UNIVERSIDAD POLITECNICA DE VALENCIA

ESCUELA POLITECNICA SUPERIOR DE GANDIA

I.T. Telecomunicación (Sist. de Telecomunicación)

"Diseño e Implementación en FPGA del Método Recursivo Gram-Schmidt"

TRABAJO FINAL DE CARRERA

Autor/es:
Franklin Salazar Logroño

Director/es:
Dña. Mª Asunción Pérez Pacual

GANDIA, 2011

MEMORIA DEL PROYECTO

INDICE DE FIGURAS

INDICE DE TABLAS

CAPITULO_1

1.1 RESUMEN

Este proyecto presenta un nuevo método que es la implementación de un algoritmo de ortogonalización que utiliza el teorema de GRAM-SCHMIDT para la compensación del desbalanceo que se produce en las transmisiones en este caso QAM debido a las diferentes interferencias provocadas por los componentes del sistema tales como convertidores (A/D) o (D/A), interferencia del canal (ruido)etc.
La realización corresponderá con la implementación y simulación del algoritmo en SIMULINK.
Se va a diseñar en el dispositivo FPGA-*Virtex*-4 da *Xilinx*, utilizando, para eso un programa llamado *Xilinx System Generator*. Finalmente, presentaremos los diferentes resultados obtenidos y la comparación con otras implementaciones.

1.-2 OBJETIVOS.

Gracias al conocimiento adquirido tratar de solucionar el problema de desbalanceo que se produce por interferencias y debido a los desfases provocados por los componentes electrónicos que constituyen nuestro sistema.
Como también poder afianzar los conceptos adquiridos durante parte de la carrera y ponerlos en práctica a la hora de interpretar y solucionar este proyecto.
Dar a conocer las ventajas que ofrecen las herramientas simulink y system generator a la hora de poder simular y comprobar el perfecto funcionamiento como también hacernos una referencia real de lo que se produce en las transmisiones y recepciones de señales.

1.-3 INTRODUCION

El propósito de un sistema de comunicaciones es entregar un mensaje de información entre usuarios que estén separados, en un formato que se pueda ser entendido por los dos. Por esta razón un transmisor modifica el mensaje para poder transmitir de una forma apropiada atreves del canal. Esta modificación realizada es conocida como modulación, el receptor recoge la señal original y la representa mediante un método llamado demodulación que es el procedimiento inverso al de modulación en el transmisor.

Debido a que el canal introduce interferencias o ruido, o a los dispositivos electrónicos utilizados la señal recibida es la señal original pero con degradaciones.
En la grafica podemos conocer las fases de un sistema de comunicaciones digital.

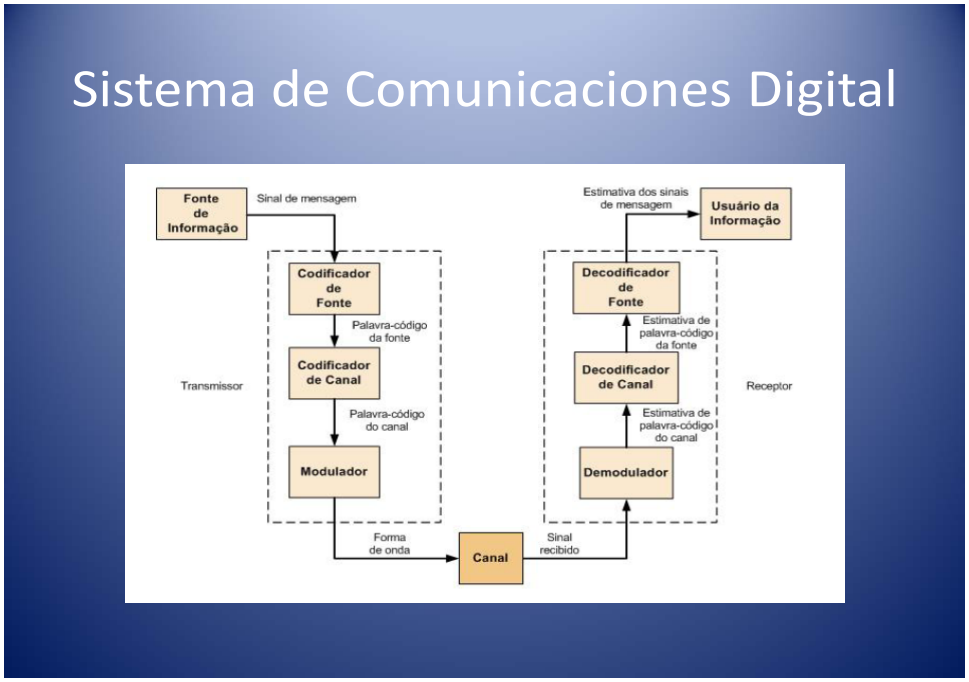

Fig. 1.1.- Sistema de Comunicaciones Digital.

• Codificador - Decodificador de fuente;
• Codificador - Decodificador de canal;
• Modulador - Demodulador.

Un codificador fuente elimina información redundante del mensaje y se responsabiliza de la utilización eficiente del canal ,la secuencia de símbolos resultantes llamada código de fuente es procesada por el codificador de canal el cual igualmente produce una secuencia de símbolos llamada código de canal el cual es mayor que el anterior debido a la redundancia introducida en su construcción .Finalmente el modulador representa cada símbolo o código de canal por su símbolo analógico correspondiente la secuencia de símbolos producida por el modulador es llamada forma de onda , que es la apropiada para transmitir por el canal.
En el receptor a la salida del canal (señal recibida) es procesada de forma inversa que en el transmisor, reconstruyendo así una versión reconocible de la señal original.

Una transmisión de datos requiere el uso de un canal con un ancho de banda suficientemente grande para acomodar el contenido frecuencial de flujo de datos.

1.3.1 DESBALANCEO DE LA RAMA I/Q.

El crecimiento de la demanda de servicios de comunicaciones y el límite en el espectro de frecuencias disponibles ha forzado el uso de modulaciones más eficientes espectralmente. La mayoría de estas modulaciones eficientes en ancho de banda son modulaciones de envolvente no constante. Esto unido a las alinealidades de la cadena transmisora (principalmente al amplificador de potencia) producen una expansión del espectro de la señal transmitida en los canales adyacentes. Este efecto se denomina Interferencia en el Canal Adyacente (ICA). Algunos sistemas de comunicaciones, son muy restrictivos en la emisión de espúreos en el canal adyacente. La principal desventaja de este método es su elevada sensibilidad a los desbalances de ganancia y fase que se producen ambas ramas .Pero el método que se presenta se distingue por el uso de técnicas adaptativas de procesado de señal. Dicho método se realiza en banda base y completamente digital. Se han realizado rigurosas simulaciones evaluando diversas alternativas.

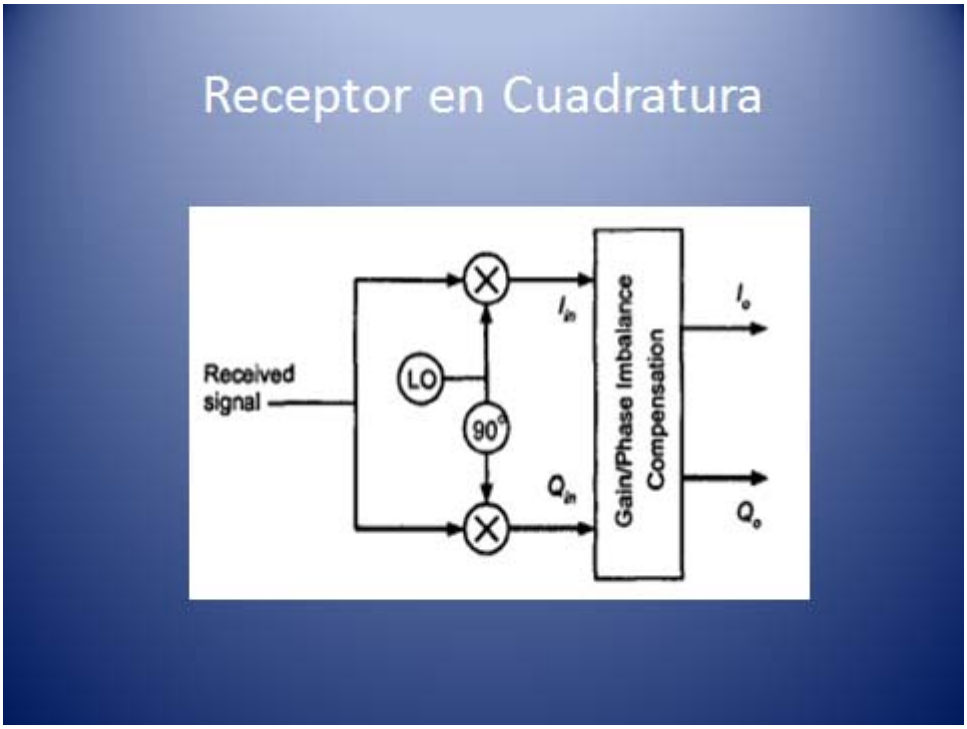

Fig. 1.2 Receptor en Cuadratura

Como podemos ver en la grafica el desbalanceo de ganancia y fase que se produce entre canal I y el canal Q se encuentra justo en el bloque para la

fabricación y representación de la señal, estos desbalanceos provocan generalmente una degradación en los sistemas de comunicaciones.

1.3.2 METODOS ADAPTATIVOS PARA LA CORRECCION DEL DESBALANCEO.

Como lo que se pretende es compensar el desbalanceo que se produce entre ramas y obtener el menor error posible a la salida de la señal para poder obtener una señal idéntica o casi idéntica a la original la idea es utilizar métodos adaptativos.

Debido al gran desarrollo de los sistemas de comunicaciones, los problemas que se presentan aumentan en la misma medida, lo que incrementa la complejidad de diseño de los sistemas para evitar el menor número de errores en la transmisión de información. Por ejemplo, cancelación de eco, ruido aditivo, interferencia íntersímbolo, son problemas considerables que se presentan a menudo en sistemas de comunicaciones digitales; su solución está en el uso de filtros adaptivos. Los filtros adaptivos son sistemas que adaptan sus parámetros de acuerdo a las variables que se presentan en su entorno y tienen una estructura como la que se muestra en la Fig.1. 3, en donde se puede ver, que son sistemas de cuatro terminales con una señal de entrada *x*, la señal deseada a la cual se quiere aproximar la respuesta del filtro *d*, además genera una señal de salida *y* con una señal de error *e* igual a la diferencia entre la señal deseada y de salida del filtro adaptivo. La señal de error se emplea como referencia para adaptar los parámetros del filtro (Carusone y Johns, 2000).

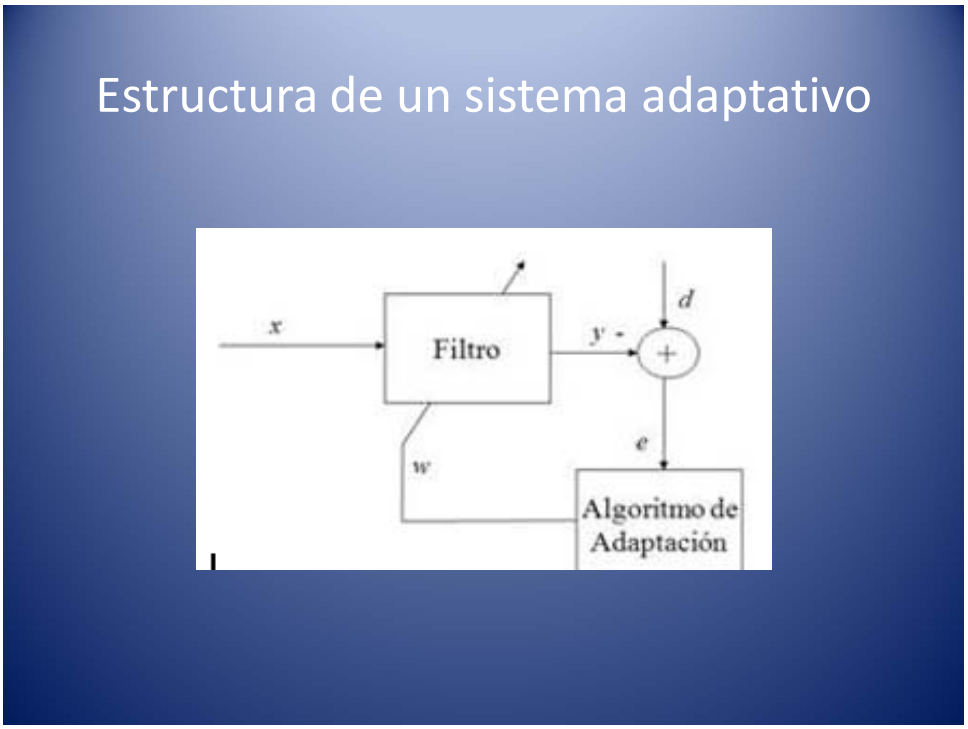

Fig.1.3.- Método Adaptativo

La eficiencia de los filtros adaptivos depende principalmente de la técnica de diseño utilizada y el algoritmo de adaptación. Los filtros adaptivos pueden ser diseños analógicos (Carusone y Johns, 2000; Pérez et al., 2001; Lee y Razavi, 2001), digitales o mixtos los cuales presentan sus ventajas y desventajas, por ejemplo, los filtros analógicos son de bajo consumo de potencia y respuesta rápida, pero presentan problemas de desbalanceo (*offset*), el cual afecta el funcionamiento del algoritmo de adaptación (Shoval et al., 1995). Los filtros digitales son libres del desbalanceo y ofrecen una respuesta de mayor precisión, pero son de gran complejidad debido a las operaciones de punto flotante. Los diseños mixtos (Bhupendra et al., 1979a, 1979b; Figueroa et al., 2004), presentan problemas de desacoplamiento entre los bloques analógicos y digitales. Estas técnicas realizan el procesamiento de la señal en el dominio del tiempo, pero también existen filtros adaptivos que trabajan en el dominio de la frecuencia (Jackson, 1998; Bogucka y Wesolowski, 2000; Van Acker et al., 2001), los cuales realizan filtrado adaptivo en sub-bandas con buena respuesta, pero el diseño de estos filtros requieren de procesadores digitales de señales. Otras propuestas realizan una modificación directa a la estructura del filtro para aumentar la eficiencia en la convolución (Chen et al., 1996).

Los algoritmos de adaptación más conocidos, son el algoritmo de mínimos cuadrados recursivos (*RLS – Recursive Least Square*), el algoritmo de mínimos cuadrados (*LMS – Least Mean Square*) y el algoritmo de Gram

Schmidt, en donde el algoritmo RLS ofrece mayor velocidad de convergencia con respecto al algoritmo LMS, pero en cuanto complejidad computacional, el algoritmo LMS tiene la ventaja. Por su simplicidad computacional, el algoritmo LMS es el que tiene mayor uso en el diseño e implementación de filtros adaptivos digitales (Avalos et al., 2007).

1.3.2.1-EL ALGORITMO LMS.

Utiliza una aproximación estocástica del vector gradiente como criterio de minimización de error medio cuadrático para obtener los coeficientes de este modo LMS se vuelve excepcionalmente simple desde el punto de vista computacional, destacándose también por su robustez.
Para analizar la formulación matemática del algoritmo LMS inicialmente definimos los siguientes vectores complejos y escalares:

*Un vector de señal de entrada

u(n)=[u(n) u(n-1) ……..u(n-N+1)]T,

*Un vector de coeficientes del filtro:

w(n)=[w0 w1 …….w_{n-1}]T,

*Una señal de salida y(n), dada por:

y(n)=w^n *u(n),

*La señal de error:

e(n)=d(n)-y(n),

Donde d(n) es la señal de referencia.

*Actualización de coeficientes (pesos):

$$\mathbf{w}[n+1] = \mathbf{w}[n] + 2\mu\, e[n]\mathbf{x}[n]$$

Considerando las componentes de fase (I) y cuadratura (Q)

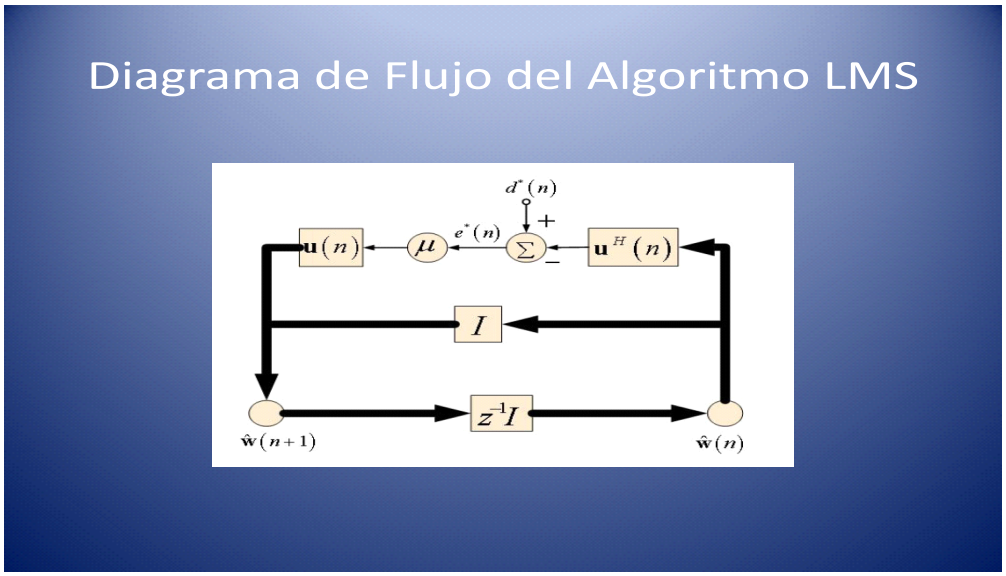

Fig.1.4.- Diagrama de flujo del algoritmo LMS

Modelo canónico del Algoritmo LMS complejo:

Un vector de señal de entrada, una señal deseada, un vector de coeficientes del filtro y una señal de error que pueden ser reescritos:

