ANÁLISIS DEL DESEMPEÑO DE UN SISTEMA DE COMUNICACIONES CON MODULACIÓN DBPSK/DQPSK BASADO EN HARDWARE RECONFIGURABLE

Fabián Alveiro Burbano Robles
Mario Andrés Ramos Goyes

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Telecomunicaciones
GRIAL – Grupo de Radio e InALámbricas
Señales y Sistemas de Acceso y Difusión Basados en Radio.
GNTT – Grupo I+D Nuevas Tecnologías en Telecomunicaciones
Señales y Sistemas de Telecomunicaciones
Popayán, 2014

ANÁLISIS DEL DESEMPEÑO DE UN SISTEMA DE COMUNICACIONES CON MODULACIÓN DBPSK/DQPSK BASADO EN HARDWARE RECONFIGURABLE

Proyecto de Trabajo de Grado

Fabián Alveiro Burbano Robles
Mario Andrés Ramos Goyes

Director: PhD. Pablo Emilio Jojoa Gómez.

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Telecomunicaciones
GRIAL – Grupo de Radio e InALámbricas
Señales y Sistemas de Acceso y Difusión Basados en Radio.
GNTT – Grupo I+D Nuevas Tecnologías en Telecomunicaciones
Señales y Sistemas de Telecomunicaciones
Popayán, 2014

NOTA DE ACEPTACIÓN

_____

_____

_____

_____

_____
Presidente del Jurado

_____
Jurado

_____
Jurado

Popayán (Cauca), Junio del 2014

AGRADECIMIENTOS

A Dios por darme la fuerza, la sabiduría y el amor que me han permitido ganar cada batalla y así alcanzar este triunfo. A mis padres, porque siempre han sido mi luz de esperanza al final del camino, a mi hermano por sus sacrificios y ese amor infinito que me ha dado desde siempre, a mi abuela que ha sido mi gran apoyo y a pesar de las adversidades me ha dado una sonrisa para salir adelante. A Yovana, por todo el tiempo que me ha permitido pasar a su lado y porque me ha ayudado a dar más amor a las personas. A mis profesores, amigos, compañeros y demás personas que han estado en este proceso, sólo resta decirles muchas gracias. Finalmente, a Mario mi compañero y gran amigo con quien he compartido muchas alegrías y tristezas, gracias a su dedicación hemos podido culminar con satisfacción esta gran etapa de nuestras vidas.

Fabián

A Dios por llenarme con su amor, sus bendiciones, su sabiduría y por estar siempre conmigo en todos los días de mi vida para alcanzar este objetivo. A mis padres, que han sido mis pilares y constructores de este triunfo con su apoyo, con su guía y con su ejemplo. A mi hermano, familiares, profesores y amigos que han aportado de diferentes maneras a alcanzar este título. Por ultimo a mi compañero Fabián que siempre ha sido no solo un apoyo en la parte académica, sino un gran amigo que siempre ha estado presente y que Dios quiera esta amistad que nació en un aula de clase perdure por siempre para compartir muchas vivencias más.

Mario

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ACRÓNIMOS

8PSK (*8 Phase Shitf Keying*): Modulación por desplazamiento de fase de 8 estados.
16PSK (*16 Phase Shitf Keying*): Modulación por desplazamiento de fase de 16 estados.
ASK (*Amplitude Shitf Keying*): Modulación por desplazamiento de amplitud.
AWGN (*Additive White Gaussian Noise*): Ruido aditivo blanco Gaussiano.
BER (*Bit Error Rate*): tasa de error de bit.
BPSK (*Binary Phase Shitf Keying*): Modulación por desplazamiento diferencial de fase binaria.
CLB (*Configurable logic block*): Bloque lógico configurable.
CPLD (*Complex Programmable Logic Device*): Dispositivo lógico programable complejo
DBPSK (*Differential Binary Phase Shitf Keying*): Modulación por desplazamiento diferencial de fase binaria.
DPSK (*Differential Phase Shitf Keying*): Modulación por desplazamiento diferencial de fase.
DQPSK (*Differential Quadrature Phase Shitf Keying*): Modulación por desplazamiento diferencial de fase en cuadratura.
FIET: Facultad de Ingeniería Electrónica y Telecomunicaciones.
FPGA (*Field Programmable Gate Array)*: Arreglo de compuertas programable
FSK (*Frequency Shitf Keying*): Modulación por desplazamiento de frecuencia.
GNTT: Grupo de Investigación y Desarrollo Nuevas Tecnologías en Telecomunicaciones
GRIAL: Grupo de Radio e InALámbricas
HDL (*Hardware Description Language*): Lenguaje de Descripción Hardware.
IOBs (*Input / Output Blocks*): Bloques de entrada / salida.
ISE (*Integrated Software Enviroment*): Entorno de Software integrado.
LUT (*Look-up Tables*): Tablas de consulta.
PSK (*Phase Shitf Keying*): Modulación por desplazamiento de fase.
QAM (*Quadrature Amplitude Modulation*): Modulación de Amplitud en cuadratura.
QPSK (*Binary Phase Shitf Keying*): Modulación por desplazamiento de fase en cuadratura.
RAM (*Random Access Memory*): Memoria de acceso aleatoria.
ROM (*Read Only Memory*): Memoria de solo lectura.
VHDL (*VHSIC Hardware Description Language*): Lenguaje de Descripción Hardware VHSIC.
VHSIC (*Very High Speed Integrated Circuit*): Circuito integrado de muy alta velocidad.

INTRODUCCIÓN

En la actualidad, las modulaciones digitales son ampliamente utilizadas en diferentes sistemas de telecomunicaciones ya que permiten grandes velocidades de transmisión, presentan mayor inmunidad al ruido con respecto a sistemas analógicos [1], permiten manejar con mayor facilidad las tasas de error y a su vez tratar de corregirlas [2]. Dentro de estas se tiene la familia de las modulaciones de fase (PSK) que se basa en la variación de fase de una señal portadora, algunas de ellas son: BPSK (modulación por desplazamiento de fase binaria), QPSK (modulación por desplazamiento de fase en cuadratura), 8PSK (modulación por desplazamiento de fase de 8 estados), 16PSK (modulación por desplazamiento de fase de 16 estados); además existen modulaciones diferenciales de fase como lo son DBPSK y DQPSK [3], que serán objeto de estudio en este trabajo. Estas modulaciones presentan un nivel de seguridad de la información, ya que no la llevan en estados absolutos sino que lo hacen a través de la transición de sus estados [4].

Teniendo en cuenta la importancia de los sistemas de comunicaciones digitales, su masiva utilización y sus múltiples utilidades en diferentes aplicativos, se evidencia una línea de conocimiento importante para explotar [2]. Por otra parte debido a los elevados costos para la implementación de dichos sistemas se hace necesaria una solución que permita diseñar y evaluar sistemas digitales, utilizando diferentes tipos de modulaciones para evidenciar el funcionamiento dentro de un entorno académico. Además se pondrá en práctica los conceptos teóricos y se desarrollarán pruebas de desempeño de los sistemas con un costo asequible.

Por ende la finalidad de este proyecto es evaluar y analizar el desempeño de un sistema de comunicaciones con modulaciones DBPSK y DQPSK en hardware reconfigurable. Para cumplir dicho objetivo se debe, inicialmente, determinar la viabilidad del proyecto a través de la determinación de las características mínimas de los sistemas diseñados, para luego implementarlos y por ultimo analizar su desempeño.

Un sistema de comunicaciones digital, en este proyecto, será tratado en el contexto de un enlace de comunicaciones vía radio, en el cual entre el transmisor y el receptor se tiene un canal ideal caracterizado por la ausencia de desvanecimientos u otro tipo de fenómenos que afecte en gran medida al sistema [5].

La estructura de este trabajo se describe a continuación:

En el capítulo 1 se abordan las generalidades y los conceptos de carácter teórico que dan el soporte a este proyecto en cuanto a las modulaciones DBPSK y DQPSK.

A partir de un reconocimiento teórico de las modulaciones mencionadas, se continúa en el capítulo 2 exponiendo la metodología a utilizar para diseñar, implementar y validar los sistemas de comunicaciones; comparando los resultados que serán obtenidos a través de ecuaciones teóricas de desempeño con los de los sistemas modelados en la herramienta *System Generator* de *Xilinx*. Tomando los resultados de simulación se realiza el montaje físico en la FPGA que permite la medición de la BER para comprobar la incidencia del canal AWGN.

En el capítulo 3 se muestran las capacidades mínimas que se necesitan para poder implementar los sistemas; se exponen los resultados de los sistemas implementados y se analiza su desempeño.

Para terminar, en el capítulo 4 se consignan las conclusiones que se obtienen del análisis del presente trabajo y se dan a conocer propuestas de trabajos futuros y recomendaciones a partir de la experimentación.

CAPÍTULO 1. GENERALIDADES

Independientemente del proceso de comunicación que se utilice, existen tres elementos básicos para un sistema de comunicación que son: un transmisor, un receptor y un canal. El transmisor se ubica en cualquier punto en el espacio y su función es generar un mensaje o información, para luego adecuarla a través de cualquier tipo de modulación y finalmente enviarla a través del canal. El receptor se ubica a cierta distancia del transmisor y su función es detectar la información que le es enviada; en él se ven procesos inversos presentes en transmisión[1]; lo anterior para que la información sea descifrada correctamente. El canal se define como el medio físico que se encuentra situado entre el transmisor y el receptor y su función es transportar la información [5].

1.1. SISTEMAS DE COMUNICACIÓN DIGITALES

Un sistema de comunicación digital se caracteriza por tener un generador de información el cual tiene una señal moduladora que es una onda binaria; esta sigue siendo digital sin importar el tipo de tratamiento que se le haga (cambio de fase, amplitud, etc.). Además del generador existe un modulador que se encarga de dar tratamiento a la información para que pueda ser enviada a través del canal; y un demodulador digital que se encarga de recuperar la información a partir de la señal recibida después del canal. Estos módulos fundamentales se pueden observar en la figura 1.1.

Dentro de las modulaciones digitales hay diferentes tipos como ASK, FSK, y PSK. En este proyecto se abordan modulaciones diferenciales que se encuentran dentro de la familia PSK que son DBPSK y DQPSK [6].

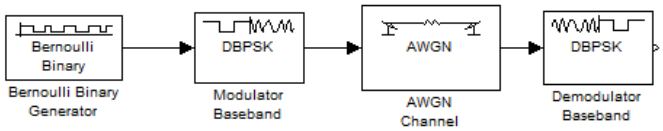

Figura 1.1. Esquema de un sistema de comunicaciones con modulación digital.

1.1.1. MODULACIÓN DPSK.

La modulación por desplazamiento diferencial de fase (DPSK) es una variación de PSK que mantiene un esquema de modulación digital y transmite datos cambiando la fase de referencia de la señal (la onda portadora). También se define como “Pseudo-PSK” porque utiliza la información de fase del símbolo anterior para poder establecer el valor del símbolo actual [4].

[1] Por ejemplo si en el transmisor se utiliza modulación en el receptor demodulación.

Cualquier esquema de modulación digital utiliza un finito número de señales diferentes para representar los datos digitales. PSK utiliza un número finito de fases, cada una asignada a un patrón único de dígitos binarios . Por lo general, cada fase codifica un número igual de bits y cada patrón de bits forma un símbolo que está representado por la fase particular [7].

En general las señales DPSK son menos eficientes que las PSK, ya que los errores en DPSK tienden a propagarse a los símbolos adyacentes debido a las formas de onda de salida de las señales.

Una forma de ver la diferencia entre PSK y DPSK es que en la primera se compara la señal recibida con una referencia limpia; en la segunda se comparan dos señales ruidosas entre sí[1]. Se podría decir que hay el doble de ruido asociado a señalización DPSK en comparación con la señalización PSK. Por consiguiente, como una primera aproximación, se puede estimar que la probabilidad de error para DPSK es aproximadamente dos veces[2] la de PSK; esta degradación disminuye rápidamente con la relación señal a ruido creciente, por lo que la compensación por esta pérdida de rendimiento reduce la complejidad del sistema [4].

1.1.2. MODULACIÓN I/Q

Es una técnica de modulación adecuada a los procesos digitales en la que "I" es el componente "en fase" de la forma de onda, y "Q" representa el componente “en cuadratura”. En sus diversas formas, la modulación IQ es una manera eficiente para la transferencia de información [8].

Se puede escribir la señal como un vector suma de dos señales I y Q como se muestra en la siguiente ecuación [9]:

$$S = [I \;\; Q] \tag{1.1}$$

La figura 1.2 presenta dos formas de representar las señales IQ por la forma espacial y por la forma polar.

[1] Estas modulaciones son más complejas ya que se deben implementar como sistemas con memoria, lo que hace que guarden un símbolo anterior para luego poder compararlo con un símbolo actual, y así obtener la señal de salida.
[2] O estar 3dB por debajo.

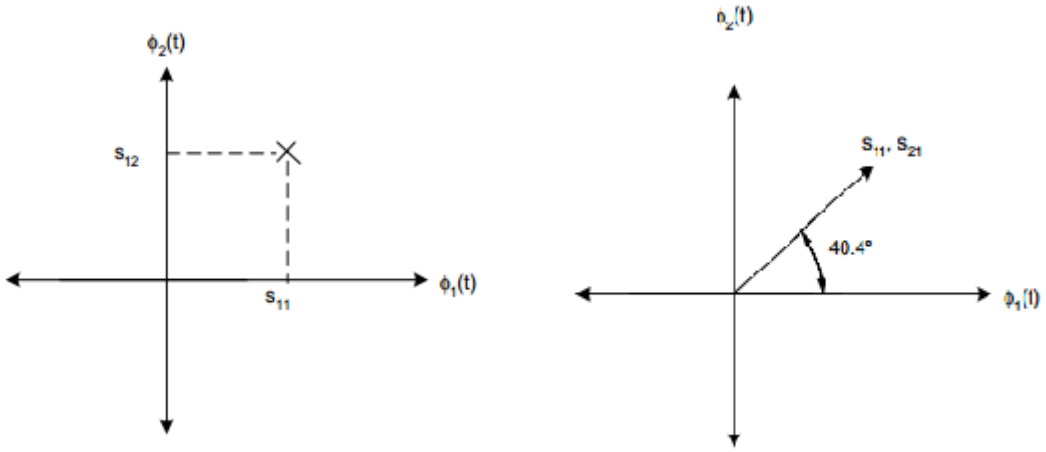

Figura 1.2. (a)Proyección I y Q. (b) Forma Polar

1.1.3. MODULACIÓN DBPSK.

DBPSK (*Differential Binary Phase Shifter Keying*) es una técnica de modulación digital que no maneja estados absolutos, sino que tiene en cuenta la diferencia que tenga la señal de salida. Para realizar esta modulación se necesita saber el bit anterior para luego compararlo con el bit actual [10]. Si bien es cierto que la diferencia entre los dos símbolos es de 180°, cabe aclarar que en la modulación BPSK solo hay cambios de fase cuando hay transiciones de 1 a 0 ó de 0 a 1, mientras que en DBPSK los cambios de fase se realizan a partir de la operación lógica XNOR que implica un símbolo actual y uno pasado, como se muestra en la figura 1.3.

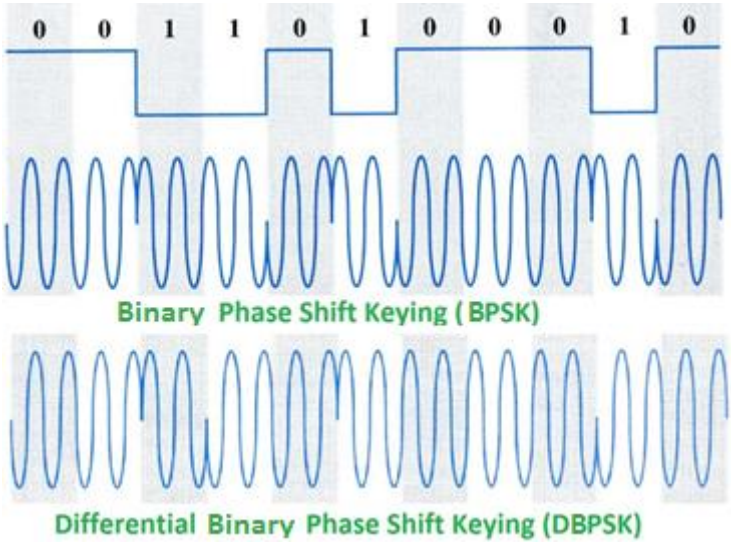

Figura 1.3. Comparación entre BPSK y DBPSK.

La forma más simple de la modulación DPSK es la binaria (DBPSK, *Binary DPSK*), en la cual se utilizan dos valores posibles de fase, es decir, se

emplean sólo dos símbolos con un bit de información cada uno. El diagrama de constelación para DBPSK se muestra en la Figura 1.4. (a). La distancia de cada uno de los símbolos hasta el origen de los ejes representa la amplitud de la señal, la cual no cambia para la modulación PSK, y la distancia angular del vector respecto al semieje positivo I representa la fase de esta. Como es notable en el diagrama de constelación de la figura 1.4 (a), la señal yace en un solo eje (eje en fase *I*) y la diferencia entre ambos símbolos es de 180° [11]. Como se observa en las figuras 1.4 (a) y (b), los diagramas de constelación y fase son los mismos que los de la modulación BPSK. También cabe mencionar que el símbolo, al estar asociado a un solo eje, toma como intrascendente el eje de cuadratura.

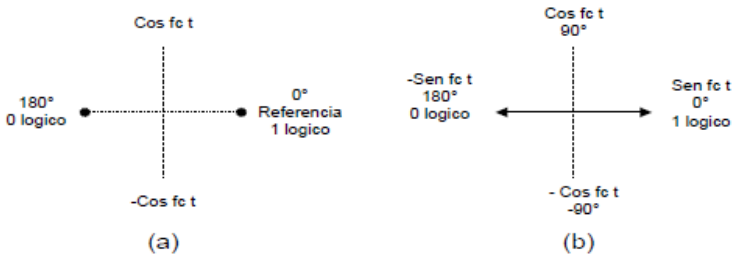

Figura 1.4. Diagramas (a) de constelación y (b)fasorial de DBPSK [3].

1.1.3.1. TRANSMISOR DBPSK.

El modulador DBPSK maneja, al inicio de su esquema en bloques, una compuerta XNOR[1] entre la señal actual de información y el bit de información transmitido anteriormente. La salida de esta operación será la entrada de un modulador PSK convencional como se muestra en la figura 1.5 [10].

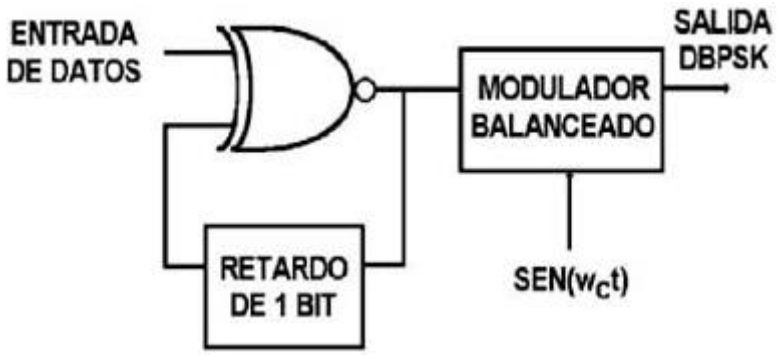

Figura 1.5. Modulador DBPSK.

Para visualizar mejor el funcionamiento se tiene el diagrama de tiempos de la figura 1.6 del modulador DBPSK.

[1] Excluse Nor, es una compuerta lógica muy utilizada en los circuitos digitales.

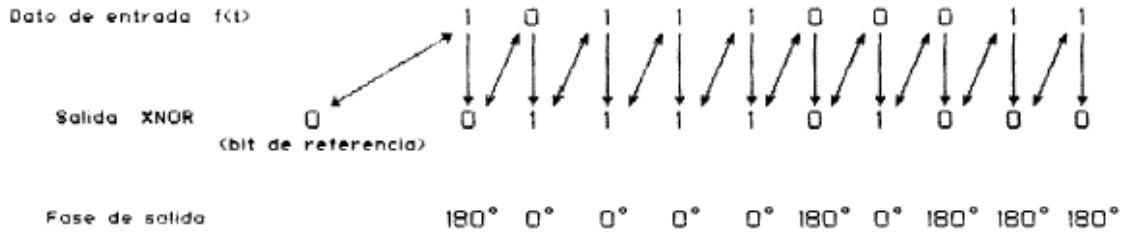

Figura 1.6. Diagrama de tiempos modulación DBPSK [10].

Teniendo en cuenta lo anterior, el tratamiento que se hace en el modulador se resume en la tabla 1.1:

Bit actual	Bit anterior	Salida XNOR	Fase de salida
0	0	1	0
0	1	0	180
1	0	0	180
1	1	1	0

Tabla 1. 1. Comportamiento modulador DBPSK [10]

La fase de salida correspondiente se puede ver utilizando la modulación I/Q; este tipo de modulación utiliza un componente real y otro imaginario para representar la señal como un número complejo, entonces, para obtener una fase de 0, se manda una señal en el componente I con amplitud uno (1), mientras que en el componente Q se manda un cero (0). Así mismo cuando se requiere mandar una señal con fase de 180° el único cambio que se hace es que ahora se envía un -1 por el componente I, mientras que en el componente Q no se hace ningún cambio. Lo anterior se refleja en la tabla 1.2:

Fase de salida	Salida en I	Salida en Q
0	1	0
180	-1	0
180	-1	0
0	1	0

Tabla 1.2. Descripción por componentes de la señal de salida del modulador DBPSK.

1.1.3.2. ANCHO DE BANDA REQUERIDO PARA DBPSK

En un modulador DBPSK, a la señal de entrada del modulador balanceado se le asignan dos posibles valores:

$$x(t)=\begin{cases}+1, & si\ se\ recibe\ un\ 1\ logico\\ -1, & si\ se\ recibe\ un\ 0\ logico\end{cases}$$

En sistemas de comunicaciones que son netamente en banda base, su salida al canal de transmisión se describe como la función x(t) antes mencionada.

Para un modulador teórico que implementa su sección de RF o bloque pasa banda, tiene como entrada la señal x(t) y la procesa multiplicándola por el valor de la señal portadora. Entonces, la señal de salida de este se puede ver descrita por una forma de onda senoidal de la siguiente manera:

$$s(t)=\begin{cases}+sen(2\pi fc*t), & si\ se\ recibe\ un\ 1\ logico\\ -sen(2\pi fc*t), & si\ se\ recibe\ un\ 0\ logico\end{cases}$$

La primera representación muestra una señal que está en fase con el oscilador local; y la segunda una que esta 180° fuera de fase; cada vez que cambia la condición de entrada cambia la salida. En consecuencia para DBPSK la razón de cambio de salida (BAUDIO) es igual a la razón de cambio de entrada (bits/s), al igual que como sucede en BPSK, y el ancho de banda más grande es necesario cuando los datos binarios a la entrada son una secuencia alternada de unos y ceros, para la cual la frecuencia fundamental (fm) es igual a la mitad de la razón de bits (fb[1]/2).

El espectro de salida de un modulador DBPSK es una señal de doble banda lateral con portadora suprimida, en donde las frecuencias laterales superiores e inferiores están separadas de la frecuencia portadora por un valor igual a la mitad de la razón de bit. En consecuencia, el mínimo ancho de banda requerido para permitir el peor caso de la señal de salida del modulador DBPSK es igual a la razón de bit de entrada [12]. La figura 1.7 muestra la fase de salida para una forma de onda DBPSK.

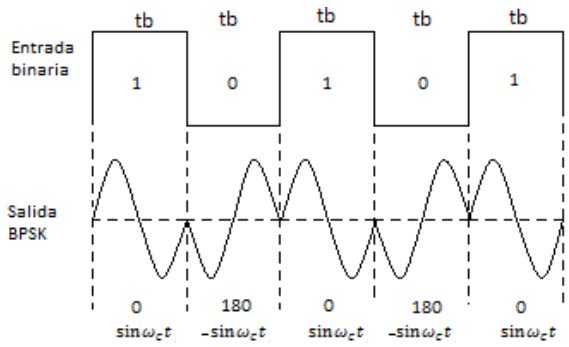

Figura 1.7. Salida de la señal DBPSK modulada en pasabanda [12].

1.1.3.3. RECEPTOR DBPSK

Para este proyecto, un sistema DBPSK se caracteriza por hacer uso de detección coherente, lo que se refiere a que emplea un demodulador que está diseñado para operar sincrónicamente con la señal que se genera en el

[1] Para este caso fb=Rb o tasa de transmisión de bits, ya que DBPSK tiene una eficiencia espectral de 1 bit/Hz

transmisor; por ende los posibles símbolos ya son conocidos. Se dice entonces que este tipo de demodulación es más efectiva que la no coherente [13].

La demodulación se refiere en sí a procedimientos de detección de símbolos en los que viene inmersa la información. Para el proceso de demodulación se debe tener en cuenta la comparación entre bits consecutivos, mostrada en la tabla 1.3, es por esto que se utiliza una compuerta XOR que compare dichos bits y de esta manera logre recuperar el dato transmitido, como se observa en la figura 1.8 [4].

La demodulación se rige por una referencia que viene dada por un estado anterior de la señal recibida; si alguna de las dos señales comparadas esta errada la salida será errada.

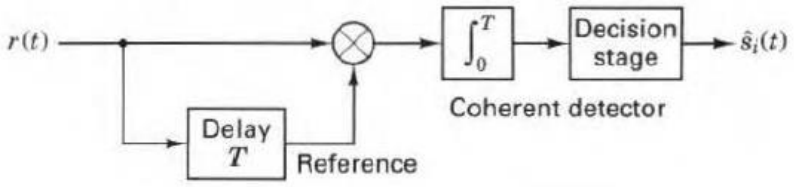

Figura 1.8. Demodulador DBPSK [4].

Bit actual	Bit anterior	Salida XOR
0	0	1
0	1	0
1	0	0
1	1	1

Tabla 1.3. Tabla de verdad decodificador diferencial

1.1.4. MODULACIÓN DQPSK

DQPSK (*Diferential Quadrature Phase Shift Keyting*) es una técnica de modulación digital en donde la información de símbolo se codifica como el cambio de fase de un periodo de símbolo con respecto a uno anterior, en lugar de una fase absoluta como en QPSK. En este caso, el receptor tiene que detectar cambios de fase, como se observa se la figura 1.9, y no el valor absoluto de la fase, lo que evita la necesidad de un portador local sincronizado y reduce la complejidad del sistema [14].

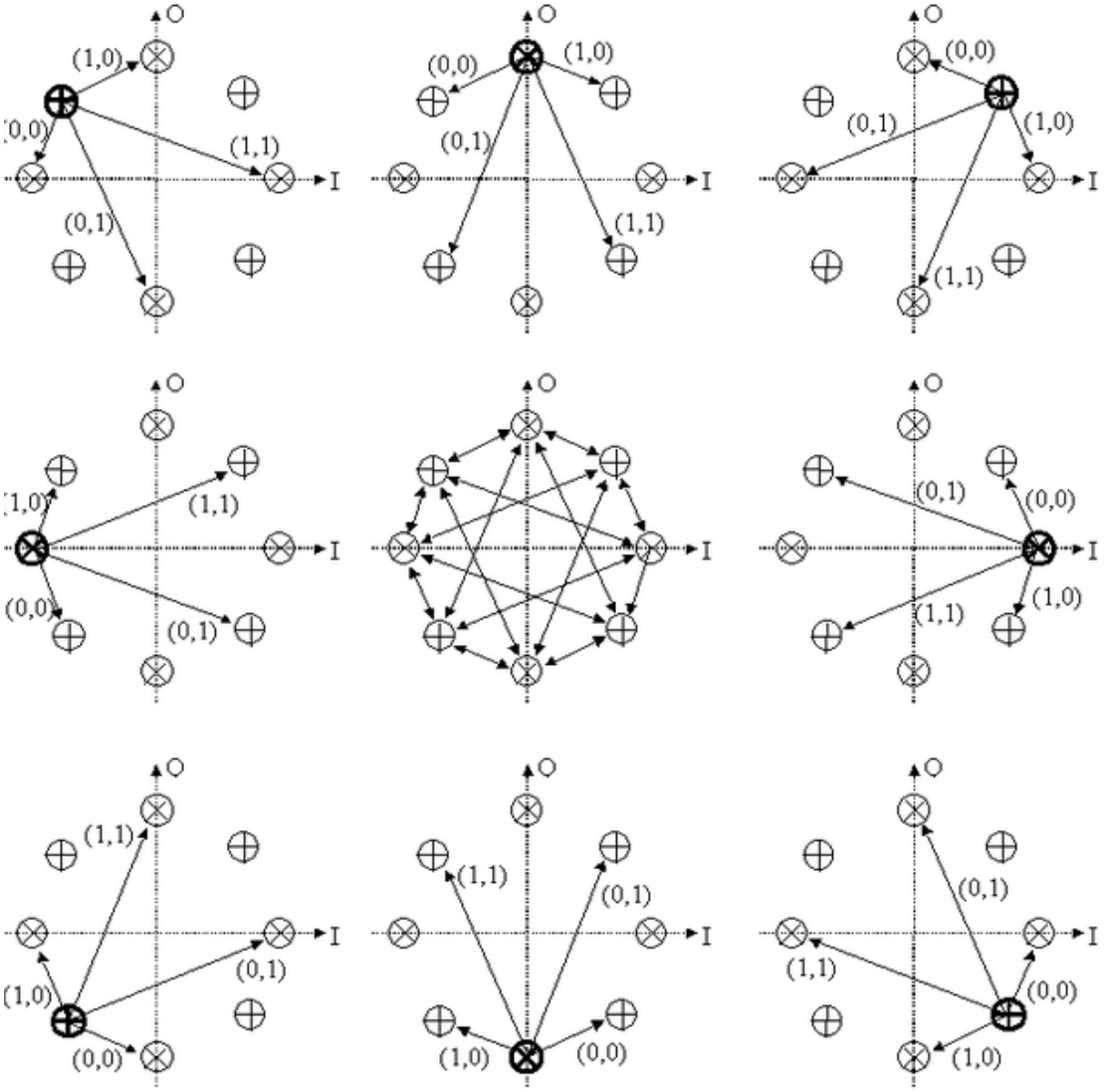

Figura 1.9. Cambios de fase modulación DQPSK [15].

