ESCUELA POLITECNICA NACIONAL

FACULTAD DE INGENIERIA ELECTRICA

TESIS DE GRADO

PREVIA A LA OBTENCION DEL
TITULO DE INGENIERO EN
ELECTRONICA Y TELECOMUNICACIONES

ANALISIS Y SIMULACION DE UN SISTEMA DE TRANSMISION DIGITAL UTILIZANDO UN MICROCOMPUTADOR

IVAN GUSTAVO CASTRO LLANES

MARCO ANTONIO ORBE ROMERO

MARZ0 DE 1995

CERTIFICACION

Certifico que, bajo mi dirección, la presente tesis fue realizada en su totalidad por los señores Iván Castro Ll. y Marco Orbe R.

Ing. Pablo Hidalgo Lascano

DIRECTOR DE TESIS

DEDICATORIA

A mis padres,

a quienes les debo

todo lo que soy.

Iván

DEDICATORIA

A mis padres,

por su invaluable sacrificio,

A mis hermanos,

por su constante apoyo,

A mi esposa,

por su amor y comprensión.

Marco

AGRADECIMIENTO

Al Ing. Pablo Hidalgo, por su acertada dirección y apoyo permanente para la realización de esta tesis. A los profesores de la Escuela Politécnica Nacional y a todas las personas que contribuyeron a su culminación.

INDICE

INTRODUCCION

El gran desarrollo de las telecomunicaciones en la actualidad se debe a los avances e investigaciones logrados en el campo de la Comunicación Digital. Como en toda ciencia, en Electrónica y Telecomunicaciones los conceptos deben ser bien cimentados, esto es, deben cubrir la teoría básica de las comunicaciones analógicas y digitales.

Para el estudio de Comunicación Digital es necesario una base matemática avanzada, ya que los tópicos tratados en esta ciencia así lo exigen.

La simulación es una de las herramientas de análisis más poderosas que disponen quienes se dedican al diseño y estudio de sistemas complejos. Una vez que se describe el sistema, se utiliza un lenguaje de programación que concrete la descripción conceptual en un programa de computador que ejecuta la simulación.

Por lo tanto, la simulación de un Sistema de Transmisión Digital por computadora puede ser útil para la enseñanza y estudio de los conceptos utilizados en Comunicación Digital, ya que con la introducción de la computadora y el avance de la informática se tiene la posibilidad de desarrollar paquetes de programas dedicados a la enseñanza, no sólo de estudiantes a nivel de pre-grado sino de profesionales pertenecientes al área de telecomunicaciones.

Es por esto que en esta tesis se desarrolla un programa de computación para el análisis y simulación de un Sistema de Transmisión Digital, que principalmente trata a la codificación y modulación digital, ya que éstas constituyen una parte muy importante en la transmisión de datos.

El programa es de tipo didáctico, y pretende, consolidar los conceptos que se imparten en la materia de

Comunicación Digital, a los estudiantes de Electrónica y Telecomunicaciones de la facultad.

Para ello, la tesis consta de 7 capítulos y por la extensión de los tópicos a tratarse se ha realizado entre dos personas.

En el capítulo 1 se describen los elementos básicos de un sistema de transmisión digital como son: generación binaria, transmisor, canal de transmisión y receptor; luego de lo cual se tratan las anomalías que afectan al sistema.

En el capítulo 2 se realiza un estudio de las señales en el dominio del tiempo y la frecuencia, empleándose a las Series y Transformadas de Fourier como las herramientas matemáticas para la comprensión del comportamiento de los códigos y las modulaciones, desde el punto de vista de la densidad espectral de potencia.

El capítulo 3 trata los diversos tipos de codificación, características de los mismos y algoritmos de implementación. En el capítulo 4 se enfoca la modulación digital binaria y multinaria, así como su comparación.

En el capítulo 5 se desarrollan los algoritmos para la implementación de la densidad espectral de códigos y modulaciones que son considerados en el programa.

El capítulo 6 es la implementación de todo el estudio hecho en los capítulos anteriores consolidados en el programa TELECOM, que esta realizado en el lenguaje de programación Turbo Pascal 6.0. Se escogió este lenguaje por las mejores características gráficas que presenta.

El capítulo 7 contempla las conclusiones, apéndices y bibliografía.

A criterio de los autores los objetivos propuestos en la tesis se han cumplido, debiendo destacar el aporte académico y didáctico que se propuso el presente trabajo como herramienta auxiliar para el aprendizaje de la Comunicación Digital.

CAPITULO 1

SISTEMA BASICO DE TRANSMISION DIGITAL

1.1. ESTRUCTURA BASICA DE UN SISTEMA DE TRANSMISION DIGITAL

1.1.1. Generalidades

Los sistemas de comunicación se encuentran en todas partes donde se transmite información de un punto a otro. El teléfono, la radio y la televisión son factores de la vida diaria.

Los circuitos de larga distancia cubren el globo terráqueo llevando texto, voces e imágenes. Los sistemas de radar y telemetría desempeñan papeles importantes, vitales en navegación, defensa e investigación científica. Las computadoras "hablan" a otras computadoras por medio de enlaces transcontinentales de datos. Ciertamente se han logrado grandes avances desde los días de la transmisión por código Morse.

Desde un punto de vista general los sistemas de comunicación tienen la misma función básica, es decir la **transmisión de información**.

Comunicación es todo proceso por medio del cual la información se transfiere desde un punto llamado **fuente**, en espacio y tiempo, a otro punto que es el **destino** o usuario. Un sistema de comunicación es la totalidad de mecanismos que proporcionan el enlace para la transferencia de información entre fuente y destino. Un sistema de comunicación eléctrico es aquel que ejecuta esta función principal, pero no exclusivamente, por medio de dispositivos o fenómenos eléctricos. Si la comunicación es eventual y en un solo sentido

se dice quese envía un mensaje. Si es interactiva se dice que existe un diálogo compuesto de mensajes sucesivos en uno y otro sentido.

Información es la esencia de la comunicación, es la diferencia entre el conocimiento previo y posterior a la comunicación. La información recibida no solo depende de los mensajes recibidos sino del desconocimiento previo del interlocutor que la recibe.

Hay muchas clases de fuentes de información , por eso los mensajes aparecerán en muchas formas: una secuencia de símbolos o letras discretas como palabras escritas en forma telegráfica o las perforaciones de una tarjeta IBM, una magnitud sencilla variando con el tiempo como la presión acústica producida por la voz o la música, las funciones del tiempo como la intensidad de la luz y el color de una escena de televisión. Sea cual fuere el mensaje, el objeto de un sistema de comunicación es proporcionar una réplica aceptable de él en su destino.

En la figura 1.1 se muestran los elementos funcionales de un sistema completo de comunicación.

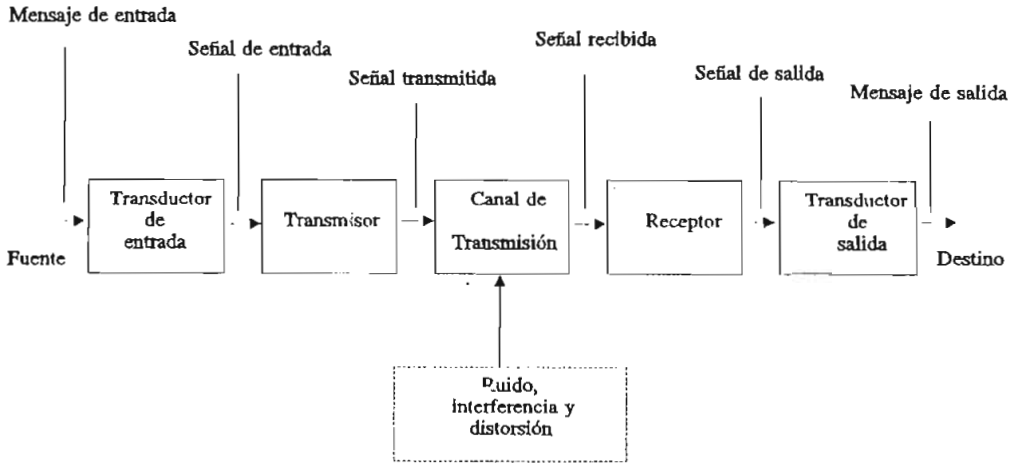


Figura 1.1. Sistema básico de Comunicación

El mensaje producido por la fuente no siempre es eléctrico, por lo tanto es necesario un transductor de entrada. Este transductor convierte el mensaje en una señal, una magnitud eléctrica, tal como un voltaje o una corriente. Similarmente, otro transductor en el destino convierte la señal de salida a la forma apropiada del mensaje. Tanto la señal como el mensaje son la materialización física de la información.

Sin tomar en cuenta a los transductores, hay tres partes esenciales en un sistema de comunicación eléctrica, el transmisor, el canal de transmisión y el receptor, teniendo cada uno su función característica.

Transmisor: El transmisor pasa el mensaje al canal en forma de señal. Para lograr una transmisión eficiente y efectiva, se deben desarrollar varias operaciones de procesamiento de la señal, siendo una de ellas la modulación, un proceso que se distingue por el acoplamiento de la señal transmitida a las propiedades del canal, por medio de una onda portadora.

Canal de transmisión: El canal o medio de transmisión es el enlace eléctrico entre el transmisor y el receptor. Puede ser un par de alambres, un cable coaxial, una onda de radio o un rayo láser. Sin importar el tipo, todos los medios de transmisión se caracterizan por la atenuación, esto es la disminución progresiva de la potencia de la señal conforme aumenta la distancia. La magnitud de la atenuación puede ser pequeña o muy grande.

Receptor: La función del receptor es extraer del canal la señal deseada y entregarla al transductor de salida. Como las señales son frecuentemente muy débiles, como resultado de la atenuación, el receptor debe tener varias etapas de amplificación.

En el caso del proceso de modulación del transmisor, se ejecutará en el receptor la demodulación o detección, con lo cual volverá la señal a su forma original.

1.1.2. Sistema de Transmisión Digital

En general un sistema de Transmisión Digital está compuesto por mensajes digitales o binarios. Esta forma de transmitir se ha convertido rápidamente en la más usual, utilizando directamente el mensaje en forma digital, como en el caso de la salida de una computadora, o convirtiéndolo a una forma digital, como por ejemplo, la transmisión de datos de telemetría y actualmente en las comunicaciones telefónicas.

Por un mensaje binario o una señal binaria se entiende una secuencia de pulsos o formas de onda conocidas, que se presentan a intervalos regularmente espaciados en el tiempo, tal como se muestra en la figura 1.2.

Aunque la forma de los pulsos se supone conocida, la ocurrencia de los mismos, es decir, del 1_L o del 0_L no es conocida de antemano y la información transmitida está dada por la secuencia particular de unos y ceros que llegan.

Los pulsos se muestran presentándose regularmente cada $\mathbf{T_0}$ segundos. $\mathbf{T_0}$ se conoce comúnmente como intervalo binario y se dice que la fuente de señal está generando $1/\mathbf{T_0}$ dígitos binarios o bits por segundo. Por ejemplo si $\mathbf{T_0}=10^{-3}$s, entonces $1/\mathbf{T_0}$ = 1000 bps. Si $\mathbf{T_0}$ = 1 μs, $1/\mathbf{T_0} = 10^6$ bps.

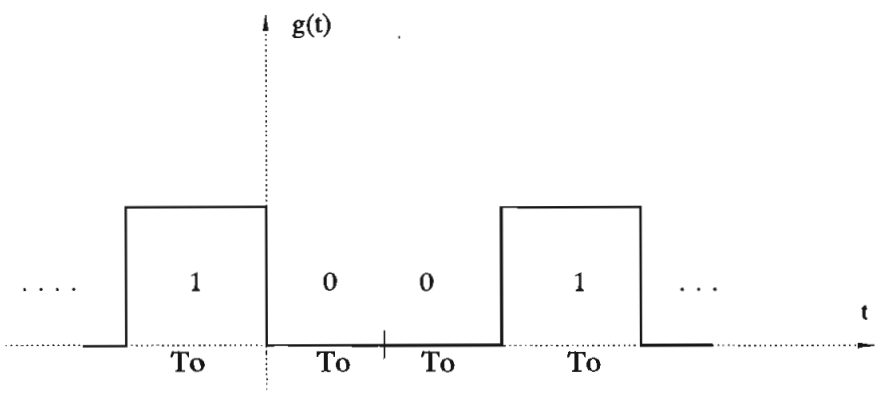


Figura 1.2. Secuencia de dígitos binarios

En la figura 1.3 se muestran las unidades básicas comprendidas en un sistema de Transmisión Digital. No todos los sistemas incluyen las operaciones indicadas, aunque si emplean siempre algún medio de transmisión de alguna clase.

El codificador elige la mejor forma de la señal para optimizar su detección en la salida. El decodificador efectúa la operación inversa para tomar la mejor decisión, de que un mensaje dado fue efectivamente enviado basado en las señales disponibles.

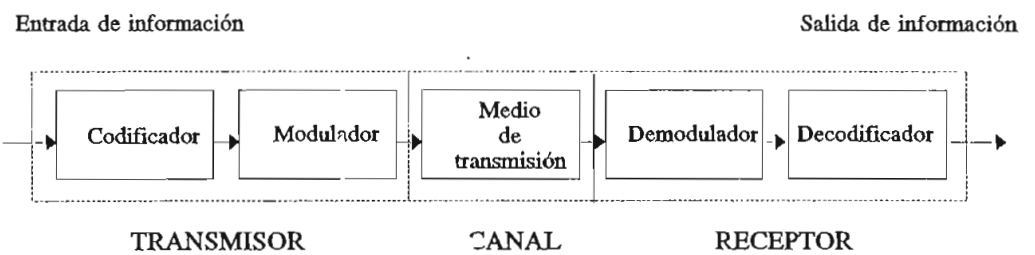


Figura 1.3. Sistema básico de Transmisión Digital

El diseño del codificador y el decodificador debe basarse en una detallada descripción matemática de la transmisión de información.

El modulador produce una señal variable a su salida que es proporcional, de algún modo, a la señal que aparece en sus terminales de entrada. Por ejemplo, un modulador senoidal puede variar la amplitud, la frecuencia o la fase de una señal senoidal en proporción directa al voltaje de entrada. Las funciones del codificador y del modulador son semejantes en que ambos preparan la señal para una más eficiente transmisión. Sin embargo, el proceso de codificación está concebido para optimizar el control de errores en un mensaje que está siendo transmitido, mientras que el proceso de modulación está diseñado para enviar la señal de información sobre la onda portadora. El demodulador realiza la operación inversa a la del modulador para restaurar la señal en su forma original.

El medio de transmisión es la piedra angular del sistema, sin él no existiría comunicación. El medio de transmisión puede incluir la ionósfera, la tropósfera, el espacio libre o simplemente una línea de transmisión. En todo caso, el uso de este medio introduce atenuación y distorsión, sumándose a esto las señales de ruido generadas en los medios y en los equipos de transmisión y recepción.

1.2. GENERACION DE UNA SEÑAL BINARIA

En general la información a transmitirse en un sistema de transmisión de datos es digital. La transmisión digital presenta como principal desventaja la de requerir un mayor ancho de banda. Si se tiene un pulso rectangular, para determinar su ancho de banda se deberá realizar el análisis espectral del pulso, ya que todos los sistemas de transmisión digital producen en último término señales en forma de pulso.

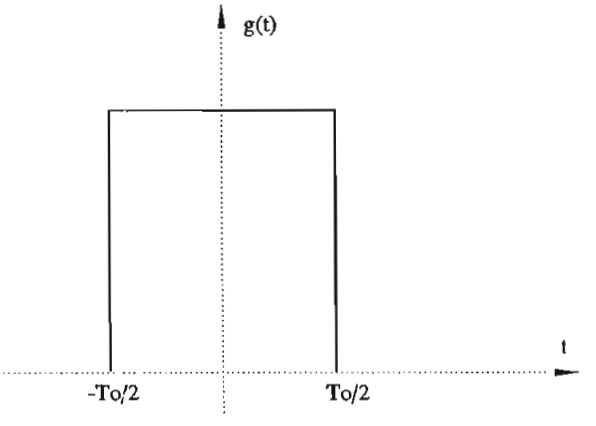


Figura 1.4. Pulso rectangular

El espectro de frecuencia del pulso rectangular de la figura 1.4, viene dado por la expresión:

$$G(\omega) = T_o \, Sen \, \frac{(\omega T_o/2)}{(\omega T_o/2)} \qquad (1.1)$$

donde $\omega = 2\pi f_0$ es la frecuencia angular y T_0 es la duración de un intervalo de la señal.

La forma espectral del pulso está indicada en la figura 1.5, donde se indica la potencia espectral total para varios anchos de banda. La mayor cantidad de energía se encuentra en la primera componente espectral es decir para $f_0 = 1/T_0$.

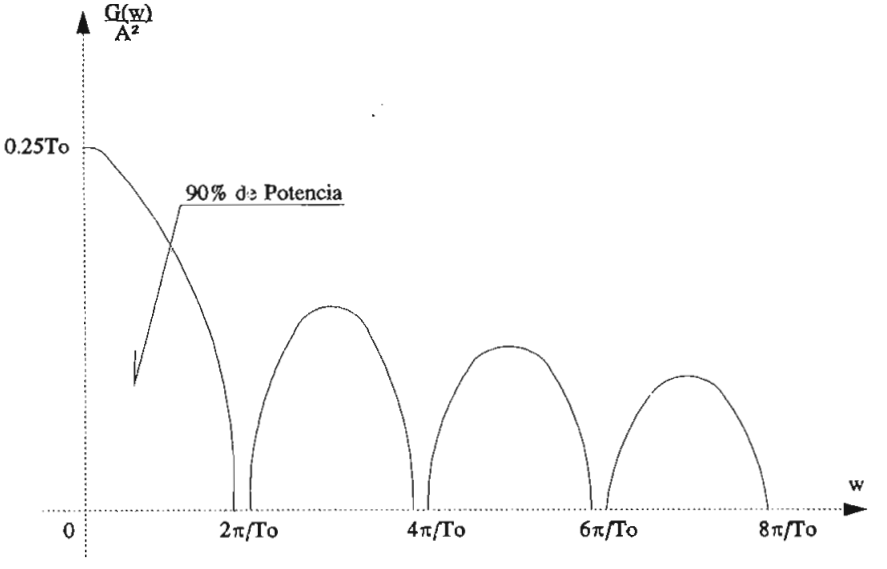


Figura 1.5. Espectro de potencia del pulso rectangular

El alto porcentaje de energía dentro de esta banda indica que la señal puede ser confinada a un ancho de banda de $1/T_0$. Como en una señal digital se considera que la señal conmuta entre 1_L y 0_L, el ancho de banda será $1/2T_0$.

De lo anteriormente expuesto se deduce la ecuación del Teorema de Nyquist:

$$R_{max} = 2\ B \qquad (1.2)$$

donde R_{max} es el ritmo de la señal binaria y es igual a $1/T_0$ y

B es el ancho de banda. Esta ecuación determina que el máximo ritmo de la señal binaria, a través de un ancho de banda pasabajos sin ninguna interferencia intersímbolo, es igual a dos veces el ancho de banda. Si se desea transmitir $R = 1/\mathbf{T}_0$ bps, en un canal de banda base teniendo un ancho de banda de B Hz, el sistema de transmisión es considerado más efectivo si, en un ancho de banda dado, se pueden transmitir más bits por segundo (bps).

Si la velocidad de transmisión se normaliza para un ancho de banda de B = 1Hz, entonces, la eficiencia del sistema está dada en términos de bits por segundo por hertzio (bps/Hz); esto es muy conveniente y frecuentemente empleado para la representación de un sistema de transmisión digital.

En el teorema de Nyquist, se demuestra que para un sistema de transmisión de ancho de banda mínimo, es posible transmitir R símbolos por segundo independientes en un canal (filtro pasabajos), teniendo un ancho de banda de solo $B = R/2$ Hertzios y alcanzando sus valores completos en los instantes de muestreo. Esto en la práctica no se cumple ya que el filtro de Nyquist es ideal y produce una degradación (además se necesitaría una sincronización precisa entre el transmisor y receptor). En realidad el canal de comunicación tiene un ancho de banda limitado, lo cual hace que los pulsos transmitidos se dispersen. Esta dispersión del pulso produce un solapamiento entre los pulsos adyacentes, dando lugar a una distorsión conocida como interferencia entre símbolos (**ISI**).

A menos que esta interferencia sea compensada, el efecto en el receptor podría dar lugar a decisiones erróneas.

Un método de controlar la interferencia intersímbolos es conformar los pulsos transmitidos a una manera apropiada. Para comprender este problema consideremos una secuencia binaria $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$; transmitida en intervalos de $\mathbf{T}_0$ segundos. Estos dígitos ingresan al canal cuya respuesta impulsiva es $h(t)$.

La señal recibida se puede escribir como:

$$g(t) = \sum_{n=-\infty}^{\infty} a_n X(t-nT_o) \qquad (1.3)$$

Una manera obvia de restringir la interferencia intersímbolo (ISI) es tratando que la respuesta impulsiva $h(t)$ de este filtro sea igual a cero en los instantes de muestreo nT_o excepto en n = 0.

Una forma de pulso que produzca una interferencia intersímbolo (ISI) nula es la función:

$$h(t) = \frac{Sen(\frac{\pi t}{T_o})}{(\frac{\pi t}{T_o})} \qquad (1.4)$$

Esta es la respuesta impulsiva de un filtro ideal, como se muestra en la figura 1.6.

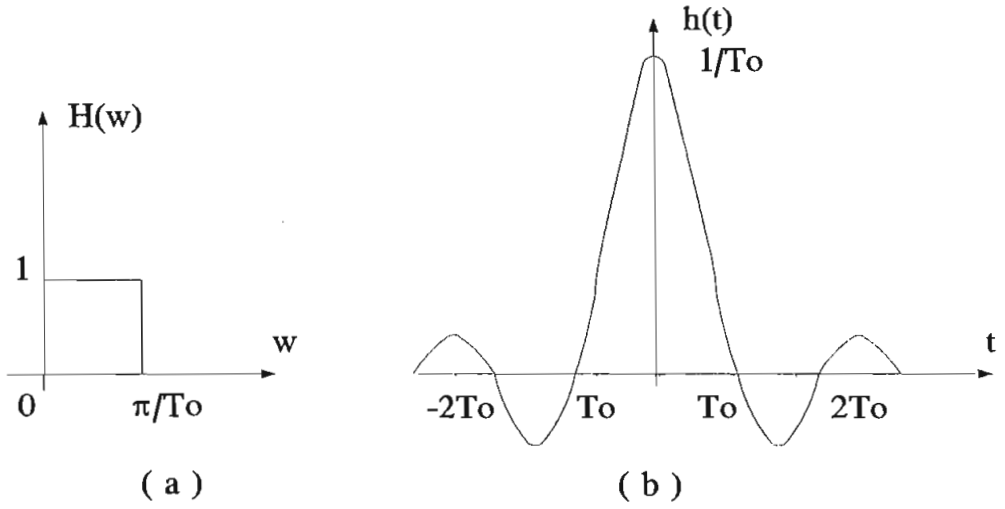


Figura 1.6.a) Característica de frecuencia del filtro
b) Respuesta impulsiva $(Sen\ X)/X$

Se debe notar que $h(t)$ es 0 en intervalos igualmente espaciados (intervalos de muestreo) que son múltiplos de $\mathbf{T}_0$.

Se observa que el ancho de banda B es $\pi/\mathbf{T}_0$ (rad/seg) o $1/2\mathbf{T}_0$ Hertz.

Sin embargo, en la práctica existen dificultades con este tipo de filtro, primero porque es físicamente irrealizable y segundo porque esta forma de onda depende principalmente de la precisión en la temporización.

La variación de la temporización en el receptor podría resultar en una excesiva ISI a causa del lento decrecimiento de las colas del pulso $(Sen\ X)/X$.

El filtro pasabajos ideal se puede modificar para dar una clase de forma de onda descrita por Nyquist que cumpla el requerimiento de ISI nula, pero que es más simple de obtener en la práctica.

Nyquist sugirió obtener la frecuencia de corte en una manera gradual, con un filtro cuya característica tenga simetría impar alrededor del punto de corte del filtro pasabajos ideal. La satisfacción de dicha característica dio lugar a un conjunto de formas de pulso de Nyquist que cumplían el requerimiento de tener ceros a intervalos igualmente espaciados. Para mantener una velocidad de $2B$ pulsos por segundo, este conjunto de características de filtro requiere de un ancho de banda adicional sobre el denominado ancho de banda de Nyquist definido por el filtro pasabajos ideal en la figura 1.6.

Un ejemplo de filtro comúnmente utilizado que cumple este criterio de Nyquist es la característica "roll-off" de coseno alzado, llamado también de coseno levantado, cuyo espectro tiene una magnitud plana en las frecuencias bajas y una porción progresiva de forma senoidal, como se muestra en la figura 1.7.

La característica del filtro "roll-off" de coseno alzado está dado por:

$$H(\omega) = \begin{cases} 1 & [0 \le \omega \le \frac{\pi}{T_o}(1-\alpha)] \\ Cos^2\{\frac{T_o}{4\alpha}[\omega - \frac{\pi}{T_o}(1-\alpha)]\}, & [\frac{\pi}{T_o}(1-\alpha) \le \omega \le \frac{\pi}{T_o}(1+\alpha)] \\ 0 & [\omega > \frac{\pi}{T_o}(1+\alpha)] \end{cases} \qquad (1.5)$$

El parámetro α, se define como la cantidad de ancho de banda en exceso respecto al ancho de banda de Nyquist. La respuesta impulsiva correspondiente es:

$$h(t) = \left(\frac{Sen(\frac{\pi t}{T_o})}{(\frac{\pi t}{T_o})}\right)\left\{\frac{Cos\alpha(\frac{\pi t}{T_o})}{1-(\frac{4\alpha^2 t^2}{T_o^2})}\right\} \qquad (1.6)$$

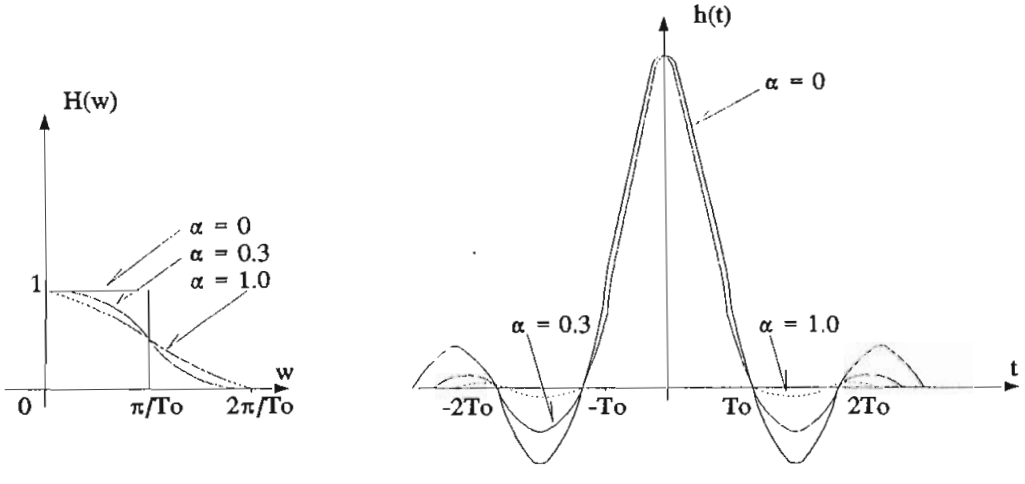


Figura 1.7. Característica de filtro "roll-off" de coseno alzado y forma del pulso

En la figura 1.7 se muestra un gráfico de $H(\omega)$ y $h(t)$ para diversos valores de α. Nótese que para el caso de $\alpha = 0$ coincide con el filtro pasabajos ideal mostrado en la figura 1.6. El caso $\alpha = 1$ corresponde al "roll-off" coseno alzado completo (100%), que duplica el ancho de banda de Nyquist. Como ejemplo se tiene que en un canal normalizado con ancho de banda $B = 1$ Hz es posible transmitir 2 (bps/Hz) con $\alpha = 0$, mientras que si se tiene un factor de " roll-off " $\alpha = 0.3$ se tiene (2/1.3) = 1.54 (bps/Hz).

1.3. TRANSMISOR DIGITAL

En la figura 1.8 se observa una descripción más detallada de un sistema básico de Transmisión Digital. El papel del transmisor es preparar la información para enviarla de forma tal que pueda superar lo mejor posible las limitaciones impuestas por el canal.

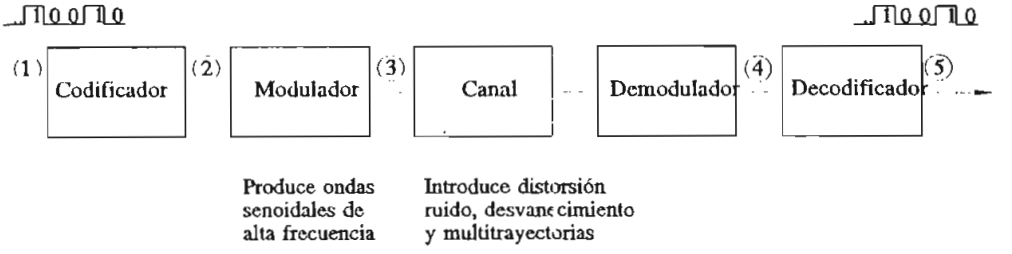


Figura 1.8. Diagrama detallado de un Sistema de Transmisión Digital

El proceso de modulación es necesario para permitir que las señales sean efectivamente irradiadas al espacio (o a cualquier otro medio representado por el canal mostrado).

Algunas formas de onda típicas, correspondientes a

los puntos numerados de la figura 1.8, se muestran esquemáticamente en la figura 1.9.

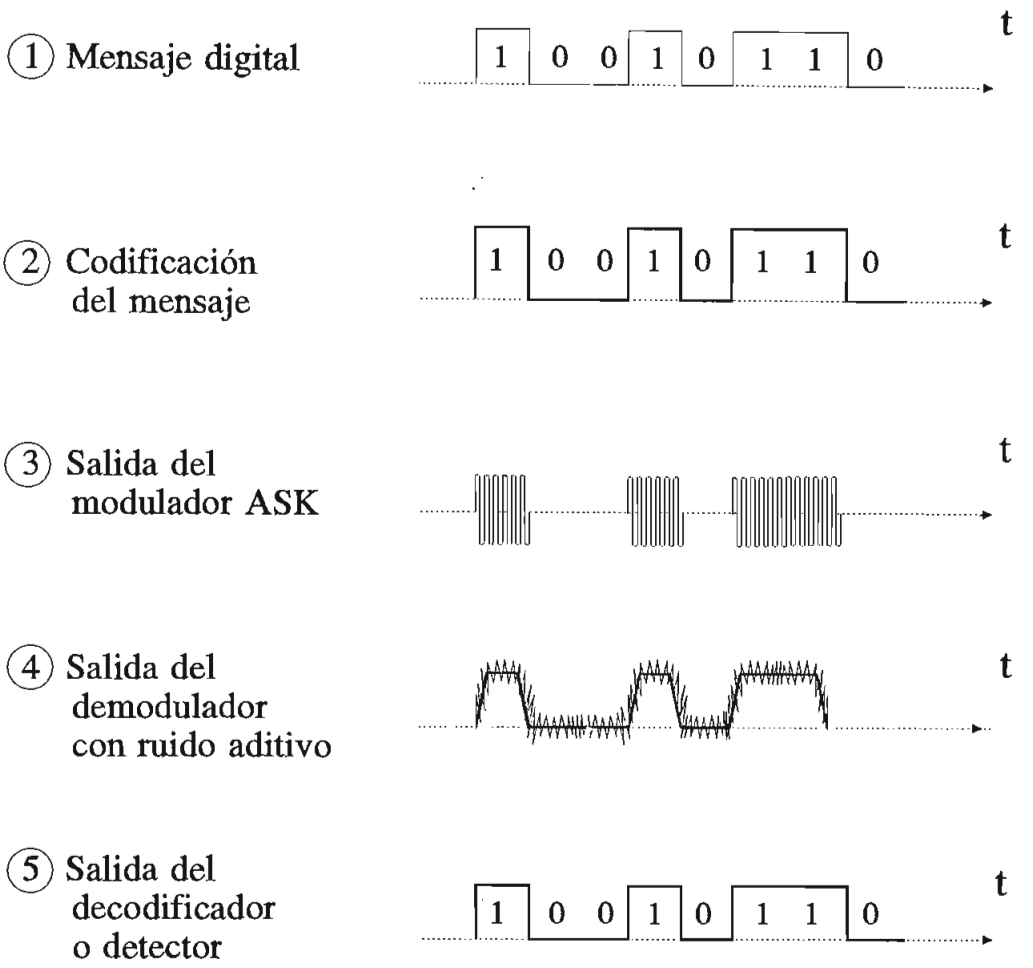


Figura 1.9. Formas de onda en el trayecto de un Sistema de Transmisión Digital

El 1_L del mensaje digital es codificado por un pulso, y el 0_L por ausencia de éste, denominándose a este proceso Codificación en banda base NRZ[1].

El modulador que se ha usado resulta ser del tipo de modulación en amplitud ASK[2], en el cual un oscilador de onda senoidal ajusta su amplitud de acuerdo a la señal de entrada.

[1]Codificación en banda base NRZ, Capítulo 3, numeral 3.3.1.

[2]Modulación digital de amplitud ASK, Capítulo 4, numeral 4.2.1.

1.4. CANAL Y MEDIO DE TRANSMISION

El conjunto de factores que intervienen en la transmisión, incluyendo el medio físico se llama Canal de Transmisión.

Un canal es el medio más el transmisor y el receptor. Un circuito puede tener un sólo canal fijo y servir sólo para comunicar en un sentido como sucede en televisión. Este circuito se llama simplex (SX).

En muchos casos es deseable mantener una comunicación en dos sentidos o, al menos, poder regresar un mensaje a su origen para una posible verificación, una comparación o un control. Una manera de obtenerlo es utilizando el mismo canal alternadamente para transmitir en ambas direcciones, denominándose a este circuito semi-dúplex o half dúplex (HDX). Aunque la comunicación fluye en ambas direcciones, en un momento dado el flujo de información es en un sólo sentido.

Finalmente, un tercer tipo de circuito lo constituye el dúplex total o full dúplex (FDX). Este circuito tiene dos canales simultáneos, en este caso la información fluye en ambas direcciones en forma independiente y al mismo tiempo. Nótese que, tanto en la transmisión HDX como en la FDX, los moduladores y demoduladores operan en parejas. Esta combinación de modulador y demodulador se llama **MODEM** (**MO**dulador-**DE**modulador) en los sistemas de transmisión de datos. También los codificadores y los decodificadores trabajan en pares, llamándose a esta configuración **CODEC** (**CO**dificador-**DEC**odificador).

1.4.1. Capacidad del canal

Un requisito indispensable en un sistema de transmisión es el empleo de un ancho de banda que contenga las frecuencias de las señales. En un canal de transmisión el único perjuicio del mismo proviene del ruido aditivo. Un ancho

de banda mayor que el requerido permitiría más interferencia del ruido, por lo que es importante mantener el ancho de banda de dicho canal tan reducido como sea posible.

El ruido presente se caracteriza por su potencia media N y la señal transmitida por su potencia media S. Si la potencia media del ruido es relativamente pequeña, la potencia de la señal no necesita ser muy grande para que el receptor determine qué información se está enviando. Por el contrario la potencia media de la señal debe ser relativamente grande cuando la potencia del ruido es grande; de esto se deduce que lo que importa es la razón de la potencia media de la señal a la potencia media del ruido S/N (signal-to-noise ratio).

Todos los sistemas de transmisión pueden juzgarse en términos de ancho de banda, relación señal a ruido y factores económicos, siendo el más importante el compromiso entre ancho de banda y relación señal a ruido. Se puede decir que la cantidad de errores disminuirá si aumenta S/N. También parece razonable que podrían reducirse los errores con un receptor diseñado para procesar señales de mayor complejidad. En consecuencia el interés se centra tanto en la eficiencia como en la precisión de la comunicación.

Surge la pregunta: Si para un canal dado con una rapidez de información dada, es posible mejorar el sistema con el objeto de reducir los errores?. El trabajo teórico de Claude Shannon[1] responde a este cuestionamiento diciendo:

"Si la rapidez de información es tal que R < C donde C es la capacidad del canal, la capacidad viene dada por la ley de Hartley-Shannon":

$$C = B \log (1 + S/N) \quad bps. \qquad (1.7)$$

donde B es el ancho de banda del canal en Hz, y S/N es la relación señal a ruido, expresada en forma numérica.

[1] Communication in the Presence of Noise, Proceedings of the IRE, Vol.37, Enero de 1979, p. 10-21

Si se intenta enviar información con demasiada rapidez, es decir $R > C$ los errores empiezan a aumentar rápidamente y no tiene sentido tratar de diseñar un sistema para mejorar la situación. Si se quiere aumentar la rapidez se debería aumentar la capacidad del canal y esto determinaría un aumento o del ancho de banda o de la relación señal a ruido; sin embargo lo sorprendente de la ecuación es que la misma establece que las dos variables pueden intercambiarse. Por tanto si el ancho de banda se aumenta puede ser adecuada una razón S/N menor y viceversa.

Se deduce que la ley de Hartley-Shannon es aplicable a sistemas tanto continuos como discretos, no obstante su aplicación se restringe a canales con ruido aditivo y no se incluyen efectos tales como la distorsión y la interferencia. Aunque la ley de Hartley-Shannon establece que existen cotas para la máxima rapidez de información en un canal dado, y sugiere que el ancho de banda puede intercambiarse por S/N, no ofrece un método para el diseño de un sistema que cubra estos requisitos; en otras palabras fija una cota con la que puede compararse el comportamiento de los sistemas que se diseñan pero no da un procedimiento para diseñar sistemas cuyo comportamiento se ajuste a esa cota.

1.5. RECEPTOR DIGITAL

El papel del receptor es efectuar las operaciones inversas del transmisor para recuperar la información con la menor cantidad de errores posible. En un sentido amplio, el transmisor y el receptor en pareja están diseñados específicamente para combatir los efectos perniciosos del canal en la transmisión de información.

El demodulador en el receptor sirve para separar la modulación de la onda senoidal de alta frecuencia, que se ha formado en el modulador del transmisor.

La función del detector en el receptor es reproducir "tan fielmente como sea posible", la secuencia original de señales que representan los datos digitales que se transmiten.

En la figura 1.8 se mostró el canal como si el ruido se lo introdujera durante la transmisión, de manera que la salida del demodulador representa la suma de las formas de onda de las señales y del ruido. Se debe notar que el ruido introducido tiende a ocultar la señal si sus magnitudes son comparables. Algunos canales (la línea telefónica, por ejemplo) producen distorsión en la señal en forma paralela.

Otros canales introducen desvanecimiento "fading", en los cuales la amplitud de la señal recibida fluctúa aleatoriamente. Estas alteraciones se denominan también efectos multitrayectoria, que consisten en que la energía radiada por el transmisor para un símbolo, sigue varias trayectorias para llegar al receptor, donde aparece como una secuencia de símbolos recibidos. Ejemplos de este último tipo de canales con desvanecimiento aleatorio, incluyen las transmisiones de radio de onda corta a través de la ionósfera, así como las comunicaciones submarinas y sísmicas, entre otras.

1.6. RUIDO, INTERFERENCIA, ANOMALIAS QUE AFECTAN LA TRANSMISION DE UNA SEÑAL DIGITAL

Durante la transmisión de la señal ocurren ciertos efectos no deseados. Uno de ellos es la atenuación, la cual reduce la intensidad de la señal; sin embargo, son más serios la distorsión, la interferencia y el ruido, los cuales se manifiestan como alteraciones de la forma de la señal. Al introducirse estas contaminaciones al sistema, es una práctica común y conveniente imputárselas al canal, pues el transmisor y el receptor son considerados ideales.

En términos generales, cualquier perturbación no intencional de la señal se puede clasificar como "ruido", y

algunas veces es difícil distinguir las diferentes causas que originan una señal contaminada. Existen buenas razones y bases para separar esos tres efectos, de la manera siguiente:

Distorsión.- Es la alteración de la señal debida a la respuesta imperfecta del sistema a ella misma. A diferencia del ruido y la interferencia, la distorsión desaparece cuando la señal deja de aplicarse. El diseño de sistemas perfeccionados o redes de compensación reduce la distorsión. En teoría es posible lograr una compensación perfecta. En la práctica debe permitirse cierta distorsión, aunque su magnitud debe estar dentro de límites tolerables.

Interferencia.- Es la contaminación por señales extrañas, generalmente artificiales y de forma similar a las de la señal. El problema es particularmente común en emisiones de radio, donde pueden ser captadas dos o más señales simultáneamente por el receptor. La solución al problema de interferencia es obvia: eliminar en una u otra forma la señal interferente o su fuente. En este caso es posible una solución perfecta si bien no siempre práctica.

Ruido.- El desempeño de un sistema de comunicaciones depende de la exactitud con lo cual puede determinarse en el receptor la señal transmitida. Esta exactitud se encuentra defectivamente limitada por la presencia del ruido. Por ruido se debe entender las señales aleatorias e impredecibles de tipo eléctrico originadas en forma natural dentro o fuera del sistema.

Las fuentes y orígenes del ruido son numerosas admitiendo clasificaciones diversas. Muchos pueden eliminarse o al menos reducirse a niveles prácticamente impredecibles. No obstante existen algunos tipos de ruido que por su naturaleza resultan inevitables a pesar de los arduos esfuerzos desplegados en su eliminación.

1.6.1. Ruido térmico

Una de las causas inevitables de ruido es la agitación térmica de los electrones por lo cual se lo denomina ruido térmico. El mismo contamina la señal en forma aditiva, siendo la modulación digital una de las técnicas más contundentes de combatirlo.

En la mayoría de los casos prácticos el ruido térmico se comporta como una variable aleatoria y gaussiana.

Interesará caracterizar este ruido en condiciones de adaptación, es decir con la máxima potencia que puede entregar la fuente de ruido a la carga. Más que la potencia total interesará la densidad espectral de potencia disponible en condiciones de adaptación, la cual está dado por:

$$Gn(f) = \frac{1}{2} K \cdot T \qquad (watt/Hz) \qquad (1.8)$$

donde: $K = 1,37 \times 10^{-23}$ (joule/°K) y es la constante de Boltzman.

$Gn(f)$ = Densidad espectral de potencia del ruido

T = Temperatura (°K)

1.6.2. Distorsión de forma de onda, interferencia entre símbolos

La interferencia entre símbolos es producida por las características pasabanda del medio de transmisión. Su valor es la suma de los voltajes interferentes, generados en el instante de decisión por las formas de onda distorsionadas de los pulsos adyacentes. Su valor está afectado por el proceso estocástico puesto que el tren de pulsos *toma* valores aleatorios.

1.6.2.1. Diagrama de ojo

Las imperfecciones del canal se evalúan frecuentemente por medio del diagrama de ojo, que es una técnica de medición experimental para determinar la cantidad promedio de interferencia.

Se denomina así, porque su representación se parece a un ojo humano. En la figura 1.10.(a), la señal $V_i(t)$ es una señal NRZ balanceada aplicada a la entrada del filtro pasabajos ideal de Nyquist, la señal $V_o(t)$ será como la representada en la figura 1.10.(b). El máximo valor de cada bit transmitido se lo alcanzará en los instantes de muestreo $\mathbf{c}(t-nT_o)$, pero como el filtro en la realidad es imperfecto la señal de salida $V_o(t)$ será como la línea entrecortada de la figura 1.10.(b), en donde se observa que en los instantes de muestreo el bit de transmisión no tiene su máximo valor, lo que introduce ISI.

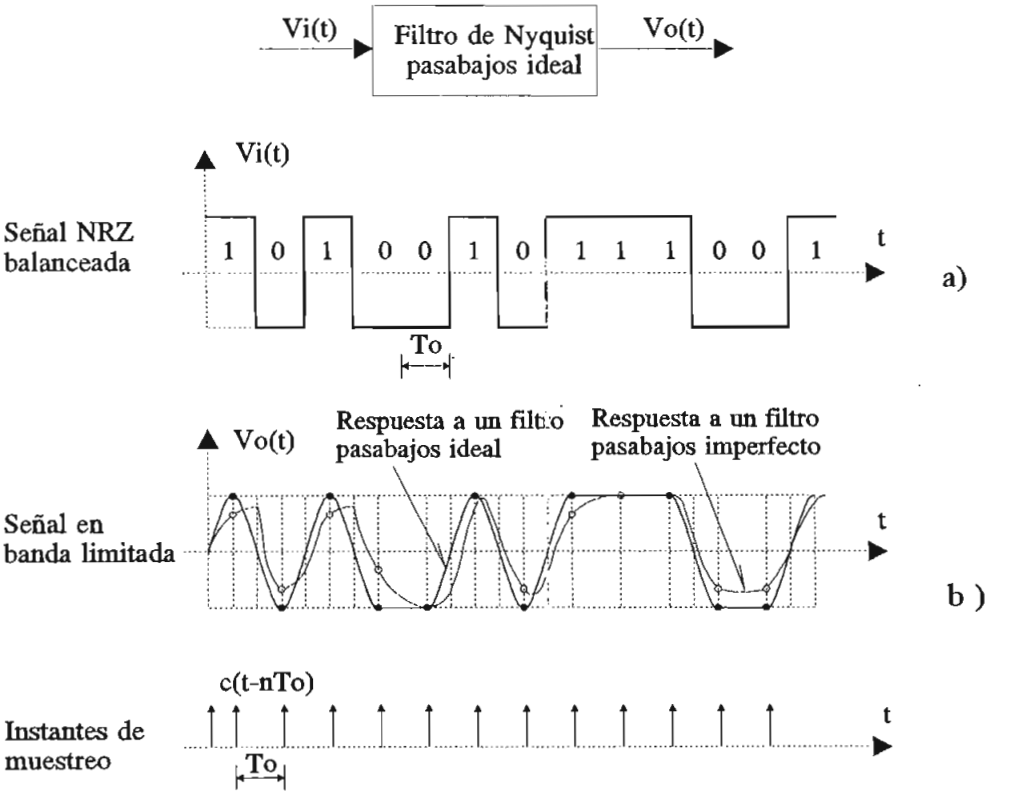


Figura 1.10.a) Señal de entrada al filtro de Nyquist
b) Señal de salida

El diagrama de ojo se puede obtener en un osciloscopio, tal como se indica en la figura 1.11. La señal $V_o(t)$ de la figura 1.10.(b) se alimentará a la entrada vertical del osciloscopio, y a la entrada horizontal se pondrá la señal de muestreo **c**(t-nTs), siendo el barrido de tiempo horizontal igual a la duración de un bit.

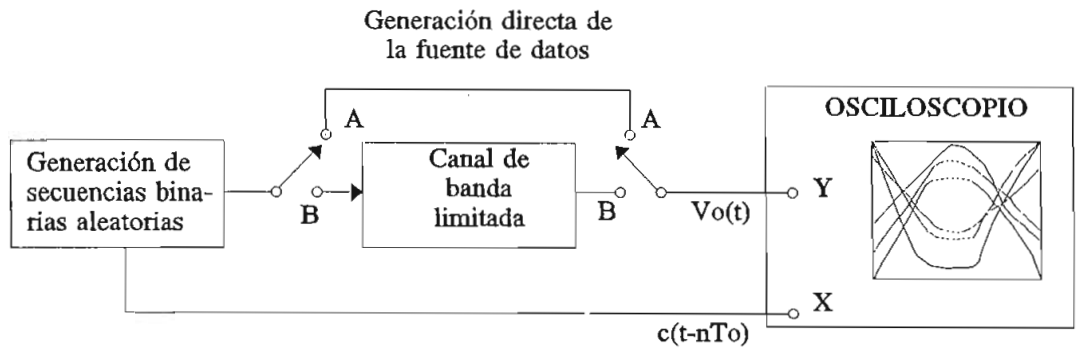


Figura 1.11. Obtención del Diagrama de ojo

El momento para muestrear la onda recibida es cuando la apertura del ojo es mayor.

La presencia de ISI también hace que se produzca el cierre del ojo, y la cantidad relativa de cierre en el mejor momento de muestreo da una indicación de la degradación causada por la ISI.

En realidad no sólo se debe considerar las limitaciones del ancho de banda, sino además la presencia de ruido que provocará que el diagrama de ojo se cierre y se vuelva más difuso.

Además del análisis de la ISI, el diagrama del ojo presenta mayor información de las características de un sistema de transmisión digital, tal como se puede observar en el modelo de la figura 1.12, donde se han señalado los puntos de referencia más importantes para el análisis de dicho esquema.

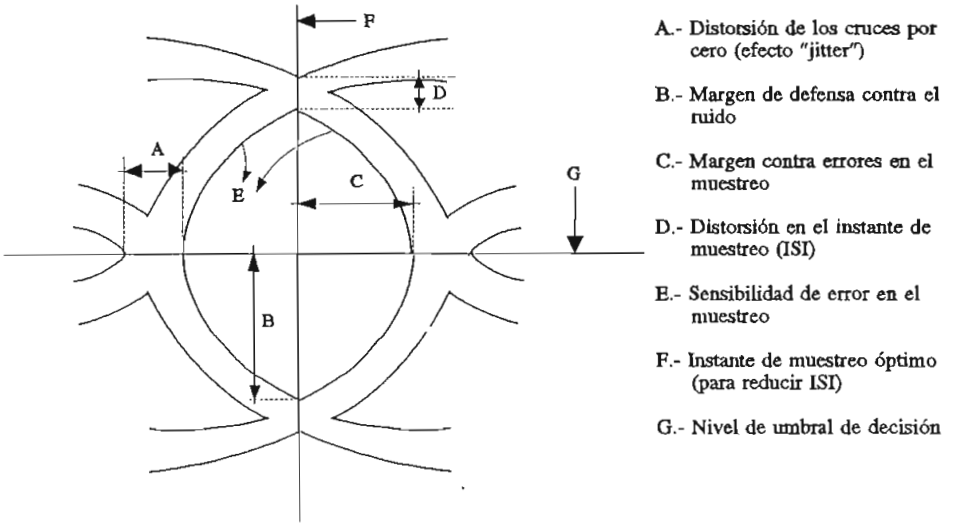


Figura 1.12. Modelo de diagrama de ojo

Si la relación señal a ruido S/N en el receptor es alta, entonces, se pueden realizar las siguientes observaciones del diagrama del ojo:

1.- El mejor tiempo para muestrear la forma de onda recibida es cuando la apertura del ojo es mayor (F).

2.- La máxima distorsión se indica por el ancho vertical de las 2 ramas en el tiempo de muestreo (D).

3.- El margen de ruido o inmunidad al ruido es proporcional al ancho de la apertura del ojo (B).

4.- La sensibilidad del sistema a errores de temporización se indica por la proporción del cierre del ojo.

5.- El tiempo de muestreo está a mitad de camino entre cruces por cero; si la señal de reloj es derivada de los cruces por cero entonces la cantidad de distorsión de cruces por cero indica la cantidad de "Jitter"[1] o variaciones en la velocidad del reloj y fase (A).

[1] El concepto de "Jitter" es explicado en el numeral 1.6.4.

6.- Las asimetrías en el diagrama del ojo indican alinealidades en el canal, dado que en un sistema estrictamente lineal con datos verdaderamente aleatorios todas las aperturas del ojo serán idénticas (E).

1.6.3. Extracción de la temporización

1.6.3.1. Repetidor Regenerativo

Una de las ventajas de los sistemas digitales sobre los analógicos es su capacidad de reconstruir el tren de pulsos de información a lo largo del canal, con el fin de evitar la acumulación de distorsión de la señal y el ruido. El proceso de regeneración lo realizan los repetidores regenerativos que tienen tres funciones:

1.- Dar una nueva forma a los pulsos entrantes por medio de un compensador o ecualizador.

2.- La extracción de la temporización o sincronización que se requiere para muestrear los pulsos entrantes en los instantes óptimos.

3.- La toma de decisiones con base en las muestras de los pulsos.

En la figura 1.13 se representa un repetidor regenerativo, en el cual el tren de pulsos de entrada es atenuado y distorsionado por el medio de transmisión. La distorsión es causada por las componentes de alta frecuencia del tren de pulsos, siendo necesario que el ecualizador del repetidor tenga una característica de frecuencia inversa a la del medio de transmisión, lo que permitirá restaurar las componentes de altas frecuencias y eliminar la distorsión de los pulsos; desafortunadamente el ruido también se incrementa con el ancho de banda al aumentarse sus componentes de alta frecuencia.

---

[1] El concepto de "Jitter" es explicado en el numeral 1.6.4.

Sin embargo para las señales digitales no es muy importante la ecualización, ya que el detector tiene que tomar decisiones relativamente sencillas, tales como decidir si el pulso es positivo o negativo. La distorsión ocasiona errores en la detección de los pulsos.

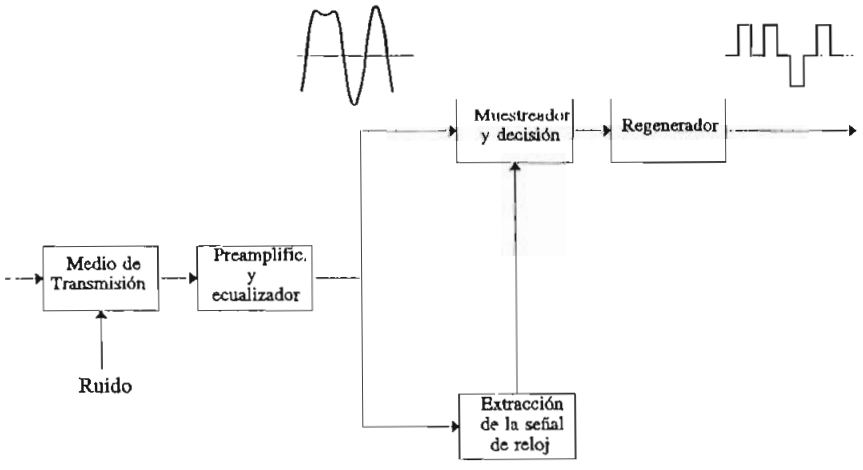


Figura 1.13. Repetidor Regenerativo

Un receptor digital necesita información de temporización para poder interpretar apropiadamente la secuencia de la señal recibida. Cada símbolo de la secuencia debe ser muestreado cuando su valor ha sido completamente establecido.

Esto requiere de una señal de reloj del receptor, sincronizada con la señal de reloj del transmisor, lo cual se denomina "sincronización de símbolo" y en el caso de símbolos binarios se denomina "sincronización de bit". En un sistema tipo serie donde grupos de símbolos se usan para representar un caracter, es también necesario determinar los límites de tales grupos. A esto lo llamaremos temporización de caracter o grupo. Los caracteres a su vez están organizados para producir un mensaje.

Es así necesario reconocer el arranque y el final de estos mensajes. Los varios tipos de información de temporización deben ser transportados por el mismo canal de

datos o por un canal de temporización auxiliar.

Una señal digital tal como una señal binaria RZ[1] contiene una componente aleatoria y una componente periódica con la misma frecuencia fundamental de reloj. Cuando esta señalización RZ se aplica a un circuito tanque resonante, sintonizado a la frecuencia de reloj, la señal de salida será la señal de reloj que se desea. No todas las señales binarias contienen una componente discreta de la frecuencia de reloj, tal es el caso de la señal bipolar; en estos casos, la temporización se puede extraer utilizando una operación no lineal, para ello una simple rectificación convierte a la señal bipolar en señal RZ, la cual se puede usar para extraer la temporización, estas marcas unipolares son dispuestas para estimular al circuito tanque.

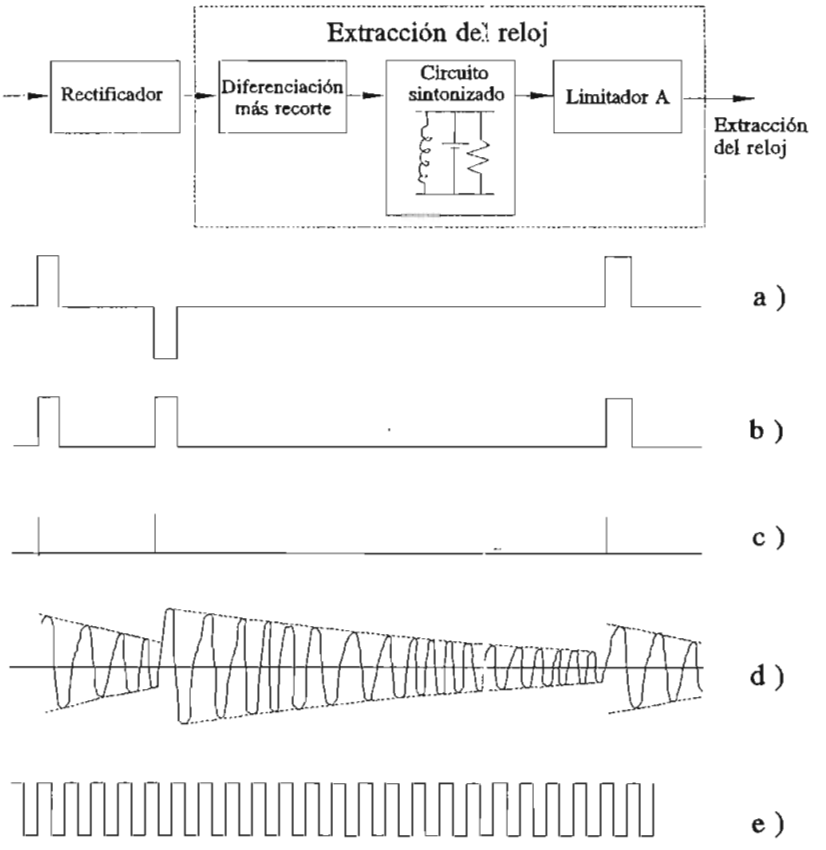


Figura 1.14. Extracción de la temporización
a) Señal bipolar de entrada
b) Señal RZ rectificada
c) Marcas unipolares
d) Excitación del circuito tanque
e) Señal de temporización

Esta operación se puede ver en la figura 1.14 donde es evidente que la distancia máxima entre unos no debe ser muy grande. La frecuencia de resonancia del tanque será exactamente equivalente a la del reloj transmitido originalmente. Aunque la fuente emita los pulsos en los instantes correctos, las operaciones subsecuentes en los repetidores durante la transmisión tenderán a desviar los pulsos de sus posiciones originales. El Q (factor de calidad) del circuito sintonizado que se utiliza para la extracción de la temporización debe ser suficientemente grande para que proporcione una supresión adecuada de la variación de temporización. El circuito tanque oscila con una amplitud que decae exponencialmente, hasta que la siguiente marca estimula el tanque nuevamente.

1.6.4. Irregularidades de la temporización o "Jitter"

Las fuentes de mensajes digitales emiten casi siempre los elementos de la señal a una tasa uniforme, pero durante la transmisión se tiende a perder esta uniformidad y los elementos de la señal llegan de una manera ligeramente irregular. Cuando la señal de reloj de la señal recibida, que es la que sincroniza con la señal transmitida regenerada, varía en fase se llama a este fenómeno efecto **"Jitter"**.

El "Jitter" es producido en los repetidores regenerativos, ya que la frecuencia del circuito tanque sintonizado es diferente a la de los datos recibidos que tienen un elemento de tiempo asociado, provocándose diferencias entre los impulsos de tiempo y el circuito sintonizado. Otra de las causas para que se produzca fluctuación de fase o "Jitter" es la presencia del ruido impulsivo debido a la interferencia de otros sistemas, separación entre pulsos, etc.

Existen técnicas para reducir la magnitud del "Jitter" que aparece a la salida de un sistema. Una de ellas es prevenir la generación y acumulación sistemáticas del jitter, lo cual involucra registrar la señal en un dispositivo conocido como "data scrambler" (aleatorizador de datos). Este

aleatorizador toma la secuencia binaria entrante, para convertirla en otra secuencia que está sistemáticamente relacionada con el mensaje original, pero que en sentido general es más aleatoria, y evita las secuencias largas de unos y ceros repetitivas.

Esta técnica impide la aparición de un "Jitter" sistemático, que ocurre cuando la señal transmitida cambia de un modelo repetitivo a otro. La desventaja de los "data scramblers" es que introducen errores en la transmisión. En la recepción el mensaje original puede ser recuperado por medio del desaleatorizador.

1.6.5. Probabilidad de error en la detección

La señal recibida en el detector consta del tren de pulsos que se ha transmitido más un ruido de canal aleatorio, dando lugar a errores en la detección de los pulsos.

Se considera el caso de la transmisión Polar[1], usando un pulso básico $p(t)$ de amplitud pico $+A$ tal como se observa en la figura 1.15.(a). Un típico tren de pulsos recibidos aparece en la figura 1.15.(b), en la que se observa que los pulsos se muestrean en sus valores pico. Si no existiera ruido, la muestra del pulso positivo que corresponde a 1_L será $+A$ y la del pulso negativo correspondiente a 0_L sería $-A$. Debido al ruido, estas muestras serían $\pm A + n$, donde n es la amplitud del ruido aleatorio. Por la simetría de la situación, el umbral de detección es cero, o sea, si el valor de muestra de un pulso es positivo, el dígito se detecta como 1_L; si el valor es negativo, el dígito se detecta como 0_L.

La decisión de si se transmite un 1_L o un 0_L podría tomarse rápidamente de la muestra del pulso, excepto que n es aleatoria, lo que significa que su valor exacto es impredecible (puede tener un valor grande o pequeño y puede ser tanto negativo como positivo).

[1] Codificación en banda base NRZ Polar, Capítulo 3, numeral 3.3.2.1.

Es posible que se transmita un 1_L, pero n en el instante de muestreo puede tener un valor negativo grande. Esto hará al valor de muestra $+A + n$ pequeño o incluso negativo. Por otra parte, si se transmite un 0_L y n tiene un valor positivo grande en el instante de muestreo, el valor de muestra $-A + n$ puede ser positivo y el dígito se detectará erróneamente como 1_L. Esto se muestra claramente en la figura 1.15.(b).

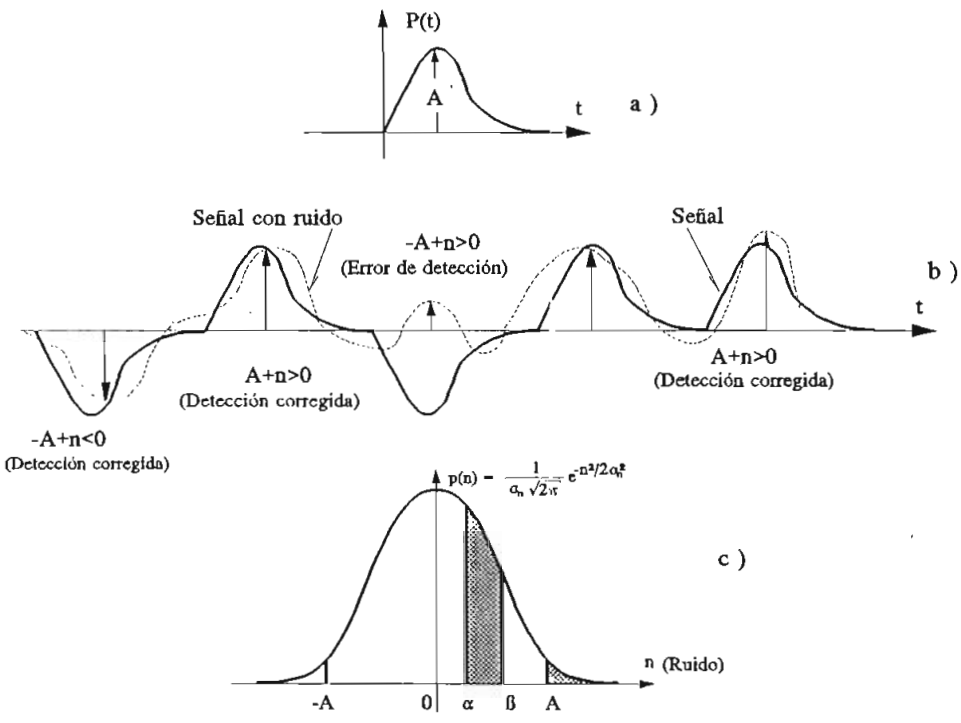


Figura 1.15. Probabilidad de error en detección de umbral
a) Pulso de transmisión
b) Señal Polar recibida
c) Función densidad de probabilidad

Resumiendo n puede tomar valores grandes positivos o negativos dando lugar a errores en la detección. Cuando se transmite un 0_L, el valor de muestra del pulso recibido es $-A + n$. Si $n > A$, el valor de muestra es positivo y el dígito será detectado en forma errónea como 1_L. Si $P(\epsilon \mid 0_L)$ es la verosimilitud o probabilidad de error, dado que se transmite un

0_L, entonces

$$P(\epsilon \mid 0_L) = \text{Probabilidad de que } n > A$$

de manera similar;

$$P(\epsilon \mid 1_L) = \text{Probabilidad de que } n < -A$$

Las probabilidades, en la ecuaciones anteriores, pueden calcularse si se conoce la distribución relativa de las amplitudes del ruido aleatorio. Tal distribución se especifica mediante la función de densidad de probabilidad (FDP) de n. Para el caso de un ruido gaussiano, la FDP que se representa por $p(n)$ se obtiene de:

$$p(n) = \frac{1}{\sigma_n \sqrt{2\pi}} e^{(-n^2/2\sigma_n^2)} \qquad (1.9)$$

donde σ_n es el valor rms de la señal de ruido. Esta FDP de la figura 1.15.(c), muestra la distribución relativa de los valores de n. La probabilidad de que una amplitud de n esté dentro de un rango (α , β) resulta del área bajo la FDP en el rango (α , β) esto es:

$$P(\alpha < n < \beta) = \frac{1}{\sigma_n \sqrt{2\pi}} \int_{\alpha}^{\beta} e^{-n^2/2\sigma_n^2} \, dn \qquad (1.10)$$

De la ecuación (1.10) se deduce que:

$$p(\varepsilon|0_L) = \frac{1}{\sigma_n \sqrt{2\pi}} \int_{A}^{\infty} e^{-n^2/2\sigma_n^2} \, dn$$

$$= \frac{1}{\sqrt{2\pi}} \int_{A/\sigma_n}^{\infty} e^{-x^2/2} \, dx \qquad (1.11)$$

Esta integral no puede obtenerse en una forma cerrada. Se ha calculado numéricamente y puede encontrarse en tablas matemáticas estándar. Esta integral, siendo una función de A/σ_n, se puede representar convenientemente por $Q(A/\sigma_n)$. Entonces,

$$P(\varepsilon|0_L) = Q(\frac{A}{\sigma_n}) \qquad (1.12)$$

Una muy buena aproximación de la función Q es:

$$Q(x) \approx \frac{1}{x\sqrt{2\pi}} (1 - \frac{0.7}{x^2}) e^{-x^2/2} \qquad x > 2 \qquad (1.13)$$

Se observa que ya que la FDP de n es simétrica con respecto al origen, la probabilidad de que $n > A$ es la misma que la probabilidad de que $n < -A$. En consecuencia, $P(\varepsilon \mid 1_L)$ es la misma que $P(\varepsilon \mid o_L)$.

Para tener una idea aproximada de los órdenes de magnitud, se considera la amplitud pico de los pulsos A como k veces el valor rms del ruido, esto es, $A = k\sigma_n$. En este caso:

$$P(\varepsilon \mid 0_L) = P(\varepsilon \mid 1_L) = Q(k) \qquad (1.14)$$

En la tabla 1.1 se muestra las probabilidades de error para varios valores de k.

k	1	2	3	4	5	6
$P(\epsilon \mid \mathbf{0}_L)$	0.1587	0.0227	0.0013	3.16×10^{-5}	2.87×10^{-7}	9.9×10^{-10}

Tabla 1.1. Probabilidad de error dado que se transmite un 0_L.

La probabilidad de error de 10^{-6} significa, en promedio, que sólo uno en un millón de pulsos se detectará en forma errónea. Así, cuando A es cinco veces la amplitud rms del ruido, la probabilidad de error es de 2.87×10^{-7}. Esto significa, en promedio, que sólo 1 en 3'484.320 pulsos se detectará en forma errónea.

CAPITULO 2

TEORIA DE SEÑALES EN EL DOMINIO DEL TIEMPO Y LA FRECUENCIA

2.1. SEÑALES EN EL DOMINIO DEI TIEMPO Y LA FRECUENCIA

2.1.1. Señales en el dominio del tiempo

Una señal es un suceso que sirve para iniciar una acción, es decir es una función univaluada del tiempo en donde a cada instante de tiempo (variable independiente) corresponde un valor único de la función (variable dependiente); este valor puede ser un número real en cuyo caso se tiene una señal con valor real, o puede ser complejo y se tendrá una señal con valor complejo. En cualquier caso la variable independiente tiene valor real.

Una señal eléctrica puede ser una onda de voltaje o de corriente que puede describirse matemáticamente, siendo de mucha importancia la variación en el tiempo de estas señales.

Las señales senoidales juegan un papel primordial en el análisis de los sistemas de comunicación. Tales señales g(t) pueden representarse por la ecuación:

$$g(t) = A \cdot Cos(\omega t + \theta) \qquad (2.1)$$

donde A es la amplitud, θ es la fase y ω es la rapidez de cambio de fase o frecuencia angular de la sinusoide en radianes por segundo.

El principio de los métodos de Fourier para el análisis de señales es descomponerlas todas en sumatorias de

componentes senoidales. Esto proporciona la descripción de una señal dada, en términos de frecuencias senoidales. Un importante objetivo es la descripción de cómo la energía y la potencia de la señal (y la respuesta) están distribuidas en términos de tales frecuencias. Cualquier descripción de una respuesta a una señal dada mostrará, por supuesto, las características del sistema.

2.1.1.1. Señales de energía y señales de potencia

Una señal de energía es una señal en forma de pulso que normalmente existe sólo durante un intervalo finito de tiempo o, si está presente por un lapso infinito tiene, al menos, la mayor parte de su energía concentrada en un intervalo finito de tiempo.

Para los sistemas eléctricos, una señal es un voltaje o una corriente. La potencia instantánea, para un voltaje e(t) disipada por una resistencia R es:

$$P = |e(t)|^2 / R \quad watts. \tag{2.2}$$

y para una corriente i(t):

$$P = |i(t)|^2 \cdot R \quad watts. \tag{2.3}$$

En cada caso, la potencia instantánea es proporcional a la magnitud de la señal, al cuadrado.

Para una resistencia de 1 ohm, estas ecuaciones toman la misma forma, por lo que es usual, en el análisis de señales, referirse a la potencia instantánea asociada con una señal dada g(t) como:

$$P = |g(t)|^2 \quad watts. \tag{2.4}$$

Aunque la ecuación anterior puede no parecer dimensionalmente correcta, la convención implica la multiplicación o división por una resistencia adecuada.

De acuerdo con esta convención, la energía disipada por la señal durante un intervalo de tiempo (t_1, t_2) es:

$$E = \int_{t_1}^{t_2} |g(t)|^2 \, dt \qquad joules. \tag{2.5}$$

Se define como una señal de energía aquella para la cual la ecuación anterior es finita, aún cuando el intervalo de tiempo sea infinito; esto es, cuando:

$$\int_{-\infty}^{\infty} |g(t)|^2 \, . \, dt \; < \; \infty \tag{2.6}$$

En la figura 2.1 se muestran varios ejemplos de señales de energía.

La potencia media disipada por la señal g(t) durante el intervalo de tiempo (t_1, t_2) es:

$$P = \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} |g(t)|^2 \, dt \tag{2.7}$$

Cuando el término de la derecha de la ecuación anterior se mantiene finito (no cero) en un intervalo de tiempo infinito[1], la señal g(t) tiene potencia media finita y se

---

[1] $0 < \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} |g(t)|^2 \, dt < \infty$

Stremler F.G., Sistemas de Comunicación Digital, Fondo E. Interamericano,p.10

llama señal de potencia.

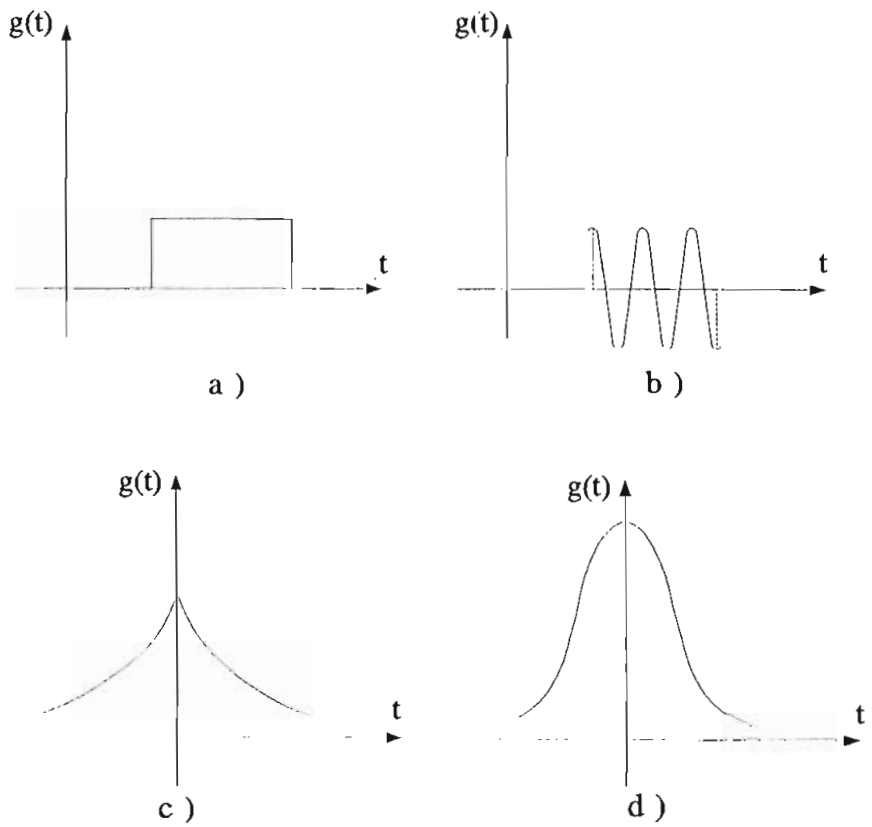


Figura 2.1. Señales en el tiempo
a) Pulso rectangular
b) Sinusoide pulsante
c) Pulso exponencial bilateral
d) Pulso gaussiano

2.1.1.2. Señales periódicas y no periódicas

Una señal periódica es la que se repite exactamente a sí misma después de un lapso fijo. Por tanto, la señal g(t) es periódica si existe un número T_o tal que:

$$g(t+T_o) = g(t) \qquad T_o \neq 0 \qquad (2.8)$$

El valor más pequeño de T_o que satisface la ecuación (2.8) se llama período. El período define la duración de un

ciclo completo de g(t). Una señal periódica es una señal de potencia, si su energía por ciclo es finita, y entonces la potencia media solo necesita calcularse en un ciclo completo. Como ejemplo de señal periódica tenemos la de la figura 2.2.

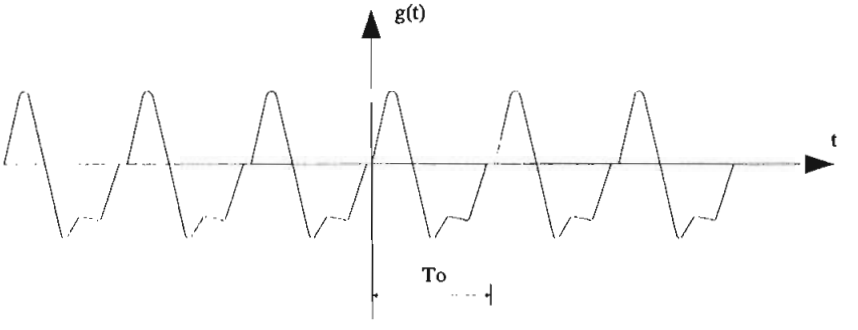


Figura 2.2. Señal periódica

Cualquier señal para la que no exista un valor de T_o que satisfaga la ecuación (2.8), se dice que es no periódica o aperiódica. Un caso intermedio entre las señales periódicas y no periódicas es el de la "señal casi periódica". Este tipo de señal está compuesto por la suma de dos o más señales periódicas con períodos inconmensurables[1]. Un ejemplo de tal señal es:

$$g(t) = Sen(t) + Sen(\sqrt{2}t) \qquad (2.9)$$

La función de la ecuación (2.9) es "casi periódica" dado que cada término de la derecha es periódico, aunque no hay un período T_o en el cual g(t) se repita exactamente a sí misma.

2.1.1.3. Señales aleatorias y determinísticas

Una señal aleatoria es sobre la que hay algún grado de incertidumbre antes que ocurra realmente. Esta señal puede considerarse perteneciente a un conjunto de señales, todas

[1] Dos períodos son conmesurativos si pueden ser expresados como el cociente de dos enteros. Stremler F.G., Sistemas de Comunicación, Fondo E. Interamericano, p.10

ellas diferentes. Si se elige (al azar) una de estas señales, podría suceder que estuviera bastante bien definida, como por ejemplo, una sinusoide de frecuencia fija pero cuya fase de inicio es incierta. Sin embargo, si se eligiera una segunda sinusoide de este conjunto, no podría asegurarse que tiene la misma fase de inicio. En otros casos, los valores futuros de la señal pueden no ser predecibles aún después de la observación de valores pasados. Un ejemplo de señal aleatoria es la salida de un receptor de radio cuando, al sintonizarlo, responde al ruido proveniente de alteraciones atmosféricas y de sus circuitos internos.

Una señal no aleatoria o determinística es aquella sobre cuyos valores no existe incertidumbre. En casi todos los casos, pueden escribirse una expresión matemática explícita para tales señales.

2.1.2. Señales en el dominio de la frecuencia

Cuando las señales están en el dominio del tiempo la corriente o voltaje varía con el tiempo, siendo este último la variable independiente. Las señales en el dominio de la frecuencia tienen como variable independiente la frecuencia.

La representación en el dominio frecuencial, permite describir una señal dada desde un punto de vista diferente del dominio temporal.

Básicamente, el trasladar del dominio temporal al frecuencial puede entenderse imaginando que una cierta señal g(t) está compuesta por la suma de ondas simples, senoidales, de amplitud y fase apropiadas. La representación de estas amplitudes y fases en función de la frecuencia es lo que se denomina espectro de la señal y se lo representa por G(w).

El análisis espectral está basado en la Serie y la Transformada de Fourier, las cuales son las herramientas matemáticas que permiten pasar del dominio temporal al frecuencial y viceversa. La expansión en series de Fourier

resulta aplicable a funciones periódicas mientras que la transformada nos permitirá el tratamiento de funciones aperiódicas.

2.2. SEÑALES PERIODICAS. SERIES DE FOURIER

Sea g(t) una función periódica del tiempo de período T_o, la frecuencia de repetición $\mathbf{f}_o$ es $1/T_o$. Una señal periódica se conserva sin cambio, al aplicarle un corrimiento positivo o negativo de cualquier entero múltiplo del período T_o.

Esto significa que una verdadera señal periódica debe comenzar en $t = -\infty$ y continuar indefinidamente hasta $t = +\infty$.

Considerando una señal g(t) formada por la suma de senoides de frecuencias 0, $\mathbf{f}_o$, $2\mathbf{f}_o, \ldots, k\mathbf{f}_o$.

$$g(t) = a_o + a_1 Cos(2\pi f_o t) + a_2 Cos2(2\pi f_o t) + \ldots + a_k Cos(2\pi k f_o t)$$

$$+ b_1 Sen(2\pi f_o t) + b_2 Sen2(2\pi f_o t) + \ldots + b_k Sen(2\pi k f_o t)$$

$$= a_o + \sum_{n=1}^{k} [a_n Cos2\pi n f_o (t+T_o) + b_n Sen2\pi n f_o (t+T_o)] \qquad (2.10.a)$$

Se puede ver fácilmente que esta señal es periódica con período $T_o = 1/\mathbf{f}_o$. Esto se deduce del hecho que:

$$g(t+T_o) = a_o + \sum_{n=1}^{k} [a_n Cos(2\pi n f_o (t+T_o)) + b_n Sen(2\pi n f_o (t+T_o))]$$

$$= a_o + \sum_{n=1}^{k} [a_n Cos(2\pi n f_o t + 2\pi n) + b_n Sen(2\pi n f_o t + 2\pi n)]$$

$$= a_o + \sum_{n=1}^{k} [a_n Cos(2\pi n f_o t + 2\pi n) + b_n Sen(2\pi n f_o t + 2\pi n)]$$

$$= g(t)$$

Por lo tanto, cualquier combinación de sinusoides de frecuencias 0, $\mathbf{f}_o$, $2\mathbf{f}_o$, ... , $k\mathbf{f}_o$ es una señal periódica con período T_o, sin importar los valores de las amplitudes a_n y b_n.

La condición recíproca también es verdadera, esto es cualquier señal periódica g(t) con período T_o puede expresarse como una suma de sinusoides de frecuencias $\mathbf{f}_o$ y de todos sus enteros múltiplos ($\mathbf{f}_o=1/T_o$), tal como se indica en la ecuación (2.10.a).

Se puede observar que la frecuencia 0 tiene una componente continua. La frecuencia $\mathbf{f}_o$ se conoce como la frecuencia fundamental, y la frecuencia $n\mathbf{f}_o$ es la n-ésima frecuencia armónica. Haciendo $\omega_0 = 2\pi f_0$, en la ecuación (2.10.a), se tiene:

$$g(t) = a_o + \sum_{n=1}^{k} [a_n Cos(n\omega_o t) + b_n Sen(n\omega_o t)] \qquad (2.10.b)$$

Para determinar los coeficientes a_n y b_n, se debe observar que:

$$\int_{t_o}^{(t_o+T_o)} Cos(k\omega_o t)\, Cos(n\omega_o t)\, dt = \begin{cases} 0 & \rightarrow k \neq n \\ T_o/2 & \rightarrow k = n \end{cases}$$

$$\int_{t_o}^{(t_o+T_o)} Sen(k\omega_o t)\, Sen(n\omega_o t)\, dt = \begin{cases} 0 & \rightarrow k \neq n \\ T_o/2 & \rightarrow k = n \end{cases}$$

$$\int_{t_o}^{(t_o+T_o)} Sen(k\omega_o t)\, Cos(n\omega_o t)\, dt = 0 \qquad (2.11)$$

Al integrar los miembros de la ecuación (2.10.b) en

un período t_o a (t_o+T_o), y considerando las ecuaciones 2.11 se obtiene a_0:

$$a_o = \frac{1}{T_o} \int_{t_o}^{t_o+T_o} g(t)\ dt \qquad (2.12.a)$$

Multiplicando los dos miembros de la ecuación (2.10.b) por $Cos\ (k\omega_0 t)$, e integrando en un periodo (t_o, t_o+T_o), cada término del segundo miembro, excepto aquel para $n = k$, se anulará debido a las ecuaciones (2.11), dando a_k ó a_n como:

$$a_n = \frac{2}{T_o} \int_{t_o}^{t_o+T_o} g(t)\, Cos(n\omega_o t)\, dt \qquad \omega_o=2\pi f_o=\frac{2\pi}{T_o} \qquad (2.12.b)$$

En forma similar, al multiplicar los dos miembros de la ecuación (2.10.b) por $Sen\ (n\omega_0 t)$ e integrar se obtiene:

$$b_n=\frac{2}{T_o} \int_{t_o}^{t_o+T_o} g(t)\, Sen(n\omega_o t) \qquad (2.12.c)$$

en donde t_o es arbitrario.

La serie trigonométrica de Fourier de la ecuación (2.10.a) se puede expresar de una manera más compacta y significativa como sigue:

$$g(t) = C_o + \sum_{n=1}^{\infty} C_n Cos(n\omega_o t+\theta_n) \qquad \omega_o=2\pi f_o \qquad (2.13)$$

Por la identidad trigonométrica[1], se deduce que:

$$C_o = a_o$$
$$C_n = \sqrt{a_n^2 + b_n^2} \qquad (2.14)$$

y que:

$$\theta_n = -\tan^{-1}\frac{b_n}{a_n}$$

De la Serie compacta de Fourier se deduce que g(t) consta de señales senoidales de frecuencias 0, $\mathbf{f}_o$, $2\mathbf{f}_o$, . . . , $n\mathbf{f}_o$.

La n-ésima armónica $C_n.Cos(n\omega_0 t + \theta_n)$ tiene amplitud C_n y fase θ_n. Se puede trazar el diagrama del espectro de magnitud (C_n contra ω) y del espectro de fase (θ_n contra ω) para una señal periódica dada.

Una señal en el tiempo muy importante para nuestro análisis posterior es un tren de pulsos rectangulares $k(t)$ tal como se muestra en la figura 2.3.(a). Aquí el período es T_o, $\mathbf{f}_o = 1/T_o$ y $\omega_0 = 2\pi f_0$.

$$k(t) = a_o + \sum_{n=1}^{\infty} [a_n Cos(n\omega_o t) + b_n Sen(n\omega_o t)]$$

$$a_o = \frac{1}{T_o}\int_{-\frac{T_o}{2}}^{\frac{T_o}{2}} k(t)\,dt = \frac{1}{T_o}\int_{-\frac{\tau}{2}}^{\frac{\tau}{2}} A.dt = \frac{A\tau}{T_o} \qquad (2.15.a)$$

---

[1] $a.Cos\ x + b.Sen\ x = \sqrt{a^2+b^2}\ Cos(x - \tan^{-1}\frac{b}{a})$

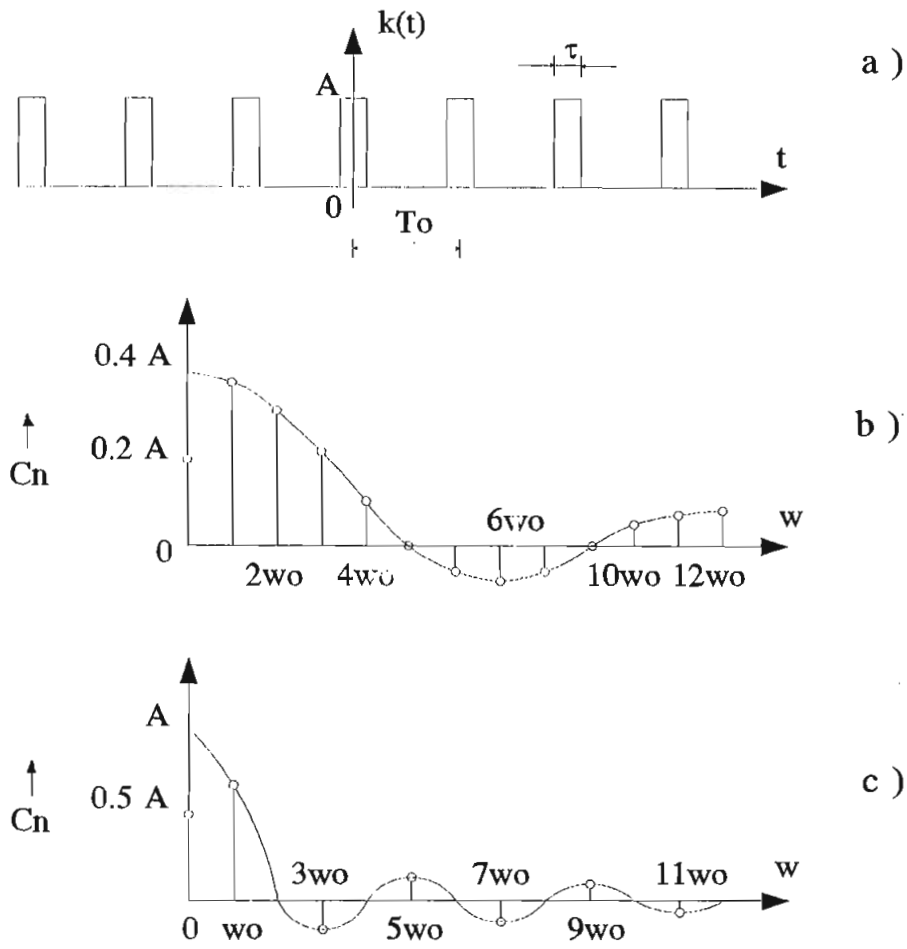


Figura 2.3. a) Tren de pulsos rectangulares
b) Espectro de magnitud para $\tau = T_o/5$
c) Espectro de magnitud para $\tau = T_o/2$

$$a_n = \frac{2}{T_o}\int_{-\frac{\tau}{2}}^{\frac{\tau}{2}} A.Cos(n\omega_o t)\,dt = \frac{2A}{\pi n}Sen(\frac{n\pi\tau}{T_o})$$

$$b_n = \frac{2}{T_o}\int_{-\frac{\tau}{2}}^{\frac{\tau}{2}} A.Sen(n\omega_o t)\,dt = 0 \qquad (2.15.b)$$

Por lo tanto,

$$k(t) = C_o + \sum_{n=1}^{\infty} C_n \, Cos(n\omega_o t + \theta_n) \qquad (2.16.a)$$

en donde por la ecuación (2.14) se tiene:

$$C_o = \frac{A\tau}{T_o} \qquad (2.16.b)$$

$$C_n = \frac{2A}{\pi n} \, Sen\left(\frac{n\pi\tau}{T_o}\right) \qquad (2.16.c)$$

$$\theta_n = 0 \qquad (2.16.d)$$

El espectro de magnitud[1] C_n es el de la figura 2.3.(b) para el caso de $\tau = T_o/5$.

Un tren de pulsos cuadrados es un caso especial del tren de pulsos rectangulares. Cuando $\tau = T_o/2$, se tiene:

$$C_o = \frac{A}{2}$$

$$C_n = \begin{cases} \dfrac{2A}{\pi} \dfrac{(-1)^{\frac{(n-1)}{2}}}{n} & n, impar \\ 0 & n, par \end{cases} \qquad (2.17.a)$$

La serie de Fourier de la ecuación (2.16.a) se convierte en:

---

[1] Estrictamente hablando, un espectro de magitud es siempre positivo. Cuando $\theta_n = 0$, como en la ecuación (2.16.a) los valores de C_n pueden ser tanto positivos como negativos. El signo negativo puede tener como explicación un corrimiento de fase de π ó $-\pi$.

$$k(t) = \frac{A}{2} + \frac{2A}{\pi} \sum_{n=1,3,5,..}^{\infty} \frac{-1^{\frac{(n-1)}{2}}}{n} Cos(n\omega_o t) \qquad (2.17.b)$$

$$= \frac{A}{2}\left[1 + \frac{4}{\pi}\left(Cos(\omega_o t) - \frac{1}{3}Cos(3\omega_o t) + \right.\right.$$

$$\left.\left. + \frac{1}{5}Cos(5\omega_o t) - \frac{1}{7}Ccs(7\omega_0 t) + ...\right]\right. \qquad (2.17.c)$$

La figura 2.3.(c), muestra el espectro de magnitud para este caso.

Otro caso especial de interés se presenta cuando $\tau \to 0$ y $A \to \infty$ tal que $A\tau = 1$. En este caso, cada pulso de la figura 2.3.(a) llega a ser un impulso de intensidad unitaria, y $k(t)$ es simplemente un tren uniforme de impulsos unitarios, como se muestra en la figura 2.4.(a).

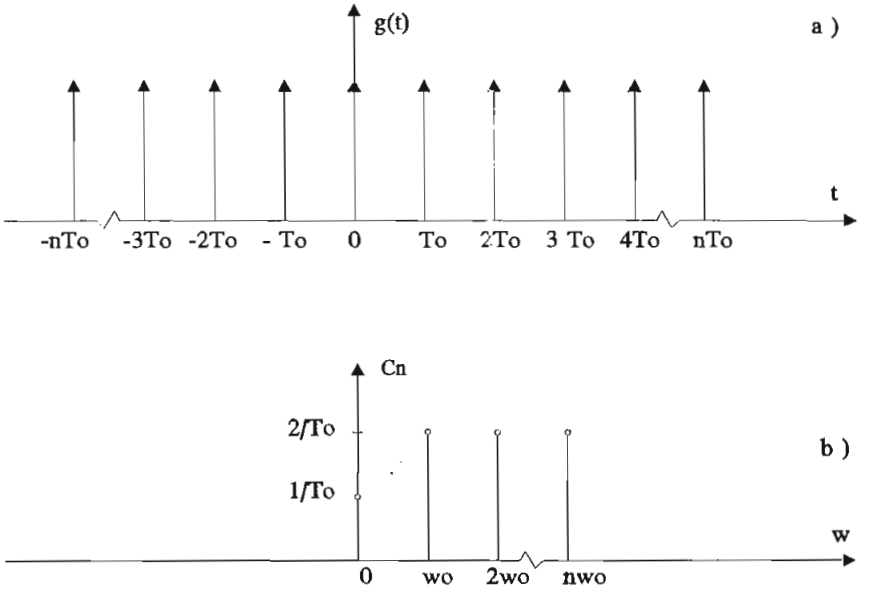


Figura 2.4. a) Tren de Impulsos
b) Espectro de frecuencia

Los resultados del tren de pulsos rectangulares de la figura 2.3 son válidos aquí con los límites apropiados, de esta

forma[1]:

$$\sum_{n=-\infty}^{\infty} \delta(t-nT_o) = C_o + \sum_{n=1}^{\infty} C_n Cos(n\omega_o t) \qquad (2.18)$$

Utilizando $A\tau = 1$ y $\tau \to 0$, de la ecuación (2.16.b) se tiene:

$$C_o = \frac{A\tau}{T_o} = \frac{1}{T_o} \qquad (2.19.a)$$

$$C_n = \lim_{\tau \to \infty} \frac{2A}{\pi n} Sen\left(\frac{n\pi\tau}{T_o}\right)$$

$$C_n = \lim_{\tau \to 0} \frac{2}{T_o} \frac{Sen\left(\frac{(n\pi\tau)}{T_o}\right)}{\left(\frac{n\pi\tau}{T_o}\right)}$$

$$C_n = \frac{2}{T_o} \qquad (2.19.b)$$

Este último resultado se deduce de la regla de L'Hospital donde se prueba que:

$$\lim_{X \to 0} \frac{Sen\,X}{X} = 1$$

En consecuencia:

$$\sum_{n=-\infty}^{\infty} \delta(t-nT_o) = \frac{1}{T_o}[1+2(Cos(\omega_o t)+Cos(2\omega_o t)+$$

$$+Cos(3\omega_o t)+Cos(4\omega_o t)+...Cos(n\omega_o t))] \qquad (2.19.c)$$

en donde $\omega_0 = 2\pi/T_0$.

---

[1] $\delta(t - nT_o) = \begin{cases} 1 & t = nT_o \\ 0 & t \neq nT_o \end{cases}$

Función Delta de Dirac, Papoulis Athanasios, Sistemas Digitales y Analógicos, Transformadas de Fourier, Marcombo, p.2.

El espectro de frecuencia aparece en la figura 2.4.(b). Obsérvese que las amplitudes de todas las componentes de frecuencia, excepto DC, son las mismas, o sea $2/T_o$. La amplitud de la componente de DC es $1/T_o$.

2.3. SERIE EXPONENCIAL DE FOURIER

Una señal senoidal de frecuencia angular $n\omega_0$ se puede expresar en términos de las señales exponenciales $e^{jn\omega_o t}$ y $e^{-jn\omega_o t}$. Una señal periódica g(t) con un período T_o se puede también representar mediante una serie que consta de componentes exponenciales, en la forma:

$$g(t) = G_o + G_1 e^{j\omega_o t} + G_2 e^{j2\omega_o t} + \ldots + G_n e^{jn\omega_o t} + \ldots + G_{-1} e^{-j\omega_o t} + G_{-2} e^{-j2\omega_o t} + \ldots + G_{-n} e^{-jn\omega_o t} + \ldots \quad (2.20.a)$$

$$g(t) = \sum_{n=-\infty}^{\infty} G_n e^{jn\omega_o t} \qquad \omega_o = 2\pi f_o = \frac{2\pi}{T_o} \quad (2.20.b)$$

Los coeficientes G_n de esta serie se pueden obtener multiplicando los dos miembros de la ecuación (2.20.b) por $e^{-jn\omega_o t}$ e integrando a través de un ciclo (t_o, t_o+T_o) para cualquier valor de t_o. Esto conduce a:

$$\int_{t_o}^{t_o+T_o} g(t)\, e^{-jk\omega_o t}\, dt = \sum_{n=-\infty}^{\infty} G_n \int_{t_o}^{t_o+T_o} e^{j(n-k)\omega_o t}\, dt \quad (2.21)$$

Para evaluar las integrales del segundo miembro de la ecuación (2.21), se utiliza el resultado:

$$\int_{t_o}^{t_o+T_o} e^{j(n-k)\omega_o t}dt = \begin{matrix} 0 & n\neq k \\ T_o & n=k \end{matrix} \qquad (2.22)$$

La sustitución de la ecuación (2.22) en la ecuación (2.21) produce:

$$G_k = \frac{1}{T_o}\int_{t_o}^{t_o+T_o} g(t)\, e^{-jk\omega_o t}dt \qquad (2.23)$$

La expresión de la ecuación (2.20.a) se conoce como **Representación en Serie Exponencial de Fourier** de una función periódica.

En conclusión, una forma de onda periódica g(t) con período T_o se puede expresar mediante una serie exponencial de Fourier:

$$g(t) = \sum_{n=-\infty}^{\infty} G_n e^{jn\omega_o t} \qquad \omega_o = 2\pi f_o = \frac{2\pi}{T_o} \qquad (2.24.a)$$

Los coeficientes G_n en general son complejos y se obtienen de:

$$G_n = \frac{1}{T_o}\int_{t_o}^{t_o+T_o} g(t)\, e^{-jn\omega_o t}dt \qquad (2.24.b)$$

La constante t_o es arbitraria y, si se escoge en forma apropiada, puede simplificar la integración de la ecuación (2.24.b).

Las Series Trigonométrica y Exponencial de Fourier no son dos series diferentes, sino que representan dos modos

distintos de escribir la misma serie. Los coeficientes de una serie pueden obtenerse a partir de los de la otra. De las ecuaciones (2.12.a), (2.12.b), (2.12.c), (2.24.a) y (2.24.b) se deduce que:

$$\left.\begin{aligned} a_o &= G_o \\ a_n &= G_n + G_{-n} \\ b_n &= j(G_n - G_{-n}) \end{aligned}\right\} n \neq 0$$

y

$$\left.\begin{aligned} G_n &= \frac{1}{2}(a_n - jb_n) \\ G_{-n} &= \frac{1}{2}(a_n + jb_n) \end{aligned}\right\} n \geq 1 \qquad (2.25)$$

Para una función g(t) real, a_n y b_n son números reales [ec. (2.12)], y los coeficientes G_n y G_{-n} son conjugados:

$$G_{-n} = G_n^* \qquad (2.26.a)$$

Por lo tanto, si:

$$\begin{aligned} G_n &= |G_n| \; e^{j\theta_n} \\ G_{-n} &= |G_n| \; e^{-j\theta_n} \end{aligned} \qquad (2.26.b)$$

$|G_n|$ es la magnitud y θ_n es el ángulo o la fase de G_n. En consecuencia, para una g(t) real, $|G_{-n}| = |G_n|$, y el espectro de magnitud $|G_n|$ contra ω es una función par de ω.

En forma similar, el espectro de fase θ_n versus ω es una función impar de ω ya que $\theta_{-n} = -\theta_n$.

De la misma manera que en el caso de la serie trigonométrica es importante encontrar el espectro de la serie exponencial de Fourier de un tren de pulsos rectangulares $k(t)$ como el de la figura 2.3.(a).

$$k(t) = \sum_{n=-\infty}^{\infty} K_n e^{jn\omega_o t} \qquad \omega_o = 2\pi f_o = \frac{2\pi}{T_o} \qquad (2.27.a)$$

en donde:

$$K_n = \frac{1}{T_o} \int_{-\frac{\tau}{2}}^{\frac{\tau}{2}} A e^{-jn\omega_o t} dt$$

$$= \frac{A}{\pi n} Sen\left(\frac{n\omega_o \tau}{2}\right)$$

$$= \frac{A}{\pi n} Sen\left(\frac{n\pi\tau}{T_o}\right) \qquad (2.27.b)$$

El espectro de magnitud para el caso $\tau = T_o/5$ aparece en la figura 2.5.(a). Para el caso de un tren de pulsos cuadrados, $\tau = T_o/2$, $K_n = (A/\pi n)\, Sen(n\pi/2)$, y el espectro K_n versus ω se encuentra en la figura 2.5.(b). Para el caso donde $\tau \to 0$ y $A\tau = 1$ (tren de impulso unitario de la figura 2.4.(a)), por la ecuación (2.27.b) se tiene:

$$K_n = Lim_{\tau \to 0} \frac{1}{T_o} \left(\frac{Sen\left(\frac{n\pi\tau}{T}\right)}{\frac{(n\pi\tau)}{T}} \right) = \frac{1}{T_o} \qquad (2.28.a)$$

$$\sum_{n=-\infty}^{\infty} \delta(t - nT_o) = \frac{1}{T_o} \sum_{n=-\infty}^{\infty} e^{jn\omega_o t} \qquad \omega_o = \frac{2\pi}{T_o} \qquad (2.28.b)$$

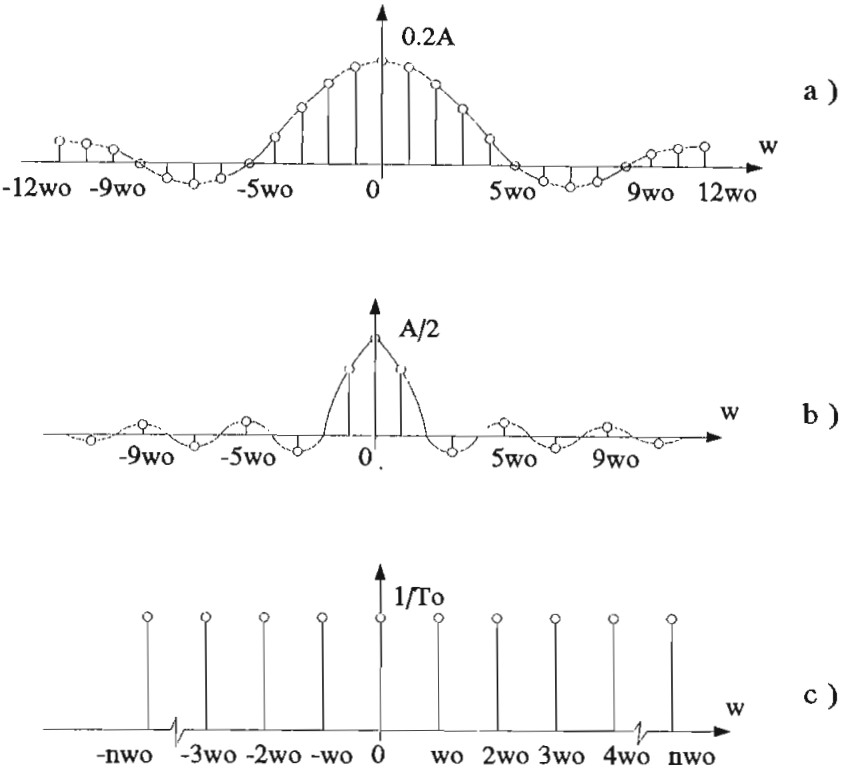


Figura 2.5. Espectro de pulsos rectangulares
a) Espectro de magnitud para $\tau = T_o/5$
b) Espectro de magnitud para $\tau = T_o/2$
c) Espectro de magnitud para $\tau = 0$

El espectro tiene componentes de frecuencias $n\omega_0$, n variando desde $-\infty$ hasta $+\infty$, incluyendo a 0, todas ellas con una intensidad igual de $1/T_o$, como se muestra en la figura 2.5.(c).

2.4. DENSIDAD ESPECTRAL DE POTENCIA

Si una señal g(t) existe en todo el intervalo $(-\infty , +\infty)$, se define la potencia P_g de una señal real g(t), como la potencia media que se disipa en una resistencia de 1 ohm cuando se aplica un voltaje g(t) a través de él [o se hace pasar una corriente g(t) a través de él]. De este modo:

$$P_g = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} g^2(t)\, dt \qquad (2.29.a)$$

La potencia P_g definida anteriormente es simplemente el valor cuadrático medio, o el promedio en el tiempo, del cuadrado de la señal. Los promedios en el tiempo se representan mediante una barra en la parte superior. Así que,

$$P_g = \overline{g^2 t} = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} g^2(t)\, dt \qquad (2.29.b)$$

Dicho de otra manera, la potencia promedio de una señal es su valor cuadrático medio. No todas las señales tienen energía finita, algunas tienen energía infinita aunque el promedio en el tiempo puede ser finito. Las señales para las cuales la energía es finita se conocen como señales de energía, y aquellas para las que potencia P_g es diferente de cero y finita se conocen como señales de potencia.

Para encontrar la expresión de la potencia P_g, en el dominio de la frecuencia, obsérvese que las señales de potencia pueden tener energía infinita y, por lo tanto, pueden no tener transformadas de Fourier.

Por consiguiente, en este caso se considera la señal truncada (figura 2.6), la cual se define como:

$$g_T(t) = \begin{cases} g(t) & |t| < \frac{T}{2} \\ 0 & |t| > \frac{T}{2} \end{cases}$$

Mientras T sea finita, $g_T(t)$ tendrá energía finita,

y será transformable en el sentido de Fourier.

Sea:

$$g_T(t) \leftrightarrow G_T(\omega)$$

La energía E_T de g_T (t) se obtiene de:

$$E_T = \int_{-\infty}^{\infty} g_T^2(t)\ dt = \frac{1}{2\pi}\int_{-\infty}^{\infty} |G_T(\omega)|^2\ d\omega$$

Pero:

$$\int_{-\infty}^{\infty} g_T^2(t)\ dt = \int_{-T/2}^{T/2} g^2(t)\ dt$$

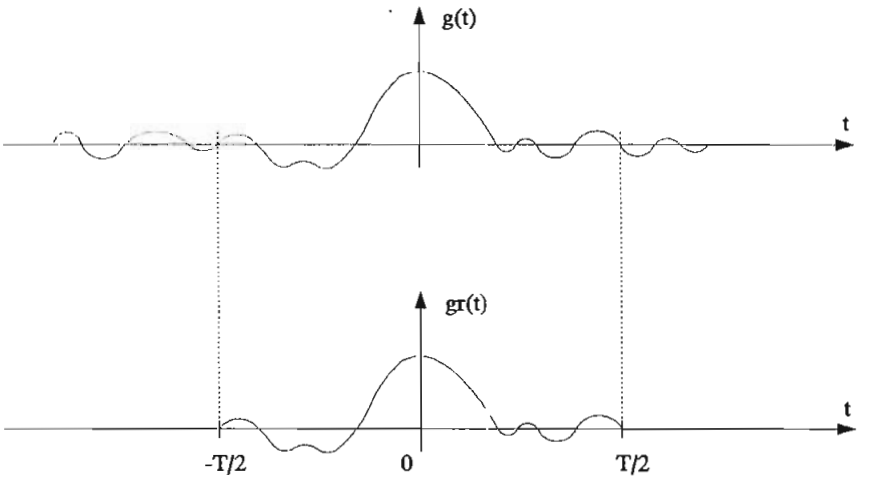


Figura 2.6. Señal de energía finita

En consecuencia, la potencia P_g resulta de [ec.(2.29.b)]

$$P_g = \lim_{T\to\infty}\frac{E_T}{T} = \lim_{T\to\infty}\frac{1}{T}\left[\frac{1}{2\pi}\int_{-\infty}^{\infty} |G_T(\omega)|^2\ d\omega\right] \qquad (2.30)$$

A medida que aumenta T, la energía E_T de $g_T(t)$ también aumenta. De esta manera, $|G_T(\omega)|^2$ aumenta con T, y cuando $T \to \infty$, $|G_T(\omega)|^2$ también se aproxima a ∞.

Sin embargo, $|G_T(\omega)|^2$ debe aproximarse a ∞ en la misma razón que T, ya que para una señal de potencia, la integral del segundo miembro de la ecuación (2.30) debe converger. Esta convergencia permite intercambiar el orden del proceso de límite y de integración en la ecuación (2.30), y así se tiene:

$$P_g = \frac{1}{2\pi} \int_{-\infty}^{\infty} \lim_{T\to\infty} \frac{|G_T(\omega)|^2}{T} \, d\omega$$

Se define $S_g(\omega)$ como la densidad espectral de potencia (DEP):

$$S_g(\omega) = \lim_{T\to\infty} \frac{|G_T(\omega)|^2}{T} \qquad (2.31)$$

Entonces:

$$P_g = \frac{1}{2\pi} \int_{-\infty}^{\infty} S_g(\omega) \, d\omega \qquad (2.32.a)$$

$$= \frac{1}{\pi} \int_{0}^{\infty} S_g(\omega) \, d\omega \qquad (2.32.b)$$

La densidad espectral de potencia $S_g(\omega)$ es una función positiva, real y par de ω. Las unidades de $S_g(\omega)$ son watts por hertzios, y las de P_g son watts.

La densidad espectral de potencia sólo retiene información sobre la magnitud, omitiendo la referente a la fase; de ahí que todas las señales con la misma densidad espectral de magnitud tienen idénticos espectros de potencia, es decir hay una densidad espectral de potencia para una señal dada pero puede haber muchas señales con la misma densidad espectral de potencia.

2.4.1. Correlación en el tiempo de las señales de potencia

Para dos señales reales de potencia $g_1(t)$ y $g_2(t)$, se define la función de correlación cruzada en el tiempo $\mathbb{R}g_1g_2(\tau)$ como:

$$\mathbb{R}_{g_1g_2}(\tau) = \overline{g_1(t)g_2(t+\tau)} = \lim_{T\to\infty}\frac{1}{T}\int_{-T/2}^{T/2} g(t)\,g_2(t+\tau)\,dt \qquad (2.33.a)$$

La función de autocorrelación en el tiempo $\mathbb{R}_g(\tau)$ de una señal real g(t) se define como:

$$\mathbb{R}_g(\tau) = \overline{g(t)g(t+\tau)} = \lim_{T\to\infty}\frac{1}{T}\int_{-T/2}^{T/2} g(t)\,g(t+\tau)\,dt \qquad (2.33.b)$$

Al cambiar de variable x = t + τ:

$$\mathbb{R}_x(\tau) = \lim_{T\to\infty}\frac{1}{T}\int_{(-T/2)-\tau}^{(T/2)+\tau} g(x)\,g(x-\tau)\;dx = \overline{g(t)g(t-\tau)} \qquad (2.33.c)$$

Lo que permite escribir la ecuación (2.33.b) de la forma:

$$\mathbb{R}_g(\tau) = \overline{g(t)g(t-\tau)} = \lim_{T\to\infty}\frac{1}{T}\int_{-T/2}^{T/2} g(t)\,g(t-\tau)\;dt \qquad (2.33.d)$$

De las ecuaciones (2.33.b y 2.33.d) se deduce que $\mathbb{R}_g(\tau)$ es una función par de τ.

$$\mathbb{R}_g(-\tau) = \mathbb{R}_g(\tau) \qquad (2.33.e)$$

Para las señales de potencia la DEP $S_g(\omega)$ es la transformada de Fourier de la función de autocorrelación en el

tiempo $\mathbb{R}_g(\tau)$ de la ecuación (2.33.d).

Esto puede probarse considerando la señal truncada anterior $g_T(t)$. De la ecuación (2.33.b),

$$\mathbb{R}_g(\tau) = \lim_{T\to\infty} \int_{-\infty}^{\infty} g_T(t)\, g_T(t+\tau)\, dt$$

En consecuencia como se demostrará en el análisis de la densidad espectral de energía, según las ecuaciones (2.40.a) y (2.50.b), se tendrá que:

$$\mathcal{F}[\mathbb{R}_g(\tau)] = \lim_{T\to\infty} \frac{|G_T(\omega)|^2}{T} = S_g(\omega) \qquad (2.34)$$

2.5. SEÑALES NO PERIODICAS.TRANSFORMADA DE FOURIER

La representación de las señales no periódicas mediante señales exponenciales se puede llevar a cabo mediante un simple proceso de límite, haciendo que las señales no periódicas, en general, puedan expresarse como una suma continua de señales exponenciales.

De la señal no periódica g(t) que aparece en la figura 2.7.(a) se construye una nueva señal periódica $g_p(t)$ que constará de la señal g(t) que se repite cada T_o segundos, como se muestra en la figura 2.7.(b). El período T_o se hace suficientemente largo para que no exista traslape entre los pulsos repetitivos. Esta nueva señal $g_p(t)$ es una señal periódica y, en consecuencia, se puede representar mediante una serie exponencial de Fourier. En el límite, si se hace que T_o llegue a ser infinito, los pulsos de la señal periódica se repetirán después de un intervalo infinito, y

$$\lim_{(T\to\infty)} g_p(t) = g(t) \qquad (2.35)$$

Por lo tanto, la serie de Fourier que representa a $g_p(t)$ también representará a g(t), en el límite.

La serie exponencial de Fourier para $g_p(t)$ es:

$$g_p(t) = \sum_{n=-\infty}^{\infty} G_n e^{jn\omega_o t} \qquad \omega_o = \frac{2\pi}{T_o} \qquad (2.36)$$

. Y

$$G_n = \frac{1}{T_o} \int_{-\frac{T}{2}}^{\frac{T}{2}} g_p(t)\, e^{-jn\omega_o t}\, dt \qquad (2.37)$$

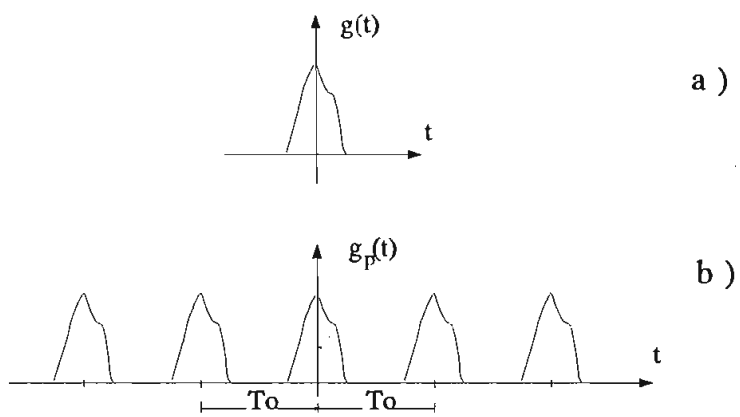


Figura 2.7. a) Representación de una señal g(t) aperiódica
b) Representación de g(t) como señal periódica

A medida que T_o llega a ser más grande, ω_0 (la frecuencia fundamental) se hace más pequeña y el espectro se hará más denso. Como se ve en la ecuación (2.37), las

amplitudes de las componentes individuales llegan a ser también más pequeñas; sin embargo, la forma del espectro de frecuencia no se altera. En el límite, cuando $T_o \to \infty$, la magnitud de cada una de las componentes se vuelve infinitamente pequeña, pero existe ahora un número infinito de componentes de frecuencia $(\omega_0 \to 0)$.

El espectro existe para todos los valores de ω y no es ya más una función discreta sino continua de ω. Para ilustrar este punto, se hace un ligero cambio en la notación. Ya que en el límite, cuando $T_o \to \infty$, $\omega_0 \to 0$, la cantidad ω_0 es infinitesimal y se puede representar por $\Delta\omega$.

$$T_o = \frac{2\pi}{\omega_o} = \frac{2\pi}{\Delta\omega}$$

De la ecuación (2.37),

$$T_o G_n = \int_{-\frac{T}{2}}^{\frac{T}{2}} g_p(t)\, e^{-jn\Delta\omega t}\, dt \qquad (2.38.a)$$

La cantidad $T_o\, G_n$ es una función de $n\Delta\omega$. Se introduce

$$T_o G_n = G(n\Delta\omega) \qquad (2.38.b)$$

De las ecuaciones (2.36 y 2.37),

$$g_p(t) = \sum_{n=-\infty}^{\infty} \frac{G(n\Delta\omega)}{T_o}\, e^{jn\Delta\omega t} \qquad (2.39.a)$$

$$= \sum_{n=-\infty}^{\infty} \left[\frac{G(n\Delta\omega)\,\Delta\omega}{2\pi}\right] e^{jn\Delta\omega t}$$

Aquí $g_p(t)$ se expresa como una suma de exponenciales de frecuencias 0, $\pm\Delta\omega$, $\pm 2\Delta\omega$, $\pm 3\Delta\omega$, $\pm 4\Delta\omega$, ...

La intensidad de la componente de frecuencia $n\Delta\omega$ es $G(n\Delta\omega)\Delta\omega/2\pi$, que es infinitesimal y se aproxima a cero en el límite cuando $T_o \to \infty$. Se presenta una situación extraña al tener una componente de cada frecuencia posible ($\Delta\omega \to 0$), pero la intensidad de cada componente se va acercando a cero.

Se debe observar que aunque las amplitudes de todas las componentes de frecuencia se aproximan a cero, sus intensidades relativas se obtienen de $G(n\Delta\omega)$.

Debido a que en el límite cuando $T_o \to \infty$, $g_p(t) \to g(t)$,

$$g(t) = \lim_{(T_o \to \infty)} g_p(t) = \lim_{(\Delta\omega \to 0)} \frac{1}{2\pi} \sum_{n=-\infty}^{\infty} G(n\Delta\omega)\, e^{jn\Delta\omega t} \Delta\omega \qquad (2.39.b)$$

El segundo miembro, por definición, es la integral:

$$g(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} G(\omega)\, e^{j\omega t} d\omega$$

También, de las ecuaciones (2.38.a y 2.38.b),

$$\begin{aligned} G(n\Delta\omega) &= \lim_{(T \to \infty)} \int_{-\frac{T_o}{2}}^{\frac{T_o}{2}} g_p(t)\, e^{-jn\Delta\omega t} dt \\ &= \int_{-\infty}^{\infty} g(t)\, e^{-jn\Delta\omega t} dt \end{aligned} \qquad (2.40)$$

Por lo tanto,

$$G(\omega) = \int_{-\infty}^{\infty} g(t)\, e^{-j\omega t}\, dt \qquad (2.41)$$

Se demuestra que :

$$g(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} G(\omega)\, e^{j\omega t}\, d\omega \qquad (2.42)$$

en donde

$$G(\omega) = \int_{-\infty}^{\infty} g(t)\, e^{-j\omega t}\, dt \qquad (2.43)$$

En conclusión se ha llegado a la representación de una señal no periódica g(t) en términos de señales exponenciales. La ecuación (2.42) representa a g(t) como una suma continua de funciones exponenciales con frecuencias que están en el intervalo $(-\infty < \omega < +\infty)$. La amplitud de la componente a cualquier frecuencia ω es proporcional a $G(\omega)$. Por lo tanto, $G(\omega)$ representa el espectro de frecuencia de g(t). El espectro de frecuencia es continuo y existe para todos los valores de ω.

En el caso de transmisión digital es necesario encontrar la transformada de Fourier de la función pulso $\prod(t)$ definida por: (figura 2.8.(a))

$$\prod(t) = \begin{cases} 1 & |t| < \frac{1}{2} \\ 0 & |t| > \frac{1}{2} \end{cases} \qquad (2.44)$$

Ya que una función g(t/**a**) es la misma que g(t) modificada por un factor **a** en la escala del tiempo para **a > 1**, $\prod(t/\tau)$ es una función pulso de ancho τ con centro en el origen (figura 2.8.(b)).

$$\mathcal{F}[\prod(\frac{t}{\tau})] = \int_{-\frac{\tau}{2}}^{\frac{\tau}{2}} e^{-j\omega t}dt$$

$$= \frac{1}{j\omega}[e^{j\omega\frac{\tau}{2}} - e^{-j\omega\frac{\tau}{2}}] \qquad (2.45)$$

$$= \tau\frac{Sen(\omega\frac{\tau}{2})}{\omega\frac{\tau}{2}}$$

El espectro de frecuencia de $\prod(t/\tau)$ aparece en la figura 2.8.(c). La función con la forma $(Sen\ X)/X$ de la ecuación (2.45) juega un papel importante en la teoría de la comunicación.

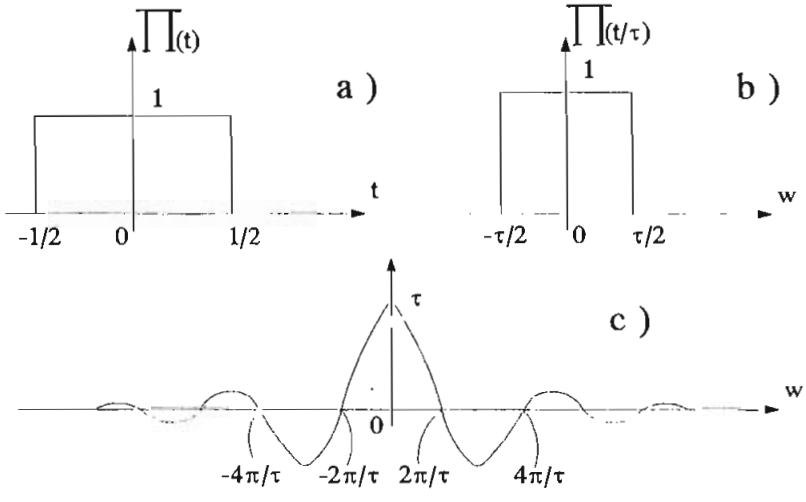


Figura 2.8. a) Función pulso $\Pi(t)$
b) Función pulso $\Pi(t/\tau)$
c) Espectro de $\Pi(t/\tau)$

Para lograr uniformidad, se utiliza una notación especial para esta función. Se define como:

$$Sinc(X) = \frac{Sen(\pi X)}{\pi X} \qquad (2.46)$$

y de la ecuación (2.45),

$$\prod(\frac{t}{\tau}) \leftrightarrow \tau . Sinc(\frac{\omega\tau}{2\pi}) \qquad (2.47)$$

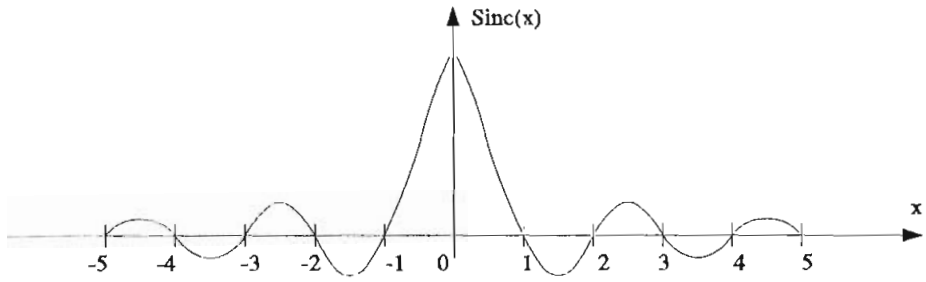


Figura 2.9. Función *Sinc* (*X*)

La función *Sinc* (*X*) aparece en la figura 2.9. Utilizando la regla de L'Hospital, se encuentra que *Sinc*(0) = 1. También, *Sinc*(*X*) = 0 para todos los valores enteros de X.

Entonces, Sinc($\omega\tau/2\pi$) = 0 para todos los valores enteros de $\omega\tau/2\pi$, o cuando $\omega = \pm(2\pi/\tau), \pm(4\pi/\tau), \pm(6\pi/\tau), \ldots$

2.5.1. Función de correlación en tiempo y energía

La función de correlación en el tiempo $\Psi g_1 g_2(\tau)$ de dos señales reales $g_1(t)$ y $g_2(t)$ se define como:

$$\psi_{g_1 g_2}(\tau) = \int_{-\infty}^{\infty} g_1(t)\, g_2(t+\tau)\ dt \qquad (2.48.a)$$

Si se hace t = -x, se tiene:

$$\psi_{g_1 g_2}(\tau) = \int_{-\infty}^{\infty} g_1(-x)\, g_2(\tau - x)\ dx \qquad (2.48.b)$$

$$\psi_{g1g2}(\tau) = g_1(-\tau) * g_2(\tau) \qquad (2.48.c)$$

La función $\Psi g_1 g_2(\tau)$ se conoce también como función de correlación cruzada en el tiempo de $g_1(t)$ y $g_2(t)$. En contraste, se define $\Psi g(\tau)$, la función de autocorrelación en el tiempo de una g(t) real, como:

$$\psi_g(\tau) = \int_{-\infty}^{\infty} g(t)\, g(t+\tau)\, dt \qquad (2.49.a)$$

$$\psi_g(\tau) = \int_{-\infty}^{\infty} g(-x)\ g(\tau - x)\ dx \qquad (2.49.b)$$

$$\psi_g(\tau) = g(-\tau) * g(\tau) \qquad (2.49.c)$$

La aplicación de la propiedad de la convolución en el tiempo a las ecuaciones (2.48.c) y (2.49.c) da por resultado:

$$\psi_{g_1 g_2}(\tau) \leftrightarrow G_1(-\omega)\ G_2(\omega) \qquad (2.50.a)$$

$$\psi_g(\tau) \leftrightarrow G(-\omega)\ G(\omega) = |G(\omega)|^2 \qquad (2.50.b)$$

De las ecuaciones (2.48.a) y (2.50.a), se tiene:

$$\int_{-\infty}^{\infty} g_1(t)\, g_2(t+\tau)\, dt = \frac{1}{2\pi}\int_{-\infty}^{\infty} G_1(-\omega)\, G_2(\omega)\, e^{j\omega\tau}\, d\omega$$

Se hace $\tau = 0$ y se tiene:

$$\int_{-\infty}^{\infty} g_1(t)\, g_2(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} G_1(-\omega)\, G_2(\omega)\, d\omega \qquad (2.51.a)$$

En forma similar, si se considera a $\Psi g_1 g_2(\tau)$ se puede demostrar que:

$$\int_{-\infty}^{\infty} g_1(t)\, g_2(t)\, dt = \frac{1}{2\pi}\int_{-\infty}^{\infty} G_1(\omega)\, G_2(-\omega)\, d\omega \qquad (2.51.b)$$

Si se hace $g_1(t) = g_2(t) = g(t)$, se tiene:

$$\int_{-\infty}^{\infty} g^2(t)\, dt = \frac{1}{2\pi}\int_{-\infty}^{\infty} |G(\omega)|^2\, d\omega \qquad (2.51.c)$$

La integral en el primer miembro de la ecuación (2.51.c) es E_g, la energía de g(t). El segundo miembro expresa la energía de la señal en el dominio de la frecuencia. Esta relación, ecuación (2.51 c), se conoce como **Teorema de Parseval**.

2.6. ENERGIA Y DENSIDAD ESPECTRAL DE ENERGIA

La energía de una señal se define como:

$$E_g = \int_{-\infty}^{\infty} g^2(t)\, dt \qquad (2.52)$$

El concepto de energía de una señal es válido sólo si el área bajo $g^2(t)$ es finita. Las señales que cumplen esta

condición se llaman **señales de energía** y satisfacen también la condición de transformabilidad en el sentido de Fourier. La energía puede también expresarse en términos del espectro $G(\omega)$ en la forma [ver la ec. (2.51.c)]:

$$E_g = \int_{-\infty}^{\infty} g^2(t)\,dt = \frac{1}{2\pi}\int_{-\infty}^{\infty} |G(\omega)|^2\,d\omega \qquad (2.53)$$

Para una resistencia de 1 ohm, la cantidad $|G(\omega)|^2$ está en términos de energía por unidad de frecuencia, por esta razón se denomina **Densidad Espectral de Energía DEE de g(t)**. Es decir que $|G(\omega)|^2$ es una función que describe la cantidad relativa de energía en función de la frecuencia, el área bajo dicha función es la energía de la señal.

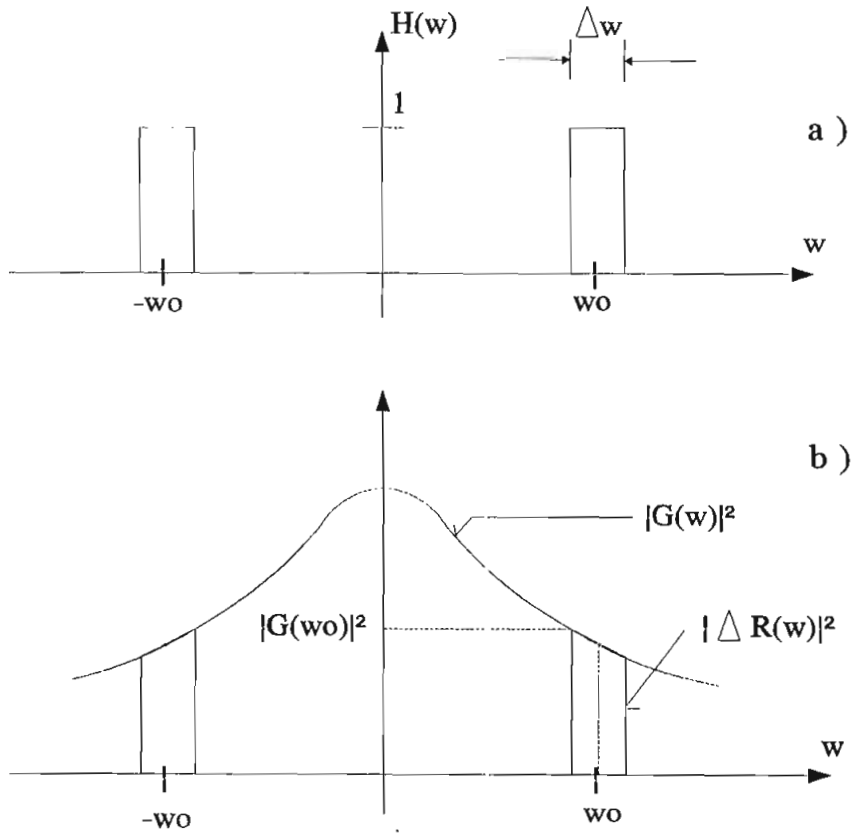


Figura 2.10. a) Función de transferencia de un filtro pasabajos
b) Densidad espectral de energía

Si la señal g(t) se aplica a la entrada de un filtro ideal pasabanda cuya función de transferencia $H(\omega)$ aparece en la figura 2.10.(a), este filtro suprime todas las frecuencias, excepto una banda angosta $\Delta\omega$ ($\Delta\omega \to 0$) con centro en la frecuencia ω_0 (figura 2.10.(b)). Si $\Delta R(\omega)$ es la transformada de Fourier de la respuesta $\Delta R(t)$ de este filtro, entonces:

$$\Delta R(\omega) = G(\omega) . H(\omega) \qquad (2.54)$$

y la energía $\Delta E_R(t)$ de la señal de salida $\Delta R(t)$ se obtiene de [ec. (2.51.c)],

$$\Delta E_R = \frac{1}{2\pi} \int_{-\infty}^{\infty} |G(\omega) . H(\omega)|^2 \, d\omega \qquad (2.55)$$

Ya que $H(\omega) = 0$ en todas partes, excepto a través de una banda angosta $\Delta\omega$ en donde es igual a la unidad, se tiene (para $\Delta\omega \to 0$).

$$\Delta E_R = 2 \frac{1}{2\pi} |G(\omega_o)|^2 \, d\omega$$

$$= 2 |G(\omega_o)|^2 \, \Delta f \qquad (2.56)$$

Como se observa en la figura 2.10.(b), sólo las componentes de frecuencia de g(t) que están en la banda angosta $\Delta\omega$ se transmiten intactas a través del filtro. Las componentes restantes de frecuencia se suprimen por completo.

Por lo tanto, $2 |G(\omega_0)|^2 \, \Delta f$ representa la contribución a la energía de g(t) de las componentes de frecuencia de g(t) que pertenecen a la banda angosta Δf con centro en ω_0 . En consecuencia, $|G(\omega)|^2$ es la energía por unidad de ancho de banda (en Hz) con que contribuyen las componentes de frecuencia con

centro en ω.

Se tiene contribución de energía tanto de las componentes de frecuencia negativa como de las positivas. Más aún, la contribución que hacen las componentes de frecuencia positiva y negativa es igual ya que $|G(\omega)|^2 = |G(-\omega)|^2$. Por lo tanto, se puede interpretar que la cantidad $|G(\omega)|^2$ [la mitad de la energía $2|G(\omega)|^2$] fue contribuida por las componentes de frecuencia positiva y la restante $|G(\omega)|^2$ fue contribuida por las componentes de frecuencia negativa.

La función $|G(\omega)|^2$, o DEE, representa la energía por unidad de ancho de banda (ya sea positiva o negativa). En conclusión la DEE se simboliza por $\Psi_g(\omega)$ y se define como:

$$\psi_g(\omega) = |G(\omega)|^2 \qquad (2.57)$$

CAPITULO 3

CODIGOS DE LINEA. CARACTERIZACION Y TEORIA

3.1. TRANSMISION EN BANDA BASE

Los sistemas digitales están constituidos de circuitos lógicos, los cuales funcionan con dos niveles de tensión, es decir con señales binarias. Para la representación de la información binaria se utilizan señales digitales, las cuales en un sistema de transmisión real tienen una banda de paso limitada estableciéndose así una anchura de pulso mínima. La transmisión en banda base es la técnica por la cual se transmite información en forma digital sin modular.

Las señales binarias no pueden ser normalmente transmitidas directamente sobre largas distancias, pues se hace necesario transmitir además de la señal de datos una señal de reloj que indique donde comienza y termina cada bit.

Para ello, se realiza la codificación de la información en códigos de línea, para así adaptar el espectro de la señal digital a las características del canal, con el objeto de tener filtros de transmisión y recepción físicamente realizables.

Si un canal tiene una pobre respuesta de amplitud con una alta atenuación en el rango de baja frecuencia, con los filtros de transmisión y recepción y con la señalización adecuada se podría tener alta ganancia de potencia en los rangos de baja frecuencia. Este problema podría ser evitado si el espectro de la señal transmitida es alterado de manera que tenga un pequeño contenido de baja potencia en las bajas frecuencias.

Existen diversos medios de transmisión digital a distancia que permiten la transmisión en banda base, siendo los principales:

- Par balanceado (conductores de cobre)
- Pares simétricos especiales para PCM
- Cables coaxiales
- Fibra óptica

Generalmente el medio que más se utiliza es el par balanceado. Los sistemas digitales que están instalados en las zonas urbanas e interurbanas utilizan el par balanceado, siendo necesario en el trayecto de la transmisión introducir repetidores regenerativos de la señal, los cuales se diseñan usando transformadores de las centrales. Por esta razón los códigos deberán franquear los transformadores y deberán carecer de energía en las frecuencias bajas.