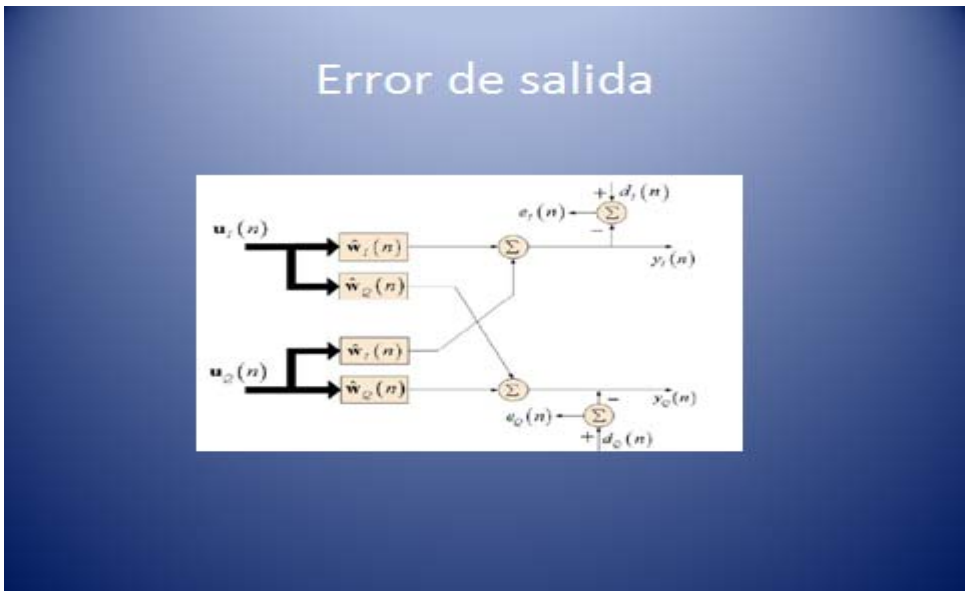

Fig.1.5.- Proceso de cálculo de error

$$e(n) = e_I(n) + je_Q(n)$$

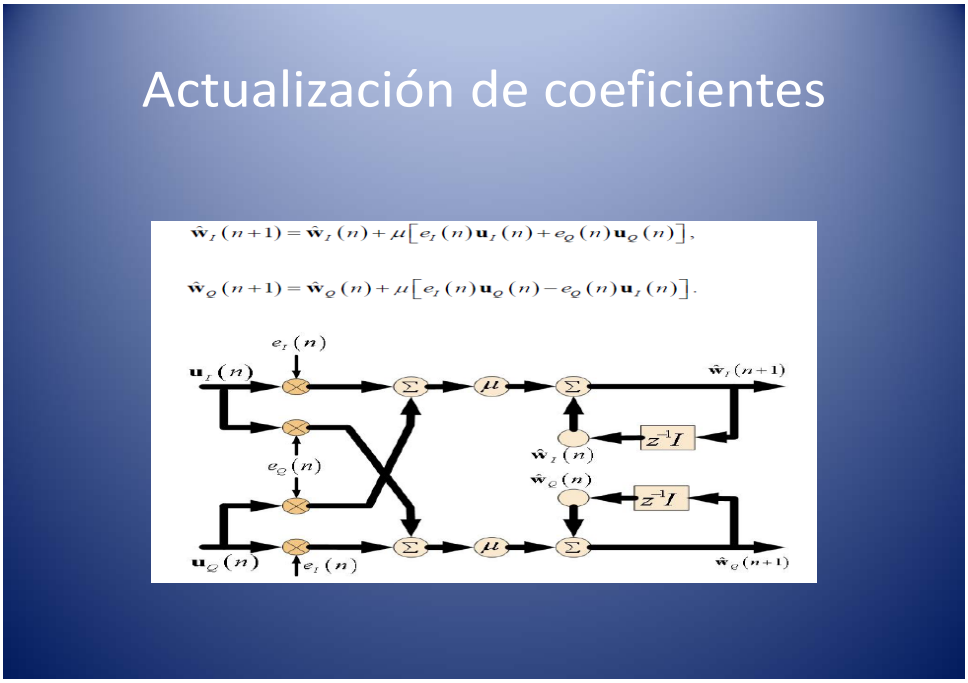

Fig.1.6.- Diagrama y formulas de actualización de coeficientes utilizando LMS.

1.3.2.2 ALGORITMO ECLMS.

En esta sección se presentan los resultados obtenidos, con el propósito de realizar un análisis comparativo de la modificación propuesta. Los resultados que se muestran, realizan la comparación con respecto al algoritmo LMS convencional y el algoritmo LMS sobreadaptado cuando el filtro adaptivo lleva a cabo la predicción lineal y la identificación de sistemas, además se realiza el análisis de comportamiento del número de bits de codificación y la resolución del codificador.

Predictor lineal

La predicción lineal es una técnica de estimación espectral, la cual se emplea para modelar procesos aleatorios correlacionados, con el objetivo de encontrar la representación paramétrica de estos procesos. Los resultados de la comparación del algoritmo LMS contra el algoritmo modificado, se muestra en la Fig. 1.7

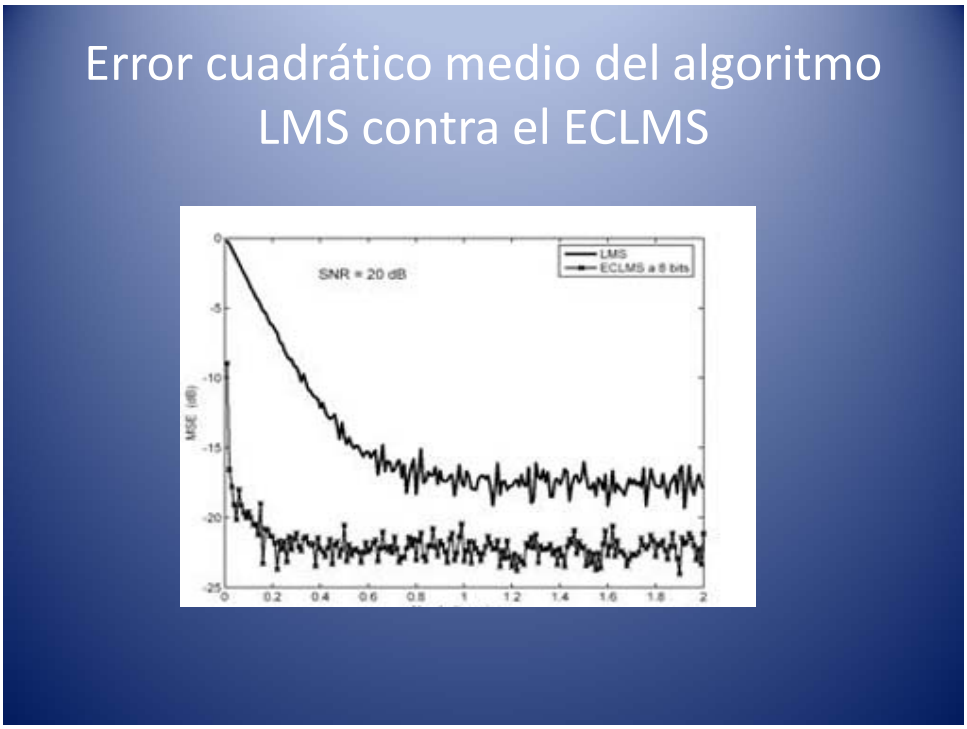

Fig.1.7.- Error cuadrático medio

En los resultados mostrados en la Fig. 1.7, se puede observar que el error cuadrático medio (*MSE – Mean Square Error*) para el algoritmo ECLMS, converge mucho más rápido con respecto al algoritmo LMS convencional, además de que no se empeora el error cuadrático mínimo en ECLMS. El aumento de la velocidad de convergencia con ECLMS se debe a que de manera indirecta se está afectando al factor μ, debido a que inicialmente el error es grande y por lo tanto el error codificado corresponde a un valor grande, pero como avanza el proceso de adaptación, el error va disminuyendo hasta tener un comportamiento similar a la variante de signo del algoritmo LMS, debido a que el error codificado tendrá valores de uno o cero.

1.3.2.3.- ALGORITMO LMS-DD(Least Square-Decision Directed)

Utilice los métodos tradicionales de operación de un ecualizador adaptativo pero en modo decisión directa (DD) como se muestra en la fig1. 8. Durante la secuencia de adaptación tenemos, una secuencia conocida y transmitida en una versión sincronizada de la (d(n)) y generada por el receptor. Una señal de error e(n)=d(n)-y(n) y formada también por un ecualizador adaptativo.

Los coeficientes de ecualización son así mismo ajustados de acuerdo con el algoritmo LMS utilizando un criterio de minimización de error cuadrático medio. Cuando un proceso ha terminado el ecualizador ha conmutado por un segundo modo de operación o modo de decisión directa (DD).

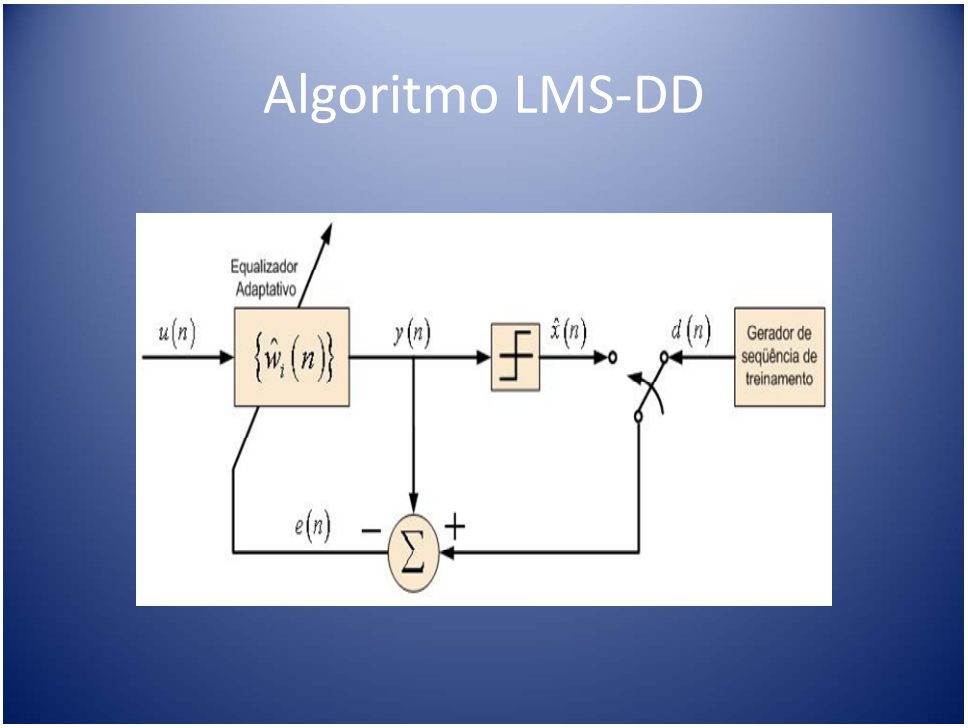

Fig.1.8.- Diseño de algoritmo adaptativo LMS-DD

Este algoritmo se creó pensando en utilizar la salida del propio ecualizador para rastrear las variables del canal, con ello se mejoraba los valores de los coeficientes si fuese necesario. El algoritmo LMS-DD es muy similar al algoritmo LMS mas solo se diferencia en la obtención de la señal deseada que pasa a ser una decisión realizada sobre una salida del propio ecualizador, así la función de decisión queda:

$$e(n)= \ dec(y(n))-y(n).$$

De esta forma el error pasa a ser una función no lineal de coeficientes del filtro, es decir deja de ser cuadrática. Esa no linealidad impuesta por la función acarrea una formación de mínimos locales indeseados.

Tomando los pasos de μ de valor fijo se muestra que el algoritmo LMS-DD converge para una solución óptima de este caso descrito por Wiener.

Caso contrario como hemos mencionado antes este algoritmo presentara convergencia para otros valores mínimos no deseados.

1.3.2.4.-ALGORITMO DE ORTOGONALIDAD DE GRAM SCHMIDT.

Ortogonalidad:

En matemáticas, el termino ortogonalidad es una generalización de la noción geométrica de perpendicularidad. En el espacio euclideo convencional el término ortogonal y el término perpendicular son sinónimos. Sin embargo, en espacios de dimensión finita y en geométricas no euclideas el concepto de ortogonalidad generaliza al de perpendicularidad.

Existe un procedimiento que permite generar un conjunto de señales ortogonales conocido bajo el nombre de ortogonalizacion de Gram – Schmidt.

Consiste en definir una señal y generar a partir de ellas señales ortogonales.

- Suponga que x1 y x2 son dos vectores independientes.
- Se desea generar un conjunto de vectores ortogonales y1 y y2 a partir de x1 y x2. De la figura se desprende que

$$x_2 = y_2 + c_{12}x_1 = y_2 + c_{12}y_1$$

$$c_{12} = x_1.x_2/\,|x1|^2 = y_1.x_2\,/\;\;|y1|^2$$

$$y_2 = x_2 - c_{12}y_1$$

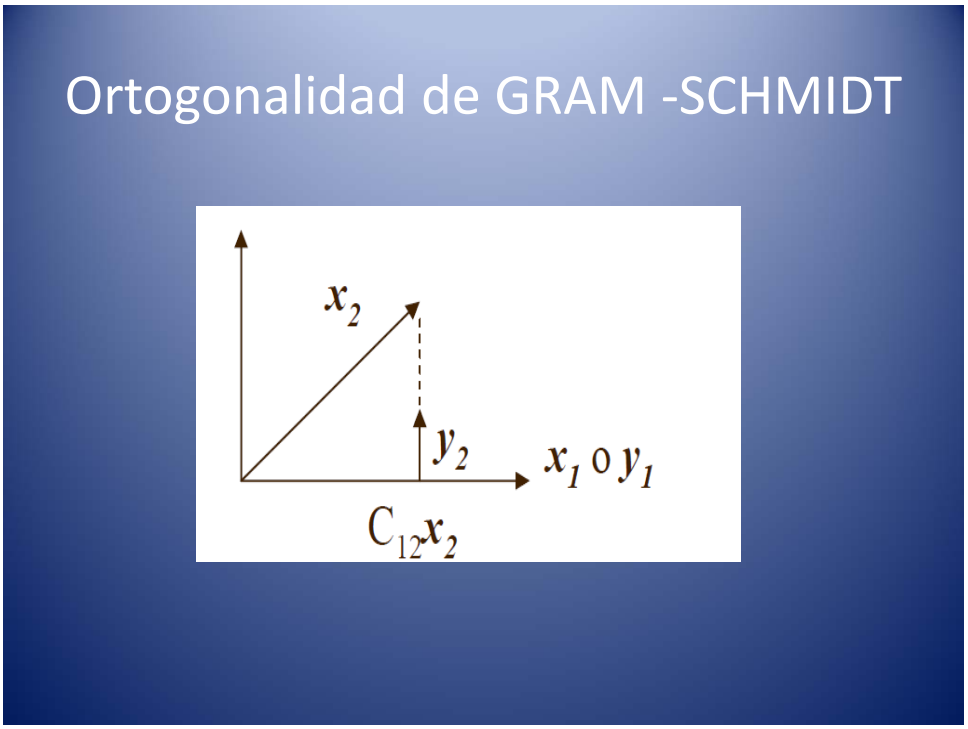

Fig.1. 9.- Principio de ortogonalidad de Gram schmidt

CAPÍTULO_2

2.1 METODO DE GRAM SCHMIDT APLICADO A MODULACION QAM.

2.1.1 MODULACION QAM.

2.1.1.1 INTRODUCCIÓN_ QAM.

La Modulación de Amplitud en Cuadratura o QAM es una modulación digital en la que el mensaje está contenido tanto en la amplitud como en la fase de la señal transmitida. Se basa en la transmisión de dos mensajes independientes por un único camino. Esto se consigue modulando una misma portadora, desfasada 90º entre uno y otro mensaje. Esto supone la formación de dos canales ortogonales en el mismo ancho de banda, con lo cual se mejora en eficiencia de ancho de banda que se consigue con esta modulación.
La importancia de este sistema de modulación se debe a la gran cantidad de aplicaciones asociadas a ella:

• Es empleada por módems para velocidades superiores a los 2400 bps (por ejemplo V.22 bis y V.32).

• Es la modulación empleada en multitud de sistemas de transmisión de televisión, microondas, satélite...

• Es la base de la modulación TCM (Trellis Coded Modulation), que consigue velocidades de transmisión muy elevadas combinando la modulación con la codificación de canal.

• Es la base de los módems ADSL (*Asymmetric Digital Suscriber Line*) que trabajan en el bucle de abonado, a frecuencias situadas entre 24KHz y 1104KHz, pudiendo obtener velocidades de hasta9Mbps, modulando en QAM diferentes portadoras.

2.1.1.2 TRANSMISOR QAM BÁSICO.

El esquema de un transmisor en QAM básico se muestra a continuación.

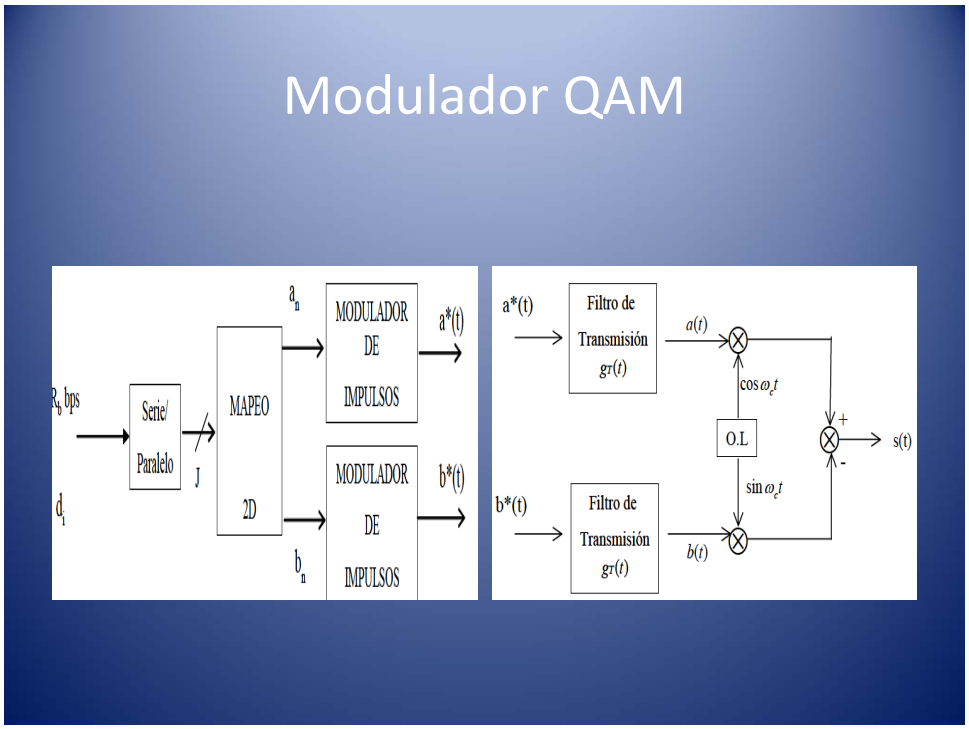

Fig.2.1 Diseño de un modulador QAM

Los datos *de* serie de entrada, generados a velocidad *Rb* bps se agrupan mediante un conversor serie/paralelo, formando palabras de J bits que pasarán al módulo de mapeo de estas palabras. Este módulo se encarga de seleccionar un símbolo de entre los M=2J posibles símbolos, ubicados sobre un espacio bidimensional. A la salida, los símbolos se producen por tanto a una velocidad de

Fs=Rd/J símbolos por segundo o baudios.

