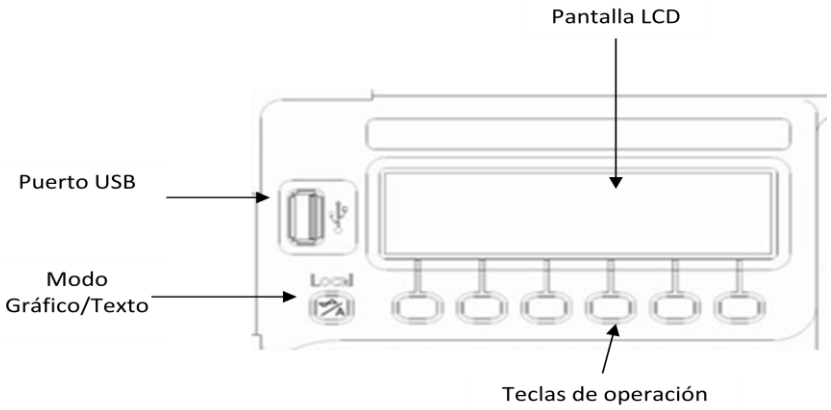


Figura 4.2 Panel con pantalla LCD

4.1.1.1. Pantalla LCD

La pantalla es una interfaz que le permite al operador interactuar con los botones del generador y tener control sobre ellos. Hay dos modos de visualizar la información arrojada en pantalla, que se puede elegir haciendo uso de la tecla de cambio de Modo Gráfico/Texto.

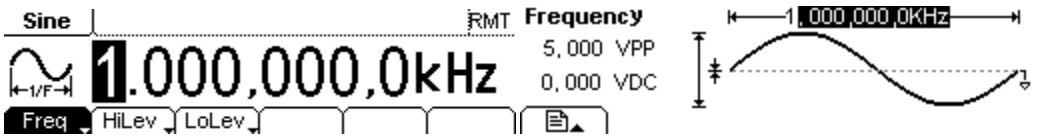


Figura 4.3 **(a)modo texto (b)modo gráfico**

Como se muestra en la figura 4.3, en el caso de modo texto básicamente se observa el valor numérico y la unidad que se está trabajando, ya sea Hz, Vpp, etc. Del lado izquierdo muestra una figura representativa del tipo de señal con la que se esté trabajando. Debajo se encuentra un menú que es controlado por las teclas de operación, esto es para visualizar y editar parámetros de la señal. Por otro lado el modo gráfico es simplemente una forma diferente de visualizar los parámetros de una señal.

4.1.1.2. Formas de onda

En la parte inferior se ubica una línea de botones con las diferentes formas de onda con las que se puede trabajar, basta con presionar directamente alguno de ellos para cambiar la forma de la señal. Como lo muestra la figura 4.4, cada botón lleva el nombre del tipo de señal y una pequeña representación.

Figura 4.4 Botones de selección de forma de onda

A su lado izquierdo se ubica el botón de encendido/apagado. Este grupo de botones se iluminan de color verde cuándo son seleccionados y se apagan al ser presionados de nuevo ó al presionar otro botón en el caso del grupo de botones de selección de formas de onda.

4.1.1.3. Teclado numérico y otras funciones

En la figura 4.5 se muestra un teclado numérico (a) y un botón de dirección acompañado de dos botones auxiliares de desplazamiento (b). Este grupo da facilidad al usuario de introducir las magnitudes deseadas con la exactitud deseada. Puede ser de una forma rápida tecleando los números ó girando el botón de dirección en ambos sentidos. Esta característica del generador lo hace muy versátil, para todo tipo de usuario.

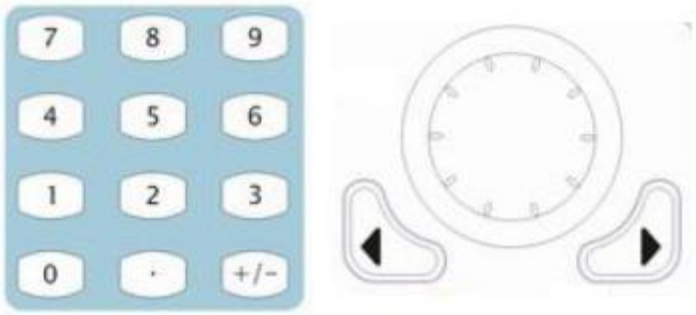


Figura 4.5 (a)Teclado numérico (b)Botones de dirección

Figura 4.6 Grupo de botones

El panel de control de la figura 4.6 se encuentran botones que funciones como: modulación, barrido, ayuda, almacenamiento de formas de onda, entre otras.

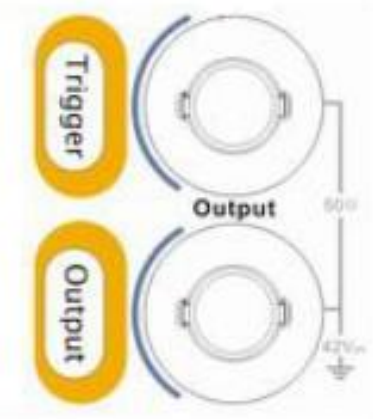


Figura 4.7 Puertos de salida tipo BNC

En la parte inferior derecha se encuentran dos puertos tipo BNC. Cada puerto tiene un botón de control que se encarga de habilitar ó deshabilitar el puerto. El puerto a utilizar será el inferior, que es controlado por el botón “Output” véase figura 4.7.

4.1.2. Software

El generador RIGOL cuenta con un software que brinda todas las herramientas necesarias para crear, manipular y medir señales arbitrarias. Su nombre es Ultrawave.exe y viene incluido en un CD como parte de accesorios al adquirir el generador, también está disponible su descarga gratuita en el siguiente sitio web: http://www.rigol.com/prodserv/DG2000/, dónde también se puede encontrar documentación referente al dispositivo, en caso de no tener la facilitación del CD.

4.1.2.1. Instalación e interacción con el software

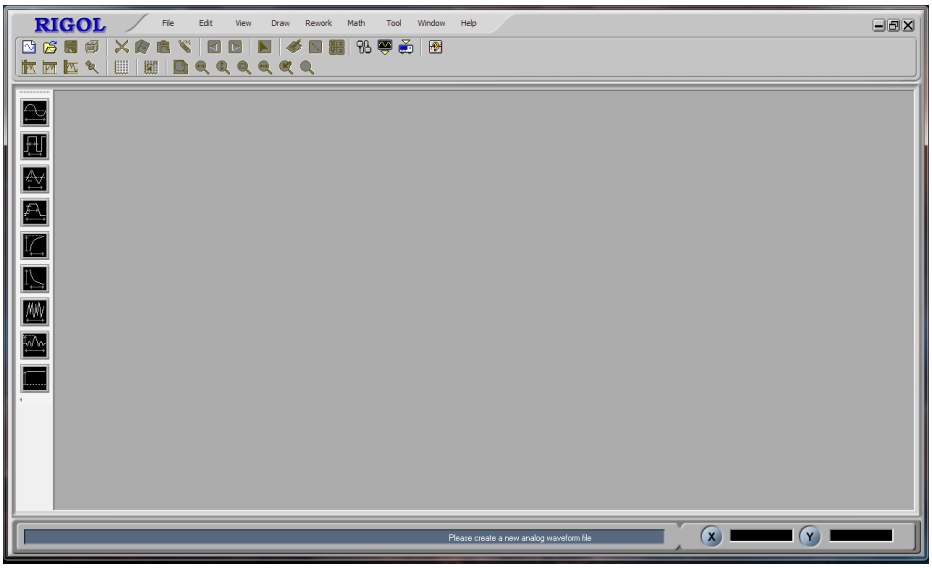


Figura 4.8 Ventana principal de Ultrawave.exe

El software Ultrawave.exe es de fácil instalación y demanda mínimos recursos de la PC. Es importante instalar también los controladores USB necesarios para lograr la conexión entre el Generador y la PC. El controlador viene incluido en el

CD del dispositivo, sin embargo también es posible descargarlo gratuitamente desde el siguiente sitio web: http://rigolusa.com/support/drivers.html .

Una vez instalado el programa, se genera un acceso directo en el escritorio de Windows. Haciendo doble clic se ejecuta Ultrawave.exe (figura 4.8).

Ultrawave.exe permite crear señales tipo seno, cuadrada, rampa, pulso, exponencial, ruido, sampling y DC. Adicionalmente es posible dibujar formas de onda arbitrarias directamente con el mouse. A continuación se mencionan características de los menús que brinda el software para la creación y edición de señales arbitrarias.

Archivo. Esté menú permite crear, guardar, abrir e imprimir archivos de formas de onda.

Edición. Este menú incluye funciones como; copiar, cortar, pegar, borrar, etc.

Visualización. Este menú auxilia para tener una mejor apreciación de las señales según lo requiera el usuario, incluye opciones como; cuadrícula, tipos de zoom, coordinación en tiempo, etc.

Dibujar. Este menú contiene la función para dibujar manualmente formas de onda.

Rehacer: En este menú es posible modificar los parámetros de la señal como frecuencia, resolución, amplitud, calidad, etc.

Matemáticas: Este menú contiene funciones matemáticas básicas para operar sobre un par de señales. También incluye un filtro pasa-bajas.

Herramientas: Este menú nos permite activar o desactivar la comunicación entre el Generador y la PC, así como almacenar formas de onda para ser habilitadas y enviadas.

Ventana: Este menú contiene algunas opciones para visualizar las ventanas abiertas, cuando se tiene abierta en el espacio de trabajo más de una señal.

Ayuda: Este menú incluye una guía de usuario y selección de idioma Inglés/Chino.

4.2. Creación de señales arbitrarias

Para crear señales arbitrarias primeramente se debe abrir un espacio nuevo de trabajo. Los parámetros necesarios son:

Puntos de resolución ó número de muestras: Este parámetro se refiere al número de muestras o puntos que se desea que tenga una determinada señal. Entre más puntos se asignen, mejor será la resolución de la señal en tiempo.

Vpp: Este parámetro es el voltaje pico a pico del espacio de trabajo y de igual forma de la señal que se genere.

Cuantificación: Este parámetro se refiere a los niveles de cuantificación con los que se desea que sean creadas las señales, normalmente se trabaja con 14-bits.

Periodo/Frecuencia: Este parámetro especificará el periodo ó frecuencia del espacio de trabajo, la señal generada se verá desplegada dentro de éste intervalo de tiempo.

En la figura (4.9) se observa un ejemplo con las siguientes características:

Número de muestras = 1000

Vpp = 10 Volts

Cuantificación = 14 bits

Periodo = 1 segundo

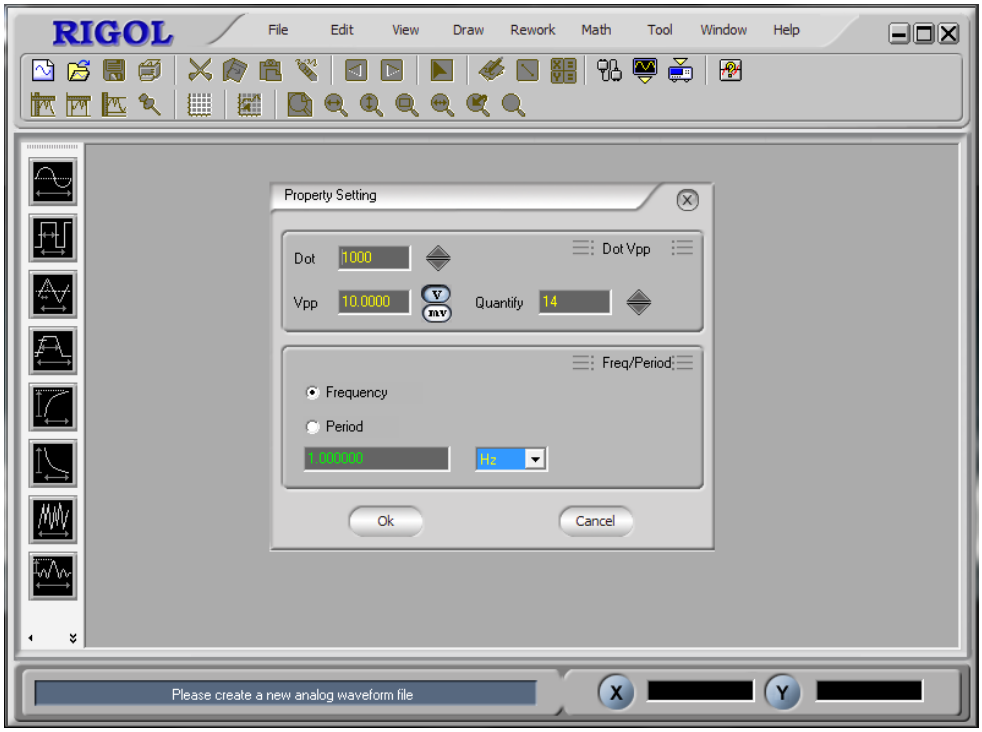
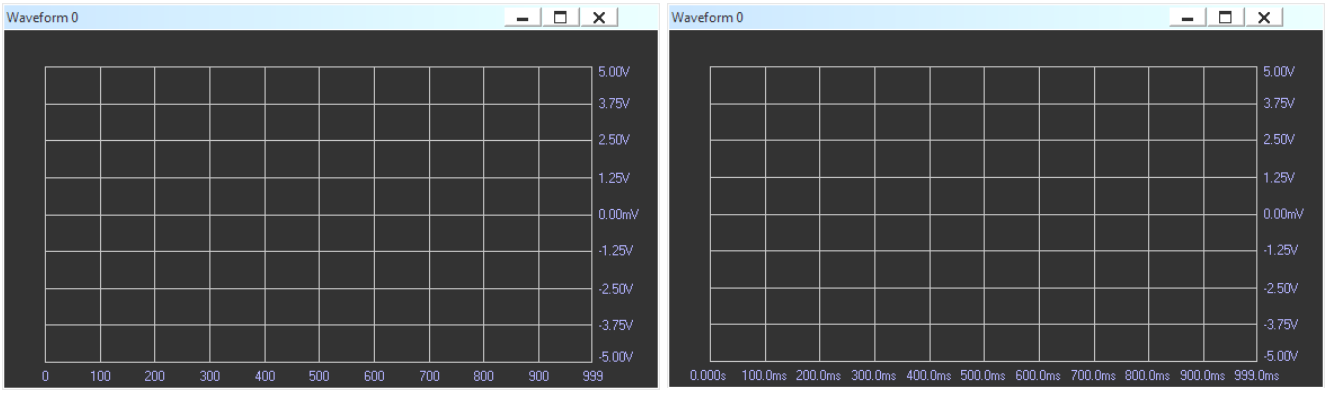


Figura 4.9 Parámetros iniciales de nuestro espacio de trabajo

Después de definir los parámetros se presiona “Ok” y se desplegará nuestro espacio de trabajo. Existen dos formas de observar el eje horizontal, puede ser respecto al número de muestras ó respecto al tiempo. Esta opción se encuentra dentro del menú “Visualización” .

En la figura 4.10 se puede observar que el eje horizontal de 1000 muestras (es decir de 0 a 999 muestras) es equivalente al eje de 0 a 999 mili segundos, que da como total 1 segundo, que anteriormente se ha definido como el periodo de trabajo. Esta equivalencia en tiempo indica que cada muestra tendrá un tiempo de separación igual a un mili segundo. Por lo tanto el tiempo entre muestras queda definido por la siguiente ecuación (4.1):



(a) (b)

Figura 4.10 Espacio de trabajo.
(a) Eje horizontal acotado en número de muestras. (b) Eje horizontal acotado en tiempo.

$$Tiempo\ entre\ muestras = \frac{Periodo\ de\ trabajo}{Número\ de\ muestras} \quad \textbf{(4.1)}$$

Para un periodo de 1 segundo se tendrá:

$$Tiempo\ entre\ muestras = \frac{1\ segundo}{1000\ muestras} = .001\ segundos\ entre\ muestras \quad \textbf{(4.2)}$$

Ahora bien, se encuentra listo el espacio de trabajo. Lo que sigue es crear alguna forma de onda, puede elegirse alguna función ya definida o dibujarla. En este caso se generará una señal senoidal con los siguientes parámetros, los cuáles serán introducidos en una venta como se muestra en la figura 4.11.

Tipo de señal: Seno

Número de muestras a utilizar del espacio de trabajo = 1000 muestras

Amplitud = 5 volts pico a pico

Offset = 0 volts

Número de ciclos a visualizar = 1

Fase = 0

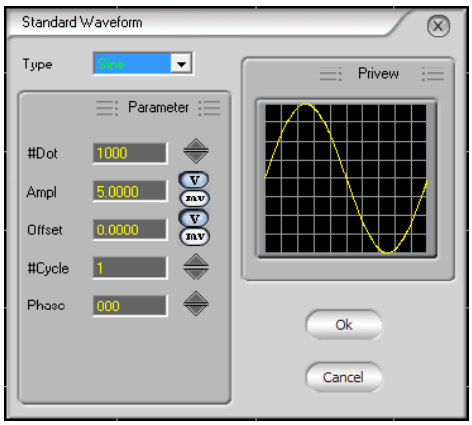


Figura 4.11 Representación de forma de onda y sus parámetros

Presionando “Ok” se desplegará la señal como se muestra en la figura (4.12).

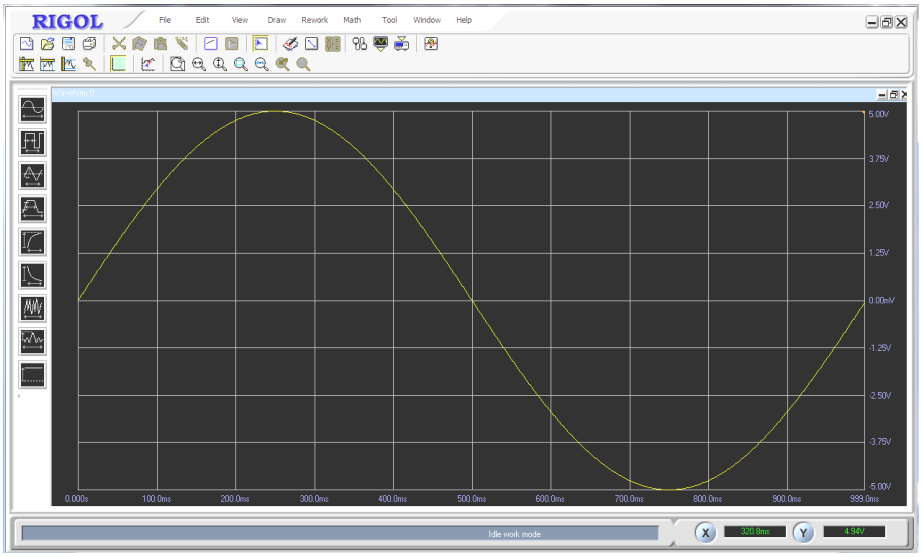


Figura 4.12 Señal generada sobre el espacio de trabajo

Un parámetro muy importante es el número de muestras que se desea asignar a la forma de onda a generar. Como máximo se tiene en este caso las 1000 muestras que se asignaron al espacio de trabajo. Pero se puede asignar menor cantidad de muestras, esto impactará en la duración de la onda y en la resolución

de la misma. La figura 4.13 muestra un ejemplo de la señal anterior ahora con 500 muestras.