Para aclarar los cambios de fase descritos en la figura 1.9, se tiene la siguiente tabla que combina los cuatro estados basados en codificación GRAY con su respectivo cambio de fase independiente del estado en que se encuentre.

Entrada codificador GRAY	Cambio de fase
00	π/4
01	3π/4
11	- 3π/4
10	- π/4

Tabla 1.4. Tabla de verdad codificador Gray.

1.1.4.1. TRANSMISOR DQPSK

La figura 1.10 muestra el proceso, en un sistema de bloques, de un modulador DQPSK. Inicialmente se cambia el flujo de información binaria de bits a uno de bits bipolar -1 y 1. A continuación, la información se procesa a través del circuito de mapeo, el cual es capaz de convertir el flujo de bits bipolar en pares de bits y asignarlos respectivamente a los canales I (Ich) y Q (Qch). Los codificadores diferenciales convierten de manera separada cada par de bits en un cambio de fase correspondiente a la modulación DQPSK, utilizando las reglas descritas en la tabla 1.3. Posteriormente se aplica un filtro de conformación de impulsos para reducir las señales no deseadas producidas por la salida del codificador diferencial. Finalmente, la salida del filtro conformador de impulsos se convierte en señal analógica y se modula mediante señales RF senoidales: cos(2 π *fc t*) para la señal de Ich y sen(2 π *fc t*) para Qch, donde fc es la frecuencia de la portadora; tanto las señales de Ich y Qch se combinan para producir la señal DQPSK [16].

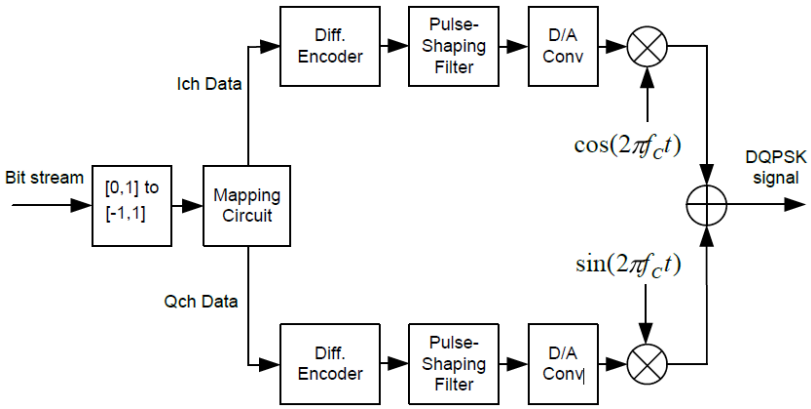

Figura 1.10. Modulador DQPSK [16]

1.1.4.2. ANCHO DE BANDA REQUERIDO PARA DQPSK

Ya que los datos de entrada se dividen en dos canales, la tasa de bits también se divide, es decir, tanto para el canal I como para el Q es fb/2, donde fb es la frecuencia de bit. En esencia, el derivador de bits hace que los bits I y Q tengan el doble de tamaño respecto a los de entrada; en consecuencia, la frecuencia fundamental más alta se presenta a la entrada del modulador y es fb/4. Como resultado, la salida de los moduladores balanceados I y Q requieren de un mínimo de ancho de banda de *Nyquist,* igual a la mitad de la tasa de bits que están entrando, ya que el modulador genera dos bandas laterales [12].

1.1.4.3. RECEPTOR DQPSK

En el demodulador DQPSK se invierte el proceso del modulador; la figura 1.11 muestra el diagrama de bloques del demodulador DQPSK.

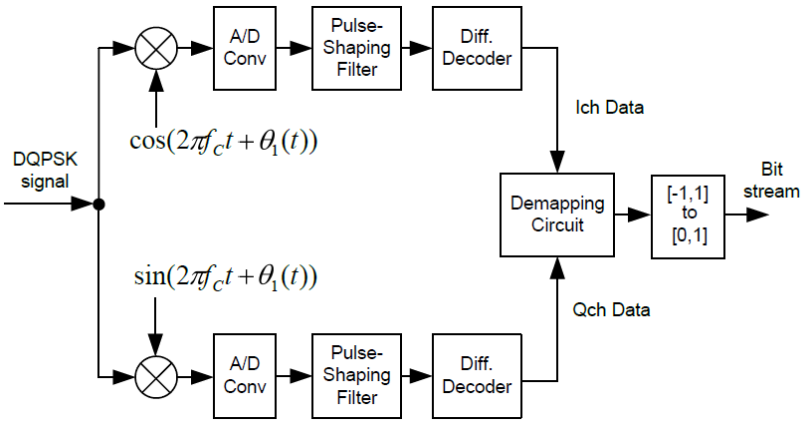

Figura 1.11. Demodulador DQPSK [16]

El demodulador DQPSK elimina primero la frecuencia de la portadora para la recuperación de las señales de los canales I y Q. Los convertidores A / D se encargan de transformar estas señales en formatos digitales, luego pasan a través de un filtro de conformación de impulsos y posteriormente por un decodificador diferencial. Este último genera la información dibit de acuerdo a la tabla 1.5. Finalmente el circuito de demapeo convierte la información contenida en el dibit a un flujo de bits - 1 y 1; la recuperación de los datos transmitidos por el modulador DQPSK se garantiza con la conversión a flujo de bits de 0's y 1's [16].

DIFERENCIA DE FASE DE ENTRADA	Dibit de Salida
π/4	00
3π/4	01
- 3π/4	11
- π/4	10

Tabla 1.5. Tabla de verdad decodificador DQPSK.

1.1.5. CANAL AWGN

La señal de información transmitida en un sistema de comunicaciones se ve afectada por una adición de ruido que va a degradar la señal recibida, como se muestra en la siguiente figura.

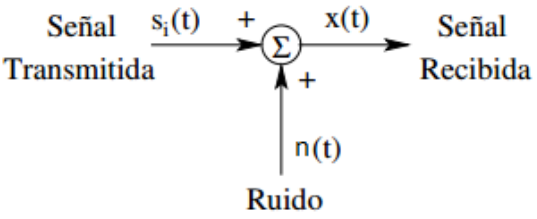

Figura 1.12. Adición del ruido AWGN al sistema [17].

Dado que n(t) es ruido blanco, implica un proceso aleatorio con una densidad espectral de potencia plana (PSD) para todas las frecuencias, como se puede observar en la expresión 1.2, donde se asume una PSD como si fuera estacionaria [17].

$$N(f) = No/2 \qquad -\infty < f < \infty \tag{1.2}$$

Lo anterior implica que un proceso blanco tiene componentes en todo el espectro de frecuencias, por supuesto sólo es una idealización matemática; de acuerdo con el teorema de Wiener-Khinchine, la función de autocorrelación del canal AWGN es [18]:

$$R_{(T)} = E_{(n(t)n(t-T))} = \int_{-\infty}^{\infty} N_{(f)} e^{2\pi jT}\, df$$

$$= \int_{-\infty}^{\infty} \frac{N_o}{2} e^{2\pi jT}\, df = \frac{N_o}{2}\delta(T) \tag{1.3}$$

donde δ(T) es la función delta de Dirac. La expresión (1.3) demuestra que los componentes del ruido no están correlacionados, sin importar lo cerca que estén en tiempo; estos también son independientes, debido a que el proceso es gaussiano [18].

En cualquier instante de tiempo, la amplitud de n(t) obedece a una función de densidad de probabilidad gaussiana dada por:

$$p(n) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\left(-\frac{n^2}{2\sigma^2}\right)} \tag{1.4}$$

La varianza para este caso es:

$$E(n^2) = \frac{No}{2} \tag{1.5}$$

Como se observa en el anexo 2.

Se supone siempre que el receptor esta sincronizado con el transmisor, a nivel de símbolo, lo que significa que el receptor sabe los instantes temporales de cambio de símbolo. En algunos casos, también se va a suponer que el receptor está en sincronismo de fase con el transmisor, por lo que el receptor emplea detección coherente. En otros casos, se debe utilizar detección no coherente, debido a que el transmisor y el receptor no están en sincronismo de fase [18].

Se sabe que un generador AWGN es una fuente de ruido normalizado con media 0 y varianza 1, por lo que si se multiplica la señal entregada por una constante denominada sigma, se conoce la cantidad de ruido que se está introduciendo al sistema [19].

$$Generador\ de\ ruido\ AWGN: N(media, varianza) = N(0,1) \tag{1.6}$$

$$\sigma N(0,1) = N(0,\sigma^2) \tag{1.7}$$

Sin embargo para observar curvas de desempeño de los sistemas de comunicaciones, se debe contar con los valores de BER y de Eb/No, por lo que la solución es encontrar una relación entre el valor de sigma σ y el valor de energía de bit a densidad de ruido, así:

$$\left(\frac{Eb}{No}\right) = \left(\frac{p\ señal}{p\ ruido}\right)\ \left(\frac{1}{\log_2 M}\right) \tag{1.8}$$

donde M es el número de estados de la constelación.

Además se sabe que

$$SNR = \frac{p\ señal}{p\ ruido} \tag{1.9}$$

Por lo que se tiene:

$$\left(\frac{Eb}{No}\right) = SNR * \left(\frac{1}{\log_2 M}\right) \tag{1.10}$$

En dB:

$$\left(\frac{Eb}{No}\right)[dB] = SNR\ [dB] + 10log\left(\frac{1}{\log_2 M}\right)[dB] \tag{1.11}$$

Despejando SNR de la ecuación (1.11):

$$SNR\ [dB] = \left(\frac{Eb}{No}\right)[dB] - 10log\left(\frac{1}{\log_2 M}\right)[dB] \tag{1.12}$$

Como la señal está normalizada en potencia se tiene que la $p\ señal = 1$, por lo que la ecuación (1.9) será:

$$SNR = \frac{1}{p\ ruido} \tag{1.13}$$

Como se utilizan 2 fuentes de ruido AWGN (real e imaginaria) la potencia de ruido es:

$$p\ ruido = {\sigma_R}^2 + {\sigma_I}^2 \tag{1.14}$$

Y por lo tanto:

$$SNR = \frac{1}{{\sigma_R}^2 + {\sigma_I}^2} \tag{1.15}$$

Suponiendo $\sigma_R = \sigma_I$, se tiene:

$$SNR = \frac{1}{2\sigma^2} \tag{1.16}$$

Despejando σ de la ecuación (1.16):

$$\sigma = \frac{1}{\sqrt{2 * SNR}} \tag{1.17}$$

La ecuación (1.17) sirve para hallar el valor de sigma asociado a cada variación de Eb/No; este valor de sigma debe tenerse en cuenta para el correcto diseño de un canal de ruido AWGN.

1.1.6. MEDIDAS DE DESEMPEÑO

El desempeño en telecomunicaciones se define como la calificación que se le puede dar a un sistema, teniendo en cuenta la calidad que brinde, es por esto, que se tienen diferentes mecanismos para medir el desempeño que permiten evaluar sistemas y mantener comparaciones entre unos y otros [20].

1.1.6.1. BER

La tasa de error de bit es la relación del número de bits recibidos incorrectamente, en comparación con el número total de bits transmitidos.

$$BER = \frac{\#\ de\ bits\ erroneos}{\#\ de\ bits\ totales\ transmitidos} \tag{1.19}$$

La BER es una de las maneras más importantes para determinar la calidad de un sistema de transmisión digital [21]; por lo que la totalidad de los sistemas de comunicaciones que utilizan modulaciones digitales describen su desempeño en la BER; este parámetro por lo general está dado en términos de la función Q, que describe la probabilidad de error complementaria y se define así [19]:

$$Q(x) = \frac{1}{2}\ erfc\left(\frac{x}{\sqrt{2}}\right) \tag{1.20}$$

1.1.6.2. PROBABILIDAD DE ERROR TEÓRICA EN DBPSK

Para el sistema DBPSK, se obtiene la Pb (probabilidad de error de bit) a partir de la demostración que se encuentra en el apéndice 1. Esta probabilidad está

dada a partir de un sistema que utiliza detección coherente y su ecuación característica se muestra a continuación [4]:

$$Pb = 2Q\left(\sqrt{\frac{2Eb}{No}}\right)\left[1 - Q\left(\sqrt{\frac{2Eb}{No}}\right)\right] \quad (1.21)$$

Al reemplazar la ecuación (1.20) en (1.21) la probabilidad de error es igual a:

$$Pb = erfc\left(\sqrt{\frac{Eb}{No}}\right)\left[1 - \frac{1}{2}erfc\left(\sqrt{\frac{Eb}{No}}\right)\right] \quad (1.22)$$

En la figura 1.13 se muestra gráficamente la representación de la BER teórica para el caso del sistema DBPSK, que se obtiene a partir de la ecuación (1.22).

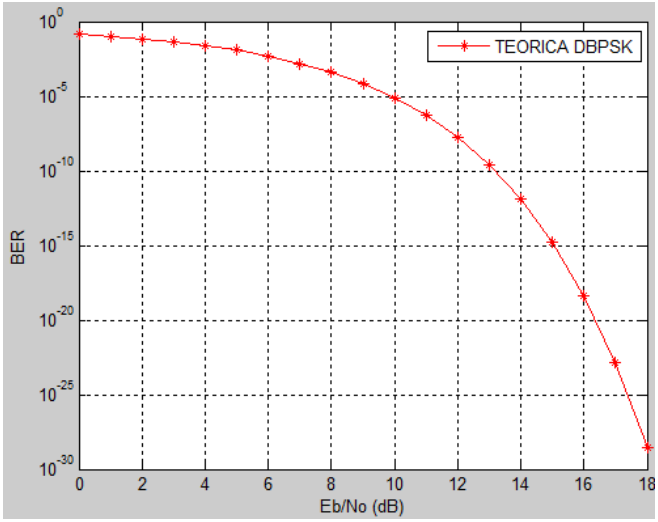

Figura 1.13. BER teórica en DBPSK.

1.1.6.3. PROBABILIDAD DE ERROR TEÓRICA EN DQPSK

Para DQPSK se tiene la ecuación (1.24) que describe la Ps (probabilidad de error de símbolo) de este sistema. Ps es una forma de expresar el rendimiento de un sistema y se define como la relación entre la cantidad de símbolos erróneos por la totalidad de símbolos enviados, como se muestra a continuación:

$$SER = \frac{\#\ de\ simbolos\ erroneos}{\#\ de\ simbolos\ totales\ transmitidos} \quad (1.23)$$

Para este caso, al igual que en la anterior modulación, se tiene una base teórica de detección coherente, como se observa en la siguiente ecuación.

$$Ps = 4Q\left(\sqrt{\frac{2Eb}{No}}\right) - 8Q^2\left(\sqrt{\frac{2Eb}{No}}\right) + 8Q^3\left(\sqrt{\frac{2Eb}{No}}\right) - 4Q^4\left(\sqrt{\frac{2Eb}{No}}\right) \quad (1.24)$$

o lo que es lo mismo, en términos de la función complementaria de error, como se observa a continuación [22]:

$$Ps = 2 * erfc\left(\sqrt{\frac{Eb}{No}}\right) - 2 * erfc^2\left(\sqrt{\frac{Eb}{No}}\right) + erfc^3\left(\sqrt{\frac{Eb}{No}}\right) - \frac{1}{4}erfc^4\left(\sqrt{\frac{Eb}{No}}\right) \quad (1.25)$$

Para validar los datos del sistema con modulación DQPSK es necesario hacer una conversión de Ps a Pb, que es el parámetro genérico de rendimiento de un sistema de comunicación digital. Si un sistema de comunicaciones utiliza codificación Gray[1] en su modulador, la Pb (probabilidad de error de bit) se define así [23]:

$$Pb = \frac{Ps}{\log_2 M} \quad (1.26)$$

donde M es el orden de la modulación.

Entonces, para el sistema DQPSK la Pb es igual a:

$$Pb = \frac{Ps}{2} \quad (1.27)$$

Reemplazando la ecuación (1.27) en (1.25) se tiene que:

$$Pb = erfc\left(\sqrt{\frac{Eb}{No}}\right) - erfc^2\left(\sqrt{\frac{Eb}{No}}\right) + \frac{1}{2}erfc^3\left(\sqrt{\frac{Eb}{No}}\right) - \frac{1}{8}erfc^4\left(\sqrt{\frac{Eb}{No}}\right) \quad (1.28)$$

En la figura 1.14 se muestra gráficamente la representación de la BER teórica, para el caso del sistema DQPSK, que se obtiene a partir de la ecuación (1.28).

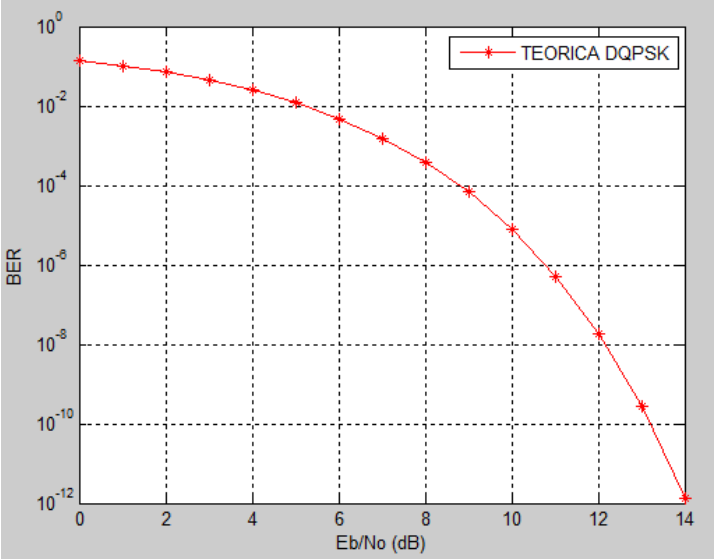

Figura 1.14. BER teórica DQPSK.

[1] La codificación gray se utiliza para asegurar que entre cada símbolo exista solamente un cambio de un bit.

A partir de las curvas teóricas de BER, para los sistemas con modulaciones DBPSK y DQPSK, se hace una comparación para ver cuál es el sistema que presenta mejor rendimiento, dando como resultado que ambos sistemas tienen un rendimiento parecido, lo que se puede observar en la figura 1.15; esto tiene sentido ya que BPSK y QPSK, sus modulaciones madre, presentan rendimientos iguales [20].

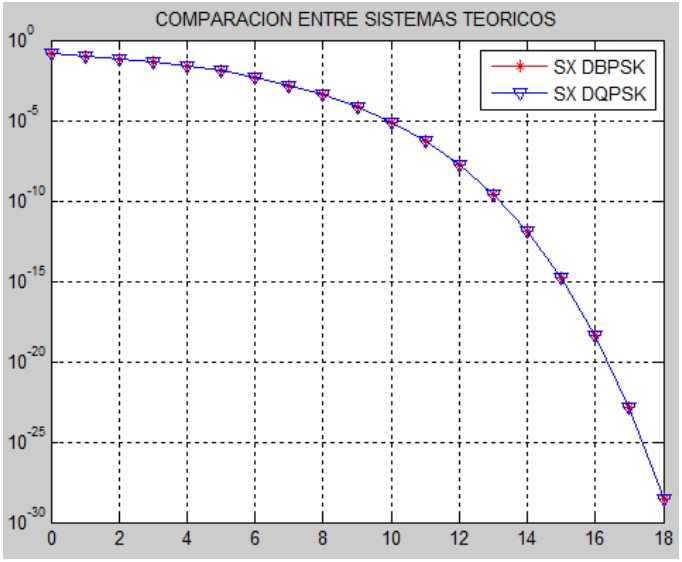

Figura 1.15. BER DBPSK y DQPSK.

1.1.7. ARQUITECTURA DE PUNTO FIJO

La arquitectura de punto fijo fue introducida a comienzos de la década de 1980, y está basada en una representación que contiene una cantidad fija de dígitos después del punto decimal. Al no requerir de Unidad de Punto Flotante (FPU), la mayoría de los chips DSP de bajo costo utilizan esta arquitectura, aunque en determinados casos esta alternativa ofrece también mejor desempeño así como mayor exactitud [24].

Los bits a la izquierda del punto decimal se denominan bits de magnitud, estos representan valores enteros, en cambio los bits a la derecha representan valores fraccionales[1] (potencias inversas de 2).

Se puede calcular la representación de punto fijo de un número positivo cualquiera, de la siguiente manera:

$$2^{m-1} = \frac{1}{2^f} \tag{1.29}$$

donde m, representa los bits de magnitud y f los bits fraccionales. Igualmente los números negativos se representan por:

$$-2^{m-1} \tag{1.30}$$

[1] Es decir que el primer bit fraccional es ½, el segundo es ¼, el tercero es 1/8, etc.

1.1.8. APLICACIONES DE LA MODULACIÓN DPSK[1]

La modulación DBPSK se ha hecho presente en diversas aplicaciones en sistemas de comunicación digital, entre las cuales se encuentran los sistemas de fibra óptica a velocidades de 2.5, 5, 10 y 50 Gbps. Mientras que DQPSK se utiliza en sistemas de fibra óptica para trabajar con velocidades superiores a 100 Gbps [25].

Además, en redes inalámbricas de área local es relevante la utilización de las modulaciones DBPSK, para transmitir a una velocidad de 1Mbps, y DQPSK, para trabajar a una velocidad de 2 Mbps. También es importante mencionar que Bluetooth 2 maneja DQPSK para una velocidad de 2Mbps [12].

1.2. HERRAMIENTAS HW

Dentro de las herramientas hardware se destaca la utilización de FPGA's en el diseño de sistemas digitales en el campo del hardware reconfigurable. En este proyecto se usa una FPGA como herramienta de despliegue, puesto que presenta altas capacidades; por ende es una muy buena opción de implementación de sistemas de comunicación digital y una buena alternativa para mejorar sistemas haciéndolos más robustos. Además teniendo en cuenta que las FPGA's cada día incrementan sus capacidades, y variedad disponible en el mercado, facilitan la implementación de sistemas más complejos e innovadores.

1.2.1. FPGA

La tecnología de arreglos de compuertas programables en campo (FPGA) continúa siendo impulsada debido a su flexibilidad y moderado costo. Además, es útil en la implementación de sistemas innovadores en diferentes campos de la ingeniería [26].

Las FPGA's utilizan bloques de lógica pre-construidos y recursos para ruteo programables; estos permiten configurar los chips de la herramienta para implementar funcionalidades personalizadas en hardware sin tener que utilizar una tablilla de prototipos o un cautín. Para implementar un sistema sólo se necesita desarrollar tareas de cómputo digital en software y compilarlas en un archivo de configuración o "bitstream", que contenga información de cómo deben conectarse los componentes. También, las FPGA's son completamente reconfigurables y al instante adquieren una nueva "personalidad", al ser compilada con una diferente configuración de circuitos.

[1] DPSK se define como la familia de las modulaciones PSK diferenciales, o sea que contiene a DBPSK, DQPSK, D8PSK, etc.

Anteriormente sólo los ingenieros con un profundo entendimiento de diseño de hardware digital podían trabajar con la tecnología FPGA. Sin embargo, el aumento de herramientas de diseño de alto nivel está cambiando las reglas de programación de FPGA's, apoyándose en nuevas tecnologías que convierten los diagramas a bloques gráficos o hasta el código ANSI C a circuitos de hardware digital.

1.2.2. ARQUITECTURA DE LA SPARTAN 3A DE XILINX™

Las FPGA Spartan 3A de Xilinx están conformadas por un conjunto de Bloques Lógicos Configurables ("*Configurable Logic Blocks*": CLBs) rodeados por un perímetro de Bloques Programables de entrada/salida ("Programmable Input/Output Blocks": IOBs). Estos elementos funcionales están interconectados por una jerarquía de canales de conexión ("Routing Channels"), dicha jerarquía incluye una red de baja capacitancia para la distribución de señales de reloj de alta frecuencia. Adicionalmente, el dispositivo cuenta con 24 bloques de memoria RAM de 2Kbytes de doble puerto, cuyos anchos de buses son configurables, así mismo dispone de 12 bloques de multiplicadores dedicados de 18 X 18 bits [27].

Los cinco elementos funcionales programables que componen la Spartan 3A son los siguientes:

Bloques de entrada/salida ("Input/Output Blocks" – IOBs): controlan el flujo de datos entre los pines de entrada/salida y la lógica interna del dispositivo. Soportan flujo bidireccional más operación tri-estado y un conjunto de estándares de voltaje e impedancia controlados de manera digital.
Bloques Lógicos configurables ("*Configurable Logic Blocks*" – CLBs): contienen *Look-Up Tables* (LUTs) basadas en tecnología RAM para implementar funciones lógicas y elementos de almacenamiento que pueden ser usados como *flip-flop*s o como *latches*.
Bloques de memoria RAM ("*Block RAM*"): proveen almacenamiento de datos en bloques de 18 Kbits con dos puertos independientes cada uno.
Bloques de multiplicación: aceptan dos números binarios de 18 bits como entrada y entregan uno de 36 bits.
Administradores digitales de reloj ("*Digital Clock Managers*" – DCMs): estos elementos proveen funciones digitales auto calibradas que se encargan de: distribuir, dividir y multiplicar las señales de reloj de todo el circuito. Adicionalmente retrasan la señal de manera arbitraria en pocos grados para desfasarla en 90, 180, y 270 grados.

Los elementos descritos con anterioridad están organizados como se observa en la figura 1.16. Inicialmente se muestra un anillo de IOBs que rodea un arreglo regular de CLBs, este arreglo es atravesado por una columna de bloques de memoria RAM de 18 Kbit, cada uno de los cuales está asociado

con un multiplicador dedicado. Por último, los DCMs están ubicados en los extremos de dichas columnas.

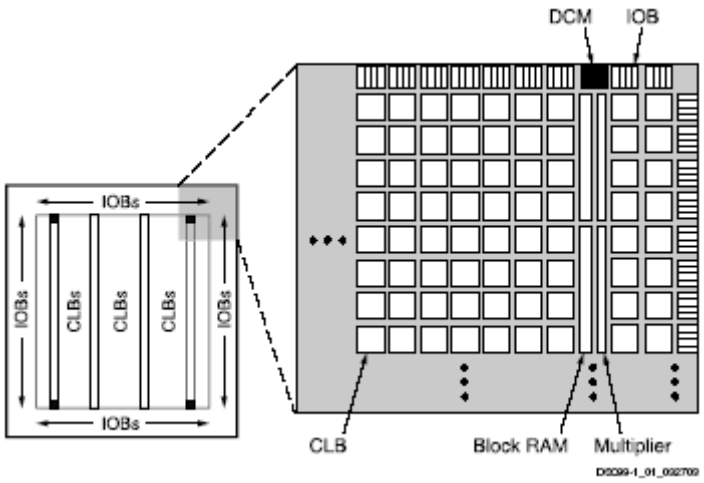

Figura 1.16. Arquitectura de la Spartan III [27]

En la figura 1.17 se muestra la FPGA *Spartan* 3A que se utiliza para este proyecto, donde se distinguen los diferentes puertos de interconexión, pantalla de despliegue y demás elementos que constituyen este hardware reconfigurable.

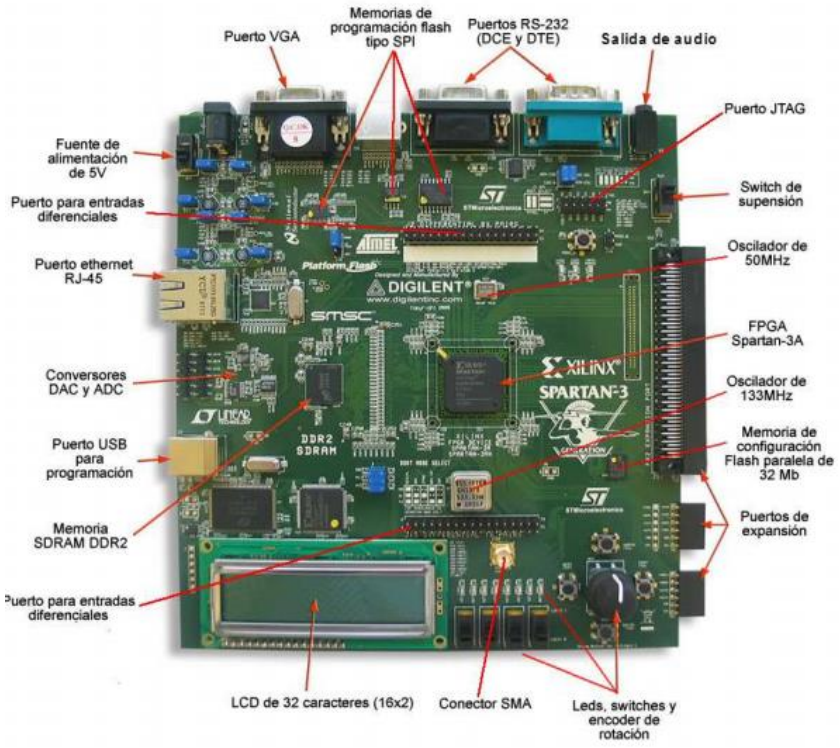

Figura 1.17. FPGA Spartan 3A [28].