El espectro de la señal transmitida en un sistema de transmisión digital en banda base depende de la forma de onda del pulso y de las propiedades estadísticas de la secuencia de dígitos transmitidos. La forma de onda del pulso en un sistema en banda base es especificada por los requerimientos de ISI. Mientras podría ser posible conformar el espectro de la señal transmitida cambiando la forma del pulso, tales cambios podrían conducir a un aumento de la ISI. Una manera más sencilla es, conformar el espectro de la señal transmitida por medio de la alteración de las propiedades estadísticas de la secuencia de símbolos, esto es por medio de la codificación.

3.2. RAZONES Y VENTAJAS DE LA CODIFICACION DE LINEA

El proceso de codificación que se introduce entre la fuente y el trayecto de transmisión, se puede considerar como una operación de adaptación de la señal. Estos códigos deben reunir ciertas características que permitan la transmisión en forma digital, sin modular, a través de un medio de transmisión físico; entre éstas, se pueden mencionar las siguientes:

a) Adaptación de señal de datos y señal de reloj en un mismo par conductor.

b) Transparencia, es decir que cualquier secuencia de bits puede ser codificada, transmitida y recibida sin importar la secuencia estadística de pulsos de la fuente.

c) Decodificación única para que cada símbolo de entrada sea decodificado sin ambigüedad.

d) Eficiencia, los códigos de entrada y salida no necesitan ser de la misma longitud; cualquier adición de bits representa una pérdida de la eficiencia.

e) Los códigos de línea deben tener un espectro de energía favorable y una adecuada información de temporización para efectos de recuperar la señal de reloj. El espectro adecuado es necesario para adaptar la señal a las características del trayecto de transmisión y minimizar la interferencia intersímbolo.

f) La señal codificada debe tener una componente DC no significativa.

Conceptualmente la manera más sencilla de obtener un código de línea es usar un nivel de señal diferente para codificar cada símbolo transmitido, en un sistema la forma más común es un código cerrado/abierto que usa 3 voltios para 1_L y 0 voltios para 0_L.

Sobre un enlace de transmisión, es más eficiente en términos de potencia media codificar los datos binarios con una diferencia equivalente en niveles, pero simétricamente balanceados alrededor de 0 voltios, por ejemplo 1.5 voltios para un 1_L y -1.5 voltios para un 0_L.

3.3. CLASES DE CODIGOS DE LINEA

Existen gran cantidad de códigos de línea diferentes, sin embargo sólo algunos han encontrado aplicación práctica teniéndose los siguientes tipos:

- Código binario NRZ (Non Return to Zero)
- Código binario RZ (Return to Zero)
- Código polar: NRZ y RZ
- Código bipolar o AMI (Alternate Mark Inversion): NRZ y RZ
- Código HDB_3 (High Density Binary Order 3)
- Código bipolar con sustitución BnZS
- Código Bifase L o Manchester
- Código Bifase M
- Código Bifase S
- Código de Miller
- Código 4B3T (4 Binary 3 Ternary)
- Código MS43 (Monitoring State 43)
- Código CMI (Code Mark Inversion)
- Código PST (Pair Select Ternary)

3.3.1. CODIGO NRZ (NON RETURN TO ZERO)

Es el más sencillo de los códigos. A cada símbolo 0_L o 1_L se le asigna uno de los niveles 0 o A tal como se representa en la figura 3.1.

Este código es utilizado como interfaz a 34368 Kbps. Una señal NRZ no contiene transiciones para una serie de unos o ceros consecutivos, lo cual es una dificultad si se quiere tener una referencia para controlar una señal horaria, como por ejemplo la utilizada para la regeneración del tren de pulsos binarios. Por lo tanto si una codificación NRZ se usa sobre una línea de transmisión, es necesario utilizar un procedimiento para asegurar las transiciones necesarias para la temporización.

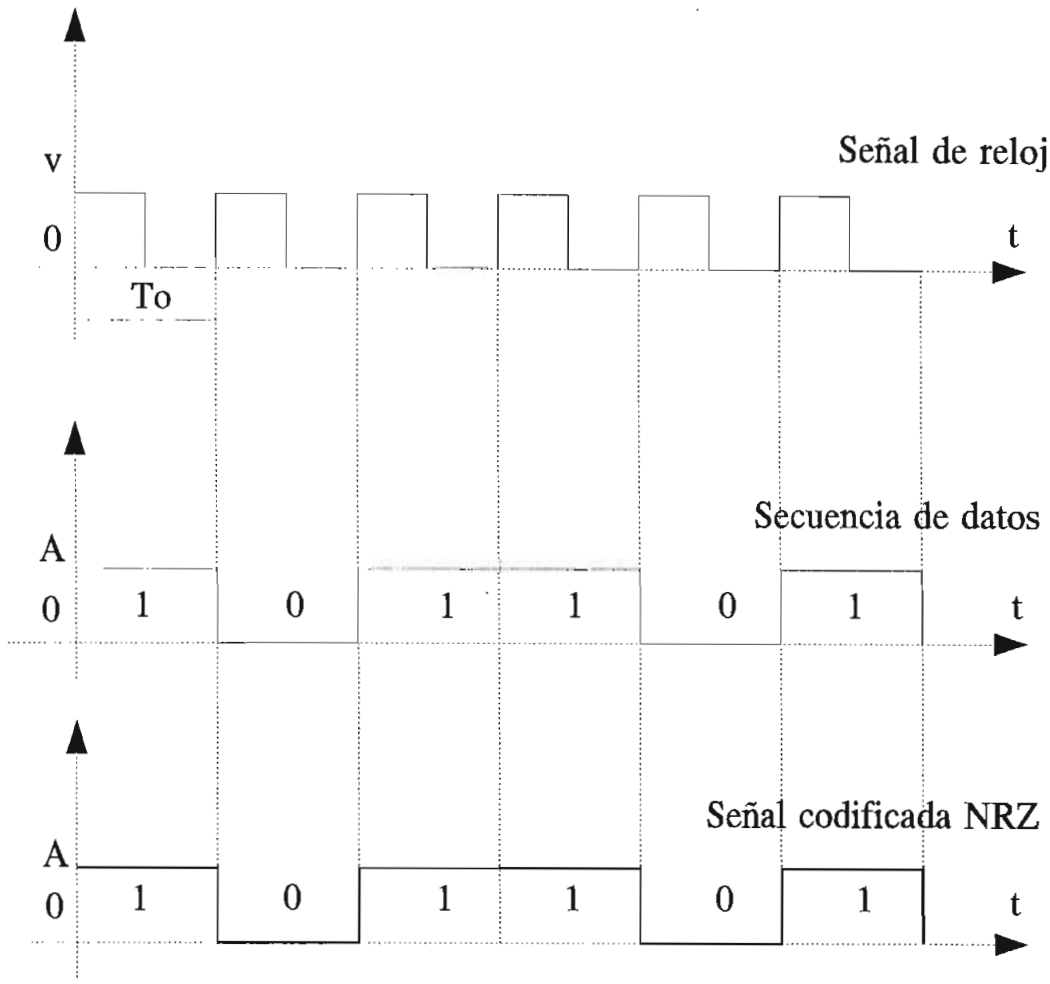


Figura 3.1. Codificación NRZ

Si el sistema de transmisión tiene una respuesta pobre en las componentes de baja frecuencia, esto causará que la señal NRZ en un tren consecutivo de unos o ceros decaiga gradualmente su amplitud. Por lo tanto no sólo el receptor pierde información de temporización durante estos trenes consecutivos de pulsos, sino que también pierde su referencia de amplitud para discriminar óptimamente entre un nivel 1_L y un nivel 0_L.

La existencia de bajas frecuencias en una señal aleatoria de datos es la razón básica para que la codificación NRZ no sea usada en transmisión a larga distancia.

3.3.2. CODIGO RZ (RETURN TO ZERO)

Se dice que el código es con retorno a cero, porque cuando existe un 1_L, durante cierto porcentaje del tiempo T_o, regresa nuevamente a 0.

El tiempo durante el cual 1_L retorna a 0_L es una característica del código. Si el tiempo de retorno a 0_L corresponde a $T_o/2$ se dice que se trata de un código RZ al 50%. La representación de este código puede observarse en la figura 3.2.

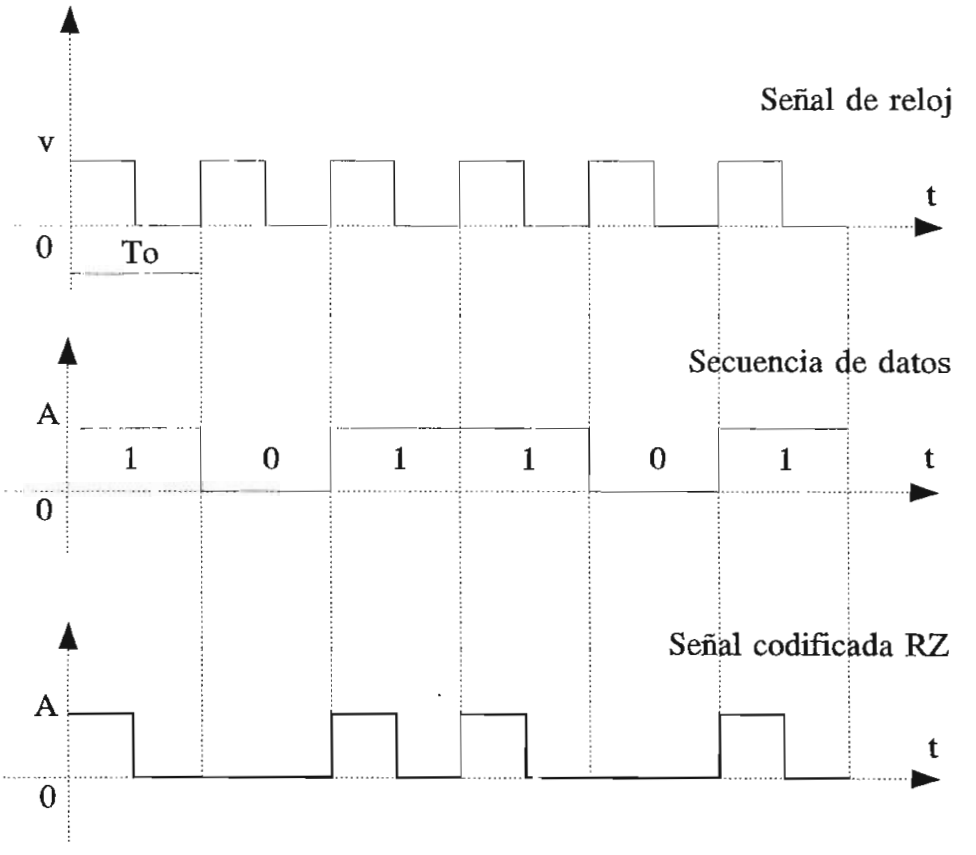


Figura 3.2. Codificación RZ

3.3.2.1. Código Polar

En esta codificación un 1_L se transmite mediante un pulso positivo de amplitud A y un 0_L mediante un pulso de polaridad negativa con amplitud -A. Puede existir códigos

polares que presenten una polaridad contraria a la señalada.

Este código es bastante similar al NRZ, presentando las mismas desventajas que éste, pero es más eficiente. Existe codificación polar RZ y NRZ según el ancho del pulso usado. La rectificación de la señal polar RZ da como resultado una señal periódica de frecuencia de reloj que se puede usar para extraer la sincronización.

La codificación polar NRZ se puede observar en la figura 3.3 para una secuencia de dígitos binarios aleatoria.

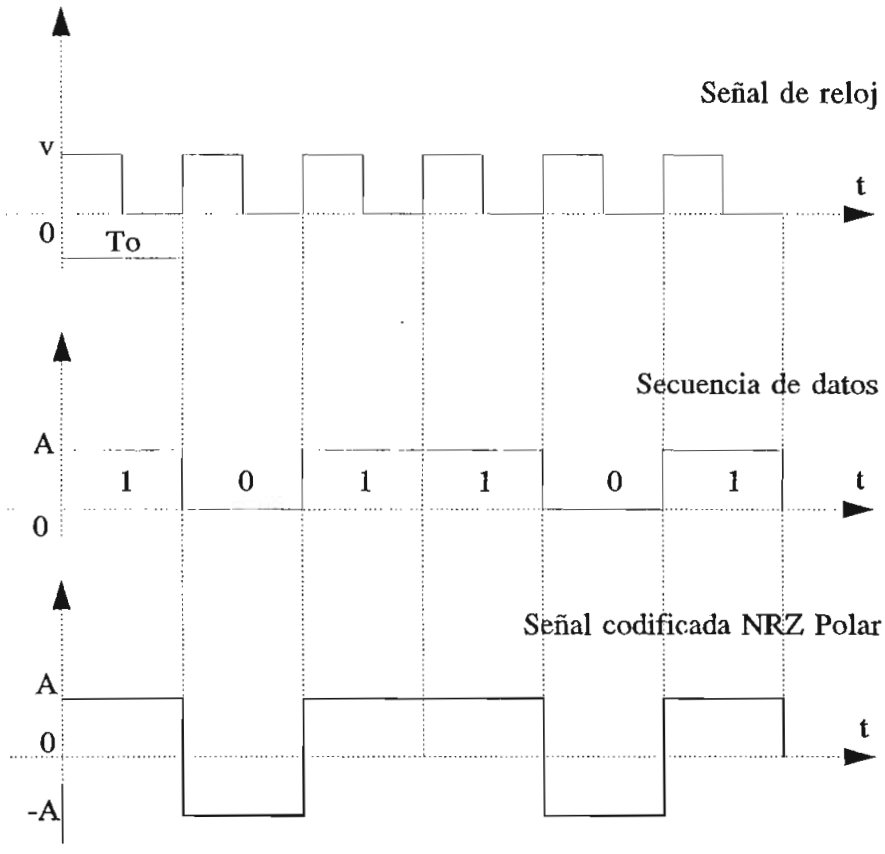


Figura 3.3. Codificación Polar NRZ

3.3.3. CODIGO BIPOLAR O AMI (ALTERNATE MARK INVERSION)

Los códigos precedentes NRZ y RZ poseen ambos una componente continua y por tanto energía en las frecuencias bajas. A fin de suprimir esta componente, se alterna la

polaridad de los 1_L, obteniéndose lo que se denomina un código bipolar o AMI (Alternate Mark Inversion). Este tipo de código presenta la característica de poseer energía cero en las frecuencias bajas, lo que permite franquear sin dificultad los transformadores que encuentra en su recorrido. El proceso de conversión interpreta cada pulso binario o 1_L como una marca positiva o una marca negativa en forma alternada, mientras que la condición de 0_L binario se transmite como tal.

Las marcas transmitidas pueden ocupar un intervalo de tiempo completo que es el caso NRZ o emplear pulsos con retorno a cero RZ como se observa en las figuras 3.4 y 3.5 respectivamente. El escoger entre NRZ y RZ depende del espectro de energía exhibido por el código y la dificultad de extraer información de temporización de la señal de línea transmitida.

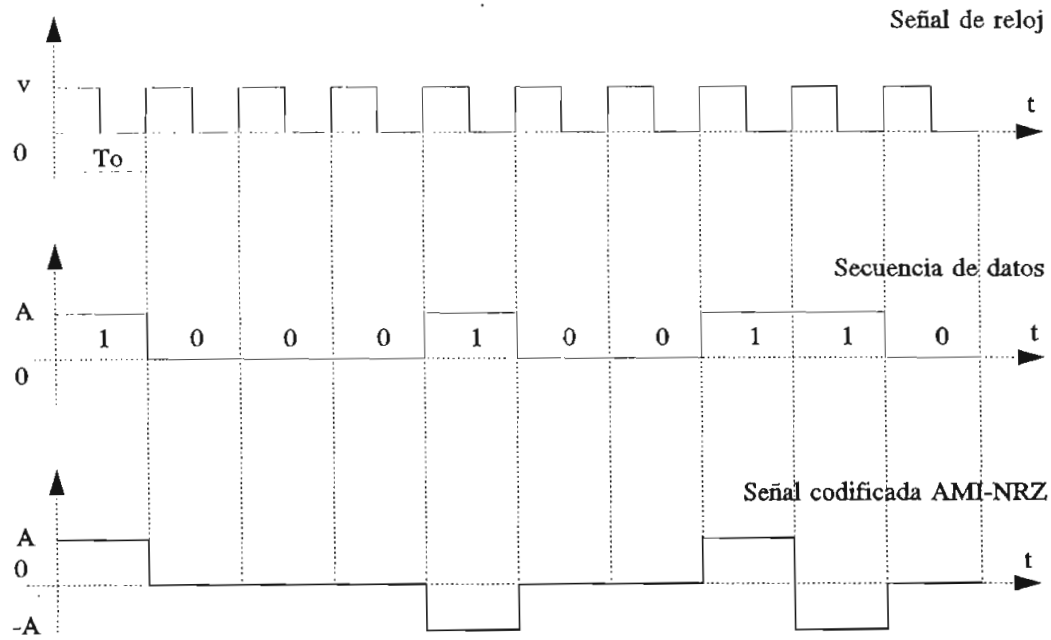


Figura 3.4. Codificación AMI NRZ

Este tipo de código es usado como interfaz a 64 Kbps. Típicamente una relación igual de marca o espacio es usada en el caso de transmisión RZ aunque no es absolutamente necesario, en los sistemas ópticos basados en emisores láser emplean una

relación de marca a espacio de 10 a 30% para incrementar el tiempo de vida del láser.

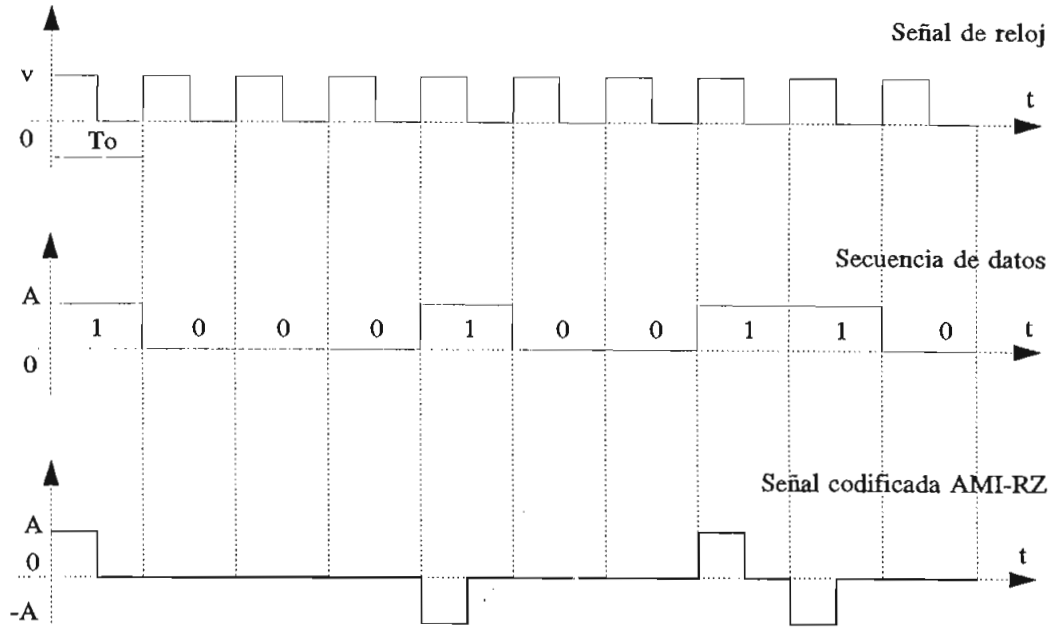


Figura 3.5. Codificación AMI RZ

El código AMI tiene un número igual de marcas positivas y negativas y por lo tanto garantiza un nivel de DC constante. La ausencia de una componente DC es importante por varias razones:

- El nivel de decisión en el cual una marca es considerada presente o ausente será inicialmente lineal.

- El cable de transmisión generalmente suministra una corriente continua a los repetidores. En este caso un nivel constante de DC es importante.

3.3.4. CODIGO HDB_n (High Density Binary de orden n)

El código HDB_n no admite un número superior a n ceros consecutivos para una señal AMI. A continuación se estudiará el código HDB_3 (Código binario de alta densidad de

orden 3), por ser el más utilizado, el cual no admite más de 3 ceros consecutivos, y sustituye con un pulso el cuarto cero consecutivo (pulso de violación). El HDB_3 se trata de un código seudo ternario ya que si bien ocupa tres niveles de tensiones bipolares, el nivel positivo y negativo corresponden al símbolo $1_{\pm}$. Como los unos son reemplazados por pulsos no se requieren 2 pares de conductores, es decir que datos y señal de reloj son transmitidos por el mismo par. Como existe una alta densidad bipolar se asegura la transmisión de la señal de reloj para su recuperación en el receptor.

Se deben tomar las precauciones necesarias para eliminar este pulso suplementario o pulso de violación en el extremo del enlace y para ello es necesario diferenciarlo de los pulsos normales. Este pulso es transmitido con una polaridad idéntica a la del pulso que la precede y se la conoce como violación de paridad (el receptor reconoce que hay una violación pues detecta dos pulsos con la misma polaridad), y por tanto borra la violación. Para asegurar una componente continua nula se deben transmitir tantas violaciones positivas como negativas en forma alternada. Esta condición de alternabilidad de las violaciones para mantener una componente nula, obliga a colocar un pulso de relleno cuando el pulso que precede a la violación no tiene su misma polaridad.

REGLA DE SUSTITUCION DEL HDB_3		
Polaridad del pulso precedente	Número de unos desde la última sustitución	
	Impar	Par
-	000-	+00+
+	000+	-00-

Tabla 3.1. Regla de sustitución de código HDB_3

Es importante anotar que cualquier circuito decodificador HDB_n puede ser usado sin restricción para la decodificación de señales AMI. La presencia de errores ocasionales puede ser detectada observando la alternabilidad de

las violaciones, ya que la introducción de un simple error causará una discontinuidad en la secuencia de alternabilidad de las violaciones. Estos simples errores ocasionales causarán más de un error en la señal binaria decodificada.

El código HDB_3 se usa como interfaz entre multiplexores digitales a nivel de señal de 2048 Kbps, 8448 Kbps y 34368 Kbps.

En la tabla 3.1 se puede observar la regla de sustitución para el HDB_3. Hay que notar que las sustituciones producen violaciones solamente en la cuarta posición, y que sucesivas sustituciones producen polaridades alternadas. A continuación se muestran dos ejemplos de codificación AMI y HDB_3.

1) Secuencia de bits: 110000110000010

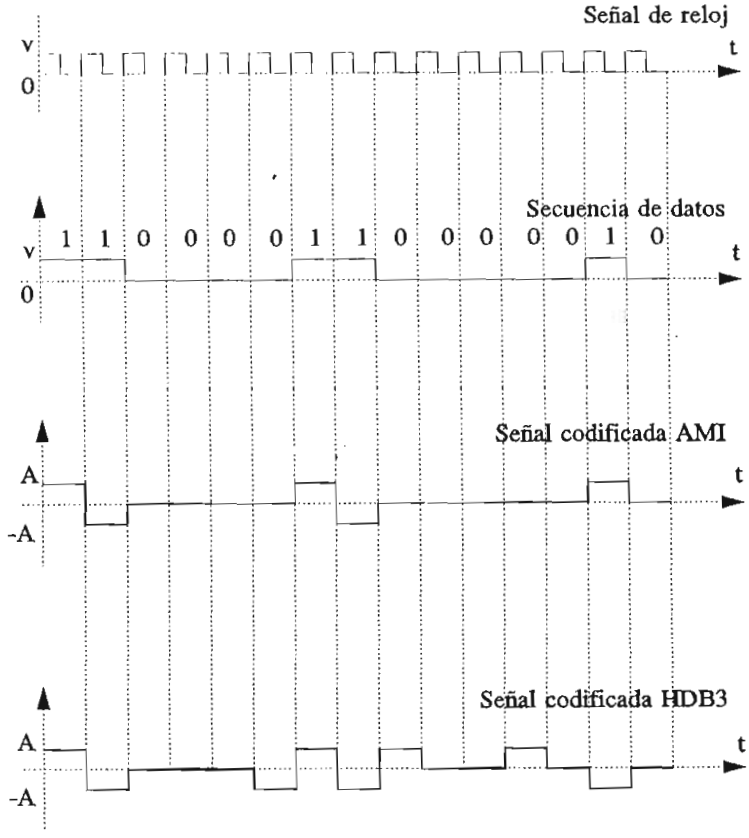


Figura 3.6. Ejemplo 1 de codificación AMI y HDB_3

2) Secuencia de bits: 1100J1000000001

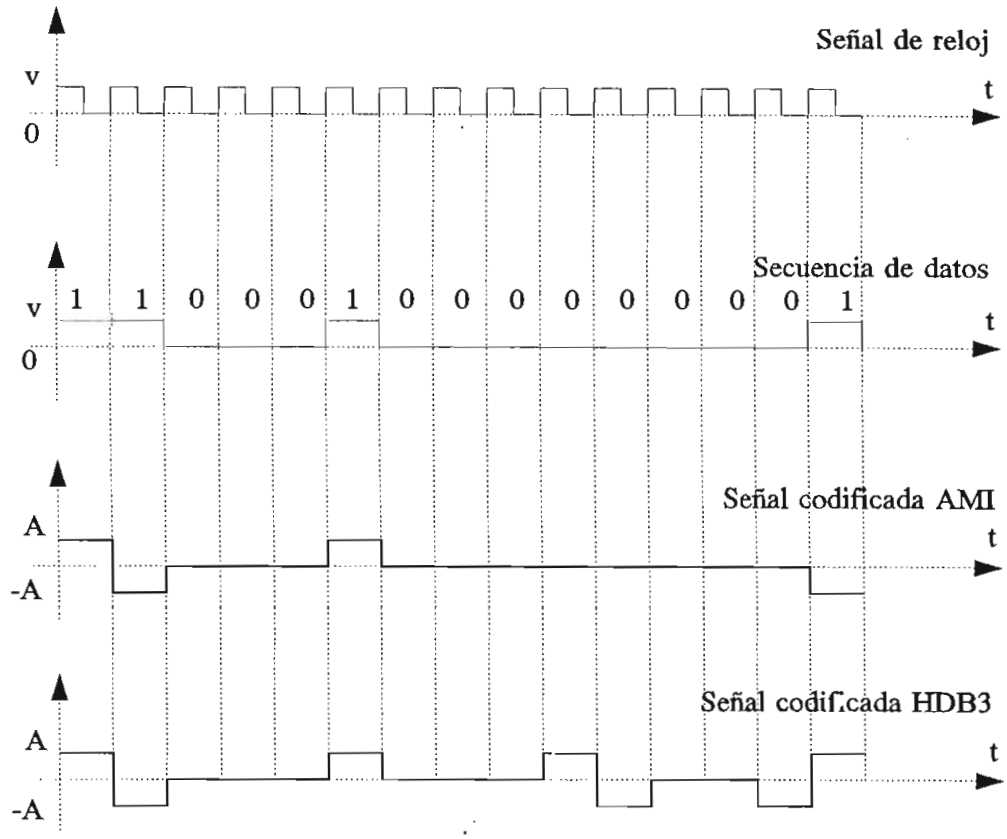


Figura 3.7. Ejemplo 2 de codificación AMI y HDB_3

3.3.5. CODIGO BIPOLAR CON SUSTITUCION DE n CEROS BnZS

El código BnZS (Binary n Zero Substitution) reemplaza n ceros por un código de longitud n que contiene varios pulsos que producen violaciones bipolares.

De esta manera la densidad de pulsos se aumenta, mientras los datos originales se obtienen reconociendo los códigos de violación bipolar y reemplazándolos en el terminal receptor con n ceros.

Para el caso del código B3ZS cada secuencia de tres ceros de la fuente de datos se codifica de acuerdo con la regla dada por tabla 3.2.

NUMERO DE UNOS DESDE LA ULTIMA SUSTITUCION	
IMPAR	PAR
00V	B0V

Tabla 3.2 Regla de Codificación B3ZS

- Donde el valor de B es igual al que tiene la violación V que le sucede.

- Los valores de V deben alternar su polaridad y tener la polaridad del pulso precedente para poder distinguirlos en el receptor.

- Los unos que aparecen fuera de los intervalos de sustitución, se alternan de acuerdo a la regla AMI y tomando en cuenta la polaridad de las sustituciones (el 1er uno del tren de datos se considera positivo).

Lo expresado anteriomente se puede ver con mayor detalle en la tabla 3.3, en la que se indica la regla de sustitución B3ZS.

REGLA DE SUSTITUCION DEL CODIGO B3ZS		
Polaridad del pulso precedente	Número de unos desde la última sustitución	
	Impar	Par
- +	00- 00+	+0+ -0-

Tabla 3.3. Regla de sustitución del código B3ZS

En la figura 3.8 se observa un ejemplo de codificación B3ZS para la secuencia:

1 0 1 0 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0 0 1

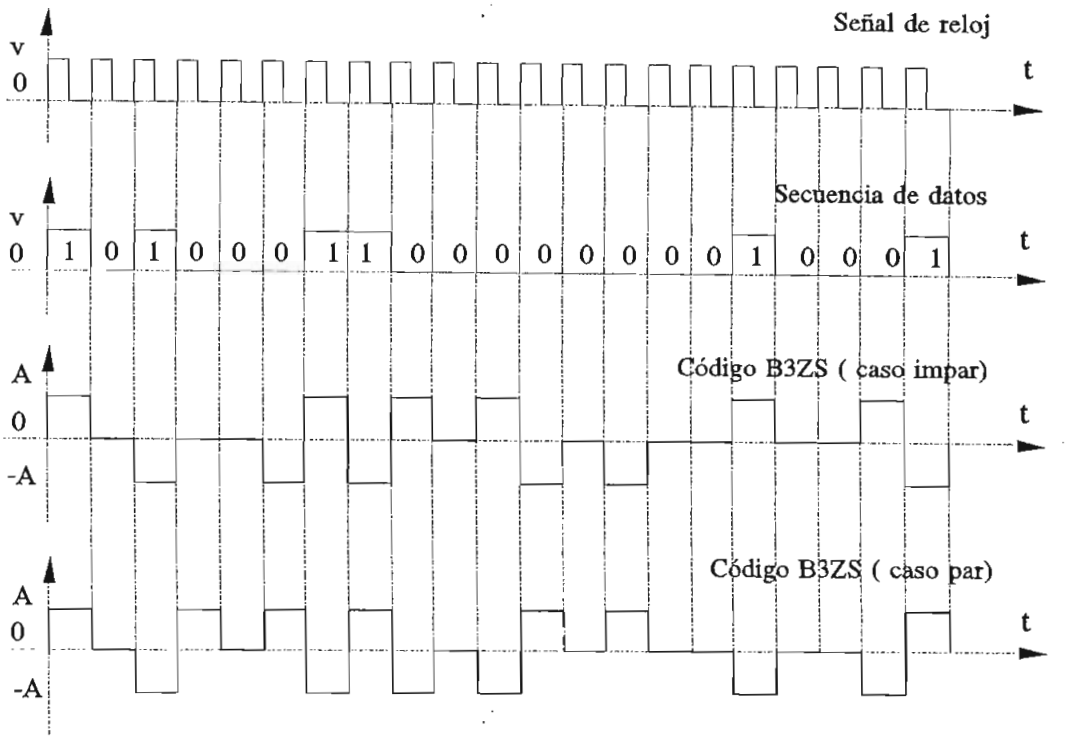


Figura 3.8. Ejemplo de codificación B3ZS

Se utiliza el signo + para indicar un pulso positivo, y el signo - para un pulso negativo y 0 para indicar una ausencia de pulso.

Hay dos posibles secuencias de codificación dependiendo si un número par o impar de pulsos han sido transmitidos antes de la primera violación, y éstas son :

Secuencia:	101	**000**	11	**000**	**000**	001	**000**	1
Caso Impar:	+0-	00-	+-	+0+	-0-	00+	00+	-
Caso Par:	+0-	+0+	-+	-0-	+0+	00-	00-	+

Como se puede ver este código B3ZS aumenta la densidad mínima de pulsos en línea. En realidad la densidad mínima es 33%, mientras que la densidad promedio está sobre el

60%. De aquí se deduce que el código B3ZS garantiza una componente continua mínima. Se debe notar que todos los códigos B3ZS garantizan una información de temporización continua sin restricciones sobre la fuente de datos, por lo que codificación B3ZS se puede aplicar de una manera totalmente transparente.

Otro algoritmo de codificación usado es el B6ZS, el cual produce violaciones bipolares en la segunda y quinta posición de la secuencia sustituida. La regla de codificación para este código se muestra a continuación en la tabla 3.4.

REGLA DE SUSTITUCION DEL B6ZS	
Polaridad del pulso que precede inmediatamente a los 6 ceros a sustituirse	Sustitución
- +	0 - + 0 + - 0 + - 0 - +

Tabla 3.4. Regla de sustitución del código B6ZS

Un ejemplo de la codificación B6ZS se observa en la figura 3.9 para la siguiente secuencia de datos binarios:

1 0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1

Existen 2 posibles secuencias de codificación dependiendo que valor toma el primer 1_L.

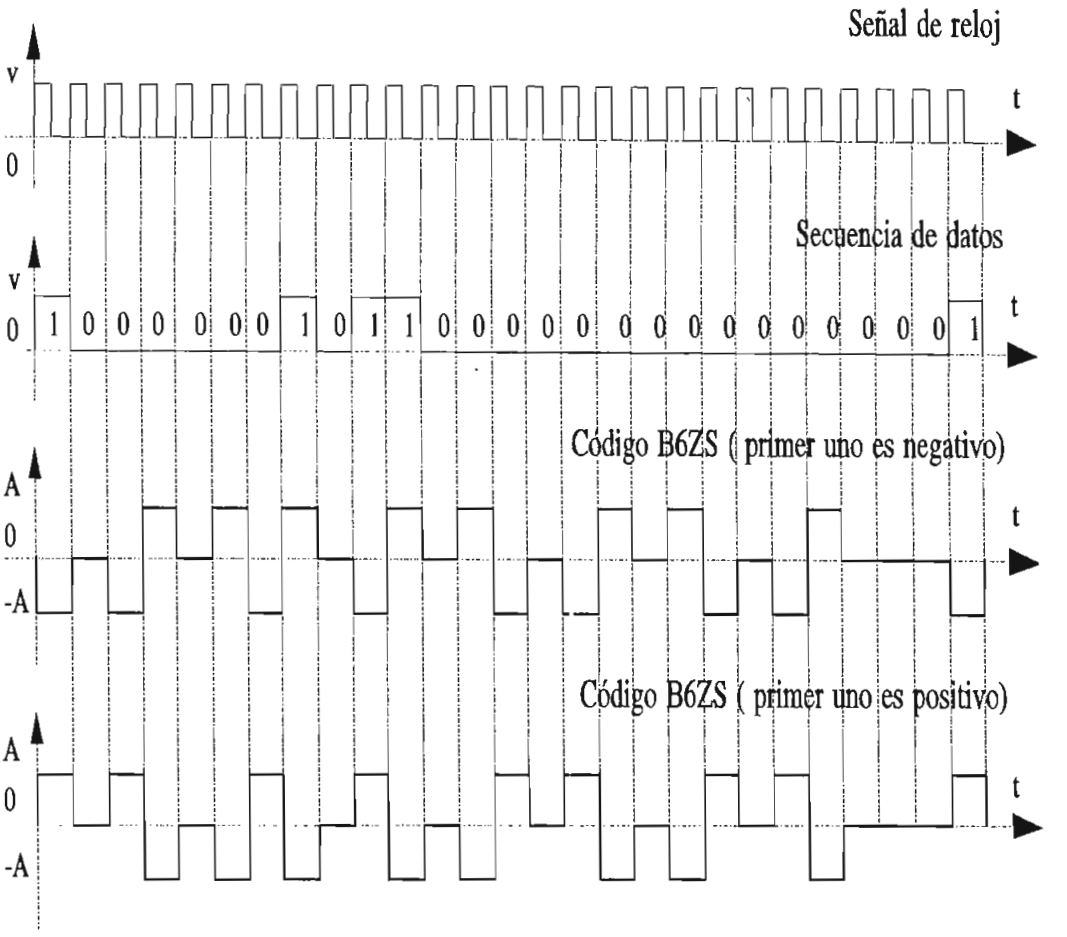


Figura 3.9. Ejemplo de codificación B6ZS

1) Si el primer 1_L es negativo la codificación es:

- 0 - + 0 + - + 0 - + 0 + - 0 - + 0 + - 0 - + 0 0 0 -

2) Si el primer 1_L es positivo la codificación es:

+ 0 + - 0 - + - 0 + - 0 - + 0 + - 0 - + 0 + - 0 0 0 +

3.3.6. CODIGO BIFASE L O CODIGO MANCHESTER

Para este código las transiciones se dan a $T_o/2$ del intervalo del estado lógico, entre los niveles extremos +A y -A. El símbolo 1_L tiene una polaridad prefijada y el símbolo 0_L la opuesta, tal como se puede ver en la figura 3.10.

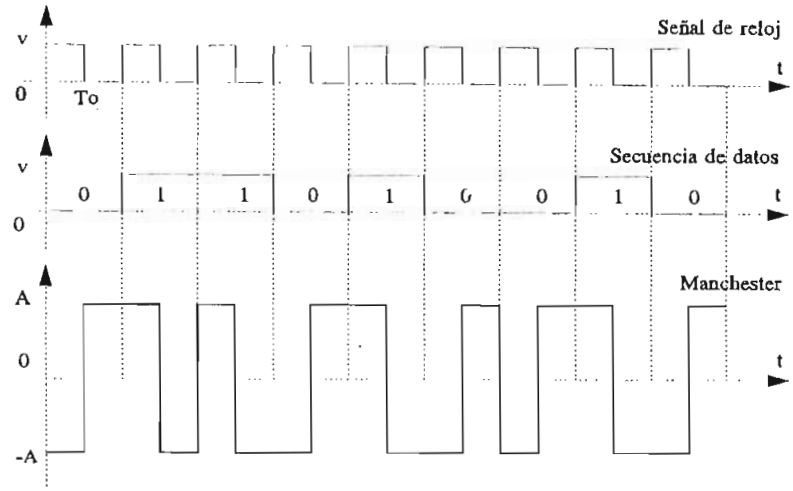


Figura 3.10.Codificación Manchester

3.3.7. CODIGO BIFASE M

Es un código de dos niveles, +A y -A. Una transición aparece siempre al principio del intervalo. El símbolo 1_L produce otra transición medio intervalo $T_o/2$ después, el símbolo 0_L no produce transición. Un ejemplo de este clase de codificación se puede observar en la figura 3.11.

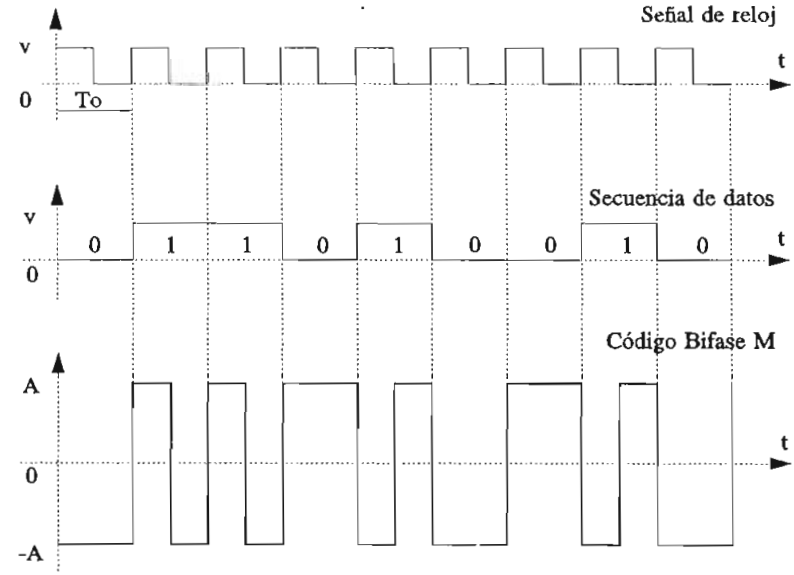


Figura 3.11. Codificación bifase M

3.3.8. CODIGO BIFASE S

Es un código también de dos niveles contrario al código bifase M. En este código el símbolo 1_L es ahora el que no produce transición en la mitad del intervalo T_o, como se puede apreciar en la figura 3.12, en la que se codifica una secuencia de datos binarios.

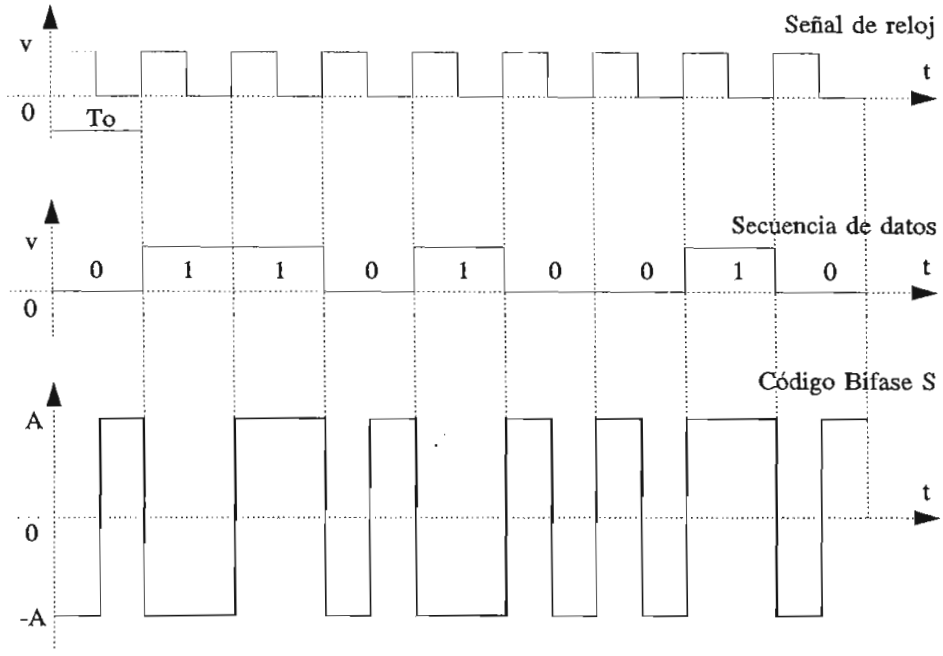


Figura 3.12. Codificación bifase S

3.3.9. CODIGO DE MODULACION POR RETARDO (DM) O CODIGO DE MILLER

Las transiciones se efectúan entre los niveles -A y A. El símbolo 1_L produce una transición en el punto medio del intervalo unitario T_o.

El símbolo 0_L no produce ninguna transición, a menos que vaya seguido por otro 0_L, en cuyo caso se produce una transición entre ambos ceros, al final del primer intervalo unitario como se ve en la figura 3.13.

Este código es insensible a la ambigüedad inicial de 180° que aparece en los códigos NRZ y de Manchester. Es eficiente en términos de ancho de banda cuando no se requiere de una buena respuesta en bajas frecuencias.

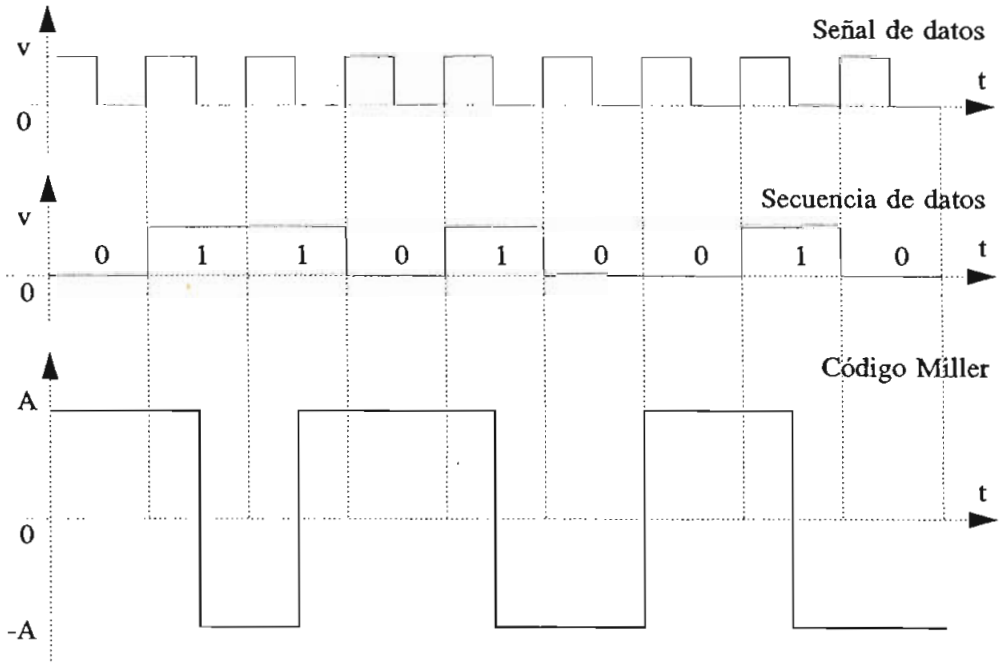


Figura 3.13. Codificación de Miller

3.3.10. CODIGO 4B3T

En este código se hacen corresponder tres señales ternarias a cuatro señales binarias.

En el caso del código 4B3T el codificador convierte grupos de 4 dígitos binarios en grupos de tres dígitos ternarios. Dado que las palabras binarias de 4 bits sólo requieren 16 de las 27 posibles palabras del código ternario de 3 dígitos, existe una cierta flexibilidad en la selección de los códigos ternarios. Esta redundancia permite detectar errores de transmisión.

En la tabla 3.5 se presentan las palabras binarias y su posible codificación, las palabras ternarias en la columna

del medio están balanceadas en su contenido de DC.

Las palabras del código de la primera y tercera columna se eligen para mantener el balance de DC. Si se transmiten más pulsos positivos que negativos, se debe seleccionar la tercera columna. Cuando la polaridad entre pulsos positivos y negativos cambia (más pulsos negativos) se debe elegir la primera columna. Se debe notar que la palabra 000 no es utilizada, por esto es posible mantener un fuerte contenido de temporización.

Dado que los circuitos de codificación operan en una frecuencia que es 3/4 de la entrada, hace que el 4B3T sea una codificación muy empleada para sistemas de alta capacidad.

Palabra binaria	Palabra ternaria			Polaridad
	Modo +	Modo 0	Modo -	
0000		0-+		0
0001		-+0		0
0010		-0+		0
1000		0+-		0
1001		+-0		0
1010		+0-		0
0011	+-+		-+-	1
1011	+00		-00	1
0101	0+0		0-0	1
0110	00+		00-	1
0111	-++		+--	1
1110	++-		--+	1
1100	+0+		-0-	2
1101	++0		--0	2
0100	0++		0--	2
1111	+++		---	3

Tabla 3.5. Regla de codificación 4B3T

En la figura 3.14 se observa el diagrama de estados del codificador 4B3T.

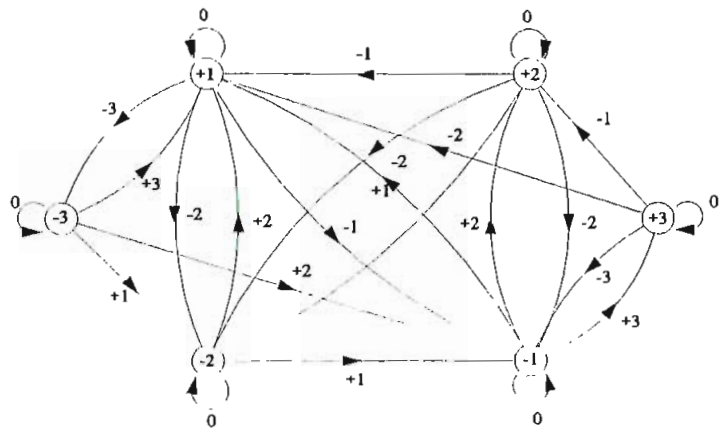


Figura 3.14. Diagrama de estados de codificación 4B3T

Es importante indicar que el diagrama de estado no contiene ningún estado de polaridad 0; de igual manera se ve que la polaridad acumulada siempre permanecerá en el rango -3 a +3, esto en la práctica no siempre ocurre por errores ocasionales de transmisión.

Un ejemplo de codificación 4B3T se observa en la figura 3.15 para una secuencia binaria:

Palabra binaria:

0000 0110 1000 0111 1011 0101 1111 0000 0000 0000 1001

Polaridad acumulada:

(0) (-1) (0) (-1) (+1) (-1) (+3) (0) (0) (0) (-2)

Palabra ternaria:

0-+ 00- 0+- +-- +00 0-0 +++ 0-+ 0-+ 0-+ +-0

Polaridad inicial:

(+2) (+1) (+1) (-1) (+1) (-1) (+3) (+3) (+3) (+3) (+1)

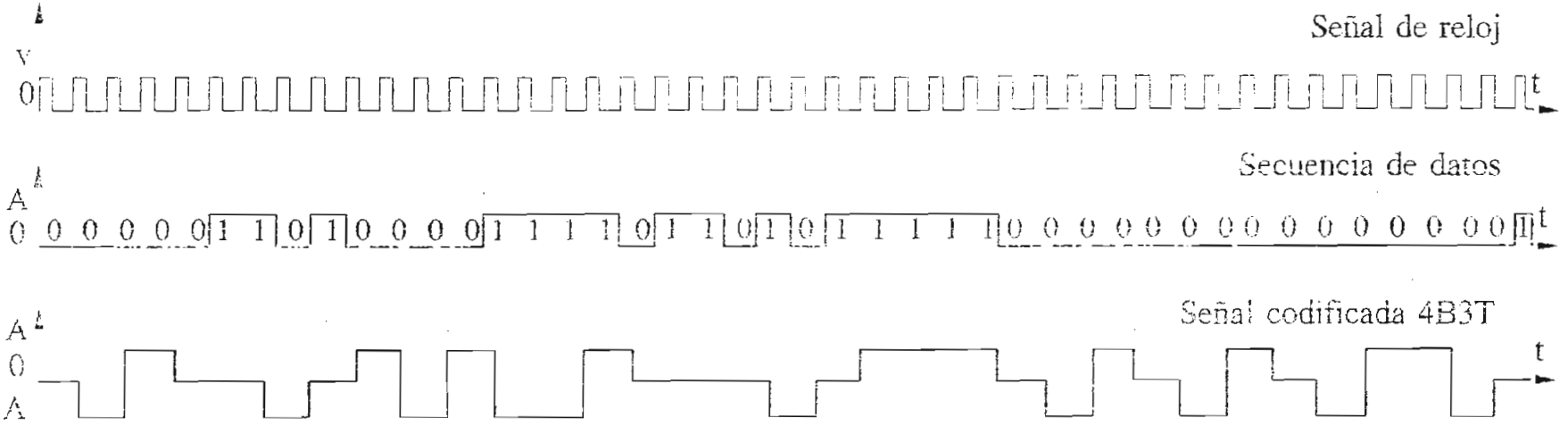


Figura 3.15. Ejemplo de codificación 4B3T

3.3.11. CODIGO MS43

El código MS43 (Monitoring State 43) fue adoptado por la Administración Alemana de Telecomunicaciones para las redes digitales de servicios integrados (RDSI).

El MS43 tiene un concepto similar al 4B3T descrito anteriormente. En este caso 4 dígitos binarios se codifican para dar una secuencia ternaria de 3 dígitos con lo cual se reduce el ancho de banda ocupado por la señal transmitida.

Palabras binarias	Palabras ternarias				Paridad
	Modo 1	Modo 2	Modo 3	Modo 4	
0011	0-+	0-+	0-+	0-+	0
0101	-0+	-0+	-0+	-0+	0
0110	-+0	-+0	-+0	-+0	0
1110	+-0	+-0	+-0	+-0	0
1101	+0-	+0-	+0-	+0-	0
1011	0+-	0+-	0+-	0+-	0
1000	+-+	+-+	+-+	---	+1,+1,+1,-3
1001	00+	00+	00+	--0	+1,+1,+1,-2
1010	0+0	0+0	0+0	-0-	+1,+1,+1,-2
1100	+00	+00	+00	0--	+1,+1,+1,-2
0111	-++	-++	--+	--+	+1,+1,-1,-1
1111	++-	++-	+--	+--	+1,+1,-1,-1
0001	++0	00-	00-	00-	+2,-1,-1,-1
0010	+0+	0-0	0-0	0-0	+2,-1,-1,-1
0100	0++	-00	-00	-00	+2,-1,-1,-1
0000	+++	-+-	-+-	-+-	+3,-1,-1,-1

Tabla 3.6. Regla de codificación MS43

Existen más de 2 modos de codificación que pueden ser seleccionados. En la tabla 3.6. se encuentra la forma de conversión del código MS43 donde se tienen las 16 combinaciones

de cuatro bits y los cuatro modos posibles de combinaciones ternarias.

La selección entre los diferentes modos se combina de tal forma que el número de estados +, 0, - sea equiprobable.

El diagrama de transición o de estados para el código MS43 está dado en la figura 3.16.

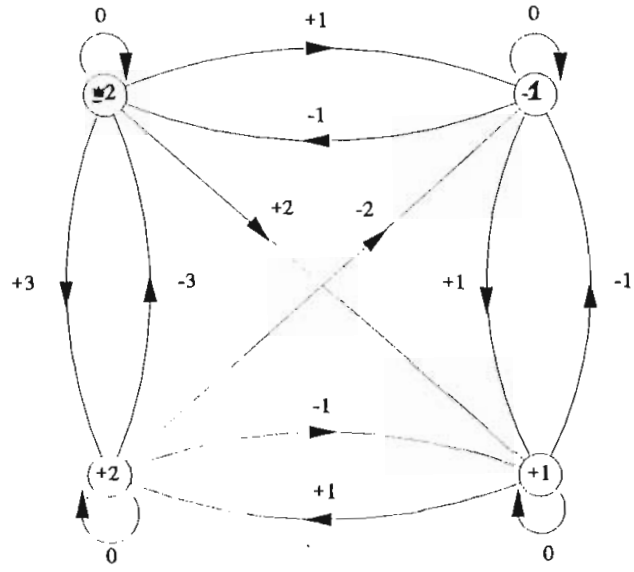


Figura 3.16. Diagrama de transición del código MS43

3.3.12. CODIGO CMI (CODE MARK INVERSION)

Basados en el código Manchester se han desarrollado una gran variedad de códigos de línea. De éstos el código CMI, de inversión de marcas codificadas, es el más representativo. Este código bipolar de dos niveles o binario responde a la siguiente regla de codificación: el 0_L en vez de ser representado por una ausencia de señal, es emitido con un cambio de polaridad negativo a positivo, que se produce a la mitad del período $T_o/2$ del bit. El 1_L es codificado con pulsos alternados de duración T_o.

Este código es equivalente a uno de velocidad doble donde el símbolo 1_L se halla alternativamente con 11 y 00 y el símbolo 0_L con 01.

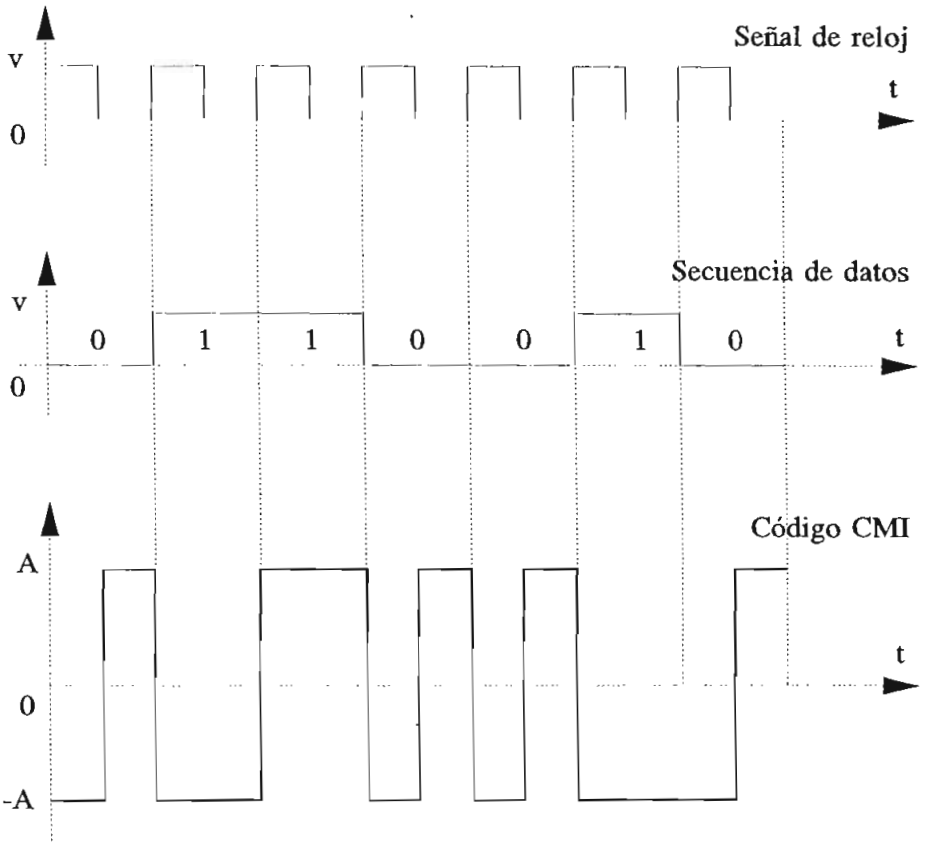


Figura 3.17. Codificación CMI

En la figura 3.17 se muestra un ejemplo de codificación CMI. Aquí se repite la equiprobabilidad de polaridad y la consiguiente ausencia de componente continua; al igual que en los códigos HDB_n, también se repite la alta densidad bipolar para mantener presente la señal de reloj y la posibilidad de transmitir datos y temporización por el mismo par.

El código CMI se utiliza como código de interfaz entre multiplexores digitales a nivel de señal de alta velocidad a 139264 Kbps.

3.3.13. CODIGO PST (PAIR SELECTED TERNARY)

En el código PST (selección de pares ternarios) la entrada binaria se transforma para la transmisión en un código de dos dígitos ternarios, como se puede ver en la tabla 3.7, donde se indica la regla de codificación para este código.

La forma de codificar es la siguiente: por ejemplo a la entrada binaria 01 le corresponde un primer modo 0+ cuando vuelva a aparecer le corresponderá 0-, en la tercera aparición le corresponde 0+, es decir, se alternan los modos. De igual manera ocurre para el caso de la entrada binaria 10. Las entradas binarias 00 y 11 no cambian el valor de sus modos.

Entrada binaria	Modo +	Modo -
00	-+	-+
01	0+	0-
10	+0	-0
11	+-	+-

Tabla 3.7. Regla para codificación PST

Este formato de codificación no solamente asegura una fuerte componente de temporización, sino que también anula la componente de corriente continua DC, al conmutar entre dos modos y así mantener un balance entre los pulsos positivos y negativos.

En la figura 3.18 se tiene un ejemplo de codificación PST para el siguiente tren de pulsos binarios:

0 1 0 0 1 1 1 0 1 0 1 1 0 0

Existen dos posibles soluciones dependiendo si el codificador está en el modo positivo o negativo en el comienzo de la secuencia.

	01	00	11	10	10	11	00
Modo + (Caso 1)	0+	-+	+-	-0	+0	+-	-+
Modo - (caso 2)	0-	-+	+-	+0	-0	+-	-+

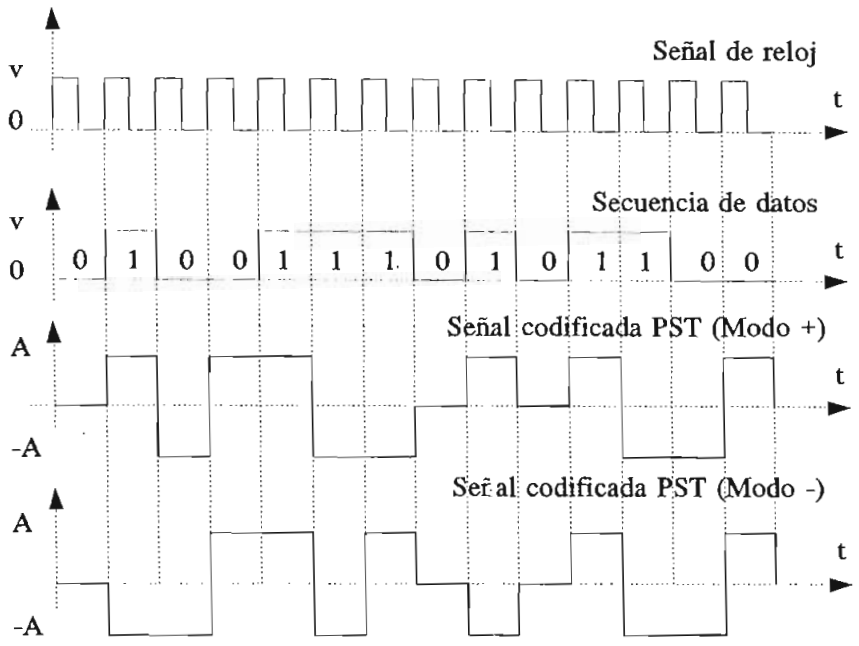


Figura 3.18. Codificación PST

Un inconveniente de PST es que los datos binarios deben ser encuadrados o entramados en pares, De aquí que un decodificador PST debe reconocer y mantener los límites, lo cual no es difícil ya que datos aleatorios que son transmitidos con errores producirán códigos no permitidos (00, ++).

3.4. COMPARACION DE LA EFICIENCIA DE LOS CODIGOS DE LINEA

Para la comparación de los códigos de línea se debe considerar aspectos como los siguientes:

- Si en el proceso de transmisión la forma de codificación ha sido la más adecuada y si la señal es recuperada fielmente en el receptor.

- La codificación debe ser transparente a la información binaria a transmitirse, es decir, que no deben existir restricciones con el tipo de mensaje a transmitir.

- Por razones establecidas anteriormente, no debe tener componente continua DC significativa y la energía en bajas frecuencias debe ser la mímina posible.

- La señal codificada debe tener un considerable número de transiciones por cero que garanticen una fuerte componente de temporización, y facilite la posterior extracción de la señal de reloj necesaria para la sincronización de la señal en el receptor.

- Facilidad de implementación del codificador, no siempre el código que es más idóneo teóricamemente, se lo puede implementar fácilmente en la práctica, o si lo es, los costos de construcción son elevados y esto restringe su utilización.

- Otro aspecto importante que se tiene en cuenta es el ancho de banda de la señal (determinado por el espectro de frecuencias del código utilizado en la transmisión y su relación directa con la velocidad de transmisión que está dada por la ley de Hartley-Shannon), estableciendo que un aumento de la velocidad de transmisión de una señal disminuirá el ancho de banda del canal, pero con el consiguiente aumento de la tasa de errores.

- Un adecuado comportamiento de la señal codificada ante la presencia de ruido aditivo, ya que es el factor principal para que se produzca errores en la transmisión. Por esto existen estructuras de codificación que sacrifican la eficiencia de la transmisión, para permitir la facilidad de detección.

Por lo tanto siendo el ruido aditivo la principal causa para introducir errores en la señal, es importante analizar las codificaciones desde el punto de vista de la

probabilidad de error BER (Bit Error Rate) en función de la relación que existe entre la señal codificada y el ruido (relación señal a ruido S/N).

Es importante indicar que el BER no sólo dependerá de la forma de codificación sino de los procesos de modulación y demodulación de la misma, así como de los canales de transmisión utilizados.

En la figura 3.19 se compara el BER teórico en función de la relación señal a ruido (S/N) para diversos códigos de línea; en esta figura se puede observar que las codificaciones NRZ y Manchester tienen mejor comportamiento ante la presencia de ruido, comparándolas con la codificación de Miller, PST y Bipolar.

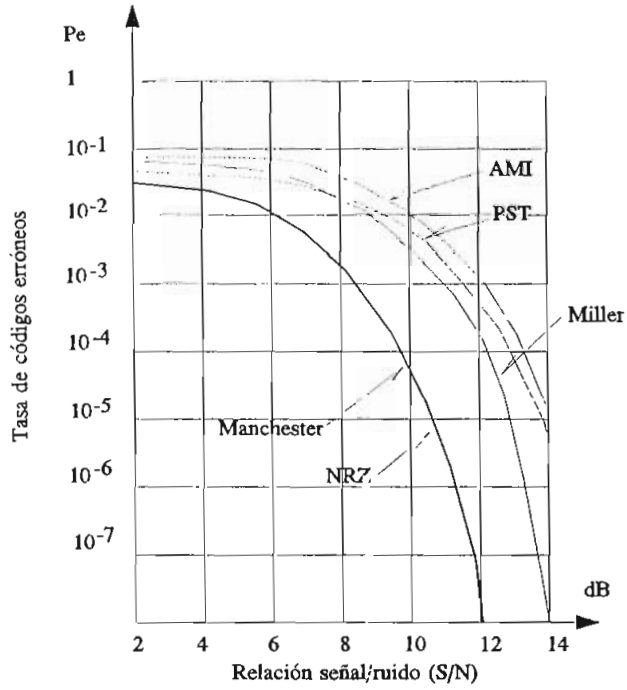


Figura 3.19. Comparación del BER en función de la relación señal a ruido para los diversas codificaciones

Si se analiza para una probabilidad de error dada, el código de Miller necesitará 3 dB más en la relación señal a

ruido que el NRZ y Manchester, mientras que el Bipolar y el PST necesitarán aproximadamente de 6 dB más, para mantener la misma calidad o el mismo BER.

Esto se debe a que en la recepción de una señal con codificación de Miller se tienen 4 elementos para discernir, transición positiva, negativa, pulso y espacio lo cual hace que la decisión sea más difícil, mientras que en NRZ sólo hay dos posibilidades, presencia de pulso o ausencia de éste.

Códigos NRZ

Es el código más sencillo por su implementación, son muy utilizados para generar datos digitales. La desventaja de la codificación NRZ es que tiene componente DC y no tiene una buena componente de temporización lo que en algunos casos no permite la recuperación de la señal de reloj en forma adecuada.

Códigos RZ

En esta codificación el pulso tendrá menor duración que los NRZ lo que aumentará la velocidad. El ancho de banda de la señal es mayor, tiene D.C y mayor capacidad de sincronización que el NRZ. Debido a su simplicidad es usada en algunas transmisiones elementales y en equipos de grabación.

Códigos Bifase

Los códigos bifase producen transiciones en el intervalo del bit (en la mitad del periodo $T_o/2$). En estos códigos el ancho de banda es mayor que para el NRZ. Tienen la ventaja de tener una gran componente de temporización, debido a que existe una transición predecible. Otra ventaja es que no presentan componentes de corriente continua DC. Facilitan la detección de errores, ya que la ausencia de una transición esperada es la indicación de que ha ocurrido un error.

Los códigos bifase son técnicas de codificación muy utilizadas para transmisión de datos, teniendo mayor aplicación en la grabación de cintas magnéticas.

Código de Miller

En este código existe al menos una transición en dos periodos de bit, debido a esto tiene la capacidad de recuperación de sincronización y un ancho de banda menor que los códigos bifase.

Código Bipolar o AMI

La capacidad de sincronización no es muy buena cuando se tiene varios 0 seguidos, por lo que se intenta anular la componente continúa en base a la alternabilidad de la polaridad de los pulsos. Su ancho de banda no es excesivo y menor que los bifase. Si se rectifica una señal AMI se obtiene una señal NRZ que tiene una componente discreta en la frecuencia de reloj. Una señal bipolar requiere dos veces tanta potencia como la que se requiere para una señal polar.

Código HDB_3

La principal característica de HDB_3 es que posee un gran desempeño en la sincronización, así como una pequeña componente de energía a bajas frecuencias.

Código 4B3T

Otro de los bipolares que tiene ancho de banda aceptable, tiene gran cantidad de energía a bajas frecuencias.

Su principal ventaja es la disminución de la velocidad de transmisión a un factor de 3/4 de la binaria Es

utilizada en transmisión por fibra óptica

Código PST

Este código mejora la sincronización ya que provee de pulsos positivos y negativos. El ancho de banda es mayor que el del código AMI y del PST, tiene un nivel de energía más alto que el AMI.

3.5. PROGRAMA DE SIMULACION DE LOS CODIGOS DE LINEA

El programa de simulación de códigos de línea está compuesto de un conjunto de subrutinas llamadas también procedimientos, los mismos que permiten visualizar en el monitor de un computador tres señales: la señal de reloj, la señal binaria o los bits de datos que van a codificarse y la señal codificada de acuerdo con el tipo de código elegido. Esta presentación es de tipo didáctica ya que facilita su comprensión así como la enseñanza y el aprendizaje de los diversos códigos que se estudian en la teoría de Transmisión Digital.

El ingreso de los bits de datos se lo puede hacer de dos formas: manual de acuerdo con la secuencia que se desee codificar y automática basada en la generación aleatoria de los bits mediante una subrutina del programa, que se detalla en el Capítulo 6.

Una vez que se tienen los bits de datos para ser codificados, éstos deben ingresar a los diferentes procedimientos de codificación, que permiten a su vez obtener los coeficientes que se utilizan en el proceso para graficar la señal codificada, mediante el procedimiento **GRAFICAR** del programa principal. Este principio básico que es utilizado para obtener las gráficas de las señales se muestra en el esquema de la figura 3.20.(a). En esta figura que dependiendo del pulso se multiplica un coeficiente positivo, negativo o cero por el valor del pulso.

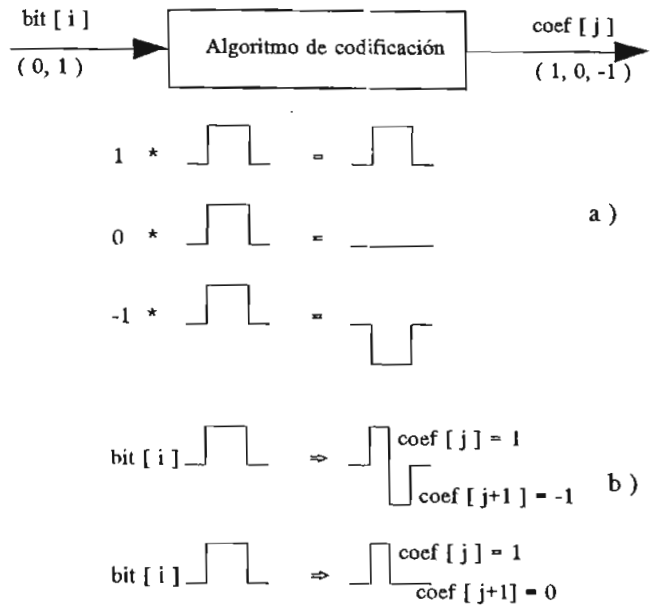


Figura 3.20. Forma de obtener los pulsos de las señales codificadas

En el caso de los códigos que tienen transiciones a la mitad del intervalo T_o, al bit de datos se le asigna dos coeficientes que permiten obtener las transiciones en las señales de los códigos como se observa en la figura 3.20.(b).

Las variables generales que se utilizan en los procedimientos de simulación de los códigos de línea son las siguientes:

bit[i]: Es un vector en el que se almacenan los bits de datos binarios y que se ingresan o se generan aleatoriamente; los valores que toman los bits son **0** o **1** y representan a los símbolos 0_L y 1_L respectivamente.

coef[j]: Es un vector en el que se almacenan los coeficientes que multiplicados por un pulso rectangular, permiten obtener las señales codificadas. Los valores que pueden tomar los coeficientes son: **1**, **0** y **-1** que representan un pulso positivo, ausencia de pulso y un pulso negativo respectivamente.

i, j: Son variables enteras que se utilizan como contadores. La variable **i** es utilizada para contar los bits, mientras que la variable **j** se utiliza para la cuenta de los coeficientes.

A continuación se indican los algoritmos utilizados para cada uno de los procedimientos de los códigos de línea propuestos, así como el diagrama de flujo respectivo.

3.5.1. Algoritmo para el código NRZ

Para este código los coeficientes de salida son iguales a los bits de entrada, esto se debe a que las señales binarias que se utilizan en los sistemas digitales son de no retorno a cero NRZ. El diagrama de flujo para la implementación del algoritmo del código NRZ se lo puede ver en la figura 3.21.

3.5.2. Algoritmo para el código RZ

En el código RZ se hace corresponder a un **bit[i]** dos coeficientes **coef[j]** y **coef[j+1]**. Para 0_L, **coef[j]=coef[j+1]**=0, lo que da ausencia de pulso, en cambio para 1_L **coef[j]**=1 y **coef[j+1]**=0, lo que da como resultado la forma característica de la señal para este código; en la figura 3.22 se indica el diagrama de flujo correspondiente.

3.5.3. Algoritmo para el código NRZ Polar

Para el código NRZ Polar cuyo diagrama de flujo se encuentra en la figura 3.23, se utiliza el siguiente algoritmo de codificación:

Se ingresa un bit igual a 0_L y se le asigna un coeficiente **coef**=-1, caso contrario, es decir si el bit es igual a 1_L se asigna el coeficiente **coef**=+1.

3.5.4. Algoritmo para el código RZ Polar

Para el código RZ Polar, se hace corresponder al **bit[i]**, dos coeficientes **coef[j]** y **coef[j+1]**, de la siguiente manera:

Para 0_L, los coeficientes correspondientes son **coef[j]**=-1 y **coef[j+1]**=0.

Si **bit[i]**=1, le corresponden los coeficientes **coef[j]**=1 y **coef[j+1]**=0; el retorno a cero de la señal se lo hace a la mitad del periodo T_0. El diagrama de flujo del algoritmo para el código RZ Polar se encuentra en la figura 3.24.

3.5.5. Algoritmo para el código AMI

El código AMI que se representa es el código AMI-NRZ o código bipolar, no se considera el código AMI-RZ por ser este código una extensión del AMI-NRZ.

Para la programación en el caso de tener un bit igual a **0**, se asigna el coeficiente **COEF(i)**=0. Si el bit es igual a **1**, se recurre a una nueva variable **COF**, que tiene un valor inicial de **-1** con la cual se obtienen los coeficientes alternados **1** o **-1** que corresponderán a la variable **COEF**, de acuerdo con la teoría del código AMI-NRZ. Para una mejor comprensión de lo explicado, el diagrama de flujo se muestra en la figura 3.25.

3.5.6. Algoritmo para el código HDB_3

Para el programa de codificación HDB_3, teniendo presente que esta codificación no permite cuatro ceros consecutivos, se procede primero a verificar si a partir del bit que se lee, con los tres bits siguientes se conforma una secuencia **0000**, en cuyo caso se procede a asignar los valores de los coeficientes **COEF(i)** de acuerdo con la tabla de codificación HDB_3, (tabla 3.1).

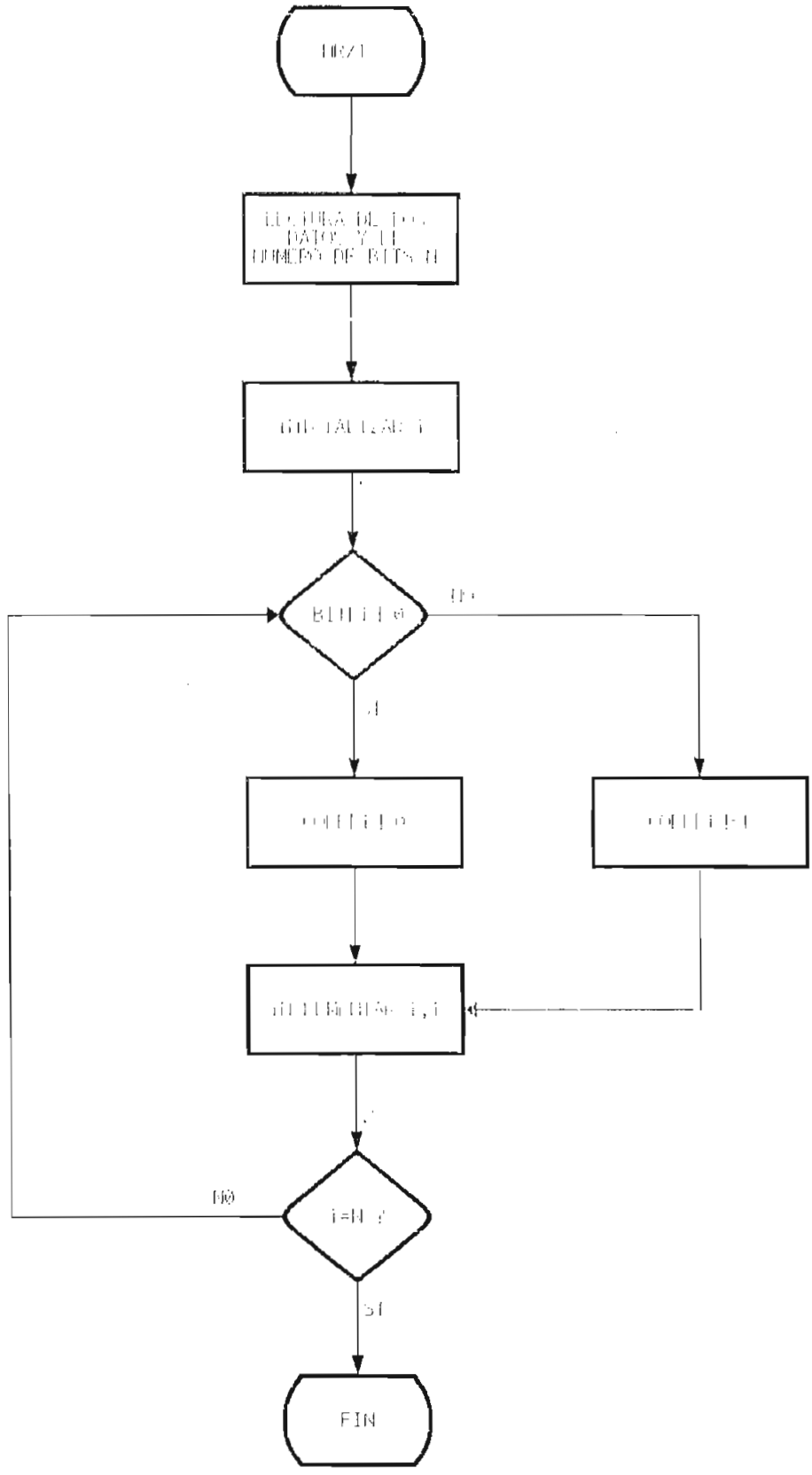


Figura 3.21. Diagrama de Flujo para el Código NRZ.

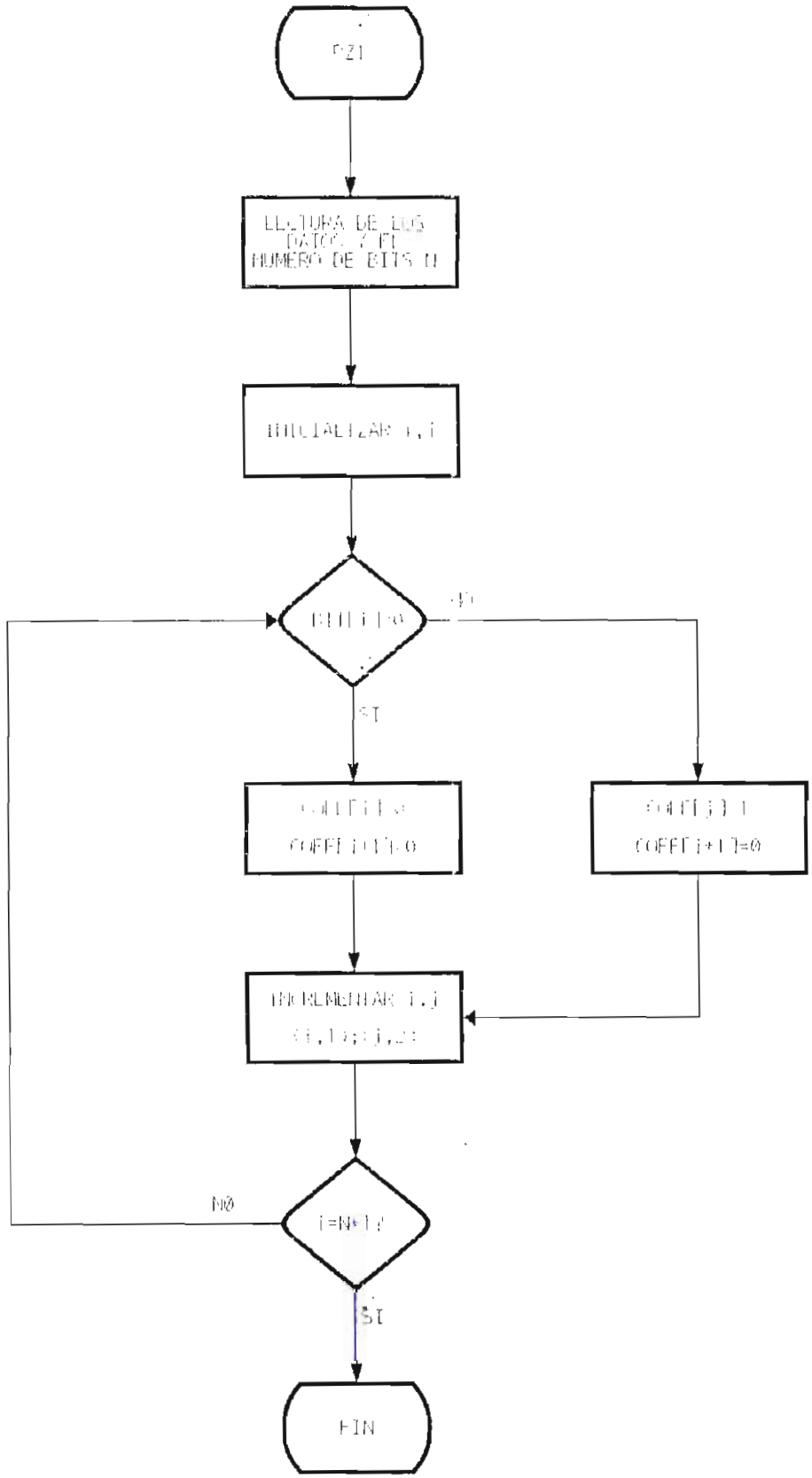


Figura 3.22. Diagrama de flujo para el Código RZ.

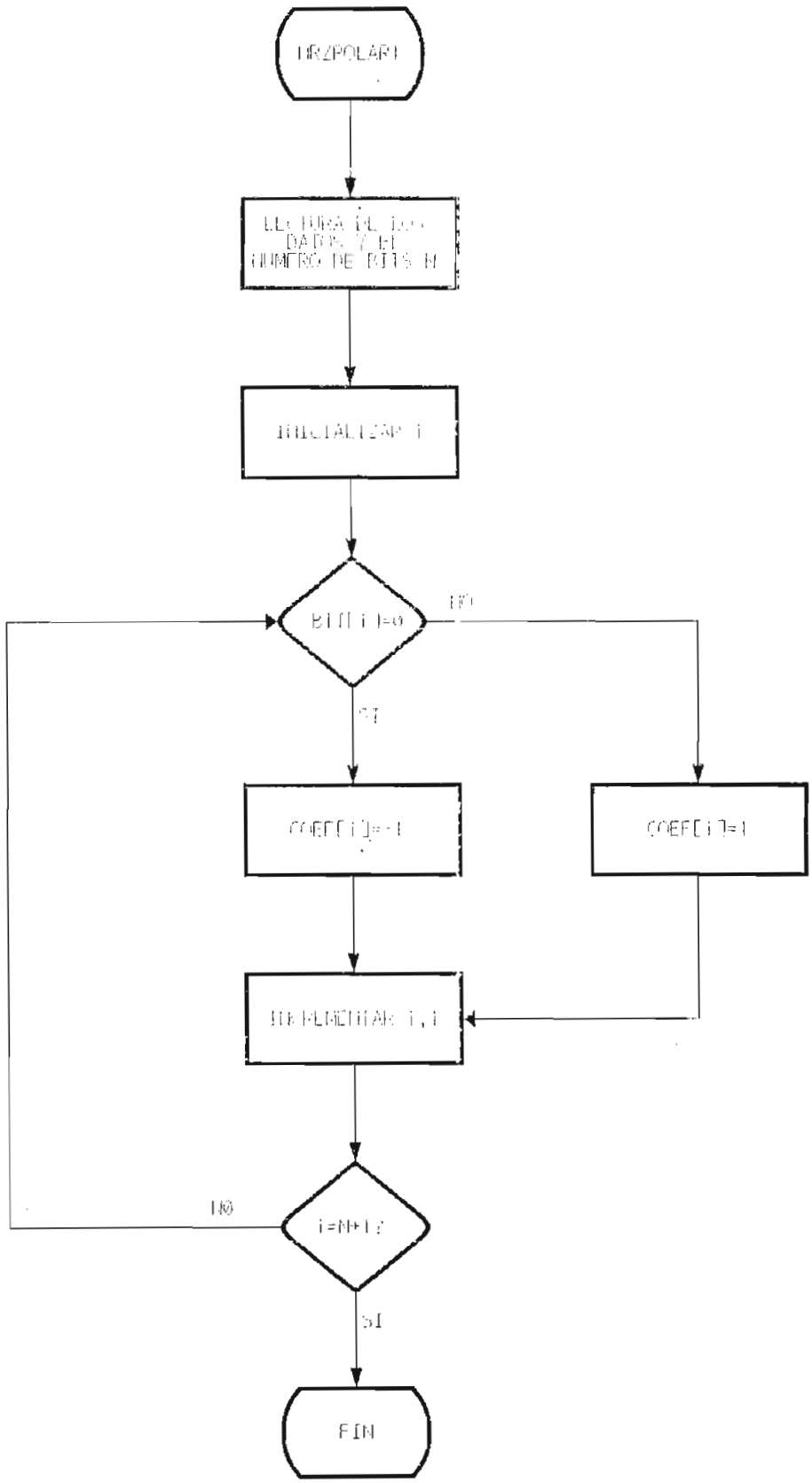


Figura 3.23. Diagrama de flujo para el Código NRZ-Polar.

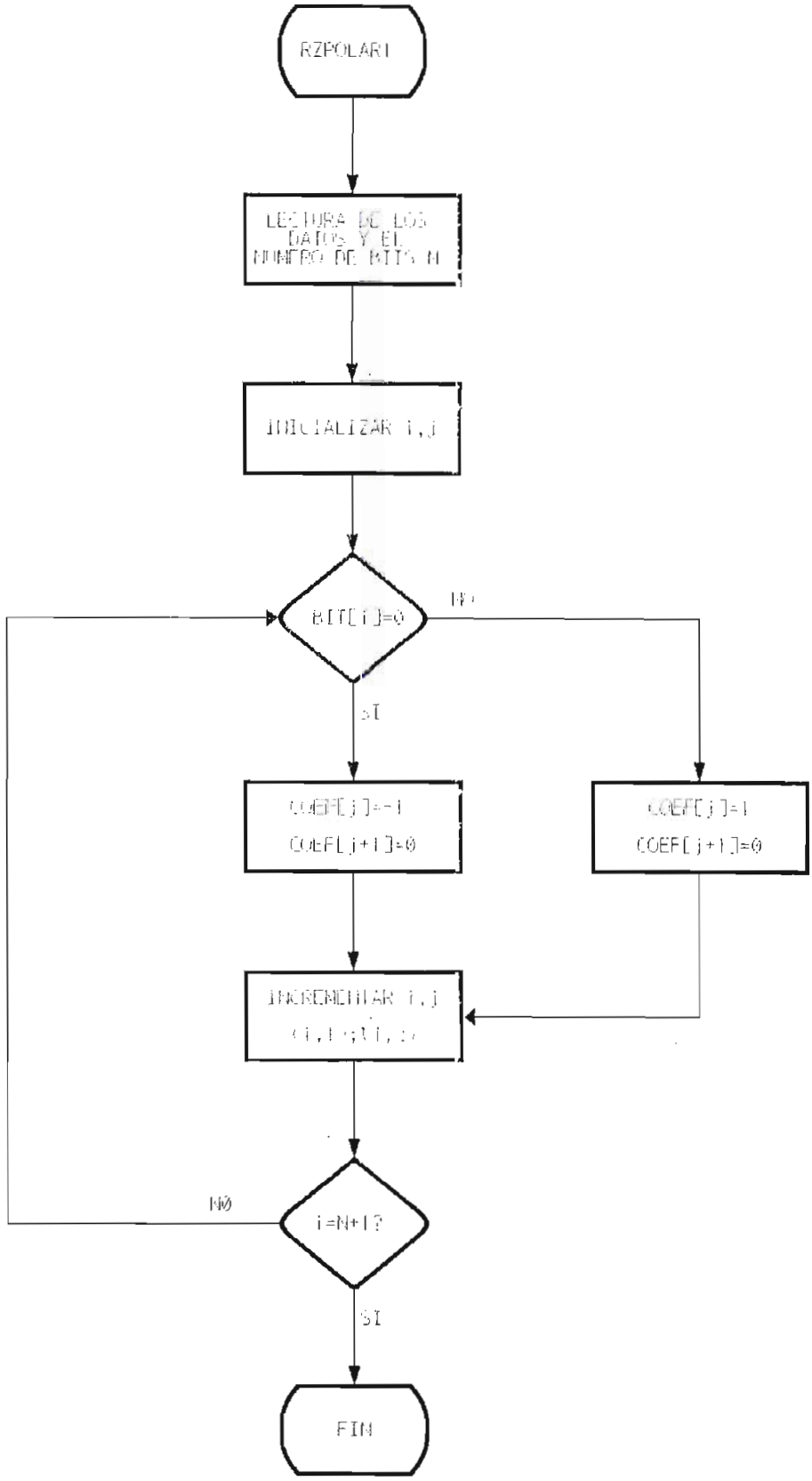


Figura 3.24. Diagrama de flujo para el Código RZ-Polar.

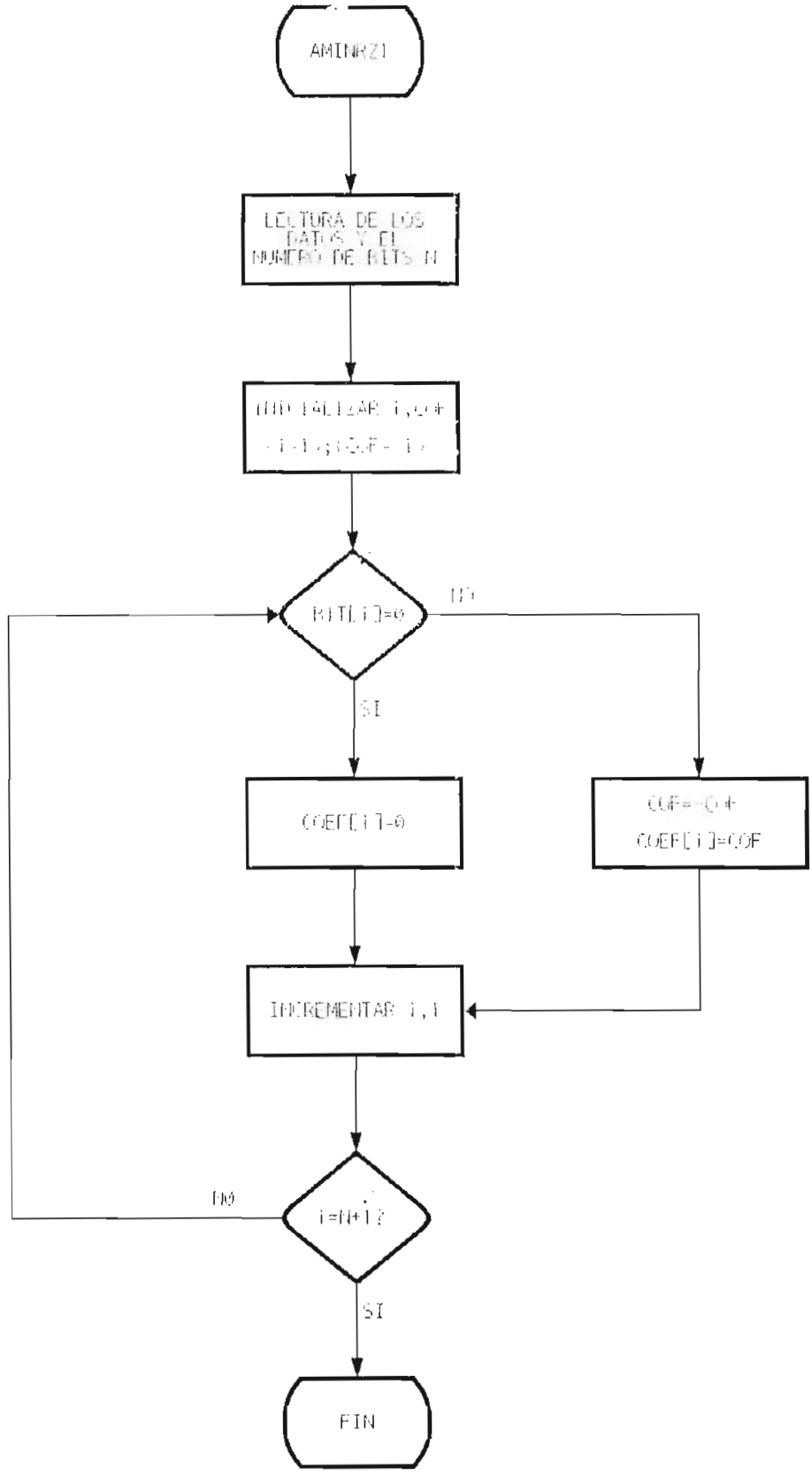


Figura 3.25. Diagrama de flujo para el Código AMI.

Si los **bit[i]=bit[i+1]=bit[i+2]=bit[i+3]**=0, además si el número de unos desde la última sustitución **k** es par y la polaridad del coeficiente anterior a la sustitución **coef[i-1]** es negativa,, los valores de los coeficientes serán:

coef[i] = 1,
coef[i+1] = 0,
coef[i+2] = 0,
coef[i+3] = 1, y **COF** = **coef[i+3]** = 1

Por otro lado si el número de unos desde la última sustitución **k** es impar y la polaridad del coeficiente anterior a la sustitución **coef[i-1]** es negativa,, los valores de los coeficientes serán:

coef[i] = 0,
coef[i+1] = 0,
coef[i+2] = 0,
coef[i+3] =-1, y **COF** = **coef[i+3]** =-1

Si lo anterior no ocurre, entonces, el procedimiento para obtener los coeficientes es de manera similar al algoritmo de código AMI. El diagrama de flujo del código HDB_3 se observa en la figura 3.26.

Si la primera secuencia de cuatro ceros consecutivos ocurre en los cuatro primeros bits de la secuencia, se asume que tendrá la codificación **0001**, por ser la más convencional, aunque se podría escoger cualquiera de las otras secuencias de codificación.

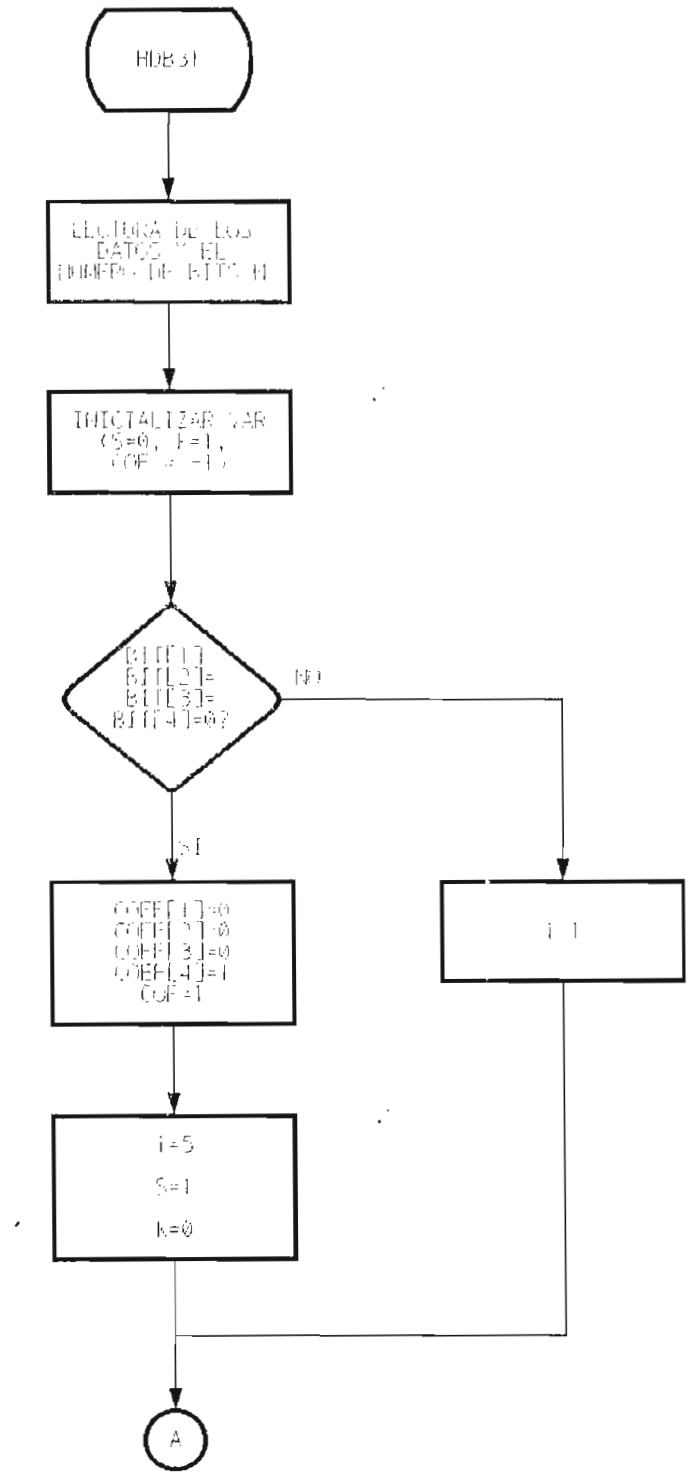


Figura 3.26. Diagrama de flujo para el Código HDB_3.

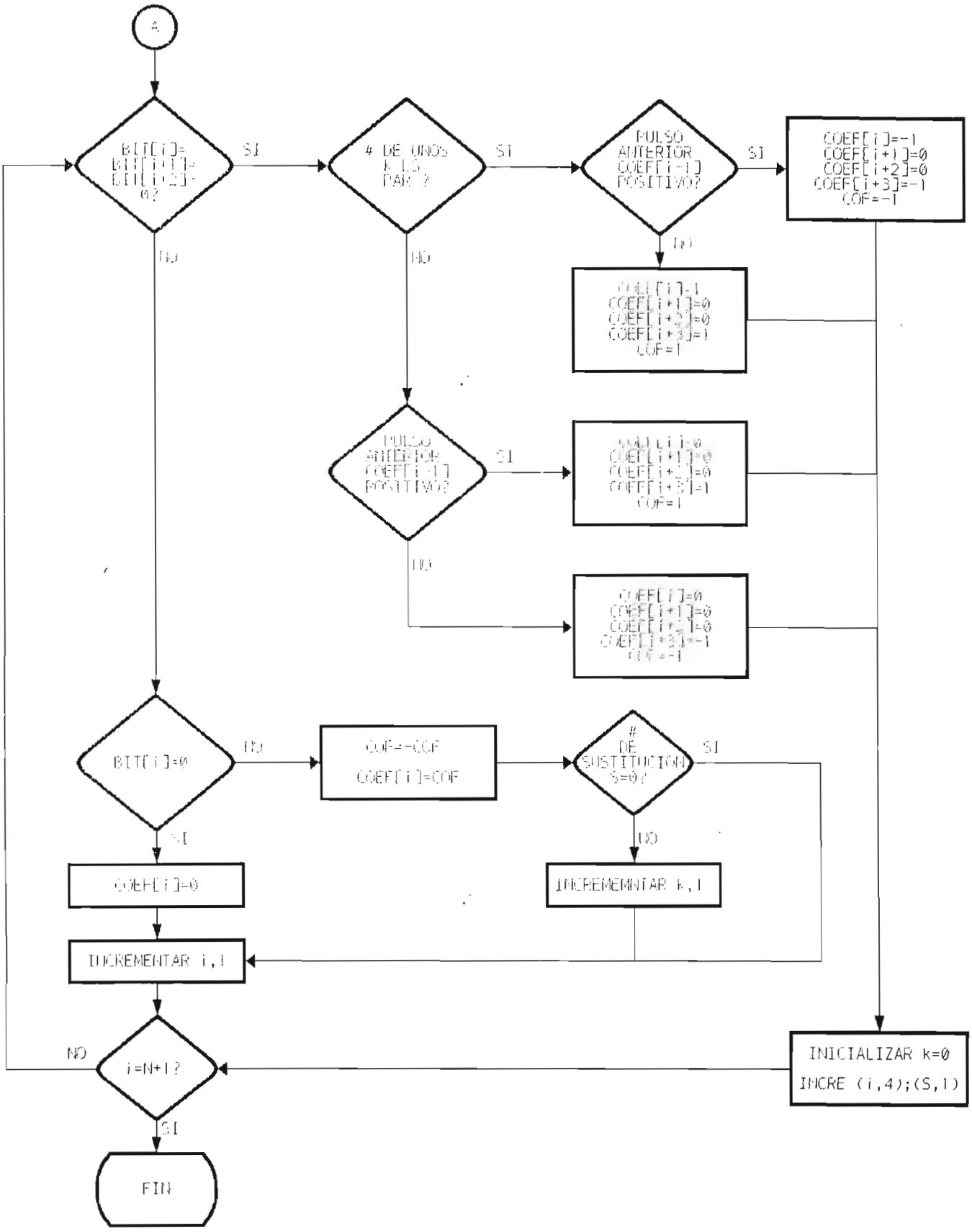


Figura 3.26. Diagrama de flujo para el Código HDB_3. (cont.)

3.5.7. Algoritmo para el código B3ZS

Para el código B3ZS, el cual no permite una secuencia de tres ceros consecutivos, se inicia la rutina detectando si a partir del bit que se lee, los siguientes dos bits son iguales a cero, esto es, si **bit[i]**, **bit[i+1]**, y **bit[i+2]** son iguales a cero. Si esto ocurre para los primeros tres bits de datos, se asume que los coeficientes correspondientes serán **0,0,1** por convención, caso contrario la forma de asignar los valores para los coeficientes se lo hace con la tabla de codificación B3ZS correspondiente, (tabla 3.3).

Si **bit[i]=bit[i+1]=bit[i+2]**=0, y se supone que el número de unos **k** desde la última sustitución es par y la polaridad del pulso anterior **coef[i-1]** es negativa, entonces los coeficientes tomarán los siguientes valores:

coef[i] = 1
coef[i+1] = 0
coef[i+2] = 1, y **COF** = **coef[i+2]** = 1

El valor del último coeficiente se asigna a la variable **COF**. Cuando no se tiene una secuencia de tres ceros consecutivos el procedimiento para encontrar los valores de los coeficientes es similar al que se utiliza en la rutina del código AMI, en el que se utiliza la variable **COF**, para conseguir la alternabilidad de los pulsos de codificación. En la figura 3.27 se representa el diagrama de flujo de este código.

3.5.8. Algoritmo para el código Bifase L o Manchester

En los códigos bifase, para elaborar su rutina de codificación se hace corresponder a cada bit de datos dos coeficientes así:

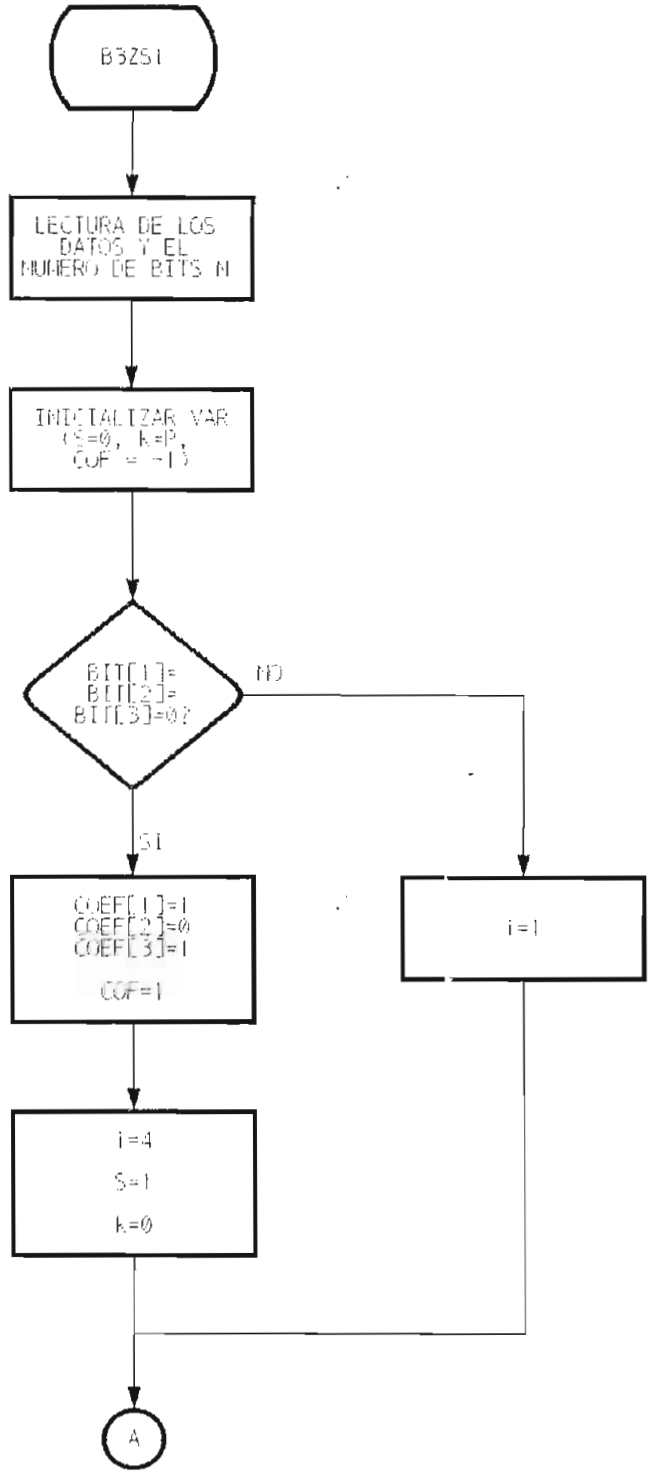


Figura 3.27 Diagrama de flujo para el Código B3ZS.

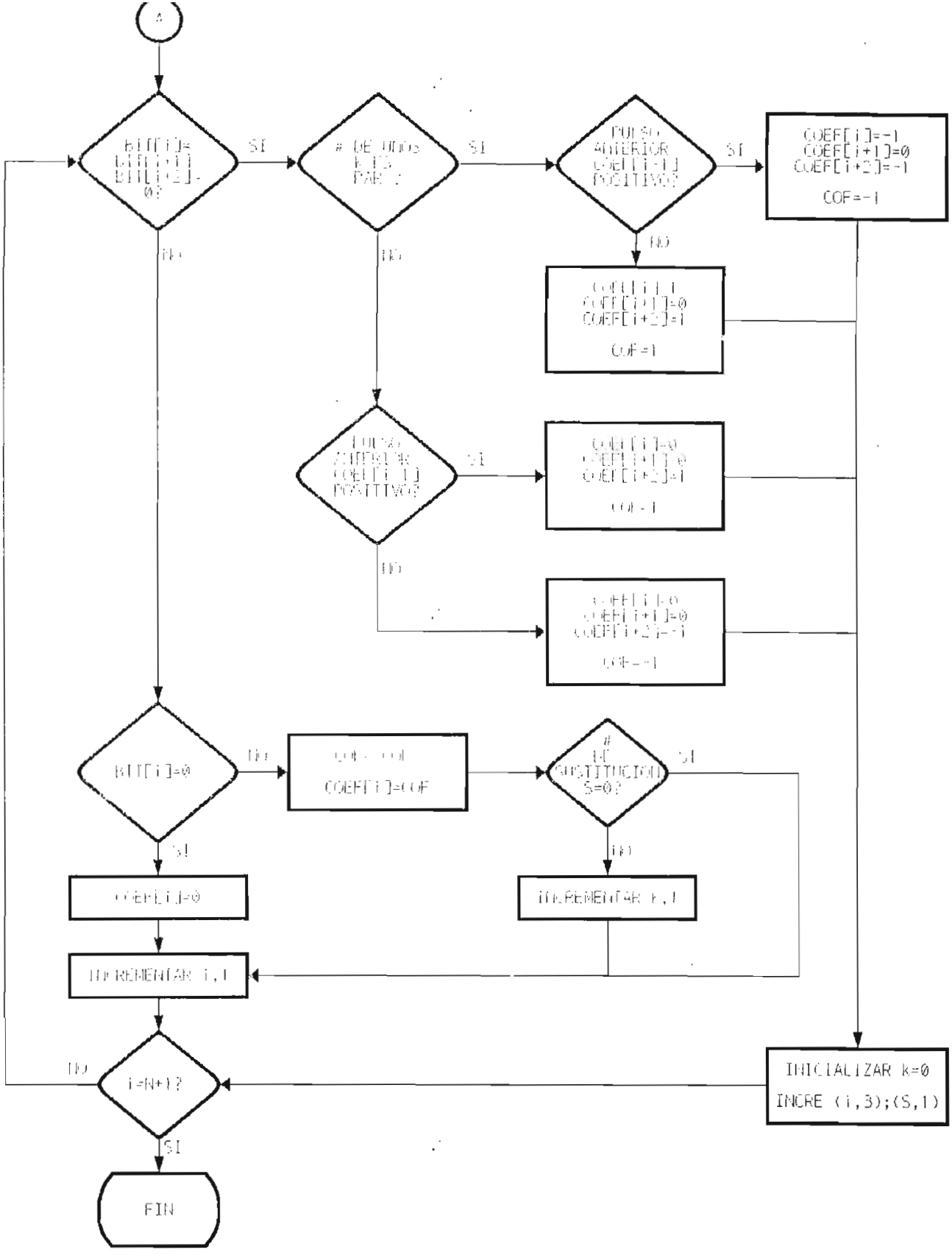


Figura 3.27. Diagrama de flujo para el CódigoB3ZS (cont.)

Para el **bit[i]** se tiene los coeficientes **coef[j]** y **coef[j+1]**.

La rutina de programación del código Manchester asigna los coeficientes de la siguiente forma:

Para **bit[i]**=0 se tiene **coef[j]**=-1; **coef[j+1]**=1, lo que corresponde a una transición positiva a mitad del periodo T_o. Si **bit[i]**=1, **coef[j]**=1; **coef[j+1]**=-1 que corresponde a una transición negativa a mitad del periodo T_o. El diagrama de flujo de este código se muestra en la figura 3.28.

3.5.9. Algoritmo para el código Bifase M

Para la rutina de programación del código Bifase M los valores de los coeficientes a partir de los bits de entrada se los obtiene de la siguiente manera:

Cuando **bit[i]**=1 se tiene: **coef[j]=-coef[j-1]**;
coef[j+1]=coef[j-1]

Como **coef[j]=-coef[j-1]**, existe una transición al inicio del intervalo T_o y con **coef[j+1]=coef[j-1]** se tiene otra transición con sentido inverso a la anterior a mitad del respectivo intervalo T_o.

Para el caso en que **bit[i]**=0, se tiene que los coeficientes toman los valores **coef[j]=coef[j+1]=-coef[j-1]**, esto es, se produce una transición al inicio del periodo T_o, y el estado resultante se mantiene hasta el final del período, es decir, que para 0_L no se tiene transición a mitad del intervalo. El diagrama de flujo del código Bifase M se lo representa en la figura 3.29.

3.5.10. Algoritmo para el código Bifase S

La forma de codificación para el código Bifase S es contraria al código Bifase M, por lo tanto el algoritmo para

obtener los coeficientes **coef** de este código es opuesto al anterior, con lo que se tiene:

Para **bit[i]**=0 los coeficientes son **coef[j]=-coef[j-1]**, **coef[j+1]=coef[j-1]**, en este caso para 0_L se tiene una transición al inicio del intervalo T_o y otra a la mitad del intervalo T_o.

Si **bit[i]**=1, los coeficientes son: **coef[j]=coef[j+1]=-coef[j-1]**, con lo que se tiene para 1_L solamente transición al inicio del periodo T_o. Esto se puede apreciar en mejor forma en el diagrama de flujo de la figura 3.30.

3.5.11. Algoritmo para el código de Miller

El algoritmo para este código se lo realiza como se indica a continuación:

Cuando el bit de datos es 1_L se tiene una transición a mitad del periodo, esto se consigue con **coef[j] = coef[j-1]** y **coef[j+1]= -coef[j-1]**.

Cuando el bit de datos es **0** no se tiene transición, esto es, **coef[j]=coef[j+1]=coef[j-1]**; pero si el bit anterior también es igual a **0**, se tiene transición y los valores de los coeficientes son **coef[j]=coef[j+1]=-coef[j-1]**.

El diagrama de flujo para este código se observa en la figura 3.31.

3.5.12. Algoritmo para el código 4B3T

El programa para obtener los coeficientes en el código 4B3T es más complejo y necesita de otros subprogramas para obtener los resultados, conforme a las reglas de codificación 4B3T y del diagrama de estados correspondiente; el procedimiento se desarrolla de la siguiente manera:

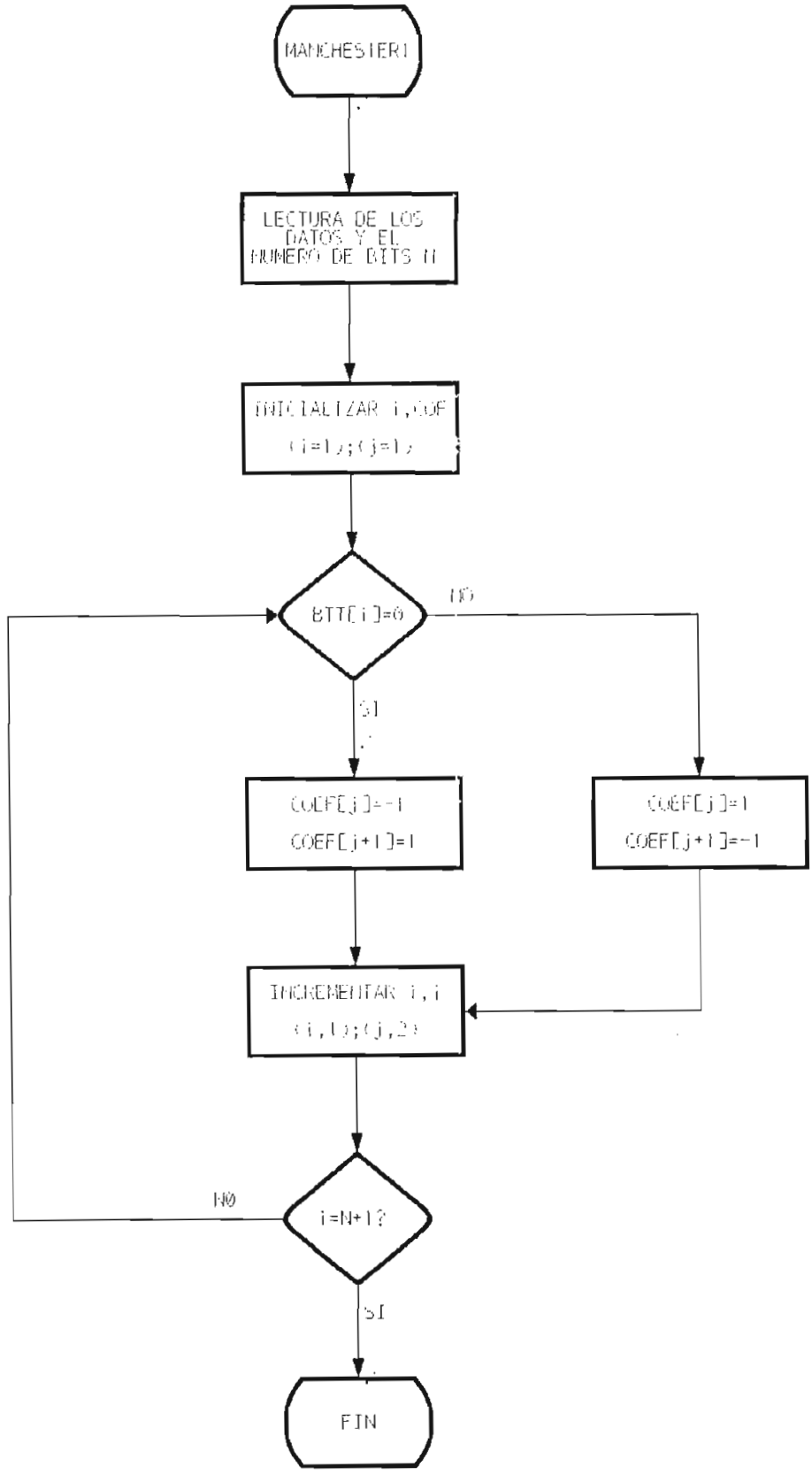


Figura 3.28. Diagrama de flujo para el Código Manchester.

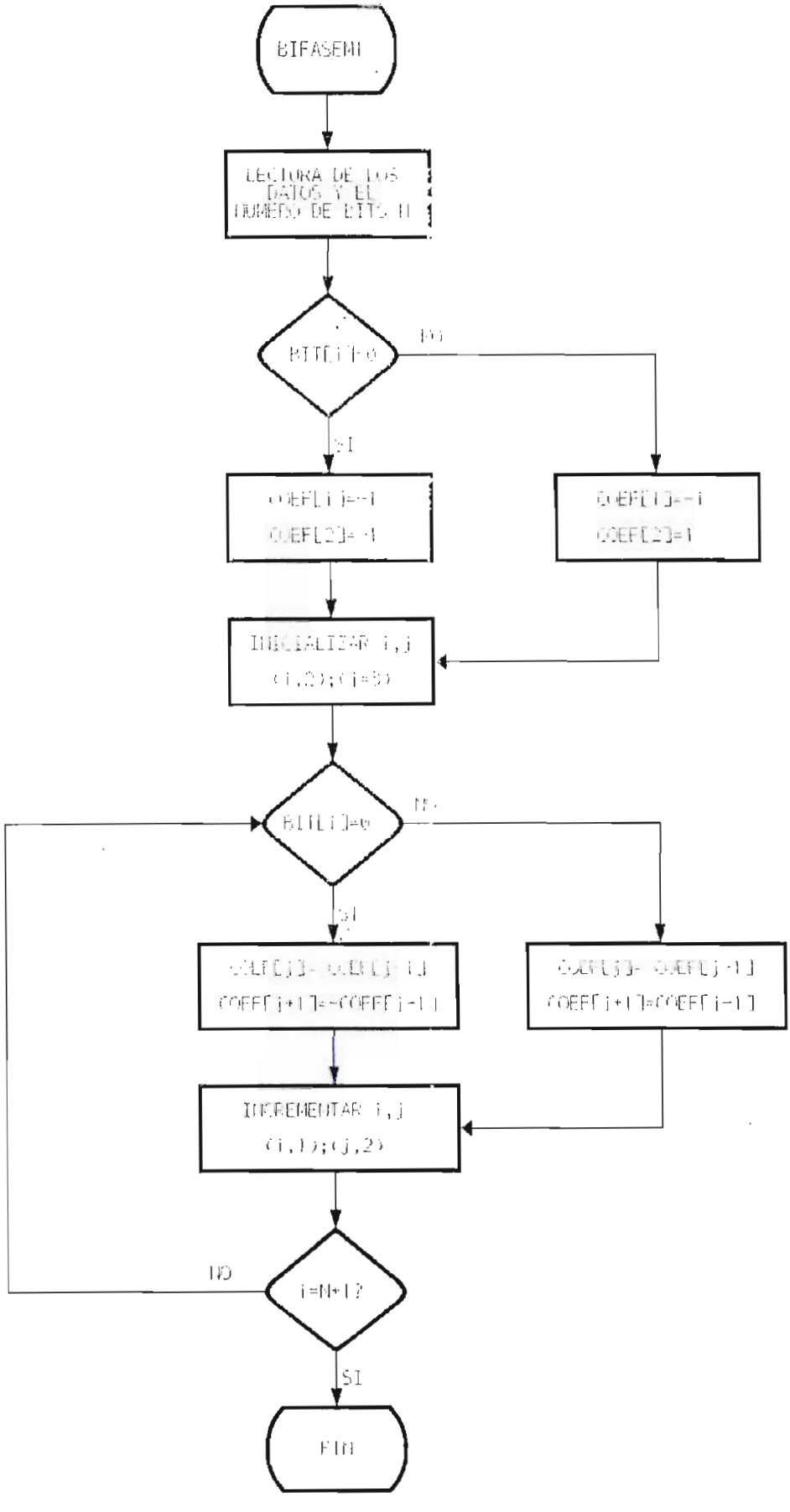


Figura 3.29. Diagrama de flujo para el Código Bifase M.

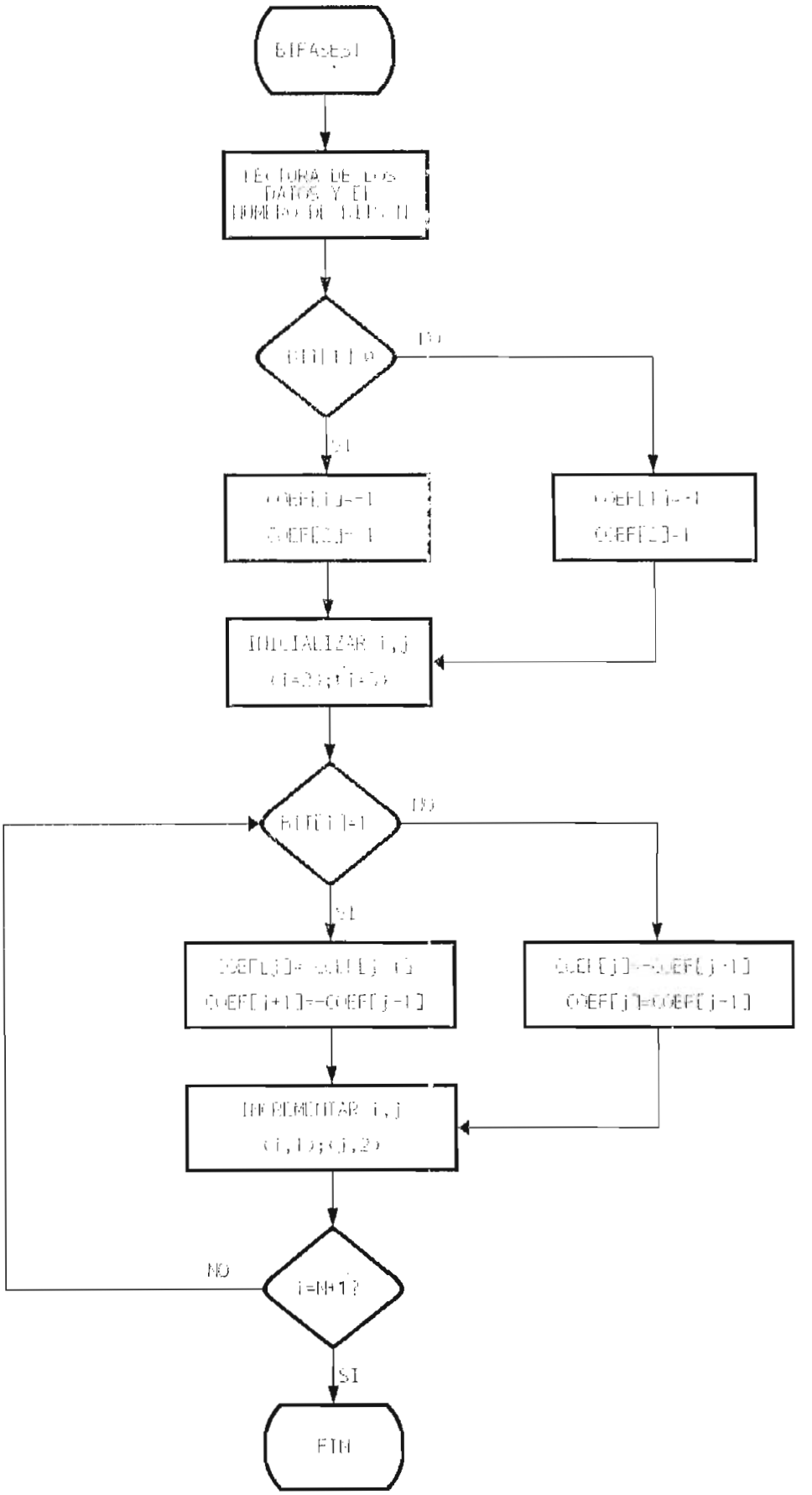


Figura 3.30. Diagrama de flujo para el Código Bifase S.

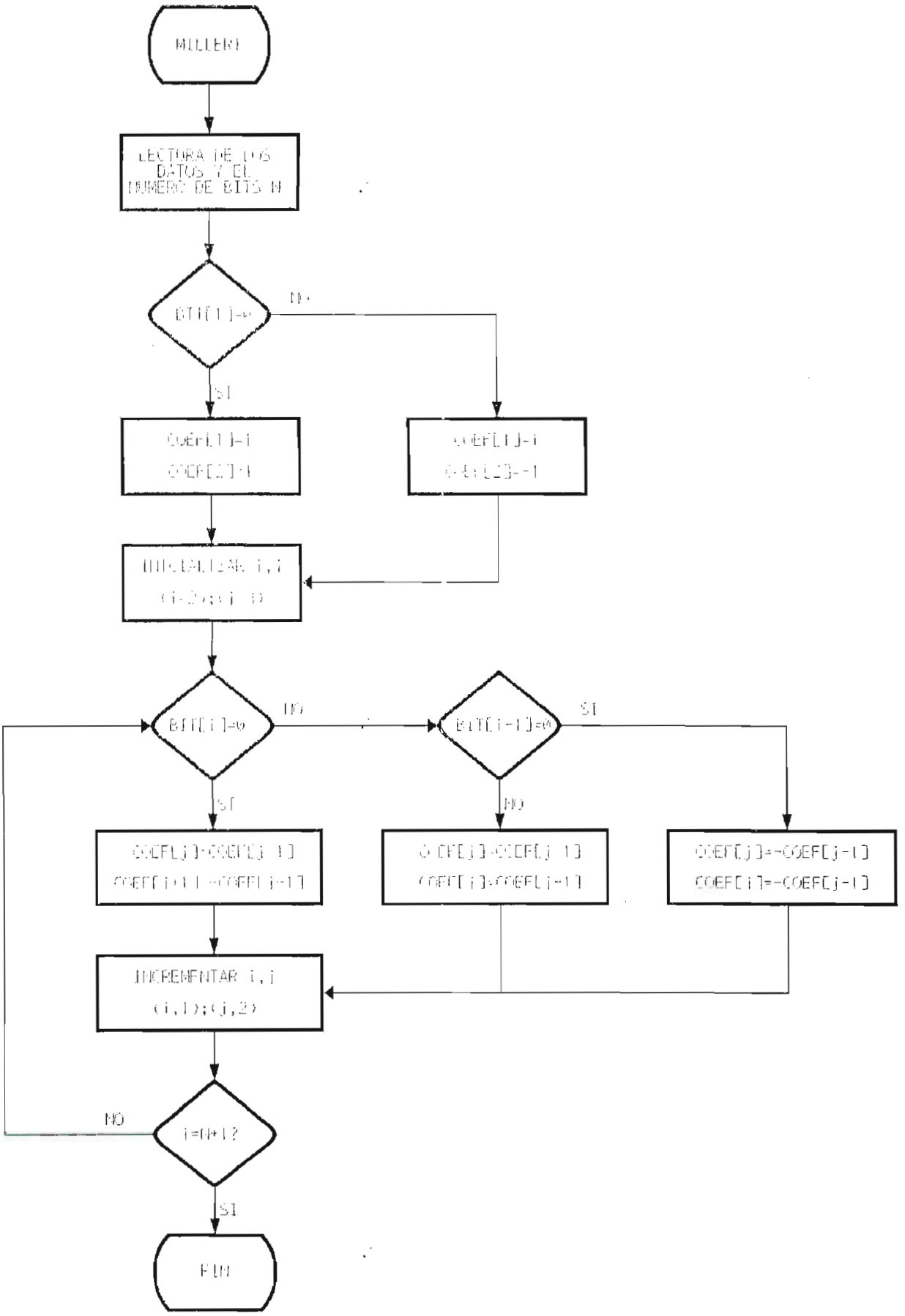


Figura 3.31. Diagrama de flujo para el Código de Miller

Se leen los bits de datos y se agrupan en palabras de 4 bits cada una, se lee además la polaridad inicial que es una variable que se ingresa como dato y que puede tener valores enteros desde -3 a +3.

A cada palabra de cuatro bits le corresponde una polaridad acumulada de acuerdo con la tabla de codificación (tabla 3.5), a partir de la polaridad inicial y la polaridad acumulada se realiza el cambio de estado, conforme al diagrama de estados de la codificación 4B3T.

Si la polaridad acumulada es igual a cero le corresponde el **ModoCero** para la palabra ternaria. Si la polaridad acumulada es positiva, entonces para la codificación se escoge el **ModoNegativo**, caso contrario se escoge el **ModoPositivo** y se realiza el correspondiente cambio de estado.

Las subrutinas o procedimientos que se utilizan son las siguientes:

- **ParidadAcumulada**, este procedimiento o subrutina agrupa los bits en palabras de cuatro bits cada una y determina la polaridad acumulada que le corresponde.

- Los procedimientos **ModoCero**, **ModoPositivo** y **ModoNegativo** asignan la palabra ternaria para cada palabra de cuatro bits; éstos son los coeficientes que se necesitan para la representación gráfica de la señal codificada.

- El procedimiento **CambioEstado** comprende los procedimientos **Uno**, **Dos**, **Tres**, **MenosUno**, **MenosDos** y **MenosTres**, los mismos que son utilizados para realizar el cambio de estado dependiendo de los valores de polaridad inicial y polaridad acumulada.

El diagrama de flujo del algoritmo para el código 4B3T se observa en la figura 3.32.

3.5.13. Algoritmo para el código CMI

Para obtener los coeficientes del código CMI se procede como sigue:

En el caso de que el bit de datos sea igual a **0** se lo representa por una transición positiva a mitad del periodo T_0 por lo tanto para **bit[i]**=0, los coeficientes son **coef[j]**=-1 y **coef[j+1]**=1. Si el bit de datos es igual a **1** entonces la forma de codificación es similar al algoritmo del código AMI, en la que interviene la variable **COF** que permite tener la alternabilidad de los pulsos. La figura 3.33 representa el diagrama de flujo del código de inversión de marcas CMI.

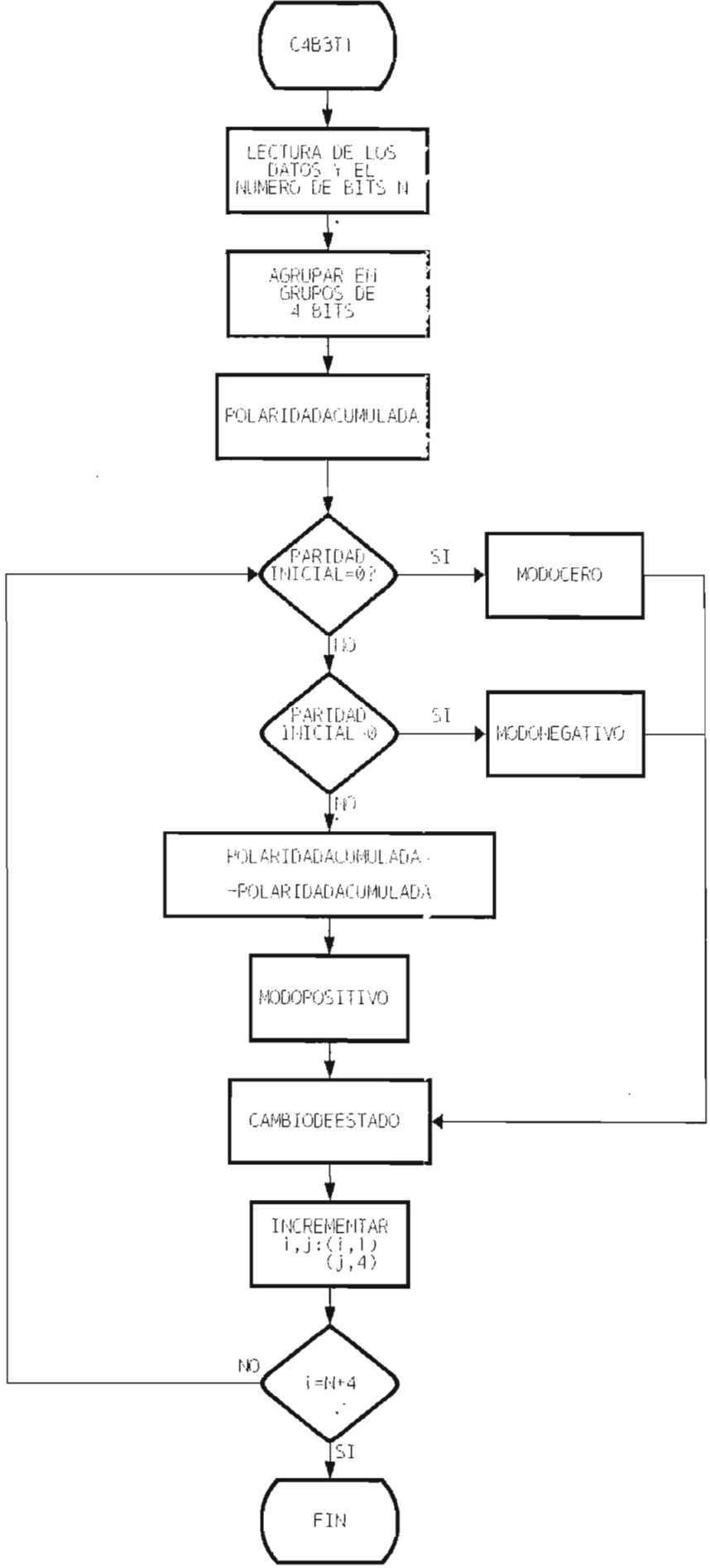
3.5.14. Algoritmo para el código PST

En el código PST la secuencia binaria se debe transformar en un código de dos dígitos ternarios, es decir, para realizar el algoritmo primeramente se agrupan los bits de dos en dos y la forma de escoger los coeficientes será de acuerdo con la tabla de codificación PST (tabla 3.7). Por ejemplo para las secuencias de bits **00** y **11** los coeficientes asignados serán:

bit[i]=0 y **bit[i+1]**=0 corresponde a **coef[i]**=-1, **coef[i+1]**=1
bit[i]=1 y **bit[i+1]**=1 corresponde a **coef[i]**=1, **coef[i+1]**=-1,
para estas secuencias los coeficientes son constantes.