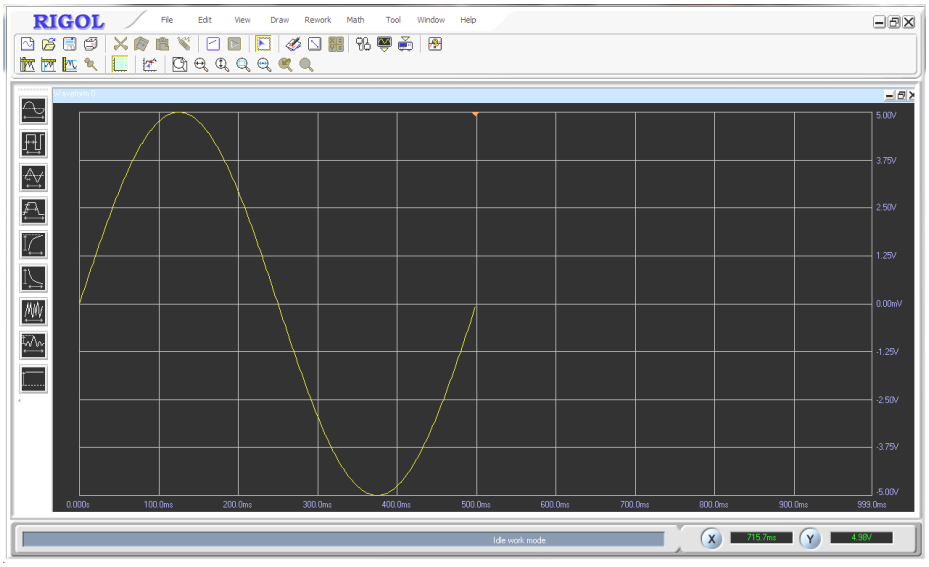
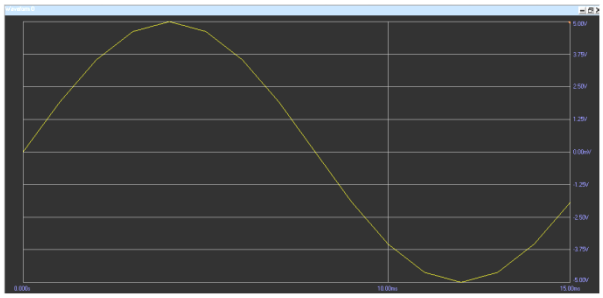
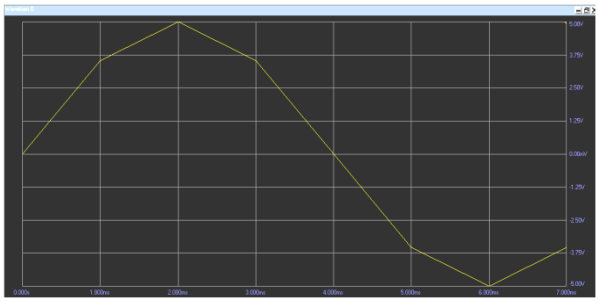


Figura 4.13 La duración de la señal es propocional a número de muestras

El ciclo de la onda durará la mitad, es decir 500 mili segundos. La forma de la onda y la duración se verán afectadas con forme se disminuya la resolución de la señal. La calidad de la señal se irá perdiendo, pues se contará con menos puntos para definirla y perderá precisión. En la figura 4.14 se observan un par de ejemplos con baja resolución.



(a)



(b)

Figura 4.14 (a) 16 puntos de resolición (b) 8 puntos de resolución

Se logra apreciar la diferencia entre ambas, pierden calidad y definición, su duración es menor pues es proporcional al número de muestras.

4.2.1. Guardar y abrir señales arbitrarias

Una vez generada la señal sobre el espacio de trabajo, se debe almacenar en la PC. Esto se puede hacer en tres formatos diferentes: “.txt” (archivo de texto), “.csv” (archivo de microsoft excel) y “.rdf” (archivo de forma arbitraria).

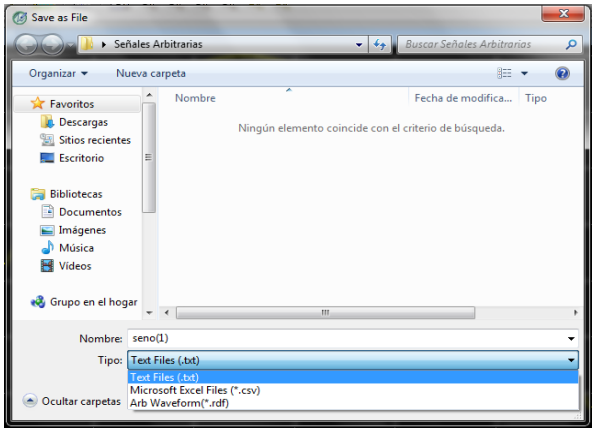


Figura 4.15 Opciones para almacenar formas de onda

Se debe asignar un nombre a la forma de onda, se sugiere que sea un nombre corto y fácil de identificar, también es conveniente crear un directorio específico para concentrar las formas de onda, véase figura 4.15. Por último se elige un tipo de archivo y dando clic en "guardar" la forma de onda será guardada satisfactoriamente. Ahora bien, se veran las diferencias entre los tipos de archivos.

Para este caso ".txt" y "csv", el archivo contedrá un formato como se muestran en las fiura 4.17 para tipo ".txt" y la figura 4.18 para tipo ".csv". Los archivos ".rdf" es un formato exclusivo de Ultrawave.exe.

Nota. Estos archivos corresponden a una forma de onda tipo seno con las mismas características que las anteriores propuestas, excepto los puntos de resolucón, se trabajará ahora sólo con 10 puntos para poder apreciar todos datos almacenados en los archivos. Los parametros se observean en la figura 4.16.

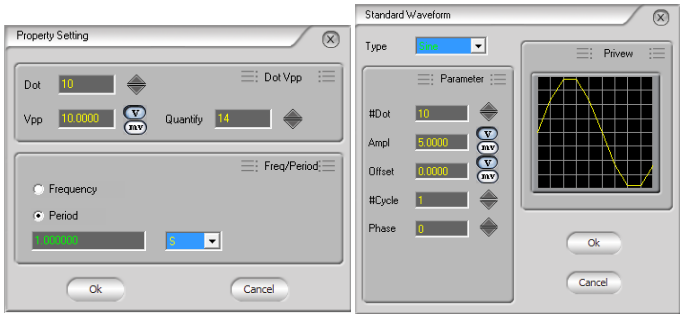


Figura 4.16 Parámetros introducidos para almacenar una nueva señal

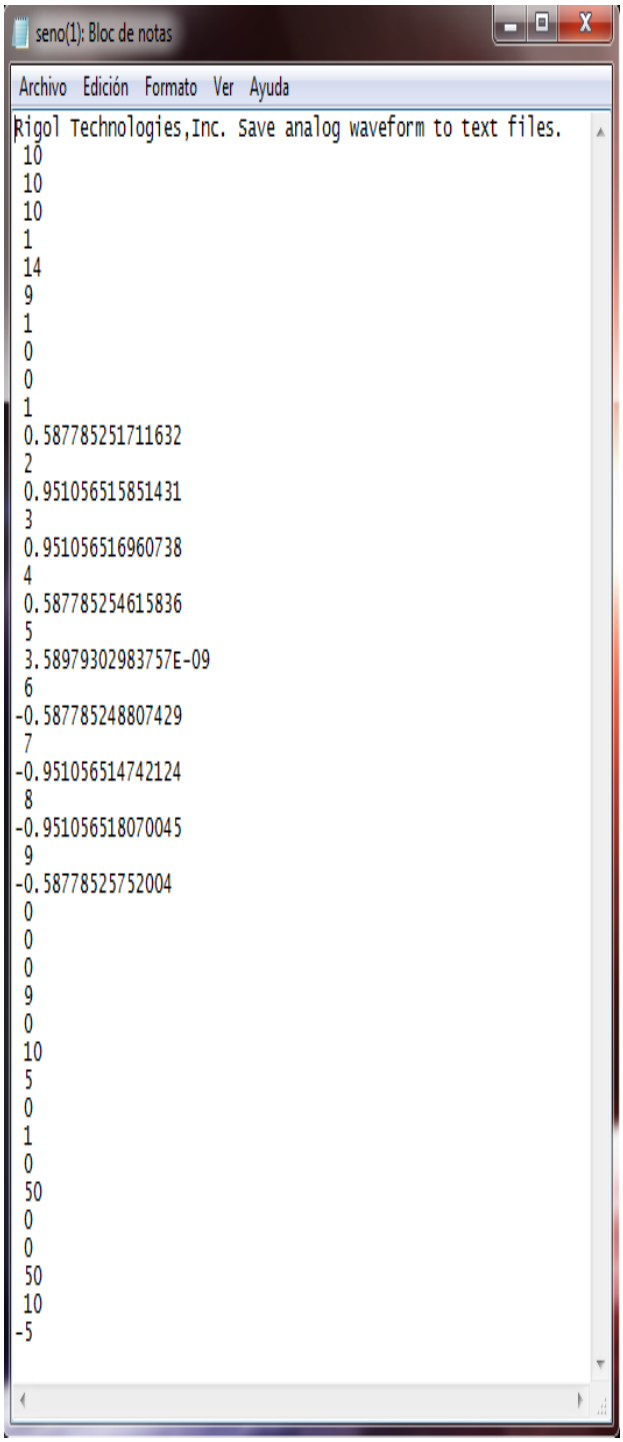


Figura 4.17 Archivo ".txt" que alamcena toda la información necesaria de la forma de onda.

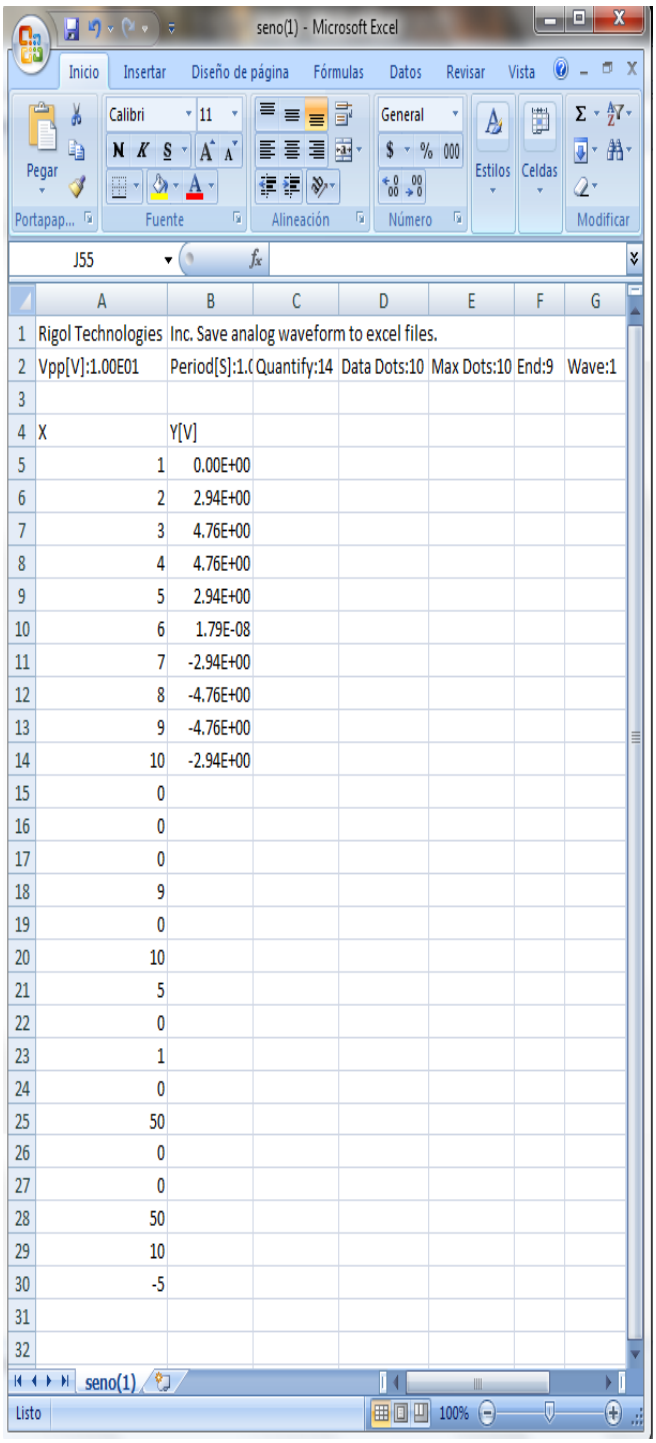


Figura 4.18 Archivo ".csv" que alamcena toda la información necesaria de la forma de onda.

También es posible abrir estos archivos con gran facilidad, siguiendo el menú Archivo>abrir>forma de onda arbitraria y seleccionando el archivo con el nombre que previamente ha sido almacenado dentro del directorio correspondiente. En la figura 4.19 muestra la ventana auxiliar para abrir un archivo de forma de onda que previamente se guardó. Dependiendo del tipo de archivo se debe elegir si se trata de un archivo de texto, una hoja de Microsoft Excel o tipo ".rdf". Al presionar "abrir" la forma de onda se muestra en pantalla.

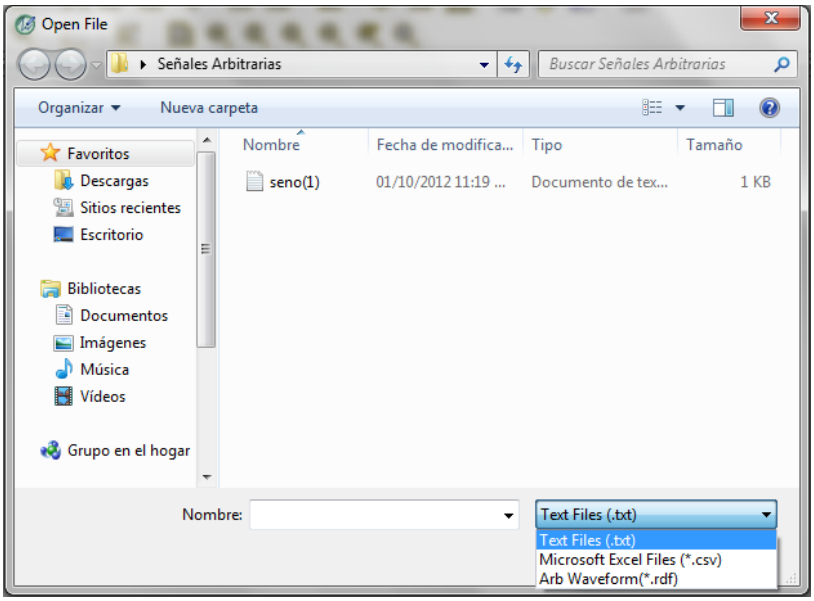


Figura 4.19 Abrir achivo

4.2.2. Formato de archivos de texto

Se ha visto que es posible almacenar señales arbitrarias en diferentes tipos de archivo, ".txt", ".csv" y ".rdf". Sin embargo sólo se analizará el formato que tienen los archivos de texto, pues son los que tienen aplicación para éste proyecto.

4.2.2.1. Cabeceras de archivos de texto

Los archivos de texto comienzan con unas cabeceras que son creadas por el software y a su vez son necesarias para que los archivos sean válidos y puedan ser abiertos. Si las cabeceras no cuentan con el formato adecuado, el software lo marcará como archivo inválido y será imposible abrir formas de ondas.

A continuación se describen las cabeceras principales y los parámetros contenidos en el archivo de texto generado, tomando como ejemplo el caso anterior (figura 4.17).

"Rigol Technologies,Inc. Save analog waveform to text files." Este enunciado indica al software que la forma de onda será guardada en un archivo de texto.

"10" Indica el número de muestras total del espacio de trabajo.

"10" Indica el número de muestras utilizadas para la creación de la señal. Este parámetro puede ser menor o igual al segundo parámetro.

"10" Este parámetro representa el valor de voltaje pico a pico de la señal.

"1" Indica la duración del periodo de trabajo en segundos.

"14" Representa los niveles de cuantificación.

“9” Número de muestras a partir de la muestra 1, es decir e igual al número de muestras menos uno.

“1” Indica el número onda.

Después de estos valores se enlistarán cada una de las muestras ordenadas desde la muestra cero hasta la última muestra, con el siguiente orden:

“0” Número de muestra.

“0” Valor de la muestra

“1” Número de muestra

“0.587785251711632” Valor de la muestra.

Y así sucesivamente hasta la última muestra. Posteriormente se imprimen una serie de números que son indicadores propiamente del software, de manera que estos parámetros en su mayoría serán siempre los mismos sin importar las características de la forma de onda.

“0” Indicador de software.

“0” Indicador de software.

“0” Indicador de software.

“9” Número de muestras a partir de la muestra 1

“0” Indicador de software.

“10” Indica el número de muestras utilizadas para la creación de la señal.

“5” Indicador de software.

“0” Indicador de software.

“1” Número de ciclos generados en el espacio de trabajo.

“0” Indicador de software.

“50” Indicador de software.

“0” Indicador de software.2

“0” Indicador de software.

“50” Indicador de software.

“10” Indicador de software.

“-5” Indicador de software.

Conocer este formato permite editar señales sin hacer uso de las herramientas del software, es decir, es posible crear señales con la ayuda de otro tipo de software y almacenar todos los parámetros necesarios con la estructura anterior dentro de un archivo de texto, para que después pueda ser abierta en Ultrawave.exe.

4.2.3. Generación eléctrica de señales arbitrarias

Para generar las señales creadas por el software ultrawave.exe es necesario que se realice la conexión del dispositivo Rigol vía USB con la PC. Para esto requiere conectar el cable en los puertos USB respectivamente. Posteriormente se debe seleccionar el menú “herramientas” y seleccionar la opción que se muestra en la figura 4.20.

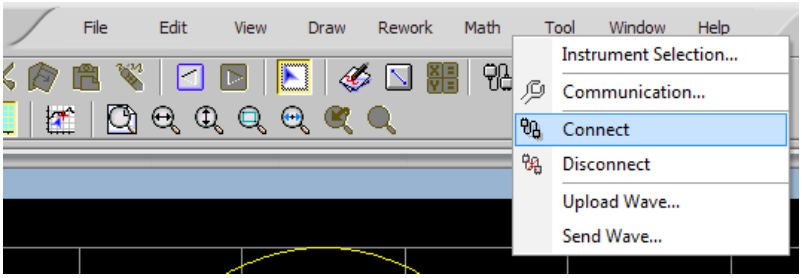


Figura 4.20 Realizar la conexión al generador via USB

Una vez que el software haya logrado conectarse con éxito y teniendo lista la señal a generar sobre el espacio de trabajo se selecciona la opción de la figura 4.21. Esto desplegará una ventana que muestra las localidades de memoria disponibles para almacenar y enviar señales arbitrarias. Se requiere elegir una localidad de memoria para cada señal. Es importante tomar en cuenta que sólo existen 4 localidades, de tal manera que si se encuentran ocupadas todas se deberá borrar alguna para liberar espacio y ocuparlo. También se le asigna un nombre a la señal que será guardada en alguna localidad, presionando el botón "guardar" quedará almacenada en la localidad elegida. Este procedimiento está descrito en la figura 4.22. Sólo resta elegir la señal guardada para activarla y visualizarla con la ayuda de un osciloscopio, véase figura 4.23.

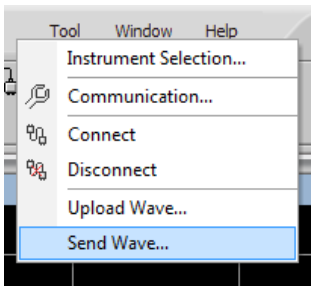


Figura 4.21 Enviar señal

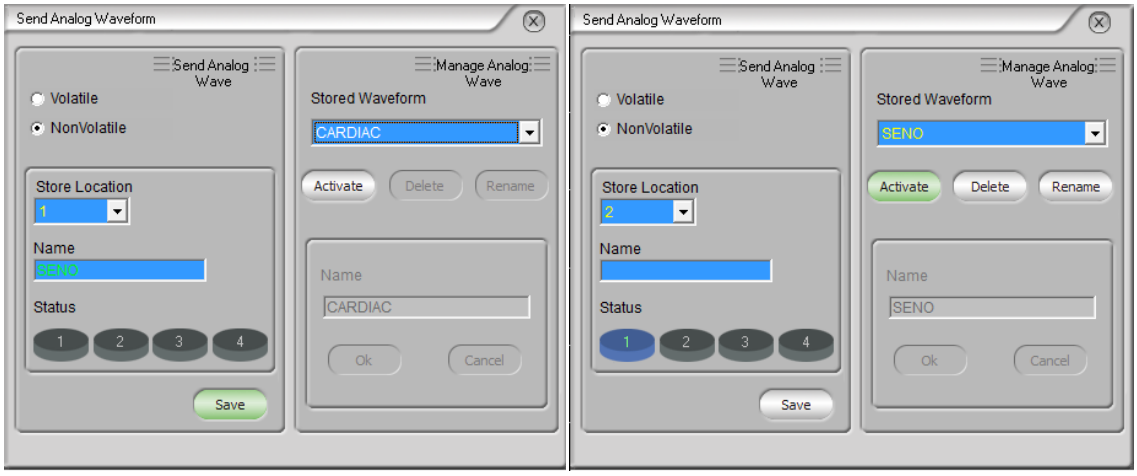


Figura 4.22 Procedimiento para almacenar en memoria señales arbitrarias y puedan ser generadas electricamente.