1.2.3. EL DISPOSITIVO XC3 S700A

La FPGA *Spartan* 3A XC3 S700A posee un sistema de 700.000 compuertas lógicas y presenta unos bloques funcionales que se resumen en la siguiente tabla [27].

FPGA XC3S700A		
Sistema de compuertas		700 k
Celdas lógicas		13248
Arreglo de CLB's	Filas	48
	Columnas	32
	CLB's totales	1472
	Slices Totales	5888
RAM Distribuida		92 kbits
Bloque de Memoria RAM		360 kbits
Multiplicadores Dedicados		20
DCM's		8
Entradas/Salidas de usuario		372

Tabla 1.6. Resumen de Bloques funcionales Spartan 3A [27].

1.2.4. FLUJO DE DISEÑO

En este proyecto se diseñan e implementan sistemas digitales basados en FPGA's usando la herramienta *System Generator* de Xilinx. Para cumplir con las anteriores tareas, se recomienda seguir el flujo de diseño indicado en la figura 1.18; este flujo empieza con el diseño lógico, el cual consiste en describir mediante un lenguaje HDL (*Hardware Description language)* el circuito o sistema digital. La herramienta *System Generator*® también permite realizar esta descripción mediante bloques esquemáticos, que después son traducidos a un lenguaje HDL.

Después de haber diseñado el sistema digital, este debe pasar por el proceso de síntesis, el cual consiste en transformarlo desde un nivel de abstracción alto a uno bajo, en el cual se especifican los componentes de hardware y sus interconexiones. El proceso de síntesis lo realiza el software de manera automática, en este caso, el paquete de software *System Generator®.* El resultado obtenido consiste en un *Netlist*, que es un archivo que contiene los elementos de hardware y sus interconexiones; este viene a ser la representación a nivel de compuertas del sistema descrito en el lenguaje HDL seleccionado. Al finalizar el proceso de síntesis se genera un archivo que tiene un formato NGC (*Native Generic Circuit*).

Una vez se ha generado el archivo NGC comienzan los procesos de implementación: el primero es la optimización; en este proceso se minimizan los retardos de interconexión y retardos relacionados con los niveles lógicos. El segundo es el proceso de *translate* en el cual el archivo de netlist original con formato NGC se convierte a un formato estándar llamado NGD (Native Generic Database). El tercero es el de *Mapping* que consiste en mapear el contenido del archivo NGD en los componentes reales que posee el FPGA seleccionado para el diseño. Después, se ejecuta el proceso de *Placement* encargado de posicionar los componentes mapeados en el proceso previo en lugares específicos del FPGA; y finalmente, el proceso de *Routing* que se encarga de interconectarlos. Los procesos de *Placement* y *Routing* producen una carga computacional considerable por lo que su tiempo de finalización puede durar, dependiendo de la complejidad del diseño y características del PC, hasta decenas de minutos [27].

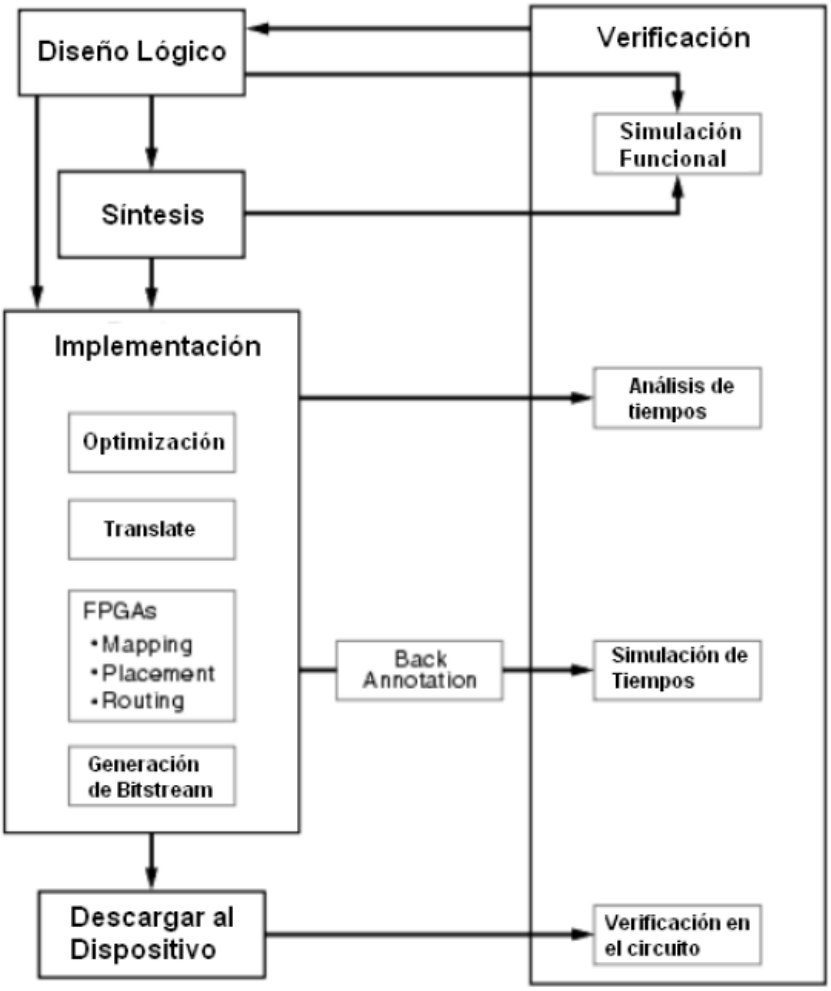

Figura 1.18. Flujo de diseño [27].

CAPÍTULO 2. MODELADO, SIMULACIÓN E IMPLEMENTACIÓN

2.1. METODOLOGÍA DE SIMULACIÓN

La metodología de simulación adoptada para llevar a cabo el desarrollo del proyecto es una adaptación de las descritas por Zeidman [29] y Astaiza et al. [30], la cual consta de las fases que se muestran en la figura 2.1.

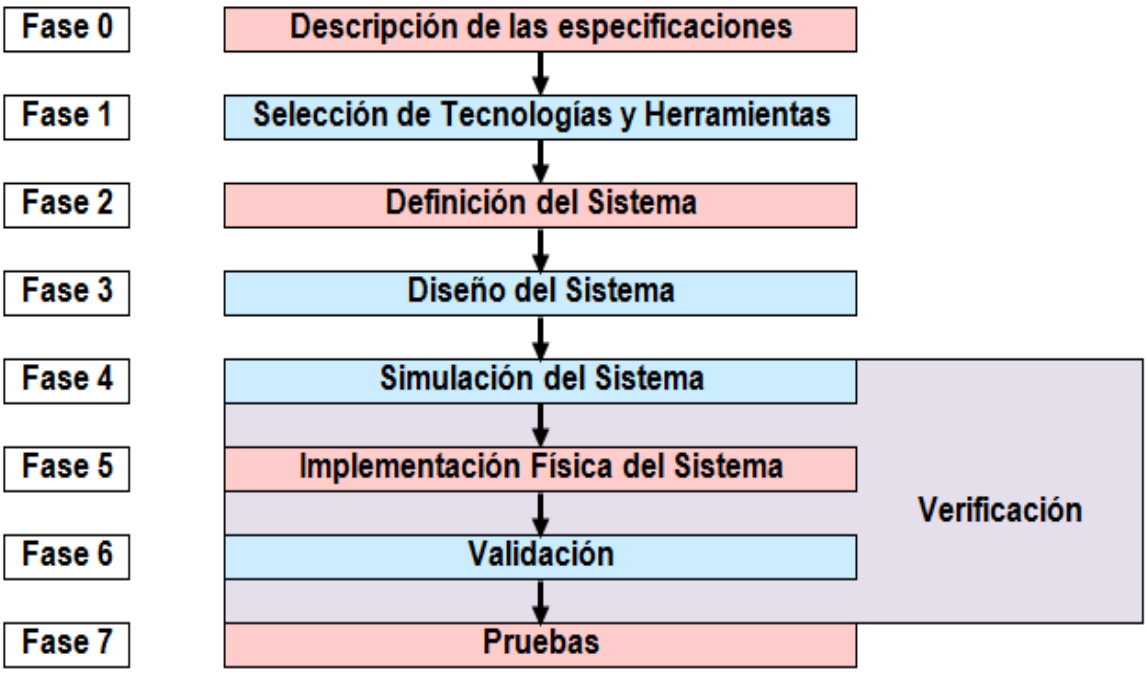

Figura 2. 1. Diagrama de flujo de la metodología empleada para la simulación

A continuación se explica detalladamente cada fase de la metodología implementada.

2.2. DESCRIPCIÓN DE LAS ESPECIFICACIONES

En este trabajo de grado se diseñó, modeló e implementó un sistema de comunicaciones digital con modulaciones DBPSK y DQPSK. Para el diseño y el modelamiento de los sistemas fue necesario un software que permitió hacer simulaciones, dando lugar a la utilización del método de ensayo y error para que fuera posible conseguir los modelos teóricos de las modulaciones en banda base; esta operación se facilitó al trabajar en una interfaz amigable y muy utilizada, por lo que fue requerido un software de diseño de fácil manejo.

Antes de proseguir a la implementación, fue necesario la validación de los modelos obtenidos; proceso que se hace a través de comparaciones de

rendimiento teórico de cada modulación. La implementación se hace a través de la síntesis de un modelo, a partir de la cual se requiere conseguir un archivo que es el encargado de programar el hardware a utilizar. Como paso final se obtuvieron datos de los sistemas implementados que fueron objeto de comparación con los resultados teóricos y además con los obtenidos a través de la simulación.

2.3. SELECCIÓN DE HERRAMIENTAS HARDWARE Y SOFTWARE

Se sabe que la complejidad de trabajar con un sistema de comunicaciones hardware requiere de pruebas para poder establecer sus niveles de desempeño; por lo que se necesita la disponibilidad de un equipo netamente hardware que cumpla la función de un sistema de comunicación y además que sea de tipo reconfigurable y flexible.

Por lo anterior y teniendo en cuenta los costos de implementación hardware de sistemas de comunicaciones, se considera la importancia de realizar análisis de desempeño real de diferentes sistemas desarrollados en plataformas asequibles; una buena alternativa es la utilización de hardware reconfigurable como lo son las FPGA's para implementar y evaluar el desempeño de diferentes modelos de sistemas de comunicación digital. Ya que se dispone de un hardware que satisface las necesidades de implementación y configuración se da como un hecho la elección de la FPGA *Spartan* 3A como hardware de despliegue.

El software elegido para la realización del proyecto fue el paquete *Xilinx*, el cual cuenta con los suficientes elementos para diseñar, modelar e implementar los sistemas requeridos. *System Generator* es uno de los elementos del paquete *Xilinx*, este fue utilizado para diseñar y modelar el comportamiento de los sistemas digitales, ya que es fácil de usar, utiliza un lenguaje de programación de muy alto nivel y además presenta compatibilidad con el hardware disponible; así entonces se facilitó la tarea del diseño. Este software tiene compatibilidad con una plataforma de despliegue muy utilizada y a la vez muy precisa como lo es *Simulink* de *Matlab®,* a través de la cual se da la posibilidad de modelar un sistema de comunicaciones con bloques funcionales que ya están creados. Por ende, *System Generator* es capaz de obtener resultados confiables de cada simulación, mientras que para la obtención de los resultados teóricos se usa la ayuda del *Simulink* y del *Bertool*[1] de *Matlab®*.

Los resultados fueron datos de BER obtenidos teóricamente y en simulación, los cuales indican el desempeño de un sistema de comunicación digital; estos se compararon para lograr un diseño exacto de los sistemas. *Xilinx* como

[1] A partir del cual se obtienen los resultados y graficas de BER en términos del Eb/No.

proveedor brinda un software para sintetizar el modelo de bloques a un código VHDL y otro para implementar el modelo en la FPGA. Para un mejor entendimiento se hará una breve explicación de cada elemento software de *Xilinx* que se utilizó en este proyecto y además se aclararán conceptos del software de *Matlab*® con su interfaz de *Simulink,* en los anexos 4 y 5 respectivamente.

2.4. DISEÑO DE LOS SISTEMAS

En esta fase se presenta el diseño de dos modelos de sistemas de comunicaciones, cada uno de ellos con una modulación de fase distinta, ya sea DBPSK o DQPSK en banda base, ambas con un canal AWGN; además se analiza detalladamente todo el proceso que atraviesa la señal que proviene desde la fuente de información; estos sistemas diseñados serán implementados sobre hardware reconfigurable, obteniéndose de los mismos el valor de la BER para diferentes valores de Eb/No.

Es importante mencionar que no solo existe la BER como parámetro de desempeño puesto que existen otros como:

- SER[1], este parámetro también pudo haber sido utilizado para medir el desempeño de estos sistemas, pero por definición del macroproyecto "diseño e implementación de un prototipo de comunicación de datos basado en hardware reconfigurable fase 1" se utilizó la BER como parámetro estándar de medida de desempeño.

- BLER[2], este parámetro no podía ser utilizado ya que la información en los sistemas de comunicación no estaba agrupada en tramas, puesto que se hacía tratamiento bit a bit.

- Throughput[3], este parámetro no podía ser utilizado puesto que al trabajar en banda base no se tenía en cuenta aspectos como el ancho de banda, ni demás limitaciones del sistema, ni mucho menos el manejo de enrutamiento y tráfico de datos entre varios sistemas.

Por lo tanto la BER es el parámetro idóneo para medir el desempeño de los sistemas, puesto que en estos sistemas se hizo tratamiento bit a bit y se tuvieron en cuenta curvas de referencia teóricas.

[1] Tasa de error de símbolo.
[2] Tasa de error de bloque.
[3] Tasa efectiva de transmisión de datos.

2.4.1. MODELO DBPSK

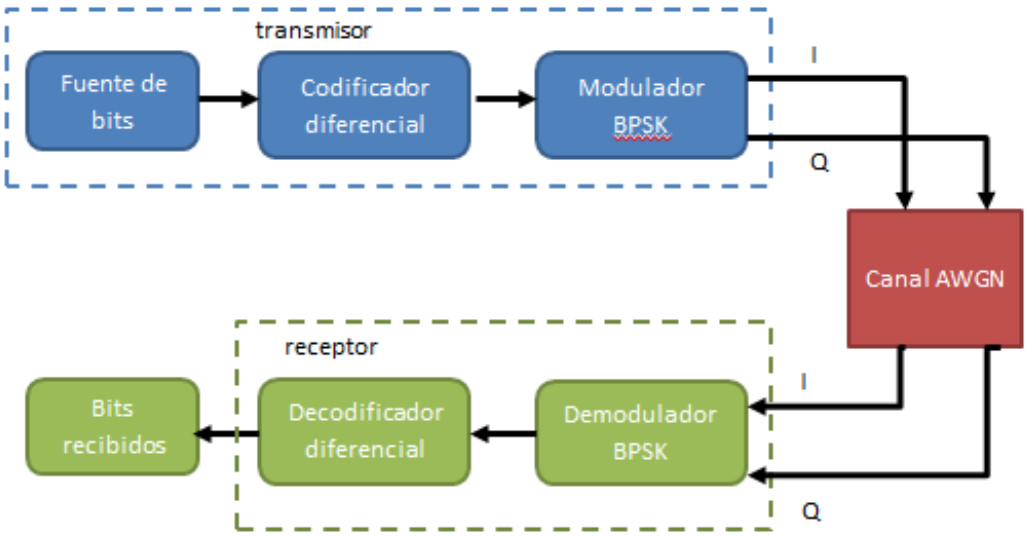

Figura 2.2. Modelado DBPSK.

Subsistema 1.Transmisor

Inicialmente este subsistema está conformado por un generador de bits pseudoaleatorio y equiprobable; posterior a este, se tiene un codificador diferencial que se encarga de realizar una comparación entre los bits actuales y los anteriores, teniendo como referencia el comportamiento teórico que se puede observar en la tabla 1.1. Por último se encuentra un modulador BPSK típico, por lo que se genera un diagrama de constelación igual a BPSK, como se puede observar en la figura 2.3.

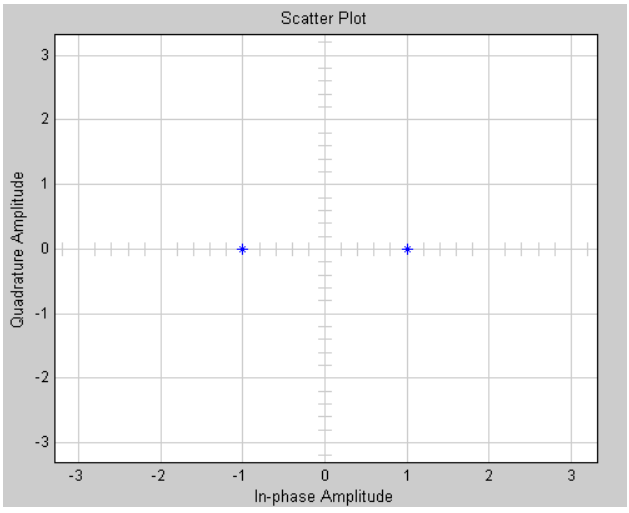

Figura 2.3. Diagrama de constelación DBPSK en transmisión.

Subsistema 2. Canal AWGN

Este subsistema cumple la función de ser el medio de transmisión en el cual transita la información, para este caso es un canal AWGN que posee ruido aditivo gaussiano; para esta modulación se adapta el modelado del canal como se hace en la sección 1.1.5, considerando un orden de modulación 2, o lo que es lo mismo modelar el canal con el parámetro M fijado en 2; por consiguiente, se obtienen los siguientes valores de desviación estándar (σ), consignados en la tabla 2.1.

Eb/No[dB]	SNR(dB)	Desviación estándar (σ)
0	0	0.707106781
1	1	0.630209582
2	2	0.561674881
3	3	0.500593264
4	4	0.446154216
5	5	0.397635364
6	6	0.354392891
7	7	0.315852997
8	8	0.281504279
9	9	0.250890953
10	10	0.223606797
11	11	0.199289768
12	12	0.177617192
13	13	0.158301489
14	14	0.141086351

Tabla 2.1. Valores de desviación estándar para cada uno de los valores de Eb/No usado en DBPSK.

Subsistema 3. Receptor

En primera instancia se utiliza un demodulador BPSK, el cual se encarga de pasar la señal compleja a formas de onda binaria, para posteriormente decodificar los bits, a partir de estados actuales y anteriores. En el diagrama de constelación de la figura 2.4 se puede observar cómo el ruido producido por el canal AWGN afecta los datos binarios.

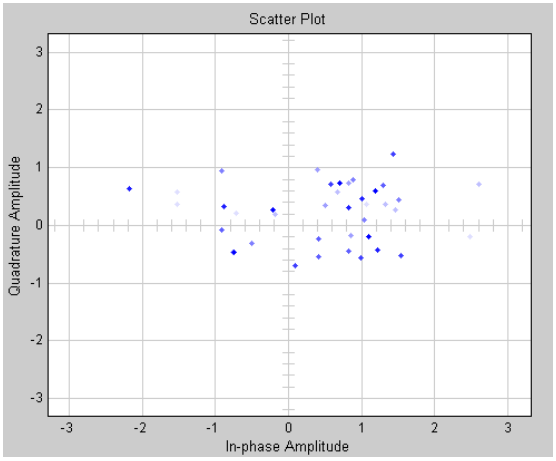

Figura 2. 4. Diagrama de constelación DBPSK después del canal AWGN.

Por lo anterior, para diseñar el demodulador, un aspecto fundamental es la detección, para lo cual se hace uso de regiones de decisión[1][5] que fueron consideradas a partir del conocimiento de los posibles valores de los componentes de fase y cuadratura de la señal. Entonces, lo que hace este detector es mirar netamente el componente de fase (I), es decir que el umbral de decisión es la recta definida como I = 0. La condición de diseño es que si el valor de I recibido es mayor que cero el símbolo detectado se ubica en la región 1, lo que quiere decir que si el símbolo cae en esta región es porque la diferencia de fase entre símbolos es 0°, de lo contrario se ubica en la región 2 porque los símbolos adyacentes tienen fase contraria, con lo que se hace una sectorización de regiones para saber cuál es el símbolo recibido, como se muestra en la figura 2.5.

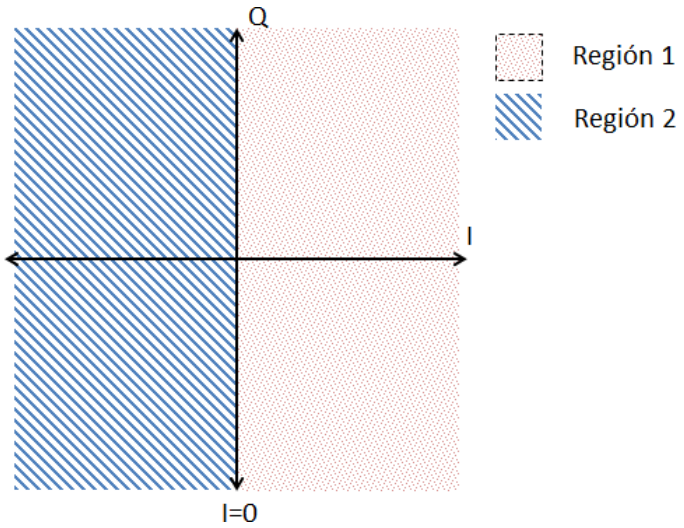

Figura 2. 5. Sectorización detección DBPSK.

[1] Este método está basado en el utilizado en la sección 6.3.2 de [5], se utiliza para ambas modulaciones.

Terminado el proceso de detección se procede a decodificar los datos obtenidos, para esto se tuvo en cuenta la tabla 1.3.

2.4.2. MODELO DQPSK

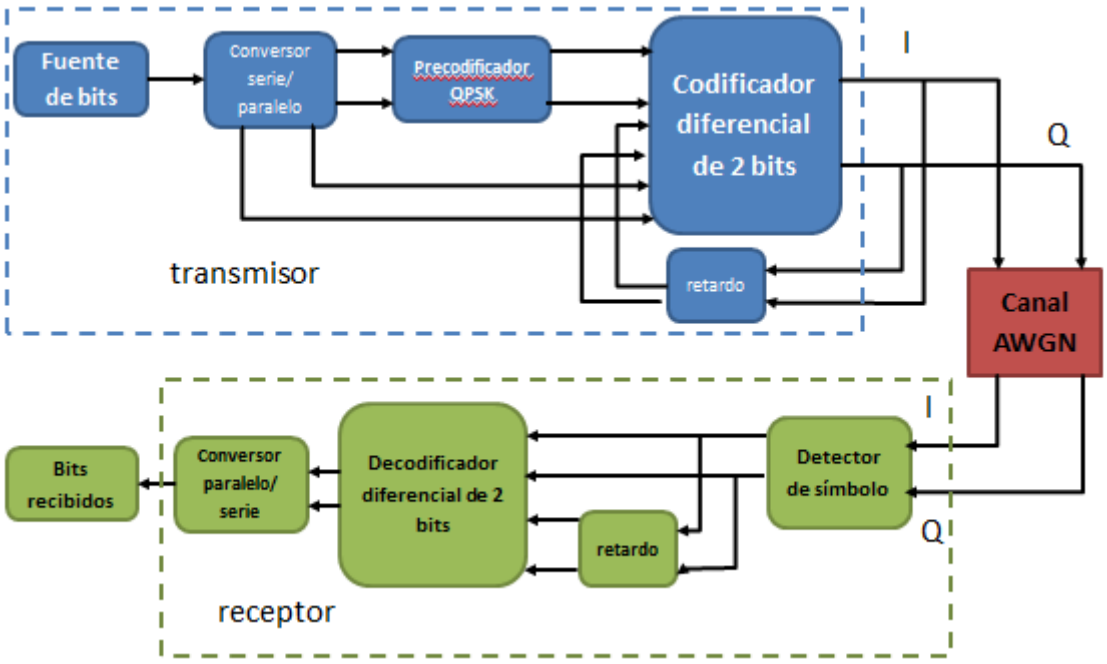

Figura 2. 6. Modelado DQPSK.

Subsistema 1.Transmisor

En este caso, manteniendo el modelo de un sistema de comunicaciones digital, se utiliza un generador de bits. La información proporcionada por este, inicialmente pasa por un conversor serie paralelo que se encarga de dividir los bits en pares e impares; posteriormente detecta un estado inicial con el precodificador QPSK que transmite símbolos con sus dos respectivos componentes en fase y en cuadratura; finalmente se tiene un codificador diferencial con un respectivo retardo para cada canal, la función de este es mapear el símbolo actual a partir del símbolo enviado anteriormente y el par de bits actuales que lleguen desde el divisor de datos. El funcionamiento del codificador esta descrito en la tabla 1.4, este permite asignar un símbolo dependiendo la diferencia de fases entre símbolos adyacentes, lo cual se puede corroborar con el diagrama de constelación obtenido, tal como se observa en la figura 2.7.

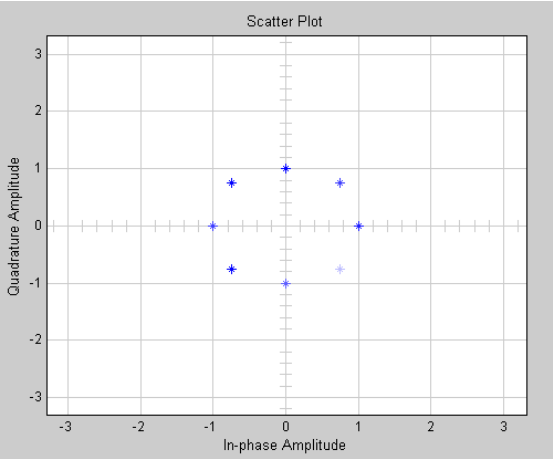

Figura 2. 7. Diagrama de constelación DQPSK en transmisión.

Subsistema 2. Canal AWGN

Este subsistema cumple con la función de ser el medio de transmisión en el cual transita la información, para este caso es un canal AWGN que posee ruido aditivo gaussiano; para esta modulación se adapta el modelado del canal como se hace en la sección 1.1.5., considerando un orden de modulación 4, o lo que es lo mismo modelar el canal con el parámetro M fijado en 4; por consiguiente, se obtienen los siguientes valores de desviación estándar (σ), consignados en la tabla 2.2.

Eb/No[dB]	SNR[dB]	Desviación estándar (σ)
0	3	0.5
1	4	0.445625469
2	5	0.397164117
3	6	0.353972892
4	7	0.315478672
5	8	0.281170662
6	9	0.250593616
7	10	0.223341796
8	11	0.199053585
9	12	0.177406694
10	13	0.158113883
11	14	0.140919146
12	15	0.125594321
13	16	0.111936057
14	17	0.099763116

Tabla 2.2. Valores de desviación estándar para cada uno de los valores de Eb/No usado en DQPSK.

Subsistema 3. Receptor

Al inicio se tiene un detector de símbolos que recibe la señal dividida en sus dos componentes, los cuales se ven muy afectados por el ruido que dificulta la función de detección de símbolos, esto se puede observar en la figura 2.8 donde se muestra el diagrama de constelación con los símbolos obtenidos después del canal AWGN.

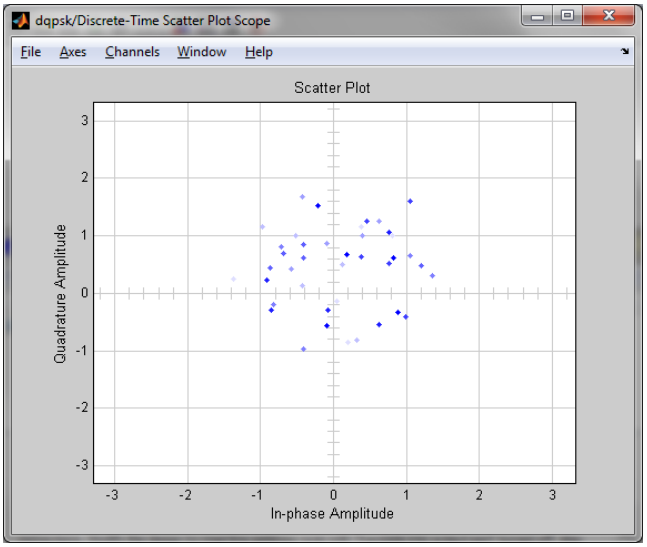

Figura 2. 8. Diagrama de constelación DQPSK después del canal.

Para solucionar el problema de detección se plantean dos métodos de detección, descritos a continuación.

- Método 1: sectorización octal

Por el diagrama de constelación obtenido se sabe que los posibles símbolos tienen una distancia equiprobable de 8 estados, muy parecida a la de la modulación 8PSK; sin embargo en la modulación DQPSK cada símbolo guarda la diferencia de fase del símbolo actual con respecto a uno anterior, por lo que se dice que cada símbolo va a tener una tolerancia de ± ¼ de cuadrante, es decir, la fase puede variar ±22.5° y se encapsula en una de las posibles regiones descritas. Su desventaja radica en que es muy perceptible al ruido lo que ocasiona el aumento en los errores en detección.

Para este método se utiliza la siguiente lógica de detección:

Región 1: Entre Q=I/2 y Q=-I/2 para I > 0.
Región 2: Entre Q=I/2 y Q= -2I para I > 0 y Q > 0.
Región 3: Entre Q=2I y Q= -2I para Q > 0.
Región 4: Entre Q= -I/2 y Q=-2I para I < 0 y Q < 0.

Región 5: Entre Q= -I/2 y Q= I/2 para I < 0.
Región 6: Entre Q=I/2 y Q=-I/2 para I < 0 y Q < 0.
Región 7: Entre Q=2 y Q=-2 para Q > 0.
Región 8: Entre Q= -I/2 y Q=-2 para I > 0 y Q < 0.