Las secuencias de bits **10** y **01** en cambio varían de acuerdo con el modo inicial escogido **P**, positivo o negativo. Así por ejemplo, para la secuencia **10**, si se tiene un modo inicial positivo se tendrá:

Si **bit[i]**=1 y **bit[i+1]**=0, corresponde a **coef[i]**=1 y **coef[i+1]**=0, si aparece nuevamente esta secuencia le corresponderá entonces **coef[i]**=-1 y **coef[i+1]**=0 y así alternadamente, igual ocurre para la secuencia **01**. El diagrama de flujo para el código PST se muestra en la figura 3.34.



3.32. Diagrama de flujo para el Código 4B3T.

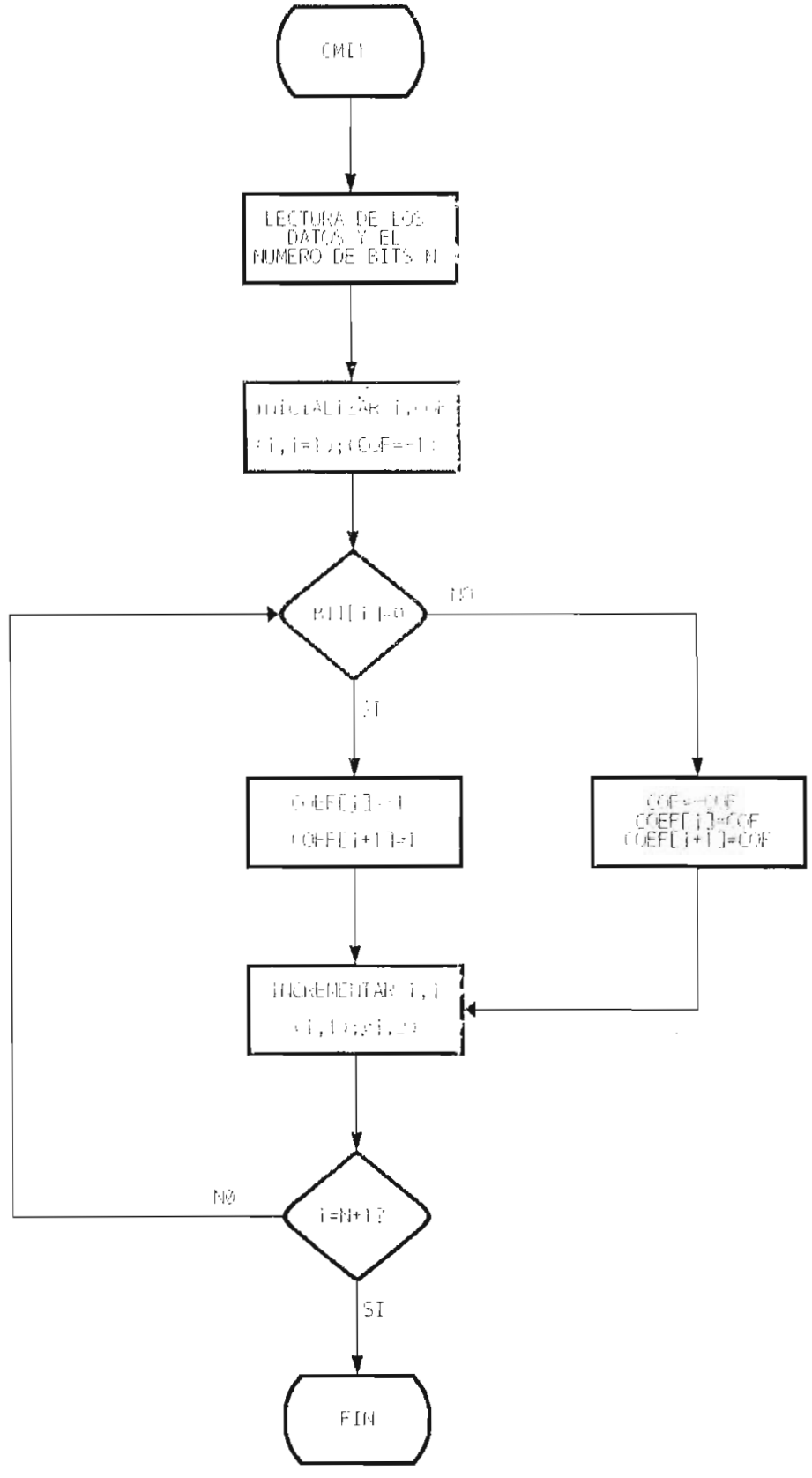


Figura 3.33. Diagrama de flujo para el Código CMI.

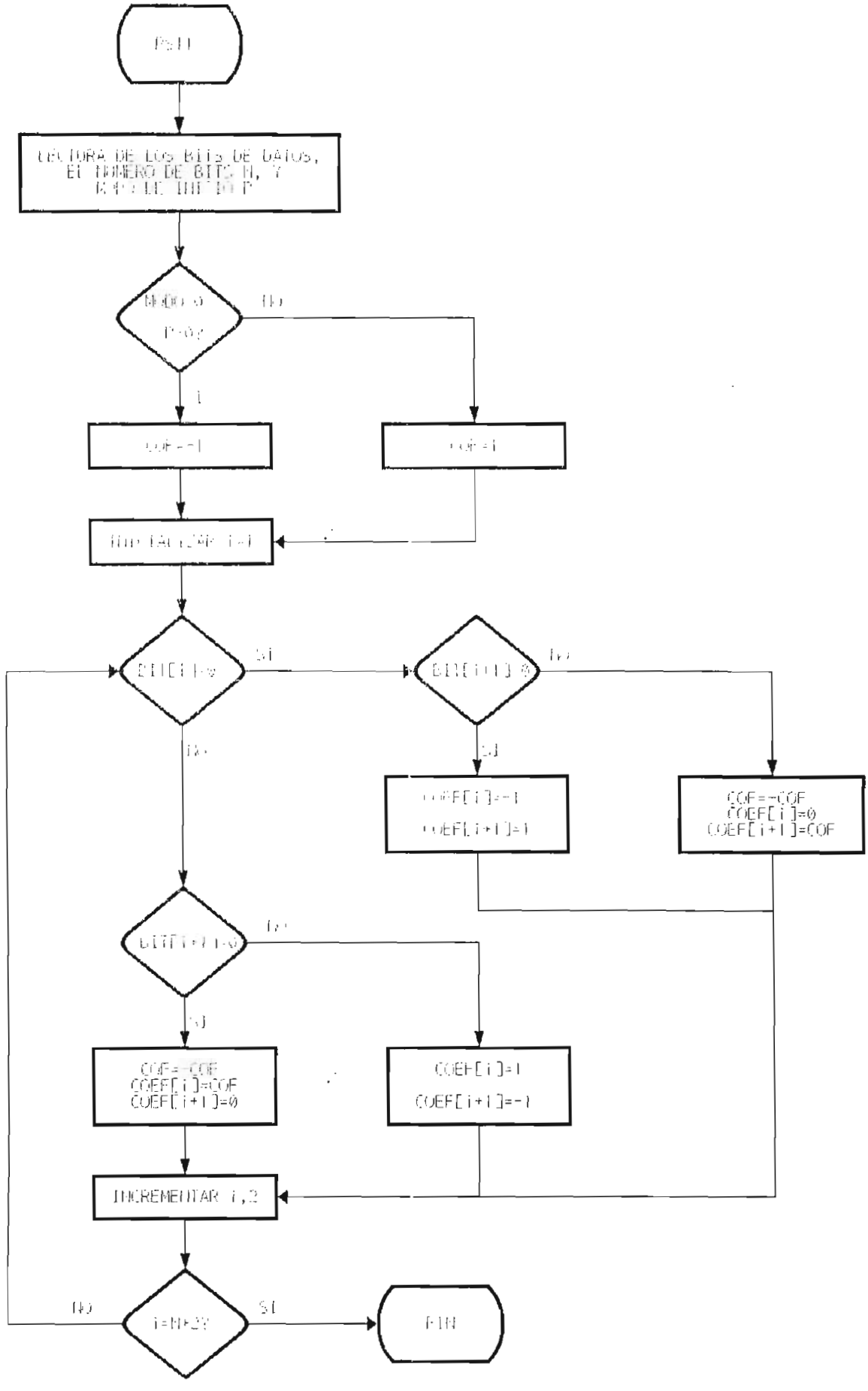


Figura 3.34. Diagrama de flujo para el Código PST.

3.6. EJEMPLOS DE APLICACION DE LA SIMULACION DE CODIGOS DE LINEA

El programa de simulación implementado en base de los algoritmos analizados en el numeral anterior, se ejecutó y probó para todos los códigos, aquí se indican parte de ellos como ejemplos de aplicación.

La secuencia en que se muestran los ejemplos está conforme al orden presentado en la teoría y al desarrollo de los algoritmos de los procedimientos o subrutinas.

Los ejemplos siguientes permiten ver los resultados de los mismos. En los gráficos de los ejemplos se observan, en la parte superior la señal de reloj, que es fundamental en transmisión digital, a continuación está representada la señal de datos elegida para codificar, en la parte inferior se encuentra la señal codificada de acuerdo a la clase de código elegido.

Se debe indicar que para los códigos en los que se debe elegir una polaridad inicial para el proceso de codificación esto es, para los códigos B3ZS, 4B3T y PST, se representan ejemplos para cada una de las polaridades permitidas en cada caso. Por ejemplo el código B3ZS tiene dos posibles polaridades par o impar, lo que no sucede con el código 4B3T en el que la polaridad inicial a elegir está entre -3 y +3.

La secuencia **1 0 1 0 0 0 0 1 1 1 0 0 0 0 1 1** elegida, al codificarla permite observar que se cumplen las reglas de codificación para todos los códigos, incluso en los casos de los códigos B3ZS y 4B3T en los que se necesitan tener una o más secuencias de tres o cuatro ceros consecutivos para aplicarlas.

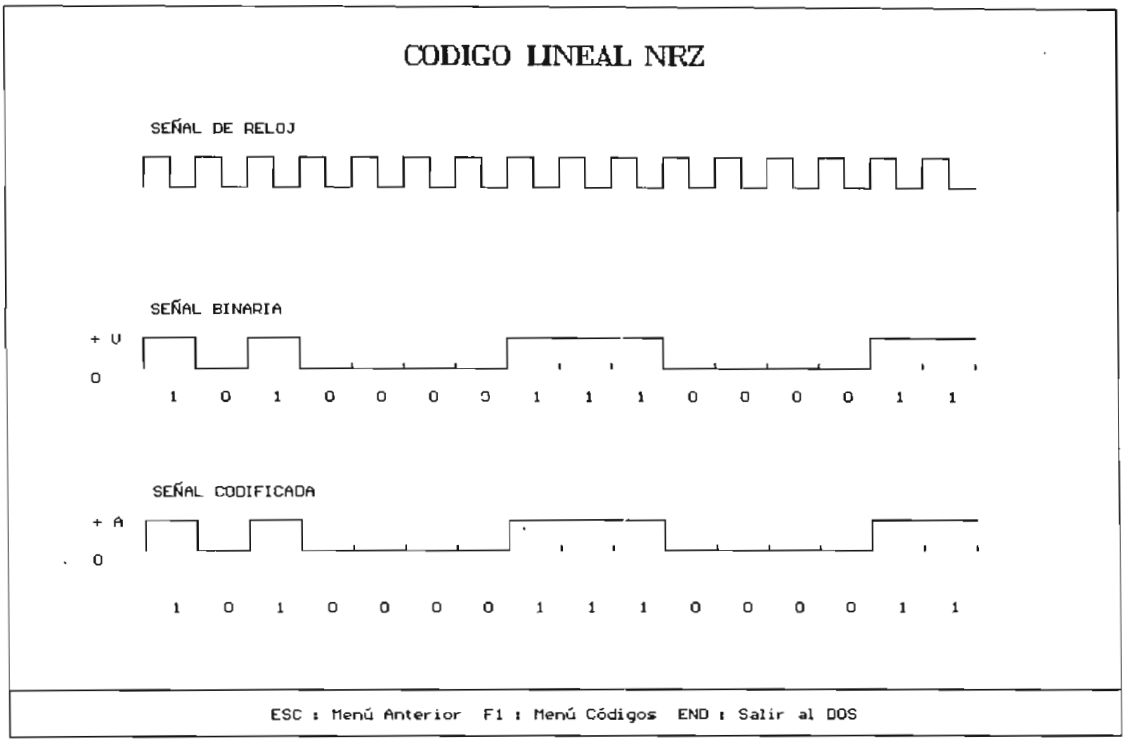


Figura 3.35. Ejemplo para el Código NRZ

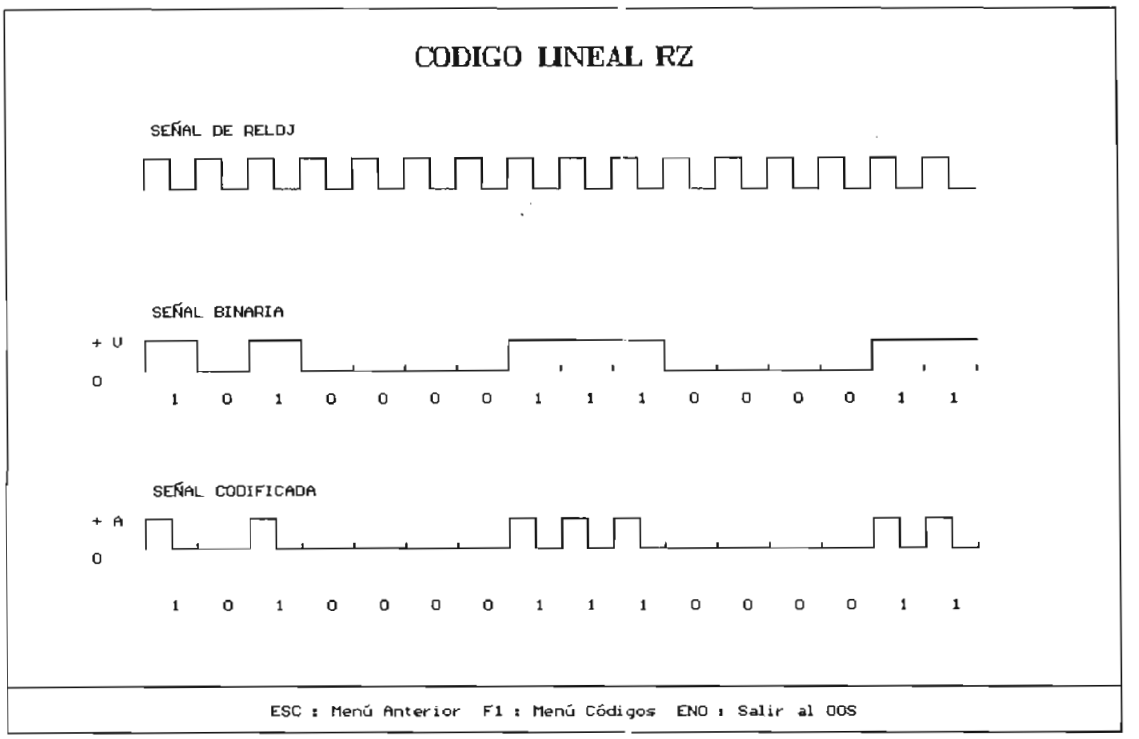


Figura 3.36. Ejemplo para el Código RZ

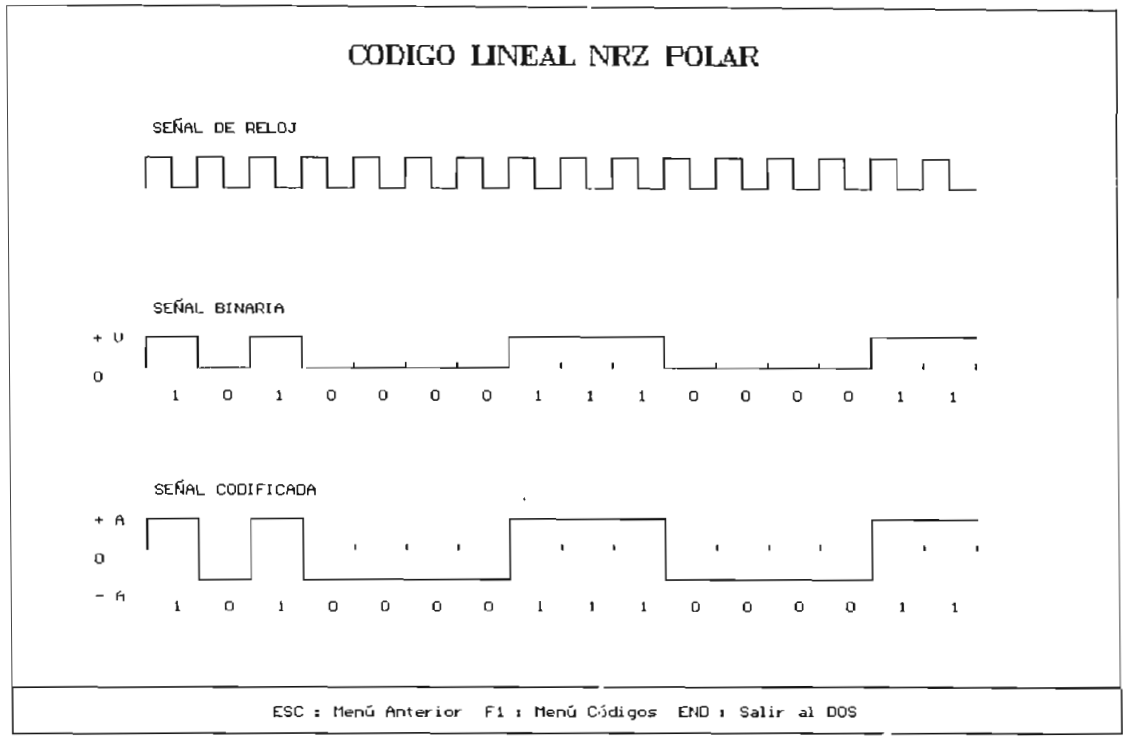


Figura 3.37. Ejemplo para el Código NRZ polar

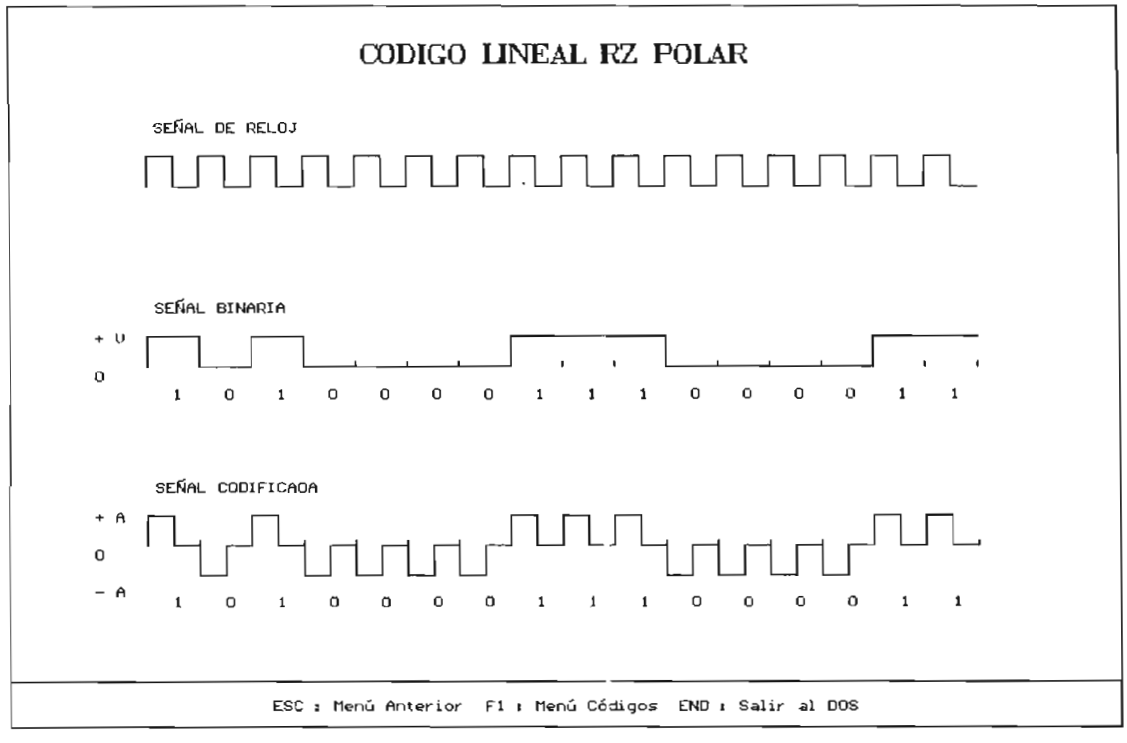


Figura 3.38. Ejemplo para el Código RZ polar

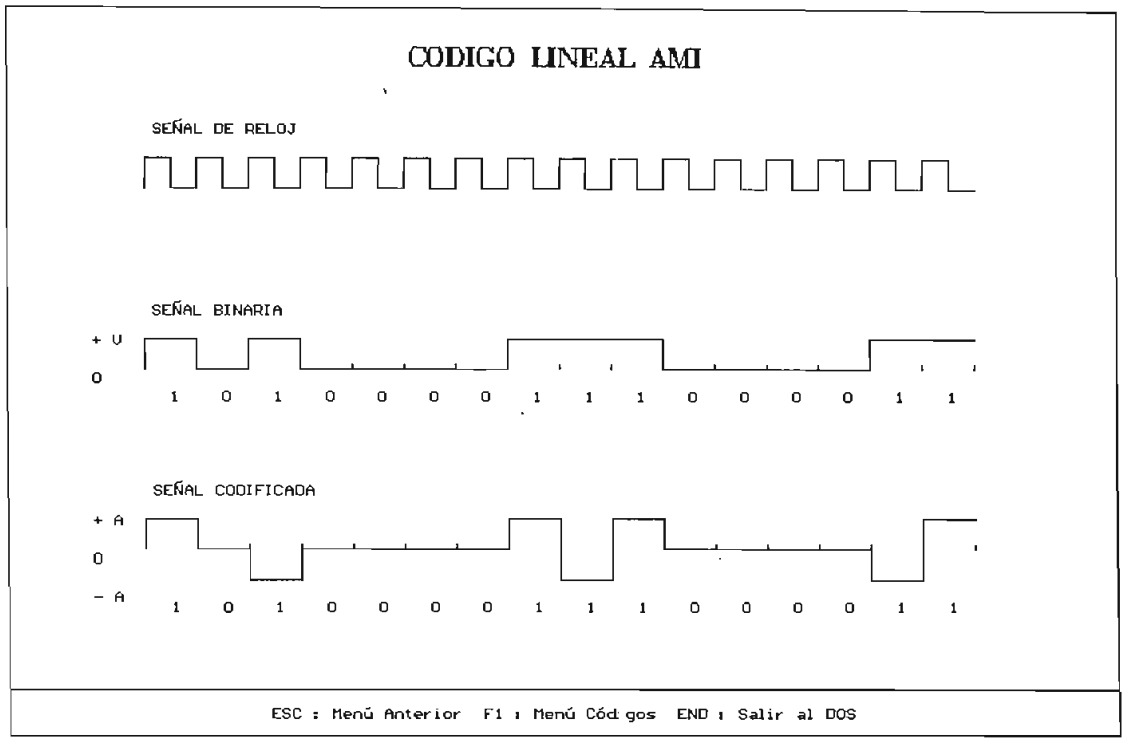


Figura 3.39. Ejemplo para el Código AMI-NRZ

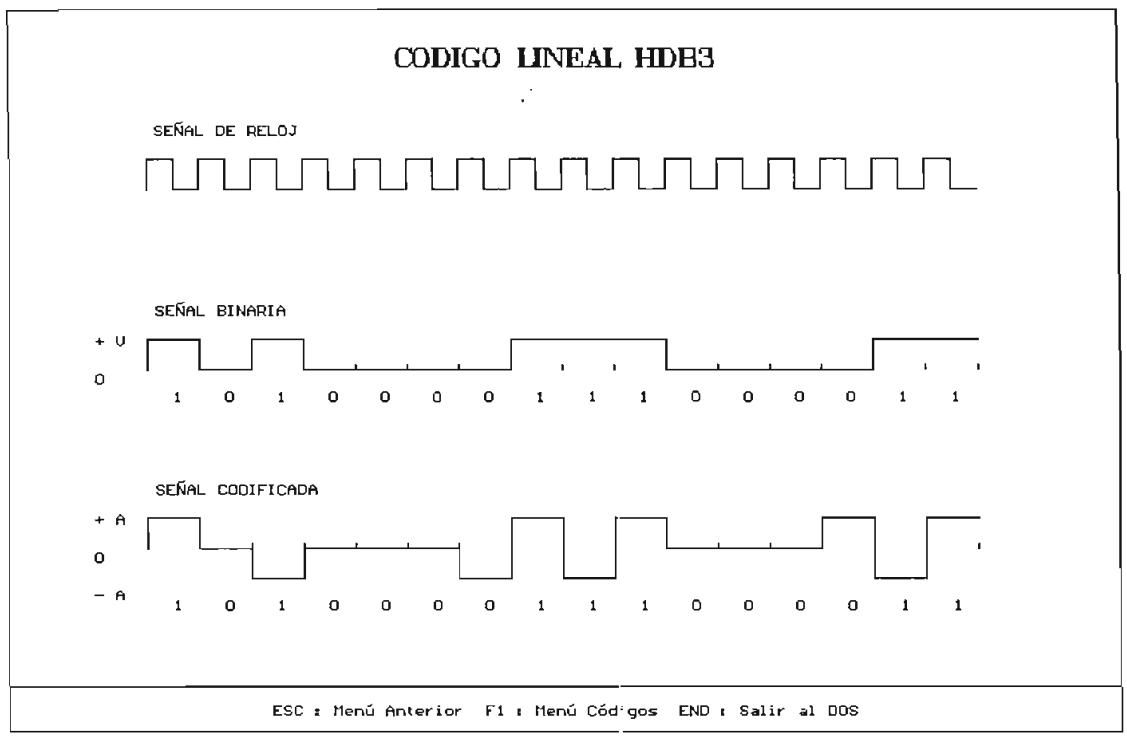


Figura 3.40. Ejemplo para el Código HDB_3

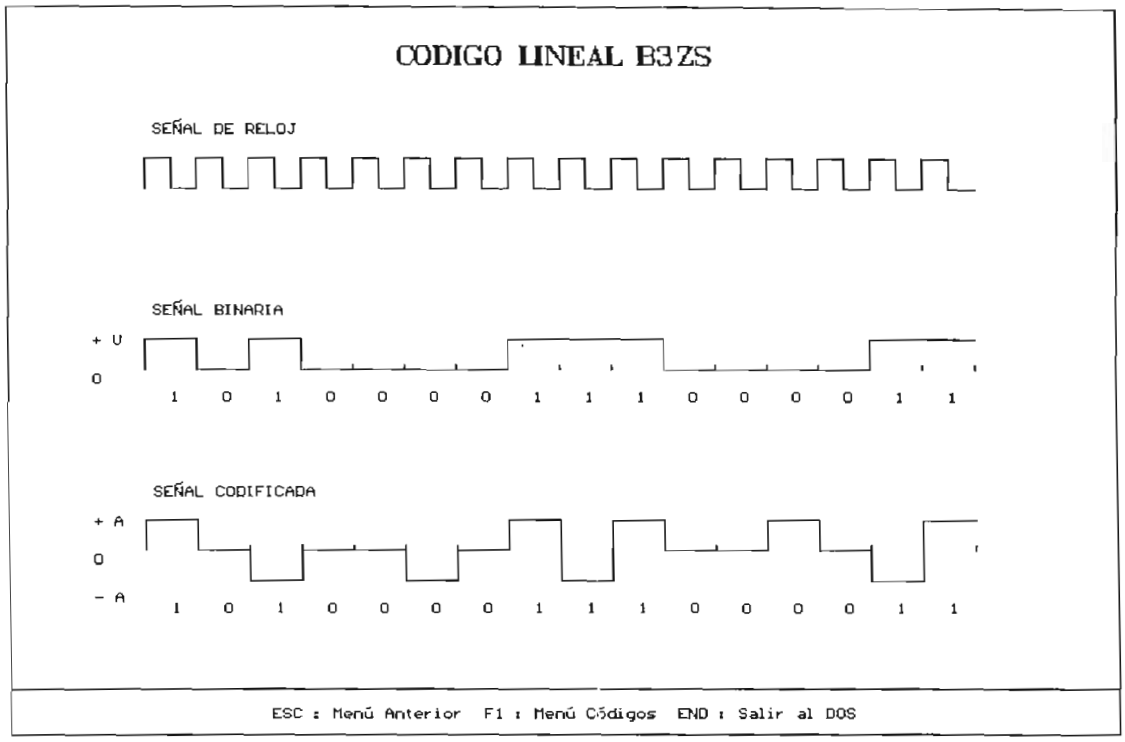


Figura 3.41. Ejemplo para el Código B3ZS
Polaridad par P = 0

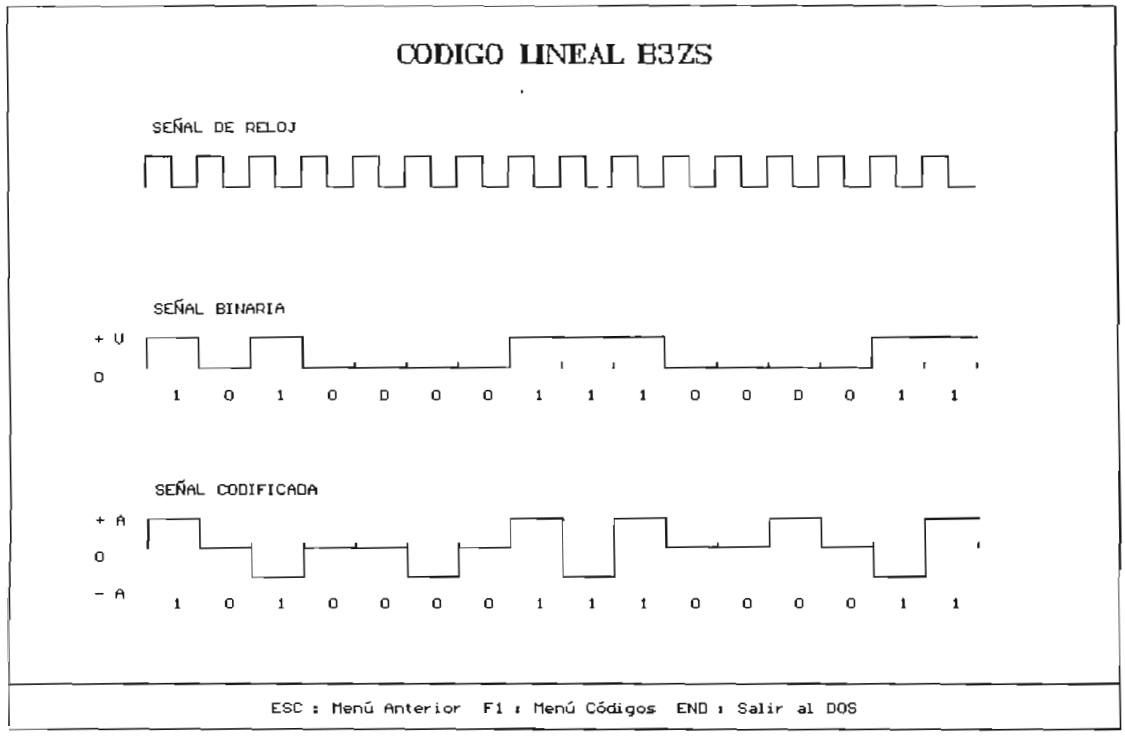


Figura 3.42. Ejemplo para el Código B_3ZS
Polaridad impar P = 1

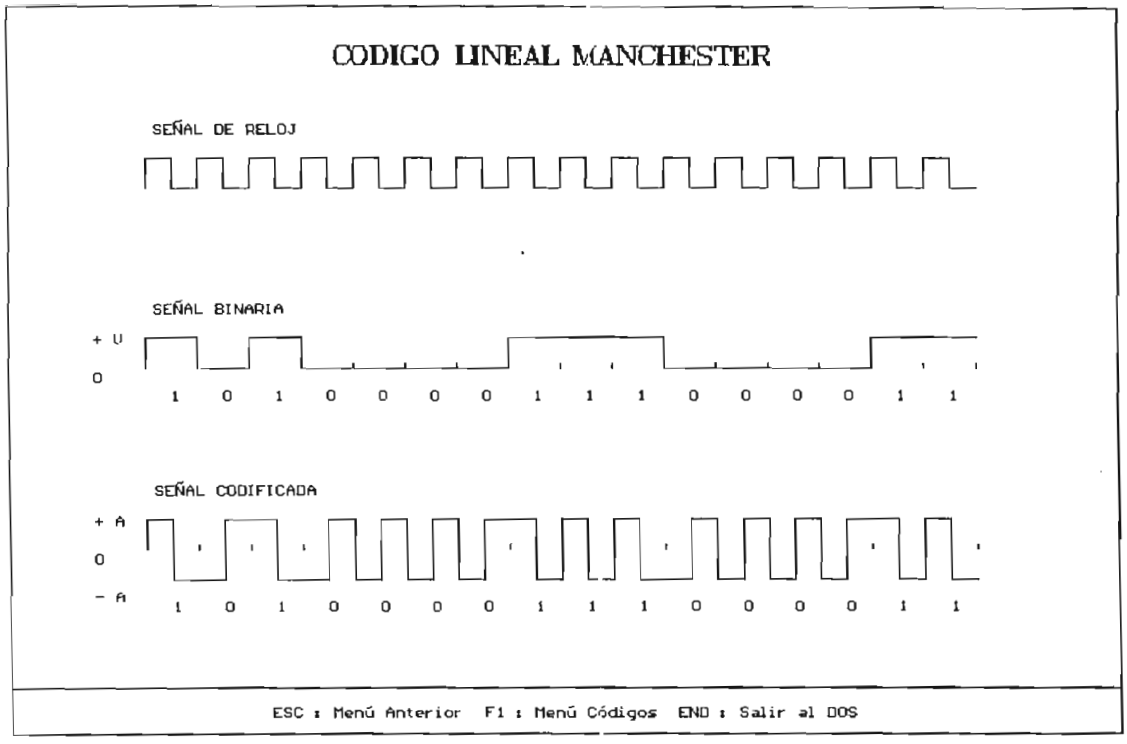


Figura 3.43. Ejemplo para el Código Bifase L o Manchester

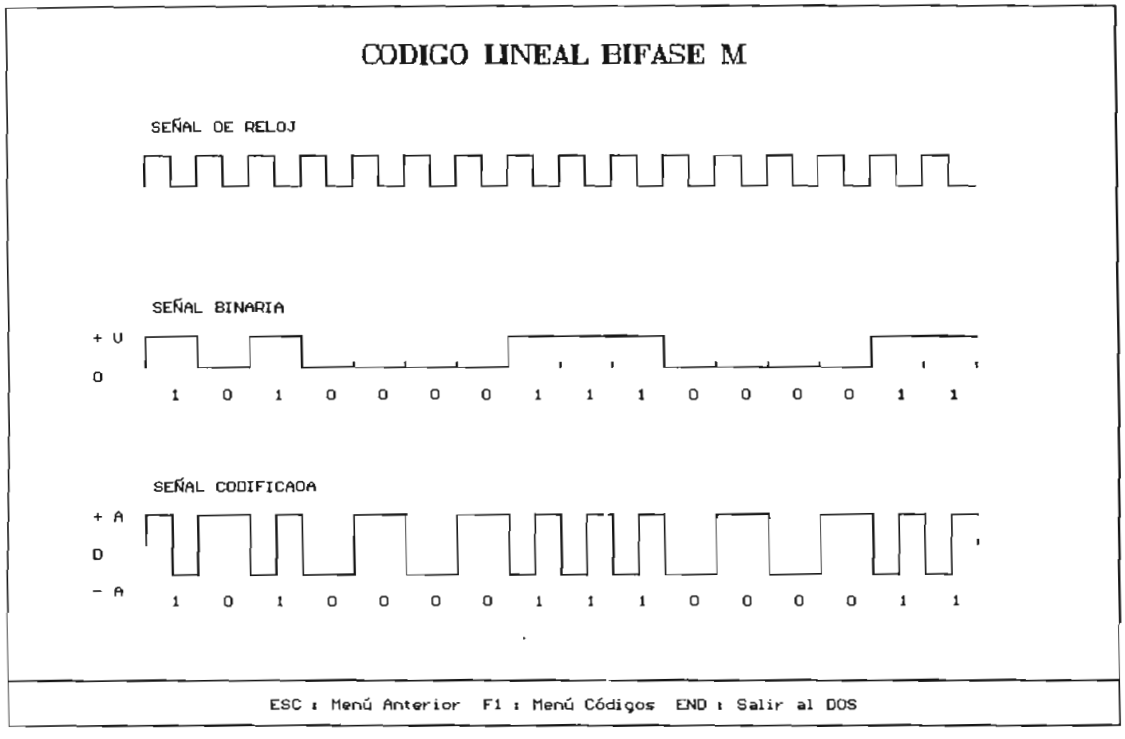


Figura 3.44. Ejemplo para el Código Bifase M

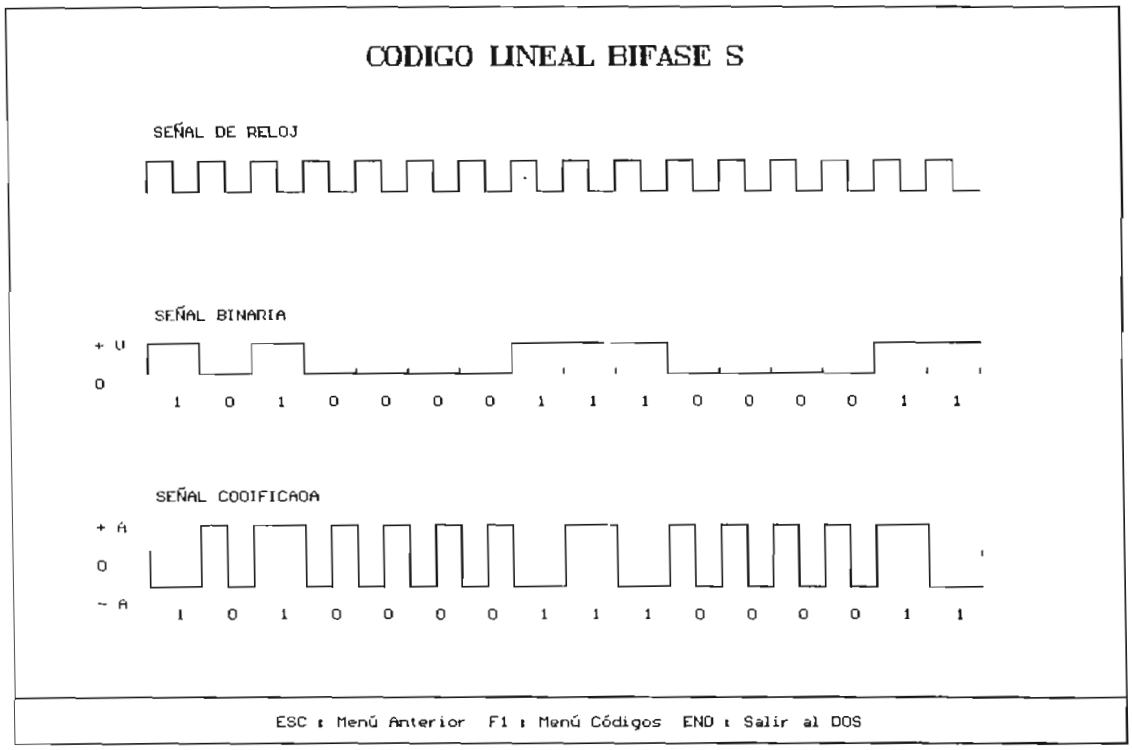


Figura 3.45. Ejemplo para el Código Bifase S

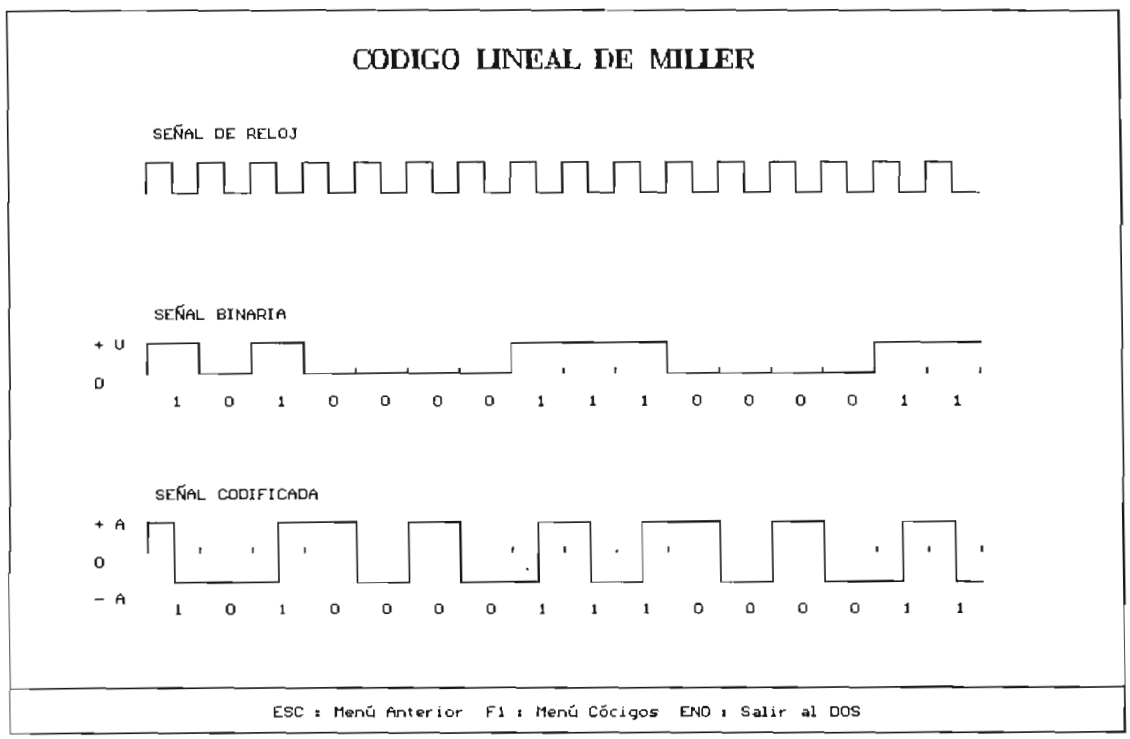


Figura 3.46. Ejemplo para el Código de Miller

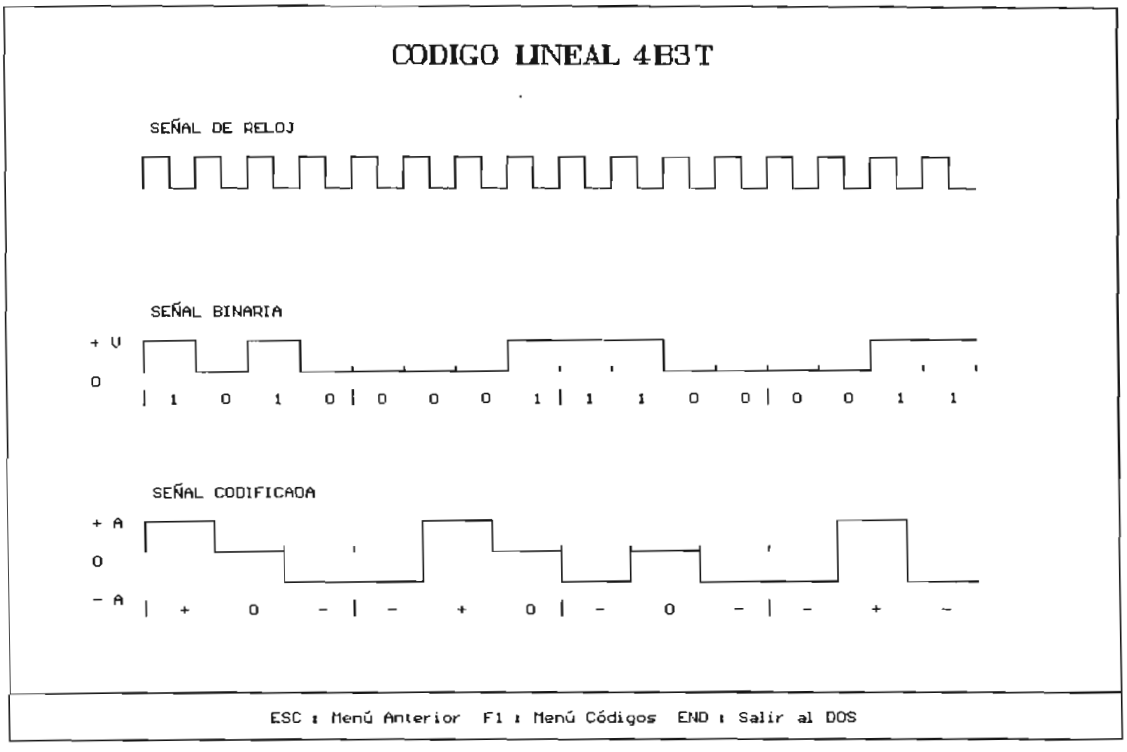


Figura 3.47. Ejemplo para el Código 4B3T
Polaridad P = -3

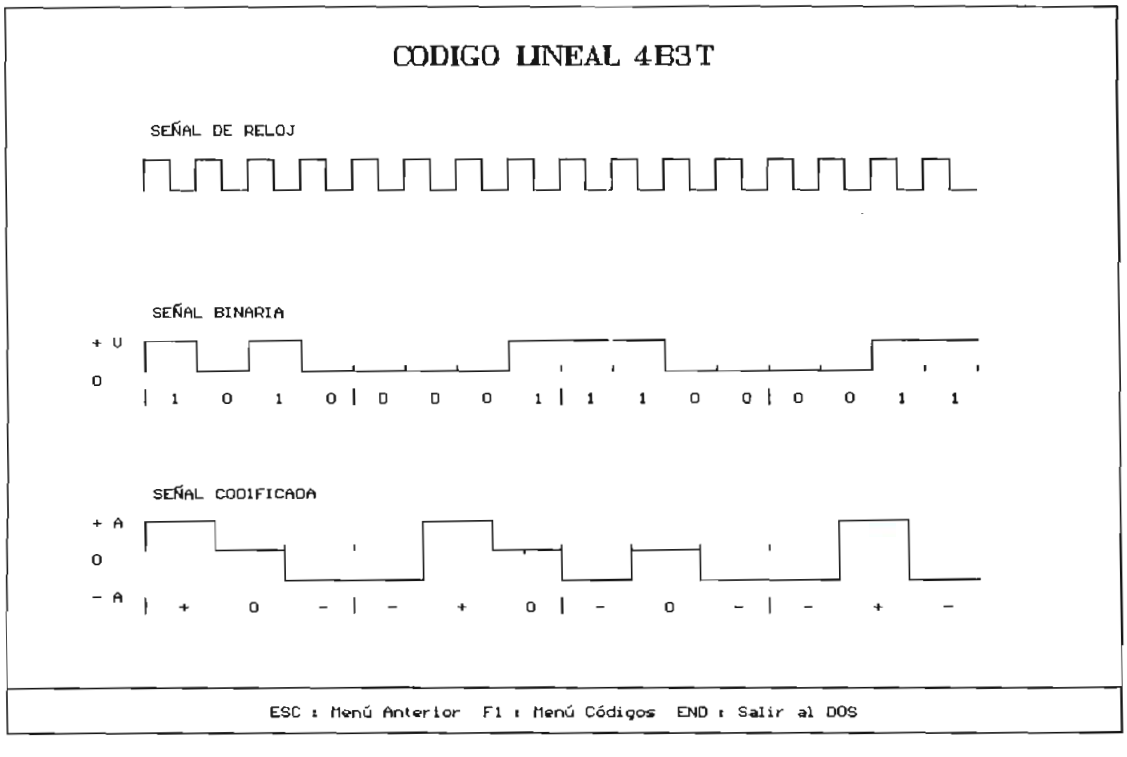


Figura 3.48. Ejemplo para el Código 4B3T
Polaridad P = -2

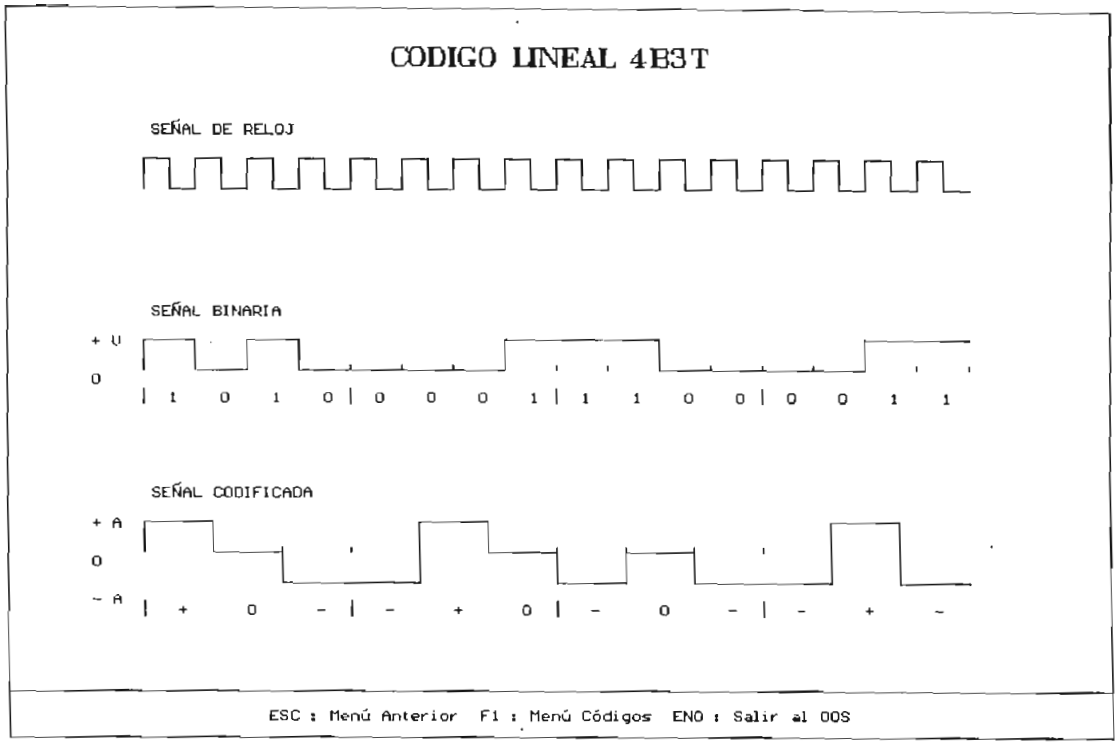


Figura 3.49. Ejemplo para el Código 4B3T
Polaridad P = -1

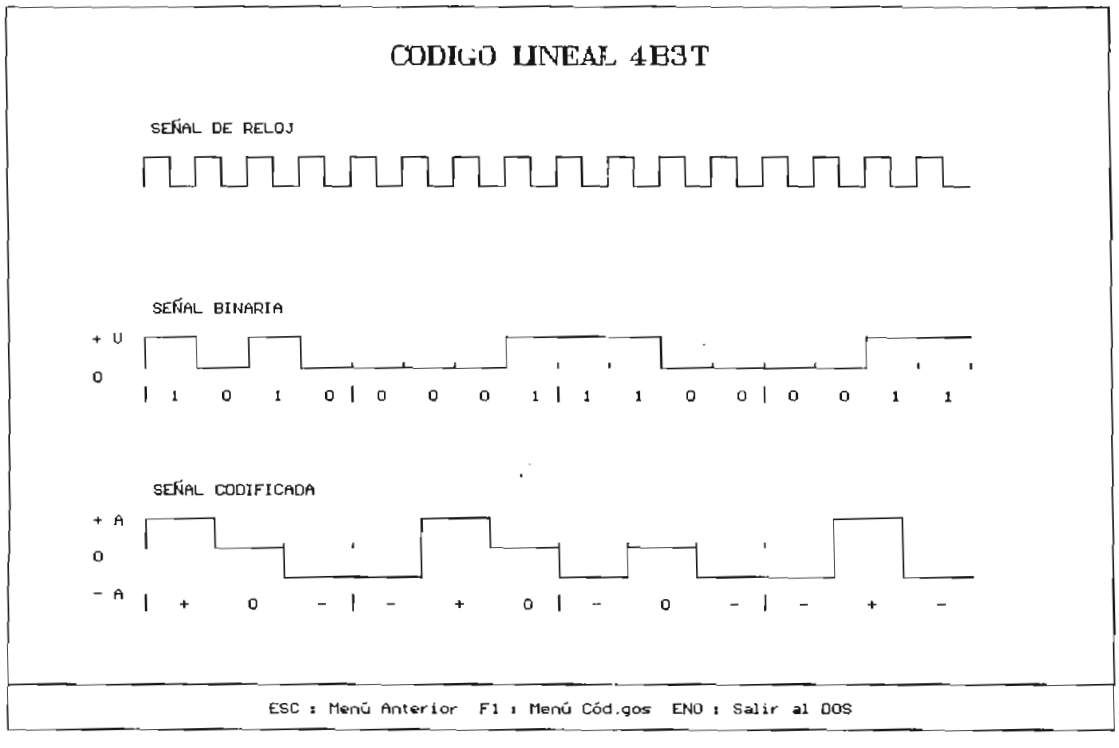


Figura 3.50. Ejemplo para el Código 4B3T
Polaridad P = 1

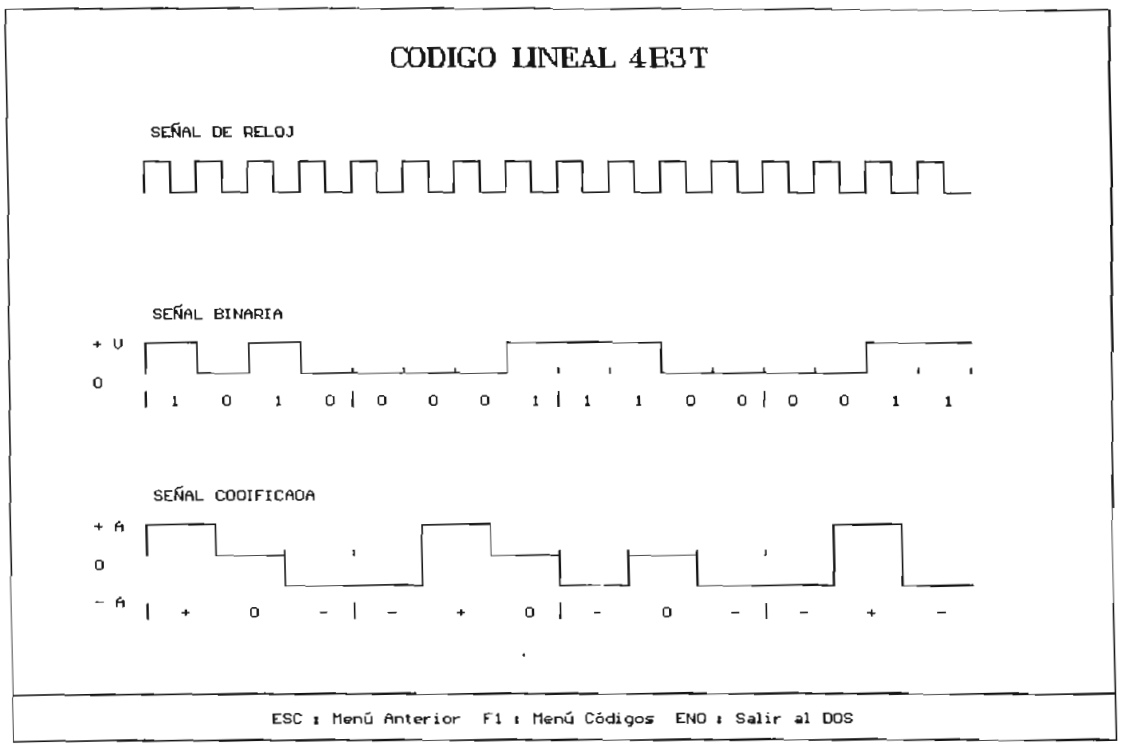


Figura 3.51. Ejemplo para el Código 4B3T
Polaridad P = 2

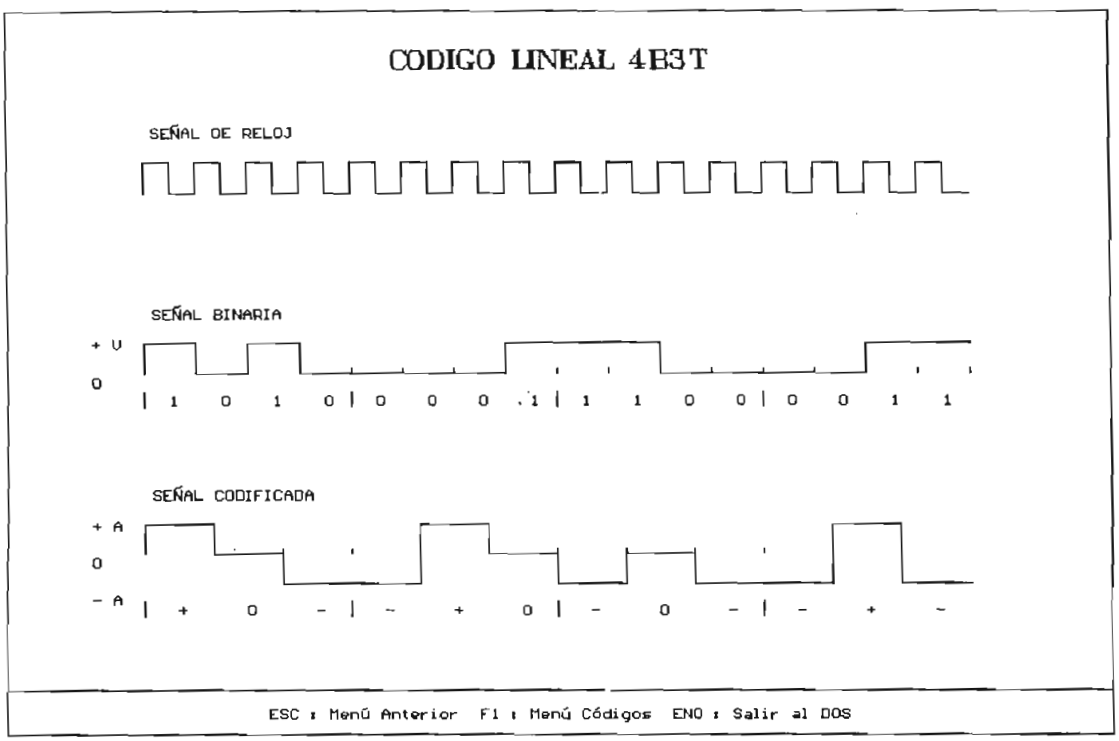


Figura 3.52. Ejemplo para el Código 4B3T
Polaridad P = 3

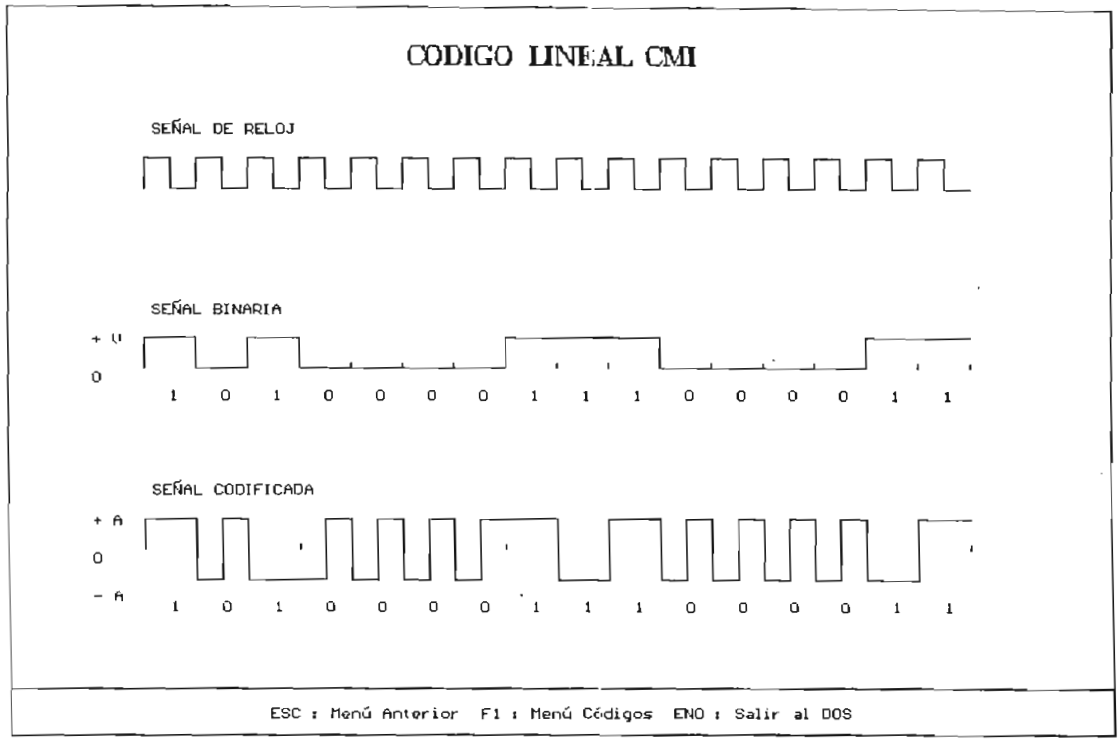


Figura 3.53. Ejemplo para el Código CMI

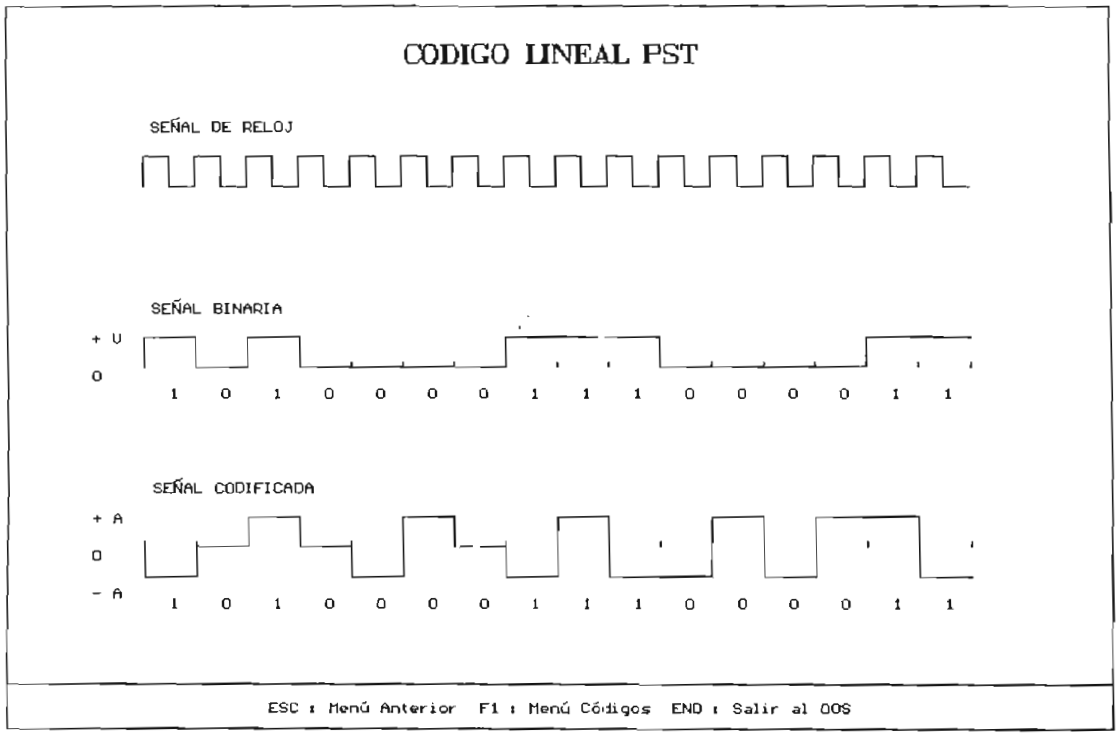


Figura 3.54. Ejemplo para el Código PST
Modo 0

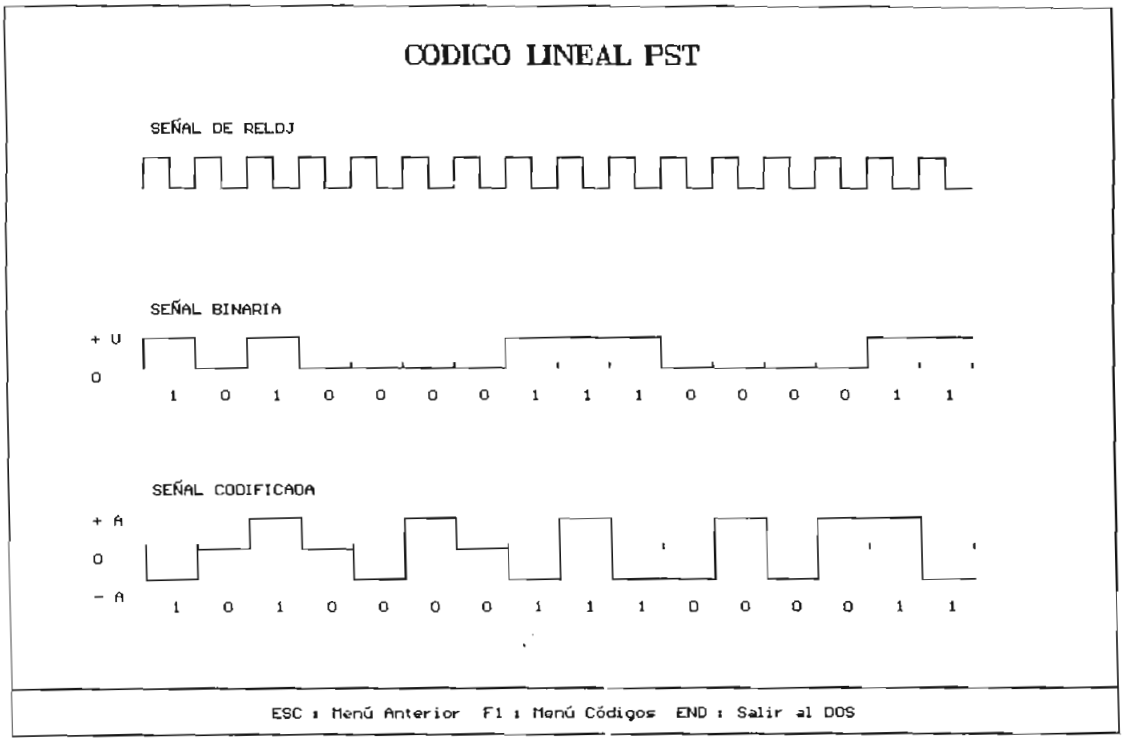


Figura 3.55. Ejemplo para el Código PST Modo 1

CAPITULO 4

MODULACION DIGITAL

4.1. INTRODUCCION

La transmisión de las señales de datos en los sistemas digitales presentan en general la misma apariencia independientemente de la fuente que les dio origen.

Es así como el flujo de dígitos que suministra un terminal de computadora no presenta diferencias apreciables, salvo la velocidad de señalización, con relación a la que presenta una señal PCM o una imagen de TV digitalizada.

Estas señales digitales, que se las denomina en banda base tienen que ser procesadas de alguna forma para su transmisión sobre líneas, cables u otros medios de transmisión. La codificación y la modulación realizan este proceso de adaptación de la señal.

La señal en banda base tiene que ser desplazada a frecuencias superiores, para que la transmisión sea más eficiente y a mayor distancia. Esto se logra por medio de la variación de amplitud, frecuencia o fase o la combinación de ellas de una onda senoidal portadora en general de alta frecuencia, de acuerdo con la información que se va a transmitir. Este proceso de adaptación recibe el nombre de **modulación**; en este caso al ser la señal modulante una señal digital surge el nombre de **Modulación Digital**. El uso de frecuencias superiores proporciona una radiación de la energía eléctrica más eficiente y pone al alcance anchos de banda superiores para una transferencia de un mayor volumen de información del que es posible con frecuencias inferiores.

Puede prestarse a confusión el hecho de que otros procesos de transmisión digital de señales analógicas tal como el PCM (Modulación por código de pulsos), ΔM (Modulación diferencial o delta) también se los denomina modulación; similar denominación suele aplicarse a las técnicas en las cuales se inscribe información analógica sobre un tren de pulsos, ya sea variando su amplitud PAM (Modulación por amplitud de pulso), PPM (Modulación por posición de pulso) y PDM (Modulación por ancho de pulso).

De todas maneras, estas señales donde la información está contenida en alguna característica de un tren de pulsos son sólo aptas para transmitirse en banda base. Consecuentemente, para adaptar una señal digital a un medio diferente de una línea deberá realizarse la modulación de una onda senoidal, también denominado proceso de modulación de onda continua CW (Wave Continous).

En conclusión, la modulación digital es el proceso de modulación de una onda senoidal, con la salvedad de que la señal modulante es digital. Esta modulación puede considerarse como una etapa más en el proceso de adaptación de la fuente de información al canal.

La señal digital en banda base constituye la señal moduladora y la señal que resulta del proceso es la portadora modulada.

Aunque las técnicas de modulación que se usan para las señales digitales (codificadas por pulsos) y las analógicas son conceptualmente iguales, existen razones para dar mayor importancia a la modulación digital:

- La creciente importancia de la transmisión de datos y el consecuente desarrollo de la industria adecuada, que se dedica a la producción especializada de equipos de modulación digital, indica que debe hacerse incapié en esta área.

- En la transmisión en banda base el ancho de banda no es normalmente una limitación muy rígida, no siendo así en los sistemas de radioenlace o en los canales telefónicos, en los cuales los anchos de banda son establecidos rígidamente por criterios de asignación.

Consecuentemente para lograr una mayor eficiencia en el uso de estos canales se requieren técnicas de señalización multinivel, las cuales están asociadas en general con el proceso de modulación.

Recordando el Teorema de Nyquist:

$$R_{max} = 2 \, . \, B \qquad (4.1)$$

donde R_{max} es el máximo ritmo de la señal (bits/seg) y B es el ancho de banda (Hz).

Así como la densidad de información δ :

$$\delta = R_{max} / B \qquad (4.2)$$

Se tiene que para un ancho de banda normalizado B = 1 Hz, la máxima densidad de información binaria es 2 (bps/Hz). Estos valores aumentan mediante las técnicas multinivel lográndose densidades de 3.2 (bps/Hz), para el caso de líneas telefónicas no conmutadas.

La modulación de CW no solo es necesaria para enviar señales por un medio determinado, sino que también conduce a la multicanalización por división de frecuencia FDM, es decir varios canales se transmiten al mismo tiempo en bandas de frecuencias adyacentes que no se traslapan. Esto quiere decir que muchas comunicaciones telefónicas pueden transmitirse por un simple par de alambres, dependiendo del ancho de banda total

del sistema, y fundamentalmente de los hilos de comunicación.

4.2. CLASES DE MODULACION

Existen esencialmente cuatro maneras de modular una portadora senoidal simple: variando su amplitud, su frecuencia, su fase, y su amplitud y fase de acuerdo a la información que se va a transmitir.

En el caso binario esto corresponde a la conmutación de uno de los tres parámetros o la combinación de ellos entre dos valores posibles.

En base a esto la modulación binaria se puede clasificar en las siguientes clases:

- Modulación de amplitud ASK (Amplitude Shift Keying)
- Modulación de frecuencia FSK (Frecuency Shift Keying)
- Modulación de fase PSK (Phase Shift Keying)
- Modulación QAM (Quadrature Amplitude Modulation)

4.2.1. Modulación de Amplitud ASK

La modulación ASK constituye probablemente la primera técnica de modulación digital que se haya implementado, dado que ha sido usada extensamente en la transmisión de telegrafía en código Morse.

En esta forma de modulación la amplitud de la portadora se varía entre dos niveles predeterminados en correspondencia con la señal binaria de datos. Esta clase de modulación también es denominada **modulación lineal**.

La ecuación matemática que relaciona esta modulación es:

$$g(t) = [\,1 + k \,.\, b_n(t)\,]\; Cos\,(\omega_c\, t) \qquad (4.3)$$

donde: k = índice de modulación ($0 < k \leq 1$)

b_n = tren de impulsos rectangulares de señal en banda base NRZ de amplitud unitaria.

$\omega_c = 2.\pi.f_c$, frecuencia angular de la portadora

Modulación Lineal implica que el espectro de la señal modulada se obtiene trasladando el espectro de la banda base a la banda de frecuencia de la portadora seleccionada.

En la figura 4.1 se representa la modulación ASK en diagrama de bloques.

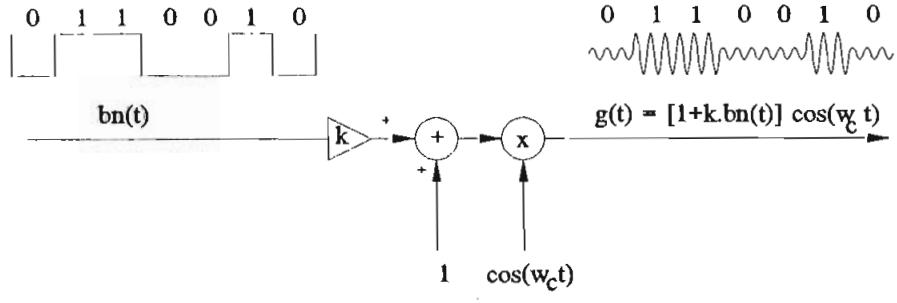


Figura 4.1. Diagrama de bloques de la modulación ASK.

Para reducir la potencia de transmisión se emplea el método de modulación equilibrada, en el que se suprime la onda portadora; si además se utiliza una modulación al 100% (**k**=1), g(t) es representado por la siguiente fórmula:

$$g(t) = b_n(t) \cdot Cos(\omega_c \cdot t) \qquad (4.4)$$

A la modulación indicada por esta ecuación se le llama de doble banda lateral DSB (Double Side Band) con portadora suprimida.

Si $b_n(t)$ representa un tren de pulsos unipolares, la forma de onda de g(t) resulta como la indicada en la figura 4.2.(b). La marca se representa por 1 y el espacio por 0. Esta forma de onda es igual a la lograda mediante manipulación **ON-OFF** de la onda continua usando un interruptor ideal, por eso se llama **ON/OFF ASK** u **ON/OFF Keying (OOK)**. El diagrama de bloques de este tipo de modulación se puede observar en la figura 4.2.(a).

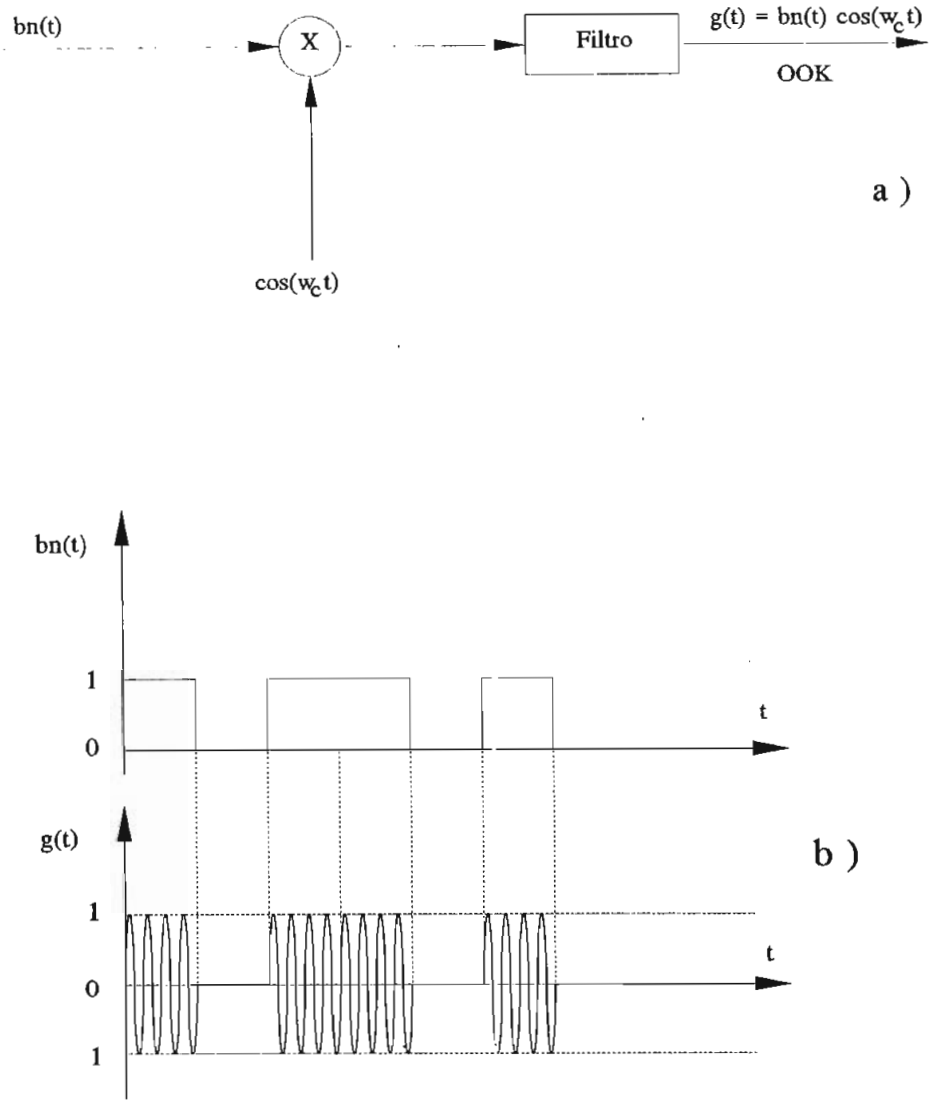


Figura 4.2.a) Diagrama de bloques de la modulación OOK
b) Señal modulada OOK

Si $b_n(t)$ representa un tren de pulsos bipolares la g(t) resulta como la de la figura 4.2.(c). En este caso la marca se representa por una amplitud de +1/2 y el espacio por una amplitud de -1/2.

Esta clase de modulación se le conoce como ASK por inversión de fase o modulación 2-PSK.

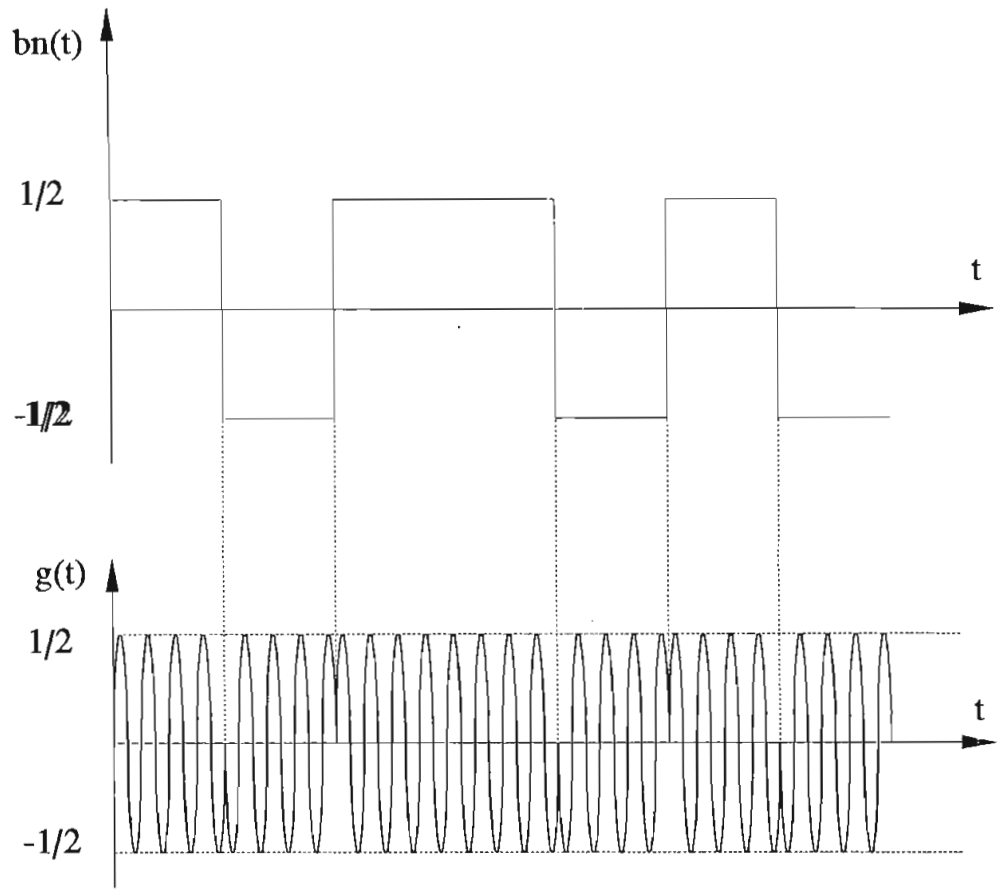


Figura 4.2.c) Señal modulada por inversión de fase

Comparando 4.2.(b) con 4.2.(c), ambas señales de banda base tienen la misma potencia media y la única diferencia está en que los niveles de polarización de corriente directa son distintos.

4.2.1.1. Demodulación ASK

En la figura 4.3 se representa un diagrama de bloques de un demodulador ASK. El filtro de recepción es normalmente un filtro adaptado a los pulsos de alta frecuencia. La demodulación puede realizarse en forma coherente (sincrónica) o no coherente. La demodulación no coherente es la más sencilla de implementarse. La salida del detector, una vez filtrada para remover componentes indeseables, equivale a una señal recibida en un sistema de banda base a la que se le aplica la decodificación, muestreo y decisión propios de estos sistemas.

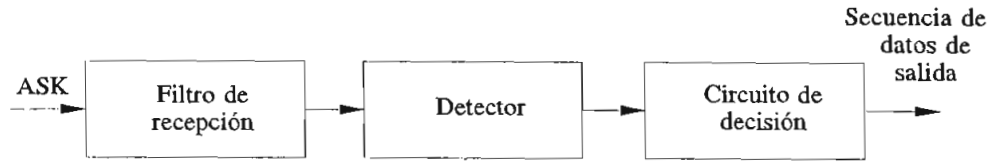


Figura 4.3. Demodulador ASK

4.2.2. Modulación de frecuencia FSK

FSK transporta la información digital como una variación de la frecuencia de la portadora, proporcionando una señal de amplitud constante, lo cual constituye una de sus ventajas respecto a ASK porque permite transmitir niveles de potencia mayores. La modulación FSK es usada principalmente en radios digitales de banda angosta, siendo más estable que ASK en presencia de desvanecimiento de señal. También es usada para transmisión de datos a través de línea telefónica.

La expresión matemática para una señal FSK es la siguiente:

$$g(t) = Cos\ (\omega_c + \frac{b_n . \delta\omega}{2})\ .\ t \qquad (4.5)$$

donde: ω_c = Frecuencia de la portadora

b_n = Señal digital NRZ en banda base de 2 niveles

$\delta\omega$ = Diferencia de frecuencia entre las dos señales

De esto se deduce que la amplitud de la señal en banda base produce una desviación de frecuencia de $+\delta\omega/2$ para un 1_L y de $-\delta\omega/2$ para un 0_L.

Un típico modulador y demodulador FSK se muestra en la figura 4.4. El modulador es un oscilador controlado por voltaje VCO, que es polarizado para producir la frecuencia central de la portadora si ninguna modulante es aplicada. El demodulador está implementado con un PLL, un VCO, un detector de fase y un filtro para minimizar los efectos de ruido.

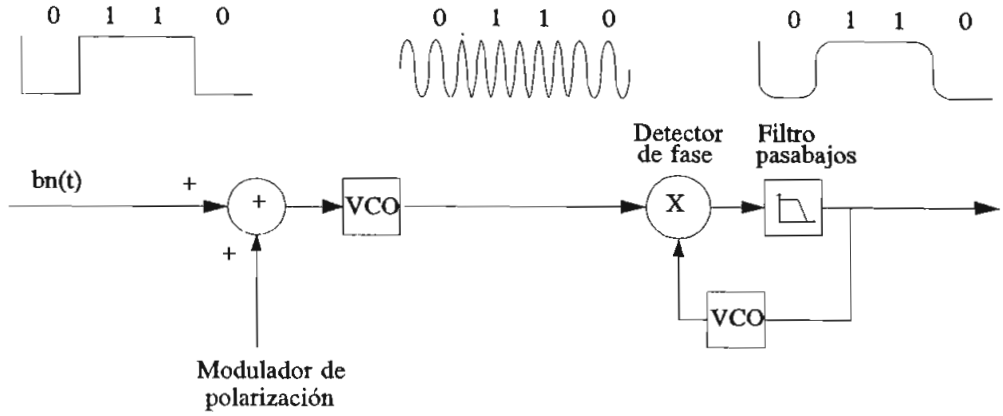


Figura 4.4. Modulación y Demodulación FSK

Existen dos métodos de generar la señal FSK. El primero, el más utilizado, consiste en generar una señal en banda base que luego se aplica a un modulador típico de frecuencia, así el instante de realizarse el cambio o conmutación de frecuencia se mantendrá la fase de la señal llamándose a este tipo de modulación como **Coherente.**

El segundo método consiste en disponer de dos osciladores a las frecuencias deseadas e ir conectando atendiendo a la secuencia de la señal de datos a la salida de uno u otro. Este método se lo conoce como Modulación **No Coherente,** en el cual el momento de la conmutación de

frecuencia no se mantiene la fase de la señal, produciéndose en la mayoría de conmutaciones saltos de fase.

Este tipo de generación es el más simple, pero las discontinuidades en los instantes significativos hacen que el espectro tenga una caída muy suave, por lo que requieren un ancho de banda mayor que el otro tipo de modulador con fase continua.

En la figura 4.5 se tienen los dos tipos de modulaciones FSK.

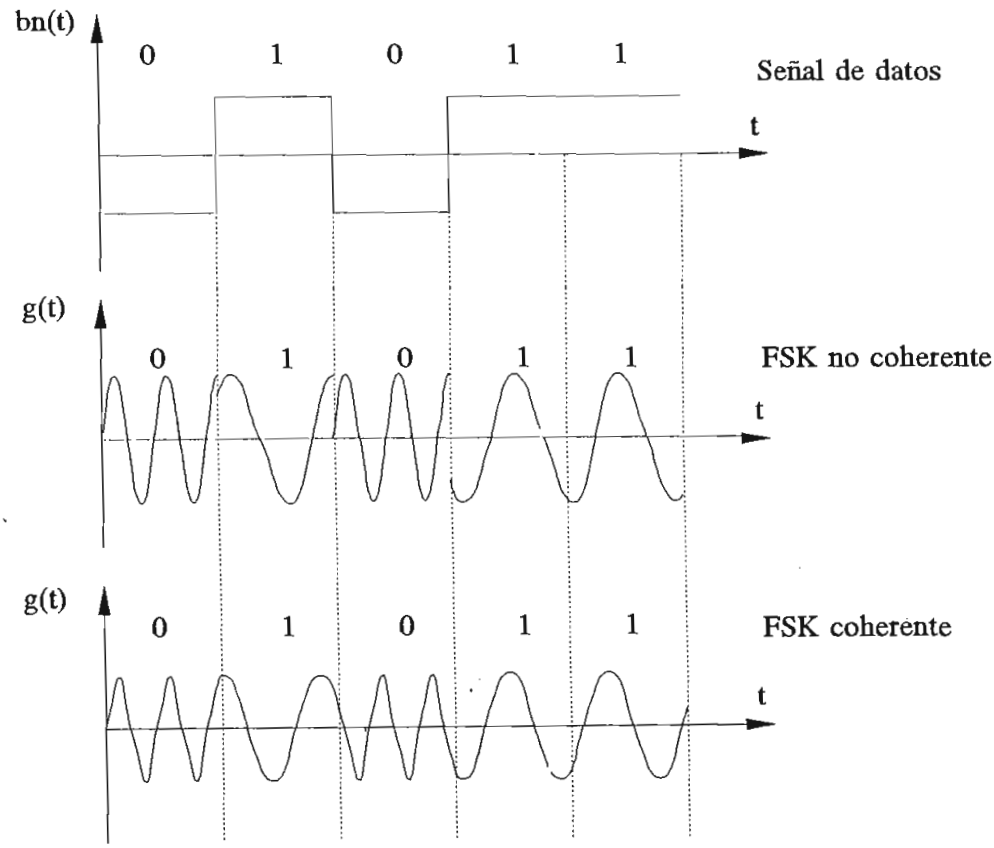


Figura 4.5. Modulación FSK Coherente y No Coherente

La modulación FSK aunque muy utilizada por su robustez frente a perturbaciones y la sencillez de los equipos, tiene gran dificultad en el análisis teórico debido a la no linealidad del proceso, no permitiendo además la utilización

del canal con un buen rendimiento.

4.2.2.1. Demodulación FSK

Las señales FSK pueden ser demoduladas de manera coherente o no coherente. El detector coherente requiere del conocimiento exacto de la fase de la señal de entrada, su esquema se representa en la figura 4.6.(a).

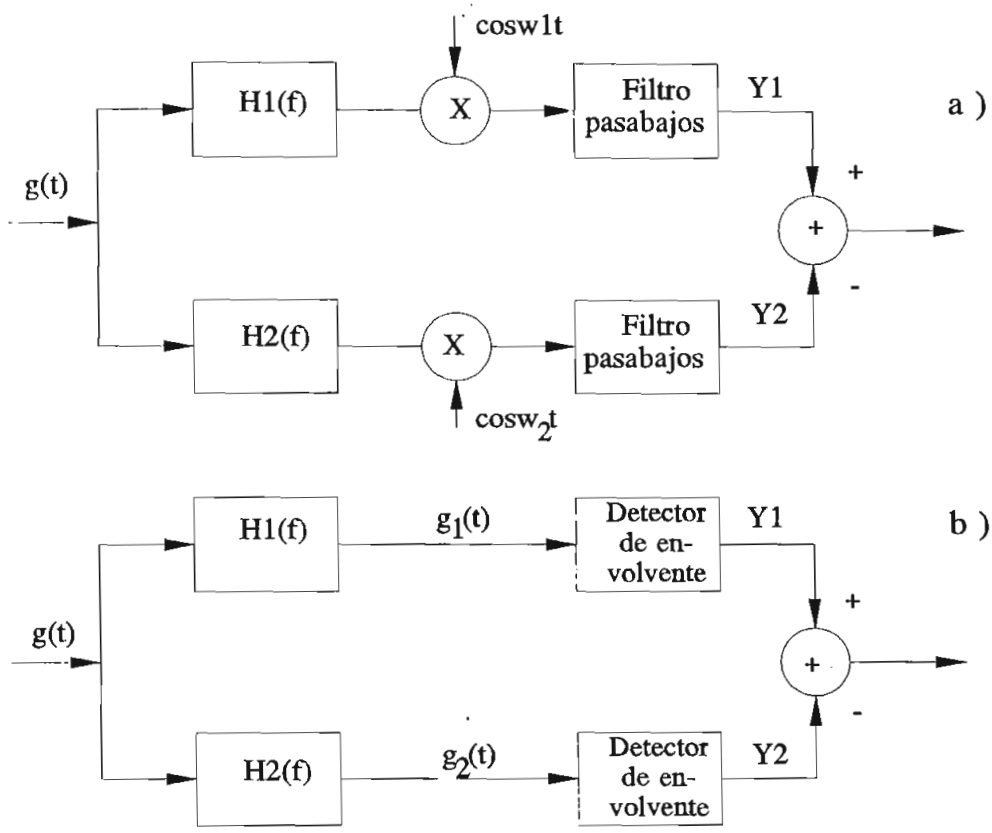


Figura 4.6. Demodulación FSK
a) Sincrónica
b) De envolvente

Cuando no se conoce exactamente la fase de la señal de entrada se debe recurrir a detectores incoherentes. En la figura 4.6.(b) se indica un detector de este tipo que emplea detectores de envolvente.

En la entrada del demodulador de la figura 4.6.(b) se tiene la señal g(t), de naturaleza FSK y por tanto de amplitud constante, la que conmuta instantáneamente entre dos frecuencias f_1 y f_2. Esta señal puede considerarse como dos señales OOK que se verán a la salida de cada filtro $H_1(f)$ y $H_2(f)$. Es decir que $g_1(t)$ y $g_2(t)$ serán dos señales OOK que serán demoduladas por los detectores de envolvente obteniéndose Y_1 y Y_2, que sumadas proporcionarán una señal bipolar la cual es analizada en los circuitos de decisión.

4.2.3. Modulación de fase PSK

Este tipo de modulación también denominado modulación discreta de fase es sumamente eficiente, ampliamente utilizada en sistemas tales como enlaces satelitales, radioenlaces de banda ancha, enlaces telefónicos, etc.

En esta técnica de modulación la información se transmite en la fase de una portadora de amplitud constante. Cuando dicha información está representada por el valor absoluto de la fase, es decir referida a una portadora sin modular se tiene el sistema PSK convencional; si la información está contenida en las variaciones de fase, es decir referida a la fase del estado anterior se tienen los denominados sistemas diferenciales.

El tipo de modulación convencional ya se observó anteriormente en la modulación ASK por inversión de fase, en la que cada intervalo de la señal utiliza una de las dos fases separadas 180°. La única diferencia es que en ASK por inversión de fase la portadora se conmuta entre las amplitudes +1/2 y -1/2 mientras que en 2-PSK, la portadora se conmuta entre +1 y -1. La gran mayoría de los enlaces de baja capacidad de 2 (Mbps), 8 (Mbps), y los de mediana capacidad 34 (Mbps) utilizan M-PSK.

La expresión matemática para la modulación 2-PSK es:

$$g(t) = Cos\,(\omega_c . t + \frac{b_n . \delta\phi}{2}) \qquad (4.6)$$

donde: $\delta\phi = \pi$ es la separación entre fases de señales adyacentes

b_n = Señal simétrica NRZ en banda base con valores +1 y -1

Esta expresión tiene la misma forma que una modulación lineal, por lo tanto se pueden generar con los mismos métodos utilizados para tales modulaciones.

En la figura 4.7 se presenta un modulador 2-PSK, con puntos de código colocados a intervalos de π en una circunferencia que representa la fase de la onda portadora.

$$b_n = +1 \qquad g(t) = Cos\,(\omega_c . t + \pi/2) \qquad (4.7)$$

$$b_n = -1 \qquad g(t) = Cos(\omega_c . t - \pi/2) \qquad (4.8)$$

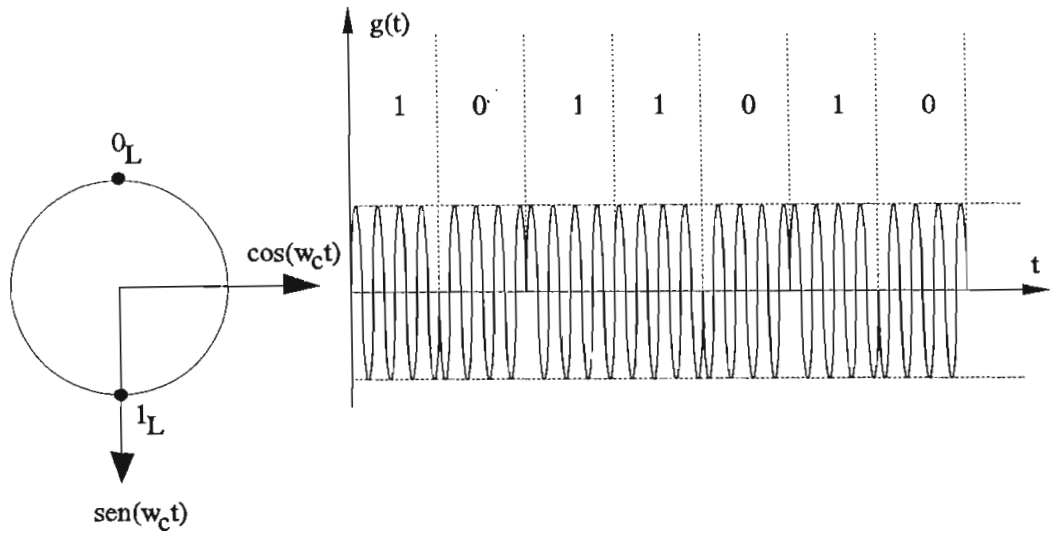


Figura 4.7. Modulación 2-PSK

4.2.3.1. Demodulación 2-PSK

La figura 4.8.(a) representa un diagrama de bloques de un demodulador PSK. La etapa fundamental en el demodulador la constituye el detector sincrónico de la figura 4.8.(b), donde se multiplica la señal modulada con la portadora. El producto así obtenido se hace pasar por un filtro pasabajos y esta señal irá al circuito de decisión.

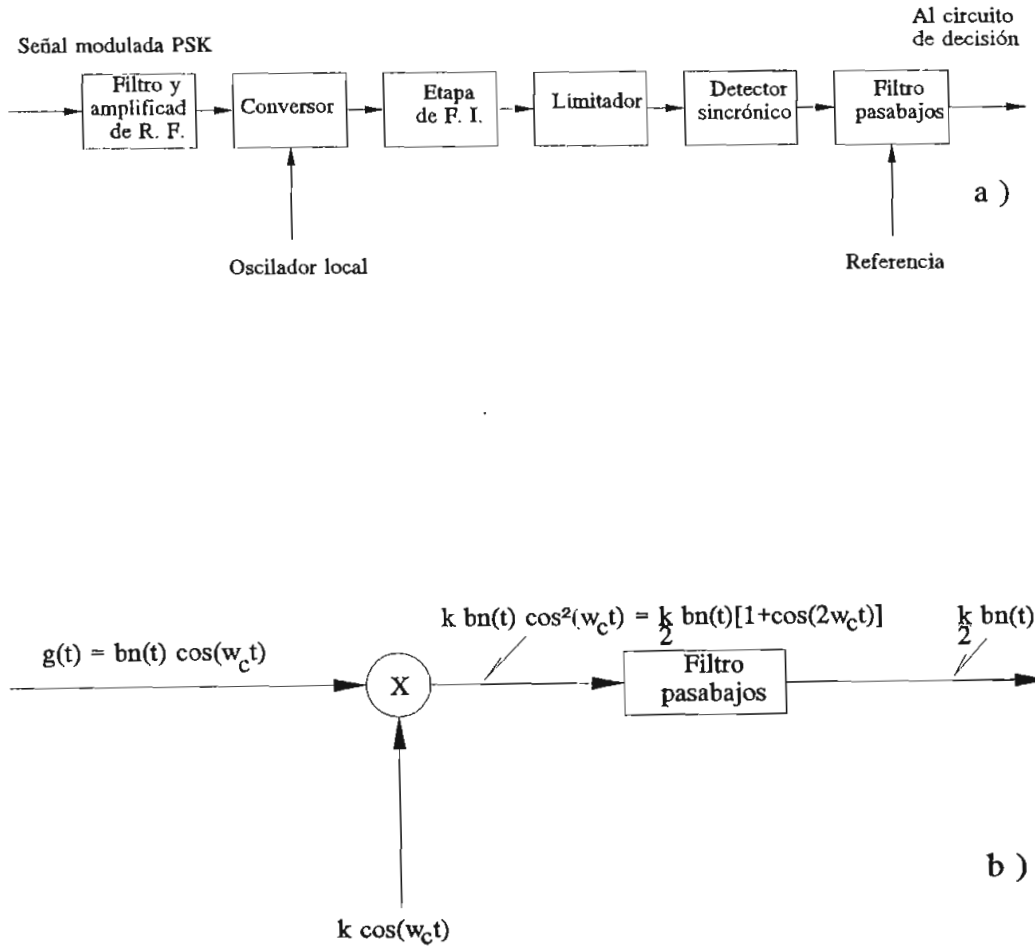


Figura 4.8.a) Diagrama de bloques del Demodulador PSK
b) Esquema de un detector síncrono

4.2.3.2. Modulación DPSK

En la modulación PSK en su etapa de detección se producen errores de sincronismo, por lo cual suele utilizarse una modulación denominada **DPSK** o "PSK diferencial", en la cual la información se codifica usando las diferencias de fases entre dos bits sucesivos. Del mensaje binario $b_n(t)$ de entrada en el transmisor, se genera una secuencia diferencial $d_n(t)$ que tiene un dígito adicional de comienzo arbitrario, el cual se supone por ejemplo que es 1. Los dígitos sucesivos del código diferencial se obtienen mediante:

$$d_n = d_{n-1} \cdot b_n \oplus \overline{d_{n-1}} \cdot \overline{b_n} \qquad (4.9)$$

En la figura 4.9 se puede observar el modulador y demodulador DPSK.

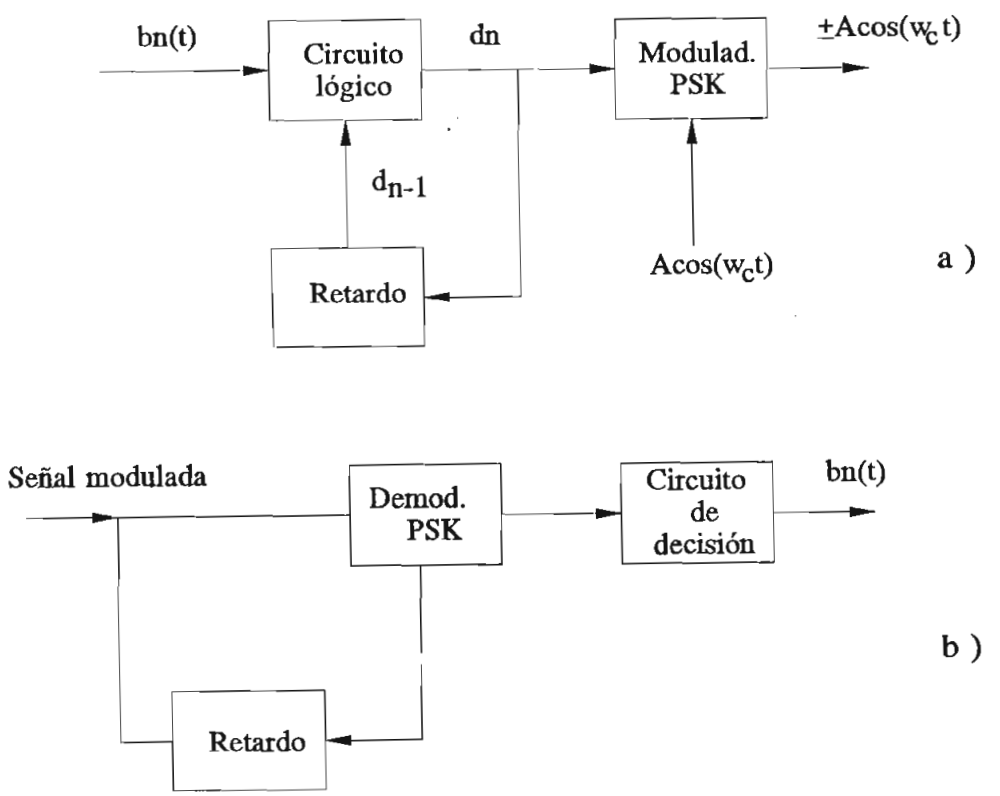


Figura 4.9. a) Modulador DPSK
b) Demodulador DPSK

Un ejemplo de codificación y decodificación DPSK se puede observar en la tabla 4.1.

Entrada $b_n(t)$		1	1	0	1	0	0	0	1
Mensaje diferencial $d_n(t)$	1	1	1	0	0	1	0	1	1
Fase transmitida	0	0	0	π	π	0	π	0	0
Comparación de fases a la salida		+	+	-	+	-	-	-	+
Salida $b_n(t)$		1	1	0	1	0	0	0	1

Tabla 4.1. Codificación y Decodificación Diferencial

En el mensaje diferencial el dígito 1 tiene fase 0 y el dígito 0 tiene fase π. En el receptor DPSK la fase del dígito previo servirá como referencia, si las fases de dos dígitos sucesivos son iguales ocurre una salida positiva (1_L), y si son diferentes la salida será negativa (0_L), decodificando así exactamente el mensaje original.

Debido a que en DPSK la modulación de un bit se determina sobre la base de la señal recibida en dos intervalos sucesivos de bit, su desventaja radica en que los errores de bit tienden a ocurrir en pares; en efecto si sucede un error esto originará un error de decodificación en el bit siguiente.

En conclusión se implementará DPSK cuando se quiere evitar ambigüedades de fase, ya que en DPSK la información ya no está contenida en la fase absoluta sino en las transiciones.

4.2.4. MODULACION MULTISIMBOLICA M-PSK

En los sistemas de banda base se señaló que ha efectos de reducir el ancho de banda, puede recurrirse a esquemas de señalización de más de dos niveles llamados de señalización multinivel. Estos esquemas consisten en la combinación de sucesivos pulsos binarios para formar un pulso

de mayor duración, lo que en consecuencia requerirá un menor ancho de banda de transmisión.

Específicamente por Nyquist se sabe que se puede transmitir 2 (bps/Hz) por el canal de ancho de banda de *B* Hertzios. Si se usa un conjunto de M = 2^n símbolos, siendo n el número de dígitos binarios sucesivos que se combinan para formar el símbolo adecuado que se va a transmitir, pueden transmitirse **2.n** (bps/Hz) usando la banda de Nyquist.

Consecuentemente la señalización multinivel puede extenderse a las diversas técnicas de modulación consideradas.

Se tiene modulación multisimbólica (o multinaria) en ASK, PSK, FSK y QAM. Comúnmente se prefiere utilizar esquemas multinarios cuando se desea conservar el ancho de banda a expensas de un aumento en la potencia transmitida.

En general las técnicas de multifase y multinivel combinadas así como la de multifase, se aplican para reducir ancho de banda y son de gran aplicación en sistemas telefónicos satelitales y en telefonía. Los esquemas multifrecuenciales, por el contrario, suelen producir mayores anchos de banda y como contrapartida presentan una mejor inmunidad frente al ruido.

En base a esto, en los esquemas de modulación multisimbólica se supone que el modulador toma bloques de n dígitos binarios y asigna una de las M = 2^n formas de onda posibles a cada una de las diferentes M combinaciones de los n dígitos binarios.

En el esquema M-PSK la fase de la portadora puede tomar uno de los M valores posibles separados en un ángulo $\delta\phi$:

$$\delta\phi = \frac{2\pi}{M} \qquad (4.10)$$

En consecuencia, las M formas de onda posibles de ser transmitidas están dadas por la siguiente expresión general:

$$g(t) = Cos\,(\omega_c \,.\, t + \frac{b_n(t)\,.\,\delta\phi}{2}) \qquad (4.11)$$

donde $b_n(t)$ es una señal de banda base, simétrica, NRZ, cuyos niveles son: ±1, ±3, ±5, ...

4.2.4.1. Modulación 4-PSK

En esta modulación M = 4 y es conocida como 4-PSK o Q-PSK (donde Q corresponde a Quaternary). En este caso se combinan los dígitos binarios en 00, 01, 10, y 11, con lo cual deben existir cuatro ángulos de fase que les correspondan siendo $\delta\phi = \pi/2$.

$$b_n = +1 \qquad \phi_1 = (\frac{b_n(t)\,.\,\delta\phi}{2}) = +\frac{\pi}{4} \qquad (4.12)$$

$$b_n = +3 \qquad \phi_2 = (\frac{b_n(t)\,.\,\delta\phi}{2}) = +\frac{3\pi}{4} \qquad (4.13)$$

$$b_n = -3 \qquad \phi_3 = (\frac{b_n(t)\,.\,\delta\phi}{2}) = -\frac{3\pi}{4} \qquad (4.14)$$

$$b_n = -1 \qquad \phi_4 = (\frac{b_n(t)\,.\,\delta\phi}{2}) = -\frac{\pi}{4} \qquad (4.15)$$

Debe quedar claro que estos dos pulsos binarios sucesivos se almacenan para luego emitir el símbolo (la forma de onda) correspondiente.

Expandiendo la ecuación general y para el caso de M = 4 se tiene:

$$g(t) = a_x .\ Cos\ \omega_c . t + b_x .\ Sen\ \omega_c . t \qquad (4.16)$$

donde:

$$a_x = Cos\left(\frac{b_n(t) \cdot \delta\phi}{2}\right) \qquad (4.17)$$

y

$$b_x = -Sen\left(\frac{b_n(t) \cdot \delta\phi}{2}\right) \qquad (4.18)$$

Se tendrá en consecuencia los siguientes pares ordenados:

$$a_x, b_x = \begin{matrix} (1,1)/\sqrt{2} \\ (-1,1)/\sqrt{2} \\ (-1,-1)/\sqrt{2} \\ (1,-1)/\sqrt{2} \end{matrix} \qquad (4.19)$$

Las ecuaciones 4.16, 4.17, y 4.18 corresponden a dos componentes en cuadratura las cuales se transmiten en forma simultánea. Dichas componentes solamente podrán tomar dos valores diferentes según el flujc de datos modulante. Estas componentes reciben la denominación de componentes en fase por el Coseno, y en cuadratura por el Seno, debido a su posición referida a la portadora sin modular.

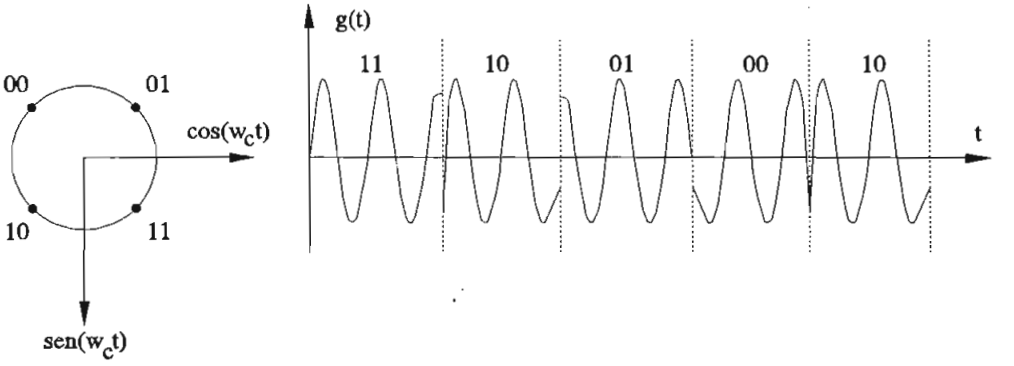


Figura 4.10. Diagrama de Constelación y formas de onda 4-PSK

Es sumamente ilustrativo indicar las cuatro posibles posiciones que ocupa la portadora modulada, recurriendo a la

representación en el espacio de la señal. Por claridad se suele representar las señales por un punto, el cual no es más que el extremo del vector que le corresponde en dicho espacio, formando lo que se denomina Diagrama de Constelación. En la figura 4.10 se representa este diagrama y las respectivas formas de onda para 4-PSK.

En la tabla 4.2 se indica el valor de las componentes en cuadratura en función de los dígitos binarios de entrada. Se muestra también la expresión correspondiente a la portadora modulada.

Dígitos binarios	Coeficientes		Portadora modulada
	a_x	b_x	
01	0,707	-0,707	$Cos(Wct+\pi/4)$
00	-0,707	-0,707	$Cos(Wct+3\pi/4)$
10	-0,707	0,707	$Cos(Wct-3\pi/4)$
11	0,707	0,707	$Cos(Wct-\pi/4)$

Tabla 4.2. Componentes en cuadratura de 4-PSK

La modulación 4-PSK puede realizarse mediante diversos métodos. El primero de ellos se basa en la combinación lineal de dos señales en cuadratura tal como se muestra en la figura 4.11. En este diagrama un par de bits de información digital se almacenan en un registro de desplazamiento; los mismos se aplican a dos moduladores PSK binarios, obteniendo de esta forma dos ondas moduladas en los ejes I y Q cuyas portadoras están en cuadratura entre sí. Cada modulador recibe un flujo de datos a la mitad de la velocidad con la que ingresa la información digital. Las dos señales PSK binarias son luego sumadas para reproducir los cuatro estados posibles que constan en la tabla 4.2.

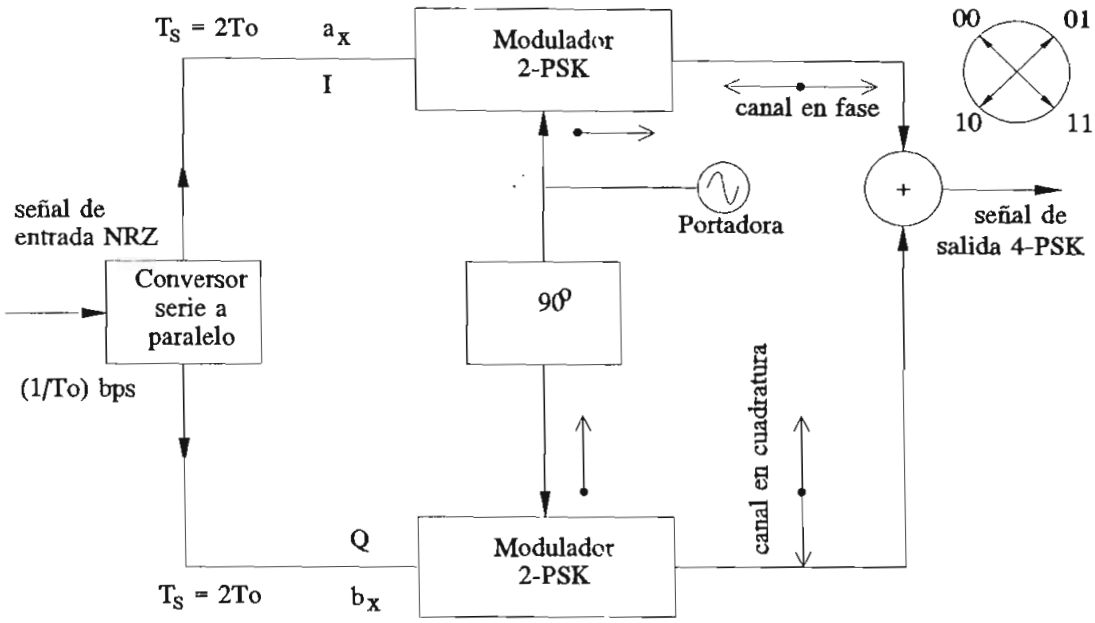


Figura 4.11. Modulador 4-PSK. Diagrama de bloques

4.2.4.2. Demodulación 4-PSK

Para la demodulación o detección de 4-PSK primeramente se tiene una referencia sincrónica que provee:

$$ya_x = Cos(\omega_c t + \frac{b_n(t)\,\delta\phi}{2})\, Cos\omega_c t \qquad (4.20)$$

la cual convenientemente filtrada dará:

$$yA_x = Cos(\frac{b_n(t)\,.\,\delta\phi}{2}) \qquad (4.21)$$

Es necesaria otra referencia ortogonal:

$$yb_x = Cos(\omega_c t + \frac{b_n(t)\,.\,\delta\phi}{2})\, Sen\omega_c t \qquad (4.22)$$

la cual después del filtro será:

$$yB_x = -Sen\left(\frac{b_n(t).\delta\phi}{2}\right) \qquad (4.23)$$

De esta manera todas las decisiones del detector se basan en la polaridad de $y\,A_x$ e $y\,B_x$. Si se observa la tabla 4.1, resulta que el primer bit es 0 cuando el ángulo de fase es positivo ($\pi/4$ *o* $3\pi/4$). Es decir que este primer bit será completamente especificado por la polaridad de:

$$Sen\left(\frac{b_n(t).\delta\phi}{2}\right) \qquad (4.24)$$

que es precisamente la salida yB_x del receptor sincrónico con referencia ortogonal. Con un análisis similar, el segundo bit será 1 cuando la fase es $\pm\pi/4$, o sea cuando el coseno es positivo. En la figura 4.12 se representa el demodulador 4-PSK con su etapa de regeneración, éste consiste en dos canales cada uno de ellos operando en forma binaria; cada canal posee un detector de fase seguido de un filtro pasabajos. La salida de los mismos se aplica a un regenerador y también al circuito de recuperación de reloj. Posteriormente el flujo de datos regenerado pasa por un decodificador diferencial, el cual elimina la decodificación realizada en el transmisor. Finalmente los dos canales son multiplexados bit a bit compensando el efecto producido por el registro de desplazamiento del modulador.

Considerando las figuras 4.11 y 4.12 se puede resumir que la información digital de entrada se divide en dos secuencias binarias intercaladas de velocidad ($1/2T_o$) bps., las cuales modulan en PSK binaria dos componentes en cuadratura. Estas secuencias binarias son recién reintegradas en el

multiplexador de salida del detector. Debido a este análisis 4-PSK puede verse como dos PSK en cuadratura.

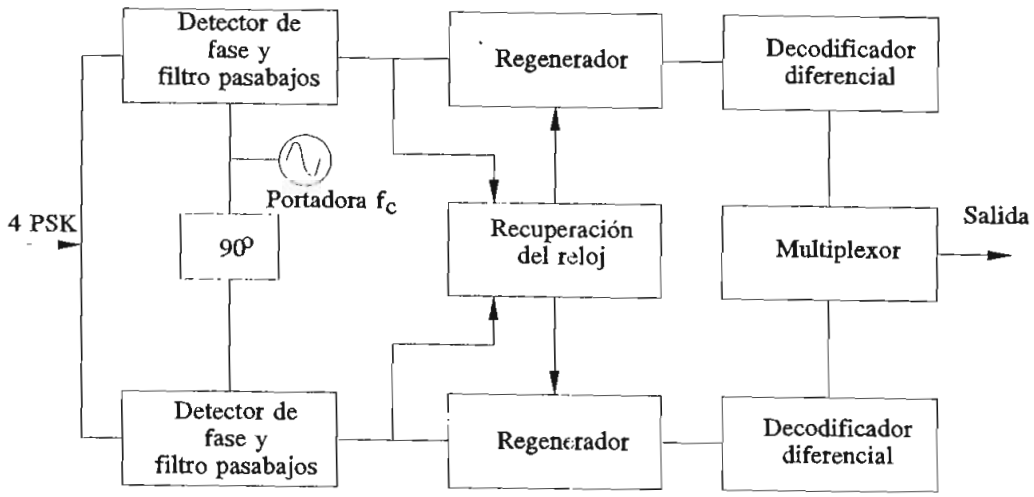


Figura 4.12. Diagrama de bloques del Demodulador 4-PSK

4.2.4.3. Modulación 8-PSK

Para este caso se tiene que M = 8 denominándose a esta modulación 8-PSK, en la que las 8 fases diferentes estarán separadas un ángulo $\delta\phi = \pi/4$, siendo evidente que cada fase representa un grupo de tres dígitos binarios.

En la tabla 4.3 se indica la expresión de la portadora correspondiente a las ocho posibles combinaciones de tres dígitos binarios, en la cual se indica también el valor de las componentes en cuadratura.

En la figura 4.13 se muestra el Diagrama de Constelación y las formas de onda para la Modulación 8-PSK.

Para la generación de señales 8-PSK, podrían aplicarse los métodos expuestos para 4-PSK, debiéndose notar sin embargo que ahora las componentes en cuadratura ya no sólo varían en signo, sino también en amplitud. Para ello habría que transformar el registro de desplazamiento del modulador 4-PSK

de la figura 4.11 en un generador de señales en banda base, que entregará a los moduladores las magnitudes de a_x y b_x según los valores del tribit que se almacene en un intervalo dado.

Dígitos binarios	a_x	b_x	Portadora modulada
011	0,924	-0,383	Cos(Wct+π/8)
010	0,383	-0,924	Cos(Wct+3π/8)
000	-0,383	-0,924	Cos(Wct+5π/8)
001	-0,924	-0,383	Cos(Wct+7π/8)
101	-0,924	0,383	Cos(Wct-7π/8)
100	-0,383	0,924	Cos(Wct-5π/8)
110	0,383	0,924	Cos(Wct-3π/8)
111	0,924	0,383	Cos(Wct-π/8)

Tabla 4.3. Componentes en cuadratura para modulación 8-PSK

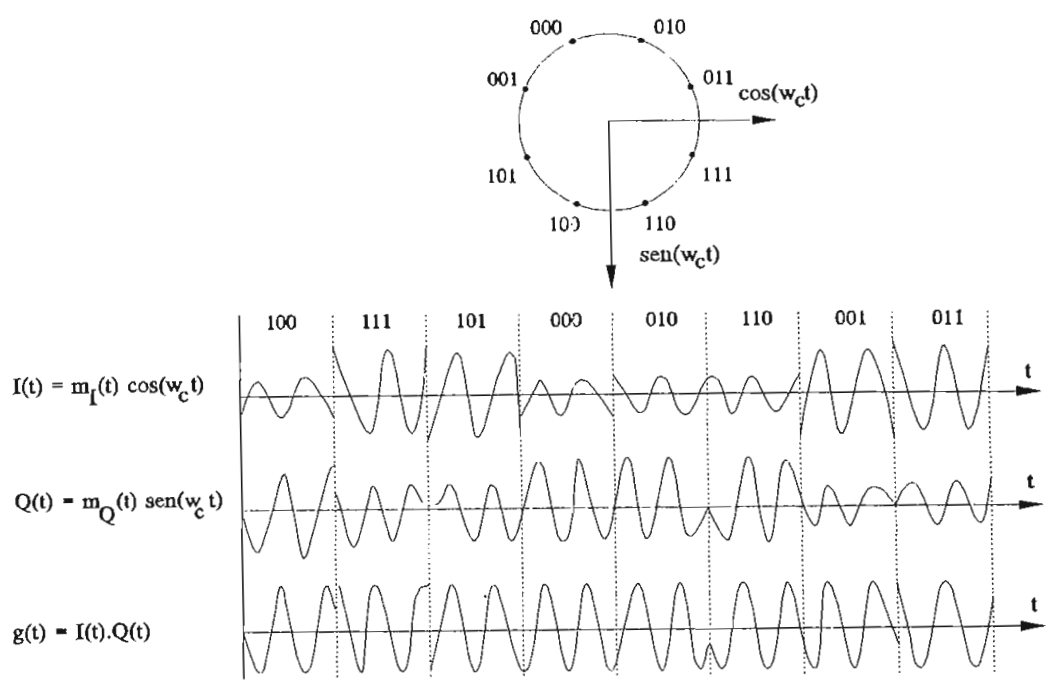


Figura 4.13.Diagrama de Constelación y formas de onda para modulación 8-PSK

Un esquema en diagrama de bloques de un modulador 8-PSK se observa en la figura 4.14. El mismo consiste en dos moduladores 4-PSK y un modulador PSK.

La salida del modulador 4-PSK superior está dada por los dos primeros bits A y B del tribit que se está transmitiendo. Lo mismo sucede con el modulador 4-PSK inferior si el tercer bit C es un 0; si el tercer bit C es un 1 su constelación se modifica tal como se indica en la figura 4.14. La composición de ambas salidas en el sumador, entrega finalmente la constelación 8-PSK deseada.

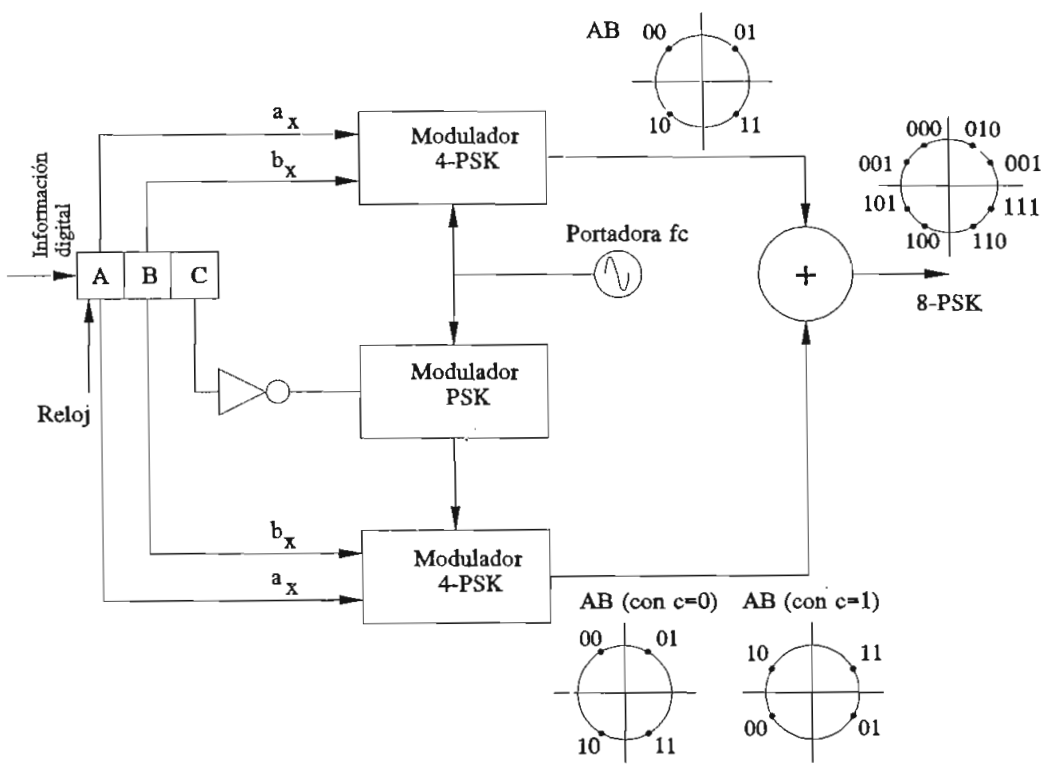


Figura 4.14. Diagrama de bloques de modulación 8-PSK

4.2.4.4. Demodulación 8-PSK

Para la demodulación 8-PSK, se ve que si se aplica un circuito como el de la figura 4.12 con sólo dos referencias Coseno y Seno, no se puede discriminar entre los puntos de Constelación de un mismo cuadrante. Se deben introducir dos

referencias adicionales en el receptor y medir la fase de la señal recibida también con respecto a ellas. En la figura 4.15 se indica la posición de las nuevas fases de referencia denominadas **c** y **d** las cuales son $Cos(\omega_c t + \pi/4)$ y $Sen(\omega_c t + \pi/4)$ respectivamente como las ya usadas **a** y **b**.

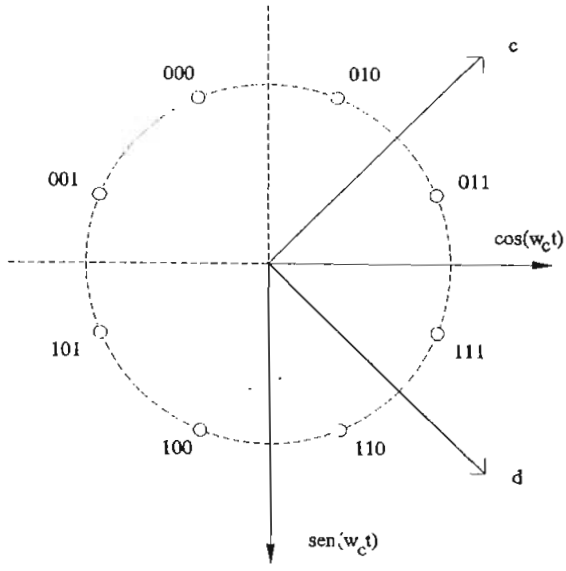


Figura 4.15. Fases de referencia para 8-PSK

La salida de los cuatro detectores son determinadas por:

$$ya_x = Cos[\omega_c t + \frac{b_n(t).\delta\phi}{2}] \; . \; Cos\ \omega_c t \qquad (4.25)$$

$$yb_x = Cos[\omega_c t + \frac{b_n(t).\delta\phi}{2}] \; . \; Sen\ \omega_c t \qquad (4.26)$$

$$yc_x = Cos[\omega_c t + \frac{b_n(t).\delta\phi}{2}] \; . \; Cos(\omega_c t + \pi/2) \qquad (4.27)$$

$$yd_x = Cos[\omega_c t + \frac{b_n(t).\delta\phi}{2}] \; . \; Sen(\omega_c t + \pi/2) \qquad (4.28)$$

De estas salidas convenientemente filtradas mediante un filtro pasabajos se obtiene:

$$yA_x = Cos[\frac{b_n(t).\delta\phi}{2}] \quad (4.29)$$

$$yB_x = -Sen[\frac{b_n(t).\delta\phi}{2}] \quad (4.30)$$

$$yC_x = 0,707\,[Cos(\frac{b_n(t).\delta\phi}{2}) + Sen(\frac{b_n(t).\delta\phi}{2})] \quad (4.31)$$

$$yD_x = 0,707\,[Cos(\frac{b_n(t).\delta\phi}{2}) - Sen(\frac{b_n(t).\delta\phi}{2})] \quad (4.32)$$

Examinando la figura 4.15 se puede decir que:

- El primer bit será 1 si $yB_x > 0$
- El segundo bit será 1 si $yA_x > 0$
- El tercer bit será 1 si yC_x e yD_x son simultáneamente positivos o negativos.

En la figura 4.16 se representa un demodulador 8-PSK, en el cual no se toma en cuenta el valor de ponderación 0,707 ya que es suficiente con las polaridades obtenidas.

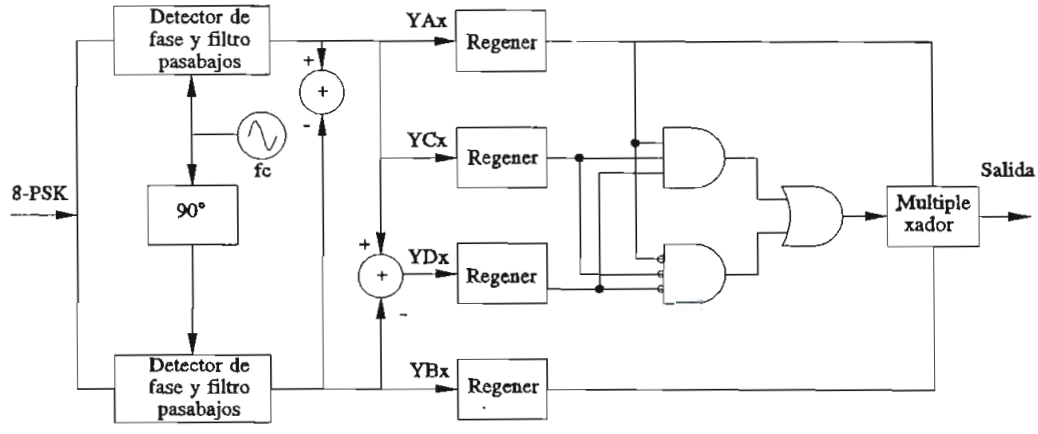


Figura 4.16. Demodulador 8-PSK de dos referencias

4.2.4.5. Modulación 16-PSK

Para la modulación 16-PSK se tiene que M = 16; en este caso las 16 fases diferentes estarán separadas un ángulo $\delta\phi = \pi/8$, representando cada fase un grupo de cuatro dígitos binarios. En la tabla 4.4. se indica la expresión de la portadora correspondiente a las 16 posibles combinaciones de cuatro dígitos binarios, también se indica el valor de las componentes en cuadratura.

Dígitos binarios	a_x	b_x	Portadora modulada
0000	0,980	-0,195	Cos(Wct+π/16)
0001	0,831	-0,555	Cos(Wct+3π/16)
0011	0,555	-0,831	Cos(Wct+5π/16)
0010	0,195	-0,980	Cos(Wct+7π/16)
0110	-0,195	-0,980	Cos(Wct+9π/16)
0111	-0,555	-0,831	Cos(Wct+11π/16)
0101	-0,831	-0,555	Cos(Wct+13π/16)
0100	-0,980	-0,195	Cos(Wct+15π/16)
1000	-0,980	0,195	Cos(Wct-15π/16)
1001	-0,831	0,555	Cos(Wct-13π/16)
1011	-0,555	0,831	Cos(Wct-11π/16)
1010	-0,195	0,980	Cos(Wct-9π/16)
1110	0,195	0,980	Cos(Wct-7π/16)
1111	0,555	0,831	Cos(Wct-5π/16)
1101	0,831	0,555	Cos(Wct-3π/16)
1100	0,980	0,195	Cos(Wct-π/16)

Tabla 4.4. Componentes en cuadratura para 16-PSK

En la figura 4.17 se muestra la modulación 16-PSK y su Diagrama de Constelación.

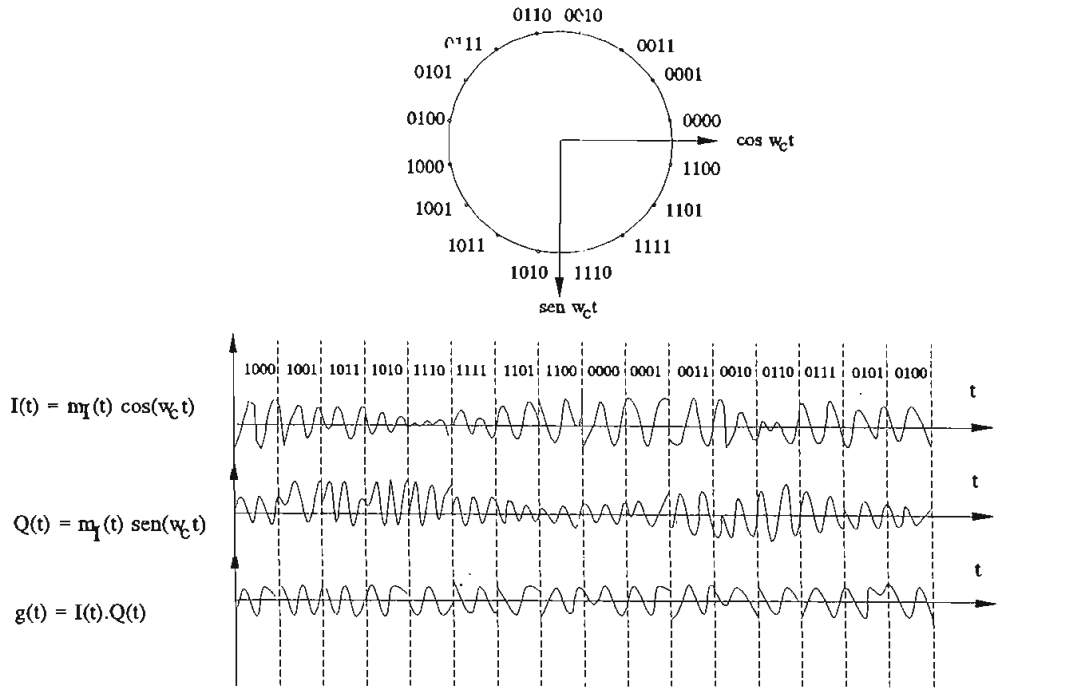


Figura 4.17. Diagrama de constelación y formas de onda 16-PSK

4.2.5. MODULACION MULTISIMBOLICA DE AMPLITUD EN CUADRATURA M-QAM

Como se ha visto en los esquemas PSK todos los puntos de la Constelación se encuentran sobre una circunferencia, lo cual implica una amplitud constante de la señal. Por consiguiente en los sistemas M-PSK al tener dos canales en cuadratura los niveles de las señales modulantes (banda base) en cada canal no son independientes, pues la composición de ambos debía resultar en una señal de amplitud constante.