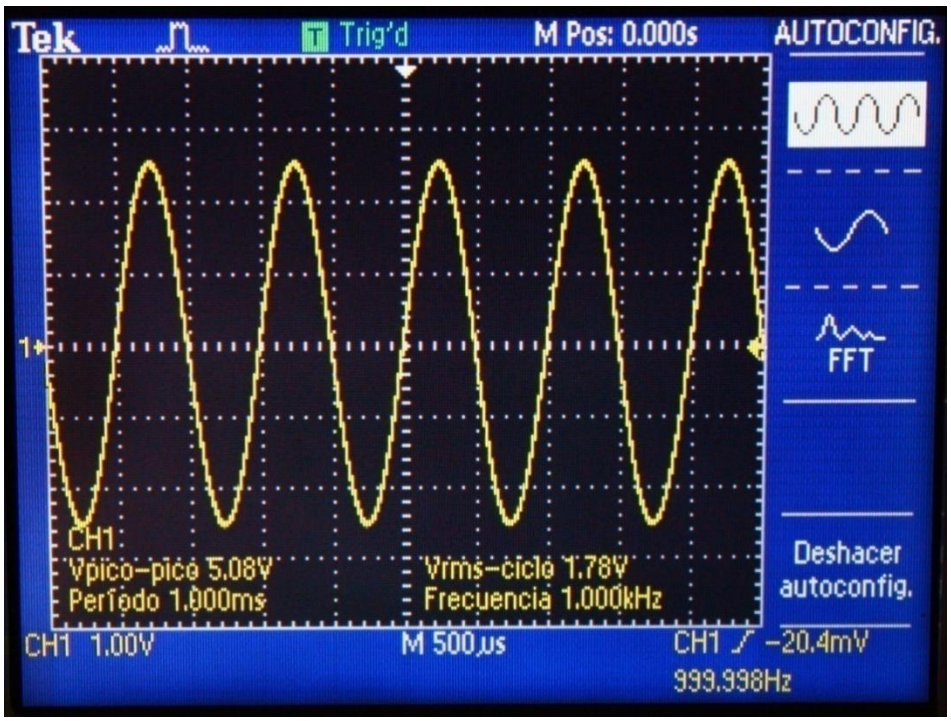


Figura 4.23 Señal generada eléctricamente y de forma periódica**.**

4.3. Implementación de una interfaz gráfica mediante MATLAB.

Es posible generar señales arbitrarias eléctricamente con el generador Rigol, también es posible crear estas señales por medio de archivos de textos. Haciendo uso de Matlab se ha desarrollado una interfaz gráfica compuesta por un generador de secuencias aleatorias de bits y diversos elementos de codificación.

Se analizará el desarrollo y funcionamiento de cada bloque implementado dentro de la interfaz gráfica. De esta manera se pretende que sea para el usuario una herramienta útil dentro del laboratorio de comunicaciones.

4.3.1. Generador de secuencias de bits

Como elemento principal, se ha diseñado un generador de secuencias de bits, que tiene como objetivo crear una forma pseudo-aleatoria de secuencias binarias compuestas por los símbolos “0” y “1”. Este generador deberá estar en función de otro elemento de codificación para poder ser representado gráficamente y eléctricamente.

4.3.2. Fuente binaria

Los parámetros principales necesarios para generar una secuencia de bits son; el número de símbolos a generar y la probabilidad con la que se desea que ocurra alguno de ellos.

Es decir, tratándose de dos símbolos, {“0” y “1”} que son dos posibles eventos que pueden ocurrir, esto forma el espacio muestra. Ahora bien, se creará una función en Matlab para efectuar un proceso aleatorio.

Se utilizará la función “rand”, que devuelve un número entre 0 y 1, ocupando 4 cifras significativas después del punto decimal. Por ejemplo, para almacenar números aleatorios dentro de una variable con una distribución uniforme, se tendría lo siguiente.

Variable U=**rand;**

U=0.8909
U=0.3245
U=0.3465
U= valor aleatorio entre 0 y 1.

De esta manera es posible obtener una distribución uniforme entre 0 y 1. Gráficamente se puede verificar mediante la figura 4.24. Dónde se han graficado 10,000 muestras con la función “rand” y se muestra el histograma, que verifica la distribución uniforme que sigue entre valores de 0 a 1.

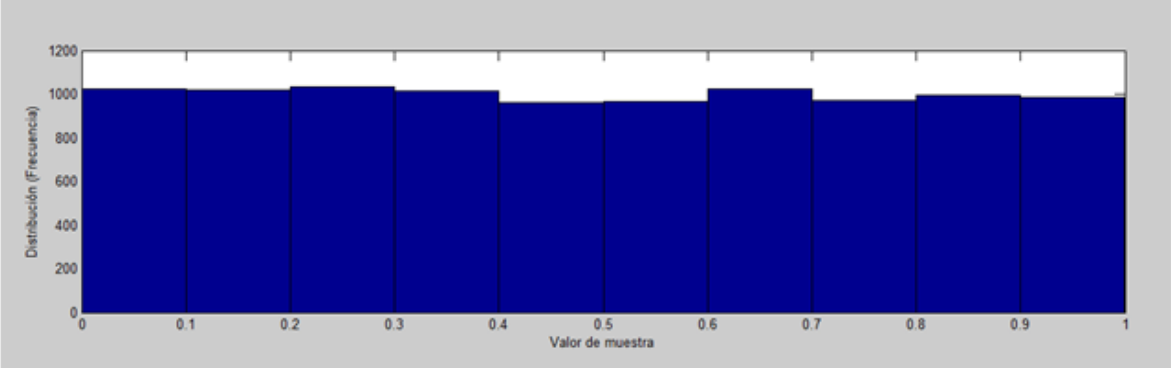


Figura 4.24 Histograma de la función “rand” a 10,000 muestras.

Como se trata de dos símbolos diferentes a generar (0 y 1), se pretende que aparezcan con diferente probabilidad, para que en algunos casos se observe mayor número de “0’s” o mayor número de “1’s”. Si se espera que ambos tengan la misma probabilidad de suceder se refiere a que cada símbolo tendrá una probabilidad del 50% ó 0.5. La probabilidad está definida desde un valor cero, que indica que no sucede ningún evento, hasta el valor de la unidad, que se refiere a que el evento es seguro.

Las probabilidades entre el par de símbolos (0 y 1) se pueden complementar conservando la unidad. Es decir, proponiendo una probabilidad de 0.7 para símbolo “0” entonces la probabilidad complementaria sería 0.3 para el símbolo “1”.

Para definir los símbolos con respecto a su probabilidad, se hará uso de la función “rand” que se encargará de proveer valores aleatorios que serán comparados con la probabilidad de alguno de los símbolos, y así definir qué símbolo será generado, este algoritmo se describe con el siguiente diagrama de flujo de la figura 4.25.

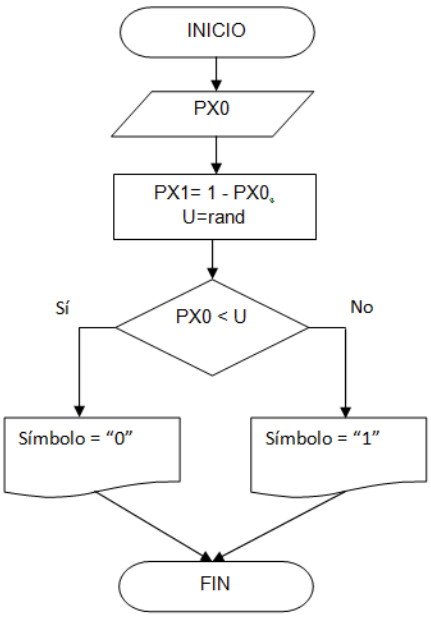


Figura 4.25 Diagrama de flujo para la fuente binaria

El diagrama de flujo anterior comienza por recibir el valor de “PX0” que representa la probabilidad que ocurra el símbolo “0”. A continuación se realiza la operación para determinar la probabilidad del símbolo “1” y a su vez asigna a la variable “U” un valor aleatorio entre 0 y 1. Después se comparan los valores PX0 y U, de tal manera que si PX0 es menor al valor generado en U el resultado será arrojar el símbolo “0” y en caso contrario el símbolo “1”. De esta manera es como se ha desarrollado una fuente binaria aleatoria. Es importante que este procedimiento esté contenido en un bucle repetitivo, para que cada vez que se ejecute se genere un símbolo de manera aleatoria.

Se ha implementado el siguiente código en Matlab, para generar un número variable de símbolos de forma aleatoria:

Función que define los símbolos

```
function[X]=fuente_binaria(PX0,PX1);

U=rand;   %genera un numero aleatorio entre 0 y 1

if U<PX0,

    X=0;

else

    X=1;

end;
```

Programa principal que manda llamar la función anterior.

```
PX0=0.5;

N=10;

PX1=1-PX0;

for k=1:N,

    [X]=fuente_binaria(PX1,PX2);
```

```
    Xi(k)=X;
end;
Xi
```

En este código se realiza de forma cíclica la generación de símbolos. El valor de la variable N determina el número de símbolos que se desean generar, estos símbolos son almacenados en un arreglo lineal ó vector denominado “Xi”

Como resultado se tendrán algunos casos como estos:

Xi= [1 0 1 1 0 0 1 1 0 0]

Xi= [1 1 0 1 0 0 0 1 1 0]

Xi= [0 0 0 1 1 1 1 1 0 0]

Esta rutina como se ha visto genera una secuencia de bits aleatoria, dependiendo de la probabilidad asignada a algún símbolo. El número de símbolos, es decir el tamaño de la secuencia generada y la probabilidad de símbolo son parámetros que el usuario debe asignar.

4.3.3. Graficar secuencias aleatorias de bits

Es necesario visualizar las secuencias de bits de forma gráfica, para esto se utilizará un método sencillo de representación llamado codificación unipolar. Como se ha visto en el capítulo anterior, la forma de representar dos símbolos diferentes con una señal de voltaje con respecto al tiempo es como se muestra en la figura 4.26.

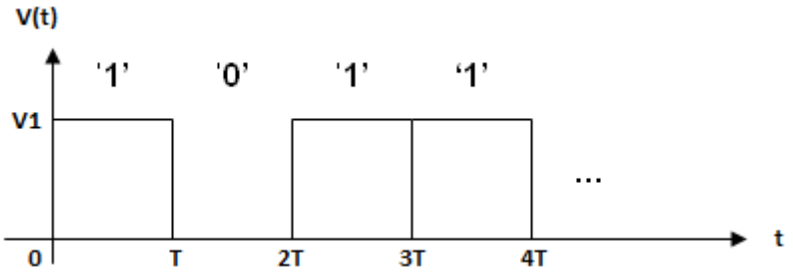


Figura 4.26 Representación unipolar.

Cada símbolo es representado con un nivel de voltaje (V1) durante un periodo de tiempo, denominado “T” que indica la duración del pulso. La presencia de este pulso dentro del intervalo “T” determina el símbolo “1”, mientras que la ausencia de este pulso indica que se trata del símbolo “0”.

Ahora bien, haciendo uso de los recursos de Matlab, es posible generar estas gráficas de diferentes maneras. La más sencilla y práctica es con el uso de la función “stairs ( )” y “stem ( )”, que al igual que la función “plot ()” brinda la facilidad de graficar la correspondencia entre dos conjuntos de valores o arreglos sobre un plano.

4.3.3.1. Uso de *stairs* y *stem* en GUI de MATLAB

Se ha implementado una sencilla interfaz gráfica dónde se grafica una secuencia de bits, de dos maneras diferentes, de manera continua y discreta en tiempo.

La figura 4.27 representa el diseño de un proyecto en GUI Matlab. En este diseño, se colocó un “Axes”, que es un espacio para graficar funciones, acompañado de dos campos de edición que son utilizados para que el usuario introduzca valores y dos “botones” que ejecutan una rutina en específico cuando son seleccionados. Los campos de edición están acompañados de dos textos estáticos que describen los textos de edición.

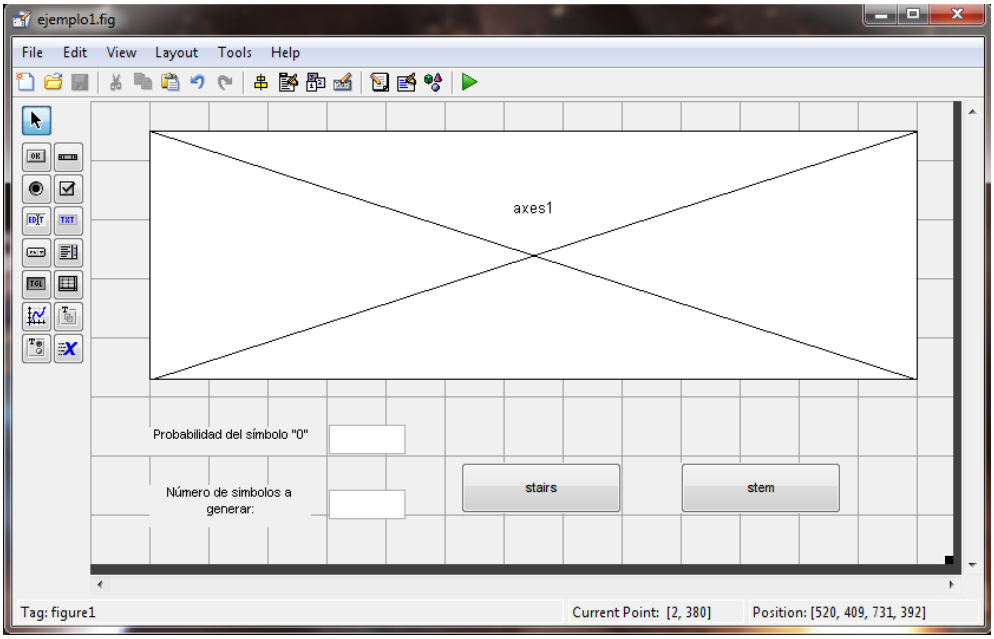


Figura 4.27 Prototipo en GUI Matlab para “stairs” y “stem”

La rutina del botón “stairs” consiste en graficar la secuencia de bits de forma continua en tiempo, tomando los valores introducidos en los campos de edición, es decir probabilidad y número de bits ó símbolos.

La función stairs retiene el valor que se la asigne (para este caso igual a 1) dependiendo del incremento en tiempo que se defina, para este caso el incremento (periodo) de tiempo es la unidad. Véase figura 4.28.

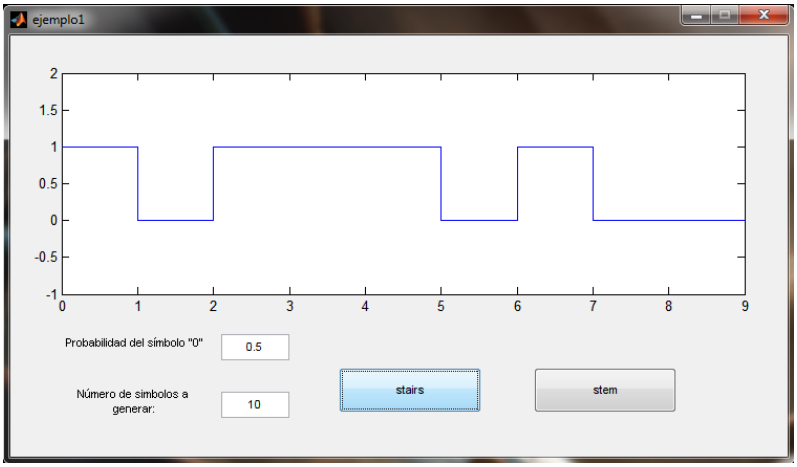


Figura 4.28 Función “stairs”

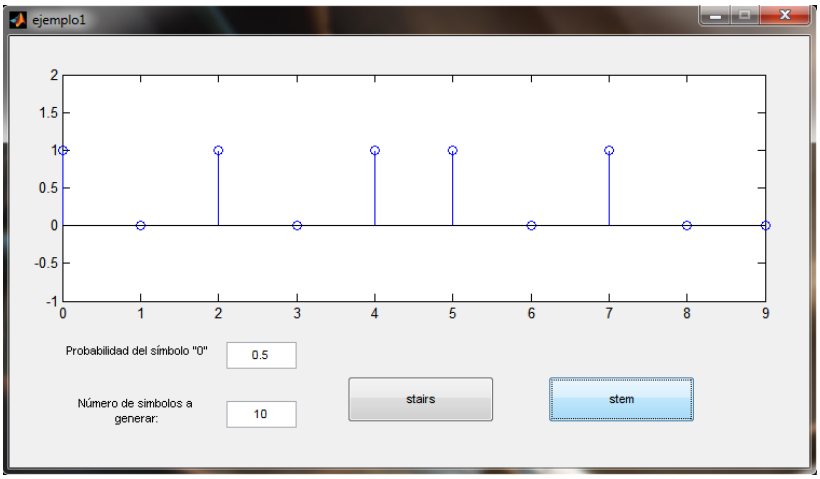


Figura 4.29 Función “stem**”**

Por otro lado la figura 4.29 representa una forma discreta en tiempo de representar los bits, esto se logra con la función “stem”. Recordando que estas secuencias se generan de forma aleatoria cada vez que se ejecutan las rutinas de tiempo a través de los botones, por esta razón las secuencias serán diferentes.

En los casos anteriores se han graficado secuencias de 10 símbolos, pero utilizando este método de representación gráfica es posible generar hasta 80,000 símbolos.

Como se ha visto, la forma de graficar estas secuencias se puede decir que son de forma automática, es decir, no existe un control de las muestras que se generan ó de la resolución de la misma. Pues las funciones “stem” y “stairs” ya están definidas por el compilador de Matlab.

4.3.3.2. Uso de *plot* en GUI de MATLAB

La función “plot” permite graficar la relación entre dos variables sobre un plano. A diferencia de “stairs” y “stem”, la función “plot” se encarga de unir con una línea cada uno de los puntos sobre la gráfica. Esto significa que entre más puntos se utilicen para una gráfica, esta tendrá mejor resolución y podrá tomar cualquier forma continua sobre el eje horizontal.

La sintaxis básica de la función “plot” es la siguiente:

plot (x,y);

dónde;

“x” es un vector ó arreglo que contiene los valores sobre el eje horizontal (vector tiempo).

“y” es un vector ó arreglo que contiene los valores sobre el eje vertical (valores de amplitud).

De esta manera, para graficar señales, el vector tiempo debe estar dividido en incrementos iguales, dónde cada unos de ellos será correspondido a un y solo un valor de amplitud. El tamaño de los arreglos ó variables deben tener el mismo tamaño o longitud. Por ejemplo, para una señal con 300 valores de amplitud, se deben definir exactamente 300 valores de tiempo.

Se puede observar que con la función “plot” se tiene un control preciso del número de muestras. Esto es de gran utilidad, pues será posible generar señales arbitrarias mediante archivos de texto que contengan el número de muestras, entre otras características de la señal.

Para graficar secuencias de bits, utilizando el tipo de representación unipolar se utilizará el siguiente procedimiento.

Resolución de bit. Se definirá la resolución de bit ó pulso, a la cantidad de muestras utilizadas para formar un bit ó símbolo.