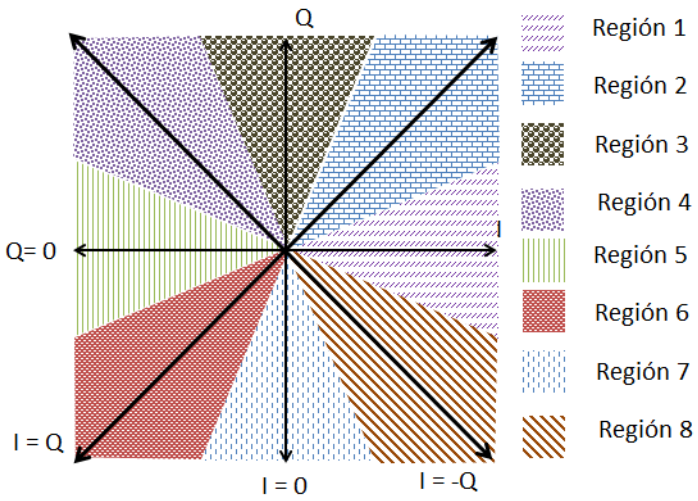

Figura 2. 9. Sectorización octal.

- Método 2: sectorización 4 con sincronización

Para el segundo método se plantea un tipo de detección más complejo que el método octal, pues debe tener sincronización con un reloj. Este método es posible si se tiene en cuenta que en tiempos impares los símbolos van a estar situados entre los ejes y en tiempos pares los símbolos van a estar situados exactamente en los ejes; por tanto la detección va a depender del tiempo, razón por la cual se debe tener sincronización en el sistema.

La constelación del sistema DQPSK se puede ver como la de dos sistemas QPSK rotados entre si 45°, lo que explica la generación de los ocho posibles símbolos que se visualizan en el diagrama de constelación de la figura 2.7. Lo anterior significa que para un tiempo t_{impar}=1,3,5,… la constelación es como la que se forma en QPSK y para un tiempo t_{par}=2,4,6,8,… la constelación es como la que se forma en QPSK rotada 45°.

La figura 2.10 muestra la detección en tiempos impares entonces estableciendo los umbrales para cada detección de símbolo.

Se tienen las siguientes consideraciones:

Región 1: Para Q > 0 y I > 0.
Región 2: Para Q < 0 y I > 0.

Región 3: Para Q < 0 y I < 0.
Región 4: Para Q > 0 y I < 0.

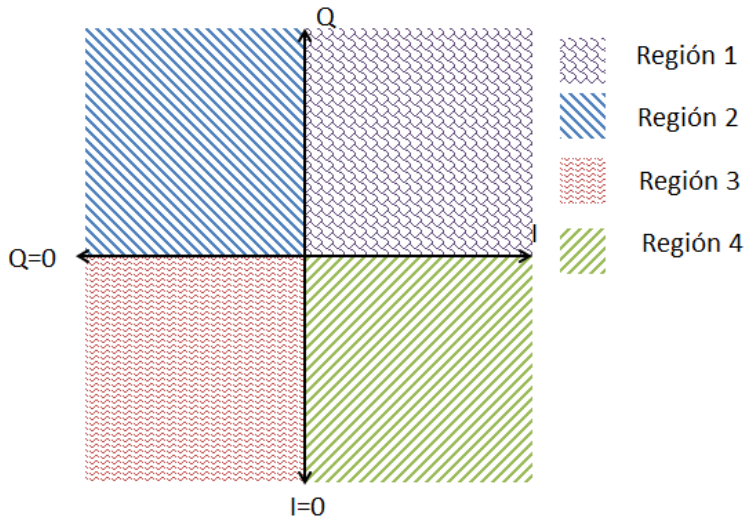

Figura 2.10. Detección tiempos pares.

La figura 2.11 muestra la detección en tiempos pares, estableciendo que los umbrales entre un símbolo y otro están definido por las rectas I = Q e I = -Q.

Se tienen las siguientes consideraciones:

Región 1: Entre Q=I y Q= -I para I > 0.
Región 2: Entre Q=I y Q= -I para Q > 0.
Región 3: Entre Q=I y Q= -I para I < 0.
Región 4: Entre Q=I y Q= -I para Q < 0.

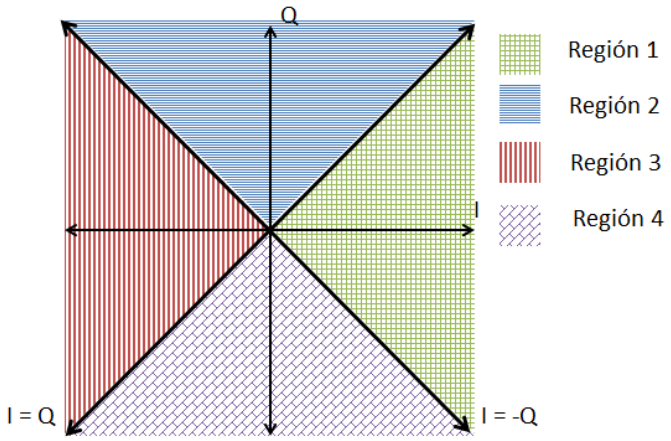

Figura 2.11. Detección tiempos impares.

Teniendo en cuenta los dos métodos de detección y que el sistema está basado en detección coherente, donde el transmisor y el receptor están sincronizados, se elige el segundo método para el diseño. Este último permite una detección eficiente y procesa de manera especializada los bits detectados, debido a la clasificación previa que hace de estos entre pares e impares. Además, la utilización del segundo método implica trabajar con regiones de detección mucho más grandes que el método de sectorización octal, donde existe mayor probabilidad de detectar errores.

Después del proceso de detección se hace uso de un retardo para cada canal I y Q debido a que se utiliza un decodificador diferencial de 2 bits, en el cual se obtienen los bits por dos ramas, indicando que por una van los bits pares y por la otra los bits impares. Esta operación se ejecuta con respecto a las fases del símbolo anterior y del símbolo actual, para ello se toma como referencia el comportamiento descrito en la tabla 1.5. Finalmente se realiza la conversión de paralelo a serie y se recupera el tren de bits enviado.

2.4.3. MODELO CALCULADOR DE BER

Para analizar el desempeño de los sistemas diseñados se realizó un subsistema que calcula la cantidad de bits recibidos erróneos con respecto a la cantidad de bits transmitidos, para esto se diseñó un modelo que obtiene el valor de BER ("*Bit Error Rate*" o Tasa de error de Bit); este modelo consta de bloques que realizan el cálculo mediante una programación realizada en *Matlab*®.

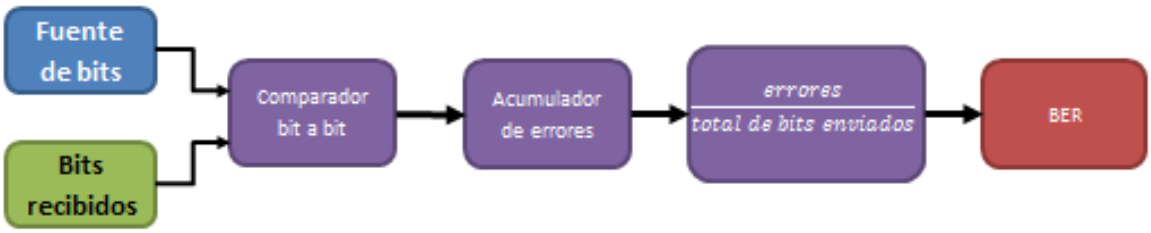

Figura 2. 12. Esquema calculador de BER.

Este subsistema tiene dos entradas: los bits que se generaron antes del modulador y los recibidos después de cada demodulador. Después estos son procesados en un bloque que compara bit a bit; cuando estos no son iguales se produce un error haciendo activar el acumulador de errores. Al finalizar el envío del tren de bits se hace la división entre los bits erróneos y el total de bits enviados, para posteriormente mostrar el resultado a partir de cualquier método de visualización.

2.5. SIMULACIÓN DE LOS SISTEMAS DISEÑADOS

2.5.1. MODELO DBPSK

Como se mencionó en la sección 1.1, en un sistema de modulación digital se necesita un generador de datos binarios, que para este caso se encuentra dentro del bloque modulador DBPSK; cabe mencionar que dicha modulación se realizará en banda base. Posteriormente se tiene un canal AWGN que diferencia las señales de fase y cuadratura. Luego, se presenta el demodulador DBPSK que se encarga de recuperar la información original a partir del componente en fase (I). Por último, se encuentra el "BER calculator" que realiza el cálculo de BER a partir de los bits erróneos y transmitidos, lo cual es visible en la figura 2.13.

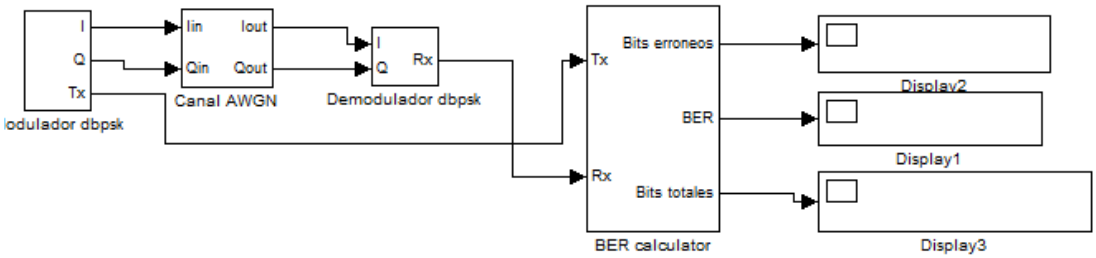

Figura 2. 13. Esquema de un sistema de comunicaciones con modulación digital DBPSK.

2.5.1.1. TRANSMISOR DBPSK

Inicialmente el esquema de un modulador DBPSK empieza con un generador de bits pseudoaleatorio, conformado por tres LSFR[1] y un multiplexor, para de esta manera garantizar que el generador sea equiprobable. Después del generador se tiene el modulador DBPSK que está conformado por un modulador BPSK y una compuerta XOR ubicada al inicio de este, que compara el estado inicial con el estado inmediatamente anterior, esto se puede ver en la figura 2.14.

[1] Es un registro de desplazamiento en el cual la entrada es un bit proveniente de aplicar una función de transformación lineal a un estado anterior.

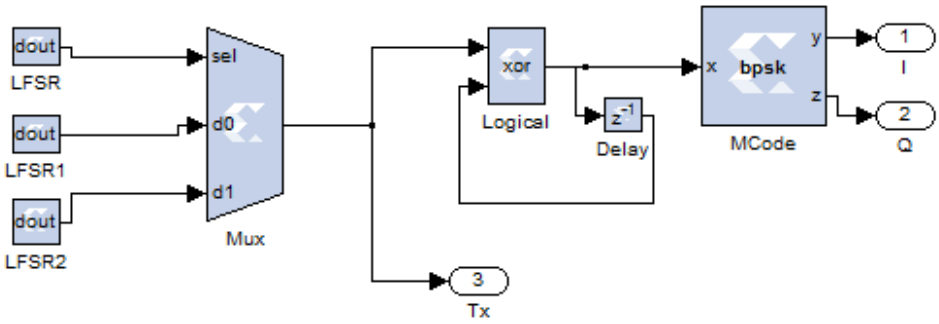

Figura 2. 14. Modulador y generador DBPSK.

En la figura 2.15 se realiza la visualización de la señal en diferentes puntos estratégicos del transmisor DBPSK para hacer más entendible su funcionamiento. El primer punto de visualización hace referencia a la señal transmitida por el generador de bits pseudoaleatorio, continuando, de acuerdo al diseño teórico otro punto a visualizar es el punto después del retardo, el cual sirve para comparar el estado anterior emitido por el generador y el estado actual que se encuentra después de la compuerta lógica XOR. Al final se tienen las dos componentes de la señal I y Q moduladas antes del canal; cabe mencionar que para este tipo de modulación la componente Q no tiene relevancia en el diseño, puesto que la información solo se ve reflejada en dos fases; por ende solo se hace uso de la componente real I con sus respectivos valores positivos y negativos.

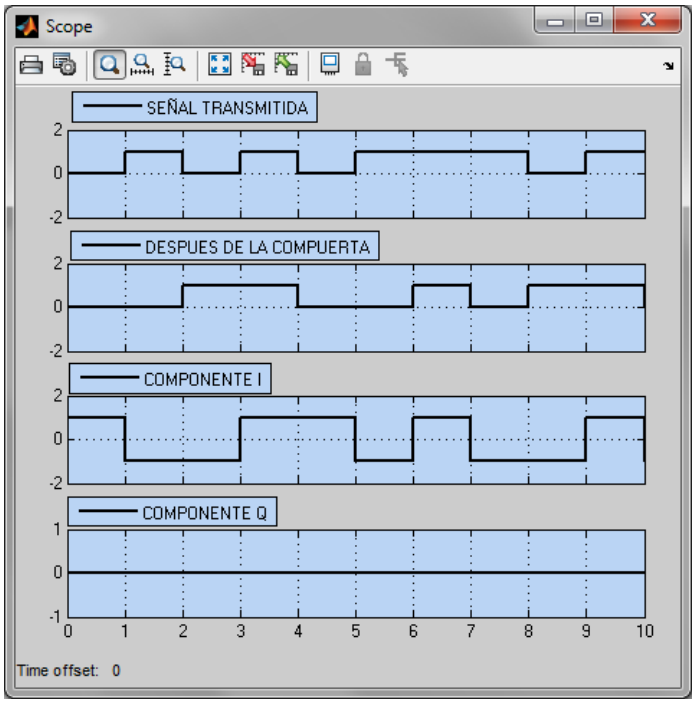

Figura 2. 15. Visualización de señales en el transmisor DBPSK.

Con el fin de verificar el correcto funcionamiento del procesamiento binario del transmisor DBPSK diseñado, se ha utilizado un "*scope*" que permite comparar las señales moduladas tanto en I como en Q en el *System Generator* de *Xilinx* con la señales moduladas en I y en Q del transmisor DBPSK tomado de *Simulink,* obteniendo el mismo comportamiento, como se puede corroborar en la siguiente figura.

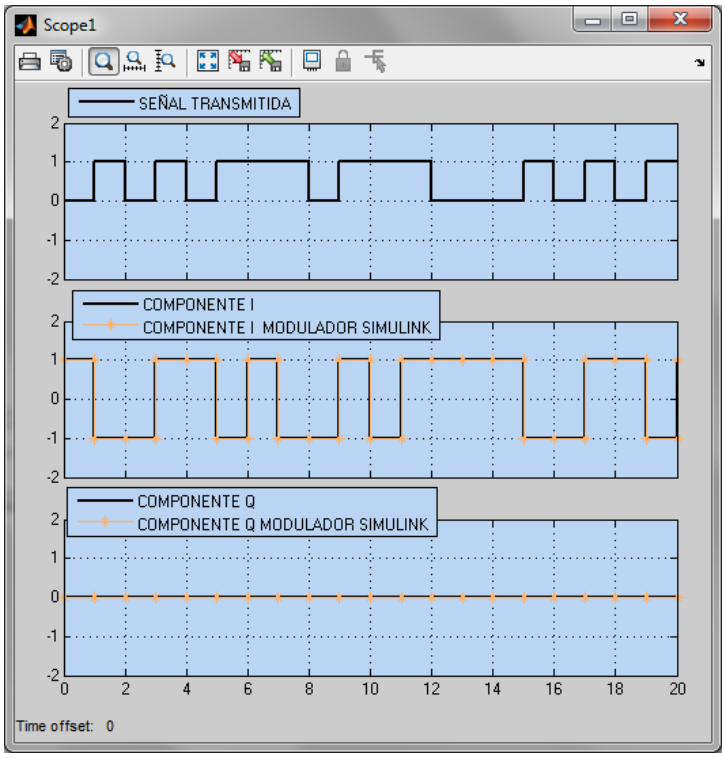

Figura 2. 16. Verificación del diseño del transmisor DBPSK.

2.5.1.2. RECEPTOR DBPSK

En la figura 2.17 se observa el demodulador que está conformado por un detector inicial[1] que recibe la señal del canal AWGN; dicho detector se encarga de convertir la señal compleja a bits; luego la señal pasa por un retardo para permitir la realización de la operación XOR entre el dato actual y el anterior, para de esta manera recuperar la señal original.

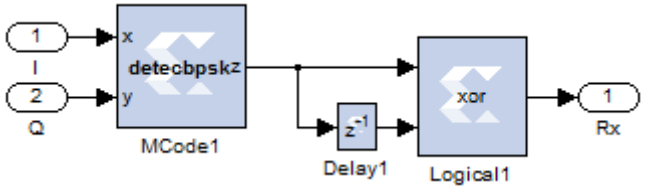

Figura 2. 17. Demodulador DBPSK.

[1] Este tiene el mismo funcionamiento que un demodulador BPSK.

En la figura 2.18 se realiza la visualización de la señal en diferentes puntos estratégicos del receptor DBPSK para hacer más entendible el proceso de recuperación de la señal transmitida. El primer punto de medición está situado después del canal AWGN, donde se muestra que la señal es afectada en sus respectivos componente I y Q, posteriormente se mide la señal después del "detecbpsk"[1] que deja observar la señal codificada diferencialmente y la señal codificada y retardada un tiempo de bit, a partir de las cuales se aplica la función lógica XOR obteniendo como resultado la señal decodificada y recuperada de manera correcta.

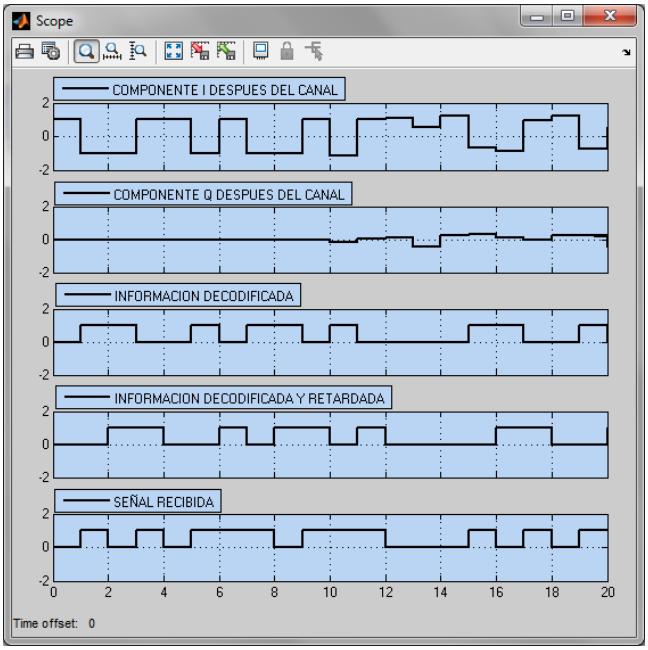

Figura 2. 18. Visualización de señales en el receptor DBPSK.

Con el fin de verificar el procesamiento binario en recepción, se ha utilizado un "*scope*" que permite comparar la señal binaria transmitida originalmente con la señal binaria que fue recibida después del sistema DBPSK. En la siguiente figura se muestra el funcionamiento del sistema, aunque cabe aclarar que para este ejemplo se ha tomado un valor de Eb/No de 10 dB que es bastante alto y por ende garantiza la correcta funcionalidad de este.

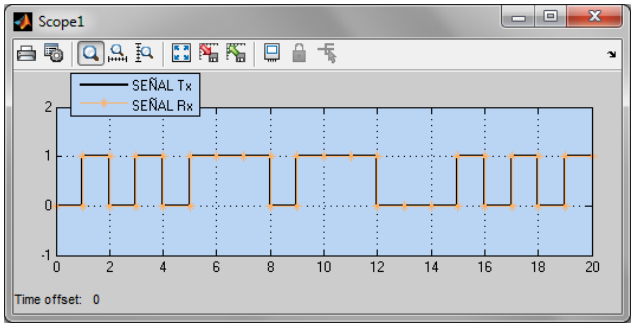

Figura 2. 19. Verificación del diseño del receptor DBPSK.

[1] Cabe aclarar que la mayoría de los errores presentados en el sistema se deben a la detección errónea producto de la afectación de la señal por el canal AWGN.

2.5.2. SISTEMA DQPSK

Esta modulación al igual que DBPSK se realiza en banda base. En primera instancia se cuenta con un bloque modulador DQPSK dentro del cual se encuentra un generador de bits; posteriormente se tiene el canal AWGN que recibe la señal diferenciada en fase y en cuadratura. Después del canal se tiene el demodulador DQPSK que recibe la señal en I y en Q, el cual se encarga de detectar y recuperar la señal original. Por último se encuentra el "BER calculator" que realiza el cálculo de BER a partir de los bits erróneos y transmitidos; este esquema se puede observar en la figura 2.20.

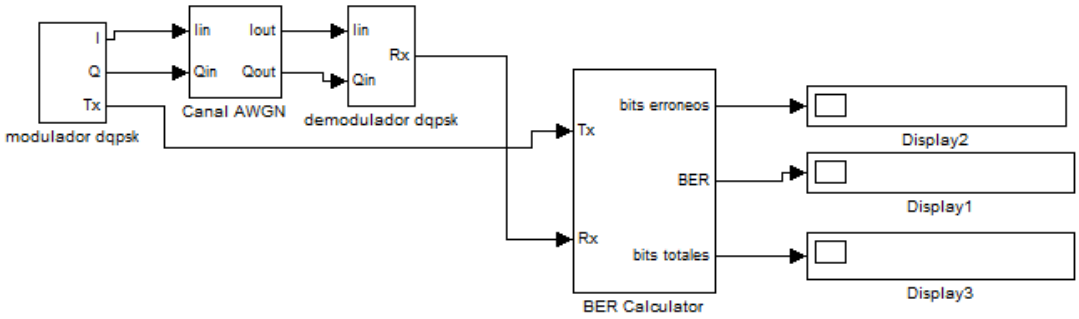

Figura 2. 20. Esquema de un sistema de comunicaciones con modulación digital DQPSK.

2.5.2.1. TRANSMISOR DQPSK

Al igual que el anterior caso de modulación se tiene el generador de bits pseudoaleatorio conformado por tres LFSR y un multiplexor. Posteriormente se realiza la división de los bits en pares e impares, por lo cual la tasa de trasmisión se divide a la mitad. Luego la señal pasa por un precodificador que establece los posibles estados iniciales en I y en Q. El siguiente bloque se denominó "mapp"; este cumple la función de codificar la señal diferencialmente; para esto, se guarda el primer estado y dependiendo de este se obtienen los siguientes estados; a partir de aquí la información está lista para ser enviada por el canal AWGN. Cabe mencionar que cada vez que llegue un nuevo par de bits al bloque "mapp" este hace un cambio de fase de por lo menos 45°; lo anterior se puede observar en la figura 2.21.

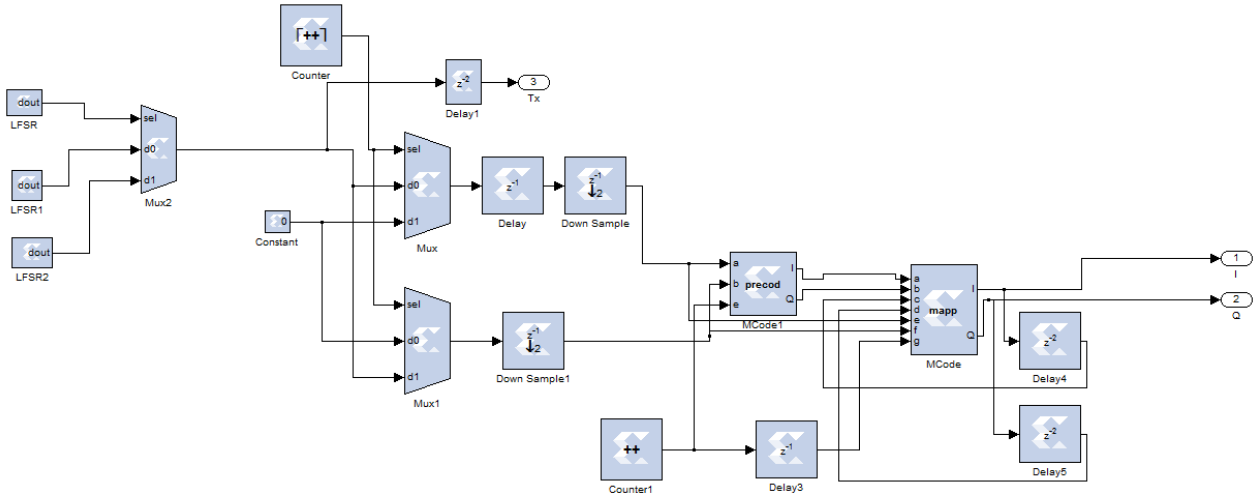

Figura 2. 21. Modulador y generador DQPSK.

En la figura 2.22 se realiza la visualización de la señal en diferentes puntos estratégicos del transmisor DQPSK. El primer punto de visualización está ubicado a la salida del generador de bits pseudoaleatorio; otros puntos a visualizar son los puntos ubicados después del conversor de serie a paralelo[1], los cuales sirven para referenciar el símbolo que se envía. Por medio de los otros componentes se guarda un estado anterior, y dependiendo de este y del estado actual se mapean los símbolos. Al final se tienen las dos componentes de la señal I y Q moduladas antes del canal.

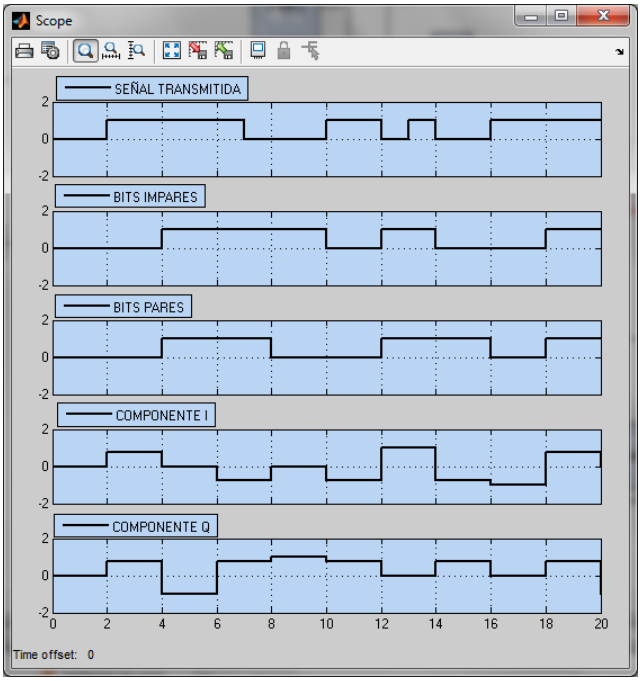

Figura 2. 22. Visualización de señales en el transmisor DQPSK.

[1] O del divisor de datos en dos ramas, disminuye la velocidad de transmisión a la mitad y que además produce un fenómeno intrínseco que es la inserción de un retardo de dos tiempos de bit.

Con el fin de verificar el procesamiento binario del transmisor DQPSK diseñado, se ha utilizado un "*scope*" que permite comparar las señales moduladas tanto en I como en Q en *System Generator* de *Xilinx* con la señales moduladas en I y en Q del transmisor DQPSK tomado de *Simulink,* obteniendo como resultado el mismo comportamiento de los dos transmisores, como se puede observar en la siguiente figura.

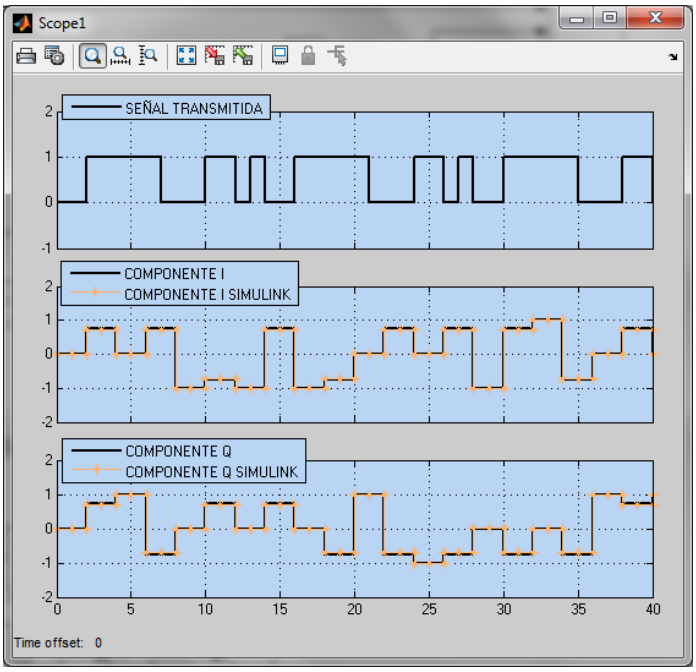

Figura 2. 23. Verificación del diseño del transmisor DQPSK..

2.5.2.2. RECEPTOR DQPSK

En la figura 2.24 se muestra el demodulador DQPSK que tiene como primer paso detectar la posición del símbolo en I y en Q; para esta función está el bloque "*detec2*" con la ayuda de un contador, ya que con este es capaz de diferenciar los dos posibles momentos: en el primero, el símbolo va a estar más cerca a los ejes y en el otro va a estar entre ellos[1]. A continuación, la información pasa por el "*dqpskdem*" que es el encargado de demodular las señales a bits pares e impares; esta operación se hace a partir de los componentes I antes, Q antes, I ahora y Q ahora; se puede decir que la modulación DQPSK depende de la diferencia de fase entre cada símbolo. Por último, se encuentra un multiplexor que es el encargado de concatenar los bits pares e impares en un solo tren de bits, recuperando así la señal que fue enviada originalmente.

[1] De esta manera se acude al método de sectorización 4 con sincronización descrito anteriormente.