Si ahora se abandona esta condición y se permite que las señales de banda base en los dos canales en cuadratura sean totalmente independientes, se está en presencia de un esquema denominado **QAM** (Quadrature Amplitude Modulation) o también denominado APK (Amplitud Phase Keying).

Dicho esquema consiste entonces en la modulación multinivel de amplitud de dos portadoras en cuadratura en forma

independiente. En consecuencia los dos canales en cuadratura son completamente independientes incluyendo su codificación en banda base. Los radioenlaces de alta capacidad utilizan modulación digital QAM.

La ecuación matemática que representa la modulación QAM es la siguiente:

$$g(t) = a_i.Cos(\omega_c t) + b_i.Sen(\omega_c t) \qquad (4.33)$$

donde a_i y b_i toman en forma independiente los valores discretos previstos según el número de niveles establecidos, siendo $M = L^2$ (L es el número de niveles de cada canal en cuadratura).

Para el caso especial de dos niveles (± 1) para cada canal, el sistema no es más que el 4-PSK. En la figura 4.18 se representa el esquema donde cada canal en cuadratura puede tomar cuatro niveles distintos lo cual resulta en el denominado 16-QAM, es decir L = 4 y M = 16. Las variables a_i y b_i pueden tener los valores (-3,-1,+3,+1). Cada valor de a_i ó b_i son asociados con un dibit. Aquí se debe asumir la regla de codificación de Gray que dice que dos dibits consecutivos deben variar en un solo bit.

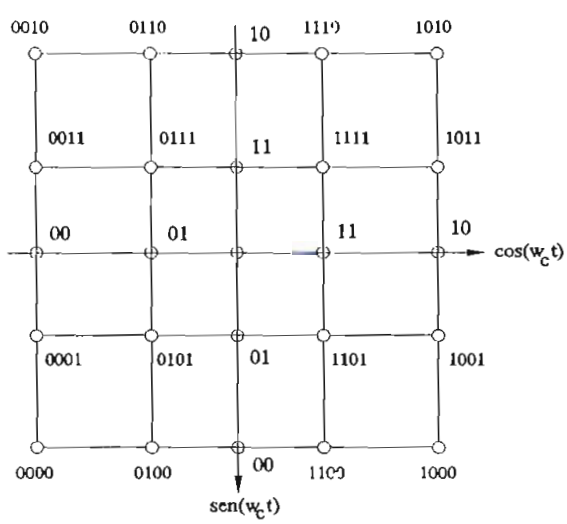


Figura 4.18. Diagrama de Constelación de la modulación 16-QAM

Obsérvese que a diferencia de 16-PSK la envolvente de la portadora modulada ya no es constante.

Por otro lado, la distancia entre puntos en un esquema QAM es siempre mayor que un esquema PSK equivalente para M > 4.

Esto puede verificarse a partir de las ecuaciones:

$$d = Sen(\pi/M) \qquad (4.34)$$

y

$$d = \sqrt{2}/(L-1) \qquad (4.35)$$

Con las ecuaciones 4.34 y 4.35, se calcula la distancia entre puntos adyacentes para los esquemas M-PSK y M-QAM respectivamente. Esta característica tiene una importancia fundamental para ambos sistemas en lo referente al ruido.

El diagrama de bloques de un modulador M-QAM se representa en la figura 4.19.

El flujo de datos de entrada, con una tasa de señalización de $(1/T_o)$ bits/s, se divide en dos flujos de velocidad $(1/2T_o)$. A continuación el conversor de 2 a L niveles transformará cada dos bits a un nivel determinado, tal como se muestra en la figura 4.20.

La señal multinivel presentará una velocidad:

$$(1/T_s) = \frac{1}{2T_o} \cdot \frac{1}{\log_2 L} \ (simb/seg) \qquad (4.36)$$

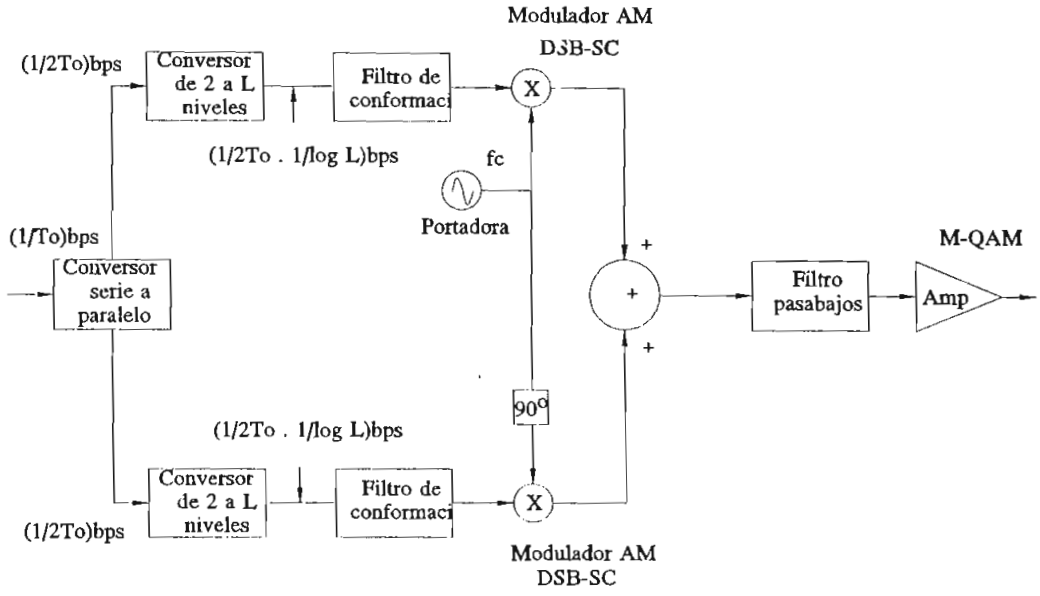


Figura 4.19. Diagrama de bloques de un modulador QAM

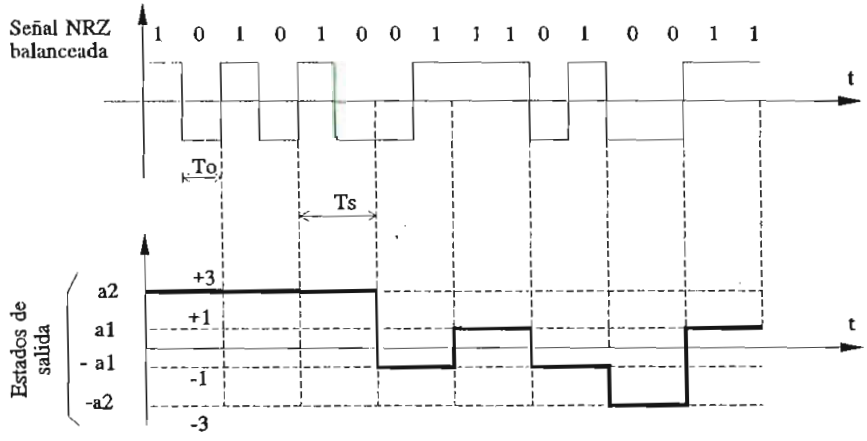


Figura 4.20. Convertidor de 2 bits a 4 niveles para 16-QAM

Si por ejemplo, la fuente de datos tiene una velocidad de $(1/T_o) = 10$ (Mbps), el flujo dividido será 5 (Mbps), y si se usa un sistema 16-QAM con L=4 niveles en cada señal a_i y b_i, su velocidad será:

$$(1/T_s) = \frac{10\ Mbits/seg}{2} \cdot \frac{1}{\log_2 4} = 2.5\ Msimb/seg. \qquad (4.37)$$

El ancho de banda final requerido dependerá del filtro de conformación ubicado a la salida de los conversores de 2 a L niveles.

4.2.5.1. Demodulación M-QAM

El diagrama de bloques de un demodulador M-QAM se muestra en la figura 4.21, donde $M = L^2$. El conversor de L a 2 niveles consiste en L-1 comparadores, los cuales proveen un 1_L a su salida si en el instante de muestreo la señal de entrada supera el nivel de umbral preestablecido. En caso contrario se tendrá un 0_L. Las salidas de los L-1 comparadores se conectan a un circuito lógico el cual finalmente determinará la "palabra" binaria de salida correspondiente. Para M = 16 se tiene **L** = 4 y un dibit de salida por cada instante de muestreo del conversor **L** a 2.

El muestreo se realiza a $(1/2T_o).(1/(2\log_2 L))$, y su tasa de salida binaria es $(1/2T_o)$. Finalmente el conversor paralelo-serie multiplexa los dos flujos de datos proveyendo la salida deseada a $(1/T_o)$ bps.

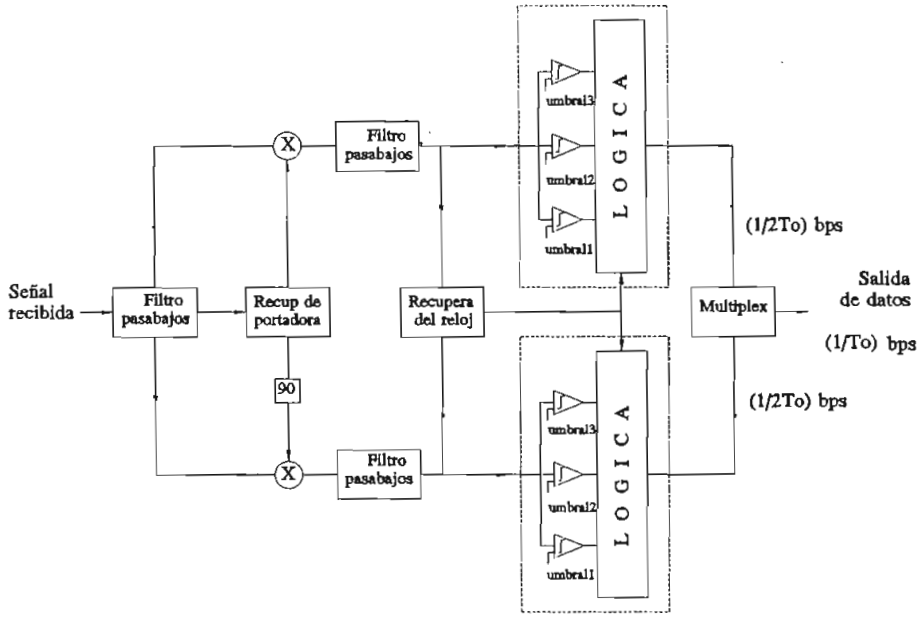
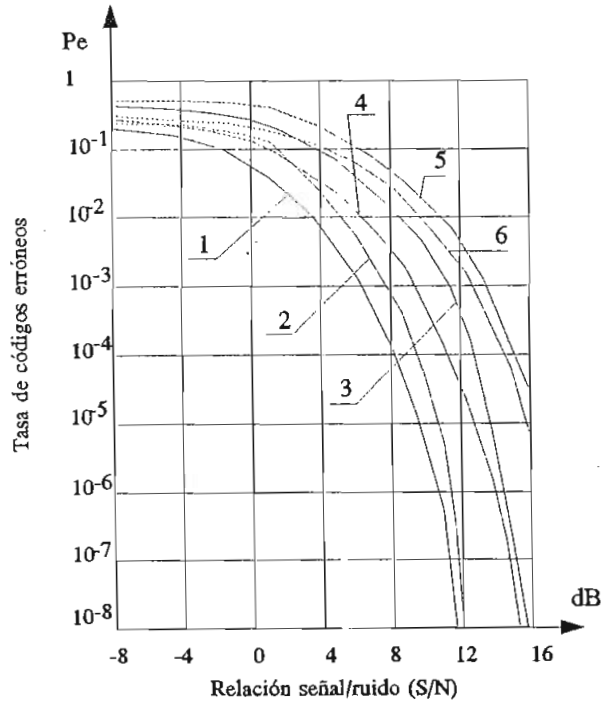


Figura 4.21. Diagrama de bloques de un demodulador M-QAM

4.3. COMPARACION DE LA EFICIENCIA DE LAS TECNICAS DE MODULACION DIGITAL

4.3.1. Comparación entre las tasas de error

En la figura 4.22 se muestran las curvas de las tasas de error, siendo la relación señal/ruido el parámetro del eje horizontal. De esta figura se nota que el método de demodulación que da la mejor tasa de error para una P_e dada, es la detección coherente PSK. Entre los métodos de detección coherente, para obtener una tasa de error igual que PSK, la FSK requiere 3 dB más en la relación S/N que PSK, en tanto que ASK requiere 6 dB más.



1: detección coherente PSK
2: detección DPSK
3: detección no coherente FSK
4: detección coherente FSK
5: detección no coherente ASK
6: detección coherente ASK

Figura 4.22. Tasas de error de señal binaria

En cuanto a la modulación multisimbólica se puede calcular la tasa de error bajo ciertas condiciones. En la figura 4.23 se muestran curvas de la tasa de error correspondientes a las diferentes modulaciones con detección coherente PSK.

En este gráfico se puede apreciar según se aumenta el número de niveles, aumenta también la relación S/N requerida para una tasa de error dada; sin embargo también aumenta la magnitud de información conténida en un símbolo; por tanto en el caso de que la velocidad de información sea fija, la modulación multisimbólica requerirá de menor velocidad de transmisión de símbolos que la binaria.

La velocidad de transmisión de símbolos es uno de los parámetros que deciden la forma de la envolvente del espectro de la onda portadora. Entonces disminuir la velocidad significa que la banda requerida puede ser más estrecha. Es decir el ancho de banda del caso de la modulación M-aria se reduce en $1/(\log_2 M)$ en comparación con el caso de la modulación binaria, y por tanto, puesto que el espectro de ruido es plano, la relación S/N se mejora en proporción a la disminución del ancho de banda.

En los métodos de transmisión multisimbólica, la modulación QAM da menor tasa de error que la PSK.

En la figura 4.24 se muestra las tasas de errores para los diversos métodos de modulación y demodulación en cuanto a la transmisión multisimbólica 16-PSK, 16-QAM y 16-ASK.

En esta figura se ve claramente la superioridad de la modulación 16-QAM respecto a 16-PSK y 16-ASK en donde la 16-PSK requiere 4 dB más en la S/N que 16-QAM para obtener una tasa de error igual, en tanto que 16-ASK necesta 8 dB más.

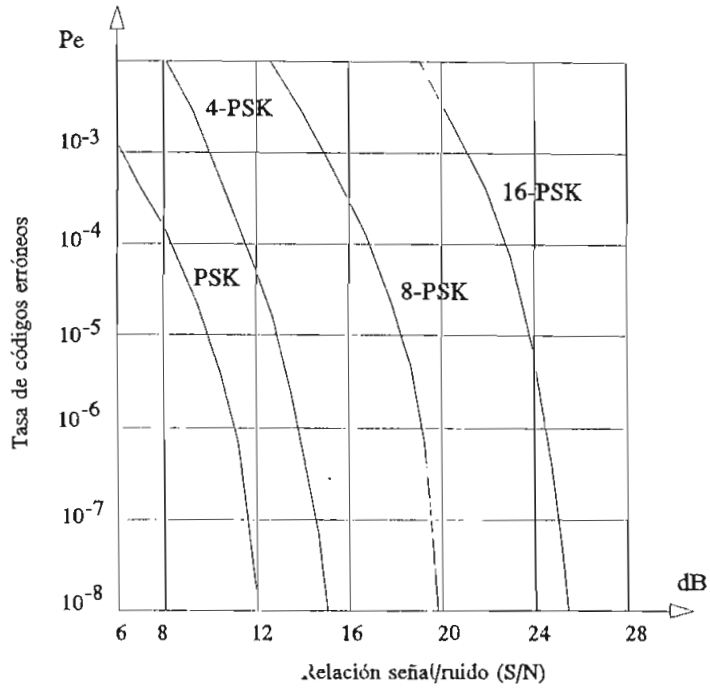


Figura 4.23. Tasas de error para diversos modos de detección coherente PSK

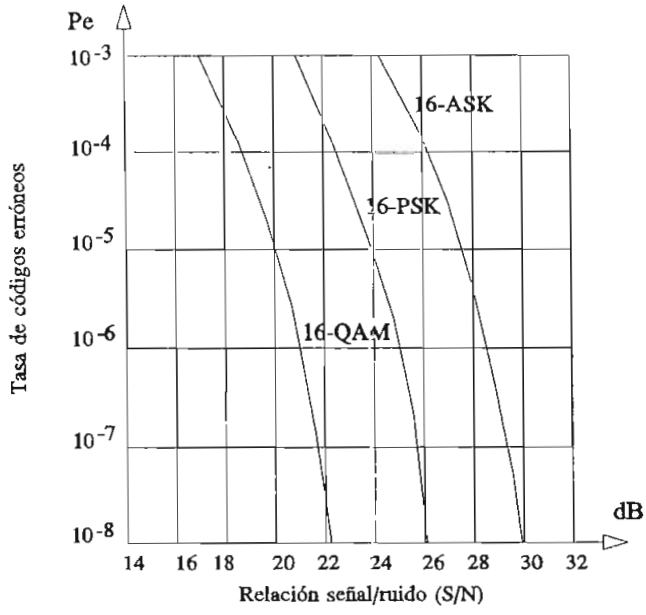


Figura 4.24. Tasas de códigos erróneos para diversos modos de modulación de 16 estados

4.3.2. Análisis de la eficiencia de los esquemas de modulación

Las diferentes formas de modulación tienen su aplicación práctica de acuerdo a diferentes factores a determinar en la transmisión como son: Ancho de banda disponible, velocidad de transmisión, defensa contra ruido, grado y tipo de perturbaciones de línea que deberán soportar, sencillez en los moduladores y demoduladores que incidirá en el costo de los mismos, etc. Los métodos representativos de modulación digital se ordenan en la figura 4.25 de acuerdo a la complejidad inherente del equipo. Sin embargo, una elección específica puede depender de otros factores.

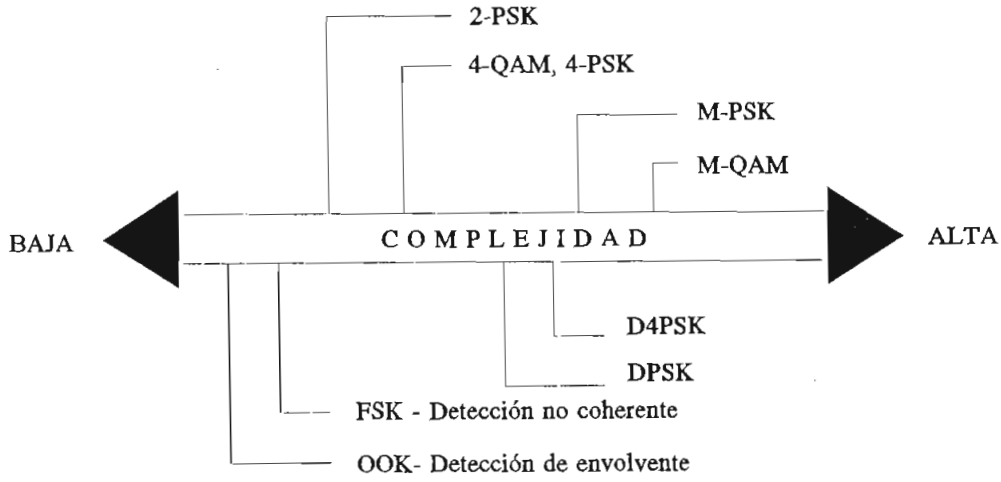


Figura 4.25. Complejidad relativa de esquemas de modulación representativos

En la figura 4.26 se compara la eficiencia para las diferentes modulaciones M-PSK (línea entrecortada) en presencia de ruido, para una probabilidad de error $P_e = 10^{-4}$. En ordenadas se expresa la relación velocidad de transmisión/ancho de banda en (bps/Hz) y en abscisas la relación señal a ruido (S/N).

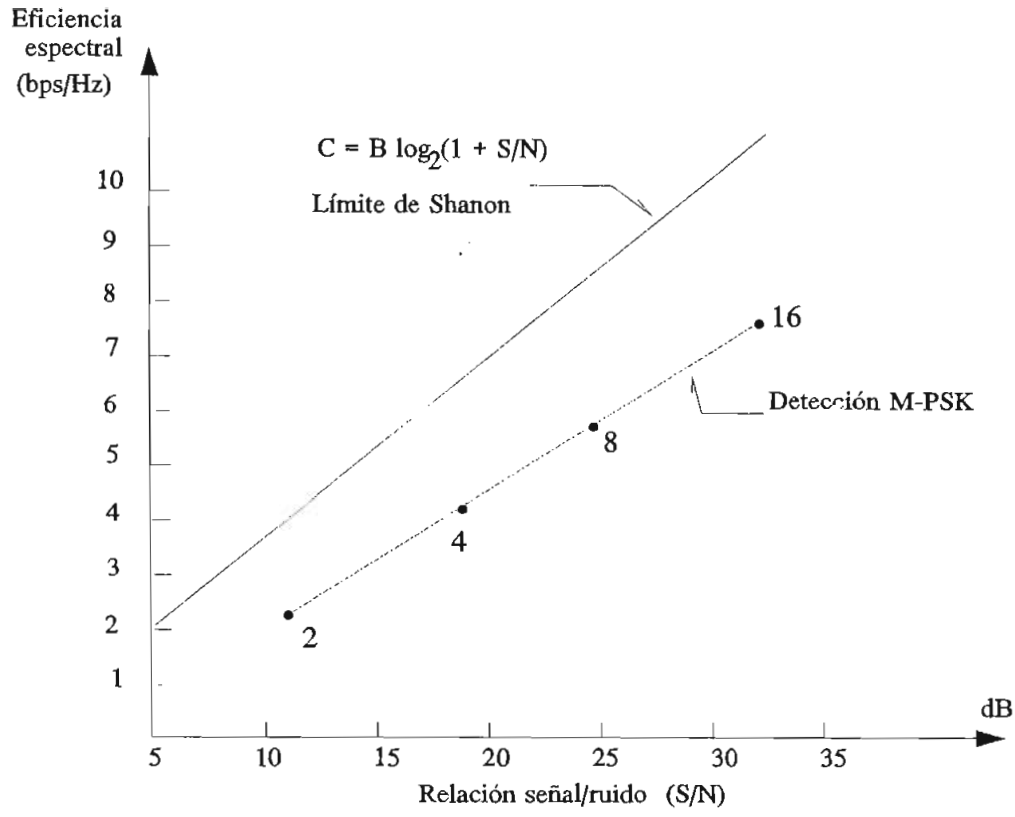


Figura 4.26. Eficiencia espectral de los sistemas de modulación M-PSK en función de S/N

La línea continua corresponde a la máxima capacidad de transmisión de información dada por Shannon.

Comparando las modulacicnes 4-PSK con 2-PSK se ve que la primera requiere de 8 dB más en la S/N que la segunda, mientras que la 8-PSK requiere de 14 dB más que 2-PSK, de lo que se puede concluir que a medida que se utiliza un esquema de modulación de orden superior se incrementa la relación señal a ruido para mantener la misma probabilidad de error con respecto al sistema 2-PSK.

4.4. PROGRAMA DE SIMULACION DE MODULACION DIGITAL BINARIA ASK, FSK Y 2-PSK.

El programa de Simulación de Modulación Digital Binaria, es un conjunto de subprogramas o procedimientos que permiten visualizar en la pantalla del computador las señales principales que intervienen en el proceso de modulación; éstas son: la señal de datos binaria, la señal portadora y la señal modulada.

En la simulación realizada no se ha tomado en cuenta la relación entre las frecuencias de la portadora y la de los bits, debido a limitaciones de graficación. Por esta razón se ha establecido una relación de frecuencias que permitan graficar y comprender los métodos de modulación en forma sencilla.

El programa desarrollado tiene procedimientos similares para las modulaciones ASK, FSK y 2-PSK. Inician con la lectura de cuántos y cuáles son los bits que van a modular la señal portadora; con estos datos se llama al procedimiento **GraficarBits** que permite visualizar la secuencia de los bits en la parte superior de la pantalla. Luego el procedimiento **GraficarPortadora** dibuja en la parte media de la pantalla la señal portadora.

A continuación, dependiendo del parámetro de variación (amplitud, frecuencia o fase) se asigna a la variable **k** el valor correspondiente de amplitud, fase o frecuencia, que permitirá graficar la función correspondiente a cada modulación y a cada símbolo o bit.

En la figura 4.27 se presenta el diagrama de flujo de la modulación ASK, (procedimiento **ASK1**); en este procedimiento **k** toma los valores de 0 a 1 para los símbolos 0_L y 1_L respectivamente el mismo que será utilizado en la modulación OOK:

$$Y = k \cos (G) \qquad (4.38)$$

En esta expresión Y es la señal modulada, k es el bit de información y cos (G) corresponde a la portadora.

Para el caso de la modulación FSK, dado por el procedimiento **FSK1**, el diagrama de flujo está en la figura 4.28. En éste se observa que se grafican las dos señalesportadoras, es decir, se utiliza el método de las dos portadoras para generar la señal modulada FSK. La variable k tomará entonces los valores correspondientes a las dos frecuencias, W_0 para el caso de 0_L, y de W_1 para el caso de 1_L, multiplicadas por el tiempo.

La señal modulada Y que se grafica en este caso es:

$$Y = \cos\ (k) \qquad (4.39)$$

donde k como se indica toma los valores de W_0t ó W_1t dependiendo del valor del bit de dato.

De forma similar a las anteriores se puede observar en la figura 4.29 el diagrama de flujo del procedimiento **PSK1** para la modulación de fase 2-PSK. La variable k corresponde ahora a la fase de la señal, para cero y para uno, la ecuación que se tiene para graficar es, entonces:

$$Y = \cos\ (G + k), \qquad (4.40)$$

que corresponde a la modulación de fase 2-PSK.

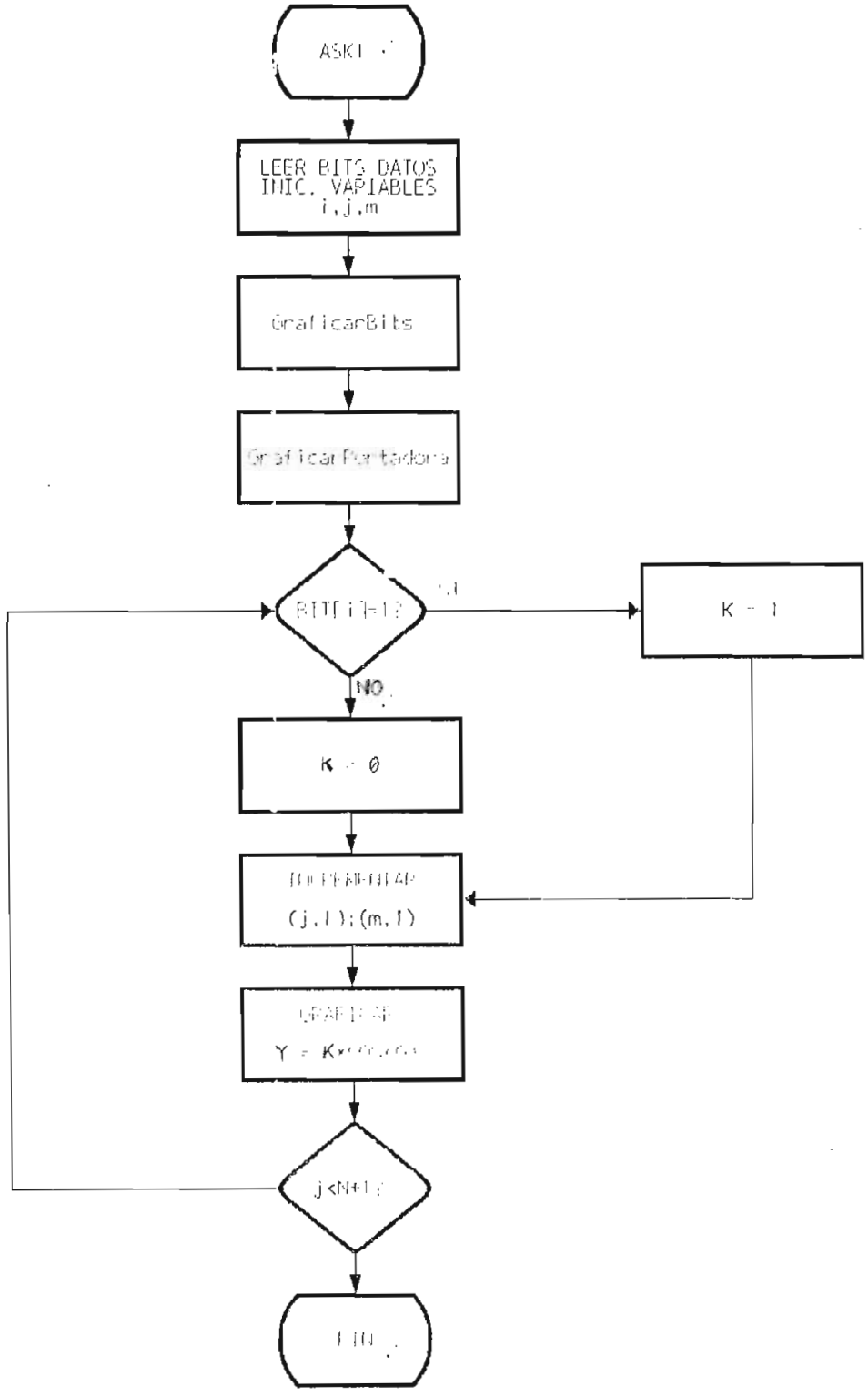


Figura 4.27. Diagrama de flujo para la modulación ASK

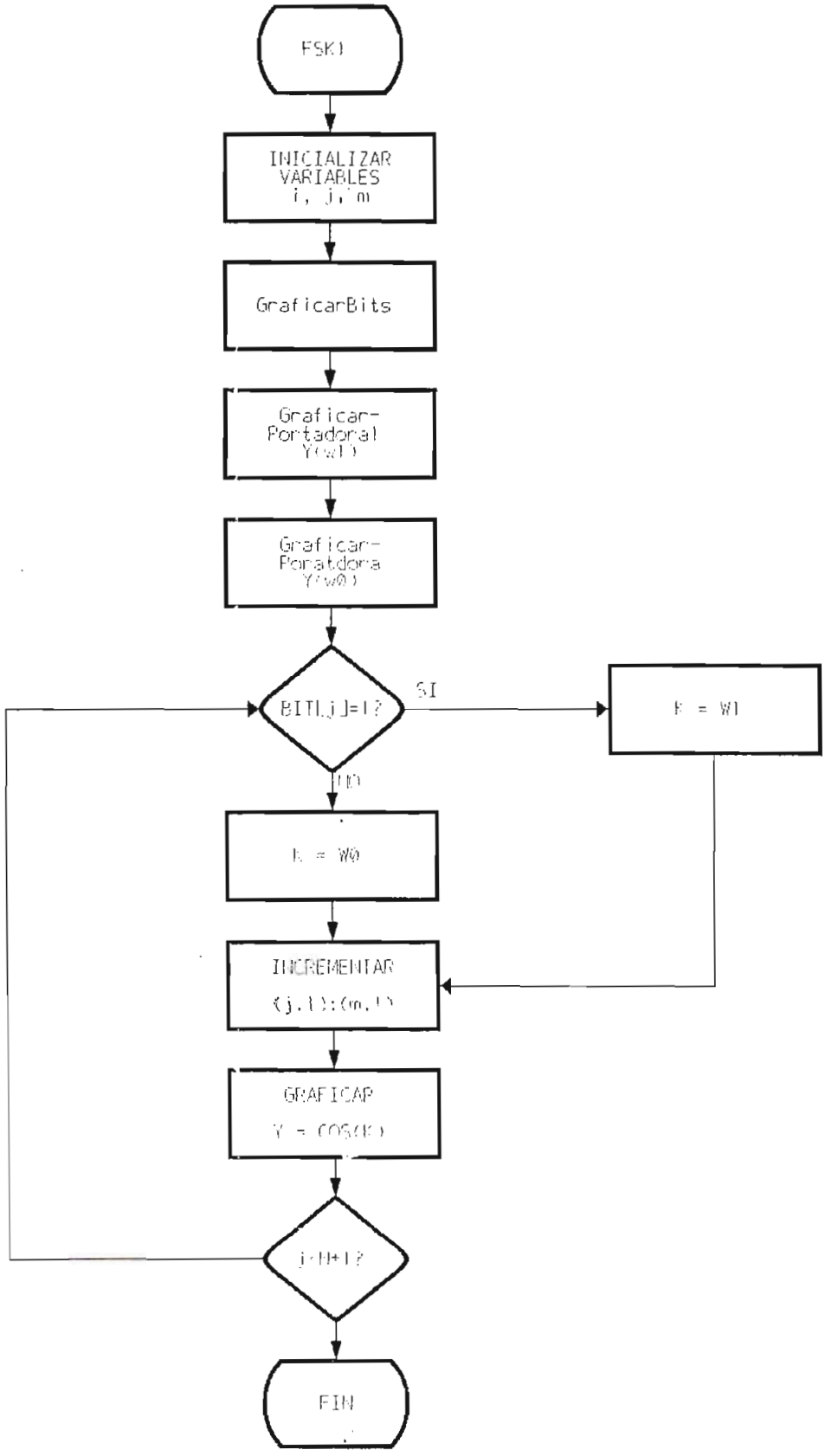


Figura 4.28. Diagrama de flujo para la modulación FSK

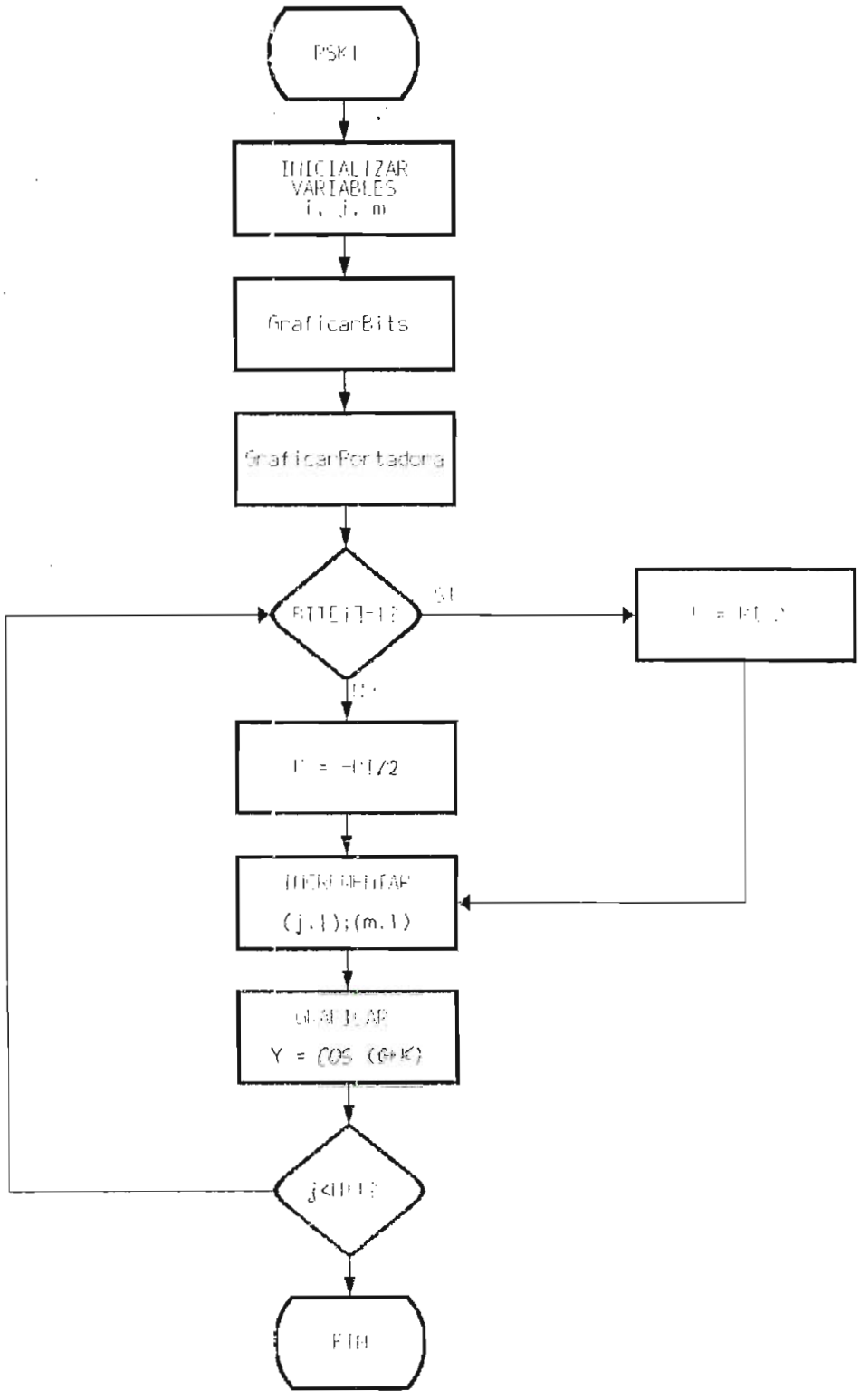


Figura 4.29. Diagrama de flujo para la modulación 2-PSK

4.5. PROGRAMA DE SIMULACION DE MODULACION DIGITAL BINARIA MULTISIMBOLICA M-PSK Y M-QAM

4.5.1. Algoritmo para modulación M-PSK

Para elaborar la simulación de las modulaciones M-PSK y M-QAM, se toman como base los procedimientos o subprogramas de las modulaciones binarias.

En los procedimientos realizados, se agrupan los bits de acuerdo con el tipo de modulación así:

2 bits para modulaciones 4-PSK y 4-QAM
3 bits para modulación 8-PSK
4 bits para modulaciones 16-PSK y 16-QAM

Como se analizó, en el numeral 4.2.4, en las modulaciones M-PSK, la fase de la portadora toma uno de los M valores posibles que dependen de las combinaciones de los dígitos binarios.

Por lo tanto a cada una de las combinaciones de dígitos, les corresponden las fases que están en las tablas de las componentes en cuadratura para 4-PSK, 8-PSK y 16-PSK.

Una vez que se asigna la fase a la secuencia, se procede a graficar la función

$$Y = \cos\ (G + k) \qquad (4.41)$$

donde Y es la señal modulada, G es la fase de referencia de la portadora y k la fase asignada.

Los diagramas de flujo de los procedimientos para las modulaciones 4-PSK, 8-PSK y 16-PSK, están en las figuras 4.30, 4.31 y 4.32 y corresponden a los procedimientos o subrutinas **M4PSK1**, **M8PSK1** y **M16PSK1** respectivamente.

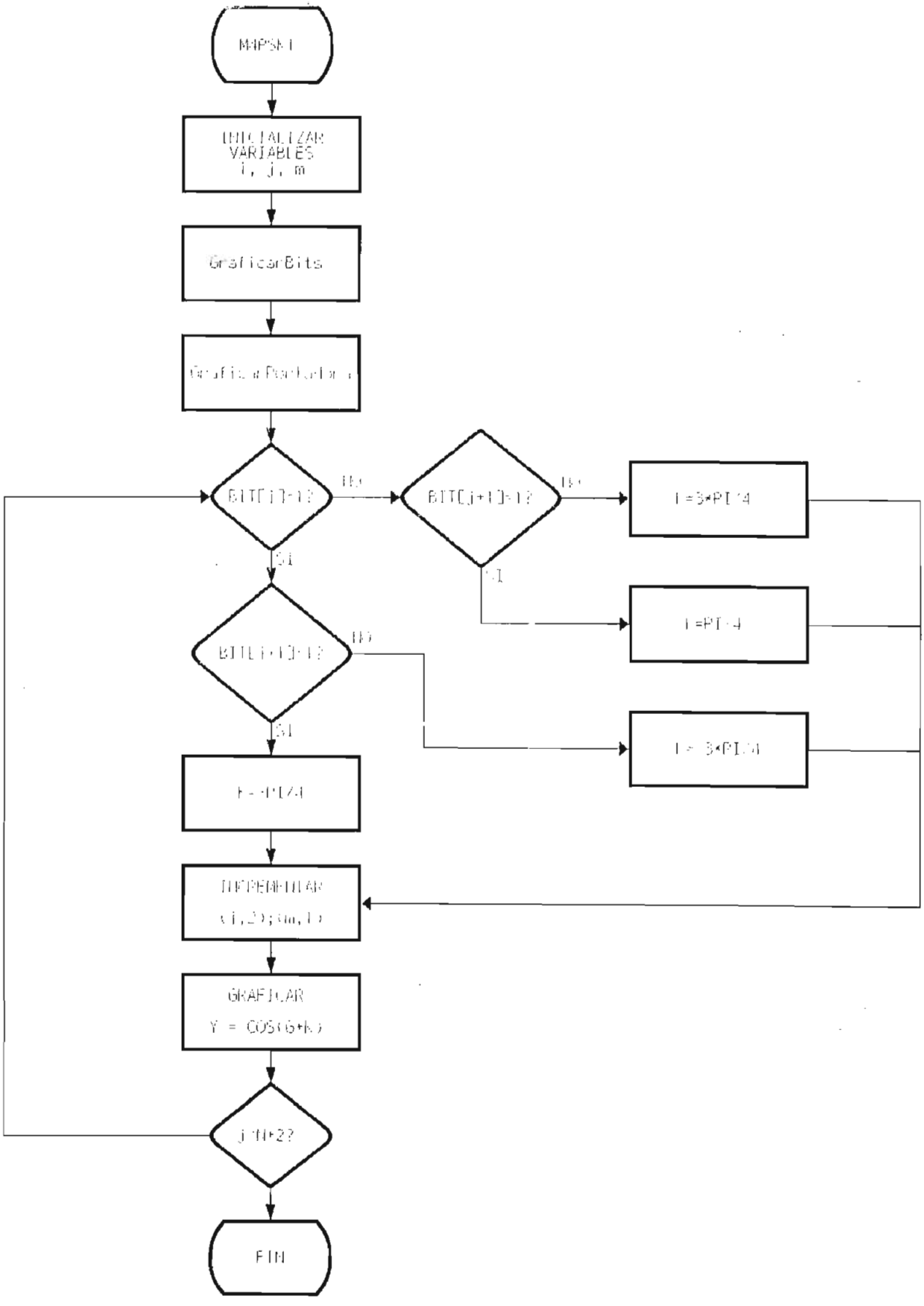


Figura 4.30. Diagrama de flujo para la modulación 4-PSK

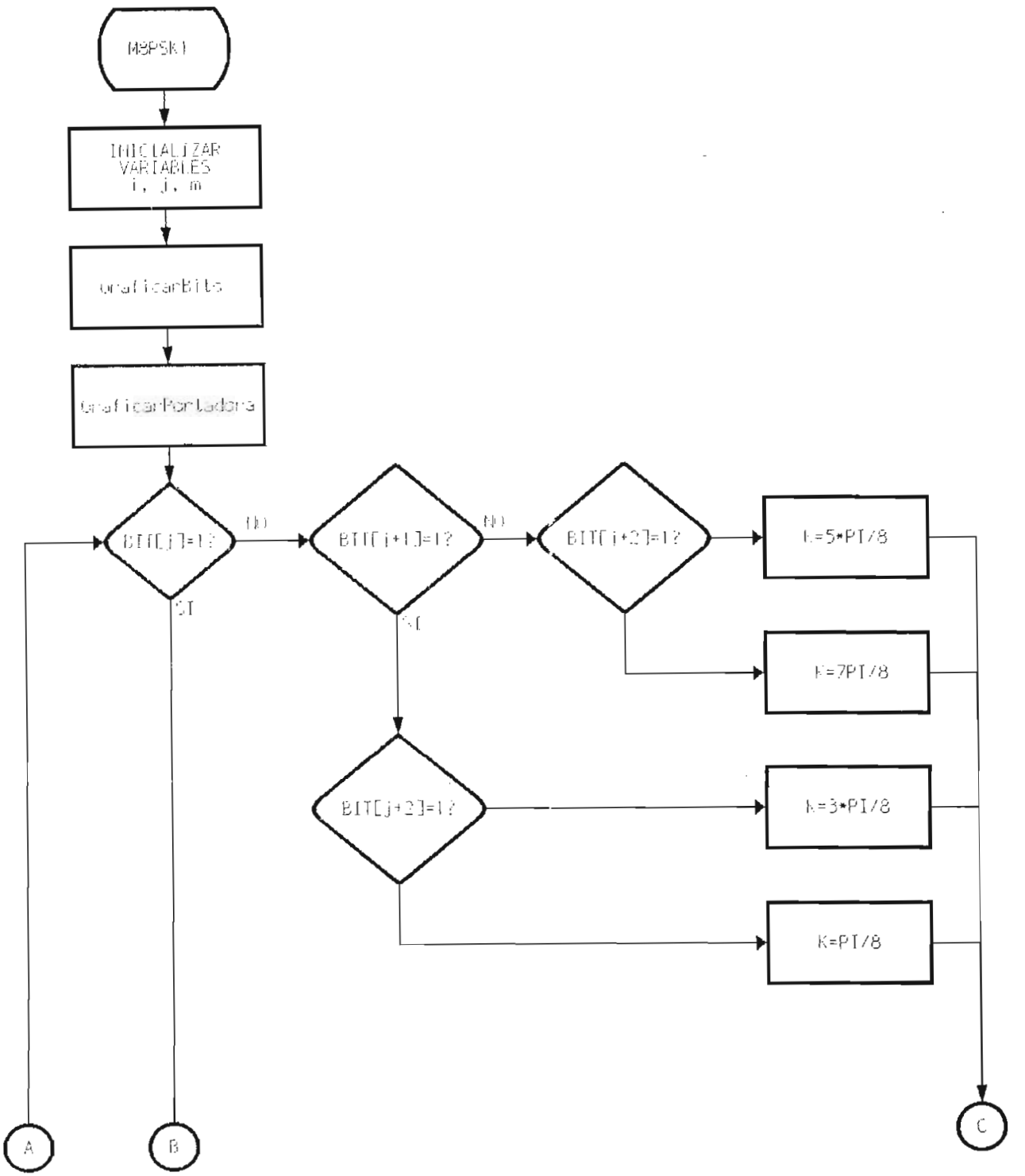


Figura 4.31. Diagrama de flujo para la modulación 8-PSK

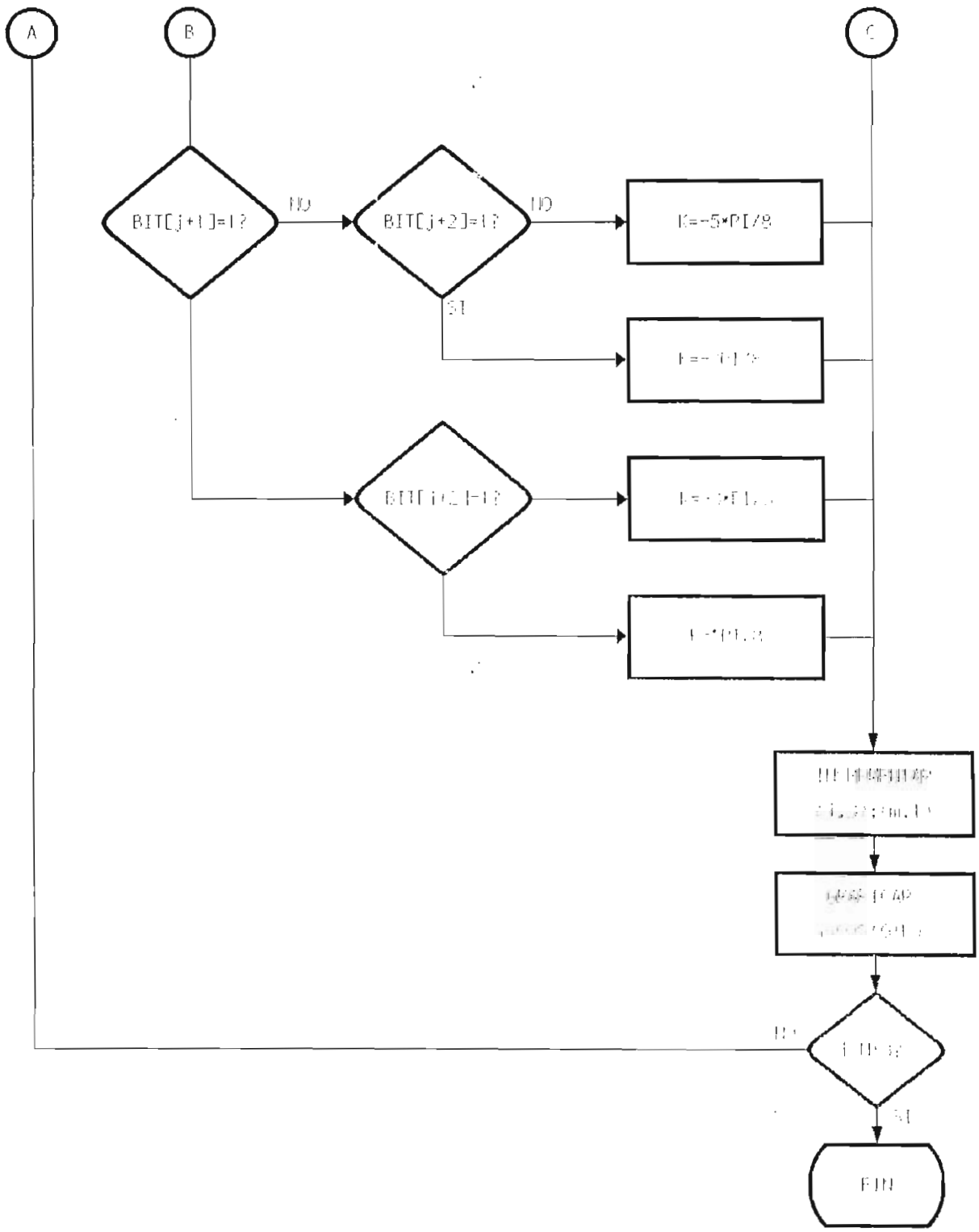


Figura 4.31. Diagrama de flujo para la modulación 8-PSK (cont)

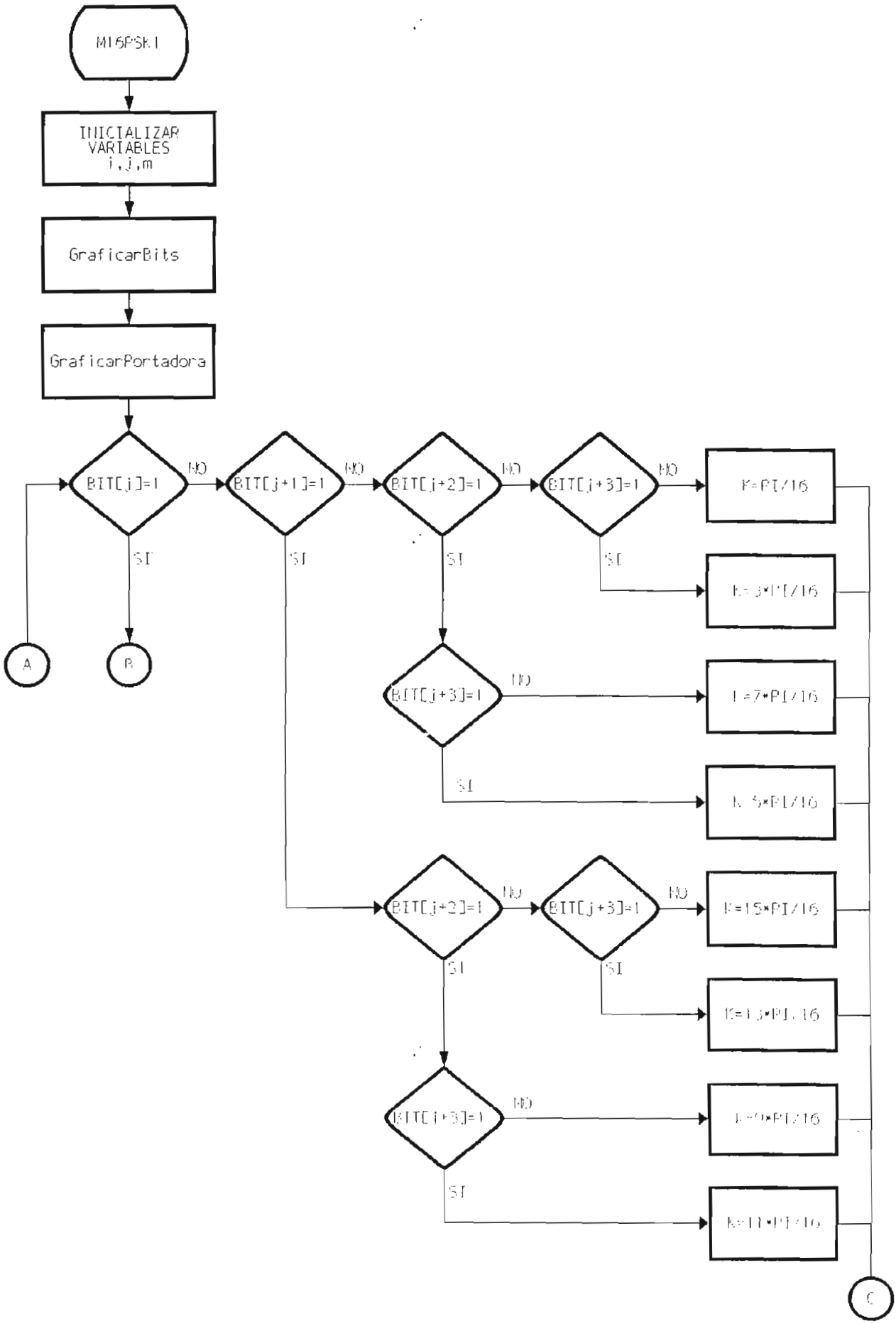


Figura 4.32. Diagrama de flujo para la modulación 16-PSK

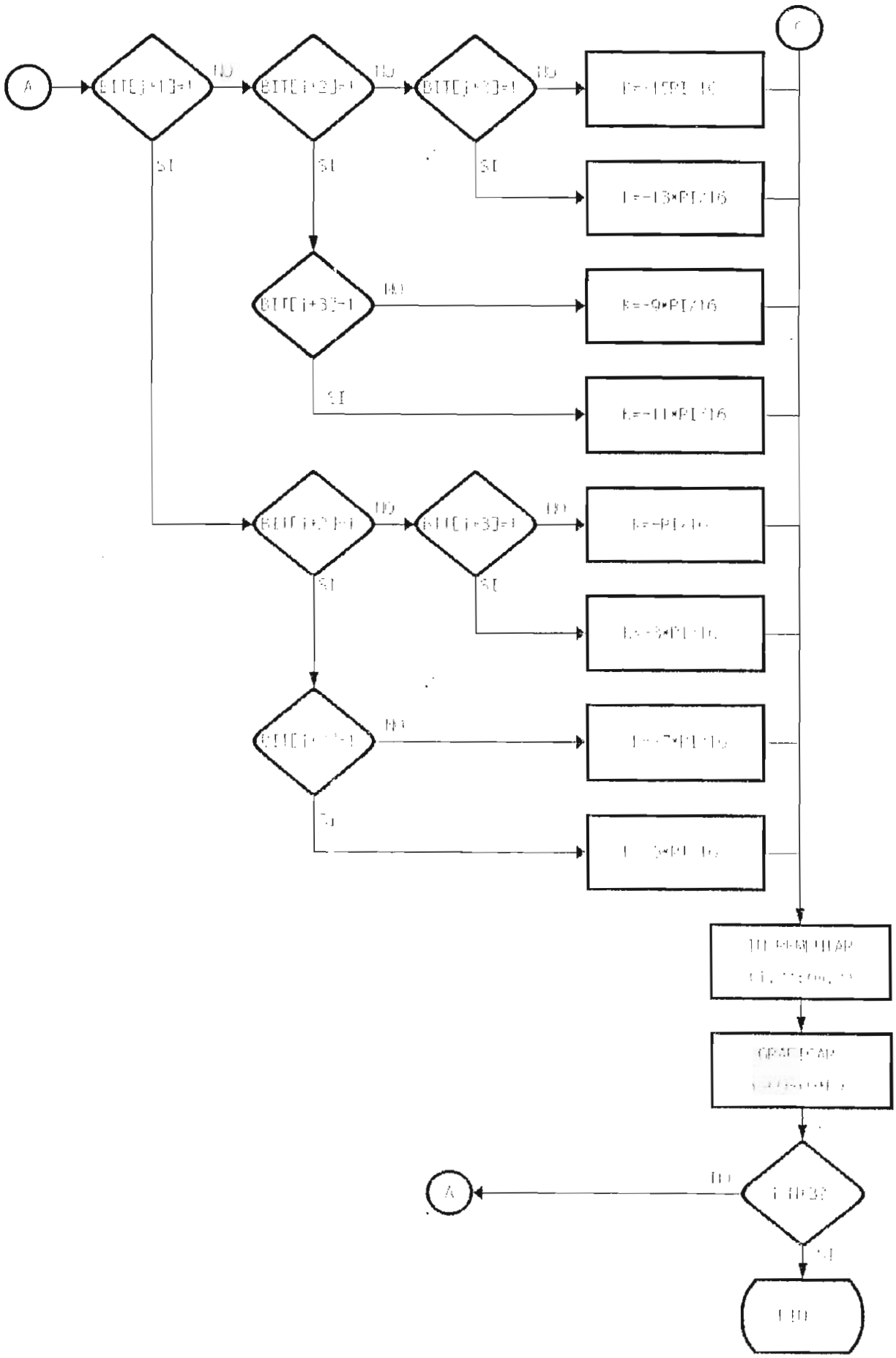


Figura 4.32. Diagrama de flujo para la modulación 16-PSK (cont)

4.5.2 Algoritmo para Modulación M-QAM

En los sistemas de modulación M-QAM la amplitud y la fase de la onda portadora varían simultáneamente, por lo que en este caso para cada combinación de dígitos binarios le corresponde diferentes valores de amplitud y fase.

Los valores de las amplitudes y fases para las combinaciones de los bits se los pueden obtener de los diagramas de constelación de los sistemas 4-QAM y 16-QAM.

Las variables utilizadas son k para la fase de la señal y k_1 para la amplitud correspondiente y la función que se grafica en estos casos es:

$$Y = K_1 \cos (G + k) \qquad (4.42)$$

El diagrama de flujc para el procedimiento de modulación 4-QAM está en la figura 4.33 y en la figura 4.34 se tiene el de la modulación 16-QAM.

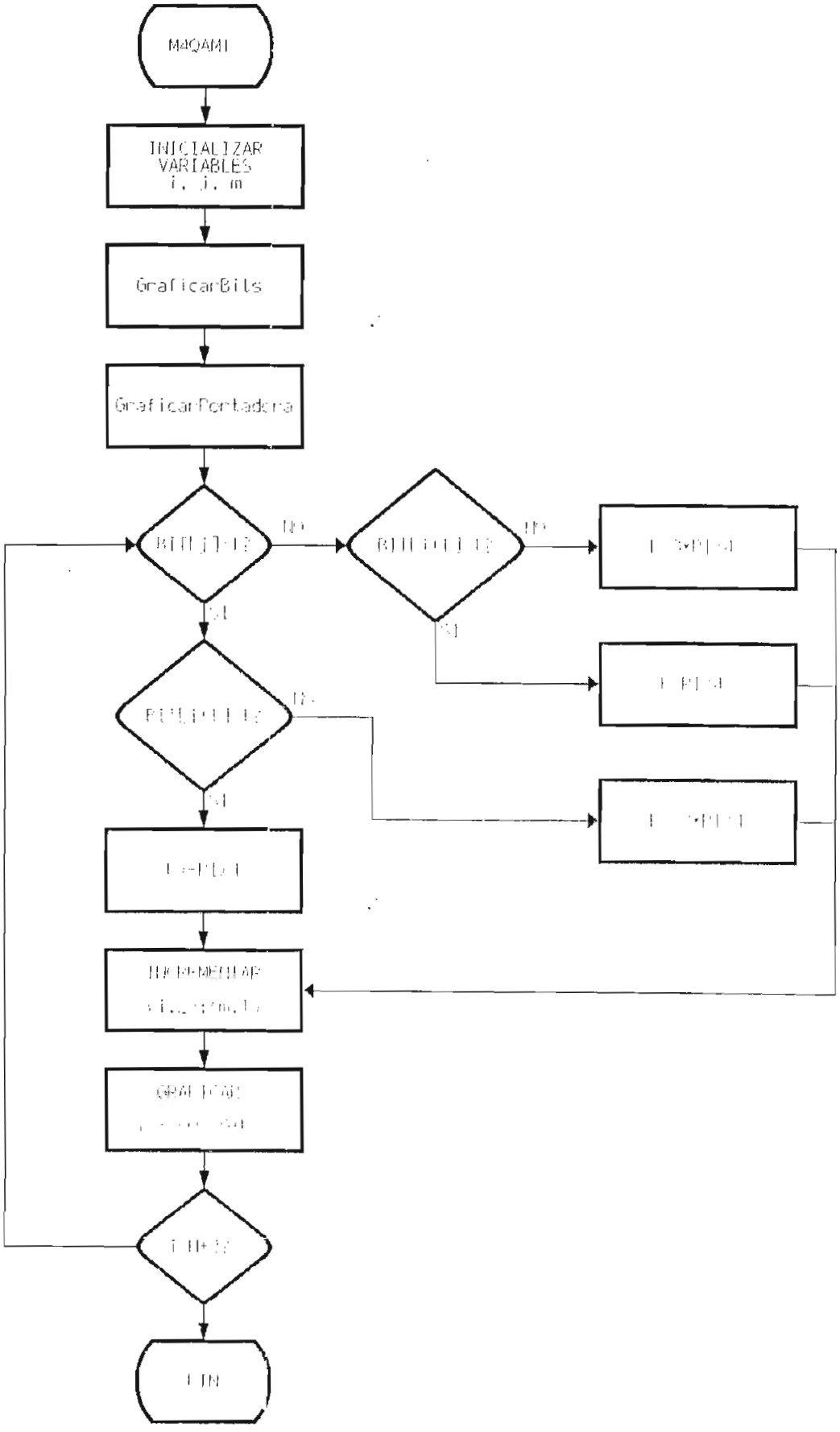


Figura 4.33. Diagrama de flujo para la modulación 4-QAM

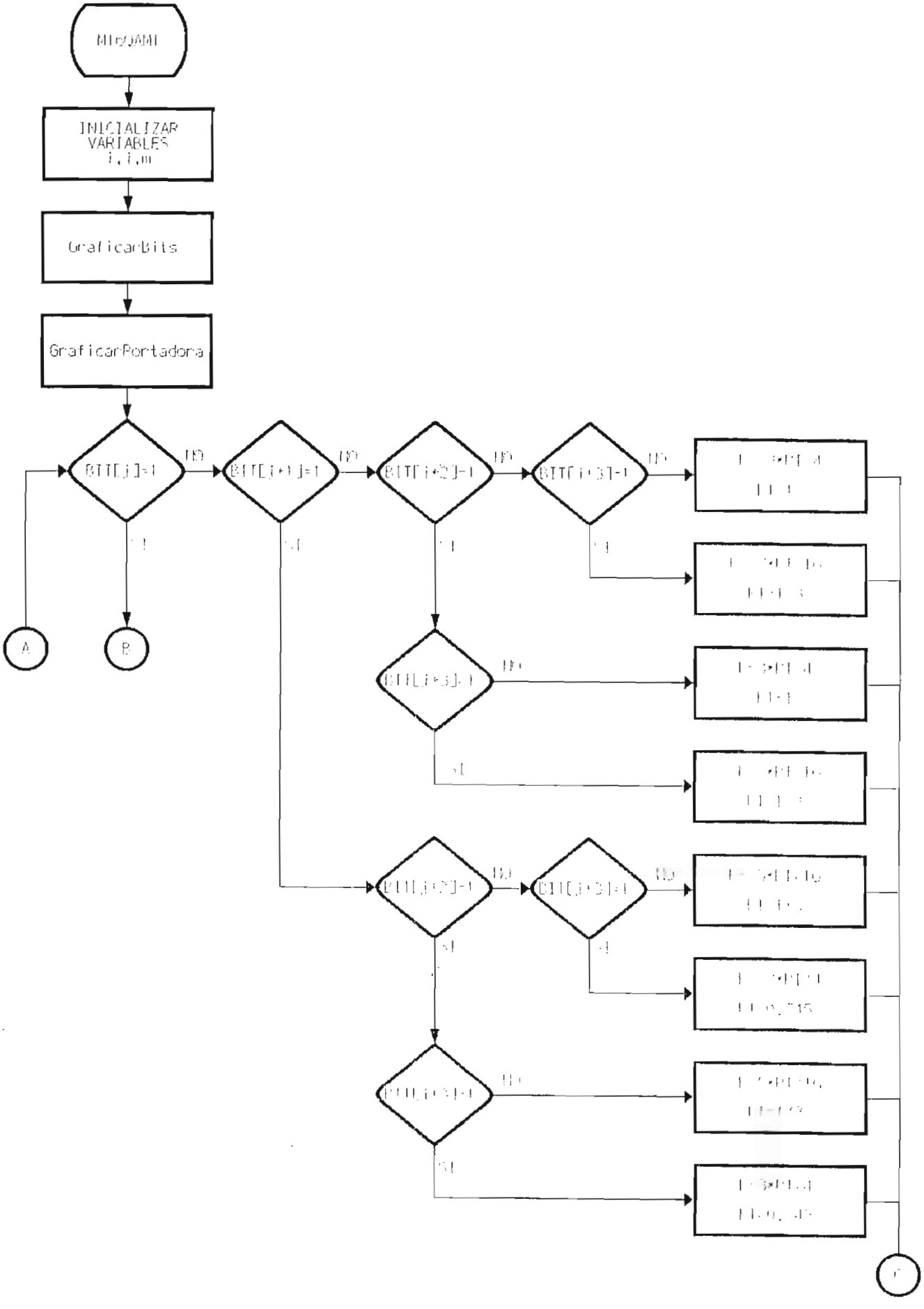


Figura 4.34. Diagrama de flujo para la modulación 16-QAM

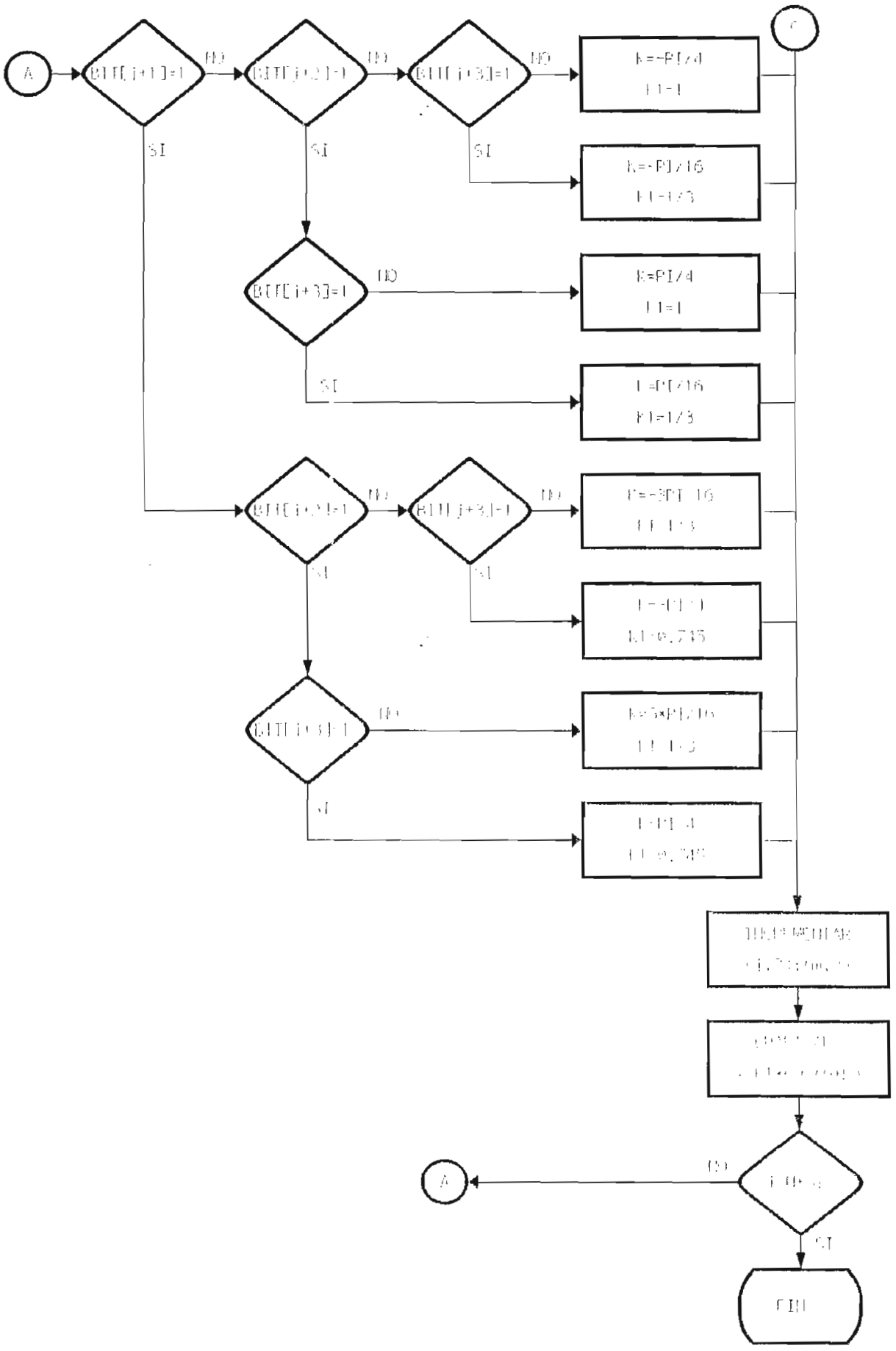


Figura 4.34. Diagrama de flujo para la modulación 16-QAM (cont)

4.6. EJEMPLOS DE APLICACION DE MODULACION DIGITAL BINARIA

A continuación se presentan los ejemplos de modulación digital binaria, que se obtuvieron al ejecutar el programa para la parte correspondiente a Modulación. En la parte superior de los gráficos de las modulaciones ASK, PSK y las multisimbólicas se puede observar la señal binaria (tren de datos), en la parte intermedia está la señal portadora, y en la parte inferior se tiene la representación de la señal modulada.

Para FSK se tiene en la parte intermedia las dos portadoras pertenecientes a las dos frecuencias de modulación.

En cada uno de los ejemplos representados se pueden apreciar las características más importantes de cada uno de los tipos de modulación. Para estos ejemplos se ingresó la secuencia de bits siguiente:

Secuencia de bits: 1 1 1 0 0 1 0 1 1 0

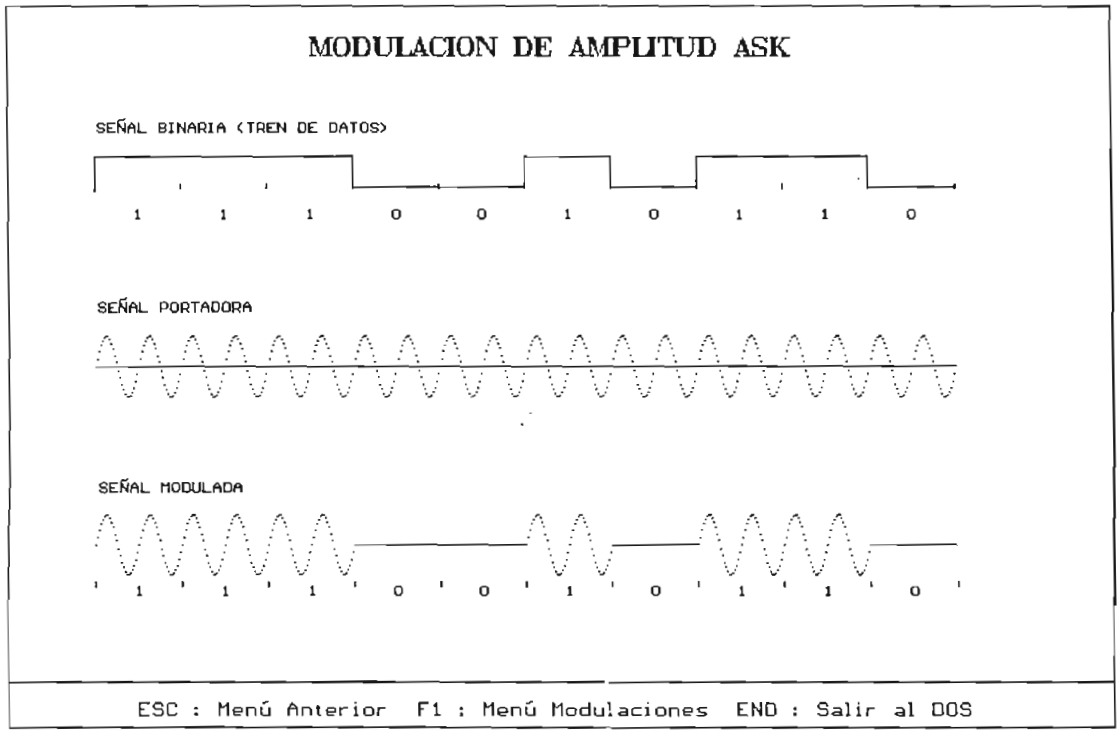


Figura 4.35. Ejemplo de Modulación ASK

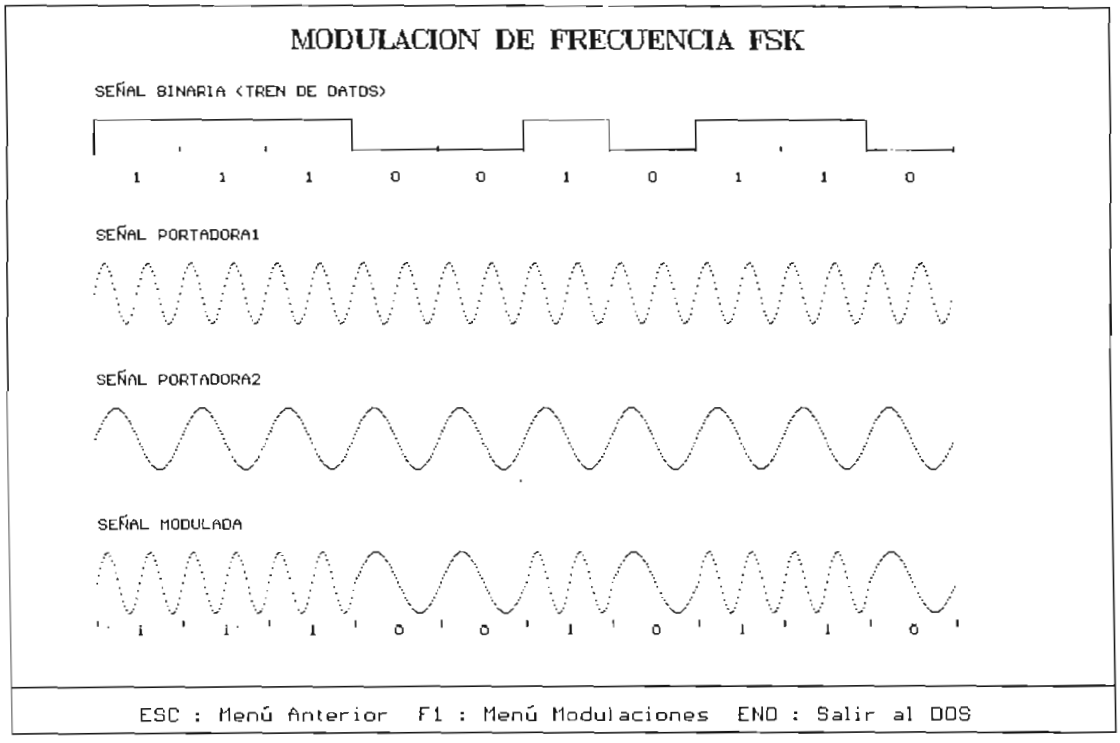


Figura 4.36. Ejemplo de Modulación FSK

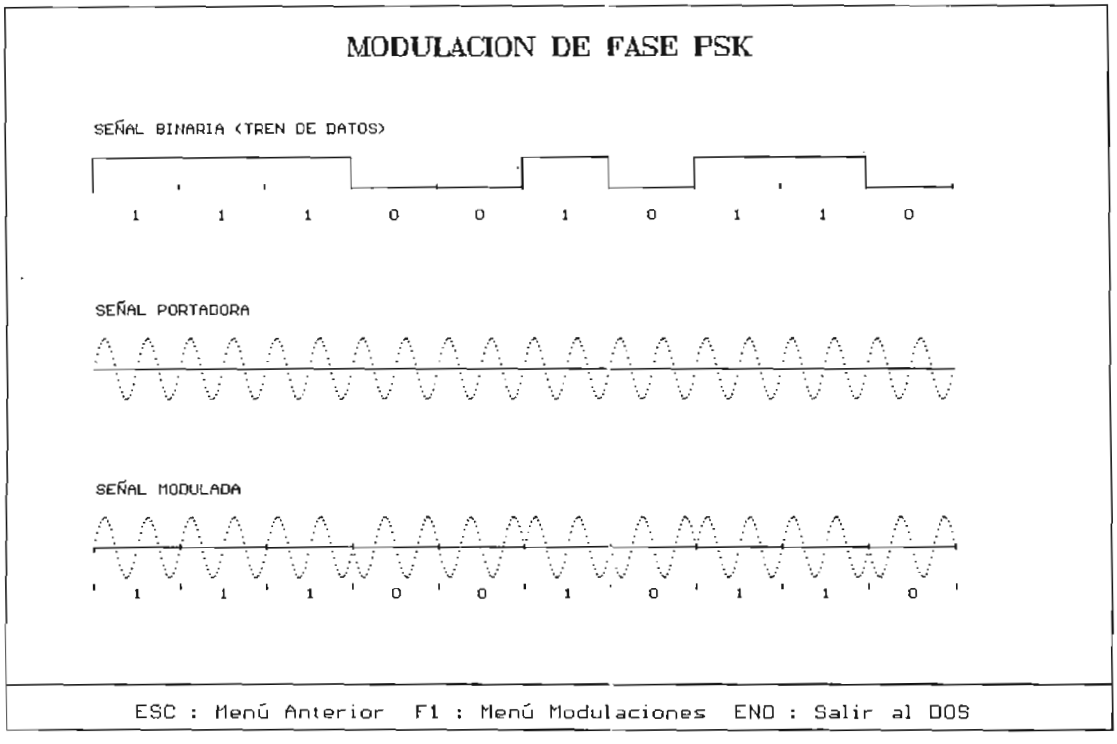


Figura 4.37. Ejemplo de Modulación PSK

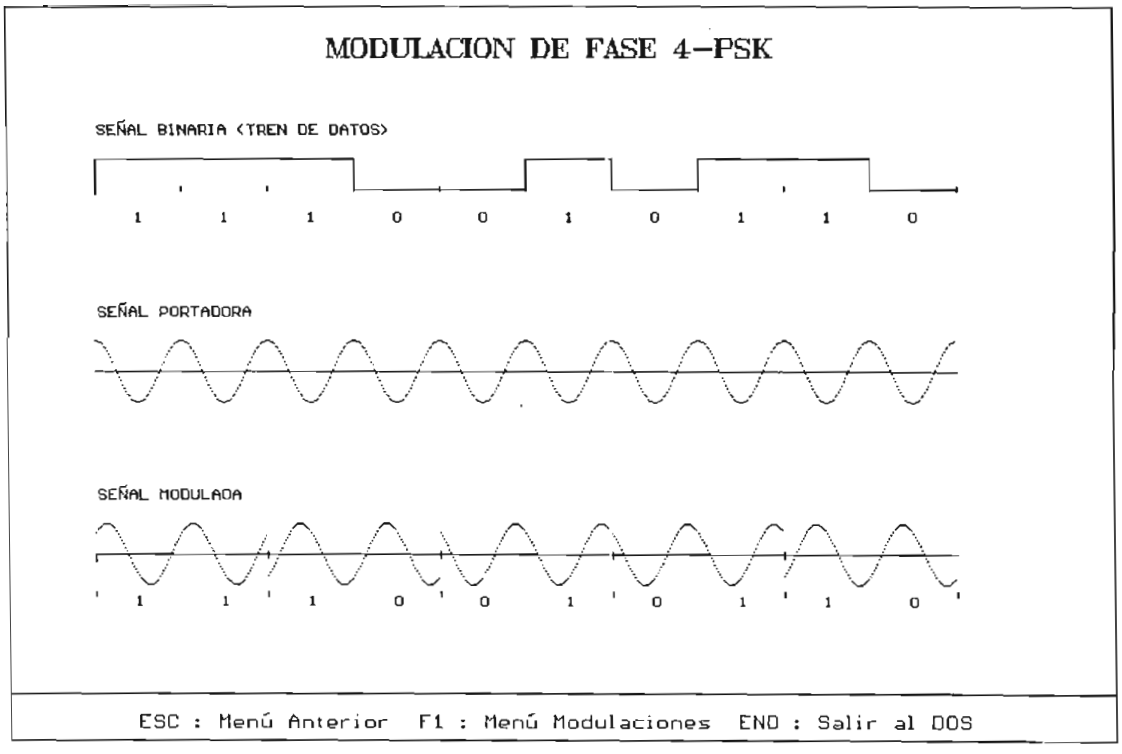


Figura 4.38. Ejemplo de Modulación 4-PSK

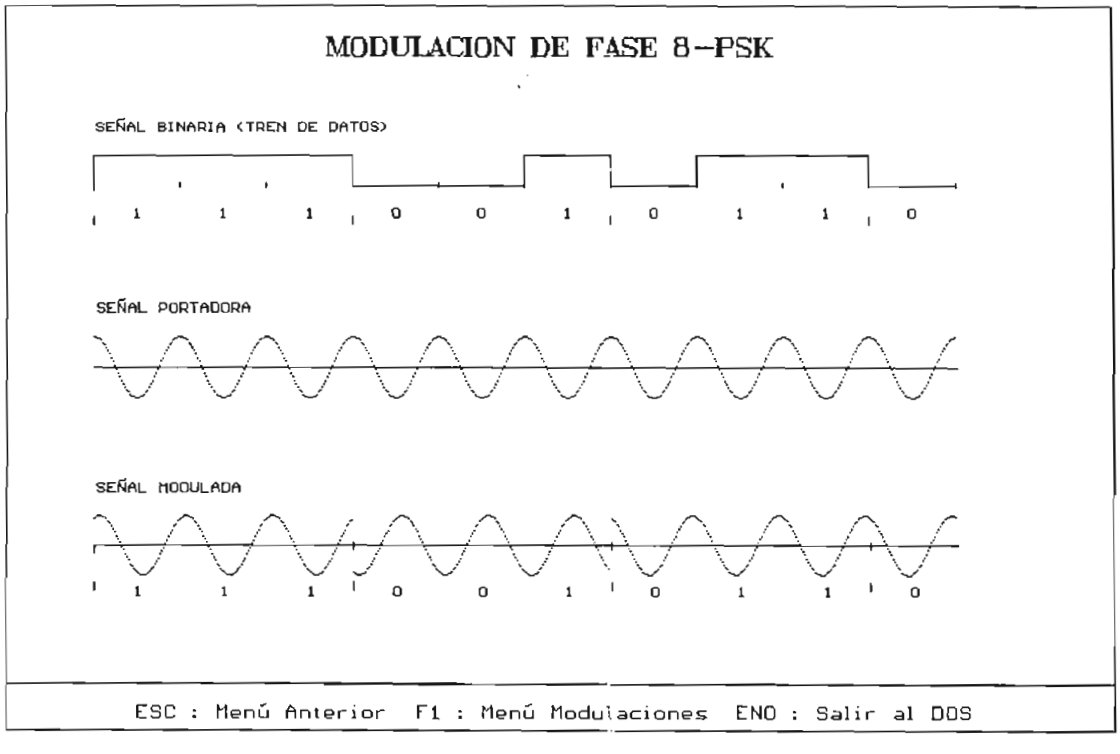


Figura 4.39. Ejemplo de Modulación 8-PSK

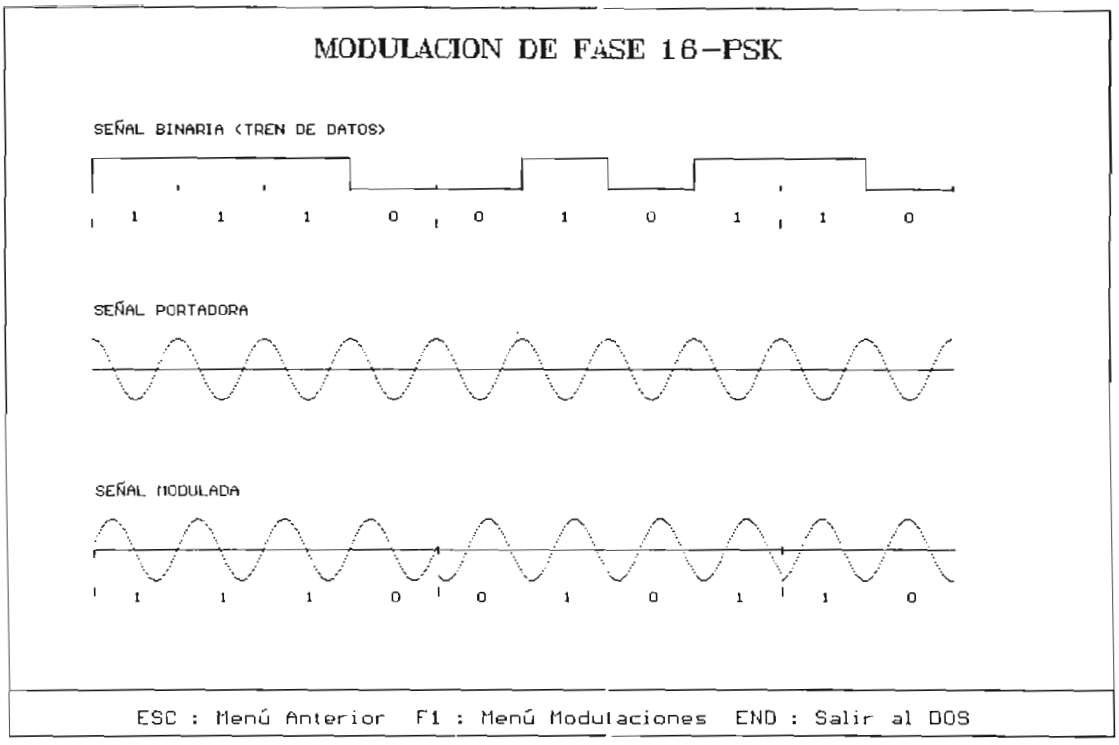


Figura 4.40. Ejemplo de Modulación 16-PSK

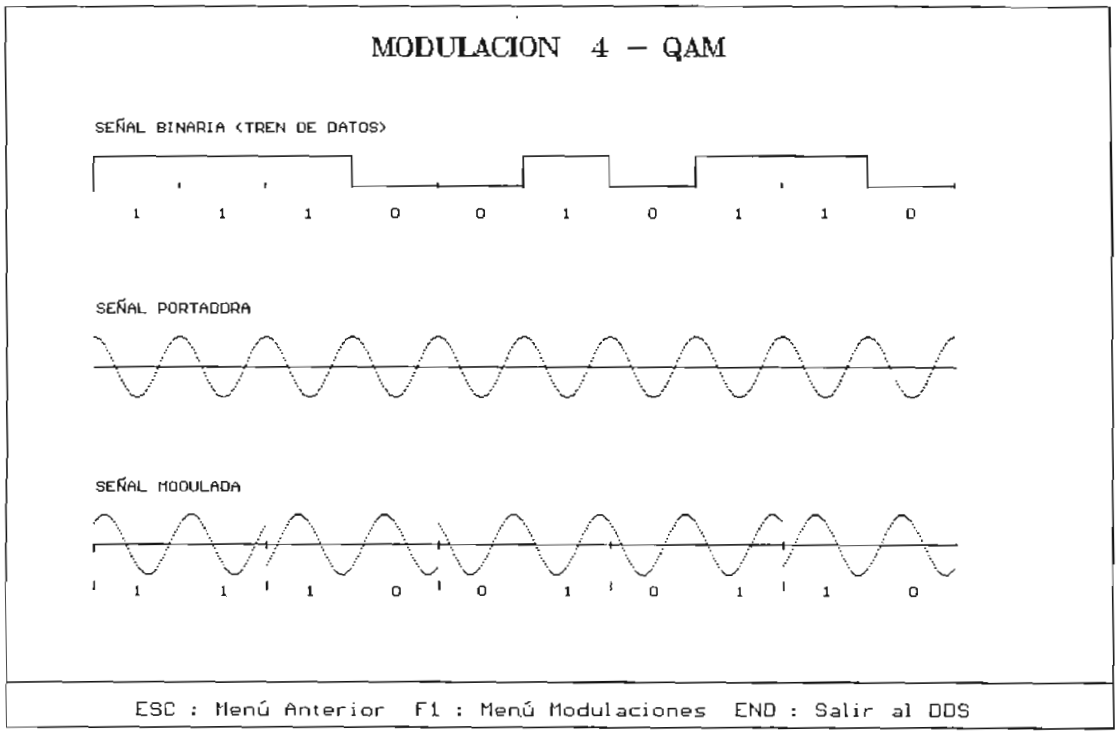


Figura 4.41. Ejemplo de Modulación 4-QAM

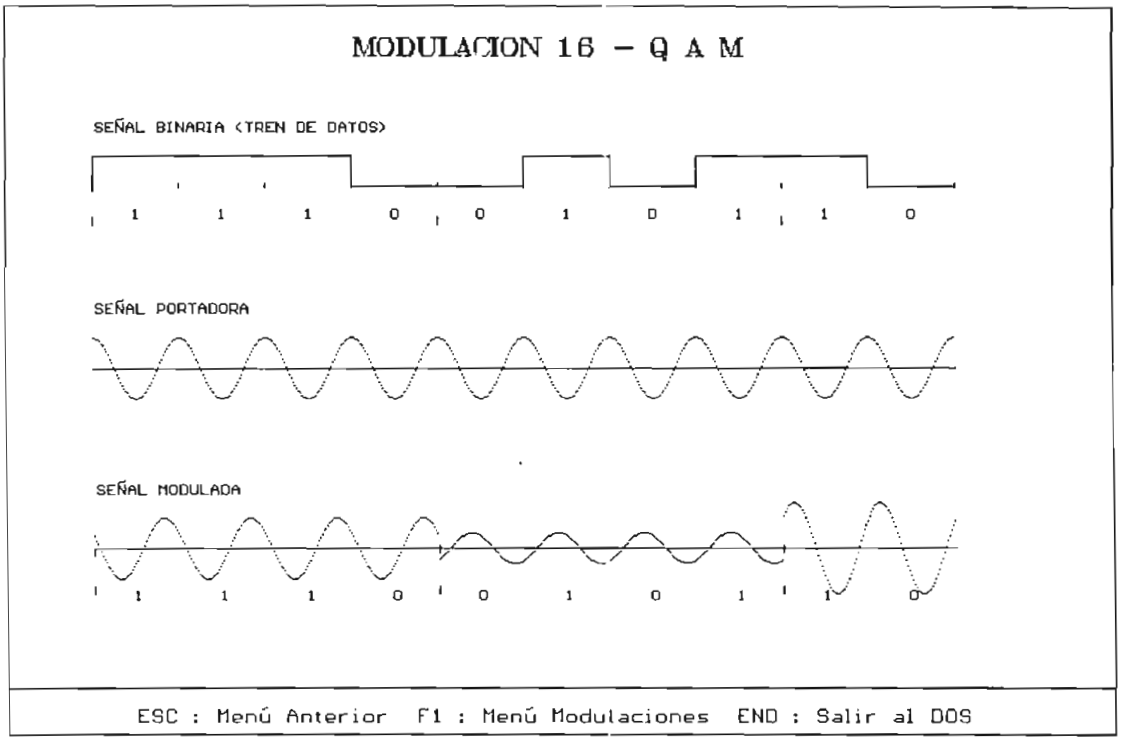


Figura 4.42. Ejemplo de Modulación 16-QAM

CAPITULO 5

DENSIDAD ESPECTRAL DE POTENCIA DE LOS CODIGOS DE LINEA Y SEÑALES MODULADAS

5.1. INTRODUCCION

La densidad espectral de potencia **DEP** de los códigos de línea, en un sistema de transmisión depende del código utilizado. La elección del código está condicionada a las características de la transmisión.

Una de las características que debe tener un código de línea es una densidad espectral de potencia favorable, es decir el espectro de la señal debe igualarse a la respuesta de frecuencia del canal. Si un canal posee alta atenuación en las frecuencias más bajas, el espectro de la señal debe tener una DEP pequeña dentro de este rango para evitar excesiva distorsión de la señal. Es también deseable tener DEP cero cuando $\omega = 0$ (DC), ya que el acoplamiento a CA (corriente alterna), se utiliza en los repetidores regenerativos.

Considérese el tren de pulsos rectangulares de la figura 5.1.(a), estos pulsos tienen un ancho t_o y se repiten cada intervalo T_o con amplitudes arbitrarias. Las señales NRZ, polares y bipolares son casos especiales de este tren de pulsos. Por lo tanto, analizando su DEP, se puede llegar a las DEP de los diversos códigos.

Desafortunadamente la desventaja es que este proceso estaría restringido a una sola forma de pulso rectangular; para ello se usa un artificio el cual considera el tren de pulsos $x(t)$ de la figura 5.1.(b) con impulsos que se repiten cada T_o, siendo a_k la intensidad del impulso en kT_o.

Si $x(t)$ se aplica a la entrada de un filtro que tiene una respuesta de impulso unitario $p(t)$, entonces la salida $y(t)$ será una señal similar al tren de pulsos de la figura 5.1.(a), pero con los pulsos rectangulares reemplazados por pulsos de la forma $p(t)$ como los de la figura 5.3.(b), debido a la respuesta del filtro. Los pulsos se repiten cada T_o segundos, y el pulso en kT_o es $a_k\, p(t)$. Conforme a lo estudiado en el Capítulo 2, la DEP de $y(t)$, es:

$$S_y(\omega) = |P(\omega)|^2\, S_x(\omega)$$

donde: $S_x(\omega)$ es la DEP de la señal de entrada x(t)

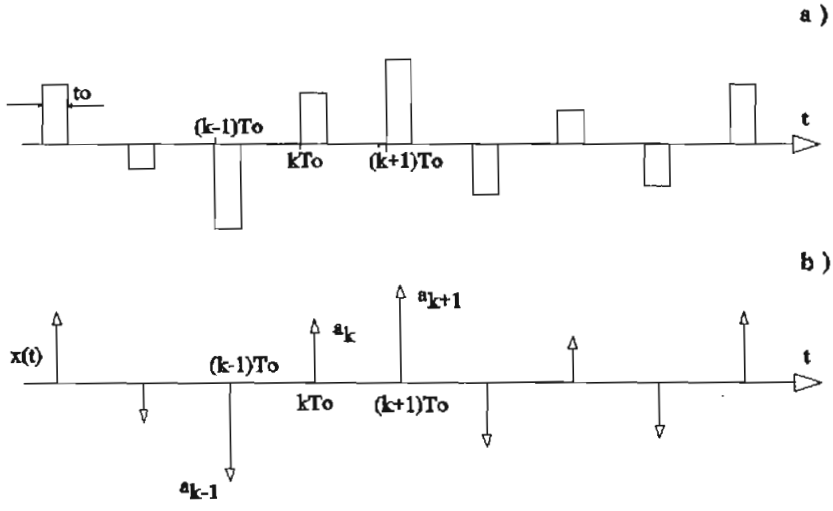


Figura 5.1.a) Pulsos rectangualres
b) Tren de impulsos $x(t)$

Para encontrar la DEP de $y(t)$ se necesita derivar a $\Re_x(\tau)$, que es la función de autocorrelación en el tiempo del tren de impulsos $x(t)$. Esto se puede hacer de manera conveniente, si se considera los impulsos como una forma limitante de los pulsos rectangulares, como se muestra en la

figura 5.2.(a). Cada pulso tiene un ancho $\epsilon \to 0$ y la altura del k-ésimo pulso es h_k. Ya que la intensidad del k-ésimo impulso es a_k.

$$h_k \epsilon = a_k$$

Si se designa al tren de pulsos rectangulares correspondiente como $\hat{x}(t)$, entonces por definición [ec. (2.33.c), del Capítulo 2],

$$\Re_{\hat{x}}(\tau) = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} \hat{x}(t)\,\hat{x}(t-\tau)\,dt \qquad (5.1)$$

Ya que $\Re_{\hat{x}}(\tau)$ es una función par de τ [ec. (2.33.d)], se necesita considerar sólo la τ positiva. En principio se considera el caso de $\tau < \epsilon$.

Aquí la integral en la ecuación (5.1) es el área bajo la señal $\hat{x}(t)$ multiplicada por $\hat{x}(t)$ retardada por τ, en la que $(\tau<\epsilon)$.

Como se observa en la figura 5.2.(b), el área asociada con el k-ésimo pulso es $h_k^2\,(\epsilon-\tau)$, entonces:

$$\Re_{\hat{x}}(\tau) = \lim_{T\to\infty} \frac{1}{T} \sum_k^{\infty} h_k^2\,(\epsilon-\tau)$$

$$= \lim_{T\to\infty} \frac{1}{T} \sum_k^{\infty} a_k^2 \left(\frac{\epsilon-\tau}{\epsilon^2}\right) \qquad (5.2.a)$$

$$= \frac{R_o}{\epsilon T_o}\,(1-\frac{\tau}{\epsilon})$$

en donde:

$$R_o = \lim_{T\to\infty} \frac{T_o}{T} \sum_k^{\infty} a_k^2 \qquad (5.2.b)$$

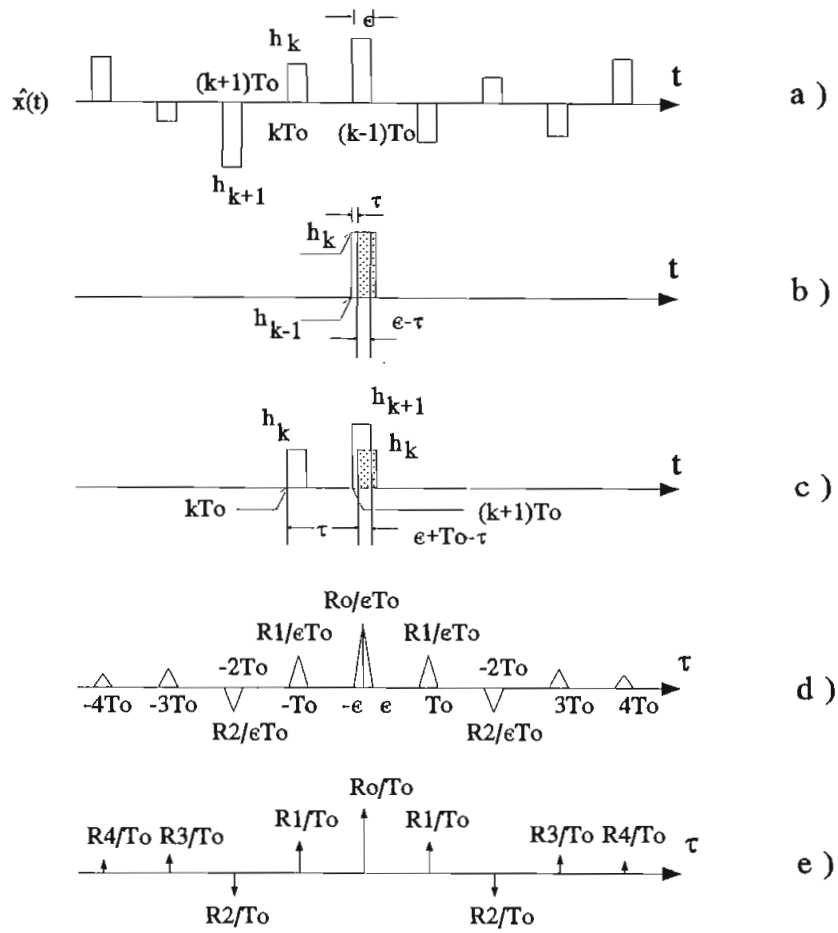


Figura 5.2. Diagrama de correlación
a) Pulsos rectangulares
b) Area de un pulso
c) Traslape de pulsos
d) Correlación con pulsos triangulares
e) $\Re_x(\tau)$ cuando ϵ tiende a 0 en $\Re_{\hat{x}}(\tau)$

Ya que $\Re_{\hat{x}}(\tau)$ es una función par de τ,

$$\Re_{\hat{x}}(\tau) = \frac{R_o}{\epsilon T_o}\left(1-\frac{|\tau|}{\epsilon}\right) \qquad |\tau| < \epsilon \tag{5.3}$$

Este es un pulso triangular de altura $R_o/\epsilon T_o$ y ancho 2ϵ con centro en $\tau=0$, tal como se observa en la figura 5.2.(d).

La autocorrelación $\Re_{\hat{x}}(\tau)$ se aproxima a 0 cuando τ tiende a ϵ, ya que para $\tau = \epsilon$, la señal retardada $\hat{x}(t-\tau)$ no se

traslapa más a $\hat{x}(t)$, y el productc $\hat{x}(t)\hat{x}(t-\tau)$ es igual a cero.

Entre más se incrementa τ, se encuentra que el k-ésimo pulso de $\hat{x}(t-\tau)$ comenzará a traslaparse al (k+1)-ésimo pulso de $\hat{x}(t)$ cuando τ se aproxima a T_o como se aprecia en la figura 5.2.(c). Repitiendo el argumento anterior, se ve que $\mathfrak{R}_{\hat{x}}(\tau)$ tendrá otro pulso triangular de ancho 2ϵ con centro en $\tau = T_o$ y de altura $R_1/\epsilon T_o$ donde:

$$R_1 = \lim_{T\to\infty} \frac{T_o}{T} \sum_{k}^{\infty} a_k\, a_{k+1}$$

Sucede una cosa similar alrededor de $\tau = 2T_o, 3T_o....$ y así sucesivamente. En consecuencia, $\mathfrak{R}_{\hat{x}}(\tau)$ consta de una sucesión de pulsos triangulares de ancho 2ϵ con centro en $\tau = 0, \pm T_o, \pm 2T_o...$, y así sucesivamente. La altura de los pulsos con centro en $\pm nT_o$ es $R_n/\epsilon T_c$ donde:

$$R_n = \lim_{T\to\infty} \frac{T_o}{T} \sum_{k}^{\infty} a_k a_{k+n}$$

Para encontrar $\mathfrak{R}_x(\tau)$, se hace que $\epsilon\to 0$ en $\mathfrak{R}_{\hat{x}}(\tau)$. Cuando $\epsilon\to 0$, el ancho de cada pulso triangular tiende a 0 y la altura tiende a ∞ en tal forma que el área será aún finita. Para el n-ésimo pulso con centro en nT_o, la altura es $R_n/\epsilon T_o$ y el área será R_n/T_o. En consecuencia de la figura 5.2.(e) se tiene:

$$\mathfrak{R}_x(\tau) = \frac{1}{T_o} \sum_{n=-\infty}^{\infty} R_n \delta(\tau - nT_o) \qquad (5.4)$$

donde:

$$R_n = \lim_{T\to\infty} \frac{T_o}{T} \sum_{k}^{\infty} a_k\, a_{k+n} \qquad (5.5)$$

La DEP $S_x(\omega)$ es la transformada de Fourier de $\Re_x(\tau)$. Por lo tanto,

$$S_x(\omega) = \frac{1}{T_o} \sum_{n=-\infty}^{\infty} R_n \, e^{-jn\omega T_o} \qquad (5.6)$$

Si se reconoce el hecho de que $R_{-n} = R_n$, se tiene

$$S_x(\omega) = \frac{1}{T_o} \left[R_o + 2 \sum_{n=1}^{\infty} R_n \, Cos(n\omega T_o) \right] \qquad (5.7)$$

Considérese el tren de pulsos $y(t)$ de la figura 5.3.(b), formado por un pulso básico $p(t)$ como el de la figura 5.3.(a) con el késimo pulso $a_k \, p(t)$ localizado en kT_o.

Si el tren de impulsos $x(t)$ se aplica a la entrada de un filtro con respuesta de impulso unitario $p(t)$, la salida del filtro será la señal deseada $y(t)$.

Por lo tanto:

$$S_y(\omega) = | P(\omega) |^2 \, S_x(\omega) \qquad (5.8.a)$$

$$S_y(\omega) = \frac{| P(\omega) |^2}{T_o} \left[R_o + 2 \sum_{n=1}^{\infty} R_n \, Cos(n\omega T_o) \right] \qquad (5.8.b)$$

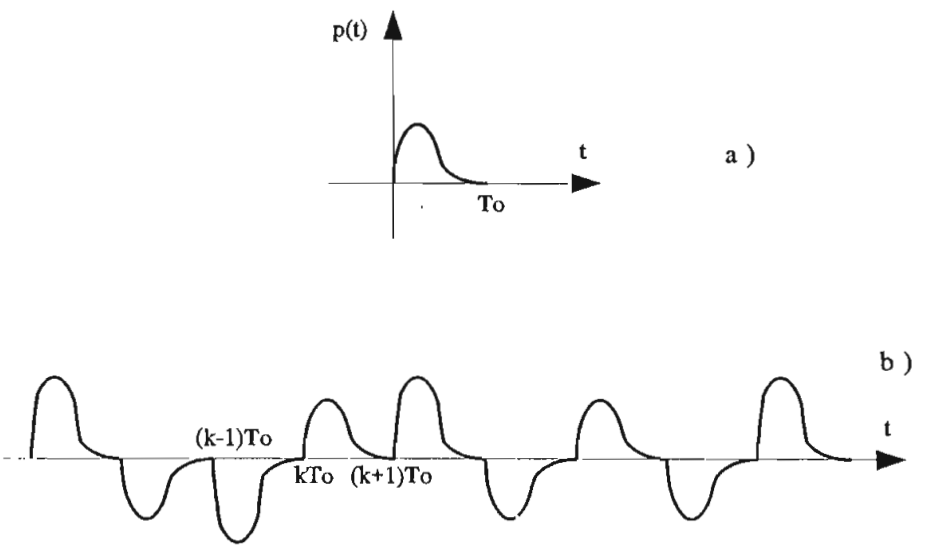


Figura 5.3. Tren de pulsos $p(t)$

A continuación se deducen las expresiones matemáticas de los diferentes códigos, las cuales se utilizarán como base de los algoritmos para la graficación de las densidad espectral de potencia de cada uno de ellos y su posterior comparación.

5.1.1. DEP del código RZ

Para este código el valor de a_k puede ser un 1 o un 0. Existe un total de T/T_o posiciones de pulsos en el intervalo (- T/2, T/2). Se supone que 1_L y 0_L son igualmente probables, $a_k = 1$ para $T/2T_o$ pulsos, y $a_k = 0$ para el resto de los $T/2T_o$ pulsos. En consecuencia:

$$R_o = \lim_{T\to\infty} \frac{T_o}{T} \sum_{k}^{\infty} a_k^2$$

$$= \frac{T}{T_o} \left(\frac{T}{2T_o}\right) (1)^2 = \frac{1}{2} \qquad (5.9)$$

y para los demás coeficientes en forma general:

$$R_n = \lim_{T\to\infty} \frac{T_o}{T} \sum_{k}^{\infty} a_k\, a_{k+n}$$

El producto de $a_k\, a_{k+n}$ puede ser un 1 o un 0, ya que a_k es 1 la mitad del tiempo y 0 la otra mitad. El caso con a_{k+n} es similar al de a_k, por lo tanto, se tienen cuatro posibilidades (1 x 1, 1 x 0, 0 x 1 ó 0 x 0), todas ellas igualmente probables. Por consiguiente, el producto de $a_k\, a_{k+n}$ será 1 para una cuarta parte de los términos, en promedio, y 0 para los términos restantes. Ya que existen T/T_o términos en el intervalo (- T/2, T/2).

$$R_n = \frac{T_o}{T}\ (\frac{T}{4T_o})\ (1) = \frac{1}{4} \qquad (5.10)$$

Por lo tanto de la ecuación 5.6,

$$S_x(\omega) = \frac{1}{4T_o} + \frac{1}{4T_o} \sum_{n=-\infty}^{\infty} e^{-jn\omega T_o} \qquad (5.11)$$

Utilizando la relación:

$$\sum_{n=-\infty}^{\infty} e^{-jn\omega T_o} = \frac{2\pi}{T_o} \sum_{n=-\infty}^{\infty} \delta(\omega - \frac{2\pi n}{T_o})$$

entonces:

$$S_x(\omega) = \frac{1}{4T_o} + \frac{2\pi}{4T_o^2} \sum_{n=-\infty}^{\infty} \delta(\omega - \frac{2\pi n}{T_o}) \qquad (5.12.a)$$

y la DEP deseada de la forma de onda $y(t)$ del código RZ es:

$$S_y(\omega) = \frac{|P(\omega)|^2}{4T_o} \cdot [1+\frac{2\pi}{T_o} \sum_{n=-\infty}^{\infty} \delta(\omega-\frac{2\pi n}{T_o}] \qquad (5.12.b)$$

donde $P(\omega)$ es la transformada de Fourier del pulso básico $p(t)$ que se usa. Para el caso del código RZ que usa un pulso rectangular de la mitad de ancho:

$$p(t) = \prod(\frac{t}{T_o/2}) = \prod(\frac{2t}{T_o}) \qquad (5.13.a)$$

$$P(\omega) = \frac{T_o}{2} Sinc(\frac{\omega T_o}{4\pi}) \qquad (5.13.b)$$

entonces:

$$S_y(\omega) = \frac{T_o}{16} Sinc^2(\frac{\omega T_o}{4\pi}) [1+\frac{2\pi}{T_o} \sum_{n=-\infty}^{\infty} \delta(\omega-\frac{2\pi n}{T_o})] \qquad (5.14)$$

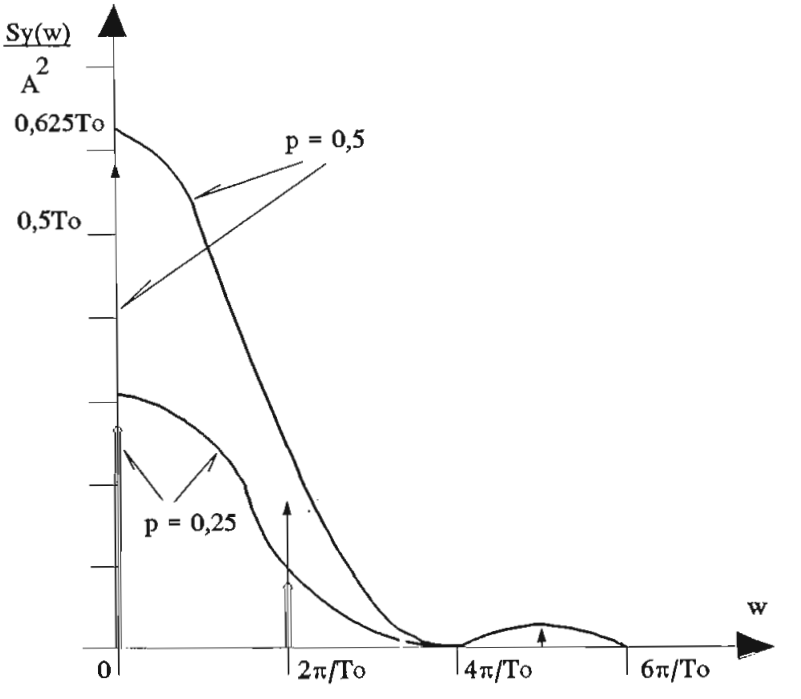


Figura 5.4. DEP para el código RZ

Este espectro se muestra en la figura 5.4. Obsérvese que el espectro consta tanto de una parte continua como de una discreta. Está presente una componente discreta a la frecuencia de reloj $f_o = 1/T_o$. En esta figura se puede observar que el ancho de banda de la señal es $2f_o$ donde f_o es la frecuencia de reloj. Igualmente se puede concluir que el ancho de banda de la señal es cuatro veces el ancho de banda teórico (ancho de banda de Nyquist) requerido.

Una desventaja de la señal RZ es que tiene una DEP diferente de cero en DC($\omega = 0$). lo que dificultará el acoplamiento a CA durante la transmisión.

En segundo lugar, los requisitos de ancho de banda de transmisión son excesivos; además, la señal RZ no tiene detección de errores ni capacidad de correlación y, por último, no es transparente. Una sucesión larga de 0_L puede crear errores en la extracción de la temporización.

5.1.2. DEP del código RZ según la probabilidad

Es importante encontrar la DEP de una señal RZ si el 1_L y el 0_L no son igualmente probables. Para analizar esto se supone que la probabilidad de transmitir 1_L es Q y la de transmitir 0_L es P = 1-Q donde ($0 \leq Q \leq 1$). Esto significa que si se transmiten N dígitos, entonces, en promedio, NQ dígitos serán 1_L y N(1-Q) dígitos será 0_L ($N < \infty$).

En este caso existen $(T/T_o)Q$ pulsos diferentes de cero (pulsos de encendido), y

$$R_o = \frac{T_\delta}{T} \left(\frac{TQ}{T_o}\right) (1)^2 = Q$$

Para calcular R_n, se debe observar que en el intervalo (-T/2, T/2), solamente $(T/T_o)Q$. a_k son 1_L. Para cada

una de estas a_k la probabilidad de encontrar $a_{k+n} = 1$ es Q (o sea, fuera de las $(TQ/T_o)\,a_k$ que son 1_L, sólo una fracción de Q encontrará un compañero a_{k+1} que es también un 1_L). Por lo tanto:

$$\sum_{k}^{\infty} a_k\, a_{k+n} = Q\,(\frac{TQ}{T_o}) = \frac{TQ^2}{T_o}$$

Y

$$R_n = \frac{T_o}{T}\,(\frac{TQ^2}{T_o}) = Q^2$$

De la ecuación (5.8.b) se tiene:

$$S_y(\omega) = |P(\omega)|^2\,(\frac{Q}{T_o})\,[1 + 2Q\sum_{n=1}^{\infty} Cos(n\omega T_o)]$$

$$= |P(\omega)|^2\,(\frac{Q}{T_o})\,[(1-Q) + Q(1 + 2\sum_{n=1}^{\infty} Cos(n\omega T_o)]$$

$$= |P(\omega)|^2\,(\frac{Q}{T_o})\,[(1-Q) + \frac{2\pi Q}{T_o}\sum_{n=-\infty}^{\infty} \delta(\omega - \frac{2\pi n}{T_o})]$$

La forma del espectro es similar a la de la figura 5.4 excepto por el peso relativo de las componentes continuas y discretas que dependen de la probabilidad.

5.1.3. DEP del código NRZ

Partiendo de la ecuación (5.12.c) el pulso básico $p(t)$ que se usa para un código NRZ es un pulso rectangular de ancho completo; dando como resultado:

$$p(t) = \prod(\frac{t}{T_o}) \qquad (5.15.a)$$

$$P(\omega) = T_o Sinc(\frac{\omega T_o}{2\pi}) \qquad (5.15.b)$$

$$S_y(\omega) = \frac{T_o}{4} Sinc^2(\frac{\omega T_o}{2\pi}) [1 + \frac{2\pi}{T_o} \sum_{n=-\infty}^{\infty} \delta(\omega - \frac{2\pi n}{T_o})] \qquad (5.15.c)$$

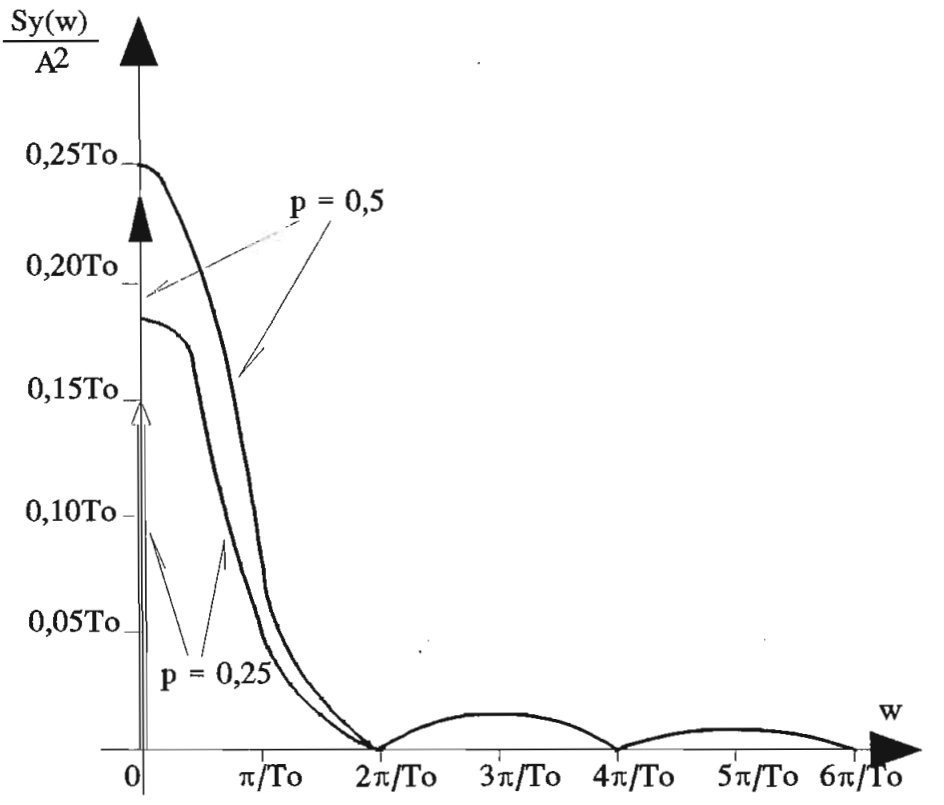


Figura 5.5. DEP para el código NRZ

5.1.4. DEP del código POLAR

5.1.4.1. DEP del código POLAR RZ

En la señal polar, 1_L se transmite mediante un pulso $p(t)$ y 0_L se transmite mediante $-p(t)$. En este caso a_k tiene

igual probabilidad de ser 1 ó -1, y a_k^2 será siempre 1. En consecuencia:

$$R_o = \lim_{T\to\infty} \frac{T_o}{T} \sum_{k}^{\infty} a_k^2 = \frac{T_o}{T}\left(\frac{T}{T_o}\right)(1) = 1$$

En forma similar, $a_k\, a_{k+n}$ puede ser 1 ó -1. Para la mitad de la combinación es 1, y para la mitad restante es -1. Por lo tanto, $R_n = 0$ y

$$S_y(\omega) = \frac{|P(\omega)|^2}{T_o} R_o \qquad (5.16)$$

$$= \frac{|P(\omega)|^2}{T_o}$$

Para un pulso rectangular de la mitad de ancho código RZ polar [ecuación (5.13.b)],

$$S_y(\omega) = \frac{T_o}{4} Sinc^2\left(\frac{\omega T_o}{4\pi}\right) \qquad (5.17)$$

Este espectro es idéntico a la componente continua de la señal de encendido-apagado.

5.1.4.2. DEP del código POLAR NRZ

Para el código polar NRZ partiendo de la ecuación (5.16) y usando un pulso rectangular de ancho completo se tiene:

$$S_y(\omega) = T_o\, Sinc^2\left(\frac{\omega T_o}{2\pi}\right) \qquad (5.18)$$

El espectro de este código se muestra en la figura 5.7.

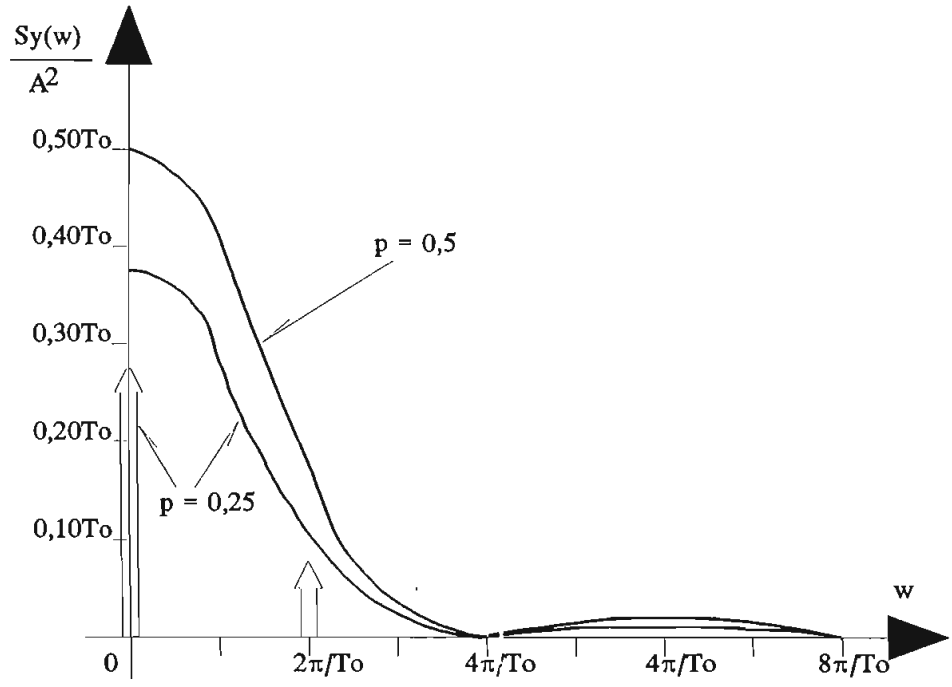


Figura 5.6. DEP del código polar RZ

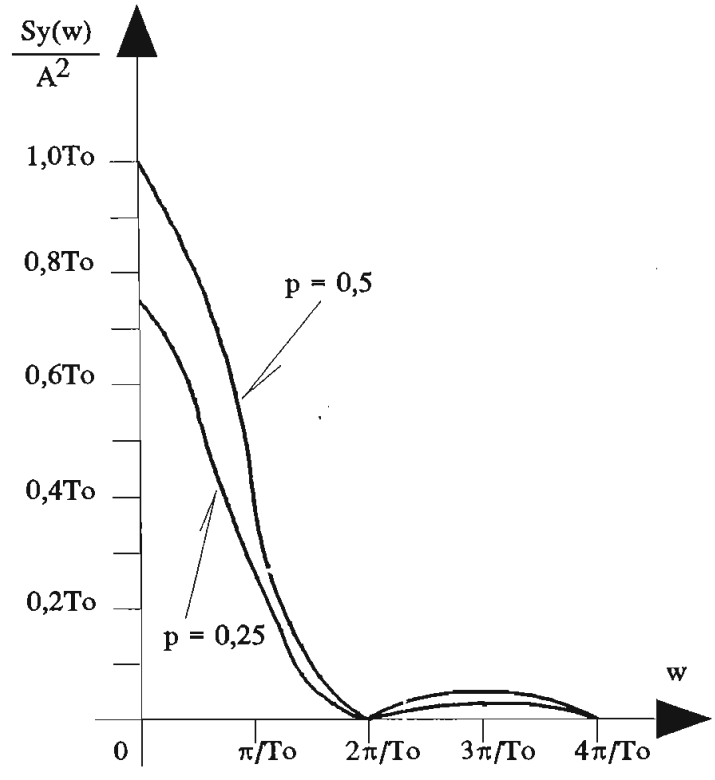


Figura 5.7. DEP del código polar NRZ

5.1.5. DEP del código BIPOLAR O AMI

5.1.5.1. DEP del código BIPOLAR RZ

En esta señal, un 0_L se transmite por ausencia de pulso, y un 1_L se transmite mediante un pulso $p(t)$ o $-p(t)$, dependiendo de si el 1_L anterior se transmitió mediante $-p(t)$ o $p(t)$. Alternando pulsos consecutivos se puede ocasionar una DC nula en la DEP. La señal bipolar actualmente utiliza tres símbolos [$p(t)$, 0 y $-p(t)$], en consecuencia, es en realidad ternaria más que binaria.

Para calcular la DEP, se tiene

$$R_o = \lim_{T\to\infty} \frac{T_o}{T} \sum_k^{\infty} a_k^2$$

En promedio, la mitad de las $a_k = 0$ y la otra mitad son ya sea 1 ó -1, con $a_k^2 = 1$. Puesto que existen $T/2T_o$ pulsos en el intervalo (-T/2, T/2),

$$R_o = \frac{T_c}{T} \left(\frac{t}{2T_o}\right) (\pm 1)^2 = \frac{1}{2}$$

$$R_1 = \lim_{T\to\infty} \frac{T_o}{T} \sum_k^{\infty} a_k\, a_{k+1}$$

Debido a que los pulsos positivos y negativos se alternan, a_k y a_{k+1} no pueden ser 1 simultáneamente; en consecuencia, $a_k.a_{k+1}$ es tanto 0 como -1. Dos dígitos binarios consecutivos tienen sólo cuatro patrones posibles: (11), (10), (01) y (00), los cuatro se presentan en la misma proporción.

El producto $a_k.a_{k+1}$ de estas combinaciones es -1, 0,0,0, respectivamente. En consecuencia, fuera de T/T_o pulsos

en el intervalo (-T/2, T/2), $T/4T_o$ produciría $a_k.a_{k+1} = -1$, y el resto daría por resultado $a_k.a_{k+1} = 0$. Por lo tanto, $R_1 = -1/4$

Para n > 1, $a_k.a_{k+n}$ es 0, 1 ó -1. La probabilidad del producto $a_k.a_{k+n}$ siendo 1, ó -1 es la misma (1/8). Por lo tanto, la suma $\sum a_k a_{k+n}$ será igual a cero.

$$R_n = 0 \qquad n > 1$$

y

$$S_y(\omega) = \frac{|P(\omega)|^2}{2T_o} [1 - Cos(\omega T_o)] \qquad (5.19.a)$$

$$S_y(\omega) = \frac{|P(\omega)|^2}{T_o} Sen^2 (\frac{\omega T_o}{2}) \qquad (5.19.b)$$

Obsérvese que $S_y(\omega) = 0$ para $\omega = 0$ (DC), independientemente de $P(\omega)$. En consecuencia, la DEP tiene una DC nula, lo cual es deseable para el acoplamiento a CA. En el caso del código AMI-RZ que usa pulsos rectangulares de la mitad de ancho [ec (5.13)] se tiene:

$$S_y(\omega) = \frac{T_o}{4} Sinc^2 (\frac{\omega T_o}{4\pi}) Sen^2 (\frac{\omega T_o}{2}) \qquad (5.20)$$

Esto se muestra en la figura 5.8. El ancho de banda esencial de la señal es f_o ($f_o = 1/T_o$), que es la mitad de la señal RZ unipolar y dos veces el ancho de banda mínimo teórico.

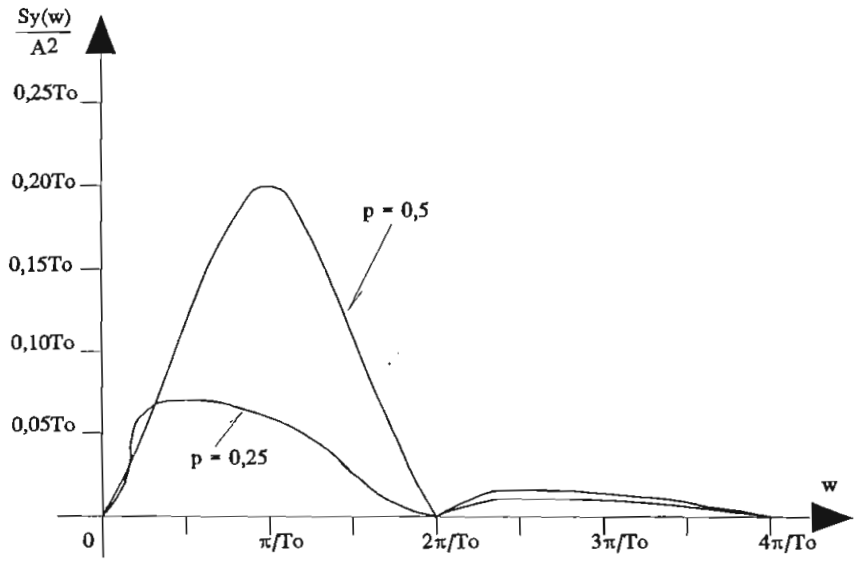


Figura 5.8. DEP de señal bipolar RZ

5.1.5.2. DEP del código BIPOLAR NRZ

Partiendo de la ecuación (5.19.b) para un pulso de ancho completo, se tiene la siguiente ecuación del código bipolar NRZ:

$$S_y(\omega) = T_o \, Sinc^2 \left(\frac{\omega T_o}{2\pi}\right) Sen^2 \left(\frac{\omega T_o}{2}\right) \qquad (5.21)$$

El espectro de este código se representa en la figura 5.9.

5.1.6. DEP del código BIFASE L o MANCHESTER

Debido a que la DEP $S_y(\omega)$ es el producto de $|P(\omega)|^2$ por $S_x(\omega)$, el espectro puede ser formado controlando a $P(\omega)$ o $S_x(\omega)$. La DC nula se obtuvo en el caso bipolar haciendo $S_x(\omega)=0$ en $\omega = 0$.

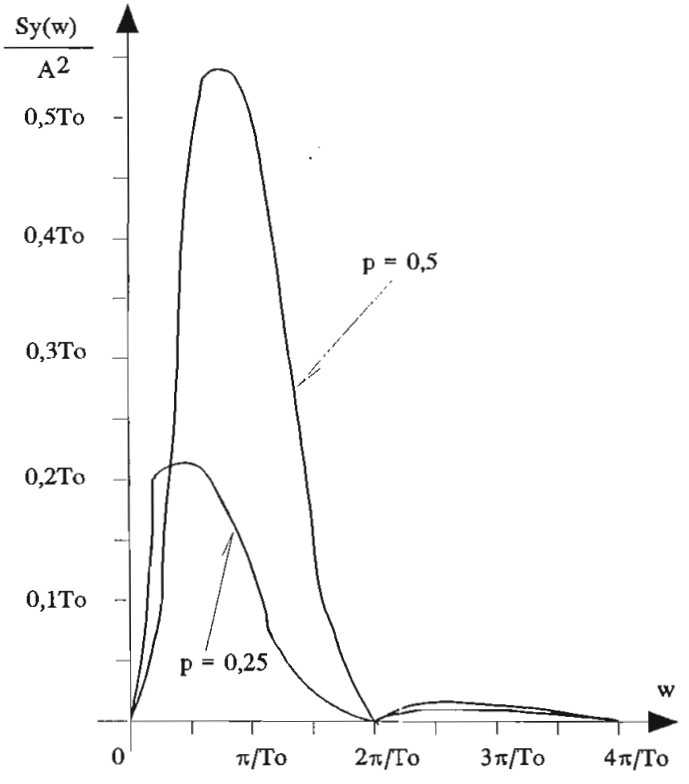


Figura 5.9. DEP de una señal bipolar NRZ

El mismo efecto se obtiene al elegir a $p(t)$ de modo que $P(\omega)=0$ en $\omega=0$. Ya que:

$$P(\omega) = \int_{-\infty}^{\infty} p(t)\ e^{-j\omega t}\ dt$$

entonces

$$P(0) = \int_{-\infty}^{\infty} p(t)\ dt \qquad (5.22)$$

Por lo tanto, si el área bajo $p(t)$ se hace igual a cero, $P(0)$ será también cero, y se tendrá una DC nula en la DEP. Existen varias formas de hacerlo. Para un pulso rectangular, en la figura 5.10.(a) se muestra una forma posible

de $p(t)$.

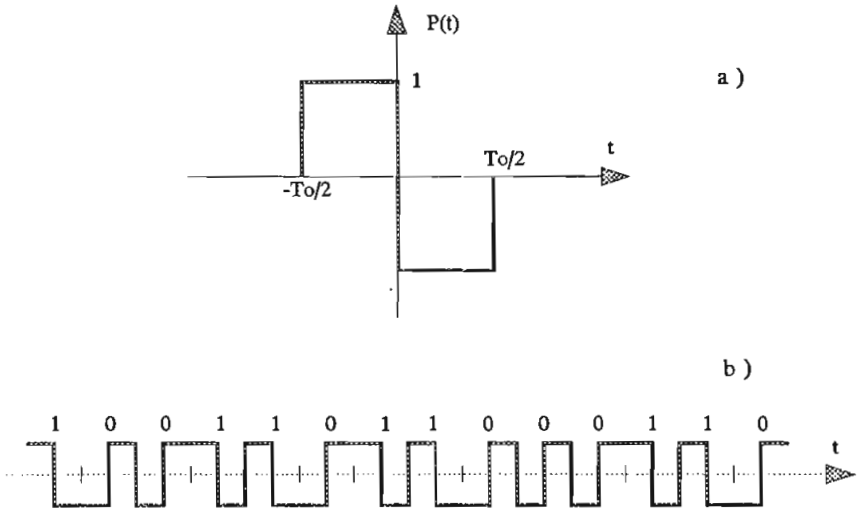


Figura 5.10. Una señal de fase dividida
a) Pulso rectangular
b) Señal de Manchester

Aquí cada bit se representa mediante dos pulsos de polaridad opuesta. El binario 0_L se transmite mediante $-p(t)$, como se muestra en la figura 5.10.(b). Esta señal se conoce como de Manchester, o de fase dividida (también binaria gemela).

Ya que ésta es una señal polar, de la ec. 5.16 se tiene que la DEP $S_y(\omega)$ es:

$$S_y(\omega) = \frac{|P(\omega)|^2}{T_o}$$

Para el pulso $p(t)$ de la figura 5.10.(a) se tiene:

$$p(t) = \prod \left(\frac{t+\frac{T_o}{4}}{\frac{T_o}{2}} \right) - \prod \left(\frac{t-\frac{T_o}{4}}{\frac{T_o}{2}} \right)$$

En consecuencia:

$$P(\omega) = \frac{T_o}{2}\ Sinc\ (\frac{\omega T_o}{4\pi})\ e^{j\omega T_o/4} - \frac{T_o}{2}\ Sinc\ (\frac{\omega T_o}{4\pi})\ e^{-j\omega T_o/4}$$

entonces:

$$P(\omega) = jT_o\ Sinc\ (\frac{\omega T_o}{4\pi})\ Sen\ (\frac{\omega T_o}{4}) \qquad (5.23)$$

$$S_y(\omega) = T_o\ Sinc^2\ (\frac{\omega T_o}{4\pi})\ Sen^2\ (\frac{\omega T_o}{4}) \qquad (5.24)$$

La figura 5.11 muestra la densidad espectral de este código.