Para graficar un pulso, haciendo uso de “plot” se requiere varios puntos para mantener la línea en un mismo nivel. Por ejemplo, en la figura 4.30 se especifica la forma de representación de un bit o símbolo, ocupando 10 puntos o muestras para formarlo. El pulso tiene una duración “T” y como resultado tendría que, el tiempo o separación entre muestras está dado por la siguiente expresión.

$$Tiempo\ entre\ muestras = \frac{Duración\ del\ pulso\ (T)}{Número\ de\ muestras} \quad \textbf{(4.3)}$$

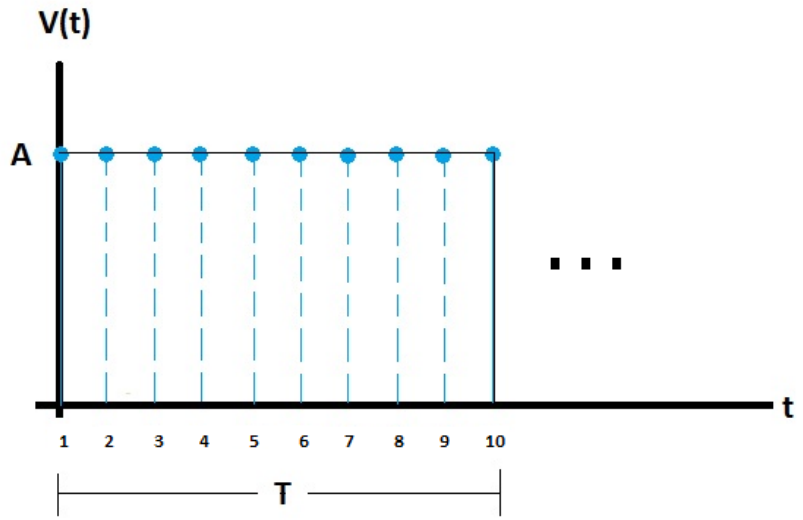


Figura 4.30 Representación de bit con 10 muestras.

Se observa que a diferencia de hacer uso de las funciones “stem” y “stairs”, con “plot” los pulsos se deben formar con una serie de puntos.

Haciendo uso de las herramientas de GUI Matlab, se ha complementado el prototipo anterior (figura 4.27) con un nuevo botón que ejecuta una nueva rutina haciendo uso de “plot”, siguiendo el siguiente algoritmo de programación.

Datos de entrada: (PX0) Probabilidad de ceros y (N) número de símbolos a generar.

Datos de salida: (Xi) Secuencia aleatoria binaria de longitud (N).

De acuerdo con la figura 4.30 se deben asignar 10 puntos para cada símbolo. De esta forma, para una secuencia aleatoria dada Xi = [1 0 1 0 0 1 0 1 1 1]; se asignarán las muestras de la siguiente manera:

Mediante un ciclo “for” se leerá la secuencia “Xi” y con la ayuda de un “if” se llevará a cabo el reconocimiento se los símbolos “0”y “1” para asignarles a cada uno, una pequeña secuencia de muestras, llamadas contenedores, estos serán arreglos de la siguiente forma, para 10 muestras:

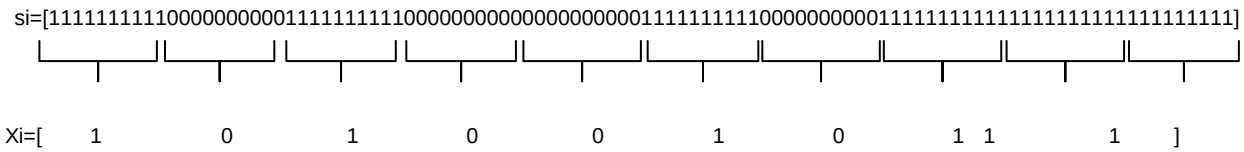
sj=[0, 0, 0, 0, 0, 0, 0, 0, 0, 0] y sj=[1,1, 1, 1, 1, 1, 1, 1, 1, 1]

Cuando sea leída la secuencia “Xi” y el símbolo encontrado sea “0”, se usará el contenedor sj=[0, 0, 0, 0, 0, 0, 0, 0, 0, 0] y lo será concatenado dentro de un contenedor más grande definido como un arreglo vacío “s=[ ]. Por otro lado si el símbolo siguiente encontrado es “1”, se usá el contenedor sj=[1,1, 1, 1, 1, 1, 1, 1, 1, 1] y será concatenado con el anterior dentro del arreglo de salida si=[ ].

Siguiendo este procedimiento, a partir de la secuencia de entrada;

Xi = [1 0 1 0 0 1 0 1 1 1]

Se tendrá a la salida la siguiente secuencia:



Se muestra la relación de la secuencia original, con la de la salida “si”. La secuencia “si” será ahora la secuencia utilizada para graficar. Se graficará siguiendo la sintaxis de la función “plot”, dónde se debe crear un vector de tiempo, el cual estará en función de la cantidad de elementos de “si”. Para este caso “si” tiene 100 muestras, dónde cada 10 representan un bit.

Para un periodo de duración T=1, la separación entre estas muestras, es decir el incremento del vector tiempo será:

$$Tiempo\ entre\ muestras = \frac{1}{10} = 0.1 \qquad \textbf{(4.4)}$$

La figura 4.31 muestra el resultado de graficar mediante la función “plot” la secuencia “si”. Se puede observar que la resolución no es la óptima pues se alcanza a apreciar la subida y bajada de los pulsos. Sin embargo se le puede asignar cualquier resolución, tomando en cuenta la relación del periodo y número de muestras. La figura 4.32 muestra una secuencia con 100 muestras por símbolo, lo que da como resultado incrementos de 0.01 unidades del vector tiempo, para conservar nuestro periodo unitario. Con esta nueva resolución se aprecia una mejor calidad de la señal.

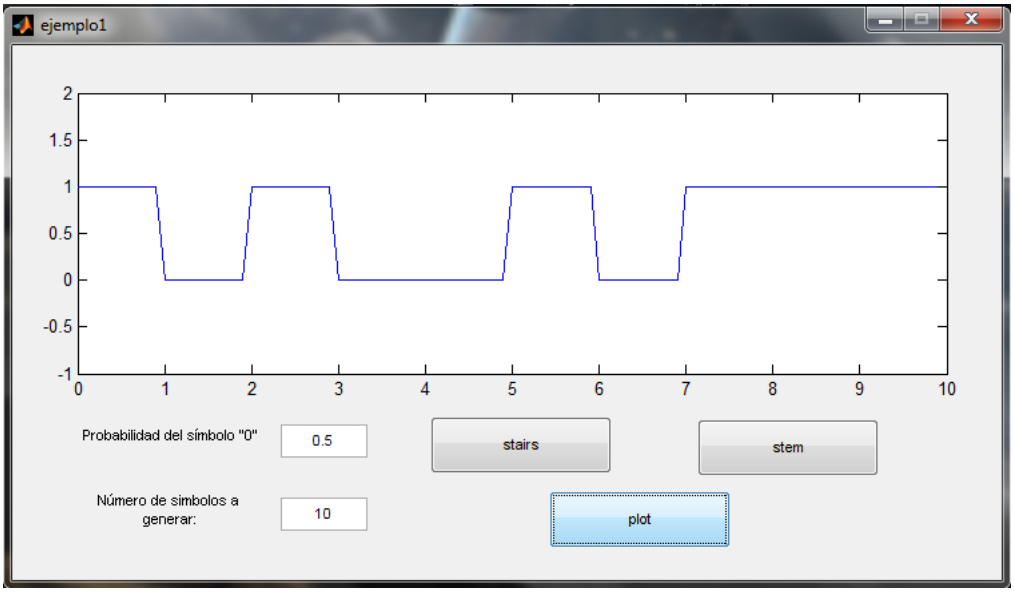


Figura 4.31 Función “plot” – 10 muestras por símbolo.

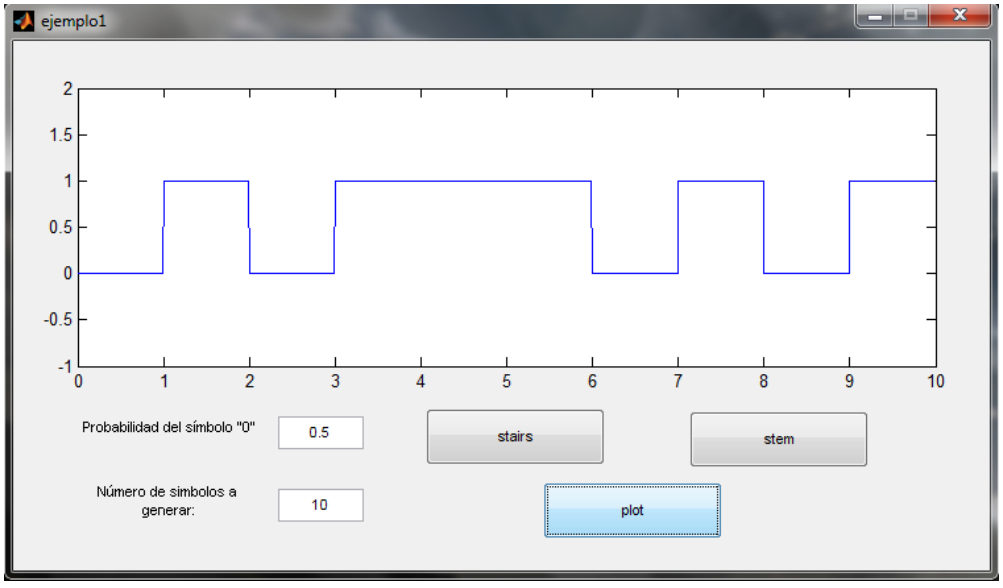


Figura 4.32 Función “plot” – 100 muestras por símbolo

4.3.4. Características eléctricas de las secuencias de bit

Se ha descrito la manera de generar y graficar secuencias bits mediante Matlab, dónde se han utilizado valores unitarios de amplitud y duración en los pulsos. Sin embargo las secuencias de bits deben tener las siguientes características eléctricas:

Amplitud [Volts]

Periodo o tiempo de bit [Segundos]

Adicionalmente es necesario definir un número de símbolos a generar y la probabilidad con el que se desean generar.

4.3.4.1. Tiempo de bit o duración del pulso

Para definir el tiempo de bit y acotar adecuadamente el eje horizontal se debe retomar la ecuación (4.1), que nos describe la relación del periodo con el número de muestras. Basta simplemente con definir un periodo deseado y un número de muestras por símbolo para conocer el incremento del vector tiempo que se debe manejar, para tener como resultado la duración de símbolo deseada..

Por ejemplo, para el caso de 100 muestras por bit y proponiendo un periodo de 10 mili segundos se tendrá como se muestra en la ecuación (4.5):

$$Tiempo\ entre\ muestras = \frac{Duración\ del\ pulso\ (T)}{Número\ de\ muestras} = \frac{10\ mili\ segundos}{100\ muestras} = 0.0001 \quad \textbf{(4.5)}$$

Se requerirá manejar un incremento de 0.0001 segundos entre cada muestra. Esta operación se ha implementado dentro del prototipo de prueba, anexando un texto de edición, el cuál servirá para capturar el tiempo de bit que se desea, mientras que dentro de la rutina que grafica la secuencia se implementó la ecuación (4.5).

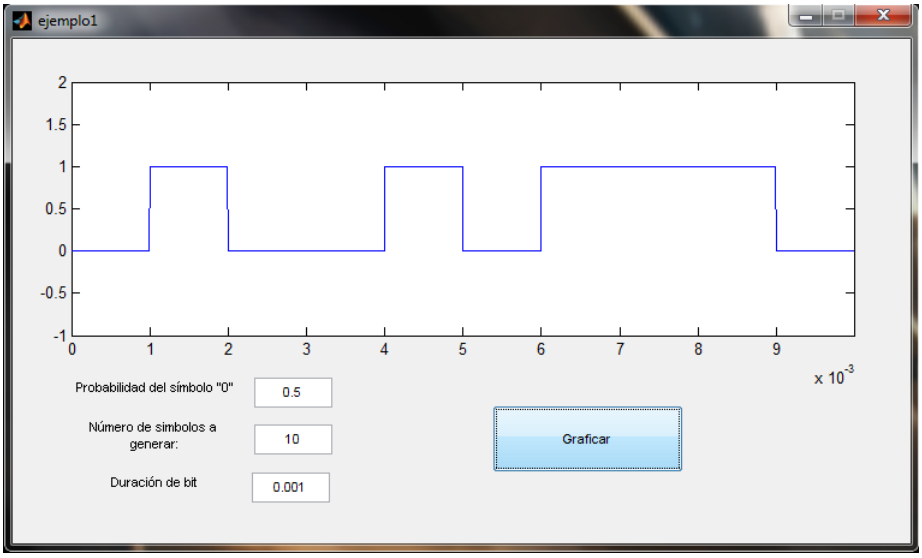


Figura 4.33 Función "plot" – 100 muestras por símbolo y duración variable definida por el usuario.

Como resultado se observa que la duración de bit ahora es manipulable. Véase figura 4.33. Este prototipo grafica la secuencia utilizando la función plot, ocupando 100 muestras por bit.

Amplitud del pulso.

El siguiente parámetro eléctrico importante es la amplitud del pulso, que será representada en volts. Es sencillo manipular la amplitud de la secuencia de bits, ya que está representada por “1’s” y “0’s”, de esta manera se puede escalar de forma directa el valor de cada muestra. Por ejemplo para asignar una amplitud a una secuencia basta con multiplicar todo el arreglo generado por aquel valor de amplitud que se desea tener.

Secuencia generada:

si=[0000000000000000000011111111111111111111000000000011111111111111111111000000000011111111110000000000]

Secuencia escalada en amplitud con un factor de 5:

5 x si=[0000000000000000000055555555555555555550000000000555555555555555555500000000005555555555000000000]

Aplicando esta operación a la rutina para graficar y anexando otro cuadro de edición para capturar la amplitud deseada, el prototipo queda de la siguiente manera, mostrando el ejemplo de la figura 4.34.

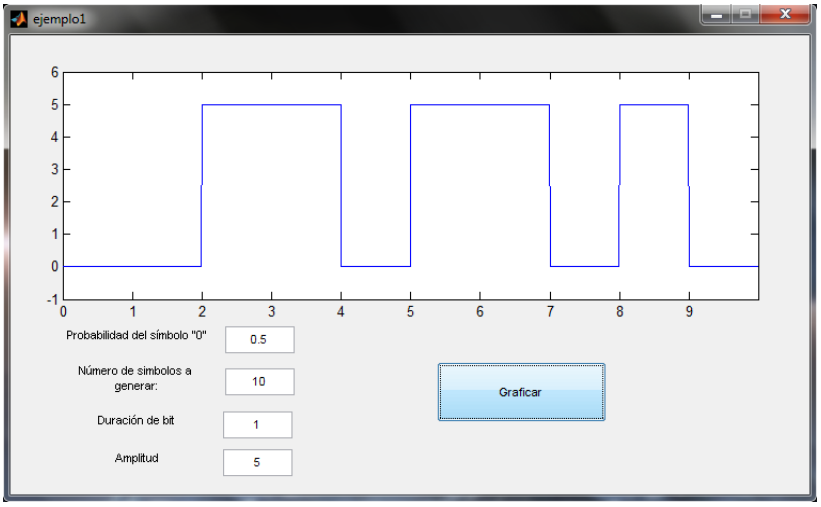


Figura 4.34 Función “plot” – Implementando amplitud variable**.**

4.3.5. Creación de archivos de texto basados en secuencias de bits

Uno de los objetivos principales de este trabajo es representar eléctricamente las secuencias de bits generadas mediante una interfaz gráfica en Matlab. Como se vió anteriormente en el tema 4.2.2 es posible editar señales arbitrarias mediante archivos texto, para más tarde generarlas eléctricamente a través del Generador RIGOL.

Partiendo del prototipo que se ha diseñado, el cual genera y grafica una secuencia aleatoria de bits de acuerdo a diferentes parámetros como: probabilidad de símbolo, número de símbolos, duración de bit y amplitud, se propuso un algoritmo de programación para registrar todos estos parámetros y ortos más, dentro de un archivo de texto, siguiendo el formato compatible con el Software Ultrawave.exe.

El formato está compuesto principalmente por tres encabezados formados por, parámetros propios de la señal, muestras e indicadores de software. Para ello definirá un arreglo para cada encabezado. De tal manera que se tienen los siguientes arreglos:

Arreglo para el primer encabezado:

```
encabezado1=[M M 2*A T1 14 M-1 1];
```

donde las variables representan lo siguiente,

M=número de muestras

A =amplitud

T1=duración de la secuencia

Arreglo para el segundo encabezado:

```
encabezado2=[sn;si];
```

donde;

sn= numeración del 0 al último número de muestras que se utilicen.

si=arreglo que contiene la secuencia de bits.

Arreglo para el tercer encabezado.

```
encabezado3=[0 0 0 M-1 6 M 1 0 1 0 50 0 0 50 10 5];
```

Este arreglo contiene en su mayoría indicadores de software ya predeterminados.

Se requiere imprimir este contenido dentro de un archivo de texto, y para ello se ha definido la siguiente rutina en matlab;

```
%código para crear un directorio en C:/

PathCurrent = ('C:/work');

FolderName = ['streams'];

PathFolder = [PathCurrent '/Interface/' FolderName];

mkdir([PathCurrent '/Interface/'], FolderName);

NameFile = [PathFolder '/ejemplo1.txt'];

fid = fopen(NameFile,'wt');

fprintf(fid,'%s\n','Rigol Technologies,Inc. Save analog waveform to text
files.');

fprintf(fid,'%6.0f\n  %12.15f\n',encabezado1);

fprintf(fid, '%6.2f\n %12.8f\n',encabezado2);

fprintf(fid,'%6.0f\n  %12.0f\n',encabezado3);
```

A su vez el código anterior comienza por generar un directorio en la unidad de almacenamiento “C”, dónde creará la carpeta “Interface” y dentro una carpeta llamada “streams”, dónde alojará el archivo de texto creado. El nombre que se

asigna al archivo de texto es “ejemplo1.txt”. Esta rutina se ejecuta cada vez que se presione el botón de “Graficar”, de tal forma que el archivo se actualizará con los parámetros de la última señal generada.

La figura 4.35 muestra un ejemplo de una secuencia generada con 10 símbolos. Se imprimen un total de 1000 muestras. Ya que cada símbolo contiene 100 muestras.

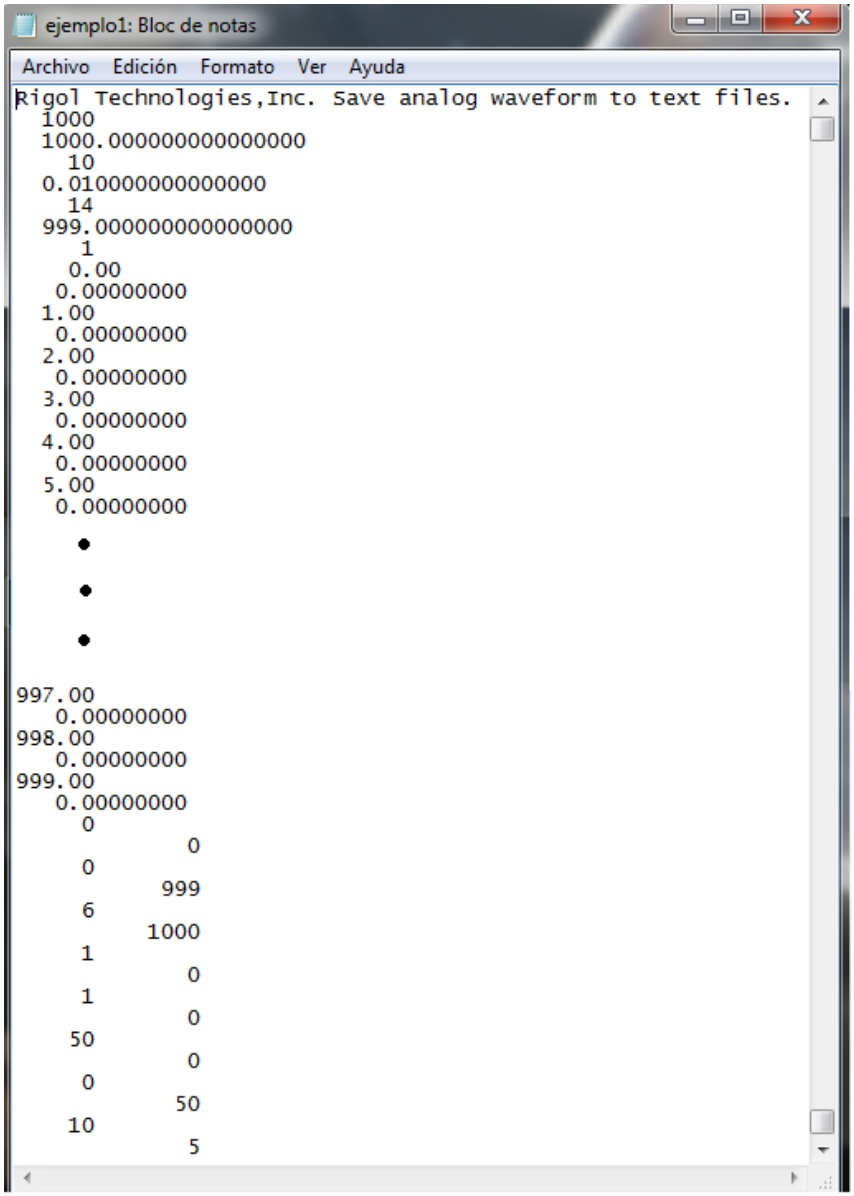


Figura 4.35 Archivo .txt de una secuencia binaria de 10 símbolos**.**

Una vez que se genera el archivo, será posible abrirlo dentro de Ultrawave.exe, véase figura 4.36. La secuencia generada por la interfaz gráfica es ahora mostrada con sus características eléctricas, lista para ser generada eléctricamente y analizada con la ayuda de algún dispositivo de medición, como un osciloscopio o un analizador de espectros.