Los símbolos a transmitir son números complejos que representaremos como $k\ k\ k\ c = a + jb$. Así, el alfabeto lo forman el conjunto de números complejos que podremos transmitir. Este alfabeto se puede representar en el plano complejo, formando la constelación de la modulación. En la siguiente gráfica se presentan diferentes constelaciones posibles.

Fig.2.2 Constelaciones para modulaciones QAM

A continuación, los símbolos se introducen en los moduladores de impulsos, uno para cada componente, obteniendo las señales:

$a^*(t) = \Sigma ak\ \delta\ (t - kT)$

$b^*(t) = \Sigma bk\ \ \delta(t - kT)$

Estas dos señales atraviesan los filtros de transmisión:

$a(t) = \Sigma ak\ gT\ (t - kT)$

$b(t) = \Sigma bk\ gT\ (t - kT)$

$g_T(t)$ es el filtro de transmisión y será de tipo paso bajo. Sobre este filtro aplica todo lo dicho para los filtros de transmisión en el capítulo correspondiente a la transmisión en banda base. En una implementación discreta, los filtros actúan de filtros interpoladores, produciendo L muestras por cada símbolo de entrada, de forma que la frecuencia de trabajo de los filtros será de $L{\cdot}fs$.

La señal QAM se obtiene modulando en DBL estas señales:

$s(t) = a(t)\cos wct - b(t)\,sen\,wct$

Así, $a(t)$ es la componente en fase de la señal QAM y $b(t)$ la componente en cuadratura. El equivalente paso bajo de la señal QAM, tomando como frecuencia de referencia fc será:

$\tilde{s}\,(t) = \; a(t) + jb(t) = \sum_k ak\, gT(t - kT) + j \sum_K bkgT(t - kT) =$
$= \sum_k (ak + jbk) gT(t - kT) = \sum_k CkgT(t - kT)$

La señal analítica: $s^+(t) = \sum CkgT(t - kT)\; e^{jwct}$

En donde la señal QAM es $s(t) = \Re\varepsilon\{s^+\,(t)\}$

De forma esquemática:

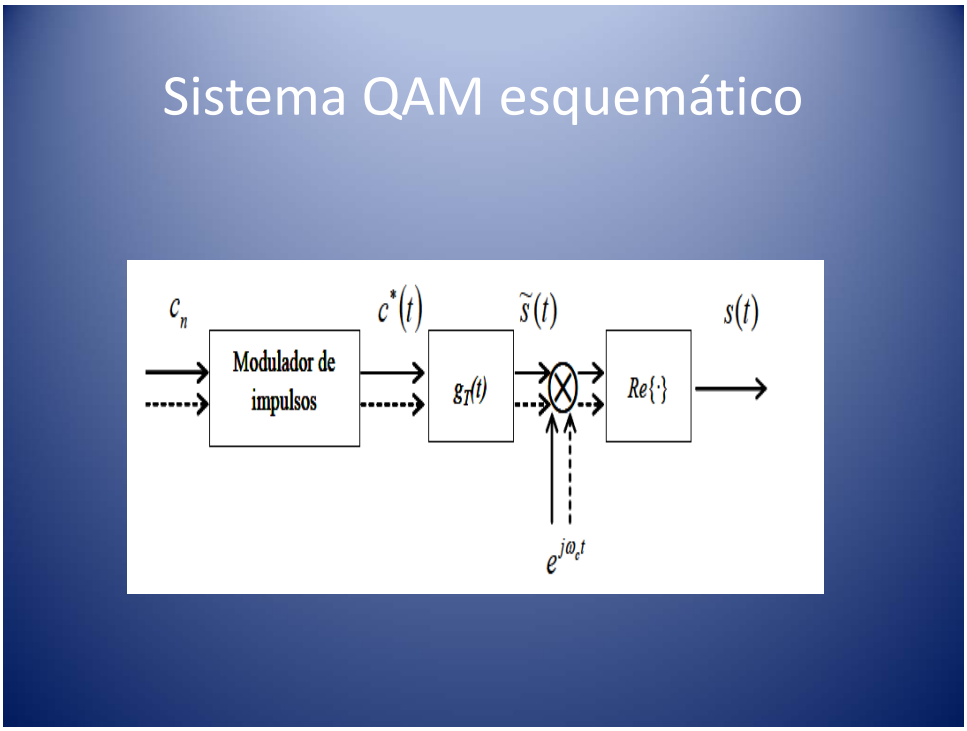

Fig.2.3 Sistema QAM esquemático.

Como podemos observar, en el esquema de modulación propuesto se obtiene primero la señal paso bajo que se modula más tarde en DBL.

2.1.1.3 RECEPTOR QAM: DESCRIPCIÓN GENERAL.

Un receptor QAM sigue el esquema que se presenta en la siguiente figura. Como puede observarse, el esquema del receptor es considerablemente más complejo que el del transmisor.

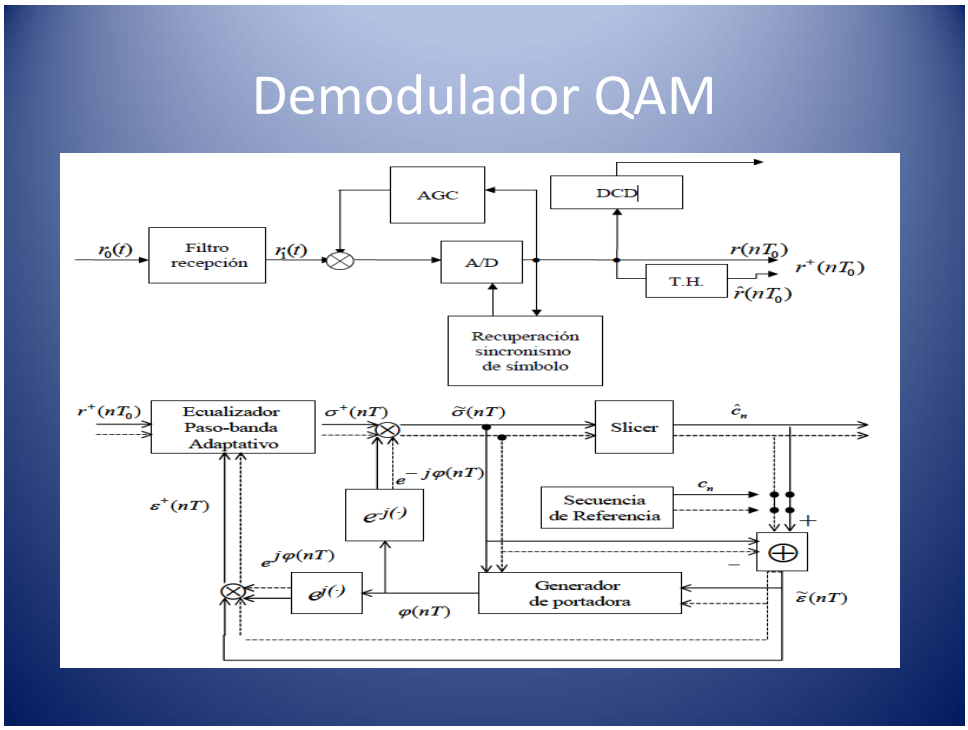

Fig.2.4 Sistema demodulador QAM.

ro(t) , señal de entrada al receptor es la señal QAM transmitida, distorsionada por el canal y con ruido añadido. La función principal del filtro de recepción: es eliminar ruido fuera de banda, y también, en combinación con el filtro de transmisión, conformar el pulso recibido para que se produzca IES sobre un canal ideal. Debido a que las señales pueden llegar muy atenuadas, la salida del filtro de recepción se escala por el Control Automático de Ganancia (CAG) para incrementar su amplitud y así utilizar todo el rango dinámico de los convertidores.

El muestreo en los convertidores se realiza normalmente a una frecuencia superior a la frecuencia de símbolo:

Fo=1/To =no*(1/T)=no*fs

que además será de al menos el doble de la máxima frecuencia contenida en la señal QAM. Las muestras obtenidas serán utilizadas por el módulo de CAG para calcular el factor adecuado.

También se utilizan en el DCD (*Data Carrier Detect*) para determinar si una señal es señal o es sólo ruido.

Los instantes de muestreo adecuados se determinarán en el recuperador de sincronismo de símbolo a partir de las muestras obtenidas.

El último módulo forma la señal analítica *r+ (nTo)* . Este subsistema que forma esta señal analítica, se conoce como *Phase Splitter* .

Un canal real no tiene respuesta plana y retardo constante de envolvente (como tendría un canal ideal) y esto causa interferencia entre símbolos en la señal recibida. El ecualizador adaptativo paso banda compensa la

respuesta del canal para minimizar la IES. Este filtro debe ser adaptativo ya que a priori no se conoce la respuesta frecuencial del canal.

DETECCIÓN DE PORTADORA DE DATOS (*DATA CARRIER DETECT*).

La función del bloque de detección de portadora es detectar cuándo tenemos señal presente y cuándo no, por lo general para indicárselo a un módulo de recepción de datos. Se trata de un módulo muy simple, pero de gran importancia, ya que permitirá validar o rechazar los datos recibidos por módulos situados en niveles superiores.
La mayoría de los métodos deciden la existencia de señal basándose en la energía de la señal recibida.

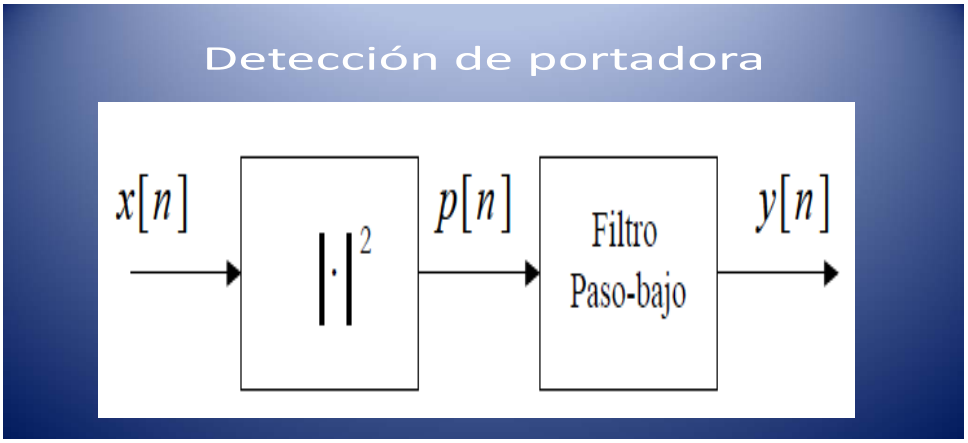

Fig.2.5.- Detección de la portadora en modulación QAM

RECUPERACIÓN DE PORTADORA.

Para simplificar el proceso, y también porque los dos problemas pueden tratarse de forma independiente, supondremos en este apartado que la frecuencia de símbolo exacta es conocida.
El problema que presentamos en este apartado, es la recuperación de la frecuencia y fase de la portadora.
Consideramos la señal analítica:

$$S^{\pm}(t) = \sum Cm * g(t - mT).e^{j(Wct+\theta(t)}$$

en donde g(t)=gT(t)*c(t)*gR(t) es la respuesta global del canal, y los símbolos complejos Cm=am + jbm.
Supongamos que de modulamos esta señal con

$$e^{-j(Wct+\theta(t))}$$

Entonces el valor del símbolo recibido es:

$$q_k = s^+(t).e^{-j(wct+\emptyset(kT))}.\sum_m Cmg\big((k-m)T\big) \qquad t = kT$$

Si no hay IES ni ruido: $q_k = e^{j(\theta_k - \emptyset_k)}.ck$ es decir, un error en la fase de la portadora utilizada en la detección, girará la constelación obtenida en ese mismo ángulo. Si el error cometido fuera un error de frecuencia, $\theta_k - \emptyset_k = w_e kT$ entonces observaríamos una constelación girando con velocidad ω*e* rad/s. Esto puede perjudicar enormemente la detección si no se corrige.

2.1.2 METODO GRAM SCHMIDT APLICADO A QAM.

Se trata de un algoritmo de ortogonalizacion que utiliza el teorema de Gram-Schmidt el cual transforma un conjunto de variables sin ortogonalidad en un conjunto de variables ortogonales.

ECUACIONES DE LA IMPLEMENTACION RECURSIVA DEL TEOREMA DE GRAM SCHMIDT.

$Vn(t) = Xn(t) - \sum_{i=1}^{n-1} \gamma ni * Ui(t)$
$Un(t) = \frac{Vn(t)}{\sqrt{\gamma^2}_n}$

para n=1,……..N.

función de correlación $\gamma ni(t) = (1-\rho).\gamma ni(t-1) + \rho.Xn(t).ui(t-1)$
i=1…….n-1:
varianza $\gamma n^2(t) = (1-\rho).\gamma^2(t-1) + \rho.Vn^2(t-1)$

Para el cálculo de error cuadrático medio podemos dar un valor de

$$\rho = 0.01 \qquad o \qquad \rho = 0.001$$

De este valor depende si la convergencia es más rápida o en saltos más grandes o si por el contrario la convergencia tarda más tiempo.

MODELO DEL DESBALANCEO

$$c(n) = I(n).\cos(wn) - \alpha Q(n).sen(wn + \emptyset)$$

Desarrollo trigonométrico:

$$c(n) = I(n).\cos(wn) - \alpha Q(n).[sen(wn).\cos(\emptyset) + cos(wn).sen(\emptyset)] =$$

$$= \big(I(n) - \alpha Q(n)\big).\cos(wn) - \alpha Q(n)\cos(\emptyset)\,.sen(wn).$$

Sin embargo cálculos de división o de raíz cuadrada pueden ser bastante problemáticos y requieren muchos ciclos de reloj o puertas lógicas para poder obtener la salida.

Por lo cual para el cálculo de este algoritmo es importante la no utilización de estas operaciones y tratar en su lugar de implementar con sumas y multiplicaciones que es muy aceptable y eficiente en tiempo real ya que se logra tener una alta velocidad en la implementación de DSP.

ALGORITMO SIN DIVISION NI RAIZ CUADRADA.

Finalmente las ecuaciones para la implementación del algoritmo deseado:

$$Sn(t) = \alpha.sen(t-1)\big(1 - \beta Un^2(t-1)\big)$$

$$\alpha = {1}\Big/{\sqrt{1-\rho}} \qquad \beta = \frac{\rho}{2(1-\rho)}$$

$$\gamma ni(t) = \gamma ni(t-1) + \rho.\big(Xn(t-1).Ui(t-1)\big) - \gamma ni(t-1))$$

$$Vn(t) = Xn(t) - \sum_{i=1}^{n-1} \gamma ni.Ui(t)$$

$$Un(t) = Vn(t).Sn(t)$$

Donde:

X1(t) = Iin(t) U1(t) = Io(t)

X2(t) = Qin(t) U2(t) = Qo(t)

CAPITULO_3

3.1 IMPLEMENTACION DEL ALGORITMO GRAM-SCHMIDT EN FPGA

3.1.1 IMPLEMENTACION CON SIMULINK.

Es una herramienta matemática de matlab que permite una simulación de la modulación que pretendemos realizar prácticamente, trabaja directamente con XILINX que a su vez trabaja con la herramienta system generator que es la que utilizamos para la implementación en a FPGA.
Estas herramientas permiten la integración del proceso de diseño e implementación simulando y programando el dispositivo. Facilitan la elaboración de esquemas y el mejoran el tiempo de ejecución del proyecto. Es recomendable trabajar con estas herramientas ya que se basan en el uso de principios matemáticos similares a los utilizados en el procesamiento digital de señales.

3.1.1.1 DISEÑO IMPLENTADO EN SIMULINK

Para la implementación del algoritmo adaptativo de gran Schmidt hemos utilizado la modulación 16QAM.

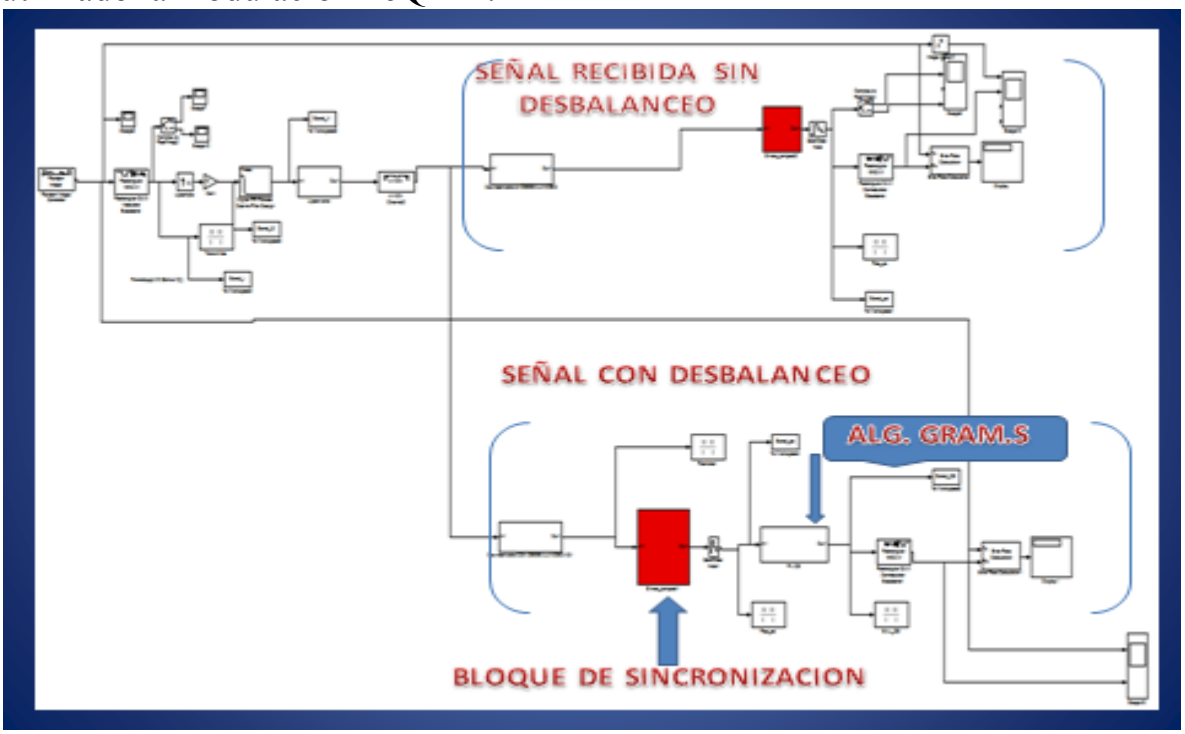

Fig.3.1 Implementación en simulink del modulador y demodulador 16QAM.