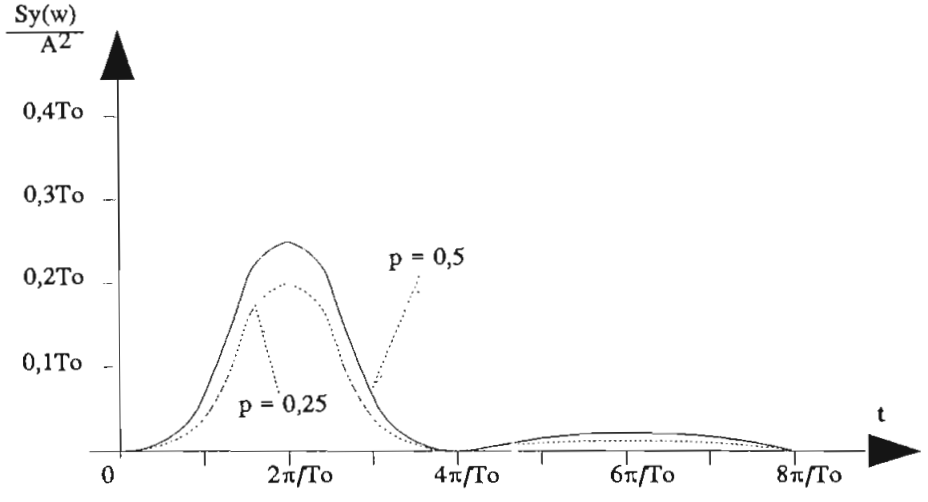


Figura 5.11. DEP del código Manchester

La comparación de ésta densidad espectral de potencia con la del código bipolar de la ecuación (5.20) demuestra que el ancho de banda para la señal de fase dividida es dos veces el de la señal bipolar.

Sin embargo, la fase dividida presenta una ventaja sobre la bipolar, esta es la de presentar transiciones en cada período, ya que cada posición de pulso se encuentra ocupada y una larga secuencia de 0_L no causará dificultad alguna en la

extracción de temporización.

5.1.7. DEP del CODIGO HDB_3

Para el código HDB_3 la deducción de su DEP es mucho más compleja y requiere cálculos de R_n para valores grandes de n, tal es así que se debería calcular hasta R_{63}. Para una entrada equiprobable de 0_L y 1_L tiene la siguiente expresión matemática aproximada:

$$S_y(\omega) = \frac{40 - 32\cos\phi}{465T(1025 - 64cos5\phi)}[7258.5 - 1929\cos\phi -$$

$$- 1024\cos(2\phi) - 160\cos(3\phi) + 32\cos(4\phi) +$$

$$- \frac{131288.5 - 41399\cos\phi - 86112\cos(2\phi)}{85 - 44\cos\phi - 24\cos(2\phi) - 16\cos(3\phi)}] \quad [P(\omega)]^2$$

(5.25)

donde $\phi = \omega T$

La DEP para el código HDB_3 se observa en la figura 5.12.

5.1.8. DEP del código de MILLER

Al igual que el código HDB_3, el cálculo de la DEP del código de modulación por retardo o código de MILLER es bastante complejo y se necesita de un largo proceso matemático para encontrar una expresión matemática exacta. La ecuación aproximada es la siguiente:

$$S_y(\omega) = \frac{1}{\theta^2 T(17 + 8\cos\theta)}[23 - 2\cos\theta - 22\cos(2\theta) - 12\cos(3\theta) +$$

$$+5\cos(4\theta) + 12\cos(5\theta) + 2\cos(6\theta) - 8\cos(7\theta) + 2\cos(8\theta)]$$

(5.26)

donde $\theta = \frac{\omega T}{2}$

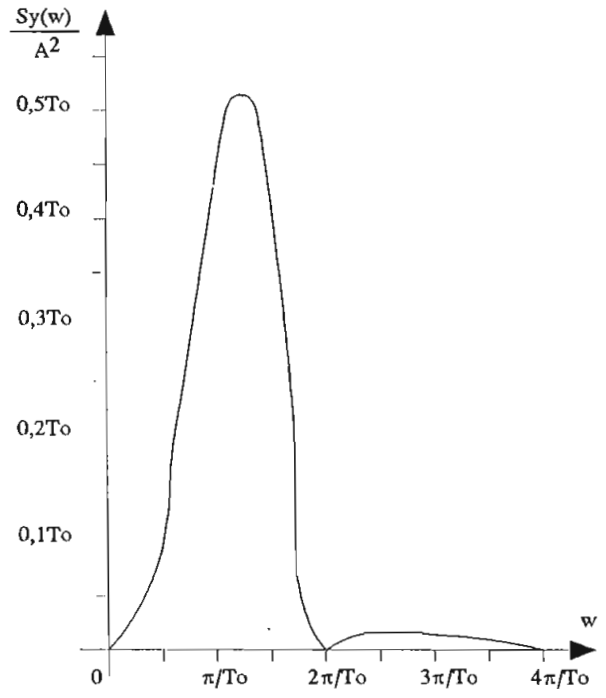


Figura 5.12. DEP del código HDB_3

La potencia en función de la frecuencia DEP se observa en la figura 5.13.

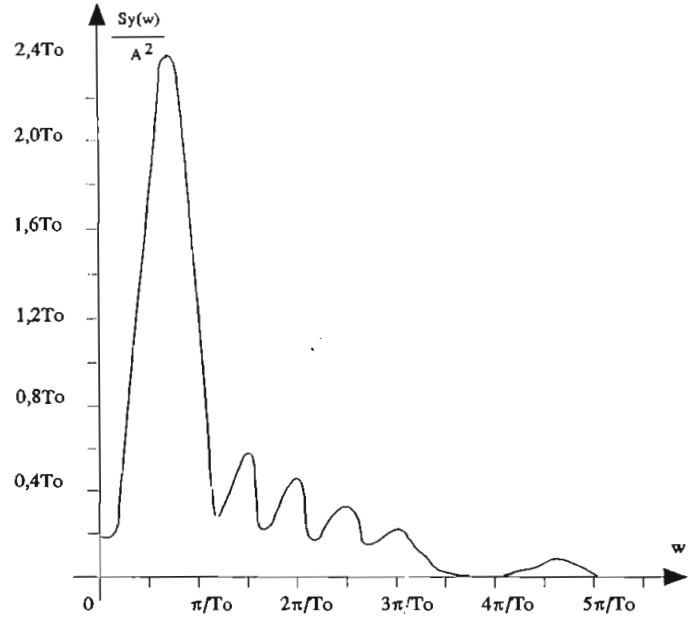


Figura 5.13. DEP del código de Miller

5.2. PROGRAMA PARA ANALISIS DE LOS ESPECTROS DE FRECUENCIA PARA LOS CODIGOS DE LINEA

El programa para el análisis de los espectros de frecuencia de los códigos de línea, consta de una serie de procedimientos que se encargan de representar gráficamente las expresiones matemáticas analizadas anteriormente para cada uno de los códigos.

En el diagrama de flujo de la figura 5.14 se ejemplifica para el código AMI-NRZ el procedimiento básico para llegar al gráfico de la ecuación.

Todos los procedimientos, inicializan las variables que permiten dimensionar el gráfico y sus escalas; luego se inicializan las variables que permiten calcular la función de los espectros de potencia. Las variables utilizadas son:

G: Toma los valores de la frecuencia angular, su valor inicial es cero.

X,7*Y2: Son las coordenadas del origen de los ejes.

P: Es el valor de la probabilidad, que se ingresa como dato.

Y: Es la variable que toma los valores de la función de los espectros de potencia.

IdentificarProbabilidad: Es un subprograma (en el caso de dos probabilidades) que identifican a cada probabilidad en las dos curvas del gráfico con P1 y P2 respectivamente. En el caso de que P1=P2 solamente existe una identificación para la curva.

En resumen los procedimientos evalúan y grafican la función.

$$S_y(\omega) = Y = f(G, P)$$

donde f es la función densidad de probabilidad correspondiente

a cada código, que para ejemplo es la correspondiente del código AMINRZ.

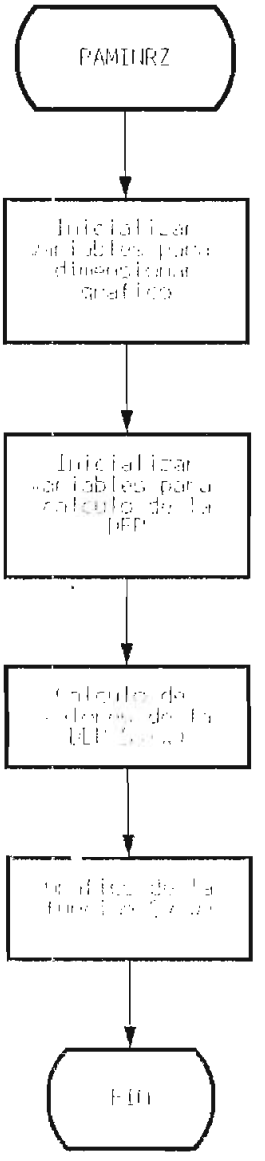


Figura 5.14. Diagrama de flujo para la DEP del código AMI-NRZ

El procedimiento **ESPECTRODEPOTENCIA** que se describe en el Capítulo 6, presenta los subprogramas adicionales para el análisis de los espectros de potencia de los códigos de línea. Este procedimiento presenta dos opciones:

1) La representación de los espectros de potencia de cada código para una condición de equiprobabilidad (P=0.5), y

2) La representación de los espectros de potencia de un mismo código para dos probabilidades distintas.

La primera opción permite observar las características espectrales de cada código en una condición de equiprobabilidad; esta opción permitirá la comparación de parámetros como ancho de banda y presencia de componente continua. La segunda opción facilita el observar la variación de las densidades espectrales de potencia con la probabilidad variable.

Como ejemplo; en las figuras 5.15 hasta la 5.23 se presentan los espectros de los principales códigos para una condición de equiprobabilidad (P=0,5); en la figura 5.24 se observa una comparación de las densidades espectrales de potencia de todos ellos para la misma condición.

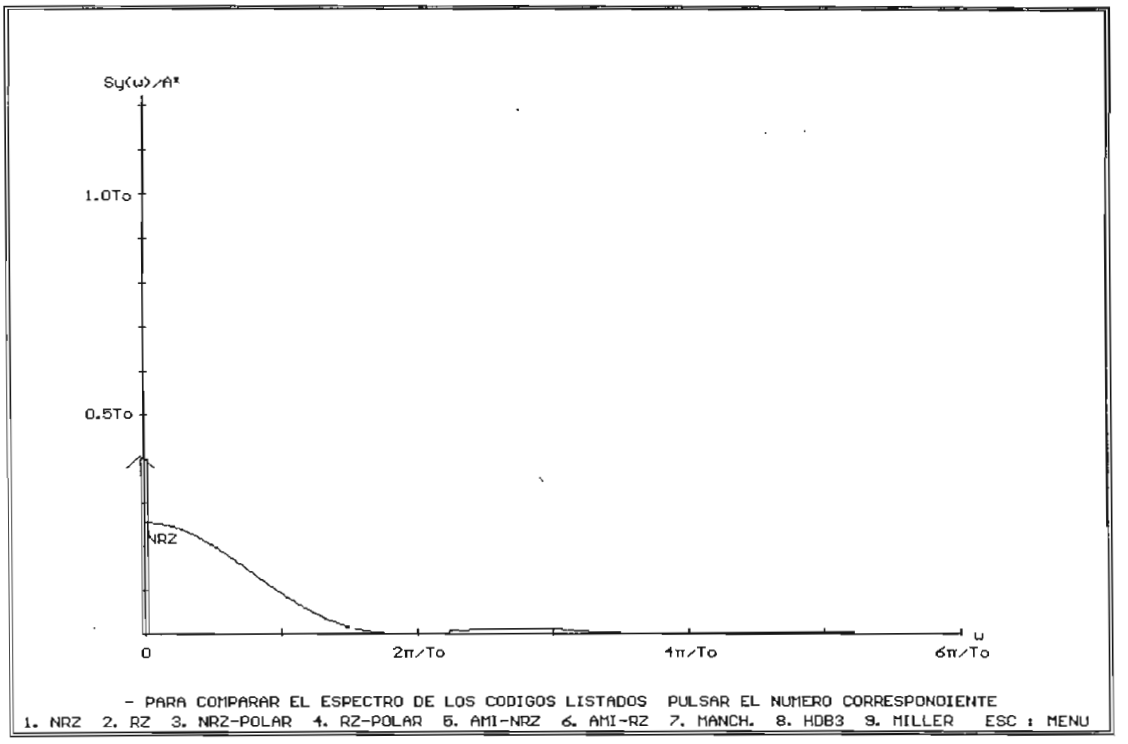


Figura 5.15. Gráfico de DEP para el código NRZ (P=0.5)

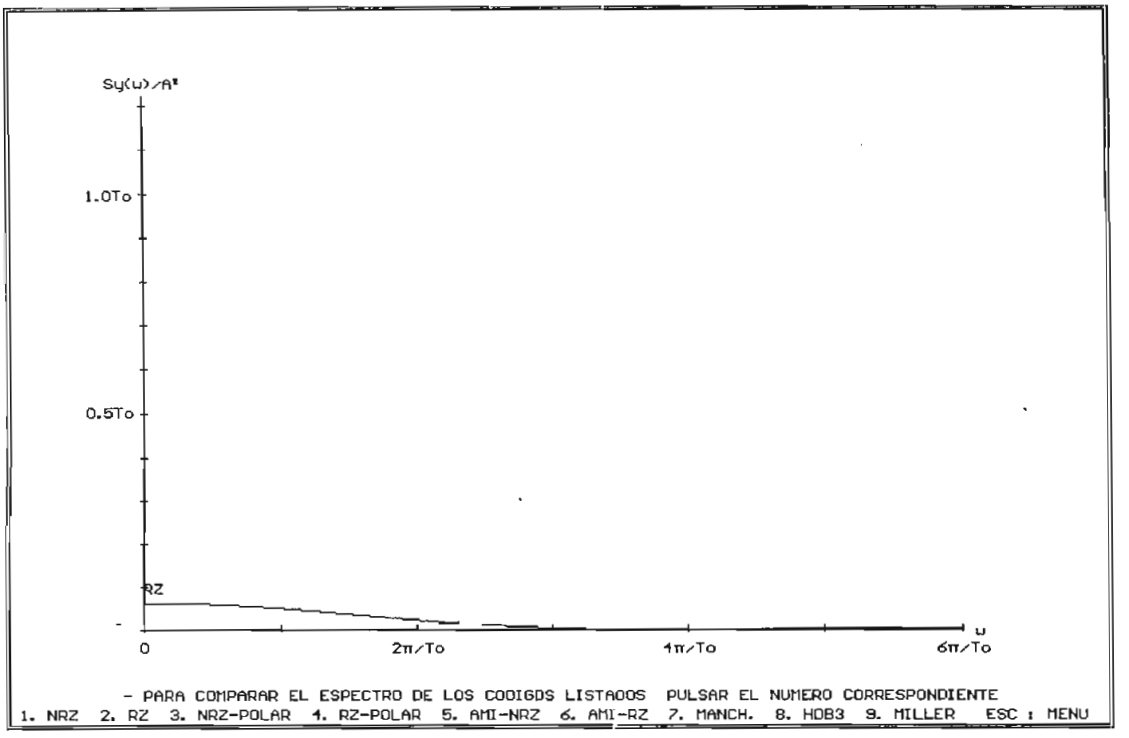


Figura 5.16. Gráfico de DEP para el código RZ (P=0.5)

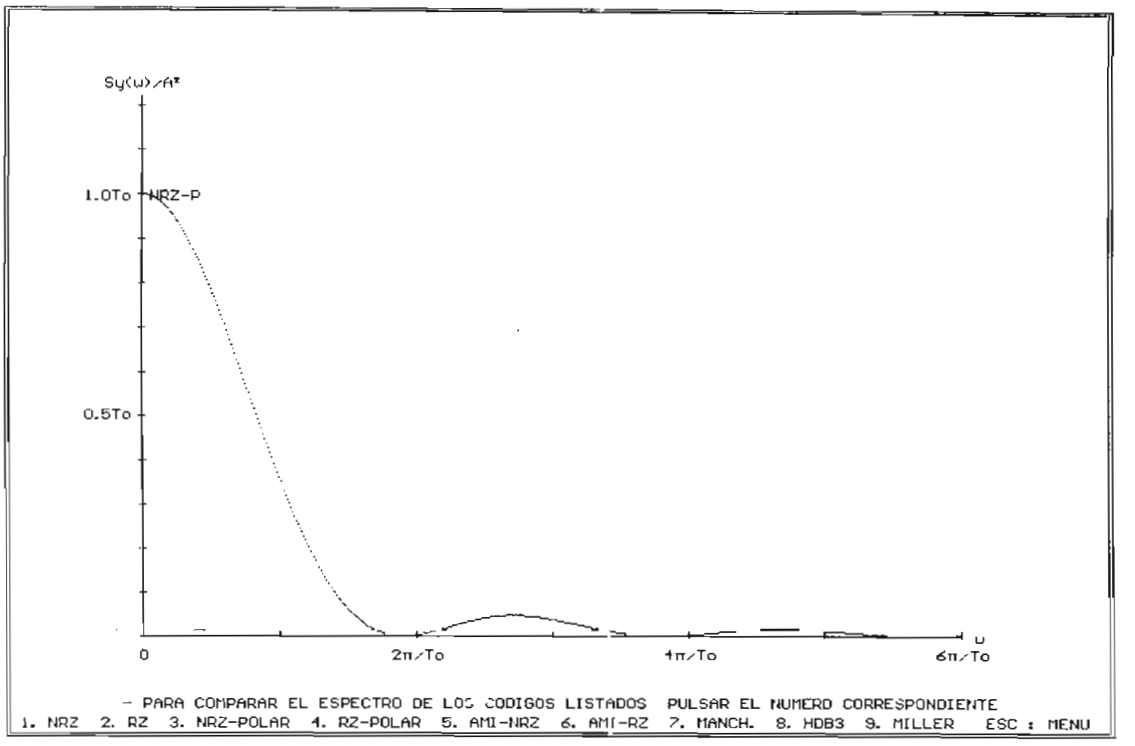


Figura 5.17. Gráfico de DEP para el código NRZ Polar (P=0.5)

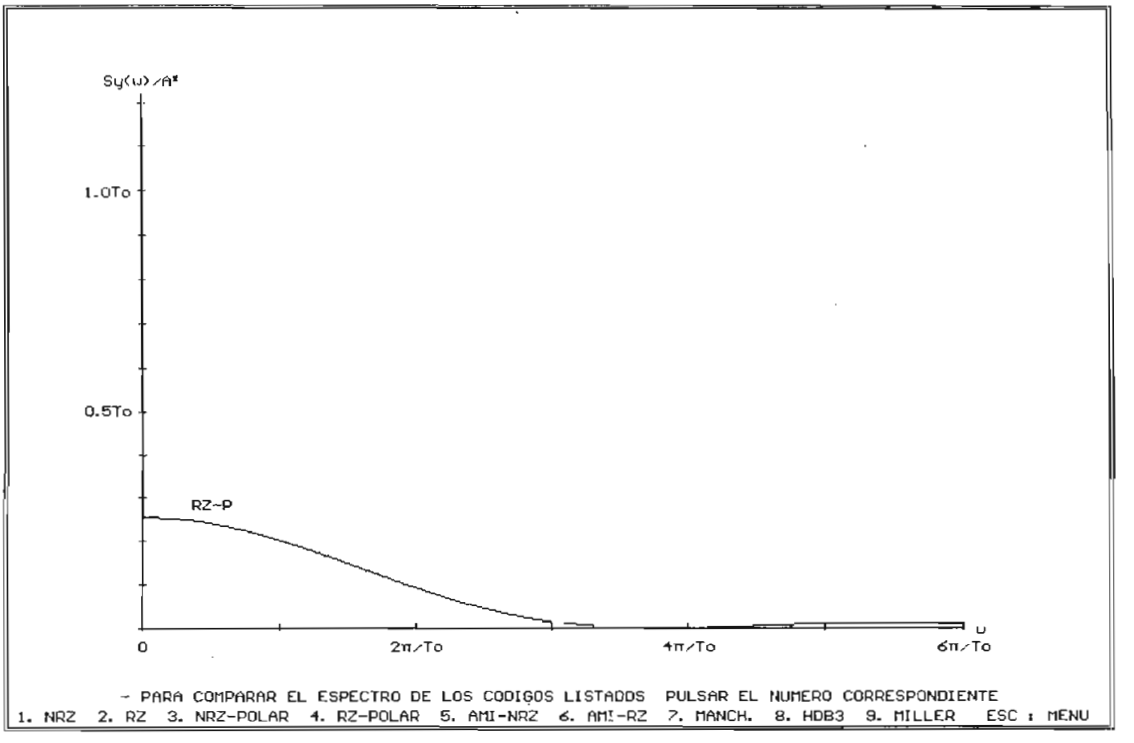


Figura 5.18. Gráfico de DEP para el código RZ Polar (P=0.5)

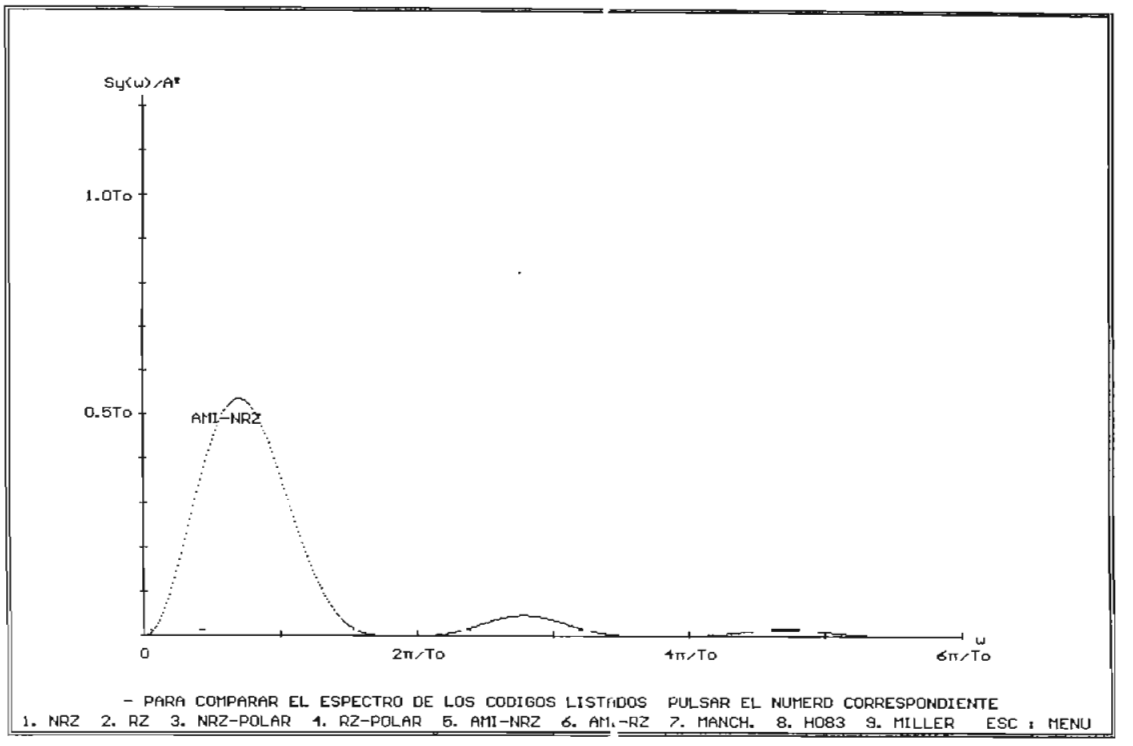


Figura 5.19. Gráfico de DEP para el código AMI NRZ (P=0.5)

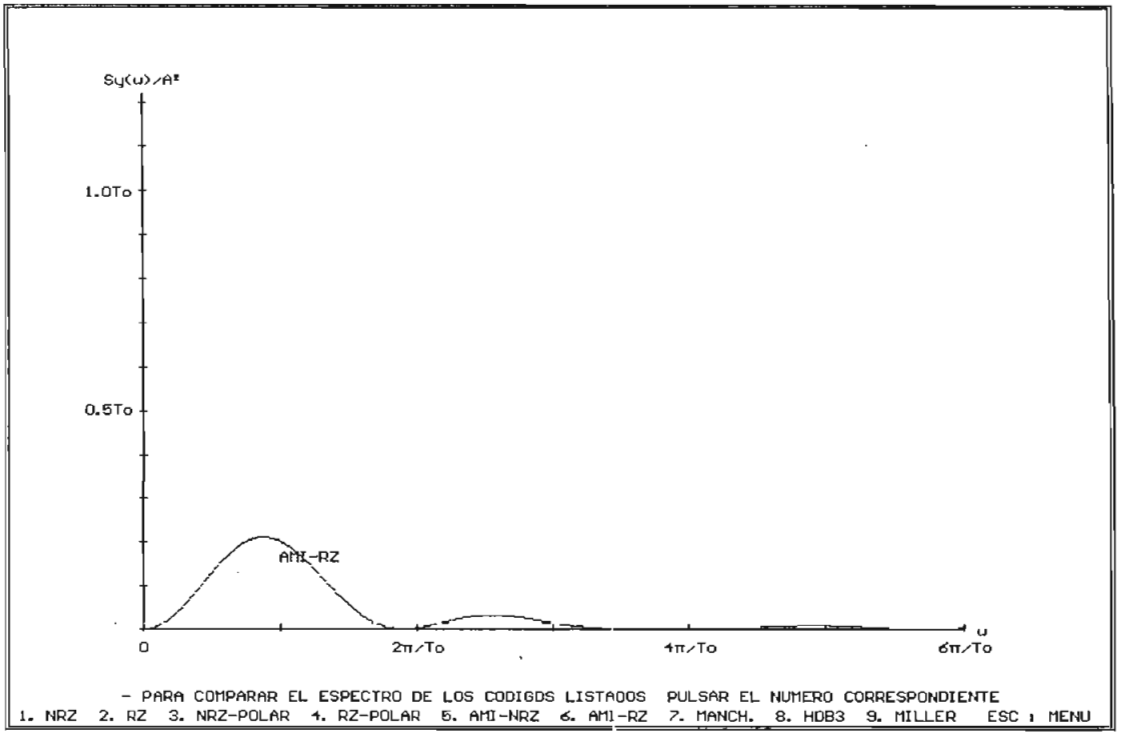


Figura 5.20. Gráfico de DEP para el código AMI RZ (P=0.5)

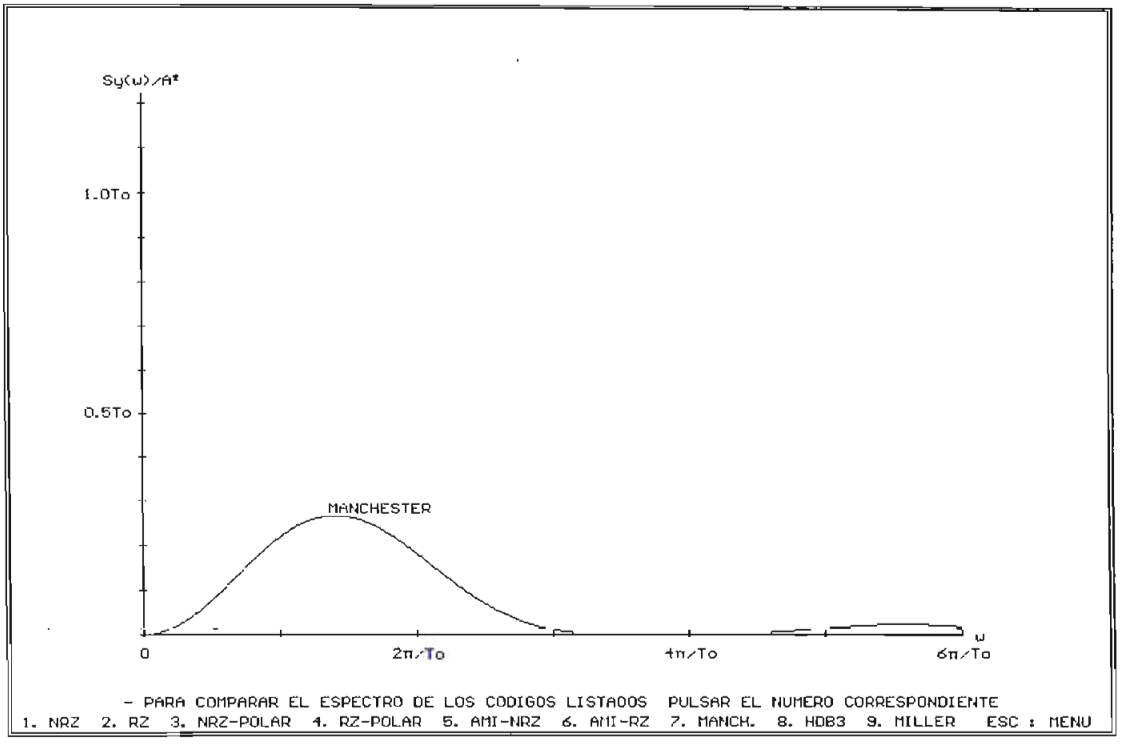


Figura 5.21. Gráfico de DEP para el código MANCHESTER (P=0.5)

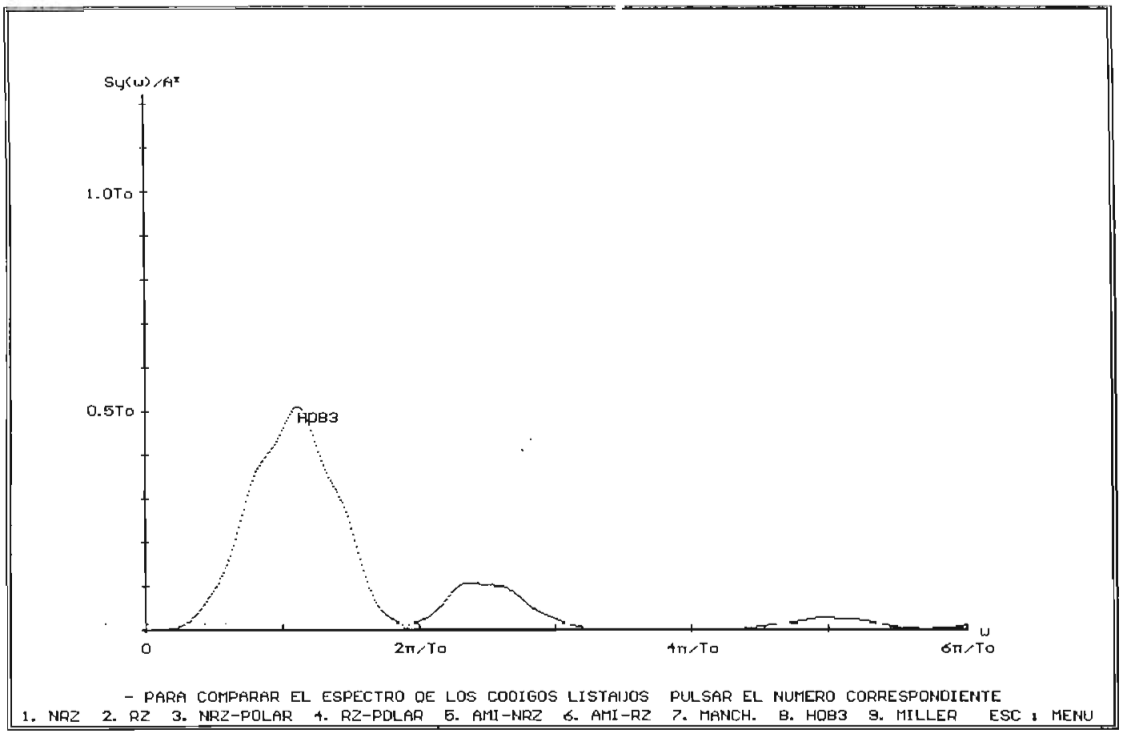


Figura 5.22. Gráfico de DEP para el código HDB_3 (P=0.5)

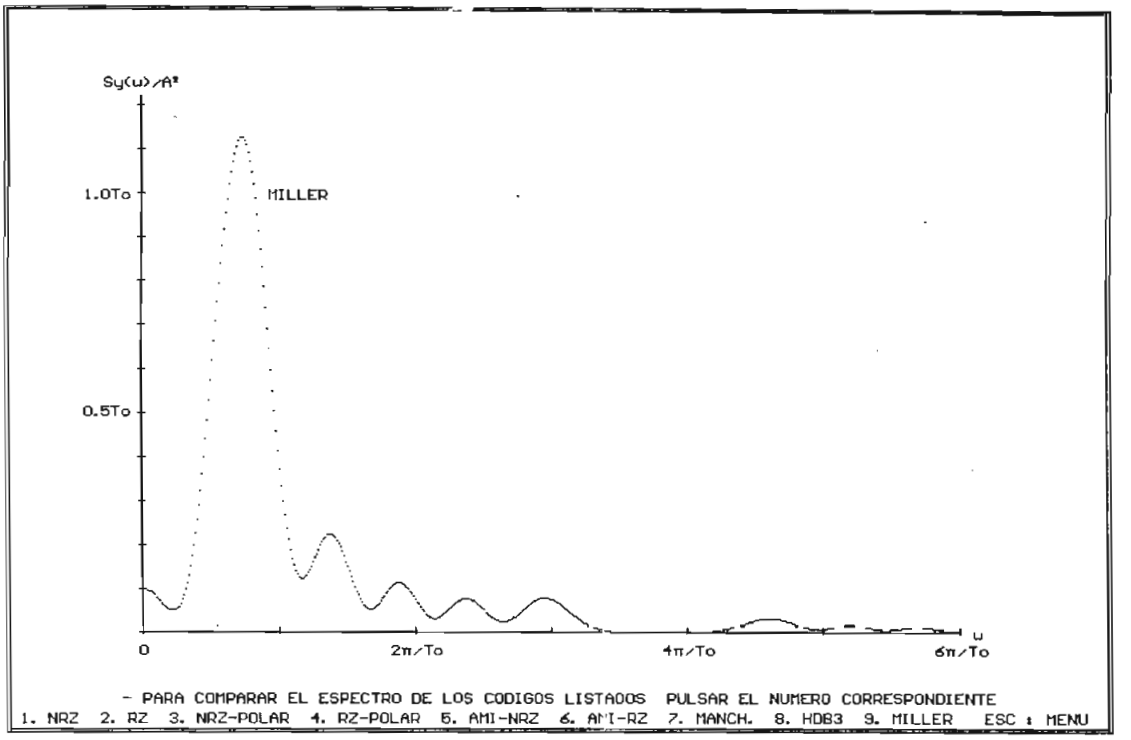


Figura 5.23. Gráfico de DEP para el código MILLER (P=0.5)

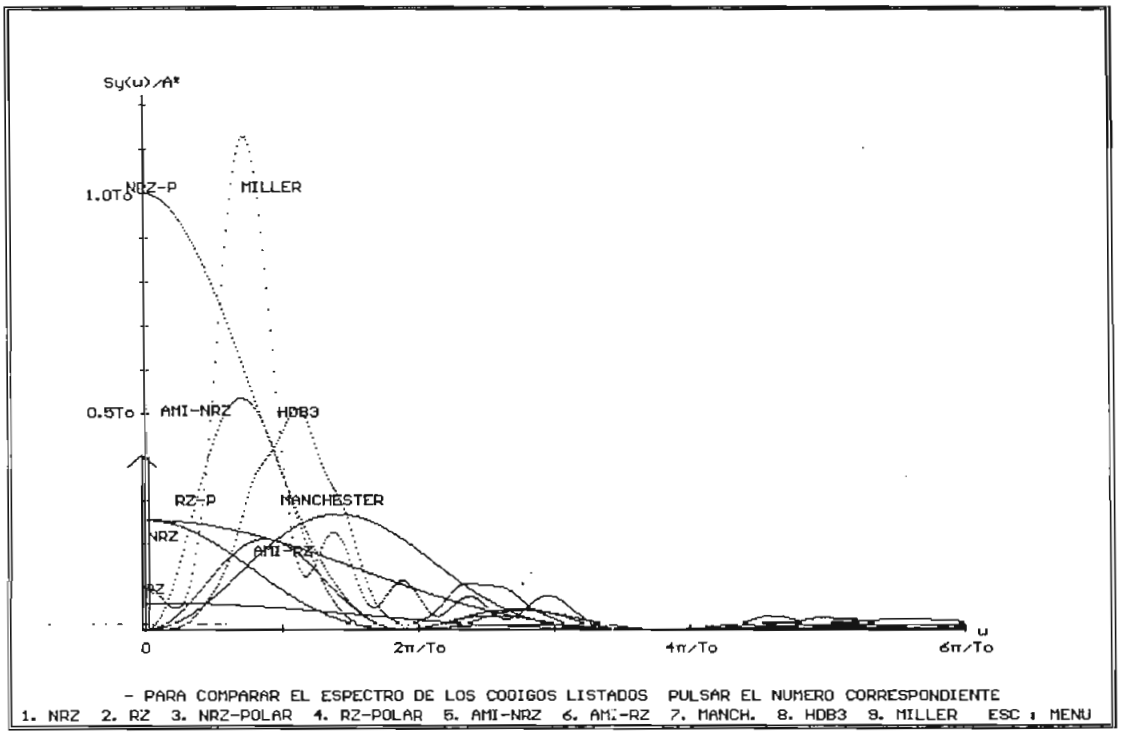


Figura 5.24. Comparación de las DEP de los códigos (P=0.5)

5.3. COMPARACION DE LOS ESPECTROS DE LOS DIFERENTES CODIGOS DE LINEA

- El espectro de la codificación RZ consta tanto de una parte continua como de una discreta. Está presente una componente discreta a la frecuencia de reloj, mientras que para la codificación NRZ la componente discreta de la frecuencia de reloj se anula y el ancho de banda se reduce a la mitad.

- La codificación de encendido-apagado (RZ o NRZ) es atractiva desde el punto de vista de simplicidad del circuito terminal, pero presenta algunas desventajas. Para una potencia de transmisión dada, es menos inmune a la interferencia de ruido que el esquema polar, el cual utiliza un pulso positivo para 1_L y un pulso negativo para 0_L. Esto se debe a que la inmunidad al ruido depende de las amplitudes que representan a 1_L y 0_L.

- La codificación polar es más eficiente que la de encendido-apagado. De hecho, para una potencia de transmisión dada, la codificación polar es el esquema de mayor eficiencia, además de ser transparente. Se debe observar que en la codificación polar no existe componente discreta a la frecuencia de reloj, sin embargo, la rectificación de la señal RZ polar da como resultado una señal periódica de frecuencia de reloj de la cual se puede extraer la sincronización.

- La señal bipolar presenta varias ventajas: 1) su espectro tiene una DC nula; 2) su ancho de banda no es excesivo; 3) tiene capacidad de detección de un solo error. Esto se debe a que si se hace la detección de un solo error, se ocasionará una violación bipolar de la regla del pulso alternante y se detectará de inmediato. Si se rectifica una señal bipolar, se obtiene una señal de encendido-apagado que tiene una componente discreta en la frecuencia de reloj.

- Una desventaja de la señal bipolar es que requiere dos veces tanta potencia (3 dB) de la que se requiere para una señal polar, siendo además la bipolar no transparente.

- El ancho de banda para la señal de fase dividida es dos veces el de la señal bipolar. Sin embargo, la fase dividida presenta una ventaja sobre la bipolar en que es transparente, ya que cada posición de pulso se encuentra ocupada y una redundancia de ceros no dificultará la extracción de la señal de reloj.

5.4. PROGRAMA PARA ANALISIS DE LOS ESPECTROS DE FRECUENCIA DE LOS TIPOS DE MODULACION

El programa para análisis de los espectros de frecuencia para los diversos tipos de modulación, en forma similar a la codificación, es estructurado en base de procedimientos cuyo diagrama de flujo general se encuentra en la figura 6.12 del Capítulo 6.

5.4.1. DEP de la modulación ASK

De la ecuación de la modulación ASK equilibrada:

$$g(t) = b_n(t) . Cos(\omega_o t) \qquad (5.27)$$

se tiene que $b_n(t)$ es 1 o 0, siendo ésta la modulación OOK. La DEP para la modulación ASK será la misma que la DEP del código NRZ con un pulso de ancho completo desplazado a $\pm\omega_c$.

Por la propiedad de las series de Fourier para las señales moduladas:

$$S_y(\omega) = \frac{1}{2} [S_{b_n}(\omega-\omega_c) + S_{b_n}(\omega+\omega_c)] \qquad (5.28)$$

donde $S_{b_n}(\omega-\omega_c)$ es la DEP de un pulso rectangular de ancho T_o. desplazado a la frecuencia de la portadora.

Se tiene que la DEP de la modulación ASK para encendido-apagado y equiprobabilidad es:

$$S_y(\omega) = \frac{T_o}{8} Sinc^2\left(\frac{(\omega-\omega_c)T_o}{2\pi}\right)\left[1+\frac{2\pi}{T_o}\sum_{n=-\infty}^{\infty}\delta\left((\omega-\omega_c).\frac{2\pi}{T_o}+\right.\right.$$

$$\left.\left.+\frac{T_o}{8}Sinc^2\left(\frac{(\omega-\omega_c)T_o}{2\pi}\right)\left[1+\frac{2\pi}{T_o}\sum_{n=-\infty}^{\infty}\delta\left((\omega+\omega_c)\frac{2\pi}{T_o}\right]\right.\right. \qquad (5.29)$$

En la figura 5.25 se ve este espectro. A partir de este gráfico se puede observar que el proceso de modulación traslada el espectro de frecuencia hacia arriba y hacia abajo de la frecuencia de la portadora ω_c.

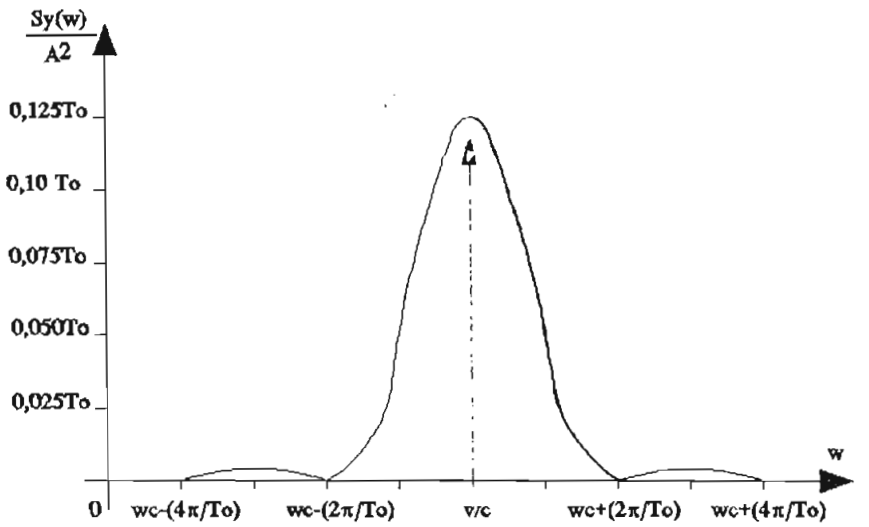


Figura 5.25. DEP de la modulación ASK-OOK

Se debe notar que para una señal aleatoria NRZ su ancho de banda es B, el ancho de banda con ASK es el doble que aquel, es decir 2B hertzios, o ±B hertzios alrededor de la frecuencia portadora.

5.4.2. DEP de la modulación FSK

Aqui los datos se transmiten variando la frecuencia, el estado 1_L se transmitirá con una frecuencia f_1 y el 0_L con una frecuencia f_2; es por esto que la modulación FSK puede verse como la suma de dos señales ASK intercaladas según la ecuación:

$$g(t) = Cos(\omega_c + \frac{b_n(t) \cdot \Delta\omega}{2}) t \qquad (5.30)$$

Generalmente $\omega_1 = 2\pi f_1$ y $\omega_2 = 2\pi f_2$ son frecuencias

mayores que $(2\pi/T_o)$ (ancho de banda de señal en banda base NRZ).

Suponiendo que las dos frecuencias son múltiplos del recíproco del período binario T_o. Es decir $f_1=m/T_o$, $f_2=n/T_o$ (donde m y n son enteros) y están sincronizadas en fase, la señal modulada puede también visualizarse como la superposición lineal de dos ondas periódicas de OOK. La frecuencia entonces se desvía $\pm\Delta\omega$ alrededor de ω_c, lo que se denomina comúnmente desviación de frecuencia.

$$\omega_1=\omega_c-\Delta\omega$$

$$\omega_2=\omega_c+\Delta\omega$$

La DEP de FSK tiene la siguiente ecuación:

$$S_{b_n}(\omega_1-\omega_c)=\frac{T_o}{4}Sinc^2\left(\frac{(\omega_1-\omega_c)}{2\pi}T_o\right)$$

$$S_{b_n}(\omega_1+\omega_c)=\frac{T_o}{4}Sinc^2\left(\frac{(\omega_1+\omega_c)}{2\pi}T_o\right)$$

$$S_{b_n}(\omega_2-\omega_c)=\frac{T_o}{4}Sinc^2\left(\frac{(\omega_2-\omega_c)}{2\pi}T_o\right)$$

$$S_{b_n}(\omega_2+\omega_c)=\frac{T_o}{4}Sinc^2\left(\frac{(\omega_2+\omega_c)}{2\pi}T_o\right) \qquad (5.31)$$

$$S_f(\omega)=\frac{1}{2}\left[S_{b_n}(\omega_1-\omega_c)+S_{b_n}(\omega_1+\omega_c)\right]+\frac{1}{2}\left[S_{b_n}(\omega_2-\omega_c)+S_{b_n}(\omega_2+\omega_c)\right]$$

El espectro de FSK es una superposición lineal de dos espectros OOK como se muestra en la figura 5.26.

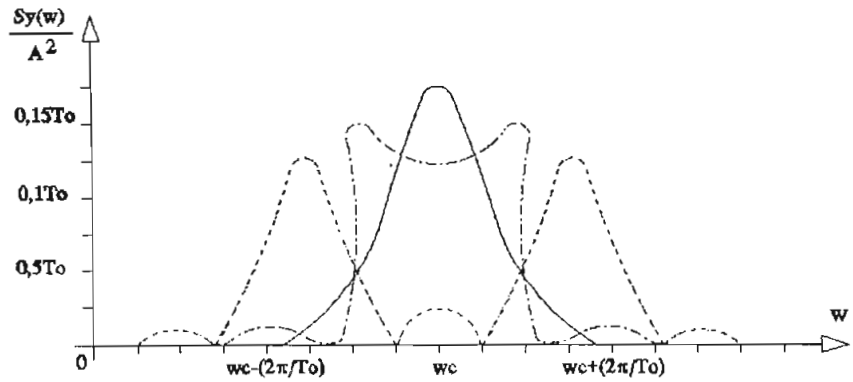


Figura 5.26. DEP de la modulación FSK para diferentes valores de desviación de frecuencia $\Delta\omega$

Se puede observar que en el espectro de FSK no aparecen componentes discretas cuando se elige apropiadamente ω_1 y ω_2, el ancho de banda de FSK es mayor que el de ASK o que el de 2-PSK.

5.4.3. DEP de la modulación 2-PSK

En este caso se transmite un 1_L con un pulso $Cos\,\omega_c t$ y 0_L como -Cos $\omega_c t$, en consecuencia los dos pulsos se encuentran separados π radianes en fase; es decir que la modulación 2-PSK da una señal $y(t)Cos(\omega_c t)$ donde $y(t)$ es una señal polar.

Se concluye entonces que la DEP de una señal 2-PSK es la misma que la de una señal polar de banda base desplazada a $\pm\omega_c$.

Partiendo de la ecuación de la modulación para 2-PSK:

$$g(t) = Cos(\omega_c . t \pm \frac{\pi}{2}) \qquad (5.32)$$

La DEP de 2-PSK tiene la forma:

$$S_y(\omega) = \frac{1}{2}\left[S_{b_n}(\omega-\omega_c) + S_{b_n}(\omega+\omega_c)\right]$$

$$= \frac{1}{2}\left[T_o Sinc^2\left(\frac{(\omega-\omega_c)}{2\pi}T_o\right) + T_o Sinc^2\left(\frac{(\omega+\omega_c)}{2\pi}T_o\right)\right] \qquad (5.33)$$

En la figura 5.27 se puede observar la DEP de esta clase de modulación, aunque la densidad espectral de potencia de la señal 2-PSK modulada aleatoriamente es más alta alrededor de la portadora, no hay líneas espectrales o impulsos en la frecuencia portadora. Por lo tanto la 2-PSK es realmente una técnica de modulación de doble banda lateral con portadora suprimida.

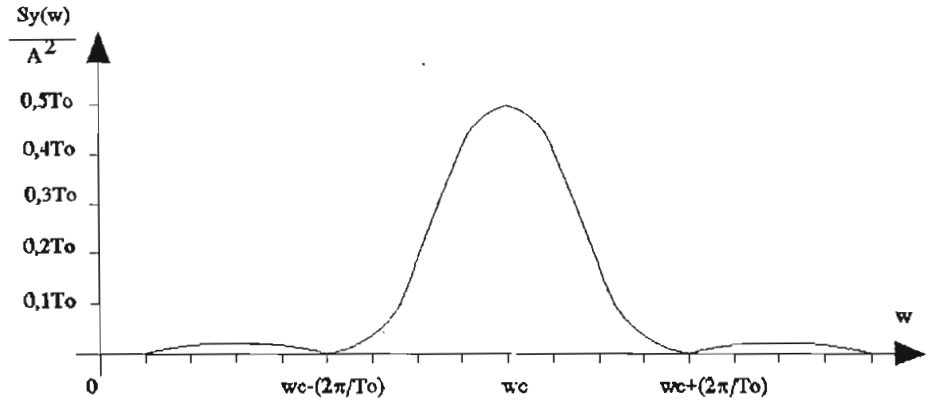


Figura 5.27. DEP de la modulación 2-PSK

5.4.4. DEP de la modulación M-PSK

Los métodos multinivel son de creciente interés para sistemas de comunicación digital que requieren manejar altos ritmos de transmisión razones de datos dentro de consideraciones fijas de ancho de banda.

La ecuación general de la modulación PSK M-aria es:

$$g(t) = Cos(\omega_c . t + (\frac{b_n(t)\,\delta\phi}{2})) \qquad (5.34)$$

La DEP unilateral para PSK M-aria, para una entrada aleatoria con 1_L y 0_L equiprobable es:

$$S_y(\omega) = T_s . Sinc^2(\frac{(\omega - \omega_c)}{2\pi} T_s) \qquad (5.35)$$

donde T_s es la duración de un símbolo unitario dado por:

$$T_s = T_o(Log_2 M) \qquad (5.36)$$

Se puede observar la DEP para M-PSK para diversos valores de M, en la figura 5.28.

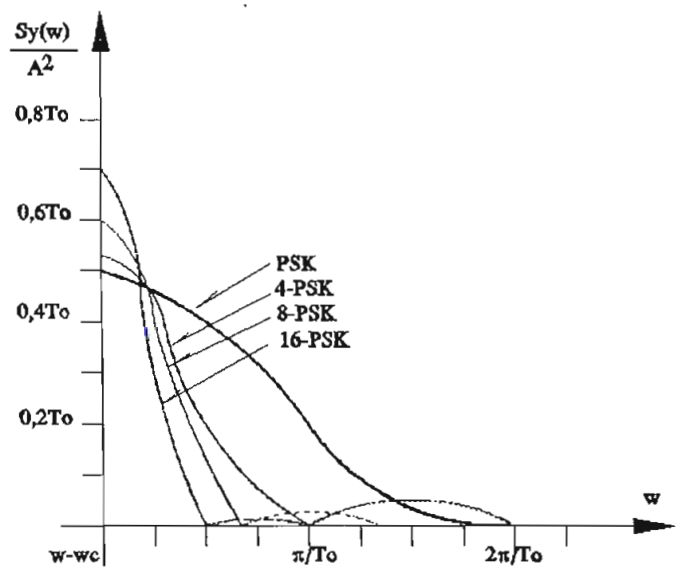


Figura 5.28. DEP de la modulación M-PSK

5.4.5. DEP de la modulación M-QAM

En muchas aplicaciones un sistema resulta más económico si, en un ancho de banda determinado, pueden transmitirse más bits por segundo; esto conduce a métodos de modulación M-aria en la que se transmite una de M posibles

señales durante cada intervalo de señalización.

Para aumentar la eficiencia espectral se utiliza el método de la multiplexión de cuadratura QAM, en donde se combinan dos señales moduladas en cuadratura de fase.

Partiendo de la ecuación matemática para QAM:

$$g(t) = a_i Cos(\omega_c t) + b_i Sen(\omega_c t) \qquad (5.37)$$

La DEP de una modulación M-QAM para una entrada de datos aleatoria NRZ y para valores de frecuencias positivas tiene la forma:

$$S_y(\omega) = C\left[\frac{Sen(\omega-\omega_c)\frac{T_s}{2}}{(\omega-\omega_c)\frac{T_s}{2}}\right]^2 \qquad (5.38)$$

donde C es proporcional a la potencia media M-QAM transmitida.

Para el caso de modulación 4-QAM, donde se tiene que $f_s = f_o/2$ ($T_s = 2T_o$) la eficiencia del ancho de banda es 2 (bps/Hz). En la figura 5.29 se observa esta DEP.

La modulación 4-QAM puede verse como una modulación 4-PSK si sus componentes en cuadratura tienen magnitudes iguales.

Teóricamente ambos sistemas de modulación tienen la misma densidad espectral de potencia, debido a esta similitud y a la popularidad de los sistemas 4-PSK, este término se utiliza con bastante frecuencia también para 4-QAM.

En las figuras 5.30 hasta la 5.37 se presentan los espectros de potencia para las modulaciones analizadas; y en la figura 5.38 se tiene un esquema de modulaciones M-PSK, el mismo que permite comparar sus espectros cuando varía el número de fases.

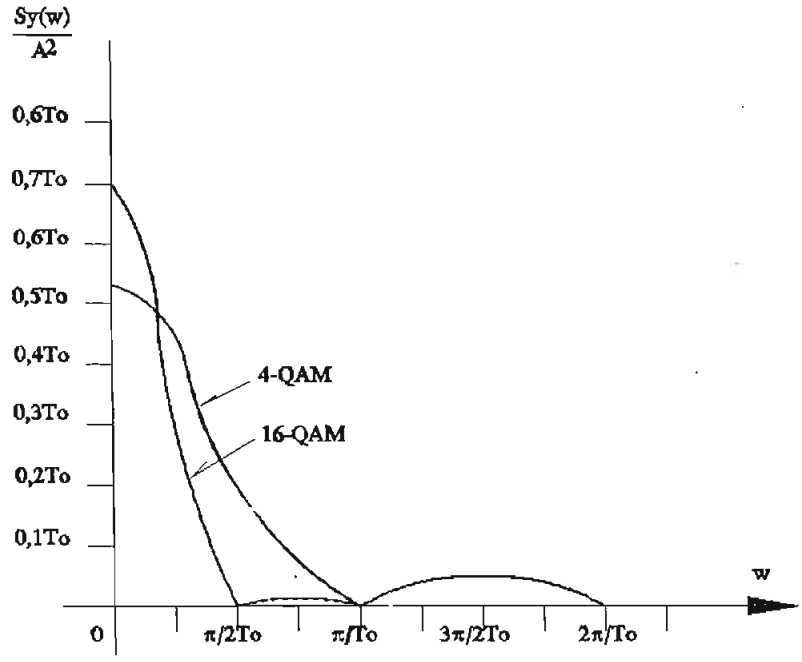


Figura 5.29. DEP de la modulación M-QAM

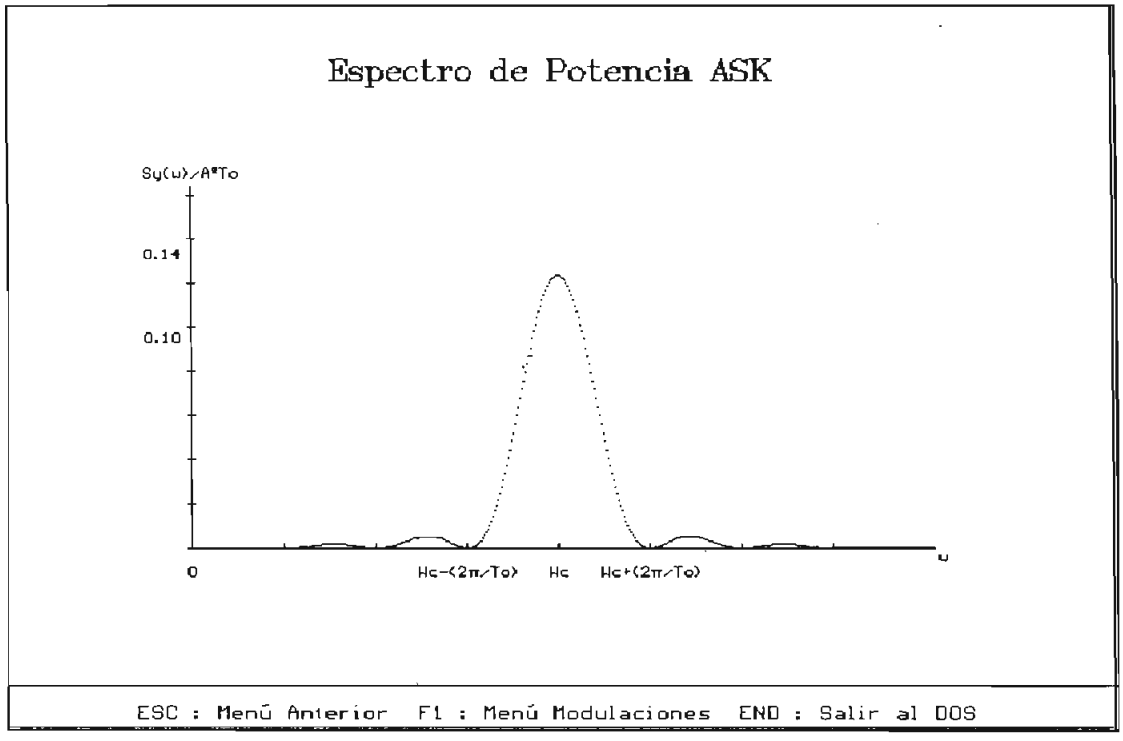


Figura 5.30. DEP de modulación ASK

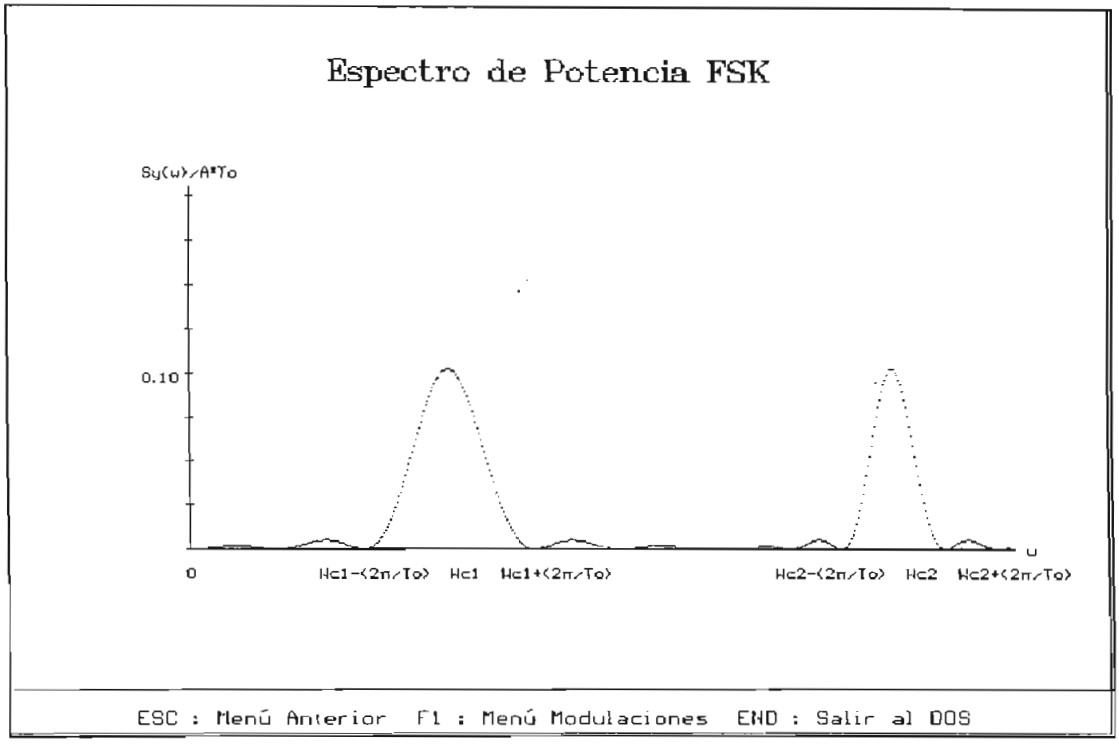


Figura 5.31. DEP de modulación FSK

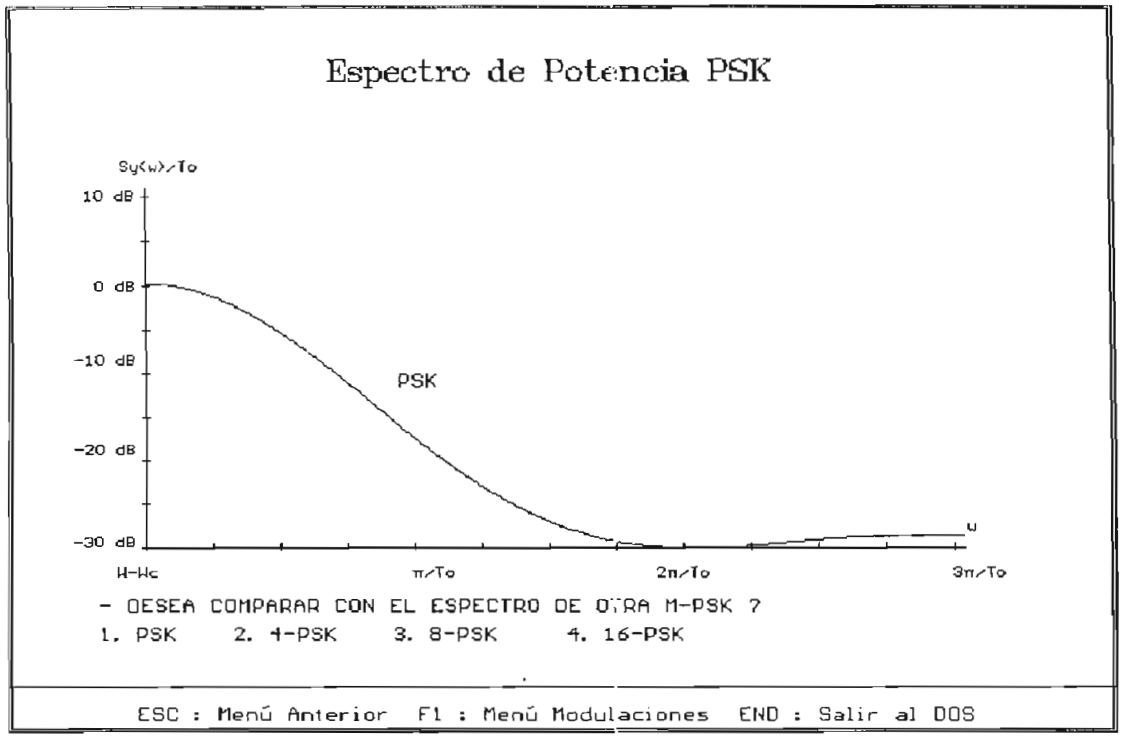


Figura 5.32. DEP de modulación 2-PSK

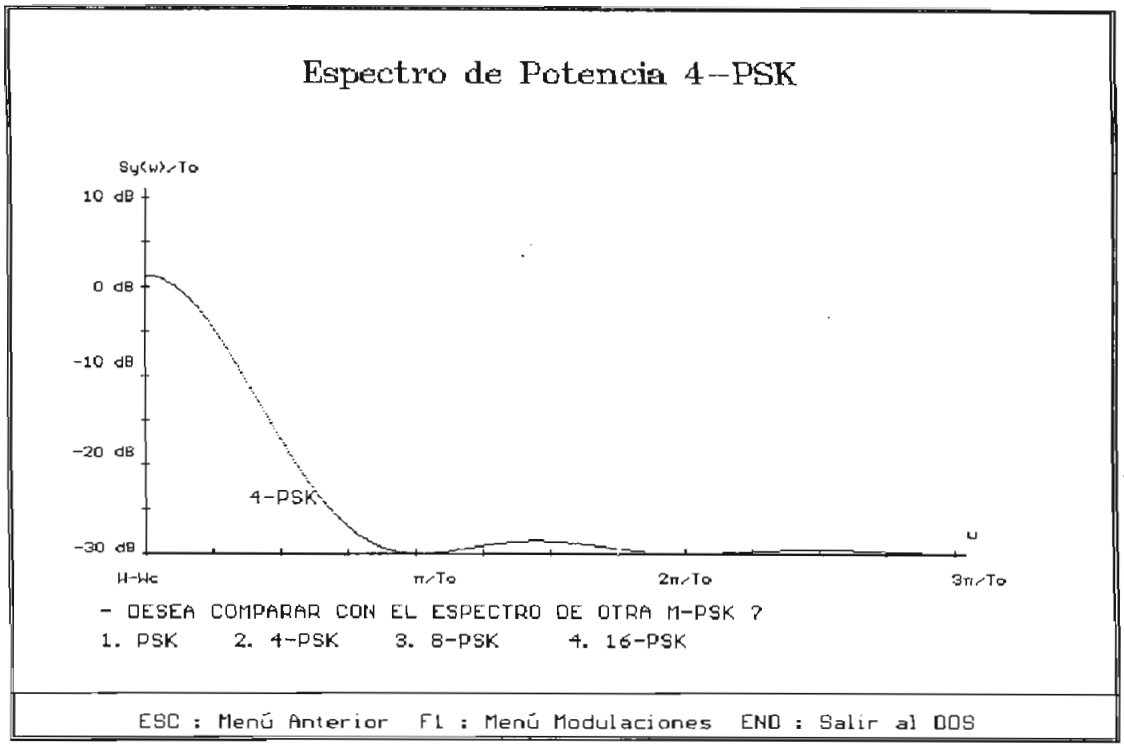


Figura 5.33. DEP de modulación 4-PSK

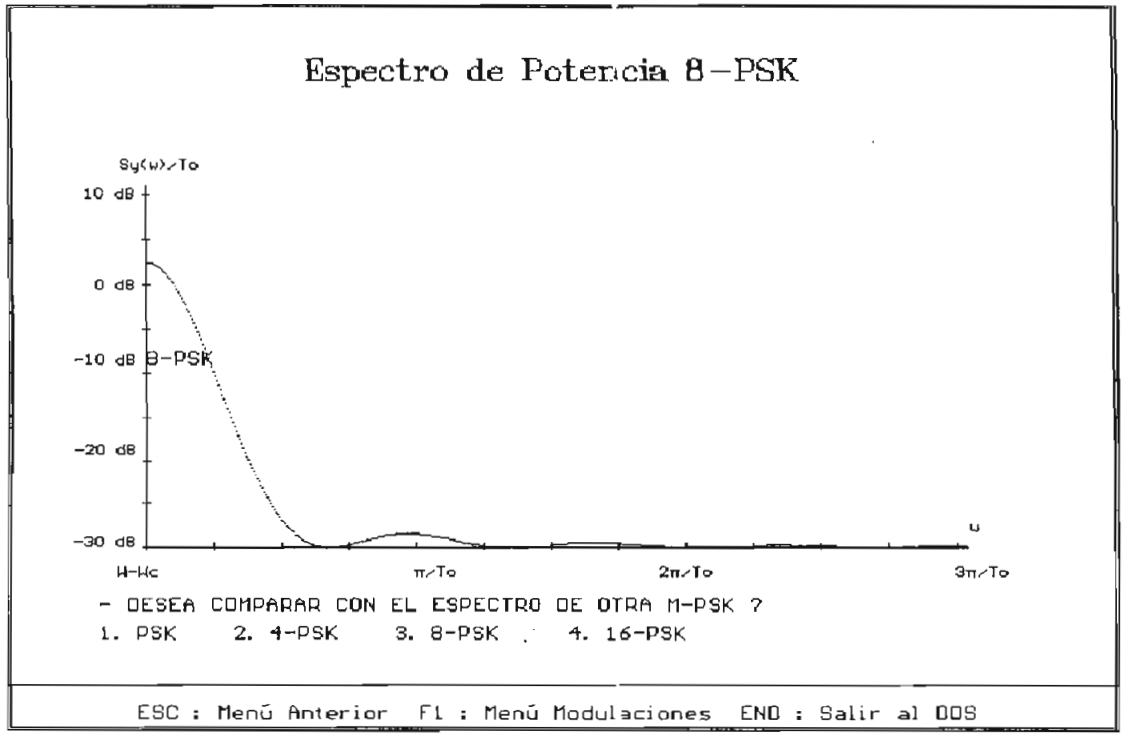


Figura 5.34. DEP de modulación 8-PSK

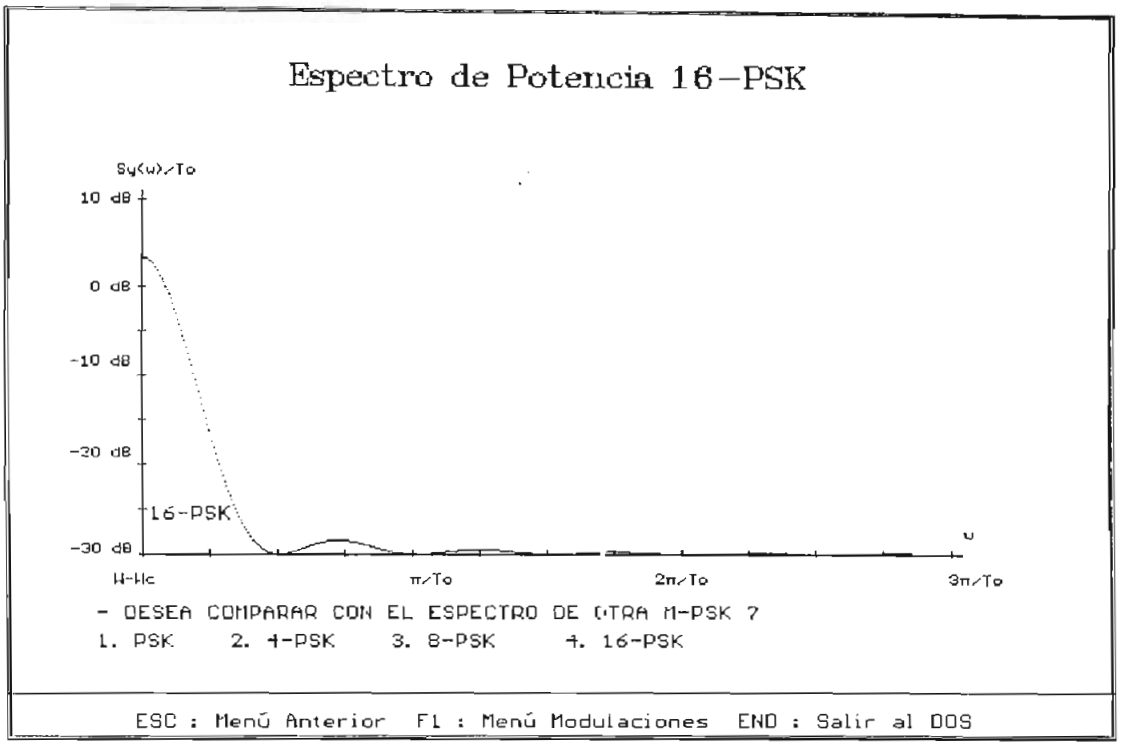


Figura 5.35. DEP de modulación 16-PSK

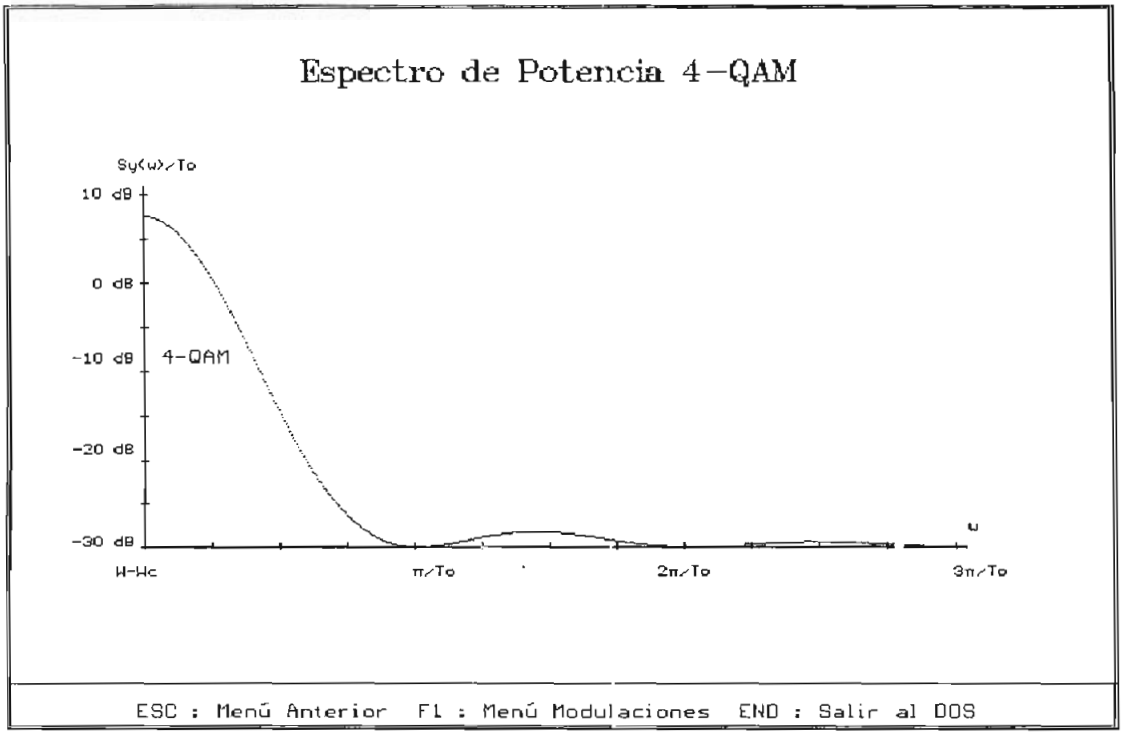


Figura 5.36. DEP de modulación 4-QAM

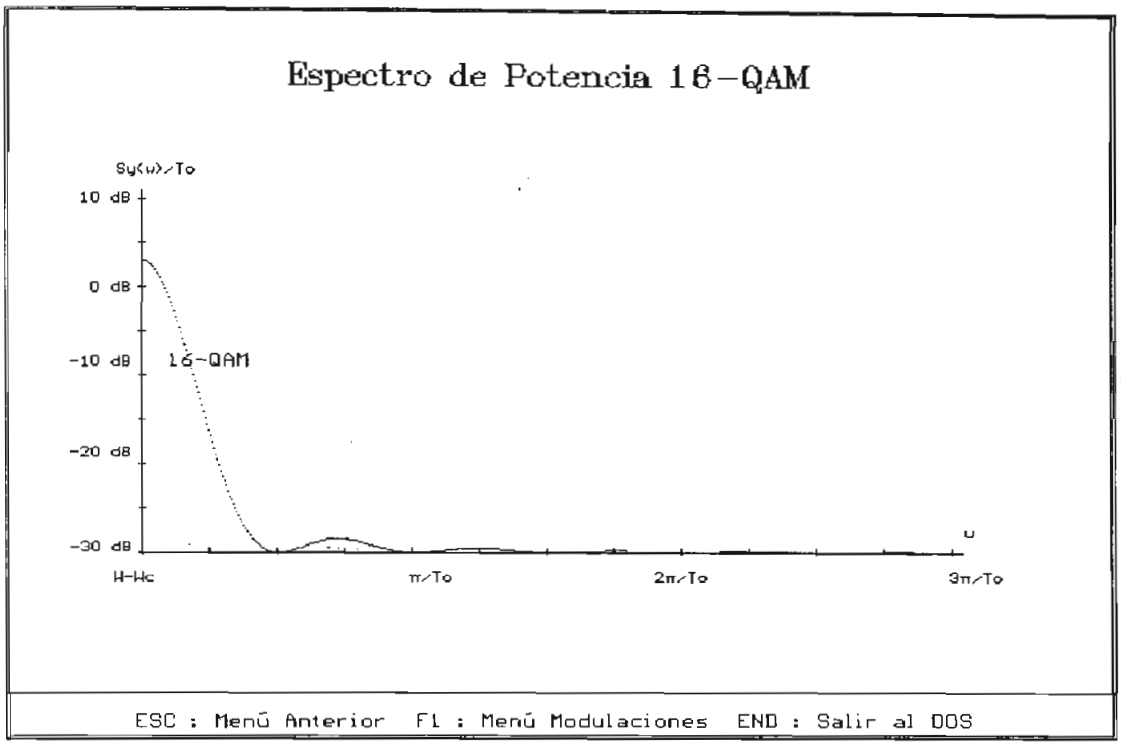


Figura 5.37. DEP de modulación 16-QAM

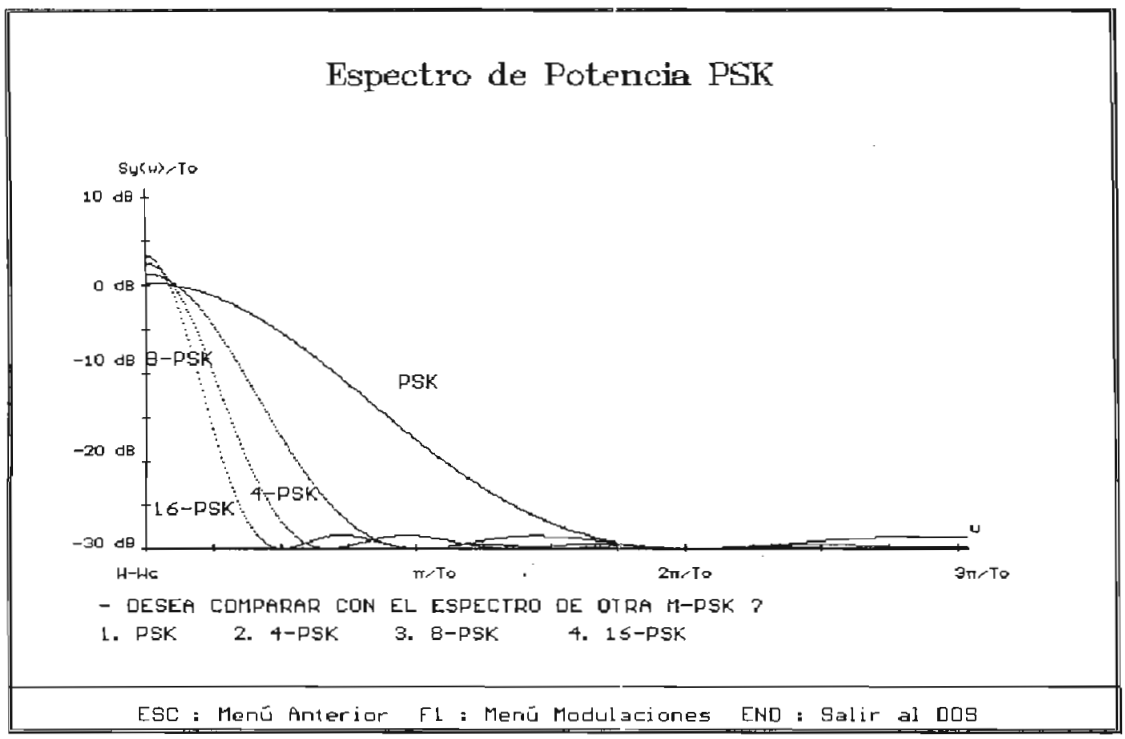


Figura 5.38. Comparación de las DEP de modulaciones M-PSK

5.5. COMPARACION DE LOS ESPECTROS DE LAS DIFERENTES MODULACIONES

Para la comparación de los diversos espectros de los diferentes tipos de modulación, se debe suponer que una transmisión ideal no tendrá un contenido de energía significativo a bajas o altas frecuencias. Básicamente la conformación del espectro se realizará bajo tres parámetros:

- La forma del pulso
- La tasa de repetición del pulso
- La cantidad de niveles lógicos que pueden ser transmitidos.

Si se quiere reducir la cantidad de energía a altas frecuencias de la señal modulada, se debe variar el número de niveles a transmitir. Esto puede ser logrado con la modulación multinaria, que transmite la misma información que en el caso binario pero que reduce la tasa de repetición del pulso. Los sistemas multinarios requieren circuitos complejos por lo que solo son usados para transmisión en alta capacidad.

Del análisis de las DEP para las modulaciones se concluye:

- El espectro para la modulación ASK (OOK) es el mismo que el de la señal en banda base NRZ , trasladado a la frecuencia de la portadora.

  En cuanto al ancho de banda éste se duplica con respecto al de la señal en banda base NRZ. El espectro de la modulación tiene una componente discreta (impulso) a la frecuencia de la portadora.

- El espectro de la señal FSK puede visualizarse como la superposición de dos señales OOK desplazadas relativamente en frecuencia. No aparecen componentes discretas en este espectro. Se puede demostrar que eligiendo adecuadamente w_1 y w_2 se pueden eliminar las componentes discretas.

- En general se puede decir que el ancho de banda de la

modulación FSK es mayor que para ASK o para 2-PSK

- Ninguno de los métodos ASK, FSK y 2-PSK es particularmente eficiente en términos de ancho de banda; la elección del tipo de modulación digital dependen de los intercambios entre rendimiento, costos, ancho de banda, etc.

- El espectro de la modulación 2-PSK tiene la misma característica de doble banda lateral que la modulación OOK, con la importante ausencia de un impulso en la frecuencia de la portadora. La ausencia de una componente discreta en la modulación 2-PSK le da una mejor eficiencia de potencia que OOK aunque igual eficiencia espectral.

- Los requerimientos de ancho de banda de una señal 2-PSK son los mismos que una ASK, a pesar que este último proceso de modulación es lineal, y el primero en general no lo es. En 2-PSK se puede observar un lóbulo principal cuyo ancho es el doble de la tasa de señalización, notándose además que los lóbulos secundarios decrecen muy lentamente, lo cual hace necesario filtrar la señal de salida para evitar interferencia en los sistemas adyacentes. Esto reduce la potencia transmitida por lo que normalmente debe incrementarse la potencia del transmisor a efectos de mantener la calidad del sistema.

- En la modulación multisimbólica M-PSK, tomando como base el espectro de la modulación 2-PSK, el ancho de banda requerido para 4-PSK es la mitad del 2-PSK a igual velocidad de señalización. El ancho de banda de la modulación 8-PSK es la tercera parte que 2-PSK, es decir que para 16-PSK el espectro se reduce en cuatro veces. En conclusión en los sistemas M-PSK, a medida que aumenta el número de estados, en ancho de banda se reduce con respecto a los 2-PSK en un factor $\log_2 M$.

- Se debe mencionar que el esquema de modulación 4-PSK es el más difundido; sin embargo cuando la reducción de ancho de banda es el objetivo primordial será necesario recurrir a

métodos de modulación más complejos.

- Los espectros de las señales QAM se determinan como en todos los moduladores de producto, a partir de las señales de banda base aplicadas a los canales en cuadratura. Como estas señales presentan estructuras similares a los M-PSK los espectros M-QAM son similares a los M-PSK de igual orden, concretamente el espectro de 16-PSK será equivalente a 16-QAM.

- Como conclusión final, se puede decir que en las diversas técnicas de modulación se ha analizado con más detenimiento los anchos de banda necesarios en la transmisión, es decir que el ancho de banda viene a ser una de las limitaciones fundamentales en los sistemas de comunicaciones para mantener la distorsión y en consecuencia la interferencia intersimbólica por debajo de límites aceptables. Cabe mencionar que otra de las limitaciones fundamentales lo constituye el ruido el cual combinado con la atenuación en su trayecto transmisor-canal de transmisión-receptor, conspira contra la eficiente transmisión de la información.

CAPITULO 6

PROGRAMA GENERAL PARA ANALISIS Y SIMULACION DE UN SISTEMA DE TRANSMISION DIGITAL

6.1. INTRODUCCION

La simulación en una computadora digital es una herramienta muy útil para análisis y diseño de sistemas de telecomunicaciones; en muchos casos es utilizada en conjunto con un análisis teórico y pruebas del equipo (hardware) en el laboratorio.

Un modelo de simulación puede consistir de un programa bastante complejo, escrito en un lenguaje de propósito general diseñado propiamente para el análisis del sistema de comunicación que se desea estudiar; también puede consistir de un simple programa que utiliza un paquete de software, escrito en un lenguaje de alto nivel (FORTRAN, PASCAL, C, etc.). Para la descripción de la operación del sistema, esta última alternativa es la utilizada en esta tesis.

Un paquete de simulación debe ser muy versátil, esto es, debe posibilitar la simulación de muchos sistemas diferentes y permitir modificación fácil de los parámetros del sistema. Sin embargo el requerimiento más importante es que el paquete diseñado sea muy fácil de usar.

El programa desarrollado en esta tesis, es un paquete de "software" para el análisis y simulación de los Códigos de Línea y de la Modulación Digital de un Sistema de Transmisión Digital, basado en un lenguaje simple para la descripción del sistema.

6.2. DESCRIPCION Y ALCANCES DEL PROGRAMA

El programa para análisis y simulación de un sistema de transmisión digital consiste de cinco unidades, cada una de ellas contiene una serie de procedimientos o subrutinas de soporte; las unidades se enlazan mediante un programa principal.

Tanto las unidades como el programa principal están escritos en Turbo Pascal en la versión 6.0, bajo el sistema operativo DOS.

El programa está dividido en dos partes: Codificación y Modulación Digital; cada una de éstas permite la simulación y el análisis de los conceptos básicos de las diversas técnicas descritas en los capítulos 3, 4 y 5 de este trabajo.

La simulación persigue presentar en forma gráfica los diferentes tipos de códigos de línea, a partir de una secuencia de datos binaria para cada uno los tipos de codificación descritos. Los resultados se muestran en forma gráfica en la pantalla del monitor, en la que se pueden observar la señal de reloj, la señal de datos binaria y la señal codificada.

También se dispone de la teoría correspondiente a cada código, como soporte para una mayor comprensión de las reglas de codificación utilizados. Adicionalmente como parte de la codificación se incluye un ejemplo para cada tipo de codificación.

Para el análisis se presentan los gráficos de las funciones de densidad espectral de potencia de la mayoría de los códigos descritos; este análisis se complementa con la comparación de los espectros de potencia de los códigos posibles, que junto a la teoría dada en el capítulo 5 permiten una mayor comprensión de las características de los códigos.

Un esquema similar al de la codificación se tiene para la modulación digital. Para los gráficos de los tipos de

modulación, se presentan en la pantalla la señal de datos binarios (señal modulante), la señal portadora y la señal modulada correspondiente, incluyendo adicionalmente con la teoría el ejemplo respectivo.

El análisis de las diferentes técnicas de modulación es más complejo y generalmente se lo hace comparando el comportamiento de la probabilidad de error de la señal modulada en función de la variación de la señal con respecto al ruido, aquí se lo ha hecho en base de los espectros de potencia. En el programa la comparación de los espectros, se la realiza exclusivamente para el tipo de modulaciones M-PSK.

La figura 6.1 muestra el diagrama de flujo del Programa Principal, este programa permite seleccionar una de las dos opciones: Códigos de Línea o Modulación Digital; y dentro de éstas opciones sus diferentes tipos o clases. Una vez que se ha escogido la clase de código o modulación, se elige el tipo de aplicación que se desea observar. Una vez que finaliza una aplicación determinada el programa puede continuar con otra aplicación o terminar (salir al sistema operativo DOS).

6.2.1. Programa para Códigos de Línea

Previa a la elección de una aplicación determinada de los Códigos de Línea, se tienen dos procedimientos que permiten hacer la selección.

El procedimiento **CuadroCódigos**, es una pantalla de selección de la clase de código, en ella se listan los códigos disponibles desde la A a la N. Con la letra S se puede seleccionar el análisis de los espectros de potencia.

Otro procedimiento **CuadroSelecciónCódigos** presenta una pantalla en la que se debe elegir entre Codificación y Teoría, las cuales son aplicaciones disponibles para todos los códigos.

Para la **Codificación** de una señal binaria el programa

necesita primero leer los bit de datos, para luego codificarlos y finalmente graficar. A continuación se explican los procesos que se siguen en el programa para la codificación de una señal.

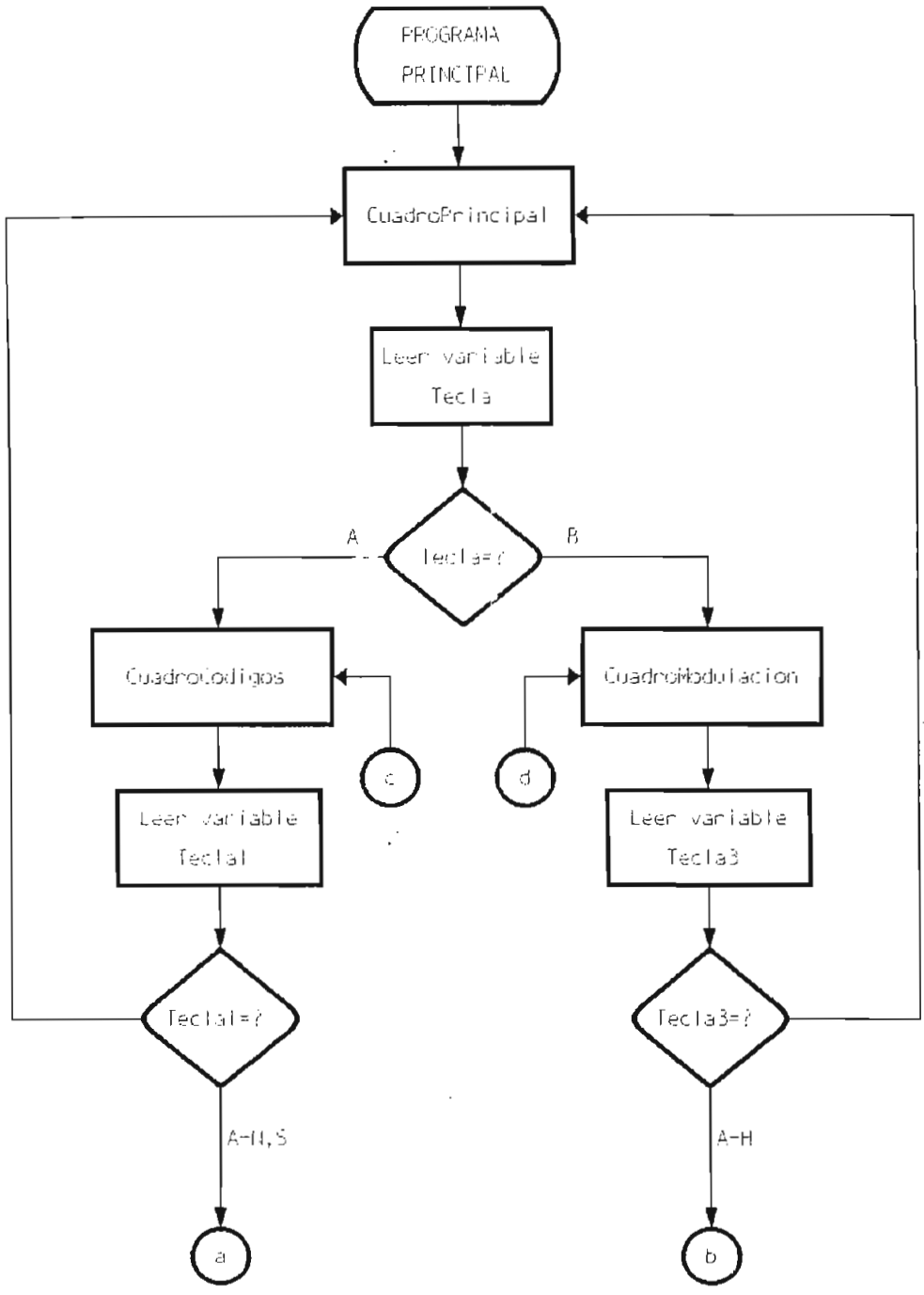


Figura 6.1. Diagrama de flujo del Programa Principal

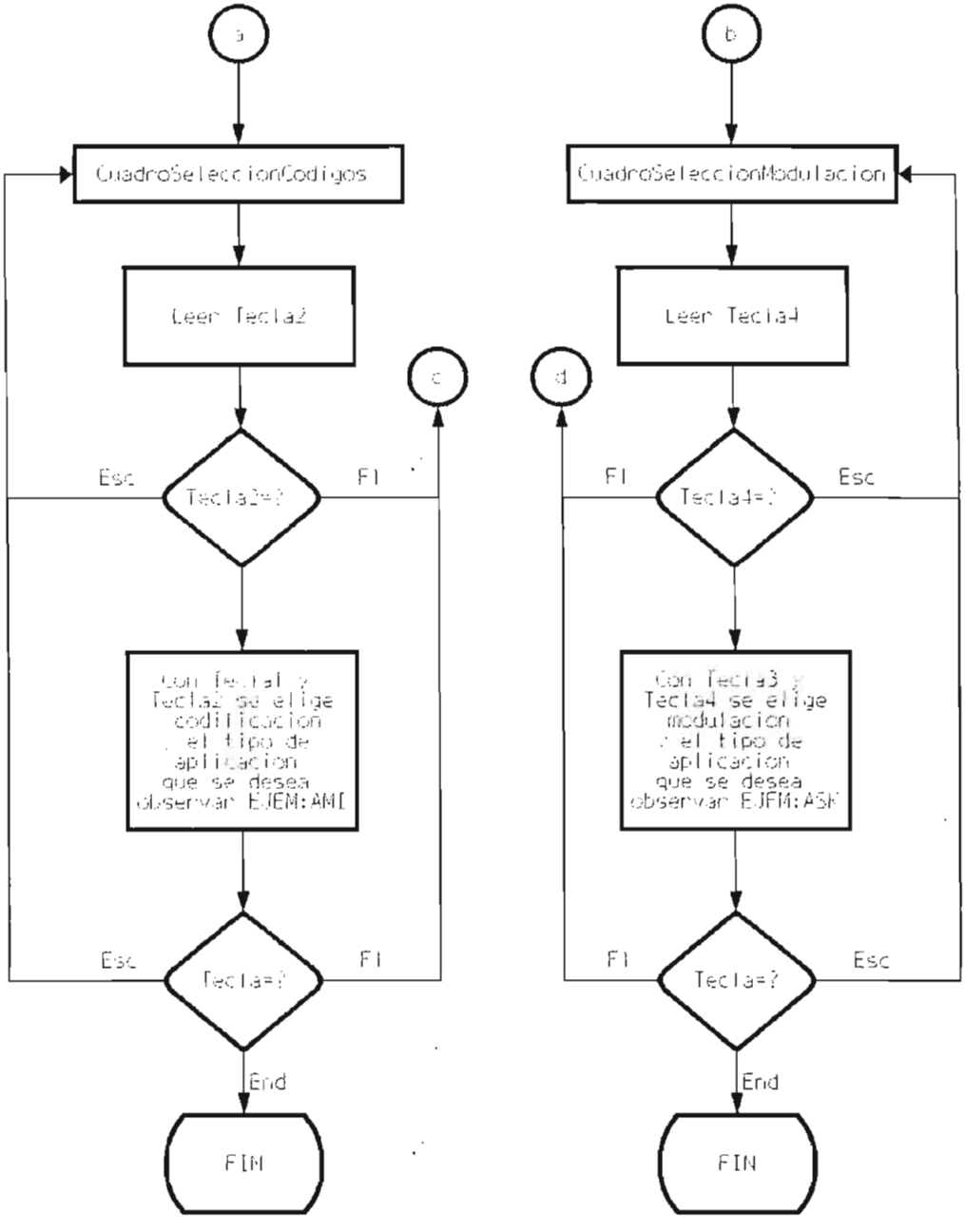


Figura 6.1. Diagrama de flujo del Programa Principal (cont.)

- **Ingreso de datos al programa para codificación:**

Como se mencionó en el capítulo 3, los datos de mayor importancia en el programa son los bits de datos (señal binaria). En el programa el ingreso de los bits de datos se lo hace de dos maneras: en forma manual o por la generación automática de ellos.

El ingreso manual de datos se lo hace utilizando el procedimiento **IngresoDatos** y la generación de datos se la realiza con el procedimiento **Aleatorio**; los datos obtenidos se almacenan en el vector **bit**. En el programa se debe ingresar como dato el número de bits **N**, para ambos casos.

El número de bits N es mínimo tres (3) y máximo veinte (20). Se han considerado estos límites, ya que para la comprensión de la codificación se necesitan más de tres bits; además para un número mayor a veinte bits se dificulta la visualización en el monitor y no cumple su función didáctica.

MantenerDatos, es un procedimiento que muestra los datos anteriores en el caso de que la codificación no se ejecute por primera vez y da la opción de cambiarlos, para lo cual se utilizan los procedimientos anteriores de ingreso de datos.

- **Codificación :**

Luego que se tienen los bits, se procede a codificarlos, cada una de las clases de códigos tiene su propio algoritmo para la codificación de las señales. En el capítulo 3, en el numeral 3.5 se desarrollaron los algoritmos de codificación utilizados para los procedimientos de codificación de cada uno de los códigos, y en los mismos que se indican las variables que utilizan cada uno de éstos y su respectivo diagrama de flujo.

A continuación, en la tabla 6.1 se listan los códigos con sus correspondientes procedimientos de codificación:

CODIGOS	PROCEDIMIENTO
NRZ	NRZ1
RZ	RZ1
NRZ POLAR	NRZPOLAR1
RZ POLAR	RZPOLAR1
AMI	AMI1
HDB3	HDB31
MANCHESTER	MANCHESTER1
BIFASE M	BIFASEM1
BIFASE S	BIFASES1
4B3T	C4B3T1
CMI	CMI1
PST	PST1

Tabla 6.1 Procedimientos de Codificación

- Graficar :

Los procedimientos indicados en la tabla anterior, dan los valores de los coeficientes que permiten graficar la señal codificada. Los valores de los coeficientes obtenidos se guardan en el vector **coef**.

De igual manera en el capítulo 3 se indica el proceso empleado para graficar un pulso utilizando los coeficientes **coef[i]**. El procedimiento **Graficar** es el encargado de dibujar las tres señales indicadas.

Para graficar la señal de reloj, los coeficientes están en vector **reloj**; mientras que los coeficientes de la señal de datos están en el vector **dato**, y son los valores de los datos ingresados.

El código 4B3T tiene su propio procedimiento **Graficar4B3T** el cual permite esquematizar las señales, debido a que su patrón de codificación es diferente a los anteriores y no es posible utilizar el procedimiento general **Graficar**.

A manera de ejemplo en la figura 6.2 se muestra el diagrama de flujo de la aplicación **Codificación** para el código **AMI**, en el que constan los procedimientos explicados anteriormente.

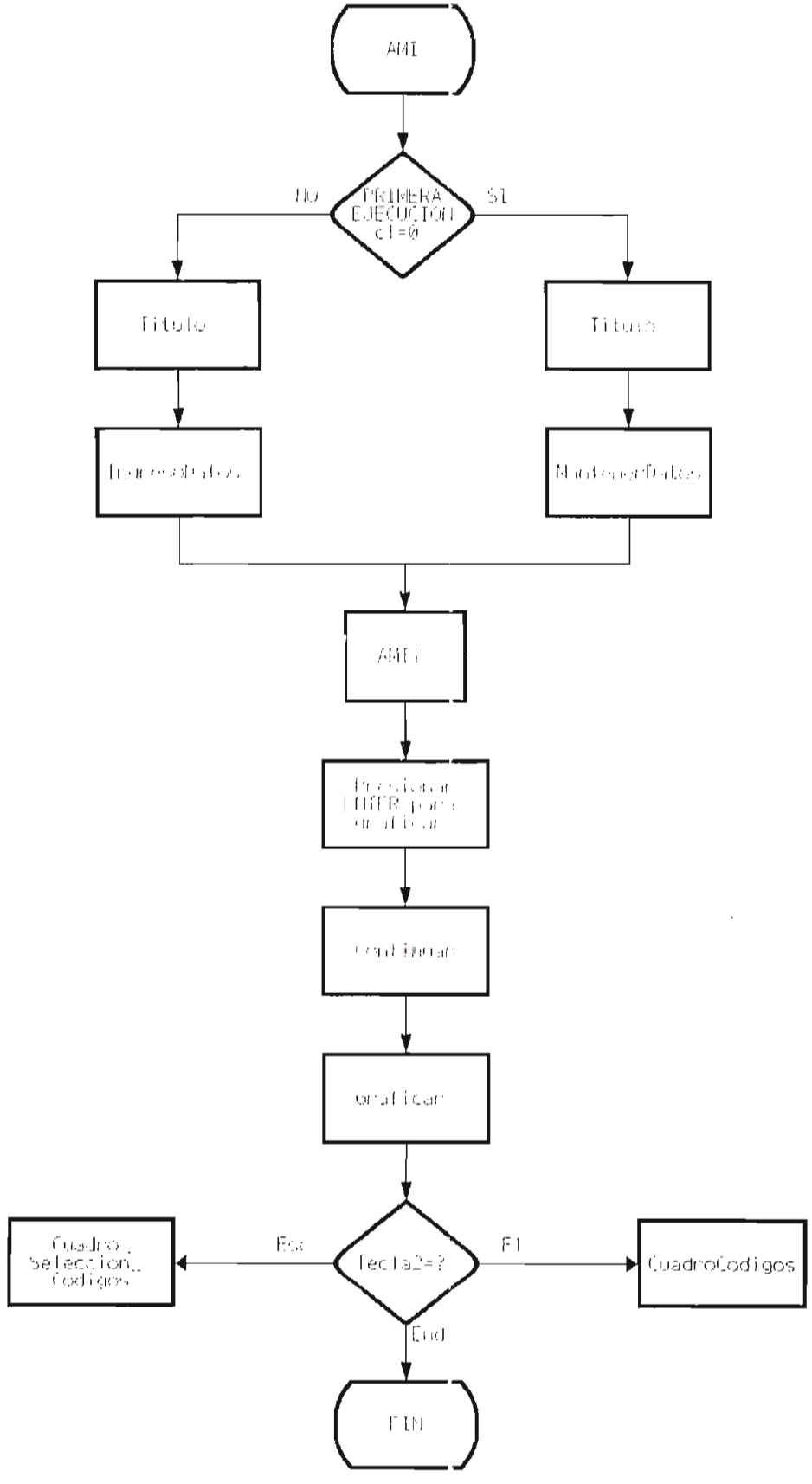


Figura 6.2. Diagrama de flujo del Procedimiento AMI
(Para el gráfico de las señales)

- **Teoría de los Códigos de Línea :**

Para cada uno de los códigos, se establecen procedimientos que presentan en modo gráfico, el texto de la teoría correspondiente. Por ejemplo el código AMI utiliza el procedimiento **AMITEORIA**.

Para la teoría se tiene aproximadamente ocho líneas de texto en las que se resumen: el concepto, las principales características y la regla de codificación. Para que la regla de codificación sea más comprensible se añade un ejemplo de la codificación.

Para la realización del ejemplo, a cada código se le asignó una secuencia fija, en la que se puede apreciar las características propias de cada una de las clases de códigos. Así por ejemplo para el código B3ZS se da una secuencia de bits que contiene más de tres ceros consecutivos.

Con esta secuencia, se calculan los coeficientes **coef[i]** de codificación con los mismos procedimientos utilizados en **Codificación**, luego de lo cual se proceden a graficar la señales. El procedimiento **GraficarEjemplo** realiza los gráficos de la secuencia de bits y de la señal codificada, de manera similar al procedimiento **Graficar**.

En la figura 6.3, como ejemplo, se presenta el diagrama de flujo para el procedimiento **AMITEORIA**.

- **Espectros de Potencia de los Códigos :**

Para realizar el análisis de las características de los Códigos de Línea, se recurre al estudio de las densidades espectrales de potencia y la comparación entre ellas.

El diagrama de flujo del procedimiento **ESPECTROSDE POTENCIA**, se lo puede observar en la figura 6.4.

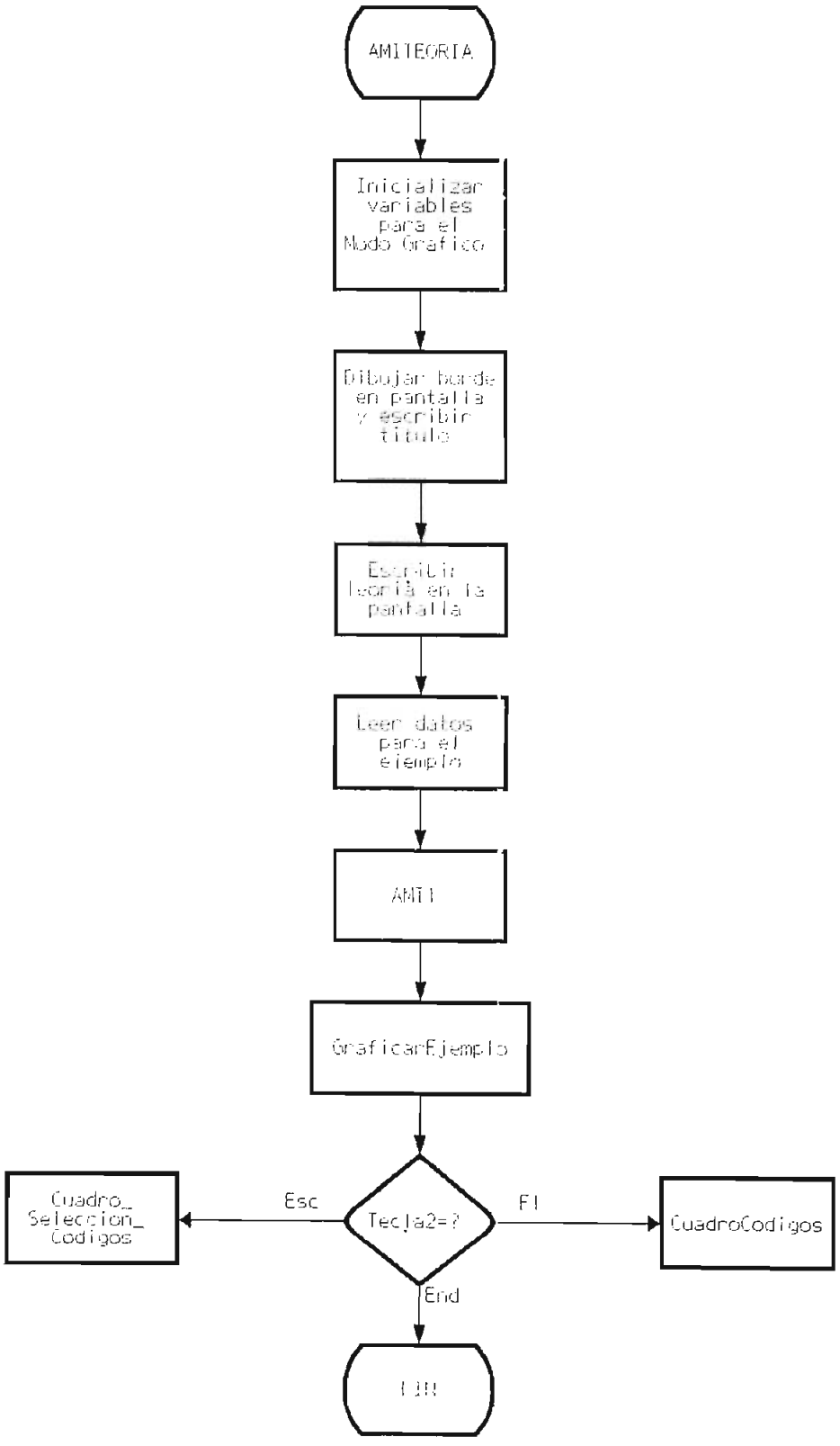


Figura 6.3. Diagrama de flujo del Procedimiento AMITEORIA

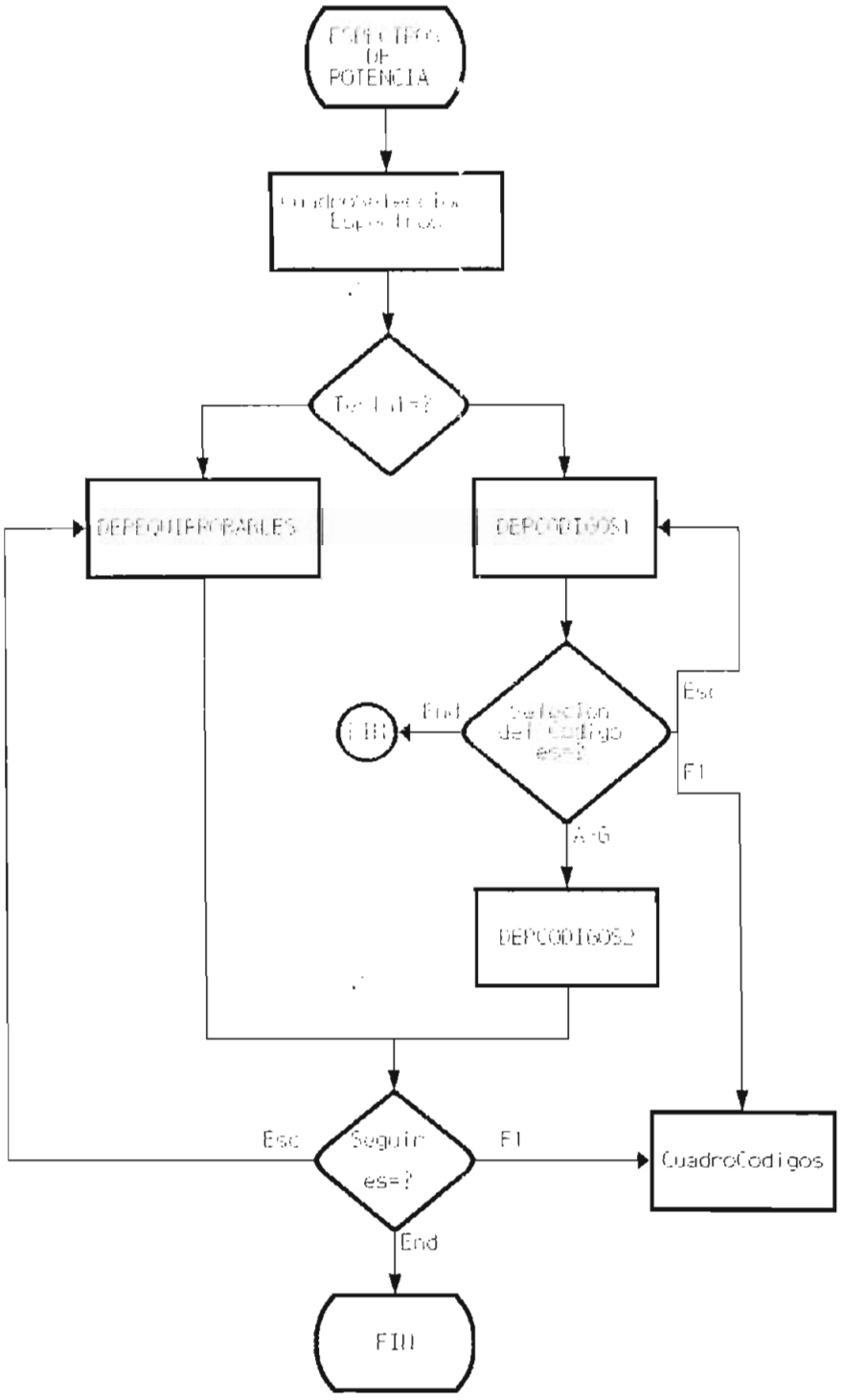


Figura 6.4. Diagrama de flujo del Procedimiento ESPECTROSDEPOTENCIA

El programa permite dos formas de análisis de los espectros de potencia:

a) Comparar las densidades espectrales de potencia de los códigos para la probabilidad p = 0.5, o

b) Comparar las densidades espectrales de potencia de un código para dos valores de probabilidad.

La primera opción se lo hace empleando el procedimiento **DEPEQUIPROBABLES**, en tanto que la segunda utiliza el procedimiento **DEPCODIGOS1**.

En el Capítulo 5 se hace el desarrollo matemático para cada una de las densidades espectrales de potencia, y se obtienen las ecuaciones, con las que se realizan los procedimientos correspondientes. También se explica como se realizan los gráficos, las variables que se utilizan y el diagrama de flujo del procedimiento desarrollado.

Básicamente, los procedimientos o subrutinas de los espectros de potencia toman las ecuaciones y las evalúan, con lo que se tiene las coordenadas para graficar las funciones correspondientes. No todos los espectros de los códigos están disponibles en el programa. En la tabla 6.2 se encuentran los códigos, los procedimientos y ecuaciones utilizadas, que están disponibles para el caso de equiprobabilidad.

CODIGO	PROCEDIMIENTO	ECUACION
NRZ	PNRZ	5.15c
RZ	PRZ	5.14c
NRZ POLAR	PNRZPOLAR	5.18
RZ POLAR	PRZPOLAR	5.17
AMI NRZ	PAMINRZ	5.21
AMI RZ	PAMIRZ	5.20
MANCHESTER	PMANCH	5.24
HDB3	PHDB3	5.25
MILLER	PMILLER	5.26

Tabla 6.2. Procedimientos y ecuaciones para DEP en condiciones de equiprobabilidad

Los subprogramas que utiliza el procedimiento **DEPEQUIPROBABLES** son:

CuadroEspectros1 : Utilizando el modo gráfico del Turbo Pascal, se dibuja un borde doble, los ejes coordenados y las escalas para los gráficos de las ecuaciones. En la parte inferior de la pantalla se escribe el procedimiento que se debe seguir para la comparación de los espectros.

CompararEspectros : Es el procedimiento que permite la compararación los espectros siguiendo el procedimiento indicado en **Cuadro-Espectros1**, el mismo que consiste en pulsar una tecla numérica del 1 al 9. Al hacer esto, este procedimiento llama al correspondiente al código cuyo espectro se desea graficar. De la misma forma se hace para comparar con los otros códigos disponibles.

En las figuras 6.5 y 6.6 se encuentran los diagramas de flujo del procedimiento **DEPEQUIPROBABLES**, y de su subprograma **CompararEspectros** respectivamente.

La comparación de los espectros de potencia de un código, para dos valores de probabilidad, se tiene para los códigos que se listan en la tabla 6.3. En esta tabla también se indican los procedimientos o subprogramas que se utilizan y sus ecuaciones respectivas.

CODIGO	PROCEDIMIENTO	ECUACION
NRZ	PNRZ, NRZPOTENCIA	5.15c
RZ	PRZ, RZPOTENCIA	5.14c
NRZ POLAR	PNRZPOLAR, NRZPOLARPOTENCIA	5.18
RZ POLAR	PRZPOLAR, RZPOLARPOTENCIA	5.17
AMI NRZ	PAMINRZ, AMINRZPOTENCIA	5.21
AMI RZ	PAMIRZ, AMIRZPOTENCIA	5.20
MANCHESTER	PMANCH, MANCHESTERPOTENCIA	5.21

Tabla 6.3 Procedimientos y ecuaciones para comparación de de espectros para dos valores de probabilidad

Los espectros de potencia para los códigos HDB_3 y MILLER, sólo se tiene para el caso de equiprobabilidad (P=0,5).

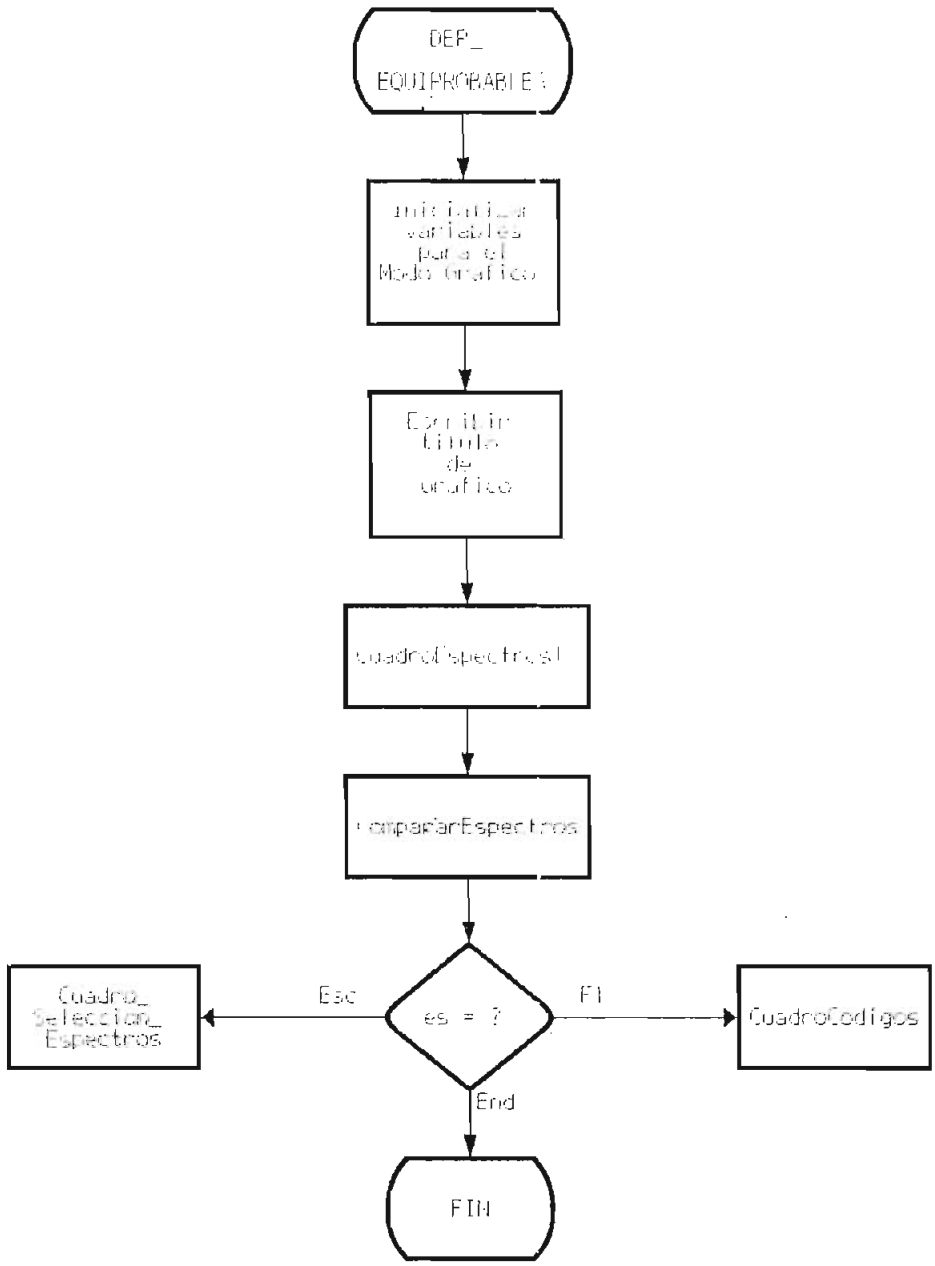


Figura 6.5. Diagrama de flujo del procedimiento DEPEQUIPROBABLES

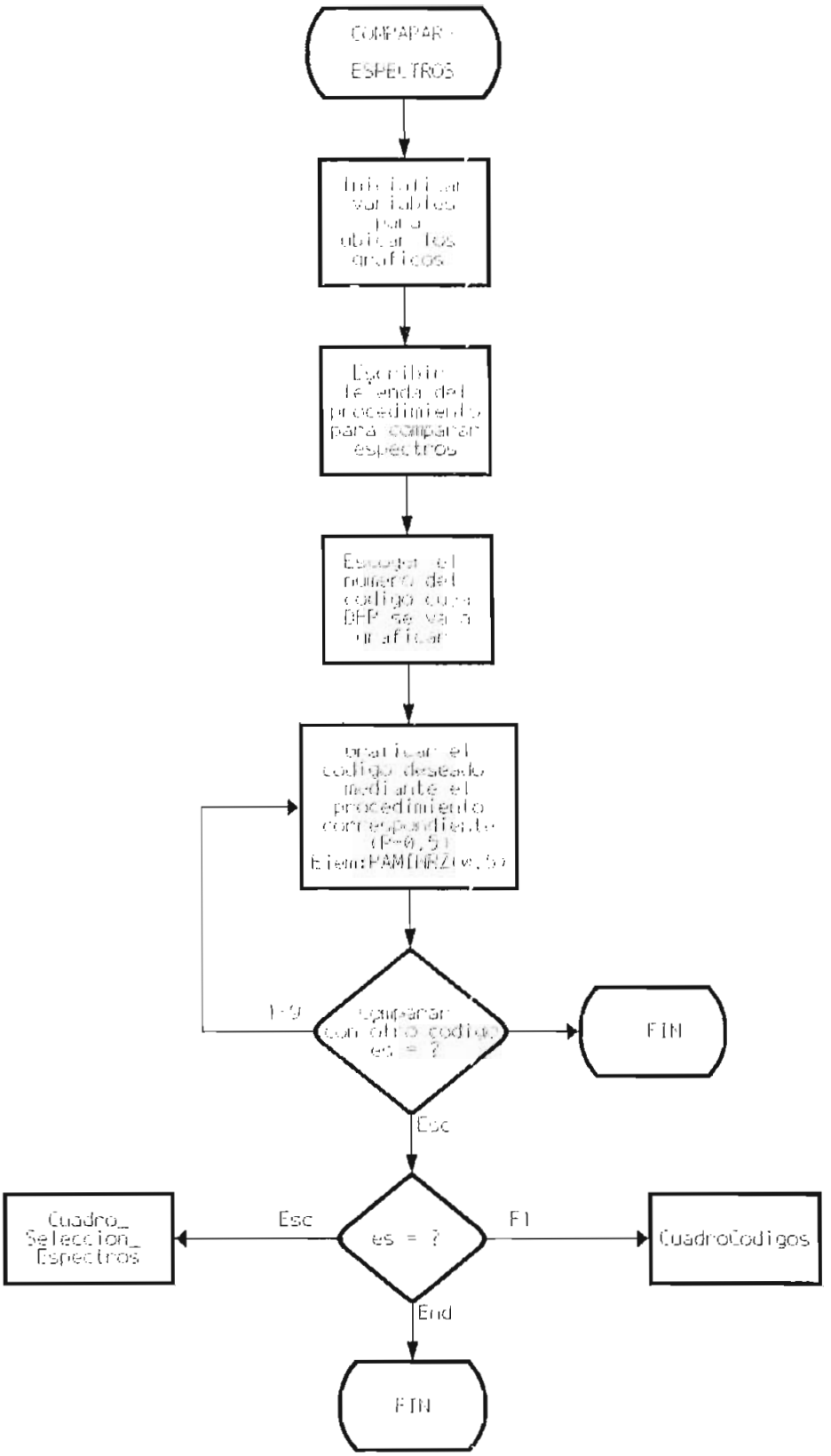


Figura 6.6. Diagrama de flujo del procedimiento COMPARARESPECTROS

Para la comparación de los espectros se utilizan los siguientes subprogramas dentro del procedimiento **DEPCODIGOS1**:

DEPCODIGOS2 : Mediante este procedimiento se escoge el código y los valores de las probabilidades P1 y P2 con los que calcularán las ecuaciones de los espectros respectivos, mediante las llamadas a los procedimientos ***POTENCIA** (Ejm: AMINRZPOTENCIA).

Los procedimientos ***POTENCIA**, preparan la presentación de los espectros en el modo gráfico. Escriben el título, dibujan los bordes, los ejes coordenados y las escalas adecuadas para graficar la función espectral de potencia, llamando a los procedimientos **P*** (Ejm. PAMINRZ), encargados de calcular los valores y graficar la función.

El diagrama de flujo del subprograma **DEPCODIGOS2** se muestra en la figura 6.7, en tanto que la figura 6.8 está a manera de ejemplo el diagrama de flujo del subprograma **AMINRZPOTENCIA**, para los otros códigos se tendrá un diagrama similar.

6.2.2 Programa para Modulación Digital

Antes de la selección de una aplicación de Modulación Digital se tienen dos procedimientos para elegirla. El procedimiento **CuadroModulación** es una pantalla de selección en la que se tiene la lista de las clases de Modulaciones disponibles, a cada una de éstas se le asigna una letra de la A a la H.

El segundo procedimiento es **CuadroSelección Modulación**, en esta se presentan las tres opciones que se disponen: Modulación, Teoría y Espectros de Potencia. Una vez que se ha seleccionado la clase de modulación y la aplicación correspondiente se procede a ejecutar la aplicación deseada.

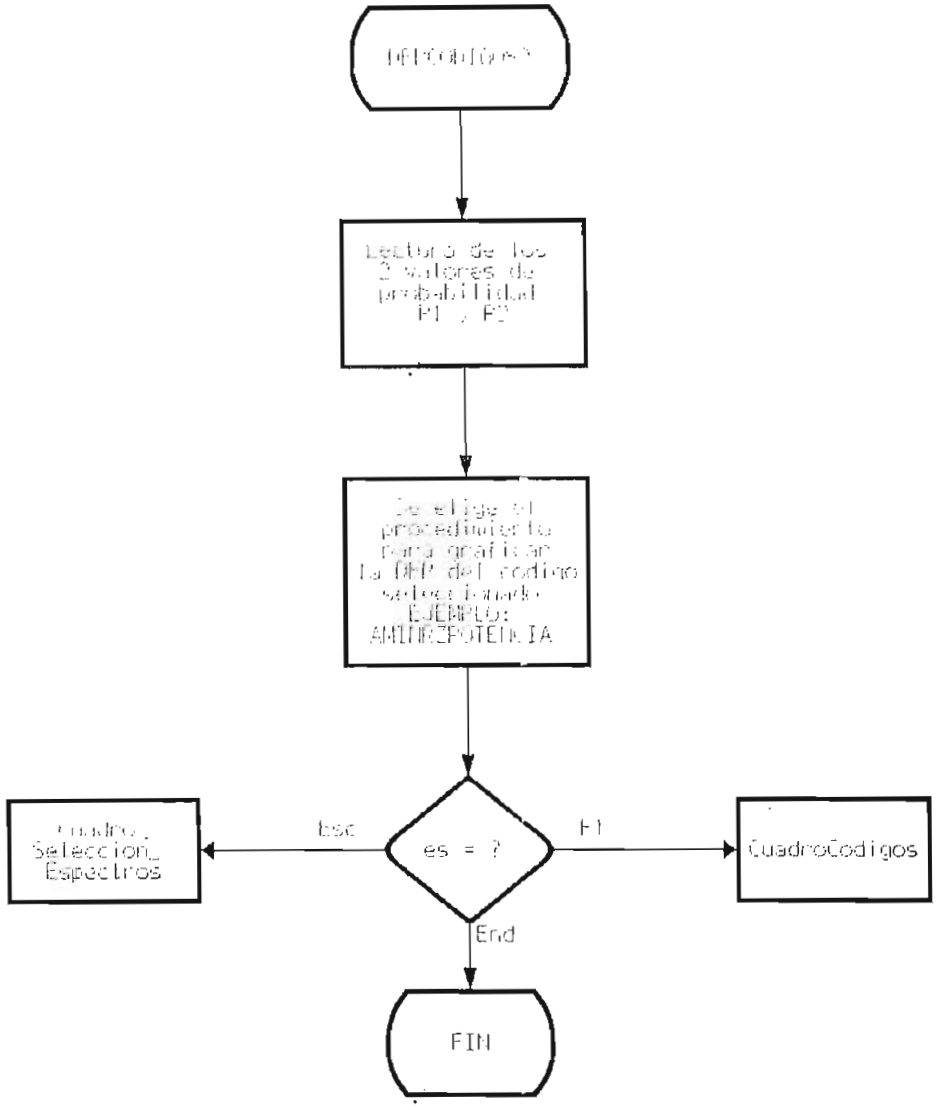


Figura 6.7. Diagrama de flujo del procedimieto DEPCODIGOS2

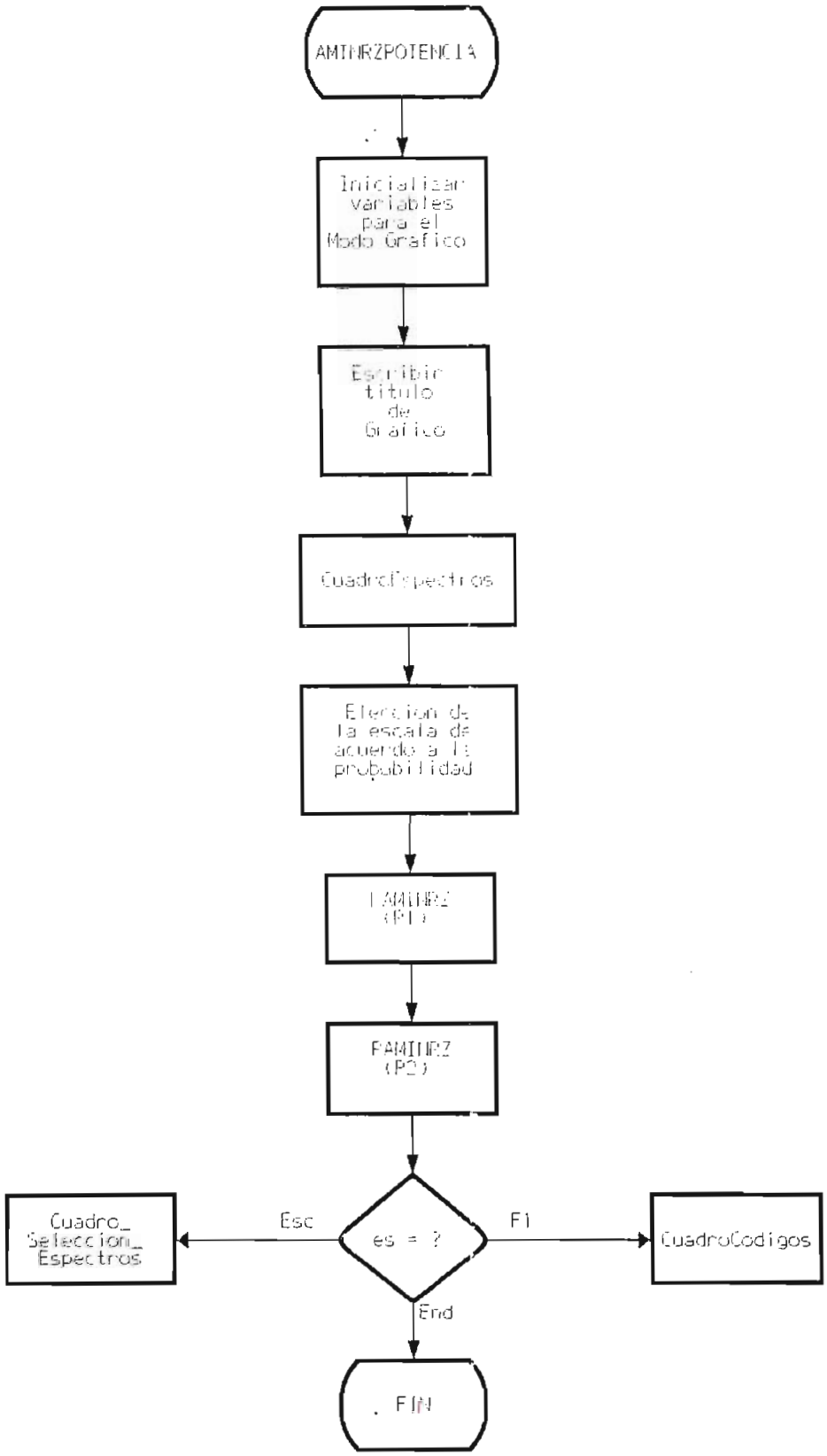


Figura 6.8. Diagrama de flujo del procedimiento AMINRZPOTENCIA

- **Ingreso de datos para Modulación :**

Dos son las maneras de ingresar los datos que se requieren para modular la señal, en forma manual utilizando el procedimiento **IngresoDatos2** y por generación automática de los datos en forma aleatoria con el procedimiento **Aleatorio2**. Todos los datos se almacenan en el vector **bit**.

El número de datos para modulación se restringe a un mínimo de tres (3) y un máximo de once (11); se consideró este número porque permite una buena apreciación del gráfico de la señal y sobre todo observar los cambios de fase en la señal modulada.

Con el procedimiento **MantenerDatos**, se tiene la opción de utilizar los datos anteriores o de cambiarlos si se desea, esto se realiza siempre y cuando se haya ejecutado anteriormente la modulación.

- **Modulación :**

Los algoritmos para las diferentes clases de modulaciones se estudiaron y se desarrollaron en el Capítulo 4, en los numerales 4.4 y 4.5. En base de estos algoritmos se escribieron los procedimientos respectivos.

A continuación se indican las clases de modulaciones disponibles y el nombre del procedimiento correspondiente para realizar las diferentes técnicas de modulación:

MODULACIONES	PROCEDIMIENTOS
ASK	ASK1
FSK	FSK1
PSK	PSK1
4-PSK	M4PSK1
8-PSK	M8PSK1
16-PSK	M16PSK1
4-QAM	M4QAM1
16-QAM	M16QAM1

Tabla 6.4 Procedimientos para las diferentes clases de modulación

Estos procedimientos se ejecutan una vez que se tienen ingresados los datos, luego de lo cual se procede a la graficación de las señales. En la parte superior de la pantalla se dibuja la señal de datos binaria, la señal portadora en la parte media y la señal modulada en la parte inferior.

Para graficar las señales portadora y modulada se lo hace con los procedimientos **GraficarBits** y **GraficarPortadora**. En la figura 6.9 se muestra el diagrama de flujo para la modulación, tomando como base modulación ASK.