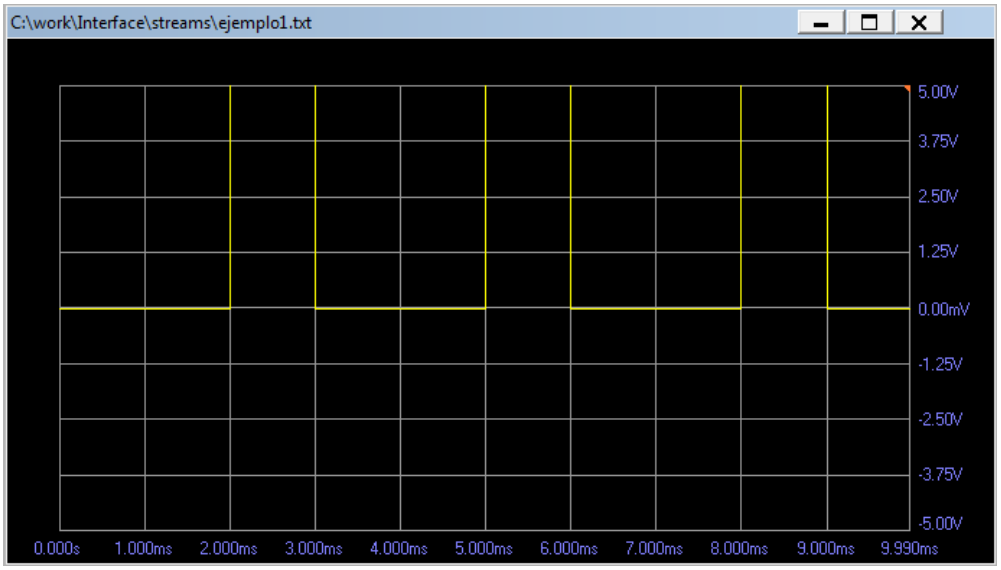


Figura 4.36 Señal generada en matlab, importada a Ultrawave.exe.

4.4. Implementación de módulos de codificación mediante MATLAB

Se han implementado mediante GUI Matlab una interfaz gráfica compuesta por diferentes módulos de codificación. La interfaz gráfica está integrada de la siguiente manera:

Generador de secuencias de bits.

Codificador de formas de onda.

Codificador de bloques lineales.

Codificador convolucional.

Cada módulo ha sido diseñado con la finalidad de proveer al usuario una herramienta de software que le sea útil dentro del laboratorio de comunicaciones. Contiene ejemplos del funcionamiento del codificador con el fin de comprobar los valores teóricos obtenidos dentro del curso de Teoría de Codificación. Así también contiene ejemplos de funciones que codifican y grafican secuencias de bits aleatorias, para después ser generadas eléctricamente por medio del generador RIGOL y así mismo puedan ser analizadas las formas de onda físicamente y llevar a cabo un análisis espectral de los fenómenos espectrales que presentan las secuencias binarias cuando sufren un proceso de codificación.

El prototipo de la interfaz gráfica se muestra en la figura 4.37. Se pueden observar los módulos mencionados anteriormente. También se puede identificar principalmente el uso de textos estáticos y textos de edición, los cuáles son de ayuda para interactuar con el usuario y de esa manera aportar y recibir información, es decir, datos que serán procesados y arrojados numéricamente o gráficamente a través de la ejecución de los botones, que realizan funciones específicas según sea el caso.

A continuación se describirá de manera breve el diseño y funcionamiento de cada módulo de codificación.

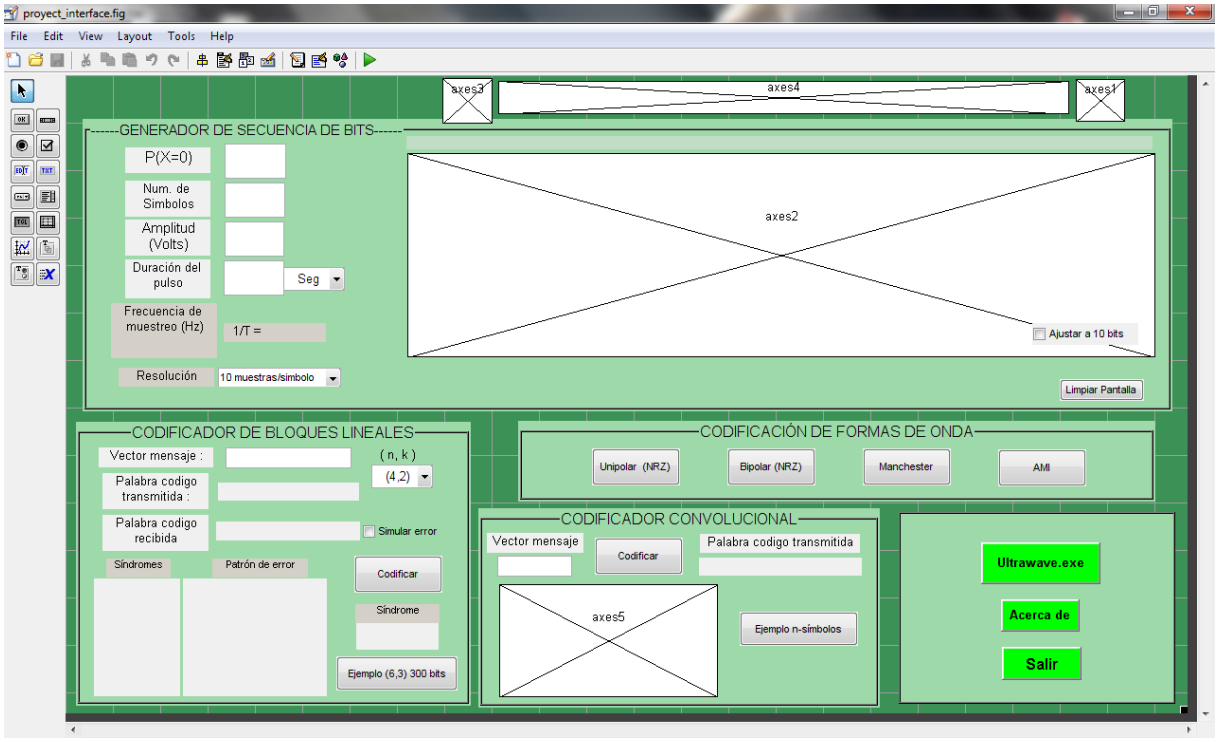


Figura 4.37 Prototipo de la Interfaz en GUI Matlab**.**

4.4.1. Codificador de formas de onda

El módulo de codificación por formas de onda, como se muestra en la figura 4.38, contiene 4 diferentes tipos de codificación; Unipolar, Bipolar, Manchester y AMI. Este módulo está vinculado directamente con el generador de secuencias de bits, es decir, primero es necesario asignar las características eléctricas de los pulsos para después elegir una codificación a las secuencias y poder visualizarlas gráficamente.

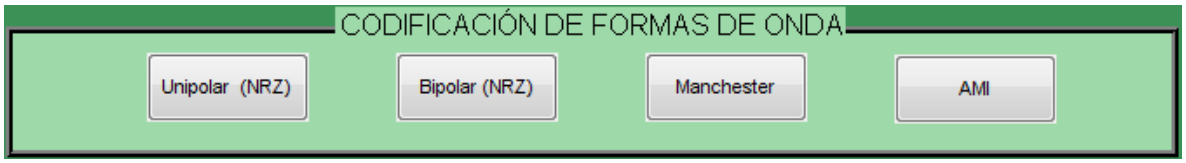


Figura 4.38 Códigos de línea: Unipolar, Bipolar(NRZ), Manchester y AMI.

El módulo del generador de bits presenta el diseño de la figura 4.39. Este módulo está compuesto a su vez por un espacio en blanco y acotado, donde serán mostradas las gráficas. Cuenta también con una opción de ajuste a 10 bits, que limita a visualizar sólo 10 bits, esto es utilizable en los casos que se generen grandes cantidades de bits y no se tenga una buena apreciación de estos. En la parte inferior derecha se aloja un botón para limpiar la pantalla.

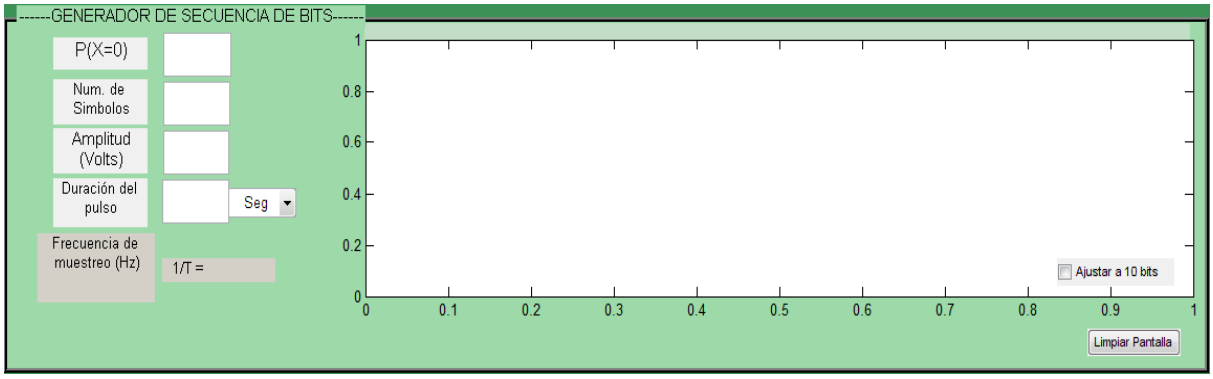


Figura 4.39 Generador de secuencias de bits.

Funcionamiento.

Cada botón ejecuta una rutina, utilizando como base el método de representación de bit, que consiste en crear un pulso constituido por una secuencia de puntos ó muestras.

4.4.1.1. Unipolar NRZ

Este método de codificación fue implementado siguiendo la siguiente representación de bit, ver figura 4.40. Dónde la presencia de un voltaje “A” durante un intervalo determinado representa el símbolo “1”. En este caso se

utilizan 10 muestras para formar un pulso, pero es importante observar que la duración del primer pulso será 10% menor a la de todos los demás, ya que comenzará a partir de su muestra número uno. Por otra parte los siguientes pulsos partirán de la última muestra del pulso anteriormente generado, hasta llegar a su décima muestra, de esta forma su duración será completa. Este detalle en la resolución de los pulsos es inevitable, por lo tanto no se tendrán pulsos ideales, sin embargo, es factible trabajar con ellos para cuestiones de visualización, es importante recordar que con una mayor resolución elevando el número de muestras por pulso se logrará una mejor precisión en la forma de onda del pulso.

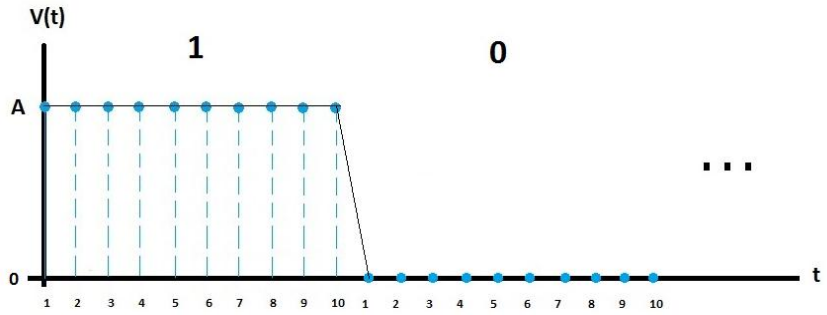


Figura 4.40 Represntación unipilar para los símbolos (1 y 0.)

4.4.1.2. Bipolar, Manchester y AMI

Estos tipos de codificación utilizan como base el método de representación mostrado en la figura anterior 4.40. Para estos casos los símbolos se diferencian por un cambio de polaridad en el voltaje, obsérvese la figura 4.41, se aprecia el valor de voltaje representado por “A” y “–A”. En el caso de Manchester el cambio de polaridad se alterna a la mitad de la duración del bit y en el código AMI se presenta una alternación de polaridad cada vez que se presente el símbolo “1”.

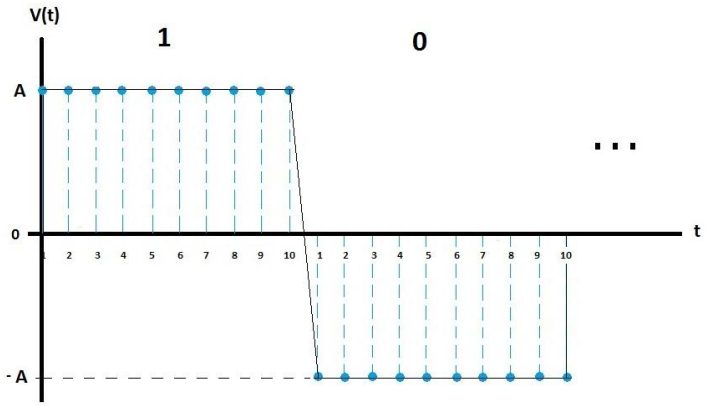


Figura 4.41 Represntación bipolar para los símbolos (1 y 0.)

4.4.1.3. Ejemplos de codificación por forma de onda

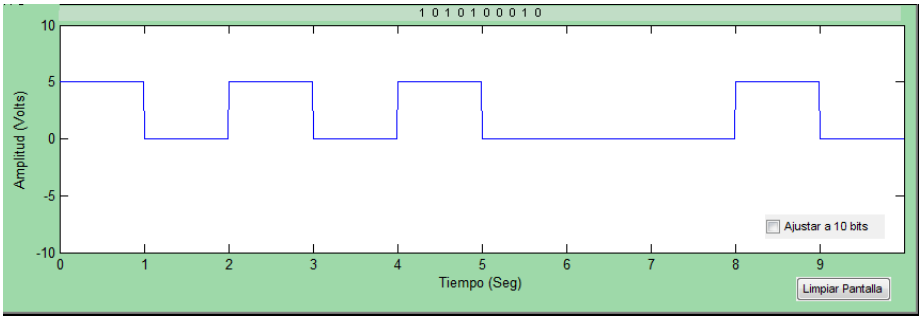
Para visualizar el funcionamiento de los diferentes tipos de codificación por forma de onda, se generará una secuencia de 10 símbolos con las siguientes características, observe los ejemplos presentados en la figura 4.42.

P(x=0) = 0.5

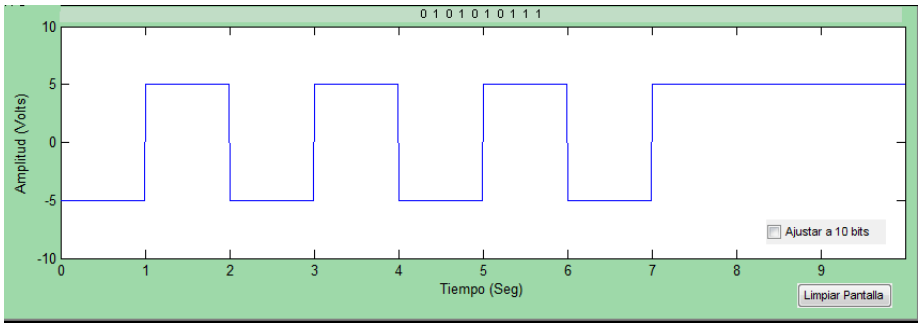
Amplitud = 5 volts

Duración = 1 segundo

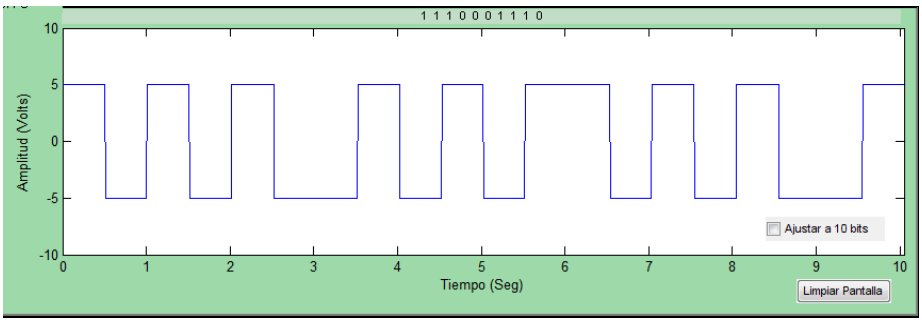
Muestras por símbolo = 100 muestras.



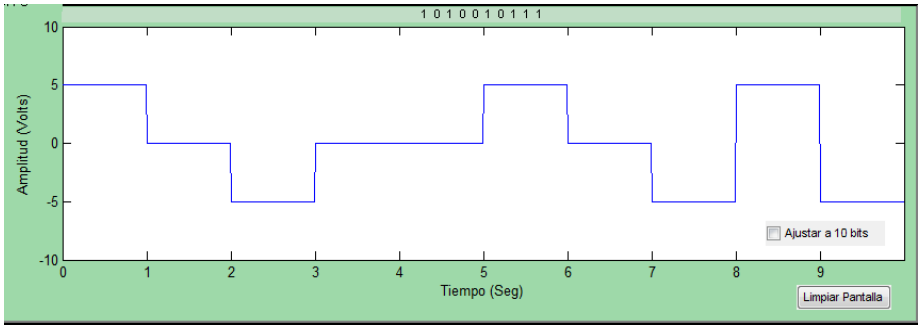
(a)



(b)



(c)



(d)

Figura 4.42 Ejemplos, (a) Unipolar, (b)Bipolar, (c)Manchester, (d)AMI.

4.4.2. Códigos de bloques lineales

El diseño del módulo codificador de bloques lineales presenta el aspecto mostrado en la figura 4.43. Como se observa, tiene una lista de menú para seleccionar el tipo de bloque lineal con que se requiera trabajar, cuenta con bloques del tipo (4,2), (5,2), (6,3), (7,4), (7,5) (9,6), (11,7) y (11,8).

El campo de entrada “vector mensaje” captura el mensaje binario que el usuario tecleará, dejando un espacio entre cada símbolo. Existe la opción de simular un error, este error se genera de manera aleatoria a lo largo de la palabra codificada generada a partir de un vector mensaje. En el caso de generar un error, el campo “palabra código transmitida” mostrará el mensaje codificado, mientras que el campo “palabra código recibida” mostrará el mensaje codificado con un símbolo erróneo. A su vez, debajo de estos campos se imprimen las tablas de síndromes con su respectivo patrón de error, de tal forma que a partir de la palabra código recibida se obtiene su síndrome, para después localizarlo con la tabla y así conocer el patrón de error correspondiente, de esta manera es posible localizar la ubicación del error generado a lo largo de la palabra.