Como se observa en la figura tenemos el bloque completo de un modulador –demodulador QAM en el cual se puede apreciar tanto la señal sin compensar el desbalanceo y el receptor que si utiliza el algoritmo adaptativo de Gram Schmidt para compensar el desbalanceo.

Las 2 señales que se reciben tienen los mismos elementos y señales como también valores tan solo varía el bloque del algoritmo implementado que esta formulado en base a las ecuaciones resueltas anteriormente son:

La señal transmitida por el modulador 16QAM es la que se puede observar en la fig.17.Esta modulación utiliza un alfabeto de 16 símbolos. Por lo tanto, usa palabras de cuatro bits (J=4). La constelación es la siguiente:
Vemos que la señal está perfectamente definida y no tenemos desfase ni descompensación en amplitud entre los 16 símbolos.

Fig. 3.2 Señal transmitida 16-QAM

Los valores más determinantes y a tener en cuenta en el sistema de modulación y demodulación es:

EbNo: Es el valor que indica el ruido que introduce el canal en el sistema

ρ : Es el parámetro que me indica la rápida o lenta convergencia para hallar el error cuadrático medio y es un valor pequeño cercano a cero.

Simulación con:

EbNo=1000.

ρ =0.001

Realizamos la simulación con un valor pequeño de ruido y con un valor pequeño de convergencia para obtener los mejores resultados aunque se tarde más en la simulación, aunque en posteriores capítulos variaremos los valores para establecer una comparación y la obtención del valor más indicado para esta implementación.

RECEPCION DE LA SEÑAL SIN DESBALANCEO

En la figura 3.3 podemos observar que la señal que estamos recibiendo por la rama sin desbalanceo es parecida a la original con pequeñas interferencias debidas al ruido que introduce el canal.
Es evidente que lo que estamos recibiendo son los 16 símbolos transmitidos con error de amplitud en algunas muestras.

Fig.3.3 Recepción de la señal sin desbalanceo.

SEÑAL RECIBIDA CON DESBALANCEO Y ANTES DE BLOQUE DE SINCRONIZACION.

La señal que tenemos en la fig.3.4 corresponde con la rama que si tenemos el desbalanceo es decir es donde tenemos el problema real que sufren los sistemas y aplicaciones normalmente y lo que estamos recibiendo es una señal entrada +interferencia que provoca IES y que además el desfase provocado causa que no se pueda distinguir los símbolos transmitidos, podemos apreciar los grabes problemas que se tienen en la descompensación de fase y ganancia que es lo que se espera resolver o minimizar con el algoritmo de G.S .
Lo que se ve en la figura son un sinnúmero de muestras cada una con un valor de amplitud diferente e imposible de distinguir a simple vista, los símbolos y amplitud con que fueron transmitidos es imposible de apreciar y peor detectar sin cometer un error es decir que lo que estamos recibiendo es nuestra señal pero con IES (interferencia entre símbolos).

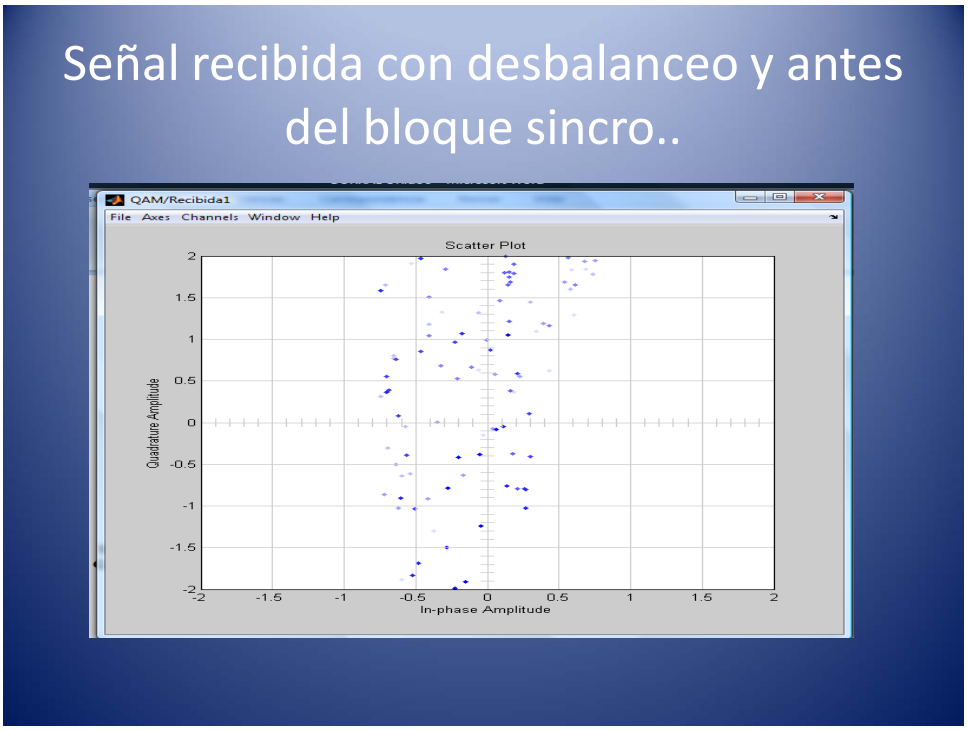

Fig.3.4 Señal recibida

SEÑAL RECIBIDA ANTES DEL BLOQUE G.S
Con la ayuda de un bloque de sincronización que :
• Es la etapa más crítica de un sistema de transmisión.
• El receptor debe estimar:
– Los instantes de símbolo
– La frecuencia de la portadora (en transmisión paso banda)
– La fase de la portadora (receptores coherentes)
• La implementación puede ser: analógica, digital o híbrida
• Se debe distinguir dos etapas: adquisición y seguimiento
– Pueden funcionar con parámetros diferentes
• Interferencias típicas:
– Movimiento del transmisor o receptor
• Cambios frecuencia portadora (Doppler), en fase (cambio en camino de transmisión dominante), interrupción de la recepción (fading)
– Ruido en el canal, en el emisor o en el receptor

Lo que nos permite poder recuperar la señal original con la mayor precisión y certeza para evitar al máximo la IES que estábamos teniendo antes de este bloque.

Fig.3.5 Señal sincronizada.

IMPLEMENTACION DEL BLOQUE GRAM-SCHMIDT.

Normalmente es como recibimos nuestra señal en recepción debido al desbalanceo vemos como tenemos un desfase y por lo cual una descompensación de amplitud que se muestra también evidente si a eso le sumamos el problema de IES es evidente que vamos a tener una probabilidad de error mayor.
Al apreciar en la grafica 20 en la recepción de la señal vamos a tener al momento de muestrear el error de distinguir un símbolo de otro.
La introducción del bloque de GS lo que nos permite que la señal recibida se va acoplando poco a poco gracias al sistema adaptativo a la señal que realmente queremos obtener.
Inicialmente el valor de error es mayor y poco a poco va memorando ya que la señal recibida va convergiendo hacia el valor mínimo de error, en el cual la señal que obtenemos es muy similar a la transmitida.
Y el bloque configurado está compuesto por 3 partes fundamentales:

- Señal de entrada.
- Bloque adaptativo.
- Señal de salida compensada con GS.

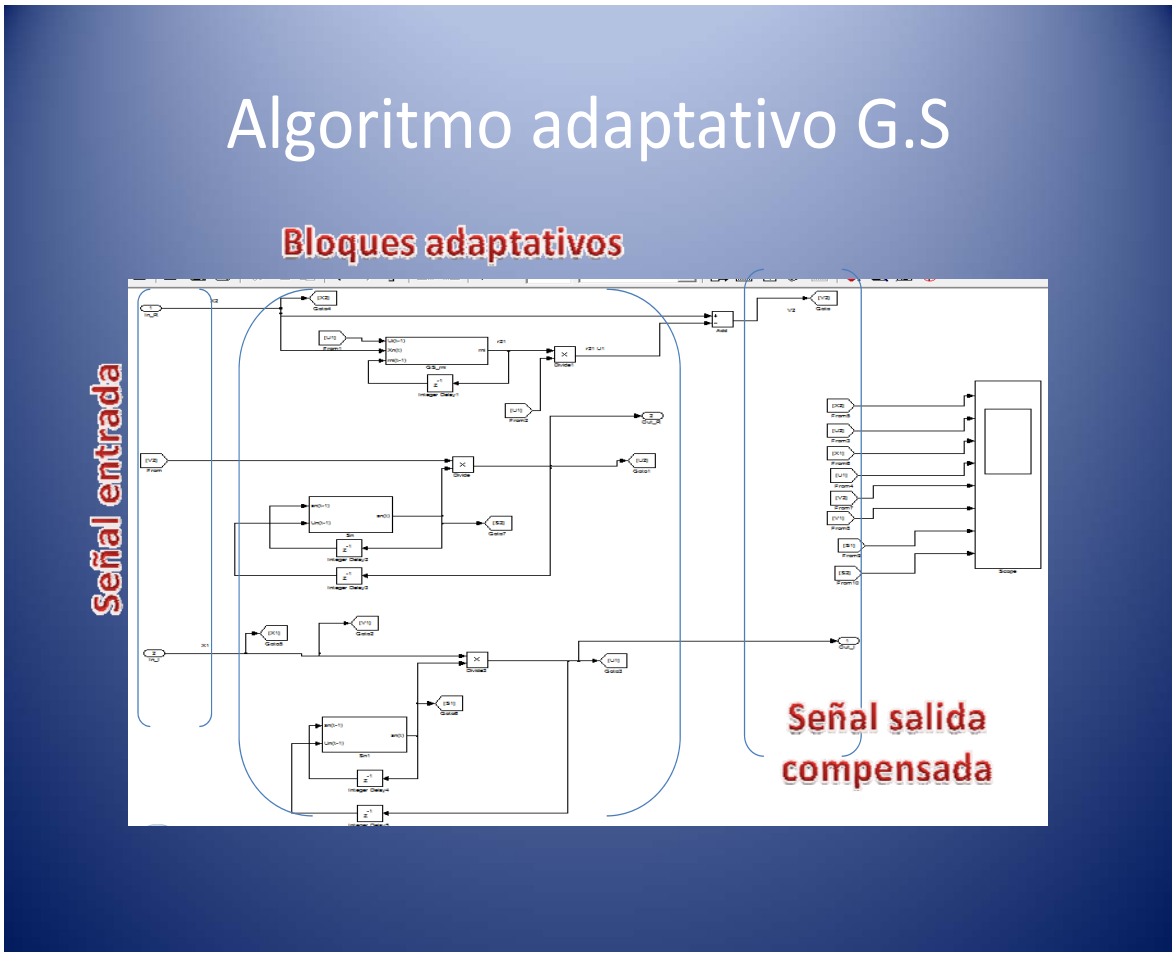

Fig.3.6 Bloque del algoritmo adaptativo de G.S.

SEÑAL RECIBIDA DESPUES DEL BLOQUE GRAM-SCHMIDT.

Como era de esperar el bloque de gran - schmidt lo que pretendía era compensar la ganancia y fase que tenia la señal debido como ya comentamos anteriormente a los diferentes elementos electrónicos o interferencia delo canal.
Como podemos observar en la fig.3.7 obtenemos los 16 símbolos transmitidos pero es verdad que es evidente que tenemos un error que es difícil de evitar pero que no es tan importante como el que teníamos antes de nuestra implementación de G.S.
Lo que se pretendía al inicio de esta implementación se ha conseguido solucionar y este algoritmo de adaptación para la compensación del desbalanceo es ahora también una buena forma de evitar estos efectos tan adversos que teníamos en los sistemas de comunicación hoy en día es evidente y la grafica lo demuestra que podremos obtener la señal que transmitimos de forma segura y podremos ser capaces de diferenciar los símbolos que hemos transmitido.

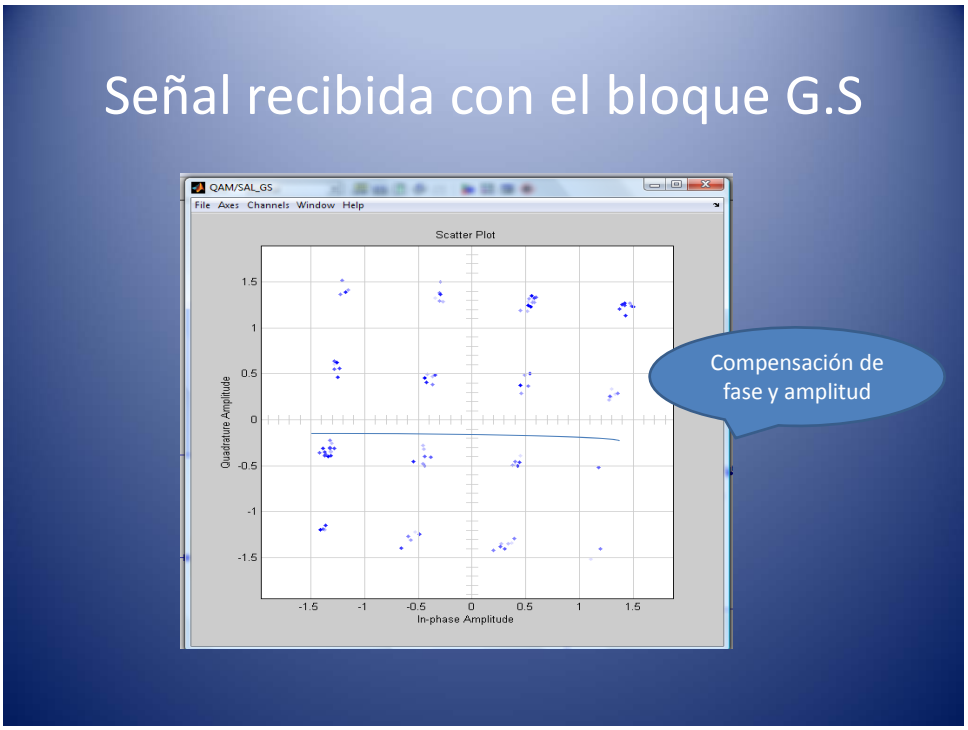

Fig.3.7 Señal obtenida después del bloque de G.S

3.1.1.2 TABLAS COMPARATIVAS DE IRR, BER, SEÑALES.

* COMPARACION ENTRE SEÑALES SIN DESBALANEO Y CON DESBALANCEO.

Una vez comprobada la funcionalidad de nuestra implementación lo que vamos hacer es una comparación de las diferentes señales que vamos obteniendo para poder diferenciar el efecto de compensar el desbalanceo.
La siguiente grafica nos muestra la señal sin desbalanceo es decir la señal que recibimos por la rama superior es decir sin error de fase ni de amplitud (azul) y la señal que si le hemos introducido el desbalanceo (rojo).
Tomando un margen de tiempo prudente para poder observar los valores de amplitud que van tomando las señales y poder distinguir el error que se va cometiendo por el hecho de utilizar el desbalanceo.

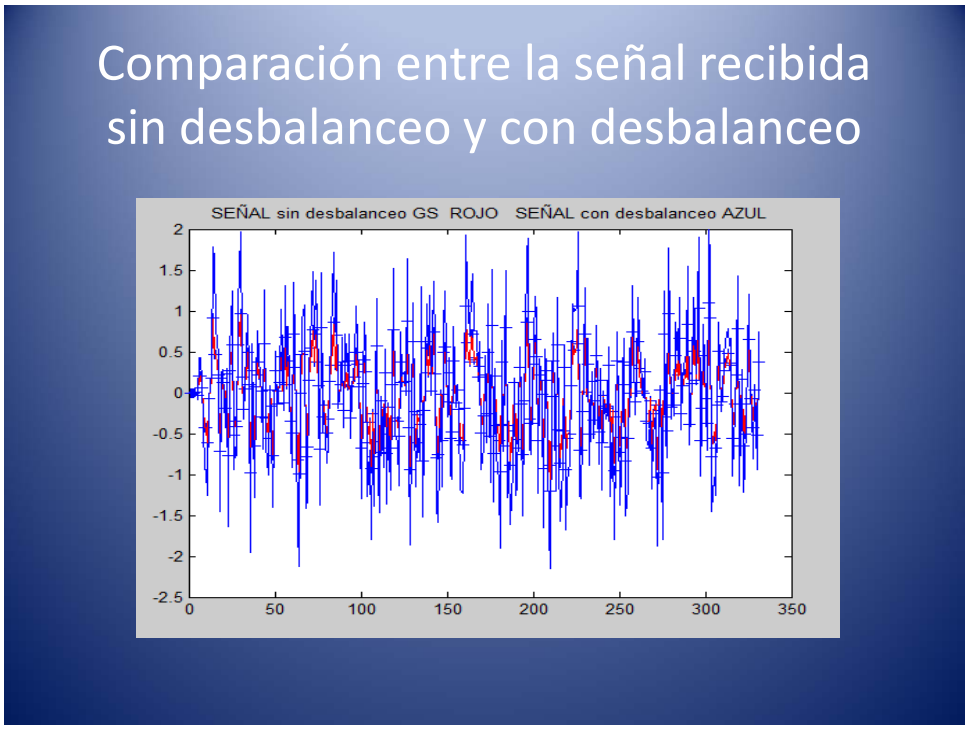

Fig.3.8 Comparación de señales con y sin desbalanceo.

*COMPARACION DE LAS SEÑALES SIN DESBALANCEO Y LA SEÑAL CON DESBALANCEO PERO UTLIZANDO G.S.

La obtención de la señal en las 2 ramas es el resultado de nuestra implementación en ella podemos ver claramente como la señal en rojo que antes aparecía en intermedio de la otra de color azul ahora se enmascara debido a que las dos señales toman valores casi idénticos de amplitud por lo cual nosotros podemos decir que el error que se comete es mínimo.
Nuestra simulación la hemos caracterizado para unas 4000 iteraciones y es fácil de observar que a medida que la señal se acerca a las 2000 iteraciones y se aleja de las 2500 es donde obtenemos el menor error posible en todos los casos casi prácticamente igual con diferentes valores de ruido en el canal.