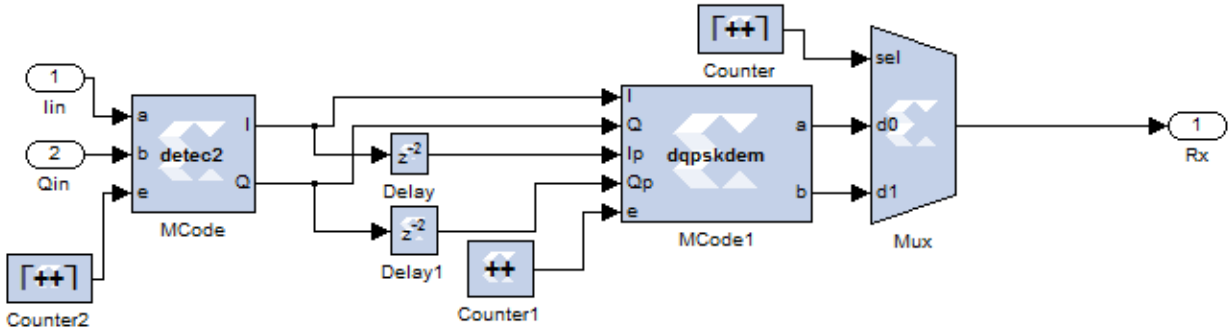

Figura 2. 24. Demodulador DQPSK.

En la figura 2.25 se realiza la visualización de la señal en diferentes puntos estratégicos del receptor DQPSK. El primer punto de medición está situado después del canal AWGN, donde se muestra que la señal es afectada en sus respectivos componente I y Q, posteriormente se mide la señal después del "dqpskdem" que deja observar la señal decodificada diferencialmente en paralelo, la cual está dividida en bits pares e impares, a partir de los cuales se aplica un multiplexor que será encargado de enlazar las dos señales y convertirlas en la señal decodificada y recibida de manera correcta, esta última es monitoreada para corroborar el diseño.

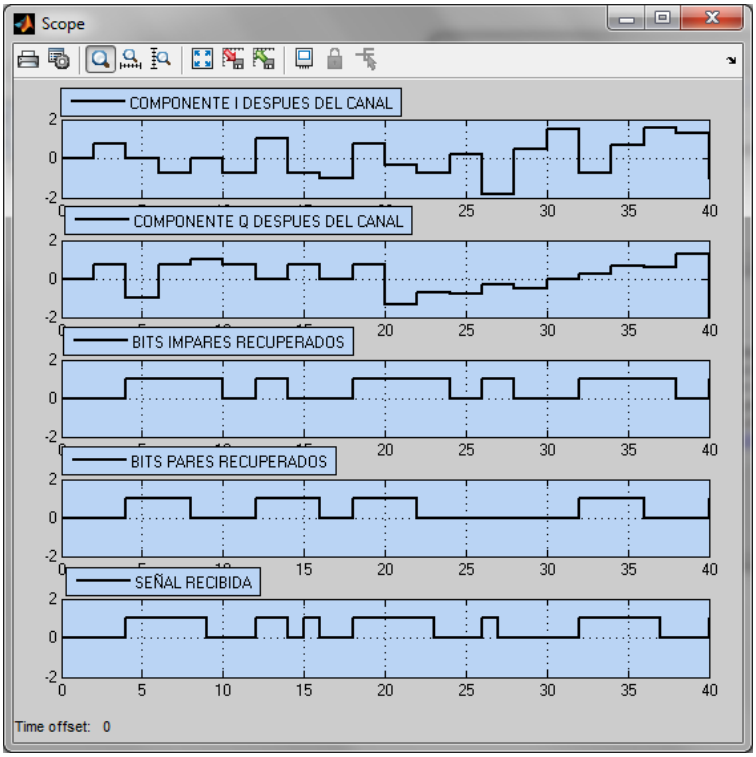

Figura 2. 25. Visualización de señales en el receptor DQPSK.

Con el fin de verificar el procesamiento binario en recepción, se ha utilizado un "*scope*" que permite comparar la señal binaria transmitida originalmente con la

señal binaria que fue recibida después del sistema DQPSK. En la siguiente figura se muestra el funcionamiento del sistema, aunque cabe aclarar que para este ejemplo se ha tomado un valor de Eb/No de 13 dB, bastante alto y que garantiza la correcta funcionalidad de este.

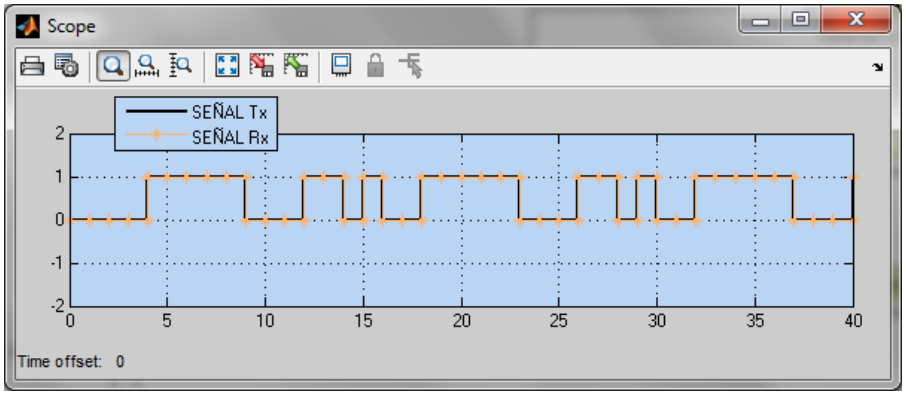

Figura 2. 26. Verificación del diseño del receptor DQPSK.

2.5.3. MODELO DEL CANAL AWGN

Una vez la señal ha sido modulada y normalizada debe atravesar el canal AWGN, sin embargo, en este punto debe resaltarse que *System Generator* no posee en su librería un bloque de canal AWGN, como si lo tiene *Simulink*; este problema se ve resuelto a partir de modelar el canal AWGN como dos canales AWGN no correlacionados; estos se construyen a partir de dos bloques generadores de ruido WGN que serán las fuentes de ruido normalizado N(0,1).

Además se requiere que los canales sean aditivos, por esto es necesario multiplicar el ruido generado por la desviación estándar encontrada anteriormente para cada condición de canal; de esta manera se realiza la suma de las potencias de ruido de cada uno para obtener la potencia total de ruido del sistema.

Otro aspecto que debe tenerse en cuenta es que el bloque generador WGN no tiene un parámetro específico que pueda ser modificado, como la relación energía de bit a densidad de ruido que posee el bloque del canal AWGN de *Simulink*, por lo que no es posible conocer la cantidad de ruido que se está introduciendo en el sistema.

Para modelar el canal AWGN se tuvieron en cuenta diferentes valores de desviación estándar (σ), los cuales fueron calculados a partir de las ecuaciones 1.9, 1.11 y 1.17 que dependen del valor de Eb/No y el orden de la modulación que sean asumidos (ver figura 2.27); este paso es muy importante porque a partir del canal AWGN se estudia el análisis del desempeño a través de curvas de BER.

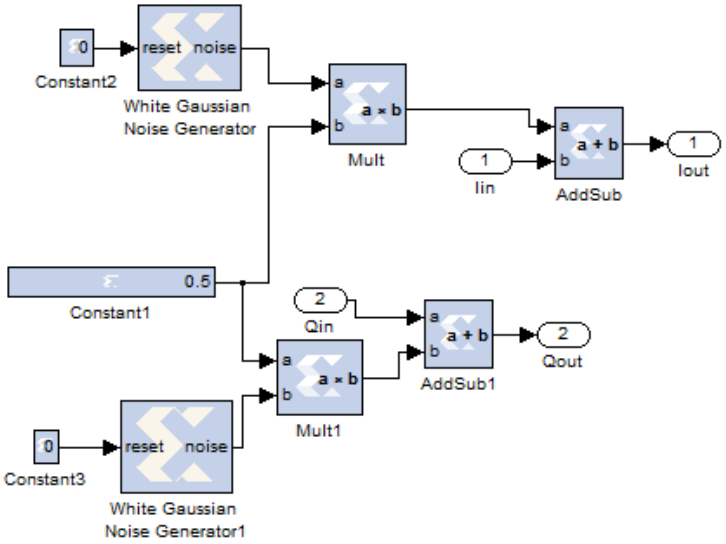

Figura 2. 27. Canal AWGN.

De acuerdo a las figuras 2.18 y 2.25 se puede deducir que el diseño del canal utilizado en los modelos de los sistemas afecta la señal después de 10 tiempos de símbolo, cabe aclarar que para la modulación DBPSK son equivalentes a 10 tiempos de bit y que en DQPSK son equivalentes a 20 tiempos de bit [40].

2.5.4. MÓDULO CALCULADOR DE BER

Para realizar el cálculo de la BER, bits erróneos y bits totales se tiene una comparación de los bits transmitidos y recibidos, para lo cual se utilizan las compuertas xor y xnor (ver figura 2.28). Al realizar la operación xor se detectan los bits erróneos que se acumulan a partir de un contador situado después de dicha compuerta. Mientras que la operación xnor permite detectar los bits correctos que se acumulan mediante su respectivo contador. Teniendo los bits erróneos y los correctamente recibidos se puede realizar la suma de estos para obtener los bits totales, permitiendo la división entre los bits erróneos y los totales, de esta manera se obtiene el valor de la BER.

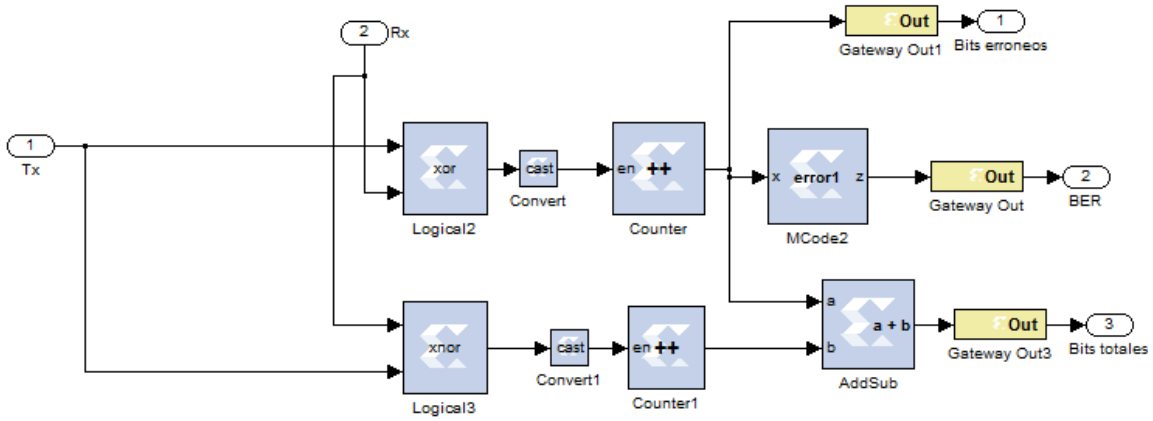

Figura 2. 28. Calculador de BER.

2.5.5. MÓDULO PARA EL DESPLIEGUE DE DATOS

Para ejecutar esta tarea se utiliza la pantalla LCD integrada que posee la FPGA *Spartan* 3A de *Xilinx*, la cual imprime los valores de BER para cada valor de Eb/No de manera automática y para cada tipo de modulación. El método de impresión de caracteres en la LCD se basa en la siguiente matriz que se observa en la figura 2.29 que permite obtener los elementos de visualización.

Figura 2. 29. Conjunto de caracteres LCD.

Para este despliegue se tiene el bloque de la figura 2.30 en *Xilinx* que permite la utilización de la pantalla LCD como elemento de despliegue de datos. En primera instancia se tiene un conversor binario decimal que se conecta a un registro; los valores almacenados en este registro son multiplexados, posteriormente mediante el bloque “LCD_Control” se reciben los datos y con el uso de la matriz que se observa en la figura 2.29, se elige los elementos a visualizar. Por último se tiene el bloque “Black Box_LCD” que es un bloque de implementación donde se mantiene el ciclo de la LCD en ejecución permanente.

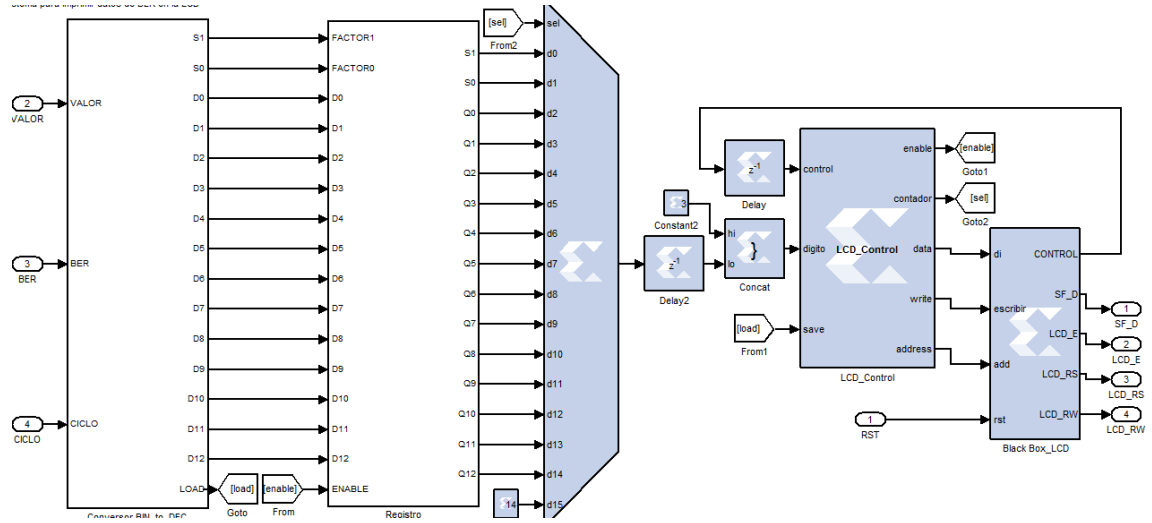

Figura 2. 30. LCD Xilinx despliegue de datos.

2.6. IMPLEMENTACION FISICA DE LOS SISTEMAS

Para realizar la implementación de todos los sistemas en la FPGA se necesita el archivo ejecutable con el que se programa; este proceso es lento y se crea al compilar los sistemas, únicamente accediendo al bloque *System Generator*®, y activando el botón *Generate* como se muestra en la figura 2.31; ahí empieza el proceso.

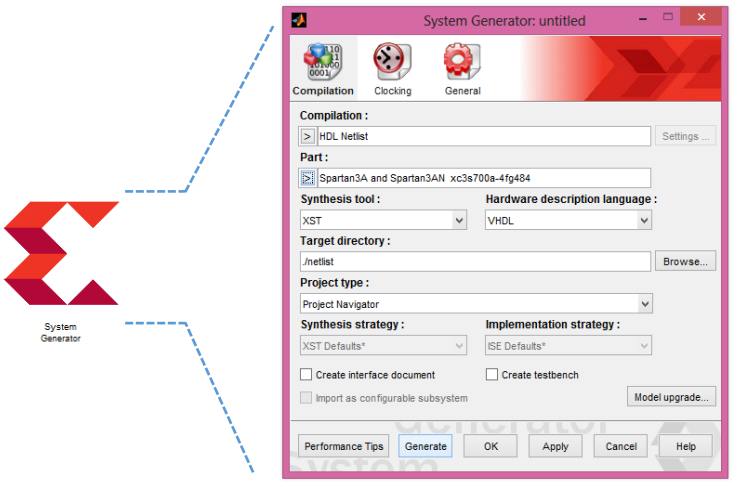

Figura 2. 31. Generación del archivo ejecutable

A continuación se abre una ventana emergente, en la cual aparece que se está procesando el sistema para poder convertirlo a un modelo en lenguaje HDL, en este momento aparece una ventana de inicialización de cada uno de los elementos, así como se muestra en la figura 2.32.

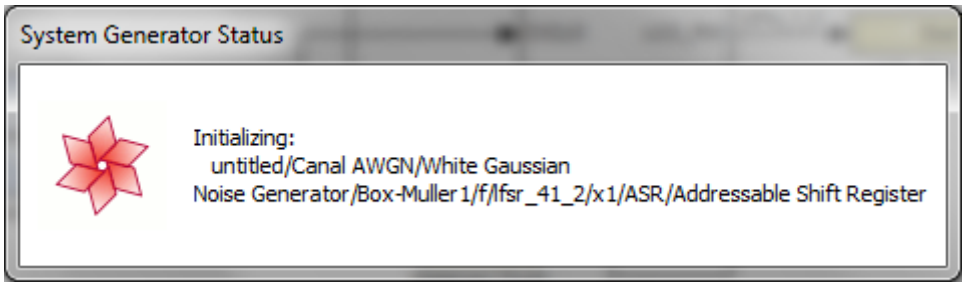

Figura 2. 32. Inicio de ejecución.

Si existen errores en el modelamiento del sistema, tales como: bloques de entrada desconectados, formatos inadecuados de las señales, uso de periodos de señales inapropiados, entre otros; aparece una ventana emergente que indica que la compilación del archivo no se puede completar, impidiendo el avance respecto al flujo de implementación de archivos; también aparece esta ventana si existen incoherencias que son tomadas como advertencias del sistema, entre ellas se tienen: uso de periodos muy pequeños en comparación al periodo de reloj de la FPGA, conexión de elementos de *Simulink* sin el uso de las respectivas *gateways* de salida, entre otros; cabe aclarar que estos sucesos no afectan el funcionamiento del sistema aunque siempre se deben evitar. Un indicador de que la generación del sistema se realizó correctamente es la activación de la ventana emergente de la figura 2.33.

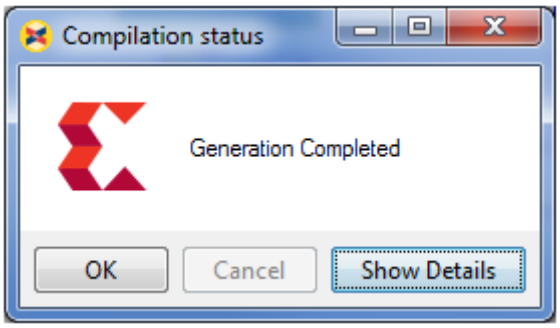

Figura 2. 33. Generación completada.

A partir de aquí se genera el archivo ".xise" que es leído como un proyecto en el *"Project Navigator"*; al abrirlo se puede observar que este se inicia correctamente como se muestra en la figura 2.34, para luego mostrar una interfaz amigable con el usuario en la cual el proceso a seguir es el básico, iniciando con la sintetización, continuando con la implementación del diseño y terminando con la generación del archivo ".bit"; estos pasos se encuentran al lado izquierdo inferior como se muestra en la figura 2.35.

Figura 2. 34. Inicio Project Navigator.

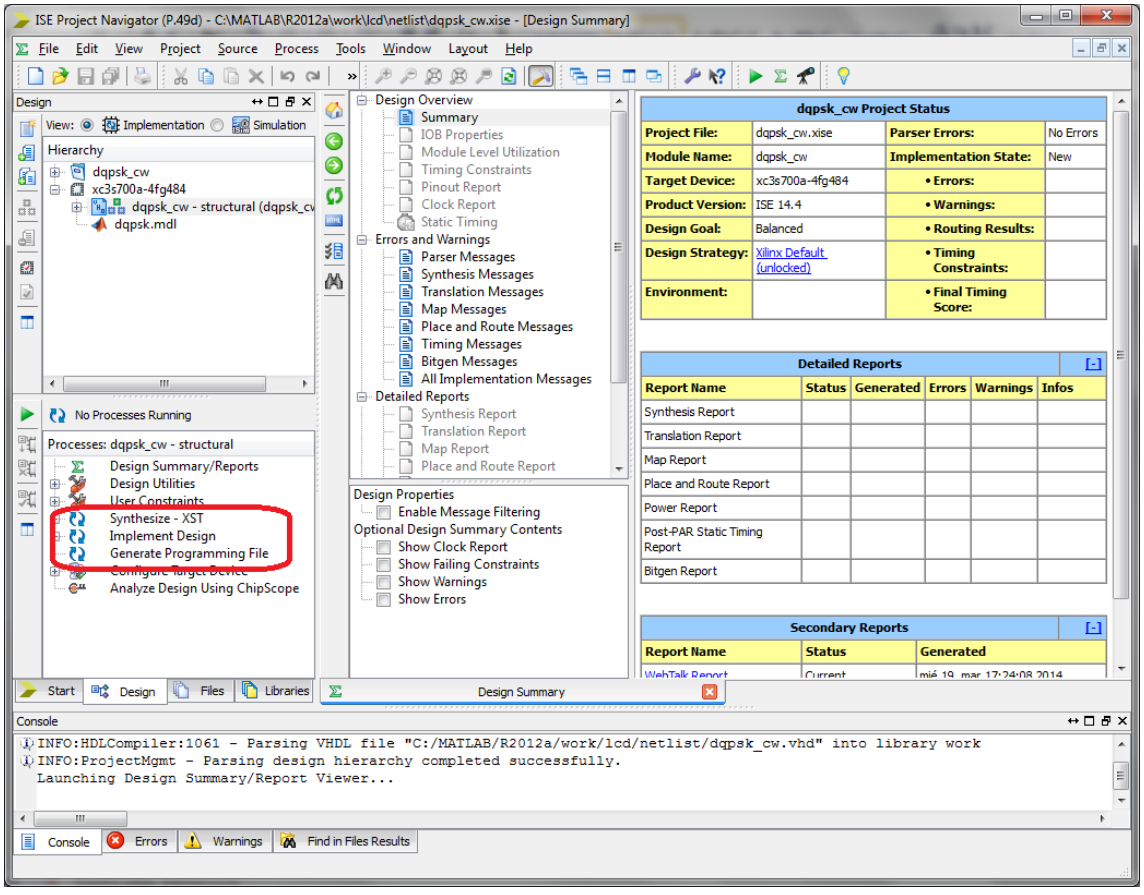

Figura 2. 35. Interfaz del Project Navigator.

Al finalizar la generación de los archivos ".bit" se puede observar unos parámetros muy importantes como son los *timing constraints* (limitaciones de tiempo), los cuales van a dictaminar el retardo máximo que el sistema estaría dispuesto a soportar; por lo tanto este parámetro será objeto de estudio en los dos siguientes ítems.

2.6.1. SISTEMA DBPSK

Para el sistema DBPSK se obtuvo solo una limitación de tiempo como se puede ver en la figura 2.36 que casi es imperceptible, por lo que este puede trabajar a la velocidad pico que es la misma frecuencia de oscilación del sistema que es de 50 MHz. Como se sabe que un bit es generado en un ciclo de reloj, se asume entonces que la tasa de transmisión máxima del sistema es de 50 Mbps; esta es la misma velocidad a la que puede trabajar el sistema DBPSK tomando en cuenta que se dejó la misma configuración de la simulación.

	Met	Constraint	Check	Worst Case Slack	Best Case Achievable	Timing Errors	Timing Score
1	No	TS_clk_dcc96e13 = PERIOD TIMEGRP "clk_dcc96e13" 10 ns HIGH 50%	SETUP HOLD	-9.044ns 0.331ns	19.044ns	442 0	1455265 0

Figura 2. 36. Timing Constraints DBPSK.

2.6.2. SISTEMA DQPSK

El sistema DQPSK fue muy afectado por las limitaciones de tiempo como se puede ver en la figura 2.37; este tipo de retardos hacen que el sistema pierda el sincronismo, o lo que es lo mismo, aparezca el fenómeno de ISI[1] que es algo muy indeseado en los sistemas de comunicaciones, por lo que este debe disminuir la velocidad de trabajo a través de la configuración del *System Generator*. Entonces, al disminuir la frecuencia de muestreo del sistema se obtuvo un resultado como se muestra en la figura 2.38.

	Met	Constraint	Check	Worst Case Slack	Best Case Achievable	Timing Errors	Timing Score
1	Yes	TS_clk_6539b8ef = PERIOD TIMEGRP "clk_6539b8ef" 10 ns HIGH 50%	SETUP HOLD	4.455ns 0.847ns	5.545ns	0 0	0 0
2	Yes	TS_ce_64_6539b8ef_group_to_ce_64_6539b8ef_group = MAXDELAY FROM TIMEGRP "ce_64_6539b8ef_group" TO TIMEGRP "ce_64_6539b8ef_group" 640 ns	SETUP HOLD	595.744ns 0.251ns	44.256ns	0 0	0 0
3	Yes	TS_ce_128_6539b8ef_group_to_ce_64_6539b8ef_group = MAXDELAY FROM TIMEGRP "ce_128_6539b8ef_group" TO TIMEGRP "ce_64_6539b8ef_group" 640 ns	SETUP HOLD	606.205ns 2.666ns	33.795ns	0 0	0 0
4	Yes	TS_ce_64_6539b8ef_group_to_ce_128_6539b8ef_group = MAXDELAY FROM TIMEGRP "ce_64_6539b8ef_group" TO TIMEGRP "ce_128_6539b8ef_group" 640 ns	SETUP HOLD	636.671ns 0.966ns	3.329ns	0 0	0 0
5	Yes	TS_ce_128_6539b8ef_group_to_ce_128_6539b8ef_group = MAXDELAY FROM TIMEGRP "ce_128_6539b8ef_group" TO TIMEGRP "ce_128_6539b8ef_grou...					

Figura 2. 37. Timing Constraints DQPSK.

	Met	Constraint	Check	Worst Case Slack	Best Case Achievable	Timing Errors	Timing Score
1	No	TS_clk_473a3f48 = PERIOD TIMEGRP "clk_473a3f48" 10 ns HIGH 50%	SETUP HOLD	-26.615ns 0.311ns	36.615ns	399 0	2571687 0
2	Yes	TS_ce_2_473a3f48_group_to_ce_2_473a3f48_group = MAXDELAY FROM TIMEGRP "ce_2_473a3f48_group" TO TIMEGRP "ce_2_473a3f48_gro...					

Figura 2. 38. Timing Constraints DQPSK con reducción de tasa.

[1] Interferencia inter-simbólica se refiere a que los pulsos de la señal se solapan entre sí, por lo que se generan errores en detección.

Este resultado se obtuvo al disminuir la frecuencia de muestreo 32 veces; de aquí en adelante se podrá obtener una respuesta óptima del sistema y valores reales de rendimiento.

2.7. VALIDACIÓN DE SIMULACIÓN DE LOS SISTEMAS

Para validar los modelos de los sistemas diseñados se recurrió al método de simulación a través de la realización de curvas de BER para cada sistema; luego se comparan estas curvas con resultados teóricos obtenidos a través de las ecuaciones teóricas propuestas para cada sistema, pero antes de esto cabe aclarar que los escenarios de simulación tuvieron la presencia de un canal AWGN lo que permitió el desarrollo de la validación.

Además de lo anterior se tuvo en cuenta para cada simulación que el número total de bits a enviar fueron 1048576; este número se escogió ya que para el cálculo de la BER era necesario hacer una división; la FPGA o hardware reconfigurable permite hacer este tipo de operaciones solo con un denominador (en este caso el número total de bits a enviar) que fuera un número potencia de 2; este número de bits asegura que se cumple con el objetivo de que los sistemas lleguen a una BER de 10^{-6}.

Para la obtención de datos de manera confiable se usó el método de promedio estadístico[1], el cual para este caso consiste en tomar 10 muestras para la representación de cada valor con respecto a la variación del parámetro de Eb/No, en vista de que cada simulación utilizaba mucho tiempo fue pertinente realizarlo solo diez veces y posteriormente procesar la información obtenida y promediarla.

De esta manera se calculó el número de errores de los datos recibidos con respecto a los transmitidos en función de la relación energía de bit a potencia de ruido generado (Eb/No). En los siguientes ítems se presentan tablas y graficas comparativas tanto para el caso de DBPSK como para DQPSK.

2.7.1. SISTEMA DBPSK

Para el sistema con modulación DBPSK se obtuvo que la curva de BER simulada mantuvo un comportamiento similar a la curva de BER teórica obtenida a través de la ecuación 1.22[2], haciendo la aclaración que a partir de aproximadamente 8 dB en el valor de Eb/No el sistema simulado parece tener mejor desempeño, pero esto se debe a la resolución de la BER, es decir, para obtener un dato más preciso se debió hacer una simulación con más bits; lo

[1] A través del cual se obtienen “n” valores, los cuales posteriormente se someten al proceso del promedio y así obtener un valor más cercano con respecto a un valor esperado.
[2] Esta muestra el comportamiento de un sistema DBPSK que utiliza demodulación coherente

anterior se puede corroborar a través de los datos consignados en la tabla 2.3 o también se puede deducir de la figura 2.39.

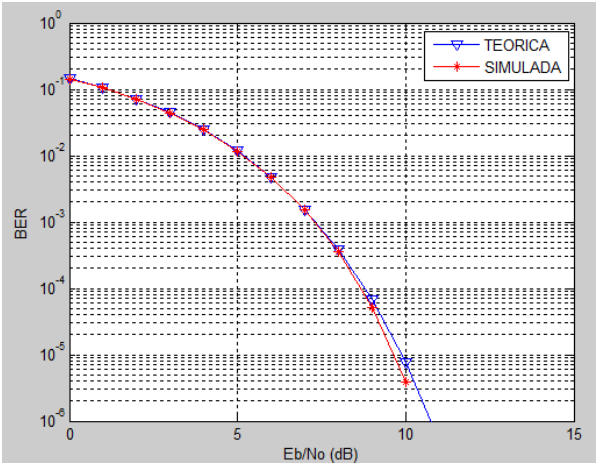

Figura 2. 39. Comparación de curvas de BER DBPSK.