- **Teoría de las Modulaciones :**

Para cada una de las clases de modulación se tienen procedimientos que presentan en modo gráfico el texto de la teoría respectiva, ésta resume las principales características y conceptos de las técnicas de modulación. Además se indican las ecuaciones matemáticas que relacionan a cada una de las modulaciones.

En la teoría de las técnicas de modulación multinivel se grafican los diagramas de fase o constelación que indican los posibles estados que pueden adoptar la señales moduladas.

Finalmente para que la teoría expuesta sea más comprensible se puede acceder a un ejemplo de modulación mediante el Procedimiento **Opciones1**. Para realizar este ejemplo se utilizan los procedimientos señalados anteriormente para modulación de la señal, los cuales se llaman con la subrutina **EjemploMod**.

A manera de ejemplo en las figuras 6.10 y 6.11 se muestran los diagramas de flujo del procedimiento **ASKTEORIA** y de su subprograma **EjemploMod**.

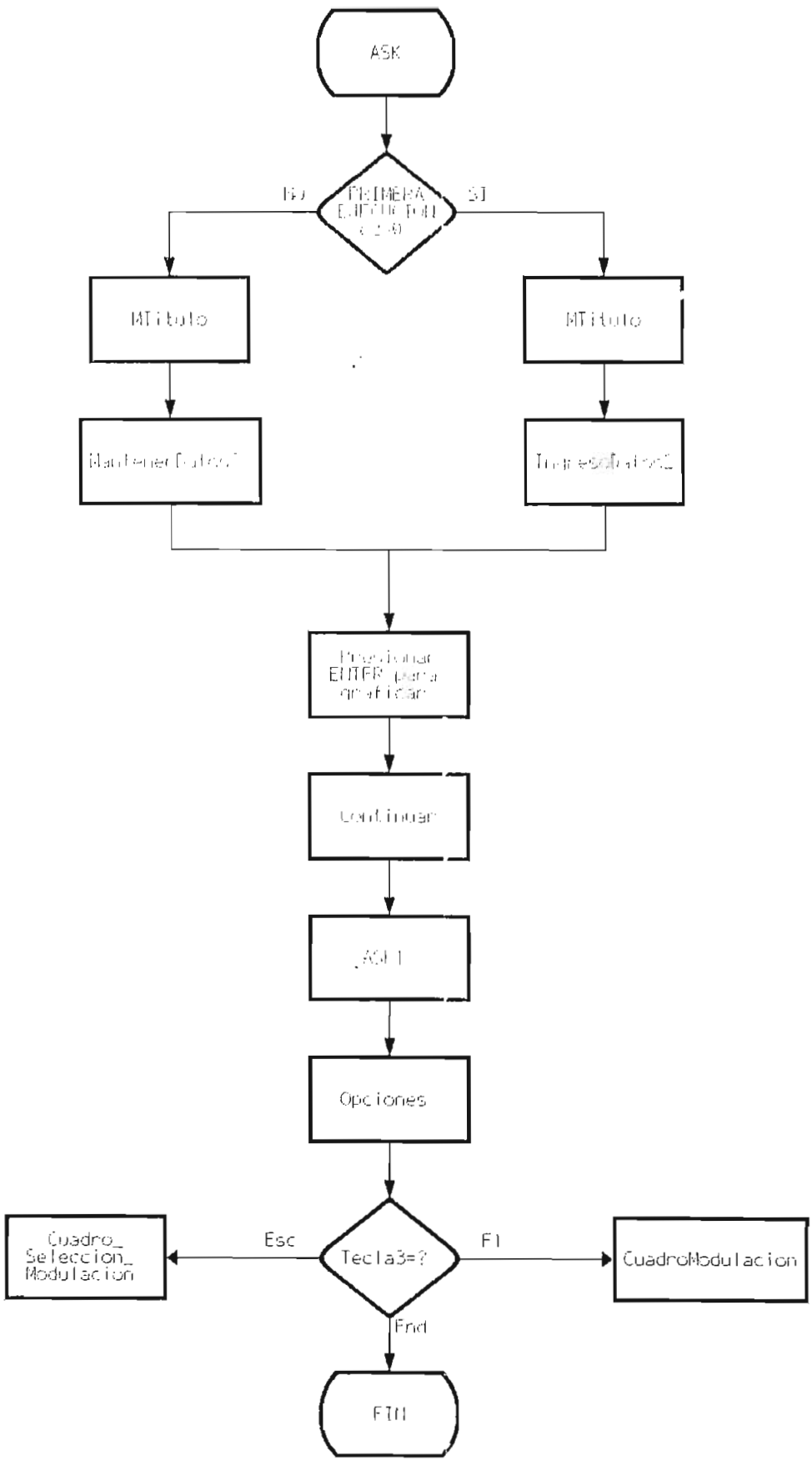


Figura 6.9. Diagrama de Flujo del procedimiento ASK

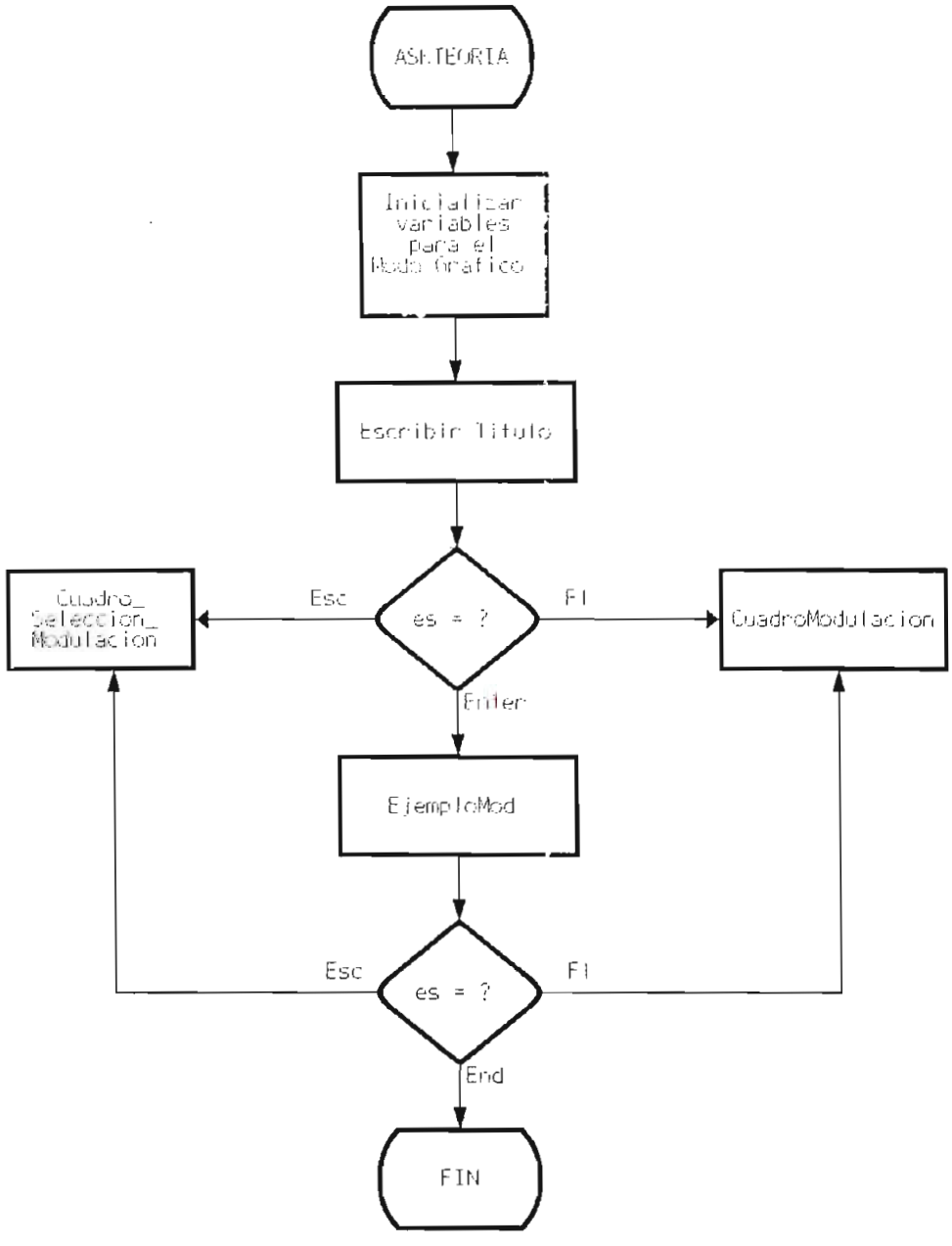


Figura 6.10. Diagrama de Flujo del procedimiento ASKTeoría

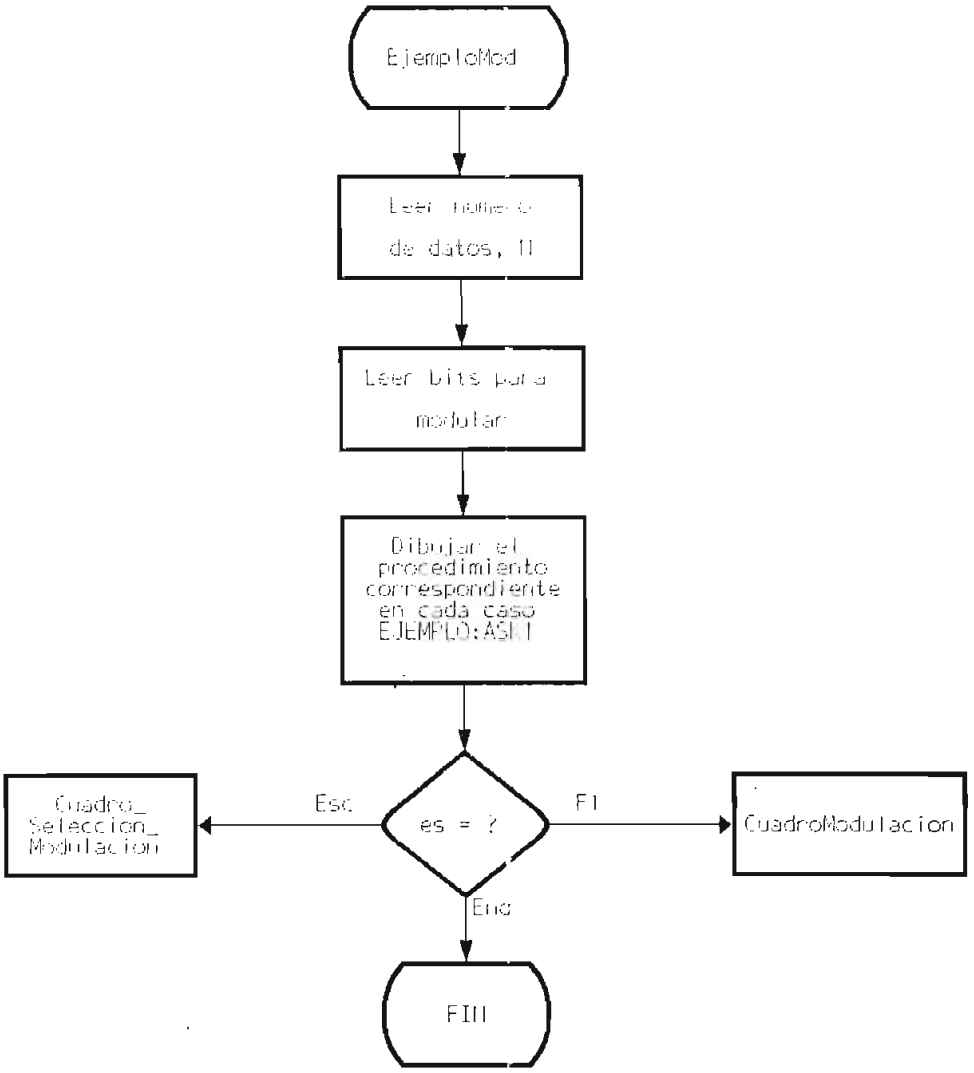


Figura 6.11. Diagrama de Flujo del procedimiento EjemploMod

- Espectros de Potencia para Modulaciones :

A diferencia de lo que realizó para la parte de espectros en los códigos, para modulación digital los espectros de potencia se tienen para todas las modulaciones y constituyen una aplicación más en el menú de selección de cada modulación.

Los procedimientos y ecuaciones que permiten realizar los gráficos de los espectros de potencia para las diferentes modulaciones digitales se listan en la tabla 6.4 a continuación:

MODULACION	PROCEDIMIENTO	ECUACION
ASK	ASKPOTENCIA	5.28
FSK	FSKPOTENCIA	5.31
PSK	PSKPOTENCIA	5.33
4-PSK	M4PSKPOTENCIA	5.35
8-PSK	M8PSKPOTENCIA	5.35
16-PSK	M16PSKPOTENCIA	5.35
4-QAM	M4QAMPOTENCIA	5.38
16-QAM	M16QAMPOTENCIA	5.35

Tabla 6.4 Procedimmientos y ecuaciones para la DEP de las diferentes modulaciones

Las ecuaciones indicadas se obtuvieron en el capítulo 5, numeral 5.4 en el cual se analizan matemáticamente cada una de ellas.

Los procedimientos señalados anteriormente, evalúan las ecuaciones y grafican las funciones correspondientes a las densidades espectrales de potencia. En la figura 6.12 se tiene el diagrama de flujo del procedimiento **ASKPOTENCIA**, la que se ha tomado como ejemplo para este caso.

En el caso de las modulaciones multinivel M-PSK, se añade la opción de comparación de sus espectros de potencia para los esquemas de modulación 2-PSK, 4-PSK, 8-PSK, 16-PSK, con el procedimiento **CompararEspectrosMod**.

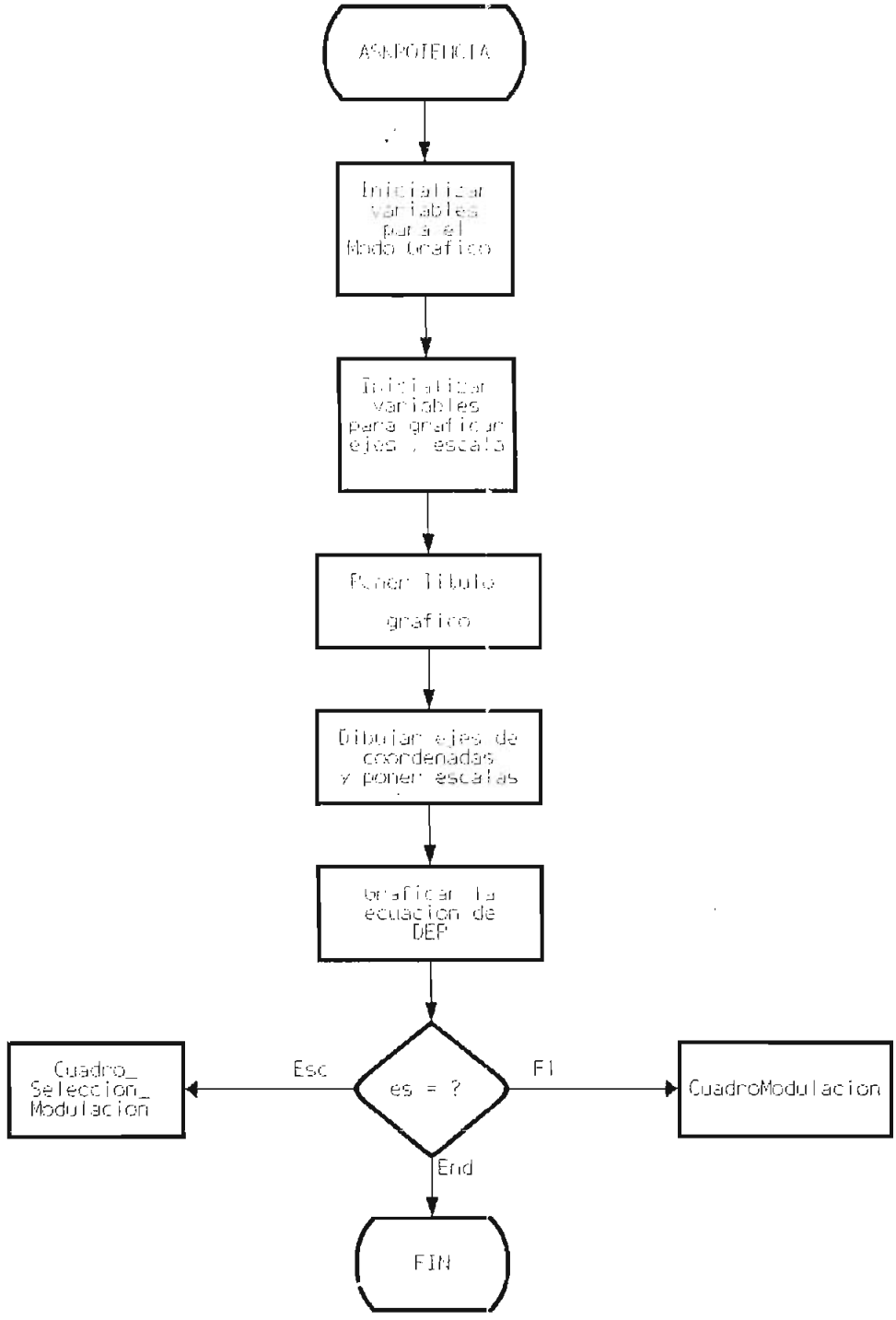


Figura 6.12. Diagrama de Flujo del procedimiento ASKPotencia

6.3. GUIA DE UTILIZACION DEL PROGRAMA

En esta sección se presentan los requerimientos de "hardware" y "software" necesarios para la ejecución del programa, también se indican las instrucciones que se deben seguir paso a paso para la correcta utilización del programa.

- REQUERIMIENTOS DE HARDWARE :

Para utilizar el programa se necesita el siguiente equipo:

- Computador PC, AT, 386, 486 IBM o compatible.
- Memoria RAM de 512 KB como mímimo.
- Monitor monocromático o a color; se recomienda utilizar monitor a color VGA o SVGA porque el programa emplea las ventajas que da la utilización de los colores en los gráficos, de manera especial para la comparación de espectros y señales.

- REQUERIMIENTOS DE SOFTWARE :

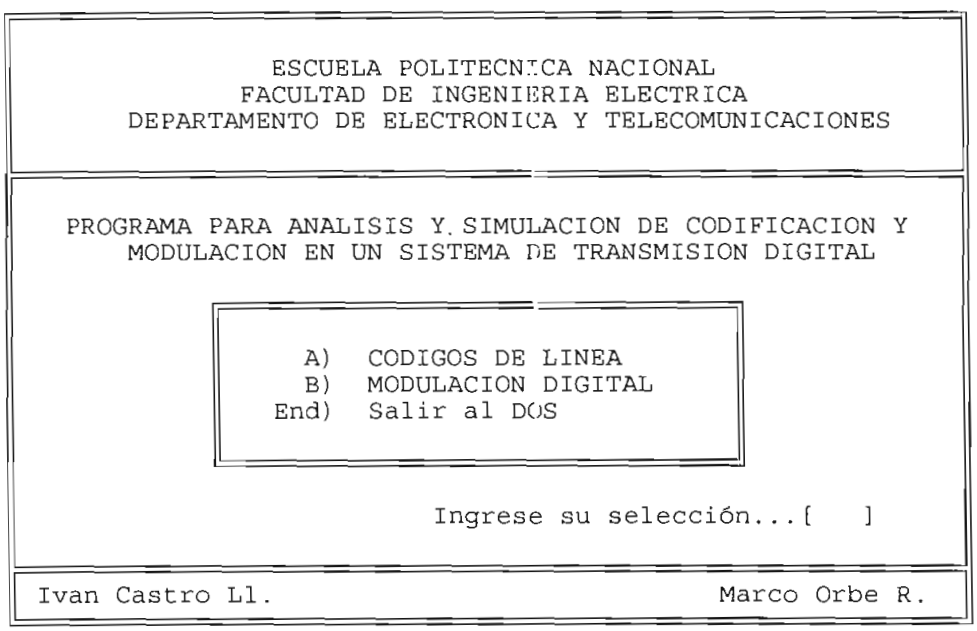
El programa para análisis y simulación de un sistema de comunicación digital denominado TELECOM, está escrito en Turbo Pascal versión 6.0, para su utilización se necesitan de los siguientes requerimientos de "software":

- Sistema Operativo D.O.S. de Microsoft versión 3.3. o superiores
- Programa TELECOM.EXE, así como las unidades o módulos CODIGOS1.TPU, CODIGOS2.TPU, CODIGOS3.TPU que contiene los procedimientos para los códigos de línea y MODULAC1.TPU, MODULAC2.TPU que contienen los procedimientos para modulación digital.
- Directorio BGI, que contiene los archivos de los controladores de gráficos, de extensión .BGI, que son necesarios para inicializar la tarjeta adaptadora del monitor de video, la misma que debe estar instalada en el PC para poder visualizar los gráficos.

Al programa TELECOM se ingresa desde el DOS con el archivo TELECOM.EXE. El programa permite inicialmente el aparecimiento de la pantalla de presentación que contiene el menú principal.

- MENU PRINCIPAL

Es la pantalla de presentación del programa, se tienen tres opciones a elegir, códigos de línea, modulación digital y salida del programa. Estas opciones se pueden acceder pulsando la tecla correspondiente. En caso de no haber pulsado una de las teclas indicadas se emite un sonido y no se realiza acción alguna.

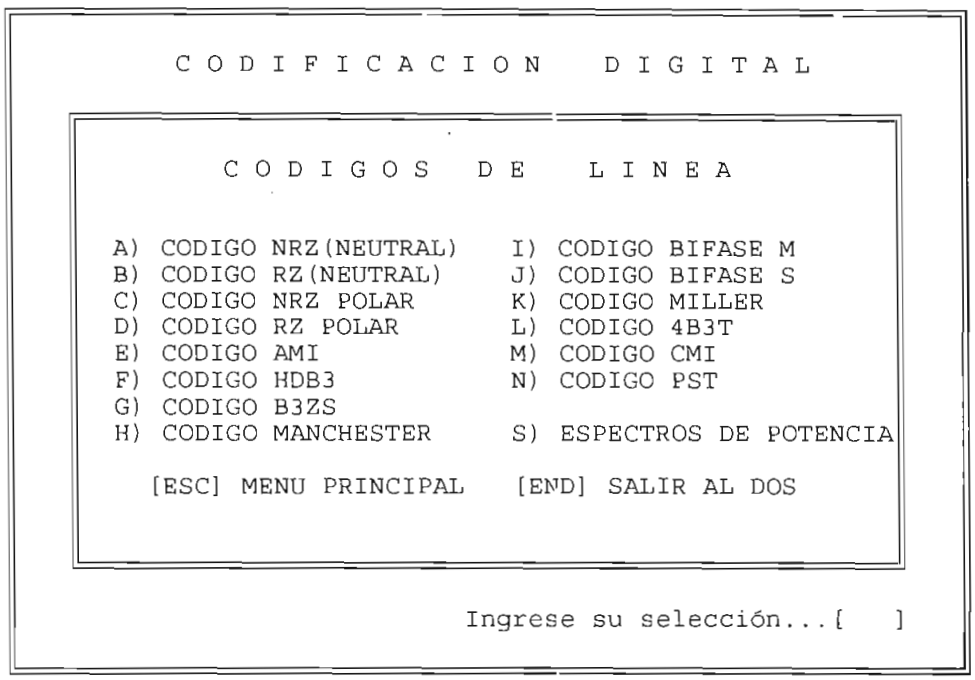


El programa acepta para su selección el ingreso de letras mayúsculas o minúsculas indistintamente. Para las opciónes CODIGOS DE LINEA y MODULACION DIGITAL el programa presenta los menúes correspondientes.

Para salir del programa se debe pulsar la tecla END, con lo que finaliza la ejecución del programa y sale al DOS "limpiando" la pantalla.

- **MENU CODIGOS DE LINEA**

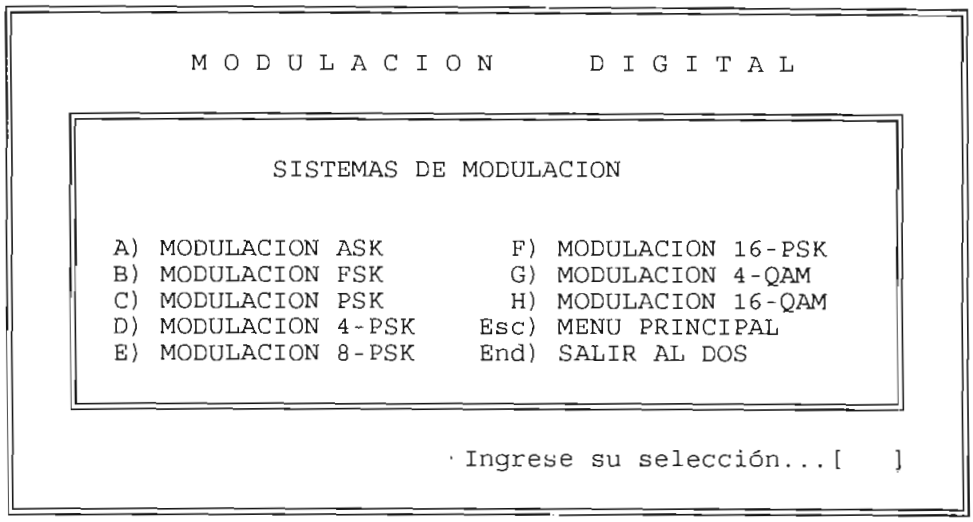
Este menú contiene la lista de los códigos de línea disponibles para la simulación, a los que se puede ingresar pulsando la tecla correspondiente mayúscula o minúscula. La opción ESPECTROS DE POTENCIA permite el análisis de los códigos y la comparación entre ellos.



Pulsando la tecla ESC se puede regresar al Menú Principal o se puede también terminar la ejecución del programa con la tecla END y salir al sistema operativo DOS.

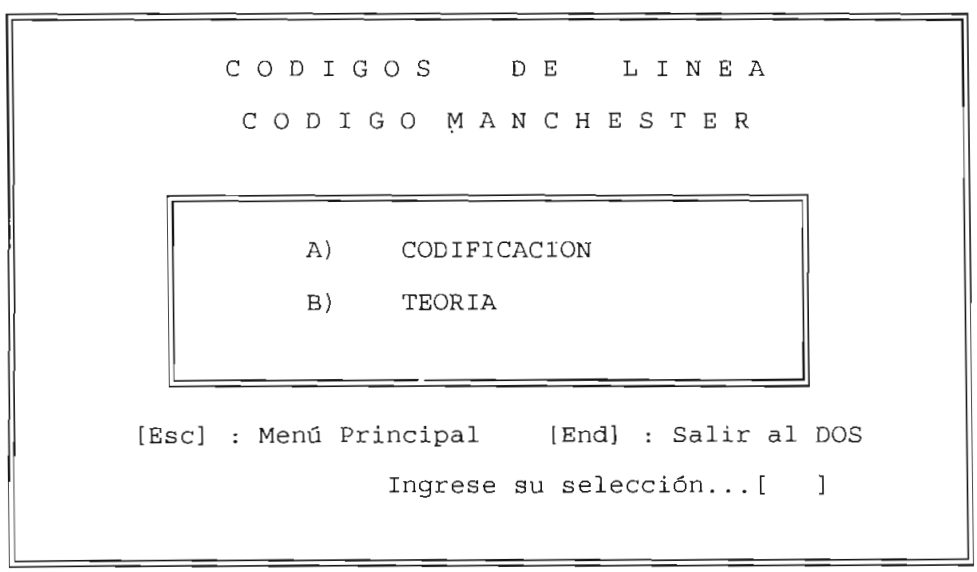
- **MENU MODULACION DIGITAL**

En este menú se presenta la lista de las modulaciones digitales disponibles para análisis y simulación. Los tipos de modulación se eligen en igual forma, pulsando la tecla correspondiente a cada modulación. Al igual que en el caso anterior se tienen las mismas opciones para regresar al Menú Principal o salir al sistema operativo DOS.

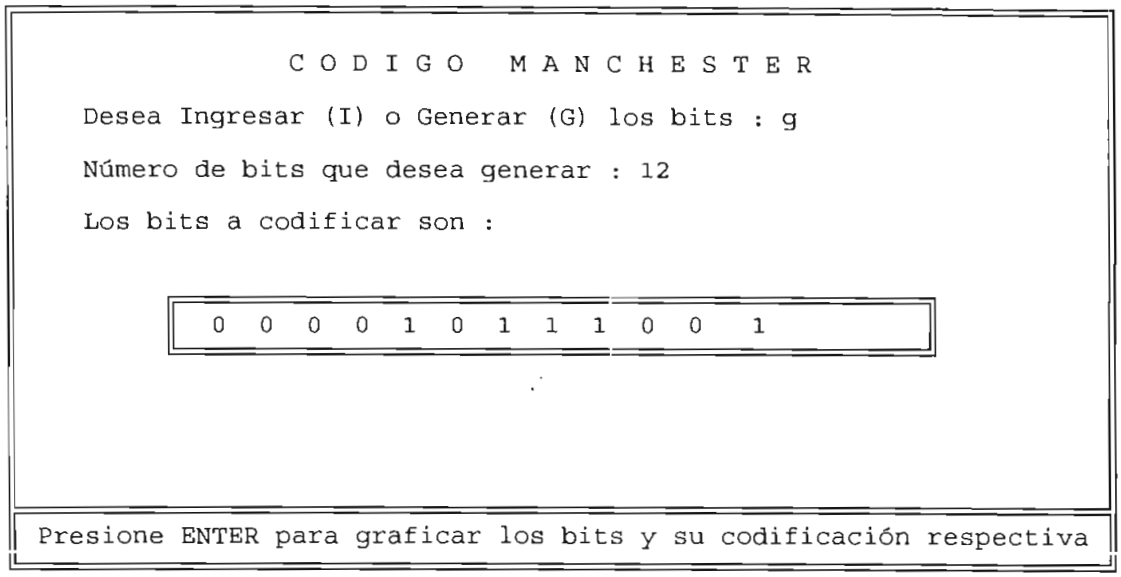


- MENU DE OPCIONES PARA CODIGOS

Una vez que se ingresa a cualquiera de los esquemas de codificación, existen dos opciones para cada uno de los códigos, las mismas que se escojen del menú de opciones: con CODIFICACION se puede codificar una secuencia de bits, en tanto que TEORIA muestra en la pantalla las características y reglas de codificación de cada código complementándolo con un ejemplo.



Si se escoje la primera opción CODIFICACION se tiene una pantalla para el ingreso o generación de una secuencia de los bits, tal como se muestra en el siguiente gráfico:



A la pregunta, desea ingresar o generar los bits? se debe pulsar la tecla i o g según sea el caso; se debe ingresar a continuación el número de bits y pulsar la tecla ENTER para continuar. Si el número de bits es correcto (de 3 a 20) el programa permite el ingreso de los datos o presenta en el cuadro intermedio los bits generados aleatoriamente.

El ingreso de datos se lo hace únicamente con las teclas 0 y 1, no siendo posible corregir la secuencia y solamente se permite ingresar el número de bits deseados.

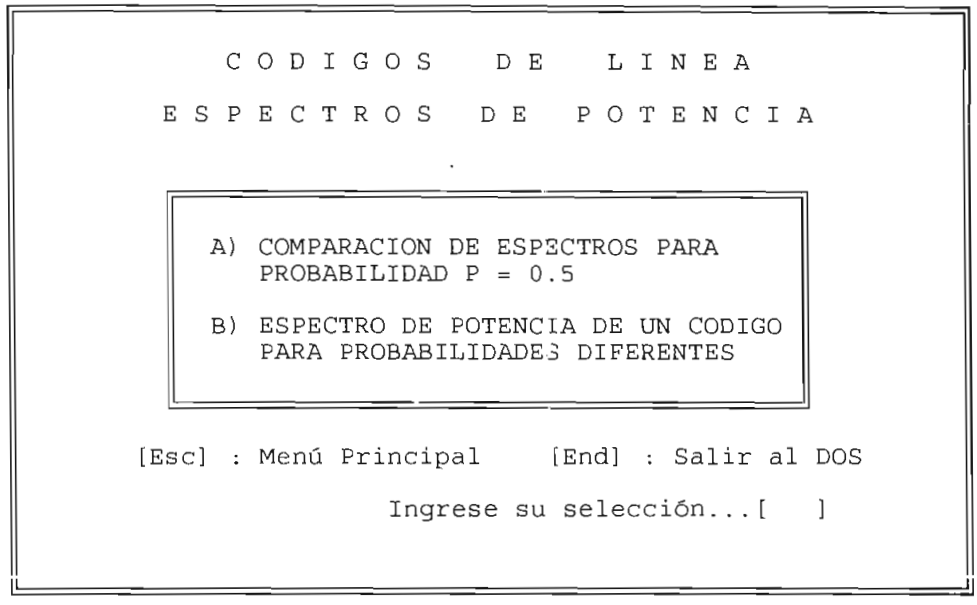
En cada uno de los pasos indicados en el recuadro de la parte inferior aparece una información de ayuda. El paso final es presionar la tecla ENTER para visualizar el gráfico resultante.

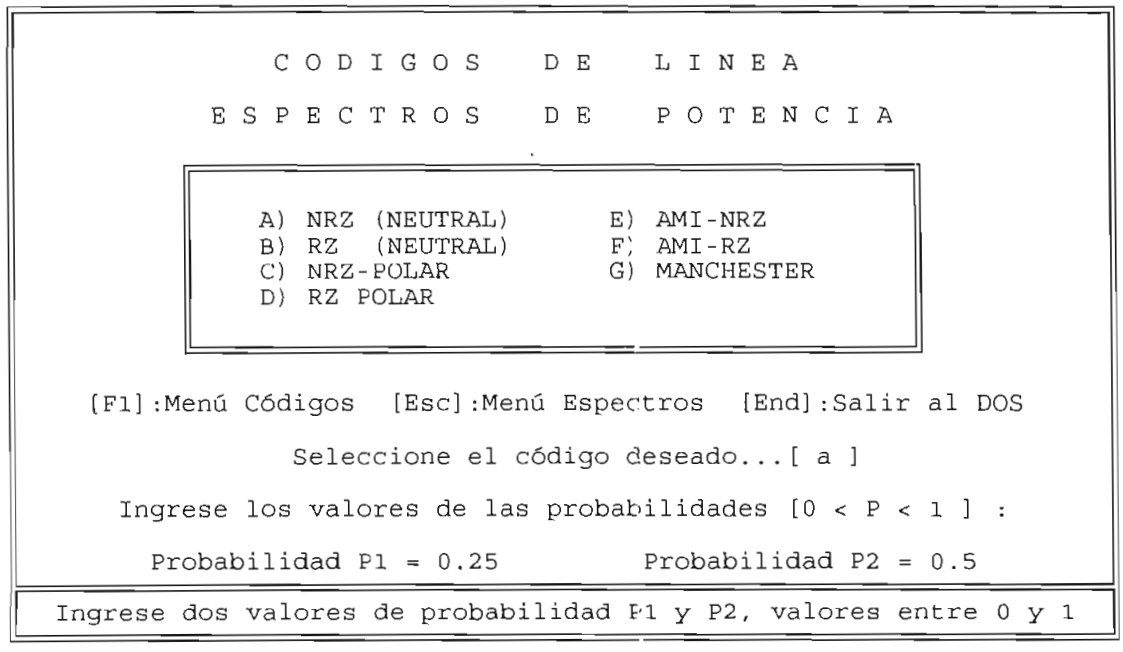
- ESPECTROS DE POTENCIA PARA CODIGOS DE LINEA

Al seleccionar ESPECTROS DE POTENCIA en el Menú se presenta el menú ESPECTROS DE POTENCIA, en el que se tiene dos opciones para graficar las DEP de los códigos. Para la alternativa A, no es necesario ingresar información adicional.

En la alternativa B, se selecciona un código y se comparan las densidades de potencia para dos probabilidades P1 y P2 diferentes, por lo que se deben ingresar la letra del

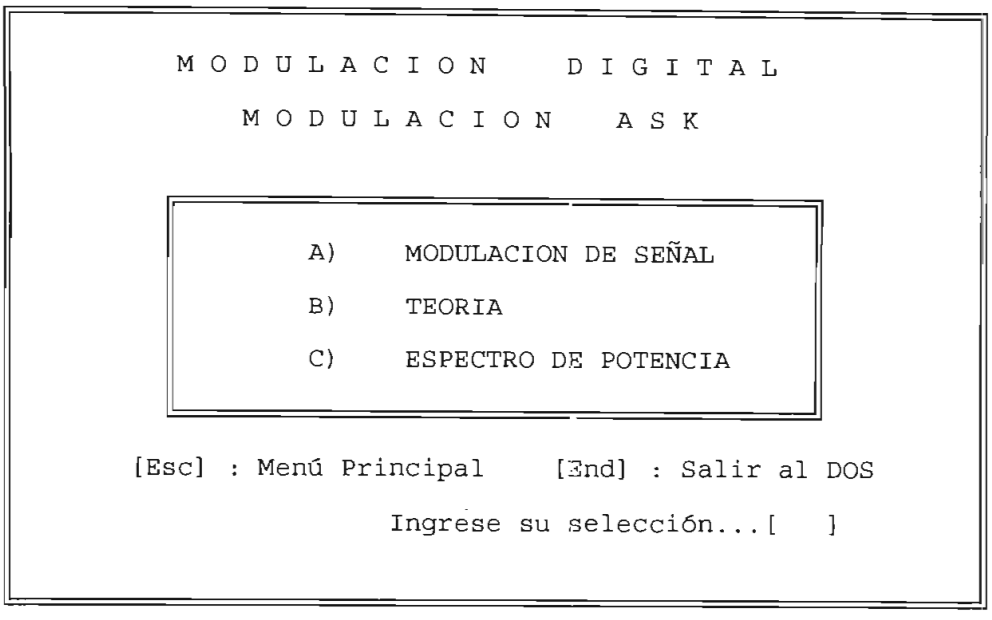
código elegido y los dos valores de las probabilidades (números reales entre 0 y 1). Para graficar se debe pulsar la tecla ENTER.



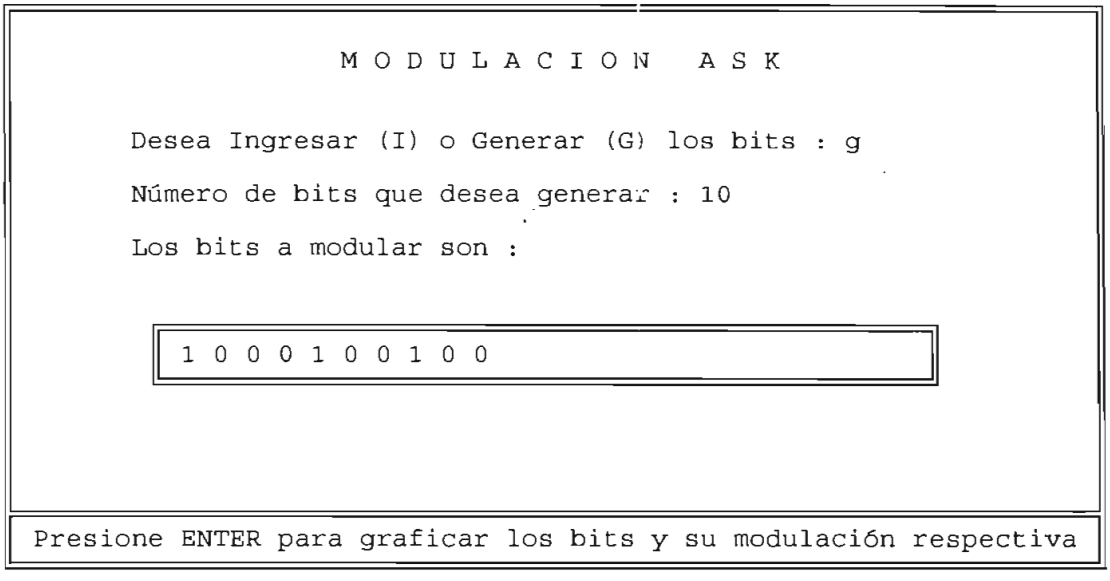


- MENU DE OPCIONES PARA MODULACION

En el caso de Modulación Digital, este menú tiene tres opciones: MODULACION DE SEÑAL, TEORIA, y el ESPECTRO DE POTENCIA correspondiente.



Las opciones B y C no requieren de datos adicionales para ejecutar los gráficos correspondientes. La MODULACION DE SEÑAL, presenta una pantalla que permite el ingreso de los bits de manera similar a la que se hace para codificación, con la diferencia que el número de bits de ingreso permitido está entre 3 y 11.



El proceso a seguir para obtener el gráfico de la señal modulada para la secuencia seleccionada es igual al explicado para codificación.

6.4. EJEMPLOS DE UTILIZACION

A continuación se grafican los resultados obtenidos luego de da haber utilizado el programa, siguiendo las instrucciones dadas en el numeral anterior. Para el código Manchester en la figura 6.13 se observa la codificación de la secuencia de bits, en la figura 6.14 está la teoría explicativa luego de haber escogido la opción Teoría y en la figura 6.15 se observa la Densidad Espectral de Potencia.

Como ejemplo de Modulación se escoge la modulación ASK con la secuencia escogida en los menúes anteriores, en la figura 6.16 se observa la modulación de la señal, en la figura 6.17 se tiene la teoría respectiva, en la figura 6.18 un ejemplo de teoría ; el espectro de la señal modulada está en la figura 6.18.

En la figura 6.20, se presenta la teoría del código 16-QAM en la que se incluye el diagrama de constelación correspondiente.

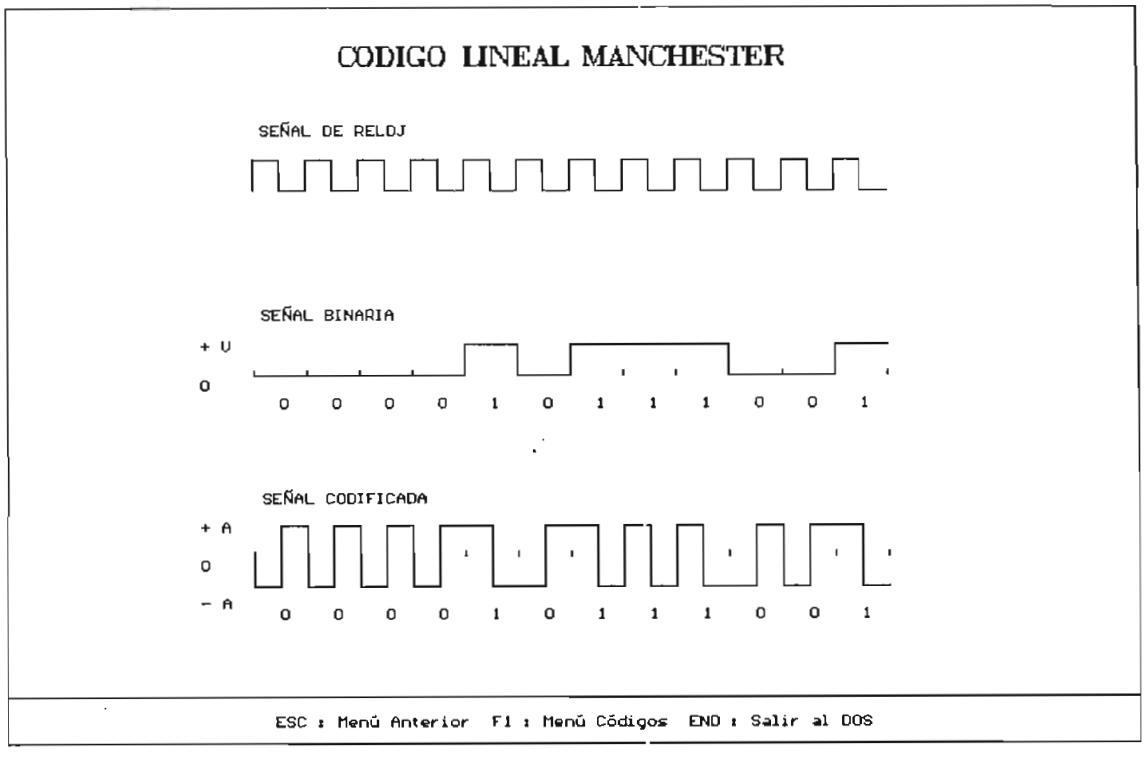
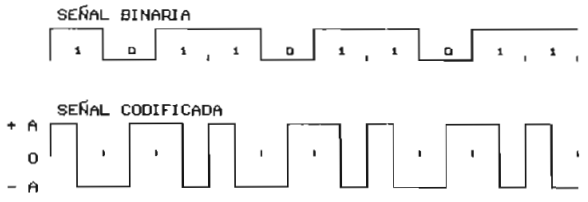


Fig. 6.13. Codificación Manchester

C O D I G O L I N E A L M A N C H E S T E R

En este código se producen transiciones positivas y negativas entre +A y -A en la mitad del intervalo unitario To, el pulso binario 1L tiene una transición prefijada y símbolo OL la opuesta.
Este código tiene una mayor facilidad para recuperar la señal de reloj y y evita la componente DC, pero con el aumento del ancho de banda .
En este caso se ha asignado a 1L la transición negativa.

E J E M P L O :
Codificación de la secuencia de bits : 1 0 1 1 0 1 1 0 1 1



ESC : Menú Anterior F1 : Menú Códigos END : Salir al DOS

Fig. 6.14. Teoría de codificación Manchester

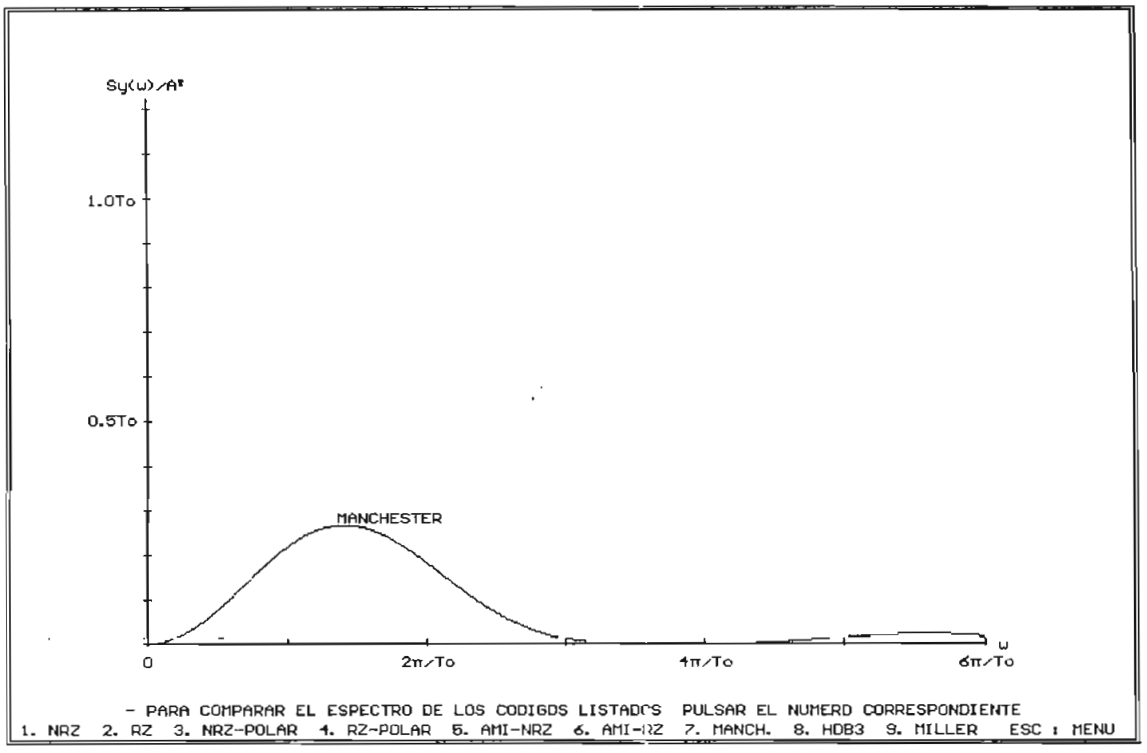


Fig 6.15. Espectro de potencia Manchester

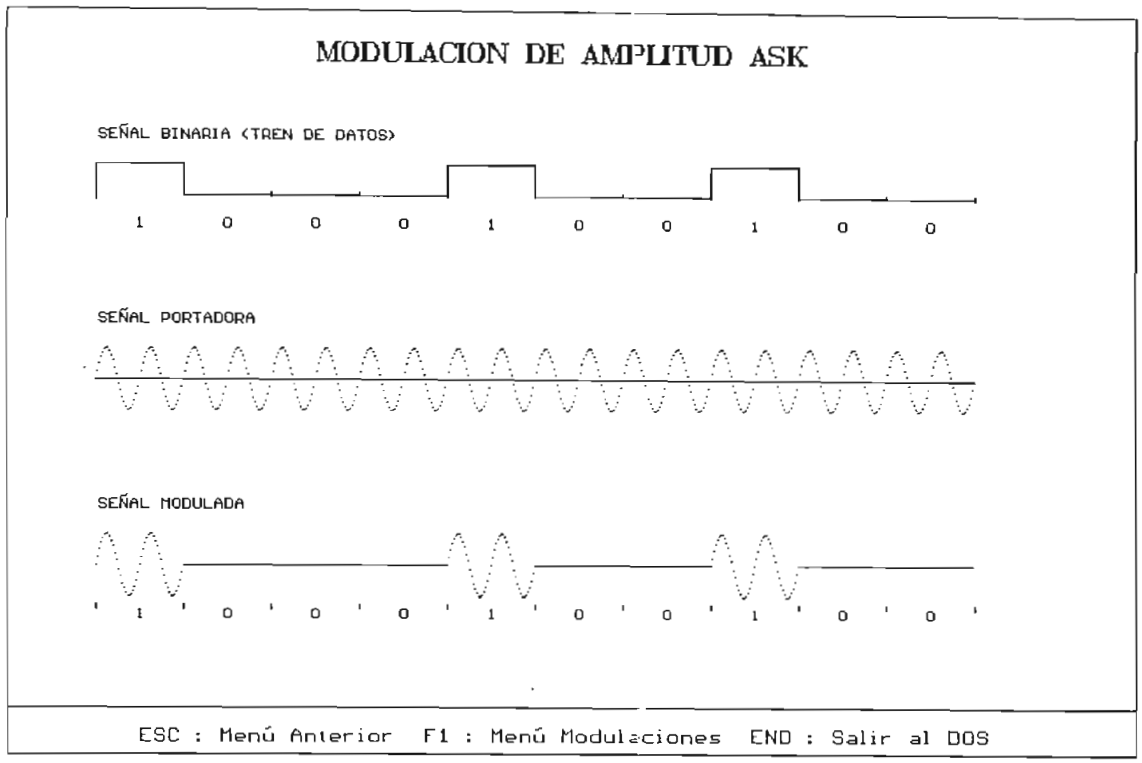


Fig. 6.16. Señal modulada ASK

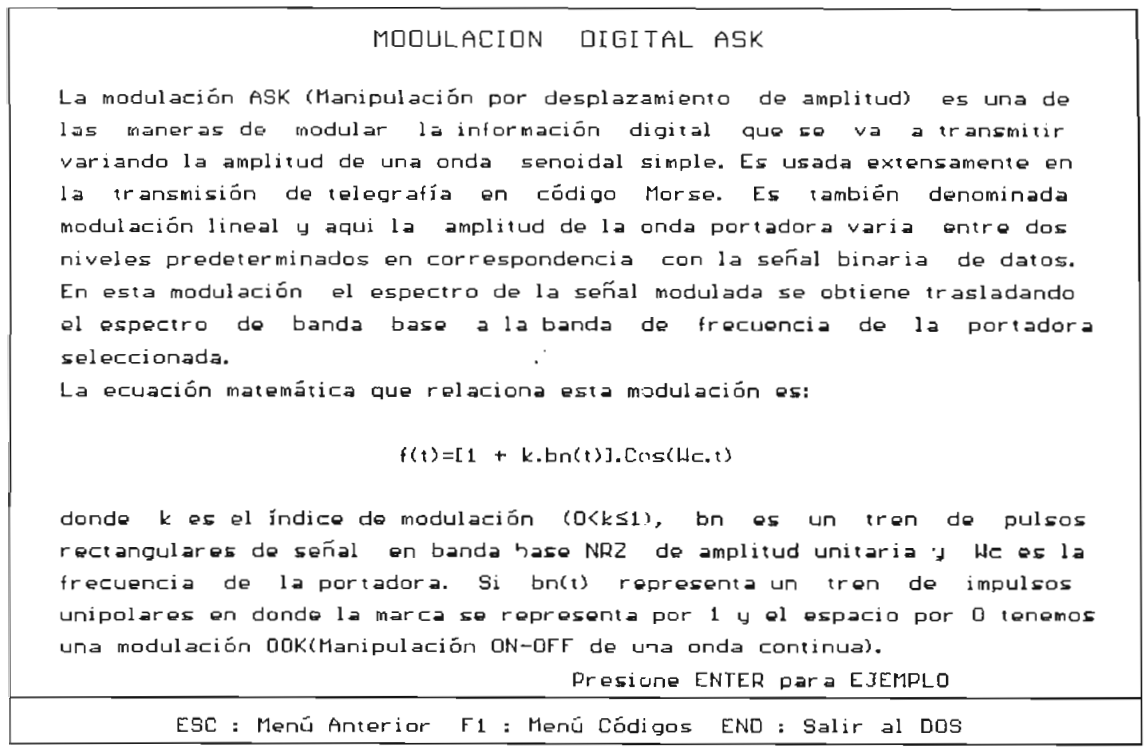


Fig. 6.17. Teoría de modulación ASK

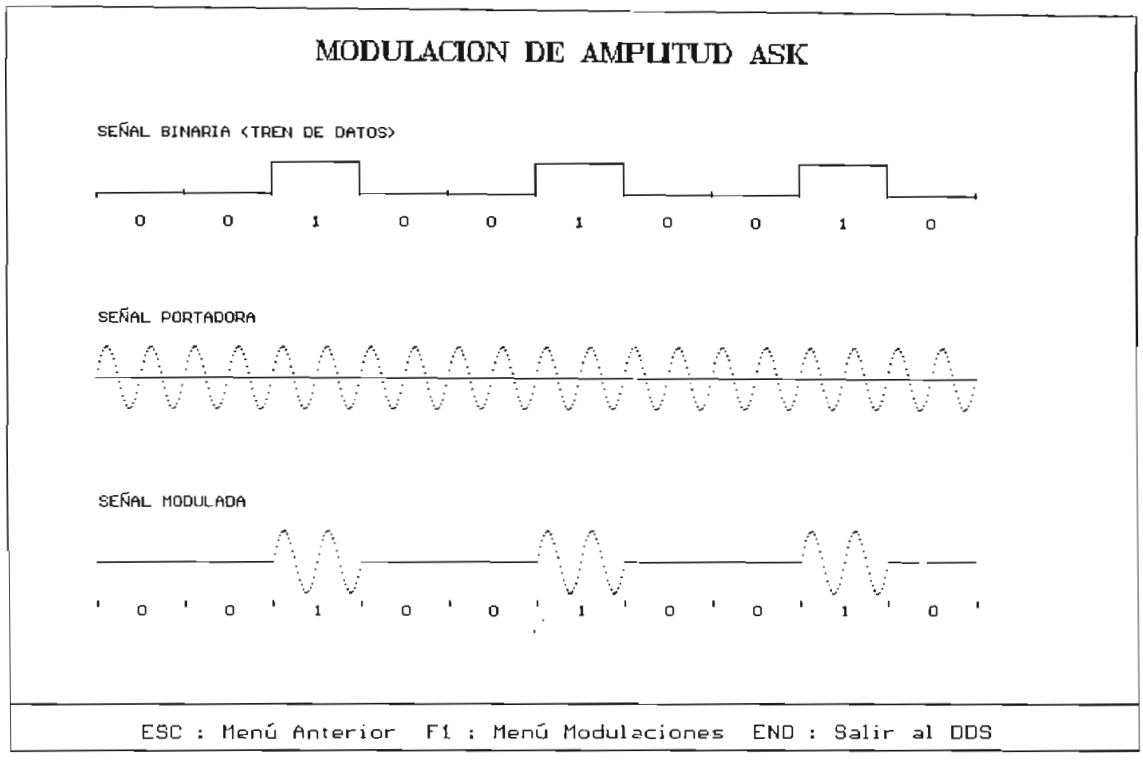


Fig. 6.18. Ejemplo de TEORIA ASK

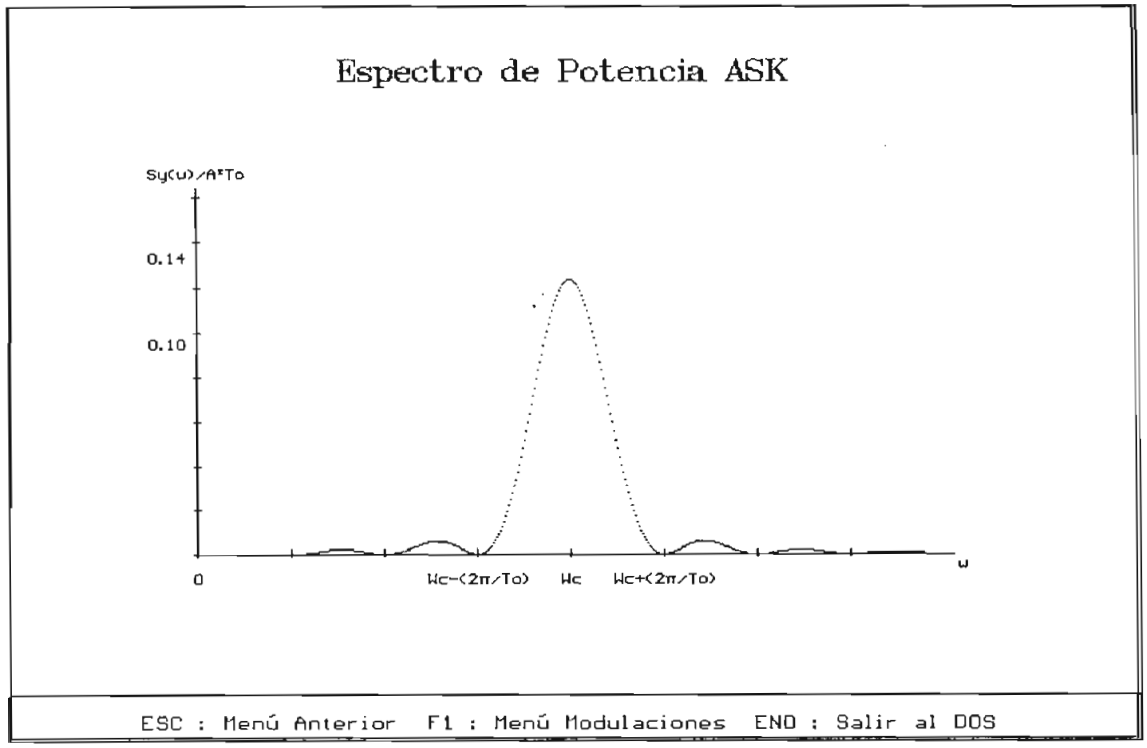


Fig. 6.19. Espectro de potencia ASK

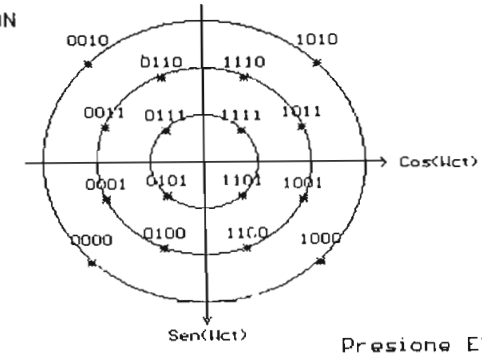
MODULACION DIGITAL 16-QAM

En este tipo de modulación la amplitud de la señal no permanece constante, variando al igual que la fase, este esquema consiste en la modulación multinivel de amplitud de dos portadoras independientes, la ecuación para este tipo de modulación es la siguiente:

$$g(t) = ai*Cos(Wc.t) + bi*Sen(Wc.t)$$

donde ai y bi toman en forma independiente los valores discretos previstos según el número de niveles establecidos siendo M=L².
En el caso de 16-QAM, M=16 y L=4, las variables ai y bi pueden tener los valores (-3, -1, +3, +1). Cada valor de ai o bi son asociados con un dibit.

DIAGRAMA DE CONSTELACION MODULACION 16-QAM:



Presione ENTER para EJEMPLO

ESC : Menú Anterior F1 : Menú Modulaciones END : Salir al DOS

Fig. 6.20. Teoría 16-QAM

CAPITULO 7

CONCLUSIONES

La transmisión de información mediante los sistemas de comunicación ha ido evolucionando desde mucho tiempo atrás, siendo tan importante en la actualidad por su extensa aplicación en diferentes campos de la industria, la banca e incluso para bienestar y defensa de las naciones.

Las características de los elementos principales de un sistema de transmisión como lo son el transmisor, el medio de propagación o canal y el receptor darán la calidad de transmisión de los datos, siendo muy importante considerar los factores que afectan la señal durante todo su trayecto de transmisor-receptor.

La codificación y la modulación son parte muy importante en los procesos de transmisión, hasta los modems más sofisticados de la actualidad los emplean en su estructura.

El análisis de las señales en el dominio del tiempo y la frecuencia cumplen un papel muy importante en los sistemas de comunicación, ya que se constituye en la base matemática que permiten conocer las características de densidad espectral de los diversos códigos y modulaciones y consecuentemente sus requerimientos de ancho de banda, potencia, y el comportamiento de las señales con respecto al ruido.

En cuanto al proceso de codificación en banda base, la codificación NRZ tiene componente DC a bajas frecuencias y por esto no es usada en transmisión a larga distancia, mientras que el esquema RZ tiene la desventaja de requerir de mayor ancho de banda que las otras, necesitando de menor potencia para su transmisión.

Las codificaciones polares tienen la característica de no poseer componente DC a bajas frecuencia, por lo que son ampliamente usadas en transmisión serial.

Las codificaciones AMI disminuyen los niveles de componente continua y son fáciles de implementarse por lo que son ampliamente utilizadas.

Las transiciones de los códigos bifase representan una ventaja, ya que ayudan a la recuperación de la señal de reloj para sincronismo entre el transmisor y el receptor, pero por su complejidad de diseño y alto costo no son frecuentemente utilizados.

Las codificaciones HDB_n, y BnZS requerirán de mayor complejidad para su implementación por lo que su uso estará restringido a sistemas de gran capacidad de transmisión de datos.

En lo que se refiere a la modulación digital, la 2-PSK es mejor que ASK y FSK, ya que requiere de menor relación señal a ruido (S/N) para una probabilidad de error dada. Es así que 2-PSK necesita menos potencia media que ASK o FSK, pero tienen la desventaja de necesitar detección coherente en la recepción, razón por la cual se utiliza la modulación diferencial DPSK que evita la sincronización en la detección pero que es mas difícil de implementar que ASK o FSK.

Si se usa modulación multisimbólica para transmitir mayores razones de datos dentro de un ancho de banda específico, se tendrá, que a medida que aumente el número de estados M de la señal y aumente la cantidad de información dada por cada símbolo con respecto a la modulación binaria, se deberá aumentar la potencia de la señal. El diseño de moduladores y demoduladores multisimbólicos será más complejo que los binarios, lo que incidirá en el costo del equipo.

La modulación M-QAM presenta mayores ventajas que M-PSK, ya que esta última necesitará de mayor potencia para

conservar la misma probabilidad de error, es por esto que M-QAM es usado en sistemas de radio de alta capacidad.

Se debe mencionar que, en el análisis matemático de las codificaciones y modulaciones se supone al ruido como el único factor que produce errores en la recepción, anotando que la interferencia intersímbolos ISI también incidirá en la transmisión.

El carácter didáctico del programa que se ha diseñado nos permite observar las características principales de la codificación y la modulación como son; ancho de banda, potencia requerida y la facilidad de recuperación de reloj, lo que será de gran ayuda al momento de escoger el tipo de código o modulación para la transmisión de información.

De lo expuesto, se concluye que el programa realizado en esta tesis cumple con el objetivo propuesto, ya que el uso del mismo complementará los conocimientos teóricos dados en la facultad en el área de Comunicación Digital.

Se recomienda realizar trabajos similares de simulación de sistemas en los diversos campos de las comunicaciones mediante el uso de un computador y el aprovechamiento de paquetes computacionales como por ejemplo Lenguaje C, ya que procedimientos analíticos basados en matemáticas muy avanzadas pueden ser tratados más fácilmente con la ayudas de estas herramientas.

ANEXO A

RECOMENDACION G.703 DEL CCITT

ANEXO A

(a la Recomendación G.702)

Velocidades binarias utilizables disponibles para los servicios

En el caso del acceso a la RDSI para los servicios de banda ancha, en las Recomendaciones de la serie I.200 se especifican las velocidades binarias hasta el primer nivel de jerarquía.

En general, con referencia a las velocidades binarias disponibles para el transporte de las señales de servicio, se aplicarán las siguientes directrices:

A.1 En el caso de las redes que utilizan las jerarquías basadas en la velocidad primaria de 1544 kbit/s, se ha establecido el principio por el que algunos bits de la trama deben reservarse, en particular para el control de extremo a extremo de la calidad de los trayectos digitales cuando hay varias secciones digitales en tándem. Un ejemplo de la aplicación de este principio lo ofrece la velocidad de 1544 kbit/s, en la que se reservan algunos bits a tal fin (véase la Recomendación G.704). Dicho principio no implica por necesidad que existe ninguna restricción básica con respecto a la provisión de la jerarquía completa de velocidades binarias. Por ejemplo, a 6312 kbit/s no existe ninguna restricción fundamental respecto de la utilización de la capacidad total del trayecto digital. No obstante, quizá sea preciso tomar en cuenta los principios mencionados.

A.2 En el caso de redes que utilizan la jerarquía basada en 2048 kbit/s, no hay ninguna restricción básica a la utilización de la capacidad total del trayecto digital. Sin embargo, se reconoció que la compatibilidad con las estructuras de trama recomendadas para los diversos niveles de la jerarquía de 2 Mbit/s (por ejemplo, utilización de los mismos esquemas de alineación de trama) podría ser una solución preferida, puesto que ofrece las siguentes ventajas:

- utilización de los mismos dispositivos de codificación para las aplicaciones conmutadas y no conmutadas;
- control de la calidad de extremo a extremo realizada por la red cuando la entidad de mantenimiento que termina el servicio (por ejemplo, el dispositivo de codificación) no pertenece a la red;
- posibilidad de realizar otras funciones necesarias de gestión de red, según las aplicaciones.

Podría reconsiderarse la preferencia por la compatibilidad de las estructuras de trama recomendadas para las aplicaciones en las que puedan identificarse importantes restricciones sobre la utilización eficaz de la capacidad del trayecto digital.

Recomendación G.703

CARACTERÍSTICAS FÍSICAS Y ELÉCTRICAS DE LOS INTERFACES DIGITALES JERÁRQUICOS

(Ginebra, 1972, modificada posteriormente)

El CCITT

considerando

que se necesitan especificaciones sobre interfaces para poder interconectar los componentes de las redes digitales (secciones digitales, equipo múltiplex, centrales) a fin de formar un enlace digital internacional o una conexión digital internacional;

que la Recomendación G.702 define los niveles jerárquicos;

que la Recomendación G.704 trata de las características funcionales de los interfaces asociados con los modos de la red;

que la serie I.430 de Recomendaciones trata de las características de la capa 1 para los interfaces usuario-red de la RDSI;

recomienda

que las características físicas y eléctricas de los interfaces, a las diferentes velocidades binarias jerárquicas, estén conformes a la descripción dada en la presente Recomendación.

Observación 1 – Las características de los interfaces a las velocidades binarias no jerárquicas se especifican en las Recomendaciones pertinentes sobre el equipo.

Observación 2 – Las especificaciones de los valores de fluctuación de fase contenidas en los § 6, 7, 8 y 9 están destinadas a su aplicación en los puntos de interconexión internacional.

Observación 3 – Los interfaces descritos en los § 2 a 9 de la presente Recomendación corresponden a los accesos T (acceso de salida) y T' (acceso de entrada) conforme se recomienda para la interconexión en la Recomendación AC/9 del CCIR con referencia al Informe AH/9 de la Comisión de Estudio 9 del CCIR (en dicho Informe se definen los puntos T y T').

1 Interfaz a 64 kbit/s

1.1 *Requisitos funcionales*

1.1.1 Para el diseño del interfaz se han recomendado los requisitos fundamentales siguientes:

1.1.2 Tres señales atraviesan el interfaz en los dos sentidos, transmisión y recepción, a saber:

- la señal de información a 64 kbit/s;
- la señal de temporización de 64 kHz;
- la señal de temporización de 8 kHz.

Observación 1 – Se debe generar una señal de temporización de 8 kHz, pero no será obligatorio para el equipo en el lado de servicios del interfaz (por ejemplo, señales de datos o señalización) utilizar la señal de temporización de 8 kHz procedente del múltiplex MIC o del equipo de acceso a un intervalo de tiempo, ni proporcionar una señal de temporización de 8 kHz al equipo MIC.

Observación 2 – La detección de una avería en un punto situado hacia el origen puede transmitirse a través de un interfaz a 64 kbit/s enviando una señal de indicación de alarma (AIS), interrumpiendo la señal de temporización de 8 kHz en el sentido de recepción, o de ambas formas.

1.1.3 El interfaz debe ser independiente de la secuencia de bits a 64 kbit/s.

Observación 1 – Pueden transmitirse a través del interfaz señales a 64 kbit/s sin ninguna restricción. Sin embargo, esto no implica que puedan realizarse, sobre una base global, trayectos a 64 kbit/s no sujetos a restricción alguna. Esto se debe a que algunas Administraciones se proponen instalar o están instalando vastas redes compuestas de secciones de línea digital cuyas características no permiten la transmisión de largas secuencias de 0. (La Recomendación G.733 prevé equipos múltiplex MIC con características apropiadas para estas secciones de línea digital.) En lo que respecta específicamente a fuentes de trenes binarios con temporización de octetos, en redes digitales a 1544 kbit/s se exige que haya, por lo menos, un 1 binario en cada uno de los octetos de una señal digital a 64 kbit/s. En los trenes binarios no sujetos a temporización de octetos, la señal a 64 kbit/s no podrá tener más de 7 ceros consecutivos.

Observación 2 – Aunque el interfaz es independiente de la secuencia de bits, la utilización de la señal AIS (secuencia todos 1) puede dar lugar a la imposición de ciertas limitaciones de menor importancia a la fuente de 64 kbit/s. Por ejemplo, una señal de alineación de trama todos 1 podría ocasionar problemas.

1.1.4 *Se han previsto tres tipos de interfaces*

1.1.4.1 *Interfaz codireccional*

El término codireccional se utiliza para describir un interfaz a través del cual la información y las señales de temporización asociadas se transmiten en el mismo sentido (véase la figura 1/G.703).

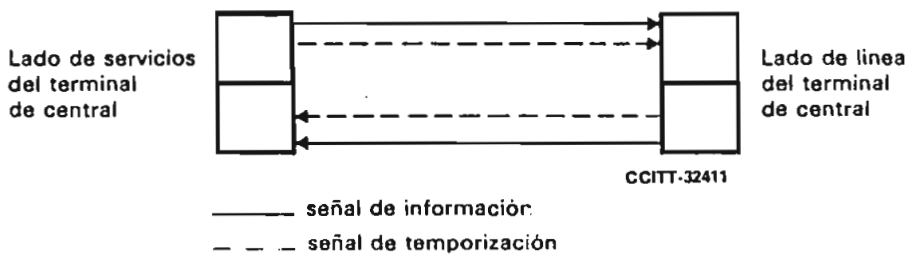


FIGURA 1/G.703
Interfaz codireccional

1.1.4.2 *Interfaz de reloj centralizado*

El término reloj centralizado se utiliza para describir un interfaz donde, para ambos sentidos de transmisión de la señal de información, las señales de temporización asociadas tanto al terminal de central en el lado de línea como al terminal de central en el lado de servicios se toman de un reloj centralizado que puede derivarse, por ejemplo, de ciertas señales de línea de llegada (véase la figura 2/G.703).

Observación – El interfaz codireccional o el interfaz de reloj centralizado deben utilizarse para redes sincronizadas y para redes plesiócronas cuyos relojes tengan la estabilidad requerida (véase la Recomendación G.811), a fin de asegurar un intervalo adecuado entre los deslizamientos.

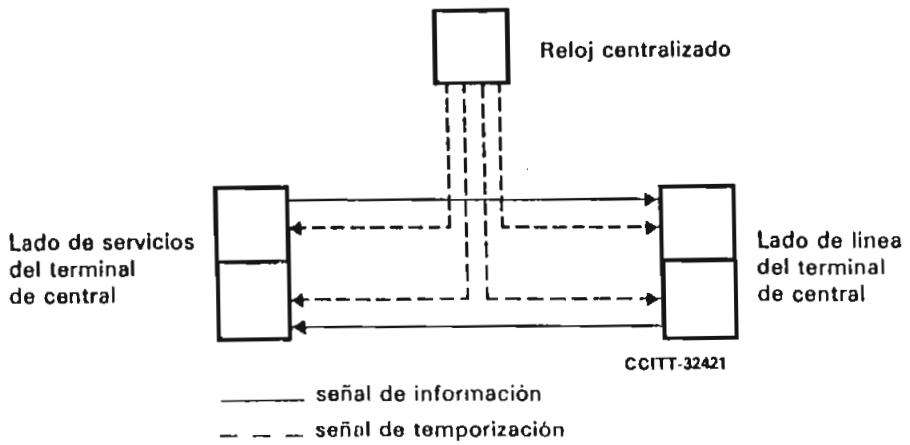


FIGURA 2/G.703

Interfaz de reloj centralizado

1.1.4.3 *Interfaz contradireccional*

El término contradireccional se utiliza para caracterizar un interfaz a través del cual las señales de temporización asociadas a ambas direcciones de transmisión se dirigen hacia el lado de servicios (por ejemplo, datos o señalización) del interfaz (véase la figura 3/G.703).

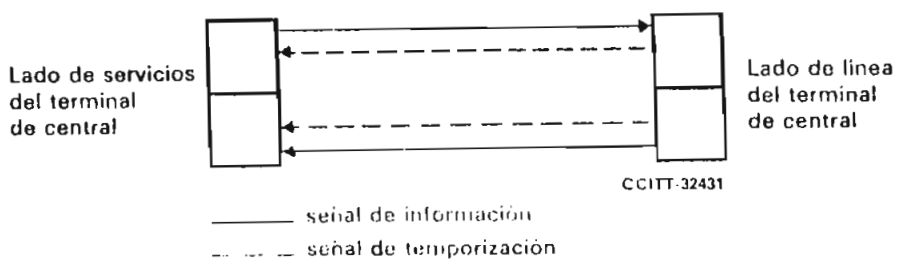


FIGURA 3/G.703

Interfaz contradireccional

1.2 *Características eléctricas*

1.2.1 *Características eléctricas del interfaz codireccional a 64 kbit/s*

1.2.1.1 *Consideraciones generales*

1.2.1.1.1 Velocidad binaria nominal: 64 kbit/s.

1.2.1.1.2 Tolerancia máxima para las señales transmitidas a través del interfaz: ± 100 ppm.

1.2.1.1.3 Las señales de temporización de 64 kHz y 8 kHz se transmitirán codireccionalmente con la señal de información.

1.2.1.1.4 Se utilizará un par simétrico para cada sentido de transmisión; se recomienda la utilización de transformadores.

1.2.1.1.5 *Reglas de conversión de código:*

Paso 1 – Un periodo de un bit a 64 kbit/s se divide en cuatro intervalos unitarios.

Paso 2 – Un 1 binario se codifica como un bloque constituido por los cuatro bits siguientes:

1 1 0 0

Paso 3 – Un 0 binario se codifica como un bloque constituido por los cuatro bits siguientes:

1 0 1 0

Paso 4 – La señal binaria se convierte en una señal de tres niveles alternando la polaridad de los bloques consecutivos.

Paso 5 – La alternancia de la polaridad de los bloques se viola cada octavo bloque. El bloque con violación indica el último bit de un octeto.

Estas reglas de conversión se ilustran en la figura 4/G.703.

1.2.1.2 *Especificaciones en los accesos de salida* (véase el cuadro 1/G.703)

1.2.1.3 *Especificaciones en los accesos de entrada*

La señal digital presentada en los accesos de entrada deberá corresponder a la definición precedente, con las modificaciones que introduzcan las características de los pares de interconexión. La atenuación de estos pares está comprendida entre 0 y 3 dB a la frecuencia de 128 kHz. Esta atenuación tendrá en cuenta posibles pérdidas debidas a la presencia de un repartidor digital entre los equipos.

Observación – Si el par simétrico está blindado, el blindaje se conectará a tierra en el acceso de salida, y se preverá, en caso necesario, su conexión a tierra en el acceso de entrada.

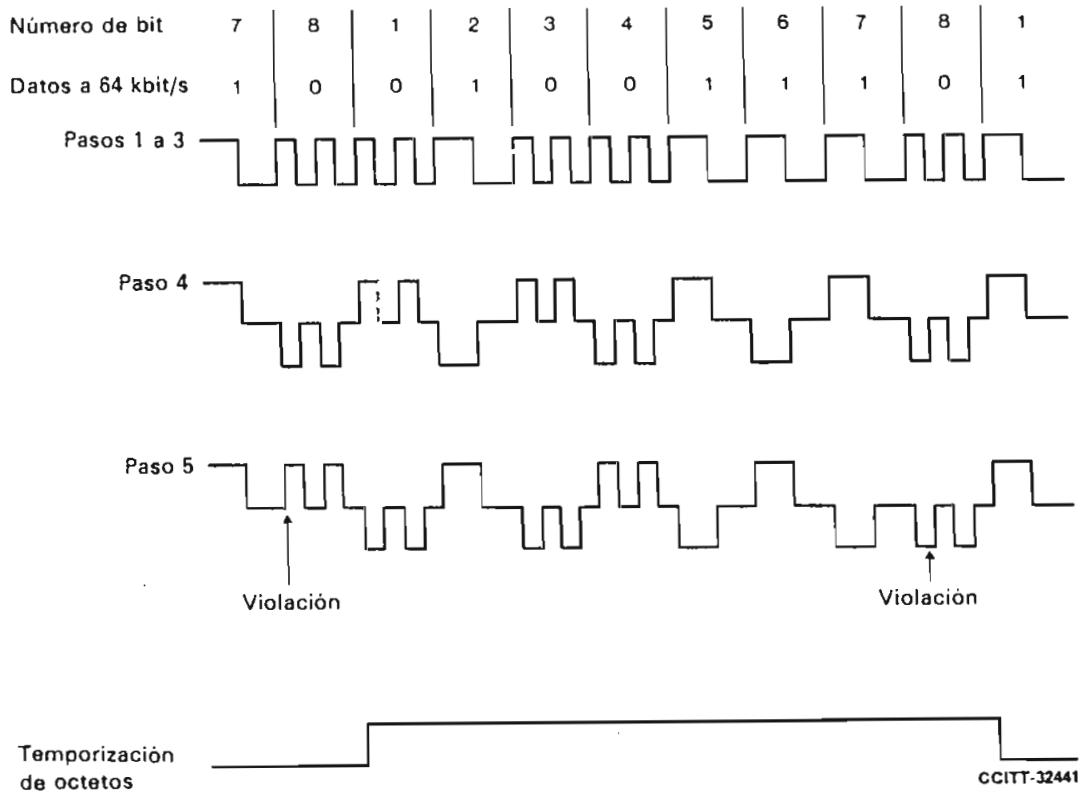
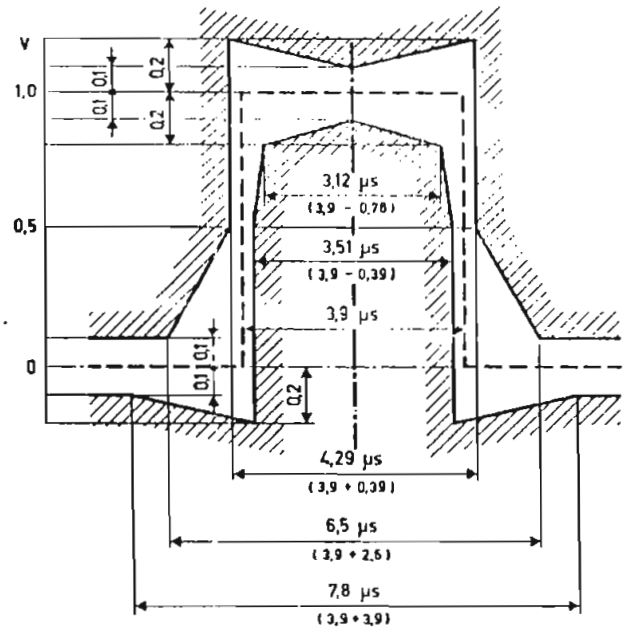
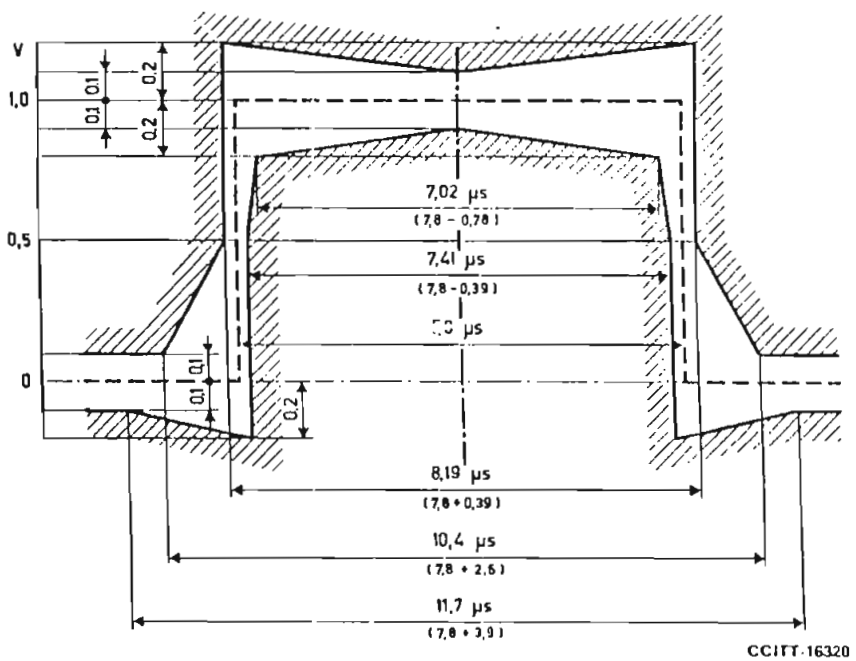


FIGURA 4/G.703



a) Plantilla para un impulso simple



b) Plantilla para un impulso doble

Observación – Los límites se aplican a impulsos de cualquier polaridad.

FIGURA 5/G.703

Plantillas para los impulsos en el caso de un interfaz codireccional a 64 kbit/s

CUADRO 1/G.703

Velocidad de símbolos	256 kbaudios
Forma del impulso (forma nominal, rectangular)	Todos los impulsos de una señal válida deben ajustarse a la plantilla de la figura 5/G.703, sea cual fuere la polaridad
Par(es) en cada sentido de transmisión	Un par simétrico
Impedancia de carga de prueba	120 ohmios, resistiva
Tensión de cresta nominal de una «marca» (impulso)	1,0 V
Tensión de cresta de un «espacio» (ausencia de impulso)	0 V ± 0,10 V
Anchura nominal del impulso	3,9 µs
Relación entre la amplitud de los impulsos positivos y la de los negativos en el centro del intervalo unitario	De 0,95 a 1,05
Relación entre la anchura de los impulsos positivos y la de los negativos en el punto de semiamplitud nominal	De 0,95 a 1,05

1.2.2 *Características eléctricas del interfaz de reloj centralizado a 64 kbit/s*

1.2.2.1 *Consideraciones generales*

1.2.2.1.1 Velocidad binaria nominal: 64 kbit/s. La tolerancia viene determinada por la estabilidad del reloj de la red (véase la Recomendación G.811).

1.2.2.1.2 Para cada sentido de transmisión deberá haber un par simétrico de hilos para la señal de datos. Además, deberá haber pares simétricos de hilos para transportar la señal de temporización compuesta (64 kHz y 8 kHz) de la fuente de reloj central al equipo terminal de central. Se recomienda la utilización de transformadores.

1.2.2.1.3 *Reglas de conversión de código*

Las señales de datos se codifican en código AMI y los impulsos tienen una relación de trabajo de 100%. Las señales compuestas de temporización transportan la información de temporización de bits a 64 kHz en código AMI con una relación de trabajo de 50 a 70%, y la información sobre la fase del octeto a 8 kHz mediante violaciones a la regla de codificación. La estructura de las señales y sus relaciones de fase nominales se muestran en la figura 6/G.703.

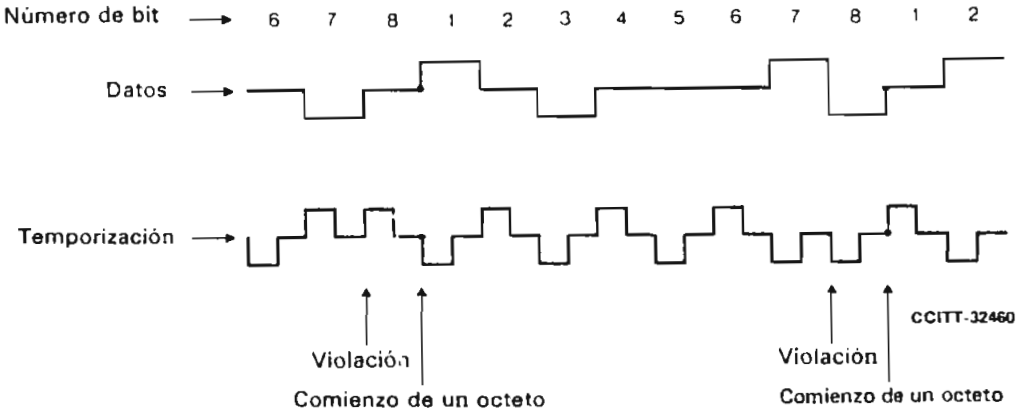


FIGURA 6/G.703

Estructura de las señales en los accesos de salida del terminal de central para el interfaz de reloj centralizado a 64 kbit/s

La corriente de datos en los accesos de salida debe temporizarse por el frente anterior del impulso de temporización, y el instante de detección en los accesos de entrada debe temporizarse por el frente posterior de cada impulso de temporización.

1.2.2.2 *Características de los accesos de salida* (véase el cuadro 2/G.703).

CUADRO 2/G.703

Parámetros	Datos	Temporización
Forma del impulso	Forma nominal rectangular, con tiempos de establecimiento y caída inferiores a 1 µs	Forma nominal rectangular, con tiempos de establecimiento y caída inferiores a 1 µs
Impedancia de carga nominal de prueba	110 ohmios, resistiva	110 ohmios, resistiva
Tensión de cresta de una «marca» (impulso)	a) 1,0 ± 0,1 V b) 3,4 ± 0,5 V	a) 1,0 ± 0,1 V b) 3,0 ± 0,5 V
Tensión de cresta de un «espacio» (ausencia de impulso)	a) 0 ± 0,1 V b) 0 ± 0,5 V	a) 0 ± 0,1 V b) 0 ± 0,5 V
Anchura nominal del impulso	a) 15,6 µs b) 15,6 µs	a) 7,8 µs b) 9,8 a 10,9 µs

Observación – La elección entre los juegos de parámetros a) y b) permite tener en cuenta diferentes ambientes de ruido de [illegible] y diferentes longitudes máximas de cable entre los tres equipos de central implicados.

1.2.2.3 *Características de los accesos de entrada*

Las señales digitales presentadas en los accesos de entrada deberán corresponder a la definición precedente, con las modificaciones que introduzcan las características de los pares de interconexión. Los parámetros variables del cuadro 2/G.703 permitirán obtener distancias de interconexión máximas típicas de 350 a 450 m.

1.2.2.4 *Características del cable*

Las características de transmisión del cable que ha de utilizarse deben seguir estudiándose.

1.2.3 *Características eléctricas del interfaz contradireccional a 64 kbit/s*

1.2.3.1 *Consideraciones generales*

1.2.3.1.1 Velocidad binaria: 64 kbit/s.

1.2.3.1.2 Tolerancia máxima para las señales que se transmitan por el interfaz: ± 100 ppm.

1.2.3.1.3 Para cada sentido de transmisión deberá haber dos pares simétricos: uno para la señal de datos y otro para una señal de temporización compuesta (64 kHz y 8 kHz). Se recomienda la utilización de transformadores.

Observación – Si es necesario, a escala nacional, proporcionar una indicación de alarma separada a través del interfaz, esto puede realizarse interrumpiendo la señal de temporización de 8 kHz en el sentido de que se trate, es decir, inhibiendo las violaciones de código introducidas en la señal de temporización compuesta correspondiente (véase más adelante).

1.2.3.1.4 *Reglas de conversión de código*

Las señales de datos se codifican en código AMI y los impulsos tienen una relación de trabajo del 100%. Las señales compuestas de temporización transportan la información de temporización de bits a 64 kHz mediante el empleo del código AMI con una relación de trabajo del 50%, y la información sobre la fase de la señal de temporización de octetos a 8 kHz, introduciendo violaciones a la regla de codificación. La estructura de las señales [illegible] ones de fase en los [illegible] de salida [illegible] datos se muestran en la figura 7/G.703.

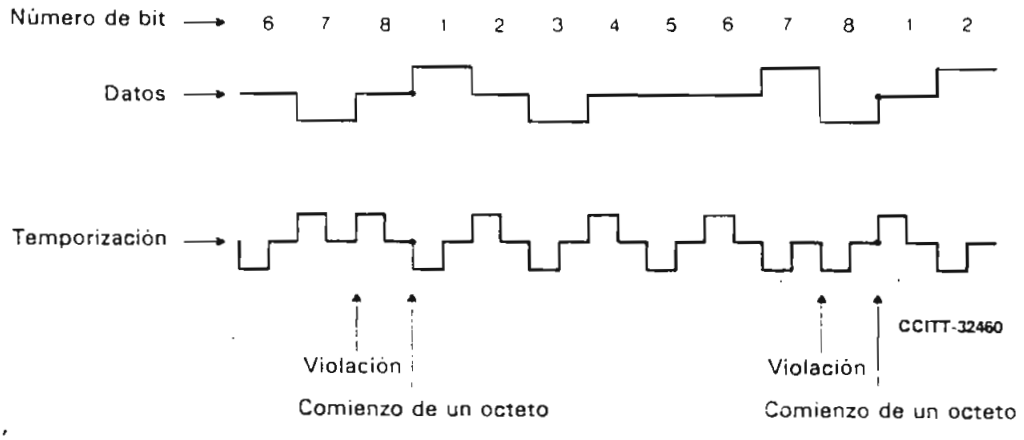


FIGURA 7/G.732

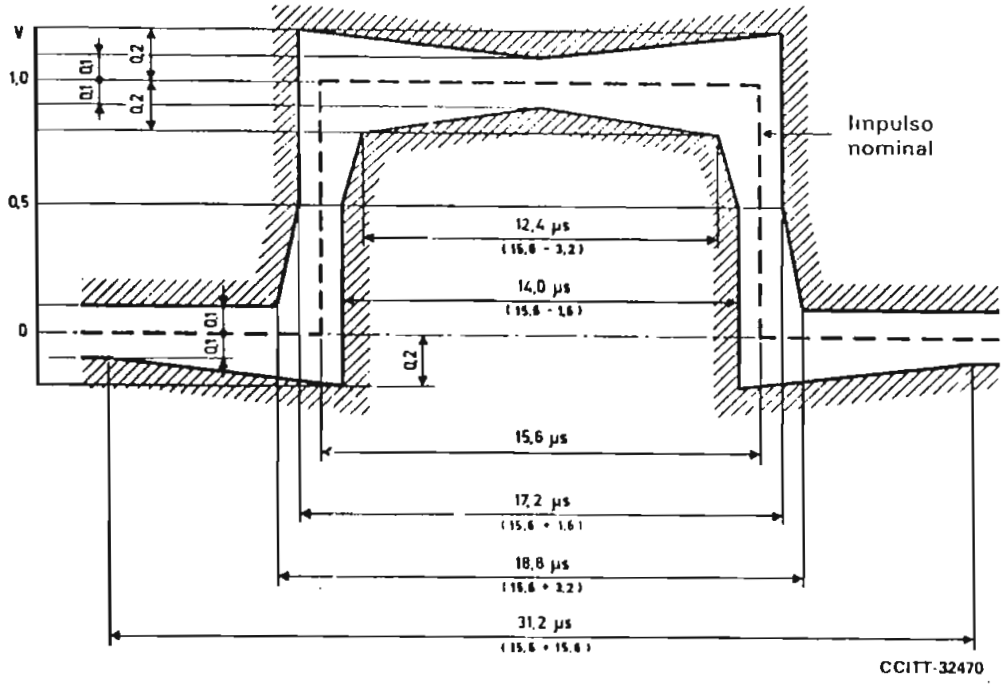
Estructura de las señales en los accesos de salida de datos para el interfaz contradireccional a 64 kbit/s

Los impulsos de datos recibidos del lado de servicios (por ejemplo: datos o señalización) del interfaz se retardarán algo en relación con los impulsos de temporización correspondientes. El instante de detección de un impulso de datos recibido del lado de línea (por ejemplo: MIC) del interfaz deberá situarse, pues, en el flanco anterior del siguiente impulso de temporización.

1.2.3.1.5 *Especificaciones en los accesos de salida* (véase el cuadro 3/G.703)

CUADRO 3/G.703

Parámetros	Datos	Temporización
Forma del impulso (forma nominal, rectangular)	Todos los impulsos de una señal válida deben ajustarse a la plantilla de la figura 8/G.703, sea cual fuere la polaridad	Todos los impulsos de una señal válida deben ajustarse a la plantilla de la figura 9/G.703, sea cual fuere la polaridad
Par(es) en cada sentido de transmisión	Un par simétrico	Un par simétrico
Impedancia de carga de prueba	120 ohmios, resistiva	120 ohmios, resistiva
Tensión de cresta nominal de una « marca » (impulso)	1,0 V	1,0 V
Tensión de cresta de un « espacio » (ausencia de impulso)	0 V ± 0,1 V	0 V ± 0,1 V
Anchura nominal del impulso	15,6 µs	7,8 µs
Relación entre la amplitud de los impulsos positivos y la de los negativos en el centro del intervalo de un impulso	De 0,95 a 1,05	De 0,95 a 1,05
Relación entre la anchura de los impulsos positivos y la de los negativos en el punto de semiamplitud nominal	De 0,95 a 1,05	De 0,95 a 1,05



Observación 1 – Cuando un impulso va inmediatamente seguido de otro de polaridad opuesta, los límites de tiempo para el paso por los puntos de amplitud cero de los impulsos serán ± 0,8 µs.

Observación 2 – Los instantes en los que debe producirse la transición de un estado a otro de la señal de datos los determina la señal de temporización. En el lado de servicios (p.e., datos o señalización) del interfaz es esencial que estas transiciones no sean iniciadas antes de los instantes definidos por la señal de temporización recibida.

FIGURA 8/G.703

Plantilla para el impulso de datos en el caso de un interfaz contradireccional a 64 kbit/s

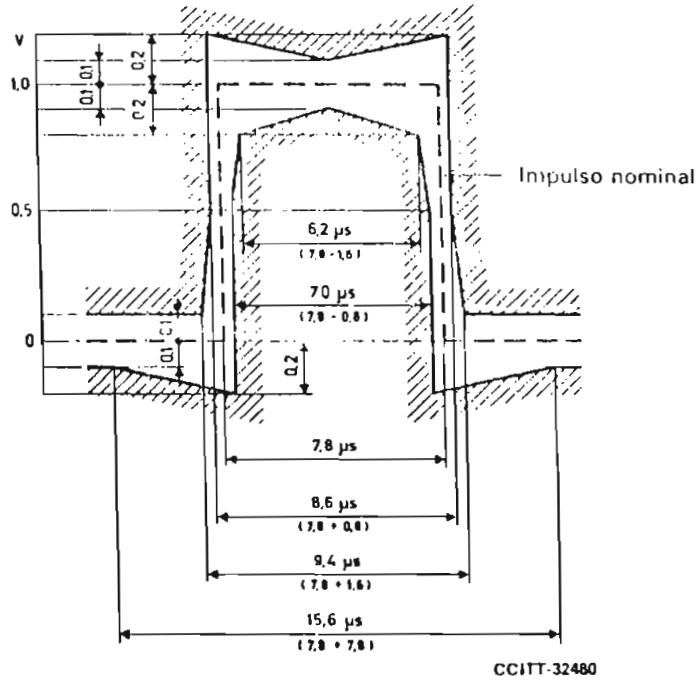


FIGURA 9/G.703

Plantilla para el impulso de temporización en el caso de un interfaz contradireccional a 64 kbit/s

CUADRO 4/G.703

Interfaz digital a 1544 kbit/s[a]

Ubicación		Repartidor digital
Velocidad binaria		1544 kbit/s
Par(es) en cada sentido de transmisión		Un par simétrico
Código		AMI[b] o B8ZS[c]
Impedancia de carga de prueba		100 ohmios, resistiva
Forma nominal del impulso		Rectangular
Nivel de la señal[d]	Potencia a 772 kHz	De +12 dBm a +19 dBm
	Potencia a 1544 kHz	Por lo menos 25 dB por debajo del nivel de potencia a 772 kHz

a) La plantilla del impulso para el interfaz digital de primer orden se reproduce en la figura 10/G.703.
b) Véase el § 2.5.
c) El código B8ZS es un código AMI modificado en el cual se reemplazan ocho ceros consecutivos por 000+−0−+ si el impulso precedente era +; y por 000−+0+− si era −.
d) El nivel de la señal es el nivel de potencia medido en una banda de 3 kHz en el jack de entrada para una secuencia «todos 1» transmitida.

3 Interfaz a 6312 kbit/s

3.1 La interconexión de señales a 6312 kbit/s a los fines de la transmisión se hace en el repartidor digital.

3.2 La velocidad binaria de la señal debe ser de 6312 kbit/s ± 30 ppm.

3.3 Se utilizará un par simétrico con una impedancia característica de 110 ohmios, o un par coaxial con una impedancia característica de 75 ohmios, para cada sentido de transmisión. El jack del repartidor conectado a un par por el que llegan las señales al repartidor se denomina jack de entrada. El jack del repartidor conectado a un par por el que salen las señales del repartidor se denomina jack de salida.

3.4 La impedancia de carga de prueba será resistiva de 110 o de 75 ohmios según proceda.

3.5 Se utilizará un código seudoternario como se indica en el cuadro 5/G.703.

3.6 La forma de un impulso aislado medido en el jack de salida o en el de entrada deberá quedar dentro de los límites de la plantilla de la figura 11/G.703 o la de la figura 12/G.703, y cumplir las demás condiciones indicadas en el cuadro 5/G.703.

3.7 La tensión en un intervalo de tiempo que contenga un 0 (espacio) no será superior al mayor de los dos valores siguientes: valor producido en dicho intervalo por otros impulsos (marcas) conformes a la plantilla de la figura 11/G.703, o ± 0,1 de la amplitud de cresta del impulso (marca).

4 Interfaz a 32 064 kbit/s

4.1 La interconexión de señales a 32 064 kbit/s para fines de transmisión se efectúa en un repartidor digital.

4.2 La señal deberá tener una velocidad binaria de 32 064 kbit/s con una tolerancia de ± 10 ppm.

4.3 Se utilizará un par coaxial para cada sentido de transmisión. El jack del repartidor conectado a un par coaxial por el que entran las señales en el repartidor se denomina jack de entrada. El jack del repartidor conectado a un par coaxial por el que salen las señales del repartidor se denomina jack de salida.

CUADRO 5/G.703

Interfaz digital a 6312 kbit/s[a]

Ubicación	Repartidor digital	
Velocidad binaria	6312 kbit/s	
Par(es) en cada sentido de transmisión	Un par simétrico	Un par coaxial
Código	B6ZS [b]	B8ZS[c]
Impedancia de carga de prueba	110 ohmios, resistiva	75 ohmios, resistiva
Forma nominal del impulso	Rectangular, determinada por la atenuación del cable (véase la figura 11/G.703)	Rectangular (véase la figura 12/G.703)
Nivel de la señal	Cuando se transmite una secuencia todos 1 deben obtenerse los siguientes niveles de potencia, medidos en una banda de 3 kHz:	
	3156 kHz: de 0,2 a 7,3 dBm 6312 kHz: −20 dBm o menos	3156 kHz: de 6,2 a 13,3 dBm 6312 kHz: −14 dBm o menos

[a] En las figuras 11/G.703 y 12/G.703 se reproduce la plantilla del impulso para el interfaz digital de segundo orden.

[b] El código B6ZS es un código AMI modificado en el cual seis ceros consecutivos se reemplazan por 0+−0−+ si el impulso anterior era +, y por 0−+0+− si era −.

[c] El código B8ZS es un código AMI modificado en el cual se reemplazan ocho ceros consecutivos por 000+−0−+ si el impulso precedente era +, y por 000− +0+− si era −.

4.4 La impedancia de carga de prueba deberá ser de 75 ohmios ± 5%, resistiva, y el método de prueba deberá ser directo.

4.5 Se utilizará un código AMI seudoaleatorizado.

4.6 La forma de un impulso aislado medido en el jack de entrada deberá estar comprendida en la plantilla de la figura 13/G.703.

4.7 La tensión en un intervalo de tiempo que contenga un 0 (espacio) no será superior al mayor de los dos valores siguientes: el valor producido en ese intervalo de tiempo por otros impulsos (marcas) comprendidos en la plantilla de la figura 13/G.703, o ± 0,1 de la amplitud de cresta del impulso (marca).

4.8 Para una secuencia «todos uno» transmitida, la potencia medida en una banda de 3 kHz en el jack de entrada será la siguiente:

16 032 kHz: de +5 dBm a +12 dBm,
32 064 kHz por lo menos 20 dB por debajo del nivel de potencia a 16 032 kHz.

4.9 Impedancia de los conectores y pares coaxiales en el repartidor: 75 ohmios ± 5%.

	T	Fórmula de la curva
Curva inferior	$T \leqslant -0{,}41$ $-0{,}41 \leqslant T \leqslant 0{,}24$ $0{,}24 \leqslant T$	0 $0{,}5\left[1+\operatorname{sen}\frac{\pi}{2}\left(1+\frac{T}{0{,}205}\right)\right]$ $0{,}331\, e^{-1{,}9\,(T-0{,}3)}$
Curva superior	$T \leqslant -0{,}72$ $-0{,}72 \leqslant T \leqslant 0{,}2$ $0{,}2 \leqslant T$	0 $0{,}5\left[1+\operatorname{sen}\frac{\pi}{2}\left(1+\frac{T}{0{,}36}\right)\right]$ $0{,}1+0{,}72\, e^{-2{,}13\,(T-0{,}2)}$

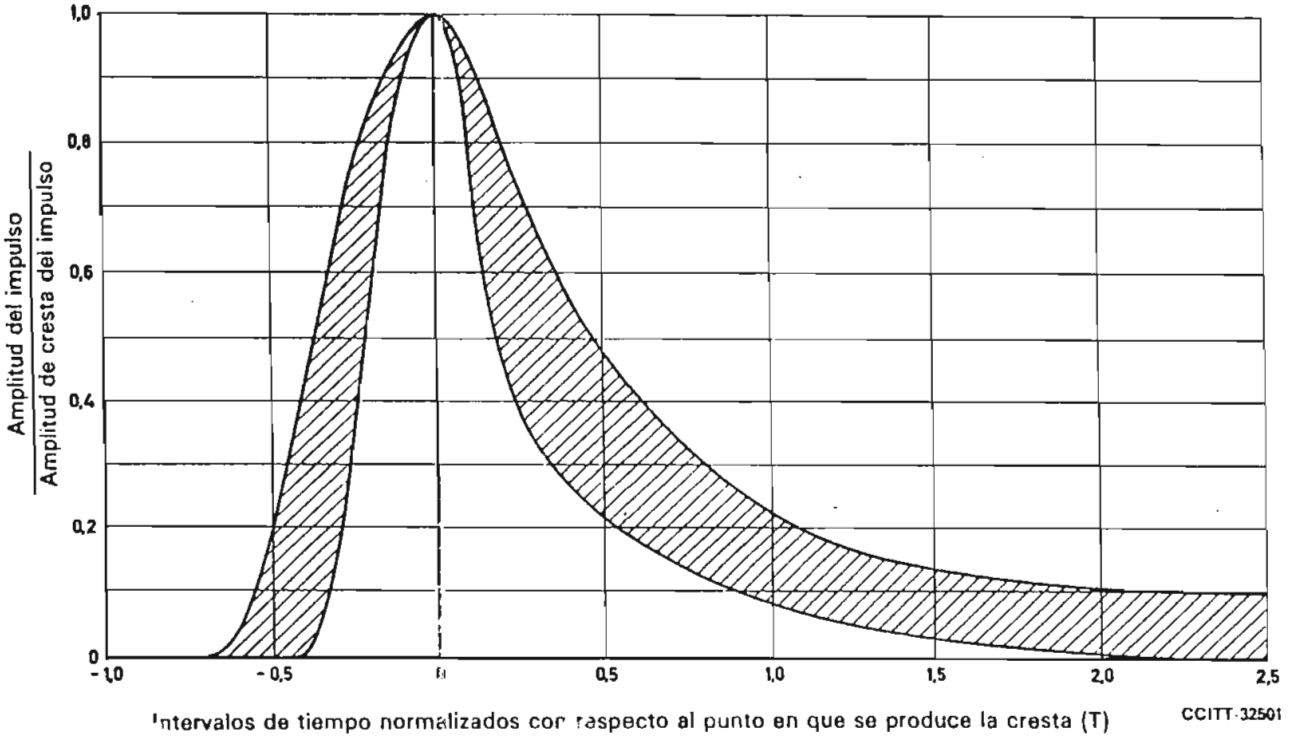
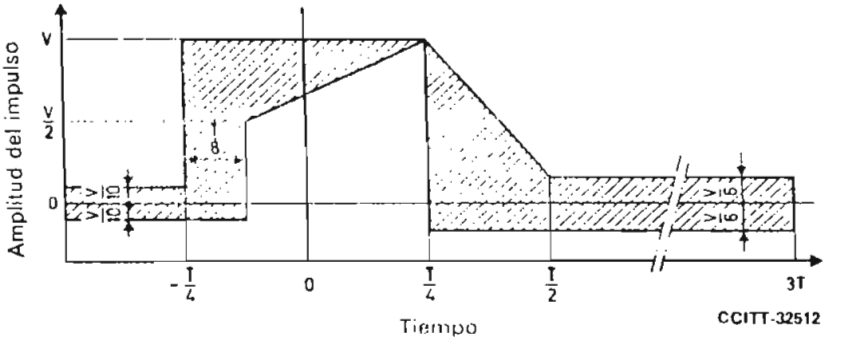


FIGURA 11/G.703

Plantilla del impulso para el interfaz de pares simétricos a 6312 kbit/s



T Anchura del intervalo de tiempo

FIGURA 12/G.703

Plantilla del impulso para el interfaz de pares coaxiales a 6312 kbit/s

	T	Fórmula de la curva
Curva inferior	$-0{,}36 \leqslant T < -0{,}30$	$5{,}76\ T + 2{,}07$
	$-0{,}30 \leqslant T < 0$	$0{,}5 \left[1 + \text{sen} \frac{\pi}{2} \left(1 + \frac{T}{0{,}25}\right)\right]$
	$0 \leqslant T < 0{,}22$	$0{,}5 \left[1 + \text{sen} \frac{\pi}{2} \left(1 + \frac{T}{0{,}16}\right)\right]$
	$0{,}22 < T$	$0{,}11\ e^{-3{,}42\ (T\ -\ 0{,}3)}$
Curva superior	$-0{,}65 \leqslant T < 0$	$1{,}05\ [1 - e^{-4{,}6\ (T\ -\ 0{,}65)}]$
	$0 \leqslant T < 0{,}25$	$0{,}5 \left[1 - \text{sen} \frac{\pi}{2} \left(1 + \frac{T}{0{,}28}\right)\right]$
	$0{,}25 < T$	$0{,}11 + 0{,}407\ e^{-2{,}1\ (T\ -\ 0{,}29)}$

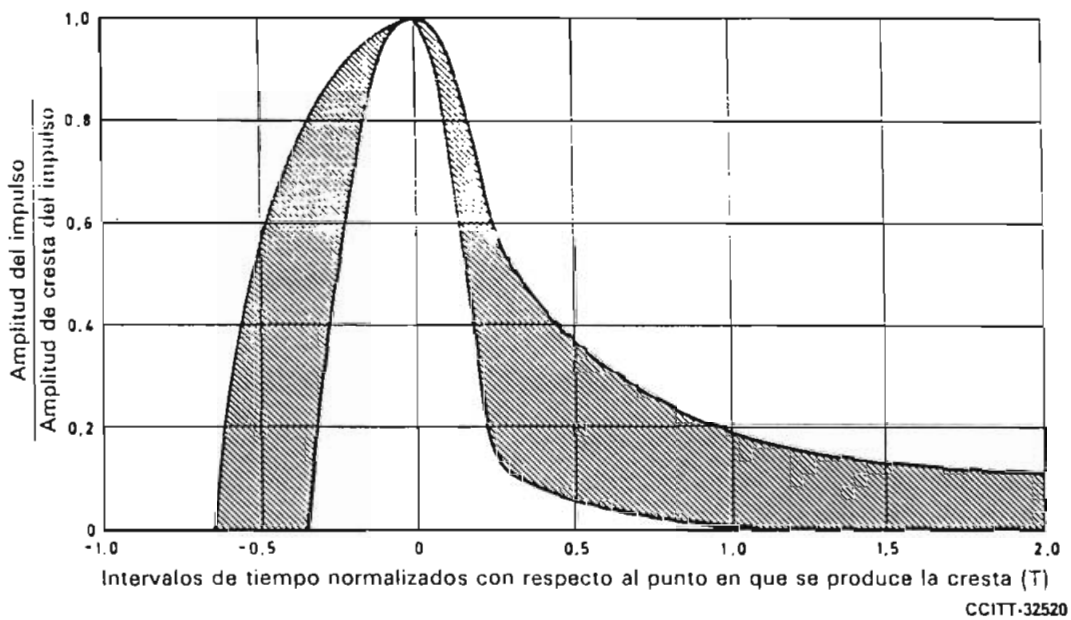


FIGURA 13/G.703

Plantilla del impulso para el interfaz de pares coaxiales a 32 064 kbit/s

5 Interfaz a 44 736 kbit/s

5.1 La interconexión de señales a 44 736 kbit/s para fines de transmisión se hace en un repartidor digital.

5.2 La velocidad binaria de la señal debe ser de 44 736 kbit/s ± 20 ppm.

5.3 Se utilizará un par coaxial para cada sentido de transmisión. El jack del repartidor conectado a un par coaxial por el que entran las señales al repartidor se denomina jack de entrada. El jack del repartidor conectado a un par por el que salen las señales del repartidor se denomina jack de salida.

5.4 La impedancia de carga de prueba será de 75 ohmios ± 5%, resistiva, y el método de prueba será directo.

5.5 Se utilizará un código bipolar como el especificado en el § 5.5.1.

5.5.1 *Código B3ZS*

El código B3ZS *(bipolar with three-zero substitution)* es una versión modificada del formato bipolar de impulsos, denominada código bipolar con sustitución de tres ceros. Los bits lógicos 1 tienen un ciclo de trabajo del 50%, y son, generalmente, positivos y negativos alternativamente con respecto al nivel lógico cero. Las excepciones están constituidas por aquellos casos en que aparecen tres ceros lógicos consecutivos en el tren de

bits. En el formato B3ZS, cada bloque de tres ceros consecutivos se sustituye por B0V o 00V, donde B representa un impulso conforme a la regla bipolar y V representa un impulso que viola la regla bipolar. Se elige entre B0V y 00V de tal manera que el número de impulsos B entre impulsos V consecutivos sea impar. Deben insertarse bits de alineación de trama de conformidad con la Recomendación G.752.

5.6 La forma de un impulso aislado medido en el jack de entrada deberá ajustarse a la plantilla de la figura 14/G.703.

5.7 La tensión en un intervalo de tiempo que contenga un cero (espacio) no será superior al mayor de los dos valores siguientes: el valor producido en dicho intervalo de tiempo por otros impulsos (marcas) conformes a la plantilla de la figura 14/G.703 o ± 0,05 de la amplitud de cresta del impulso (marca).

5.8 Cuando se transmita una secuencia todos 1, la potencia medida en una banda de 3 kHz en el jack de entrada deberá ser la siguiente:

22 368 kHz: de −1,8 a +5,7 dBm,
44 736 kHz: por lo menos 20 dB por debajo del nivel de potencia a 22 368 kHz.

5.9 El repartidor digital para señales a 44 736 kbit/s tendrá las características especificadas en los § 5.9.1 y 5.9.2.

5.9.1 La atenuación entre los jacks de entrada y de salida en el repartidor será la siguiente:

0,60 ± 0,55 dB a 22 368 kHz
(para cualquier combinación de características de atenuación uniforme o conformada).

5.9.2 Impedancia de los conectores y cables coaxiales en el repartidor: 75 ohmios ± 5%.

	T	Fórmula de la curva
Curva inferior	$T \leq -0,36$	0
	$-0,36 \leq T \leq 0,28$	$0,5\left[1 + \text{sen}\,\frac{\pi}{2}\left(1 + \frac{T}{0,18}\right)\right]$
	$0,28 \leq T$	$0,11\,e^{-3,42\,(T\,-\,0,3)}$
Curva superior	$T \leq -0,65$	0
	$-0,65 \leq T \leq 0$	$1,05\,[1 - e^{-4,6\,(T\,+\,0,65)}]$
	$0 \leq T \leq 0,36$	$0,5\left[1 + \text{sen}\,\frac{\pi}{2}\left(1 + \frac{T}{0,34}\right)\right]$
	$0,36 \leq T$	$0,05 + 0,407\,e^{-1,84\,(T\,-\,0,36)}$

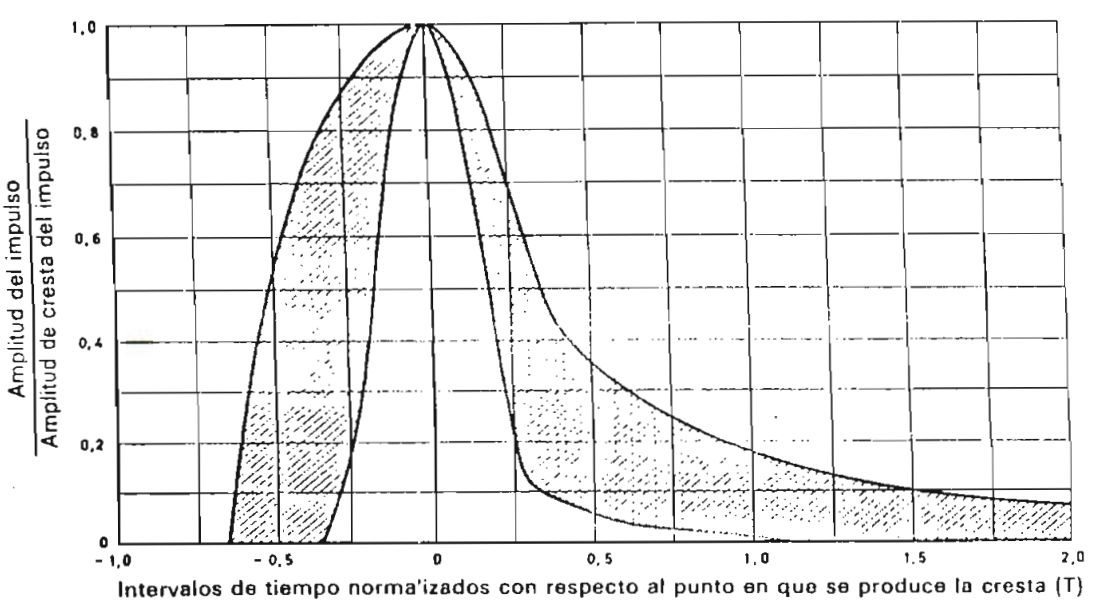


FIGURA 14/G.703

Plantilla del impulso para el interfaz de pares coaxiales a 44 736 kbit/s

6 Interfaz a 2048 kbit/s

6.1 *Características generales*

Velocidad binaria: 2048 kbit/s ± 50 ppm
Código: HDB3 (bipolar de alta densidad de orden 3) (la descripción de este código figura en el anexo A)

6.2 *Especificaciones en los accesos de salida* (véase el cuadro 6. G.703)

CUADRO 6/G.703

Forma del impulso (forma nominal : rectangular)	Todas las marcas de una señal válida deberán ajustarse a la plantilla (figura 15/G.703), independientemente del signo. El valor V corresponde al valor nominal de cresta	
Par(es) en cada sentido de transmisión	Un par coaxial (vease el § 6.4)	Un par simétrico (vease el § 6.4)
Impedancia de carga de prueba	75 ohmios, resistiva	120 ohmios, resistiva
Tensión nominal de cresta de una marca (impulso)	2,37 V	3 V
Tensión de cresta de un espacio (ausencia de impulso)	0 ± 0,237 V	0 ± 0,3 V
Anchura nominal del impulso	244 ns	
Relación entre la amplitud de los impulsos positivos y la de los negativos en el punto medio del intervalo de un impulso	De 0,95 a 1,05	
Relación entre la anchura de los impulsos positivos y la de los negativos en los puntos de semiamplitud nominal	De 0,95 a 1,05	
Fluctuación de fase máxima cresta a cresta en un acceso de salida	Véase el § 2 de la Recomendación G.823	

6.3 *Especificaciones en los accesos de entrada*

La señal digital presentada en los accesos de entrada deberá corresponder a la definición precedente, con las modificaciones que introduzcan las características de los pares de interconexión. La atenuación de estos pares deberá seguir una ley $\sqrt{f}$ y la atenuación a la frecuencia de 1024 kHz deberá estar comprendida entre 0 y 6 dB. Esta atenuación tendrá en cuenta posibles pérdidas debidas a la presencia de un repartidor digital entre los equipos.

En lo relativo a la fluctuación de fase que ha de tolerarse en los accesos de entrada, véase el § 3 de la Recomendación G.823.

7 Interfaz a 8448 kbit/s

7.1 *Características generales*

Velocidad binaria: 8448 kbit/s ± 30 ppm
Código: HDB3 (la descripción de este código figura en el anexo A)

7.2 *Especificaciones en los accesos de salida* (indicadas en el cuadro 7/G.703)

CUADRO 7/G.703

Forma del impulso (forma nominal : rectangular)	Todas las marcas de una señal válida deberán ajustarse a la plantilla (figura 16/G.703), independientemente del signo
Par(es) en cada sentido de transmisión	Un par coaxial (vease el § 7.4)
Impedancia de carga de prueba	75 ohmios, resistiva
Tensión nominal de cresta de una marca (impulso)	2,37 V
Tensión de cresta de un espacio (ausencia de impulso)	0 ± 0,237 V
Anchura nominal del impulso	59 ns
Relación entre las anchuras de los impulsos positivos y la de los negativos en el punto medio del intervalo de un impulso	De 0,95 a 1,05
Relación entre las anchuras de los impulsos positivos y los negativos para los puntos de semiamplitud nominal	De 0,95 a 1,05
Fluctuación de fase máxima cresta a cresta en un acceso de salida	Véase el § 2 de la Recomendación G.823

7.3 *Especificaciones en los accesos de entrada*

La señal digital presentada en los accesos de entrada deberá corresponder a la definición precedente, con las modificaciones que introduzcan las características de los pares de interconexión. La atenuación de estos pares deberá seguir una ley $\sqrt{f}$ y la atenuación a la frecuencia de 4224 kHz deberá estar comprendida entre 0 y 6 dB. Esta atenuación tendrá en cuenta posibles pérdidas debidas a la presencia de un repartidor digital entre los equipos.

En lo relativo a la fluctuación de fase que ha de tolerarse en los accesos de entrada, véase el § 3 de la Recomendación G.823.

La pérdida de retorno en los accesos de entrada deberá tener los siguientes valores mínimos provisionales:

Frecuencias correspondientes al porcentaje de la velocidad binaria nominal	Pérdida de retorno
2,5 a 5%	12 dB
5 a 100%	18 dB
100 a 150%	14 dB

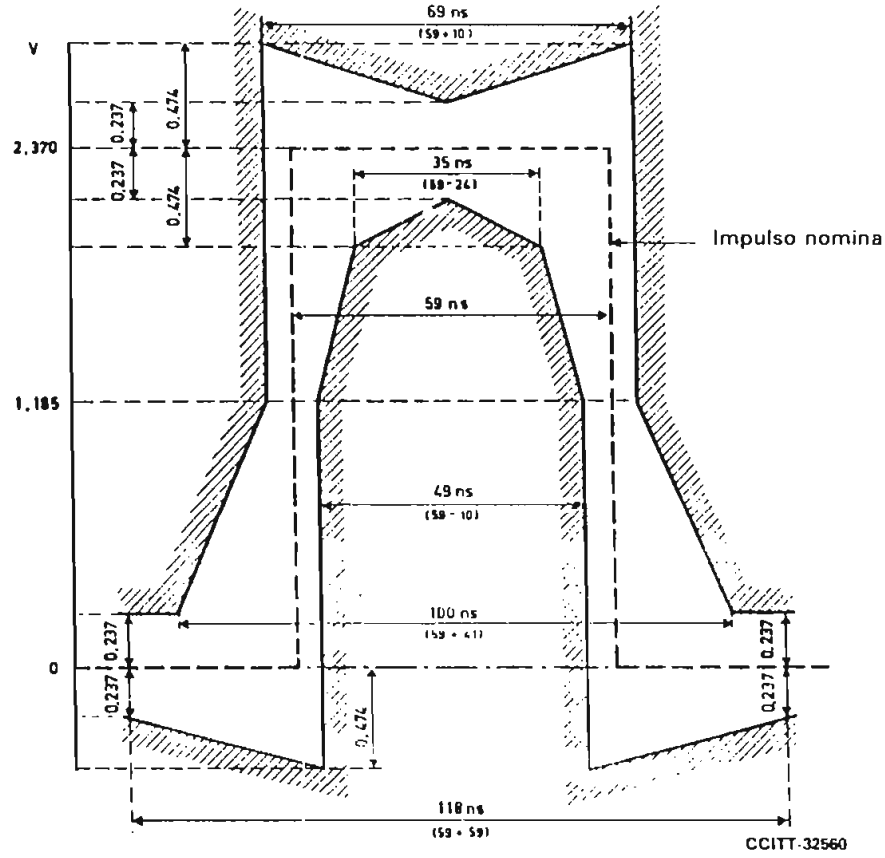


FIGURA 16/G.703

Plantilla para el impulso en el caso de un interfaz a 8448 kbit/s

Nota – La necesidad de incluir en la presente Recomendación un requisito en materia de inmunidad a la interferencia se halla en estudio.

7.4 *Puesta a tierra del conductor exterior o del blindaje*

El conductor exterior del par coaxial deberá conectarse a tierra en el acceso de salida y también deberá preverse la conexión a tierra de este conductor en el acceso de entrada, si es necesario.

8 Interfaz a 34 368 kbit/s

8.1 *Características generales*

Velocidad binaria: 34 368 kbit/s ± 20 ppm
Código: HDB3 (en el anexo A figura una descripción de este código)

8.2 *Especificación en los accesos de salida* (indicada en el cuadro 8/G.703)

8.3 *Especificaciones en los accesos de entrada*

La señal digital presentada en los accesos de entrada deberá corresponder a la definición precedente, con las modificaciones que introduzcan las características del cable de interconexión. Deberá asegurarse que la atenuación de este cable siga una ley $\sqrt{f}$ y que la atenuación a la frecuencia de 17 184 kHz esté comprendida entre 0 y 12 dB.

En lo relativo a la fluctuación de fase que ha de tolerarse en los accesos de entrada, véase el § 3 de la Recomendación G.823.

CUADRO 8/G.703

Forma del impulso (forma nominal : rectangular)	Todas las marcas de una señal válida deberán ajustarse a la plantilla (figura 17/G.703), independientemente del signo
Par(es) en cada sentido de transmisión	Un par coaxial (véase el § 8.4)
Impedancia de carga de prueba	75 ohmios, resistiva
Tensión nominal de cresta de una marca (impulso)	1,0 V
Tensión de cresta de un espacio (ausencia de impulso)	0 ± 0,1 V
Anchura nominal del impulso	14,55 ns
Relación entre la amplitud de los impulsos positivos y la de los negativos en el punto medio del intervalo de un impulso	De 0,95 a 1,05
Relación entre la anchura de los impulsos positivos y la de los negativos, en los puntos de semiamplitud nominal	De 0,95 a 1,05
Fluctuación de fase máxima cresta a cresta en un acceso de salida	Véase el § 2 de la Recomendación G.823

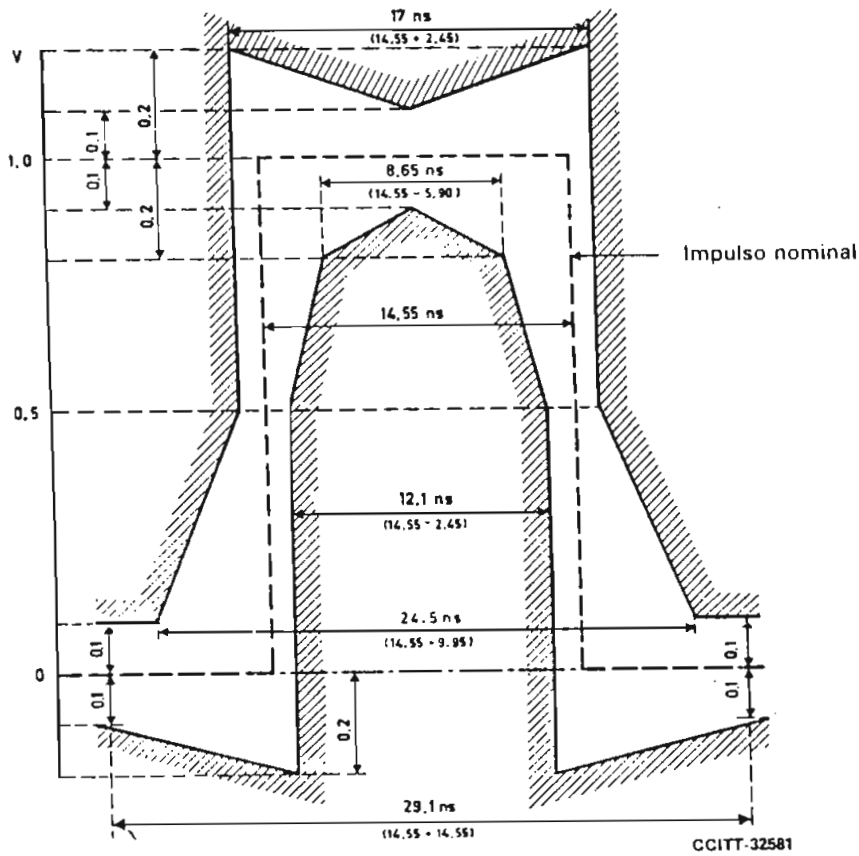


FIGURA 17/G.703

Plantilla para el impulso en el caso de un interfaz a 34368 kbit/s

La pérdida de retorno en los accesos de entrada deberá tener los siguientes valores mínimos provisionales:

Frecuencias correspondientes al porcentaje de la velocidad binaria nominal	Pérdida de retorno
2,5 a 5%	12 dB
5 a 100%	18 dB
100 a 150%	14 dB

Nota – La necesidad de incluir en la presente Recomendación un requisito en materia de inmunidad a la interferencia se halla en estudio.

8.4 *Puesta a tierra del conductor exterior o del blindaje*

Observación – El conductor exterior del par coaxial deberá conectarse a tierra en el acceso de salida: también deberá preverse la conexión a tierra de este conductor en el acceso de entrada, si es necesario.

9 Interfaz a 139 264 kbit/s

9.1 *Características generales*

Velocidad binaria: 139 264 kbit/s ± 15 ppm
Código: CMI (Coded Mark Inversion)

El código CMI es un código de 2 niveles sin retorno a cero en el cual el 0 binario se codifica de manera que los dos niveles de amplitud, A_1 y A_2, se obtienen consecutivamente, cada uno durante un periodo igual a la mitad de un intervalo unitario (T/2).

El 1 binario se codifica de modo que los niveles de amplitud, A_1 y A_2, se obtienen alternativamente cada uno durante un periodo igual a un intervalo unitario completo (T).

En la figura 18/G.703 se da un ejemplo.

Observación 1 – Para el 0 binario, existe siempre una transición positiva en el punto medio del intervalo de tiempo unitario binario.

Observación 2 – Para el 1 binario:

a) existe una transición positiva al comienzo del intervalo de tiempo unitario binario si el nivel precedente era A_1;

b) existe una transición negativa al comienzo del intervalo de tiempo unitario binario si el último 1 binario estaba codificado en el nivel A_2.

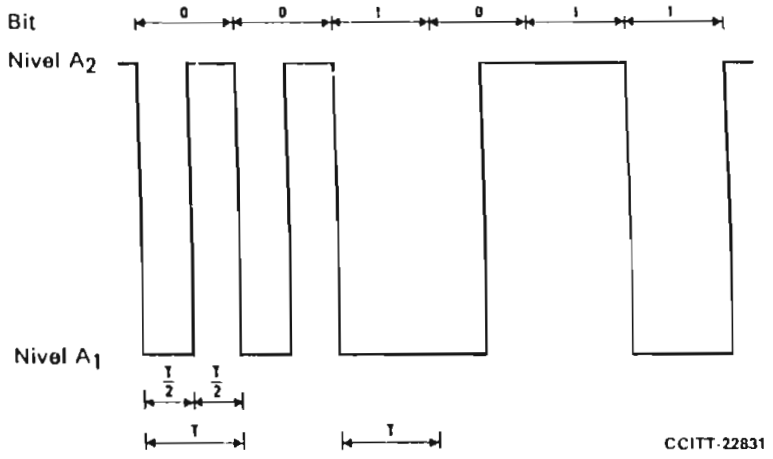


FIGURA 18/G.703
Ejemplos de señal bi[illegible]ificada en CMI

9.2 *Especificaciones en los accesos de salida* (indicadas en el cuadro 9/G.703)

Observación 1 – Se considera que un método basado en la medición de los niveles de la componente fundamental y del segundo (y posiblemente del tercer) armónico de una señal correspondiente a todos 0 binarios y todos 1 binarios es adecuado para verificar el cumplimiento de los requisitos indicados en el cuadro 9/G.703.

Los valores pertinentes están en estudio.

Observación 2 – Las plantillas de las figuras 19/G.703 y 20/G.703 se dan sólo como indicación, y no deben utilizarse necesariamente para mediciones.

9.3 *Especificaciones en los accesos de entrada*

La señal digital presentada en el acceso de entrada debe ser conforme a las indicaciones del cuadro 9/G.703, teniendo en cuenta las modificaciones producidas por las características del par coaxial de interconexión.

Debe suponerse que la atenuación del par coaxial sigue aproximadamente una ley $\sqrt{f}$ y que la pérdida de inserción máxima es de 12 dB a 70 MHz.

En lo relativo a la fluctuación de fase que ha de tolerarse en los accesos de entrada, véase el § 3 de la Recomendación G.823.

La característica de pérdida de retorno debe ser la misma que la especificada para el acceso de salida.

CUADRO 9/G.703

Forma nominal de los impulsos	Rectangular
Par(es) en cada sentido de transmisión	Un par coaxial
Impedancia de carga de prueba	75 ohmios, resistiva
Tensión cresta a cresta	1 ± 0,1 voltios
Sobreoscilación	≤ 5% de la tensión medida de cresta a cresta
Tiempo de subida entre el 10% y el 90% de la amplitud medida	≤ 2 ns
Tolerancia para la temporización de las transiciones (referida al valor medio de los puntos de semiamplitud de transiciones negativas)	Transiciones negativas: ± 0,1 ns Transiciones positivas en los extremos del intervalo unitario: ± 0,5 ns Transiciones positivas en el punto medio del intervalo unitario: + 0,35 ns
Pérdida de retorno	≥ 15 dB en la gama de frecuencias de 7 MHz a 210 MHz
Fluctuación de fase cresta a cresta máxima en un acceso de salida	Véase el § 2 de la Recomendación G.823

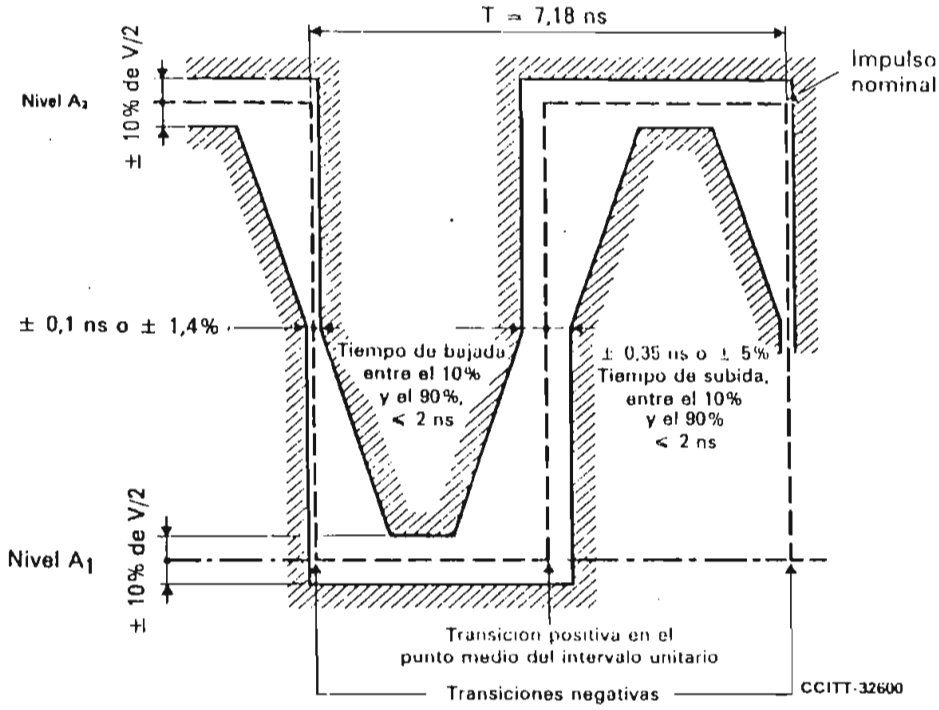
9.4 *Puesta a tierra del conductor exterior o del blindaje*

El conductor exterior del par coaxial debe estar conectado a tierra en el acceso de salida y debe preverse la puesta a tierra de este conductor, si es necesario, en el acceso de entrada.