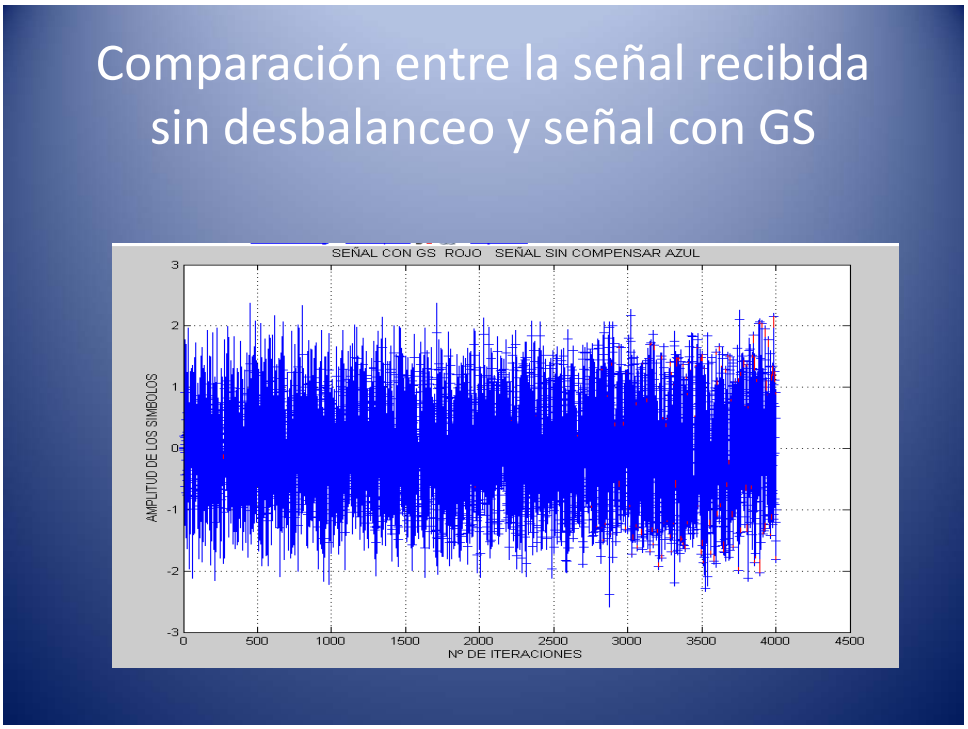

Fig.3.9 Comparación de la señal sin desbalanceo y la señal con desbalanceo y la utilización de G.S

Nuestra simulación la hemos caracterizado para unas 4000 iteraciones y es fácil de observar en la fig.3.10 que a medida que la señal se acerca a las 2000 iteraciones y se aleja de las 2500 es donde obtenemos el menor error posible en todos los casos casi prácticamente igual con diferentes valores de ruido en el canal.

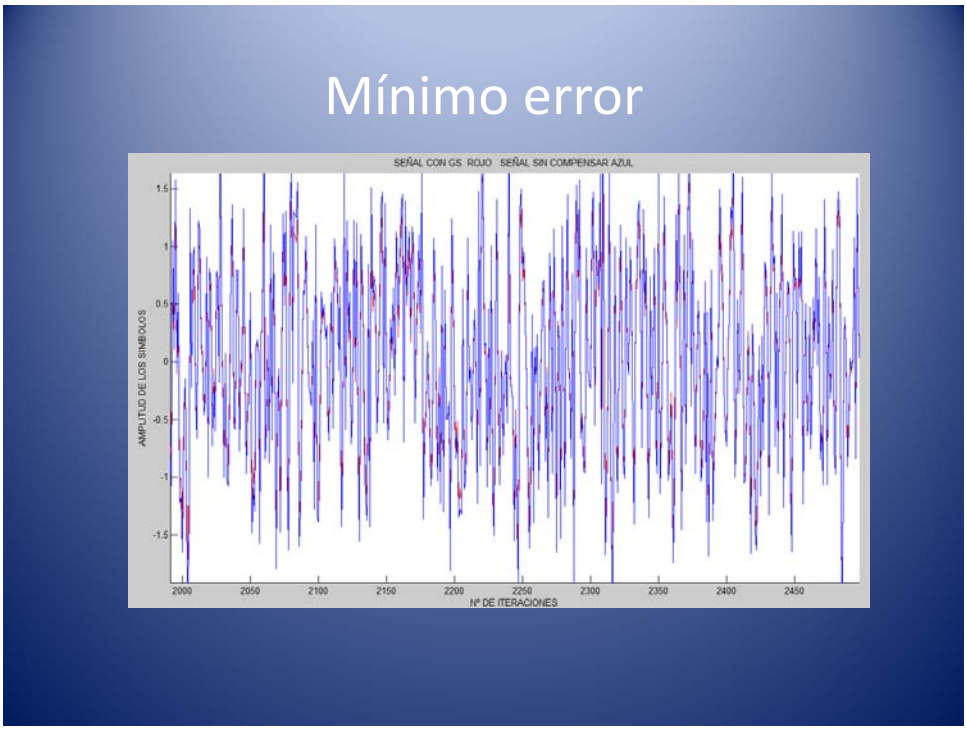

Fig.3.10 Márgenes del mínimo error.

COMPARACION DE LA SEÑAL SIN DESBALANCEO, LA SEÑAL QUE UTILIZA G.S Y LA SEÑAL TRANSMITIDA.

Para poder afirmar el efecto que tiene el bloque implementado comparamos la señal que originalmente trasmitimos con las 2 señales que recibimos en cada una de las diferentes ramas.

La señal que nos marca con magenta es la que el transmisor nos envía y tiene representada las muestras con los valores de amplitud correspondientes a los símbolos que se envíen. por otro lado la señal sin desbalanceo que está representada con azul y la señal que utiliza el bloque de G.S que la matizamos en rojo.

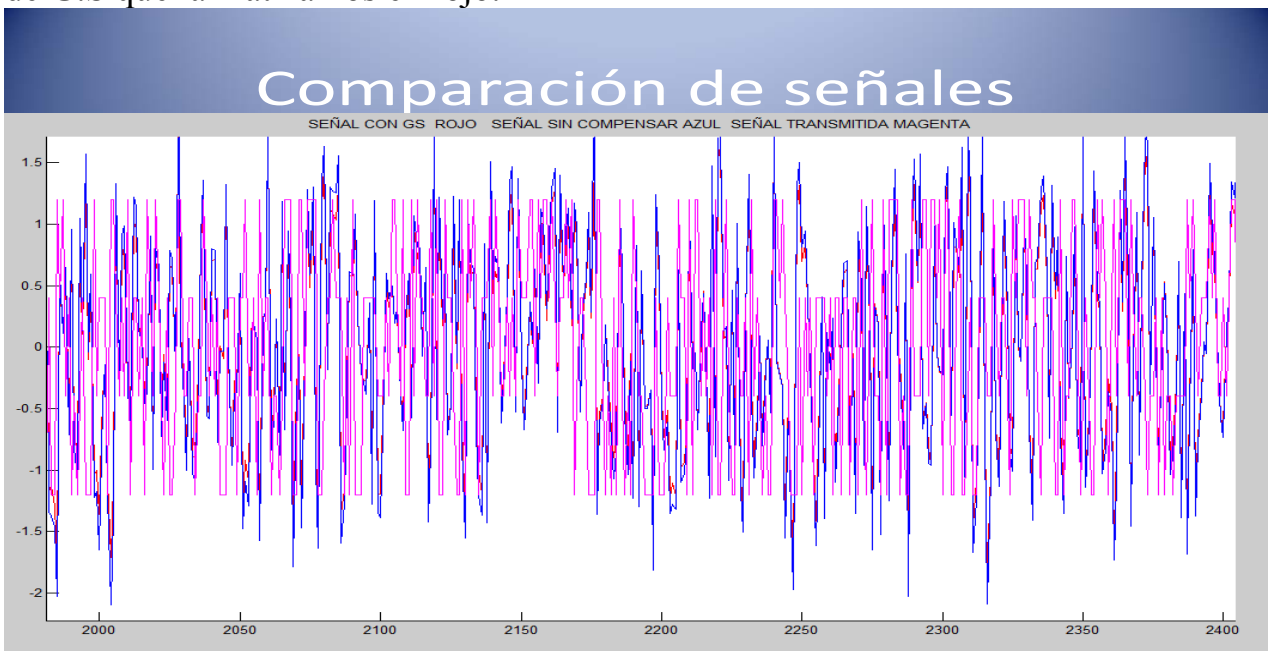

Fig.3.11 Comparación general.

Para poder apreciar tomamos los valores en el cual el algoritmo implementado es el más eficiente y lo comparamos con las señales antes mencionadas nos damos cuenta que el valor o margen de error entre señales es el mínimo y que realmente estamos logrando recibir la señal en el receptor con la mejor calidad y representación posible.

Como se observa mejoramos la rama que ni siquiera tiene desbalanceo es decir nuestro modelo se acerca mucho más a la señal original.

*CALCULOS DEL ERROR CUADRATICO MEDIO IRR

El propósito de esta prueba es calcular el error (IRR) del nivel del sistema propuesto desarrollado y establecer posibles soluciones para compensar de mejor manera el desbalanceo producido. Para la prueba se utilizo las comparaciones de las señales recibidas.

```
inter = mean([real(Senal_sd - Senal_GS)].^2);
```

El valor inter es la relación entre la señal sin desbalanceo (Senal_sd) y la señal que utiliza el bloque G.S (Senal_GS).
Finalmente el valor del error que se obtiene en esta implementación y con estos valores de simulación es:

IRR=10*log10 (inter./(mean((real(Senal_GS)).^2)))

IRR =-10.4431
Como comentamos antes este valor está sujeto al número de interacciones que hagamos, como también al valor de ρ que es el valor que especifica la rápida o lenta que es la convergencia de la señal adaptada y del ruido del canal (EbNo).

*REPRESENTACION DE LA SEÑAL ERROR (IRR)
Mostramos las fluctuaciones y los valores de error que toma la en esta podemos observar la variación que toma y entorno a que muestra se realiza.

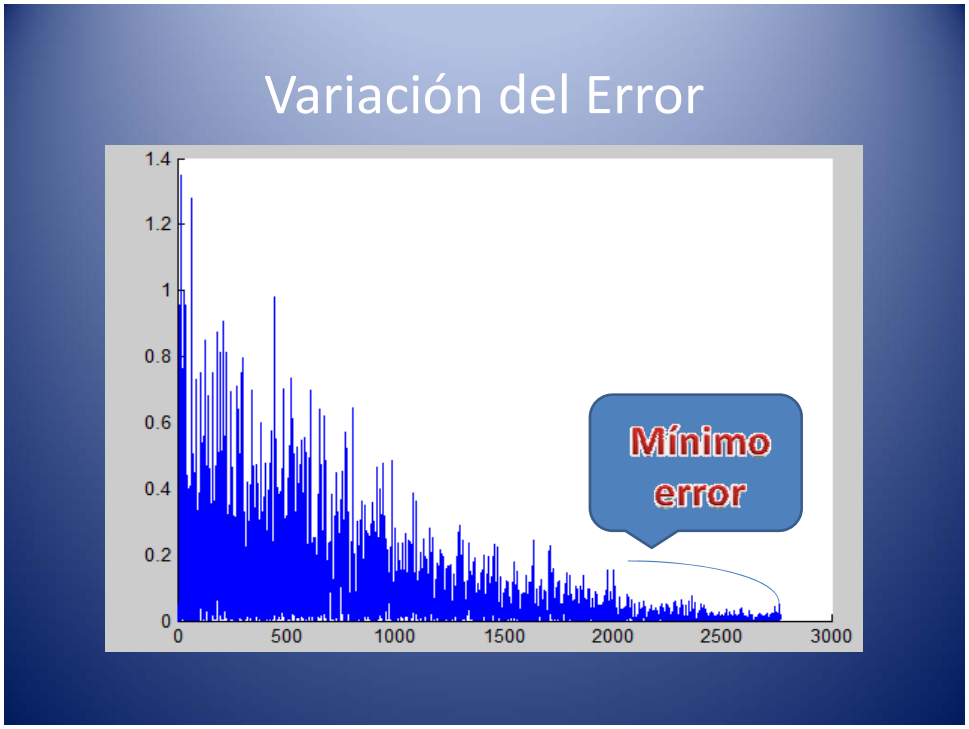

Fig.3.12 Representación del error IRR.
Al ser un algoritmo adaptativo se observa como conforme aumenta el número de iteraciones el valor del error va reduciendo acercándose en algunos instantes a ser casi nulo, pero nunca se llega a anular por completo el factor negativo es que pasado un numero de iteraciones el valor del error empieza otra vez a tener valores mucho mayores que en nuestra implementación ya no resultan de utilidad.

*OBTENCION DE LA SEÑAL BER

Para el cálculo del valor que nos da la tasa de error por bit BER nos valemos de la implementación de display que nos indiquen todas las tasas que vemos a continuación. Este valor lo tomamos en el caso de la señal recibida sin desbalanceo

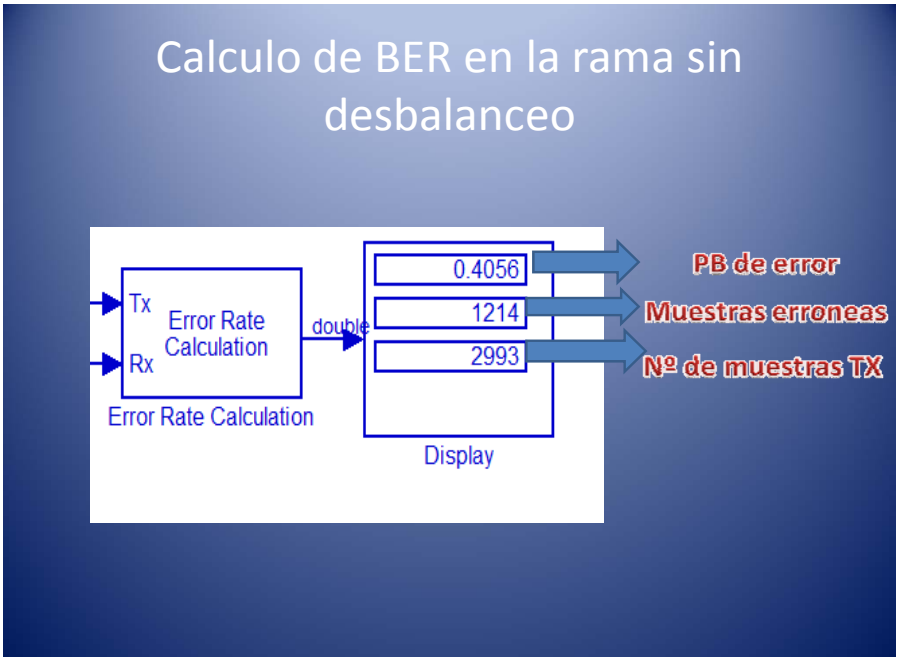

Fig.3.13 Calculo de BER en recepción sin desbalanceo.

Para la tasa en la con desbalanceo procedemos de igual manera y Obtenemos los resultados que presentamos en la grafica.

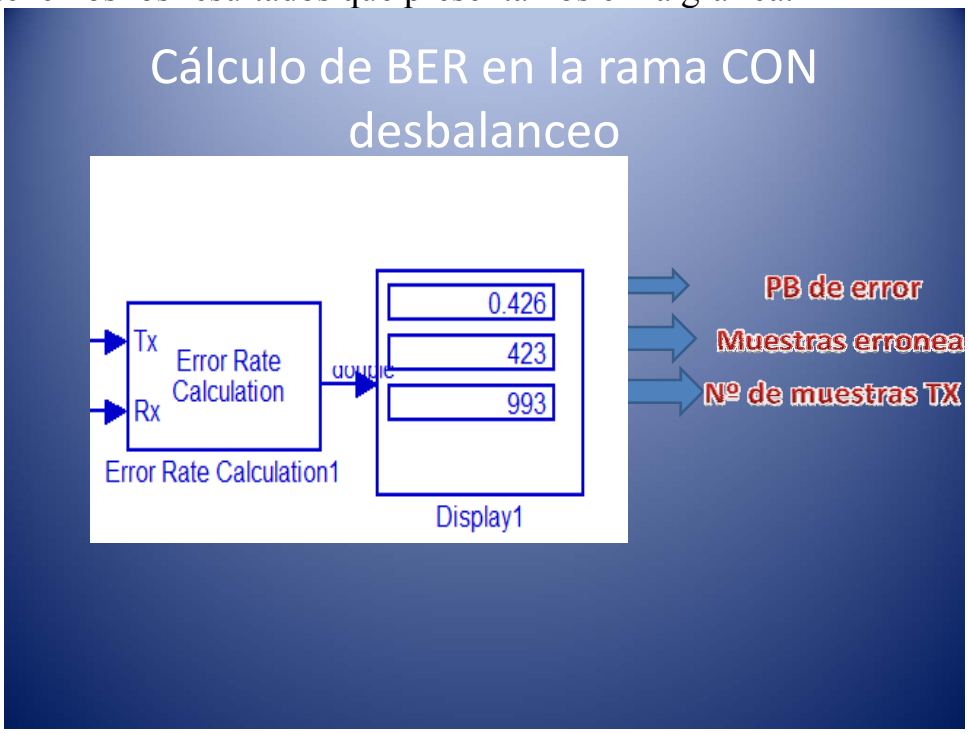

Fig.3.14 Cálculo de ber con desbalanceo.

VALORES QUE AFECTA A LA IMPLEMENTACION.