Eb/No[dB]	BER TEORICA	BER SIMULADA
0	0.144927686780961	0.142904853821
1	0.106228587716503	0.105002784729
2	0.072198837388899	0.071999645233
3	0.044709972057108	0.044471359253
4	0.024689095178100	0.024606704712
5	0.011836837227531	0.011758041382
6	0.004765173696157	0.004653930664
7	0.001544155578016	0.001485252380
8	0.000381742656596	0.000387382507
9	0.000067252195258	0.000062751770
10	0.000007744186445	0.000007057190
11	0.000000522613454	0.000000190735
12	0.000000018012021	0
13	0.000000000266586	0
14	0.000000000001362	0
15	0.000000000000002	0

Tabla 2.3. Comparación de BER con modulación DBPSK.

2.7.2. SISTEMA DQPSK

Para el sistema con modulación DQPSK se obtuvo que la BER del sistema diseñado mantuvo un comportamiento similar a la curva de la BER teórica obtenida a través de la ecuación 1.28, haciendo la aclaración que a partir de aproximadamente 3 dB en el valor de Eb/No el sistema simulado tiende a mejorar levemente el desempeño que se propone a partir de la curva teórica; y

a partir de 10 dB se puede decir que el sistema DQPSK diseñado tiene un desempeño menor o igual al presentado en el sistema teórico, pero esto se debe al error en la precisión de la simulación. El análisis anterior se puede corroborar a través de los datos consignados en la tabla 2.4 o también se puede deducir de la figura 2.40.

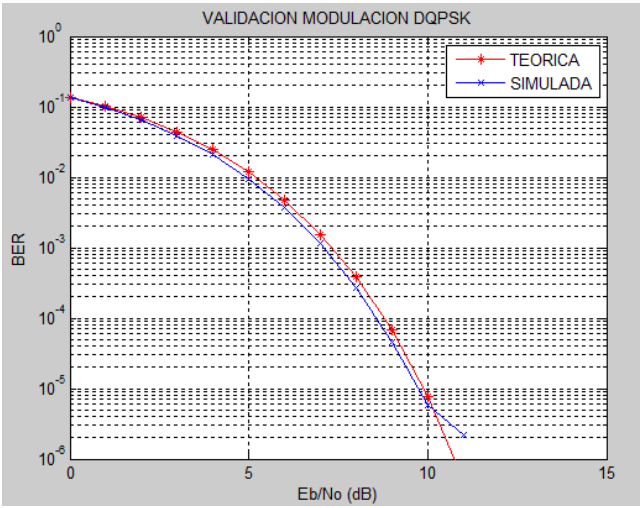

Figura 2. 40. Comparación de curvar de BER DQPSK.

Eb/No[dB]	BER TEÓRICA	BER SIMULADA
0	0.134425669583121	0.134666760763
1	0.100586331292382	0.096404817369
2	0.069592501328745	0.064151128133
3	0.043710481256435	0.038585768806
4	0.024384319467743	0.020715289646
5	0.011766781869755	0.009515232510
6	0.004753820255980	0.003690507677
7	0.001542963369792	0.001152568393
8	0.000381669792868	0.000274372101
9	0.000067249933829	0.000044918060
10	0.000007744156458	0.000005878723
11	0.000000522613318	0.000002193451
12	0.000000018012020	0
13	0.000000000266586	0
14	0.000000000001362	0

Tabla 2.4. Comparación de BER con modulación DQPSK.

De esta manera con las observaciones hechas de los sistemas simulados se puede decir que sus comportamientos son acordes a los teóricos; entonces, considerando los sistemas de comunicación completos, compuestos por un transmisor, un canal y un receptor, fueron validados mediante el método de

análisis a través de curvas de BER, y además descritos anteriormente con su respectiva representación, en las figuras 2.39 y 2.40; estos modelos permiten hacer la evaluación de desempeño de los sistemas de comunicación mediante la implementación en hardware reconfigurable.

CAPÍTULO 3. EXPERIMENTACIÓN Y ANALÍSIS DE RESULTADOS.

3.1. PLAN DE PRUEBAS

Para el análisis de desempeño de los sistemas de comunicación con modulaciones DBPSK y DQPSK se plantea una serie de pruebas que se realizan mediante la medición de la BER en función de la Eb/No; las primeras para verificar las simulaciones ya hechas en el capítulo 2, donde se obtuvieron los datos de simulación, los cuales fueron validados a través de la comparación de los mismos con resultados teóricos manteniendo las mismas condiciones de ruido.

A continuación se realizan pruebas en implementación en la FPGA; los datos obtenidos en implementación se comparan con los datos validados a través de la simulación. Siempre se buscó que el sistema total fuese capaz de visualizar tasas de errores hasta de 10^{-6}; para cumplir con este objetivo y restringiéndose al uso de divisiones de potencias de 2 en FPGA se usó un número total de bits enviados igual a 2^{20}; entonces se buscó mantener las mismas condiciones del sistema sin importar el proceso, por lo que se esperaba que los resultados obtenidos en implementación fueran acordes a los obtenidos a través de la simulación; además estas dos condiciones se tienen ya que no se pudo contar con un equipo de procesamiento bastante alto, esto para poder hacer una simulación más exacta de los sistemas .

Se calculan los valores de BER para obtener curvas comparativas entre teóricas, simuladas e implementadas para cada tipo de modulación y entre tipos de modulación, siempre bajo las mismas condiciones.

La experimentación se realizó con un número de diez corridas, puesto que para la simulación se utilizó el mismo número de iteraciones, garantizando de esta manera las mismas condiciones tanto en simulación como en implementación en cuanto al número de valores que son procesados a través del método de promedio estadístico.

Se establecieron unos aspectos a tener en cuenta en los resultados de implementación, estos fueron visualización de las formas de onda de los sistemas y obtención de datos de BER acordes a los simulados. Teniendo en cuenta que en la simulación de los sistemas no había ningún inconveniente a la hora de variar velocidad, entonces se hicieron estas variaciones en cuanto a la implementación, por lo que se tuvieron en cuenta 6 escenarios, estos se describirán a continuación.

ESCENARIO 1

TASA DE TRANSMISION DE DATOS	195,3125 Kbps
MODULACION	DBPSK

Tabla 3.1. Descripción escenario 1.

ESCENARIO 2

TASA DE TRANSMISION DE DATOS	195,3125 Kbps
MODULACION	DQPSK

Tabla 3.2. Descripción escenario 2.

ESCENARIO 3

TASA DE TRANSMISION DE DATOS	1,5625 Mbps
MODULACION	DBPSK

Tabla 3.3. Descripción escenario 3.

ESCENARIO 4

TASA DE TRANSMISION DE DATOS	1,5625 Mbps
MODULACION	DQPSK

Tabla 3.4. Descripción escenario 4.

ESCENARIO 5

TASA DE TRANSMISION DE DATOS	50 Mbps
MODULACION	DBPSK

Tabla 3.5. Descripción escenario 5.

ESCENARIO 6

TASA DE TRANSMISION DE DATOS	50 Mbps
MODULACION	DQPSK

Tabla 3.6. Descripción escenario 6.

3.2. ANALÍSIS DE LOS ESCENARIOS DE SIMULACIÓN.

ESCENARIOS DE IMPLEMENTACIÓN	VISUALIZACIÓN FORMAS DE ONDA		BER ACORDE A SIMULACIÓN	
	SI	NO	SI	NO
1	**X**		**X**	
2	**X**		**X**	
3		**X**	**X**	
4		**X**	**X**	
5		**X**	**X**	
6		**X**		**X**

Tabla 3.7. Análisis de escenarios.

A partir de la tabla 3.7 se pudo determinar que solo los escenarios 1 y 2 fueron aptos para realizar las pruebas correspondientes que van a ser descritas en el siguiente subcapítulo referentes a la obtención de las formas de onda transmitidas y recibidas. Además que el escenario 6 fue el único que no nos permitió obtener unos datos de desempeño acordes a los datos simulados, por ende decimos que el sistema de comunicaciones DQPSK es implementable solo hasta la velocidad de transmisión de 1,5625 Mbps, esto acorde a la restricción determinada en la sección 2.6.2. En cuanto al sistema de comunicaciones que utilizo la modulación DBPSK solo se restringió por la frecuencia máxima de trabajo de la FPGA con lo que solo se puede trabajar hasta una tasa de transmisión de 50 Mbps.

3.3. CONFIGURACIÓN DE LOS ELEMENTOS DEL SISTEMA

Para cumplir con lo propuesto en el plan de pruebas se deben configurar los parámetros del sistema de comunicaciones como son: tasa de transmisión, bits enviados, bits transmitidos y además el periodo de bit y de símbolo.

Posteriormente se realiza la medición del tiempo de ocupación de recursos por bit transmitido, sabiendo que se van a transmitir 2^{20} bits además de contar con una frecuencia de trabajo de 50 Mhz propia de la FPGA que es la misma frecuencia de muestreo. El tiempo de ocupación de recursos se halla teniendo en cuenta que se toma una muestra en cada ciclo de reloj asignándole a cada un bit, entonces para el momento de despliegue se tiene inicialmente que:

$$f_t = 50\ Mhz \rightarrow R = 50\ Mbps \tag{3.1}$$

donde f_t: es la frecuencia de trabajo de la FPGA

R: es la tasa de transmisión de datos de la FPGA

Para estas condiciones se aplica la ecuación inversa para saber la duración de un bit.

$$T_b = 1/\mathrm{R} = 1/(50\,\mathrm{x}10^6) = 20\ \mathrm{ns} \quad (3.2)$$

donde T_b: es el periodo de bit.

En el proceso de implementación se tiene una tasa de bit de entrada de 50 Mbps que va a ser manipulada de acuerdo al diseño de las pruebas. Para la visualización de las señales de entrada (información mensaje) y salida (información recibida después del receptor) fue necesario disminuir la tasa de transmisión a valores de frecuencia soportadas por el equipo visualizador; en este caso se utilizó como equipo visualizador un osciloscopio digital que soporta frecuencias de muestreo de hasta 50 Mhz; pero teniendo en cuenta que esta frecuencia debe ser mucho mayor a la de la señal medida, la tasa de datos fue disminuida 256 veces obteniendo así unas formas de onda acordes a unas señales digitales aproximadamente cuadradas, como se observa en la figura 3.1 y a partir de las consideraciones de los escenarios 1 y 2.

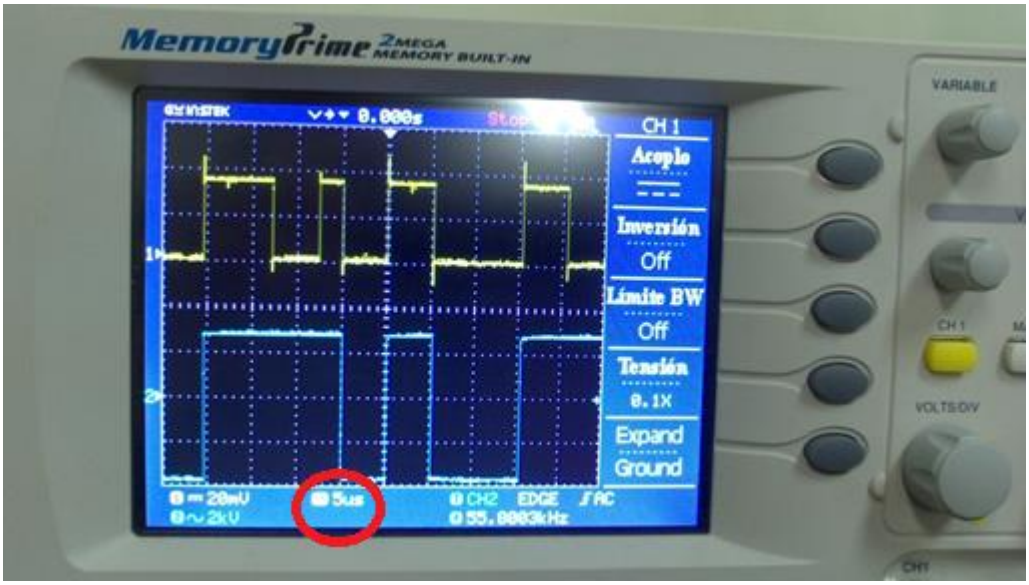

Figura 3.1. Visualizacion de señales enviadas y recibidas.

Se establece entonces un factor de tasa de transmisión que define cuantas veces se reducirá la tasa de transmisión y también cuantas veces aumentara el tiempo de bit, así:

$$R_k = \frac{R}{k} \quad (3.3)$$

$$Tb_k = k\ x\ Tb \quad (3.4)$$

donde R_k: es la tasa de transmisión de bit modificada.

k: es el factor de tasa de transmisión.

Tb_k: es el periodo de bit modificado.

Entonces la tasa de datos y el periodo de bit para el caso de la visualización en el osciloscopio digital son:

$$R_{256} = \frac{R}{256} = \frac{50\ Mbps}{256} = 195{,}3125\ Kbps \tag{3.5}$$

$$Tb_{256} = 256\ x\ Tb = 256\ x\ 20\ ns = 5{,}12\ us \tag{3.6}$$

En cambio para la implementación lo que importó fue la velocidad de procesamiento de toda la información, o sea el tiempo en el cual se procesan los 2^{20} bits; para hallar este tiempo se aplica la siguiente ecuación:

$$Tp = \frac{k\ x\ L}{R} \tag{3.7}$$

donde Tp : es el tiempo de procesamiento.

L: es la longitud del mensaje y está dado en unidades de bit.

Este tiempo es un parámetro muy importante que se tomó de referencia para el muestreo de los datos de BER, sabiendo que los sistemas diseñados arrojan un dato de BER cada vez que se procesan todos los bits; después del tiempo de procesamiento se reinicia todo el sistema para que cuando se cumpla nuevamente dicho tiempo se envíe otra muestra que será visualizada a través de la LCD de la FPGA.

Para que exista un buen espaciamiento de tiempo entre muestras se escogió un tiempo de procesamiento aproximado de 10 segundos; para llegar a este resultado se puede variar solo el factor de tasa. Se procede a calcular el factor de tasa definido con la ecuación 3.8, así:

$$k = \frac{Tp\ x\ R}{L} = \frac{10\ s\ x\ 50\ Mbps}{1.048\ Mb} = \frac{500}{1.048} = 476.84 \tag{3.8}$$

Pero como se quiere trabajar todo en términos de potencias de 2, se busca la potencia de 2 más cercana a ese valor, dando como resultado lo siguiente:

$$k \approx 512 \tag{3.9}$$

A continuación se muestra la ventana de configuración del factor de tasa para asegurar la tasa máxima de transmisión que es de 50 Mbps, o lo que es lo mismo se deja el periodo de reloj de la FPGA en 20 ns y por defecto para la *Spartan* 3A se ubica el reloj en el pin E12; se muestra que el sistema está configurado para que la señal sea visualizada en el osciloscopio, pero es de notar que se debe introducir el reciproco del factor de tasa como se muestra en la figura 3.2.

Figura 3.2. Configuración del periodo de bit.

Además, para garantizar que sean tomados los 2^{20} bits es necesario configurar el bloque “control”, el cual controla cada cuantos bits será enviada la señal para que sea leída la BER, ya que al siguiente estado se envía la orden de reiniciar todos los elementos que conforman el sistema, para poder cambiar automáticamente a otro valor de EB/No a través de su desviación estándar. Lo anterior se puede observar en la figura 3.3.

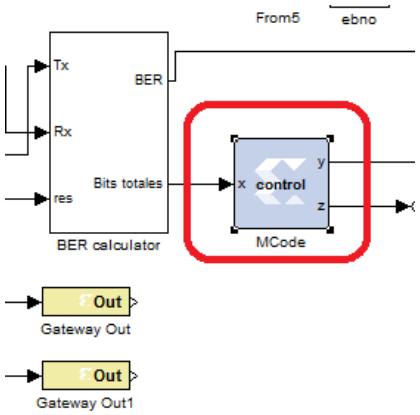

Figura 3.3. Visualización del bloque control.

A continuación se muestra la configuración del bloque “control”:

```
function [y,z] = control(x)
  if x == 1048576    y = 1;
    z = xfix({xlBoolean},0);
  elseif x == 1048577
     y = 0;
     z = xfix({xlBoolean},1);
  else
    y = 0;
    z = xfix({xlBoolean},0);
  end
```

3.4. DETERMINACIÓN DE LAS CAPACIDADES MÍNIMAS DE LA FPGA

Para determinar las capacidades mínimas de la FPGA y poder suplir los requerimientos de implementar una modulación DPSK, se tienen en cuenta los recursos utilizados tanto para la modulación DBPSK y DQPSK (ambos mostrados en las tablas 3.8 y 3.9), como era de esperarse DQPSK necesita mayores capacidades y por ende define los recursos mínimos para la implementación de una modulación diferencial de fase, esta comparación se muestra en la tabla 3.10.

Device Utilization Summary

	Used	Available	Utilization
Total Number Slice Registers	1,111	11,776	9%
Number used as Flip Flops	1,110		
Number used as Latches	1		
Number of 4 input LUTs	1,398	11,776	11%
Number of occupied Slices	1,145	5,888	19%
Number of Slices containing only related logic	1,145	1,145	100%
Number of Slices containing unrelated logic	0	1,145	0%
Total Number of 4 input LUTs	1,572	11,776	13%
Number used as logic	1,023		
Number used as a route-thru	174		
Number used as Shift registers	375		
Number of bonded IOBs	11	372	2%
Number of BUFGMUXs	1	24	4%
Number of MULT18X18SIOs	8	20	40%
Number of RAMB16BWEs	8	20	40%
Average Fanout of Non-Clock Nets	2.33		

Tabla 3.8. Recursos requeridos DBPSK Spartan 3a.

Device Utilization Summary

	Used	Available	Utilization
Total Number Slice Registers	2,093	11,776	17%
Number used as Flip Flops	2,092		
Number used as Latches	1		
Number of 4 input LUTs	2,906	11,776	24%
Number of occupied Slices	2,160	5,888	36%
Number of Slices containing only related logic	2,160	2,160	100%
Number of Slices containing unrelated logic	0	2,160	0%
Total Number of 4 input LUTs	3,239	11,776	27%
Number used as logic	2,144		
Number used as a route-thru	333		
Number used as Shift registers	762		
Number of bonded IOBs	11	372	2%
Number of BUFGMUXs	1	24	4%
Number of MULT18X18SIOs	14	20	70%
Number of RAMB16BWEs	16	20	80%
Average Fanout of Non-Clock Nets	2.79		

Tabla 3.9. Recursos requeridos DQPSK Spartan 3a.

COMPONENTES	DBPSK	DQPSK
Total Number Slice Registers	9%	17%
Number of 4 input LUTs	11%	24%
Number of occupied Slices	19%	36%
Number of Slices containing only related logic	100%	100%
Number of Slices containing unrelated logic	0%	0%
Total Number of 4 input LUTs	13%	27%
Number of bonded IOBs	2%	2%
Number of BUFGMUXs	4%	4%
Number of MULT18X18SIOs	40%	70%
Number of RAMB16BWEs	40%	80%

Tabla 3.10. Comparación de recursos de FPGA para DBPSK y DQPSK

Con base a los resultados se da cumplimiento con el primer objetivo específico del proyecto que pretendía determinar las capacidades mínimas del hardware reconfigurable de acuerdo a las necesidades del sistema de comunicaciones; en este caso se hace referencia a sistemas que utilicen modulación diferencial en banda base.

A continuación se procede a realizar un gráfico comparativo que se muestra en la figura 3.4, con las siguientes convenciones:

1. Total Number Slice Registers.
2. Number of 4 input LUTs.
3. Number of occupied Slices.
4. Number of Slices containing only related logic.
5. Number of Slices containing unrelated logic.
6. Total Number of 4 input LUTs.
7. Number of bonded IOBs.
8. Number of BUFGMUXs.
9. Number of MULT18X18SIOs.
10. Number of RAMB16BWEs.

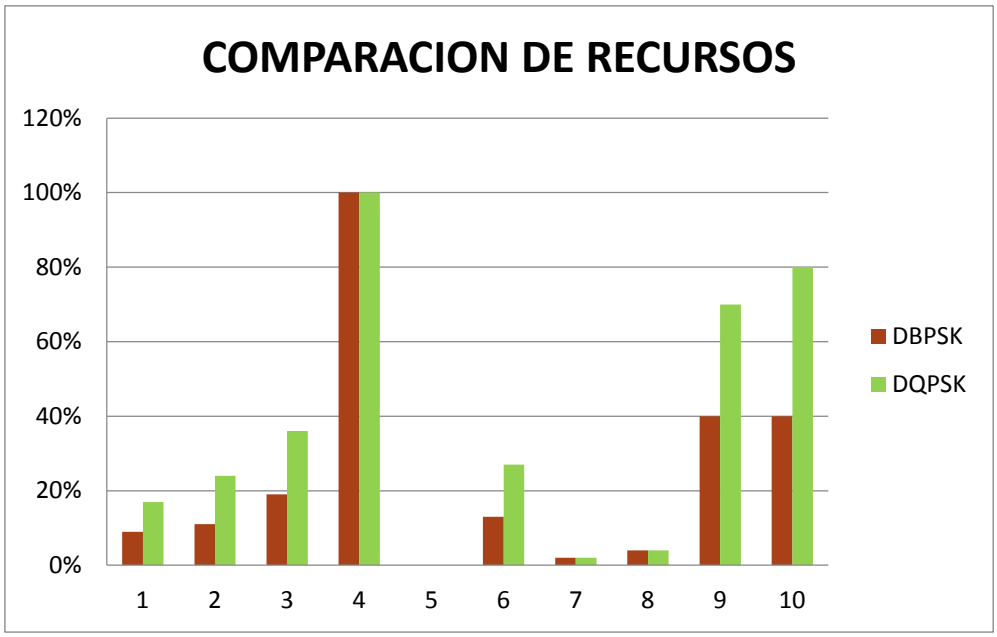

Figura 3.4. Comparación de uso recursos entre DBPSK y DQPSK.

En cuanto a la utilización de recursos se entiende que DQPSK es el sistema que más recursos necesita para la implementación; pero cabe mencionar que para sistemas de comunicaciones en banda base se utiliza un número de recursos menor en comparación con modelos pasa banda, por esto los dos sistemas no tuvieron problemas de recursos, incluso hubo una gran disponibilidad de estos.

Teniendo en cuenta la gráfica 3.4 se puede observar que los recursos utilizados por DQPSK tanto en *LUTs* y en *Slices* son mayores que DBPSK casi en el doble; esto es claro puesto que DQPSK maneja prácticamente el doble de compuertas que DBPSK, mientras que en la utilización de *IOBs* y demás recursos son manejados en un mismo porcentaje. Para determinar las capacidades mínimas de los sistemas se realizó la implementación sin necesidad de utilizar la memoria flash, puesto que los recursos fueron suficientes.

3.5. VALIDACIÓN DE LOS MODELOS IMPLEMENTADOS

Es importante analizar las diferencias entre la simulación en *System Generator* y el diseño implementado en la FPGA para validar los modelos de sistemas de comunicaciones DBPSK y DQPSK; este análisis compara la simulación normal de *System Generator* con la implementación hecha de los modelos en la FPGA. Para obtener una información bastante completa y confiable de los datos implementados al igual que los simulados se debió tomar varias muestras para cada cambio en las condiciones del sistema, en este caso se consideró suficiente 10 muestras por cada Eb/No.

Con la realización de las pruebas en implementación se da por cumplido el segundo objetivo específico definido anteriormente y se deja listo el camino para la realización de análisis del desempeño de los sistemas DBPSK y DQPSK. Los resultados mostrados a continuación son tomados de los escenarios 3 y 4, puesto que las velocidades de transmisión usadas en estos se adaptan a las características de la FPGA y además generar resultados acordes a simulación.

3.5.1. SISTEMA DBPSK

Eb/No[dB]	BER SIMULADA	BER IMPLEMENTADA.
0	0.142904853821	0.143187376018
1	0.105002784729	0.104866640363
2	0.071999645233	0.072027183976
3	0.044471359253	0.044379381835
4	0.024606704712	0.024542562011
5	0.011758041382	0.011737124249
6	0.004653930664	0.004627328179
7	0.001485252380	0.001492653228
8	0.000387382507	0.000371514759
9	0.000062751770	0.000063907168
10	0.000007057190	0.000006675720
11	0.000000190735	0.000000500865
12	0	0
13	0	0
14	0	0

Tabla 3.11. Comparación de resultados simulados e implementados en DBPSK.

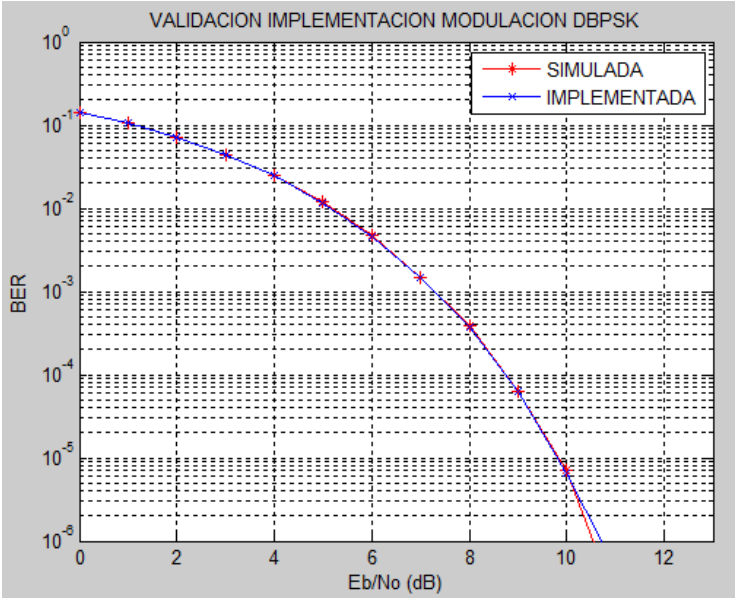

Figura 3.5.Validación de los resultados implementados en DBPSK.

Para el sistema DBPSK los resultados de validación se muestran en la tabla 3.11 o en la figura 3.5, donde se tiene una muy buena aproximación entre los valores simulados e implementados con una diferencia mínima de tasa de error por encima para los valores del sistema implementado. De esta manera se observa una coherencia con los valores obtenidos en simulación los cuales fueron validados, por ende, el sistema implementado muestra un comportamiento real en cuanto rendimiento y deja las bases para posteriores sistemas más complejos basados en la modulación DBPSK.

3.5.2. SISTEMA DQPSK

Eb/No[dB]	BER SIMULADA	BER IMPLEMENTADA.
0	0.134666760763	0.134740238939
1	0.096404817369	0.096347756453
2	0.064151128133	0.064239481091
3	0.038585768806	0.038649201393
4	0.020715289646	0.020729183850
5	0.009515232510	0.009529706835
6	0.003690507677	0.003682714700
7	0.001152568393	0.001127812266
8	0.000274372101	0.000263053178
9	0.000044918060	0.000043950812
10	0.000005878723	0.000004863719
11	0.000002193451	0.000000298610
12	0	0
13	0	0
14	0	0

Tabla 3.12. Comparación de resultados simulados e implementados en DQPSK.

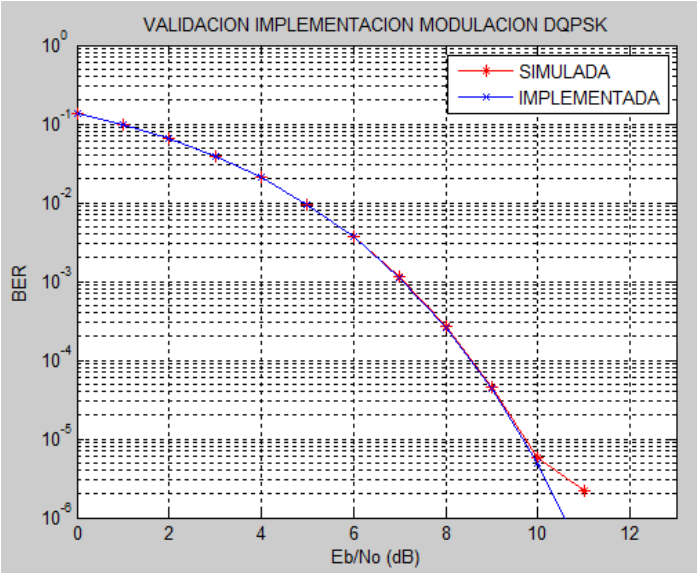

Figura 3.6.Validación de los resultados implementados en DQPSK.

Para la validación del sistema DQPSK se tiene una aproximación bastante cercana entre la curva de BER simulada e implementada, aclarando que a partir de 10 dB en la relación Eb/No la curva de BER simulada cambia su comportamiento y se mantiene por encima de la implementada como lo muestra la figura 3.12 o la tabla 3.5; cabe mencionar que el desempeño teórico y de implementación en DQPSK presentan una regularidad, por lo que se concluye que el sistema DQPSK implementado presenta un comportamiento real y este se puede utilizar para posteriores implementaciones de sistemas más complejos basados en la modulación DQPSK.

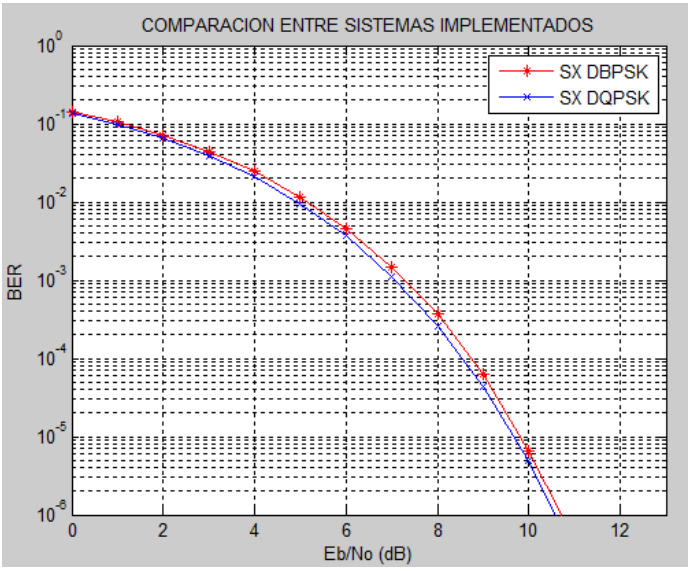

Figura 3.7. Comparación de los resultados implementados entre DBPSK y DQPSK.