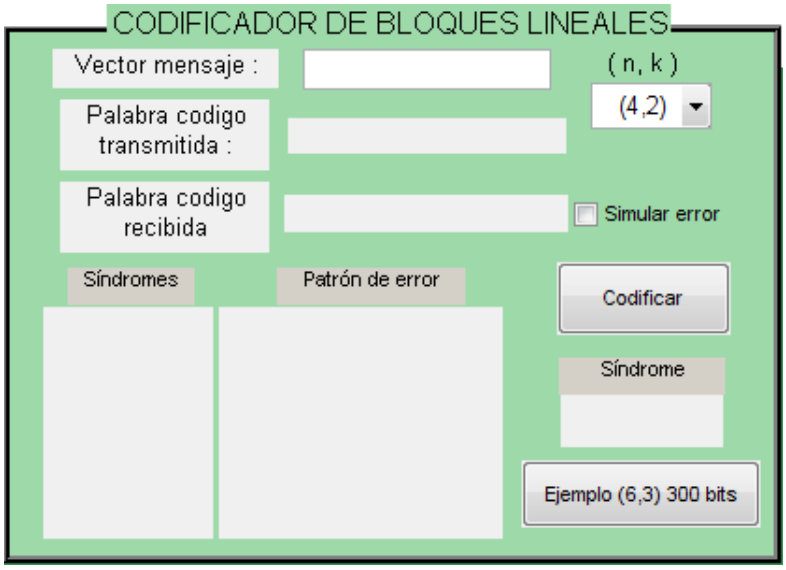


Figura 4.43 Modulo de codificación de bloques líneas, implementado dentro de la interfaz

Adicionalmente a la función de codificación que realiza este módulo, cuenta con un ejemplo de un bloque de tipo (6,3) que es aplicado a una secuencia de 300 bits. Este ejemplo es un caso particular definido por una rutina de Matlab, con el fin de analizar de forma eléctrica los fenómenos que se presentan con este tipo de codificación.

Funciones utilizadas.

Para aplicar una codificación de bloque lineal a un vector mensaje, se utilizó la siguiente función que provee la librería de Matlab:

```
encode(msg,n,k,'linear',genmat);
```

donde,

msg es el vector mensaje que se desea codificar.

n es tamaño del vector mensaje.

k es el tamaño de la palabra codificada.

genmat contiene la matriz generadora.

Para la obtención de los síndromes y el patrón de error se utilizaron las funciones;

```
syndtable(H)
```

Esta función arroja la tabla con los patrones de error, dónde;

H representa la matríz de verificación de paridad.

Para definir la tabla de síndromes, se calcula con la ayuda de la función

```
rem(e*H',2);
```

dónde se realiza la multiplicación de cada patrón de error (e) y la matriz transpuesta de verificación de paridad (H'). Esta función despliega de manera automática la tabla de síndromes.

Para generar un bit erróneo dentro de una palabra codificada se utiliza la siguiente rutina:

```
coderror=[randerr(1,n) zeros(1,0)];

codrx=bitxor(codmsg,coderror);
```

El arreglo “coderror” genera vectores de tamaño “n”, es decir del mismo tamaño que tendrá la palabra codificada, compuesto por un bit “1” generado de manera aleatoria a lo largo del vector y los demás “0´s”, por ejemplo:

Para n=6 tendríamos vectores como los siguientes;

coderror=[0 1 0 0 0 0];
coderror=[0 0 0 0 1 0];
coderror=[0 0 0 1 0 0];

Mientras que la función bitxor() se encargará de realizar la suma módulo 2 entre el vector codificado “codmsg” y el vector con error aleatorio “coderror”. Como resultado se tendrá un vector “codrx” modificado.

coderror 0 1 0 0 0 0 (sum2) codmsg 1 0 1 0 1 1 = codrx [1 1 1 0 1 1]

El vector “codrx” es ahora la palabra código errónea, la cual se ocupa para identificar su síndrome y localizar su patrón de error y detectar la posición del bit erróneo, Este procedimiento está implementado en la interfaz con fines educativos.

A manera de ejemplo en la figura 4.44 se muestra una captura de pantalla del módulo de bloques lineales,

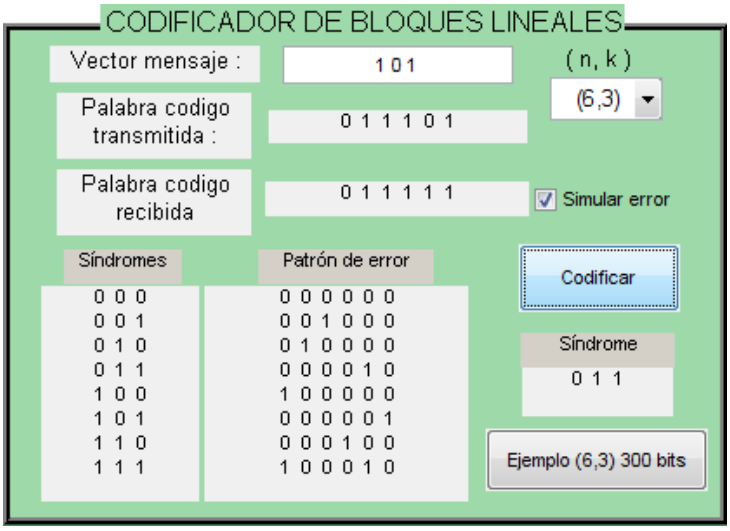


Figura 4.44 Funcionamiento del codificador de bloques lineales.

4.4.3. Códigos convolucionales

El módulo codificador convolucional presenta un diseño muy sencillo, dónde sólo se muestra un caso particular de diagrama de conexión, que cuenta con dos salidas y tres espacios de memoria como se muestra en la figura 4.45.

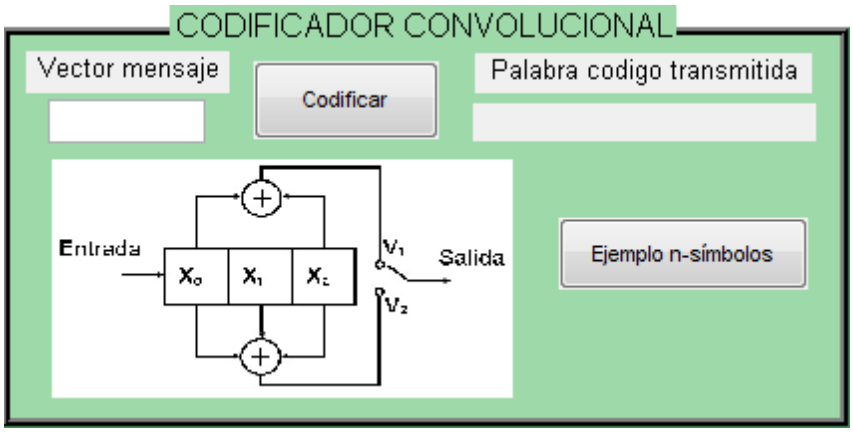


Figura 4.45 Caso particular de diagrama de conexión para un codificador convolucional.

El desarrollo de este codificador consistió únicamente en generar la rutina que realice las operaciones necesarias, (sumas modulo 2), para obtener las salidas del codificador.

Las funciones utilizadas son:

```
V1=bitxor(X(0),X(2));
```

y

```
V2=bitxor(V1,X(1));
```

Dónde X(0), X(1) y X(2) representan los símbolos que se encuentran dentro de la memoria del codificador en determinado instante de tiempo. A la salida se obtendrá una secuencia de símbolos codificados. La figura 4.46 representa un ejemplo de codificación para un vector de tres símbolos. Es importante recordar que se debe introducir el vector mensaje separando cada símbolo con un espacio desde el teclado.

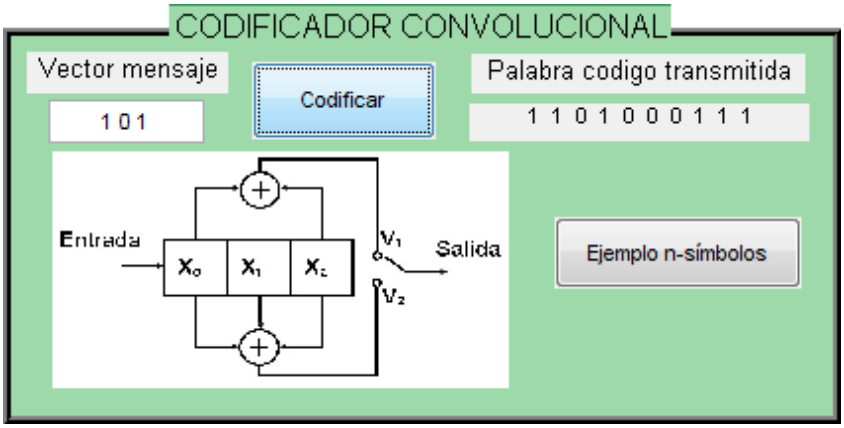


Figura 4.46 Funcionamiento del codificador convolucional

4.5. Resultados experimentales

La interfaz implementada tiene como objetivo crear esquemas de codificación aplicados a secuencias binarias aleatorias, que a su vez son generadas eléctricamente con la ayuda del software Ultrawave.exe a través del generador Rigol, para después ser analizadas con dispositivos de medición, como el osciloscopio y el analizador de espectros.

4.5.1. Formas de onda obtenidas eléctricamente

Se analizarán de forma experimental secuencias binarias obtenías por el modulo codificador de formas de onda con las siguientes características eléctricas para los cuatro casos (unipolar, bipolar, Manchester y AMI).

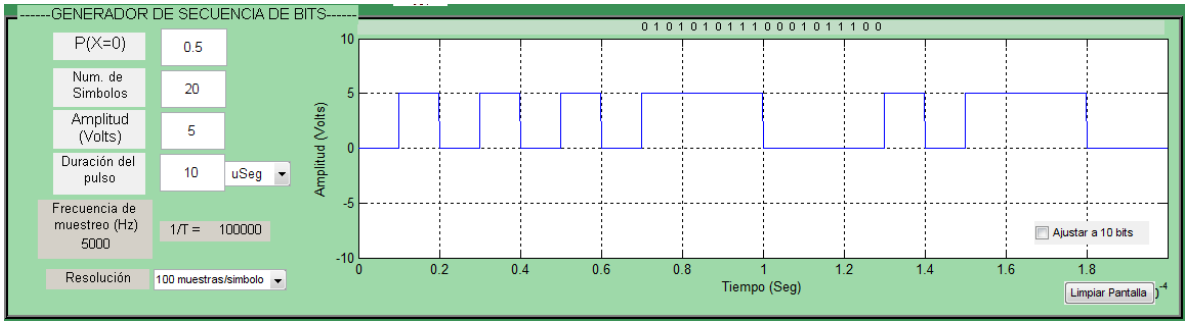
Probabilidad de ceros = 0.5

Número de símbolos = 20

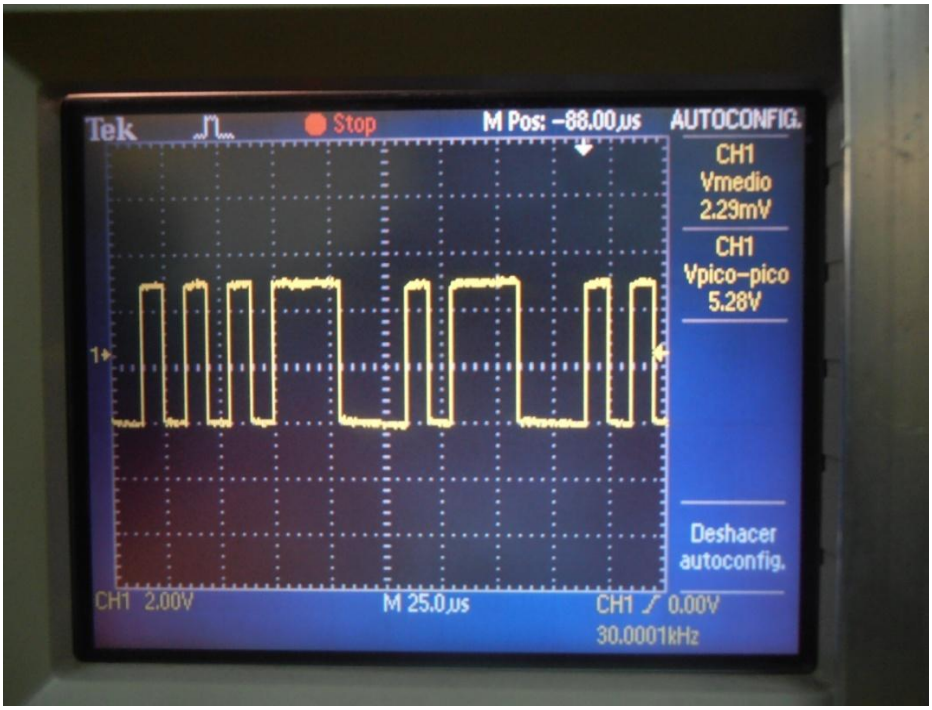
Amplitud = 5 volts

Duración del pulso = 10 micro segundos.

Resolución de símbolo = 100 muestras/símbolo.

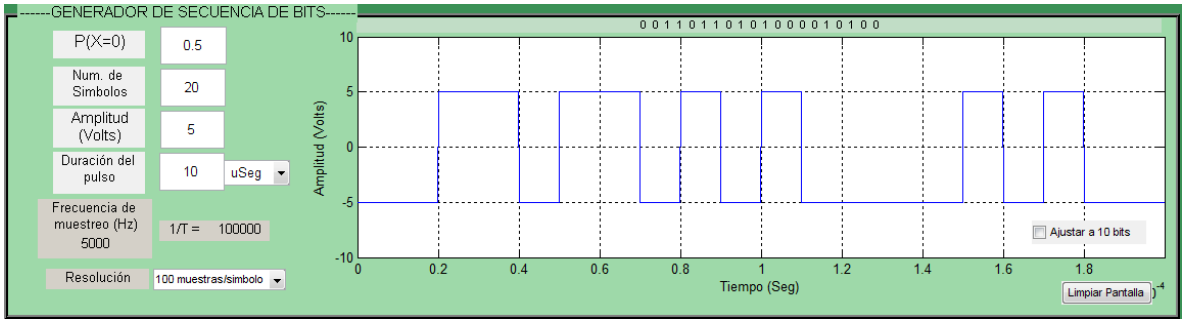


(a)

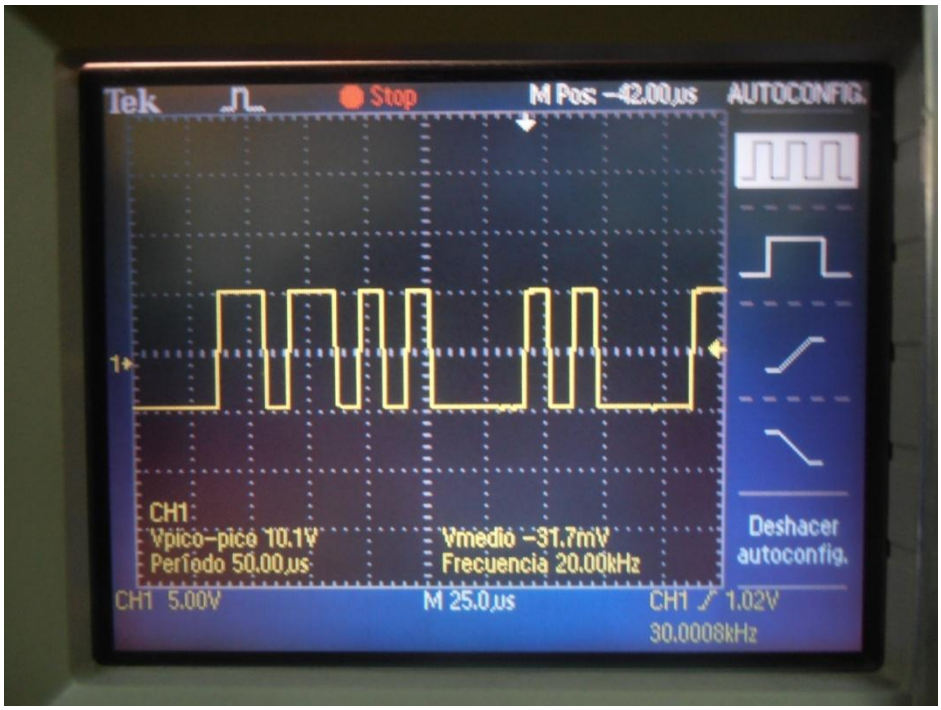


(b)

Figura 4.47 Codificación Unipolar aplicada a una secuencia aleatoria;
(a) Representación mediante Matlab. (b) Representación eléctrica.

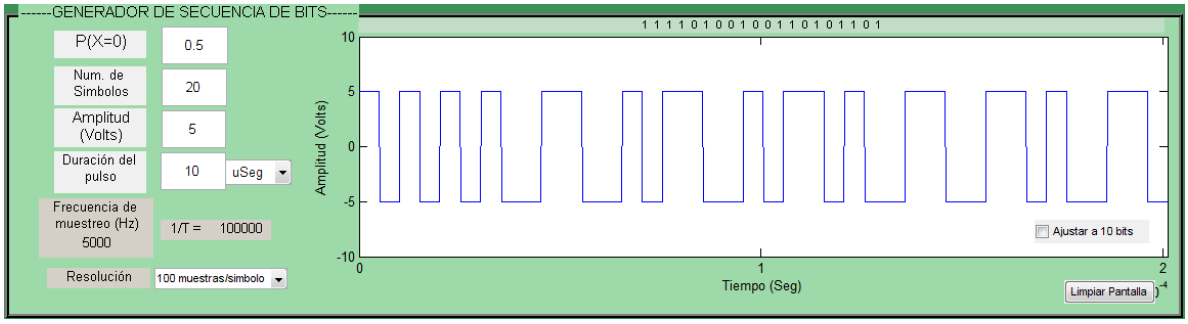


(a)

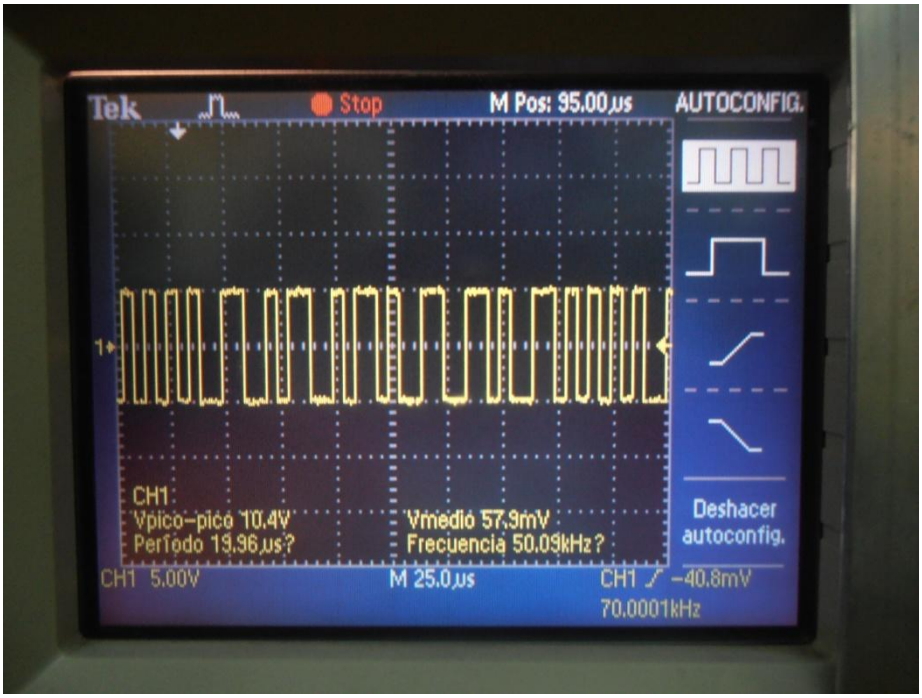


(b)

Figura 4.48 Codificación Bipolar aplicada a una secuencia aleatoria;
(a) Representación mediante Matlab. (b) Representación eléctrica.