Como ya mencionamos nosotros simulamos con valores que mejoran nuestro sistema de comunicaciones pero para comprobar que si cambiamos los valores de: EbNo o de ρ los valores cambian e incluso las graficas como vemos a continuación:

Nosotros hemos desarrollado nuestro modelo con valores de:

ρ =0.001 ideal

Ahora cambiamos por el de

ρ =0.01

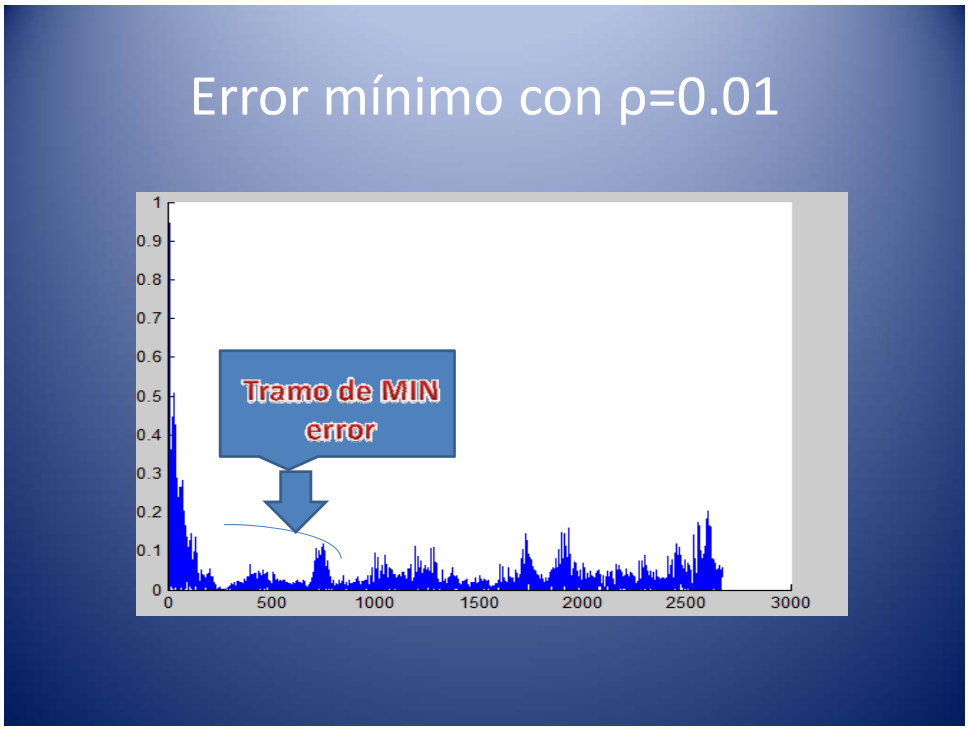

Fig.3.15 Grafica del error con valor de p=0.01.

El valor afecta en lo rápido que se adapta ahora para llegar a encontrar el mínimo, es verdad que los saltos son más grandes y que ahora las iteraciones necesarias son menos como vemos en la figura.

Nuestro sistema de esta manera se comporta como igualmente como un sistema adaptativo y tiene la misma característica que antes es decir mejora el error y también compensa el desbalanceo, pero con la peculiaridad que lo hace más rápido y por el contrario como ya dijimos antes que luego de algunas iteraciones el sistema vuelve a empeorar y el valor del error vuelve a ser grande esto no puede a la postre muy útil para sistemas que necesiten un tiempo de procesado mayor debido a que todo empeora como es el caso de la probabilidad de error , la interferencia en la señal de salida en el valor de IRR(erro cuadrático medio),etc. Como vemos en la grafica:

IRR = -16.5023

Para nuestro caso empeora el sistema que deseamos diseñar.

El otro valor que tenemos que tener en cuenta es EbNo que es el ruido del canal nosotros asumíamos en nuestro caso un ruido de EbNo=1000, ahora probamos con EbNo=10 y observamos como varia considerablemente ya que al añadir más ruido la interferencia que tenemos y que afecta a mi señal en recepción es mucho mayor que antes y por lo cual empeora todos los valores que considerábamos aceptables como es el caso de la probabilidad de error a la hora de diferenciar un símbolo con otro, el valor de error (IRR) ,etc.

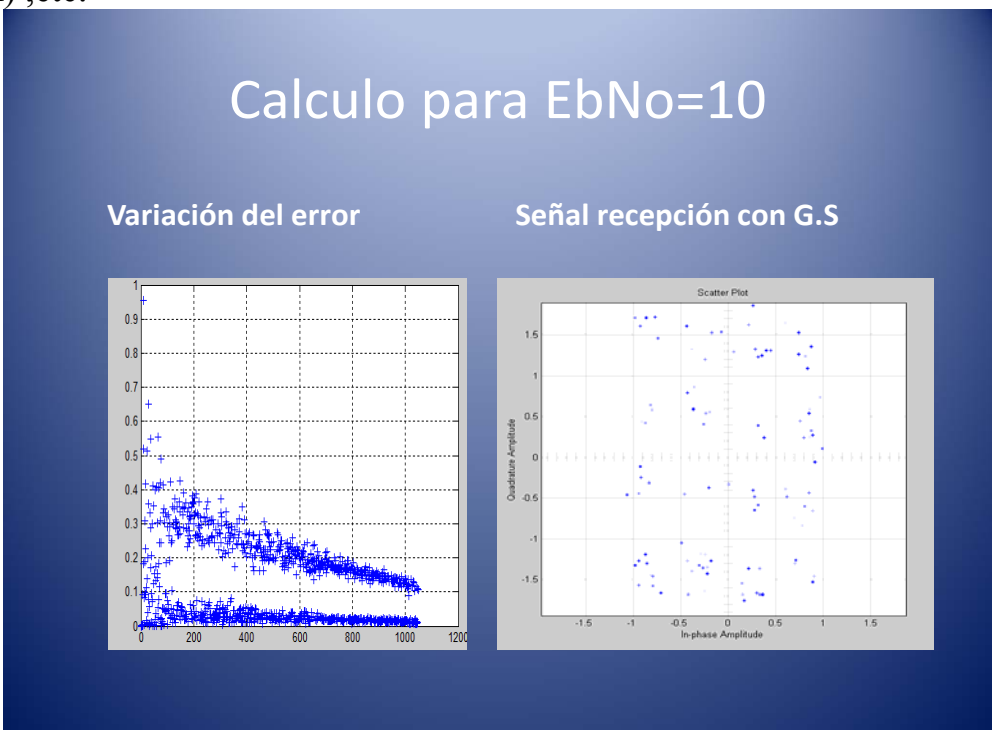

Fig.3.16 Grafica con aumento de ruido EbNo=10.

Como conclusión los valores son los que directamente pueden variar la implementación de nuestro sistema de comunicaciones por lo cual debemos estudiar lo que necesitamos implementar y luego elegir los datos y valores adecuados para obtener el mejor resultado posible.

CAPITULO_4

4.1 IMPLEMENTACION EN SYSTEM GENERATOR.

El hardware utilizado fue la plataforma de evaluación Virtex 4 ML401. Esta es una plataforma de desarrollo que provee acceso a los recursos del dispositivo FPGA Virtex-4 LX25 que se encuentra en la tarjeta. Tiene dos puertos para reloj (osciladores de 100 MHz), memoria SDRAM, SRAM, flash y EEPROM, un display de LCD, y varios conectores y puertos que trabajan en conjunto con el FPGA. El módulo diseñado e implementado en este dispositivo puede ser implementado en otra tarjeta con FPGA con facilidad ya que se debería tomar en cuenta los requerimientos mínimos del módulo en cuanto a limitaciones del hardware.

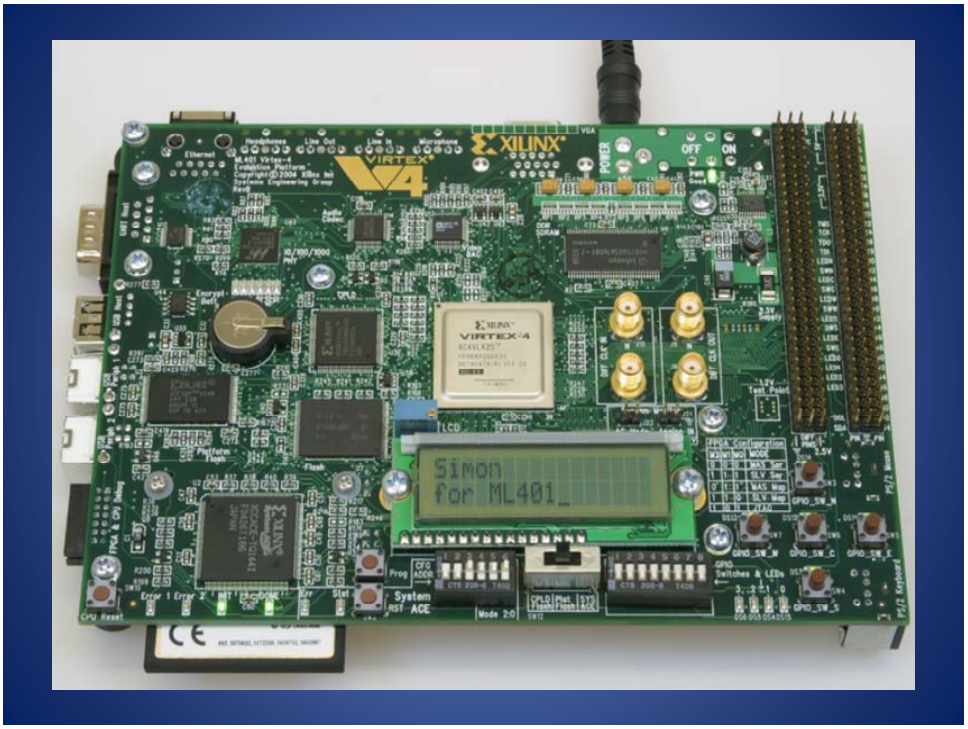

Fig.4.1 Plataforma de evaluación virtex 4.

Para la implementación en hardware se utiliza el programa Xilinx System Generator y Xilinx AccelDSP, ambos de Xilinx ISE Design Suite 10.1, en conjunto con Simulink de Matlab. Éste último fue el ambiente en donde se realizaron las pruebas. Se generaron los datos y luego se analizaron mediante el uso de bloques específicos de Simulink. El programa Xilinx System Generator es una herramienta de diseño que facilita el uso del entorno de diseño basado en modelo de Simulink de The Mathworks en el diseño de FPGAs. Los diseños se crearon empleando bloques específicos de Xilinx dentro del ambiente de Simulink. Los pasos de implementación,

incluyendo síntesis y posición y ruta se ejecutaron automáticamente para generar un archivo de programación de FPGA.

Modelos algorítmicos de Matlab pueden ser incorporados a System Generator mediante el uso de AccelDSP. Este programa tiene una gran capacidad de síntesis de algoritmos, tal que puede generar un modelo completamente planificado de punto fijo a base de un código de punto flotante de Matlab. Algunas características de este programa incluyen: conversión de punto flotante a punto fijo, exploración de diseño y programación algorítmica.

Fue necesario determinar ciertos parámetros funcionales tales como la tasa de datos, el número, tipo y precisión de bits de las entradas, etc.

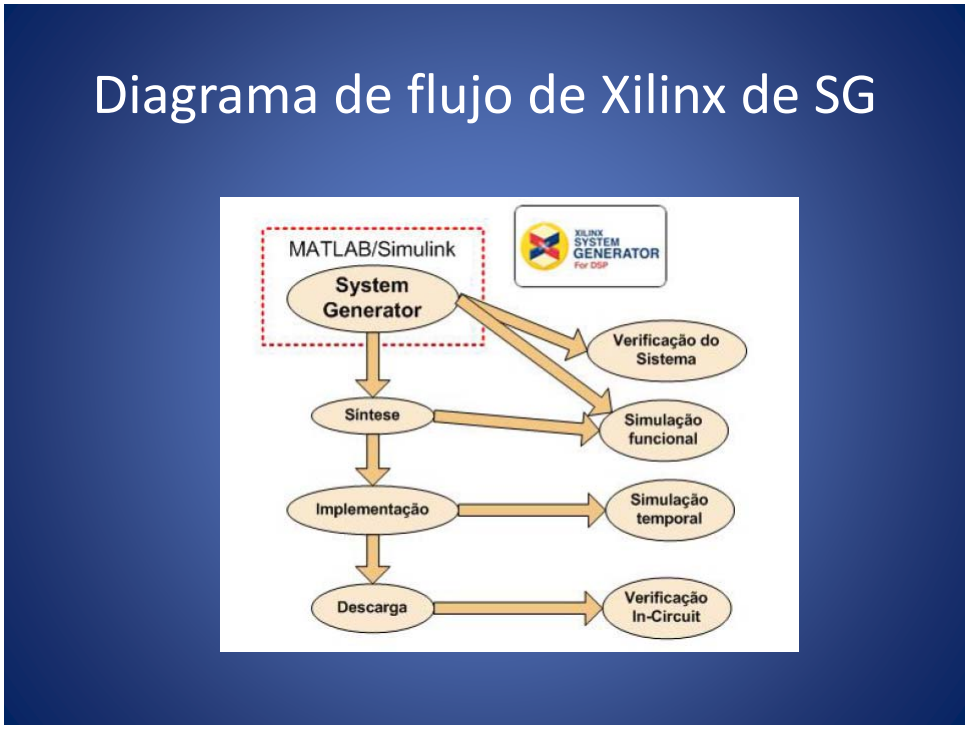

Fig.4.2 Diagrama de flujo de XILINX.

SIMULINK (blockset) es una biblioteca de bloques que están conectados con el editor de bloques de System Generator creando modelos funcionales de sistemas dinámicos.

Esos bloques ofrecen abstracción de funciones matemáticas, lógicas, de memoria DSP (Digital Signal Processing), que pueden ser usadas para construir sofisticados sistemas de procesamiento de señal y de otros tipos. Existen también bloques que forman interfaces graficas para otras herramientas de software (por ejemplo, FDATool y ModelSim), o bien como una generación automática de código de lenguaje de programación VHDL (VHSIC Hardware Description Languaje, VHSIC-Very High Speed Integrated Circuit).

La tabla 1.1 muestra el conjunto de bibliotecas de bloques básicos Xilinx System Generator y la tabla 1.2 muestra un conjunto de bloques de referencia Xilinx que se usan en la composición de bloques básicos de System Generator.

CONJUNTO DE BLOQUES DE XILINX	
Biblioteca	**Descripción**
Elementos Básicos	Bloques de formación para lógica digital
Comunicación	FEC (Forward Error Correction)o bloques moduladores comúnmente usados en sistemas de comunicación digital
Lógica de control	Bloque para control de circuitos de estado de máquina.
Tipos de datos	Bloques que convierten tipos de datos
DSP	Bloque de procesamiento digital de señales
Matemática	Bloques que implementan funciones matematicas
Memoria	Bloques que implementan memorias
Memoria compartida	Bloques que implementan el acceso memoria compartida Xilinx
Herramientas	Generación de código(bloque System Generator)recursos de estimación .HDL(Hardware Description Language) o simulación .etc

Tabla 1 .- Conjunto de bloques Xilinx.

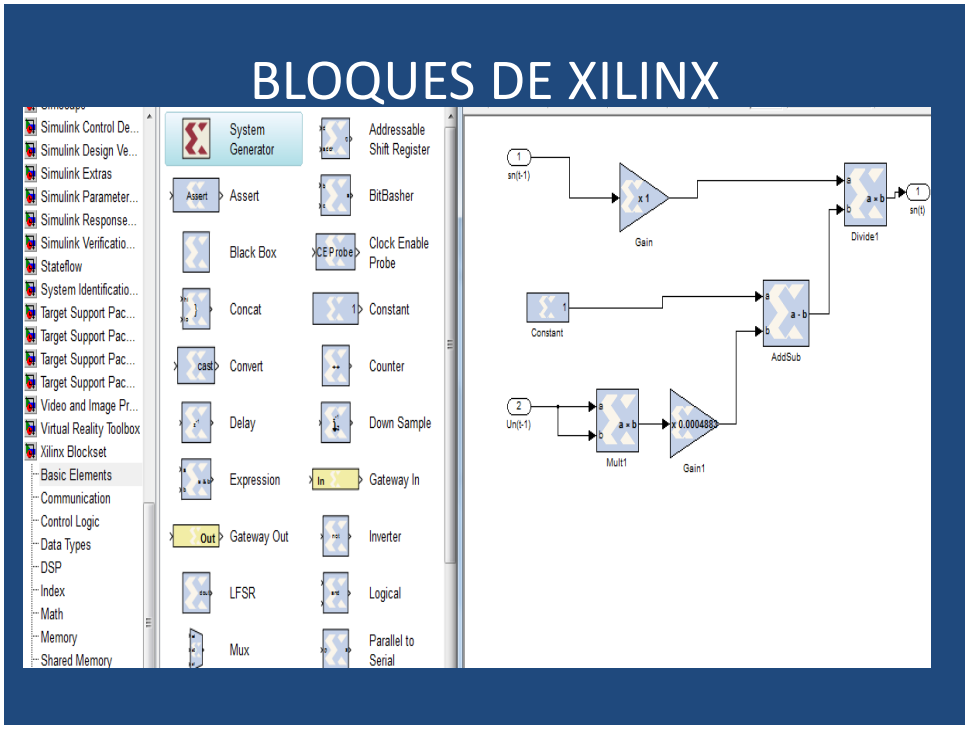

Fig.4.3 Diagrama de bloques de XILINX.

CONJUNTO DE BLOQUES DE REFERENCIA XILINX	
BIBLIOTECA	DESCRIPCION
Comunicación	Bloques comúnmente usados en sistemas de comunicación digital
Lógica de Control	Bloques para el control de circuitos y estado de máquina.
DSP	Bloques de procesamiento digital de señales
Imagen	Bloques de procesamiento de imagen
Matemática	Bloques de implementación de funciones matemáticas

Tabla 1.2.- bloques de referencia Xilinx.

Los conexión entre bloques Xilinx y bloques que no pertenecen a la biblioteca de Xilix .

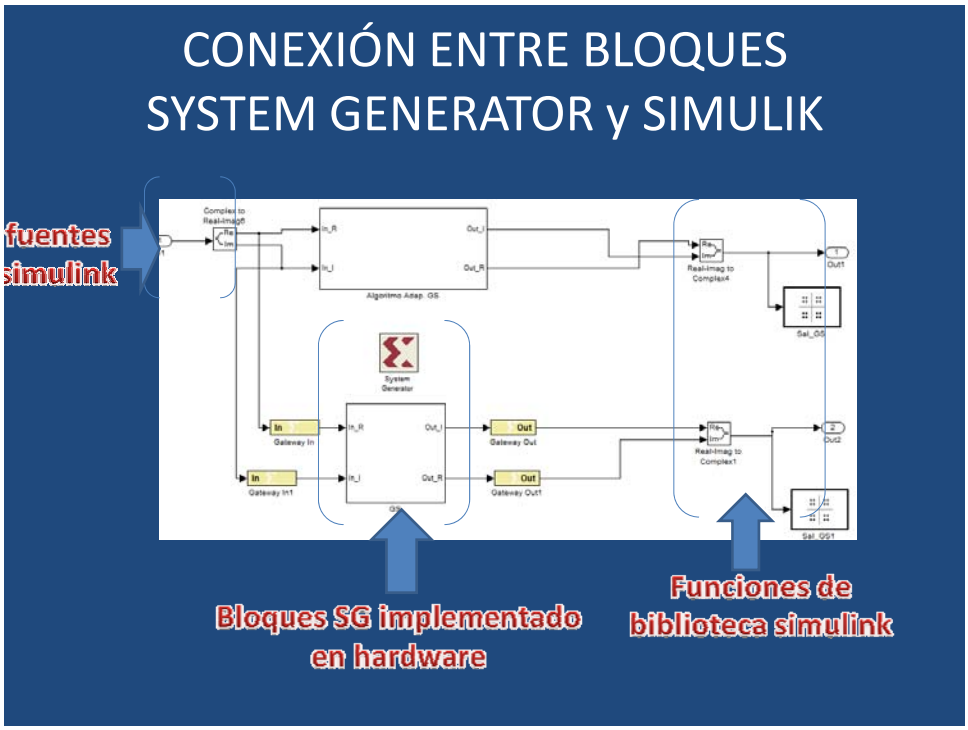

Fig.4.4 Diagrama de conexión entre bloques S.G y Simulink.