La BER del sistema DQPSK tiene un comportamiento similar a la BER del sistema DBPSK aunque se mantiene levemente por debajo; teóricamente se da

un comportamiento similar que solidifica las bases para que estos resultados tengan validez.

Con este análisis se cumple el tercer objetivo específico que se definió como realizar el análisis del desempeño de los sistemas DBPSK y DQPSK implementados en hardware reconfigurable.

CAPÍTULO 4. CONCLUSIONES

- De acuerdo con las observaciones hechas de las tablas 2.3 y 2.4, se puede decir que para valores de Eb/No bajos cercanos a cero, el sistema que mejor respuesta tiene con respecto a la BER es el DQPSK, mientras que para valores altos de Eb/No ya se nota la gran diferencia en el rendimiento, y se puede observar que el sistema que mejor se comporta frente a rendimiento es el DBPSK.

- Para la implementación de sistemas diferenciales de fase en banda base en Hardware reconfigurable basta con utilizar una arquitectura de procesos que trabaje con punto fijo, puesto que como se observa en las figuras 2.36 y 2.37 los resultados teóricos y simulados son bastante aproximados, lo que garantiza un correcto funcionamiento con una precisión adecuada.

- Un sistema DQPSK en banda base en Hardware reconfigurable con canal AWGN presenta un mejor desempeño que un sistema DBPSK con las mismas condiciones, como se observa en la figura 3.7.

- Una FPGA *Spartan* 3A soporta sin inconvenientes la implementación de un sistema de comunicaciones con modulaciones diferenciales de fase bajo un modelo banda base; por ende el hardware reconfigurable se constituye en una opción viable para implementar sistemas digitales.

- De acuerdo a las formas de onda visualizadas, de los sistemas diseñados, se observa que el canal AWGN tiene un retraso en su funcionamiento de diez ciclos de reloj, los cuales son iguales a los tiempos de símbolo del sistema; por ende el funcionamiento del canal se ve restringido al tipo de modulación y a la cantidad de bits por símbolo que esta maneje.

- Los bloques construidos para el sistema de comunicaciones digitales con modulación DQPSK tuvieron una mejor respuesta ante el ruido, siendo DBPSK un sistema mucho más sensible a estos agentes a la hora de la simulación e implementación, esto como se puede ver figura 3.7.

- El proyecto está abierto para realizar cambios y mejoras en los componentes de los sistemas, creando de esta manera nuevas opciones como la implementación de un sistema que involucre la modulación D8PSK, debido a que la arquitectura modular del proyecto hace posible estas nuevas adiciones.

- Los dispositivos diseñados e implementados en este proyecto sirven como apoyo en las diferentes investigaciones sobre nuevas tecnologías de punta en comunicaciones digitales. De esta manera, se entrega una herramienta con un alto nivel didáctico para su uso en laboratorios, comprensión de sistemas de comunicación digital DMPSK y la mejora e implementación de nuevos diseños basados en este mismo.

- Gracias a la portabilidad y la arquitectura modular de sus componentes, este proyecto puede ser usado en diferentes prácticas de laboratorio tanto para el análisis de los sistemas de comunicaciones DMPSK como en el diseño de componentes para el procesamiento de señales digitales en asignaturas con este enfoque. Las alternativas de uso están abiertas para que el alumno pueda manipular uno a uno los elementos del dispositivo, teniendo la opción de hacer mejoras o simplemente, aprender del funcionamiento y arquitectura que este posee sin tener que dedicarse al diseño y construcción de un sistema completo de procesamiento digital.

- El Hardware implementado genera la posibilidad de desarrollar nuevos proyectos que complementen su interactividad con el usuario, además sirve como base en la elaboración de sistemas digitales en el área académica y profesional, dando solución a problemas en el sector de las comunicaciones digitales, aprovechando la era digital en la que se encuentra el país.

4.1. RECOMENDACIONES

- Debido a que la frecuencia de a la que trabaja una FPGA por lo general es elevada en comparación con dispositivos de medición como el osciloscopio se recomienda reducir la frecuencia para poder realizar la medición.

4.2. TRABAJOS FUTUROS

- Análisis del desempeño de un sistema de comunicaciones con modulación digital DBPSK/DQPSK completo en hardware reconfigurable, incluyendo sus componentes banda base y pasa banda.

- Uso de la SER para evaluación del desempeño de un sistema de comunicaciones con modulación digital de fase diferencial.

- Análisis del desempeño de un sistema de comunicaciones con modulación digital D8PSK en banda base o completo basado en hardware reconfigurable, con un canal multitrayecto.

- Descripción e implementación de bloque adicionales de codificación de canal y de fuente a los sistemas DBPSK/DQPSK basados en hardware reconfigurable.

- Análisis del desempeño de sistemas de comunicación digitales con modulaciones diferenciales de fase basados en equipos de procesamiento de alta capacidad.

ANEXOS

ANEXO 1: PROBABILIDAD DE ERROR DE BIT EN BPSK

En esta sección se muestra la deducción de la ecuación teórica de la BER que describe el desempeño de un sistema que utiliza modulación BPSK a través de un canal de ruido aditivo blanco Gaussiano (AWGN) [35].

Inicialmente se sabe que con BPSK, los dígitos binarios 1 y 0 serán representados respectivamente por valores analógicos $+\sqrt{Eb}$ y $-\sqrt{Eb}$ como se observa en la figura A1, el sistema BPSK se muestra en la figura A2.

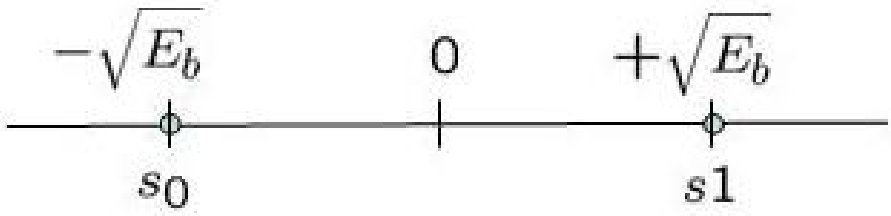

Figura A1. Mapeo de símbolos en BPSK.

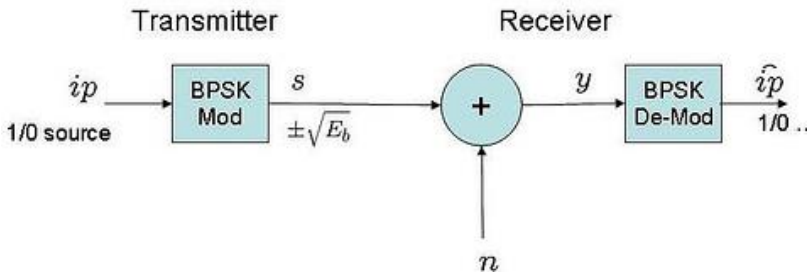

Figura A2. Sistema de comunicaciones BPSK.

La señal *n* mostrada en la figura A2 representa el ruido introducido por el canal, el cual será capaz de afectar el desempeño del sistema. En este caso el ruido introducido por el canal, se describe a través de la siguiente función de distribución de probabilidad.

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-u)^2}{2\sigma^2}} \qquad \text{(A1.1)}$$

Donde u es el valor esperado y σ^2 se denomina como la varianza, para este caso la ecuación es válida para una variable cuyo u=0 y σ^2=No/2.

Entonces se dice que la señal siempre va a estar afectada por el mismo ruido independientemente si se envía un 1 o un 0, esto se ve reflejado en la ecuación A1.2.

$$y = s + n \qquad \text{(A1..2)}$$

La ecuación A1.2 será dividida para los dos casos posibles.

$$y = s_0 + n, \text{ para cuando se trasmite un 0.} \qquad \text{(A1.3)}$$

$$y = s_1 + n, \text{ para cuando se trasmite un 1.} \qquad \text{(A1.4)}$$

Para esto las funciones de distribución de probabilidad para cada uno de los casos se verán expuestas en las ecuaciones A1.3 y A1.4, y posteriormente graficadas en la figura A3.

$$p(y/s_0) = \frac{1}{\sqrt{\pi No}} e^{\frac{-(x+\sqrt{Eb})^2}{No}} \qquad \text{(A1.5)}$$

$$p(y/s_1) = \frac{1}{\sqrt{\pi No}} e^{\frac{-(x-\sqrt{Eb})^2}{No}} \qquad \text{(A1.6)}$$

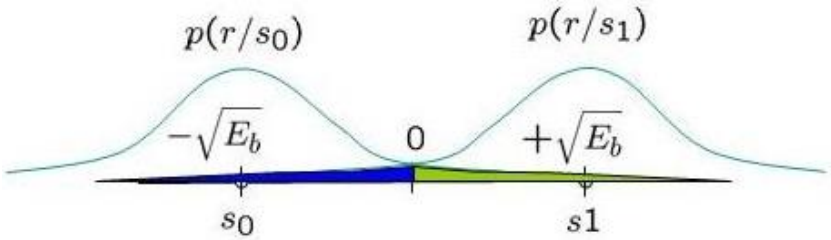

Figura A3. FDP[1] en BPSK.

Entonces se dice que la decisión de tomar un bit, ya sea cero o uno, depende del signo que tome el valor de la señal recibida *y*, que sí es positiva se asume que fue transmitido un 1, caso contrario si *y* es negativa se asume entonces que fue transmitido un 0.

Suponiendo que es enviado un 1 y se quiere calcular la probabilidad de que este salga erróneo aplicamos la integral a la función de distribución de probabilidad así:

$$p(e/s_1) = \frac{1}{\sqrt{\pi No}} \int_{-\infty}^{0} e^{\frac{-(x-\sqrt{Eb})^2}{No}} dx$$

Para que quede más clara la integral se hace un cambio de variable así:

$$z = \frac{x-\sqrt{Eb}}{\sqrt{No}} \qquad \text{(A1.7)}$$

[1] Funcion de densidad de probabilidad, denota la probabilidad que podría tomar cualquier variable aleatoria en cualquier instante.

$$dz = \frac{dx}{\sqrt{No}} \tag{A1.8}$$

Los límites de esta integral ahora serán:

$$z(-\infty) = \frac{-\infty - \sqrt{Eb}}{\sqrt{No}} = -\infty \tag{A1.9}$$

$$z(0) = \frac{-0 - \sqrt{Eb}}{\sqrt{No}} = -\sqrt{\frac{Eb}{No}} \tag{A1.10}$$

La integral que define el error cuando es enviado un 1, será:

$$p(e/s_1) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{-\sqrt{\frac{Eb}{No}}} e^{-z^2}\, dz \tag{A1.11}$$

Por la simetría que presenta la función de distribución de probabilidad, la probabilidad de error se puede ver así:

$$p(e/s_1) = \frac{1}{\sqrt{\pi}} \int_{\sqrt{\frac{Eb}{No}}}^{\infty} e^{-z^2}\, dz$$

Sabiendo de antemano que:

$$erfc(x) = \frac{2}{\sqrt{\pi}} \int_{x}^{\infty} e^{-z^2}\, dz \tag{A1.12}$$

Entonces

$$p(e|s_1) = \frac{1}{2} erfc\left(\sqrt{\frac{Eb}{No}}\right)$$

Haciendo el mismo procedimiento para cero se obtiene:

$$p(e|s_0) = \frac{1}{2} erfc\left(\sqrt{\frac{Eb}{No}}\right)$$

Ahora para el cálculo de la probabilidad total es necesario decir que la fuente generadora de bits es equiprobable, en cuanto a la generación de un cero o un uno, así:

$$p(s1) = p(s0) = \frac{1}{2} \tag{A1.13}$$

Obteniendo lo siguiente.

$$Pb = p(s1)p(e|s1) + p(s0)p(e|s0) \tag{A1.14}$$

$$Pb = \left(\frac{1}{2}\right)\left(\frac{1}{2} erfc\left(\sqrt{\frac{Eb}{No}}\right)\right) + \left(\frac{1}{2}\right)\left(\frac{1}{2} erfc\left(\sqrt{\frac{Eb}{No}}\right)\right) = \frac{1}{2} erfc\left(\sqrt{\frac{Eb}{No}}\right) \tag{A1.15}$$

ANEXO 2: VARIANZA DE n.

Se tiene que:

n= $\int_{-\infty}^{\infty}(n_{(t)}\phi_{(t)}\,dt)$

Ahora tenemos que la varianza de **n** está dada por:

E(n^2)= $\mathrm{E}(\int_{-\infty}^{\infty}(n_{(t)}\phi_{(t)}\,dt)^2)$ (A 2.1)

= $E(\iint_{-\infty}^{\infty} n_{(t)}\phi_{(t)}n_{(t)}\phi_{(t)}dtdT)$ (A 2.2)

= $\iint_{-\infty}^{\infty} E(n_{(t)}n_{(t)})\phi_{(t)}\phi_{(T)}dtdT$ (A 2.3)

= $\iint_{-\infty}^{\infty} \frac{N_0}{2}\delta_{(t-T)}\,\phi_{(t)}\,\phi_{(T)}dtdT$ (A 2.4)

= $\iint_{-\infty}^{\infty} \frac{N_0}{2}\delta_{(t-T)}\,\phi_{(t)}\,\phi_{(T)}dtdT$ (A 2.5)

= $\frac{N_0}{2}\iint_{-\infty}^{\infty}\,\phi_{(t)}{}^2\,dtdT$ (A 2.6)

La varianza de n es:

E(n^2)= $\frac{N_0}{2}$ (A 2.7)

ANEXO 3: REQUISITOS MÍNIMOS DE INSTALACION DE XILINX.

Las recomendaciones de Hardware para *System Generator* en diferentes sistemas operativos se muestran en las tablas A.1 y A.2.

Recomendaciones	Notas
2.00 GB de RAM	
600 MB de espacio en disco duro	Requerimiento mínimo
Plataforma Xilinx Co-Simulation	Necesarios para el flujo de hardware Co-Simulación

Tabla A.1. Requisitos para Sistema operativo Windows.

Recomendaciones	Notas
4.00 GB de RAM	
600 MB de espacio en disco duro	Requerimiento mínimo
Plataforma Xilinx Co-Simulation	Necesarios para el flujo de hardware Co-Simulación

Tabla A.2. Requisitos para sistema operativo Linux.

ANEXO 4: DESCRIPCIÓN DEL PAQUETE XILINX.

A continuación se hará una descripción del paquete en el cual viene incluido el *System Generator* llamado *Xilinx.*

SYSTEM GENERATOR.

System Generator para DSP es una plataforma software que usa las herramientas de *The MathWorkMATLAB/Simulink* para representar una visión abstracta de alto nivel del sistema de DSP, y que automáticamente genera el código HDL de la función de DSP desarrollada usando los más optimizados[12].

De esta forma, *System Generator* permite modelar directamente mediante un entorno de alto nivel muy flexible, robusto y fácil de utilizar sistemas de DSP y de alto rendimiento para una plataforma hardware específica. Un diseño desarrollado con esta herramienta puede componerse de una gran variedad de elementos: bloques específicos de *System Generator*, código de un lenguaje de descripción de hardware tradicional (*VHDL, Verilog*) y funciones derivadas del lenguaje programación *MATLAB*. Así, todos estos elementos pueden ser usados simultáneamente, simulados en conjunto y sintetizados para obtener una función de DSP sobre FPGA. El aspecto más interesante de trabajar en *MATLAB/Simulink* es poder emplear la poderosa herramienta de simulación de sistemas *Simulink* para realizar la verificación del diseño.

Una de las características más importantes de *Xilinx System Generator* es que posee abstracción aritmética, es decir, trabaja con representaciones en punto fijo con una precisión arbitraria, incluyendo la cuantización y el sobreflujo. También puede realizar simulaciones tanto en doble precisión como en punto fijo.

Más de 90 bloques de construcción de DSP se encuentran en el *Blockset Xilinx DSP* para *Simulink*. Estos bloques son los bloques DSP comunes de construcción tales como sumadores, multiplicadores y registros. También se incluyen un conjunto de bloques de construcción compleja DSP, tales como bloques de corrección de errores, filtros FFT y memorias. Diversidad de bloques de *System Generator* utilizados se explican en el anexo 6.

PROJECT NAVIGATOR

Navegador de proyectos gestiona tus archivos de diseño y le permite ejecutar procesos para mover el diseño de la creación del diseño a través de la implementación de la programación del dispositivo de destino Xilinx ®. Las

secciones siguientes describen las diferentes partes del entorno del *Project Navigator* [31].

Las siguientes son las principales áreas del Navegador de proyectos, que se muestran en la siguiente figura:

- **Toolbar -** proporciona un cómodo acceso a los comandos de menú de uso frecuente.
- **Panel Diseño -** proporciona acceso a las siguientes áreas:
  - **Vista Diseño -** le permite seleccionar una fase de diseño de las "Fuentes" Area, que controla los archivos de fuentes que se muestran en el panel Jerarquía.
  - **Panel Jerarquía -** le permite ver y administrar los archivos de origen del diseño.
  - **Panel Procesos -** le permite ejecutar procesos, que se mueven el diseño de la creación del diseño a través de la programación del dispositivo. Además, puede ejecutar los procesos que le permiten analizar y mejorar su diseño.
- **Espacio de trabajo -** le permite ver y editar su diseño utilizando diversas herramientas, y permite el acceso a los informes en el Resumen de diseño.
- **Ventana Transcripción -** le permite ver el registro de salida, mensajes, errores y advertencias, ya que se genera.

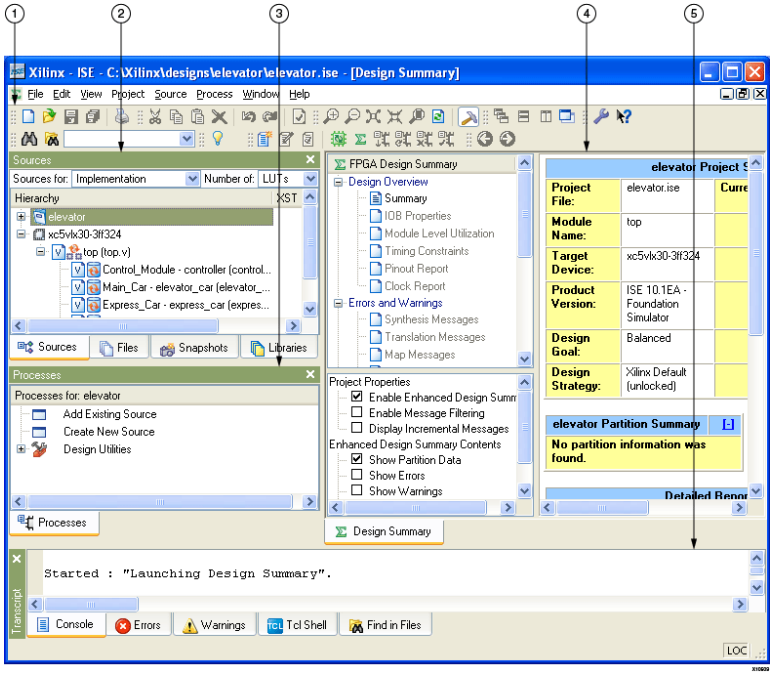

Figura A4. Project Navigator.

ISE IMPACT

Para la transferencia de la configuración al hardware la herramienta IMPACT permite compilar los archivos .bit en una única imagen para una única cadena

en los diferentes modos de configuración del FPGA. También sirve para compilar los distintos conjuntos de datos en una imagen de la Memoria Flash (fichero *.MPM), donde se guardan también los datos de configuración de cada conjunto de datos. Utilizando este software también se programa la memoria Flash [12].

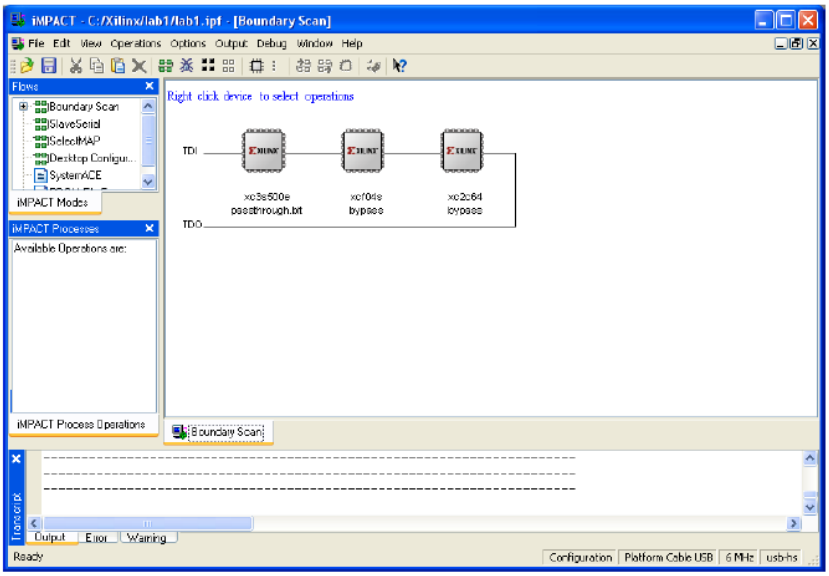

Figura A5. Interfaz IMPACT.

ANEXO 5: DESCRIPCIÓN DEL PAQUETE DE MATHWORK.

Para el correcto funcionamiento del software requiere como plataforma de despliegue otro software de soporte contenido en el paquete de *MathWork* descrito a continuación:

MATLAB

MATLAB ® es un lenguaje de alto nivel y un entorno interactivo para el cálculo numérico, visualización y programación. Usando MATLAB, puede analizar los datos, desarrollar algoritmos y crear modelos y aplicaciones. El lenguaje, las herramientas y funciones matemáticas integradas que permiten explorar múltiples enfoques y llegar a una solución más rápida que con hojas de cálculo o lenguajes de programación tradicionales, como C / C + + o Java [32].

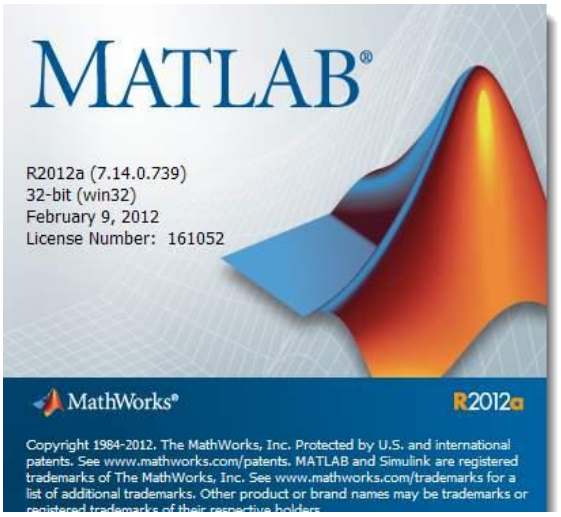

Figura A6. Matlab 2012 a.

SIMULINK

Simulink es un entorno de diagrama de bloques para la simulación multidominio y diseño basado en modelos. Es compatible con la simulación, generación automática de código, y la prueba continua y verificación de sistemas embebidos [33].

Simulink ofrece un editor gráfico, bibliotecas de bloques personalizables y solucionadores para el modelado y simulación de sistemas dinámicos. Está integrado con MATLAB, lo que le permite incorporar algoritmos de MATLAB en modelos y resultados de la simulación de exportación a MATLAB para su posterior análisis.

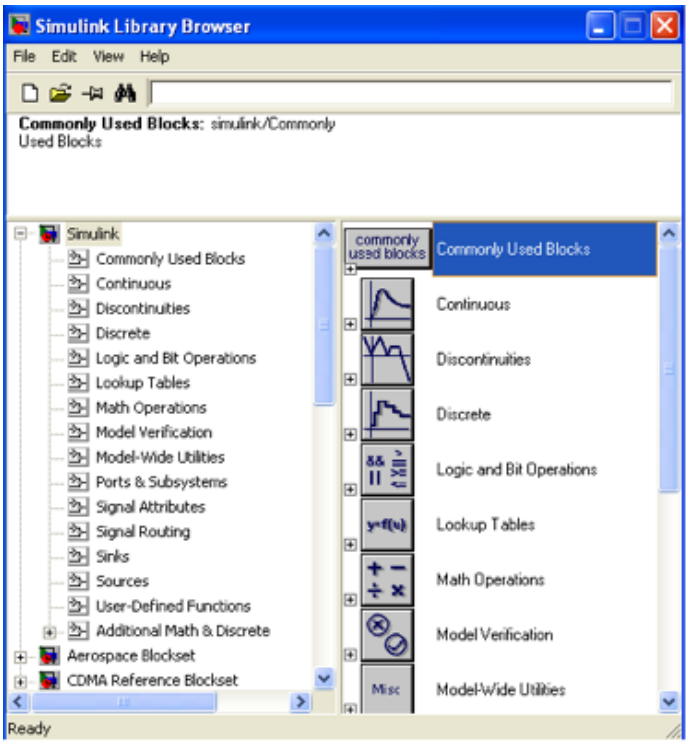

Figura A7. Herramienta Simulink.

BERTOOL

La aplicación BERTool le permite analizar la tasa de error de bit (BER) la ejecución de los sistemas de comunicaciones. BERTool calcula la BER en función de la relación señal-ruido. Analiza el rendimiento, ya sea con simulaciones Monte-Carlo de funciones de MATLAB y modelos de Simulink o con expresiones teóricas de forma cerrada para determinados tipos de sistemas de comunicación [34].

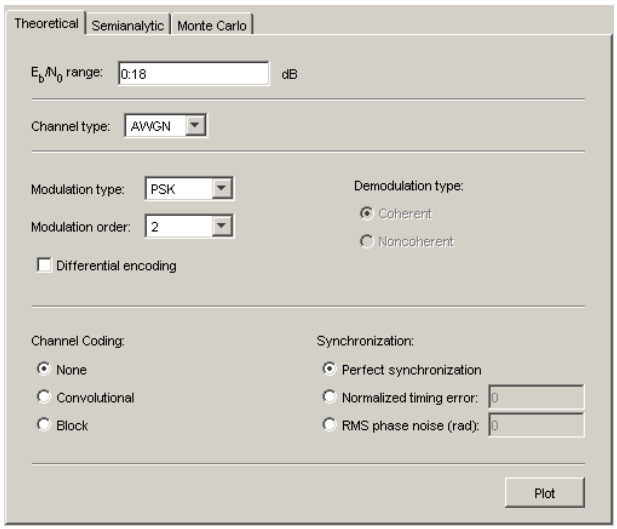

Figura A8. Herramienta Bertool.

ANEXO 6: ELEMENTOS UTILIZADOS EN EL MODELO EN SYSTEM GENERATOR [36].

A continuación se detallan los elementos utilizados en el modelo en System Generator, los cuales cuentan con los siguientes parámetros comunes:

Arithmetic Type. Permite escoger el tipo de dato de salida. Los tipos de datos disponibles son: booleano, sin signo y con signo en complemento a dos.
Number of bits. Especifica el número de bits con el que se representa el valor de salida del elemento.
Binary point. Indica la posición del punto decimal del valor de salida.
Quantization and Overflow. *Quantization* cuenta con dos opciones, truncar y redondeo. La primera para eliminar los bits que se encuentran a la derecha del Bit Menos Significativo (LSB, Least Significant Bit) representable y la segunda que permite redondear al valor más cercano representable. Para Overflow, *System Generator* ofrece las opciones recortar y saturar. Recortar para descartar los bits que están a la izquierda del Bit Más Significativo (MSB, Most Significant Bit) representable y saturar que permite tomar el valor positivo más grande y el valor negativo más pequeño.
Sample Period. Permite especificar un período de muestreo particular, aunque por lo general, cada elemento detecta la tasa de muestreo de entrada para generar la tasa de muestro de salida.
Latency. Es el número de ciclos de reloj en que se retarda la señal de salida, por lo cual debe ser un valor entero positivo.

Elemento 1. *System Generator Token*. Mostrado en la figura A9, funciona como un panel de control para administrar los parámetros del sistema modelado y la simulación. También es usado para invocar el generador de código de programación del FPGA.

Todo modelo de Simulink® que contenga algún elemento de Xilinx debe contar por lo menos con un System Generator Token, a partir del cual es posible especificar cómo debe ser configurado el generador de código y la simulación.

Los parámetros específicos a configurar son:

- ***Compilation.*** Especifica el tipo de resultado de compilación que debe producirse cuando se invoca el generador de código.
- ***Part*.** Define el FPGA a usarse.
- ***Synthesis tool.*** Especifica la herramienta a usarse para sintetizar el diseño. Las posibilidades son Synplicity's Synplify Pro, Synplify, y Xilinx's XST.
- ***Hardware Description Language*.** Especifica el lenguaje HDL a usarse para la compilación del diseño. Las posibilidades son VHDL and Verilog.