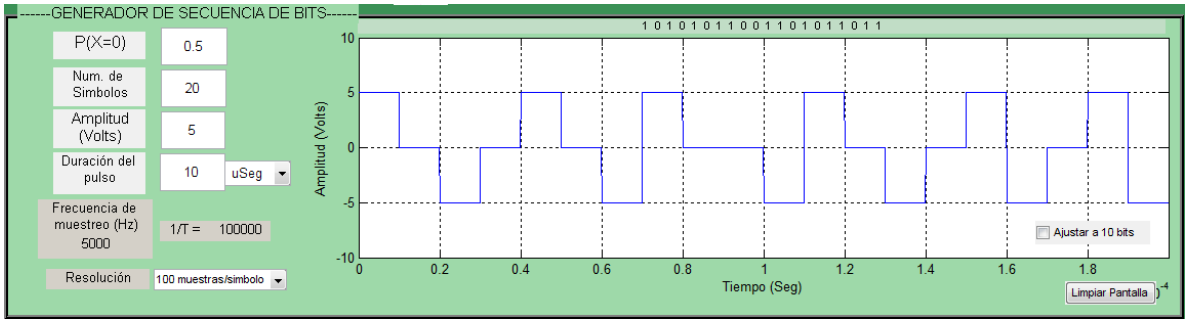


(a)

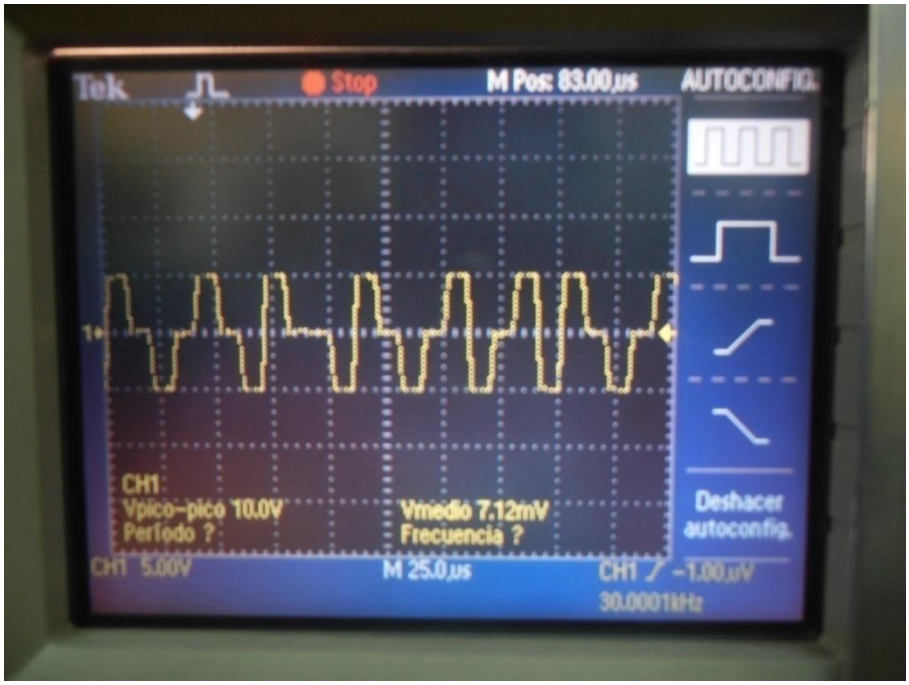


(b)

Figura 4.49 Codificación Manchester aplicada a una secuencia aleatoria;
(a) Representación mediante Matlab. (b) Representación eléctrica.



(a)



(b)

Figura 4.50 Codificación AMI aplicada a una secuencia aleatoria;
(a) Representación mediante Matlab. (b) Representación eléctrica.

Nota: Es importante mencionar que el generador Rigol genera señales arbitrarias de forma periódica, de tal manera que las secuencias de 20 símbolos se repetirán una de tras de la otra. Por esta razón en los ejemplos mostrados en las figuras 4.49 y 4.50 se puede hacer una comparación entre la señal creada en Matlab y la señal visualizada en el osciloscopio y es posible observar que las secuencias se repiten. Esto es apreciable cuando se manejan secuencias pequeñas. Para secuencias mas grandes se puede evitar ver este proceso periódico sobre la pantalla del osciloscopio.

4.5.2. Análisis espectral

Para llevar a cabo un análisis espectral de las secuencias aleatorias se requiere generar grandes cantidades de símbolos. Para ellos se utilizará una resolución de 10 muestras/símbolo, lo cual permite generar eléctricamente un máximo de 400 símbolos.

Haciendo uso de un analizador de espectros y un osciloscopio se llevó a cabo la medición de la distribución en frecuencia de secuencias codificadas con códigos de línea y las siguientes características:

Probabilidad de ceros = 0.5

Número de símbolos = 400

Amplitud = 0.125 volts

Duración del pulso [Tb] = 10 y 1 micro segundos.

Resolución de símbolo = 10 muestras/símbolo.

Se utilizaron dos diferentes duraciones de pulso (Tb) (10 y 1 micro segundos) con la finalidad de observar y verificar la relación teórica que existe entre la tasa de bits y el ancho de banda. A continuación se presentan los espectros obtenidos.

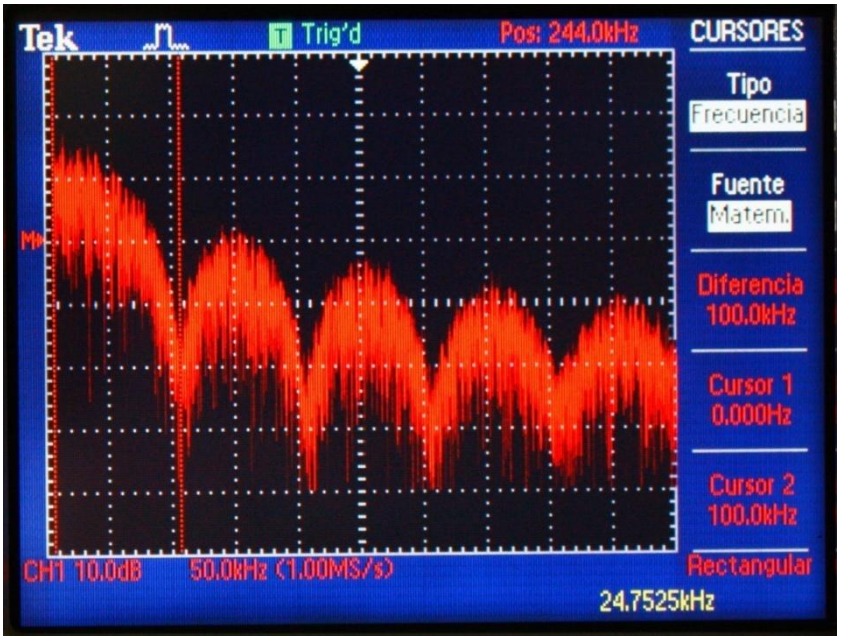


Figura 4.51 **Espectro de código unipolar, Tb= 10 mili segundos - Osciloscopio**

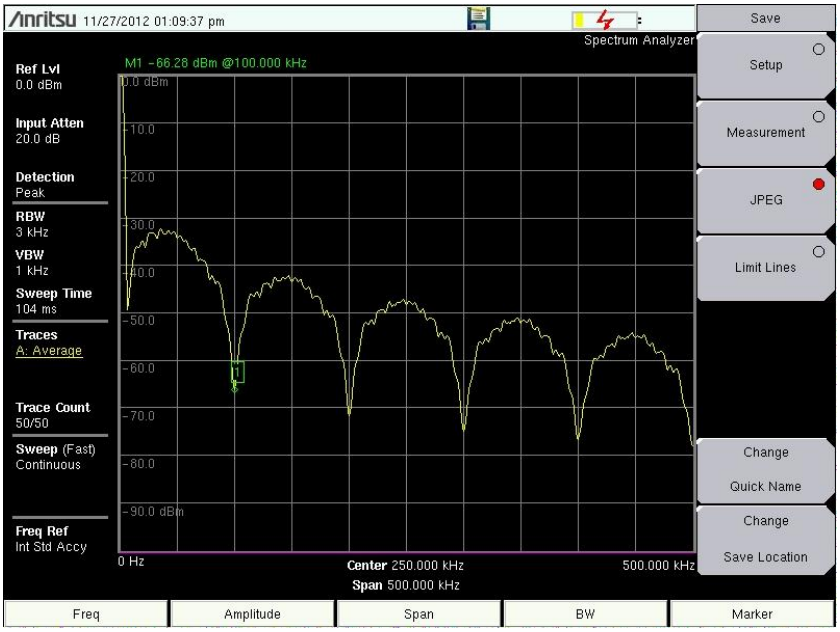


Figura 5 .52 **Espectro de código unipolar, Tb= 10 mili segundos – Analizador de espectros.**

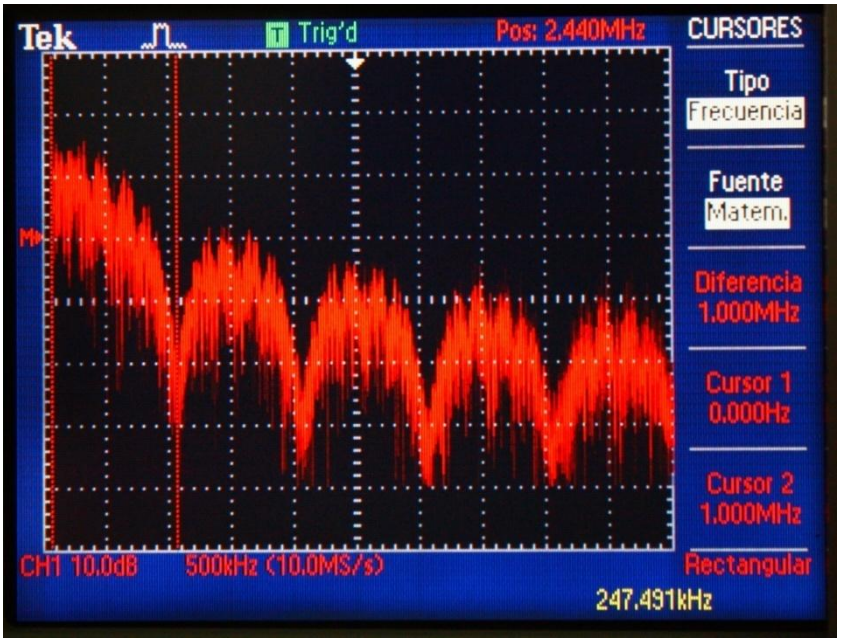


Figura 4.53 **Espectro de código unipolar, Tb= 1 mili segundo - Osciloscopio**

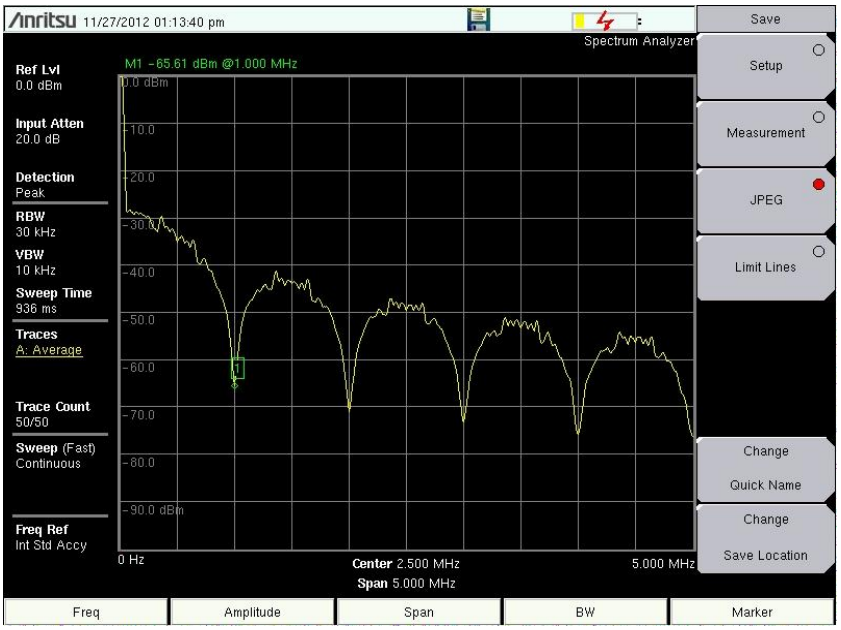


Figura 4.54 **Espectro de código unipolar, Tb= 1 mili segundo – Analizador de espectros.**

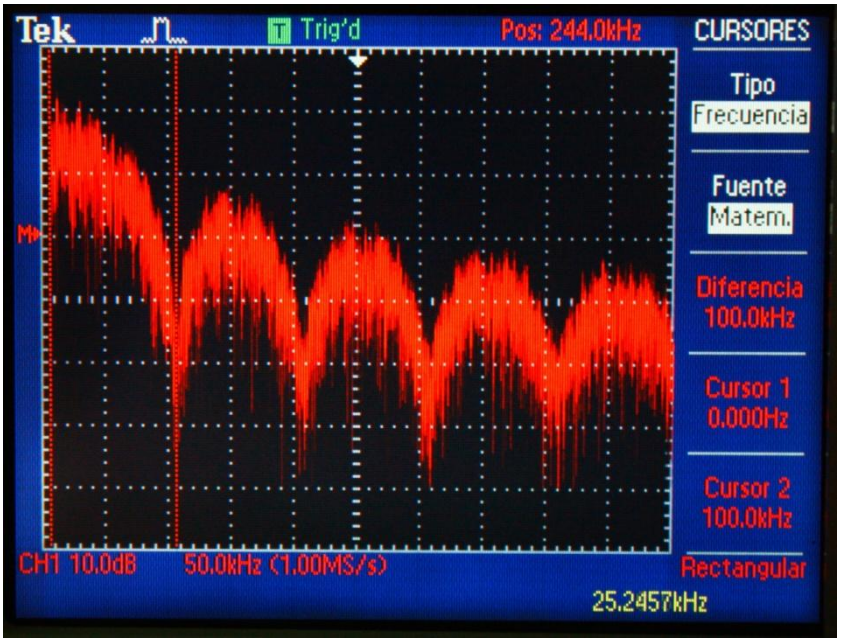


Figura 4.55 **Espectro de código biipolar, Tb= 10 mili segundos – Osciloscopio.**

Figura 4.56 **Espectro de código biipolar, Tb= 10 mili segundos – Analizador de espectros.**

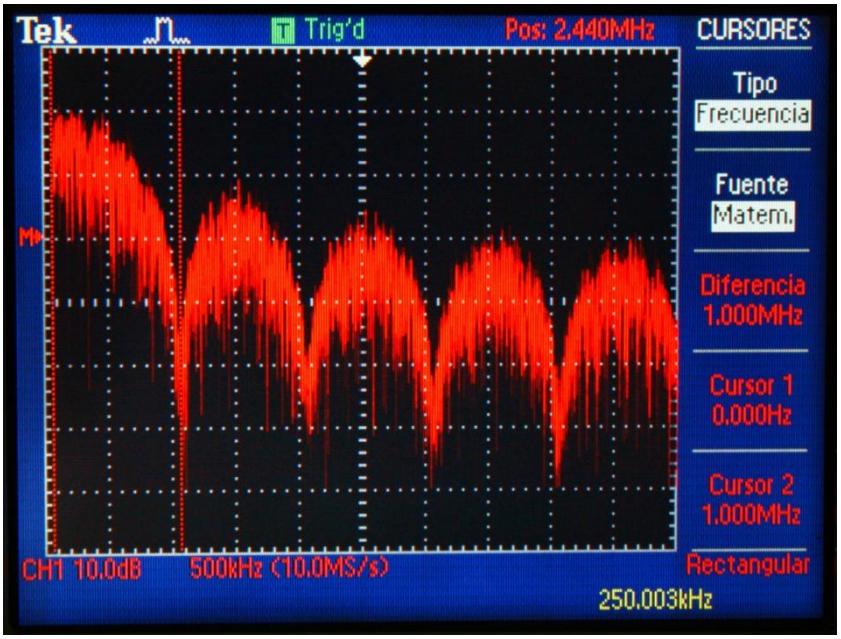


Figura 4.57 **Espectro de código biipolar, Tb= 1 mili segundo – Osciloscopio.**

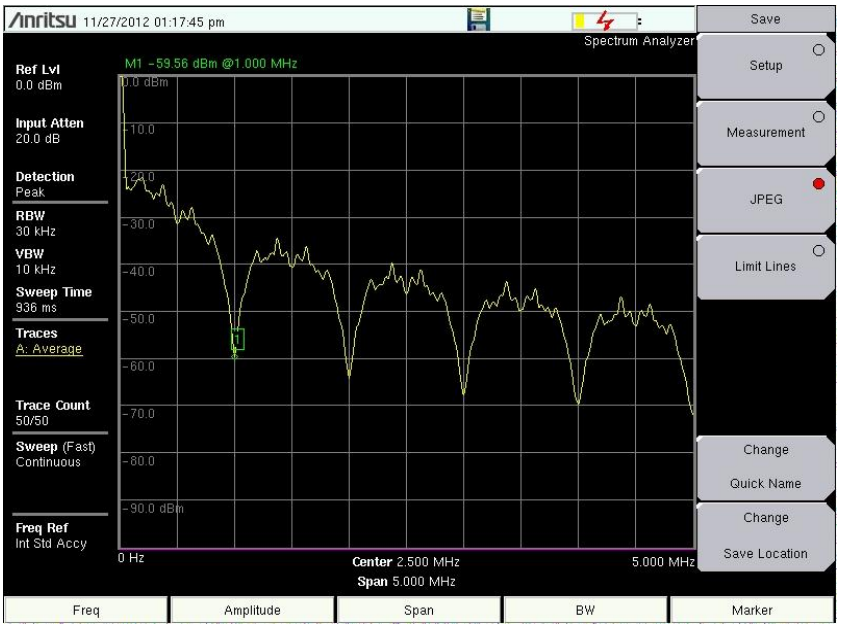


Figura 4.58 **Espectro de código biipolar, Tb= 1 mili segundo – Analizador de espectros.**

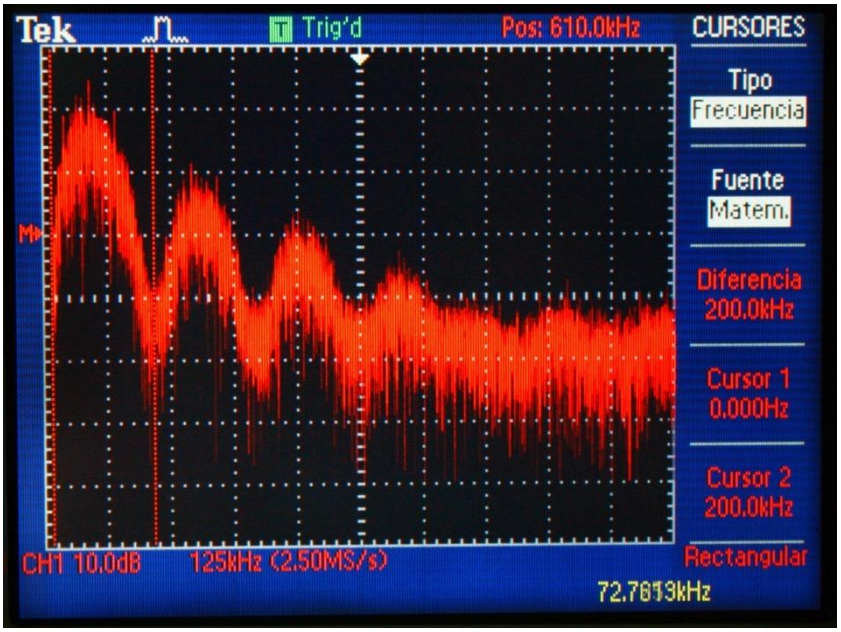


Figura 4.59 **Espectro de código Manchester, Tb= 10 mili segundos – Osciloscopio.**

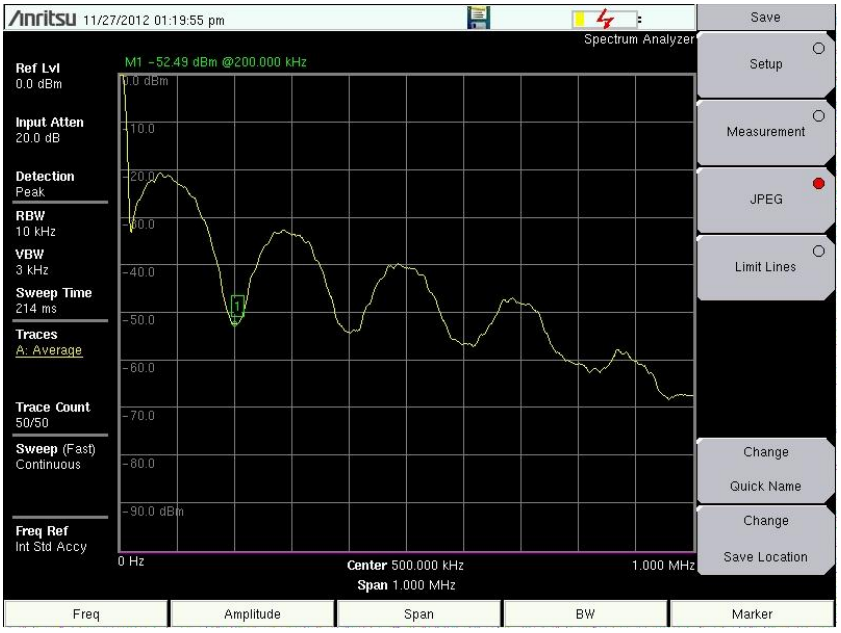


Figura 4.60 **Espectro de código Manchester, Tb= 10 mili segundos – Analizador de espectros.**