Un bloque Gateway In convierte una señal de doble presicion en una señal binaria que puede ser interpretado por el conjunto de bloques de la biblioteca Xilinx y un bloque gateway Out que realiza el proceso inverso.Son capaces de determinar un tipo apropiado de salida basado en su tipo de entrada. La mayoría de bloques trabaja con dos tipos de precisión:

Precisión completa: System Generator escoge un tipo de salida para garantizar la precisión necesaria para representar un valor.

Precisión definida por el usuario: permite al usuario especificar el tipo de salida, como lo mostramos en la figura ,, en la cual definimos la precisión en función de dos parámetros **cuantificación y overflow.**
La cuantificación tiene dos posibilidades redondeo y truncamiento (wrap).
La opción de overflow tiene la posibilidad de saturación, truncamiento y generación de error cuando ocurre un overfolw.

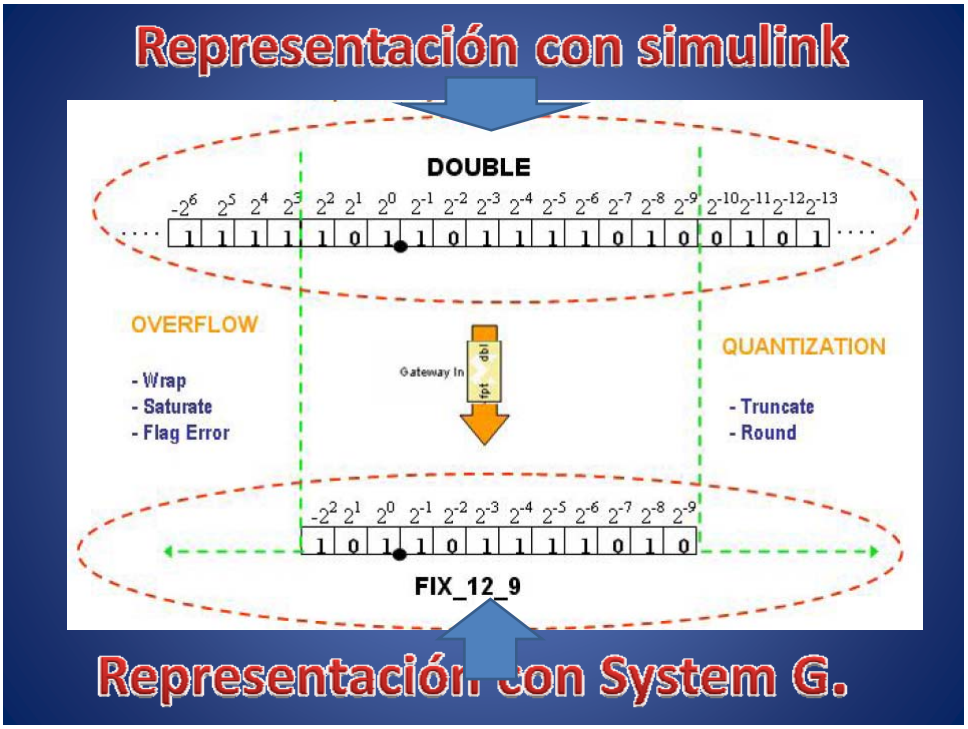

Fig 4.5 Representación de la señal en simulink (Representación en punto flotante) y en SG(Representación en punto fijo)

4.1.1 SIMULACION EN HARWARE.

System generator produce un bloque de simulación en harware asociado a un bitstream(configuración de FPGA que está disponible para poder funcionar con la plataforma de harware de FPGA).
A partir de ahora vamos a utilizar para la implementación del proyecto en system generator.
En el grafico podemos observar los bloques que conforman un Modulador-Demodulador de QAM igual que el utilizado en Simulink con la única diferencia de que ahora la implementación esta realizada en System Generator y se diferencia con un tono de color diferente cada etapa que compone el sistema .

- Muestreo
- Interpolación
- Desbalanceo y Desbalanceo
- Sincronización
- Diezmado

- Representación , etc.

SIMULACION DEL BLOQUE DE G.S

Tenemos varias etapas de nuestro bloque de GS en cada una de las cuales se han utilizado diversos componentes de System Generator para la implementación de nuestro sistema.

Cada uno de los bloques necesita una forma diferente de configuración, en los cuales tenemos en cuenta cosas como la frecuencia de muestreo, el periodo, el valor de inicialización, muestras por segundo, etc.

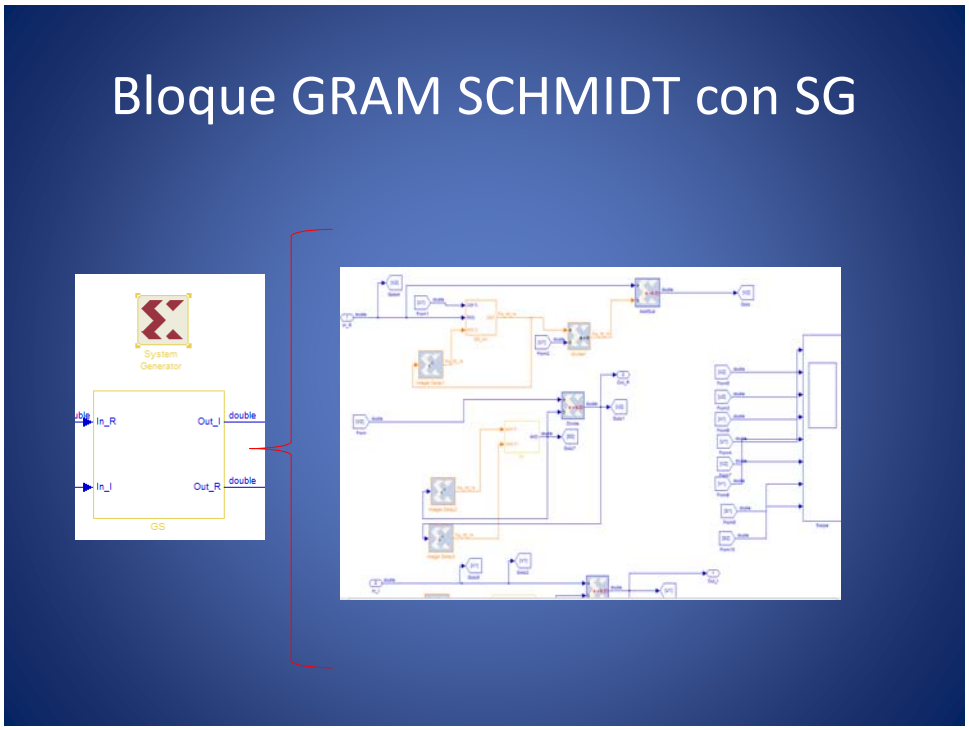

Figura 4.6 Representación de G S con System Generator.

Los principales bloques utilizados en esta implementación son:

Multiplicadores: Todos los multiplicadores se han configurado del siguiente modo :

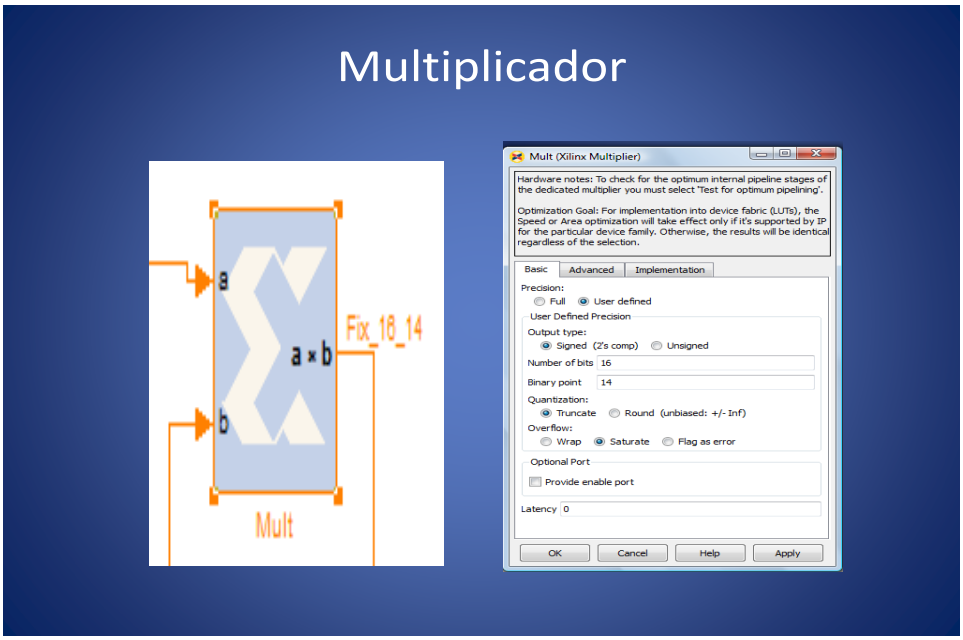

Fig 4.7 Representación del multiplicador G S con System Generator.

- output type = Signed (2's comp)
- Number of bits = 16
- Binary point = 14
- Latency = 0
- Quantization = truncate
- Overflow = satúrate
- Used embedded multipliers
- Optimize for = area

Los multiplicadores que utilizamos en nuestro diseño están implementados como se observa en la grafica utilizando la lógica definida que nos da el fabricante, además trabajamos con lógica con signo, utilizamos 16 bits 14 de ellos fraccionarios ya que es lo necesario para que la representación sea la correcta. Usamos truncado, saturación y con latencia 0 ya que no existe retardo.

AddSub's: Todos los AddSub's se han configurado de la siguiente manera:

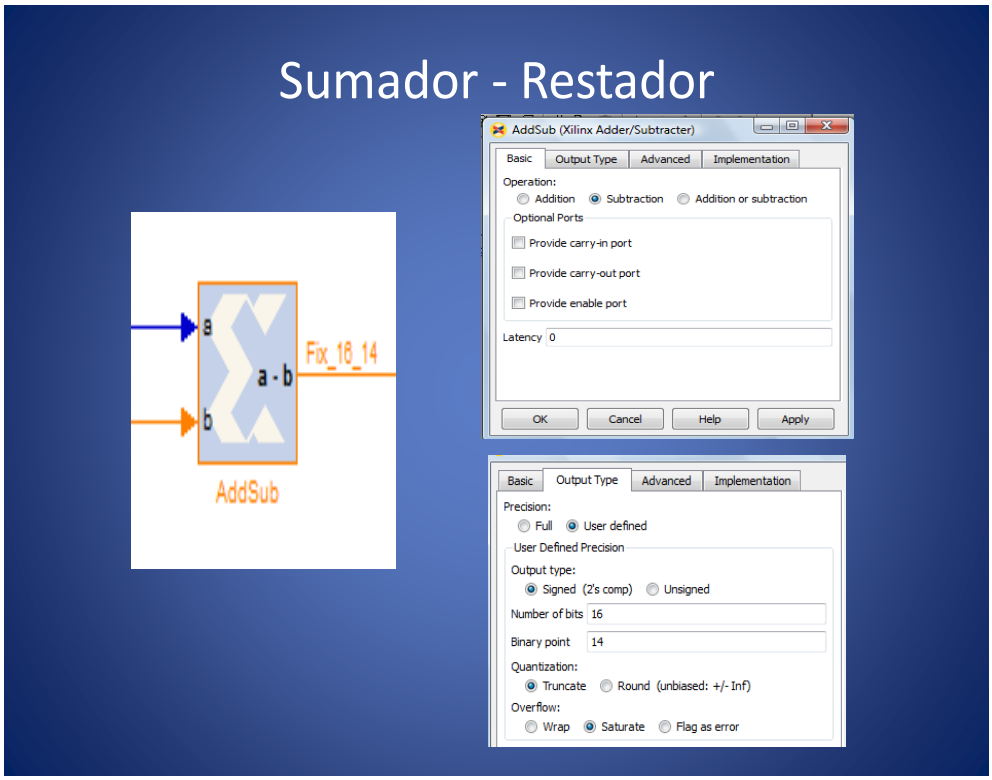

Fig 4.8 Representación de sumadores/restador con System Generator.

Los sumadores restadores según nuestra necesidad están configurados de forma que su salida tenga 16 bits 14 de ellos fraccionarios de igual forma que se trabaje con signo, igualmente utilizamos truncado, saturación y sin retardo es decir latencia 0.

- Output type = Signed (2's comp)
- Number of bits = 16
- Binary point = 14
- Latency = 0
- Quantization = truncate
- Overflow = satúrate

Divide: Todos los Divide's se han configurado de la siguiente manera:

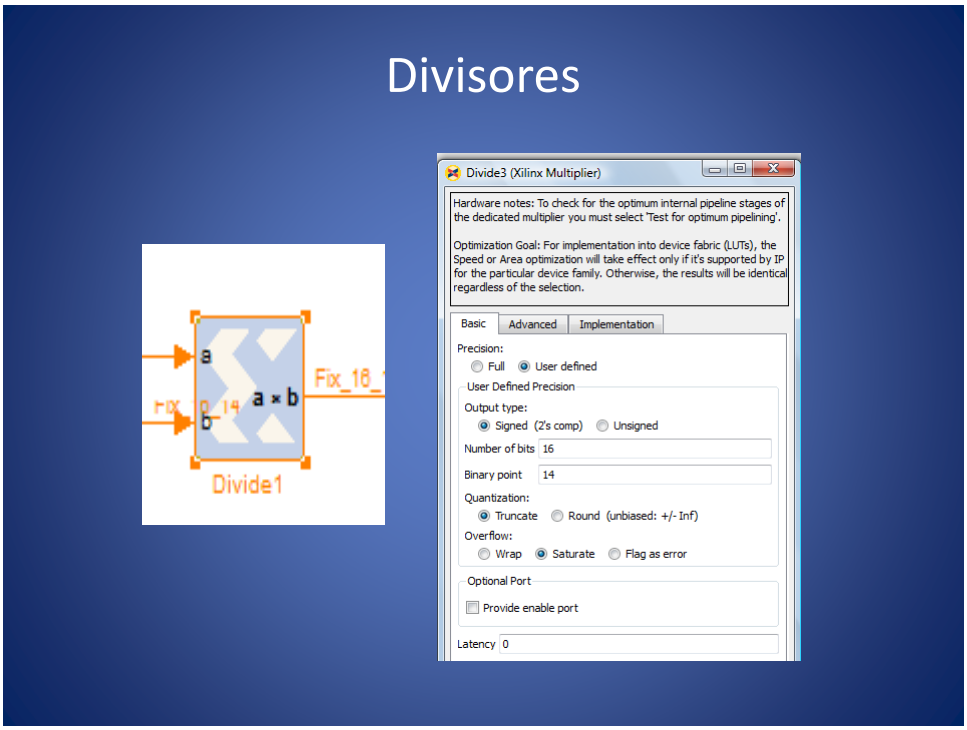

Fig 4.9 Representación de divide con System Generator.

Al igual que los multiplicadores los divisores están configurados con las mismas características que los multiplicadores.

- output type = Signed (2's comp)
- Number of bits = 16
- Binary point = 14
- Latency = 0
- Quantization = truncate
- Overflow = satúrate
- Used embedded multipliers
- Optimize for = area

Registros: Todos los Registro's se han configurado de la siguiente manera:

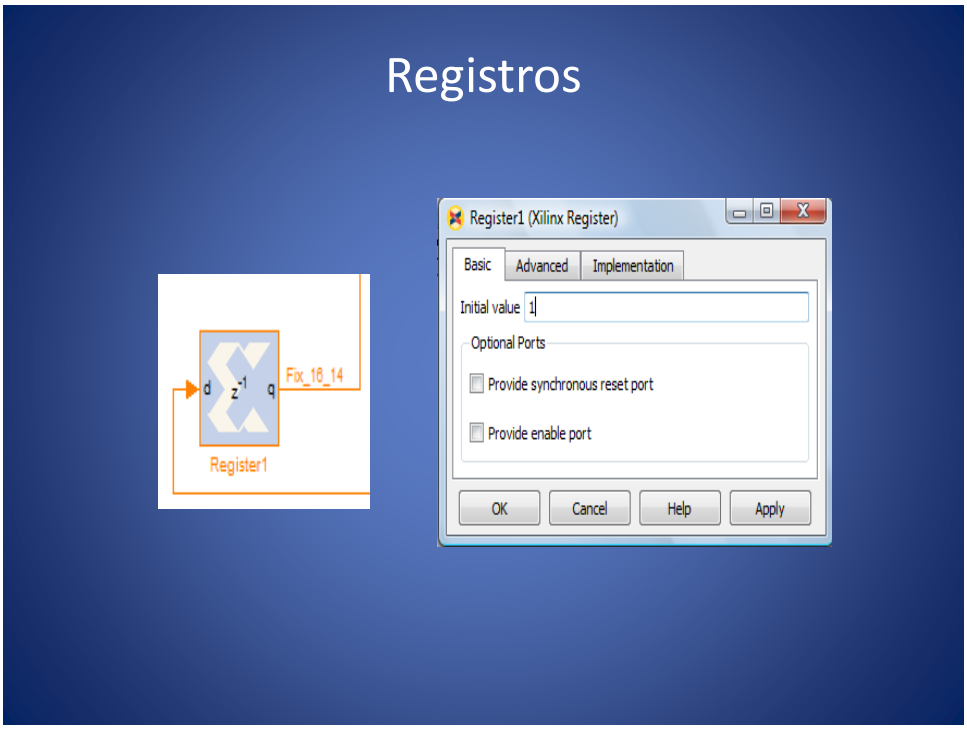

Fig 4.10 Representación de registros con System Generator.

Lo que vamos a tener con esta etapa es poder ir registrando las entradas.
El valor inicial que le otorgamos es 1.