- ***Target directory***. Especifica dónde debe guardar System Generator los resultados de compilación.
- ***Project type*.** Selecciona el tipo de archivo de proyecto que debe generarse, si Project Navigator o PlanAhead.
- ***FPGA clock period(ns).*** Dado en nanosegundos, define el periodo del reloj del sistema. El valor debe ser entero. El período es pasado a una herramienta de implementación de Xilinx a través de un archivo de restricciones, donde es usado como el periodo de restricción global.
- ***Clock pin location*.** Define el pin de ubicación del reloj del hardware. Esta información es pasada a la herramienta de implementación de Xilinx a través de un archivo de restricción. Esta opción no debe ser especificada si el diseño en System Generator va a ser incluido como parte de un diseño más grande.
- ***Simulink system period(sec).*** Define el período del sistema de Simulink® en unidades de segundos. El período de sistema de Simulink® es el máximo común divisor de los periodos de muestreo que aparecen en el modelo. Estos periodos de muestreo se establecen explícitamente en los cuadros de diálogo de cada uno de los elementos, y se heredan de un elemento a otro.

Figura A9. System Generator Token

Elemento 2. Registro de desplazamiento con realimentación lineal. Mostrado en la figura A10, implementa un Registro de Desplazamiento con Realimentación Lineal (LFSR, Linear Feedback Shift Register), compuesto por una cadena de biestables, el cual puede generar largas secuencias pseudoaletorias de unos y ceros (secuencias m).

Los parámetros específicos a configurar son:

- **Type Fibonacci o Galois.** Este campo especifica la estructura de realimentación. Fibonacci tiene una compuerta XOR (o XNOR) al inicio del registro de desplazamiento que opera los derivadores[1] de realimentación junto con el valor de entrada del primer biestable. Galois cuenta con una compuerta XOR (o XNOR) por cada derivador, la cual realiza operaciones a partir de los datos provenientes del derivador y datos provenientes del último biestable, ubicado en la salida del registro de desplazamiento.

[1] Las salidas de cada biestable que influyen en la entrada del registro de desplazamiento (realimentación) se denominan derivadores y se constituyen en cada uno de los valores del polinomio generador.

- **Gate type XOR o XNOR.** Este campo especifica la compuerta usada por las señales de realimentación.
- **Number of bits in LFSR.** Especifica el número de biestables del registro de desplazamiento.
- **Feedback polynomial.** Este campo especifica el polinomio generador de la secuencia pseudoaleatoria, y con ello, los puntos de ubicación de cada derivador en el registro de desplazamiento, el valor debe ingresarse en hexadecimal con comillas sencillas.
- **Initial Value.** Especifica el valor de semilla a partir del cual el LFSR inicia la secuencia pseudoaleatoria. El valor inicial no puede ser ceros cuando se ha escogido la compuerta tipo XOR y no puede ser unos cuando se ha escogido XNOR, de ser así el LFSR no funciona.

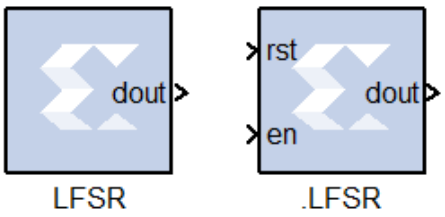

Figura A10. Elemento LFSR

Elemento 3. Convert. Mostrado en la figura A11, cambia la cuantificación de un número pero no el valor. Este elemento puede alterar el número de bits usado para representar un número y puede ser utilizado para convertir del tipo sin signo al tipo con signo y viceversa. Frecuentemente es usado para limitar el número de bits de la parte decimal del número de salida de una operación de multiplicación.

Este elemento no cuenta con parámetros específicos.

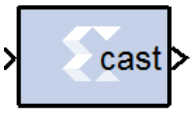

Figura A11. Elemento Convert

Elemento 4. Counter. Mostrado en la figura A12, implementa un contador que puede correr libremente a lo largo del rango establecido para tal efecto y después del cual, reinicia el conteo. La salida del elemento puede ser del tipo sin signo o con signo, y se le puede dar un valor inicial o bien proporcionarle un puerto por el cual cargar un valor determinado. También se puede configurar el tamaño de paso que incrementa el contador.

Los parámetros específicos a configurar son:

- ***Counter type***. Especifica el modo de conteo, ya sea libre o limitado.
- ***Initial value*.** Indica el valor en el que comienza a contar.

- ***Count to value*.** En el modo limitado indica hasta qué valor debe contar. Si se configura con *Inf* cuenta hasta el máximo valor posible. Este número no puede ser el mismo que aparece en *Initial value*.
- ***Step*.** Tamaño del paso con el que se cambia de un valor a otro en el conteo.
- ***Count direction.*** Indica si es un contador ascendente o descendente, o bien se habilita un puerto para que se indique si va a ser ascendente o descendente.

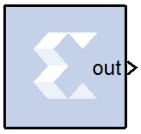

Figura A12. Elemento Counter

Elemento 5. Mux. Mostrado en la figura A13, implementa un multiplexor donde la entrada de selección es del tipo sin signo y el número de entradas de datos va desde 2 hasta 32.

El parámetro específico a configurar es:

- ***Number of inputs***. Especifica el número de buses de datos de entradas.

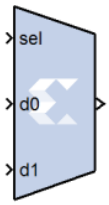

Figura A13. Elemento Mux

Elemento 6. Delay. Mostrado en la figura A14, es un registro de desplazamiento, de longitud configurable, que induce un retardo sobre la señal. Permite añadir latencia al diseño. Los datos de la entrada aparecerán a la salida después de un determinado número de períodos de la señal de reloj.

Este elemento no cuenta con parámetros específicos.

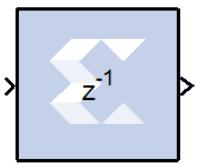

Figura A14. Elemento Delay

Elemento 7. *Serial to Parallel*. Mostrado en la figura A15, toma los datos en serie de entrada y los convierte a datos en paralelo a la salida, creando un único dato de salida para cada determinada cantidad de datos de entrada. Cuenta con un retardo mínimo de un periodo de muestreo, con respecto al

periodo de muestreo de la señal de salida. Se produce un error cuando el número de bits de salida no se puede dividir por el número de bits de entrada.

El parámetro específico a configurar es:

- ***Input order.*** Asignación del primer bit de salida al LSB o MSB de la secuencia.

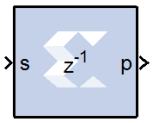

Figura A15. Elemento Serial to Parallel

Elemento 8. *Down Sample*. Mostrado en la figura A16, reduce la tasa de muestreo en el sitio donde se coloque, para lo cual muestrea cada cierto intervalo de tiempo, dando la posibilidad de elegir si se desea la muestra del inicio o del final del intervalo; el valor de cada muestra es mantenido hasta que la próxima muestra es tomada.

Los parámetros específicos a configurar son:

- ***Sampling Rate***. Tiene que ser un entero mayor o igual que dos. Es la relación entre la tasa de entrada y la de salida, es esencialmente un divisor de tasa de muestreo.
- ***Sample***. Puede ser Primer Valor de la Muestra o Último Valor de la Muestra, indicando si se desea la muestra inicial o la muestra final de la ventana de muestreo respectivamente.

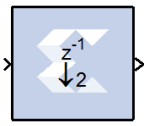

Figura A16. Elemento Down Sample

Elemento 9. *Mcode*. Mostrado en la figura A17, permite el modelado de estructuras de control hardware a partir del lenguaje de programación de *Matlab*®; para este fin se genera un script que es ejecutado dentro del *Mcode*.

Este elemento no es adecuado para códigos que describen una operación algorítmica tal como un filtro de Respuesta Finita al Impulso (FIR, Finite Impulse Response) o matriz inversa, pero en cambio proporciona un método conveniente y eficiente para la aplicación de máquinas de estado y condiciones complejas de multiplexación.

El *Mcode* soporta las siguientes operaciones en lenguaje de programación Matlab®:

- Asignación de sentencias.

- Sentencias simples y compuestas *if/else/eslseif*.
- Sentencias *switch*.
- Expresiones aritméticas que involucran solo suma y resta.
- Multiplicación.
- División por una potencia de dos.

Este elemento no cuenta con parámetros específicos.

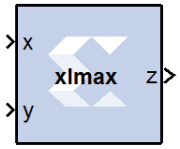

Figura A17. Elemento Mcode

Elemento 10. *Constant*. Mostrado en la figura A18, genera un valor constante que puede ser sin signo, con signo o booleano.

Los parámetros específicos a configurar son:

- ***Constant Value*.** Indica el valor de la constante.
- ***Sampled Constant*.** Permite asociar un período determinado a la salida constante del elemento. Esta opción suele ser usada para interconectar el elemento *Constant* con algunos otros que requieren esta característica.

Figura A18. Elemento Constant

Elemento 11. *Mult*. Mostrado en la figura A19, implementa un multiplicador de dos entradas.

Este elemento no cuenta con parámetros específicos.

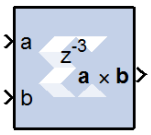

Figura A19. Elemento Mult

Elemento 12. *White Gaussian Noise Generator*. El elemento Generador de Ruido Blanco Gaussiano (WGNG, *White Gaussian Noise Generator*), mostrado en la figura A20, genera ruido AWGN usando una combinación del algoritmo de *Box-Muller* y el teorema del límite central[1].

[1] Indica que la suma de n variables aleatorias independientes entre sí, corresponde a una variable aleatoria con distribución normal cuando la suma de estas variables es lo suficientemente grande.

El algoritmo de Box-Muller genera una variable aleatoria normal estándar de la forma $X \sim N(\mu_x, \sigma_x^2)$ utilizando una transformación de dos variables aleatorias independientes entre sí que están uniformemente distribuidas en (0,1). Posteriormente dichos valores se almacenan en Memorias de Solo Lectura (ROM, *Read Only Memory*) direccionados con variables aleatorias uniformes.

Las salida de cuatro subsistemas paralelos Box-Muller son promediados para obtener una PDF gaussiana, generado así ruido normalizado N(0,1). La latencia total del generador es de diez ciclos de reloj y el puerto de salida del ruido es un número de doce bits con signo y siete bits después del punto binario.

El parámetro específico a configurar es:

- ***Seed.*** Define el patrón de ruido del WGNG.

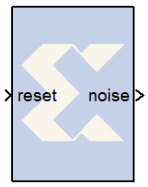

Figura A20. Elemento White Gaussian Noise Generator

Elemento 13. *AddSub*. Mostrado en la figura A21, implementa un sumador o un restador a la vez o, de acuerdo a la configuración, permite implementar las dos operaciones de manera secuencial de acuerdo a alguna condición externa.

El parámetro específico a configurar es:

- ***Operation***. Suma, resta o suma/resta. Cuando la operación suma/resta está seleccionada aparece un puerto de entrada que controla la operación a realizar; la señal que controla este puerto debe ser de tipo booleano. Si está a nivel alto el elemento actúa como restador, en caso contrario como sumador.

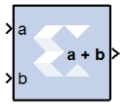

Figura A21. Elemento AddSub

Elemento 14. *Up Sample*. Mostrado en la figura A22, incrementa la tasa de muestreo en el punto en el que se coloque. La entrada es sobremuestreada L veces en un período de muestreo de la señal de entrada. La salida es constituida ya sea por L muestras de la señal de entrada, o por una muestra de la señal de entrada y L-1 ceros.

Los parámetros específicos a configurar son:

- ***Sampling rate***. Tiene que ser un entero mayor o igual a dos. Este valor es la relación entre el período de muestreo a la salida y a la entrada del elemento. Por ejemplo, si este valor es dos, indica que la tasa de muestreo de salida es dos veces la de entrada.
- ***Copy samples.*** Permite elegir qué hacer con las muestras adicionales debidas al incremento de la tasa de muestreo. Si se selecciona esta casilla se copian las muestras de entradas, en caso contrario se insertan ceros.

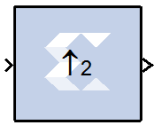

Figura A22. Elemento Up Sample

Elemento 15. *Parallel to Serial.* Mostrado en la figura A23, toma los datos de entrada en paralelo y los separa a la salida para transmitirlos en serie, generando por cada dato de entrada un conjunto de datos de salida en serie. Se puede configurar el orden de salida para que empiece por el LSB o por el MSB.

El parámetro específico a configurar es:

- ***Output order.*** Indica si se empieza a transmitir primero el LSB o el MSB.

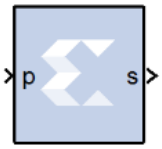

Figura A23. Elemento Parallel to Serial.

Elemento 16. *Accumulator.* Mostrado en la figura A24, implementa un acumulador de suma o de resta, donde el tamaño de la salida y la entrada deben tener la misma cantidad de bits de la parte entera y decimal y deben ser del mismo tipo.

El parámetro específico a configurar es:

- ***Operation.*** Suma o resta.

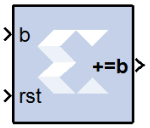

Figura A24. Elemento Accumulator.

Elemento 17. *Relational.* Mostrado en la figura A25, implementa un operador relacional lógico. Las comparaciones permitidas son:

- Igual (r = t).
- Distinto (r != t).
- Menor que (r < t).
- Mayor que (r > t).
- Menor o igual que (r <= t).
- Mayor o igual que (r >= t).

El parámetro específico a configurar es:

- ***Comparison.*** Indica la comparación a realizar.

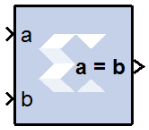

Figura A25. Elemento Relational.

Elemento 18. *Dual Port Ram.* Mostrado en la figura A26, implementa una Memoria de Acceso Aleatorio (RAM, *Random Accesss Memory*). Cuenta con dos conjuntos independientes de puertos para la lectura y escritura simultánea. Puertos independientes de habilitación para dirección, datos y escritura permiten el acceso compartido a un único espacio de memoria. Por defecto, cada conjunto de puertos tiene un puerto de salida y tres puertos de entrada: dirección, datos de entrada, y habilitación de escritura. Opcionalmente, se puede agregar un puerto de habilitación y uno de señal de reseteo síncrono a cada conjunto de puertos de entrada.

Los parámetros específicos a configurar son:

- ***Depth.*** Indica el tamaño de la memoria (el número de palabras que se pueden guardar), debe ser un entero positivo.
- ***Initial value vector***. Indica el contenido inicial de la memoria. Cuando la longitud del vector excede el tamaño de la memoria, los valores con índice mayor se ignoran. Cuando es menor, las posiciones superiores de la memoria se inicializan en cero.

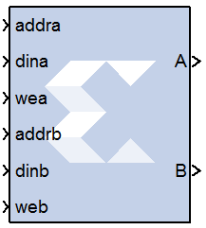

Figura A26. Elemento Dual Port Ram.

Elemento 19. *Gateway In*. Mostrado en la figura A27, son las entradas de la parte *Xilinx* del diseño en *Simulink*®. Convierten los tipos de datos enteros, dobles y de punto fijo de *Simulink*® en los tipos de datos de punto fijo para *System Generator*. Cada elemento define un puerto de entrada en el código de *HDL* generado por *System Generator*.

El parámetro específico a configurar es:

- ***IOB Pad Location***, e.g. ('MSB',…, 'LSB'). En este campo se indican, de más significativo a menos significativo, la localización de los pines (hardware) de entrada asociados a este elemento.

Figura A27. Elemento Gateway In.

Elemento 20. *Gateway Out*. Mostrado en la figura A28, son las salidas de la parte *Xilinx* del diseño en *Simulink*®. Este elemento convierte los datos de punto fijo a datos de tipo doble.

Cada elemento define un puerto de salida en el código *HDL* generado, o bien se usa simplemente como un punto de prueba que después será eliminado del diseño hardware.

Los parámetros específicos a configurar son:

- ***Translate into Output Port*.** Representa el elemento como un puerto en la generación del código HDL.
- ***Specify IOB Location Constraint*.** Habilita un campo para la descripción de la localización de los pines de salida para estos puertos.

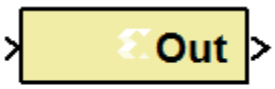

Figura A28. Elemento Gateway Out.

APÉNDICES

APÉNDICE 1: PROBABILIDAD DE ERROR DE BIT EN DBPSK

En esta sección se muestra la deducción de la ecuación teórica de la BER que describe el desempeño de un sistema que utiliza modulación DBPSK a través de un canal de ruido aditivo blanco Gaussiano (AWGN). Basándose en la demostración de la expresión teórica de la BER para BPSK. El sistema DBPSK se muestra en la figura Ap1.

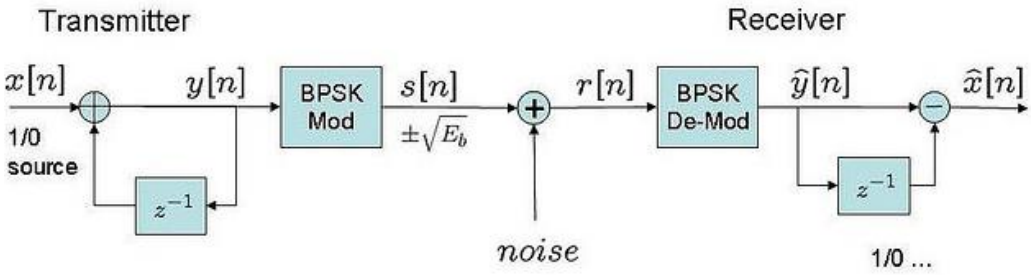

Figura Ap1. Sistema de comunicaciones DBPSK.

La ecuación que representa el modulador DBPSK es:

$$y[n] = y[n-1] \oplus x[n] \qquad \text{(AP.1)}$$

La ecuación que representa el demodulador DBPSK es:

$$\hat{y}[n] = \hat{y}[n-1] \oplus \hat{x}[n] \qquad \text{(AP.2)}$$

Teniendo así que la probabilidad de error del sistema se define si $x[n]$, que es la señal de entrada del sistema, sería diferente de $\hat{x}[n]$ que es la señal que se recibe, esto se muestra en la ecuación AP.3.

$$Pb_{dbpsk} = Prob\{\hat{x}[n] \neq x[n]\} \qquad \text{(AP.3)}$$

Acudiendo a las ecuaciones AP.1 y AP.2, se puede decir que para que el evento $\hat{x}[n] \neq x[n]$ se cumpla pueden suceder dos eventos alternos, que $\hat{y}[n-1] \neq y[n-1]$ o que $\hat{y}[n-1] = y[n-1]$, entonces la expresión de probabilidad queda de la siguiente manera:

$$Pb_{dbpsk} = P\{\hat{x}[n] \neq x[n] \mid \hat{y}[n-1] \neq y[n-1]\}.P\{\hat{y}[n-1] \neq y[n-1]\} + P\{\hat{x}[n] \neq x[n] \mid \hat{y}[n-1] = y[n-1]\}.P\{\hat{y}[n-1] = y[n-1]\} \qquad \text{(AP.4)}$$

La ecuación AP.4 se basa en la veracidad del evento.

$$\hat{x}[n] \neq x[n]$$

Pero este sucede si y solo sí.

$$\hat{y}[n] \ominus \hat{y}[n-1] \neq y[n] \ominus y[n-1] \qquad \text{(AP.5)}$$

Que expresado de otra manera quedaría así:

$$\hat{y}[n] \ \neq y[n] \ominus y[n-1] \oplus \hat{y}[n-1] \qquad \text{(AP.6)}$$

De la ecuación AP.6 se puede deducir que puede aparecer un error en el sistema en solo dos casos. El primer caso es que si $\hat{y}[n] = y[n]$, y además se presente que $\hat{y}[n-1] \neq y[n-1]$, entonces el segundo caso sería que si $\hat{y}[n] \neq y[n]$, pero $\hat{y}[n-1] = y[n-1]$, a partir de la anterior deducción, la expresión de probabilidad de error quedaría resuelta así:

$$Pb_{dbpsk} = P\{\hat{y}[n] = y[n]\}.P\{\hat{y}[n-1] \neq y[n-1]\} + P\{\hat{y}[n] \neq y[n]\}.P\{\hat{y}[n-1] = y[n-1]\} \qquad \text{(AP.7)}$$

Pero como se sabe que un sistema DBPSK es una derivación de uno BPSK, la expresión anterior puede ser detallada en términos de la probabilidad de error presente en un sistema BPSK, así:

$$Pb_{dbpsk} = (1 - Pb_{bpsk})Pb_{bpsk} + Pb_{bpsk}(1 - Pb_{bpsk}) \qquad \text{(AP.8)}$$

Obteniendo como resultado

$$Pb_{bpsk} = 2Pb_{bpsk}(1 - Pb_{bpsk}) = erfc\left(\sqrt{\frac{Eb}{No}}\right)\left(1 - \frac{1}{2}erfc\left(\sqrt{\frac{Eb}{No}}\right)\right) \qquad \text{(AP.9)}$$

REFERENCIAS BIBLIOGRÁFICAS

[1] E. Hernández y M. Robles, “Material de Clase: Sistemas de Telecomunicaciones II,” Facultad de Ingeniería en tecnologías de la información y Comunicación, Universidad Tecnológica de Puebla.

[2] R. Herrera y J. M. Gutierrez, “Conocimiento, innovación y desarrollo,” Primera Edición, 2011. ISBN 978-9968-900-10, San José, Costa Rica, .Disponible en: http://catedrainnovacion.ucr.ac.cr/librocid.pdf

[3] W. Tomasi. *Sistemas De Comunicaciones Electrónicas*. Cuarta Edicion, Pearson educación, 2003. ISBN 9702603161, 9789702603160, pp 467-524.

[4] B. Sklar, “Digital Communications: Fundamentals & Applications,” Segunda Edición, Prentice-Hall PTR, 2001. ISBN 9780130847881, California, Los Angeles, .Disponible en: http://www.2shared.com/file/KwaE8t8-/Comunicaciones_Digitales_Berna.html

[5] S. Haykin, “Sistemas de Comunicaciones,” Cuarta Edición, Limusa Wiley, S.A., 2002. ISBN 9789681863074, Toronto, Canadá, .Disponible en: http://www.slideshare.net/leosparr/sistemas-de-comunicacion-simon-haykin

[6] A. M. Wyglinski, D. Pu, “Digital Communication Systems Engineering with Software-Defined Radio,” Artech House, 2013. ISBN 9781608075263 .Disponible en: http://books.google.com.co/books?id=3z2uklY5O3oC&printsec=frontcover&hl=es&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

[7] R. J. Tocci y N. S. Widmer, “Sistemas digitales: principios y aplicaciones,” Octava Edición, Pearson educación, 2003. ISBN 9789702602972. Disponible en: http://books.google.com.co/books?id=bmLuH0CsIh0C&printsec=frontcover&hl=es&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

[8] Agilent (2010). IQ Modulation. Santa Clara CA, EE.UU. Recuperado (2014, abril 1) de http://www.home.agilent.com/upload/cmc_upload/All/IQ_Modulation.htm?cmpid=zzfindnw_iqmod&cc=CO&lc=eng

[9] Complex to Real (2002). *All about Modulation - Part I*. Charan Langton, EE.UU. Recuperado (2014, abril 1) de http://people.seas.harvard.edu/~jones/cscie129/papers/modulation_1.pdf

[10] J. Luque y S. Clavijo, “Modulación de señales Digitales,” Facultad de Ingeniería Electrónica y Telecomunicaciones, Universidad de Sevilla, Departamento de Tecnología Electrónica.

[11] E. Rudas." Análisis Comparativo de BER para Modulaciones BPSK y QPSK en distintos tipos de canales", Universidad Tecnológica de Panamá.

[12] G. Fierro, “implementación de la etapa de recepción de un sistema de comunicaciones utilizando la tecnología FPGA”, Presentado como trabajo final de la carrera en Ingeniería Electrónica, Departamento de Eléctrica y Electrónica, Escuela Politécnica del Ejército, Sangolqui, Ecuador, 2010.

[13] T. Wong, T. Lok." Theory of Digital Communications", University of Hong Kong. Disponible en: http://www.wireless.ece.ufl.edu/twong/notes2.html

[14] Y. Chuang, “Experimental Study of DQPSK in WDM Communication System”, Presentado como trabajo final de la Maestria en Ingeniería Electró-optica, University of Tokyo, Japon, 2010.

[15] Quicknet (2010). π/4 DQPSK or 4-QAM. Estocolmo, Suecia. Recuperado (2014, marzo 23) de http://www.quicknet.se/hdc/ord/info/p4dqpsk.htm

[16] E. Marpanaji et. al, “Experimental Study of DQPSK Modulation on SDR Platform”, Departement of Electronics Engineering, Yogyakarta State University Karangmalang, Indonesia, 2007.

[17] M. Fernández, “Transmisión Digital Paso Banda”, Material de clase, Universidad de Valladolid, España. Disponible en: http://www.lpi.tel.uva.es/lpi/dld/tts/tema8.pdf

[18] F.Xiong, "Digital Modulation Techniques”, Artech House, Reino Unido, 2000.

[19] H. Van, "Detection, Estimation, and Modulation Theory, Radar-Sonar Signal Processing and Gaussian Signals in Noise", Wiley. Signal Processing and Detection.

[20] M. Simon, M-S. Alouini, “Digital Communication over Fading Channels,” Segunda Edicion, Wiley, 2001. ISBN 9780471715238. Disponible en: http://books.google.com.co/books?id=OYrDN0Q6BacC&printsec=frontcover&hl=es&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

[21] J. Martinez y J. Covaleda, “Análisis de desempeño a nivel físico del enlace de subida de LTE”, Presentado como trabajo final de la carrera en Ingeniería Electrónica, Facultad de Ingeniería Electrónica y Telecomunicaciones, Universidad del Cauca, Popayán, Colombia, 2012.

[22] MathWorks (2014). Bit Error Rate (BER). Massachusetts, EE.UU. Recuperado (2014, marzo 27) de http://www.mathworks.com/help/comm/ug/bit-error-rate-ber.html

[23] M.F Zanuy, “Sistemas de Comunicaciones,” Primera Edicion, Marcombo, S.A., 2001. ISBN 9788426713049, p. 171. Disponible en: http://books.google.com.co/books?id=_arH8J1d1FYC

[24] A. Mayo, “PUNTO FIJO VERSUS PUNTO FLOTANTE: principios básicos de funcionamiento ventajas y desventajas en el caso protools”, Vicepresidente AES, Región América Latina. Disponible en: http://www.andresmayo.com/images/Punto%20fijo%20vs%20Punto%20flotante.pdf

[25] Optoplex Corporation (2009). Optical DPSK Demodulator. EE.UU. Recuperado (2014, marzo 27) de http://www.optoplex.com/download/Optoplex%20Datasheets_Integrated%20100G%20Coherent%20Receiver.pdf

[26] National Instrument: Introducción a la Tecnología FPGA: Los Cinco Beneficios Principales, National Instrument, 2011. Disponible en <http://www.ni.com/white-paper/6984/es/> [consulta: Jueves, 27 de marzo de 2014]

[27] P. Jimenez, “Diseño e implementación de un sistema de adquisición, compresión y almacenamiento de imágenes empleando VHDL y FPGAs”, Presentado como trabajo final de la carrera en Ingeniería Electrónica, Facultad de Ingenieria de Eléctrica y Electrónica, Escuela Politécnica Nacional, Quito, Ecuador, 2011.

[28] Xilinx (2008). *System Generator for DSP User Guide*. EE.UU. Recuperado (2014, marzo 27) de http://www.xilinx.com/support/sw_manuals/sysgen_user.pdf

[29] Zeidman B. (2002). *The Universal Design Methodology—taking hardware from conception through production*. EE.UU. Recuperado (2014, marzo 23) de http://m.eet.com/media/1143713/19107-265493.pdf

[30] Astaiza E., Bermúdez H. & Muñoz P. (2007). Simulación De Sistemas De Telecomunicaciones. Colombia: Padilla Bejarano, Ferney.

[31] Xilinx: Project Navigator Overview, ISE, 2009. Disponible en <http://www.xilinx.com/support/documentation/sw_manuals/xilinx11/ise_c_project_navigator_overview.htm> [consulta: Jueves, 27 de marzo de 2014]

[32] MathWorks (2014). Matlab. Massachusetts, EE.UU. Recuperado (2014, marzo 27) de http://www.mathworks.com/products/matlab/

[33] MathWorks (2014). Simulink. Massachusetts, EE.UU. Recuperado (2014, marzo 27) de http://www.mathworks.com/products/simulink/

[34] MathWorks (2014). Bertool. Massachusetts, EE.UU. Recuperado (2014, marzo 27) de http://www.mathworks.com/help/comm/ref/bertool.html

[35] J. Proakis. *Digital Communications*. Cuarta Edición, McGraw-Hill Higher Education, 2001. ISBN 0072321113, 9780072321111, pp 257-260.

[36] J. Muñoz y J. Zemanate, “Análisis del desempeño de un sistema de comunicaciones con modulación 16/64 QAM basado en hardware reconfigurable”, Presentado como trabajo final de la carrera en Ingeniería Electrónica y Telecomunicaciones, Facultad de Ingeniería Electrónica y Telecomunicaciones, Universidad del Cauca, Colombia, 2014.

[37] Xilinx (2008). *System Generator for DSP Reference Guide*. EE.UU. Recuperado (2014, marzo 27) de http://www.xilinx.com/support/sw_manuals/sysgen_ref.pdf

[38] R. Das,"FPGA Implementation of Digital Modulation Schemes: BPSK and QPSK Using VHDL" Revista IJECT [Online], Volumen 5 Issue 2 (March 2014)Dept. of ECE, Sir J. C. Bose School of Engineering, West Bengal, India. Disponible en: http://www.iject.org/vol5/spl2/ec1136.pdf

[39] E. Arias y D. Sánchez, “Desarrollo de bloques hardware para la transmisión y recepción de señales digitales moduladas en fase diferencial”, Presentado como trabajo final de la carrera en Ingeniería Electrónica, Facultad de Ingeniería, Universidad del Quindío, Colombia, 2011.

[40] La Trobe University (2010*). Lecture 7 - Modulation: Making the Message Fit the Medium*. Melbourne, Australia. Recuperado (2014, marzo 23) de http://ironbark.xtelco.com.au/subjects/DC/lectures/7/