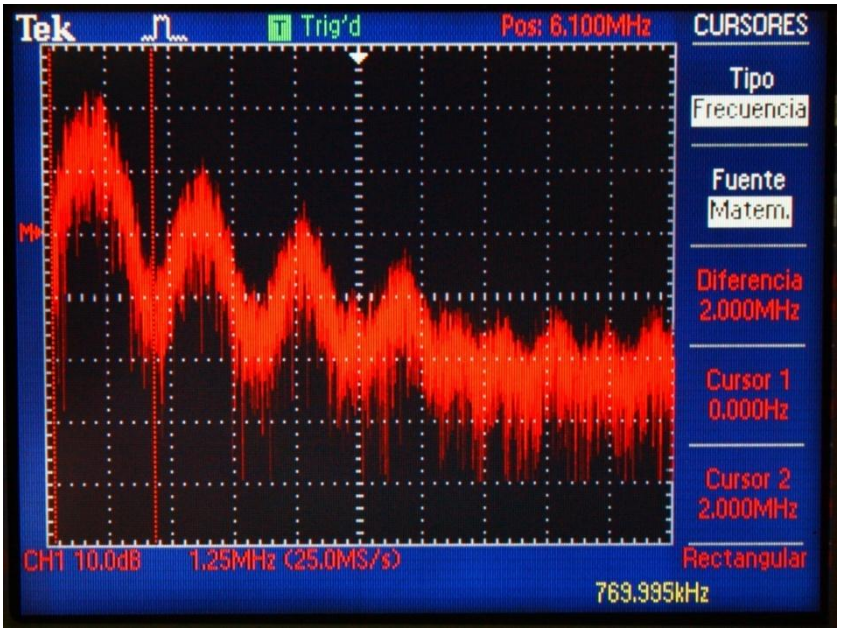


Figura 4.61 **Espectro de código Manchester, Tb= 1 mili segundo – Osciloscopio.**

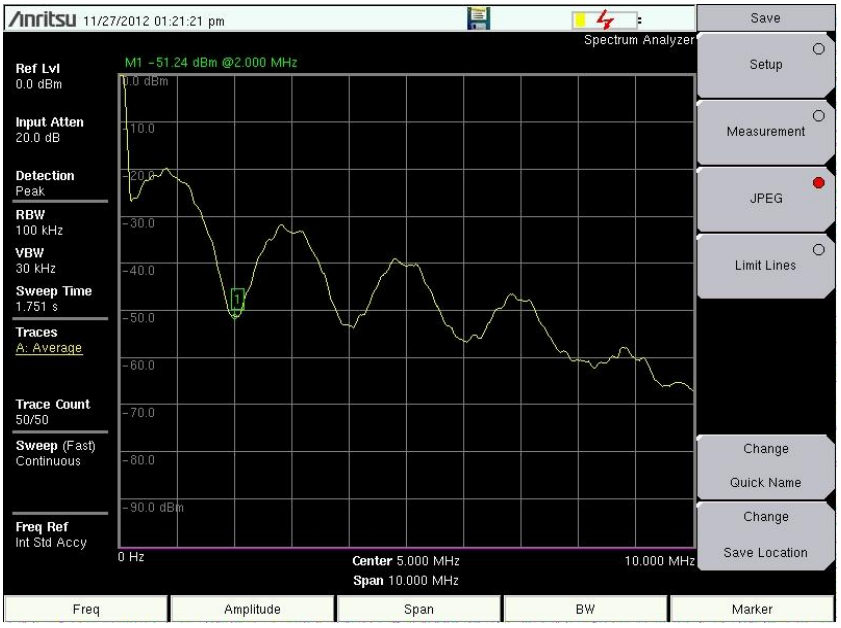


Figura 4.62 **Espectro de código Manchester, Tb= 1 mili segundo – Analizador de espectros.**

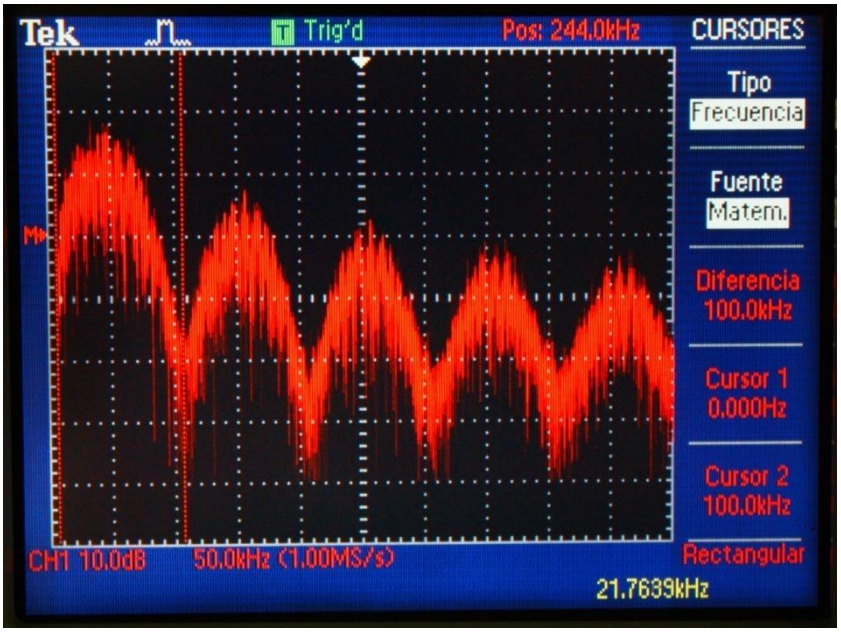


Figura 4.63 **Espectro de código AMI, Tb= 10 mili segundos – Osciloscopio.**

Figura 4.64 **Espectro de código AMI, Tb= 10 mili segundos – Analizador de espectros.**

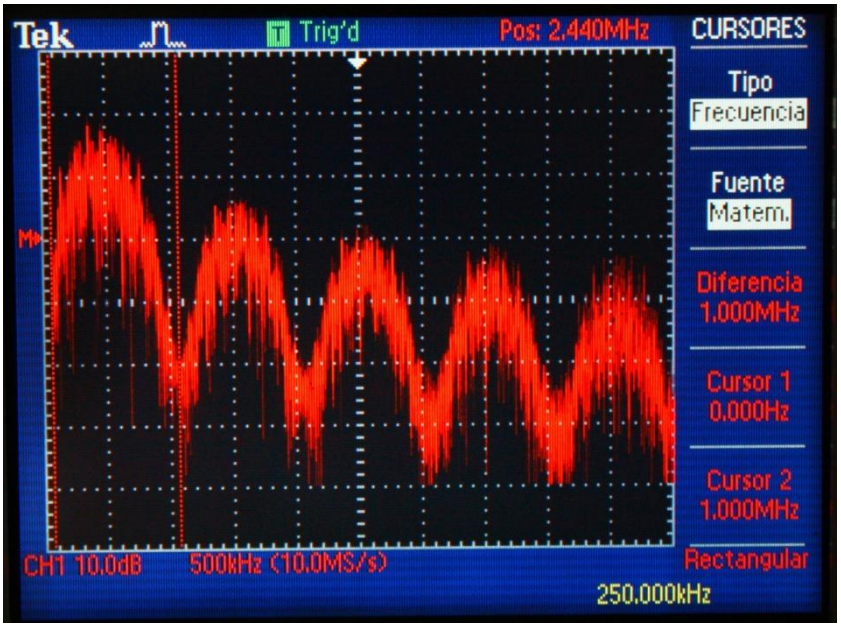


Figura 4.65 **Espectro de código AMI, Tb= 1 mili segundo – Osciloscopio.**

Figura 4.66 **Espectro de código AMI, Tb= 1 mili segundo – Analizador de espectros.**

El análisis anterior comprueba de manera experimental las formas de espectro definidas de manera teórica en este trabajo. De igual manera sea realizaron mas mediciones con la finalidad de observar los cambios que ocurren en el espectro de una señal al pasar por un proceso de codificación de bloques lineales y codificación convolucional.

El ejemplo de códigos de bloques, consiste en generar una secuencia aleatoria de 300 símbolos, aplicando un bloque del tipo (6,3) y utilizando a su vez una representación de forma de onda unipolar. Las características de la secuencia son las siguientes:

Probabilidad de ceros = 0.5

Número de símbolos = 300

Amplitud = 0.125 volts

Duración del pulso [Tb] = 1 micro segundo.

Resolución de símbolo = 10 muestras/símbolo.

La figura 4.67 representa la distribución en frecuencia de una secuencia aleatoria con las características anteriores, es decir, una secuencia binaria sin codificar.

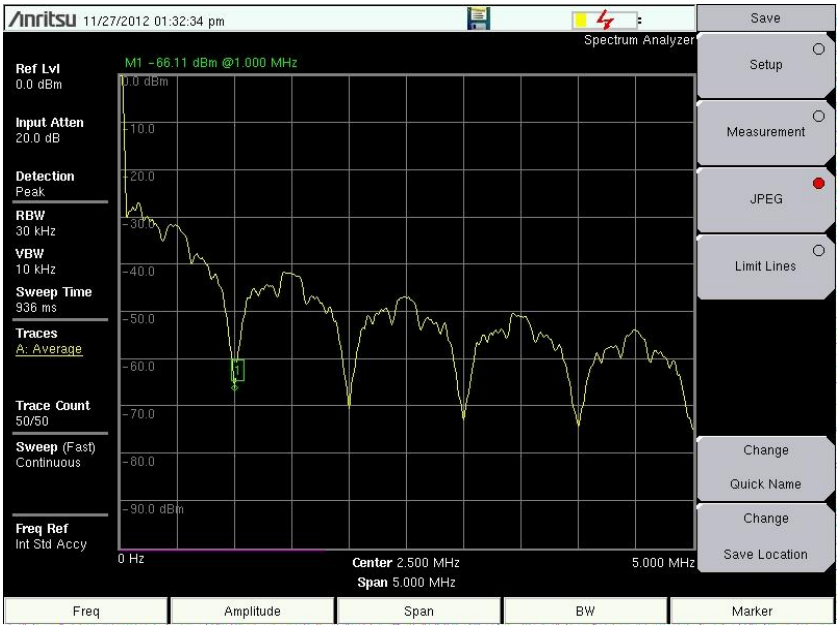


Figura 4.67 **Secuencia de bits sin codificar.**

Posteriormente al aplicar por una codificación de bloques lineales se obtuvo la siguiente distribución en frecuencia. Véase figura 4.68.

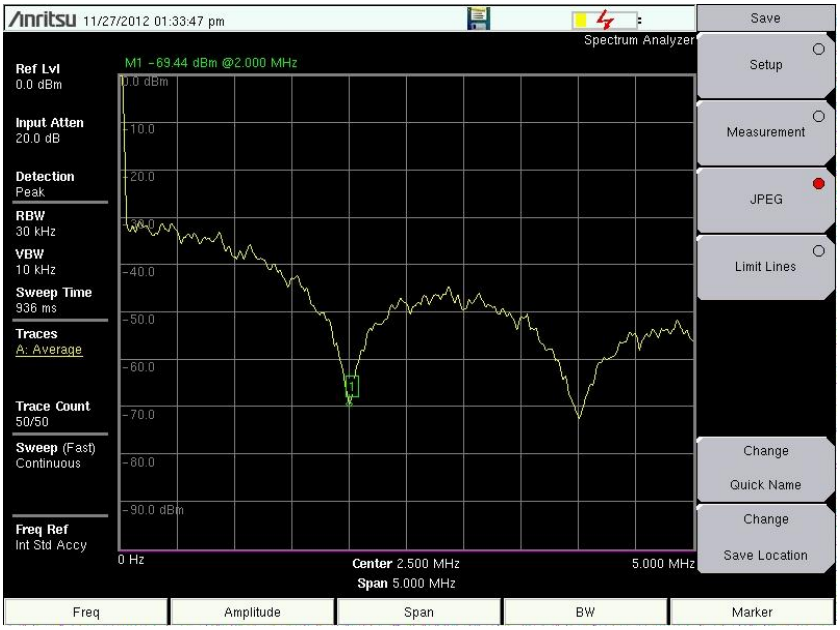


Figura 4.68 **Secuencia de bits codificados.**

Se puede observar lo que ocurre cuando se utiliza un bloque del tipo (6,3) para codificar una secuencia. El espectro de la secuencia codificada (figura 4.68) presenta un desplazamiento al doble de su ancho de banda con respecto a la secuencia sin codificar (figura 4.67). Los cursores están situados en el primer corte por cero en la distribución de frecuencia de ambas señales y se puede verificar la diferencia entre estas, que va de 1MHz a 2MHz, es decir, se puede comprobar experimentalmente que la expansión en ancho de banda es $\frac{1}{R} = \frac{n}{k} = \frac{6}{3} = 2$.

Otro ejemplo experimental fue observar el fenómeno en frecuencia que arroja un proceso de codificación convolucional presentada en la interfaz desarrollada para este trabajo. Para este caso, las características de la secuencia son las mismas que la anterior, al igual que su representación unipolar, solo con la diferencia que en este caso se generaron 400 símbolos.

A continuación, en la figura 4.69 se presentan la distribución en frecuencia de la secuencia aleatoria generada sin codificar, es decir, antes de entrar al codificador convolucional. Y por otro lado en la figura 4.70 se observa el espectro de la misma secuencia después de salir del codificador convolucional.

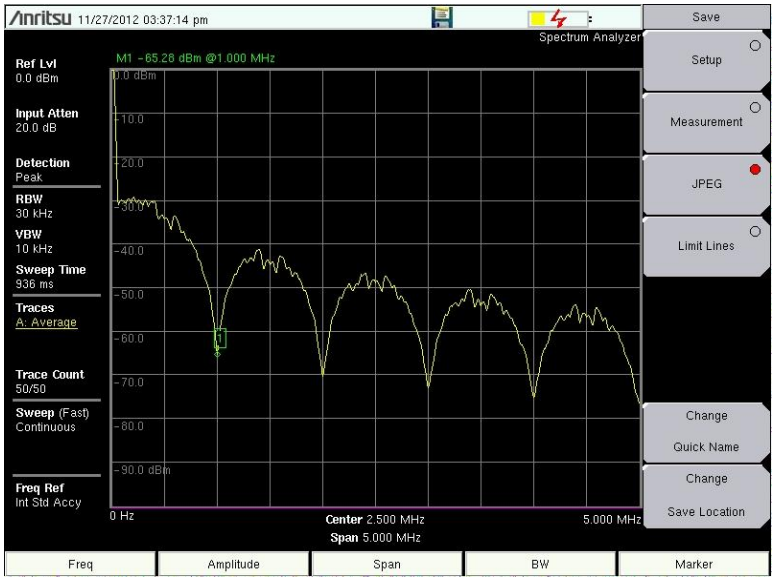


Figura 4.69 **Secuencia de bits a la entrada del codificador convolucional.**

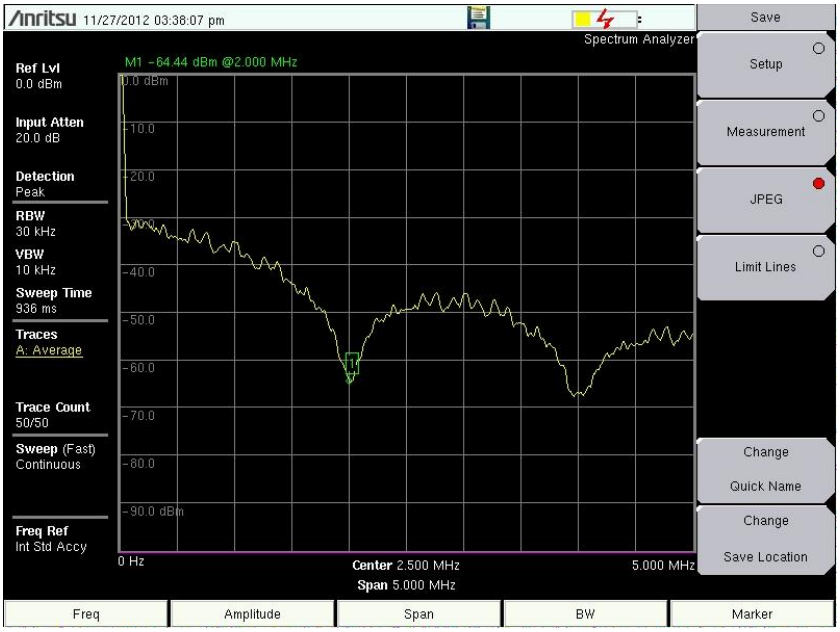
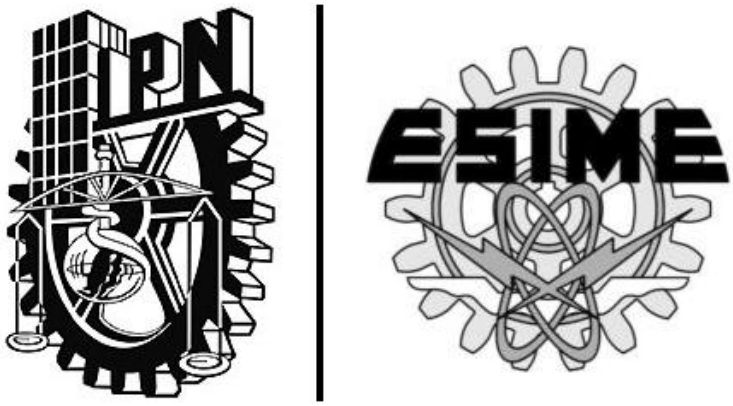


Figura 4.70 **Secuencia de bits a la salida del codificador convolucional.**

Como se observa en ambas figuras (4.69 y 4.70) el fenómeno es similar al anterior, pues el ancho de banda de la señal se desplaza al doble a la salida del codificador convolucional.



“Conclusiones”

CONCLUSIONES

Al realizar el trabajo de forma experimental y el haber obtenido resultados apegados con los teóricos, garantiza la aplicación de la interfaz diseñada como una herramienta de trabajo y apoyo académico dentro del laboratorio de comunicaciones. Es importante mencionar que el prototipo tiene limitaciones en el rango de duración de los pulsos generados eléctricamente. Un rango aceptable fue de 100 mili segundos a 1 mili segundo. Dentro de este rango se tiene una buena apreciación de las formas de onda de las secuencias.

La implementación de este prototipo favorecerá el aprendizaje del ingeniero y es una muestra de retroalimentación, pues los alumnos contribuyen con sus conocimientos diseñando herramientas que faciliten el desarrollo de las futuras generaciones.

BIBLOGRAFÍA / REFERENCIAS

- Sklar, B. (1988) *Digital Communications: Fundamentals and Applications,* Nueva Jersey, Prentice Hall.
- Proakis, John G. (2002) *Communication systems engineering,* Nueva Jersey, Prentice Hall.
- Shanmugam, K.S. (1979), *Digital and analog communication systems,* Canada, John Wiley & Sons, Inc.
- Lathi, B. P. (), *Modern digital and analog communication systems,* xxx, Oxford University Press.
- Hwei Hsu, (2003), *Schaum's outlines: Analog and digital communications*, USA, McGraw-Hill Companies, Inc.
- Haykin, Simon (2001) *Communication systems,* Nueva York, John Wiley & Sons, Inc.
- Ziemer, Rodger E. (1992) *Introduction to Digital Communication,* Nueva York, Macmillan Publishing Company.
- Cover, Thomas M. (2006) *Elements of information theory,* Canada, John Wiley & Sons, Inc.
- Tomasi, Wayne (2003) *Sistemas de comunicaciones electrónicas,* México, Pearson Eduación.
- Wesolowski, Krzysztof (2009) *Introduction to digital communications systems, UK,* John Wiley & Sons, Inc.
- Rigol (2007) *User manual DG2000 Series Function/Arbitrary Waveform Generator,* Rigol Technologies, Inc.
- Anritsu (2010), *User guide Spectrum Master Handheld Spectrum Analyzer, MS2712E and MS2713E,* USA, Anritsu Company.