Retardos: Todos los Retardo's se han configurado de la siguiente manera:

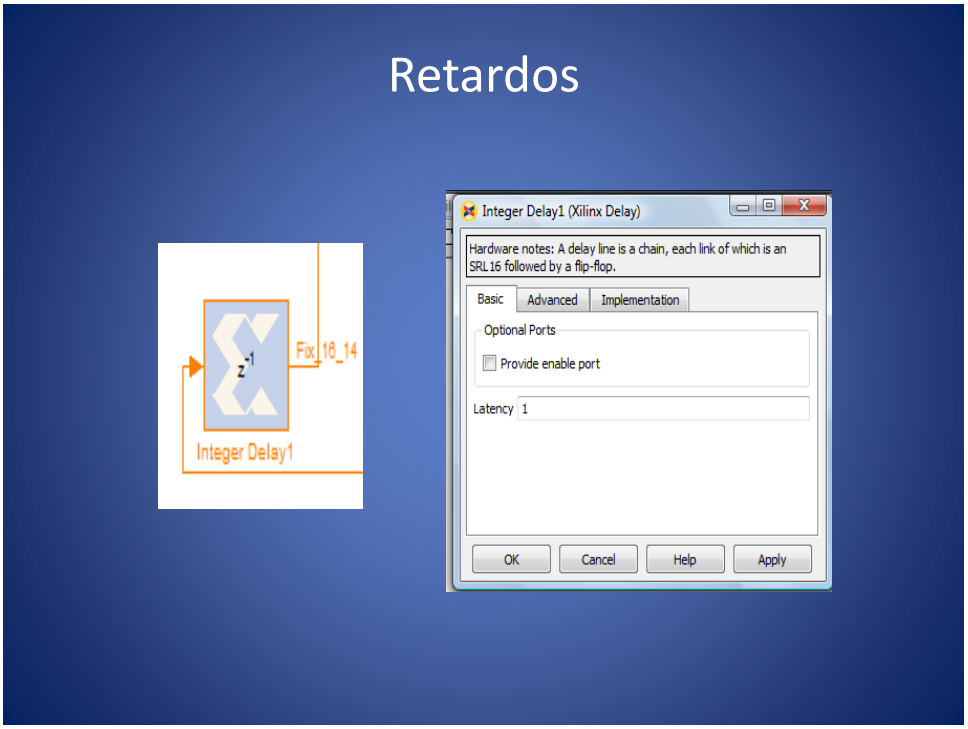

Fig 4.11 Representación de retardos con System Generator.

- Latency = 1

Ganancias: Todos los ganancia's se han configurado de la siguiente manera:

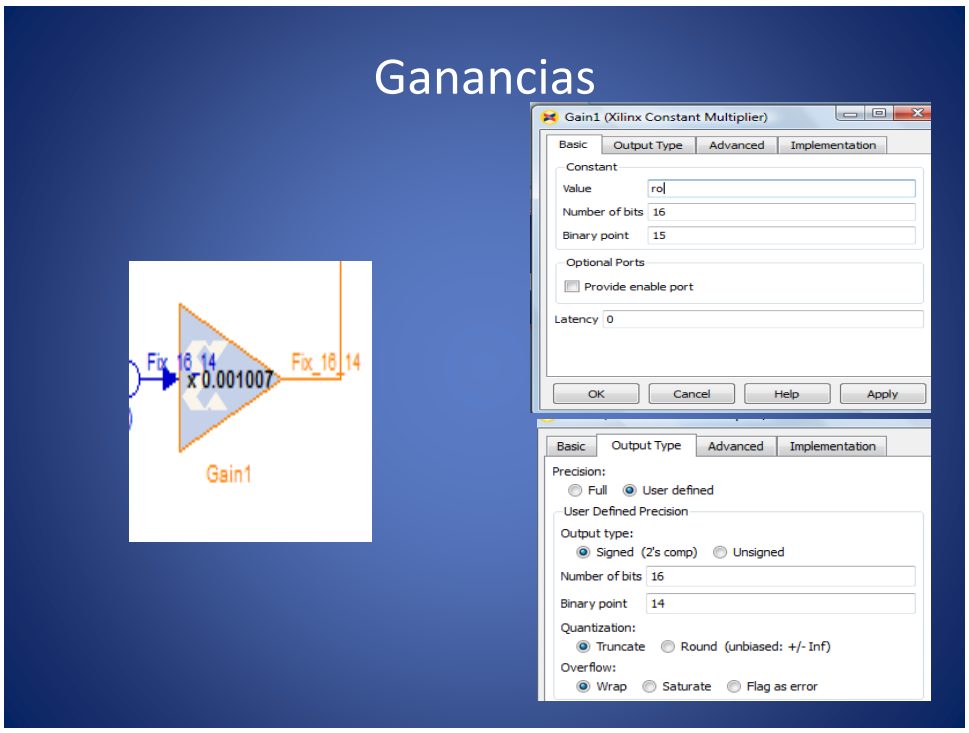

Fig 4.12 Representación de ganancias con System Generator.

- value = ro que tiene el valor 0.001
- output type = Signed (2's comp)
- Number of bits = 16
- Binary point = 15
- Latency = 0
- Quantization = truncate
- Overflow = Wrap

Bloque System Generator :

A continuación se muestra el detalle de como se ha configurado este bloque Teniendo en cuenta los valores que tenemos determinados al inicio como son la frecuencia de muestreo la virtex que vamos a necesitar ,etc.

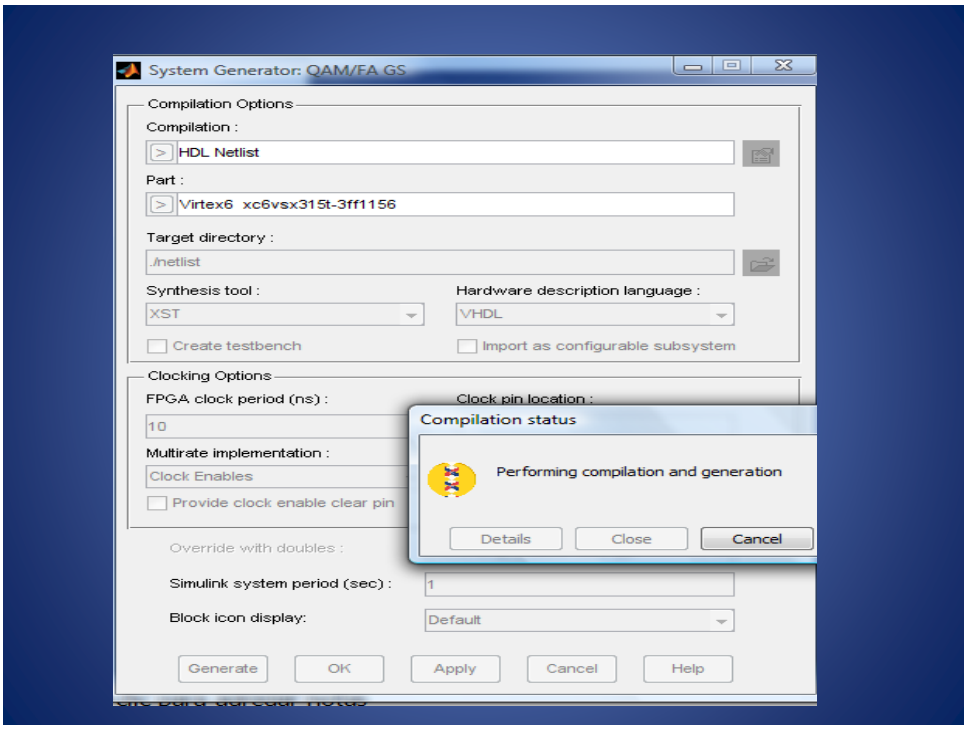

Fig 4.13 Configuración de G S con System Generator.

Una vez que hemos terminado de implementar nuestro diseño con System Generator podemos comprobar que estamos obteniendo lo que se esperaba es decir la compensación de la señal después de pasar por el modulo de G.S, por lo tanto recogemos muestras de la señal con desbalaceo y observamos que es la misma que la señal sin desbalanceo .

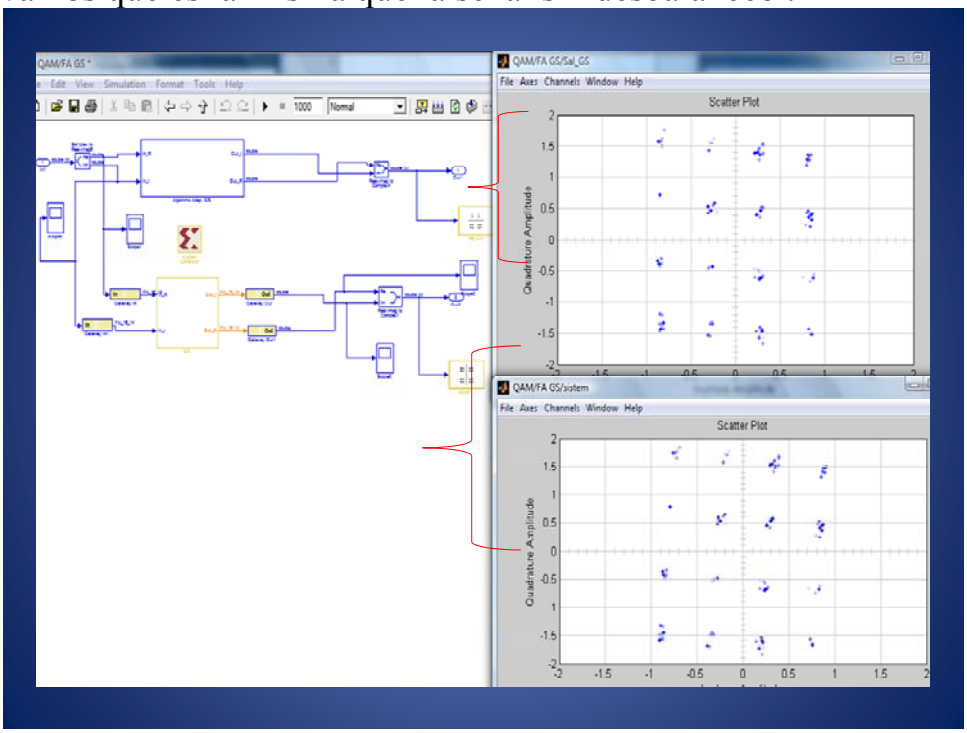

Fig 4.14 Comparación de las señales de salida.

Resultados Implementacion:

A la vista de estos resultados, se concluye que el circuito funciona perfectamente en simulación, por lo que se procede a la generación del archivo bitstream con el que se programará la FPGA de usuario. Para ello,primero hay que generar el código VHDL a partir del diseño en System Generator, pulsando en el botón generate en la ventana mostrada en lafig.4.4. Una vez generado el código VHDL se sintetiza y se implementa mediante el software project navigator, para finalmente crear el archivo bitstream. En este punto podemos ver :

- Synthesis report,
- Los recursos de la FPGA que van a ser utilizados,
- Timming report, el mínimo período de trabajo:

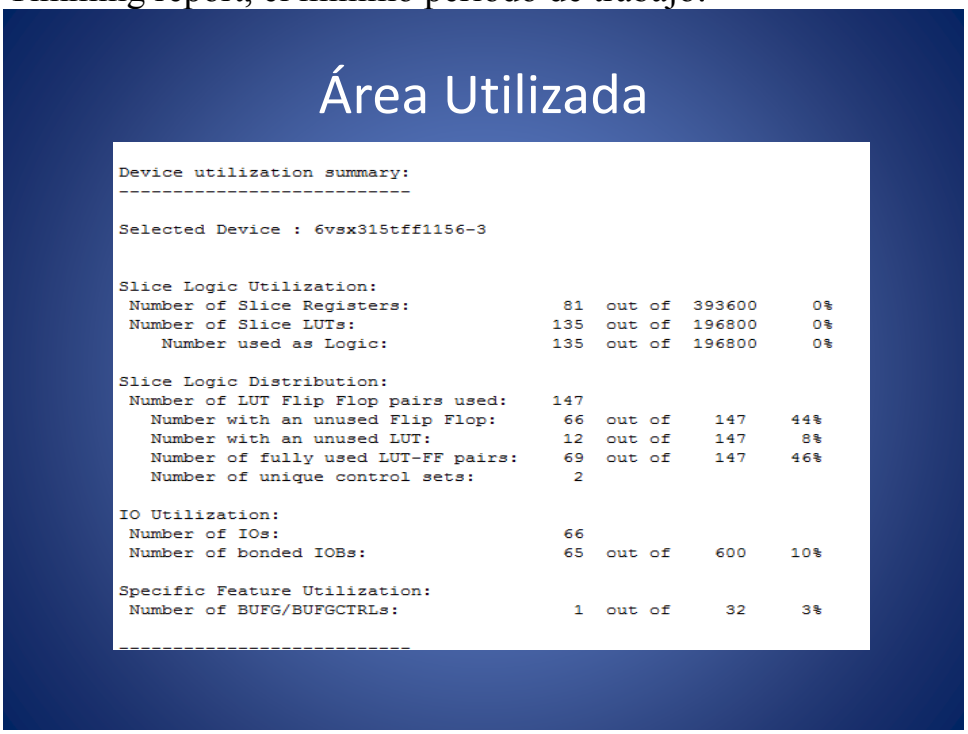

Figura 4.15 Synthesis Report.

El informe obtenido corresponde con el área que va a ser utilizada en nuestra implementación.

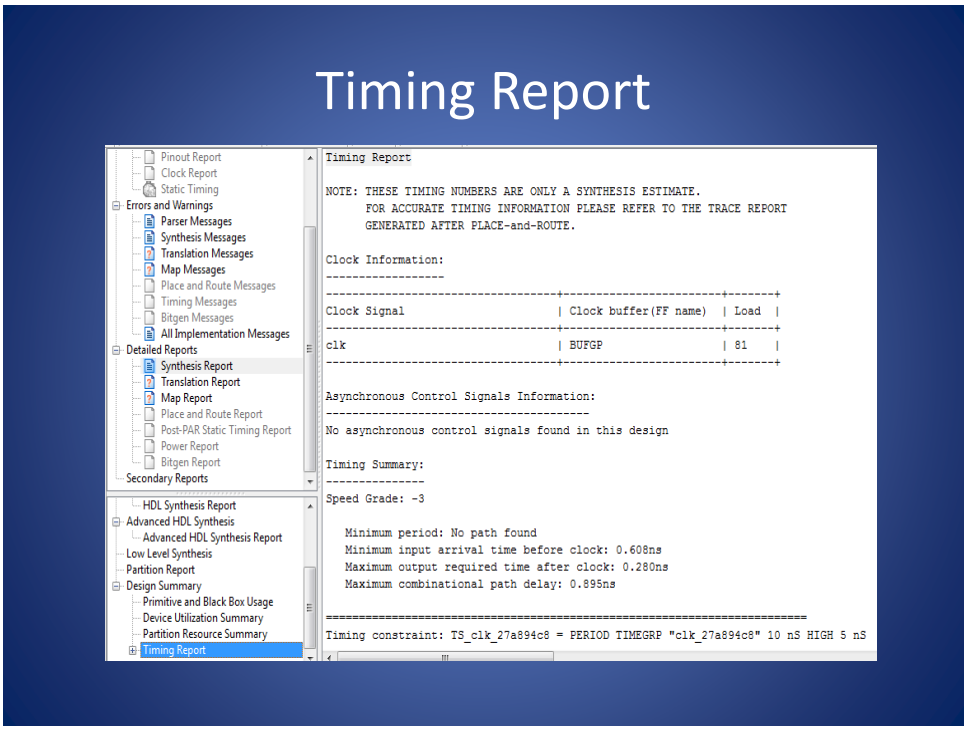

Fig 4.16 Timing Report.

En el timming report obtenemos la frecuencia máxima de trabajo, f = 1 / (0.28 ns) = 3.57GHz. El hecho de que no coincida con la frecuencia de reloj del sistema, se debe a los retardos de propagación de las señales en la FPGA.

CONCLUSIONES:

En este proyecto hemos descrito las características más importantes de System Generator, herramienta no instalada en el conjunto de Matlab, y que permite realizar una co-simulacion en el nivel de harware.

Nuestro proyecto ha consistido en el desarrollo de un compensador de error debido al desfase , para lo cual hemos desarrollado un bloque al que le llamamos Gram Schimdt el cual es un sistema adaptativo el cual lo hemos implementado en System Generator y posteriormente en la FPGA para la simulación. El objetivo de este proyecto era proporcionar al lector unos fundamentos básicos sobre las modulaciones y demodulaciones en QAM.

También hemos comentado los problemas ocasionados por los errores de fase y la forma en que afectan a las señales, hemos descrita de forma sencilla la implementación y su diseño en SG en el cual hemos comentado bloque a bloque todo el diseño que se ha realizado y a la vez damos a conocer los resultados obtenidos tanto de forma teórica como grafica para demostrar el resultado positivo de nuestro diseño.

REFERENCIAS BIBLIOGRAFICAS.

Apuntes y prácticas de Electrónica de Comunicaciones Digitales.

Apuntes y prácticas de Diseño Micro electrónico Digital.

[1]HAYKIN,S. Adaptative Filter Theory.4.ed. Upper Saddle River N.J :Prentice-Hall,2002.1

Edward A. Lee, David G. Messerschmitt
Digital Communication
Segunda edicion . KAP, 1994. (Ch. 16.- Carrier Recovery)

Steven A. Tretter
Communicaction System design Using DSP Algorithms. With Laboratory Experiments for the TMS320C30

Teoria y aplicaciones de comunicacion.
Series Editor: R.W. Lucky, Bellcore. Plenum Press. NY. (1995)
Simon Haykin

Comicaciones digitales
Wiley, 1988 (John G. Proakis *Digital Communications*
McGraw-Hill, 3º Ed. 1995

Ho, Sam, "Adaptive Modulation (QPSK, QAM)," Intel Application Note.

Tratamiento de señal de tiempo discreto
Alan V. Oppenheim, Ronald W. Schafer
Ed. Prentice-Hall

Tratamiento Digital de la Señal. Teoría y aplicaciones
Antonio Albiol, Valery Naranjo, Josep Prades; Ed. UPV

Problemas de tratamiento digital de la señal
Antonio Albiol ... [et al.], Ed. UPV

Theory andDesign ofAdaptive Filters
J. Treichler, C.R. Johnson, M. Larimore; Ed. Prentice Hall

Digital Signal Processing with examples in MATLAB
S. Stearns; Ed. CRC Press

Adaptive Signal Processing
B. Widrow, S. Stearns; Ed. Prentice Hall