10 Interfaz de sincronización a 2048 kHz

10.1 *Características generales*

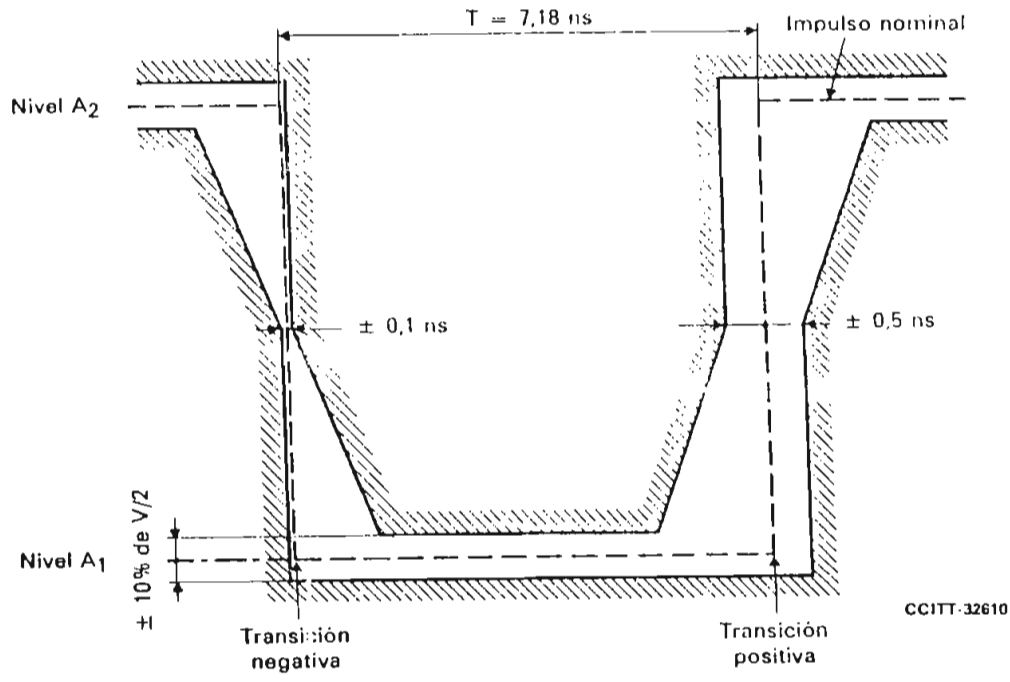
Se recomienda la utilización de este interfaz en todas aquellas aplicaciones donde se necesite sincronizar un equipo digital mediante una señal de sincronización externa de 2048 kHz.



Observación 1 – V es la amplitud nominal cresta a cresta.
Observación 2 – La plantilla no incluye la tolerancia para la sobreoscilación; véase el cuadro 9/G.703.

FIGURA 19/G.703

Plantilla para un impulso que corresponde a un 0 binario



Observación 1 – El impulso inverso tendrá las mismas características.
Observación 2 – V es la amplitud nominal cresta a cresta.
Observación 3 – La plantilla no incluye la tolerancia para la sobreoscilación; véase el cuadro 9/G.703.

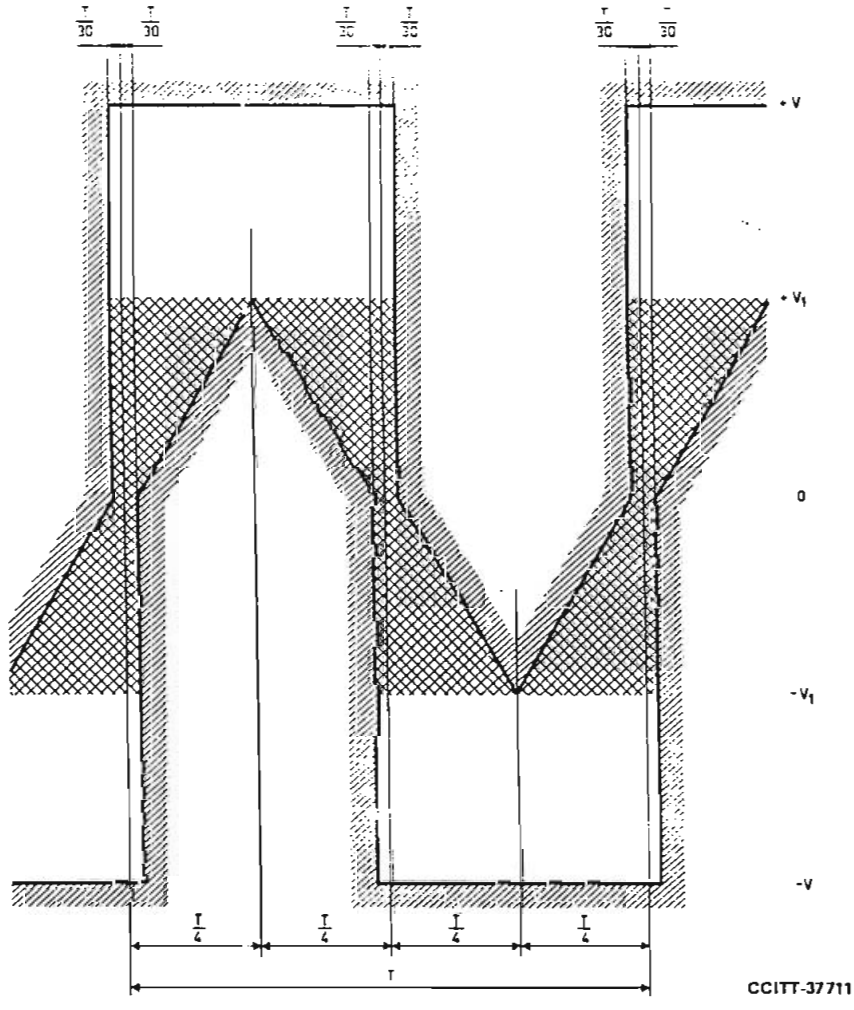
FIGURA 20/G.703

Plantilla para un impulso que corresponde a un 1 binario

10.2 *Especificaciones en el acceso de salida* (véase el cuadro 10/G.703)

CUADRO 10/G.703

Frecuencia	2048 kHz ± 50 ppm	
Forma de los impulsos	La señal debe ajustarse a la plantilla (figura 21/G.703) El valor V corresponde al valor de cresta máximo El valor V_1 corresponde al valor de cresta mínimo	
Tipo de par	Par coaxial (véase la observación en el § 10.3)	Par simétrico (véase la observación en el § 10.3)
Impedancia de carga de prueba	75 ohmios, resistiva	120 ohmios, resistiva
Tensión de cresta máxima (V_{op})	1,5	1,9
Tensión de cresta mínima (V_{op})	0,75	1,0
Fluctuación de fase máxima en el acceso de entrada	En estudio	



Zona sombreada en la que la señal debe ser monotónica

T Periodo medio de la señal de sincronización

FIGURA 21/G.703

Forma de la onda en un acceso de salida

10.3 *Especificaciones en los accesos de entrada*

La señal presentada en los accesos de entrada deberá corresponder a la definición precedente, con las modificaciones que introduzcan las características del par de interconexión.

Se supone que la atenuación de este par obedece a la ley $\sqrt{f}$, y la atenuación a la frecuencia de 2048 kHz deberá estar comprendida entre 0 y 6 dB (valor mínimo). Esta atenuación deberá tomar en cuenta cualquier pérdida provocada por la presencia de un repartidor digital entre los equipos.

El acceso de entrada deberá ser capaz de tolerar una señal digital con estas características eléctricas, pero modulada por una fluctuación de fase. Los valores de la fluctuación de fase se hallan en estudio.

La atenuación de retorno a 2048 kHz debe ser $\geq$ 15 dB.

Observación – El conductor exterior del par coaxial o el blindaje del par simétrico deberán conectarse a tierra en el acceso de salida; también deberá preverse la conexión a tierra de estos elementos en el acceso de entrada, si es necesario.

11 Interfaz a 97 728 kbit/s

11.1 La interconexión de señales a 97 728 kbit/s a los fines de la transmisión se hace en un repartidor digital.

11.2 La velocidad binaria de la señal debe ser de 97 728 kbit/s ± 10 partes por millón (ppm).

11.3 Se utilizará un par coaxial para cada sentido de transmisión. El jack del repartidor conectado a un par por el que llegan las señales al repartidor se denomina jack de entrada. El jack del repartidor conectado a un par por el que salen las señales del repartidor se denomina jack de salida.

11.4 La impedancia de carga de prueba será de 75 ohmios ± 5%, resistiva.

11.5 Se utilizará un código AMI [1] aleatorizado.

11.6 La forma de la señal a 97728 kbit/s en el acceso de salida estará comprendida dentro de los límites de la plantilla de la figura 22/G.703. La forma de la señal en el jack de entrada estará modificada por las características del cable de interconexión.

11.7 Los conectores y los pares en cable en el repartidor tendrán una resistencia de 75 ohmios ± 5%.

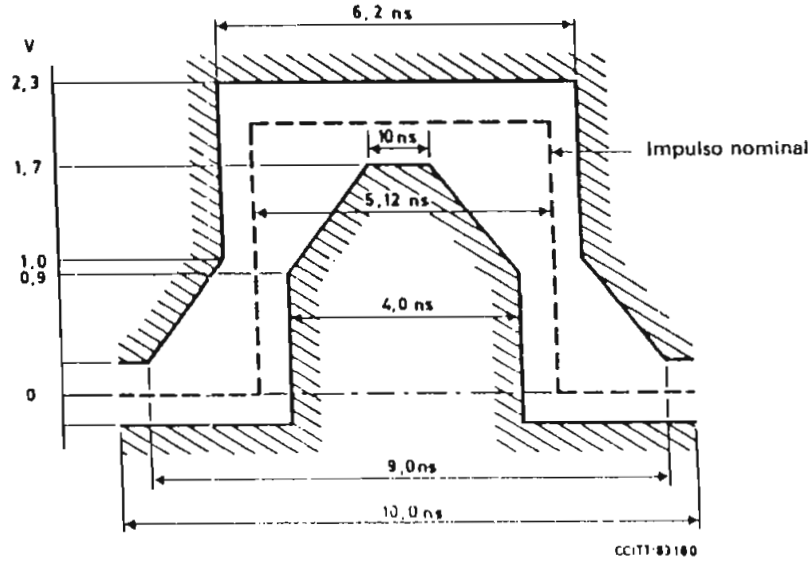


FIGURA 22/G.703

Plantilla del impulso en el acceso de salida a 97 728 kbit/s

[1] Un código AMI se aleatoriza mediante un aleatorizador de cinco pasos, con reiniciación y con el polinomio generador $x^5 + x^3 + 1$.

ANEXO B

LISTADO DEL PROGRAMA

LISTADO DEL PROGRAMA

```
(*PROGRAMA PRINCIPAL TELECOM.PAS, QUE ENLAZA A LAS UNIDADES INDICADAS*)

Program Codigos;

(*LISTA DE UNIDADES QUE UTILIZA EL PROGRAMA PRINCIPAL*)

Uses CRT, GRAPH, CODIGOS1, CODIGOS2,
     CODIGOS3, MODULAC1, MODULAC2;

(*VARIABLES GENERALES DEL PROGRAMA*)

Var
  M, N, cp ,c1, c2, c3, ControlGr, ModoGr, Modo : Integer;
  Tecla, Tecla1, Tecla2, Tecla3,
  Tecla4, Tecla5, Tecla6 : Char;
  P1, P2 : Real;
  dato,reloj, coef : Array[1..60] Of Integer;
  bit : Array[1..30] Of Char;
  coef1 : Array[1..30] Of String;
  PRB1, PRB2 : string[6];
  TX : Array [1..16] Of string[10];
Label
  Lazo, Lazo1, Lazo2, Lazo3, Lazo4;
(***********************************************)
Procedure CuadroPrincipal;
(*PROCEDIMIENTO QUE REALIZA LA PANTALLA DE PRESENTACION INICIAL, EN LA QUE
  SELECCIONA, CODIGOS DE LINEA, MODULACION DIGITAL O SALIR AL DOS*)

Var
   i,j : Integer;
Begin;
ClrScr;
PintarFondo(1,1,80,25,7);
DibujarCuadro(1,2,80,25,1,7);
GotoXY(1,7);Write(#204);GotoXY(80,7);Write(#185);
For i:=2 To 79 Do
   Begin
   GotoXY(i,7);Write(#205);
   End;
GotoXY(1,22);Write(#204);GotoXY(80,22);Write(#185);
For i:=2 To 79 Do
   Begin
   GotoXY(i,22);Write(#205);
   End;

TextColor(1);
GotoXY(26,3);Write('ESCUELA POLITECNICA NACIONAL');
GotoXY(24,4);Write('FACULTAD DE INGENIERIA ELECTRICA');
GotoXY(16,5);Write('DEPARTAMENTO DE ELECTRONICA Y TELECOMUNICACIONES');
TextColor(0);TextBackground(15);
GotoXY(14,9);Write(' PROGRAMA PARA ANALISIS Y SIMULACION DE CODIFICACION Y
');
GotoXY(14,10);Write('     MODULACION EN UN SISTEMA DE TRANSMISION DIGITAL
 ');
DibujarCuadro(20,12,60,18,1,7);
TextColor(4);
GotoXY(29,14);Write('A)  CODIGOS DE LINEA');
GotoXY(29,15);Write('B)  MODULACION DIGITAL');
GotoXY(27,16);Write('End)  Salir al DOS');
TextColor(4);
GotoXY(3,23);Write('Iván Castro Ll.');GotoXY(65,23);Write('Marco Orbe R.');
Textcolor(1);
GotoXY(35,20);Write('Ingrese su selección...[   ]');
GotoXY(60,20);
End;

(*********************************************************)
                    PROGRAMA PRINCIPAL
```

```
(*************************************************************)
Begin
ClrScr;
Modo := LastMode;
Lazo:
CuadroPrincipal;
Tecla2 := ' ';
Repeat
If KeyPressed Then
  Begin
  Tecla2:= ReadKey;
  Case Tecla2 Of
  'A'..'B','a'..'b' :Begin
                     Write(Tecla2);Delay(1000);
                     End;
  Char(79)          :Begin
                     TextMode(Modo);Halt;
                     End;
  Else Write(^g);
  End;
  End;
  Until Tecla2 in ['A'..'b','a'..'b'];
(*************************************************************)
                      CODIGOS DE LINEA
(*************************************************************)

  Case Tecla2 Of
  'A','a' : Begin
            c1:=0;
            Lazo1:
            CuadroCodigos;
            Tecla := ' ';
            Repeat
            If KeyPressed Then
            Begin
            Tecla := ReadKey;
            Case Tecla Of
            'A'..'N','S','a'..'n','s' :Begin
                                       Write(Tecla);Delay(1000);
                                       End;
            Char(27)                  :Goto Lazo;
            Char(79)                  :Begin
                                       TextMode(Modo);Halt;
                                       End
            Else Write(^g);
            End;
            End;
            Until Tecla in ['A'..'N','S','a'..'n','s'];
            Lazo2:
            Case Tecla Of
            'A'..'N','a'..'n' : CuadroSeleccion;
            'S','s' : CuadroSeleccionEspectros;
            End;
            Tecla1 := ' ';
            Repeat
            If KeyPressed Then
            Begin
            Tecla1 := ReadKey;
            Case Tecla1 Of
            'A'..'B','a'..'b' : Begin
                                Write(Tecla1);Delay(1000);
                                End;
            Char(27)          : Goto Lazo1;
            Char(79)          : Begin
                                TextMode(Modo);Halt;
                                End
            Else Write(^g);
            End;
              End;
```

```
Until Tecla1 in ['A'..'B','a'..'b',Char(79)];

Case Tecla Of
'A' ,'a': Begin
          Case Tecla1 Of
          'A','a': NRZ;
          'B','b': NRZTEORIA;
          End;
          End;
'B' ,'b': Begin
          Case Tecla1 Of
          'A','a': RZ;
          'B','b': RZTEORIA;
          End;
          End;
'C' ,'c': Begin
          Case Tecla1 Of
          'A','a': NRZPOLAR;
          'B','b': NRZPOLARTEORIA;
          End;
          End;
'D' ,'d': Begin
          Case Tecla1 Of
          'A','a': RZPOLAR;
          'B','b': RZPOLARTEORIA;
          End;
          End;
'E' ,'e': Begin
          Case Tecla1 Of
          'A','a': AMI;
          'B','b': AMITEORIA;
          End;
          End;
'F' ,'f': Begin
          Case Tecla1 Of
          'A','a': HDB3;
          'B','b': HDB3TEORIA;
          End;
          End;
'G' ,'g': Begin
          Case Tecla1 Of
          'A','a': BNZS;
          'B','b': BNZSTEORIA;
          End;
          End;
'H' ,'h': Begin
          Case Tecla1 Of
          'A','a': MANCHESTER;
          'B','b': MANCHESTERTEORIA;
          End;
          End;
'I' ,'i': Begin
          Case Tecla1 Of
          'A','a': BIFASEM;
          'B','b': BIFASEMTECRIA;
          End;
          End;
'J' ,'j': Begin
          Case Tecla1 Of
          'A','a': BIFASES;
          'B','b': BIFASESTECRIA;
          End;
          End;
'K' ,'k': Begin
          Case Tecla1 Of
          'A','a': MILLER;
          'B','b': MILLERTEORIA;
          End;
          End;
'L' ,'l': Begin
```

```
                    Case Tecla1 Of
                    'A','a': CUATROB3T;
                    'B','b': C4B3TTEORIA;
                    End;
                    End;
          'M' ,'m': Begin
                    Case Tecla1 Of
                    'A','a': CMI;
                    'B','b': CMITEORIA;
                    End;
                    End;
          'N' ,'n': Begin
                    Case Tecla1 Of
                    'A','a': PST;
                    'B','b': PSTTEORIA;
                    End;
                    End;
           'S','s':Begin
                    Case Tecla1 Of
                    'A','a': DEPEQUIPROBABLES;
                    'B','b':Begin
                            DEPCODIGOS1;
                             Case Tecla5 Of
                             Char(59) :  Goto Lazo1;
                             Char(27) :  Goto Lazo2;
                             Char(79) :  Begin
                                         TextMode(Modo);Halt;
                                         End
                             End;
                            DEPCODIGOS2;
                            End;
                    End;
                    End;
          End;

                    Case Tecla5 Of
                    Char(59) : Goto Lazo1;
                    Char(27) : Goto Lazo2;
                    Char(79) : Begin
                               TextMode(Modo);Halt;
                               End
                    End;
                    End;

(*************************************************************)
                  MODULACION DIGITAL
(*************************************************************)

 'B','b' :  Begin
            c2:=0;
            Lazo3:
            CuadroModulacion;

            Tecla3 := ' ';
            Repeat
            If KeyPressed Then
            Begin
            Tecla3 := ReadKey;
            Case Tecla3 Of
           'A'..'H','a'..'h' :Begin
                              Write(Tecla3);Delay(1000);
                              End;
            Char(27)          :Goto Lazo;
            Char(79)          :Begin
                               TextMode(Modo);Halt;
                               End
            Else Write(^g);
            End;
            End;
            Until Tecla3 in ['A'..'H','a'..'h'];
```

```
          Lazo4:
          CuadroSeleccionMod(Tecla3);
          Tecla4 := ' ';
          Repeat
          If KeyPressed Then
          Begin
          Tecla4 := ReadKey;
          Case Tecla4 Of
          'A'..'C','a'..'c' : Begin
                    Write(Tecla4);Delay(1000);
                    End;
          Char(27)          : Goto Lazo3;
          Char(79)          : Begin
                              TextMode(Modo);Halt;
                              End
          Else Write(^g);
          End;
          End;
          Until Tecla4 in ['A'..'C','a'..'c'];
          Case Tecla3 Of
'A' ,'a': Begin
          Case Tecla4 Of
          'A','a': Begin
                   ASK(Tecla3,Tecla6);
                   inc(c2);
                   End;
          'B','b': ASKTEORIA(Tecla3,Tecla6);
          'C','c': ASKPOTENCIA;
          End;
          End;
'B' ,'b': Begin
          Case Tecla4 Of
          'A','a': Begin
                   FSK(Tecla3,Tecla6);
                   inc(c2);
                   End;
          'B','b': FSKTEORIA(Tecla3,Tecla6);
          'C','c': FSKPOTENCIA;
          End;
          End;
'C' ,'c': Begin
          Case Tecla4 Of
          'A','a': Begin
                   PSK(Tecla3,Tecla6);
                   inc(c2);
                   End;
          'B','b': PSKTEORIA(Tecla3,Tecla6);
          'C','c': M2PSKPOTENCIA;
          End;
          End;
'D' ,'d': Begin
          Case Tecla4 Of
          'A','a': Begin
                   M4PSK(Tecla4,Tecla6);
                   inc(c2);
                   End;
          'B','b': M4PSKTEORIA(Tecla4,Tecla6);
          'C','c': M4PSKPOTENCIA;
          End;
          End;
'E' ,'e': Begin
          Case Tecla4 Of
          'A','a': Begin
                   M8PSK(Tecla4,Tecla6);
                   inc(c2);
                   End;
          'B','b': M8PSKTEORIA(Tecla4,Tecla6);
          'C','c': M8PSKPOTENCIA;
          End;
          End;
```

```
  'F' ,'f': Begin
            Case Tecla4 Of
            'A','a': Begin
                     M16PSK(Tecla4,Tecla6);
                     inc(c2);
                     End;
            'B','b': M16PSKTEORIA(Tecla4,Tecla6);
            'C','c': M16PSKPOTENCIA;
            End;
            End;
  'G' ,'g': Begin
            Case Tecla4 Of
            'A','a': Begin
                     M4QAM(Tecla4,Tecla6);
                     inc(c2);
                     End;
            'B','b': M4QAMTEORIA(Tecla4,Tecla6);
            'C','c': M4QAMPOTENCIA;
            End;
            End;
  'H' ,'h': Begin
            Case Tecla4 Of
            'A','a': Begin
                     M16QAM(Tecla4,Tecla6);
                     inc(c2);
                     End;
            'B','b': M16QAMTEORIA(Tecla4,Tecla6);
            'C','c': M16QAMPOTENCIA;
            End;
            End;
      End;
      Case Tecla6 Of
      '2' : Goto Lazo3;
      '1' : Goto Lazo4;
      '3' : Halt
      End;
      End;
End;
End.
```

UNIDAD CODIGOS1.TPU

```
(*UNIDAD CODIGOS1 CONTIENE LOS PROCEDIMIENTOS O SUBRUTINAS
  PARA LA OPCION CODIFICAR, DE CODIGOS DE LINEA*)
Unit Codigos1;

INTERFACE

(*VARIABLES GLOBALES QUE UTILIZA LA UNIDAD*)

Uses CRT, GRAPH;
Var
  M, N, cp ,c1, ControlGr, ModoGr : Integer;
  Tecla, es : Char;
  dato,reloj, coef : Array[1..60] Of Integer;
  bit : Array[1..30] Of Char;
  coef1 : Array[1..30] Of String;

(***********************************************)
(*LISTADO DE PROCEDIMIENTOS QUE UTILIZA LA UNIDAD*)

Procedure DibujarCuadro(x1,y1,x2,y2,pp,f : integer);
Procedure PintarFondo(x1,y1,x2,y2,f: integer);
Procedure CuadroCodigos;
Procedure CuadroSeleccion;
Procedure CuadroSeleccionEspectros;
Procedure Titulo;
Procedure Opciones(x1, y1 : integer);
Procedure Graficar;
Procedure Graficar4B3T;
Procedure Continuar;
Procedure NUMEROS(x1, y1, x2, y2, B, C : Integer; var A : Integer);
Procedure Aleatorio;
Procedure LeerDatos;
Procedure IngresoDatos;
Procedure MantenerDatos;
Procedure NRZ1;
Procedure RZ1;
Procedure NRZPOLAR1;
Procedure RZPOLAR1;
Procedure AMI1;
Procedure HDB31;
Procedure MANCHESTER1;
Procedure BnZS1(P : Integer);
Procedure BIFASEM1;
Procedure BIFASES1;
Procedure MILLER1;
Procedure CUATROB3T1(P : Integer);
Procedure CMI1;
Procedure PST1(P : Integer);
Procedure NRZ;
Procedure RZ;
Procedure NRZPOLAR;
Procedure RZPOLAR;
Procedure AMI;
Procedure HDB3;
Procedure MANCHESTER;
Procedure BnZS;
Procedure BIFASEM;
Procedure BIFASES;
Procedure MILLER;
Procedure CUATROB3T;
Procedure CMI;
Procedure PST;

IMPLEMENTATION

(***********************************************)
(*DESARROLLO DE CADA UNO DE LOS PROCEDIMIENTOS QUE UTILIZA LA UNIDAD
  CODIGOS1*)
```

```
  CODIGOS1*)
(************************************************)

Procedure DibujarCuadro(x1,y1,X2,y2,pp,f : Integer);
(*DIBUJA LOS BORDES PARA LAS PANTALLAS DE PRESENTACION EN MODO TEXTO*)
Var
  i : integer;
Begin
TextColor(pp);TextBackground(f);
For i := (x1 + 1)  To (x2 - 1) Do
  Begin
    GotoXY(i,y1);Write(#205);
    GotoXY(i,y2);Write(#205);
  End;
For i := (y1 + 1) To (y2 - 1) Do
  Begin
    GotoXY(x1,i);Write(#186);
    GotoXY(x2,i);Write(#186);
  End;
GotoXY(x1,y1);Write(#201);
GotoXY(x2,y1);Write(#187);
GotoXY(x1,y2);Write(#200);
GotoXY(x2,y2);Write(#188);
End;
(************************************************)
Procedure PintarFondo(x1,y1,x2,y2,f: integer);
(*PONE EL COLOR DE FONDO EN LAS PANTALLAS DE PRESENTACION*)

Var
  i,j : integer;
Begin
ClrScr;
For i := x1 To x2 Do
  For j := y1 To y2 Do
    Begin
      GotoXY(i,j);
      Textbackground(f);
      Write(#0);
    End;
End;
(************************************************)
Procedure CuadroCodigos;
(*PANTALLA DE SELECCION QUE CONTIENE LA LISTA DE LAS CLASES DE CODIGOS
  DISPONIBLES EN EL PROGRAMA*)

Begin
ClrScr;
PintarFondo(1,1,80,24,7);
PintarFondo(8,7,73,21,3);
DibujarCuadro(1,2,80,25,1,7);
DibujarCuadro(8,6,73,21,1,3);
TextColor(14);TextBackground(1);
GotoXY(20,4);
Write(' C O D I F I C A C I O N   D I G I T A L ');
TextColor(4);TextBackground(3);
GotoXY(25,8);
Write('C O D I G O S   D E   L I N E A');
TextColor(1);
GotoXY(17,10);Write('A) CODIGO NRZ (Neutral)');
GotoXY(17,11);Write('B) CODIGO RZ (Neutral) ');
GotoXY(17,12);Write('C) CODIGO NRZ POLAR');
GotoXY(17,13);Write('D) CODIGO RZ POLAR');
GotoXY(17,14);Write('E) CODIGO AMI');
GotoXY(17,15);Write('F) CODIGO HDB3');
GotoXY(17,16);Write('G) CODIGO B3ZS');
GotoXY(17,17);Write('H) CODIGO MANCHESTER');
GotoXY(45,10);Write('I) CODIGO BIFASE M');
GotoXY(45,11);Write('J) CODIGO BIFASE S');
GotoXY(45,12);Write('K) CODIGO MILLER');
GotoXY(45,13);Write('L) CODIGO 4B3T');
```

```
GotoXY(45,14);Write('M) CODIGO CMI');
GotoXY(45,15);Write('N) CODIGO PST');
GotoXY(45,17);Write('S) ESPECTROS DE POTENCIA');
GotoXY(20,19);Write('[ESC] MENU PRINCIPAL');
GotoXY(45,19);Write('[END] SALIR AL DOS');
TextColor(1);TextBackground(7);
GotoXY(45,22);Write('Ingrese su selección...[   ] ');
GotoXY(70,22);
End;
(*********************************************************)
Procedure CuadroSeleccion;
(*PERMITE ESCOJER LAS APLICACIONES EN CODIGOS (CODIFICACION O TEORIA)*)

Var
  i, j : integer;
Begin
ClrScr;
PintarFondo(1,1,80,25,1);
PintarFondo(18,11,64,16,3);
DibujarCuadro(1,2,80,25,15,1);
DibujarCuadro(18,10,64,16,1,3);
TextColor(4);TextBackground(7);
GotoXY(25,4);WriteLn(' C O D I G O S   D E   L I N E A ');
TextColor(1);
Case Tecla Of
  'A','a' : Begin
        GotoXY(31,6);
        Write(' C O D I G O   N R Z ');
        End;
  'B' ,'b': Begin
        GotoXY(31,6);
        Write(' C O D I G O   R Z ');
        End;
  'C','c' : Begin
        GotoXY(24,6); Write(' C O D I G O   N R Z  P O L A R ');
        End;
  'D' ,'d': Begin
        GotoXY(26,6); Write(' C O D I G O  R Z  P O L A R ');
        End;
  'E' ,'e': Begin
        GotoXY(31,6); Write(' C O D I G O   A M I ');
        End;
  'F' ,'f': Begin
        GotoXY(30,6); Write(' C O D I G O   H D B 3 ');
        End;
  'G' ,'g': Begin
        GotoXY(30,6); Write(' C O D I G O   B 3 Z S ');
        End;
  'H' ,'h': Begin
        GotoXY(24,6); Write(' C O D I G O   M A N C H E S T E R ');
        End;
  'I' ,'i': Begin
        GotoXY(26,6); Write(' C O D I G O  B I F A S E  M ');
        End;
  'J' ,'j': Begin
        GotoXY(26,6); Write(' C O D I G O  B I F A S E  S ');
        End;
  'K' ,'k': Begin
        GotoXY(25,6); Write(' C O D I G O  D E  M I L L E R ');
        End;
  'L' ,'l': Begin
        GotoXY(30,6); Write(' C O D I G O  4 B 3 T ');
        End;
  'M' ,'m': Begin
        GotoXY(31,6); Write(' C O D I G O  C M I ');
        End;
 'N' ,'n': Begin
        GotoXY(31,6); Write(' C O D I G O  P S T ');
        End;
   End;
```

```
TextColor(15);TextBackground(3);
GotoXY(30,12);Write('A)  CODIFICACION');
GotoXY(30,14);Write('B)  TEORIA');
Textcolor(2);TextBackground(1);
GotoXY(20,18);Write('[Esc] : Menú Códigos    [End] : Salir al DOS');
GotoXY(35,21);Write('Ingrese su selección...[   ]');
GotoXY(60,21);
End;
(*************************************************)
Procedure CuadroSeleccionEspectros;
(*PANTALLA DE SELECCION DE LAS DOS OPCIONES PARA LOS ESPECTROS
  DE POTENCIA PARA LOS CODIGOS DE LINEA*)

Var
  i, j : integer;
Begin
ClrScr;
PintarFondo(1,1,80,25,1);
DibujarCuadro(1,2,80,25,15,1);
TextColor(4);TextBackground(7);
GotoXY(25,4);WriteLn(' C O D I G O S  D E  L I N E A ');
TextColor(0);
GotoXY(18,6);
Write(' E S P E C T R O S  D E  P O T E N C I A ');
DibujarCuadro(15,9,67,17,15,1);
TextColor(14);
GotoXY(22,11);Write('A)  COMPARACION DE ESPECTROS PARA');
GotoXY(22,12);Write('    PROBABILIDAD P = 0.5');
GotoXY(22,14);Write('B)  ESPECTROS DE POTENCIA DE UN CODIGO');
GotoXY(22,15);Write('    PARA 2 PROBABILIDADES DIFERENTES');
GotoXY(20,18);Write('[Esc] : Menú Códigos    [End] : Salir al DOS');
Textcolor(15);
GotoXY(35,21);Write('Ingrese su selección...[   ]');
GotoXY(60,21);
End;
(*************************************************)
Procedure Titulo;
(*COLOCA EL TITULO EN LA PANTALLA DE INGRESO DE DATOS DE ACUERDO
  CON EL CODIGO SELECCIONADO*)

Var
  i : integer;
Begin
ClrScr;
DibujarCuadro(1,2,80,25,14,1);
GotoXY(1,22);Write(#204);GotoXY(80,22);Write(#185);
For i:=2 To 79 Do
   Begin
     GotoXY(i,22);Write(#205);
   End;
TextColor(0);TextBackground(15);
Case Tecla Of
  'A','a' : Begin
        GotoXY(31,4);
        Write(' C O D I G O  N R Z ');
        End;
  'B' ,'b': Begin
        GotoXY(31,4);
        Write(' C O D I G O  R Z ');
        End;
  'C','c' : Begin
        GotoXY(24,4);
        Write(' C O D I G O  N R Z  P O L A R ');
        End;
  'D' ,'d': Begin
        GotoXY(26,4);
        Write(' C O D I G O  R Z  P O L A R ');
        End;
  'E' ,'e': Begin
        GotoXY(31,4);
```

```
        Write(' C O D I G O    A M I ');
        End;
  'F' ,'f': Begin
        GotoXY(30,4);
        Write(' C O D I G O    H D B 3 ');
        End;
  'G' ,'g': Begin
        GotoXY(30,4);
        Write(' C O D I G O    B 3 Z S ');
        End;
  'H' ,'h': Begin
        GotoXY(24,4);
        Write(' C O D I G O    M A N C H E S T E R ');
        End;
  'I' ,'i': Begin
        GotoXY(26,4);
        Write(' C O D I G O   B I F A S E   M ');
        End;
  'J' ,'j': Begin
        GotoXY(26,4);
        Write(' C O D I G O   B I F A S E   S ');
        End;
  'K' ,'k': Begin
        GotoXY(25,4);
        Write(' C O D I G O   D E   M I L L E R ');
        End;
  'L' ,'l': Begin
        GotoXY(30,4);
        Write(' C O D I G O   4 B 3 T ');
        End;
  'M' ,'m': Begin
        GotoXY(31,4);
        Write(' C O D I G O   C M I ');
        End;
 'N' ,'n': Begin
        GotoXY(31,4);
        Write(' C O D I G O   P S T ');
        End;
   End;
End;
(************************************************)
Procedure Opciones(x1, y1 : integer);
(*CONTINUA LA EJECUCION DEL PROGRAMA , CAMBIAR DE APLICACIION
  O TERMINAR, AAAL PULSAR LAS TECLAS ESC, ENTER, F1, END*)

Begin
  SetColor(14);
  MoveTo(0,y1-30);LineTo(x1,y1-30);
  SetTextStyle(2,0,5);SetTextJustify(1,2);
  OutTextXY(X1 Div 2,Y1-20,' ESC : Menú Anterior  F1 : Menú Códigos  END :
Salir al DOS ');
  es:=' ';inc(c1);
  Repeat
  If KeyPressed Then
  es := ReadKey;
  Until es in [Char(27),Char(59),Char(79)];
End;
(************************************************)
Procedure Graficar;
(GRAFICA LAS TRES SEñALES: SEÑAL DE RELOJ, DE DATOS Y CODIFICADA*)

Var
  h, i, j, k, X, Y, X1, X2, Y1, Y2 : Integer;
  numero : Array[1..25] of String;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1:=GetMaxX; Y1:=GetMaxY;
  Rectangle(0,0,X1,Y1);Rectangle(2,2,X1-2,Y1-2);
  SetBkColor(1);SetColor(14);
```

```
SetTextStyle(1,0,1);SetTextJustify(1,2);
Case Tecla Of
'A','a' : OutTextXY(X1 Div 2,20,'CODIGO LINEAL NRZ');
'B','b' : OutTextXY(X1 Div 2,20,'CODIGO LINEAL RZ');
'C','c' : OutTextXY(X1 Div 2,20,'CODIGO LINEAL NRZ POLAR');
'D','d' : OutTextXY(X1 Div 2,20,'CODIGO LINEAL RZ POLAR');
'E','e' : OutTextXY(X1 Div 2,20,'CODIGO LINEAL AMI');
'F','f' : OutTextXY(X1 Div 2,20,'CODIGO LINEAL HDB3');
'G','g' : OutTextXY(X1 Div 2,20,'CODIGO LINEAL B3ZS');
'H','h' : OutTextXY(X1 Div 2,20,'CODIGO LINEAL MANCHESTER');
'I','i' : OutTextXY(X1 Div 2,20,'CODIGO LINEAL BIFASE M');
'J','j' : OutTextXY(X1 Div 2,20,'CODIGO LINEAL BIFASE S');
'K','k' : OutTextXY(X1 Div 2,20,'CODIGO LINEAL DE MILLER');
'M','m' : OutTextXY(X1 Div 2,20,'CODIGO LINEAL CMI');
'N','n' : OutTextXY(X1 Div 2,20,'CODIGO LINEAL PST');
End;
If (N <= 10) Then j :=20
Else If (N > 10) And (N <= 16)  Then j := 15
Else If (N >=17) And (N <= 23)  Then j := 12;
h := 2 * j;
If cp = 1 Then k:=h  Else k := j;
X2 := (X1 - (N*h)) Div 2; Y2 := Y1 Div 4;
MoveTo(X2,Y2);SetColor(12);
For i:=1 to 2*N do
  begin
    lineto(X2+j*(i-1),Y2-20*reloj[i]);
    lineto(X2+j*i,Y2-20*reloj[i]);
  end;
MoveTo(X2,2*Y2);SetColor(13);
For i:=1 to N do
  begin
    lineto(X2+h*(i-1),(2*Y2)-20*dato[i]);
    lineto(X2+h*i,(2*Y2)-20*dato[i]);
  end;
MoveTo(X2,3*Y2);SetColor(14);
For i:=1 to M do
  begin
    lineto(X2+k*(i-1),(3*Y2)-20*coef[i]);
    lineto(X2+k*i,(3*Y2)-20*coef[i]);
  end;
SetColor(15);
MoveTo(X2,2*Y2-3);
For i:= 1 To N+1 Do
  Begin
    LineRel(0,3); X := GetX;
    X := X + h; Y := 2*Y2-3;
    MoveTo(X,Y);
  End;
MoveTo(X2,3*Y2-3);
For i:= 1 To N+1 Do
  Begin
    LineRel(0,3); X := GetX;
    X := X + h; Y := 3*Y2-3;
    MoveTo(X,Y);
  End;
For i:= 1 To N Do
  numero[i] := bit[i];
SetTextStyle(2,0,4);
SetTextJustify(0,2);
MoveTo(X2+(h Div 2),2*Y2+12);
For i:= 1 To N Do
  Begin
    X:=GetX; Y:=GetY;
    OutTextXY(X,Y,numero[i]);
    X := X + h;
    MoveTo(X,Y);
  End;
MoveTo(X2+(h Div 2),3*Y2+32);
For i:= 1 To N Do
  Begin
```

```
      X:=GetX; Y:=GetY;
      OutTextXY(X,Y,numero[i]);
      X := X + h;
      MoveTo(X,Y);
    End;
  OutTextXY(X2+5,Y2-45,'SEÑAL DE RELOJ');
  OutTextXY(X2+5,2*Y2-45,'SEÑAL BINARIA');
  OutTextXY(X2+5,3*Y2-45,'SEÑAL CODIFICADA');
  OutTextXY(X2-30,2*Y2-25,'+ V');OutTextXY(X2-30,2*Y2,'0');
  OutTextXY(X2-30,3*Y2-25,'+ A');OutTextXY(X2-30,3*Y2,'0');
  If (Tecla <> 'A') and (Tecla <> 'a') and
    (Tecla <> 'B') and (Tecla <> 'b') Then
    OutTextXY(X2-30,3*Y2+25,'- A');
  Opciones(X1, Y1);
  CloseGraph;
End;
(************************************************)
Procedure Graficar4B3T;
(*GRAFICA LAS TRES SEñALES: SEÑAL DE RELOJ, DE DATOS Y CODIFICADA
  UNICAMENTE PARA EL CODIGO 4B3T)

Var
  h, i, j, k, X, Y, X1, X2, Y1, Y2 : Integer;
  numero : Array[1..25] of String;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1:=GetMaxX; Y1:=GetMaxY;
  Rectangle(0,0,X1,Y1);
  SetBkColor(1);SetColor(14);
  SetTextStyle(1,0,1);SetTextJustify(1,2);
  OutTextXY(X1 Div 2,20,'CODIGO LINEAL 4B3T');
  SetTextJustify(0,2);
  If (N <= 10) Then j :=20
  Else If (N > 10) And (N <= 16)  Then j := 15
  Else If (N >= 17) And (N <= 23)  Then j := 12;
  h := 2 * j; k := (8*j) Div 3;
  Y2:=Y1 Div 4;X2:=(X1-(N*h)) Div 2;
  SetColor(12);MoveTo(X2,Y2);
  For i:=1 to 2*N do
    begin
      lineto(X2+j*(i-1),Y2-20*reloj[i]);
      lineto(X2+j*i,Y2-20*reloj[i]);
    end;
  SetColor(13);MoveTo(X2,2*Y2);
  For i:=1 to N do
    begin
      lineto(X2+h*(i-1),(2*Y2)-20*dato[i]);
      lineto(X2+h*i,(2*Y2)-20*dato[i]);
    end;
  SetColor(14);MoveTo(X2,3*Y2);
  For i:=1 to (N Div 4)*3 do
    begin
      lineto(X2+k*(i-1),(3*Y2)-20*coef[i]);
      lineto(X2+k*i,(3*Y2)-20*coef[i]);
    end;
  MoveTo(X2,2*Y2-3);
  For i:= 1 To N+1 Do
    Begin
      LineRel(0,3); X := GetX;X := X + h; Y := 2*Y2-3;
      MoveTo(X,Y);
    End;
  MoveTo(X2,2*Y2+12);
  For i:= 1 To (M Div 3)+1  Do
    Begin
      LineRel(0,10); X := GetX; X := X + 4*h; Y := 2*Y2+12;
      MoveTo(X,Y);
    End;
  MoveTo(X2,3*Y2-3);
  For i:= 1 To M+1 Do
```

```
    Begin
      LineRel(0,3); X := GetX; X := X + k; Y := 3*Y2-3;
      MoveTo(X,Y);
    End;
  MoveTo(X2,3*Y2+32);
  For i:= 1 To (M Div 3)+1 Do
    Begin
      LineRel(0,10); X := GetX;X := X + 4*h; Y := 3*Y2+32;
      MoveTo(X,Y);
    End;
  For i:= 1 To N Do
    numero[i] := bit[i];
  SetTextStyle(2,0,4);SetTextJustify(0,2);
  MoveTo(X2+(h Div 2),2*Y2+12);
  For i:= 1 To N Do
    Begin
      X:=GetX; Y:=GetY;
      OutTextXY(X,Y,numero[i]);X := X + h;
      MoveTo(X,Y);
    End;
  MoveTo(X2+(k Div 2),3*Y2+32);
  For i:= 1 To ((N Div 4)*3) Do
    Begin
      X:=GetX; Y:=GetY;
      OutTextXY(X,Y,coef1[i]);X := X + k;
      MoveTo(X,Y);
    End;
  OutTextXY(X2+5,Y2-45,'SEÑAL DE RELOJ');
  OutTextXY(X2+5,2*Y2-45,'SEÑAL BINARIA');
  OutTextXY(X2+5,3*Y2-45,'SEÑAL CODIFICADA');
  OutTextXY(X2-30,2*Y2-25,'+ V');OutTextXY(X2-30,2*Y2,'0');
  OutTextXY(X2-30,3*Y2-25,'+ A');OutTextXY(X2-30,3*Y2,'0');
  OutTextXY(X2-30,3*Y2+25,'- A');
  Opciones(X1, Y1);
  CloseGraph;
End;

(************************************************)
Procedure Continuar;
(*PERMITE LA EJECUCION DEL PROGRAMA UNICAMENTE PULSANDO LA TECLA ENTER*)

Var
  s : Char;
Begin
  s := ' ';
  Repeat
    If KeyPressed Then
      s := ReadKey;
  Until s = Char(13);
End;
(************************************************)
Procedure NUMEROS(x1, y1, x2, y2, B, C : Integer; var A : Integer);
(*LEE EL INGRESO DEL NUMERO N DE BITS Y DA UN MENSAJE DE ERROR EN CASO DE
  NO SER EL DATO CORRECTO*)

Var
  i, j , codigo : integer;
  NU : Array [1..10] of char;
  numero1 : String[10];
Label
  Repetir;

Begin
REPETIR:
TextColor(14);
GotoXY(x1,y1);
Write('                    ');
GotoXY(x1,y1);
i := 1;
Repeat
```

```
If KeyPressed Then
  Begin
  NU[i] := ReadKey;
  Write(NU[i]);
  inc(i,1);
  End;
Until ( NU[i-1] = Char(13)) Or (i = 10);
Numero1:=NU[1];
For j:=2 To i-2 Do
Numero1:=Numero1 + NU[j];
Val(Numero1,A,codigo);
If (codigo <> 0) Or ((A > B) or (A < C)) Then
  Begin
  TextColor(139);
  GotoXY(x2,y2);Write('Dato incorrecto Presione ENTER para continuar');
  Continuar;
  GotoXY(x2,y2);Write('                                                    ');
  Goto Repetir;
  End;
End;
(************************************************)
Procedure Aleatorio;
(*GENERA UNA SECUENCIA DE BITS ALEATORIOS PARA SER CODIFICADOS*)

Var
  i, x, N1 : Integer;
Begin
GotoXY(3,23);
Write('   Ingresar el Numero de bits a generar [4 ≤ n ≤ 20] y luego pulsar
ENTER   ');
GotoXY(6,8);Write('Número de bits que desea generar : ');
GotoXY(41,8);
NUMEROS(41, 8, 20, 10, 20, 4, N1);
N := N1;
Randomize;
For i := 1 To N Do
  Begin
  x := Random(100);
  If (x <= 50) Then
    Begin
      bit[i] := '0';dato[i] := 0;
    End
  Else
    Begin
      bit[i] := '1';dato[i] := 1
    End
  End;
GotoXY(6,10);Write('Los bits a codificar son : ');
DibujarCuadro(6,12,73,16,15,1);
TextColor(14);GotoXY(3,23);
Write('                    Generando secuencia de bits a codificar
       ');
GotoXY(8,14);
For i := 1 To N Do
  Begin
    Write(bit[i]:3);
    Delay(500);
  End;
End;
(************************************************)
Procedure LEERDATOS;
(*PANTALLA PARA EL INGRESO DE DATOS MANUALES DESDE EL TECLADO*)

Var
  i, N1 : Integer;
  s : Char;
Begin
GotoXY(3,23);
Write('  Ingresar el Numero de bits a codificar [4 ≤ n ≤ 20] y luego pulsar
ENTER  ');
```

```
TextColor(14);TextBackground(1);
GotoXY(6,8);Write('Numero de bits : ');
GotoXY(23,8);
NUMEROS(23, 8, 20, 10, 20, 4, N1);
N := N1;
GotoXY(6,10);Write('Ingrese los datos a codificar : ');
DibujarCuadro(6,12,73,16,15,1);
TextColor(14);
GotoXY(3,23);
Write('   Ingrese la secuencia de bits (ceros y unos) a codificar,
secuencialmente  ');
GotoXY(10,14);
i := 1; s := ' ';
Repeat
  If KeyPressed Then
    Begin
      s := ReadKey;
      Case s Of
      '0' : Begin
            Write('  ',s);bit[i] := '0';Inc(i,1);
            End;
      '1' : Begin
            Write('  ',s);bit[i] := '1';Inc(i,1);
            End;
      Else Write(^g);
      End;
    End;
  Until i= N+1;
GotoXY(3,23);
Write('                        Presione ENTER para continuar
       ');
Continuar;
For i:=1 To N Do
  If bit[i] = '0' Then dato[i] := 0
  Else dato[i] := 1;
End;
(************************************************)
Procedure INGRESODATOS;
(*PANTALLA DEL INGRESO DE LOS DATOS, MANUAL O POR GENERACION ALEATORIA DE
  BITS*)

Var
  i : Integer;
  s : Char;

Begin
TextColor(14);TextBackground(1);
GotoXY(6,6);Write('Desea Ingresar (I) o Generar (G) los bits : ');
GotoXY(3,23);
Write('  Teclear I para ingresar datos manualmente o G para generar
automaticamente ');
GotoXY(50,6);
s := ' ';
Repeat
  If KeyPressed Then
     Begin
     s := ReadKey;
     Case s Of
     'G','g' : Begin
               Write(s);Delay(1000);
               End;
     'I','i' : Begin
               Write(s);Delay(1000);
               End;
     Else Write(^g);
     End;
     End;
Until s in ['I','G','i','g'];

Case s Of
```

```
'G','g' : Aleatorio;
'I','i' : LeerDatos;
End;
i := 1;
While i <= 2*N Do
  Begin
  reloj[i] := 1;reloj[i+1] := 0;
  inc(i,2);
  End;
End;
(*************************************************************)
Procedure MANTENERDATOS;
(*GUARDA LOS DATOS ANTERIORES PARA UTILIZARLOS EN LA CODIFICACION*)

Var
  i, j : Integer;
  s : Char;
Begin
TextColor(14);TextBackground(1);
GotoXY(6,7);Write('Los bits a codificar son : ');
DibujarCuadro(6,10,73,14,14,1);
GotoXY(8,12);
For i := 1 To N Do
  Write(bit[i]:3);
GoToXY(6,17);Write('¿ Desea cambiar la secuencia de bits Si (S) o No (N)
?');
GotoXY(3,23);
Write('Teclear S si desea cambiar la secuencia de bits anterior o N para
mantenerla');
GotoXY(62,17);
s := ' ';
Repeat
  If KeyPressed Then
     Begin
     s := ReadKey;
     Case s Of
     'S','s' : Begin
               Write(s);Delay(1000);
               End;
     'N','n' : Begin
               Write(s);Delay(1000);
               End;
     Else Write(^g);
     End;
     End;
Until s in ['N','n','S','s'];
Case s Of
'S','s' : Begin
          For j := 5 To 20  Do
            For i := 5 To 73 Do
              Begin
              GOTOXY(i,j);Write(' ');
              End;
          IngresoDatos;
          End;
End;
End;
(*************************************************************)
Procedure NRZ1;
(*PERMITEN ENCONTRAR LOS COEFICIENTES PARA GRAFICAR LA SEÑAL
  CODIFICADA NRZ*)

Var
  i : Integer;
Begin
M := N;cp := 1;
For i := 1 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
```

```
    coef [i] := 0;
    End
  Else
    Begin
    coef[i] := 1;
    End;
  End;
End;
(************************************************)
Procedure RZ1;
(*PERMITEN ENCONTRAR LOS COEFICIENTES PARA GRAFICAR LA SEÑAL
  CODIFICADA RZ*)

Var
  i, j : Integer;
Begin
M := N*2;cp:=2;
j := 1;
For i := 1 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    coef [j] := 0;
    coef [j+1] := 0;
    j := j + 2;
    End
  Else
    Begin
    coef[j] := +1;
    coef[j+1] := 0;
    j := j + 2;
    End;
  End;
End;
(************************************************)
Procedure NRZPOLAR1;
(*PERMITEN ENCONTRAR LOS COEFICIENTES PARA GRAFICAR LA SEÑAL
  CODIFICADA NRZPOLAR*)

Var
  i : Integer;
Begin
M := N;cp := 1;
For i := 1 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    coef [i] := -1;
    End
  Else
    Begin
    coef[i] := 1;
    End;
  End;
End;
(************************************************)
Procedure RZPOLAR1;
(*PERMITEN ENCONTRAR LOS COEFICIENTES PARA GRAFICAR LA SEÑAL
  CODIFICADA RZPOLAR*)

Var
  i, j : Integer;
Begin
M := N*2;cp:=2;
j := 1;
For i := 1 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    coef [j] := -1;
```

```
    coef [j+1] := 0;
    j := j + 2;
    End
  Else
    Begin
    coef[j] := +1;
    coef[j+1] := 0;
    j := j + 2;
    End;
  End;
End;
(************************************************)
Procedure AMI1;
(*PERMITEN ENCONTRAR LOS COEFICIENTES PARA GRAFICAR LA SEÑAL
  CODIFICADA AMI*)

Var
  i, COF : Integer;
Begin
COF := -1;
M := N;cp := 1;
For i := 1 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    coef [i] := 0;
    End
  Else
    Begin
    COF := (COF * -1);
    coef[i] := COF;
    End;
  End;
End;
(************************************************)
Procedure HDB31;
(*PERMITEN ENCONTRAR LOS COEFICIENTES PARA GRAFICAR LA SEÑAL
  CODIFICADA HDB3*)

Var
  i, k, S, COF : Integer;
Begin
COF := -1;cp := 1;
M := N; S := 0;
k := 1;
If (bit[1] = '0') And (bit[2] = '0')
   And (bit[3]='0') And (bit[4] = '0') Then
  Begin
  coef[1] := 0; coef[2] := 0;
  coef[3] := 0; coef[4] := 1;
  i := 5; S:= 1;
  K:=2;
  End
Else
  i :=1;

While (i <= N) Do
  Begin
  If (bit[i] = '0') And (bit[i+1] = '0')
     And (bit[i+2] = '0') And (bit[i+3] = '0') Then
    Begin
    If ( K MOD 2 = 0)   Then
      Begin
      If (coef[i-1] > 0) Then
        Begin
        coef[i] := -1; coef[i+1] := 0;
        coef[i+2] := 0; coef[i+3] := -1;
        COF := -1;
        S := S + 1;
        i := i + 4;
```

```
        k := 0;
        End
      Else
        Begin
        coef[i] := 1; coef[i+1] := 0;
        coef[i+2] := 0; coef[i+3] := 1;
        COF := 1;
        S := S + 1;
        i := i + 4;
        k := 0;
        End
      End
    Else
      Begin
      If (coef[i-1] > 0) Then
        Begin
        coef[i] := 0; coef[i+1] := 0;
        coef[i+2] := 0; coef[i+3] := 1;
        COF := 1;
        S := S + 1;
        i := i + 4;
        k := 0;
        End
      Else
        Begin
        coef[i] := 0; coef[i+1] := 0;
        coef[i+2] := 0; coef[i+3] := -1;
        COF := -1;
        S := S + 1;
        i := i + 4;
        k := 0;
        End
      End
    End
  Else
    Begin
    If (bit[i] = '0') Then
      Begin
      coef[i] := 0;
      i := i + 1;
      End
    Else
      Begin
      COF := COF * (-1);
      coef[i] := COF;
      inc(i,1);
      If (S <> 0) Then
        Begin
        k := k+1;
        End
      End
    End
  End;
End;
(*********************************************)
Procedure BnZS1(P : Integer);
(*PERMITEN ENCONTRAR LOS COEFICIENTES PARA GRAFICAR LA SEÑAL
  CODIFICADA B3ZS*)

Var
  i, S, COF : Integer;
Begin
M := N; cp := 1;
COF := -1;
S := 0;
If (bit[1] = '0') And (bit[2] = '0')
   And (bit[3] = '0') Then
  Begin
  coef[1] := 1;
  coef[2] := 0;
```

```
  coef[3] := 1; COF := 1;
  i := 4;
  S := 1;
  P := 0;
  End
Else
  i := 1;
While (i <= N) Do
  Begin
  If (bit[i] = '0') And (bit[i+1] = '0')
     And (bit[i+2] = '0') Then
    Begin
    If ( P MOD 2 = 0 )  Then
      Begin
      If (coef[i-1] > 0) Then
        Begin
        coef[i] := -1;
        coef[i+1] := 0;
        coef[i+2] := -1; COF := -1;
        S := S + 1;
        i := i + 3;
        P := 0;
        End
      Else
        Begin
        coef[i] := 1;
        coef[i+1] := 0;
        coef[i+2] := 1; COF := 1;
        S := S + 1;
        i := i + 3;
        P := 0;
        End
      End
    Else
      Begin
      If (coef[i-1] > 0) Then
        Begin
        coef[i] := 0;
        coef[i+1] := 0;
        coef[i+2]  := 1; COF := 1;
        S := S + 1;
        i := i + 3;
        P := 0;
        End
      Else
        Begin
        coef[i] := 0;
        coef[i+1] := 0;
        coef[i+2] := -1; COF := -1;
        S := S + 1;
        i := i + 3;
        P := 0;
        End
      End
    End
  Else
    Begin
    If (bit[i] = '0') Then
      Begin
      coef[i] := 0;
      inc(i,1);
      End
    Else
      Begin
      COF := COF*(-1);
      coef[i] := COF;
      inc(i,1);
      If (S <> 0) Then
       Begin
        inc(P,1);
```

```
        End;
       End
     End
  End;
End;
(***********************************************)
Procedure MANCHESTER1;
(*PERMITEN ENCONTRAR LOS COEFICIENTES PARA GRAFICAR LA SEÑAL
  CODIFICADA MANCHESTER*)

Var
  i, j : Integer;
Begin
M := N*2;cp := 2;
j := 1;
For i := 1 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    coef [j] := -1;
    coef [j+1] := +1;
    j := j + 2;
    End
  Else
    Begin
    coef[j] := +1;
    coef[j+1] := -1;
    j := j + 2;
    End;
  End;
End;
(***********************************************)
Procedure BIFASEM1;
(*PERMITEN ENCONTRAR LOS COEFICIENTES PARA GRAFICAR LA SEÑAL
  CODIFICADA BIFASEM*)
Var
  i, j : Integer;
Begin
M := N*2; cp :=2;
If (bit[1] = '0') then
  Begin
  coef[1] := -1;
  coef[2] := -1;
  End
Else
  Begin
  coef[1] := 1;
  coef[2] := -1;
  End;

j := 3;
For i := 2 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    coef [j] := coef[j-1] * (-1);
    coef [j+1] := coef [j-1] * (-1);
    j := j + 2;
    End
  Else
    Begin
    coef[j] := coef[j-1] * (-1);
    coef[j+1] := coef[j-1];
    j := j + 2;
    End;
  End;
End;
(***********************************************)
Procedure BIFASES1;
(*PERMITEN ENCONTRAR LOS COEFICIENTES PARA GRAFICAR LA SEÑAL
```

```
  CODIFICADA BIFASE S*)

Var
  i, j : Integer;
Begin
M := N*2;cp := 2;
If (bit[1] = '1') then
  Begin
  coef[1] := -1;
  coef[2] := -1;
  End
Else
  Begin
  coef[1] := -1;
  coef[2] := 1;
  End;

j := 3;
For i := 2 To N Do
  Begin
  If (bit[i] = '1') Then
    Begin
    coef [j] := coef[j-1] * (-1);
    coef [j+1] := coef [j-1] * -1;
    j := j + 2;
    End
  Else
    Begin
    coef[j] := coef[j-1] * (-1);
    coef[j+1] := coef[j-1];
    j := j + 2;
    End;
  End;
End;
(************************************************)
Procedure MILLER1;
(*PERMITEN ENCONTRAR LOS COEFICIENTES PARA GRAFICAR LA SEÑAL
  CODIFICADA MILLER*)

Var
  i, j : Integer;
Begin
M := N*2;cp:=2;
If (bit[1] = '0') then
  Begin
  coef[1] := 1;
  coef[2] := 1;
  End
Else
  Begin
  coef[1] := 1;
  coef[2] := -1;
  End;

j := 3;
For i := 2 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    If (bit[i-1] = '0') Then
      Begin
      coef [j] := coef[j-1] * (-1);
      coef [j+1] := coef [j-1] * -1;
      j := j + 2;
      End
    Else
      Begin
      coef[j] := coef[j-1];
      coef[j+1] := coef[j-1];
      j := j + 2;
```

```
      End
    End
  Else
    Begin
    coef[j] := coef[j-1];
    coef[j+1] := coef[j-1] * -1;
    j := j + 2;
    End
  End;
End;
(************************************************)
Procedure CUATROB3T1(P : Integer);
(*PERMITEN ENCONTRAR LOS COEFICIENTES PARA GRAFICAR LA SEÑAL
  CODIFICADA 4B3T*)

Type
  MaxCad = String;
Var
  i, j, S, MM : Integer;
  ParidadInicial, ParidadAcum : Integer;
  alfa : Array[1..20]Of Char;
  ivan : Array[1..10]Of String;
  ex : MaxCad;
Procedure ParidadAcumulada;
Var
  word : Array[1..10] Of String;
  Begin
  word[j] := bit[i] + bit[i+1] + bit[i+2] + bit[i+3];
  If word[j] = '0000' Then
    alfa[j] := 'A'
  Else If word[j] = '0001' Then
    alfa[j] := 'B'
  Else If word[j] = '0010' Then
    alfa[j] := 'C'
  Else If word[j] = '1000' Then
    alfa[j] := 'D'
  Else If word[j] = '1001' Then
    alfa[j] := 'E'
  Else If word[j] = '1010' Then
    alfa[j] := 'F'
  Else If word[j] = '0011' Then
    alfa[j] := 'G'
  Else If word[j] = '1011' Then
    alfa[j] := 'H'
  Else If word[j] = '0101' Then
    alfa[j] := 'L'
  Else If word[j] = '0110' Then
    alfa[j] := 'R'
  Else If word[j] = '0111' Then
    alfa[j] := 'S'
  Else If word[j] = '1110' Then
    alfa[j] := 'T'
  Else If word[j] = '1100' Then
    alfa[j] := 'U'
  Else If word[j] = '1101' Then
    alfa[j] := 'V'
  Else If word[j] = '0100' Then
    alfa[j] := 'W'
  Else If word[j] = '1111' Then
    alfa[j] := 'X';

  case alfa[j] Of
  'A'..'F' : ParidadAcum := 0;
  'G'..'T' : ParidadAcum := 1;
  'U'..'W' : ParidadAcum := 2;
  'X' : ParidadAcum := 3;
  End;
End;
(*****************************************************)
Procedure ModoCero;
```

```
  Begin
  Case alfa[j] Of
  'A' : ivan[j] := '0 - +';
  'B' : ivan[j] := '- + 0';
  'C' : ivan[j] := '- 0 +';
  'D' : ivan[j] := '0 + -';
  'E' : ivan[j] := '+ - 0';
  'F' : ivan[j] := '+ 0 -';
  End;
End;

(*****************************************************)
Procedure ModoPositivo;
  Begin
  Case alfa[j] Of
  'G' : ivan[j] := '+ - +';
  'H' : ivan[j] := '+ 0 0';
  'L' : ivan[j] := '0 + 0';
  'R' : ivan[j] := '0 0 +';
  'S' : ivan[j] := '- + +';
  'T' : ivan[j] := '+ + -';
  'U' : ivan[j] := '+ 0 +';
  'V' : ivan[j] := '+ + 0';
  'W' : ivan[j] := '0 + +';
  'X' : ivan[j] := '+ + +';
  End;
End;
(*****************************************************)
Procedure ModoNegativo;
  Begin
  Case alfa[j] Of
  'G' : ivan[j] := '- + -';
  'H' : ivan[j] := '- 0 0';
  'L' : ivan[j] := '0 - 0';
  'R' : ivan[j] := '0 0 -';
  'S' : ivan[j] := '+ - -';
  'T' : ivan[j] := '- - +';
  'U' : ivan[j] := '- 0 -';
  'V' : ivan[j] := '- - 0';
  'W' : ivan[j] := '0 - -';
  'X' : ivan[j] := '- - -';
  End;
End;
(*****************************************************)
Procedure Uno;
  Begin
  If (ParidadAcum = -1) Then
    ParidadInicial := -1
  Else If (ParidadAcum = -2) Then
    ParidadInicial := -2
  Else If (ParidadAcum = -3) Then
    ParidadInicial := -3
  End;
(*****************************************************)
Procedure Dos;
  Begin
  If (ParidadAcum = -1) Then
    ParidadInicial := 1
  Else If (ParidadAcum = -2) Then
    ParidadInicial := -1
  Else If (ParidadAcum = -3) Then
    ParidadInicial := -2
  End;
(*****************************************************)
Procedure Tres;
  Begin
  If (ParidadAcum = -1) Then
    ParidadInicial := 2
  Else If (ParidadAcum = -2) Then
    ParidadInicial := 1
```

```
  Else If (ParidadAcum = -3) Then
    ParidadInicial := -1
  End;
(*******************************************************)
Procedure MenosUno;
  Begin
  If (ParidadAcum = 1) Then
    ParidadInicial := 1
  Else If (ParidadAcum = 2) Then
    ParidadInicial := 2
  Else If (ParidadAcum = 3) Then
    ParidadInicial := 3
  End;
(*******************************************************)
Procedure MenosDos;
  Begin
  If (ParidadAcum = 1) Then
    ParidadInicial := -1
  Else If (ParidadAcum = 2) Then
    ParidadInicial := 1
  Else If (ParidadAcum = 3) Then
    ParidadInicial := 2
  End;
(*******************************************************)
Procedure MenosTres;
  Begin
  If (ParidadAcum = 1) Then
    ParidadInicial := -2
  Else If (ParidadAcum = 2) Then
    ParidadInicial := -1
  Else If (ParidadAcum = 3) Then
    ParidadInicial := 1
  End;
(*******************************************************)
Procedure CambioEstado;
  Begin
  If (ParidadInicial = 1) Then
    Uno
  Else If (ParidadInicial = 2) Then
    Dos
  Else If (ParidadInicial = 3) Then
    Tres
  Else If (ParidadInicial = -1) Then
    MenosUno
  Else If (ParidadInicial = -2) Then
    MenosDos
  Else If (ParidadInicial = -3) Then
    MenosTres
  End;
(*******************************************************)
Begin
S := N div 4;
j := 1;
i := 1;
While j <= (N div 4) Do
  Begin
  ParidadAcumulada;
  If (ParidadAcum = 0) Then
    Begin
    ModoCero;
    j := j + 1;
    i := i + 4;
    End
  Else
    Begin
    If (ParidadInicial > 0) Then
      Begin
      ModoNegativo;
      CambioEstado;
      j := j + 1;
```

```
      i := i + 4;
      End
    Else
      Begin
      ParidadAcum := ParidadAcum * -1;
      ModoPositivo;
      CambioEstado;
      j := j + 1;
      i := i + 4;

      End
    End
  End;
ex := ivan[1] ;
For i := 2 To S Do
   ex := ex + ivan[i];

MM := Length(ex);
i := 1; j := 1;
While j <= MM Do
  Begin
    If ex[j] = '+' Then
      Begin
      coef[i] := 1;coef1[i] := '+';
      inc(i,1);
      End
    Else If ex[j] = '0' Then
      Begin
      coef[i] := 0;coef1[i] := '0';
      inc(i,1);
      End
    Else If ex[j] = '-' Then
      Begin
      coef[i] := -1;coef1[i] := '-';
      inc(i,1);
      End;
    inc(j,1);
    End;

M := MM div 2;cp:=1;
End;
(************************************************)
Procedure CMI1;
Var
  i, j, COF : Integer;
Begin
M := N*2;cp := 2;
COF := -1;
j := 1;
For i := 1 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    coef [j] := -1;
    coef [j+1] := +1;
    j := j + 2;
    End
  Else
    Begin
    COF := (COF * -1);
    coef[j] := COF;
    coef[j+1] := cof;
    j := j + 2;
    End;
  End;
End;
(************************************************)
Procedure PST1(P : Integer);
(*PERMITEN ENCONTRAR LOS COEFICIENTES PARA GRAFICAR LA SEÑAL
  CODIFICADA PST*)
```

```
Var
  i, COF : Integer;
Begin
M := N;cp := 1;
If (P = 0) Then
  Begin
  COF := -1;
  i:= 1;
  End
Else
  Begin
  COF := 1;
  i := 1;
  End;

While (i <= N) Do
  Begin
  If (bit[i] = '0') Then
    Begin
    If (bit[i+1] = '0') Then
      Begin
      coef[i] := -1;
      coef[i+1] := 1;
      i := i + 2;
      End
    Else
      Begin
      COF := COF * (-1);
      coef[i] := 0;
      coef[i+1] := COF;
      i := i + 2;
      End
    End
  Else
    Begin
    If (bit[i+1] = '0') Then
      Begin
      COF := COF * (-1);
      coef[i] := COF;
      coef[i+1] := 0;
      i := i +2;
      End
    Else
      Begin
      coef[i] := 1;
      coef[i+1] := -1;
      i := i + 2;
      End
    End
  End;
End;
(*******************************************************)
Procedure NRZ;
(*PERMITEN ENCONTRAR LOS COEFICIENTES PARA GRAFICAR LA SEÑAL
  CODIFICADA NRZ*)
Var
  i : Integer;
Begin
If c1 = 0 Then
  Begin
  Titulo; IngresoDatos;
  End
Else
  Begin
  Titulo; MantenerDatos;
  End;
NRZ1;
GotoXY(3,23);
Write('       Presione ENTER para graficar los bits y su codificación
respectiva     ');
```

```
Continuar;
Graficar;
End;
(***********************************************)
Procedure RZ;
(*REALIZA LAS LLAMADAS A LOS TODOS PROCEDIMIENTOS PARA
  GRAFICAR LA CODIFICACION RZ*)
Var
  i, j : Integer;
Begin
If c1 = 0 Then
  Begin
  Titulo; IngresoDatos;
  End
Else
  Begin
  Titulo;
  MantenerDatos;
  End;
RZ1;
GotoXY(3,23);
Write('       Presione ENTER para graficar los bits y su codificación
respectiva     ');
Continuar;
Graficar;
End;
(***********************************************)
Procedure NRZPOLAR;
(*REALIZA LAS LLAMADAS A LOS TODOS PROCEDIMIENTOS PARA
  GRAFICAR LA CODIFICACION NRZPOLAR*)

Var
  i : Integer;
Begin
ClrScr;
If c1 = 0 Then
  Begin
  Titulo; IngresoDatos;
  End
Else
  Begin
  Titulo; MantenerDatos;
  End;
NRZPOLAR1;
GotoXY(3,23);
Write('       Presione ENTER para graficar los bits y su codificación
respectiva     ');
Continuar;
Graficar;
End;
(***********************************************)
Procedure RZPOLAR;
(*REALIZA LAS LLAMADAS A LOS TODOS PROCEDIMIENTOS PARA
  GRAFICAR LA CODIFICACION RZPOLAAR*)

Var
  i, j : Integer;
Begin
If c1 = 0 Then
  Begin
  Titulo; IngresoDatos;
  End
Else
  Begin
  Titulo; MantenerDatos;
  End;
RZPOLAR1;
GotoXY(3,23);
Write('       Presione ENTER para graficar los bits y su codificación
respectiva     ');
```

```
Continuar;
Graficar;
End;
(***********************************************)
Procedure AMI;
(*REALIZA LAS LLAMADAS A LOS TODOS PROCEDIMIENTOS PARA
  GRAFICAR LA CODIFICACION AMI*)

Var
  i, COF : Integer;
Begin
If c1 = 0 Then
  Begin
  Titulo;IngresoDatos;
  End
Else
  Begin
  Titulo; MantenerDatos;
  End;
AMI1;
GotoXY(3,23);
Write('       Presione ENTER para graficar los bits y su codificación
respectiva     ');
Continuar;
Graficar;
End;
(***********************************************)
Procedure HDB3;
(*REALIZA LAS LLAMADAS A LOS TODOS PROCEDIMIENTOS PARA
  GRAFICAR LA CODIFICACION HDB3*)
Var
  i, k, S, P, COF : Integer;
Begin
If c1 = 0 Then
  Begin
  Titulo; IngresoDatos;
  End
Else
  Begin
  Titulo; MantenerDatos;
  End;
HDB31;
GotoXY(3,23);
Write('       Presione ENTER para graficar los bits y su codificación
respectiva     ');
Continuar;
Graficar;
End;
(***********************************************)
Procedure BnZS;
(*REALIZA LAS LLAMADAS A LOS TODOS PROCEDIMIENTOS PARA
  GRAFICAR LA CODIFICACION BnZS*)
Var
  P : Integer;
Begin
If c1 = 0 Then
  Begin
  Titulo; IngresoDatos;
  End
Else
  Begin
  Titulo; MantenerDatos;
  End;
GotoXY(3,23);
Write(' Ingrese el valor de paridad par = 0 o paridad impar = 1 y luego
pulse ENTER');
GotoXY(6,20);Write('Paridad Inicial (Par : 0, Impar : 1) : ');
GotoXY(45,20);
NUMEROS(45,20,20,21,1,0,P);
BnZS1(P);
```

```
GotoXY(3,23);
Write('       Presione ENTER para graficar los bits y su codificación
respectiva     ');
Continuar;
Graficar;
End;
(***********************************************)
Procedure MANCHESTER;
(*REALIZA LAS LLAMADAS A LOS TODOS PROCEDIMIENTOS PARA
  GRAFICAR LA CODIFICACION MANCHESTER*)
Var
  i, j : Integer;
Begin
If c1 = 0 Then
  Begin
  Titulo; IngresoDatos;
  End
Else
  Begin
  Titulo; MantenerDatos;
  End;
MANCHESTER1;
GotoXY(3,23);
Write('       Presione ENTER para graficar los bits y su codificación
respectiva     ');
Continuar;
Graficar;
End;
(***********************************************)
Procedure BIFASEM;
(*REALIZA LAS LLAMADAS A LOS TODOS PROCEDIMIENTOS PARA
  GRAFICAR LA CODIFICACION BIFASEM*)
Var
  i, j : Integer;
Begin
If c1 = 0 Then
  Begin
  Titulo; IngresoDatos;
  End
Else
  Begin
  Titulo; MantenerDatos;
  End;
BIFASEM1;
GotoXY(3,23);
Write('       Presione ENTER para graficar los bits y su codificación
respectiva     ');
Continuar;
Graficar;
End;
(***********************************************)
Procedure BIFASES;
(*REALIZA LAS LLAMADAS A LOS TODOS PROCEDIMIENTOS PARA
  GRAFICAR LA CODIFICACION BIFASES*)

Var
  i, j : Integer;
Begin
If c1 = 0 Then
  Begin
  Titulo; IngresoDatos;
  End
Else
  Begin
  Titulo; MantenerDatos;
  End;
BIFASES1;
GotoXY(3,23);
Write('       Presione ENTER para graficar los bits y su codificación
respectiva     ');
```

```
Continuar;
Graficar;
End;
(************************************************)
Procedure MILLER;
(*REALIZA LAS LLAMADAS A LOS TODOS PROCEDIMIENTOS PARA
  GRAFICAR LA CODIFICACION MILLER*)

Var
  i, j : Integer;
Begin
If c1 = 0 Then
  Begin
  Titulo; IngresoDatos;
  End
Else
  Begin
  Titulo; MantenerDatos;
  End;
MILLER1;
GotoXY(3,23);
Write('       Presione ENTER para graficar los bits y su codificación
respectiva     ');
Continuar;
Graficar;
End;
(************************************************)
Procedure CUATROB3T;
(*REALIZA LAS LLAMADAS A LOS TODOS PROCEDIMIENTOS PARA
  GRAFICAR LA CODIFICACION 4B3T*)

Var
  P : integer;
Begin
If c1 = 0 Then
  Begin
  Titulo; IngresoDatos;
  End
Else
  Begin
  Titulo; MantenerDatos;
  End;
GotoXY(3,23);
Write('Ingrese el valor de paridad inicial valores desde -3 a 3 y luego
pulse ENTER');
GotoXY(6,20);Write('Paridad Inicial ( -3 ... +3 ) : ');
GotoXY(38,20);
NUMEROS(38,20,20,21,3,-3,P);
CUATROB3T1(P);
GotoXY(3,23);
Write('       Presione ENTER para graficar los bits y su codificación
respectiva     ');
Continuar;
Graficar4B3T;
End;
(************************************************)
Procedure CMI;
(*REALIZA LAS LLAMADAS A LOS TODOS PROCEDIMIENTOS PARA
  GRAFICAR LA CODIFICACION CMI*)

Var
  i, j, COF : Integer;
Begin
If c1 = 0 Then
  Begin
  Titulo; IngresoDatos;
  End
Else
  Begin
  Titulo; MantenerDatos;
```

```
  End;
CMI1;
GotoXY(3,23);
Write('        Presione ENTER para graficar los bits y su codificación
respectiva     ');
Continuar;
Graficar;
End;
(************************************************)
Procedure PST;
(*REALIZA LAS LLAMADAS A LOS TODOS PROCEDIMIENTOS PARA
  GRAFICAR LA CODIFICACION PST*)
Var
  i, P, COF : Integer;
Begin
If c1 = 0 Then
  Begin
  Titulo; IngresoDatos;
  End
Else
  Begin
  Titulo; MantenerDatos;
  End;
GotoXY(3,23);
Write(' Ingrese el valor de paridad par = 0 o paridad impar = 1 y luego
pulse ENTER');
GotoXY(6,20);Write('Paridad Inicial (Par : 0, Impar : 1) : ');
NUMEROS(45,20,20,21,1,0,P);
PST1(P);
GotoXY(3,23);
Write('        Presione ENTER para graficar los bits y su codificación
respectiva     ');
Continuar;
Graficar;
End;
(*****************************************************)
Begin
c1 := 0; es:=' ';
End.
```

UNIDAD CODIGOS2.TPU

```
(*UNIDAD CODIGOS2 CONTIENE LOS PROCEDIMIENTOS O SUBRUTINA PARA
  LA OPCION ESPECTROS DE POTENCIA DE LOS CODIGOS DE LINEA*)
Unit Codigos2;

INTERFACE

Uses CRT, GRAPH;

(*VARIABLES GLOBALES QUE UTILIZA LA UNIDAD*)

Var
  cp, c3, ControlGr, ModoGr : Integer;
  es : Char;
  P1, P2 : Real;
  PRB1, PRB2 : string[6];

(*LISTADO DE LOS PROCEDIMIENTOS QUE UTILIZA LA UNIDAD CODIGOS2*)

Procedure DibujarCuadro(x1,y1,x2,y2,pp,f : integer);
Procedure PintarFondo(x1,y1,x2,y2,f: integer);
Procedure Opciones(x1, y1 : integer);
Procedure Continuar;
Procedure PNRZ(a,c, k :integer; P : real);
Procedure PRZ(a,c, k :integer; P : real);
Procedure PNRZPOLAR(a, c, k :integer; P : real);
Procedure PRZPOLAR(a, c, k :integer; P : real);
Procedure PAMINRZ(a, c :integer; k, P : real);
Procedure PAMIRZ(a, c :integer; k, P : real);
Procedure PMANCH(a, c :integer; k, P : real);
Procedure PHDB3;
Procedure PMILLER;
Procedure CuadroEspectros;
Procedure CompararEspectros;
Procedure DEPEquiprobables;
Procedure NRZPotencia;
Procedure RZPotencia;
Procedure NRZPOLARPotencia;
Procedure RZPOLARPotencia;
Procedure AMINRZPotencia;
Procedure AMIRZPotencia;
Procedure MANCHESTERPotencia;
Procedure DEPCodigos1;
Procedure DEPCodigos2;

IMPLEMENTATION


(************************************************)
(*DESARROLLO DE CADA UNO DE LOS PROCEDIMIENTOS QUE CONTIENE
  LA UNIDAD CODIGOS2.TPU*)
(************************************************)

Procedure DibujarCuadro(x1,y1,X2,y2,pp,f : Integer);
(*DIBUJA LOS BORDES PARA LAS PANTALLAS DE PRESENTACION EN MODO TEXTO*)
Var
  i : integer;
Begin
TextColor(pp);TextBackground(f);
For i := (x1 + 1)  To (x2 - 1) Do
  Begin
    GotoXY(i,y1);Write(#205);
    GotoXY(i,y2);Write(#205);
  End;
For i := (y1 + 1) To (y2 - 1) Do
  Begin
    GotoXY(x1,i);Write(#186);
    GotoXY(x2,i);Write(#186);
```

```
  End;
GotoXY(x1,y1);Write(#201);
GotoXY(x2,y1);Write(#187);
GotoXY(x1,y2);Write(#200);
GotoXY(x2,y2);Write(#188);
End;
(************************************************)
Procedure PintarFondo(x1,y1,x2,y2,f: integer);
(PONE EL COLOR DE FONDO EN LAS PANTALLAS DE PRESENTACION*)

Var
  i,j : integer;
Begin
ClrScr;
For i := x1 To x2 Do
  For j := y1 To y2 Do
    Begin
      GotoXY(i,j);
      Textbackground(f);
      Write(#0);
    End;
End;
(************************************************)
Procedure Opciones(x1, y1 : integer);
(*INDICA LAS OPCIONES PARA SEGUIR EN EL PROGRAMA O FINALIZAR
  PARA ESTO UTILIZA LAS TECLAS ESC, F1, END*)

Begin
  SetColor(14);
  MoveTo(0,y1-30);LineTo(x1,y1-30);
  SetTextStyle(2,0,5);SetTextJustify(1,2);
  OutTextXY(X1 Div 2,Y1-20,' ESC : Menú Anterior  F1 : Menú Códigos  END :
Salir al DOS ');
  es:=' ';
  Repeat
  If KeyPressed Then
  es := ReadKey;
  Until es in [Char(27),Char(59),Char(79)];
End;
(************************************************)
Procedure Continuar;
(*PERMITE CONTINUAR CON EL PROGRAMA UNICAMENTE CUANDO SE PULSE LA
  TECLA ENTER*)

Var
  s : Char;
Begin
  s := ' ';
  Repeat
    If KeyPressed Then
      s := ReadKey;
  Until s = Char(13);
End;
(************************************************)
Procedure NUMEROS2(x1, y1, x2, y2 : Integer;
                   B, C : Real; Var A : Real);
(*LEE LOS DATOS DE PROBABILIBAD P1 Y P2 NUMEROS REALES, UN MENSAJE
  APARECE CUANDO EL DATO ES INCORRECTO*)

Var
  i, j , codigo : integer;
  NU : Array [1..10] of char;
  Numero1 : String[10];
Label
  Repetir;
Begin
REPETIR:
TextColor(14);
GotoXY(x1,y1);
Write('            ');
```

```
GotoXY(x1,y1);
i := 1;
Repeat
If KeyPressed Then
  Begin
  NU[i] := ReadKey;
  Write(NU[i]);
  inc(i,1);
  End;
Until ( NU[i-1] = Char(13)) Or (i = 10);
Numero1:=NU[1];
For j:=2 To i-2 Do
Numero1:=Numero1 + NU[j];
Val(Numero1,A,codigo);
If (codigo <> 0) Or ((A > B) or (A < C)) Then
  Begin
  TextColor(139);
  GotoXY(x2,y2);Write('Dato incorrecto Presione ENTER para continuar');
  Continuar;
  TextColor(15);
  GotoXY(x2-1,y2);Write('═══════════════════════════════════════════════');
  Goto Repetir;
  End;
End;
(************************************************)
Procedure IdentificarProbabilidad(x, y, a : integer);
(*IDENTIFICA A CADA UNA DE LAS CURVAS DE LAS DEP Y SE INDICA CON FLECHAS
  SUS RESPECTIVAS PROBABILIDADES*)

Var
 X1, Y1, X2, X3,Y2, Y3 : Integer;
 S : string[5];
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 8;
  SetTextJustify(1,1);
If (x = X2+20) and (a = 1) Then
  Begin
  SetColor(15);
  MoveTo(x,y);LineRel(0,-3);
  MoveTo(x,y);LineRel(20,-15);
  X3 := GetX; Y3 := GetY; S := 'P1=0.';
  OutTextXY(X3,Y3-10,S+PRB1);
  End;
If (P1 <> P2) Then
  Begin
  If (x = 2*X2+20) and (a = 2) Then
    Begin
    SetColor(14);
    MoveTo(x,y);LineRel(0,-3);
    MoveTo(x,y);LineRel(20,-15);
    X3 := GetX; Y3 := GetY; S := 'P2=0.';
    OutTextXY(X3,Y3-10,S+PRB2);
    End;
  End;
SetColor(15);
End;
(************************************************)
Procedure PNRZ(a, c, k :integer; P:real);
(*EVALUA LA ECUACION DE LA DEP PARA EL CODIGO NRZ*)
Var
 i, X, Y, X1, Y1, X2, Y2, Y3, Y4, Y5 : Integer;
 G, Q1 : Real;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 8; Y3 := Y2 ;
  i := 0;G:=0.00001;
  While i < 6*x2 Do
    Begin
      X := i + x2;
```

```
      Q1:= (sin(G))/G;
      Y := (7*y2)-ROUND(4*1.25*y3*k*(Q1*Q1)*(P-P*P));
      PutPixel(X,Y,c);G := G + (pi/150);
      inc(i,1);
      IdentificarProbabilidad(X,Y,a);
    End;
  Y4:=Round(5*k*Y3*P*(1-P)*PI/3.5);
  Y5:=7*Y2-Y4;
  SetColor(c);
  SetLineStyle(0,0,3);
  MoveTo(x2,7*Y2);LineTo(x2,Y5);
  SetTextstyle(2,0,4);
  SetLineStyle(0,0,1);
  SetColor(15);
End;
(*********************************************)
Procedure PRZ(a, c, k :integer; P:real);
(*EVALUA LA ECUACION DE LA DEP PARA EL CODIGO RZ*)

Var
 i, X, Y, X1, Y1, X2, Y2, Y3, Y4,Y5,Y6,Y7 : Integer;
 G,Q1 : Real;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 8; Y3 := Y2 ;
  i := 0;G:=0.00001;
  While i < 6*x2 Do
    Begin
      X := i + x2;
      Q1:=(Sin(0.5*G))/(0.5*G);
      Y := (7*y2)-ROUND(4*0.3125*y3*k*(Q1*Q1)*(P-P*P));
      PutPixel(X,Y,c);G := G + (pi/150);
      inc(i,1);
      IdentificarProbabilidad(X,Y,a);
    End;
  Y4:=Round(4*0.3125*k*Y3*P*(1-P)*PI/3.5);
  Y5:=7*Y2-Y4;
  SetColor(c);
  SetLineStyle(0,0,3);
  MoveTo(x2,7*Y2);LineTo(x2,Y5);
  SetTextstyle(2,0,4);
  SetLineStyle(0,0,1);
  SetColor(15);
End;
(*********************************************)
Procedure PNRZPOLAR(a, c, k :integer; P:real);
(*EVALUA LA ECUACION DE LA DEP PARA EL CODIGO NRZ POLAR*)

Var
 i, X, Y, X1, Y1, X2, Y2, Y3, Y4, Y5 : Integer;
 G, Q1 : Real;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 8;Y3:=Y2 Div 2

  i := 0;G:=0.00001;
  While i < 6*x2 Do
    Begin
      X := i + x2;
      Q1:=(Sin(G))/(G);
      Y := (7*y2)-ROUND((4*10*Y3)*k*(Q1*Q1)*(P-P*P));
      PutPixel(X,Y,C);G := G + (pi/150);
      inc(i,1);
      IdentificarProbabilidad(X,Y,a);
    End;
  Y4:=ABS(Round(40*k*Y3*(2*P-1)*PI/((1-P)*3.5)));
  Y5:=7*Y2-Y4;
  SetColor(c);
  SetLineStyle(0,0,3);
  MoveTo(x2,7*Y2);LineTo(x2,Y5);
```

```
  SetTextstyle(2,0,4);
  SetLineStyle(0,0,1);
  SetColor(15);
End;
(***********************************************)
Procedure PRZPOLAR(a, c, k :integer; P:real);
(*EVALUA LA ECUACION DE LA DEP PARA EL CODIGO RZ POLAR*)

Var
 i, X, Y, X1, Y1, X2, Y2, Y3, Y4, Y5 : Integer;
 G, Q1 : Real;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 8; Y3 := Y2 ;
  i := 0;G:=0.00001;
  While i < 6*x2 Do
    Begin
      X := i + x2;
      Q1:=(Sin(0.5*G))/(0.5*G);
      Y := (7*y2)-ROUND((4*1.25*Y3)*k*(Q1*Q1)*(P-P*P));
      PutPixel(X,Y,c);G := G + (pi/150);
      inc(i,1);
      IdentificarProbabilidad(X,Y,a);
      End;
  Y4:=ABS(Round(5*k*Y3*(2*P-1)*PI/((1-P)*3.5*4)));
  Y5:=7*Y2-Y4;
  SetColor(c);
  SetLineStyle(0,0,3);
  MoveTo(x2,7*Y2);LineTo(x2,Y5);
  SetTextstyle(2,0,4);
  SetLineStyle(0,0,1);
  SetColor(15);
End;

(***********************************************)
Procedure PAMINRZ(a, c:integer; k, P:real);
(*EVALUA LA ECUACION DE LA DEP PARA EL CODIGO AMI NRZ*)

Var
 i, X, Y, X1, Y1, X2, Y2, Y3 : Integer;
 G, Q1, Q2, Q3 : Real;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 8; Y3 := Y2 ;
  i := 0;G:=0.00001;
  While i < 6*x2 Do
    Begin
      X := i + x2;
      Q1:=(Sin(G))/(G);
      Q2:=Sin(G);
      Q3:=(1+(2*P-1)*(2*P-1)+2*(2*P-1)*COS(2*G));
      Y := (7*y2)-ROUND(20*Y3*k*Q1*Q1*Q2*Q2*((P-P*P)/Q3));
      PutPixel(X,Y,c);G := G + (pi/150);
      inc(i,1);
      IdentificarProbabilidad(X,Y,a);
    End;
End;
(***********************************************)
Procedure PAMIRZ(a, c :integer; k, P:real);
(*EVALUA LA ECUACION DE LA DEP PARA EL CODIGO AMI RZ*)

Var
 i, X, Y, X1, Y1, X2, Y2, Y3 : Integer;
 G, Q1, Q2, Q3 : Real;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 8; Y3 := Y2 ;
  i := 0;G:=0.00001;
  While i < 6*x2 Do
    Begin
```

```
      X := i + x2;
      Q1:=(Sin(0.5*G))/(0.5*G);
      Q2:=Sin(G);
      Q3:=(1+(2*P-1)*(2*P-1)+2*(2*P-1)*COS(2*G));
      Y := (7*y2)-ROUND(5*Y3*k*Q1*Q1*Q2*Q2*((P-P*P)/Q3));
      PutPixel(X,Y,c);G := G + (pi/150);
      inc(i,1);
      IdentificarProbabilidad(X,Y,a);
    End;
End;
(************************************************)
Procedure PMANCH(a, c:integer; k, P:real);
(*EVALUA LA ECUACION DE LA DEP PARA EL CODIGO MANCHESTER*)

Var
 i, X, Y, X1, Y1, X2, Y2, Y3 : Integer;
 G, S1, S2, S3 : Real;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 8; Y3 := Y2 ;
  i := 0;G:=0.00001;
  While i < 6*x2 Do
    Begin
      X := i + x2;
      S1 := Sin(0.25*G);
      S2 := 0.25*G;
      S3 := 4*P*(1-P);
      Y := (7*y2)-ROUND(5*y3*k*(S1/S2)*(S1/S2)*S3*S1*S1);
      PutPixel(X,Y,c);G := G + (pi/75);
      inc(i,1);
      IdentificarProbabilidad(X,Y,a);
    End;
End;
(*****************************************************)
Procedure PHDB3;
(*EVALUA LA ECUACION DE LA DEP PARA EL CODIGO HDB3*)

Var
 i, X, Y, X1, Y1, X2, Y2, Y3 : Integer;
 G, S, S1, S2, Q, Q1, Q2, Q3, Q4, Q5 : Real;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 8; Y3 := Y2 ;
  i := 0;G:=0.00001;
  While i < 6*x2 Do
    Begin
      X := i + x2;
      Q := 2*G;
      Q1:= 40-32*Cos(Q);
      Q2:= (465*(1025-64*Cos(5*Q)));
      Q3:= (7258.5-1929*Cos(Q)-1424*Cos(2*Q)-160*Cos(3*Q)+32*Cos(4*Q));
      Q4:= 131288.5-41399*Cos(Q)-86112*Cos(2*Q);
      Q5:= 85-44*Cos(Q)-24*Cos(2*Q)-16*Cos(3*Q);
      S1:= (Sin(0.5*G))/(0.5*G);
      S2:= (Sin(0.5*G));
      Y := (7*y2)-ROUND(5*Y3*(S1*S1)*(S2*S2)*((Q1/Q2)*(Q3-(Q4/Q5))));
      PutPixel(X,Y,14);G := G + (pi/150);
      inc(i,1);
    End;
End;
(*****************************************************)
Procedure PMILLER;
(*EVALUA LA ECUACION DE LA DEP PARA EL CODIGO MILLER*)

Var
 i, X, Y, X1, Y1, X2, Y2, Y3 : Integer;
 G, S, Q, Q1, Q2 : Real;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 8;Y3 := Y2 Div 2;
```

```
  MoveTo(x2-2,7*y2);
  For i:= 1 To 13 Do
    Begin
      LineRel(4,0); Y := GetY;
      Y :=Y -j; X :=x2-2;
      MoveTo(X,Y);
    End;
  SetTextStyle(2,0,4);SetTextJustify(1,2);
  OutTextXY(x2,7*y2+6,'0');OutTextXY(3*x2,7*y2+6,'2π/To');
  OutTextXY(5*x2,7*y2+6,'4π/To');OutTextXY(7*x2,7*y2+6,'6π/To');
  OutTextXY(7*X2+10,7*y2-5,'w');
  x:=x2 div 2; y:= y2 div 2;
  OutTextXY(x2,y2-15,'Sy(w)/A²');Delay(1000);
  SetTextJustify(2,2);
  OutTextXY(x2-5,(9*y),'0.5To');OutTextXY(x2-5,2*Y2,'1.0To');
  End;
(*****************************************************)
Procedure ComparарEspectros;
(*REALIZA LA COMAPARACION DE ESPECTROS, LLAMANDO A LOS PROCEDIMIENTOS PARA
  GRAFICAR LA DEP DE LOS CODIGOS CORRESPONDIENTES, A LA TECLA PULSADA*)

Var
  X1, Y1, X2, Y2, X3, Y3 : Integer;
  T1, T2, T3, T4 : String;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 8;
  X3:= Y1 Div 16; Y3:=Y1 Div 16;
  Delay(1000);SetTextJustify(1,2);
  T1 := '- PARA COMPARAR EL ESPECTRO DE LOS CODIGOS ';
  T2 := 'LISTADOS  PULSAR EL NUMERO CORRESPONDIENTE';
  OutTextXY(X1 Div 2,Y1-28, T1+T2);
  T3 := '1. NRZ  2. RZ  3. NRZ-POLAR  4. RZ-POLAR  5. AMI-NRZ  ';
  T4 := '6. AMI-RZ  7. MANCH.  8. HDB3  9. MILLER  ESC : MENU ';
  OutTextXY(X1 Div 2,Y1-15,T3+T4);SetTextJustify(0,2);
  es := ' ';
  Repeat
    If KeyPressed Then
      Begin
        es := ReadKey;
        Case es Of
        '1' : Begin
              PNRZ(0,13,1,0.5);SetColor(13);
              OutTextXY(X2+10,6*Y2-5,'NRZ')
              End;
        '2' : Begin
              PRZ(0,15,1,0.5);SetColor(15);
              OutTextXY(X2+5,7*Y2-Y3,'RZ')
              End;
        '3' : Begin
              PNRZPOLAR(0,14,1,0.5);SetColor(14);
              OutTextXY(X2+5,2*Y2,'NRZ-P')
              End;
        '4' : Begin
              PRZPOLAR(0,7,1,0.5);SetColor(7);
              OutTextXY(X2+X3,6*Y2-Y3,'RZ-P')
              End;
        '5' : Begin
              PAMINRZ(0,9,1,0.5);SetColor(9);
              OutTextXY(X2+X3,4*Y2+Y3,'AMI-NRZ')
              End;
        '6' : Begin
              PAMIRZ(0,11,1,0.5);SetColor(11);
              OutTextXY(2*X2,6*Y2+5,'AMI-RZ')
              End;
        '7' : Begin
              PMANCH(0,13,0.5,0.5);SetColor(13);
              OutTextXY(2*X2+Y3,6*Y2-Y3,'MANCHESTER')
              End;
        '8' : Begin
```

```
                PHDB3;SetColor(15);
                OutTextXY(2*X2+10,5*Y2-Y3,'HDB3')
                End;
          '9' : Begin
                PMILLER;SetColor(14);
                OutTextXY(2*X2-5,2*Y2,'MILLER')
                End;
          End;
          SetColor(15);
        End
  Until es = Char(27);
End;
(*******************************************************)
Procedure DEPEQUIPROBABLES;
(*MANEJA LOS PROCEDIMIENTOS PARA GRAFICAR LA COMPARACION DE LOS ESPECTROS
  PARA EQUIPROBABILIDAD*)

Var
  X1 : Integer;
  s: Char;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1 := GetMaxX;c3:=0;SetColor(2);
  SetColor(11);
  SetTextStyle(2,0,5);SetTextJustify(1,2);c3:=0;
  OutTextXY(X1 Div 2,15,'COMPARACION DE LOS ESPECTROS DE POTENCIA PARA
PROBABILIDAD P = 0.5');
  CuadroEspectros1;
  CompararEspectros;
  Closegraph;
End;
(*******************************************************)
Procedure NRZPotencia;
(*LLAMA A LOS PROCEDIMIENTOS NECESARIOS PARA GRAFICAR LA DEP DEL CODIGO NRZ
  PARA DOS VALORES DE PROBABILIDAD*)
Var
  X1, Y1 ,X2, Y2 : Integer;
  s: Char;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1 := GetMaxX; Y1 := GetMaxY;
  X2 := X1 Div 8; Y2 := Y1 Div 8;
  SetColor(11);
  SetTextStyle(2,0,5);SetTextJustify(1,2);
  OutTextXY(X1 Div 2,15,'ESPECTROS DE POTENCIA DEL CODIGO NRZ PARA DOS
  PROBABILIDADES ');
  OutTextXY(X1 Div 2,30,'P1 = 0.'+PRB1+'          '+'P2 = 0.'+PRB2);
  CuadroEspectros;
  PNRZ(1,15,2,P1); PNRZ(2,14,2,P2);
  OutTextXY(x2-37,(9*(Y2 Div 2)),'0.25To');OutTextXY(x2-37,2*Y2,'0.5To');
  Opciones(x1,y1);
  Closegraph;
End;
(*******************************************************)
Procedure RzPotencia;
(*LLAMA A LOS PROCEDIMIENTOS NECESARIOS PARA GRAFICAR LA DEP DEL CODIGO RZ
  PARA DOS VALORES DE PROBABILIDAD*)

Var
  X1, Y1 ,X2, Y2, Y3 : integer;
  s: Char;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1 := GetMaxX; Y1 := GetMaxY;
  X2 :=X1 Div 8; Y2 := Y1 Div 8; Y3 := Y2 Div 2;
  SetColor(11);
  SetTextStyle(2,0,5);SetTextjustify(1,2);
```

```
  OutTextXY(X1 Div 2,15,'ESPECTROS DE POTENCIA DEL CODIGO RZ PARA DOS
PROBABILIDADES ');
  CuadroEspectros;
  OutTextXY(x2-40,(9*Y3),'0.125To');OutTextXY(x2-40,2*Y2,'0.25To');
  PRZ(1,15,4,P1);PRZ(2,14,4,P2);
  Opciones(x1,y1);
  Closegraph;
End;
(*********************************************)
Procedure AmiNRZPotencia;
(*LLAMA A LOS PROCEDIMIENTOS NECESARIOS PARA GRAFICAR LA DEP DEL CODIGO AMI
  NRZ PARA DOS VALORES DE PROBABILIDAD*)

Var
  X1, Y1 ,X2, Y2 : integer;
  s: Char;
Begin
 ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1 := GetMaxX; Y1 := GetMaxY;
  X2 := X1 Div 8; Y2 := Y1 Div 8;
  SetColor(11);
  SetTextStyle(2,0,5);SetTextjustify(1,2);
  OutTextXY(X1 Div 2,15,'ESPECTROS DE POTENCIA DEL CODIGO AMI NRZ PARA DOS
PROBABILIDADES ');
  OutTextXY(X1 Div 2,30,'P1 = 0.'+PRB1+'          '+'P2 = 0.'+PRB2);
  CuadroEspectros;
  If ((P1 > 0.7) or (P2 > 0.7)) Then
    Begin
    PAMINRZ(1,15,1,P1);PAMINRZ(2,14,1,P2);
    OutTextXY(x2-37,(9*(Y2 Div 2)),'1.0To');OutTextXY(x2-37,2*Y2,'2.0To');
    End
  Else
    Begin
    PAMINRZ(1,15,1,P1);PAMINRZ(2,14,1,P2);
    OutTextXY(x2-37,(9*(Y2 Div 2)),'0.5To');OutTextXY(x2-37,2*Y2,'1.0To');
    End;
  Opciones(x1,y1);
  Closegraph;
End;
(*********************************************)
Procedure AmiRZPotencia;
(*LLAMA A LOS PROCEDIMIENTOS NECESARIOS PARA GRAFICAR LA DEP DEL CODIGO AMI
  RZ POTENCIA PARA DOS VALORES DE PROBABILIDAD*)

Var
  X1, Y1 ,X2, Y2 : integer;
  s: Char;
Begin
 ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1 := GetMaxX; Y1 := GetMaxY;
  X2 := X1 Div 8; Y2 := Y1 Div 8;
  SetColor(11);
  SetTextStyle(2,0,5);SetTextjustify(1,2);
  OutTextXY(X1 Div 2,15,'ESPECTROS DE POTENCIA DEL CODIGO AMI RZ PARA DOS
PROBABILIDADES ');
  OutTextXY(X1 Div 2,30,'P1 = 0.'+PRB1+'          '+'P2 = 0.'+PRB2);
  CuadroEspectros;
  If ((P1 > 0.7) or (P2 > 0.7)) Then
    Begin
    PAMIRZ(1,15,1,P1);PAMIRZ(2,14,1,P2);
    OutTextXY(x2-37,(9*(Y2 Div 2)) '1.0To');OutTextXY(x2-37,2*Y2,'2.0To');
    End
  Else
    Begin
    PAMIRZ(1,15,2,P1);PAMIRZ(2,14,2,P2);
    OutTextXY(x2-37,(9*(Y2 Div 2)),'0.5To');OutTextXY(x2-37,2*Y2,'1.0To');
    End;
  Opciones(x1,y1);
```

```
  Closegraph;
End;
(***********************************************)
Procedure NRZPOLARPOTENCIA;
(*LLAMA A LOS PROCEDIMIENTOS NECESARIOS PARA GRAFICAR LA DEP DEL CODIGO NRZ
  POLAR PARA DOS VALORES DE PROBABILIDAD*)

Var
  X1, Y1 ,X2, Y2 : integer;
  s: Char;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1 := GetMaxX; Y1 := GetMaxY;
  X2 := X1 Div 8; Y2 := Y1 Div 8;
  SetColor(11);
  SetTextStyle(2,0,5);SetTextjustify(1,2),
  OutTextXY(X1 Div 2,15,'ESPECTROS DE POTENCIA DEL CODIGO NRZ POLAR PARA DOS
PROBABILIDADES ');
  OutTextXY(X1 Div 2,30,'P1 = 0.'+PRB1+'          '+'P2 = 0.'+PRB2);
  CuadroEspectros;
  PNRZPOLAR(1,15,1,P1); PNRZPOLAR(2,14,1,P2);
  OutTextXY(x2-37,(9*(Y2 Div 2)),'0.5To'),OutTextXY(x2-37,2*Y2,'1.0To');
  Opciones(x1,y1);
  Closegraph;
End;
(***********************************************)
Procedure RZPOLARPOTENCIA;
(*LLAMA A LOS PROCEDIMIENTOS NECESARIOS PARA GRAFICAR LA DEP DEL CODIGO RZ
  POLAR PARA DOS VALORES DE PROBABILIDAD*)

Var
  X1, Y1 ,X2, Y2 : integer;
  s: Char;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1 := GetMaxX; Y1 := GetMaxY;
  X2 := X1 Div 8; Y2 := Y1 Div 8;
  SetColor(11);
  SetTextStyle(2,0,5);SetTextjustify(1,2);
  OutTextXY(X1 Div 2,15,'ESPECTROS DE POTENCIA DEL CODIGO RZ POLAR PARA DOS
PROBABILIDADES ');
  OutTextXY(X1 Div 2,30,'P1 = 0.'+PRB1+'          '+'P2 = 0.'+PRB2);
  CuadroEspectros;
  PRZPOLAR(1,15,2,P1); PRZPOLAR(2,14,2,P2);
  OutTextXY(x2-37,(9*(Y2 Div 2)),'0.25To');OutTextXY(x2-37,2*Y2,'0.5To');
  Opciones(x1,y1);
  Closegraph;
End;
(***********************************************)
Procedure ManchesterPotencia;
(*LLAMA A LOS PROCEDIMIENTOS NECESARIOS PARA GRAFICAR LA DEP DEL CODIGO
  MANCHESTER PARA DOS VALORES DE PROBABILIDAD*)

Var
  X1, Y1 ,X2, Y2 : integer;
  s: Char;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1 := GetMaxX; Y1 := GetMaxY;
  X2 := X1 Div 8; Y2 := Y1 Div 8;
  SetColor(11);
  SetTextStyle(2,0,5);SetTextjustify(1,2);
  OutTextXY(X1 Div 2,15,'ESPECTROS DE POTENCIA DEL CODIGO MANCHESTER PARA
DOS PROBABILIDADES ');
  OutTextXY(X1 Div 2,30,'P1 = 0.'+PRB1+'          '+'P2 = 0.'+PRB2);
  CuadroEspectros;
  PNRZPOLAR(1,15,1,P1); PNRZPOLAR(2,14,1,P2);
```

```
  OutTextXY(x2-37,(9*(Y2 Div 2)),'0.5To');OutTextXY(x2-37,2*Y2,'1.0To');
  Opciones(x1,y1);
  Closegraph;
End;
(***********************************************)
Procedure DEPCodigos1;
(*PANTALLA DE PRESENTACION Y SELECCION DEL CODIGO CUYA DEP SE DESEA COMPARAR
  PARA DOS VALORES DE PROBABILIDAD P1 Y P2*)

Var
  i : integer;
  s : char;
Begin
ClrScr;c3:=1;
PintarFondo(1,1,80,25,1);
DibujarCuadro(1,2,80,25,15,1);
TextColor(4);TextBackground(7);
GotoXY(25,3);WriteLn(' C O D I G O S   D E   L I N E A ');
TextColor(0);
GotoXY(18,5);
Write(' E S P E C T R O S   D E   P O T E N C I A ');
DibujarCuadro(16,7,64,14,15,1);
TextColor(14);
GotoXY(23,9);Write('A) NRZ (Neutral) ');
GotoXY(23,10);Write('B) RZ  (Neutral)');
GotoXY(23,11);Write('C) NRZ-POLAR ");
GotoXY(23,12);Write('D) RZ-POLAR');
GotoXY(45,9);Write('E) AMI-NRZ');
GotoXY(45,10);Write('F) AMI-RZ');
GotoXY(45,11);Write('G) MANCHESTER ');
GotoXY(10,15);Write('[F1]:Menú Códigos   [Esc]:Menú Espectros   [End]:Salir
al DOS');
Textcolor(15);
GotoXY(1,22);Write(#204);GotoXY(80,22);Write(#185);
For i:=2 To 79 Do
   Begin
   GotoXY(i,22);Write(#205);
   End;
GotoXY(3,23); TextColor(14);
Write('      Ingrese la tecla correspondiente al código cuya DEP desea
observar      ');
GotoXY(26,17);Write('Seleccione el código deseado...[   ]');
GotoXY(59,17);
es := ' ';
Repeat
If KeyPressed Then
   Begin
   es := ReadKey;
   Case es Of
        'A'..'G','a'..'g' : Begin
                            Write(es);Delay(500);
                            End;
Char(27),Char(59),Char(79): Write(' ');
                            Else
                            Begin
                            Write(^g);Delay(500);
                            End;
                            End;
                            End;
Until es in ['A'..'G','a'..'g',Char(27),Char(59),Char(79)];
End;
(***************************************************)
Procedure DEPCodigos2;
(PERMITE EL INGRASO DE LOS VALORES DE PROBABILIDAD P1 Y P2*)
Var
  P3, P4 : Integer;
  s : char;
Begin;
GotoXY(3,23);
Write('     Ingrese dos valores de probabilidad P1 y P2, valores entre 0 y
```

```
1       ');
GotoXY(14,19);
Write('Ingrese los valores de las probabilidades  [0 < P < 1 ] : ');
GotoXY(16,21);
Write('Probabilidad P1 =  ');
NUMEROS2(34,21,20,22,0.9999,0.0001,P1);
GotoXY(46,21);
Write('Probabilidad P2 =  ');
NUMEROS2(64,21,20,22,0.9999,0.0001,P2);
P3 := ROUND(P1*1000);
P4 := ROUND(P2*1000);
STR(P3,PRB1); STR(P4,PRB2);
GotoXY(3,23);
Write(' Presione ENTER para graficar los ESPECTROS DE POTENCIA del código
elegido  ');
Continuar;
Case es Of
 'A','a':NRZPOTENCIA;
 'B','b':RZPOTENCIA;
 'C','c':NRZPOLARPOTENCIA;
 'D','d':RZPOLARPOTENCIA;
 'E','e':AMINRZPOTENCIA;
 'F','f':AMIRZPOTENCIA;
 'G','g':MANCHESTERPOTENCIA;
 End;
End;
(************************************************)
FIN DE LA UNIDAD CODIGOS 2
(************************************************)
End.
```

UNIDAD CODIGOS3.TPU

```
(*UNIDAD CODIGOS3, CONTIENE LOS PROCEDIMEINTOS O SUBRUTINAS
  PARA LA OPCION TEORIA DE CODIGOS DE LINEA*)
Unit Codigos3;

INTERFACE

(*VARIABLES GLOBALES QUE UTILIZA LA UNIDAD*)

Uses CRT, GRAPH;
Var
  M, N, cp, ControlGr, ModoGr : Integer;
  Tecla, es : Char;
  dato, reloj, coef : Array[1..60] Of Integer;
  bit : Array[1..30] Of Char;
  coef1 : Array[1..30] Of Char;

(*LISTA DE PROCEDIMIENTOS QUE UTILIZA LA UNIDAD*)

Procedure Opciones(x1, y1 : integer);
Procedure Continuar;
Procedure NRZ1;
Procedure RZ1;
Procedure NRZPOLAR1;
Procedure RZPOLAR1;
Procedure AMI1;
Procedure HDB31;
Procedure MANCHESTER1;
Procedure BnZS1(P : Integer);
Procedure BIFASEM1;
Procedure BIFASES1;
Procedure MILLER1;
Procedure CUATROB3T1(P : Integer);
Procedure CMI1;
Procedure PST1(P : Integer);
Procedure GraficarEjemplo(h,k,x2,y2,y3 : integer);
Procedure GrafEjmCB3T(h, k, X2, Y2, Y3 : integer);
Procedure NRZTeoria;
Procedure RZTeoria;
Procedure NRZPOLARTeoria;
Procedure RZPOLARTeoria;
Procedure AMITeoria;
Procedure HDB3Teoria;
Procedure MANCHESTERTeoria;
Procedure BnZSTeoria;
Procedure BIFASEMTeoria;
Procedure BIFASESTeoria;
Procedure MILLERTeoria;
Procedure C4B3TTeoria;
Procedure CMITeoria;
Procedure PSTTeoria;
IMPLEMENTATION
(***********************************************)
Procedure Opciones(x1, y1 : integer);
(*PERMITE ELEGIR ENTRE CONTINUAR EL PROGRAMA O SALIR AL DOS UTILIZANDO LAS
  TECLAS ESC, F1, END*)

Begin
  SetColor(14);
  MoveTo(0,y1-30);LineTo(x1,y1-30);
  SetTextStyle(2,0,5);SetTextJustify(1,2);
  OutTextXY(X1 Div 2,Y1-20,' ESC : Menú Anterior  F1 : Menú Códigos  END :
Salir al DOS ');
  es:=' ';
  Repeat
  If KeyPressed Then
  es := ReadKey;
  Until es in [Char(27),Char(59),Char(79)];
End;
```

```
(************************************************)
Procedure Continuar;
(*CONTINUA LA EJECUCION DEL PROGRAMA AL PULSAR LA TECLA ENTER SOLAMENTE*)

Var
  s : Char;
Begin
  s := ' ';
  Repeat
    If KeyPressed Then
      s := ReadKey;
  Until s = Char(13);
End;
(************************************************)
Procedure NRZ1;
(*ENCUENTRA LOS COEFICIENTES NECESARIOS PARA REALIZAR EL GRAFICO DE LA SEÑAL
  CODIFICADA NRZ EN EL EJEMPLO DE LA TEORIA*)
Var
  i : Integer;
Begin
M := N;cp := 1;
For i := 1 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    coef [i] := 0;
    End
  Else
    Begin
    coef[i] := 1;
    End;
  End;
End;
(************************************************)
Procedure RZ1;
(*ENCUENTRA LOS COEFICIENTES NECESARIOS PARA REALIZAR EL GRAFICO DE LA SEÑAL
  CODIFICADA RZ EN EL EJEMPLO DE LA TEORIA*)

Var
  i, j : Integer;
Begin
M := N*2;cp:=2;
j := 1;
For i := 1 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    coef [j] := 0;
    coef [j+1] := 0;
    j := j + 2;
    End
  Else
    Begin
    coef[j] := +1;
    coef[j+1] := 0;
    j := j + 2;
    End;
  End;
End;
(************************************************)
Procedure NRZPOLAR1;
(*ENCUENTRA LOS COEFICIENTES NECESARIOS PARA REALIZAR EL GRAFICO DE LA SEÑAL
  CODIFICADA NRZ POLAR EN EL EJEMPLO DE LA TEORIA*)

Var
  i : Integer;
Begin
M := N;cp := 1;
For i := 1 To N Do
  Begin
```

```
  If (bit[i] = '0') Then
    Begin
    coef [i] := -1;
    End
  Else
    Begin
    coef[i] := 1;
    End;
  End;
End;
(************************************************)
Procedure RZPOLAR1;
(*ENCUENTRA LOS COEFICIENTES NECESARIOS PARA REALIZAR EL GRAFICO DE LA SEÑAL
  CODIFICADA NR POLAR EN EL EJEMPLO DE LA TEORIA*)

Var
  i, j : Integer;
Begin
M := N*2;cp:=2;
j := 1;
For i := 1 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    coef [j] := -1;
    coef [j+1] := 0;
    j := j + 2;
    End
  Else
    Begin
    coef[j] := +1;
    coef[j+1] := 0;
    j := j + 2;
    End;
  End;
End;
(************************************************)
Procedure AMI1;
(*ENCUENTRA LOS COEFICIENTES NECESARIOS PARA REALIZAR EL GRAFICO DE LA SEÑAL
  CODIFICADA AMI NRZ EN EL EJEMPLO DE LA TEORIA*)

Var
  i, COF : Integer;
Begin
COF := -1;
M := N;cp := 1;
For i := 1 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    coef [i] := 0;
    End
  Else
    Begin
    COF := (COF * -1);
    coef[i] := COF;
    End;
  End;
End;
(************************************************)
Procedure HDB31;
(*ENCUENTRA LOS COEFICIENTES NECESARIOS PARA REALIZAR EL GRAFICO DE LA SEÑAL
  CODIFICADA HDB3 EN EL EJEMPLO DE LA TEORIA*)

Var
  i, k, S, COF : Integer;
Begin
COF := -1;cp := 1;
M := N; S := 0;
k := 1;
```

```
If (bit[1] = '0') And (bit[2] = '0')
   And (bit[3]='0') And (bit[4] = '0') Then
  Begin
  coef[1] := 0; coef[2] := 0;
  coef[3] := 0; coef[4] := 1;
  i := 5; S:= 1;
  K:=2;
  End
Else
  i :=1;

While (i <= N) Do
  Begin
  If (bit[i] = '0') And (bit[i+1] = '0')
     And (bit[i+2] = '0') And (bit[i+3] = '0') Then
    Begin
    If ( K MOD 2 = 0)   Then
      Begin
      If (coef[i-1] > 0) Then
        Begin
        coef[i] := -1; coef[i+1] := 0;
        coef[i+2] := 0; coef[i+3] := -1;
        COF := -1;
        S := S + 1;
        i := i + 4;
        k := 0;
        End
      Else
        Begin
        coef[i] := 1; coef[i+1] := 0;
        coef[i+2] := 0; coef[i+3] := 1;
        COF := 1;
        S := S + 1;
        i := i + 4;
        k := 0;
        End
      End
    Else
      Begin
      If (coef[i-1] > 0) Then
        Begin
        coef[i] := 0; coef[i+1] := 0;
        coef[i+2] := 0; coef[i+3] := 1;
        COF := 1;
        S := S + 1;
        i := i + 4;
        k := 0;
        End
      Else
        Begin
        coef[i] := 0; coef[i+1] := 0;
        coef[i+2] := 0; coef[i+3] := -1;
        COF := -1;
        S := S + 1;
        i := i + 4;
        k := 0;
        End
      End
    End
   Else
    Begin
    If (bit[i] = '0') Then
      Begin
      coef[i] := 0;
      i := i + 1;
      End
    Else
      Begin
      COF := COF * (-1);
      coef[i] := COF;
```

```
      inc(i,1);
      If (S <> 0) Then
        Begin
        k := k+1;
        End
      End
    End
  End;
End;
(***********************************************)
Procedure BnZS1(P : Integer);
(*ENCUENTRA LOS COEFICIENTES NECESARIOS PARA REALIZAR EL GRAFICO DE LA SEÑAL
  CODIFICADA BNZS EN EL EJEMPLO DE LA TEORIA*)

Var
  i, S, COF : Integer;
Begin
M := N; cp := 1;
COF := -1;
S := 0;
If (bit[1] = '0') And (bit[2] = '0')
   And (bit[3] = '0') Then
  Begin
  coef[1] := 1;
  coef[2] := 0;
  coef[3] := 1; COF := 1;
  i := 4;
  S := 1;
  P := 0;
  End
Else
  i := 1;
While (i <= N) Do
  Begin
  If (bit[i] = '0') And (bit[i+1] = '0')
     And (bit[i+2] = '0') Then
    Begin
    If ( P MOD 2 = 0 )  Then
      Begin
      If (coef[i-1] > 0) Then
        Begin
        coef[i] := -1;
        coef[i+1] := 0;
        coef[i+2] := -1; COF := -1;
        S := S + 1;
        i := i + 3;
        P := 0;
        End
      Else
        Begin
        coef[i] := 1;
        coef[i+1] := 0;
        coef[i+2] := 1; COF := 1;
        S := S + 1;
        i := i + 3;
        P := 0;
        End
      End
    Else
      Begin
      If (coef[i-1] > 0) Then
        Begin
        coef[i] := 0;
        coef[i+1] := 0;
        coef[i+2]  := 1; COF := 1;
        S := S + 1;
        i := i + 3;
        P := 0;
        End
      Else
```

```
        Begin
        coef[i] := 0;
        coef[i+1] := 0;
        coef[i+2] := -1; COF := -1;
        S := S + 1;
        i := i + 3;
        P := 0;
        End
      End
    End
  Else
    Begin
    If (bit[i] = '0') Then
      Begin
      coef[i] := 0;
      inc(i,1);
      End
    Else
      Begin
      COF := COF*(-1);
      coef[i] := COF;
      inc(i,1);
      If (S <> 0) Then
       Begin
        inc(P,1);
       End;
      End
    End
  End;
End;
(************************************************)
Procedure MANCHESTER1;
(*ENCUENTRA LOS COEFICIENTES NECESARIOS PARA REALIZAR EL GRAFICO DE LA SEÑAL
  CODIFICADA MANCHESTER EN EL EJEMPLO DE LA TEORIA*)

Var
  i, j : Integer;
Begin
M := N*2;cp := 2;
j := 1;
For i := 1 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    coef [j] := -1;
    coef [j+1] := +1;
    j := j + 2;
    End
  Else
    Begin
    coef[j] := +1;
    coef[j+1] := -1;
    j := j + 2;
    End;
  End;
End;
(************************************************)
Procedure BIFASEM1;
(*ENCUENTRA LOS COEFICIENTES NECESARIOS PARA REALIZAR EL GRAFICO DE LA SEÑAL
  CODIFICADA BIFASE M EN EL EJEMPLO DE LA TEORIA*)

Var
  i, j : Integer;
Begin
M := N*2; cp :=2;
If (bit[1] = '0') then
  Begin
  coef[1] := -1;
  coef[2] := -1;
  End
```

```
Else
  Begin
  coef[1] := 1;
  coef[2] := -1;
  End;

j := 3;
For i := 2 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    coef [j] := coef[j-1] * (-1);
    coef [j+1] := coef [j-1] * (-1);
    j := j + 2;
    End
  Else
    Begin
    coef[j] := coef[j-1] * (-1);
    coef[j+1] := coef[j-1];
    j := j + 2;
    End;
  End;
End;
(*************************************************)
Procedure BIFASES1;
(*ENCUENTRA LOS COEFICIENTES NECESARIOS PARA REALIZAR EL GRAFICO DE LA SEÑAL
  CODIFICADA BIFAFE S EN EL EJEMPLO DE LA TEORIA*)

Var
  i, j : Integer;
Begin
M := N*2;cp := 2;
If (bit[1] = '1') then
  Begin
  coef[1] := -1;
  coef[2] := -1;
  End
Else
  Begin
  coef[1] := -1;
  coef[2] := 1;
  End;

j := 3;
For i := 2 To N Do
  Begin
  If (bit[i] = '1') Then
    Begin
    coef [j] := coef[j-1] * (-1);
    coef [j+1] := coef [j-1] * -1;
    j := j + 2;
    End
  Else
    Begin
    coef[j] := coef[j-1] * (-1);
    coef[j+1] := coef[j-1];
    j := j + 2;
    End;
  End;
End;
(*************************************************)
Procedure MILLER1;
(*ENCUENTRA LOS COEFICIENTES NECESARIOS PARA REALIZAR EL GRAFICO DE LA SEÑAL
  CODIFICADA MILLER EN EL EJEMPLO DE LA TEORIA*)

Var
  i, j : Integer;
Begin
M := N*2;cp:=2;
If (bit[1] = '0') then
```

```
  Begin
  coef[1] := 1;
  coef[2] := 1;
  End
Else
  Begin
  coef[1] := 1;
  coef[2] := -1;
  End;

j := 3;
For i := 2 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    If (bit[i-1] = '0') Then
      Begin
      coef [j] := coef[j-1] * (-1);
      coef [j+1] := coef [j-1] * -1;
      j := j + 2;
      End
    Else
      Begin
      coef[j] := coef[j-1];
      coef[j+1] := coef[j-1];
      j := j + 2;
      End
    End
  Else
    Begin
    coef[j] := coef[j-1];
    coef[j+1] := coef[j-1] * -1;
    j := j + 2;
    End
  End;
End;
(************************************************)
Procedure CUATROB3T1(P : Integer);
(*ENCUENTRA LOS COEFICIENTES NECESARIOS PARA REALIZAR EL GRAFICO DE LA SEÑAL
  CODIFICADA 4B3T EN EL EJEMPLO DE LA TEORIA*)

Type
  MaxCad = String;
Var
  i, j, S, MM : Integer;
  ParidadInicial, ParidadAcum : Integer;
  alfa : Array[1..20]Of Char;
  ivan : Array[1..10]Of String;
  ex : MaxCad;
Procedure ParidadAcumulada;
Var
  word : Array[1..10] Of String;
  Begin
  word[j] := bit[i] + bit[i+1] + bit[i+2] + bit[i+3];
  If word[j] = '0000' Then
    alfa[j] := 'A'
  Else If word[j] = '0001' Then
    alfa[j] := 'B'
  Else If word[j] = '0010' Then
    alfa[j] := 'C'
  Else If word[j] = '1000' Then
    alfa[j] := 'D'
  Else If word[j] = '1001' Then
    alfa[j] := 'E'
  Else If word[j] = '1010' Then
    alfa[j] := 'F'
  Else If word[j] = '0011' Then
    alfa[j] := 'G'
  Else If word[j] = '1011' Then
    alfa[j] := 'H'
```

```
  Else If word[j] = '0101' Then
    alfa[j] := 'L'
  Else If word[j] = '0110' Then
    alfa[j] := 'R'
  Else If word[j] = '0111' Then
    alfa[j] := 'S'
  Else If word[j] = '1110' Then
    alfa[j] := 'T'
  Else If word[j] = '1100' Then
    alfa[j] := 'U'
  Else If word[j] = '1101' Then
    alfa[j] := 'V'
  Else If word[j] = '0100' Then
    alfa[j] := 'W'
  Else If word[j] = '1111' Then
    alfa[j] := 'X';

  case alfa[j] Of
  'A'..'F' : ParidadAcum := 0;
  'G'..'T' : ParidadAcum := 1;
  'U'..'W' : ParidadAcum := 2;
  'X' : ParidadAcum := 3;
  End;
End;
(*******************************************************)
Procedure ModoCero;
  Begin
  Case alfa[j] Of
  'A' : ivan[j] := '0 - +';
  'B' : ivan[j] := '- + 0';
  'C' : ivan[j] := '- 0 +';
  'D' : ivan[j] := '0 + -';
  'E' : ivan[j] := '+ - 0';
  'F' : ivan[j] := '+ 0 -';
  End;
End;

(*******************************************************)
Procedure ModoPositivo;
  Begin
  Case alfa[j] Of
  'G' : ivan[j] := '+ - +';
  'H' : ivan[j] := '+ 0 0';
  'L' : ivan[j] := '0 + 0';
  'R' : ivan[j] := '0 0 +';
  'S' : ivan[j] := '- + +';
  'T' : ivan[j] := '+ + -';
  'U' : ivan[j] := '+ 0 +';
  'V' : ivan[j] := '+ + 0';
  'W' : ivan[j] := '0 + +';
  'X' : ivan[j] := '+ + +';
  End;
End;
(*******************************************************)
Procedure ModoNegativo;
  Begin
  Case alfa[j] Of
  'G' : ivan[j] := '- + -';
  'H' : ivan[j] := '- 0 0';
  'L' : ivan[j] := '0 - 0';
  'R' : ivan[j] := '0 0 -';
  'S' : ivan[j] := '+ - -';
  'T' : ivan[j] := '- - +';
  'U' : ivan[j] := '- 0 -';
  'V' : ivan[j] := '- - 0';
  'W' : ivan[j] := '0 - -';
  'X' : ivan[j] := '- - -';
  End;
End;
(*******************************************************)
```

```
Procedure Uno;
  Begin
  If (ParidadAcum = -1) Then
    ParidadInicial := -1
  Else If (ParidadAcum = -2) Then
    ParidadInicial := -2
  Else If (ParidadAcum = -3) Then
    ParidadInicial := -3
  End;
(*****************************************************)
Procedure Dos;
  Begin
  If (ParidadAcum = -1) Then
    ParidadInicial := 1
  Else If (ParidadAcum = -2) Then
    ParidadInicial := -1
  Else If (ParidadAcum = -3) Then
    ParidadInicial := -2
  End;
(*****************************************************)
Procedure Tres;
  Begin
  If (ParidadAcum = -1) Then
    ParidadInicial := 2
  Else If (ParidadAcum = -2) Then
    ParidadInicial := 1
  Else If (ParidadAcum = -3) Then
    ParidadInicial := -1
  End;
(*****************************************************)
Procedure MenosUno;
  Begin
  If (ParidadAcum = 1) Then
    ParidadInicial := 1
  Else If (ParidadAcum = 2) Then
    ParidadInicial := 2
  Else If (ParidadAcum = 3) Then
    ParidadInicial := 3
  End;
(*****************************************************)
Procedure MenosDos;
  Begin
  If (ParidadAcum = 1) Then
    ParidadInicial := -1
  Else If (ParidadAcum = 2) Then
    ParidadInicial := 1
  Else If (ParidadAcum = 3) Then
    ParidadInicial := 2
  End;
(*****************************************************)
Procedure MenosTres;
  Begin
  If (ParidadAcum = 1) Then
    ParidadInicial := -2
  Else If (ParidadAcum = 2) Then
    ParidadInicial := -1
  Else If (ParidadAcum = 3) Then
    ParidadInicial := 1
  End;
(*****************************************************)
Procedure CambioEstado;
  Begin
  If (ParidadInicial = 1) Then
    Uno
  Else If (ParidadInicial = 2) Then
    Dos
  Else If (ParidadInicial = 3) Then
    Tres
  Else If (ParidadInicial = -1) Then
    MenosUno
```

```
  Else If (ParidadInicial = -2) Then
    MenosDos
  Else If (ParidadInicial = -3) Then
    MenosTres
  End;
(*****************************************************)
Begin
S := N div 4;
j := 1;
i := 1;
While j <= (N div 4) Do
  Begin
  ParidadAcumulada;
  If (ParidadAcum = 0) Then
    Begin
    ModoCero;
    j := j + 1;
    i := i + 4;
    End
  Else
    Begin
    If (ParidadInicial > 0) Then
      Begin
      ModoNegativo;
      CambioEstado;
      j := j + 1;
      i := i + 4;
      End
    Else
      Begin
      ParidadAcum := ParidadAcum * -1;
      ModoPositivo;
      CambioEstado;
      j := j + 1;
      i := i + 4;

      End
    End
  End;
ex := ivan[1] ;
For i := 2 To S Do
   ex := ex + ivan[i];

MM := Length(ex);
i := 1; j := 1;
While j <= MM Do
  Begin
    If ex[j] = '+' Then
      Begin
      coef[i] := 1;coef1[i] := '+';
      inc(i,1);
      End
    Else If ex[j] = '0' Then
      Begin
      coef[i] := 0;coef1[i] := '0';
      inc(i,1);
      End
    Else If ex[j] = '-' Then
      Begin
      coef[i] := -1;coef1[i] := '-';
      inc(i,1);
      End;
    inc(j,1);
    End;

M := MM div 2;cp:=1;
End;
(*************************************************)
Procedure CMI1;
(*ENCUENTRA LOS COEFICIENTES NECESARIOS PARA REALIZAR EL GRAFICO DE LA SEÑAL
```

```
  CODIFICADA CMI EN EL EJEMPLO DE LA TEORIA*)

Var
  i, j, COF : Integer;
Begin
M := N*2;cp := 2;
COF := -1;
j := 1;
For i := 1 To N Do
  Begin
  If (bit[i] = '0') Then
    Begin
    coef [j] := -1;
    coef [j+1] := +1;
    j := j + 2;
    End
  Else
    Begin
    COF := (COF * -1);
    coef[j] := COF;
    coef[j+1] := cof;
    j := j + 2;
    End;
  End;
End;
(*****************************************)
Procedure PST1(P : Integer);
(*ENCUENTRA LOS COEFICIENTES NECESARIOS PARA REALIZAR EL GRAFICO DE LA SEÑAL
  CODIFICADA PST EN EL EJEMPLO DE LA TEORIA*)

Var
  i, COF : Integer;
Begin
M := N;cp := 1;
If (P = 0) Then
  Begin
  COF := -1;
  i:= 1;
  End
Else
  Begin
  COF := 1;
  i := 1;
  End;

While (i <= N) Do
  Begin
  If (bit[i] = '0') Then
    Begin
    If (bit[i+1] = '0') Then
      Begin
      coef[i] := -1;
      coef[i+1] := 1;
      i := i + 2;
      End
    Else
      Begin
      COF := COF * (-1);
      coef[i] := 0;
      coef[i+1] := COF;
      i := i + 2;
      End
    End
  Else
    Begin
    If (bit[i+1] = '0') Then
      Begin
      COF := COF * (-1);
      coef[i] := COF;
      coef[i+1] := 0;
```

```
      i := i +2;
      End
    Else
      Begin
      coef[i] := 1;
      coef[i+1] := -1;
      i := i + 2;
      End
    End
  End;
End;
(*******************************************************)
Procedure GraficarEjemplo(h, k, X2, Y2, Y3 : integer);
(*GRAFICA LA SEÑAL DE DATOS Y LA SEÑAL CODIFICADA, LUEGO DE LA TEORIA
  CORRESPONDIENTE*)

Var
  i, X ,Y, X1, Y1 : Integer;
  numero : Array [1..10] Of String;
Begin
  X1 := GetMaxX; Y1 := GetMaxY;
  SetColor(15);
  MoveTo(X2,Y2);
  For i:=1 To N Do
    If bit[i] = '0' Then dato[i] := 0
    Else dato[i] := 1;
  i := 1;
  While i <= 2*N Do
    Begin
    reloj[i] := 1;reloj[i+1] := 0;
    inc(i,2);
    End;
  For i:=1 to N do
    begin
      lineto(X2+h*(i-1),Y2-20*dato[i]);
      lineto(X2+h*i,Y2-20*dato[i]);
    end;
  MoveTo(X2,Y3);
  For i:=1 to M do
    begin
      lineto(X2+k*(i-1),Y3-20*coef[i]);
      lineto(X2+k*i,Y3-20*coef[i]);
    end;
  MoveTo(X2,Y2-3);
  For i:= 1 To N+1 Do
    Begin
      LineRel(0,3); X := GetX;
      X := X + h; Y := Y2-3;
      MoveTo(X,Y);
    End;
  MoveTo(X2,Y3-3);
  For i:= 1 To N+1 Do
    Begin
      LineRel(0,3); X := GetX;
      X := X + h; Y := Y3-3;
      MoveTo(X,Y);
    End;
  For i:= 1 To N Do
    numero[i] := bit[i];
  SetTextStyle(2,0,3);
  MoveTo(X2+(h Div 2),Y2-10);
  For i:= 1 To N Do
    Begin
      X:=GetX; Y:=GetY;
      OutTextXY(X,Y,numero[i]);
      X := X + h;
      MoveTo(X,Y);
    End;
  SetTextStyle(2,0,4);
  OutTextXY(X2+5,Y2-35,'SEÑAL BINARIA');
```

```
  OutTextXY(X2+5,Y3-35,'SEÑAL CODIFICADA');
  OutTextXY(X2-24,Y3-25,'+ A');OutTextXY(X2-12,Y3-4,'0');
  If (Tecla <> 'A') and (Tecla <> 'a') and
    (Tecla <> 'B') and (Tecla <> 'b') Then
    OutTextXY(X2-24,Y3+14,'- A');
  Opciones(X1, Y1);
  Closegraph;
End;
(*******************************************************)
Procedure GrafEjmCB3T(h, k, X2, Y2, Y3 : integer);
(*GRAFICA EL EJEMPLO DE CODIFICACION DEL CODIGO 4B3T, LUEGO DE LA RESPECTIVA
  TEORIA*)

Var
  i, X ,Y, X1, Y1 : Integer;
  numero : Array [1..10] Of String;
Begin
  X1 := GetMaxX; Y1 := GetMaxY;
  SetColor(15);
  MoveTo(X2,Y2);
  For i:=1 To N Do
    If bit[i] = '0' Then dato[i] := 0
    Else dato[i] := 1;
  While i <= 2*N Do
    Begin
    reloj[i] := 1;reloj[i+1] := 0;
    inc(i,2);
    End;
  For i:=1 to N do
    begin
      lineto(X2+h*(i-1),Y2-20*dato[i]);
      lineto(X2+h*i,Y2-20*dato[i]);
    end;
  MoveTo(X2,Y3);
  For i:=1 to (N Div 4)*3 do
    begin
      lineto(X2+k*(i-1),Y3-20*coef[i]);
      lineto(X2+k*i,Y3-20*coef[i]);
    end;

  MoveTo(X2,Y2-3);
  For i:= 1 To N+1 Do
    Begin
      LineRel(0,3); X := GetX;
      X := X + h; Y := Y2-3;
      MoveTo(X,Y);
    End;

  MoveTo(X2,Y2+6);
  For i:= 1 To (M Div 3)+1  Do
    Begin
    LineRel(0,10); X := GetX;
    X := X + 4*h; Y := Y2+6;
    MoveTo(X,Y);
    End;

  MoveTo(X2,Y3-3);
  For i:= 1 To M+1 Do
    Begin
    LineRel(0,3); X := GetX;
    X := X + k; Y := Y3-3;
    MoveTo(X,Y);
    End;

  For i:= 1 To N Do
    numero[i] := bit[i];

  SetTextStyle(2,0,3);
  MoveTo(X2 + (h Div 2),Y2+12);
  For i:= 1 To N Do
```

```
    Begin
    X:=GetX; Y:=GetY;
    OutTextXY(X,Y,numero[i]);
    X := X + h;
    MoveTo(X,Y);
    End;

  MoveTo(X2+(k Div 2),Y3+32);
  For i:= 1 To ((N Div 4)*3) Do
    Begin
    X:=GetX; Y:=GetY;
    OutTextXY(X,Y,coef1[i]);
    X := X + k;
    MoveTo(X,Y);
    End;

  SetTextStyle(2,0,4);
  OutTextXY(X2+5,Y2-35,'SEÑAL BINARIA');
  OutTextXY(X2+5,Y3-35,'SEÑAL CODIFICADA');
  OutTextXY(X2-24,Y3-25,'+ A');OutTextXY(X2-12,Y3-4,'0');
  OutTextXY(X2-24,Y3+14,'- A');
End;
(*****************************************************)
Procedure NRZTeoria;
(*COLOCA EL TEXTO LA TEORIA DEL CODIGO NRZ, Y LLAMA A LOS PROCEDIMIENTOS
  ENCARGADOS DE GRAFICAR EL EJEMPLO DE LA CODIFICACION*)
Var
  h, i, k, X1, Y1, Y2, X2, Y3, Y4 : Integer;
  Texto:Array[1..15] of string;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  SetColor(14);SetBkColor(1);
  X1:=GetMaxX; Y1:=GetMaxY; Y3:=Y1 Div 24;
  Rectangle(0,0,X1,Y1);
  SetTextStyle(2,0,6); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,10,'C O D I G O  L I N E A L  N R Z');
  SetTextStyle(2,0,5); SetTextJustify(0,2);
  Texto[1]:='';
  Texto[2]:='Es el más sencillo de los códigos. A cada símbolo binario
0L o 1L se le';
  Texto[3]:='asigna uno de los niveles 0 o A. ';
  Texto[4]:='Este tipo de codificación tiene dificultad para controlar una
señal horaria';
  Texto[5]:='utilizada para la regeneración del tren de pulsos
binarios.';
  Texto[6]:='Esta es la razón por la cual la codificación NRZ no es usada
en transmisión';
  Texto[7]:='de datos a larga distancia.';
  Texto[8]:='';
  Texto[9]:='E J E M P L O :';
  Texto[10]:='';
  Texto[11]:='Codificación de la secuencia de bits : 1 0 1 1 0 1 1 0 1 1';
  X2:= (x1 - 580) Div 2;
  MoveTo(X2,30);X2:=GetX;Y2:=GetY;
  For i:=1 to 11 Do
    Begin
    OutTextXY(X2,Y2,Texto[i]);
    Y2:=Y2+Y3;
    End;
  X2:=(X1-300) Div 2; Y4:=(Y1-11*Y3-60) Div 3;
  Y2:= Y1-2*Y4-30; Y3:=Y2+Y4;
  N:=10;
  bit[1]:='1';bit[2]:='0';bit[3]:='1';
  bit[4]:='1';bit[5]:='0';bit[6]:='1';
  bit[7]:='1';bit[8]:='0';bit[9]:='1';bit[10]:='1';
  NRZ1;
  GraficarEjemplo(30,30,X2,Y2,Y3);
Closegraph;
End;
```

```
(*************************************************)
Procedure RzTeoria;
(*COLOCA EL TEXTO LA TEORIA DEL CODIGO RZ, Y LLAMA A LOS PROCEDIMIENTOS
  ENCARGADOS DE GRAFICAR EL EJEMPLO DE LA CODIFICACION*)

Var
  h, i, k, X1, Y1, Y2, X2, Y3, Y4 : Integer;
  Texto:Array[1..15] of string;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  SetColor(14);SetBkColor(1);
  X1:=GetMaxX; Y1:=GetMaxY; Y3:=Y1 Div 24;Y4:=Y1 Div 12;
  Rectangle(0,0,X1,Y1);
  SetTextStyle(2,0,6); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,10,'C O D I G O  L I N E A L  R Z');
  SetTextStyle(2,0,5);SetTextjustify(0,2);
  Texto[1]:='';
  Texto[2]:='Es un código de retorno a cero, cuando existe un 1L de
amplitud A, vuelve a';
  Texto[3]:='cero en una parte del intervalo To. Si el tiempo de retorno a
cero correspon-';
  Texto[4]:='de a la mitad del período To/2 , se dice que se trata de un
código RZ al 50%.';
  Texto[5]:=' ';
  Texto[6]:='Es utilizado para simplificar la recuperación de la señal de
reloj. El ancho';
  Texto[7]:='de banda correspondiente a este código es el doble del NRZ,
para un ciclo de';
  Texto[8]:='trabajo del 50%.';
  Texto[9]:=' ';
  Texto[10]:='E J E M P L O :';
  Texto[11]:=' ';
  Texto[12]:='Codificación de la secuencia de bits : 1 0 1 1 0 1 1 0 1 1';
  X2:= (x1 - 600) Div 2;
  MoveTo(X2,30);X2:=GetX;Y2:=GetY;
  For i:=1 to 12 do
    Begin
    OutTextXY(X2,Y2,Texto[i]);
    Y2:=Y2+Y3;
    End;
  N:=10;
  X2:=(X1-300) Div 2; Y4:=(Y1-12*Y3-60) Div 3;
  Y2:= Y1-2*Y4-30; Y3:=Y2+Y4;
  bit[1]:='1';bit[2]:='0';bit[3]:='1';
  bit[4]:='1';bit[5]:='0';bit[6]:='1';
  bit[7]:='1';bit[8]:='0';bit[9]:='1';bit[10]:='1';
  RZ1;
  GraficarEjemplo(30,15,X2,Y2,Y3);
Closegraph;
End;
(*************************************************)
Procedure NRZPOLARTeoria;
(*COLOCA EL TEXTO LA TEORIA DEL CODIGO NRZ POLAR, Y LLAMA A LOS
  PROCEDIMIENTOS ENCARGADOS DE GRAFICAR EL EJEMPLO DE LA CODIFICACION*)

Var
  h, i, k, X1, Y1, Y2, X2, Y3, Y4 : Integer;
  Texto:Array[1..15] of string;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  SetColor(14);SetBkColor(1);
  X1:=GetMaxX; Y1:=GetMaxY; Y3:=Y1 Div 22;Y4:=Y1 Div 12;
  Rectangle(0,0,X1,Y1);
  SetTextStyle(2,0,6); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,10,'C O D I G O  L I N E A L  N R Z  P O L A R');
  SetTextStyle(2,0,5); SetTextJustify(0,2);
  Texto[1]:='';
  Texto[2]:='Es un código igual que el NRZ, sin retorno a cero, en este
```

```
caso se  asigna';
  Texto[3]:='al símbolo 1L un nivel de señal +A y al símbolo 0L un nivel -A
.';
  Texto[4]:='';
  Texto[5]:='Se lo denomina código POLAR porque su codificación varía entre
dos niveles';
  Texto[6]:='de señal, +A y -A.';
  Texto[7]:='';
  Texto[8]:='E J E M P L O :';
  Texto[9]:=' ';
  Texto[10]:='Codificación de la secuencia de bits : 1 0 1 1 0 1 1 0 1 1 ';
  X2:= (x1 - 580) Div 2;
  MoveTo(X2,30);X2:=GetX;Y2:=GetY;
  For i:=1 to 10 do
    Begin
      OutTextXY(X2,Y2,Texto[i]);
      Y2:=Y2+Y3;
    End;
  N:=10;
  X2:=(X1-300) Div 2; Y4:=(Y1-10*Y3-60) Div 3;
  Y2:= Y1-2*Y4-30; Y3:=Y2+Y4;
  bit[1]:='1';bit[2]:='0';bit[3]:='1';
  bit[4]:='1';bit[5]:='0';bit[6]:='1';
  bit[7]:='1';bit[8]:='0';bit[9]:='1';bit[10]:='1';
  NRZPOLAR1;
  GraficarEjemplo(30,30,X2,Y2,Y3);
Closegraph;
End;
(*****************************************************)
Procedure RZPOLARTeoria;
(*COLOCA EL TEXTO LA TEORIA DEL CODIGO RZ POLAR, Y LLAMA A LOS
  PROCEDIMIENTOS ENCARGADOS DE GRAFICAR EL EJEMPLO DE LA CODIFICACION*)

Var
  h, i, k, X1, Y1, Y2, X2, Y3,Y4 : Integer;
    Texto:Array[1..15] of string;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  SetColor(14);SetBkColor(1);
  X1:=GetMaxX; Y1:=GetMaxY; Y3:=Y1 Div 22;Y4:=Y1 Div 12;
  Rectangle(0,0,X1,Y1);
  SetTextStyle(2,0,6); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,10,'C O D I G O   L I N E A L   R Z   P O L A R');
  SetTextStyle(2,0,5); SetTextJustify(0,2);
  Texto[1]:='';
  Texto[2]:='Es un código igual al RZ, con retorno a cero, en el cual se
asigna un nivel';
  Texto[3]:='de señal +A al símbolo 1L y un nivel -A al símbolo 0L.  Para
el caso  de RZ';
  Texto[4]:='Polar al 50% el pulso vuelve a cero a la mitad del intervalo
To.';
  Texto[5]:='';
  Texto[6]:='El ancho de banda de este código es el doble del NRZ Polar.';
  Texto[7]:='';
  Texto[8]:='E J E M P L O :';
  Texto[9]:=' ';
  Texto[10]:='Codificación de la secuencia de bits : 1 0 1 1 0 1 1 0 1 1';
  X2:= (x1 - 580) Div 2;
  MoveTo(X2,30);X2:=GetX;Y2:=GetY;
  For i:=1 to 10 do
    Begin
    OutTextXY(X2,Y2,Texto[i]);
    Y2:=Y2+Y3;
    End;
  N:=10;
  X2:=(X1-300) Div 2; Y4:=(Y1-10*Y3-60) Div 3;
  Y2:= Y1-2*Y4-30; Y3:=Y2+Y4;
  bit[1]:='1';bit[2]:='0';bit[3]:='1';
  bit[4]:='1';bit[5]:='0';bit[6]:='1';
```

```
  bit[7]:='1';bit[8]:='0';bit[9]:='1';bit[10]:='1';
  RZPOLAR1;
  GraficarEjemplo(30,15,X2,Y2,Y3);
Closegraph;
End;
(***********************************************)
Procedure AmiTeoria;
(*COLOCA EL TEXTO LA TEORIA DEL CODIGO AMI, Y LLAMA A LOS PROCEDIMIENTOS
  ENCARGADOS DE GRAFICAR EL EJEMPLO DE LA CODIFICACION*)

Var
  h, i, k, X1, Y1, Y2, X2, Y3, Y4 : Integer;
  Texto:Array[1..15] of string;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  SetColor(14);SetBkColor(1);
  X1:=GetMaxX; Y1:=GetMaxY; Y3:=Y1 Div 22;Y4:=Y1 Div 12;
  Rectangle(0,0,X1,Y1);
  SetTextStyle(2,0,6); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,10,'C O D I G O   L I N E A L   A M I');
  SetTextStyle(2,0,5); SetTextJustify(0,2);
  Texto[1]:='';
  Texto[2]:='Este código asigna a cada pulso binario 1L con un nivel
positivo o  negativo';
  Texto[3]:='(marcas) en forma alternada mientras que la condición de cero
lógico se la';
  Texto[4]:='transmite  con ausencia de marca o nivel cero. Las marcas
pueden  ser  NRZ,';
  Texto[5]:='o RZ dependiendo del pulso escogido.';
  Texto[6]:='Este es el código más simple que satisface los requerimientos
de los códigos';
  Texto[7]:='de línea.';
  Texto[8]:='';
  Texto[9]:='E J E M P L O :';
  Texto[10]:='Codificación de la secuencia de bits : 1 0 1 1 0 1 1 0 1 1 ';
  Texto[11]:='';
  X2:= (x1 - 580) Div 2;
  MoveTo(X2,30);X2:=GetX;Y2:=GetY;
  For i:=1 to 11 do
    Begin
      OutTextXY(X2,Y2,Texto[i]);
      Y2:=Y2+Y3;
    End;
  N:=10;
  X2:=(X1-300) Div 2; Y4:=(Y1-11*Y3-60) Div 3;
  Y2:= Y1-2*Y4-40; Y3:=Y2+Y4;
  bit[1]:='1';bit[2]:='0';bit[3]:='1';
  bit[4]:='1';bit[5]:='0';bit[6]:='1';
  bit[7]:='1';bit[8]:='0';bit[9]:='1';bit[10]:='1';
  AMI1;
  GraficarEjemplo(30,30,X2,Y2,Y3);
Closegraph;
End;
(***********************************************)
Procedure BnZSTeoria;
(*COLOCA EL TEXTO LA TEORIA DEL CODIGO BNZS, Y LLAMA A LOS PROCEDIMIENTOS
  ENCARGADOS DE GRAFICAR EL EJEMPLO DE LA CODIFICACION*)

Var
  h, i, k, X1, Y1, Y2, X2, Y3, Y4 : Integer;
  Texto:Array[1..17] of string;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  SetColor(14);SetBkColor(1);
  X1:=GetMaxX; Y1:=GetMaxY; Y3:=Y1 Div 32;Y4:=Y1 Div 12;
  Rectangle(0,0,X1,Y1);
  SetTextStyle(2,0,6); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,10,'C O D I G O   L I N E A L   B 3 Z S');
```

```
  SetTextStyle(2,0,5); SetTextJustify(0,2);
  Texto[1]:='';
  Texto[2]:='El código B3ZS  codifica 3  ceros por un código de longitud
3, por lo  que';
  Texto[3]:='codifica reemplazando el tercer cero consecutivo con un pulso
que es trans-';
  Texto[4]:='mitido con igual polaridad al pulso precedente (violación).';
  Texto[5]:='Para conservar una componente c. c. nula se deben transmitir
tantas  viola-';
  Texto[6]:='ciones positivas como negativas alternadamente.';
  Texto[7]:='La regla para codificar es la siguiente.';
  Texto[8]:='';
  Texto[9]:='                POLARIDAD DEL       NUMERO DE UNOS DESDE';
  Texto[10]:='                   PULSO          LA ULTIMA SUSTITUCION';
  Texto[11]:='                PRECEDENTE           IMPAR         PAR';
  Texto[12]:='                    -                0 0 -       + 0 +';
  Texto[13]:='                    +                0 0 +       - 0 -';
  Texto[14]:='';
  Texto[15]:='E J E M P L O :';
  Texto[16]:='';
  Texto[17]:='Codificación de la secuencia de bits : 1 0 1 0 0 0 1 1 0 0 0
0 0 1';
  X2:= (x1 - 580) Div 2;
  MoveTo(X2,30);X2:=GetX;Y2:=GetY;
  For i:=1 to 17 do
  Begin
    OutTextXY(X2,Y2,Texto[i]);
    Y2:=Y2+Y3;
    End;
  N:=14;
  X2:=(X1-300) Div 2; Y4:=(Y1-17*Y3-60) Div 3;
  Y2:= Y1-2*Y4-30; Y3:=Y2+Y4;
  bit[1]:='1';bit[2]:='1';bit[3]:='0';
  bit[4]:='0';bit[5]:='0';bit[6]:='0';
  bit[7]:='1';bit[8]:='1';bit[9]:='0';bit[10]:='0';
  bit[11]:='0';bit[12]:='0';bit[13]:='0';bit[14]:='1';
 BNZS1(1);
 SetTextStyle(2,0,4);
 MoveTo(0,y1-30);LineTo(x1,y1-30);SetTextjustify(1,2);
 OutTextXY(X1 Div 2,Y1-20,' ESC : Menú Anterior  F1 : Menú Códigos  END :
Salir al DOS ');
 GraficarEjemplo(18, 18, X2, Y2, Y3);
Closegraph;
End;
(************************************************)
Procedure HDB3Teoria;
(*COLOCA EL TEXTO LA TEORIA DEL CODIGO HDB3, Y LLAMA A LOS PROCEDIMIENTOS
  ENCARGADOS DE GRAFICAR EL EJEMPLO DE LA CODIFICACION*)
Var
  h, i, k, X1, Y1, Y2, X2, Y3, Y4 : Integer;
  Texto:Array[1..17] of string;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  SetColor(14);SetBkColor(1);
  X1:=GetMaxX; Y1:=GetMaxY; Y3:=Y1 Div 32;Y4:=Y1 Div 12;
  Rectangle(0,0,X1,Y1);
  SetTextStyle(2,0,6); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,10,'C O D I G O   L I N E A L   H D B 3');
  SetTextStyle(2,0,5); SetTextJustify(0,2);
  Texto[1]:='';
  Texto[2]:='Este código no acepta más de 3 ceros consecutivos, por esta
razón  codifica';
  Texto[3]:='reemplazando el cuarto cero consecutivo con un pulso que es
transmitido con';
  Texto[4]:='igual polaridad al pulso precedente (violación).';
  Texto[5]:='Para conservar una componente  DC  nula  se deben transmitir
tantas  viola-';
  Texto[6]:='ciones positivas como negativas alternadamente.';
  Texto[7]:='La regla para codificar es la siguiente:';
```

```
  Texto[8]:='';
  Texto[9]:='                          POLARIDAD DEL      NUMERO DE UNOS DESDE';
  Texto[10]:='                            PULSO          LA ULTIMA SUSTITUCION';
  Texto[11]:='                            PRECEDENTE        IMPAR         PAR';
  Texto[12]:='                               -            0 0 0 -      + 0 0 +';
  Texto[13]:='                               +            0 0 0 +      - 0 0 -';
  Texto[14]:='';
  Texto[15]:='E J E M P L O :';
  Texto[16]:='';
  Texto[17]:='Codificación de la secuencia de bits : 1 1 0 0 0 0 1 1 0 0 0
0 0 1';
  X2:= (x1 - 580) Div 2;
  MoveTo(X2,30);X2:=GetX;Y2:=GetY;
  For i:=1 to 17 do
    Begin
    If (i > 8) and (i <14) then SetTextstyle(2,0,4)
    else SetTextstyle(2,0,5);
    OutTextXY(X2,Y2,Texto[i]);
    Y2:=Y2+Y3;
    End;
  N:=14;
  X2:=(X1-300) Div 2; Y4:=(Y1-17*Y3-60) Div 3;
  Y2:= Y1-2*Y4-30; Y3:=Y2+Y4;
  bit[1]:='1';bit[2]:='1';bit[3]:='0';
  bit[4]:='0';bit[5]:='0';bit[6]:='0';
  bit[7]:='1';bit[8]:='1';bit[9]:='0';bit[10]:='0';
  bit[11]:='0';bit[12]:='0';bit[13]:='0';bit[14]:='1';
  HDB31;
  SetTextStyle(2,0,4);
  MoveTo(0,y1-30);LineTo(x1,y1-30);SetTextjustify(1,2);
  OutTextXY(X1 Div 2,Y1-20,' ESC : Menú Anterior  F1 : Menú Códigos  END :
Salir al DOS ');
  GraficarEjemplo(18,18,X2,Y2,Y3);
Closegraph;
End;
(************************************************)
Procedure ManchesterTeoria;
(*COLOCA EL TEXTO LA TEORIA DEL CODIGO MANCHESTER, Y LLAMA A LOS
  PROCEDIMIENTOS ENCARGADOS DE GRAFICAR EL EJEMPLO DE LA CODIFICACION*)

Var
  h, i, k, X1, Y1, Y2, X2, Y3,Y4 : Integer;
  Texto:Array[1..15] of string;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  SetColor(14);SetBkColor(1);
  X1:=GetMaxX; Y1:=GetMaxY; Y3:=Y1 Div 22;Y4:=Y1 Div 12;
  Rectangle(0,0,X1,Y1);
  SetTextStyle(2,0,6); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,10,'C O D I G O  L I N E A L  M A N C H E S T E R');
  SetTextStyle(2,0,5); SetTextJustify(0,2);
  Texto[1]:='';
  Texto[2]:='En este código se producen transiciones positivas y negativas
entre +A';
  Texto[3]:='y -A en la mitad del intervalo unitario To, el pulso binario
1L tiene una';
  Texto[4]:='transición prefijada y símbolo 0L la opuesta.';
  Texto[5]:='Este código tiene una mayor facilidad para recuperar la señal
de reloj y ';
  Texto[6]:='y evita la componente DC, pero con el aumento del ancho de
banda .';
  Texto[7]:='En este caso se ha asignado a 1L la transición negativa.';
  Texto[8]:='';
  Texto[9]:='E J E M P L O :';
  Texto[10]:='Codificación de la secuencia de bits : 1 0 1 1 0 1 1 0 1 1';
  Texto[11]:='';
  X2:= (x1 - 580) Div 2;
  MoveTo(X2,30);X2:=GetX;Y2:=GetY;
  For i:=1 to 11 do
```

```
    Begin
    OutTextXY(X2,Y2,Texto[i]);
    Y2:=Y2+Y3;
    End;

  N:=10;
  X2:=(X1-300) Div 2; Y4:=(Y1-11*Y3-60) Div 3;
  Y2:= Y1-2*Y4-30; Y3:=Y2+Y4;
  bit[1]:='1';bit[2]:='0';bit[3]:='1';
  bit[4]:='1';bit[5]:='0';bit[6]:='1';
  bit[7]:='1';bit[8]:='0';bit[9]:='1';bit[10]:='1';
  MANCHESTER1;
  GraficarEjemplo(30,15,X2,Y2,Y3);
Closegraph;
End;
(***********************************************)
Procedure BifaseMTeoria;
(*COLOCA EL TEXTO LA TEORIA DEL CODIGO BIFASEM, Y LLAMA A LOS PROCEDIMIENTOS
  ENCARGADOS DE GRAFICAR EL EJEMPLO DE LA CODIFICACION*)

Var
  h, i, k, X1, Y1, Y2, X2, Y3, Y4 : Integer;
  Texto:Array[1..15] of string;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  SetColor(14);SetBkColor(1);
  X1:=GetMaxX; Y1:=GetMaxY; Y3:=Y1 Div 22;Y4:=Y1 Div 12;
  Rectangle(0,0,X1,Y1);
  SetTextStyle(2,0,6); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,10,'C O D I G O  L I N E A L  B I F A S E  M');
  SetTextStyle(2,0,5); SetTextJustify(0,2);
  Texto[1]:='';
  Texto[2]:='Es un código que varía entre dos niveles +A y -A.  Al inicio
de cada inter-';
  Texto[3]:='valo se  produce una transición,  además el pulso binario  1L
produce otra ';
  Texto[4]:='transición medio intervalo To después.';
  Texto[5]:='El símbolo 0L no produce transición a mitad del intervalo y
se alterna en-';
  Texto[6]:='tre los niveles +A y -A.';
  Texto[7]:='Este código contrario al Código Bifase S. ';
  Texto[8]:='';
  Texto[9]:='E J E M P L O :';
  Texto[10]:='';
  Texto[11]:='Codificación de la secuencia de bits : 1 0 1 1 0 1 1 0 1 1';
  X2:= (x1 - 580) Div 2;
  MoveTo(X2,30);X2:=GetX;Y2:=GetY;
  For i:=1 to 11 do
    Begin
    OutTextXY(X2,Y2,Texto[i]);
    Y2:=Y2+Y3;
    End;

  N:=10;
  X2:=(X1-300) Div 2; Y4:=(Y1-11*Y3-60) Div 3;
  Y2:= Y1-2*Y4-30; Y3:=Y2+Y4;
  bit[1]:='1';bit[2]:='0';bit[3]:='1';
  bit[4]:='1';bit[5]:='0';bit[6]:='1';
  bit[7]:='1';bit[8]:='0';bit[9]:='1';bit[10]:='1';
  BIFASEM1;
  GraficarEjemplo(30,15,X2,Y2,Y3);
Closegraph;
End;
(***********************************************)
Procedure BifaseSTeoria;
(*COLOCA EL TEXTO LA TEORIA DEL CODIGO BIFASE S, Y LLAMA A LOS
  PROCEDIMIENTOS ENCARGADOS DE GRAFICAR EL EJEMPLO DE LA CODIFICACION*)

Var
```

```
  h, i, k, X1, Y1, Y2, X2, Y3, Y4 : Integer;
  Texto:Array[1..15] of string;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  SetColor(14);SetBkColor(1);
  X1:=GetMaxX; Y1:=GetMaxY; Y3:=Y1 Div 22;Y4:=Y1 Div 12;
  Rectangle(0,0,X1,Y1);
  SetTextStyle(2,0,6); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,10,'C O D I G O  L I N E A L  B I F A S E  S');
  SetTextStyle(2,0,5); SetTextJustify(0,2);
  Texto[1]:='';
  Texto[2]:='Es un código que varía entre dos niveles +A y -A.  Al inicio
de cada inter-';
  Texto[3]:='valo se produce una transición,además el símbolo 0L produce
otra transición';
  Texto[4]:='medio intervalo después.';
  Texto[5]:='El símbolo 1L no produce transición a mitad del intervalo y se
alterna en-';
  Texto[6]:='tre los niveles +A y -A.';
  Texto[7]:='Este código contrario al Código Bifase M. ';
  Texto[8]:='';
  Texto[9]:='E J E M P L O :';
  Texto[10]:='';
  Texto[11]:='Codificación de la secuencia de bits : 1 0 1 1 0 1 1 0 1 1';
  X2:= (x1 - 580) Div 2;
  MoveTo(X2,30);X2:=GetX;Y2:=GetY;
  For i:=1 to 11 do
    Begin
    OutTextXY(X2,Y2,Texto[i]);
    Y2:=Y2+Y3;
    End;

  N:=10;
  X2:=(X1-300) Div 2; Y4:=(Y1-11*Y3-60) Div 3;
  Y2:= Y1-2*Y4-30; Y3:=Y2+Y4;
  bit[1]:='1';bit[2]:='0';bit[3]:='1';
  bit[4]:='1';bit[5]:='0';bit[6]:='1';
  bit[7]:='1';bit[8]:='0';bit[9]:='1';bit[10]:='1';
  BIFASES1;
  GraficarEjemplo(30,15,X2,Y2,Y3);
Closegraph;
End;
(***********************************************)
Procedure MillerTeoria;
(*COLOCA EL TEXTO LA TEORIA DEL CODIGO MILLER, Y LLAMA A LOS PROCEDIMIENTOS
  ENCARGADOS DE GRAFICAR EL EJEMPLO DE LA CODIFICACION*)

Var
  h, i, k, X1, Y1, Y2, X2, Y3,Y4 : Integer;
  Texto:Array[1..15] of string;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  SetColor(14);SetBkColor(1);
  X1:=GetMaxX; Y1:=GetMaxY; Y3:=Y1 Div 22;Y4:=Y1 Div 12;
  Rectangle(0,0,X1,Y1);
  SetTextStyle(2,0,6); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,10,'C O D I G O  L I N E A L  M I L L E R');
  SetTextStyle(2,0,5); SetTextJustify(0,2);
  Texto[1]:='';
  Texto[2]:='En este código se producen transiciones entre los niveles +A
y -A.';
  Texto[3]:='El símbolo 1L produce una transición a la mitad del intervalo
To,  el sím-';
  Texto[4]:='bolo 0L no produce ninguna transición siempre y cuando no vaya
precedido de';
  Texto[5]:='otro OL, es este caso produce transición entre ambos ceros,
al terminar el';
  Texto[6]:='primer intervalo unitario To.';
```

```
  Texto[7]:='Se lo llama también CODIGO DE MODULACION POR RETARDO.';
  Texto[8]:='';
  Texto[9]:='E J E M P L O :';
  Texto[10]:='';
  Texto[11]:='Codificación de la secuencia de bits : 0 1 1 0 1 0 0 1 1 0 ';
  X2:= (x1 - 580) Div 2;
  MoveTo(X2,30);X2:=GetX;Y2:=GetY;
  For i:=1 to 11 do
    Begin
    OutTextXY(X2,Y2,Texto[i]);
    Y2:=Y2+Y3;
    End;

  N:=10;
  X2:=(X1-300) Div 2; Y4:=(Y1-11*Y3-60) Div 3;
  Y2:= Y1-2*Y4-30; Y3:=Y2+Y4;
  bit[1]:='0';bit[2]:='1';bit[3]:='1';
  bit[4]:='0';bit[5]:='1';bit[6]:='0';
  bit[7]:='0';bit[8]:='1';bit[9]:='1';bit[10]:='0';
  MILLER1;
  GraficarEjemplo(30,15,X2,Y2,Y3);
Closegraph;
End;
(***********************************************)
Procedure C4B3TTeoria;
(*COLOCA EL TEXTO LA TEORIA DEL CODIGO 4B3T, Y LLAMA A LOS PROCEDIMIENTOS
  ENCARGADOS DE GRAFICAR EL EJEMPLO DE LA CODIFICACION*)
Var
  h, i, k, X, Y, X1, Y1, Y2, X2, Y3, Y4, P : Integer;
  Texto:Array[1..15] of string;
  Numero : Array[1..25] Of String;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  SetColor(14);SetBkColor(1);
  X1:=GetMaxX; Y1:=GetMaxY; Y3:=Y1 Div 22;Y4:=Y1 Div 12;
  Rectangle(0,0,X1,Y1);
  SetTextStyle(2,0,6); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,10,'C O D I G O  L I N E A L  4 B 3 T ');
  SetTextStyle(2,0,5); SetTextJustify(0,2);
  Texto[1]:='';
  Texto[2]:='A diferencia de la codificación bipolar en este código se
hacen correspon-';
  Texto[3]:='der tres señales ternarias a cuatro señales binarias.';
  Texto[4]:='Los cuatro dígitos binarios representan 16 palabras de código,
las cuales';
  Texto[5]:='pueden ser codificadas en 27 posibles combinaciones ternarias.
El codifica-';
  Texto[6]:='dor selecciona el modo apropiado, de acuerdo a la señal
existente y de';
  Texto[7]:='acuerdo al diagrama de estados del código, seis estados son
especificados';
  Texto[8]:='equivalentes a los valores de paridad -3,-2,-1,+1,+2,+3.';
  Texto[9]:='';
  Texto[10]:='E J E M P L O :';
  Texto[11]:='Codificación de la secuencia : 0 0 0 0 0 1 1 0 1 0 0 0 0 1 1
1 1 0 1 1';

  X2:= (x1 - 580) Div 2;
  MoveTo(X2,30);X2:=GetX;Y2:=GetY;
  For i:=1 to 11 do
    Begin
    OutTextXY(X2,Y2,Texto[i]);
    Y2:=Y2+Y3;
    End;

  N:=20; P:=1;
  X2:=(X1-300) Div 2; Y4:=(Y1-11*Y3-60) Div 3;
  Y2:= Y1-2*Y4-30; Y3:=Y2+Y4;
  bit[1]:='0';bit[2]:='0';bit[3]:='0';
```

```
  bit[4]:='0';bit[5]:='0';bit[6]:='1';
  bit[7]:='1';bit[8]:='0';bit[9]:='1';bit[10]:='0';
  bit[11]:='0';bit[12]:='0';bit[13]:='0';bit[14]:='1';
  bit[15]:='1';bit[16]:='1';bit[17]:='1';bit[18]:='0';
  bit[19]:='1';bit[20]:='1';
  h:=18; k:=24;
  X2:=(X1-400) Div 2;
  CUATROB3T1(P);
  GrafEjmCB3T(h,k,X2,Y2,Y3);
  Opciones(X1,Y1);
Closegraph;
End;
(************************************************)
Procedure CmiTeoria;
(*COLOCA EL TEXTO LA TEORIA DEL CODIGO CMI, Y LLAMA A LOS PROCEDIMIENTOS
  ENCARGADOS DE GRAFICAR EL EJEMPLO DE LA CODIFICACION*)
Var
  h, i, k, X1, Y1, Y2, X2, Y3, Y4 : Integer;
  Texto:Array[1..15] of string;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  SetColor(14);SetBkColor(1);
  X1:=GetMaxX; Y1:=GetMaxY; Y3:=Y1 Div 22;Y4:=Y1 Div 12;
  Rectangle(0,0,X1,Y1);
  SetTextStyle(2,0,6); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,10,'C O D I G O  L I N E A L  C M I ');
  SetTextStyle(2,0,5); SetTextJustify(0,2);
  texto[1]:='';
  Texto[2]:='En este código el símbolo 0L produce una transición
positiva a medio';
  Texto[3]:='intervalo To. El símbolo 1L no produce transición y se
alterna entre los ';
  Texto[4]:='niveles +A y -A por lo que no genera componente DC.';
  Texto[5]:='Este código es equivalente a uno de velocidad doble donde
el 1L se halla ';
  Texto[6]:='alternadamente con 11 y 00 , y el símbolo 0L con 01.';
  Texto[7]:='';
  Texto[8]:='E J E M P L O :';
  Texto[9]:=' ';
  Texto[10]:='Codificación de la secuencia de bits : 0 1 1 0 1 0 0
1 1 0 ';
  X2:= (x1 - 580) Div 2;
  MoveTo(X2,30);X2:=GetX;Y2:=GetY;
  For i:=1 to 10 do
    Begin
    OutTextXY(X2,Y2,Texto[i]);
    Y2:=Y2+Y3;
    End;

  N:=10;
  X2:=(X1-300) Div 2; Y4:=(Y1-10*Y3-60) Div 3;
  Y2:= Y1-2*Y4-30; Y3:=Y2+Y4;
  bit[1]:='0';bit[2]:='1';bit[3]:='1';
  bit[4]:='0';bit[5]:='1';bit[6]:='0';
  bit[7]:='0';bit[8]:='1';bit[9]:='1';bit[10]:='0';
  CMI1;
  GraficarEjemplo(30,15,X2,Y2,Y3),
Closegraph;
End;
(************************************************)
Procedure PstTeoria;
(*COLOCA EL TEXTO LA TEORIA DEL CODIGO PST, Y LLAMA A LOS PROCEDIMIENTOS
  ENCARGADOS DE GRAFICAR EL EJEMPLO DE LA CODIFICACION*)
Var
  h, i, k, X1, Y1, Y2, X2, Y3, Y4, P : Integer;
  Texto:Array[1..19] of string;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
```

```
  SetColor(14);SetBkColor(1);
  X1:=GetMaxX; Y1:=GetMaxY; Y3:=Y1 Div 35;Y4:=Y1 Div 12;
  Rectangle(0,0,X1,Y1);
  SetTextStyle(2,0,6); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,10,'C O D I G.O  L I N E A L   P S T');
  SetTextStyle(2,0,5); SetTextJustify(0,2);
  Texto[1]:='';
  Texto[2]:='Este código le hace  corresponder a cada  pareja  de dígitos
binarios  una';
  Texto[3]:='pareja de dígitos ternarios.';
  Texto[4]:='La forma de asignar las parejas es la siguiente:';
  Texto[5]:='';
  Texto[6]:='                        ENTRADA BINARIA       SALIDA TERNARIA';
  Texto[7]:='                                              MODO +   MODO -';
  Texto[8]:='                                 00             - +      - +';
  Texto[9]:='                                 01             0 +      0 -';
  Texto[10]:='                                10             + 0      - 0';
  Texto[11]:='                                11             + -      + -';
  Texto[12]:='Para la entrada binaria  01  le  corresponde un  primer modo
"0 -"  cuando';
  Texto[13]:='vuelva aparecer le corresponderá 0+, es decir se alternan  los
modos  para';
  Texto[14]:='la eliminación de la componente continua. Lo mismo ocurre
para la entrada';
  Texto[15]:='binaria 10, las entradas binarias 00 y 11,no cambian el valor
de sus modos';
  Texto[16]:='';
  Texto[17]:='E J E M P L O :';
  Texto[18]:='';
  Texto[19]:='Codificación de la secuencia de bits: 0 1 0 0 1 1 1 0 1 0';
  X2:= (x1 - 580) Div 2;
  MoveTo(X2,30);X2:=GetX;Y2:=GetY;
  For i:=1 to 19 do
    Begin
    If (i > 5) and (i <12) then SetTextstyle(2,0,4)
    else SetTextstyle(2,0,5);
    OutTextXY(X2,Y2,Texto[i]);
    Y2:=Y2+Y3;
    End;
  N:=10; P:=1;
  X2:=(X1-300) Div 2; Y4:=(Y1-19*Y3-60) Div 3;
  Y2:= Y1-2*Y4-30; Y3:=Y2+Y4;
  bit[1]:='0';bit[2]:='1';bit[3]:='0';
  bit[4]:='0';bit[5]:='1';bit[6]:='1';
  bit[7]:='1';bit[8]:='0';bit[9]:='1';bit[10]:='0';
  PST1(P);
  GraficarEjemplo(30,30,X2,Y2,Y3);
Closegraph;
End;
(************************************************)
FIN DE LA UNIDAD CODIGOS3.TPU
(************************************************)
End.
```

UNIDAD MODULAC1.TPU

```
(*UNIDAD MODULAC1 CONTIENE LOS PROCEDIMIENTOS O SUBRUTINAS
  PARA LAS OPCIONES MODULACION Y TEORIA DE MODULACIONES*)
Unit Modulac1;

INTERFACE

Uses CRT, GRAPH;

(*VARIABLES GLOBALES QUE UTILIZA LA UNIDAD*)

Var
  c2, N, ControlGr, ModoGr : Integer;
  dato, reloj, coef : Array[1..60] Of Integer;
  bit : Array[1..30] Of Char;
  TX : Array[1..16] of String[10];

(*LISTADO DE PROCEDIMIENTOS QUE UTILIZA LA UNIDAD MODULAC1*)

Procedure DibujarCuadro(x1,y1,x2,y2,pp,f : Integer);
Procedure PintarFondo(x1,y1,x2,y2,f: integer);
Procedure CuadroSeleccionMod(Letra: Char);
Procedure CuadroModulacion;
Procedure Continuar;
Procedure TituloMod(Letra: Char);
Procedure GraficarBits(Y2, h : integer);
Procedure GraficarPortadora(Y2, m :Integer);
Procedure Opcion1(X1, Y1 : Integer; Var es: Char);
Procedure Opciones(X1, Y1 : Integer; Var es: Char);
Procedure Numeros3(x1, y1, x2, y2, B, C : Integer; var A : Integer);
Procedure Aleatorio2;
Procedure LeerDatos2;
Procedure IngresoDatos2;
Procedure MantenerDatos2;
Procedure ASK1(var L: Char);
Procedure FSK1(var L: Char);
Procedure PSK1(var L: Char);
Procedure M4PSK1(var L: Char);
Procedure M8PSK1(var L: Char);
Procedure M16PSK1(var L: Char);
Procedure M4QAM1(var L: Char);
Procedure M16QAM1(var L: Char);
Procedure ASK(Letra1: Char; var Letra2: Char);
Procedure FSK(Letra1: Char; var Letra2: Char);
Procedure PSK(Letra1: Char; var Letra2: Char);
Procedure M4PSK(Letra1: Char; var Letra2: Char);
Procedure M8PSK(Letra1: Char; var Letra2: Char);
Procedure M16PSK(Letra1: Char; var Letra2: Char);
Procedure M4QAM(Letra1: Char; var Letra2: Char);
Procedure M16QAM(Letra1: Char; var Letra2: Char);
Procedure Diagrama(x,y,r,nu : integer);
Procedure Diagrama1(x,y,r,nu : integer);
Procedure EjemploMod(Letra1: Char; var Letra2: Char);
Procedure ASKTeoria(Letra1: Char; var Letra2: Char);
Procedure FSKTeoria(Letra1: Char; var Letra2: Char);
Procedure PSKTeoria(Letra1: Char; var Letra2: Char);
Procedure M4PSKTeoria(Letra1: Char; var Letra2: Char);
Procedure M8PSKTeoria(Letra1: Char; var Letra2: Char);
Procedure M16PSKTeoria(Letra1: Char; var Letra2: Char);
Procedure M4QAMTeoria(Letra1: Char; var Letra2: Char);
Procedure M16QAMTeoria(Letra1: Char; var Letra2: Char);

IMPLEMENTATION

(***********************************************)
(*DESARROLLO DE CADA UNO DE LOS PROCEDIMIENTOS QUE UTILIZA LA UNIDAD
  MODULAC1*)
(***********************************************)
```

```
Procedure DibujarCuadro(x1,y1,x2,y2,pp,f : Integer);
(*DIBUJA LOS BORDES PARA LAS PANTALLAS DE PRESENTACION EN MODO TEXTO*)
Var
  i : integer;
Begin
TextColor(pp);TextBackground(f);
For i := (x1 + 1)  To (x2 - 1) Do
  Begin
  GotoXY(i,y1);Write(#205);
  GotoXY(i,y2);Write(#205);
  End;
For i := (y1 + 1) To (y2 - 1) Do
  Begin
  GotoXY(x1,i);Write(#186);
  GotoXY(x2,i);Write(#186);
  End;

GotoXY(x1,y1);Write(#201);
GotoXY(x2,y1);Write(#187);
GotoXY(x1,y2);Write(#200);
GotoXY(x2,y2);Write(#188);
End;
(************************************************)
Procedure PintarFondo(x1,y1,x2,y2,f: integer);
(*PONE EL COLOR DE FINDO EN LAS PANTALLAS DE PRESENTACION*)

Var
  i,j : integer;
Begin
ClrScr;
For i := x1 To x2 Do
  For j := y1 To y2 Do
    Begin
      GotoXY(i,j);
      Textbackground(f);
      Write(#0);
    End;
End;
(*****************************************************)
Procedure Continuar;
(*PERMITE CONTINUAR CON EL PROGRAMA SOLAMENTE CUANDO SE PULSA LA TECLA
  ENTER*)

Var
  s : Char;
Begin
s := ' ';
Repeat
  If KeyPressed Then
    s := ReadKey;
Until s = Char(13);
End;
(************************************************)
Procedure NUMEROS3(x1, y1, x2, y2, B, C : Integer; var A :Integer);
(*LEE EL NUMERO DE DATOS DE ENTRADA N Y PONE UN MENSAJE DE ERROR EN CASO DE
  NO SER EL DATO APROPIADO*)

Var
  i, j , codigo : integer;
  NU : Array [1..10] of char;
  numero1 : String[10];
Label
  Repetir;
Begin
REPETIR:
TextColor(14);
GotoXY(x1,y1);
Write('                    ');
GotoXY(x1,y1);
i := 1;
```

```
Repeat
If KeyPressed Then
  Begin
  NU[i] := ReadKey;
  Write(NU[i]);
  inc(i,1);
  End;
Until ( NU[i-1] = Char(13)) Or (i = 10);
Numero1:=NU[1];
For j:=2 To i-2 Do
Numero1:=Numero1 + NU[j];
Val(Numero1,A,codigo);
If (codigo <> 0) Or ((A > B) or (A < C)) Then
  Begin
  TextColor(139);
  GotoXY(x2,y2);Write('Dato incorrecto Presione ENTER para continuar');
  Continuar;
  GotoXY(x2,y2);Write('                                                    ');
  Goto Repetir;
  End;
End;
(************************************************)
Procedure CuadroModulacion;
(*PANTALLA DE SELECCION QUE CONTIENE LA LISTA DE LAS CLASES DE MODULACION
  DISPONIBLES EN EL PROGRAMA*)

Begin
ClrScr;
PintarFondo(1,2,80,25,7);
PintarFondo(8,8,73,19,3);
DibujarCuadro(1,2,80,25,1,7);
DibujarCuadro(8,7,73,19,1,3);
TextColor(14);TextBackground(1);
GotoXY(23,4);
Write(' M O D U L A C I O N   D I G I T A L ');
TextColor(4);TextBackground(3);
GotoXY(20,8);
Write('S I S T E M A S   D E   M O D U L A C I O N');
TextColor(1);
GotoXY(19,11);Write('A) MODULACION ASK');
GotoXY(19,12);Write('B) MODULACION FSK');
GotoXY(19,13);Write('C) MODULACION PSK');
GotoXY(19,14);Write('D) MODULACION 4-PSK');
GotoXY(19,15);Write('E) MODULACION 8-PSK');
GotoXY(45,11);Write('F) MODULACION 16-PSK');
GotoXY(45,12);Write('G) MODULACION 4-QAM');
GotoXY(45,13);Write('H) MODULACION 16-QAM');
GotoXY(43,14);Write('Esc) MENU PRINCIPAL');
GotoXY(43,15);Write('End) SALIR AL DOS');
TextColor(1);TextBackGround(7);
GotoXY(45,21);Write('Ingrese su selección...[   ] ');
GotoXY(70,21);
End;
(*********************************************************)
Procedure CuadroSeleccionMod(Letra: Char);
(*PANTALLA DE SELECION QUE MUESTRA LAS OPCIONES (MODULACION, TEORIA,
  ESPECTROS) Y PERMITE SELECCIONARLAS*)

Begin
ClrScr;
PintarFondo(1,1,80,25,1);
PintarFondo(19,10,63,17,3);
DibujarCuadro(1,2,80,25,15,1);
DibujarCuadro(18,9,64,17,1,3);
TextColor(4);TextBackground(7);
GotoXY(23,4);WriteLn(' M O D U L A C I O N   D I G I T A L ');
TextColor(1);
Case Letra Of
  'A','a' : Begin
        GotoXY(27,6);
```

```
        Write(' M O D U L A C I O N   A S K ');
        End;
  'B' ,'b': Begin
        GotoXY(27,6);
        Write(' M O D U L A C I O N   F S K ');
        End;
  'C','c' : Begin
        GotoXY(24,6);
        Write(' M O D U L A C I O N    2 - P S K ');
        End;
  'D' ,'d': Begin
        GotoXY(24,6);
        Write(' M O D U L A C I O N    4 - P S K ');
        End;
  'E' ,'e': Begin
        GotoXY(24,6);
        Write(' M O D U L A C I O N    8 - P S K ');
        End;
  'F' ,'f': Begin
        GotoXY(24,6);
        Write(' M O D U L A C I O N   16 - P S K ');
        End;
  'G' ,'g': Begin
        GotoXY(24,6);
        Write(' M O D U L A C I O N    4 - Q A M ');
        End;
  'H' ,'h': Begin
        GotoXY(24,6);
        Write(' M O D U L A C I O N   16 - Q A M ');
        End;
   End;
TextColor(15);TextBackGround(3);
GotoXY(30,11);Write('A)  MODULACION DE SEÑAL');
GotoXY(30,13);Write('B)  TEORIA');
GotoXY(30,15);Write('C)  ESPECTRO DE POTENCIA');
TextColor(2);TextBackGround(1);
GotoXY(17,19);Write('[Esc] : Menú Modulaciones    [End] : Salir al DOS');
Textcolor(15);
GotoXY(35,22);Write('Ingrese su selección...[  ]');
GotoXY(60,22);
End;
(***********************************************)
Procedure TituloMod(Letra: Char);
(*COLOCA EL TITULO EN EL CUADRO DE INGRESO DE DATOS DE ACUERDO CON LA CLASE
  DE MODULACION*)

Var
  i : Integer;
Begin
ClrScr;
DibujarCuadro(1,2,80,25,14,1);
GotoXY(1,22);Write(#204);GotoXY(80,22);Write(#185);
For i:=2 To 79 Do
   Begin
     GotoXY(i,22);Write(#205);
   End;
TextColor(0);TextBackground(15);
Case Letra Of
  'A','a' : Begin
        GotoXY(27,4);
        Write(' M O D U L A C I O N   A S K ');
        End;
  'B' ,'b': Begin
        GotoXY(27,4);
        Write(' M O D U L A C I O N   F S K ');
        End;
  'C','c' : Begin
        GotoXY(27,4);
        Write(' M O D U L A C I O N   P S K ');
        End;
```

```
  'D' ,'d': Begin
        GotoXY(24,4);
        Write(' M O D U L A C I O N     4 - P S K ');
        End;
  'E' ,'e': Begin
        GotoXY(24,4);
        Write(' M O D U L A C I O N     8 - P S K ');
        End;
  'F' ,'f': Begin
        GotoXY(24,4);
        Write(' M O D U L A C I O N    16 - P S K ');
        End;
  'G' ,'g': Begin
        GotoXY(24,4);
        Write(' M O D U L A C I O N     4 - Q A M ');
        End;
  'H' ,'h': Begin
        GotoXY(24,4);
        Write(' M O D U L A C I O N    16 - Q A M ');
        End;
   End;
End;
(*********************************************)
Procedure GraficarBits(Y2, h : integer);
(*REALIZA EL GRAFICO DE LOS BITS DE DATOS QUE SE UTILIZAN EN LA MODULACION*)

Var
  i, X, Y : Integer;
  numero : Array[1..10] Of String;
Begin
MoveTo(50,y2);
For i:=1 to N do
  Begin
    lineto(h*(i-1)+50,y2-20*dato[i]);
    lineto(h*i+50,y2-20*dato[i]);
  End;
SetColor(14);
MoveTo(50,y2);
For i:= 1 To N+1 Do
  Begin
    LineRel(0,3); X := GetX;
    X := X + h; Y := y2-2;
    MoveTo(X,Y);
  End;
For i:= 1 To N Do
  numero[i] := bit[i];
SetTextStyle(2,0,4);
MoveTo(50+(h Div 2),Y2+12);
For i:= 1 To N Do
  Begin
    X:=GetX; Y:=GetY;
    OutTextXY(X,Y,numero[i]);
    X := X + h; MoveTo(X,Y);
  End;
SetColor(11);SetTextJustify(0,2);
OutTextXY(52,y2-45,'SEÑAL BINARIA (TREN DE DATOS)');
End;
(*********************************************)
Procedure GraficarPortadora(Y2, m :Integer);
(*REALIZA EL GRAFICO DE LA SEÑAL PORTADORA UTILIZADA EN LA MODULACION*)

Var
  i, X, Y : Integer;
  G : Real;
Begin
i:=50;G:=0.0;
While i < (N+1)*50 Do
  Begin
    X := i;
    Y := 2*Y2-ROUND(20*sin(G));
```

```
    PutPixel(X,Y,13);
    G := G + (pi/(m*12.5));
    i := i + 1;
  End;
SetColor(13);SetTextJustify(0,2);
OutTextXY(52,2*y2-45,'SEÑAL PORTADORA');
End;
(***********************************************)
Procedure Opcion1(X1, Y1 : Integer; Var es: Char);
(*UNA VEZ FINALIZADA LA APLICACION ESTE PROCEDIMIENTO INDICA SI SE DEBE
  CONTINUAR O FINALIZAR CON EL PROGRAMA*)

Begin
SetColor(14);
MoveTo(0,y1-30);LineTo(x1,y1-30);
SetTextStyle(2,0,5);SetTextJustify(1,2);
SetColor(15);
OutTextXY(X1 Div 2,Y1-20,' ESC : Menú Anterior  F1 : Menú Modulaciones  END
: Salir al DOS ');
es:=' ';
Repeat
  If KeyPressed Then
    es := ReadKey;
Until es in ['1','2','3']
End;
(***********************************************)
Procedure Opciones(X1, Y1 : Integer; Var es: Char);
(*UNA VEZ FINALIZADO LA PRESENTACION DEL EJEMPLO EN LA TEORIA DE LA
  MODULACION ESTE PROCEDIMIENTO INDICA SI SE DEBE CONTINUAR O FINALIZAR CON
  EL PROGRAMA*)

Begin
SetColor(14);SetTextStyle(2,0,5);
MoveTo(0,y1-30);LineTo(x1,y1-30);
SetColor(15); SetTextJustify(2,2);
OutTextXY(x1-30,Y1-50,'Presione ENTER para EJEMPLO');
SetTextJustify(1,2);
OutTextXY(X1 Div 2,Y1-20,' ESC : Menú Anterior  F1 : Menú Modulaciones  END
: Salir al DOS ');
es:=' ';
Repeat
  If KeyPressed Then
    es := ReadKey;
Until es in [Char(27),Char(59),Char(79),Char(13)]
End;
(***********************************************)
Procedure Aleatorio2;
(*GENERA LOS N BITS DE DATOS ALEATORIOS, UTILIZANDO EL COMANDO RANDOM*)

Var
  i, x1, N1 : Integer;
Begin
GotoXY(3,23);
TextColor(14);
Write('   Ingresar el número de bits a generar [3 ≤ n ≤ 11] y luego pulsar
ENTER   ');
GotoXY(6,9);Write('Número de bits que desea generar : ');
GotoXY(41,9);
NUMEROS3(41,9,20,11,11,3,N1);
N := N1;
Randomize;
For i := 1 To N Do
  Begin
  x1 := Random(100);
  If (x1 <= 50) Then
    Begin
    bit[i] := '0';dato[i] := 0;
    End
  Else
    Begin
```

```
    bit[i] := '1';dato[i] := 1
    End
  End;
WriteLn;
TextColor(14);
GotoXY(6,11);Write('Los bits a modular son : ');
DibujarCuadro(6,13,73,17,15,1);
GotoXY(3,23);TextColor(14);
Write('                    Generando secuencia de bits a modular
      ');
GotoXY(8,15);
For i := 1 To N Do
  Begin
  Write(bit[i]:3);Delay(500);
  End;
End;
(*******************************.***************)
Procedure LEERDATOS2;
(*PERMITE EL INGRESO LOS N BITS DE DATOS EN FORMA MANUAL DESDE EL TECLADO*)

Var
  i, N1 : Integer;
  s :Char;
Begin
GotoXY(3,23);
Write('   Ingresar el número de bits a modular [3 ≤ N ≤ 11] y luego pulsar
ENTER');
TextColor(14);TextBackground(1);
GotoXY(6,9);Write('Número de bits : ');
GotoXY(23,9);
NUMEROS3(23,9,20,11,11,3,N1);
N := N1;
GotoXY(6,11);Write('Ingrese los datos a modular : ');
DibujarCuadro(6,13,73,17,14,1);
GotoXY(3,23);
Write('       Ingrese los bits de datos (ceros y unos) a modular,
secuencialmente');
GotoXY(10,15);
i := 1; s := ' ';
Repeat
  If KeyPressed Then
    Begin
      s := ReadKey;
      Case s Of
      '0' : Begin
            Write('  ',s);bit[i] := '0';Inc(i,1);
            End;
      '1' : Begin
            Write('  ',s);bit[i] := '1';Inc(i,1);
            End;
      Else Write(^g);
      End;
    End;
  Until i= N+1;
For i:=1 To N Do
  If bit[i] = '0' Then dato[i] := 0
  Else dato[i] := 1;
End;
(***********************************************)
Procedure IngresoDatos2;
(*PANTALLA DE PRESENTACION PARA EL INGRESO DE DATOS, SELECCIONA EL TIPO DE
  INGRESO DE DATOS ELEGIDO*)

Var
  i : Integer;
  s : Char;

Begin
TextColor(14);TextBackground(1);
GotoXY(6,7);Write('Desea Ingresar (I) o Generar (G) los bits : ');
```

```
GotoXY(3,23);
Write(' Teclear I para ingresar datos manualmente o G para generar
automaticamente ');
GotoXY(50,7);
s := ' ';
Repeat
  If KeyPressed Then
     Begin
     s := ReadKey;
     Case s Of
     'G','g' : Begin
               Write(s);Delay(1000);
               End;
     'I','i' : Begin
               Write(s);Delay(1000);
               End;
     Else Write(^g);
     End;
     End;
Until s in ['I','G','i','g'];

Case s Of
'G','g' : Aleatorio2;
'I','i' : LeerDatos2;
End;

i := 1;
While i <= 2*N Do
  Begin
  reloj[i] := 1; reloj[i+1] := 0;
  inc(i,2);
  End;
End;
(***********************************************)
Procedure MantenerDatos2;
(*GUARDA LOS DATOS DEL PROCESO DE MODULACION ANTERIOR Y DA LA OPCION DE
  MANTENERLOS O CAMBIARLOS*)

Var
  i, j : Integer;
  s : Char;
Begin
TextColor(14);TextBackground(1);
GotoXY(6,9);Write('Los bits a modular son : ');
DibujarCuadro(6,12,73,16,15,1);
TextColor(14);GotoXY(8,14);
For i := 1 To N Do
  Write(bit[i]:3);
GoToXY(6,19);Write('¿ Desea cambiar la secuencia de bits Si (S) o No (N)
?');
GotoXY(3,23);
Write('Teclear S si desea cambiar la secuencia de bits anterior o N para
mantenerla');
GotoXY(62,19);
s := ' ';
Repeat
  If KeyPressed Then
    Begin
      s := ReadKey;
      Case s Of
      'S','s' : Begin
                Write(s);Delay(1000);
                End;
      'N','n' : Begin
                Write(s);Delay(1000);
                End;
      Else Write(^g);
      End;
    End;
Until s in ['N','n','S','s'];
```

```
Case s Of
'S','s' : Begin
          For j := 5 To 20  Do
            For i := 5 To 73 Do
              Begin
               GOTOXY(i,j);Write(' ');
              End;
          IngresoDatos2;
          End;
End;
End;
(***********************************************)
Procedure ASK1(var L: Char);
(*REALIZA LA MODULACION ASK*)

Var
  h, i, j, m, X, Y, X1, Y1, Y2 : Integer;
  k, G : Real;
  numero : Array[1..10] Of String;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1:=GetMaxX; Y1:=GetMaxY;
  SetColor(14);SetBkColor(1);
  rectangle(0,0,X1,Y1);Rectangle(2,2,X1-2,Y1-2);
  SetTextStyle(1,0,1); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,15,'MODULACION DE AMPLITUD ASK');
  SetColor(11);
  Y2:=Y1 Div 4;h := 50;
  GraficarBits(Y2,h);
  GraficarPortadora(Y2,1);
  MoveTo(50,2*y2);Lineto(50*(N+1),2*y2);
  i := 50;G:=0.0;j:=1;m:=1;
  While i < (N+1)*50 Do
    Begin
      X := i;
      If (X = m*50)  Then
        Begin
          If bit[j] = '1' Then k := 1
          Else k := 0;
          inc(j);inc(m);
        End;
      Y := 3*y2-ROUND(20*k*sin(G));
      PutPixel(X,Y,15);
      G := G + (pi/12.5);
      inc(i);
    End;
  SetColor(14);
  MoveTo(50,3*y2+25);
  For i:= 1 To N+1 Do
    Begin
      LineRel(0,3); X := GetX;
      X := X + h; Y := 3*y2+25;
      MoveTo(X,Y);
    End;
  For i:= 1 To N Do
  numero[i] := bit[i];
  SetTextStyle(2,0,4);
  MoveTo(50+(h Div 2),3*Y2+25);
  For i:= 1 To N Do
    Begin
    X:=GetX; Y:=GetY;
    OutTextXY(X,Y,numero[i]);
    X := X + h; MoveTo(X,Y);
    End;
  SetTextJustify(0,2);
  OutTextXY(52,3*y2-45,'SEÑAL MODULADA');
  SetTextStyle(2,0,4);
  Opcion1(X1, Y1, L);
  CloseGraph;
```

```
End;
(*******************************************************)
Procedure FSK1;
(*REALIZA EL PROCESO DE MODULACION FSK*)

Var
  h, i, j, k, m, X, Y, X1, Y1, Y2 : Integer;
  G : Real;
  numero : Array[1..10] Of String;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1:=GetMaxX; Y1:=GetMaxY;
  SetColor(14);SetBkColor(1);
  Rectangle(0,0,X1,Y1);Rectangle(2,2,X1-2,Y1-2);
  SetTextStyle(1,0,1); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,10,'MODULACION DE FRECUENCIA FSK');
  SetColor(11);
  Y2:=Y1 Div 5;h := 50;
  GraficarBits(Y2,h);
  GraficarPortadora(Y2,2);
  Y2 :=Round(1.5* Y2);
  GraficarPortadora(Y2,1);
  Y2 := Y1 Div 5;
  If bit[1]='1' Then k:= 1
    Else k:= 2;
  i := 50;G:=0.0;j:=1;m:=1;
  While i < (N+1)*50 Do
      Begin
        X := i;
        If (X = m*50) Then
        Begin
          If bit[j] = '1' Then k:=1
            Else k:=2;
        inc(j,1);inc(m,1);
        End;
        Y := 4*Y2-ROUND(20*sin(G));
        PutPixel(X,Y,15);
        G := G + (pi/(k*12.5));
        inc(i,1);
      End;
  For i:= 1 To N Do
  numero[i] := bit[i];
  SetTextStyle(2,0,4);
  SetColor(15);
  MoveTo(50,4*y2+25);
  For i:= 1 To N+1 Do
    Begin
      LineRel(0,3); X := GetX;
      X := X + h; Y := 4*y2+25;
      MoveTo(X,Y);
    End;
  MoveTo(50+(h Div 2),4*Y2+25);
  For i:= 1 To N Do
    Begin
    X:=GetX; Y:=GetY;
    OutTextXY(X,Y,numero[i]);
    X := X + h; MoveTo(X,Y);
    End;
  SetTextJustify(0,2);
  OutTextXY(52,3*y2-45,'SEÑAL MODULADA');
  Opcion1(X1, Y1, 1);
  CloseGraph;
End;
(*******************************************************)
Procedure PSK1;
(*REALIZA EL PROCESO DE MODULACION 2-PSK)

Var
  h, i, j, m, X, Y, X1, Y1, Y2 : Integer;
```

```
  G, k : Real;
  numero : Array[1..10] Of String;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1:=GetMaxX; Y1:=GetMaxY;
  SetColor(14);SetBkColor(1);
  rectangle(0,0,X1,Y1);Rectangle(2,2,X1-2,Y1-2);
  SetTextStyle(1,0,1); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,15,'MODULACION DE FASE 2-PSK');
  SetColor(11);
  Y2:=Y1 Div 4;h := 50;
  GraficarBits(Y2,h);
  GraficarPortadora(Y2,1);
  MoveTo(50,2*Y2);Lineto(50*(N+1),2*Y2);
  If bit[1]='1' Then k:= 0
    Else k:= PI;
  i := 50;G:=0.0;j:=1;m:=1;
  While i < (N+1)*50 Do
      Begin
      X := i;
      If (X = m*50) Then
        Begin
          If bit[j] = '1' Then k:=(pi/2)
          Else k:=(-pi/2);
          inc(j,1);inc(m,1);
        End;
        Y := 3*Y2-ROUND(20*cos(G+k));
        PutPixel(X,Y,15);
        G := G + (pi/12.5);
        inc(i,1);
      End;
  SetColor(14);
  MoveTo(50,3*Y2);
  For i:= 1 To N+1 Do
    Begin
      LineRel(0,3); X := GetX;
      X := X + h; y:= 3*Y2-2;
      MoveTo(X,Y);
    End;
  MoveTo(50,3*Y2);Lineto(50*(N+1),3*Y2);
  For i:= 1 To N Do
  numero[i] := bit[i];
  SetTextStyle(2,0,4);
  MoveTo(50,3*y2+25);
  For i:= 1 To N+1 Do
    Begin
      LineRel(0,3); X := GetX;
      X := X + h; Y := 3*y2+25;
      MoveTo(X,Y);
    End;
  MoveTo(50+(h Div 2),3*Y2+25);
  For i:= 1 To N Do
    Begin
    X:=GetX; Y:=GetY;
    OutTextXY(X,Y,numero[i]);
    X := X + h; MoveTo(X,Y);
    End;
  SetTextJustify(0,2);
  OutTextXY(52,3*y2-45,'SEÑAL MODULADA');
  Opcion1(X1, Y1, L);
  CloseGraph;
End;
(*******************************************************)
Procedure M4PSK1;
(*REALIZA EL PROCESO DE MODULACION 4-PSK*)

Var
  h, i, j, m, X, Y, X1, Y1, Y2 : Integer;
  G, k : Real;
```

```
  numero : Array[1..10] Of String;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1:=GetMaxX; Y1:=GetMaxY;
  SetColor(14);SetBkColor(1);
  rectangle(0,0,X1,Y1);Rectangle(2,2,X1-2,Y1-2);
  SetTextStyle(1,0,1); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,15,'MODULACION DE FASE 4-PSK');
  SetColor(11);
  Y2:=Y1 Div 4;h := 50;
  GraficarBits(Y2,h);
  GraficarPortadora(Y2,2);
  MoveTo(50,2*Y2);Lineto(50*(N+1),2*Y2);
  i := 50;G:=0.0;j:=1;
  While i < (N+1)*50 Do
      Begin
        X := i;
        If (X = j*50) Then
          Begin
            If bit[j] = '1' Then
              Begin
                If bit[j+1] = '0' Then
                  Begin
                    k:=(-3*pi/4);inc(j,2);
                  End
                Else
                  Begin
                    k:=(-pi/4);inc(j,2);
                  End;
              End
            Else
              Begin
                If bit[j+1] = '0' Then
                  Begin
                    k:=(3*pi/4);inc(j,2);inc(m,1);
                  End
                Else
                  Begin
                    k:=(pi/4);inc(j,2);inc(m,1);
                  End;
              End;
          End;
        Y := 3*Y2-ROUND(20*cos(G+k));
        PutPixel(X,Y,15);
        G := G + (pi/25);
        i := i + 1;
      End;
  MoveTo(50,3*Y2);Lineto(50*(N+1),3*Y2);
  SetColor(14);
  MoveTo(50,3*Y2);i:=1;
  While i< N+1 Do
    Begin
      LineRel(0,5); X := GetX;
      X := X + 2*h; Y := 3*Y2-2;
      MoveTo(X,Y); inc(i,2);
    End;
  For i:= 1 To N Do
  numero[i] := bit[i];
  SetTextStyle(2,0,4);
  MoveTo(50,3*y2+25);
  For i:= 1 To ((N Div 2)+1) Do
    Begin
      LineRel(0,3); X := GetX;
      X := X + 2*h; Y := 3*y2+25;
      MoveTo(X,Y);
    End;
  MoveTo(50+(h Div 2),3*Y2+25);
  For i:= 1 To N Do
    Begin
```

```
    X:=GetX; Y:=GetY;
    OutTextXY(X,Y,numero[i]);
    X := X + h; MoveTo(X,Y);
    End;
  SetTextJustify(0,2);
  OutTextXY(52,3*y2-45,'SEÑAL MODULADA');
  Opcion1(X1, Y1, L);
  CloseGraph;
End;
(*******************************************************)
Procedure M8PSK1;
(*REALIZA EL PROCESO DE MODULACION 8-PSK*)

Var
  h, i, j, m, X, Y, X1, Y1, Y2 : Integer;
  G, k : Real;
  numero : Array[1..10] Of String;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1:=GetMaxX; Y1:=GetMaxY;
  SetColor(14);SetBkColor(1);
  rectangle(0,0,X1,Y1);Rectangle(2,2,X1-2,Y1-2);
  SetTextStyle(1,0,1); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,15,'MODULACION DE FASE 8-PSK');
  SetColor(11);
  Y2:=Y1 Div 4;h := 50;
  GraficarBits(Y2,h);
  GraficarPortadora(Y2,2);
  MoveTo(50,2*Y2);Lineto(50*(N+1),2*Y2);
  i :=50 ;G:=0.0;j:=1;
  While i < (N+1)*50 Do
      Begin
      X := i;
      If (X = j*50) Then
        Begin
        If bit[j] = '1' Then
          Begin
          If bit[j+1] = '1' Then
            Begin
            If bit[j+2] = '1' Then
              Begin
              k:=(-pi/8);inc(j,3);
              End
            Else
              Begin
              k:=(-3*pi/8);inc(j,3);
              End;
            End
          Else
            Begin
            If bit[j+2] = '0' Then
              Begin
              k:=(-5*pi/8);inc(j,3);
              End
            Else
              Begin
              k:=(-7*pi/8);inc(j,3);
              End;
            End;
          End
        Else
          Begin
          If bit[j+1] = '0' Then
            Begin
            If bit[j+2] = '0' Then
              Begin
              k:=(5*pi/8);inc(j,3);
              End
            Else
```

```
              Begin
              k:=(7*pi/8);inc(j,3);
              End;
            End
          Else
            Begin
            If bit[j+2] = '0' Then
              Begin
              k:=(3*pi/8);inc(j,3);
              End
            Else
              Begin
              k:=(pi/8);inc(j,3);
              End;
            End;
        End;

        Y := 3*Y2-ROUND(20*cos(G+k));
        PutPixel(X,Y,15);
        G := G + (pi/25);
        i := i + 1;
        End;
    MoveTo(50,3*Y2);Lineto(50*(N+1),3*Y2);
    SetColor(14);
    For i:= 1 To N Do
    numero[i] := bit[i];
    SetTextStyle(2,0,4);
    MoveTo(50,3*Y2+25);i:=1;
    While i <= ((N/3)+1) Do
      Begin
        LineRel(0,5); X := GetX;
        X := X + 3*h; Y := 3*Y2+25;
        inc(i,1); MoveTo(X,Y);
      End;
    MoveTo(50+(h Div 2),3*Y2+25);
    For i:= 1 To N Do
      Begin
      X:=GetX; Y:=GetY;
      OutTextXY(X,Y,numero[i]);
      X := X + h; MoveTo(X,Y);
      End;
    SetTextJustify(0,2);
    OutTextXY(52,3*y2-45,'SEÑAL MODULADA');
    Opcion1(X1, Y1, L);
    CloseGraph;
End;
(*******************************************************)
Procedure M16PSK1;
(*REALIZA EL PROCESO DE MODULACION 16-PSK*)

Var
  h, i, j, m, X, Y, X1, Y1, Y2 : Integer;
  G, k : Real;
  numero : Array[1..10] Of String;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1:=GetMaxX; Y1:=GetMaxY;
  SetColor(14);SetBkColor(1);
  rectangle(0,0,X1,Y1);Rectangle(2,2,X1-2,Y1-2);
  SetTextStyle(1,0,1); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,15,'MODULACION DE FASE 16-PSK');
  SetColor(11);
  Y2:=Y1 Div 4;h := 50;
  GraficarBits(Y2,h);
  GraficarPortadora(Y2,2);
  MoveTo(50,2*Y2);Lineto(50*(N+1),2*Y2);
  i := 50;G:=0.0;j:=1;
  While i < (N+1)*h Do
```

```
Begin
X := i;
If (X = j*h) Then
  Begin
  If bit[j] = '1' Then
    Begin
    If bit[j+1] = '1' Then
      Begin
      If bit[j+2] = '1' Then
        Begin
        If bit[j+3] = '1' Then
          Begin
          k:=(-5*pi/16);inc(j,4);
          End
        Else
          Begin
          k:=(-7*pi/16);inc(j,4);
          End;
        End
      Else
        Begin
        If bit[j+3] = '1' Then
          Begin
          k:=(-3*pi/16);inc(j,4);
          End
        Else
          Begin
          k:=(-pi/16);inc(j,4);
          End;
        End;
      End
    Else
      Begin
      If bit[j+2] = '1' Then
        Begin
        If bit[j+3] = '1' Then
          Begin
          k:=(-11*pi/16);inc(j,4);
          End
        Else
          Begin
          k:=(-9*pi/16);inc(j,4);
          End;
        End
      Else
        Begin
        If bit[j+3] = '1' Then
          Begin
          k:=(-13*pi/16);inc(j,4);
          End
        Else
          Begin
          k:=(-15*pi/16);inc(j,4);
          End;
        End;
      End;
    End
  Else
    Begin
    If bit[j+1] = '1' Then
      Begin
      If bit[j+2] = '1' Then
        Begin
        If bit[j+3] = '1' Then
          Begin
          k:=(11*pi/16);inc(j,4);
          End
        Else
          Begin
          k:=(9*pi/16);inc(j,4);
```

```
              End;
            End
          Else
            Begin
            If bit[j+3] = '1' Then
              Begin
              k:=(13*pi/16);inc(j,4);
              End
            Else
              Begin
              k:=(15*pi/16);inc(j,4);
              End;
            End;
        End
      Else
        Begin
        If bit[j+2] = '1' Then
          Begin
          If bit[j+3] = '1' Then
            Begin
            k:=(5*pi/16);inc(j,4);
            End
          Else
            Begin
            k:=(7*pi/16);inc(j,4);
            End;
          End
        Else
          Begin
          If bit[j+3] = '1' Then
            Begin
            k:=(3*pi/16);inc(j,4);
            End
          Else
            Begin
            k:=(pi/16);inc(j,4);
            End;
          End;
        End;
      End;
    End;

    Y := 3*Y2-ROUND(20*cos(G+k));
    PutPixel(X,Y,15);
    G := G + (pi/25);
    i := i + 1;
    End;
MoveTo(50,3*Y2);i:=1;
While i <= ((N/4)+1) Do
   Begin
     LineRel(0,5); X := GetX;
     X := X + 4*h; Y :=3*Y2-2 ;
     inc(i,1); MoveTo(X,Y);
   End;
MoveTo(50,3*Y2);Lineto(50*(N+1),3*Y2);
SetColor(14);
For i:= 1 To N Do
numero[i] := bit[i];
SetTextStyle(2,0,4);
MoveTo(50,3*Y2+25);i:=1;
While i <= ((N/4)+1) Do
  Begin
    LineRel(0,5); X := GetX;
    X := X + 4*h; Y :=3*Y2+25;
    inc(i,1);MoveTo(X,Y);
  End;
MoveTo(50+(h Div 2),3*Y2+25);
For i:= 1 To N Do
  Begin
  X:=GetX; Y:=GetY;
```

```
    OutTextXY(X,Y,numero[i]);
    X := X + h; MoveTo(X,Y);
    End;
  SetTextJustify(0,2);SetColor(14);
  OutTextXY(52,3*y2-45,'SEÑAL MODULADA');
  Opcion1(X1, Y1, L);
  CloseGraph;
End;
(*****************************************************)
Procedure M4QAM1;
(*REALIZA EL PROCESO DE MODULACION 4-QAM*)

Var
  h, i, j, m, X, Y, X1, Y1, Y2 : Integer;
  G, k : Real;
  numero : Array[1..10] Of String;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1:=GetMaxX; Y1:=GetMaxY;
  SetColor(14);SetBkColor(1);
  Rectangle(0,0,X1,Y1);Rectangle(2,2,X1-2,Y1-2);
  SetTextStyle(1,0,1); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,15,'MODULACION  4 - QAM');
  SetColor(11);
  Y2:=Y1 Div 4;h := 50;
  GraficarBits(Y2,h);
  GraficarPortadora(Y2,2);
  MoveTo(50,2*Y2);Lineto(50*(N+1),2*Y2);
  i := 50;G:=0.0;j:=1;
  While i < (N+1)*50 Do
    Begin
      X := i;
      If (X = j*50) Then
        Begin
        If bit[j] = '1' Then
          Begin
          If bit[j+1] = '0' Then
            Begin
            k:=(5*pi/4);inc(j,2);
            End
          Else
            Begin
            k:=(7*pi/4);inc(j,2);
            End;
          End
        Else
          Begin
          If bit[j+1] = '0' Then
            Begin
            k:=(3*pi/4);inc(j,2);inc(m,1);
            End
          Else
            Begin
            k:=(pi/4);inc(j,2);inc(m,1);
            End;
          End;
        End;
      Y := 3*Y2-ROUND(20*cos(G+k));
      PutPixel(X,Y,15);
      G := G + (pi/25);
      i := i + 1;
      End;
  MoveTo(50,3*Y2);i:=1;
  While i< N+1 Do
    Begin
      LineRel(0,5); X := GetX;
      X := X + 2*h; Y := 3*Y2-2;
      MoveTo(X,Y); inc(i,2);
    End;
```

```
  MoveTo(50,3*Y2);Lineto(50*(N+1),3*Y2);
  SetColor(14);
  For i:= 1 To N Do
  numero[i] := bit[i];
  SetTextStyle(2,0,4);
  MoveTo(50,3*Y2+25);i:=1;
  While i< N+1 Do
    Begin
      LineRel(0,5); X := GetX;
      X := X + 2*h; Y := 3*Y2+25;
      MoveTo(X,Y); inc(i,2);
    End;
  MoveTo(50+(h Div 2),3*Y2+25);
  For i:= 1 To N Do
    Begin
    X:=GetX; Y:=GetY;
    OutTextXY(X,Y,numero[i]);
    X := X + h; MoveTo(X,Y);
    End;
  SetTextJustify(0,2);
  OutTextXY(52,3*y2-45,'SEÑAL MODULADA');
  Opcion1(X1, Y1, L);
  Closegraph;
End;
(*******************************************************)
Procedure M16QAM1;
(*REALIZA EL PROCESO DE MODULACION 16-PSK*)

Var
  h, i, j, m, X, Y, X1, Y1, Y2 : Integer;
  G, k, k1 : Real;
  numero : Array[1..10] Of String;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1:=GetMaxX; Y1:=GetMaxY;
  SetColor(14);SetBkColor(1);
  rectangle(0,0,X1,Y1);Rectangle(2,2,X1-2,Y1-2);
  SetTextStyle(1,0,1); SetTextJustify(1,2);
  OutTextXY(X1 Div 2,15,'MODULACION 16 - Q A M');
  SetColor(11);
  Y2:=Y1 Div 4;h := 50;
  GraficarBits(Y2,h);
  GraficarPortadora(Y2,2);
  MoveTo(50,2*Y2);Lineto(50*(N+1),2*Y2);
  i := 50;G:=0.0;j:=1;
  While i < (N+1)*h Do
      Begin
      X := i;
      If (X = j*h) Then
        Begin
        If bit[j] = '1' Then
          Begin
          If bit[j+1] = '1' Then
            Begin
            If bit[j+2] = '1' Then
              Begin
              If bit[j+3] = '1' Then
                Begin
                k:=(pi/4);k1:=0.5;inc(j,4);
                End
              Else
                Begin
                k:=(3*pi/8);k1:=1;inc(j,4);
                End;
              End
            Else
              Begin
              If bit[j+3] = '1' Then
                Begin
```

```
          k:=(-pi/4);k1:=0.5;inc(j,4);
          End
        Else
          Begin
          k:=(-3*pi/8);k1:=1;inc(j,4);
          End;
        End;
      End
    Else
      Begin
      If bit[j+2] = '1' Then
        Begin
        If bit[j+3] = '1' Then
          Begin
          k:=(pi/8);k1:=1;inc(j,4);
          End
        Else
          Begin
          k:=(pi/4);k1:=1.2;inc(j,4);
          End;
        End
      Else
        Begin
        If bit[j+3] = '1' Then
          Begin
          k:=(-pi/8);k1:=1;inc(j,4);
          End
        Else
          Begin
          k:=(-pi/4);k1:=1.5;inc(j,4);
          End;
        End;
      End;
    End
  Else
    Begin
    If bit[j+1] = '1' Then
      Begin
      If bit[j+2] = '1' Then
        Begin
        If bit[j+3] = '1' Then
          Begin
          k:=(3*pi/4);k1:=0.5;inc(j,4);
          End
        Else
          Begin
          k:=(5*pi/8);k1:=1;inc(j,4);
          End;
        End
      Else
        Begin
        If bit[j+3] = '1' Then
          Begin
          k:=(-3*pi/4);k1:=0.5;inc(j,4);
          End
        Else
          Begin
          k:=(-5*pi/8);k1:=1;inc(j,4);
          End;
        End;
      End
    Else
      Begin
      If bit[j+2] = '1' Then
        Begin
        If bit[j+3] = '1' Then
          Begin
          k:=(7*pi/8);k1:=1;inc(j,4);
          End
        Else
```

```
                Begin
                k:=(3*pi/4);k1:=1.5;inc(j,4);
                End;
              End
            Else
              Begin
              If bit[j+3] = '1' Then
                Begin
                k:=(-7*pi/8);k1:=1;inc(j,4);
                End
              Else
                Begin
                k:=(-3*pi/4);k1:=1.5;inc(j,4);
                End;
              End;
            End;
          End;

          Y := 3*Y2-ROUND(20*k1*cos(G+k));
          PutPixel(X,Y,15);
          G := G + (pi/25);
          i := i + 1;
          End;
      MoveTo(50,3*Y2);i:=1;
      While i <= ((N/4)+1) Do
        Begin
          LineRel(0,5); X := GetX;
          X := X + 4*h; Y := 3*Y2-2;
          inc(i,1); MoveTo(X,Y);
        End;
      MoveTo(50,3*Y2);Lineto(50*(N+1),3*Y2);
      SetColor(14);
      For i:= 1 To N Do
      numero[i] := bit[i];
      SetTextStyle(2,0,4);
      MoveTo(50,3*Y2+25);i:=1;
      While i <= ((N/4)+1) Do
        Begin
          LineRel(0,5); X := GetX;
          X := X + 4*h; Y :=3*Y2+25;
          inc(i,1);MoveTo(X,Y);
        End;
      MoveTo(50+(h Div 2),3*Y2+25);
      For i:= 1 To N Do
        Begin
        X:=GetX; Y:=GetY;
        OutTextXY(X,Y,numero[i]);
        X := X + h; MoveTo(X,Y);
        End;
      SetTextJustify(0,2);
      OutTextXY(52,3*y2-45,'SEÑAL MODULADA');
      Opcion1(X1, Y1, L);
      Closegraph;
    End;
    (*****************************************************)
    Procedure ASK(Letra1: Char; var Letra2: Char);
    (*LLAMA A TODOS LOS PROCEDIMIENTOS PARA GRAFICAR LA MODULACION ASK
      COMPLETA*)

    Begin
    ClrScr;
    If c2 = 0 Then
      Begin
      TituloMod(Letra1);
      IngresoDatos2;
      End
    Else
      Begin
      TituloMod(Letra1);
```

```
  MantenerDatos2;
  End;

GotoXY(3,23);
Write('          Presione ENTER para graficar los bits y su modulación
respectiva     ');
Continuar;
ASK1(Letra2);
End;
(************************************************)
Procedure FSK(Letra1: Char; var Letra2: Char);
(*LLAMA A TODOS LOS PROCEDIMIENTOS PARA GRAFICAR LA MODULACION FSK
  COMPLETA*)

Begin
ClrScr;
If c2 = 0 Then
  Begin
  TituloMod(Letra1);
  IngresoDatos2;
  End
Else
  Begin
  TituloMod(Letra1);
  MantenerDatos2;
  End;
GotoXY(3,23);
Write('          Presione ENTER para graficar los bits y su modulación
respectiva     ');
Continuar;
FSK1(Letra2);
End;
(************************************************)
Procedure PSK(Letra1: Char; var Letra2: Char);
(*LLAMA A TODOS LOS PROCEDIMIENTOS PARA GRAFICAR LA MODULACION 2-PSK
  COMPLETA*)

Begin
ClrScr;
If c2 = 0 Then
  Begin
  TituloMod(Letra1);
  IngresoDatos2;
  End
Else
  Begin
  TituloMod(Letra1);
  MantenerDatos2;
  End;
GotoXY(3,23);
Write('          Presione ENTER para graficar los bits y su modulación
respectiva     ');
Continuar;
PSK1(Letra2);
End;
(*****************************************************)
Procedure M4PSK(Letra1: Char; var Letra2: Char);
(*LLAMA A TODOS LOS PROCEDIMIENTOS PARA GRAFICAR LA MODULACION 4-PSK
  COMPLETA*)

Begin
ClrScr;
If c2 = 0 Then
  Begin
  TituloMod(Letra1);
  IngresoDatos2;
  End
Else
  Begin
  TituloMod(Letra1);
```

```
  MantenerDatos2;
  End;

GotoXY(3,23);
Write('          Presione ENTER para graficar los bits y su modulación
respectiva     ');
Continuar;
M4PSK1(Letra2);
End;
(*********************************************************)
Procedure M8PSK(Letra1: Char; var Letra2: Char);
(*LLAMA A TODOS LOS PROCEDIMIENTOS PARA GRAFICAR LA MODULACION 8-PSK
  COMPLETA*)

Begin
ClrScr;
If c2 = 0 Then
  Begin
  TituloMod(Letra1);
  IngresoDatos2;
  End
Else
  Begin
  TituloMod(Letra1);
  MantenerDatos2;
  End;
GotoXY(3,23);
Write('          Presione ENTER para graficar los bits y su modulación
respectiva     ');
Continuar;
M8PSK1(Letra2);
End;
(*********************************************************)
Procedure M16PSK(Letra1: Char; var Letra2: Char);
(*LLAMA A TODOS LOS PROCEDIMIENTOS PARA GRAFICAR LA MODULACION 16-PSK
  COMPLETA*)

Begin
ClrScr;
If c2 = 0 Then
  Begin
  TituloMod(Letra1);
  IngresoDatos2;
  End
Else
  Begin
  TituloMod(Letra1);
  MantenerDatos2;
  End;

GotoXY(3,23);
Write('          Presione ENTER para graficar los bits y su modulación
respectiva     ');
Continuar;
M16PSK1(Letra2);
End;
(*********************************************************)
Procedure M4QAM(Letra1: Char; var Letra2: Char);
(*LLAMA A TODOS LOS PROCEDIMIENTOS PARA GRAFICAR LA MODULACION 4-QAM
  COMPLETA*)

Begin
ClrScr;
If c2 = 0 Then
  Begin
  TituloMod(Letra1);
  IngresoDatos2;
  End
Else
  Begin
```

```
  TituloMod(Letra1);
  MantenerDatos2;
  End;

GotoXY(3,23);
Write('          Presione ENTER para graficar los bits y su modulación
respectiva     ');
Continuar;
M4QAM1(Letra2);
End;
(*******************************************************)
Procedure M16QAM(Letra1: Char; var Letra2: Char);
(*LLAMA A TODOS LOS PROCEDIMIENTOS PARA GRAFICAR LA MODULACION 16-QAM
COMPLETA*)
Begin
ClrScr;
If c2 = 0 Then
  Begin
  TituloMod(Letra1);
  IngresoDatos2;
  End
Else
  Begin
  TituloMod(Letra1);
  MantenerDatos2;
  End;

GotoXY(3,23);
Write('          Presione ENTER para graficar los bits y su modulación
respectiva     ');
Continuar;
M16QAM1(Letra2);
End;
(*************************************************)
Procedure Diagrama(x,y,r,nu : integer);
(*DIBUJA EL DIAGRAMA DE CONSTELACION LOS LAS MODULACIONES DE TEORIA,
  COLOCANDO LEYENDA EN LOS EJES*)
var
  i, x1, y1 : Integer;
  G, G1 : Real;
Begin
SetTextStyle(2,0,4);
G := (2*pi/nu); G1:=(pi/nu);
SetColor(11);
For i:=1 to nu Do
  Begin
  x1 := round(r*cos(G1));
  y1 := round(r*sin(G1));
  SetTextJustify(1,1);
  OutTextXY(x + x1 ,y - y1 ,'*');
  SetTextJustify(1,1);
  OutTextXY(x + x1, y - y1 - 15, TX[i]);
  G1 := G1 + G;
  End;
MoveTo(x-r-10,y);LineTo(x+r+10,y);OutTextXY(x+r+10,y,'>');
MoveTo(x,y-r-10);LineTo(x,y+r+10);OutTextXY(x,y+r+10,'v');
SetTextJustify(0,1);
OutTextXY(x+r+20,y,'Cos(Wct)');
SetTextJustify(1,2);
OutTextXY(x,y+r+15,'Sen(Wct)');
SetColor(9);
Circle(x,y,r);
SetColor(14);
End;
(*************************************************)
Procedure Diagrama1(x,y,r,nu : integer);
(*REALIZA EL DIAGRAMA DE CONSTELACION SIN COLOCAR LEYENDAS EN LOS EJES*)
var
  i, x1, y1 : Integer;
  G, G1 : Real;
```

```
Begin
SetTextStyle(2,0,4);
G := (2*pi/nu); G1:=(pi/nu);
SetColor(11);
For i:=1 to nu Do
  Begin
  x1 := round(r*cos(G1));
  y1 := round(r*sin(G1));
  SetTextJustify(1,1);
  OutTextXY(x + x1 ,y - y1 ,'*');
  SetTextJustify(1,1);
  OutTextXY(x + x1 ,y - y1 - 10, TX[i]);
  G1 := G1 + G;
  End;
MoveTo(x-r-10,y);LineTo(x+r+10,y);
MoveTo(x,y-r-10);LineTo(x,y+r+10);
SetColor(9);
Circle(x,y,r);
SetColor(14);
End;
(***********************************************)
Procedure EjemploMod(Letra1: Char; var Letra2: Char);
(*LLAMA A LOS PROCEDIMIENTOS DE MODULACION PARA CADA UNA DE LAS CLASES CON
  LO QUE SE PUEDE VER EL EJEMPLO DE MODULACION, ES UTILIZADO PARA LA
  APLICACION TEORIA DE MODULACIONES*)

Var
 i : integer;
Begin
N:=10;
bit[1]:='0'; bit[2]:= '0'; bit[3]:='1';
bit[4]:='0'; bit[5]:= '0'; bit[6]:='1';
bit[7]:='0'; bit[8]:= '0'; bit[9]:='1';bit[10]:='0';

For i:=1 To N Do
  Begin
  If bit[i] = '1' Then
     dato[i] := 1
  Else
     dato[i] := 0;
  End;
Case Letra1 Of
     'A','a' : ASK1(Letra1);
     'B','b' : FSK1(Letra1);
     'C','c' : PSK1(Letra1);
     'D','d' : M4PSK1(Letra1);
     'E','e' : M8PSK1(Letra1);
     'F','f' : M16PSK1(Letra1);
     'G','g' : M4QAM1(Letra1);
     'H','h' : M16QAM1(Letra1);
     End;
End;
(***********************************************)
Procedure ASKTeoria(Letra1: Char; var Letra2: Char);
(*MUESTRA EN LA PANTALLA LA TEORIA, LAS ECUACIONES Y EJEMPLO PARA EL CODIGO
 ASK*)

Var
  h, i, k, X1, Y1, Y2, X2, Y3, Y4 : Integer;
  Texto:Array[1..20] of string;
Begin
ControlGr := Detect;
InitGraph(ControlGr, ModoGr, 'c:\tp\bgi ');
SetColor(14);SetBkColor(1);
X1:=GetMaxX; Y1:=GetMaxY; Y3:=Y1 Div 22;Y4:=Y1 Div 12;
rectangle(0,0,X1,Y1);
SetTextStyle(2,0,6); SetTextJustify(1,2);
OutTextXY(X1 Div 2,10,'MODULACION  DIGITAL ASK');
SetTextStyle(2,0,5); SetTextJustify(0,2);
Texto[1]:='La modulación ASK (Manipulación por desplazamiento  de amplitud)
```

```
TX[1]:='   0' ;TX[2]:='   1';
Diagrama(X1 Div 2,Y1-(8*Y3), 4*Y3, 2);
  Opciones(X1, Y1, Letra2);
  Closegraph;
  Case Letra2 Of
  Char(13) : EjemploMod(Letra1, Letra2);
  End;
End;
(*****************************************************)
Procedure M4PSKTeoria(Letra1: Char; var Letra2: Char);
(*MUESTRA EN LA PANTALLA LA TEORIA, LAS ECUACIONES Y EJEMPLO PARA EL CODIGO
  4-PSK*)

Var
  h, i, k, X1, Y1, Y2, X2, Y3: Integer;
  Texto:Array[1..23] of string;
 Begin
 ControlGr :=Detect;
 InitGraph(ControlGr, ModoGr,'c:\tp\bgi');
 SetColor(14); SetBkColor(1);
 X1:=GetMaxX; Y1:=GetMaxY; Y3:= y1 div 28;
 rectangle(0,0,X1,Y1);
 SetTextStyle(2,0,6);SetTextJustify(1,2);
 OutTextXY(X1 Div 2,10,'MODULACION  DIGITAL 4-PSK');
 SettextStyle(2,0,5);SetTextJustify(0,2);
Texto[1]:='Corresponde al  tipo de modulación PSK (Manipulación  por
desplazamiento de';
Texto[2]:='fase) es  utilizada a efectos  de reducir el  ancho  de  banda
para lo cual';
Texto[3]:='cual se  utilizan estos esquemas de  señalización de más de 2
niveles, esta';
Texto[4]:='modulación consiste en  la combinación  de pulsos binarios
sucesivos  para';
Texto[5]:='formar un pulso de mayor amplitud.';
Texto[6]:='La expresión matemática para este tipo de modulación es la
siguiente:';
Texto[7]:='                                              bn(t).δΦ';
Texto[8]:='                                 g(t)=Cos(Wc + ----------).t';
Texto[9]:='                                                  2';
Texto[10]:='donde δΦ = 2 π/M es la separación entre fases de señales
adyacentes,';
Texto[11]:='(M=4 para 4-PSK), bn es una señal simétrica NRZ con valores +1,
+3, -1, -3.';
Texto[12]:=' ';
Texto[13]:='DIAGRAMA DE CONSTELACION';
Texto[14]:='MODULACION 4-PSK:';
 X2 := (X1-580) Div 2;
 MoveTo(X2,40);X2:=GetX;Y2:=GetY;
 For i:=1 to 14 do
  Begin
  OutTextXY(X2,Y2,Texto[i]);
  Y2:=Y2+Y3;
  End;
TX[1]:=' 0 1';TX[2]:=' 0 0';TX[3]:=' 1 0';TX[4]:=' 1 1';
Diagrama(X1 Div 2,Y1-(8*Y3), 4*Y3, 4);
  Opciones(X1, Y1, Letra2);
  Closegraph;
  Case Letra2 Of
  Char(13) : EjemploMod(Letra1, Letra2);
  End;
End;
(*****************************************************)
Procedure M8PSKTeoria(Letra1: Char; var Letra2: Char);
(*MUESTRA EN LA PANTALLA LA TEORIA, LAS ECUACIONES Y EJEMPLO PARA EL CODIGO
  8-PSK*)
Var
  h, i, k, X1, Y1, Y2, X2, Y3: Integer;
  Texto:Array[1..23] of string;
Begin
ControlGr :=Detect;
```

```
InitGraph(ControlGr, ModoGr,'c:\tp\bgi');
SetColor(14);SetBkColor(1);
X1:=GetMaxX; Y1:=GetMaxY; Y3:= y1 div 28;
rectangle(0,0,X1,Y1);
SetTextStyle(2,0,6);SetTextJustify(1,2);
OutTextXY(X1 Div 2,10,'MODULACION  DIGITAL 8-PSK');
SetTextStyle(2,0,5);SetTextJustify(0,2);
Texto[1]:='Corresponde  al  tipo  de  modulación  PSK  (Manipulación  por
desplazamiento de';
Texto[2]:='fase) es  utilizada a efectos  de reducir el  ancho de banda para
lo cual';
Texto[3]:='este esquema de señalización utiliza 8 niveles (M=8).';
Texto[4]:='La expresión matemática para este tipo de modulación es la
siguiente:';
Texto[5]:=' ';
Texto[6]:='                                                bn(t).δΦ';
Texto[7]:='                                  g(t)=Cos(Wc + ----------).t';
Texto[8]:='                                                   2';
Texto[9]:=' ';
Texto[10]:='Las ocho fases diferentes estarán separadas un ángulo δΦ = π/4,
bn es una';
Texto[11]:='señal simétrica de 8 niveles NRZ, con valores: +1, +3, +5, +7,
-1, -3, -5,';
Texto[12]:=' ';
Texto[13]:=' ';
Texto[14]:='DIAGRAMA DE CONSTELACION';
Texto[15]:='MODULACION 8-PSK:';
X2:=(X1-580) Div 2;
MoveTo(X2,40);X2:=GetX;Y2:=GetY;
For i:=1 to 15 do
  Begin
  OutTextXY(X2,Y2,Texto[i]);
  Y2:=Y2+Y3;
  End;
  TX[1]:='011';TX[2]:='010';TX[3]:='000';TX[4]:='001';
  TX[5]:='101';TX[6]:='100';TX[7]:='110';TX[8]:='111';
  Diagrama(X1 Div 2, Y1-(8*Y3), 4*Y3, 8);
  Opciones(X1, Y1, Letra2);
  Closegraph;
  Case Letra2 Of
  Char(13) : EjemploMod(Letra1, Letra2);
  End;
  End;
(*****************************************************)
Procedure M16PSKTeoria(Letra1: Char; var Letra2: Char);
(*MUESTRA EN LA PANTALLA LA TEORIA, LAS ECUACIONES Y EJEMPLO PARA EL CODIGO
  16-PSK*)
Var
  h, i, k, X1, Y1, Y2, X2, Y3: Integer;
  Texto:Array[1..23] of string;
Begin
ControlGr :=Detect;
InitGraph(ControlGr, ModoGr,'c:\tp\bgi');
SetColor(14);SetBkColor(1);
X1:=GetMaxX; Y1:=GetMaxY; Y3:= y1 div 28;
rectangle(0,0,X1,Y1);
SetTextStyle(2,0,6);SetTextJustify(1,2);
OutTextXY(X1 Div 2,10,'MODULACION  DIGITAL 16-PSK');
SetTextStyle(2,0,5);SetTextJustify(0,2);
Texto[1]:='Corresponde  al  tipo  de  modulación  PSK  (Manipulación  por
desplazamiento';
Texto[2]:='de fase) es  utilizada a efectos  de reducir el  ancho de  banda
para lo';
Texto[3]:='cual se  utiliza esquemas de  señalización de 16 niveles.';
Texto[4]:='La expresión matemática para este tipo de modulación es la
siguiente:';
Texto[5]:='                                                bn(t).δΦ';
Texto[6]:='                                  g(t)=Cos(Wc + ----------).t';
Texto[7]:='                                                   2';
Texto[8]:='donde  δΦ  =  2  π/M  es  la  separación  entre  fases  de  señales
```

```
adyacentes,';
Texto[9]:='(M=16 para 16-PSK),bn es una señal  simétrica NRZ con (niveles)
valores :';
Texto[10]:='+1, +3, +5, +7, +9, +11, +13, +15, -1, -3, -5, -7, -9, -11, -13,
-15.';
Texto[11]:=' ';
Texto[12]:='DIAGRAMA DE CONSTELACION';
Texto[13]:='MODULACION 16-PSK:';
X2 := (X1-580) Div 2;
MoveTo(X2,40);X2:=GetX;Y2:=GetY;
For i:=1 to 13 do
  Begin
  OutTextXY(X2,Y2,Texto[i]);
  Y2:=Y2+Y3;
  End;
  TX[1]:=' 0000';TX[2]:=' 0001';TX[3]:=' 0011';TX[4]:=' 0010';
  TX[5]:=' 0110';TX[6]:=' 0111';TX[7]:=' 0101';TX[8]:=' 0100';
  TX[9]:=' 1000';TX[10]:='1001 ';TX[11]:=' 1011';TX[12]:=' 1010';
  TX[13]:=' 1110';TX[14]:=' 1111';TX[15]:=' 1101';TX[16]:=' 1100';
  Diagrama(X1 Div 2, Y1-9*Y3, 5*Y3, 16);
  Opciones(X1, Y1, Letra2);
  Closegraph;
  Case Letra2 Of
  Char(13) : EjemploMod(Letra1, Letra2);
  End;
  End;
(*****************************************************)
Procedure M4QAMTeoria(Letra1: Char; var Letra2: Char);
(*MUESTRA EN LA PANTALLA LA TEORIA, LAS ECUACIONES Y EJEMPLO PARA EL CODIGO
  4-QAM*)

Var
  h, i, k, X1, Y1, Y2, X2, Y3: Integer;
  Texto:Array[1..23] of string;
Begin
ControlGr :=Detect;
InitGraph(ControlGr, ModoGr,'c:\tp\bgi');
SetColor(14);SetBkColor(1);
X1:=GetMaxX; Y1:=GetMaxY; Y3:= y1 div 28;
rectangle(0,0,X1,Y1);
SetTextStyle(2,0,6);SetTextJustify(1,2);
OutTextXY(X1 Div 2,10,'MODULACION  DIGITAL QAM');
SetTextStyle(2,0,5);SetTextJustify(0,2);
Texto[1]:='Corresponde al tipo de modulación QAM (Quadrature Amplitude
Modulation),';
Texto[2]:='a diferencia de PSK la amplitud de la  onda  senoidal  ya  no
permanece';
Texto[3]:='constante variando al  igual que  la fase, este esquema consiste
en la ';
Texto[4]:='modulación  multinivel  de  amplitud  de  dos  portadoras
independientes';
Texto[5]:='en cuadratura.';
Texto[6]:='La expresión matemática para este tipo de modulación es la
siguiente:';
Texto[7]:='';
Texto[8]:='                              g(t)=ai.Cos(Wc.t) + bi.Sen(Wc.t)';
Texto[9]:='                                                  ';
Texto[10]:='donde ai y bi toman en forma independiente los valores discretos
previstos';
Texto[11]:='según el número de niveles establecidos siendo M=L². Para QAM
n=4, L=2.';
Texto[12]:=' ';
Texto[13]:='DIAGRAMA DE CONSTELACION';
Texto[14]:='MODULACION 4-QAM.';
X2 := (X1-580) Div 2;
MoveTo(X2,40);X2:=GetX;Y2:=GetY;
For i:=1 to 13 do
  Begin
  OutTextXY(X2,Y2,Texto[i]);
  Y2:=Y2+Y3;
```

```
  End;
TX[1]:=' 0 1';TX[2]:=' 0 0';TX[3]:=' 1 0';TX[4]:=' 1 1';
Diagrama(X1 Div 2, Y1-9*Y3, 4*Y3, 4);
  Opciones(X1, Y1, Letra2);
  Closegraph;
  Case Letra2 Of
  Char(13) : EjemploMod(Letra1, Letra2);
  End;
End;
(*****************************************************)
Procedure M16QAMTeoria(Letra1: Char; var Letra2: Char);
(*MUESTRA EN LA PANTALLA LA TEORIA, LAS ECUACIONES Y EJEMPLO PARA EL CODIGO
  16-QAM*)

Var
  h, i, k, X1, Y1, Y2, X2, Y3: Integer;
  Texto:Array[1..23] of string;
Begin
ControlGr :=Detect;
InitGraph(ControlGr, ModoGr,'c:\tp\bgi');
SetColor(14);SetBkColor(1);
X1:=GetMaxX; Y1:=GetMaxY; Y3:= y1 div 30;
rectangle(0,0,X1,Y1);
SetTextStyle(2,0,6);SetTextJustify(1,2);
OutTextXY(X1 Div 2,10,'MODULACION  DIGITAL 16-QAM');
SetTextStyle(2,0,5);SetTextJustify(0,2);
Texto[1]:='En este tipo de modulación la amplitud de la señal no permanece
constante,';
Texto[2]:='variando al  igual que  la fase, este esquema consiste en la
modulación';
Texto[3]:='multinivel de amplitud de dos portadoras independientes, la
ecuación para';
Texto[4]:='este tipo de modulación es la siguiente:';
Texto[5]:=' ';
Texto[6]:='                          g(t) = ai*Cos(Wc.t) + bi*Sen(Wc.t)';
Texto[7]:='                                                 ';
Texto[8]:='donde ai y bi toman en forma independiente los valores discretos
previstos';
Texto[9]:='según el número de niveles establecidos siendo M=L².';
Texto[10]:='En el caso de 16-QAM, M=16 y L=4, las variables ai y bi  pueden
tener los';
Texto[11]:='valores (-3, -1, +3, +1). Cada valor de ai o bi son asociados
con un dibit.';
Texto[12]:=' ';
Texto[13]:='DIAGRAMA DE CONSTELACION';
Texto[14]:='MODULACION 16-QAM:';
X2 := (X1-580) Div 2;
MoveTo(X2,40);X2:=GetX;Y2:=GetY;
For i:=1 to 14 do
  Begin
  OutTextXY(X2,Y2,Texto[i]);
  Y2:=Y2+Y3;
  End;
  TX[1]:='1111';TX[2]:='0111';TX[3]:='0101';TX[4]:='1101';
  Diagrama1(X1 Div 2,Y1-10*Y3-10,2*Y3,4);
  TX[1]:='1011';TX[2]:='1110';TX[3]:='0110';TX[4]:='0011';
  TX[5]:='0001';TX[6]:='0100';TX[7]:='1100';TX[8]:='1001';
  Diagrama1(X1 Div 2,Y1-10*Y3-10,4*Y3,8);
  TX[1]:='1010';TX[2]:='0010';TX[3]:='0000';TX[4]:='1000';
  Diagrama(X1 Div 2,Y1-10*Y3-10,6*Y3,4);
  Opciones(X1, Y1, Letra2);
  Closegraph;
  Case Letra2 Of
  Char(13) : EjemploMod(Letra1, Letra2);
  End;
End;End.
(*************************************************)
         FIN DE LA UNIDAD MODULAC1
(*************************************************)
```

LISTADO DEL PROGRAMA

```
(*UNIDAD MODULAC2 CONTIENE LOS PROCEDIMIENTOS O SUBRUTINAS
  PARA LA OPCION ESPECTROS DE POTENCIA DE LAS MODULACIONES*)
Unit Modulac2;

INTERFACE
Uses CRT, GRAPH;

(*VARIABLES GLOBALES QUE UTILIZA LA UNIDAD*)

Var
  ControlGr, ModoGr : Integer;
  es : Char;

(*LISTA DE PROCEDIMIENTOS QUE UTILIZA LA UNIDAD*)

Procedure Opcion1(X1, Y1 : Integer);
Procedure P2PSK;
Procedure P4PSK;
Procedure P8PSK;
Procedure P16PSK;
Procedure CuadroEspectrosmod;
Procedure CompararEspectrosmod;
Procedure ASKPotencia;
Procedure FSKPotencia;
Procedure M2pSKPotencia;
Procedure M4PSKPotencia;
Procedure M8PSKPotencia;
Procedure M16PSKPotencia;
Procedure M4QAMPotencia;
IMPLEMENTATION
(***********************************************)
Procedure Opcion1(X1, Y1 : Integer);
(*PERMITE ELEGIR, SI SE CONTINUA EN EL PROGRAMA O SE FINALIZA*)

Begin
SetColor(14);
MoveTo(0,y1-30);LineTo(x1,y1-30);
SetTextStyle(2,0,5);SetTextJustify(1,2);
SetColor(15);
OutTextXY(X1 Div 2,Y1-20,' ESC : Menú Anterior  F1 : Menú Modulaciones  END
: Salir al DOS ');
es:=' ';
Repeat
  If KeyPressed Then
    es := ReadKey;
Until es in [Char(27),Char(59),Char(79)];
End;
(***********************************************)
Procedure P2PSK;
(*EVALUA, Y REALIZA EL GRAFICO LA ECUACION DEL LA DEP DE LA MODULACION
  2-PSK)

Var
 i, X, Y, X1, Y1, X2, Y2 : Integer;
 G : Real;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 4;

  i := 0;G:=0.00001;
  While i < 6*x2 Do
    Begin
      X := i + x2;
      Y := (3*y2)-ROUND((1.5*y2-3)*((sin(G)/G)*(sin(G)/G)));
      PutPixel(X,Y,7);G := G + (pi/(4*x2));
      inc(i,1);
    End;
SetColor(7);SetTextjustify(1,2);
```

```
OutTextXY(3*X2,2*y2,'PSK');SetColor(15);
End;
(***********************************************)
Procedure P4PSK;
(*EVALUA, Y REALIZA EL GRAFICO LA ECUACION DEL LA DEP DE LA MODULACION
  4-PSK)
Var
 i, X, Y, X1, Y1, X2, Y2 : Integer;
 G : Real;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 4;

  i := 0;G:=0.00001;
  While i < 6*x2 Do
    Begin
      X := i + x2;
      Y := (3*y2)-ROUND((1.5*y2+3)*((sin(G)/G)*(sin(G)/G)));
      PutPixel(X,Y,14);G := G + (pi/(2*x2));
      inc(i,1);
    End;
SetColor(14);SetTextjustify(1,2);
OutTextXY(x2+80,3*y2-45,'4-PSK');SetColor(15);
End;
(***********************************************)
Procedure P8PSK;
(*EVALUA, Y REALIZA EL GRAFICO LA ECUACION DEL LA DEP DE LA MODULACION
  8-PSK)
Var
 i, X, Y, X1, Y1, X2, Y2 : Integer;
 G : Real;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 4;

  i := 0;G:=0.00001;
  While i < 6*x2 Do
    Begin
      X := i + x2;
      Y := (3*y2)-ROUND((1.5*y2+10)*((sin(G)/G)*(sin(G)/G)));
      PutPixel(X,Y,15);G := G + (pi/((1.333)*x2));
      inc(i,1);
    End;
SetColor(15);SetTextjustify(1,2);
OutTextXY(x2+20,2*y2-15,'8-PSK');SetColor(15);
End;
(***********************************************)
Procedure P16PSK;
(*EVALUA, Y REALIZA EL GRAFICO LA ECUACION DEL LA DEP DE LA MODULACION
  16-PSK)
Var
 i, X, Y, X1, Y1, X2, Y2 : Integer;
 G : Real;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 4;

  i := 0;G:=0.00001;
  While i < 6*x2 Do
    Begin
      X := i + x2;
      Y := (3*y2)-ROUND((1.5*y2+15)*((sin(G)/G)*(sin(G)/G)));
      PutPixel(X,Y,13);G := G + (pi/(x2));
      inc(i,1);
    End;
  SetColor(13);SetTextjustify(1,2);
  OutTextXY(x2+28,3*y2-35,'16-PSK');
End;
(***********************************************)
Procedure P4QAM;
```

```
(*EVALUA, Y REALIZA EL GRAFICO LA ECUACION DEL LA DEP DE LA MODULACION
  4-QAM)
Var
 i, X, Y, X1, Y1, X2, Y2 : Integer;
 G : Real;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 4;

  i := 0;G:=0.00001;
  While i < 6*x2 Do
    Begin
      X := i + x2;
      Y := (3*y2)-ROUND((2*y2-20)*((sin(G)/G)*(sin(G)/G)));
      PutPixel(X,Y,11);G := G + (pi/(2*x2));
      inc(i,1);
    End;

  SetColor(11);
  SetTextJustify(1,2);SetTextStyle(2,0,5);
  OutTextXY(x2+30,2*y2-15,'4-QAM');
End;
(************************************************)
Procedure P16QAM;
(*EVALUA, Y REALIZA EL GRAFICO LA ECUACION DEL LA DEP DE LA MODULACION
  16-QAM)
Var
 i, X, Y, X1, Y1, X2, Y2, X3 : Integer;
 G : Real;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 8; Y2:=Y1 Div 4; X3 := X2 Div 2;

  i := 0;G:=0.00001;
  While i < 6*x2 Do
    Begin
      X := i + x2;
      Y := (3*y2)-ROUND((1.5*y2+13)*((sin(G)/G)*(sin(G)/G)));
      PutPixel(X,Y,13);G := G + (pi/(x2));
      inc(i,1);
    End;
  SetColor(13);
  SetTextJustify(1,2);SetTextStyle(2,0,5);
  OutTextXY(x2+x3,2*y2-15,'16-QAM');
End;
(*********************************************************)
Procedure CuadroEspectrosMod;
(*REALIZA LOS BORDES Y DIBUJA LOS EJES PARA LA COMPARACIO DE LAS DEP DE LAS
  MODULACIONES M-PSK*)

Var
  i, X, Y, X1, Y1, X2, Y2, X3, Y3 : Integer;
Begin
  SetTextStyle(1,0,1);
  SetBkColor(1);SetColor(15);
  X1:=GetMaxX; Y1:=GetMaxY;
  Rectangle(0,0,X1,Y1);Rectangle(2,2,X1-2,Y1-2);
  SetColor(14);
  X2:=X1 Div 8; Y2:=Y1 Div 4;X3:=X2 Div 2;Y3:=Y2 Div 2;
  MoveTo(x2,y2);LineTo(x2,3*y2);LineTo(7*x2,3*y2);
  MoveTo(x2,3*y2-3);
  For i:= 1 To 13 Do
    Begin
    LineRel(0,4); X := GetX;
    X := X + x2 Div 2; Y := 3*y2-3;
    MoveTo(X,Y);
    End;

  MoveTo(x2-2,3*y2);
  For i:= 1 To 9 Do
```

```
    Begin
    LineRel(4,0); Y := GetY;
    Y := Y - (y2 Div 4); X := x2-2;
    MoveTo(X,Y);
    End;
SetColor(14);
SetTextStyle(2,0,4);
OutTextXY(x2-5,3*y2+10,'W-Wc');OutTextXY(10*x3,3*y2+10,'2π/To');
SetTextJustify(0,2);
OutTextXY(6*x3,3*y2+10,'π/To');
OuttextXY(14*X3,3*Y2+10,'3π/To');
OutTextXY(15*X3-30,3*y2-20,'w');
OutTextXY(X2-14,Y2-20,'Sy(w)/To');
SetTextJustify(2,2);
OutTextXY(X2-5,3*Y2-10,'-30 dB');
OutTextXY(X2-5,5*Y3-10,'-20 dB');
OutTextXY(X2-5,4*Y3-10,'-10 dB');
OutTextXY(X2-5,3*y3,'0 dB');
OutTextXY(X2-5,2*Y3,'10 dB');Delay(1000);
End;
(*******************************************************)
Procedure ComparарEspectrosMod;
(*LLAMA A LOS PROCEDIMIENTOS RELACIONADOS CON LA COMPARACION DE LOS
  ESPECTROS DE LAS MODULACIONES M-PSK*)

Var
  X1, Y1, X2, Y2 : Integer;
Begin
  X1:=GetMaxX; Y1:=GetMaxY;
  X2:=X1 Div 6; Y2:=Y1 Div 16;
  Delay(1000);
  SetTextJustify(0,2);SetColor(14);
  OutTextXY(x2 Div 2,13*y2+10,'- DESEA COMPARAR CON EL ESPECTRO DE OTRA
M-PSK ?');
  OutTextXY(x2 Div 2,14*y2,'1. PSK    2. 4-PSK    3. 8-PSK    4. 16-PSK');
  SetColor(15);
  MoveTo(0,y1-30);LineTo(x1,y1-30);
  SetTextStyle(2,0,5);SetTextJustify(1,2);
  OutTextXY(X1 Div 2,Y1-20,' ESC : Menú Anterior  F1 : Menú Modulaciones
END : Salir al DOS ');
  es := ' ';
  Repeat
   If KeyPressed Then
     Begin
       es := ReadKey;
       Case es Of
        '1' : P2PSK;
        '2' : P4PSK;
        '3' : P8PSK;
        '4' : P16PSK;
       End;
     End
  Until es in [Char(27),Char(59),Char(79)];
End;
(*******************************************************)
Procedure ASKPotencia;
(*REALIZA LA EVALUACION Y EL GRAFICO DE LA DEP DE LA MODULACION ASK*)
Var
  i, X, Y, X1, Y1, X2, Y2, X3, Y3 : Integer;
  G : Real;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1:=GetMaxX; Y1:=GetMaxY;
  SetColor(14);SetBkColor(1);
  Rectangle(0,0,X1,Y1);Rectangle(1,1,X1-2,Y1-2);
  X2:=X1 Div 6; Y2:=Y1 Div 4;
  X3:=x2 Div 2; Y3:=Y2 Div 2;
  SetTextStyle(1,0,2);SetTextJustify(1,2);
  SetColor(13);
```

```
  OutTextXY(X1 Div 2,30,'Espectro de Potencia ASK');
  SetColor(7);
  MoveTo(x2,y2);LineTo(x2,3*y2);LineTo(5*x2,3*y2);
  MoveTo(x2,3*y2-3);
  For i:= 1 To 8 Do
    Begin
      LineRel(0,4); X := GetX;
      X := X + X3; Y := 3*y2-3;
      MoveTo(X,Y);
    End;
  MoveTo(x2-2,3*y2);
  For i:= 1 To 9 Do
    Begin
      LineRel(4,0); Y := GetY;
      Y := Y - (y2 Div 4); X := x2-2;
      MoveTo(X,Y);
    End;
  SetColor(14);
  SetTextStyle(2,0,4);SetTextJustify(1,2);
  OutTextXY(x2,3*y2+10,'0');OutTextXY(3*x2,3*y2+10,'Wc');
  OutTextXY(5*x3,3*y2+10,'Wc-(2π/To)');
  OutTextXY(7*x3,3*y2+10,'Wc+(2π/To)');
  OutTextXY(X2,y2-15,'Sy(w)/A²To');OutTextXY(5*x2+10,3*y2,'w');
  SetTextJustify(2,2);
  OutTextXY(X2-5,2*Y2-25,'0.10');
  OutTextXY(X2-5,2*y2-80,'0.14');
  i := 0;G:=(-3*PI);
  While i < Round(120*pi) Do
    Begin
      X := i +3*x3;
      Y := (3*y2)-ROUND(((2*y2)-Y3)*((sin(G)/G)*(sin(G)/G)));
      PutPixel(X,Y,11);G := G + (pi/53);
      inc(i,1);
    End;
  Opcion1(X1, Y1);
  Closegraph;
End;
(*******************************************************)
Procedure FSKPotencia;
(*REALIZA LA EVALUACION Y EL GRAFICO DE LA DEP DE LA MODULACION FSK*)

Var
  i, X, Y, X1, Y1, X2, Y2, X3, Y3 : Integer;
  G : Real;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  X1:=GetMaxX; Y1:=GetMaxY;
  SetColor(14);SetBkColor(1);
  Rectangle(0,0,X1,Y1);Rectangle(1,1,X1-2,Y1-2);
  X2:=X1 Div 6; Y2:=Y1 Div 4;
  X3:=x2 Div 2; Y3:=Y2 Div 2;
  SetTextStyle(1,0,2);SetTextJustify(1,2);
  SetColor(13);
  OutTextXY(X1 Div 2,30,'Espectro de Potencia FSK');
  SetColor(7);
  MoveTo(x2,y2);LineTo(x2,3*y2);LineTo(5*x2+x3,3*y2);
  MoveTo(x2-2,3*y2);
  For i:= 1 To 9 Do
    Begin
      LineRel(4,0); Y := GetY;
      Y := Y - (y2 Div 4); X := x2-2;
      MoveTo(X,Y);
    End;
  i := 0;G:=(-3*PI);
  While i < Round(100*pi) Do
    Begin
      X := i + x2;
      Y := (3*y2)-ROUND((y2)*((sin(G)/G)*(sin(G)/G)));
      PutPixel(X,Y,13);G := G + (pi/50);
```

```
      inc(i,1);
    End;
  i := 0;G:=(-3*PI);
  While i < Round(50*pi) Do
    Begin
      X := i + 4*x2;
      Y := (3*y2)-ROUND((y2)*((sin(G)/G)*(sin(G)/G)));
      PutPixel(X,Y,11);G := G + (pi/30);
      inc(i,1);
    End;
  SetColor(14);
  SetTextStyle(2,0,4);SetTextJustify(1,2);
  OutTextXY(x2,y2-15,'Sy(w)/A²To');OutTextXY(11*X3+10,3*y2-5,'w');
  OutTextXY(x2,3*y2+10,'0');OutTextXY(5*X3,3*y2+10,'Wc1');
  OutTextXY(10*X3,3*y2+10,'Wc2');
  OutTextXY(11*x3,3*y2+10,'Wc2+(2π/To)');
  OutTextXY(9*X3,3*y2+10,'Wc2-(2π/To)');
  OutTextXY(3*x2,3*y2+10,'Wc1+(2π/To)');
  OutTextXY(2*x2,3*y2+10,'Wc1-(2π/To)');
  SetTextJustify(2,2);
  OutTextXY(x2-5,2*y2,'0.10');
  SetColor(14);
  MoveTo(x2-2,3*y2);
  For i:= 1 To 9 Do
    Begin
      LineRel(4,0); Y := GetY;
      Y := Y - (y2 Div 4); X := x2-2;
      MoveTo(X,Y);
    End;
  Opcion1(X1, Y1);
  Closegraph;
End;
(***********************************************)
Procedure M2PSKPotencia;
(*REALIZA LA PANTALLA DE PRESENTACION EN MODO GRAFICO DEL ESPECTRO DE
  POTENCIA CORRESPONDIENTE*)

Var
  x1, y1: Integer;
  s: Char;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  x1:=GetmaxX; y1:=GetmaxY;
  SetTextStyle(1,0,2);SetTextJustify(1,2);
  SetColor(13);
  OutTextXY(X1 Div 2,30,'Espectro de Potencia 2-PSK');
  CuadroEspectrosMod;
  P2PSK;
  SetColor(14);
  CompararEspectrosMod;
  Closegraph;
End;
(*****************************************************)
Procedure M4PSKPotencia;
(*REALIZA LA PANTALLA DE PRESENTACION EN MODO GRAFICO DEL ESPECTRO DE
  POTENCIA CORRESPONDIENTE*)

Var
  x1, y1: Integer;
  s: Char;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  x1:=GetmaxX;y1:=GetmaxY;
  SetTextStyle(1,0,2);SetTextjustify(1,2);
  SetColor(13);
  OutTextXY(X1 Div 2,30,'Espectro de Potencia 4-PSK');
  CuadroEspectrosMod;
  P4PSK;
```

```
  SetColor(14);
  CompararEspectrosMod;
  Closegraph;
End;
(*******************************************************)
Procedure M8PSKPotencia;
(*REALIZA LA PANTALLA DE PRESENTACION EN MODO GRAFICO DEL ESPECTRO DE
  POTENCIA CORRESPONDIENTE*)

Var
  x1, y1: Integer;
  s: Char;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  x1:=GetmaxX;y1:=GetmaxY;
  SetTextStyle(1,0,2);SetTextjustify(1,2);
  SetColor(13);
  OutTextXY(X1 Div 2,30,'Espectro de Potencia 8-PSK');
  CuadroEspectrosMod;
  P8PSK;
  SetColor(14);
  CompararEspectrosMod;
  Closegraph;
End;
(*******************************************************)
Procedure M16PSKPotencia;
(*REALIZA LA PANTALLA DE PRESENTACION EN MODO GRAFICO DEL ESPECTRO DE
  POTENCIA CORRESPONDIENTE*)
Var
  x1, y1: Integer;
  s: Char;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  x1:=GetmaxX;y1:=GetmaxY;
  SetTextStyle(1,0,2);SetTExtjustify(1,2);
  SetColor(13);
  OutTextXY(X1 Div 2,30,'Espectro de Potencia 16-PSK');
  CuadroEspectrosMod;
  P16PSK;
  SetColor(14);
  CompararEspectrosMod;
  Closegraph;
  End;
(*********************************************)
Procedure M4QAMPotencia;
(*REALIZA LA PANTALLA DE PRESENTACION EN MODO GRAFICO DEL ESPECTRO DE
  POTENCIA CORRESPONDIENTE*)

Var
  h, i, j, m, X, Y, X1, Y1, X2, Y2 : Integer;
  G, k : Real;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  x1:=GetmaxX; y1:=GetmaxY;
  SetTextStyle(1,0,2);SetTextJustify(1,2);
  SetColor(13);
  OutTextXY(X1 Div 2,30,'Espectro de Potencia 4-QAM');
  CuadroEspectrosMod;
  P4QAM;
  Opcion1(x1, y1);
  Closegraph;
End;
(*********************************************************)
Procedure M16QAMPotencia;
(*REALIZA LA PANTALLA DE PRESENTACION EN MODO GRAFICO DEL ESPECTRO DE
  POTENCIA CORRESPONDIENTE*)
```

```
Var
  h, i, j, m, X, Y, X1, Y1, X2, Y2 : Integer;
  G, k : Real;
Begin
  ControlGr := Detect;
  InitGraph(ControlGr, ModoGr, 'c:\tp\bgi');
  x1:=GetmaxX; y1:=GetmaxY;
  SetTextStyle(1,0,2);SetTextJustify(1,2);
  SetColor(13);
  OutTextXY(X1 Div 2,30,'Espectro de Potencia 16-QAM');
  CuadroEspectrosMod;
  P16QAM;
  Opcion1(x1, y1);
  Closegraph;
 End;
(*************************************************)
FIN DE LA UNIDAD MODULAC2.TPU
(*************************************************)
End.
```

ANEXO C

TABLAS MATEMATICAS

TABLAS MATEMATICAS

.1 IDENTIDADES TRIGONOMETRICAS

$$\operatorname{sen}(A \pm B) = \operatorname{sen} A \cos B \pm \cos A \operatorname{sen} B$$
$$\cos(A \pm B) = \cos A \cos B \mp \operatorname{sen} A \operatorname{sen} B$$

$$\operatorname{sen} A \operatorname{sen} B = \tfrac{1}{2}[\cos(A - B) - \cos(A + B)]$$
$$\cos A \cos B = \tfrac{1}{2}[\cos(A + B) + \cos(A - B)]$$
$$\operatorname{sen} A \cos B = \tfrac{1}{2}[\operatorname{sen}(A + B) + \operatorname{sen}(A - B)]$$

$$\operatorname{sen} 2A = 2 \operatorname{sen} A \cos A$$
$$\cos 2A = 2\cos^2 A - 1 = 1 - 2\operatorname{sen}^2 A = \cos^2 A - \operatorname{sen}^2 A$$

$$\operatorname{sen}^2 A = \tfrac{1}{2}(1 - \cos 2A)$$
$$\cos^2 A = \tfrac{1}{2}(1 + \cos 2A)$$

$$\operatorname{sen} A = \frac{1}{2j}(e^{jA} - e^{-jA})$$
$$\cos A = \tfrac{1}{2}(e^{jA} + e^{-jA})$$
$$e^{\pm jA} = \cos A \pm j \operatorname{sen} A$$

.2 IDENTIDADES DE FUNCIONES COMPLEJAS

$$z = x + jy = \sqrt{x^2 + y^2}\, e^{j\tan^{-1}(y/x)}$$
$$z^* = x - jy = \sqrt{x^2 + y^2}\, e^{-j\tan^{-1}(y/x)}$$
$$|z|^2 = zz^* = x^2 + y^2$$
$$\mathcal{R}e\{z\} = \tfrac{1}{2}[z + z^*]$$
$$\mathcal{I}m\{z\} = \frac{1}{2j}[z - z^*]$$

$$\mathcal{R}e\{z_1 z_2\} = \mathcal{R}e\{z_1\}\mathcal{R}e\{z_2\} - \mathcal{I}m\{z_1\}\mathcal{I}m\{z_2\}$$
$$\mathcal{I}m\{z_1 z_2\} = \mathcal{R}e\{z_1\}\mathcal{I}m\{z_2\} + \mathcal{I}m\{z_1\}\mathcal{R}e\{z_2\}$$

.3 SERIES

MacLaurin:

$$f(x) = f(0) + f'(0)x + \frac{1}{2!}f''(0)x^2 + \frac{1}{3!}f'''(0)x^3 + \cdots.$$

Exponencial:

$$e^x = 1 + x + \frac{1}{2!}x^2 + \frac{1}{3!}x^3 + \frac{1}{4!}x^4 + \cdots.$$

Trigonométrica:

$$\operatorname{sen} x = x - \frac{1}{3!}x^3 + \frac{1}{5!}x^5 - \frac{1}{7!}x^7 + \cdots.$$

$$\cos x = 1 - \frac{1}{2!}x^2 + \frac{1}{4!}x^4 - \frac{1}{6!}x^6 + \cdots.$$

Binomial: (para $x^2 < 1$):

$$(1 \pm x)^n = 1 \pm nx + \frac{1}{2!}n(n-1)x^2 \pm \frac{1}{3!}n(n-1)(n-2)x^3 + \cdots.$$

$$(1 \pm x)^{-n} = 1 \mp nx + \frac{1}{2!}n(n+1)x^2 \mp \frac{1}{3!}n(n+1)(n+2)x^3 + \cdots.$$

.4 INTEGRALES INDEFINIDAS

$$\int \operatorname{sen} ax\, dx = -\frac{1}{a}\cos ax$$

$$\int \cos ax\, dx = \frac{1}{a}\operatorname{sen} ax$$

$$\int \operatorname{sen}^2 ax\, dx = \frac{x}{2} - \frac{\operatorname{sen} 2ax}{4a}$$

$$\int \cos^2 ax\, dx = \frac{x}{2} + \frac{\operatorname{sen} 2ax}{4a}$$

$$\int \operatorname{sen} ax \cos ax\, dx = \frac{1}{2a}\operatorname{sen}^2(ax)$$

$$\int x \operatorname{sen} ax\, dx = \frac{1}{a^2}(\operatorname{sen} ax - ax \cos ax)$$

$$\int x \cos ax\, dx = \frac{1}{a^2}(\cos ax + ax \operatorname{sen} ax)$$

$$\int \operatorname{sen} ax \operatorname{sen} bx\, dx = \frac{\operatorname{sen}(a-b)x}{2(a-b)} - \frac{\operatorname{sen}(a+b)x}{2(a+b)} \qquad a^2 \neq b^2$$

$$\int \cos ax \cos bx\, dx = \frac{\operatorname{sen}(a-b)x}{2(a-b)} + \frac{\operatorname{sen}(a+b)x}{2(a+b)} \qquad a^2 \neq b^2$$

$$\int \operatorname{sen} ax \cos bx\, dx = -\frac{\cos(a-b)x}{2(a-b)} - \frac{\cos(a+b)x}{2(a+b)} \qquad a^2 \neq b^2$$

$$\int e^{ax}\, dx = \frac{1}{a}e^{ax}$$

$$\int xe^{ax}\, dx = \frac{1}{a^2}e^{ax}(ax-1)$$

$$\int x^2 e^{ax}\, dx = \frac{1}{a^3}e^{ax}(a^2x^2 - 2ax + 2)$$

$$\int e^{ax} \operatorname{sen} bx\, dx = \frac{1}{a^2+b^2}e^{ax}(a \operatorname{sen} bx - b \cos bx)$$

$$\int e^{ax} \cos bx\, dx = \frac{1}{a^2+b^2}e^{ax}(a \cos bx + b \operatorname{sen} bx)$$

$$\int \left[\frac{\operatorname{sen} ax}{x}\right]^2 dx = a\int \frac{\operatorname{sen} 2ax}{x}\, dx - \frac{\operatorname{sen}^2 ax}{x}$$

$$\int \frac{dx}{a^2 + b^2x^2} = \frac{1}{ab}\tan^{-1}\left(\frac{bx}{a}\right)$$

$$\int \frac{x^2\, dx}{a^2 + b^2x^2} = \frac{x}{b^2} - \frac{a}{b^3}\tan^{-1}\left(\frac{bx}{a}\right)$$

$$\int \frac{dx}{(a^2 + b^2x^2)^2} = \frac{x}{2a^2(a^2 + b^2x^2)} + \frac{1}{2a^3b}\tan^{-1}\left(\frac{bx}{a}\right)$$

$$\int \frac{x^2\, dx}{(a^2 + b^2x^2)^2} = \frac{-x}{2b^2(a^2 + b^2x^2)} + \frac{1}{2ab^3}\tan^{-1}\left(\frac{bx}{a}\right)$$

$$\int \frac{dx}{(a^2 + b^2x^2)^3} = \frac{x}{4a^2(a^2 + b^2x^2)^2} + \frac{3x}{8a^4(a^2 + b^2x^2)} + \frac{3}{8a^5b}\tan^{-1}\left(\frac{bx}{a}\right)$$

.5 INTEGRALES DEFINIDAS

$$\int_0^\infty \frac{\operatorname{sen} ax}{x}\, dx = \begin{cases} \pi/2 & a > 0 \\ 0 & a = 0 \\ -\pi/2 & a < 0 \end{cases}$$

$$\int_0^x \frac{\operatorname{sen} u}{u}\, du \triangleq \operatorname{Si}(x) \quad \text{(integral tabulada como función de } x\text{)}$$

$$\int_0^\infty \frac{\operatorname{sen}^2 ax}{x^2}\, dx = |a|\,\pi/2$$

BIBLIOGRAFIA

- AVILA N., Diseño y Construcción de un Codec Didáctico para Transmisión Digital en Banda Base, tesis EPN, Quito, 1994

- BELLAMY J., Digital telephony, John Wiley & Sons, New York, 1982

- BORLAND, Turbo Pascal 6.0, programer's guide, U.S.A., 1990

- FEHER K., Digital Communications, Prentice Hall, Englewood Clifs, 1981

- FREEMAN R., Sistemas de Telecomunicaciones, LIMUSA, México, 1991

- IEEE, Transactions on Communications, Digital Radio, Associate Guest Editors, K. Feher y otros, Vol. COM-27 (No. 12,1979)

- INICTEL, División de Transmisiones, Transmisión Digital por Microondas, Quito.

- KUSTRA R. y TUJSNAIDER o., Principios de comunicaciones digitales, vol. II, colección AHCIET-ICI, FICUM S.A., 1978

- JOYANES L. Turbo Pascal 6.0 a su alcance, McGRAW-HILL, España, 1993.

- LATHI B.P. Random Signals and Communication Theory, International Textbook Company, Pensilvania, 1968.

- LATHI B.P. Sistemas de Comunicación, McGRAW-HILL, México, 1991.

- PAPOULIS A. Sistemas Digitales y Analógicos, Transformadas de Fourier, Estimación Espectral, MARCOMBO S.A., España, 1978.

- STEPHEN O'BRIEN, Turbo Pascal 6, Manual de Referencia, McGRAW-HILL, México, 1992.

- SCHWARTZ MISCHA, Transmisión de Información, Modulación y Ruido, McGRAW-HILL, 1980.

- STREMLER F.G. Sistemas de Comunicación, Fondo Educativo Interamericano, México. 1989.

- VIDALLER J. y OTROS, Transmisión de Datos, E.T.S. Ingenieros Telecomunicaciones-Universidad Politécnica de Madrid, Madrid, 2da ed., 1979